Turing Machine

Dr. Yangjun Chen

- Deterministic Turing Machine
- Nondeterministic Turing Machine

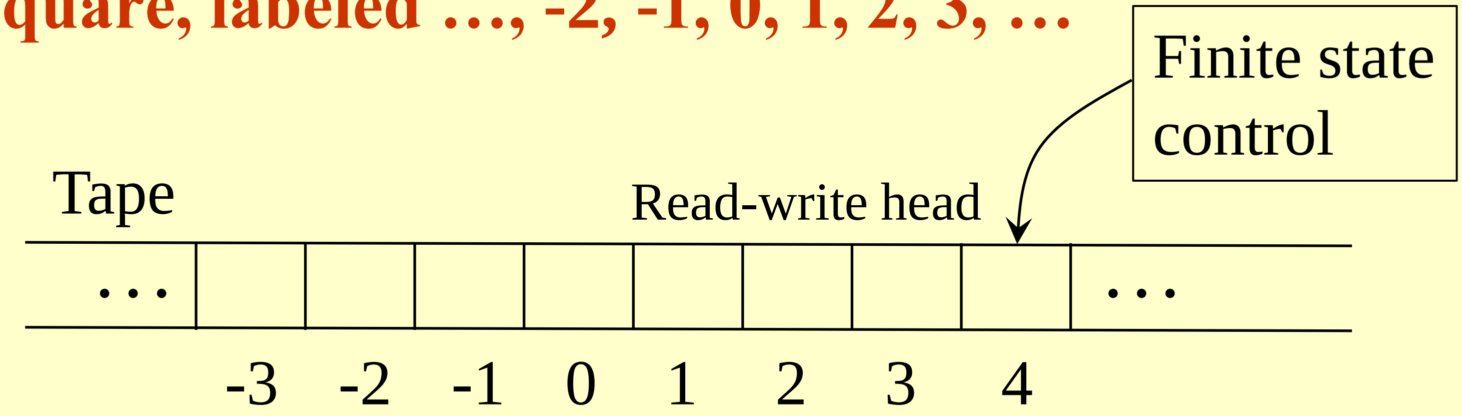
Deterministic Turing Machine

- **In order to formalize the notion of an algorithm, we will need to fix a particular model for computation.**
- **One of them is the deterministic one-tape Turing machine (DTM for short). It consists of**

  - a finite state control,

  - a read-write head, and

  - a tape made up of a two-way infinite sequence of table square, labeled …, -2, -1, 0, 1, 2, 3, …



A Program for a DTM

A program for a DTM specifies the following information:

(1) A finite set Γ of tape symbols, including a subset $\Sigma \subset \Gamma$ of input symbols and a distinguished blank $b \in \Gamma - \Sigma$;

(2) State set Q, initial state q_0, halt states q_y and q_n ; and

(3) Transition function $\delta: (Q - \{q_y, q_n\}) \rightarrow Q \times \Gamma \times \{l, r\}$.

The operation of such a program is straihtforward. The *input* to the DTM is a string $x \in \Sigma^*$. The string x is placed in tape squares 1 through $|x|$, one symbol per square.

DTM

- All other squares initially contain the blank symbol.
- The program starts its operation in state q_0, with the read-write head scanning tape square 1.
- The computation then proceeds in a step-by-step manner.
- If the current state q is either q_y or q_n , then the computation has ended, with the answer being 'yes' if $q = q_y$ and 'no' if $q = q_n$.
- Otherwise, the current state q belongs to $Q - \{q_y$, $q_n\}$, some symbol $s \in \Gamma$ is in the tape square being scanned, the value if $\delta(q, s)$ is defined.

DTM

Example

$$Q = \{q_0, q_1, q_2, q_3, q_r, q_N\}$$

$$\Gamma = \{0, 1, b\}, \Sigma = \{0, 1\}$$

$$\delta: (q_0, 0) \rightarrow (q_0, r); (q_0, 1) \rightarrow (q_0, r); (q_0, b) \rightarrow (q_1, l);$$

