DP Optimization 3

나정휘

https://justiceHui.github.io/

목차

- Convex Hull Trick
- Li Chao Tree
- Hirschberg’s Algorithm
- Divide and Conquer Optimization
- Monotone Queue Optimization
- Aliens Trick

Aliens Trick

Aliens Trick

- 간단한 인터랙티브 문제
  - 아래로 볼록한 함수 f(x)가 있을 때, x_0가 주어지면 $f(x_0)$를 구하는 문제
  - f가 볼록하다는 것 외에는 알고 있는 정보가 없음
  - 할 수 있는 쿼리는 실수 c에 대해 f(x) + cx가 최소인 좌표를 얻는 것

  - 어떻게 해결해야 할까?

Aliens Trick

- 간단한 인터랙티브 문제
  - 아래로 볼록한 함수 $f(x)$, $f(x) + cx$가 최소인 좌표를 구하는 쿼리, $f(x_0)$의 값을 구하는 문제
  - 예시
    - $f(x) = (x-3)^2 + 1 = x^2 - 6x + 10$, $x_0 = 2$
    - $c = -2$ 이면 $(4, -6)$
    - $c = 0$ 이면 $(3, 1)$
    - $c = 2$ 이면 $(2, 6)$
    - $c = 4$ 이면 $(1, 9)$
    - $f(x) + c_0x$일 때 극점이 (x_0, y_0)이면 $f(x_0) = y_0 - c_0x_0$
  - 이분 탐색으로 적당한 c값을 찾을 수 있음
    - $g(x) = f(x) + cx$의 도함수는 $g'(x) = f'(x) + c$임
    - $c = -f'(x_0)$을 찾으면 되고, 볼록 함수는 기울기가 증가하므로 이분 탐색 가능

Aliens Trick

- 간단한 인터랙티브 문제
  - 아래로 볼록한 함수 f(x), f(x) + cx가 최소인 좌표를 구하는 쿼리, $f(x_0)$의 값을 구하는 문제

  - 기울기가 변하지 않는 구간이 있는 경우
    - $f(x) = \begin{cases} -2x + 5, & x \le 1 \\ -x + 4, & 1 \le x \le 4 \\ x - 4, & 4 \le x \end{cases}$
    - $f(x) + x = \begin{cases} -x + 5, & x \le 1 \\ 4, & 1 \le x \le 4 \\ 2x - 4, & 4 \le x \end{cases}$
    - 최소 지점이 유일하지 않을 수 있음
    - 기울기가 0인 구간은 유일하므로 동일하게 이분 탐색으로 해결 가능
      - ex. 최소 지점이 x_0 이상인 가장 큰 c 찾기

질문?

Aliens Trick

- DP에서의 활용
  - 점화식이 다음과 같은 형태일 때, 시간 복잡도에서 K 대신 log X를 붙일 수 있음
    - D(k, i) = i번째 원소까지 고려했을 때, k개의 원소(또는 구간 등)을 선택하는 최소 비용
    - $D(k+1, n) - D(k, n) \le D(k+2, n) - D(k+1, n)$
      - 즉, f(k) = D(k, n)일 때 f가 아래로 볼록해야 함
    - 최대 비용을 구해야 하는 경우 $D(k+1, n) - D(k, n) \ge D(k+2, n) - D(k+1, n)$

Aliens Trick

- BOJ 19672 Feast
  - 길이가 N인 수열이 주어짐
  - 구간이 서로 겹치지 않도록 K개의 구간을 만들 때, 구간에 포함된 수들의 합을 최대화
  - $K \le N \le 300'000$, 길이가 0인 구간도 가능

  - 점화식
    - D(k, i, 0) = i번째 원소까지 k개의 구간을 선택, i번째 원소는 구간에 포함 안 됨
    - D(k, i, 1) = i번째 원소까지 k개의 구간을 선택, i번째 원소는 구간에 포함됨
    - $D(k, i, 0) = \max\{ D(k, i-1, 0), D(k, i-1, 1) \}$
    - $D(k, i, 1) = \max\{ D(k-1, i-1, 0), D(k, i-1, 1) \} + A[i]$
    - 시간 복잡도 O(KN), 토글링하면 공간 복잡도 O(N)

Aliens Trick

- BOJ 19672 Feast
  - f(k) = D(k, n)은 위로 볼록한 함수
    - 구간을 추가할 때마다 얻는 이득이 매번 감소한다는 것은 직관적으로 알 수 있음
    - 따라서 Aliens Trick 적용 가능

  - $D_c(n)$ = n번째 원소까지 고려했을 때 최대 비용, 구간을 하나 만들 때마다 비용 c 추가
    - 구간의 개수(k)가 증가할 때마다 비용이 c 만큼 증가
    - $D_c(n)$을 계산하는 것 = f(k) + ck의 극점을 구하는 것
    - 위로 볼록한 함수의 함숫값을 구하는 것은 이분 탐색으로 가능

    - c가 양의 무한대라면 구간을 n개 만드는 것이 최적
    - c가 음의 무한대라면 구간을 1개 만드는 것이 최적
    - 이분 탐색을 이용해 구간을 k개 만드는 것이 최적이 되도록 c를 조절

Aliens Trick

- BOJ 19672 Feast
  - $D_c(n)$ = n번째 원소까지 고려했을 때 최대 비용, 구간을 하나 만들 때마다 비용 c 추가
    - $D_c(i, 0) = \max\{ D_c(i-1, 0), D_c(i-1, 1) \}$
    - $D_c(i, 0) = \max\{ D_c(i-1, 0) + c, D_c(i-1, 1) \} + A[i]$
  - 구간의 개수도 함께 저장
    - 크기가 0인 구간도 선택할 수 있으므로, 최댓값이 같다면 구간의 개수가 작은 값을 저장

Aliens Trick

```cpp
#include <bits/stdc++.h>
using namespace std;
using ll = long long;

struct Data{
    ll v, c; // dp, cnt
    Data() : Data(0, 0) {}
    Data(ll v, ll c) : v(v), c(c) {}
    bool operator < (const Data &t) const { return v != t.v ? v < t.v : c > t.c; }
    Data operator + (const Data &t) const { return {v + t.v, c + t.c}; }
};

ll N, K, A[303030];
Data D[303030][2];

Data Solve(ll c){
    D[0][0] = {0, 0}; D[0][1] = {(ll)-1e18, 0};
    for(int i=1; i<=N; i++){
        D[i][0] = max(D[i-1][0], D[i-1][1]);
        D[i][1] = max(D[i-1][0] + Data(A[i]+c, 1), D[i-1][1] + Data(A[i], 0));
    }
    return max(D[N][0], D[N][1]);
}

int main(){
    ios_base::sync_with_stdio(false); cin.tie(nullptr);
    cin >> N >> K;
    for(int i=1; i<=N; i++) cin >> A[i];
    ll l = -1e15, r = 0, res = 1e18;
    while(l < r){
        ll m = (l + r + 1) >> 1;
        if(Solve(m).c <= K) l = m;
        else r = m - 1;
    }
    cout << Solve(l).v - K*l;
}
```

질문?

Aliens Trick의 충분 조건

Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - DP값이 볼록하다는 것을 직접 증명하는 것은 어려움
  - DnC Opt에서의 monge 조건처럼 aliens에도 자주 등장하고 유용한 충분 조건 몇 개 있음

  - MCMF
    - D(k, n) = 유량을 k 만큼 흘리는 최소(또는 최대) 비용으로 모델링되는 경우
    - Minimum Cost Flow에서 증가 경로의 비용은 단조 증가하므로 볼록함
    - BOJ 19672 Feast는 MCMF로 모델링할 수 있음
      - 정점 분할한 다음 { In(v), Out(v), 1, A[i] }, { Out(v-1), In(v), 1, 0 } 간선 추가
      - { S, T, K, 0 }, { S, In(v), 1, 0 }, { Out(v), T, 1, 0 } 간선 추가
      - 유량을 K 만큼 흘릴 때 최대 비용을 구하면 됨

Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k) = D(k, n)$은 볼록함
    - https://koosaga.com/243 의 Theorem 1 참고

