CS 3313
Foundations of Computing:

Modifications to the Turing Machine Model: Non-deterministic Turing Machine

http://gw-cs3313.github.io

1

Outline..

- Turing Machine model
  - TM as an automaton
- Computing functions on a Turing machine
  - Turing machine as a "computer"
  - First step is encoding the integer arguments to the TM
- TM "programming" techniques
  - Storage in the state, Checking symbols, Subroutines…
- Modifications to the basic TM model
  - Multiple tracks on the tape $\delta: Q \times \Gamma \rightarrow Q \times \Gamma^k \times \{L,R\}$
  - Semi-infinite tape
  - Multi-Tape TMs $\delta: Q \times \Gamma \rightarrow Q \times \Gamma^k \times \{L,R\}^k$
  - Non-deterministic TMs

2

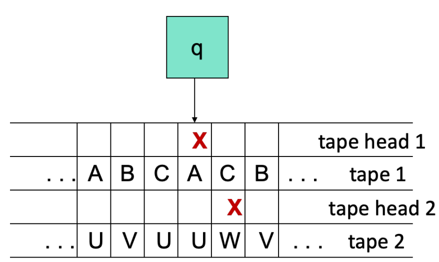
Summary of Results

- Theorem 1: A k-track single tape TM is equivalent to a one tape one track TM.
- Theorem 2: A two-way infinite TM is equivalent to a one way infinite tape TM.
- Theorem 3: A k-tape TM is equivalent to a one tape TM
  - Simulation used a 2k track TM, but then apply Theorem 1 to get equivalence to the basic TM
- Other results:
  - A Multi-dimensional TM Is equivalent to a one dimensional tape TM
  - A multi-tape head single tape TM is equivalent to a one head one tape TM

3

Simulating k Tapes by One: capturing the ID (snapshot)

- Use 2k tracks.
- Each tape of the k-tape machine is represented by a track.
- The head position for each track is represented by a mark on an additional track.



4

Simulating k Tapes by One

- 2k tracks to simulate the k tape TM
  - For each tape: 1 track for tape contents and 1 track to indicate location of tape head (indicated by an X)
- How many *X*'s to find = *k*
- Where to store symbol above X (read by a tape) = in the state!
- State in 1-tape *2k* track TM is $[q,\ a_1,a_2,..a_k]$
- To read all k symbols from the k tapes (on the k tracks)
  1. Sweep tape (Left to Right) past k "X" markers and store the k tape symbols in the state
  2. Sweep tape (Right to Left)
     1. Write symbol to the track
     2. Write X below to track below it and move R or L
  3. If TM halts in final state then accept, else go to 1

5

5

Next: Non-determinism

- From a single state, the machine can go to any of k states
  - Abstraction model: simultaneously go to all k, and replicate the machine
  - In reality: we cannot replicate the machine to arbitrarily large numbers
- In NFA and PDA: machine accepts the input w, if there is one sequence of choices that lead it to a final state.
  - Some of the sequence of choices may not lead to acceptance.
- NFA to DFA simulation: constructed power set of states
  - Won't work for TM (or PDA) since we also have to construct power set of the infinite storage
    - Power set of an infinite set is an uncountable set !!!

6

Why use non-determinism

- Powerful expressive model to describe a solution
  - If we are only interested in showing there is a solution
  - So first focus on what non-deterministic machines can solve and how to construct solution
- Can simplify the solution in some cases
  - Ex: “guess” the mid point of *ww* non-deterministically
- Coding Analogy: imagine you spawn multiple threads, as many as you want, and then wait for one of the threads to complete.
  - Since multiple “transitions” may be applied at each step:

    the program (i.e., machine) may have multiple active simultaneous threads,

    any of which may accept the input string when the thread halts

7

Nondeterministic TM’s

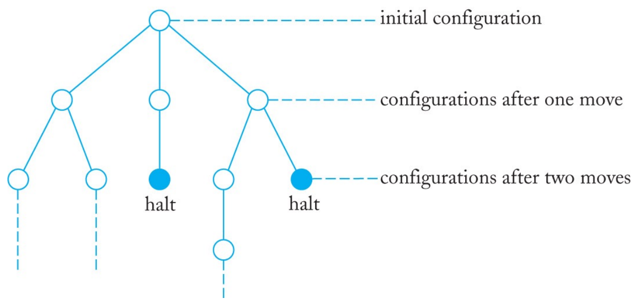
- Allow the TM to have a choice of move at each step.
  - Each choice is a state-symbol-direction triple, as for the deterministic TM.
- The TM accepts its input if a sequence of choices leads to an accepting state.
- Transition function: $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$
  - Set of choices
  - Each choice: goes to a state, writes to tape, moves L or R
  - $\delta(q_0, a) = \{(q_1, b, R), (q_2, c, L)\}$
  - *Ex:* $aq_0ab \vdash abq_1b$ *OR* $aq_0ab \vdash q_2acb$

