CS 302.1 - Automata Theory

Lecture 10

Shantanav Chakraborty

Center for Quantum Science and Technology (CQST)

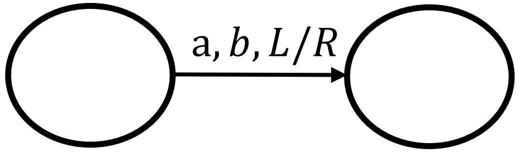
Center for Security, Theory and Algorithms (CSTAR)

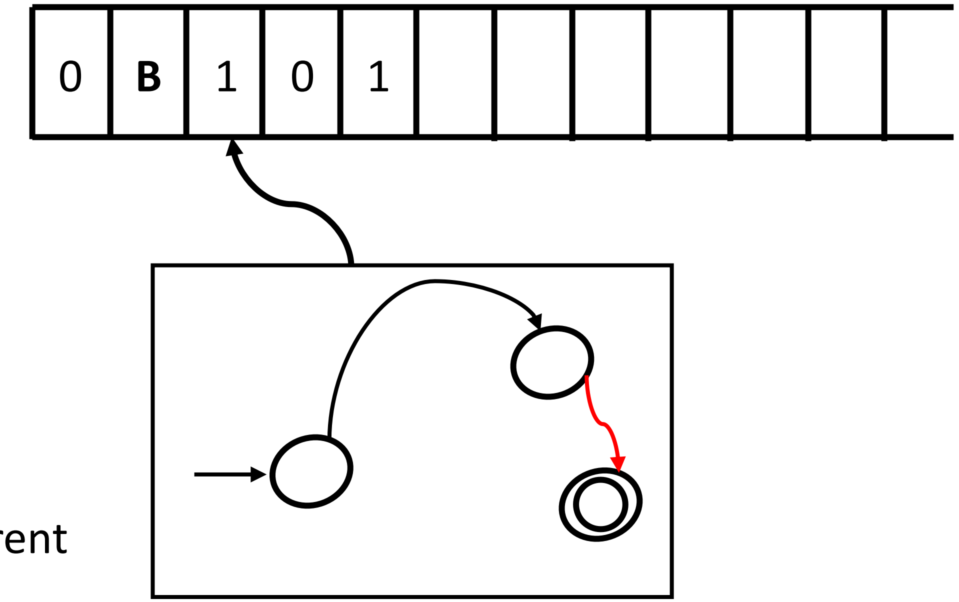
IIIT Hyderabad

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY

HYDERABAD

Quick Recap



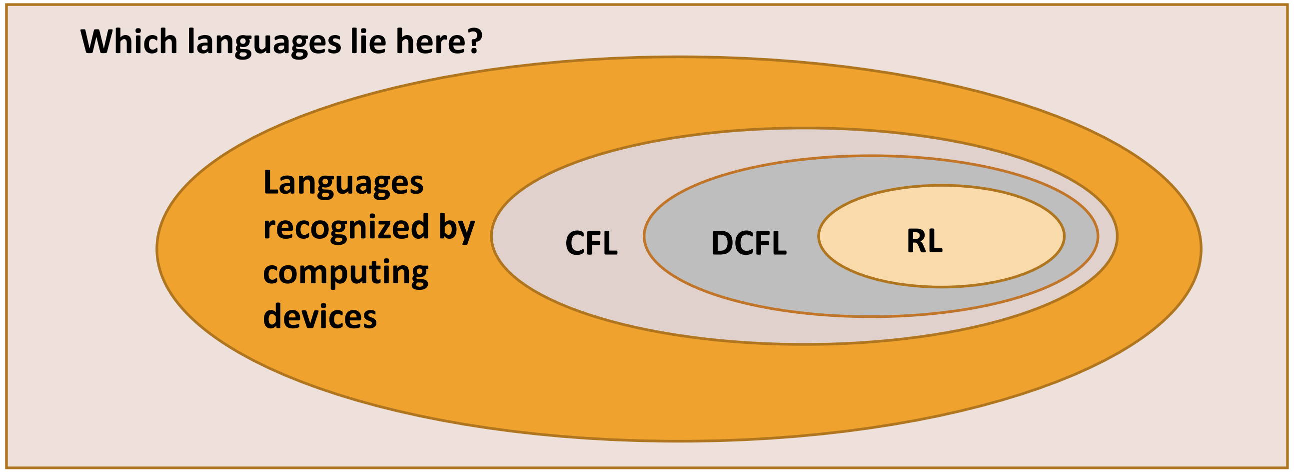


Turing Machines:

- TM **halts** and **accepts/rejects** on reaching the **accept** or **reject** states
- TM may never halt – it may loop forever.

Configuration of a TM: Combination of the current tape contents, the current state and the current head location.

$$X\ 0\ 0\ \underset{\uparrow q_1}{1}\ 1\ 1\ B\ B\ B\ B\ \dots$$



A TM ***M*** **accepts *w*** if there exists a sequence of configurations C_1 to C_k, where

- C_1 is the start configuration M on w.
- Each C_i yields C_{i+1}.
- C_k is an accepting configuration

Variants of Turing Machine models

A TM model $\mathcal{M}_1$ **is equivalent to another model** $\mathcal{M}_2$:

- $\mathcal{M}_2$ **can be simulated by** $\mathcal{M}_1$ **and vice versa**.

Is the standard TM M more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We had a two-way infinite tape, instead of one?
- TM has a printer attached?
- We introduced non-determinism?

