INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD
Department of Computer Science and Engineering
CS5616 Computational Complexity
January – May 2024

IIT PALAKKAD

Problem Set – 4
Name: Neeraj Krishna N
Roll no: 112101033

Total Points – 50
Given on 15 Mar
Due on 5 Apr

Instructions

- Use of resources other than class notes and references is forbidden.
- Collaboration is not allowed. Credit will be given for attempts and partial answers.

1. (10 points) [**Resource ?**]

   Consider the following definitions where M is a valid Turing machine encoding and $x \in \Sigma^*$. We use $\perp$ for value being undefined. Check if the follows are a resource using Blum's axioms. In case they don't, give a formal argument.
   (*Hint*: Decidability of the policing language !)

   (a) (5 points) (Head turns)
   $$h(M,x) = \begin{cases} \# \text{ turns that input head of } M \text{ makes on } x \text{ before halting} & M \text{ halts on } x \\ \perp & \text{otherwise} \end{cases}$$

   (b) (5 points) (Count) Let q_0 be a state in the finite control of M.
   $$c(M,x) = \begin{cases} \# \text{ of times } M \text{ visits a state } q_0 \text{ until it accepts or rejects} & M \text{ halts on } x \\ \perp & \text{otherwise} \end{cases}$$

Solution:

(a) The policing language is $L_P = \{(M,x,t) \mid h(M,x) \le t \text{ if } M \text{ halts on } x\}$ Let the machine deciding L_P be M_P

We will show that $\overline{\mathsf{HP}} \le_m L_P$

Reduction machine on input (M,x) an instance of $\overline{\mathsf{HP}}$ outputs $(N,\epsilon,0)$ where N is a 2 tape turing machine, where tape 1 is the input tape and tape 2 is the work tape
Description of N is as follows
" On input y

1. Run M on x using the work tape alone and for each step of M on x, move the input tape head towards the right by 1 cell
2. If M halts on x, move the input tape head towards left by 1 cell and halt

"

$(M, x) \notin \overline{\mathsf{HP}}$
$\Rightarrow M$ halts on x
$\Rightarrow N$ on any input (in particular ϵ), turns the head once $\Rightarrow h(N, \epsilon) = 1 \Rightarrow M_P$ rejects $(N, \epsilon, 0) \Rightarrow (N, \epsilon, 0) \notin L_P$

$(M, x) \in \overline{HP}$
$\Rightarrow M$ loops on x
$\Rightarrow$Input tape head of N always moves right, never making a head turn
$\Rightarrow h(N, x) = 0$
$\Rightarrow (N, \epsilon, 0) \in L_P$

Thus $\overline{\mathsf{HP}} \leq_m L_P$ and since we know $\overline{\mathsf{HP}}$ is undecidable, there does not exist a policing language for h defined in the above manner.

The below is another attempt at showing that $h(M, x)$ is not decidable
Suppose h is decidable by a Total turing machine X

Consider the following language described via a 2-tape (first is the input tape and the second is the work tape) turing machine M_D
$M_D =$
" On input x

1. Copy the input to the work tape and move input tape back to left end marker
2. Compute $k \leftarrow h(M_x, x)$ where M_x is the turing machine corresponding to the string x (Note that computation is done on the work tape nothing is being done using the input tape now)
3. If k is a finite number
   Move the input tape back and forth for infinite number of times (i.e make the machine loop by moving the input tape head back and forth)

"

Let z be the string representation of the machine M_D

Now what will be $h(M_z, z)$?

$h(M_z, z) = k$ for some finite k
$\Rightarrow$In step 2 of the turing machine N, we will get $h(M_z, z)$ as k and in step 3, N will move the input tape back and forth infinite times
$\Rightarrow N$ on input z loops
$\Rightarrow h(N, z) = \perp$
$\Rightarrow h(M_z, z) = \perp$ since N and M_z are the same

$h(M_z, z) = \perp$
$\Rightarrow M_z$ on z loops
$\Rightarrow$Step 3 would never halt

$\Rightarrow$Only place where input head turned was in step 1 for copying the input to the worktape
$\Rightarrow$number of turns, input tape head of N made on z is 1
$\Rightarrow h(N, z) = 1$
$\Rightarrow h(M_z, z) = k$ where $k = 1$ (since M_z and N are the same)

