Planar Maximum Coverage Location Problem with Partial Coverage, Continuous Spatial Demand, and Adjustable Quality of Service

Manish Bansal and Parshin Shojaee
Department of Industrial and Systems Engineering, Virginia Tech
Email: bansal@vt.edu, parshinshojaee@vt.edu

First Draft: December 10, 2020

Abstract. We consider a generalization of the classical planar maximum coverage location problem (PMCLP) in which partial coverage is allowed, facilities have adjustable quality of service (QoS) or service range, and demand zones and service zone of each facility are represented by two-dimensional spatial objects such as rectangles, circles, polygons, etc. We denote this generalization by PMCLP-PC-QoS. A key challenge in this problem is to simultaneously decide position of the facilities on a continuous two-dimensional plane and their QoS. We present a greedy algorithm and a pseudo-greedy algorithm for it, and showcase that the solution value corresponding to the greedy (or pseudo-greedy) solution is within a factor of $1 - 1/e$ (or $1 - 1/e^{\eta}$) of the optimal solution value where e is the base of natural logarithm and $\eta \leq 1$. We also investigate theoretical properties and propose exact algorithms for solving: (1) PMCLP-PC-QoS where demand and service zones are represented by axis-parallel rectangles (denoted by PMCLP-PCR-QoS), which also has applications in camera surveillance and satellite imaging; and (2) one dimensional PMCLP-PC-QoS which has applications in river cleanups. These results extend and strengthen the only known exact algorithm for PMCLP-PCR-QoS with fixed and same QoS by Bansal and Kianfar [INFORMS Journal on Computing 29(1), 152-169, 2017]. We present results of our computational experiments conducted to evaluate the performance of our proposed exact and approximation algorithms.

Keywords: planar maximum coverage location problem; partial coverage; adjustable quality of service; greedy approach; spatial demand representation; branch-and-bound exact algorithm

1 Introduction

Over the years, many classes of problems related to locating service facilities to cover the demand for service have been studied; refer to Church and Murray (2013), Drezner and

Hamacher (2002) for extensive reviews. A well-known classical facility location problem is Maximum Coverage Location Problem (MCLP) for locating p service facilities having known service range with the objective of maximizing the total covered demand (Church and ReVelle 1974). In literature, various forms of MCLP consider a finite set of predetermined candidate positions for the facilities (Church and ReVelle 1974, Murray and O'Kelly 2002, Daskin et al. 1989). However, a generalization of the classical MCLP, referred to as the planar MCLP (PMCLP), considers locating the facilities anywhere in a continuous two-dimensional plane (Church 1984, Watson-Gandy 1982, Mehrez and Stulman 1982). These problems have received considerable attention in the literature due to their widespread applicability ranging from locating emergency healthcare centers, locating fire fighting stations, and making policy through geographical informative systems (GIS) to solve clustering problems which themselves find applications in data mining, machine learning, and bio-informatics (Chung 1986, Farahani et al. 2012, Schilling et al. 1993). With the growing availability of spatial data in the foregoing domains, the state-of-the-art facility location analytical tool set needs to evolve. In this paper, we consider a generalization of the PMCLP, by utilizing spatial representation of demand and service zones of facilities, allowing adjustable quality of service (QoS) or service range for the facilities, and allowing partial coverage in its true sense.

In most of the PMCLP literature, demands are represented as aggregated points, which are obtained by aggregating the demand of each zone at a single representative point (e.g., its centroid). Murray and Tong (2007) considered PMCLP where the demand zones are represented as line segments or polygons instead of limiting them to be represented by aggregated points. However, similar to the PMCLP with point representation of demand, they assume that demand zones (points, line segments, or polygons) are either completely covered or not covered at all by any service zone. This coverage assumption is referred to as "binary coverage" (Bansal and Kianfar 2017). The binary coverage is a simplifying assumption to model PMCLP as binary programs, but it ignores partial coverage of demand zones. Researchers have studied its impact and referred to it as region misrepresentation coverage error (RMCE) (Current and Schilling 1990, Tong and Church 2012, Murray 2016). These analysis represent that the modeling solutions are sensitive to how demand is represented in the PMCLP (Murray and O'Kelly 2002), and there are evidences that the point representation introduces unintended measurement and interpretation errors (Current and Schilling 1990, Tong and Murray 2009) which lead to gaps in actual coverage.

In Figure 1(a), we present an example of PMCLP where demand zones are represented using points and the service zone of a facility with fixed service range r, using Euclidean

distance, is a circle of radius r centered at the facility. Observe that because of the binary coverage assumption, only two demand zones are completely covered by the service zone, whereas others are not at all covered. This causes RMCE because in reality, as shown in 1(b), five demand zones (represented by spatial objects) are partially covered by the service zone. Note that the service zone of a facility, using the rectilinear distance, is a diamond (a square rotated 45 degrees) with a diagonal of $2d$ centered at the facility. Lately, Bansal and Kianfar (2017) developed the first exact algorithm for the PMCLP where "partial coverage" is allowed, and service and demand zones are defined by axis-parallel rectangles. It generalizes PMCLP with rectilinear distance. However, they assumed that all facilities have fixed and same service range or QoS, i.e., dimensions of the rectangular service zones are fixed and same.

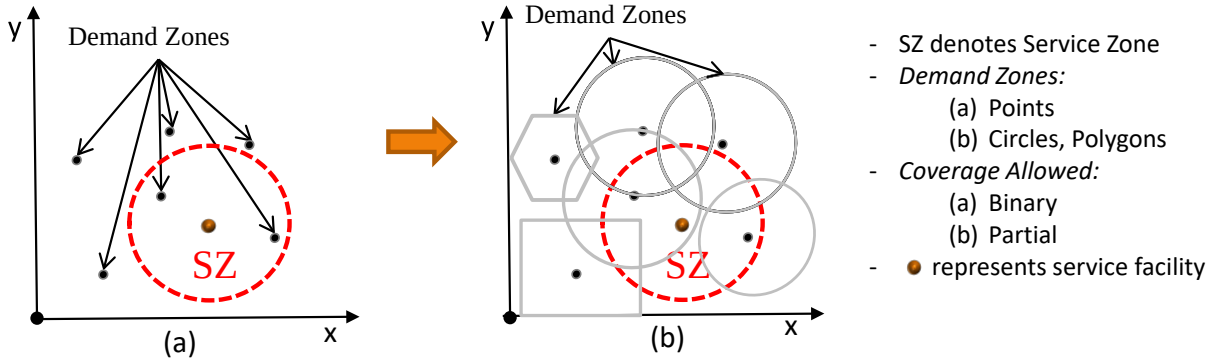


Figure 1: (a) Classical Planar MCLP; (b) Planar MCLP with partial coverage and general representation of demand and service zones.

Although attempts have been made to separately address non-point representation of demand, partial coverage, and adjustable QoS, but to our knowledge, no study has tackled them together. Few studies have considered MCLP with gradual coverage, i.e., the coverage level of a demand zone depends on their distance from the facilities (Berman and Krass 2002, Church and Roberts 1983, Berman et al. 2003), but similar to MCLP, the demand zones are still represented by points and there is a finite set of pre-specified candidate positions for the facilities. This problem is referred to as gradual coverage location problem. Drezner et al. (2004) considered planar version of the forgoing problem with only single facility. (See section 2 for more details.) However so far, no study has considered PMCLP with (i) general representation of demand and service zones using two-dimensional spatial objects, (ii) facilities having adjustable QoS or service range, and (iii) partial coverage in its true sense, i.e., when covering only part of a demand zone is allowed and the coverage accrued in the objective function as a result of this is proportional to the demand of the covered area only. We denote this generalization of PMCLP by PMCLP-PC-QoS.

1.1 Problem definition: PMCLP-PC-QoS

We define the PMCLP-PC-QoS as follows. Let $\mathcal{D} = \{d_i, i = 1, \ldots, n\}$ be a set of n (possibly overlapping) spatial objects, referred to as demand zones (DZs), on a two-dimensional plane such that dimensions and location of each DZ are known. Now, consider another set of spatial objects that provide coverage of facilities, referred to as service zone (SZ), and denote the set of SZs by $\mathcal{S} := \{s_j, j = 1, \ldots, p\}$. We assume that shape and orientation of all SZs are same and known, but their location and dimensions are unknown. Let s_0 be a spatial object with same shape and orientation as of the SZs, but with known dimensions. We refer to s_0 as "base" SZ because it provides base reward rate (per unit area) $v_i \in \mathsf{R}_+$ for covering each DZ d_i. Dimensions of each SZ $s \in \mathcal{S}$ is a scalar $z_s \geq 1$ multiple of the dimensions of s_0. We refer to z_s as the scaling factor of SZ s. The base reward rate is utilized to compute the reward rate $v_i^z = v_i/\eta(z) \leq v_i$ for each DZ d_i covered by a SZ with scaling factor z, where $\eta(z)$ is a strictly increasing function. When scaling factor z increases, the dimensions of a SZ increases but the corresponding reward rate decreases. Therefore, by increasing the scaling factor z, the service range of a facility increases, but the QoS decreases. We assume that $\Xi_s := \{\xi_1^s, \ldots, \xi_m^s\}$ is a finite set of m possible scaling factors for SZ $s \in \mathcal{S}$, which allows adjustable QoS for the SZs. In this paper, we will use terms QoS, scaling factor, or service range interchangeably for a SZ s and associate all of them with $z_s \in \Xi_s$ for $s \in \mathcal{S}$.

The goal of PMCLP-PC-QoS is to identify the location and QoS of all SZs, i.e., $(x_{s_j}, y_{s_j}, z_{s_j}) \in \mathsf{R}^2 \times \Xi_s$ for all $j = 1, \ldots, p$, where (x_{s_j}, y_{s_j}) denotes coordinates of either center or a corner (if exists) of SZ s_j. The objective function of this problem is to maximize the total reward captured by these p SZs, i.e.,

$$\max_{\mathbf{x},\mathbf{y},\mathbf{z}} \left\{ f(\mathbf{x},\mathbf{y},\mathbf{z}) := \sum_{i=1}^{n} f_i(\mathbf{x},\mathbf{y},\mathbf{z}) = \mathcal{T}_i\Big(d_i \cap \Big(\cup_{j=1}^{p} s_j \Big)\Big) \right\} \tag{1}$$

where $\cup$ and $\cap$ denote the union and intersection of coverage zones, $\mathcal{T}_i(.)$ returns the total reward captured from DZ d_i by SZs $s_1, \ldots, s_p$, and $(\mathbf{x},\mathbf{y},\mathbf{z}) = (x_{s_1}, \ldots, x_{s_p}, y_{s_1}, \ldots, y_{s_p}, z_{s_1}, \ldots, z_{s_p})$. In case each SZ offers same QoS, i.e., $z_{s_j} = \hat{z}$ for all $j \in \{1, \ldots, p\}$, the function $f(\mathbf{x},\mathbf{y},\mathbf{z})$ reduces to $\sum_{i=1}^{n} v_i^{\hat{z}} \times A\Big(d_i \cap \Big(\cup_{j=1}^{p} s_j \Big)\Big)$ where $A(.)$ returns the area of its argument. In this paper, we present greedy and pseudo-greedy approximation algorithms for the PMCLP-PC-QoS as well as exact algorithms for its following variants:

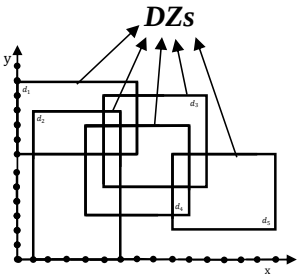
(a) *PMCLP-PC-QoS with axis-parallel rectangular service and demand zones* (denoted by PMCLP-PCR-QoS). Given a set of rectangular DZs identified by coordinates of their lower left corner and their dimensions (width and length), denoted by $(x_{d_i}, y_{d_i}) \in \mathsf{R}^2$

and $(w_{d_i}, l_{d_i}) \in \mathsf{R}^2_+$, respectively, for $i = 1, \ldots, n$. We assume that the base SZ s_0 is an axis-parallel rectangle with width w_{s_0} and length l_{s_0}. Consequently, the SZs $s_1, \ldots, s_p$ are also axis-parallel rectangles. We also assume that the set of scaling factors, $\Xi_s = \Xi := \{\xi_1, \xi_2, \ldots, \xi_m\}$ for all $s \in \mathcal{S}$. The goal of PMCLP-PCR-QoS is to find locations $(\mathbf{x}, \mathbf{y})$ as well as scaling factor $\mathbf{z}$ of all SZ such that the maximum reward is covered. Observe that the width and length of SZ s depend on the scaling factor $z_s \in \Xi$ associated to it, i.e., $(w^z_s, l^z_s) = (z_s w_{s_0}, z_s l_{s_0})$. A motivation behind studying this problem is its application in telerobotics, camera surveillance, and satellite imaging (refer to the next section for details). Also, the problems studied by Song et al. (2006) and Bansal and Kianfar (2017) are special cases of PMCLP-PCR-QoS where $p = 1$ and $\Xi = \{1\}$, i.e., the facilities cannot adjust QoS and their SZs have fixed and same dimensions, respectively. Bansal and Kianfar (2017) also proved that if p is a part of input, PMCLP-PCR-QoS with $\Xi = \{1\}$, denoted by PMCLP-PCR, is NP-hard. Therefore, PMCLP-PC-QoS and PMCLP-PCR-QoS are also NP-hard because PMCLP-PCR is a special case of these problems.

(b) *One-dimensional PMCLP-PC-QoS* (1D-PMCLP-PC-QoS): As the name suggests, the one-dimensional facility location problem has DZs and SZs placed on a line. Specifically, we assume that the DZs are line segments on x-axis whose coordinate of left corner or end point, x_{d_i}, $i = 1, \ldots, n$, and width, w_{d_i}, $i = 1, \ldots, n$, are known. Likewise, the SZs are also defined as line segments on x-axis with different QoS, i.e., $\Xi_s = \{\xi^s_1\}$. The motivation behind studying this special case is its application in locating trash booms (facilities) for cleaning trash zones (or DZs) in a river (represented by 1D line). In the next section, we discuss more about this application as well. Another application of this problem is in locating p regional wastewater treatment plants on a river that has a given set of fixed segments (DZs) with high priority (reward rate). To our knowledge, the 1D-PMCLP-PC-QoS has not been studied before. In literature, one-dimensional plant location problem is studied by Brimberg and ReVelle (1998), and Brimberg et al. (2001) where DZs are represented by points, and facilities are uncapacitated and capacitated, respectively. As a result, the authors were able to provide mixed binary programs and dynamic programs, respectively, for them. They also show that under certain conditions, these problems are polynomially solvable using linear programs.

In Figure 2, we provide an example to illustrate PMCLP-PCR-QoS with five DZs $\{d_1, d_2, \ldots, d_5\}$ with known location, width, length, and base reward rates, a base SZ s_0 with known dimensions, and a set of scaling factors $\Xi = \{\xi_1, \ldots, \xi_m\}$ for two SZs, i.e. $p = 2$. A similar example for $m = 1$ has been considered by Bansal and Kianfar (2017). The locations and dimensions are considered integers just for the simplicity of representation. In this example, we assume that $\eta(z) = z$ and therefore, $v^z_i = v_i/z$. Using the algorithms

presented in this paper, we present optimal and greedy solutions (along with solution values, i.e., reward) for $m \in \{1, \ldots, 4\}$ and $\xi_k = k$, $k = 1, \ldots, 4$. Notice that for $m \in \{1, 2\}$, the greedy solution is same as the optimal solution for this example. However, for $m \in \{3, 4\}$, the optimal solution value is better than the greedy solution value. Moreover, with the increase in m, the captured reward is non-decreasing because the solution space for smaller value of m is a subset of solution space for larger value of m. This demonstrates a significance of considering facilities with adjustable QoS.



DZ	lower left corner (x_{d_i}, y_{d_i})	width w_{d_i}	length l_{d_i}	Reward rate $v_i^z = v_i/z$			
				$v_i^{z_1}$	$v_i^{z_2}$	$v_i^{z_3}$	$v_i^{z_4}$
d_1	(0,7)	7	5	4	2	1.33	1
d_2	(1,0)	5	10	6	3	2	1.5
d_3	(5,5)	6	6	7	3.5	2.33	1.75
d_4	(4,3)	6	6	3	1.5	1	0.75
d_5	(9,2)	6	5	10	5	3.33	2.5

Dimension of base SZ s_0		scaling factor	SZ dimension
w_{s_0}	l_{s_0}	$z \in \Xi$	(w_s^z, l_s^z)
2	3	ξ_1=1	(2,3)
		ξ_2=2	(4,6)
		ξ_3=3	(6,9)
		ξ_4=4	(8,12)

$\lvert\Xi\rvert = m$	Greedy Solution		Optimal Solution	
	$(x_{s_1}, y_{s_1}, z_{s_1})$	$(x_{s_2}, y_{s_2}, z_{s_2})$	$(x_{s_1}, y_{s_1}, z_{s_1})$	$(x_{s_2}, y_{s_2}, z_{s_2})$
1	(5,7,1)	(9,4,1)	(5,7,1)	(9,4,1)
2	(4,5,2)	(9,2,2)	(4,5,2)	(9,2,2)
3	(5,2,3)	(1,4,2)	(1,2,3)	(7,2,3)
4	(3,0,4)	(11,1,2)	(1,0,4)	(9,2,3)

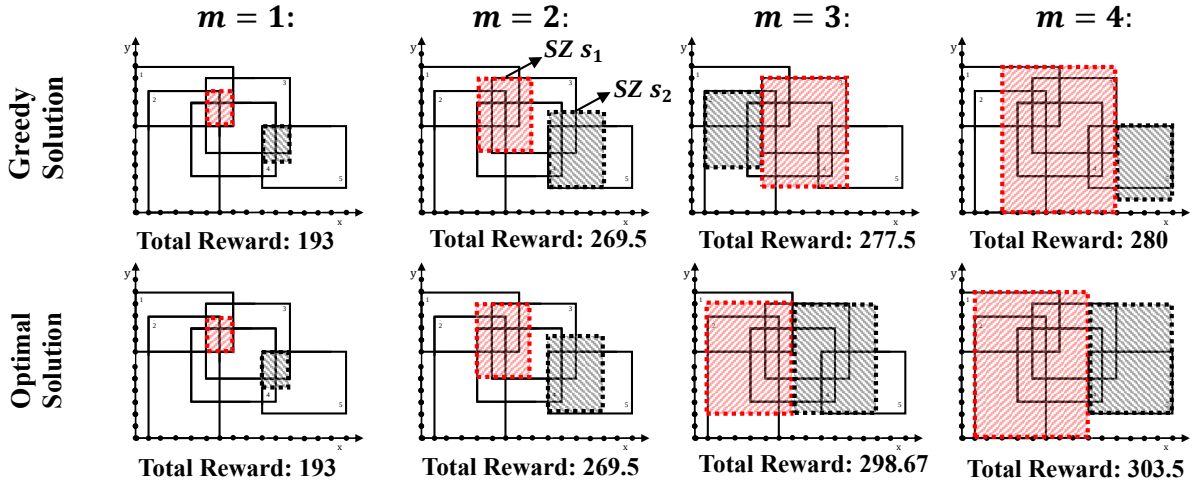


Figure 2: An example of PMCLP-PC-QoS with a set of five axis-parallel rectangular DZs $\{d_1, d_2, \ldots, d_5\}$ with known location, width, length, and base reward rates, a base SZ s_0, and a set of scaling factors $\Xi = \{\xi_1, \ldots, \xi_m\}$ for two SZs, i.e. $p = 2$. The optimal and greedy solutions for $m \in \{1, \ldots, 4\}$ are obtained from algorithms presented in this paper.

