HOA SEN UNIVERSITY

www.hoasen.edu.vn

Cấu Trúc Dữ Liệu & Giải Thuật

Data Structures & Algorithms

GV: Phan Hồng Trung

HOA SEN UNIVERSITY

www.hoasen.edu.vn

Bài 12 – Các thuật toán nâng cao

- Sắp xếp theo cơ số
- Thuật toán Tham lam
- Thuật toán Hungary

Sắp xếp theo cơ số (Radix Sort)

- Radix Sort là một thuật toán sắp xếp thông minh và trực quan. Radix Sort sắp xếp các phần tử theo thứ tự bằng cách so sánh các chữ số trong các số.
- Hãy xem xét 9 số sau:
  - 493 812 715 340 195 437 710 582 385
- Chúng ta nên bắt đầu sắp xếp bằng cách so sánh và sắp xếp chữ số hàng đơn vị:
  - Các số được thêm vào danh sách theo thứ tự chúng được tìm thấy, đó là lý do tại sao các số dường như không được sắp xếp trong mỗi danh sách con trong hình bên.
  - Bây giờ, chúng ta tập hợp các danh sách con (theo thứ tự từ danh sách con 0 đến danh sách con 9) vào danh sách chính một lần nữa:
  - 340 710 812 582 493 715 195 385 437
  - Thứ tự chúng ta chia và lắp ráp lại danh sách cực kỳ quan trọng vì đây là một trong những nền tảng của thuật toán này.

Digit	Sublist
0	340 710
1	
2	812 582
3	493
4	
5	715 195 385
6	
7	437
8	
9	

Sắp xếp theo cơ số (Radix Sort)

- 340 710 812 582 493 715 195 385 437

- Bây giờ, các danh sách con được tạo lại, lần này dựa trên chữ số hàng chục & lắp ráp lại:
  - 710 812 715 437 340 582 385 493 195
- Cuối cùng, các danh sách con được tạo ra theo chữ số hàng tram & lắp ráp lại:
  - 195 340 385 437 493 582 710 715 812
- Kết quả là danh sách đã được sắp xếp.
- Radix Sort rất đơn giản và máy tính có thể thực hiện nhanh chóng. Khi được lập trình đúng cách, Radix Sort thực sự là một trong những thuật toán sắp xếp nhanh nhất cho số hoặc chuỗi chữ cái.

Digit	Sublist
0	
1	710 812 715
2	
3	437
4	340
5	
6	
7	
8	582 385
9	493 195

Digit	Sublist
0	
1	195
2	
3	340 385
4	437 493
5	582
6	
7	710 715
8	812
9	

Comparing Sorting Algorithms

#	Algorithms	Worst-Case	Best-Case	Average
1	Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
2	Insertion Sort	$O(n^2)$	O(n)	$O(n^2)$
3	Bubble Sort	$O(n^2)$	O(n)	$O(n^2)$
4	Quick Sort	$O(n^2)$	O(n log n)	O(n log n)
5	Radix Sort	O(n)	O(n)	O(n)

Thuật toán Tham lam

- Thuật toán Tham lam lần lượt chọn phương án tối ưu cục bộ (local optimum) tại mỗi bước, với hy vọng rằng kết quả cuối cùng sẽ là tối ưu tổng thể (global optimum).
- Thuật toán được gọi là "tham lam" vì chiến lược của nó là luôn chọn giải pháp tốt nhất tại từng bước cụ thể mà không quan tâm đến các bước tiếp theo.
- Tên gọi này phản ánh cách tiếp cận của thuật toán: tập trung vào lợi ích cục bộ, giống như một người "tham lam" chỉ nhìn thấy lợi ích ngay trước mắt mà không suy nghĩ xa hơn.

Đặc điểm

- Không quay lui.
- Quyết định tại thời điểm hiện tại mà không xem xét tất cả các tùy chọn khác.
- Độ phức tạp: O(n log n)

Điều kiện áp dụng

- Tính tham lam phải dẫn tới kết quả tối ưu.
- Bài toán phải được chia nhỏ thành các bước con.

Ví dụ

- Giả sử bạn cần trả một số tiền là 63 đồng. Bạn có các mệnh giá tiền như sau:
  - 1 đồng
  - 5 đồng
  - 10 đồng
  - 20 đồng
- Yêu cầu: Tìm cách trả số tiền với số lượng tờ tiền ít nhất.

