**최근접 쌍 찾기 알고리즘**

2018182024 이동현

최근접 쌍 찾기의 경우 점마다 거리를 계산해 최근접 쌍을 구할 경우 ) 만큼의 시간이 걸리게 된다. 하지만 이를 분할해서 최근접 쌍을 구하면 이보다 더 적은 시간이 걸리게 된다. 따라서 분할을 하여 최근접 쌍을 구할 것이다.

우선 구조체 배열에 점들의 좌표를 넣고, 이를 x좌표에 대해 정렬을 한다. 정렬의 경우 퀵 정렬 또는 합병 정렬 사용 시 의 시간이 소요된다.

그리고 정렬된 이 점들을 왼쪽, 오른쪽으로 절반을 나누고, 나뉘어진 부분에 대해서도 분할을 계속한다. 분할할 때, 분할된 공간안의 점들의 개수가 3개 이하일 경우엔 분할을 하지 않고, 점들의 거리를 계산해 최솟값 d를 구한다.

분할된 공간 뿐만 아니라 분할하는 부분 근처에서 최근접 쌍이 존재할 수 있으므로, 분할하는 경계선 기준으로 d만큼 떨어진 거리에 대해서도 최솟값을 구해 d와 비교하고 최솟값을 정한다.

위의 알고리즘은 최적화 되어있지 않기에 이를 최적화하면 더 빠른 알고리즘을 얻을 수 있는데, 최적화의 방법은 최솟값을 구하고 최솟값보다 비교하는 점들의 x의 거리 또는 y의 거리가 크면 이 값들은 비교하지 않는 것이다.

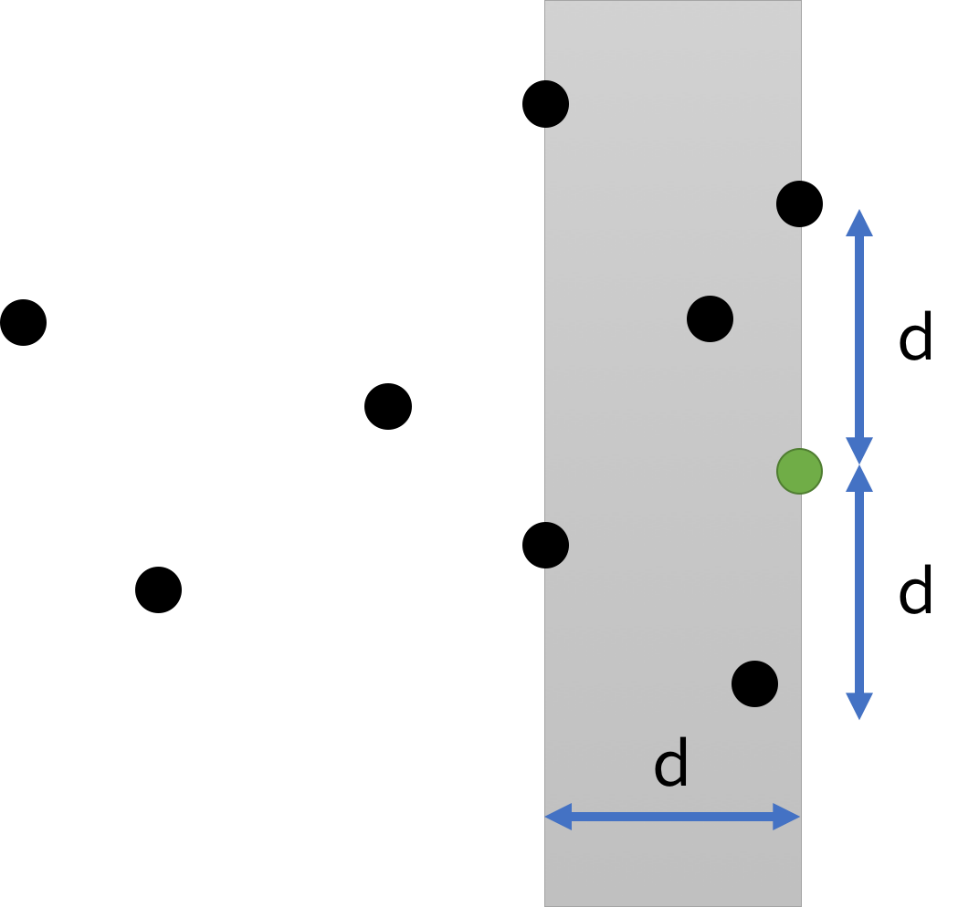
우선 우리는 분할된 왼쪽, 오른쪽을 먼저 계산하는 것을 알 수 있다. 앞서 점들은 x좌표에 대해 오름차순으로 정렬되어 있고, 각 부분에서 가장 왼쪽에 있는 점을 기준으로 오른쪽으로 가며 점들에 대해 거리를 비교한다고 하면, 기준점에서 x가 d만큼 떨어진 구간 내의 점들에 대해서만 비교를 하고 또 구간 내에서 기준점에서 y가 d만큼 떨어진 구간 내의 점만 검사를 한다. 만약 x가 d보다 큰 점을 만나면, 기준점을 오른쪽으로 옮기고 위의 검사를 반복해 왼쪽, 오른쪽의 최솟값을 구한다. 이때, 왼쪽에서 구한 최솟값을 이용해 오른쪽도 구간을 나눠 검사하는 방식이다.

왼쪽, 오른쪽 검사를 하고 왼쪽부분에서 x의 최댓값, 오른쪽부분에서 x의 최솟값의 평균을 경계선으로 잡고, 경계선을 기준으로 x가 d만큼 떨어진 구간 내의 점들, 즉 중간구간에 대해 최솟값을 비교해주면 된다.

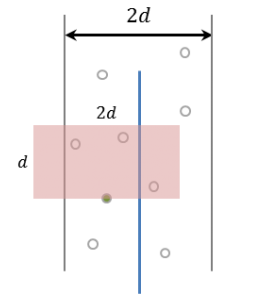
중간구간의 점들을 y값을 오름차순으로 따로 정렬하고, 가장 아래의 점을 기준으로 위로 올라가면서 거리를 비교하되, 기준점을 중심으로 y거리가 d만큼 떨어진 구간 내의 점만 검사를 하고, 그 구간내에서도 x거리가 d만큼 떨어진 구간 내의 점만 검사를 하는 식으로 비교한다.

최근접 쌍을 구하기 위해 처음 x값에 대해 오름차순 정렬시 의 시간이 소요되고, 문제를 분할하고, 합병하는 과정에서 의 시간이 소요되고, 중간구간에서 y값에 대해 오름차순 정렬에 의 시간이 소요되고, 중간중간 상수값 만큼의 시간이 소요된다.

즉 위의 알고리즘의 시간 복잡도는 이 됨을 알 수 있다.



왼쪽, 오른쪽 구간 비교 시 기준점을 중심으로 x좌표가 d만큼 떨어진 구간 내, 그 구간 내에서 y좌표가 d만큼 떨어진 구간 내의 점들만 검사함



중간 구간 비교 시 기준점을 중심으로 y좌표가 d만큼 떨어진 구간 내, 그 구간 내에서 x좌표가 d만큼 떨어진 구간 내의 점들만 검사함