CS648

Smallest Enclosing Circle

Problem

Given a Set of n points on a plain, no 3 of them collinear, find the smallest circle enclosing all the points.

Expected Runtime $= O(n)$

Worst Case Runtime $= O(n^3)$

Incremental Approach

We had initially considered various ways to solve the problem including :

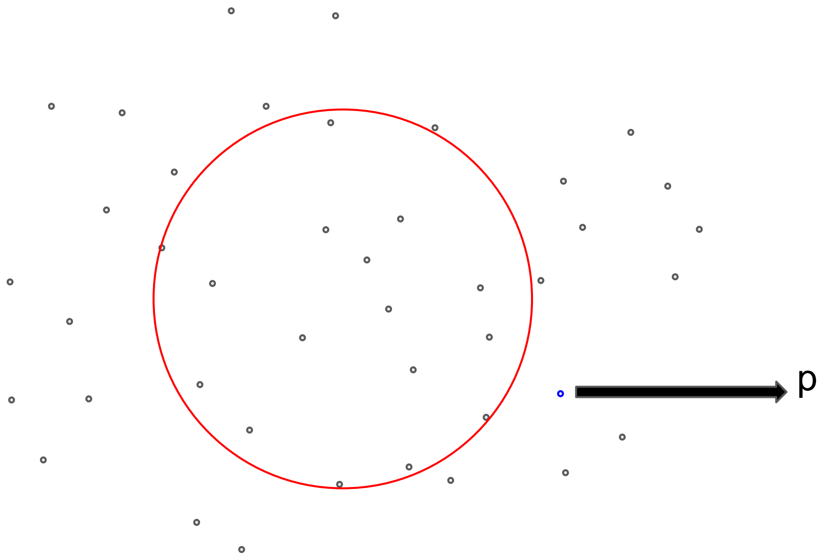
- direct/geometry based
- Decremental approach with a big initial circle
- Incremental approach

But the incremental approach seemed promising as it allowed us to us the information about the points previously processed. Also, the geometrical knowledge required wasn't much.

Incremental Approach

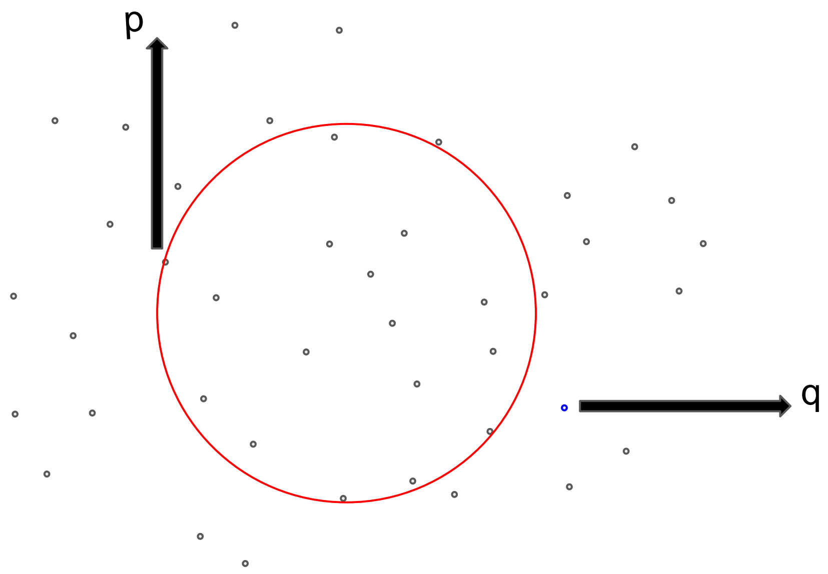
Main()

1. S = set of all points //random permutation
2. p1, q1 = S.pop(), S.pop()
3. C = circle(p1,q1) // as diametric points
4. while S $\neq$ Φ:
5. p = S.pop()
6. if p does not lie inside C:
7. C = Update_1(C,p)
8. else: C $\longleftarrow$ p
9. return C



