# 점 집합을 두 개의 부채꼴로 포함하는 알고리즘 개발

## (Algorithms for Covering a Point Set with Two Wedges)

김 성 권 † 김 순 석 †† 신 찬 수 ††† 여 상 수 ††

(Sung-Kwon Kim) (Soon-Seok Kim) (Chan-Su Shin) (Sang-Soo Yeo)

**요 약** 평면상의 $n$ 개의 점으로 구성된 집합 $S$가 있을 때, 같은 각의 두 (무한) 부채꼴을 이용하여 $S$의 점들을 모두 포함하는 문제를 본 논문에서 다루고자 한다. 즉, $S \subset W_1 \cup W_2$ 를 만족하는 각이 최소인 두 부채꼴 $W_1$과 $W_2$를 구하고자 한다. 부채꼴의 정점은 반드시 $S$의 점에만 놓는다. 두 부채꼴의 배치에 따라서 여러 가지 경우로 나눌 수 있는데 각 경우에 효율적인 알고리즘을 제시한다.

***Abstract*** In this paper we investigate the problem of covering a set $S$ of $n$ points in the plane with two equi-angular (infinite) wedges. In other words, we want to find two wedges $W_1$ and $W_2$ of minimum angle such that $S \subset W_1 \cup W_2$. Apexes of wedges should be at points of $S$. Depending on how the wedges are placed we have several cases of the problem and for each of them an efficient algorithm is given.

## 1. 서 론

$A$를 평면에 있는 $n$개의 점으로 구성된 집합이라 하자. $A$에 대한 **표준 2-중심문제**는 $A$의 점을 모두 포함하면서 지름이 되도록 작은 동일한 두 원을 찾는 것이다. Sharir가 $O(n\log^9 n)$ 시간 알고리즘을 만든 후 [11], Eppstein이 $O(n\log^2 n)$ expected 시간을 가지는 알고리즘을 제시하였다 [8]. 찾고자하는 두원의 중심이 $A$에 속하는 점들에 위치하도록 제한을 하면 좀 더 어려운 문제가 되는데 이를 **이산 2-중심문제**라 부른다. 이는 Agarwal 등에 의해서 $O(n^{4/3}\log^5 n)$ 시간에 풀렸다 [7].

만약 두 원 대신에 두 정사각형으로 $A$의 점들을 모두 포함하려고 하면, 문제가 훨씬 복잡해지며, 정사각형 표준 (이산) 2-중심 문제가 된다. 두 정사각형이 서로 평행하다는 조건에서 표준 2-중심문제는 $O(n^2)$에 풀렸다 [9]. 최근에 Katz 등은 이산 2-중심문제에 대해서 연구를 했는데 [10], 두 정사각형이 서로 평행한 경우는 $O(n^2\log^4)$ 시간에, 두 정사각형이 마음대로인 경우에는 $O(n^3\log^2 n)$ 시간에 해결할 수 있음을 보였다.

본 논문에서는 앞의 문제들에 이어서 두 개의 부채꼴을 이용하여 모든 점들을 포함하는 문제를 다루고 이를 해결하는 알고리즘을 제시하고자 한다. 평면상의 한 점에서 출발하는 두 개의 반직선에 의해서 평면은 두 무한 영역으로 나누어진다. 두 영역 중에서 두 반직선이 이루는 각이 작은 쪽의 영역을 그 점과 그 두 반직선에 의해서 결정되는 **무한 부채꼴** 또는 간단히 **부채꼴(wedge)**이라 한다. 그 점은 부채꼴의 **정점 (apex)**이라 부르고, 두 반직선을 그 부채꼴의 **경계**라 부른다. 또한, 각을 부채꼴의 **각** 또는 **크기**라 부른다. 본 논문에서는 부채꼴의 각이 180도 보다 작은 것만 고려한다. 두 반직선을 구별하기 위해서 각각 **오른쪽 경계 (반직선)**, **왼쪽 경계 (반직선)**라 부른다. 편의상 부채꼴은 오른쪽 경계에서 반시계 방향으로 돌아서 왼쪽 경계에 이르는 영역을 지칭하기로 하며, 부채꼴은 경계를 포함한다. 부채꼴을 평행 이동하여, 평면의 원점 (0,0)과 정점을 일치시킨 후, 양의 $x$-축에서 반시계 방향으로 부채꼴의 오른쪽 경계에 이르는 각을 부채꼴의 **기울기**라 한다.

• 이 논문은 한국학술진흥재단 (선도연구자지원 2000-041-E00300)지원을 받았음.

† 종신회원 중앙대학교 컴퓨터공학과 교수
skkim@cau.ac.kr
†† 비 회 원 . 중앙대학교 컴퓨터공학과
sskim@alg.cse.cau.ac.kr
ssyeo@alg.cse.cau.ac.kr
††† 정 회 원 : 한국과학기술원 전산학과 BK21 연구교수
cssin@jupiter.kaist.ac.kr

논문접수 : 2001년 1월 31일
심사완료 : 2001년 5월 4일

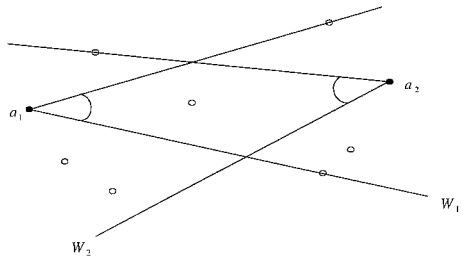


그림 1

두 부채꼴의 기울기가 같거나 기울기의 차가 180도인 경우에 두 부채꼴은 **평행**하다고 말한다. 평면상의 $n$개의 점으로 구성된 집합 $S$가 있을 때, 같은 크기의 두 부채꼴을 이용하여 S의 점들을 모두 포함하는 문제를 본 논문에서 다루고자 한다. 즉, $S \subset W_1 \cup W_2$를 만족하는 크기가 최소이면서 동일한 두 부채꼴 $W_1$과 $W_2$를 구하고자 한다. 이 경우, 두 부채꼴의 정점의 위치와 기울기, 그리고 각을 구하면 된다. 이런 부채꼴을 **최적** 부채꼴이라 부른다. 부채꼴의 정점을 어느 곳에 위치시키느냐에 따라서 두 종류의 문제를 생각할 수 있다. 먼저, 부채꼴의 정점이 반드시 $S$의 점에만 오도록 한정한 경우인데, 이를 **이산적**(discrete) 부채꼴이라 한다. 그림 1에 이산적 부채꼴의 예가 있다. 다른 하나는 부채꼴의 정점이 $S$의 볼록 껍질(convex hull) 내의 어느 곳에 위치하더라도 상관없는 경우인데, 이를 **연속적**(continuous) 부채꼴이라 한다. 연속적 부채꼴인 경우, $S$의 볼록껍질은 볼록 다각형이므로, 부채꼴의 정점이 이 다각형내의 어느 점이더라도(꼭 $S$의 점이 아니더라도) 상관없다. 본 논문에서는 이산적 부채꼴에 대한 것만 다루기로 한다. 부채꼴의 각이 180도이면 이것은 반평면에 해당하고, 두 개의 반평면으로는 어떤 $S$라도 포함할 수 있으므로, 부채꼴의 크기를 앞에서 언급했듯이 180도 미만으로 제한한다.

본 논문의 결과는 다음 표 1에 요약되어 있다. 두 부채꼴이 평행인가 아닌가에 따라 크게 두 경우로 먼저 구별한다. 두 부채꼴이 평행인 경우는 기울기가 같은가, 아니면 기울기의 차가 180도인가로 나눌 수 있다. 두 부채꼴이 평행이 아니고 마음대로인 경우는 부채꼴의 정점이 다른 부채꼴에 포함되느냐에 따라서 네 가지로 구분한다. 표에 있듯이 모두 일곱 경우로 나눌 수 있는데, 각 경우마다 사용하는 기하학적 성질과 알고리즘 기법이 다르므로 일곱 경우를 2절부터 8절까지 하나씩 설명한다.

**표기법:** 부채꼴을 그 정점을 중심으로 180도 회전하면 같은 크기의 부채꼴이 생기는데 이들은 서로 **맞꼭지 부채꼴**이 된다. 앞으로 특별한 언급이 없으면, $W_1$과 $W_2$은 두 최적 부채꼴을 나타내고, 그들의 정점은 $\alpha_1$, $\alpha_2$로 나타낸다. 부채꼴이 **겹치지 않는다**는 것은 두 부채꼴의 내부가 겹치지 않는다는 것을 의미한다. 즉, 두 부채꼴의 경계는 접해도 괜찮다.

## 2. 두 부채꼴의 기울기가 같은 경우

기울기가 같은 두 부채꼴로 $S$를 모두 포함하고자 한다. 이 경우는 두 최적 부채꼴이 같은 방향을 향하고 있으므로, 꼭 겹치게 되며, 정점 $\alpha_1, \alpha_2$가 있을 수 있는 위치가 특수하다. $S$의 볼록껍질을 $CH(S)$라 하자.

