Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

An elliptical cover problem in drone delivery network design and its solution algorithms

Yanchao Liu

Department of Industrial and Systems Engineering, Wayne State University, 4815 4th Street Rm 2169, Detroit, MI 48201, USA

ARTICLE INFO

Article history:
Received 15 November 2021
Accepted 22 April 2022
Available online 28 April 2022

Keywords:
Location
Area coverage
Ellipses
Drone delivery
NP-hard problem

ABSTRACT

Given n demand points in a geographic area, the elliptical cover problem is to determine the location of p depots (anywhere in the area) so as to minimize the maximum distance of an economical delivery trip in which a delivery vehicle starts from the nearest depot to a demand point, visits the demand point and then returns to the second nearest depot to that demand point. We show that this problem is NP-hard, and adapt Cooper's alternating locate-allocate heuristic to find locally optimal solutions for both the point-coverage and area-coverage scenarios. Experiments show that most locally optimal solutions perform similarly well, suggesting their sufficiency for practical use. The one-dimensional variant of the problem, in which the service area is reduced to a line segment, permits recursive algorithms that are more efficient than mathematical optimization approaches in practical cases. The solution also provides the best-known lower bound for the original problem at a negligible computational cost.

1. Introduction

1.1. Problem statement

In this paper, we study the *Euclidean p-Elliptical Cover* problem, stated as follows. Given n demand points with coordinates $(a_i, b_i), i = 1, \dots, n$ on the plane, find p depot locations X_j, $j = 1, \dots, p$, in order to

$$\text{minimize}_{X_1,\dots,X_p} \left\{ \max_{1\le i\le n} \left\{ \min_{1\le j<j'\le p} \left\{ D_i(X_j) + D_i(X_{j'}) \right\} \right\} \right\} \tag{1}$$

where $X_j := (x_j, y_j)$ for $j = 1, \dots, p$ is the location of depot j, and $D_i(X_j) := [(x_j - a_i)^2 + (y_j - b_i)^2]^{1/2}$ is the Euclidean distance between demand point i and depot j. We will refer to this problem simply as the *Elliptical Cover* problem in the sequel.

The problem can be formulated into a mixed integer nonlinear programming (MINLP) model. Define the binary variable

$$z_{ij} = \begin{cases} 1 \text{ if demand point } i \text{ is assigned to depot } j \\ 0 \text{ otherwise} \end{cases}$$

then problem (1) is equivalent to

$$\text{minimize} \quad L \tag{2}$$

$$\text{s.t.} \quad L \ge \sum_{j=1}^{p} z_{ij}[(x_j - a_i)^2 + (y_j - b_i)^2]^{1/2} \quad \text{for } i = 1, \dots, n \tag{3}$$

$$\sum_{j=1}^{p} z_{ij} = k \quad \text{for } i = 1, \dots, n \tag{4}$$

$$z_{ij} \in \{0, 1\} \quad \text{for } i = 1, \dots, n;\ j = 1, \dots, p \tag{5}$$

$$x_j, y_j \in \mathbb{R} \quad \text{for } j = 1, \dots, p \tag{6}$$

with parameter $k = 2$. The Euclidean p-center (EPC) problem (Megiddo & Supowit, 1984), which attempts to find p depots to minimize the maximum distance from a demand point to its respective nearest depot, has the same MINLP formulation with $k = 1$. The minimum enclosing polyellipsoid (MEP) problem (Blanco & Puerto, 2021) can be viewed as a variant of the above model with $k = p$ and with additional constraints on foci locations. These and other related problems and their solution ideas are reviewed in Section 1.3. It is worth noting that in the former case ($k = 1$) the problem is $\mathcal{NP}$-hard, while in the latter case ($k = p$) the problem is convex and polynomially solvable. Therefore, there is not a general conclusion about the problem's complexity based on its mathematical formulation, and good algorithms may have to exploit the geometric properties available under the particular problem setting. In this paper, we focus on analyzing the Elliptical Cover case ($k = 2$), show that it is an $\mathcal{NP}$-hard problem, and propose algorithms for different variants of the problem.

E-mail address: yanchaoliu@wayne.edu

1.2. Motivating application

On-demand delivery of light goods by unmanned aerial vehicles (UAVs, or drones) has emerged as a new mode of last-mile delivery, and has attracted great interest from both industrial and academic communities. Several modes of drone delivery operations have been explored, including drones launched from fixed and moving depots, and drone routes having a single or multiple stops for pickup and delivery, see, e.g., Agatz, Bouman, and Schmidt (2018); Liu (2019); Murray and Chu (2015); Poikonen and Campbell (2020). Given the limited battery and carrying capacities of multicopter drones, we believe that a feasible mode of operation can be as simple as this: single delivery per trip, with the drone starting from a fixed depot, visiting the customer location for dropoff, then returning to (the same or a different) depot. Commercial operations of drone delivery services, including those piloted by Alphabet Wing (Wing, 2021) and Zipline (Zipline, 2021), fall under the “depot-customer-depot” paradigm with stationary depots. Even though Zipline currently adopts a hub-and-spoke network with a single depot at the hub, optimally locating multiple depots will become relevant when its service region expands beyond the round-trip flight range of the battery-powered drone in its fleet.

The location of depots across the service area is an important network design decision that will determine the initial infrastructure cost as well as the safety, efficiency and service level in daily operations. Liu (2021, 2022) discussed the drone depot location problem from the safety and emergency landing perspective, and employed the p-center design to cover the service area with depots such that the required flight distance in a worst-case emergency landing is minimized. It is also important to examine the problem from the cost and operation efficiency perspective.

In designing the system, the flight range requirement on drones is a primary factor to consider. If depots and customer locations are too far apart, the drones must be equipped with heavier battery packs to cover a longer range, which adds costs to the fleet equipment and maintenance activities. Like the depot location decision, the choice of battery capacity for the fleet is a system design question that should be determined early on, because modifying the battery design may involve making changes to the airframe and electrical system on each drone in the fleet, which can be very costly.

Here is how specifically the flight range of a drone, denoted by L, plays a role in the depot location decision. A delivery trip (type 1) that starts and ends at the same depot must have the demand point lie in a circular area of radius $L/2$ centered at the depot, while a delivery trip (type 2) that starts and ends at different depots must have the demand point lie in an elliptical area having the depots as foci and the major axis length of L. Clearly, a customer that is serviceable by a type 2 trip is also serviceable by a type 1 trip, but not vice versa. During service operations, a customer demand can be fulfilled only if the origin depot has the ordered item in stock and has an available drone to fly the delivery trip. Thus, everything else held equal, a customer location serviceable by a type 2 trip is twice as likely to have her demand fulfilled compared to a customer serviceable by only a type 1 trip. Ensuring all customer locations to be serviceable by a type 2 trip is therefore a good objective in the network design. It not only increases service coverage, but also enables a higher operational efficiency. For instance, it leaves more flexibility in drone-order matching in task assignments, and permits more choices for drone re-locationing after a delivery task, to respond more quickly to the next customer demand. The relationship between depot location and service level and flexibility is illustrated in Fig. 1. This particular consideration in drone delivery network design motivates the Elliptical Cover problem studied in this paper.

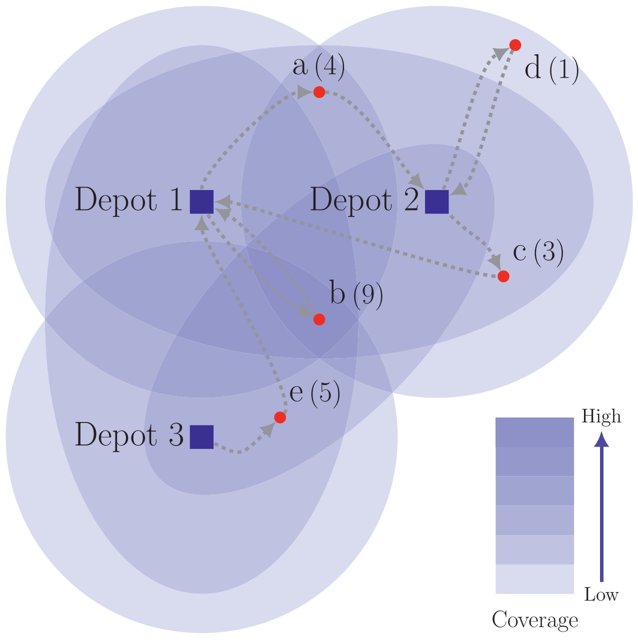

Fig. 1. A drone delivery service area having three depots and five customer locations. Customer e is serviceable by 5 delivery routes while customer d is serviceable by only one route. In this paper, we require all customers to be serviceable by at least 3 routes, i.e., covered by an ellipse.

1.3. Related problems and literature

Determining the location of service depots falls in the class of facility location problems (Aikens, 1985; Daskin, 1995; Klose & Drexl, 2005; Revelle & Laporte, 1996), and more specifically the continuous location problems (Berman, Drezner, Tamir, & Wesolowsky, 2009; Blanco, Puerto, & Ben-Ali, 2014; 2016; Brimberg & Salhi, 2005; Drezner & Brimberg, 2020; Rosing, 1992). Covering problems in facility location have been surveyed by Schilling, Jayaraman, and Barkhi (1993) and Farahani, Asgari, Heidari, Hosseininia, and Goh (2012).

The problem of covering demand points on a plane by ellipsis-like shapes has been investigated in recent literature. Canbolat and Massow (2009) studied the problem of selecting k out of m axis-parallel ellipses of given sizes and costs and determining their center locations to cover a subset of demand points on the plane in the most economical way. The authors presented a MINLP formulation which was shown to be difficult to solve even for small instances, and proposed an algorithm based on Simulated Annealing for practical solutions. Andretta and Birgin (2013) investigated the same problem and proposed an enumeration approach for obtaining its global solution. In this approach, each plausible ellipsis-demand matching pattern was examined whereas the feasibility of the matching was checked by solving a convex nonlinear program. A tree-like data structure was employed to organize the search and to skip unpromising candidates. However, when the ellipses were permitted to be oriented freely (i.e., not fixed to axis-parallel orientation as studied in Canbolat and Massow (2009)), the feasibility subproblem became nonconvex and thus more difficult to solve. On the same problems, Tedeschi and Andretta (2021) further proved that the number of possible locations of the m ellipses that may appear in the optimal solution to the problem with n demand points is upper bounded by n^{2m} and n^{3m}, when the ellipses have fixed and flexible orientations, respectively. The authors then developed enumeration algorithms with several algorithmic improvements to find the optimal solutions faster than those in the prior work, i.e., Andretta and Birgin (2013). There are several notable differences between the problem studied in the above-mentioned literature and the one presented in this paper. First, the former problem seeks a maximal (profit) coverage solution while our problem seeks a full coverage solution; second, the number and shapes (in terms of the semi-major and semi-minor axis lengths) of the covering ellipses are given in the former problem, whereas in our problem neither the number nor the shapes of the covering ellipses is known *a priori* as they depend on the location of the foci points. As a result, the geometric analysis approach that was useful for enumerating the possible solutions of the former problem (similar ideas were also found in Church (1984), Drezner (1984) and Chazelle & Lee (1986)) is not applicable (at least to the author's knowledge) to the Elliptical Cover problem studied in this paper.