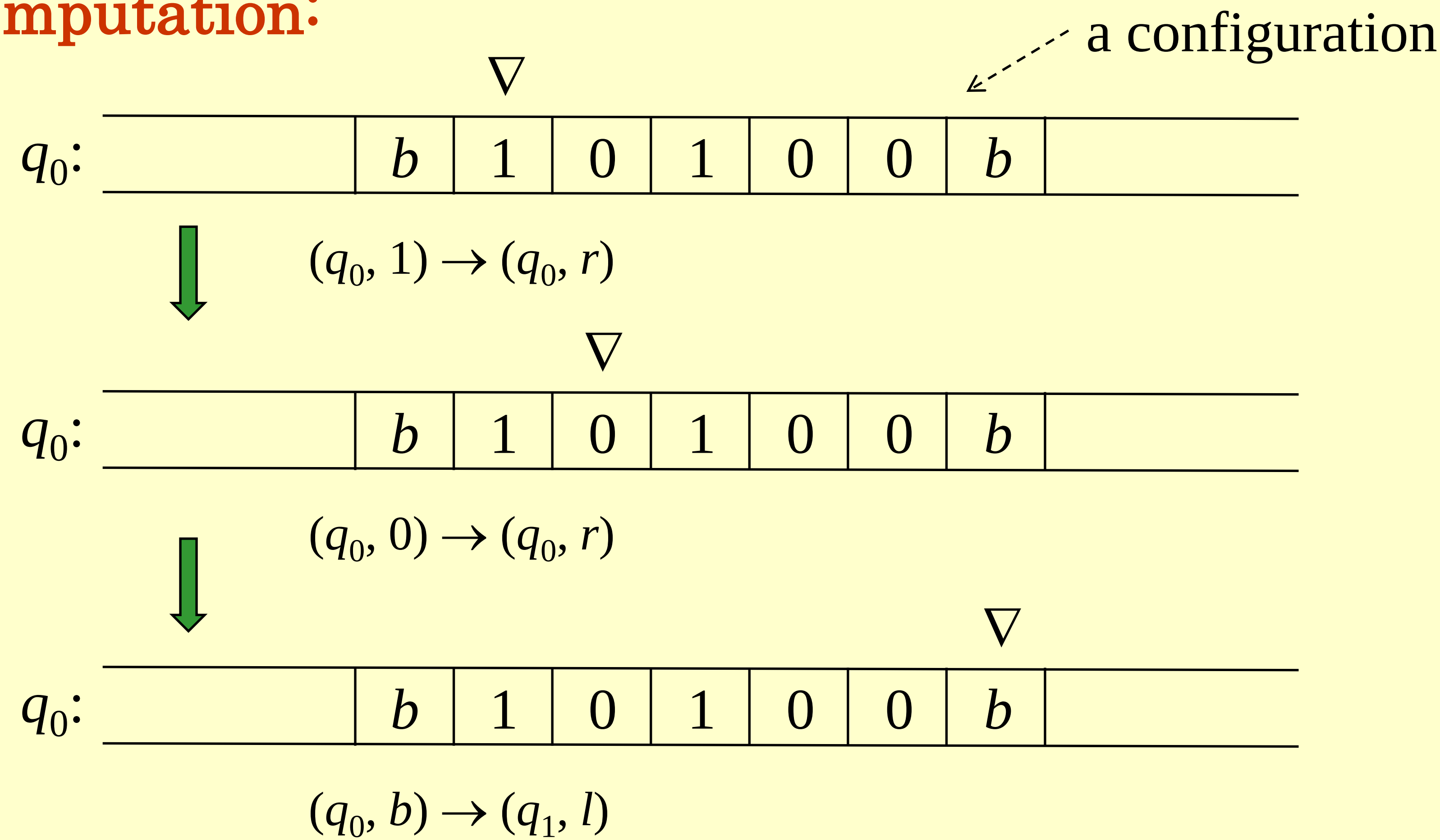
$$(q_1, 0) \rightarrow (q_2, b); (q_1, 1) \rightarrow (q_3, b); (q_1, b) \rightarrow (q_n, b);$$

$$(q_2, 0) \rightarrow (q_y, b); (q_2, 1) \rightarrow (q_n, b); (q_2, b) \rightarrow (q_2, l);$$

