Segment Tree

CONTENTS

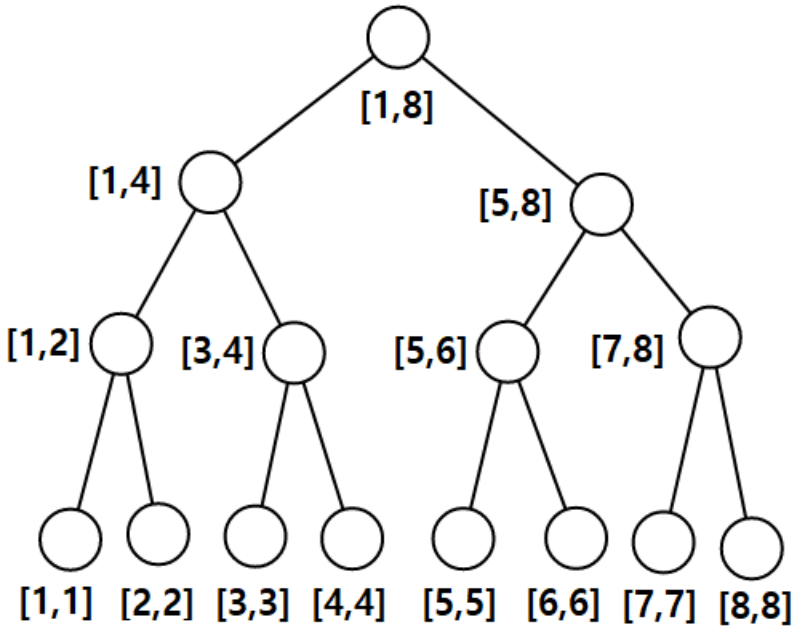
Ⅰ. Segment Tree

I . Segment Tree

Ⅰ. Segment Tree

1. 개념

- 한글로 번역하면 **구간 트리**
- **트리는 트리인데 각각의 노드가 특정 구간에 대한 query의 답을 갖고 있음**
- **Full Binary Tree로 구현한다**
- **상위 노드일 수록 더 넓은 구간에 대한 답을 가지고 있음**
- **Divide & Conquer한 결과를 메모이제이션 하자!!**
- **Update와 Query에 답을 구하는 연산 2가지를 제공**



Ⅰ. Segment Tree

1. 개념

- 예를 들어, 배열이 주어지고 각 구간에서의 최댓값을 구하는 문제가 있다고 하자
- 또한 배열의 특정 위치의 값을 바꾸는 연산이 일어난다
- 이를 어떻게 풀어야 할까???

- 가장 단순하게 생각하면 모든 Query에 대하여 해당 구간을 순회하면서 값을 구하면 될 것이다
- 자명하게도 시간복잡도는 $O(QN)$
- 이를 $O(Q \log N)$에 가능하게 해주는 자료구조

Ⅰ. Segment Tree

1. 개념

- 예를 들어, 배열이 주어지고 각 구간에서의 최댓값을 구하는 문제가 있다고 하자
- 또한 배열의 특정 위치의 값을 바꾸는 연산이 일어난다
- 이를 어떻게 풀어야 할까???

