CSE 311: Foundations of Computing I

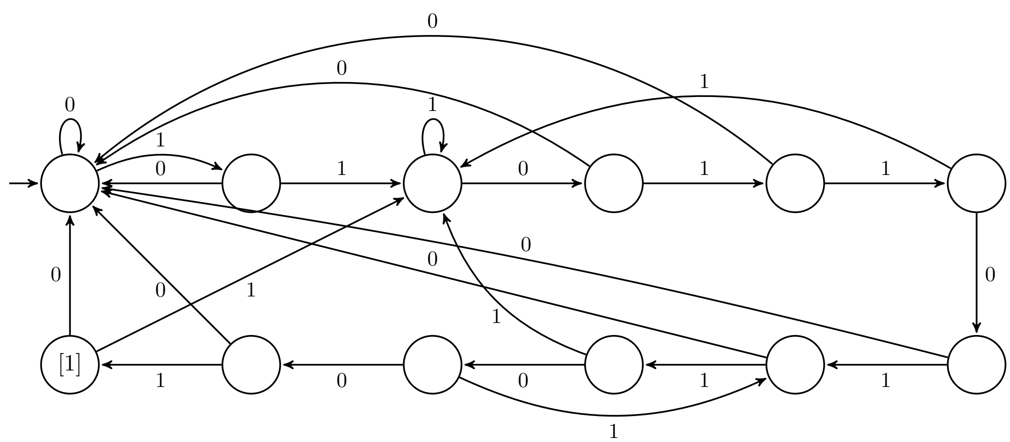
Homework 9 Solutions

1. Pattern Matching [Online] (15 points)

Use the method given in class to design a DFA to determine all occurrences of the string 11011011001 in strings over the alphabet $\{0, 1\}$.

You must submit and check your answers to this question using
https://grinch.cs.washington.edu/cse311/fsm.

Solution:



2. Diagonalization (20 points)

Let B be the set of all infinite binary sequences that are 1 in odd positions, i.e., any string in B is of the form

$$1 * 1 * 1 * 1 * \ldots$$

where we can have 0 or a 1 instead of each $*$. Show that B is uncountable using a proof by diagonalization.

Solution:

We prove by contradiction. Suppose there is a listing of the set L of all infinite binary that are 1 in odd positions, $s_1, s_2, s_3, \ldots$. We construct a string $t \in L$ which is not listed in the sequence. For all i we define the i-th position of t as follows: If i is odd, then we have a 1 in the i-th position; otherwise we let the i-the position of t be the negation of the i-th position of the string $s_{i/2}$.

First of all, since t has a 1 in all odd positions, $t \in L$. Secondly, we show that for all i, $t \neq s_i$. This is because for all i, the bit in the $2i$-th position of t is different from the bit in the $2i$-th position of s_i. Therefore, t is not listed in the sequence which is a contradiction.

3. Countability (20 points)

Complex numbers can be written as $a + bi$ where a, b are real numbers and i is the square root of -1. Show that set R of complex numbers given by

$$R = \{a + bi : a, b \text{ are rational}\}$$

is countable

Solution:

We proved in class that the set of rational numbers is countable; so suppose there is a sequence $x_1, x_2, x_3, \ldots$ that includes all rational numbers. So, we can assume that each number in R can be written as $x_a + ix_b$ for some positive integers a, b. We use a dovetailing idea similar to the one we used to count rational numbers. Consider the following sets

$$\begin{aligned} S_1 &= \{x_1 + ix_1\} \\ S_2 &= \{x_1 + ix_2, x_2 + ix_1\} \\ S_3 &= \{x_1 + ix_3, x_2 + ix_2, x_3 + ix_1\} \\ \cdots \end{aligned}$$

In general, let $S_{k+1} = \{x_1 + ix_k, x_2 + ix_{k-1}, \ldots, x_k + ix_1\}$ be the set of all numbers $x_a + ix_b$ in R such that $a + b = k + 1$. Obviously there are only k numbers in S_{k+1} for all k. Also, note that each number in R is included in one of the sets S_i.

We list all numbers in R in the following order: First we write down numbers in S_1, then we write numbers in S_2 and so on. For each S_i we write numbers in S_i in an increasing order of their real part. Since each S_i has finitely many elements, we will map a finite natural number to any number in R. In particular, for any number $x_a + ix_b$ we map the integer $1 + 2 + \cdots + (a + b - 2) + a$ to $x_a + ix_b$.

4. Irregularity (30 points)

Using the method shown in class prove that that the following languages are not regular.

(a) [15 Points] The set of binary strings of the form $\{0^n 1^m 0^n : m < n\}$.

Solution:

We prove by contradiction. Suppose there is a DFA M that recognizes the langugae $B = \{0^n 1^m 0^n : m < n\}$. We show that M accepts or rejects a string it shouldn't.

