International Journal of Computer & Information Technologies (IJOCIT)
www.ijocit.org & www.ijocit.ir
ISSN = 2345-3877

An Efficient Algorithm for the Shortest Path in the Delaunay Triangulation

Keivan Borna[1], Ahmad Rahnama Zadeh[2]
Department of Mathematical Sciences and Computer[1], Faculty of Computer and Information Technology Engineering[2]
Kharazmi University[1], Azad University Qazvin[2]
Tehran[1], Qazvin[2]
Iran
borna@khu.ac.ir[1], rahnamazade@gmail.com[2]

Abstract: The goal of the current article is to optimize the time complexity to find new points in the Delaunay Triangulation. Adding these points to the triangulated set in the current Delaunay triangulation, the shortest path between two nodes in the new Delaunay triangulation has improved. Regarding the position of these points in the intersection of Delaunay circles, we present a sweep line algorithm to find the intersection points between the Delaunay circles. This algorithm has optimal time complexity using two techniques. First, one can see that intersection of the two circles is not calculated when they are distinct. Second, after finding the intersection points of a circle with other circles, that circle is removed from the algorithm calculations cycle. In conclusion the time complexity of our algorithm equals with the number of intersection of Delaunay circles and is optimal regardless of the additional calculation. Considering to the applications of Delaunay triangulation in geometric routing protocol in Wireless Sensor Networks, new sensors can be placed on the points that calculated by our algorithm and improve the shortest path between two sensors, so increase the data transfer speed in Wireless Sensor Networks.

Keywords: Circle intersection, Delaunay triangulation, Geometric routing, Wireless Sensor Network.

1. Introduction

Ad-hoc networks were used for the first time in 1970 by DARPA[1] for the PRNET[2] project [8]. These networks include a set of distributed nodes which make a temporary network without the support of the central management. Some of its most important usages can be mentioned as military environments, including the soldiers, tanks, airplanes, as well as, the battles with a remote control for the military connections or in non-military environments, such as, taxi-networks, sport fields, agricultural fields, searching and saving operations, helping situations for bad and urgent accidents. One of the benefits of using such networks is the lack of need to the physical structure and the possibility to make changes in their virtual structures. This characteristic has caused the routing protocols of such kind of networks to be attractive [7]. Wireless Sensor Networks are one of the kinds of Ad-hoc Networks. Regarding the geographical distribution of the sensors in this kind of network, the geometric routing protocol, which benefits from the Delaunay triangulation, is one of the routing protocols that always has the shortest distance between the nodes because of the special characteristics of the Delaunay triangulation. Dobkin article [5] assumes that where the total distance travelled by any packet is never more than a small constant factor times the network distance between source and destination. But it has not been considered due to its high time cost. Many researchers have tried to decrease this time cost by offering different algorithms, among them, the presented algorithm for calculating the dynamic Delaunay triangulation for ad-hoc wireless networks by Ming Li [6] and the presented algorithm by Filipe Araújo [2] can be mentioned. Along with optimizing the length of the shortest path in the Delaunay triangulation, Manuel Abellanas introduced a method that could decrease the minimum distance between two nodes to the least possible amount by adding some points to

[1] Defense Advanced Research Projects Agency
[2] Packet Radio Network

the current triangulation [1] and decrease the ratio that presented by Dobkin[5] .In fact, this method could be used for positioning new sensors in the geographical area of a wireless sensor network, so that shortening the length of the route between the sensors for transferring the data we could reach the maximum speed. This algorithm has high time complexity. In this paper we decrease the time to find such points to the optimum amount. Along this, we introduced an algorithm that gains the intersection of the Delaunay circles in an optimal time.

2. Descriptions and Assumptions

2.1. Delaunay Triangulation

In mathematics and computational geometry, a Delaunay triangulation for a set P of points in a plane is a triangulation DT(P) such that no point in P is inside the circumcircle of any triangle in DT(P). Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation; they tend to avoid skinny triangles. It is named after Boris Delaunay for his work on this topic from 1934 [4].