**소정리 1:** 선분 $(\alpha_1, \alpha_2)$는 $CH(S)$의 한 에지가 된다.

**증명:** $\alpha_1$과 $\alpha_2$를 잇는 직선에 의해서 평면이 두 개의 반평면으로 나뉜다. 둘 중에서 $W_1$이 없는 쪽 반평면을 $H$라 한다. 그러면 $H \cap S$는 공집합이어야 한다. 그렇지 않으면 이 점들은 $W_2$에 포함되어야 하는데, 이러면 두 부채꼴의 기울기가 같다는 가정을 위반하게 된다. $H$

표 1

| 두 부채꼴의 배치 모양 | | | 수행 시간 | 본 논문의 절 |
|---|---|---|---|---|
| 두 부채꼴이 평행한 경우 | 둘의 기울기가 같은 경우 | | $O(n^2)$ | 2절 |
| | 기울기의 차가 180도인 경우 | 둘이 겹치지 않는 경우 | $O(n^2)$ | 3절 |
| | | 둘이 겹치는 경우 | $O(n^2\log^3 n)$ | 4절 |
| 두 부채꼴의 기울기가 마음대로인 경우 | 둘이 겹치지 않는 경우 | | $O(n^2\log n)$ | 5절 |
| | 둘이 겹치는 경우 | 두 정점 모두 상대방 부채꼴 밖에 있는 경우 | $O(n^2\log n)$ | 6절 |
| | | 한 정점은 상대방 밖에 있고 다른 점은 상대방 안에 있는 경우 | $O(n^2\log^2 n)$ | 7절 |
| | | 두 정점이 모두 상대방 안에 있는 경우 | $O(n^3\log n)$ | 8절 |

$\cap S$가 공집합이므로 $\alpha_1$은 $CH(S)$의 꼭지점이여야 한다. 비슷하게, $\alpha_2$도 $CH(S)$의 꼭지점이다. 또, $H \cap S$가 공집합이므로 ( $\alpha_1, \alpha_2$)는 $CH(S)$의 에지가 된다. □

위 사실을 이용하여 알고리즘을 만든다. $CH(S)$의 모든 꼭지점은 반시계 방향으로 소트되어 있다고 가정한다. 한 에지의 양끝점을 ($b_1$, $b_2$)라 하고 부채꼴 $W_1$의 정점은 $b_1$에 $W_2$의 정점은 $b_2$에 둘 경우에 두 부채꼴의 크기를 계산해 본다. 반시계 방향으로 할 때, $b_1$ 앞에 나오는 $CH(S)$의 꼭지점을 $b_0$, $b_2$뒤에 나오는 꼭지점을 $b_3$라 하자. 먼저 $W_1$의 왼쪽 경계를 에지 ($b_0$, $b_1$)에 일치시킨다. 당연히 $W_2$의 왼쪽 경계는 $b_2$를 지나면서 ($b_0$, $b_1$)에 평행하게 정해진다. $W_2$의 오른쪽 경계는 $b_2$, $b_3$와 일치하게 그린다. 이제 $W_1$의 오른쪽 경계를 정하는데, $S$-$W_2$의 점 중에서 $W_1$의 왼쪽 경계에서 반시계방향으로 각이 가장 큰 점과 $b_1$을 지나는 직선에 의해서 정해진다. 이 직선은 $S$의 모든 점을 고려하면 되므로 $O(n)$에 가능하다. 이제 $W_1$과 $W_2$의 각 중에서 큰 것이 에지 ($b_1$, $b_2$)의 양끝점에 정점을 둘 경우의 두 부채꼴의 최적 크기가 된다. 이 작업을 $CH(S)$의 모든 에지에 대해서 수행하면 되므로, 전체 수행시간은 $O(n^2)$이 된다.

**정리 1:** 점 집합 $S$를 포함하는 기울기가 같은 두 최적 부채꼴을 $O(n^2)$ 시간에 구할 수 있다.

## 3. 두 부채꼴의 기울기가 180도 차이고 둘이 겹치지 않는 경우

$W_1$과 $W_2$는 기울기가 180도 차가 나고 둘은 겹치지 않는다. 앞 서론에서 언급했듯이 겹치지 않는다는 것은 둘의 내부가 겹치지 않는다는 의미이다. 둘의 경계는 서로 접할 수 있다.

**소정리 2:** $\alpha_1, \alpha_2$ 둘 다 $CH(S)$의 꼭지점여야 한다.

**증명:** $W_1$, $W_2$의 맞꼭지 부채꼴을 $W_1'$, $W_2'$이라 하자. $W_1$의 두 반직선을 확장하여 직선으로 만들면 이 두 직선에 의해서 평면이 네 개의 부채꼴로 나뉜다. 이들을 $W_1$에서 시작하여 반시계 방향으로 $W_1$, $W$, $W_1'$, $W'$, 이라 부르자. $W_1' \cap S$, $W_2' \cap S$가 공집합이냐 아니냐에 따라 경우를 구분한다.

(1) $W_1' \cap S$, $W_2' \cap S$ 둘 다 공집합이 아닌 경우.

(1-1) $\alpha_2 \in W_1'$인 경우.

$\alpha_1$과 $\alpha_2$를 잇는 직선이 수평이라 가정하고 $W_1$이 $W_2$의 왼쪽에 있다고 가정하자. $CH(S)$의 에지 중에서 $W_1 \cup W_2$에 완전히 포함되지 않는 에지가 두 개가 존재한다. 둘다 $CH(S) \cap W_1$과 $CH(S) \cap W_2$를 연결하는 다리(bridge)인데, 위쪽 다리가 닿는 $CH(S) \cap W_1$쪽의 꼭지점을 $p_2$라 하고 아래쪽 다리가 닿는 $CH(S) \cap W_2$쪽의 꼭지점을 $p_1$이라 하자. 그림 2-1 참조.

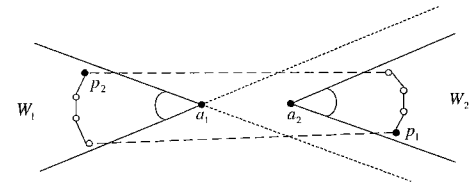


그림 2-1

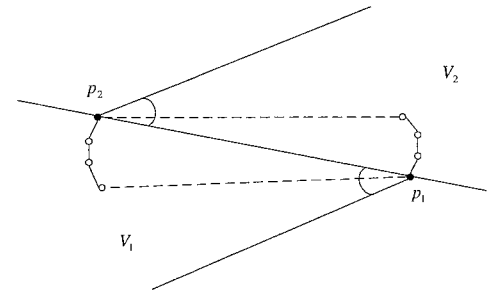


그림 2-2

이제 $W_1$을 그대로 평행 이동하여 정점을 $p_1$에 일치시켜 생긴 부채꼴을 $V_1$이라 부르고, $W_2$을 그대로 평행 이동하여 정점을 $p_2$에 일치시켜 생긴 부채꼴을 $V_2$라 부르자. $V_1 \cup V_2$ 역시 $S$ 전체를 포함한다. 그러나 $V_1$과 $V_2$는 서로 내부가 겹친다. $p_1$과 $p_2$를 잇는 직선을 $l$이라 하자. $V_1$을 그 정점을 중심으로 반시계방향으로 돌려서 오른쪽 경계가 $l$에 일치하도록 하고, $V_2$을 그 정점을 중심으로 반시계방향으로 돌려서 오른쪽 경계가 $l$에 일치하도록 한다. 그림 2-2 참조. 이제 $V_1 \cup V_2$은 $S$를 모두 포함하고, 서로 겹치지 않으며 기울기의 차가 180도이다. 또한 크기는 $W_1$과 $W_2$와 같으며, $p_1$, $p_2$ 모두 $CH(S)$의 꼭지점들이다.

(1-2) $\alpha_2 \in W'$인 경우.

$W_1$의 왼쪽 경계와 평행하게 $CH(S) \cap W_2$에 두 접선을 긋고, 둘 중 $W_1$을 교차하지 않는 접선의 접점을 $p_1$이라 한다. 비슷하게 $CH(S) \cap W_1$에 접선을 긋고 $W_2$를 교차하지 않는 접선의 접점을 $p_2$라 한다. $W_1$, $W_2$를 평행 이동하여 정점을 각각 $p_1$, $p_2$에 일치시킨 부채꼴을 $V_1$, $V_2$라 하자. 이제 $V_1$, $V_2$ 각각을 반시계방향으로 돌려서 각각의 오른쪽 경계가 $p_1$과 $p_2$를 잇는 직선 $l$에 일치하도록 한다. 그러면, $V_1$, $V_2$ 모두 우리가 원하는 조건을 만족하고 있다. 그림 3 참조.

(1-3) $\alpha_2 \in W$인 경우.

