COM S 531

Theory of Computation

Name: Monoshi Kumar Roy

Email: monoshi@iastate.edu

Student_id: 375360043

Homework Set 1

Problem 1: Prove that for any two sets, the following two conditions are equivalent

1. There is a function $f: A \xrightarrow{1-1} B$
2. $A = \emptyset$ or there is a function $g: B \xrightarrow{onto} A$

Conclude that, a set A is countable if and only if there is a function $f: A \xrightarrow{1-1} N$

Solution:

First let's proof that $1 \rightarrow 2$.

Case 1: If domain $A = \emptyset$ then the condition hold by Empty function's definition.

Case 2: If $A \neq \emptyset$, a one to one function $f: A \xrightarrow{1-1} B$ is given. So, every $x \in A$ will map to a unique element of co-domain set B. Now, two scenarios can be considered in this case. Firstly, if $f(A) = B$, then the function will become a bijective function. So, an inverse function of f will exist such that $g: B \xrightarrow{onto} A$. In this case, every element of A will have a pre-image in domain of ~~g(B)~~ function g, which is B. In the second scenario, let $f(A) \neq B$, ~~Here $\exists y \in B$ and $\exists y \notin f(A)$~~ which means some elements in B may not have pre-image in A. However we can still design a function $g: B \xrightarrow{onto} A$ such that those elements of B will now have an image in set A. And this function will be onto because every element in A will have at least one pre-image in B.

Now, let's proof (2) $\rightarrow$ (1)

<u>Case 1:</u> If $A = \emptyset$, we can design a one-to-one function by empty function's definition. So, the condition holds.

<u>Case 2:</u> If $A \neq 0$, then there is a function $g: B \xrightarrow{onto} A$ So, every element in A will have at least one pairing pre-image in domain B. Let's assume all these pre-images are unique. Then, g will become a bijective function and there will be an inverse function f such that $f: A \xrightarrow{1-1} B$. Now, let us consider that A has some elements which have more than one pre-image in B. In this case, while ~~definining~~ defining function f, every element $x \in A$ must be mapped to a unique element of B such that $f: A \xrightarrow{1-1} B$.

Now, for the concluding part,

(I) A is countable

(II) There is a function $f: A \xrightarrow{1-1} N$

let us proof that (I) $\rightarrow$ (II)

Given is a set A countable. Then from definition we know that there is a function $g: N \xrightarrow{onto} A$. So, from the previous part's proof, we can conclud that there exists a function $f: A \xrightarrow{1-1} N$

For, (II) $\rightarrow$ (I)

There is a function $f: A \xrightarrow{1-1} N$. From the proof of previous part, we can show that there exists a function $g: N \xrightarrow{onto} A$. So, we can conclude that A is countable.

Problem 2: Let C_n be a set of languages $A \subseteq \{0,1\}^*$ for each $n \in N$. Prove: If each of the sets C_n is countable, then so is the set $\bigcup_{n=0}^{\alpha} C_n$.

Solution: From the given condition, it is true that set of languages C_0, C_1, C_2 ---- are countable individually. We have to prove that the union set of all these countable sets are also countable.

So, to proof $\bigcup_{n=0}^{\alpha} C_n$ is countable, let, K is a natural number whose value can be up to α. Now, we will proof $\bigcup_{n=0}^{K} C_n$ is countable by using induction method.

Basis:

Suppose, ~~n=0~~ $K=0$

$$\bigcup_{n=0}^{0} C_n = C_0$$

Which is a countable set.

~~when n=1~~, when $K=1$

$$\bigcup_{n=0}^{1} C_n = C_0 \cup C_1$$ which is countable.

[Lemma 8]

Hypothesis:

Let, $K=m$

So, $\bigcup_{n=0}^{m} C_n = C_0 \cup C_1 \cup C_2 \cup C_3 \cdots \cup C_m$

$= X$

We are ~~assummin~~ assuming the resultant of union of sets $C_0, C_1, C_2 \cdots, C_m$ is X which is countable.

Inductive:

we have to proof that $\bigcup_{n=0}^{K} C_n$ is countable for $K=m+1$.