8

8

Behavior of NDTMs

- From a configuration, i.e., ID, $\alpha p a \beta$ after one move the TM goes to another ID $\alpha c q_i \beta$ if $\delta(p, a) = (q_i, c, R)$ is contained in $\delta(p, a)$
  - (q_i, c, R) is one of the choices of moves the machine can make from (p,a)
- Machine starts in ID $q_0 w$



9

Non-deterministic TM - Example

Input= *aab*

- $\delta(q_0, a) = \{(q_1, a, R), (q_2, b, R)\}$
- $\delta(q_0, b) = \{(q_0, b, R)\}$

- $\delta(q_1, a) = \{(q_1, a, R), (q_3, a, L)\}$
- $\delta(q_1, b) = \{(q_1, a, R), (q_2, b, R)\}$

- $\delta(q_2, a) = \{(q_2, a, R), (q_1, a, L)\}$
- $\delta(q_2, a) = \{(q_1, b, R)\}$
- $\delta(q_2, B) = \{(q_4, B, R)\}$

- $\delta(q_3, a) = \{(q_3, a, L), (q_1, a, R)\}$
- $\delta(q_3, b) = \{(q_1, b, R), (q_1, B, L)\}$
- $\delta(q_3, B) = \{(q_3, B, L)\}$

10

Non-determinism and Sequence of Moves and Choices

- A NDTM has at most *k* choices of moves from its current state/configuration (if *k=1* then it is deterministic)
- Observation: we can number each of these *k* choices from *1 to k*
- Observation: At each configuration (state of NDTM), if we specify the choice it must make then machine is deterministic.
  - Assumption: if we specify a choice that does not exist then we assume machine halts and rejects
  - Ex: $\delta(q_2, a) = \{(q_1, b, R)\}$ does not have "choice 2"

$$\delta(q_2, a) = \{\overset{1}{(q_2, a, R)}, \overset{2}{(q_1, a, L)}\}$$

$$\delta(q_2, a) = \{\overset{1}{(q_1, b, R)}\}$$

11

Non-determinism and Sequence of Moves and Choices

- Generate a string of length *n*, using symbols *{1,2,…,k}*
- At each move *i, i-th* symbol specifies the choice that TM makes
  - Ex: String 121 – specifies choice 1 in first move, choice 2 in second move, choice 1 in third move.
  - IF the choice is not defined then M rejects and halts.
- This machine will behave in a deterministic manner !

$\delta(q_0, a) = \{\overset{1}{(q_1, a, R)}, \overset{2}{(q_2, b, R)}\}$
$\delta(q_0, b) = \{\overset{1}{(q_0, b, R)}\}$

$\delta(q_1, a) = \{\overset{1}{(q_1, a, R)}, \overset{2}{(q_3, a, L)}\}$
$\delta(q_1, b) = \{\overset{1}{(q_1, a, R)}, \overset{2}{(q_2, b, R)}\}$

$\delta(q_2, a) = \{\overset{1}{(q_2, a, R)}, \overset{2}{(q_1, a, L)}\}$
$\delta(q_2, a) = \{\overset{1}{(q_1, b, R)}\}$
$\delta(q_2, B) = \{\overset{1}{(q_4, B, R)}\}$

Input: ab
Sequence string: 12

$q_0ab \vdash$

Sequence= 11

$q_0ab \vdash$

12

Non-deterministic TM - Example

- $\delta(q_0, a) = \{(q_1, \overset{1}{a}, R), (q_2, \overset{2}{b}, R)\}$
- $\delta(q_0, b) = \{(q_0, \overset{1}{b}, R)\}$

- $\delta(q_1, a) = \{(q_1, \overset{1}{a}, R), (q_3, \overset{2}{a}, L)\}$
- $\delta(q_1, b) = \{(q_1, \overset{1}{a}, R), (q_2, \overset{2}{b}, R)\}$

- $\delta(q_2, a) = \{(q_2, \overset{1}{a}, R), (q_1, \overset{2}{a}, L)\}$
- $\delta(q_2, a) = \{(q_1, \overset{1}{b}, R)\}$
- $\delta(q_2, B) = \{(q_4, \overset{1}{B}, R)\}$

- $\delta(q_3, a) = \{(q_3, \overset{1}{a}, L), (q_1, \overset{2}{a}, R)\}$
- $\delta(q_3, b) = \{(q_1, \overset{1}{b}, R), (q_1, \overset{2}{B}, L)\}$
- $\delta(q_3, B) = \{(q_3, \overset{1}{B}, L), (q_1, \overset{2}{B}, R)\}$

