Convex Hull Algorithm

2019.05.21

17 유혜경

ARGOS

INDEX

ARGOS

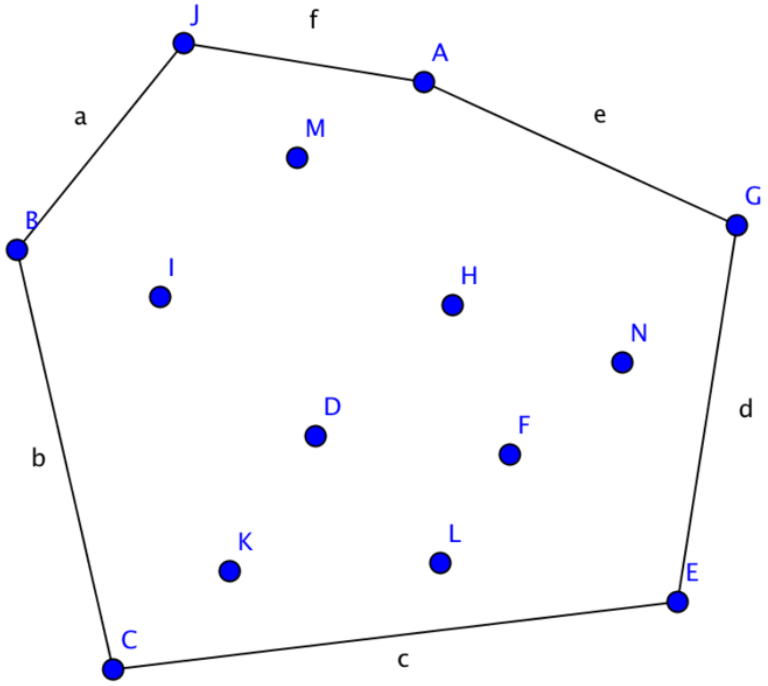
Convex hull algorithm

Convex hull algorithm

- 정의
- 종류

ARGOS

Convex Hull Algorithm이란



-볼록 껍질 알고리즘

-여러 개의 점이 주어졌을 때,

모든 점들을 포함하는 최소 크기의 볼록 다각형

Convex Hull Algorithm 종류

-Gift wrapping

-**Graham's scan**

-Quickhull

-Divide and conquer

-Monotone chain aka Andrew's algorithm

