Intro to Theory of Computation

CS 464

LECTURE 13

Last time:

- Turing Machines and Variants

Today

- Turing Machine Variants
- Church-Turing Thesis

Sofya Raskhodnikova

TMs are equivalent to…

TMs are equivalent to multitape TMs

(last time)

TMs are equivalent to nondeterministic TMs

(today)

TMs are equivalent to doubly unbounded TMs

(today)

TMs are equivalent to enumerators.

(today)

TMs are equivalent to FIFO automata.

(HW problem)

TMs are equivalent to primitive recursive functions.

TMs are equivalent to cellular automata.

CS 464

I-clicker problem (frequency: AC)

Which of these statements are valid descriptions of nondeterministic steps (in a PDA)?

A. Nondeterministically read the input and push it onto the stack.

B. Nondeterministically either read *a* and push it onto the stack or read *b* and pop *b* from the stack.

C. Nondeterministically read the input character *a* and either push it onto the stack or pop *b* from the stack.

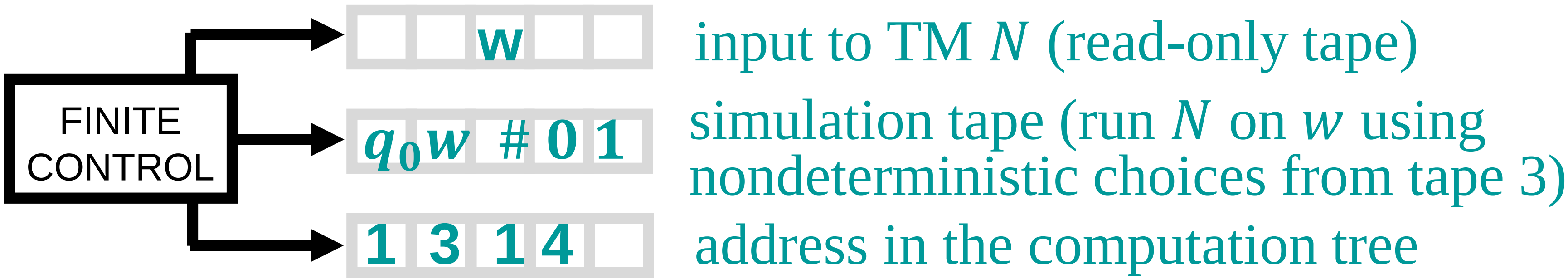
D. Nondeterministically push one of positive integers onto the stack.

E. None of the above.

NTMs are equivalent to TMs

Theorem. A deterministic TM can simulate a nondeterministic TM.

Proof idea: Consider an NTM N. Use a 3-tape TM.



- Let b be the largest # of nondeterministic choices N has in a step. Use alphabet $\{1, \dots, b\}$ for addresses.
- Do a BFS of the computation tree.

Doubly unbounded TMs

A TM with doubly unbounded tape is like an ordinary TM but

- Its tape is infinite on the left and on the right.

Initially, only the input is written on the tape and the head is on the first nonblack symbol.

CS 464

I-clicker problem (frequency: AC)

A TM can simulate a doubly unbounded TM U

A. by marking the leftmost ``investigated'' square and using a nondeterministic step every time U moves to the left of it.

B. by using 2 tapes: one for input + squares to the right; the other for squares to the left of the input.

C. by using each square of the tape to keep two characters from U's tape alphabet (2 tracks on the tape).

D. None of the above.

E. More than one choice above works.

TM variant: enumerator

- Starts with a blank tape
- Prints strings

L(E) = set of strings that E eventually prints.

Enumerator E **enumerates** language L(E).

May never terminate even if the language is finite.

May print the same string many times.

TMs vs. enumerators

Theorem. A language is Turing-recognizable ⇔ some enumerator enumerates it.

Proof:

⇐ Start with an enumerator E that enumerates A.

Give a TM that recognizes A.

⇒ Start with a TM that recognizes A.

Give an enumerator E that enumerates A.

CS 464

TMs are equivalent to…

TMs are equivalent to multitape TMs

(last time)

TMs are equivalent to nondeterministic TMs

(on the board)

TMs are equivalent to double unbounded TMs

(on the board)

TMs are equivalent to enumerators.

(on the board)

TMs are equivalent to FIFO automata.

(HW problem)

TMs are equivalent to primitive recursive functions.

TMs are equivalent to cellular automata.

The Church-Turing Thesis (1936)

L is recognized by a program for some computer*

↕

L is recognized by a TM

History

- **23 Hilbert's problems (1900)**
  - **stated at International Congress of Mathematicians**
  - **10th problem: Give a procedure for determining if a polynomial in *k* variables has an integral root.**

* The computer must be "reasonable"

The Church-Turing Thesis is consistent with all known "reasonable" computers

R1:	1101001…
R2:	1011001…
⋮	
RAM:	#1011#1101101#1011001#...#

Programs for a computer have instructions like
ADD R1, R2, R3; LOAD R1, R2; STORE R1,R2; MUL R1, R2, R2; BRANCH R1, X;…

Programming languages

- Programming languages like Java, Python, Scheme, C, … are equivalent to TMs
- We call such languages **Turing-complete**

Corollary. If two programming languages are Turing-complete, then they can recognize exactly the same set of languages.