1.2 Other Applications of the PMCLP-PC-QoS

The PMCLP-PC-QoS also has direct applications in variety of emerging domains:

(a) *Telerobotics.* The advent of network-based telerobotic camera systems enable multiple participants or researchers in space exploration, health-care, and distance learning, to interact with a remote physical environment using shared resources. This system of p networked robotic cameras with discrete resolutions receives rectangular requests or subregions (DZs) from multiple users for monitoring (Song et al. 2006, Xu et al. 2010, 2008). Each request has an associated reward rate (per-unit area) that may depend on the priority of the user or the importance level associated with monitoring that subregion and the resolution level (QoS) of the camera utilized to cover the subregion. The goal is to select the best view frame for the cameras (rectangular SZs) and their resolution (QoS) to maximize the total reward from the captured parts of the requested subregions. Interestingly, this problem is same as the PMCLP-PCR-QoS. In the literature, Song et al. (2006) studied the PMCLP-PCR-QoS for single camera, i.e. $p = 1$, and Xu et al. (2010, 2008) considered the PMCLP with rectangular DZs and binary coverage where the rectangular SZs are not allowed to overlap. The PMCLP-PCR-QoS subsumes the foregoing problems.

(b) *River Cleaning.* The main motivation behind studying the 1D-PMCLP-PC-QoS is its application in locating trash-booms for river cleanup. Rivers are the primary recipient of stormwater and as a result, the amount of trash, floating debris, and litter in them is rapidly increasing. The major sources along the river that contribute to the trash problem include industrial and recreational parks, and tributaries. There are numerous on-going efforts to clean rivers in the US and also across the globe, for example, Nile in Egypt, Yangtze in China, Ganga in India, etc. One way to reduce/cover trash in rivers is by appropriately locating trash-booms in and along the rivers. Since the point representation of a trash zone does not take into account the displacement of the trash from source locations. Therefore, an analogous of 1D-PMCLP-PC-QoS is a trash-boom location problem, where a river is represented by a one-dimensional axis, and trash zones (or DZs) and coverage of trash-booms (SZs) are represented by line segments. In this problem, the service range of each trash boom is known but can be different, and the partial coverage is allowed.

1.3 Organization of this Paper

In Section 2, we discuss challenges in solving PMCLP-PC-QoS using well-known approaches for solving (planar) MCLP and its variants. We present a greedy algorithm and a pseudo-greedy algorithm for solving the PMCLP-PC-QoS (Section 3), and showcase that the solution value corresponding to the greedy solution is within a factor of $1 - 1/e$ of the optimal solution value where e is the base of natural logarithm. This ex-

tends the similar results of Cornuéjols et al. (1977) and Hochbaum and Pathria (1998) for special cases of PMCLP-PC-QoS (see Section 2.2 for details). In Section 4, we analyze the objective function and solution space of the PMCLP-PCR-QoS. We strengthen the theoretical properties provided for PMCLP-PCR in Bansal and Kianfar (2017) by reducing the solution search space and thereby, improve the computational efficiency of their algorithm. We also introduce theoretical properties to reduce search space for optimal solution of PMCLP-PCR-QoS, and utilize these properties to develop a branch-and-bound based exact algorithm for it. In Section 5, we introduce 1D-PMCLP-PC-QoS and provide theoretical properties for its solution space along with an exact algorithm for it. We also conduct computational experiments to evaluate the performance of our exact algorithms and greedy approach for PMCLP-PCR, PMCLP-PCR-QoS, and 1D-PMCLP-PC-QoS (Section 6). Finally, we provide our concluding remarks in Section 7.

2 Challenges in solving PMCLP-PC-QoS

In this section, we present two well-known approaches for solving (planar) MCLP and its variants, and discuss how they cannot be directly utilized for solving the PMCLP-PC-QoS.

2.1 Linear Binary Programming

The motivation for the binary coverage assumption in the (planar) MCLP (where demand is represented by points, line segments, or polygons) is to make the problem manageable by readily formulating them as a linear binary program (LBP). This is easy to do for the MCLP because of the discrete nature of candidate locations for service facilities (as per the definition). Moreover, even in studies considering a planar setting, i.e., allowing the facilities to be located anywhere in the continuous plane, coverage is still assumed to be binary (Murray and Tong 2007) because this helps to show that a finite number of potential facility locations, called the circle intersection point set (CIPS) (Church 1984) and polygon intersection point set (PIPS) (Murray and Tong 2007), exist which contain an optimal solution to this problem. Thereby resulting in the following well-known LBP for the (planar) MCLP:

$$\max\left\{\sum\nolimits_i v_i x_i : \sum\nolimits_j a_{ij} y_j \geq x_i \text{ for all } i,\ \ \sum\nolimits_j y_j = p,\ \ x_i, y_j \in \{0,1\} \text{ for all } i,j\right\}, \quad (2)$$

where binary variable $x_i = 1$ if demand zone i is covered and $x_i = 0$ otherwise, variable $y_j = 1$ if a service facility is sited at the candidate/PIP/CIP point j and $y_i = 0$ otherwise, v_i is the given total demand of demand zone i, and a_{ij} is the given binary value which is

1 if demand zone i is covered by locating a facility at candidate/CIP/PIP point j, i.e., distance between point i is no greater than known service range of a facility (denoted by r) located at point j or $dist(i,j) \leq r$.

Furthermore in literature, LBP formulations have also been used to tackle so-called gradual coverage location problem (GCLP). Similar to the MCLP, in GCLP, the demand zones are still represented by points but the coverage level depends on their distance from the facilities. So far LBP formulations are known only for the GCLP defined over a finite set of pre-specified candidate positions for the facilities (Berman and Krass 2002, Berman et al. 2003, Church and Roberts 1983), and planar GCLP with single facility, i.e. $p = 1$ (Drezner et al. 2004). More specifically, given a set of n demand points and a set $\mathcal{Y}$ of finite number of positions where p facilities can be placed, the GCLP can be stated as follows: $\max_{F \subseteq Y} \left\{ \sum_{i=1}^{n} v_i g_i(\Delta_i(F)) : |F| = p \right\}$ where F is a set of locations where facilities are located, $\Delta_i(F) = \min_{j \in F}\{dist(i,j)\}$ is the minimum distance between demand point i and any facility location in F, and $g_i(.) \in [0,1]$ is a pre-defined coverage function. The following LBP formulations have been derived for the GCLP with linear decay coverage function (Berman et al. 2003, Drezner et al. 2004), i.e., $g_i(\Delta) = 1 - \beta\Delta$ where $\beta > 0$ is a constant, or step-coverage function (Berman and Krass 2002, Church and Roberts 1983), i.e., $g_i(\Delta) = \delta_k$ if $\Delta \in (r^i_{k-1}, r^i_k]$ for $k = 1, \ldots, K$ where $\delta_1 = 1 > \delta_2 > \ldots > \delta_K = 0$ (coverage levels) and $r^i_0 = 0 < r^i_1 = r < r^i_2 < \ldots < r^i_K$ (coverage radii):

$$\max \left\{ \sum_{i=1}^{n} \sum_{j \in Y} c_{ij} x_{ij} \Big| \;\; a_{ij} y_j \geq x_{ij}, \; \forall i,j; \; \sum_{j \in Y} a_{ij} x_{ij} \leq 1 \; \forall i; \right.$$
$$\left. \sum\nolimits_{j \in Y} y_j = p, x_{ij}, y_j \in \{0,1\}, \; \forall i,j \right\} \tag{3}$$

where $c_{ij} = v_i g_i(dist(i,j))$, $x_{ij} = 1$ if facility located at point j covers the demand point i and $x_{ij} = 0$ otherwise, and a_{ij} is the given binary value which is 1 if demand at point i can be partially/completely covered by locating a facility at candidate point j, i.e., $dist(i,j) < r^i_K$. Note that if $g_i(\Delta) = 1$ for $\Delta \leq r$ and $g_i(\Delta) = 0$ for $\Delta > r$, then the GCLP reduces to the MCLP.

PMCLP-PC-QoS with its features of partial coverage, adjustable QoS, and general spatial representation of DZs and SZs is significantly harder to solve compared to the (planar) MCLP problem with binary coverage, even when a demand zone is represented by a line segment or polygon, and the GCLP. Using LBPs to solve the PMCLP-PC-QoS is no more feasible.

2.2 Greedy-Based Algorithms

Since even MCLP is an NP-hard problem, approximation algorithms have also been developed in the literature for solving (planar) MCLP. Among them, greedy approximation algorithm is a well-known approach because it requires solving single facility problem for multiple (p) times. In this direction, Cornuéjols et al. (1977) and Hochbaum and Pathria (1998) provided greedy algorithms for solving variants of MCLP along with their approximation ratios. More specifically, in a seminal paper on locating bank accounts (or facilities) in at most p out of m known cities to cover n clients (or DZs represented by points), Cornuéjols et al. (1977) considered a variant of MCLP (or GCLP) in which fixed cost of locating accounts is also deducted in the objective function and it is assumed that the coverage function $g_i(dist(i,j)) = \phi_{ij}$ is constant for each pair of client i and facility location j. They derived an LBP formulation which is same as (3) when the fixed costs are zero, and also presented a greedy approach which provides a solution whose value is within a factor of $1-1/e$ of the optimal solution value. Clearly with zero fixed costs, this problem is a special case of PMCLP-PC-QoS.

Hochbaum and Pathria (1998) extended the results in Cornuéjols et al. (1977) by considering a so-called maximum p-coverage problem (MCP) which is defined as follows: Given a universal set of elements U where each element $i \in U$ has weight v_i associated to it and a class $\mathcal{V}$ of subsets of U, the goal is to select p members (or subsets of U) from the class $\mathcal{V}$ such that the sum of the weights of the elements in the union of these subsets is maximum. We can build its correspondence with facility location problem, in particular PMCLP-PC-QoS, by considering each element of the set U as a DZ and each member of the class $\mathcal{V}$ as a set of DZs that are completely covered by a SZ located at a pre-specified candidate position and have fixed QoS. Observe that the foregoing problem does not allow partial coverage of elements/DZs and restricts selected members to belong to the class $\mathcal{V}$, which is equivalent to SZs to be located at pre-specified candidate positions. This implies that the MCP is a special case of PMCLP-PC-QoS. Hochbaum and Pathria (1998) provided greedy and pseudo-greedy algorithms for MCP, and showcase that the solution value corresponding to the greedy (or pseudo-greedy) solution is within a factor of $1-1/e$ (or $1-1/e^{\eta}$) of the optimal solution value where e is the base of natural logarithm and $\eta \leq 1$. In this paper, we further extend their algorithmic and theoretical results for PMCLP-PC-QoS. Note that because of the binary coverage assumption of MCP and discrete nature of the bank account location problem, the results in Hochbaum and Pathria (1998) and Cornuéjols et al. (1977), respectively, cannot be directly applied on the PMCLP-PC-QoS.

3 Approximation Algorithms for PMCLP-PC-QoS

In this section, we present greedy and pseudo-greedy algorithms for PMCLP-PC-QoS with $\Xi_s = \Xi = \{\xi_1, \ldots, \xi_m\}$ for all $s \in \mathcal{S}$.

Greedy Algorithm. Assuming that there exists an exact algorithm for solving the PMCLP-PC-QoS with $p = 1$, referred to as Single SZ Problem (SSP), we solve multiple SSPs in our greedy-based algorithm for the PMCLP-PC-QoS. The pseudocode is presented in Algorithm 1, where for a set of DZs $\mathcal{D}_j^g$, the function $\texttt{SingleSZProblem}(\mathcal{D}_j^g, s_j)$, $j \in \{1, \ldots, p\}$, returns the maximum reward, ψ_g^j, covered by the SZ s_j along with its optimal position $(x_{s_j}^g, y_{s_j}^g)$ and QoS $z_{s_j}^g$ (Line 4). We initialize the algorithm in Line 2 by setting $\mathcal{D}_1^g = \mathcal{D}$ (the original set of all given DZs). We also use the function $\texttt{TrimOut}(d, s_j, x_{s_j}^g, y_{s_j}^g, z_{s_j}^g)$ to eliminate the parts of DZ d that are covered by SZ s_j positioned at $(x_{s_j}^g, y_{s_j}^g)$ with dimensions $z_{s_j}^g$ times the dimensions of base SZ s_0. In Lines 6-8, we create set $\mathcal{D}_{j+1}^g$ for the next iteration by replacing each DZ d in the set $\mathcal{D}_j^g$ with trimmed DZs. We denote the set of trimmed DZs, that replaces DZ d, by T_d. The summation of the maximum covered reward by calling $\texttt{SingleSZProblem}(\mathcal{D}_j^g, s_j)$ over $j \in \{1, \ldots, p\}$ gives a feasible solution and a lower bound on the optimal objective value of the PMCLP-PC-QoS. Algorithm 1 generalizes the greedy-based polynomial-time heuristic of Bansal and Kianfar (2017) for solving the PMCLP-PCR in $O(n^2p^3m)$, and Theorem 1 provides an approximation ratio for this special case as well.

Algorithm 1 Greedy Algorithm for PMCLP-PC-QoS

1: **function GreedyAlgorithm**($\mathcal{D}$, $\mathcal{S}$)
2: $\quad \mathcal{D}_1^g := \mathcal{D}; \psi_g := 0;$
3: $\quad$ **for** $j = 1, \ldots, p$ **do**
4: $\quad\quad (\psi_g^j, x_{s_j}^g, y_{s_j}^g, z_{s_j}^g) := \texttt{SingleSZProblem}\left(\mathcal{D}_j^g, s_j\right);$
5: $\quad\quad \psi_g \leftarrow \psi_g + \psi_g^j;$
6: $\quad\quad$ **for** $d \in \mathcal{D}_j^g$ **do**
7: $\quad\quad\quad \mathcal{D}_{j+1}^g \leftarrow \{\mathcal{D}_j^g \setminus d\} \cup \texttt{TrimOut}\left(d, s_j, x_{s_j}^g, y_{s_j}^g, z_{s_j}^g\right);$
8: $\quad$ **return** $\left(\psi_g, x_{s_1}^g, \ldots, x_{s_p}^g, y_{s_1}^g, \ldots, y_{s_p}^g, z_{s_1}^g, \ldots, z_{s_p}^g\right)$

Observation 1. *In iteration $j \in \{1, \ldots, p\}$ of the greedy algorithm (Algorithm 1), we exactly solve a SSP for $\mathcal{D}_j^g$ set of DZs. Observe that summing the demand covered in the iteration, i.e., ψ_g^j, for p times provides an upper bound on the optimal solution value of the PMCLP-PC-QoS with $\Xi_{s_j} = \Xi$, $s_j \in \mathcal{S}$, for $\mathcal{D}_j^g$ set of DZs. This is because the summation does not consider overlapping of the SZs.*

Pseudo-Greedy Algorithm. In the greedy algorithm, we assume that an exact algorithm for solving the SSP is known. However, in case this assumption fails and only an η-approximate algorithm for solving the SSP is known, then we utilize a pseudo-greedy algorithm for solving the PMCLP-PC-QoS. The pseudo-greedy algorithm is same as the greedy algorithm (Algorithm 1) except that the function `SingleSZProblem` is replaced by function η-`ApproxSSP` which returns an approximate demand, κ_r^j, covered by the SZ s_j that is within a known factor of η (≤ 1) of the optimal solution value returned by `SingleSZProblem`. The pseudocode is given in Algorithm 2. Note that for $\eta = 1$, the pseudo-greedy algorithm is exactly same as the greedy algorithm. Furthermore, for $\eta < 1$, the sets of DZs $\mathcal{D}_j^g$ and $\mathcal{D}_j^r$, $j = 2, \ldots, p$, in the greedy algorithm (Algorithm 1) and pseudo-greedy algorithm (Algorithm 2), respectively, are different.

