CS311: Computational Theory

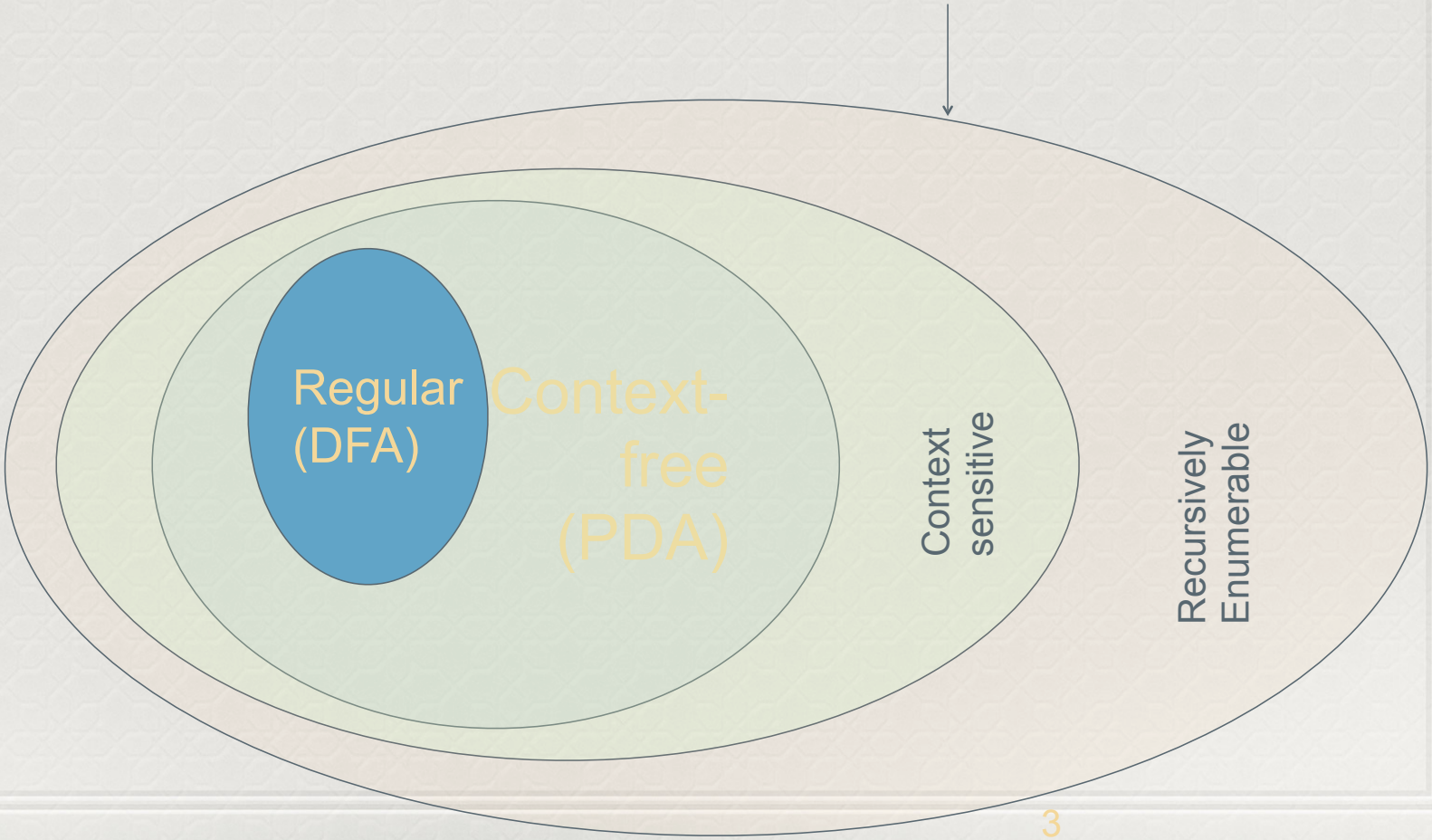
Lecture 10: THE CHURCH-TURING THESIS – Ch 3

Lecture Learning Objectives

1. *Explain the Church-Turing thesis and its significance.*

Language of the Turing Machines

- ***Recursive Enumerable (RE) language***



Introduction

- *American mathematician Alonzo Church created a method for defining functions called the λ-calculus,*
- *British mathematician Alan Turing created a theoretical model for machines, now called Turing machines, that could carry out calculations from inputs,*
- *Austrian-American mathematician Kurt Gödel, with Jacques Herbrand, created a formal definition of a class of functions whose values could be calculated by recursion.*
- *All three computational processes (recursion, the λ-calculus, and the Turing machine) were shown to be equivalent—all three approaches define the same class of functions. This has led mathematicians and computer scientists to believe that the concept of computability is accurately characterized by these three equivalent processes. [wikipedia]*

Church–Turing Thesis

- *Informally, the Church–Turing thesis states that if some method (algorithm) exists to carry out a calculation, then the same calculation can also be carried out by a Turing machine (as well as by a recursively definable function, and by a λ-function).*

Solving a Problem

- To solve a decision problem, an algorithm has to accept each instance of the problem as input, and return "Yes" or "No" depending on whether the instance is a Yes-instance.

