SCV/2021

Dessert Café

- Tree DP, Graph Traversal -

안태진(taejin7824@gmail.com)
GitHub(github.com/taejin1221)
상명대학교 소프트웨어학과
201821002

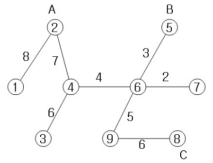
Contents

Problem

- Description
  - ICPC Seoul Regional 2020 C번 문제
    - 다른 말: ICPC 본선 C번
  - 트리가 주어졌을 때 Good Place의 개수를 구하는 문제
- Comment
  - 작년 상명대 팀이 5번이나 틀렸던...
  - 조건을 잘못 생각하면 굉장히 많이 틀림
    - 이 함정에 모두 빠졌을 것이라 생각

문제

Kim, who wishes to start a business, is trying to open a dessert cafe he has been preparing after graduating from college. The road network in the town where Kim lives forms a tree structure, that is, a connected acyclic graph as shown in the figure below. There are n candidate sites for a dessert café in the town. In the figure below, a circle represents a candidate site for a dessert café, a line segment between two candidate sites represents a road, and the value labeled on a line segment represents the length of a road.



There are k apartment complexes in this town, so he wants his dessert café to be located as close as possible to an apartment complex. In above figure, there are three apartment complexes which are located to the candidate sites labeled by A, B, and C. Considering the competitiveness and profitability, he thinks that a candidate site satisfying the following condition is a good place.

Let $d(x, y)$ be the length of the shortest path on a road network between two candidate sites x and y. A candidate site p is a good place if there exists a candidate site z where an apartment complex is located such that $d(p, z) < d(q, z)$ for each candidate site q ($\neq p$).

In above figure, candidate sites 2, 4, 5, 6, 8, and 9 are good places. More specifically, for example, candidate 6 is a good place because it is closer to apartment complex B than any other candidate sites except for candidate 5, and is closer to apartment complex A than candidate 5. That is, there exists apartment complex B on candidate 5 satisfying $d(6, 5) < d(q, 5)$ for $q \in \{1, 2, 3, 4, 7, 8, 9\}$, and there exists apartment complex A on candidate 2 satisfying $d(6, 2) < d(5, 2)$. Candidate 7 is not a good place because none of apartment complexes are closer than candidate 6.

Given the information on candidate sites and apartment complexes in the town, write a program to output the number of good places.

Contents

- Problem

- Solution
  - Condition
  - Analysis 1
  - Analysis 2
  - Solution

- Code

Solution

- Condition
  - Good Place가 되기 위한 조건
    - Let $d(x, y)$ be the length of the shortest path on a road network between two candidate sites x and y. A candidate site p is a good place if there exists a candidate site z where an apartment complex is located such that $d(p, z) < d(q, z)$ for each candidate site q ($\neq p$).
    - 모든 q가 아닌 p에 대하여 d(p,z) < d(q,z)를 만족하는 z(아파트)가 존재한다면 Good Place
    - 수식
      - $\forall q(\neq p), \exists z\ s.t.\ d(p,z) < d(q,z)$

Contents

- Problem

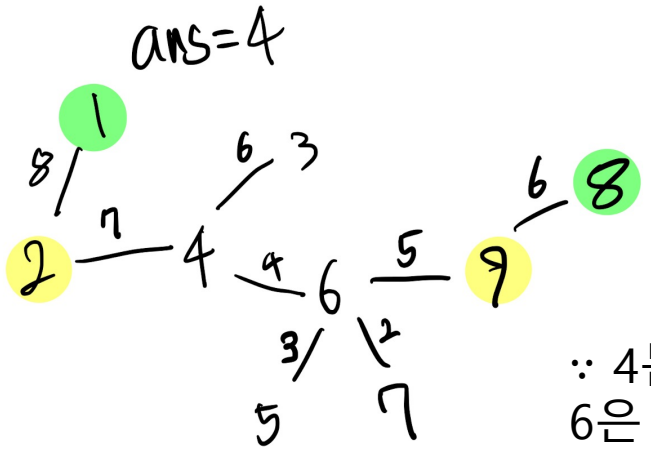
- Solution
  - Condition
  - Analysis 1
  - Analysis 2
  - Solution

- Code

Solution

- Analysis 1
  - 그렇다면 아파트는 일단 Good Place
  - 아파트 기준 한 칸 떨어진 노드 중 가장 가까운 아파트 아닌 노드가 Good Place

