Multiple Resource Network Voronoi Diagram

Ahmad Qutbuddin
Department of Computer and Electrical Engineering and Computer Science,
Florida Atlantic University, Boca Raton, FL, USA
aqutbuddin2017@fau.edu

KwangSoo Yang
Department of Computer and Electrical Engineering and Computer Science,
Florida Atlantic University, Boca Raton, FL, USA
http://faculty.eng.fau.edu/yangk/home/
yangk@fau.edu

Abstract

Given a spatial network and a set of service center nodes from k different resource types, a Multiple Resource-Network Voronoi Diagram (MRNVD) partitions the spatial network into a set of Service Areas that can minimize the total cycle distances of graph-nodes to allotted k service center nodes with different resource types. The MRNVD problem is important for critical societal applications such as assigning essential survival supplies (e.g., food, water, gas, and medical assistance) to residents impacted by man-made or natural disasters. The MRNVD problem is NP-hard; it is computationally challenging due to the large size of the transportation network. Previous work is limited to a single or two different types of service centers, but cannot be generalized to deal with k different resource types. We propose a novel approach for MRNVD that can efficiently identify the best routes to obtain the k different resources. Experiments and a case study using real-world datasets demonstrate that the proposed approach creates MRNVD and significantly reduces the computational cost.

2012 ACM Subject Classification Information systems → Geographic information systems

Keywords and phrases Network Voronoi Diagram, Resource Allocation, Route Optimization

Digital Object Identifier 10.4230/LIPIcs.GIScience.2021.I.11

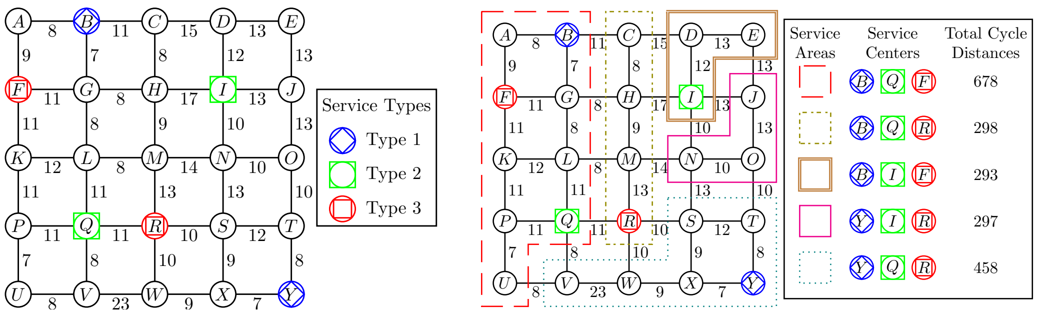
Supplementary Material The source code is available on our research group website http://faculty.eng.fau.edu/yangk/home/NSF_Career_Projects.html [3].

Funding This work is supported by the National Science Foundation CAREER under Grant No. 1844565.

Acknowledgements We would like to thank the US National Science Foundation under Grant No. 1844565. We are particularly thankful to GIScience reviewers for their helpful comments.

1 Introduction

Given a spatial network and a set of service center nodes from k different resource types (e.g. gas stations, grocery stores, shelters, hospitals, etc), a Multiple Resource-Network Voronoi Diagram (MRNVD) partitions the spatial graph into a set of Service Areas (SA) that can minimize the total cycle distances of graph-nodes to allotted k service center nodes with different resource types. Figure 1a shows an example input of MRNVD consisting of a graph with 25 graph-nodes ($i.e., A, B, \ldots, Y$) and service center nodes with three types (i.e., $Type1(B, Y)$, $Type2(I, Q)$, and $Type3(F, R)$). Figure 1b shows an example output of MRNVD where the graph is partitioned such that every graph-node is allotted to three service centers with different types. The objective is to minimize the total cycle distances of graph-nodes to allotted k service center nodes with different types. The MRNVD problem is NP-hard (a proof is provided in Section 2.1). Intuitively, the problem is computationally challenging because of the large size of the transportation network.



(a) Input with three types of service centers. **(b)** Output (Polygons show Service Areas).

Figure 1 Example of Input and Output of MRNVD (Best in Colors).

1.1 Application Domain

The MRNVD problem is important for critical societal applications such as assigning essential resources (e.g., food, water, gas, and medical assistance) to residents impacted by man-made or natural disasters. The objective of MRNVD is to minimize the total cycle distances such that residents can quickly visit their allotted service centers and back to their original location. MRNVD can help us to identify the most efficient route to visit all required service centers. In addition, the simple format of information is vital to communicate effectively during an emergency. MRNVD provides compact and simple representation of Service Areas (SA) that can mitigate panic and chaos and allow for efficient delivery of critical information to the public. Examples of such situations are provided in Table 1.

Table 1 Applications of MRNVD.

Applications	Benefit of MRNVD Service Areas
Emergency Resource Allocation	Develop an emergency plan to help citizens to minimize their travel times to obtain all required resources.
Store Choices	Provide an efficient route to save time and gas while shopping.
Tourist Site Selection	Recommend a tourist route that can visit attractions with different types.

2 Problem Definition

In our formulation of the MRNVD problem, a transportation network is represented and analyzed as an undirected graph composed of nodes and edges. Every node represents a spatial location in geographic space (e.g., road intersections), which can be used as a proxy for locations of residents. Every edge between two nodes represents a road segment and has a travel distance. Every service center has a resource type (e.g., water, food, gas, medicine, etc.). The $MRNVD(N, E, S, D)$ problem is defined as follows:

Intput: A transportation network G with

- a set of graph-nodes N and a set of edges E,
- a set of service center locations with k different resource types $S \subset N$, and
- a set of nonnegative real distances of edges $D : E \rightarrow R_0^+$

Output: A Multiple Resource Network Voronoi Diagram (MRNVD)

Objective:

- Min-sum: Minimize the total cycle distances of graph-nodes to their allotted k service center nodes with different types of resources.

Constraints:

- Service Area (SA) allotment must be k service center nodes with different types of resources.

▶ **Definition 1 (Cycle Distance).** *Given a starting point and a set of k different service centers, the cycle distance is the distance of the shortest route that visits k service centers and returns to the starting point.*

2.1 Problem Hardness

The NP-hardness of MRNVD follows from a well-known result about the NP-hardness of the traveling salesman problem.