- To solve a function problem, an algorithm has to accept each instance I of the problem as input, and return the unique sol(I).

- To solve a search problem, an algorithm has to accept each instance I of the problem as input, and return either an element of sol(I) or "No" if sol(I) is empty (the instance has no solutions).
- We interpret "algorithm" as "computable function."

Turing Machines are…

- *Very powerful (abstract) machines that could simulate any modern day computer (although very, very slowly!)*
- *Why design such a machine?*
  - *If a problem cannot be "solved" even using a TM, then it implies that the problem is* ***undecidable***
- *Computability vs. Decidability*

For every input, answer YES or NO

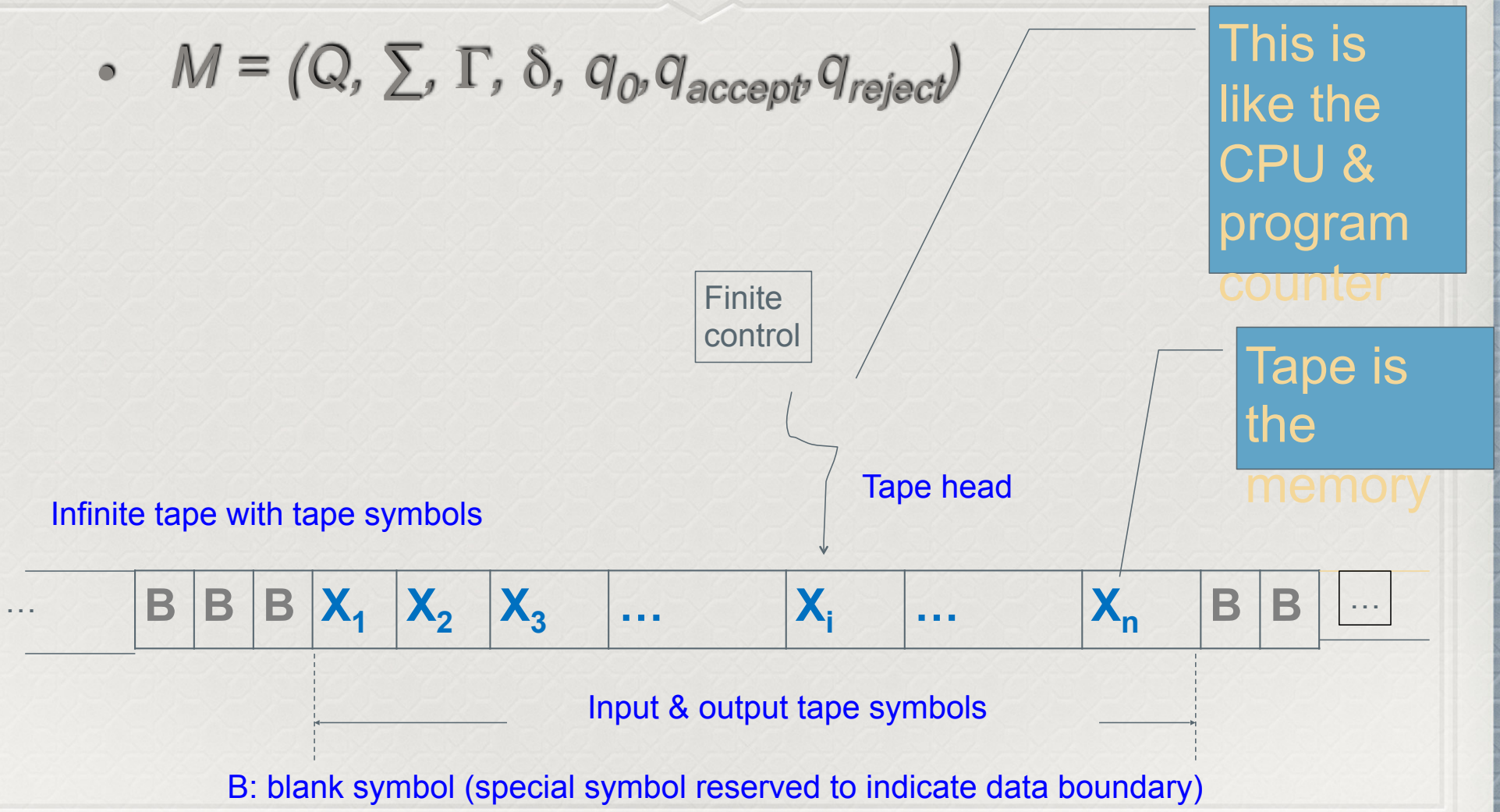
Definition

A ***Turing machine*** is a 7-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, where Q, Σ, Γ are all finite sets and

1. Q is the set of states,
2. Σ is the input alphabet not containing the ***blank symbol*** $\sqcup$,
3. Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$,
4. $\delta: Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$ is the transition function,
5. $q_0 \in Q$ is the start state,
6. $q_{accept} \in Q$ is the accept state, and
7. $q_{reject} \in Q$ is the reject state, where $q_{reject} \neq q_{accept}$.

