멋진 구간 2 – 풀이

작성자: 박영우

부분문제 1

$[A[l], \cdots, A[r]]$에 연산을 적용하여 $[B[l], \cdots, B[r]]$로 만들 수 있다고 하자. 이 과정에서 연속하게 시행된 어떤 두 연산에 대하여 처음 시행된 연산에서 증가시킨 값보다 나중에 시행된 연산에서 증가시킨 값이 더 작다면, 두 연산의 순서를 바꿔서 시행해도 된다. 따라서 $[l, r]$이 좋은 구간인 것은 $A[i] < B[i]$인 인덱스 i 중 $A[i]$가 가장 작은 인덱스부터 $A[i]$를 증가시키는 순서대로 연산을 시행하여 $[B[l], \cdots, B[r]]$로 만들 수 있는 것과 필요충분조건이다.

이를 쿼리마다 $O(N^2 \max\{B_i\})$에 확인해주면 1번 부분문제를 해결할 수 있다.

부분문제 2

배열 A가 1로만 구성되어 있을 경우, $[l, r]$이 좋은 구간일 필요충분조건은 다음과 같다.

- $[B[l], \cdots, B[r]]$에 1 이상 $\max_{l \le i \le r}\{B[i]\}$ 이하의 원소가 적어도 한 번씩 등장한다.

다음과 같이 증명할 수 있다.

$B_{max} = \max_{l \le i \le r}\{B[i]\}$로 정의하자.

- $\{B[l], \cdots, B[r]\} = \{1, \cdots, B_{max}\}$라 가정하면, 부분문제 1에서 사용한 알고리즘이 항상 성공함을 알 수 있다. 따라서 $[l, r]$은 좋은 구간이다.
- 반대로, $\{B[l], \cdots, B[r]\} \neq \{1, \cdots, B_{max}\}$라 가정하자. 이는 $c \in \{1, \cdots, B_{max}\}, c \notin \{B[l], \cdots, B[r]\}$인 정수 c가 존재함을 의미한다. $B_{max} \in \{B[l], \cdots, B[r]\}$이므로 $c < B_{max}$이다. 만약 $[l, r]$이 좋은 구간이라면, $c < B_{max}$이므로 연산이 시행되는 동안 c는 적어도 한 번의 시점에서 배열에 등장한다. 그러나 그 시점 이후 적어도 하나의 인덱스는 c이고, 배열의 최종 상태에서는 c가 없어야 하기 때문에 모순이다.

위 필요충분조건을 쿼리당 $O(N)$ 또는 $O(N \lg N)$ 정도에 확인하면 해당 부분문제를 풀 수 있다.

부분문제 3

부분문제 2의 필요충분조건을 쿼리당 $O(\lg N)$의 시간복잡도에 확인하면 된다. $\{B[l], \cdots, B[r]\} = \{1, \cdots, B_{max}\}$일 필요충분조건은 두 집합의 크기가 같다는 것이고, $|\{B[l], \cdots, B[r]\}| = B_{max}$와 동치이다. $B[l], \cdots, B[r]$에서의 서로 다른 수의 개수를 세는 것은 세그먼트 트리를 통해 총 Q개의 쿼리에 대해 $O((N + Q) \lg N)$ 시간복잡도에 구할 수 있고, $\max_{l \le i \le r}\{B[i]\}$도 세그먼트 트리를 통해 같은 시간 복잡도에 구할 수 있다. 따라서 전체 문제를 $O((N + Q) \lg N)$에 풀 수 있다.

부분문제 4

$l \le r$인 두 정수 l, r에 대해 $[l, r)$을 다음과 같이 정의하자.

- $l = r$이면 $[l, r) = \emptyset$
- $l < r$이면 $[l, r) = \{l, l+1, \cdots, r-1\}$

다음과 같은 관찰을 할 수 있다.

$[l, r]$이 좋은 구간일 필요충분조건은 다음 조건을 만족하는 것이다.

- 모든 $l \le i \le r$인 i에 대하여 $[A[i], B[i]) \subseteq \{B[l], \cdots, B[r]\}$를 만족한다. 즉, $\bigcup_{l \le i \le r} [A[i], B[i]) \subseteq \{B[l], \cdots, B[r]\}$이다.

다음과 같이 증명할 수 있다.

