Journal of Applied Statistics & Machine Learning
Volume 2; Number 1; 2023 : pp. 67-77

ALGORITHMS FOR ESTIMATING SERVICE REGIONS

Dara Nyknahad[1], Laxmi Gewali[1], Wolfgang Bein[1], Rojin Aslani[2] and Ashok Singh[3]

[1]*Department of Computer Science, University of Nevada Las Vegas, USA.*
E-mail: nyknahad@unlv.nevada.edu[2]; E-mail: laxmi.gewali@unlv.edu; Email: wolfgang.bein@unlv.edu
[2]*Department of Electrical and Computer Engineering, University of Nevada Las Vegas, USA*
E-mail: aslani@unlv.nevada.edu
[3]*Department of Hospitality, University of Nevada Las Vegas, USA*
E-mail: ashok.singh@unlv.edu

ARTICLEINFO

Received: 08 February 2022
Revised: 02 March 2023
Accepted: 06 March 2023
Online: 10 March 2023

To cite this paper:
Dara Nyknahad, Laxmi Gewali, Wolfgang Bein, Rojin Aslani & Ashok Singh (2023). Algorithms for Estimating Service Regions. *Journal of Applied Statistics & Machine Learning*. 2(1): pp. 67-77.

ABSTRACT

Algorithms for computing service areas in the presence of facility stations in two dimensions has been considered by investigators in various field of science and engineering that include operations research, computer science, and transportation network. We present a novel approximation algorithm for estimating service areas when facility stations are of varying capacities, distributed in a city surface environment. Our algorithm is based on modeling facility stations of various capacities by weighted Voronoi diagram. We introduce the idea of inserting virtual stations in the proximity of candidate station to convert an instance of weighted Voronoi diagram to an instance of the standard Voronoi diagram. Such a transformation makes it possible to estimate service areas of high-capacity sites by making use of the standard Voronoi diagram. The approximation algorithm runs in O(n log n) time, where n is the number of service stations.

Keywords: Service area estimation, Weighted Voronoi diagram, Approximation algorithms, facility locations

1. INTRODUCTION

Algorithms for locating service facilities such as healthcare systems, waste management sites, warehouse centers, and battery service/exchange sites have been investigated by researchers in operations research, computational geometry, geographic information system, and traffic management. The facilities are accessed by households and patrons distributed around urban/sub-urban areas. The objective is to reduce the average distance travelled by the customers that the facility is intended to serve. In the simplest version of the facility location problem, the input is given as a set of points distributed over the plane. For such an input, it is required to locate the position of the service center so that the average distance for the customers to reach the facility is minimized. This problem can be stated in term of the **smallest enclosing circle** problem: Given n point sites $p_1, p_2, p_3, \dots p_n$, in the plane, find the smallest circles that encloses all point sites. The smallest enclosing circle problem can be solved in straightforward manner by checking the circles formed by picking three-point sites from the input. The resulting algorithm is rather slow. A faster algorithm of time complexity $O(n \log n)$ is to use the idea of the furthest point Voronoi diagram [2].

Facility location problems are also formulated in term of a graph problem. In this version the input is a graph $G(V,E)$ and it is required to find the vertex that gives the weighted median. In generalized versions of the facility location problem the input is available as (i) the location of customers and (ii) integer k that represents the number of facilities that need to be located optimally. Most versions of the generalized facility location problems are like set covering problems and are known to be NP-Hard [3]. Voronoi diagram have been used to solve the simple versions of facility location problems.

The paper is organized as follows. In Section 2, we review the application of Voronoi diagram for capturing/solving facility location problems. In Section 3, we present an approximation algorithm for estimating service regions when facility stations have various capacities. The algorithm is obtained by converting an instance of weighted Voronoi diagram to an instance of the standard Voronoi diagram. The key idea of the algorithm is the insertion of virtual sites around a non-unit weight site. Finally, in Section 4, we discuss possible extensions and generalization of the proposed algorithm.