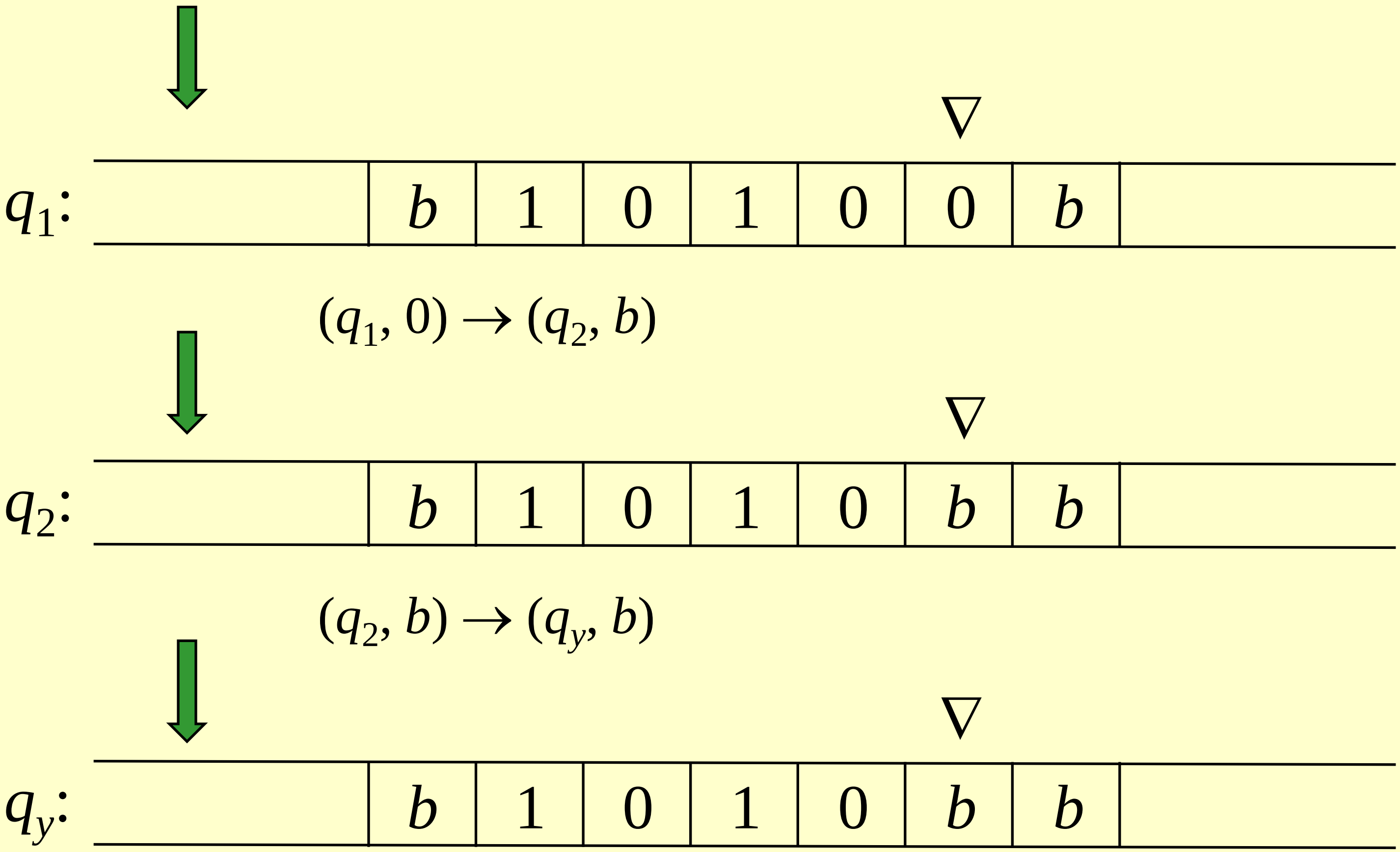
$$(q_3, 0) \rightarrow (q_n, b); (q_3, 1) \rightarrow (q_n, b); (q_3, b) \rightarrow (q_3, l).$$

DTM

On the input $x = 10100$, we have the following computation:



DTM



DTM

- The above example illustrates the computation of a simple DTM M on an input $x = 10100$, giving the state, head position, and the contents of the non-blank portion of the tape before and after each step.
- Note that this computation halts after six steps, in state q_y, so the answer for 10100 is ‘yes’.
- In general, we say that a DTM program M with input alphabet Σ accepts $x \in \Sigma^*$ if and only if M halts in state q_y when applied to input x. The language L_M is given by

$$L_M = \{x \in \Sigma^*: M \text{ accepts } x\}.$$

DTM

For an input $x \in \Sigma^*$, there are three possibilities:

(1) M accepts *x*, halting at state q_y.

(2) M does not accept *x*, halting at state q_n.

(3) M continues forever without halting.

For a DTM program to correspond to the concept of an algorithm, it must halt on all possible strings over its input alphabet.