    - $D(k, n)$ = 어떤 DAG에서 간선을 k개 사용해서 0번 정점에서 n번 정점으로 가는 최단 경로의 길이
      - $0 \le i < j \le n$에 대해, $i \rightarrow j$ 간선의 비용은 $C(i+1, j)$
      - $w(P)$를 P의 길이, $L(P)$를 P의 간선 개수라고 정의하자.
    - 두 간선 $e_1 = (b, c)$과 $e_2 = (a, d)$가 $a \le b < c \le d$를 만족하면 e_2가 e_1을 포함한다고 하자.
    - "경로 교환"이라는 연산을 다음과 같이 정의하자.
      - 각각 e_1가 e_2를 포함하는 두 경로 $P_1 = \{0, \dots, b, c, \dots, n\}$과 $P_2 = \{0, \dots, a, d, \dots, n\}$ 이 있을 때
      - $Q_1 = \{0, \dots, a, c, \dots, n\}$과 $Q_2 = \{0, \dots, b, d, \dots, n\}$으로 수정하는 것
      - C가 monge matrix이면 $w(P_1) + w(P_2) \ge w(Q_1) + w(Q_2)$
        - monge matrix: $C(a+1, c) + C(b+1, d) \le C(a+1, d) + C(b+1, c)$

Aliens Trick의 충분 조건

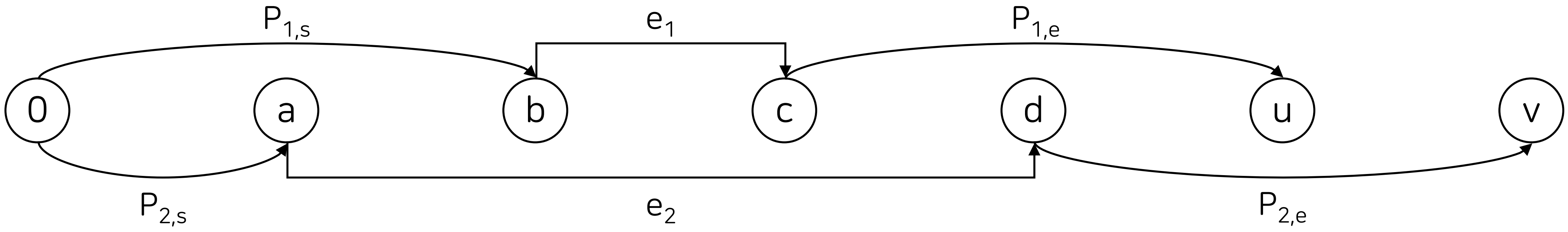
- Aliens Trick의 충분 조건
  - Monge Matrix
    - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k) = D(k, n)$은 볼록함

    - 목표: 길이가 k+1, k-1인 두 최단 경로 P_1, P_2의 간선을 적절히 교환해 $L(Q_1) = L(Q_2)$인 두 경로 생성
      - $2f(k) \le w(Q_1) + w(Q_2) \le w(P_1) + w(P_2) = f(k-1) + f(k+1)$이 되므로 f가 볼록하다는 것을 보일 수 있음
      - 따라서 항상 이런 경로 교환이 가능한 두 간선 $e_1 \in P_1$, $e_2 \in P_2$가 존재한다는 것을 보이면 됨

Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k) = D(k, n)$은 볼록함
    - 목표: 길이가 k+1, k-1인 두 최단 경로 P_1, P_2의 간선을 적절히 교환해 $L(Q_1) = L(Q_2)$인 두 경로 생성