2.2. The Image of Circle on the Coordinate Axis

The image of the circle with the center (a,b) and the radius r, on the x-axis, is considered as equal to the line segment with the ending points with the coordinates (a-r,0) and (a+r,0).

(1) $y = 0, \forall x: a - r \leq x \leq a + r$

ii) The image of the circle with the center (a,b) and the radius r, on the y-axis, is considered as equal to the line segment with the ending points with the coordinates (0,b-r) and (0,b+r). (See Figure 1)

(2) $x = 0, \forall y: b - r \leq y \leq b + r$

2.3. The Vertical Coverage Area of A Circle

The vertical coverage area of a circle with the center (a,b) and the radius r is considered as the equal to the area between two vertical lines with the coordinates x=a-r and, x=a+r. In other words, for all x between a-r and a+r (See Figure 2)

(3) $\forall x, y: a - r \leq x \leq b + r$

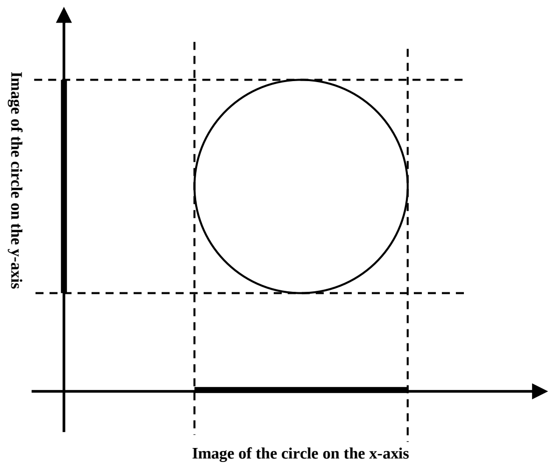
Image of the circle on the y-axis
Image of the circle on the x-axis

Figure 1: Image Of Circle On The Coordinate Axis

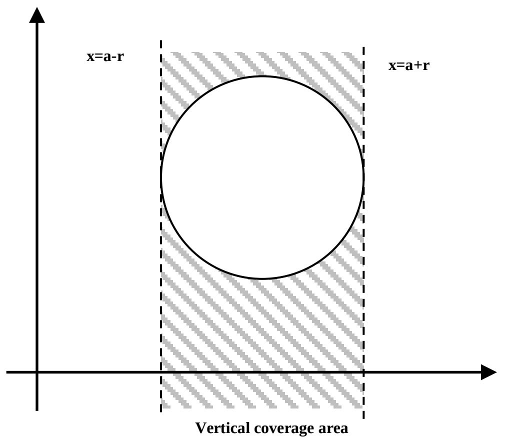

Figure 2 : Vertical Coverage Area Of A Circle

2.4. Descriptions of the Circles on The Plane

In this article, the circles available on the plane express the circles resulted from the Delaunay triangulation of a set of the points. Therefore, considering that in the Delaunay triangulation, for every three vertexes of a triangle one circle passes these three points, the center of this circle is that vertex of the Voronoi diagram. So, two circles never put inside each other, and the assumption of completely crossover circles is not posed in this article.

2. Improving the Shortest Path in the Delaunay Triangulation and Its Usage in Wireless Sensor Network

Consider that the two points of s and t, and the shortest path between the two points, have been given from the Delaunay triangulation. Manuel Abellanas has introduced a way [1] during which, at the beginning, the intersection of all the Delaunay circles on the path of these two points are being calculated, then, some points are added along the path and on the intersection of these circles so that the length of the shortest path between these two points would be improved to the possible amount. To better understand this, consider for each pair of intersecting circles (C_1, C_2), we proceed as follows. Each circle corresponds to a Delaunay triangle. Let the two triangles be t_1, t_2. For each pair of vertices $x \in t_1$ and $y \in t_2$, we first check if $\overline{xy}$ intersects $C_1 \cap C_2$. If it does, we take p as any point on $(xy \cap C_1 \cap C_2)$. Otherwise, we check the two points where C_1 and C_2 intersect, and use the one that gives the shortest path from x to y. (See Figure 3)

