Tài liệu tham khảo: Bài giảng SMA 5503 Introduction to Algorithms. 2001-5
Erik D. Demaine and Charles E. Leiserson. http://ocw.mit.edu

Bài 13: Các thuật toán sắp xếp

Giảng viên: Hoàng Thị Điệp
Khoa Công nghệ Thông tin – Đại học Công Nghệ

Nội dung chính

1. Bài toán sắp xếp
2. Sắp xếp xen vào
3. Sắp xếp trộn
4. Sắp xếp nhanh
5. Sắp xếp sử dụng cây thứ tự bộ phận
6. Sắp xếp đếm
7. Sắp xếp cơ số

diepht@vnu

Bài toán sắp xếp

- Lí do:
  - Một trong những bài toán được nghiên cứu lâu đời nhất trong CNTT
  - Chứa nhiều kĩ thuật về thuật toán

- Input: dãy số $<a_1, a_2, ..., a_n>$
- Output: 1 hoán vị của input $<a_1', a_2', ..., a_n'>$ thỏa mãn $a_1' <= a_2' <= ... <= a_n'$
- Ý nghĩa?

  - Bài toán tìm kiếm
  - Bài toán phát hiện phần tử lặp

diepht@vnu

Ví dụ bài toán tìm kiếm

- x = 5
- A = (3, 1, 4, 15, 9, 26, 53, 58, 97, 93, 23, 8, 46, 26, 4, 33, 8, 3, 2)
- B = (1, 2, 3, 3, 4, 4, 8, 8, 9, 15, 23, 26, 26, 33, 46, 53, 58, 93, 97)

❑ x có trong A?

❑ x có trong B?

Ví dụ bài toán phát hiện phần tử lặp

- A = (3, 1, 4, 15, 9, 26, 53, 58, 97, 93, 23, 8, 46, 26, 4, 33, 8, 3, 2)
- B = (1, 2, 3, 3, 4, 4, 8, 8, 9, 15, 23, 26, 26, 33, 46, 53, 58, 93, 97)

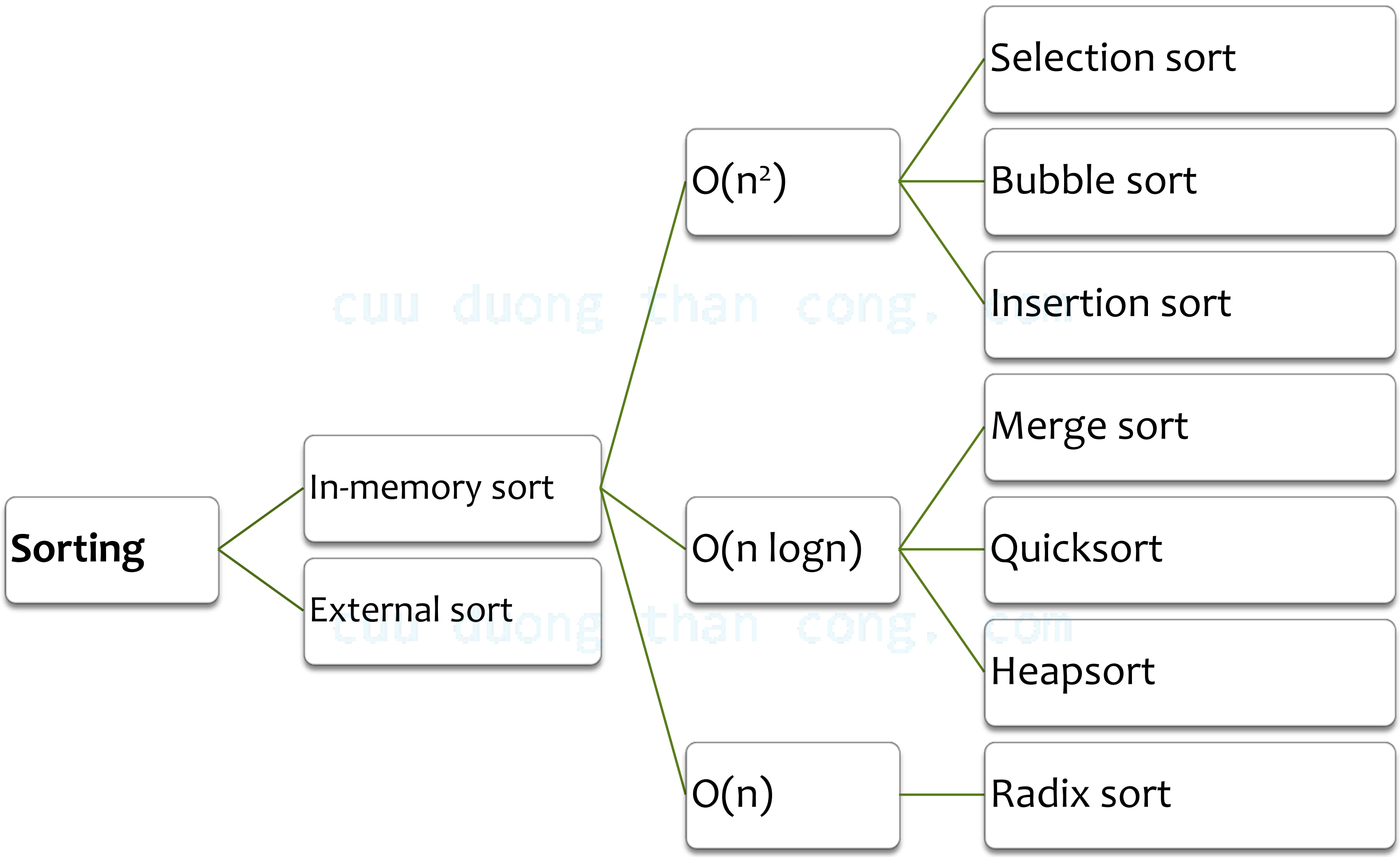
❑ Các giá trị xuất hiện hơn 1 lần trong A?

❑ Các giá trị xuất hiện hơn 1 lần trong B?

Tổng quan

- **Sắp xếp**
  - Sắp xếp trong
    - $O(n^2)$
      - TTSX lựa chọn
      - TTSX nổi bọt
      - TTSX xen vào
    - $O(n \log n)$
      - TTSX trộn
      - TTSX nhanh
      - TTSX sử dụng heap
    - $O(n)$
      - TTSX theo cơ số
  - Sắp xếp ngoài

Tổng quan


Sorting
In-memory sort
External sort
O(n²)
O(n logn)
O(n)
Selection sort
Bubble sort
Insertion sort
Merge sort
Quicksort
Heapsort
Radix sort

Với mỗi thuật toán sắp xếp

- Lịch sử ra đời
- Ý tưởng
- Giả mã
- Ví dụ
- Phân tích độ phức tạp thời gian
- Vận dụng thế nào?
- Cài đặt bằng ngôn ngữ C++
  - có trong STL không?
- Tính ổn định (stability)
- Liên hệ với các thuật toán sắp xếp khác

Insertion Sort

Thuật toán sắp xếp xen vào

"pseudocode"

INSERTION-SORT (A, n) ▷ $A[1\,.\,.\,n]$

for $j \leftarrow 2$ **to** n
  do $key \leftarrow A[j]$
    $i \leftarrow j - 1$
    while $i > 0$ and $A[i] > key$
      do $A[i+1] \leftarrow A[i]$
        $i \leftarrow i - 1$
    $A[i+1] = key$

Thuật toán sắp xếp xen vào

"pseudocode"

