JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 19, No. 2 : 450~454 Feb. 2015

주어진 정사각형 영역안의 점들의 가중치 합의 최대화

김재훈*

Maximizing the Sum of Weights of Points in a Given Square

Jae-hoon Kim*

Department of Computer Engineering, Busan University of Foreign Studies, Busan 609-815, Korea

요 약

본 논문에서는 평면상에 가중치를 가진 점들이 주어질 때, 임의의 상수 r에 대해서 변의 길이 r인 정사각형 영역을 고려해서 이 안에 속하는 점들의 가중치 합이 최대가 되는 영역을 찾고자 한다. 변의 길이가 정해져 있지 않은 경우에 임의의 사각형 영역을 찾는 문제에 대한 연구가 있었다. 본 논문에서는 상수 r이 주어질 때, 변의 길이 r인 정사각형 영역을 찾는 문제를 다룬다. 우리는 동적 환경 하에서의 일차원 문제를 풀고, 이를 이용해서 $O(n\log n+rn)$ 시간 복잡도를 갖는 알고리즘을 제안한다.

ABSTRACT

In this paper, when points with weights are given in a plane, for an arbitrary constant r, we shall find a square area S such that the sum of weights of points belonging to S is maximized. If the length of the side of S is not given, the problem to find arbitrary rectangular area has been studied. In this paper, we will consider the problem to find a square area with a side of a length r when a constant r is given. We will solve the one dimensional problem in dynamic environment and propose an algorithm with the time complexity of $O(n\log n+rn)$.

키워드 : 가중치, 정사각형 영역, 시간 복잡도, 알고리즘, 동적 환경

Key word : weight, square area, time complexity, algorithm, dynamic environment

접수일자 : 2014. 10. 28 심사완료일자 : 2014. 12. 04 게재확정일자 : 2014. 12. 16

* **Corresponding Author** Jae-Hoon Kim(E-mail: jhoon@bufs.ac.kr, Tel:+82-51-509-6226)
Department of Computer Engineering, Busan University of Foreign Studies, Busan 609-815, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2015.19.2.450 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

평면상에 n개의 점들이 임의의 위치에 주어진다. 단, 어떠한 두 점도 같은 x 또는 y좌표를 가지지 않는다고 가정한다. 각 점 p는 임의의 정수 값을 가중치 $w(p)$로 가진다. 이 가중치는 음수일 수 있음에 주목한다.

임의의 양의 정수 r이 주어질 때, 평면상에서 x 또는 y축에 평행한 변들을 가지고 모든 변의 길이가 r인 정사각형 영역 R을 고려한다. 특별히 영역 R은 닫힌(closed) 영역을 고려해서 점 p가 R에 속하다는 것은 p가 영역 R의 안쪽에 위치하거나 또는 R의 테두리 변에 위치한다는 것을 의미한다. 우리는 평면상에 주어진 임의의 정사각형 영역 R에 대해서, R에 속하는 점들의 가중치 합 $W(R) = \sum_{p \in R} w(p)$을 생각해서 영역 R의 가중치라고 부른다. 임의의 양의 정수 r이 주어질 때, x 또는 y축에 평행한 길이 r의 변을 가진 정사각형 영역 R 중에서 가중치 $W(R)$이 최대가 되는 영역을 찾고자 한다.

점들의 가중치를 어떻게 선택하느냐에 따라서 이 문제는 기계 학습(machine learnig)[1]과 데이터 분류(classification)와 클러스터링(clustering)[2]과 같은 여러 응용 분야와 관련된다. 예를 들어, 데이터 클러스터들은 원 또는 사각형과 같은 기하학적 모양을 사용해서 얻을 수 있다[2, 3].

Ⅱ. 관련 연구

이전 논문들에서는 영역의 한 변의 길이 r이 미리 정해지지 않는 경우들을 주로 다루었다. 이 경우에는 임의의 변의 길이를 가진 사각형 영역을 고려할 수 있다. 사각형 변의 길이 제한이 없는 문제의 일차원 버전은 다음과 같이 정의된 최대-합 연속 부순서(MCS, Maximum-Sum Consecutive Subsequence) 문제이다.

[MCS-문제] 수열 $a_1, a_2, ..., a_n$과 각 a_i에 대한 가중치 w_i가 주어질 때(여기서, w_i는 음수일 수 있다), $w_j + w_{j+1} + \cdots + w_k$가 최대가 되는 연속적인 부순서 $a_j, a_{j+1}, ..., a_k$를 찾으시오.

