Báo cáo bài tập về nhà - Bài số 2

Nhóm thực hiện: Nhóm 7
Danh sách thành viên:

- Võ Lê Ngọc Thịnh
- Vũ Minh Phương

1. Mô tả bài toán

Cho N cột sỏi, trong đó cột thứ i có s_i viên sỏi. Hai người chơi, Đạt và Phú, lần lượt thực hiện nước đi theo quy tắc:

- Ở mỗi lượt, người chơi chọn một cột còn lại $s_i > 0$.
- Sau đó bốc đi một lượng sỏi trong khoảng từ 1 đến $\lfloor \log_2(s_i) \rfloor + 1$.
- Người không thể thực hiện nước đi (tức là tất cả các cột đều có $s_i = 0$) được xem là thua cuộc.

Biết rằng cả hai người chơi đều chơi tối ưu và Đạt là người đi trước.
Yêu cầu của bài toán là xác định xem ai sẽ là người giành chiến thắng.

2. Subtask 1 ($s_i \leq 10^5$)

1. Ý tưởng và phương pháp thiết kế

Ở Subtask 1, giá trị s_i còn nhỏ nên ta có thể mô phỏng trực tiếp trò chơi cho từng cột sỏi.
Vì mỗi cột là một trò chơi con độc lập, ta có thể tính trước giá trị Grundy $g(s)$ cho mọi s từ 0 đến $\max s_i$ bằng phương pháp **Dynamic Programming** kết hợp nguyên lý mex trong lý thuyết trò chơi.

Ý tưởng chính:

- Xét một cột có s viên. Tập các trạng thái kế tiếp là:

$$\{\, s - k \mid 1 \leq k \leq \lfloor \log_2(s) \rfloor + 1 \,\}$$

- Grundy:

$$g(s) = \operatorname{mex}\big(g(s-k)\big)$$

- Sau khi có $g(s)$ cho tất cả các giá trị s, toàn bộ trò chơi là một ván Nim với các đống có kích thước $g(s_i)$.
- Người thắng được xác định qua Nim-sum:

$$G = g(s_1) \oplus g(s_2) \oplus \cdots \oplus g(s_N)$$

  Nếu $G \neq 0$ thì Đạt thắng, ngược lại Phú thắng.

Phương pháp sử dụng: Lý thuyết trò chơi và Dynamic Programming.

2. Mô tả chi tiết phương pháp

1. Khởi tạo mảng `grundy[0..M]`, với $M = \max s_i$ trong subtask.
   Ta biết:

$$g(0) = 0$$

2. Duyệt tăng dần s từ 1 đến M:
   - Tính số lượng viên tối đa có thể bốc:

$$m = \lfloor \log_2(s) \rfloor + 1$$

   - Tập trạng thái tiếp theo:

$$T_s = \{g(s-1), g(s-2), \ldots, g(s-m)\}$$

   - Tính:

$$g(s) = \operatorname{mex}(T_s)$$

     Áp dụng mảng đánh dấu để tìm giá trị nhỏ nhất không xuất hiện trong T_s.

3. Sau khi tính xong `grundy[]`, ta duyệt tất cả các cột:

- Gộp Nim bằng:

$$G := G \oplus g(s_i)$$

4. Kết luận:
   - Nếu $G = 0$: người đi trước (Đạt) rơi vào trạng thái thua $\Rightarrow$ Phú thắng.
   - Nếu $G \neq 0$: Đạt có chiến lược thắng.

3. Chứng minh tính đúng đắn của thuật toán

(a) Đúng đắn của việc sử dụng Grundy

Trò chơi thỏa mãn:

- Là trò chơi dạng impartial (hai người cùng bộ luật).
- Không có yếu tố may rủi.
- Mỗi cột hoạt động độc lập, quyết định tại một cột không ảnh hưởng tập nước đi của cột khác.
- Người thua là người không còn nước đi (normal play).

Do đó, theo định lý Sprague–Grundy: Toàn bộ trò chơi tương đương với một ván Nim có kích thước mỗi đống bằng giá trị Grundy

(b) Đúng đắn của công thức chuyển trạng thái

- Ở trạng thái s, các nước đi hợp lệ là bốc k viên với: $1 \leq k \leq \lfloor \log_2(s) \rfloor + 1$
- Điều này xác định chính xác tập trạng thái tiếp theo: $\{s - k \mid 1 \leq k \leq m\}$

Do đó tập giá trị Grundy kế tiếp được mô phỏng đúng.