▶ **Theorem 1.** *The MRNVD problem is NP-hard.*

Proof. The NP-hardness of MRNVD can be proved by reduction from a well known NP-hardness problem, the traveling salesman problem (TSP) [19]. Given a starting point o and a set of service centers S, TSP finds the shortest cycle distance of o. Let $A = (o, S)$ be an instance of TSP, where o is the starting point and S is a set of service centers. Let $B = (O, S)$ be an instance of the MRNVD problem, where O is a set of staring points and S is a set of service centers. Assume that every service center has a different type. Let $O = \{o\}$. Then the instance of TSP is a special case of MRNVD, where O is a set with a single element (i.e., o). Since A is constructed from B in polynomial-bounded time, the proof is complete. ◀

2.2 Our Contribution

In this paper, we propose a novel algorithm for creating MRNVD based on two Distance bounded Pruning (DP) methods. Our approach has three key components: 1) Straight-Distance bounded Pruning (SDP), 2), Triangular-Distance bounded Pruning (TDP) and 3) 2-opt cycle route computation. In addition, we design a baseline algorithm to evaluate the performance of the proposed approach. Specifically, our contribution is as follows:

- We introduce a new Network Voronoi Diagram, namely Multiple Resource Network Voronoi Diagram (MRNVD).
- We prove that the MRNVD problem is NP-hard.
- We design a baseline algorithm that can produce the optimal solution of MRNVD.
- We propose the Distance bounded Pruning (DP) algorithm based on three key ideas: 1) Straight-Distance bounded Pruning (SDP), 2), Triangular-Distance bounded Pruning (TDP) and 3) 2-opt cycle route computation.
- Our experimental results and a case study using real-world datasets demonstrate that our proposed algorithm outperforms the baseline algorithm and significantly reduces the computational cost to create a MRNVD.

2.3 Related Work

Network Voronoi Diagram (NVD) is extensively used to identify the nearest service center [7, 16, 15, 22]. However, the application of NVD is limited to a single type of resource [17]. Consider the example of the resident who is looking for gas, water, and medicine at the same time. NVD cannot minimize the travel time to visit three service centers for each resource. Recently two-site network Voronoi diagrams were proposed to identify the best route for two different resources [5, 6]. The general idea is to find the minimum triangle-perimeter to partition the spatial network to a set of Service Areas. However, two-site network Voronoi diagrams cannot be generalized into MRNVD due to the hardness of the cycle distance computation. The Voronoi based k nearest neighbor search for spatial network databases was proposed to identify k different nearest service centers [13]. However, the Voronoi k Nearest Neighbor cannot produce the minimum cycle distance because it considers only the distance of the graph-node to service center nodes. There are slightly different approaches for partitioning urban areas into functional or service regions. The multiplicatively weighted order-k Minkowski-metric Voronoi diagrams were utilized to develop a map-based emergency support system [14]. The partitioning method based on street intersections and barriers was developed to support mobility infrastructure planning and optimization in an urban environment [10]. In this work, we propose a novel approach for creating MRNVD that can minimize the total cycle distances of graph-nodes to their allotted k service center nodes with different types.

2.4 Scope and outline

The rest of the paper is organized as follows: Section 3 explains the baseline and proposed pruning approaches for the MRNVD problem. We provide correctness proofs of the proposed approaches in Section 4. In Section 5, we give a cost model of our proposed approaches. Section 6 presents the experimental observations and results. A real world example is given in Section 7 as a case study. Finally, Section 8 concludes the paper.

3 Proposed Approach for MRNVD

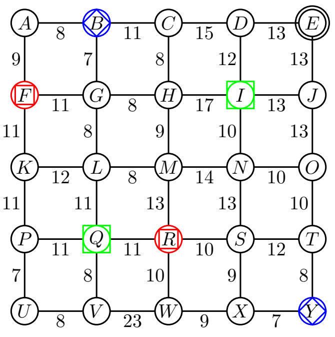
In this section, we first describe the baseline approach that creates the optimal MRNVD, and then we introduce the Distance bounded Pruning (DP) approach that can reduce the computational cost by using three key components: 1) Straight-Distance bounded Pruning (SDP), 2) Triangular-Distance bounded Pruning (TDP), and 3) 2-opt cycle route computation.

3.1 Baseline approach

The baseline approach starts by generating all possible combinations of k different service center nodes and identifies the shortest cycle distances of graph-nodes to their allotted service centers. The key component of the baseline approach is to utilize the dynamic programming technique to find the shortest cycle distance of a graph-node to its allotted service centers [2, 12].

Consider the example input of MRNVD in Figure 2a (reproduced from Figure 1a). Let us identify the cycle distance for node E. First, the baseline approach generates all combinations for three types of service centers. In this example, the service centers are of three types: Type1, Type2, and Type3, and each type has two service centers (i.e., $Type1(B, Y)$, $Type2(I, Q)$, and $Type3(F, R)$). The number of combinations is $2^3 = 8$ and these combinations are $\{B, F, I\}$, $\{B, I, R\}$, $\{B, F, Q\}$, $\{B, Q, R\}$, $\{F, I, Y\}$, $\{I, R, Y\}$, $\{F, Q, Y\}$, and $\{Q, R, Y\}$. Second, it computes the cycle distance for each combination using the Held–Karp algorithm (see Figure 2b) [12, 21]. Since combination $\{B, F, I\}$ can produce the shortest cycle for node E (i.e., $E \rightarrow I \rightarrow F \rightarrow B \rightarrow E$ (117)), the baseline approach assigns service center nodes $\{B, F, I\}$ to node E.

The baseline approach examines all graph-nodes (i.e., nodes A–Y) and computes the cycle distance for each graph-node. It creates the optimal solution for MRNVD (see Lemma 1). However, since the computational cost is exponential, it is challenging to find the optimal solution for large-size transportation network (see Section 5.1) [21].



(a) Input Example.

