CS310

Complexity

Section 7.1

November 21, 2008

Running time

- $A = \{0^K 1^K \mid k >= 0\}$
  - how long (how many steps?) will it take a single-tape TM to accept or reject a string?

- The running time
  - input of length **n**
  - worst case running time
- M is a “f(n) time TM”

Example

- $f(n) = 5n^3 + 4n^2 + 6n + 1$
  - the goal here is to see how the running time grows as n increases
  - for large n, $5n^3$ dominates this equation
  - coefficient 5 is immaterial
  - we say $f(n) = n^3$

Big Oh O( )

- Asymptotic analysis
  - estimate runtime of algorithm (or TM) on large inputs
  - only look at highest order term
  - allows us to compare runtime of two algorithms

Definition: Big Oh

- f, g are functions: f,g: $N \rightarrow R^+$

  $f(n) = O(g(n))$ if positive ints c and n_0 exist such that for every int $n >= n_0$

  $f(n) <= c*g(n)$

g(n) is an *asymptotic upper bound* for f(n)

some constant multiple of g(n) eventually dominates f(n)

- *R^+: set of non-negative real numbers*

Example

- $f(n) = 5n^3 + 2n^2 + 22n + 6$
- $O(f(n)) = n^3$
- let $c = 6$ and $n_0 = 10$

- $5n^3 + 2n^2 + 22n + 6 <= 6n^3$
  - for every $n >= n_0$
- $O(f(n)) = n^4$ as well, but we want the tightest upper bound

Logarithms

- if $x = \log_2 n$ then $2^x = n$

  so $\log_b 2^x = \log_b n$

  so $x \log_b 2 = \log_b n$

  so $x = (\log_b n) / (\log_b 2)$

  so $\log_b (n) = O(\log_2 n)$ for any base

  because $\log_b 2$ is a constant

Example

- $f(n) = 3n \log_2 n + 5n\log_2 (\log_2 n) + 2$

  $f(n) = O(g(n)) = ?$

  Since $\log_2 n <= n$ then

  $\log_2 (\log_2 n) <= \log_2 (n)$

  so $f(n) = O(n \log_2 n)$

Analyzing Algorithms

- $A = \{0^k 1^k \mid k >= 0\}$

  on input of length n:

  1) scan, reject if 0 found to right of a 1

  2) if both 0's and 1's remain, scan, cross off single 0, single 1

  3) if 0's remain after 1's crossed off or conversely, reject. otherwise accept.

Analysis

- Step 1: scan, verify: n steps forward, n steps back: 2n steps so O(n)
- Step 2: scan, cross off 0 and 1 each scan. Each scan uses O(n) steps, n/2 scans *at most*, so $O(n^2)$
- Step 3: Scan, accept or reject O(n)
- Total: $O(n) + O(n^2) + O(n)$
  - $O(n^2)$

Algorithm

- If we had a two tape TM, could we do this in O(n)?
  – linear time?

Complexity relationships between models

- Theorem 7.8: let t(n) >= n, every t(n) time multitape TM has an equivalent $O((t(n)^2)$ time single-tape TM.

- Theorem 7.9: Every t(n) >= n time ND single tape TM has an equivalent $2^{O(t(n))}$ time deterministic single tape TM