A Turing Machine (TM)

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$



Transition function

You can also use:
➔ for R
🡐 for L

- *One move (denoted by |---) in a TM does the following:*
  - *δ(q,X) = (p,Y,D)*
    - *q is the current state*
    - *X is the current tape symbol pointed by tape head*
    - *State changes from q to p*
    - *<u>After the move:</u>*
      - *X is replaced with symbol Y*
      - *If D="L", the tape head moves "left" by one position.*
        Alternatively, if D="R" the tape head moves "right" by one position.

ID of a TM

- *Instantaneous Description or ID (Configurations):*
  - $X_1X_2\ldots X_{i-1}qX_iX_{i+1}\ldots X_n$ *means:*
    - q *is the current state*
    - *Tape head is pointing to* X_i
    - $X_1X_2\ldots X_{i-1}X_iX_{i+1}\ldots X_n$ *are the current tape symbols*

- $\delta(q,X_i) = (p,Y,R)$ *is same as:*

  $X_1\ldots X_{i-1}qX_i\ldots X_n \vdash X_1\ldots X_{i-1}YpX_{i+1}\ldots X_n$

- $\delta(q,X_i) = (p,Y,L)$ *is same as:*

  $X_1\ldots X_{i-1}qX_i\ldots X_n \vdash X_1\ldots pX_{i-1}YX_{i+1}\ldots X_n$

Definitions

- *Call a language **Turing-recognizable** if some Turing machine recognizes it.*
  - ➢ *The machine may accept, reject, or loop.*
- *Call a language **Turing-decidable** or simply **decidable** if some Turing machine decides it.*
  - ➢ *The machine may accept, reject, but it never loops.*

Way to check for Membership

- *Is a string w accepted by a TM?*
- *Initial condition:*
  - *The (whole) input string w is present in TM, preceded and followed by infinite blank symbols*
- *Final acceptance:*
  - *Accept w if TM enters final state and halts*
  - *If TM halts and not final state, then reject*

Example 1: $A = \{0^{2^n} \mid n \geq 0\}$

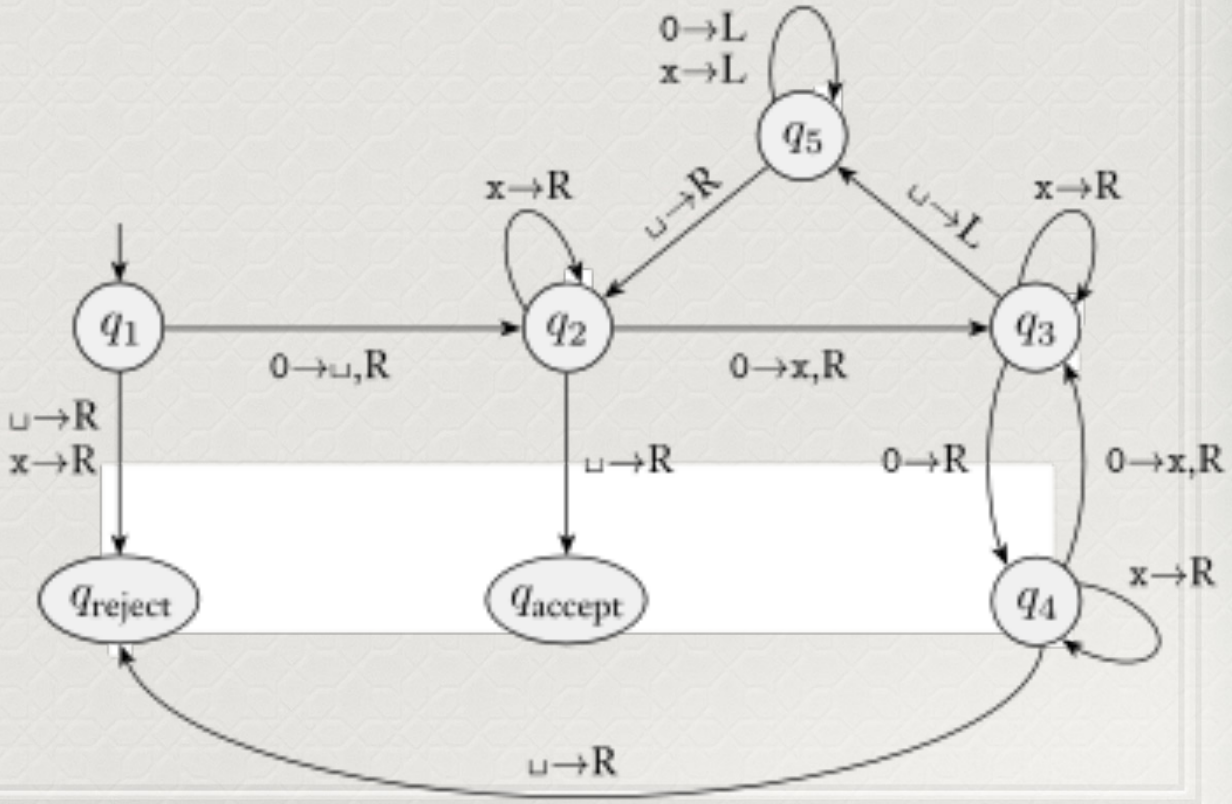
- *M_2 that decides $A = \{0^{2^n} \mid n \geq 0\}$ "On input string w:*