Combinations	Shortest Cycle	Cycle Distance
$\{B, F, I\}$	$E \xrightarrow{25} I \xrightarrow{36} F \xrightarrow{17} B \xrightarrow{39} E$	<u>117</u>
$\{F, I, Y\}$	$E \xrightarrow{25} I \xrightarrow{36} F \xrightarrow{66} Y \xrightarrow{44} E$	171
$\{B, I, R\}$	$E \xrightarrow{25} I \xrightarrow{33} R \xrightarrow{36} B \xrightarrow{39} E$	133
$\{I, R, Y\}$	$E \xrightarrow{25} I \xrightarrow{33} R \xrightarrow{26} Y \xrightarrow{44} E$	128
$\{B, F, Q\}$	$E \xrightarrow{39} B \xrightarrow{17} F \xrightarrow{30} Q \xrightarrow{63} E$	149
$\{F, Q, Y\}$	$E \xrightarrow{44} Y \xrightarrow{37} Q \xrightarrow{30} F \xrightarrow{55} E$	166
$\{B, Q, R\}$	$E \xrightarrow{39} B \xrightarrow{26} Q \xrightarrow{11} R \xrightarrow{58} E$	134
$\{Q, R, Y\}$	$E \xrightarrow{44} Y \xrightarrow{26} R \xrightarrow{11} Q \xrightarrow{63} E$	144

(b) Cycle Distance Computation for node E.

Figure 2 Example of service center allotment for node E using the cycle distance computation.

3.2 Proposed Approaches

In this section, we describe two novel pruning methods (i.e., Straight-Distance bounded Pruning (SDP) and (b) Triangular-Distance bounded Pruning (TDP)) to reduce the search space for the MRNVD problem. In addition, we introduce the 2-opt cycle route computation method to minimize the computational cost for the cycle distance.

3.2.1 Straight-Distance bounded Pruning (SDP)

The main performance bottleneck of the baseline approach is to compute the cycle distance for each combination of service centers. In this subsection, we introduce the Straight-Distance bounded Pruning (SDP) method that can reduce the search space for the combinations using a Set Window (SW).

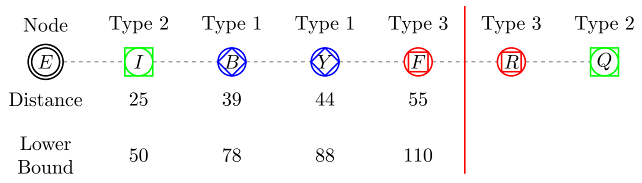
▶ **Definition 2 (Set Window).** *Given a set of service center nodes S, a node n, and a distance bound d, a Set Window (SW) is defined as a set of service centers $SW \subset S$ that are within distance of d from node n.*

The core idea of SDP is to find the lower and upper bounds of the cycle distance based on a Set Window (SW) and rule out non-optimal combinations when the center nodes in these combinations violate the bound constraints. This approach can reduce the number of computations for the cycle distance without losing the optimality of the solution (see Lemma 3 and 4). The SDP method proceeds in three steps. First, it constructs the initial Set Window (SW). Next, it incrementally increases the size of SW until it meets the lower and upper bounds. Finally it finds the minimum cycle distance in SW.

▶ **Definition 3 (Initial Set Window).** *Given a set of service center nodes S and a node n, the Initial Set Window is defined as the minimum set of closest service centers to n that contain all types of service centers.*

SDP starts by creating an initial SW for each graph-node. Given a node $n \in N$, SDP constructs an ordered-list of service centers based on the distance from node n. Then, it identifies the minimum-sized SW that includes all types of service centers. We set the minimum-sized SW as the initial SW because it contains k different service centers and creates the cycle distances that are feasible but may not be optimal (Lemma 2).

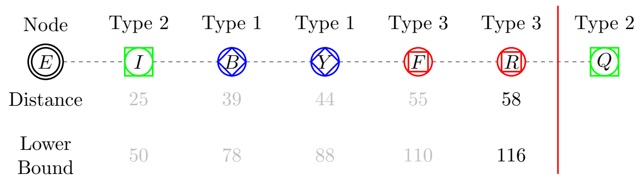
The SDP method incrementally increases the size of SW for node n and updates the lower and upper bounds of the optimal cycle distance of node n. Given a Set Window (SW), the lower-bound of the cycle distance is obtained by doubling the distance of n to the farthest service center node in SW (see Lemma 3). The upper-bound of the cycle distance is the minimum cycle distance among all combinations in SW (see Lemma 4). If the lower-bound is greater than the upper-bound, SDP stops increasing the size of SW and finds the optimal cycle distance in the current SW.



Combinations	Shortest Cycle	Cycle Distance
$\{B, F, I\}$	$E \to I \to F \to B \to E$	117
$\{F, I, Y\}$	$E \to I \to F \to Y \to E$	171

Figure 3 Initial Set Window (SW) for node E and lower and upper bounds.

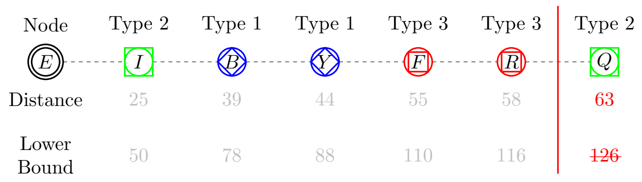
Consider again node E in Figure 2. Figure 3 shows the example of the initial SW for node E. Given a node E, all service centers are ordered by the distance from node E. The vertical bar splits the ordered-list into the left and right parts; The left part becomes the initial SW (i.e., $\{I, B, Y, F\}$) whose size is minimal and includes all types of service centers. Then SDP computes the initial lower and upper bounds of the cycle distance of node E. The lower-bound is the double of the distance from node E to the farthest node in SW (see Lemma 3). Since node F is the farthest service center node from E in SW, the lower-bound becomes 110. Next, SDP generates all possible combinations of the three different service types (i.e., $\{B, F, I\}$ and $\{F, I, Y\}$) and identifies the minimum cycle distance among these combinations. Since the minimum cycle distance is 117 in the initial SW, the upper-bound becomes 117 (see Lemma 4). Since the upper-bound is greater than the lower-bound, SDP can increase the size of SW.



Combinations	Shortest Cycle	Cycle Distance
$\{B, F, I\}$	$E \to I \to F \to B \to E$	117
$\{F, I, Y\}$	$E \to I \to F \to Y \to E$	171
$\{B, I, R\}$	$E \to I \to R \to B \to E$	133
$\{I, R, Y\}$	$E \to I \to R \to Y \to E$	128

Figure 4 SDP: Iteration 1: lower and upper bounds in SW.

After the construction of the initial SW, SDP incrementally increases the size of SW by one and updates the lower and upper bounds until SW violates the bound constraints. Figure 4 shows SW whose size is increased by one (i.e., $\{I, B, Y, F, R\}$). The new combinations generated by SW are $\{B, I, R\}$ and $\{I, R, Y\}$. The upper-bound is the same as the previous upper-bound (i.e., 117), but the lower-bound is updated to 116. Since the upper-bound is greater than the lower-bound, SDP continues to increase the size of SW.



