Complexity Analysis

Parinya Suwansrikham
Software Engineering
College of Arts, Media and Technology
Chiang Mai University

Agenda

- The Growth of Functions
  - Big-O Notation
  - The growth of combinations of function
- Algorithms
  - Searching Algorithms
- Complexity of Algorithms

The Growth of Functions

- Suppose that a computer program is doing (solving) a task (problem).
- One important concerning is how long a computer takes to solve this problem.
- For example, a computer program reorder any list of n integers into a list of ascending order.

The Growth of Functions

- The time to reorder is less than f(n) microseconds, where $f(n) = 100n \log n + 25n + 9$
- To analyze this program, we need to under stand how quickly this function grows as n grows.

Big-O Notation

- **DEFINITION 1**. Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$ if there are constants C and k such that

  $$|f(x)| \le C|g(x)|$$

  whenever $x > k$. (This is read as "$f(x)$ is big-oh of $g(x)$.")

- To show $f(x)$ is $O(g(x))$, we need only find one pair of constants C and k such that $|f(x)| \le C|g(x)|$ if $x > k$.

Big-O Notation

Example 1. Show that $f(x) = x^2 + 2x + 1$ is $O(x^2)$.

Solution:

Since

$$0 \le x^2 + 2x + 1 \le x^2 + 2x^2 + x^2 = 4x^2$$

whenever $x > 1$, it follows that $f(x)$ is $O(x^2)$, take $C = 4$ and $k = 1$.

- Observe that in the relationship $f(x)$ is $O(x^2)$, x^2 can be replaced by any function with larger value than x^2

Big-O Notation

- The fact that $f(x)$ is $O(g(x))$ is sometimes written $f(x) = O(g(x))$.
- However, the equals sign in this notation *does not* represent a genuine equality.
- This notation tells us that an inequality holds relating the values of the function f and g for sufficiently large numbers in the domains of these functions.

Big-O Notation

- When $f(x)$ is $O(g(x))$, and $h(x)$ is a function that has larger absolute values than $g(x)$ does for sufficiently large values of x, it follows that $f(x)$ is $O(h(x))$.
- The function $g(x)$ in the relationship $f(x)$ is $O(g(x))$ can be replaced by a function with larger absolute values. To see this, note that if

  $$|f(x)| \leq C|g(x)| \qquad \text{if } x > k,$$

  and if $|h(x)| > |g(x)|$ for all $x > k$, then

  $$|f(x)| \leq C|h(x)| \qquad \text{if } x > k.$$

  Hence, $f(x)$ is $O(h(x))$.

Big-O Notation

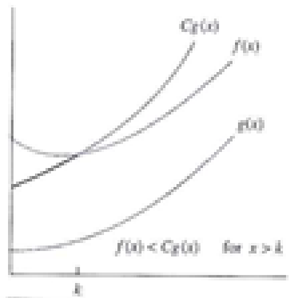

FIGURE 2 The Function $f(x)$ Is $O(g(x))$.

Big-O Notation

Example 2. Show that $7x^2$ is $O(x^3)$.

Solution:

The inequality $7x^2 < x^3$ holds whenever $x > 7$. Hence, $7x^2$ *is* $O(x^3)$, taking $C = 1$ and $k = 7$ in the definition of big-O notation.

Big-O Notation

- Polynomials can often be used to estimate the growth of functions.
- Theorem 1 shows that the leading term of a polynomial dominates its growth by asserting that a polynomial of degree n or less is $O(x^n)$.

THEOREM 1. Let $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$,
where $a_0, a_1, \ldots a_{n-1}, a_n$ are real numbers.
Then $f(x)$ is $O(x^n)$.

Big-O Notation

Example 3. How can big-O notation be used to estimate the sum of the first n positive integers?

Solution: Since each of the integers in the sum of the first n positive integers does not exceed n, it follows that

$$1 + 2 + \ldots + n \leq n + n + \ldots + n = n^2$$

From this inequality it follows that $1 + 2 + \ldots + n$ is $O(n^2)$, taking $C = 1$ and $k = 1$ in the definition of big-O notation.

Big-O Notation

- Many algorithms are made up of two or more separate subprocedures.
- The number of steps in such an algorithm is the sum of the number of steps used by these subprocedures.
- It is necessary to find big-O estimates for the number of steps used by each procedure and then combine these estimates.

Big-O Notation

THEOREM 2. Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1 + f_2)(x)$ is $O(max(g_1(x), g_2(x)))$.