- Let NDTM have at most k choices in any configuration
  - Ex: has 2 choices
- For every transition, we label each choice using 1 through k

13

Generating Sequences/Enumeration

- Recall from Lab review of countable sets, enumeration….
- How to generate (i.e., enumerate) strings (numbers?) of increasing lengths from alphabet {0,1} ?
  - 0, 1, 00, 01, 10, 11, 000, ….
- Strings of increasing length of radix k, alphabet $\{1,1,\ldots k\}$
- Define: w_i is set of strings of length i
  - w_0 = { empty string }
  - for each string w_i of length i /* $i=0$ to n. */

    for (radix k) $j=1$ to k

    print $j.w_i$ /* concatenation */

Key Takeaway: We can encode this algorithm as a Turing Machine

14

In-Class Exercise…..

- 1. Simulate the NDTM when given a sequence of choices
  - Specified as a string in base k (k =2)
- 2. Determine the properties of the tree (with all possible sequence of choices)

15

Question 1: NDTM and Sequence of Choices on input w= *aab*. Show the sequence of IDs or states

- $\delta(q_0, a) = \{(q_1, \overset{1}{a}, R), (q_2, \overset{2}{b}, R)\}$
- $\delta(q_0, b) = \{(q_0, \overset{1}{b}, R)\}$

- $\delta(q_1, a) = \{(q_1, \overset{1}{a}, R), (q_3, \overset{2}{a}, L)\}$
- $\delta(q_1, b) = \{(q_1, \overset{1}{a}, R), (q_2, \overset{2}{b}, R)\}$

- $\delta(q_2, a) = \{(q_2, \overset{1}{a}, R), (q_1, \overset{2}{a}, L)\}$
- $\delta(q_2, a) = \{(q_1, \overset{1}{b}, R) \}$
- $\delta(q_2, B) = \{(q_4, \overset{1}{B}, R) \}$

- $\delta(q_3, a) = \{(q_3, \overset{1}{a}, L), (q_1, \overset{2}{a}, R)\}$
- $\delta(q_3, b) = \{(q_1, \overset{1}{b}, R), (q_1, \overset{2}{B}, L)\}$
- $\delta(q_3, B) = \{(q_3, \overset{1}{B}, L) , (q_1,\overset{2}{B},R)\}$

Input= aab

(a)sequence= 122

$q_0 aab \vdash$?

$\vdash$?

$\vdash$?

sequence= 1121

$q_0 aab \vdash$?

$\vdash$?

$\vdash$?

$\vdash$?

$\vdash$?

16

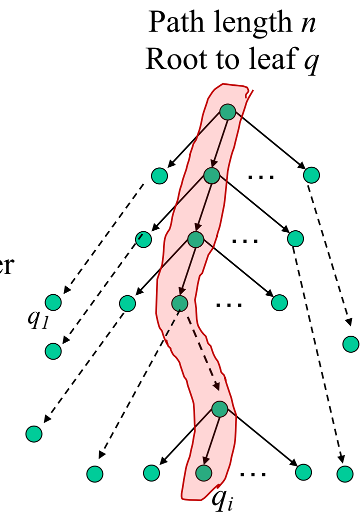
Question 2:

- Tree, each node has children $\leq k$
  - Tree with degree k
- Path from root to leaf q_i has length n

- Question A: what is the maximum number of leaves in this tree ?

- Question B: if we add the lengths of all the different paths from root to each leaf, what is the 'limit' of the total – express as Big-Oh ?

(i.e, length of root to q_1 +.. length root to q_i +…) $\leq$???



17

Now we are ready to simulate a NDTM on a TM

- Theorem: For any Non-deterministic Turing Machine, there is an equivalent deterministic Turing Machine.
  - If a language is accepted (or a function is computed) by a non-deterministic TM then there is a deterministic turing machine that accepts that language (or computes the function).
  - *NDTM and TM are equally powerful and solve the same class of problems !*

18

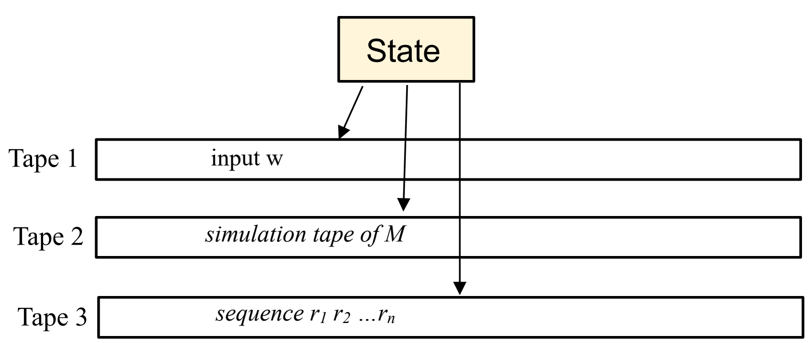
Equivalence of Non-deterministic and Deterministic Turing Machines