Combinations	Shortest Cycle	Cycle Distance
$\{B, F, I\}$	$E \to I \to F \to B \to E$	117
$\{F, I, Y\}$	$E \to I \to F \to Y \to E$	171
$\{B, I, R\}$	$E \to I \to R \to B \to E$	133
$\{I, R, Y\}$	$E \to I \to R \to Y \to E$	128

Figure 5 SDP: Iteration 2: lower and upper bounds in SW.

Figure 5 shows that SDP adds node Q to increase the size of SW by one. The lower-bound is updated to 126. Since the lower-bound is greater than the upper-bound (i.e., 117), SW violates the bound constraints. Therefore, SDP assigns $\{B, F, I\}$ to node E and stop the search immediately. The SDP method can be summarized as follows. 1) construct the initial SW, 2) incrementally increase the size of SW and update lower and upper bounds, 3) stop when SW violates the bound constraints and return the optimal cycle distance.

3.2.2 Triangle-Distance bounded Pruning Approach

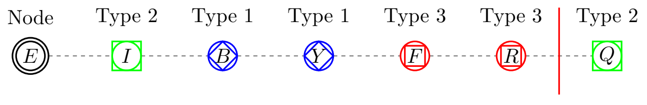
In this subsection, we introduce the Triangle-Distance bounded Pruning (TDP) method that can prune the search space for combinations using the max-min triangle-distance.

▶ **Definition 4** (**Triangle-Distance(n, s_1, s_2)**). *Given a starting node n and two service center nodes s_1 and s_2, the triangle-distance is defined as the cycle distance of $n \to s_1 \to s_2 \to n$.*

▶ **Definition 5** (**Min Triangle-Distance(n, s_1, t)**). *Given a starting node n, a service center node s_1, and a type of service centers t, the min triangle-distance is defined as the minimum Triangle-Distance(n, s_1, s_2), where type(s_2) $= t$.*

▶ **Definition 6** (**Max-Min Triangle-Distance(n, s_1)**). *Given a starting node n, a service center node s_1, and a set of types of service centers T, the max-min triangle-distance is defined as the maximum of min triangle-distance(n, s_1, $t \in T$).*

The core idea of TDP is that when Max-Min Triangle-Distance (n, s_1) is greater than the upper-bound of the cycle distance, the algorithm will not compute the cycle distance of the combinations that includes node s_1 (see Lemma 5). We refer to s_1 as the anchor-node.



Combinations	Shortest Cycle	Cycle Distance
$\{B, F, I\}$	$E \to I \to F \to B \to E$	<u>117</u>
$\{F, I, Y\}$	$E \to I \to F \to Y \to E$	171
$\{B, I, R\}$		
$\{I, R, Y\}$		

Figure 6 Node E Set Window (SW) and all combinations for TDP.

Given a Set Window (SW), TDP starts by constructing the triangle-distance table for anchor-nodes (i.e., service center nodes) and computes the max-min triangle-distance for each anchor-node. Consider the Set Window (SW) in Figure 6 (reproduced from Figure 4). First, TDP groups a set of service center nodes based on types and constructs the triangle-distance table for anchor-nodes (i.e., nodes I, B, Y, F, and R) (see Figure 7). In this example, the group of type 1 is $\{B, Y\}$, the group of type 2 is $\{I\}$, and the group of type 3 is $\{F, R\}$. Next, TDP computes the min triangle-distance for each type of service centers. For instance, the min triangle-distance with anchor-node I and Type 1 becomes 96. Then, it defines the max-min triangle-distance for each anchor-node.

Note that the upper-bound of the cycle distance in SW is 117 (see Figure 6). Since the max-min triangle-distances of nodes Y and R are greater than the upper-bound of the cycle distance (i.e., 128), nodes Y and R cannot be a part of the shortest cycle. Therefore, TDP can rule out the computations of the cycle distances for combinations $\{F, I, Y\}$ $\{B, I, R\}$ and $\{I, R, Y\}$ because these combinations cannot produce the optimal cycle distance. The TDP method can be summarized as follows. 1) group a set of service centers based on

Anchor Nodes	Type 1		Type 2	Type 3		Min Triangle-Distance			Max-Min Triangle-Distance
	B	Y	I	F	R	Type 1	Type 2	Type 3	
I	96	107	50	116	116	96	50	116	116
B	78	148	96	111	133	78	96	111	111
Y	148	88	107	165	128	88	107	128	128
F	111	165	116	110	153	111	116	110	116
R	133	128	116	153	116	128	116	116	128

Figure 7 Triangle-Distance Table for node E (Highlighted values violate the triangle-distance bound).

types, 2) compute the min-triangle-distance for each anchor-node and each type, 3) compute the max-min triangle-distance for each anchor-node, and 4) rule out the combinations that violate the triangle-distance bound.

3.3 2-opt Cycle Route Computation

Although SDP and TDP rule out the computations of the non-optimal cycle distance, the proposed DP algorithm may be inapplicable for sizable road networks because the computational cost of the optimal cycle distance is exponential in terms of the number of service types (see Section 5.2) [2, 12]. Thus we propose a more scalable algorithm using the 2-opt method [4, 8]. The 2-opt method is a heuristic that repeatedly applies 2-opt swaps to minimize the cycle distance. Our proposed approach uses the nearest neighbor heuristic to construct the initial solution and applies the 2-opt method to find the near-optimal cycle distance [1]. The novel component of our approach is to utilize the Tabu-search method that can easily transform 2-opt swaps to 4-opt or more swaps. A Tabu-search uses a Tabu-list in order to escape from local minima and search neighboring solutions until a certain stopping criterion is satisfied. The algorithm convergence of Distance bounded Pruning (DP) with Tabu-search follows from a well-known result about the convergence of the Convergence Tabu Search (CTS) [11].

Given a solution s, let $N(s)$ be the set of neighborhood solutions of s. Let $G_N = (V, E)$ be a graph induced by $N(s)$, where V is a set of solutions and E represents the neighborhood relationship between two solutions. The CTS algorithm converges and terminates after exploring all solutions S if the following two conditions hold [11]:

1. The neighborhood relation is symmetric, i.e. $x \in N(y) \Leftrightarrow y \in N(x)$ for all $x, y \in S$
2. Given a graph G_N, there exists a path between every pair of solutions $x, y \in S$.