Blanco and Puerto (2021) investigated the problem of determining the location of p depots to cover a finite set of demand points so that the largest weighted sum of the distances from a demand point to all depots is minimized. The authors termed this problem Minimum Radius Enclosing Polyellipsoid Problem with Given Foci (and short for MEP). Using the same notations in (1), the MEP problem can be mathematically stated as follows.

$$\underset{X}{\text{minimize}} \left\{ \max_{1 \le i \le n} \sum_{j=1}^{p} w_j D_i(X + X'_j) \right\} \tag{7}$$

where for each $j \in \{1, \ldots, p\}$, X'_j is a given coordinate associated with depot j and $w_j \ge 0$ is a given weight associated with depot j, with $\sum_{j=1}^{p} w_j = 1$[1]. Compared to the Elliptical Cover problem of (1), the MEP problem does not involve the assignment of demand points to nearest depots by the minimum distance principle (i.e., the innermost min operator in (1) is forgone), and thus it can be formulated as a continuous optimization problem. The authors proved that the problem is solvable in polynomial time and developed a solution algorithm with time complexity $O(n^{d+2})$, where d is the fixed data dimension (on a plane, $d = 2$). The idea was to decompose the task of solving the original problem into solving a polynomial number of smaller (convex optimization) problems. On large instances (i.e., potentially to arise in data mining contexts), the decomposition approach was demonstrated to run faster than Gurobi solving a second order cone (SOC) formulation of the problem. A similar decomposition approach also underlies our proposed method for solving the Elliptical Cover problem, but the number of possible subproblem to solve is indefinite. In brief, we will solve (7) using a local solver (i.e., CONOPT, since the problem is convex) in a locate-allocate algorithm for obtaining local solutions to the Elliptical Cover problem. Note that the decomposition approach proposed by Blanco and Puerto (2021) could potentially be used for expediting the solution of (7) as a subproblem in our context. However, given that the locate-allocate algorithm usually take a limited number of iterations to converge (see Fig. 11), in this paper we adopt an off-the-shelf solver for simplicity.

Another related problem is the α-neighbor p-center problem on a Euclidean plane (Chen & Chen, 2013), in which the goal is to cover the demand points with p circles such that each demand point is covered by at least α circles and that the radius of the largest circle is minimized. This problem permits a mathematical formulation similar to (2) to (6) - specifically, rewrite constraint (3) as

$$L \ge z_{ij}\left[(x_j - a_i)^2 + (y_j - b_i)^2\right]^{1/2}, \quad \text{for } i = 1, \ldots, n;\ j = 1, \ldots, p$$

and one can obtain the MINLP formulation for the k-neighbor p-center problem. For the same set of demand points and a given p, the optimal objective value of the 2-neighbor p-center problem is clearly a lower bound for the Elliptical Cover problem. However, there is no known polynomial algorithm for globally solving the former problem. Khuller, Pless, and Sussmann (1997) provided a polynomial time approximation algorithm for the problem defined on a graph and achieved an approximation factor of 2 for $\alpha < 4$, but an approximate solution can not serve as a lower bound. The best known exact algorithm is proposed by Chen and Chen (2013). The idea is solving a finite series of set covering problems via integer programming, similar to those proposed by Minieka (1970) and Drezner (1984) for solving the EPC problem. Numeric experiments performed by the authors showed that the optimal solution to the 2-neighbor p-center problem often coincides with the optimal solution to the EPC problem with $2p$ centers - by placing two centers at the same location to meet the 2-neighbor requirement. This observation indicates that the solution to the 2-neighbor p-center problem is unlikely to be a valuable starting point (that is worth the effort of obtaining it) for the locate-allocate algorithm to be proposed in Section 3 for solving the p-Elliptical Cover problem.

Overall, to our knowledge, the Elliptical Cover problem has not been studied in the facility location literature. Our contributions can be summarized to include: (1) a mixed integer nonlinear formulation of the problem, (2) a proof that the problem is $\mathcal{NP}$-hard, (3) an adaptation of a well-known locate-allocate algorithm to locally solve the problem with both discrete and continuous demand sets, (4) an examination of a one-dimensional variant of the problem (the shortest covering interval (SCI) problem), an exact algorithm to solve this problem, which then provides a lower bound for the Elliptical Cover problem, and (5) computational results and relevant discussions.

The remainder of the paper is organized as follows. In Section 2, we prove that the Elliptical Cover problem is $\mathcal{NP}$-hard, and present the unique challenges it poses to analytical approaches that have found success on similar problems, such as the Euclidean p-center problem. In Section 3, we propose a locate-allocate algorithm, along with its convergence proof, for finding locally optimal solutions. In Section 4, we formulate the one-dimensional version of the Elliptical Cover problem whose solution will provide a valid lower bound to the original problem. We furthermore show several useful insights into this problem, and develop an exact and a heuristic algorithm, respectively, for solving it much faster than the mathematical optimization approach. Section 5 extends the locate-allocate approach to cover an area, instead of discrete points, with ellipses. All proposed algorithms are validated in numeric experiments in Section 6, where we also present a comparison between the Elliptical Cover and the EPC solutions in the drone delivery network design context. Finally, Section 7 concludes the paper with pointers for future research.

2. Complexity analysis

Circle Covering: Given n unit circles in the plane and an integer $p > 0$, decide whether there exist p points such that each circle contains at least one point (we say that a circle contains a point if the point lies on, or in the interior of, the circle).

Megiddo and Supowit (1984) proved the $\mathcal{NP}$-hardness of the Euclidean p-center problem by first noting its equivalence to Circle Covering, then reducing 3-satisfiability (Garey & Johnson, 1978) to Circle Covering, thereby proving that Circle Covering is $\mathcal{NP}$-complete. To prove the $\mathcal{NP}$-hardness of (1) (and more generally, of problem (2) - (6)), we will reduce Circle Covering to another decision problem, called Concentric k-Circle Covering, the optimization problem counterpart to which is equivalent to (1) when $k = 2$.

First, let us define the concept of the *ring* used in the subsequent problem description. We say that k concentric circles form k ring areas (or rings) defined as follows: points enclosed by the

[1] The authors discussed the problem in a general d dimensional space catering to applications in data science, where we set $d = 2$ in formulation (7) for the ease of comparison with (1) for planar applications.

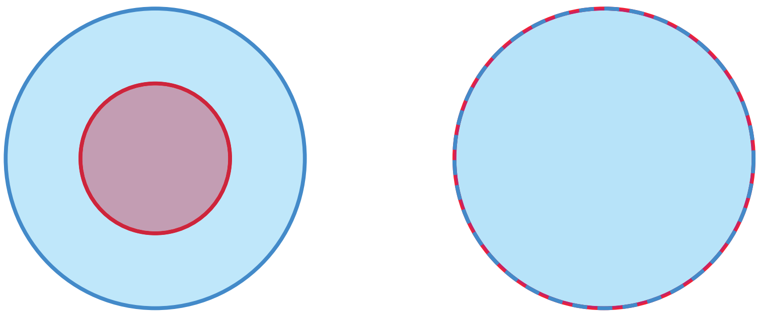

Fig. 2. Illustration of concentric rings. Left: two concentric circles of different diameters form two rings: the first ring is the red circular area, and the second ring is the blue are. Right: two concentric circle of equal diameter also form two rings, but only one ring (blue) includes the interior area, while the other ring (red) is simply the boundary circle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

smallest circle (including points that lie on the circle) are said to be in the first ring, points enclosed by the second smallest circle but outside the interior of the first ring are said to be in the second ring, and so forth. If two (or more) circles have the same radius, an arbitrary rank order is assigned to them so that points in the common interior are said to be in the smallest rank-ordered ring and the points that lie on the circles are said to be in both (all) rings. An illustration is given in Fig. 2.

Concentric k-Ring Covering: Given n points on the plane, an integer $k > 0$ and an integer $p > 0$, decide whether there exist p points, and n groups of circles arranged in such a way that each group has k concentric circles centered at one of the n given points (no two groups share the same center) and the sum of the radii of the k circles is no more than k, such that each of the nk rings formed by the nk circles contains at least one of the p points.

Proposition 1. *Concentric k-Ring Covering is $\mathcal{NP}$-complete.*

Proof. Given an instance, say n points $\{c_i, \dots, c_n\}$, and a proposed solution, say p points $\{X_1, \dots, X_p\}$, we can check whether the proposed solution constitutes a "yes" answer to the instance as follows: calculate $d(c_i, X_j)$ for each $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, p\}$, then for each i rank the distances $\{d(c_i, X_1), \dots, d(c_i, X_p)\}$ and calculate the sum of the smallest k members in the set and denote the sum by d_i. The "yes" condition is satisfied if and only if $d_i \le k$ for all $i \in \{1, \dots, n\}$. The overall time complexity is bounded by $O(n \cdot p \cdot \log(k))$. Therefore, the problem is in $\mathcal{NP}$.

We now reduce Circle Covering to Concentric k-Ring Covering. For each instance of Circle Covering, say n unit circles $\{C_1, \dots, C_n\}$ centered at points $\{c_1, \dots, c_n\}$ respectively, we can construct an instance of the Concentric k-Ring Covering for a given k by using the same given n points and the same p. Given a solution to the former instance we can construct a solution to the latter instance. Specifically, if C_i contains a point X, we can choose the radii of the k circles centered at c_i to be $d(c_i, X)$, thus all k rings contain X and the sum of the radii is no more than k (because $d(c_i, X) \le 1$). On the other hand, a solution to the Concentric k-Ring Covering instance also leads to a solution to the Circle Covering instance. Specifically, the knowledge that each of the k rings centered at c_i contains at least one of the p points indicates that the smallest ring (i.e., the inner most circular area) centered at c_i contains one of the points. Also, the radius of the smallest ring cannot exceed 1 (the unit length) since the sum of the k radii is no more than k. Therefore, the unit circle C_i must contain at least one of the p points. Since Circle Covering is known to be $\mathcal{NP}$-complete and the transformation is polynomial, the conclusion follows. □

Concentric k-Ring Covering is a decision problem counterpart to the optimization problem (2) to (6), in the same way as Circle Covering being a counterpart to EPC. The optimization problem is at least as hard as the decision problem, hence is $\mathcal{NP}$-hard. In

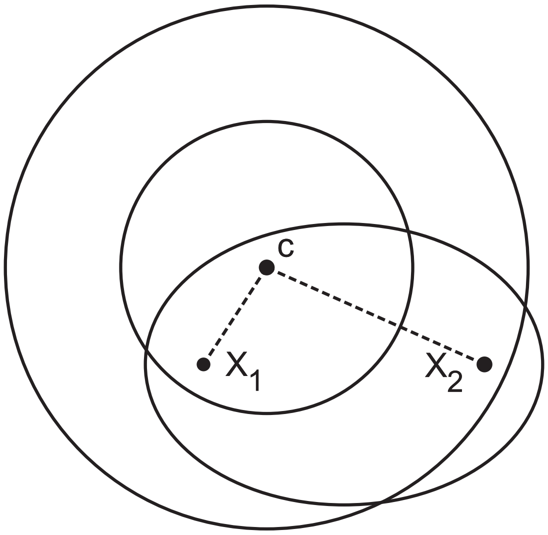

Fig. 3. The Concentric 2-ring problem is equivalent to the decision problem of Elliptical Cover. If both rings centered at c contain a point, X_1 and X_2, respectively, then c is covered by the ellipse with X_1 and X_2 as foci; and vice versa.

this paper, we focus on the problem with $k = 2$ for its real-world application. This problem permits a geometric description: find p depots such that each demand point is covered by a number of ellipses each having some pair of depots as foci and the major axis of the largest ellipse used in the coverage is minimized. Therefore, it is termed as the Elliptical Cover problem. Its equivalence to the Concentric 2-Ring Covering problem is illustrated in Fig. 3.

