CSC416 - Homework 1

Marshall Bowers

September 15, 2015

Regular Languages

1. Write English descriptions for the languages generated by the following regular expressions:

[4] (a) $(0|1|...|9|A|B|C|D|E|F)^+(x|X)$
Any combination of hexadecimal digits (`0-9`, `A-F`) one or more times, followed by either an uppercase or a lowercase `x`.

[4] (b) $(a|b)^*(a|b|\epsilon)$
A lowercase `a` or lowercase `b` zero or more times, followed by either a lowercase `a`, lowercase `b`, or nothing.

2. Write regular expressions for each of the following.

[4] (a) All strings of lowercase letters that begin and end in `a`.
$a[a-z]^*a$

[4] (b) All strings of digits that contain no leading zeros.
$[1-9][0-9]^*$

[4] (c) All strings of digits that represent even numbers.
$([1-9][0-9])^*(0|2|4|6|8)$

[4] (d) Strings over the alphabet {a,b,c} with an even number of `a`'s.
$((aa)^+(b|c)^*)|((b|c)^*(aa)^+(b|c)^*)|((b|c)^*(aa)^+)$

[4] (e) Strings over the alphabet {a,b} that contain an odd number of `a`'s or an odd number of `b`'s (or both).
$a(aa)^*|b(bb)^*|a(bb^*)b(aa)^*$

3. For each of the following regular expressions determine which of the strings `cc`, `ababb`, `bbcab`, and `ccbbab` matches it:

4 (a) `(ab)*c|a*b*c*`

`cc` ✓
`ababb` ✗
`bbcab` ✗
`ccbbab` ✗

4 (b) `(ab|bc*)+`

`cc` ✗
`ababb` ✓
`bbcab` ✗
`ccbbab` ✗

4 (c) `[ab]*cc?(ab)*`

`cc` ✓
`ababb` ✗
`bbcab` ✓
`ccbbab` ✗

Lexical Specifications

3 4. Given the string `abbbaacc` and alphabet {a,b,c} what tokenization will the following lexical specification produce?

Token Class	Regex
A	b+
B	ab*
C	ac*

`<B, abbb>`
`<B, a>` or `<C, a>`
`<B, a>` or `<C, a>`
`<A, bbb>`

3 5. Given the string `babac` and alphabet {a,b,c} what tokenization will the following lexical specification produce?

Token Class	Regex
A	a(ba)*
B	b*(ab)*

```
     C            abc
     D            c+

<B, bab>
<A, a>
<D, c>
```

6 6. Given the following lexical specification and alphabet {0,1}, which of the below strings will be successfully tokenized?

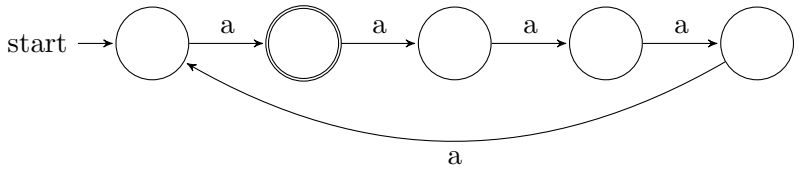
```
Token Class      Regex
-----------      -----
     A            (11)*
     B            01+
     C            10+
```

1. 1000001 ✓
2. 1110010 ✓
3. 01100100 ✓
4. 10011001 ✓

Finite Automata

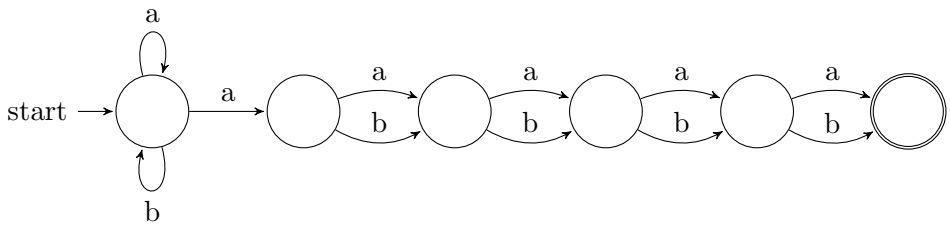
7. Explain in informal English what each of these finite-state automata recognizes.

5 (a)



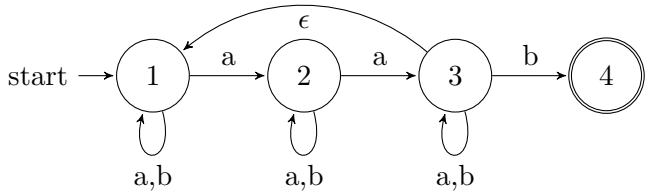
A single `a`, optionally followed by any number of sequences consisting of 5 `a`'s.

5 (b)



Any number of `a`'s or `b`'s, followed by an `a`, followed by a comibination of 4 `a`'s and `b`'s.

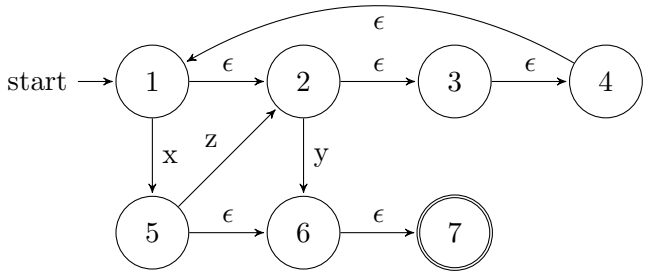
6 8. Write a regular expression whose language is equivalent to the following NFA.