Since the 2-opt method satisfies the two conditions, DP with Tabu-search converges and terminates (see Lemma 6). In addition, it can significantly reduce the computational cost of the DP algorithm (see Section 5.3).

Algorithm 1 presents the pseudo-code for Distance bounded Pruning (DP). DP computes the distance matrix for graph-nodes in G (Line 1). For each graph-node n, DP computes the cycle distance of n (Line 2-8). First, it constructs the initial Set Window (SW) and compute the lower and upper bounds of the optimal cycle distance (Line 3-4). Next, it incrementally increases the size of SW and updates the lower and upper bounds of the optimal cycle distance (Line 6-7). SDP and TDP are used to rule out the non-optimal combinations for the cycle distance computation. When SW violates the bound constraints, DP finds the optimal cycle distance and assigns service center nodes to n (Line 9). This process continues until all graph-nodes are allotted (Line 2). Finally, MRNVD is returned (Line 11).

Algorithm 1 Distance bounded Pruning algorithm (Pseudo-code).

```
Inputs:
 - A transportation network $G(N, E)$ with graph-nodes $N$ and edges $E$.
 - A set of service center locations with $k$ different resource types $S \subset N$.
 - Every edge has a distance $d(e)$
Output: Multiple Resource Network Voronoi Diagram
Steps:
 1: Compute the distance matrix for graph-nodes in $G$.
 2: for graph-node $n \in N$ in $G(N, E)$ do
 3:     Construct the initial Set Window (SW) for $n$.
 4:     Compute the initial lower and upper bounds of the cycle distance.
 5:     while the bound constraints are not violated do
 6:         Increase the size of SW by one.
 7:         Update the lower and upper bounds and prune search space using SDP and TDP.
 8:     end while
 9:     Identify the cycle distance of $n$ and allot service centers nodes to $n$.
10: end for
11: return MRNVD (i.e., allotment of graph-nodes to their service centers).
```

4 Analysis of the MRNVD proposed approaches

In this section, we prove that the proposed DP approaches are correct, i.e., the DP algorithm creates a MRNVD.

▶ **Lemma 1.** *The baseline approach to the MRNVD problem creates the optimal solution.*

Proof. The baseline approach considers all combinations of service centers for the cycle distance. For each combination, it utilizes the dynamic programming method to compute the cycle distance [12]. The optimal structure of the cycle distance is that every sub-path of the minimum cycle is itself a path with the minimum distance. Therefore, the output of the baseline approach is optimal. ◀

▶ **Lemma 2.** *The initial Set Window (SW) should be the minimum-sized SW that contains k different service centers.*

Proof. Assume that the initial SW has less than k different service centers. Then the initial SW cannot produce the feasible solution for the allotment. This contradicts the original assumption. ◀

▶ **Lemma 3.** *The lower-bound of the cycle distance is obtained by doubling the distance of n to the farthest service center node in the Set Window (SW).*

Proof. The lower-bound of the cycle distance can be proven by the mathematical induction method. Let n be the starting point, let S_{sw} be a set of service center nodes in SW, and let s_0 be the farthest service center node from n. We begin with the initial SW. The initial SW is the minimum node set that includes all different types of service centers. Therefore, the feasible solution of the minimum cycle should include the farthest service center node in the initial SW. Let the shortest distance of n to s_0 be $cost(n, s_0)$. Assume that we add one service center $s \in S$ to cycle $n \to s_0 \to n$. After the addition of the service center $s \in S_{sw}$, the shortest distance of the cycle monotonically increases according to the triangle inequality theorem. Therefore, $2 \cdot cost(n, s_0)$ becomes the lower-bound of the cycle distance for the initial SW. Next, we increase the size of SW by one. Then the new added node becomes the farthest service center node from n. Let s_1 be the farthest service center node from n. SW should include s_1 to compute the cycle distance. If not, we do not need to increase the size of SW. According to the triangle inequality theorem, $2 \cdot cost(n, s_1)$ becomes the lower-bound of the cycle distance for SW. Therefore, we complete the proof by induction. ◀

▶ **Lemma 4.** *The upper-bound of the cycle distance is the minimum cycle distance in SW.*

Proof. Let S be a set of service center nodes and S_{sw} be a set of service center nodes. Since $S_{sw} \subset S$, the minimum cycle distance in S_{sw} is greater than or equal to the optimal cycle distance in S. Therefore, the upper-bound of the cycle distance is the minimum cycle distance in SW. ◀

▶ **Lemma 5.** *If $Max\text{-}Min$ Triangle-Distance(n, s_1) is greater than the upper-bound of the cycle distance, then anchor-node s_1 cannot be a part of the optimal cycle.*

Proof. Let n be the starting point, let s_1 be the anchor point, and let T be a set of types of service centers. Min *Triangle-Distance*(n, s_1, t) becomes a lower-bound of the cycle distance for every type $t \in T$. Therefore, $\max_{t \in T}$ *Min-Triangle-Distance*(n, s_1, t) becomes a lower-bound of the cycle distance that include anchor-node s_1. Since the lower-bound cannot be greater than the upper-bound, anchor-node s_1 cannot be a part of the optimal cycle. ◀

▶ **Lemma 6.** *The 2-opt method with Tabu-search converges and terminates.*

Proof. The 2-opt method has the symmetric neighborhood relation. Moreover, every solution has a path to other solutions by swapping two nodes. Since the 2-opt method satisfies the two conditions of CTS, the proof is complete. ◀

5 Algebraic Cost Model of Pruning Algorithms

The goal of this section is to present cost models for our proposed approaches. Let n be the number of graph-nodes, let k be the number of types in service centers, let c be the maximum number of service centers for a service type.

5.1 Baseline Approach

The baseline approach starts by generating all possible combinations of k different service centers. This takes $O(c^k)$. Given a combination, the cost of computation for the cycle distance is $2^k \cdot k^2$ [1]. Since the number of combinations is bounded by $O(c^k)$, the minimum cycle distance for a graph-node can be obtained by the cost of $O(c^k \cdot 2^k \cdot k^2)$. The number of graph-nodes is n. Therefore, the baseline approach takes $O(c^k \cdot 2^k \cdot k^2 \cdot n)$.