1. ***Sweep left to right across the tape, crossing off every other 0.***
2. ***If in stage 1 the tape contained a single 0, accept .***
3. ***If in stage 1 the tape contained more than a single 0 and the number of 0s was odd, reject .***
4. ***Return the head to the left-hand end of the tape.***
5. ***Go to stage 1."***

Example 1 Solution

- $Q=\{q_1, q_2, q_3, q_4, q_5, q_{accept}, q_{reject}\}$,
- $\Sigma=\{0\}$,
- $\Gamma=\{0,x,\sqcup\}$.

The start, accept, and reject states are q_1, q_{accept}, and q_{reject}, respectively.



Example 1 Sample Run

- *Input w: 0000*
- *The sequence of configurations the machine enters are as follows:*

$q_1 0000$
$\sqcup q_2 000$
$\sqcup x q_3 00$
$\sqcup x 0 q_4 0$
$\sqcup x 0 x q_3 \sqcup$
$\sqcup x 0 q_5 x \sqcup$
$\sqcup x q_5 0 x \sqcup$
$\sqcup q_5 x 0 x \sqcup$
$q_5 \sqcup x 0 x \sqcup$
$\sqcup q_2 x 0 x \sqcup$
$\sqcup x q_2 0 x \sqcup$
$\sqcup x x q_3 x \sqcup$
$\sqcup x x x q_3 \sqcup$
$\sqcup x x q_5 x \sqcup$
$\sqcup x q_5 x x \sqcup$
$\sqcup q_5 x x x \sqcup$
$q_5 \sqcup x x x \sqcup$
$\sqcup q_2 x x x \sqcup$
$\sqcup x q_2 x x \sqcup$
$\sqcup x x q_2 x \sqcup$
$\sqcup x x x q_2 \sqcup$
$\sqcup x x x \sqcup q_{accept}$

Example 1 Sample Run

- *Input w: 000000*
- *The sequence of configurations the machine enters are as follows:*

$q_1000000 \vdash \sqcup q_200000 \vdash \sqcup xq_30000 \vdash$

$\sqcup x0q_4000 \vdash \sqcup x0xq_300 \vdash \sqcup x0x0q_40 \vdash$

$\sqcup x0x0xq_3\sqcup \vdash \sqcup x0x0q_3x\sqcup \vdash \sqcup x0xq_30x\sqcup \vdash$

$\sqcup x0q_3x0x\sqcup \vdash \sqcup xq_30x0x\sqcup \vdash \sqcup q_3x0x0x\sqcup \vdash$

$q_3\sqcup x0x0x\sqcup \vdash \sqcup q_2x0x0x\sqcup \vdash \sqcup xq_20x0x\sqcup \vdash$

$\sqcup xxq_3x0x\sqcup \vdash \sqcup xxxq_30x\sqcup \vdash \sqcup xxx0q_4x\sqcup \vdash$