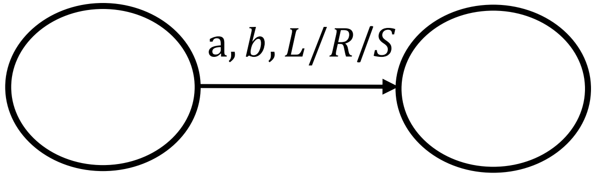
Variants of Turing Machine models

Is the standard TM M more powerful/equivalent to the following TM models where

- **The head can move left, right or stay put?**
- We have k read/write tapes instead of one?
- We had a two-way infinite tape, instead of one?
- TM has a printer attached?
- We introduced non-determinism?

A TM model $\mathcal{M}_1$ is equivalent to another model $\mathcal{M}_2$ if $\mathcal{M}_2$ can be simulated by $\mathcal{M}_1$ and vice versa.

Lazy Turing Machine: The head can either move left, move right or stay put (L, R, S)



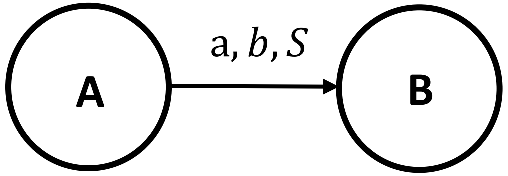
Variants of Turing Machine Models

Is the standard TM M more powerful/equivalent to the following TM models where

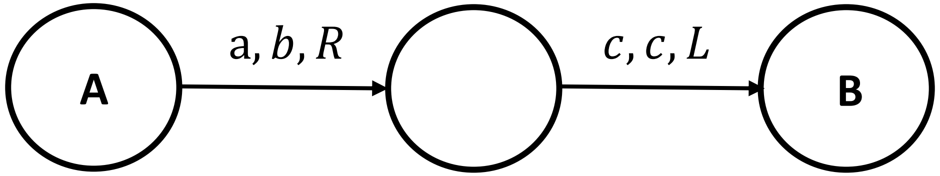
- **The head can move left, right or stay put?**
- We have k read/write tapes instead of one?
- We had a two-way infinite tape, instead of one?
- TM has a printer attached?
- We introduced non-determinism?

A TM model $\mathcal{M}_1$ is equivalent to another model $\mathcal{M}_2$ if $\mathcal{M}_2$ can be simulated by $\mathcal{M}_1$ and vice versa.

Lazy Turing Machine: The head can either move left, move right or stay put (L, R, S)



For each $c \in \Gamma$:



Hence a *lazy Turing machine* model **is equivalent** to a standard Turing Machine model.

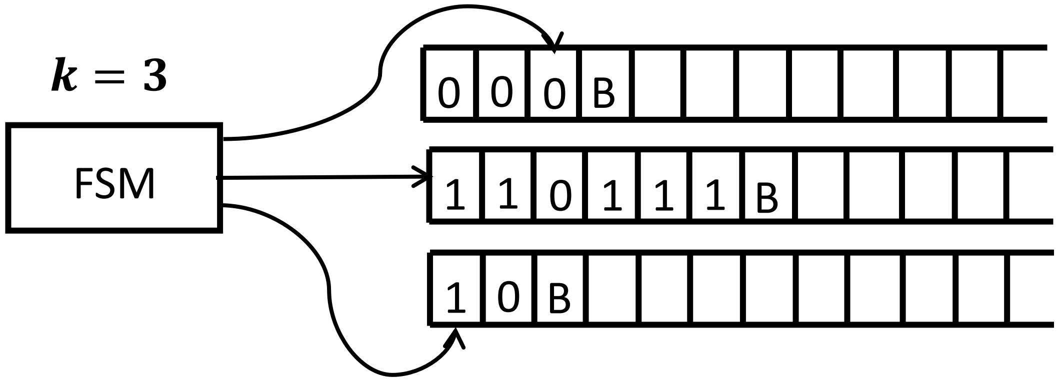
Variants of Turing Machine Models

Is the standard TM M more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- **We have k read/write tapes instead of one?**
- We had a two-way infinite tape, instead of one?
- TM has a printer attached?
- We introduced non-determinism?

A TM model $\mathcal{M}_1$ is equivalent to another model $\mathcal{M}_2$ if $\mathcal{M}_2$ can be simulated by $\mathcal{M}_1$ and vice versa.

k-read/write tapes instead of one: What does a k-tape TM S look like? k-tape TM also has k heads, each associated with a tape. New transition function $\delta_S: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$, i.e. $\delta_S(q_i, a_1, \cdots, a_k) = (q_j, b_1, \cdots, b_k, L, R, \cdots, L)$