Ví dụ

- Giải thuật Greedy:
  1. Sắp xếp các mệnh giá theo thứ tự giảm dần.
  2. Duyệt từ mệnh giá cao nhất đến thấp nhất:
     - Với mỗi mệnh giá, chọn số lượng tờ tối đa có thể dùng (không vượt quá số tiền cần trả).
     - Trừ số tiền đã trả ra khỏi tổng số tiền.

Ví dụ

1. Sắp xếp mệnh giá giảm dần: {20,10,5,1}.
2. Bắt đầu chọn tiền:
   - Số tiền cần trả ban đầu là 63.
   - Chọn 20 đồng:
     - Số tờ: $\lfloor 63/20 \rfloor = 3$ tờ.
     - Số tiền còn lại: $63 - 20 \times 3 = 3$.
   - Chọn 10 đồng:
     - Không thể chọn vì $3 < 10$.
   - Chọn 5 đồng:
     - Không thể chọn vì $3 < 5$.
   - Chọn 1 đồng:
     - Số tờ: $\lfloor 3/1 \rfloor = 3$ tờ.
     - Số tiền còn lại: $3 - 1 \times 3 = 0$.
3. Kết quả:
   - Tổng số tờ tiền: $3+0+0+3=6$ tờ.
   - Chi tiết:
     - 3 tờ 20.
     - 3 tờ 1.

Ví dụ

- Một chiếc balo có sức chứa tối đa là **50kg**. Có 4 vật phẩm với các giá trị và trọng lượng như sau:

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

- Chọn các vật phẩm hoặc một phần của vật phẩm (Knapsack dạng **fractional**) sao cho tối ưu hóa giá trị mà tổng trọng lượng không vượt quá 50kg.

Ví dụ

- Giải thuật Greedy:
  1. Sắp xếp vật phẩm theo tỷ lệ giá trị/trọng lượng giảm dần.
  2. Chọn vật phẩm theo thứ tự:
     - Bắt đầu từ vật phẩm có tỷ lệ giá trị/trọng lượng cao nhất.
     - Chọn toàn bộ hoặc một phần của vật phẩm sao cho không vượt quá sức chứa của balo.

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

Ví dụ

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

- Chọn vật phẩm 1:
  - Trọng lượng: 10kg (còn lại 50−10=40kg).
  - Giá trị: 60.

Ví dụ

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

- Chọn vật phẩm 2:
  - Trọng lượng: 20kg (còn lại 40−20=20kg).
  - Giá trị: 100.

Ví dụ

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

- Chọn một phần của vật phẩm 3:
  - Trọng lượng: 20kg (hết sức chứa 20−20=0kg).
  - Giá trị: 20 x 4.0 = 80.

Ví dụ

Vật phẩm	Trọng lượng (kg)	Giá trị (đơn vị tiền)	Tỷ lệ giá trị/trọng lượng
1	10	60	6.0
2	20	100	5.0
3	30	120	4.0
4	40	160	4.0

- Kết quả:
  - Tổng giá trị: 60+100+80=240.
  - Tổng trọng lượng: 10+20+20=50kg.

Bài toán Phân công (Assignment Problem)

- Là bài toán tìm cách phân công nhiệm vụ cho các nhân viên sao cho chi phí thực hiện nhiệm vụ ít nhất.
- Mô tả:
  - Cho ma trận chi phí C, trong đó C[i][j] là chi phí nhân viên i thực hiện nhiệm vụ j.
  - Tìm một hoàn vị (đối xứng) sao cho tổng chi phí là nhỏ nhất.
- Đặc điểm:
  - Input: Ma trận vuông n x n.
  - Output: Danh sách gán nhiệm vụ với chi phí tối thiểu.
  - Yêu cầu: Tối ưu tổng thể (global optimum).

Giới thiệu

- Thuật toán Hungary được thiết kế để giải quyết Assignment Problem một cách hiệu quả.
- Thuật toán sử dụng phương pháp tối thiểu hóa bằng biến đổi ma trận chi phí.
- Độ phức tạp: $O(n^3)$.

