# CS310

# Variants of Turing Machines

## Section 3.2

November 10, 2014

# Formal Definition

- 7 Tuple:

# Multiple Tape Turing Machine

- For k tapes
  - input string is on tape 1
  - other tapes start out blank

- Change

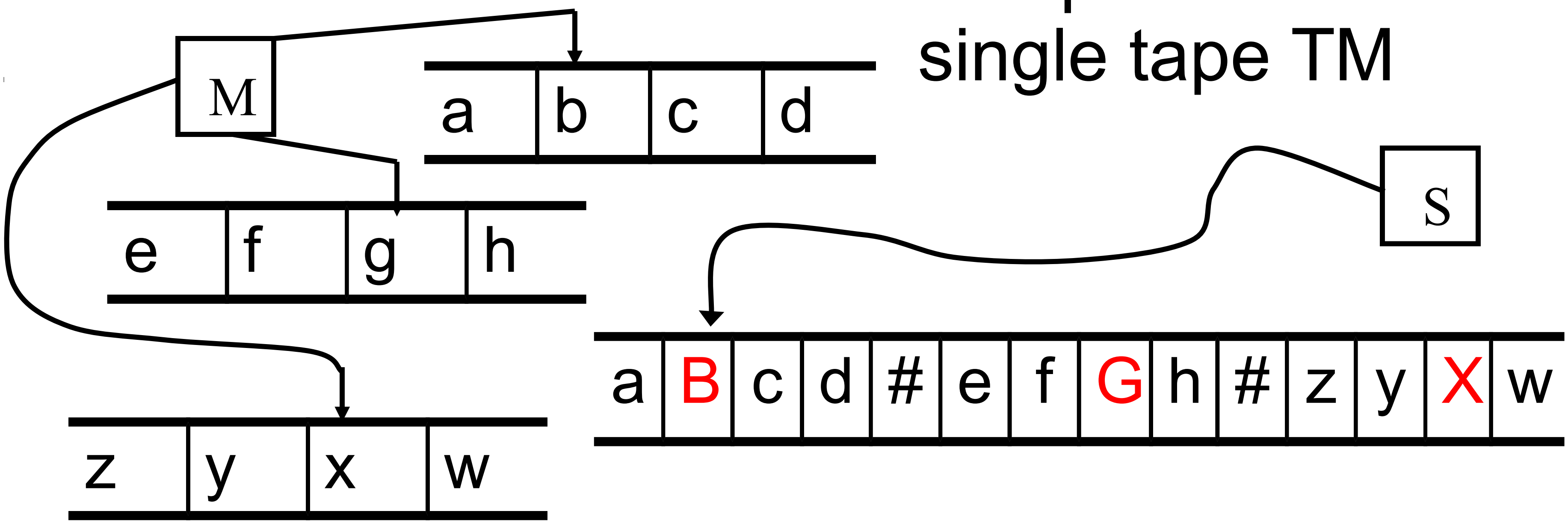
$\delta$: Q x $\Gamma \rightarrow$ Q x $\Gamma$ x {L, R}

# Example

- Construct a two-tape Turing Machine to accept $L=\{a^n b^n \mid n \geq 1\}$
- Conceptually what do we want to do?

# Theorem

- Every multi-tape Turing Machine has an equivalent single tape machine
  - *adding extra tapes does not add power to the Turing Machine*
- Proof Idea: Simulate multi-tape TM as single tape TM



# Nondeterministic TM

- Often easier to design/understand
- Design a TM to accept strings containing a *c* that is either preceded or followed by *ab*
- We can think of this computation as a tree
  – each branch from a node (state) represents one nondeterministic decision (for a single input character)

# Theorem

- Every nondeterministic TM, N, has an equivalent deterministic TM, D

- Proof Idea:
  – use a 3 tape TM (we can convert this to a one tape TM later)
  – tape 1: input tape (read-only)
  – tape 2: simulation input/output tape of the current branch of the n-d TM
  – tape 3: address tape (based on the tree) to keep track of where we are in the computation

# Practice

$\{ a^i b^j c^k \mid i > j > 0;\ k = 2i \}$

$\{ ww^R \mid \ |ww^R| \text{ is odd },\ w \in \{0,1\}^* \}$

$\{ ww \mid w \in \{0,1\}^* \}$

the complement of $\{ww^R \mid w \in \{0,1\}^* \}$

multiplication of two numbers in base 1:

11111 * 11 produces 1111111111