Data Structure

Lecture#5: Algorithm Analysis (Chapter 3)

U Kang
Seoul National University

In This Lecture

- Learn how to evaluate algorithm efficiency
- Learn the concept of average case, best case, and worst case running time
- Learn the important concept of “Asymptotic Analysis”
  - Big-Oh
  - Omega
  - Theta

Algorithm Efficiency

- There are often many approaches (algorithms) to solve a problem. How do we choose between them?
- At the heart of computer program design are two (sometimes conflicting) goals.
  1. To design an algorithm that is easy to understand, code, debug.
  2. To design an algorithm that makes efficient use of the computer's resources.

Algorithm Efficiency (cont)

- Goal (1) is the concern of Software Engineering.
- Goal (2) is the concern of data structures and algorithm analysis.
- When goal (2) is important, how do we measure an algorithm's cost?

How to Measure Efficiency?

- Measuring efficiency
  - Empirical comparison (run programs)
  - Asymptotic Algorithm Analysis
- Critical resources: time, space, programmers effort, ease of use
- For most algorithms, running time depends on "size" of the input.
- Running time is expressed as **T**(*n*) for some function **T** on input size *n*.
- We count the number of **basic** operations

How to Measure Efficiency?

- We count the number of **basic** operations
- Basic operation: its time to complete does not depend on the values of its operands or *n*.
  - Corresponds to one statement in a programming language
    - Exception: for/while loop
  - Addition, subtraction, division, multiplication
  - Assignment
  - Comparison
  - …

Examples of Growth Rate

- Example 1.

```
/** @return Position of largest value in "A“ */
static int largest(int[] A) {
  int currlarge = 0;  // Position of largest
  for (int i=1; i<A.length; i++)
    if (A[currlarge] < A[i])
       currlarge = i; // Remember pos
  return currlarge;   // Return largest pos
}
```

Examples (cont)

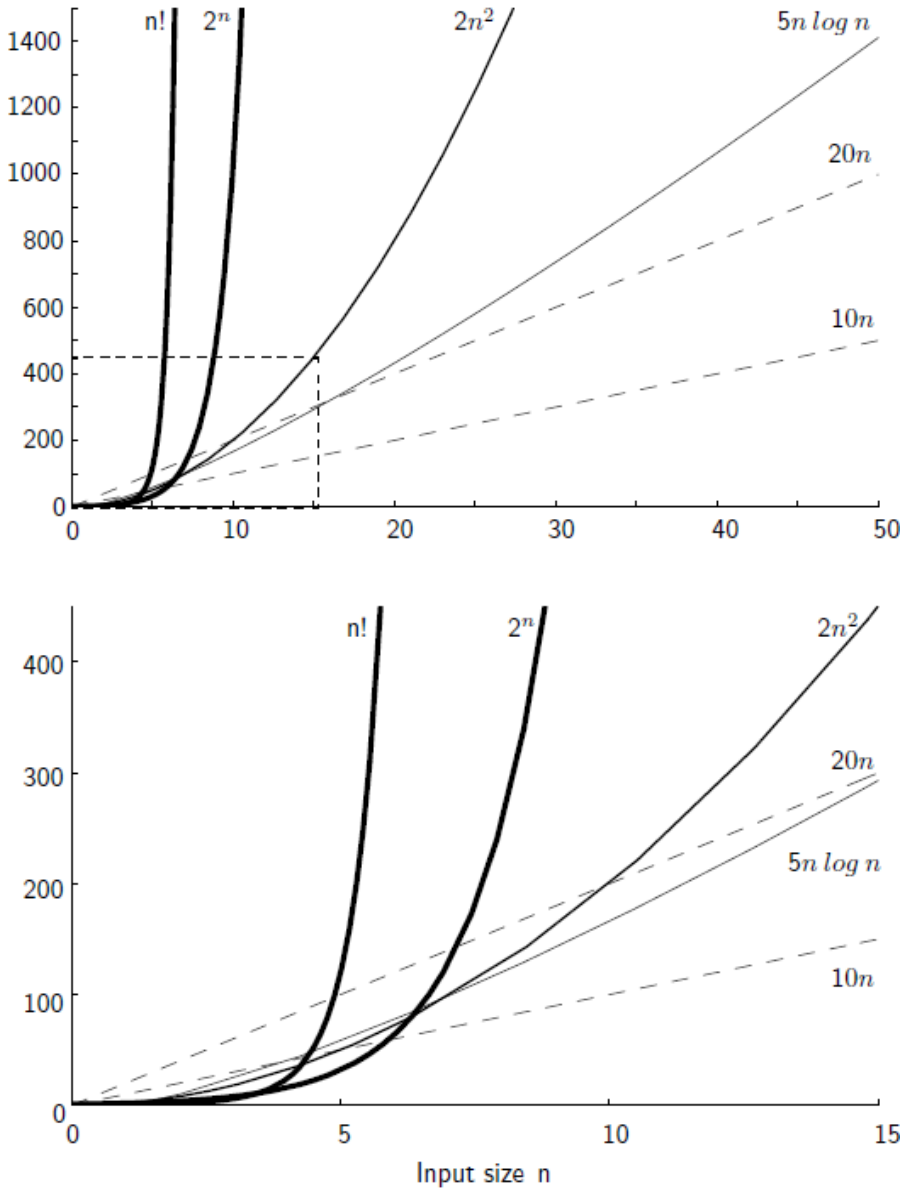
- Example 2: Assignment statement.

x = 1;

- Example 3:

```
sum = 0;
for (i=1; i<=n; i++)
  for (j=1; j<=n; j++)
    sum++;
}
```

Growth Rate Graph



Stirling's Approximation

$$n! \sim \sqrt{2\pi n}(\frac{n}{e})^n$$

Best, Worst, Average Cases

- Not all inputs of a given size take the same time to run.
- Sequential search for *K* in an array of *n* integers: begin at first element in array and look at each element in turn until K is found
  - Best case cost:
  - Worst case cost:
  - Average case cost:

Which Analysis to Use?

