Applied Automata Theory (WS 2012/2013) Technische Universität Kaiserslautern

Exercise Sheet 1

Jun.-Prof. Roland Meyer, Reiner Hüchting, Georgel Călin Due: Tue, Oct 23

Exercise 1.1 Language Shuffle as Intersection

(a) Consider NFAs A_1 and A_2 over alphabet Σ. Construct an NFA $A_1 \cap A_2$ over Σ that satisfies $L(A_1 \cap A_2) = L(A_1) \cap L(A_2)$.

(b) Consider NFAs A_1 and A_2 over disjoint alphabets $\Sigma_1 \cap \Sigma_2 = \emptyset$. Give NFAs A_1' and A_2' over $\Sigma_1 \cup \Sigma_2$ so that $L(A_1') \cap L(A_2') = L(A_1) \sqcup\!\sqcup L(A_2)$.

For (b), we define the *shuffle operation* on languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ by

$$L_1 \sqcup\!\sqcup L_2 := \bigcup_{u \in L_1, v \in L_2} \{u_1 v_1 \dots u_n v_n \mid u = u_1 \dots u_n,\ u_i \in \Sigma_1^*,\ v = v_1 \dots v_n,\ v_i \in \Sigma_2^*\}.$$

Note that $L_1 \sqcup\!\sqcup L_2$ is over alphabet $\Sigma_1 \cup \Sigma_2$. For example, $\{a.b\} \sqcup\!\sqcup \{x\} = \{a.b.x, a.x.b, x.a.b\}$.

Exercise 1.2 (Parallel) Programs as Automata

Boolean Programs are programs where all variables take Boolean values from $\mathbb{B} := \{0, 1\}$. For simplicity, we consider programs that only have reads $r(x, v)$ and writes $w(x, v)$. Here, x is taken from a finite set of variables Var and $v \in \mathbb{B}$. Reads are meant to be blocking, so $r(y, 0)$ cannot be executed if y is 1. Initially, all variables are set to 0. The following is an example program:

```
P1
 l0: w(x, 1) goto l1
 l1: r(y, 0) goto l2
 l1: r(y, 1) goto l1
 l2:                  //execution stops here
```

A finite run of program P is a sequence of read and write statements, i.e., a word over the alphabet $\Sigma := \{r, w\} \times \mathsf{Var} \times \mathbb{B}$. We denote the set of all finite runs of P by L_P.

(a) Construct an NFA A_{P_1} such that $L(A_{P_1}) = L_{P_1}$.

(b) Parallel programs $P_1 \parallel P_2$ execute multiple programs over the same variables Var. Assume program P_1 above is executed in parallel with

```
P2
 la: w(y, 1) goto lb
 lb: r(x, 0) goto lc
 lb: r(x, 1) goto lb
 lc:                  //execution stops here
```

Change your construction into an NFA $A_{P_1\|P_2}$ with $L(A_{P_1\|P_2}) = L_{P_1\|P_2}$.

What do you observe about $l2$ in P_1 and lc in P_2. If you are interested, search the web for *Dekker's protocol*.

(c) **This exercise is optional.** We can also directly understand the programs P_1 and P_2 as finite automata C_{P_1} and C_{P_2} that only reflect the control flow. This means the labels of P_1 are the states of C_{P_1} and the instructions are the transitions. In this setting, we additionally have to encode the behaviour of reads from and writes to variables. To this end, construct an automaton A_{Var} so that

$$(L(C_{P_1}) \sqcup L(C_{P_2})) \cap L(A_{\mathsf{Var}}) = L_{P_1\|P_2}.$$

Can you draw a connection between reachability and language emptiness?

Exercise 1.3 Some Proofs

Given an NFA A, prove that (a) $L(A^*) = L(A)^*$, and that (b) the powerset construction yields (i) a deterministic automaton (ii) which accepts $L(A)$.

Exercise 1.4 Arden's Lemma

Consider Arden's Lemma as given in class: *Let $U, V \subseteq \Sigma^*$ be languages such that $\varepsilon \notin U$. Then for any $L \subseteq \Sigma^*$, we have $L = UL \cup V$ if and only if $L = U^*V$.*

(a) Prove that $L = U^*V$ implies $L = UL \cup V$.

(b) Show that $\varepsilon \notin U$ is necessary for the other direction to hold. Specifically, find languages $U, V, L \subseteq \Sigma^*$ such that $L = UL \cup V$ and $L \neq U^*V$.

(c) Use language equations and Arden's Lemma to determine a regular expression for the following NFA: