CS 3313
Foundations of Computing:

Turing Machine – Part 1

http://gw-cs3313.github.io

1

Next..

- Turing Machine model
  - TM as an automaton - today
  - TM as transducer – to compute functions … Thursday
- Changing the basic TM model…..next week
  - Multiple tracks, multiple tapes, two-way tape/storage
  - Non-deterministic TM
  - Equivalence of Deterministic and Non-deterministic TMs
    - Simulation procedure
  - Simulation of RAM (Random Access Machine) on a TM
- Solvable and Unsolvable problems… week 13
- Time and space complexity
- Other models of computation: λ-calculus (functional programming)

2

Moving on from PDAs….Turing Machines

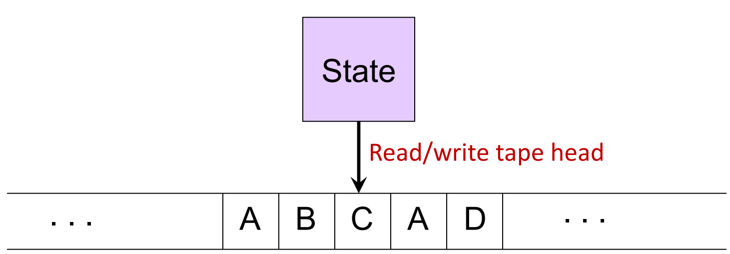
- Finite State Automata (DFA/FSM): finite number of states
  - States store summary of past input/events
  - No external storage …so cannot have a counter to store variable
- Pushdown Automata (PDA): Add a “external” stack storage to a NFA
  - Single stack – first in-last out
  - What is stored in stack comes out in reverse when it is popped
- Extend PDA…..Two stacks, two-way input tape, etc…..OR
- Generalize the storage form to random access
  - Can store into any location and read from any location
  - Instead of a “box” as storage, we move to a line of bookshelves
- Turing Machine: NFA + external storage on a tape

3

Turing Machine

Action: based on the (i) state and (ii) the tape symbol under the read/write head:

- (1) change state, (2) write a symbol back to the tape and (3) move the head (left or right) one location/cell on the tape.



Infinite tape with cells containing tape symbols chosen from a finite alphabet

4

4

Turing-Machine Formalism

- A TM is described by:
  1. A finite set of *states* Q.
  2. An *input alphabet* Σ.
  3. A *tape alphabet* Γ (contains Σ).
  4. A *transition function* $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$
  5. A *start state* q_0 (in Q).
  6. A *blank symbol* B (or □) in Γ- Σ
     – All tape except for the input is blank initially.
  7. A set of *final states* $F \subseteq Q$

Tape is an infinite (both left and right) number of tape cells

5

5

The Transition Function

- Takes two arguments:
  1. A state, in Q.
  2. A tape symbol in Γ.
- *δ(q, Z)* is either undefined or a triple of the form *(p, Y, D)*.
  - *p* is a state.
  - *Y* is the new tape symbol.
  - *D* is a *direction*, L or R – move the tape head to the Left or Right
- Convention: *If δ(q, Z) undefined then TM halts*
  - If it halts in a final state then it accepts
  - If it halts in a non-final state then it rejects

6

6

Functioning of TM

- "instruction set" = In one step/time:
  - TM reads one tape symbol on tape (cell)
  - Writes symbol back to cell on tape
  - Changes state
  - Moves tape head Left or Right
- goes through sequence of steps controlled by transition function
- Whole process *may terminate* – TM gets to a *halting state*
  - Halts when it reaches a configuration where no transition is defined
  - The state it halts in can be a final state or a non-final state
  - *Assume no transitions are defined in the final state*

7

Example 1: Turing Machine

- This TM scans its input right, looking for a 1
- If it finds one, it changes to 0, goes to final state and halts
- States = {q (start), f (final)}.
- Input symbols = {0, 1}.
- Tape symbols = {0, 1, B}.
- $\delta(q, 0) = (q, 0, R)$.
- $\delta(q, 1) = (f, 0, R)$.

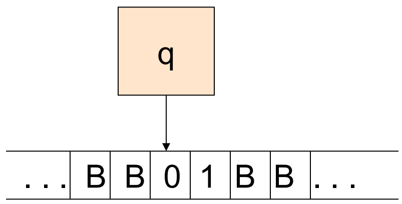
8

8

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



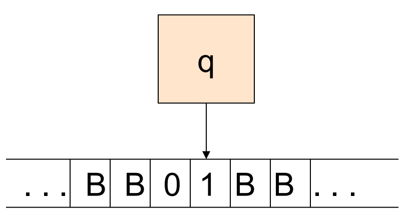
9

9

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



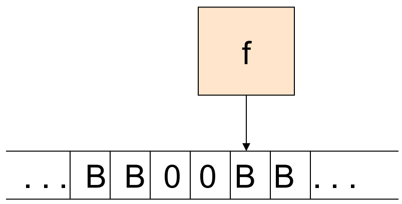
10

10

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



No move is possible. The TM halts and accepts.

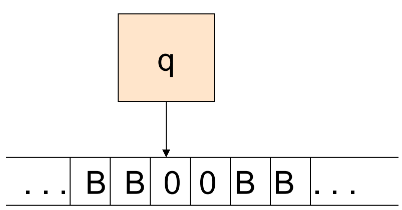
11

11

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



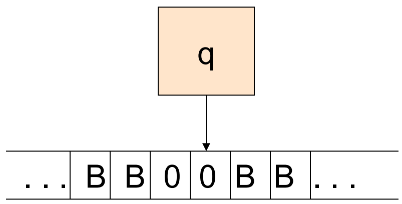
12

12

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



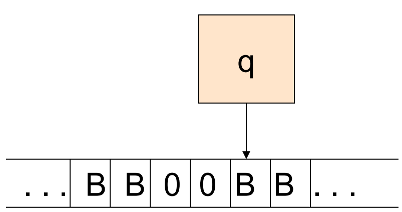
13

13

Simulation of TM

$\delta(q, 0) = (q, 0, R)$

$\delta(q, 1) = (f, 0, R)$



No move is possible.
The TM halts and
rejects.

14

14

Example 2: Turing Machine

- States = {q (start), f (final)}. Input symbols = {0, 1}.
- Tape symbols = {0, 1, B}.
- $\delta(q, 0) = (q, 0, R)$ $\delta(q, 1) = (f, 0, R)$ $\delta(q, B) = (q, B, L)$.
- What happens in this TM for input $w = 01$?
- What happens in this TM for input $w = 00$?

15

15

Snapshot of the system – Instantaneous Description (ID) of a Turing Machine

- Initially, a TM has a tape consisting of a string of input symbols surrounded by an infinity of blanks in both directions.
- The TM is in the start state, and the head is at the leftmost input symbol.
- At any instant in time, how do we specify the "system snapshot" ?
  - Analogous to ID in PDA – we want to capture the entire state of the system (tape contents, state)

16

16

Snapshot of the system – Instantaneous Description (ID) of a Turing Machine

- At any instant in time, how do we specify the "system snapshot" ?

17

17

Snapshot of the system – Instantaneous Description (ID) of a Turing Machine

- An ID is a string $\alpha q \beta$, where $\alpha\beta$ includes the tape between the leftmost and rightmost nonblanks.
  - The state q is immediately to the left of the tape symbol scanned
  - If q is at the right end, it is scanning B.
    - If q is scanning a B at the left end, then consecutive B's at and to the right of q are part of α.

18

18

TM ID's – (2)

- As for PDA's we may use symbols ⊦ and ⊦* to represent "becomes in one move" and "becomes in zero or more moves," respectively, on ID's.
- Example: The moves of the Example 2 TM are
- $\delta(q, 0) = (q, 0, R)$ $\delta(q, 1) = (f, 0, R)$ $\delta(q, B) = (q, B, L)$.

$$q01 \vdash 0q1 \vdash 00f$$

19

19

Formal Definition of Moves: Instantaneous Description (ID)

- At any instant in time, the TM is in a state q, its tape head is reading some symbol Z, the string α is to the left of the tape head, and the string β is to the right of the tape head:

  this ID is denoted as $\alpha q\, Z\beta$
- If $\delta(q, Z) = (p, Y, R)$, then $\alpha q\, Z\beta \vdash \alpha Y p\beta$
  - If Z is the blank B, then also $\alpha q \vdash \alpha Yp$
- If $\delta(q, Z) = (p, Y, L)$, then for any X, $\alpha Xq\, Z\beta \vdash \alpha p\, XY\beta$
  - In addition, $qZ\beta \vdash pBY\beta$

20

20

Languages of a TM

- A TM defines a language by final state, as usual.
- $L(M) = \{w \mid q_0 w \vdash^* I$, where I is an ID with a final state$\}$.
  - TM halts in this configuration
  - Alternate definition: accepts as long as state is a final state

