UCA
Universidad de Cádiz

Estudio de Problemas de Clasificación Supervisada y de Localización en Redes mediante Optimización Matemática

---

Supervised Classification and Network Location Problems via Mathematical Optimization

Marta Baldomero Naranjo

Tesis doctoral

Director: Dr. Antonio Manuel Rodríguez Chía

Septiembre 2021

Departamento de Estadística e Investigación Operativa

Agradecimientos

La tesis ha sido una etapa de crecimiento como científica y como persona. En estas líneas me gustaría dar las gracias a todos los que me han acompañado estos cuatro años.

Me gustaría comenzar dando las gracias a mi director, Antonio Manuel Rodríguez Chía. Una de las personas más entregadas con su trabajo que conozco. Él ha sido mi guía, me ha llevado de la mano en mis primeros pasos en el mundo de la investigación, estando siempre disponible cuando lo he necesitado tanto a nivel científico como a nivel personal. Gracias por vivir la vida con alegría, por levantarte ante las adversidades, por nunca rendirte y por recibirme siempre con una sonrisa.

Me gustaría agradecer a mi grupo de investigación, OREL, lo bien que me han acogido, gracias por escucharme y darme la oportunidad de trabajar juntos. Me gustaría dar las gracias también al departamento de Estadística e Investigación Operativa de la Universidad de Cádiz. En especial, gracias a Alfonso y Toñi por vuestros consejos y a todos los compañeros con los que he compartido docencia por vuestra ayuda. A mis profesores de Matemáticas que desde colegio hasta el máster me han enseñado a apreciar la belleza de esta ciencia.

En estos cuatro años he vivido muchas experiencias que me han hecho crecer: los congresos, las reuniones de la Red de Localización, las escuelas de doctorado, mi estancia en la Universidad de Edimburgo, etc. Me gustaría agradecer a todos los investigadores con los que he coincidido en estas experiencias que compartierais conocimiento, pero sobretodo que hayáis llenado mi vida de buenos ratos y momentos inolvidables. Gracias a mis coautores por enseñarme nuevas perspectivas para afrontar los problemas y por vuestra paciencia conmigo, he aprendido muchísimo con vosotros.

Agradecer también a todas las mujeres y los hombres que lucharon por la igualdad. Gracias a vuestro trabajo, he podido estudiar y realizar esta tesis. En estas líneas me gustaría destacar a María del Carmen Martínez Sancho, la primera mujer en doctorarse en Matemáticas en España en 1928.

A todos los europeos que contribuyen con sus impuestos a financiar la investigación pública, gracias a estos fondos he podido realizar mi tesis. En concreto, esta investigación se ha financiado con los proyectos públicos: MTM2016-74983-C2-2-R (Agencia Estatal de Investigación, Española y el Fondo Europeo de Desarrollo Regional (FEDER)), FEDER-UCA18-106895 y P18-FR-1422 (Junta de Andalucía y FEDER) y el contrato predoctoral UCA/REC01VI/2017 (Universidad de Cádiz). Agradecer también la financiación recibida por la la BritishSpanish Society y Telefónica (Scholarship Programme 2018) para realizar la estancia en la Universidad de Edimburgo.

Sin la financiación recibida no podría haber realizado mi tesis, pero tampoco la podría haber hecho sin el apoyo recibido por la familia y los amigos. Gracias por esos *desayunos energéticos* a los compañeros de la facultad. Gracias a la *overcrowded office* por los ratos de compartir dramas y cotilleos. En especial, gracias a Luisa por ser una hermana mayor, por ayudarme siempre a todo, por aconsejarme y por haber hecho estos cuatro años más fáciles. A Juanma por recordarnos que si las cosas pueden salir mal, saldrán peor. Patri, compañera de camino, somos amigas desde que empezamos la carrera, que regalo más bonito me ha hecho la vida teniéndote a mi lado todo este tiempo y lo que nos queda.

A mis amigos, a todos lo que habéis aguantado que llegara tarde porque tenía que ponerle “tarea” al ordenador o que me quedara en mi mundo porque acababa de tener una idea. Gracias a los que habéis confiado en mí desde el primer día y me habéis apoyado con todo, hasta en las votaciones para el nombre de la tesis. Gracias a mi familia lasaliana y a Ana, Blanca, Estefi, Javi, Patri, Sandra y Sara*s*.

A mi familia, mis abuelas, mis tíos, mis primos, a Elvira, a Esteban y a India, gracias por hacerme feliz y llenar mi vida de cariño.

A mis padres, gracias por la educación que me habéis dado, por enseñarme que estaba prohibido pasárselo mal en el cole, por transmitirme esa pasión por aprender. Gracias por vuestro amor infinito, por vuestra paciencia conmigo cuando tenía un mal día (en el confinamiento hubo demasiados), por vuestro apoyo, gracias a vosotros he llegado hasta aquí. Sois los padres que siempre quise tener. A mi hermano, Sergio, porque contigo aprendí el significado de la palabra incondicional. Siempre, pase lo que pase, fuiste, eres y serás mi favorito.

A Esteban, mi compañero de vida. Juntos formamos un gran equipo. Gracias por cuidarme tanto, muchas veces más que yo a mí misma. Gracias por llenar de paz mi vida, por ser mi faro, por demostrarme que la vida es cuestión de prioridades.

Espero que la defensa de esta tesis sea solo el fin de la primera etapa de un largo camino como investigadora, muchas gracias a todos los que me habéis acompañado y a los que me ayudasteis a llegar hasta aquí.

Resumen

Esta tesis doctoral aborda varios problemas en los campos de la Clasificación Supervisada y la Teoría de la Localización empleando herramientas y técnicas propias de la Optimización Matemática. A continuación, se hace una breve descripción de estos problemas y de las metodologías propuestas para su análisis y resolución.

En el primer capítulo se discuten en detalle los fundamentos de la Clasificación Supervisada y de la Teoría de la Localización, enfatizando los aspectos estudiados en esta tesis. En los dos capítulos siguientes se analizan problemas de Clasificación Supervisada. En particular, el Capítulo 2 propone procedimientos exactos de resolución para varios modelos de Máquinas de Vectores de Soporte (SVM) con *ramp loss*, un método de clasificación conocido por limitar la influencia de los valores atípicos. Los modelos resultantes se analizan para obtener una cota inicial de los parámetros *M grande* incluidos en la formulación. Posteriormente, se proponen enfoques de resolución basados en tres estrategias para obtener valores más ajustados de dichos parámetros. Dos de ellas requieren resolver una secuencia de problemas de optimización continuos, mientras que la tercera utiliza el método de la relajación lagrangiana. Los procedimientos de resolución derivados son válidos para las formulaciones *ramp loss* con norma ℓ_1 y norma ℓ_2. Estos algoritmos se han probado y comparado con los procedimientos de resolución existentes, tanto en conjuntos de datos simulados como reales, mostrando la eficacia de la metodología desarrollada.

El Capítulo 3 presenta un nuevo clasificador basado en SVM que simultáneamente aborda la limitación de la influencia de los valores atípicos y la selección de características. La influencia de los valores atípicos se controla mediante el criterio *ramp loss*, mientras que el proceso de selección de características se lleva a cabo incluyendo una nueva familia de variables binarias y varias restricciones. El modelo resultante se formula como un modelo de programación entera mixta con parámetros *M grande*. En este capítulo, se analizan las características del modelo y se proponen dos algoritmos de resolución diferentes (exacto y heurístico). El rendimiento del clasificador obtenido se compara con varios clasificadores en diversos conjuntos de datos.

Los dos capítulos siguientes abordan problemas de localización, en particular, el problema de máximo cubrimiento (MCLP) en redes. Se estudian dos variantes de este problema que responden al modelado de dos escenarios diferentes, con y sin incertidumbre en los datos de entrada. En primer lugar, en el Capítulo 4, se presenta el MCLP en redes con mejoras permitiéndose la reducción de la longitud de las aristas. Este problema tiene como objetivo ubicar p instalaciones en los nodos (de la red) para maximizar la cobertura, considerando que la longitud de las aristas puede reducirse dentro de un presupuesto. Por lo tanto, tenemos que decidir: la ubicación óptima de las p instalaciones y las reducciones óptimas de la longitud de las aristas. Para resolverlo, proponemos tres formulaciones enteras mixtas y una fase de preprocesamiento para fijar variables y eliminar algunas de las restricciones. Además, analizamos las características de estas formulaciones para fortalecerlas proponiendo desigualdades válidas. Por último, comparamos las tres formulaciones y sus correspondientes mejoras probando su rendimiento en diferentes conjuntos de datos.

El siguiente capítulo, Capítulo 5, también considera un MCLP, aunque desde la perspectiva de la incertidumbre. En concreto, este capítulo aborda una versión del MCLP de una sola instalación en una red en la que la demanda está distribuida a lo largo de las aristas y solo se conoce una cota superior y una cota inferior de dicha demanda. Proponemos un modelo que minimiza el peor de los casos (*minmax regret*) en el que la instalación del servicio puede situarse en cualquier punto de la red. Además, presentamos dos algoritmos polinómicos para encontrar la ubicación que minimiza el peor de los casos suponiendo que la realización de la demanda es una función constante desconocida o una función lineal desconocida en cada arista. También incluimos dos ejemplos ilustrativos y un estudio computacional para mostrar el potencial de la metodología propuesta.

Esta tesis doctoral finaliza con las conclusiones de la investigación realizada y la presentación de futuras líneas de trabajo.

Abstract

This PhD dissertation addresses several problems in the fields of Supervised Classification and Location Theory using tools and techniques coming from Mathematical Optimization. A brief description of these problems and the methodologies proposed for their analysis and resolution is given below.

In the first chapter, the principles of Supervised Classification and Location Theory are discussed in detail, emphasizing the topics studied in this thesis. The following two chapters discuss Supervised Classification problems. In particular, Chapter 2 proposes exact solution approaches for various models of Support Vector Machines (SVM) with ramp loss, a well known classification method that limits the influence of outliers. The resulting models are analyzed to obtain initial bounds of the big M parameters included in the formulation. Then, solution approaches based on three strategies for obtaining tighter values of the big M parameters are proposed. Two of them require solving a sequence of continuous optimization problems, while the third uses the Lagrangian relaxation. The derived resolution methods are valid for the ℓ_1-norm and ℓ_2-norm ramp loss formulations. They are tested and compared with existing solution methods in simulated and real-life datasets, showing the efficiency of the developed methodology.

Chapter 3 presents a new SVM-based classifier that simultaneously deals with the limitation of the influence of outliers and feature selection. The influence of outliers is taken under control using the ramp loss margin error criterion, while the feature selection process is carried out including a new family of binary variables and several constraints. The resulting model is formulated as a mixed-integer program with big M parameters. The characteristics of the model are analyzed and two different solution approaches (exact and heuristic) are proposed. The performance of the obtained classifier is compared with several classical ones in different datasets.

The next two chapters deal with location problems, in particular, two variants of the Maximal Covering Location Problem (MCLP) in networks. These variants respond to the modeling of two different scenarios, with and without uncertainty in the input data. First, Chapter 4 presents the upgrading version of MCLP with edge length modifications on networks. This problem

aims at locating p facilities on the nodes (of the network) so as to maximize coverage, considering that the length of the edges can be reduced within a budget. Hence, we have to decide on: the optimal location of p facilities and the optimal edge length reductions. To solve it, we propose three different mixed-integer formulations and a preprocessing phase for fixing variables and removing some constraints. Moreover, we analyze the characteristics of these formulations to strengthen them by proposing valid inequalities. Finally, we compare the three formulations and their corresponding improvements by testing their performance over different datasets.

The following chapter, Chapter 5, also considers a MCLP, albeit from the perspective of uncertainty. In particular, this chapter addresses a version of the single-facility MCLP on a network where the demand is distributed along the edges and uncertain with only a known interval estimation. We propose a minmax regret model where the service facility can be located anywhere along the network. Furthermore, we present two polynomial algorithms for finding the location that minimizes the maximal regret assuming that the demand realization is an unknown constant or linear function on each edge. We also include two illustrative examples as well as a computational study to show the potential of the proposed methodology.

This PhD dissertation ends with the conclusions of the research carried out and the presentation of future lines of work.

Contents

1 Introduction

Mathematical Optimization or Mathematical Programming is a branch of Mathematics whose objective is to provide optimal decisions for problems usually representing real situations. This area comprises the design of mathematical models, the development of theoretical results that characterize their solution set, the use and design of mathematical tools to address the problem, as well as the development of new methodologies to solve the studied problems. The resulting optimization models are characterized by three main elements.

- Sets of *variables* that represents the different decisions, which can be binary, integer, or continuous. $\mathcal{D}$ denotes the domain of the variables.
- The *objective function* to be optimized and the optimality criterion (minimize or maximize).
- A set of *constraints* that states the relationships between the variables and their limitations. These constraints define the feasible region.

Therefore, a mathematical program or formulation can be expressed as:

$$\begin{aligned} \min / \max \quad & f(x), \\ \text{s.t.} \quad & g_i(x) \leq b_i, \quad i = 1, \ldots, m, \\ & x \in \mathcal{D}, \end{aligned}$$

where $g_i : \mathcal{D} \longrightarrow \mathbb{R}$ and $b_i \in \mathbb{R}$, for $i = 1, \ldots, m$.

The origins of Mathematical Programming date back to the studies of Pierre de Fermat and Joseph-Louis Lagrange identifying optima of functions and the iterative methods developed by Sir Isaac Newton and Johann Carl Friedrich Gauss for moving towards an optimum. However, it was not until 1939 when Leonid Vitalyevich Kantorovich in the book *Matematicheskie metody organizatsi i planirovaniya proizvodstva* (translated in Kantorovich, 1960) introduced this subject. Completely independently, George Bernard

Dantzig and John von Neumann introduced in 1947 the Simplex algorithm and the duality theory respectively (Dantzig, 1949; Neumann, 1947).

Since these initial studies, this discipline has not stopped growing. This expansion has been encouraged by the development of computers, which allows us to solve larger and larger problems every day. Thanks to its versatility, Mathematical Optimization can be successfully applied to solve problems in multiple and diverse fields, as for example, Data Analysis (Mangasarian, 1997; Carrizosa and Romero-Morales, 2013), Environmental Sciences (Norton and Hazell, 1986; Archibald and Marshall, 2018; Shakhsi-Niaei et al., 2013), Manufacturing (Dutta and Fourer, 2001; Kallrath, 2000), Medicine (Ehrgott et al., 2010; Rais and Viana, 2011), among others.

In this PhD dissertation, we focus our study in the application of Mathematical Optimization tools in Supervised Classification and in Network Location problems. Next, a brief overview of these two topics will be provided.

1.1 Supervised Classification

Nowadays, a massive amount of data is generated every second. However, the large size of the datasets and their complex inherent properties make difficult to obtain valuable information from them. One of the fields that responds to this need is Classification. The goal of this discipline is to provide a decision rule for classifying individuals of a population in different classes. These problems can be addressed from two main approaches: Supervised and Unsupervised Classification. Supervised Classification considers that the classes in which the individuals should be classified are known in advance. Moreover, the characteristics of some individuals and the classes where they belong to are needed, called training sample. However, Unsupervised Classification does not require prior knowledge of the classes.

This PhD dissertation concentrates on Supervised Classification from a Mathematical Optimization perspective. This perspective was introduced by Olvi Leon Mangasarian (Mangasarian, 1965, 1968), resulting a very useful tool (Lee and Wu, 2009). In particular, we focus our attention on the study of Support Vector Machines (SVM) models. Since their introduction by Cortes and Vapnik (1995) and Vapnik (1998), SVM have been deeply analyzed in the literature. In order to provide a precise description of the problem, we consider the following notation. Given a set N of individuals partitioned into two classes $\mathcal{Y} = \{$-1,1$\}$, each individual $i \in N = \{1, \dots, n\}$ is associated with a pair $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, where d is the number of features analyzed in each individual of N, x_i contains the feature values, and y_i provides the class membership (1 or -1). The aim of the SVM is to build a separating hyperplane

$w \cdot x + b = 0$ to classify new individuals. The search for this hyperplane is based on a compromise between maximizing the distance (margin) between the two parallel hyperplanes supporting individuals of each class and minimizing the error caused by misclassified data.

The maximization of the margin between the parallel hyperplanes associated with each class can equivalently be expressed as the minimization of $\|w\|^\circ$, where $\|\cdot\|^\circ$ is the dual norm of the norm used to measure the distance to the hyperplanes in $\mathbb{R}^d$ (Carrizosa and Romero-Morales, 2013; Gonzalez-Abril et al., 2011; Liu et al., 2007). In Blanco et al. (2020c), the methodology applied to classical SVM (using ℓ_2-norm) is extended to the general ℓ_p-norm with $p \geq 1$. Although we will concentrate in this PhD dissertation on cases $p = 1$ and $p = 2$, we provide the following common formulation for any value $p \geq 1$,

$$\begin{aligned}
(\text{SVM-}\ell_p) \quad \min \quad & \frac{1}{p}\|w\|_p^p + C\sum_{i=1}^{n} \xi_i, \\
\text{s.t.} \quad & y_i\left(\sum_{k=1}^{d} w_k x_{ik} + b\right) \geq 1 - \xi_i, \quad i \in N, && (1.1)\\
& \xi_i \geq 0, \quad i \in N, && (1.2)
\end{aligned}$$

where the w- and b-variables are the separation hyperplane coefficients and the ξ-variables represent the deviation of misclassified data. C is a non-negative constant and $\|w\|_p$ represents the ℓ_p-norm of vector w with $p \geq 1$. Therefore, the objective function given above seeks to maximize the margin (first addend of the objective function) and minimize the sum of misclassified data deviations (second addend). Parameter C regulates the trade-off between both goals.

Since their introduction, SVM have been applied in many fields, including: biology (Zhang et al., 2006; Guerrero et al., 2012; Joloudari et al., 2019), medicine (Furey et al., 2000; Ren, 2012; Wang et al., 2018), bioinformatics (Byvatov and Schneider, 2003; Chen et al., 2017), machine vision (Barkana et al., 2017; Decoste and Schölkopf, 2002; Rehman et al., 2019), text classification (Zhang et al., 2008), finance (Gavrishchaka and Banerjee, 2006; Maldonado et al., 2017; Min and Lee, 2005; Ghoddusi et al., 2019), and sustainability (Mrówczyńska et al., 2019; Kim et al., 2019). More applications can be found in Cervantes et al. (2020).

Although classical SVM models have high predictive power in comparison with other state-of-the-art classifying methods, some drawbacks also arise from their use. In the next subsections we focus our attention in two of them: influence of outliers and feature selection. New techniques and models have been proposed in this PhD dissertation to address these issues.

1.1.1 Influence of outliers

As stated in Hastie et al. (2009), classical SVM is not robust against outliers and this classification method does not perform efficiently when the analyzed dataset contains anomalous values. One of the main reason for that is that the classical formulation of SVM uses unbounded continuous variables for measuring the deviations of misclassified data: ξ_i, for $i \in N$.

In this context, an outlier is an individual whose feature values are anomalous, i.e., these values are very different from the values of most individuals in the sample. The presence of outliers in real-life data is common and is caused by several factors which include but are not limited to: transcription errors, measurement errors, mislabelling, data processing errors, intentional instances (e.g. spam). There are many papers in the literature that use a SVM based model to address these issues, see for instance, Xu et al. (2017); Zhang et al. (2009); Lukashevich et al. (2009); Brooks (2011); Blanco et al. (2020a).

In particular, we focus on the approach introduced by Brooks (2011): SVM with ramp loss. In this model, the deviation of misclassified data is truncated to avoid extreme values: i.e., for a given deviation, the penalization is bounded. Due to its usefulness as a classifier, this model has attracted the attention of many researchers and has been analyzed whilst considering the ℓ_1-norm and ℓ_2-norm (see Brooks, 2011; Carrizosa et al., 2014; Belotti et al., 2016; Blanco et al., 2020b). This model has also been studied under the framework of statistical learning theory (see Huang et al., 2014).

In spite of the interest of this model for data classification avoiding the effect of outliers, the search for exact solution approaches for large datasets is still an ongoing problem, as stated in Duarte Silva (2017).

1.1.2 Feature Selection

In addition to the aforementioned issue of the presence of outliers in the data, we would like to emphasize another usual difficulty in classification, the identification of relevant features. Several real-life datasets contain a high number of features for each individual of the sample. Many often, some of them do not provide valuable information in the classification process. For this reason, it is interesting to identify and remove them in a process called feature selection. Also, another interesting aspect to carry out feature selection is that in real life, analyzing many features could imply high costs.

For example, if we want to classify whether or not a patient has a disease, each feature usually represents a medical analysis and each of these has a cost in money and time. Usually, it feels that the more tests are performed, the better the diagnosis. However, the performance of a large number of medical tests can lead to high costs for the health system or the patient, saturation of the health system, inconvenience for the patient (invasive tests, numerous visits to the clinic, etc.), and longer waiting time to obtain the diagnosis. For example, in the United States, there is a 62% of personal bankruptcies incurred in medical bills, and 90% of patients are in debt, as reported Lee et al. (2020); Jacoby and Holman (2010). Many of the aforementioned drawbacks could be avoided if only the tests that are truly meaningful were carried out. Moreover, classifiers are easier to interpret when the number of features is small rather than the ones that contain many features. Thus, the knowledge of the features that have a relevant influence in a medical diagnosis is highly applicable in real-life because it could help in opening new lines of medical research as new treatments or earlier diagnosis. Although we have given an example of the feature selection utility in medicine, similar situations can appear in many other fields of knowledge.

In general, feature selection techniques have been divided into three different groups: filter, wrapper and embedded methods.

- Filter methods are based on a preliminary study of each feature's relevance and only features with significant importance are considered for the classification method, see Guyon et al. (2006).
- Wrapper methods interact with the classification method to select the set of relevant features, see Kohavi and John (1997); Alazzam et al. (2020).
- Embedded methods study the feature selection and the classification simultaneously in the same model.

In this context, the weakness of the classical SVM model is that its mathematical formulation does not limit the number of features selected by the classifier. Many recent papers in the literature propose new SVM based classifiers to deal with feature selection. These models can be considered as embedded methods, see for example Maldonado et al. (2014); Aytug (2015); Gaudioso et al. (2017); Ghaddar and Naoum-Sawaya (2018); Jiménez-Cordero et al. (2021); Kunapuli et al. (2008); Labbé et al. (2019); Maldonado et al. (2020); Gaudioso et al. (2020); Nguyen and de la Torre (2010); Cura (2020); Lee et al. (2020).

In view of the above discussion, we can conclude that it would be very interesting to develop a support vector machines model that simultaneously limits the influence of outliers in the classifier and selects the number of features. One advantage of doing these processes simultaneously is that it prevents the loss of valuable information due to the incorrect removal of elements of the sample. Indeed, it could happen that several of the features of an individual in the sample may take anomalous values. However, it does not make sense to exclude this individual if those features do not affect the classification. The other advantage is that doing both procedures simultaneously allows us to identify as outliers data that were not initially classified as such. In fact, an individual might not be identified as outlier by a general outlier detection procedure because most of its features take similar values to the ones of their class and only some of them take very different values. However, if after the feature selection, these features with very different values are the most relevant in the classification process, this individual should be classified as outlier.

1.2 Location Theory

The Location Theory concerns the identification of the optimal site of one or several facilities in a given area to cover the demand of a given set of clients, taking into account the distances/costs between the facilities and clients, the demand of the clients and existing constraints. For this purpose, different fields of Mathematical Optimization are applied to address these types of problems, such as, mathematical modeling, the design of mathematical tools to analyze the resulting models, the characterization of solution sets, the development of efficient solution methods, among others.

The origins of this discipline trace back to ancient Greece. However, it was not until the middle of the twentieth century when this science was tackled from the perspective of Mathematical Optimization. Until that time, the proposed solutions were mainly geometric, as for example, the solution to Fermat's problem given by Evangelista Torricelli. At the end of the 19th century and the beginning of the 20th century, Carl Wilhelm Friedrich Launhardt and Alfred Weber proposed the first models to optimize decision within Location Theory. So far, the models proposed were mainly descriptive. In the 1960s, the seminal papers by Seifollah Louis Hakimi (Hakimi, 1964, 1965) were of particular relevance to set the foundations of this area. Furthermore, the first Mixed Integer Programming (MIP) approaches were derived in this period, see Manne (1964), Balinski (1965). For interested readers about history of Location Theory see Love et al. (1988); Laporte et al. (2019) and references therein.

Today, Location Theory is applied in a wide range of fields such as humanitarian supply chain (Kara and Rancourt, 2019), emergency services (Silva and Serra, 2008; Scaparra and Church, 2019), public services (Fredriksson, 2017), logistics (Melo et al., 2006; Żak and Węgliński, 2014), telecommunications (Gollowitzer and Ljubić, 2011), among others.

In what follows, a brief description of the characteristics of location problems based on the objective function, the solution space, the demand, and the facility locations is given. For a more detailed classification, the reader is directed to Puerto and Rodríguez-Chía (2004).

1.2.1 The objective function

As stated at the beginning of the chapter, one of the main elements of an optimization problem is its objective function. Below we present some of the most classic ones used in location models.

- *Median problem* or *Fermat-Weber problem*: the aim is to minimize the weighted sum of the distances between the facilities and the customers (Hakimi, 1964, 1965; ReVelle and Swain, 1970).
- *Center problems*: the aim is to minimize the largest weighted distance (Hakimi, 1964, 1965).
- *Covering problems*: in these problems a client is considered to be covered if their distance to a facility is within a given coverage radius. Depending on whether the number of services to be located is fixed or not, the following problems are considered:
  - *Set covering problems*: the aim is to minimize the cost or the number of new facilities to be located so that all customers are covered (Toregas et al., 1971).
  - *Maximal covering problems*: the aim is to maximize the covered demand taking into account that the number of facilities to be located is given (Church and ReVelle, 1974).

In some cases, it is difficult to choose a single objective. In this context, models that consider several location criteria emerge, called multi-objective location problems.

1.2.2 Solution space and demand

The space where the problem is stated is called solution space, i.e., the space where the services facilities and clients are located. The most common solution spaces discussed in the literature are:

- *Discrete*: a finite set of potential location for the facilities is given.
- *Continuous*: the facilities can be located at any point of a region, such as $\mathbb{R}^n$, a sphere, an ellipse, etc.
- *Network*: the solution space of the problem is a graph. There are two main classes of problems, either the facilities are located only at the

nodes (discrete space) or they are located at any point in the network (continuous space).

We would like to emphasize that SVM could also be considered as a location problem (Plastria and Carrizosa, 2012). In fact, obtaining a SVM classifier is equivalent to locate a hyperplane in $\mathbb{R}^m$, where m is the number of features of the sample.

In some real-world applications of location problems, the demand is sometimes not known exactly. Depending on the information we have about demand, it can be categorized as deterministic and non-deterministic (uncertain). More concretely, non-deterministic models are usually classified in two different categories:

- *Stochastic optimization*: in these models it is assumed that the demand follows a known probability distribution.
- *Robust optimization*: in these models the demand is complete uncertain and no information about its probability distribution is known. In many cases the only information available is the range of the demand variation. To deal with this situation of total uncertainty, the models usually optimize the worst-case situation. The most common criteria are to minimize the maximal cost (the minmax cost) or to minimize the maximal regret (the minmax regret).

Observe that this uncertainty can exist not only in the demand, but also in the rest of the parameters of the model.

1.2.3 Facility locations

In most location problems, the main decision variables are the locations of the facilities. Firstly, depending on the number of facilities to be located, the problem can be single or multiple. If there is more than one facility, it is necessary to know if all the services are identical or if, on the contrary, they have different properties.

Secondly, the problems are also classified according to the attractiveness of the facilities, in other words, attractive or obnoxious facilities.

- *Attractive facilities:* clients want them to be located nearby since these facilities have a positive effect, e.g., supermarkets, health centers, schools.
- *Obnoxious facilities:* consumers do not want them to be located nearby since these facilities have a negative impact, e.g., garbage bins, landfills, nuclear power plants.

In some cases, the decision variables of the problem are not the location of the services, instead, the location of the facilities is given and the decision variables are other parameters of the model. The following models are some examples:

- *Inverse problems*, the objective is to modify the parameters at minimum cost such that a given feasible solution becomes optimal.
- *Reverse problems*, the objective is to maximally improve a pre-specified solution by changing the parameters within certain limits and subject to a given budget.

There are also models that combine the identification of the optimal location of the facilities with the modification of the model parameters. Examples of this type of models are the up/downgrading problems, where an actor modifies the parameters of the model and then a reactor takes a decision. In upgrading problems, actor and reactor have the same goal; in downgrading problems, their objectives are conflicting.

1.3 Contents of the PhD dissertation

After this presentation of Supervised Classification and Location Theory, we give a brief description of the contents of this PhD dissertation.

- Chapter 2 addresses one of the drawbacks of SVM, the lack of robustness against outliers. This chapter is devoted to the development of mathematical results to improve the exact solution methods of the ramp loss model, a SVM-based classifier that limits the influence of outliers. In this chapter, valid inequalities are developed and several bounds for the big M parameters of the model with ℓ_1 and ℓ_2 are proven. Furthermore, three algorithms to find optimal solutions of the models are presented and tested over synthetic and real-world datasets.
- Chapter 3 presents a new model based on SVM. This model is specifically developed to avoid the issues caused by the presence of outliers in the training data and to enable the modeler to select the number of features to be evaluated in the classification process. The main advantage of this model is that the limitation of the outliers influence and the feature selection are done simultaneously. In this chapter, an algorithm to compute the values of the big M parameters of the model is proposed. Moreover, it is developed a heuristic algorithm based on the Adaptive Kernel Search (Guastaroba et al., 2017) to obtain the classifier. Finally, computational experiments are included

to validate the heuristic algorithm and to show the effectiveness of the classifier.

- Chapter 4 develops a new location model in a network, the upgrading version of the maximal covering location problem. As its name states, the objective is to maximize the coverage. To do so, two decisions have to be made: the location of a fixed number of facilities and the edge length upgrades subject to a budget constraint. Three different mixed integer formulations to solve the problem are proposed. Moreover, a powerful preprocessing phase is developed and several families of valid inequalities are proven. The performance of the three formulations, the preprocessing phase, and the valid inequalities are tested over different datasets.
- Chapter 5 derives a new location model in a network, the minmax regret maximal covering location problem with edge demands. Unlike the previous chapter, we now assume that the demand is completely uncertain and we only know its maximum and minimum values. Moreover, we assume that the demand is distributed along the edges and it is represented as constant and linear functions. To deal with this uncertainty, we minimize the maximal regret considering that the facility can be located anywhere in the network, i.e., it can be located not only in the nodes, but also in any point along the edge. We provide a subdivision of the domain of the objective function, such that the representation of it over each cell is a quadratic function. In addition, we derive two polynomial time algorithms to find the exact solution for constant and linear demand functions respectively. Furthermore, the chapter includes examples to illustrate the proposed methodology and computational experiments over real and synthetic networks.
- Chapter 6 exposes the conclusions of this dissertation and future research lines.

2 Exact approaches for Support Vector Machines with ramp-loss

This chapter considers various models of support vector machines with ramp loss, these being an efficient and robust tool in supervised classification for the detection of outliers. The exact solution approaches for the resulting optimization problem are of high demand for large datasets. Hence, the goal of this chapter is to develop algorithms that provide efficient methodologies to exactly solve these optimization problems. These approaches are based on three strategies for obtaining tightened values of the big M parameters included in the formulation of the problem. Two of them require solving a sequence of continuous optimization problems, while the third uses the Lagrangian relaxation to tighten the bounds. The proposed resolution methods are valid for the ℓ_1-norm and ℓ_2-norm ramp loss formulations. They were tested and compared with existing solution methods in simulated and real-life datasets, showing the efficiency of the developed methodology.

2.1 Introduction

In this chapter, we focus our attention on the challenge of providing efficient algorithms to obtain exact optimal solutions for data classification problems that avoid the effect of outliers. We first consider the ramp loss model using the ℓ_1-norm and then the presented strategies are adapted to the ℓ_2-norm. The analysis of these two norms is fully justified because the ℓ_1-norm has various interesting properties, such as its sparsity, whilst the ℓ_2-norm is the one used in classic SVMs. Indeed, using the ℓ_1-norm when the dataset contains many features is worthwhile as it results in easily interpreted classifiers with a reduced number of selected features (see Gaudioso et al. (2017); Labbé et al. (2019); Maldonado et al. (2014)).

Brooks (2011) introduces a new integer programming formulation for SVM with ramp loss that accommodate the use of nonlinear kernel functions. However, he proved that SVM with ramp loss can produce robust classifiers when using a linear kernel in the presence of outliers. For this reason, we concentrate on SVM ramp loss models using a linear kernel.

The developed methodologies in this chapter are compared with the methods proposed in Belotti et al. (2016). In particular, Belotti et al. (2016) introduce a non convex formulation for the SVM with ramp loss using the ℓ_2-norm. They also propose two methods based on bound tightening approaches for efficiently solving the model. One of the methods is focused on an iterative tightening of the bounds of variables providing locally valid bounds for the big M parameters appearing in the model. This method is included in CPLEX arsenal and it is known as *local implied bound cuts*. The second method tightens the bounds of the variables iteratively by solving MIPs. It should be remarked that the model studied in Belotti et al. (2016) provide an unique initial big M parameter which is valid for all the family of constraints in which these parameters appear.

In contrast, this work proposes strategies that provide different tightened bounds on each big M parameter. In addition, unlike Belotti et al. (2016), our strategies are based on the tightening of bounds by iteratively solving linear programming problems in the ℓ_1-norm case and quadratic ones in the ℓ_2-norm case, but non integer. In the computational results section we will analyze the positive effect of our approaches.

The remainder of the chapter is organized as follows. Section 2.2 introduces the model and presents a set of valid inequalities for the formulation. In Section 2.3, tightened values for the big M parameters in the model with ℓ_1-norm are proven and strategies for obtaining them are proposed. Some of these strategies are based on tightening values of the w-variables and others are based on the Lagrangian relaxation of the model. In Section 2.4, all the previously proposed strategies are adapted to the ℓ_2-norm model. Section 2.5 contains computational experiments carried out on simulated and real-life datasets. Our conclusions and other potential research topics are included in Section 2.6.

2.2 The model

We will focus our research on the Support Vector Machine model with ramp loss, as introduced by Brooks (2011). The model which uses the ℓ_p-norm for any $p \geq 1$ is formulated as a mixed integer program with conditional constraints:

$$(\text{RL-}\ell_p) \quad \min \quad \frac{1}{p}(||w||_p)^p + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right)$$

$$\text{s.t.} \quad \text{if } z_i = 0, \quad y_i\left(\sum_{k=1}^{d} w_k x_{ik} + b\right) \geq 1 - \xi_i, \quad i \in N, \tag{2.1}$$

$$0 \leq \xi_i \leq 2, \qquad i \in N, \tag{2.2}$$

$$z_i \in \{0,1\}, \qquad i \in N. \tag{2.3}$$

In this model, ξ_i determines the penalization if the misclassified object i is within the strip defined by the two parallel hyperplanes ($wx + b = 1$ and $wx + b = -1$) and z_i determines whether i is a misclassified object outside this strip or not. This model can be linearized by replacing the set of conditional constraints (2.1) with big M constraints as follows,

$$\begin{aligned} (\text{RL-}\ell_p\text{-M}) \quad \min \quad & \frac{1}{p}(||w||_p)^p + C\left(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n} z_i\right) \\ \text{s.t.} \quad & (2.2), (2.3), \\ & y_i\left(\sum_{k=1}^{d} w_k x_{ik} + b\right) \geq 1 - \xi_i - M_i z_i, \quad i \in N, \end{aligned} \tag{2.4}$$

where M_i is a big enough constant, for $i \in N$. In the following, we give a result that establishes a relationship between the ξ-variables and the z-variables.

Proposition 2.1.

i) An optimal solution of (RL-ℓ_p), with $p \geq 1$, (w^*, b^*, ξ^*, z^*), satisfies the following condition:

$$\xi_i^* z_i^* = 0, \quad i \in N. \tag{2.5}$$

ii) Using M_i, for $i \in N$, so that an optimal solution of (RL-ℓ_p-M) satisfies (2.5) *does not imply that (RL-ℓ_p-M) and (RL-ℓ_p) are equivalent.*

The proof of *i)* can be obtained using a simple contradiction, and for this reason it has been omitted. An example showing *ii)* is given in the following for $p = 1$ and $p = 2$.

Example 2.1. *Let N be a set of individuals partitioned into two classes, where the associated pair (x_i, y_i) for $i \in N$ is:*

$$\{((-2,1),1), ((-1,-1),1), ((-5,-3),-1), ((1,3),-1), ((1,0),-1)\}$$

and $C = 10$. The optimal solution of (RL-ℓ_1-M) is 23, a valid value for M is 41, and the optimal solution is $w^* = (-3,0)$, $b^* = 2$, $\xi^* = (0,0,0,0,0)$, and $z^* = (0,0,1,0,0)$, (the valid value for the big M parameter was computed following the strategies described in Section 2.3). Nevertheless, considering $M = 5$ in model (RL-ℓ_1-M), the optimal value is 25.8333; where $w^* = (-2.5,0)$, $b^* = 1.5$, $\xi^* = (0, 0.3333, 0, 0, 0)$, and $z^* = (0,0,1,0,0)$; i.e., *condition* (2.5) *is still fulfilled for* $i \in N$*, but its optimal objective value differs. Similarly, the optimal solution of (RL-*ℓ_2*-M) is 23.6, a valid value for M is 14, and the optimal solution is* $w^* = (-2.4, -1.2), b^* = 1.4, \xi = (0,0,0,0,0)$*, and* $z = (0,0,1,0,0)$*, (the valid value for the big M parameter was computed following the strategies described in Section 2.4). However, establishing* $M = 5$ *in formulation (RL-*ℓ_2*-M), the optimal value is 25.9333; where* $w^* = (-2.2, -0.6), b^* = 1.2, \xi^* = (0, 0.3333, 0, 0, 0)$*, and* $z = (0,0,1,0,0)$.

As a consequence of the previous result, the linearized version of condition (2.5) is given by

$$\xi_i \leq 2(1 - z_i), \quad i \in N. \tag{2.6}$$

This set of constraints will be used to strengthen the (RL-ℓ_p-M) formulation. Henceforth, we will refer to (RL-ℓ_p-M)+(2.6) as (RL-ℓ_p-M), unless it is stated otherwise.

Observe that in the (RL-ℓ_p-M) model, the choice of an appropriate value for M_i, for $i \in N$ is essential when providing efficient solution approaches. Note that M_i should be big enough for the equivalence between (RL-ℓ_p) and (RL-ℓ_p-M), but it should also be as small as possible so as to improve the linear relaxation and the computational time needed to solve it. The following proposition provides valid values for the big M parameters considering a general ℓ_p-norm, with $p \geq 1$.

Proposition 2.2. *For a given* $p \geq 1$*, the problems (RL-*ℓ_p*) and (RL-*ℓ_p*-M) are equivalent if*

$$M_i \geq \left(\max_{j \in N} \{ \|x_i - x_j\|_{\bar{q}} : y_i = y_j \} \right) \|w\|_{\bar{p}}, \quad \textit{for } i \in N,$$

with $\bar{p}, \bar{q} \geq 1$ *such that* $\frac{1}{\bar{p}} + \frac{1}{\bar{q}} = 1$ *(i.e.* $\|\cdot\|_{\bar{p}}$ *and* $\|\cdot\|_{\bar{q}}$ *are dual norms).*

Proof:

Taking into account set of constraints (2.4), it holds that:

$$M_i z_i \geq -y_i \left(\sum_{k=1}^{d} w_k x_{ik} + b \right) + 1 - \xi_i, \quad \text{for } i \in N.$$

According to Proposition 2.1, an optimal solution of (RL-ℓ_p) satisfies condition (2.5). Consequently, a valid value for M_i in the formulation (RL-ℓ_p-M) would be one satisfying:

$$M_i \geq -y_i \left(\sum_{k=1}^{d} w_k x_{ik} + b \right) + 1, \quad \text{for } i \in N.$$

Hence, a value of M_i satisfying:

$$M_i \geq \left| -y_i \left(\sum_{k=1}^{d} w_k x_{ik} + b \right) + 1 \right| = \left| y_i \left(\sum_{k=1}^{d} w_k x_{ik} + b \right) - 1 \right|, \tag{2.7}$$

would also be valid. On the other hand, the distance between x_i and the hyperplane $H(y_i w, y_i b - 1) := \{x : y_i(w \cdot x + b) - 1 = 0\}$, for $i \in N$, using the $\ell_{\bar{q}}$-norm for $\bar{q} \geq 1$ is (see Plastria and Carrizosa (2001)):

$$d_{\ell_{\bar{q}}}(x_i, H(y_i w, y_i b - 1)) = \frac{|y_i(w \cdot x_i + b) - 1|}{\|w\|_{\bar{p}}}. \tag{2.8}$$

Thus, taking into account expressions (2.7) and (2.8), the (RL-ℓ_p-M) and (RL-ℓ_p) problems will be equivalent if the following inequality holds:

$$M_i \geq d_{\ell_{\bar{q}}}(x_i, H(y_i w, y_i b - 1)) \|w\|_{\bar{p}}, \quad \text{for } i \in N. \tag{2.9}$$

Observe that the previous expression represents the distance using the $\ell_{\bar{q}}$-norm from x_i to $H(y_i w, y_i b - 1)$, i.e., the supporting hyperplane for each class. Consequently, $d_{\ell_{\bar{q}}}(x_i, H(y_i w, y_i b - 1))$ is, at most, the maximum distance between two individuals of the same class, i.e.,

$$d_{\ell_{\bar{q}}}(x_i, H(y_i w, y_i b - 1)) \leq \max_{j \in N} \{\|x_i - x_j\|_{\bar{q}} : y_i = y_j\}.$$

Thus, if M_i satisfies

$$M_i \geq \left(\max_{j \in N} \{\|x_i - x_j\|_{\bar{q}} : y_i = y_j\} \right) \|w\|_{\bar{p}}, \quad \text{for } i \in N,$$

both problems will be equivalent. □

Strategies for obtaining tighter values than the ones provided by the previous expressions will be presented in the next sections. Particularly, in Section 2.3 we will consider the ramp loss model with ℓ_1-norm while in Section 2.4, the attention will be focused on the ℓ_2-norm.