INSERTION-SORT (A, n) $\triangleright A[1 \,.\,.\, n]$

    for $j \leftarrow 2$ **to** n

        do $key \leftarrow A[j]$

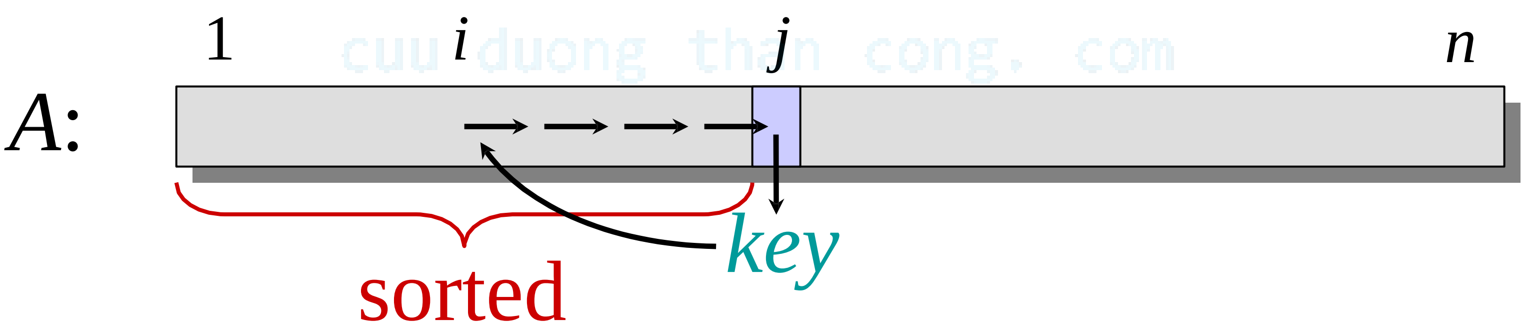
            $i \leftarrow j - 1$

            while $i > 0$ and $A[i] > key$

                do $A[i+1] \leftarrow A[i]$

                    $i \leftarrow i - 1$

            $A[i+1] = key$



Minh họa SX xen vào

8 2 4 9 3 6

Minh họa SX xen vào

8 2 4 9 3 6

Minh họa SX xen vào

8 2 4 9 3 6

2 8 4 9 3 6

Minh họa SX xen vào

8 2 4 9 3 6

2 8 4 9 3 6

cuu duong than cong. com

diepht@vnu

Minh họa SX xen vào

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6

Minh họa SX xen vào

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

diepht@vnu

Minh họa SX xen vào

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6

Minh họa SX xen vào

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6

Minh họa SX xen vào

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6
2	3	4	8	9	6

cuu duong than cong . com

Minh họa SX xen vào

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6
2	3	4	8	9	6

Minh họa SX xen vào

8	2	4	9	3	6	
2	8	4	9	3	6	
2	4	8	9	3	6	
2	4	8	9	3	6	
2	3	4	8	9	6	
2	3	4	6	8	9	*done*

diepht@vnu

Phân tích độ phức tạp

- Thời gian chạy phụ thuộc bản thân input
  - Nếu đã sắp
    - đúng thứ tự?
    - ngược thứ tự?
  - Kích thước dữ liệu vào
- Thời gian chạy xấu nhất?

diepht@vnu

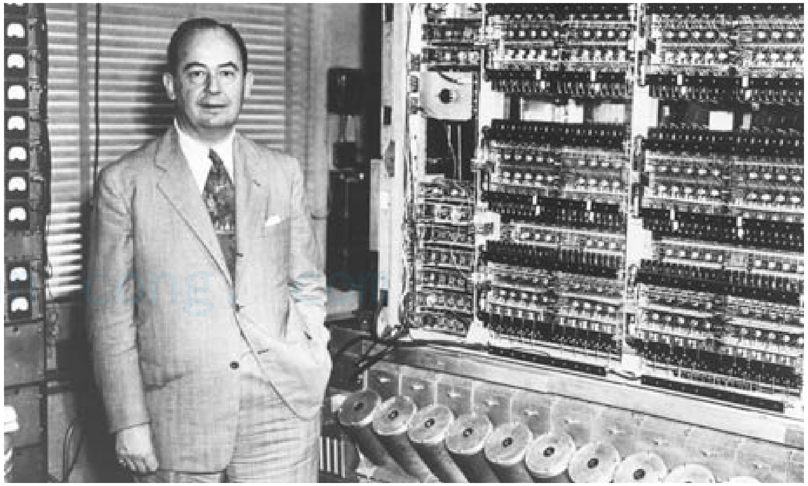
Merge Sort

Thuật toán sắp xếp trộn

MERGE-SORT $A[1 \,..\, n]$

1. If $n = 1$, done.
2. Recursively sort $A[\,1 \,..\, \lceil n/2 \rceil\,]$ and $A[\,\lceil n/2 \rceil + 1 \,..\, n\,]$.
3. "***Merge***" the 2 sorted lists.

Key subroutine: **MERGE**



John von Neumann

diepht@vnu

Trộn 2 mảng tăng

20	12
13	11
7	9
2	1

cuu duong than cong. com

Trộn 2 mảng tăng

20	12
13	11
7	9
2	1

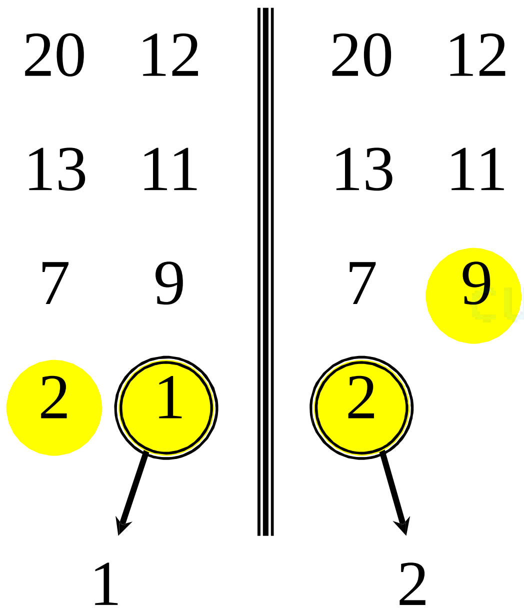
1

Trộn 2 mảng tăng

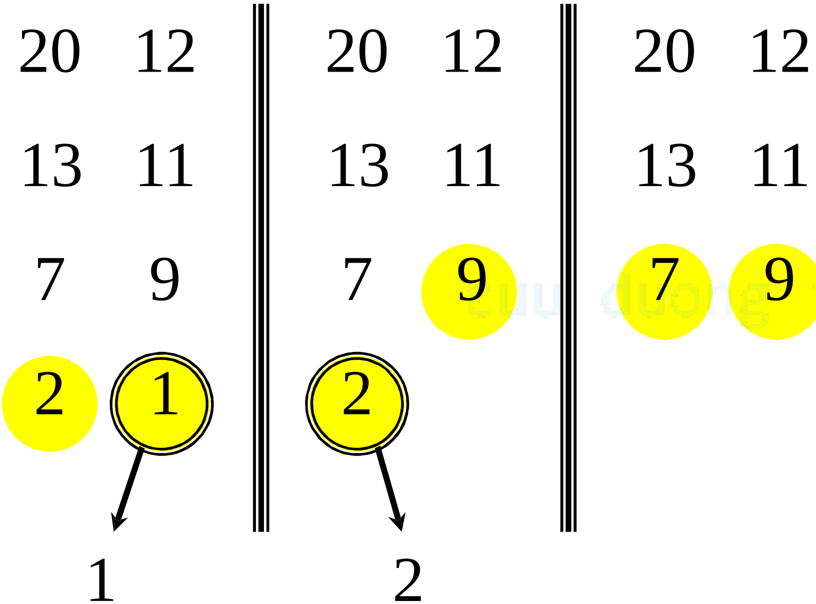
20	12		20	12
13	11		13	11
7	9		7	9
2	1		2	