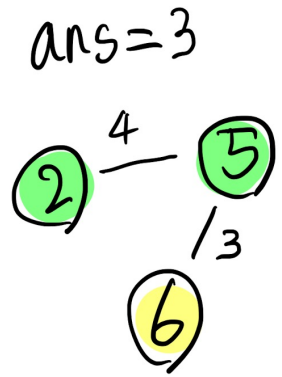
  - 아래 그래프의 답은 4



: Good Place
: Apart Complex

∵ 4는 2보다 1과 가깝지 않고,
6은 9보다 8에 가깝지 않음

- 아래 그래프의 답은 3



∵ 6은 5와 제일 가까운 노드

Solution

- Analysis 1
  - 우리가 간과한 것
    - $\forall q(\neq p), \exists z \; s.t. \; d(p,z) < d(q,z)$

1. 모든 q에 대해 z가 같을 필요가 있을까?
   - 모든 q에 대해 어떤 z가 존재하기만 하면 됨
   - 한 노드에 대해
     - d(4, 1) < d(9, 1)와 d(4, 8) < (2, 8)이 가능함
2. q는 p가 아님 (즉, z도 포함)

Contents

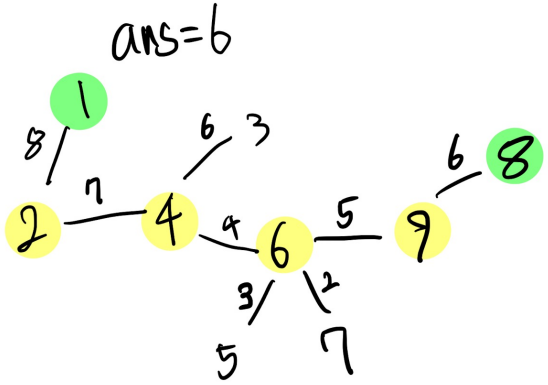
- Problem

- Solution
  - Condition
  - Analysis 1
  - Analysis 2
  - Solution

- Code

Solution

- Analysis 2
  - $\forall q(\neq p), \exists z\ s.t.\ d(p,z) < d(q,z)$
  - 아래 그래프의 답은 6



- 즉, 모든 $q \in \{1,2,3,5,6,7,8,9\}$에 대해 $d(p,z) < d(q,z)$를 만족하는 어떤 apart는 항상 존재

∵ 4는 1, 2보다 가까운 아파트가 존재 (8)
4는 3, 5, 6, 7, 8, 9 보다 가까운 아파트가 존재 (1)

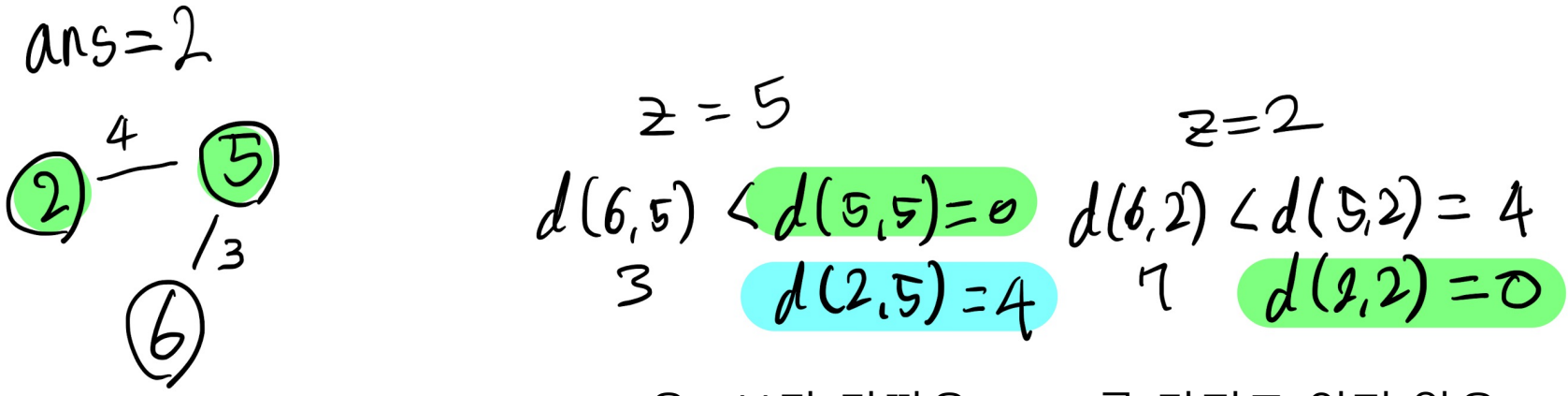
z=1
d(4,1) < d(1,1)=0
d(2,1)=8
d(3,1)=21
d(4,1)=15
d(5,1)=22
d(6,1)=19
d(7,1)=21
d(8,1)=30
d(9,1)=24

z=8
d(4,8) < d(1,8)=30
d(2,8)=22
d(3,8)=21
d(4,8)=15
d(5,8)=14
d(6,8)=11
d(7,8)=13
d(8,8)=0
d(9,8)=6