2.3 Strategies for the ℓ_1-norm case

The objective of this section is to present the model for the ℓ_1-norm case and to improve the values of big M parameters appearing in the model. As a result, two algorithms are derived. They are based on tightening bounds of w-variables in the model and using these bounds to provide tighter values of big M parameter.

A formulation of the SVM with ramp loss using the ℓ_1-norm is obtained by decomposing the unrestricted variables w_k as the difference of two non-negative variables w_k^+ and w_k^- for $k \in D$, where D is the set $\{1, \dots, d\}$, (see Labbé et al. (2019)). In this reformulation, $w_k = w_k^+ - w_k^-$, where $w_k^+, w_k^- \geq 0$, for $k \in D$. Thusly, $|w_k| = w_k^+ + w_k^-$ in any optimal solution, since $w_k^+ + w_k^-$, for $k \in D$ is part of the objective function to be minimized. This means that, at most, only one of the two variables for any $k \in D$ is non-zero in an optimal solution. The result is the following formulation:

$$\begin{aligned}
(\text{RL-}\ell_1)\ \min\ & \sum_{k=1}^{d}(w_k^+ + w_k^-) + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right) \\
\text{s.t.}\ & (2.2), (2.3), \\
& \text{if } z_i = 0,\ \ y_i\left(\sum_{k=1}^{d}(w_k^+ - w_k^-)x_{ik} + b\right) \geq 1 - \xi_i, \quad i \in N, \quad (2.10) \\
& w_k^+ \geq 0, w_k^- \geq 0, \quad k \in D. \quad (2.11)
\end{aligned}$$

This model can be linearized by replacing set of conditional constraints (2.10) with big M constraints, where M_i is a big enough constant, for $i \in N$.

$$\begin{aligned}
(\text{RL-}\ell_1\text{-M})\ \min\ & \sum_{k=1}^{d}(w_k^+ + w_k^-) + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right) \\
\text{s.t.}\ & (2.2), (2.3), (2.11), \\
& y_i\left(\sum_{k=1}^{d}(w_k^+ - w_k^-)x_{ik} + b\right) \geq 1 - \xi_i - M_i z_i, \quad i \in N. \quad (2.12)
\end{aligned}$$

2.3.1 Tightening bounds of w-variables

In the previous section, Proposition 2.2 provided valid values M_i, for $i \in N$, depending on $\|w\|_{\bar{p}}$. Below, we develop two strategies to obtain these values considering $\bar{p} = 1$ and $\bar{p} = \infty$, respectively. To do so, we will obtain bounds for the w-variables that will be included in the model to strengthen the formulation.

2.3.1 Initial big M parameters

Note that by Proposition 2.2, using $\bar{q} = 1$ and $\bar{p} = \infty$, we can consider $M_i = \text{dist}_i^1 \cdot \text{UB}_{\text{RL-}\ell_1}$ as the initial value of M_i, for $i \in N$, where $\text{dist}_i^1 = \max\limits_{j \in N} \{\|x_i - x_j\|_1 : y_i = y_j\}$ and $\text{UB}_{\text{RL-}\ell_1}$ is an upper bound of (RL-ℓ_1-M). Since $\|w\|_\infty \leq \|w\|_1$ and $\text{UB}_{\text{RL-}\ell_1}$ is an upper bound of $\|w\|_1$, we have that $\|w\|_\infty \leq \|w\|_1 \leq \text{UB}_{\text{RL-}\ell_1}$. Furthermore, an upper bound of (RL-ℓ_1-M) can be easily obtained from a feasible solution $(\tilde{w}^+, \tilde{w}^-, \tilde{b}, \tilde{\xi}, \tilde{z})$, built from the optimal solution of (SVM-ℓ_1), $(w^{\text{SVM}}, b^{\text{SVM}}, \xi^{\text{SVM}})$, establishing that $b = b^{\text{SVM}}$,

$$\tilde{w}_k^+ = \begin{cases} w_k^{\text{SVM}}, & \text{if } w_k^{\text{SVM}} \geq 0, \\ 0, & \text{otherwise,} \end{cases} \text{for } k \in D,$$

$$\tilde{w}_k^- = \begin{cases} -w_k^{\text{SVM}}, & \text{if } w_k^{\text{SVM}} \leq 0, \\ 0, & \text{otherwise,} \end{cases} \text{for } k \in D,$$

$$\tilde{\xi}_i = \begin{cases} \xi_i^{\text{SVM}}, & \text{if } \xi_i^{\text{SVM}} \leq 2, \\ 0, & \text{otherwise,} \end{cases} \text{for } i \in N,$$

$$\tilde{z}_i = \begin{cases} 0, & \text{if } \xi_i^{\text{SVM}} \leq 2, \\ 1, & \text{otherwise,} \end{cases} \text{for } i \in N.$$

This upper bound could be improved using the information given by $\tilde{z}$ values to obtain the following model:

$$\begin{array}{lll} (\overline{\text{SVM-}\ell_1})_{\tilde{z}} & \min & \displaystyle\sum_{k=1}^{d}(w_k^+ + w_k^-) + C\left(\sum_{i \in N:\tilde{z}_i=0} \xi_i\right) \\ & \text{s.t.} & (2.11), \\ & & y_i\left(\displaystyle\sum_{k=1}^{d}(w_k^+ - w_k^-)x_{ik} + b\right) \geq 1 - \xi_i, \quad i \in N : \tilde{z}_i = 0, \\ & & 0 \leq \xi_i \leq 2, \qquad\qquad\qquad\qquad i \in N : \tilde{z}_i = 0. \end{array}$$

The solution of this linear problem $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi})$ together with $\tilde{z}$ values constitute a feasible solution for (RL-ℓ_1-M) that provides a better upper bound, $\text{UB}_{\text{RL-}\ell_1}$. With this new upper bound, the value of M_i can be updated.

In Variant 1 of Algorithm 2.1, we give a pseudocode with a strategy to obtain a valid value of M_i, for $i \in N$. The main purpose of this algorithm is to solve a set of linear problems in order to compute a tightened upper bound of $\|w\|_\infty$ (Steps 5-7). Consequentially, we obtain bounds for the w-variables and add them to the problem improving the formulation (Step 8).

On the other hand, if we use $\bar{q} = \infty$ and $\bar{p} = 1$, the initial value of M_i for $i \in N$ would be $\left(\max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\}\right) \text{UB}_{\text{RL-}\ell_1}$, because $\text{UB}_{\text{RL-}\ell_1}$ is an upper bound of $\|w\|_1$. In order to improve this bound, we propose a strategy which is summarized in Variant 2 of Algorithm 2.1, the first step of which is to compute the initial values for the big M parameters. Next, we compute a tightened bound of $\|w\|_1$, thus solving a linear programming problem (Step 10 of Algorithm 2.1).

Observe that in both variants of Algorithm 2.1, the initial values of the big M parameters are computed as $M_i = \text{dist}_i^\infty \cdot \text{UB}_{\text{RL-}\ell_1}$, where $\text{dist}_i^\infty = \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\}$, because $\text{dist}_i^\infty \leq \text{dist}_i^1$ as well as $\text{UB}_{\text{RL-}\ell_1}$ is an upper bound of $\|w\|_\infty$ and $\|w\|_1$.

Algorithm 2.1: Variant 1 and 2. Computation of tightened bounds of w-variables.

```
   Data: Training sample composed by a set of $n$ individuals with $d$ features.
   Result: Update values of $M_i$ and bounds for $w_k^+ + w_k^-$.
1  Solve the problem (SVM-$\ell_1$). From its optimal solution, build a feasible solution of (RL-$\ell_1$-M), $(\tilde{w}^+, \tilde{w}^-, \tilde{b}, \tilde{\xi}, \tilde{z})$. Solve $(\overline{\text{SVM-}\ell_1})_{\tilde{z}}$ and build an improved feasible solution. Update the upper bound $\text{UB}_{\text{RL-}\ell_1}$.
2  for $i \in N$ do
3      $\text{dist}_i^\infty = \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\}, \quad M_i = \text{dist}_i^\infty \cdot \text{UB}_{\text{RL-}\ell_1}$.
4  case Variant 1 do
5      for $k_0 \in D$ do
6          Solve the following linear programming problem:
               $\max w_{k_0}^+ + w_{k_0}^-$
               s.t. $(2.2), (2.6), (2.11), (2.12)$,
               $\sum_{k=1}^d (w_k^+ + w_k^-) + C\left(\sum_{i=1}^n \xi_i + 2\sum_{i=1}^n z_i\right) \leq \text{UB}_{\text{RL-}\ell_1}$,      (2.13)
               $0 \leq z_i \leq 1, \quad i \in N.$      (2.14)
7          Let $\text{UB}_{w_{k_0}}$ be the optimal objective value of the above problem.
8      Update $M_i = \min\left\{\text{dist}_i^1 \cdot \max_{k \in D} \{\text{UB}_{w_k}\}, \text{dist}_i^\infty \cdot \text{UB}_{\text{RL-}\ell_1}\right\}$ and add the obtained bounds to the problem (RL-$\ell_1$-M) including the following set of constraints:
               $w_k^+ + w_k^- \leq \text{UB}_{w_k}, \quad k \in D.$      (2.15)
9  case Variant 2 do
10     Solve the following linear programming problem:
               $\max \sum_{k=1}^d w_k^+ + w_k^-$
               s.t. $(2.2), (2.6), (2.11) - (2.14)$.
       Let $\text{UB}_w$ be the optimal objective value of the above problem.
11     Update $M_i = \text{dist}_i^\infty \cdot \text{UB}_w$ and add the obtained bounds to the problem (RL-$\ell_1$-M) including the following set of constraints:
               $w_k^+ + w_k^- \leq \text{UB}_w, \quad k \in D.$      (2.16)
```

A detailed comparison between the performance of both strategies is carried out in Section 2.5. In general, the resolution time of this algorithm will be lower when using Variant 2 because it solves fewer problems in each iteration.

2.3.2 A Lagrangian relaxation based procedure

Similarly to the previous strategy, this subsection aims to tighten the bounds for the w-variables. Up to this point, the bounds on the w-variables were computed by solving linear problems and expressed as valid inequalities, i.e., constraints (2.15) and (2.16), using Variants 1 and 2 respectively. In the following, we will refer to set of constraints (2.15), but the same theoretical results can be obtained using Variant 2, i.e., including (2.16) instead of (2.15). In this subsection, we propose a procedure that is based on the Lagragian relaxation of (RL-ℓ_1-M). Before going into detail about this strategy, note that the linear relaxation of (RL-ℓ_1-M) is

$$\begin{array}{lll}
(\text{LP-RL-}\ell_1) & \min & \sum_{k=1}^{d}(w_k^+ + w_k^-) + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right)\\
 & \text{s.t.} & (2.2), (2.6), (2.11), (2.12), (2.14), (2.15).
\end{array}$$

We will build two new models based on the previous one, seeing as this will be necessary for the next technical result that will be used in Theorem 2.1 to provide bounds on the w-variables using Lagrangian relaxations. Given a value of $\tilde{w}_{k_0}^+ > 0$, for $k_0 \in D$, we obtain the following equivalent model to (LP-RL-ℓ_1) by making the changes of variables: $\bar{w}_{k_0}^+ = w_{k_0}^+ - \tilde{w}_{k_0}^+$, $\bar{w}_k^+ = w_k^+$, for $k \in D\setminus\{k_0\}$, and $\bar{w}_k^- = w_k^-$, for $k \in D$. The resulting formulation is named $\overline{(\text{LP-RL-}\ell_1)}_{k_0}^+$:

$$\min \sum_{k=1}^{d}(\bar{w}_k^+ + \bar{w}_k^-) + C\left(\sum_{i=1}^{n}\bar{\xi}_i + 2\sum_{i=1}^{n}\bar{z}_i\right) + \tilde{w}_{k_0}^+$$

$$\begin{array}{lllr}
\text{s.t.} & y_i\left(\sum_{k=1}^{d}(\bar{w}_k^+ - \bar{w}_k^-)x_{ik} + \bar{b}\right) \geq 1 - \bar{\xi}_i - y_i\tilde{w}_{k_0}^+ x_{ik_0} - M_i\bar{z}_i, & i \in N, & (2.17)\\
 & \bar{\xi}_i \leq 2(1-\bar{z}_i), & i \in N, & (2.18)\\
 & \bar{w}_k^+ + \bar{w}_k^- \leq \text{UB}_{w_k}, & k \in D, & (2.19)\\
 & 0 \leq \bar{\xi}_i \leq 2, & i \in N, & (2.20)\\
 & 0 \leq \bar{z}_i \leq 1, & i \in N, & (2.21)\\
 & \bar{w}_k^+ \geq 0, & k \in D\setminus\{k_0\}, & (2.22)\\
 & \bar{w}_{k_0}^+ \geq -\tilde{w}_{k_0}^+, & & (2.23)\\
 & \bar{w}_k^- \geq 0, & k \in D. & (2.24)
\end{array}$$

Similarly, given a value of $\tilde{w}_{k_0}^- > 0$, for $k_0 \in D$, the following equivalent model to (LP-RL-ℓ_1) is obtained by changing of variables $\bar{w}_{k_0}^- = w_{k_0}^- - \tilde{w}_{k_0}^-$,

$\bar{w}_k^+ = w_k^+$, for $k \in D$, and $\bar{w}_k^- = w_k^-$, for $k \in D \setminus \{k_0\}$, named $(\overline{\text{LP-RL-}\ell_1})_{k_0}^-$:

$$\begin{aligned}
\min\ & \sum_{k=1}^{d}(\bar{w}_k^+ + \bar{w}_k^-) + C\left(\sum_{i=1}^{n}\bar{\xi}_i + 2\sum_{i=1}^{n}\bar{z}_i\right) + \tilde{w}_{k_0}^- \\
\text{s.t. }\ & (2.18) - (2.21), \\
& y_i\left(\sum_{k=1}^{d}(\bar{w}_k^+ - \bar{w}_k^-)x_{ik} + \bar{b}\right) \geq 1 - \bar{\xi}_i + y_i\tilde{w}_{k_0}^- x_{ik_0} - M_i\bar{z}_i, \quad i \in N, && (2.25)\\
& \bar{w}_k^+ \geq 0, \quad k \in D, && (2.26)\\
& \bar{w}_k^- \geq 0, \quad k \in D\setminus\{k_0\}, && (2.27)\\
& \bar{w}_{k_0}^- \geq -\tilde{w}_{k_0}^-. && (2.28)
\end{aligned}$$

Lemma 2.1. *The following statements hold:*

i) Let $(\bar{w}^{+^*}, \bar{w}^{-^*}, \bar{b}^*, \bar{\xi}^*, \bar{z}^*)$ be an optimal solution of $(\overline{LP\text{-}RL\text{-}\ell_1})_{k_0}^+$ with $\bar{w}_{k_0}^{+^*} = 0$ for $k_0 \in D$, with $Z_{k_0}^+$ being its objective value, and $\bar{\alpha}^+$ a vector of optimal values for the dual variables associated with constraints (2.17)*. If $(\bar{w}^{+'}, \bar{w}^{-'}, \bar{b}', \bar{\xi}', \bar{z}')$ is an optimal solution of $(\overline{LP\text{-}RL\text{-}\ell_1})_{k_0}^+$ restricting $\bar{w}_{k_0}^+ = \hat{w}_{k_0}^+$, where $\hat{w}_{k_0}^+ > -\tilde{w}_{k_0}^+$, and $Z_{\hat{k}_0}^+$ its objective value, then:*

$$Z_{k_0}^+ + \hat{w}_{k_0}^+\left(1 - \sum_{i=1}^{n}\bar{\alpha}_i^+ y_i x_{ik_0}\right) \leq Z_{\hat{k}_0}^+. \tag{2.29}$$

ii) Let $(\bar{w}^{+^*}, \bar{w}^{-^*}, \bar{b}^*, \bar{\xi}^*, \bar{z}^*)$ be an optimal solution of $(\overline{LP\text{-}RL\text{-}\ell_1})_{k_0}^-$ with $\bar{w}_{k_0}^{-^*} = 0$, for $k_0 \in D$, with $Z_{k_0}^-$ being its objective value, and $\bar{\alpha}^-$ a vector of optimal values for the dual variables associated with constraints (2.25)*. If $(\bar{w}^{+'}, \bar{w}^{-'}, \bar{b}', \bar{\xi}', \bar{z}')$ is an optimal solution of $(\overline{LP\text{-}RL\text{-}\ell_1})_{k_0}^-$ restricting $\bar{w}_{k_0}^- = \hat{w}_{k_0}^-$, where $\hat{w}_{k_0}^- > -\tilde{w}_{k_0}^-$, and $Z_{\hat{k}_0}^-$ its objective value, then:*

$$Z_{k_0}^- + \hat{w}_{k_0}^-\left(1 + \sum_{i=1}^{n}\bar{\alpha}_i^- y_i x_{ik_0}\right) \leq Z_{\hat{k}_0}^-. \tag{2.30}$$

Proof:

i) Let $\bar{\alpha}^+$ be the vector of optimal values for the dual variables associated with family of constraints (2.17) for $(\overline{\text{LP-RL-}\ell_1})_{k_0}^+$. By the complementary slackness conditions, it holds that:

$$\begin{aligned}
Z_{k_0}^+ =& \sum_{k=1}^{d}\left(\bar{w}_k^{+^*} + \bar{w}_k^{-^*}\right) + C\left(\sum_{i=1}^{n}\bar{\xi}_i^* + 2\sum_{i=1}^{n}\bar{z}_i^*\right) + \tilde{w}_{k_0}^+ \\
&+ \sum_{i=1}^{n}\bar{\alpha}_i^+\left(1 - \bar{\xi}_i^* - y_i\tilde{w}_{k_0}^+ x_{ik_0} - M_i\bar{z}_i^* - y_i\sum_{k=1}^{d}\left(\bar{w}_k^{+^*} - \bar{w}_k^{-^*}\right)x_{ik} - y_i\bar{b}^*\right).
\end{aligned}$$

Since $\bar{w}^{+^*}_{k_0} = 0$, this variable can be removed from the summations, giving:

$$\begin{aligned} Z^+_{k_0} = &\sum_{k=1,k\neq k_0}^{d} \bar{w}^{+^*}_k + \sum_{k=1}^{d} \bar{w}^{-^*}_k + C\left(\sum_{i=1}^{n} \bar{\xi}^*_i + 2\sum_{i=1}^{n} \bar{z}^*_i\right) + \tilde{w}^+_{k_0} \\ &+ \sum_{i=1}^{n} \bar{\alpha}^+_i \left(1 - \bar{\xi}^*_i - y_i \tilde{w}^+_{k_0} x_{ik_0} - M_i \bar{z}^*_i - y_i \left(\sum_{k=1,k\neq k_0}^{d} \bar{w}^{+^*}_k x_{ik} - \sum_{k=1}^{d} \bar{w}^{-^*}_k x_{ik}\right) - y_i \bar{b}^*\right). \end{aligned} \tag{2.31}$$

Alternatively, the $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$ model with additional constraint $\bar{w}^+_{k_0} = \hat{w}^+_{k_0}$, and in which the family of constraints (2.17) has been dualized, with $\alpha_i \geq 0$ for any $i \in N$, is the following:

$$\begin{aligned} \min \quad & \sum_{k=1}^{d} (\bar{w}^+_k + \bar{w}^-_k) + C\left(\sum_{i=1}^{n} \bar{\xi}_i + 2\sum_{i=1}^{n} \bar{z}_i\right) + \tilde{w}^+_{k_0} \\ & + \sum_{i=1}^{n} \alpha_i \left(1 - \bar{\xi}_i - y_i \tilde{w}^+_{k_0} x_{ik_0} - M_i \bar{z}_i - y_i \sum_{k=1}^{d} (\bar{w}^+_k - \bar{w}^-_k) x_{ik} - y_i \bar{b}_i\right) \\ \text{s.t.} \quad & (2.18) - (2.22), (2.24), \\ & \bar{w}^+_{k_0} = \hat{w}^+_{k_0}. \end{aligned}$$

Thus, by extracting the coefficients of $\hat{w}^+_{k_0}$ from the summations, this problem can be rewritten as follows, named $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$:

$$\begin{aligned} \min \; & \sum_{k=1,k\neq k_0}^{d} \bar{w}^+_k + \sum_{k=1}^{d} \bar{w}^-_k + C\left(\sum_{i=1}^{n} \bar{\xi}_i + 2\sum_{i=1}^{n} \bar{z}_i\right) + \hat{w}^+_{k_0}\left(1 - \sum_{i=1}^{n} \alpha_i y_i x_{ik_0}\right) + \\ & \sum_{i=1}^{n} \alpha_i \left(1 - \bar{\xi}_i - y_i \tilde{w}^+_{k_0} x_{ik_0} - M_i \bar{z}_i - y_i \left(\sum_{k=1,k\neq k_0}^{d} \bar{w}^+_k x_{ik} - \sum_{k=1}^{d} \bar{w}^-_k x_{ik}\right) - y_i \bar{b}_i\right) \\ & + \tilde{w}^+_{k_0} \\ \text{s.t.} \; & (2.18) - (2.22), (2.24). \end{aligned}$$

Observe that $(\bar{w}^{+^*}, \bar{w}^{-^*}, \bar{b}^*, \bar{\xi}^*, \bar{z}^*)$, an optimal solution of $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$, is feasible for the problem above, since all constraints of $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$ are included in the former. Moreover, any feasible solution of $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$, taking $\bar{w}^+_{k_0} = 0$, is feasible for $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$ where family of constraints (2.17) has been dualized. Therefore, for $\alpha = \bar{\alpha}^+$, using (2.31), the optimal objective value of $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$ is $Z^+_{k_0} + \hat{w}^+_{k_0}\left(1 - \sum_{i=1}^{n} \bar{\alpha}^+_i y_i x_{ik_0}\right)$. Since $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$ is a Lagrangian relaxation of $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$ with the additional constraint $\bar{w}^+_{k_0} = \hat{w}^+_{k_0}$, then the optimal objective value of $(\overline{\text{Lg-RL-}\ell_1})^+_{k_0}$ is a lower bound of $Z^+_{\hat{k}_0}$.

Result *ii)* is proven by following the same argument as *i)*.

□

By applying the lemma above, we obtain the following theorem which provides the bounds of the w-variables.

Theorem 2.1. *Under the hypothesis of Lemma 2.1 we obtain the following bounds of the w-variables for the (RL-ℓ_1-M) problem:*

- $w^+_{k_0} \leq UB_{w^+_k} := \dfrac{UB_{RL\text{-}\ell_1} - Z^+_{k_0}}{1 - \sum_{i=1}^n \bar{\alpha}^+_i y_i x_{ik_0}} + \tilde{w}^+_{k_0},$
- $w^-_{k_0} \leq UB_{w^-_k} := \dfrac{UB_{RL\text{-}\ell_1} - Z^-_{k_0}}{1 + \sum_{i=1}^n \bar{\alpha}^-_i y_i x_{ik_0}} + \tilde{w}^-_{k_0},$

where $UB_{RL\text{-}\ell_1}$ is an upper bound of (RL-ℓ_1-M).

Proof:

An equivalent model $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$ $\left((\overline{\text{LP-RL-}\ell_1})^-_{k_0}\right)$ can be built from an optimal solution of (LP-RL-ℓ_1), i.e., $({w^+}^*, {w^-}^*, b^*, \xi^*, z^*)$ that satisfies that ${w^+_{k_0}}^* = \tilde{w}^+_{k_0}$ $\left({w^-_{k_0}}^* = \tilde{w}^-_{k_0}\right)$. In this situation, both models have the same optimal objective value. Since model (LP-RL-ℓ_1) is the linear relaxation of model (RL-ℓ_1-M), an upper bound of the latter would be an upper bound of former. Thusly, if the objective value of an optimal solution of $(\overline{\text{LP-RL-}\ell_1})^+_{k_0}$ restricting $\bar{w}^+_{k_0} = \hat{w}^+_{k_0}$ $\left((\overline{\text{LP-RL-}\ell_1})^-_{k_0}\right.$ restricting $\left.\bar{w}^-_{k_0} = \hat{w}^-_{k_0}\right)$ is bigger than $\text{UB}_{\text{RL-}\ell_1}$, the value $\bar{w}^+_{k_0} = \hat{w}^+_{k_0}$ $\left(\bar{w}^-_{k_0} = \hat{w}^-_{k_0}\right)$ can be discarded as an optimal solution of (RL-ℓ_1-M). This is because any solution with this value will provide a solution with an objective value that is worse than $\text{UB}_{\text{RL-}\ell_1}$. Therefore, we can restrict ourselves to the values of $\hat{w}^+_{k_0}$ $\left(\hat{w}^-_{k_0}\right)$ in such a way that $Z^+_{\hat{k}_0} \leq \text{UB}_{\text{RL-}\ell_1}$ $\left(Z^-_{\hat{k}_0} \leq \text{UB}_{\text{RL-}\ell_1}\right)$. Thus, according to Lemma 2.1, $\bar{w}^+_{k_0}$ $\left(\bar{w}^-_{k_0}\right)$ satisfies:

$$\bar{w}^+_{k_0} \leq \frac{\text{UB}_{\text{RL-}\ell_1} - Z^+_{k_0}}{1 - \sum_{i=1}^n \bar{\alpha}^+_i y_i x_{ik_0}}, \quad \bar{w}^-_{k_0} \leq \frac{\text{UB}_{\text{RL-}\ell_1} - Z^+_{k_0}}{1 + \sum_{i=1}^n \bar{\alpha}^+_i y_i x_{ik_0}}.$$

We therefore obtain the upper bounds for the w-variables of problem (RL-ℓ_1-M) by undoing the changes of the variables are obtained. $\square$

The resulting bounds will be included in the formulation of the problem as the following set of constraints:

$$w^+_k \leq \text{UB}_{w^+_k}, \quad k \in D, \tag{2.32}$$

$$w^-_k \leq \text{UB}_{w^-_k}, \quad k \in D. \tag{2.33}$$

These bounds for the w-variables will be used in the next section to obtain tightened values of big M parameters in the models.

2.3.2 Tightening values of big M parameters

The previous strategies were based on tightening the bounds for the w-variables. In this subsection, we will take advantage of these bounds to obtain tightened bounds of the big M parameters by solving linear programming problems.

First, we present a result which, for a given $i \in N$, provides a valid value of the big M parameter when solving a linear problem. Note that in this subsection we will refer to set of constraints (2.15), but the same theoretical results can be obtained using Variant 2 of Algorithm 2.1, i.e., including (2.16) instead.

Proposition 2.3. *If M_i, for all $i \in N$, is greater than or equal to the optimal objective value of the following linear problem, provided it is not unbounded:*

$$\begin{array}{lll}(UB_{M_i}) & \max & 1-\xi_i - y_i\left(\displaystyle\sum_{k=1}^{d}(w_k^+ - w_k^-)x_{ik} + b\right)\\ & s.t. & (2.2),(2.6),(2.11)-(2.15),(2.32),(2.33),\end{array}$$

then problems (RL-ℓ_1) and (RL-ℓ_1-M) are equivalent.

Proof:

It holds directly from (2.12) taking the maximum. □

In datasets with a large number of individuals, solving a linear problem for each $i \in N$ would be an inefficient computation. Thus, we now prove a result that allows us to obtain a valid value for the big M parameter associated with each individual of each class, thereby solving two linear problems.

Proposition 2.4. *Problems (RL-ℓ_1) and (RL-ℓ_1-M) are equivalent if the following statements hold:*

i) For all $i \in N$, when $y_i = +1$, M_i is greater than or equal to the optimal objective value of the following linear problem, provided it is not unbounded:

$$\begin{array}{lll}(UB_{M_+}) & \max & 1-\left(\displaystyle\sum_{k=1}^{d} w_k^+ \underline{x}_{+k} - \sum_{k=1}^{d} w_k^- \bar{x}_{+k} + b\right)\\ & s.t. & (2.2),(2.6),(2.11)-(2.15),(2.32),(2.33),\end{array}$$

where $\underline{x}_{+k} = \min\limits_{i\in N.}\{x_{ik} : y_i = +1\}$ and $\bar{x}_{+k} = \max\limits_{i\in N.}\{x_{ik} : y_i = +1\}$, for $k \in D$.

ii) For all $i \in N$, when $y_i = -1$, M_i is greater than or equal to the optimal objective value of the following linear problem, provided it is not unbounded:

$$\begin{array}{lll}(UB_{M_-}) & \max & 1+\left(\displaystyle\sum_{k=1}^{d} w_k^+ \bar{x}_{-k} - \sum_{k=1}^{d} w_k^- \underline{x}_{-k} + b\right)\\ & s.t. & (2.2),(2.6),(2.11)-(2.15),(2.32),(2.33),\end{array}$$

where $\bar{x}_{-k} = \max_{i \in N.} \{x_{ik} : y_i = -1\}$ *and* $\underline{x}_{-k} = \min_{i \in N} \{x_{ik} : y_i = -1\}$, *for* $k \in D$.

Proof:

For each $k \in D$ and $i \in N$, when $y_i = +1$, the following inequalities are satisfied: $w_k^+ x_{ik} \geq w_k^+ \underline{x}_{+k}$ and $-w_k^- x_{ik} \geq -w_k^- \bar{x}_{+k}$. Taking the summation in k, we have:

$$1 - \left(\sum_{k=1}^{d} (w_k^+ - w_k^-) x_{ik} + b \right) \leq 1 - \left(\sum_{k=1}^{d} w_k^+ \underline{x}_{+k} - \sum_{k=1}^{d} w_k^- \bar{x}_{+k} + b \right), \quad \text{for } i \in N. \tag{2.34}$$

Since $0 \leq \xi_i \leq 2$, for $i \in N$ when $y_i = +1$, we can conclude that:

$$1 - \xi_i - \left(\sum_{k=1}^{d} (w_k^+ - w_k^-) x_{ik} + b \right) \leq 1 - \left(\sum_{k=1}^{d} w_k^+ \underline{x}_{+k} - \sum_{k=1}^{d} w_k^- \bar{x}_{+k} + b \right), \text{ for } i \in N.$$

Therefore, the optimal solution of problem $\left(\mathrm{UB}_{M_+}\right)$ is an upper bound of problem (UB_{M_i}) for $i \in N$ when $y_i = +1$. Similarly, it can be proven that the optimal solution of problem $\left(\mathrm{UB}_{M_-}\right)$ is an upper bound of problem (UB_{M_i}) for $i \in N$ when $y_i = -1$. Therefore, in accordance with Proposition 2.3 the result holds. $\square$

Proposition 2.4 provides a strategy to obtain valid values for the big M parameters. Obviously these values will be less tightened than the ones resulting from the application of Proposition 2.3, but they can be obtained more quickly. In order to find an equilibrium between computational costs and the tightness of the big M values in a dataset that contains a large number of individuals, we propose the strategy which follows. First, we cluster individuals of the same class in such a way that individuals in the same cluster have certain similarities. This can be done using the "k-median" or the "k-means" algorithms in each class. Let C_+, C_- be the set of clusters of class 1 and class -1 respectively. The number of clusters, i.e., the cardinality of sets C_+ and C_- will be a parameter selected by the modeler and it should be adapted to the data. Then, for each cluster, $c_+ \in C_+$ and $c_- \in C_-$, we apply the proposition below. The proof thereof has been omitted due to its similarity to Proposition 2.4.

Proposition 2.5. *Problems (RL-ℓ_1) and (RL-ℓ_1-M) are equivalent if the following statements hold:*

i) For all $i \in c_+$ with $c_+ \in C_+$, M_i is greater than or equal to the optimal objective value of the following linear problem, provided it is not *unbounded:*

$$(UB_{Mc_+}) \quad \max \quad 1-\left(\sum_{k=1}^{d} w_k^+ \underline{x}_{c+k} - \sum_{k=1}^{d} w_k^- \bar{x}_{c+k} + b\right)$$
$$s.t. \quad (2.2),(2.6),(2.11)-(2.15),(2.32),(2.33),$$

where $\underline{x}_{c+k} = \min_{i\in c_+} \{x_{ik}\}$ *and* $\bar{x}_{c+k} = \max_{i\in c_+} \{x_{ik}\}$, *for* $k \in D$.

ii) For all $i \in c_-$ *with* $c_- \in C_-$, M_i *is greater than or equal to the optimal objective value of the following linear problem, provided it is not unbounded:*

$$(UB_{Mc_-}) \quad \max \quad 1+\left(\sum_{k=1}^{d} w_k^+ \bar{x}_{c-k} - \sum_{k=1}^{d} w_k^- \underline{x}_{c-k} + b\right)$$
$$s.t. \quad (2.2),(2.6),(2.11)-(2.15),(2.32),(2.33),$$

where $\bar{x}_{c-k} = \max_{i\in c_-} \{x_{ik}\}$ *and* $\underline{x}_{c-k} = \min_{i\in c_-} \{x_{ik}\}$, *for* $k \in D$.

Comparatively tighter values of the big M parameters will be obtained by applying Proposition 2.5 instead of Proposition 2.4, but it solves fewer problems than Proposition 2.3. Thus, the best strategy will vary depending on the dataset. We will compare them in Section 2.5.

The results of previous propositions can be summarized in Algorithm 2.2 which is described as a pseudocode. In order to avoid unboundedness in the proposed models, we put forward the strategy that follows. One of the two variants of Algorithm 2.1 is applied in the first step, with the goal of updating the values of the big M parameters and the bounds on the w-variables. In the next steps, upper and lower bounds for the b-variable are obtained and included in the model. Next, the iterative procedure starts with the solution of problem (LP-RL-ℓ_1) in Step 4. A new model is built from an optimal solution of this problem and it is solved by applying the required transformations in such a way that its optimal solution verifies the hypothesis of Theorem 2.1. Hence, Theorem 2.1 is applied to obtain new bounds. After that, the bounds are updated and Steps 2 and 3 are repeated.

The direct procedure is executed next. Said procedure leads to three variants of the algorithm: Variant I (Steps 5-7) applies Proposition 2.3, obtaining valid values of the big M parameters and solving a linear problem for each individual of the data set; Variant II (Steps 8-10) applies Proposition 2.4, in which only two linear problems are solved; finally, Variant III (Steps 11-16) applies Proposition 2.5, in which the number of linear problems solved will be selected by the modeler, depending on the number of clusters (different strategies for subdividing the data into clusters can be followed – the performance of the “k-median” and the “k-means” algorithms are tested in Section 2.5).

Algorithm 2.2: Variant I, II, and III. Computation of tightened values of big M parameters in (RL-ℓ_1-M).

```
   Data: Training sample composed by a set of $n$ individuals with $d$
         features.
   Result: Update values of $M_i$, $\mathrm{UB}_{w_k^+}$, $\mathrm{UB}_{w_k^-}$, and obtain bounds for
           $b$: $\mathrm{LB}_b$ and $\mathrm{UB}_b$.
 1 Apply Variant 1 or 2 of Algorithm 2.1.
 2 Obtain lower $\mathrm{LB}_b$ and upper bounds $\mathrm{UB}_b$ of the $b$-variable solving
     the linear problems that follows. Note that if Variant 2 of
     Algorithm 2.1 was used in Step 1, set of constraints (2.15) should be
     replaced by (2.16).
         $\max/\min \quad b$
         $\text{s.t.} \quad (2.2), (2.6), (2.11) - (2.15).$
 3 Include the following constraint in the formulation of the problem:
         $\mathrm{LB}_b \leq b \leq \mathrm{UB}_b.$                    (2.35)
   do
 4     Solve the required problems to apply Theorem 2.1 and update
         $w$-bounds ($\mathrm{UB}_{w_k^+}$ and $\mathrm{UB}_{w_k^-}$). Repeat Steps 2 and 3 including
         constraints (2.32) and (2.33).
 5     case Variant I do
 6         for $i \in N$ do
 7             Update $M_i$ as the optimal value of the problem
                 $(\mathrm{UB}_{M_i})$+(2.35).
 8     case Variant II do
 9         For $i \in N$, when $y_i = 1$, update $M_i$ as the optimal value of the
             problem $\big(\mathrm{UB}_{M_+}\big)$+(2.35).
10         For $i \in N$, when $y_i = -1$, update $M_i$ as the optimal value of
             the problem $\big(\mathrm{UB}_{M_-}\big)$+(2.35).
11     case Variant III do
12         Cluster the individuals of each class (applying the $k$-median or
             the $k$-means algorithm). Let $C_+, C_-$ be the set of clusters of
             class 1 and class $-1$ respectively.
13         for $c_+ \in C_+$ do
14             Update $M_i$, for $i \in c_+$, as the optimal value of the problem
                 $\big(\mathrm{UB}_{Mc_+}\big)$+(2.35).
15         for $c_- \in C_-$ do
16             Update $M_i$, for $i \in c_-$, as the optimal value of the problem
                 $\big(\mathrm{UB}_{Mc_-}\big)$+(2.35).
17 while an improvement of the bounds is obtained;
```

2.4 Strategies for the ℓ_2-norm case

In this section, we analyze the ramp loss model whilst considering the ℓ_2-norm. We propose strategies to find valid values of the big M parameters, with the objective of enhancing the formulation.

The problem is formulated as follows:

$$\begin{array}{lll}(\text{RL-}\ell_2) & \min & \dfrac{1}{2}\sum_{k=1}^{d} w_k^2 + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right) \\ & \text{s.t.} & (2.1) - (2.3).\end{array}$$

As in the ℓ_1-norm case, the constraints could be linearized by using a big enough constant M_i in (2.1). As a result, we obtain the following quadratic model:

$$\begin{array}{lll}(\text{RL-}\ell_2\text{-M}) & \min & \dfrac{1}{2}\sum_{k=1}^{d} w_k^2 + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right) \\ & \text{s.t.} & (2.2) - (2.4).\end{array}$$

Proposition 2.2 is still valid for obtaining certain values of the big M parameters for formulation (RL-ℓ_2-M). We therefore present a corollary that follows on from Proposition 2.2. It allows us to compute an initial valid value of the big M parameters. These are based on an upper bound on the $\text{UB}_{\text{RL-}\ell_2}$ model. An upper bound can be easily built from the optimal solution of problem (SVM-ℓ_2), denoted by $(\tilde{w}, \tilde{b}, \tilde{\xi}, \tilde{z})$, following a similar procedure to the one described in Section 2.3.1.

Moreover, this upper bound could be improved by using the information given by $\tilde{z}$ values to obtain the following model:

$$\begin{array}{llll}(\overline{\text{SVM-}\ell_2})_{\tilde{z}} & \min & \dfrac{1}{2}\sum_{k=1}^{d} w_k^2 + C\left(\sum_{i\in N:\tilde{z}_i=0}\xi_i\right) & \\ & \text{s.t.} & y_i\left(\sum_{k=1}^{d} w_k x_{ik} + b\right) \geq 1-\xi_i, & i\in N:\tilde{z}_i=0, \\ & & 0\leq \xi_i \leq 2, & i\in N:\tilde{z}_i=0.\end{array}$$

The solution of this quadratic problem $(\bar{w}, \bar{b}, \bar{\xi})$ together with $\tilde{z}$ values constitute a feasible solution for (RL-ℓ_2-M) that provides a better upper bound, $\text{UB}_{\text{RL-}\ell_2}$.

Corollary 2.4.1. *Taking the values for the big M parameters such that:*

$$M_i \geq \left(\max_{j\in N}\{\|x_i - x_j\|_2 : y_i = y_j\}\right)\sqrt{2\,UB_{RL\text{-}\ell_2}}, \quad i\in N,$$

then problems (RL-ℓ_2) and (RL-ℓ_2-M) are equivalent.