$\cancel{n=0}$ $\bigcup_{n=0}^{m+1} C_n = C_0 \cup C_1 \cup C_2 \cdots \cup C_m \cup C_{m+1}$

$= (C_0 \cup C_1 \cup C_2 \cup \cdots \cup C_m) \cup C_{m+1}$

$= X \cup C_{m+1}$

Here X and C_{m+1} are countable sets individually. So, from Lemma 8, their union will also be countable.

So, it is proved that $\bigcup_{n=0}^{\infty} C_n$ is countable.

[Proved]

Problem 3: Design a turing machine $M = (Q, \Sigma, \Delta, \Gamma, \delta, s, H)$ that decides the language $\{0^n 1 0^{3n} \mid n \in N\}$.

Solution:

Let us design a turing machine

~~M=~~

$$M = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}, \{0,1\}, \{0,1,X,Y,Z,B\}, \delta, q_0, B, \{q_7\})$$

here,

set of states $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$

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

Tape symbols $\Gamma = \{0,1,X,Y,Z,B\}$

δ is the transition function.

Starting state $\{q_0\}$

Blank Symbol B

Accepting state $\{q_7\}$

Here is the designed transition function for the turing machine.

State	0	1	X	Z	Y	B
q_0	(q_1, X, R)	$(q_6, 1, R)$	–	(q_6, Z, R)	–	–
q_1	–	(q_2, Z, R)	–	–	–	–
q_2	(q_3, Y, R)	–	–	–	–	–
q_3	(q_4, Y, R)	–	–	–	–	–
q_4	(q_5, Y, L)	–	–	–	–	–
q_5	–	–	(q_0, X, R)	(q_5, Z, L)	(q_5, Y, L)	–
q_6	–	–	–	–	(q_6, Y, R)	(q_7, B, R)
q_7	–	–	–	–	–	–

For example,

Assume string 01000.

This string should be accepted by the turing machine.

I would like to demonstrate the moves of the turing machine by "$\vdash$", and tape head by "$\downarrow$" symbol. The turing machine here is considered to be infinite to the right side.

$$q_0\underset{\downarrow}{0}1000BBB \ldots \vdash Xq_1\underset{\downarrow}{1}00BB \vdash XZq_2\underset{\downarrow}{0}00BB$$

$$\vdash XZYq_3\underset{\downarrow}{0}0BB \vdash XZYYq_4\underset{\downarrow}{0}BB \vdash XZYq_5\underset{\downarrow}{Y}YBB \ldots$$

$$\vdash \cancel{XZY}\, XZq_5\underset{\downarrow}{Y}YYBB \vdash Xq_5\underset{\downarrow}{Z}YYYBB \vdash q_5\underset{\downarrow}{X}ZYYYBB$$

P.T.O

Continue ——

$q_5 X Z Y Y Y B B \vdash X q_0 Z Y Y Y B B \vdash X Z q_6 Y Y Y B B$ ---

$\vdash X Z Y q_6 Y Y B B \vdash X Z Y Y q_6 Y B B \vdash X Z Y Y Y q_6 B B$ ---

$\vdash X Z Y Y B q_7 B$ (Accepted)

Again, let's assume a string 010. This string must not be accepted by the turing machine.

$q_0 010BB \vdash X q_1 10BB \vdash X Z q_2 0BB \vdash X Z Y q_3 BB$

Now, we found that, the current state was q_3 and from the transition function table, we can notice that there is no transition on tape symbol B from state q_3. So, the turing machine dies and doesn't accept this input.

When n=0 (as mentioned in the class that set of natural numbers starts from 0), then the input string will become 1, no zero will precede or follow 1. This turing Machine will also accept this input.

Problem 4: Prove that a language $A \subseteq \{0,1\}^*$ is c.e. if and only if there is a computable partial function $f: \subseteq \{0,1\}^* \longrightarrow \{0,1\}^*$ such that $\text{dom} f = A$.

Solution:

(I) $A \subseteq \{0,1\}^*$ is computably enumerable

(II) There is a computable partial function $f: \subseteq \{0,1\}^* \rightarrow \{0,1\}^*$ such that $\text{dom} f = A$

First, let us proof (I) $\rightarrow$ (II)

From (I), it is given that A is computably enumerable (c.e.). So, there is a TM M_A such that

$$L(M_A) = A.$$

So, for every string that belongs to language A, TM M_A will halt. This characteristics of the TM M_A can be simulated by a computable partial function f. Let this function can be computed by another TM M_f by defining the following algorithm.

