University of California, Los Angeles
CS 281 Computability and Complexity

Instructor: Alexander Sherstov
Scribe: Zeina Migeed
Date: January 10, 2019

LECTURE
2

Complexity results for variations of Deterministic TMs

Last lecture, we formalized the notion of a computation. In particular, we defined a TM, which is a model of universal computation. We will state the formal definition again in section 2. We also looked at some variations of TMs such as a TM with restricted alphabet and bidirectional tape TMs. While these TMs may seem computationally more powerful on first glance, they have the same computational power as a single-tape TM. In this course though, we are particularly interested in the efficiency of computation.

In this lecture, we will consider the complexity of several variations of TMs and formalize our findings in section 3. Secondly, we will consider the language of palindromes. We will prove our first complexity result of this course, which is that a multi-tape TM decides if a string is a palindrome in linear time in the size of the input string while a single-tape TM performs the same computation in quadratic time in the size of the input string. Throughout this lecture, we will assume that all TMs mentioned are deterministic.

2.1 Review on TMs

Let us first begin by restating the formal definition of a multi-tape TM from the past lecture. Note that a multi-tape TM is computationally equivalent to a single-tape TM. We will prove this in the next section.

Definition 2.1. A k-tape TM is a 3-tuple: (Γ, Q, δ) where Γ is the set of finite alphabet with $\{0, 1, , \sqcup\} \subseteq \Gamma$. Q is the finite set of states with $\{\mathit{Start}, \mathit{Halted}\} \in Q$. δ is the transition function which describes the dynamics of our TM. More specifically we have: $\delta : Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{\leftarrow, ., \to\}$. This means that one step of computation in our TM is determined by δ. The function δ given the current state $q \in Q$ and some symbol to read $\gamma \in \Gamma^k$ and determines the new state $q' \in Q$ as well as the symbol to write on the current cell $\gamma' \in \Gamma^{k-1}$ and the direction of the TM's head which is either to go left, right or to stay put.

2.2 Bidirectional tapes

Consider a TM with k tapes. Intuitively, a bidirectional tape TM is the same as a TM with standard tapes. The difference is that the tapes extend infinitely in both directions.

We want to show that a TM with bidirectional tapes and a TM with standard tapes have the same complexity up to a constant factor.

Before we state the theorem, we can see that there are two components in this proof. First, we have to show that both variations have the same computational power and then we show that they have the same complexity up to a constant factor. Generally, to show that two two TMs have the same computational power, we have to simulate each model using the other. Now we will state this formally.

Theorem 2.2. *$f : \{0,1\}^* \to \{0,1\}^*$ is computable in time $T(n)$ with bidirectional tapes iff f is computable in time $\mathcal{O}(T(n))$ using standard tapes.*

Proof. The forwards direction is clear because we can easily simulate a standard tape using a bidirectional tape, so we will consider the backwards direction.

Let $M = (\Gamma, Q, \delta)$ be a TM with bidirectional tapes. We will simulate M with a standard tape TM $M' = (\Gamma', Q', \delta')$ as follows. We use a new set of alphabet which will consist of all possible pairs over Γ. In particular, we have a tape extending in both directions:

```
...|_0_|_1_|_>_|_a_|_b_|...
```

Then pictorially, we will do the following:

```
step 1: combine the tapes

|_>_|_a_|_b_|...
|_1_|_0_|

step 2: remove the boundaries between the top and bottom cells.

| > | a | b |...
|_1_|_0_|___|
```

Now each cell holds a version of the alphabet consisting of all possible pairs of all elements of Γ. So we have $\Gamma' = \Gamma \times \Gamma$. How do we determine if the tape head is pointing to the top or the bottom symbol? We hold this information as a state. When a TM needs to "memorize" a finite amount of information, it stores it as a state. So we have $Q' = Q \times \{\top, \bot\}$. The transition function δ' will behave exactly the same as δ. The only change is how to determine whether the tape head should move left or right. If we are reading the top symbol, then we would move in the same direction as δ would. Otherwise, we move in the opposite direction.

