



CHƯƠNG 6

CHIẾN LƯỢC

THAM LAM



Nội dung chính chương 6

6.1 Nguyên tắc tham lam

6.2 Bài toán đổi tiền

6.3 Bài toán lập lịch

6.4 So sánh chiến lược tham lam và quy hoạch động



6.1 Nguyên tắc tham lam

- Dùng để giải quyết các bài toán tối ưu.
- Ý tưởng:

Với một tập các khả năng có thể lựa chọn, thay vì thử hết các khả năng có thể, thì tại mỗi bước lựa chọn khả năng tốt nhất tại lúc đó.

Lựa chọn tối ưu cục bộ tại mỗi bước. Không xét toàn bộ như vét cạn.

Hy vọng tối ưu cục bộ này có thể dẫn đến lời giải tối ưu toàn cục.



6.1 Nguyên tắc tham lam

- Cấu hình cần tìm có dạng $A=(a_1, a_2, \dots, a_n)$.
- Xây dựng từng phần a_i của cấu hình A bằng cách chọn giá trị tối ưu của a_i .
- Ở bước cuối, có thể không còn gì để chọn mà phải chấp nhận giá trị cuối cùng còn lại.
- Độ phức tạp: đa thức. Tốt. (Quay lui, nhánh cận: độ phức tạp là hàm mũ.)
- Chỉ đạt được phương án tốt, chưa hẳn là tối ưu.



6.1 Nguyên tắc tham lam

Ưu điểm:

- Dễ cài đặt, dễ gỡ lỗi.
- Nhanh: độ phức tạp đa thức: $O(n)$, $O(n^2)$...
- Sử dụng ít bộ nhớ.

Nhược điểm:

- Chỉ đạt được phương án tốt, chưa hẳn là tối ưu.
- Thường không mang tính tổng quát. Cần có cơ sở vững chắc về toán học (chứng minh tính đúng đắn).

6.1.1 Giải thuật tham lam

Giải thuật: các thành phần chính (không nhất thiết đủ)

- Một tập hợp các ứng viên, để từ đó tạo ra lời giải.
- **Một hàm lựa chọn**, để theo đó lựa chọn ứng viên tốt nhất để bổ sung vào lời giải.
- Một hàm khả thi, dùng để quyết định nếu một ứng viên có thể được dùng để xây dựng lời giải.
- Một hàm mục tiêu, ấn định giá trị của lời giải hoặc một lời giải chưa hoàn chỉnh.
- Một hàm đánh giá, chỉ ra khi nào ta tìm ra một lời giải hoàn chỉnh.

6.1.1 Giải thuật tham lam

```
void Greedy_Method(X, A)
{
    // Từ tập các đối tượng X, ta xây dựng cấu hình nghiệm cho A
    A= Ø;
    while (X ≠ Ø)
    {
        a= Select(X); // Hàm chọn a tốt nhất từ X
        if (A ∪ {a}) chấp nhận được then A=A ∪ {a};
        X=X-{a};
    }
    return (A);      // phương án tối ưu
}
```



6.2 Bài toán đổi tiền

Một cửa hàng sử dụng các đồng xu: 1, 5, 10, 25, 100 cents để trả lại tiền thừa cho khách.

Đưa ra cách thức trả cho khách với số lượng đồng tiền là ít nhất.

Giá trị sau ≥ 2 lần giá trị trước.

Điều kiện thỏa mãn thì tham lam chắc chắn đúng.

Nếu GT sau < 2 lần GT trước thì có thể chọn 2 GT trước chưa chắc tệ hơn chọn 1 GT phía sau.



6.2 Bài toán đổi tiền

Ví dụ:

- Cần đổi 34 cents thì sử dụng 1 đồng 25 cents, 1 đồng 5 cents và 4 đồng 1 cent
- Khách cần đổi 2 usd 89 cents:
 - 2 đồng 1 usd
 - 3 đồng 25 cents
 - 1 đồng 10 cents
 - 4 đồng 1 cent