M_f (Takes any strings $s \in \{0,1\}^*$)

1. For any string $s \in \{0,1\}^*$
2. Run the string on M_A
3. If M_A halts then
4. $s \in A$ and output s
5. Else undefined
6. End

Now, from the given algorithm, it is evident that function f will have the domain of $\{0,1\}^*$ but it will be mapping only strings $s \in A$ to it's co-domain. So, this function will be undefined for the $\{0,1\}^* \setminus A$ inputs, thus making it a partial computable function which can be written as:

$$f : \subseteq \{0,1\}^* \longrightarrow \{0,1\}^* \text{ such that } \mathrm{dom} f = A$$

Now, let's consider ~~2 → 1~~ (ii) → (i)

In this case a partial computable function f is given such that

$$f : \subseteq \{0,1\}^* \to \{0,1\}^* \text{ such that } \mathrm{dom} f = A.$$

As, f is a partial computable function, for the elements that belongs to A are defined by f. The function cannot map the values that are in $\{0,1\}^* \setminus A$. Let TM M_f emulates the given function. Then, we can make a TM M_A by following algorithm

<u>M_A (Takes any string $s \in \{0,1\}^*$)</u>

1. Run M_f on any input string s
2. If M_f halts on input s then
3. $s \in A$ and output s.
4. Else undefined
5. Exit

So, The turing machine M_A will halt only if M_f halts. M_f will halt only for it's domain A. So, M_A will accept only inputs that belong to A. Other inputs are undefined and TM M_A will loop forever. So, we can conclude that A is c.e.

Problem 5 Prove that, for every language $A \subseteq \{0,1\}^*$, the following conditions are equivalent

1. A is c.e.
2. There is a computable partial function $f: \subseteq N \to \{0,1\}^*$ such that range $f = A$
3. $A = \emptyset$ or there is a computable function $f: N \to \{0,1\}^*$ such that range $f = A$
4. A is finite or there is a computable function $f: N \xrightarrow{1-1} \{0,1\}^*$ such that range $f = A$.

Solution Let us proof ~~1→4→3→2→1~~ 1→2→3→4→1

Let's proof 1→2.

Here given A is c.e.. If A is finite, then the condition holds. Let, A is infinite, then there exists a TM M_A such that $L(M_A) = A$. Let function f of statement (2) can be simulated by TM M_2.

Now let's try to implement M_2 using the properties of M_A.

Algorithm for M_2 (Input $n \in N$)

1. Generate all strings of length 0 to length n.
2. Initialize $i := 0$
3. While $i \leq n$
4. Run M_A on all i-length strings say str_i
5. If M_A accepts str_i
6. output str_i
7. Else undefined
8. ~~i:=i+1~~ $i := i+1$
9. End

Here, the algorithm for TM M_2 is matching the strings of different length (up to n) by running ~~in o~~ those on the turing machine M_A. The strings that are in language A will be mapped to output and for other strings this mapping is undefined. So, this given algorithm is working like a partial computable function $f: \subseteq N \rightarrow \{0,1\}^*$, such that Range $f = A$.

(2) $\longrightarrow$ (3)

Given a partial computable function $f: \subseteq N \to \{0,1\}^*$ such that range $f = A$. We need to show that $A = \phi$ or there is a computable function $f: N \to \{0,1\}^*$ such that range $f = A$. Let statement (2)'s function can be emulated by TM M_2 and statement (3)'s TM is M_3. Now let's try to develop TM M_3 by using the property of M_2. For $A = \phi$, the condition holds. Now, if A is infinite, algorithm for M_3 is given below -

<u>Algorithm for M_3 (input $n \in N$)</u>

1. $n \in N$ and $n = 0, 1, 2, 3, \ldots$
2. Initialize $i := 0$
3. while $i \leq n$
4. If M_2 is defined on i
5. output $f(i)$
6. Else if M_2 is undefined on i
7. output
8. ~~i := i+1~~ $i := i+1$
9. End

(3) $\longrightarrow$ (4)

Here given $A = \phi$ or there is a computable function $f: N \longrightarrow \{0,1\}^*$ such that range $f = A$. When $A = \phi$, then A is finite and the condition holds. or let A is not empty, then there will be a function $f: N \xrightarrow{1-1} \{0,1\}^*$ such that range $f = A$. Let function of statement 3 can be enumerated by TM M_3 and statement 4's function can be emulated by TM M_4. Now let's try to develop M_4's algorithm using M_3.

