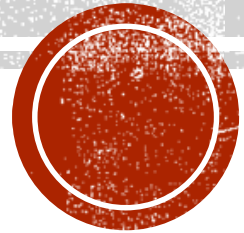
CONVEX HULL TRICK

Seo youngsun(nein) Soongsil Univ.



CONVEX HULL TRICK이란?

- 동적계획법을 해결하는 문제 중 특정 형태의 점화식 , 조건 등을 만족할 때 시간 복잡도를 줄여주는 최적화 기법이다.
- 점화식을 채워 나갈 때 점화관계에서 이전에 봐야할 위치들이 있는데, 볼 필요가 없는 부분을 없애 주는 방식이다.

CONVEX HULL TRICK이란?

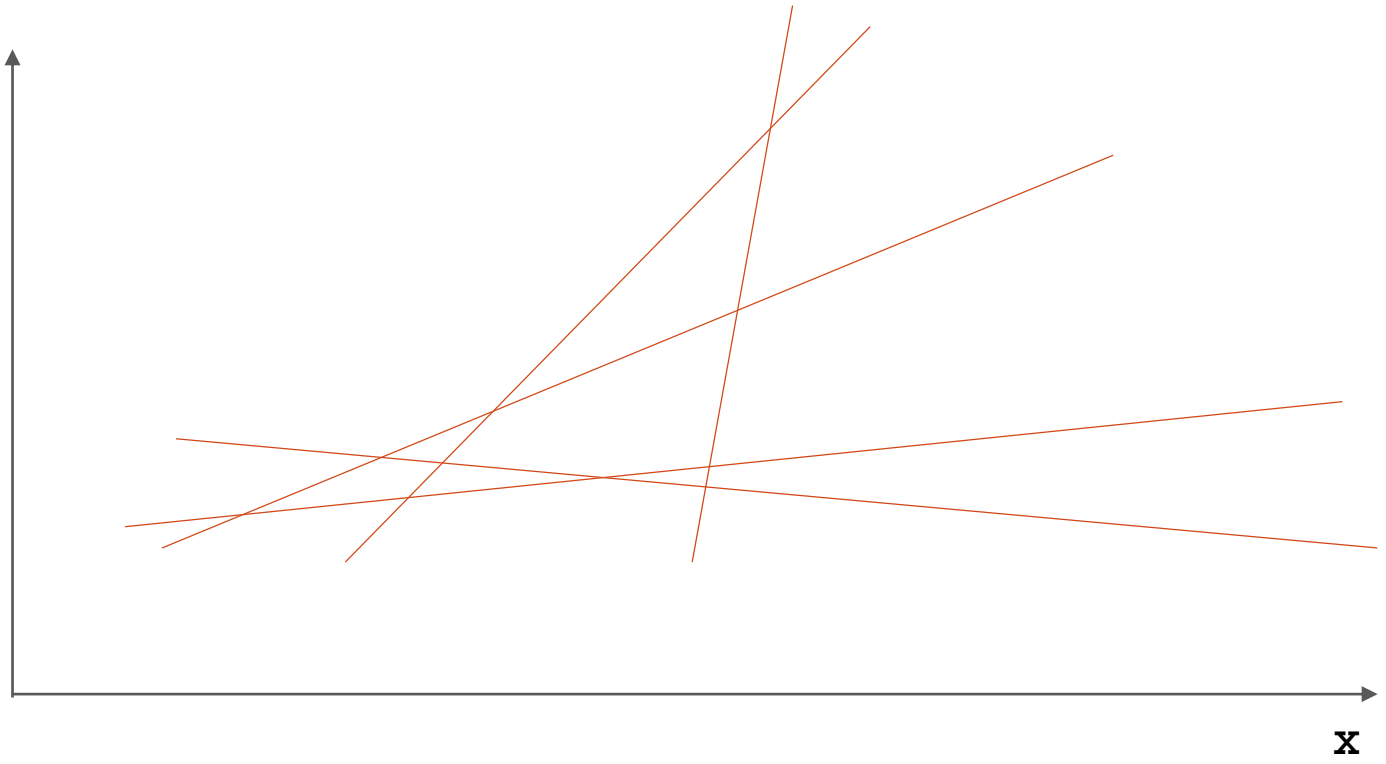
- 다음과 같은 점화식이 있다고 가정하자.
- $dy[i] = \max_{j<i}(a[i] * b[j] + c[i] + d[j])$ (단, $b[i-1] \leq b[i]$) *or*
- $dy[i] = \min_{j<i}(a[i] * b[j] + c[i] + d[j])$ (단, $b[i-1] \geq b[i]$)
- (dy[j]는 d[j]에 포함되어 있을 수 있음, 아래 설명은 **max**로함)
- dy[i]의 값을 만들기 위해서는 1,2,…,i-1까지의 b,d의 값이 필요로 하다.
- 1 : b[1]*a[i]+d[1]+c[i]
- 2 : b[2]*a[i]+d[2]+c[i]
- 3 : b[3]*a[i]+d[3]+c[i]
- …
- i-1 : b[i-1]*a[i]+d[i-1]+c[i]

