

CSE105 (spring 2008): Homework 2

Instructor: Daniele Micciancio

Due on Tuesday, April 29, 2008, in class.

Solutions to the homework should be **typed up**, printed out, and turned in at the beginning of class. A homework submission page for the JFLAP file will appear on the course webpage shortly. The files must be named *exactly* as shown in the problem description.

All NFAs in the following problems are allowed to have ϵ -transitions.

1 Nondeterministic Finite Automata

Prove that every NFA can be converted into an equivalent NFA that has a single accept state. This proof should be very short, no more than 3 sentences.

Here is an example solution to a *different* question:

- Question: Show that any NFA can be converted into an equivalent NFA that has no transitions pointing to the start state.
- Answer: Any NFA can be converted into an equivalent NFA satisfying the desired property by adding a new state to the original automaton, make this new state the start state in the new automaton, and adding an epsilon transition from the new start state to the original start state.

2 Regular Expression to NFA

Convert the regular expression $(a + bc)^*(ab + c)^*$ into an NFA using the procedure described in class. Build the NFA in JFLAP. Name the file `hw2-2.jff`.

Remark: JFLAP allows you to input a regular expression, and then automatically convert it into an automaton. This is **not** what you are being asked to do. You should convert the regular expression into an NFA *by hand*, and then draw your NFA using JFLAP.

3 Closure of Regular Languages

If L_1 and L_2 are languages, define a new language

$$\text{INTERLACE}(L_1, L_2) = \{w_1v_1w_2v_2 \dots w_nv_n \mid w_1w_2 \dots w_n \in L_1, v_1v_2 \dots v_n \in L_2\}.$$

For example, if $abc \in L_1$ and $123 \in L_2$, then $a1b2c3 \in \text{INTERLACE}(L_1, L_2)$. Notice that the interlace operation always combines strings of equal length, so that if $L_1 = \{ac, abc\}$ and $L_2 = \{123, 45\}$, then $\text{INTERLACE}(L_1, L_2) = \{a1b2c3, a4c5\}$.

Show that if L_1 and L_2 are regular languages, then $\text{INTERLACE}(L_1, L_2)$ is a regular language. [Hint: Let $M_1 = (Q_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, s_2, F_2)$ be DFAs such that $L(M_1) = L_1$ and $L(M_2) = L_2$. Now describe each component of the DFA $M = (Q, \Sigma, \delta, s, F)$ with $L(M) = \text{INTERLACE}(L_1, L_2)$.]

4 More Regular Language Closure

This problem has two parts, where the second part depends on the solution to the first part.

1. If L is a language over the alphabet Σ and $a \in \Sigma$, define the quotient $L/a = \{w \mid wa \in L\}$. For example, if $L = \{a, aab, baa\}$, then $L/a = \{\varepsilon, ba\}$. Prove that if L is regular, then so is L/a . Your solution should have the following structure:

Since L is regular, there exists a DFA $M = (Q, \Sigma, \delta, s, F)$ such that $L(M) = L$. We build a new automaton $M' = (Q, \Sigma, \delta, s, F')$ with the same set of states, transition function, and start state as the original M . The only difference is the set of accepting states, which is defined as $F' = \dots$. The language of the new automaton is $L(M') = L/a$, because ... *brief explanation* ... So, L/a is also regular.

2. If L is a language over the alphabet Σ and $a \in \Sigma$, define $a \setminus L = \{w \mid aw \in L\}$. Prove that if L is regular, then so is $a \setminus L$. For this part you are not required to give a direct proof, showing how to transform automata. You can give a simpler proof building on the previous part, and other closure properties studied in class. (E.g., the same way as we proved that the intersection of regular languages is regular, using DeMorgan's laws.) [Hint: Remember that regular languages are closed under reversal.]

5 Practical Regular Expressions

For this problem, you are going to be working with real data taken from the UCSD library system and the UNIX command line tool `egrep`. Read the manual page for `egrep` (`man egrep` from the command line) to see what format is used for regular expressions. In particular, a single period matches any character while `|` is used for the union operation. The `*` operator is the same and `()` act as grouping. Lastly, `egrep` works a line at a time and it matches at any point in the line, not necessarily at the beginning. That means, it acts as though the regular expression starts and ends with `.*` to match any number of any character.

The data file `data.txt` can be downloaded from the course web page.

For example, to find every entry that is in the SSH 6th or 7th floor, run the command:

```
egrep 'SSH (6|7)th Floor' data.txt
```

This should print out quite a lot of data, 191 lines worth, in fact. Instead of printing out all of the lines, we can just get a count of the lines by using the `-c` option to `egrep` as in the command:

```
egrep -c 'SSH (6|7)th Floor' data.txt
```

which prints out 191.

For each query below, you need to include in your write up the regular expression you used (`SSH (6|7)th Floor'` in the above example) as well as the count of the lines that match *not the lines themselves*. All of them will have at least one, so if you get zero, something went wrong. I suggest examining the output without the `-c` to ensure that you are finding the entries you want. Note that each book can have multiple call numbers in the library. You should consider each of those to be a separate book. In addition, each call number can have multiple copies, these can be treated as a single book.

1. Modify the example above to find all books that are in the SSH 6th or 7th floor and have a year of 1999. Note that years are displayed as either 1999 or c1999. Make use of `.*` to skip over data you do not care about. How many are there?
2. How many books have a call number whose first part is exactly QA1? That is, QA1 .L471 v.642 should be matched but QA171 .M648 1961 should not.
3. How many books are in the S&E Stacks which are available? (Look at the data file to see how available books are represented; it should be very obvious.)
4. How many books have a due date in June, 2008? (Note that multiple copies can be due on different dates. That's okay. If at least one copy is due in June, count it.)
5. How many books in the S&E Stacks are available and have a year between 1973 and 2001, inclusive?