(1-2)와 대칭인 경우이므로 대칭적으로 증명 가능하다.

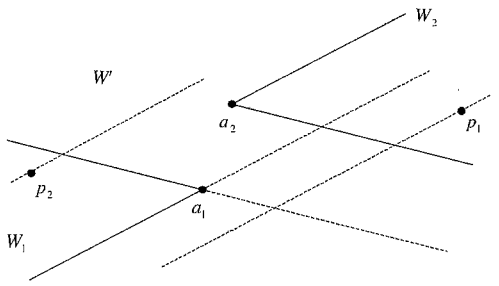


그림 3

(2) $W_1' \cap S \neq \varnothing$, $W_2' \cap S \neq \varnothing$인 경우.

이 경우는 $W_1$만 새 정점 $p_1$으로 평행 이동하면 되는데, $p_1$은 (1)의 경우처럼 찾을 수 있다. $a_2$와 $p_1$을 잇는 직선이 $l$이 된다.

(3) $W_1' \cap S \neq \varnothing$, $W_2' \cap S \neq \varnothing$ 인 경우.

이 경우는 $W_2$만 새 정점 $p_2$로 평행 이동하면 되는데, $p_2$는 (1)의 경우처럼 찾을 수 있다. $a_1$와 $p_2$를 잇는 직선이 $l$이 된다. □

$W_1$의 정점이 $CH(S)$의 한 꼭지점 $b_1$에 있는 경우를 생각하자. 정점이 $CH(S)$의 꼭지점에 있으므로 부채꼴의 한 경계를 $b_1$에 붙어 있는 에지와 일치시켜도 상관없다. $b_1$에서 반시계 방향으로 다음에 오는 꼭지점을 $b_2$라 하자. 그리고 $W_1$의 오른쪽 경계가 에지 $(b_1, b_2)$에 일치한다고 가정하자. 이 경우 $W_2$의 정점은 당연히 $(b_1, b_2)$에 평행한 직선이 $CH(S)$에 접하는 접점 $b$에 와야 하고, $W_2$의 오른쪽 경계는 $(b_1, b_2)$와 평행해야 한다. 이제 두 부채꼴의 왼쪽 경계들을 정해야 하는데, $b$와 $b_1$을 잇는 직선을 $k$라 한다. $k$를 $b_1$을 중심으로 시계방향으로 돌려서 처음 만나는 $S$의 점을 구하고 이때 $W_1$의 오른쪽 경계와 이루는 각을 $\theta_1$이라 한다. 비슷하게 $k$를 $b$를 중심으로 시계방향으로 돌려서 처음 만나는 $S$의 점을 구하고 이때 $W_2$의 오른쪽 경계와 이루는 각을 $\theta_2$라 한다. 그러면 $\max\{\theta_1, \theta_2\}$가 $W_1$의 정점을 $b_1$에 두고 그의 오른쪽 경계를 $(b_1, b_2)$에 일치시켰을 때의 부채꼴의 각이다. 그림 4 참조.

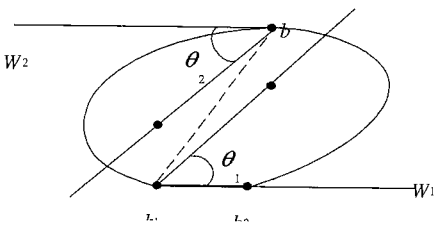


그림 4

우선, $CH(S)$의 모든 $(b_1, b_2)$에 대해서 $b$를 구해야 한다. 이는 $CH(S)$가 볼록다각형이므로 쉽게 $O(n)$에 가능하다. $\theta_1$, $\theta_2$를 구하는 것은 $(b_1, b_2)$ 에지 하나당 $O(n)$에 가능하다. 따라서 모든 에지들에 대해서 생각해야 하므로 $O(n^2)$이면 충분하다.

**정리 2:** 점 집합 $S$를 포함하는 기울기가 180도가 다르면서 겹치지 않는 두 최적 부채꼴을 $O(n^2)$ 시간에 구할 수 있다.

## 4. 두 부채꼴의 기울기가 180도 차이고 둘이 겹치는 경우

가장 복잡한 경우로 이 경우를 해결하기 위해서 알고리즘 설계 기법 중 하나인 Parametric Searching (PS) 기법 [1, 4]을 이용한다.

어떤 파라메터 $\theta$가 있을 때, 만약 $\theta \geq \theta^*$인 모든 $\theta$에 대해서 어떤 조건을 만족하고, $\theta < \theta^*$인 $\theta$에 대해서 그 조건을 만족하지 않을 경우, 그 조건을 **단조** (monotone) 조건이라 부른다. 즉, $\theta^*$를 기준으로 그보다 크거나 같은 값에 대해서는 조건이 만족되고, 그보다 작은 값에 대해서는 조건이 만족되지 않는다. PS는 최적화 알고리즘을 만드는 기법으로, 조건이 만족되는 최소값 (즉, $\theta^*$)을 찾는데 이용할 수 있는 기법이다. $\theta^*$를 찾기 위해서 PS가 쓰는 전략은 최적화 문제를 바로 해결하려 들지 않고, 이 문제의 결정 (decision) 문제에 대한 알고리즘을 개발하는 것이다. 즉, 어떤 특정 $\theta$가 주어질 때, 이 값에 대해서 조건이 만족하는지 아닌지를 결정하는 알고리즘을 먼저 개발한다. 이 말을 다시 하면, 특정 $\theta$에 대해서 $\theta \geq \theta^*$인지 아니면 $\theta < \theta^*$인지를 가려내는 알고리즘을 개발한다. 이 경우 결정 문제에 대해서 직렬 알고리즘뿐만 아니라 병렬 알고리즘도 개발한다. 그러면, 이 두 알고리즘을 잘 이용하여 시뮬레이션을 하면, 우리는 최적화 문제에 대해서 알고리즘을 만들 수 있다는 것이 PS의 아이디어이다.

결정 문제에 대한 직렬 알고리즘을 $A_s$라 하고, 병렬 알고리즘을 $A_p$라 하자. PS에서는 병렬 알고리즘 $A_p$를 $\theta = \theta^*$로 두고 직렬로 시뮬레이션한다. 그런데, $\theta^*$는 우리가 모르는 값이므로, 시뮬레이션 중간에 $\theta$를 이용한 판단(decision)을 해야 할 경우에는 직렬 알고리즘 $A_s$를 이용하여 해결한다. 따라서, $A_s$가 $O(T_s)$ 시간 걸리고, $A_p$가 $O(P)$개의 프로세서를 이용하여 $O(T_P)$ 시간 걸리면, 시뮬레이션은 총 $O(PT_P + T_sT_P \log P)$ 시간 걸린다. 즉, 최적화 알고리즘이 걸리는 시간이 이것이다. 좀 더 자세한 사항은 [1]을 참조하기 바란다. $A_p$의 구조가 어떤 특성을 만족하면 최적화 알고리즘의 시간을 약간

줄여서 $O(PT_P+T_s(T_P+\log P))$가 되게 할 수 있다 [2].

PS를 이용하여 우리 문제를 해결하기 위하여, 결정 문제를 해결하는 효율적인 직렬 알고리즘과 병렬 알고리즘을 설계해야 한다. 그러면, PS에 의해서 최적화 문제를 해결하는 직렬 알고리즘이 만들어진다. 우리 문제의 경우, 결정 문제는 $n$개의 평면상의 점으로 이뤄진 집합 $S$와 부채꼴의 각 $\theta$ 값이 입력으로 들어 올 때, 크기가 $\theta$인 두 개의 평행한 이산적 부채꼴을 이용하여, $S$의 점들을 모두 포함할 수 있느냐 없느냐를 정하는 것이다.

### 4.1 직렬 알고리즘

직렬과 병렬 알고리즘의 아이디어는 두 부채꼴 정점들 중에서 한 정점은 $S$의 특정 점 $p$로 한정하고, 다른 정점은 $S$의 어떤 점이 되도 상관없게 아무 제한도 두지 않을 경우, 과연 $S$의 점들을 모두 포함하느냐를 결정하는 알고리즘을 먼저 설계한다. 그 다음 이 알고리즘에서 $p$를 $S$의 모든 점에 대해서 반복하여 결과를 얻으면 된다. 즉, 두 정점이 자유롭게 움직일 수 있는 문제를 한 정점은 고정하고 한 정점만 자유롭게 움직이는 문제로 바꾼다.

따라서, 앞으로는 한 부채꼴의 한 정점이 점 $p$에 고정되어 있다고 가정한다. 먼저, 점 $p$를 중심으로 $S$의 점들을 모두 각의 순서대로 소트한다. 점 $p$를 정점으로 하는 크기가 $\theta$인 부채꼴을 $W_1$이라 하자. $W_1$의 기울기를 0도부터 360도까지 조금씩 증가시키면 S의 점들이 $W_1$에 들어오기도 하고 나가기도 한다. S의 점이 하나 새로 들어오거나 ($W_1$의 왼쪽 경계에 점이 위치하는 경우), 있던 점이 하나 나갈 때를 ($W_1$의 오른쪽 경계에 점이 위치하는 경우) 각각 하나의 이벤트로 생각하여 이벤트가 일어날 때마다 필요한 일을 수행한다.

