Chương 7

Quy hoạch động

Mục tiêu

- Tổng quan về phương pháp quy hoạch động
- Quá trình giải bài toán bằng quy hoạch động
- Các bài toán điển hình

Nội dung

7.1. Lý thuyết về quy hoạch động

7.2. Bài toán ba lô 1

7.3. Bài toán ba lô 2

7.4. Bài toán dãy con có tổng chia hết cho k

Quy hoạch động (1)

7.1. Lý thuyết về quy hoạch động

7.2. Bài toán ba lô 1

7.3. Bài toán ba lô 2

7.4. Bài toán dãy con có tổng chia hết cho k

7.1. Lý thuyết về quy hoạch động

Bài toán quy hoạch động

- Bài toán quy hoạch là bài toán tối ưu: gồm có một hàm f gọi là hàm mục tiêu hay hàm đánh giá, các hàm g1, g2, … gn cho giá trị logic gọi là hàm ràng buộc.
- Yêu cầu của bài toán là tìm một cấu hình x thỏa mãn tất cả các ràng buộc g1, g2,… gn: gi(x)=TRUE (1<= i<= n) và x là tốt nhất, theo nghĩa không tồn tại một cấu hình y nào khác thỏa mãn các hàm ràng buộc mà f(y) tốt hơn f(x).

7.1. Lý thuyết về quy hoạch động

Tổng quan về phương pháp quy hoạch động

- Quy hoạch động (Dynamic programming) là một kỹ thuật nhằm đơn giản hóa việc tính toán các công thức truy hồi bằng cách lưu trữ toàn bộ hay một phần kết quả tính toán tại mỗi bước với mục đích sử dụng lại.
- Bản chất của quy hoạch động là nhằm thay thế mô hình tính toán “từ trên xuống” (top-down) bằng mô hình tính toán “từ dưới lên” (bottom-up).