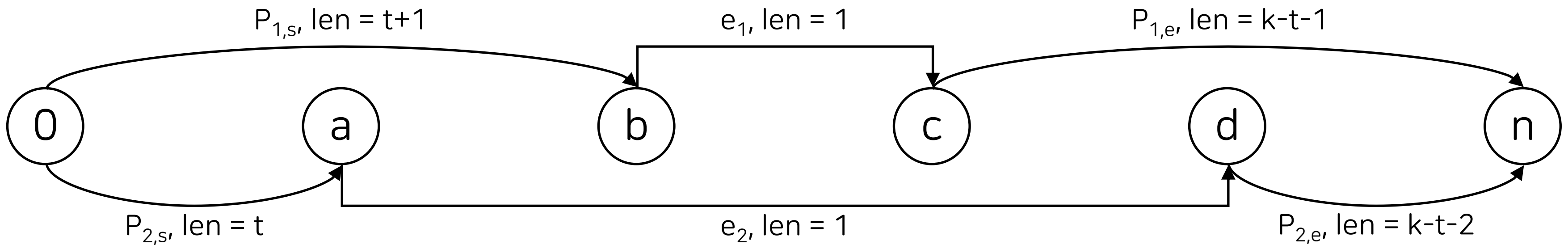
    - $u \le v$에 대해, $0 \to u$ 경로 P_1과 $0 \to v$ 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \ge k_2$이면 임의의 $0 \le x \le k_2 - k_1$에 대해, 아래 조건을 만족하는 P_1의 간선 e_1, P_2의 간선 e_2 존재
      - $e_1 = (b, c)$ $e_2 = (a, d)$라고 할 때, $a \le b < c \le d$를 만족함 (e_2가 e_1을 포함)
      - P_1을 따라 0에서 b까지 가는 경로의 간선 개수 = P_2를 따라 0에서 a까지 가는 경로의 간선 개수 + x
      - 즉, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재



Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k) = D(k, n)$은 볼록함
    - 목표: 길이가 k+1, k-1인 두 최단 경로 P_1, P_2의 간선을 적절히 교환해 $L(Q_1) = L(Q_2)$인 두 경로 생성

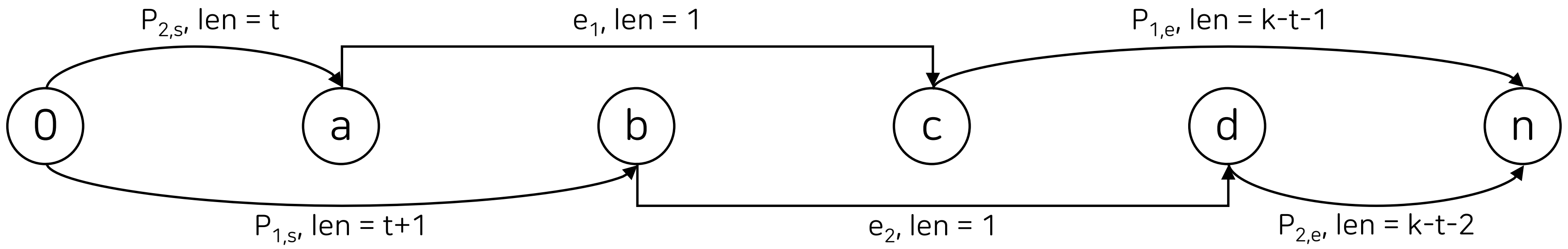
    - $u \le v$에 대해, $0 \to u$ 경로 P_1과 $0 \to v$ 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \ge k_2$이면 임의의 $0 \le x \le k_2-k_1$에 대해, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재
      - k1 = k+1, k2 = k-1이고 u = v = n일 때 x = 1인 간선 $e_1 \in P_1$, $e_2 \in P_2$가 존재함
      - 두 간선을 교환하면 길이가 k인 두 경로 Q_1, Q_2를 만들 수 있음
        - $Q_1 = P_{2,s} + (a, c) + P_{1,e}$
        - $Q_2 = P_{1,s} + (b, d) + P_{2,e}$



Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - D(k, n) = min{ D(k-1, i) + C(i+1, n) } 에서 C가 monge matrix 이면 f(k) = D(k, n)은 볼록함
    - 목표: 길이가 k+1, k-1인 두 최단 경로 P_1, P_2의 간선을 적절히 교환해 $L(Q_1) = L(Q_2)$인 두 경로 생성

    - u ≤ v에 대해, 0 → u 경로 P_1과 0 → v 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \geq k_2$이면 임의의 $0 \leq x \leq k_2-k_1$에 대해, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재
      - k1 = k+1, k2 = k-1이고 u = v = n일 때 x = 1인 간선 $e_1 \in P_1$, $e_2 \in P_2$가 존재함
      - 두 간선을 교환하면 길이가 k인 두 경로 Q_1, Q_2를 만들 수 있음
        - $Q_1 = P_{2,s} + (a, c) + P_{1,e}$
        - $Q_2 = P_{1,s} + (b, d) + P_{2,e}$