이 문제는 점들이 정렬되어 있을 때, $O(n)$ 시간에 풀 수 있다[4].

이 MCS-문제에 대해서 중간에 점들의 가중치가 바뀌는 동적 환경에 대한 연구가 있었다[5]. 이 논문에서 저자들은 점의 가중치에 변화가 있을 때 $O(\log n)$에 새로운 답을 찾는다.

이차원 문제에 대해서 [5]에서 처음 소개되었다. 이 문제에서는 변의 길이에 대한 제한이 없이 임의의 사각형을 영역을 고려해서 영역안의 점들의 가중치의 합이 최대가 되는 사각형 영역을 찾는다. [5]에서 저자들은 $O(n^2 \log n)$ 시간 알고리즘을 제안하였다. 이 시간 복잡도는 [6]에서 개선되었다. 저자들은 $O(n^2)$ 시간 알고리즘을 제안하여 이전 결과를 개선하였다.

영역이 x 또는 y축에 평행한 변을 가진 사각형과 다른 모양인 경우에 대한 연구들도 있었다. [7]에서는 영역으로 볼록 다각형(convex polygon)을 고려하였다. 저자들은 포함하는 점들의 가중치의 합이 최대가 되는 볼록 다각형을 찾는 $O(n^3)$ 시간 알고리즘을 제안하였다.

본 논문에서는 이전 결과들과 다르게 사각형 영역의 한 변의 길이 r을 입력으로 받는다. 그리고 포함하는 점들의 가중치의 합이 최대가 되는 변의 길이가 r인 정사각형 영역을 찾는다.

Ⅲ. 알고리즘

우선 본 문제의 일차원 문제를 생각해 볼 수 있다. 다시 말해서, 직선상의 n개의 점들과 그들의 가중치가 주어질 때, 길이 r인 구간 안에 속하는 점들의 가중치의 합이 최대가 되는 구간을 찾는 문제이다. 이것은 구간의 길이가 미리 주어진다는 점에서 MCS-문제와 구별된다.

이 문제는 점들이 정렬되어 있다면, 길이 r인 윈도우를 왼쪽에서 오른쪽으로 움직이면서 최대 값을 업데이트하는 방법으로 $O(n)$의 시간 복잡도로 풀 수 있다.

이 일차원 문제에서 어느 한 점의 가중치 값이 변하는 동적(dynamic) 일차원 문제를 생각해 볼 수 있다. 이 경우에 변화한 입력에 대해서 문제의 답을 빨리 찾는 방법에 대해서 생각할 수 있다. 다음 장에서 우리는 이

동적 일차원 문제에 대한 해답을 제시할 것이고 시간 복잡도 $O(\log n + r)$에 해결할 수 있음을 보일 것이다.

위의 동적 일차원 문제의 해법을 이용해서 이차원 문제의 해법을 생각해 볼 것이다. 아래 그림 1에서와 같이 평면상에 두 수평선 ℓ_1과 ℓ_2를 생각해서 이들 사이의 수직 길이가 r이라고 하자. 이 두 수평선 사이의 공간을 스트립(strip)이라고 한다. 이 스트립에 존재하는 점들을 아래 수평선 ℓ_2에 투영(projection)하면 일차원 직선상의 점들을 얻을 수 있다. 여기서 일차원 문제를 풀 면 하나의 스트립안의 점들에 대해서 변의 길이가 r인 정사각형 영역에 속한 점들의 가중치 합이 최대가 되는 영역 R을 찾을 수 있다.

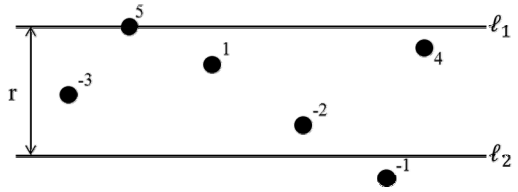

그림 1. 평면상의 스트립 S
Fig. 1 Strip S in a plane

이상과 같이 한 스트립 S에 대한 문제를 해결할 수 있음을 보았다. 이를 이용해서 본래 문제에 대한 해법을 생각해 볼 것이다. 우리는 항상 스트립 S의 위쪽 수평선에 적어도 한 점이 지난다고 가정한다. 그러면 평면상에서 위쪽에서 아래쪽 방향으로 만나는 한 점 p를 고려하고 이 점을 지나는 한 수평선 ℓ을 생각하면 ℓ을 위쪽 변으로 하는 길이 r인 스트립 S 를 생각할 수 있다. 결과적으로 평면상의 위쪽에서 아래쪽으로 스트립 S를 움직이면서 S 안에서의 문제를 해결한다.

