Smallest Enclosing Circle Problem

Raghukul,Vibhor

IIT Kanpur

November 12, 2018

Outline

Outline

Problem

- Given n points in a plane, compute the smallest radius circle enclosing all n points.

Problem

- Given n points in a plane, compute the smallest radius circle enclosing all n points.
- The time complexity of best deterministic algorithm for this problem is $O(n)$ which uses advanced geometry.

Problem

- Given n points in a plane, compute the smallest radius circle enclosing all n points.
- The time complexity of best deterministic algorithm for this problem is $O(n)$ which uses advanced geometry.
- We will present a simple randomised algorithm for this problem with expected $O(n)$ time complexity.

Problem

- Given n points in a plane, compute the smallest radius circle enclosing all n points.
- The time complexity of best deterministic algorithm for this problem is $O(n)$ which uses advanced geometry.
- We will present a simple randomised algorithm for this problem with expected $O(n)$ time complexity.

Ideas

- A circle is completely determined if we are given three or two points lying on its boundary.

Ideas

- A circle is completely determined if we are given three or two points lying on its boundary.
- We can use this idea to get a trivial $O(n^4)$ deterministic algorithm for our problem.

Ideas

- A circle is completely determined if we are given three or two points lying on its boundary.
- We can use this idea to get a trivial $O(n^4)$ deterministic algorithm for our problem.
- This algorithm can be easily modified to give a deterministic algorithm with $O(n^3)$ time complexity.

Ideas

- A circle is completely determined if we are given three or two points lying on its boundary.
- We can use this idea to get a trivial $O(n^4)$ deterministic algorithm for our problem.
- This algorithm can be easily modified to give a deterministic algorithm with $O(n^3)$ time complexity.
- The idea is that given two defining points we can find the third defining point in $O(n)$ time.

Ideas

- A circle is completely determined if we are given three or two points lying on its boundary.
- We can use this idea to get a trivial $O(n^4)$ deterministic algorithm for our problem.
- This algorithm can be easily modified to give a deterministic algorithm with $O(n^3)$ time complexity.
- The idea is that given two defining points we can find the third defining point in $O(n)$ time.
- Check whether these two circles(one constructed taking the 2 defining points as diameteric ends) are valid and return the smallest radius circle enclosing these points.

Outline

$O(n^4)$ algorithm

Input: N points in $2D$ plane
Output: Circle enclosing given N points and having minimum radius
$P \leftarrow$ list of N points
$C \leftarrow$ circle formed by taking $P[1], P[2]$ as diameter.
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow i+1$ **to** n **do**
    $C' \leftarrow$ circle formed by taking $P[i], P[j]$ as diametric ends.
    if $C'.rad < C.rad$ **then**
      $C = C'$
  end
end
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow i+1$ **to** n **do**
    for $k \leftarrow j+1$ **to** n **do**
      $C' \leftarrow$ Circumcircle of $P[i], P[j], P[k]$
      if $C'.rad < C.rad$ **then**
        $C = C'$
    end
  end
end
return C

$O(n^3)$ algorithm

Input: N points in $2D$ plane
Output: Circle enclosing given N points and having minimum radius
$P \leftarrow$ list of N points
$C \leftarrow$ circle formed by taking $P[1], P[2]$ as diameter.
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow i+1$ **to** n **do**
    P_{min} = point subtending least acute angle on line $P[i] \rightarrow P[j]$
    P_{max} = point subtending least obtuse angle on line $P[i] \rightarrow P[j]$
    $C_1 \leftarrow$ circle formed by taking $P[i], P[j]$ as diametric ends.
    $C_2 \leftarrow$ circumcircle of $P[i], P[j], P_{min}$.
    $C_3 \leftarrow$ circumcircle of $P[i], P[j], P_{max}$.
    $C' \leftarrow$ circle among C_1, C_2, C_3 having least radius.
    if $C'.rad < C.rad$ **then**
      $C = C'$
  end
end
return C

Expected $O(n)$ algorithm

Input: N points in $2D$ plane
Output: Circle enclosing given N points and having minimum radius
$P \leftarrow$ list of N points
$P^{'} \leftarrow$ random permutation of P
$C \leftarrow$ circle formed by taking $P^{'}[1], P^{'}[2]$ as diameter.
for $i \leftarrow 3$ **to** n **do**
  if $C.outside(P^{'}[i])$ **then**
    $C \leftarrow$ build_circle$(i-1, P[i])$
end
return C

build_circle

Input: k, P_d
Output: Circle enclosing first k points and P_d being on boundary
if $k = 1$ **then**
  return *circle formed by taking* $P[1] \rightarrow P_d$ *as diameter.*
$C' \leftarrow$ build_circle$(k-1, P_d)$
if $C'.inside(P[k])$ **then**
  return C'
else
  `/* Now we know that` P_k `is also defining point */`
  $l \leftarrow$ line joining $P[k], P_d$
  $P_{min} =$ point$([1, k-1])$ subtending least acute angle on l
  $C_1 \leftarrow$ circle formed by taking $P[i], P[j]$ as diametric ends.
  $C_2 \leftarrow$ circumcircle of $P[i], P[j], P_{min}$.
  $C' \leftarrow$ circle among C_1, C_2 having least radius.
  return C'

Lemma 1

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 $S \cup \{P\}$.

Lemma 1

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 $S \cup \{P\}$.
Proof : Note that there has to to at least 2 point on the circle. If not then we can compress the circle, so that there are at least 2 points on the circle.

