Turing Machines

- Turing Machine: Model of Computation
- Turing Machine: Anatomy and Working
- Turing Machine: Formal Definition and Rules of Computation
- Recognizable and Decidable Languages
- Turing Machine: Levels of Abstraction
- Varaints of Turing Machine and The Church-Turing Thesis
- Non-Deterministic Turing Machine

Imdad ullah Khan

Turing Machine Variants
The Church-Turing Thesis

Turing Machine Variants

Turing Machines are Robust

Many different variants of Turing machines can be defined

The basic variant is robust — As long as any other variant reads and write a finite number of symbol in each step, the basic variant can simulate it

TM with stay option

Turing Machine with "stay" option can keep the head at a location instead of moving left or right

Finite State Control

In each step

- **Reads a symbol at the head**
- **Changes state**
- **Writes a symbol at the head**
- **Moves head to left or right or stay**

head

I	N	P	U	C	B	A	U	T	P	U	T	␣	␣	␣	␣	$\cdots$

INFINITE REWRITABLE TAPE

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

Equivalence of computational power of TM variants

How to prove two models have equal computational power?

Show that for M_1 of one model, there is a machine M_2 of the second model such that $L(M_1) = L(M_2)$ and vice versa

We say M_2 simulates M_1

Configurations of M_1 corresponds to configurations of M_2

Note that equivalent computational power does not mean equal efficiency or speed

TM with stay option = Basic TM

A TM with stay option has equal computational power as a basic TM

A TM with stay option M_1 can simulate any basic TM M_2

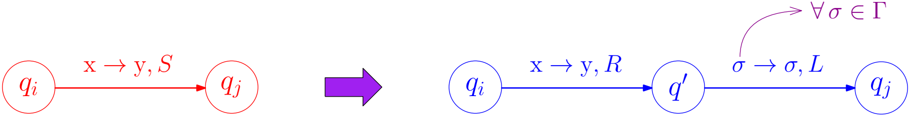
M_1 just does not use the stay option

TM with stay option = Basic TM

A TM with stay option has equal computational power as a basic TM

A basic TM M_2 can simulate any TM with stay option M_1

For every transition in M_1 with stay instruction, M_2 makes an additional transition moving the head right and then move left

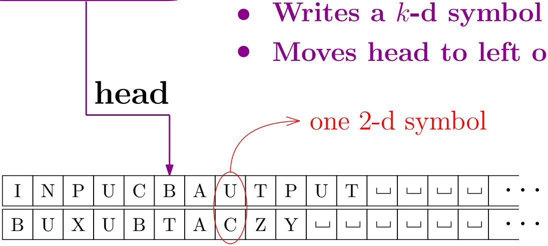


Multitrack TM

A multitrack Turing Machine has a tape with multiple tracks and a single head

In each step

- **Reads k-d symbol at the head**
- **Changes state**
- **Writes a k-d symbol at the head**
- **Moves head to left or right**



INFINITE REWRITABLE 2-TRACK TAPE

$$\delta : Q \times \Gamma^k \mapsto Q \times \Gamma^k \times \{L, R\}$$

Multitrack TM = Basic TM

A basic TM M can simulate any multitrack TM M'

Let $M' = (Q', \Sigma', \Gamma', q'_0, q'_{acc}, q'_{rej}, \delta')$

We design M to simulate M', that works on composite symbols (representing the k-d symbols of M')

Formally, $M = (Q', \Sigma, \Gamma, q'_0, q'_{acc}, q'_{rej}, \delta)$, where

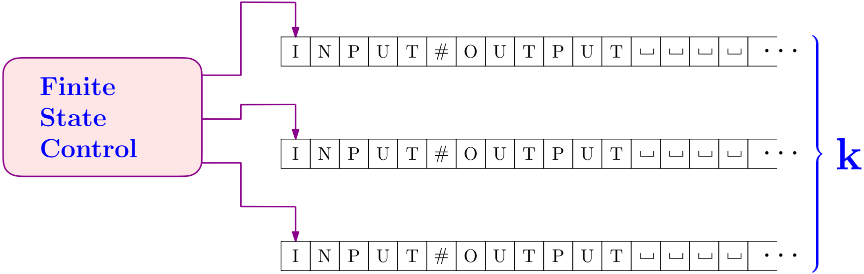
- $\Sigma = \underbrace{\Sigma' \times \Sigma' \times \ldots \times \Sigma'}_{k \text{ times for } k \text{ tracks}}$
- $\Gamma = \underbrace{\Gamma' \times \Gamma' \times \ldots \times \Gamma'}_{k \text{ times for } k \text{ tracks}}$
- $\delta(q_i, (\sigma_1, \ldots, \sigma_2)) = \delta'(q_i, \langle \sigma_1, \ldots, \sigma_k \rangle)$