Hence we get
$h(M_z, z) = k$ for some finite $k \Leftrightarrow h(M_z, z) = \perp$ which is a contradiction
Thus $h(M, x)$ is not computable

(b) Policing language is $L_p = \{(M, x, d) \mid c(M, x) \leq d, M \text{ halts on } x\}$
Suppose this is decidable via a turing machine M_p

We will show a reduction from $\overline{\mathsf{FIN}}$ to L_p

Reduction machine on input (M, x) an instance of $\overline{\mathsf{FIN}}$ outputs $(M', x, 0)$
We will consider this count with respect to the terminal state q_t of M'
The description of M' is as follows
“
On input y,

1. Run M on x
2. If M halts on x, enter accept state and halt

“

To prove $(M, x) \in \overline{\mathsf{FIN}} \Leftrightarrow M' \in L_p$

$(M, x) \in \overline{\mathsf{FIN}}$
$\Rightarrow M$ does not halts on x
$\Rightarrow M$ enters accept state 0 times
$\Rightarrow M_p$ accepts $(M', x, 0)$

$(M, x) \notin \overline{\mathsf{FIN}}$
$\Rightarrow M$ halts on x
$\Rightarrow M$ enters accept state 1 time
$\Rightarrow M_p$ rejects $(M', x, 0)$

Thus if L_p is decidable, then so is $\overline{\mathsf{FIN}}$ which we know is not the case
Hence Policing machine does not exist for L_p
$\Rightarrow c(M, x)$ **is not a resource**

2. (10 points) [**Oblivious Turing machines**]

   A Turing machine M is *oblivious* if for every input $x \in \Sigma^*$ and every $i \in \mathbb{N}$, the location of $M's$ head at the i^{th} step of execution is only a function of $|x|$ and i. In other words, the head movement does not depend on x. For a time constructible $t(n)$ show that if $L \in \mathsf{DTIME}(t(n))$, then there is a two tape oblivious Turing machine that decides L in $O(t(n)^2)$ time.

Solution: We have to prove that $\forall L \in \mathsf{DTIME}(t(n)), \exists$ an oblivious Turing machine with two tapes that decides L in $O(t(n)^2)$ time
Let M_L be a k tape Turing machine which decides L in $t(n)$ time

Let N be the oblivious Turing machine with 2 tapes.
Tape 1 : Simulates all the k-tapes of M_L using k tracks and appropriate symbols, for all a belonging to the symbols of M, we have a and $\hat{a}$ as symbols for our oblivious turing machine where $\hat{a}$ represents the head in the corresponding tape
Tape 2 : Maintains a counter (which will be described below)

Description of N is as follows : " On input x,

1. Compute $k \leftarrow t(n)$ and store it in tape 2 (takes $t(n)$ time since it's time constructible)
2. For each step of M say, ith step,
   (a) Move the head from left end marker to the ith cell, the actions at each step is as follows
   Suppose the head of N is at jth position ($j \le i$)
   For the nth tape
      i. Case 1 : $\delta_{M_L}(p, a, n) = (q, b, R)$ (n represents the nth tape in M_L) : Change the symbol in the nth track and mark the right symbol with a hat
      ii. Case 2 : $\delta_{M_L}(p, a, n) = (q, b, L)$. Continue

   (b) Move the head from ith cell to the left end marker, the actions at each step is as follows
   Suppose the head of N is at jth position ($j \le i$)
   For the nth tape
      i. Case 1 : $\delta_{M_L}(p, a, n) = (q, b, R)$: Continue
      ii. Case 2 : $\delta_{M_L}(p, a, n) = (q, b, L)$: Change the symbol in the nth track and mark the left symbol with a hat

   (N needs to remember the positions of the hatted symbols in its finite control)

"

We can see that this machine is just simulating M_L, Hence $L(N) = L(M_L) = L$
For i step of M_L, N takes i steps to go to ith cell and i steps to come back (in total $2i$ steps)
The reason is that M_L can only modify atmost i cells in i steps, So it's enough to just go till ith cell and come back