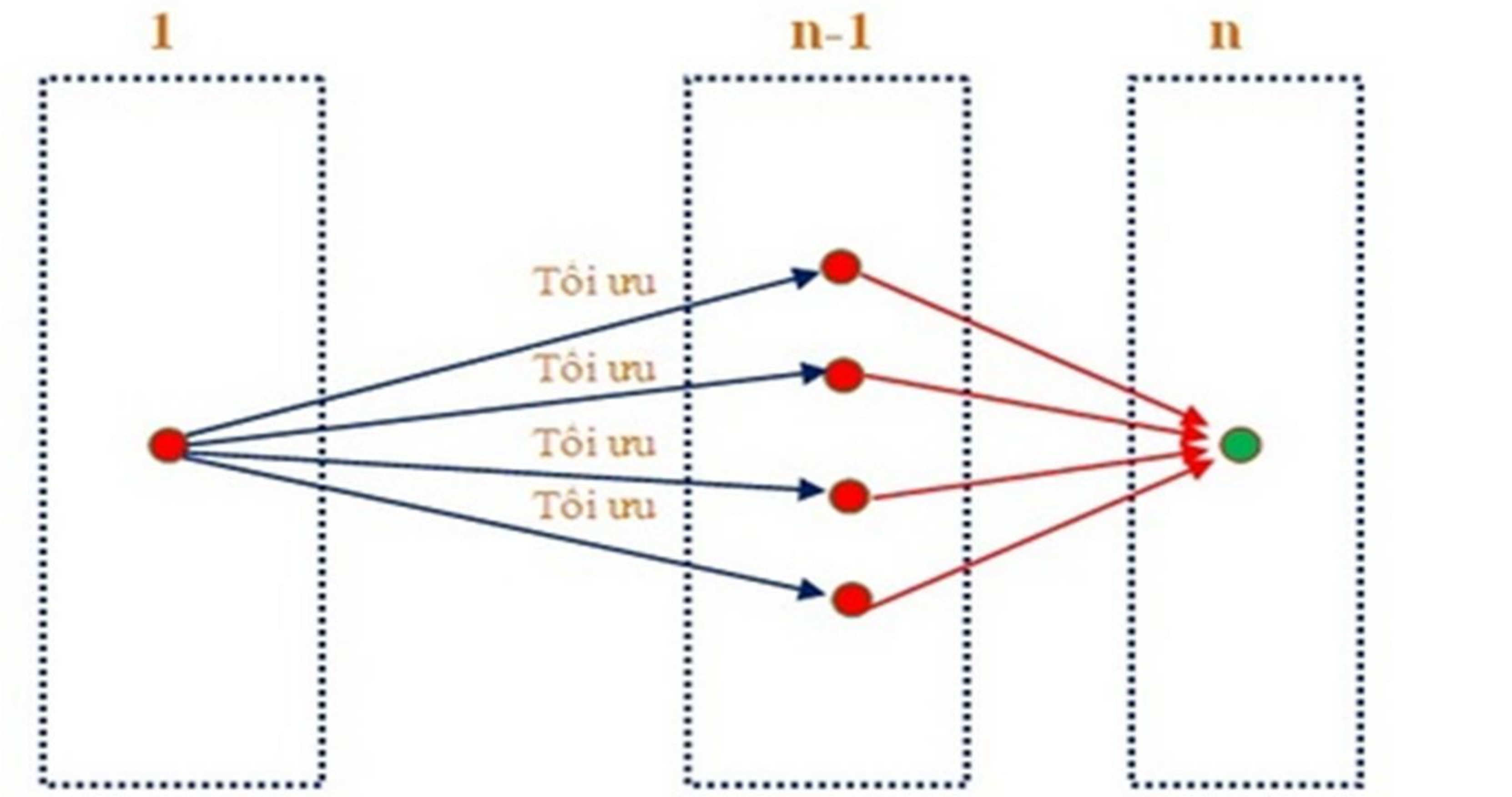
7.1. Lý thuyết về quy hoạch động

Tổng quan về phương pháp quy hoạch động

- Phương pháp quy hoạch động cùng nguyên lý tối ưu được nhà toán học Mỹ R.Bellman đề xuất vào những năm 50 của thế kỷ 20.
- Nguyên lý tối ưu của R.Bellmam được phát biểu như sau: "tối ưu bước thứ **n** bằng cách tối ưu tất cả con đường tiến đến bước **n-1** và chọn con đường có tổng chi phí từ bước **1** đến bước **n-1** và từ **n-1** đến **n** là thấp nhất (nhiều nhất).

7.1. Lý thuyết về quy hoạch động

Tổng quan về phương pháp quy hoạch động



7.1. Lý thuyết về quy hoạch động

Tổng quan về phương pháp quy hoạch động

- Quy hoạch động thường dùng giải các bài toán tối ưu có bản chất đệ qui.
  - Việc tìm nghiệm tối ưu của bài toán đã cho được thực hiện dựa trên việc tìm nghiệm tối ưu của các bài toán con.
  - Kết quả của các bài toán con được ghi nhận lại để phục vụ cho việc giải các bài toán lớn hơn và giải được bài toán yêu cầu.