$\sqcup xxx0xq_4\sqcup \vdash \sqcup xxx0x\sqcup q_{reject}\sqcup$

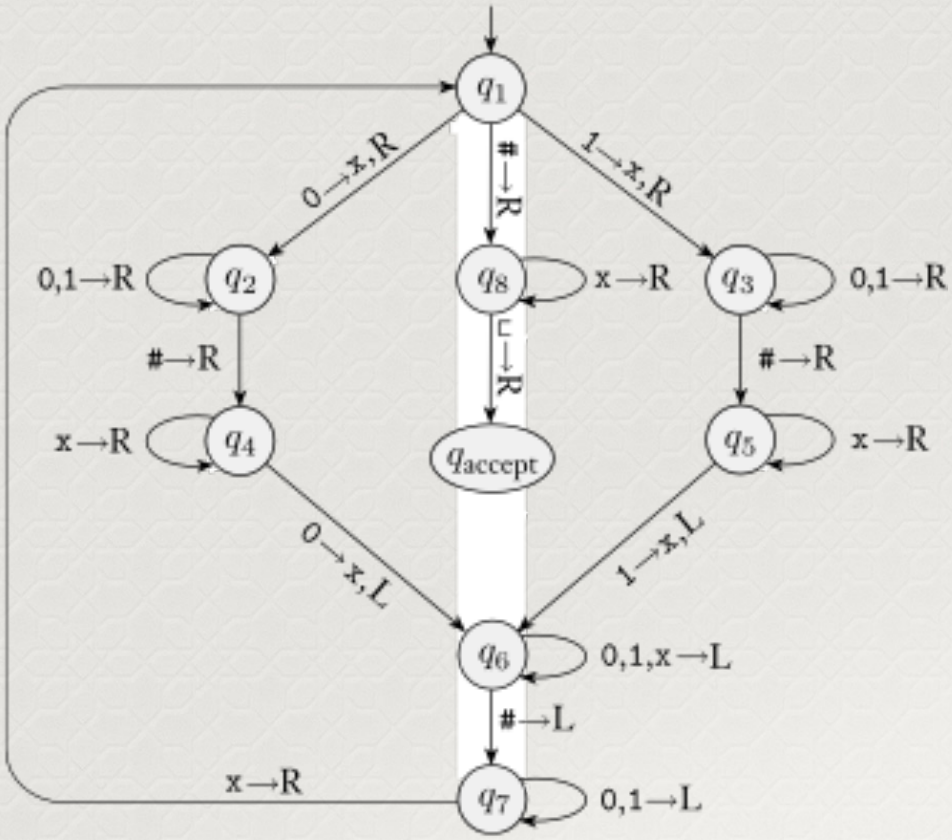
Example 2: B = {w#w| w ∈ {0,1}∗}

- *M_1 that decides B = {w#w| w ∈ {0,1}∗} on input string w:*
  - Zig-zag across the tape to corresponding positions on either side of the # symbol to check whether these positions contain the same symbol. If they do not, or if no # is found, reject. Cross off symbols as they are checked to keep track of which symbols correspond.
  - When all symbols to the left of the # have been crossed off, check for any remaining symbols to the right of the #. If any symbols remain, reject; otherwise, accept."

Example 2 Solution

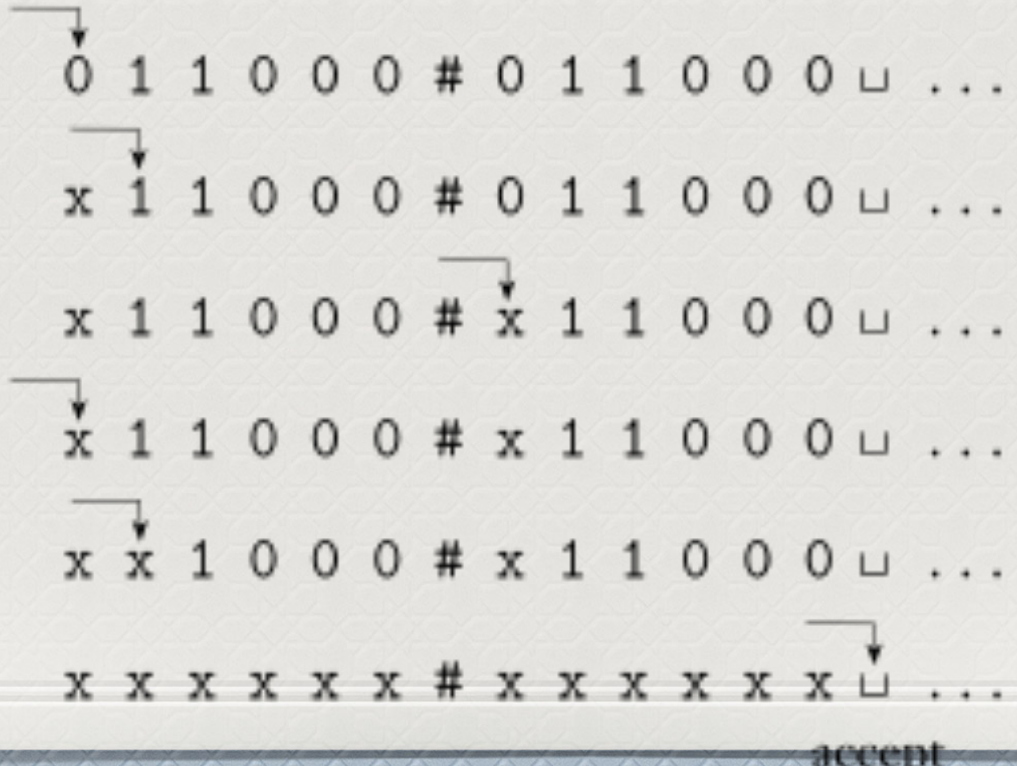
- $Q=\{q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_{accept}, q_{reject}\}$,
- $\Sigma=\{0, 1, \#\}$,
- $\Gamma=\{0,1, \#, x, \sqcup\}$.

