CS6014-2018 : Computability and Comp lexity — Max Marks : 42

Problem Set #3 — Due on : Oct 14, 22:00 (hard)

- All submissions must be in PDF form produced using LaTeX. Use the same LaTeXfile (from which this pdf file was produced) to add your information and answers.
- You are not expected to take ideas from sources other than textbooks and your classmates. And all collaborations should be explicitly cited in the solution that you submit.

Name : Rachit Garg — Roll No: CS14B050

1. (16 points) **(Computational Resources)** Check whether the following functions qualifies as *computational resources*. Assume that all of them are defined as $\perp$ if and only if M does not halt on x. Write down detailed arguments.

   (a) INK(M, x) : The total number of times (over all cells) that the machine M on input x rewrites symbols (that is, writes a different symbol to the cell than what it read from that cell) on the tape before halting.

   (b) EVEN-STATES(M, x): Number of even numbered states that the machine M visits on input x until it halts.

   (c) HEADTURNS(M, x) : Number of heard turns that the maching M's tape makes on input x until it halts.

   (d) ENERGY(M, x) : Number of 1s written to the tape when M is run on input x until the machine halt.

Solution:

(a) *Proof.* INK(M, x) is a computational resource.

The first condition of Blum's Axiom is true from definition, i.e. the function is defined if M halts on x i.e. there is a unique number that denoted the total no of distinct writes on the tape after the machine halts. The only if part is true as we defined the function to output $\perp$ if the machine does not halt.

The second condition is true as $Police_{Ink}$ is decidable. This is because as seen in class in the example of bounding the work tape, here too we can bound the work tape. If we wish to check that INK$(M, x) \leq k$, our work tape must be bound by $(|x| + k)$ as otherwise the symbol we rewrite will exceed k. Note that we do not get any useful information if we keep on copying the empty symbol at the end of the tape. If we are going beyond the length $(|x| + k)$, we must have performed k writes to reach that stage. (We can actually define our transition function such that the tape alphabet that gets written, does not have the empty tape symbol and we still don't loose on any computation). Since now the work tape is bounded we have a similar proof as seen in class, we can now bound the number of different configurations by $|Q| \times n \times \Gamma^{n+k} \times (n+k)$, where $n = |x|$. We can now keep subtracting our count of distinct configurations from a threshhold and exit once these distinct configurations reach 0. If at any point we receive two similar configurations, we can conclude that we will loop and output $\perp$. At the end of the the compuation, if we have halted before count of the distinct configuration reaches 0, or if we have reached accept stage, we can simply check if the count of the number of distinct writes is less than or equal to k. Thus outputting yes/no. □

(b) *Proof.* Even-States(M, x) is not a computational resource. Consider the following reduction, for each state in our machine M we create a machine M' which has twice the number of states as M. We add $|Q|$ dummy states in M' at all the even positions. At all the odd positions we have the meaningfull states. We make sure that every input goes through the first dummy state only once and the rest of the states are never reached. Now if we wish to find if a machine M halts on an input x, we just have to query for Police$_{Even-States}$ on $(M', x, 1)$, the answer no implies that the machine has outputted $\perp$ and hence is not halting. The answer yes, implies that the machine has indeed halted, and passed through the one state we designed for it to go through. Thus answering halting problem for us. As we know that HP is undecidable, this implies Police$_{Even-States}$ is undecidable. □

(c) *Proof.* HeadTurns(M, x) is a computational resource.

The first condition of Blum's Axiom is true from definition, i.e. the function is defined if M halts on x i.e. there is a unique number that denotes the total no of turns it has made during its run till the machine halted(We can assume that we are coming from the left direction). The only if part is true as we defined the function to output $\perp$ if the machine does not halt.

The second condition is true as $Police_{HeadTurns}$ is decidable. This is because as seen in class in the example of bounding the work tape, here too we can bound the work tape. If we wish to check that HeadTurns$(M, x) \leq k$, our work tape must be bound by $(k * |x|)$. We can imagine that we do not get any useful information if we keep on copying or working on the empty symbol at the end of the tape. We thus now work so that at the start of every new turn we make the following adjustment, we copy all the details of our previous $|x|$ length to the new $|x|$ part of the tape but in reverse, now if a new turn is reached, instead of turning on the configuration, we can go forward to the new part of the tape i.e. copied part and start working in the same direction. We continue the process till all k turns have been finished. Since now the work tape is bounded we have a similar proof as seen in class, we can now bound the number of different configurations by $|Q| \times n \times \Gamma^{nk} \times (nk)$, where $n = |x|$. We can now keep subtracting our count of distinct configurations from a threshhold and exit once these distinct configurations reach 0. If at any point we receive two similar configurations, we can conclude that we will loop and output $\perp$. At the end of the the compuation, if we have halted before count of the distinct configuration reaches 0, or if we have reached accept stage, we can simply check if the number of attempted turns is less than or equal to k. Thus outputting yes/no.