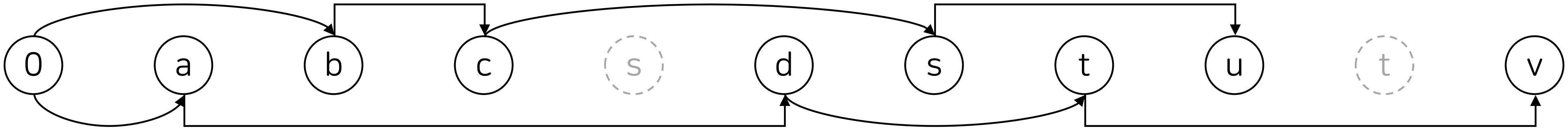
Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k) = D(k, n)$은 볼록함
    - 목표: 길이가 $k+1$, $k-1$인 두 최단 경로 P_1, P_2의 간선을 적절히 교환해 $L(Q_1) = L(Q_2)$인 두 경로 생성

    - $u \le v$에 대해, $0 \rightarrow u$ 경로 P_1과 $0 \rightarrow v$ 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \ge k_2$이면 임의의 $0 \le x \le k_2-k_1$에 대해, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재
      - 이것만 증명하면 f가 볼록하다는 것을 보일 수 있음
    - (k_1, k_2) 순서쌍에 대해 귀납법 사용
      - 기저 조건: $k_1 = k_2 = 1$, $u \le v$ 이므로 P_2의 유일한 간선이 P_1의 유일한 간선을 포함함. 따라서 $x = 0$ 가능
      - case 1. $k_2 = 1 < k_1$
      - case 2. $k_1 \ge k_2 \ge 2$이고 P_2의 마지막 간선이 P_1의 마지막 간선을 포함하지 않음
      - case 3. $k_1 \ge k_2 \ge 2$이고 P_2의 마지막 간선이 P_1의 마지막 간선을 포함함

Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $u \le v$에 대해, $0 \to u$ 경로 P_1과 $0 \to v$ 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \ge k_2$이면 임의의 $0 \le x \le k_2-k_1$에 대해, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재
      - (k_1, k_2) 순서쌍에 대해 귀납법 사용
      - case 1. $k_2 = 1 < k_1$
        - P_2의 유일한 간선이 P_1의 간선을 모두 포함하므로, e_1을 P_1의 $x+1$번째 간선으로 잡으면 됨
      - case 2. $k_1 \ge k_2 \ge 2$이고 P_2의 마지막 간선이 P_1의 마지막 간선을 포함하지 않음
        - P_1과 P_2의 마지막 간선을 각각 (s, u), (t, v)라고 하자
        - $u \le v$인데 (t, v)가 (s, u)를 포함하지 않으므로 $s < t$를 만족함
        - P_1과 P_2에서 마지막 간선을 제거하면 $s \le t$, $k_1-1 \ge k_2-1$, $k_1-k_2 = (k_1-1)-(k_2-1)$
        - 따라서 귀납 가정에 의해 e_1, e_2 를 찾을 수 있음



Aliens Trick의 충분 조건

- Aliens Trick의 충분 조건
  - Monge Matrix
    - $u \le v$에 대해, $0 \to u$ 경로 P_1과 $0 \to v$ 경로 P_2의 길이를 각각 k_1, k_2라고 하자.
    - $k_1 \ge k_2$이면 임의의 $0 \le x \le k_2 - k_1$에 대해, $L(P_{1,s}) = L(P_{2,s}) + x$ 가 되도록 하는 e_1, e_2 존재
      - (k_1, k_2) 순서쌍에 대해 귀납법 사용
      - case 3. $k_1 \ge k_2 \ge 2$이고 P_2의 마지막 간선이 P_1의 마지막 간선을 포함함
        - P_2의 마지막 간선이 P_1의 마지막 $y > 0$개의 간선을 포함한다고 하자
        - 즉, P_1의 k_1-y+1 ~ k_1번째 간선이 e_2에 포함됨
        - $k_2+x \ge k_1-y+1$이면 $e_1 = P_1$의 k_2+x번째 간선, $e_2 = P_2$의 마지막 간선으로 잡으면 됨
        - 그렇지 않은 경우, $x \ge 1$이므로 양변에서 각각 x와 1 없애면 $k_1 - y \ge k_2$
        - 따라서 P_1의 마지막 y개의 간선을 제거하면 귀납 가정에 의해 e_1, e_2 찾을 수 있음
      - 증명 끝!
      - 주의: e1, e2의 존재성을 보일 때 monge의 성질을 사용하지 않았음
    - 따라서 C가 monge matrix이면 f(k)는 볼록함

