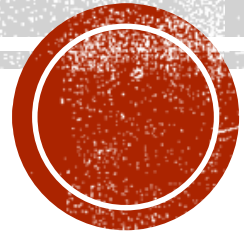
HEAVY-LIGHT DECOMPOSITION

서영선 sys7961@gmail.com



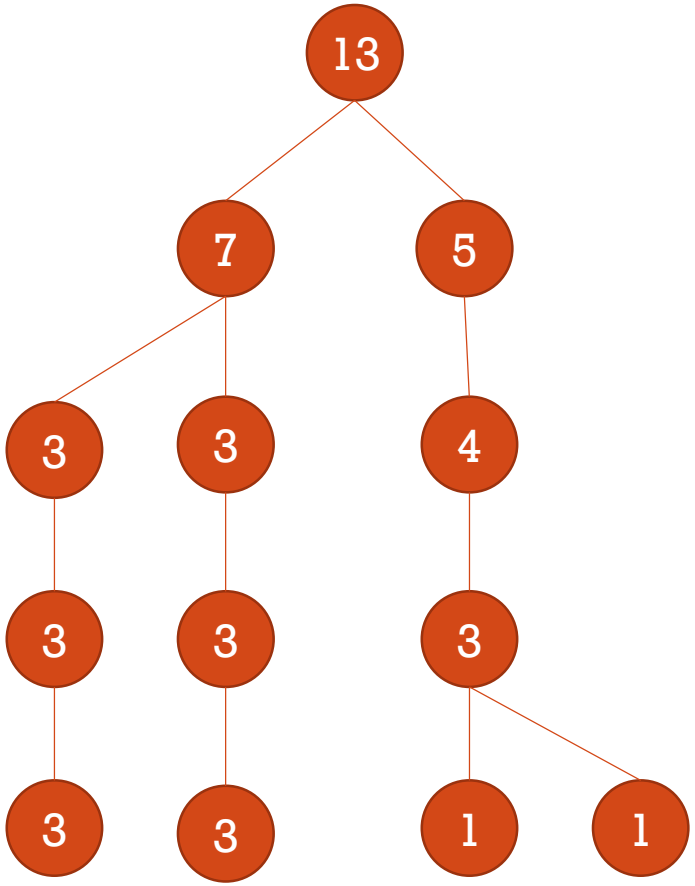
HEAVY-LIGHT DECOMPOSITION

- **Heavy-Light Decomposition**은 트리에 대해서 가지들을 적절히 잘라서 일자 경로인 묶음으로 만드는 것이다.
- 이러한 작업을 하면 두 정점 사이의 경로에 대해서 어떠한 연산을 할 때, 일자 경로를 $O(lgV)$개의 경로들에 대해 연산하는 것으로 할 수 있다.
- 이러한 일자 경로들을 세그먼트 트리, 펜윅 트리와 같은 구간에 대해서 처리해주는 자료구조를 사용한다면 $O(lg^2V)$에 처리할 수 있다.
- 이를 구현하기 위해서는
  - **LCA**(최소 공통 조상)
  - 구간에 대해서 처리해주는 자료구조**(Segment tree , Fenwick tree)**
- 하지만 이러한 지식은 이 알고리즘을 이해하는데 크게 영향을 안 미침에 따라 생략한다.

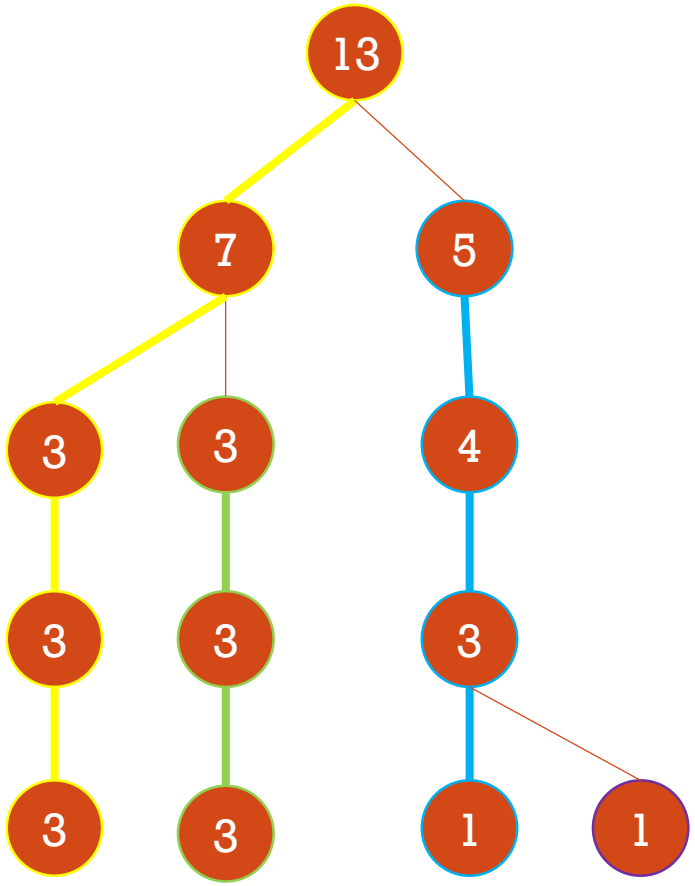
간선 분류

- 트리의 경로들을 나누기 전에 우선 정점별로 무게를 정의할 것이다.
- 무게는 자신과 해당 정점의 모든 자식 정점의 개수이다.
- 그 후 임의의 루트로부터 무게가 가장 큰 자식들로 이동하며 하나의 경로로 하고, 그 외의 자식들은 서브트리로 하여 이 과정을 반복한다.(무게가 가장 큰 자식이 여러 개라면 아무거나 해도 괜찮다.