Globally solving the optimization problem (2) to (6) is extremely difficult even for small instances. When a global MINLP solver, such as BARON (Sahinidis, 1996), is used to solve an instance, the lower bound starts at a useless level (i.e., 0) and improves extremely slowly. Indeed, relaxing the integrality of the binary variables would not make the problem convex. Though one could reformulate the problem (e.g., use the binary variable $z_{ijj'}$ to indicate if demand i is assigned to the ellipse of foci j and j') to avoid the bilinear terms $z_{ij}D_i(X_j)$, the use of big M constants would substantially weaken the reformulation, and the presence of the Euclidean distance calculation, e.g., in $D_i(X_j)$, would still call for the use of a global MINLP solver rather than an MIP solver. Moreover, the isomorphism in the problem's graph structure creates excessive symmetry that is inexpressible in algebraic formulations. For instance, for a problem with $n = 5$ and $p = 3$, the following three demand-depot assignments,

Assignment 1: $z_{ij} = 1$ for $\{i \in \{1, 2\}, j \in \{1, 2\}\}$ or $\{i \in \{3, 4, 5\}, j \in \{2, 3\}\}$

Assignment 2: $z_{ij} = 1$ for $\{i \in \{1,2\}, j \in \{1,3\}\}$ or $\{i \in \{3,4,5\}, j \in \{2,3\}\}$
Assignment 3: $z_{ij} = 1$ for $\{i \in \{1,2\}, j \in \{2,3\}\}$ or $\{i \in \{3,4,5\}, j \in \{1,3\}\}$

are equivalent for the geometric problem, but they are (unnecessarily) distinguished in the algebraic formulation. Eliminating one would not automatically eliminate the other two in the branch and bound framework, causing redundant computations.

The same issue would exist in the mathematical formulation of other geometric location problems such as the EPC problem. Consequently, successful algorithms invariably exploited some geometric and combinatorial insights into the problem structure, rather than relying on mathematical optimization approaches. For instance, Drezner (1984) uncovered two geometric properties of the EPC's optimal solution which ultimately led to good algorithms. One is the fact that the largest circle in the solution is defined by at most three demand points hence there are $O(n^3)$ possible radii with one of them being the solution radius. The other is that for a given radius there are only $O(n^2)$ possible locations where the p circles can center at. Using these insights, the author developed an exact algorithm for the EPC by solving a set-covering problem polynomial number of times. Vijay (1985) proposed another exact algorithm for EPC based on similar principles and was able to generate the set-covering subproblems more efficiently. Hwang, Lee, and Chang (1993) further improved the time bound for EPC to $O(n^{O(\sqrt{p})})$, by inventing a new algorithm to solve the circle-covering subproblem (which is at the bottleneck of the EPC algorithm, and is stated as "given a set of n demand points and two parameters p and r, determine whether there exist p circles of radius r which can cover all demand points") in time $O(n^{O(\sqrt{p})})$. Callaghan, Salhi, and Nagy (2017) devised a series of means, including exploiting early termination opportunities in subproblem solution and eliminating unpromising candidate solutions on the fly, to speed up execution of the EPC algorithm proposed in Drezner (1984).

3. A locate-allocate algorithm that converges

While the EPC is a special case ($k = 1$) of the problem (2) to (6), none of the geometric insights that sped up the EPC solution is applicable to the general cases of $2 \le k < p$. The key reason is this: a given demand-depot allocation (i.e., fixing values for z_{ij}) would decompose the EPC problem into p separate 1-center problems of the same nature, but would not do the same decomposition to the general case (i.e., $k > 1$) - the location variables for all the p points would still be coupled in a single nonlinear program (NLP). Fortunately, this NLP is convex.

Lemma 2. *With variables z_{ij}, $i \in \{1,\dots,n\}$ and $j \in \{1,\dots,p\}$, fixed to any binary assignment $\{0,1\}^{n\times p}$, the problem (2) to (6) is reduced to a convex optimization problem.*

Proof. Let $J(i) := \{j : z_{ij} = 1\}$, then the problem becomes

$$\underset{L,(x_j,y_j),\, j=1,\dots,p}{\text{minimize}} \quad L \tag{8}$$

$$\text{s.t.} \quad L \ge \sum_{j\in J(i)} [(x_j - a_i)^2 + (y_j - b_i)^2]^{1/2} \qquad \text{for } i = 1,\dots,n \tag{9}$$

The right side of constraint (9) is a convex function of the variables (x, y) on $\mathbb{R}^{2p}$ because each summand (being an L_2 norm) is convex, and the inequality (for each i) describes the epigraph of that convex function and hence forms a convex set on $\mathbb{R}^{2p+1}$. Minimizing a linear function over the intersection of n convex sets is a convex optimization problem. □

In fact, the problem (8) - (9) is a second order cone problem (SOCP), a type of convex optimization problem solvable by interior-point methods in polynomial time, see, e.g., Nesterov and Nemirovskii (1994).

The original problem can be viewed as a combinatorial assignment problem, i.e., assigning each demand point to two depots, in which the performance of a candidate assignment must be assessed via solving a convex NLP. There are $\binom{p}{k}^n$ possible assignments, an exponential function of inputs unless $k = p$. To see the complexity of enumerating all unique assignments, let us make such an attempt on an instance with $n = 10$, $p = 4$ and $k = 2$; that is, to cover 10 demand points by a collection of ellipses whose foci are from a set of four depots. Each pair of depots can serve as the foci for an ellipse, so we can think of the covering ellipses as edges in a graph in which depots are unlabeled nodes. There are ten graphs with a non-empty edge set on four unlabeled nodes, illustrated in Fig. 4.

Assigning n demand points to a number, say h, of covering ellipses (i.e., edges in the graph) is equivalent to partitioning n labeled objects into h nonempty unlabeled subsets, and the number of ways to do so is the Stirling number of the second kind, denoted by $S(n, h)$ and calculated by

$$S(n,h) = \frac{1}{h!}\sum_{i=0}^{h}(-1)^i\binom{h}{i}(h-i)^n$$

Among the ten possible arrangements of covering ellipse(s), one has $h = 1$, two $h = 2$, three $h = 3$, two $h = 4$, one $h = 5$ and one $h = 6$, so the total number of unique demand-depot assignments comes down to

$$S(10,1) + 2\cdot S(10,2) + 3\cdot S(10,3) + 2\cdot S(10,4) + S(10,5) + S(10,6) = 162575.$$

Even though this number is two orders of magnitude smaller than $\binom{4}{2}^{10} = 60466176$, the size of the search space modeled by the algebraic constraints (4) and (5), it is still impractical to evaluate such a large number of possibilities, whereas the evaluation of each possibility would require solving a nonlinear, though convex, optimization problem.

We propose the following iterative locate-allocate algorithm for approximately solving the Elliptical Cover problem. Let $\epsilon > 0$ be a small number.

Locate-Allocate Algorithm:

```
Step 1: Initialize $L^* = \infty$ and randomly sample $p$ depot locations on the plane, $X_1, \dots, X_p$.
Step 2: For each demand point $i$, compute the distances $D_i(X_j)$, $j = 1,\dots,p$ and let $J(i) = \{j_i, j_i'\}$ where $j_i = \text{argmin}_{\{1,\dots,p\}} D_i(X_j)$ and $j_i' = \text{argmin}_{\{1,\dots,p\}\setminus\{j_i\}} D_i(X_j)$.
Step 3: Solve the problem (8) - (9) using an NLP solver, to obtain the optimal value $\hat{L}$ and optimal solution $\hat{X}_j$, $j = 1,\dots,p$.
Step 4: If $\hat{L} - L^* < -\epsilon$, stop and return $\{\hat{X}_1,\dots,\hat{X}_p\}$ as solution; otherwise, update $L^* \leftarrow \hat{L}$, $\{X_1,\dots,X_p\} \leftarrow \{\hat{X}_1,\dots,\hat{X}_p\}$, go to Step 2.
```

Proposition 3. *The locate-allocate algorithm will terminate in finite iterations.*

Proof. There are only a finite number of assignments $\{J(1),\dots,J(n)\}$, and each assignment gives a definite optimal value L due to convexity. Each iteration produces a new assignment whose optimal value L is strictly smaller than that of the previous iteration. Since the algorithm iterates over a finite set, some assignment value must eventually be revisited. The revisit cycle length (i.e., number of iterations between the 1st and 2nd visit to the same assignment value) must be 1 and the algorithm

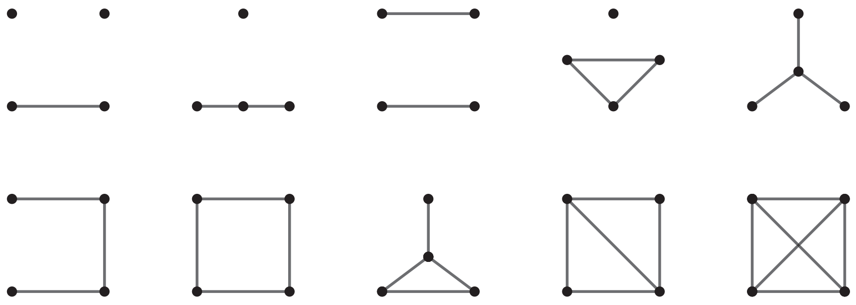
Fig. 4. Ten ways to form covering ellipses whose foci are chosen from a set of four depots.

must terminate when the revisit happens, because any other assignment visited between the 1st and 2nd visit would give an objective value strictly greater and strictly smaller than the objective value of the revisited assignment, hence a contradiction. □

The idea of alternate location-allocation is not new. Cooper (1964) proposed an iterative method employing this idea to solve the multisource Weber Problem (Brimberg, Hansen, Mladenović, & Taillard, 2000; Hansen, Mladenović, & Taillard, 1998; Raeisi Dehkordi, 2019; Wesolowsky, 1993), which concerns finding the coordinates of p depots from a continuous feasible space to minimize the sum of weighted distances from a finite set of destinations to their closest depot. The same idea underpinned the Voronoi heuristic method proposed by Suzuki and Drezner (1996) to solve the p center problem in an area, as well as the K-means clustering algorithm widely used in machine learning.