Similarly to the case of the ℓ_1-norm, we present a procedure which is based on the Lagragian relaxation to tighten the bounds for the w-variables. In order to do that, we define the model where the integrality condition of the z-variables has been relaxed, i.e.,

$$\begin{array}{lll} (\text{Re-RL-}\ell_2) & \min & \dfrac{1}{2}\sum_{k=1}^{d} w_k^2 + C\left(\sum_{i=1}^{n}\xi_i + 2\sum_{i=1}^{n} z_i\right) \\ & \text{s.t.} & (2.2), (2.6), (2.12), (2.14). \end{array}$$

Hence, we build a formulation which is based on the previous one, making the following change of the variables: given a value of $\tilde{w}_{k_0}$ for some $k_0 \in D$, $\bar{w}_{k_0} = w_{k_0} - \tilde{w}_{k_0}$ and $\bar{w}_k = w_k$, for $k \in D \setminus \{k_0\}$. We therefore obtain the following reformulation of the problem above, named $(\overline{\text{Re-RL-}\ell_2})_{k_0}$:

$$\begin{array}{ll} \min & \dfrac{1}{2}\sum_{k=1}^{d} \bar{w}_k^2 + C\left(\sum_{i=1}^{n}\bar{\xi}_i + 2\sum_{i=1}^{n} \bar{z}_i\right) + \dfrac{1}{2}\tilde{w}_{k_0}^2 + \tilde{w}_{k_0}\bar{w}_{k_0} \\ \text{s.t.} & (2.18), (2.20), (2.21), \\ & y_i\left(\sum_{k=1}^{d}\bar{w}_k x_{ik} + \bar{b}\right) \geq 1 - \bar{\xi}_i - y_i\tilde{w}_{k_0}x_{ik_0} - M_i\bar{z}_i, \qquad i \in N. \end{array} \tag{2.36}$$

Theorem 2.2. *Let $(\bar{w}^*, \bar{b}^*, \bar{\xi}^*, \bar{z}^*)$ be an optimal solution of $(\overline{Re\text{-}RL\text{-}\ell_2})_{k_0}$, with $\bar{w}_{k_0}^* = 0$, Z_{k_0} being its objective value, and $\bar{\alpha}$ a vector of the optimal values for the dual variables associated with constraints* (2.36)*. If $(\bar{w}', \bar{b}', \bar{\xi}', \bar{z}')$ is an optimal solution of $(\overline{Re\text{-}RL\text{-}\ell_2})_{k_0}$ restricting $\bar{w}_{k_0} = \hat{w}_{k_0}$, where $\hat{w}_{k_0}$ is a constant value and $Z_{\hat{k}_0}$ is its objective value, then:*

$$Z_{k_0} + \hat{w}_{k_0}\left(\frac{1}{2}\hat{w}_{k_0} + \tilde{w}_{k_0} - \sum_{i=1}^{n}\bar{\alpha}_i y_i x_{ik_0}\right) \leq Z_{\hat{k}_0}. \tag{2.37}$$

Proof:

Let $\bar{\alpha}$ be the vector of optimal values for the dual variables associated with family of constraints (2.36) of problem $(\overline{\text{Re-RL-}\ell_2})_{k_0}$. By applying Proposition 3.4.2 of Bertsekas (1999), it holds that:

$$\begin{aligned} Z_{k_0} =& \frac{1}{2}\sum_{k=1}^{d}\bar{w}_k^{*2} + C\left(\sum_{i=1}^{n}\bar{\xi}_i^* + 2\sum_{i=1}^{n}\bar{z}_i^*\right) + \frac{1}{2}\tilde{w}_{k_0}^2 + \tilde{w}_{k_0}\bar{w}_{k_0}^* \\ &+ \sum_{i=1}^{n}\bar{\alpha}_i\left(1 - \bar{\xi}_i^* - y_i\tilde{w}_{k_0}x_{ik_0} - M_i\bar{z}_i^* - y_i\sum_{k=1}^{d}\bar{w}_k^* x_{ik} - y_i\bar{b}^*\right). \end{aligned}$$

Additionally, since $\bar{w}^*_{k_0} = 0$, we have:

$$\begin{aligned} Z_{k_0} =& \frac{1}{2} \sum_{k=1,k\neq k_0}^{d} \bar{w}_k^{*2} + C\left(\sum_{i=1}^{n} \bar{\xi}_i^{\,*} + 2\sum_{i=1}^{n} \bar{z}_i^*\right) + \frac{1}{2}\tilde{w}_{k_0}^2 \\ &+ \sum_{i=1}^{n} \bar{\alpha}_i \left(1 - \bar{\xi}_i^{\,*} - y_i\tilde{w}_{k_0}x_{ik_0} - M_i\bar{z}_i^{\,*} - y_i \sum_{k=1,k\neq k_0}^{d} \bar{w}_k^* x_{ik} - y_i\bar{b}^*\right). \qquad (2.38)\end{aligned}$$

On the other hand, formulation $(\overline{\text{Re-RL-}\ell_2})_{k_0}$ with the additional constraint $\bar{w}_{k_0} = \hat{w}_{k_0}$ and where family of constraints (2.36) has been dualized is the following:

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{k=1}^{d} \bar{w}_k^2 + C\left(\sum_{i=1}^{n}\bar{\xi}_i + 2\sum_{i=1}^{n}\bar{z}_i\right) + \frac{1}{2}\tilde{w}_{k_0}^2 + \tilde{w}_{k_0}\bar{w}_{k_0} \\ & + \sum_{i=1}^{n}\alpha_i\left(1 - \bar{\xi}_i - y_i\tilde{w}_{k_0}x_{ik_0} - M_i\bar{z}_i - y_i\sum_{k=1}^{d}\bar{w}_k x_{ik} - y_i\bar{b}\right) \\ \text{s.t.} \quad & (2.18) - (2.21), \\ & \bar{w}_{k_0} = \hat{w}_{k_0}, \end{aligned}$$

where $\alpha_i \geq 0$. Therefore, this problem can be rewritten as follows, named $(\overline{\text{Lg-RL-}\ell_2})_{k_0}$:

$$\begin{aligned} \min\ & \frac{1}{2}\sum_{k=1,k\neq k_0}^{d} \bar{w}_k^{\,2} + C\left(\sum_{i=1}^{n}\bar{\xi}_i + 2\sum_{i=1}^{n}\bar{z}_i\right) + \hat{w}_{k_0}\left(\frac{1}{2}\hat{w}_{k_0} + \tilde{w}_{k_0} - \sum_{i=1}^{n}\alpha_i y_i x_{ik_0}\right) \\ & \frac{1}{2}\tilde{w}_{k_0}^2 + \sum_{i=1}^{n}\alpha_i\left(1 - \bar{\xi}_i - y_i\tilde{w}_{k_0}x_{ik_0} - M_i\bar{z}_i - y_i\sum_{k=1,k\neq k_0}^{d}\bar{w}_k x_{ik} - y_i\bar{b}\right) \\ \text{s.t.}\ \ & (2.18) - (2.21). \end{aligned}$$

Note that $(\bar{w}^*, \bar{b}^*, \bar{\xi}^*, \bar{z}^*)$, an optimal solution of $(\overline{\text{Re-RL-}\ell_2})_{k_0}$, is feasible for the problem above. Additionally, any feasible solution of problem $(\overline{\text{Lg-RL-}\ell_2})_{k_0}$, when taking $\bar{w}_{k_0} = 0$, is feasible for $(\overline{\text{Re-RL-}\ell_2})_{k_0}$ where family of constraints (2.36) has been dualized. Therefore, for $\alpha = \bar{\alpha}$ and when using (2.38), the optimal objective value of the problem above is $Z_{k_0} + \hat{w}_{k_0}\left(\frac{1}{2}\hat{w}_{k_0} + \tilde{w}_{k_0} - \sum_{i=1}^{n}\bar{\alpha}_i y_i x_{ik_0}\right)$. This is a lower bound of the optimal value of $(\overline{\text{Re-RL-}\ell_2})_{k_0}$ with the additional constraint that $\bar{w}_{k_0} = \hat{w}_{k_0}$.

□

Given the theorem above, we obtain Corollary 2.4.2, which provides bounds for the w-variables.

Corollary 2.4.2. *Under the hypothesis of Theorem 2.2, we obtain the following bounds of the w-variables for the (Re-RL-ℓ_2) problem:*

$$w_{k_0} \geq LB_{w_{k_0}} := \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0} - \sqrt{\left(\tilde{w}_{k_0} - \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0}\right)^2 - 2\left(Z_{k_0} - UB_{\text{RL-}\ell_2}\right)},$$

$$w_{k_0} \leq UB_{w_{k_0}} := \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0} + \sqrt{\left(\tilde{w}_{k_0} - \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0}\right)^2 - 2\left(Z_{k_0} - UB_{\text{RL-}\ell_2}\right)},$$

where $UB_{\text{RL-}\ell_2}$ is an upper bound of (RL-ℓ_2-M).

Proof:

An equivalent model $\overline{(\text{Re-RL-}\ell_2)}_{k_0}$ can be built from an optimal solution of (Re-RL-ℓ_2), (w^*, b^*, ξ^*, z^*), which satisfies that ${w_{k_0}}^* = \tilde{w}_{k_0}$. In this situation, both models have the same optimal objective value. Since model (Re-RL-ℓ_2) is the relaxation of model (RL-ℓ_2-M), an upper bound of (RL-ℓ_2-M) would be an upper bound of (Re-RL-ℓ_2). Thus, if the objective value of an optimal solution of $\overline{(\text{Re-RL-}\ell_2)}_{k_0}$ restricting $\bar{w}_{k_0} = \hat{w}_{k_0}$ is bigger than $\text{UB}_{\text{RL-}\ell_2}$, the value $\bar{w}_{k_0} = \hat{w}_{k_0}$ can be discarded as optimal solution of (RL-ℓ_2-M). This is because any solution with this value will provide a solution whose objective value is worse than $\text{UB}_{\text{RL-}\ell_2}$. Therefore, we can restrict ourselves to the values of $\hat{w}_{k_0}$ in such a way that $Z_{\hat{k}_0} \leq \text{UB}_{\text{RL-}\ell_2}$. Therefore, according to Theorem 2.2, it holds that:

$$Z_{k_0} + \bar{w}_{k_0}\left(\frac{1}{2}\bar{w}_{k_0} + \tilde{w}_{k_0} - \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0}\right) \leq \text{UB}_{\text{RL-}\ell_2}.$$

Therefore, $\bar{w}_{k_0}$ verifies:

$$\sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0} - \tilde{w}_{k_0} - \sqrt{\left(\tilde{w}_{k_0} - \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0}\right)^2 - 2\left(Z_{k_0} - \text{UB}_{\text{RL-}\ell_2}\right)} \leq \bar{w}_{k_0},$$

$$\bar{w}_{k_0} \leq \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0} - \tilde{w}_{k_0} + \sqrt{\left(\tilde{w}_{k_0} - \sum_{i=1}^{n} \bar{\alpha}_i y_i x_{ik_0}\right)^2 - 2\left(Z_{k_0} - \text{UB}_{\text{RL-}\ell_2}\right)}.$$

Taking into account that $\bar{w}_{k_0} = w_{k_0} - \tilde{w}_{k_0}$, the result is obtained. $\square$

As a consequence of the previous result and having computed the foregoing bounds for the w-variables, we include the following set of constraints in the formulation of problem (RL-ℓ_2-M):

$$\text{LB}_{w_k} \leq w_k \leq \text{UB}_{w_k}, \quad k \in D. \tag{2.39}$$

In the following, we present a result that allows us to compute valid values for the big M parameters, thus solving a problem for each instance of the dataset.

Proposition 2.6. *Problems (RL-ℓ_2) and (RL-ℓ_2-M) are equivalent if, for all $i \in N$, M_i is greater than or equal to the optimal objective value of the following problem, provided it is not unbounded:*

$$\begin{array}{lll} \left(UB^{\ell_2}_{M_i}\right) & \max & 1 - \xi_i - y_i \left(\sum\limits_{k=1}^{d} w_k x_{ik} + b \right) \\ & s.t. & (2.2), (2.4), (2.14), (2.39), \\ & & \frac{1}{2}\sum\limits_{k=1}^{d} w_k^2 + C\left(\sum\limits_{\ell=1}^{n} \xi_\ell + 2\sum\limits_{\ell=1}^{n} z_\ell\right) \leq UB_{RL\text{-}\ell_2}. \qquad (2.40) \end{array}$$

Proof:

It holds directly from (2.4) taking the maximum. □

Although the previous result provides tightened values of the big M parameters, solving a problem for each individual of the dataset tends to be inefficient. Hence, given a partition of the individuals, in the following result we present a strategy that allows us to obtain valid values of the big M parameters for individuals in the same element of that partition.

Proposition 2.7. *Problems (RL-ℓ_2) and (RL-ℓ_2-M) are equivalent if the following statements hold:*

i) *For all $i \in c_+$, where c_+ is a subset of the individuals with $y_i = +1$, M_i is greater than or equal to the optimal objective value of the following problem, provided it is not unbounded:*

$$\begin{array}{llll} \left(UB^{\ell_2}_{M_{c_+}}\right) & \max & 1 + \sum\limits_{k=1}^{d} v_k |\bar{x}_{c_+k}| - b & \\ & s.t. & (2.2), (2.4), (2.14), (2.39), (2.40), & \\ & & -w_k \leq v_k, & k \in D, \qquad (2.41) \\ & & w_k \leq v_k, & k \in D, \qquad (2.42) \\ & & v_k \leq \max\{|LB_{w_k}|, |UB_{w_k}|\}, & k \in D, \qquad (2.43) \end{array}$$

where $|\bar{x}_{c_+k}| = \max\limits_{i \in c_+.} \{|x_{ik}|\}$, for $k \in D$.

ii) *For all $i \in c_-$, where c_- is a subset of the individuals with $y_i = -1$, M_i is greater than or equal to the optimal objective value of the following* *problem, provided it is not unbounded:*

$$\left(UB^{\ell_2}_{M_{c_-}}\right) \quad \max \quad 1+\sum_{k=1}^{d} v_k|\bar{x}_{c_-k}|+b$$
$$s.t. \quad (2.2),(2.4),(2.14),(2.39)-(2.43),$$

where $|\bar{x}_{c_-k}| = \max_{i\in c_-.} \{|x_{ik}|\}$, *for* $k \in D$.

Proof:

For each $k \in D$ and $i \in c_+$ when $y_i = +1$, the following inequalities are satisfied: $-w_k x_{i_0k} \leq |w_k||\bar{x}_{c+k}|$. Taking the summation in k, we have:

$$1-\left(\sum_{k=1}^{d} w_k x_{ik}+b\right) \leq 1+\left(\sum_{k=1}^{d} |w_k||\bar{x}_{c+k}|-b\right), \quad \text{for } i \in c_+. \tag{2.44}$$

Since $0 \leq \xi_i \leq 2$, for $i \in c_+$ and the v-variables model the absolute value of the w-variables, we can conclude:

$$1-\xi_i-\left(\sum_{k=1}^{d} w_k x_{i_0k}+b\right) \leq 1+\left(\sum_{k=1}^{d} v_k|\bar{x}_{c+k}|-b\right), \quad \text{for } i \in c_+.$$

Therefore, the optimal solution of problem $\left(\text{UB}^{\ell_2}_{M_{c_+}}\right)$ is an upper bound of problem $\left(\text{UB}^{\ell_2}_{M_{i_0}}\right)$, for $i \in c_+$. Similarly, it can be proven that the optimal solution of problem $\left(\text{UB}^{\ell_2}_{M_{c_-}}\right)$ is an upper bound of problem $\left(\text{UB}^{\ell_2}_{M_{i_0}}\right)$, for $i \in c_-$. Therefore, according to Proposition 2.6, the result holds.

□

All the results of this section can be summarized in a strategy to solve the (RL-ℓ_2-M) model. Said strategy is described below in Algorithm 2.3. Similarly to Algorithm 2.2, Algorithm 2.3 has three different variants, depending on the procedure chosen to compute the big M parameters. Variant I solves a problem for each individual of the dataset. Variant II solves two problems (one for the individuals of class $y_i = +1$ and another for the individuals of class $y_i = -1$). In Variant III, the individuals are subdivided into clusters and a problem is solved for each cluster.

Observe that in Algorithm 2.3, unlike Algorithm 2.2, if we consider clusters containing just one individual of the dataset, the values of the big M parameters obtained in Variant III are not the same as those obtained in Variant I. This is because the w-variables are free and the bound is found on the addend $|w_k x_{ik}|$, for $i \in N, k \in D$. However, with the ℓ_1-norm, the positive and negative values of the w-variables are identified with w_k^+ and w_k^- respectively. This means that we are able to find the bound for the addend $w_k x_{ik}$, for $i \in N, k \in D$. In fact, in the tested datasets, the strategy of Proposition 2.7

Algorithm 2.3: Variant I, II, and III. Computation of tightened values of big M parameters in (RL-ℓ_2-M).

```
   Data: Training sample: $n$ elements and $d$ features.
   Result: Update values of $M_i$, $i \in N$ and obtain bounds for $w_k,\ k \in D$ and $b$.
 1 Solve the problem (SVM-$\ell_2$). From its optimal solution $(\tilde{w}, \tilde{b}, \tilde{\xi}, \tilde{z})$, build a feasible solution of (RL-$\ell_2$-M). Solve $(\overline{\text{SVM-}\ell_2})_{\tilde{z}}$ and build an improved feasible solution. Update the upper bound $\text{UB}_{\text{RL-}\ell_2}$.
 2 for $i \in N$ do
 3     $\text{dist}_i^2 = \max_{j \in N} \{\|x_i - x_j\|_2 : y_i = y_j\}, \quad M_i = \text{dist}_i^2 \cdot \sqrt{2\text{UB}_{\text{RL-}\ell_2}}$.
 4 Set $\text{LB}_{w_k} = -\sqrt{2\text{UB}_{\text{RL-}\ell_2}}$ and $\text{UB}_{w_k} = \sqrt{2\text{UB}_{\text{RL-}\ell_2}}$.
 5 do
 6     for $k_0 \in D$ do
 7         Obtain tighter values of $w$-bounds solving the following quadratic problems:
                 $\max/\min \quad w_{k_0}$
                 $\text{s.t.} \quad (2.2), (2.4), (2.14), (2.39), (2.40).$
           From the second iteration, include constraint (2.45) in the previous problem.
 8     Obtain lower and upper bounds for $b$-variable, i.e., $\text{UB}_b$ and $\text{LB}_b$ solving the following problems:
                 $\max/\min \quad b$
                 $\text{s.t.} \quad (2.2), (2.4), (2.14), (2.39), (2.40).$
 9     Include the following constraint in the formulation of the problem:
                 $\text{LB}_b \le b \le \text{UB}_b. \qquad (2.45)$
       Solve the problem (Re-RL-$\ell_2$) +(2.45), build $(\overline{\text{Re-RL-}\ell_2})_{k_0}$ + (2.45) and apply Corollary 2.4.2 to update bounds for the $w$-variables: $\text{LB}_{w_k}$ and $\text{UB}_{w_k}$.
10     case *Variant I* do
11         for $i_0 \in N$ do
12             Solve $\left(\text{UB}_{M_{i_0}}^{\ell_2}\right)$ + (2.45) and update the big M parameters.
13     case *Variant II* do
14         Update $M_i$ solving $\left(\text{UB}_{M_{c_+}}^{\ell_2}\right)$ + (2.45) for $c_+ = \{i \in N : y_i = 1\}$ and $\left(\text{UB}_{M_{c_-}}^{\ell_2}\right)$ + (2.45) for $c_- = \{i \in N : y_i = -1\}$.
15     case *Variant III* do
16         Subdivide the individuals of each class in clusters. Let $C_+, C_-$ be the set of clusters of class 1 and class $-1$ respectively.
17         for $c_+ \in C_+$ do
18             Update $M_i$, for $i \in c_+$, by solving $\left(\text{UB}_{Mc_+}\right)$+(2.45).
19         for $c_- \in C_-$ do
20             Update $M_i$, for $i \in c_-$, by solving $\left(\text{UB}_{Mc_-}\right)$+(2.45).
21 while *an improvement of the bounds is obtained*;
```

did not provide good results. Although applying the strategy only took a short amount of time, the improvement on the big M parameters was small and in some cases almost zero.

2.5 Computational Experiments

In this section, we present the results obtained with our computational experiments. Specifically, we present a comparison of several solution approaches for solving the models (RL-ℓ_1), (RL-ℓ_1-M), (RL-ℓ_2), and (RL-ℓ_2-M) using simulated and real-life datasets.

The experiments were conducted on an Intel(R) Xeon(R) W-2135 CPU 3.70 GHz 32 GB RAM computer, using CPLEX 12.7.0. in Concert Technology C++. As done in Belotti et al. (2016), due to the presence of big M parameters, the relative MIP tolerance and the integrality tolerance were fixed to zero. The remaining parameters were set to their default values.

2.5.1 Data

The computational experiments were carried out on simulated and real-life datasets. The simulated datasets used in our experiments were Type A and Type B datasets proposed in Brooks (2011). These datasets were also used in Carrizosa et al. (2014) and Belotti et al. (2016). The latter only included experiments conducted on a challenging subset of instances with the following characteristics: n=100, d=2, Type B. We used the Type A and Type B datasets with $n = 160$, i.e., 160 individuals, and $d = 2, 5, 10$, i.e., $2, 5$, or 10 features, respectively. Furthermore, the percentage of elements in each class of these instances is 50%. We denoted them as: n"number of individuals"d"number of features""Type of data". Thusly, n160d2B signifies that the dataset has 160 individuals, two features, and Type B data.

The real-life datasets are from the UCI repository Lichman (2013) and from Prokhorov (2001). They are specified in Table 2.1, where n is the number of individuals, d is the number of features and the last column states the percentage of elements in each class. Observe that these datasets have been used to analyze the performance of ramp loss models (see Brooks (2011); Carrizosa et al. (2014); Huang et al. (2014)).

In the simulated and real-life datasets, the analysis uses the following values for parameter C, as done in Brooks (2011):

$$C \in \{0.01, 0.1, 1, 10, 100\},$$

and a time limit of 7200 seconds is imposed, unless stated otherwise.

Label	Name in repository	n	d	Class(%)
Wpbc	Breast cancer Wisconsin (prognostic)	194	30	76/24
SONAR	Connectionist bench (sonar, mines vs. rocks)	208	60	54/46
SPECT	Cardiac Single Proton Emission Computed Tomography images	267	22	79/21
IONO	Ionosphere	351	33	64/36
Wdbc	Breast cancer Wisconsin (diagnostic)	569	30	63/37
WBC	Breast cancer Wisconsin (original)	683	9	65/35
Ijcnn1	IJCNN 2001 Neural Network Competition	35000	22	91/9

Table 2.1. *Real-life datasets.*

2.5.2 Exact procedure for (RL-ℓ_1-M)

In this section, the exact approaches proposed in Section 2.3 for solving the (RL-ℓ_1-M) formulation are tested and compared with the (RL-ℓ_1) formulation. We study the performance of different variants using several simulated and real-life datasets. To simplify the notation, we write the name of the algorithm followed by the chosen variant, e.g. Algorithm 2.2.1.II represents Algorithm 2.2 with Variant 1 in Step 1, i.e., we solve Algorithm 2.1 using Variant 1 and Variant II (Steps 9 and 10). In each strategy, we also analyze the improvement of the big M parameters. To do this, we compare the values obtained in Step 3 of Algorithm 2.1, M_i^{initial}, with the tightened values obtained at the end of the applied strategy, M_i^{final}, i.e., we compute the improvement of M, M's Impr. $= \dfrac{1}{n}\sum_{i=1}^{n}\dfrac{M_i^{\text{initial}} - M_i^{\text{final}}}{M_i^{\text{initial}}}$. Therefore, this measure depends on M_i^{initial}. As a consequence, we can only compare these values when the strategies are applied to the same data.

We first compare two different methods for solving (RL-ℓ_1-M) and one for solving (RL-ℓ_1) on simulated data: a) Algorithm 2.2.1.I, b) Algorithm 2.2.2.I, and c) by formulating the model with indicator constraints that generate locally valid implied bound cuts. The latter method was developed by Belotti et al. (2016) and it has been a feature of IBM-Cplex since version 12.6.1. We used the method included in IBM-Cplex 12.7.0. This solution approach is denoted as Ind. Const. (LIC).

The results are shown in Table 2.2, with the first column indicating the name of the dataset and the second column the value of parameter C. The next eight columns contain information about strategy performance and the resolution process: the first and the fifth columns detail the improvement of M during the procedure; the second and the sixth show the strategy time; the third and the seventh indicate the total time, i.e., strategy time plus final time; and the fourth and the eighth report the MIP relative GAP within the time limit. The next groups of columns show information about the behavior of Ind. Const. (LIC): the first column details the total time and the second is the GAP obtained within the time limit (7200 seconds). When the problem was solved to optimality, the strategy that took the least time is shown in bold, otherwise, the strategy that provided the best GAP within the time limit is highlighted.

Data	C	Algorithm 2.2.1.I				Algorithm 2.2.2.I				Ind. Const. (LIC)	
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	t	GAP
n160d2A	100	**96.77%**	**1.03**	**2.14**	**0.00%**	96.77%	1.19	36.19	0.00%	7207.89	52.50%
	10	**96.78%**	**1.16**	**2.12**	**0.00%**	96.78%	1.17	2.17	0.00%	7215.59	57.82%
	1	96.91%	1.47	1.67	0.00%	**97.08%**	**1.44**	**1.64**	**0.00%**	7207.02	70.17%
	0.1	**4.19%**	**0.63**	**1.35**	**0.00%**	25.02%	1.59	5.12	0.00%	7205.92	84.81%
	0.01	**99.95%**	**0.11**	**0.11**	**0.00%**	**99.95%**	**0.11**	**0.11**	**0.00%**	7208.62	82.28%
n160d5A	100	**95.09%**	**1.20**	**7202.99**	**34.95%**	95.09%	1.11	7202.82	37.04%	7233.20	58.56%
	10	94.94%	1.34	7201.46	7.91%	**94.94%**	**1.33**	**7206.68**	**5.50%**	7206.77	64.85%
	1	94.13%	1.58	10.02	0.00%	**94.11%**	**1.68**	**9.11**	**0.00%**	7205.77	82.05%
	0.1	27.69%	1.76	3.88	0.00%	**38.04%**	**1.91**	**3.59**	**0.00%**	7214.75	73.38%
	0.01	99.95%	0.11	0.11	0.00%	**99.97%**	**0.11**	**0.11**	**0.00%**	7207.12	82.05%
n160d10A	100	**93.16%**	**1.56**	**7203.39**	**52.42%**	93.16%	1.50	7205.10	53.09%	7204.69	69.99%
	10	92.89%	1.64	7216.17	44.64%	**92.91%**	**1.42**	**7203.62**	**42.90%**	7206.72	70.32%
	1	91.32%	2.50	7214.95	8.78%	**91.33%**	**2.37**	**7214.92**	**7.70%**	7219.48	77.56%
	0.1	9.31%	1.04	74.70	0.00%	**33.01%**	**2.53**	**5.39**	**0.00%**	7208.65	82.05%
	0.01	**99.80%**	**0.13**	**0.13**	**0.00%**	100.00%	0.13	0.14	0.00%	7208.42	82.05%
n160d2B	100	**55.55%**	**1.16**	**7203.32**	**52.15%**	55.56%	1.09	7202.90	52.18%	7205.64	62.71%
	10	55.56%	1.11	7203.02	5.90%	**55.60%**	**1.03**	**7207.48**	**4.27%**	7213.55	65.13%
	1	55.59%	1.33	3.28	0.00%	**56.00%**	**1.31**	**3.23**	**0.00%**	7207.61	71.05%
	0.1	56.57%	1.40	1.92	0.00%	**60.85%**	**1.39**	**1.78**	**0.00%**	7205.80	81.16%
	0.01	**96.66%**	**0.11**	**0.11**	**0.00%**	99.93%	0.16	0.16	0.00%	7207.97	79.71%
n160d5B	100	**72.92%**	**1.94**	**7203.99**	**58.47%**	72.92%	1.72	7203.87	58.98%	7206.19	69.22%
	10	**72.73%**	**1.52**	**7203.48**	**57.97%**	72.73%	1.53	7204.19	58.45%	7209.94	71.63%
	1	69.07%	1.80	7204.08	19.55%	**69.12%**	**1.77**	**7209.23**	**16.87%**	7207.62	72.30%
	0.1	47.41%	1.27	3.70	0.00%	**51.88%**	**1.22**	**3.67**	**0.00%**	7207.81	80.57%
	0.01	**99.95%**	**0.11**	**0.11**	**0.00%**	99.95%	0.11	0.13	0.00%	7207.92	80.82%
n160d10B	100	60.77%	2.10	7204.75	48.42%	**60.77%**	**1.88**	**7203.59**	**47.42%**	7206.53	67.00%
	10	59.31%	2.17	7204.57	51.76%	**59.42%**	**2.10**	**7204.57**	**48.42%**	7208.75	58.20%
	1	79.95%	1.78	7213.97	23.85%	**80.12%**	**1.78**	**7213.57**	**20.96%**	7215.72	66.13%
	0.1	42.01%	1.19	173.63	0.00%	**45.90%**	**1.19**	**137.97**	**0.00%**	7207.61	80.01%
	0.01	99.98%	0.13	0.14	0.00%	**99.79%**	**0.13**	**0.13**	**0.00%**	7208.92	80.82%

Table 2.2. *Performance of exact approaches on simulated data for solving (RL-ℓ_1-M) and (RL-ℓ_1).*

Note that our strategies improve the behavior of Ind. Const. (LIC) in all cases. In fact, in dataset n160d2A, the Ind. Const. (LIC) approach has a GAP of between 52.50% and 84.81% in two hours and, using our strategy, all the individuals are solved to optimality in less than 3 seconds. In the majority of cases, the performance of Algorithm 2.2 using Variant 2 is better than using Variant 1. Also, when Variant 1 works better than Variant 2 there are no significant differences. We can therefore conclude that, when applied to these simulated datasets, the strategy with the best performance is Algorithm 2.2.2.I. A preliminary test was carried out by solving the problems with the application of Variant II, but the obtained GAPs were worse than with Variant I. Since the time employed in the strategy is less than three seconds in all cases, there is no need to apply Variant III, in which the values of the resulting big M parameters would be less tightened. Note that a 100% of improvement of M does not mean that all the big M parameters are zero, seeing as the improvement of M is the average of the improvement of each M_i, for $i \in N$, and several of them could be improved by over 100%.

Next, we compare the three different methods for solving (RL-ℓ_1-M) and (RL-ℓ_1) respectively on real-life datasets: a) Algorithm 2.2.1.I, b) Algorithm 2.2.2.I, and c) Ind. Const. (LIC). The results are shown in Table 2.3, which uses a similar notation to Table 2.2.

Data	C	Algorithm 2.2.1.I				Algorithm 2.2.2.I				Ind. Const. (LIC)	
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	t	GAP
Wpbc	100	82.88%	3.78	7206.12	63.47%	**82.88%**	**3.75**	**7205.97**	**62.11%**	7201.56	73.44%
	10	81.76%	4.27	7217.15	59.72%	**81.75%**	**4.22**	**7216.50**	**58.19%**	7203.06	72.71%
	1	**66.11%**	**3.54**	**7216.89**	**35.94%**	66.21%	3.41	7217.17	36.63%	7210.61	70.74%
	0.1	**62.62%**	**2.32**	**2.53**	**0.00%**	69.41%	2.84	3.06	0.00%	7206.17	67.39%
	0.01	**99.97%**	**0.22**	**0.22**	**0.00%**	**99.94%**	**0.22**	**0.22**	**0.00%**	7205.88	67.39%
SONAR	100	97.92%	11.83	12.38	0.00%	97.92%	11.01	11.90	0.00%	**2.20**	**0.00%**
	10	**90.84%**	**10.25**	**2058.46**	**0.00%**	90.83%	8.76	2686.02	0.00%	7201.62	29.98%
	1	78.13%	5.00	7217.93	46.20%	**78.07%**	**4.59**	**7218.52**	**43.16%**	7202.05	69.25%
	0.1	64.74%	2.66	7215.46	9.29%	**64.72%**	**2.42**	**7214.13**	**8.94%**	7207.08	81.12%
	0.01	28.91%	0.89	4.32	0.00%	**73.80%**	**1.53**	**2.42**	**0.00%**	7206.15	85.57%
SPECT	100	63.26%	4.82	7206.89	51.75%	63.21%	4.83	7211.33	51.18%	**7202.47**	**34.16%**
	10	**64.56%**	**6.43**	**7208.20**	**27.37%**	64.22%	4.75	7206.52	31.43%	7203.11	36.40%
	1	60.43%	4.92	7209.06	16.96%	**60.44%**	**4.95**	**7207.00**	**14.51%**	7205.19	42.86%
	0.1	53.06%	3.87	17.63	0.00%	**53.77%**	**4.00**	**16.31**	**0.00%**	7203.19	43.64%
	0.01	74.25%	2.61	3.21	0.00%	**99.55%**	**0.28**	**0.31**	**0.00%**	7204.44	32.73%
IONO	100	**94.95%**	**29.86**	**153.58**	**0.00%**	94.92%	29.38	159.09	0.00%	7201.85	34.23%
	10	90.97%	26.77	7228.58	19.11%	**90.97%**	**25.61**	**7227.58**	**18.93%**	7202.60	57.00%
	1	**85.00%**	**24.11**	**7237.08**	**41.99%**	85.06%	22.71	7236.20	42.66%	7201.71	65.34%
	0.1	**78.91%**	**18.68**	**7232.26**	**20.20%**	78.95%	18.55	7232.33	22.22%	7204.15	81.53%
	0.01	**42.85%**	**11.57**	**16.17**	**0.00%**	77.23%	16.76	19.15	0.00%	7205.09	88.89%
Wdbc	100	99.09%	41.83	42.50	0.00%	99.08%	38.97	39.74	0.00%	**11.62**	**0.00%**
	10	98.18%	40.43	46.01	0.00%	**98.12%**	**38.37**	**43.90**	**0.00%**	2432.62	0.00%
	1	101.34%	45.64	46.00	0.00%	**101.33%**	**40.90**	**41.18**	**0.00%**	7204.85	63.00%
	0.1	102.21%	37.76	37.82	0.00%	**102.22%**	**29.47**	**29.58**	**0.00%**	7202.43	86.06%
	0.01	32.13%	23.52	27.09	0.00%	**38.52%**	**19.80**	**23.72**	**0.00%**	7204.90	93.87%
WBC	100	**95.88%**	**52.02**	**149.29**	**0.00%**	95.88%	50.90	163.51	0.00%	7212.06	39.09%
	10	**95.87%**	**43.90**	**125.14**	**0.00%**	95.78%	44.25	135.87	0.00%	7207.31	30.46%
	1	**95.90%**	**34.68**	**180.94**	**0.00%**	95.89%	34.13	183.21	0.00%	7207.86	39.13%
	0.1	98.11%	32.47	33.04	0.00%	**97.54%**	**32.11**	**32.77**	**0.00%**	7203.27	51.23%
	0.01	97.69%	22.79	22.85	0.00%	**97.69%**	**21.76**	**21.82**	**0.00%**	7205.73	82.77%

Table 2.3. *Performance of exact approaches on real data for solving (RL-ℓ_1-M) and (RL-ℓ_1).*

The results in Table 2.3 show that our strategies improve the behavior of Ind. Const. (LIC) on real-life datasets in almost all the cases. In fact, when the problem can be solved to optimality in less than two hours with the application of the Ind. Const. (LIC) approach, it can be solved to optimality using our approaches. Conversely, the opposite is not true. It is worth highlighting the WBC dataset instances where the Ind. Const. (LIC) approach has a GAP of between 30.46% and 82.77% in two hours. However, when using our strategy, all the individuals are solved to optimality in less than three minutes. In the majority of cases, the performance of Algorithm 2.2.2.I is better than Algorithm 2.2.1.I. A preliminary test was carried out by applying Variant II, but the obtained GAPs were worse than when Variant I was used. This is with the exception of the SPECT dataset, the results of which will be analyzed later. Since the strategy is completed in less than one minute in all cases, it is not worth applying Variant III.

Regarding the time employed in the strategy, note that in the majority of cases Algorithm 2.2.I requires less time than Algorithm 2.1.I. This seems logical because Algorithm 2.2.I solves fewer problems in each iteration. However, there are some cases, e.g. in the IONO dataset in which the value of parameter C is equal to 0.01, where the strategy completion time is longer in Algorithm 2.2.I than in Algorithm 2.1.I. This is because the number of iterations is not the same in both procedures. In fact, in the given example (IONO $C = 0.01$), the number of iterations of Algorithm 2.1.I, i.e., the number of times that Steps 4-7 are executed, is bigger than the number of iterations of Algorithm 2.2.I.

We will now discuss the particular case of the SPECT dataset. Table 2.4 compares the performances of Algorithm 2.2.1.II and Algorithm 2.2.2.II with the strategies analyzed previously. The first column states the name of the dataset and the second column the value of parameter C. The table also contains three groups of columns with information about the three different strategies: a) the performance of the best strategy tested in Table 2.3, b) the behavior of Algorithm 2.2.1.II, and c) the performance of Algorithm 2.2.2.II. We can observe that the performance of the previous best strategy from Table 2.3 is worse than using Algorithm 2.2 with Variant II for all the values of C tested. Although the big M parameters are less tightened using Variant II, the resolution of the problem is faster. If we focus our attention on $C = 100, C = 10$, and $C = 1$, we note that when using the previous strategies the problems are not solved to optimality in two hours, but when Variant II is used, they are solved to optimality in less than $29, 184$, and 30 seconds respectively. In view of these results, we can claim that tighter values of the big M parameters (on average) do not always correspond to a better performance in general, but they do in the majority of the cases.

Data	C	Best method of Table 2.3				Algorithm 2.2.1.II				Algorithm 2.2.2.II			
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP
SPECT	100	-	-	7202.47	34.16%	**41.22%**	**0.20**	**28.46**	**0.00%**	41.22%	0.16	29.98	0.00%
	10	64.56%	6.43	7208.20	27.37%	41.42%	0.14	214.82	0.00%	**41.38%**	**0.16**	**183.75**	**0.00%**
	1	60.44%	4.95	7207.00	14.51%	**41.72%**	**0.19**	**29.30**	**0.00%**	41.79%	0.16	38.12	0.00%
	0.1	53.77%	4.00	16.31	0.00%	42.60%	0.14	3.94	0.00%	**44.98%**	**0.11**	**3.31**	**0.00%**
	0.01	99.55%	0.28	0.31	0.00%	75.08%	0.13	0.70	0.00%	**97.77%**	**0.05**	**0.08**	**0.00%**

Table 2.4. *Performance of Algorithm 2.2 Variant II on SPECT for solving (RL-ℓ_1-M).*

In this paragraph, we analyze the Ijcnn1 dataset which contains a large number of individuals. In particular, we test three random subsets of 2000, 3500, and 5000 individuals respectively. These datasets maintain the same percentage of each class as the original dataset. Additionally, Ijcnn1_2000 is contained in Ijcnn1_3500 and this, in turn, is contained in Ijcnn1_5000. We compare the performance of the three variants (I, II, and II) of Algorithm 2.2 using Variant 2, because after a preliminary test, Variant 2 provided better results. For Variant III, the k-means and the k-median clusters were solved using *The C clustering library*, see de Hoon et al. (2004) for further details.

We tested performance using the k-means and the k-median clustering algorithms, establishing a different number of clusters. Specifically, we carried out experiments by defining the subsets of each class as 5%, 10%, 20%, 30%, 40%, and 50% of the number of individuals of each class. For example, in a dataset with 100 individuals where the ratio of elements in each class is 60/40, the 10% of clusters means making 6 clusters and 4 clusters for each class respectively.

Table 2.5 describes the best case performance (from among the different sizes of clusters and the clustering algorithms), i.e., when the problem is solved to optimality, the percentage that took the least time and the applied clustering algorithm are shown, otherwise, the ones providing the best GAP are selected. Therefore, the structure of Table 2.5 is quite similar to the previous one: the first column states the name of the dataset and the second column shows the value of parameter C. The following eight columns contain information about the strategy performance and the resolution process of Variants I and II. The following six columns show information about the strategy performance of Variant III: with the first stating the cluster algorithm used; the next showing the percentage chosen to determine the number of clusters; the third showing the improvement of M during the procedure; the fourth showing the strategy time plus the time spent in clustering the data; the fifth showing the total time, i.e., the clustering time, the strategy time, and the resolution time; and the sixth gives the MIP relative GAP within the time limit. The next groups of columns provide information about the behavior of Ind. Const. (LIC). Two blocks of columns provide information about this approach: in the first, the time limit is two hours and in the second, the time limit is the maximum between two hours and total time spent on the best performing Variant of Algorithm 2.2. When the problem is solved to optimality, the procedure that takes the least time is highlighted, otherwise, the procedure that provides the best GAP within the time limit is shown. Note that we did not established a time limit for the strategies.

