

# Cut the Rope - Lời Giải

Nhóm 3

November 2025

## 1 Mô tả bài toán

Bạn đang chơi một trò chơi với  $N$  cấp độ ( $1 \leq N \leq 3 \cdot 10^5$ ). Bạn cần thu thập ít nhất  $W$  ngôi sao ( $1 \leq W \leq 2N$ ) để mở hộp tiếp theo. Mỗi cấp độ  $i$  có hai thuộc tính:

- $a_i$ : Thời gian để hoàn thành cấp độ đạt 1 sao.
- $b_i$ : Thời gian để hoàn thành cấp độ đạt 2 sao (với  $a_i < b_i$ ).

Mỗi cấp độ chỉ có thể chơi một lần (lấy 1 sao, lấy 2 sao hoặc không chơi).

**Yêu cầu:** Tìm thời gian tối thiểu để đạt được ít nhất  $W$  sao và in ra cấu hình chơi (số sao đạt được ở mỗi cấp độ).

- **Input:**  $N, W$ .  $N$  dòng tiếp theo chứa cặp  $(a_i, b_i)$ .
- **Output:** Thời gian tối thiểu và chuỗi trạng thái của các cấp độ (0, 1 hoặc 2).

## 2 Mô tả thuật toán

Thuật toán lựa chọn: Tham lam (Greedy)

### 2.1 Ý tưởng tham lam

Quan sát quan trọng để giải quyết bài toán là sắp xếp các cấp độ dựa trên thời gian hoàn thành 2 sao ( $b_i$ ). Giả sử ta có một lời giải tối ưu. Nếu ta chọn một cấp độ  $k$  để lấy 2 sao, thì tất cả các cấp độ có  $b_i < b_k$  nên được chọn để lấy **ít nhất 1 sao**. Lý do: Nếu có một cấp độ  $j$  ( $b_j < b_k$ ) mà ta chưa chơi (0 sao), ta hoàn toàn có thể chuyển việc lấy 2 sao từ  $k$  sang  $j$  (hoặc lấy 1 sao ở  $j$ ) để giảm tổng thời gian mà vẫn giữ nguyên số lượng sao.

Do đó, ta có thể duyệt qua một biến  $L$  từ 0 đến  $N$  (với các cấp độ đã sắp xếp tăng dần theo  $b_i$ ):

- Giả sử  $L$  cấp độ đầu tiên chắc chắn được chơi ít nhất 1 sao. Tổng thời gian cơ bản là  $\sum_{i=1}^L a_i$ , số sao hiện có là  $L$ .

- Ta cần thêm  $W - L$  sao nữa. Các sao này có thể lấy từ:
  1. Nâng cấp một cấp độ  $i$  (trong nhóm  $1 \dots L$ ) từ 1 sao lên 2 sao với chi phí  $b_i - a_i$ .
  2. Chơi mới một cấp độ  $j$  (trong nhóm  $L + 1 \dots N$ ) để lấy 1 sao với chi phí  $a_j$ .
- Tại mỗi bước  $L$ , ta cần chọn  $W - L$  giá trị nhỏ nhất từ tập hợp các chi phí khả dụng trên.

## 2.2 Tối ưu hóa bằng Cấu trúc dữ liệu

Vì  $N$  lên tới  $3 \cdot 10^5$ , ta không thể sắp xếp lại tập chi phí ở mỗi bước  $L$ . Ta sử dụng **Fenwick Tree (BIT)** kết hợp với **Rời rạc hóa (Coordinate Compression)**:

1. Thu thập tất cả các giá trị  $a_i$  và  $b_i - a_i$ , sau đó rời rạc hóa chúng về miền giá trị nhỏ  $[1, 2N]$ .
2. Sử dụng 2 mảng BIT:
  - **bit\_cnt**: Đếm số lượng phần tử có giá trị  $v$ .
  - **bit\_sum**: Tính tổng giá trị của các phần tử có giá trị  $v$ .
3. Khi  $L$  tăng lên, một phần tử chuyển từ tập "bên ngoài" (chi phí  $a_L$ ) vào tập "bên trong" (chi phí nâng cấp  $b_L - a_L$ ). Ta cập nhật BIT và truy vấn tổng của  $K$  phần tử nhỏ nhất trong  $O(\log N)$ .

## 3 Phân tích tính chất thuật toán

- **Tại sao phù hợp?** Bài toán yêu cầu tối ưu hóa với số lượng phần tử lớn. Cách tiếp cận tham lam giúp giảm không gian tìm kiếm xuống còn  $N$  trường hợp (dựa trên tiền tố  $L$ ). Việc sử dụng BIT giúp thao tác tính tổng  $K$  phần tử nhỏ nhất (vốn tốn  $O(N)$  hoặc  $O(N \log N)$  nếu làm ngay thơ) xuống còn  $O(\log N)$ .
- **Tính đúng đắn:** Việc sắp xếp theo  $b_i$  đảm bảo rằng ta không bao giờ bỏ lỡ cấu hình tối ưu liên quan đến việc chọn các cấp độ 2 sao "rẻ nhất" theo nghĩa tổng quát.

## 4 Độ phức tạp thời gian và không gian

### 4.1 Độ phức tạp thời gian

- **Sắp xếp:** Ta sắp xếp các cấp độ theo  $b_i$  và sắp xếp các giá trị để rời rạc hóa:  $O(N \log N)$ .

- **Vòng lặp chính:** Ta duyệt  $L$  từ 0 đến  $N$ . Trong mỗi bước, ta thực hiện thao tác cập nhật BIT và truy vấn tìm kiếm nhị phân trên BIT (để tìm tổng  $K$  phần tử nhỏ nhất). Cả hai thao tác này mất  $O(\log N)$ .
- Tổng cộng:  $N \times O(\log N) = O(N \log N)$ .
- Với  $N = 3 \cdot 10^5$ ,  $N \log N \approx 5.4 \cdot 10^6$  phép tính, hoàn toàn thỏa mãn giới hạn thời gian 5 giây.

## 4.2 Độ phức tạp không gian

- Ta cần lưu trữ mảng các cấp độ, mảng rời rạc hóa và 2 cây BIT.
- Kích thước các mảng đều tỷ lệ thuận với  $N$ .
- Tổng độ phức tạp không gian:  $O(N)$ . Giới hạn 256MB là rất dư dả cho  $N = 3 \cdot 10^5$ .