2. REVIEW OF VORONOI DIAGRAM AND GENERALIZATIONS

Voronoi diagrams have been extensively used on problems like facility locations, clustering, and high clearance path planning. Formally, Voronoi diagram is defined for a given set of point sites in two dimensions. More specifically, given a set of point sites $S = \{p_1, p_2, p_3, \dots p_n\}$, in the plane, the Voronoi diagram formed by S partitions the plane into n convex regions $R = \{r_1, r_2, r_3, \dots r_n\}$. Each point site p_i is contained inside Voronoi region r_i such that any point in r_i is closer to p_i that to any other site. An example of the Voronoi diagram of point sites is illustrated in Figure 1.

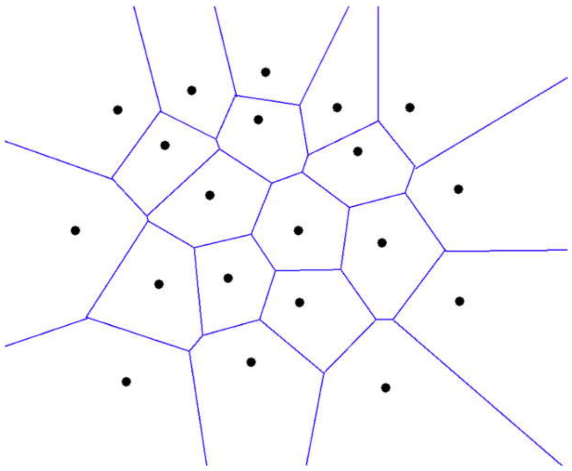

Figure 1: Voronoi diagram of point sites

It is noted that the Voronoi regions could be finite or infinite. In Figure 1 there are nine infinite regions and eleven finite regions. This structure is very useful in computing proximity properties of points distributed in the plane. Detail of such properties can be found on computational geometry literatures [2].

Service facility locations (such as fire station sites) in a city area can be conveniently modeled by Voronoi diagram: the locations of stations can be taken as point sites and the Voronoi region corresponding to each site is the service region for the corresponding station. This model is valid if all stations have identical service capacity. Furthermore, the path between the service station and the destination point, implied by the Voronoi diagram, need not be valid due to the orthogonal structures of the road network in a city environment. What is need is a model in which facility stations have different capacities and the boundary of the Voronoi region follows the boundary of the city road network.

Consider a set of point weighted point sites $S = \{p_1, p_2, p_3, \ldots p_n\}$. The weight of each point site p_i is denoted by w_i. The Voronoi region corresponding to a weighted point site is defined in such a way that a point site with a larger weight has a larger area of influence compared to the area of influence of a point site with smaller weight. For a formal definition of the weight Voronoi diagram, the notion of the weighted distance between two points x and p_i is useful as demonstrated in [1]. Specifically, the weighted distance between point x and a point p_i of weight w_i, denoted by $d_w(x, p_i)$ can be written as:

$$d_w(x, p_i) = d_e(x, p_i)/w_i \tag{1}$$

where $d_e(x, p_i)$ denotes the Euclidean distance between points x and p_i.

The region of influence of point site p_i denoted by $region(p_i)$ is defined as follows.

$$region(p_i) = \{\, x \mid d_w(x, p_i) <= d_w(x, q) \text{ for } \forall\ q \in S \,\} \tag{2}$$

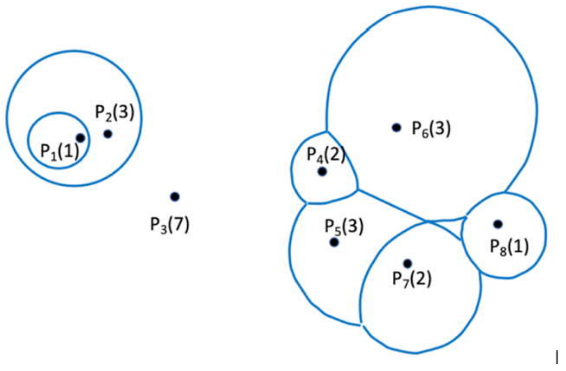

Figure 2: Illustrating weighted Voronoi diagram

The Voronoi diagrams of n equal point sites in two dimensions are planar graph with n bounded or unbounded convex faces and a linear number of edges. These properties are not necessarily valid for a weighted Voronoi diagram which are elaborated in [1]. An example of weighted Voronoi diagram is shown Figure 2. To get a further insight into the structure of a weighted Voronoi Diagram, we examine the edges and face in Figure 2.