Thuật toán

- Bước 1:
  - Nếu số hàng khác số cột, hãy thêm hang/cột giả có chi phí bằng 0 để biến nó thành ma trận vuông.
- Bước 2:
  - a. Xác định phần tử nhỏ nhất trong mỗi hàng và trừ nó khỏi mỗi phần tử của hàng đó.
  - b. Xác định phần tử nhỏ nhất trong mỗi cột và trừ nó khỏi mọi phần tử của cột đó.

Thuật toán

- Bước 3. Thực hiện phép gán trong bảng chi phí:
  - a. Xác định các hàng có đúng một số 0 chưa được đánh dấu. Thực hiện phép gán cho số 0 đơn lẻ này bằng [0] và gạch bỏ tất cả các số 0 khác trong cùng một cột.
  - b. Xác định các cột có đúng một số 0 chưa được đánh dấu. Thực hiện phép gán cho số 0 đơn lẻ này bằng [0] và gạch bỏ tất cả các số 0 khác trong cùng một hàng.
  - c. Nếu một hàng/cột có nhiều số 0 chưa được đánh dấu và không thể chọn một số bằng cách kiểm tra, thì hãy chọn ô đó một cách tùy ý.
  - d. Tiếp tục quá trình này cho đến khi tất cả các số 0 trong các hàng/cột được gán hoặc gạch bỏ.

Thuật toán

- Bước 4:
  - a. Nếu số ô được gán bằng số hàng, thì tìm thấy phép gán tối ưu và trong trường hợp có chọn ô 0 một cách tùy ý, thì có thể tồn tại một giải pháp tối ưu khác.
  - b. Nếu giải pháp tối ưu không phải là tối ưu, thì chuyển đến Bước 5.

Thuật toán

- Bước 5: Vẽ một tập hợp các đường ngang và dọc để bao phủ tất cả các số 0:
  - a. Đánh dấu (✓) vào tất cả các hàng không có [0].
  - b. Kiểm tra các hàng đã đánh dấu (✓), nếu có 0 thì đánh dấu (✓) vào cột đó.
  - c. Kiểm tra các cột đã đánh dấu (✓), nếu có [0] thì đánh dấu (✓) vào hàng đó.
  - d. Lặp lại quy trình này cho đến khi không thể đánh dấu thêm hang/cột nào nữa.
  - e. Vẽ một đường thẳng cho mỗi hàng không được đánh dấu và mỗi cột được đánh dấu.
  - f. Nếu số đường thẳng bằng số hàng thì giải pháp hiện tại là tối ưu, nếu không, hãy chuyển đến bước 6

Thuật toán

- Bước 6. Xây dựng bảng chi phí cơ hội mới đã sửa đổi:
  - a. Chọn phần tử nhỏ nhất, chẳng hạn như k, từ các ô không được bao phủ bởi bất kỳ đường thẳng nào.
  - b. Trừ k khỏi mỗi phần tử không được bao phủ bởi một đường thẳng.
  - c. Thêm k vào mỗi phần tử giao nhau của hai đường thẳng.
- Bước 7. Lặp lại các bước từ 3 đến 6 cho đến khi tìm được giải pháp tối ưu.

Ví dụ: Bước 1

- Ma trận chi phí cơ hội ban đầu, số dòng = số cột = 5

	I	*II*	*III*	*IV*	*V*	
A	10	5	13	15	16	
B	3	9	18	13	6	
C	10	7	2	2	2	
D	7	11	9	7	12	
E	7	9	10	4	12	

Ví dụ: Bước 2

- Trừ min của mỗi hàng

	I	*II*	*III*	*IV*	*V*	
A	5	0	8	10	11	(-5)
B	0	6	15	10	3	(-3)
C	8	5	0	0	0	(-2)
D	0	4	2	0	5	(-7)
E	3	5	6	0	8	(-4)

- Trừ min của mỗi cột

	I	*II*	*III*	*IV*	*V*	
A	5	0	8	10	11	
B	0	6	15	10	3	
C	8	5	0	0	0	
D	0	4	2	0	5	
E	3	5	6	0	8	
	(-0)	(-0)	(-0)	(-0)	(-0)	

Ví dụ: Bước 3

- Ô theo hàng (A,II) được chỉ định
- Ô theo hàng (B,I) được chỉ định, ô theo cột (D,I) bị gạch bỏ
- Ô theo hàng (D,IV) được chỉ định, do đó ô theo cột (C,IV), (E,IV) bị gạch bỏ.
- Ô theo cột (C,III) được chỉ định, do đó ô theo hàng (C,V) bị gạch bỏ.

