Day3 자료구조

선린인터넷고등학교 소프트웨어과
30610 나정휘

문제 목록

- BOJ1976 여행 가자
- BOJ2042 구간 합
- BOJ10999 구간 합 2
- BOJ10090 Counting Inversions
- BOJ15942 Thinking Heap
- BOJ2696 중앙값 구하기
- BOJ13306 트리
- BOJ18921 Cost of Subtree
- BOJ2957 이진 탐색 트리
- BOJ1321 군인
- BOJ5419 북서풍
- BOJ7469 K번째 수
- BOJ18193 비행기 타고 가요

여행 가자

- Union Find 구현

구간 합 구하기 1

- 세그 트리 구현 or 펜윅 트리 구현

구간 합 구하기 2

- Lazy Propagation 구현

Counting Inversions

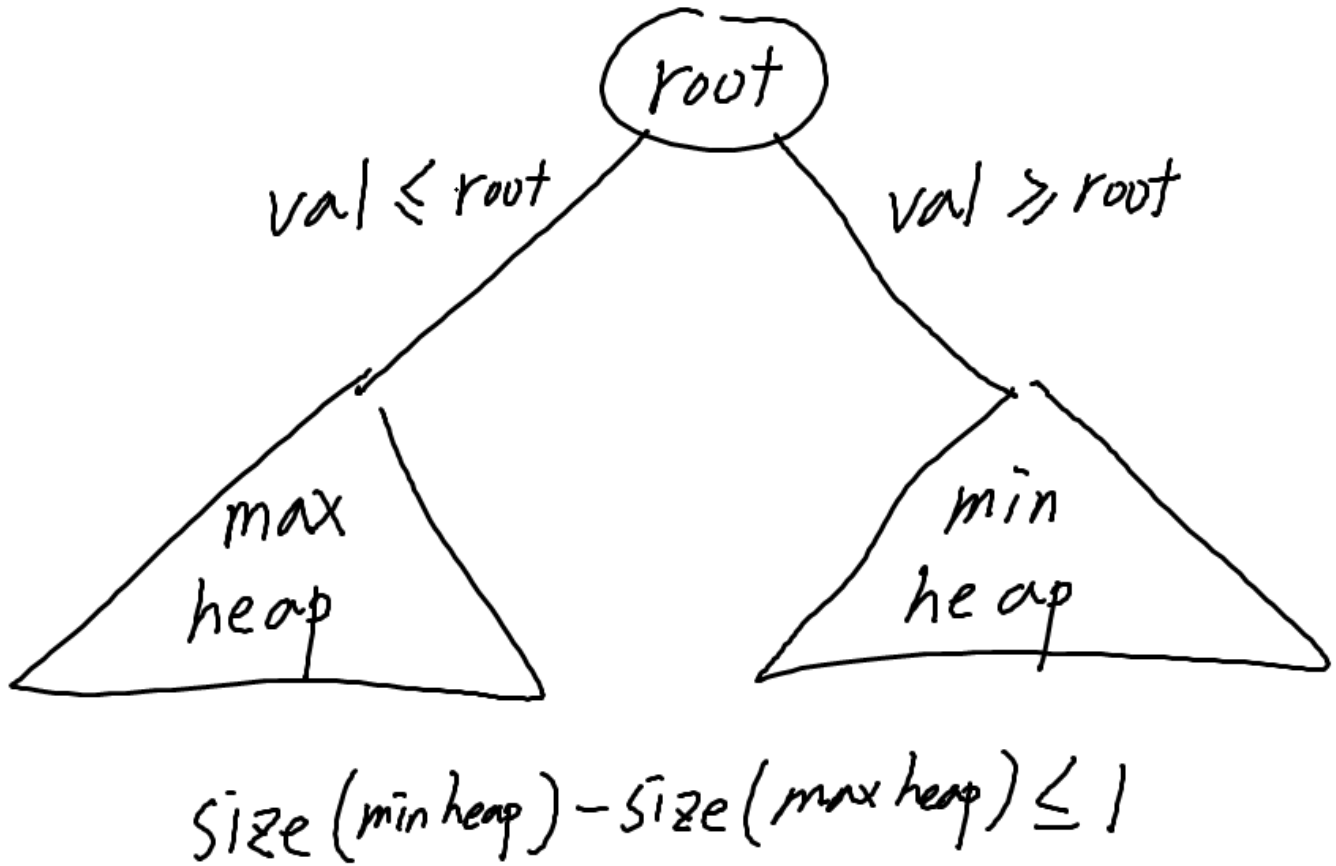
- sum(나보다 앞에 있으면서 큰 원소의 개수)

```
for(int i=1; i<=n; i++){
  ans += query(a[i]+1, MX);
  add(a[i], 1);
}
```

