THUẬT TOÁN ỨNG DỤNG

Quy hoạch động

(Dynamic programming)

Nội dung

1. Ý tưởng quy hoạch động
2. Bài toán đoạn con lớn nhất
3. Bài toán dãy con chung dài nhất
4. Bài toán đếm số dãy con có tổng cho trước
5. Bài toán xếp ba lô
6. Phân tích về quy hoạch động
7. Bài tập

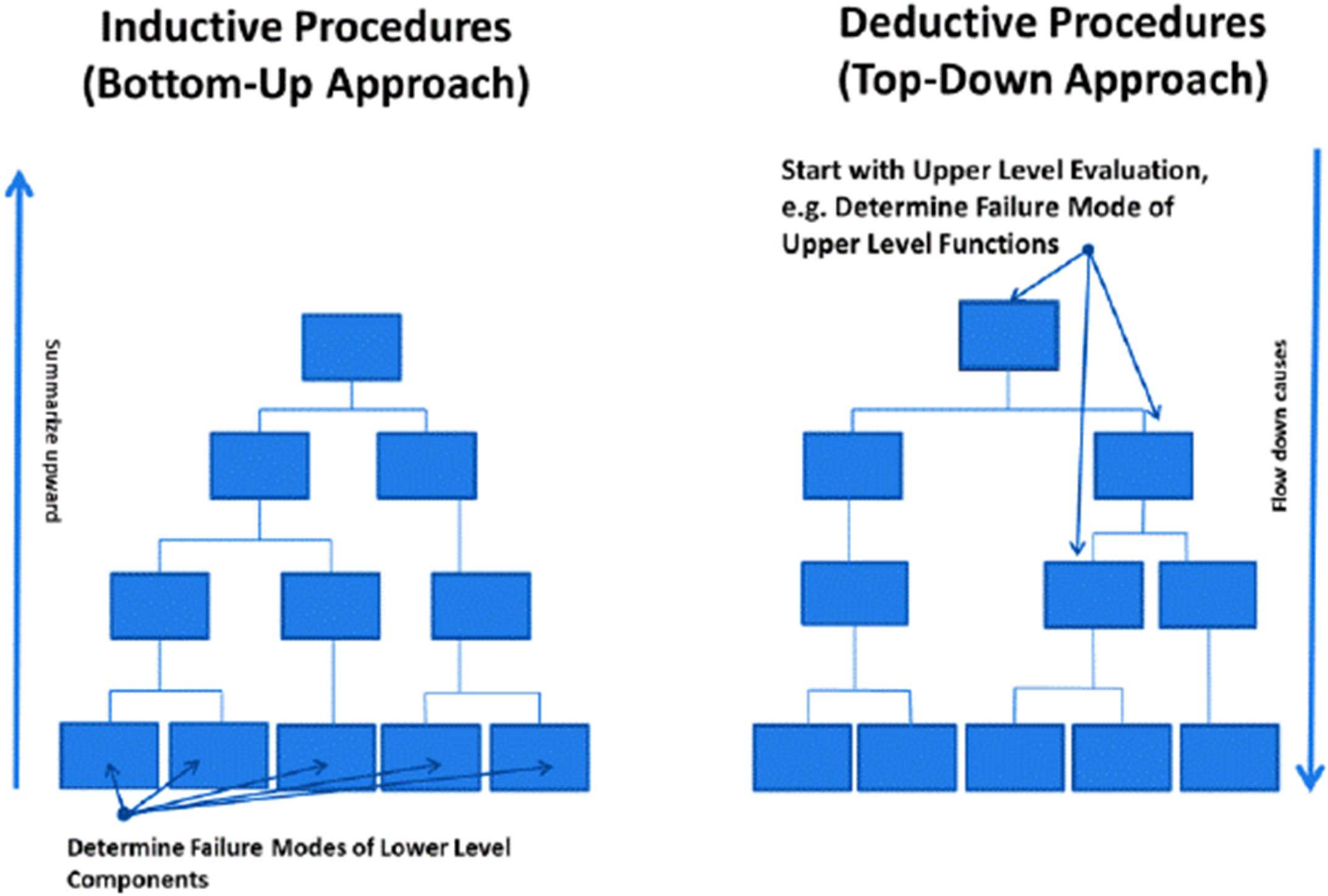
Phần 1

Ý tưởng quy hoạch động

Quy hoạch động

- Dynamic programming (Richard Bellman, 1953)
- Thường dùng cho các bài toán tối ưu
- Nguyên tắc: lời giải tối ưu của bài toán lớn sử dụng kết quả tối ưu của bài toán con

