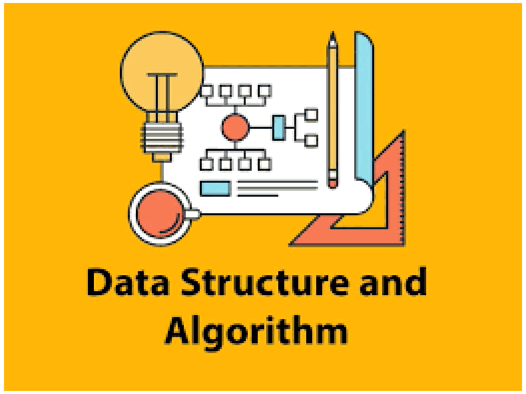
CẤU TRÚC DỮ LIỆU & GIẢI THUẬT

NGÀY 6: THAM LAM & QUY HOẠCH ĐỘNG



Giảng viên: Th.S Bùi Văn Kiên

Nội dung

- Thuật toán tham lam
  - Khái niệm
  - Tham lam gần đúng
  - Các bài toán ứng dụng
- Thuật toán quy hoạch động
  - Khái niệm
  - Các bài toán ứng dụng

1.1 Thuật toán tham lam

- Bài toán tối ưu:

Tìm $\min \{ f(X) : X \in D \}$.

hoặc tìm $\max \{ f(X) : X \in D \}$.

  - *D* là tập hữu hạn các phần tử thỏa mãn tính chất P nào đó.

    $D = \{ X = (x_1, x_2, .., x_n) \in A_1 \times A_2 \times A_n : X \text{ thỏa mãn tính chất } P \}$
  - $X \in D$: phương án
  - Hàm f(X): hàm mục tiêu
  - Miền D: Tập phương án.

1.1 Thuật toán tham lam

Giải thuật tham lam (Greedy Algorithm)

- Ý tưởng chính:
  - Lựa chọn tối ưu cục bộ tại mỗi bước
  - Mong muốn tìm ra lựa chọn tối ưu toàn cục.
- Giải thuật tham lam thường không mang tính tổng quát. Cần có điểm tựa vững chắc về mặt toán học.
- Độ phức tạp thời gian thường tốt hơn nhiều so với Duyệt toàn bộ và Nhánh cận, hay cả thuật toán Quy hoạch động.

1.1 Thuật toán tham lam

5 thành phần chính trong bài toán:

- Tập các ứng viên mà giải pháp thực hiện tạo ra.
- Hàm lựa chọn (selection function)
- Hàm thực thi (feasibility function)
- Hàm mục tiêu (objective function)

→ đánh giá độ tốt/xấu của mỗi lựa chọn trong bài toán

- Hàm giải pháp (solution function)

→ chọn hành động tốt nhất dựa trên hàm mục tiêu đã xác định ở bước trước.

1.1 Thuật toán tham lam

Nguyên tắc hoạt động

- Xác định bước tham lam:

Tại mỗi bước, ta lựa chọn hành động tốt nhất dựa trên hàm mục tiêu đã xác định ở bước trước.

- Kiểm tra điều kiện dừng:

Thuật toán tiếp tục lựa chọn bước tham lam cho đến khi đạt được kết quả tối ưu hoặc không thể lựa chọn thêm bước nào mà vẫn đảm bảo kết quả tối ưu.

1.1 Thuật toán tham lam

Ví dụ: Bài toán phân tích số

- Cho trước số nguyên N và tập hợp X = {x_1, x_2, …, x_K}.
- Hãy tìm cách phân tích N thành tổng của các số trong tập X và có ít phần tử nhất có thể.
- Trường hợp 1: X = {1, 2, 5, 10, 20, 50} và N = 72
- Phương án: cứ chọn các phần tử có giá trị lớn nhất trước sao cho tổng các phần tử được chọn vẫn thỏa mãn <= N.

→ 72 = 50 + 20 + 2

→ Đây là một kết quả tối ưu

1.1 Thuật toán tham lam

Ví dụ: Bài toán phân tích số

- Phương án: cứ chọn các phần tử có giá trị lớn nhất trước sao cho tổng các phần tử được chọn vẫn thỏa mãn <= N.

```
void greedy(int N, int X[], int ans[], int K) {
    for (int i = K; i >= 1; --i) {
        ans[i] = 0;
        while (X[i] <= N) {
            N -= X[i];
            ans[i]++;
        }
    }
}
```

1.1 Thuật toán tham lam

Ví dụ: Bài toán phân tích số

- Trường hợp 2: X = {1, 4, 9, 16, 25, 36, 49} và N = 72

- Phương án tham lam: cứ chọn các phần tử có giá trị lớn nhất trước sao cho tổng các phần tử được chọn vẫn thỏa mãn <= N.