지금 이벤트가 일어났다고 가정하자. 현재의 $W_1$에 포함된 $S$의 점들의 집합을 $R$이라 하자. 이제 $S$의 다른 점을 정점으로 하여 크기가 $\theta$인 부채꼴 $W_2$를 배치하여 $\overline{R}=S-R$에 속하는 점들을 모두 포함할 수 있느냐를 조사한다. $W_2$의 정점은 $R$중의 한 점이 꼭 되어야 한다. 왜냐하면, $W_1$과 $W_2$의 기울기가 180도 차로 평행하고 둘이 겹치므로, 둘의 정점은 상대방 부채꼴에 포함되어야 한다.

$R$의 점중에서 $W_2$의 정점이 될 수 있는 점이 있느냐를 조사해야 한다. 그러기 위해서, 먼저 $\overline{R}$의 볼록껍질 $CH(\overline{R})$을 구한다. $W_1$의 두 경계 반직선을 $e_1$, $e_2$라 하자. $e_1$과 평행하면서 $CH(\overline{R})$에 접하는 두 접선을 긋는다. 유사하게, $e_2$과 평행하면서 $CH(\overline{R})$에 접하는 두 접선을 긋는다. 이들 네 접선에 의해서 만들어지는 부채꼴이 여럿인데, 그 중에서 $CH(\overline{R})$을 포함하는 부채꼴이 네 개다. 그 중 둘은 $W_1$과 크기가 같고, 나머지 둘은 크기가 ($180-W_1$의 크기)이다. 크기가 $W_1$과 같은 둘 중에서 기울기가 $W_1$과 180도 차인 부채꼴을 $V$라 하고, 그것의 정점을 $a$라 하자. 만약 $a$가 R의 한 점과 일치하면, $W_2=V$가 된다. 그렇지 않으면, $V$의 맞꼭지 부채꼴을 $V$라 하자. 만약 $V$가 R의 점 중에서 하나 ($b$라하자)라도 포함하면, $V$를 그대로 평행 이동하여 정점이 $b$가 되도록 하여, $W_2=V$라 하면 된다. 그림 5 참조.

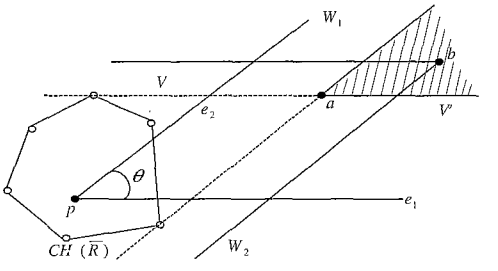


그림 5

결론적으로, $V$이 R의 점 중 하나라도 포함하면 직렬 알고리즘은 "YES"를 리턴하면 된다. 앞으로 $V$을 현 이벤트의 **후보 부채꼴**이라 부르기로 한다.

### 4.2 직렬 알고리즘의 자세한 구현

기본적으로 $CH(\overline{R})$를 매 이벤트마다 갱신하는 작업이 필요하다. 그런데, 두 이벤트 사이의 $\overline{R}$는 점 하나 차이다. 즉, 현재의 $\overline{R}$에 점이 하나 추가되거나, 하나가 삭제되거나 하여 다음 $\overline{R}$이 된다. 따라서 잘 알려진 볼록껍질 관리 알고리즘을 이용한다 [5]. 점이 하나 추가되거나 삭제됨에 따라서, 현재의 $CH(\overline{R})$에서 다음 $CH(\overline{R})$을 $O(\log^2 n)$ 시간에 구할 수 있다. 더 자세히 보면, S의 모든 점들을 p를 중심으로 각에 의해서 소트한 후 진행하므로, $\overline{R}$에 점이 추가되거나 삭제될 때도 이 순서에 따라서 진행된다. 이런 특수한 경우에는 $CH(\overline{R})$을 갱신하는데 amortized $O(\log n)$ 시간 밖에 안 걸리며 [3], 또 이 방법은 기본적인 연산 역시 $O(\log n)$에 가능하도록 지원한다. 따라서, $V$를 구하려면 $CH(\overline{R})$에 네 개의 접선을 그려야 하는데 접선 구하는 것은 한번에 $O(\log n)$ 시간에 가능하다. 따라서, S의 각 점은 $CH(\overline{R})$에 한번 추가되고 한번 삭제되며, 매 이벤트마다 네 개의 접선을 구하면 되므로, 전체 시간은 $O(n\log n)$이다.

이제 남은 일은 $V$내에 $R$의 점이 하나라도 있는가를

조사하는 것이다. 이를 위해 다음과 같은 자료구조를 사용한다. $S$에 속하는 모든 점 $t$에 대해서 $t$를 정점으로 하고 $W_1$과 같은 방향이면서 크기가 $\theta$인 부채꼴 $V_t$를 생각한다. $S$의 모든 두점 $t$, $t'$에 대해서 $t'$이 $V_t$에 포함되면 $t$를 버린다. 남은 점들만을 모아서 ($p$에 대한) 각도 순으로 나열한 리스트를 $A$라 나타내고, 이를 **외곽층**(maximal layer)라 부른다. 이 외곽층의 **기울기**는 $W_1$의 기울기와 같다. $A$의 점들은 균형 이진탐색 트리(balanced binary search tree, 줄여서 BBST)에 저장한다. $A$가 있으면, 주어진 임의의 점 $\alpha$에 (꼭 S의 점일 필요 없음)대해서 $V_\alpha$가 $R$의 점을 포함하는지를 알려면, $\alpha \in W_1$이고 $V_\alpha$가 $A$의 점을 하나라도 포함하는가를 조사하면 된다. 이 조사는 $A$가 BBST에 저장되어 있으므로 $O(\log n)$ 시간에 할 수 있다.

**소정리 3:** $\alpha \in W_1$이면, 아래 두 명제 (1)과 (2)는 동치이다.

(1) $V_\alpha$가 $R$의 한 점을 포함한다.

(2) $V_\alpha$가 $A$의 한 점을 포함한다.

**증명:** (2)가 (1)을 의미하는 것은 당연하다. (1)이 (2)를 의미하는 것을 증명하기 위해서, $V_\alpha$에 포함된 R의 한 점을 $t_1$라 하자. $t_1$이 $A$에 있으면 증명 끝. $t_1$이 $A$에 있지 않으면, 그 이유는 $V_{t_1}$이 다른 점 $t_2$를 포함하기 때문이다. 따라서 $V_\alpha$가 $t_2$를 포함한다. $t_2$가 $A$에 있으면 증명 끝. 아니면, $V_{t_2}$가 다른 점 $t_3$를 포함하기 때문이다. 그러면, $V_\alpha$가 $t_3$를 포함한다. 이런 식으로 계속 반복하면 언젠가는 $A$에 있는 점 $t_k$를 발견하게 된다. □

현재의 외곽층 $A$와 후보 부채꼴 $V$가 주어지면, 먼저 $V$의 정점이 $W_1$에 포함되는가를 조사하고, 맞으면 $V$가 $A$의 점을 하나라도 포함하는가를 조사한다 이는 $O(\log n)$ 시간에 가능하다. 그러나, 외곽층을 계산하는데 시간이 많이 소요되므로, 우리가 원하는 시간 안에 일을 마치기가 힘들다.

이를 극복하기 위하여, 다른 방법을 택하는데, 먼저 모든 후보 부채꼴들만 미리 계산한다. 지금까지 설명한 것 중에서, 외곽층 계산과 $V$가 $R$의 점을 포함하는가를 따지는 것은 전혀 하지 않고, 후보 부채꼴을 만드는 부분만을 수행한다. 후보 부채꼴 $V$에 대해서 그 정점이 현재 $W_1$에 포함되는가를 조사하여, 맞으면 집합 $CW$에 넣는다. 한 점 $p$에 대해서 최대 $n$-1 개의 후보 부채꼴이 생긴다. 이를 모든 $p$에 대해서 반복하여, 총 $O(n^2)$개의 후보 부채꼴을 만든다. $CW$를 모두 구하는 시간은 $O(n^2 \log n)$이다.

$S$의 모든 점의 쌍 $p_1$, $p_2$에 대해서 두 개의 반직선을 생각한다. 하나는 $p_1$에서 시작하여 $p_2$를 지나고, 다른 하나는 $p_2$에서 시작하여 $p_1$을 지난다. 반직선의 기울기는 시작점을 원점 (0,0)에 일치한 후, 양의 $x$-축에서 반시계 방향으로 그 반직선에 이르는 각이다. 총 $O(n^2)$개의 각이 생긴다. 이들 각각에 $\theta$를 더하면 또 $O(n^2)$개의 각이 생긴다. 지금까지 생긴 모든 각들을 소트하여 $\alpha_1$, $\alpha_2$...라 하자.