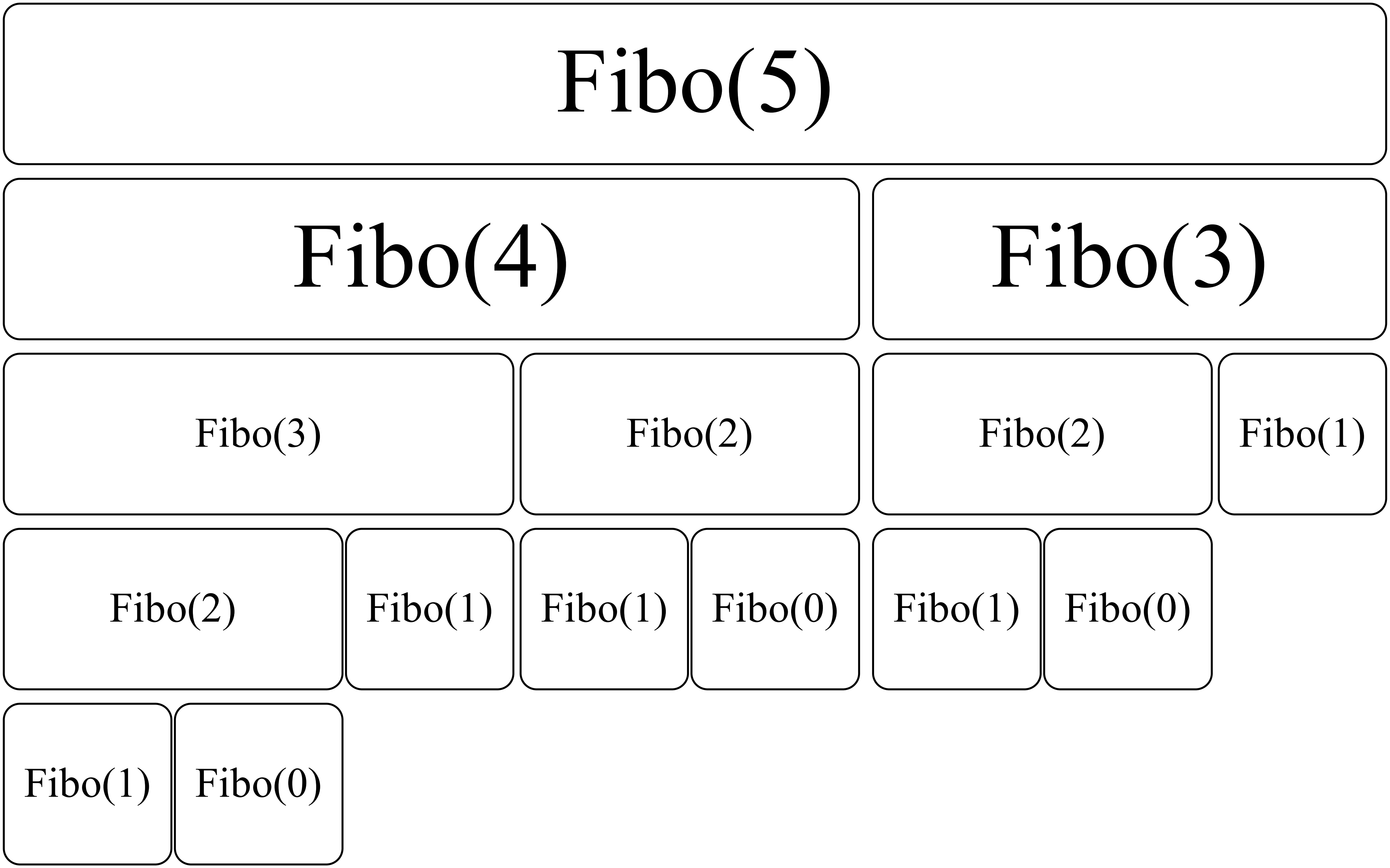
Top-down vs Bottom-up



Top-down vs Bottom-up

- Top-down:
  - Nhìn theo hướng từ trên xuống dưới
  - Chia bài toán lớn thành các bài toán nhỏ
  - Tiếp cận chia để trị
- Bottom-up:
  - Nhìn theo hướng từ dưới lên trên
  - Giải bài toán nhỏ trước
  - Tổ hợp các lời giải nhỏ thành lời giải của bài toán lớn

Top-down


Fibo(5)
Fibo(4)
Fibo(3)
Fibo(3)
Fibo(2)
Fibo(2)
Fibo(1)
Fibo(2)
Fibo(1)
Fibo(1)
Fibo(0)
Fibo(1)
Fibo(0)
Fibo(1)
Fibo(0)

Bottom-up

Fibo(1) Fibo(0)

Fibo(2) Fibo(1) ~~Fibo(1)~~ ~~Fibo(0)~~ ~~Fibo(1)~~ ~~Fibo(0)~~

Fibo(3) Fibo(2) ~~Fibo(2)~~ ~~Fibo(1)~~

Fibo(4) Fibo(3)

Fibo(5)

Quy hoạch động

- Khi nào thì chúng ta cần đến quy hoạch động?
  - Đó là một câu hỏi rất khó trả lời.
  - Không có một công thức nào cho các bài toán như vậy.
- Có một số tính chất của bài toán mà có thể nghĩ đến quy hoạch động:
  - *Bài toán có các bài toán con gối nhau*
  - *Bài toán có cấu trúc con tối ưu*
- Khi giới hạn về thời gian, cũng như bộ nhớ của chương trình, nên một thuật toán hiệu quả là cực kỳ cần thiết. Quy hoạch động là 1 thuật toán hiệu quả.