→ 72 = 49 + 16 + 4 + 1 + 1 + 1, sử dụng 6 phần tử

- Phương án tối ưu: 72 = 36 + 36, chỉ gồm 2 phần tử

Nhận xét:

- Vì sao trường hợp 1 thì giải thuật tham lam cho kết quả tối ưu, trường hợp 2 thì không?

1.1 Thuật toán tham lam

ĐÁNH GIÁ:

- Cùng một dạng bài toán nhưng các trường hợp dữ liệu khác nhau dẫn tới có thể giải bằng tham lam được hay không.
- Có những bài toán không thể giải bằng tham lam
- **Có những bài toán có thể chấp nhận tham lam cho ra kết quả gần đúng.**
- Với các bài toán áp dụng được tham lam: cần chứng minh về mặt toán học.
- Yêu cầu cơ bản: hãy cài đặt các giải thuật tham lam đã được chứng minh tính đúng đắn.

1.2 Ví dụ: thứ tự sản xuất

Xét bài toán:

- Một dự án sản xuất có tổng thời gian thực hiện là 26 tuần
- Mỗi sản phẩm (từ A đến J) có thời gian thực hiện và giá trị cho trước.
- Hãy sắp xếp thứ tự sản xuất các sản phẩm sao cho tổng giá trị là lớn nhất.

1.2 Ví dụ: thứ tự sản xuất

Xét bài toán:

Product ID	Completion Time (wks)	Expected Revenue (1000 $)
A	15	210
B	12	220
C	10	180
D	9	120
E	8	160
F	7	170
G	5	90
H	4	40
I	3	60
J	1	10

1.2 Ví dụ: thứ tự sản xuất

- Giải pháp

Tham lam theo thời gian:

- Project A: 15 tuần
- Project C: 10 tuần
- Project J: 1 tuần

- Tổng thời gian: 26 tuần

- Giá trị: $400 000

Product ID	Completion Time (weeks)	Expected Revenue (1000 $)
A	**15**	**210**
B	**12**	220
C	**10**	**180**
D	**9**	120
E	**8**	160
F	**7**	170
G	**5**	90
H	**4**	40
I	**3**	60
J	**1**	**10**

1.2 Ví dụ: thứ tự sản xuất

- Giải pháp

Tham lam theo giá trị:

  - Project B: $220K
  - Project C: $180K
  - Project H: $ 60K
  - Project K: $ 10K

- Tổng thời gian: 26 tuần
- Giá trị: $470 000

Product ID	Completion Time (wks)	Expected Revenue (1000 $)
B	12	220
A	15	210
C	10	180
F	7	170
E	8	160
D	9	120
G	5	90
I	3	60
H	4	40
J	1	10

1.2 Ví dụ: thứ tự sản xuất

Tham lam gần đúng

Tiền/giờ

Product ID	Completion Time (wks)	Expected Revenue (1000 $)	Revenue Density ($ / wk)
A	15	210	**14 000**
B	12	220	**18 333**
C	10	180	**18 000**
D	9	120	**13 333**
E	8	160	**20 000**
F	7	170	**24 286**
G	5	90	**18 000**
H	4	40	**10 000**
I	3	60	**20 000**
J	1	10	**10 000**

1.2 Ví dụ: thứ tự sản xuất

- **Tham lam gần đúng**

theo tỉ lệ giá trị / thời gian

  - Project F: $24 286/wk
  - Project E: $20 000/wk
  - Project I: $20 000/wk
  - Project G: $18 000/wk
  - Project J: $10 000/wk
- Tổng thời gian: 24 tuần
- Giá trị: $490 000

Product ID	Completion Time (wks)	Expected Revenue (1000 $)	Revenue Density ($/wk)
F	7	170	24 286
E	8	160	20 000
I	3	60	20 000
B	12	220	18 333
C	10	180	18 000
G	5	90	18 000
A	15	210	14 000
D	9	120	13 333
H	4	40	10 000
J	1	10	10 000