1

Trộn 2 mảng tăng

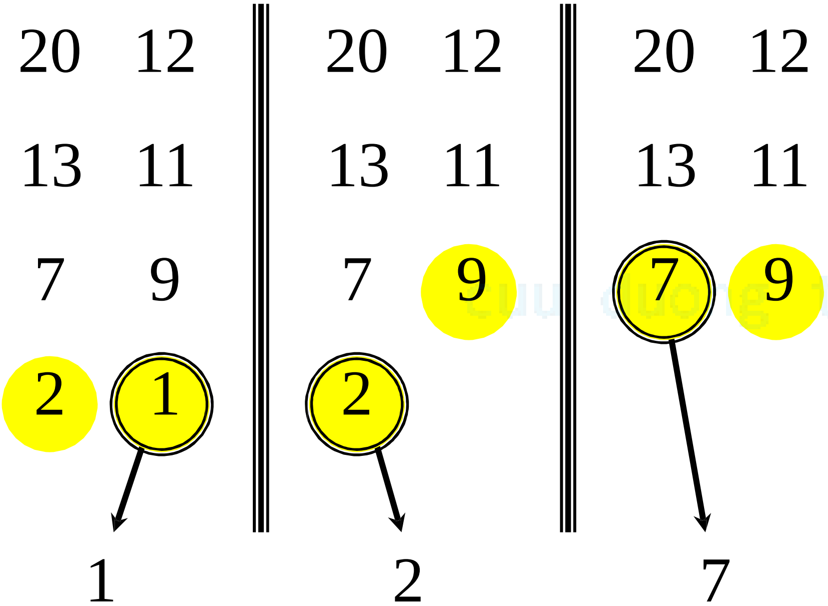


Trộn 2 mảng tăng

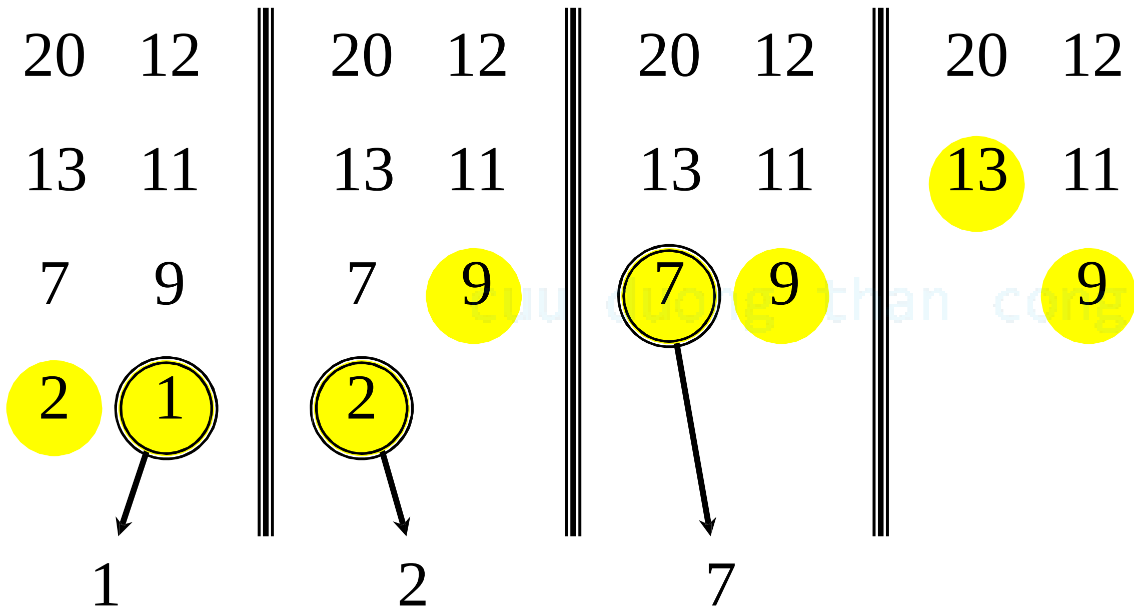


diepht@vnu

Trộn 2 mảng tăng

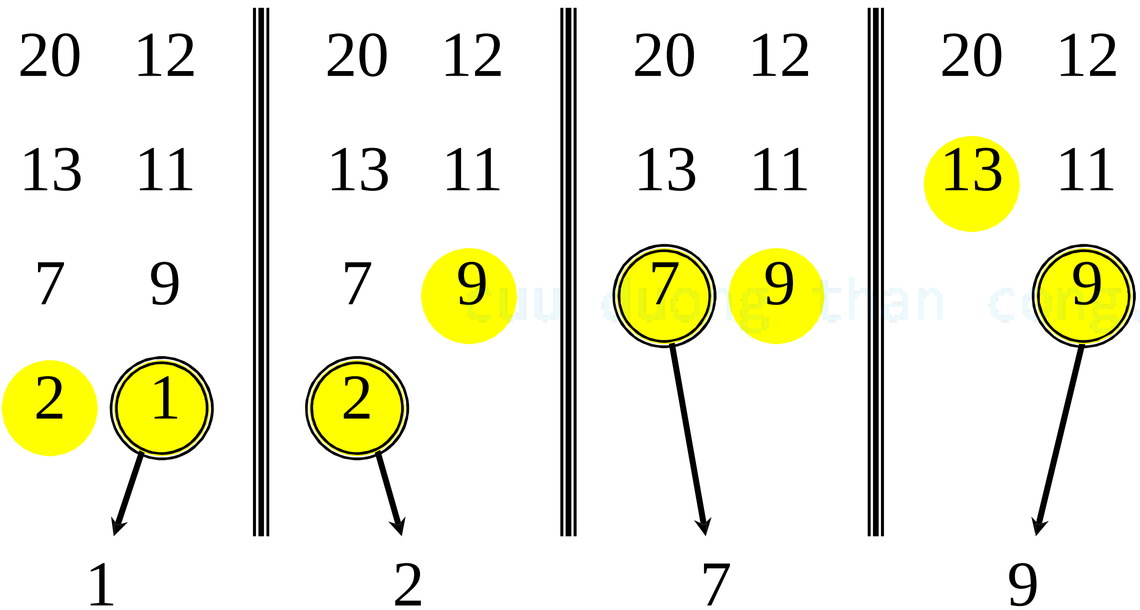


Trộn 2 mảng tăng



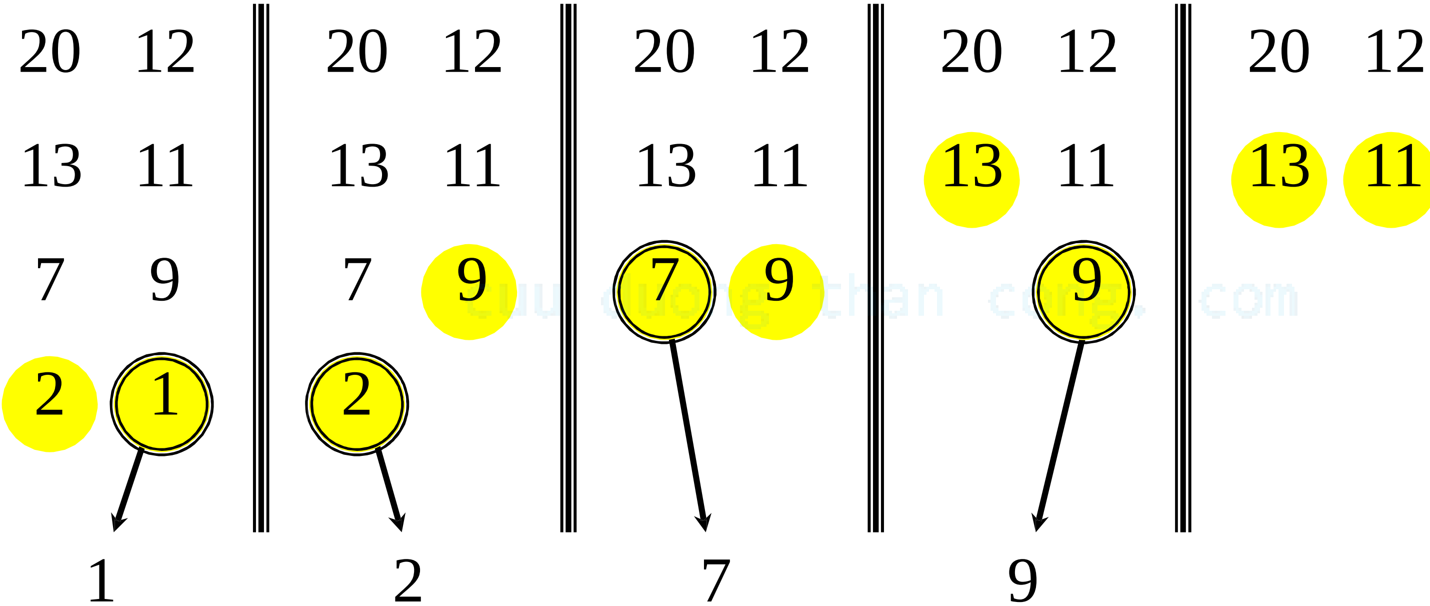
diepht@vnu

Trộn 2 mảng tăng

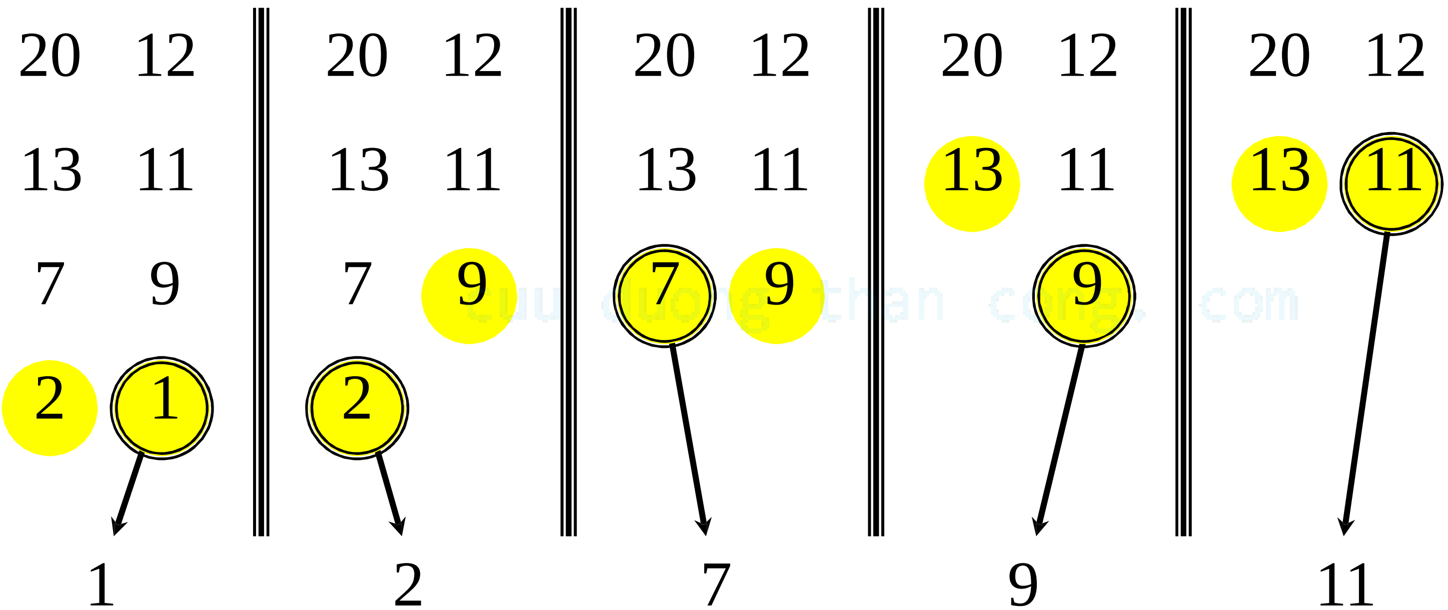