Thinking Heap

- 루트에서 p의 부모까지 1, 2, … 배치
- p의 자손들에 N, N-1, … 배치
- 남은 정점들 위에서부터 오름차순으로 배치

중앙값 구하기


root
val ≤ root
val ≥ root
max heap
min heap
size(min heap) - size(max heap) ≤ 1

트리

- **쿼리를 거꾸로 처리하자.**
- 간선 다 없앤 상태로 시작
- 간선 삭제 쿼리 -> 간선 추가 쿼리

트리

- **쿼리를 거꾸로 처리하자.**
- 간선 다 없앤 상태로 시작
- 간선 삭제 쿼리 -> 간선 추가 쿼리

- 간선 추가 쿼리 : Union
- 경로 존재 확인 : Find(u) = Find(v)?

- 웰-----논

Cost Of Subtree

- 13306 트리와 유사한 문제

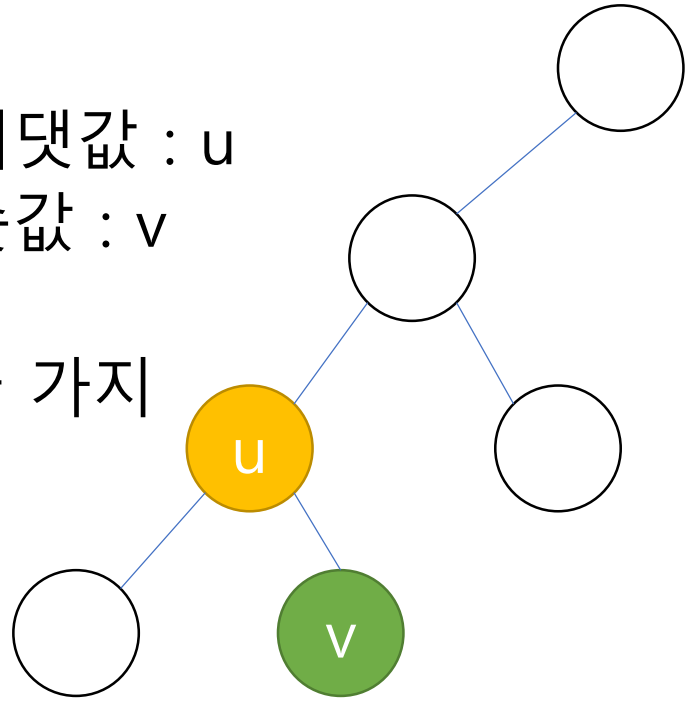
- 파이팅!