Lemma 1

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 $S \cup \{P\}$.
Proof : Note that there has to to at least 2 point on the circle. If not then we can compress the circle, so that there are at least 2 points on 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.

Lemma 2

Probability that point P_i lies outside the smallest enclosing circle of points $P_1, ..., P_{i-1}$ is $\leq \frac{3}{i}$.

Lemma 2

Probability that point P_i lies outside the smallest enclosing circle of points $P_1, ..., P_{i-1}$ is $\leq \frac{3}{i}$.

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.

Lemma 2

Probability that point P_i lies outside the smallest enclosing circle of points $P_1, ..., P_{i-1}$ is $\leq \frac{3}{i}$.

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 proability that i^{th} point is defining point for C_i is $\leq \frac{3}{i}$.

Lemma 3

Given two defining points then the third defining point is the point which form the least acute angle with these two points.

Lemma 3

Given two defining points then the third defining point is the point which form the least acute angle with these two points.

Proof : The angle subtended by a chord on the boundary of the circle is least compared to all other points in the same sector and for a point to be the defining point it must lie on the boundary.

Lemma 3

Given two defining points then the third defining point is the point which form the least acute angle with these two points.

Proof : The angle subtended by a chord on the boundary of the circle is least compared to all other points in the same sector and for a point to be the defining point it must lie on the boundary.

So the third defining point(if any) is the point which form the least acute angle with these two points.

Outline

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$.

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$. This occurs when all the points lie outside the existing circle, and while updating circle, removing q does not give a defining point on every step of recursion.

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$. This occurs when all the points lie outside the existing circle, and while updating circle, removing q does not give a defining point on every step of recursion.
- Expected time complexity of our randomized algorithm is $O(n)$.

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$. This occurs when all the points lie outside the existing circle, and while updating circle, removing q does not give a defining point on every step of recursion.
- Expected time complexity of our randomized algorithm is $O(n)$.
- By Lemma 2 probability that $build\_circle(i)$ is called is at most $\frac{3}{i}$.

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$. This occurs when all the points lie outside the existing circle, and while updating circle, removing q does not give a defining point on every step of recursion.
- Expected time complexity of our randomized algorithm is $O(n)$.
- By Lemma 2 probability that $build\_circle(i)$ is called is at most $\frac{3}{i}$.
- By Lemma 2 and the fact that finding the point subtending least acute angle takes $O(i)$ time, expected runtime of $build\_circle(i)$ function is $O(i)$.

Analysis

- Worst case time complexity of our randomized algorithm is $O(n^3)$. This occurs when all the points lie outside the existing circle, and while updating circle, removing q does not give a defining point on every step of recursion.
- Expected time complexity of our randomized algorithm is $O(n)$.
- By Lemma 2 probability that $build\_circle(i)$ is called is at most $\frac{3}{i}$.
- By Lemma 2 and the fact that finding the point subtending least acute angle takes $O(i)$ time, expected runtime of $build\_circle(i)$ function is $O(i)$.
- Thus the expected time complexity of algorithm is $\sum_{i=3}^{n} O(i) * \frac{3}{i} = O(n)$.

Experimental Results

Comparison between run time of the 3 algorithms discussed.

N	Expected $O(n)$ algorithm	$O(n^3)$ Algorithm	$O(n^4)$ Algorithm
10	0.069	0.186	0.248
20	0.093	1.112	3.203
30	0.097	3.395	14.635
40	0.133	7.888	44.322
50	0.146	15.107	105.807
60	0.185	26.222	219.731
70	0.215	42.041	408.725
80	0.229	62.511	695.651
90	0.239	88.957	1109.614
100	0.268	122.576	1700.718

Table 1: All the values given above are in ms. Tested over 50 randomly generated examples for each N.

Experimental Results

Comparison between $O(n^3)$, $O(n)$ algorithms.

N	Expected $O(n)$ algorithm	$O(n^3)$ Algorithm
100	0.269	121.167
200	0.476	959.231
300	0.679	3163.430
400	1.052	7615.576
500	1.175	14999.119
600	1.522	25769.805
700	1.813	43088.605
800	2.008	66676.151
900	2.219	93234.026
1000	2.254	120870.487

Table 2: All the values given above are in ms. Tested over 20 randomly generated examples for each N.

Experimental Results

Statistics of $O(n)$ algorithm.

N	Average Time	Worst Case Time	Standard deviation
10	0.071	0.097	0.011
100	0.303	0.895	0.149
1000	2.224	5.419	0.933
10000	22.307	47.068	8.381
100000	212.129	629.476	96.677
1000000	1786.539	3346.28	find out!!
10000000	find out!!	find out!!	find out!!

Table 3: All the values given above are in ms. Tested over 50 randomly generated examples for each N.

Experimental Results

Number of time new point lies outside current circle.

N	no. of times oint lies outside (Avg value)
10	3.235
100	8.843
1000	15.118
10000	21.059
100000	26.189
1000000	find out!!
1000000	find out!!

Table 4: Tested over 50 randomly generated examples for each N.

Experimental Results

Percentage exceeding Average run time.

Average time exceeded by(%)	10	100	1000	10000	100000	1000000
10	20.332	30.290	35.269	33.195	find out!!	find out!!
20	12.863	22.821	26.556	25.726	find out!!	find out!!
50	1.659	2.905	9.128	11.618	find out!!	find out!!
100	0	1.1	2.489	4.979	find out!!	find out!!

Table 5: Tested over 250 randomly generated examples for each N.