

CS 370

Midterm Exam

Solutions

The following exam is open book and open notes. You may feel free to use whatever additional reference material you wish, but **no electronic aids** are allowed. Please note the following instructions. There will be a ten point deduction for failure to comply with them:

- start each problem on a new sheet of paper
- write your social security number, but not your name, on each sheet of paper you turn in
- show your work whenever appropriate. There can be no partial credit unless I see how answers were arrived
- be succinct. You may lose points for facts that, while true, are not relevant to the question at hand

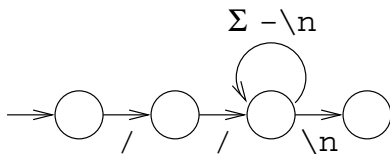
You have until 10:20 to finish the exam.

1. (15 points) C^b uses the same comment syntax as C++: A comment starts with `//`, and continues until the end of the line. Write a regular expression that will recognize C^b comments. Translate your regular expression into a finite state machine.

This needs to recognize the leading //, skip over everything that isn't a newline, and then see the newline. So it looks like this

`// [^\n]* \n`

Here's the FSM:



Points	Description
-2	loop on all ASCII, instead of all ASCII except newline
-2	~ after loop, before \n
-3	no newline at end
-1	Treated \n as two characters
-7	no FSM
-2	No * on the .

2. (10 points) A constant frustration with C comments is that they can't nest: if you have a block of code you'd like to comment out, like this:

```
/* comment out the following:
   a = b + c; /* add b to c */
   b = 2 * a;
   OK, end the commenting-out */
```

only the first line (`a = b + c;`) gets commented-out. The `*/` at the end of that line ends the comment, so the second line is compiled, and then the closing remark is a bunch of syntax errors.

Comment processing is normally handled by the scanner. Could the C scanner be modified to recognise nested comments (without doing violence to the sense in which we've used the term "scanner")? Either outline how it could be done or say why it couldn't.

No. Recognizing nested parentheses can't be done by a FSM. Modifying the scanner until it could do it would make it no longer a scanner (as the term is normally used).

Points	Description
-3	Makes it "ambiguous and unclear"
-8	Yes, just like nested loop (!)

3. (30 points) Give a nondeterministic pushdown automaton that will recognise palindromes taken from the alphabet $\{a, b, c\}$

The basic idea is that when you're in the first half of the string you push every symbol you see; when you're in the second you pop symbols off the stack and match them with the symbols coming in. So we get

$$M = (\Sigma, \Gamma, S, T, s_0, A)$$

where

$$\Sigma = \{a, b, c\}$$

$$\Gamma = \{A, B, C\}$$

$$S = \{L, R\}$$

$$T = \{$$

$$T(L, a, \epsilon) = (L, A)$$

$$T(L, b, \epsilon) = (L, B)$$

$$T(L, c, \epsilon) = (L, C)$$

$$T(L, \epsilon, \epsilon) = (R, \epsilon)$$

$$T(L, a, \epsilon) = (R, \epsilon)$$

$$T(L, b, \epsilon) = (R, \epsilon)$$

$$T(L, c, \epsilon) = (R, \epsilon)$$

$$T(R, a, A) = (R, \epsilon)$$

$$T(R, b, B) = (R, \epsilon)$$

$$T(R, c, C) = (R, \epsilon)$$

}

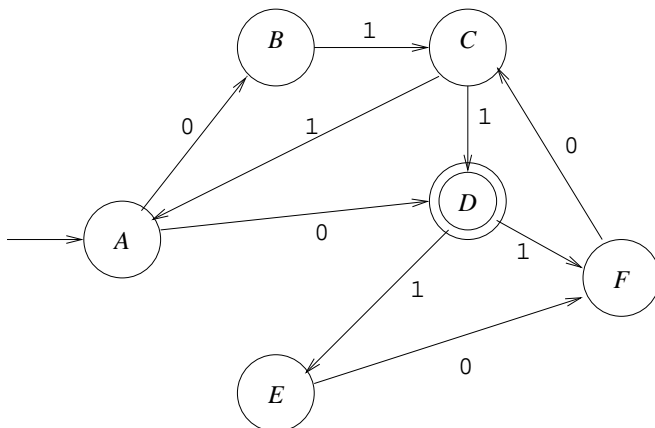
$$s_0 = L$$

$$A = \{R\}$$

The first three rules push the left side. The next four rules guess that we've reach the halfway point; if the palindrome is of odd length we just change state, while if it's even we we also push the character so we can pop it again in a second. The last three states match the stack against the incoming character and pop it.

Points	Description
15	PDA in reasonably proper form; couldn't really figure out what it was doing
5	No stack
10	Format for displaying PDA that I can't understand
28	All but odd-length palindrome
20	Extend balanced parens to three symbols

4. (20 points) Which of the following strings is accepted by the nondeterministic finite state machine in the figure?



I've labelled the states so I can talk about them in the solution.

(a) 0111001

Yes. We go through the states ABCDEFCD

(b) 010010

Yes. We go through the states DEFCAD

Oops. Only A was supposed to be accepted.... since there was a mistake on the exam, and you would reasonably have assumed from the wording that only one of the strings would be accepted, I gave people who concluded only one would be accepted a lot of slack. Somebody who showed one was accepted and quit only lost five points; even somebody who just said one was accepted and didn't show any work at all only lost ten.

<i>Points</i>	<i>Description</i>
15	<i>Only got one, didn't show work on other</i>
10	<i>Only got one, didn't show work on either</i>
18	<i>Only got one, showed work on other</i>

5. (25 points) The following is a description of a PDA.

$$M = (\Sigma, \Gamma, S, T, s_0, A)$$

where

$$\Sigma = \{0, 1\}$$

$$\Gamma = \{X, Y\}$$

$$S = \{s_0\}$$

$$T = \{$$

$$T(s_0, 0, \epsilon) = (s_0, X)$$

$$T(s_0, 0, Y) = (s_0, \epsilon)$$

$$T(s_0, 1, X) = (s_0, \epsilon)$$

$$T(s_0, 1, \epsilon) = (s_0, Y)$$

}

$$A = \{s_0\}$$

Which of the following strings is accepted by this PDA?

(a) 0010111001

Yes

(a) 01100

No

A number of people figured out what the PDA did (it made sure the number of 0's and 1's was the same), and answered based on that. Not what I had in mind, but a successful technique...