COS126 Regular Expressions, DFAs (Booksite §7.2, 7.3)

See also the online chapter in the lecture page Reading column.

Part 1

Consider the regular expression `((C|D|M|N|P|T)A)*`

- Is PAPA generated by this RE? Is MAMAN? Is NAPA? Is TAMPA? NAPA is, but MAMAN and TAMPA are not

- What two country names can be generated? PANAMA and CANADA

Part 2 — RElay Race

Write regular expressions for the following languages.

1. all binary strings `(0|1)*`

2. all non-empty binary strings `(0|1)(0|1)*`

3. all binary strings beginning and ending with 1 `1|1(0|1)*1`

4. all binary strings ending with 00 (divisible by 4) `(0|1)*00`

5. all binary strings with at least three 1s `0*10*10*1(0|1)*`, `(0|1)*1(0|1)*1(0|1)*1(0|1)*`, etc

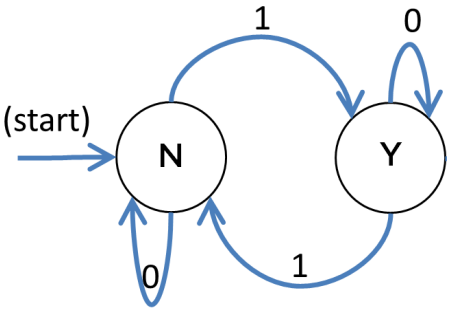
Part 3

What does `(0*10*10*)*` generate? (Describe this set of strings in English) All binary strings with an even number of 1s, except that it misses those strings consisting of just one or more 0s

Bonus

Hard bonus: can we generate set of all binary integers divisible by 3? Yes

Part 4



- Is 01101 accepted by this DFA? Is 11? 01101 is, 11 is not

- What is an English description for the set of all strings it accepts? All binary strings with an odd number of 1s

- (Optional) What is a Regular Expression description for the set of all strings it accepts? 0*10*(0*10*10*)* (other formulations possible)

Part 5

Write 5 DFAs that accept the 5 languages from Part 2: see next page

1. all binary strings

2. all non-empty binary strings

3. all binary strings beginning and ending with 1

4. all binary strings ending with 00 (divisible by 4)

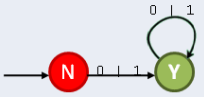
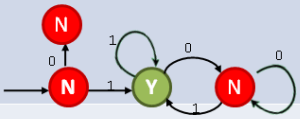
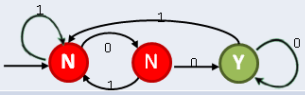
5. all binary strings with at least three 1s

Bonus

Write a DFA that accepts the set of all Java double literals. Use the RE

```
(\+|-|)([0-9]+(|\.[0-9]*)|\.[0-9]+)(|(E|e)(\+|-|)[0-9]+)
```

Recommended RE/DFA exercises from the exam archive: Fall 2011, Exam 2, question 4. Spring 2013, Exam 2, question 4.

Language	Regular Expression	DFA
All binary strings	`(0|1)*`	
All binary strings except empty string	`(0|1)(0|1)*`	
Begins with 1, ends with 1	`1	1(0|1)*1`	
Ends with 00	`(0|1)*00`	
Contains at least three 1s	`(0|1)*1(0|1)*1(0|1)*1(0|1)*`