Alternatively, this can also be seen as a consequence of the fact that we can just run the input on the machine, if we keep on looping, then the turns will at some point increase than k, and at that stage we simply output no, if instead the machine halts before reaching k turns we just ouput yes, note that here also we assume that it isn't useful to keep on going to the right when there is nothing of use to read in the tape. This can be easily prevented by making some constraints on the transition function of our TM, such that on a read of empty symbol we always go left. □

(d) *Proof.* Energy(M, x) is not a computational resource. Consider the following reduction, for each state in our machine M we create a machine M' which removes the symbol 1 from writing on the input, i.e. we create a dummy symbol $1'$ in our tape alphabet. We change our transition functions appropriately to now allow $1'$ to do the job that 1 was doing. M' on input x now preprocesses first and replaces all instances of 1 in the input with $1'$. In order to find out if a machine M halts on an input x, we just have to query for Police_{Energy} on $(M, x, 0)$, the answer no implies that the machine has outputted $\perp$ and hence is not halting. The answer yes, implies that the machine has indeed halted, and has not written any 1 on the input tape. Thus answering halting problem for us. As we know that HP is undecidable, this implies Police_{Energy} is undecidable. □

2. (7 points) **(Making TMs Oblivious)** Define a Turing Machine M to be *oblivious* if its head movements do not depend on the input but only on the input length. That is, M is oblivious if for every input $x \in \{0, 1\}^*$ and $i \in \mathbb{N}$, the location of each of M's heads at the i-th step of execution on input s is only a function of $|x|$ and i. Show that for every time constructible function $t : \mathbb{N} \to \mathbb{N}$, if $L \in \mathsf{DTIME}(t(n))$, then there is an oblivious TM that decides L in time $O(t(n)^2)$ (Hint : Use the idea of the simulation in tape reduction theorem).

Solution:

Proof. If $L \in \mathsf{DTIME}(t(n)) \implies \exists$ TTM M, s.t. $\forall x$ $time(M, x) \leq t(|x|)$ and $L(M) = L$.

The idea is to construct a machine M' that simulates M. We observe that whenever M is at step j of its execution, it only reads inputs less than equal to j. This is because if it was reading an input at a tape position more than j it must have taken more than j steps to reach the reach that tape position. To construct an oblivious turing machine that uses this property, when we are simulating step j of execution of M we will visit all positions till j in the tape of M' and then head back to the starting position. Note that since $t(n)$ is time constructible we will always be able to calculate the position j in $O(t(n))$ time (Since j is less than $t(n)$). The running time of such an algorithm is $(2 \times 1) + (2 \times 2) + (2 \times 3) + \ldots + (2 \times t(n))$ during the execution (twice because we are going back to the head position). The total execution time is $(t(n) + 1)(t(n))$ i.e. $O(t(n)^2)$.

To see that such an M' is oblivious, observe that the head position of i^{th} execution can be clearly defined in terms of n and i. Based on i find which execution step of M are we currently in. (find i' st $\sum_{j=1}^{i'}(2j) \leq i < \sum_{j=1}^{i'+1}(2j)$ and then let $h = i - \sum_{j=1}^{i'}(2j)$. If $h \leq i'$ then we are at head position h otherwise we are at head position $2(i') - h$). The transition states are appropriately defined for M' so that M can be simulated accurately. $\square$

3. (7 points) **(Undecidability of Time Bounds)** When we attempted to prove the time hierarchy thereom, we said we do not have an enumeration of Turing machines running in time $t_1(n)$. We substantiate this by this exercise. Let $t(n)$ be a function such that for all $n \in \mathbb{N}$, it holds $t(n) \geq n + 1$, then the problem is of given a Turing machine M testing if it runs in time $t(n)$ is undecidable. (Hint : Reduce from MP_ϵ).

Solution:

Proof. We claim that $\mathrm{MP}_\epsilon \leq_m Q3$.

Let $\sigma : \Sigma^* \to \Sigma^*$ be the reduction.
Input: (M)
Output: (M')
We will construct a machine M' that is as follows:

```
M' on input z where |z| = n
________________
Run M on input ϵ for n + 1 steps
If M accepts, then accept z
else run for t(n)^2 time and reject.
```

Clearly, if M accepts input ϵ, it must except in a fixed number of steps, i.e. $\exists n$ such that M accepts ϵ in $n + 1$ steps. Thus all strings z of length n will be accepted by

M' and hence M' runs in time $t(n)$, because $\forall n, t(n) \geq n+1$. If M does not accept ϵ we have that M' will run for atleast $t(n)^2$ time on all inputs and hence will not run in time $t(n)$ for any input. Thus if we have a machine that can decide $Q3$, then we have a machine that can decide MP_ϵ which we know is undecidable. Thus $Q3$ must be undecidable. □

Extra Credit(7 points): Show that the condition $t(n) \geq n+1$ is necessary in the above claim. That is, if there is at least one n_0 such that $t(n_0) \leq n_0$, then show that the problem of given a Turing machine M testing if it runs in time $t(n)$ is actually decidable.