평면상의 점들을 y좌표 값이 감소하도록 정렬해서 $p_1, p_2, \ldots, p_n$으로 표시한다. 처음에 스트립 S의 위쪽 변은 점 p_1을 지난다. 점 $p_1, \ldots, p_k$가 S에 속한다면, 동적 일차원 문제에서 초기에 모든 점들의 가중치를 0으로 하였다가 점 $p_1, \ldots, p_k$의 가중치 값을 각각 $w(p_1), \ldots, w(p_k)$로 변화시키면서 동적 일차원 문제를 푼다.

다시 S의 위쪽 변이 지나는 점을 p_1에서 p_2로 바꾸면서 스트립을 아래로 이동한다. 그러면 S에 속하는 점에서 p_1은 제외되고 새로운 점들이 추가될 수 있다. 그러면 p_1의 가중치는 0으로 바꾸고 새로 추가되는 점들의 가중치를 변화시킨다. 이러한 상태에서 역시 동적 일차원 문제를 풀어서 새로운 스트립에서의 문제를 해결한다.

이런 방식으로 스트립을 아래로 이동하면서 스트립이 p_n을 포함하는 경우까지 스트립 문제를 해결해서 점들의 가중치 값의 합이 최대가 되는 정사각형 영역 R을 구할 수 있다. 스트립의 위쪽 변이 지나는 점을 p_1부터 반복해서 고려해서 추가하고 만약 제거하면 점들은 다시 고려되지 않기 때문에 스트립 문제는 많아야 $O(n)$번 수행된다. 각 스트립에서 동적 일차원 문제가 $O(\log n + r)$시간에 해결될 수 있으므로 총 수행시간은 $O(n \log n + rn)$이다.

Ⅳ. 동적 일차원문제

이전 장에서 언급한 것과 같이 문제의 답을 찾기 위해서 일차원에서의 동적 문제를 풀어야 한다. 이 장에서는 일차원 동적 문제를 업데이트가 일어날 때 마다 $O(\log n + r)$시간에 해결하는 알고리즘을 제안할 것이다.

일직선상에 점들 $p_1, p_2, \ldots, p_n$이 위치가 증가하는 순서로 주어지고 그들의 가중치 $w_1, w_2, \ldots, w_n$이 주어질 때, 길이 r인 구간 안에 속한 점들의 가중치의 합이 최대가 되는 구간을 찾는 문제를 생각한다. 특별히 이 문제의 동적 버전, 다시 말해서, 어떤 가중치가 변하는 경우의 문제를 다룬다.

위 동적 문제의 답을 효율적으로 찾기 위해서 그림 2와 같은 균형 이진트리(balanced binary tree) T를 이용할 것이다. 트리 T의 각 노드는 직선상의 한 구간의 상태를 나타낼 것이다. 우선 잎(leaf) 노드가 나타내는 구간을 생각해 본다. 직선을 길이 r인 구간들 $[a_1, b_1), \ldots, [a_m, b_m)$로 나눈다. 여기서, $a_1 = p_1$ 이고 $b_1 = p_1 + r$. 그러면 구간 $[a_1, b_1)$에 점 p_i들이 포함될 수 있다. a_2는 구간 $[a_1, b_1)$에 포함되지 않는 점들 중에서 가장 작은 점 p_k로 정의한다. 계속해서 이런 방식으로 구간 $[a_m, b_m)$이 가장 큰 점 p_n을 포함하도록 정의한

다. 임의의 구간 $[a_k, b_k)$에 대해서, 이 구간에 속한 점들을 $p_\alpha, \ldots, p_\beta$라고 하면, 각 점 p_h는 구간 $[a_k, p_h)$와 $[p_h, b_k)$에 속한 점들의 가중치 합을 각각 저장한다. 또한 이 구간에 속한 모든 점들의 가중치 합을 저장한다.