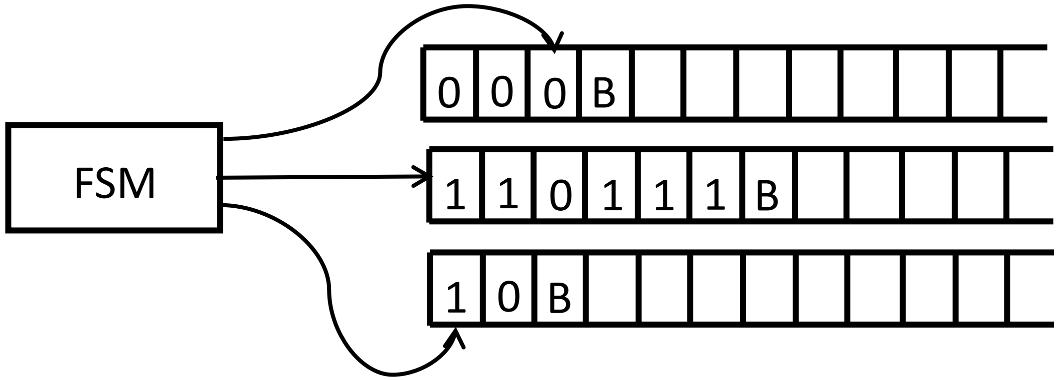


- To simulate S with M, we store the entire information of the k tapes in one single tape.
- M uses \$ to separate the contents of the k tapes.
- To keep track of the locations of the k heads, M marks the symbols where the heads would be, with a '_'.

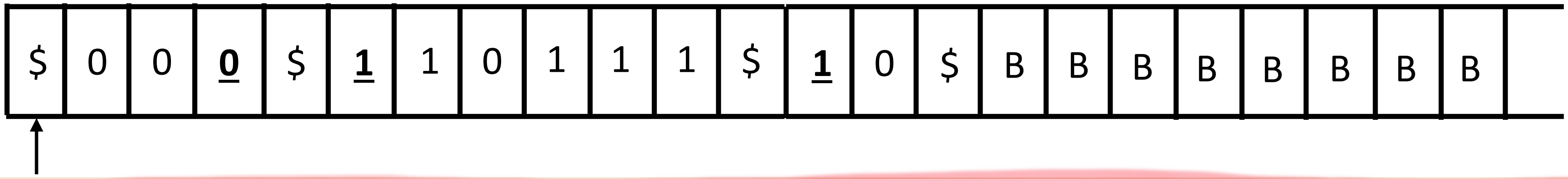
Variants of Turing Machine Models

Is the standard TM M more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- **We have k read/write tapes instead of one?**
- We had a two-way infinite tape, instead of one?
- TM has a printer attached?
- We introduced non-determinism?



k-read/write tapes instead of one: What does a k-tape TM S look like? k-tape TM also has k heads, each associated with a tape. New transition function $\delta_S: Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R\}^k$, i.e. $\delta_S(q_i, a_1, \cdots, a_k) = (q_j, b_1, \cdots, b_k, L, R, \cdots, L)$



- Single tape TM M **first scans the entire tape from leftmost** \$ **to rightmost** \$ ($k+1$ in all) to determine the symbols under the virtual heads. Then it makes a second pass to update the tape according to δ_S.

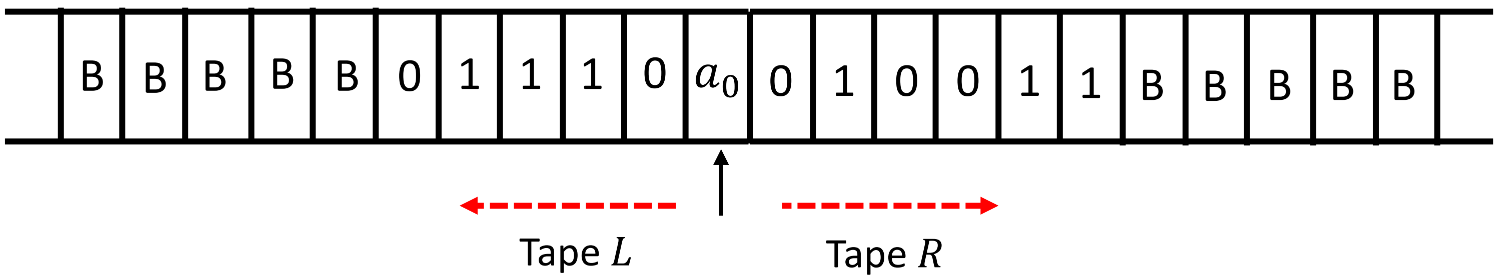
- If it so happens that ***M*'s head needs to go to the right of any of the intermediate \$** $\Rightarrow S$ has moved the head on the corresponding tape to the unread blank symbols. Starting from this cell to the rightmost \$, **shift one cell to the right to make space to write a blank on the empty tape cell** and simulate as before.

Variants of Turing Machine Models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- **We have a two-way infinite tape, instead of one?**
- TM has a printer attached?
- We introduce non-determinism?

Two-way infinite Tape: Let M_D be the TM equipped with this power.



A TM model $\mathcal{M}_1$ is equivalent (as powerful as) to another model $\mathcal{M}_2$ if $\mathcal{M}_2$ can be simulated by $\mathcal{M}_1$ and vice versa.

- Cut the two tapes of M_D into Tape R and $(\text{Tape } L)^R$. We get a two-tape TM.
- Whenever M_D uses the tape to the right of the a_0, Tape R is used.
- When M_D uses the tape to the left of a_0, $(\text{Tape } L)^R$ is used.

M_D isn't any more powerful than a one way infinite tape TM.

So a TM with a two-way infinite tape is equivalent to a TM with a one-way infinite tape.

