Issue Date: 10.11.2014
Version: 2014-11-14 17:15

2nd Exercise Sheet for Kombinatorische Algorithmen, WS 14/15

Hand In: *Until Monday, 17.11.2014, 12:00,*
deliver or email to Raphael (`reitzig@cs.uni-kl.de`).

Problem 3

1 + 4 points

We investigate a generalisation of the *string matching problem*. We now search for a *pattern* (instead of a fixed string P) which we will assume to be given as a *regular language* $L \in \Sigma^*$. We will show that there are efficient algorithms for this problem as well.

You can assume that L be given in one of the usual, finite representations, e. g. finite automata, regular expressions or left-/right-regular grammars[1].

Assume furthermore that L is *fixed*, i. e. the asymptotics in the problem statements below do *not* depend on L resp. the size of its representation. Nevertheless, your algorithms should work for *any* regular L!

a) Develop an algorithm that solves the following problem $\mathcal{O}(n)$ time:

   Regular String Matching

   Input: $w \in \Sigma^n$

   Question: Does w match the pattern L, i. e. is $w \in L$?

b) Develop an algorithm for the following problem:

   Regular Substring Matching

   Input: Text $T \in \Sigma^N$ and regular language $L \subseteq \Sigma^*$

   Output: The set of all substring matches of L in T, i. e.

   $$\mathcal{M}_L(T) = \{(i,j) \in [1:n]^2 \mid i \leq j, T_{i,j} \in L\} .$$

   You algorithm should run in time $\Theta(n+k)$ where $k = |\mathcal{M}_L(T)|$. Note that since $\Omega(n+k)$ is a trivial lower runtime bound you are looking for an asymptotically runtime-*optimal* algorithm.

   Less efficient solutions may yield partial credit depending on how far off they are.

[1] You remember from your formal language theory course(s) that these can all be derived from each other efficiently.