6.2 Bài toán đổi tiền

- N là giá trị tiền lẻ phải trả lại khách.
 - Đặt số đồng xu loại 100, 25, 10, 5, 1 cent cần trả lại khách lần lượt là (a_1, a_2, a_3, a_4, a_5).
 - $N = 100a_1 + 25a_2 + 10a_3 + 5a_4 + 1a_5$.
- Bài toán trở thành tìm cấu hình $A = (a_1, a_2, a_3, a_4, a_5)$ có $F(A) = a_1 + a_2 + a_3 + a_4 + a_5$ nhỏ nhất.
- Có thể giải bằng quay lui: xét mọi cấu hình A và ghi nhận cấu hình tốt nhất.
- Nhưng ta nhận thấy: giá trị a_1 càng lớn càng tốt.
 - Vì nếu a_1 bớt đi một, thì phải thay thế đồng xu 100 bằng các loại khác nhỏ hơn. Chẳng hạn 4 đồng xu 25 cents hoặc 10 đồng xu 10 cents hoặc 20 đồng xu 5 cent hoặc 100 đồng xu 1 cent.



6.2 Bài toán đổi tiền

■ Lập luận tương tự:

- Sau khi không thể dùng thêm các đồng xu 100 cents (tức là đã xác định được giá trị a_1): ta thấy a_2 càng lớn càng tốt, hoặc phải thay thế một đồng xu 25 cents bằng nhiều đồng xu mệnh giá nhỏ hơn.
- Khi không dùng được các đồng xu 100 cents và 25 cents: ta thấy a_3 càng lớn càng tốt, hoặc phải thay thế đồng xu 10 cents bằng các đồng xu mệnh giá bé hơn.
- ...
- “lợi ích lớn nhất” (tham lam): dùng càng nhiều tiền mệnh giá cao càng tốt.
- Chứng minh tham lam là tối ưu: quy nạp theo số N.



6.2 Bài toán đổi tiền

```
#include <iostream>
using namespace std;
const int MAX = 5;
int c[MAX] = { 100, 25, 10, 5 , 1 };//đã sắp xếp giảm dần
int a[MAX], n;

int main() {
    cout << "n = "; cin >> n;
    //Sắp xếp c[MAX]
    for (int i = 0; i < MAX; i++) {
        a[i] = n / c[i];
        cout << "So xu " << c[i] << ":" << a[i] << endl;
        n = n - c[i] * a[i];
    }
}
```



6.2 Bài toán đổi tiền

- **Thuật toán tham lam không đúng với bài toán đổi tiền tổng quát.**
- Với các đồng xu mệnh giá: 1, 10, 21, 34, 70, 100.
- Nếu cần đổi số tiền $N = 140$?
 - Tham lam: $140 = 100 + 34 + 1 + 1 + 1 + 1 + 1 + 1$
 - Tối ưu: $140 = 70 + 70$
- **Phụ thuộc vào bộ dữ liệu ta xét!**
- **Cùng 1 bài toán, cùng 1 mục tiêu nhưng bộ dữ liệu khác nhau sẽ cho tối ưu/không tối ưu.**
- **Một kỹ thuật tham lam đi kèm với kỹ thuật giải toán trên bộ dữ liệu.**



6.2.1 Không phải bài toán nào cũng tham lam được

Bài toán Sudoku

Có một hình vuông được chia thành 9 x 9 ô vuông con. Mỗi ô vuông con có giá trị trong khoảng từ 1 đến 9. Ban đầu hình vuông có một số ô vuông con cho trước (có điền sẵn số) và còn lại là trống. Hãy điền các số từ 1-9 vào các ô con lại sao cho: hàng ngang là các số khác nhau từ 1 đến 9, hàng dọc là các số khác nhau từ 1 đến 9, đường chéo là các số khác nhau từ 1 đến 9 và mỗi khối 3x3 chính là các số khác nhau từ 1 đến 9.



6.2.1 Không phải bài toán nào cũng tham lam được

Xét tham lam từ giá trị nhỏ đến lớn:

8		6				2
	4		5		1	
		7				3
	9		4			6
2						8
7		1		5		
3			9			
	1		8		9	
4			2			5

8	<u>3</u>	<u>1</u>	6	<u>4</u>	<u>?</u>			2
	4			5		1		
			7				3	
	9				4		6	
2							8	
7				1		5		
3					9			
	1			8			9	
4					2			5



6.2.2 Tham lam gần đúng

Vết cạn thì độ phức tạp quá lớn. Chấp nhận chọn tham lam gần đúng nhất trong những cách tham lam. Khi chưa có thuật toán tối ưu.

