CISC 4090
Theory of Computation

Turing machines

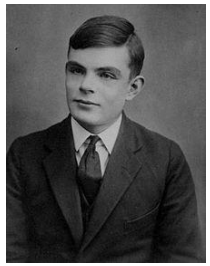
Professor Daniel Leeds
dleeds@fordham.edu
JMH 332

Alan Turing (1912-1954)

Father of Theoretical Computer Science

Key figure in Artificial Intelligence

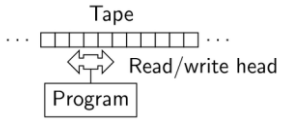
Codebreaker for Britain in World War I



Turing machine

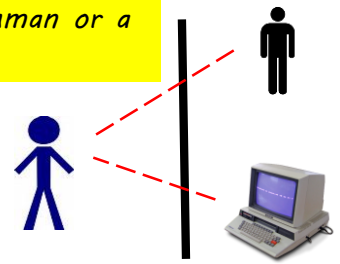
Simple theoretical machine

Can do anything a real computer can do!



Detour: "Turing test"

A computer is "intelligent" if human investigator can't tell if she's talking to a human or a computer

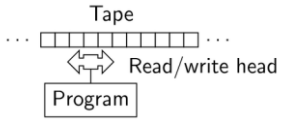


http://xkcd.com/329/

Turing machine

Simple theoretical machine

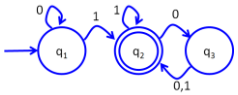
Can do anything a real computer can do!



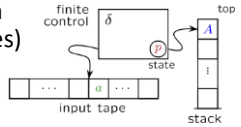
6

Review of machines

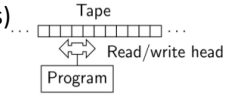
- Finite state automaton (Regular languages)



- Push down automaton (Context free languages)



- Turing machine (beyond CFLs)



7

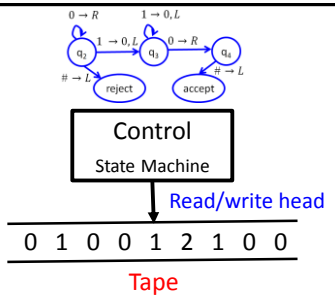
Turing machine structure

Infinite tape

At each step
- Must move left/right on tape
- Can change state
- Can change tape content

When reaches accept or reject state, terminates and outputs "accept" or "reject"

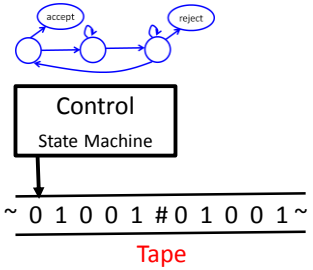
Can loop forever



8

A Turing Machine for B={w#w|$w \in \{0,1\}^*$}

Assume the string is written on the tape and you start at the beginning of the string. What can we do?



10

Strategy:

Find left-most 0-or-1 character in first word

If match left-most character in second word, X out both chars

Else reject

If no characters left, accept

~ ~ ~ 0 1 1 0 0 0 # 0 1 1 0 0 0 ~ ~ ~
~ ~ ~ X 1 1 0 0 0 # X 1 1 0 0 0 ~ ~ ~
~ ~ ~ X X 1 0 0 0 # X X 1 0 0 0 ~ ~ ~

How do we move this with single actions: move-by-one and write?

11

Turing machine: the formal definition

7 tuple: $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

Q is set of states

Σ is input alphabet

Γ is the tape alphabet; blank∈ Γ and Σ ∈ Γ

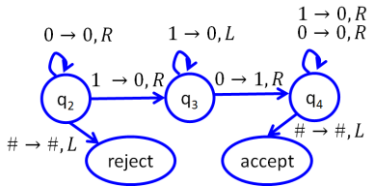
$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ transition function

Start, accept, and reject state: q_0, q_{accept}, q_{reject}

13

The transition function

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



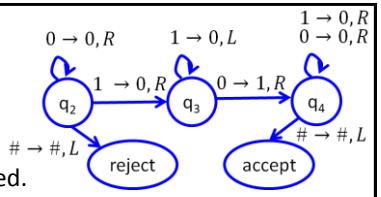
Given state q and symbol a at present location on tape,

change to state r, change symbol on tape to b, move Left or Right

Change in: (state, tape content, head location)
– called "configuration"

14

The transition function



Example:

Start at q_2. Current position underlined.