diepht@vnu

Trộn 2 mảng tăng

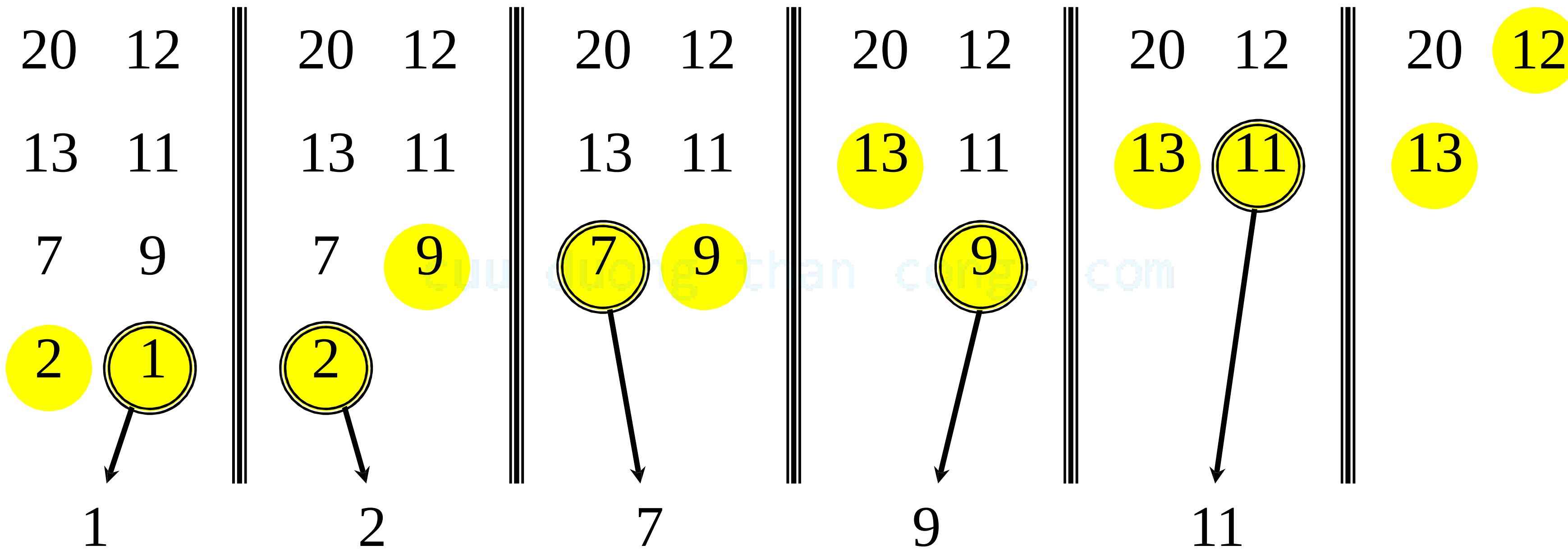


Trộn 2 mảng tăng



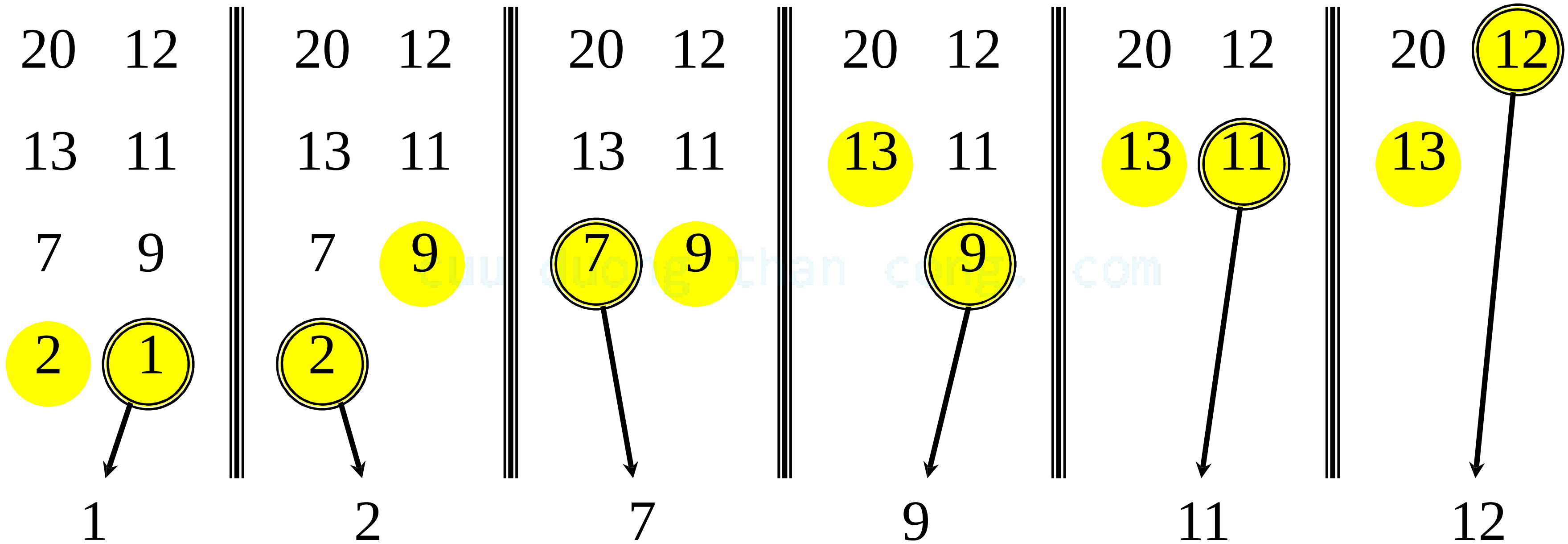
diepht@vnu

Trộn 2 mảng tăng



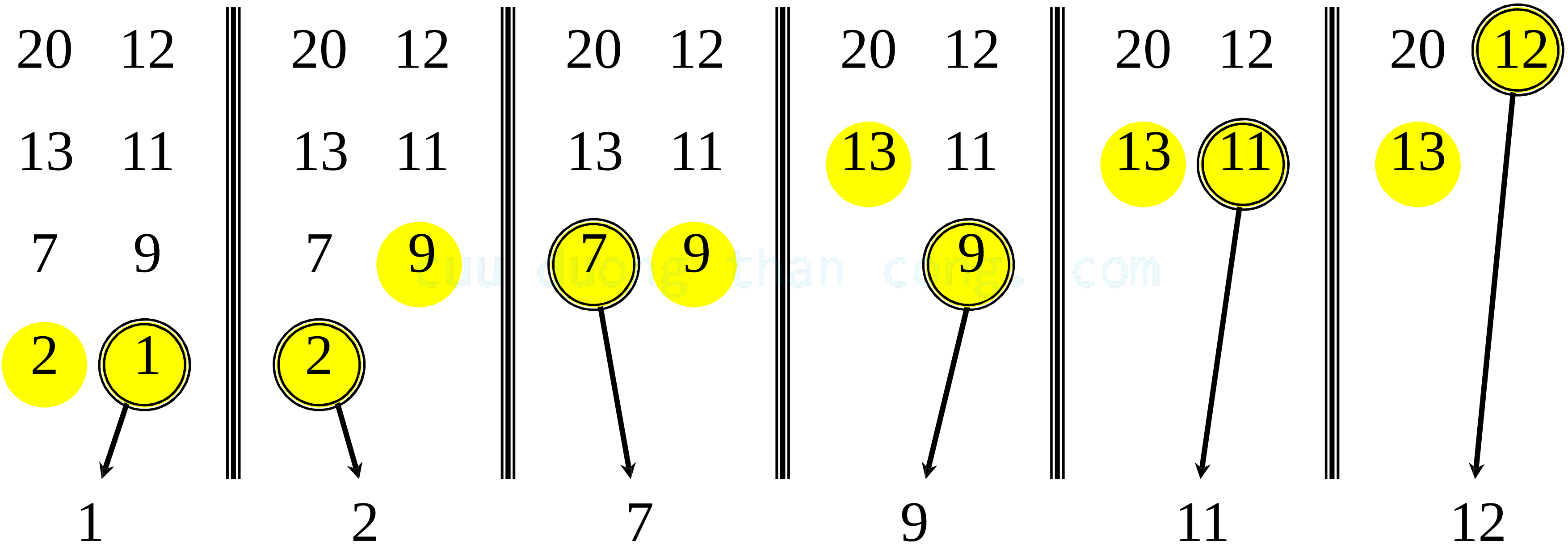
diepht@vnu

Trộn 2 mảng tăng



diepht@vnu

Trộn 2 mảng tăng



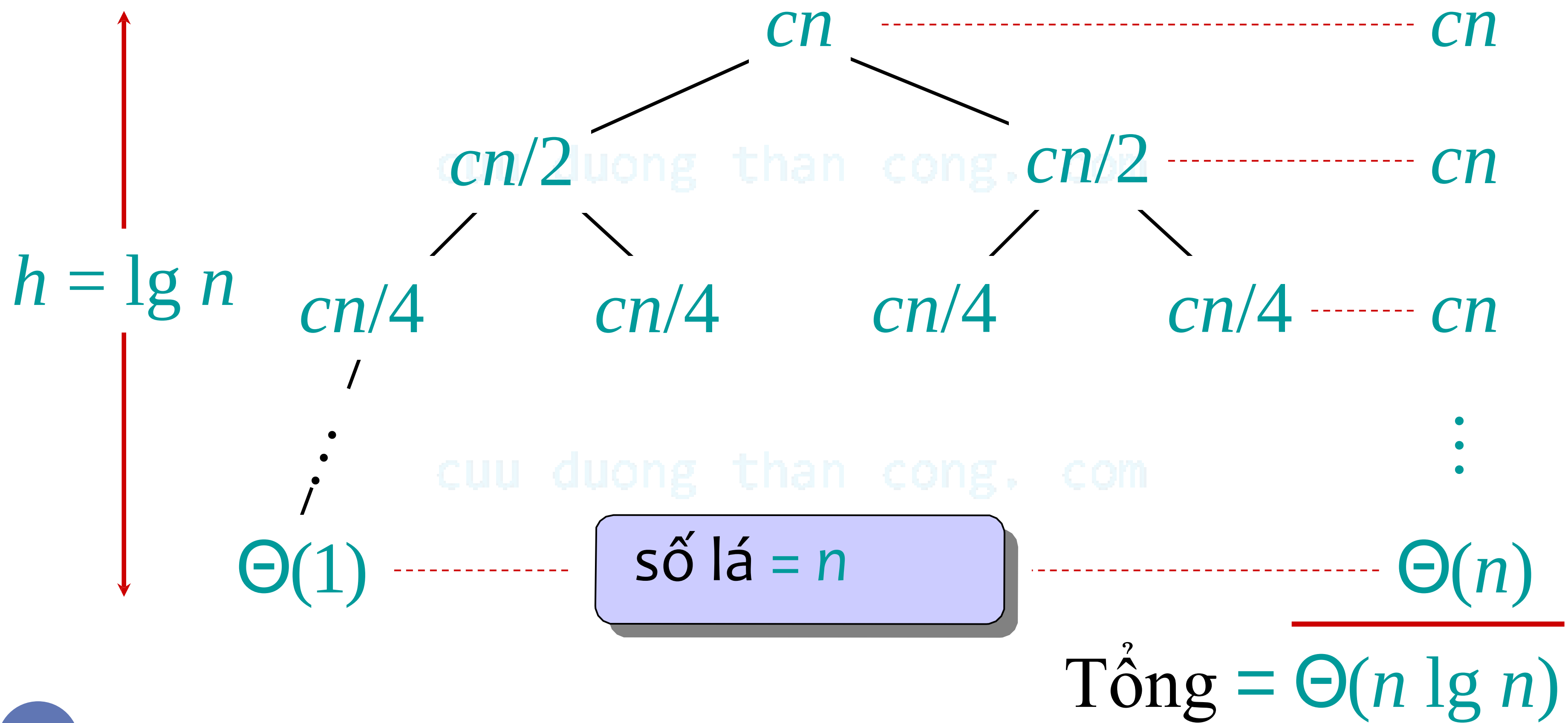
Thời gian trộn là tuyến tính

Phân tích độ phức tạp

$T(n)$	**MERGE-SORT** $A[1 \mathinner{.\,.} n]$
$\Theta(1)$	1. If $n = 1$, done.
$2T(n/2)$	2. Recursively sort $A[\,1 \mathinner{.\,.} \lfloor n/2 \rfloor\,]$ and $A[\,\lfloor n/2 \rfloor + 1 \mathinner{.\,.} n\,]$.
$\Theta(n)$	3. ***“Merge”*** the 2 sorted lists