Quy hoạch động

- Là một kĩ thuật thiết kế thuật toán theo kiểu chia bài toán lớn thành các bài toán con, sử dụng lời giải của các bài toán con để tìm lời giải cho bài toán ban đầu.
- Khác với chia để trị, quy hoạch động, thay vì gọi đệ quy, sẽ tính trước lời giải của các bài toán con và lưu vào bộ nhớ (thường là một mảng), và sau đó lấy lời giải của bài toán con ở trong mảng đã tính trước để giải bài toán lớn

Phần 2

Bài toán đoạn con lớn nhất

Bài toán đoạn con lớn nhất

- Đã giới thiệu từ ngay buổi học đầu tiên
- Cho dãy A = (a_1, a_2,... a_{n-1}, a_n), tìm đoạn con (dãy con liên tiếp) trong A có tổng các phần tử là lớn nhất
- Giải:
  - Đặt S_i là tổng lớn nhất của đoạn con kết thúc tại a_i
  - Kết quả cần tìm = max(S_1, S_2,... S_{n-1}, S_n)
  - Tính S_k:
    - $S_1 = a_1$
    - $S_k = \begin{cases} a_k & n\acute{\hat{e}}u\ S_{k-1} \le 0 \\ a_k + S_{k-1} & n\acute{\hat{e}}u\ S_{k-1} > 0 \end{cases}$
- Quy hoạch động: tính giá trị S_k sử dụng kết quả tính S_{k-1}
- Cài đặt: dễ