$\Gamma' = \{a, b, \sqcup\}$

Γ'	Γ
(a, a)	A
(a, b)	B
$(a, \sqcup)$	C
(b, a)	D
(b, b)	E
$(b, \sqcup)$	F
$(\sqcup, a)$	G
$(\sqcup, b)$	H
$(\sqcup, \sqcup)$	I

Multitape TM

Multitape Turing Machine has k read/write tapes each with its head



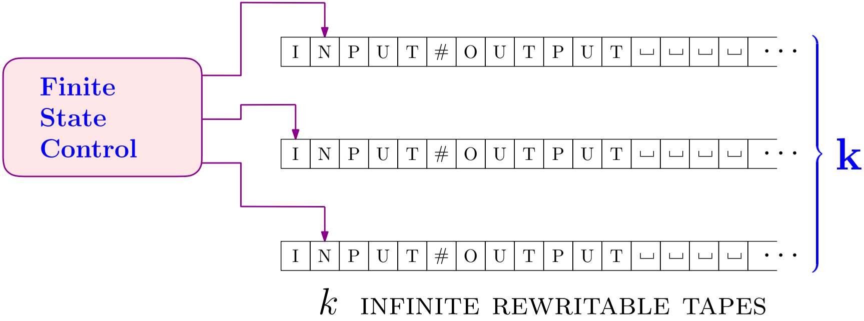
k INFINITE REWRITABLE TAPES

$$\delta : Q \times \Gamma^k \mapsto Q \times \Gamma^k \times \{L, R\}^k$$

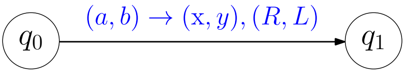
$$q_0 \xrightarrow{(a,b) \to (\mathrm{x}, y), (R, L)} q_1$$