-Incremental convex hull algorithm

-The ultimate planar convex hull algorithm

-Chan's algorithm

=>이렇게 많은 알고리즘이 있는데……

이중에서도 **Graham's scan**을 제일 많이 씀

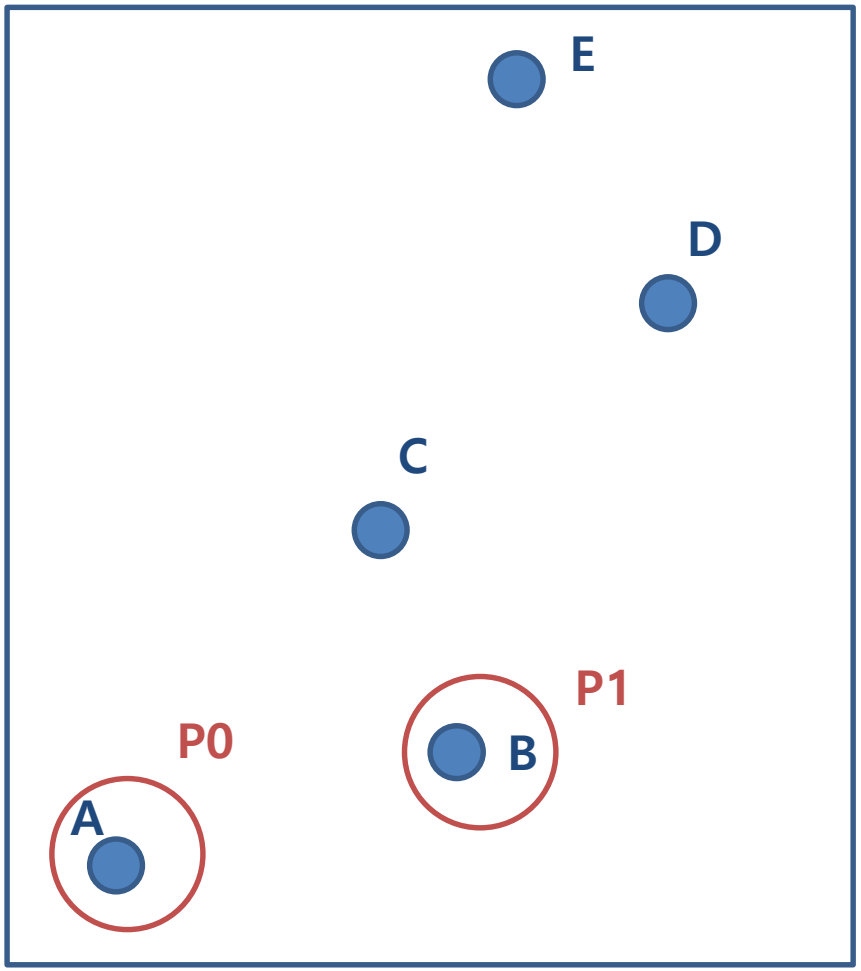
Convex hull algorithm

Graham's scan

- 동작 원리
- 시간 복잡도

ARGOS

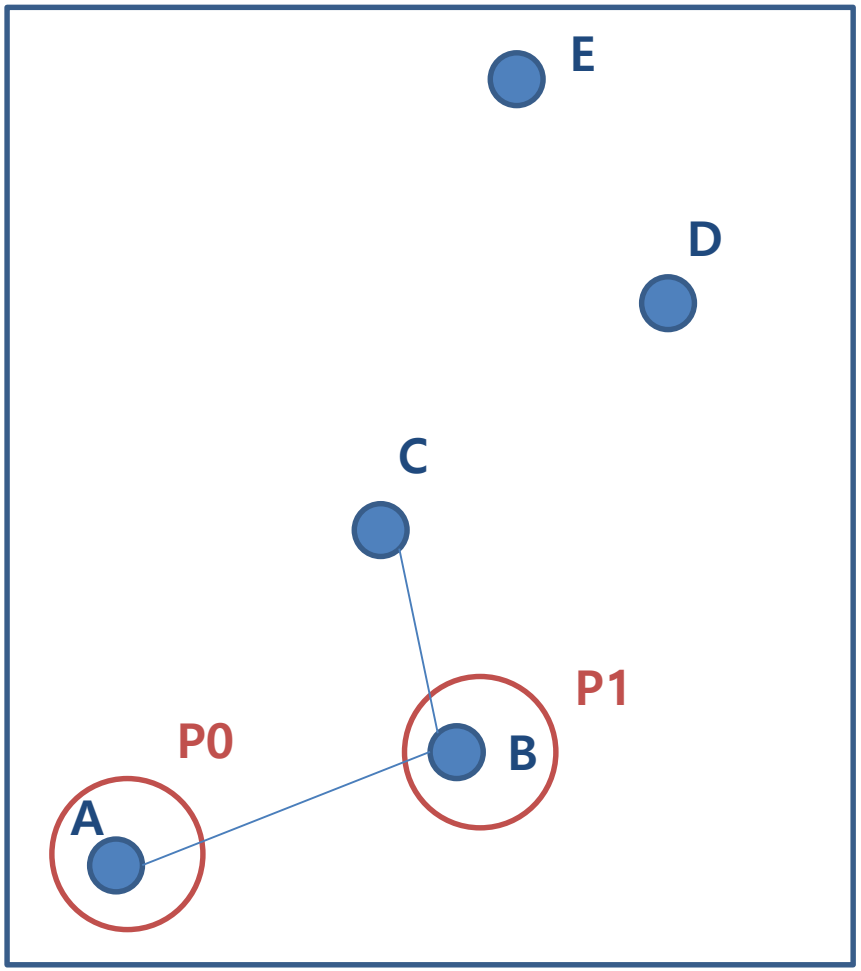
Graham's scan



1. Y좌표가 가장 작으며 x축에서 가장 왼쪽에 있는 점을 P0라고 잡는다.
2. P0와 각도가 가장 작은 점을 P1이라 잡는다.

이 두 점(P0,P1)을 stack에 push한다.