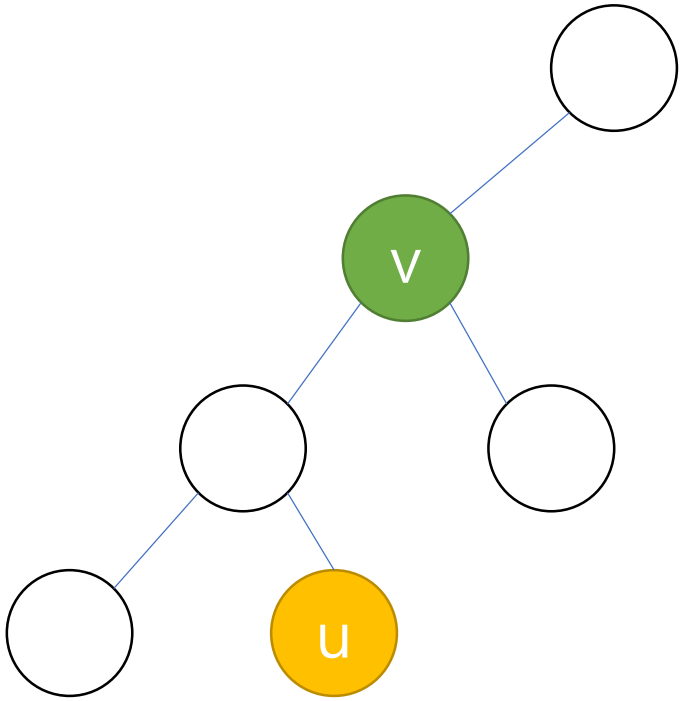
이진 탐색 트리

- 느린 O(N log N) 풀이 : N <= 300,000
- 빠른 O(N log N) 풀이 : N <= 5,000,000

이진 탐색 트리

- insert(x)
  - x보다 작은 최댓값 : u
  - x보다 큰 최솟값 : v
  - 두 형태 중 한 가지





- dep[x] = max(dep[u], dep[v]) + 1

이진 탐색 트리

- u, v를 set으로 구해주면 느린 O(N log N) 풀이 완성

- ~~std::set은 유사 로그 제곱~~

이진 탐색 트리 - 빠른 풀이

- N개의 방이 있고, 모든 방은 비어 있는 상태
- 원소를 삽입할 때마다 연속된 빈 공간이 분할됨
- 연속된 빈 공간을 하나의 정점으로 보자
  - 각 정점의 왼쪽, 오른쪽에 있는 수들을 관리

이진 탐색 트리 - 빠른 풀이

- 구간을 쪼갠다 -> 구간을 병합한다
- Union Find

- 각 서로소 집합마다 l, r값을 관리해주면 됨

군인

- (세그 or 펜윅) + 파라메트릭 서치

북서풍

- sum(오른쪽 아래에 있는 점들의 개수)
- 뇌절하고 2D Segment Tree를 쓴다 -> O(N log^2 N) 시간 초과

- **O(N log N)에 풀자.**

북서풍

- x, y좌표를 1..N으로 압축하고 생각하는 게 편하다. (좌표 압축)
- 점들을 하나씩 추가하면서 계산하자. (Counting Inversion 느낌)

북서풍

- Counting Inversion : 나보다 앞에 있는 원소들을 먼저 넣음
- 북서풍 : 나보다 아래에 있는 원소들을 먼저 넣음
  - y좌표가 같으면 오른쪽에 있는 원소 먼저

- 현재 들어간 원소 중 오른쪽에 있는 원소의 개수를 보면 된다.

북서풍

- 세그먼트 트리를 만들자.
- 자신보다 "오른쪽"에 있는 점만 중요하니까 x좌표만 봐도 된다.

```
// y좌표가 큰 점
// y좌표 같으면 x좌표가 큰 점이 먼저 오도록 정렬
sort(pt.begin(), pt.end(), comp);
for(auto i : pt){
  ans += query(i.x, MX);
  add(i.x, 1);
}
```