The start, accept, and reject states are q_1, q_{accept}, and q_{reject}, respectively.



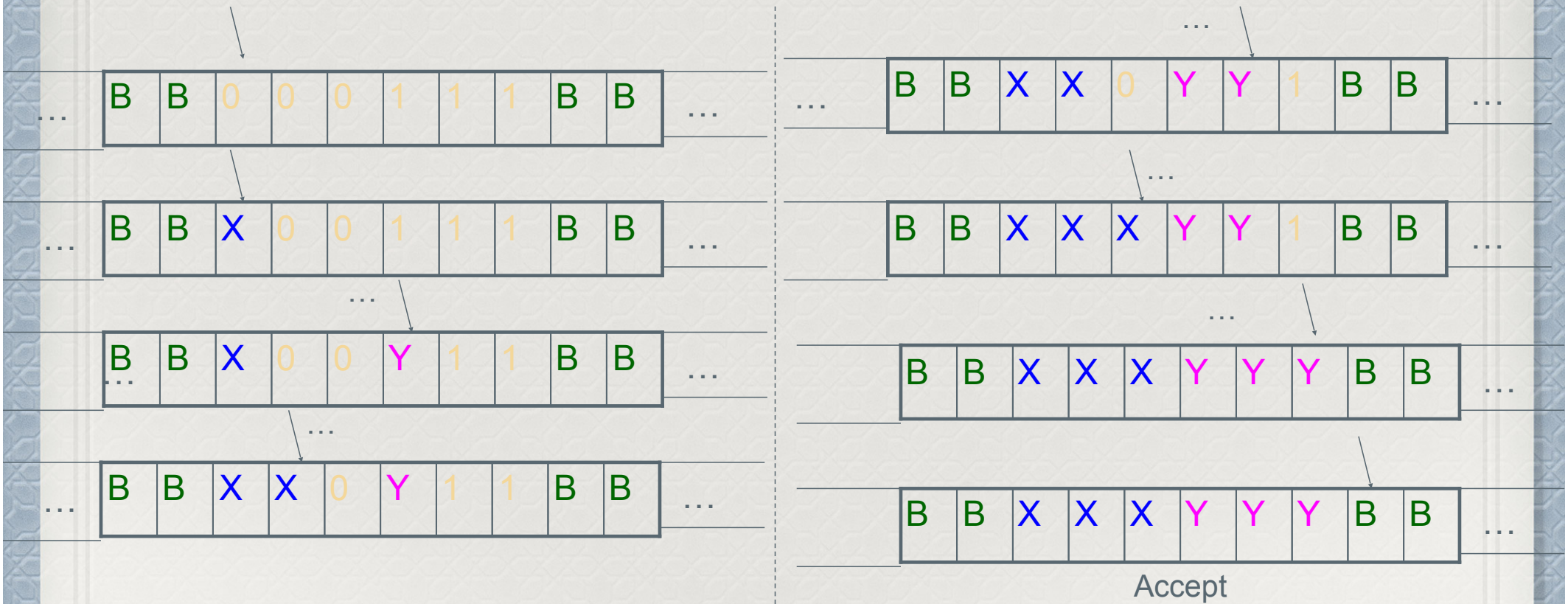
Example 2 Sample Run

- *Input w: 011000#011000*
- *The sequence of configurations the machine enters are as follows:*