Aliens Trick의 충분 조건

- 정리
  - $f(k) = D(k, n)$이라고 하자.
  - $D(k, n)$ = 유량을 k 만큼 흘리는 최소(또는 최대) 비용으로 모델링되면 $f(k)$는 볼록함
  - $D(k, n) = \min\{ D(k-1, i) + C(i+1, n) \}$ 에서 C가 monge matrix 이면 $f(k)$는 볼록함
    - (경로 교환) 두 경로에서 포함 관계인 두 간선을 풀어줘도 두 경로의 가중치 합이 커지지 않음

    - 출발점이 같은 두 경로 P_1, P_2에 대해 P_1의 끝점 ≤ P_2의 끝점이고 $L(P_1) \geq L(P_2)$이면
    - 임의의 $0 \leq x \leq L(P_1)-L(P_2)$에 대해 $e_2 = (a, d) \in P_2$가 $e_1 = (b, c) \in P_1$을 포함하고
    - L(b에서 끝나는 P_1의 prefix) = L(a에서 끝나는 P_2의 prefix) + x가 되는 간선 e_1, e_2 존재

질문?

Aliens Trick의 활용

Aliens Trick의 활용

- BOJ 16191 Utilitarianism
  - 간선에 가중치가 있는 트리에서 크기가 k인 최대 가중치 매칭을 구하는 문제

  - 트리 = 이분 그래프
  - 이분 그래프의 최대 가중치 매칭 = MCMF
  - 따라서 Aliens Trick 사용 가능

  - $D_c(v, 0)$ = v를 루트로 하는 서브 트리에서 최대 가중치 매칭, v는 매칭 안 됨
  - $D_c(v, 1)$ = v를 루트로 하는 서브 트리에서 최대 가중치 매칭, v는 매칭됨
  - 매칭을 만들 때마다 가중치 c 추가

Aliens Trick의 활용

- BOJ 10067 수열 나누기
  - 음이 아닌 정수로 이루어진 수열을 총 k+1개의 조각으로 나눠야 함
  - k+1개의 조각을 만들기 위해서는 수열을 k번 나눠야 함
  - [s, e] 구간을 [s, m]과 [m+1, e]으로 나누면 (s~m의 합) * (m+1~e의 합) 만큼 점수 얻음
  - 얻을 수 있는 최대 점수를 구하는 문제

  - S[i] = A[1] + A[2] + … + A[i]라고 정의하자.
  - D(k, n) = max{ D(k-1, i) + S[i] * (S[n] – S[i]) }
  - D(k, n) = max{ S[i] * S[n] + D(k-1, i) – S[i] * S[i] }
    - 기울기가 S[i], 절편이 D(k-1, i) – S[i] * S[i]인 직선
    - S[i]는 단조 증가하므로 스택을 이용해 CHT 가능
    - 즉, C(i, n)은 monge
  - 따라서 Aliens Trick 사용 가능

Aliens Trick의 활용

- BOJ 10067 수열 나누기
  - $D_c(n)$ = n번째 원소까지 수열을 적당히 나눴을 때 얻을 수 있는 최대 점수, 나눌 때마다 +c
  - $D_c(n) = \max\{ D_c(i) + S[i] * (S[n] - S[i]) \}$
  - 여전히 CHT 적용 가능하므로 시간 복잡도는 O(N log X)

  - 역추적은 어떻게 하지?

질문?

Aliens Trick의 역추적

Aliens Trick의 역추적

- 기울기가 변하지 않는 구간이 있는 경우
  - $D_1(n)$을 계산하면 k는 1, 2, 3, 4 모두 가능
  - 특정 k값을 강제하는 것은 어려움
  - 대신 가장 작은 k와 가장 큰 k를 구하는 것은 가능함
    - 각각 l(c), r(c)라고 하자.