1.2 Ví dụ: thứ tự sản xuất

- **Duyệt vét cạn**
  - Project C: $180 000
  - Project E: $170 000
  - Project F: $150 000
  - Project J: $ 10 000

- Tổng thời gian: 26 tuần
- Giá trị: $520 000

Product ID	Completion Time (wks)	Expected Revenue (1000 $)	Revenue Density ($/wk)
A	15	210	14 000
B	12	220	18 333
C	10	180	18 000
D	9	120	13 333
E	8	160	20 000
F	7	170	24 286
G	5	90	18 000
H	4	40	10 000
I	3	60	20 000
J	1	10	10 000

1.2 Ví dụ: thứ tự sản xuất

Nhận xét:

- Các giải pháp tham lam chỉ cho giá trị gần tối ưu.

Algorithm	Expected Revenue
Tham lam theo thời gian	$400 000
Tham lam theo giá trị	$470 000
Tham lam theo tỉ lệ giá trị/thời gian	$490 000
Vét cạn	$520 000

1.3 Ví dụ: sắp xếp công việc

Nhận xét:

- Cho tập gồm n công việc, mỗi công việc được biểu diễn bởi cặp thời gian bắt đầu s[i] và thời gian kết thúc f[i] (i=1, 2, .., n).
- Hãy lựa chọn nhiều nhất các công việc có thể thực hiện tuần tự bởi một người mà không xảy ra tranh chấp.
- Mỗi công việc chỉ thực hiện đơn lẻ tại một thời điểm.

Ví dụ:

- Input:
  - Số lượng công việc: 6
  - Thời gian bắt đầu Start [] = { 5, 3, 0, 5, 1, 8}
  - Thời gian kết thúc Finish[] = { 9, 4, 6, 7, 2, 9}
- Output: Ghi ra ID của công việc được lựa chọn theo thứ tự tăng dần
  - Phương án tối ưu ans[] = {2, 4, 5, 6}

1.3 Ví dụ: sắp xếp công việc

- Input:
  - N: số lượng công việc tập A
  - N dòng tiếp, mỗi dòng gồm 2 số nguyên S[i] và F[i].
- Ouput: Danh sách thực thi nhiều nhất.
- Actions:
  - Bước 1: Sắp xếp thứ tự tăng dần của thời gian kết thúc.
  - Bước 2 (Khởi tạo): Lựa chọn công việc đầu tiên làm phương án tối ưu (ans = 1) và loại nó khỏi tập công việc, A = A\{1};
  - Bước 3 (Lặp).
    - Với mỗi công việc j $\in A$
    - if (S[j] >= F[i]) {ans = ans $\cup$ j; i = j; A = A\{i} }
  - Bước 4 (Trả lại kết quả): ans

1.4 Ví dụ: bài toán nối dây

Bài toán:

- Cho N dây với chiều dài khác nhau. Cần phải nối các dây lại với nhau thành một dây.
- Chi phí nối hai dây lại với nhau được tính bằng **tổng độ dài hai dây**.
- Nhiệm vụ của bài toán là tìm cách nối các dây lại với nhau thành một dây sao cho chi phí nối các dây lại với nhau là ít nhất.

Ví dụ:

- Số lượng dây: N = 4. Độ dài dây L[] = {4, 3, 2, 6}
- Chi phí nối dây nhỏ nhất: OPT = 29
- Cách làm:
  - Dây số 3 nối với dây số 2 → 3 dây với độ dài 4, 5, 6.
  - Dây độ dài 4 nối với dây độ dài 5 → 2 dây với độ dài 6, 9.
  - Nối hai dây còn lại 6+9 =15.
  - Tổng chi phí nhỏ nhất là 5 + 9 + 15 = 29.

1.4 Ví dụ: bài toán nối dây

Thuật toán tham lam:

- Chọn ra 2 phần tử nhỏ nhất trong tập L[] rồi thực hiện nối.
- **Trâu bò:** sử dụng 1 vòng FOR để tìm phần tử min và second_min
- **Tối ưu:** sử dụng priority queue để tối ưu thời gian trong O(log n)

1.4 Ví dụ: bài toán nối dây

Thuật toán tham lam:

- Chọn ra 2 phần tử nhỏ nhất trong tập L[] rồi thực hiện nối.
- Actions:
  - Bước 1. Tạo priority queue pq là hàng đợi ưu tiên lưu trữ độ dài N dây.
  - Bước 2 (Lặp):

```
OPT = 0;
while (pq.size>1) {
    First = pq.top; pq.pop();
    Second = pq.top; pq.pop();
    OPT = OPT + First + Second;
    Pq.push(First + Second);
}
```

  - Bước 3(Trả lại kết quả). Return OPT;

1.4 Ví dụ: bài toán nối dây

- Input:
  - Số lượng dây N = 8.
  - Chi phí nối dây L[] = { 9, 7, 12, 8, 6, 5, 14, 4}.
- Output: Chi phí nối dây nhỏ nhất.