Multitape TM

Multitape Turing Machine has k read/write tapes each with its head

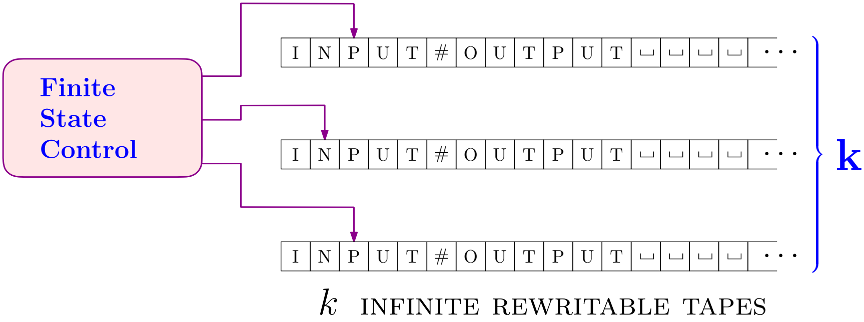


$$\delta : Q \times \Gamma^k \mapsto Q \times \Gamma^k \times \{L, R\}^k$$

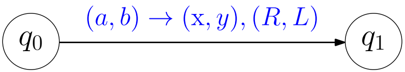


Multitape TM

Multitape Turing Machine has k read/write tapes each with its head

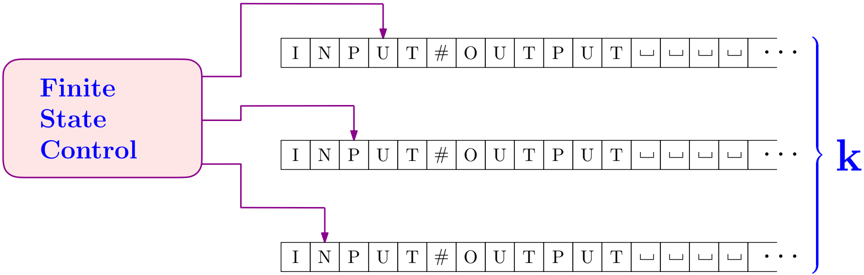


$$\delta : Q \times \Gamma^k \mapsto Q \times \Gamma^k \times \{L, R\}^k$$



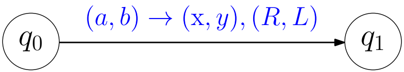
Multitape TM

Multitape Turing Machine has k read/write tapes each with its head



k INFINITE REWRITABLE TAPES

$$\delta : Q \times \Gamma^k \mapsto Q \times \Gamma^k \times \{L, R\}^k$$



Multitape TM = Basic TM

A multitape TM has equal computational power as of a basic TM

A multitape TM M_1 can simulate any basic TM M_2

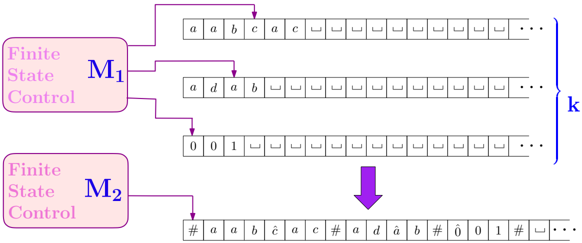
Just use the first tape

Multitape TM = Basic TM

A multitape TM has equal computational power as of a basic TM

A basic TM M_2 can simulate any multitape TM M_1

- M_2 stores content of all k tapes in its single tape with $\#$ as separator
  $\triangleright$ Assuming $\#$ is not used by M_1
- For each symbol σ (of M_1) M_2 also uses it special version $\hat{\sigma}$. For each section of the tape $\hat{\sigma}$ indicates location of the corresponding head



Multitape TM = Basic TM

A multitape TM has equal computational power as of a basic TM

A basic TM M_2 can simulate any multitape TM M_1

- M_2 stores content of all k tapes in its single tape with $\#$ as separator $\triangleright$ Assuming $\#$ is not used by M_1
- For each symbol σ (of M_1) M_2 also uses its special version $\hat{\sigma}$. For each tape section $\hat{\sigma}$ indicates location of the corresponding head

On input $w_1 = w_{11} \dots w_{1\ell}$, $w_2 = w_{21} \dots w_{2m}$, $w_3 = w_{31} \dots w_{3n}$ to M_1

- M_2's tape is $\#\hat{w_{11}} \dots w_{1\ell}\#\hat{w_{21}} \dots w_{2m}\#\hat{w_{31}} \dots w_{3m}\#\sqcup$
- To simulate a transition of M_1, M_2 move its head from first $\#$ to $(k+1)$st $\#$ to find current symbols ($\hat{\sigma}$/virtual heads)
- M_2 then makes the transition as dictated by transition of M_1 (writing new symbols and moving all virtual heads)
- If a "head" needs to be moved beyond the $\#$, M_2 first shift all tape content one step to right and continue

Multitape TM = Basic TM

If Multitape TM = Basic TM, then why study them?

Some time it is easier to construct/describe multitape TM's

$L = \{a^n b^n : n \geq 0\}$ is decidable

We design a 2-tape TM to decide L

1. Suppose $w \in \{a, b\}^*$ is given on tape 1
2. Scan tape 1 left-to-right to check if $w \in a^* b^*$
3. Copy all b's in w from tape 1 to tape 2
4. Scan both tapes left-to-right to see if every a on tape 1 has a corresponding b on tape 2 and vice-versa, if not **reject**
5. **Accept** if head on both tapes read $\sqcup$

Runtime on $a^n b^n$ of basic TM is $O(n^2)$, while that on 2 tape TM is $O(n)$

Multitape TM = Basic TM

If Multitape TM = Basic TM, then why study them?

Some time it is easier to prove closure properties

Recognizable languages are closed under union

Suppose L_1 and L_2 are recognizable languages, recognized by M_1 and M_2

We design a 2-tape TM to recognize $L_1 \cup L_2$

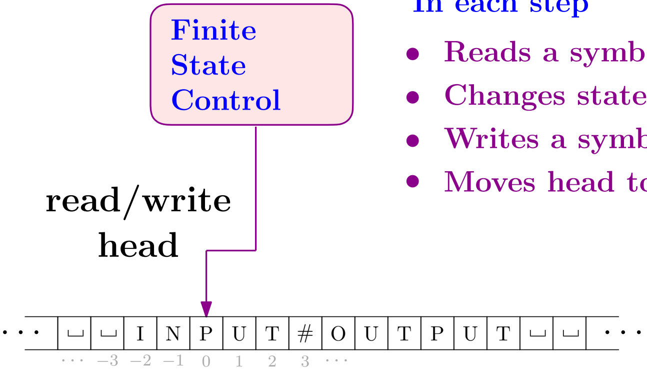
Algorithm check if $w \in L_1 \cup L_2$

1: **while true do**
2: Run M_1 on tape 1 for one step ▷ make one transition of M_1
3: Run M_2 on tape 2 for one step ▷ make one transition of M_2
4: **Accept** if either M_1 or M_2 accepts

Why not run M_1 on tape 1, then run M_2 on tape 2, accept if either does?