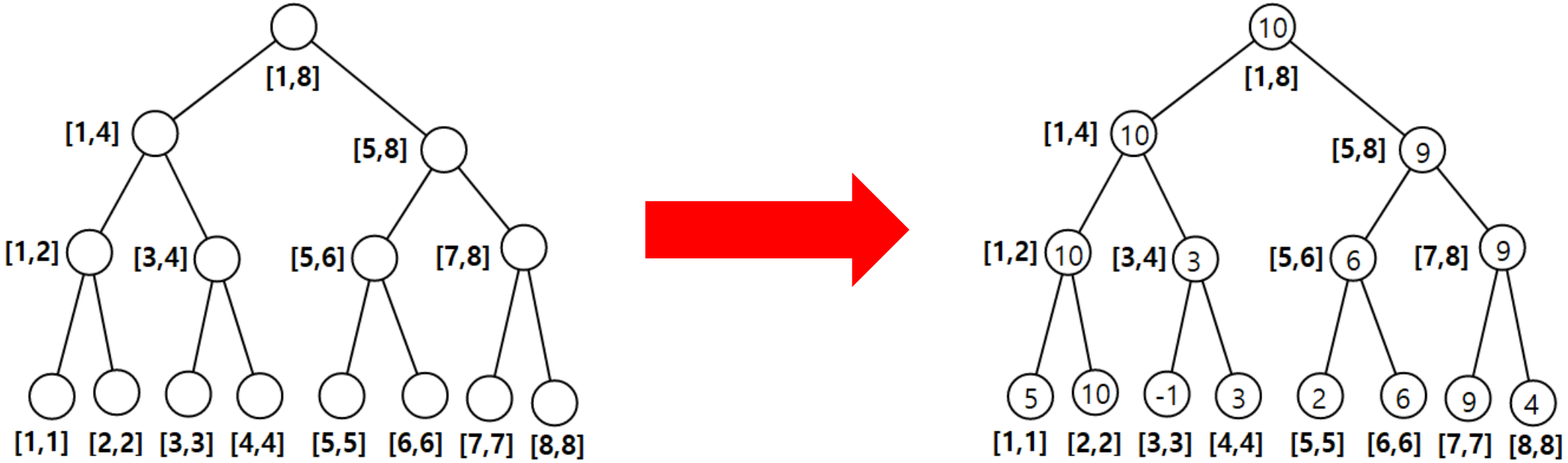
1	2	3	4	5	6	7	8
5	10	-1	3	2	6	9	4

Ⅰ. Segment Tree

1. 개념

- 예를 들어, 배열이 주어지고 각 구간에서의 최댓값을 구하는 문제가 있다고 하자

1	2	3	4	5	6	7	8
5	10	-1	3	2	6	9	4



Ⅰ. Segment Tree

1. 개념

- 예를 들어, 배열이 주어지고 각 구간에서의 최댓값을 구하는 문제가 있다고 하자

1	2	3	4	5	6	7	8
5	10	-1	3	2	6	9	4

- [1,8] 에서의 최댓값???

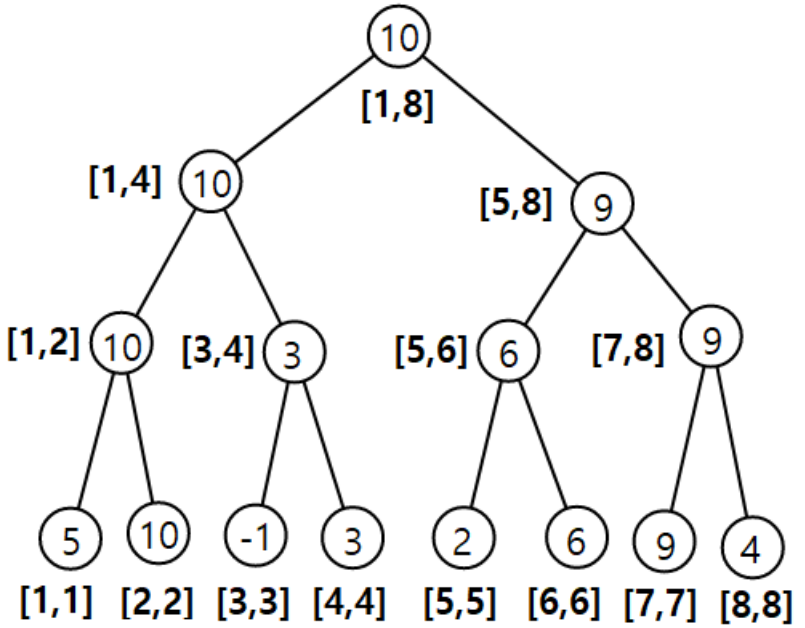
-> [1,8] 노드에 답이 있음

- [2,4] 에서의 최댓값???

-> [2,2]노드와 [3,4]노드의 값 중 최댓값

- [2,7] 에서의 최댓값???

