Nondeterministic TM I

- $\delta$:
  \[ \delta : Q \times \Gamma \to P(Q \times \Gamma \times \{L, R\}) \]
  
  $P$: power set

- Example:
  
  \[
  q_0, a \rightarrow q_1, b, R \\
  \rightarrow q_2, c, L
  \]

- What if
  
  \[
  q_0, a \rightarrow q_{accept} \\
  \rightarrow q_{reject}
  \]
For NTM, $w$ accepted if one branch works
In this sense, unless all branches are finite
NTM $\rightarrow$ accept or endless loop
NTM: like an "acceptor"
Example of NTM 1

- $A = \{w \mid w \text{ contains } aab\}$
- State diagram
Example of NTM II

\[ a, b \rightarrow a, b, R \]

- \( q_0 \)
- \( a \rightarrow R \)
- \( b \rightarrow R \)
- \( \square \rightarrow R \)

- \( q_1 \)
- \( a \rightarrow R \)
- \( b \rightarrow R \)
- \( \square \rightarrow R \)

- \( q_r \)
- \( a \rightarrow R \)
- \( \square \rightarrow R \)

- \( q_2 \)
- \( b \rightarrow R \)
- \( \square \rightarrow R \)

- \( q_a \)
- \( b \rightarrow R \)
Example of NTM III

- You may recall that this is an NFA example discussed before
- Only the first node is nondeterministic
Example of NTM I

- \( L = \{0^n \mid n \text{ composite number}\} \)
- From p. 204 of Lewis and Papadimitriou
- Composite number: product of two natural numbers
- Procedure
  - Nondeterministically choose \( p \) and \( q \)
  - Sequentially try \( p \) from 2 to \( n - 1 \)
  - Check if \( n = pq \)
    - This can be done by the earlier example

\[ \{a^n b^p c^q \mid n = p \times q\} \]
Example of NTM II

- Question: details about “non-deterministically” choose $p$ and $q$?
- If we sequentially try all $(p, q)$ combinations, then looks like we have a deterministic setting?
- Our generation of $p$ and $q$ can be non-deterministic
- Say we do a copy operation to generate $p$ elements. The TM can have an $\epsilon$ link to stop at any time point
Nondeterministic TM ≡ deterministic TM

- easy
  A deterministic TM is a nondeterministic TM
- more difficult
- Like NFA we use a tree for processing the input (# branches finite)
- To traverse a tree we can do
  depth-first search
  or
  breadth-first
Nondeterministic TM $\equiv$ deterministic TM

- If using depth-first search, one branch may lead to $\infty$ steps
  
  Then we cannot consider other branches even if the input is accepted

- Thus we should consider breadth-first

- Fig 3.17: a deterministic TM to simulate a nondeterministic TM
Nondeterministic TM $\equiv$ deterministic TM

- **Tape 1:** input, never altered

![Diagram showing CPU and tapes](#)
Nondeterministic TM $\equiv$ deterministic TM

- Tape 2: process one branch
- Tape 3: maintain the tree
- The key is the 3rd tape
- Suppose max $\#$ branches 3
  - At the 1st step: if contents of 3rd tape are 1 2
    $\Rightarrow$ can go to 1 or 2 from $q_0$
- The tree keeps growing. For example,
  1 2 12 13 2 12 13 21 22 23 121 123 13 21 22 23
Nondeterministic TM $\equiv$ deterministic TM

- What if 12 is a failed branch?
  12 13 21 22 23 12 131 132 21 22 23

- 12 fails, continue 131, no need to remove 12
Corollary 3.19

Definition: NTM is a decider if all branches halt on all inputs.

Language decidable ⇔ some NTM decides it.

⇒ easy, one TM decides it and TM is an NTM. This TM halts on all inputs (one branch).

⇐:
Now NTM terminates on all branches.

We will construct a TM to accept the language.

each branch is finite
every input halts ∃ a finite max length
Corollary 3.19 II

- # branches finite at each node
- The tree to process this input is finite
- Write it in the 3rd tape
- We know a multi-tape TM is equivalent to a single-tape TM