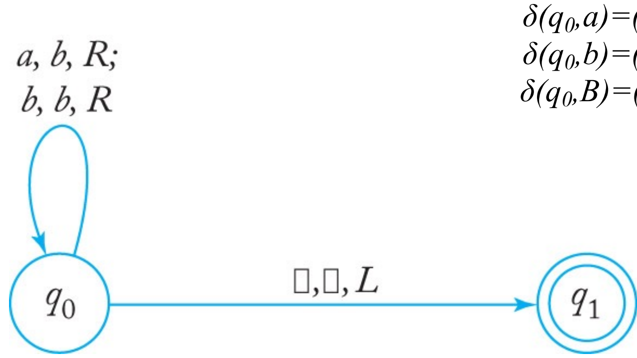
21

21

Transition Graphs for Turing Machines

- In JFLAP, you will see a transition graph for a TM
- In a Turing machine transition graph, each edge is labeled with three items:
  1. current tape symbol,
  2. new tape symbol, and
  3. direction of the head move

$\delta(q_0,a)=(q_0,b,R)$
$\delta(q_0,b)=(q_0,b,R)$
$\delta(q_0,B)=(q_1,B,L)$



22

Recursively Enumerable Languages

- Consider the class of languages L, where if w is in L then there is a TM that halts on input w
  - Does not say what happens when w is not in the language
    - The TM may never halt
- This class of languages is called the *recursively enumerable languages*.
  - Why? The term actually predates the Turing machine and refers to another notion of computation of functions.

23

23

Recursive Languages

- An *algorithm* is a TM, accepting by final state, that is *guaranteed to halt* whether or not it accepts.
- If L = L(M) for some TM M that is an algorithm, we say L is a *recursive language*.
  - Why? Again, don't ask; it is a term with a history.

24

24

Turing Machine Design: Examples

- A TM function can be captured by describing its behavior by an "algorithm"
  - First describe how TM works
  - Then capture how states can be encoded to capture specific actions/steps
  - Finally, formally specify the transition function

Example 1: L ={ $a^n b^n$ | n > 0 }

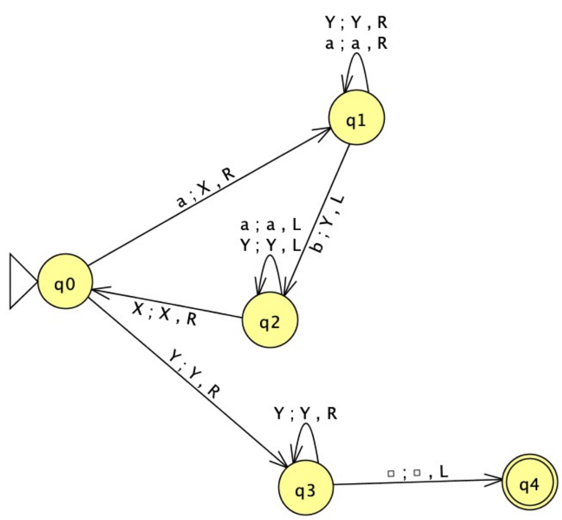
- Algorithm:
  - Each step: read symbol from tape, write symbol to tape, go to state, move left/right

1. Read *a*, write *X* (mark the *a*), move right to find a "matching" *b:* for each *a* there must be a matching *b*
   - Read a, write X , move right goto 2
2. Skip over a's (move right) until you read a *b:* mark *b* with a *Y* and then sweep left until you find the leftmost unmarked *a*
   - Read *a*, write *a*, move right -- skip right over *a's*
   - read *b*, write Y (mark the *b*), move left go to 3
   - Read Y, write Y, move right – skip right over Y's
3. Skip left until you find the leftmost a – until you read X (rightmost marked a)

Example 1: L ={ a^n b^n | n > 0 }

1. Read *a*, write *X* (mark the *a*), move right to find a "matching" *b:*
   - Read *a*, write X , move right goto 2
   - Read Y, write Y, move right, goto 4 /*now check no *b*'s unmarked */
2. Skip over a's (move right) until you read a *b:*
   - Read *a*, write *a*, move right -- skip right over *a's*
   - read *b*, write Y (mark the *b*), move left go to 3
   - Read Y, write Y, move right – skip right over Y's
3. Skip left until you find the leftmost a (until you read X)
   - Read Y/a, write Y/a, move left
   - Read X, write X, move right goto 1
4. Check no more *b*'s remaining
   - Read Y, Write Y, move right
   - Read Blank, Write Blank, move left goto Final state/accept.