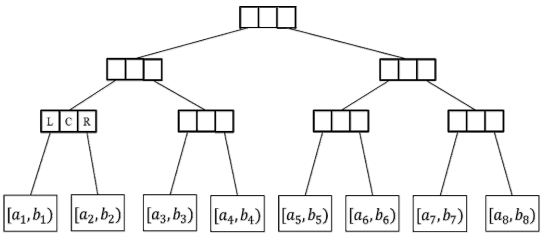

그림 2. 균형 이진트리 T
Fig. 2 Balanced binary tree T

트리 T의 내부 노드(internal node)는 직선상의 한 구간 $[a_i, b_j)$의 상태를 나타낸다. 이를 위해서 세 개의 값 L, R, C를 저장한다. 여기서 L은 a_i를 왼쪽 끝점으로 하는 길이 r인 구간 안에 속한 점들의 가중치 합이다. R은 b_j를 오른쪽 끝점으로 하는 길이 r인 구간 안에 속한 점들의 가중치 합이다. C는 a_i과 b_j를 포함하지 않는 구간 내부에서 길이 r인 구간에 속한 점들의 가중치 합 중 최대값을 나타낸다.

각 내부 노드에서 세 개의 값 L, R, C을 어떻게 계산하는지 설명한다. 구간 $[a_i, b_j)$를 나타내는 내부 노드에 대해서, 두 자식 노드는 각각 구간 $[a_i, b_k)$와 $[a_{k+1}, b_j)$를 나타낸다. 각 자식 노드의 세 값을 L_1, R_1, C_1과 L_2, R_2, C_2이라고 하자. 분명히 $L = L_1$이고 $R = R_2$이다. C를 계산하는 방법을 생각해보자. 구간 $[a_i, b_k)$에서 b_k를 오른쪽 끝점으로 하는 길이 r인 구간에 속한 임의의 점 p_h는 $[p_h, b_k)$에 속한 점들의 가중치 합을 저장하고 있다. 또한 $[a_{k+1}, b_j)$에서 a_{k+1}을 왼쪽 끝점으로 하는 길이 r인 구간에 속한 임의의 점 p_h는 $[a_{k+1}, p_h)$에 속한 점들의 가중치 합을 저장한다. 이를 이용해서 $[a_i, b_k)$의 뒷부분과 $[a_{k+1}, b_j)$의 앞부분에 걸친 길이 r인 구간들 중 가중치 합의 최대값 C_3를 $O(r)$시간에 계산할 수 있다. 그러면 C는 C_1, C_2, C_3 중 최대값이다.

위와 같은 균형 이진 트리 T를 생성하는데 걸리는 시간을 생각해보자. 트리 T의 잎노드를 생성하는 것은 점들을 왼쪽에서 오른쪽으로 고려하면서 길이 r인 각 구간 $[a_i, b_i)$을 나눌 수 있고 이 구간에 속한 각 점 p_k에서 $[a_i, p_k)$에 속한 점들의 가중치 합을 계산할 수 있다. 또한 점들을 오른쪽에서 왼쪽으로 고려하면서 각 점 p_k에서 $[p_k, b_i)$에 속한 점들의 가중치 합을 계산할 수 있다. 또한 T의 내부 노드를 생성하는 과정에서 세 개의 값들은 최대 $O(r)$시간에 계산할 수 있다. 따라서 균형 이진 트리 T를 생성하는데 총 $O(rn)$시간이 소요됨을 알 수 있다. 이진 트리 T를 이용해서 동적 일차원 문제의 효율적인 해법을 제시할 것이다. 일직선상의 점들 $p_1, p_2, \ldots, p_n$이 순서대로 주어지고 임의의 점 p_k의 가중치가 변화할 때 가중치의 합이 최대가 되는 길이 r인 구간을 새롭게 찾아야 한다.

