Lecture 12—May 11, 2006
Turing Machines

Instructor: Neil Rhodes
Turing Machine

Has a one-way infinite tape
- Input is written on the tape, with blanks afterward

Has a current location on the tape (head)

Has a state-machine
- Based on the symbol under the head:
  - Writes a new symbol
  - Moves left-or-right
- Has two final states
  - Accept
  - Reject

Can’t go off the left-hand-side of the tape
Example Turing Machine

$L = \{ x+y=z \mid x, y, z \in \{0,1\}^*, \text{ z is binary representation of sum of binary representations of } x \text{ and } y \}$
Universal Turing Machine

A Universal Turing Machine

- Takes two inputs
  - M: description of a Turing machine
  - x: an input
- Computes M(x)
  - Accepts if M on x accepts
  - Rejects if M on x rejects

How is M encoded?

- M consists of:
  - Q: set of states
  - Σ: input alphabet
  - Γ: tape alphabet
  - δ: Q x Σ → Γx{L,R} transition function
  - q₀ ∈ Q: start state (first state will be start state)
  - q_{accept} ∈ Q: accept state (second state will be accept state)
  - q_{reject} ∈ Q (q_{reject} ≠ q_{accept}): reject state (third state will be reject state)
- Encode with string over \{0, 1\}*: use unary numbers: 0^n represents n’th stater 0^m represents m’th input symbol 0^p represents p’th tape symbol. Use 1’s as separators
Example encoding

3 states: first, accept, and reject (0, 00, and 000)
2 input symbols: a, c (0 and 00)
3 tape symbols: a, c, <blank> (0, 00, and 000)
2 directions: L, R (0 and 00)
3 transitions: when reading a 0 stay in first state. When reading a 1, go to reject state. When reading a blank, go to accept state

000 11
00 11
000 11
000 11
0 1 0 1 0 1 0 1 00 11
0 1 00 1 000 1 00 1 00 11
0 1 000 1 00 1 000 1 00 11
Encoding Turing Machine with input

Encode Turing Machine M in binary

Encode input, x, in binary
  ▪ If M has more than two input symbols, encode each symbol in binary

Write out:
  ▪ M111x

Now, construct U, a Turing machine that:
  ▪ Has input M111x
  ▪ Simulates M on x
  ▪ How?
    - Tape for current state
    - Tape to hold M's simulated tape
    - Marker for current tape location
    - Starts at given start state
    - If simulation of M ever enters accept or reject state
      - U enters its own accept or reject state
Simulating Nondeterministic Turing Machines

Can we write U, the universal Turing machine as deterministic?
  - Given \( M_w \), does the nondeterministic TM \( M \) accept \( w \)?

Idea:
  - 3 tapes:
    - Tape 1 contains \( M \# w \)
    - Tape 2 contains a copy of \( M \)'s tape (on some non-deterministic branch)
    - Tape 3 contains record of non-deterministic choices
      - Will be generated deterministically. Consists of \#;\#;\#;\#;\#;\# where each \# is in range 1..maxBranchingFactorOfM
  - 4 stages of \( U \):
    - Stage 1: Tape 2 and 3 are blank
    - Stage 2: Copy \( w \) to tape 2
    - Stage 3: Use tape 2 to simulate \( M \). On every step of \( M \), consult tape 3 to determine which choice to make of the allowable transitions. If illegal, or no more symbols in tape 3, goto stage 4. If \( M \) accepts, accept
    - Stage 4: Replace string on tape 3 with lexicographically next string. Simulate next branch by going to stage 2.
Recognizing vs. Deciding

L is Turing-recognizable (recursively enumerable if)
- There exists a TM, $M$ where every string $s$ in $L$
  - is accepted by $M$

L is Turing-decidable (recursive) if
- There exists a TM, $M$ where, for every string $s$:
  - If $S$ in $L$, $M$ accepts $L$
  - If $S$ not in $L$, $M$ rejects $L$
- That is, $M$ (eventually) halts on all inputs
Closure properties of Turing-decidable languages

Union

Intersection

Concatenation

Reverse
Closure properties of Turing-recognizable languages

Union

Intersection

Concatenation

Reverse