Universität Leipzig
Institut für Informatik
Winter 2026/2027

Prof. Dr. Karin Quaas

Exercises

Verification for Infinite-State Systems

Solutions for Exercise Sheet 1 (for 22.10.2026)

Exercise 1.1

(a) Can you write a program that, for every input pushdown automaton $\mathcal{A}_S$ and finite automaton $\mathcal{A}_\varphi$, decides whether every word accepted by $\mathcal{A}_S$ is also accepted by $\mathcal{A}_\varphi$?

(b) Can you write a program that, for every input finite automaton $\mathcal{A}_S$ and pushdown automaton $\mathcal{A}_\varphi$, decides whether every word accepted by $\mathcal{A}_S$ is also accepted by $\mathcal{A}_\varphi$?

SOLUTION: (a) Yes. Construct finite automaton $\mathcal{A}_\varphi^c$ (regular languages are closed under complement), construct product automaton of $\mathcal{A}_S$ and $\mathcal{A}_\varphi^c$ (contextfree languages are closed under intersection with regular languages), decide whether the language accepted by the product automaton is empty (emptiness problem for pushdown automata is decidable). If it is empty, then every word accepted by $\mathcal{A}_S$ is also accepted by $\mathcal{A}_\varphi$, otherwise it is not the case.

(b) No. We reduce from the undecidable language universality problem for pushdown automata: given a pushdown automaton $\mathcal{A}$ over Σ, does $L(\mathcal{A})$ accept every word over Σ? So let $\mathcal{A}$ be a pushdown automaton over Σ. Define a finite automaton $\mathcal{A}_S$ that accepts every word over Σ. Then we have $\mathcal{A}$ accepts every word over Σ iff $L(\mathcal{A}_S) \subseteq L(\mathcal{A})$. Hence $L(\mathcal{A}_S) \subseteq L(\mathcal{A})$ is undecidable, too.

Exercise 1.2

Let $\mathcal{R} = \{r_1, \ldots, r_4\}$. Define $\nu_1 = \nu_\perp$, $\nu_2 = \nu_1[\{r_1, r_2\} \leftarrow 4]$, and $\nu_3 = \nu_2[\{r_1, r_4\} \leftarrow 8]$.

(a) Give a register test ϕ such that $\nu_1, 5 \models \phi$. We have $\nu_1 = (\perp, \perp, \perp, \perp), \nu_2 = (4, 4, \perp, \perp), \nu_3 = (8, 4, \perp, 8)$. For example, $\phi = r_1 \neq \mathbf{d}$.

(b) Give a datum $d \in \mathbb{N}$ such that $\nu_2, d \models r_2 = \mathbf{d} \wedge r_3 \neq \mathbf{d}$. $d = 4$.

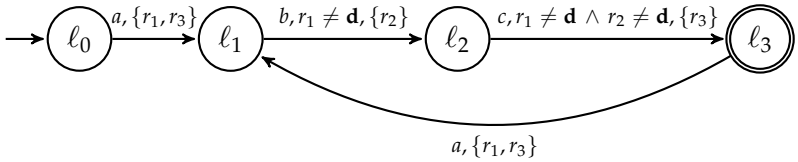
(c) Give a register test ϕ such that $\nu_3, 8 \models \phi$ and ϕ both uses an equality and an inequality test. For example, $r_2 \neq \mathbf{d} \wedge r_3 = \mathbf{d}$.

(d) Give a datum $d \in \mathbb{N}$ such that $\nu_3, d \models r_1 = \mathbf{d}$. $d = 8$.

(e) Give a register test ϕ such that $\nu_3, d \not\models \phi$ for all $d \in \mathbb{N}$. For example $\phi = r_1 = \mathbf{d} \wedge r_2 = \mathbf{d}$.

Exercise 1.3

Consider the register automaton depicted below:



(a) Give a data word w such that $|w| = 6$ and $w \in L(\mathcal{A})$. (Prove that $w \in L(\mathcal{A})$!) $(a,4)(b,2)(c,1)(a,4)(b,1)(c,3)$. For a proof, we need to give an initialized accepting run:

$$(\ell_0, \bot, \bot, \bot) \xrightarrow{a,4} (\ell_1, 4, \bot, \bot) \xrightarrow{b,2} (\ell_2, 4, 2, 4) \xrightarrow{c,1} (\ell_3, 4, 2, 1) \xrightarrow{a,4} (\ell_1, 4, 2, 4) \xrightarrow{b,1} (\ell_2, 4, 1, 4) \xrightarrow{c,3} (\ell_3, 4, 1,$$

(b) Give data values $d_1, d_2, d_3 \in \mathbb{N}$ such that the data word $(a, d_1)(b, d_2)(c, d_3)$ is not in $L(\mathcal{A})$. $d_1 = d_2 = d_3$, for all $d_1 \in \mathbb{N}$

(c) Define $L(\mathcal{A})$. $L(\mathcal{A}) = \{((a, d^1)(b, e^1)(c, f^1) \dots (a, d^n)(b, e^n)(d, f^n) \mid n \geq 1, d^j \neq e^j, f^j \neq e^j, f^j \neq d^j$ for all $1 \leq j \leq n\}$, that is, d^j, e^j, f^j are pairwise distinct for all $1 \leq j \leq n$

Exercise 1.4

Prove that the satisfiability problem for register tests is in **P**.

