Technische Universität München
Theoretische Informatik
Prof. J. Esparza / Dr.-Ing. J. Kreiker

Winter term 2008/2009
Exercise Sheet 5
November 12, 2008

Exercises "Automata and Formal Languages"

Exercise 5.1

Design an algorithm that solves the following problem for a finite alphabet Σ. Discuss the complexity of your solution.

- *Given*: $w \in \Sigma^*$ and a regular expression r over Σ.
- *Find*: A shortest prefix $w_1 \in \Sigma^*$ of w such that there exists a prefix $w_1 w_2$ of w and $w_2 \in \mathcal{L}(r)$.

Exercise 5.2

For each of the following languages give a 2DFA accepting it and compare its size to the size of the smallest DFA and NFA accepting the same language.

- $L_1 = \{wa\sigma\# \mid w \in \{a,b\}^*, \sigma \in \{a,b\}\}$ over $\{a,b,\#\}$
- $L_2 = \{\#_L w \#_R \mid |w|_{\sigma_i} = i \text{ for } i = 1,\ldots,n\}$ over $\{\sigma_1,\ldots,\sigma_n,\#_L,\#_R\}$, where $|w|_a$ is the number of occurrences of letter a in w.

Exercise 5.3

Consider the 2DFA, A, with states q_0, q_1, …, q_9 over $\{0,1\}$ with starting state q_0 and final state q_9 and the following transition function:

	q_0	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
0	(q_0,R)	(q_2,R)	(q_3,R)	(q_4,R)	(q_5,R)	(q_6,L)	(q_7,L)	(q_8,L)	(q_0,L)	(q_9,R)
1	(q_1,R)	(q_2,R)	(q_3,R)	(q_4,R)	(q_5,R)	(q_9,R)	(q_7,L)	(q_8,L)	(q_0,L)	(q_9,R)

Show A's computation on 11001010 and on 11001001. What does A accept? Construct an NFA accepting $\mathcal{L}(A)$ using the "crossing sequence" construction.

Exercise 5.4

Programming Competition! Turn page!

Programming Competition

In this competition, you are supposed to implement a *pattern matcher*. You may implement either the algorithm presented in class or any other algorithm you find. The rules are simple and the prizes are high.

- The pattern matcher takes two inputs – a text file, `text.txt`, and a string, `s`, (as a command line argument).
- It outputs the number of occurrences of `s` in `text.txt` as well as the prefix `text.txt` ending with the first occurrence of `s`.
- *Ignore* new lines (`\N`) and carriage returns (`\R`) (in contrast to `grep`).
- Your pattern matcher may be written in C, C++, Java, or an ML dialect (OCaml, SML/NJ, MoscowML, ...). No other languages allowed.
- You *must not* use any libraries other than facilities to input/output files. In particular, do not use any string manipulation packages/libraries. Don't use code found on the Internet.
- Your solutions will be checked for cheating. It is up to the jury to decide what cheating is.
- Submission deadline: **December 12**! Send an email to Jörg Kreiker containing:
  - the source code of your solution
  - instructions on how to compile and use your implementation (should be compilable on any standard Linux/Windows platform)
  - a description of the algorithm you have implemented and the optimizations made
  - the answers to the challenges below (if any) and the running times of your tool in order to answer these questions
- You may work in groups of up to three people.

Challenges

Your pattern matcher will be evaluated according to the size of the input it can handle and the time it takes to do so. At the following URL you can download a compressed archive containing four benchmark text files and a description of the challenges.

`http://www7.in.tum.de/um/courses/auto/ws0809/benchmark.zip`.

The archive contains `faust.txt`: Goethe's *Faust*, size app. 200KB; `homer.txt`: Homer's *Iliad* and *Odysee* in English, size app 1.4MB; `chromosome22.txt`: the complete primary structure of the human chromosome number 22, a string composed of the letters A, C, G, T, and N, size app. 34MB; `123.txt`: a file containing a single string of 1s, 2s, and 3s, size app. 42MB; and `challenge.txt`: a description of the challenges.

Prizes

- We will verify your solutions in a *live session* during the tutorial on *December 18* and determine the winner.
- The winner is the group whose tool solves most challenges. If more than one group solves all challenges, running times are considered.
- The winner will receive *a bottle of champagne* and a *crate of beer*. The runner-up will receive champagne only.
- There will be a special *jury's prize* awarded to the most elegant solution.

Good luck!