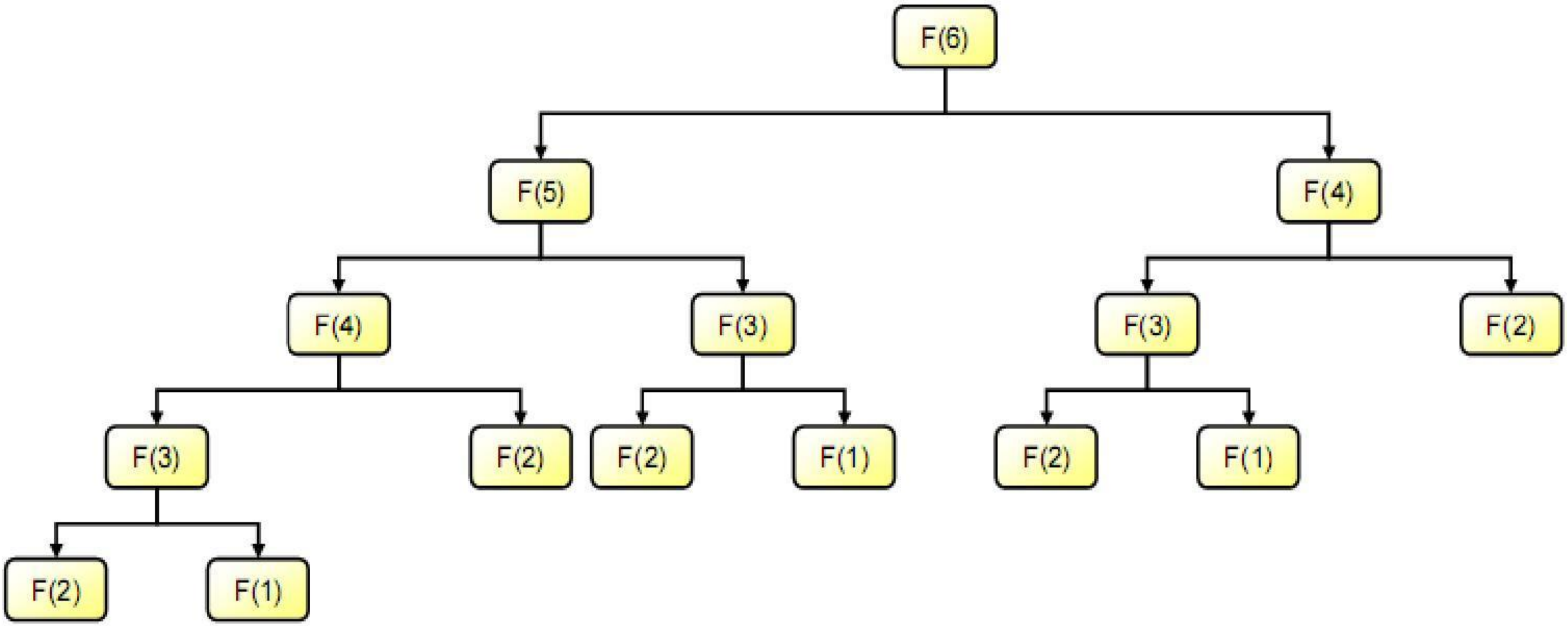
7.1. Lý thuyết về quy hoạch động

Ví dụ minh họa:

Dãy Fibonnaci là dãy các số nguyên dương được định nghĩa như sau:

$$F[i] = \begin{cases} 1 & \text{với } i <= 2 \\ F[i-1] + F[i-2] & \text{với } i > 2 \end{cases}$$

7.1. Lý thuyết về quy hoạch động



Dãy Fibonnaci được tính thông qua công thức đệ quy

7.1. Lý thuyết về quy hoạch động

``int F(int i)```	```int F[100]; //Bang phuong an```

```
int F(int i)
 {
     if (i <= 2) return 1;
     return F(i-2) + F(i-1);
 }
 void main()
 {
      cout<<F(6)<<endl;
 }
```

```
int F[100]; //Bang phuong an
 void main()
 {
   F[1] = F[2] = 1;
   for(int i = 3; i <= 6; i++)
        F[i] = F[i-2] + F[i-1];
   cout << F[6]<<endl;
 }
```

7.1. Lý thuyết về quy hoạch động

Quá trình giải bài toán bằng quy hoạch động

1. **Tìm công thức đệ qui biểu diễn nghiệm tối ưu của bài toán lớn thông qua nghiệm tối ưu của các bài toán con.**
   i. Xác định tham số thể hiện kích thước bài toán.
   ii. Lập công thức tính toán kết quả bài toán theo tham số kích thước.
   iii. Xác định kết quả bài toán con nhỏ nhất.
2. **Tổ chức dữ liệu lưu trữ kết quả tính toán (Bảng phương án)**
3. **Dựa vào kết quả ghi nhận truy vết tìm ra nghiệm tối ưu.**

7.1. Lý thuyết về quy hoạch động

Hạn chế của phương pháp quy hoạch động:

- Việc tìm công thức, phương trình truy toán hoặc tìm cách phân rã bài toán:
  - Nhiều khi đòi hỏi sự phân tích tổng hợp rất công phu, dễ sai sót.
  - Khó nhận ra như thế nào là thích hợp, đòi hỏi nhiều thời gian suy nghĩ.
  - Không phải lúc nào kết hợp lời giải của các bài toán con cũng cho kết quả của bài toán lớn hơn.

7.1. Lý thuyết về quy hoạch động

Hạn chế của phương pháp quy hoạch động:

- Khi bảng lưu trữ đòi hỏi mảng hai, ba chiều … thì khó có thể xử lý dữ liệu với kích cỡ mỗi chiều lớn hàng trăm.
- Có những bài toán không thể giải được bằng quy hoạch động.