Step 0: q_2 ~ 0 0 1 1 # ~ ~ 0 0 1 1 # ~

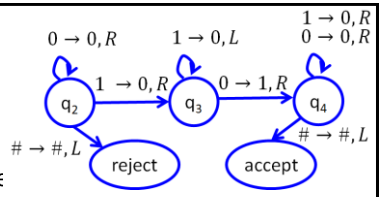
Step 1: q_3 ~ 0 0 0 1 # ~ ~ 0 0 0 1 # ~

Step 2: q_4 ~ 0 1 0 1 # ~

Step 3: q_4 ~ 0 1 0 1 # ~

16

The transition function



Example:

Start at q_2. Current position underline

Step 0: q_2 ~ 0 0 # 1 # ~ ~ 0 0 # 1 # ~

Step 1: q_2 ~ 0 0 # 1 # ~ ~ 0 0 # 1 # ~

Step 2: q_2 ~ 0 1 # 1 # ~

Step 3: reject ~ 0 1 # 1 # ~

18

Strategy: B={w#w|$w \in \{0,1\}^*$}

Find left-most 0-or-1 character in first word

If match left-most character in second word, X out both chars

Else reject

If no characters left, accept

~ ~ ~ 0 1 1 0 0 0 # 0 1 1 0 0 0 ~ ~ ~
~ ~ ~ X 1 1 0 0 0 # X 1 1 0 0 0 ~ ~ ~
~ ~ ~ X X 1 0 0 0 # X X 1 0 0 0 ~ ~ ~

19

Strategy: B={w#w|$w \in \{0,1\}^*$}

Define TM state sequence for each big-picture algorithmic step

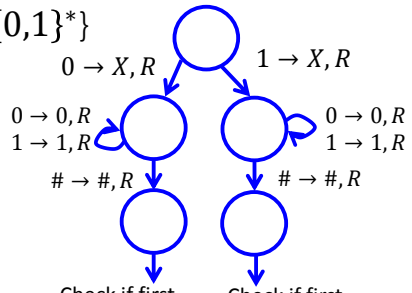
Given character s in left word

1. Move to right word
2. Check if first available symbol in right word == s
3. If match, keep going; else reject

~ ~ ~ 0 1 1 0 0 0 # 0 1 1 0 0 0 ~ ~ ~
~ ~ ~ X 1 1 0 0 0 # X 1 1 0 0 0 ~ ~ ~
~ ~ ~ X X 1 0 0 0 # X X 1 0 0 0 ~ ~ ~

20

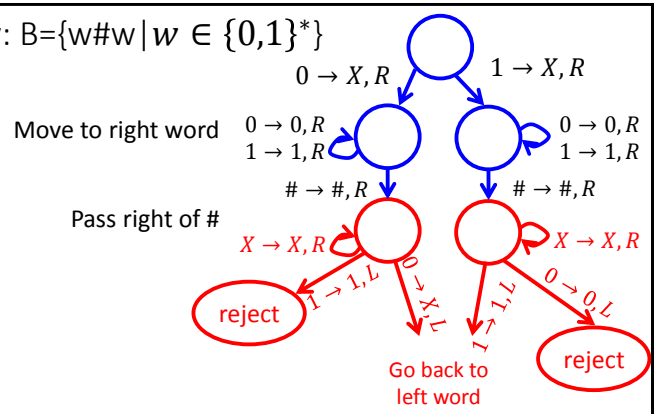
Strategy: B={w#w|$w \in \{0,1\}^*$}



Given character s in left word

1. Move to right word
2. Check if first available symbol in right word == s
3. If match, keep going; else reject

Strategy: B={w#w|$w \in \{0,1\}^*$}



Strategy: B={w#w|$w \in \{0,1\}^*$}

Typical big-picture solution

Find left-most 0-or-1 character in first word

If match left-most character in second word, X out both chars

Else reject

If no characters left, accept

~ ~ ~ 0 1 1 0 0 0 # 0 1 1 0 0 0 ~ ~ ~
~ ~ ~ X 1 1 0 0 0 # X 1 1 0 0 0 ~ ~ ~
~ ~ ~ X X 1 0 0 0 # X X 1 0 0 0 ~ ~ ~

Strategy: B={w#w|$w \in \{0,1\}^*$}

Find left-most 0-or-1 character in first word

If match left-most character in second word, X out both chars

Else reject

If no characters left, accept

~ ~ ~ 0 1 1 0 0 0 # 0 1 1 0 0 0 ~ ~ ~
~ ~ ~ X 1 1 0 0 0 # X 1 1 0 0 0 ~ ~ ~
~ ~ ~ X X 1 0 0 0 # X X 1 0 0 0 ~ ~ ~

Analysis: We will always get an answer (accept or reject), because problem gets smaller after each step