Ví dụ:

Một dự án có tổng thời gian thực hiện T tuần. Có n công việc được thực hiện. Mỗi công việc i có thời gian thực hiện là t_i và giá trị cho trước v_i .

Hãy lựa chọn công việc thực hiện sao cho tổng thời gian không vượt quá T tuần và có **tổng giá trị là lớn nhất**.



6.2.2 Tham lam gần đúng

Công việc	Thời gian (tuần)	Giá trị (đồng)	T: 26 tuần
1	15	210	
2	12	220	
3	10	180	
4	9	120	
5	8	160	
6	7	170	
7	5	90	
8	4	40	
9	3	60	
10	1	10	



6.2.2 Tham lam gần đúng

Chọn tham lam theo thời gian: Thời gian giảm dần.

Chọn: việc 1, 3, 10:

Tổng thời gian: 26 tuần, tổng giá trị: 400.

Công việc	Thời gian (tuần)	Giá trị (đồng)	T: 26 tuần
1	15	210	
2	12	220	
3	10	180	
4	9	120	
5	8	160	
6	7	170	
7	5	90	
8	4	40	
9	3	60	
10	1	10	



6.2.2 Tham lam gần đúng

Chọn tham lam theo giá trị: giá trị giảm dần.

Việc 2, 3, 9, 10:

Tổng thời gian: 26 tuần, tổng giá trị: 470.

Công việc	Thời gian (tuần)	Giá trị (đồng)	T: 26 tuần
2	12	220	
1	15	210	
3	10	180	
6	7	170	
5	8	160	
4	9	120	
7	5	90	
9	3	60	
8	4	40	
10	1	10	



6.2.2 Tham lam gần đúng

Chọn tham lam theo tỷ lệ giữa giá trị/thời gian: tỷ lệ giảm dần.

Việc 6, 5, 9, 7, 10:

Tổng thời gian: 24 tuần, tổng giá trị: 490.

Công việc	Thời gian (tuần)	Giá trị (đồng)	Tỷ lệ: Giá trị/thời gian	T: 26 tuần
6	7	170	24,286	
5	8	160	20	
9	3	60	20	
2	12	220	18,333	
3	10	180	18	
7	5	90	18	
1	15	210	14	
4	9	120	13,333	
8	4	40	10	
10	1	10	10	



6.2.2 Tham lam gần đúng

Vết cạn: tối ưu nhất.

Việc 3, 5, 6, 10:

Tổng thời gian: 26 tuần, tổng giá trị: 520.

Tham lam: $O(n^2)$ sắp xếp

Vết cạn: $O(2^n)$

Công việc	Thời gian (tuần)	Giá trị (đồng)	T: 26 tuần
1	15	210	
2	12	220	
3	10	180	
4	9	120	
5	8	160	
6	7	170	
7	5	90	
8	4	40	
9	3	60	
10	1	10	



6.2.3 Nhận xét

- Cùng dạng bài toán nhưng với bộ dữ liệu khác nhau dẫn đến có giải được bằng tham lam hay không?
- Có những bài toán không thể giải được bằng tham lam.
- Có những bài toán chấp nhận tham lam cho ra kết quả gần đúng, chưa được tối ưu nhưng chấp nhận được.
- Các bài toán áp dụng được tham lam: cần chứng minh về mặt toán học.



6.2.3 Nhận xét

Học những bài toán tham lam cơ bản đã được chứng minh tính đúng đắn:

- Bài toán đổi tiền.
- Bài toán sắp xếp công việc.
- Bài toán nối dây.
- Bài toán sắp đặt xâu ký tự.