<u>Algorithm for M_4 (input $n \in N$)</u>

1. $n \in N$ and $n = 0, 1, 2, 3 - \text{——}$
2. initialize $i := 0$
3. while $i \leq n$
4. if $f_3(i) == f_3(i+1)$
5. output $f_3(i)$
6. $i := i+1$
7. else output $f_3(i)$
8. $i := i+1$

Algorithm for M_4 (input n)

1. $n \in N$ and $n = 0, 1, 2, \ldots$
2. Initialize $i := 0$
3. while $i \leq n-1$
4. Initialize flag := 0
5. Run i on M_3
6. for $j := i+1$ to n
7. ~~if~~ Run j on M_3
8. if $f(i) == f(j)$
9. flag := 1
10. break for loop
11. if flag == 0
12. output $f(i)$
13. else
14. Go to step 15. // Repetitive value for $f(i)$ exits, discarding those values, only taking one
15. $i := i+1$
16. End

In this ~~fun~~ given algorithm, we are checking for repetitive range values for input i. We know that from the definition of One to One function

that ~~no~~ every element in the domain set must be mapped to a unique element. This given algorithm is checking for unique elements in the range set for an input value, thus making it a one to one function. So, for given (3) a function $f: N \xrightarrow{1-1} \{0,1\}^*$ where range $f = A$ holds.

Now, finally prove (4) $\rightarrow$ (1)

Given A is finite or there is a function $f: N \xrightarrow{1-1} \{0,1\}^*$ such that range $f = A$. If A is finite, then A is computably enumerable. For A being infinite, let M_f be the TM that imitates the given function f. We have to design a TM M_A such that $L(M_A) = A$.

<u>Algorithm for M_A (input any string str)</u>

1. $str \in \{0,1\}^*$
2. n := length of str
3. for ~~i=0,1,2,3~~ $i = 0, 1, 2, 3 \ldots$
4. Run M_f on i
5. if ~~f(i) == str~~ f(i) == str
6. Accept and halt
7. Else undefined
8. End

Here, The turing machine M_A is taking input as a string. TM M_A will output or range of the function f here is A. So, if there is a match between the given input string and the ~~do~~ range of f, M_A will accept the string, otherwise the string will not be a member of language A and it will be undefined. So, we can conclude that $L(M_A) = A$.

So, we have proved that (1), (2), (3), (4) statements are equivalent as we proved that (1) → (2), (2) → (3), (3) → (4) and (4) → (1) conditions are equivalent seperately.

Problem 6: Prove that a language $A \subseteq \{0,1\}^*$ is decidable if and only if A is finite or there is a computable function $f: N \rightarrow \{0,1\}^*$ such that range $f = A$, and for every $f(n)$ appears before $f(n+1)$ in the standard enumeration of $\{0,1\}^*$.

Solution:

(I) A is decidable

(II) A is finite or there is a computable function $f: N \longrightarrow \{0,1\}^*$ such that range $f = A$, and for every $f(n)$ appears before $f(n+1)$ in the standard enumeration of $\{0,1\}^*$

First let's prove (I) $\longrightarrow$ (II)

A is decidable. If A is finite, the condition holds. Now, if A is infinite, then we can design a TM M_A that will accept every string that belongs to A or reject otherwise. Now, we can simulate the given function f in (II) by using the TM M_A. Let the Turing machine for f is M_f and it's algorithm

is given here

Algorithm for M_f (Given an input n)

1. input $n \in N$
2. Initialize $i := 0$
3. while $i \leq n$
4. str$_i$ is in the standard order of $\{0,1\}^*$
5. ~~If M_A accepts~~
5. Run str$_i$ on M_A
6. If M_A accepts str$_i$ AND $i = n$
7. output str$_i$
8. $i := i+1$
9. End

From the given algorithm, we can see that whenever M_A is accepting the n-th string, it is giving it as output.