- Average time?
- Best case time?
- Worst case time?

Which Analysis to Use?

- While average time appears to be the fairest measure, it may be difficult to determine.
- When is the best case time important?
- When is the worst case time important?

Faster Computer or Algorithm?

- Suppose we buy a computer 10 times faster.
  - i.e., process 10 times larger number of operations at the same interval
- *n*: size of input that can be processed in one second on old computer
- *n'*: size of input that can be processed in one second on new computer

f(x) = # of operations for input of size x
Find n' such that f(n') = 10 f(n)

f(n)	n	n′	Change	n′/n
10n	1000	10,000	$n' = 10n$	10
20n	500	5000	$n' = 10n$	10
5n log n	250	1842	$\sqrt{10}n < n' < 10n$	7.37
$2n^2$	70	223	$n' = \sqrt{10}n$	3.16
2^n	13	16	$n' = n + 3$	−−

Asymptotic Analysis: Big-oh

- (Informal) Definition: **T**(*n*) is in the set O(*f*(*n*)) if **T**(*n*) <= *cf*(*n*) for large n (c is a constant you can choose)
- Examples
  - T(n) = 5n is in the set O(n)
  - T(n) = 5n + 6 is in the set O(n)
  - T(n) = $4n^2 + 3n + 7$ is in the set O(n^2)
  - T(n) = $4n^2 + 3n + 7$ is **not** in set O(n)
    - Why?

Asymptotic Analysis: Big-oh

- (Formal) Definition: For **T**(n) a non-negatively valued function, **T**(n) is in the set O(f(n)) if there exist two positive constants c and n_0 such that **T**(n) <= cf(n) for all $n > n_0$.
- Use: The algorithm is in O(n^2) in [best, average, worst] case.
- Meaning: For all data sets big enough (i.e., $n>n_0$), the algorithm always executes in less than cf(n) steps in [best, average, worst] case.

Big-oh Notation (cont)

- Big-oh notation indicates an upper bound.

- Example: If $\mathbf{T}(n) = 3n^2$ then $\mathbf{T}(n)$ is in $O(n^2)$.
  - Look for the tightest upper bound:
    While $\mathbf{T}(n) = 3n^2$ is in $O(n^3)$, we prefer $O(n^2)$.

Big-Oh Examples

- Example 1: Finding value *X* in an array (average cost).
  - Then $\mathbf{T}(n) = c_s n/2$.
  - For all values of $n > 1$, $c_s n/2 <= c_s n$.
  - Therefore, the definition is satisfied for $f(n)=n$, $n_0 = 1$, and $c = c_s$.
  - Hence, $\mathbf{T}(n)$ is in $O(n)$.

Big-Oh Examples (2)

- Example 2: Suppose $\mathbf{T}(n) = c_1n^2 + c_2n$, *where* c_1 and c_2 are positive.
  - $c_1n^2 + c_2n <= c_1n^2 + c_2n^2 <= (c_1 + c_2)n^2$ for all $n > 1$.
  - **Then** $\mathbf{T}(n) <= cn^2$ whenever $n > n_0$, for $c = c_1 + c_2$ and $n_0 = 1$.
  - Therefore, $\mathbf{T}(n)$ is in $O(n^2)$ by definition.
- Example 3: $\mathbf{T}(n) = d$. Then $\mathbf{T}(n)$ is in $O(1)$.

A Common Misunderstanding

- "The best case for my algorithm is n=1 because that is the fastest." WRONG!
- Big-oh refers to a growth rate as n grows to ∞.
- Best case is defined for the input of size n that is cheapest among all inputs of size n.

Big-Omega

- Definition: For $\mathbf{T}(n)$ a non-negatively valued function, $\mathbf{T}(n)$ is in the set $\Omega(g(n))$ if there exist two positive constants c and n_0 such that $\mathbf{T}(n) >= cg(n)$ for all $n > n_0$.
- Meaning: For all data sets big enough (i.e., $n > n_0$), the algorithm always requires more than $cg(n)$ steps.
- Lower bound.

Big-Omega Example

- $\mathbf{T}(n) = c_1 n^2 + c_2 n$.
  - $c_1 n^2 + c_2 n >= c_1 n^2$ for all $n > 1$.
  - $\mathbf{T}(n) >= cn^2$ for $c = c_1$ and $n_0 = 1$.
  - Therefore, $\mathbf{T}(n)$ is in $\Omega(n^2)$ by the definition.
- We want the greatest lower bound.

Theta Notation

- When big-Oh and Ω coincide, we indicate this by using Θ (big-Theta) notation.

- Definition: An algorithm is said to be in $\Theta(h(n))$ if it is in $O(h(n))$ and it is in $\Omega(h(n))$.

What you need to know

- The concept of average case, best case, and worst case running time
  - When they are important
  - Evaluate the [average, best case, worst case] running time of codes
- Intuition and definition of “Asymptotic Analysis”
  - Big-Oh, Omega, Theta
  - Evaluate the complexity of codes using asymptotic analysis

Questions?

Big O in Real World

- Built for Yeosu EXPO in 2012

Big O in Real World

- Big-O show