Each run of the algorithm, starting from a random starting point (i.e., p depot locations), will produce a "local optimum", in the sense that unilaterally changing either the location or the allocation would not lead to a better objective value. However, there are indefinitely many (upper bounded by the number of unique assignments) local optima. The multistart search strategy, which involves running the algorithm many times (optionally in parallel), can be employed to generate a pool of local optima, the best of which is then used as the final solution. The starting point strategy proposed in Drezner and Brimberg (2020) and the transfer follow-up step proposed in Drezner, Brimberg, Mladenović, and Salhi (2015) for expediting the locate-allocate algorithm and improving the local solution for the planar p-median problem can potentially be applied here, though we have not tested them in our implementation. Our computational experience has shown that the number of unique local optima identified via multistart is typically very limited even after a large number of runs. In other words, the chance that a better solution turns up decays exponentially as more and more runs are executed. In practice, if no better solution turns up in the last, say, 20 runs, then the search can stop and the best solution identified thus far is to be accepted.

4. The one-dimensional variant and its solution algorithms

There is no polynomial-time algorithm (unless $\mathcal{P}=\mathcal{NP}$) to compute and systematically improve the lower bound, hence to establish global optimality. Computing the lower bound involves solving a relaxed version of the problem, such as one that relaxes the integrality of z_{ij} in the MINLP formulation. However, as discussed in Section 2, this kind of relaxation is either still difficult to solve or too weak to be valuable. As mentioned in Section 1.3, the 2-neighbor p-center problem (Chen & Chen, 2013) can be viewed as a relaxation of the Elliptical Cover problem, and hence its optimal objective value can serve as a lower bound. However, solving that problem to global optimality is by itself an $\mathcal{NP}$-hard task, which does not simplify the task at hand.

We propose a dimension relaxation approach. The idea is based on this fact: if the demand points on the plane are enclosed by a number of ellipses, then the projection of the demand points onto any straight line on the plane must also be enclosed by the projection (line segments) of those ellipses onto the same line. Furthermore, the length of the projected line segment of an ellipse is smaller than or equal to the length of the major axis of the ellipse, which is in turn smaller than or equal to the distance between the foci of the ellipse. This means that for any given straight line, we can solve an alternative covering problem on the line, the optimal value of which will serve as a lower bound of the original problem. On top of this, some search procedure can be employed to find the straight line such that the resulting lower bound is greatest. The idea is illustrated in Fig. 5, and the alternative covering problem, called Shortest Covering Interval (SCI) problem, is stated as follows.

Shortest Covering Interval (SCI) problem: given the location of $n \geq 1$ demand points on the real line and an integer $p \geq 2$, determine the location of p depots on the line such that the maximum sum of distances from a demand point to its two nearest depots is minimized.

Apart from providing lower bounds to the Elliptical Cover problem, the SCI problem has its own applications. For instance, UAVs have been used to inspect power transmission lines and railroads. Locating battery charging depots along such infrastructure lines to cover points of interest along the lines can be formulated as a SCI problem. Therefore, it is important to understand the properties of the SCI problem and develop effective algorithms for solving it.

4.1. An MIP formulation for the SCI problem

As a one-dimensional special case of the Elliptical Cover problem, the SCI problem has the same MINLP formulation of (2) to (6) with the terms $(y_j - b_i)^2$ in constraint (3) removed. Moreover, it also permits a linear mixed integer programming (MIP) formulation as follows.

$$\begin{array}{ll}
\text{minimize} & L \\
\text{s.t.} & L \geq |x_j - a_i| + |x_{j'} - a_i| - (1 - z_{ijj'}) \\
M, & \text{for } 1 \leq i \leq n,\ 1 \leq j < j' \leq p \\
\displaystyle\sum_{j=1}^{p-1} \sum_{j'=j+1}^{p} z_{ijj'} = 1, & \text{for } 1 \leq i \leq n \\
z_{ijj'} \in \{0, 1\}, & \text{for } 1 \leq i \leq n,\ 1 \leq j < j' \leq p
\end{array}$$

where M is a big positive number. Letting $M = 2(a_n - a_1)$ is valid and most reasonable. The first constraint involving absolute functions can be expanded to four linear inequalities. The MIP model is much more amenable than the MINLP formulation. In Section 6, we will use this model to benchmark the more efficient solution method for SCI presented in the following section.

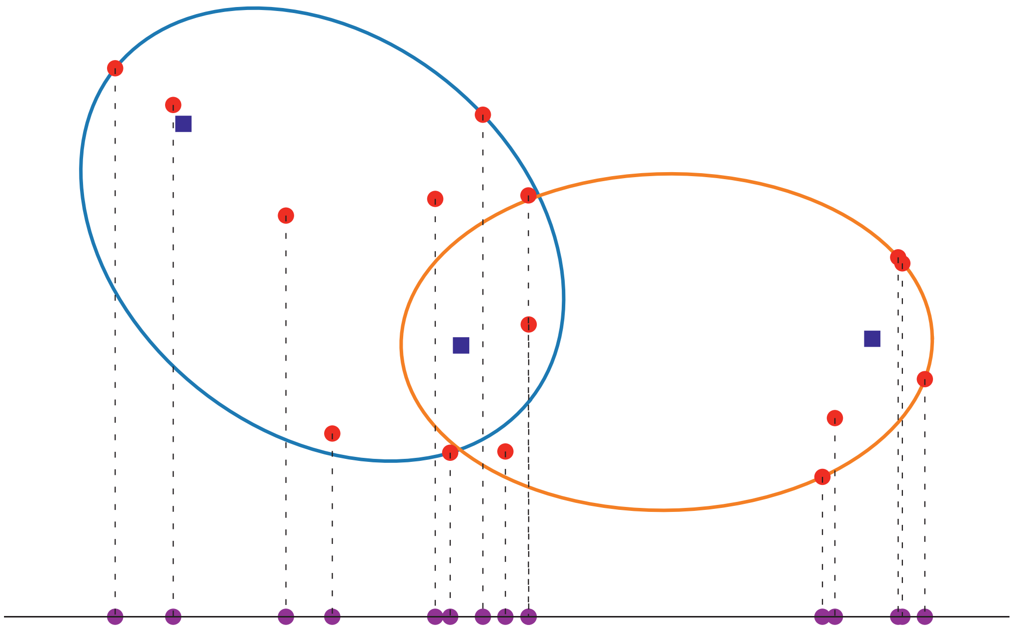
Fig. 5. Lower bounding the Elliptical Cover Problem by solving the SCI problem.

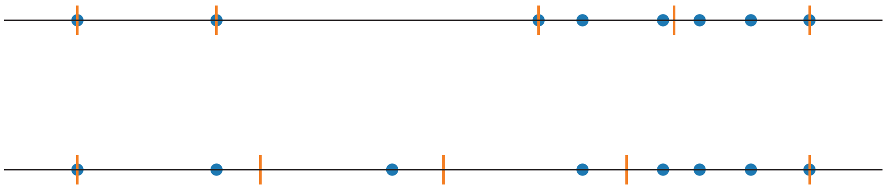
Fig. 6. Two instances, along with their optimal solutions, of the SCI problem with $n = 8$ and $p = 5$. Demand points are solid dots and depots are vertical bars. The upper solution has two clusters and the lower solution has one cluster.

4.2. Exact and heuristic algorithms for the SCI problem

The location of p depots that minimizes the objective value (i.e., the maximum sum of distances from a demand point to its two nearest depots) is called the *optimal solution* and the objective value at the optimal solution is called the *optimal value*. Let $\mathcal{A} := \{a_1, \dots, a_n\}$ be a set of n real numbers (without loss of generality, assume that the numbers are arranged in ascending order), then an instance of the problem can be denoted as SCI($\mathcal{A}$, p). Furthermore, let us denote the optimal value of SCI($\mathcal{A}$, p) as $\nu(\mathcal{A}, p)$.

We can think of the demands and depots as two sets of vertices of a graph, in which an edge connecting a demand and a depot is present if the depot is either closest or second closest to the demand point among all depots. In this way, any location of p depots for an SCI($\mathcal{A}$, p) instance will correspond to a bipartite graph. Such a graph may consist of one or multiple connected subgraphs. In this problem context, we call each connected subgraph a *cluster*. The notion of cluster is illustrated in Fig. 6.

Proposition 4. *For a given set of real numbers* $\mathcal{A}$, $\nu(\mathcal{A}, p) \geq \nu(\mathcal{A}, p+1)$, *for any* $p \geq 2$.

Proof. Given an instance SCI($\mathcal{A}$, p) and its optimal solution, we can place an extra depot at the same location of any existing depot, to construct an instance of SCI($\mathcal{A}$, $p+1$) along with a feasible solution having objective value $\nu(\mathcal{A}, p)$. Since it is an arbitrary solution, the objective value cannot exceed the optimal value, which is by definition equal to $\nu(\mathcal{A}, p+1)$. □

Proposition 5. *If we knew that the optimal solution to SCI($\mathcal{A}$, p) would have $m \geq 1$ clusters which would partition $\mathcal{A}$ into $\mathcal{A}_1, \dots, \mathcal{A}_m$, then there must exist m integers, $p_1, \dots, p_m$, that satisfy the following three conditions: (1) $p_j \geq 2$ for $j = 1, \dots, m$; (2) $\sum_{j=1}^m p_j = p$, and (3)*

$$\nu(\mathcal{A}, p) = \max_{1 \leq j \leq m} \{\nu(\mathcal{A}_j, p_j)\}$$

Proof. When $m = 1$, then we must have $p_1 = p$ and the conclusion follows. Suppose $m > 1$. Given that an optimal solution to SCI($\mathcal{A}$, p) consists of m clusters, by the definition of cluster, the p depots in this optimal solution can be partitioned into m subsets, such that each subset j, $j = 1, \dots, m$, consists of $p'_j \geq 2$ depots serving the demand set $\mathcal{A}_j$. Conditions (1) and (2) are clearly satisfied. Moreover, this set of integers, i.e., $\{p'_j\}$, $j = 1, \dots, m$, must also satisfy condition (3). If not, then either $\nu(\mathcal{A}, p) < \max_j\{\nu(\mathcal{A}_j, p'_j)\}$ or $\nu(\mathcal{A}, p) > \max_j\{\nu(\mathcal{A}_j, p'_j)\}$. The former case means that for some cluster, say j', the best objective value of using $p_{j'}$ depots to cover $\mathcal{A}_{j'}$ is greater than the optimal value $\nu(\mathcal{A}, p)$, which contradicts the fact that $(\mathcal{A}_{j'}, p_{j'})$ is a cluster in the optimal solution. The latter case is also impossible, because it implies that an objective value smaller than $\nu(\mathcal{A}, p)$ can be achieved given the cluster assignment at the optimal solution. □

We call an optimal solution to the SCI problem a *regular solution* if no demand point has both of its nearest depots fall strictly on the same side of it along the line. At a regular solution, each demand point is "covered" by the interval formed by its two nearest depots. In terms of finding a regular solution, the SCI problem is equivalent to determining a set of intervals formed by p cut points so as to minimize the length of the longest demand-covering interval.

