CS:4330 Theory of Computation

Spring 2018

# Computability Theory

## TM Variants

Haniel Barbosa

The University of Iowa

# Readings for this lecture

Chapter 3 of [Sipser 1996], 3rd edition. Section 3.2.

# Variants of Turing Machines

- ▷ There are many alternative definitions of Turing machines.
- ▷ Nondeterministic machines or machines with multiple tapes.
- ▷ These are called *variants* of the Turing machine model.
- ▷ The original model and its variants have the same expressive power: they recognize the same class of languages.
- ▷ We will explore some variants and prove their equivalence in expressive power.
- ▷ Turing machines are *robust*, allowing several variances without affecting its expressive power.

# Variants of Turing Machines

The transition function of a standard TM forces the head to move to left or right after each step. Let us vary the type of transition function permitted:

- ▷ Suppose that we allow the head to *stay put*, i.e.

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

- ▷ $S$ transitions can be represented by two standard transitions: one that moves to the left followed by one that moves to the right

- ▷ Since we can convert a TM which stays put into one that does not, the extension does not increase the expressive power of standard TMs.

## Variants of Turing Machines

The transition function of a standard TM forces the head to move to left or right after each step. Let us vary the type of transition function permitted:

- Suppose that we allow the head to *stay put*, i.e.

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

- $S$ transitions can be represented by two standard transitions: one that moves to the left followed by one that moves to the right

- Since we can convert a TM which stays put into one that does not, the extension does not increase the expressive power of standard TMs.

**General idea for proving equivalences between variants**

To show that one type of machine simulates the other.

# Multitape Turing Machines

A multitape TM is like a standard TM but with several tapes

- Each tape has its own head for reading/writing
- Initially the input is on tape 1 and other tapes are blank
- Transition function allows reading, writing, and moving the heads on all tapes simultaneously, i.e.

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L,R\}^k, \quad \text{where } k \text{ is the number of tapes}$$

- $\delta(q_i, \mathrm{a}_1, \ldots, \mathrm{a}_k) = \langle q_j, \mathrm{b}_1, \ldots, \mathrm{b}_k, L, R, \ldots, L \rangle$ means that if the machine is in state $q_i$ and heads 1 through $k$ are reading symbols $\mathrm{a}_1$ through $\mathrm{a}_k$, the machine goes to state $q_j$, writes symbols $\mathrm{b}_1$ through $\mathrm{b}_k$ and directs each head to move left or right as specified.

# Example of a Multitape TM

Let $L = \{0^a 1^b 2^c \mid c = \lfloor log_a b \rfloor, a > 1, b > 0\}$. Note that $a^c \leq b < a^{c+1}$.

We may use a three-tape TM to recognize this language: the input tape, the second tape containing $x$ and the third tape containing $k$, where $x = a^{k+1}$, based on the following algorithm:

```
x = a; k = 0;
while (x <= b)
   x = x*a; k = k+1;
return k;
```

# Example of a Multitape TM

Let $L = \{0^a 1^b 2^c \mid c = \lfloor log_a b \rfloor, a > 1, b > 0\}$. Note that $a^c \leq b < a^{c+1}$.

$M =$"On input string $w$:

1. Check if $w \in 00^+1^+2^+$; if not, *reject*
2. Copy all $0$s on the input tape to tape $2$
3. If number of 0s on tape 2 exceeds number of 1s on the input tape, go to stage 6
4. Multiply 0s on tape 2 by number of 0s on the input tape and keep the result on tape 2
5. Add a symbol "2" to tape 3; go to stage 3.
6. If number of 2s n tape 3 is the same number of 2s on the input tape, *accept*; otherwise *reject*"

# Equivalence between multi- and singletape TMs

**Theorem**

*Every multitape Turing machine has an equivalent single-tape Turing machine.*

**Proof idea**

Show how to convert a multitape TM $M$ into a single-tape TM $S$. The important step is how to simulate $M$ with $S$.
Assuming $M$ has $k$ tapes:

- $S$ simulates the effect of $k$ tapes by storing their information on its single tape
- S uses a new symbol # as a delimiter to separate the contents of different tapes
- S keeps track of the location of the heads by marking with a • the symbols where the heads would be.

# General construction

$S =$"On input string $w = w_1, \ldots, w_n$:

1. Put $S$ in the format that represents all $k$ tapes of $M$:

$$S = \# \dot{w}_1\, w_2 \cdots w_n \# \dot{\sqcup} \# \dot{\sqcup} \# \cdots \#$$

2. To simulate a single move, $S$ scans its tape from the first #, which marks the left-hand end, to the $(k+1)$-st #, which marks the right-hand end, in order to determine the symbols under the virtual heads. Then $S$ makes a second pass to update the tapes according to the way $M$'s transition function dictates.
3. If at any point $S$ moves one of the virtual heads to the right onto a #, this action means that $M$ has moved the corresponding head onto the previously unread blank portion of that tape. So $S$ writes a blank symbol on this tape cell and shifts the tape contents, from this cell until the rightmost #, one unit to the right. Then it continues to simulate as before."

# Other variants

Notes on Nondeterministic Turing Machines and Enumerators done in class in the blackboard.