As shown in Table 2.5, Variant III allows us to find an equilibrium between the time taken by the strategy and the resolution time. Variant III is especially significant in datasets with a large number of individuals, seeing as the strategy time of Variant I with the same set is huge. In the case of Ijcnn1_5000 when parameter C is equal to 0.1, the time taken by Variant I strategy is around three and a half hours, but when using Variant III, the strategy takes less than one hour and a half. Furthermore, the problem is solved to optimality in less than one minute after applying the strategy, while the Ind. Const. (LIC) approach provides a GAP of 87.72% in two hours. In almost all cases, Variant III provides the best results for these datasets and when Variant I finds the

Data	C	Algorithm 2.2.2.I				Algorithm 2.2.2.II				Algorithm 2.2.2.III						Ind. Const. (LIC) T.L.:7200 s		Ind. Const. (LIC) T.L.: Best variant of Alg. 2	
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	Cluster	Per	M's Impr.	$t_{st} + t_{cl}$	t_{total}	GAP	t	GAP	t	GAP
Ijcnn1_2000	100	110.76%	12.65	12.78	0.00%	75.55%	0.34	0.56	0.00%	k-means	0.05	94.94%	2.56	2.76	0.00%	**0.17**	**0.00%**	**0.17**	**0.00%**
	10	110.75%	70.64	70.75	0.00%	57.60%	0.48	1.33	0.00%	k-means	0.05	93.52%	4.84	5.11	0.00%	**0.39**	**0.00%**	**0.39**	**0.00%**
	1	106.07%	231.43	231.62	0.00%	**42.04%**	**0.53**	**3.54**	**0.00%**	k-means	0.05	87.52%	7.51	8.64	0.00%	86.90	0.00%	86.90	0.00%
	0.1	**92.90%**	**246.30**	**618.37**	**0.00%**	81.48%	2.02	7214.31	6.05%	k-means	0.2	92.91%	48.79	7255.61	4.61%	7202.15	71.84%	7202.15	71.84%
	0.01	75.08%	511.19	7714.37	12.65%	31.35%	1.33	7204.47	33.22%	***k*-median**	**0.4**	**74.50%**	**363.30**	**7577.83**	**7.65%**	7202.85	93.88%	7580.57	93.88%
Ijcnn1_3500	100	99.83%	3015.88	3017.33	0.00%	11.89%	1.03	4.41	0.00%	k-means	0.05	71.46%	17.42	20.87	0.00%	**0.79**	**0.00%**	**0.79**	**0.00%**
	10	100.45%	2463.74	2464.90	0.00%	17.96%	1.14	34.97	0.00%	***k*-means**	**0.05**	**76.45%**	**21.99**	**32.94**	**0.00%**	90.64	0.00%	90.64	0.00%
	1	99.72%	3834.23	3863.66	0.00%	23.26%	1.33	7208.13	20.68%	***k*-means**	**0.4**	**95.09%**	**958.97**	**1185.13**	**0.00%**	7207.04	47.34%	7207.04	47.34%
	0.1	105.72%	4204.94	4205.96	0.00%	35.93%	2.64	7220.43	38.41%	***k*-medians**	**0.2**	**97.85%**	**582.63**	**602.70**	**0.00%**	7202.48	83.45%	7202.48	83.45%
	0.01	79.82%	2853.94	10059.62	27.80%	31.64%	6.16	7209.88	36.16%	***k*-median**	**0.2**	**76.49%**	**678.20**	**7887.54**	**24.09%**	7202.71	96.04%	7890.85	96.04%
Ijcnn1_5000	100	99.76%	9715.88	9721.47	0.00%	10.96%	1.86	198.49	0.00%	***k*-means**	**0.05**	**75.10%**	**53.85**	**165.84**	**0.00%**	406.17	0.00%	406.17	0.00%
	10	99.73%	9228.51	9285.11	0.00%	13.95%	1.97	5482.24	0.00%	***k*-means**	**0.05**	**77.13%**	**79.84**	**799.48**	**0.00%**	7202.35	37.50%	7202.35	37.50%
	1	**99.50%**	**11661.60**	**18867.71**	**7.77%**	19.74%	2.56	7214.08	46.08%	k-means	0.5	95.97%	3547.25	10754.09	16.43%	7205.38	65.69%	18881.80	63.32%
	0.1	104.69%	11977.60	11978.90	0.00%	32.72%	3.03	7216.60	46.93%	***k*-medians**	**0.5**	**100.86%**	**4268.44**	**4315.35**	**0.00%**	7202.62	87.72%	7202.62	87.72%
	0.01	78.71%	9264.50	16471.30	32.38%	32.21%	10.90	7213.46	50.00%	***k*-means**	**0.4**	**77.47%**	**3810.58**	**11017.05**	**32.17%**	7203.10	96.68%	11021.00	96.68%

Table 2.5. *Performance of exact approaches for solving (RL-ℓ_1-M) and (RL-ℓ_1) on Ijcnn.*

optimal solution within the time limit, Variant III tends to find it in less time. On the other hand, we noted that Variant I provides better bounds on the big M parameters than Variants II and III (the improvement of M is greater in all cases). However, the time it takes means that it is impractical. Whatsmore, even for a small number of clusters, i.e., 5%, Variant III provides a significant improvement of the big M parameters compared to Variant II. In conclusion, we have proposed several strategies that allow us to solve many more problems to optimality within the time limit than if the indicator constraints had been used, i.e., by applying the Ind. Const. (LIC) approach. In the large dataset in which there is a high computational cost for the proposed resolution methods, we compared three different variants of Algorithm 2.2. The objective of this was to achieve an equilibrium between the time employed in the strategy (which is directly related to the improvement of the big M parameters) and the time spent in solving the problem.

2.5.3 Exact procedure for (RL-ℓ_2-M)

In this section, the different variants of the exact approaches proposed in Section 2.4 for solving the (RL-ℓ_2-M) formulation are tested and compared with the (RL-ℓ_2) formulation. Specifically, we compare performance using several simulated and real-life datasets with three different resolution methods: a) the variants of Algorithm 2.3 – we analyze the improvement of the big M parameters when applying the strategy, similarly to the ℓ_1-norm case; b) the resolution method proposed in Belotti et al. (2016) taking advantage of our strategies to tighten the values of big M parameters (since this procedure needs an unique initial big M parameter, we establish it as $\mathrm{M} = \max\limits_{i \in N}\{\mathrm{M}_i\}$, where M_i are the best values obtained with our strategy, Algorithm 2.3.I); c) the resolution method proposed in Belotti et al. (2016) using Corollary 2.4.1 to establish the initial big M parameter; d) solving the (RL-ℓ_2) formulation by applying the method developed in Belotti et al. (2016) which is included in IBM-Cplex 12.7.0, i.e., generating locally valid implied bound cuts. This latter solution approach is denoted as Ind. Const. (LIC).

We first compare the four different methods for solving (RL-ℓ_2-M) and (RL-ℓ_2) on simulated data, using Variant I of Algorithm 2.3. The results are shown in Table 2.6, with the first column indicating the name of the dataset and the second column the values of parameter C. The following groups of columns contain information about the strategy's performance and the resolution process: a) the improvement of M during the procedure; b) the time taken by the strategy; c) the total time, i.e., strategy time plus resolution time and; d) the MIP relative GAP within the time limit (7200 seconds).

The next group of columns presents information about the resolution method proposed in Belotti et al. (2016) but using the big M parameter obtained after applying our strategy: a) the percentage of improvement of the big M parameter with respect to the one obtained by applying Corollary 2.4.1, b) the time taken by the strategies; c) the total time, i.e., strategy time plus resolution time and; d) the MIP relative GAP within the time limit. The third group of columns report the results of the same method of Belotti et al. (2016) but with the initial big M parameter (Corollary 2.4.1): a) the time taken by the strategy; b) the total time, i.e., strategy time plus resolution time and; c) the MIP relative GAP within the time limit. The last group of columns shows the behavior of Ind. Const. (LIC): the first column shows the total time and the second shows the GAP obtained within the time limit. The strategy with the best performance is highlighted.

Data	C	Algorithm 2.3.I				Belotti et al. (2016) Big M: Algorithm 2.3.I				Belotti et al. (2016) Big M: Corollary 2.4.1			Ind. Const. (LIC)	
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	t_{st}	t_{total}	GAP	t	GAP
n160d2A	100	**95.13%**	**21.20**	**23.28**	**0.00%**	58.31%	30.43	7230.49	52.94%	9.80	7209.83	54.38%	7204.46	57.71%
	10	**95.98%**	**20.21**	**20.47**	**0.00%**	68.09%	30.21	473.61	0.00%	7.63	7207.74	58.49%	7203.04	69.05%
	1	**92.80%**	**33.82**	**33.86**	**0.00%**	75.46%	46.92	7247.01	25.02%	12.34	7212.40	59.52%	7202.29	79.63%
	0.1	**34.31%**	**29.94**	**7232.15**	**35.63%**	37.01%	40.88	7240.91	59.67%	9.75	7209.80	65.12%	7202.11	83.21%
	0.01	**37.43%**	**25.06**	**7227.78**	**0.87%**	38.91%	36.19	7236.22	3.27%	8.84	7208.92	13.47%	7202.82	84.39%
n160d5A	100	**91.82%**	**21.92**	**7231.52**	**19.89%**	52.79%	33.76	7233.81	60.78%	10.24	7210.29	66.21%	7202.70	60.16%
	10	**92.63%**	**25.31**	**920.06**	**0.00%**	60.55%	40.12	7214.93	53.72%	11.17	7240.24	68.06%	7202.57	71.49%
	0.1	**40.67%**	**34.60**	**7236.97**	**41.55%**	60.46%	43.96	7209.43	46.69%	9.39	7244.03	52.77%	7201.94	84.20%
	1	**87.19%**	**38.24**	**7238.28**	**2.00%**	37.42%	46.75	7208.65	29.77%	10.37	7246.89	65.56%	7202.43	79.69%
	0.01	**36.06%**	**40.81**	**7242.89**	**3.16%**	51.97%	53.65	7212.95	8.28%	12.14	7253.76	52.44%	7202.49	85.67%
n160d10A	100	**88.80%**	**23.80**	**7225.69**	**50.92%**	47.88%	31.99	7232.31	81.70%	7.51	7207.62	84.98%	7202.79	69.39%
	10	**88.93%**	**26.71**	**7228.74**	**39.06%**	51.62%	37.27	7210.62	71.99%	8.27	7237.33	76.19%	7203.00	73.20%
	1	**80.82%**	**44.99**	**7246.83**	**49.48%**	45.03%	54.31	7209.33	69.49%	7.67	7254.32	72.83%	7205.35	82.99%
	0.1	**32.85%**	**44.83**	**7246.93**	**55.43%**	23.53%	58.27	7215.04	65.28%	9.79	7259.87	65.23%	7204.11	84.65%
	0.01	**22.29%**	**42.85**	**7244.36**	**27.69%**	18.80%	53.88	7211.13	29.37%	11.69	7253.98	35.19%	7206.03	85.88%
n160d2B	100	**45.72%**	**34.55**	**7234.60**	**14.72%**	37.03%	43.94	7243.96	65.37%	8.96	7209.04	67.99%	7207.06	63.91%
	10	**45.71%**	**32.08**	**7232.14**	**7.86%**	37.11%	42.43	7242.54	62.53%	9.67	7209.68	63.03%	7211.17	69.15%
	1	**45.83%**	**27.53**	**6342.20**	**0.00%**	37.56%	36.90	7236.91	5.03%	7.94	7208.03	55.79%	7206.39	74.79%
	0.1	49.92%	25.93	7226.06	8.71%	46.79%	34.96	7234.98	15.03%	**7.65**	**7207.77**	**6.95%**	7202.59	82.60%
	0.01	**99.95%**	**12.53**	**12.55**	**0.00%**	**99.98%**	**12.55**	**12.55**	**0.00%**	10.83	7210.89	8.47%	7207.95	82.65%
n160d5B	100	**51.88%**	**36.08**	**7236.26**	**63.96%**	34.82%	44.90	7244.95	72.96%	7.93	7207.97	73.86%	7207.12	65.73%
	10	**51.50%**	**32.79**	**7232.90**	**50.58%**	34.81%	44.22	7244.23	74.30%	11.18	7211.23	71.17%	7204.61	72.91%
	1	**44.65%**	**29.60**	**7229.63**	**36.32%**	33.80%	38.59	7238.61	52.32%	10.96	7211.09	49.82%	7204.81	75.30%
	0.1	**31.88%**	**23.45**	**7225.80**	**47.80%**	31.55%	31.56	7232.10	41.38%	16.17	7216.26	54.28%	7206.88	81.36%
	0.01	**50.31%**	**33.46**	**7236.20**	**6.06%**	57.05%	40.83	7240.89	17.02%	7.28	7207.39	53.45%	7202.51	83.54%
n160d10B	100	**30.36%**	**28.84**	**7234.94**	**51.97%**	32.10%	44.32	7244.52	80.00%	16.31	7216.37	82.20%	7202.03	65.16%
	10	**29.70%**	**32.81**	**7234.46**	**53.72%**	31.95%	43.32	7243.42	78.16%	8.31	7208.51	79.61%	7217.02	68.35%
	1	**27.33%**	**28.70**	**7230.35**	**51.52%**	31.04%	45.42	7245.52	67.32%	7.23	7207.26	72.71%	7202.42	70.35%
	0.1	**25.35%**	**29.57**	**7231.70**	**50.20%**	30.23%	45.23	7245.44	60.36%	12.55	7214.25	70.18%	7202.04	77.84%
	0.01	**25.02%**	**25.67**	**7229.13**	**33.20%**	32.80%	36.14	7236.21	49.54%	14.77	7214.86	60.16%	7202.42	84.91%

Table 2.6. *Performance of exact approaches on simulated data for solving (RL-ℓ_2-M) and (RL-ℓ_2).*

Note that, in almost all the cases, the behavior of our strategy is better than the algorithm proposed in Belotti et al. (2016), which in turn is better than solving the problem using the Ind. Const. (LIC) approach. For instance, in dataset n160d2A with a C parameter equal to 1, our strategy provides the optimal solution in a total time of 34 seconds. Meanwhile, the GAP of the algorithm from Belotti et al. (2016) using the improved big M derived from Algorithm 2.3.I and with a time limit of two hours is 25.02%, and the same algorithm without the improved big M is 59.02%. Finally, the GAP of the Ind. Const. (LIC) approach is 79.63%. Observe that the strategies take less than one minute in all cases. It should also be remarked that the improvement of algorithm of Belotti et al. (2016) provides better final GAPs when Algorithm 2.3.I is used to initialize the big M parameter.

Next, we compare the performance of the three strategies using real-life datasets. The results are reported in Table 2.7 which has the same structure as Table 2.6.

Data	C	Algorithm 2.3.I				Belotti et al. (2016) Big M: Algorithm 2.3.I				Belotti et al. (2016) Big M: Corollary 2.4.1			Ind. Const. (LIC)	
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	t_{st}	t_{total}	GAP	t	GAP
Wpbc	100	**75.03%**	**83.20**	**7283.31**	**61.50%**	57.31%	113.60	7313.72	89.49%	29.02	7229.30	91.18%	7200.30	74.89%
	10	**73.12%**	**91.57**	**7298.90**	**54.19%**	53.52%	137.31	7246.03	85.81%	31.97	7232.50	88.93%	7337.60	79.39%
	1	**62.64%**	**78.19**	**7280.04**	**56.31%**	35.68%	121.74	7243.93	81.73%	30.26	7231.83	84.48%	7322.12	79.00%
	0.1	**61.73%**	**64.19**	**7265.96**	**54.23%**	31.12%	119.18	7255.07	71.00%	49.84	7250.18	73.79%	7319.26	78.06%
	0.01	**63.82%**	**53.60**	**7255.50**	**37.44%**	30.47%	84.93	7231.53	46.73%	30.76	7230.86	55.59%	7285.13	76.30%
SONAR	100	**90.09%**	**162.79**	**173.73**	**0.00%**	79.77%	392.81	7593.58	44.02%	325.01	7525.11	55.63%	511.82	0.00%
	10	**78.74%**	**149.84**	**7350.08**	**17.15%**	64.47%	286.62	7337.49	76.65%	223.61	7423.68	82.16%	7487.33	62.90%
	1	**73.85%**	**119.74**	**7321.56**	**39.72%**	58.61%	319.97	7400.57	75.73%	229.65	7429.91	83.53%	7520.31	75.27%
	0.1	**64.39%**	**115.54**	**7317.44**	**50.75%**	49.44%	197.02	7282.00	74.16%	70.71	7270.97	82.77%	7397.54	82.49%
	0.01	**50.34%**	**147.20**	**7348.83**	**47.62%**	36.81%	206.31	7259.84	66.96%	56.31	7257.78	75.84%	7407.04	88.27%
SPECT	100	**70.21%**	**189.01**	**836.86**	**0.00%**	38.47%	203.93	7404.06	24.99%	16.10	7216.29	29.13%	7200.23	76.40%
	10	70.13%	158.47	7361.31	35.42%	**38.50%**	**183.31**	**7383.41**	**33.16%**	22.40	7222.62	35.03%	7200.29	78.39%
	1	69.89%	150.57	7353.08	51.00%	**38.56%**	**183.86**	**7383.96**	**33.91%**	26.49	7226.53	38.57%	7200.35	79.05%
	0.1	65.31%	170.15	7376.85	36.75%	**34.11%**	**204.98**	**7405.02**	**31.65%**	21.31	7221.50	42.62%	7200.19	79.81%
	0.01	55.10%	95.92	7298.00	53.97%	**14.43%**	**113.45**	**7313.56**	**41.12%**	14.01	7214.08	41.58%	7209.16	80.60%
IONO	100	**88.33%**	**266.56**	**7467.05**	**22.43%**	65.76%	398.20	7598.75	31.49%	118.78	7319.17	33.80%	7200.40	58.52%
	10	**85.44%**	**245.85**	**7446.34**	**35.13%**	61.24%	398.64	7599.30	39.67%	209.33	7409.57	42.83%	7200.96	65.65%
	1	**81.51%**	**228.17**	**7429.80**	**42.81%**	57.00%	306.68	7507.11	43.37%	92.62	7293.17	49.11%	7200.51	78.51%
	0.1	**78.84%**	**207.92**	**7411.75**	**40.79%**	56.07%	255.25	7460.68	47.12%	48.75	7250.88	53.06%	7201.60	83.29%
	0.01	**69.17%**	**214.34**	**7430.09**	**48.28%**	45.62%	258.26	7458.58	50.96%	40.51	7240.53	58.76%	7200.25	90.34%
Wdbc	100	97.51%	227.89	230.00	0.00%	81.18%	289.28	289.28	0.00%	**112.28**	**112.28**	**0.00%**	249.77	0.00%
	10	**98.62%**	**319.62**	**325.70**	**0.00%**	83.09%	412.12	7612.31	27.02%	207.31	7407.50	43.83%	7200.44	34.59%
	1	**103.31%**	**398.20**	**398.62**	**0.00%**	87.79%	439.49	439.49	0.00%	93.88	7296.41	58.05%	7200.28	68.70%
	0.1	**103.59%**	**410.37**	**410.53**	**0.00%**	83.46%	421.18	421.18	0.00%	61.46	7262.07	56.14%	7200.49	87.19%
	0.01	**101.29%**	**403.58**	**403.63**	**0.00%**	71.79%	416.01	416.01	0.00%	61.18	7261.19	57.22%	7200.27	95.10%
WBC	100	**93.37%**	**244.73**	**313.64**	**0.00%**	68.10%	296.78	7497.08	34.13%	46.19	7246.58	51.41%	7200.33	40.99%
	10	**93.65%**	**211.91**	**271.58**	**0.00%**	68.34%	253.61	7453.80	29.22%	53.19	7253.63	43.74%	7200.50	43.27%
	1	**94.49%**	**214.41**	**277.75**	**0.00%**	68.99%	249.87	7450.14	22.38%	51.34	7251.51	38.64%	7200.71	43.56%
	0.1	**99.24%**	**355.86**	**356.35**	**0.00%**	72.64%	360.38	360.37	0.00%	30.73	7230.96	27.19%	7200.31	60.36%
	0.01	105.06%	364.61	364.69	0.00%	70.68%	369.28	369.28	0.00%	**9.47**	**9.47**	**0.00%**	7200.27	80.36%

Table 2.7. *Performance of exact approaches on real data for solving (RL-ℓ_2-M) and (RL-ℓ_2).*

Table 2.7 shows that behavior using real-life datasets is quite similar to simulated datasets, i.e., our strategy performs better in almost all the cases. The Wdbc and WBC datasets are of particular interest as the problem is solved in less than 411 seconds in all the cases. However, four cases remain unsolved after two hours when using the method proposed in Belotti et al. (2016) using the improved big M parameters derived from Algorithm 2.3.I, providing GAPs of between 22.38% and 34.13%. In addition, eight cases remain unsolved after two hours when using the method in Belotti et al. (2016) using Corollary 2.4.1 and nine cases when using the Ind. Const. (LIC). These two last methods provide GAPs of between 27.19% and 95.10%.

Lastly, we analyze three large datasets: Ijcnn1_2000, Ijcnn1_3500, and Ijcnn1_5000. We compare the performance of Variants I and II of Algorithm 2.3 with the performance of Ind. Const. (LIC) and with the resolution method proposed in Belotti et al. (2016). The results of Variant III have been omitted because it performed worse than Variant II in all the cases, i.e., it did not improve the GAP or the overall time.

The results are shown in Table 2.8, which has a similar structure to Table 2.6: the first column shows the name of the dataset and the second column the value of parameter C. The following eleven columns contain information

Data	C	Algorithm 2.3.I				Algorithm 2.3.I v2				Algorithm 2.3.II		
		M's Impr.	t_{st}	t_{total}	GAP	M's Impr.	t_{st}	t_{total}	GAP	t_{st}	t_{total}	GAP
Ijcnn1_2000	100	107.69%	2150.59	2150.61	0.00%	107.31%	1506.21	1506.23	0.00%	25.31	25.61	0.00%
	10	106.74%	5214.58	5214.61	0.00%	103.04%	2032.12	2035.20	0.00%	19.98	21.92	0.00%
	1	104.42%	4492.10	4494.41	0.00%	102.40%	2022.32	2025.60	0.00%	20.55	5431.52	0.00%
	0.1	103.79%	5582.22	5583.06	0.00%	**101.83%**	**3581.91**	**3584.28**	**0.00%**	14.71	7218.11	55.74%
	0.01	76.60%	2970.38	10175.25	67.46%	**76.81%**	**3182.42**	**10388.21**	**67.11%**	15.46	7220.78	73.36%
Ijcnn1_3500	100	97.04%	6990.69	6993.54	0.00%	49.26%	3617.80	3626.25	0.00%	69.50	94.29	0.00%
	10	94.98%	6841.82	7210.65	0.00%	**47.28%**	**2960.29**	**5446.03**	**0.00%**	64.77	7265.28	19.99%
	1	99.98%	33883.80	33989.79	0.00%	**98.81%**	**19193.70**	**21152.66**	**0.00%**	63.84	7264.28	59.93%
	0.1	100.73%	45828.20	45828.87	0.00%	**98.92%**	**28850.40**	**28868.85**	**0.00%**	61.88	7262.03	72.26%
	0.01	78.69%	17913.20	25116.00	70.17%	**78.63%**	**15786.20**	**22988.27**	**70.11%**	45.37	7247.83	79.03%
Ijcnn1_5000	100	97.40%	17403.40	17799.19	0.00%	**50.24%**	**9374.54**	**10840.45**	**0.00%**	117.28	7317.57	9.19%
	10	99.61%	60964.80	64717.57	0.00%	**99.32%**	**44729.00**	**50071.69**	**0.00%**	111.71	7311.93	58.03%
	1	100.13%	88536.30	95737.49	6.75%	**100.19%**	**69876.50**	**71848.85**	**0.00%**	136.72	7337.00	68.23%
	0.1	103.16%	128749.00	128749.59	0.00%	**100.85%**	**78689.90**	**78690.51**	**0.00%**	117.02	7317.40	70.86%
	0.01	81.79%	37206.80	44408.84	68.86%	**81.38%**	**26749.60**	**33951.29**	**68.89%**	67.68	7267.831	82.22%

Data	C	Belotti et al. (2016) Big M: Algorithm 2.3.I v2 T.L.: 7200 s				Belotti et al. (2016) Big M: Corollary 2.4.1 T.L.: 7200 s			Belotti et al. (2016) Big M: Corollary 2.4.1 T.L.: Best variant of Alg. 3			Ind. Const. (LIC) T.L.: 7200 s		Ind. Const. (LIC) T.L.: Best variant of Alg. 3	
		M's Impr.	t_{st}	t_{total}	GAP	t_{st}	t_{total}	GAP	t_{st}	t_{total}	GAP	t_{total}	GAP	t_{total}	GAP
Ijcnn1_2000	100	100.00%	1506.30	1506.30	0.00%	13.17	13.17	0.00%	13.17	13.17	0.00%	**0.19**	**0.00%**	**0.19**	**0.00%**
	10	100.00%	2032.20	2032.20	0.00%	11.28	11.28	0.00%	11.28	11.28	0.00%	**7.61**	**0.00%**	**7.61**	**0.00%**
	1	94.66%	2096.32	2096.32	0.00%	**180.87**	**180.87**	**0.00%**	**180.87**	**180.87**	**0.00%**	7200.91	42.98%	7200.91	42.98%
	0.1	93.87%	3582.38	3585.46	0.00%	1272.02	8472.07	60.63%	1272.02	8472.07	60.63%	7202.76	85.99%	7202.76	85.99%
	0.01	28.90%	4032.29	11232.78	67.84%	245.79	7446.41	79.24%	262.92	10915.31	78.73%	7202.34	95.29%	10845.10	95.26%
Ijcnn1_3500	100	23.07%	3673.52	3673.52	0.00%	**74.68**	**74.68**	**0.00%**	**74.68**	**74.68**	**0.00%**	432.85	0.00%	432.85	0.00%
	10	22.92%	3998.42	11198.63	54.48%	1068.75	8268.93	62.61%	1068.75	8268.93	62.61%	7201.19	53.97%	7201.19	53.97%
	1	90.09%	19649.06	26849.16	29.95%	1657.10	8860.39	69.04%	1666.30	21161.42	65.34%	7202.72	78.18%	21156.60	61.83%
	0.1	93.64%	29455.93	36674.59	7.09%	1487.32	8687.86	73.70%	1395.12	28778.88	73.18%	7201.95	92.10%	28874.30	91.18%
	0.01	36.54%	17059.32	24260.11	77.67%	579.66	7781.42	81.00%	589.97	23164.37	80.92%	7202.58	97.43%	22996.00	97.42%
Ijcnn1_5000	100	21.68%	11457.14	18657.74	56.98%	2438.92	9639.52	62.26%	2456.31	10858.61	60.77%	7201.14	57.52%	10842.30	53.73%
	10	91.37%	45451.16	52651.21	39.32%	3052.90	10270.38	73.14%	3072.17	50094.04	66.06%	7201.44	70.90%	50077.30	65.91%
	1	91.89%	70700.40	77900.52	39.83%	2682.25	9882.31	76.04%	2692.03	71880.28	69.67%	7201.63	84.37%	71856.40	81.66%
	0.1	93.78%	79927.33	87129.11	11.96%	2730.07	9941.87	75.95%	2754.24	78720.34	72.61%	7202.71	94.84%	78708.30	93.17%
	0.01	41.07%	28747.69	35951.31	77.42%	1045.90	8257.49	83.86%	1051.38	33959.29	82.52%	7202.60	97.96%	33958.50	97.52%

Table 2.8. *Performance of exact approaches on Ijcnn for solving (RL-ℓ_2-M) and (RL-ℓ_2).*

about the strategy's performance and the resolution process of Variant I, Variant I version two (v2) and Variant II. Version two of Variant I modifies steps 11-12 of Algorithm 2.3: it only solves the problems for $i \in N$ such that M_i is greater than the median of the M_i values. The next groups of columns show information about the behavior of the Algorithm of Belotti et al. (2016) and the Ind. Const. (LIC) approach. Three blocks of columns provide information about Belotti et al. (2016) method: the first block reports the results of this method using the improved initial big M parameters derived from our strategy; the second block is the same method using Corollary 2.4.1 to initialize the big M and establishing a time limit of two hours; and the last one reports the results of the method using Corollary 2.4.1 to initialize the big M and establishing as time limit the total time of the best performing Variant of Algorithm 2.3 whenever said time is greater than two hours, otherwise it is two hours. Finally, two blocks of columns provide information about the Ind. Const. (LIC) approach: in the first, the time limit is two hours and in the second, it is the total time of the best performing Variant of Algorithm 2.3 if it is greater than two hours or two hours otherwise. When the problem is solved to optimality, the strategy that takes the least time is shown in bold, otherwise, the strategy that provides the best GAP within the time limit is highlighted. The column for the improvement of M in Algorithm 2.3.II has been omitted because it is almost zero in all cases.

Note that our strategy works well even in large datasets. In fact, Algorithm 2.3.I v2 outperforms Algorithm 2.3.I. For example, using Algorithm 2.3.I v2 for the Ijcnn1_2000 dataset, with the value of parameter C being equal to 0.1 and after a preprocessing of 3581.91 seconds, we are able to solve the problem in less than three seconds. When using Belotti et al. (2016) method, this instance can only be solved when the big M parameter is the one derived from the Algorithm 2.3.I v2. When applying the Ind. Const. (LIC) approach, the GAP is 85.99% after two hours. Although the time employed in the strategy is huge in some cases, we are able to solve almost all the problems to optimality. Over the same amount of time, the GAPs provided by the Algorithm of Belotti et al. (2016) using Corollary 2.4.1 and the Ind. Const. (LIC) approach are considerably bigger. For example, when using Algorithm 2.3.I v2 for the Ijcnn1_3500 dataset, with the value of parameter C being equal to 0.1 and after a preprocessing of 28850.40 seconds, we are able to solve the problem in less than twenty seconds. While using the Algorithm of Belotti et al. (2016) using Corollary 2.4.1 or the Ind. Const. (LIC) approach, the GAPs are 73.18% and 92.10%, respectively. We observed that we are able to solve many more problems to optimality when using Algorithm 2.3.I v2 than when using Algorithm 2.3.II. In almost all the cases, the performance of Algorithm 2.3.I v2 is better than the Algorithm of Belotti et al. (2016) and the Ind. Const. (LIC) approach.

To summarize, we tested Algorithm 2.3 in several datasets and in the majority of the cases it performed better than the latest algorithms published in the literature. Therefore, our strategies could be considered as useful tools for the enhancement of the (RL-ℓ_2-M) formulation.

2.6 Concluding remarks

As stated in Duarte Silva (2017), the search for exact solutions of the optimization problems resulting from the ramp loss models with large datasets is an open problem. This chapter has presented various new exact approaches which are applicable to larger datasets and they have proven to be faster than the state-of-the-art algorithms. Unlike other resolution methods in the literature, these exact approaches do not use any auxiliary mixed integer programming model. In fact, the valid inequalites included in the formulation, the theorems that tighten the big M parameters, and the techniques for obtaining bounds on the variables only make use of auxiliary relaxed problems. Such problems allow the improvement of the relaxation of the original model. Consequentially, they enhance the behavior of the branch and bound algorithm.

The results presented in this chapter are published in Baldomero-Naranjo et al. (2020).

The next chapter presents a new SVM-based classifier. This model is an extension of the one presented in this chapter since it limits the influence of outliers and controls the number of selected feature simultaneously.

3 A robust SVM-based approach with feature selection and outliers detection for classification problems

This chapter proposes a robust classification model, based on support vector machines (SVM), which simultaneously deals with outliers detection and feature selection. The classifier is built considering the ramp loss margin error and it includes a budget constraint to limit the number of selected features. The search of this classifier is modeled using a mixed-integer formulation with big M parameters. Two different approaches (exact and heuristic) are proposed to solve the model. The heuristic approach is validated by comparing the quality of its solutions with the ones of the exact approach. In addition, the classifiers obtained with the heuristic method are tested and compared with existing SVM-based models to demonstrate their efficiency.

3.1 Introduction

In this chapter, we introduce a new SVM based model to deal with the two drawbacks of SVM exposed in Chapter 1, namely: the lack of robustness against outliers and feature selection. More concretely, we develop a SVM classifier derived from (RL-ℓ_1) (presented in the previous chapter), which includes feature selection and it is formulated as a mixed integer linear program. In this model we use the ℓ_1-norm, the norm commonly used in the literature for feature selection in SVM based models. Adopting this norm results in classifiers where the number of selected features is lower than using ℓ_2-norm due to its sparse property. Besides, the fact that the proposed model determines the outliers and selects the features simultaneously have some advantages in practical environments with respect to other approaches that do these processes independently.

One advantage of the proposed model is that it prevents the loss of valuable information due to the incorrect removal of elements of the sample. Indeed, it could happen that several of the features of an individual in the sample may take anomalous values. However, it does not make sense to exclude this individual if those features do not affect the classification. For example, consider a dataset with three features, see Figure 3.1.

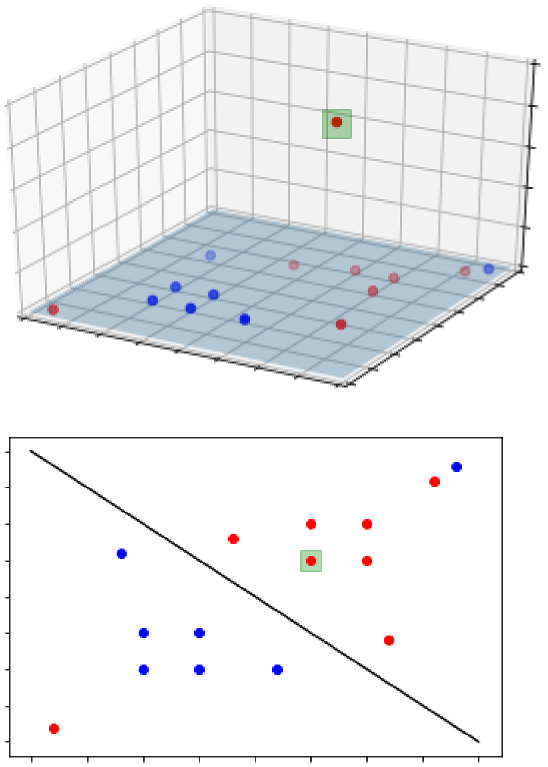


Figure 3.1. *Illustrative example.*

A priori, assuming that all features have influence in the classification process, the point highlighted in green in the first picture of Figure 3.1 is a clear candidate to become an outlier. However, if only the first and the second feature are selected in the feature selection process; clearly, this point is not an outlier. This is particularly striking when dealing with medical data, where usually a large sample is not available.

Conversely, another advantage is that the developed model allows us to identify as outliers data that were not initially classified as such. Indeed, an individual might not be identified as outlier for a general outlier detection procedure because most of its features take similar values to the ones of their class and only some of them take very different values. However, if after the feature selection, these features with very different values are the most relevant in the classification process, this individual should be classified as outlier. Precisely, in the proposed model, this situation is avoided since feature selection and outliers detection processes are carried out simultaneously.

In addition to the introduction of a new model, we develop some strategies to improve the resolution time. Furthermore, we provide a heuristic method that allows us to obtain good quality solutions very quickly. Concretely, we adapt a heuristic approach that has been previously used in different mixed integer linear models (see Guastaroba and Speranza, 2012; Angelelli et al., 2010; Guastaroba et al., 2017). This heuristic is known as Kernel Search (KS) and its main idea is to iteratively provide a better feasible solution to the problem by solving a sequence of restricted MILPs obtained from the original model.

The remainder of this chapter is structured as follows. Section 2.2 introduces the model, presents a valid inequality and develops a procedure to tighten the big M parameters of the formulation. Moreover, in this section, valid values for the big M parameters are computed. Section 3.3 presents a heuristic based on the Kernel Search for solving the proposed model. Section 3.4 contains computational experiments carried out on real-life datasets. In this section, we validate the heuristic algorithm analyzing the best solutions reported by this algorithm and the exact approach. We also test the efficiency of the proposed classifier on real-life datasets comparing its predictive power with other SVM-based models existing in the literature. Our conclusions and some future research topics are included in Section 3.5.

3.2 The model

In this section, we introduce a model based on SVM that tries to eliminate the adverse effects of outliers using a small number of relevant features. Regarding feature selection, this model includes a budget constraint in the formulation to limit the maximum number of features required for classification. This budget constraint has also been used in Maldonado et al. (2014) and Labbé et al. (2019). The model could be slightly modified associating a cost with each feature. Thus, the resulting budget constraint would restrict the cost of classifying a new individual. With respect to the adverse effects of outliers, this model uses the ramp loss margin error introduced by Brooks (2011) to avoid outliers influence. Finally, this model considers the ℓ_1-norm to measure the distances and it is formulated as a mixed integer program with conditional constraints:

$$\begin{aligned}
\min \quad & \sum_{k=1}^{d} \eta_k + C\left(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n} z_i\right) & & \\
\text{s.t.} \quad & (2.1) - (2.3), & & \\
& -l_k v_k \le w_k \le u_k v_k, & k \in D, & \qquad (3.1)\\
& -\eta_k \le w_k \le \eta_k, & k \in D, & \qquad (3.2)\\
& \sum_{k=1}^{d} v_k \le B, & & \qquad (3.3)\\
& v_k \in \{0,1\}, & k \in D, & \qquad (3.4)\\
& \eta_k \ge 0, & k \in D, & \qquad (3.5)
\end{aligned}$$

where η-variables represent the absolute value of the hyperplane coefficients w and v-variables are binary variables that indicate whether the associated feature is selected or not. Observe that constraint (3.3) limits the number of selected features.

This model can be reformulated by expressing each w_k-variable for $k \in D$ as the difference between two non-negative variables w_k^+ and w_k^-. Since $w_k^+ + w_k^-$, for $k \in D$ is part of the objective function, then $|w_k| = w_k^+ + w_k^-$ and, at most, one of the two variables for any $k \in D$ is non-zero in an optimal solution, see Chapter 2. Hence, the model above can be equivalently reformulated as,

$$\begin{aligned}
\text{(RL-FS)} \quad \min \quad & \sum_{k=1}^{d} \left(w_k^+ + w_k^-\right) + C\left(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n} z_i\right) & & \\
\text{s.t.} \quad & (2.2), (2.3), (2.10), (2.11), (3.3), (3.4), & & \\
& w_k^+ \le u_k v_k, & k \in D, & \qquad (3.6)\\
& w_k^- \le l_k v_k, & k \in D. & \qquad (3.7)
\end{aligned}$$

Moreover, (RL-FS) can be linearized substituting constraints (2.10) by constraints (2.12). We will refer to the resulting model as (RL-FS-M). The choice of an appropriate value for M_i, for $i \in N$ is essential to provide efficient solution approaches for the model; i.e., M_i should be big enough so that both models are equivalent, but it should be also as small as possible to provide good linear relaxation and to reduce the computational time for solving this model. In the following proposition, we give a result (similar to Proposition 2.1) that establishes a relationship between ξ-variables and z-variables.

Proposition 3.1. *An optimal solution of (RL-FS), $({w^+}^*, {w^-}^*, b^*, \xi^*, z^*, v^*)$, satisfies condition* (2.5)*.*

The proof can be obtained by contradiction in an easy way, for this reason, it has been omitted. As a consequence of the previous result, the linearized version of condition (2.5) given by (2.6) will be used to strengthen the formulations (RL-FS-M). Henceforth, we will refer to (RL-FS-M)+(2.6) as (RL-FS-M), unless stated otherwise.

In order to solve the model, it is necessary to provide valid values of big M parameters. Model (RL-FS-M) contains two sets of big M parameters: the ones associated with the families of constraints (3.6) and (3.7), and also the ones associated with the set of constraints (2.12). The following subsections will focus on initializing and improving these big M parameters in order to efficiently solve the model.

3.2.1 Initial bounds for the big M parameters

In Brooks (2011), a mixed integer linear model for the classical SVM with ramp loss was introduced. In addition, in Chapter 2 and Baldomero-Naranjo et al. (2020) a result establishing theoretical bounds for the big M parameters appearing in the ramp loss model was proposed. In fact, this result could be adapted for (RL-FS-M) obtaining the following proposition,

Proposition 3.2. *Taking the values M_i, for $i \in N$, such that,*

$$M_i \geq \left(\max_{j \in N} \{ \|x_i - x_j\|_q : y_i = y_j \} \right) \|w\|_p,$$

where $\|\cdot\|_p$ represents the ℓ_p-norm and $\|\cdot\|_q$ its dual, the problems (RL-FS) and (RL-FS-M) are equivalent.