(a) Whereas the edges of the standard Voronoi diagrams are straight line segments, the edges of the weighted Voronoi diagram could be both straight line segments and circular arcs.

(b) For the standard Voronoi diagram, the number of Voronoi edges is linear in the number of sites. This is not necessarily true for the weighted Voronoi diagram as shown in [1]; there exists a weighted point site distribution such that the number of regions is quadratic in the number of point sites.

(c) While the regions of the standard Voronoi diagram are connected, the regions of the weighted Voronoi diagram can be disconnected. For example, the region of influence of point site p_3 in Figure 2 is not connected.

3. APPROXIMATION ALGORITHM FOR WEIGHTED VORONOI DIAGRAM

We propose an approximation method for estimating the regions for weighted point sites by carefully inserting **virtual sites**. Virtual sites are inserted to convert an instance of Weighted Voronoi Diagram (WVD) problem to an instance of the Standard Voronoi Diagram (SVD) problem. The regions of the SVD, corresponding to virtual sites, are evaluated to determine if they can be merged to make larger regions. To clarify the scheme of the placement of virtual nodes, and the process of region-merging, we start with an instance of the weighted Voronoi diagram, shown in Figure 3. This figure shows the distribution of several point sites with four different weights. While the sites colored black have weight 1, the colored ones (red, green, and magenta) have weights 2, 3, and 4, respectively. If we treat all sites to have the same weight, the resulting Voronoi diagram will be as shown in Figure 4. The region corresponding to a higher weight site needs to be expanded to reflect its servicing capacity.

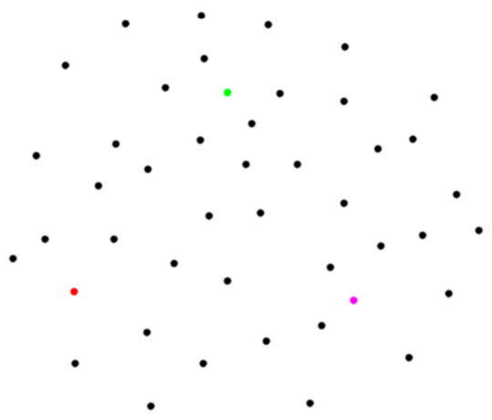

Figure 3: A instance of four different weighted sites

Generation and placement of virtual sites

The size of the region corresponding site p_i should depend on its weight w_i, higher the weights larger the region. Each site can be abstractly viewed to have certain circle of influence. The sites with larger weights will have larger circle of influence than those for smaller weights. To estimate the coverage area of a site with higher capacity, virtual sites are placed on its circle of influence. The number of virtual sites depends on the desired level of accuracy and is not necessarily proportional to the weight of the

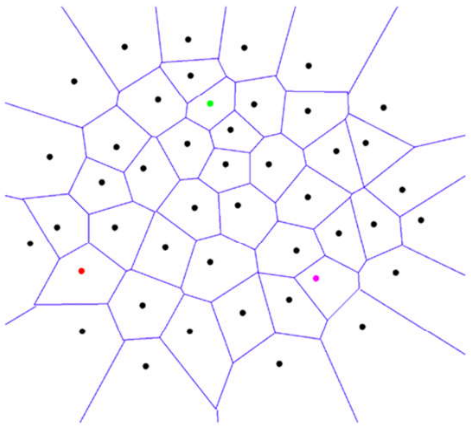

Figure 4: The standard Voronoi diagram of weighted sites

site. Virtual sites are only created for sites that have weight greater than 1. We refer to these sites as **non-unit** weight sites. For each non-unit weight site p_i with weight w_i, k virtual sites are created, where k is the average size of Voronoi regions in a standard Voronoi diagram. It has been established by computational geometry investigators [2] that the average values of Voronoi region cannot exceed 6. The minimum size of a Voronoi region is obviously 3 – as triangle is the polygon with smallest number of edges. Hence a reasonable number of virtual nodes corresponding to a non-unit site is taken between 3 and 8, inclusive.