5.2 Distance Bounded Pruning (DP) Approach

The Distance bounded Pruning (DP) approach starts by ordering service centers based on the distance. This takes $O(c \cdot k \cdot \log(c \cdot k))$. Next, The Straight-Distance bounded Pruning (SDP) method creates an initial Set Window (SW) and incrementally increase the size of SW. The size of SW is bounded by $O(c \cdot k)$. During this incremental process, the cost for computing the lower-bounds is $O(c \cdot k)$ and the cost for computing the upper-bound is $O(c^k \cdot 2^k \cdot k^2)$. The Triangle-Distance bounded Pruning (TDP) method creates the Triangle-Distance Tables to compute the max-min triangle-distances. This takes $O(c^2 \cdot k^2)$. Thus, the total cost of the allotment for each node is $O(c \cdot k + c^2 \cdot k^2 + c^k \cdot 2^k \cdot k^2) = O(c^k \cdot 2^k \cdot k^2)$. Since the number of graph-nodes is n, DP takes $O(c^k \cdot 2^k \cdot k^2 \cdot n)$. In worst case, the cost model of DP is the same as the cost model of the baseline approach. However, DP can rule out the non-optimal combinations and significantly reduce the number of computations of the cycle distances using SDP and TDP.

5.3 DP with 2-opt cycle route computation

DP with 2-opt cycle route computation (DP 2-opt) uses the Tabu-search method and reduces the computational cost of the cycle distances. The Tabu-search requires multiple iterations to find the near-optimal solution [9, 20]. At each iteration, it swaps two pairs of nodes and temporally locks these nodes for the iteration. Each swap takes $O(k^2)$. The number of swaps is bounded by $O(k)$. Thus, the cost of each iteration takes $O(k^3)$. Assume that the number of iteration is bounded by $O(i)$. Then, the cost of computations of the cycle distance is $O(k^3 \cdot i)$. Therefore, the cost model for DP 2-opt is $O(c^k \cdot k^3 \cdot i \cdot n)$. Since the number of iterations (i.e., i) until convergence is often small, the cost model in practice is considered to be $O(c^k \cdot k^3 \cdot n)$.

6 Experimental Evaluation

We conducted experiments to evaluate performance of Baseline and Distance bounded Pruning (DP) approaches. The overall goal was to show the performance improvements to create a MRNVD that can be obtained by the DP approach. We wanted to answer four questions: (1) What is the effect of the number of service types? (2) What is the effect of the number of service centers? (3) What is the effect of the size of the network (i.e., number of graph-nodes)? (4) Is DP algorithm correct, and is the solution quality preserved?

6.1 Experiment Layout

Figure 8 shows our experimental setup. We chose five different municipal areas in the U.S. from OpenStreetMap [18]. We used the locations of service centers in these areas and created a Multiple Resource Network Voronoi Diagram (MRNVD). We tested three approaches: (1) Baseline approach (BL), (2) Distance bounded Pruning approach (DP), and (3) DP with 2-opt heuristic for cycle distance calculation approach (DP 2-opt).

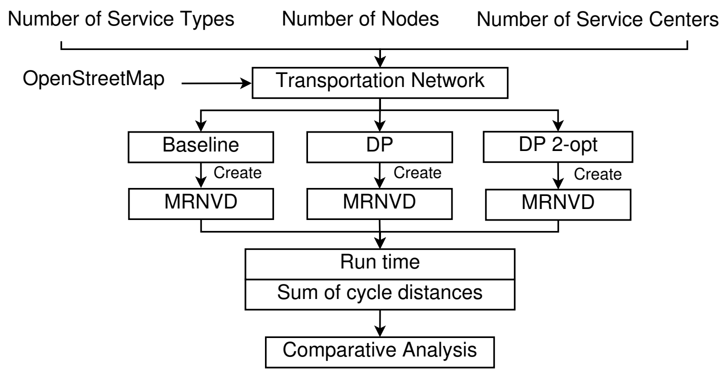


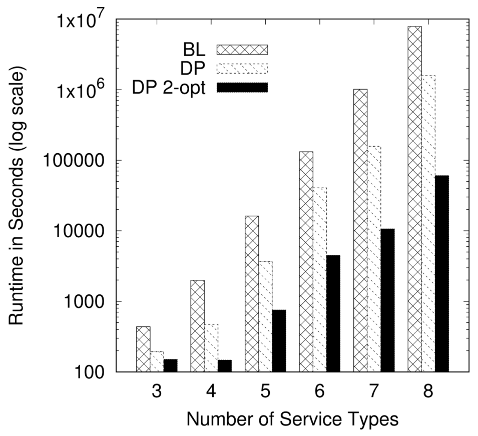
Figure 8 Experiment Layout.

6.2 Experiment Results and Analysis

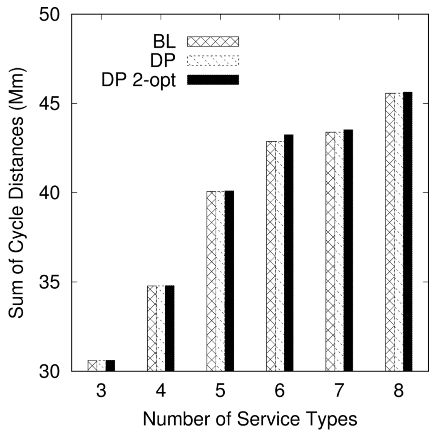
We experimentally evaluated the proposed algorithms by comparing the impact on performance of (1) number of service types, (2) number of service centers per type and (3) size of the transportation network. We extracted the locations of service centers from OpenStreetMap datasets and then randomly chose a set of service centers from extracted ones to vary the number of service centers. The algorithms were implemented in Java with a 32 GB memory run-time environment. All experiments were performed on an Intel Core i5 machine running Windows 10 with 32 GB of RAM.

6.2.1 Effect of Number of Service Types

The first set of experiments evaluated the effect of the number of service types on the performance of the algorithms. We used a Florida road map consisting of 460,791 nodes and 653,392 edges. We fixed the number of nodes to 5,000 and the number of service centers to 3. We varied the number of service types from 3 to 8. We randomly chose the locations of service centers and constructed 45 test cases. Performance measurements were execution time and the sum of the cycle distances. The performance measurements were averaged over 45 test runs. Figure 9a gives the execution times. As can be seen, the DP approaches outperforms the baseline approach. This is because the number of combinations for the cycle distance computation increases as the number of service types increases. DP with 2-opt heuristic outperforms other approaches because it can reduce the computational cost for the cycle distance. When comparing the sum of the cycle distances, we see that the DP approach produce the optimal solution (see Figure 9b). This means that SDP and TDP have no effect on the solution quality. DP with 2-opt heuristic (DP 2-opt) performs almost identically to the optimal approaches. As the number of service types increase, the sum of the cycle distances increases.