This proposition can be proven analogously to Proposition 2.2 (Baldomero-Naranjo et al., 2020). This result bounds the big M parameters by a value composed by two terms. The first term, $\max_{j \in N} \{ \|x_i - x_j\|_q : y_i = y_j \}$, is easily computed for any norm, but computing the second term, $\|w\|_p$, is not that easy. However, note that if we establish $p = 1$ and we have an upper bound (UB) on the optimal objective value of (RL-FS-M), then UB $\geq \|w\|_1$. Consequently, initial big M parameters for the proposed model can be calculated.

The first step is to compute an upper bound (UB) on the optimal objective value. To do so, we solve the classical SVM model using ℓ_1-norm denoted as (SVM-ℓ_1) and formulated in Chapter 2. From its optimal solution,

$(w^{\mathrm{SVM}}, b^{\mathrm{SVM}}, \xi^{\mathrm{SVM}})$, we build $(\tilde{w}^+, \tilde{w}^-, \tilde{b}, \tilde{\xi}, \tilde{z}, \tilde{v})$, as follows:

$$\tilde{w}_k^+ = \begin{cases} w_k^{\mathrm{SVM}}, & \text{if } w_k^{\mathrm{SVM}} \geq 0, \\ 0, & \text{otherwise,} \end{cases} \text{ for } k \in D,$$

$$\tilde{w}_k^- = \begin{cases} -w_k^{\mathrm{SVM}}, & \text{if } w_k^{\mathrm{SVM}} \leq 0, \\ 0, & \text{otherwise,} \end{cases} \text{ for } k \in D,$$

$$\tilde{\xi}_i = \begin{cases} \xi_i^{\mathrm{SVM}}, & \text{if } \xi_i^{\mathrm{SVM}} \leq 2, \\ 0, & \text{otherwise,} \end{cases} \text{ for } i \in N,$$

$$\tilde{z}_i = \begin{cases} 0, & \text{if } \xi_i^{\mathrm{SVM}} \leq 2, \\ 1, & \text{otherwise,} \end{cases} \text{ for } i \in N,$$

$$\tilde{v}_k = \begin{cases} 0, & \text{if } w_k^{\mathrm{SVM}} = 0, \\ 1, & \text{otherwise,} \end{cases} \text{ for } k \in D,$$

$$\tilde{b} = b^{\mathrm{SVM}}.$$

If constraint (3.3) is fulfilled, a feasible solution of problem (RL-FS-M) is obtained. If not, we sort the components of w^{SVM}-vector in non-decreasing order and we fix to zero the first $n - B$ w-variables. Fixing these variables to zero, we solve the model again (SVM-ℓ_1). From its optimal solution, we build a feasible solution of (RL-FS-M) following the procedure described before, i.e., we update $(\tilde{w}^+, \tilde{w}^-, \tilde{b}, \tilde{\xi}, \tilde{z}, \tilde{v})$. This feasible solution could be improved using the information given by $\tilde{v}$ and $\tilde{z}$ values to obtain the model below. Note that constraints associated with a $\tilde{z}$-value equal to 1 are not considered:

$$\begin{aligned} (\overline{\text{SVM-}\ell_1})_{\tilde{v},\tilde{z}} \min \quad & \sum_{k \in D: \tilde{v}_k = 1} (w_k^+ + w_k^-) + C \left(\sum_{i \in N: \tilde{z}_i = 0} \xi_i \right) \\ \text{s.t.} \quad & y_i \left(\sum_{k \in D: \tilde{v}_k = 1} (w_k^+ - w_k^-) x_{ik} + b \right) \geq 1 - \xi_i, \quad i \in N : \tilde{z}_i = 0, \\ & w_k^+ \geq 0, w_k^- \geq 0, \quad k \in D : \tilde{v}_k = 1, \\ & 0 \leq \xi_i \leq 2, \quad i \in N : \tilde{z}_i = 0. \end{aligned}$$

A feasible solution of (RL-FS-M), $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$, can be obtained considering the solution of the above linear problem $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi})$ together with $\bar{z} = \tilde{z}$ and $\bar{v} = \tilde{v}$. From this feasible solution, we compute an upper bound of the model, named UB. Then, we can establish initial bounds of big M parameters as follows:

$$M_i = \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\} \cdot \text{UB}, \text{ for } i \in N,$$

$$\begin{aligned} u_k &= \text{UB}, \text{ for } k \in D, \\ l_k &= \text{UB}, \text{ for } k \in D. \end{aligned}$$

3.2.2 Improving big M parameters for (RL-FS-M)

In order to tighten the big M parameters for (RL-FS-M), some strategies have been developed. In fact, these strategies are based on solving linear models derived from the original model. We will now detail how each of these parameters could be improved.

3.2.1 Tightening bounds on w_k^+ and w_k^--variables

By solving the next linear model, we are maximizing the value that the sum of w_k^+ and w_k^--variable solutions can have in the feasible region of the original model.

$$\begin{aligned} (\text{UB-}w)\ \max\ & \sum_{k=1}^{d} \left(w_k^+ + w_k^-\right) \\ \text{s.t.}\ \ & (2.2), (2.11), (2.12), (3.3), (3.6), (3.7), \\ & \sum_{k=1}^{d} \left(w_k^+ + w_k^-\right) + C\left(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n} z_i\right) \leq \text{UB}, && (3.8) \\ & 0 \leq v_k \leq 1, \qquad k \in D, && (3.9) \\ & 0 \leq z_i \leq 1, \qquad i \in N. && (3.10) \end{aligned}$$

The optimal objective value of the previous model allows us to obtain an upper bound on $\sum_{k=1}^{d} \left(w_k^+ + w_k^-\right)$. We denote it as UB_w. Then, the following valid inequalities are obtained and added to the formulation:

$$w_k^+ + w_k^- \leq \text{UB}_w, \text{ for } k \in K. \tag{3.11}$$

Observe that this upper bound tightens the big M parameters, by establishing,

$$\begin{aligned} M_i &= \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\} \cdot \text{UB}_w, \text{ for } i \in N, \\ u_k &= \text{UB}_w, \text{ for } k \in D, \\ l_k &= \text{UB}_w, \text{ for } k \in D. \end{aligned}$$

3.2.2 Tightening bounds on b-variable

In a similar way, b-variable can be tightened by solving

$$\begin{aligned} (\text{UB-}b)/(\text{LB-}b)\quad & \max / \min\quad b \\ & \text{s.t.}\qquad (2.2), (2.11), (2.12), (3.3), (3.6) - (3.11). \end{aligned}$$

The solution of this model allows us to obtain a valid lower and upper bound on the b-variable of the model. We include them in the formulation of the problem as the following constraint:

$$\mathrm{LB}_b \leq b \leq \mathrm{UB}_b. \tag{3.12}$$

3.2.3 Tightening bounds on M_i parameters for $i \in N$

This subsection is focused on tightening the bounds of M_i for $i \in N$. Observe that for $z_i = 1$, $i \in N$, constraint (2.12) is equivalent to,

$$M_i \geq 1 - \xi_i - y_i \left(\sum_{k=1}^{d} (w_k^+ - w_k^-) x_{ik} + b \right). \tag{3.13}$$

We introduce the following model in which the right-hand side of (3.13) is maximized over the feasible region of the original model.

$$\begin{array}{lll} (\mathrm{UB}_{M_i}) & \max & 1 - \xi_i - y_i \left(\displaystyle\sum_{k=1}^{d} (w_k^+ - w_k^-) x_{ik} + b \right) \\ & \text{s.t.} & (2.2), (2.11), (2.12), (3.3), (3.6) - (3.12). \end{array}$$

For each $i \in N$, the objective value of this model provides a tightened upper bound on M_i. However, in datasets that contain a huge number of individuals, this strategy is computationally demanding. For this reason, we propose the following variant.

i) For all $i \in N$, when $y_i = +1$, we establish M_i as the optimal objective value of the following linear problem:

$$\begin{array}{lll} (\mathrm{UB}_{M_+}) & \max & 1 - \left(\displaystyle\sum_{k=1}^{d} w_k^+ \underline{x}_{+k} - \sum_{k=1}^{d} w_k^- \bar{x}_{+k} + b \right) \\ & \text{s.t.} & (2.2), (2.11), (2.12), (3.3), (3.6) - (3.12), \end{array}$$

where $\underline{x}_{+k} = \min\limits_{i \in N.} \{x_{ik} : y_i = +1\}$ and $\bar{x}_{+k} = \max\limits_{i \in N.} \{x_{ik} : y_i = +1\}$, for $k \in D$.

ii) For all $i \in N$, when $y_i = -1$, we establish M_i as the optimal objective value of the following linear problem:

$$\begin{array}{lll} (\mathrm{UB}_{M_-}) & \max & 1 + \left(\displaystyle\sum_{k=1}^{d} w_k^+ \bar{x}_{-k} - \sum_{k=1}^{d} w_k^- \underline{x}_{-k} + b \right) \\ & \text{s.t.} & (2.2), (2.11), (2.12), (3.3), (3.6) - (3.12), \end{array}$$

where $\bar{x}_{-k} = \max\limits_{i \in N.} \{x_{ik} : y_i = -1\}$ and $\underline{x}_{-k} = \min\limits_{i \in N} \{x_{ik} : y_i = -1\}$, for $k \in D$.

The strategy to initialize and tighten the bounds on the big M parameters of the model is summarized in Algorithm 3.1 which is described as a pseudocode.

Algorithm 3.1: Variant 1 and 2. Computation of big M parameters.

```
Data: Training sample composed by a set of $n$ individuals with $d$ features.
Result: Updated values of $M_i, u_k$, and $l_k$, for $i \in N$ and $k \in D$.
1  Solve the problem (SVM-$\ell_1$). From its optimal solution, build and
     solve $(\overline{\text{SVM-}\ell_1})_{\bar{v},\bar{z}}$. From its solution build a feasible solution of
     (RL-FS-M) and obtain an upper bound (UB).
2  for $i \in N$ do  $M_i = \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\} \cdot \text{UB}.$;
3  for $k \in D$ do  $u_k = \text{UB}, \quad l_k = \text{UB}.$;
4  Solve (UB-$w$). Let $\text{UB}_w$ be the optimal objective value of this
     problem.
5  Update $M_i = \max_{j \in N} \{\|x_i - x_j\|_\infty : y_i = y_j\} \cdot \text{UB}_w, u_k = \text{UB}_w$, and
     $l_k = \text{UB}_w$. Add the obtained bounds to the formulation (RL-FS-M)
     including the set of constraints (3.11).
6  Obtain $\text{LB}_b$ and $\text{UB}_b$, lower and upper bounds respectively, of the
     $b$-variable by solving (LB-$b$) and (UB-$b$).
7  Include the constraint (3.12) in the formulation (RL-FS-M).
8  do
9      Repeat Steps 6 and 7 including constraints (3.11) and (3.12) in
         model (UB-$w$).
10     case Variant I do
11         for $i \in N$ do
12             Update $M_i$ as the optimal value of the problem ($\text{UB}_{M_i}$).
13     case Variant II do
14         For $i \in N$, when $y_i = 1$, update $M_i$ as the optimal value of the
             problem $\left(\text{UB}_{M_+}\right)$.
15         For $i \in N$, when $y_i = -1$, update $M_i$ as the optimal value of
             the problem $\left(\text{UB}_{M_-}\right)$.
16 while an improvement of the bounds is obtained;
```

To summarize, in this section we have introduced a model based on SVM using the ℓ_1-norm that includes feature selection and avoids the effect of outliers. For this model, we have developed a mixed integer linear formulation (RL-FS-M). In addition, we have introduced a methodology for initializing and tightening the big M parameters appearing in the model. In the next section, we present a heuristic approach to obtain adequate feasible solutions for the model.

3.3 Heuristic approach

In the previous section, some strategies to tighten big M parameters of (RL-FS-M) have been presented. Consequently, the formulation is strengthened by the obtained bounds easing its exact solution approach. However, the exact solution of this model cannot be obtained in reasonable times for large datasets. For this reason, it is necessary to develop a heuristic solution method to deal with such datasets.

In this section, we develop a procedure for obtaining good quality solutions. It is based on the Adaptive Kernel Search (AKS) proposed by Guastaroba et al. (2017). The idea of the AKS is to solve a sequence of mixed integer problems derived from the original one, where the majority of the binary variables are fixed and only some of them are considered as integer – this set of integer variables is called Kernel.

The heuristic approach that we propose for (RL-FS-M) has two main characteristics. On the one hand, we introduce an AKS for v-variables, see Labbé et al. (2019). On the other hand, the z-variables are dynamically fixed using the solution of the previous sub-problems. This strategy considers sub-problems where the number of binary variables and constraints containing the big M parameter are reduced with respect to the original problem. As a consequence, these sub-problems have a shorter solution time. This heuristic method will be referred to as Dynamic Adaptive Kernel Search (DAKS). DAKS allows us to obtain adequate feasible solutions of (RL-FS-M) in datasets of large size, as can be appreciated in Section 3.4. The improvement is justified by the reduction in the number of integer variables and big M constraints in the sub-problems considered in this procedure.

3.3.1 Dynamic Adaptive Kernel Search (DAKS)

The DAKS algorithm has three phases.

- The first one computes an initial feasible solution to the problem. Using this solution, the big M parameters appearing in the model are initialized and some z-variables are fixed.
- The second one determines the initial set of integer v-variables; i.e., the initial kernel. Furthermore, in this step, the remaining v-variables are ranked. Note that the most promising variables to take a value of 1 in the optimal solution are considered first. The chosen order has a huge influence on the solution obtained by the heuristic, so this

order should be adapted to the dataset under study. Observe that although in the original algorithm several sets of integer variables can be considered, in this version, we only apply the Adaptive Kernel Search over v-variables, i.e., z-variables are adjusted using a different methodology during the procedure.

- In the third phase, a sequence of restricted MILP derived from the original problem are solved, including constraints that ensure that a progressively better bound on the solution is obtained. The solution and the computational effort required to solve the previously restricted MILP guide the construction of the subsequent Kernels. In each iteration, the Kernel is updated to test other promising variables and to remove useless ones. Moreover, taking into account the solution of the previous sub-problem, each z-variable is fixed to zero, fixed to one, or considered as a binary variable, i.e., this set of variables is updated dynamically. After some iterations, the second phase is repeated with the objective of reordering v-variables using the information of the current z-variables values.

Given $\mathcal{K} \subseteq D$, the restricted MILP derived from the original problem will be referred to as (RL-FS-M)$(\mathcal{K})_{\hat{z}}$, where if $k \in \mathcal{K}$, v_k-variable is considered a binary variable, it is otherwise fixed to zero. Furthermore, with the objective of simplifying the notation of these sub-problems, we include an auxiliary vector $\hat{z}$ which indicates the corresponding z-variables values. Each element of vector $\hat{z}$ can take one of the following values with the following meanings:

$$\hat{z}_i = \begin{cases} 0, & \text{it indicates that } z_i \text{ is fixed to 0 in the sub-problem,} \\ 1, & \text{it indicates that } z_i \text{ is fixed to 1 in the sub-problem,} \\ 2, & \text{it indicates that } z_i \text{ is a binary variable in the sub-problem.} \end{cases}$$

Therefore, (RL-FS-M)$(\mathcal{K})_{\hat{z}}$ model is formulated as described below.

$$\begin{aligned} \min \quad & \sum_{k\in\mathcal{K}} \left(w_k^+ + w_k^-\right) + C\left(\sum_{i\in N:\hat{z}_i\neq 1} \xi_i + 2\sum_{i\in N:\hat{z}_i=2} z_i + 2\sum_{i\in N:\hat{z}_i=1} 1\right) \\ \text{s.t.} \quad & y_i\left(\sum_{k\in\mathcal{K}} \left(w_k^+ - w_k^-\right) x_{ik} + b\right) \geq 1 - \xi_i - M_i z_i, \quad && i \in N : \hat{z}_i = 2, \\ & y_i\left(\sum_{k\in\mathcal{K}} \left(w_k^+ - w_k^-\right) x_{ik} + b\right) \geq 1 - \xi_i, && i \in N : \hat{z}_i = 0, \\ & \xi_i \leq 2(1 - z_i), && i \in N : \hat{z}_i = 2, \\ & \sum_{k\in\mathcal{K}} v_k \leq B, \end{aligned}$$

$$
\begin{aligned}
&0 \le w_k^+ \le u_k v_k, && k \in \mathcal{K},\\
&0 \le w_k^- \le l_k v_k, && k \in \mathcal{K},\\
&v_k \in \{0,1\}, && k \in \mathcal{K},\\
&0 \le \xi_i \le 2, && i \in N : \hat{z}_i \neq 1,\\
&z_i \in \{0,1\}, && i \in N : \hat{z}_i = 2.
\end{aligned}
$$

In the next subsections, the three main phases of DAKS procedure are described in detail and a pseudocode of this heuristic approach is also presented in Algorithm 3.2.

3.3.1 Initial phase

In the first phase, Algorithm 3.1 is applied with the aim of obtaining an initial upper bound of the problem and valid values for the big M parameters. We can set a time limit for the algorithm or establish a fixed number of iterations. The modeler should decide the best strategy to follow for the dataset under study and choose one of the variants of this algorithm, taking into account the number of individuals and features considered in the dataset.

Furthermore, in this phase, z-variables are fixed. Let $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$ be the initial solution built in Algorithm 3.1. Then, vector $\hat{z}$ is determined as follows,

$$
\hat{z_i} = \begin{cases} 0, & \text{if } \bar{z}_i = 0 \text{ and } \bar{\xi}_i < 1,\\ 1, & \text{if } \bar{z}_i = 1,\\ 2, & \text{if } \bar{z}_i = 0 \text{ and } \bar{\xi}_i \ge 1, \end{cases}
$$

for $i \in N$. Note that if $\hat{z}_i$ is established as two, then z_i will be a binary variable in (RL-FS-M)$(\mathcal{K})_{\hat{z}}$.

The idea of this procedure is to discard as outliers the well-classified individuals in the initial solution, i.e., $\hat{z}_i$ is fixed to zero. Furthermore, the individuals whose corresponding $\bar{z}_i$-values are one in the initial solution are set as outliers, i.e., $\hat{z}_i$ is fixed to one. Finally, we establish as "possible outliers" (considering the associated z-variable as binary) the wrong-classified individuals corresponding to $\bar{\xi}$-values in the interval [1,2].

3.3.2 Second phase

In the second phase, we will create an initial kernel on v-variables and the remaining variables will be ordered according to how likely these variables will take a value of one in the optimal solution. This stage is essential for the success of the algorithm.

We propose the following strategy for ordering the variables. First, we solve the problem (RL-FS-M)$(D)_{\hat{z}}$ considering v-variables as continuous. We will denote the previously described model as $(\text{Rel}_v\text{-RL-FS-M})(D)_{\hat{z}}$. Observe that in this relaxed problem, only the z-variables whose corresponding $\hat{z}$ take value 2 are binary (recall that $\hat{z}$ was computed in the initial phase). The optimal objective value of this problem will be called $\text{LB}_{\hat{z}}$. Again, we solve it by fixing the values of the binary variables in order to obtain the reduced costs.

Let $\breve{w}_k^+$ and $\breve{w}_k^-$ be the optimal values of variables w_k^+ and w_k^- and r_k^+ and r_k^- the corresponding reduced cost of these variables. Hence, the variables are ordered in non-decreasing order with respect to vector r, which is computed as:

$$r_k = \begin{cases} -(\breve{w}_k^+ + \breve{w}_k^-), & \text{if } \breve{w}_k^+ + \breve{w}_k^- > 0, \\ \min\{r_k^+, r_k^-\}, & \text{otherwise.} \end{cases}$$

The initial kernel, $\mathcal{K}_0$, is composed by $k \in D$ such that w_k^+ or w_k^- variable takes a positive value in the solution of $(\text{Rel}_v\text{-RL-FS-M})(D)_{\hat{z}}$ model considering z-variables fixed to their optimal values in $(\text{Rel}_v\text{-RL-FS-M})(D)_{\hat{z}}$. Moreover, the first time that this phase is executed, $\mathcal{K}_0$ includes the indexes $k \in D$ such that w_k^+ or w_k^- variables take a positive value in the initial solution built in the first step of Algorithm 3.1.

Once this initial kernel $\mathcal{K}_0$ is defined, the model $(\text{RL-FS-M})(\mathcal{K}_0)_{\hat{z}}$ is solved. Its solution is an upper bound (UB) on (RL-FS-M). Let $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$ be the solution of the problem, i.e., the solution of the current upper bound. After this step, vector $\hat{z}$ is updated as follows:

- If $\hat{z}_i$ is equal to zero and $\bar{\xi}_i > 1$, for $i \in N$, (i.e., the individual was discarded as outlier, but in the current solution it is wrongly classified), $\hat{z}_i$ will be fixed to two (i.e., it will be set as a "possible outlier").
- If $\hat{z}_i$ is equal to one and $y_i \left(\sum_{k \in D} \left(\bar{w}_k^+ - \bar{w}_k^-\right) x_{ik} + \bar{b}\right) \geq 0$, for $i \in N$, (i.e., the individual was set as an outlier, but in the current solution it is well classified), $\hat{z}_i$ is updated to two (i.e., it will be set as a "possible outlier").

Note that the criterion of being a "possible outlier" can be adjusted to the dataset, imposing tighter conditions, e.g. $\bar{\xi}_i > 0$, or relaxing them, e.g. $\bar{\xi}_i > 1.5$.

3.3.3 Third phase

An iterative procedure starts in this phase. In each iteration, (it), a new set of indexes, named $B_{it} \subseteq D$ is added to the kernel. These indexes are included in the order determined by vector r, i.e., the most promising v-variables to take value one in the optimal solution are considered first. Note that each index is included at most once. The initial size of B_{it} is the size of $\mathcal{K}_0$, see Guastaroba and Speranza (2012); Labbé et al. (2019). In the subsequent iterations, the size of B_{it} will be adjusted.

The model (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$ is solved adding the following constraints:

$$\sum_{k \in \mathcal{K} \cup B_{it}} (w_k^+ + w_k^-) + C \left(\sum_{i=1}^{n} \xi_i + 2 \sum_{i=1}^{n} z_i \right) \leq \text{UB}, \tag{3.14}$$

$$\sum_{k \in B_{it}} v_k \geq 1. \tag{3.15}$$

Constraint (3.14) ensures that a better upper bound of the problem is obtained, and constraint (3.15) ensures that at least one variable is selected for the new B_{it} set. If the previous iteration found a feasible solution but the optimality of this solution was not proved because the time limit was reached, constraint (3.15) will be replaced by:

$$\sum_{k \in B_{it} \cup \{k \in \mathcal{K} : \bar{v}_k = 0\}} v_k \geq 1, \tag{3.16}$$

where $\bar{v}_k$ represents the solution associated with the current upper bound.

A feasible solution of this sub-problem, i.e., (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$+(3.14)+(3.15) if the previous solution was optimal or (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$ + (3.14) + (3.16) if the previous solution was feasible, will improve the upper bound. However, this problem may not be feasible and for this reason, a time limit t_{limit} is imposed. Furthermore, we also add a time limit t_{fea} for finding a feasible solution, i.e, the sub-problem is stopped if no feasible solution is found within t_{fea}. In addition, as the big M parameters affect the lower bounds of the problem, proving optimality is hard. Consequently, if the incumbent solution is not improved after a fixed time t_{inc}, the resolution of the sub-problem is stopped.

After this, the kernel is updated: the selected variables of the set B_{it} are added, denoted as $\mathcal{K}^+$, and the variables of the kernel that were not selected in the last p iterations are removed from the kernel, denoted as $\mathcal{K}^-$. Moreover, vector $\hat{z}$ is updated as follows:

Previous value of vector $\hat{z}$	Previous and current solutions	Updated value of vector $\hat{z}$
$\hat{z}_i$=0	Current solution $\bar{\xi}_i \geq 1$	$\hat{z}_i$=2
$\hat{z}_i$=1	$y_i \left(\sum_{k \in D} \left(\bar{w}_k^+ - \bar{w}_k^- \right) x_{ik} + \bar{b} \right) \geq 0$	$\hat{z}_i$=2
$\hat{z}_i$=2	Last q solutions $\bar{z}_i = 1$	$\hat{z}_i$=1
	Last q solutions $\bar{z}_i = 0$	$\hat{z}_i$=0

Table 3.1. *Procedure to update vector $\hat{z}$.*

The idea of the previous procedure is: if the individual was discarded as outlier but in the current solution it is wrongly classified, it will be set as a "possible outlier". If the individual was set as outlier, but in the current solution it is well classified, it will be set as a "possible outlier". Finally, if the individual was set as "possible outlier" but it has the same value in the last q solutions, the variable will be fixed to this value. Observe that the criterion of being a "possible outlier" can be adjusted to the dataset, imposing tighter conditions or relaxing them, as explained at the end of Section 3.3.2.

In the following iterations, the problem (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$ is solved. The time employed in solving the previous iteration determines the size of $\mathcal{K} \cup B_{it}$, $S_{K \cup B_{it}}$. If the problem related to an iteration is easily solved, i.e., if $t_{K \cup B_{it}} \leq t_{Easy}$, the number of v-variables that are binary in the next step is incremented, i.e., $S_{K \cup B_{it+1}} = (1+\delta) S_{K \cup B_{it}}$, where $0 \leq \delta \leq 1$.

This iterative process (phase 3) will be stopped if any of the following situations occur:

i) If $\text{LB}_{\hat{z}} = \text{UB}$ and $\hat{z}$ remains unchanged, the algorithm finishes. If $\text{LB}_{\hat{z}} = \text{UB}$, we know that the obtained solution cannot be improved having fixed z-variables to these values. Since different $\hat{z}$-values to the current ones are not proposed for the next iteration, the algorithm finishes. Note that $\text{LB}_{\hat{z}}$ is an upper bound of the lower bound (LB) of the problem. Hence, the solution could be a local minimum or the optimal solution of the problem.

ii) If $\text{LB}_{\hat{z}} = \text{UB}$, the current $\hat{z}$ is the same as the one used to compute $\text{LB}_{\hat{z}}$, and different values are proposed for the next iteration to update $\hat{z}$ in the procedure described in Table 3.1; return to phase two. If this situation happens, our solution cannot be improved if z-variables are fixed to these values. However, it may improve if z-variables are fixed to different values. In order to recompute the initial kernel and sort the v-variables considering the updated values of $\hat{z}$, the algorithm returns to phase two.

iii) After a criterion defined by the user (e.g. doing a fixed number of iterations, establishing a time limit or fixing a percentage of v-variables that should be analyzed), the algorithm returns to phase two. The objective is to sort the v-variable taking into account the current $\hat{z}$.

Note that the number of iterations that the algorithm does in Phase 2 and 3 is a criterion determined by the user that should be adapted to the dataset. Similarly, the proposed heuristic has many parameters $p, q, \delta, t_{limit}, t_{Easy}, \ldots$ that the user should adapt to the dataset and their needs, finding the desired balance between time employed and required precision. In Algorithm 3.2, this procedure is summarized and described as a pseudocode.

In conclusion, in this section we have proposed a new algorithm to obtain accurate feasible solutions of our model based on the kernel search algorithm. In particular, we have developed a different strategy to seek the most promising values of z-variables with the objective of avoiding using the reduced costs of these variables due to the high influence that the big M parameters have over them.

3.4 Computational Experiments

In this section, we set out the results of several computational experiments. Firstly, we analyze the performance of Algorithm 3.2 comparing the solution provided using this algorithm to the solution obtained applying an exact resolution method. The results demonstrate that the proposed algorithm computes high-quality solutions. Secondly, we present a comparison of the classification done by several models over real-life datasets showing the efficiency of the proposed classifier.

The experiments were conducted on an Intel(R) Xeon(R) W-2135 CPU 3.70 GHz 32 GB RAM, using CPLEX 12.9.0. in Concert Technology C++. As seen in Belotti et al. (2016) due to the presence of big M parameters, the relative MIP tolerance and the integrality tolerance were fixed to zero. The remaining parameters were left to their default values unless stated otherwise.

3.4.1 Data

The computational experiments were carried out on real-life datasets. They are specified in Table 3.2, where n is the number of individuals, d is the number of features, and the last column reports the percentage of elements in each class. Leukemia dataset is from Golub et al. (1999), Colon dataset is from Alon et al. (1999), and DLBCL dataset is from Shipp et al. (2002). All other datasets are from the UCI repository (see Lichman, 2013). Observe that these datasets were previously used to analyze the performance of models based on SVM (see, for instance, Brooks, 2011; Labbé et al., 2019; Maldonado et al., 2014) and some of these datasets were also used in Section 2.5 to compare the performance of the exact resolution methods proposed in Chapter 2.

In these datasets, the following values for the parameter C will be analyzed, $C \in \{0.01, 0.1, 1, 10, 100\}$, as in Brooks (2011). We tested five different values for the parameter B, which is different for each dataset. To choose it, we solved (RL-FS-M) on ten randomly generated samples containing 90% of individuals from the original dataset being B equal to the number of features. Note that in each sample, the percentage of elements from each class is the same as in the original dataset. We compute the average from the number of selected features for each C. The maximum average is the greatest B analyzed, the other four are proportional to it, being $\frac{2}{3}$, $\frac{1}{2}$, $\frac{1}{3}$, and $\frac{1}{5}$ of the greatest value of B. In Table 3.3, these values are depicted. Moreover, since the two greatest values of B for the Arrhythmia dataset (161 and 107) and Mfeat

Algorithm 3.2: Dynamic Adaptive Kernel Search (DAKS).

```
   Data: Training sample ($n$ individuals with $d$ features).
   Result: Heuristic solution of (RL-FS-M).
1  Initialize big M parameters applying Algorithm 3.1. Let
     $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$ be the initial solution built in this Algorithm.
2  for $i \in N$ do
3      if $\bar{z}_i = 1$ then $\hat{z}_i$=1;
4      if $\bar{z}_i = 0$ and $\bar{\xi}_i > 1$ then $\hat{z}_i$=2;
5      if $\bar{z}_i = 0$ and $\bar{\xi}_i \leq 1$ then $\hat{z}_i$=0;
6  do
7      Solve (Rel$_v$-RL-FS-M)$(D)_{\hat{z}}$. Solve it again fixing the values of the
         binary $z$-variables. Sort $v$-variables in non-decreasing order with
         respect to vector $r$.
8      Constitute initial $\mathcal{K} = \mathcal{K}_0$. Solve (RL-FS-M)$(\mathcal{K})_{\hat{z}}$. Let
         $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$ be the solution of the current UB.
9      for $i \in N$ do
10         if ($\hat{z}_i = 0$ and $\bar{\xi}_i > 1$) or ($\hat{z}_i$=1 and
             $y_i \left(\sum_{k \in D} \left(\bar{w}_k^+ - \bar{w}_k^-\right) x_{ik} + \bar{b}\right) \geq 0$) then $\hat{z}_i$=2;
11     $it = 0$
12     do
13         if $it > 0$ and the solution of $it - 1$ was feasible but not optimal
             then
14             Let $B_{it}$ be the new set of promising $v$-variables related to
                 iteration $it$.
15             Solve (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$ + (3.14) + (3.16), where $B_{it}$ is
                 the set of promising $v$-variables associated with iteration
                 (iter.) $it$.
16         else
17             Solve (RL-FS-M)$(\mathcal{K} \cup B_{it})_{\hat{z}}$ + (3.14) + (3.15).
18         if an optimal or feasible solution is obtained then
19             Update $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$ and UB.
20             Let:
               $\mathcal{K}^+ := \{k \in B_{it} : \bar{v}_k = 1\}$.
               $\mathcal{K}^- := \{k \in \mathcal{K} : k$ has not been selected in the last $p$ feasible iter.$\}$.
               Update $\mathcal{K} = \mathcal{K} \cup \mathcal{K}^+ \setminus \mathcal{K}^-$ and $\hat{z}$ as described in Table 3.1.
21         if $LB_{\hat{z}} = UB$ and $\hat{z}$ remains unchanged then Go to Line 26.;
22         if $LB_{\hat{z}} = UB$ and $\hat{z}$ is updated then Go to Line 6.;
23         $it = it + 1$.
24     while a criterion is fulfilled (e.g. number of iterations, time
         limit);
25 while a criterion is fulfilled (e.g. number of iterations, time limit);
26 return $(\bar{w}^+, \bar{w}^-, \bar{b}, \bar{\xi}, \bar{v}, \bar{z})$.
```

Label	Name in repository	n	d	Class(%)
Colon	Analysis of tumor and normal colon tissues	62	2000	35/65
Leukemia	Gene expression measurements on leukemia patients	72	5327	47/53
DLBCL	Measurements from biopsies of Diffuse Large B-cell Lymphoma patients	77	7129	75/25
SONAR	Connectionist bench (sonar, mines vs. rocks)	208	60	54/46
IONO	Ionosphere	351	33	64/36
Arrhythmia	Cardiac arrhythmia	420	258	57/43
Wdbc	Breast cancer Wisconsin (diagnostic)	569	30	63/37
Mfeat	Multiple Features - Handwritten numerals	2000	649	10/90
Lepiota	Mushroom - Species of mushrooms in the Agaricus and Lepiota Family	8124	109	52/48

Table 3.2. *Real-life datasets*

dataset (404 and 269) computed by the above description are too big, we have removed them and we have included the following values of B: 16 and 8 for Arrhythmia dataset; 30 and 10 for Mfeat dataset.

Label	B
Colon	$\{34, 23, 17, 11, 7\}$
Leukemia	$\{48, 32, 24, 16, 10\}$
DLBCL	$\{44, 29, 22, 15, 9\}$
SONAR	$\{57, 38, 29, 19, 11\}$
IONO	$\{33, 22, 17, 11, 7\}$
Arrhythmia	$\{81, 54, 32, 16, 8\}$
Wdbc	$\{29, 19, 15, 10, 6\}$
Mfeat	$\{202, 135, 81, 30, 10\}$
Lepiota	$\{33, 22, 17, 11, 7\}$

Table 3.3. *Values of parameter B.*

3.4.2 Validation of DAKS algorithm

In this section, we validate the efficiency of Algorithm 3.2, i.e, we compare the solution obtained by the exact method with the one provided by Algorithm 3.2. The exact method consists of applying Algorithm 3.1 and solving the resulting formulation using CPLEX. The resolution of this model is quite hard, and for this reason, a time limit of 7200 seconds has been set.

We solved (RL-FS-M) for all datasets and the following C values:

$$C \in \{0.01, 0.1, 1, 10, 100\},$$

and the smallest value of B tested, i.e., Colon $B = 7$, Leukemia $B = 10$, DLBCL $B = 9$, SONAR $B = 11$, IONO $B = 7$, Arrhythmia $B = 8$, Wdbc $B = 6$, Mfeat $B = 10$, and Lepiota $B = 7$. For the Mfeat and Lepiota datasets, we applied Variant 2 of Algorithm 3.1 to initialize the big M parameters, while Variant 1 was used in the rest of the cases.

As done in Guastaroba et al. (2017) for the AKS algorithm, parameter δ was set to 0.35 in DAKS. Moreover, we established $p = 2$, $q = 2$, $t_{Easy} = 10$, $t_{fea} = 120$, and $t_{inc} = 160$. Furthermore, in the restricted problems, we limited the computational time (t_{limit}) to 400 seconds. In order to improve the behavior, as in Guastaroba et al. (2017), we used CPLEX with default values, except for the following parameters: BndStrenInd=1, MIPEmphasis=HiddenFeas, FPHeur=1. The parameters previously mentioned to avoid the negative effect of big M values were also used.

The results are depicted in Table 3.4. The first column indicates the name of the dataset and the second column the value of parameter C. The next column reports the total time of the exact solution method in seconds, i.e., the strategy time plus the solving time. The following column contains information about the MIP relative GAP reported by CPLEX. The fifth column indicates the time in seconds that the heuristic took. Finally, the sixth column reports the percentage difference between the best solution found by heuristic algorithm (BS_h) and the best solution found by the exact solution method (BS_e). This difference is computed as follows:

$$\%\ \mathrm{BS} = \frac{\mathrm{BS}_h - \mathrm{BS}_e}{\mathrm{BS}_e}.$$

Data	C	t_e	GAP	t_h	%BS
Colon	100	7225.01	23.59%	49.06	0.00%
	10	7219.15	21.95%	34.66	2.40%
	1	7218.79	5.35%	15.26	1.54%
	0.1	2856.23	0.00%	6.69	0.00%
	0.01	1.25	0.00%	1.24	0.00%
Leukemia	100	7230.38	11.45%	120.14	3.23%
	10	7241.98	12.77%	101.67	2.05%
	1	7281.67	14.82%	92.79	-1.90%
	0.1	41.10	0.00%	43.90	0.00%
	0.01	4.75	0.00%	4.71	0.00%
DLBCL	100	7250.71	10.42%	73.15	0.37%
	10	7266.44	10.76%	82.42	0.37%
	1	7326.72	10.35%	113.93	2.33%
	0.1	2437.37	0.00%	58.74	0.00%
	0.01	7.10	0.00%	7.36	0.00%
SONAR	100	7217.41	94.46%	302.70	4.83%
	10	7211.48	90.50%	202.88	1.91%
	1	7205.42	59.18%	578.35	0.56%
	0.1	7207.64	55.09%	1487.38	0.00%
	0.01	7207.57	1.02%	2.76	0.00%
IONO	100	7222.98	84.66%	77.35	-8.55%
	10	7226.58	72.06%	173.36	0.00%
	1	7237.98	63.71%	117.58	-3.07%
	0.1	7221.80	46.56%	305.99	-0.07%
	0.01	7222.49	12.13%	378.45	0.00%

Data	C	t_e	GAP	t_h	%BS
Arrhythmia	100	7244.87	98.89%	1844.90	-10.00%
	10	7268.66	98.38%	1860.49	-8.91%
	1	7246.76	89.60%	1844.65	-1.91%
	0.1	7255.37	83.78%	1832.81	0.19%
	0.01	7258.67	59.64%	537.19	0.00%
Wdbc	100	1954.27	0.00%	148.65	1.92%
	10	204.05	0.00%	44.96	0.00%
	1	41.17	0.00%	39.31	0.00%
	0.1	29.74	0.00%	29.17	0.00%
	0.01	7221.52	16.27%	378.81	0.00%
Mfeat	100	7223.05	9.31%	131.33	2.21%
	10	7218.46	8.04%	96.09	2.21%
	1	7228.61	7.75%	89.22	0.80%
	0.1	1060.93	0.00%	70.27	0.00%
	0.01	7233.85	57.84%	101.85	0.00%
Lepiota	100	63.53	0.00%	26.52	0.00%
	10	41.76	0.00%	29.15	0.00%
	1	55.67	0.00%	30.75	0.00%
	0.1	7249.02	11.64%	58.62	0.00%
	0.01	7403.56	1.75%	276.12	0.00%

Table 3.4. *Comparison between the exact resolution method and the heuristic procedure.*

As can be appreciated in Table 3.4, the proposed heuristic resolution method reported the same solution as the exact solution method in most cases, with the advantage that the heuristic algorithm is much less time-consuming than the exact resolution method. Note also, that in some cases, Algorithm 3.2 found a better solution than the exact solution method. For instance, in the case of Leukemia dataset with $C = 1$, Algorithm 3.2 found a 1.9% better solution in one minute and a half than the exact solution method found in over two hours. Besides, in the case of IONO dataset with $C = 100$, Algorithm 3.2 found a 8.55% better solution in less than two minutes than the exact solution method found in over two hours. Similarly, in the case of Arrhythmia dataset with $C = 100$ and $C = 10$, Algorithm 3.2 found 10% and 8.91% better solutions respectively than the exact solution method found in over two hours.

In summary, we compared the performance of Algorithm 3.2 with an exact solution approach for a challenging B parameter value. We proved that DAKS provides quite good solutions in the 45 cases analyzed for different C parameter values. On average, the difference between the best solution found by the exact solution method and the heuristic is -0.17%, while the heuristic has significantly reduced the amount of time required.

3.4.3 Model validation

The aim of this section is to compare the performance of the proposed classifier (RL-FS-M) with other well-known methods based on SVM in the literature. We focus our attention on efficient models that deal with outliers detection (RL-ℓ_1-M) or feature selection: (FS-SVM), (Fisher-SVM), and (RFE-SVM). More concretely, the purpose of (RL-ℓ_1-M) is to avoid the influence of outliers, see Brooks (2011). We used the methodology proposed in Chapter 2 (Baldomero-Naranjo et al., 2020) to solve this model, imposing 1800 seconds of time limit. On the other hand, the objective of (FS-SVM) is to limit the number of features selected by the classifier and it was solved by applying the heuristic algorithm proposed in Labbé et al. (2019). Similarly, the goal of Fisher Criterion Score with SVM (Fisher-SVM) and Recursive Feature Elimination algorithm with SVM (RFE-SVM) is also restricting the number of features selected. They were introduced in Guyon et al. (2006, 2002), respectively. Finally, (RL-FS-M) was also compared with the classical (SVM-ℓ_1).