Bước	Giá trị First, Second	OPT=?	Trạng thái hàng đợi ưu tiên.
		0	4, 5, 6, 7, 8, 9, 12, 14
1	First=4; Second=5	9	6, 7, 8, 9, 9, 12, 14
2	First=6; Second=7	22	8, 9, 9, 12, 13, 14
3	First=8; Second=9	39	9, 12, 13, 14, 17
4	First=9; Second=12	60	13, 14, 17, 21
5	First=13; Second=14	87	17, 21, 27
6	First=17; Second=21	125	27, 38
7	First=27; Second=38	190	65
OPT = 190			

2. Thuật toán Quy hoạch động

- Khái niệm Quy hoạch động
- Các bài toán áp dụng:
  - Dãy con chung dài nhất
  - Dãy con liên tiếp có tổng lớn nhất
  - Dãy con tăng dài nhất
  - Dãy con có tổng bằng S
  - Xâu con đối xứng dài nhất
  - Tính tổ hợp
  - Phân tích số

2.1 Thuật toán Quy hoạch động

Dynamic programming

Phương pháp quy hoạch động dùng để giải lớp các bài toán thỏa mãn:

- Bài toán lớn cần giải có thể phân rã được thành nhiều bài toán con.
- Sự kết hợp lời giải của các bài toán con cho ta lời giải của bài toán lớn:
  - Bài toán con có lời giải đơn giản → gọi là cơ sở của quy hoạch động.
  - Công thức phối hợp nghiệm của các bài toán con → công thức truy hồi
- Có không gian vật lý lưu trữ lời giải các bài toán con (Bảng phương án của quy hoạch động).
- Quá trình giải quyết từ bài toán cơ sở (bài toán con) để tìm ra lời giải bài toán lớn phải được thực hiện sau hữu hạn bước dựa trên bảng phương án của quy hoạch động.

2.1 Thuật toán Quy hoạch động

Các yếu tố của Quy hoạch động

- Cơ sở của quy hoạch động:
  - Những trường hợp đơn giản có thể tính trực tiếp.
- Cấu trúc con tối ưu:
  - Phương pháp chia nhỏ các bài toán cho đến khi gặp được bài toán cơ sở.
- Tổng hợp:
  - Hệ thức truy hồi tính giá trị tối ưu của hàm mục tiêu của bài toán lớn qua giá trị tối ưu của các bài toán con thành phần.

2.1 Thuật toán Quy hoạch động

- Ưu điểm:
  - Độ phức tạp tốt.

- Nhược điểm:
  - Chỉ áp dụng cho các bài toán tối ưu nhưng không cần biết đầy đủ các phương án tối ưu, chỉ quan tâm đến kết quả tối ưu
  - Bảng phương án sẽ tốn bộ nhớ
  - Cần kĩ thuật truy vết đáp án

2.2 Bài toán con ếch

Bài toán:

- Mỗi bước con ếch có thể nhảy 1 bước, 2 bước hoặc 3 bước.
- Hãy đếm số cách con ếch có thể nhảy tới đích có độ dài N bước?

- Input:
  - 2
  - 5
- Output:
  - 2
  - 13

2.2 Dãy con chung dài nhất

Bài toán:

- Cho hai dãy X = $(x_1,x_2,\ldots,x_n)$ và Y = $(y_1,y_2,\ldots,y_m)$ gồm các số nguyên.
- Cần tìm dãy con chung dài nhất của hai dãy X và Y.

- Bài toán tương tự: Cho hai xâu ký tự X độ dài n và Y độ dài m. Hãy tìm xâu con chung dài nhất.

2.2 Dãy con chung dài nhất

Bài toán con cơ sở:

- $C[0, j] = 0$ với $\forall j = 0..\ n$ và $C[i,0] = 0$ với $\forall i = 0..\ m$.

(là độ dài dãy con chung dài nhất của dãy rỗng với một dãy khác).