SOLUTION: The satisfiability problem is the problem to decide whether a given register test ϕ over $\mathcal{R} = \{r_1, \dots, r_n\}$ is satisfiable, that is, whether there exist a valuation $\nu : \mathcal{R} \to \mathbb{N}_\bot$ and $d \in \mathbb{N}$ such that $\nu, d \models \phi$. Note ϕ is of the form $\psi_1 \wedge \psi_2 \wedge \dots \wedge \psi_k$, where ψ_i is an atomic register test of the form `true`, $r = \mathbf{d}$, or $r \neq \mathbf{d}$ for some $r \in \mathcal{R}$.

Claim: The following two statements are equivalent.

1. ϕ is not satisfiable.
2. There exist $r \in \mathcal{R}$ and $1 \leq i, j \leq k$ such that ψ_i is of the form $r = \mathbf{d}$ and ψ_j is of the form $r \neq \mathbf{d}$.

For the direction from 2. to 1., suppose 2. Towards contradiction, suppose that 1. does not hold, that is, ϕ is satisfiable. Then there exist $\nu : \mathcal{R} \to \mathbb{N}_\bot$ and $d \in \mathbb{N}$ such that $\nu, d \models \phi$. Hence $\nu, d \models \psi_i \wedge \psi_j$. Hence $\nu(r) = d$ and $\nu(r) \neq d$, contradiction. Hence 1. must hold.

For the direction from 1. to 2., we prove the contraposition, that is, we assume that 2. does not hold and prove that 1. does not hold. So suppose that for all registers

$r \in \mathcal{R}$ and $1 \leq i \leq k$, if ψ_i is of the form $r = \mathbf{d}$ then ψ_j is not of the form $r \neq \mathbf{d}$ for all $1 \leq j \leq k$ with $i \neq j$. We can hence partition the set of registers into two sets: $\mathcal{R}_{=}\{r \in \mathcal{R} \mid r = \mathbf{d} \text{ occurs in } \phi\}$ and $\mathcal{R}_{\neq} = \mathcal{R} \setminus \mathcal{R}_{=}$. We define a valuation $\nu : \mathcal{R} \to \mathbb{N}$ and datum $d \in \mathbb{N}$ such that $\nu, d \models \phi$. The datum d can be chosen arbitrarily, e.g. $d = 1$. We define

$$\nu(r) = \begin{cases} 1 & \text{if } r \in \mathcal{R}_{=} \\ 0 & \text{otherwise} \end{cases}.$$

It is easy to see that $\nu, d \models \phi$.

Our algorithm exploits the preceding claim and simply checks whether ϕ contains two such contradicting atomic register tests: if yes, it returns NO; otherwise it returns YES. Here is the algorithm:

1. For all $1 \leq i \leq k$
2. If ψ_i is `true`, then check the next atomic register test ψ_{i+1}
3. If ψ_i is $r = \mathbf{d}$ for some r, then check whether there exists $i < j \leq k$ such that $\psi_j = r \neq \mathbf{d}$. If yes, return NO, otherwise check the next atomic register test ψ_{i+1}.
4. If ψ_i is $r \neq \mathbf{d}$ for some r, then check whether there exists $i < j \leq k$ such that $\psi_j = r = \mathbf{d}$. If yes, return NO, otherwise check the next atomic register test ψ_{i+1}.
5. If all atomic register tests are checked, then return YES.

This can be done in time polynomial in k.

Exercise 1.5

Prove the following statement.

Let $\mathcal{A} = (Q, \Sigma, Q_{\text{in}}, \delta, Q_{\text{acc}})$ be a finite automaton. The following two statements are equivalent.

(a) There exists some initialized accepting run of $\mathcal{A}$.
(b) There exists some initialized accepting run of $\mathcal{A}$ of length[1] strictly smaller than the number of states in Q.

SOLUTION: The direction from (b) to (a) is trivial. So let us prove (a) to (b). Suppose there exists some initialized accepting run of $\mathcal{A}$ of the form $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \ldots \xrightarrow{a_k} q_k$. Let n be the number of states in Q. If $k < n$, then we are done. Suppose $k \geq n$. By the pigeonhole principle, there must exist $0 \leq i < j \leq k$ such that $q_i = q_j$. Then $q_0 \xrightarrow{a_1} \ldots \xrightarrow{a_i} q_i \xrightarrow{a_{j+1}} \ldots \xrightarrow{a_k} q_k$ is an initialized accepting run of length strictly greater than k. Repeat this argument until finally the length is strictly smaller than n.

[1] By the length of a run we mean the number of participating transitions.