Đại học quốc gia TP.HCM
Trường đại học công nghệ thông tin

Ngành ...

Môn học: ...

Tiêu đề Bài tập CS112 - Nhóm 9

Nhóm 8:
Tên SV1: Nguyễn Phạm Phương Nam – MSSV: 23520978, Tên SV2: Phạm Huỳnh Long vũ – MSSV: 23520978,

Giảng viên:
Tên Giảng Viên: Nguyên Thanh Sơn

Giải bài tập nhóm 9: Quy hoạch động

Phần 1: Bài tập lý thuyết

Câu 1: Có phải mọi bài toán đều có thể giải quyết bằng quy hoạch động không? Tại sao?

Không, không phải mọi bài toán đều giải được bằng quy hoạch động. Quy hoạch động yêu cầu bài toán phải thỏa mãn hai tính chất chính:

1. **Tính chất con tối ưu (Optimal Substructure):** Nghiệm tối ưu của bài toán lớn có thể xây dựng từ nghiệm tối ưu của các bài toán con.
2. **Tính lặp lại (Overlapping Subproblems):** Các bài toán con lặp lại nhiều lần.

Nếu bài toán không có hai tính chất trên, thì quy hoạch động không áp dụng được.

Câu 2: Trong thực tế, bạn đã gặp bài toán nào có thể áp dụng quy hoạch động? Hãy chia sẻ cách tiếp cận.

Ví dụ: Bài toán "Con ếch nhảy qua các phiến đá" (Frog Problem):

- **Ý tưởng:** Tính chi phí tối thiểu để ếch nhảy từ phiến đá đầu tiên đến phiến đá cuối cùng.
- **Cách tiếp cận:**
    1. Xây dựng công thức truy hồi để tính chi phí nhảy qua từng phiến đá.
    2. Sử dụng mảng động để lưu trữ kết quả từng bước.

Câu 3: Phân tích ưu, nhược điểm của hai phương pháp Top-down và Bottom-up.

- **Top-down:**
  - **Ưu điểm:**
    - Dễ cài đặt, trực quan nhờ đệ quy.
  - **Nhược điểm:**
    - Có thể gây tốn bộ nhớ stack (gây lỗi tràn stack nếu bài toán quá lớn).
    - Cần sử dụng kỹ thuật memoization để giảm tính toán thừa.
- **Bottom-up:**
  - **Ưu điểm:**
    - Không gây tràn stack, tối ưu hơn về mặt bộ nhớ.
  - **Nhược điểm:**
    - Có thể khó hình dung hơn trong những bài toán phức tạp.

Lựa chọn ưu tiên: Phương pháp Bottom-up thường được ưu tiên hơn vì hiệu quả bộ nhớ và tránh rủi ro khi làm việc với bài toán lớn.

Phần 2: Bài tập thực hành

Bài tập 1: Bài toán Frog Problem

Đề bài: Tính chi phí tối thiểu để ếch nhảy qua các phiến đá. Giả sử có n phiến đá, ếch bắt đầu từ phiến đầu tiên. Với mỗi phiến đá i, ếch có thể nhảy tới phiến $i+1$ hoặc $i+2$, chi phí là hiệu độ cao giữa hai phiến đá.

Ý tưởng

- Định nghĩa $dp[i]$ là chi phí tối thiểu để nhảy tới phiến i.
- Công thức truy hồi:

$$dp[i] = \min(dp[i-1] + |h[i] - h[i-1]|, dp[i-2] + |h[i] - h[i-2]|)$$

- Điều kiện ban đầu:

$$dp[0] = 0,\ dp[1] = |h[1] - h[0]|$$

Mã giả

```
function minCostJump(h: array of int):
    n = length(h)
    dp = array of size n
    dp[0] = 0
    dp[1] = abs(h[1] - h[0])
    for i = 2 to n-1:
        dp[i] = min(dp[i-1] + abs(h[i] - h[i-1]), dp[i-2] + abs(h[i] - h[i-2]))
    return dp[n-1]
```

Độ phức tạp

- **Thời gian:** $O(n)$
- **Không gian:** $O(n)$ (có thể tối ưu xuống $O(1)$).