A language recognized by TM \(\Rightarrow\) recognized by NTM

A deterministic TM is a nondeterministic TM

A language recognized by NTM \(\Rightarrow\) recognized by TM

more difficult

We must simulate NTM by TM

How did we run NTM?
Like NFA we use a tree for processing the input (number of branches finite)

- To traverse a tree we can do
  - depth-first search
  - breadth-first

- If using depth-first search, one branch may lead to an infinite number of steps
Nondeterministic TM ≡ deterministic TM

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
Nondeterministic TM $\equiv$ deterministic TM

- Tape 2: copy input from tape 1 and run one branch up to certain layer
- 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,
Nondeterministic TM $\equiv$ deterministic TM

What if 12 is a failed branch?

12 13 21 22 23
12 13 21 22 23
12 131 132 21 22 23

12 fails, continue 131, no need to remove 12
Therefore, an NTM can be simulated by a three-tape TM.

We have shown that a multi-tape TM can be simulated by a single-tape TM.

Thus the proof is completed.
Corollary 3.19 I

- Definition: NTM is a decider if all branches halt on all inputs
- Language decidable ⇔ some NTM decides it
- ⇒ easy, one TM decides it and a TM is an NTM
  This TM halts on all inputs (one branch)
- ⇐:
  Now NTM terminates on all branches
  We can construct a TM to decide 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
- Thus the three-tape TM used earlier can accept/reject the input in a finite number of steps