CS 205 - Problem Set 5 - Invariants, Languages and DFA's

Sections 01-03 and 07-09

Due Date, Wednesday November 23, 2022

Instructions

- This is an individual worksheet. NO group work is permitted.
- You are authorized to seek help from course staff ONLY
- This handout is also available under Canvas Files
- If you have any questions about a specific problem, ask questions in Piazza
- Submit solutions to gradescope
- late submissions will not be accepted.

your work may not be graded without you signing below

I certify that this paper represents my own work and I have read RU academic integrity policies https://www.cs.rutgers.edu/academic-integrity/introduction

Sign and PRINT your name : Akshaj Kammari

Your recitation section and/or day of the week : 7 (Tuesday)

NetID : ak1990

Question 1 - Invariants

The function foo takes an array of ints and perform some computation.

```
void foo(int[] a)
{
 int i=0, c=0, k=1;
 int  n = a.length;
 while (i < n)
  {
    a[i] = c;
    c = c + k;
    k = k + 2;
    i = i + 1;
  }
}
```

1. Create a table of the values a[i], c, k, i for i = 0,1,2,3,4

i	a[i]	c	k
0	.	0	1
1	0	1	3
2	1	4	5
3	4	9	7
4	9	16	9

2. Find two loop invariants

1. $i \le n$

2. $k = 2i + 1$

3. Prove one of the loop invariants

$i \le n$:

$i = 0 \le n$ $\quad$ $n \ge 0$ bc it is length

$i < n \Rightarrow i + 1 \le n$

$i' = i + 1 \Rightarrow i' \le n$ ✓

Question 2 - Post condition

Consider the following code. Assume that x is any real number.

```
p = 1;   i = 1;
while (i <= n) {
   p = p*x
   i = i + 1
}
return p;
```

1. Find two non-trivial loop invariants that involve variables i, and p (and n which is a constant). They must be strong enough to get the post condition.

i	p	n
1	1	n
2	x	n
3	x^2	n
4	x^3	n
n	x^{n-1}	n

1. $i \leq n+1$

2. $\log_x p = i - 1$

2. prove that each one is indeed a loop invariant.

1. $\underline{i \leq n+1}$

$i \leq n$

$i' = i+1$

$i+1 \leq n+1$

$i' \leq n+1 \Rightarrow$ $i \leq n+1$ is a loop invariant

2. $\underline{\log_x p = i - 1}$

$p = x^{i-1}$ $i' = i+1$

$p' = px$ $i = i - 1$

$px = x^i$

$p' = x^i$

$p' = x^{i'-1}$

$\log_x p' = i' - 1 \Rightarrow$ $\log_x p' = i' - 1$ is a loop invariant

3. What does this program compute?

X^n

4. Use the loop invariant and post condition to prove that this program indeed correctly computes what you specified before.

Because the post condition is $i > n$, $n = i - 1$ when the loop ends due to $\log_x p = i - 1$ & $\log_x p = n \Rightarrow x^n = p$.

Problem 3 Phrase Structure Grammar

Construct a phrase-structure grammar for the set of all fractions of the form a/b, where a is a signed integer in decimal notation and b is a positive integer.