먼저, 기울기가 $\alpha_1$인 외곽층을 계산하여 $A_1$이라 하고 이를 BBST에 저장한다. 이는 $O(n \log n)$에 가능하다 [6]. $CW$의 후보 부채꼴들 중에서 기울기가 $\alpha_1$이상이고 $\alpha_2$ 미만인 것들의 집합을 $CW_1$이라 한다. $CW_1$의 각 후보 부채꼴 $V$이 $A_1$의 점을 하나라도 포함하는가를 조사한다. 하나라도 "YES"가 나오면 알고리즘은 바로 종료한다.

**소정리 4:** $CW_1$의 한 부채꼴 $V'$의 기울기를 $\alpha$라 하자. 그러면, 기울기가 $\alpha$인 외곽층과 $A_1$은 동일하다.

**증명:** 기울기가 $\alpha$인 외곽층을 $A$라 하자. 만약 $A_1$의 한 점 $s$가 $A$에는 없다고 하면, $s$를 $A$에 못나오게 하는 점 $t$가 $A$에 있어야 한다. (즉, $V_s$가 $t$를 포함한다.) 그러면 $s$에서 $t$로 가는 반직선의 기울기는 $\alpha_1$과 $\alpha_2$사이가 된다. 이는 모순이다. 다른 경우도 비슷하게 증명할 수 있다. □

다음에는 기울기가 $\alpha_2$인 외곽층 $A_2$를 만들어야 한다. 여기서 기울기가 $\alpha_2$인 반직선을 $u$에서 $v$로 가는 것이라 하면, $A_1$과 $A_2$의 관계는 $A_2 = A_1$이거나 $u$, $v$중 하나가 $A_1$에 추가되거나 삭제되거나 하는 것이다. 즉, $A_1$에 $A_2$는 기껏 한 점 차이다. $A_1$가 BBST에 저장되어 있어서 $A_1$에 추가나 삭제가 $O(\log n)$ 시간에 가능하므로 $A_1$에서 $O(\log n)$ 시간에 $A_2$로 변환할 수 있다. 이제는 $CW$의 부채꼴 중에서 기울기가 $\alpha_2$ 이상이고 $\alpha_3$ 미만인 것들의 집합을 $CW_2$이라 한다. 이런 식으로 $\alpha_3, \alpha_4, \ldots$로 하면서 $CW$의 모든 후보 부채꼴들에 대해서 그것이 외곽층의 한 점을 포함하는가를 조사 할 수 있다. 걸리는 시간은 $O(n^2 \log n)$ 이다.

**소정리 5:** 점 집합 $S$와 각 $\theta$가 주어질 때, 기울기 차가 180도이고 서로 겹치는 크기 $\theta$인 두 부채꼴로 $S$를 모두 포함할 수 있는가는 직렬로 $O(n^2 \log n)$ 시간에 판단할 수 있다.

### 4.3 병렬 알고리즘

앞절의 직렬 알고리즘을 병렬화 하려고 한다. 대부분의 스텝은 병렬화가 잘 되지만, 가장 어려운 부분이 후보 부채꼴들을 구하는 일과 후보 부채꼴들이 $R$의 점을 포함하는가를 조사하는 일이다. 둘다 아래처럼 세그멘트 트리를 이용한다.

먼저, $S$-{$p$}의 점들을 $p$를 기준으로 각으로 소트한 리스트를 $p_1$, $p_2$..., $p_{n-1}$라 하자. 리프노드가 $n$-1개인 완

전 이진트리 형태의 세그멘트 트리 $T$를 만들고 리프노드들을 차례대로 $p_1, p_2..., p_{n-1}$과 일대일 대응시킨다. $T$의 내부노드 $v$는 $v$를 루트로 하는 서브트리의 리프노드에 딸린 점들로 이루어진 CS(canonical subsequence)에 대응한다. 세그멘트 트리의 성질상, 어떤 $i \le j$에 대해서도 시퀀스 $(p_i, p_{i+1}..., p_j)$는 $O(\log n)$개의 $CS$들로 나누어진다. $T$의 각 노드는 그 CS의 점들의 볼록껍질을 기억하고, 이를 CCH (canonical CH) 라 부른다. $T$의 루트의 CCH는 $CH(S)$이다. $T$를 만드는 일은 $\theta$와 무관하므로, 병렬로 굳이 할 필요는 없지만, 병렬로는 $O(\log^2 n)$시간과 $O(n)$ 프로세서 사용하며, 직렬로는 $O(n\log^2 n)$ 시간에 가능하다.

$T$를 이용하여 후보 부채꼴을 구한다. 이를 위해서는 $CH(\overline{R})$에 접하는 네 접선을 구해야 한다. 그런데, 어떤 $i$, $j$에 대해서, $\overline{R}=\{p_i, p_{i+1}, ..., p_j\}$이다. 한 직선이 주어질 때, 이 직선에 평행한 CH$(p_i, p_{i+1}, ..., p_j)$의 두 접선은 이의 $O(\log n)$개의 CCH들 각각에 대해서 두 접선을 구한 다음, 이 접선들 중에서 가장 바깥 쪽의 두개를 택하면 된다. 시간은 하나의 프로세서를 가지고 $O(\log^2 n)$ 시간이면 된다.

따라서, 하나의 후보 부채꼴 $V$은 하나의 프로세서를 이용하여 $O(\log^2 n)$ 시간에 구한다.

역시 $T$를 이용하여 후보 부채꼴이 $R$의 점을 포함하는가를 병렬로 진행한다. 후보 부채꼴을 정점이 $\alpha$인 $V_\alpha$라 하자. $p$에서 시작하여 $\alpha$를 지나는 반직선이 어떤 $i$에 대해서 두 점 $p_{i-1}$과 $p_i$ 사이를 지나고, 반대방향의 반직선이 어떤 $j$에 대해서 두점 $p_{i-1}$와 $p_j$사이를 지난다고 가정하자. 이 두 개의 반직선에 의해서 $S$가 두 집합으로 나뉜다. $V_\alpha$의 오른쪽 경계 반직선을 $e_1$, 왼쪽 경계 반직선을 $e_2$라 한다. $\{p_i, p_{i+1}, ..., p_{j-1}\}$의 점들 중에서 $V_\alpha$에 포함되는 것이 있는가를 알기 위해서, $\{p_i, p_{i+1}, ..., p_{j-1}\}$의 $O(\log n)$개의 CCH를 하나씩 생각하면서, $e_2$가 CCH를 교차하는가를 조사한다. 하나의 CCH가 $e_2$를 교차하면 그 CCH에 있는 점 중 적어도 하나가 $V_\alpha$에 속한다는 말이 된다. 모든 CCH들이 $e_2$를 교차하지 않으면 $\{p_i, p_{i+1}, ..., p_{j-1}\}$ 점 중에서 어느 점도 $V_\alpha$에 포함되지 않는다는 말이 된다. 비슷하게, $\{p_j, p_{j+1}, ..., p_{i-1}\}$의 점 중에서 $V_\alpha$에 속하는 것이 있는가도 조사할 수 있다. 따라서 $V_\alpha$에 $S$의 점 중 하나라도 속하는지는 하나의 프로세서를 사용하여 $O(\log^2 n)$ 시간에 할 수 있다. 전체적으로 $O(n^2)$개의 후보 부채꼴이 있으므로, 다음은 증명되었다.

**소정리 6:** 점 집합 $S$와 각 $\theta$가 주어질 때, 기울기 차가 180도이고 서로 접치는 크기 $\theta$인 두 부채꼴로 $S$를 모두 포함할 수 있는가는 병렬로 $O(\log^2 n)$ 시간과 $O(n^2)$개의 프로세서로 판단할 수 있다.

**정리 3:** 점 집합 $S$를 포함하는 기울기 차가 180도이고 서로 접치는 가장 작은 두 부채꼴을 $O(n^2\log^3 n)$시간에 구할 수 있다.

**증명:** PS에 의해서 $O(n^2\log^4 n)$ 시간 알고리즘은 쉽게 나오고, Cole기법을 [2] 적용하기 위해서 병렬 알고리즘을 살펴보면, 먼저 세그멘트 트리 $T$를 만드는 것은 $\theta$와 무관하므로, 병렬로 할 필요가 없다. 볼록껍질에 접선을 구하는 일과 볼록껍질과 직선의 교차여부를 조사하는 일은 둘 다 이진탐색의 변형이므로 Cole의 기법에 맞다. 따라서, log항을 하나 줄여서 원하는 시간을 얻을 수 있다. □

## 5. 두 부채꼴의 기울기가 마음대로이면서 둘이 겹치지 않는 경우

**소정리 7:** $\alpha_1, \alpha_2$ 중 적어도 하나는 $CH(S)$의 꼭지점이다.

**증명:** $W_1$, $W_2$의 맞꼭지 부채꼴을 $W_1'$, $W'_2$이라 하자.

(1) $W'_1 \cap W_2 = \varnothing$이면, $W_1 \cup W'_1$을 제외하면 평면이 두 영역으로 나뉘는데 $W_2$는 이 두 영역중 한 부분에만 있을 수 있다. 따라서, 당연히 $\alpha_1$은 $CH(S)$의 꼭지점이다. $W'_2 \cap W_1 = \varnothing$

(2) $W'_2 \cap W_1 = \varnothing$인 경우도 위와 비슷하게 증명 가능하다.