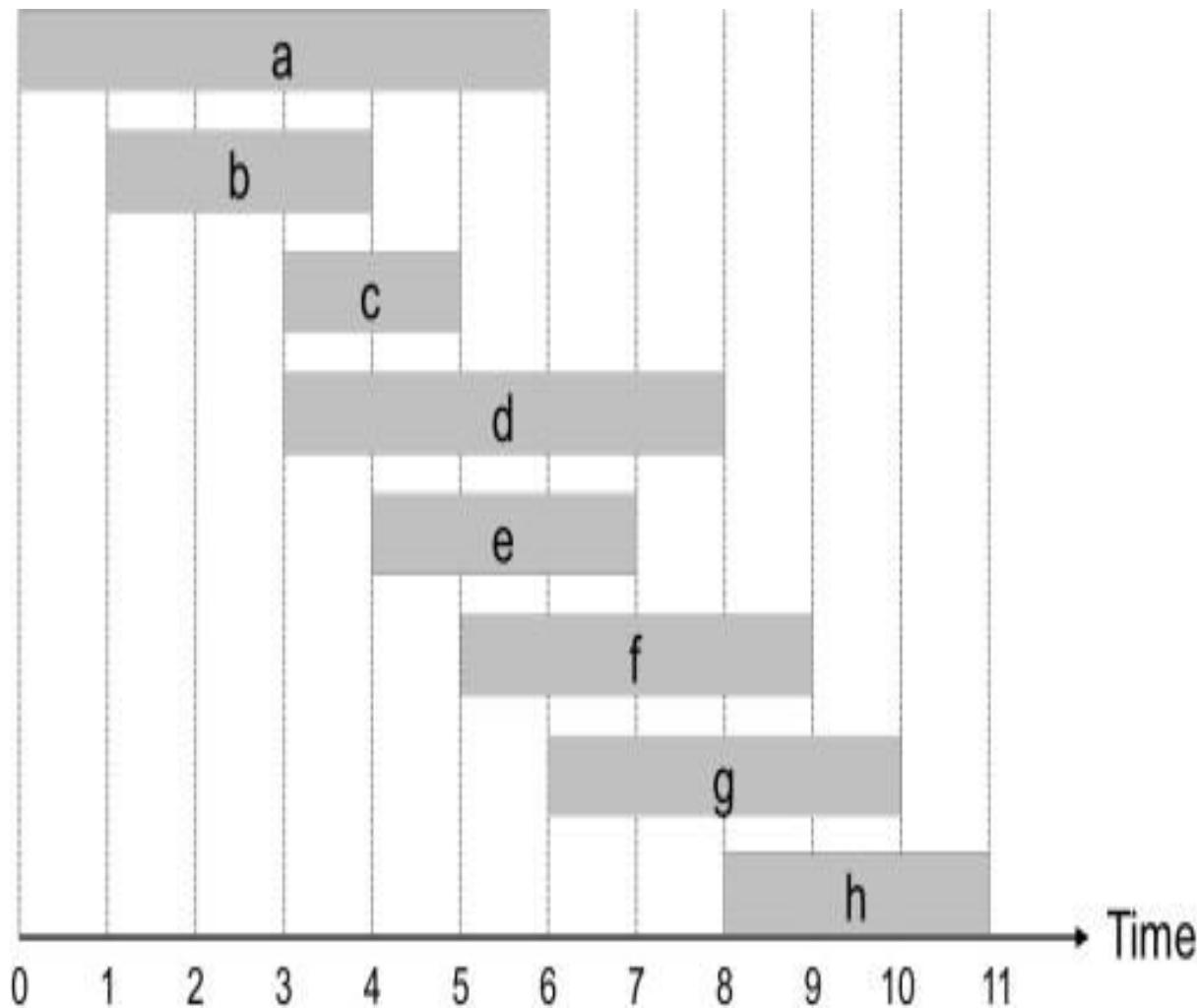


6.3 Bài toán lập lịch

- Công ty dự kiến tổ chức N cuộc họp, cuộc họp thứ k bắt đầu từ thời điểm s_k và kết thúc ở thời điểm f_k .
- Phòng họp công ty không thể tổ chức hai cuộc họp cùng một lúc, vì vậy phải loại bỏ một số cuộc họp nếu chúng có thời gian họp chồng lấn lên nhau.
- Tìm cách tổ chức **nhiều cuộc họp nhất** có thể.



