15-251: Great Theoretical Ideas In Computer Science

Recitation 4

Announcements

Reminders:
- Midterm 1 during February 20 writing session.
- Solution session on Sunday, regrades due Wednesday

Review

(Un)decidability and reductions:
- A **decider** is a TM that halts on all inputs.
- A language \( L \) is **undecidable** if there is no decider TM \( M \) such that \( M(x) \) accepts if and only if \( x \in L \).
- A language \( A \) **reduces** to \( B \) if it is possible to decide \( A \) given access to a subroutine that decides \( B \), Denote this as \( A \leq_T B \) or simply \( A \leq B \) (read: \( B \) is at least as hard as \( A \)).

Time Complexity and big-Oh:
- The running time of an algorithm \( A \) is a function \( T_A : \mathbb{N} \rightarrow \mathbb{N} \) defined by
  \[ T_A(n) = \max_{I \in S} \{ \text{number of steps } A \text{ takes on } I \} \], where \( S \) is the set of instances \( I \) of size \( n \).
- For \( f, g : \mathbb{N}^+ \rightarrow \mathbb{R}^+ \), we say \( f(n) = O(g(n)) \) if there exist constants \( c, n_0 > 0 \) such that \( \forall n \geq n_0 \), we have \( f(n) \leq cg(n) \).
- For \( f, g : \mathbb{N}^+ \rightarrow \mathbb{R}^+ \), we say \( f(n) = \Omega(g(n)) \) if there exist constants \( c, n_0 > 0 \) such that \( \forall n \geq n_0 \), we have \( f(n) \geq cg(n) \).
- For both of the above, your choice of \( c \) and \( n_0 \) cannot depend on \( n \).
- For \( f, g : \mathbb{N}^+ \rightarrow \mathbb{R}^+ \), we say \( f(n) = \Theta(g(n)) \) if \( f(n) = O(g(n)) \) and \( f(n) = \Omega(g(n)) \).

Can’t tell if one is regular

Prove that the following language is undecidable where \( M \) refers to a TM:

\[ \text{REGULAR} = \{ \langle M \rangle : L(M) \text{ is regular} \} \]

\( O, I \) Think I Understand Asymptotics Now

Let \( f, g, h \) be functions from \( \mathbb{N} \) to \( \mathbb{N} \). Prove or disprove the following:

(a) If \( f = O(g) \) and \( g = O(h) \), then \( f = O(h) \)

(b) If \( f = O(g) \), then \( g = O(f) \)

(c) \( f = O(g) \) or \( f = \Omega(g) \)
**Bits and Pieces**

Determine which of the following problems can be computed in worst-case polynomial-time, i.e. $O(n^k)$ time for some constant $k$, where $n$ denotes the number of bits in the binary representation of the input. If you think the problem can be solved in polynomial time, give an algorithm in pseudo-code, explain briefly why it gives the correct answer, and argue carefully why the running time is polynomial. If you think the problem cannot be solved in polynomial time, then provide a proof.

(a) Give an input positive integer $N$, output $N!$.

(b) Given as input a positive integer $N$, output True iff $N = M!$ for some positive integer $M$.

(c) Given as input a positive integer $N$, output True iff $N = M^2$ for some positive integer $M$.

(Extra) Lose All Scripted Responses. Improvisation Only

Let $\text{FINITE} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is finite}\}$. and $\text{TOTAL} = \{\langle M \rangle : M \text{ halts on all inputs}\}$. Show that $\text{TOTAL} \leq_T \text{FINITE}$.

(Extra) Asymptotically super sub

Name a function $f(n)$ which is asymptotically super-polylogarithmic, i.e., $f(n) = \Omega(\log^c n)$ for any constant $c > 1$, and at the same time asymptotically sub-polynomial, i.e., $f(n) = O(n^\epsilon)$ for any constant $\epsilon > 0$. 