(3) $W_1' \cap W_2 \neq \varnothing$, $W_2' \cap W_1 \neq \varnothing$인 경우.

$W_1$의 두 반직선을 확장하여 직선으로 만들면 이 두 직선에 의해서 평면이 네 개의 부채꼴로 나뉜다. 이들을 $W_1$에서 시작하여 반시계 방향으로 $W_1$, $W$, $W'_1$, $W'$라 부르자. $W_1' \cap S = \varnothing$이면 $\alpha_1$은 바로 $CH(S)$의 꼭지점이 된다. 비슷하게 $W_2' \cap S = \varnothing$이면 $\alpha_2$가 $CH(S)$의 꼭지점이다. 따라서, $W_1' \cap S$, $W_2' \cap S$ 둘 다 공집합이 아닌 경우만 증명하면 된다. 이 부분의 증명은 3절의 소정리 2와 동일하다. 단지, $W_1$과 $W_2$가 평행하냐 아니냐의 차이가 있을 뿐이다. □

$W_1$의 정점이 $CH(S)$의 한 꼭지점 $p$에 있다고 가정하고 $S$의 점들을 반시계방향으로 각도로 소트하여 $p_1$, $p_2$, ..., $p_{n-1}$이라 한다. $W_1$의 오른쪽 경계가 $p$에 붙어 있는 에지와 일치한다고 가정해도 무방하다. $W_1$의 왼쪽 경계가 $p_i$를 지난다고 할 때, $W_1$은 $p_1$, $p_2$, ..., $p_i$를 포함하고, $W_2$는 나머지 점들 $S_i=\{p_{i+1}, ..., p_{n-1}\}$ 모두를 포함해야 한다. 물론, $W_2$의 정점은 $CH(S_i)$의 꼭지점에 놓이게 되고, $W_2$의 두 경계는 정점에 인접한 $CH(S_i)$의 두 에지에 의해서 결정된다. 물론, $W_2$가 $W_1$을 겹치면 안되므로

$W_2$의 정점이 될 수 있는 $CH(S_i)$의 꼭지점들이 한정된다.

$p$를 지나면서 $CH(S_i)$에 접하는 두 접선 중에서 $CH(S_i)$와 $W_1$ 사이를 지나는 접선의 접점을 $\alpha_i$라 한다.

정의는 이렇게 했지만 실상은 $\alpha_i=p_{i+1}$이다. $W_1$의 왼쪽 경계와 평행하면서 $CH(S_i)$에 접하는 두 접선 중에서 $CH(S_i)$와 $W_1$ 사이를 지나는 접선의 접점을 $b_i$라 한다. 그러면, $CH(S_i)$의 꼭지점들 중에서 $b_i$부터 $p_{i+1}$까지 반시계 방향으로 도는 체인 $C_i$에 있는 것들만 $W_2$의 정점이 될 수 있다. 이들 중에서 내각이 가장 작은 꼭지점에 $W_2$를 배치하면 된다. 그림 6을 참조하시오.

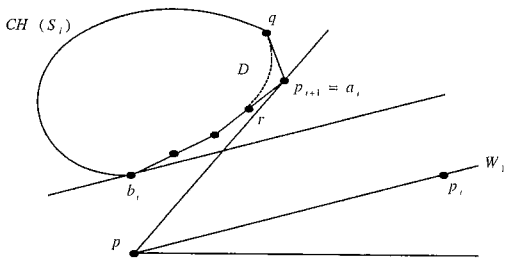


그림 6

**소정리 8:** $\sum_{i=1}^{n-1}|C_i|=O(n)$이다.

**증명:** 현재 $i$에 대해서 $p_{i+1}$와 $b_i$를 알고 있다고 하자. $CH(S_i)$에서 $p_{i+1}$에 이웃한 두 꼭지점 중 $p_{i+1}$에서 반시계방향으로 갈 때 만나는 것을 $q$라 하고, 나머지를 $r$이라 하자. 그러면, $p_{i+1}$이 $CH(S_i)$에서 빠지고 대신 $CH(S_{i+1})$에는 $r$에서 반시계방향으로 $q$에는 이르는 체인에 ($CH(S_i)$에는 없는) 새로운 꼭지점들이 생긴다. 이 꼭지점들의 집합을 $D$라 하자. 그러면, $p_{i+2}$와 $b_{i+1}$가 될 수 있는 점들은 $D\cup\{q\}\cup\{r\}$이다. $CH(S_i)$가 볼록다각형이므로 $q$, $r$ 이외의 다른 $CH(S_i)$의 꼭지점들은 $p_{i+2}$가 될 수 없다. $p_{i+2}$는 $p$를 지나는 직선이 $CH(S_{i+1})$에 접하는 접점이다. $b_i$와 $r$ 사이의 $CH(S_i)$의 꼭지점들 역시 $b_{i+1}$이 될 수 없다. 만약 되면, $CH(S_i)$의 볼록 성질에 위배된다. $b_{i+1}$은 $p$와 $p_{i+1}$을 잇는 직선과 평행한 직선이 $CH(S_{i+1})$에 접하는 접점이다. 따라서 $C_i\cap C_{i+1}$에 있을 수 있는 점은 $r$뿐이다. 모든 $1\le i\le n-1$에 대해서 $C_i\subset S$이므로 $\sum|C_i|<2(n-1)$이다. □

$CH(S_i)$에서 $CH(S_{i+1})$를 구하는 것은 점 $p_{i+1}$을 삭제하는 것이다. 삭제하는 순서가 각도 순서대로 진행되므로, [3]의 볼록껍질 관리 알고리즘을 이용하면 한번 $O(\log n)$ 시간에 가능하고, 전체로는 $O(n\log n)$시간에 가능하다. $b_{i+1}$을 구하려면 $b_i$에서 반시계방향으로 $CH(S_{i+1})$을 따라 가면서 하면 되므로, 모든 $b_i$를 구하는 것은 $O(n)$ 시간에 가능하다. 매 $i$마다 $C_i$에 있는 꼭지점 중에서 최소 내각을 이루는 것을 찾아야 한다. 앞의 소정리에 의해서 $C_i$의 크기를 모든 $i$에 대해 더하더라도 $O(n)$에 불과하므로 최소 내각을 구하는 것은 $O(n)$에 가능하다.

따라서, $W_1$의 정점을 $CH(S)$의 특정 꼭지점 $p$에 두는 경우 $S$를 모두 포함하는 두 최적 부채꼴을 $O(n\log n)$ 시간에 구할 수 있다. 이를 모든 $p$에 대해서 수행해야 하므로 다음을 얻을 수 있다.

**정리 4:** 점 집합 $S$를 포함하는 서로 겹치지 않는 두 최적 부채꼴을 $O(n^2\log n)$ 시간에 구할 수 있다.

## 6. 두 부채꼴의 기울기가 마음대로이고 두 정점 모두 상대방 부채꼴 밖에 있으면서 둘이 겹치는 경우

이 경우에는 소정리 1과 유사한 소정리를 증명할 수 있다.

**소정리 9:** 선분 $(\alpha_1,\alpha_2)$는 $CH(S)$의 한 에지가 된다.

**증명:** $\alpha_1$과 $\alpha_2$를 잇는 직선에 의해서 평면이 두 개의 반평면으로 나뉜다. 둘 중에서 $W_1$이 없는 쪽 반평면을 $H$라 한다. 그러면 $H\cap S$는 공집합이어야 한다. 그렇지 않으면 이 점들은 $W_2$에 포함되어야 하는데, 이러면 두 정점이 상대방 부채꼴 밖에 있고 두 부채꼴은 겹친다는 가정을 위반하게 된다. $H\cap S$가 공집합이므로 $\alpha_1$은 $CH(S)$의 꼭지점이여야 한다. 비슷하게, $\alpha_2$도 $CH(S)$의 꼭지점이다. 또, $H\cap S$가 공집합이므로 $(\alpha_1,\alpha_2)$는 $CH(S)$의 에지가 된다. □

이제는 $CH(S)$의 한 에지 $(a,b)$에 대해서 $a$와 $b$에 정점을 둔 두 부채꼴로 $S$의 모든 점을 포함하는 문제를 풀고, 이를 모든 에지에 적용하면 된다. 그런데, 이 문제는 폭이 같은 수직 띠와 수평 띠를 이용하여 $S$의 점들을 포함하는 문제와 동일하다. 수직 띠는 두 수직선 사이의 영역인데 이것은 정점을 $y$-좌표가 $\infty$인 점에 두는 부채꼴로 생각할 수 있다. 이 경우의 부채꼴의 각은 수직 띠의 폭과 같은 개념이 된다. 비슷하게 수평 띠는 두 수평선 사이의 영역으로서 정점을 $x$-좌표가 $\infty$인 점에 둔 부채꼴과 같다. $S$의 점들을 $a$를 중심으로 해서 잰 각과 $b$를 중심으로 해서 잰 각으로 나타낸다. $S$의 각 점에 대해서 두각이 생기는데 하나는 $x$-좌표로 다른 하나는 $y$-좌표로 한다. 그러면 두 부채꼴을 구하는 것과 두 띠를 찾는 것은 동일한 문제가 된다.