Quy hoạch động (1)

7.1. Lý thuyết về quy hoạch động

7.2. Bài toán ba lô 1

7.3. Bài toán ba lô 2

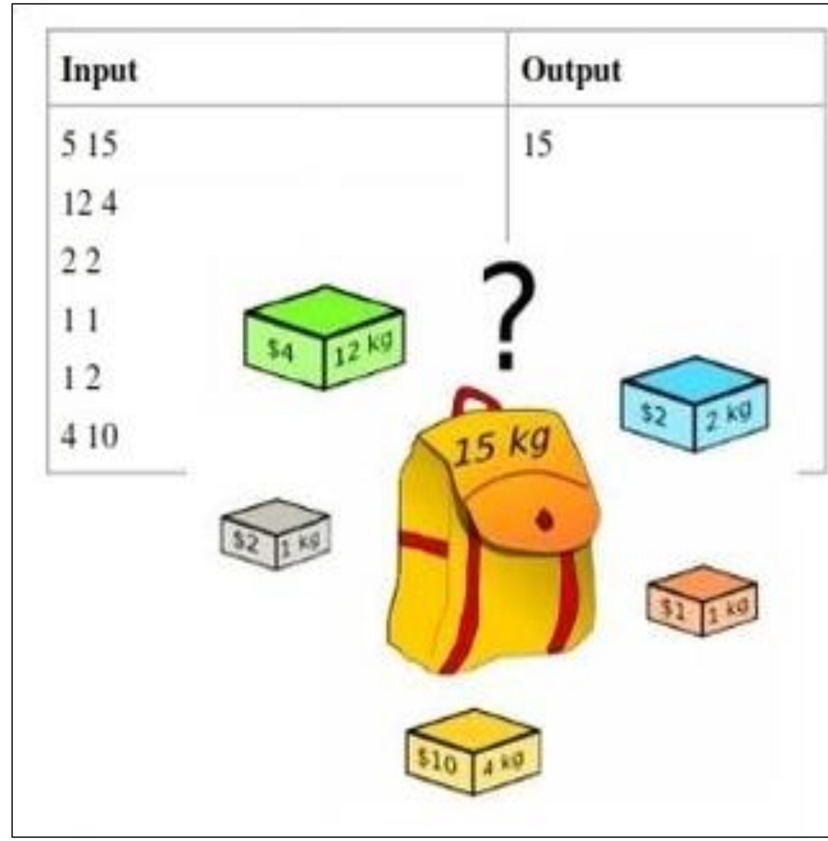
7.4. Bài toán dãy con có tổng chia hết cho k

7.2. Bài toán ba lô 1

Cho **n** gói hàng.

Gói hàng thứ **i** có khối lượng là **A[i]** và giá trị **C[i]**.

Cần chọn những gói hàng nào để bỏ vào một ba lô sao tổng giá trị của các gói hàng đã chọn là **lớn nhất** nhưng tổng khối lượng của chúng **không vượt quá** khối lượng **M** cho trước.



Input	Output
5 15 12 4 2 2 1 1 1 2 4 10	15

Mỗi gói **chỉ chọn 1** hoặc **không chọn**.

7.2. Bài toán ba lô 1

- **Ví dụ: n = 5; M = 13**

  Tức là cho 5 gói hàng. Các gói hàng thứ từ 1 đến 5 lần lượt có khối lượng là **A[i]** và giá trị **C[i]** được cho như trong bảng minh hoạ:

i	1	2	3	4	5
A[i]	3	4	5	2	1
C[i]	4	5	6	3	1

- **Yêu cầu**: Chọn những gói hàng bỏ vào một ba lô sao tổng giá trị của các gói hàng đã chọn là lớn nhất nhưng tổng khối lượng của chúng không vượt quá **M = 13**.
- **Kết quả**: Tổng giá trị của các gói hàng bỏ vào ba lô: **16**

  Các gói được chọn: **1(3, 4) 2(4, 5) 3(5, 6) 5(1, 1)**

7.2. Bài toán ba lô 1

Tham số thể hiện kích thước bài toán

Kết quả bài toán là tổng giá trị lớn nhất của các món hàng được chọn trong n món sao cho tổng khối lượng không lớn hơn **M** cho trước, ký hiệu là **F(n)**

- ❖ Tham số thể hiện kích thước bài toán là số món hàng n
- ❖ Giá trị của **F(n)** có thể được tính từ giá trị của **F(n-1)** cộng thêm hoặc không cộng thêm giá trị của món hàng thứ n nhưng tổng khối lượng không lớn hơn M.