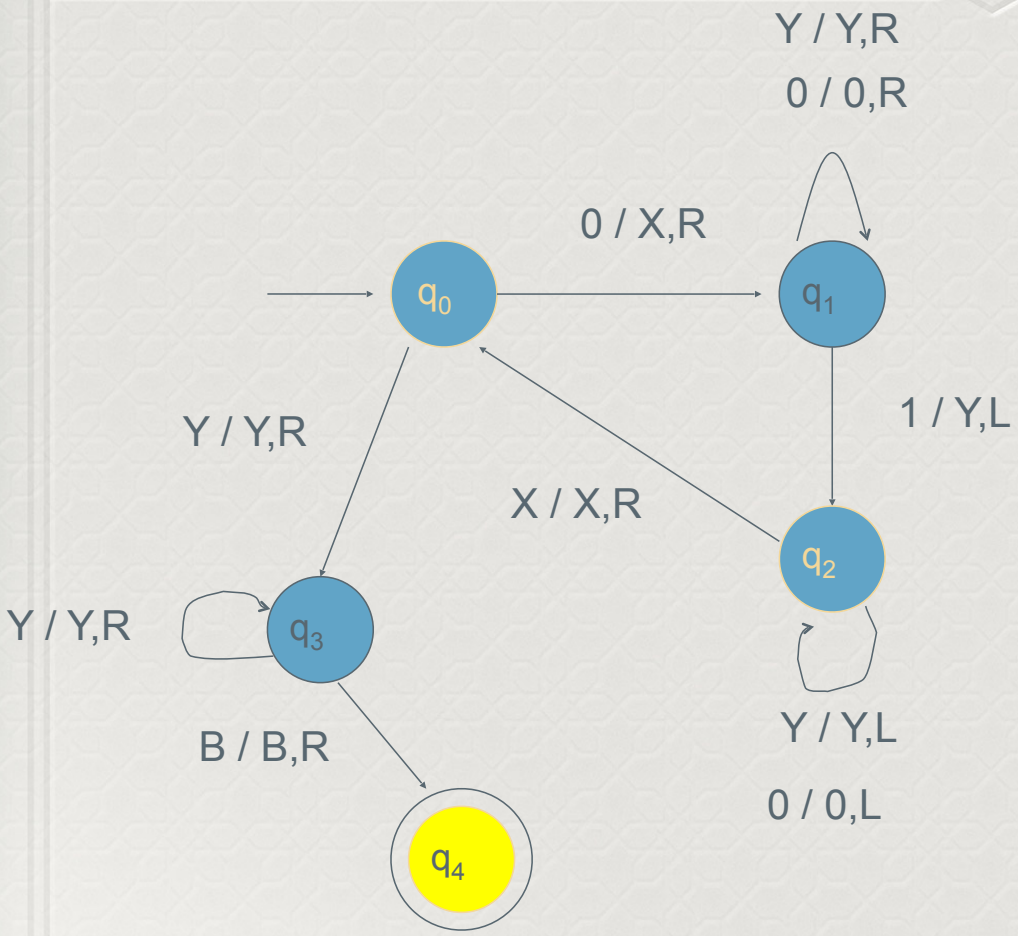


Example 3: L = {0^n1^n | n≥1}

- *Strategy:* *w = 000111*



Example 3 Solution



1. *Mark next unread 0 with X and move right*
2. *Move to the right all the way to the first unread 1, and mark it with Y*
3. *Move back (to the left) all the way to the last marked X, and then move one position to the right*
4. *If the next position is 0, then goto step 1.*
   Else move all the way to the right to ensure there are no excess 1s. If not move right to the next blank symbol and stop & accept.

*state diagram representation preferred

Ex 3: TM for $\{0^n1^n \mid n \geq 1\}$

	Next Tape Symbol				
Curr. State	0	1	X	Y	B
q_0	(q_1,X,R)	-	-	(q_3,Y,R)	-
q_1	$(q_1,0,R)$	(q_2,Y,L)	-	(q_1,Y,R)	-
q_2	$(q_2,0,L)$	-	(q_0,X,R)	(q_2,Y,L)	-
q_3	-	-	-	(q_3,Y,R)	(q_4,B,R)
$*q_4$	-	--	-	-	-

Table representation of the state diagram

TMs for calculations

- *TMs can also be used for calculating values*
  - *Like arithmetic computations*
  - *Eg., addition, subtraction, multiplication, etc.*