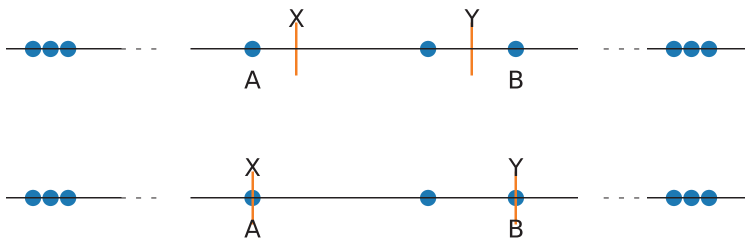

Fig. 7. Upper: The two nearest depots, X and Y, to the demand point A both fall on the right side of A. This is not a regular solution, and can be changed to a regular solution (Lower) without affecting the objective value.

Proposition 6. *Let $\mathcal{A} = \{a_1, \ldots, a_n\}$ be a set of real numbers arranged in ascending order, i.e., $a_i < a_j$ whenever $i < j$, and $p \geq 2$ be a given integer. The following statements are true for SCI($\mathcal{A}$, p).*

1. *There exists a regular solution.*
2. *At a regular solution, one depot must be located at a_1 and another depot must be located at a_n.*
3. *Suppose we divide the interval $[a_1, a_n]$ into $p-1$ equal-length sub-intervals, $[a_1, c_1]$, $[c_1, c_2], \ldots, [c_{p-3}, c_{p-2}]$, $[c_{p-2}, a_n]$, and say that a sub-interval is empty if it does not contain any element of $\mathcal{A}$ in its interior (i.e., the open interval between the two end points), either of these two cases will ensue:*
   (a) *If none of the sub-intervals is empty, then the cut points $a_1, c_1, \ldots, c_{p-2}, a_n$ constitute a regular solution, and the optimal value is equal to the length of the sub-interval, i.e., $\nu(\mathcal{A}, p) = (a_n - a_1)/(p-1)$.*
   (b) *If some sub-interval(s) is empty, then any optimal solution must consist of more than one clusters, with optimal value smaller than $(a_n - a_1)/(p-1)$.*

Proof. Let us first define a concept: if the sum of distances from a demand point to its two nearest depots is equal to the objective value, this demand point is called a *binding point* at the solution.

For 1: We will show that if an optimal solution does not satisfy the regularity condition, we can always find another optimal solution that does. Specifically, suppose that we are at an optimal solution with optimal value ν, and demand point A finds both its nearest and second nearest depots, denoted by X and Y, respectively, on its right side, see Fig. 7 (upper part) for illustration. If A is not a binding point, then moving depot X to A can cause an increase in the objective value only if there is a demand point, say B, on the left of Y whose second nearest depot is X, which will become binding as X moves leftward before reaching A. (Note that if such a point B does not exist, then we can move X to A without affecting the objective value and the proof is done.) Let us move X leftward until point B becomes binding, that is $\overline{BY} + \overline{BX} = \nu$. Now let us move Y rightward until point A becomes binding, that is $\overline{AX} + \overline{AY} = \nu$. At this point, Y must still be on the left of B because X is on the right of A. In fact, the above two equations together with the equation that $\overline{AB} = \overline{AX} + \overline{XB} = \overline{AY} + \overline{YB}$ indicates that $\overline{AX} = \overline{BY}$. Now if we simultaneously move X leftward to A and move Y rightward to B, we will still have $\overline{AX} + \overline{AY} = \overline{BY} + \overline{BX} = \nu$. Any demand point that lies between X and Y will never become binding in the process. On the other hand, if A is a binding point in the first place, we can simultaneously move X leftward and move Y rightward by the same distance until X reaches A. For the same reason above, such a move will not cause an increase in the objective value. The same process can be applied to any other demand point having both of its nearest depots fall on the same side of it. By the end, we will have created an optimal solution that satisfies the stated condition.

For 2: If the leftmost depot is located on the right side of a_1, then a_1 must have both of its nearest depots fall on its right side, and the solution cannot be a regular solution, a contradiction. Suppose the leftmost depot is located on the left side of a_1, if a_1 is a binding point, then moving the leftmost depot rightward would reduce the objective value, leading to a contradiction; if a_1 is not a binding point, suppose a_j is a binding point, then simultaneously moving all depots on the left of a_j rightward for the same (small) distance would reduce the objective value, leading to a contradiction. So the left most depot must be located exactly at a_1. The same arguments apply to the case of a_n.

For 3: Claim 1 and 2 imply that any regular solution takes the form of dividing the interval $[a_1, a_n]$ into $p-1$ sub-intervals (zero-length sub-interval is allowed), and the objective value is equal to the length of the longest non-empty sub-interval. If we knew a priori that there is a regular solution in which all the $p-1$ sub-intervals are non-empty, then the best solution having this property is the one that equally divides the interval $[a_1, a_n]$ into $p-1$ segments so the longest segment is minimized. This proves case (a). A better solution (i.e., one that gives an objective value smaller than $(a_n - a_1)/(p-1)$) is possible only if at least one of the $p-1$ sub-intervals has an empty interior (i.e., no element of $\mathcal{A}$ takes the two end points of this interval as its two nearest depots) and simultaneously has a length greater than $(a_n - a_1)/(p-1)$, making room for other intervals to become shorter than $(a_n - a_1)/(p-1)$. By definition of cluster and regularity, the two end points of such an interval must belong to different clusters. This proves case (b). □

The above properties of the SCI problem suggest a recursive approach for solving the problem. Specifically, Proposition 6 Claim 3 says that either the equispaced depot location is optimal or the input points are separated into at least two clusters. As a corollary of Claim 3, if the input points are to be broken into two clusters, the break point can only occur between successive input points that are more than $(a_n - a_1)/(p-1)$ apart. There are at most $\min\{n-1, p-2\}$ such break points. Given the cluster composition, Proposition 5 then promises the existence of a specific allocation of the p depots to clusters so that when each cluster's SCI problem is solved independently, the original problem is solved. Finally, Proposition 4 together with the condition (3) in Proposition 5 suggests that a greedy incremental allocation of depots to clusters will lead to the optimal allocation.

Algorithm 1 outlines a recursive function that returns the optimal value and optimal solution to the SCI instance given by ($\mathcal{A}$, p). In Line 2, the function Linspace returns p equispaced numbers spanning the interval $[a_1, a_n]$ as a candidate solution. Line 3 checks the condition in Proposition 6 Claim 3(a). If the condition is met, the solution is returned in Line 4; otherwise, Lines 6 - 7 identify all possible two-cluster break points and store them in $\mathcal{B}$. All two-cluster scenarios are evaluated and the best one will be picked and returned, whereas each scenario is evaluated in a recursive fashion. Specifically, in a given scenario k, each cluster j, $j = 1, 2$, is initialized with $p_j = 2$ depots located at the minimum and maximum elements of the cluster, respectively, and the cluster's optimal value ν_j is calculated. Line 11 calculates the number of remaining depots to be allocated to clusters. Lines 12 - 15 allocate the available

Algorithm 1 Exact Algorithm for the Shortest Covering Interval Problem.

```
Require: $\mathcal{A} := \{a_1, \dots, a_n\}$, sorted in ascending order, $p \geq 2$
 1: function SCI($\mathcal{A}$, $p$)
 2:     $\{c_1, \dots, c_p\} \leftarrow$ Linspace($a_1, a_n, p$), $\nu \leftarrow (a_n - a_1)/(p-1)$
 3:     if $\mathcal{A} \cap [c_j, c_{j+1}] \neq \emptyset$ for all $j = 1, \dots, p-1$ then
 4:         return $\left(\nu, \{c_1, \dots, c_p\}\right)$
 5:     end if
 6:     Initialize $\mathcal{B} \leftarrow \emptyset$, $\nu' \leftarrow \infty$, $C' \leftarrow \emptyset$
 7:     for $i = 1, \dots, n-1$ do
 8:         if $a_{i+1} - a_i > \nu$ then $\mathcal{B} \leftarrow \mathcal{B} \cup \{i\}$
 9:         end if
10:     end for
11:     for $k \in \mathcal{B}$ do
12:         $\mathcal{C}_1 \leftarrow \{a_1, \dots, a_k\}$, $\nu_1 \leftarrow (a_k - a_1)$, $C_1 \leftarrow \{a_1, a_k\}$, $p_1 \leftarrow 2$
13:         $\mathcal{C}_2 \leftarrow \{a_{k+1}, \dots, a_n\}$, $\nu_2 \leftarrow (a_n - a_{k+1})$, $C_2 \leftarrow \{a_{k+1}, a_n\}$, $p_2 \leftarrow 2$
14:         $f \leftarrow (p-4)$, $\nu^* \leftarrow \max\{\nu_1, \nu_2\}$
15:         while $f > 0$ do
16:             $j^* \leftarrow argmax\{\nu_1, \nu_2\}$, $\nu^* \leftarrow \max\{\nu_1, \nu_2\}$
17:             $p_{j^*} \leftarrow p_{j^*} + 1$, $f \leftarrow f - 1$
18:             $(\nu_{j^*}, C_{j^*}) \leftarrow$ SCI($\mathcal{C}_{j^*}, p_{j^*}$)
19:         end while
20:         if $\nu^* < \nu'$ then
21:             $\nu' \leftarrow \nu^*$, $C' \leftarrow C_1 \cup C_2$
22:         end if
23:     end for
24:     return $(\nu', C')$
25: end function
```

Algorithm 2 Heuristic Algorithm for the Shortest Covering Interval Problem.

```
Require: $\mathcal{A} := \{a_1, \dots, a_n\}$, sorted in ascending order, $p \geq 2$
 1: function SCI($\mathcal{A}$, $p$)
 2:     Generate equispaced cut points $\{c_1, \dots, c_p\} \leftarrow$ Linspace($a_1, a_n, p$)
 3:     Initialize cluster set $\mathcal{C} \leftarrow \emptyset$ and head index $h \leftarrow 1$
 4:     for $i = 1, \dots, p-1$ do
 5:         if $c_{i+1} < a_h$ then Continue
 6:         end if
 7:         if $\mathcal{A} \cap [c_i, c_{i+1}] = \emptyset$ then
 8:             $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{a \in \mathcal{A} \mid a_h \leq a < c_i\}\}$, $h \leftarrow \min\{1 \leq k \leq n \mid a_k > c_{i+1}\}$
 9:         end if
10:     end for
11:     Add the last cluster $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{a_h, \dots, a_n\}\}$
12:     if $|\mathcal{C}| = 1$ then return $((a_n - a_1)/(p-1), \{c_1, \dots, c_n\})$
13:     end if
14:     for $j = 1, \dots, |\mathcal{C}|$ do
15:         $\mathcal{C}_j \leftarrow$ j-th element of $\mathcal{C}$
16:         $\nu_j \leftarrow \max(\mathcal{C}_j) - \min(\mathcal{C}_j)$, $C_j \leftarrow \{\min(\mathcal{C}_j), \max(\mathcal{C}_j)\}$, $p_j \leftarrow 2$
17:     end for
18:     $f \leftarrow (p - 2|\mathcal{C}|)$, $\nu^* \leftarrow \max\{\nu_j \mid j = 1, \dots, |\mathcal{C}|\}$
19:     while $f > 0$ do
20:         $j^* \leftarrow argmax\{\nu_j \mid j = 1, \dots, |\mathcal{C}|\}$, $\nu^* \leftarrow \max\{\nu_j \mid j = 1, \dots, |\mathcal{C}|\}$
21:         $p_{j^*} \leftarrow p_{j^*} + 1$, $f \leftarrow f - 1$
22:         $(\nu_{j^*}, C_{j^*}) \leftarrow$ SCI($\mathcal{C}_{j^*}, p_{j^*}$)
23:     end while
24:     return $(\nu^*, \cup_{j=1}^{|\mathcal{C}|} C_j)$
25: end function
```

depots, one at a time, to whichever cluster that sets the objective value ν^* at the time, in order to further reduce ν^*. The optimal value and optimal solution are updated for the chosen cluster via a call to the SCI function. The ν^* value upon exiting the while loop is the best objective value achievable in the current two-cluster scenario. It is then compared to ν', the best value found so far (over all scenarios evaluated), to update the incumbent solution, in Lines 16 - 17. In essence, the algorithm performs an exhaustive search for the optimal two-cluster scenario, whereas each cluster in each scenario is subject to the same evaluation scheme via recursion. In this way, all credible clustering scenarios are evaluated. Therefore, the algorithm is guaranteed to return the optimal solution.