Therefore the total number of steps taken by N to decide L is

$$\sum_{i=0}^{t(n)} 2i = t(n) * (t(n) + 1) \qquad (1)$$

$$= O(t(n)^2) \qquad (2)$$

Claim 1. *N is oblivious*

Proof. For each step i of M_L, N moves till ith cell and comes back
$\Rightarrow$Corresponding to ith configuration of M_L, there are $2i$ configurations for N,
Let us write the configurations of N in serial order starting from $i = 0$, i.e for ith step of M_L we write down configurations of N in serial order
If we index this sequence, what we get is a bijection from natural numbers to the configurations of N and this bijection does not depend on x.
Hence N is oblivious □

3. (15 points) [**More on crossing sequences**]

In class, we saw $o(\log \log n)$ space restricted Turing machines must be regular. Here, we show similar results for time restricted machines.

(a) (5 points) Show that there exists a non-regular language that can be accepted by a one tape $O(n \log n)$ time Turing machine.

(b) (10 points) Show that any language accepted by a one tape deterministic Turing machine in time $o(n \log n)$ must be regular.

Solution:

(a) Let the language L be

$L = \{0^k 1^k \mid k \geq 0\}$

Input length is $n = 2k$

First, we can check if the input is of the form 0^*1^*, by using a DFA, which can be encoded in the finite control of the Turing machine

We can check if the number of 0's in first half of the string and number of 1's in the second part of the string using the turing machine and a DFA

The below is the description of the turing machine M which accepts L in $O(n \log n)$

$M =$
" On input x,

1. Check if it is of the form 0^*1^* using the DFA encoded in the finite control. If not, **reject**.

2. Move the input tape from left end marker towards the right as long as it only sees zero. While moving the head, Use a parity DFA to find if the number of zeros is even or odd and save it in register.
3. Once we reach 1, continue moving towards right, but this time restart the parity DFA and find the parity of number of 1's. And if this is not same as parity of number of zeros which was saved before, **reject**
4. Do the following steps as long as possible
   (a) Move the tape head from left to right and strike every alternate zeros and once we reach one, continue moving towards right, but strike off every alternating zero
   (b) Move the tape head from right to left and strike every alternate 1's and once we reach a zero, continue moving towards left
   (c) If in some run (either left to right or right to left), if we have not deleted 0, and we deleted some 1's, **reject** (since it means more number of 0's than 1's). similarly if we have not deleted 1, and we deleted some 0's, **reject** (since it means more number of 1's than 0's
   (d) If the tape becomes empty, **accept**

"

Claim 2. $L(N) = L$

Proof. N rejects every string of the form other than the regular expression 0*1*
N rejects every string where the number of 1's $\mod 2$ is not equal to number of 0's $\mod 2$
After step 3, we will be sure that either number of 0's and 1's are both odd or both even and it is of the form 0*1*. Now, we are moving the input head left to right and right to left repeatedly and deleting all alternate symbols of the same kind (in every run, either left to right or right to left).
Let m_i denote the number of symbols in the tape (which are not deleted) in the ith iteration (one iteration is one run of the input tape head either from left to right or right to left, $\Rightarrow$All left to right runs are odd iterations and right to left runs are even iterations in this case)
$m_0 = n$
At every iteration, half of the symbols (0 or 1) of the then content of the tape will get deleted
$m_1 = \frac{n}{2}$
$m_2 = \frac{n}{2^2}$
$\vdots$
After the final iteration, there are 3 cases

1. Tape contains only 1's $\Rightarrow$input had more 1's than 0's $\Rightarrow$we reject

2. Tape contains only 0's $\Rightarrow$input had more 0's than 1's $\Rightarrow$we reject
3. No content in the tape $\Rightarrow$input had equal number of 1's and 0's $\Rightarrow$we accept

Thus $L(N) = L$ □

Claim 3. *Running time of N is $O(n \log n)$*

Proof. Not that every run of the tape head, time taken is n steps and the overall running time would be number of runs * n steps
Number of runs would be atmost $\log n$, since in each run m_i reduces by half till the tape becomes empty or the N rejects
Therefore the overall running time is $O(n \log n)$ □

(b) We need to prove that any turing machine which accepts a non regular language will need at least $\Omega(n \log n)$ time

Claim 4. *If time is bounded by a finite number k, then the turing machine can only accept regular languages*