Tổng hợp:

Với $i > 0$, $j > 0$. Tính $C[i, j]$. Có hai tình huống:

- Nếu $x_i = y_j$ thì dãy con chung dài nhất của X_i và Y_i = bổ sung x_i vào dãy con chung dài nhất của hai dãy X_{i-1} và Y_{j-1}
- Nếu $x_i \neq y_j$: dãy con chung dài nhất của X_i và Y_j sẽ là dãy con dài hơn trong hai dãy con chung dài nhất của (X_{i-1} và Y_j) và của (X_i và Y_{j-1}).

2.2 Dãy con chung dài nhất

Công thức truy hồi:

- C[i,j] = 0 nếu i = 0 hoặc j = 0
- C[i,j] = C[i-1,j-1] + 1 nếu $x_i = y_j$
- C[i,j] = Max{ C[i-1,j], C[i,j-1] } nếu $x_i \neq y_j$

2.3 Dãy con liên tiếp có tổng lớn nhất

Bài toán:

- Cho dãy A dưới dạng mảng A[1..N] các số nguyên, cả âm và dương.
- Hãy tìm dãy con các phần tử liên tiếp của dãy A có tổng lớn nhất.

- Kết quả: In ra tổng lớn nhất.

- Input:

6

2 -1 3 4 -5 -2

- Output:

8

2.3 Dãy con liên tiếp có tổng lớn nhất

Giải pháp

- A(i) = a[1], …., a[i], i = 1,2,…, n
- S(i) là tổng của dãy con liên tiếp lớn nhất trong dãy a[1]..a[i]

S(i) = a[x] + a[x+1] + … + a[y] với $1 \le x \le y \le i$

- E(i) là tổng của dãy con liên tiếp lớn nhất trong dãy a[1]..a[i] và có chứa chính phần tử a[i].

E[i] = a[x] + a[x+1] + …+ a[i-1] + a[i]

- Xét một trong hai trường hợp:
  - Các dãy con liên tiếp có chứa a[i] => Tổng lớn nhất là E(i)
  - Các dãy con liên tiếp không chứa a[i] => Tổng lớn nhất là S(i-1)
- Tổng hợp: S(i) = max {S(i-1), E(i)}.

2.3 Dãy con liên tiếp có tổng lớn nhất

Giải pháp

- Gọi S(i) là tổng của dãy con lớn nhất trong dãy i phần tử
  - A(i) = a[1], …., a[i], i = 1,2,…, n
  - S(i) = a[x] + a[x+1] + … + a[y] với $1 \le x \le y \le i$
  - S(n) là giá trị cần tìm.

- Bài toán con cơ sở:

  Với i =1 ta có S(1) = a[1].

2.3 Dãy con liên tiếp có tổng lớn nhất

Giải pháp

- E(i) là tổng của dãy con liên tiếp lớn nhất trong dãy a[1]..a[i] và có chứa chính phần tử a[i].
- Công thức tính E(i):
  - Với i = 1: E(i) = a[1];
  - Với i > 1, có hai khả năng:
    - Nếu dãy chứa a[i-1], độ dài lớn nhất có thể là E(i-1)+a[i], xảy ra nếu E(i-1) > 0.
    - Nếu dãy không chứa a[i-1] thì E(i) chỉ có một phần tử duy nhất là a[i]. Xảy ra khi E(i-1) < 0.

  - Tổng hợp: E[i] = max {a[i] , E[i-1] + a[i] } với i > 1.

2.4 Dãy con tăng dài nhất

Bài toán:

- Cho dãy số A có N phần tử, bài toán yêu cầu tìm dãy con dài nhất sao cho phần tử sau của dãy con luôn lớn hơn phần tử trước.
- Dãy con của một dãy số là dãy có được sau khi loại bớt một số phần tử, các phần tử khác giữ nguyên vị trí.
- Dãy con tăng của A là một dãy $A(i_1), A(i_2),..., A(i_k)$ thỏa mãn

$$i_1 < i_2 <...< i_k \quad \text{và} \quad A(i_1) < A(i_2) <...< A(i_k)$$

2.4 Dãy con tăng dài nhất

Ví dụ:

- A = {1 3 2 5 4 6}
- Dãy con tăng dài nhất:
  - {1 2 5 6}
  - {1 3 4 6}
  - …

2.4 Dãy con tăng dài nhất

Giải pháp:

- Gọi F(i) là độ dài dãy con tăng dài nhất kết thúc ở A(i), ta có công thức tính:
  - F(1) = 1
  - F(i) = max{F(j) + 1} với i, j thỏa mãn $1 \le j < i$ và $A(j) < A(i)$
- Kết quả bài toán là max{F(i)}
- Truy vết: yêu cầu in ra một cấu hình thỏa mãn

2.5 Dãy con có tổng bằng S

Bài toán:

- Cho dãy A_1, A_2, ... , A_N. Xác định có hay không một dãy con của dãy đó có tổng bằng S.