Variants of Turing Machine models

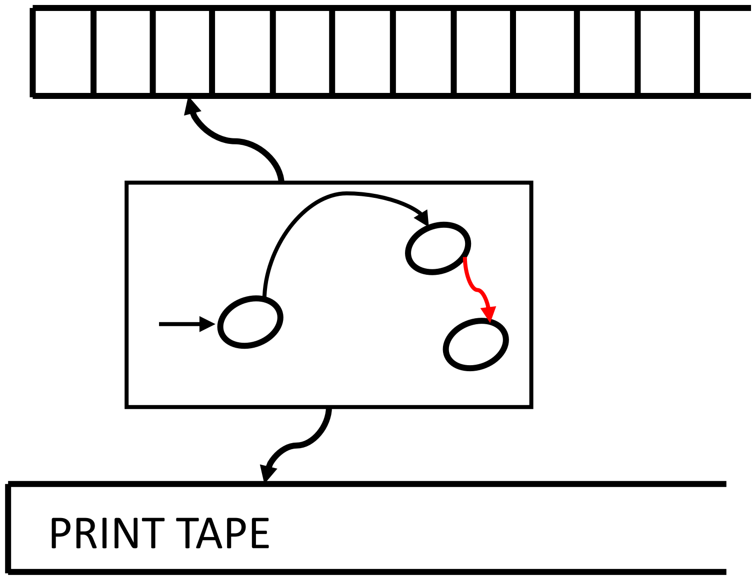
Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **TM with a printer**
- We introduce non-determinism?

Enumerators: TM attached with a printer

- The Enumerator E uses the print tape to output strings
- The input tape is initially blank
- The language of E is the set of strings that it prints out
- If E does not halt, it may print infinitely many strings in some order

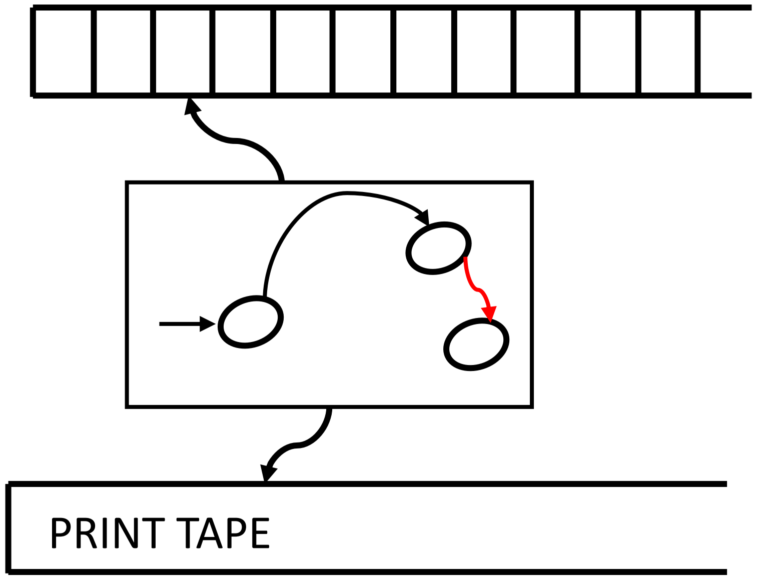
$$\mathcal{L}(E) = \{\text{w} \in \Sigma^* | \text{w is printed by E}\}$$



Variants of Turing Machine models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **TM with a printer**
- We introduce non-determinism?



Enumerators: TM attached with a printer

If $L(M)$ is the language recognized by a Turing Machine M then there exists an enumerator E that enumerates it.

The set of all finite length (binary) strings is **countably infinite**.

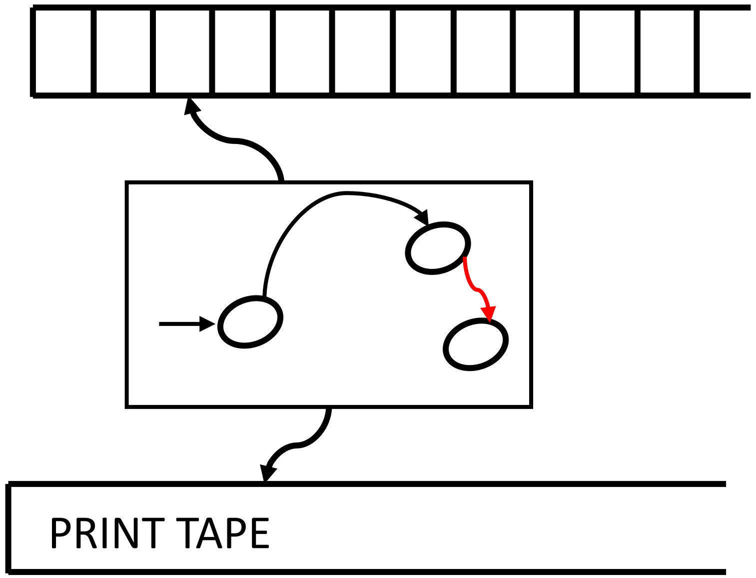
- Lexicographically generate all binary strings one after the other. There exists a one-one correspondence with $\mathbb{N}$.
- We can lexicographically generate all (binary) strings and number them:

$$s_1 = 0, s_2 = 1, s_3 = 00, \cdots$$

Variants of Turing Machine models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **TM with a printer**
- We introduce non-determinism?



Enumerators: TM attached with a printer

If $L(M)$ is the language recognized by a Turing Machine M then there exists an enumerator E that enumerates it.