Cây đệ quy

Giải $T(n) = 2T(n/2) + cn$, với hằng $c > 0$

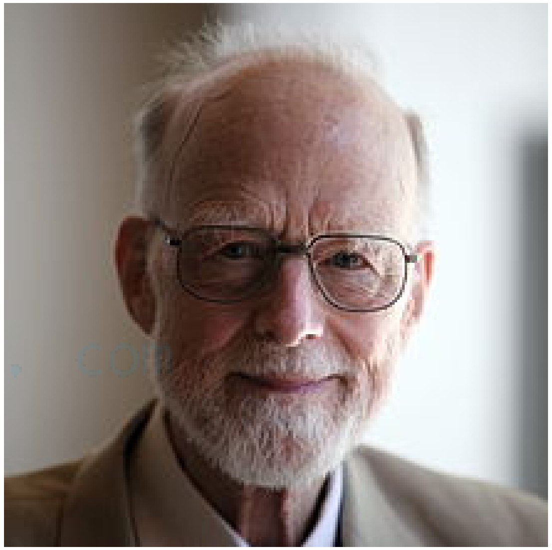


diepht@vnu

Quicksort

Thuật toán sắp xếp nhanh

- Chia để trị
- “in place”
- Hiệu quả trên dữ liệu thực
  - tuning
- Ý tưởng …



Tony Hoare

Mô tả

Sắp xếp nhanh mảng n phần tử

1. ***Chia***: Phân hoạch (chia) mảng cần sắp thành 2 mảng con ở 2 phía của ***chốt*** *x ;* sao cho các phần tử ở mảng con bên trái <= x, còn các phần tử ở mảng con bên phải >= x
2. ***Trị***: Sắp xếp đệ quy các mảng con
3. ***Kết hợp:*** không làm gì.

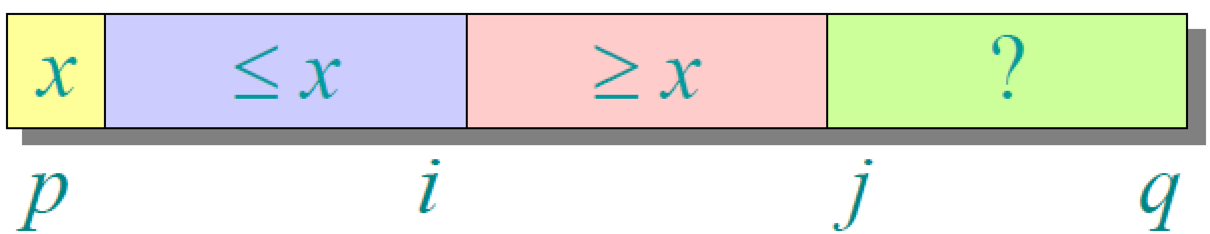
Điểm then chốt: thủ tục phân hoạch chạy trong thời gian tuyến tính.

Giả mã thủ tục phân hoạch

PARTITION(A, p, q) ▷ $A[p\,..\,q]$
    $x \leftarrow A[p]$ ▷ pivot $= A[p]$
    $i \leftarrow p$
    for $j \leftarrow p + 1$ **to** q **do**
        if $A[j] \le x$
            then $i \leftarrow i + 1$
                exchange $A[i] \leftrightarrow A[j]$
    exchange $A[p] \leftrightarrow A[i]$
    return i

Thời gian chạy là O(n)

Duy trì:



Minh họa thuật toán phân hoạch

6	10	13	5	8	3	2	11

i j

Minh họa thuật toán phân hoạch

6	10	13	5	8	3	2	11

i ----> j

Minh họa thuật toán phân hoạch

6	10	13	5	8	3	2	11

i ----> j

Minh họa thuật toán phân hoạch

6	5	13	10	8	3	2	11

----> i j

Minh họa thuật toán phân hoạch

6	5	13	10	8	3	2	11
	i			----> j			

Minh họa thuật toán phân hoạch

6	5	13	10	8	3	2	11

i

----> j

Minh họa thuật toán phân hoạch

6	5	3	10	8	13	2	11

----> i   j

diepht@vnu

Minh họa thuật toán phân hoạch

6	5	3	10	8	13	2	11

i ----> j

Minh họa thuật toán phân hoạch

6	5	3	2	8	13	10	11

----> i   j

Minh họa thuật toán phân hoạch

6	5	3	2	8	13	10	11

i ----> j

Minh họa thuật toán phân hoạch

6	5	3	2	8	13	10	11

i ----> j

Minh họa thuật toán phân hoạch

2	5	3	6	8	13	10	11

i

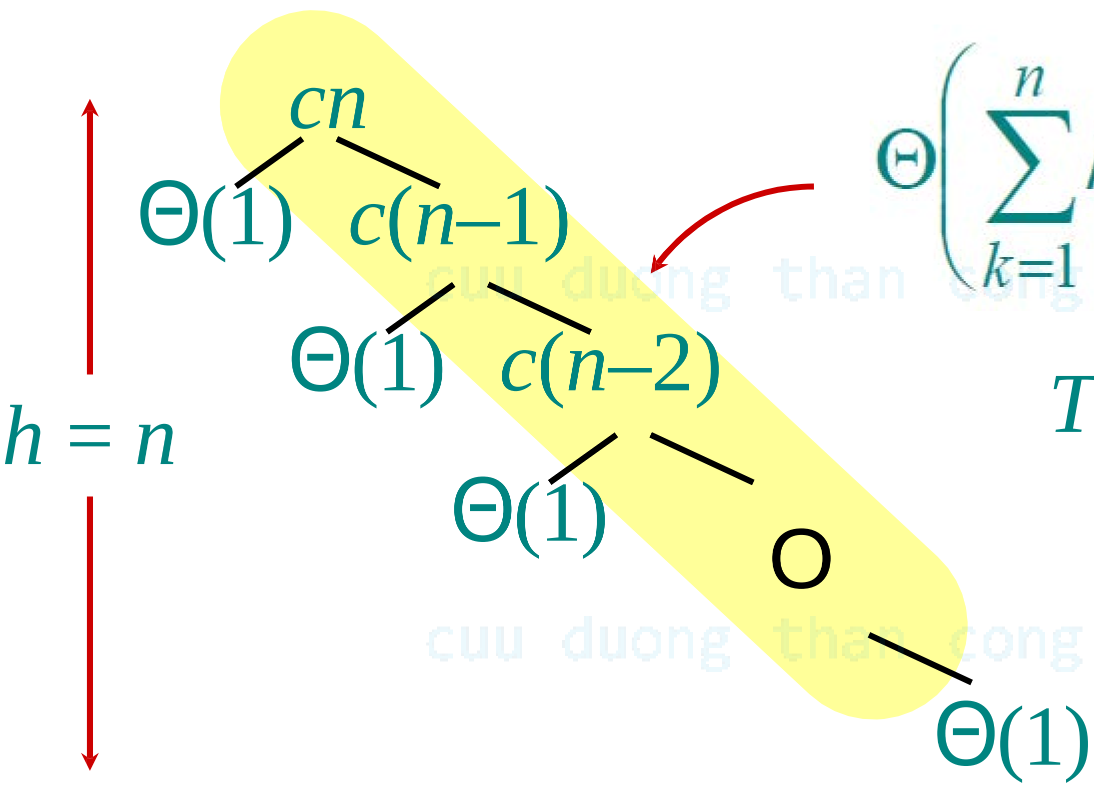
Giả mã thuật toán sắp xếp nhanh

QUICKSORT(A, p, r)
  if $p < r$
    then $q \leftarrow$ PARTITION(A, p, r)
      QUICKSORT$(A, p, q-1)$
      QUICKSORT$(A, q+1, r)$

Lời gọi ban đầu: QUICKSORT$(A, 1, n)$

Cây đệ quy trường hợp xấu nhất

$$T(n) = T(0) + T(n-1) + cn$$



$$\Theta\left(\sum_{k=1}^{n} k\right) = \Theta(n^2)$$

$$T(n) = \Theta(n) + \Theta(n^2)$$
$$= \Theta(n^2)$$

diepht@vnu

Trường hợp tốt nhất?