1. 조건을 만족하면 좋은 구간이다. $A[i] < B[i]$이며 $A[i]$가 가장 작은 인덱스 i를 고른다. 그러한 인덱스가 없다면 이미 조건을 충족한다.
   어떤 $B[j]$가 있어 $A[i] = B[j]$이다. 만약 $i = j$라면 $A[i] = B[i]$이므로 $i \neq j$이다.
   만약 $A[j] < B[j]$일 경우 $A[j] < A[i]$이므로 i의 정의에 모순이다.
   따라서 $A[j] = B[j]$이고, (i, j)에 대해 연산을 시행하면 된다.
2. 좋은 구간은 항상 위 조건을 만족한다.
   연산은 배열의 값을 증가시키므로 첫 번째 조건은 항상 충족한다. 만약 어떤 배열에 x라는 값이 있었다면 이 상태에서 연산을 몇 번을 시행하더라도 배열에 여전히 x는 존재한다.
   만약 조건을 만족하지 않는 인덱스 i가 있고, 어떤 $c \in [A[i], B[i])$가 있어 $c \neq B[l], \cdots, B[r]$이라 하자. 어떤 시점에서 i번 인덱스의 값은 c에서 $c+1$로 증가했고, 이 시점에서 배열에는 값이 c인 인덱스가 존재했다. 따라서 배열의 최종 상태 $[B[l], \cdots, B[r]]$에도 c가 존재하며, 모순이다.

이 필요충분조건을 쿼리당 $O(N)$또는 $O(N \lg N)$ 시간에 확인하면 해당 부분문제를 해결할 수 있다.

부분문제 5

다양한 풀이가 존재한다. $[l, r]$에 $A[i] = A[j], B[i] = B[j]$인 두 인덱스 i, j가 있다면 두 인덱스 중 하나만 고려해도 된다.

$B[i] \le 2$일 경우 $(A[i], B[i])$로 가능한 순서쌍은 $(1, 1), (1, 2), (2, 2)$로 총 세가지로, 구간에 각각에 해당하는 인덱스가 있는지 확인한 후 총 2^3가지 경우에 대해 문제를 해결해주면 된다.

부분문제 6

입력으로 주어진 배열 $[B[1], B[2], \cdots, B[N]]$의 최댓값을 B라 하자.

부분문제 4의 필요충분조건을 쿼리당 $O(B)$에 확인하면 된다.

모든 $1 \le i \le N, 1 \le j \le B$인 인덱스 i, j에 대해 다음 두 값을 구한다.

- 1 이상 i 이하인 모든 인덱스 k 중 $j \in [A[k], B[k])$인 인덱스의 개수
- 1 이상 i 이하인 모든 인덱스 k 중 $j = B[k]$인 인덱스의 개수 다음 두 값은 전체 $O(NB)$에 구할 수 있다.

따라서 각 질문 $[l, r]$에 대하여 1 이상 B 이하의 임의의 원소 c에 대해 $c \in \bigcup_{l \le i \le r} [A[i], B[i])$인지, $c \in \{B[l], \cdots, B[r]\}$인지 여부를 알 수 있다.

따라서 각 쿼리당 $O(B)$의 시간에 문제를 해결할 수 있고, 총 $O((N+Q)B)$ 시간복잡도에 문제를 해결할 수 있다.

부분문제 7

l을 고정하고, 구간의 왼쪽 끝점이 l인 질문들만 고려하자.

1 이상 $B(= \max_{1 \le i \le N}\{B[i]\})$ 이하인 모든 정수 c에 대해 $c \in [A_i, B_i)$이며 $l \le i$인 최소 인덱스 i를 X_c, $c = B_i$이며 $l \le i$인 최소 인덱스를 Y_c라 하자.

$[l, r]$이 좋은 구간이 아닐 필요충분조건은 어떤 c가 존재하여 $c \notin \{B[l], \cdots, B[r]\}$이면서 $c \in \bigcup_{l \le i \le r} [A[i], B[i])$인 것이다. 즉, 어떤 c가 존재하여 $X_c \le r < Y_c$를 만족한다면 $[l, r]$은 좋은 구간이 아니다.

각각의 r에 대해 $X_c \le r < Y_c$를 만족하는 서로 다른 정수 c의 개수를 세그먼트 트리를 통해 관리한다. 이 세그먼트 트리의 i번째 인덱스를 $seg[i]$라 하자. 즉, $X_c < Y_c$인 c는 $i \in [X_c, Y_c)$인 모든 정수 i에 대해 $seg[i]$에 1 만큼 기여한다.

이를 위해, 다음 집합들을 관리한다.

- $X_c < Y_c$인 c들의 집합 S
- $X_c \ge Y_c$인 c들의 집합 T

고정된 왼쪽 끝점 l을 $l-1$로 줄일 때 $c \in [A[l-1], B[l-1])$인 모든 X_c와 $Y_{B[l-1]}$은 $l-1$로 감소한다.

