Chapter 3: Algorithms

Trần Hòa Phú

Ngày 3 tháng 2 năm 2023

Objectives

- Algorithms
- The Growth of functions
- Complexity of Algorithms

Algorithms

Definition

An **algorithm** is a **finite sequence of precise instructions** for performing a computation or for solving a problem.

ALGORITHM 1

Finding the Maximum Element in a Finite Sequence

procedure max($a_1, \ldots, a_n$: distinct integers) day huu han

max $:= a_1$

for $i := 2$ to n

  if $max < a_i$ then $max := a_i$

return max $\{max$ is the largest element $\}$.

ALGORITHM 2
The Linear Search Algorithm

procedure linearsearch $(x$: integer, $a_1, \ldots, a_n$: distinct integers)

$i := 1$

While ($i \leq n$ and $x \neq a_i$)

  $i := i + 1$

if $i \leq n$ then location $i := i$

else location:=0

return location

ALGORITHM 3

The Binary Search Algorithm

Example To search 19 in the list

1 2 3 5 6 8 10 12 13 15 16 18 19 20 22

Split the list

1 2 3 5 6 8 10   12 13 15 16 18 19 20 22

$19 > 10$?, Yes. split the right list

12 13 15 16   18 19 20 22

$19 > 16$?, Yes. split the right list

18 19   20 22

$19 > 19$?, No. split the left list

18   19

$19 > 18$? No. A comparision is made

procedure binary search (x : integer, $a_1, \ldots, a_n$: increasing integers)

$i := 1\{i$ is left endpoint of search interval $\}$

$j := n\{j$ is right endpoint of search interval $\}$

While $i < j$

  $m := [(i + j)/2]$

  if $x > a_m$ then $i := m + 1$

  else $j := m$

if $x = a_i$ then location $:= i$

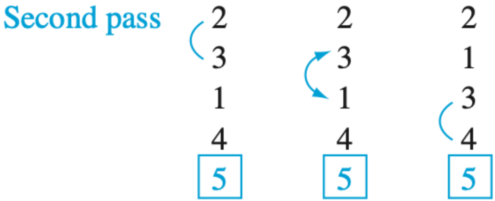
else location $:= 0$

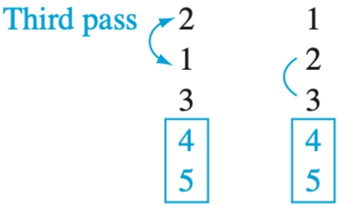
return location (location is the subscript i of the term a_i equal to x, or 0 if x is not found)

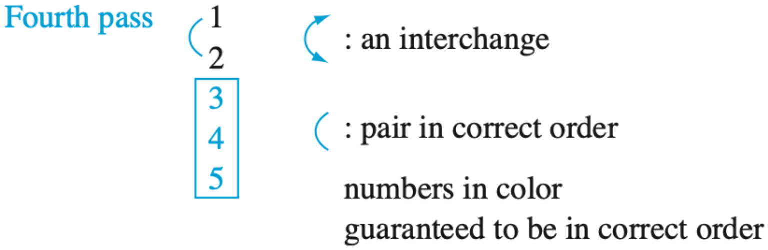
ALGORITHM 4
Bubble Sort

Example Use the bubble sort to put 3,2,4,1,5 into increasing order.

First pass				
	3	2	2	2
	2	3	3	3
	4	4	4	1
	1	1	1	4
	5	5	5	5


Second pass
2
3
1
4
5
2
3
1
4
5
2
1
3
4
5


Third pass
2
1
3
4
5
1
2
3
4
5


Fourth pass
1
2
3
4
5
: an interchange
: pair in correct order
numbers in color
guaranteed to be in correct order

procedure bubblesort($a_1, \ldots, a_n$: real numbers with $n \geq 2$)

for $i = 1$ to $n - 1$

  for $j := 1$ to $n - i$

    if $a_j > a_{j+1}$ then interchange a_j and a_{j+1}

$\{a_1, a_2, \ldots, a_n$ is in increasing order $\}$

Exercise

Procedure **Bubblesort**($a_1, a_2, \ldots, a_n$: integer)

```
for i = 1 to (n-1) do
    for j = 1 to (n-i) do
       if a_j > a_{j+1} then
         swap(a_j, a_{j+1})
```

If input = 3, 2, 4, 7, 1, 6, 5, find the order of the elements in the list after the first pass (i = 1).