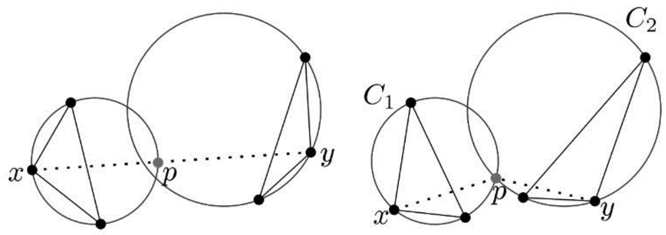

Figure 3: optimal ways to connect x and y

We know that, in fact, this method could be used for positioning new sensors in the geographical area of a wireless sensor network, so that shortening the length of the route between the sensors for transferring the data we could reach the maximum speed. Adding the description that, the time presented by him to find the intersection points of these circles equals to $O(n \log n + I)$, where I is the number of pairs of Delaunay circles that intersect, and in the worst case, it equals to $O(n^2)$. In the sequel, we present an algorithm to decrease this time.

3. Finding Intersection of Delaunay Circles

In this section, first, we study and prove some lemma and theorems. Then, we try to present an algorithm to find the circle intersection in the Delaunay triangulation in an optimal time.

Lemma 1. *Suppose that C_1 and C_2 are two circles on a plane; and their images on the x-axis are S_{xc_1} and S_{xc_2}, and their images on the y-axis are S_{yc_1} and S_{yc_2}. Then, it could be said that C_1 and C_2 are two separated circles, if their images on the x-axis and their images on the y-axis has no intersection with each other.*

Proof: Considering that the images of two circles on the y-axis have no junctions with each other, if "a" is the distance between the images of the center of the two circles on the y-axis and "b" is the distance between the images of the center of the two circles on the x-axis, then:

(4) $$a > r_1 + r_2$$

(5) $b > r_1 + r_2$

Now, if the Euclidean distance between the centers of the two circles is shown with “c”, regarding the Pythagorean Theorem on the right triangle, we would have:

(6) $c = \sqrt{a^2 + b^2}$

(7) $c = \sqrt{2}(r_1 + r_2) > r_1 + r_2$

We can see that the distance between the centers of the two circles is larger than the sum of the radii of the two circles; therefore, the circles are not crossover. (See Figure 4)

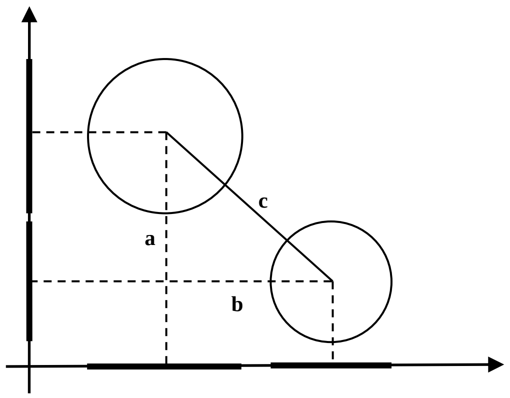

Figure 4: Show Two Separate Circle on Coordinate Axis

Lemma 2. *Suppose that c_1 and c_2 are two circles on a plane that their coverage areas have no common point with each other; then, the two circles are not crossover.*

Proof: Suppose the lines l_1 and l_2 as two parallel lines of the y-axis passing from the centers of the circles c_1 and c_2. Regarding the hypothesis, it is clear that the distance between the two given lines is more than the sum of the radii of the two circles. Thus, in the most optimistic situation, only when their centers have the same y, then the two circles c_1 and c_2 are near each other; and in this situation, the two circles are not crossed-over again, considering the distance between the lines l_1 and l_2, which is more than the sum of the radii of the two circles.