1. Construct the Backus–Naur rules/form for this grammar.

function ::= <signed integer>/<integer>

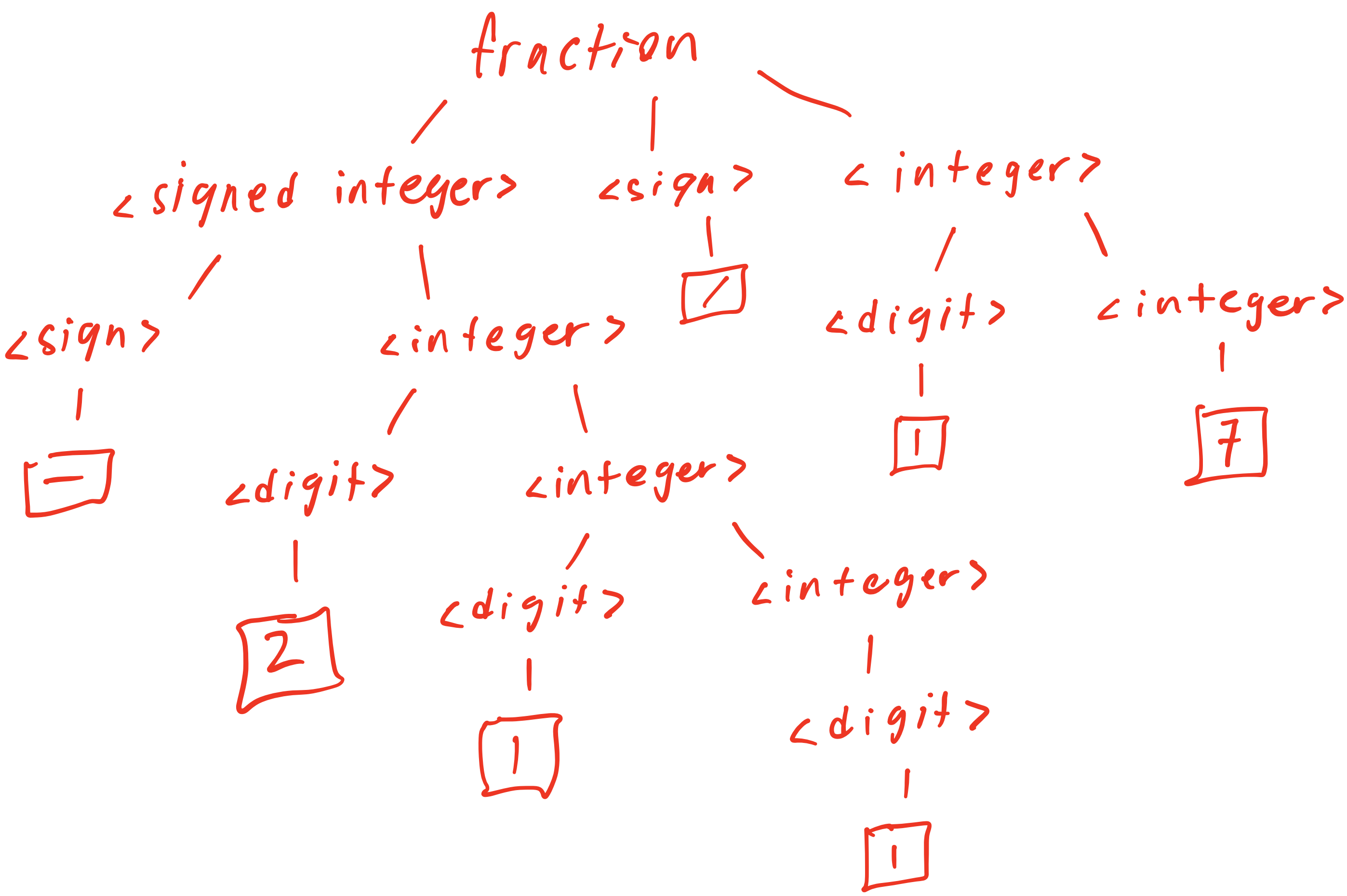
signed integer ::= <sign><integer>

sign ::= +/-

integer ::= <digit> | <digit><integer>

digit ::= 0|1|2|3|4|5|6|7|8|9

2. Construct a derivation tree for -211/17 to prove that it is a valid symbol in the grammar.



Problem 4 Finite State Machines

3-way switches are commonly used to control one light fixture from two different locations. For example, a long hallway or stairway might use a three-way switch at each end so that lights can be turned on when approaching one end of the hall or stairway, then shut off from the other end. Consider designing a 3-way switch using a FSM. We can call the locations {A,B} and the state of the light as {ON,OFF}. Only one switch can be flipped at a time. A switch can be in one of two modes, up(1) or down(0). For example, {A1,B0} means location A is up, and location B is down. Initially, both location switches will be set to down positions.

