Homework 4: Regular Expressions and NFAs
CIS 352: Programming Languages
10 February 2016, Version 1

Administrivia

- **No teams**, this assignment is a solo effort.
- Document in the cover sheet any ideas you use from other students or other sources.
- For Part I, *legible* hand written answers are fine.
- Let me know if any of my QuickCheck tests seem dodgy.
- **Turn in Part I by:** dropping the papers in the CIS 352 bin on the 4th floor of SciTech. Include a paper copy of your cover sheet.

Grading Criteria

- The homework is out of 100 points.
- Each programming problem is $\approx 70\%$ correctness and $\approx 30\%$ testing.
- Omitting your name(s) in the source code looses you 5 points.
- **No teams**, this assignment is a solo effort.
- Document in the cover sheet any ideas you use from other students or other sources.
- For Part I, *legible* hand written answers are fine.
- Let me know if any of my QuickCheck tests seem dodgy.
- **Turn in Part I by:** dropping the papers in the CIS 352 bin on the 4th floor of SciTech. Include a paper copy of your cover sheet.

Part I: Problems on Paper

The languages (i.e., set of strings) considered in Problems 1, 2, and 3 are all over the alphabet \{a, b\}.

**Problem 1 (40 points)**

Use the rules on page 5 of the *Lexical Analysis* slides to give a formal derivation of each of the following. Each part is 4 points except for (i) which is 8 points.

(a) \((a|b)c) \downarrow a\)  (d) \((ab)c) \downarrow abc\)  (g) \((a|bc))^* \downarrow a\)
(b) \((a|b)c) \downarrow b\)  (e) \((ab)c) \downarrow abc\)  (h) \((a|bc))^* \downarrow bc\)
(c) \((a|b)c) \downarrow c\)  (f) \((a|bc))^* \downarrow \epsilon\)  (i) \((a|bc))^* \downarrow abca\)

**Problem 2 (16 points)**

(a) **BACKGROUND.** Let \(L_1 = \{ w \in \{a, b\}^* : w \text{ ends with } ab \}\). A regular expression for this language is: \((ab)^*ab\) and an NFA is \(M_1 = (\{0, 1, 2\}, \text{Moves}_1, 0, \{2\})\) where

\[
\text{Moves}_1 = \{ 0 \rightarrow a, 1 \rightarrow b, 0 \rightarrow 0, 1 \rightarrow 1, 1 \rightarrow b, 2 \rightarrow a, 1 \rightarrow 1, 2 \rightarrow 0 \}\]

or see Figure 1 for the diagram form.

**Your Problem:** (4 points) Give an \(M_1\)-accepting path for babbab. (See pages 18 and 19 of the *Lexical* slides.)
(b) Background. Let

\[ L_2 = \{ w \in \{a, b\}^* : w \text{ has substring } ab \text{ or substring } ba \} \]

which has \((a|b)^*ab(a|b)^*|(a|b)^*(ba)(a|b)^*\) as a regular expression. An NFA is \(M_2 = (\{0, \ldots, 6\}, \text{Moves}_2, 0, \{3, 6\})\) where

\[
\text{Moves}_2 = \begin{cases}
0 \xrightarrow{\epsilon} 1, & 0 \xrightarrow{\epsilon} 4, \\
1 \xrightarrow{a} 1, & 1 \xrightarrow{b} 1, \\
2 \xrightarrow{b} 3, & 3 \xrightarrow{a} 3, \\
4 \xrightarrow{a} 4, & 4 \xrightarrow{b} 4, \\
5 \xrightarrow{a} 6, & 6 \xrightarrow{a} 6, \\
\end{cases}
\]

or see Figure 2 for the diagram form.

Your problem: (12 points) Give four distinct \(M_2\)-accepting paths for \texttt{ababa}. (See pages 18 and 19 of the Lexical slides.)

 pena problem 3 (16 points) pena

For each of the following languages over \{a, b\}, give both (i) a regular expression and (ii) a NFA that precisely captures it.

