

1 Non-Deterministic Finite Automata

See posted JFLAP files.

2 Extended Regular Expressions

- (a) “ $E?$ ” is equivalent to “ $(E \cup \epsilon)$ ”.
- (b) “ E^+ ” is equivalent to “ (EE^*) ”.
- (c) “ $E_1 <> E_2$ ” is equivalent to “ $(E_1E_2 \cup E_2E_1)$ ”.
- (d) “ $((a(b?)) <> c)^+$ ” is equivalent to “ $(a(b \cup \epsilon)c \cup ca(b \cup \epsilon))(a(b \cup \epsilon)c \cup ca(b \cup \epsilon))^*$ ”.

3 Practical Regular Expressions

- 1. `egrep -c 'SSH.*(6th|7th).*2002' data.txt` matches 7 lines.
- 2. `egrep -c 'QA1 ' data.txt` matches 3 lines.
- 3. `egrep -c 'S&E Stacks.*AVAILABLE' data.txt` matches 109 lines.
- 4. `egrep -c '.*;.*S&E Stacks.*;.*;.*(199.|200.)' data.txt` matches 67 lines.
- 5. `egrep -c '.*;.*S&E Stacks.*;.*;.*AVAILABLE.*(197(5|6|7|8|9)|198.|199.)' data.txt` matches 64 lines.

4 DFA/NFA/Regex

First we created a NFA which simulates dividing the input (as a binary number) by 5. The NFA ignores the quotient, but keeps track of the remainder in 5 states corresponding to a remainder of 0 through 4 respectively. The last thing we added to our NFA was a new start state to ensure that the input begins with a 1. We then converted this NFA to a regular expression using the algorithm presented in the book and in class. The resulting regular expression is “ $1 + 10(10^*10)^*10^*1 + (11 + 10(10^*10)^*10^*11)(01 + (1 + 00)(10^*10)^*10^*11)^*(0 + (1 + 00)(10^*10)^*10^*1) + (10(10^*10)^*0 + (11 + 10(10^*10)^*10^*11)(01 + (1 + 00)(10^*10)^*10^*11)^*(1 + 00)(10^*10)^*0)(1 + 0(01 + (1 + 00)(10^*10)^*10^*11)^*(1 + 00)(10^*10)^*0)^*0(01 + (1 + 00)(10^*10)^*10^*11)^*(0 + (1 + 00)(10^*10)^*10^*1)$ ”, where “ $+$ ” is the JFLAP syntax for $\cup$. See posted JFLAP files for details.