- 업데이트되는 X_c들의 경우, 변경되기 전 X_c값이 같은 구간들로 분할하고 같은 구간에 포함되는 인덱스들에 대해 세그먼트 트리를 한번에 업데이트한다. 배열 X를 std::set, 세그먼트 트리 등의 자료구조를 통해 관리하면 한 구간 당 $O(\lg B)$의 시간에 찾을 수 있다.
  - $c \in [s, e]$인 모든 c에 대해 X_c의 값이 p에서 $l-1$로 업데이트된다 하자.
  - 업데이트 전 $[s, e] \cap S$에 있는 원소 c들은 업데이트 이후에도 S에 있어야 하며, $seg[X_c], \cdots seg[Y_c - 1]$에 1씩 기여하고 있다. 따라서 X_c의 값을 $l-1$로 바꿔주기 위해 $seg[l-1], \cdots, seg[p-1]$에 $|[s, e] \cap S|$를 더해주어야 한다. $|[s, e] \cap S|$는 S를 세그먼트 트리, bbst 등의 자료 구조를 통해 관리하면 $O(\lg B)$ 시간에 구할 수 있다.
  - 업데이트 후 $[s, e] \cap T$에 있는 원소 c들은 업데이트 이후에는 S에 있게 된다. 따라서 $seg[l-1], \cdots, seg[Y_c - 1]$에 1이 더해주어야 한다.
- $Y_{B[l-1]}$의 경우, $[X_{B[l-1]}, Y_{B[l-1]})$을 다시 계산하여 세그먼트 트리에 업데이트해주면 된다. 또한, 이 시점부터 $B[l-1]$은 T에 있게 된다.

위 과정에 따라 l을 $N-1$부터 0까지 줄여가며 세그먼트 트리를 업데이트해주고, 각각의 $[l, r]$ 쿼리에 대해 $seg[r]$이 0이라면 $[l, r]$이 좋은 구간이라 판정하는 방법으로 풀 수 있다. 이에 사용되는 시간복잡도는 $O(Q \lg B)$이다.

고정된 왼쪽 끝점을 l에서 $l-1$로 감소시킬 때 세그먼트 트리의 업데이트 과정에서 X 배열의 변화에 의해 세그먼트 트리를 업데이트하는 횟수는 다음 값의 합에 비례한다.

- 업데이트되는 서로 다른 구간 $[s, e]$의 수: $i \in (A[l-1], B[l-1])$, $X_{i-1} \ne X_i$인 인덱스 i의 개수
- $|T \cap [A[l-1], B[l-1])|$

업데이트 이후 $X_{A[l-1]}, \cdots X_{B[l-1]-1}$의 값은 모두 $l-1$로 변하므로, 전체 알고리즘이 시행되는 동안 첫 번째 값의 합은 $O(N)$이다.

$T \cap [A[l-1], B[l-1])$에 속한 모든 원소 c들은 업데이트 이후 T에서 S로 이동하고, 고정된 왼쪽 끝점을 한 번 옮기는 동안 최대 하나의 값($B[l-1]$)만 S에서 T로 이동하므로 전체 알고리즘이 시행되는 동안 두 번째 값의 합은 $O(N)$이다.

또한, 고정된 왼쪽 끝점을 l에서 $l-1$로 감소시킬 때 세그먼트 트리의 업데이트 과정에서 Y 배열의 변화에 의해 세그먼트 트리를 업데이트하는 횟수는 1회이다.

따라서 전체 시간복잡도는 $O((N+Q)\lg B)$이다.

부분문제 8

$[A[i], B[i]) \not\subseteq \{B[0], \cdots, B[N-1]\}$인 인덱스 i들은 std::set 등의 자료구조를 통해 $O(N \lg N)$ 시간에 전부 구할 수 있다. 이러한 인덱스들을 전부 모은 집합을 I라 하자, 만약 어떤 구간 $[l, r]$이 I에 있는 인덱스 중 하나를 포함했다면 이 구간은 반드시 좋은 구간이 아니다.

이 사실을 이용하면 부분 문제 4의 필요충분조건을 다음과 같이 변형할 수 있다.

- $[l, r] \cap I = \emptyset$이고,
- $\bigcup_{l \le i \le r} [A[i], B[i]) \cap \{B[0], \cdots, B[N-1]\} \subseteq \{B[l], \cdots, B[r]\}$이다.

즉, I의 원소를 포함하는 $[l, r]$의 구간을 전처리해 주면 A, B 배열을 좌표압축한 이후 부분 문제 7의 풀이를 그대로 적용할 수 있다.

전체 시간복잡도는 $O((N+Q)\lg N)$이다.