$S$의 모든 점을 $x$-좌표에 의해 소트하여 $p_1$, $p_2$, ...,

$p_n$이라 하고, 이점을 리프노드로 갖는 세그멘트 트리를 만든다. 각 내부노드는 자신을 루트로 하는 서브 트리의 리프노드에 딸린 점들의 $y$-좌표들 중에서 최소와 최대를 계산하여 가지고 있다. 이 세그멘트 트리는 다음과 같이 이용한다. 만약, $\{p_i, p_{i+1}, ..., p_j\}$을 수직 띠로 포함하고 있다면, $S-\{p_i, p_{i+1}, ..., p_j\}$를 포함하는 수평 띠의 폭 (높이)은 세그멘트 트리를 이용하여 $O(\log n)$ 시간에 계산할 수 있다.

행렬 $A(i,j)$를 $\{p_i, p_{i+1}, ..., p_j\}$를 포함하는 수직 띠의 폭이라 하고, $B(i,j)$를 나머지 점들을 포함하는 수평 띠의 폭이라 하자. 행렬 $M(i,j)=\max\{A(i,j), B(i,j)\}$, $i\le j$으로 정의하면, 우리가 찾고자 하는 것은 이 행렬의 최소값이다. 행렬의 최소값을 찾기 위해 각 열의 최소값의 위치를 살펴보면, 열 최소(row minima)들은 행렬의 왼쪽 위에서 오른쪽 아래로 내려가는 계단모양으로 나타남을 쉽게 알 수 있다. 또한, 고정된 $i$에 대해서 $M(i,j)$값은 $j$가 증가함에 따라서 감소하다가 증가하는 bitonic 성질을 가지고 있어 $i$번째 열의 최소를 쉽게 찾을 수 있다. 이 행렬에서 오른쪽으로 한 칸 이동하거나 아래로 한 칸 이동할 때마다 $A(i,j)$와 $B(i,j)$를 수정하는 일은 세그멘트 트리를 이용하여 항상 $O(\log n)$ 시간에 가능하다. 그러므로 행렬 전체의 최소값을 찾는 것은 길이가 $O(n)$ 계단을 쭉 따라 내려가면 되므로 $O(n\log n)$ 시간에 가능하다.

이 일을 $CH(S)$의 모든 에지들에 대해서 해야하므로, 전체 시간은 $O(n^2\log n)$이다.

**정리 5:** 점 집합 $S$를 포함하는 정점이 상대방 부채꼴 밖에 있고 서로 겹치는 두 최적 부채꼴을 $O(n^2\log n)$ 시간에 구할 수 있다.

## 7. 두 부채꼴의 기울기가 마음대로이면서 한 정점만 상대방 부채꼴 안에 있는 경우

**소정리 10:** 상대방 부채꼴에 들어가지 않는 정점은 $CH(S)$의 꼭지점이 되어야 한다.

**증명:** 그림 7처럼 $a_2\in W_1$이고 $a_2\notin W_1$인 경우를 살펴보자. 만약 $a_1$이 $CH(S)$의 내부점이면 $a_1$을 포함하는 $CH(S)$ 경계상의 세 점으로 구성된 삼각형 $\Delta bcd$가 있어야 한다. 세점 $b, c, d$는 $W_1\cup W_1$에 속해야 하는데, 그림에서 보듯이 어떻게 $b, c, d$를 놓더라도 $a_1$을 포함할 수 없다. 따라서 $a_1$은 $CH(S)$의 꼭지점이다. □

또한, $W_1$의 한쪽 반직선(그림의 경우 오른쪽 반직선)은 $a_1$이 붙어있는 $CH(S)$의 에지와 일치해야 한다는 것도 알 수 있다. $CH(S)$의 한 꼭지점 $p$에 $W_1$의 정점이 온다고 가정하자. $p$에 대해 $S$의 점들을 $W_1$의 오른쪽 반직선을 기준으로 각으로 소트하여 $p_1, p_2, ..., p_{n-1}$라 하고, 해당되는 각을 $\alpha_1, \alpha_2, ..., \alpha_{n-1}$이라 하자. ($\alpha_1\le\alpha_2\le...\le\alpha_{n-1}$) 만약, $W_1$의 왼쪽 반직선이 $p_i$에 의해 결정되면, $W_2$의 정점은 $\{p_1, p_2, ..., p_i\}$중 하나이고 그것의 경계는 $CH\{p_{i+1}, ..., p_{n-1}\}$에 대한 두 접선으로 구성된다. $\{p_1, p_2, ..., p_i\}$내의 각 점에서 $CH(p_{i+1}, ..., p_{n-1})$에 두 접선을 그어 생긴 각들 중에서 최소값을 $\beta_i$라 하자. 그러면 $\beta_1\ge\beta_2\ge...\ge\beta_{n-1}$이다.

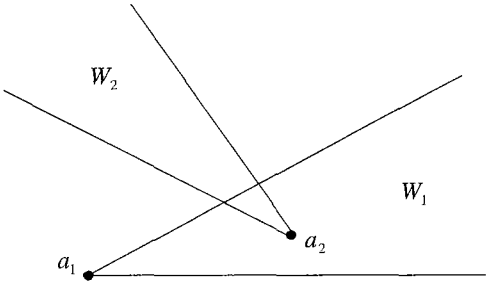


그림 7

이제 배열 $A(i)=\max\{\alpha_i, \beta_i\}$, $1\le i\le n-1$에서 최소를 찾으면 된다. $\alpha_i$들은 증가하고 $\beta_i$들은 감소하므로 $A(i)$는 감소 후 증가하는 bitonic 시퀀스이다. 따라서 이진탐색으로 최소값을 찾을 수 있다. 그런데, 특정 $i$에 대해서 $\beta_i$를 계산하는 시간이 $O(n\log n)$이므로, 최소 $A(i)$는 $O(n\log^2 n)$ 시간에 찾을 수 있다. 이를 $CH(S)$의 모든 꼭지점에 대해서 수행해야 하므로 $O(n^2\log^2 n)$ 시간이 걸린다.

**정리 6:** 점 집합 $S$를 포함하면서 한 정점만 상대방 부채꼴 안에 있는 두 최적 부채꼴을 $O(n^2\log^2 n)$ 시간에 구할 수 있다.

## 8. 두 부채꼴의 기울기가 마음대로이면서 두 정점 모두 상대방 부채꼴 안에 있는 경우

4절과 비슷하게 진행된다. 먼저 결정 알고리즘을 생각하는데 주어진 부채꼴 크기를 $\theta$라 한다. $S$의 한 점 $p$에 크기가 $\theta$인 부채꼴을 하나 놓는다고 가정할 때, $S$의 다른 점들 중에서 한 점을 골라, 점 $p$와 그 점에 두 개의 부채꼴을 배치하여 $S$ 전체를 포함하는 것이 가능한가를 먼저 조사한다. 그리고 이 작업을 모든 $p$에 대해서 반복한다.

따라서, 앞으로는 한 부채꼴의 한 정점이 점 $p$에 고정되어 있다고 가정한다. 먼저, 점 $p$를 중심으로 $S$의 점들을 모두 각의 순서대로 소트한다. 점 $p$를 정점으로 하는 크기가 $\theta$인 부채꼴을 $W_1$이라 하자. $W_1$의 기울기

를 0도부터 360도까지 조금씩 증가시키면 $S$의 점들이 $W_1$에 들어오기도 하고 나가기도 한다. $S$의 점이 하나 새로 들어오거나 ($W_1$의 왼쪽 경계에 점이 위치하는 경우), 있던 점이 하나 나갈 때를 ($W_1$의 오른쪽 경계에 점이 위치하는 경우) 각각 하나의 이벤트로 생각하여 이벤트가 일어날 때마다 필요한 일을 수행한다.