Consider the set of half strings $S = \{001, 0001, 00001, \ldots\} = \{0^n 1 : n > 1\}$. Since there are finitely many states in M and S has infinitely many strings, there exists strings $0^a 1, 0^b 1 \in S$ with $a \neq b$ that end in the same state of M.

Now, consider appending 0^a to both strings. Since $0^a 1, 0^b 1$ end in the same state, $0^a 1 0^a$ and $0^b 1 0^a$ also end in the same state, call it q. We show M makes a mistake: Since $0^a 1 \in S$, we have $a > 1$. So, $0^a 1 0^a \in B$, so q must be an accepting state. On the other hand, since $a \neq b$, $0^b 1 0^a \notin B$. But, then M accepts $0^b 1 0^a$ which is a contradiction.

Since M was arbitrary, there is no DFA which recognizes the lanugage B.

(b) [15 Points] The set of strings 0^n where n is a perfect square, i.e., $n = k^2$ for some $k \in \mathbb{N}$.

Solution:

We prove by contradiction. Suppose there is a DFA M that recognizes the language P of all strings 0^n where n is a perfect square. We show that M accepts or rejects a string it shouldn't.

Consider the set of half strings $S = \{0, 00, 000, \ldots\} = \{0^n : n \geq 0\}$. Since there are finitely many states in M and S has infinitely many strings, there exists strings 0^a and 0^b in S with $a \neq b$ that end in the same state of M.

We consider two cases.

Case 1: $a > b$: Consider appending 0^{a^2-b} to both strings. Since $0^a, 0^b$ end in the same state $0^a 0^{a^2-b} = 0^{a^2+a-b}$ and $0^b 0^{a^2-b} = 0^{a^2}$ also end in the same state, call it q. We show M makes a mistake: Since a^2 is perfect square, $0^{a^2} \in B$; so q must be an accepting state. Now, we show $a^2 + a - b$ is not a perfect square. This is because on one hand, since $a > b$

$$a^2 + a - b > a^2$$

On the other hand,

$$a^2 + a - b < a^2 + 2a + 1 = (a+1)^2.$$

Since, there are no perfect squares between a^2, and $(a+1)^2$, $a^2 + a - b$ is not a perfect square. So, $0^{a^2+a-b} \notin B$. But, then M accepts 0^{a^2+a-b} which is a contradiction.

Case 2: $b > a$: This similar to the previous case. By a similar argument we get a contradiction.

Since M was arbitrary, there is no DFA that recognizes the language P.

5. Undecidability (15 points)

Consider the set

$$\mathbf{Prime} = \{(\text{CODE}(\mathbf{P}), \mathbf{x}) : \mathbf{P} \text{ reads } \mathsf{x} \text{ and halts if and only if } \mathsf{x} \text{ is a prime}\}$$

Show that $\mathbf{Prime}$ is undecidable using the fact that the Halting Problem is undecidable.

Solution:

Assume for the sake of contradiction that the set **Prime** is decidable. So, there is a program $\mathbf{I}$(String input, String x) which returns true if and only if $(\text{CODE}(\mathbf{P}), \mathbf{x}) \in \mathbf{Prime}$.

Consider an input $(\text{CODE}(\mathbf{Q}), \mathbf{y})$ to the Halting problem. We construct a program $\mathbf{P}$ such that $\mathbf{P}$ halts on input $\mathbf{2}$ if and only if $\mathbf{Q}$ halts on input $\mathbf{y}$.

Let $\mathbf{P}$ be the program $\mathbf{Q}$ where the input $\mathbf{y}$ is hard-coded. Suppose CODE($\mathbf{Q}$) reads input $\mathbf{y}$ into variable var. To construct CODE($\mathbf{P}$), add a hard-coded assignment statement after $\mathbf{Q}$ reads its input: var $= \mathbf{y}$. In other words, $\mathbf{Q}$ reads its input, and then assumes it is $\mathbf{y}$.

So if we have a program $\mathbf{I}$ to decide $\mathbf{Prime}$ then we can use it as a subroutine as follows to decide the Halting Problem, which we know is impossible: On input CODE($\mathbf{Q}$) and $\mathbf{y}$, produce CODE($\mathbf{P}$). Then run $\mathbf{I}$ on input $(\text{CODE}(\mathbf{P}), 2)$ and output the answer that $\mathbf{I}$ gives. Since $\mathbf{2}$ is a prime, $\mathbf{I}$ must output true if and only if $\mathbf{P}$ halts. But, $\mathbf{P}$ halts if and only if $\mathbf{Q}$ halts on input $\mathbf{y}$. So, $\mathbf{I}$ decides the halting problem which is a contradiction. Therefore, $\mathbf{Prime}$ is undecidable.