Algorithm 2 Pseudo-Greedy Method

function PseudoGreedyAlgorithm($\mathcal{D}$, $\mathcal{S}$)
  $\mathcal{D}_1^r$:= $\mathcal{D}$; κ_r := 0;
  for j = 1, ..., p **do**
    $(\kappa_r^j, x_{s_j}^r, y_{s_j}^r, z_{s_j}^r)$:= η-`ApproxSSP`$\left(\mathcal{D}_j^r, s_j\right)$;
    $\kappa_s \leftarrow \kappa_r$ + κ_r^j;
    for $d \in \mathcal{D}_j^r$ **do**
      $\mathcal{D}_{j+1}^r \leftarrow \{\mathcal{D}_j^r \setminus d\} \cup$ `TrimOut`$\left(d, s_j, x_{s_j}^r, y_{s_j}^r, z_{s_j}^r\right)$;
  return $\left(\kappa_r, x_{s_1}^r, \ldots, x_{s_p}^r, y_{s_1}^r, \ldots, y_{s_p}^r, z_{s_1}^r, \ldots, z_{s_p}^r\right)$

Remark 1. *The greedy (or pseudo-greedy) algorithm depends on* `SingleSZProblem` *(or* η-`ApproxSSP`*) and* `TrimOut`*, and the number of trimmed DZs generated after each iteration. Till date these functions are not known, except for PMCLP-PCR-QoS with $p = 1$ (Bansal and Kianfar 2017, Song et al. 2006). However, whenever they will be developed in future, we can embed them within our greedy (or pseudo-greedy) algorithm to provide solutions for the PMCLP-PC-QoS along with their computational complexity.*

3.1 Approximation Ratios

In the following theorem, we provide approximation ratios associated with the greedy and pseudo-greedy algorithms for the PMCLP-PC-QoS. Let $(\mathbf{x}, \mathbf{y}, \mathbf{z}) \in \mathsf{R}^{2p} \times \Xi$ be an optimal solution and $(\mathbf{x}^a, \mathbf{y}^a, \mathbf{z}^a) \in \mathsf{R}^{2p} \times \Xi$ be an approximate solution for the PMCLP-PC-QoS. Then the approximation ratio corresponding to the approximate solution (or algorithm) is $\gamma_a = \frac{f(\mathbf{x}^a, \mathbf{y}^a, \mathbf{z}^a)}{f(\mathbf{x}, \mathbf{y}, \mathbf{z})}$.

Theorem 1. *Let the approximation ratios for the greedy algorithm (Algorithm 1) and the*

pseudo-greedy algorithm (Algorithm 2) be denoted by γ_g and γ_r, respectively. Then

$$\gamma_g > 1 - \frac{1}{e} \text{ and } \gamma_r > 1 - \frac{1}{e^\eta}, \tag{4}$$

where e is the base of natural logarithm.

Proof. Recall that the notations ψ_g^j in Algorithm 1 and κ_r^j in Algorithm 2 denote the covered reward returned by the functions $\texttt{SingleSZProblem}(\mathcal{D}_j^g, s_j)$ and $\eta\text{-}\texttt{ApproxSSP}(\mathcal{D}_j^r, s_j)$, respectively, for $j \in \{1, \ldots, p\}$. In other words, ψ_g^j is the maximum reward and κ_r^j is the η-approximate reward covered by the SZ s_j for the given set of DZs $\mathcal{D}_j^g$ and $\mathcal{D}_j^r$, respectively. This implies that $\kappa_r^j \geq \eta\psi_r^j$ where ψ_r^j is the maximum reward covered by the SZ s_j for $\mathcal{D}_j^r$ set of DZs. For the sake of convenience, in this proof, we denote the optimal solution value for the PMCLP-PC-QoS instance, i.e., $f(\mathbf{x}, \mathbf{y}, \mathbf{z})$, by f. Therefore, the approximation ratio for the greedy algorithm (Algorithm 1) and the pseudo-greedy algorithm (Algorithm 2) are given by

$$\gamma_g = \frac{1}{f}\left(\sum_{l=1}^{p} \psi_g^l\right) \text{ and } \gamma_r = \frac{1}{f}\left(\sum_{l=1}^{p} \kappa_r^l\right), \tag{5}$$

respectively. Let the optimal solution value of PMCLP-PC-QoS instance with $\mathcal{D}_j^g$ (or $\mathcal{D}_j^r$) as the input set of DZs be denoted by ζ_j^g (or ζ_j^r). Then, based on Observation 1,

$$p\psi_g^j \geq \zeta_j^g \text{ and } p\kappa_r^j \geq p\eta\psi_r^j \geq \eta\zeta_j^r \tag{6}$$

where ψ_r^j is the covered reward returned by $\texttt{SingleSZProblem}(\mathcal{D}_j^r, s_j)$, for $j = 1, \ldots, p$. Note that for $j = 1$, $\mathcal{D}_1^g = \mathcal{D}_1^r = \mathcal{D}$, and hence $\zeta_1^g = \zeta_1^r = f$. Also, since after each iteration $l \in \{1, \ldots, j-1\}$ of the greedy algorithm and pseudo-greedy algorithm, we "trim-out" DZs of total reward ψ_g^l and κ_r^l, respectively, we get

$$p\psi_g^j \geq \zeta_j^g \geq f - \sum_{l=1}^{j-1} \psi_g^l \text{ and } p\kappa_r^j \geq \eta\zeta_j^r \geq \eta\left(f - \sum_{l=1}^{j-1} \kappa_r^l\right), \tag{7}$$

where $\psi_g^0 = \kappa_r^0 = 0$. This implies that

$$p\sum_{l=1}^{j} \psi_g^l \geq f + \left(p-1\right)\sum_{l=1}^{j-1} \psi_g^l \geq f\left(1 + \frac{p-1}{p} + \left(\frac{p-1}{p}\right)^2 + \ldots + \left(\frac{p-1}{p}\right)^{j-1}\right) \tag{8}$$

and

$$p\sum_{l=1}^{j}\kappa_r^l \geq \eta f \ + \left(p-\eta\right)\sum_{l=1}^{j\ \ 1}\kappa_r^l \geq \eta f \ \left(1+\frac{p-\eta}{p}+\left(\frac{p-\eta}{p}\right)^{2}+\ldots+\left(\frac{p-\eta}{p}\right)^{j\ \ 1}\right). \tag{9}$$

For $j = p$, Inequalities (8) and (9) reduce to

$$\gamma_g = \frac{1}{f}\left(\sum_{l=1}^{p}\psi_g^l\right) \geq \left(1-\left(\frac{p-1}{p}\right)^{p}\right) \text{ and } \gamma_r = \frac{1}{f}\left(\sum_{l=1}^{p}\kappa_r^l\right) \geq \left(1-\left(\frac{p-\eta}{p}\right)^{p}\right),$$

respectively. Now because the functions in the right-hand sides of the last two inequalities are decreasing in p, we compute limit of these functions as p approaches infinity and get $\gamma_g > 1 - 1/e$ and $\gamma_r > 1 - 1/e^{\eta}$.

Corollary 1. *For a given $p \geq 1$, the greedy and pseudo-greedy algorithms provide solutions whose values are at least $1-[(p-1)/p]^p$ and $1-[(p-\eta)/p]^p$, respectively, times the optimal value for the PMCLP-PC-QoS instance with $\Xi_s = \Xi$ for all $s \in \mathcal{S}$.*

4 Exact Algorithm for PMCLP-PCR-QoS

In this section, we analyze the objective function of PMCLP-PCR-QoS with axis-parallel rectangular DZs and SZs. We prove some theoretical properties that help us validate the reduction of search space for an optimal solution. We also present an implicit enumeration technique, a customized branch-and-bound based exact algorithm, to solve PMCLP-PCR-QoS. Then, we computationally evaluate performance of this approach. As mentioned before, Bansal and Kianfar (2017) presented a branch-and-bound algorithm and theoretical properties of the objective function for PMCLP-PCR-QoS with $\Xi = \{1\}$, i.e., all SZs have same and fixed dimensions or QoS (denoted by PMCLP-PCR). Our algorithm not only generalizes their approach, but we further strengthen the theoretical properties provided for PMCLP-PCR, thereby improving the computational efficiency of their approach.

4.1 Theoretical Properties for PMCLP-PCR-QoS

We first extend some definitions introduced for PMCLP-PCR (Bansal and Kianfar 2017) and introduce some new definitions.

Definition 1 (DZ Critical Values, denoted by DCVs)**.** *For each DZ d_i and scaling factor*

$z \in \Xi$, we define a set of x DCVs and a set of y DCVs as follows.

$$X_D^{d_i,z} := \left\{x_{d_i,z}^{O1} = x_{d_i} - w_{s_0} z, \quad x_{d_i,z}^{I1} = x_{d_i}, \quad x_{d_i,z}^{I2} = x_{d_i} + w_{d_i} - w_{s_0} z, \quad x_{d_i,z}^{O2} = x_{d_i} + w_{d_i}\right\},$$
$$Y_D^{d_i,z} := \left\{y_{d_i,z}^{O1} = y_{d_i} - l_{s_0} z, \quad y_{d_i,z}^{I1} = y_{d_i}, \quad y_{d_i,z}^{I2} = y_{d_i} + l_{d_i} - l_{s_0} z, \quad y_{d_i,z}^{O2} = y_{d_i} + l_{d_i}\right\}.$$

We denote the set of all x DCVs and the set of all y DCVs for a fixed scaling factor $z \in \Xi$ by $X_D^z = \bigcup_i X_D^{d_i,z}$ and $Y_D^z = \bigcup_i Y_D^{d_i,z}$, respectively. Likewise, we denote the set of all possible x DCVs (or y DCVs) by $X_D = \bigcup_{z2} \ X_D^z$ (or $Y_D = \bigcup_{z2} \ Y_D^z$).

Definition 2 (Inner and Outer DCVs, denoted by IDCVs and ODCVs, respectively)**.** *We classify DCVs into two categories: Inner DCVs and Outer DCVs. Specifically, for a given DZ d and scaling factor z, $x_{d,z}^{O1}$ and $x_{d,z}^{O2}$ are x ODCVs and $x_{d,z}^{I1}$ and $x_{d,z}^{I2}$ are x IDCVs. Let $X_{ID}^z := \{x_{d,z}^{I1}, x_{d,z}^{I2}\}_{d2D}$ and $X_{OD}^z := \{x_{d,z}^{O1}, x_{d,z}^{O2}\}_{d2D}$, for $z \in \Xi$. We denote the set of all possible x IDCVs and x ODCVs by $X_{ID} = \cup_{z2} \ X_{ID}^z$ and $X_{OD} = \cup_{z2} \ X_{OD}^z$, respectively. We also define*

$$D_{IX}^z(x) = \left\{i \in N : x \in \left\{x_{d_i,z}^{I1}, x_{d_i,z}^{I2}\right\}\right\} \quad \text{and} \quad D_{OX}^z(x) = \left\{i \in N : x \in \left\{x_{d_i,z}^{O1}, x_{d_i,z}^{O2}\right\}\right\}$$

for given $x \in \mathsf{R}$ and $z \in \Xi$. Likewise, sets Y_{ID}^z , Y_{OD}^z, Y_{ID}, Y_{OD}, $D_{IY}^z(y)$, and $D_{OY}^z(y)$ are defined for y DCVs.

We consider a DZ d_i and a SZ s_1 with known $y_{s_1} = \hat{y}_{s_1}$ (y coordinate of its lower left corner). Figure 3 shows that $f_i(x_{s_1}, \hat{y}_{s_1}, \xi_1)$, $f_i(x_{s_1}, \hat{y}_{s_1}, \xi_2)$ and $f_i(x_{s_1}, \hat{y}_{s_1}, \xi_3)$, i.e., reward function in (1) for a single SZ with fixed scaling factor $z_{s_1} \in \Xi = \{\xi_1, \xi_2, \xi_3\}$, are piecewise linear functions of x_{s_1} with breakpoints at x DCVs, i.e., $X_D^{d_i,z_{s_1}}$. Now, note that $\max_{z_{s_1} 2} \ f_i(x_{s_1}, \hat{y}_{s_1}, z_{s_1})$ is also piecewise linear (represented by bold blue line segments in Figure 3) which has breakpoints not only at x DCVs, but also at so-called x QoS critical values (x QCVs). A set of x QCVs for DZ d_i, denoted by $X_Q^{d_i}$, includes all break points $\hat{x}_{s_1} \notin X_D$ of $\max_{z_{s_1} 2} \ f_i(x_{s_1}, \hat{y}_{s_1}, z_{s_1})$ where functions $f_i(x_{s_1}, \hat{y}_{s_1}, \hat{z}_{s_1})$ for different QoS or $\hat{z}_{s_1} \in \Xi$ intersect. We denote the set of all x QCVs by $X_Q = \bigcup_i X_Q^{d_i}$, and define $D_{QX}(x) = \left\{i \in N : x \in X_Q^{d_i}\right\}$ for given $x \in \mathsf{R}$. Likewise, sets $Y_Q^{d_i}$, $Y_Q = \cup_i Y_Q^{d_i}$ and $D_{QY}(y)$ are defined for y QCVs.

Observation 2. *Function $\max_{z_{s_1} 2} \ f_i(x_{s_1}, \hat{y}_{s_1}, z_{s_1})$ has locally convex break point at $x_{s_1} \in X_Q^{d_i}$ because maximum of two (intersecting) affine functions is locally convex at intersecting point.*

Definition 3 (SZ Critical Values, denoted by SCVs)**.** *Given a scaling factor $z \in \Xi$ and a set of positioned SZs $\{s_j\}_{j2J}$, $\mathcal{J} \subset P$, such that each SZ s_j, $j \in \mathcal{J}$, has known coordinates of its lower left corner (x_{s_j}, y_{s_j}) and QoS z_{s_j}. For $j \in \mathcal{J}$ and $z \in \Xi$, we define a set of x*

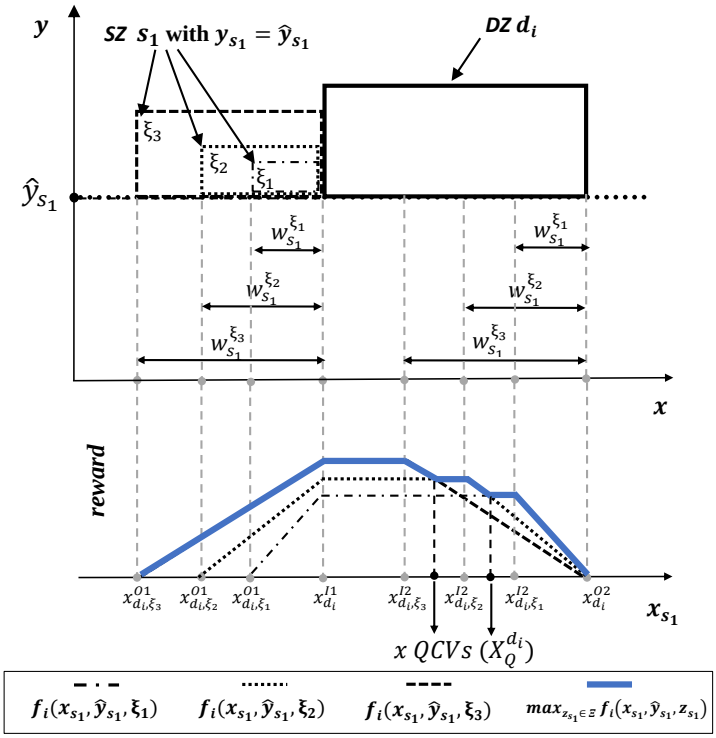


Figure 3: An example of reward function with break points at DCVs and QCVs for DZ d_i, SZ s_1 (with fixed $y_{s_1} = \hat{y}_{s_1}$) and $\Xi = \{\xi_1, \xi_2, \xi_3\}$.

SCVs and a set of y SCVs as follows.

$$\mathcal{X}_S^{s_j,z} := \{x_{s_j,z}^{O1} = x_{s_j} - w_{s_0} z,\quad x_{s_j,z}^{I1} = x_{s_j},\quad x_{s_j,z}^{I2} = x_{s_j} + w_{s_0} z_{s_j} - w_{s_0} z,\quad x_{s_j,z}^{O2} = x_{s_j} + w_{s_0} z_{s_j}\},$$
$$\mathcal{Y}_S^{s_j,z} := \{y_{s_j,z}^{O1} = y_{s_j} - l_{s_0} z,\quad y_{s_j,z}^{I1} = y_{s_j},\quad y_{s_j,z}^{I2} = y_{s_j} + l_{s_0} z_{s_j} - l_{s_0} z,\quad y_{s_j,z}^{O2} = y_{s_j} + l_{s_0} z_{s_j}\}.$$

We denote the set of all x SCVs for a specific scaling factor $z \in \Xi$ by $\mathcal{X}_S^z(\mathcal{J}) = \bigcup_{j2J} \mathcal{X}_S^{s_j,z}$. The set of all x SCVs is denoted by $\mathcal{X}_S(\Xi, \mathcal{J}) = \bigcup_{z2} \bigcup_{j2J} \mathcal{X}_S^{s_j,z}$. Likewise, the set $\mathcal{Y}_S^z(\mathcal{J})$ and $\mathcal{Y}_S(\Xi, \mathcal{J})$ are defined for y SCVs. Notice that when $\Xi = \{1\}$, $x_{s_j,z}^{I1} = x_{s_j,z}^{I2}$ and $y_{s_j,z}^{I1} = y_{s_j,z}^{I2}$ because $z = z_{s_j} = 1$ for all $j \in P$.