Example 1: $L =\{ a^n\ b^n \mid n > 0 \}$



Exercise: $L = \{ a^n b^n c^n \mid n > 1 \}$

- Describe algorithm for a TM that accepts L

29

Example 2: $L = \{ wcw \mid w \text{ in } \{a.b\}^* \}$

- Recall this was not a CFL…
- Design a TM to accept this language.
- Algorithm Outline:
  - For each symbol in first *w*, check if corresponding 'location' in second *w* has the same symbol
    - The second *w* starts to the right of midpoint symbol *c*
  - If we read *a*, then check if symbol in corresponding location of second *w* is also an *a*
    - How do we know where second *w* = after we read input *c*
  - If we read *b*, then check if symbol in corresponding location of second *w* is also an *b*
  - If all symbols match then accept else reject

30

TM Design Strategy: Storage in state

- In example *{wcw}* we read the symbol in first substring w and need to "remember" it (i.e, "store" it) so we can check if matching location in second w has the same symbol….Where/how do we store this information ?
- Recall – a state can store/summarize finite amount of information…..
  - We have read an "a"
  - We have read a "b"
  - ….etc.
- *Think of the state as having two components [q, X]*
  - State q, and symbol X

31

Example 1: $L_2 = \{ wcw \mid w \text{ in } \{a,b\}^* \}$

- Algorithm:
  1. Read symbol (*a* or *b*), mark it as checked (with an X) and "store" it in the state
     - If the symbol read is a *c* then check if no more unmarked symbols, goto 5
  2. Move right until you hit a c, then move to second substring w.
  3. Check if first unmarked symbol on tape = symbol stored in state, mark with Y, move left, go to 4
  4. Go to the leftmost unmarked symbol on tape/input – skip over all symbols until you hit an X, move one tape cell to the right, and goto 1.
  5. Skip right over all marked symbols Y, until you read Blank - accept

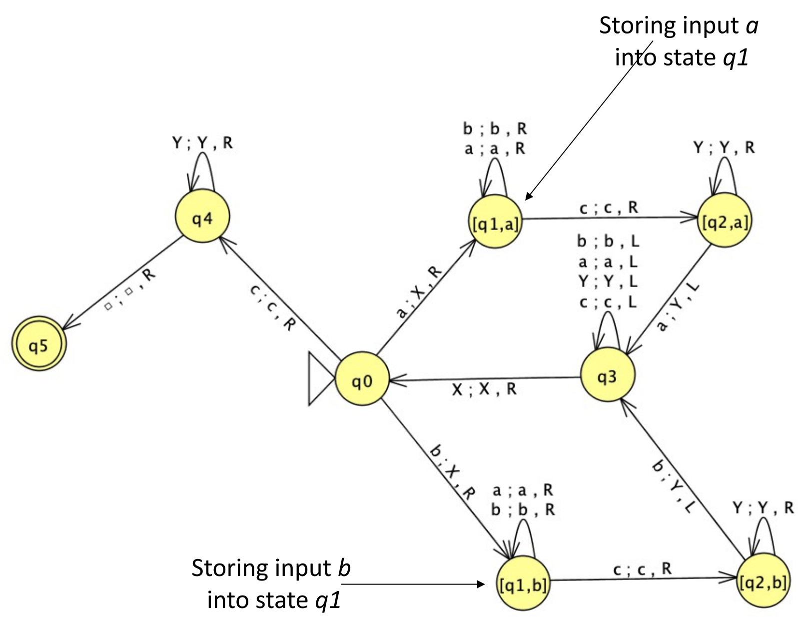
32

Example 2: L_2 = { wcw | w in {a,b}* }

- Algorithm:
  1. Read tape symbol (*a* or *b*), mark it as checked (with an X) and "store" it in the state goto 2
     - If the symbol read is a *c* then goto 5 (& check no more unchecked)
  2. Find leftmost unchecked symbol in second w: Move right until you read c, then move right goto 3.
  3. *Skip right* over checked symbols until you read unchecked symbol: Check if symbol on tape = symbol stored in state, mark with X, move left, go to 4
     - Else halt and reject
  4. Go to the leftmost unmarked input – skip over all symbols until you hit an X, move one tape cell to the right, and goto 1
  5. Check if no more symbols left in second substring w: Skip right over X until you read B then goto final state
     - Else reject

33

Example 2: L_2 = { wcw | w in {a,b}* }



34

Next….