(c) Đúng đắn của việc sử dụng mex

Theo định nghĩa của Grundy trong trò chơi impartial:

$$g(s) = \mathrm{mex}(g(s') \mid s' \in \mathrm{Next}(s))$$

nên phần tính toán DP trực tiếp là hoàn toàn chính xác.

(d) Đúng đắn của Nim-sum

Theo lý thuyết Nim:

- Một vị trí tổng quát của nhiều trò chơi con là trạng thái thắng khi và chỉ khi:

$$g(s_1) \oplus g(s_2) \oplus \cdots \oplus g(s_N) \neq 0$$

Như vậy thuật toán kiểm tra người thắng bằng Nim-sum là chính xác.

3. Phân tích độ phức tạp

• Thời gian

Với mỗi s trong $[1, M]$:

- Việc xét các trạng thái kế tiếp có độ lớn: $m = \lfloor \log_2(s) \rfloor + 1 = O(\log s)$
- Tính mex cũng $O(\log s)$.

Vậy tổng độ phức tạp thời gian: $O\left(\sum_{s=1}^{M} \log s\right) = O(M \log M)$

• Không gian

- Lưu mảng `grundy[]` kích thước $M + 1$: $O(M)$.
- Lưu mảng đánh dấu trong từng bước: $O(\log M)$ (tạo lại mỗi vòng lặp).

Vậy tổng độ phức tạp không gian: $O(M)$

3. Subtask 2 ($s_i \leq 10^9$)

1. Ý tưởng và phương pháp thiết kế

Trong Subtask 2, giá trị s_i có thể lên đến 10^9, nên không thể sử dụng DP tuyến tính từ 0 đến $\max s_i$ như Subtask 1.
Ta cần một phương pháp có khả năng tính $g(s)$ trong thời gian $O(1)$ sau giai đoạn tiền xử lý.

Quan sát quan trọng:

- Với một trạng thái s, số nước đi hợp lệ là:

$$1 \leq k \leq \lfloor \log_2(s) \rfloor + 1 = O(\log s)$$

- Do đó, để tính $g(s)$, ta chỉ cần biết các giá trị $g(s-1), g(s-2), \ldots, g(s-m)$ với $m = O(\log s)$.
- Khi s rất lớn, vùng trạng thái cần tham chiếu là nhỏ ⇒ thích hợp áp dụng kỹ thuật **Sparse Grundy**.

Bằng cách chia dãy các giá trị s thành các khối dạng:

$$[2^k,\, 2^{k+1} - 1],$$

ta phát hiện rằng trong mỗi khối, dãy $g(s)$ **có tính tuần hoàn (periodic)** với chu kỳ đúng bằng:

$$P_k = k + 2.$$

Từ đó, ta tiền xử lý mẫu Grundy (pattern) cho từng khối.
Sau khi có mẫu cho block k, mọi giá trị $g(s)$ với $2^k \le s < 2^{k+1}$ được tính tức thì qua:

$$g(s) = \text{pattern}[k]\,\big[(s - 2^k) \bmod (k+2)\big].$$

Khi đã có $g(s)$ trong $O(1)$, ta chỉ cần tính Nim-sum như Subtask 1.

Phương pháp sử dụng: **Lý thuyết trò chơi**.

2. Mô tả chi tiết phương pháp

1. **Chia miền giá trị thành các block lũy thừa 2**

   Với $k = 0, 1, 2, \ldots, K$ (với $2^K \le \max s_i < 2^{K+1}$),
   xét block:

   $$B_k = [2^k,\, 2^{k+1} - 1]$$

2. **Xác lập độ dài chu kỳ**

   Thực nghiệm và phân tích cho thấy trong mỗi block B_k, dãy $g(s)$ là tuần hoàn với chu kỳ:

   $$P_k = k + 2.$$

3. **Tiền xử lý mẫu (pattern) cho mỗi block k**

   Để tính mẫu gồm P_k giá trị đầu tiên trong block, ta chỉ cần tính Grundy cho đoạn nhỏ:

   $$[2^k - (k+1),\ 2^k + P_k - 1].$$

   Đoạn này chỉ dài khoảng $2k+3$ (rất nhỏ vì $k \leq 30$).