Definition 4 (Inner and Outer SCVs, denoted by ISCVs and OSCVs, respectively)**.** *Similar to DCVs, we also classify SCVs into two categories: Inner SCVs, i.e., $\mathcal{X}_{IS}^{s_j,z} := \{x_{s_j,z}^{I1}, x_{s_j,z}^{I2}\}$ and $\mathcal{Y}_{IS}^{s_j,z} := \{y_{s_j,z}^{I1}, y_{s_j,z}^{I2}\}$, and Outer SCVs, i.e., $\mathcal{X}_{OS}^{s_j,z} := \{x_{s_j,z}^{O1}, x_{s_j,z}^{O2}\}$ and $\mathcal{Y}_{OS}^{s_j,z} := \{y_{s_j,z}^{O1}, y_{s_j,z}^{O2}\}$, for $z \in \Xi$ and $j \in \mathcal{J}$. For a fixed scaling factor $z \in \Xi$, let $\mathcal{X}_{IS}^z(\mathcal{J}) = \bigcup_{j2J} \mathcal{X}_{IS}^{s_j,z}$ and $\mathcal{X}_{OS}^z(\mathcal{J}) = \bigcup_{j2J} \mathcal{X}_{OS}^{s_j,z}$. Also, we define sets $\mathcal{X}_{IS}(\Xi, \mathcal{J}) = \bigcup_{z2} \bigcup_{j2J} \mathcal{X}_{IS}^{s_j,z}$ and $\mathcal{X}_{OS}(\Xi, \mathcal{J}) = \bigcup_{z2} \bigcup_{j2J} \mathcal{X}_{OS}^{s_j,z}$. Similarly, $\mathcal{Y}_{IS}^z$, $\mathcal{Y}_{OS}^z$, $\mathcal{Y}_{IS}$ and $\mathcal{Y}_{OS}$ are defined for y SCVs.*

In Figure 4, we provide an example to illustrate how presence of multiple SZs impact

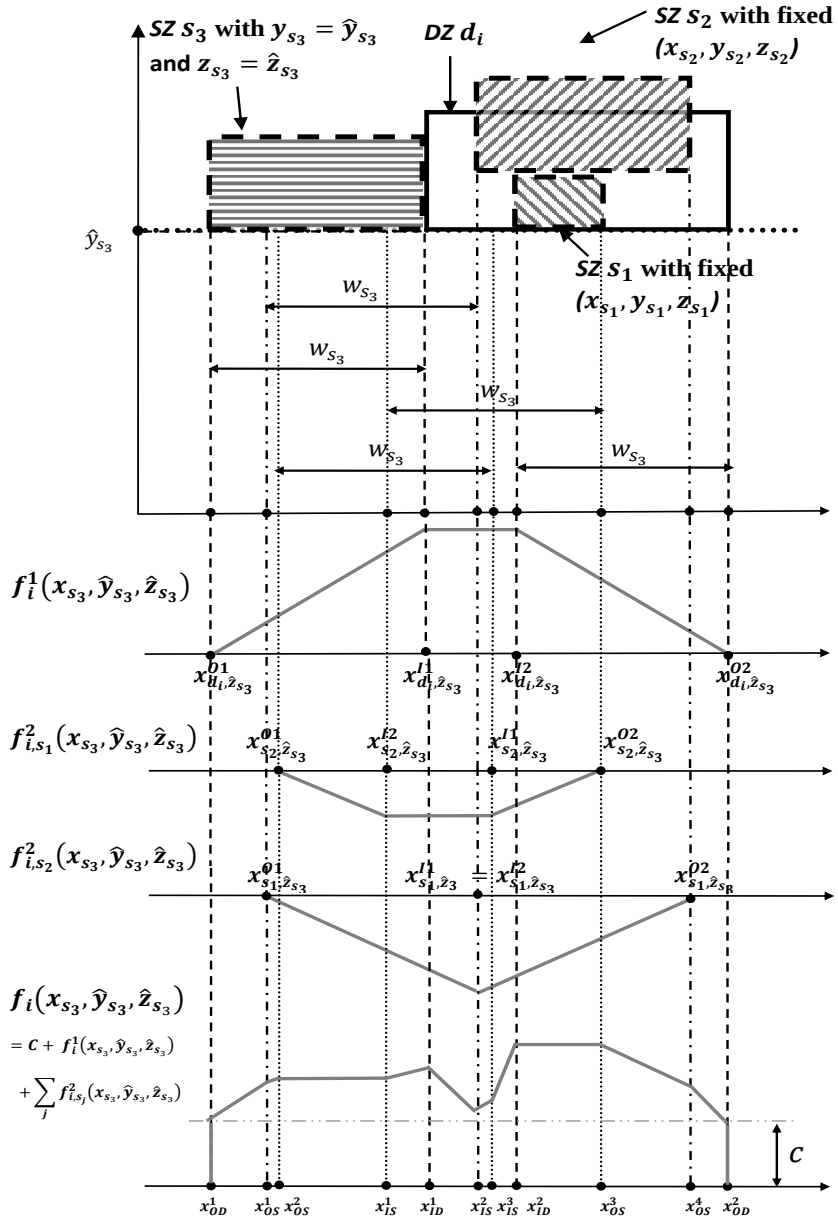


Figure 4: An example of piecewise linear reward function $f_i(\mathbf{x}, \mathbf{y}, \mathbf{z})$ in (1) with respect to argument x_{s_3} (i.e., x coordinate of lower left corner of SZ s_3) where s_3 has fixed scaling factor $\hat{z}_{s_3} \in \Xi$ and y coordinate of its lower left corner $\hat{y}_{s_3}$, and DZ d_i is partially captured by positioned SZ s_1 and SZ s_2 with fixed $(x_{s_1}, y_{s_1}, z_{s_1})$ and $(x_{s_2}, y_{s_2}, z_{s_2})$, respectively.

the reward function and give rise to SCVs. We consider DZ d_i being partially covered by SZs s_1 and s_2 with fixed $(x_{s_1}, y_{s_1}, z_{s_1})$ and $(x_{s_2}, y_{s_2}, z_{s_2})$, respectively, such that $z_{s_1} \leq z_{s_2}$. In the rest of the paper, if a variable is not mentioned in the list of arguments that $f_i(\mathbf{x}, \mathbf{y}, \mathbf{z})$ and $f(\mathbf{x}, \mathbf{y}, \mathbf{z})$ take, then it means it has a fixed value. For SZ s_3 with $y_{s_3} = \hat{y}_{s_3}$ and $z_{s_3} = \hat{z}_{s_3} \in \Xi$, we construct the following functions: $f_i^1(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3})$ is the reward function without considering overlap of DZ d_i with SZs s_1 and s_2. It is

a piecewise linear function with break points at x DCVs. To obtain $f_i(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3})$ in (1) that also considers overlap of DZ d_i with SZs s_1 and s_2, we define two negative reward functions $f^2_{i,s_1}(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3}) = -\min\{v_i^{\hat{z}_{s_3}}, v_i^{\hat{z}_{s_1}}\} \times A(s_3 \cap s_1)$ and $f^2_{i,s_2}(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3}) = -\min\{v_i^{\hat{z}_{s_3}}, v_i^{\hat{z}_{s_2}}\} \times A(s_3 \cap (s_2 \setminus s_1))$ for SZ s_1 and s_2, respectively, then we add them to $f^1_i(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3}) + c$, where $c = v_i^{\hat{z}_{s_1}} A(d_i \cap s_1) + v_i^{\hat{z}_{s_2}} A(d_i \cap (s_2 \setminus s_1))$ is the sum of reward captured by SZ s_1 and s_2 from DZ d_i. Observe that function $f^2_{i,s}(.)$, $s \in \{s_1, s_2\}$, is piecewise linear with break points at x SCVs belonging to $\mathcal{X}_S^{s,\hat{z}_{s_3}}$. In case $s_1 \cap s_2 \neq \emptyset$, $f^2_{i,s_2}(.)$ will have break points at x SCVs belonging to $\mathcal{X}_S^{s_1,\hat{z}_{s_3}} \cup \mathcal{X}_S^{s_2,\hat{z}_{s_3}}$. As a result, function $f_i(x_{s_3}, \hat{y}_{s_3}, \hat{z}_{s_3})$ is also piecewise linear with breakpoints at x_{s_3} belonging to x DCVs and x SCVs, i.e., $X_D^{d_i,\hat{z}_{s_3}} \cup \mathcal{X}_S^{\hat{z}_{s_3}}(\mathcal{J})$ where $\mathcal{J} = \{1, 2\}$.

Observation 3. *Given a SZ $s_t \in \mathcal{S}$ and a set of positioned SZs, $\{s_j\}_{j2J}$, $\mathcal{J} = P \setminus \{t\}$ such that each SZ s_j, $j \in \mathcal{J}$, has known coordinates of its lower left corner and QoS, function $f_i(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ at $x_{s_t} = \hat{x}_{s_t} \in X_{OD}^{\hat{z}_{s_t}} \cup X_{IS}^{\hat{z}_{s_t}}(\mathcal{J})$ either is linear with no break point or has a locally convex break point.*

Observation 4. *Assume that each SZ $s \in \mathcal{S}$ has fixed coordinates of its lower left corner $(\hat{x}_s, \hat{y}_s)$ with known scaling factor $\hat{z}_s$. For each DZ d_i, we define a function $g_i(x) = f_i(\hat{x}_{s_1}+x, \hat{x}_{s_2}+x, \ldots, \hat{x}_{s_p}+x, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ where $x \in \mathsf{R}^1$, i.e., a reward function obtained by moving all fixed SZs simultaneously and parallel to x axis. To analyze this function, we consider the following two cases. In case there is no overlap between any two SZs, i.e., $s_j \cap s_k = \emptyset$ for all $j, k \in P$ and $j \neq k$, it is easy to observe that $g_i(x)$ is a piecewise linear function of x and its break points (if exist) occur whenever $\hat{x}_s + x \in X_D^{d_i,\hat{z}_s}$ for any $s \in \mathcal{S}$. We demonstrate that these properties of $g(x)$ still hold, even in case there is overlap between SZs. Note that a set of overlapping rectangles (SZs) can be represented by non overlapping rectangles such that x (or y) coordinate of each new vertical (or horizontal, respectively) edge, if exists, of non-overlapping rectangles coincide with x (or y) coordinate of an edge of original overlapping SZs (see Figure 5). Consequently, the reward function obtained by moving these non overlapping rectangles simultaneously parallel to x axis also has break points (if any) whenever $\hat{x}_s + x \in X_D^{d_i,\hat{z}_s}$ for any $s \in \mathcal{S}$. The same is true for a reward function obtained by simultaneously moving all SZs parallel to y axis.*

Theorem 2. *There exists an optimal solution $(\mathbf{x}, \mathbf{y}, \mathbf{z}) = (x_{s_1}, \ldots, x_{s_p}, y_{s_1}, \ldots, y_{s_p}, z_{s_1}, \ldots, z_{s_p})$ where $(x_{s_j}, y_{s_j}, z_{s_j}) \in \mathsf{R}^2 \times \Xi$ for all $j = 1, \ldots, p$ such that the following conditions are satisfied:*

(i) *There exists a non-repetitive sequence $s_{t_1}, s_{t_2}, \ldots, s_{t_p} \in \mathcal{S}$ of the SZs such that $x_{s_{t_1}} \in X_{ID}$ and for $k = 2, \ldots, p$, $x_{s_{t_k}} \in X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_k)$ where $T_k = \{t_1, \ldots, t_{k-1}\}$ and $t_k \in P$.*

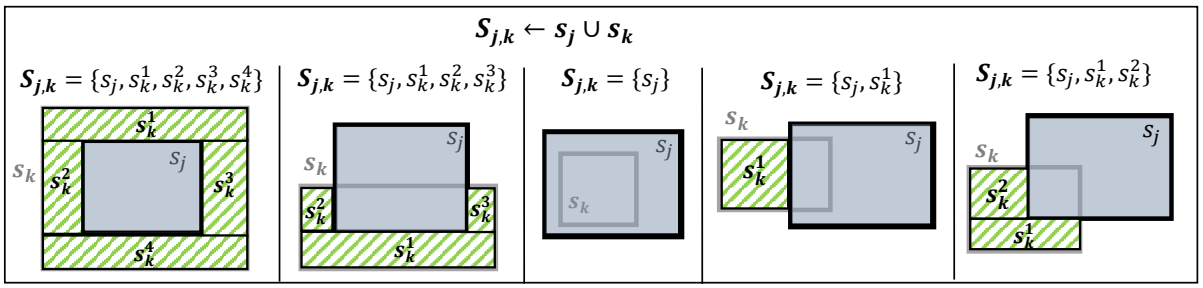


Figure 5: Representing two overlapping SZs (s_k and s_j) as a set of non overlapping rectangles

(ii) *There exists a non-repetitive sequence* $s_{q_1}, s_{q_2}, \ldots, s_{q_p} \in \mathcal{S}$ *of the SZs such that* $y_{s_{q_1}} \in Y_{ID}$ *and for* $k = 2, \ldots, p$, $y_{s_{q_k}} \in Y_{ID} \cup \mathcal{Y}_{OS}(\Xi, Q_k)$ *where* $Q_k = \{q_1, \ldots, q_{k-1}\}$ *and* $q_k \in P$.

In other words, x_{s_j} *is an x IDCVs or x OSCVs for all* $j \in P$ *and* $x_{s_t} \in X_{ID}$ *for at least one SZ* s_t, $t \in P$. *The same is true for* y_{s_j}, $j \in P$.

Proof. Assume there is an optimal solution $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}) = (\hat{x}_{s_1}, \ldots, \hat{x}_{s_p}, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ such that there is no $t \in P$ for which $\hat{x}_{s_t} \in X_{ID}$. For any $t \in P$, if $\hat{x}_{s_t} \in X_{OD}$ or $\hat{x}_{s_t} \in X_Q$, then $D_{OX}(\hat{x}_{s_t}) \neq \emptyset$ or $D_{QX}(\hat{x}_{s_t}) \neq \emptyset$, respectively. Thus, for $i \in D_{OX}(\hat{x}_{s_t})$, function $f_i(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ at $x_{s_t} = \hat{x}_{s_t}$ either is linear with no break point or has a locally convex break point (Observation 3). Also, for $i \in D_{QX}(\hat{x}_{s_t})$, function $f_i(x_{s_t}, \hat{y}_{s_t}, z_{s_t})$ has a locally convex break point at $x_{s_t} = \hat{x}_{s_t}$ and $z_{s_t} = \hat{z}_{s_t}$ (see Figure 3). Since there is no t for which $\hat{x}_{s_t} \in X_{ID}$, then it means for all $i \in N \setminus (D_{OX}(\hat{x}_{s_t}) \cup D_{QX}(\hat{x}_{s_t}))$, $f_i(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ is linear with no break point at $x_{s_t} = \hat{x}_{s_t}$. Therefore,

$$f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t}) = \sum_{i \in D_{OX}(\hat{x}_{s_t}) \cup D_{QX}(\hat{x}_{s_t})} f_i(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t}) + \sum_{i \in N \setminus (D_{OX}(\hat{x}_{s_t}) \cup D_{QX}(\hat{x}_{s_t}))} f_i(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$$

either (a) has a locally convex break point at $x_{s_t} = \hat{x}_{s_t}$ that contradicts the optimality of $(\hat{x}_s, \hat{y}_s, \hat{z}_s)$, and hence, $\hat{x}_{s_t} \notin (X_{OD} \cup X_Q)$, or (b) is linear with no break point which implies $\hat{x}_{s_t} \notin X_Q$. Let $g(x) = f(\hat{x}_{s_1} + x, \hat{x}_{s_2} + x, \ldots, \hat{x}_{s_p} + x, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ where $x \in \mathsf{R}$. At $x = 0$, $g(x)$ is linear and since $g(0)$ is equal to the optimal solution value, it is also constant. From Observation 4, we know that $g(x)$ is a piecewise linear function of x and its break points occur whenever $\hat{x}_s + x \in X_D$ for any $s \in \mathcal{S}$. Let point $(\mathbf{x}, \hat{\mathbf{y}}, \hat{\mathbf{z}}) = (x_{s_1}, \ldots, x_{s_p}, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ be a break point closest to $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$, such that $x_s = \hat{x}_s + x$ for all $s \in \mathcal{S}$. Note that $f(\mathbf{x}, \hat{\mathbf{y}}, \hat{\mathbf{z}}) = f(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$. We claim that there exists $t_1 \in P$ such that $x_{s_{t_1}} = \hat{x}_{s_{t_1}} + x \in X_{ID}$ because of the following reasons: *Case I.* If $x_{s_{t_1}} \notin X_{ID}$ and $x_{s_{t_1}} \in X_{OD}$, then either $(\mathbf{x}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ is a locally convex break point

which contradicts its optimality, or $g(x)$ is linear at x with no break point. *Case II.* If $x_{s_{t_1}} \notin X_{ID}$ and $x_{s_{t_1}} \in X_Q$, then again $(\mathbf{x}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ is a locally convex break point which contradicts its optimality. The same arguments can be applied to get an optimal solution $(\mathbf{x}, \mathbf{y}, \hat{\mathbf{z}})$ such that there exists $q_1 \in P$ for which $y_{s_{q_1}} \in Y_{ID}$.