Algorithm 1 can become quite intractable when the set $\mathcal{B}$ is big, usually caused by a relatively large p compared to n. We propose a heuristic variant of the algorithm, Algorithm 2, which restricts the exploration of optimal clusters to those possibilities identified by the initial equispaced cut points. Lines 3 - 8 generates these candidate clusters, and lines 10 - 17 finds the best depot allocation among these candidates in the same recursive fashion as in Algorithm 1. Being a heuristic, Algorithm 2 is able to find the global solution in most cases within a fraction of the time taken for Algorithm 1. Their performance comparison is demonstrated in Section 6.

5. Extension for area coverage

Area coverage is a useful extension of Euclidean p-center problems. Instead of covering a finite set of demand points on the plane, the area coverage problem aims to cover all points in a given area, see Suzuki and Drezner (1996), Wei, Murray, and Xiao (2006) and Liu (2021) for algorithms based on Voronoi diagrams. The Elliptical Cover problem for an area is stated as follows: given an area R on the plane, find p depot locations X_j, $j = 1, \dots, p$, in order to

$$\text{minimize}_{X_1,\dots,X_p} \left\{ \max_{r \in R} \left\{ \min_{1 \leq j < j' \leq p} \{D(r, X_j) + D(r, X_{j'})\} \right\} \right\}$$

where $D(r, X)$ denotes the Euclidean distance between the two points r and X. In the drone delivery application, a solution to this problem ensures that any possible demand point located in an area can be served by a route that starts from a depot and ends at a different depot, and that the battery capacity needed to support such a service mode is minimized.

We now propose a heuristic algorithm to find elliptical covers for a convex polygon. The idea is to repeatedly solve the Elliptical Cover problem for a growing set of demand points that approximate the polygonal area.

Area Coverage Algorithm:

Step 1: Take all corner points of the convex polygon and randomly sample a small number of points within the polygon. These points will be used as the initial set of demand points, denoted by $\mathcal{C}$.

Step 2: Run the Locate-Allocate algorithm on $\mathcal{C}$ to obtain the optimal value L^* and the optimal solution $X_1^*, \dots, X_p^*$.

Step 3: Solve the following convex optimization problem on variables $r \in \mathbb{R}^2$ and $d \in \mathbb{R}$, which is to compute the maximum route distance to serve a demand in region R based on the current depot locations.

$$\begin{aligned} &\text{maximize}_{r,d} \quad d \\ &\text{s.t.} \quad d \leq D(r, X_j^*) + D(r, X_{j'}^*) \quad \text{for } 1 \leq j < j' \leq p \\ &\qquad r \in R \end{aligned}$$

Let r^* be the optimal solution and d^* be the optimal objective value of this problem.

Table 1
Time comparison between MIP and the recursive methods for solving the SCI problem.

n	p	MIP	Alg1	NC	n	p	MIP	Alg1	NC	n	p	MIP	Alg1	NC
10	3	0.2	0.0	1	20	3	0.2	0.0	1	50	3	0.5	0.0	1
10	4	0.2	0.0	1	20	4	0.2	0.0	1	50	5	2.5	0.0	1
10	5	0.5	0.0	1	20	5	0.8	0.0	1	50	7	37.8	0.0	1
10	6	0.9	0.0	1	20	6	4.7	0.0	1	50	9	205.5	0.0	1
10	7	2.2	0.0	1	20	7	6.4	0.0	1	100	3	0.3	0.0	1
10	8	2.6	0.0	19	20	8	10.8	0.0	1	100	5	15.6	0.0	1
10	9	6.6	0.0	32	20	9	28.4	0.0	1	100	7	461.2	0.0	1
10	10	22.5	0.0	50	20	10	129.8	0.0	1	100	9	-	0.0	1

Step 4: If $d^* - L^* < \epsilon$, stop and return the solution $\{X_1^*, \ldots, X_p^*\}$. Otherwise, $\mathcal{C} \leftarrow \mathcal{C} \cup \{r^*\}$, go to Step 2.

Note that in each iteration, the depot locations updated in the preceding iteration should be used as the starting point for the Locate-Allocate algorithm. This ensures that the new point r^* is not covered by the incumbent solution, forcing the discovery of a new solution that covers all points (including the new point) in $\mathcal{C}$. Like the Voronoi heuristic for the area-covering EPC problem (Suzuki & Drezner, 1996), the above algorithm is a heuristic method for finding good feasible solutions for the area-covering elliptical cover problem. In numerical experiments, the algorithm never failed to converge to a reasonable solution within relatively few iterations.

6. Numerical experiments

In this section, we demonstrate the performance of the proposed algorithms using simulated data cases. For the SCI problem, we generate random instances of different sizes to compare the MIP approach and the recursive algorithms in Algorithm 1 and 2. Specifically, each value in the input vector $\{a_1, \ldots, a_n\}$ is uniformly sampled from the interval [0,100]. For the Elliptical Cover problem, we focus on a fictitious scenario of locating drone depots to serve geographically scattered customers in a city. We use the city of Troy, Michigan, in the backdrop, which spans an approximately 10 km by 10 km square area. All experiments were run on a Dell Precision Tower 8520 with an Intel(R) Core(TM) i9-9900X CPU @ 3.50 GHz, 64 GB RAM on Windows 10 Enterprise Operating System. The computer programs and data files used in the experiments are available at https://github.com/profyliu/elliptical_cover.

6.1. Solving the SCI problem

Table 1 lists the time taken (in seconds) for the MIP model (via CPLEX 12.10) and the Algorithm 1, respectively, to solve random instances of size n and p. The column NC lists the number of calls to the SCI function in Algorithm 1's recursive process. Both methods return the same optimal values for all cases, as expected. For the case of $(n, p) = (100, 9)$, the MIP was not solved within 1800 seconds. We can see that Algorithm 1 scales much better than the MIP method. In particular, when p is relatively small compared to n, it is likely that the equispaced cut points are indeed the optimal solution, hence, only one call to the SCI function is required, regardless of how big n is. When p approaches n, however, the curse of nested recursion starts to take effect, as exhibited in the cases with $n = 10$ and $p = 8, 9$ and 10. Indeed, the greater p is compared to n, the more cluster break points there will be, hence, the larger the search space will become. In such cases, Algorithm 2 can be used to find a near-optimal solution more quickly. The performances of the exact and heuristic algorithms in cases with large p values are compared in Table 2. We can see that the number of function calls in the exact algorithm increased exponentially as p increases. In contrast, the heuristic algorithm invoked much fewer function calls, and was able to find the optimal solution for nine out of the ten cases. The MIP method was unable to complete in 1800 seconds for all cases with $p \geq 6$, so its performance was not included in the table. These results suggest that practical instances of the SCI problem, i.e., when n is reasonably larger than p, is easy to solve. For artificial cases where p is larger than n, Algorithm 2 is faster, and can find the optimal solution most of the time.

Table 2
Efficiency comparison between the exact and heuristic algorithms for solving the SCI problem.

n	p	Alg1 (Exact)			Alg2 (Heuristic)		
		Objval	Time	NC	Objval	Time	NC
10	2	91.030	0.0	1	91.030	0.0	1
10	4	30.343	0.0	1	30.343	0.0	1
10	6	18.206	0.0	1	18.206	0.0	1
10	8	12.003	0.0	19	12.003	0.0	3
10	10	8.893	0.0	50	9.003	0.0	5
10	12	5.230	0.0	519	5.230	0.0	5
10	14	3.570	0.1	2537	3.570	0.0	11
10	16	1.785	0.5	7978	1.785	0.0	9
10	18	0.650	1.2	20320	0.650	0.0	46
10	20	0.000	2.7	45310	0.000	0.0	32

Table 3
Performance of the locate-allocate algorithm with multistart.

n	p	When	NUniq	Min	Med	Max	Time	Objval	LB
20	3	13	24	2	3	5	0.70	7977.61	4610.22
20	4	4	55	2	4	9	1.37	6435.75	3073.48
20	5	39	90	2	4	10	1.50	5366.45	2305.11
20	6	13	125	2	5	12	2.81	4645.92	1844.09
50	3	4	29	2	3	7	0.96	8502.95	5467.31
50	4	16	55	2	4	11	1.23	6988.37	3644.87
50	5	12	120	2	5	11	1.52	5846.27	2733.65
50	6	31	185	2	6	12	1.92	5185.91	2186.92
100	3	5	31	2	4	8	1.19	9244.64	6263.97
100	4	30	48	2	5	11	1.41	7590.22	4175.98
100	5	29	115	2	6	17	1.84	6381.85	3131.99
100	6	9	232	2	6	15	1.80	5622.63	2505.59

6.2. Solving the elliptical cover problem

We generated three scenarios by randomly scattering $n = 20, 50$ and 100 demand points in the city's perimeter. The latitude and longitude of each demand point were generated independently using a uniform distribution in the applicable range, points that fell outside the city's perimeter were discarded. For each demand scenario, we solved the Elliptical Cover problem for different p values ranging between 3 and 6. Table 3 lists the results. While solving an instance, the locate-allocate algorithm was repeated 500 times, each time starting from random depot locations generated via a uniform distribution, and the solution having the minimum objective value was returned. In Table 3, the column When represents in which (out of the 500) iteration the best found solution first appeared and NUniq is the number of unique local optima encountered in all 500 iteration. The overall small values (as compared to

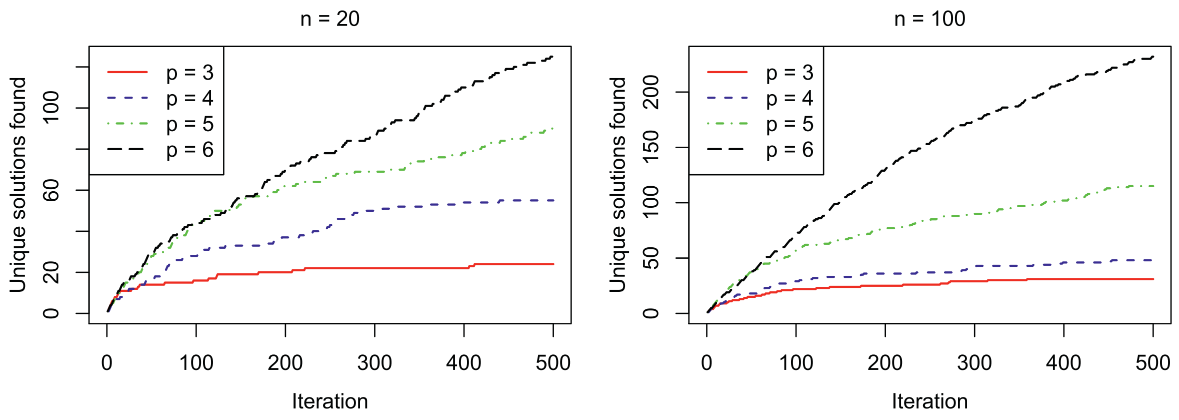

Fig. 8. The number of local optima found increases slowly over repeated runs of the random-start locate-allocate algorithm for a small p value, and increases more quickly for larger p values.