: p
: z
: d(p, z) < d(q, z)를 만족

Solution

- Analysis 2
  - 아래 그래프의 답은 2



∵ 6은 5보다 가까운 apart를 가지고 있지 않음

- 즉, 모든 $q = 5$에 대해
$d(p, z) < d(q, z)$를 만족하는 어떤 apart가 존재 하지 않음

Solution

- Analysis 2
  - 사실 다 알려주고 있었음

In above figure, candidate sites 2, 4, 5, 6, 8, and 9 are good places. More specifically, for example, candidate 6 is a good place because it is closer to apartment complex B than any other candidate sites except for candidate 5, and is closer to apartment complex A than candidate 5. That is, there exists apartment complex B on candidate 5 satisfying $d(6, 5) < d(q, 5)$ for $q \in \{1, 2, 3, 4, 7, 8, 9\}$, and there exists apartment complex A on candidate 2 satisfying $d(6, 2) < d(5, 2)$. Candidate 7 is not a good place because none of apartment complexes are closer than candidate 6.

Contents

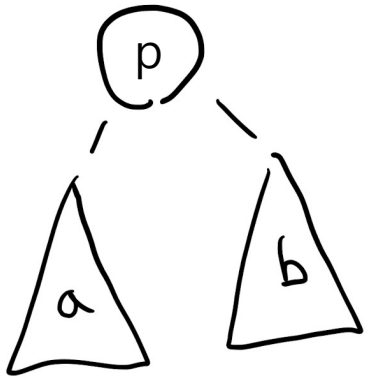
- Problem

- Solution
  - Condition
  - Analysis 1
  - Analysis 2
  - Solution

- Code

Solution

- Solution
  - 답은 $\forall q(\neq p), \exists z\ s.t.\ d(p,z) < d(q,z)$ 를 만족하는 p를 찾기
  - $d(a,b)$
    - 주어진 Graph는 Tree이기 때문에 a, b로 가는 경로가 1개이자 Shortest Path
    - ∴ 오른쪽 그림의 p의 subtree들이 서로 도달하려면 p를 지나야 함
  - 답
    - p가 apart를 포함한 subtree가 2개 이상이라면 p는 Good Place
    - ∵ $\forall q(\neq p), \exists z\ s.t.\ d(p,z) < d(q,z)$ 에서
      오른쪽 subtree의 노드들의 z를 왼쪽 apart로 설정,
      왼쪽 subtree의 노드들의 z를 오른쪽 apart로 설정 한다면 무조건 만족



가중치는 의미가 없다!

Solution

- Solution
  - Algorithm
    - apart부터 dfs를 시작하여 apart를 포함하는 subtree의 개수 count
    - 모든 노드를 탐색하며 apart이거나 count가 1보다 크거나 같은 노드의 수가 answer
      - ∵ 자신의 부모에 무조건 apart가 존재하므로, count의 개수가 1개 이상이면
        자신과 연결된 subtree들 중 apart가 존재하는 subtree는 2개 이상인 것

Contents

- Problem
- Solution
  - Condition
  - Analysis 1
  - Analysis 2
  - Solution
- Code

Code

- Include header file and declare global variable
  - isApart[i]: Node i가 apart인가?
  - hasApartSub[i]: Node i의 subtree 중 apart를 가지고 있는 subtree의 개수
  - adjList: adjacent list
    - 가중치는 입력하지 않음

```cpp
#include <iostream>

#include <vector>

#define MAX 100'001

using namespace std;

bool isApart[MAX];
int hasApartSub[MAX];
vector<int> adjList[MAX];
```

Code

- main function
  1. 빠른 입출력
  2. Tree 입력
  3. Apart 입력
     - 가장 작은 apart 찾기
       - apart부터 dfs를 시작하여야하기 때문
       - 아무 apart여도 상관 X

```
int main(void) {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);

    int n, k;
    cin >> n >> k;

    for (int i = 0; i < n - 1; i++) {
        int a, b, c;
        cin >> a >> b >> c;

        adjList[a].push_back(b);
        adjList[b].push_back(a);
    }

    int minApart = MAX;
    for (int i = 0; i < k; i++) {
        int apart;
        cin >> apart;

        isApart[apart] = true;
        minApart = min(minApart, apart);
    }
```

Code

- dfs function
  - 자신의 subtree 중 apart를 포함하는 subtree를 세기

```
dfs(minApart, -1);
```

```
int dfs(int me, int mom) {
    for (int& adj : adjList[me]) {
        if (adj != mom)
            if (dfs(adj, me))
                hasApartSub[me]++;
    }

    return isApart[me] ? 1 : hasApartSub[me];
}
```

Code

- answer
  - 아파트이거나, subtree 중 apart를 가진 subtree가 1개 이상인 노드 ++

```
int ans = 0;
for (int i = 1; i <= n; i++)
    if (isApart[i] || hasApartSub[i] >= 1)
        ans++;

cout << ans << '\n';
```

감사합니다!