6.3 Bài toán lập lịch





6.3 Bài toán lập lịch

- Bài toán trở thành tìm cấu hình $A = (a_1, a_2, \dots, a_n)$.
 - (a_1, a_2, \dots, a_n) lần lượt là số thứ tự các cuộc họp được tổ chức theo trực thời gian tăng dần.
 - Cuộc họp a_k không xung đột với các cuộc họp $(a_1, a_2, \dots, a_{k-1})$.
 - Cuộc họp a_k tổ chức sau các cuộc họp $(a_1, a_2, \dots, a_{k-1})$.
 - Cuộc họp a_k tổ chức sau khi cuộc họp a_{k-1} kết thúc.
 - Số n càng lớn càng tốt.
- Có thể giải bằng quay lui:
 - Chọn dần các giá trị a_k sao cho không xung đột với $(a_1, a_2, \dots, a_{k-1})$
 - Ghi nhận cấu hình có k lớn nhất.



6.3 Bài toán lập lịch

■ Tham lam:

- Cuộc họp a_k không xung đột với các cuộc họp $(a_1, a_2, \dots, a_{k-1})$.
- Cuộc họp a_k bắt đầu sau khi a_{k-1} kết thúc.
- Cuộc họp a_k kết thúc càng sớm càng tốt.

■ Thuật toán:

- Sắp xếp các cuộc họp tăng dần theo thời điểm kết thúc.
- Lần lượt chọn các cuộc họp theo tiêu chí:
 - Bắt đầu sau cuộc họp trước đó (tránh xung đột).
 - Kết thúc sớm nhất có thể.



6.3 Bài toán lập lịch

```
#include <iostream>
#include <algorithm>
using namespace std;

const int MAX = 8;
int s[MAX] = { 0, 1, 3, 3, 4, 5, 6, 8 };
int f[MAX] = { 6, 4, 5, 8, 7, 9, 10, 11 };
int a[MAX], e;

bool sapxep(int i, int j) { return f[i] < f[j]; }

int main() {
    for (int i = 0; i < MAX; i++)
        a[i] = i;
    sort(a+0, a+MAX, sapxep);
    cout << "Cac cuoc hop: " << a[0];
    e = f[a[0]];
    for (int i = 1; i < MAX; i++)
        if (e <= s[a[i]]) {
            cout << " " << a[i];
            e = f[a[i]];
        }
}
```



Tóm lược về tiếp cận tham lam

- Tham lam là dạng đơn giản hóa của quay lui:
 - Thay vì duyệt nhiều phương án (rẽ nhánh), ta chọn phương án tốt nhất để đi thẳng.
 - Không cần quay lui (vì không có nhánh khác để lựa chọn).
 - Trong đa phần các bài toán lời giải tham lam có độ phức tạp tuyến tính hoặc bậc hai theo số lượng thành phần con (theo bậc của cấu hình A).
- Trong cuộc sống, chiến lược này giống như việc chỉ tính trước một nước và chọn nước đi đem lại nhiều lợi nhất.
- Mô hình toán học thì tiếp cận tham lam là lựa chọn tối ưu cục bộ, như vậy những bài toán quy hoạch lồi thì tiếp cận này sẽ cho ra kết quả tối ưu.



Tóm lược về tiếp cận tham lam

- Lớp các bài toán mà tiếp cận tham lam đem lại kết quả tối ưu gọi là “Greedoid”.
- Nhiều thuật toán nổi tiếng thuộc lớp greedoid: Dijkstra, Prim, Kruskal, mã hóa huffman,...
- Tham lam hữu ích ngay cả khi thuật toán này không đem lại kết quả tối ưu:
 - Thực tế thì người ta cũng không cần kết quả tối ưu mà chỉ cần những phương án đủ tốt, tiệm cận với tối ưu.
 - Sử dụng tham lam làm cận khi duyệt: thay vì duyệt bằng nhánh cận ngay từ đầu, ta có thể dùng tiếp cận tham lam để tìm ra một nghiệm đủ tốt, và sử dụng kết quả này làm cận trên cho việc tìm kiếm nghiệm tối ưu.



6.4 So sánh chiến lược tham lam và quy hoạch động

Giống nhau:

- Giải quyết vấn đề tối ưu.
- Chia một vấn đề lớn thành nhỏ hơn và tìm giải pháp cho từng vấn đề nhỏ.
- Ghi nhớ/lưu trữ tạm để tránh tính toán lại và cải thiện hiệu quả về thời gian.
- Đều có đặc tính là "từ dưới lên".
- Giải pháp tối ưu cho vấn đề khác nhau, nhưng tính chính xác của giải pháp đều phụ thuộc vào toán học.