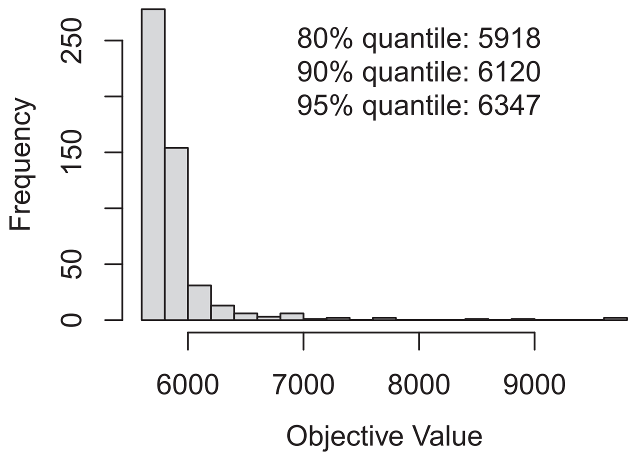

Fig. 9. The distribution of objective values found in the case $n = 100$, $p = 6$. The skewness suggests that most local optima are near-optimal.

500) in these columns suggest that the chance of finding a better solution in the next iteration rapidly decreases as more iterations are performed; thus, an economical search need not consist of too many iterations. To demonstrate this point more clearly, Fig. 8 plots how the number of unique solutions found increases over the iterations, which, in addition, reveals that the value of p plays a more important role than n in determining the intractability of the problem - each increment in p would substantially enlarge the solution space, hence, making it more difficult to obtain a good solution within limited number of iterations. This observation agrees with those in the SCI experiments.

Despite the large number of local optima, the distribution of the local optima is extremely skewed, with the majority concentrated in the lower (better) end. For instance, in the case of $(n, p) = (100, 6)$, 400 out of the 500 iterations (i.e., 80%) returned a local optima only 5% worse than the best-found solution. The distribution of local optima in this case is plotted in Fig. 9. This trend enhances the evidence that the multistart locate-allocate algorithm is quite effective at discovering satisfactory local solutions.

The columns Min, Med and Max in Table 3 are the minimum, median and maximum number of iterations taken for the location-allocation process to converge to a local optimum, and the column Time is the average time (in seconds) taken for the process to converge. The small numbers suggest that convergence of the location-allocation process has been consistently fast. Note that $\epsilon = 50$ m (less than 1% of the objective value) has been used for the termination condition. The columns Objval and LB are the best objective value and the lower bound value, in meters. The lower bounds are obtained via solving the SCI problem using Algorithm 1, which are the best lower bounds achievable within 0.1 second. Finding tight lower bounds is an $\mathcal{NP}$-hard problem, though novel relaxation approaches that are Pareto better than the SCI approach may be explored in future research.

Fig. 10 visualizes the best-found solutions for $n = 100$ and $p = 3$ and 6. All demand points are covered by some ellipse whose foci are a pair of depots in the solution. The number of covering ellipses formed by the p depots is uncertain until a solution is presented, as this number depends on the actual location of the depots. The lengths of the major axes of all ellipses involved in the solution are the same, which are equal to the objective value. This property is intuitive, and is to be expected in all “single-cluster” solutions found by the locate-allocate algorithm.

6.3. Area coverage using ellipses and comparison with the p-center solution

We applied the area coverage algorithm presented in Section 5 to the area of Troy, for various p values, to demonstrate the algorithm’s effectiveness. The goal is to find ellipses based on p foci to cover all points in the area. The initial small set of points consisted of the corners of the region’s convex hull and $2p$ other points randomly sampled in the interior of the region. Fig. 11 plots the convergence paths for 20 runs for the case $p = 6$. The Gap shown along the vertical axis is the value of $d^* - L^*$ calculated in Step 4 of the algorithm. The non-monotone reduction in the gap is expected, since L^* is not intended to be the global minimum in any iteration. Overall, the solution process terminated successfully in all cases attempted, and for the 20 runs exhibited in the figure, they all terminated within 55 iterations, whereas the termination tolerance ϵ was set to 50 m. The best solution found in the 20 runs, along with the dummy demand points generated in the solution process, is demonstrated on the left part of Fig. 12.

To demonstrate the practical significance of the Elliptical Cover solution in drone delivery network design, we also obtained the EPC solution (with $p = 6$) using the algorithm developed in Liu (2021). The goal of the EPC problem is to place the depots such that the distance from any point in the area to its nearest depot is minimized. We ran the algorithm 20 times and presented the best found solution on the right part of Fig. 12. The dashed ellipses are formed by taking the depots as foci and having major axis length equal to the (common) diameter of the EPC covering circles. We can see that the required flight ranges (or battery capacities) from the two solutions do not differ much, i.e., 5766 m by the Elliptical

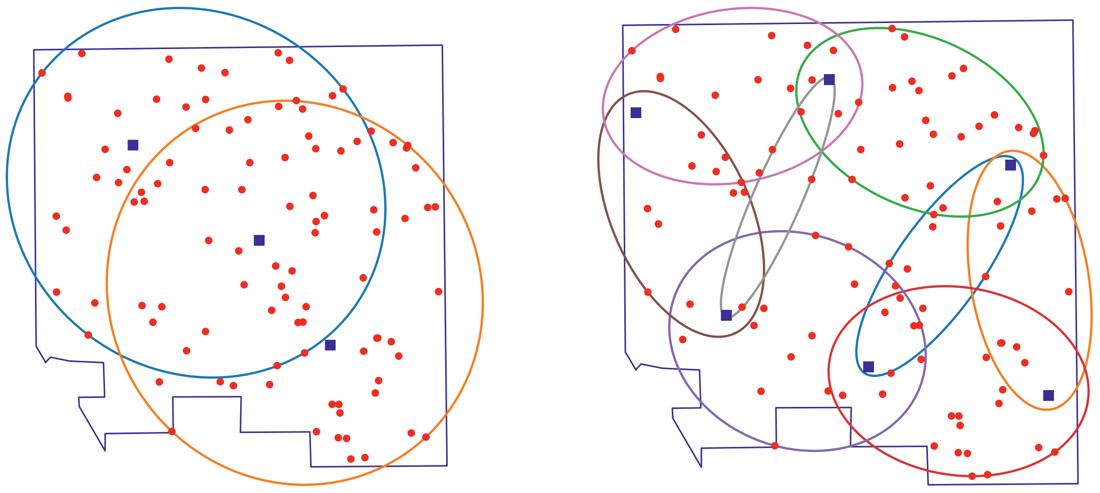

Fig. 10. Demonstration of using 3 and 6 depots to cover 100 demand points in the Troy area.

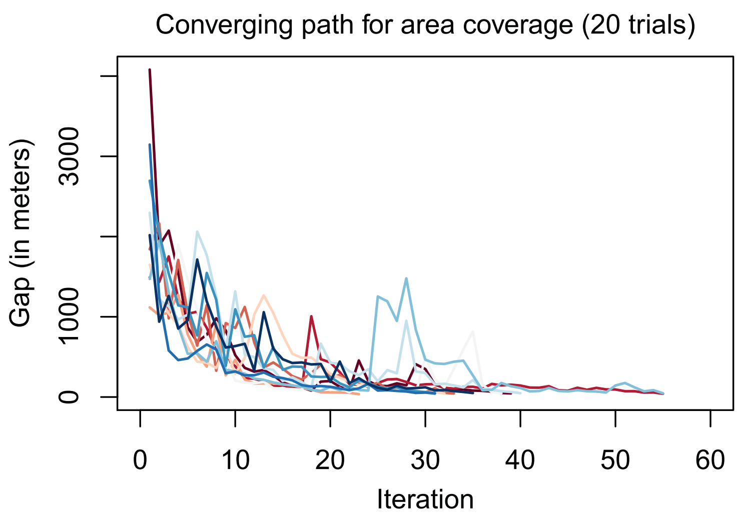

Fig. 11. Convergence pattern of the area coverage algorithm applied to Troy with $p = 6$ depots.

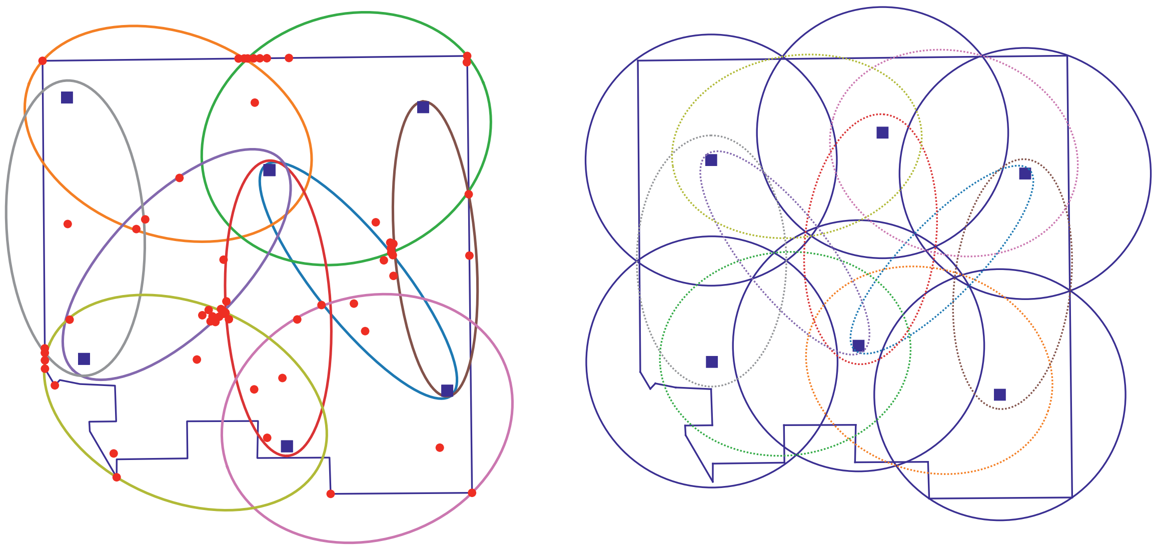

Fig. 12. Comparison between the p-Elliptical Cover and Euclidean p-center solutions. Left: The best Elliptical Cover solution found to cover the area of Troy with 6 depots, with objective value (required flight range) 5776 m. Right: The best p-center solution to cover the area with 6 depots, with circle diameter (required flight range) 5662 m. Note that the ellipses formed by the p-center solution cannot cover the whole area, leaving an appreciable portion of the area serviceable by only a type 1 trip of a single depot.