Nếu chọn thêm món hàng thứ n thì tổng khối lượng được chọn trong **(n-1)** món hàng không lớn hơn **(M-A[n])**

- ❖ Suy ra bài toán có 2 tham số: số món hàng và khối lượng giới hạn

7.2. Bài toán ba lô 1

Lập công thức đệ qui

- Gọi **F(i, v)** là tổng giá trị lớn nhất của các gói hàng được chọn trong i gói hàng sao cho tổng khối lượng không lớn hơn **v**.

  - Trường hợp **A[i] > v**: **F(i, v) = F(i -1, v)**

  - Trường hợp **A[i] <= v**:

    \- Nếu gói hàng thứ i không được chọn thì: **F(i, v) = F(i -1, v)**

    \- Nếu gói hàng thứ i được chọn thì:

    F(i, v) = Max{ F(i -1, v); F(i -1, v – A[i]) + C[i] }

- Bài toán nhỏ nhất ứng với **i = 0** ta có: **F(0, v) = 0**

7.2. Bài toán ba lô 1

Xây dựng bảng phương án:

❖ **Cấu trúc bảng phương án:**

Dùng mảng **F [0..n] [0..M]** chứa giá trị của các **F(i, v)**

❖ **Cách tính giá trị trên bảng phương án:**

Điền số **0** cho các ô trên dòng **0**

Sử dụng công thức đệ qui và giá trị trên dòng **(i -1)** để tính dòng **i**

- Trường hợp **A[i] > v**: **F(i, v) = F(i -1, v)**

- Trường hợp **A[i] <= v**:

F(i, v) = Max{ F(i -1, v); F(i -1, v – A[i]) + C[i] }

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 0:

Điền số 0 cho các ô trên dòng 0

C	A	V \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 1:

- Trường hợp **A[1] > v**: **F(1, v) = F(0, v)**
- Trường hợp **A[1] <= v**:

F(1, v) = Max{ F(0, v); F(0, v – A[1]) + C[1]) }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	**1**	0	0	0	4	4	4	4	4	4	4	4	4	4	4

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 2:

- Trường hợp **A[2] > v**: **F(2, v) = F(1, v)**
- Trường hợp **A[2] <= v**:

F(2, v) = Max{ F(1, v); F(1, v – A[2]) + C[2] }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	**1**	0	0	0	4	4	4	4	4	4	4	4	4	4	4
5	4	**2**	0	0	0	4	5	5	5	9	9	9	9	9	9	9

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 3:

- Trường hợp **A[3] > v**: **F(3, v) = F(2, v)**
- Trường hợp **A[3] <= v**:

F(3, v) = Max{ F(2, v); F(2, v – A[3]) + C[3] }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	**1**	0	0	0	4	4	4	4	4	4	4	4	4	4	4
5	4	**2**	0	0	0	4	5	5	5	9	9	9	9	9	9	9
6	5	**3**	0	0	0	4	5	6	6	9	10	11	11	11	15	15

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 4:

- Trường hợp **A[4] > v**: **F(4, v) = F(3, v)**
- Trường hợp **A[4] <= v**:

F(4, v) = Max{ F(3, v); F(3, v – A[4]) + C[4] }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	**1**	0	0	0	4	4	4	4	4	4	4	4	4	4	4
5	4	**2**	0	0	0	4	5	5	5	9	9	9	9	9	9	9
6	5	**3**	0	0	0	4	5	6	6	9	10	11	11	11	15	15
3	2	**4**	0	0	3	4	5	7	8	9	10	12	13	14	15	15

7.2. Bài toán ba lô 1

Ví dụ bảng phương án:

Với i = 5:

- Trường hợp **A[5] > v**: **F(5, v) = F(4, v)**
- Trường hợp **A[5] <= v**:

F(5, v) = Max{ F(4, v); F(4, v – A[5]) + C[5] }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	**1**	0	0	0	4	4	4	4	4	4	4	4	4	4	4
5	4	**2**	0	0	0	4	5	5	5	9	9	9	9	9	9	9
6	5	**3**	0	0	0	4	5	6	6	9	10	11	11	11	15	15
3	2	**4**	0	0	3	4	5	7	8	9	10	12	13	14	15	15
1	1	**5**	0	1	3	4	5	7	8	9	10	12	13	14	15	**16**

7.2. Bài toán ba lô 1

Bảng phương án được hoàn thành với:

- Trường hợp **A[i] > v**: **F(i, v) = F(i -1, v)**

- Trường hợp **A[i] <= v**:

F(i, v) = Max{ F(i -1, v); F(i -1, v – A[i]) + C[i] }

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	1	0	0	0	4	4	4	4	4	4	4	4	4	4	4
5	4	2	0	0	0	4	5	5	5	9	9	9	9	9	9	9
6	5	3	0	0	0	4	5	6	6	9	10	11	11	11	15	15
3	2	4	0	0	3	4	5	7	8	9	10	12	13	14	15	15
1	1	5	0	1	3	4	5	7	8	9	10	12	13	14	15	**16**

7.2. Bài toán ba lô 1

Thuật toán tạo bảng phương án

```
void TaoBangPhuongAn(F [0..n] [0..M])
{
    for (v=0; v <= M; v++) F[0, v] = 0; // Điền số 0 cho dòng
                                            0 của bảng
    for (i = 1; i <= n; i++)
        for (v=0; v <= M; v++)
        {
            F[i, v] = F[i-1, v];
            if (A[i] <= v && F[i, v] < F[i-1, v - A[i] ] + C[i])
                F[i, v] = F[ i-1, v - A[i] ] + C[i];
        }
}
```

7.2. Bài toán ba lô 1

Truy vết tìm lại các gói hàng đã chọn:

Bắt đầu từ ô **F[n, M]** trên dòng n ta dò ngược về dòng **0** theo nguyên tắc:

- Nếu **F[i, v] <> F[i–1, v]** thì gói thứ **i** được chọn, ta truy tiếp ô **F[i-1, v-A[i]]**.
- Nếu **F[i, v] = F[i–1, v]** thì gói thứ **i** không được chọn, ta truy tiếp ô **F[i-1, v]**.

C	A	v \ i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
		0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0
4	3	1	0	0	0	**4**	4	4	4	4	4	4	4	4	4	4
5	4	2	0	0	0	4	5	5	5	**9**	9	9	9	9	9	9
6	5	3	0	0	0	4	5	6	6	9	10	11	11	11	**15**	15
3	2	4	0	0	3	4	5	7	8	9	10	12	13	14	**15**	15
1	1	5	0	1	3	4	5	7	8	9	10	12	13	14	15	**16**

7.2. Bài toán ba lô 1

Thuật toán truy vết tìm lại các gói hàng đã chọn

```
void TruyVet(F [0..n] [0..M])
{   Bắt đầu từ ô F[n, M] trên dòng n: i = n; v = M;
    for (; i > 0; i --)
        if (F[i, v] != F[i-1, v])
        {
            <Món hàng thứ i được chọn >;
            v = v – A[i];
        }
}
```

Tổng kết

Lý thuyết về quy hoạch động

- Bài toán quy hoạch động
- Tổng quan về quy hoạch động
- Quá trình giải bài toán bằng quy hoạch động
- Hạn chế của phương pháp quy hoạch động

Bài toán ba lô 1

- Tham số thể hiện kích thước bài toán
- Lập công thức đệ quy
- Xây dựng bảng phương án lựa chọn các gói hàng → p/ án tối ưu
- Truy vết tìm lại các gói hàng đã chọn

Bài tập

1. Cho một balo có khối lượng M= 11.

   Có 4 đồ vật lần lượt như sau: khối lượng A[i]={4,2,4,3} với giá trị tương ứng C[i]={1,5,2,7}.

   Hãy chọn các đồ vật để cho vào balo sao cho các đồ vật được chọn có giá trị lớn nhất và khối lượng không vượt quá khối lượng của balo. Biết rằng mỗi đồ vật có thể được chọn 1 lần hoặc không chọn.

Tài liệu tham khảo

[1]. Giáo trình Cấu trúc dữ liệu và giải thuật – Lê Văn Vinh, NXB Đại học quốc gia TP HCM, 2013

[2]. Cấu trúc dữ liệu & thuật toán, Đỗ Xuân Lôi, NXB Đại học quốc gia Hà Nội, 2010.

[3]. Trần Thông Quế, *Cấu trúc dữ liệu và thuật toán (phân tích và cài đặt trên C/C++)*, NXB Thông tin và truyền thông, 2018

[4]. Robert Sedgewick, *Cẩm nang thuật toán*, NXB Khoa học kỹ thuật, 2004 .

[5]. PGS.TS Hoàng Nghĩa Tý, *Cấu trúc dữ liệu và thuật toán*, NXB xây dựng, 2014