- Computing functions using Turing machines
  - Input= x (integer), Output = $f(x)$
- Labs- more examples and using JFLAP
- Next week – adding “features” to standard turing machine
  - Multiple tracks on tape
  - Multiple tapes
  - Non-determinism

35

Example 3: $L = \{ w \ w^R \mid w \text{ in } \{a,b\}^* \}$

- We know this can be accepted by a PDA…so use this example to set things up.
- Input on the tape: string x followed by Blanks (B)
  - Tape head is at the leftmost symbol of x

36

Example 3: $L = \{ w \ w^R \mid w \text{ in } \{a,b\}^* \}$

- Algorithm:
  1. Read symbol (a or b), mark it as checked (with an X) and "store" it in the state
     - If the symbol read is a Blank then accept.
  2. Move right until you hit a B, then move one position left.
  3. Check if symbol on tape = symbol stored in state, mark with B, move left, go to 4
     1. Else halt and reject
  4. Go to the left end of the input – skip over all symbols until you hit an X, move one tape cell to the right, and goto 1.

37

Example 1: $L = \{ w \ w^R \mid w \text{ in } \{a,b\}^* \}$

- Algorithm:
  1. State q_0 :
     - Read symbol a, write X to tape, move right, goto state $[q_1,a]$
     - Read symbol b, write X to tape, move right, goto state $[q_1,b]$
     - Read B, write B, move right goto final state q_f
  2. State q_1: Move right until you hit a B, then move one position left and goto state q_2…but keep the symbol stored in the state
     - From $[q_1,a]$ goto $[q_2,a]$
     - From $[q_1,b]$ goto $[q_2,b]$
  3. State q_2: Check if symbol on tape = symbol stored in state, mark with B, move left, go to 4 else halt and reject
     - If state = $[q_2,a]$ then check if input is a then goto q_3 write B to tape
     - If state = $[q_2,b]$ then check if input is b then goto q_3 write B to tape
  4. State q_3: Go to the left end of the input – skip over all symbols until you hit an X, move one tape cell to the right, and goto 1 (state q_0)

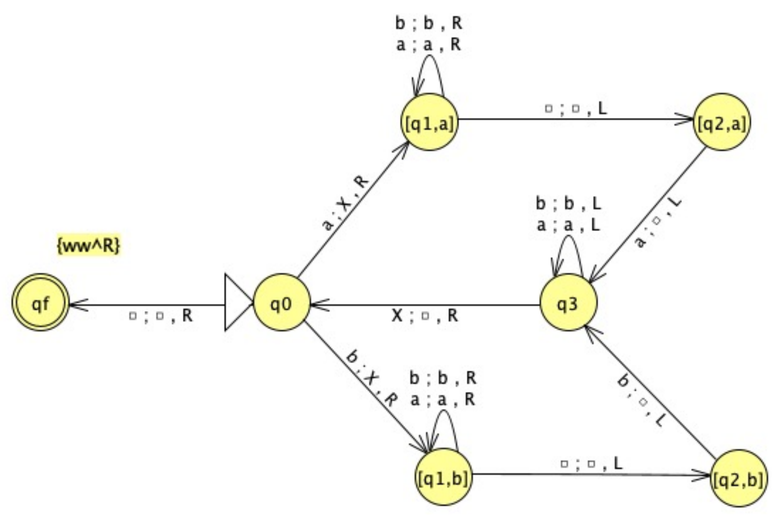
38

Example 1: L = { w w^R | w in {a,b}* }

- Transition Function: start state is q_0
  1. state q_0
     - $\delta(q_0, a) = ([q_1,a], X, R)$ /* read symbol, store in state */
     - $\delta(q_0, b) = ([q_1,b], X, R)$ /* write X , move right */
     - $\delta(q_0, B) = (q_f, B, R)$ /* change state to q1 */
  2. State q_1:
     - $\delta([q_1, b], a/b) = ([q_1,b], a/b, R)$ /* skip over all symbols until */
     - $\delta([q_1, b], B) = ([q_2,b], B, L)$ /* you read B */
     - $\delta([q_1, a], B) = ([q_2,a], B, L)$
  3. State q_2:
     - $\delta([q_2, a], a) = (q_3, B, L)$ /* check if symbol on tape equal to */
     - $\delta([q_2, b], b) = (q_3, B, L)$ /* symbol stored in state, mark with B */
       /* else reject and halt
  4. State q_3:
     - $\delta(q_3, a) = (q_3, a, L)$ /* go to left, skip over all symbols until X */
     - $\delta(q_3, b) = (q_3, b, L)$ /* you are going to the leftmost input that */
     - $\delta(q_3, X) = (q_0, X, R)$ /* not been checked yet */

39

Transition graph for $\{ww^R\}$



40