	I	*II*	*III*	*IV*	*V*	
A	5	**[0]**	8	10	11	
B	**[0]**	6	15	10	3	
C	8	5	**[0]**	~~8~~	~~11~~	
D	~~0~~	4	2	**[0]**	5	
E	3	5	6	~~0~~	8	

Ví dụ: Bước 4

- Số phép gán = 4 != 5 = số hàng, do đó giải pháp không tối ưu.

	I	*II*	*III*	*IV*	*V*	
A	5	**[0]**	8	10	11	
B	**[0]**	6	15	10	3	
C	8	5	**[0]**	~~8~~	~~3~~	
D	~~0~~	4	2	**[0]**	5	
E	3	5	6	~~0~~	8	

Ví dụ: Bước 5

- Phủ số 0 bằng số đường thẳng ít nhất

  (1) Đánh dấu (✓) hàng E vì nó không có phép gán

  (2) Đánh dấu (✓) cột IV vì hàng E có 0 trong cột này

  (3) Đánh dấu (✓) hàng D vì cột IV có phép gán trong hàng D này.

  (4) Đánh dấu (✓) cột I vì hàng D có 0 trong cột này

  (5) Đánh dấu (✓) hàng B vì cột I có phép gán trong hàng B này.

  (6) Vì không thể đánh dấu các hàng hoặc cột khác, do đó hãy vẽ các đường thẳng qua các hàng A, C không được đánh dấu và các cột I, IV được đánh dấu

	I	*II*	*III*	*IV*	*V*	
A	~~5~~	**[0]**	~~8~~	~~10~~	~~11~~	
B	**[0]**	6	15	10	3	✓(5)
C	~~8~~	~~5~~	**[0]**	~~0~~	~~0~~	
D	~~0~~	4	2	**[0]**	5	✓(3)
E	3	5	6	~~0~~	8	✓(1)
	✓ (4)			✓ (2)		

Ví dụ: Bước 6

- Chọn k = 2 là phần tử nhỏ nhất trong số các ô không được bao phủ bởi bất kỳ đường thẳng nào.
- Trừ k = 2 cho mọi phần tử trong các ô không được bao phủ bởi các đường thẳng.
- Thêm k = 2 vào mọi phần tử trong ô giao nhau của hai đường thẳng.
- Lặp lại các bước từ 3 đến 6 cho đến khi tìm được giải pháp tối ưu.

	I	*II*	*III*	*IV*	*V*	
A	7	0	8	12	11	
B	0	4	13	10	1	
C	10	5	0	2	0	
D	0	2	0	0	3	
E	3	3	4	0	6	

Ví dụ: Bước 3

- Thực hiện gán trong bảng chi phí cơ hội
  - (1) Ô theo hàng (A,II) được chỉ định
  - (2) Ô theo hàng (B,I) được chỉ định, do đó ô theo cột (D,I) được gạch bỏ.
  - (3) Ô theo hàng (E,IV) được chỉ định, do đó ô theo cột (D,IV) được gạch bỏ.
  - (4) Ô theo cột (C,V) được chỉ định, do đó ô theo hàng (C,III) được gạch bỏ.
  - (5) Ô theo hàng (D,III) được chỉ định

	I	*II*	*III*	*IV*	*V*	
A	7	**[0]**	8	12	11	
B	**[0]**	4	13	10	1	
C	10	5	~~0~~	2	**[0]**	
D	~~0~~	2	**[0]**	~~0~~	3	
E	3	3	4	**[0]**	6	

Ví dụ: Bước 4

- Số phép gán = 5 = số hàng, do đó giải pháp là tối ưu.

	I	*II*	*III*	*IV*	*V*	
A	7	**[0]**	8	12	11	
B	**[0]**	4	13	10	1	
C	10	5	~~0~~	2	**[0]**	
D	~~0~~	2	**[0]**	~~0~~	3	
E	3	3	4	**[0]**	6	

- Giải pháp tối ưu:

Work	Job	Cost
A	*II*	5
B	*I*	3
C	*V*	2
D	*III*	9
E	*IV*	4
	Total	23


Q
A