More Turing Machines

Sipser 3.2 (pages 148-154)

Multitape Turing Machines

• Formally, we need only change the transition function to

\[ \delta: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L,R\}^k \]
Seems more powerful, but...

- Theorem 3.13: Every multitape Turing machine has an equivalent single-tape Turing machine.

Turing-recognizable languages

- Corollary 3.15: A language is Turing-recognizable if and only if some multitape Turing machine recognizes it.
Recognizing Composite Numbers

• Let \( L = \{I^n \mid n \text{ is a composite number}\} \)
• Designing a Turing machine to accept \( L \) would seem to involve factoring \( n \)
• However, if we could guess ...

Guessing

• Design a machine \( M \) that on input \( I^n \) performs the following steps:
  1. Nondeterministically choose two numbers \( p, q > 1 \) and transform the input into \( #I^n#I^p#I^q# \)
  2. Multiply \( p \) by \( q \) to obtain \( #I^n#I^{pq}# \)
  3. Checks the number of \( I \)'s before and after the middle \( # \) for equality
  • Accepts if equal, and rejects otherwise
Nondeterministic Turing machines

• Simply modify the transition function to satisfy:
  \[ \delta : Q \times \Gamma \rightarrow P(Q \times \Gamma \times \{L,R\}) \]

Guessing doesn’t buy us anything!

• Theorem 3.16: Every nondeterministic Turing machine has an equivalent deterministic Turing machine.
Equivalence with Turing machines

- Theorem 3.21: A language is Turing-recognizable if and only if some enumerator enumerates it.
- Proof
  \((\Leftarrow)\) Suppose enumerator \(E\) enumerates \(L\).
  Define \(M = \text{"On input \(w\):}\)
  1. Run \(E\). Every time that \(E\) outputs a string, compare it with \(w\).
  2. If \(w\) ever appears in the output of \(E\), accept."
Equivalence with Turing machines

- Theorem 3.21: A language is Turing-recognizable if and only if some enumerator enumerates it.
- Proof
  \( \Rightarrow \) Suppose TM \( M \) recognizes \( L \). Build a lexicographic enumerator to generate the list of all possible strings \( s_1, s_2, \ldots \) over \( \Sigma^* \). Define \( E = \) "Ignore input."
  1. Repeat the following for \( i = 1, 2, 3, \ldots \)
  2. Run \( M \) for \( i \) steps on each input \( s_1, s_2, \ldots, s_i \).
  3. If any computations accept, print out corresponding \( s_i \).

TMs take their own sweet time...

- Recognizers, like enumerators may take a while to answer yes, ... and even longer to answer no
- A TM that halts on all inputs is called a decider
- A decider that recognizes a language is said to decide that language
- Call a language Turing-decidable if some Turing machine decides it
Recognizers and Deciders

• Theorem: A language is Turing-decidable if and only if both it and its complement are Turing-recognizable
• Proof:
  ($\Rightarrow$) By definition.

($\Leftarrow$) Simulate, in parallel, $M_1$ on tape 1 and $M_2$ on tape 2.