6.4 So sánh chiến lược tham lam và quy hoạch động

- Chọn chiến lược nào phụ thuộc vấn đề và giải pháp mong muốn.
 - Tham lam thường được sử dụng khi không thể tìm được giải pháp tối ưu toàn cục hoặc quá tốn thời gian.
 - QHĐ được sử dụng khi có thể tìm được giải pháp tối ưu toàn cục.



6.4 So sánh chiến lược tham lam và quy hoạch động

Khác nhau:

1. Cách tiếp cận :

Tham lam : "lựa chọn tốt nhất hiện tại".

QHĐ: dựa trên "chia để trị".

2. Tính tối ưu :

QHĐ : đảm bảo tính tối ưu của giải pháp.

Tham lam: thì không.

3. Đặc điểm của bài toán :

QHĐ : bài toán có cấu trúc con tối ưu /chồng lấn.

Tham lam: bài toán có tính chất lựa chọn tham lam tối ưu.



6.4 So sánh chiến lược tham lam và quy hoạch động

- 4. Độ phức tạp về thời gian:** QHĐ có độ phức tạp về thời gian cao hơn so với tham lam.
- 5. Độ phức tạp về không gian:** QHĐ có độ phức tạp về không gian cao hơn so với tham lam.
- 6. Ghi nhớ:** QHĐ sử dụng ghi nhớ để lưu trữ các giải pháp cho các bài toán con, tham lam thì không.
- 7. Khả năng tái sử dụng:** Các giải pháp QHĐ có thể được tái sử dụng tham lam chỉ phù hợp với trường hợp cụ thể của vấn đề.
- 8. Quyết định:** Các giải pháp QHĐ dựa trên một loạt các quyết định, tham lam dựa trên một lựa chọn tối ưu cục bộ duy nhất.



Bài tập

1. Máy tính cần thực hiện N tác vụ. Tác vụ thứ k có deadline là d_k và điểm p_k , nghĩa là nếu tác vụ hoàn thành không muộn hơn thời điểm d_k thì sẽ được thưởng p_k điểm, ngược lại thì sẽ không được thưởng gì.

Mỗi tác vụ đều cần 1 đơn vị thời gian để thực hiện, máy tính bắt đầu thực hiện từ thời điểm 0. Hãy chỉ ra lịch trình thực hiện để có thể được nhiều điểm thưởng nhất.

Ví dụ:	$N = 5$	Tác vụ:	0	1	2	3	4
		$D:$	2	1	2	1	3
		$P:$	100	19	27	25	15

Thứ tự thực hiện là 0-2-4-1-3 với tổng điểm thưởng là 142
(hoặc thực hiện theo thứ tự 0-2-4-3-1)



Bài tập

2. Một cửa hàng có N đồ vật, đồ vật thứ k có trọng lượng w_k và giá trị p_k .

Một tên trộm lọt vào cửa hàng, hắn chỉ có thể lấy trộm số đồ vật có tổng trọng lượng tối đa W đơn vị.

Hãy chỉ ra cách lựa chọn các đồ vật ăn trộm sao cho tổng giá trị lấy trộm được là lớn nhất.

Ví dụ: $W = 19, N = 3$

Đồ vật:	1	2	3
Giá trị:	20	16	8
Trọng lượng:	14	10	6

Phương án tối ưu: lấy đồ vật 2 và 3



Bài tập

3. Tìm số nguyên dương K nhỏ nhất mà tích các chữ số của K bằng N. Nếu không tồn tại K thì in ra thông báo “NO”.

Ví dụ: $N = 12 \quad K = 26$

$N = 0 \quad K = 10$

$N = 33 \quad NO$

4. Một trang trại có N con bò sữa, con thứ k cho sản lượng sữa a_k lít trong mỗi lần vắt sữa. Tuy nhiên, mỗi khi vắt sữa một con bò thì những con còn lại sẽ e sợ và số sữa giảm đi 1 lít. Hãy chỉ ra thứ tự vắt sữa để thu được nhiều nhất.

Ví dụ: $N = 4, A = (4, 4, 4, 4) \quad KQ = 10$

$N = 4, A = (2, 1, 4, 3) \quad KQ = 6$

Thank You!