For the complexity, each step would take approximately $\mathcal{O}(1)$. So we showed how to simulate one step of computation for a bidirectional tape using a standard tape in constant time. So we are done. □

2.3 Single Tape TMs

A single tape TM will have one tape for input, work and output. Formally, we may define it as the following 3-tuple: (Γ, Q, δ) with Q and Γ defined in the same way as before, and $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, ., \rightarrow\}$.
As we mentioned in the introduction, a multi-tape TM and a single tape TM are equivalent in computational power. However, as we might expect, a single tape TM is less efficient.

As before, we will first show that both machines are equivalent in computational power, then prove our complexity results.

Theorem 2.3. *$f : \{0,1\}^* \to \{0,1\}^*$ is computable in time $T(n)$ on a k-tape TM iff f is computable in time $\mathcal{O}(T(n^2))$ on a single-tape TM.*

Before we start the proof, note that as a consequence of this theorem, the complexity is still polynomial with respect to the input which means that the class P is unaffected.

Proof. The forwards direction is straight forward, so we consider the backwards direction. We will show how to simulate a k-tape TM $M = (\Gamma, Q, \delta)$ with a single-tape TM $M' = (\Gamma', Q', \delta')$. The first step is to combine our k-tapes into a single tape using the same idea in the previous proof, which is to consider a different version of the alphabet which consists of all possible k-tuples over the alphabet. Pictorially we have:

```
|_>_|_0_|_1_|...
|_>_|_a_|_b_|...
  .   .   .
  .   .   .

step 1: remove the boundaries between the top and bottom cells.

| > | 0 | 1 |...
|_>_|_a_|_b_|...
  .   .   .
  .   .   .
```

At this step, we have that $\Gamma' = \Gamma^k$. However, in a k-tape TM, we have k read/write heads and each can move in any direction. We only have one tape head in M'. The solution is to use the same idea again and consider a "marked" version of each symbol, to denote that the head of the tape is pointing to that symbol. Pictorially:

```
Step 2: Consider a marked version of the alphabet
to indicate the position of the read/write head.

|  >  |  0'  |  1  |...
```

```
|__>__|__a__|__b'_|...
   .     .    .
   .     .    .
```

We can achieve this by having $\Gamma' = (\Gamma \times \{0,1\})^k$ where the 0 and 1 indicate whether the head is pointing to the given symbol or not.

Now we will simulate one step of computation of M using M'. First, we scan the entire tape left to right memorizing the k marked symbols in state. Then scan the tape again updating the circled locations and moving the marks according to the original transition function δ.

Now that we have shown that both models are computationally equivalent, we move to the complexity results.

We simulate the step of computation mentioned above on a single tape TM M' in $T(n)$ since we are scanning the entire input which has size n. What is the bound on the number of times we have to simulate this step? The bound is the running time of our original TM t, which is $T(n)$. □

2.4 Deciding Palindromes

We will now prove the main complexity result of this lecture which is that to decide if a string is in the language of palindromes on a multi-tape TM is computed in $\mathcal{O}(n)$ while on a single-tape in $\mathcal{O}(n^2)$.

Last time we showed how to decide PAL on a multi-tape TM. We have the input on one tape and we make a copy of it on another tape, then with the first tape's head at the beginning of the input, and the second tape's head at the end of the input, we move the heads in opposite directions and check if the current symbols are equal. If all symbols are equal then the string is a palindrome and otherwise it is not. Clearly, this takes n steps of computation.

Now we will show how to decide PAL using a single tape TM. We will first need some definitions.

Definition 2.4. Consider a single tape TM M and input $x \in \{0,1\}^*$. The crossing sequence $\mathcal{C}_i(x)$ is the sequence of states $q_{1_i}, q_{2_i}, ...q_{n_i}$ of M where q_{i_k} is the state of M after crossing the i^{th} boundary for the k^{th} time where $k \in \{1, ..., n\}$.