(a) Those strings in which every b is immediately followed by at least two a’s.
(b) Those strings that have both ab and ba as substrings

\[ \text{Part II: Programming Problems} \]

You will need the files in \url{http://www.cis.syr.edu/courses/cis352/code/RegExp2/} and you will end up turning in changed versions of Matches2.hs and BuildNFA2.hs. This code is a modified version of Simon Thompson’s regular expressions and automata library.

\[ \text{Problem 4 (12 points) } \]

Background. Let \(w^R\) denote the reverse of string \(w\).

Your Problem. In Matches2.hs add a Haskell function:

\[ \text{revExp :: Reg } -> \text{ Reg} \]

\((\text{regExp } e), \text{ on regular expression } e, \text{ returns a new regular expression such that, for each String } w, \)

\[ e \text{ matches } w \iff (\text{revExp } e) \text{ matches } w^R. \]

Testing: Matches2.hs defines Reg’s \(\text{re1} \equiv a(a|b|c|d)^* \quad \text{re2} \equiv (a|b|c|d)^*ab(a|b|c|d)^* \)

Run:

\[
\begin{align*}
&\text{quickCheck (\text{rev_prop re1})} \\
&\text{quickCheck (\text{rev_prop re2})}
\end{align*}
\]

Also, come up with some convincing tests of your own.


4 Example. \((abcd)^R = dcba\). Example. If \(e\) represents \((ab|cd)^*\), then \((\text{revExp } e)\) represents \((ba|dc)^*\).
Problem 5 (16 points)

Background. On page 13 Mogensen defines the shorthands

\[ r? = \text{def} \ r|\epsilon \quad r^+ = \text{def} \ r(r^*) \]

A start at modifying Thompson’s library to handle these two new forms can be found in:

http://www.cis.syr.edu/courses/cis352/code/RegExp2/

Your Problems.

(a) In Matches2.hs the function matches does not have cases for Opt or Plus expressions. Add the missing cases to matches.

Testing: Run (quickCheck prop_equivA). Also come up with some convincing tests of your own.

(b) In BuildNFA2.hs the function build is missing cases for Opt or Plus expressions. Add the missing cases to build.

Testing: Run (quickCheck prop_equivB). Also come up with some convincing tests of your own.

Reference rule-sets

Rules for a big-step rules for regular expression matching

\[
\begin{align*}
\epsilon: & \quad \epsilon \Downarrow \epsilon \\
|: & \quad r_1 \Downarrow s \\
& \quad (r_1|r_2) \Downarrow s \\
& \quad (r_1|r_2) \Downarrow s \\
\text{Lit:} & \quad x \Downarrow x \\
\text{Seq:} & \quad r_1 \Downarrow s_1 \\
& \quad r_2 \Downarrow s_2 \\
& \quad (s = s_1s_2) \\
\text{Star:} & \quad r^* \Downarrow \epsilon \\
& \quad r^* \Downarrow s \\
& \quad r^* \Downarrow s \\
& \quad (s = s_1s_2)
\end{align*}
\]

A small-step semantics for an NFA. For \( M = (\text{States}, \text{Moves}, \text{start}, \text{Final}) \):

\[
M \vdash s \xrightarrow{a} s' \quad ((s,a,s') \in \text{Moves})
\]

\[
M \vdash s \xrightarrow{\epsilon} s' \quad ((s,\epsilon,s') \in \text{Moves})
\]

Example. See page 6 of the Lexical slides for sample derivations.

Example. For the NFA with diagram:

\[
\begin{align*}
1 & \xrightarrow{a} 2 \\
1 & \xrightarrow{b} 3 \\
2 & \xrightarrow{a} 1 \\
2 & \xrightarrow{c} 3 \\
1 & \xrightarrow{c} 2 \\
1 & \xrightarrow{b} 3
\end{align*}
\]