Update_1(S,p)

```
q = S.pop()
C = circle(p,q) // as diametric points
while S ≠ Φ:
    q = S.pop()
    if q does not lie inside C:
        C = Update_2(C,p,q)
    else: C ⟵ q
return C
```



Update_2(S,p,q)

C = circle(p,q) // as diametric points

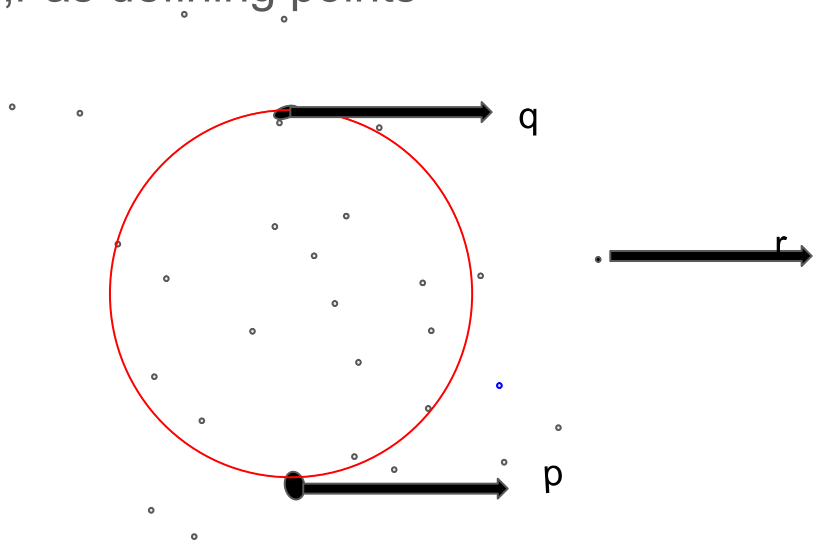
while S ≠ Φ:

    r = S.pop()

    if r does not lie inside C **and** Triangle(p,q,r) is acute:

        C = circle(p,q,r) // circle with p,q,r as defining points

return C



Runtime analysis

Let $\epsilon_i =$ event that i^{th} point changed the circle C (in the main function), i.e, Update_1 function was called.
Now, assuming that circle C_i was made up of 3 defining points, then i^{th} point must be one of them.

$$\therefore \quad P(\epsilon_i) = \frac{3}{i}$$

Similarly, in function Update_1: there are i points (from the main function) in S.
So, ϵ_j is the event that j^{th}point changed the circle, then:

$$\therefore P(\epsilon_j) = \frac{2}{j}$$

Runtime analysis

Let $X_i = 1, \text{if Update\_1 was called for } i^{th} \text{ point}$
$= 0, \text{otherwise}$

And for Update_1:

Let $Y_j = 1, \text{if Update\_2 was called for } j^{th} \text{ point}$ (out of the i points in Update_1)
$= 0, \text{otherwise}$

$\therefore P(X_i = 1) = \frac{3}{i}$ and $P(Y_j = 1) = \frac{2}{j}$

X_i and Y_j are independent random variables, as $P(Y_j | X_i) = P(Y_j)$.

Runtime analysis

Runtime of Update_2 = $O(j)$, for j points being passed to it from Update_1

Runtime of Update_1 = $T(i) = O(i) + \sum_{j=1}^{i} Y_j O(j)$ for i points passed to it.

Runtime of Main = $O(n) + \sum_{i=1}^{n} X_i T(i)$

$$\leq an + \sum_{i=1}^{n} (X_i bi + \sum_{j=1}^{i} X_i Y_j cj)$$

Runtime analysis

∴ Expected Runtime = $E[O(n) + \sum_{i=1}^{n} X_i T(i)]$

$$\leq an + \sum_{i=1}^{n}(E[X_i]bi + \sum_{j=1}^{i} E[X_i Y_j]cj)$$

And, $E[X_i Y_j] = E[X_i]E[Y_j]$ (independent random variables)