We will say i^{th} boundary to denote the boundary between the cell i and the cell $i+1$. Note that the direction from which we cross a given boundary alternates from left to right.

```
                    |
|_>_|___|___|___|...|___|...
                    |
    1   2   3   4   i   i+1
```

Consider any pair of input x, y such that $M(x) = M(y)$ and consider the following partitions of x as x' and x'' and y as y' and y''.

```
Parition for x:

    |     x'    |         x''         |
|_>_|_\_|_\_|_\_|___|___|___|___|___|___|___|___|___
    |           |                     |
                i

Partition for y:

    |         y'        |     y''     |
|_>_|___|___|___|___|___|_\_|_\_|_\_|___|___|___|___
    |                   |           |
                        j
```

Claim 2.5. *Let $M(x) = M(y)$ and $\mathcal{C}_i(x) = \mathcal{C}_j(y)$ with $x = x'x''$, $y = y'y''$, $i = |x'|$ and $j = |y'|$. Then $M(x) = M(y) = M(x'y'')$.*

Proof. So suppose $|\mathcal{C}_i(x)| = |\mathcal{C}_j(y)| = 1$. Then we have only one state in the sequence. By that point, the TM in both cases will process the rest of the input in both cases in the same way because because it cannot distinguish between them, since all it can remember is the information in the state. Then inductively the same holds for $|\mathcal{C}_i(x)| = |\mathcal{C}_j(y)| \geq 1$. So we have $\mathcal{C}_i(x) = \mathcal{C}_i(x'x'') = \mathcal{C}_i(x'y'') = \mathcal{C}_i(y'y'') = \mathcal{C}_i(y)$.

$\square$

We consider a single-tape TM that decides PAL. How do we know that this algorithm is optimal?

Theorem 2.6. *Any single-tape TM M that decides PAL runs in $\Omega(n^2)$ time.*

Proof. Consider inputs of the form $x0^nx^R$ where $x \in \{0,1\}^n$ and $(x_1...x_n)^R = x_n...x_1$ for $x_i \in \{0,1\}$.

```
Pictorial representation:

              i
              |
    x     |0..|..0|   x^R
              |
<-------> <-------> <------->
    n         n        n
**************
partition a
             j
             |
    y     |0.|...0|   y^R
             |
<-------> <-------> <------->
    n         n        n
             ****************
                partition b

We will concatenate a and b and derive a
contradiction.
```

If $\mathcal{C}_i(x0^nx^R) = \mathcal{C}_j(y0^ny^R)$ for some $i, j \in \{n, n{+}1, ..., 2n\}$ and $y \in \{0,1\}^n$ then $M(x0^i0^{n-j}y^R) = 1$ by claim 2.5. However, by inspection, we can see that $M(x0^i0^{n-j}y^R) = 0$.

We conclude that $\mathcal{C}_i(x0^nx^R) \neq \mathcal{C}_j(y0^ny^R)$ for any $i, j \in \{n, n+1, ..., 2n\}$. This is intuitive because the TM should be able to distinguish between the two inputs at these positions.

The next step in the proof is to show that some input of the form $x0^nx^R$ will only have long crossing sequences. We will define precisely what we mean by "long", but longer crossing sequences would require more steps of computation.

Let $\mathcal{L}$ be $\max\limits_{x\in\{0,1\}^n} \min\limits_{\{n\leq i\leq 2n\}} |c_i(x0^nx^R)|$. Let Q be the set of states of M.

Consider the number of distinct sequences of length at most $\mathcal{L}$. We have 1 sequence of length 0. Q sequences of length 1. Q^2 sequences of length 3 etc. So the number of distinct sequences of length at most $\mathcal{L}$ is $(|Q|+1)^{\mathcal{L}}$.

The number of inputs of the form $x0^nx^R = 2^n$.

We know that the number of distinct sequences of length at most $\mathcal{L}$ is at least the number of inputs of the form $x0^nx^R$

So $(|Q|+1)^{\mathcal{L}} \geq 2^n$. So we have $\mathcal{L} \geq \frac{n}{log(|Q|+1)}$.

So some input will only have long sequences at lines $n, n+1, ..., 2n$ where "long" is $\frac{n}{log|Q|+1}$.

Since every crossing takes one step and we are crossing each index approximately n times, we have that M runs in time $\frac{n^2}{log(|Q|+1)}$.

$\square$

2.5 Beyond two tapes

We proved that a 2-tape TM is quadratically more optimal than a single-tape TM. The question arises of whether this claim generalizes to k-tape TMs. The answer is No.
Next lecture, we will prove the following Theorem.

THEOREM 2.7. *If $f : \{0,1\}^* \to \{0,1\}$ is computable in time $T(n)$ on a k-tape TM, then f is computable in time $\mathcal{O}(T(n)logT(n))$ on a 2-tape TM.*