1. identify the set of inputs to FSM

states: $\{00, 01, 10, 11\}$

inputs: $\{A, B\}$

$S_1 : 00 - A_0 B_0$

$S_2 : 01 - A_0 B_1$

$S_3 : 10 - A_1 B_0$

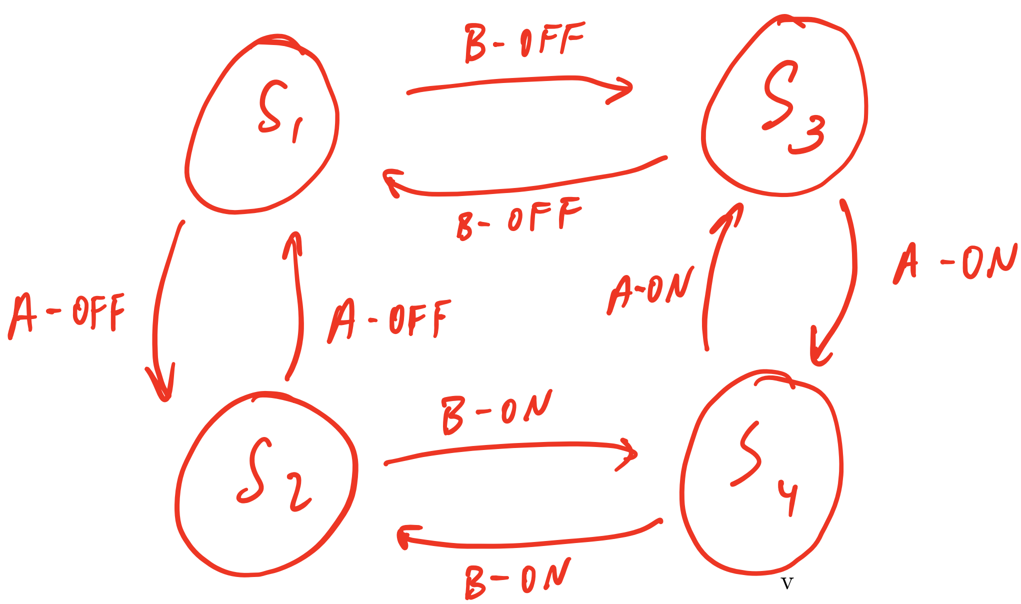
$S_4 : 11 - A_1 B_1$

2. identify the set of outputs

$\{ON, OFF\}$

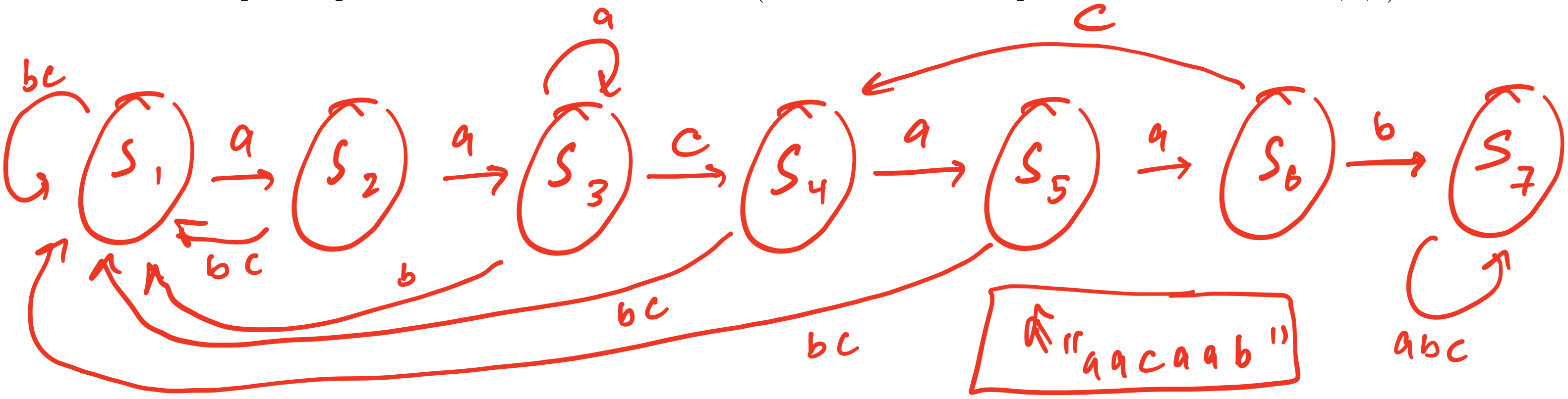
3. Determine how many states are needed. Draw the FSM.