K번째 수

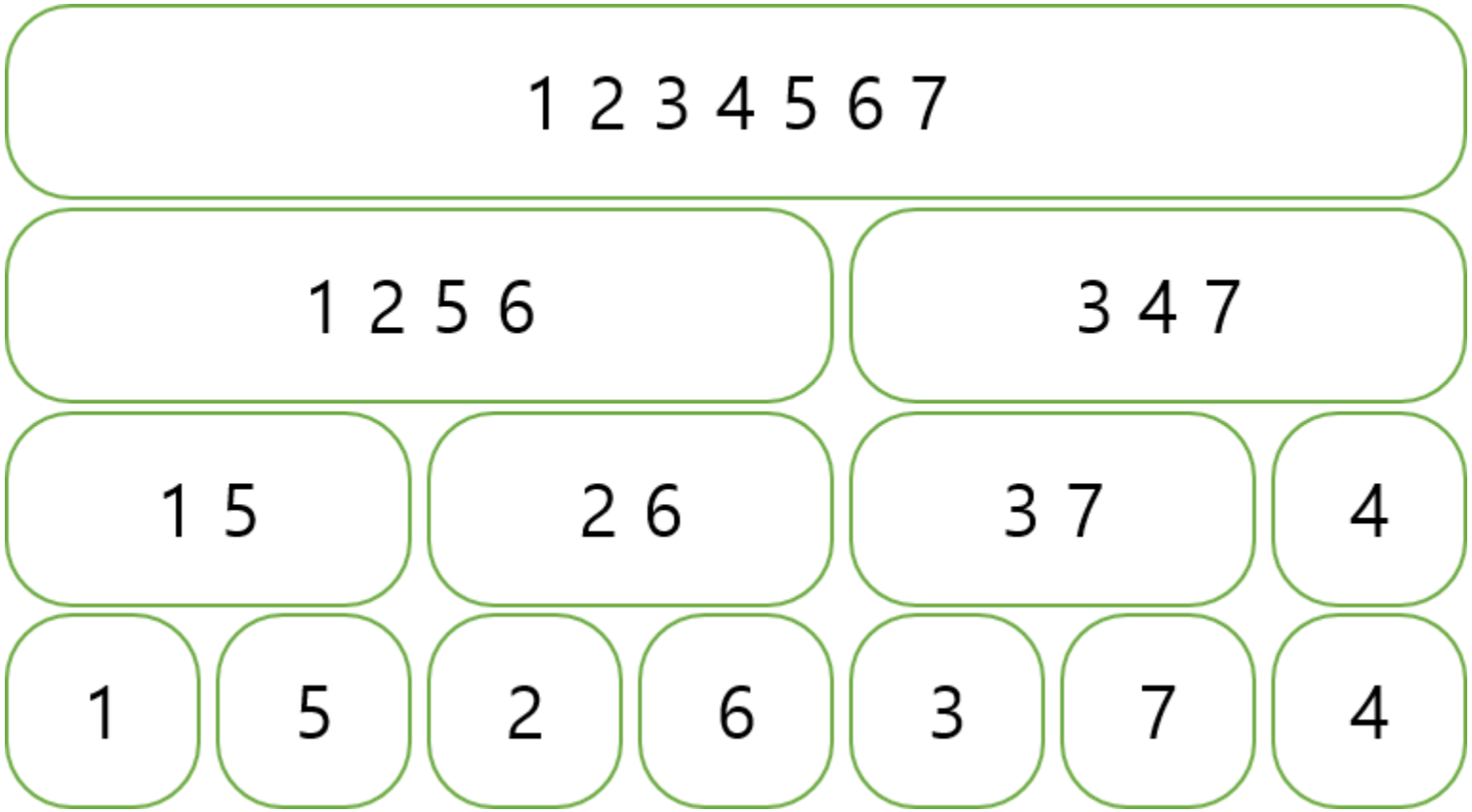
- Merge Sort Tree / PST
- O(Q log^3 N) / O(Q log N)

- Merge Sort Tree : 19916KB, 120ms, 1400B
- Persistent Segment Tree : 59280KB, 104ms, 2247B
- 내 띵킹에 의하면 Merge Sort Tree 짜는게 이득

Merge Sort Tree

- Do You Know “Merge Sort”?

Merge Sort Tree



Merge Sort Tree

```
vector<int> tree[1 << 18];
const int sz = 1 << 17;

void add(int x, int v){
  x += sz;
  tree[x].push_back(v);
}

void build(){
  // i번째 리프노드에 원소 추가
  for(int i=1; i<=n; i++) add(i, a[i]);

  // 내부 노드 구성
  for(int i=sz-1; i>=1; i--){
    // i*2, i*2+1은 이미 정렬되어 있음
    // O(SZ)만에 합쳐줌
    merge(tree[i*2].begin(), tree[i*2].end(),
          tree[i*2+1].begin(), tree[i*2+1].end(),
          back_inserter(tree[i]));
  }
}
```

K번째 수

- Merge Sort Tree가 할 수 있는 일
  - 구간 [s, e]에서 x보다 큰/작은/크거나 같은/작거나 같은 원소의 개수
  - 정렬되어 있으므로 lower_bound, upper_bound 등을 이용