(a) Run Time Comparison.

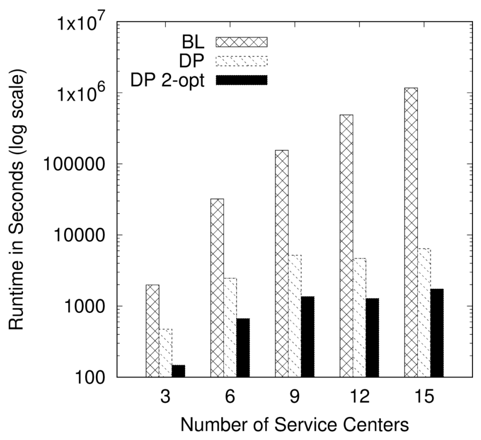


(b) Comparison of Solution Quality.

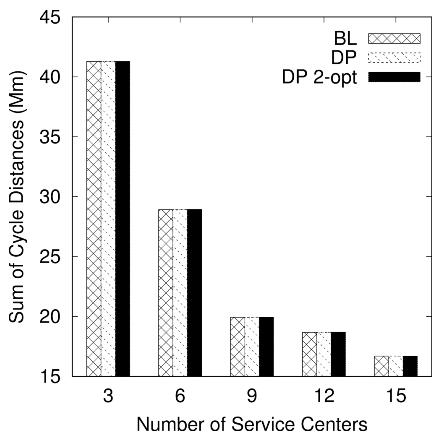
Figure 9 Effect of number of service types ($n = 5,000$, $c = 3$).

6.2.2 Effect of Number of Service Centers

The second set of experiments evaluated the effect of the number of service centers on the performance of the algorithms. Performance measurements were execution time and the sum of the cycle distances. We fixed the number of nodes to 5,000 and the number of service types to 4. The number of service centers was varied from 3 to 15. Locations of service centers were randomly chosen in 54 test cases. Figure 10a shows that the DP approaches significantly outperform the baseline approach. The performance gap increases as the number of service centers increases. This is because the number of combinations for the cycle distance computation increases as the number of service centers increases. DP 2-opt significantly outperforms other approaches due to the reduced computational cost for the cycle distance. Figure 10b shows that the DP approach performs exactly the same as the baseline approach. We can see that DP 2-opt was faster than DP, albeit slightly lower performance in terms of sum of cycle distances. As the number of service centers increases, the sum of cycle distance decreases.



(a) Run Time Comparison.

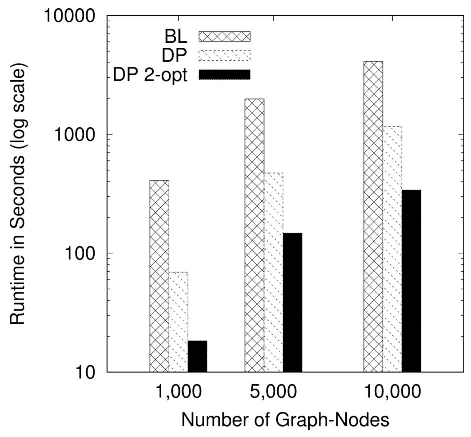


(b) Comparison of Solution Quality.

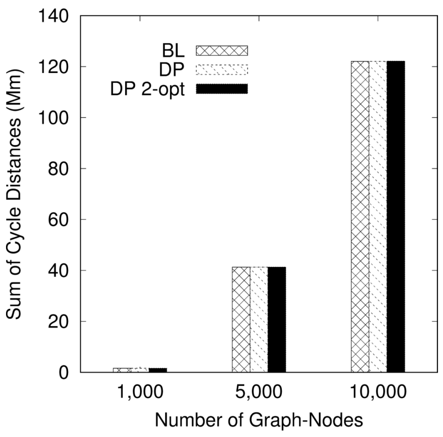
Figure 10 Effect of number of service centers ($n = 5,000$, $k = 4$).

6.2.3 Effect of Network Size

The third set of experiments evaluated the effect of the network size on algorithm performance. We fixed the number of service types to 4 and the number of centers per type to 3. We increased the number of nodes from $1,000$ to $10,000$. Service center locations were chosen randomly and execution times were averaged over 30 test runs for each road network. Figure 11a shows that the DP approaches significantly outperforms the baseline (BL) approach. This is because the size of the Service Areas increases as the number of nodes increases. DP 2-opt outperforms others due to the reduction of computational cost for the cycle distance. Figure 11b shows that BL and DP perform identical. DP 2-opt performs almost identically to the optimal approaches. As the number of nodes increases, the sum of cycle distance increases.



(a) Run Time Comparison



(b) Comparison of Solution Quality.

Figure 11 Effect of network size ($k = 4$, $c = 3$).

6.2.4 Discussion

The proposed DP approach achieves a significant computation performance gain over the optimal baseline approach. This improvement was obtained by using three key components: 1) Straight-Distance bounded Pruning (SDP), 2) Triangle-Distance Bounded Pruning (TDP), and 3) 2-opt cycle route computation. The baseline approach computes the optimal cycle distances using dynamic programming; However the computational cost of the baseline approach is prohibitive on the large sized networks [12]. To remedy this issue, SDP creates a Set Window (SW) and reduces the number of computations for the cycle distance by using the bound constraints. TDP reduces the number of computations by identifying the anchor nodes that violate the upper-bound constraint. The 2-opt cycle route computation further reduces the computational cost by utilizing the Tabu-search method. The experimental results shows that the proposed approaches significantly reduce the computational cost to create a MRNVD. The source code is available on our research group website [3].

7 Case Study with Boca Raton road network

In our case study, we created a MRNVD that can identify a set of Service Areas (SAs) to minimize the travel time for citizens to visit all required service centers. For transportation network, we used a Boca Raton, FL road map consisting of $18,679$ nodes and $25,835$ edges (Figure 12a). We chose five different service types (i.e. grocery stores, gas stations, pharmacies, healthcare facilities and law enforcement departments) and nine service centers for each service type. Each circle symbol represents a different type of service centers (Figure 12b). Figure 12c shows the MRNVD constructed thirteen Service Areas that can minimize the total cycle distances. The sum of cycle distances using DP and DP 2-opt are $159,872km$ and $160,021km$ respectively. Our case study showed that the run-time of the baseline approach took 4 hours to produce a MRNVD. DP took 6 minutes whereas DP 2-opt took 1 minute. The solution of the DP approach is exactly the same as that produced by the baseline approach.

