THUẬT TOÁN ỨNG DỤNG

Thuật toán Tham lam

Nội dung

1. Ý tưởng tham lam
2. Bài toán đổi tiền
3. Bài toán lập lịch
4. Tóm lược về tiếp cận tham lam
5. Bài tập

Phần 1

Ý tưởng tham lam

Ý tưởng tham lam

- Bài toán tối ưu: tìm cực trị theo hàm mục tiêu

$$z = min\{f(x)|x \in X\}$$

hoặc:

$$\bar{x} = argmin\{f(x)|x \in X\}$$

- Quay lui:
  - Duyệt toàn bộ mọi cấu hình x
  - Ghi nhận cực trị của hàm f(x)
- Nhánh cận:
  - Vẫn là quay lui
  - Quay lui sớm nếu đánh giá thấy nhánh hiện tại không đủ tốt
- Vấn đề: thời gian thực hiện vẫn quá lớn (do độ phức tạp tính toán vẫn là hàm mũ)

Ý tưởng tham lam

- Cấu hình đầu tiên là rỗng: A = ()
- Tìm cách xây dựng dần dần các phần tử a_1, a_2,... a_N
- Quy tắc xây dựng phần tử a_k:
  - Nếu k > N: cấu hình A đã hoàn chỉnh, ghi nhận và kết thúc
  - Xây dựng tập S_k chứa mọi giá trị có thể của a_k
  - Chọn **một giá trị** trong S_k đem lại "lợi ích lớn nhất" cho hàm mục tiêu, gán cho a_k nhận giá trị này
  - Lặp lại quá trình để xây dựng phần tử a_{k+1}
- Như vậy, tham lam dựa trên quay lui, nhưng:
  - Không có bước quay lui
  - Không cần viết đệ quy
  - Thế nào là "lợi ích lớn nhất" <~~ lý do của tên gọi "tham lam"

Phần 2

Bài toán đổi tiền

Bài toán đổi tiền

Một cửa hàng bán lẻ 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 sử dụng ít đồng tiền.

Ví dụ:

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

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 = 100 x a_1 + 25 x a_2 + 10 x a_3 + 5 x a_4 + 1 x a_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 10 cent hoặc 100 đồng xu 1 cent.

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

Bài toán đổi tiền

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

const int MAX = 5;
int c[MAX] = { 100, 25, 10, 5 , 1 };
int a[MAX], n;

int main() {
    cout << "N = "; cin >> n;
    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];
    }
}
```

Bài toán đổi tiền

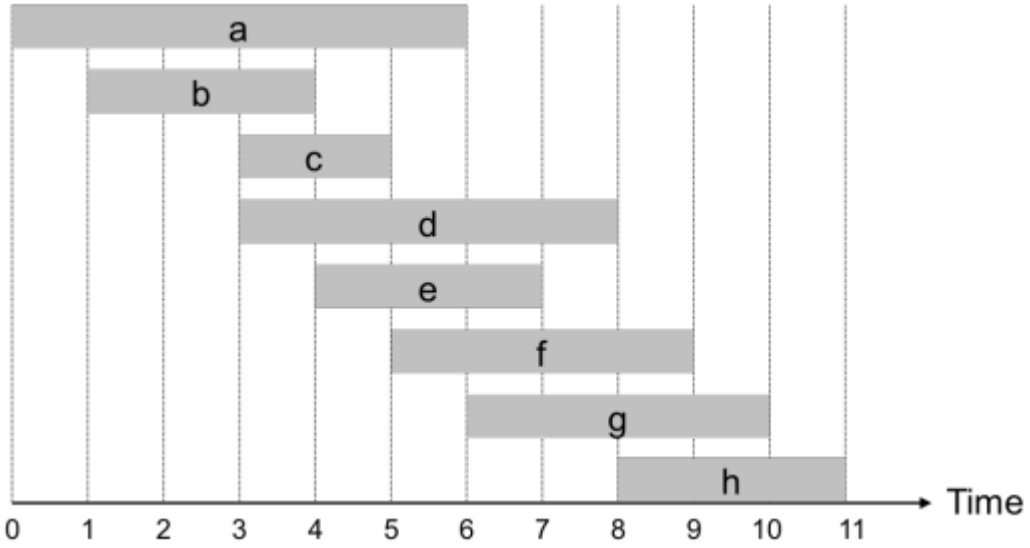
- Thuật toán tham lam không đúng với bài toán đổi tiền tổng quát
- Chẳng hạn, với các đồng xu mệnh giá: 1, 10, 21, 34, 70, 100, 350, 1225, 1500. Nếu cần đổi số tiền N = 140?
  - Tham lam: 140 = 100 + 34 + 1 + 1 + 1 + 1 + 1 + 1
  - Tối ưu: 140 = 70 + 70
- Như vậy tham lam chỉ tối ưu nếu may mắn? Có thể :D

Phần 3

Bài toán lập lịch

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 lần lên nhau
- Tìm cách tổ chức nhiều cuộc họp nhất có thể



Bài toán lập lịch

- Bài toán trở thành tìm cấu hình A = (a_1, a_2,... a_M)
  - (a_1, a_2,... a_M) lần lượt là số thứ tự các cuộc họp được tổ chức theo trục thời gian tăng dần
  - Cuộc họp a_k không xung đột với các cuộc họp (a_1, a_2,... a_{k-1})
  - Cuộc họp a_k tổ chức sau các cuộc họp (a_1, a_2,... a_{k-1})
  - Cuộc họp a_k tổ chức sau khi cuộc họp a_{k-1} kết thúc
  - Số M 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,... a_{k-1})
  - Ghi nhận cấu hình có k lớn nhất

Bài toán lập lịch

- Tham lam:
  - Cuộc họp a_k không xung đột với các cuộc họp (a_1, a_2,... 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
- Chứng minh thuật toán này tối ưu? Dễ

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]];
        }
}
```

Phần 4

Tóm lược về tiếp cận tham lam

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

Phần 5

Bài tập

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 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 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 trôm 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 "NO".

Ví dụ:	*N = 12*	*K = 26*
	N = 0	*K = 10*
	N = 33	*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)*	*KQ = 10*
	N = 4, A = (2, 1, 4, 3)	*KQ = 6*