지금 이벤트가 일어났다고 가정하자. 현재의 $W_1$에 포함된 S의 점들의 집합을 $R$이라 하자. 이제 $S$의 다른 점을 정점으로 하여 크기가 $\theta$인 부채꼴 $W_2$를 배치하여 $\overline{R}=S-R$에 속하는 점들을 모두 포함할 수 있느냐를 조사한다. $W_2$의 정점은 $R$중의 한 점이 꼭 되어야 하고, $W_2$는 $p$를 포함하여야 한다. $R$의 각 점에서 $CH(\overline{R})$에 두 개의 접선을 그려서 생긴 부채꼴이 $p$를 포함하고 각이 $\theta$ 이하인 것이 있는가를 조사하는 것과 같다.

이를 위해서는 $CH(\overline{R})$를 계산하는 작업이 필요한데, 두 이벤트 사이의 $\overline{R}$은 점 하나 차이다. 따라서 이것은 볼록껍질 관리 알고리즘을 이용한다[5]. 추가나 삭제 하나당 $O(\log^2 n)$ 시간이 걸리므로, $CH(\overline{R})$을 $W_1$이 360도 회전하는 동안 관리하는데는 $O(n\log^2 n)$시간이면 충분하다.

$R$의 각 점에서 $CH(\overline{R})$에 접선을 두 개씩 긋는데, 매 이벤트마다 $R$의 모든 점에서 접선을 긋는 일을 하면, 매 이벤트마다 $O(n\log n)$의 일을 해야하고 전체로는 $O(n^2\log n)$ 일을 해야 한다. log 항을 하나 줄이기 위해서, 직전 이벤트에서 구한 접선을 기억한다. 즉, $R'$을 직전 이벤트에서의 $R$이라 하고, $R$을 현재 것이라 하자. 그리고, $R'$의 한 점 $r$에서 $CH(\overline{R'})$에 이르는 두 접선의 접점 $r_1$, $r_2$를 알고 있다고 가정하자. 그러면, $r$에서 $CH(\overline{R})$에 접하는 두 접점 $r_3$, $r_4$은 어떻게 구하는지를 알아보자. 먼저, $CH(\overline{R})=CH(\overline{R'})$이면 할 일이 없으므로 둘이 다르다고 가정하자.

점이 하나 추가된 경우 (즉, $\overline{R}=\overline{R'}\cup\{q\}$), $q$는 반드시 $CH(\overline{R})$의 꼭지점이다. 이유는 $q$는 $CH(\overline{R'})$ 외부에 존재하기 때문이다. 따라서, $r_3=r_1$이고 $r_4\in\{r_2, q\}$가 된다. 즉, 두 접점이 모두 변하지는 않으며, 하나가 변할 경우에는 후보는 $q$가 된다. $r$과 $q$를 지나는 직선이 $CH(\overline{R})$에 접하면 $r_4=q$이고, 아니면 $r_4=r_2$가 된다. $W_1$이 한번 회전하는 동안, 최대 $n$번의 점 추가가 생기므로, $r$ 하나에 대해서 하는 일은 $O(n)$이고, 모든 $r$에 대해서 고려하면 $O(n^2)$이다.

점이 하나 삭제가 된 경우, (즉, $\overline{R}=\overline{R'}-\{q\}$), $q$가 사라진 $CH(\overline{R})$에는 여러개의 꼭지점이 새로 생기고 따라서 새로운 내각이 여러 개 생긴다. 주의: $q$는 반드시 $CH(\overline{R'})$의 꼭지점이다. 즉, $q_1, q_2$를 $CH(\overline{R'})$에서 $q$가 이웃한 두 꼭지점이라 하면, $q_1, q_2$는 여전히 $CH(\overline{R})$에서도 꼭지점이 된다. 따라서 $CH(\overline{R})$의 꼭지점 중에서 $q_1$과 $q_2$ 사이에 있는 꼭지점들은 새로 생긴 것들이다. 따라서, $r_3=r_1$이고, $r_4\in\{r_2, q_1$부터 $q_2$까지의 $CH(\overline{R})$의 꼭지점들$\}$이다. $q_1$부터 $q_2$까지의 $CH(\overline{R})$의 꼭지점들을 하나씩 보면서 그것과 $r$을 잇는 직선이 $CH(\overline{R})$에 접하는가를 조사하여 접하면 그 점이 $r_4$가 되고, 모든 점에서 접하지 않으면 $r_4=r_2$가 된다. 따라서 $q_1$과 $q_2$ 사이에 새로 생기는 꼭지점의 수가 중요한데, 이들은 한번 생기면 삭제될 때까지 그대로 볼록껍질에 남아 있다. 따라서, 이들의 수를 $W_1$이 한번 회전하는 동안의 최대 $n$번의 삭제들에 대해서 전부 합하면 $r$하나에 대해서 역시 $O(n)$이고, 모든 $r$에 대해서 하면 $O(n^2)$이다.

따라서, 한 부채꼴의 정점을 $p$에 고정한 경우, 두 부채꼴로 모든 점들을 포함할 수 있는가를 조사하는데 걸리는 시간은 $O(n^2)$시간이다. 이를 모든 $p$에 대해서 반복해야 하니까, 주어진 $\theta$에 대해서 결정 알고리즘의 시간은 $O(n^3)$이 된다.

그런데, $\theta$가 될 수 있는 후보의 개수가 모두 $O(n^3)$이므로 ($p$ 하나마다 $O(n^2)$개의 후보가 있음), 이들 후보들을 모두 계산한 후 소트하여 리스트를 만든 후, 방금 만든 결정 알고리즘을 이용하여 이진 탐색을 하여 최적값 $\theta^*$를 찾는다.

**정리 7:** 점 집합 $S$를 포함하면서 두 정점 모두 상대방 부채꼴 안에 있는 두 최적 부채꼴을 $O(n^3\log n)$ 시간에 구할 수 있다.

## 참 고 문 헌


[1] P.K. Agarwal and M. Sharir, Efficient algorithms for geometric optimization, ACM Computing Surveys, vol. 30, pp. 412-458, 1998.

[2] R. Cole, Slowing down sorting networks to obtain faster sorting algorithms, Journal of ACM, 34, pp. 200--208, 1987.

[3] J. Friedman, J. Hershberger, and J. Snoeyink, Efficiently planning compliant motion in the plane, SIAM Journal on Computing, vol. 25, no. 3, pp. 562--599, 1996.

[4] N. Megiddo, Applying parallel computation algorithms in the design of serial algorithms, Journal of ACM, vol. 30, pp. 852--865, 1983.

[5] M.H. Overmars and J. van Leeuwen, Maintenance

of configurations in the plane, Journal of Computer and Stystem Science, vol. 23, pp. 166-204, 1981.
[6] F.P. Preparata and M.I. Shamos, Computational Geometry: an Introduction, Springer-Verlag, New York, 1985.
[7] P.K. Agarwal, M. Sharir and E. Welzl, The discrete 2-center problem, Proceedings of 13th Annual Symposium on Computational Geometry, pp. 147--155, 1997.
[8] D. Eppstein, Faster construction of planar two-centers, Proceedings of 8th ACM-SIAM Symposium on Discrete Algorithms, pp. 131--138, 1997.
[9] J.W. Jaromczyk and M. Kowaluk, Orientation independent covering of points in $R^2$ with pairs of rectangles, Lecture Notes in Computer Science, vol. 871, pp. 71-78, 1996.
[10] M.J. Katz, K. Kedem, and M. Segal, Discrete rectilinear 2-center problems, Computational Geometry, vol. 15, pp. 203--214, 2000.
[11] M. Sharir, A near-linear algorithm for the planar 2-center problem, Proceedings of 12th Annual ACM Symposium on Computational Geometry, pp. 106--112, 1996.

김 성 권
1981년 2월 서울대학교 계산통계학과 학사. 1983년 2월 한국과학기술원 전산학과 석사. 1990년 8월 University of Washington 전산학 박사. 1991년 3월 ~ 1996년 2월 경성대학교 전산통계학과 조교수. 1996년 3월 ~ 현재 중앙대학교 컴퓨터공학과 부교수. 관심분야는 계산기하학, 암호 응용 및 정보보호, 생물정보학

김 순 석
1997년 2월 진주대학교 컴퓨터공학과 학사. 1999년 2월 중앙대학교 컴퓨터공학과 석사. 1999년 3월 ~ 현재 중앙대학교 대학원 컴퓨터 공학과 박사과정. 관심분야는 암호 프로토콜, 이동통신 보안, 정보보호

신 찬 수
정보과학회논문지 : 시스템 및 이론
제 28 권 제 2 호 참조

여 상 수
1997년 2월 중앙대학교 컴퓨터공학과 학사. 1999년 2월 중앙대학교 컴퓨터공학과 석사. 2000년 3월 ~ 현재 중앙대학교 대학원 컴퓨터공학과 박사과정. 관심분야는 컴퓨터이론, 암호 응용 및 정보보호, 생물정보학