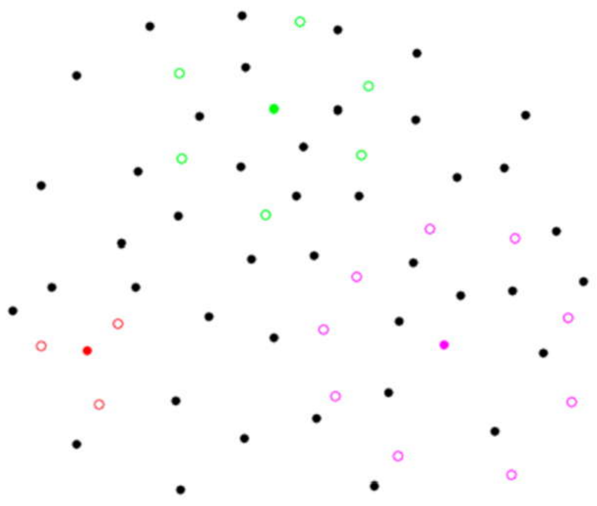

Figure 5: Illustrating the placement of virtual sites

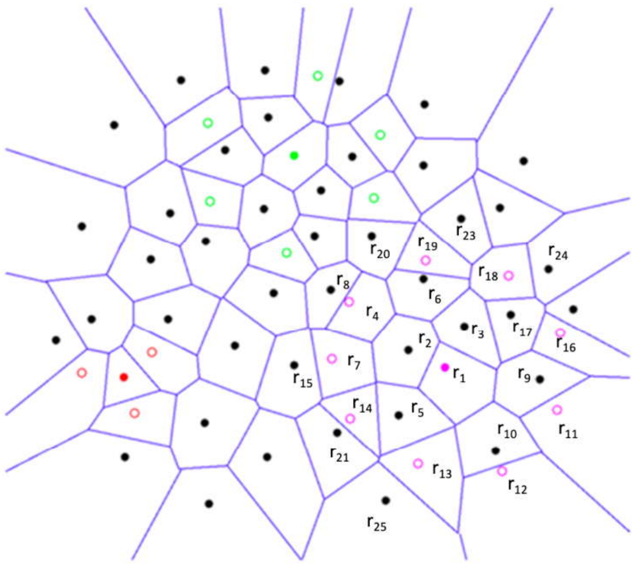

Figure 6: Illustrating Voronoi diagram when virtual sites are added

To place the virtual sites, we first determine a threshold distance ä, which depends on how the input sites are distributed on the plane. For very compactly distributed sites, ä will be smaller than for sparsely distributed sites. One approach to pick the value of ä is to use the value of the Euclidean distance between the closest pair of input sites. Virtual sites corresponding to site p_i with non-unit weight w_i are then placed around the circle (centered at p_i) of radius $äw_i$.

Figure 5 shows an example of the placement of virtual sites around non-unit weight sites. Virtual sites corresponding to each non-unit weight sites (red, green, and magenta) are drawn as rings with the same color. There are three virtual sites for red site, six virtual sites for green site and eight virtual sites for magenta site.

The standard Voronoi diagram of the aggregate of original and virtual sites of Figure 5 is shown in Figure 6, where some regions are labelled as r_1, r_2, r_3, … r_{25}, for later explanation of some terms.