Proof. If time is bounded by a finite number k, the number of possible runs of the turing machine is N^k where N is the total number of possible configurations
$N = |Q| \times s(n) \times d^{s(n)}$, where $s(n)$ is the space used, but note that any turing machine running in k time can only use atmost k cells. hence $s(n)$ is bounded by k. Therefore the total number of runs is finite which can be stored in the finite control of the turing machine itself and we don't require a work tape
Hence, the Turing machine can be simulated by a 2 way DFA, and thus it is proven that any turing machine whose time is bounded by a fixed finite number for all inputs, accept only regular language □

Therefore, without loss of generality, suppose a turing machine accepts a non regular language, For all k, there exists an input x which uses atleast k time. Let x_k^* denote the minimum length string, which uses atleast k time. Let C be the crossing sequences of configurations. The number of possible total number of configurations can be bounded as a function of $|Q|, d, t(n)$, where because in $t(n)$ time, atmost $d * t(n)$ cells can be used (d-tape turing machine)

4. (15 points) [**Definition of space**]

In this question, we explain why the definition of space complexity counts the cells that the machine scans and not just the ones that are written. Let $L = \{w\#w \mid w \in \{0,1\}^*\}$.

(a) (5 points) Show that $L \in \mathsf{DSPACE}(\log n)$.

(b) (10 points) Show that there exists a non-determinisitic Turing machine that writes only on cell to the right of left end marker symbol of its worktape and accepts $\overline{L}$. The machine may scan any number of worktape cells without writing on them.

Solution:

(a) If input is not of the form $\Sigma^*\#\Sigma^*$, reject, which can be done using a DFA, which can be encoded in the finite control of the Turing machine

For inputs of the form, $x\#y$ check if $|x| = |y|$ using binary counters. Maximum value of the binary counter is n, which can be represented in $\log(n)$ space

2 tape turing machine accepting L is as follows
$M =$
" On input z,

1. Scan the entire input from left to right(in the read only tape), and then come back to left end marker while simulating the DFA which accepts $\Sigma^*\#\Sigma^*$. If DFA rejects, reject z
2. Maintain a counter, called i initialized to 0.
3. Increment the i and go to the ith cell and store it in the register. Go to the ith cell after the # and check if it is equal to the stored value. If not reject.
4. Go back to left end marker and increment i and repeat this process.
5. At the end when we encounter # in ith cell for some i, the corresponding cell i.e i cells after the # must be blank symbol, else reject
6. Else accept

"

We are only maintaining a counter which uses $\log n$ space. Hence $L \in \mathsf{DSPACE}(\log n)$

(b) If input is not of the form $\Sigma^*\#\Sigma^*$, accept, which can be done using a DFA, which can be encoded in the finite control of the Turing machine

The description of turing machine N (non deterministic) is as follows

$N =$ " On input z,

1. Scan the entire input from left to right(in the read only tape), and then come back to left end marker while simulating the DFA which accepts $\Sigma^*\#\Sigma^*$. If DFA rejects, accept z
2. Now input is of the form $z = x\#y$
3. Move input tape head and working tape head till simultaneously till the input tape head reaches the end of input

4. Now move the input tape head back to #
5. Move input tape head and working tape head towards left till input reaches left end marker
6. If $|x| = |y|$, then working tape head would have reached #. Hence if working tape head is at a cell having some other symbol, Accept (since $|x|! = |y| \Rightarrow x\#y \in \overline{L}$
7. Now we have verified $z = x\#y$ and $|x| = |y|$, if $|x|! = |y|$, there exists a position i wherein $x_i! = y_i$.
8. Move the input tape head and working tape head simultaneously till input tape head reaches #. After that move back the input tape head to left end marker.
9. Now input tape head is at left end marker and working tape head is at #.
10. Move the input tape head and working tape head towards right simultaneously.
11. Non deterministically choose when to stop
12. if the symbol in cell at the input tape head is not the same as symbol in the cell at the work tape head, Accept (since we have found a position at which both strings x and y differ)

“

The claim N accepts $\overline{L}$ is immediate (every step in the description of N has the corresponding reason)
Note that we are not in anyways using a counter to track where we are. We are only using the heads of 2 tapes in a certain way so as to do the same job as counter. And thus we are not using any space for writing