The Smallest Enclosing Circle Problem

Submitted By-
Prakhar Rai and Naman Jain

Under the Guidance of -
Dr. Kapil Ahuja

Department of Computer Science and Engineering

Indian Institute of Technology Indore

Overview

1. Introduction

   -Motivation

   -Smallest Enclosing Circle : Formal Definition

2. Solution : Smallest Enclosing circle problem

   - Design and Analysis Of Various Algorithms

   - Welzl's Algorithm

3. Experimental Results

   - Time Analysis

Our Mall is Centrally Located!!

In real world scenario we have heard the above line with respect to several facilities.For example,

The mall.
Schools, colleges and other educational institutions

Hospitals.

Housing Societies.

So, the ”centrally located” terminology is apparently known to all of us.

Also, it gives us an intuitive idea that in some sense it is good for us in terms of ”reduced distance” !!

Lets look at it more closely then !!

Motivation

Let us consider a simple scenario in which we want to build a charitable hospital in some rural area.

Obvious aim would be to minimize the distance between the hospital and residance of each person in the locality

One solution could be :

- Consider each residance of the locality as a point.
- Find the smallest circle that encloses all the points.
- If the hospital is located at the center of the circle, it minimizes the distance between the hospital and the farthest residence.

Hence, solves the above problem.

Formal Definition: Smallest Enclosing Circle

Given a set of S of 2D points, find the circle C with the smallest radius such that all the points in S are contained in either C or its circumference.

