University of Nevada, Las Vegas Computer Science 456/656 Spring 2021
Assignment 2: Due Friday September 16 2022, 11:59 PM

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet. Turn in the assignment in the manner given to you by our grader, Janeen Sudiacal.

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time. In the questions below, $\mathcal{P}$ and $\mathcal{NP}$ denote $\mathcal{P}$-TIME and $\mathcal{NP}$-TIME, respectively.

   (i) **T** The language $\{a^nb^n \mid n \geq 0\}$ is context-free.

   (ii) **F** The language $\{a^nb^nc^n \mid n \geq 0\}$ is context-free.

   (iii) **T** The language $\left\{a^ib^jc^k \mid j = i + k\right\}$ is context-free.

   (iv) **T** The intersection of any two regular languages is regular.

   (v) **T** The intersection of any regular language with any context-free language is context-free.

   (vi) **F** The intersection of any two context-free languages is context-free.

   (vii) **T** If L is a context-free language over an alphabet with just one symbol, then L is regular.

   (viii) **T** The set of strings that your high school algebra teacher would accept as legitimate expressions is a context-free language.

   (ix) **T** Every language accepted by a non-deterministic machine is accepted by some deterministic machine.

   (x) **T** The language $\{a^nb^nc^nd^n \mid n \geq 0\}$ is recursive.

   (xi) **T** The language $\{a^nb^nc^n \mid n \geq 0\}$ is in the class $\mathcal{P}$-TIME.

   (xii) **O** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a binary numeral.

   (xiii) **F** Every problem that can be mathematically defined has an algorithmic solution.

   (xiv) **T** Every $\mathcal{NP}$ language is decidable.

   (xv) **T** The intersection of two $\mathcal{NP}$ languages must be $\mathcal{NP}$.

   (xvi) **O** $\mathcal{P} = \mathcal{NP}$.

   (xvii) **O** $\mathcal{NP} = \mathcal{P}$-SPACE

   (xviii) **T** Primality, using binary numerals, is $\mathcal{P}$-TIME.

   (xix) **T** Every context-free language is in $\mathcal{P}$.

(xx) **T** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a unary ("caveman") numeral.

(xxi) **F** Every bounded function is recursive.

We say that a function f is *bounded* if there is a constant K such that $|f(n)| \le K$ for all n.

(xxii) **O** If L is $\mathcal{NP}$ and also co-$\mathcal{NP}$, then L must be $\mathcal{P}$.

(xxiii) **T** If L is RE (recursively enumerable) and also co-RE, then L must be recursive.

(xxiv) **T** Every language is enumerable.

(xxv) **F** If a language L is undecidable, then there can be no machine that enumerates L.

(xxvi) **T** There exists a mathematical proposition that can be neither proved nor disproved.

(xxvii) **T** There is a non-recursive function which grows faster than any recursive function.

(xxviii) **T** There exists a machine that runs forever and outputs the string of decimal digits of π (the well-known ratio of the circumference of a circle to its diameter).

(xxix) **F** For every real number x, there exists a machine that runs forever and outputs the string of decimal digits of x.

(xxx) **O** There is a polynomial time algorithm which determines whether any two regular expressions are equivalent.

(xxxi) **F** The set of real numbers is enumerable, that is, countable.

(xxxii) **T** The set of equivalence classes of deterministic abstract machines is enumerable, that is, countable.

Recall that two deterministic abstract machines are *equivalent* if, given the same input (possibly empty input), either they both halt or neither halts, and they give the same output (possibly empty output).

2. The handout *finiteAutomata.pdf* contains 16 figures, each accompanied by an exercise. Work the exercises for Figures 1, 2, 3, 5, 10, 15.

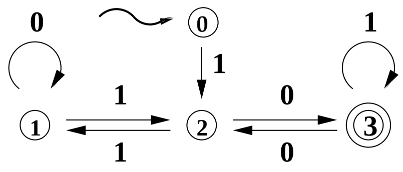

Figure 1: Describe the language accepted by this DFA. The word "numeral" should be in your answer.