A. 2, 3, 4, 1, 5, 6, 7

B. None of the other choices is correct

C. 2, 3, 1, 4, 5, 6, 7

D. 2, 3, 4, 1, 6, 5, 7

E. 2, 3, 1, 4, 6, 5, 7

ALGORITHM 5
Insertion Sort

Example

Use the insertion sort to put the elements of the list $3, 2, 4, 1, 5$ in increasing order.

3 2 4 1 5

Step 1: Compare $2 > 3$? No.

2 3 4 1 5

Compare $4 > 2$? Yes. Compare $4 > 3$? Yes.

2 3 4 1 5

Compare 1 với $2, 3, 4$? $1 > 2$ No.

1 2 3 4 5

Compare 5 với $1, 2, 3, 4$? Yes

1 2 3 4 5

procedure insertion sort ($a_1, \ldots, a_n$: real numbers with $n \geq 2$)

for $j := 2$ to n

  $i := 1$

  while $a_j > a_i$

    $i := i + 1$

  $m := a_j$

  for $k := 0$ to $j - i - 1$

    $a_{j-k} := a_{j-k-1}$

  $a_i := m$

$\{a_1, \ldots, a_n$ is in increasing order $\}$

ALGORITHM 6
Greedy Algorithm

Optimization problems :

Finding a route between two cities with smallest total mileage

Determining a way to encode messages using the fewest bits possible

Greedy Algorithm: select the best choice at each step instead of considering of all sequences of steps

Ví dụ: Cho 5 loại tiền giấy có mệnh giá lần lượt là 1 ngàn, 2 ngàn, 5 ngàn, 10 ngàn.

Hỏi số tờ tiền ít nhất cho tương ứng với 28 ngàn là bao nhiêu?

Thuận toán Tham Lam:

- Đổi 1 tờ 10 ngàn. Do đó còn 18 ngàn.
- Đổi 1 tờ 10 ngàn. Còn 8 ngàn.
- Đổi 1 tờ 5 ngàn. Còn 3 ngàn.
- Đổi 1 tờ 2 ngàn. Còn 1 ngàn.
- Đổi 1 tờ 1 ngàn.

Câu hỏi: Tối ưu mỗi bước có tối ưu toàn cục?

ALGORITHM 6 Greedy Change-Making Algorithm.

procedure *change*($c_1, c_2, \ldots, c_r$: values of denominations of coins, where
  $c_1 > c_2 > \cdots > c_r$; n: a positive integer)
for $i := 1$ **to** r
  $d_i := 0$ {d_i counts the coins of denomination c_i used}
  while $n \geq c_i$
    $d_i := d_i + 1$ {add a coin of denomination c_i}
    $n := n - c_i$
{d_i is the number of coins of denomination c_i in the change for $i = 1, 2, \ldots, r$}

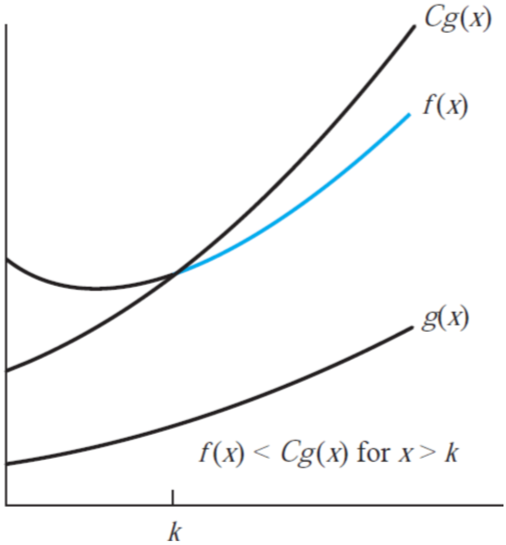
The Growth of Functions

Big-O Notation

Definition

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

$$|f(x)| \leq C|g(x)| \quad \forall x > k$$


$Cg(x)$
$f(x)$
$g(x)$
$f(x) < Cg(x)$ for $x > k$
k

Example Show that $5x + 3$ is $O(x)$.

Chứng minh.

We have

$$|5x + 3| \leq 5|x| + 3 \leq 5|x| + 3|x| = 8|x| \quad \forall x > 1 \tag{1}$$

$\square$

Example Show that $5x + 3$ is $O(x^2)$.

Chứng minh.