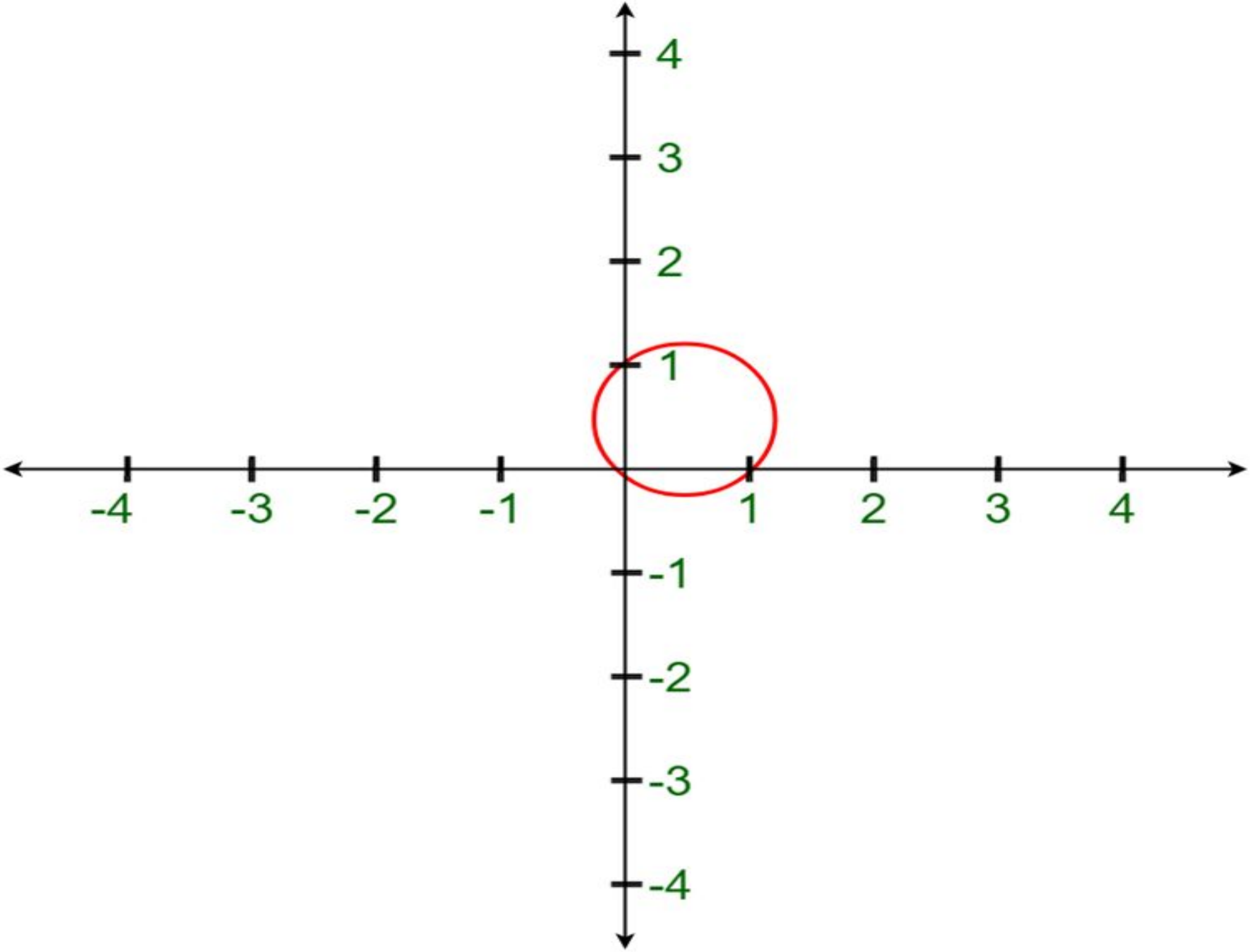
EXAMPLE

Input: array[][] = {{0, 0}, {0, 1}, {1, 0}}

Output: Center = {0.5, 0.5}, Radius = 0.7071

Explanation:

On plotting the above circle with radius 0.707 and center (0.5, 0.5), it can be observed clearly that all the mentioned points lie either inside or on the circle.


4
3
2
1
-4
-3
-2
-1
1
2
3
4
-1
-2
-3
-4

Solution : A Naive Approach

A naive algorithm solves the problem in time O(n4) by testing the circles determined by all pairs and triples of points.

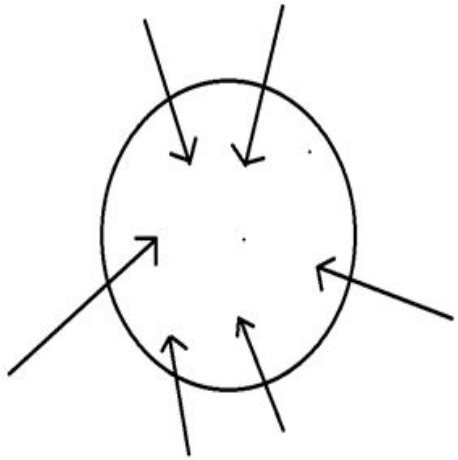
Running Time :

- The are O(n 3) such circles
- Testing all points corresponding to each circle takes O(n) time.
- The overall algorithm runs in O(n 4) time.

Naive Approach: This problem can be solved by making a few observations.

1. The first observation which can be made is that the MEC intersects at least one point. That's because if the MEC does not intersect at any point, then the circle could be further shrunk until it intersects at one of the points.

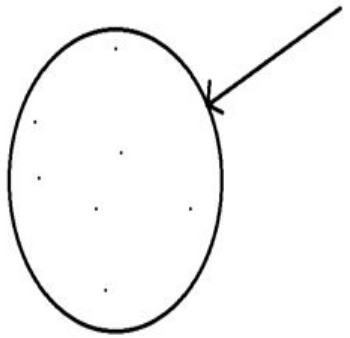
It can be easily seen the given figure that if our MEC is the one present in the figure then we could further shrink the circle to get a circle with less radius thar the given circle that encloses all the given n point of array.



Naive Approach: This problem can be solved by making a few observations.

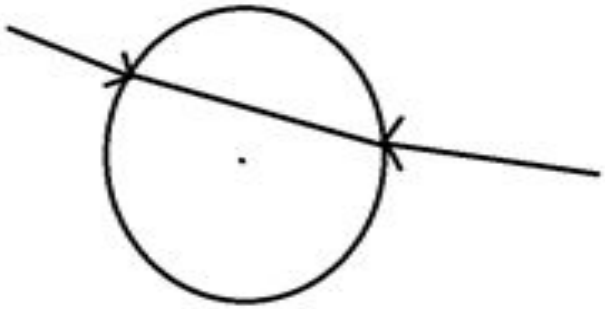
2. The second observation which can be made is that given a circle that encloses all the points and intersects at a single point, the circle can further be shrunk by moving the center towards that point while keeping the point on the circle boundary until the circle intersects one or more additional points.