The regions around virtual nodes need to be merged and make larger to reflect the high capacity of the corresponding high-capacity colored site. The regions in the proximity of high-capacity colored site can be distinguished into three kinds: A **type-1** region is the region corresponding to the filled colored site; **type-2** regions are the regions corresponding to unfilled colored sites (virtual sites); and **type-3** regions are the regions of black sites (non-unit sites) surrounded by more than one type-1 or type-2 regions. Referring to Figure 6, region r_1 is a type-1 region. There are eight type-2 regions corresponding to magenta-colored nodes. Region r_3 is a type-3 region as it is surrounded by two type-2 regions r_1 and r_{18}. Similarly, r_2, r_3, r_5, etc. are type-2 regions. Region r_{24} is not surrounded by 2 or more type-1 or type-2 sites and hence it is not a type-3 node.

After determining the types of a site's regions, the next step is to merge them by using the following rules.

Merging rules

Rule 1: Adjacent type-2 regions of the same color are merged.

Rule 2: A type-3 region is merged with an adjacent type-2 region.

Rule 3: The merged aggregation of type-2 regions is merged with the type-1 region.

The result of merging the regions corresponding to magenta-colored site are shown in Figure 7.

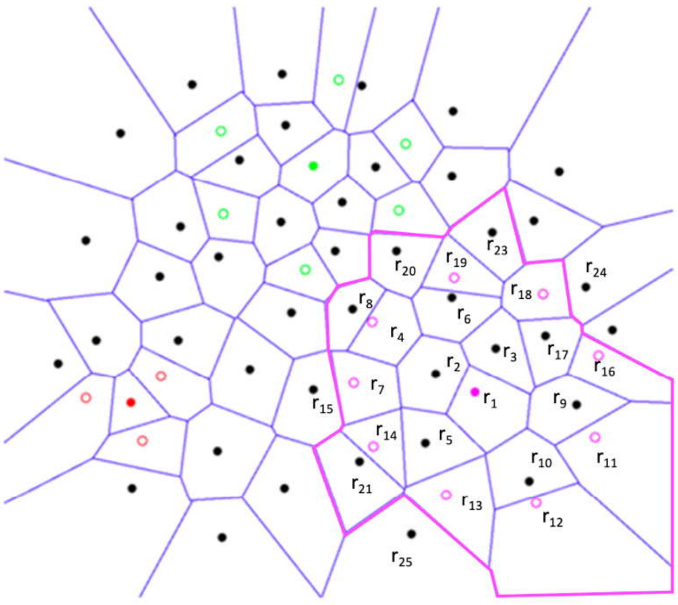

Figure 7: The result of merging regions for magenta site

A formal sketch of the algorithm for estimation the service region of a non-unit site is listed below (Algorithm 1)

```
Algorithm 1: Service Area Estimation Algorithm
Input: A set of point sites $p_1, p_2, \cdots, p_n$ with weights $w_1, w_2, \cdots, w_n$, respectively
Output: The coverage area for each point site $p_i$
  Step 1: Find the closest pair of point sites $p_i$ and $p_j$ among the input sites. Use
          the Euclidean distance between the closest pair as the threshold distance $\delta$.
  Step 2: For each non-unit point site $p_i$ of weight $w_i$, construct $k$ virtual sites
          where $3 \leq k \leq 8$. Locate $k$ virtual sites uniformly on the circle of radius $\delta w_i$
          centered at the location of site $p_i$.
  Step 3: Compute the standard Voronoi diagram of sites consisting of the input
          sites and virtual sites. Represent the virtual diagram in a doubly connected edge
          list data structure. Let $VD'$ denote the resulting Voronoi diagram.
  Step 4:
      for each region $r_i$ corresponding to non-unit site $p_i$ do
          for each adjacent region $r_j$ do
              if the neighboring unit region $r_j$ has more than one common edge with $r_i$ then
                  merge $r_j$ into $r_i$
              end if
          end for
      end for
```

The time complexity of Algorithm 1 can be analyzed as follows.

(a) Finding the closest pair: The closest pair in a set of n input point sites in the Euclidean plane can be found in O(n log n) time using a divide-and-conquer algorithm like the one used in merge sort [2]. Hence Step 1 takes O(n log n) time.