- K번째 수 : 구간 [s, e]에서 k번째 수
  - Parametric Search : f(x) = x보다 작거나 같은 수가 k개 미만인가?
  - f(x)를 구하는데 O(log^2 N), 이분 탐색 O(log N) -> O(log^3 N)

비행기 타고 가요

- 그래프를 잘 만들고 다익스트라

- Naïve 풀이
  - 정점 N개, 간선 N^2M개 -> 시간초과

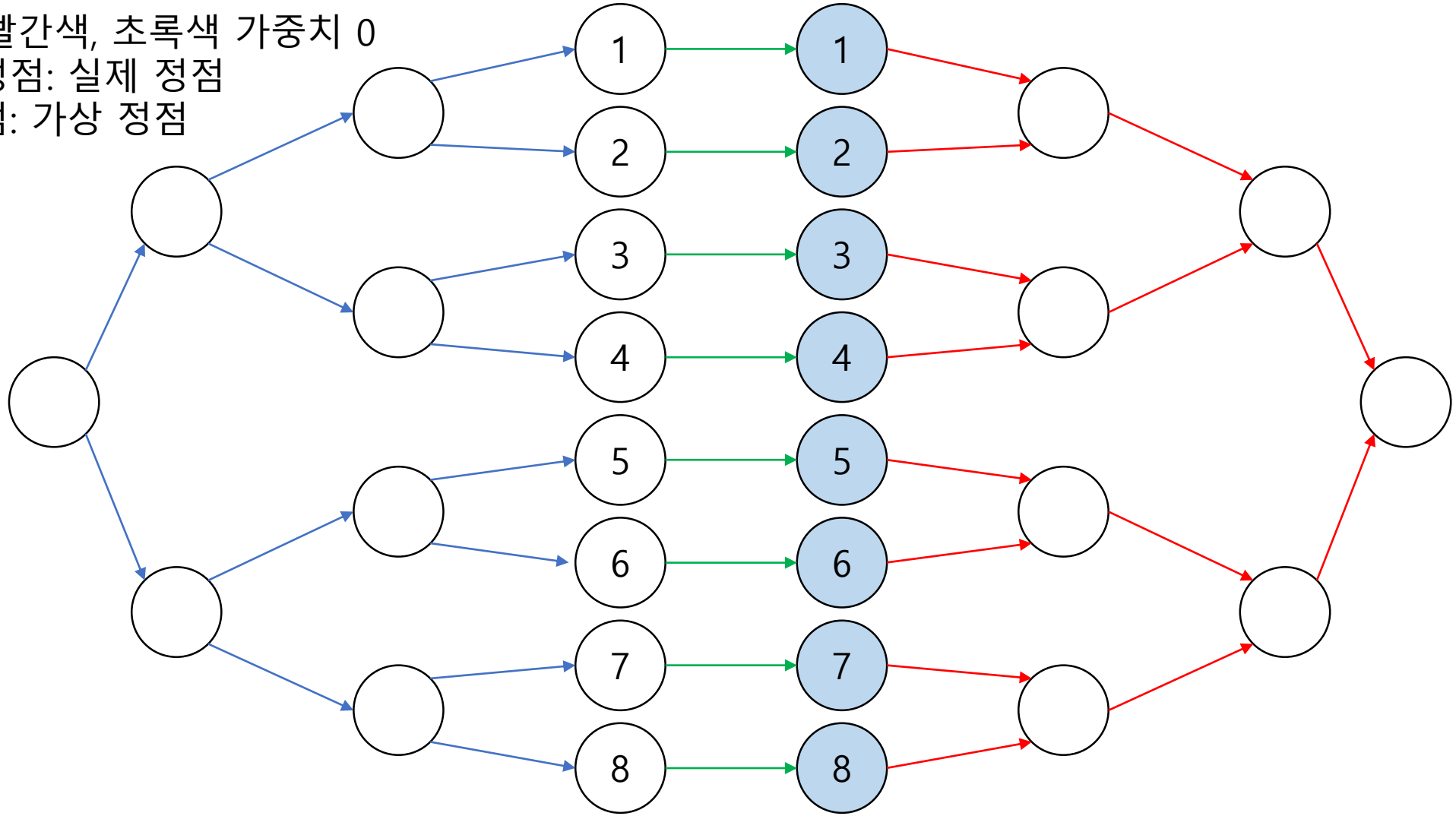
```
for(int i=b; i<=c; i++){
  for(int j=d; j<=e; j++){
    addEdge(i, j, a);
  }
}
dijkstra();
```