CONVEX HULL TRICK이란?

- c[i]는 공통된 값이므로 무시하자.
- 특정 i의 값을 갱신할 때, 이전의 b,d의 값만 있으면 a[i]를 통하여 갱신할 수 있다.
- a[i]=x로 치환해보자.
- 1: b[1]*x+d[1]
- 2: b[2]*x+d[2]
- 3: b[3]*x+d[3]
- …
- i-1: b[i-1]*x+d[i-1]
- 이제 갱신되는 b,d를 직선으로 보자.

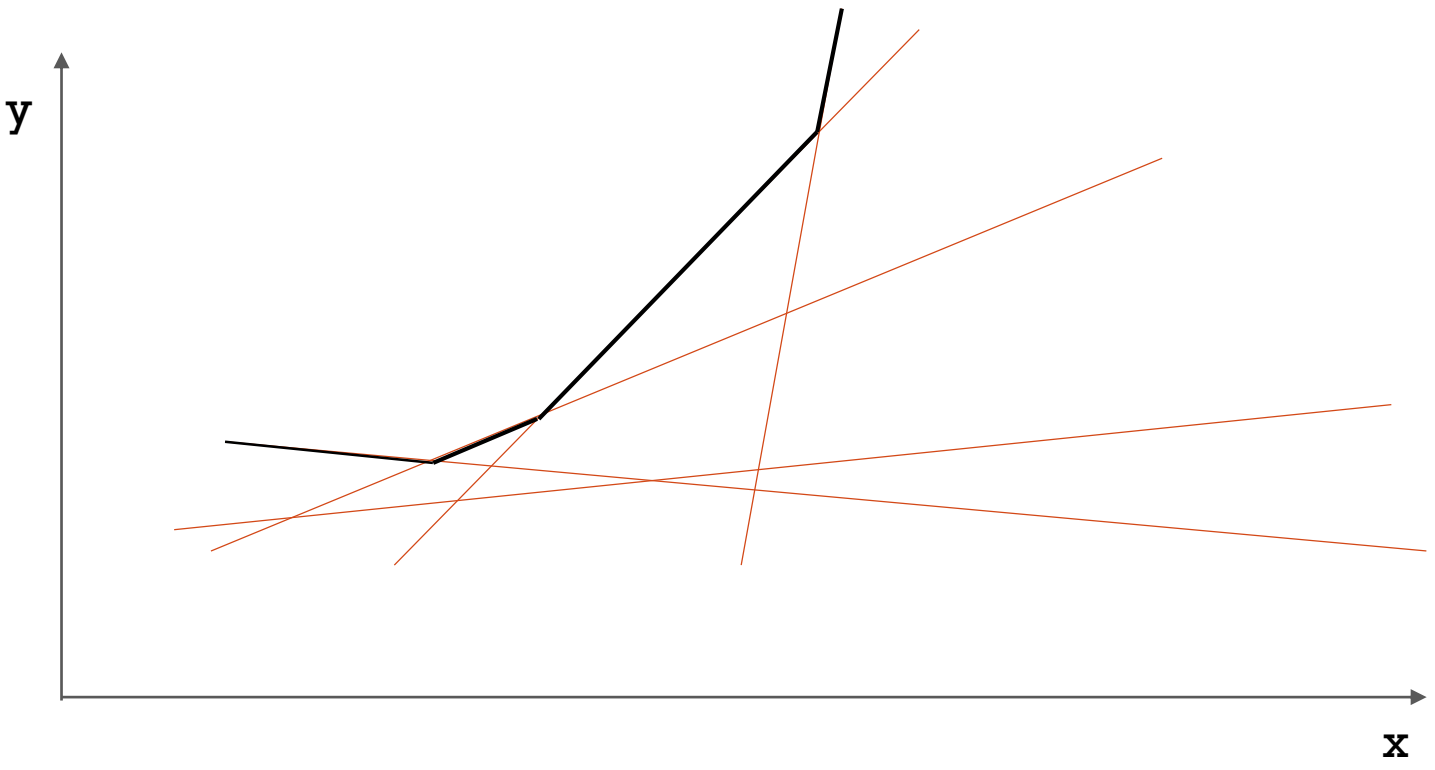
CONVEX HULL TRICK이란?