(b) Generating virtual sites: For each non-unit weight site, k virtual sites are added. The value of k can be between 3 and 8 inclusive. For n input sites, the number of virtual sites added is between 3n and 8n, which is O(n). Hence Step 2 takes O(n) time.

(c) Computing Voronoi diagram: Computing the Voronoi diagram of m points takes O(m log m) time using standard algorithms such as Fortune's algorithm [2]. In the proposed algorithm, we first generate the virtual sites and then compute the Voronoi diagram VD^2 of the appended sites, which can be represented in a doubly connected edge list (DCEL) data structure [2]. Therefore, the total time complexity for computing VD^2 is $O(n \log n)$. Hence Step 3 takes $O(n \log n)$ time.

(d) Merging of regions: We can use DCEL structure to navigate over all non-unit point sites and their adjacent regions, checking if they share more than one common edge, and merging them if they do. This takes $O(n)$ time. Hence Step 4 takes $O(n)$ time. To find the total time complexity, we can add up the time complexities of the three steps. Since the time complexity of Step 1 and Step 2 dominate over the other two steps, we can ignore the lower order terms and write the total time complexity as $O(n \log n)$. Therefore, the total time complexity of the proposed method is $O(n \log n)$. Hence, we have the following theorem.

Theorem 1: The coverage area estimation algorithm (Algorithm 1) can be executed in $O(n \log n)$ time.

4. DISCUSSIONS

We have examined the application of Voronoi diagram for finding the service regions of facility stations located in two-dimensional plane. When all service stations have identical capacity the standard Voronoi diagram is quite effective in capturing service regions. However, when the service stations have different capacities the standard Voronoi diagram fails to properly model the service regions. Weighted Voronoi diagram cannot be applied directly due to the fact that the service regions could be disconnected. Furthermore, algorithms for computing the weighted Voronoi diagram are very complicated and difficult to implement. We have proposed here an approximate algorithm for estimating service region by converting an instance of weighted Voronoi diagram to an instance of the standard Voronoi diagram. The key ingredient of the conversion is the insertion if virtual sites near high-capacity sites. The time complexity of the algorithm is quite fast i.e., $O(n \log n)$. There could be several improvements and extensions of the proposed approximation algorithm. We have not been able to obtain the bound of the approximation and it would be desirable to find a value that bounds the quality of the result obtained by the proposed algorithm. In the proposed algorithm we only consider the merging of regions corresponding to colored site and black site. An important issue is when the region of a colored site (say magenta) is surrounded by regions of another colored site (say red). A logical approach would be merge them and make the merged region as the region of higher capacity site. Detail about this process is worth investigating. Application of the proposed algorithm for estimating service areas of battery charging stations has been done in [5]. It would be interesting to find other similar application avenues.

Reference

[1] F. Aurenhammer and H. Edelsbrunner, (1984) An Algorithm for Constructing the Weighted Voronoi Diagram in the Plane, *Pattern Recognition*, Vol. 17, No 2, pp. 251-257.

[2] De Berg M., Otfried Cheong, Mark van Kreveld, and Mark Overmars. (2008) *Computational Geometry: Algorithms and Applications*, third edition, 2008.

[3] Ferran Hurtado, Vera Sacristan and Godfried T. Toussaint, (2000) Some constrained minimax and maximin location problems, Studies in Locational Analysis, *Special Issue on Computational Geometry in Locational Analysis*, Guest Editor: Juan Mesa, Issue No 15, pp 17-35.

[4] Dara Nyknahad, PhD Dissertation, (2023) University of Nevada, Las Vegas, USA

[5] Dara Nyknahad, Laxmi Gewali, Wolfgang Bein, Rosin Aslani, and Ashok Singh, Estimating the Service Regions for Battery Exchange Stations, to appear in the *proceedings of ICSEng*.

[6] Zanjirani Reza, Narnin Asgari, Nooshin Heidari, Mehtab Hosseininia, and Mark Goh, (2012) Covering problems in facility locations: a review, *Computers and Industrial Engineering* 62, pp. 368-407.