Graham's scan



stack에 두 개 이상의 원소가 있을 경우,

P[i]를 stack에 push할 수 있을 때까지

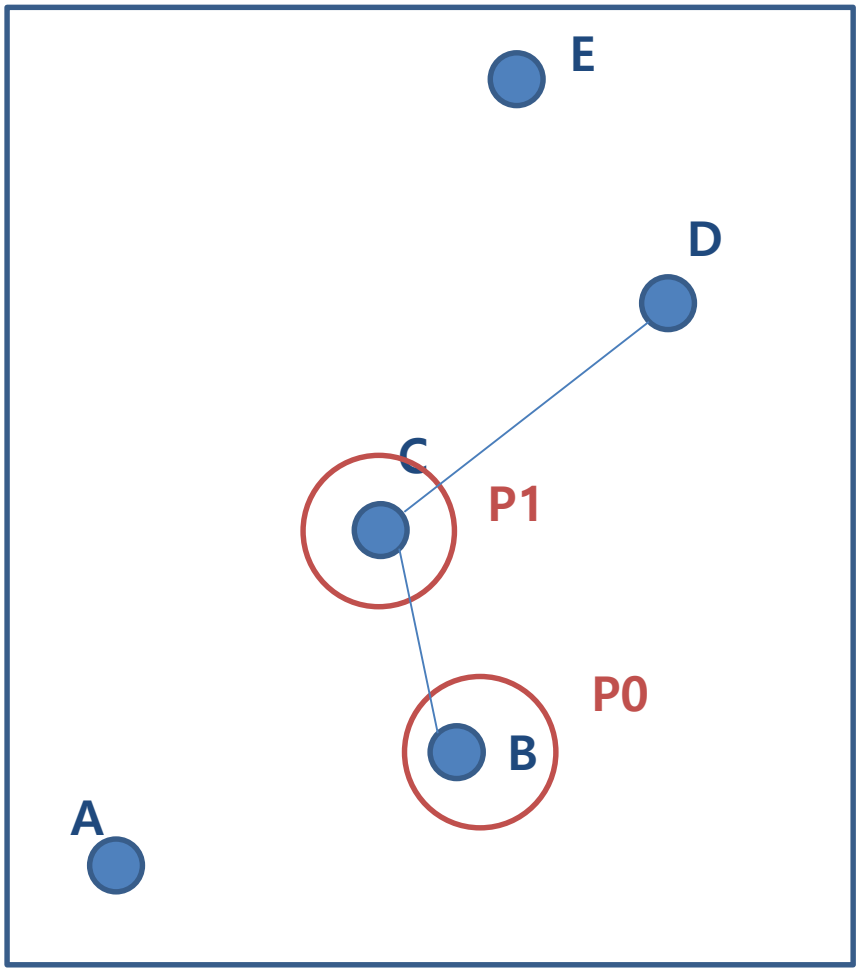
pop!!!!!!!

stack : A, B

C 고려하면 A-B-C는 반시계 방향으로 볼록해 C를 push

Stack : A, B, C

Graham's scan



B를 P0로 C를 P1으로 둔다.

stack : A, B, C

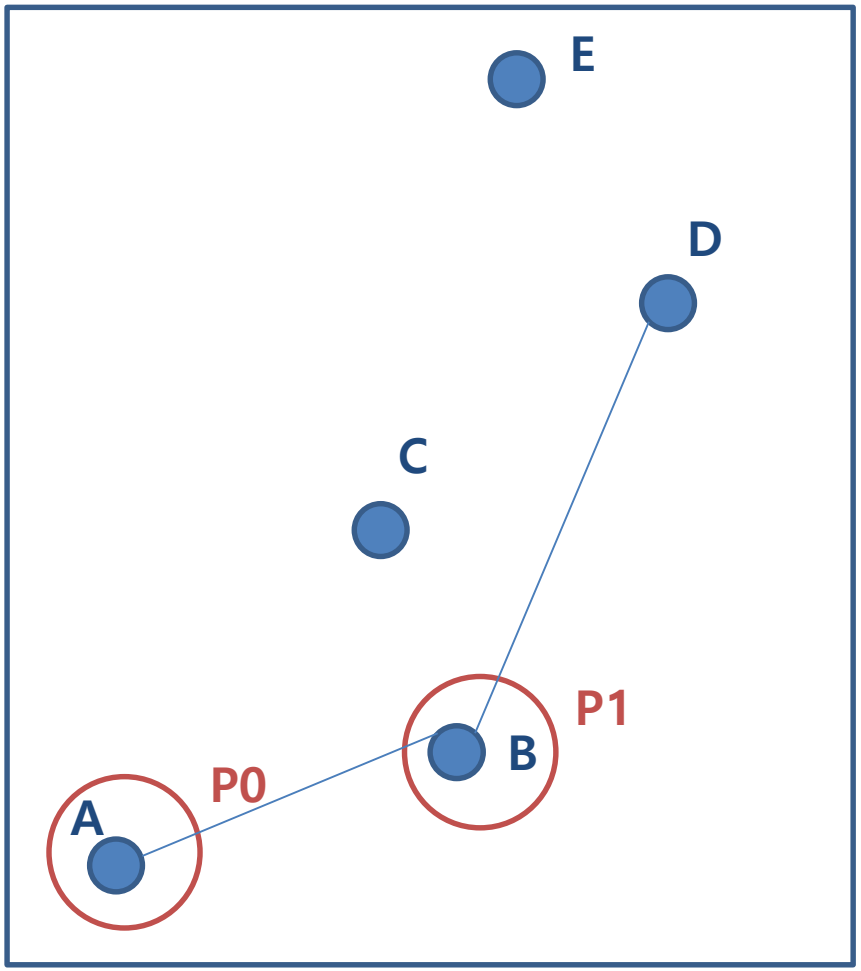
D를 고려했을 때

B-C-D는 반시계 방향으로 볼록하지 않아

C를 pop한다.

stack : A, B

Graham's scan



A를 P0로 B를 P1으로 둔다.