Now using Lemma 1 and 2, we pose the following theorem:

Theorem 3. *Suppose that c_1 and c_2 are two circles on the plane with common coverage areas. Then, c_1 and c_2 are two non-crossover circles if their images have no crossover on the y-axis.*

Proof: We use the method described in Lemma 1 to prove it. In this situation, the difference between the widths of the canters of the two circles is larger than the sum of the radii of the two circles, regarding that their images have no intersection on the y-axis. We consider that, regarding the unity of the coverage areas, the difference between the canters of the two circles is less than the sum of the radii of the two circles, which could be supposed as zero in the best situation. Now, considering the Pythagorean Theorem for the right triangle:

(8) $$c = \sqrt{a^2 + b^2} = \sqrt{0 + b^2} = b > r_1 + r_2$$

Thus, the distance between the canters of the two circles is more than the sum of the radii of the two circles and the circles are not crossover.

Now, regarding the mentioned theorems, we present an algorithm to find the intersection points between the Delaunay circles which are sensitive to the output, and they do not regard separated circles while the plane is scanned, and they do not have any sensitivity to calculate the crossover. Therefore, we optimize the finding time of the crossover circles. As well, in the presented algorithm, because of the lack of calculation of the crossovers between all the paired circles, we improve the time a little in the worst situation through using the sweep line.

Consider that, in order to improve the shortest path in the Delaunay triangulation, we must find the intersection between all the paired circles of the Delaunay triangulation in the path between t and s; and the time complexity to find a point with the best improvement in the shortest path was calculated equal to $O(n \log n + I)$, that "I" expresses the number of the intersection points between the circles; which, in the worst case, is equal to all the binary compositions of the circles and it is equal to $O(n^2)$; [1]. This time was calculated considering the algorithm method to find the line intersection by Balaban [3].

4. Algorithm

First, we encircle two vertical line segments for each circle with the center (a,b) and the radius r; one on the left side of the circle with the line equation of $x = a - r$ and the coordinates of the ending point equal to (a-r,b-r) and (a-r,b+r), and the other on the right side of the circle with the line equation of $x = a + r$ and the coordinates of the ending point equal to (a+r,b-r) and (a+r,b+r). (See Figure 5)

The resulted line segments would be named regarding the encircled circle and the matter that whether the line segment is on the right side of the circle or on the left, so that, considering the name of any line segment we know it belongs to which circle and that it is on which side of that.

Now, using the vertical sweep line, we start from the most left side line segment and we move on to the right to scan all the line segments; and we present a list of all intersections of the crossover circles in the end. Suppose that we have used a DCEL (doubly connected edge list) named D.

The rules related to scanning the line segments and the operations needed to be done after reaching any line segment are as follows:

It is also needed to remove the line segment on the left side of the circle from the list D, if the scanned line segment is the line segment on the right side of the circle.

If the scanned line segment is the line segment on the left side of the circle, at the beginning we save it on the list, then we move towards the beginning of the list and we introduce any line segment overlapping the mentioned line segment related to the y-axis (the images of the circles were intersected on the y-axis) in the list, and we try to find the intersection points using the equations of two circles.

