CS388T: ANSWER TO EXTRA CREDIT PROBLEMS

VISHVAS VASUKI

1. Question

(10 points) Prove that any language decided by a k-string nondeterministic Turing machine within time f(n) can be decided by a 2-string nondeterministic turing machine within time O(f(n)).

HINT: Note that in general we cannot assume that the simulating machine 'knows' the function f. This is part of the challenge in this problem.

1.1. **Partial solution and extra credit question.**

Note that, for a k-tape turing machine, originally only the first tape will have the input, and the remaining tapes will be empty.

A guess sequence consists of the nondeterministic choice made, the k symbols seen by the k heads and the current state. A guess sequence of length t will consist of a t(k+2) tuple of integers.

A guess sequence is produced on tape 2. This guess sequence is verified for each of the k tapes in tape 1. If the verification fails, that path of computation is terminated. If the verification succeds, and if the halting state was reached, the appropriate result is produced. If the verification succeeds and the halting state was not reached, the guess sequence is extended. As f(n) is not known beforehand, guess sequences of length t= 1,2,4,... are produced.

During such simulation, when we switch to the value 2t from t, the original input will have been overwritten. How do we keep verifying the guess sequence?

1.2. **Solution.**

1.2.1. *Preliminaries.*

Note that verification includes the following:

- Checking to see if the symbols on the tape being simulated match the symbols 'guessed' to be under the head.
- Checking to see if the state transition at the nth step allows for the state transition guessed for the (n+1)th step to happen.

Suppose that a "proper" guess sequence of length t has been produced. We verify this guess sequence for the first tape, then we verify this guess sequence for the second tape, and so on. While verifying the guess sequence for the second tape, it is possible that the contents of the original tape were overwritten.

It is important to note that the simulation of a guess sequence of length t can require manipulations of at most t cells of the input.

1.2.2. *Workaround.*

One workaround is to do extend the guess sequence in the following way: A guess sequence of length t will include a guess about the first t characters of the original input. Let us call this the **guess-input-string**. If t exceeds n, it will merely mean that the guess-input-string will include some empty spaces in the end.

The **verification procedure** is extended to include the following check:

- Suppose that a guess sequence of length t has been produced.
- Before the verification process begins, the prefix of the input string (of length t) is marked off with special characters (The alphabet can be increased to handle this.). This is called the **workspace**.
- Simultaneously, the **guess-input-string** is verified for correctness. If this verification fails, the computation path is terminated.
- The verification of the guess sequence over the k tapes proceeds as usual. But, the verification procedure is always made to start at the beginning of the workspace.

If the guess sequence is successfully verified on all tapes and if the halting state was not reached, before the guess sequence is extended, the following **restoration procedure** is applied: The contents of the guess-input-string are copied to the workspace.

Note that all the modified **verification procedure** and the **restoration procedure** take O(t) time, where t is the length of the guess sequence. Hence, we have shown how a two tape nondeterministic turing machine can simulate a k tape turing machine in O(f(n)) time without getting thwarted by the "overwriting" problem.

2. Question

Prove that a deterministic turing machine deciding PALINDROMES requires atleast c log n space for some constant c.

2.1. Solution.

PALINDROMES with the input string w, needs to compare the ith character in w with the $(|w| - i)$th character in w. There must be atleast n/2 such comparisons. This is by the definition of the problem, and no turing machine, even with multiple tapes, can avoid this.

Suppose that the machine has to remember the position (i) of the character it is comparing. This information cannot be stored as a "state" because i grows with n, the length of the input. As was proved by Shannon when he developed the concept of information entropy, atleast log i bits are required to store this information. But i can range from 0 to n/2. Hence, such a deterministic turing machine requires $\Omega(logn)$ space to decide PALINDROMES.

Suppose that the turing machine does not have to remember the position of the character it is checking. In this case, as it does not retain any memory of what the value of 'i' is, it cannot recognize the character at position $(|w| - i)$ during a future transition. So, in this case, the comparison of the characters at i and $(|w|-i)$ cannot happen in multiple time steps. Due to the structure of the transition fucntion, this comparison can be accomplished within one transition if and only if two different heads of the turing machine are reading those characters simultaneously. So, some characters of the input will have to be replicated on the work-tapes. As the turing machine is agnostic of the position of the characters it is comparing, it cannot pick and choose which characters are replicated on the work tapes. So, as atleast n/2 comparisons are necessary, atleast n/2 characters will need to be replicated on the work tapes. Hence, such a turing machine requires $\Omega(n)$ space to decide PALINDROMES.

Hence, a deterministic turing machine requires $\Omega(logn)$ space to decide PALINDROMES.