COT 6405
ANALYSIS OF ALGORITHMS

Recurrences

Computer & Electrical Engineering and Computer Science Department
Florida Atlantic University

Recurrence

A *recurrence* is an equation or inequality that describes a function in terms of its value on smaller inputs

Examples:

$T(n) = 2T(n/2) + n \quad \text{for } n > 1$

$T(1) = \Theta(1)$

$T(n) = T(n-1) + n \quad \text{for } n > 0$

$T(0) = 0$

Methods for Solving Recurrences

- No universal method that can be used to solve every recurrence
- Techniques:
  - Method of forward substitutions
  - Method of backward substitutions
  - Master Theorem

Method of Forward Substitutions

- Start from the initial term(s) and use the recurrence equation to generate the first few terms, in the hope of seeing a pattern that can be expressed by a closed-end formula
- If such a formula is found, check its validity:
  - Substitute into the recurrence equation and the initial condition, OR
  - Prove using mathematical induction

Method of Forward Substitutions

Example:

$$T(n) = 2T(n-1) + 1 \quad \text{for } n > 1$$
$$T(1) = 1$$

Solution: $T(1) = 1$

$T(2) = 3$

$T(3) = 7$

$T(4) = 15$

Observation: these numbers are one less than consecutive powers of 2

$$T(n) = 2^n - 1 \quad \text{for } n \geq 1$$

- Check validity

Method of Backward Substitutions

- Using the recurrence, express $T(n – 1)$ as a function of $T(n – 2)$ and substitute into the original equation to get $T(n)$ as a function of $T(n – 2)$
- Repeat this step and get $T(n)$ as a function of $T(n – 3)$
- So on…. in the hope of seeing a pattern in expressing $T(n)$ as a function of $T(n – i)$, $i = 1, 2, \ldots$
- Selecting i to make $n – i$ reach the initial condition and using one of the standard summation formulas often leads to a closed-end formula

Method of Backward Substitutions

Example:

$$T(n) = T(n-1) + n \quad \text{for } n > 0$$

$$T(0) = 0$$

Solution:

$T(n-1) = T(n-2) + n - 1 \Rightarrow T(n) = T(n-2) + (n-1) + n$

$T(n-2) = T(n-3) + n - 2 \Rightarrow T(n) = T(n-3) + (n-2) + (n-1) + n$

After i substitutions:

$$T(n) = T(n-i) + (n-i+1) + (n-i+2) + \ldots + n$$

Taking i = n, we get:

$$T(n) = T(0) + 1 + 2 + 3 + \ldots + n = n(n+1)/2 \quad \text{(arithmetic series)}$$

Master Theorem (CLRS pg 95)

Let a $\geq$ 1 and b >1 be constants, let f(n) be a function, and let T(n) be defined on nonnegative integers by the recurrence:

T(n) = $aT\left(\frac{n}{b}\right)$ + f(n)

1. If f(n) = $O(n^{\log_b^a - \varepsilon})$ for some constant ε > 0, then T(n) = $\Theta(n^{\log_b^a})$
2. If f(n) = $\Theta(n^{\log_b^a})$, then T(n) = $\Theta(n^{\log_b^a} \lg n)$
3. If f(n) = $\Omega(n^{\log_b^a + \varepsilon})$ for some constant ε > 0, and if

   af(n/b) $\leq$ cf(n) for some constant c < 1 and all sufficiently large n, then

   T(n) = $\Theta(f(n))$