~~So, the output of the~~

So, the range of function f would be A. And we are getting decision for n-th string before (n+1)th string, which can be said as f(n) comes before f(n+1)

Now, let's prove (II) → (I)
For, A being empty, the condition holds
Here we are given a computable function $f: N \to \{0,1\}^*$ such that range $f = A$ and for every $n \in N$, $f(n)$ appears before $f(n+1)$ in the standard enumeration of $\{0,1\}^*$.

Let's assume the TM for the given function is M_f.
Now, we will have to develop a turing machine for language A by using M_f.

<u>Algorithm for M_A</u> (Input a string str)

1. $n \in N$, $str \in \{0,1\}^*$
2. For $n = 0, 1, 2, \ldots$
3. Run M_f on n
4. If $f(n) == str$
5. Accept and halt
6. Else if $f(n) > str$
7. Reject and halt
8. End

In the function, every $f(n)$ comes before $f(n+1)$.
So, when $f(n) > str$, the string is obviously not in A, so it rejects. If str is a member of A, then

$f(n) = str$ holds and the string is accepted. So, from the discussion above, we can ~~co~~ conclude that A is decidable.

Problem 7: Let ϕ be the following statement. For every computable function $f: \{0,1\}^* \to \{0,1\}^*$, there is a computable function $g: \{0,1\}^* \to \{0,1\}^*$, such that for all $x \in \{0,1\}^*$, $g(f(x)) = x$

1. Prove that ϕ is false by giving a counterexample.
2. Strengthen the hypothesis of ϕ just enough to obtain a statement ϕ' that is true.
3. Prove your statement ϕ'.

Solution: 1. The given problem ϕ states that for every computable function f, there exists a function g such that g maps back every range of f to it's domain set. This statement is not true for every function. Suppose, $f(x) = |x|$. For this function, there doesn't exist a function g such that every element of domain set can be retrived. Because for both $x=2$ and $x=-2$, the $f(2)$ and $f(-2)$ will be 2.

2. The given statement ϕ is about the existence of inverse function such that the range of a function can be mapped back again to it's domain. So, if we impose the function f to be bijective or one to one correspondence, then the new statement would be true.

3. In statement ϕ', we have considered the function f to be a bijective function. Only then there will exist a function like g that will behave like an inverse function of f.

Here given a function $f: \{0,1\}^* \longrightarrow \{0,1\}^*$ which is bijective. That means for every $x \in \text{dom} f$ will have one unique image in $y \in \text{range} f$ (Property for one to one). And by onto property every image in range f will have at most one preimage in dom f. So, if ~~the~~ these two conditions are satisfied, there will be an inverse function g such that $g: \{0,1\}^* \longrightarrow \{0,1\}^*$ and for $x \in \{0,1\}^*$, $g(f(x)) = x$

For $g(f(x)) = x$, let, x be an element in dom f.
Then $f(x)$ is defined to be a unique element mapped by function f. Since $f(x)$ is mapped backed to dom f by ~~or~~ at most one element (by onto property), $g(f(x)) = x$.
Similarly, it can be also shown that $f(g(b)) = b$.
So, function f being a bijective function will satisfy the ~~of~~ statement.

Problem 8: Prove that every infinite c.e language $A \subseteq \{0,1\}^*$ has an infinite decidable subset.

Solution:

Here given that language A is computably enumerable. So, there is a Turing machine M_A such that $L(M_A) = A$. Now, if language A has a infinite decidable subset, to be decidable, ~~then~~ there must be turing machine M_S, such that it can enumerate all the strings of the decidable set in the standard order of $\{0,1\}^*$ or in the lexicographic order. (Proof of problem 6)

Algorithm for M_S

1. Run M_A to generate all strings that is in A.
2. prev_value = " "
3. For every string generated by M_A denoted by str
4. If str > prev_value
5. output str
6. prev_value = str
7. End

From the given algorithm, it is evident that $L(M_S) \subseteq L(M_A)$. And, from previous proof (Number 6), it can be said that the ~~lang~~ strings that M_S taking are decidable. So, it is proved that language A has a decidable subset.

Now, let's ~~proof~~ prove that the subset is also infinite. We will prove it by contradiction. Suppose the decidable subset is finite. Then we can conclude that the prev_value variable will hold the largest value of $\{0,1\}^*$ standard enumeration not only of the decidable subset but also for the given language A, which will make the c.e language also finite. So, it can be concluded that the decidable subset will be infinite.