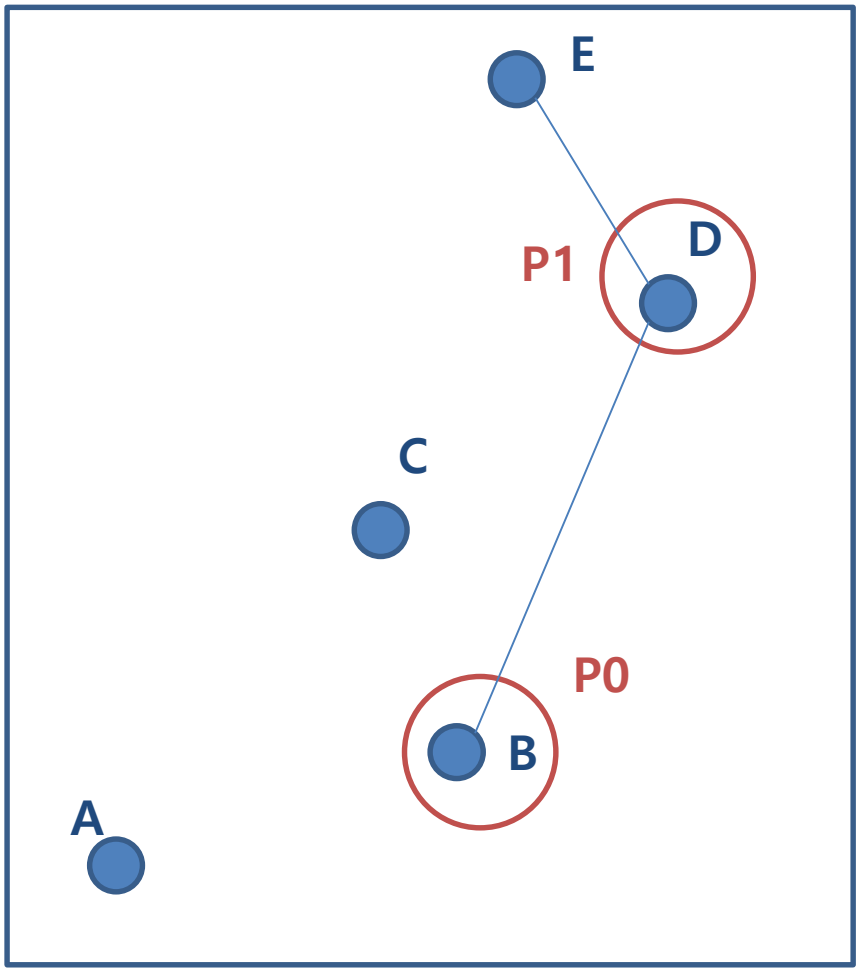
stack : A, B

D를 고려했을 때

A-B-D는 반시계 방향으로 볼록하므로 D를 push

stack : A, B, D

Graham's scan



B를 P0로 D를 P1으로 둔다.