- PARTITION chia mảng thành 2 nửa bằng nhau

$$T(n) = 2T(n/2) + \Theta(n)$$
$$= \Theta(n \lg n)$$

diepht@vnu

Heapsort

Sắp xếp sử dụng cây thứ tự bộ phận

- Ý tưởng
  - Nếu cần sắp tăng dần, dùng max heap
  - Nếu cần sắp giảm dần, dùng min heap
  - 1: Bố trí lại dữ liệu trong mảng để nó thỏa mãn tính chất của heap.
  - 2: Lặp lại:
    - Đảo chỗ gốc và đỉnh cuối của heap
    - Giảm cỡ của heap đi 1 rồi khôi phục tính chất của heap

Giả mã

Algorithm *heapSort(A, n)*

```
buildHeap(A, n) // tao 1 max-heap tu A
for end <- n-1 to 1 do
    swap(A[0], A[end])
    downheap(A, end)
```

Thuật toán sắp xếp có thể nhanh tới cỡ nào?

Cận dưới của sắp xếp

- Các thuật toán sắp xếp dựa trên so sánh các cặp phần tử
  - comparison sorting, comparison model
  - có thời gian xấu nhất không thể tốt hơn ***O(nlogn)***
- Chứng minh bằng mô hình cây quyết định.
- Tham khảo: Lecture 5

diepht@vnu

Sắp xếp trong thời gian tuyến tính

- Thuật toán sắp xếp đếm
  - counting sort
  - không so sánh các cặp phần tử
- Giả sử dãy số nguyên nằm trong một khoảng nào đó

Counting sort

- ***Input***: $A[1\,..\,n]$, trong đó $A[j] \in \{1, 2, \ldots, k\}$.
- ***Output***: $B[1\,..\,n]$ được sắp.
- ***Mảng nhớ phụ trợ***: $C[1\,..\,k]$.

Counting sort: giả mã

for $i \leftarrow 1$ **to** k
    do $C[i] \leftarrow 0$
for $j \leftarrow 1$ **to** n
    do $C[A[j]] \leftarrow C[A[j]] + 1$ $\triangleright$ $C[i] = |\{\text{key} = i\}|$
for $i \leftarrow 2$ **to** k
    do $C[i] \leftarrow C[i] + C[i-1]$ $\triangleright$ $C[i] = |\{\text{key} \le i\}|$
for $j \leftarrow n$ **downto** 1
    do $B[C[A[j]]] \leftarrow A[j]$
        $C[A[j]] \leftarrow C[A[j]] - 1$

diepht@vnu

Minh họa counting sort

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4

B:

Vòng for thứ nhất

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
0	0	0	0

B:

for $i \leftarrow 1$ **to** k

do $C[i] \leftarrow 0$

Vòng for thứ 2

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
0	0	0	1

B:

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright\ C[i] = |\{\text{key} = i\}|$

Vòng for thứ 2

	1	2	3	4	5
A:	4	1	3	4	3

	1	2	3	4
C:	1	0	0	1

B:					

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ $\quad \triangleright\ C[i] = |\{\text{key} = i\}|$

Vòng for thứ 2

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	1	1

B:

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ $\quad \triangleright\ C[i] = |\{\text{key} = i\}|$

Vòng for thứ 2

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	1	2

B:

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright\ C[i] = |\{\text{key} = i\}|$

Vòng for thứ 2

	1	2	3	4	5
A:	4	1	3	4	3

	1	2	3	4
C:	1	0	2	2

B:					

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright\ C[i] = |\{\text{key} = i\}|$

diepht@vnu

Vòng for thứ 3

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	2	2

B:

C':

1	1	2	2

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ $\quad \triangleright\ C[i] = |\{\text{key} \leq i\}|$

diepht@vnu

Vòng for thứ 3

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	2	2

B:

C':

1	1	3	2

$\textbf{for } i \leftarrow 2 \textbf{ to } k$

$\quad \textbf{do } C[i] \leftarrow C[i] + C[i-1] \qquad \triangleright\ C[i] = |\{\text{key} \le i\}|$

Vòng for thứ 3

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	2	2

B:

C':

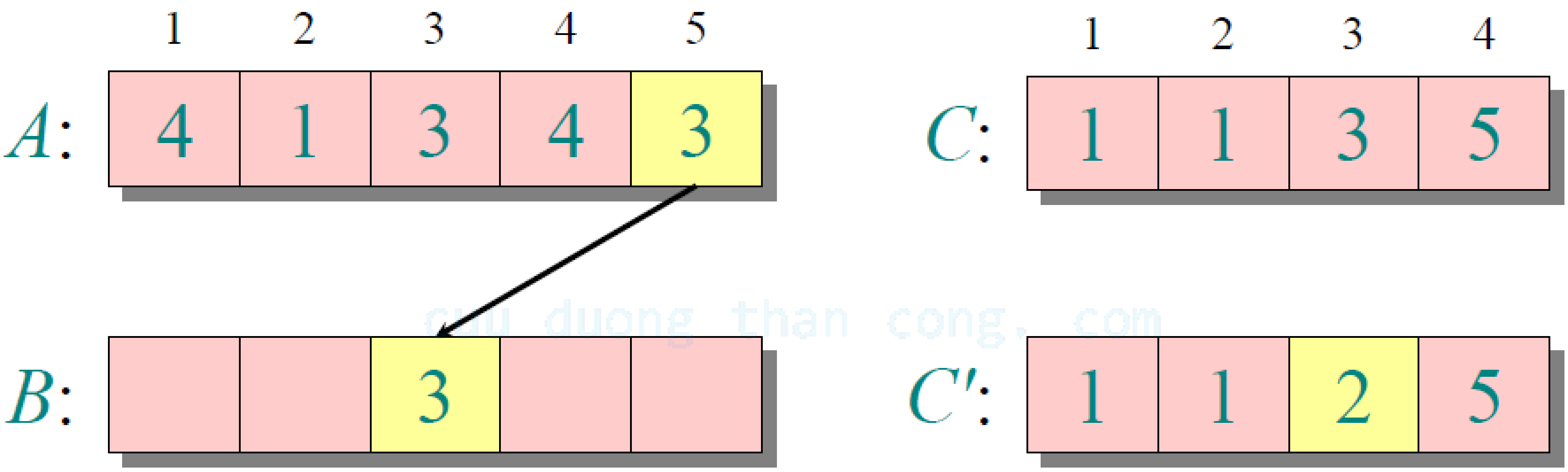
1	1	3	5

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ $\triangleright$ $C[i] = |\{\text{key} \leq i\}|$