P-class of problem:

$P \equiv \{L \subset \{0, 1\}^* |$ L can be accepted by a M in polynomial time. $\}$

NDTM

Nondeterministic Turing Machin - NDTM

A program for a NDTM specifies the following information:

(1) A finite set Γ of tape symbols, including a subset $\Sigma \subset \Gamma$ of input symbols and a distinguished blank $b \in \Gamma - \Sigma$;

(2) State set Q, initial state q_0, halt states q_r and q_N; and

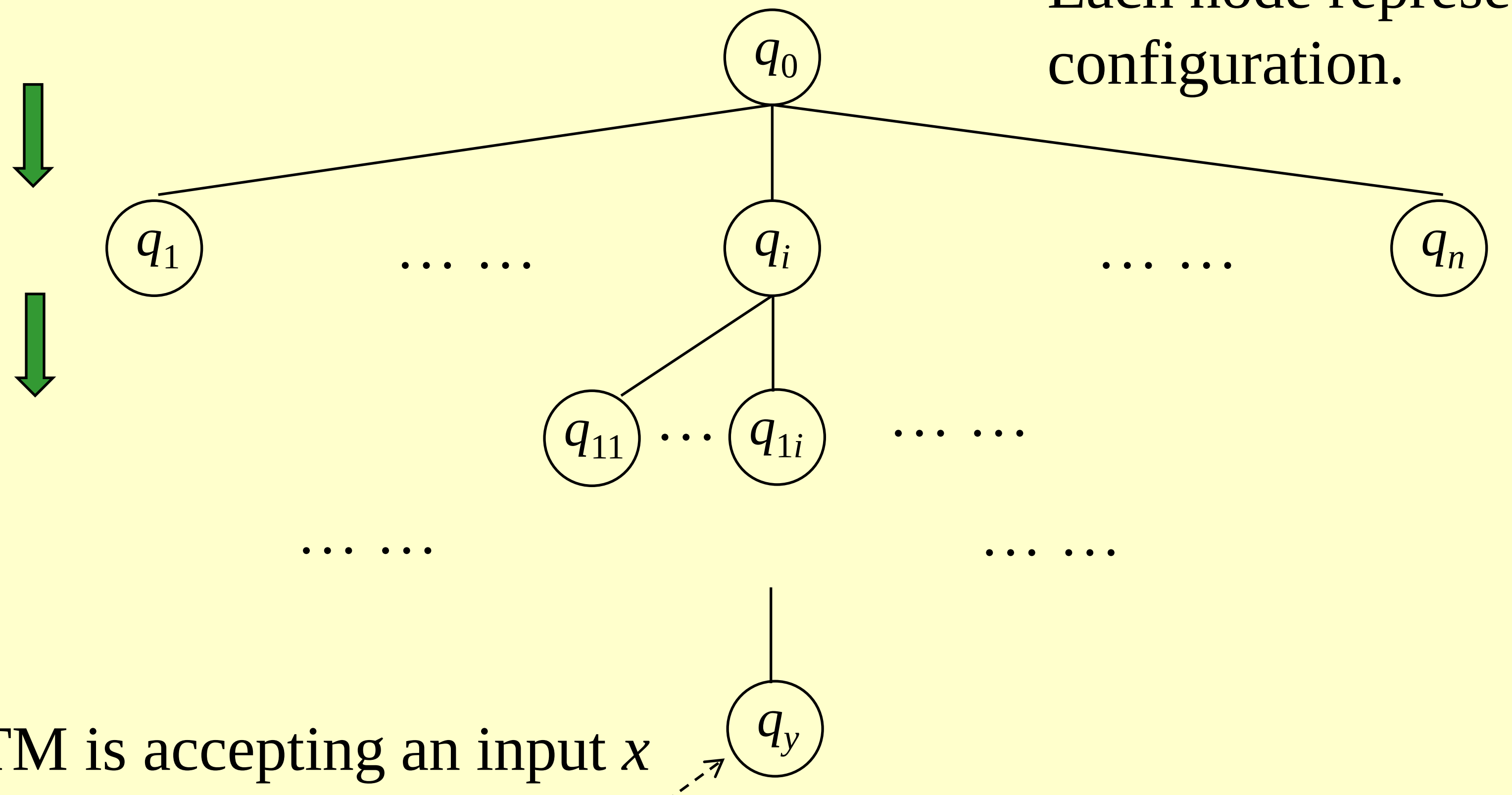
(3) Transition function δ: $(Q - \{q_y, q_n\}) \rightarrow Q \times \Gamma \times \{l, r\}$.

(4) At any state, the machine can be transferred from a configuration to multiple successor configurations.

NDTM

The computation of NDTM on an input x can be illustrated as an tree of size exponential in $|x|$.

Each node represents a configuration.



NDTM is accepting an input x iff some sequence of choices goes to q_y.

NDTM

NP-class of problem:

$NP \equiv \{L \subset \{0, 1\}^* |$ L can be accepted by an *NDTM* in polynomial time. $\}$