Cover solution versus 5662 m by the EPC solution. However, the depot locations suggested by the EPC solution are unable to provide full, type-2 trip coverage for the whole service area - an appreciable portion of the demand area is only serviceable by a type 1 trip. For instance, if the demanded item is out of stock in the southeast depot, then customer demands near the southeast corner of Troy cannot be fulfilled if the system (including the depot location and the fleet's battery capacity) is configured according to the EPC solution. In comparison, if the system is configured by the Elliptical Cover solution, the service will be robust against this kind of stock-out situations. Furthermore, the Elliptical Cover solution always allows a drone to relocate to a different depot (in preparation for the next delivery task starting from that depot) after performing a delivery task but the EPC solution does not guarantee such flexibility.

7. Conclusion

In this paper, we have studied a novel geometric facility location problem, namely, the Euclidean p-Elliptical Cover problem, motivated by the network design of drone delivery systems. We have proven the $\mathcal{NP}$-hardness of this problem and analyzed the unique challenges it poses to known algorithms for similar problems, due to its graph isomorphism and non-decomposibility. We have proposed a locate-allocate algorithm that is able to converge to a local optimum typically in a few iterations. Repeatedly running this fast algorithm from random starting points has been the only viable approach for pumping up satisfactory solutions, for both the point-coverage problem and the area-coverage problem. We have furthermore investigated the one-dimensional variant of the problem, the Shortest Covering Interval problem, which not only provides a Pareto best (in terms of performance and computing time) lower bound to the Elliptical Cover problem, but also finds its application when the service area is reduced to a line segment, such as one along a power transmission line or a railway line. We have developed an exact and a heuristic algorithm based on proven properties of the solution. The proposed algorithms have been validated to be effective and efficient in practical data cases.

Compared to covering the demand locations with circles, covering them with ellipses enables higher levels of service availability, network connectivity and vehicle utilization. Future research could develop methods to optimally cover a density map of the service area, i.e., requiring regions with higher demand rates to be covered by more service routes, or to cover a service area having forbidden areas, such as lakes and no-fly zones. The depot network topology and connectivity may impose explicit constraints in certain application scenarios. For instance, in case of highly clustered demand distribution or of irregularly shaped demand regions, the elliptical cover solution may contain "weak" coverage links, e.g., two demand centers connected by a single pair of depots, which may cause air traffic congestion for drones shuttling between the depots. Additional constraints would be needed to address such issues. The elliptical cover idea could also be extended to the problem of locating and routing "moving depots" intended to cover probabilisitic and time-varying demand point locations. Finally, the mathematical formulation of the p-Elliptical Cover problem permits the use of other distance metrics other than the Euclidean distance. An investigation into the solution approaches as well as into data mining applications could be interesting future work.

Acknowledgment

The research is supported by the National Science Foundation via grant number 1944068.

References

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization approaches for the traveling salesman problem with drone. *Transportation Science, 52*(4), 965–981. https://doi.org/10.1287/trsc.2017.0791.

Aikens, C. H. (1985). Facility location models for distribution planning. *European Journal of Operational Research, 22*(3), 263–279. https://doi.org/10.1016/0377-2217(85)90246-2. URL http://www.sciencedirect.com/science/article/pii/0377221785902462

Andretta, M., & Birgin, E. G. (2013). Deterministic and stochastic global optimization techniques for planar covering with ellipses problems. *European Journal of Operational Research, 224*(1), 23–40. URL https://EconPapers.repec.org/RePEc:eee:ejores:v:224:y:2013:i:1:p:23-40

Berman, O., Drezner, Z., Tamir, A., & Wesolowsky, G. O. (2009). Optimal location with equitable loads. *Annals of Operations Research, 167*(1), 307–325.

Blanco, V., & Puerto, J. (2021). Covering problems with polyellipsoids: A location analysis perspective. *European Journal of Operational Research, 289*(1), 44–58. https://doi.org/10.1016/j.ejor.2020.06.048. URL https://www.sciencedirect.com/science/article/pii/S0377221720306093

Blanco, V., Puerto, J., & Ben-Ali, S. E.-H. (2014). Revisiting several problems and algorithms in continuous location with l_τ norms. *Computational Optimization and Applications, 58*(3), 563–595.

Blanco, V., Puerto, J., & Ben-Ali, S. E.-H. (2016). Continuous multifacility ordered median location problems. *European Journal of Operational Research, 250*(1), 56–64.

Brimberg, J., Hansen, P., Mladenović, N., & Taillard, E. D. (2000). Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem. *Operations Research, 48*(3), 444–460.

Brimberg, J., & Salhi, S. (2005). A continuous location-allocation problem with zone-dependent fixed cost. *Annals of Operations Research, 136*(1), 99–115.

Callaghan, B., Salhi, S., & Nagy, G. (2017). Speeding up the optimal method of Drezner for the p-centre problem in the plane. *European Journal of Operational Research, 257*(3), 722–734.

Canbolat, M. S., & Massow, M. v. (2009). Planar maximal covering with ellipses. *Computers & Industrial Engineering, 57*(1), 201–208. https://doi.org/10.1016/j.cie.2008.11.015.

Chazelle, B., & Lee, D. (1986). On a circle placement problem. *Computing, 36*, 1–16. https://doi.org/10.1007/BF02238188.

Chen, D., & Chen, R. (2013). Optimal algorithms for the alpha-neighbor p-center problem. *European Journal of Operational Research, 225*, 36–43. https://doi.org/10.1016/j.ejor.2012.09.041.

Church, R. L. (1984). Symposium on location problems: In memory of leon cooper. *Journal of Regional Science, 24*(2), 185–201. https://doi.org/10.1111/j.1467-9787.1984.tb01031.x.

Cooper, L. (1964). Heuristic methods for location-allocation problems. *SIAM Review, 6*(1), 37–53. https://doi.org/10.1137/1006005.

Daskin, M. S. (1995). *Network and discrete location: Models, algorithms, and applications*. John Wiley & Sons, Inc.

Drezner, Z. (1984). The p-centre problem-heuristic and optimal algorithms. *The Journal of the Operational Research Society, 35*(8), 741–748.

Drezner, Z., & Brimberg, J. (2020). Improved starting solutions for the planar p-median problem. *Yugoslav Journal of Operations Research, 31*(1), 45–64.

Drezner, Z., Brimberg, J., Mladenović, N., & Salhi, S. (2015). New heuristic algorithms for solving the planar p-median problem. *Computers & Operations Research, 62*, 296–304. https://doi.org/10.1016/j.cor.2014.05.010.

Farahani, R. Z., Asgari, N., Heidari, N., Hosseininia, M., & Goh, M. (2012). Covering problems in facility location: A review. *Computers & Industrial Engineering, 62*, 368–407.

Garey, M. R., & Johnson, D. S. (1978). *Computers and intractability: A guide to the theory of NP-Completeness* (1st). W. H. Freeman.

Hansen, P., Mladenović, N., & Taillard, E. (1998). Heuristic solution of the multisource Weber problem as a p-median problem. *Operations Research Letters, 22*(2), 55–62.

Hwang, R. Z., Lee, R. C. T., & Chang, R. C. (1993). The slab dividing approach to solve the Euclideanp-center problem. *Algorithmica, 9*, 1–22.

Khuller, S., Pless, R., & Sussmann, Y. J. (1997). Fault tolerant k-center problems. In G. Bongiovanni, D. P. Bovet, & G. Di Battista (Eds.), *Algorithms and complexity - 3rd Italian conference, CIAC 1997, proceedings*. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (pp. 37–48). Springer Verlag.

Klose, A., & Drexl, A. (2005). Facility location models for distribution system design. *European Journal of Operational Research, 162*(1), 4–29. Logistics: From Theory to Application

Liu, Y. (2019). An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. *Computers & Operations Research, 111*, 1–20.

Liu, Y. (2021). A faster algorithm for the constrained minimum covering circle problem to expedite solving p-center problems in an irregularly shaped area with holes. *Naval Research Logistics*. https://doi.org/10.1002/nav.22023.

Liu, Y. (2022). Two lower-bounding algorithms for the p-center problem in an area. *Computational Urban Science, 2*.

Megiddo, N., & Supowit, K. (1984). On the complexity of some common geometric location problems. *SIAM Journal on Computing, 13*(1), 182–196.

Minieka, E. (1970). The m-center problem. *SIAM Review, 12*(1), 138–139. https://doi.org/10.1137/1012016.

Murray, C. C., & Chu, A. G. (2015). The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies, 54*, 86–109. https://doi.org/10.1016/j.trc.2015.03.005.

Nesterov, Y., & Nemirovskii, A. (1994). *Interior-point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9781611970791.

Poikonen, S., & Campbell, J. (2020). Future directions in drone routing research. *Networks, 77*. https://doi.org/10.1002/net.21982.

Raeisi Dehkordi, A. (2019). The optimal solution set of the multi-source weber problem. *Bulletin of the Iranian Mathematical Society, 45*(2), 495–514.

Revelle, C. S., & Laporte, G. (1996). The plant location problem: New models and research prospects. *Operations Research, 44*(6), 864–874.

Rosing, K. E. (1992). An optimal method for solving the (generalized) multi-Weber problem. *European Journal of Operational Research, 58*(3), 414–426. https://doi.org/10.1016/0377-2217(92)90072-H. URL http://www.sciencedirect.com/science/article/pii/037722179290072H

Sahinidis, N. V. (1996). BARON: A general purpose global optimization software package. *Journal of Global Optimization, 8*(2), 201–205.

Schilling, D. A., Jayaraman, V., & Barkhi, R. (1993). A review of covering problems in facility location. *Location Science, 1*, 25–55.

Suzuki, A., & Drezner, Z. (1996). The p-center location problem in an area. *Location Science, 4*(1), 69–82.

Tedeschi, D., & Andretta, M. (2021). New exact algorithms for planar maximum covering location by ellipses problems. *European Journal of Operational Research, 291*(1), 114–127. https://doi.org/10.1016/j.ejor.2020.09.02. URL https://ideas.repec.org/a/eee/ejores/v291y2021i1p114-127.html

Vijay, J. (1985). An algorithm for the p-center problem in the plane. *Transportation Science, 19*(3), 235–245.

Wei, H., Murray, A. T., & Xiao, N. (2006). Solving the continuous space p-centre problem: planning application issues. *IMA Journal of Management Mathematics, 17*(4), 413–425.

Wesolowsky, G. O. (1993). The Weber problem: History and perspectives. *Location Science, 1*, 5–23.

Wing (2021). Learn about how wing delivery works. https://wing.com/about-delivery/.

Zipline (2021). Put autonomy to work. https://flyzipline.com/how-it-works/.