diepht@vnu

Vòng for thứ 4



for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$

Vòng for thứ 4

	1	2	3	4	5
A:	4	1	3	4	3
B:			3		4

	1	2	3	4
C:	1	1	2	5
C':	1	1	2	4

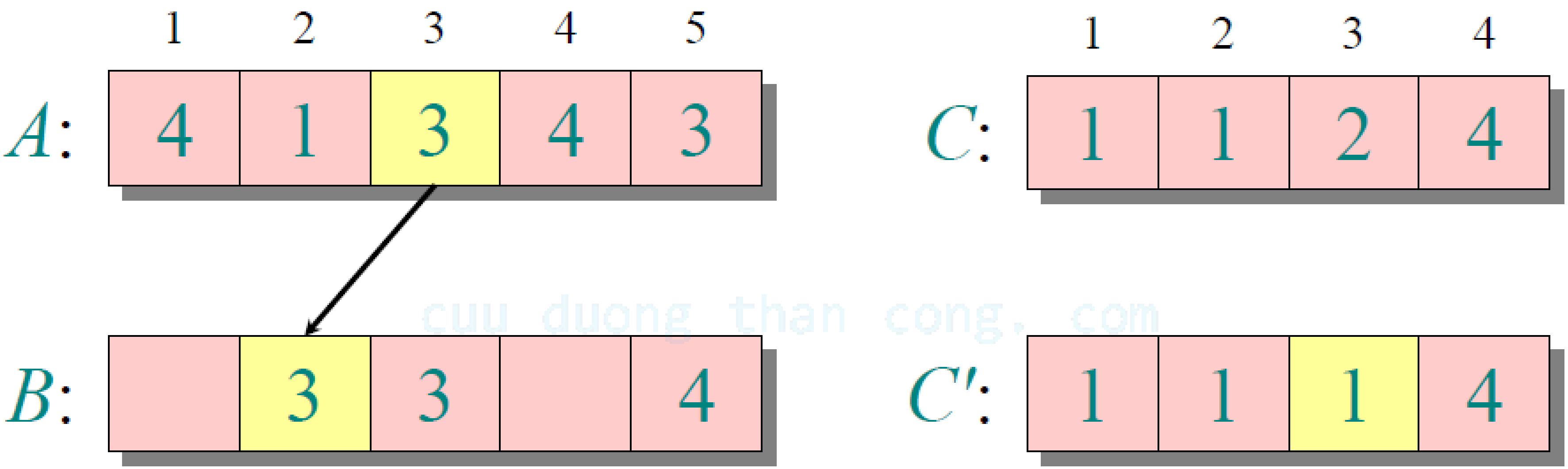
for $j \leftarrow n$ **downto** 1

  do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

    $C[A[j]] \leftarrow C[A[j]] - 1$

diepht@vnu

Vòng for thứ 4

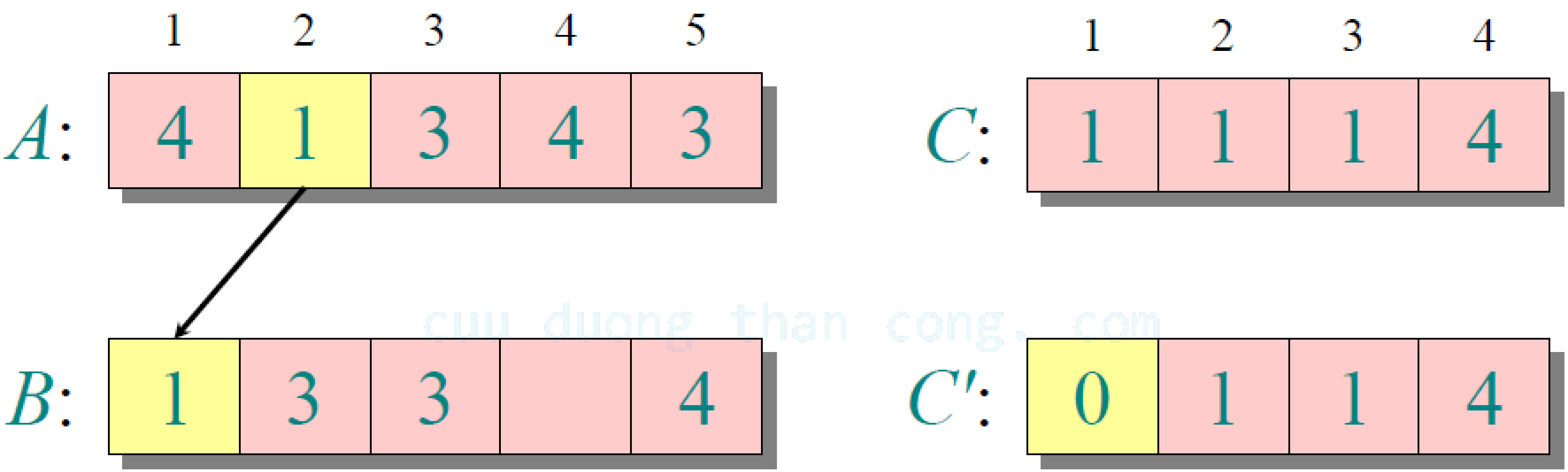


for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$

Vòng for thứ 4



for $j \leftarrow n$ **downto** 1

  do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

    $C[A[j]] \leftarrow C[A[j]] - 1$

Vòng for thứ 4

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
0	1	1	4

B:

1	3	3	4	4

C':

0	1	1	3

for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$

Phân tích độ phức tạp

$\Theta(k)$
for $i \leftarrow 1$ **to** k
  do $C[i] \leftarrow 0$

$\Theta(n)$
for $j \leftarrow 1$ **to** n
  do $C[A[j]] \leftarrow C[A[j]] + 1$

$\Theta(k)$
for $i \leftarrow 2$ **to** k
  do $C[i] \leftarrow C[i] + C[i-1]$

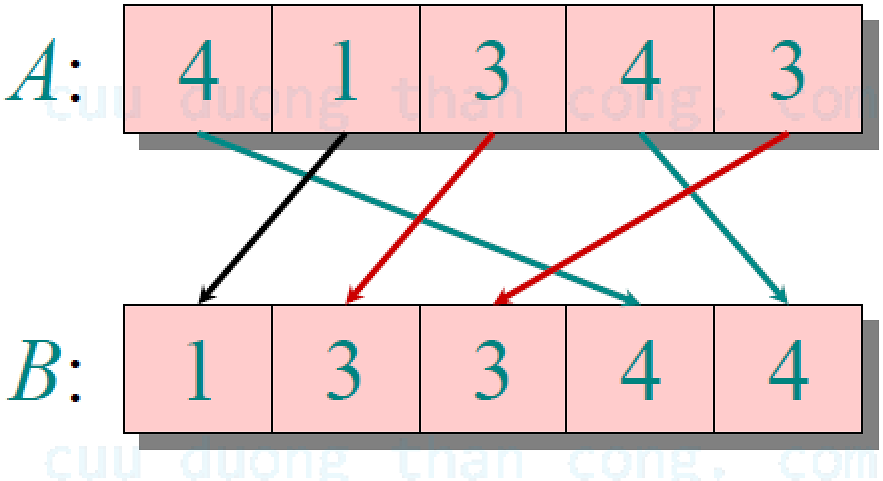
$\Theta(n)$
for $j \leftarrow n$ **downto** 1
  do $B[C[A[j]]] \leftarrow A[j]$
    $C[A[j]] \leftarrow C[A[j]] - 1$

$\Theta(n + k)$

diepht@vnu

Tính ổn định của thuật toán sắp xếp

- Thuật toán sắp xếp đếm có tính ổn định: nó bảo toàn được thứ tự giữa các phần tử có giá trị bằng nhau.



diepht@vnu

Thuật toán sắp xếp cơ số (radix sort)

- Sắp xếp theo từng "chữ số"
  - bằng 1 thuật toán sắp xếp ổn định. VD: counting sort
- Xuất phát từ chữ số ít quan trọng hơn

diepht@vnu

Minh họa radix sort

3 2 9	7 2 0	7 2 0	3 2 9
4 5 7	3 5 5	3 2 9	3 5 5
6 5 7	4 3 6	4 3 6	4 3 6
8 3 9	4 5 7	8 3 9	4 5 7
4 3 6	6 5 7	3 5 5	6 5 7
7 2 0	3 2 9	4 5 7	7 2 0
3 5 5	8 3 9	6 5 7	8 3 9

Chuẩn bị tuần tới

- Lý thuyết: Bài tập
  - SV rà soát các chương học sau thi giữa kì
- Thực hành: Các chiến lược thiết kế thuật toán