To compare the classifiers, we used the recognized classification performance metrics: the accuracy (ACC) and the area under the curve (AUC). The accuracy is computed as

$$\text{ACC} = \frac{TP + TN}{TP + TN + FP + FN},$$

where TP are true positives, TN are true negatives, FP false positives and FN false negatives. The area under the curve is computed as

$$\text{AUC} = \frac{\dfrac{TP}{TP + FN} + \dfrac{TN}{TN + FP}}{2}.$$

We followed the Ten Fold Cross Validation procedure (TFCV) to obtain these metrics. It consists in randomly partitioning the dataset into 10 subsets. In each iteration, nine of these subsets constitute the training set and the remaining partition is the test set. The performance of the classifier is evaluated by the independent test set.

Although many of the tested dataset are known to contain outliers, in order to check the robustness of the classifiers, we perturbed the original data including label noise and SVM outliers. The label noise can come from several real situations which produce (intentionally or unintentionally) errors in the class of the individuals by adding outliers to the dataset, see Salgado et al. (2016). For this purpose, we randomly changed the class of 5% of the individuals in the training sample. On the other hand, for adding SVM outliers, we solved problem (SVM-ℓ_1) in the training sample. We sorted the individuals of each class in non-increasing order according to the result of $y_i \left(\sum_{k=1}^{d} (w_k^* x_{ik}) + b^* \right)$, where $i \in N, k \in D$. Note that w^* and b^* are the optimal solution values of (SVM-ℓ_1). We changed the class of the first 5% ranked individuals.

The tables in which this information is depicted are structured as follows: in the first column the name of the classifier is shown and in the second column the value of parameter B is reported. The following five columns describe information about the best case of C when 5% of the dataset contains label noise. The best case of C is the value of parameter C that reported the highest accuracy. Note that in the event of a tie, the best case is the one that provided a larger area under the curve. If this value is also the same, the best case is the one that took the least time. More concretely, the third column of the tables reports the above described value of parameter C, the next one shows the average time of the ten iterations, the following column depicts the average number of selected features for each classifier. Finally, the penultimate and last column of this group report the average ACC and AUC obtained for the ten iterations. The following five columns depict information about the best case of C when 5% of the dataset are SVM outliers. These columns are structured as the previous ones.

In Table 3.5, the computational experiments on the Colon dataset are depicted. We can observe that the best classification performance when 5% of the dataset contains label noise is obtained by (Fisher-SVM), with the following metrics: 88.75% of ACC and 88.75% of AUC. On the other hand, the best classifier is (RL-ℓ_1-M) when 5% of the dataset are SVM outliers, providing 88.75% ACC and 87.50% AUC. Slightly smaller classification metrics are

COLON, $n = 62, d = 2000$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	34	1	18.13	32.00	85.83%	83.75%	1	16.92	34.00	84.58%	83.75%
	23	1	20.11	23.00	87.50%	85.00%	1	18.93	23.00	83.75%	82.50%
	17	1	19.99	17.00	87.50%	86.25%	1	18.86	17.00	87.08%	86.25%
	11	1	18.42	11.00	84.58%	82.50%	1	14.08	11.00	87.08%	85.00%
	7	1	16.53	7.00	82.50%	81.25%	1	14.70	7.00	83.75%	81.25%
(FS-SVM)	34	1	0.80	32.60	85.83%	83.75%	10	3.33	34.00	87.08%	86.25%
	23	1	3.00	23.00	87.08%	85.00%	10	184.12	23.00	85.42%	83.75%
	17	1	33.63	17.00	87.50%	86.25%	100	524.55	17.00	83.75%	82.50%
	11	10	249.24	11.00	80.83%	78.75%	1	180.45	11.00	85.42%	85.00%
	7	1	20.11	7.00	85.83%	85.00%	10	885.16	7.00	83.75%	82.50%
(Fisher-SVM)	34	0.1	0.01	34.00	82.08%	81.25%	0.1	0.01	34.00	85.42%	86.25%
	23	1	0.01	23.00	82.08%	81.25%	0.1	0.01	23.00	85.42%	86.25%
	17	1	0.01	17.00	85.42%	85.00%	1	0.01	17.00	86.25%	86.25%
	11	10	0.01	11.00	85.83%	86.25%	0.1	0.01	11.00	83.75%	81.25%
	7	**1**	**0.01**	**7.00**	**88.75%**	**88.75%**	0.1	0.01	7.00	82.50%	78.75%
(RFE-SVM)	34	0.1	0.01	34.00	85.42%	84.17%	1	0.01	34.00	83.75%	83.75%
	23	0.1	0.01	23.00	85.42%	82.92%	1	0.01	23.00	82.08%	82.50%
	17	10	0.01	17.00	83.75%	82.92%	1	0.01	17.00	82.08%	82.50%
	11	0.1	0.01	11.00	82.08%	78.75%	1	0.01	11.00	82.08%	82.50%
	7	1	0.01	7.00	82.08%	80.83%	1	0.01	7.00	85.42%	85.00%
(RL-ℓ_1-M)	-	1	23.29	35.80	87.08%	85.00%	**1**	**21.66**	**38.20**	**88.75%**	**87.50%**
(SVM-ℓ_1)	-	1	0.67	32.70	85.83%	83.75%	1	0.68	36.70	84.17%	83.75%

Table 3.5. *Best average ACC and AUC for the Colon dataset.*

Leukemia, $n = 72, d = 5327$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	48	0.1	29.26	11.70	92.86%	93.75%	0.1	33.48	13.70	94.29%	95.00%
	32	0.1	29.31	11.70	92.86%	93.75%	**100**	**120.56**	**32.00**	**94.60%**	**95.54%**
	24	0.1	29.75	11.70	92.86%	93.75%	0.1	33.45	12.60	94.29%	95.00%
	16	0.1	29.50	11.80	92.86%	93.75%	0.1	33.48	12.60	94.29%	95.00%
	10	**0.1**	**38.63**	**9.90**	**92.86%**	**93.75%**	0.1	38.16	9.90	94.29%	95.00%
(FS-SVM)	48	0.1	1.99	22.80	90.00%	90.83%	100	6.85	48.00	92.86%	93.33%
	32	0.1	2.07	22.80	90.00%	90.83%	10	517.30	32.00	94.29%	94.58%
	24	0.1	2.08	18.80	90.00%	90.83%	1	1402.60	24.00	93.17%	93.87%
	16	100	956.59	16.00	90.32%	90.95%	1	1798.81	16.00	92.06%	93.15%
	10	0.1	2.55	10.00	90.00%	90.83%	0.1	4.38	10.00	88.57%	89.17%
(Fisher-SVM)	48	1	0.02	48.00	82.06%	81.49%	0.1	0.02	48.00	90.32%	90.54%
	32	0.1	0.01	32.00	79.52%	79.94%	0.01	0.02	32.00	91.43%	90.83%
	24	10	0.01	24.00	79.52%	79.94%	1	0.01	24.00	92.86%	92.92%
	16	0.1	0.01	16.00	74.92%	74.82%	0.1	0.01	16.00	91.75%	91.79%
	10	100	0.02	10.00	78.10%	78.27%	0.1	0.01	10.00	90.95%	91.61%
(RFE-SVM)	48	0.01	0.01	48.00	87.46%	88.04%	0.01	0.01	48.00	90.00%	90.00%
	32	0.01	0.01	32.00	86.03%	86.79%	0.01	0.01	32.00	92.86%	92.92%
	24	0.01	0.01	24.00	87.46%	88.04%	0.01	0.01	24.00	94.29%	94.58%
	16	0.01	0.01	16.00	86.03%	86.79%	0.01	0.01	16.00	92.86%	92.92%
	10	0.01	0.01	10.00	87.46%	88.04%	0.01	0.01	10.00	92.86%	93.33%
(RL-ℓ_1-M)	-	0.1	258.50	11.80	91.75%	93.04%	0.1	632.51	12.00	94.29%	95.00%
(SVM-ℓ_1)	-	0.1	2.52	11.40	90.00%	90.83%	100	1.96	51.30	92.86%	93.33%

Table 3.6. *Best average ACC and AUC for the Leukemia dataset.*

obtained by (RL-FS-M) selecting only eleven features on average in contrast with the 38.20 selected by (RL-ℓ_1-M).

The classification performance for the different models in the Leukemia dataset is depicted in Table 3.6. Note that the best classifier when 5% of the dataset contains label noise is (RL-FS-M), providing 92.86% ACC and 93.75% AUC. Moreover, observe that our model is able to classify a new individual by analyzing only 9.9 features on average. Similarly, the largest classification

DLBCL, $n = 77, d = 7129$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	44	100	20.10	43.30	88.75%	82.50%	10	42.63	43.70	92.50%	90.00%
	29	100	59.70	29.00	86.25%	79.17%	100	85.01	29.00	91.25%	89.17%
	22	10	74.67	22.00	88.75%	84.17%	10	96.91	22.00	90.00%	86.67%
	15	10	86.92	15.00	85.50%	76.67%	**10**	**126.05**	**15.00**	**95.00%**	**95.00%**
	9	100	159.07	9.00	85.50%	78.33%	100	425.14	9.00	92.50%	90.00%
(FS-SVM)	44	1	4.72	43.60	88.75%	82.50%	1	7.20	43.80	92.50%	90.00%
	29	1	127.71	29.00	87.50%	81.67%	1	339.38	29.00	91.25%	87.50%
	22	100	599.93	22.00	88.75%	85.83%	1	1137.20	22.00	92.50%	90.00%
	15	1	1348.37	15.00	85.50%	83.33%	10	1708.56	15.00	93.75%	92.50%
	9	10	1494.80	9.00	86.75%	79.17%	1	1678.26	9.00	91.25%	85.83%
(Fisher-SVM)	44	0.1	0.01	44.00	81.75%	64.17%	0.1	0.01	44.00	83.00%	85.42%
	29	0.1	0.01	29.00	76.75%	52.50%	0.1	0.01	29.00	78.50%	75.42%
	22	1	0.01	22.00	85.00%	85.00%	0.1	0.01	22.00	79.75%	74.58%
	15	1	0.01	15.00	78.00%	67.08%	0.1	0.01	15.00	81.00%	70.42%
	9	1	0.02	9.00	80.00%	66.67%	0.1	0.01	9.00	78.00%	58.33%
(RFE-SVM)	44	1	0.02	44.00	83.75%	81.67%	0.1	0.01	44.00	91.25%	90.83%
	29	1	0.02	29.00	81.25%	81.67%	0.1	0.01	29.00	91.25%	90.83%
	22	0.1	0.01	22.00	82.50%	80.67%	0.1	0.01	22.00	91.25%	89.17%
	15	0.1	0.01	15.00	82.50%	77.33%	0.1	0.01	15.00	95.00%	93.33%
	9	10	0.01	9.00	80.00%	80.83%	0.1	0.01	9.00	91.25%	87.50%
(RL-ℓ_1-M)	-	**1**	**74.40**	**51.10**	**91.25%**	**85.83%**	1	74.99	51.60	92.50%	91.67%
(SVM-ℓ_1)	-	1	3.86	43.80	88.75%	82.50%	1	3.59	47.20	92.50%	90.00%

Table 3.7. *Best average ACC and AUC for the DLBCL dataset.*

SONAR, $n = 208, d = 60$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	57	1	47.40	32.70	76.42%	76.05%	1	81.06	33.10	75.94%	75.59%
	38	**10**	**12.16**	**38.00**	**77.52%**	**77.03%**	0.1	237.67	7.50	75.51%	75.01%
	29	1	63.37	28.90	76.47%	75.97%	10	33.59	29.00	75.94%	75.48%
	19	10	148.01	19.00	75.94%	75.09%	**10**	**220.12**	**19.00**	**76.94%**	**76.90%**
	11	0.1	268.60	11.00	76.89%	76.76%	0.1	402.95	8.10	75.51%	75.01%
(FS-SVM)	57	0.1	0.07	14.40	76.99%	76.73%	0.1	0.08	17.80	72.61%	72.19%
	38	0.1	0.06	14.40	76.99%	76.73%	0.1	0.07	17.80	72.61%	72.19%
	29	0.1	0.06	14.40	76.99%	76.73%	10	678.46	29.00	73.61%	73.81%
	19	0.1	0.07	14.40	76.99%	76.73%	10	753.49	19.00	74.09%	74.08%
	11	0.1	0.07	8.20	76.99%	76.73%	1	137.99	11.00	73.18%	72.95%
(Fisher-SVM)	57	1	0.03	57.00	75.94%	76.76%	0.1	0.03	57.00	73.61%	74.17%]
	38	1	0.02	38.00	74.51%	75.21%	0.1	0.02	38.00	72.61%	73.26%
	29	1	0.02	29.00	76.52%	77.00%	1	0.02	29.00	72.03%	72.42%
	19	1	0.02	19.00	71.60%	72.00%	1	0.02	19.00	70.45%	71.12%
	11	1	0.03	11.00	73.03%	73.45%	1	0.03	11.00	70.03%	70.54%
(RFE-SVM)	57	1	0.03	57.00	75.94%	76.76%	0.1	0.03	57.00	73.61%	74.26%
	38	1	0.02	38.00	71.18%	71.76%	0.01	0.02	38.00	74.09%	74.49%
	29	0.1	0.02	29.00	72.13%	72.66%	10	0.02	29.00	75.04%	75.44%
	19	1	0.02	19.00	73.61%	73.94%	0.1	0.02	19.00	71.65%	72.26%
	11	0.1	0.03	11.00	72.13%	72.44%	0.1	0.03	11.00	71.18%	71.66%
(RL-ℓ_1-M)	-	1	1806.60	32.90	76.94%	76.47%	1	1805.03	33.10	76.37%	75.79%
(SVM-ℓ_1)	-	0.1	0.02	7.20	76.99%	76.73%	0.1	0.02	8.90	72.61%	72.19%

Table 3.8. *Best average ACC and AUC for the SONAR dataset.*

metrics (94.60% ACC and 95.54% AUC) are obtained by (RL-FS-M) when 5% of the dataset are SVM outliers.

The results from the DLBCL dataset are reported in Table 3.7. As can be appreciated, when 5% of the dataset contains label noise the best classifier is (RL-ℓ_1-M), providing 91.25% ACC and 85.83% AUC. However, this classifier used 51.1 features on average. Slightly smaller classification metrics are obtained by (RL-FS-M) and (FS-SVM) selecting fewer than half of the features on average. On the other hand, when 5% of the dataset are SVM outliers, (RL-FS-M) is the classifier that provides the largest ACC (95%) and AUC (95%) with nearly a point and a half difference in percentage compared to the rest of the models. Observe that this classifier selected only fifteen features on average from the 7129 provided by the data.

IONO, $n = 351, d = 33$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	33	1	75.71	25.00	88.06%	84.34%	1	108.15	25.30	86.57%	82.58%
	22	1	119.80	21.90	87.96%	84.00%	1	111.09	22.00	85.19%	80.49%
	17	10	34.13	17.00	87.13%	83.35%	1	134.82	17.00	86.02%	81.98%
	11	10	185.26	11.00	86.02%	81.64%	1	206.58	11.00	86.57%	82.08%
	7	10	197.31	7.00	86.76%	83.74%	1	367.34	7.00	85.37%	80.87%
(FS-SVM)	33	1	0.07	33.00	87.04%	82.95%	100	0.06	33.00	78.61%	71.63%
	22	100	0.92	22.00	87.22%	84.13%	100	3.05	22.00	72.59%	62.55%
	17	1	1.89	17.00	87.41%	83.90%	100	16.82	17.00	72.04%	62.28%
	11	**1**	**3.81**	**11.00**	**88.15%**	**84.66%**	100	40.39	11.00	73.70%	63.58%
	7	100	20.20	7.00	87.69%	84.23%	100	35.78	7.00	71.94%	61.06%
(Fisher-SVM)	33	0.1	0.04	33.00	87.96%	84.00%	0.01	0.04	33.00	86.85%	82.46%
	22	0.01	0.04	22.00	85.19%	80.65%	0.1	0.04	22.00	86.94%	82.13%
	17	0.01	0.04	17.00	84.35%	79.83%	1	0.04	17.00	86.11%	81.64%
	11	1	0.06	11.00	84.35%	80.17%	**100**	**0.05**	**11.00**	**87.78%**	**83.45%**
	7	0.01	0.05	7.00	81.85%	76.37%	10	0.05	7.00	87.50%	82.90%
(RFE-SVM)	33	0.1	0.05	33.00	84.17%	80.36%	0.01	0.04	33.00	86.85%	82.46%
	22	0.1	0.04	22.00	84.44%	80.61%	0.01	0.03	22.00	86.85%	82.46%
	17	1	0.05	17.00	84.44%	80.24%	0.01	0.04	17.00	86.94%	82.63%
	11	1	0.05	11.00	83.06%	78.57%	0.01	0.05	11.00	86.57%	81.91%
	7	1	0.06	7.00	82.87%	78.42%	0.01	0.04	7.00	85.37%	80.53%
(RL-ℓ_1-M)	-	1	1805.00	25.60	88.06%	84.34%	1	1804.40	24.60	85.74%	81.59%
(SVM-ℓ_1)	-	1	0.01	28.00	87.04%	82.95%	1	0.01	26.40	86.02%	81.81%

Table 3.9. *Best average ACC and AUC for the IONO dataset.*

Arrhythmia, $n = 420, d = 258$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	81	1	1629.88	79.60	73.57%	72.08%	1	1866.23	77.30	73.33%	71.62%
	54	1	1793.33	54.00	76.43%	75.17%	1	1860.52	54.00	70.95%	69.24%
	32	10	1834.59	32.00	75.48%	74.68%	1	1830.04	32.00	72.38%	71.48%
	16	1	1832.23	16.00	75.48%	74.01%	1	1827.54	16.00	74.52%	72.63%
	8	1	1845.67	8.00	72.14%	70.31%	10	1846.35	8.00	70.71%	68.76%
(FS-SVM)	81	1	267.70	81.00	75.00%	73.71%	1	615.29	80.80	70.95%	69.68%
	54	10	919.43	54.00	73.57%	73.18%	1	890.52	54.00	72.62%	70.49%
	32	10	882.25	32.00	72.86%	72.00%	10	755.65	32.00	73.33%	72.03%
	16	10	862.04	16.00	76.19%	74.55%	10	805.40	16.00	75.95%	74.13%
	8	**10**	**745.59**	**8.00**	**76.67%**	**74.74%**	100	875.68	8.00	73.57%	71.46%
(Fisher-SVM)	81	0.1	0.05	81.00	72.38%	69.00%	10	0.06	81.00	71.67%	67.99%
	54	1	0.05	54.00	70.95%	67.39%	10	0.06	54.00	70.95%	67.65%
	32	10	0.02	32.00	68.81%	64.65%	10	0.04	32.00	69.29%	65.35%
	16	100	0.02	16.00	66.67%	62.52%	10	0.04	16.00	67.62%	63.76%
	8	100	0.02	8.00	62.38%	57.32%	10	0.08	8.00	66.90%	63.76%
(RFE-SVM)	81	0.01	0.06	81.00	70.24%	67.36%	1	0.07	81.00	70.71%	67.81%
	54	0.1	0.06	54.00	69.76%	66.46%	0.01	0.05	54.00	72.38%	69.34%
	32	0.1	0.04	32.00	70.71%	67.64%	1	0.04	32.00	73.33%	70.63%
	16	0.1	0.03	16.00	70.48%	67.43%	0.1	0.02	16.00	69.76%	65.86%
	8	0.1	0.04	8.00	68.33%	65.00%	0.1	0.03	8.00	69.52%	65.36%
(RL-ℓ_1-M)	-	1	1802.37	84.70	73.57%	72.07%	**1**	**1801.78**	**79.30**	**75.95%**	**74.56%**
(SVM-ℓ_1)	-	1	0.25	97.00	74.52%	73.15%	1	0.27	103.40	71.90%	69.93%

Table 3.10. *Best average ACC and AUC for the Arrhythmia dataset.*

The computational results for the Sonar dataset are shown in Table 3.8. When 5% of the dataset contains label noise, the largest ACC (77.52%) and AUC (77.03%) are obtained by (RL-FS-M). Likewise, in the case of having a 5% of SVM outliers, the best classification performance (76.94% ACC and 76.90% AUC) is reported by (RL-FS-M).

In Table 3.9, the classification performance of the analyzed models on the IONO dataset is provided. Although the highest performance metrics when 5% of the dataset contains label noise are reported by (FS-SVM) whose average ACC is 88.15% and average AUC is 84.66%, slightly smaller accuracy and area under the curve (88.06% and 84.34% respectively) are provided by (RL-FS-M) and (RL-ℓ_1-M). On the contrary, when 5% of the dataset are SVM outliers, the best classifier is (Fisher-SVM).

The experiments carried out on the Arrhythmia dataset are depicted in Table 3.10. Observe that when 5% of the dataset contains label noise, (FS-SVM) is the classifier that provides the largest ACC (76.67%) and AUC (74.74%) on average. However, when 5% of the dataset are SVM outliers, (RL-ℓ_1-M) reported the largest classification metrics: 75.95% ACC and 74.56% AUC. Slightly smaller classification metrics are obtained by (FS-SVM) selecting only 16 features on average instead of 79.30.

The classification performance comparison for the Wdbc dataset is shown in Table 3.11. In the first case, i.e. 5% of label noise, almost all tested formulations provided nearly the same accuracy and area under the curve with the exception of (RL-FS-M), which reported the largest classification metrics, selecting only 16.4 features on average. In the second case, the best classifiers are (RL-FS-M) and (RL-ℓ_1-M) reporting 97.54% ACC and 97.06% AUC.

The computational results of Mfeat dataset are depicted in Table 3.12. As can be observed, the classification metrics are almost the same for all tested formulations when 5% of the dataset contains label noise. The one that provided the highest ACC (100%) and AUC (100%) is (FS-SVM). On the other hand, when 5% of the dataset are SVM outliers, the best classifier is (RL-FS-M) reporting 99.90% ACC and 99.50% AUC.

Finally, the computational experiments carried out on the Lepiota dataset are reported in Table 3.13. We can appreciate that the best classification performance is obtained by (RL-FS-M) when 5% of the dataset contains label noise, with the following metrics: 100% ACC and 100% AUC on average. Moreover, this classifier analyzed only 10.7 features on average. In the same manner, when 5% of the dataset are SVM outliers, the best classifier is (RL-FS-M) with more than one and a half unit difference in percentage compared to the rest of the models. It reported 99.73% ACC and 99.72% AUC. Observe that this classifier selected only 6.4 features on average.

Therefore, in this subsection we have demonstrated the efficiency of the proposed classifier. We have observed that in the majority of cases, (RL-FS-M)

Wdbc, $n = 569, d = 30$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
	29	**10**	**34.93**	**16.40**	**97.54%**	**97.06%**	1	39.05	9.90	97.54%	97.06%
	19	10	35.41	16.40	97.37%	96.83%	**1**	**39.00**	**9.90**	**97.54%**	**97.06%**
(RL-FS-M)	15	10	35.15	14.90	97.19%	96.69%	1	39.17	8.80	97.54%	97.06%
	10	1	50.45	9.80	96.67%	95.97%	1	39.93	8.90	97.54%	97.06%
	6	1	62.25	6.00	96.49%	95.63%	1	44.39	6.00	97.19%	96.69%
	29	1	0.10	22.40	96.49%	95.56%	1	0.12	22.30	96.13%	95.00%
	19	1	0.10	13.80	96.49%	95.56%	1	0.12	14.30	96.13%	95.00%
(FS-SVM)	15	1	0.11	12.20	96.49%	95.56%	1	0.14	13.90	96.13%	95.00%
	10	1	0.18	10.00	96.49%	95.56%	1	0.36	10.00	95.96%	94.78%
	6	1	0.25	6.00	96.66%	95.79%	10	5.45	6.00	96.67%	95.97%
	29	0.1	0.13	29.00	94.91%	95.35%	1	0.27	29.00	95.60%	95.70%
	19	1	0.19	19.00	93.14%	93.65%	0.1	0.28	19.00	95.43%	95.70%
(Fisher-SVM)	15	10	0.20	15.00	93.50%	93.85%	0.01	0.17	15.00	93.85%	92.62%
	10	10	0.20	10.00	94.19%	94.59%	1	0.26	10.00	93.50%	93.40%
	6	0.1	0.17	6.00	92.09%	92.18%	0.1	0.18	6.00	92.44%	90.75%
	29	10	0.14	29.00	90.86%	90.71%	1	0.45	29.00	95.78%	95.85%
	19	10	0.18	19.00	91.22%	90.92%	1	0.39	19.00	95.78%	95.95%
(RFE-SVM)	15	10	0.18	15.00	91.57%	91.20%	1	0.34	15.00	95.43%	95.57%
	10	10	0.19	10.00	90.69%	90.15%	1	0.23	10.00	95.61%	95.82%
	6	0.1	0.14	6.00	90.34%	90.03%	0.1	0.22	6.00	95.61%	95.71%
(RL-ℓ_1-M)	-	1	1807.89	9.40	96.67%	95.97%	1	1111.60	8.70	97.54%	97.06%
(SVM-ℓ_1)	-	1	0.02	11.20	96.49%	95.56%	1	0.02	13.30	96.13%	95.00%

Table 3.11. *Best average ACC and AUC for the Wdbc dataset.*

Mfeat, $n = 2000, d = 649$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
	202	0.1	516.33	35.50	99.90%	99.50%	0.1	1080.72	57.80	99.85%	99.25%
	135	0.1	516.10	35.50	99.90%	99.50%	0.1	1083.40	45.60	99.85%	99.25%
(RL-FS-M)	81	0.1	483.46	35.20	99.90%	99.50%	0.1	1092.25	45.00	99.90%	99.50%
	30	0.1	381.24	29.80	99.90%	99.50%	**0.1**	**581.82**	**29.90**	**99.90%**	**99.50%**
	10	0.1	268.69	10.00	99.95%	99.75%	1	350.67	5.10	99.85%	99.25%
	202	0.1	9.69	171.50	99.90%	99.50%	0.1	9.79	183.40	99.80%	99.22%
	135	0.1	9.74	91.70	99.90%	99.50%	0.1	9.79	92.70	99.80%	99.22%
(FS-SVM)	81	0.1	154.87	81.00	99.90%	99.50%	0.1	354.21	81.00	99.80%	99.22%
	30	**1**	**997.05**	**30.00**	**100.00%**	**100.00%**	0.1	832.68	30.00	99.75%	98.97%
	10	0.1	750.47	10.00	99.90%	99.72%	0.1	807.60	10.00	99.65%	98.69%
	202	0.1	1.58	202.00	100.00%	100.00%	0.01	1.57	202.00	99.80%	99.67%
	135	0.1	1.37	135.00	99.00%	99.22%	0.01	1.78	135.00	99.80%	99.50%
(Fisher-SVM)	81	0.01	0.90	81.00	95.75%	97.64%	0.01	1.97	81.00	99.85%	99.69%
	30	0.01	0.61	30.00	94.05%	70.47%	0.1	0.72	30.00	98.60%	99.22%
	10	0.1	2.59	10.00	94.70%	88.17%	0.1	2.91	10.00	95.65%	97.36%
	202	0.1	1.08	202.00	94.90%	83.43%	0.01	2.32	202.00	97.30%	98.28%
	135	0.01	0.75	135.00	94.90%	83.58%	0.01	1.23	135.00	98.05%	98.69%
(RFE-SVM)	81	0.01	0.47	81.00	94.85%	83.40%	0.01	0.93	81.00	98.15%	98.75%
	30	0.01	0.30	30.00	94.90%	83.58%	0.01	0.32	30.00	99.40%	99.44%
	10	0.01	0.81	10.00	94.20%	83.32%	0.01	1.49	10.00	98.60%	99.00%
(RL-ℓ_1-M)	-	0.1	1809.61	47.00	99.90%	99.50%	0.1	1809.92	70.30	99.70%	99.39%
(SVM-ℓ_1)	-	0.1	15.49	90.80	99.90%	99.50%	0.1	13.41	91.70	99.80%	99.22%

Table 3.12. *Best average ACC and AUC for the Mfeat dataset.*

provided the highest accuracy and area under the curve using less information to classify a new individual than (RL-ℓ_1-M). Note also that the performance of (SVM-ℓ_1) is significantly influenced by outliers, providing in several cases the worst results despite selecting many features. Although (FS-SVM) reported quite good results with label noise, its efficiency is decreased when the dataset contains SVM outliers. (Fisher-SVM) and (RFE-SVM) presents good results for some dataset but their behaviour is not as robust as our model.

Lepiota, $n = 8124, d = 109$											
		5% label noise					5% SVM outliers				
Form.	B	C	Time	Av. F	ACC	AUC	C	Time	Av. F	ACC	AUC
(RL-FS-M)	33	1	496.16	12.20	100.00%	100.00%	10	1741.94	23.60	96.75%	96.64%
	22	**1**	**529.65**	**10.70**	**100.00%**	**100.00%**	10	1759.38	21.80	96.71%	96.60%
	17	1	508.80	11.70	99.99%	99.99%	10	1730.30	16.70	96.86%	96.75%
	11	1	841.15	11.00	99.95%	99.95%	0.1	370.89	8.60	97.51%	97.42%
	7	10	962.17	7.00	99.84%	99.83%	**0.1**	**414.28**	**6.40**	**99.73%**	**99.72%**
(FS-SVM)	33	10	11.21	19.20	99.95%	99.95%	0.01	9.23	7.50	88.50%	63.81%
	22	10	11.10	17.60	99.95%	99.95%	0.01	9.23	7.50	88.50%	63.81%
	17	10	11.81	16.30	99.95%	99.95%	0.01	9.22	7.50	88.50%	63.81%
	11	10	12.84	11.00	99.91%	99.91%	100	10.77	8.90	90.98%	89.22%
	7	0.1	10.69	6.00	99.70%	99.69%	100	10.87	6.40	87.27%	86.77%
(Fisher-SVM)	33	100	4.41	33.00	80.92%	81.44%	0.1	4.66	33.00	95.88%	95.72%
	22	100	4.43	22.00	61.10%	59.89%	0.1	4.65	22.00	95.26%	95.08%
	17	1	4.45	17.00	57.80%	56.55%	0.01	4.59	17.00	95.37%	95.20%
	11	100	4.46	11.00	52.83%	51.07%	0.01	4.57	11.00	95.37%	95.20%
	7	10	4.47	7.00	52.68%	50.92%	0.01	4.70	7.00	92.59%	92.31%
(RFE-SVM)	33	10	4.37	33.00	94.95%	94.94%	0.01	4.50	33.00	96.75%	96.63%
	22	10	4.72	22.00	94.95%	94.94%	0.01	4.66	22.00	96.63%	96.50%
	17	100	4.48	17.00	94.95%	94.94%	0.01	4.61	17.00	96.59%	96.46%
	11	0.1	4.49	11.00	94.78%	94.76%	0.1	4.60	11.00	96.98%	96.87%
	7	0.1	4.48	7.00	94.78%	94.76%	0.1	4.83	7.00	98.09%	98.02%
(RL-ℓ_1-M)	-	10.00	1806.04	15.80	99.98%	99.97%	0.1	1807.20	10.78	96.53%	96.40%
(SVM-ℓ_1)	-	10.00	3.80	17.30	99.95%	99.95%	0.1	3.80	11.40	96.53%	96.41%

Table 3.13. *Best average ACC and AUC for the Lepiota dataset.*

In summary, choosing the parameters which provide the best accuracy in each dataset when contains 5% of label noise, the average (maximum) percentages of improvement for ACC and AUC of our model with respect the others in the studied datasets are 2.63% (24.54%) and 2.94% (22.78%), respectively. Similarly, when the dataset contains 5% of SVM outliers, the average (maximum) percentages of improvement for ACC and AUC are 2.06% (14.46%) and 2.36% (14.58%), respectively. More specifically, the average (maximum) percentages of improvement for ACC and AUC with respect to each model is depicted in Table 3.14.

	5% Label noise		5% SVM outliers	
	Av. ACC impr. (Max. ACC impr.)	Av. AUC impr. (Max. AUC impr.)	Av. ACC impr. (Max. ACC impr.)	Av. AUC impr. (Max. AUC impr.)
(FS-SVM)	0.44 (2.81)	0.32 (3.08)	2.71(10.13)	3.70 (14.58)
(Fisher-SVM)	5.50 (23.58)	5.00 (22.78)	3.39 (14.46)	3.15 (11.22)
(RFE-SVM)	5.49 (8.08)	7.01 (19.35)	1.11 (2.54)	1.26 (2.82)
(RL-ℓ_1-M)	0.51 (3.88)	0.75 (4.29)	0.50 (3.32)	0.64 (3.64)
(SVM-ℓ_1)	1.19 (3.17)	1.66 (3.21)	2.58 (5.97)	3.06 (6.53)

Table 3.14. *Improvement of (RL-FS-M) with respect to the rest of the models.*

The aforementioned results show that (RL-FS-M) is a very robust classification method improving the existing ones or being among the best in terms of ACC and AUC. Moreover model (RL-FS-M) involves a reduced number of features in the obtained classifier in contrast to the models (RL-ℓ_1) and (SVM-ℓ_1).

3.5 Concluding remarks

In this chapter, we have developed a new model based on support vector machines especially designed for datasets with a large number of features that may contain outliers. We have formulated the model as a mixed-integer problem and proposed an exact strategy for computing the big M parameters of the formulation. Moreover, we have developed a heuristic algorithm to solve it efficiently. We have also validated the heuristic, proving that the obtained upper bound is of high quality.

Furthermore, we have compared the performance of the proposed classifier with other classifiers based on support vector machines that deal with feature selection or with outlier detection. We have showed the efficiency of our model, whose competitive advantage is that it deals simultaneously with both aspects. Finally, we think that analyzing the proposed model using other ℓ_p-norms would be interesting for future research. Note that the results presented in this chapter are published in Baldomero-Naranjo et al. (2021d).

In the previous two chapters, we use a separator hyperplane as a method to classify data. Actually, we decide on the location of a hyperplane in $\mathbb{R}^m$. In the following chapter we also address a location problem, although the problem is stated in a network rather than in $\mathbb{R}^m$. In particular, we present the upgrading version of the maximal covering location problem.

4 Upgrading edges in the Maximal Covering Location Problem

We study the upgrading version of the maximal covering location problem with edge length modifications on networks. This problem aims at locating p facilities on the vertices (of the network) so as to maximize coverage, considering that the length of the edges can be reduced at a cost, subject to a given budget. Hence, we have to simultaneously decide on: the optimal location of p facilities and the optimal edge length reductions.

This problem is NP-hard on general graphs. To solve it, we propose three different mixed-integer formulations and an effective preprocessing phase for fixing variables and removing some of the constraints. Moreover, we strengthen the proposed formulations including valid inequalities. Finally, we compare the three formulations and their corresponding improvements by testing their performance over different datasets.

4.1 Introduction

The maximal covering location problem (MCLP) was first introduced by Church and ReVelle (1974). Given a set of clients, each with their own demand, the aim is to locate a fixed number of facilities so as to maximize the amount of covered demand. A client is hereby considered to be covered if their distance to a facility is smaller than or equal to a given coverage radius. Since its origins, this model has been widely studied in the literature under different perspectives. One of the most distinguishing aspects is the solution domain of the problem: continuous (Church, 1984; Plastria, 2002; Bansal and Kianfar, 2017), discrete (Church and ReVelle, 1974; Avella et al., 2009; García and Marín, 2019; Cordeau et al., 2019), or on networks (Church and Meadows, 1979; Berman et al., 2016; Fröhlich et al., 2020). Furthermore, the maximal covering location problem has been solved dealing with alternative coverage assumptions, like gradual coverage (Berman and Krass, 2002) and cooperative coverage (Averbakh et al., 2014; Karatas and Eriskin, 2021), and with uncertainty, for example uncertainty in the customer demand (Berman and Wang, 2011; Baldomero-Naranjo et al., 2021b), in the availability of facilities to provide coverage (Daskin, 1983; Marín et al., 2018; Vatsa and Jayaswal, 2021), or in all relevant parameters together (Arana-Jiménez et al., 2020).

Common to all those problems is, however, that the parameters of the network and the problem remain fixed. In this chapter, we propose a different approach dealing with the maximal covering location problem on networks assuming that edges can be *upgraded* and the total cost of all upgrades is subject to a budget constraint. Upgrading an edge hereby means reducing its length, usually within certain limits, at a given cost. There are three main problems in the literature in which two key parameters of the network, demand weights and edge lengths, are adjusted:

- In *inverse problems*, the objective is to modify one of the two parameters at minimum cost such that a given feasible solution becomes optimal, see e.g. Heuberger (2004); Burkard et al. (2004b); Baroughi Bonab et al. (2011); Wu et al. (2013); Alizadeh and Etemad (2016); Yang and Zhang (2008); Nguyen and Sepasian (2016); Gassner (2012).
- In *reverse problems*, the objective is to maximally improve a prespecified solution by changing the edge lengths within certain limits and subject to a given budget, see e.g. Burkard et al. (2006, 2008); Wang and Bai (2010); Zhang et al. (1999).
- In *up/downgrading problems*, an actor modifies the parameters of the network and then a reactor takes a decision. In upgrading problems, actor and reactor have the same goal; in downgrading problems, their objectives are conflicting.

Summing up, the main difference between inverse (reverse) problems and up/downgrading problems is that in the former there is a given solution to improve, while in the latter there is not. In this chapter, we will focus on the upgrading maximal covering location problem with variable edge lengths.

Next, we briefly review the literature. The upgrading version of many classical problems has been studied during the last decades, e.g. for the spanning tree problem (Álvarez-Miranda and Sinnl, 2017), for the hub-location problem (Blanco and Marín, 2019), for bottleneck problems (Burkard et al., 2004a), for minimum flow cost problems (Demgensky et al., 2002), for the maximal shortest path interdiction problem (Zhang et al., 2021), or for communication and signal flow problems (Paik and Sahni, 1995). In the context of location problems on networks and vertex weight modifications (the weight of the vertices is modified subject to a prespecified budget), the following problems have been analyzed: the 1-median problem (Gassner, 2007), the 1-center problem (Gassner, 2009), the Euclidean 1-median problem (Plastria, 2016), and the p-median problem (Sepasian and Rahbarnia, 2015), among others.

In the context of upgrading location problems with variable edge lengths, we are aware of only two publications: upgrading the 1-center problem by Sepasian (2018) and upgrading the obnoxious p-median problem on trees by Afrashteh et al. (2020). Therefore, the first aim of this chapter is to fill the gap in the literature by studying the upgrading maximal covering location problem with variable edge lengths.

This problem has several interesting applications in real-life. Note that two decisions are made at the same time. On the one hand, decide where to locate the p facilities, and on the other hand, determine which edges to upgrade and by how much.

One application of this problem arises when a public administration wants to improve the accessibility of public services for citizens, e.g. for health centers, educational facilities or social welfare facilities. As the improvement is closely linked with distances (Ensor and Cooper, 2004), one way to achieve this is to invest in the infrastructure in order to reduce travel times to those services. Such an investment is often a combination of building new facilities and improving the means to get to them, for example by upgrading roads (developing a road into a highway, adding new lanes, etc.) and enhancing public transport (incorporating high-speed lines, adding dedicated bus lines, increasing the frequency of service along links, etc.).

An interesting application in the private sector is for telecommunication companies. To improve their transmission rates and broadband coverage, they will have to increase the bandwidth on existing network links as well as build new or extend existing switching centers. Similar problems are faced by gas and electricity companies who wish to increase their coverage.

Finally, we would like to highlight another useful application in shopping centers, airports, etc. The aim is to locate services such as defibrillators and information posts, in combination with building additional passenger conveyors or escalators to make sure that as many people as possible are within a fixed walking distance of these facilities.

In this chapter, we derive three mixed-integer linear programming formulations for the maximal covering location problem with edge upgrades. Furthermore, we develop an effective preprocessing phase that allows us to reduce the dimension of the proposed formulations. Besides, we include several sets of valid inequalities in order to eliminate symmetries and improve the resolution times of the formulations.

The rest of the chapter is structured as follows. In Section 4.2 the problem is introduced. Section 4.3 presents the first formulation for the problem based on flow variables. Moreover, a preprocessing phase and valid inequalities are developed. Next, in Section 4.4 and Section 4.5 two new formulations are proposed. In addition, several valid inequalities to enhance them are presented. Section 4.6 contains computational experiments in which we compare the three formulations. We also test the efficiency of the developed valid inequalities. Finally, our conclusions and some future research topics are included in Section 4.7.