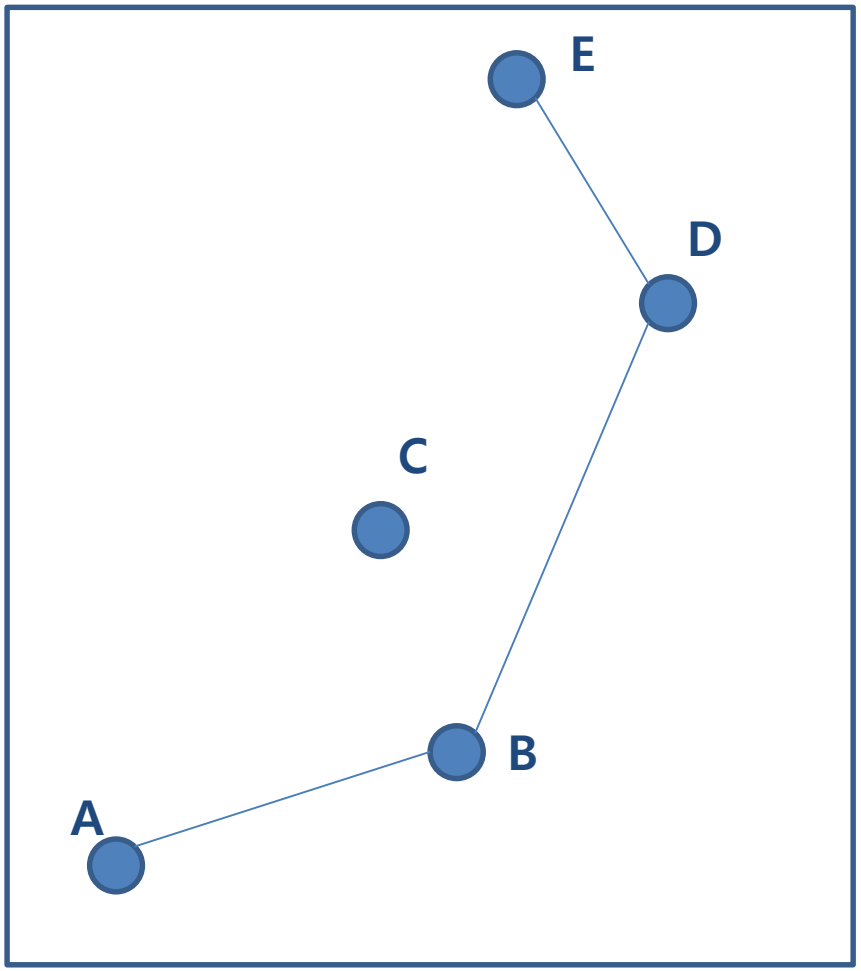
stack : A, B, D

E를 고려했을 때

B-D-E는 반시계 방향으로 볼록하므로 E를 push

stack : A, B, D, E

Graham's scan



stack : A, B, D, E

-> Convex Hull구성

Graham's scan 시간 복잡도

1. 최하단 점 찾는 작업

   -> O(n)

2. 각도 정렬

   -> 정렬 알고리즘 O(nlogn)

3. stack 구성하는 과정

   -> O(n)

Graham's scan code

ARGOS

```
class Hull{
    int x, y;
    Hull(int x, int y){
        this.x = x;
        this.y = y;
    }
}
public class temp {
    static int N;
    static Hull list[];

    public static void main(String[] args) throws Exception {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(System.out));
        StringTokenizer st;

        N = Integer.parseInt(br.readLine());
        list = new Hull[N+1];
        for(int i=1; i<=N; i++){
            st = new StringTokenizer(br.readLine());
            int a = Integer.parseInt(st.nextToken());
            int b = Integer.parseInt(st.nextToken());

            list[i] = new Hull(a, b);
        }
        // 1. 기준점 선정
        for(int i=1; i<=N; i++){
            if(list[1].y > list[i].y || list[1].y == list[i].y && list[1].x > list[i].x){
                Hull temp = list[1];
                list[1] = list[i];
                list[i] = temp;
            }
        }

        // 2. 기준점 기준으로 반시계방향으로 정렬
```

Graham's scan code

```
        // 2. 기준점 기준으로 반시계방향으로 정렬
        Arrays.sort(list, 2, N+1, new Comparator<Hull>() {

            @Override
            public int compare(Hull a, Hull b) {
                // TODO Auto-generated method stub
                int v = ccw(new Hull(list[1].x, list[1].y), a, b);
                if( v > 0)    return -1;
                if(v<0)    return 1;
                return (Math.abs(a.x) + a.y) - (Math.abs(b.x) + b.y);
            }
        });
        // 3. stack
        Stack<Integer> stack = new Stack<>();
        stack.push(1);
        for(int i=2; i<=N; i++){
            while(stack.size() > 1 && ccw(list[stack.get(stack.size()-2)], list[stack.peek()], list[i]) <=0 ){
                stack.pop();
            }
            stack.add(i);
        }
        bw.write(stack.size() + "\n");
        bw.flush();
    }
    protected static int ccw(Hull A, Hull B, Hull C) {
        long cal = 0;
        cal = (long)(B.x - A.x) * (C.y - A.y) - (long)(C.x-A.x) * (B.y-A.y);
        if(cal > 0)    return 1;
        else if (cal< 0)    return -1;
        else    return 0;
    }


}
```

문제 추천

1708번: 볼록 껍질

6850번: Cows

2254번: 감옥 건설

7420번: 맹독 방벽

3878번: 점 분리

9240번: 로버트 후드

10254번: 고속도로

Q & A

Thank You for Listening