- Outline of Proof: we construct a multi-tape deterministic TM to simulate the moves of NDTM with at most *k* choices, for any *k*
- Observation 1: If we "force" the TM to make a choice *j* from the *k* choices then the TM is deterministic.
- Observation 2: For any length *n*, we can generate a sequence of *k-ary* (base *k*) numbers $r_1, r_2, \ldots r_n$ where $0 < r_i < k-1$
- Observation 3: Given a length *n* *k*-ary number, $r_1, r_2, \ldots r_n$, at each move j (from each configuration) the TM selects choice r_j of the NDTM, then this is a deterministic sequence of moves.

19

Equivalence of Non-deterministic and Deterministic Turing Machines: Simulation

- A NDTM N can be simulated on a 3-tape Deterministic TM M
- Tape 1 stores the input string w (input to the NDTM)
- Tape 2 is used for simulation of the NDTM behavior on the sequence of choices generated on Tape 3
- Tape 3 has the sequence of choices (generated by the TM M)



20

NDTM Simulation "algorithm"

1. Initially tape 1 contains input w, and tapes 2,3 are empty
   - Initialize tape 3 to λ
2. Copy tape 1 to tape 2
3. Use tape 2 to simulate NDTM N with input w on one branch of its computation -- specified by the sequence generated on tape 3.
4. Replace string on tape 2 with the next string in the string ordering (enumeration). Go to step 3.

21

NDTM Simulation "algorithm" – Step 3

a) (before each step of NDTM N) Look up symbol j on tape 3
   - Tape 3 moves left to right after each move of M in step (b)

b) M simulates choice j of NDTM N on tape 2
   - If j is the symbol read on tape 2, then $\delta(p,a) = (q,X,D)$ where (q,X,D) is the j-*th* choice in N
   - M goes to state q and writes X on tape 2 and moves in direction D

c) If no more symbols remain on tape 3 OR if this j-*th* choice is invalid then abort this branch and goto Step 4.

   If a rejecting configuration is achieved, then goto Step 4

   If an accepting configuration is achieved, then *accept* and halt.

22

Question:

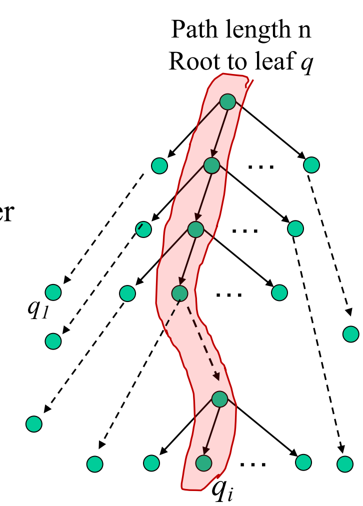
- For an input w, NDTM N accepts the string w after n steps/time (moves of N)
- The deterministic TM M accepts the string w in at most **???** steps (moves of M)

23

Recall observations from Question 2:

- Tree, each node has children $\leq k$
- Path from root to leaf q_i has length n
- Question A: what is the maximum number of leaves in this tree ? $O(k^n)$
- Question B: if we add the lengths of all the different paths from root to each leaf, what is the 'limit' of the total ?

$O(n\ k^n)$



24

BIG question.....

- Our simulation "algorithm" can simulate the NDTM on a DTM.
- If the NDTM can solve a problem (accept a language/compute a function) in O(n^i) steps then DTM can solve the problem in O(k^{n^i}) steps.
- Question: Can we design a simulation algorithm that can simulate the NDTM in O(n^i) steps ?
  - If you can help me design this, I will ask CS@GW to sign off on your graduation this semester !
  - Assigned as an extra credit homework ?

25

HUGE question.....

- A problem for which we have a polynomial time deterministic algorithm/Turing machine is said to be in the class P
- A problem for which we have a polynomial time non-deterministic algorithm/Turing machine is said to be in the class NP
- Is $P = NP$? *Open problem*

26

Where are we….

- Turing machine and equivalence of different models
- Equivalence of Non-deterministic and Deterministic TMs
  - Concept of "efficiency" of an algorithm…..
  - Polynomial time Deterministic Time (P) vs Non-deterministic Polynomial Time (NP)

- Next up…..
- Is there a "general purpose" Turing machine
  - It's input is a Turing machine and it simulates that TM

    analogous to a general purpose computer
- Are there problems that cannot be solved by a TM ?....Decidable and Undecidable Problems

27