From observation 1 we got that at least one point must lie on The boundary. Now from the given figure it can be easily Seen that the circle radius can be reduced by moving the center towards that point while keeping the point on the Circle boundary until the circle intersects one or more additional points.



Naive Approach: This problem can be solved by making a few observations.

3. If the circle intersects at two points(A and B) and the distance AB is equal to the circle diameter, then the circle cannot be shrunk anymore. Else, the centre of the circle can be moved towards the midpoint of AB until the circle intersects a third point(at which the circle cannot be shrunk anymore)



Conclusions

From the above observations, it can be concluded that the MEC either:

1. Intersects 2 points A and B, where AB = circle diameter. For this case, the circle's centre would be the midpoint of A and B and the radius would be half of the distance AB.
2. Intersects 3 or more points.

Naive Algorithm

```
// Go over all pair of points
for (int i = 0; i < n; i++) {
    for (int j = i + 1; j < n; j++) {

        // Get the smallest circle that
        // intersects P[i] and P[j]
        Circle tmp = circle_from(P[i], P[j]);

        // Update MEC if tmp encloses all points
        // and has a smaller radius
        if (tmp.R < mec.R && is_valid_circle(tmp, P))
            mec = tmp;
    }
}
```

Naive Algorithm

```
// Go over all triples of points
   for (int i = 0; i < n; i++) {
      for (int j = i + 1; j < n; j++) {
         for (int k = j + 1; k < n; k++) {

            // Get the circle that intersects P[i], P[j], P[k]
            Circle tmp = circle_from(P[i], P[j], P[k]);

            // Update MEC if tmp encloses all points
            // and has smaller radius
            if (tmp.R < mec.R && is_valid_circle(tmp, P))
               mec = tmp;
         }
      }
   }
```

Can we do better?

Fortunately, the answer is yes !!

- There are algorithm that make use of the convex hull, and has the running time ranging from O(nh) to O(h 3 n).
- There also exist several algorithms with running time ranging from O(n 4) to O(nlog n).
- The best known algorithms are the Megiddo's Algorithm (O(n) running time) and Welzl's Algorithm (expected O(n) running time).

Further we will have a closer look at the Welzl's Randomized Algorithm for solving the problem.

Welzl's Algorithm

Proposed by Emo Welzl in 1991. Solves the smallest enclosing disc problem, in expected O(n) time.
Insight :

- The algorithm uses the power of randomized incremental construction.
- It also exploits the fact, that if the point is not present in the disc constructed so far, then it will be present on the boundary of the new disc.
- Basically, these are the known facts, and a generalized version of these facts are used in the algorithm.

Let's have a closer look at the algorithm !!

Design : Welzl's Algorithm

Let md(P) denote the minimum enclosing disc enclosing the set of points "P" ,md(P) for a given set P of n points, is calculated in an Randomized Incremental fashion.

- Let P be the set of n points and D = md{p 1 , p 2 , ...p i }, for 1 ≤ i ≤ n points seen so far.
- Now, if p i+1 ∈ D, the we need not do anything, and now,D = md{p 1 , p 2 , ...p i , p i+1 }, and we proceed to next point.
- Else, we use the fact that p i+1 will lie on the boundary of D 0 = md{p 1 , p 2 , ...p i , p i+1 }.
- We can compute it by calling b minidisk (A, p) , which calculates the smallest disk enclosing A = {p 1 , p 2 , ...p i } with p = p i+1 on its boundary.

Design : Welzl's Algorithm

The algorithm takes a set of points P and a set R that's initially empty and used to represent the points on the boundary of the MEC as the input.The base case of the algorithm is when P becomes empty or the size of the set R is equal to 3:

- If P is empty, then all the points have been processed.
- If |R| = 3, then 3 points have already been found that lie on the circle boundary, and since a circle can be uniquely determined using 3 points only, the recursion can be stopped.