비행기 타고 가요

- 간선을 M log^2 N개만 만들 수 있다?
- 구간 + log -> 세그먼트 트리

비행기 타고 가요

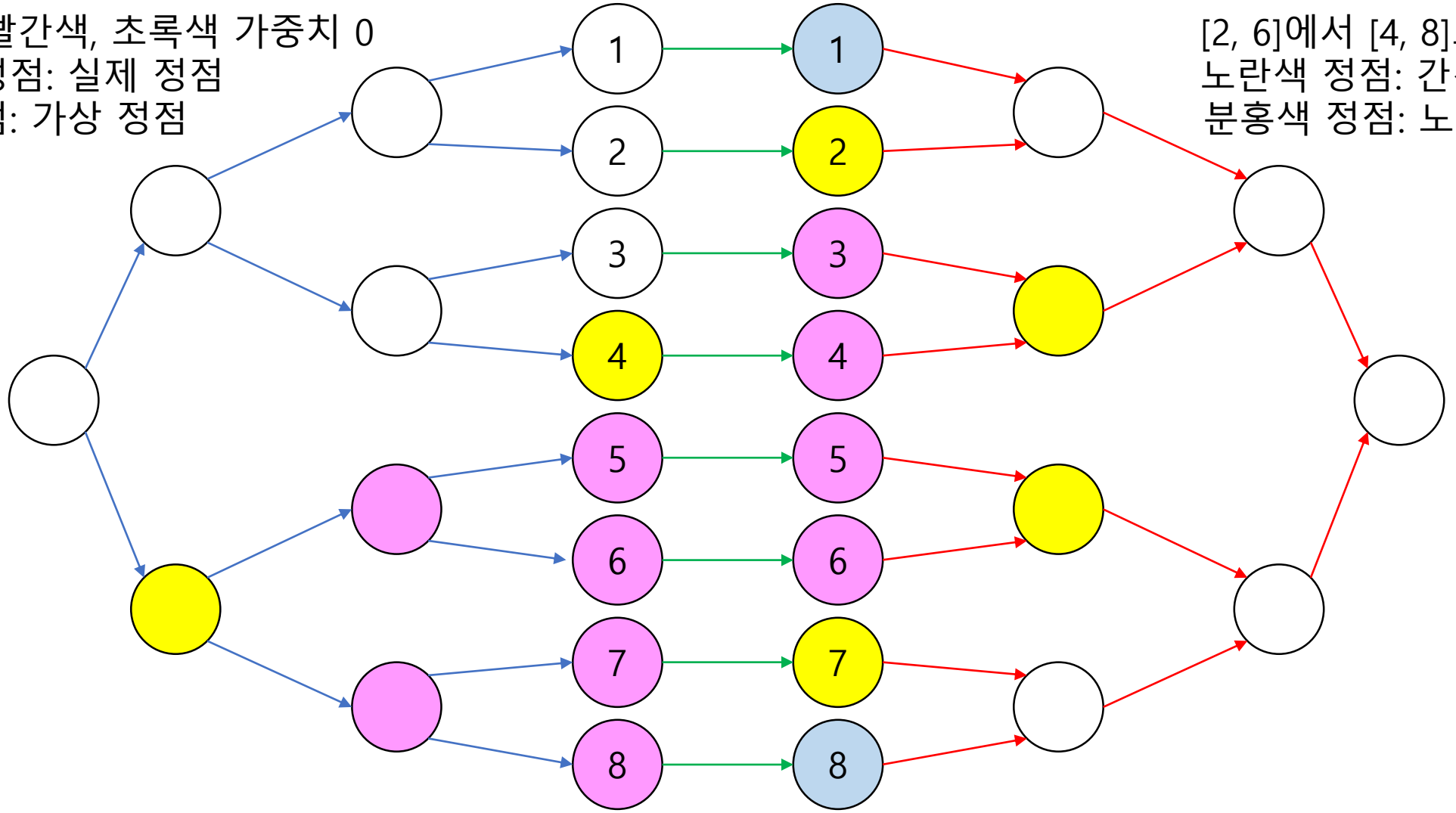
파란색, 빨간색, 초록색 가중치 0
하늘색 정점: 실제 정점
흰색 정점: 가상 정점