$$= (1 \times P(X_i = 1))(1 \times P(Y_j = 1))$$

$$= \frac{6}{ij}$$

Runtime analysis

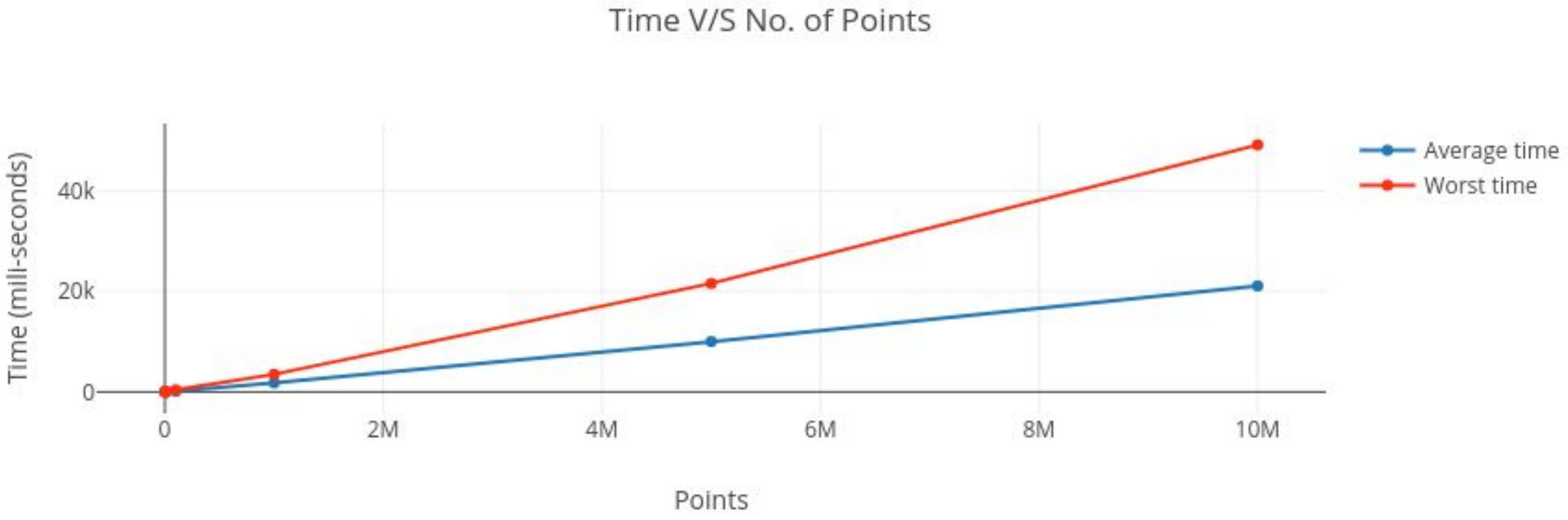
$\therefore$ Expected Runtime $\leq an + \sum_{i=1}^{n} \left(\frac{3}{i} bi + \sum_{j=1}^{i} \frac{6}{ij} cj\right)$

$$\leq an + kn$$

$$= O(n)$$

Runs.

Number of Points	Average time (ms)	Worst case (ms)
1000	0	2
5000	4	14
10000	9	24
100000	130	392
1000000	1800	3499
5000000	10010	21569
10000000	21080	49121


Time V/S No. of Points
Time (mili-seconds)
40k
20k
0
0
2M
4M
6M
8M
10M
Points
Average time
Worst time

	Number of Points (% of runs)				
Average Time Exceeded by %	10000	100000	1000000	5000000	10000000
10%	34.6	35.2	38	40	39
20%	26.7	28.9	30	29	33
50%	9.2	13.9	12	7	6
100%	1.4	2.2	1	0	0

How to prove the correctness.

1. There exists a unique circle passing through 3 non-collinear points.

2. For a new point **p** outside the current smallest enclosing circle, the new smallest enclosing circle must **p** as one of its defining points.

3. This can be used recursively until we have 3 distinct points (or 2 in some cases) as the defining points, thus we have found the unique circle that is the smallest enclosing circle for the set of points under consideration.