Design : Welzl's Algorithm

When the algorithm reaches the base case above, it returns the trivial solution for R, being:

- If |R| = 1, we return a circle centered at R[0] with radius = 0
- If |R| = 2, we return the MEC for R[0] and R[2]
- If |R| = 3, we return the MEC by trying the 3 pairs (R[0], R[1]), (R[0], R[2]), (R[1], R[2])
  - If none of these pairs is valid, we return the circle defined by the 3 points in R
  - If the base case is not reached yet, we do the following:
- Pick a random point p from P and remove it from P
- Call the algorithm on P and R to get circle d
- If p is enclosed by d, then we return d
- otherwise, p must lie on the boundary of the MEC
  - Add p to R
  - Return the output of the algorithm on P and R

Analysis : Welzl's Algorithm

If a point P is outside the smallest enclosing circle of set S then it must be one of the defining points of smallest enclosing circle of set S ∪ {P }.

Proof : Note that there has to to at least 2 point on the circle. If not then we can compress the circle.

Also the boundary points define the circle completely, if not, then again we can compress the circle. Suppose P is not a defining point, by above statement there are some defining point on this new circle. These defining points were present before P was added, so the radius of circle was at least equal to the radius of circle defined by these points. After adding P they are defining point, so the radius of this equal to the radius defined by these point. In other words the circle didn't change, after adding P . But this contradicts our assumption that P is outside the smallest enclosing circle.

Jung's Theorem

Probability that point P i lies outside the smallest enclosing circle of points P 1 , ..., P i−1 is ≤ 3/ i .

Proof : By Lemma 1, if the smallest enclosing circle is changed on adding the i th point then it must be one of the defining points of smallest enclosing circle of first i points, call this circle C i .

There can be at most 3 defining points for C i because C i−1 is not same

as C i , so the probability that i th point is defining point for C i is ≤ 3/ i .

Welzl's Algorithm

Algorithm 1 : minidisk (P)

1: if $P = \phi$ then

2:$D \leftarrow \phi$

3: else

4:choose $p \in P$

5:$D \leftarrow$ minidisk$(P - \{p\})$

6:if $p \notin D$ then

7:$D \leftarrow$ b minidisk$(P - \{p\}, p)$

8:end if

9: end if

10: return D

Welzl's Algorithm

Algorithm 2 : b minidisk (P, R)

1: if $P = \phi$[or $|R| = 3$] then

2:D = b minidisk(ϕ, R)

3: else

4:choose random $p \in P$

5:$D \leftarrow$ b minidisk($P - \{p\}$, R)

6:if [D defined and]$p \notin D$ then

7:$D \leftarrow$ b minidisk($P - \{p\}$, $R \cup \{p\}$)