비행기 타고 가요

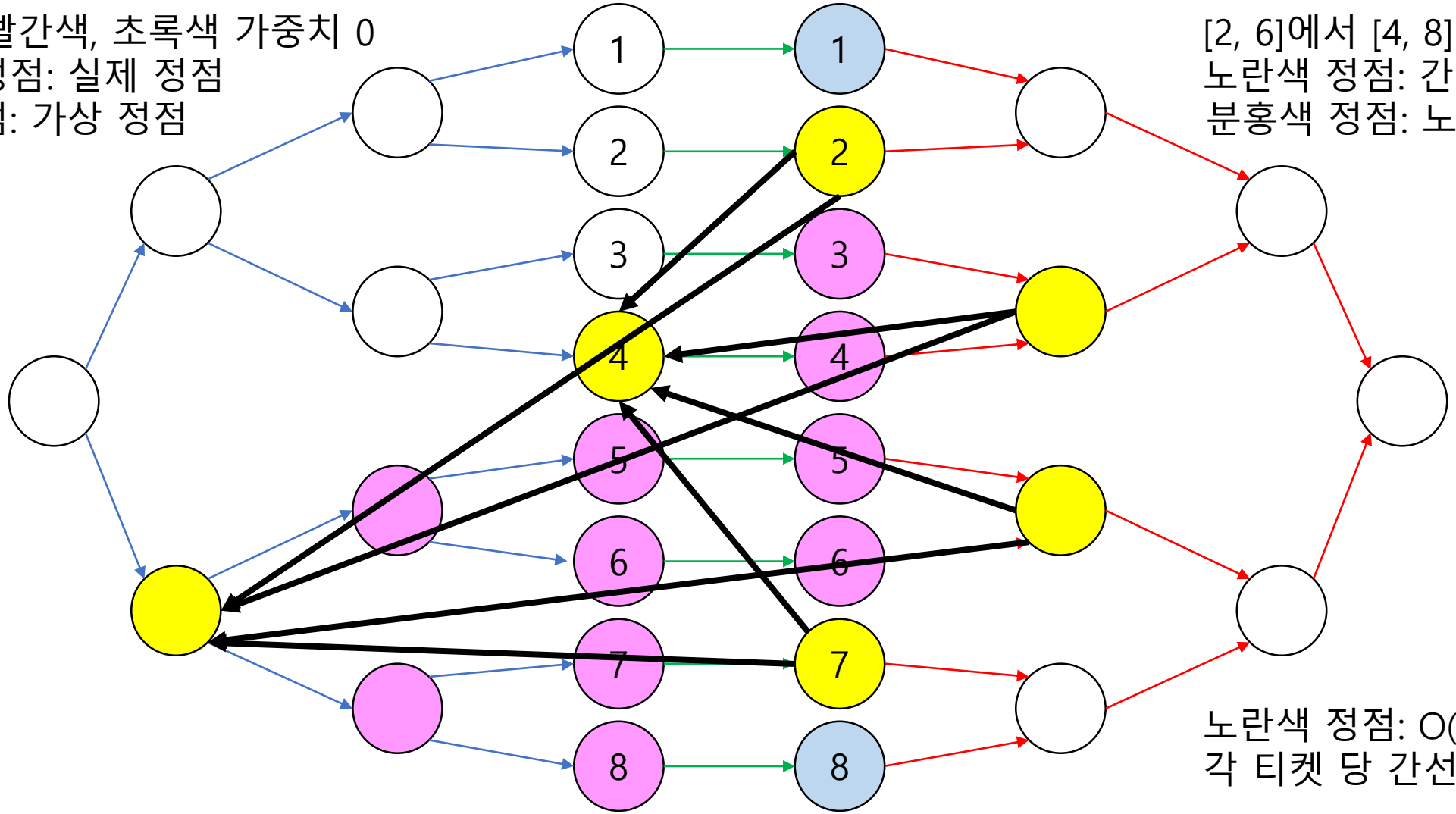
파란색, 빨간색, 초록색 가중치 0
하늘색 정점: 실제 정점
흰색 정점: 가상 정점