- b를 기울기, d를 y절편으로 보면 직선으로 볼 수 있다.
- 오른쪽과 같이 직선들에서 특정 x좌표에서 max를 구한다고 볼 수 있다.



CONVEX HULL TRICK이란?

- 생각해보면 **max**로 사용되는 직선들의 구간은 직관적으로 알 수 있다.
- 바로 직선들로 이루어진 볼록껍질 **(convex hull)** 부분이다.
- 따라서 볼록껍질을 이루는 직선들과 그 구간들을 알면 특정 위치의 **max**를 알 수 있다.

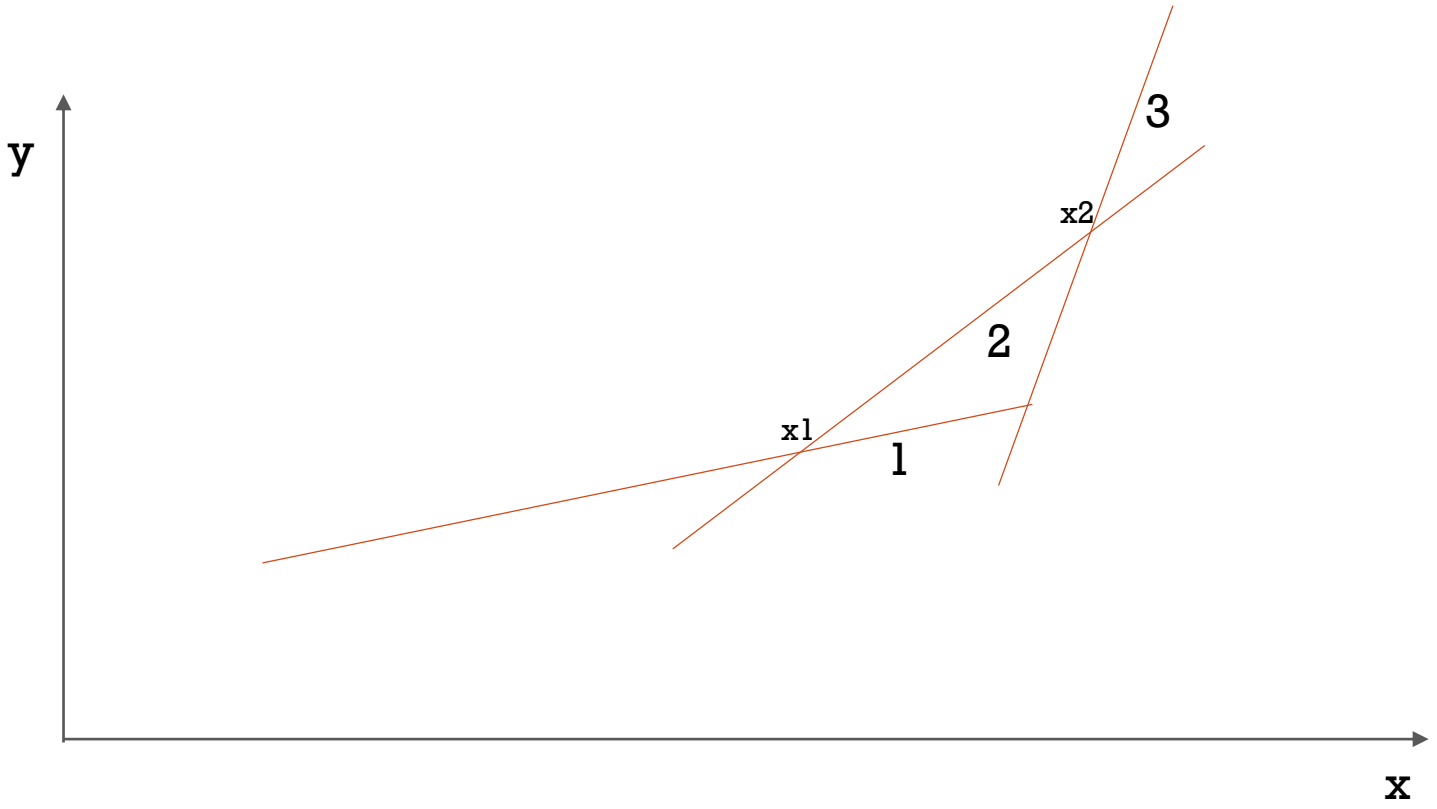


CONVEX HULL 만들기

- 결국 **convex hull trick**을 쓰기 위해서는 볼록 껍질을 만들어야 한다.
- 이를 위해서 기울기가 증가하는 형태여야 한다.
- 기울기가 증가하면 직선이 반시계 방향 순서라고 볼 수 있다.
- 따라서, 잘 알려진 **convex hull** 알고리즘인 **Graham scan**의 방식을 참고해 볼 수 있다.