Example 4: Proper Subtraction

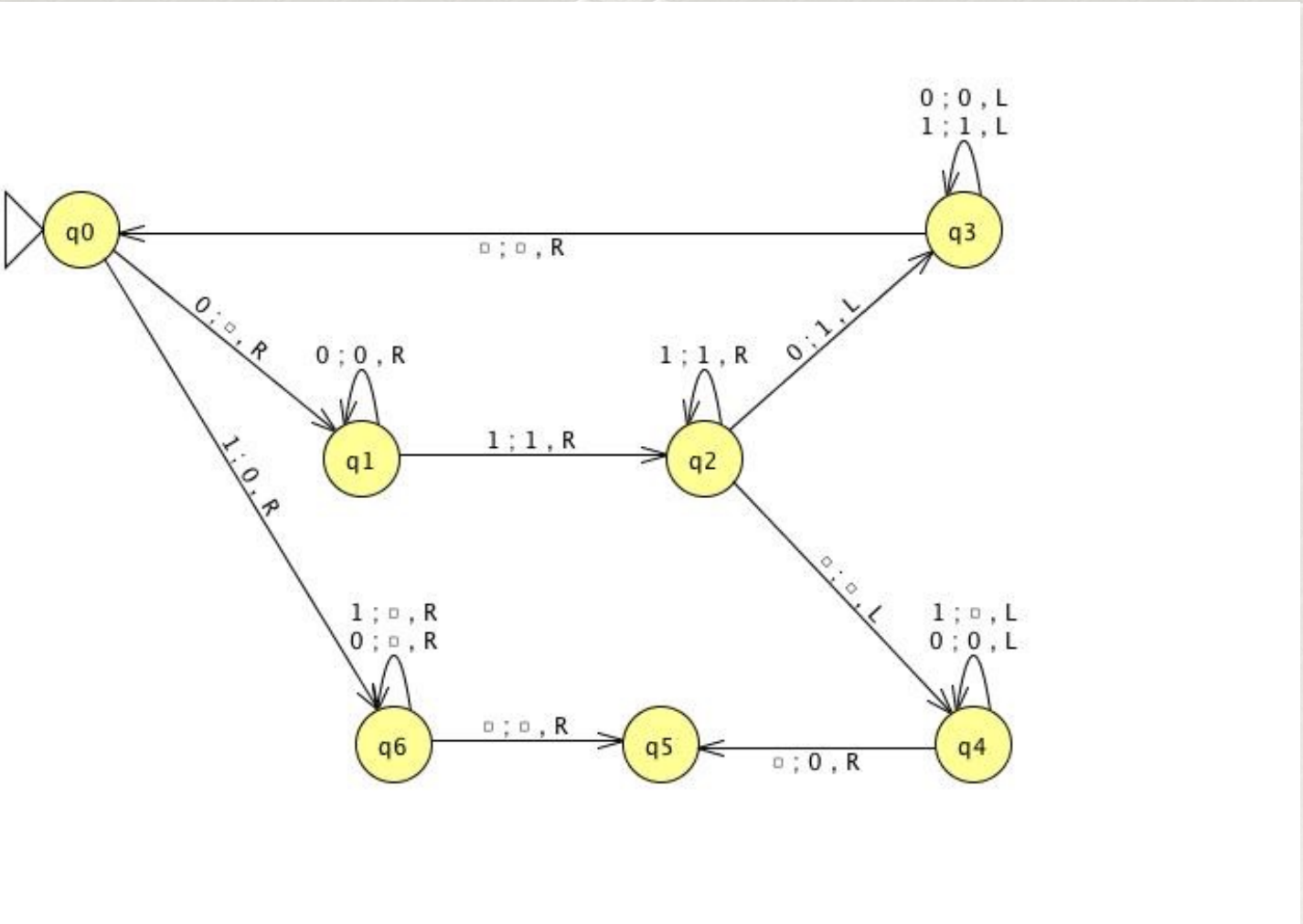
"m -- n" = max{m-n,0}

$0^m10^n \rightarrow$ ...B 0^{m-n} B.. (if m>n)
...BB…B.. (otherwise)

1. For every 0 on the left (mark B), mark off a 0 on the right (mark 1)
2. Repeat process, until one of the following happens:
    1. // No more 0s remaining on the left of 1
       Answer is 0, so flip all excess 0s on the right of 1 to Bs (and the 1 itself) and halt
    2. //No more 0s remaining on the right of 1
       Answer is m-n, so simply halt after making 1 to B

Give state diagram

Example 4: Solution



Example 4: Solution

- $\sum = \{ 0, 1 \}$, $Q = \{q0, q1, q2, q3, q4, q6\}$, $q_{accept} = \{q5\}$
- transition function =
- $\delta(q0, 0) = (q1, B, R)$ $\delta(q0, 1) = (q6, B, R)$
- $\delta(q1, 0) = (q1, 0, R)$ $\delta(q1, 1) = (q2, 1, R)$
- $\delta(q2, 0) = (q3, 1, L)$ $\delta(q2, 1) = (q2, 1, R)$
- $\delta(q2, B) = (q4, B, L)$ $\delta(q3, 0) = (q3, 0, L)$
- $\delta(q3, 1) = (q3, 1, L)$ $\delta(q3, B) = (q0, B, R)$
- $\delta(q4, 0) = (q4, 0, L)$ $\delta(q4, 1) = (q4, B, L)$
- $\delta(q4, B) = (q5, 0, R)$ $\delta(q6, 0) = (q6, B, R)$
- $\delta(q6, 1) = (q6, B, R)$ $\delta(q6, B) = (q5, B, R)$.

Exercise 5: Multiplication

- 0^m10^n1 (input), $0^{mn}1$ (output)
- *Pseudocode:*
  1. Move tape head back & forth such that for every 0 seen in 0^m, write n 0s to the right of the last delimiting 1
  2. Once written, that zero is changed to B to get marked as finished
  3. After completing on all m 0s, make the remaining n 0s and 1s also as Bs

Give state diagram

http://turingmaschine.klickagent.ch/einband/?lang=en#2_*_2

Calculations vs. Languages

A "calculation" is one that takes an input and outputs a value (or values)

The "language" for a certain calculation is the set of strings of the form "<input, output>", where the output corresponds to a valid calculated value for the input

A "language" is a set of strings that meet certain criteria

E.g., The language L_{add} for the addition operation

"<0#0,0>"
"<0#1,1>"
…
"<2#4,6>"
…

Membership question == verifying a solution
e.g., is "<15#12,27>" a member of L_{add} ?