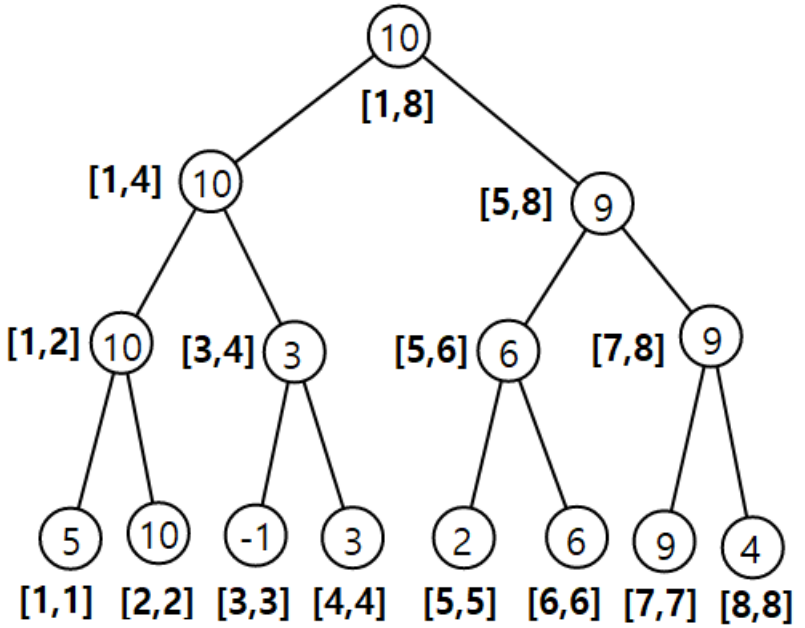
-> [2,2], [3,4], [5,6], [7,7] 노드의 값 중 최댓값



Ⅰ. Segment Tree

1. 개념

- 즉, Segment Tree를 contruct 한 뒤에 **구하고자 하는 구간에 포함되는 노드들을 방문하여 해당 Query를 처리**한다.
- 이렇게 하면 각각의 Query에 대하여 $O(\log N)$ 라는 시간복잡도에 처리가 가능하다.
- 이는 각 depth에서 방문 되는 노드가 최대 2개이기 때문이다.

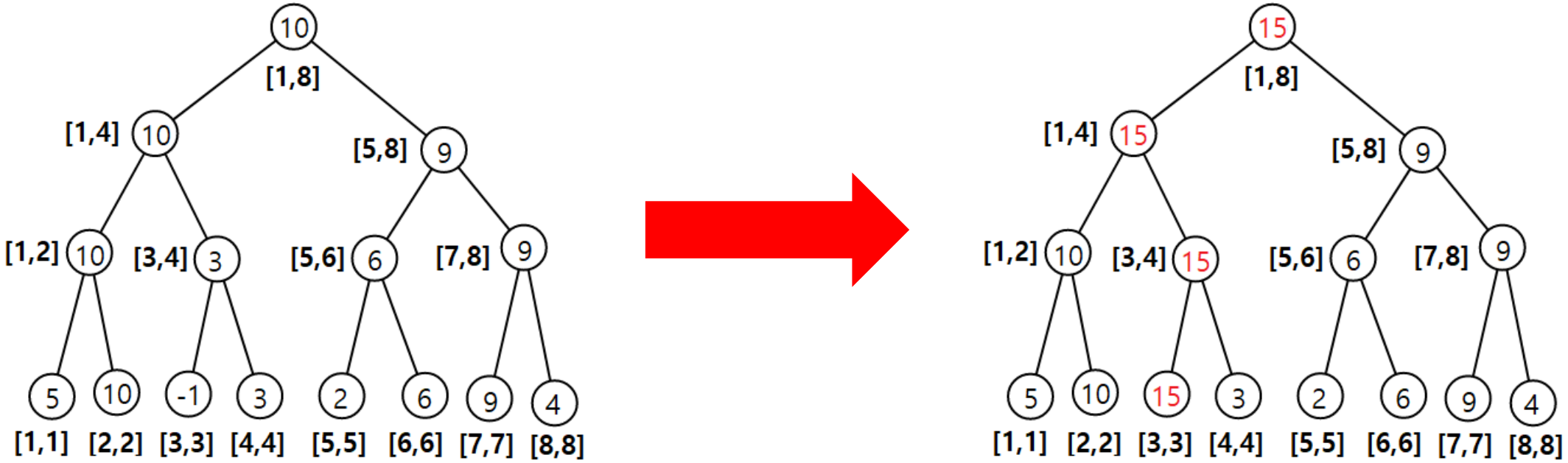


Ⅰ. Segment Tree

1. 개념

- 3번째 원소를 15로 바꾸는 연산이 일어났다고 하면

1	2	3	4	5	6	7	8
5	10	15	3	2	6	9	4



Ⅰ. Segment Tree

1. 개념

- 즉, 해당 위치의 **bottom부터 시작해서 parent 노드 까지 값을 update**한다.
- 보다시피 자명하게 update 한 번에 소요되는 시간은 $O(\log N)$

- 만약 특정 구간의 값을 모두 바꾸고 싶으면???

-> 일반적인 update로는 $O(\mathrm{M}\log N)$ (M=구간길이)

-> **Lazy Propagation**이라는 새로운 개념이 필요

뒤에서 설명할 예정이다.