간선 분류

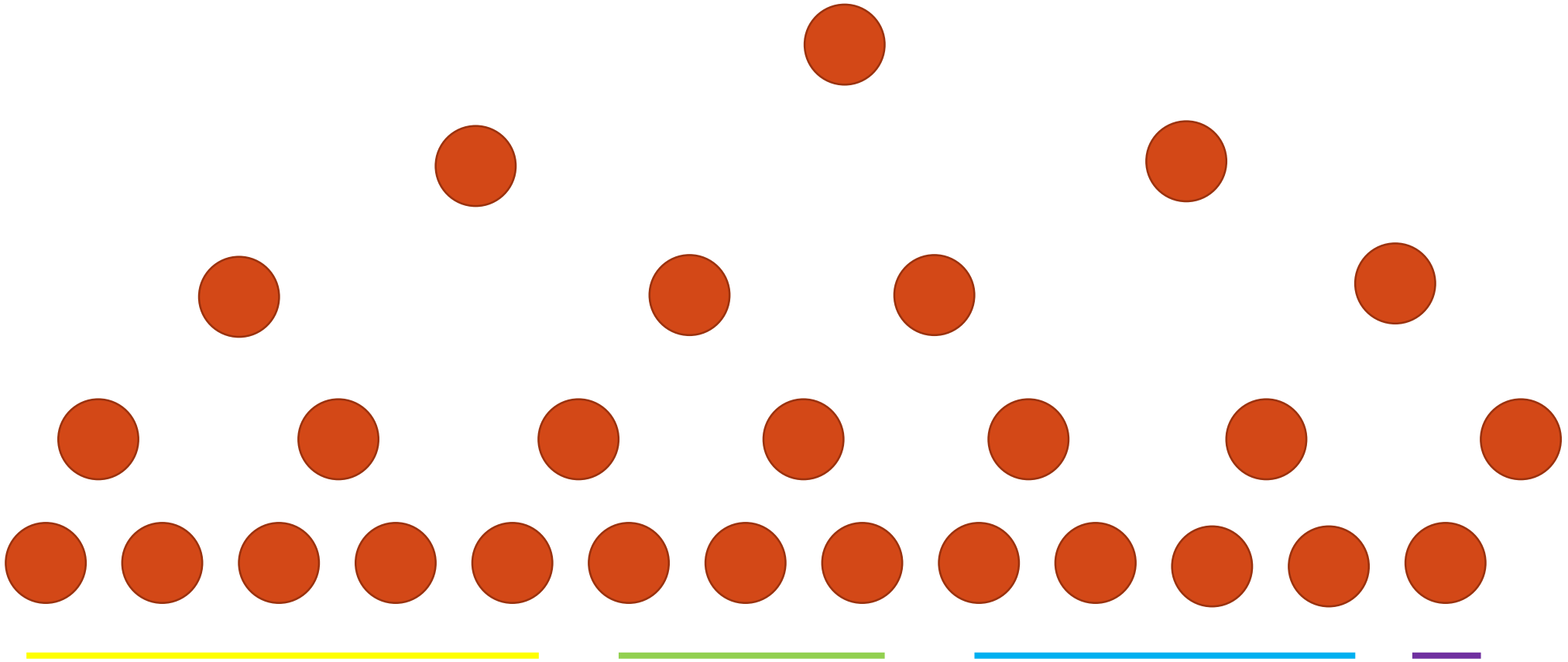


간선 분류



간선 분류

Segment tree



QUERY

- 간선들을 적절하게 분류하여 일자 경로들로 만들었다.
- 이제 두 정점 사이의 경로에 **query**를 날릴 수 있어야 한다.(트리에서는 두 정점사이의 경로의 유일함은 증명되어 있다.)
- 또한, **rooted tree**에서는 **LCA**(최저 공통 조상)을 거쳐가는 경로가 두 정점사이의 경로이다.
- 따라서 두 정점 각각에 대해서 **LCA**까지 따로 **query**를 날리면 된다.

QUERY

- **query**를 날릴 때, 현재 정점이 **LCA**와 같은 분류안에 존재하지 않는 다면, 그 분류의 가장 최고 위치까지 **query**를 날리고 그 위치의 부모로 올라가고 반복한다.
- 현재 정점이 **LCA**와 같은 분류안에 존재한다면, 해당 점에서 **LCA**까지 **query**를 날리고 종료한다.
- 최고 위치의 부모를 통하여 다른 분류의 정점으로 이동하는 것은 $O(lgV)$보다 크지 않다는 것은 증명할 수 있으므로, 한 **query**에 $O(lgV)$라면 전체 쿼리는 $O(lg^2V)$인 것을 알 수 있다.

QUERY

- 간선을 분류할 때, 가장 무거운 자식이 같은 분류가 되기 때문에, 다른 분류의 자식으로 이동할 때는 현재 무게 의 절반미만이라는 것을 알 수 있다.
- 왜냐하면 절반 이상을 들고 있다면 해당 자식보다 더 무거운 자식이 존재할 수 없으므로 그 자식이 가장 무거운 자식이 된다.
- 따라서 다른 분류의 정점으로 이동할 때마다 정점의 수(무게)는 /2 이상 된다는 것을 알 수 있다.
- 따라서 한 정점이 **LCA**까지 올라가는데 거치는 분류는 $O(lgV)$임을 알 수 있다.

구현 SOURCE

- https://gist.github.com/neinsys/7da3b0cf6989d7d23f33434b9fa35799