Fall, 2006
Lecture 12—November 2, 2006
Nondeterministic Turing Machines
Church-Turing Thesis
Decidable Languages

Instructor: Neil Rhodes
Non-deterministic TM

Transition function maps to set of:
- state
- new tape symbol
- Left or Right

Computation is a tree

TM accepts if:

TM rejects if:
Any Non-det TM has an equivalent Det TM

Given:
- M, a nondeterministic one-way-infinite one-tape TM

Compute
- M' which simulates M

Idea:
- 3 tapes:
  - Tape 1 contains w, the input
  - 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 M':
  - 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.
Church-Turing Thesis

Informal definition of algorithm:
- Collection of simple instructions for carrying out some task

Formal definition of algorithm:
- Turing machine
- Lambda calculus

Thesis:
- Any informal algorithm can be carried out with one of the two formal definitions

Extended thesis
- Any two programming languages have the same power
  - As long as they are Turing-complete
Example of Equivalence

Emulate x86 with Turing Machine
- Program
- Memory
- Instruction set
Decidable Languages

3 levels of description of a Turing machine algorithm

- **Formal description**
  - Complete state machine

- **Implementation description**
  - English description of how the Turing machine works (moves its head, stores its data)
  - No details of states or transition function

- **High-level description**
  - Describe algorithm, but no mention of tapes or head movement
  - From now on, we’ll mostly use this level

- **Input to TM is a string**
  - How to input a graph, grammar, automata, etc.

  - Given object O, encoding of object is represented as <O>
Decidable Languages

\[ A_{DFA} = \{<B, w>| B \text{ is a DFA that generates string } w\} \]

\[ A_{DFA} \text{ is decidable} \]
Decidable Languages

\[ A_{\text{NFA}} = \{ <B, w> | B \text{ is a NFA that generates string } w \} \]

\[ A_{\text{NFA}} \text{ is decidable} \]
Decidable Languages

\[ E_{DFA} = \{ <A> \mid A \text{ is a DFA and } L(A) = \emptyset \} \]

\[ E_{DFA} \text{ is decidable} \]
Decidable Languages

\[ \text{EQ}_{\text{DFA}} = \{ <A, B> | A \text{ and } B \text{ are DFAs and } L(A) = L(B) > \} \]

\[ \text{EQ}_{\text{DFA}} \text{ is decidable} \]