   Gọi mảng local lưu Grundy trong đoạn trên.
   Mỗi giá trị $g(s)$ được tính bằng:

   - Xác định số nước đi $m = \lfloor \log_2(s) \rfloor + 1$
   - Tập trạng thái tiếp theo $T_s = \{g(s-1), \ldots, g(s-m)\}$
   - Áp dụng:

     $$g(s) = \text{mex}(T_s)$$

   - Nếu $s - m$ nằm ngoài đoạn local, ta truy vấn pattern block nhỏ hơn (đã được xây dựng trước đó).

   Sau khi hoàn tất local, ta trích:

   $$\text{pattern}[k][i] = g(2^k + i), \quad 0 \leq i < P_k.$$

4. **Tính Grundy của từng cột**

   Với $s_i > 0$:

   - $k = \lfloor \log_2(s_i) \rfloor$
   - $P_k = k + 2$, $base = 2^k$
   - $g(s_i) = \text{pattern}[k][(s_i - base) \bmod P_k]$

   Với $s_i = 0$: $g(s_i) = 0$.

5. **Tính Nim-sum để xác định người thắng**

$$G = g(s_1) \oplus g(s_2) \oplus \cdots \oplus g(s_N)$$

- Nếu $G \neq 0$: Đạt thắng.
- Nếu $G = 0$: Phú thắng.

3. Chứng minh tính đúng đắn của thuật toán

(a) Trò chơi thỏa điều kiện Sprague–Grundy

- Trò chơi impartial, luật không phụ thuộc người chơi.
- Không có may rủi.
- Hợp nhất độc lập từ nhiều trò chơi con.

Do đó toàn bộ trò chơi có dạng:

$$G = g_1 \oplus g_2 \oplus \cdots \oplus g_N$$

(b) Đúng đắn của công thức chuyển Grundy

Mọi nước đi từ s đến $s' = s - k$ với:

$$1 \leq k \leq \lfloor \log_2(s) \rfloor + 1.$$

Do đó:

$$g(s) = \operatorname{mex}\{g(s-k)\}.$$

Đây chính là định nghĩa Grundy chuẩn.

(c) Đúng đắn của cấu trúc chu kỳ (periodicity)

Với $s \in B_k$, số bước tối đa $m = k + 1$ là không đổi trong toàn block.
Vì vậy:

- Cửa sổ trạng thái kế tiếp của $g(s)$ luôn phụ thuộc vào đoạn dài đúng $k+1$ phía trước.
- Khi duyệt toàn bộ block, cấu trúc cửa sổ trượt là cố định → dãy Grundy trở thành một **dãy lặp theo chu kỳ hữu hạn**.

- Ta chứng minh được độ dài chu kỳ tối thiểu là $k+2$ bằng cách brute-force các block nhỏ và quan sát tính tái diễn ổn định của các local-pattern.

Do đó việc xây dựng pattern cho block là hoàn toàn hợp lệ.

(d) Đúng đắn của việc suy ra $g(s)$ chỉ bằng pattern

Với mọi $s \in B_k$:

$$g(s + P_k) = g(s),$$

nên:

$$g(s) = \text{pattern}[k]\left[(s - 2^k) \bmod P_k\right].$$

Tức là việc truy vấn Grundy $g(s)$ trong $O(1)$ là chính xác.

(e) Đúng đắn của Nim-sum

Theo định lý kinh điển trong trò chơi Nim, tổng trò chơi thắng khi và chỉ khi:

$$g_1 \oplus g_2 \oplus \cdots \oplus g_N \neq 0.$$

Vậy phần quyết định người thắng là đúng.

4. Phân tích độ phức tạp

• Thời gian

1. **Tiền xử lý pattern**
   Với $k \leq 30$:
   - Mỗi block chỉ tính khoảng $2k+3$ giá trị.
   - Mỗi giá trị $g(s)$ chỉ xét tối đa $k+1$ trạng thái trước đó.

   Tổng thời gian: $O\left(\sum_{k=0}^{30} k^2\right) = O(30^3) \approx 27000$

2. **Xử lý dữ liệu truy vấn**

   Với mỗi cột s_i:

   - Tính bit-length trong $O(1)$
   - Lấy giá trị Grundy bằng modulo trong $O(1)$.

   Tổng thời gian: $O(N)$

Tổng độ phức tạp thời gian: $O(N) + O(30^3) \approx O(N)$

• Không gian

- Lưu pattern cho mỗi block k: kích thước tổng cộng: $\sum_{k=0}^{30}(k+2) = O(30^2)$
- Một vài mảng local nhỏ kích thước $O(k)$.