$\underbrace{\{S_1, S_2, S_3, S_4\}}_{\text{4 states}}$

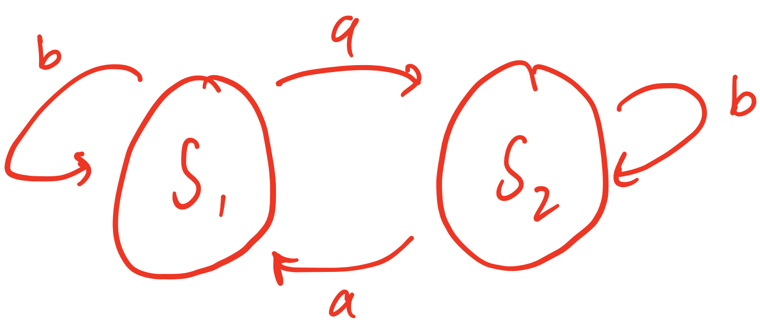


Problem 5 - Deterministic Finite Automata (DFA)

1. Construct a DFA for the string "aacaab". That is, given an infinite tape of symbols, this DFA accepts all patterns of the form "aacaab" (assume that the alphabet has characters a,b,c)

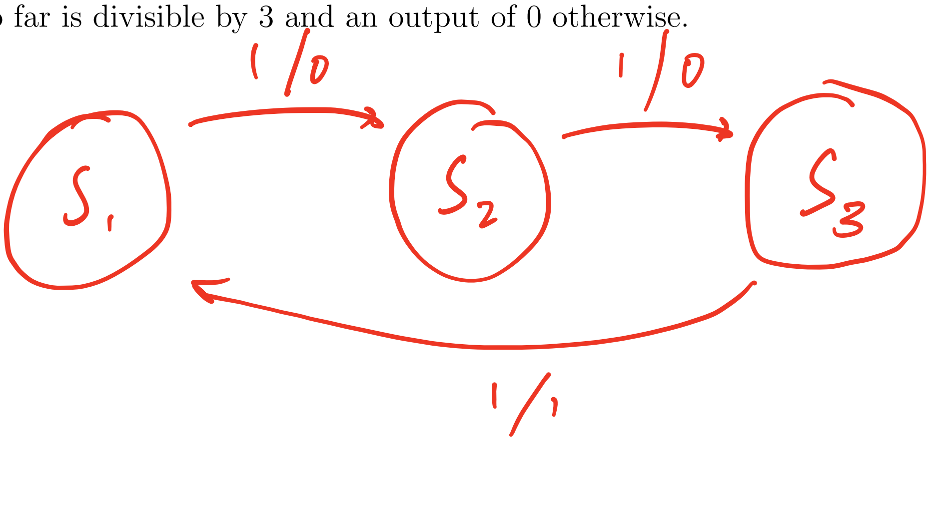


2. Construct a DFA that accepts all patterns that have an odd number of a's. Assume that the language alphabet only has characters a,b and machine processes an infinite tape of a's and b's



Q_2: odd # of a's

3. Construct a finite-state machine that gives an output of 1 if the number of input symbols read so far is divisible by 3 and an output of 0 otherwise.



1 may be any symbol