

## Last class

- **TM Variants:** Multitape TM, RAM TM
- All Python/C++ programs with time  $T(n)$  can be converted to a single-tape TM with time  $O(T(n)^4)$
- (Strong) Church-Turing Thesis (Exception: Quantum Computing)

---

**Def 3.13** Let  $T : \mathbb{N} \rightarrow \mathbb{N}$ . **Language**  $L \subseteq \{0, 1\}^*$  is in  $DTIME(T(n)) \Leftrightarrow \exists \mathbf{A}$  (multitape) TM  $M$  that decides  $L$  in time  $O(T(n))$ .

---

**Def 3.14**  $P = \bigcup_{c \geq 1} DTIME(n^c)$

- $L \in DTIME(n^{100}) \Rightarrow L \in P$
- All Languages  $\supseteq$  Decidable Languages  $\supseteq P \supseteq$  Regular Languages

## Computability

### Universal TM

#### Encoding of a multitape TM

- Assume our encoding of the TMs satisfy the following properties:
  1. Every string  $\alpha \in \{0, 1\}^*$  represents some TM. (On encoding  $\alpha$ ,  $M_\alpha$  always rejects)
  2. Every TM is represented by infinitely many strings.
    - k-tape TM  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$
    - $\Sigma = \{0, 1\}$ .  $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k$

Encoding of TM  $M$  has size  $O_M(1)$ .

---

**Thm 4.1 (Universal TM)** There exists a multitape TM  $U$  s.t. for all  $x, \alpha \in \{0, 1\}^*$ ,  $U(x, \alpha) = M_\alpha(x)$ . Moreover, if  $M_\alpha$  halts on input  $x$  within  $T$  steps, then  $U(x, \alpha)$  halts in  $O_{M_\alpha}(T \log T)$  steps. (Weaker version:  $O_{M_\alpha}(T^2)$ )

*Proof of the weaker version:*

By Lem 3.9, every multitape TM can be converted to a single-tape TM  $M_\beta$  with time  $O_M(T^2)$ .

- Input tape:  $x$
- Work tapes:
  - Simulation of single-tape  $M$ 's work tape
  - $\alpha(O_M(1))$
  - Current state of  $M$  ( $O(\log_2 |Q|)$ )

For each step of  $M$ , our UTM  $U$ :

1. Reads the symbol  $a$  on the work tape of  $M$
2. Reads the current state  $q_1$  of  $M$

3. Scans through the description of  $M$  to find  $\delta(q_1, a)$
4. Updates the symbol (on the work tape of  $M$ ), and moves the head if needed
5. Updates "the current state of  $M$ " from  $q_1$  to  $q_2$

In total, the above simulation (for one step) takes

$$O(1) + O_M(1) + O_M(1) + O(1) + O_M(1) = O_M(1)$$

- $f(n) = O(1) \Leftrightarrow (\exists c)(\exists N)(\forall n \geq N)(|f(n)| \leq c)$
- $f(n) = O_M(1) \Leftrightarrow (\forall M)(\exists c = c(M))(\exists N)(\forall n \geq N)(|f(n)| \leq c)$

*Proof of Thm 4.1*

Let  $k$  be the number of work tapes of  $M_\alpha$ , and let  $\Gamma$  be its alphabet. Assume  $U$  uses alphabet  $\Gamma^k$ .  $U$  moves the tape, instead of moving the head.

eg.

...	□	□	u	n	i	v	e	r	s	a	l	□	□	□	...
...	□	□	t	u	r	i	n	g	□	□	□	□	□	□	...
...	□	m	a	c	h	i	n	e	□	□	□	□	□	□	...
					↑										

- □: buffer symbol

However, the simulation takes  $O_M(T^2)$  time.

For convenience, think of  $U$ 's parallel tapes as infinite in both directions. We split each of  $U$ 's parallel tapes into zones, denoted by  $R_0, L_0, R_1, L_1, \dots, R_{\lceil \log T \rceil}, L_{\lceil \log T \rceil}$  (The center cell is not in any of these zones), where

$$|R_i| = |L_i| = 2^{i+1}$$

We maintain the following invariants:

1. **Each zone is empty, full or half-full.**

eg.

□	□	⊠	⊠	u	n	i	⊠	⊠	v	e	⊠	⊠	r	⊠	s	⊠	a	l	...
$L_1$	$L_1$	$L_1$	$L_1$	$L_0$	$L_0$	0	$R_0$	$R_0$	$R_1$	$R_1$	$R_1$	$R_1$	$R_2$	$R_2$	$R_2$	$R_2$	$R_2$	$R_2$	...

2. **The number of ⊠ in  $L_i \cup R_i$  is exactly  $2^{i+1}$ .**

That is, we have:

- $R_i$  is empty and  $L_i$  is full.
- or
- $R_i$  is full and  $L_i$  is empty.
- or
- $R_i$  is half-full and  $L_i$  is half-full.

- Perform a left shift

1. Find the smallest  $i_0$  s.t.  $R_{i_0}$  is not empty. (Equivalently, it is the smallest  $i_0$  s.t.  $L_{i_0}$  is not full)
2. Put the leftmost non- $\boxtimes$  symbol in  $R_{i_0}$  to location 0, and shift the remaining leftmost  $(2^{i_0} - 1)$  non- $\boxtimes$  symbols from  $R_{i_0}$  into  $R_0, R_1, \dots, R_{i_0-1}$ , filling up exactly half. ( $\sum_{k=0}^{i_0-1} 2^{k+1} = 2^{i_0+1} - 2$ )
3.  $U$  performs the symmetric operation to the left. That is, let  $j$  be from  $i_0 - 1$  down to 0, moving half of  $|L_0| + \dots + |L_{i-1}|$  plus one non- $\boxtimes$  symbols to  $L_{i_0}$ .
4. After our shift,  $R_0, L_0, \dots, R_{i_0-1}, L_{i_0-1}$  are half-full.  $R_{i_0}$  is either empty or half-full,  $L_{i_0}$  is either full or half-full.

Once we perform a shift with index  $i_0$ , the next  $2^{i_0} - 1$  shifts will have index less than  $i_0$ .

Thus, at most  $\frac{1}{2^{i_0}}$  fraction of shifts have index  $i_0$ .

- $index(i) \stackrel{def}{=} \text{the index of } i^{th} \text{ shift.}$

Total time complexity:

$$\begin{aligned}
& k \cdot \sum_{i=1}^T O(2^{index(i)}) \\
&= k \cdot \sum_{j=0}^{O(\log_2 T)} O(2^j) \cdot \#\{1 \leq i \leq T | index(i) = j\} \\
&= k \cdot \sum_{j=0}^{O(\log_2 T)} O(2^j) \cdot \frac{T}{2^j} \\
&= k \cdot \sum_{j=0}^{O(\log T)} O(T) \\
&= k \cdot O(T \log T) \\
&= O_M(T \log T)
\end{aligned}$$

Q.E.D

---