We have

$$|5x+3| = |5||x| + 3 \leq 5|x||x| + 3|x|^2 = 8|x^2| \quad \forall x \geq 1 \tag{2}$$

$\square$

Theorem

Let $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$ *where* $a_0, \ldots, a_n \in \mathbb{R}$.
Then $f(x)$ *is* $O(x^n)$

Example 1 $8x^{10} + 5x^8 + x^7 + 1$ is $O(x^{10})$

Example 2 $\dfrac{1}{100}x^8 - 6^4 + 2x^2$ is $O(x^8)$

Example Show that $1 + 2 + ... + n$ is $O(n^2)$.

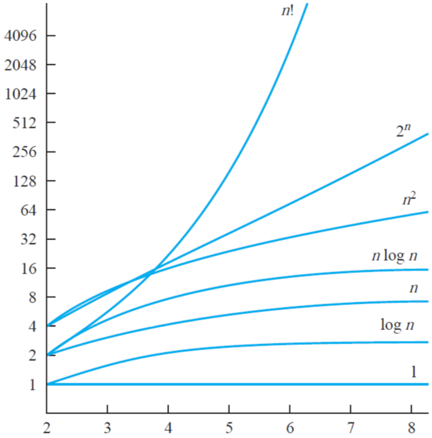
Chứng minh.

We have

$$1 + 2 + 3 + ... + n \leq n + n + n + ... + n = n.n = n^2 \quad \forall n \geq 1 \quad (3)$$

$\square$

$$1 \le \log n \le n \le n \log n \le n^2 \le 2^n \le n! \qquad n \ge 5$$



The Growth of Combinations of Functions

Theorem

Suppose that
$f_1(x)$ *is* $O(g_1(x))$ *and* $f_2(x)$ *is* $O(g_2(x))$. *Then*

$$(f_1 + f_2)(x) \quad \text{is} \quad O(\max(|g_1(x)|, |g_2(x)|)) \tag{4}$$

Example 1 The function $2x + \log x$ is $O(x)$.
Example 2 The function $\log x + 2^x$ is $O(2^x)$
Example 3 The function $x^2 + x \log x$ is $O(x^2)$

Theorem

Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1f_2)(x)$ is $O(g_1(x)g_2(x))$.

Example $(x^5 + 3x + 1)(x^3 - 2)$ is $O(x^8)$

Example $(x + x^3)(x \log x + x^2)$ is $O(x^5)$

Exercise

Determine whether each of these functions is $O(x)$

a) $f(x) = 10$

b) $f(x) = 3x + 7$

c) $f(x) = x^2 + x + 1$

d) $f(x) = 5 \log x$

Exercise

Determine whether each of these functions is $O(x^2)$

a) $f(x) = 17x + 11$

b) $f(x) = x^2 + 1000$

c) $f(x) = x \log x$

d) $f(x) = \dfrac{x^4}{2}$

e) $f(x) = 2^x$

f) $f(x) = \dfrac{x^3 + 2x}{2x + 1}$

Exercise

Determine whether each of these functions is

a. $5\log x$ is $O(x)$

b. $\dfrac{x^3+2x}{2x+1}$ is $O(x^2)$

c. x^2+x^4 is $O(x^3)$

d. 2^x is $O(x^2)$

Exercise

Give a big-O estimate for each of these functions

a. $(n^2 + 8)(n + 1)$

b. $(6n + 4n^5 - 4)(7n^2 - 3)$

c. $1^2 + 2^2 + ... + n^2$

d. $1.2 + 2.3 + ... + (n - 1)n$

Exercise

Let f be a function such that $f(x) = 3x^2 \log x + 8x$.

Find the least positive integer n such that $f(x)$ is $O(x^n)$?

A. 1

B. 2

C. 3

D. 4

Big-Omega

Definition

We say that $f(x)$ is $\Omega(g(x))$ if there are positive constants C and k such that

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

whenever $x > k$.

Theorem

$f(x) = \Omega(g(x)) \Leftrightarrow g(x) = O(f(x))$

Example Show that $3x^2 + x + 5$ is $\Omega(x^2)$?

Chứng minh.

We have

$$|3x^2 + x + 5| = 3x^2 + x + 5 \geq 3x^2 \quad \forall x \geq -5 \tag{5}$$

□

Big-Theta

Definition

We say that $f(x)$ is $\Theta(g(x))$ if $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$. It means there are C_1, C_2, k such that

$$C_1|g(x)| \leq f(x) \leq C_2|g(x)) \quad \forall x > k \tag{6}$$

When $f(x)$ is $\Theta(g(x))$, we say that f is big-Theta of $g(x)$, that $f(x)$ is of order $g(x)$, and that $f(x)$ and $g(x)$ are of the same order.