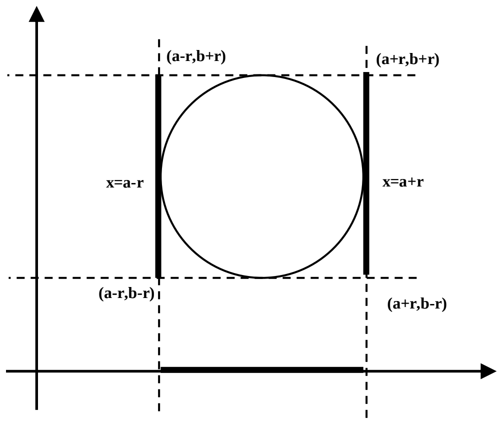

Figure 5: Vertical Line Segment On The Circle Surrounded

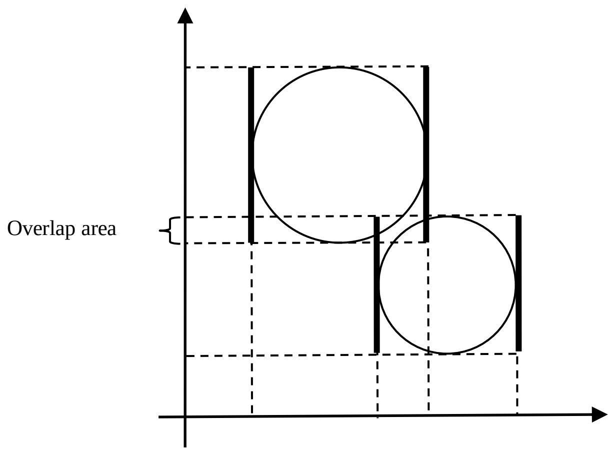

Figure 6: Degenerated Case

In the case of simultaneous reaching of the sweep line to two line segments – we are aware that in this case, the two line segments are necessarily related to two different circles, because we do not have any circle with the radius zero in the Delaunay circles – three conditions may happen:

a. Both of the line segments are on the left side of their circles; in this case, we choose one of them randomly;

b. Both of the line segments are on the right side of their circles; in this case, we choose one of them randomly;

c. One of the line segments is on the right and the other is on the left; in this case, the priority belongs to the line segment which is encircled on the left side of the circle.

The algorithm can now be described as follows:

```
Algorithm FIND-DELAUNAYCIRCLESINTERSECTIONS(S)
Input. A set S of vertical line segments on a Delaunay circle surrounded.
Output. The set of intersection points among the Delaunay circles.
Initialize an empty event queue Q. Next, insert name of segments that surrounded on a Delaunay circle (circle name and left inside or right inside) into Q.
Initialize an empty status structure D.
While Q is not empty do
    Determine the next event segment s in Q and delete it.
    HANDLEEVENT(s)
End

HANDLEEVENT(s)
If (s) is the right side segment then
    Remove Segment corresponding to the left on D
Else
    Insert (s) to D
    While not (Reached the top of the list D)
```

```
                If (s) and D overlap on y-axis
                    Find intersection between (s) and element of D
                    Go to Prev element of D
                Else
                    Go to Prev element of D
                End
        End
End
```

5. Algorithm Evaluation

5.1. Studying The Algorithm Correctitude

Considering the steps of the algorithm, we can see Intersection of the two circles is calculated only when the area is covered by overlapping them (Lemma 2), also, their images on the y-axis (Lemma 1) are intersected. Thus, the states in which the two circles are separated are being removed (Theorem 3), and we just pass along the separated (non-overlapped) circles, and it can be done in $O(I)$. In fact, sweep line moves from left line segment to right line segment of a circle and all lines that belong to the other circles that be seen, we calculate the intersection if line segment overlapping the mentioned left line segment related to the y-axis. When we calculate the intersection between two circles such as x and y, not necessary to calculate intersection between y and x so reduced the half of the calculation.

5.2. Degenerate Case

Maybe the two circles have unity in the vertical coverage area (a circle starts before the other one is passed), as well, their images on the y-axis have unity but not crossed-over. In such cases, considering the calculation of the intersection points and the lack of production of the output in this condition, there is no fault for the produced output, and we would have just intersection points of the crossed-over circles. (See Figure 6)

5.3. Time Complexity Analysis

The time complexity of the algorithm in normal state is being calculated as $O(I)$, where "I" stands for the number of intersection points between the circles; which, in the worst case, that all the circles are overlapped and no element is removed from the list, the algorithm is forced to move on the list D to the beginning. To do the related operation for any scanning, it is clear that in this state while scanning, the i'th line segment of the list has i-1 elements, and totally, the number of all the reviews would be $n(n-1)/2$, and the time complexity in this state would be $O(1/2(n^2))$. In the best case it would be equal to number of Delaunay triangulation. Thus

(9) $\Omega(I) \le \theta(I) \le O(n^2)$,

where I is the number of intersections between the circles.