Assume that there exists an optimal solution $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}) = (\hat{x}_{s_1}, \ldots, \hat{x}_{s_p}, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ such that there exists $t_1 \in P$ for which $\hat{x}_{s_{t_1}} \in X_{ID}$, but there does not exist a non-repetitive sequence $s_{t_1}, s_{t_2}, \ldots, s_{t_p}$ of the SZs such that for $k = 2, \ldots, p$, $\hat{x}_{s_{t_k}} \in X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_k)$ where $t_k \in P$ and $T_k = \{t_1, \ldots, t_{k-1}\}$. This implies that for all $t \in P \setminus \{t_1\}$, $\hat{x}_{s_t}$ is neither an x IDCV nor an x OSCV. In case $\hat{x}_{s_t} \in X_{OD}^{\hat{z}_{s_t}} \cup \mathcal{X}_{IS}^{\hat{z}_{s_t}}(P \setminus \{t\})$ for any $t \in P \setminus \{t_1\}$, then based on Observation 3, $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ at $x_{s_t} = \hat{x}_{s_t}$ either has a locally convex break point (which contradicts its optimality) or is linear with no break point. Likewise, in case $\hat{x}_{s_t} \in X_Q$ for any $t \in P \setminus \{t_1\}$, $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ at $x_{s_t} = \hat{x}_{s_t}$ has a locally convex break point, and thereby contradicts its optimality. Hence, for each $t \in P \setminus \{t_1\}$, $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ is linear at $x_{s_t} = \hat{x}_{s_t}$, and it is also constant at $x_{s_t} = \hat{x}_{s_t}$ because $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ is an optimal solution. For $t \in P \setminus T_2$ where $T_2 := \{t_1\}$, let $x_{s_t} \in X_D \cup \mathcal{X}_S^{\hat{z}_{s_t}}(T_2)$ be a break point of $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ that is closest to $\hat{x}_{s_t}$ and $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t}) = f(\hat{x}_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$. Now, pick $t_2 \in P \setminus T_2$ such that $|\hat{x}_{s_{t_2}} - x_{s_{t_2}}|$ is minimum. Since $(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, \{\hat{x}_{s_t}\}_{t2PnT_3}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ is also an optimal solution and a break point, $x_{s_{t_2}} \notin X_{OD} \cup \mathcal{X}_{IS}(\Xi, T_2) \cup X_Q$ because of the same reasons previously explained. Hence, $x_{s_{t_2}} \in X_{ID} \cup \mathcal{X}_{OS}^{\hat{z}_{s_{t_2}}}(T_2) \subseteq X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_2)$.

Next, we use induction to show that another optimal solution $(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, x_{s_{t_3}}, \ldots, x_{s_{t_p}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ can be found such that $\hat{x}_{s_{t_1}} \in X_{ID}$ and for $k = 2, \ldots, p$, $x_{s_{t_k}} \in X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_k)$ where $t_k \in P$ and $T_k = \{t_1, \ldots, t_{k-1}\}$. To do so, assume that there exists $b \in \{2, \ldots, p\}$ and an optimal solution $\left(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, \ldots, x_{s_{t_b}}, \{\hat{x}_{s_t}\}_{t2PnT_{b+1}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\right)$ such that $x_{s_{t_k}} \in X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_k)$ for $k = 2, \ldots, b$, and $\hat{x}_{s_t} \notin X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_k)$ for $t \in P \setminus T_{b+1}$. Note that for $b = 2$, we have already proved the existence of such optimal solution and in case $b = p$, then $P \setminus T_{b+1} = \emptyset$ and hence we have an optimal solution that satisfies condition (i). Now for $b \in \{2, \ldots, p-1\}$, we have to find another optimal solution $\left(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, \ldots, x_{s_{t_b}}, x_{s_{t_{b+1}}}, \{\hat{x}_{s_t}\}_{t2PnT_{b+2}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\right)$ such that $x_{s_{t_{b+1}}} \in X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_{b+1})$ where $t_{b+1} \in P \setminus T_{b+1}$. Recall that for each $t \in P \setminus \{t_1\}$, $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ is linear at $x_{s_t} = \hat{x}_{s_t}$, and it is also constant at $x_{s_t} = \hat{x}_{s_t}$. Since $P \setminus T_{b+1} \subseteq P \setminus \{t_1\}$, for $t \in P \setminus T_{b+1}$, $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ is linear with no break point at $\left(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, \ldots, x_{s_{t_b}}, \{\hat{x}_{s_t}\}_{t2PnT_{b+1}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\right)$. Let $x_{s_t} \in X_D \cup \mathcal{X}_S^{\hat{z}_{s_t}}(T_{b+1})$, $t \in P \setminus T_{b+1}$, be a break point of $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$ that is closest to $\hat{x}_{s_t}$ and $f(x_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t}) = f(\hat{x}_{s_t}, \hat{y}_{s_t}, \hat{z}_{s_t})$. Select index $t_{b+1} \in P \setminus T_{b+1}$ such that $\left|\hat{x}_{s_{t_{b+1}}} - x_{s_{t_{b+1}}}\right|$ is minimum to get another optimal solution $\left(\hat{x}_{s_{t_1}}, x_{s_{t_2}}, \ldots, x_{s_{t_b}}, x_{s_{t_{b+1}}} \{\hat{x}_{s_t}\}_{t2PnT_{b+2}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\right)$ where $x_{s_{t_{b+1}}} \in X_{ID} \cup \mathcal{X}_{OS}^{\hat{z}_{s_{t_{b+1}}}}(T_{b+1}) \subseteq$

$X_{ID} \cup \mathcal{X}_{OS}(\Xi, T_{b+1})$. Note that $x_{s_{t_{b+1}}} \notin X_{OD} \cup \mathcal{X}_{IS}(\Xi, T_{b+1}) \cup X_Q$ because otherwise $f(x_{s_{t_{b+1}}}, \hat{y}_{s_{t_{b+1}}}, \hat{z}_{s_{t_{b+1}}})$ either has a locally convex break point or is linear with no break point at $x_{s_{t_{b+1}}} = x_{s_{t_{b+1}}}$. This implies that by sequentially setting $b = 2, \ldots, p-1$, we can eventually get an optimal solution that satisfies condition (i). By applying the same arguments for y coordinates, we reach an optimal solution that satisfies condition (ii). □

Remark 2. *For $m = 1$, Theorem 2 in this paper strengthens Theorem 3 of Bansal and Kianfar (2017). According to the latter, there exists an optimal solution for PMCLP-PCR-QoS with $\Xi = \{1\}$ (PMCLP-PCR), denoted by $(x_{s_1}, \ldots, x_{s_p}, y_{s_1}, \ldots, y_{s_p})$, such that x_s is an x IDCV or x SCV for all $j \in P$. Whereas, according to Theorem 2, there exists an optimal solution for PMCLP-PCR such that x_s is an x IDCV or x OSCV for all $j \in P$, thereby reducing the solution search space. In Section 6, we will observe how this improves the exact algorithm of Bansal and Kianfar (2017) for PMCLP-PCR and reduces the time taken to solve it.*

4.2 Exact Branch-and-Bound Algorithm for PMCLP-PCR-QoS

In this section, we present a generalization of the branch-and-bound (B&B) algorithm presented by Bansal and Kianfar (2017) for PMCLP-PCR-QoS with $\Xi = \{1\}$ to solve PMCLP-PCR-QoS with $\Xi = \{\xi_1, \xi_2, \ldots, \xi_m\}$ where $m \geq 1$, thereby allowing adjustable QoS for each SZ. The former can be utilized to solve PMCLP-PCR-QoS with an additional restriction of $z_{s_j} = z \in \Xi$ (same QoS for all SZs) by solving PMCLP-PCR m number of times. For $m = 1$, our new algorithm implicitly enumerates a reduced solution search space, in comparison to the solution space considered in Bansal and Kianfar (2017), and as a result, our proposed algorithm is computationally faster (see Section 6 for more details). For $m \geq 2$, though the outline of our approach is similar to a generic B&B approach, its key components such as branching, upper bound computation, lower bound updates, and node selection, as well as the data structures to efficiently implement them are customized for PMCLP-PCR-QoS. It is important to note that PMCLP-PCR-QoS can also be solved by brute-force search, i.e., explicit enumeration of all possible combinations of x and y critical values, in particular IDCVs and OSCVs. However, this approach is computationally very expensive. For an example, to solve an instance of PMCLP-PCR-QoS with $p = m = 2$ and $n = 100$ using explicit enumeration, we have to evaluate reward function for at least 2.56 million solutions. In comparison, our approach explores only 13,092 nodes (average for 10 instances) of B&B tree where for most of these nodes, it computes upper bound that is computationally less expensive than evaluating reward function.

Since some attributes of our B&B approach for PMCLP-PCR-QoS are inherited from B&B approach for PMCLP-PCR, we use similar notations and nomenclatures, to the extent possible, in the ensuing subsections for the sake of reader's convenience. Before we present the algorithm, we highlight its main additional features. In PMCLP-PCR-QoS, there is an additional set of decisions of choosing QoS (i.e., scaling factor) for each SZ. Therefore, we need different strategies and data structures to incorporate these decisions in building B&B tree, designing branching routine, and computing upper bound at each node (see Sections 4.2.2 and 4.2.3, respectively, for more details). The branching strategies also influence the generation of OSCVs that depends on scaling factor (dimensions) of SZs. Moreover, to compute lower bound at a leaf node (covered reward by p SZs having different QoS), we ensure that for a subregion (partially) covered by multiple SZs with different QoS, the reward rate corresponding to the best QoS among these SZs is considered in the reward calculations. In B&B approach for PMCLP-PCR, the authors construct two auxiliary trees (one for each of x and y axes) to store partitions of initial set of IDCVs. This construction provides computational advantages and avoid repetition of computing partitions of the set of IDCVs for different SZs. However, in order to extend this procedure for PMCLP-PCR-QoS, we would need $2m$ number of such auxiliary trees (two for each scaling factor). Therefore, instead of using the auxiliary trees, we evaluate four different procedures to design the B&B tree, and consider the one that helps us to focus on the regions with high reward, provides a strong lower bound quickly, and makes the overall algorithm computationally efficient (refer to Section 4.2.2 for more details).

4.2.1 Main Body

Algorithm 3 provides a pseudocode for the proposed customized B&B algorithm. We initialize the algorithm (Line 1) by computing an initial feasible solution and lower bound of the objective function of PMCLP-PCR-QoS using **GreedyAlgorithm** for the set of DZs $\mathcal{D}$ and p SZs (discussed in Section 3). Recall that according to Theorem 1, this lower bound is at least 63.2% of the optimal solution value for PMCLP-PCR-QoS instance. Thereafter, we construct the root node Q_0 of the B&B tree in line 2 that stores p sets of all x IDCVs, $X^{s_j}_{Q_0} = X_{ID} = \cup_{z2}\ X^z_{ID}$, $j \in P$, and y IDCVs, $Y^{s_j}_{Q_0} = Y_{ID} = \cup_{z2}\ Y^z_{ID}$ for all $j \in P$. Similarly, at each node Q of the B&B tree, we store p subsets of x CVs (IDCVs and OSCVs), denoted by $X^{s_j}_Q, j \in P$, and p subsets of y CVs (IDCVs and OSCVs), denoted by $Y^{s_j}_Q, j \in P$, along with three indicators: (i) a $p \times 1$ vector $\mathcal{Z}(Q)$ to store scaling factor associated with each SZ, (ii) $\mathcal{BA}(Q)$ to store branching axis, i.e x or y, and (iii) $\mathcal{BS}(Q) \in \mathcal{S}$ to store branching SZ and $index(\mathcal{BS}(Q)) \in P$ denotes the index of the SZ. At root node Q_0, we set $\mathcal{Z}(Q_0) \leftarrow \mathbf{0}$ because scaling factor is not yet selected

for each SZ, $\mathcal{BA}(Q_0) \leftarrow$ "X", $\mathcal{BS}(Q_0) \leftarrow s_1$, and $index(\mathcal{BS}(Q_0)) \leftarrow 1$. Let LCN denotes a list of candidate nodes that is initialized by including Q_0 in it. Each node in this list represents a restricted PMCLP-PCR-QoS problem (or subproblem) where SZ s_j, $j \in P$, has $(x_{s_j}, y_{s_j}) \in X_Q^{s_j} \times Y_Q^{s_j}$ and z_{s_j} is the j^{th} element of $\mathcal{Z}(Q)$. A zero element of $\mathcal{Z}(Q)$ implies that the scaling factor of the corresponding SZ has not been selected yet and it belongs to Ξ.

Algorithm 3 Exact Branch-and-Bound Algorithm: Main Body

1: Using **GreedyAlgorithm**(D; ; p), obtain an initial feasible solution and lower bound (LB);
2: Construct a root node Q_0 using x and y IDCVs with $Z(Q_0)$ **0**, $BA(Q_0)$ X, and $BS(Q_0)$ 1;
3: Initialize a list of candidate nodes LCN fQ_0g;
4: Select a node Q belonging to LCN using depth rst strategy;
5: Calculate an upper bound for node Q:
$UB(Q)$ **UpperBound**(Q; D; ; p);
6: **if** $UB(Q)$ LB **then** prune the node Q and remove it from LCN
7: **else if** Node Q is a leaf node **then** LB $UB(Q)$ and remove Q from LCN
8: **else** Create child nodes of node Q using **Branching**(Q) routine and add them to LCN;
9: Go to Step 4 if LCN is nonempty;
10: **return** LB and Optimal solution (x_{s_1}; : : : ; x_{s_p}; y_{s_1}; : : : ; y_{s_p}; z_{s_1}; : : : ; z_{s_p});

To implicitly enumerate the solution space, we select a node Q belonging to LCN and calculate an upper bound (UB) of the optimal objective value for the restricted problem associated to node Q, using **UpperBound** function. Refer to Section 4.2.3 for pseudocode and more details regarding this function. To avoid explicit enumeration of the critical points, a strong upper bound is needed so that nodes (or subregions) that cannot provide a feasible solution better than the best known solution, be pruned. In line 6, we prune a node if $UB \leq LB$ (best known lower bound) and remove it from LCN. Otherwise, if node Q is a leaf node, i.e., $|X_Q^{s_j}| = |Y_Q^{s_j}| = 1$ for all $j \in P$, and $UB > LB$, the **UpperBound** function returns an improved lower bound for the original problem (Line 7), and the leaf node is pruned and removed from LCN. However, in case node Q is not a leaf node and $UB > LB$, we create child nodes of node Q (also referred to as parent node) using **Branching** function and add them to LCN (Line 8). A child node is a subproblem that inherits (some) restrictions from its parent node along with new restrictions; refer to Section 4.2.2 for pseudocode and more details of the **Branching** function. We use depth-first strategy to traverse the B&B tree. This implies that in case the current node is pruned, we select a node that is the right sibling of the node or one of its ancestor, and in case the current node is branched, we select its first (or left most child) from LCN and repeat Lines 4-9. The algorithm is terminated when LCN is empty and it returns an optimal solution, i.e., location and scaling factor of each SZ $(x_{s_1}, \ldots, x_{s_p}, y_{s_1}, \ldots, y_{s_p}, z_{s_1}, \ldots, z_{s_p})$ as well as the optimal captured reward value (i.e., best known LB).

4.2.2 Branching

This function creates a list of child nodes for a given parent node Q (refer to Algorithm 4 for psuedocode). These child nodes are added in LCN and the most left child node of Q is selected from LCN in the next iteration. At any given node Q, indicators $\mathcal{BA}(Q)$ and $\mathcal{BS}(Q)$ determine branching axis and branching SZ. These two properties are defined to select a set among the sets $X_Q^{s_j}$ and $Y_Q^{s_j}$ for $j \in P$ that is to be partitioned to create new child nodes. We use the function *NewChildNode(Q)* to create child node T for node Q, which inherits the properties of its parent. We classify the branching procedure into three categories:

(a) *QoS Branching.* When $X_Q^{\mathcal{BS}(Q)} = X_{ID}$ (or $Y_Q^{\mathcal{BS}(Q)} = Y_{ID}$), i.e., the scaling factor for SZ $\mathcal{BS}(Q)$ has not been selected, we create $|\Xi| = m$ child nodes for a parent node Q. Each child node corresponds to a fixed scaling factor $z \in \Xi = \{\xi_1, \ldots, \xi_m\}$ for the branching SZ. As a result, a child node T of node Q corresponding to scaling factor $z \in \Xi$ has $\mathcal{Z}(T, \mathcal{BS}(T)) = z$ and subsets $X_T^{\mathcal{BS}(T)} = X_{ID}^{Z(T,\mathcal{BS}(T))}$ and $Y_T^{\mathcal{BS}(T)} = Y_{ID}^{Z(T,\mathcal{BS}(T))}$ (Lines 4-8 of Algorithm 4). Note that $\mathcal{Z}(T, s_j)$ denotes the j^{th} element of vector $\mathcal{Z}(T)$.

(b) *Partition-Based Branching.* In case $\mathcal{BA}(Q) =$ "X" (or "Y"), $X_Q^{\mathcal{BS}(Q)} \neq X_{ID}$ (or $Y_Q^{\mathcal{BS}(Q)} \neq Y_{ID}$), and $|X_Q^{\mathcal{BS}(Q)}| \neq 1$ (or $|Y_Q^{\mathcal{BS}(Q)}| \neq 1$), node Q is branched to child nodes by partitioning the set $X_Q^{\mathcal{BS}(Q)}$ (or $Y_Q^{\mathcal{BS}(Q)}$). Each child node T of parent node Q corresponds to a partition where the partition is assigned to $X_T^{\mathcal{BS}(T)}$ (or $Y_T^{\mathcal{BS}(T)}$) of node T (lines 10-14). In addition, whenever $X_Q^{\mathcal{BS}(Q)} = X_{ID}^{Z(Q,\mathcal{BS}(Q))}$ (or $Y_Q^{\mathcal{BS}(Q)} = Y_{ID}^{Z(Q,\mathcal{BS}(Q))}$) and at least one SZ, other than $\mathcal{BS}(Q)$, has fixed x (or y) coordinates, we also create child node T corresponding to each x (or y) OSCVs generated by SZs with fixed x (or y) positions, where SZ $\mathcal{BS}(Q)$ can be placed (lines 15-20). Each child node T defines a further restricted problem (or subproblem) in which SZ s_j, $j \in P$, can only be positioned at the locations $(x, y) \in X_T^{s_j} \times Y_T^{s_j}$. At node T, we set the indicators $\mathcal{BA}$, $\mathcal{BS}$, and $\mathcal{Z}$ same as the parent node Q, except when $X_T^{\mathcal{BS}(Q)}$ (or $Y_T^{\mathcal{BS}(Q)}$) becomes singleton set, i.e., SZ $\mathcal{BS}(Q)$ has a fixed x (or y) coordinate. For the foregoing scenario, we update $\mathcal{BS}(T)$ and/or $\mathcal{BA}(T)$ so that while branching on node T, if needed, an appropriate set among the sets $X_T^{s_j}$ and $Y_T^{s_j}$ for $j \in P$ is selected for partitioning. It is important to note that the efficiency and effectiveness of the B&B algorithm to solve PMCLP-PCR-QoS depends on effective generation of partitions and updates of indicators $\mathcal{BS}(T)$ and/or $\mathcal{BA}(T)$.

