

Day10 기출 풀이

선린인터넷고등학교 소프트웨어과

30610 나정휘

<https://justicehui.github.io>

문제 목록

- BOJ10166 관중석 (KOI'14 고등 #1)
- BOJ5525 IOIOI (JOI'09 #1)
- BOJ2655 가장높은탑쌓기 (KOI'98 중등 #3)
- BOJ5502 팰린드롬 (IOI'00 Day1 #1)
- BOJ18780 Timeline (USACO '20 Feb G1)
- BOJ2673 교차하지 않는 원의 현들의 최대집합 (KOI'96 고등 #2)
- BOJ5467 Type Printer (IOI'08 Day1 #1)
- BOJ10167 금광 (KOI'14 중등 #4)

관중석

- 각도가 같다 -> (좌석 번호) / (해당 원의 좌석 개수)
 - $0/3, 1/3, 2/3$
 - $0/4, 1/4, 2/4, 3/4$
 - $0/5, 1/5, 2/5, 3/5, 4/5$
 - $0/6, 1/6, 2/6, 3/6, 4/6, 5/6$
 - $1/3 = 2/6, 2/3 = 4/6, \dots$
- 서로 다른 기약 분수의 개수를 구하는 문제

관중석

- $A[i][j] = 1$ 이고 i 와 j 가 서로소일 때 i/j 형태의 분수가 존재하는가?
- 열심히 구현하면 된다.

IOIOI

- S에서 IOIOI...I들의 길이를 뽑아서 저장하자.
 - 00IOIOIOIOIOIOII
 - 00 / IOIOIOIOI / IOI / I
- P_n의 길이는 $2n+1$ 이다. 위에서 뽑아낸 문자열들의 길이를 이용해 정답을 잘 구해주면 된다.

가장높은탑쌓기

- 넓이를 기준으로 정렬하고 DP를 돌리자.
 - $D(i) = \max\{ D(j) + \text{Height}[i] \} \quad (\text{Weight}[j] > \text{Weight}[i])$
 - 이때 각 i 마다 $(D(j) + \text{Height}[i])$ 가 최대가 되는 j 를 $\text{prv}[i]$ 라고 하자.
- $D(i)$ 가 최대가 되는 i 를 찾은 뒤, $\text{prv}[i]$ 값들을 이용해 역추적을 해주면 된다.

팰린드롬

- N - LCS(문자열, 뒤집은 문자열)

- LCS(S, T)

- $D(i, j) = D(i-1, j-1) + 1$

$$S[i] = T[j]$$

- $D(i, j) = \max\{ D(i-1, j), D(i, j-1) \}$

$$S[i] \neq T[j]$$

Timeline

- S_i 조건이 없다고 생각하자.
- (a, b, x) 라는 정보는 b 가 a 보다 최소 x 일 이상 늦게 시작한다는 것을 의미한다.
 - a 에서 b 까지 가는 가중치가 x 인 간선을 만들자.
- C 개의 순서쌍을 모두 만족하는 각 세션의 최소 날짜는 각 정점까지의 "최장 거리"와 동치이다.

Timeline

- 그래프 모델링 아이디어만 있다면, s_i 조건을 처리하는 방법은 간단하다.
- 0일차에 열리는 0번 세션을 만들고
- 0번 정점에서 i 번 정점까지 가는 가중치 s_i 간선을 만들면 된다.
- $O(N+C)$ 에 풀 수 있다.

교차하지 않는 원의 현들의 최대집합

- 구간에 대한 DP
- $D(i, j) = \max\{ D(i, k) + D(k+1, j) + C(i, j) \}$
 - $C(i, j) = i$ 와 j 를 잇는 현이 존재하면 1, 존재하지 않으면 0
- $O(100^3)$

Type Printer

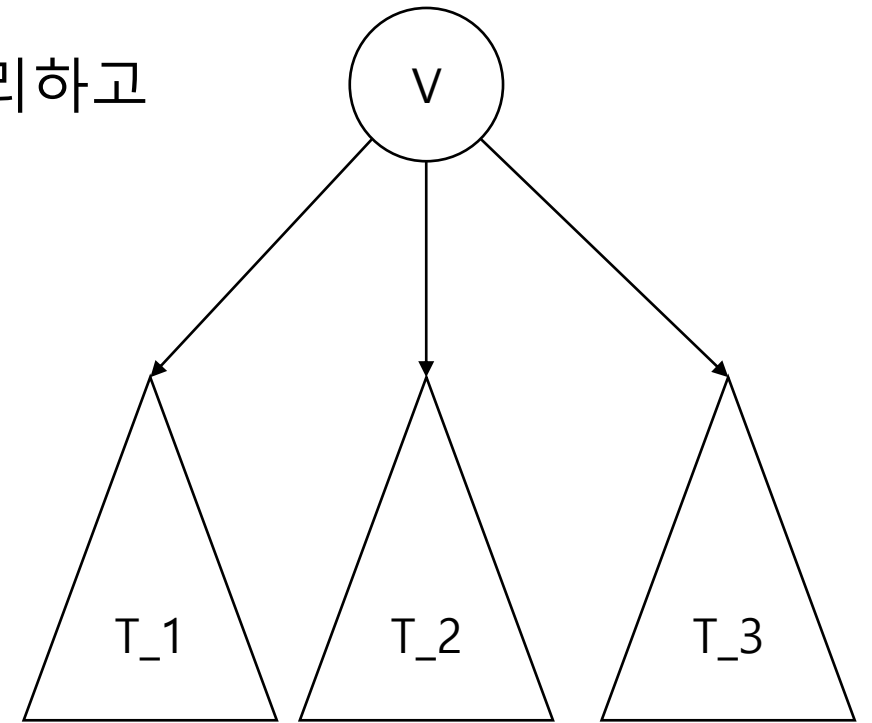
- 출력할 순서만 잘 정해주면 그 다음은 단순 구현 문제다.

Type Printer

- Trie를 만들자.
- 명령 개수 최소화
 - Trie에서 문자열의 마지막 글자를 나타내는 정점들을 순회하면서
 - Trie 상에서 이동 거리를 최소화하는 것

Type Printer

- 귀납적으로 생각하자.
 - T_1, T_2, T_3 와 같은 서브 트리는 귀납적으로 처리하고
 - V 에서 서브 트리들을 방문할 순서를 정해보자.
- V 에서 $T_{\{P1\}}$ 에 들어감
- $T_{\{P1\}}$ 에서 V 로 빠져나옴
- $T_{\{P2\}}$ 에 들어갔다가 나옴
- $T_{\{P3\}}$ 에 들어감
- $T_{\{P3\}}$ 에서 종료하거나 빠져나옴
- 서브 트리의 깊이 순서대로 정렬해주면 된다.



Type Printer

```
struct TrieNode{
    TrieNode *ch[26] = {0};
    int ter, len;
    TrieNode() : ter(-1), len(0) {}

    void insert(const char *key, int num, int ln){
        if(*key == 0){
            ter = num; len = max(ln, len); return;
        }
        if(!ch[*key - 'a']) ch[*key - 'a'] = new TrieNode();
        ch[*key - 'a']->insert( key: key+1, num, ln);
        len = max(len, ln);
    }
};
```

```
vector<int> order;
void dfs(TrieNode *v){
    if(v->ter != -1){
        order.push_back(v->ter);
    }
    sort(v->ch, last: v->ch+26, comp: [&](TrieNode *a, TrieNode *b){
        if(a == 0) return false;
        if(b == 0) return true;
        return a->len < b->len;
    });
    for(int i=0; i<26; i++){
        if(v->ch[i]) dfs(v->ch[i]);
    }
}
```

금광

- 직사각형의 각 변에 하나 이상의 금광이 걸쳐 있는 경우만 봐도 정답을 찾을 수 있다.
- x, y 좌표를 각각 압축해주면 서로 다른 x, y 좌표 값이 최대 $2N(= 6000)$ 개씩 존재한다.

금광

- $O(N^3)$ 풀이
 - 직사각형의 가로 변 2개를 선택하는 경우는 $O(N^2)$ 가지이다.
 - 최대 부분합 문제로 바뀌게 되고, 각 경우에 대해 $O(N)$ 에 풀 수 있다.
 - 시간 초과가 난다.
- 가로 변을 선택하는 건 필수적이므로 최대 부분합 문제를 빠르게 풀 방법을 생각해야 한다.

금광

- 최대 부분합 문제는 분할 정복으로 $O(N \log N)$ 에 풀 수 있다.
- 세그먼트 트리는 분할 정복을 메모이제이션하는 자료구조이다.
- 두 개를 합치자.

금광

- 세그먼트 트리로 최대 부분합 문제를 푸는 것은 쉽다.
- 각 정점마다
 - 구간의 왼쪽 끝 점을 포함하는 최대 부분합
 - 구간의 오른쪽 끝 점을 포함하는 최대 부분합
 - 구간의 합
 - 구간 내에서 최대 부분합
- 을 저장하고 있으면 최대 부분합 문제를 풀 수 있다.

금광

- 이제 정해를 알아보자.
- 일단 점들을 y 좌표 기준으로 정렬한다.
- 직사각형의 가로 변 하나($= y_1$)를 고정하자.
- y 좌표가 y_1 이상인 점들을 세그먼트 트리에 순서대로 넣어주면서, 동시에 최대 부분합을 구해 최댓값을 갱신해주면 된다.
 - y 좌표가 같은 점들은 한 번에 넣어야 하는 것을 주의해야 한다.
- 세그먼트 트리에 $O(N^2)$ 번 쿼리를 날리므로 $O(N^2 \log N)$