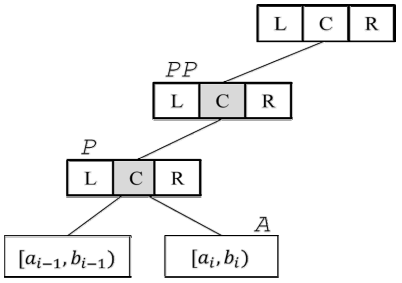

그림 3. T에서 최대값의 계산
Fig. 3 Computation of maximum in T

점 p_k가 T의 하나의 잎 노드 A가 나타내는 구간 $[a_i, b_i)$에 속한다고 가정하자. 이 구간에 속한 점들의 저장 값이 변화하고 $O(r)$시간에 업데이트 할 수 있다. 이 변화가 A의 부모 노드 P의 L, R 또는 C 값을 변화시킬 수 있지만 L과 R은 $O(1)$시간에 업데이트 할 수 있고 C 값은 $O(r)$시간에 계산할 수 있다. 또한 P의 부모 노드 PP의 C값 계산에도 $O(r)$시간을 소요해야 한다. 하지만 PP의 부모 노드에서부터는 $O(1)$시간에 세 값 모두를 계산할 수 있다. 예를 들어, 그림 3에서 노드 $A = [a_i, b_i)$의 한 점 p_k의 가중치가 변해서 구간의 점들의 저장 값이 변해야 한다고 가정하자. 그

러면, 그림의 어두운 색으로 표시된 부분만이 $O(r)$ 시간의 계산을 요구한다. 이와 같이 루트 노드로의 경로를 따라서 경로상의 내부 노드들의 세 값들을 업데이트 해 나가면 루트에서 문제의 새로운 답을 찾을 수 있다. 따라서 이와 같은 업데이트에 소요되는 총 시간은 $O(\log n + r)$ 이 된다.

V. 결 론

이차원 점들에 대한 문제에서 이전 논문들[5, 6]은 사각형 영역의 변의 길이에 제한이 없었다. 본 논문에서는 변의 길이 r 주어질 때, 가중치 합이 최대가 되는 점들을 포함하는 정사각형 영역을 찾는 문제를 다루었다. 우리는 본 문제의 일차원 동적 문제를 풀고 이를 이용해서 이차원 문제에 대한 답을 제안하였다. 또한 본 논문이 제안한 알고리즘의 시간 복잡도가 $O(n \log n + rn)$ 임을 보였다.

앞으로의 연구로는 본 문제의 3차원 버전을 생각해 볼 수 있다. 논문에서 제안한 알고리즘을 3차원 상의 점들에 대해서 적용해서 시간 복잡도를 분석해 볼 수 있을 것이다. 또한 다른 모양의 영역에 대한 문제를 연구해 볼 수 있을 것이다.

감사의 글

본 연구는 2014년도 부산외국어대학교 학술연구조성비에 의해 연구되었으므로, 관계부처에 감사드립니다.

김재훈(Jae-Hoon Kim)

1994년 서강대학교 수학과 이학사
1996년 KAIST 수학과 이학석사
2003년 KAIST 전산과 공학박사
2003년 ~ 현재 부산외국어대학교 컴퓨터공학과 교수
※관심분야 : 알고리즘, 최적화, 스케줄링

REFERENCES

[1] D. Dobkin, D. Gunopulos, and W. Maass, "Computing the maximum bichromatic discrepancy, with applications to computer graphics and machine learning," *J. Comput. Syst. Sci.,* vol. 52, no. 3, pp. 453-470, 1996.

[2] J. Eckstein, P. Hammer, Y. Liu, M. Nediak, and B. Simeone, "The maximum box problem and its application to data analysis," *Comput. Optim. Appl.,* vol. 23, no. 3, pp. 285-298, 2002.

[3] B. Aronov and S. Har-Peled, "On approximating the depth and related problems," *SIAM J. Comput.,* vol. 38, no. 3, pp. 899-921, 2008.

[4] J. Bentley, "Programming pearls: algorithm design techniques," *Commun. ACM,* vol. 27, no. 9, pp. 865-873, 1984.

[5] C. Cortes, J. M. Diaz-Banez, P. Perez-Lantero, C. Seara, J. Urrutia and I. Ventura, "Bichromatic separability with two boxes: A general approach," *Journal of Algorithms Cognition, Informatics and Logic,* vol. 64, no. 1, pp. 79-88, 2009.

[6] J. Barbay, Timothy M. Chan, Gonzalo Navarro, and Pablo Perez-Lantero, "Maximum-weight planar boxes in $O(n^2)$ time (and better)," *Information Processing Letters,* vol. 114, no. 1, pp. 437-445, 2014.

[7] C. Bautista-Santiago, J. M. Diaz-Banez, D. Lara, P. Perez-Lantero, J. Urrutia, and I. Ventura, "Computing optimal islands," *Operational Research Letters,* vol. 39, no. 4, pp. 246-251, 2011.