Theorem

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

$$f(x) \text{ is } \Theta(x^n)$$

Example $x^5 - 10x + 42$ is $\Theta(x^5)$

Example $\frac{1}{2}x^8 - 20x^3$ is $\Theta(x^8)$

Complexity of Algorithms

How can the efficiency of an algorithm be analyzed?

- Time Complexity: the number of operations used by the algorithm.
- Space Complexity: amount of computer memory required to implement the algorithm
- Computational Complexity = Time Complexity + Space Complexity

Describe the time complexity of the algorithm for finding the largest element in a set:

Procedure max ($a_1, a_2, \ldots, a_n$: integers)

max:= a_1

for i:=2 to n

If max < a_i **then** max:= a_i

Time Complexity is θ(n)

i	Number of comparisons
2	2
3	2
…	2
n	2
n+1	1, max< a_i is omitted

2(n-1) +1 = 2n-1 comparisons

Describe the average-case time complexity of the linear-search algorithm :

Procedure **linear search** (x: integer, $a_1,a_2,\ldots,a_n$:distinct integers)

i:=1

while ($i \le n$ **and** $x \ne a_i$) i:=i+1

if $i \le n$ **then** location:= i

else location:=0

i	Number of comparisons done
1	2
2	4
…	
n	2n
n+1	1, $x \ne a_i$ is omitted

Avg-Complexity= [(2+4+6+…+2n)]/n +1 +1

= [2(1+2+3+…+n) /n+2

= [2n(n+1)/2]/n + 2

=[n(n+1)]/n + 2

= n+1 + 2 = n+3

= θ(n)

See demonstrations about the worst-case complexity: Examples 5,6 pages 195, 196

Worst-Case Complexity: the largest number of operations needed to solve the given problem using this algorithm on input of specified size.

Average-Case Complexity: the average number of operations used to solve the problem over all possible inputs of a given size is found in this type of analysis.

TABLE 1 **Commonly Used Terminology for the Complexity of Algorithms.**

Complexity	*Terminology*
$\Theta(1)$	Constant complexity
$\Theta(\log n)$	Logarithmic complexity
$\Theta(n)$	Linear complexity
$\Theta(n \log n)$	Linearithmic complexity
$\Theta(n^b)$	Polynomial complexity
$\Theta(b^n)$, where $b > 1$	Exponential complexity
$\Theta(n!)$	Factorial complexity

Exercise

Consider the linear search algorithm

procedure linearsearch (x : integer, $a_1, \ldots, a_n$: distinct integers)
$i := 1$
While ($i \leq n$ and $x \neq a_i$)
    $i := i + 1$
if $i \leq n$ then location $i := i$
else location:=0
return location

Given the sequence 3 1 5 7 4 6. How many comparisions for searching $x = 7$?

Exercise procedure binary search (x : integer, $a_1, \ldots, a_n$: increasing integers)

$i := 1\{i$ is left endpoint of search interval $\}$

$j := n\{j$ is right endpoint of search interval $\}$

While $i < j$

  $m := [(i + j)/2]$

  if $x > a_m$ then $i := m + 1$

  else $j := m$

if $x = a_i$ then location $:= i$

else location $:= 0$

return location

How many comparisons that are used if we use this algorithm to search for x=3 in the list $[1, 3, 4, 5, 6, 8, 9, 12, 17, 20, 26]$?

Exercise

Give the best big-O complexity of the following algorithm.

procedure giaithuat($a_1, .., a_n$: integers)

count:$= 0$

for i$= 1$ to n do

    if $a_i > 0$ then count $:=$ count $+ 1$

print (count)

Exercise

2. Consider the algorithm:

```
procedure GT(n : positive integer)
F:=1
for i:= 1 to n do
    F: = F * i
Print(F)
```

Give the best big-O complexity for the algorithm above.

Exercise

3. Consider the algorithm:

```
procedure max(a ,a ,...,a : reals )
max:=a
for i:=2 to n
if max<a then max:=a
```

Give the best big-O complexity for the algorithm above.

Exercise

Give the best big-O complexity of the following algorithm

procedure giaithuat($a_1, a_2, ..., a_n$: integers)

sum =0;

for $i = 1$ to n do

  for $j = 1$ to n do

    if $a_i > a_j$ then sum = sum+1,

print(sum)

A. $O(n)$ B. $O(n^2)$ C. $O(\log n)$ D. $O(n \log n)$