For partitioning, we assign a priority score to each element of the set X_{ID}^z, $z \in \Xi$, i.e., $x_k \in X_{ID}^z$ for $k \in \{1, \ldots, |X_{ID}^z|\}$ has priority score $p(x_k) = \sum_{d 2 \mathcal{D}_k} v_d^z$ where $\mathcal{D}_k := \{d \in \mathcal{D} : x_d \leq x_k < x_d + w_d\}$, i.e., sum of reward rate corresponding to scaling factor z for DZs such that x_k lies between x coordinate of vertical edges of DZ d. We also

Algorithm 4 Branching

```
1: function Branching(Q)
2:     if BA(Q) = "X" then
3:         if index(BS(Q))    p (node Q has at least a SZ whose x coordinate is not   xed) then
4:             if X_Q^{BS(Q)} = X_ID then
5:                 for z     1 to   m do
6:                     T     NewChildNode(Q)
7:                     X_T^{BS(T)}     X_ID^z; Y_T^{BS(T)}     Y_ID^z
8:                     Z(T; BS(T))     z
9:             else
10:                for n_x     1 to NumPartitions(X_Q^{BS(Q)}) do
11:                    T     NewChildNode(Q)
12:                    X_T^{BS(T)}     GetPartition(X_Q^{BS(Q)}; n_x)
13:                    if jX_T^{BS(T)}j = 1 then
14:                        index(BS(T))     index(BS(Q)) + 1
15:                if X_Q^{BS(Q)} = X_ID^{Z(Q,BS(Q))} then
16:                    L_x     fj 2 P : jX_Q^{s_j}j = 1g
17:                    for x* 2 (⋃_{j2L_x} X_OS^{s_j,Z(Q,BS(Q))}) n X_ID do
18:                        T     NewChildNode(Q)
19:                        X_T^{BS(T)}     x*
20:                        index(BS(T))     index(BS(Q)) + 1
21:        else
22:            l_1     smallest SZ index such that x position of the SZ is   xed at an x IDCV
                         and y position of the SZ is not   xed;
23:            L     set of SZ indices whose x position is   xed at x OSCV and y position is not   xed;
24:            for l 2 L [ fl_1g do
25:                T     NewChildNode(Q)
26:                index(BS(T))     l
27:                BA(T)     "Y"
28:    else if BA(Q) = "Y" then
29:        if jY_Q^{BS(Q)}j = 1 then do Steps 22 to 27
30:        else
31:            for n_y     1 to NumPartitions(Y_Q^{BS(Q)}) do
32:                T     NewChildNode(Q)
33:                Y_T^{BS(T)}     GetPartition(Y_Q^{BS(Q)}; n_y)
34:            if Y_Q^{BS(Q)} = Y_ID^{Z(Q,BS(Q))}) then
35:                L_y     fj 2 P : jY_Q^{s_j}j = 1g
36:                for y* 2 (⋃_{j2L_y} Y_OS^{s_j,Z(Q,BS(Q))}) n Y_ID do
37:                    T     NewChildNode(Q)
38:                    Y_T^{BS(T)}     y*
```

considered $p(x_k) = \frac{\sum_{d2D_k} v_d^z}{jD_kj}$ (average of reward rates), but observed that this choice of priority score is not effective. Then, we sort all elements of the set X_{ID}^z based on their priority score in the descending order. The sorted set is also denoted by $\{x_1, x_2, \ldots, x_{jX_{ID}^z j}\}$ for the sake of convenience. Similarly, the priority score $p(y_k)$ is defined for each $y_k \in Y_{ID}^z$ and elements of set Y_{ID}^z are also sorted in descending order of their priority score. At node Q, we split $X_Q^{BS(Q)}$ (or $Y_Q^{BS(Q)}$) by creating disjoint subsets when $x_{i+1} - x_i > \beta \max_{j=1,\ldots,jX_Q^{BS(Q)}j \ 1}\{x_{j+1} - x_j\}$ for any $x_i, i \in \{1, \ldots, |X_Q^{BS(Q)}| - 1\}$ where $\beta \in (0, 1)$. These partitions are assigned to child nodes using *GetPartition* in Lines 12 and 33. The smaller values of β increase the number of partitions (*NumPartitions* in Lines 10 and

31), and the bigger values of β lead to either no partitioning or very small number of subsets, thereby reducing the computational efficiency of the B&B algorithm. Based on our computational experiments, we select $\beta = 0.5$ as the most appropriate value resulting in an efficient algorithm. This ensures that the B&B algorithm traverses regions with high concentration of rewards faster, and hence, it quickly finds good lower bounds.

In our computational studies, we also considered the strategy of sorting child nodes based on their priority score in each branching step (similar to Bansal and Kianfar (2017)). Specifically, instead of first sorting the set of IDCVs based on priority scores and then creating partitions (as discussed above), the set $X_Q^{BS(Q)}$ (or $Y_Q^{BS(Q)}$) sorted in ascending order of IDCVs is partitioned. Then, a priority score is assigned to each child node T of node Q, defined by $\sum_{x_k 2 X_T^{BS(T)}} p(x_k)/|X_T^{BS(T)}|$, and the child nodes are sorted based on their priority score. A child node with highest priority score is placed as the left-most child node of Q in the B&B tree. Though this approach is computationally comparable to our approach in terms of solution time of the B&B algorithm, it resulted in more number of traversed nodes. We also defined priority score for child node T as $\sum_{x_k 2 X_T^{BS(T)}} p(x_k)$, but this increased the solution time.

We update indicator $\mathcal{BS}(T)$ by incrementing $index(\mathcal{BS}(T))$ by one (Line 14 and 20), when $\mathcal{BA}(Q) =$ "X", $X_T^{BS(Q)}$ becomes a singleton set, and $Y_Q^j = Y_{ID}$ for all $j \in P$. As a result, the sequence $s_{t_1}, s_{t_2}, \ldots, s_{t_p}$ in Theorem 2 is equal to $1, \ldots, p$. This is justified because each SZ has the same set of scaling factors $\Xi = \{\xi_1, \ldots, \xi_m\}$ as well as the same set of initial x IDCVs associated with it (i.e., all SZs are homogeneous at the beginning). Therefore, different sequences $s_{t_1}, s_{t_2}, \ldots, s_{t_p}$ will lead to the same set of solutions. In contrast, when all SZs have fixed x coordinate, they are no longer homogeneous. So, we cannot consider $q_1, q_2, \ldots, q_p$ in Theorem 2 to be equal to $1, \ldots, p$, and hence a permutation branching is required. Note that we can also set $\mathcal{BA}(Q_0) =$ "Y" at the root node Q_0. In that case, the sequence $q_1, q_2, \ldots, q_p$ will be equal to $1, \ldots, p$ and permutation branching will be required for branching along x axis.

(c) *Permutation Branching* (Lines 22-27). For a node Q where all p SZs have fixed x coordinates, i.e., $\mathcal{BA}(Q) =$ "X" and $index(\mathcal{BS}(Q)) > p$, and fixed QoS, i.e., $\mathcal{Z}(Q, s_j) \neq 0$ for all $j \in P$, we start the branching along the y axis using Y_Q^j, $j \in P$. At this node, SZs are not necessarily homogeneous because they can have different x coordinates and QoS. Therefore, according to Theorem 2, we consider different sequences $q_1, q_2, \ldots, q_p$ of branching SZs along y axis. However, we observe that because of some symmetries in the B&B tree, we can ignore a few sequences. For an example, when all SZs are fixed on x IDCVs (say $\hat{x}_{s_1}, \hat{x}_{s_2}, \ldots, \hat{x}_{s_p}$) with QoS (say $\hat{z}_{s_1}, \hat{z}_{s_2}, \ldots, \hat{z}_{s_p}$), we only consider sequence

$1, \ldots, p$ for unique solutions. This is because for each sequence $q_1, q_2, \ldots, q_p$, there exists another node in the B&B tree where all SZs are fixed on x IDCVs $(\hat{x}_{s_{q_1}}, \hat{x}_{s_{q_2}}, \ldots, \hat{x}_{s_{q_p}})$ with QoS $\hat{z}_{s_{q_1}}, \hat{z}_{s_{q_2}}, \ldots, \hat{z}_{s_{q_p}}$ as the initial set of IDCVs X_{ID} and scaling factors Ξ are same for all SZs. In contrast, when a SZ has an x coordinate fixed at an x OSCV, the foregoing property does not hold. Let $\bar{L}$ be the set of SZ indices whose x coordinate is fixed at x OSCVs (line 23). Then, we consider all permutations of $s_1, s_2, \ldots, s_p$ where SZs belonging to the set $\{s_j : j \in P \setminus \bar{L}\}$ are considered as identical objects. This is done in Algorithm 4 as follows. Let l_1 be the smallest SZ index having x positioned at x IDCVs and y is not fixed (line 22). Now, for each $l \in l_1 \cup \bar{L}$, we create a child node T of node Q where $index(\mathcal{BS}(T)) = l$ and $\mathcal{BA}(T) =$ "Y" (line 24-27).

4.2.3 Upper Bound

This function calculates an upper bound of the reward for a given node (see Algorithm 5). Any node Q in the tree is considered as a leaf node in case $|X_Q^{s_j}| = |Y_Q^{s_j}| = 1$ for all $j \in P$. Each leaf node provides a feasible solution $(\hat{x}_{s_1}, \ldots, \hat{x}_{s_p}, \hat{y}_{s_1}, \ldots, \hat{y}_{s_p}, \hat{z}_{s_1}, \ldots, \hat{z}_{s_p})$ for the problem. In case node Q is not a leaf node (i.e., there exist at least one $j \in P$ such that $|X_Q^j| > 1$ or $|Y_Q^j| > 1$), lines 5-16 compute the upper bound of the objective value for the subproblem associated with node Q. We assume that for each $z \in \Xi = \{\xi_1, \ldots, \xi_m\}$, $X_{ID}^z := \{x_1^z, \ldots, x_{jX_{ID}^z j}^z\}$ and $Y_{ID}^z := \{y_1^z, \ldots, y_{jX_{ID}^z j}^z\}$ are sorted in the ascending order (Line 3). Let M_z, $z \in \Xi$, be a matrix whose rows and columns are indexed by $x \in \{x_1^z, \ldots, x_{jX_{ID}^z j}^z\}$ and $y \in \{y_1^z, \ldots, y_{jY_{ID}^z j}^z\}$. Each entry of this matrix is denoted by $M_z(x, y)$ and it returns the total captured reward for DZs $\mathcal{D}$ by a SZ with fixed $(x_s, y_s, z_s) = (x, y, z)$. Matrices M_z for all $z \in \Xi$ are created only once while computing initial lower bound using greedy approach and then utilized in each call of **UpperBound** function. In particular, we store the objective value by solving the PMCLP-PCR with $p = 1$ for chosen $z \in \Xi$ using improved PVT algorithm of Bansal and Kianfar (2017).

To compute an upper bound at node Q, we first find the maximum values in p submatrices of M_z, $z \in \Xi$, whose rows and columns correspond to $X_Q^{s_j}$ and $Y_Q^{s_j}$, respectively, for $j \in P$ (lines 6-15), and then we aggregate these values (lines 8 and 16). We use matrix M_z for a SZ whose scaling factor z is known at node Q, and otherwise, we use $\max_{z2} M_z$ for computing the upper bound. Observe that when $X_Q^{s_j}$ contains a single x OSCVs (say $\hat{x}$) such that $z_{s_j} = z \in \Xi$ is known and $\hat{x} \notin X_{ID}^z$, we consider a submatrix of M_z with rows and columns corresponding to set $\bar{X} \subset X_{ID}^z$ and set $\bar{Y} \subset Y_{ID}^z$, respectively. We define the sets $\bar{X}$ and $\bar{Y}$ as follows. In case $x_1^z < \hat{x} < x_{jX_{ID}^z j}^z$, set $\bar{X}$ is defined by $\{\underline{x}, \overline{x}\}$, where $\underline{x}$ and $\overline{x}$ are closest x IDCVs in X_{ID}^z to $\hat{x}$ which are greater and smaller than $\hat{x}$, respectively (line 11). Otherwise, set $\bar{X}$ includes either x_1^z (if $\hat{x} < x_1^z$) or $x_{jX_{ID}^z j}^z$(if $\hat{x} > x_{jX_{ID}^z j}^z$) in lines

12-13. Likewise, set $\bar{Y} \subset Y_{ID}^{z}$ is also defined by the same procedure (line 15).

Algorithm 5 Upper Bound Calculation

```
1: function UpperBound(Q; D;  ; p)
2:     UB    0;
3:     fx_1^z; : : : ; x_{j X_ID^z j}^z g    list of elements in X_ID^z sorted in ascending order for any z 2
4:     if Node Q is a leaf node then UB    CoveredReward(Q; D; S);
5:     else
6:         for j 2 P do
7:             ẑ    Z(Q; s_j);
8:             if ẑ = 0 then UB    UB + max_{z2}  {M_z(x; y) : (x; y) 2 X_ID^z    Y_ID^z}
9:             else
10:                if X_Q^{s_j} contains an x OSCV (say x̂) and x̂ 2 X_ID^ẑ then
11:                    if x_1^ẑ < x̂ < x_{j X_ID^ẑ j}^ẑ then X    f x̲; x̄g    . x̄ and x̲ are the closest components in X_ID^ẑ to x̂
                                                                                that are larger and smaller than x̂, respectively;
12:                    else if x̂    x_1^ẑ then X    f x_1^ẑ g
13:                    else X    f x_{j X_ID^ẑ j}^ẑ g
14:                else X    X_Q^{s_j}
15:                Repeat steps 10 to 14 for Y_Q^{s_j} to obtain Y    Y_ID
16:                UB    UB + max f M_ẑ(x; y) : (x; y) 2 X    Y g;
17:    return UB
```

For a leaf node, **UpperBound** calls the **CoveredReward** function (Algorithm 6) to calculate the total reward that has been captured by p SZs with known coordinates and scaling factor. In order to take care of the maximum coverage when multiple SZs with different QoS are overlapping, we need to position SZs with smaller scaling factor (higher QoS) first. Therefore, we sort SZs $\{s_1, \ldots, s_p\}$ based on their scaling factor in the ascending order. Starting with a SZ having smallest scaling factor, the captured reward is calculated by $\sum_{d2D} v_d^{z_{s_j}} A(s_j \cap d)$ for $j \in P$ and then, by using **TrimOut** function, we remove the covered parts from the DZs and store a new set of DZs from remaining parts. This procedure is repeated for each fixed SZ and the total calculated covered reward value is aggregated.

Algorithm 6 Covered Reward Function

```
1: function CoveredReward(Q; D; p)
2:     C    0, D    D;
3:     fs_{i_1}; : : : ; s_{i_p} g list of p SZs such that
       Z(Q; s_{i_1})    Z(Q; s_{i_2})    : : : Z(Q; s_{i_p});
4:     for j    i_1 to i_p do
5:         (x_{s_j}; y_{s_j}; z_{s_j})    (X_Q^{s_j}; Y_Q^{s_j}; Z(Q; s_j));
6:         for d 2 D do C    C + v_d^{z_{s_j}} A(s_j \ d);
7:             D    TrimOut(D; x_{s_j}; y_{s_j}; z_{s_j});
8:     return C
```

5 Exact Algorithm for 1D-PMCLP-PC-QoS

In this section, we provide theoretical properties for the solution space of 1D-PMCLP-PC-QoS where $\Xi_s = \{\xi_s^1\}$ for all $s \in \mathcal{S}$, and an exact algorithm to solve it. Recall that in this problem, the DZs (wastewater or trash zones) and SZs (of treatment plants or trash booms) are line segments on x-axis (a line representing river). Since scaling factor z_s of each SZ s is fixed but different, i.e., ξ_s^1, and $y_d = y_s = 0$ for all $d \in \mathcal{D}$ and $s \in \mathcal{S}$, Problem (1) reduces to

$$\max_{\mathbf{x}} \left\{ f(\mathbf{x}) := \sum_{i=1}^{n} f_i(\mathbf{x}) = \mathcal{T}_i\Big(d_i \cap \Big(\cup_{j=1}^{p} s_j \Big)\Big) \right\}, \tag{10}$$

where $\mathcal{T}_i(.)$ returns the total reward from line segment captured by $s_1, \ldots, s_p$. In other words, $\mathbf{z} = \{\xi_{s_1}^1, \ldots, \xi_{s_p}^1\}$ and $\mathbf{y} = \mathbf{0}$ in (1). Assume that the base SZ s_0 for 1D-PMCLP-PC-QoS is a line segment with known width w_{s_0}. Note that the reward rates for covering each DZ d_i are reward per unit length. For $p = 1$, 1D-PMCLP-PC-QoS is a special case of the PMCLP-PCR and therefore, the properties for solution space and exact algorithms provided by Song et al. (2006) and Bansal and Kianfar (2017) for PMCLP-PCR are directly applicable to 1D-PMCLP-PC. However, the same is not true for $p \geq 2$, primarily because each SZ has different width.