Proof:

For $i = 1, 2, \cdots$

For $j = 1, 2, \cdots, i$

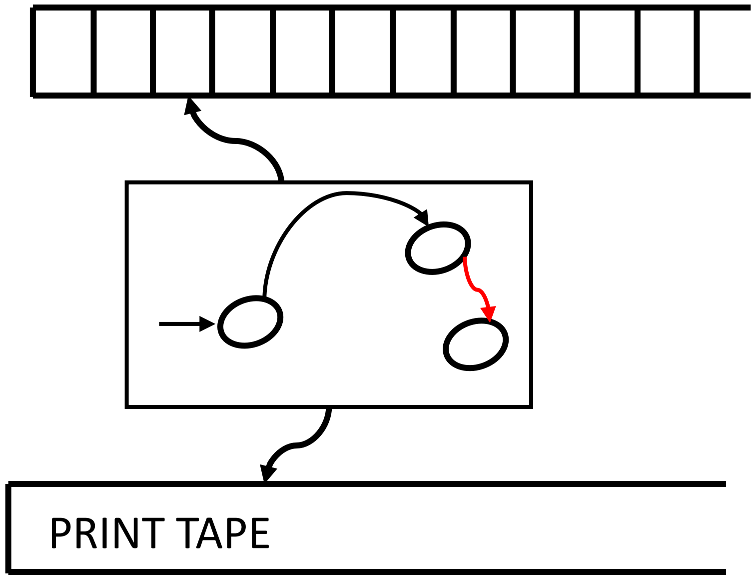
Run M with string s_j for i steps.

If any string is accepted, then PRINT it.

Variants of Turing Machine models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **TM with a printer**
- We introduce non-determinism?



Enumerators: TM attached with a printer

If there exists an Enumerator E, then there exists a Turing Machine M such that $L(M) = L(E)$.

Proof:

$M =$ On input w:

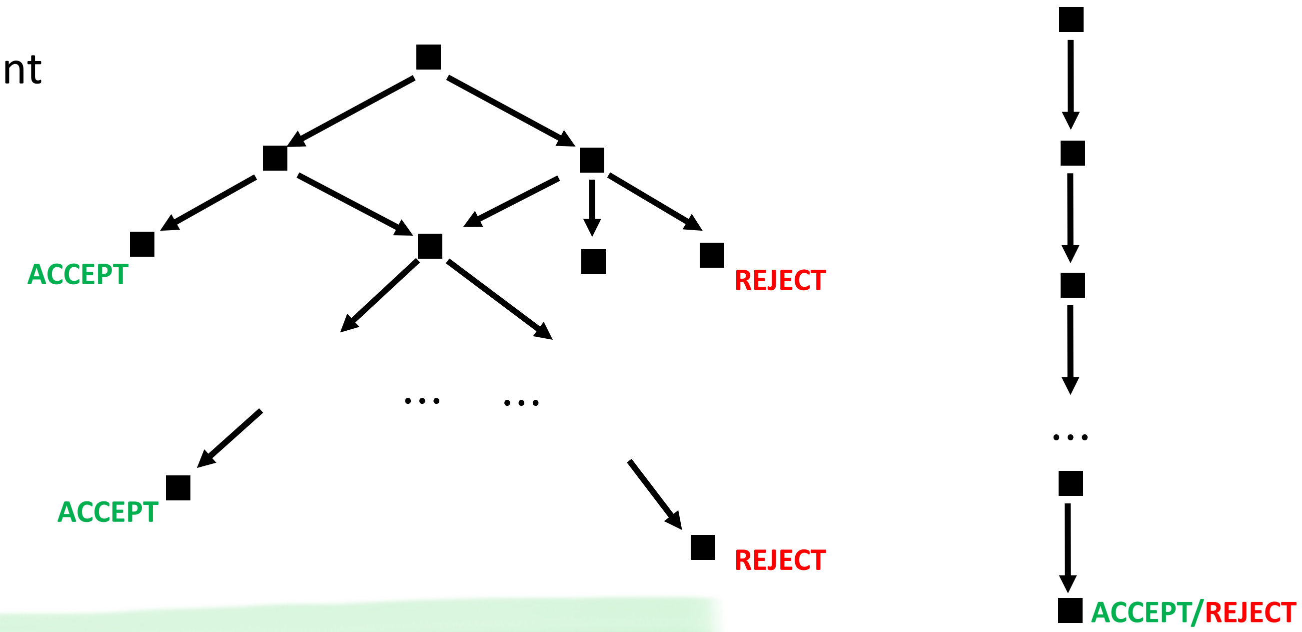
1. Run E. Every time E prints some string, compare it with w.
2. If they match, ACCEPT.

E and M are equivalent

Variants of Turing Machine Models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **We introduce non-determinism?**



Non-deterministic Turing Machines (NTM):

- For a DTM, from a given configuration, exactly one configuration available to it at any stage.
- For an NTM, any point in the computation, several possible configurations are available.
- Its transition function is $\delta_N: Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\})$, i.e. $\delta(q_i, a) \rightarrow \{(q_j, b, R), (q_k, c, L) \cdots\}$
- The computation corresponds to a configuration tree: From the starting configuration, the computation has several branches, each of which leads to a different configuration.
- If any branch of the computation, leads to an accepting configuration, the NTM accepts. Immediately, DTM is a special case of NTM.