4.2 Definitions and Problem Description

Let $N = (V, E, \ell)$ be an undirected network with node set $V = \{1, \dots, n\}$ and edge set E, where $|E| = m$. Every edge $e = [k, q] = [q, k] \in E$, $k, q \in V$, has a positive length $\ell_e = \ell_{[k,q]}$ and is assumed to be rectifiable. For $i, j \in V$, $d(i, j)$ is the length of the shortest path connecting i with j. Furthermore, we are given a fixed coverage radius $R > 0$. We say that a node $i \in V$ is *covered* by a facility at node j if $d(i, j) \leq R$. Finally, for each node $i \in V$ we are given a non-negative amount w_i that specifies the demand at the node.

The length ℓ_e of each edge $e \in E$ can be reduced by an amount lower than or equal to $u_e \in [0, \ell_e)$, $e \in E$. Without loss of generality, we assume that $\ell_e - u_e \leq R$, for $e \in E$ (if that were not the case, i.e., there were an edge $e \in E$ such that $\ell_e - u_e > R$, then e can be removed from the network without affecting the optimal solution). Moreover, any unit of reduction of the length of the edge e comes at a cost of c_e and there is a budget constraint B on the overall cost of reduction. Again without loss of generality, we assume that $c_e u_e \leq B$, for $e \in E$ (if that were not the case, i.e., there were a cost c_e for $e \in E$ such that $c_e u_e > B$, then u_e can be substituted by $u_e = B/c_e$ without affecting the optimal solution). Finally, we assume that facilities can only be located at nodes. The upgrading maximal covering location problem (Up-MCLP) aims to locate p service facilities covering the maximum demand taking into account that the total cost for the edge length reductions is within the given budget.

Let $\delta = (\delta_e)_{e \in E}$ denote a vector of edge length reductions, $0 \leq \delta_e \leq u_e$, for $e \in E$. Moreover, let $d(i, j, \delta)$ be the length of a shortest path between nodes i and j after the edge length reductions δ have been applied, i.e., a shortest path in the network $(V, E, \ell(\delta))$ where $\ell_e(\delta) = \ell_e - \delta_e$, for $e \in E$. Finally, for $p \in \mathbb{N}$ let $X_p \subseteq V$ denote a set of p nodes and let $C(X_p, \delta) = \{i \in V \mid$

$\exists j \in X_p : d(i,j,\delta) \leq R\}$ denote the set of all nodes covered by a facility in X_p after the edge upgrades. Then, Up-MCLP can be formulated as:

$$\max \left\{ \sum_{i \in C(X_p,\delta)} w_i \;\middle|\; \sum_{e \in E} c_e \delta_e \leq B, X_p \subseteq V, |X_p| = p, 0 \leq \delta_e \leq u_e, e \in E \right\}.$$

Table 4.1 summarizes the notation used in this chapter.

A	Set of all arcs in the induced directed network.
B	Budget.
c_e	Unit cost of reducing the length of edge e, $e \in E$.
$d(i,j)$	Distance between nodes i and j before upgrading, $i,j \in V$.
$d(i,j,\beta)$	Distance between nodes i and j after the edge length reductions β_e, $e \in E$. In particular, $d(i,j,\delta^{ij})$ represents the distance after the most favorable feasible edge length reductions in the path from i to j and $d(i,j,u)$ the distance in a network with edge lengths $\ell_e - u_e$, $e \in E$.
Γ_i	Set of edges incident to node i for each $i \in V$.
Γ_i^+ (Γ_i^-)	Set of outgoing (incoming) arcs for each $i \in V$.
m	Number of edges.
n	Number of nodes.
$N = (V, E, \ell)$	Network with node set V, edge set E, where $e \in E$ has length ℓ_e.
p	Number of facilities.
R	Coverage radius.
u_e	Maximum amount that edge e can be reduced, $e \in E$.
$\hat{V}_i$	Set of nodes whose distance to i before upgrading is lower than or equal to R, i.e., $\{j \in V \setminus \{i\} : d(i,j) \leq R\}$.
w_i	Demand of node i, for $i \in V$.

Table 4.1. *Notation used in the chapter.*

Observe that this problem is NP-hard because the maximal covering location problem (MCLP) is a particular case of Up-MCLP (setting $u_e = 0$ for all $e \in E$). The NP-hardness of the maximal covering location problem is proved in Hochbaum (1997).

4.3 Flow coverage formulation

In this section, we propose the first of our three MIP formulations for Up-MCLP. Using flow variables, the idea of this formulation is to model a path between those pairs of nodes for which the distance between them is smaller than or equal to R after the edge length reductions have been applied. That is, if $d(i,j,\delta) \leq R$, then this will be reflected in the formulation by a unit flow between nodes i and j. If, however, $d(i,j,\delta) > R$, then the flow between i and j will be zero. We note that in the former case, any path of length $\leq R$ will do to assert coverage of i (j) by a service facility located at site j (i), so we do not insist on finding the shortest path.

To facilitate the use of flow variables, we consider a directed network $N_D = (V, A, \ell)$ with node set $V = \{1, \dots, n\}$ and arc set A containing arcs (i,j) and (j,i) for each edge $[i,j] \in E$. We denote $e_a \in E$ the undirected edge corresponding to $a \in A$ and we define Γ_i^+ (Γ_i^-) as the set of outgoing (incoming) arcs for each $i \in V$. The set of variables used in the formulation is summarized below.

Decision variables

x_j	1, if there is a facility at node j, and 0, otherwise, for $j \in V$.
y_{ij}	1, if node i is assigned to a facility at node j, and 0, otherwise, for $i, j \in V, i \neq j$.
δ_e	The amount of reduction of the length of edge e, for $e \in E$.
f_a^{ij}	1, if a path of length $\leq R$ from i to j traverses arc a, and 0, otherwise, for $i, j \in V, i < j$, $a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right)$.
α_a^{ij}	The length of arc a, if this arc belongs to a path of length $\leq R$ from node i to node j ($\alpha_a^{ij} = 0$ otherwise), for $i, j \in V, i < j$, $a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right)$.

Observe that in the definition of the y-variables, we use the term "assign to" instead of "covered by". A node can potentially be covered by more than one service facility and we decided to resolve this ambiguity by explicitly assigning a node to a facility as this simplifies the explanations of the formulations. Taking into account the notation presented above, the flow coverage formulation (Flow-Cov) for Up-MCLP is:

$$\max \sum_{i \in V} w_i \left(\sum_{j \in V \setminus \{i\}} y_{ij} + x_i \right)$$

$$\text{s.t.} \sum_{j \in V} x_j = p, \tag{4.1}$$

$$\sum_{j \in V \setminus \{i\}} y_{ij} + x_i \leq 1, \qquad i \in V, \tag{4.2}$$

$$y_{ij} \leq x_j, \qquad i, j \in V, i \neq j, \tag{4.3}$$

$$\sum_{e \in E} c_e \delta_e \leq B, \tag{4.4}$$

$$0 \leq \delta_e \leq u_e, \qquad e \in E, \tag{4.5}$$

$$\sum_{a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right)} \alpha_a^{ij} \leq R, \qquad i, j \in V, i < j, \tag{4.6}$$

$$\alpha_a^{ij} \geq f_a^{ij} \ell_{e_a} - \delta_{e_a}, \qquad i, j \in V, i < j, a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right), \tag{4.7}$$

$$\sum_{a \in \Gamma_k^+, a \notin \Gamma_i^-} f_a^{ij} - \sum_{a \in \Gamma_k^-, a \notin \Gamma_j^+} f_a^{ij} = 0, \quad i, j \in V, i < j, k \in V \setminus \{i, j\}, \tag{4.8}$$

$$\sum_{a\in\Gamma_i^+} f_a^{ij} = y_{ij} + y_{ji}, \qquad i,j\in V, i<j, \tag{4.9}$$

$$\sum_{a\in\Gamma_j^-} f_a^{ij} = y_{ij} + y_{ji}, \qquad i,j\in V, i<j, \tag{4.10}$$

$$0 \le \alpha_a^{ij} \le \ell_{e_a} - \delta_{e_a}, \qquad i,j\in V, i<j, a\in A\backslash\left(\Gamma_i^-\cup\Gamma_j^+\right), \tag{4.11}$$

$$x_j \in \{0,1\}, \qquad j\in V, \tag{4.12}$$

$$y_{ij} \in \{0,1\}, \qquad i,j\in V, i\neq j, \tag{4.13}$$

$$f_a^{ij} \in \{0,1\}, \qquad i,j\in V, i<j, a\in A\backslash\left(\Gamma_i^-\cup\Gamma_j^+\right). \tag{4.14}$$

The objective of the problem is to maximize the amount of covered demand. Constraint (4.1) fixes the number of located facilities. The family of constraints (4.2) guarantees that either node i is itself a service facility or is assigned to at most one node. The family of constraints (4.3) ensures that a node is assigned to an open facility. The families of constraints (4.4) and (4.5) force that the reduction on the length of the edges in the network is feasible. The families of constraints (4.6)–(4.10) ensure that if y_{ij} (y_{ji}) takes value one, there exists a path shorter than or equal to R from i to j (from j to i). Note that constraints (4.2) and (4.3) imply $y_{ij}+y_{ji}\le 1$, for $i,j\in V, i\neq j$.

Lemma 4.1. *An equivalent formulation of (Flow-Cov) is obtained substituting the set of constraints* (4.12) *by:*

$$0\le x_j\le 1, \quad j\in V, \tag{4.15}$$

and the set of constraints (4.13) *by:*

$$0\le y_{ij}\le 1, \quad i,j\in V, i\neq j. \tag{4.16}$$

Proof:

Concerning the first part of the lemma, the x-variables inherit the integrality condition from y-variables due to constraints (4.2) and (4.3). Let x^* and y^* be optimal values for the x- and y-variables, respectively, of formulation (Flow-Cov) when constraints (4.12) are substituted by (4.15). For any $i\in V$ such that $\sum_{j\in V\backslash\{i\}} y_{ij}^* = 1$, constraint (4.2) ensures that $x_i^*=0$. On the other hand, for any $i\in V$, such that there exists $j_0\in V$, $j_0\neq i$, with $y_{j_0 i}^*=1$, constraints (4.3) guarantee that $x_i^*=1$. Finally, the model will choose to locate the remaining service facilities (up to a total of p) at the uncovered nodes with the largest demand.

Regarding the second part of the lemma, following a similar argument than before, we conclude that the y-variables inherit the integrality condition from the f-variables due to constraints (4.9) and (4.10) and the condition that $y_{ij}+y_{ji}\le 1$ (derived by constraints (4.2) and (4.3)). $\square$

Observe that even though the f-variables are the intuitive candidates for relaxation (since their number is much larger than the number of x- and y-variables), there are examples where the optimal value differs when the integrality condition of these variables is relaxed.

An alternative formulation for Up-MCLP can be derived from the formulation (Flow-Cov) by replacing constraints (4.6), (4.7), and (4.11) with the following ones:

$$\sum_{a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right)} \left(f_a^{ij} \ell_{e_a} - \gamma_a^{ij} \right) \leq R, \quad i,j \in V, i<j, \tag{4.17}$$

$$\gamma_a^{ij} \leq u_{e_a} f_a^{ij}, \qquad i,j \in V, i<j, a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right), \tag{4.18}$$

$$0 \leq \gamma_a^{ij} \leq \delta_{e_a}, \qquad i,j \in V, i<j, a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right), \tag{4.19}$$

where γ_a^{ij} represents the reduction on the length of arc a if this arc belongs to a path of length $\leq R$ from node i to node j, for $i,j \in V, i<j$, $a \in A \setminus \left(\Gamma_i^- \cup \Gamma_j^+\right)$. A preliminary computational analysis showed that the alternative formulation (Flow-Cov) for Up-MCLP where constraints (4.6), (4.7), and (4.11) are replaced with (4.17), (4.18), and (4.19) is better than the original (Flow-Cov).

4.3.1 Preprocessing phase

Next, we present two results for preprocessing the model, reducing the number of constraint and variables of the above formulation. The idea of the first is that if the distance between node i and node j is smaller than or equal to R even before modifying the edge lengths of the network, then node i can always be covered by a facility located at node j (and vice versa) regardless of the edge length reductions made in the network.

Proposition 4.1. *If $d(i,j) \leq R$, for $i,j \in V, i<j$ then it is not necessary to include either the f_a^{ij}-variables or the α_a^{ij}-variables (γ_a^{ij}-variables) in formulation (Flow-Cov). Moreover, we can remove the constraints associated with these variables from the family of constraints* (4.6)–(4.11) *and* (4.14) *(*(4.8)–(4.10)*,* (4.14)*,* (4.17)–(4.19)*).*

The second result analyses the opposite case, i.e., Proposition 4.2 considers the situation in which the distance between node i and node j is greater than R independently of the edge length reductions.

Proposition 4.2. *If one of the following four conditions is fulfilled for $i,j \in V$, $i<j$, the variables $y_{ij}, y_{ji}, f_a^{ij}, \alpha_a^{ij}(\gamma_a^{ij})$ for $a \in A$ can be removed from the (Flow-Cov) formulation. Moreover, the constraints associated with this pair of nodes can be deleted from* (4.2)*,* (4.3)*,* (4.6)–(4.11)*,* (4.13)*,* (4.14) *(*(4.2)*,* (4.3)*,* (4.8)–(4.10)*,* (4.13)*,* (4.14)*,* (4.17)–(4.19)*).*

i) $d(i,j) > R + \sum_{e \in E} u_e$, *for* $i,j \in V, i < j$.

ii) $d(i,j,u) > R$, *for* $i,j \in V, i < j$, *where* $d(i,j,u)$ *is the length of the shortest path from* i *to* j *in a graph with edge lengths* $\ell_e - u_e$, *for* $e \in E$.

iii) $d(i,j) > R + \sum_{k=1}^{\bar{k}} u_{e_{\sigma(k)}} + \dfrac{B - \sum_{k=1}^{\bar{k}} u_{e_{\sigma(k)}}}{c_{e_{\sigma(\bar{k}+1)}}}$ *for* $i,j \in V, i < j$, *where* $\bar{k}$ *is the largest index* k *that satisfies the following condition:*

$$\sum_{h=1}^{k} u_{e_{\sigma(h)}} c_{e_{\sigma(h)}} \leq B, \tag{4.20}$$

and $\sigma(\cdot)$ *is a permutation of* $\{1,\dots,m\}$ *that sorts the unit upgrade costs in non-decreasing order.*

iv) *The optimal value of the following problem is greater than* R, *for* $i,j \in V$, $i < j$,

$$\begin{aligned}
\left(P_{d(i,j,\delta^{ij})}\right) \quad \min \quad & \sum_{a \in A} (f_a \ell_{e_a} - \gamma_a) \\
s.t. \quad & (4.4), (4.5), \\
& \sum_{a \in \Gamma_k^+} f_a - \sum_{a \in \Gamma_k^-} f_a = g_k, \quad k \in V, && (4.21) \\
& \gamma_a \leq u_{e_a} f_a, \quad a \in A, && (4.22) \\
& \gamma_a \leq \delta_{e_a}, \quad a \in A, && (4.23) \\
& f_a \in \{0,1\}, \quad a \in A, && (4.24)
\end{aligned}$$

where the f*-variables and* γ*-variables are defined as above (we dropped the indices* i *and* j *for the ease of exposition), and*

$$g_k = \begin{cases} 1, & \text{if } k = i, \\ -1, & \text{if } k = j, \\ 0, & \text{otherwise.} \end{cases}$$

Proof:

Each of the items of the proposition is proven below.

i) The first condition considers the case where the distance from i to j is greater than R even when reducing the length of every edge by the maximum amount allowed. Therefore, it is straightforward to conclude that the distance between the two nodes cannot be less than or equal to R.

ii) In the previous condition, the maximum amount of reduction in the whole network was considered without taking into account the edges for which this reduction is made. Now we compute the shortest path between two nodes in the network assuming an unlimited budget, i.e., the full discount is applied to all edges. For each edge e, let $\ell_{e_u} = \ell_e - u_e$. For $i, j \in V$, let $d(i, j, u)$ be the length of the shortest path connecting i with j where the length of the edges are ℓ_{e_u}, for $e \in E$. Hence, even if reducing the maximum amount allowed on all edges, the distance is greater than R, clearly, node i cannot be assigned to node j, and vice versa. Although condition *i)* is weaker than *ii)*, *i)* can be checked more efficiently.

iii) This condition is similar to the first one, but takes into account the budget constraint (4.4). In this case, we calculate the maximum reduction in the network allowed by the budget. For this purpose, we sort the upgrade costs c_e, for $e \in E$, in non-decreasing order. Let σ be a permutation of $\{1, \ldots, m\}$ such that $c_{e_{\sigma(1)}} \leq c_{e_{\sigma(2)}} \leq \ldots \leq c_{e_{\sigma(m)}}$. Then, we compute the maximum total length reduction over the network, i.e., we spend the budget on upgrading the cheapest edges. Let $\bar{k}$ be the largest index k that satisfies condition (4.20). Therefore, the right-hand side of *iii)* minus R is the maximum length reduction between any two nodes of the network. Taking into account the above arguments, we conclude that node i cannot be assigned to a facility located at node j, and vice versa.

iv) Condition *iii)* provides the maximal reduction without taking into account whether this reduction can be achieved in a path form i to j. For this reason, that bound can be tightened, but it requires to solve a separate problem for each pair of vertices. Formulation $\left(P_{d(i,j,\delta^{ij})}\right)$ computes the shortest path between node i and node j assuming that all the budget can be spent just for the path between those two nodes. Therefore, the optimal value of this problem, named $d(i, j, \delta^{ij})$, is the minimal distance between node i and node j after the most favorable edges length reductions. Hence, if $d(i, j, \delta^{ij})$ is greater than R, node i can never be assigned to a facility at node j, and vice versa. □

As stated in Demgensky et al. (2002), the shortest path problem where the length of the edges can be reduced, $\left(P_{d(i,j,\delta^{ij})}\right)$, is NP-hard. However, the optimal value of the LP relaxation of formulation $\left(P_{d(i,j,\delta^{ij})}\right)$ provides a valid bound that can still be used instead, albeit yielding a weaker condition. If this value is greater than R, then i can never cover j, and vice versa.

4.3.2 Valid inequalities

In the previous subsection, we have presented two results to preprocess the model, reducing the number of constraints and variables. In this one, we propose several families of valid inequalities to strengthen the (Flow-Cov) formulation.

Proposition 4.3. *Let $\hat{V}_i := \{j \in V \backslash \{i\} : d(i,j) \leq R\}$. The following families of constraints are valid inequalities for (RL-FS):*

$$f^{ij}_{(k,q)} + f^{ij}_{(q,k)} \leq 1, \qquad [k,q] \in E, i,j \in V, i < j, k,q \neq i,j, \tag{4.25}$$

$$y_{kj} \geq y_{ij} + f^{ij}_a - 1, \qquad i,j,k \in V, i < j, k \neq i, k \neq j, a \in \Gamma^-_k, \tag{4.26}$$

$$y_{ki} \geq y_{ji} + f^{ij}_a - 1, \qquad i,j,k \in V, i < j, k \neq i, k \neq j, a \in \Gamma^-_k, \tag{4.27}$$

$$x_j \leq \sum_{k:k\neq i, d(i,k)\leq d(i,j)} y_{ik} + x_i, \quad i \in V, j \in \hat{V}_i, \tag{4.28}$$

$$x_j \leq \sum_{k:k\neq i, d(i,k,\delta^{ik})\leq d(i,j)} y_{ik} + x_i, \; i \in V, j \in \hat{V}_i. \tag{4.29}$$

Proof:

The proof of valid inequalities is given below:

The first family of constraints (4.25) ensure that an edge is not traversed in both directions on a path from i to j, $i < j$.

The second and the third families of constraints, (4.26) and (4.27), are based on the fact that a path between two non-adjacent nodes i and j will traverse at least one other node. Therefore, if there exists a path whose length is less than or equal to R that connects a facility at node i with demand point j and traverses node k, then node k will also be assigned to facility i. More concretely, given $i, j \in V, i \leq j$, if $f^{ij}_a = 1$ for some $a \in A$, such that $a \in \Gamma^-_k$, $k \neq i, k \neq j$ and $y_{ij} = 1$ ($y_{ji} = 1$), then the constraints impose that $y_{kj} = 1$ ($y_{ki} = 1$).

Regarding (4.28), these constraints ensure that a node will be served by the closest service facility that is within the covering distance before upgrading the network (whenever at least one service facility is closer than the coverage radius before upgrading the network). Observe that these constraints eliminate symmetries and are valid also for formulation (Flow-Cov) because nodes might not be assigned to the closest service facility in the upgraded network. Nevertheless, the situation would be incompatible with constraints (4.26) and (4.27), as explained in the following remark (Remark 4.3.1).

Finally, whenever at least one service facility j is closer to a node i than the coverage radius before upgrading the network, constraints (4.29) ensure that this node will either host a facility itself or be assigned to this service facility or to a facility that can be closer after upgrading the network (it considers the distances in the range of $d(i,j)$ and the most favorable edge length reductions, i.e., $d(i,k,\delta^{ik})$ for any $k \neq i$.). $\square$

Remark 4.3.1. *Regarding the valid inequalities presented in the previous result, note:*

i) *Constraints* (4.26) *and* (4.27) *might be incompatible with* (4.28)*, i.e., constraints* (4.26)–(4.28) *cannot be included in the formulation simultaneously.*

ii) *The family of valid inequalities* (4.28) *is tighter than* (4.29)*, but* (4.29) *are not incompatible with* (4.26) *and* (4.27)*.*

In the following, we present an example illustrating the first part of Remark 4.3.1.

Example 4.1. *Consider the network depicted in Figure 4.1. For each edge, its length, its upper bound of reduction, and its cost per unit of reduction,* (ℓ_e, u_e, c_e)*, are printed next to the edge. Let* $R = 1, p = 2, B = 0.75$*, and the demand of nodes* $w_i = 1$*,* $w_j = 1$*,* $w_k = 1$*,* $w_q = 1$*,* $w_r = 1000$*,* $w_s = 1000$*. It is straightforward to conclude that the optimal location of the services are the dark nodes, i.e.,* $x_i^* = 1$ *and* $x_q^* = 1$*, and that the optimal edge length reduction is* $\delta^*_{[k,q]} = 0.75$*.*

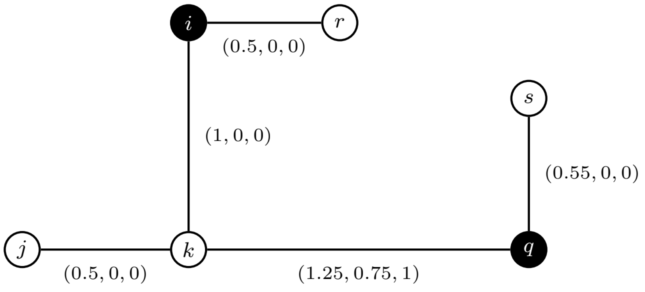


Figure 4.1. *Illustration of incompatibility.*

In this case, from constraints (4.28) *we obtain that* $x_i \leq y_{ki} + y_{kj} + x_k$*. Then,* $y^*_{ki} = 1$*. On the other hand, facility q is the only one that covers node j, then* $y^*_{jq} = 1$*. Moreover, as the path from node j to node q traverses node k, we obtain that* $f^{jq*}_{(j,k)} = 1$*. Therefore, from constraint* (4.26)*, we obtain that* $y^*_{kq} = 1$*. Thus, we have found that these families of constraints are incompatible (*y_{ki} *and* y_{kq} *can not take value one simultaneously due to constraints* (4.2)*).*

Note that the ideas behind constraints (4.28) and (4.29) are practically identical. The reason why constraints (4.29) are not incompatible with (4.26) and (4.27) is that the constraints (4.29) do not force that the node is assigned to the closest service facility before upgrading, instead for given i, j such that

$i \neq j$ and $d(i,j) \leq R$, it enables the node to be assigned to another node whose distance after the most favorable edge length reductions is smaller than or equal to $d(i,j)$. In Espejo et al. (2012) a detailed description of closest service assignment constraints is given.

Observe that the variables dropped from the formulation in the preprocessing phase (Propositions 4.1 and 4.2), can also be removed from the valid inequalities presented in this subsection. In the next section, an alternative formulation for this problem is developed.

4.4 Path Formulation

In this section we present our second formulation for Up-MCLP. It contains fewer variables and constraints than (Flow-Cov). However, this comes at the expense of reducing the scope of preprocessing the model.

This formulation again models paths of length at most R from a customer node i to a service provider. However, in contrast to (Flow-Cov), the path from i is not modeled as a flow but through the immediate successor of i on a path of length $\leq R$ to a facility. For this purpose, we introduce two new binary variables z_{ij} (z_{ji}) for $[i,j] \in E$, such that z_{ij} (z_{ji}) is equal to one if node j (i) is the next node on a path of length at most R from i (j) to a service facility. In Figure 4.2 we illustrate this family of variables where the dark node represents a facility. If i is covered by a facility at q, then also j must be covered. Note that a feasible solution resembles a forest rooted at the facilities.

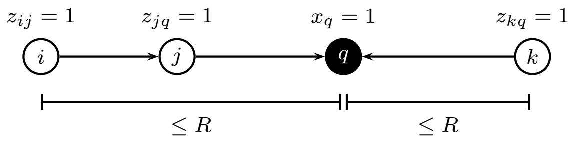


Figure 4.2. *Illustration of z-variables.*

For the sake of clarity, a description of the decision variables used in the formulation is given next.

Decision variables

x_j	1, if there is a facility at node j, and 0, otherwise, for $j \in V$.
z_{ij}	1, if node j is the next node on a path of length $\leq R$ from i to a facility, and 0, otherwise, for $[i,j] \in E$.
d_i	An upper bound of the length of the built path from node i to its assigned service facility, for $i \in V$.
$\delta_e = \delta_{[i,j]}$	The amount of reduction of the length of edge $e = [i,j]$, for $e \in E$.

The formulation for problem Up-MCLP using these variables, (Path), is as follows:

$$
\begin{aligned}
\text{(Path)} \quad \max \quad & \sum_{i \in V} w_i \left(x_i + \sum_{j:[i,j]\in E} z_{ij} \right) \\
\text{s.t.} \quad & \text{(4.1), (4.4), (4.12),} \\
& \sum_{j:[i,j]\in E} z_{ij} + x_i \leq 1, && i \in V, && (4.30) \\
& \sum_{j:[i,j]\in E, j\neq k} z_{ij} + x_i \geq z_{ki}, && [k,i] \in E, && (4.31) \\
& 0 \leq d_i \ \leq \ R \sum_{j:[i,j]\in E} z_{ij}, && i \in V, && (4.32) \\
& d_i \geq d_j + \ell_{[i,j]} z_{ij} - \delta_{[i,j]} - R(1 - z_{ij}), && [i,j] \in E, && (4.33) \\
& 0 \leq \delta_e \leq u_e(z_{ij} + z_{ji}), && e = [i,j] \in E, && (4.34) \\
& z_{ij} \in \{0,1\}, && [i,j] \in E. && (4.35)
\end{aligned}
$$

The family of constraints (4.30) states that each node is assigned to at most one facility or this node is itself a service facility. The family of constraints (4.31) ensures that a node k is not assigned to its service facility through a node i, unless node i is also covered or a facility itself. The family of constraints (4.32) and (4.33) set the value of d_i, a bound on the distance from node i to its facility, if there exists a path of length at most R. We note that (4.33) are equivalent to the well-known Miller-Tucker-Zemlin subtour elimination constraints, extended by our edge length reduction variables. In Figure 4.3, an illustration of constraints (4.33) is depicted, in which the dark node represents a facility. Finally, the families of constraints (4.4) and (4.34) establish the bounds on the amount of length edge reductions.

Note that constraints (4.30) and (4.31) ensure that

$$z_{ij} + z_{ji} \leq 1, \quad [i,j] \in E. \tag{4.36}$$

The following result provides an improvement to the previous formulation.

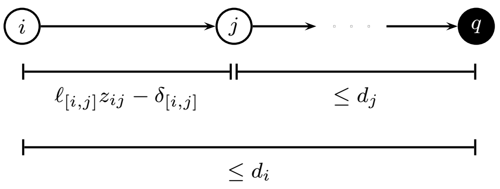


Figure 4.3. *Illustration of constraints* (4.33).

Proposition 4.4. *The formulation (Path) can be enhanced as follows:*

i) The binary condition for the x-variables can be relaxed.

ii) The following are valid inequalities for (Path).

$$d_i \geq \sum_{j:[i,j]\in E} \left(\ell_{[i,j]} - u_{[i,j]}\right) z_{ij}, \quad i \in V. \tag{4.37}$$

iii) Constraints (4.33) *can be reinforced as follows*

$$d_i \geq d_j + \ell_{[i,j]}z_{ij} - \delta_{[i,j]} - R(1 - z_{ij}) + z_{ji}(R - \ell_{[i,j]})^+, \quad [i,j] \in E, \tag{4.38}$$

where $a^+ := \max\{a, 0\}$.

Proof:

The proof of *i)* is very similar to the proof of the first part of Lemma 4.1.

Regarding statement *ii)*, the idea behind these constraints is based on the fact that if a non-facility node is covered, the distance from that node to its assigned facility will be at least the length of the adjacent edge in the path to the service provider, minus the maximally allowed edge length reduction, i.e.,

$$d_i \geq \left(\ell_{[i,j]} - u_{[i,j]}\right) z_{ij}, \quad [i,j] \in E. \tag{4.39}$$

Moreover, each node is linked to at most one other node in the path to its service facility because of constraint (4.30).

In order to prove result *iii)*, we analyze the possible cases. Since the z-variables are binary and constraints (4.36) are satisfied, we get the following four possibilities in the optimal solution for $[i,j] \in E$: a) $z^*_{ij} = z^*_{ji} = 0$, b) $z^*_{ij} = 1, z^*_{ji} = 0$, c.1) $z^*_{ij} = 0$, $z^*_{ji} = 1$, with $R - \ell_{[i,j]} \leq 0$, and c.2) $z^*_{ij} = 0$, $z^*_{ji} = 1$, with $R - \ell_{[i,j]} > 0$. In cases a), b), and c.1) constraints (4.33) are fulfilled. Hence, (4.38) is valid. Therefore, we focus on case c.2). In this case, since the lower bounds of d_i is only given by (4.33), we can assume without loss of generality that (4.33) is satisfied with equality, i.e.:

$$d^*_j = d^*_i + \ell_{[i,j]} - \delta^*_{[i,j]}.$$

Therefore,

$$d_i^* = d_j^* - \ell_{[i,j]} + \delta^*_{[i,j]} \geq d_j^* - \ell_{[i,j]} - \delta^*_{[i,j]}.$$

Hence, since $z_{ij}^* = 0$, $z_{ji}^* = 1$, we have that:

$$d_i^* \geq d_j^* + \ell_{[i,j]} z_{ij}^* - \delta^*_{[i,j]} - R(1 - z_{ij}^*) + z_{ji}^*(R - \ell_{[i,j]}),$$

and consequently the family of inequalities (4.38) holds. Finally, this clearly strengthens the family of constraints (4.33). □

In what follows, we will refer to (Path) as the formulation where the above proposition has been applied. Next, we present some valid inequalities linking x- and z-variables. In this formulation, it is not possible to represent which service is assigned to a given node. Therefore, the ideas of Proposition 4.2 cannot be used. Although it is possible to obtain valid inequalities for this formulation based on constraints (4.28).

Proposition 4.5. *The following families of constraints are valid inequalities for (Path):*

$$x_j \leq \sum_{k:[i,k]\in E, \ell_{[i,k]} - u_{[i,k]} \leq d(i,j)} z_{ik} + x_i, \qquad i \in V, j \in \hat{V}_i, \tag{4.40}$$

$$\sum_{i\in W} \sum_{j \in W:[i,j]\in E} z_{ij} \leq |W| - 1, \qquad W \subset V, 3 \leq |W| \leq n - p, \tag{4.41}$$

$$d_i \geq \sum_{j:[i,j]\in S_1} (d_j - R(1 - z_{ij})) + \sum_{j:[i,j]\in S_2} \left(\ell_{[i,j]} z_{ij} - \delta_{[i,j]}\right), \quad i \in V, S_1, S_2 \subseteq \Gamma_i. \tag{4.42}$$

Proof:

If a facility is open at some node j whose distance to node i before upgrading the network was lower than or equal to the coverage radius (hypothesis of Proposition 4.1), we can be sure that node i will be covered by some facility. Therefore, in order to eliminate possible symmetries, we assume that either i is a facility itself or the immediate successor of node i on its path to a service facility is a node whose distance to i after upgrading can be smaller than or equal to $d(i, j)$. Using the above argument, the family of constraints (4.40) is obtained.

Secondly, we can include the valid inequalities (4.41) to avoid cycles. These inequalities are not required in (Path) because the family of constraints (4.33) or equivalently (4.38) avoid cycles in any feasible solution. However, they can improve the linear relaxation bounds.

Finally, we prove that constraints (4.42) are valid inequalities. Using constraint (4.30), we know that in any feasible solution, for each $i \in V$, there is at most one $j_0 \in V$, $[i, j_0] \in E$ such that $z_{ij_0} = 1$.

On the one hand, if $\sum_{j:[i,j]\in E} z_{ij} = 0$, we obtain that $d_i = 0$ by (4.32). Furthermore, this latter set of constraints ensures that $d_j \leq R$, for $j \in V$. Then, $d_j - R(1 - z_{ij}) \leq 0$, for $[i,j] \in E$. Moreover, since the δ-variables are non-negative and $z_{ij} = 0$, for $j \in V$, we obtain that $\ell_{[i,j]} z_{ij} - \delta_{[i,j]} \leq 0$, for $[i,j] \in E$. Hence, it holds that:

$$0 = d_i \geq \sum_{j:[i,j]\in S_1} (d_j - R(1 - z_{ij})) + \sum_{j:[i,j]\in S_2} \left(\ell_{[i,j]} z_{ij} - \delta_{[i,j]}\right), \quad S_1, S_2 \subseteq \Gamma_i.$$

On the other hand, if exists $j_0 \in V$, $[i, j_0] \in E$, such that $z_{ij_0} = 1$, using (4.33) we know that:

$$d_i \geq d_{j_0} - R(1 - z_{ij_0}) + \ell_{[i,j_0]} z_{ij_0} - \delta_{[i,j_0]}.$$

Furthermore, $d_j - R(1 - z_{ij}) \leq 0$, for $[i,j] \in E$, such that $j \neq j_0$, and $\ell_{[i,j]} z_{ij} - \delta_{[i,j]} \leq 0$, for $[i,j] \in E$, such that $j \neq j_0$. Therefore:

$$d_i \geq \sum_{j:[i,j]\in S_1} (d_j - R(1 - z_{ij})) + \sum_{j:[i,j]\in S_2} \left(\ell_{[i,j]} z_{ij} - \delta_{[i,j]}\right), \quad S_1, S_2 \subseteq \Gamma_i.$$

Thus, we conclude that constraints (4.42) are valid inequalities. $\square$

Observe that in preliminary computational experiments the addition of the following constraints from family (4.41) as cuts in the branching tree was quite effective (more details are provided in Section 4.6):

$$z_{ij} + z_{ji} + z_{jk} + z_{kj} + z_{ik} + z_{ki} \leq 2, \quad [i,j], [j,k], [i,k] \in E. \tag{4.43}$$

Next, we solve the separation problem in the family of constraints (4.42), i.e., given a solution of the LP-relaxation of the formulation, find one or more constraints in family (4.42) that are not satisfied. Hence, sets S_1 and S_2 that maximizes the right-hand-side of the inequality have to be identified. Let $\bar{d}$, $\bar{\delta}$, and $\bar{z}$ be the optimal vectors of values of the d-, δ-, and z-variables, respectively, in a node of the branching tree during the resolution of an instance of formulation (Path). Then, it is straightforward to conclude that one of the following constraints maximizes the right-hand-side of (4.42):

$$d_i \geq \sum_{j:[i,j]\in E, \bar{d}_j > R(1-\bar{z}_{ij})} (d_j - R(1 - z_{ij})) + \sum_{j:[i,j]\in E, \ell_{[i,j]}\bar{z}_{ij} > \bar{\delta}_{[i,j]}} \left(\ell_{[i,j]} z_{ij} - \delta_{[i,j]}\right), i \in V, \tag{4.44}$$

$$d_i \geq \sum_{j:[i,j]\in E, \bar{d}_j + \ell_{[i,j]}\bar{z}_{ij} > R(1-\bar{z}_{ij}) + \bar{\delta}_{[i,j]}} \left(d_j + \ell_{[i,j]} z_{ij} - R(1 - z_{ij}) - \delta_{[i,j]}\right), \quad i \in V. \tag{4.45}$$

In Section 4.6, the performance of this formulation and the effectiveness of the valid inequalities will be analyzed.

4.5 Path-Coverage Formulation

In this section, we introduce a third formulation, which merges components from the first formulation with the second formulation. More precisely, we add the assignment variables y of (Flow-Cov) to (Path). For the sake of clarity, all variables of this formulation are explained below.

Decision variables

x_j	1, if there is a facility at node j, and 0, otherwise, for $j \in V$.
y_{ij}	1, if node i is assigned to a facility at node j, and 0, otherwise, for $i, j \in V, i \neq j$.
z_{ij}	1, if node j is the next node on a path of length $\leq R$ from i to its service facility, and 0, otherwise, for $[i,j] \in E$.
d_i	An upper bound of the length of the built path from node i to its service facility, for $i \in V$.
$\delta_e = \delta_{[i,j]}$	The amount of reduction of the length of edge $e = [i,j]$, for $e \in E$.

Next, we present the formulation (Path-Cov) of problem Up-MCLP using the variables described above:

$$\max \sum_{i \in V} w_i \left(x_i + \sum_{j:[i,j] \in E} z_{ij} \right)$$

s.t. (4.1), (4.3), (4.4), (4.12)–(4.13), (4.30)–(4.32), (4.34), (4.35), (4.38),

$$\sum_{k \in V \setminus \{i\}} y_{ik} = \sum_{j:[i,j] \in E} z_{ij}, \qquad i \in V, \tag{4.46}$$

$$y_{ik} \geq z_{ij} + z_{ji} + y_{jk} - 1, \qquad k \in V \setminus \{i,j\}, [i,j] \in E, \tag{4.47}$$

$$y_{ij} \geq z_{ij} + z_{ji} + x_j - 1, \qquad [i,j] \in E. \tag{4.48}$$

The family of constraints (4.46) establishes that if a node is assigned to a service facility, then there is a path from this node to its facility and vice versa. Constraints (4.47) ensure that if two nodes are on the same path, i.e., $z_{ij} = 1$, they must be assigned to the same facility k. Constraints (4.48) represent the particular case where node j hosts a service provider. Observe that the objective function can also be expressed as follows:

$$\sum_{i \in V} w_i \left(x_i + \sum_{j \in V \setminus \{i\}} y_{ij} \right).$$

But in preliminary computational experiments, we have found that the objective function with the z-variables outperforms the one with the y-variables.

Lemma 4.2. *The binary condition on the x-variables and the y-variables can be relaxed.*

Proof:

The proof of the first part of this lemma is very similar to the proof of the first part of Lemma 4.1. Regarding the integrality condition on the y-variables, since their values are given by the values of the z-variables, it is straightforward to conclude that given an optimal solution it is possible to find another optimal solution in which the y-variables are integer. □

In contrast to (Path), this formulation controls the service facilities to which the nodes are assigned with the y-variables. This information allows us to use a more sophisticated preprocessing phase. For doing so, one of the results presented in Subsection 4.3.1 is used. Under the hypothesis of Proposition 4.2, i.e., a facility at node i will never be assigned to a facility located at node j, for $i, j \in V$, and vice versa, variables y_{ij} and y_{ji} are removed from all the constraints in which they are included (fixed to zero and not included in the formulation to save memory). Furthermore, using the information obtained in the preprocessing phase we can develop new valid inequalities, which are discussed in the next subsection.

4.5.1 Valid inequalities

This subsection is devoted to presenting valid inequalities for formulation (Path-Cov). We start by remarking that the valid inequalities (4.29) obtained for formulation (Flow-Cov) can also be implemented in (Path-Cov). Similarly, all the valid inequalities obtained for formulation (Path) are still valid for (Path-Cov), namely, the families of constraints (4.37) and (4.39)–(4.45). However, the additional information provided by the y-variables in formulation (Path-Cov) can be used to strengthen some of them. The ones that can be enhanced using the covering variables are described below.

First, the lower bound for the d-variables can be improved, i.e., constraint (4.37) can be enhanced as:

$$d_i \geq \sum_{j \in V \setminus \{i\}} d(i, j, \delta^{ij}) y_{ij}, \quad i \in V. \tag{4.49}$$

Recall that $d(i, j, \delta^{ij})$ represents the distance between nodes i and j using the most favorable edge length reductions satisfying the budget constraint (4.4). In what follows, we will refer to (Path-Cov) as the formulation (Path-Cov) in which constraints (4.49) are included.