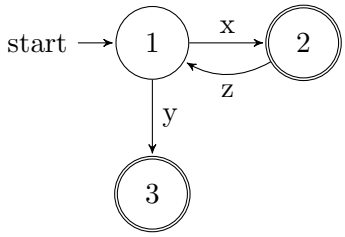


$((a|b)^*a(a|b)^*a(a|b)^*)^+b$

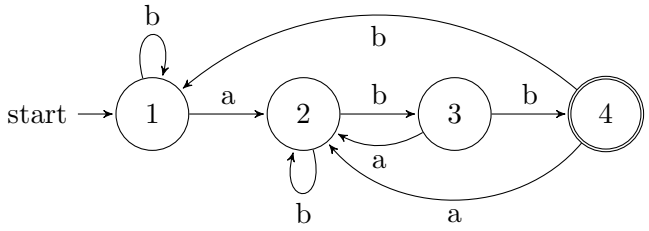
9. Convert these NFAs into DFAs.

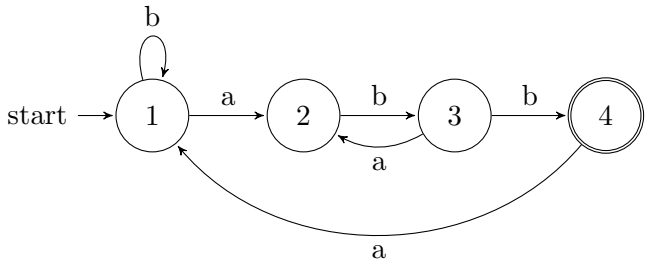
6 (a)





6 (b)





10. Construct DFA's for each of the following regular expressions. Do it in two steps: construct the NFA using Thompson's construction, then the DFA from the NFA. Let the alphabet be {a,b}.

6 (a) `a*|b*`

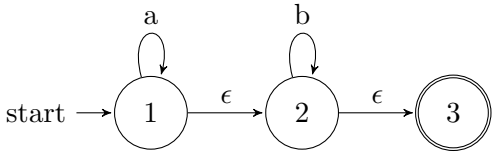


Figure 1: NFA

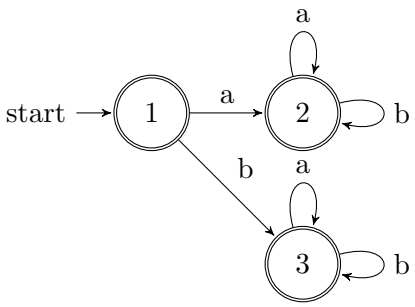


Figure 2: DFA

7 (b) `a*(ab)*a*`

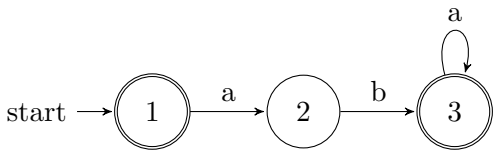


Figure 3: NFA

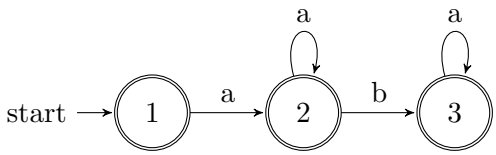


Figure 4: DFA

7 (c) **CSC416 ONLY:**

`[ab]*abab`

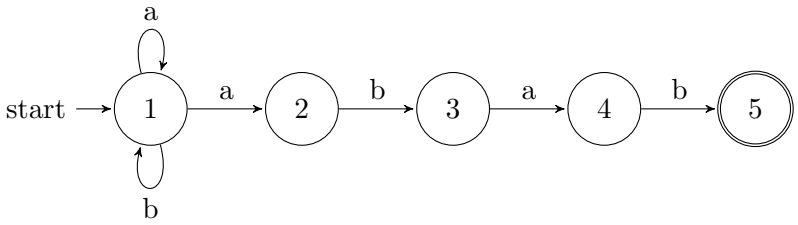


Figure 5: NFA

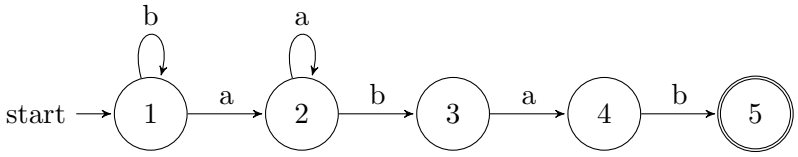


Figure 6: DFA

CSC565 ONLY:

`[ab]*abab[ab]*`

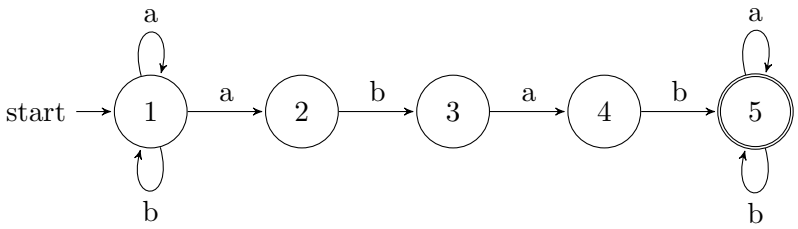


Figure 7: NFA

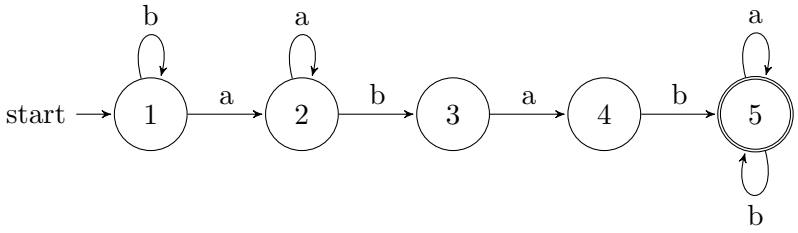


Figure 8: DFA