Overview

- Asymptotic Notations – Review
- Recurrence relation
  - Factorial
  - Binary Search
  - Merge Sort
- Solution of recurrence

Asymptotic Notations - Review

- O (Oh), Ω (Omega), θ (Theta)
  - Small Oh, Small Omega - o ω
- Definitions (sets)
- Insertion Sort – Analysis
  - Worst Case
  - Best Case

Asymptotic Notations - Exercises

1. **Prove that $2n^2 + 3n + 6$ is $\theta(n^2)$**
2. $T(n) = \theta(1)$?
3. $T(n) = 10000n + 10n^2$. Is $T(n)$ $O(n^2)$?
4. **Which algorithm do you prefer?**
   a. $\theta(n^2)$ or $\theta(n)$
   b. $\theta(n)$ or $\theta(\lg n)$

Factorial - Recursive function

```
int factorial (int n){
// returns the factorial of n, given n>=0
    if (n<=1)
        return 1;
    else
        return n * factorial (n-1);
}
```

Running Time T(n)= ?

Factorial – Running Time

$$T(n) = T(n-1) + c \quad \text{if } n>1$$
$$ = d \quad \text{if } n<=1$$

Asymptotic Running Time?

Binary Search – Running Time

$$
\begin{aligned}
T(n) &= T(n/2) + c && \text{if } n>1 \\
&= d && \text{if } n<=1
\end{aligned}
$$

Merge Sort – Running Time

$$T(n) = 2T(n/2) + cn \qquad \text{if } n>1$$
$$ = c \qquad \text{if } n=1$$

Running Time - Recurrence equation

- Running time of recursive algorithms described by a recurrence equation or recurrence
- $T(n)$ in terms of running times of smaller subproblems
- Solve the recurrence using mathematical tools to get bounds on the running time

Solving recurrence - Factorial

$$T(n) = c + T(n\text{-}1) \quad \text{if } n>1$$

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

$$T(n) = c + c + T(n\text{-}2) \quad \text{if } n>2$$

$$= 2c + T(n\text{-}2) \quad \text{if } n>2$$

Factorial – Running Time

$T(n) \;=\; 2c + T(n\text{-}2) \quad if\ n>2$

$T(n) \;=\; 3c + T(n\text{-}3) \quad if\ n>3$

In general,

$T(n) \;=\; ic + T(n\text{-}i) \quad if\ n>i$

when $i = n\text{-}1$**,**

$$T(n) \;=\; (n\text{-}1)c + T(1) \;= (n\text{-}1)c + d \;= cn - c + d$$

T(n) is Θ(n)

Solving Recurrence – Iteration method

- Expand (iterate) the recurrence
- Express as a summation of terms dependent only on *n*
- **Recursion Tree** - Visualize the iteration of recurrence

Divide and Conquer – Recurrence

$$T(n) = \Theta(1) \qquad \text{if } n <= c$$
$$= a\,T(n/b) + D(n) + C(n) \qquad \text{otherwise}$$

- Number of subproblems – ***a***
- Each subproblem size is ***1/b*** the size of the original
- ***D(n)*** – time to divide the problem into subproblems
- ***C(n)*** – time to combine the solutions

Divide and Conquer – Recurrence

$$T(n) = d \qquad \textbf{if } n<=1$$

$$= 2\,T(n/2) + c \qquad \textbf{otherwise}$$

Solve using iteration method

Divide and Conquer – Recurrence

$$T(n) = d \qquad \text{if } n<=1$$

$$= 2\,T(n/2) + cn \qquad \text{otherwise}$$

Solve using iteration method

Reference

T H Cormen, C E Leiserson, R L Rivest, C Stein *Introduction to Algorithms,* 3rd ed., PHI, 2010