5.1 Solution Space of 1D-PMCLP-PC-QoS

We establish relation between PMCLP-PCR-QoS with $\Xi_s = \{\xi_1, \ldots, \xi_m\}$ for all $s \in \mathcal{S}$ and 1D-PMCLP-PC-QoS with $\Xi_s = \{\xi_s^1\}$ for all $s \in \mathcal{S}$. Notice that in case $\xi_s^1 \in \{\xi_1, \ldots, \xi_m\}$ for all $s \in \mathcal{S}$, then the objective function of 1D-PMCLP-PC-QoS is conceptually same as the objective function of PMCLP-PCR-QoS with fixed $(\mathbf{y} = \mathbf{0}, \mathbf{z})$ for SZs in the 1D solution space. In other words, the objective function of 1D-PMCLP-PC-QoS is a piece-wise linear function with break points at x DCVs and SCVs. Since each SZ has different width in 1D-PMCLP-PC-QoS, we define DCVs associated to each SZ s_j, $j \in \{1, \ldots, p\}$, by $X_D^{z_{s_j}} = \cup_i X_D^{d_i, z_{s_j}}$ (see Definition 1) and denote the set of all IDCVs and ODCVs associated to SZ s_j by $X_{ID}^{z_{s_j}}$ and $X_{OD}^{z_{s_j}}$, respectively. Each SZ with a fixed position creates a set of three or four SCVs for other SZs. The set of SCVs generated by SZ s_j, positioned at x_{s_j}, for another SZ s_k $(k \neq j)$ is defined by $\mathcal{X}_S^{s_j, z_{s_k}}$ (see Definition 3). Likewise, we define $\mathcal{X}_S^{z_s}(\mathcal{J})$, $\mathcal{X}_{IS}^{z_s}(\mathcal{J})$, and $\mathcal{X}_{OS}^{z_s}(\mathcal{J})$ for $s \in \mathcal{S} \setminus \mathcal{J}$. Since the scaling factor of all SZs is fixed, there is no QCV, i.e., $X_Q^{d_i} = \emptyset$ for all i. Furthermore, Observations 3 and 4 are valid for the objective function of 1D-PMCLP-PC-QoS.

Theorem 3. *There exists an optimal solution* $\mathbf{x} = (x_{s_1}, \ldots, x_{s_p}) \in \mathsf{R}^p$ *of the 1D-PMCLP-*

PC-QoS and a non-repetitive sequence $s_{t_1}, s_{t_2}, \ldots, s_{t_p}$ *of the SZs such that* $x_{s_{t_1}} \in X_{ID}^{z_{s_{t_1}}}$ *and for* $k = 2, \ldots, p$, $x_{s_{t_k}} \in X_{ID}^{z} \cup \mathcal{X}_{OS}^{z}(T_k)$ *where* $z = \xi^{1}_{s_{t_k}}$, $T_k = \{t_1, \ldots, t_{k\ 1}\}$, *and* $t_k \in P$.

Proof. This theorem can be proved using the same arguments as used in the proof of Theorem 2 for condition (i), except that $z_s = \hat{z}_s = \xi^{1}_{s}$ for all $s \in S$ and $X_Q = \emptyset$ for all i.

5.2 Exact Algorithm for 1D-PMCLP-PC-QoS

The 1D-PMCLP-PC-QoS for $p = 1$ and $p \geq 2$ with $\Xi_s = \{1\}$ for all $s \in \mathcal{S}$, are special cases of the PMCLP-PCR, and therefore, the algorithms provided by Song et al. (2006) and Bansal and Kianfar (2017), respectively, are directly applicable for these special cases. However, when $p \geq 2$ and each SZ has different scaling factor, these approaches do not guarantee to provide an optimal solution for 1D-PMCLP-PC-QoS. Assuming that $\xi^{1}_{s} \in \Xi = \{\xi_1, \ldots, \xi_m\}$ for all $s \in \mathcal{S}$, our proposed exact algorithm for PMCLP-PCR-QoS with $\mathbf{y} = \mathbf{0}$, $\Xi := \{\xi^{1}_{s_1}, \ldots, \xi^{1}_{s_p}\}$, and $z_{s_j} = \xi^{1}_{s_j}$, $j \in P$, can be applied to find an optimal solution for 1D-PMCLP-PC-QoS. In this section, we provide another more computationally efficient B&B based exact algorithm for the 1D-PMCLP-PC-QoS. The main body and upper bound computation in this algorithm are similar to Algorithms 3 and 5 with some minor modifications (discussed below), but the branching routine involves features pertinent to this problem.

5.2.1 Main Body and Upper Bound

The outline of this approach is also given by Algorithm 3, where node Q stores p subsets of x CVs (IDCVs and OSCVs), $X_Q^{s_j}$, $j \in P$, and an indicator $\mathcal{BS}(Q)$. Each node represents a restricted 1D-PMCLP-PC-QoS problem where SZ $s_j \in \mathcal{S}$ has $x_{s_j} \in X_Q^{s_j}$. In this B&B tree, a leaf node Q has $|X_Q^{s_j}| = 1$ for all $j \in P$, and at root node Q_0, set $X_{Q_0}^{s} = X_{ID}^{\xi^{1}_{s}}$ for all $s \in \mathcal{S}$. In order to compute an upper bound (UB) on the objective function of the restricted subproblem at node Q, we utilize and modify Algorithm 5 as follows. For a leaf node, we call **CoveredReward** function (Algorithm 6) that provides a feasible solution $(\hat{x}_{s_1}, \ldots, \hat{x}_{s_p})$ for this problem as well. In case node Q is not a leaf node, an upper bound is computed using Lines 5-16 of Algorithm 5, where $Y_{ID}^{z} = \bar{Y} = \{0\}$ for all z, $\Xi = \{\xi^{1}_{s_1}, \ldots, \xi^{1}_{s_p}\}$, and $\mathcal{Z}(Q, s_j) = \xi^{1}_{s_j}$, $j \in P$. As a result, M_z, $z \in \Xi$, becomes a row vector whose rows are indexed by $x \in \{x^{z}_{1}, \ldots, x^{z}_{jX^{z}_{ID}j}\}$ and each entry of this vector is denoted by $M_z(x, 0)$ that returns the total captured reward for DZs $\mathcal{D}$ by a SZ with fixed $(x_s, y_s, z_s) = (x, 0, z)$.

5.2.2 Branching

This function generates a list of child nodes for a given parent node Q. These child nodes are added to the LCN and the most left child node of Q is selected from LCN in the next iteration. Since the scaling factor of each SZ comes from non-homogeneous sets of scaling factors ($\Xi_{s_t} \neq \Xi_{s_w}$ for $t \neq w$), then according to Theorem 3 we have to consider all sequences of SZs, $t_1, t_2, \ldots, t_p \in P$ in the branching. Observe that the symmetries in the B&B tree for PMCLP-PCR-QoS that resulted in the reduction of sequences of branching SZs along an axis, no longer exist for 1D-PMCLP-PC-QoS. We introduce a different strategy to truncate B&B tree for the latter, thereby leading to a new branching routine (Algorithm 7), denoted by **Branching1D**(Q). For root node $Q = Q_0$, we create p child nodes (Lines 2-6 of Algorithm 7) where each child node T has same properties as Q_0, except that indicator $\mathcal{BS}(T) \in \mathcal{S}$ is different for all child nodes. We refer to these nodes as first-level branching nodes, denoted by $\overline{Q}_0^j$, $j \in P$, where $index(\mathcal{BS}(\overline{Q}_0^j)) = j$. For each node Q in the subtree with $\overline{Q}_0^j$, $j \in P$, as root node, we define another indicator BSFL(Q) to store $index(\mathcal{BS}(\overline{Q}_0^j))$, i.e., j, and harness this information to reduce the number of nodes in the B&B tree.

Algorithm 7 Branching for 1D-PMCLP-PC-QoS

```
1: function Branching1D(Q)
2:     if Q is the root node Q_0 then
3:         for j 2 P = f1;:::;pg do
4:             T    NewChildNode(Q)
5:             BS(T)    s_j
6:             BSFL(T)    j
7:     else
8:         if jX_Q^{BS(Q)}j = 1 then
9:             L    fk 2 P : jX_Q^{s_k}j = 1g
10:            for l 2 P n L do
11:                for x 2 X_OS^{s_l}(L) do
12:                    T    NewChildNode(Q)
13:                    BS(T)    s_l
14:                    X_T^{BS(T)}    x
15:                if l > BSFL(Q) then
16:                    T    NewChildNode(Q)
17:                    BS(T)    s_l
18:                    X_T^{BS(T)}    X_ID^{ξ^1_{s_l}}
19:        else
20:            for n_x    1 to NumPartitions(X_Q^{BS(Q)}) do
21:                T    NewChildNode(Q)
22:                X_T^{BS(T)}    GetPartition(X_Q^{BS(Q)}; n_x)
```

In case $|X_Q^{BS(Q)}| \neq 1$ at a node Q, function **Branching1D**(Q) performs the partition-based branching in Lines 20-22 (also discussed in Section 4.2.2). However, whenever $|X_Q^{BS(Q)}| = 1$ at a node Q, i.e., SZ $\mathcal{BS}(Q)$ has fixed x coordinate, function **Branching1D**(Q) conducts a revised permutation branching (Lines 9-18) that works as follows. For 1D-PMCLP-PC-QoS with $p = 2$, according to Theorem 3, an optimal solution

(x_{s_1}, x_{s_2}) belongs to the set

$$\left\{(x_{s_1}, x_{s_2}) : x_{s_{t_1}} \in X_{ID}^{z_{s_{t_1}}} \text{ and } x_{s_{t_2}} \in X_{ID}^{z_{s_{t_2}}} \cup \mathcal{X}_{OS}^{z_{s_{t_2}}}(\{t_1\}) \text{ for all } (t_1, t_2) \in \{(1,2),(2,1)\}\right\}. \quad (11)$$

We observe that $\left\{\left(x_{s_{t_1}}, x_{s_{t_2}}\right) \in X_{ID}^{z_{s_{t_1}}} \times X_{ID}^{z_{s_{t_2}}}\right\}$ is same for each pair $(t_1, t_2) \in \{(1,2),(2,1)\}$ and hence, considering them once reduces the number of nodes in the B&B tree. Based on this observation, set (11) that can be rewritten as

$$\left\{(x_{s_1}, x_{s_2}) : X_{ID}^{z_{s_1}} \times \left(X_{ID}^{z_{s_2}} \cup \mathcal{X}_{OS}^{z_{s_2}}(\{1\})\right)\right\} \bigcup \left\{(x_{s_2}, x_{s_1}) : X_{ID}^{z_{s_2}} \times \left(X_{ID}^{z_{s_1}} \cup \mathcal{X}_{OS}^{z_{s_1}}(\{2\})\right)\right\}$$

is equivalent to

$$\left\{(x_{s_1}, x_{s_2}) : X_{ID}^{z_{s_1}} \times \left(X_{ID}^{z_{s_2}} \cup \mathcal{X}_{OS}^{z_{s_2}}(\{1\})\right)\right\} \bigcup \left\{(x_{s_2}, x_{s_1}) : X_{ID}^{z_{s_2}} \times \mathcal{X}_{OS}^{z_{s_1}}(\{2\})\right\}.$$

We incorporate this observation in revised permutation branching by considering: (a) $x_{s_1} \in X_{ID}^{z_{s_1}}$ and $x_{s_2} \in X_{ID}^{z_{s_2}} \cup \mathcal{X}_{OS}^{z_{s_2}}(\{1\})$ in subtree with $\overline{Q}_0^1$ as root node, and (b) $x_{s_2} \in X_{ID}^{z_{s_2}}$ and $x_{s_1} \in \mathcal{X}_{OS}^{z_{s_1}}(\{2\})$ in subtree with $\overline{Q}_0^2$ as root node. In general, for each subtree with $\overline{Q}_0^j$, $j \in P$, as root node we set $s_{t_1} = index(\mathcal{BS}(\overline{Q}_0^j)) = j$ and explore the solution space

$$\left\{\left(x_{s_{t_1}}, x_{s_{t_2}}, \ldots, x_{s_{t_p}}\right) : x_{s_{t_1}} \in X_{ID}^{z_{s_{t_1}}},\ x_{s_{t_k}} \in \mathcal{X}_{OS}^{z_{s_{t_k}}}(\{T_k\}) \text{ if } t_k < t_1,\ x_{s_{t_k}} \in X_{ID}^{z_{s_{t_k}}} \cup \mathcal{X}_{OS}^{z_{s_{t_k}}}(\{T_k\}) \text{ if } t_k > t_1\right\}$$

for all non-repetitive sequences of $t_2, \ldots, t_p \in P \setminus \{t_1\}$. Accordingly, we develop the B&B tree using the revised permutation branching (Lines 9-18). More specifically, whenever $|X_Q^{\mathsf{BS}(Q)}| = 1$ at a node Q, we define a set L of SZ indices whose x coordinate is fixed and for each $l \in P \setminus L$, we create $|\mathcal{X}_{OS}^{s_l}(L)|$ number of child nodes using the *NewChildNode* routine where each child node T has $\mathcal{BS}(T) = l$ and singleton $X_T^{\mathsf{BS}(T)}$ contains an x OSCV $\hat{x} \in \mathcal{X}_{OS}^{s_l}(L)$. In addition, for each $l \in P \setminus L$ such that $l > \mathsf{BSFL}(Q)$, we also create a child node T of node Q that has $\mathcal{BS}(T) = l$ and $X_T^{\mathsf{BS}(T)}$ contains x IDCVs corresponding to SZ s_l, i.e., $X_{ID}^{\mathcal{E}_{s_l}^1}$. As mentioned before, this technique reduces in the number of nodes of the B&B tree, thereby leading to a computationally efficient solution approach for 1D-PMCLP-PC-QoS.

6 Computational Experiments

For computational study, we perform three sets of experiments to evaluate the effectiveness and efficiency of the solution approaches presented in this paper. Specifically, in the first set of the experiments, we compare the performance of our proposed exact method for the PMCLP-PCR (i.e., PMCLP-PCR-QoS with fixed and same scaling factor $\Xi = \{1\}$), with the existing algorithm proposed by Bansal and Kianfar (2017). The second and third sets of experiments are conducted to evaluate the performance of our proposed exact and approximation algorithms for PMCLP-PCR-QoS with $\Xi = \{\xi_k = k\}_{k=1}^{m}$ and for 1D-PMCLP-PC-QoS with $\Xi_{s_j} = \{\xi_{s_j}^1 = j\}$ for all $j \in P$. We assume that $\eta(z) = z$. All three sets of experiments are conducted by solving randomly generated instances. We implemented these experiments in the Python 3.5.5 and ran them on a Dell Precision 5820 workstation with 3.80 GHz Intel Core i7-9800 Processor with 32.0 GB RAM and Windows 10.

6.1 Instance Generation Procedure

To generate instances for our computational experiments, we adopt the instance generation procedure of Song et al. (2006), Bansal and Kianfar (2017). We consider a square region of size 1000×1000 in which the DZs are located. Each rectangular DZ, $d \in \mathcal{D}$, is specified by lower left corner coordinates (x_d, y_d), width w_d, length l_d, and reward rate v_d. The coordinates of lower left corner (x_d, y_d) of each DZ is generated as follows. First, we randomly generate three center points in the mentioned square region using a uniform distribution. These center points are referred to as concentration points and we associate a circular region of radius $r = 270$ around each center point. To randomly generate and position each DZ, we first decide whether it is "anchored" to a concentration point (with probability of 0.31 for each concentration point), or it is "free" (with probability of 0.07). The lower left corner of an anchored DZ is randomly located within the circular region assigned to the corresponding concentration point, while the free DZ is randomly located anywhere in the entire square region (1000×1000). This DZ positioning approach is used to mimic the real-word situation where the large percentage of demand is concentrated around some population centers, while a small percentage is scattered all over the region. If the DZ is anchored, the x (and y) coordinate of its lower left corner are generated from a normal distribution, where mean is the x (and y) coordinate of the corresponding concentration point and standard deviation is $r/3$. If the DZ is free, it will be placed in the square region using a uniform distribution. The width and length of each DZ follow a uniform distribution with $w_d, l_d \sim$ uniform$[5, 50]$. The reward rate is also generated based on a uniform distribution where $v_d \sim$ uniform$[1, 10]$. Also, we consider the base SZ

s_0 of dimensions $(w_{s_0}, h_{s_0}) = (50, 40)$. The instance generation for 1D-PMCLP-PC-QoS follows the same procedure on a line segment of length 1000 units on x-axis where $w_d = 0$ and $y_d = 0$ for all $d \in \mathcal{D}$, and $w_{s_0} = 0$.

6.2 Computational Results for PMCLP-PCR

In Table 1, we present results of our first set of computational experiments where we compare the performance of our new exact algorithm (that incorporates Theorem 2) with an existing (benchmark) algorithm of Bansal and Kianfar (2017) for PMCLP-PCR. Each row in Table 1 represents the average over 10 instances. In this table, we use N, T, and T_1 to denote the total number of branch-and-bound nodes, total solution time (in seconds), and time taken in seconds to find an optimal solution, respectively. The subscripts $\mathcal{S}$ and $\mathcal{OS}$ denote the results for the existing algorithm (Bansal and Kianfar 2017) and our proposed algorithm, respectively. These subscripts represent that in the former all the SCVs are considered in the solution search space, whereas in the latter the solution space is reduced to only OSCVs. Table 1 shows that our proposed method always has lesser number of nodes traversed, time taken to solve, and time taken to find an optimal solution, compared to the benchmark. The last three columns provides percentage improvement in N, T, and T_1 that are quantified by $\texttt{ImprN} := 100 \times \left(1 - \frac{N_{\mathcal{OS}}}{N_{\mathcal{S}}}\right)$, $\texttt{ImprT} := 100 \times \left(1 - \frac{T_{\mathcal{OS}}}{T_{\mathcal{S}}}\right)$, and $\texttt{ImprT}_1 := 100 \times \left(1 - \frac{T_{1\mathcal{OS}}}{T_{1\mathcal{S}}}\right)$, respectively. Notice that for $p = 2$, 3, and 4, we have in average 32.06%, 33.23%, and 41.36% improvements, respectively, in terms of number of traversed nodes (N) as well as 35.67%, 26.89%, and 39.67% average improvements, respectively, in terms of total time (T). Both N and T have maximum reduction of 67% and 64.9%, respectively, using our proposed method. However, the improvement in terms of the time taken to find an optimal solution (T_1) has more variation among different number of DZs (varies between 0.0% and 87.5%). For instance, when $p = 2$, the T_1 improves in average 35.4%, and when $p = 3$ and $p = 4$, we get even better average reduction of 58.25% and 51.43%, respectively.