Phần 3

Bài toán dãy con chung dài nhất

Bài toán dãy con chung dài nhất

- Longest common subsequence (LCS)
- Cho 2 dãy $A = (a_1, a_2, \ldots a_{m-1}, a_m)$ và $B = (b_1, b_2, \ldots b_{n-1}, b_n)$
- Dãy con = dãy được lập ra từ dãy cha bằng cách chọn lấy một số phần tử, giữ nguyên thứ tự
  - Không nhất thiết phải liên tiếp
  - Có thể không chứa phần tử nào
- Dãy con chung của A và B: là dãy con của cả A và B
- Cần tìm: dãy con có nhiều phần tử nhất (dài nhất)
- Ví dụ:
  - A = (3, 1, 2, 0, 4, 3) B = (1, 2, 3, 4, 3, 2, 1)
  - KQ = (1, 2, 4, 3)

Bài toán dãy con chung dài nhất

- Hàm S(p, q) trả về độ dài của dãy con chung dài nhất của $A_p = (a_1, a_2, \ldots a_{p-1}, a_p)$ và $B_q = (b_1, b_2, \ldots b_{q-1}, b_q)$
- Như vậy việc của chúng ta là tính S(m, n)
- Công thức tính S(p, q) như thế nào?

$$S(p,q) = \begin{cases} 0 & \textit{nếu } p = 0 \textit{ hoặc } q = 0 \\ S(p-1, q-1) + 1 & \textit{nếu } a_p = b_q \\ \max\{S(p-1, q), S(p, q-1)\} & \textit{nếu } a_p \neq b_q \end{cases}$$

- Hai cách tính:
  - Top-down: tính từ S(m, n) trở đi, chia nhỏ dần bài toán
  - Bottom-up: tính từ nhỏ tăng dần kích cỡ cho đến S(m, n)
- Sử dụng bộ nhớ để lưu lại các giá trị đã tính toán

Phần 4

Bài toán đếm số dãy con có tổng cho trước

Đếm số dãy con có tổng cho trước

- Bài của buổi trước, giờ hãy thử giải nó bằng kĩ thuật quy hoạch động
- Cho số nguyên S và dãy A = (a_1, a_2,... a_{n-1}, a_n).
- Hãy đếm xem có bao nhiêu dãy con của A có tổng các phần tử đúng bằng S
- Ví dụ:
  - S = 7
  - A = (1, 7, 6, 3, 3)
  - Kết quả: 3 dãy
    - 7 = 1 + 3 + 3
    - 7 = 1 + 6
    - 7 = 7

Đếm số dãy con có tổng cho trước

- Hàm F(S, n) = số dãy con của A có tổng đúng bằng S
- Có hai loại dãy:
  - Dãy con không chứa a_n:
    - Đếm số dãy con của A = (a_1, a_2,... a_{n-2}, a_{n-1}) có tổng bằng S
    - Chính là F(S, n-1)
  - Dãy con có chứa a_n:
    - Đếm số dãy con của A = (a_1, a_2,... a_{n-2}, a_{n-1}) có tổng bằng S-a_n
    - Chính là F(S-a_n, n-1)
- Suy ra: F(S, n) = F(S, n-1) + F(S-a_n, n-1)
- Sử dụng bộ nhớ để lưu lại các kết quả đã tính toán
- Tạm thời hạn chế $a_i > 0$, lời giải tổng quát các bạn tự tìm hiểu như là bài tập

Phần 5

Bài toán xếp ba lô

Bài toán xếp ba lô

- Bài toán cái túi, knapsack problem,...
- Có N đồ vật, đồ vật thứ i có trọng lượng a_i và giá trị b_i. Hãy chọn ra một số đồ vật có tổng trọng lượng tối đa là W và có tổng giá trị lớn nhất.
- Giải thiết các tham số đều nguyên dương:
  - $A = (a_1, a_2, \ldots a_{N-1}, a_N)$
  - $B = (b_1, b_2, \ldots b_{N-1}, b_N)$
  - W
- Hàm f(k, h) là phương án tối ưu (tổng giá trị lớn nhất) trong trường hợp sử dụng k đồ vật đầu tiên và giới hạn tổng trọng lượng là h
- Như vậy ta cần tính f(N, W)

