Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen (LEA)
Prof. Dr. Ernst W. Mayr
Moritz Fuchs

Winter Semester 2014/15
Problem set 8
November 25, 2014

# Automata and Formal Languages

*Due December 02, 2014 before class!*

## Exercise 1 (Master automaton - 10 points)

(a) Construct the fragment of the master automaton that captures the languages

- $L_1 = \{0011, 0110\}$
- $L_2 = \{1101, 1010, 0000\}$
- $L_3 = \{1111, 1010\}$

(b) Construct initial states and necessary intermediate states for $L_1 \cap L_2$ and $L_2 \cup L_3$.

## Exercise 2 (Variable Ordering - 10 points)

Let $\Sigma = \{0, 1\}$ and define $a \cdot b$ as the usual multiplication for $a, b \in \Sigma$. Furthermore define $a \oplus b$ to be 0 if $a = b = 0$ and 1 otherwise.
Consider the function $f : \Sigma^6 \to \Sigma$ defined as

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = (x_1 \cdot x_2) \oplus (x_3 \cdot x_4) \oplus (x_5 \cdot x_6)$$

(a) Construct the minimal DFA recognizing $L_1 = \{x_1x_2x_3x_4x_5x_6 \mid f(x_1, x_2, x_3, x_4, x_5, x_6) = 1\}$

(b) Construct the minimal DFA recognizing $L_1 = \{x_1x_3x_5x_2x_4x_6 \mid f(x_1, x_2, x_3, x_4, x_5, x_6) = 1\}$

(c) Consider $f(x_1, ..., x_{2n}) = \bigoplus_{i \leq k \leq n}(x_{2k-1} \cdot x_{2k})$ as well as the orderings $x_1x_2...x_{2n-1}x_{2n}$ and $x_1x_3x_5...x_{2n-1}x_2x_4...x_{2n}$. How big is the minimal DFA recognizing these two orderings depending on $n$?

## Exercise 3 (Number of words - 10 points)

Give an efficient algorithm that computes the number of words a given minimal DFA for a fixed-length language accepts.

## Exercise 4 (Verification - 10 points)

In this exercise we want to model and verify mutual exclusion protocols. Let there be two agents with id 0 and 1. They both run the following program:

```
function agent(id)
  While(true) {
    enter(id)
    critical commands
    leave(id)
    non-critical commands
  }
```

(a)
```
turn = 0

function enter(id)
  While(turn=1-id) {
     Skip
  }

function leave(id)
   turn=1-id
```

- Design an asynchronous network of automata capturing this bahaviour.
- Construct the asynchronous product of the automata from the previous exercise. Can agent 0 and 1 run critical commands at the same time?
- Is it true that whenever an agent wants to enter the critical section he will eventually be allowed to?

(b)
```
flag[0] = false
flag[1] = false

function enter(id)
  flag[id] = true
  While(flag[1-id]) {
    skip
  }

function leave(id)
  flag[id]=false
```

- Design an asynchronous network of automata capturing this bahaviour.
- Can a deadlock occur?

(c) Now we combine both approaches.

```
flag[0] = false
flag[1] = false
turn = 0

 function enter(id)
  turn=1-id
  flag[id]=true
  While(flag[1-id] & turn=1-id) {
    skip
  }

 function leave(id)
  flag[id]=false
```

- Can a deadlock occur?
- Can any type of starvation occur?