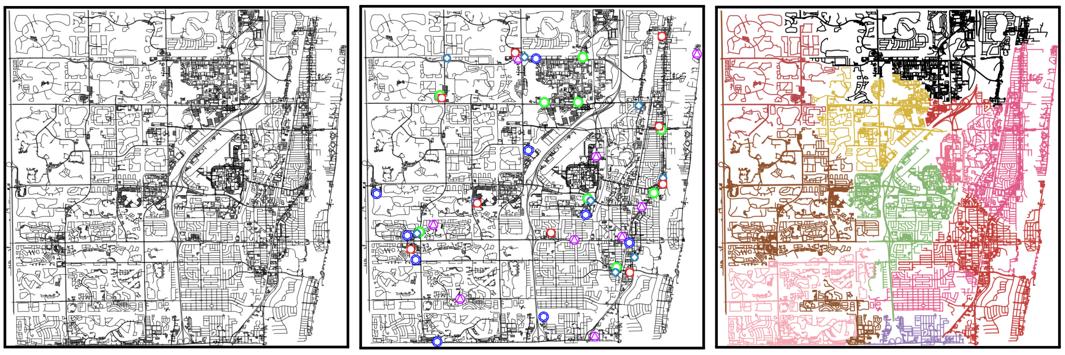

(a) Boca City Road Network. **(b)** Locations of Service Centers. **(c)** MRNVD with 13 Service Areas.

Figure 12 Case Study: Boca Raton, FL road map (Best in Colors).

8 Conclusion and Future work

We presented the problem of creating a Multiple Resource Network Voronoi Diagram (MRNVD). An important societal application of MRNVD is promoting transportation resiliency before or after a disaster. The MRNVD problem is challenging due to multiple different types of resources. General Network Voronoi Diagram uses the distance metric and divides the region based on the closest service center. However, the distance for multiple resources cannot utilize the absolute distance metric because it uses the cycle distance metric for visiting all required service centers. In this paper, we introduced a novel Distance bounded Pruning (DP) approach for creating a MRNVD that can minimize the total cycle distances of graph-nodes to allotted k service center nodes. We presented experiments and case study using a Boca Raton road map.

In future work, we plan to further explore new optimal pruning methods to reduce the computational cost for creating a MRNVD. In addition, we will develop a parallel formulation of the propose approaches to handle continental-sized transportation networks. We will also investigate the effect of applying spatial filters on reducing the size of the network and improving the performance of MRNVD. MRVND with Monte Carlo simulation may solve the facility location problem. We will study new method that determines the near-optimal positions of service facilities. Lastly, we plan to design new MRNVD problem that includes the capacity constraint for each service center and the directional constraint based on directed graphs.

References

1 David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study*. Princeton university press, 2006.

2 Richard Bellman. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, 9(1):61–63, 1962.

3 MRNVD Source Code. `http://faculty.eng.fau.edu/yangk/home/NSF_Career_Projects.html`,Retrieved Jun. 2020.

4 Georges A Croes. A method for solving traveling-salesman problems. *Operations research*, 6(6):791–812, 1958.

5 Matthew T Dickerson and Michael T Goodrich. Two-site voronoi diagrams in geographic networks. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, pages 1–4, 2008.

6 Matthew T Dickerson, Michael T Goodrich, Thomas D Dickerson, and Ying Daisy Zhuo. Round-trip voronoi diagrams and doubling density in geographic networks. In *Transactions on Computational Science XIV*, pages 211–238. Springer, 2011.

7 Martin Erwig. The graph voronoi diagram with applications. *Networks: An International Journal*, 36(3):156–163, 2000.

8 Merrill M Flood. The traveling-salesman problem. *Operations research*, 4(1):61–75, 1956.

9 Fred Glover. Tabu search—part i. *ORSA Journal on computing*, 1(3):190–206, 1989.

10 Anita Graser. Tessellating urban space based on street intersections and barriers to movement. *GI_Forum 2017*, 5(1):114–125, 2017.

11 Said Hanafi. On the convergence of tabu search. *Journal of Heuristics*, 7(1):47–58, 2001.

12 Michael Held and Richard M Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied mathematics*, 10(1):196–210, 1962.

13 Mohammad Kolahdouzan and Cyrus Shahabi. Voronoi-based k nearest neighbor search for spatial network databases. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 840–851, 2004.

14 Ickjai Lee, Kyungmi Lee, and Christopher Torpelund-Bruin. Raster voronoi tessellation and its application to emergency modeling. *Geo-spatial Information Science*, 14(4):235–245, 2011.

15 Atsuyuki Okabe, Barry Boots, and Kokichi Sugihara. Nearest neighbourhood operations with generalized voronoi diagrams: a review. *International Journal of Geographical Information Systems*, 8(1):43–71, 1994.

16 Atsuyuki Okabe, Toshiaki Satoh, Takehiro Furuta, Atsuo Suzuki, and Kyoko Okano. Generalized network voronoi diagrams: Concepts, computational methods, and applications. *International Journal of Geographical Information Science*, 22(9):965–994, 2008.

17 Atsuyuki Okabe and Kokichi Sugihara. *Spatial analysis along networks: statistical and computational methods*. John Wiley & Sons, 2012.

18 OpenStreetMap. `http://goo.gl/Hso0`,Retrieved Feb. 2020.

19 Daniel J Rosenkrantz, Richard E Stearns, and Philip M Lewis, II. An analysis of several heuristics for the traveling salesman problem. *SIAM journal on computing*, 6(3):563–581, 1977.

20 Shigeru Tsubakitani and James R Evans. An empirical study of a new metaheuristic for the traveling salesman problem. *European Journal of Operational Research*, 104(1):113–128, 1998.

21 Gerhard J Woeginger. Exact algorithms for np-hard problems: A survey. In *Combinatorial optimization—eureka, you shrink!*, pages 185–207. Springer, 2003.

22 KwangSoo Yang, Apurv Hirsh Shekhar, Dev Oliver, and Shashi Shekhar. Capacity-constrained network-voronoi diagram. *IEEE Transactions on Knowledge and Data Engineering*, 27(11):2919–2932, 2015.