Centroid Decomposition 문제풀이

서강대학교 전해성(seastar105)

BOJ 16121번 사무실 이전

문제 요약

크기가 N인 트리가 주어진다. 모든 간선의 길이는 1이다. 거주지인 정점이 M개 주어지고, 후보지인 정점이 K개 주어질 때 모든 거주지와 후보지간의 거리의 제곱의 합을 998,244,353으로 나눈 나머지를 출력하시오.

N <= 300,000, 1 <= M,K <= N

BOJ 16121번 사무실 이전

일단 rooted tree로 생각하자.

어떤 서브 트리에 속해 있는 거주지의 root와의 거리를 A_i, 어떤 서브트리에 속해 있는 후보지의 root와의 거리를 B_i라고 하자.

그러면 거주지와 후보지사이의 경로 중에서 root를 지나는 경로들의 길이의 제곱의 합은 $\sum\sum(A_i + B_j)^2$이 된다.

BOJ 16121번 사무실 이전

$\sum\sum(A_i + B_j)^2$를 일일이 다 계산하면 당연히 시간초과가 난다. 식을 한 번 풀어쓰자.

sum(A)는 A_i의 합, count(A)는 A_i의 개수, sq(A)는 A_i^2의 합으로 정의하자.

그러면 위 식은 아래와 같이 바뀐다.

$\sum\sum(A_i + B_j)^2$ = count(B)*sq(A) + 2*sum(A)*sum(B) + count(A)*sq(B)

따라서, 트리를 순회하면서 sum, count, sq만 잘 갱신하면 답을 구할 수 있다.

BOJ 16121번 사무실 이전

하나의 rooted tree에서 답을 구하는 것은 각 서브트리를 두 번씩 탐색하면 된다.

처음 DFS에서 서브트리를 순회하다가 root와의 거리가 d인 거주지를 만났다고 하자.

그러면 sq(B) + count(B)*d*d + 2*sum(B)*d를 답에 더해주면 된다.

후보지를 만났다면 B만 A로 바꿔서 답을 갱신한다.

그 다음 DFS에서는 A, B에 대한 count, sum, sq를 갱신한다.

BOJ 16121번 사무실 이전

지금까지의 내용은 rooted tree에서 root를 지나는 경로들의 길이의 제곱의 합을 구하는 과정이었다.

Centroid Decomposition을 하면서 위를 진행하면 답을 구할 수 있다. 시간 복잡도는 O(NlogN)이 된다.

BOJ 13514번 트리와 쿼리 5

문제 요약

크기 N인 트리가 주어진다. 다음 두 쿼리를 처리해야한다.

1. i번 정점의 색을 바꾼다. 색은 흰색과 검은색이 있다.
2. 정점 번호 v가 주어진다. v에서 가장 가까운 흰색 정점의 거리를 출력하라. 흰색 정점이 없으면 -1.

정점들은 처음에 검은색이다.

N <= 100,000, Q <= 100,000

사실 요약할 것도 없습니다. 그리고 어렵습니다.

BOJ 13514번 트리와 쿼리 5

먼저 다음과 같은 dp식을 생각해봅시다.

dp[i] = i번을 루트로 하는 서브트리에서 가장 가까운 흰색 정점의 거리

1번 쿼리가 들어오면 부모를 타고 가면서 부모들의 dp값을 수정해주고, 2번 쿼리가 들어오면 부모를 타고 가면서 가장 가까운 거리를 찾을 수 있습니다.

하지만, 어떤 노드의 조상의 최대 개수는 N-1개이기 때문에 시간초과가 납니다.

BOJ 13514번 트리와 쿼리 5

여기서 사용할 수 있는 것이 센트로이드 트리입니다.
센트로이드 트리는 높이가 O(logN)이기 때문에 쿼리가 들어왔을 때 최대 O(logN)개의 조상을 보게 됩니다. 그리고 원래 트리에서의 모든 경로를 나타낼 수 있습니다.

따라서, 센트로이드 트리를 명시적으로 만들어준 뒤에 트리 상에서 dp를 진행하면 답을 구할 수 있습니다.

BOJ 13514번 트리와 쿼리 5

이제 dp[i]는 센트로이드 트리에서 i를 루트로 하는 서브트리에서 가장 가까운 흰색 정점과의 거리가 된다.

1번 쿼리를 생각하자. v가 흰색으로 바뀌는 경우에는 조상을 타고 올라가면서 조상 p에 대해서 dist(p,v)를 dp[p]에 반영해주면 된다.

검은색으로 바뀌는 경우도 마찬가지다.

BOJ 13514번 트리와 쿼리 5

다만, dp[i]는 i의 모든 자식노드에 대해서 삽입 삭제를 지원해야 한다.

이는 multiset이나 priority_queue로 가능하다. 즉, dp값을 관리하기 위해서 정점마다 pq 혹은 multiset을 관리할 필요가 있다.

조상 p와 v사이의 거리를 계산할 때 주의하자. 센트로이드 트리 상에서 정점 간 거리가 아닌 원래 트리 상에서의 거리를 계산해줘야 한다. 이는 LCA를 사용하면 쉽게 계산이 가능하다.

BOJ 13514번 트리와 쿼리 5

1번 쿼리의 시간복잡도를 계산하자.

조상이 O(logN)개 있고, 조상들의 dp값을 수정하는 데에 O(logN)이 걸린다.

조상 p와 쿼리로 들어온 정점 v간의 거리를 계산하는 것은 구현에 따라서 O(logN) 혹은 O(1)이다.

따라서, $O(\log^2 N)$에 쿼리를 처리할 수 있다.

BOJ 13514번 트리와 쿼리 5

2번 쿼리를 생각하자.

조상을 타고 올라가면서 조상 p에 대해서 min(dp[p]+dist(p,v))를 계산하면 답이 나온다.

BOJ 13514번 트리와 쿼리 5

시간복잡도는 조상들을 모두 보는 데에 O(logN), pq로 구현시에는 dp값을 계산하는 데에 O(logN), multiset이라면 O(1)이 걸린다. dist 계산도 O(logN)이다.

따라서, $O(\log^2 N)$ 혹은 O(logN)에 가능하다.

Problem Set

- Essential
  - 20297 - Confuzzle
  - 16121 - 사무실 이전
  - 13514 - 트리와 쿼리 5
- Practice
  - 5820 - 경주
  - 13431 - 트리 문제
  - 14176 - 트리와 소수
  - 13513 - 트리와 쿼리 4