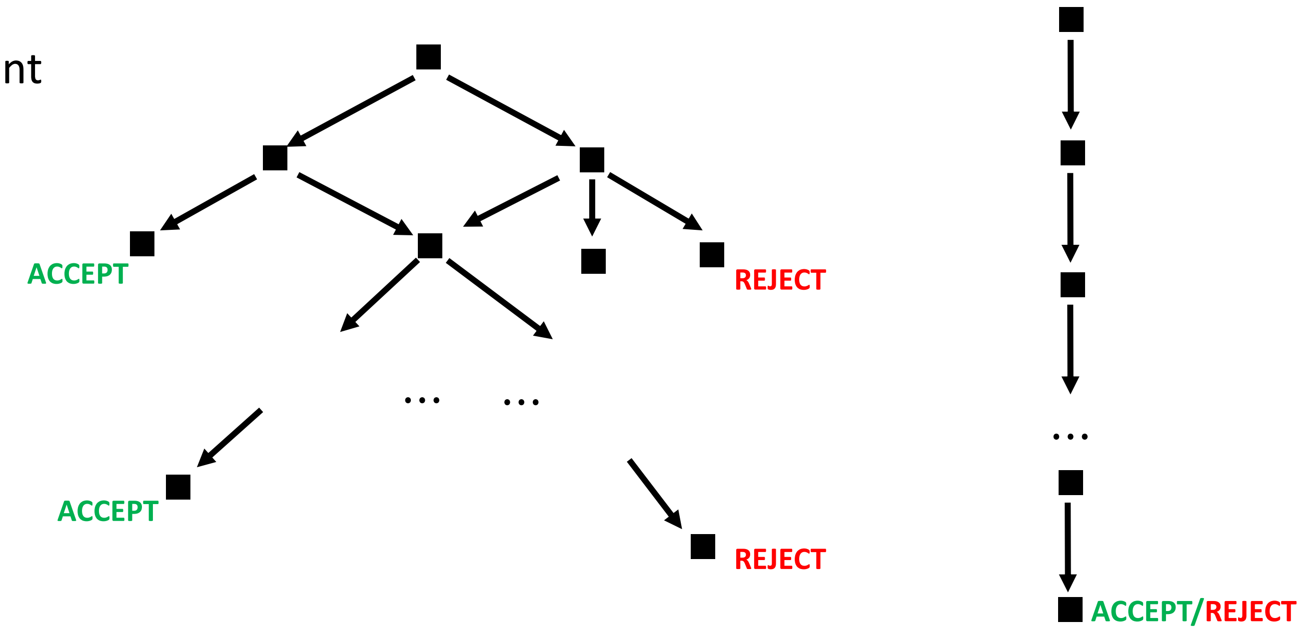
Are NTMs more powerful than DTMs? No. Any NTM can be simulated by a DTM. How?

Variants of Turing Machine Models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **We introduce non-determinism?**

Any NTM can be simulated by a DTM



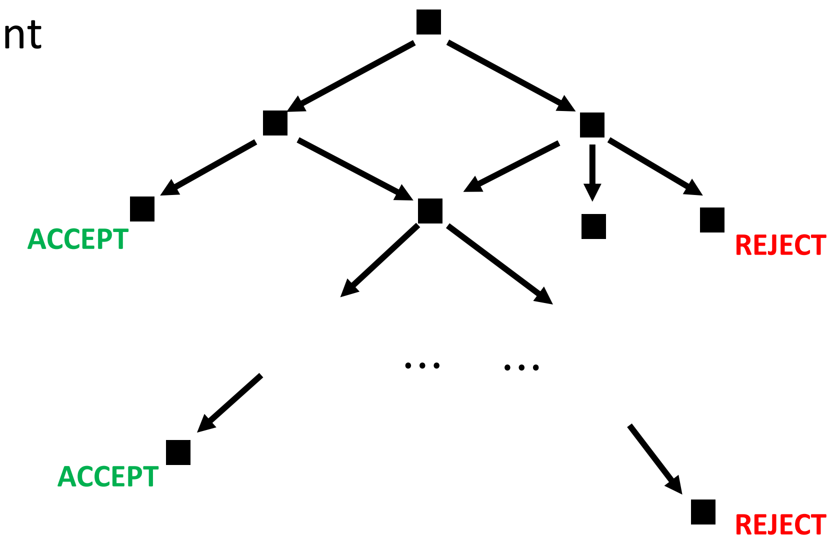
- The DTM searches from among the configurations of the NTM for an accepting configuration.
- Clearly an ordinary Depth First Search shouldn't work
- A branch of the configuration tree can be of infinite depth (when the TM loops forever for that sequence of configuration) and hence the DTM can miss a much shorter accepting configuration.

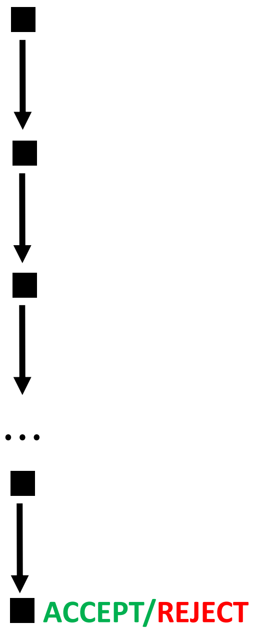
Variants of Turing Machine Models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **We introduce non-determinism?**