Solution:

Proof. The above reduction doesn't work as if there is a machine M that accepts ϵ in exactly n_0+1 steps, then it will be accepted by our machine M' in n_0+1 steps, this is not within $t(n)$. □

4. (7 points) **(Tape Reduction for Non-deterministic Turing Machines)** Explore a definition of time in non-deterministic Turing machines : A non-deterministic Turing machine is said to run in time $t(n)$ if it runs in time $t(n)$ in all its non-deterministic computation paths. Let $t(n) \geq n$. Show that any language in $\mathsf{NTIME}(t(n))$ can be accepted by a non-deterministic 2-tape Turing Machine in time $O(t(n))$.

Solution:

Proof. In short we wish to show tape reduction for non deterministic turing machines, i.e. any NTM N running in time $t(n)$ with k tapes can be simulated by a NTM N' running in time $O(t(n))$ with two tapes. To do this, we follow a similar approach in class, i.e. we change our encoding of tape alphabets and include hat symbols to denote our head postion. i.e $\Gamma' = (\Gamma \cup \hat{\Gamma})^k$. Now we must note the head position. We realize that the head position is the only reason that we had to be content with a quadratic time in the algorithm we studied in class. Instead, here we can extract the non determistic capability of the machine. Let q on input tape symbols(assuming 4 tape) (a_1, a_2, a_3, a_4) we can have multiple head positions, though the head position will always be a subset of 2^4 i.e. 16. □

5. (5 points) **(Extension of Crossing Sequences Argument)** We showed in class (Oct 8, 2018) that any 1-tape Turing machine for PALINDROME language should run in $\Omega(n^2)$ time. Extend this argument to show that for $t(n) \geq n$ and $s(n) \geq \log n$ and for any Turing machine that is both $t(n)$ time-bounded and $s(n)$ space-bounded and deciding

PALINDROME, it must hold that the product $t(n)s(n)$ is $\Omega(n^2)$.
(Note that this shows a time-space trade-off for palindrome language, and this trade-off is valid even for the 2-tape linear-time Turing machine that we designed in class.)

Solution:

Proof. Going along the same proof in class, we claim that

Let M be a turing machine accepting PAL and having space $s(n)$, $\exists x, x \in \{0,1\}^{n/4}$ such that $x\#^{n/2}x^R$ runs on time $\Omega(\frac{n^2}{s(n)})$.

Let $C_i(x)$ be a sequence of (states,worktape) on which M crosses the index i. Let

$$C(x) = \{C_i(x) | n/4 \leq i \leq 3n/4\}$$

Claim seen in class, $\forall x \neq y, C(x) \cap C(y) = \phi$. This is still true in our case as we store all the relevant information in order to simulate the turing machines given the states. The cut and paste argument still holds as the work tape is also now same when we cut and paste one part over the other. Since the work tape and the state, input is same, we proceed in an exact same manner and we would end up excepting a string that is not pallindrome if the claim was not true. We don't prove this claim rigorously in this exercise. (Its exactly same to that studied in class).

Analysis:

Let, $r(x) =$ shortest crossing sequence in $C(x)$. Let r_i be $r(x_i)$ where $x_i \in \{0,1\}^{n/4}$. Then all of $r_1, r_2, \ldots, r_{2^{n/4}}$ must be distinct due to the uniqueness of crossing sequences at any position.

Let l be the $\max_x |r_x|$. Number of sequences of length j are given by $(|Q||\Gamma|^{s(n)})^j$, where Γ is the tape alphabet and $s(n)$ is the size of our work tape. (The different possibilites of the work tape are $|\Gamma|^{s(n)}$, this gets included with every state.) Let $|Q||\Gamma|^{s(n)} = q$. We count the number of distince crossing sequences that can be produced with length $\leq l$ i.e. $= q^l + q^{l-1} + \ldots 1$. This must be greater than equal to the distinct crossing sequences that we have i.e.

$$\begin{aligned}
q^l + q^{l-1} + \ldots + 1 &\geq 2^{n/4} \\
\frac{q^{l+1} - 1}{q - 1} &\geq 2^{n/4} \\
q^{l+1} &\geq 2^{n/4} \\
l &= \Omega(\frac{\log 2^{n/4}}{\log q}) \\
l &= \Omega(\frac{n/4}{\log |Q||\Gamma|^{s(n)}}) \\
l &= \Omega(\frac{n}{s(n)})
\end{aligned}$$

Hence there exists some string such that the running time of M is $(n/2)\Omega(\frac{n}{s(n)}) = \Omega(\frac{n^2}{s(n)})$. $\square$