6. Conclusions

Regarding the application of the Delaunay triangulation in the geometric routing protocol of the Wireless Sensor Networks, in this article, we have tried to decrease the finding time of a point to improve the shortest path in the Delaunay triangulation to the possible amount. To this end, an algorithm has been presented to find the intersection points of the Delaunay circles using the sweep line, so that, by passing from a circle, its intersection with other circles is being calculated; and if two circles are separated and do not have any intersection point, the algorithm doesn't spend any time to find the intersection point between them and just skip them. Therefore, the finding time of the crossover points of the Delaunay circles on a path from a node like "s" is decreased to "t", so that, if the number of the intersection of the circles is "I", its time complexity would be $O(I)$, and in the worst case, it would be equal to $O(1/2(n^2))$; regarding the coefficient of ½, it has improved slightly.

References

[1] Abellanas, M., et al., "Improving shortest paths in the Delaunay triangulation". International Journal of Computational Geometry & Applications, Vol. 22, No. 6, 2012, pp: 559-576.

[2] Araújo, F. and L. Rodrigues, "Fast localized delaunay triangulation", in Principles of Distributed Systems. 2005, Springer. p. 81-93.

[3] Balaban, I.J. "An optimal algorithm for finding segments intersections". in Proceedings of the eleventh annual symposium on Computational geometry. 1995. ACM.

[4] Delaunay, B., "Sur la sphère vide", Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk, 7:793–800, 1934

[5] Dobkin, D. P. , Friedman S. J., and Supowit K. J., "Delaunay graphs are almost as good as

complete graphs", Discrete Comput. Geom., 5, pp. 399–407, 1990

[6] Li, M., Lu, X., and Peng, W. "Dynamic delaunay triangulation for wireless ad hoc network", in Advanced Parallel Processing Technologies. 2005, Springer, pp. 382-389.

[7] Rajaraman, R., "Topology control and routing in ad hoc networks: a survey", ACM SIGACT News, 2002. 33(2), pp. 60-73.

[8] YOSHIDA, M., "Study on Routing Protocols for Vehicle Ad-Hoc Networks", 2007

Authors Profile:

Keivan Borna

He is an Assistant Professor in the Department of Computer Science at Faculty of Mathematics and Computer Science of Kharazmi University of Tehran since 2008. He completed his Ph.D. in September 2008 from the Department of Mathematics, Statistics and Computer Science of University of Tehran in Computational Commutative Algebra. He was a visiting scholar in ``Dipartemento di Matematicha, Universita' di Genova- Italia" and ``Department of Mathematik and Informatik at Essen University, Germany'', from Sep. 2007 to Apr. 2008 during his graduate work. Previously, he received his M.Sc. at Department of Mathematics, Statistic and Computer Science at University of Tehran in 2004. He is very interested in studying and researching in interdisciplinary topics like ``Game Theory'', ``Cryptography'', ``3-sum problem and motion planning'', ``Approximate Algorithms'', ``Persian strings sorting'' and ``Computational Geometry''. He wrote some papers in such areas. He is the author of the ``Advanced Programming in JAVA'' (in Persian) and is the member of ``Elite National Foundation of Iran''\.

Ahmad Rahnama Zadeh

He is currently pursuing his master of Computer Engineering, Software Engineering, at Faculty of Electrical and Computer Engineering in Qazvin Islamic Azad University. He already received his B.Sc. of Computer Engineering, Hardware Engineering, at Islamic Azad University of Qazvin, Iran, working on Design & implementation of a Control System based on RS-232.