Any NTM can be simulated by a 3-tape DTM





Input string w

Generate runs lexicographically

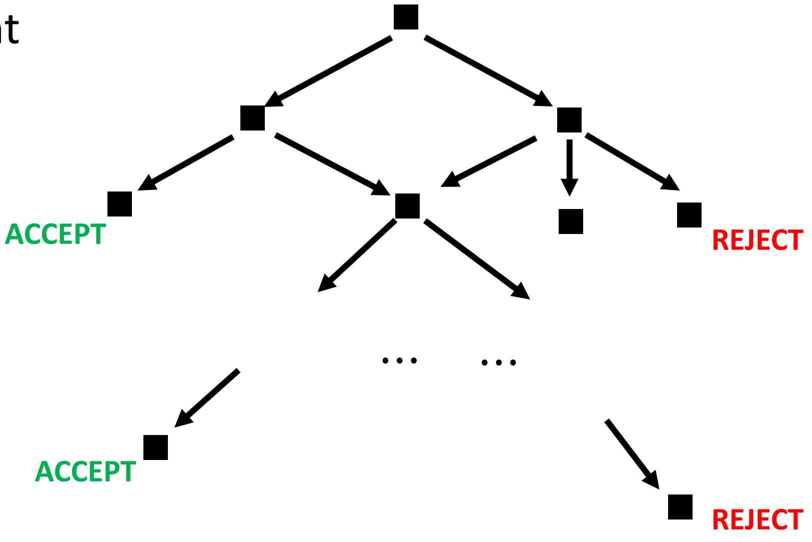
Simulate the run for i/p w

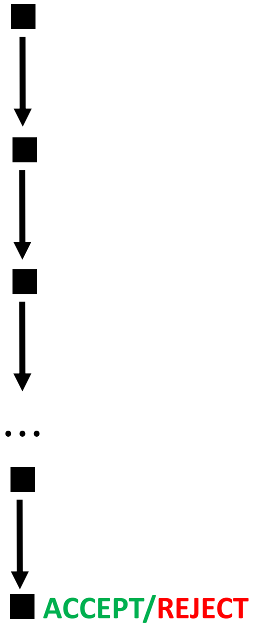
- Tape 1 holds the input string w.
- As for the content of Tape 2, note that we can always obtain a bound for the maximum number of nodes at any level of the configuration tree (say b).
- Let $C = \{1,2,\cdots,b\}$, then we can define a run by a string $s \in C$. E.g: 122: choose the first node from level 1, second node from level 2, third node from level 3.

Variants of Turing Machine Models

Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **We introduce non-determinism?**





Any NTM can be simulated by a 3-tape DTM

$1100011BB$

121

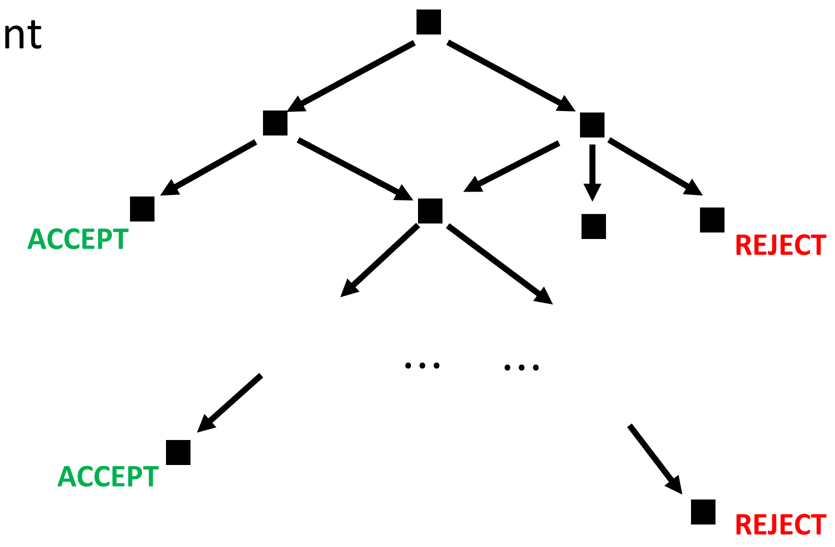
Simulate the run for i/p w

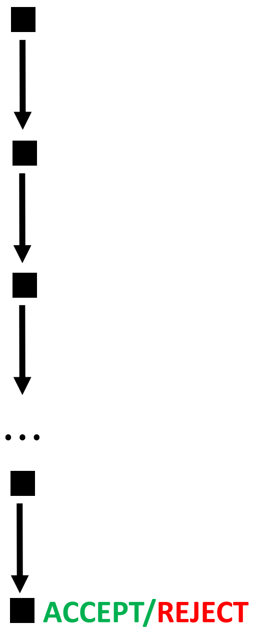
- Tape 1 holds the input string w.
- Tape 2 generates a string in $C = \{1,2,\cdots,b\}$ lexicographically: Generate all strings of length 1, then strings of length 2 and so on, i.e. $\{1,2,\ldots,b,11,12,21,22,\ldots\}$.
- Some of these runs may be invalid or too short to lead to any accept/reject state.