Bài toán xếp ba lô

- Hàm f(k, h) là phương án tối ưu (tổng giá trị lớn nhất) trong trường hợp sử dụng k đồ vật đầu tiên và giới hạn tổng trọng lượng là h
- Ở phương án tối ưu của f(k, h) có 2 tình huống xảy ra:
  - Có sử dụng độ vật thứ k*: $f(k, h) = f(k-1, h-a_k) + b_k$
  - Không sử dụng độ vật thứ k: $f(k, h) = f(k-1, h)$
- Như vậy $f(k, h) = \max \{ f(k-1, h), f(k-1, h-a_k) + b_k \}$
- Triển khai:
  - Top-down: viết đệ quy từ trên xuống
  - Bottom-up: tính từ dưới lên

Phần 6

Phân tích về quy hoạch động

Tóm lược về quy hoạch động

- Có 2 nguyên tắc cơ bản:
  - Phương án tối ưu của bài toán lớn dựa trên kết quả tối ưu của từng bài toán con
  - Sử dụng bộ nhớ để lưu lại kết quả tính toán, tránh phải tính lại
- Cài đặt:
  - Top-down: đệ quy có nhớ, tính từ bài toán lớn giảm dần xuống
  - Bottom-up: vòng lặp, tính từ bài toán nhỏ tăng dần lên
- Thường có 2 loại bài toán:
  - Bài toán đếm (tìm số lượng cấu hình)
  - Bài toán tối ưu (tìm cấu hình min, max, max của min, min của max,...)

Ưu điểm của quy hoạch động

- Nhanh
- Viết mã đơn giản
- Viết đệ quy thích hợp với tư duy top-down nhưng thường chạy chậm hơn
- Bottom-up chạy nhanh hơn nhưng đôi khi tính thừa không cần thiết
- Đánh đổi bộ nhớ lấy tốc độ

Nhược điểm của quy hoạch động

- Hầu hết các vấn đề giải được bằng quy hoạch động là bài giải bằng chia để trị
- Nhưng không phải bài chia để trị nào cũng giải được bằng quy hoạch động
- Nếu số bài toán con tăng quá nhanh, quy hoạch động sẽ không khả thi
- Thích hợp với xử lý số nguyên hơn là số thực
- Đòi hỏi mọi bài toán con phải được giải tối ưu

Phần 7

Bài tập

Bài tập

1. *Cho dãy số nguyên A = (a_1, a_2,... a_{n-1}, a_n). Hãy tìm dãy con không giảm dài nhất của A.*
   - *Dãy con mà phần tử đứng sau không bé hơn phần tử đứng trước*
   - *Nhiều phần tử nhất*
2. *Cho 2 xâu ký tự A và B. Được phép thực hiện các thao tác sau trên xâu A:*
   - Chèn một kí tự bất kì vào vị trí nào đó
   - Xóa một kí tự ở vị trí bất kì
   - Thay thế một kí tự ở vị trí nào đó bằng một kí tự khác
   Tính số thao tác ít nhất để biến đổi từ A thành B.

Bài tập

3. *Một lưới ô vuông M x N, trên mỗi ô vuông có điền một giá trị nguyên là chi phí phải trả để có thể đi qua ô đó. Một robot di chuyển xuyên qua lưới ô từ trên xuống dưới. Robot có thể bắt đầu ở bất kì ô nào của dòng đầu tiên, sau đó chỉ có thể đi xuống một trong các ô ở dòng dưới chung cạnh hoặc đỉnh với ô hiện tại. Tìm phương án di chuyển có chi phí tối thiểu.*

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