8:end if

9: end if

10: return D

Time Analysis

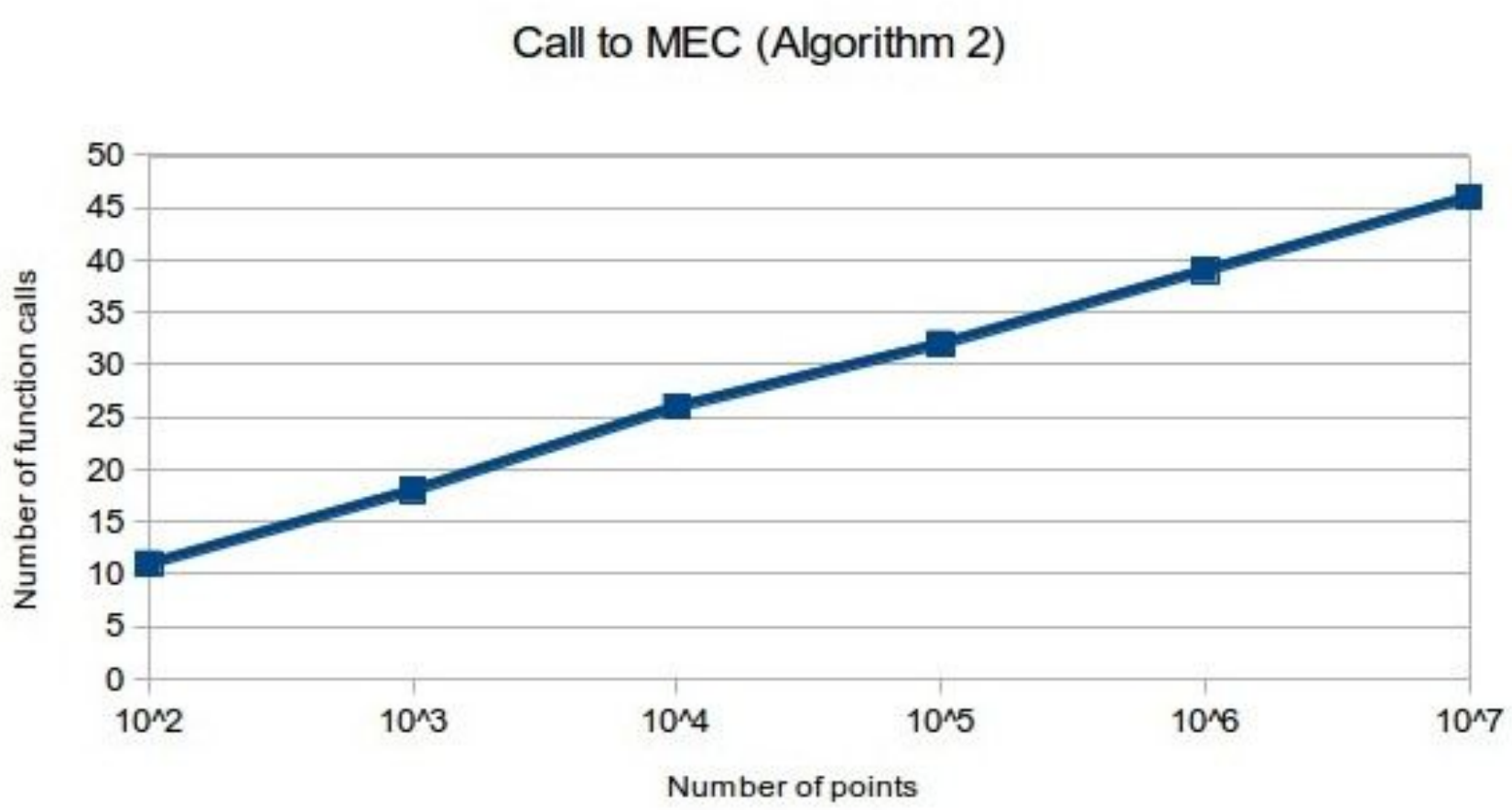
This algorithm has an expected time and space complexity of O(N) where N is the number of points. The space is due to the fact recursion is being used. To understand why the time complexity is linear, we can observe the number of different states to know how many calls can happen to the recursive function. With every call, the size of P gets reduced by one. Also, the size of R can remain the same or can be increased by one. Since |R| cannot exceed 3, then the number of different states would be 3N. Therefore, this makes the time complexity to be O(N).

In order to ensure that the input points have only a small chance of being arranged in a way that gives the worst-case running time, the points are shuffled in a random order. Welzl showed that if the points have a random ordering then the algorithm runs in expected linear time.