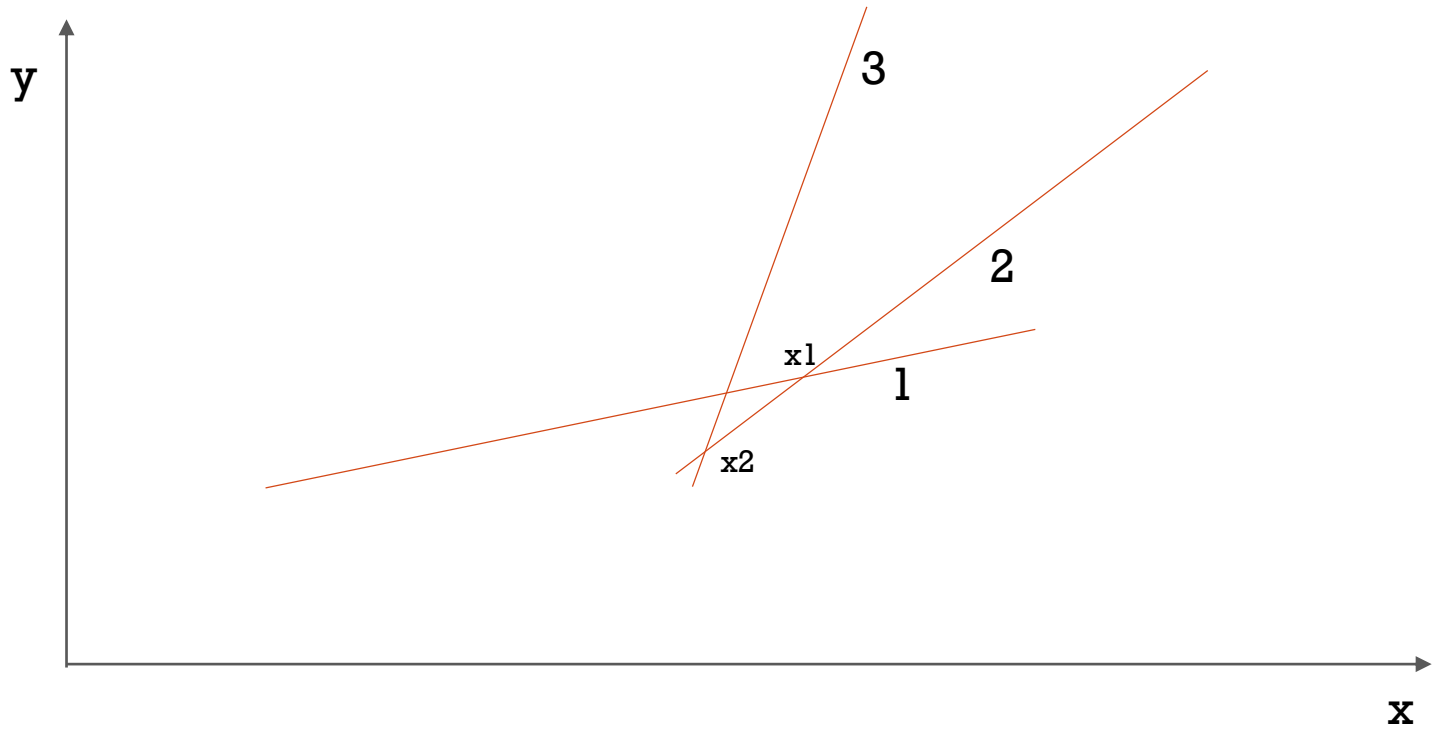
CONVEX HULL 만들기

- 다음과 같이 1, 2, 3 이 순서대로 있고, 1번직선과 2번직선의 교점 x1, 2번직선과 3번직선의 교점 x2가 있다고 하자.
- x1<x2라면 2번 직선이 최대인 구간 [x1,x2]이 존재하며 볼록 껍질에 포함된다.



CONVEX HULL 만들기

- 다음과 같은 경우는 볼록 껍질이 1번 직선에서 3번 직선으로 바로 가는 경우이다.
- x1>x2라면 2번 직선이 최대인 구간이 존재하지 않으며 볼록 껍질에 포함되지 않는다.



CONVEX HULL 만들기

- 위와 같은 조건들을 가지고 **Graham scan** 처럼 볼록 껍질을 만들어 가면 된다.

1. **stack**에 직선이 **2**개 이상이라면 상위 **2**개와 넣으려는 직선을 통하여 조건을 확인한다.
2. 만약 중간 직선이 빠져야 한다면 빼고 반복한다.
3. **stack**에 새로운 직선을 넣는다.

- 이와 같은 방법으로 각 **i**에 대하여 **convex hull**을 들고 있을 수 있다.

최대값 찾기

- 볼록 껍질을 만들었으니 이제 거기서 최대값을 찾아야 드디어 점화식을 채울 수 있다.
- 따라서 볼록 껍질에서 최대값이 되는 직선을 찾아야 하는데
- 다음 2가지 방법으로 찾을 수 있다.
- 1. 이분 검색 O(lgN)
- 2. 스위핑 O(1) (단, a[i-1]<=a[i])

최대값 찾기

- 이분 검색의 경우 볼록 껍질의 직선들을 관리할 때, 교점들의 **x**좌표들도 같이 관리해주면 된다.
- 볼록 껍질이므로 교점 또한 정렬되어 있을 것이고, **lower_bound**와 같은 함수를 통하여 쉽게 찾을 수 있다.

최대값 찾기

- 스위핑의 경우 찾는 **x**좌표가 증가하는 경우에만 사용 가능한 것으로, **x**좌표는 늘어나므로 현재 찾으려는 직선은 이전에 찾은 직선이거나 그 다음에 있을 것이다.
- 따라서, 이를 이용해 스위핑을 해서 답을 찾아낼 수 있다.