Giải pháp:

- Sử dụng bảng phương án F là một ma trận nhị phân:
  - Đặt F[i,t] = 1 nếu có thể tạo ra tổng t từ một dãy con của dãy gồm các phần tử A_1, A_2, ... , A_i.
  - Ngược lại thì F[i,t] = 0.
  - Nếu F[n,S]=1 thì đáp án của bài toán trên là “có”.

2.5 Dãy con có tổng bằng S

Giải pháp:

- Ta có thể tính F[i,t] theo công thức:
  - F[i, t] = 1 nếu F[i−1, t] = 1 → không chọn a[i]
  - hoặc F[i−1, t−a[i]] = 1 → có chọn a[i]

$F[i, t] = 0$ với $t > 1$

$F[i, 0] = 1$

```
for t := 1 to S do
    for i := 1 to n do
        if (t >= a[i] && (F[i-1][t-a[i]] == 1 || F[i-1][t] == 1)) then
            F[i][t]:=1;
```

2.5 Dãy con có tổng bằng S

Giải pháp:

- Ta có thể tính F[i,t] theo công thức:
  - F[i, t] = 1 nếu F[i−1, t] = 1 → không chọn a[i]
  - hoặc F[i−1, t−a[i]] = 1 → có chọn a[i]

Nhận xét:

- Độ phức tạp: O(n*S)
- Để tính dòng thứ i, ta chỉ cần dòng i−1.
- Bảng phương án khi đó chỉ cần 1 mảng 1 chiều L[0..S]

→ giảm chi phí không gian

2.6 Xâu con đối xứng dài nhất

Bài toán:

- Cho một xâu S, độ dài không quá 1000 kí tự. Tìm xâu đối xứng dài nhất là xâu con liên tiếp của S (hay một dãy các kí tự liên tiếp).

Giải pháp:

- Dùng ma trận F[i, j] có ý nghĩa: F[i, j] = true/false nếu đoạn gồm các kí tự từ i đến j của S **có/không** là xâu đối xứng.

2.6 Xâu con đối xứng dài nhất

Giải pháp:

- Dùng ma trận F[i, j] có ý nghĩa: F[i, j] = true/false nếu đoạn gồm các kí tự từ i đến j của S có/không là xâu đối xứng.

Công thức:

- F[i, i] = True: xâu 1 ký tự luôn đối xứng.
- F[i, j] = F[i+1, j-1] nếu $S_i = S_j$.
- F[i, j] = False nếu $S_i \neq S_j$.

2.6 Xâu con đối xứng dài nhất

- Kết quả: Xâu từ i → j dài nhất thỏa mãn F[i,j]=True.
- Độ phức tạp thuật toán là O(N^2).

```
for i := 1 to n do
    F[i, i] := True;
for k := 1 to (n-1) do
    for i := 1 to (n-k) do
        begin
            j := i + k;
            F[i, j] := (F[i+1, j-1]) and (s[i] = s[j]);
        end;
    end;
end;
```

2.7 Tính tổ hợp C(n, k)

Bài toán:

- Tính C(n,k) với n đến 1000.
- Phương pháp quy hoạch động: Áp dụng công thức
- Chú ý: Vì giá trị sẽ rất lớn nên cần chia dư sau mỗi bước tính.
- Trạng thái cơ sở: C(0, i) hay C(i, i) = 1
- Công thức truy hồi: C(i,j) = C(i-1, j-1) + C(i-1, j)

2.8 Bài toán phân tích số

- Cho trước số nguyên N và tập hợp X = $\{x_1, x_2, \ldots, x_K\}$.
- Hãy tìm cách phân tích N thành tổng của các số trong tập X và có ít phần tử nhất có thể.
- Trường hợp 1: X = {1, 2, 5, 10, 20, 50} và N = 72

72 = 50+20+2

- dp[i] là số phần tử ít nhất được sử dụng trong X để có tổng bằng i.

QUESTIONS & ANSWERS