THEOREM 3. Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1\ f_2)(x)$ is $O((g_1(x)\, g_2(x)))$.

Big-O Notation

- Big-O notation is used to estimate the number of operations needed to solve a problem using a specified procedure or algorithm.
- The functions used in these estimates often include the following:

  1, log n, n, n log n, n^2, 2^n, $n!$

Big-O Notation

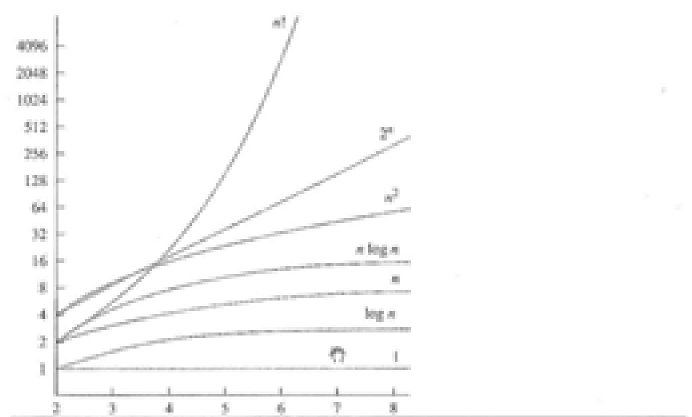

FIGURE 3 A Display of the Growth of Functions Commonly Used in Big-O Estimates.

Algorithm

- An algorithm is a finite set of precise instructions for performing a computation or for solving a problem

Example 4. Describe an algorithm for finding the maximum (largest) value in a finite sequence of integers.

Solution:

1. Set the temporary maximum equal to the first integer in the sequence.
2. Compare the next integer in the sequence to the temporary maximum, and if it is larger than the temporary maximum, set the temporary maximum equal to this integer.
3. Repeat the previous step if there are more integers in the sequence.
4. Stop when there are no integers left in the sequence. The temporary maximum at this point is the largest integer in the sequence.

Algorithm

- An algorithm can also be described using a computer language.
- But the description of the algorithm that is complicated and difficult to understand.
- Furthermore, since many programming languages are in common use, it would be undesirable to choose one particular language.

Algorithm

- Instead of using a particular computer language to specify algorithms, a form of pseudocode will be used.
- Pseudocode provides an intermediate step between an English language description of and algorithm and an implementation of this algorithm in a programming language.

Algorithm

- A pseudocode description of the algorithm for finding the maximum element in a finite sequence follows.

ALGORITHM 1. Finding the Maximum Element in a Finite Sequence.

method *findmax*($a_1, a_2, \ldots\ a_{n-1}, a_n$: integers)

$max = a_1$;

for $i = 1$ to n

    if $max < a_i$ **then** $max = a_i$

Algorithm

- There are several properties that algorithm generally share.
  - Input
  - Output
  - Definiteness
  - Correctness
  - Finiteness
  - Effectiveness
  - Generality

Searching Algorithm

- The problem of locating an element in an ordered list occurs in many contexts.
- For instance, a program that checks the spelling of words in dictionary.
- The general searching problem can be described as follows:
  - Locate an element x in a list of distinct elements $a_1, a_2, \ldots a_{n-1}, a_n$, or determine that it is not in the list.
  - The solution to this search problem is the location of the term in the list that equals x.

Searching Algorithm: Linear Search

- The linear search algorithm begins by comparing x and a_1.
- When $x = a_1$, the solution is the location of a_1.
- When $x \neq a_1$, compare x with a_2, if $x = a_2$, the solution is the location of a_2. Continue this process until a match found.

Searching Algorithm: Linear Search

ALGORITHM 2. The Linear Search Algorithm
METHOD linear_search(x: integer, a_1, a_2, ... a_{n-1}, a_n: distinct integers)

```
    SET i = 1
    WHILE i ≤ n and x ≠ a_i
        SET i = i + 1
    ENDWHILE
    IF i ≤ n
        location = i
    ELSE
        location = 0
    ENDIF
```

Searching Algorithm: Linear Search

- Note that ALGORITHM 1 and ALGORITHM 2 use different style of pseudocode.
- Because of pseudocode is no standard. It vary from company to company or developer to developer.

Exercise

- Page 90:
  - Problem 1 a,b,c
  - Problem 2 a,b,d,e
  - Problem 3, 4, 5, 6, 10, 11, 15, 16