ĐẠI HỌC QUỐC GIA TPHCM

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH

BỘ MÔN CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Báo cáo

Đề tài: Giải bài toán Set Cover và TSP bằng thuật toán gần đúng

Môn học: Phân tích và thiết kế thuật toán

Sinh viên thực hiện:

Nguyen Minh Huy(23520634)

Do Quang Luc(23520902)

Giáo viên hướng dẫn:

Nguyen Thanh Son

Ngày 5 tháng 12 năm 2024

Đề bài

Hằng năm ở thành phố X, tổ chức một cuộc thi đối kháng hai người. Người làm $p = 0$ trước là người thắng. Quy tắc chơi:

- p lẻ: tăng hoặc giảm 1.
- p chẵn: giảm p xuống một nửa.

Người chơi A luôn đi trước B. Nếu cả hai đều chơi tối ưu, liệu A có luôn thắng không?

Phân tích bài toán

Trò chơi thuộc loại **trò chơi có tổng không đổi**.

Giới hạn $p \leq 10^6$

Ý tưởng: Sử dụng quy hoạch động. Với mỗi p, tính trạng thái thắng/thua $dp[p]$.

Mã giả:

```
function solve(p):
    dp = array of size (p+1) initialized to 0
    dp[0] = 0  # Người làm p = 0 thua
    for i = 1 to p:
        if i % 2 == 0:
            dp[i] = 1 - dp[i/2]
        else:
            dp[i] = max(1 - dp[i-1], 1 - dp[i+1])
    return dp[p]
```

Giới hạn $p \leq 10^{18}$

Ý tưởng: Dựa vào tính chất:

- p chẵn: bắt buộc giảm.
- p lẻ: người chơi có thể ép đối thủ thua.

Đề bài

Hai người chơi lần lượt bốc số đồng xu từ một chồng gồm n đồng xu. Trong mỗi lượt, người chơi được chọn bốc từ 1 đến k đồng xu ($k \leq n$). Người không thể bốc được là người thua.
Yêu cầu là tìm các giá trị k sao cho người chơi A luôn thắng khi cả hai đều chơi tối ưu.
—

Phân tích bài toán

Trò chơi này thuộc loại **trò chơi có tổng không đổi**. Các trạng thái của trò chơi có thể được xác định là **trạng thái thắng** hoặc **trạng thái thua**:

- **Trạng thái thua:** Nếu người chơi hiện tại không thể thực hiện nước đi nào để đưa đối thủ vào trạng thái thua.
- **Trạng thái thắng:** Nếu người chơi hiện tại có thể thực hiện ít nhất một nước đi đưa đối thủ vào trạng thái thua.

—

1 Giải pháp

1.1 Giới hạn $n \leq 1000$

Ý tưởng

Sử dụng quy hoạch động (*Dynamic Programming*) để tính trạng thái thắng/thua của mỗi n. Gọi $dp[i]$ là trạng thái của i đồng xu:

$$dp[i] = \begin{cases} 1, & \text{nếu tồn tại } x \in [1, k] \text{ sao cho } dp[i-x] = 0, \\ 0, & \text{nếu không tồn tại.} \end{cases}$$

Mã giả

```
function solve(n, k):
    dp = array of size (n+1) initialized to 0
    dp[0] = 0  # Kh ng c n   ng   xu => n g  Íi  ch i  thua
    for i = 1 to n:
        for x = 1 to k:
            if i >= x and dp[i-x] == 0:
                dp[i] = 1
                break
    return dp
```

Listing 1: Quy hoạch động với $n \leq 1000$

Độ phức tạp

- Thời gian: $O(n \cdot k)$ - Không gian: $O(n)$

—

1.2 Giới hạn $n \leq 10^{18}$

Ý tưởng

Khi n rất lớn, không thể tính và lưu trữ toàn bộ trạng thái. Chúng ta áp dụng tính chất chu kỳ:

- Nếu n là trạng thái thua ($dp[n] = 0$), thì tất cả các trạng thái $n + x$ ($x \in [1, k]$) sẽ là trạng thái thắng.
- n là trạng thái thua nếu $n \mod (k+1) = 0$.

Mã giả

```
function solve_large_n(n, k):
    if n % (k + 1) == 0:
        return "LOSE"
    else:
        return "WIN"
```

Listing 2: Giải quyết với $n \leq 10^{18}$

Độ phức tạp

- Thời gian: $O(1)$ - Không gian: $O(1)$

—

Kết luận

- Với $n \leq 1000$: Sử dụng quy hoạch động để xác định trạng thái thắng/thua cho từng giá trị n.
- Với $n \leq 10^{18}$: Dựa vào tính chất chu kỳ modulo để tính kết quả nhanh chóng mà không cần lưu trữ nhiều.