  - k가 정수일 때 f(k)가 항상 정수라고 가정하자.
    - 그래프에서 꺾이는 지점을 제외하면 항상 기울기는 정수임
    - 기울기를 반정수 범위에서 탐색하면 꺾이는 지점을 찾을 수 있음
    - 따라서 가장 작은 k와 가장 큰 k 구할 수 있음
      - $l(c) = D_{c+1/2}(n)$에서의 k
      - $r(c) = D_{c-1/2}(n)$에서의 k

Aliens Trick의 역추적

- 기울기가 변하지 않는 구간이 있는 경우
  - $D_1(n)$을 계산하면 k는 1, 2, 3, 4 모두 가능
  - 가장 작은 k인 l(c)와 가장 큰 k인 r(c)를 구하는 것은 가능함
  - l(c)와 r(c)를 구했다면 원하는 k를 찾는 것은 어렵지 않음
    - $k_1 = r(c)$, $k_2 = l(c)$라고 하자.
    - $k_1 \neq k$이고 $k_2 \neq k$인 상황, 즉 $k_2 < k < k_1$인 상황만 생각하자.
    - P_1 = 간선을 k_1개 사용하는 최단 경로, P_2 = 간선을 k_2개 사용하는 최단 경로
    - $0 \leq x = k-k_2 \leq k_2-k_1$에 대해 $e_2 = (a, d) \in P_2$가 $e_1 = (b, c) \in P_1$을 포함하고
    - $L(P_1$의 b까지의 prefix$)$ = $L(P_2$의 a까지의 prefix$)$ + x를 만족하는 간선 e_1, e_2를 구할 수 있음
    - e_1과 e_2에 대해 경로 교환 수행하면 $2f(k) \leq w(Q_1)+w(Q_2) \leq f(k_1)+f(k_2)$를 만족하는 Q_1 Q_2 나옴
    - k, k_1, k_2에서의 기울기는 모두 동일하므로 $2f(k) = f(k_1) + f(k_2)$
    - 따라서 Q_1과 Q_2는 모두 간선이 k개인 최단 경로
    - 끝!

Aliens Trick의 역추적

```cpp
// pair<T, vector<int>> f(T c): return opt_val, prv
// cost function must be multiplied by 2
template<class T, bool GET_MAX = false>
pair<T, vector<int>> AliensTrick(int n, int k, auto f, T lo, T hi){
    T l = lo, r = hi;
    while(l < r){
        T m = (l + r + (GET_MAX?1:0)) >> 1;
        vector<int> prv = f(m*2+(GET_MAX?-1:+1)).second;
        int cnt = 0; for(int i=n; i; i=prv[i]) cnt++;
        if(cnt <= k) (GET_MAX?l:r) = m;
        else (GET_MAX?r:l) = m + (GET_MAX?-1:+1);
    }
    T opt_value = f(l*2).first / 2 - k*l;

    vector<int> prv1 = f(l*2+(GET_MAX?1:-1)).second, p1{n};
    vector<int> prv2 = f(l*2-(GET_MAX?1:-1)).second, p2{n};
    for(int i=n; i; i=prv1[i]) p1.push_back(prv1[i]);
    for(int i=n; i; i=prv2[i]) p2.push_back(prv2[i]);
    reverse(p1.begin(), p1.end()); reverse(p2.begin(), p2.end());
    assert(p2.size() <= k+1 && k+1 <=p1.size());
    if(p1.size() == k+1) return {opt_value, p1};
    if(p2.size() == k+1) return {opt_value, p2};

    for(int i=1, j=1; i<p1.size(); i++){
        while(j < p2.size() && p2[j] < p1[i-1]) j++;
        if(p1[i] <= p2[j] && i - j == k+1 - (int)p2.size()){
            vector<int> res;
            res.insert(res.end(), p1.begin(), p1.begin()+i);
            res.insert(res.end(), p2.begin()+j, p2.end());
            return {opt_value, res};
        }
    }
    assert(false);
}
```

질문?