6.3 Computational Results for PMCLP-PCR-QoS with $m \geq 2$

Through the results of the second set of experiments, we analyze the performance of our proposed greedy approximation and branch-and-bound exact algorithms for PMCLP-PCR-QoS with $|\Xi| = m \geq 2$. These experiments are conducted on randomly generated instances with number of SZs $p \in \{2, 3, 4\}$. The results are presented in Table 2, where each row provides an average over results for 10 instances. Columns labelled as T and T_1 report the total solution time and time taken to find an optimal solution, respectively.

Table 1: Computational results for PMCLP-PCR

p	n	This Paper			Bansal and Kianfar (2017)			Improvement (%)		
		N_{OS}	T_{OS}(s)	$T_{1_{OS}}$(s)	N_S	T_S(s)	T_{1S}(s)	ImprN	ImprT	ImprT_1
2	10	29	0.013	0.000	88	0.037	0.000	67.0	64.9	0.0
	20	201	0.095	0.010	303	0.159	0.044	33.6	40.0	77.2
	30	297	0.187	0.055	356	0.226	0.082	16.6	17.3	32.9
	50	564	0.449	0.075	666	0.604	0.098	15.3	25.7	48.9
	70	651	0.674	0.094	976	1.028	0.105	33.3	34.5	10.4
	100	3,601	5.448	1.253	4,843	7.962	1.31	26.6	31.6	4.3
3	10	2,801	1.371	0.021	3,806	2.053	0.023	26.4	33.3	9.5
	20	9,395	7.457	0.753	17,748	13.98	1.427	47.0	46.7	47.2
	30	62,563	66.61	3.92	86,697	85.60	9.37	27.8	22.1	58.1
	50	109,065	155.6	9.024	146,582	216.6	32.76	25.6	28.4	72.4
	70	237,434	354.6	5.742	290,098	389.3	28.73	18.1	8.99	80.0
	100	545,852	1,311	31.54	1.21×10^6	2,731	178.7	54.5	51.9	82.3
4	10	894,255	720.4	0.012	1.9×10^6	1,782	0.096	53.9	59.5	87.5
	20	1.4×10^6	1,904	397.3	2.2×10^6	2,513	441.5	34.9	24.2	10.0
	30	6.9×10^6	10,390	1,002	10.7×10^6	16,132	2,321	35.3	35.6	56.8

Both T and T_1 are reported in seconds. In this experiment, we set a time limit of 5 hours (18,000 seconds). Therefore, the algorithm terminates after 5 hours and reports the best solution found in these 5 hours. If we cannot guarantee that we have found an optimal solution within the mentioned time limit, we will use $18,000+$ in the T column and '-' in the T_1, T_1/T, and α columns. Also, we report the total number of nodes traversed in our proposed B&B tree at the termination.

Table 2 illustrates that the difficulty of PMCLP-PCR-QoS substantially increases as the number of SZs (p) increases. We have already shown that PMCLP-PCR-QoS is NP-hard when p is a part of the input. However, our proposed method can solve relatively large instances in a short amount of time. For example, for $p = 2$, $n = 10$ (number of DZs), and $m \in \{2, 3, 4, 5\}$, the total solution time (T) of our proposed algorithm is less than 1 seconds. Also, for $p = 2$, $n \leq 100$, and $m \in \{2, 3, 4\}$, the total time is less than 1 minutes. When $p = 3$, $n \leq 50$, and $m = 2$, the maximum total time is around 7 minutes. Moreover, the time taken by our proposed method to solve the provided instances is greatly smaller than the time taken by the explicit enumeration of all CVs. For example, in case $p = 2$,

Table 2: Computational results for PMCLP-PCR-QoS with $m \geq 2$

p	m	# of DZs	# of Nodes	$T(s)$	$T_1(s)$	T_1/T	$T_H(s)$	α
2	2	10	96	0.040	0.000	0.000	0.042	0.999
		50	2,915	3.042	0.250	0.082	3.251	0.996
		100	13,092	19.19	5.681	0.295	23.82	0.990
	3	10	223	0.083	0.000	0.000	0.072	0.999
		50	7,983	7.082	1.436	0.202	5.712	0.986
		100	23,379	37.63	5.347	0.142	35.45	0.993
	4	10	857	0.346	0.018	0.052	0.124	0.993
		50	12,292	12.70	1.934	0.152	6.138	0.990
		100	37,655	69.37	12.89	0.185	47.81	0.984
	5	10	2,121	0.803	0.137	0.170	0.13	0.996
		50	37,019	33.27	2.94	0.088	7.71	0.999
		100	66,661	113.64	34.59	0.268	57.64	0.981
3	2	10	51,803	32.12	4.27	0.132	0.063	0.992
		25	215,453	210.42	33.93	0.161	7.322	0.982
		50	326,283	422.35	129.83	0.307	49.81	0.974
	3	10	77,092	38.84	0.953	0.024	0.122	0.989
		25	511,442	442.15	103.26	0.233	11.28	0.967
		50	2.2×10^6	3,342.2	403.29	0.120	74.351	0.985
	4	10	550,522	287.60	43.54	0.151	0.127	0.995
		25	2.2×10^6	1,968.9	193.50	0.098	13.02	0.996
		50	5.8×10^6	9,248.8	1,679	0.181	94.64	0.973
	5	10	1.9×10^6	1,161.8	258.16	0.222	0.169	0.979
		25	4.2×10^6	3,430.3	429.84	0.125	15.08	0.987
		50	8.4×10^6	12,476	1965.9	0.157	114.5	0.986
4	2	10	4.1×10^6	3,287.1	196.05	0.059	0.137	0.988
		25	9.6×10^6	11,827	350.70	0.029	10.95	0.995
		50	10.2×10^6	18,000+	{	{	66.41	{
	3	10	6.9×10^6	4,818.7	0.288	0.058	0.161	0.987
		25	12.8×10^6	14,696	1,218.4	0.082	12.13	0.982
		50	19.3×10^6	18,000+	{	{	98.91	{
	4	10	9.9×10^6	14,570	1,321.3	0.09	0.184	0.986
		25	13.7×10^6	15,085	1,776.7	0.118	17.64	0.973
		50	21.8×10^6	18,000+	{	{	121.7	{
	5	10	14.8×10^6	16,873	569.25	0.033	0.221	0.998
		25	22.5×10^6	18,000+	{	{	21.63	{
		50	31.2×10^6	18,000+	{	{	145.6	{

$m = 2$, and $n = 100$, we need to traverse $1600 \times 1600 = 2,560,000$ number of nodes in the explicit enumeration, whereas our proposed method find an optimal solution by only traversing $13,092$ number of nodes in the B&B tree. In other words, our proposed algorithm benefits from the implicit enumeration of breakpoints by concentrating on areas with high rewards, and using good lower bounds that are provided by our proposed greedy algorithm. The column labelled as T_1/T provides the ratio of T_1 and T. Based on Table 2, the values of the column T_1/T are quite small (i.e., mostly less than 0.2). This means that because of starting from a good lower bound, our proposed algorithm reaches an optimal solution fast and spends the remaining time to prove the optimality of this solution.

One of the advantages of proposing the exact algorithm for PMCLP-PCR-QoS is to evaluate the performance of any heuristic/approximation algorithm for it. In Table 2, the column T_H refers to the total time taken by the proposed greedy algorithm to solve PMCLP-PCR-QoS instances. As we see, the T_H is smaller than the T as number of SZs (p) and number of possible scaling factors (m) increase. The column labelled as α provides the empirical approximation ratio of the reward captured by the greedy algorithm to the optimal covered reward by the exact algorithm. Based on Theorem 1, the approximation ratio of the proposed greedy solution is $1 - \frac{1}{e}$ (almost 63.2%). However, after finding an optimal solution, we can ensure that the greedy algorithm suboptimal solution has a comparably good quality. For example, for the instances considered in Table 2, this ratio is at least 96.9%. In conclusion, Table 2 will help in deciding whether to use the greedy or the exact algorithm depending on the input parameters of PMCLP-PCR-QoS, and the desired tolerance and available computational resources. For example, when $p = 2$, the difference between T_H and T is few seconds, as a result, we prefer to use the exact algorithm and reach to the optimal coverage. However, for $p = 4$, the gap between T_H and T is huge, thus, one may prefer to use faster greedy algorithm and find a good suboptimal coverage instead of an optimal solution. Instances for which we could not obtain optimal solution, we cannot guarantee the high quality of the greedy solution because we cannot compute α.

6.4 Computational Results for 1D-PMCLP-PCR-QoS

In the third set of experiments, we plan to evaluate the performance of our proposed greedy approximation and branch-and-bound exact algorithms for 1D-PMCLP-PC-QoS with $\Xi_s = \{\xi_s\}$, for all $s \in \mathcal{S}$. These experiments are performed on the randomly generated instances with number of SZs $p \in \{2, 3, 4\}$, and the results of the experiments are reported in Table 3. In Table 3, the columns labelled as T, T_1, T_1/T, T_H, and α are defined similar to the Table 2. When an optimal solution cannot be found within time limit of 18000

seconds, we use $18,000+$ in the T column and '-' in the T_1, T_1/T, and α columns. Based on Table 3, our proposed exact method can solve the 1D-PMCLP-PC-QoS instances in less than 1 second when $p = 2$. Also, in case $p = 3$, our algorithm solved the instances with number of DZs $n \in \{10, 20, 30, 50, 70\}$ in less than 1 minutes. However, when p increases (e.g., $p = 4$) the total solution time (T) grows exponentially.

Table 3: Computational results for 1D-PMCLP-PCR-QoS

p	# of DZs	# of Nodes	$T(s)$	$T_1(s)$	T_1/T	$T_H(s)$	α
2	10	48	0.020	0.003	0.149	0.0003	0.989
	20	86	0.049	0.01	0.204	0.0121	0.983
	50	196	0.191	0.033	0.172	0.0672	0.981
	70	258	0.336	0.078	0.232	0.1332	0.989
	100	381	0.689	0.129	0.187	0.2412	0.995
3	10	1,438	0.750	0.159	0.212	0.0008	0.987
	20	5,163	4.228	1.079	0.255	0.0243	0.986
	50	32,558	53.96	12.09	0.224	0.1212	0.988
	70	26,063	57.23	12.37	0.216	0.1872	0.985
	100	55,623	166.4	40.14	0.241	0.2799	0.994
4	10	88,812	60.36	11.20	0.185	0.012	0.967
	20	1.05×10^6	1,226	91.73	0.074	0.066	0.994
	50	4.39×10^6	10,258	2,886	0.281	0.2231	0.983
	70	5.87×10^6	17,308	6,125	0.353	0.4215	0.984
	100	6.32×10^6	18,000+	{	{	0.749	{

Similar to the second set of experiments, it can be observed from Table 3 that T_H (time taken by greedy approximation method for 1D-PMCLP-PC-QoS) is always less than 1 second. The last column, α, refers to the ratio of reward captured by the greedy approach to the optimal covered reward. Table 3 represents that α is at least 0.967. Therefore, Table 3 represents the existing trade off between computational time and optimal/suboptimal coverage in the 1D-PMCLP-PC-QoS. It can help in deciding whether to use greedy or exact algorithm for a specific instances category depending on requested accuracy and available computational resources.

7 Conclusion

We introduced a new generalization of the classical planar maximum coverage location problem where demand zones (DZs) and service zone (SZ) of facilities are represented by two-dimensional spatial objects (e.g., polygons, circles, etc.), the partial coverage is

allowed in its true sense, facilities are allowed to be located anywhere on the continuous plane, and each facility has adjustable service range or quality of service (QoS). We denoted this problem by PMCLP-PC-QoS. We presented greedy and psuedo-greedy algorithms for the PMCLP-PC-QoS that have approximation ratio of $1 - 1/e$ and $1 - 1/e^{\eta}$ for $\eta \leq 1$, respectively. We investigated theoretical properties of the objective function of PMCLP-PC-QoS with rectangular DZs and SZs, denoted by PMCLP-PCR-QoS, and reduced the solution search space. We also proposed exact branch-and-bound based algorithm for the PMCLP-PCR-QoS and one-dimensional PMCLP-PC-QoS (1D-PMCLP-PC-QoS) with SZs of different dimensions. For PMCLP-PCR-QoS with fixed and same QoS ($m = 1$), the proposed algorithm is faster than the existing benchmark algorithm (Bansal and Kianfar 2017). Based on the computational results, we observed that the proposed greedy and exact algorithms for PMCLP-PCR-QoS and 1D-PMCLP-PC-QoS are computationally efficient and effective as well.

Acknowledgments. This research is funded by Automotive Research Center (ARC) in accordance with Cooperative Agreement W56HZV-19-2-0001 U.S. Army CCDC Ground Vehicle Systems Center (GVSC) Warren, MI and by National Science Foundation Grant CMMI– 1824897, which are gratefully acknowledged.

References

Bansal M, Kianfar K (2017) Planar maximum coverage location problem with partial coverage and rectangular demand and service zones. *INFORMS Journal on Computing* 29(1):152{169.

Berman O, Krass D (2002) The generalized maximal covering location problem. *Computers & Operations Research* 29:563{581, ISSN 0305-0548.

Berman O, Krass D, Drezner Z (2003) The gradual covering decay location problem on a network. *European Journal of Operational Research* 151(3):474{480.

Brimberg J, Korach E, Eben-Chaim M, Mehrez A (2001) The capacitated p-facility location problem on the real line. *International Transactions in Operational Research* 8(6):727{738, URL http://dx.doi.org/10.1111/1475-3995.t01-1-00334.

Brimberg J, ReVelle C (1998) Solving the plant location problem on a line by linear programming. *TOP: An O cial Journal of the Spanish Society of Statistics and Operations Research* 6(2):277{286, URL http://dx.doi.org/10.1007/BF02564792.

Chung C (1986) Recent applications of the maximal covering location planning (M.C.L.P.) model. *The Journal of the Operational Research Society* 37:735{746, ISSN 0160-5682.

Church RL (1984) The planar maximal covering location problem. *Journal of Regional Science* 24:185, ISSN 00224146.

Church RL, Murray AT (2013) *Location covering models history applications and advancements* (Springer).

Church RL, ReVelle C (1974) The maximal covering location problem. *Papers of the Regional Science Association* 32:101{118.

Church RL, Roberts KL (1983) Generalized coverage models and public facility location. *Papers of the Regional Science Association* 53(1):117{135.

Cornuejols G, Fisher ML, Nemhauser GL (1977) Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. *Management Science* 23(8):789{ 810.

Current JR, Schilling DA (1990) Analysis of errors due to demand data aggregation in the set covering and maximal covering location problems. *Geographical Analysis* 22:116{126, ISSN 1538-4632.

Daskin MS, Haghani AE, Khanal M, Malandraki C (1989) Aggregation e ects in maximum covering models. *Annals of Operations Research* 18(1-4):115 { 139, ISSN 02545330.

Drezner Z, Hamacher HW, eds. (2002) *Facility Location: Applications and Theory* (New York: Springer), ISBN 3540213457.

Drezner Z, Wesolowsky GO, Drezner T (2004) The gradual covering problem. *Naval Research Logistics (NRL)* 51(6):841{855.

Farahani RZ, Asgari N, Heidari N, Hosseininia M, Goh M (2012) Covering problems in facility location: A review. *Computers & Industrial Engineering* 62(1):368{407.

Hochbaum DS, Pathria A (1998) Analysis of the greedy approach in problems of maximum k-coverage. *Naval Research Logistics* 45(6):615{627.

Mehrez A, Stulman A (1982) The maximal covering location problem with facility placement on the entire plane. *Journal of Regional Science* 22(3):361{365, URL `http://dx.doi.org/https://doi.org/10.1111/j.1467-9787.1982.tb00759.x`.

Murray AT (2016) Maximal Coverage Location Problem: Impacts, Signi cance, and Evolution. *International Regional Science Review* 39(1):5{27.

Murray AT, O'Kelly ME (2002) Assessing representation error in point-based coverage modeling. *Journal of Geographical Systems* 4:171{191, ISSN 1435-5930.

Murray AT, Tong D (2007) Coverage optimization in continuous space facility siting. *International Journal of Geographical Information Science* 21:757, ISSN 1365-8816.

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

Song D, van der Stappen A, Goldberg K (2006) Exact algorithms for single frame selection on multiaxis satellites. *IEEE Transactions on Automation Science and Engineering* 3:16{28, ISSN 1545-5955.

Tong D, Church RL (2012) Aggregation in continuous space coverage modeling. *International Journal of Geographical Information Science* 26(5):795{816.

Tong D, Murray AT (2009) Maximising coverage of spatial demand for service. *Papers in Regional Science* 88:85{97, ISSN 10568190.

Watson-Gandy C (1982) Heuristic procedures for the m-partial cover problem on a plane. *European Journal of Operational Research* 11(2):149 { 157, ISSN 0377-2217, URL http://dx.doi.org/https://doi.org/10.1016/0377-2217(82)90109-6.

Xu Y, Song D, Yi J (2010) Exact algorithms for non-overlapping 2-frame problem with non-partial coverage for networked robotic cameras. *IEEE Conference on Automation Science and Engineering (CASE)*, 503{508.

Xu Y, Song D, Yi J, Stappen AFvd (2008) An approximation algorithm for the least overlapping p-Frame problem with non-partial coverage for networked robotic cameras. *2008 IEEE International Conference on Robotics and Automation*, 1011{1016.