The set of all binary numerals for positive integers n such that $n\%3 = 2$

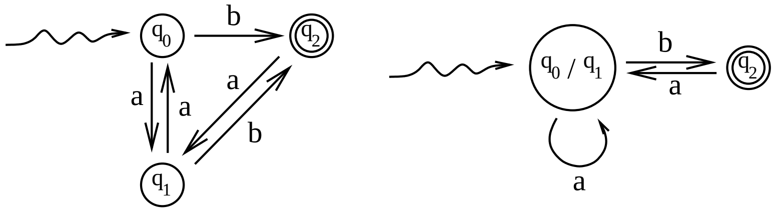

Figure 2: Draw an equivalent minimal DFA

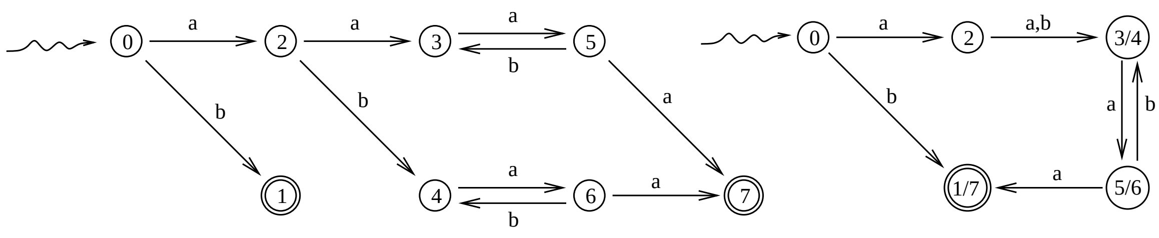

Figure 3: Minimize this DFA.

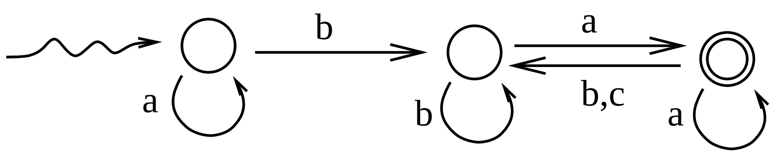

Figure 5: Write a regular expression for the language accepted by this DFA.

$$\mathbf{a^*bb^*a(a + (b + c)b^*a)^*}$$

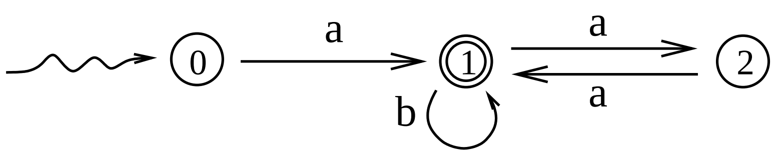

Figure 10: Write a regular expression for the language accepted by this DFA.

$$\mathbf{ab^*(b + aa)^*}$$

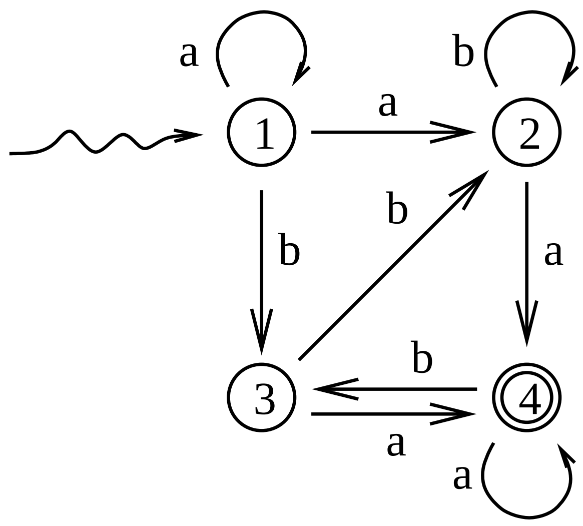

Figure 15: Give a right-linear grammar for the language accepted by this NFA.

Let the variables S, A, B, and C correspond to states 1, 2, 3, 4, respectively.
$S \to aS \,|\, aA \,|\, bB$
$A \to bA \,|\, aC$
$B \to bA \,|\, aC$
$C \to aC \,|\, bB \,|\, \lambda$

3. Please help me make a very important decision! As you know, the question of whether there is a $\mathcal{P}$–TIME algorithm which finds the factors of an integer is one of the most important unsolved problems in computation theory. In fact, the security of RSA encryption, which is used many times every day, depends on the hypothesis that no such algorithm exists. After many years of work, I have found an algorithm for the problem. Here is my code. I have run it, and it works perfectly!

```
void primefactors(int n)
 // lists prime factors of n separated by commas
 {
  int f = 2;
  while (n%f != 0) f++;
   if(f == n) cout << n << endl; // n is prime
   else if(n%f == 0) // f is a prime factor of n
    {
     cout << f << ",";
     primefactors(n/f);
    }
 }
```

You can easily see that the running time of the code is $O(n)$, which is certainly polynomial!

I could either publish this result and become insanely famous, or keep it to myself (I'm sure you won't tell) and become insanely rich breaking RSA encryption for enormous fees. What should I do?

Unfortunately, I had to cancel my reservations for my triumphant world tour. The time complexity of my algorithm is a polynomial function of the input n, but is an exponential function of the number of bits of input, roughly the number of digits of the binary numeral for n.