[2, 6]에서 [4, 8]로 가는 티켓
노란색 정점: 간선으로 이을 정점
분홍색 정점: 노란색 정점에 영향을 받는 정점



비행기 타고 가요

파란색, 빨간색, 초록색 가중치 0
하늘색 정점: 실제 정점
흰색 정점: 가상 정점



[2, 6]에서 [4, 8]로 가는 티켓
노란색 정점: 간선으로 이을 정점
분홍색 정점: 노란색 정점에 영향을 받는 정점

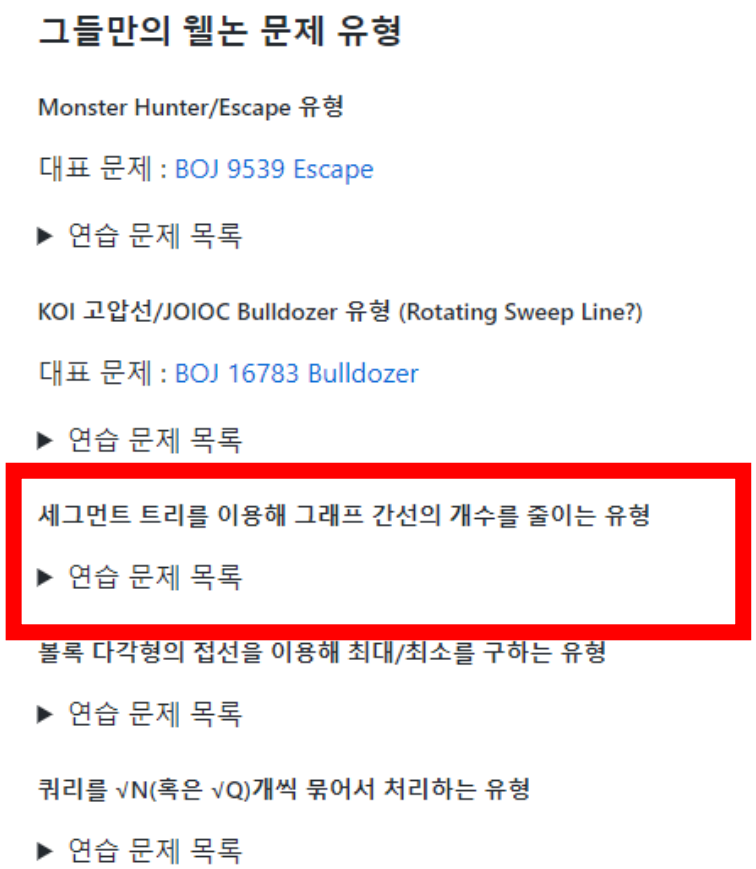
노란색 정점: O(log N)개
각 티켓 당 간선 O(log^2 N)개

비행기 타고 가요

- 정점 O(N)개, 간선 O(M log^2 N)
- O(M log^3 N)에 풀 수 있음

비행기 타고 가요

- https://github.com/justiceHui/Unknown-To-Wellknown



그들만의 웰논 문제 유형

Monster Hunter/Escape 유형

대표 문제 : BOJ 9539 Escape

▶ 연습 문제 목록

KOI 고압선/JOIOC Bulldozer 유형 (Rotating Sweep Line?)

대표 문제 : BOJ 16783 Bulldozer

▶ 연습 문제 목록

세그먼트 트리를 이용해 그래프 간선의 개수를 줄이는 유형

▶ 연습 문제 목록

볼록 다각형의 접선을 이용해 최대/최소를 구하는 유형

▶ 연습 문제 목록

쿼리를 √N(혹은 √Q)개씩 묶어서 처리하는 유형

▶ 연습 문제 목록