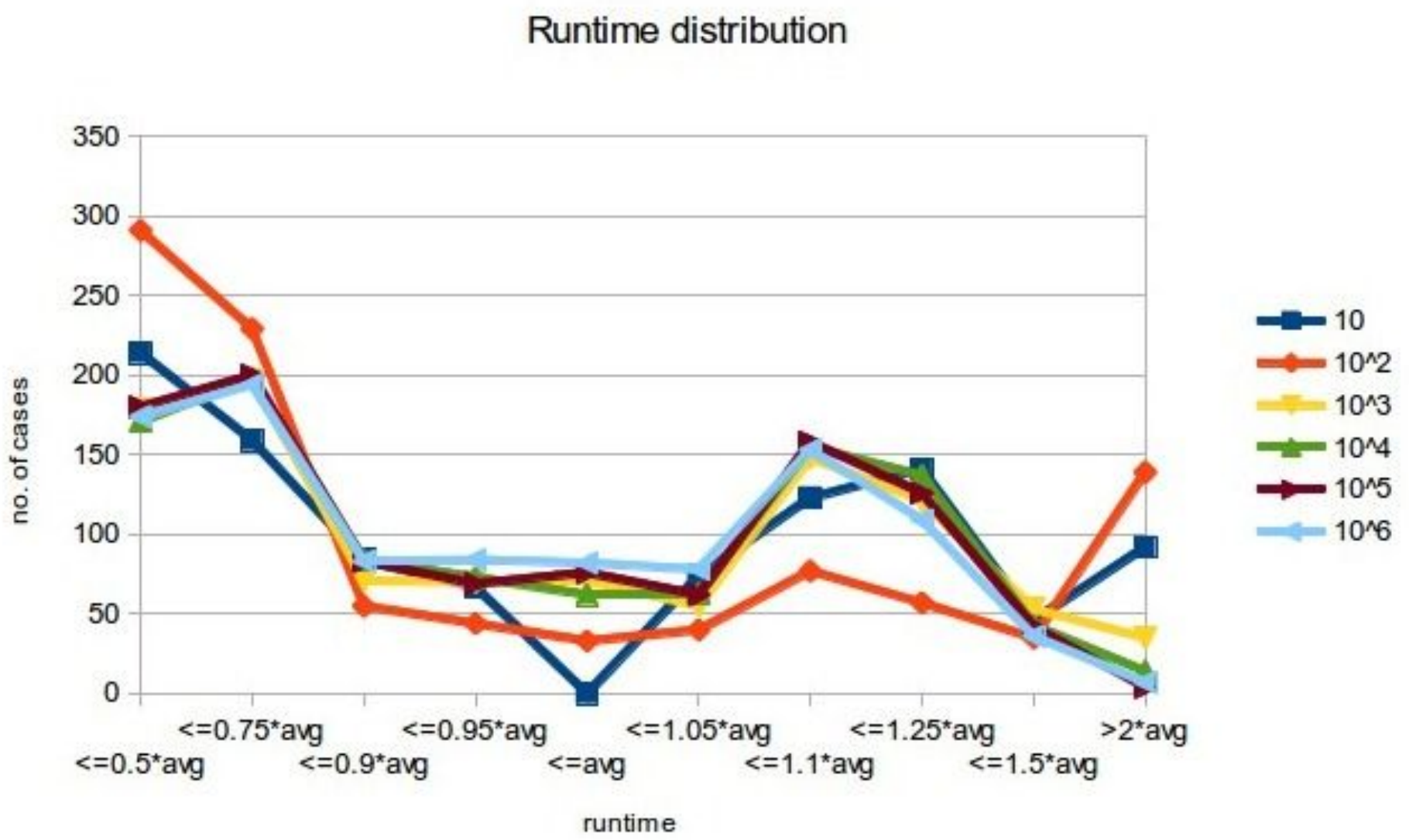
Time Analysis

n(Input Size)	Running Time(μsec)
10	14
10^2	77
10^3	619
10^4	6156
10^5	83488
10^6	1051354
10^7	12889873

Running Time



Experimental Results



When does Randomised Algorithm work?

Randomized incremental construction algorithms of this sort (compute an 'optimal' thing) work if:

- The test whether the next input object violates the current optimum must be possible and fast
- If the next input object violates the current optimum, finding the new optimum must be an easier problem than the general problem
- The thing must already be defined by O(1) of the input objects

Ultimately: the analysis must work out!!

References

A) Xu, S., Freund, R.M. & Sun, J. Solution Methodologies for the Smallest Enclosing Circle Problem. Computational Optimization and Applications 25, 283–292 (2003). https://doi.org/10.1023/A:1022977709811

B) P. Chrystal, “On the problem to construct the minimum circle enclosing n given points in a plane,” in Pro-ceedings of the Edinburgh Mathematical Society, Third Meeting, 1885, p. 30.

C) J.Adarsh, S.Sahai," The Smallest Enclosing Circle Problem", Indian Institute of Technology, Kanpur, November-2013.

D) J. Eliosoff and R. Unger, “Minimal spanning circle of a set of points,” Computer Science 308-507: Computational Geometry Project, School of Computer Science, McGill University, 1998.

E)Jung, H.W.E. "Über die kleinste Kugel, die eine räumliche Figur einschliesst." J. reine angew. Math. 123, 241-257, 1901.