DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics and Computer Science

# CSE2315 Lab Course
# Assignment 3

- This assignment consists of several exam-level questions and sometimes an optional extra exercise.
- See Brightspace, *Course Information* for instructions about the lab course, rules and grading procedure.
- Your solution has to be handed in on paper, typeset in LaTeX, using a word processor or readable handwriting. Handing in via email is not permitted.
- Cooperating with a colleague is permitted and *strongly encouraged!* In such cases, please hand in a single copy of the work.
- Total number of pages, without this cover page: 3.

1. Suppose we have the following language $L$ over the alphabet $\Sigma = \{a, 0, 1\}$:

$$L = \{ua^k v \mid u, v \in \{0,1\}^* \text{ and } c_0(u) + c_1(v) > k\}.$$

Here, $c_x(w)$ is defined as the number of occurrences of $x$ in $w$. Use the pumping lemma to show that $L$ is not regular.

2. Suppose we have the context-free grammar $G = (\{S, T, V\}, \{a, b, c\}, R, S)$ with $R$ containing the following rules:

$$\begin{aligned} S &\to Tc \mid cV \\ T &\to aTb \mid bTa \mid \varepsilon \\ V &\to a \mid b \end{aligned}$$

   (a) Describe the set $L(G)$ in your own words.
   (b) Construct a PDA $M$ such that $L(M) = L(G)$. Use no more than 8 states.
   (c) Explain (in no more than 12 lines) how your PDA works.

3. *Old exam question midterm 21-22:* Consider the following rules $R$ of a CFG $G = (V, \Sigma, R, S)$, with $V = \{S, A, B\}$ and $\Sigma = \{a, b\}$:

$$\begin{aligned} S &\to bB \mid abB \\ A &\to bA \mid AA \\ B &\to AS \mid Sa \mid \varepsilon \end{aligned}$$

Convert $G$ into Chomsky Normal Form (CNF). Use the procedure from Sipser, showing intermediate steps.

4. You are designing a new data format called TUDON (short for TU Delft Object Notation[1]). In TUDON, you can represent dictionaries of key/value pairs. Such a data format is useful when sending data between computers: the receiver can lookup the value the sender supplied for a key.

More precisely, each key in a dictionary is a sequence of lowercase letters (at least one letter). The value in a dictionary is either also such a sequence, or a dictionary itself. A dictionary starts with a < and ends with a >. Each key/value pair is separated by a colon, and key/value pairs themselves are separated by a semicolon. Each word in TUDON is a dictionary.

Examples of TUDON words are:

```
<>
<hello:world>
<a:b;test:<example:x;abc:<def:ghi>>;xyz:aaa>
```

Let $L$ be the set of all TUDON words.

   (a) Give a CFG $G$ such that $L(G) = L$, using no more than 6 variables and $9 + 26$ rules. (You'll need 26 rules to represent all lowercase letters. You are allowed to omit (abbreviate) most of these rules as long as it's clear what you mean.)
   (b) Explain why your grammar generates $L$. (For example, explain the purpose/meaning of each rule.)
   (c) Is your grammar ambiguous? If so, give a word $w$ that shows ambiguity. If not, make plausible why.

5. *Revision, old exam question resit 19-20:* We're introducing a new operator @ that transforms all words of a language in words of the same length but only consist of a's, formally defined as $A@ = \{a^{|w|} \mid w \in A\}$. Prove that regularity is closed under @, meaning if $A$ is regular, then $A@$ is also regular.

6. *Revision, old exam question resit 22-23:*
   (a) Explain why the following claim is true: Let $N$ be an arbitrary NFA with $k$ states. If $L(N) \neq \emptyset$, then there is a word of length $\leq k$ in $L(N)$.

[1] See also `http://json.org`, which contains a CFG you might want to use as inspiration.

(b) Provide a counterexample and explanation showing the following claim is false: Let $D$ be an arbitrary DFA $D$ with $k > 1$ states. There is a DFA $D'$ with $k-1$ states, such that $L(D') = L(D)$. Your counterexample only needs to give a formal representation or a transition diagram of $D$, you should then explain why no $D'$ can exist.

See next page for an optional extra ~~exercise~~ puzzle.

**Optional extra exercise** This last exercise is not part of the learning material. However, if you like puzzles and have some time left, you might like to do this challenging puzzle.

After the adventure of last week, there is still room for improvement. You realize that last week, the robot might have immediately vanished if you had chosen the right teleportation device at the start instead of the left one. Therefore, you have decided to make multiple copies of the robot, and to only pick up the treasure once you have explored the mansion sufficiently. For the first copy, you press a few buttons to explore (given below).

| Start | Left | Left | Right | Left | Right | Right | Right | Left |
|---|---|---|---|---|---|---|---|---|
| {Red} | {Yellow} | {Yellow} | {Green} | {Red} | {Yellow} | {Green} | {Blue} | {} |

This is even more trivial than you thought. Apparently, there are only 4 room colours, and since there are still a maximum of 4 rooms, each room has a unique colour. You can easily map out which teleportation devices in which rooms lead to which rooms. Clearly, the teleporation device that leads outside is the right one in the blue room. You can get there with $left \to right \to right \to right$. So you send out the second copy of the robot, which appears in the red room, and let it pick up the treasure. You press left, letting the robot arrive in the yellow room. However, when you then press right, you get a message telling you the robot is in a yellow room. That's weird. Earlier, the right teleportation device in the yellow room lead to the green room.

You decide to study a little more, and then you find it. Apparently, in your enthousiasm, you didn't read everything about this mansion. The teleportation devices in this mansion sometimes attach components to the robot. If the robot has no components, the teleportation device deconstructs and reconstructs it in 0 or more rooms as usual, but it might also attach 1 component to it (this can be different for each room). If the robot does have a component, the most recently placed component determines in which 0 or more **additional** rooms the robot is constructed. In these additional constructions (so not in the other constructions), the most recently placed component is removed, but another 1 component might be attached (again, this can be a different component per construction). Finally, there are 2 different components.

Knowing this, you think you know what to do. Maybe you need to press the left button and then press the right button. But again, you seem stuck in the yellow room. What is the quickest way that makes sure to get the treasure out?

| Start | Left | Right | Left | Right |
|---|---|---|---|---|
| {Red} | {Yellow} | {Yellow} | {Yellow} | {Yellow} |

- Each room has 2 teleportation devices, a left one and a right one
- Teleportation devices are deterministic
  - The same device in the same room will always lead to a given set of rooms, **each with 0 or 1 given extra component**
  - The same device in the same room **with the same most recent component** will always lead to a given set of **additional rooms, each with the most recent component removed and 0 or 1 given extra component**
- If 2 (or more) teleportation devices teleport a robot **with the same components** to the same room at the same time, only 1 of those will be build (the device cannot see that they are 2 signals)
- At least 1 teleportation device leads outside, **irrespective of its components**
- There are at most 4 rooms
- Rooms will not change colour (the same room will always have the same colour)
- **There are 2 different components**
- After the robot has taken the teleportation devices as given in the table (and entered rooms with wall colours as given in the table), what is the quickest way to get the robot outside to collect the treasure?

End of the assignment