Variants of Turing Machine Models

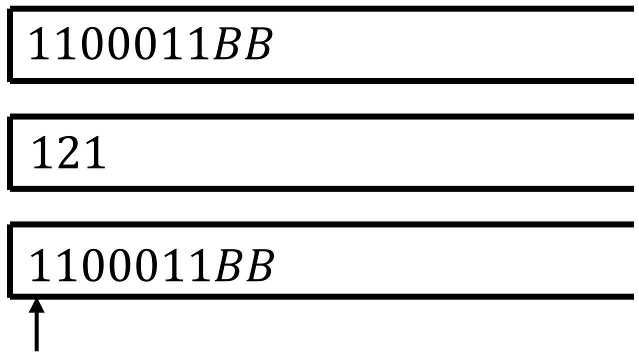
Is the standard TM model $\mathcal{M}_1$ more powerful/equivalent to the following TM models where

- The head can move left, right or stay put?
- We have k read/write tapes instead of one?
- We have a two-way infinite tape, instead of one?
- **We introduce non-determinism?**





Any NTM can be simulated by a 3-tape DTM



- Tape 1 holds the input string w.
- Tape 2 generates runs lexicographically
- Tape 3 simulates one branch of the configuration tree corresponding to the run generated in Tape 2. At each step of the simulation consult Tape 2 to decide the next configuration.
- If the simulation leads to an accept state – accept the computation
- During the simulation, if the run in Tape 2 is found to be invalid, abort and generate the next lexicographic string

3-tape DTM $\equiv$ 1-tape DTM $\Rightarrow$ NTM $\equiv$ DTM

Variants of Turing Machine Models

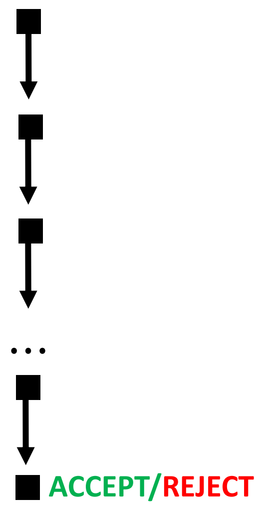
Variants of TM:

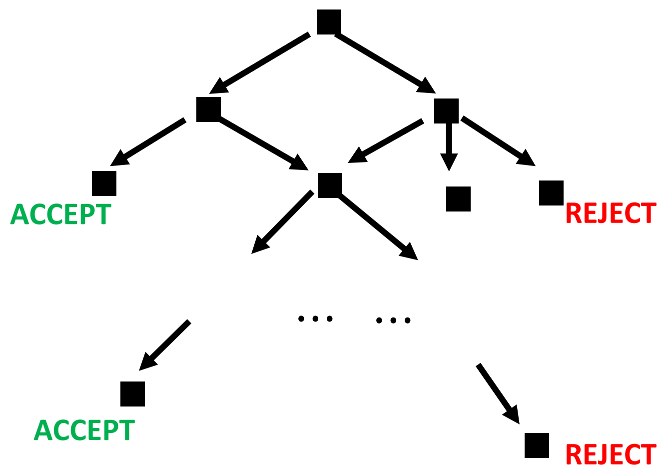
- **The head can move left, right or stay put?**
- **We have k read/write tapes instead of one?**
- **We have a two-way infinite tape, instead of one?**
- **TM with a printer attached?**
- **We introduce non-determinism?**

Key takeaway: A standard TM is quite robust. Adding extra power seems to make no difference in computing power

Variants of Turing Machine Models

- As an aside, although $NTM \equiv DTM$, a DTM may require several more steps to perform the same computation.
- For a moment, consider problems that are computable (TM halts on all inputs).
- For a given decision problem L, let for all input strings $|w| = n$, suppose $\exists$
- **DTM that halts within $\boldsymbol{DTIME}$ steps and outputs YES if $\boldsymbol{w \in L}$ and NO if $\boldsymbol{w \notin L}$.**





Variants of Turing Machine Models

- As an aside, although $NTM \equiv DTM$, a DTM may require several more steps to perform the same computation.
- For a moment, consider problems that are computable (TM halts on all inputs).
- For a given decision problem L, let for all input strings $|w| = n$, suppose $\exists$
- **DTM that halts within $\boldsymbol{DTIME}$ steps and outputs YES if $\boldsymbol{w} \in \boldsymbol{L}$ and NO if $\boldsymbol{w} \notin \boldsymbol{L}$.**

$\boldsymbol{P}$ = Set of all problems for which $DTIME$ is a polynomial in n
(e.g.: $n^{1000} + 3n^2$)

Variants of Turing Machine Models

- As an aside, although $NTM \equiv DTM$, a DTM may require several more steps to perform the same computation.
- For a moment, consider problems that are computable (TM halts on all inputs).
- For a given decision problem L, let for all input strings $|w| = n$, suppose $\exists$
- **NTM that halts within $\boldsymbol{NTIME}$ steps and outputs YES if $\boldsymbol{w \in L}$ and NO if $\boldsymbol{w \notin L}$.**

$\boldsymbol{NP} =$ Set of all problems for which $NTIME$ is a polynomial in n.

Variants of Turing Machine Models

- For a given decision problem L, let for all input strings w such that $|w| = n$, suppose $\exists$
- **DTM that halts within $\boldsymbol{DTIME}$ steps and outputs YES if $\boldsymbol{w \in L}$ and NO if $\boldsymbol{w \notin L}$.**
- **NTM that halts within $\boldsymbol{NTIME}$ steps and outputs YES if $\boldsymbol{w \in L}$ and NO if $\boldsymbol{w \notin L}$.**

$\boldsymbol{P}$ = Set of all problems for which $DTIME$ is a polynomial in n.

(e.g.: $n^{1000} + 3n^2$)

$\boldsymbol{NP}$ = Set of all problems for which $NTIME$ is a polynomial in n.

Clearly, $\boldsymbol{P \subseteq NP}$. However, is $\boldsymbol{P = NP}$?

A million dollar problem

Thank You!