TM with 2-way infinite tape

A Turing Machine with 2-way infinite tape can move its head left and right unrestricted



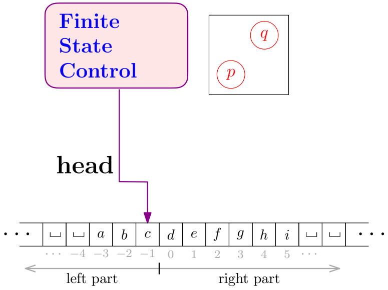
2-WAY INFINITE REWRITABLE TAPE

In each step

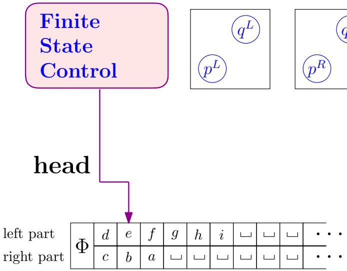
- **Reads a symbol at the head**
- **Changes state**
- **Writes a symbol at the head**
- **Moves head to left or right**

TM with 2-way infinite tape = Basic TM

A 2 track TM M can simulate any TM M' with a 2-way infinite tape



2-WAY INFINITE TAPE



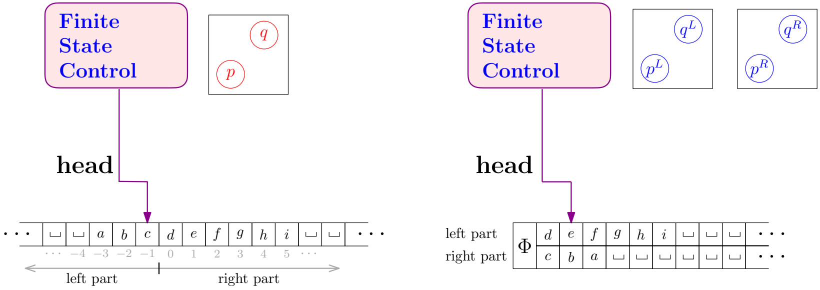
INFINITE 2-TRACK TAPE

To simulate a move of M', M operates as follows

- If working on upper track, use states in Q^R, move head in same direction as M'
- If working on lower track, use states in Q^L, move head in opp. direction as M'
- If move results in hitting Φ, switch to the other track

TM with 2-way infinite tape = Basic TM

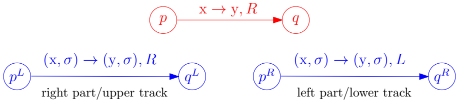
A 2 track TM M can simulate any TM M' with a 2-way infinite tape



2-WAY INFINITE TAPE

INFINITE 2-TRACK TAPE

To simulate a move of M', M operates as follows



Turing Machine Variants

Turing Machines have equal computational power as

- TMs with stay option
- TMs with 2-way infinite tapes
- TMs with multiple tapes
- TMs with multitrack tapes
- TMs with multidimensional tapes
- Offline TMs
- Nondeterministic TMs
- TMs with RAM
- Enumerators
- λ-Calculus (primitive recursive functions)
- Cellular Automata

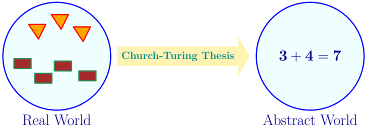
Turing Machine Variants: Church-Turing thesis

Church-Turing thesis: Computable = Computable by TM

Church-Turing Thesis

Any computational problem that can be solved by a physical device, can be solved by a Turing Machine

Any computational that can be performed by mechanical means can be carried out by a Turing Machine



Not a theorem ▷ but no known computational model has more power than TM

Algorithms = Turing Machines

An algorithm to compute $f(w)$ is a TM which computes $f(w)$