Finally, we present a new family of valid inequalities that reinforces constraints (4.47). It is based on the fact that if two nodes are linked (the sum of their z-variables is one), then both nodes will be assigned to the same service facility.

Lemma 4.3. *The following are valid inequalities for (Path-Cov):*

$$z_{ij}+z_{ji} \leq \sum_{k\in W, k\neq i} y_{ik}+x_i\mathcal{I}_W(i)+\sum_{k\notin W, k\neq j} y_{jk}+x_j\left(1-\mathcal{I}_W(j)\right),\ [i,j]\in E, W\subseteq V, \tag{4.50}$$

where $\mathcal{I}_W(i)$ is the indicator function, i.e., $\mathcal{I}_W(i)=1$ if $i\in W$ and 0 otherwise.

Note that, for the case $W=\{k\}$, for $k\in V$, we obtain $z_{ij}+z_{ji}\leq y_{ik}+\sum_{t\in V, t\neq k, t\neq j} y_{jt}+x_j$, using constraints (4.30) and (4.46), it holds that $z_{ij}+z_{ji}\leq y_{ik}+\sum_{t\in V,t\neq k,t\neq j} y_{jt}+x_j\leq y_{ik}+1-y_{jk}$. Hence, some constraints of family (4.50) are tighter than (4.47). As the cardinality of (4.50) is exponential, we solve the separation problem in this family of constraints. Therefore, the set W that minimizes the right-hand-side of constraints (4.50) has to be identified. Let $\bar{y}$ ($\bar{x}$) be the optimal vector values of y-variables (x-variables) in a node of the branching tree during the resolution of an instance of formulation (Path-Cov). Then, it is straightforward to conclude that the following constraints minimize the right-hand-side of (4.50).

$$z_{ij}+z_{ji} \leq \sum_{k\in V:\bar{y}_{ik}\leq\bar{y}_{jk}, \bar{y}_{ik}\leq\bar{x}_j} y_{ik}+x_i\mathcal{I}_{\{k:\bar{x}_k\leq\bar{y}_{jk}\}}(i)+\sum_{k\in V:\bar{y}_{jk}<\bar{y}_{ik}, \bar{y}_{jk}<\bar{x}_i} y_{jk}+x_j\mathcal{I}_{\{k:\bar{x}_k<\bar{y}_{ik}\}}(j),\ [i,j]\in E. \tag{4.51}$$

Note that if a pair of nodes satisfies at least one of the conditions of Proposition 4.2, their corresponding y-variables can be removed from all the constraints including the valid inequalities presented in this subsection.

In the following section, the performance of the three proposed formulations for Up-MCLP are compared.

4.6 Computational Results

In this section, we present the results of several computational experiments which compare the performance of the three proposed formulations and show the improvements achieved thanks to the preprocessing phase and the inclusion of the valid inequalities developed throughout the chapter. The experiments were conducted on an Intel(R) Xeon(R) W-2135 CPU 3.70 GHz 32 GB RAM, using CPLEX 20.1.0 in Concert Technology C++ with a time limit of 1800 seconds.

Regarding the preprocessing phase for (Flow-Cov) and (Path-Cov), aiming to find a balance between preprocessing time and the quality of the $d(i,j,\delta^{ij})$ bounds for each pair $i,j\in V$, the following strategy has been implemented. First, we computed the matrix of pairwise shortest distances without upgrading and the matrix of pairwise shortest distances after upgrading all edges to their full maximum ($d(i,j,u)$) using the Floyd-Warshall algorithm. Then, we checked if the hypothesis of Proposition 4.1 or if the hypotheses *i)-iii)* of Proposition 4.2 are fulfilled. If either of these conditions is satisfied for a pair $i, j \in V$, we removed the corresponding variables and constraints and we used $d(i, j, u)$ as $d(i, j, \delta^{ij})$ in the valid inequalities that are required (as e.g. constraints (4.29)). This could be done because $d(i, j, u)$ is a lower bound of $d(i, j, \delta^{ij})$. If neither of these conditions were fulfilled for a given pair $i, j \in V$, we solved the linear relaxation of $\left(P_{d(i,j,\delta^{ij})}\right)$. The minimum between its optimal solution and $d(i, j, u)$ is the value that we used as $d(i, j, \delta^{ij})$ in the corresponding valid inequalities. As before, this can be done because both values are lower bounds of $d(i, j, \delta^{ij})$.

In preliminary computational experiments, we checked the performance of the alternative formulation (Flow-Cov) for Up-MCLP and the valid inequalities to identify which ones performed best in each formulation. After this preliminary choice, we conclude that the best formulations are:

a) Formulation (Flow-Cov) where constraints (4.6), (4.7), and (4.11) have been replaced with (4.17), (4.18), and (4.19) and the family of constraints (4.25) is included, named (Flow-Cov) for short.
b) Formulation (Path) where constraints (4.37) are included and constraints (4.33) have been reinforced by constraints (4.38). In what follows, we call it formulation (Path).
c) Formulation (Path) with constraints (4.40) as valid inequalities and (4.43) as particular case of (4.41) in a pool of user cuts, named (Path) + VI for short.
d) Formulation (Path-Cov) where constraints (4.49) are included, we call it formulation (Path-Cov).
e) Formulation (Path-Cov) including constraints (4.29) and (4.43) as particular case of (4.41) in a pool of user cuts, named (Path-Cov) + VI for short.

We would like to remark that including the constraints (4.44), (4.45), and (4.51) in the branching tree was effective, as the number of nodes in which the instances were solved decreased. However, this procedure is time-consuming, causing that even though the instances were solved on fewer nodes the overall computation time increased.

The rest of the section is structured as follows. First, the data used in the computational experiments are described. Second, the advantages of the preprocessing phase are shown. Then, the following subsections compare the different formulations with and without valid inequalities in complete graphs and in sparse graphs, respectively. These subsections illustrate the great value of the preprocessing phase and the addition of valid inequalities.

4.6.1 Data

The computational experiments were carried out on two different types of networks.

First, we generated instances adapting the procedure used in ReVelle et al. (2008); Cordeau et al. (2019), among others. Nodes were given by points whose coordinates followed a uniform distribution over [0,30]. Then, we computed the complete graph where the length of the edges is the Euclidean distance between the nodes. We named these instances as "graph" followed by the number of vertices, e.g., "graph30" is a complete graph with 30 nodes and 435 edges.

Secondly, we used the uncapacitated p-median datasets from the OR-Library, called pmed, see Beasley (1990). As said in the documentation of these datasets, Floyd's algorithm was applied to obtain a symmetric allocation cost matrix that satisfied the triangle inequality. The main difference with respect to the previous datasets is that the p-median instances are sparser graphs (the number of edges is $n^2/50$).

The parameters have been chosen as described below. The number of facilities, p, was proportional to the number of vertices, i.e., $p \in \{1, n/10, n/20\}$. The node weights or demands, w_i for $i \in V$, were integers randomly generated between 1 and 100. We tested three different coverage radii, R, such that we could cover approximately $50\%, 60\%$, and 70% of the total demand when solving the maximal covering location problem without upgrading (DT_{MCLP}), i.e.,

$$R \in \{R(50\%DT_{\text{MCLP}}), R(60\%DT_{\text{MCLP}}), R(70\%DT_{\text{MCLP}})\}.$$

Upgrading costs, c_e, for $e \in E$, were randomly generated between 1 and 3. The upper bounds u_e, for $e \in E$, were randomly generated from $(0, 30\%\ell_e)$, for $e \in E$. Then, the length of the edges was modified as $\ell_e + u_e$, for $e \in E$. This implies that the instances satisfy the triangle inequality when the full discount is applied in all edges. Finally, the budget B was computed as follows. First, we sorted the upgrade costs $c_e u_e$, for $e \in E$, in non-increasing order. Let ρ be a permutation of set E such that

$$c_{e_{\rho(1)}} u_{e_{\rho(1)}} \geq c_{e_{\rho(2)}} u_{e_{\rho(2)}} \geq \ldots \geq c_{e_{\rho(m)}} u_{e_{\rho(m)}}.$$

Then, since we are constructing a forest with p components (as seen in Section 4.4), we can assume that at most $n-p$ edges will be upgraded. Therefore, we computed the maximum required budget for upgrading the most expensive edges,

$$B_{max} = \sum_{t=1}^{n-p} u_{e_{\sigma(t)}} c_{e_{\sigma(t)}},$$

and selected $B \in \{0.5\% B_{max}, 1\% B_{max}, 5\% B_{max}\}$.

4.6.2 Preprocessing phase

In this subsection, we show the enhancements provided by the preprocessing, i.e., Propositions 4.1 and 4.2. For doing so, we solve (Flow-Cov) and (Path-Cov) with and without preprocessing.

As an illustrative example, we include the results for graph40 in Table 4.2. The performance was similar in the rest of the datasets. The results are the average over five instances generated with the same procedure, varying only the random seed for the generator. The first column indicates the name of the dataset, the number of nodes and the number of edges. Next, the percentage of the maximal budget ($B\%$), and the number of located facilities are depicted (p), followed by the approximate percentage of covered demand in the MCLP without upgrading using this radius ($R\%$). The following four columns describe information about (Flow-Cov) without preprocessing. The first one shows the average time (in seconds) of solving the corresponding five instances. Then, the following column of this group depicts the MIP relative gap reported by CPLEX ($G\%$) and in brackets the number of instances solved to optimality within the time limit. Next, it is provided the best solution gap, ($G^t_{BS}\%$), computed as follows:

$$\mathrm{G}^t_{BS}\% = \frac{\mathrm{BS}^t - \mathrm{BS}}{\mathrm{BS}^t} \cdot 100,$$

where BS is the best MIP objective value found within the time limit by the formulation and BS^t is the best MIP solution value found within the time limit across all formulations. Finally, it is shown the linear relaxation gap, ($\mathrm{G}^t_{LP}\%$), computed as follows:

$$\mathrm{G}^t_{LP}\% = \frac{\mathrm{LP} - \mathrm{BS}^t}{\mathrm{BS}^t} \cdot 100,$$

where LP is the optimal solution value of the linear relaxation of the formulation. Note that $\mathrm{G}^t_{LP}\%$ enables us to compare the linear relaxation of the formulations with each other (it could be possible that $G\%$ is greater than $\mathrm{G}^t_{LP}\%$). The following blocks of columns depict information about the rest of formulations, (Flow-Cov) with preprocessing and (Path-Cov) with and without preprocessing. Observe that the blocks corresponding to formulations with preprocessing include eight columns. The first column of these blocks reports the average time of the preprocessing phase in seconds, the next one shows the average total time (in seconds) of solving the corresponding five instances including the preprocessing time. The third, the fourth, and the fifth columns report the average $G\%$, G^t_{BS}, and G^t_{LP} respectively. Then, the average percentage of reduction in the number of constraints ($R_c\%$), variables ($R_v\%$), and binary variables ($R_{bv}\%$) are depicted. The percentage of reduction in the number of constrains is computed for formulation (Flow-Cov) as follows:

$$R_c\% = \frac{\#\text{const. of (Flow-Cov) without prep.} - \#\text{const. of (Flow-Cov) with prep.}}{\#\text{const. of (Flow-Cov) without prep.}} \cdot 100.$$

The others are calculated analogously.

Data	$B\%$	p	$R\%$	(Flow-Cov)												(Path-Cov)											
				Without preprocesing				With preprocesing								Without preprocesing				With preprocesing							
				t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	$R_c\%$	$R_v\%$	$R_{bv}\%$	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	$R_c\%$	$R_v\%$	$R_{bv}\%$
graph40 $\lvert V\rvert = 40, \lvert E\rvert = 780$	0.5	1	50	1042.5	0.0(5)	0.0	93.2	0.1	0.7	0.0(5)	0.0	2.3	97.0	97.0	96.9	1807.9	99.9(0)	5.7	93.2	0.1	2.3	0.0(5)	0.0	2.4	69.9	46.2	52.5
			60	1490.0	15.4(3)	0.0	52.3	0.1	3.8	0.0(5)	0.0	0.7	95.8	95.7	95.7	1804.2	74.8(0)	12.4	52.3	0.2	8.8	0.0(5)	0.0	0.7	63.1	37.6	43.5
			70	1556.8	7.3(4)	0.0	35.4	0.2	2.0	0.0(5)	0.0	1.0	96.0	96.0	95.9	1807.5	59.6(0)	13.4	35.4	0.2	724.2	1.0(3)	0.0	1.0	58.3	32.0	37.5
		2	50	1578.7	74.4(1)	3.9	87.0	0.1	0.4	0.0(5)	0.0	1.2	98.4	98.4	98.4	61.3	0.0(5)	0.0	88.7	0.1	0.4	0.0(5)	0.0	3.3	80.8	68.7	74.3
			60	1801.2	64.1(0)	4.2	57.0	0.1	0.4	0.0(5)	0.0	0.9	97.8	97.8	97.7	1093.2	4.3(3)	0.0	57.0	0.1	0.4	0.0(5)	0.0	0.9	77.5	60.0	66.2
			70	1801.7	40.5(0)	4.4	34.1	0.1	0.7	0.0(5)	0.0	1.8	97.9	97.8	97.8	1769.3	17.5(1)	1.2	34.1	0.1	2.4	0.0(5)	0.0	1.8	74.3	53.1	59.5
		4	50	2.9	0.0(5)	0.0	39.4	0.1	0.1	0.0(5)	0.0	1.0	99.1	99.1	99.0	0.3	0.0(5)	0.0	67.2	0.1	0.1	0.0(5)	0.0	2.8	86.7	83.4	87.3
			60	27.5	0.0(5)	0.0	48.6	0.1	0.2	0.0(5)	0.0	0.8	99.0	98.9	98.9	8.4	0.0(5)	0.0	56.8	0.1	0.2	0.0(5)	0.0	1.7	84.9	78.5	83.1
			70	1546.9	25.6(1)	0.6	34.3	0.1	1.0	0.0(5)	0.0	3.9	98.3	98.2	98.2	305.4	0.0(5)	0.0	34.3	0.1	0.6	0.0(5)	0.0	4.4	81.9	70.8	76.3
	1	1	50	1384.9	18.2(4)	0.1	86.0	0.1	1.1	0.0(5)	0.0	0.4	96.0	96.0	96.0	1807.3	98.4(0)	6.6	86.0	0.1	3.6	0.0(5)	0.0	0.5	68.8	45.5	51.7
			60	1611.6	17.1(3)	0.0	51.8	0.1	1.5	0.0(5)	0.0	0.4	95.2	95.2	95.1	1809.0	71.3(0)	10.7	51.8	0.1	5.4	0.0(5)	0.0	0.4	62.5	37.2	43.1
			70	1801.9	27.9(0)	0.0	34.0	0.2	2.6	0.0(5)	0.0	1.5	94.7	94.7	94.7	1816.8	58.6(0)	13.8	34.0	0.2	135.3	0.0(5)	0.0	1.5	56.9	31.3	36.7
		2	50	1579.3	69.8(1)	1.9	84.9	0.1	0.4	0.0(5)	0.0	0.9	98.1	98.1	98.0	95.8	0.0(5)	0.0	86.5	0.1	0.3	0.0(5)	0.0	1.4	80.4	68.4	74.1
			60	1801.4	63.8(0)	6.1	53.2	0.1	0.8	0.0(5)	0.0	2.8	96.8	96.8	96.8	1609.6	12.4(1)	0.0	53.2	0.1	1.7	0.0(5)	0.0	2.8	76.5	59.3	65.4
			70	1801.6	43.3(0)	7.6	31.6	0.1	1.9	0.0(5)	0.0	3.0	96.7	96.7	96.6	1804.5	23.9(0)	1.2	31.6	0.1	41.9	0.0(5)	0.0	3.1	73.1	52.3	58.6
		4	50	2.9	0.0(5)	0.0	39.4	0.1	0.1	0.0(5)	0.0	1.0	99.1	99.1	99.0	0.3	0.0(5)	0.0	67.2	0.1	0.1	0.0(5)	0.0	2.8	86.7	83.4	87.3
			60	27.6	0.0(5)	0.0	48.6	0.1	0.2	0.0(5)	0.0	0.8	99.0	98.9	98.9	8.4	0.0(5)	0.0	56.8	0.1	0.2	0.0(5)	0.0	1.7	84.9	78.5	83.1
			70	1547.3	25.7(1)	0.6	34.3	0.1	1.0	0.0(5)	0.0	3.9	98.3	98.2	98.2	301.5	0.0(5)	0.0	34.3	0.1	0.6	0.0(5)	0.0	4.4	81.9	70.8	76.3
	5	1	50	1805.7	88.6(0)	3.3	82.1	0.1	1.7	0.0(5)	0.0	0.3	95.0	95.0	95.0	1807.6	87.3(0)	3.3	82.1	0.1	0.6	0.0(5)	0.0	0.3	67.7	44.9	51.0
			60	1805.2	46.(0)	0.4	45.4	0.2	2.1	0.0(5)	0.0	0.5	94.3	94.3	94.3	1807.6	68.3(0)	12.8	45.4	0.1	16.3	0.0(5)	0.0	0.5	61.6	36.7	42.5
			70	1802.5	43.1(0)	7.4	31.0	0.2	1.9	0.0(5)	0.0	0.0	94.2	94.2	94.1	1811.1	44.1(0)	7.9	31.0	0.2	3.6	0.0(5)	0.0	0.0	56.3	31.0	36.3
		2	50	1800.6	106.7(0)	12.8	79.8	0.1	0.4	0.0(5)	0.0	1.1	97.4	97.4	97.3	211.0	0.0(5)	0.0	81.6	0.1	0.4	0.0(5)	0.0	1.1	79.7	67.9	73.5
			60	1801.3	61.7(0)	8.6	47.6	0.1	1.1	0.0(5)	0.0	0.3	96.1	96.0	96.0	1804.8	14.4(0)	0.0	47.6	0.1	0.7	0.0(5)	0.0	0.3	75.7	58.7	64.8
			70	1802.0	41.7(0)	9.9	27.6	0.1	1.5	0.0(5)	0.0	0.0	95.5	95.4	95.4	1804.3	21.3(0)	1.5	27.6	0.1	2.0	0.0(5)	0.0	0.0	71.8	51.4	57.6
		4	50	3.1	0.0(5)	0.0	36.1	0.1	0.1	0.0(5)	0.0	1.2	98.9	98.8	98.8	0.3	0.0(5)	0.0	63.4	0.1	0.1	0.0(5)	0.0	1.3	86.4	83.1	87.1
			60	512.0	0.0(5)	0.0	45.9	0.1	0.2	0.0(5)	0.0	0.0	98.5	98.4	98.4	6.4	0.0(5)	0.0	54.1	0.1	0.2	0.0(5)	0.0	0.0	84.4	78.1	82.7
			70	1801.1	35.4(0)	6.0	27.1	0.1	1.0	0.0(5)	0.0	1.5	97.4	97.3	97.3	463.7	1.4(4)	0.0	27.1	0.1	0.8	0.0(5)	0.0	1.5	80.8	70.1	75.5

Table 4.2. *Performance of formulations (Flow-Cov) and (Path-Cov) with and without preprocessing.*

As can be appreciated in Table 4.2, the preprocessing phase yields a huge reduction in computation time. For example, using formulation (Flow-Cov) without preprocessing, only 53 instances are solved to optimality within the time limit, while using formulation (Flow-Cov) with preprocessing, all instances are solved (135) and the average time of solving each instance (including the time of preprocessing) is 1.1 seconds. Furthermore, the reduction in the number of constraints, variables and binary variables is also very large, approximately 97% on average. In formulation (Path-Cov), the reduction of time in the resolution process and the reduction of the size of the problem are also very substantial.

Based on the above results, we conclude that the preprocessing phase presented in the chapter is extremely useful and effective. Therefore, in the subsequent computational experiments, the preprocessing phase is included.

4.6.3 Results for complete graphs

In this subsection, we compare the proposed formulations for complete graphs highlighting the effectiveness of the valid inequalities developed.

The results of the smaller datasets (graph30 and graph40) are depicted in Table 4.3. As before, the provided results are the average over five instances generated with the same procedure, varying only the random seed for the generator. The table describes information about (Flow-Cov), (Path), (Path) + VI and (Path-Cov), and its structure is similar to that of Table 4.2. Observe that the blocks corresponding to the (Path) and (Path) + VI formulations have no preprocessing time, since we did not provide a preprocessing for these formulations. Note also that the results of (Path-Cov) + VI are not included because they are really similar to (Path-Cov). The differences between these formulations will be shown in instances with a larger number of nodes and edges. Moreover, since several of the valid inequalities are included as cuts in the branching tree, the linear relaxation gap $\mathrm{G}^t_{LP}\%$ of the formulation with valid inequalities is practically the same as the one for the formulation without valid inequalities. Therefore, they do not appear again in the table. Finally, the formulation that provided the smallest average total time is highlighted. If any of the five instances were not solved to optimality, the formulation that solved more instances is shown in bold.

The results depicted in Table 4.3 show that formulations (Flow-Cov) and (Path-Cov) outperform (Path) and (Path) + VI (the resolution times, the number of instances solved, the MIP relative gap, the best solution gap, and the linear relaxation gap of these formulations are worse). Moreover, it is

				(Flow-Cov)					(Path)				(Path) + VI			(Path-Cov)				
Data	$B\%$	p	$R\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{total}	$G\%$	$G^t_{BS}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$
graph30, $\lvert V\rvert = 30, \lvert E\rvert = 435$	0.5	1	50	**0.1**	**0.5**	**0.0(5)**	**0.0**	**2.2**	54.6	0.0(5)	0.0	84.8	13.2	0.0(5)	0.0	0.1	0.7	0.0(5)	0.0	2.2
			60	**0.1**	**0.4**	**0.0(5)**	**0.0**	**0.9**	533.2	6.1(4)	0.0	50.9	144.1	0.0(5)	0.0	0.1	1.4	0.0(5)	0.0	0.9
			70	**0.1**	**0.5**	**0.0(5)**	**0.0**	**0.7**	1162.9	14.4(2)	0.2	34.7	865.0	8.2(3)	0.0	0.1	0.8	0.0(5)	0.0	0.7
		2	50	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.6**	0.3	0.0(5)	0.0	83.4	0.3	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.6**
			60	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.0**	0.7	0.0(5)	0.0	54.3	0.5	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.0**
			70	**0.1**	**0.3**	**0.0(5)**	**0.0**	**1.0**	5.0	0.0(5)	0.0	36.0	2.9	0.0(5)	0.0	**0.0**	**0.3**	**0.0(5)**	**0.0**	**1.1**
		3	50	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.4**	**0.1**	**0.0(5)**	**0.0**	**79.2**	**0.1**	**0.0(5)**	**0.0**	**0.0**	**0.1**	**0.0(5)**	**0.0**	**3.0**
			60	**0.0**	**0.1**	**0.0(5)**	**0.0**	**2.6**	0.3	0.0(5)	0.0	53.8	0.3	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**2.7**
			70	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.7**	0.8	0.0(5)	0.0	32.0	0.7	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.7**
	1	1	50	**0.1**	**0.8**	**0.0(5)**	**0.0**	**4.1**	102.7	0.0(5)	0.0	77.2	49.7	0.0(5)	0.0	0.1	1.9	0.0(5)	0.0	4.2
			60	**0.1**	**0.6**	**0.0(5)**	**0.0**	**0.7**	620.2	2.9(4)	0.0	45.3	479.4	1.3(4)	0.0	0.1	2.2	0.0(5)	0.0	0.7
			70	**0.1**	**0.8**	**0.0(5)**	**0.0**	**2.0**	1370.0	10.0(3)	0.0	32.4	967.0	11.9(3)	0.0	0.1	4.1	0.0(5)	0.0	2.1
		2	50	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.6**	0.3	0.0(5)	0.0	83.4	0.3	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.6**
			60	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.3**	0.7	0.0(5)	0.0	52.8	0.5	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.3**
			70	**0.1**	**0.6**	**0.0(5)**	**0.0**	**1.5**	11.7	0.0(5)	0.0	32.5	6.1	0.0(5)	0.0	0.0	0.9	0.0(5)	0.0	1.5
		3	50	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.4**	**0.1**	**0.0(5)**	**0.0**	**79.2**	**0.1**	**0.0(5)**	**0.0**	**0.0**	**0.1**	**0.0(5)**	**0.0**	**3.0**
			60	**0.0**	**0.1**	**0.0(5)**	**0.0**	**2.6**	0.3	0.0(5)	0.0	53.8	0.3	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**2.7**
			70	**0.1**	**0.1**	**0.0(5)**	**0.0**	**0.7**	0.8	0.0(5)	0.0	32.0	0.7	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**0.7**
	5	1	50	0.1	0.6	0.0(5)	0.0	0.0	254.6	0.0(5)	0.0	66.2	98.3	0.0(5)	0.0	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.0**
			60	0.1	0.7	0.0(5)	0.0	0.0	1097.0	5.5(3)	0.0	42.9	653.4	3.7(4)	0.0	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.0**
			70	**0.1**	**0.8**	**0.0(5)**	**0.0**	**0.0**	1384.6	13.3(2)	0.0	28.8	1329.7	14.(2)	0.0	0.1	1.5	0.0(5)	0.0	0.0
		2	50	0.0	0.2	0.0(5)	0.0	2.2	0.4	0.0(5)	0.0	70.6	0.4	0.0(5)	0.0	**0.0**	**0.1**	**0.0(5)**	**0.0**	**3.6**
			60	**0.0**	**0.2**	**0.0(5)**	**0.0**	**1.4**	1.0	0.0(5)	0.0	49.4	0.8	0.0(5)	0.0	**0.0**	**0.2**	**0.0(5)**	**0.0**	**1.4**
			70	0.1	0.5	0.0(5)	0.0	1.2	20.4	0.0(5)	0.0	30.7	11.1	0.0(5)	0.0	**0.1**	**0.4**	**0.0(5)**	**0.0**	**1.2**
		3	50	**0.0**	**0.1**	**0.0(5)**	**0.0**	**1.1**	**0.1**	**0.0(5)**	**0.0**	**72.1**	**0.1**	**0.0(5)**	**0.0**	**0.0**	**0.1**	**0.0(5)**	**0.0**	**1.8**
			60	**0.1**	**0.2**	**0.0(5)**	**0.0**	**3.2**	0.5	0.0(5)	0.0	47.4	0.5	0.0(5)	0.0	**0.0**	**0.2**	**0.0(5)**	**0.0**	**4.6**
			70	0.1	0.3	0.0(5)	0.0	4.0	1.3	0.0(5)	0.0	26.2	0.9	0.0(5)	0.0	**0.0**	**0.2**	**0.0(5)**	**0.0**	**4.2**
graph40, $\lvert V\rvert = 40, \lvert E\rvert = 780$	0.5	1	50	**0.1**	**0.7**	**0.0(5)**	**0.0**	**2.3**	1395.1	31.1(2)	0.0	93.2	1089.2	39.(3)	6.1	0.1	2.3	0.0(5)	0.0	2.4
			60	**0.1**	**3.8**	**0.0(5)**	**0.0**	**0.7**	1801.0	48.3(0)	5.1	52.3	1802.6	45.9(0)	5.8	0.2	8.8	0.0(5)	0.0	0.7
			70	**0.2**	**2.0**	**0.0(5)**	**0.0**	**1.0**	1801.6	43.6(0)	8.2	35.4	1803.7	62.1(0)	16.9	0.2	724.2	1.0(3)	0.0	1.0
		2	50	**0.1**	**0.4**	**0.0(5)**	**0.0**	**1.2**	6.2	0.0(5)	0.0	88.6	4.5	0.0(5)	0.0	**0.1**	**0.4**	**0.0(5)**	**0.0**	**3.3**
			60	**0.1**	**0.4**	**0.0(5)**	**0.0**	**0.9**	143.6	0.0(5)	0.0	57.0	53.9	0.0(5)	0.0	**0.1**	**0.4**	**0.0(5)**	**0.0**	**0.9**
			70	**0.1**	**0.7**	**0.0(5)**	**0.0**	**1.8**	878.8	2.3(3)	0.0	34.1	549.8	1.1(4)	0.0	0.1	2.4	0.0(5)	0.0	1.8
		4	50	**0.1**	**0.1**	**0.0(5)**	**0.0**	**1.0**	0.2	0.0(5)	0.0	67.2	0.2	0.0(5)	0.0	**0.1**	**0.1**	**0.0(5)**	**0.0**	**2.8**
			60	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.8**	0.8	0.0(5)	0.0	56.8	0.6	0.0(5)	0.0	**0.1**	**0.2**	**0.0(5)**	**0.0**	**1.7**
			70	0.1	1.0	0.0(5)	0.0	3.9	6.9	0.0(5)	0.0	34.3	4.4	0.0(5)	0.0	**0.1**	**0.6**	**0.0(5)**	**0.0**	**4.4**
	1	1	50	**0.1**	**1.1**	**0.0(5)**	**0.0**	**0.4**	1636.2	36.3(1)	3.0	86.0	1367.6	32.2(3)	3.2	0.1	3.6	0.0(5)	0.0	0.5
			60	**0.1**	**1.5**	**0.0(5)**	**0.0**	**0.4**	1800.5	48.2(0)	7.1	51.8	1801.1	55.8(0)	9.7	0.1	5.4	0.0(5)	0.0	0.4
			70	**0.2**	**2.6**	**0.0(5)**	**0.0**	**1.5**	1801.0	42.2(0)	8.1	34.0	1802.9	48.6(0)	11.9	0.2	135.3	0.0(5)	0.0	1.5
		2	50	0.1	0.4	0.0(5)	0.0	0.9	10.1	0.0(5)	0.0	86.5	5.9	0.0(5)	0.0	**0.1**	**0.3**	**0.0(5)**	**0.0**	**1.4**
			60	**0.1**	**0.8**	**0.0(5)**	**0.0**	**2.8**	255.7	0.0(5)	0.0	53.2	168.2	0.0(5)	0.0	0.1	1.7	0.0(5)	0.0	2.8
			70	**0.1**	**1.9**	**0.0(5)**	**0.0**	**3.0**	1058.5	3.1(3)	0.0	31.6	817.9	1.8(3)	0.0	0.1	41.9	0.0(5)	0.0	3.1
		4	50	**0.1**	**0.1**	**0.0(5)**	**0.0**	**1.0**	0.2	0.0(5)	0.0	67.2	0.2	0.0(5)	0.0	**0.1**	**0.1**	**0.0(5)**	**0.0**	**2.8**
			60	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.8**	0.8	0.0(5)	0.0	56.8	0.6	0.0(5)	0.0	**0.1**	**0.2**	**0.0(5)**	**0.0**	**1.7**
			70	0.1	1.0	0.0(5)	0.0	3.9	6.9	0.0(5)	0.0	34.3	4.4	0.0(5)	0.0	**0.1**	**0.6**	**0.0(5)**	**0.0**	**4.4**
	5	1	50	0.1	1.7	0.0(5)	0.0	0.3	1549.1	37.7(1)	0.0	82.1	1488.2	36.2(1)	0.0	**0.1**	**0.6**	**0.0(5)**	**0.0**	**0.3**
			60	**0.2**	**2.1**	**0.0(5)**	**0.0**	**0.5**	1800.6	50.4(0)	8.8	45.4	1806.5	44.6(0)	7.3	0.1	16.3	0.0(5)	0.0	0.5
			70	**0.2**	**1.9**	**0.0(5)**	**0.0**	**0.0**	1802.9	38.6(0)	7.2	31.0	1803.4	41.2(0)	8.6	0.2	3.6	0.0(5)	0.0	0.0
		2	50	**0.1**	**0.4**	**0.0(5)**	**0.0**	**1.1**	15.7	0.0(5)	0.0	81.5	10.5	0.0(5)	0.0	**0.1**	**0.4**	**0.0(5)**	**0.0**	**1.1**
			60	0.1	1.1	0.0(5)	0.0	0.3	499.0	0.2(4)	0.0	47.6	234.3	0.0(5)	0.0	**0.1**	**0.7**	**0.0(5)**	**0.0**	**0.3**
			70	**0.1**	**1.5**	**0.0(5)**	**0.0**	**0.0**	1468.1	4.6(1)	0.1	27.6	1202.5	2.9(2)	0.1	0.1	2.0	0.0(5)	0.0	0.0
		4	50	**0.1**	**0.1**	**0.0(5)**	**0.0**	**1.2**	0.2	0.0(5)	0.0	63.4	0.2	0.0(5)	0.0	**0.1**	**0.1**	**0.0(5)**	**0.0**	**1.3**
			60	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.0**	0.8	0.0(5)	0.0	54.1	0.7	0.0(5)	0.0	**0.1**	**0.2**	**0.0(5)**	**0.0**	**0.0**
			70	0.1	1.0	0.0(5)	0.0	1.5	39.4	0.0(5)	0.0	27.1	13.5	0.0(5)	0.0	**0.1**	**0.8**	**0.0(5)**	**0.0**	**1.5**

Table 4.3. *Performance of formulations (Flow-Cov), (Path), (Path) + VI, (Path-Cov) on graph30 and graph40.*

clear from these results that the valid inequalities improve the performance of formulation (Path) as shown in the number of instances solved to optimality (208 instances with respect to 217 instances) and the average total time (489 seconds with respect to 416 seconds). However, this improvement is not large enough to make this formulation competitive with respect to formulations (Flow-Cov) and (Path-Cov) on complete graphs. Nevertheless, it can be appreciated that formulations (Path) and (Path) + VI in graph40 solve many more instances than formulations (Flow-Cov) and (Path-Cov) without preprocessing (85 and 91 instances with respect to 53 and 64 instances), as can be seen in Table 4.2.

In Table 4.4, a second set of computational experiments is reported. Here, datasets of larger size (graph100 and graph120) are solved so that formulations (Flow-Cov), (Path-Cov) and (Path-Cov) + VI can be compared. Table 4.4 has the same structure as Table 4.3, but now (Path-Cov) + VI is included whereas (Path) and (Path) + VI are not. For the purpose of a clearer comparison of these formulations, the performance profile graph of the number of solved instances is depicted in Figure 4.4.

				(Flow-Cov)					(Path-Cov)					(Path-Cov) + VI			
Data	$B\%$	p	$R\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$
graph100, $\lvert V\rvert = 100, \lvert E\rvert = 4950$	0.5	1	50	**6.8**	**1512.9**	**5.7(1)**	**0.0**	**6.9**	6.8	1807.1	10.6(0)	3.2	7.0	6.8	1564.4	7.9(1)	0.8
			60	**7.1**	**1716.5**	**1.9(1)**	**0.0**	**2.4**	7.3	1807.5	3.3(0)	0.8	2.4	7.2	1807.5	2.6(0)	0.2
			70	**8.1**	**1770.6**	**1649.1(1)**	**14.7**	**3.2**	7.9	1808.3	4.4(0)	1.1	3.3	7.9	1808.4	4.8(0)	1.4
		5	50	1.2	12.0	0.0(5)	0.0	1.6	1.2	7.5	0.0(5)	0.0	1.7	**1.2**	**5.7**	**0.0(5)**	**0.0**
			60	**1.3**	**455.3**	**0.3(4)**	**0.0**	**4.8**	1.3	710.2	0.0(4)	0.0	5.1	1.3	654.6	0.0(4)	0.0
			70	**1.6**	**624.0**	**0.0(4)**	**0.0**	**2.5**	1.6	1101.1	1.5(2)	0.3	2.5	1.6	1096.8	1.5(2)	0.5
		10	50	1.1	2.0	0.0(5)	0.0	2.8	**1.1**	**1.4**	**0.0(5)**	**0.0**	**3.8**	**1.1**	**1.4**	**0.0(5)**	**0.0**
			60	1.2	8.4	0.0(5)	0.0	3.5	1.2	3.0	0.0(5)	0.0	4.0	**1.2**	**2.1**	**0.0(5)**	**0.0**
			70	1.2	19.7	0.0(5)	0.0	3.3	1.2	6.5	0.0(5)	0.0	3.6	**1.2**	**6.2**	**0.0(5)**	**0.0**
	1	1	50	7.2	1285.7	3.3(2)	0.0	3.6	**7.0**	**1262.4**	**4.1(2)**	**0.4**	**3.7**	6.9	1265.4	3.7(2)	0.0
			60	**7.1**	**1383.9**	**0.6(3)**	**0.0**	**0.6**	7.4	1557.2	1.5(2)	0.8	0.6	7.3	1378.9	0.9(2)	0.3
			70	**8.1**	**1643.4**	**16.3(2)**	**11.1**	**0.9**	8.1	1692.6	1.6(1)	0.7	0.9	8.1	1710.9	2.0(1)	1.1
		5	50	1.2	16.0	0.0(5)	0.0	1.3	**1.2**	**6.7**	**0.0(5)**	**0.0**	**1.3**	1.2	6.9	0.0(5)	0.0
			60	**1.3**	**413.9**	**0.3(4)**	**0.0**	**3.9**	1.3	473.8	0.3(4)	0.0	4.1	1.3	448.4	0.3(4)	0.0
			70	**1.6**	**244.2**	**0.0(5)**	**0.0**	**1.8**	1.7	1091.4	0.8(2)	0.1	1.9	1.6	1087.7	1.5(2)	0.4
		10	50	1.1	1.9	0.0(5)	0.0	2.8	1.1	**1.4**	**0.0(5)**	**0.0**	**3.8**	**1.1**	**1.4**	**0.0(5)**	**0.0**
			60	1.2	8.7	0.0(5)	0.0	3.5	1.2	2.8	0.0(5)	0.0	3.7	**1.2**	**2.2**	**0.0(5)**	**0.0**
			70	1.2	22.1	0.0(5)	0.0	2.7	1.2	14.8	0.0(5)	0.0	2.8	**1.2**	**14.1**	**0.0(5)**	**0.0**
	5	1	50	6.9	747.2	0.0(5)	0.0	0.0	7.1	284.7	0.0(5)	0.0	0.0	**7.0**	**278.3**	**0.0(5)**	**0.0**
			60	7.3	1064.1	0.0(5)	0.0	0.0	7.3	432.2	0.0(5)	0.0	0.0	**7.2**	**368.9**	**0.0(5)**	**0.0**
			70	7.9	1148.0	0.0(5)	0.0	0.0	**8.1**	**513.4**	**0.0(5)**	**0.0**	**0.0**	8.1	632.4	0.0(5)	0.0
		5	50	1.2	4.0	0.0(5)	0.0	0.0	**1.2**	**1.7**	**0.0(5)**	**0.0**	**0.0**	**1.2**	**1.7**	**0.0(5)**	**0.0**
			60	1.3	8.6	0.0(5)	0.0	0.0	**1.3**	**4.9**	**0.0(5)**	**0.0**	**0.0**	1.2	5.3	0.0(5)	0.0
			70	**1.7**	**21.1**	**0.0(5)**	**0.0**	**0.1**	1.7	403.0	0.0(4)	0.0	0.1	1.6	22.9	0.0(5)	0.0
		10	50	1.1	1.7	0.0(5)	0.0	0.9	1.2	1.4	0.0(5)	0.0	1.0	**1.1**	**1.3**	**0.0(5)**	**0.0**
			60	1.2	3.3	0.0(5)	0.0	0.3	**1.2**	**1.5**	**0.0(5)**	**0.0**	**0.3**	**1.1**	**1.5**	**0.0(5)**	**0.0**
			70	1.2	7.6	0.0(5)	0.0	0.5	1.2	4.7	0.0(5)	0.0	0.5	**1.2**	**5.2**	**0.0(5)**	**0.0**
graph120, $\lvert V\rvert = 120, \lvert E\rvert = 7140$	0.5	1	50	**10.6**	**1718.2**	**6503.6(1)**	**16.9**	**3.3**	10.8	1820.2	3.6(0)	0.3	3.3	10.9	1811.3	4.2(0)	0.8
			60	12.2	1814.7	9004.2(0)	24.8	3.3	12.1	1812.6	4.0(0)	0.6	3.3	12.2	1812.5	7.5(0)	3.3
			70	13.3	1815.4	10993.9(0)	28.8	3.4	13.4	1814.1	3.6(0)	0.1	3.4	13.1	1814.0	4.7(0)	1.2
		6	50	2.1	462.0	0.7(4)	0.0	4.5	**2.1**	**396.3**	**0.5(4)**	**0.0**	**4.7**	2.1	405.6	1.1(4)	0.1
			60	**2.2**	**1353.5**	**0.1(3)**	**0.0**	**3.5**	2.2	1247.1	0.8(2)	0.1	3.6	2.2	1309.7	0.8(2)	0.1
			70	**3.9**	**1248.9**	**0.9(2)**	**0.0**	**3.7**	3.9	1592.4	5.(1)	1.7	3.8	3.9	1744.7	4.9(1)	1.7
		12	50	2.0	4.2	0.0(5)	0.0	1.7	**2.0**	**2.3**	**0.0(5)**	**0.0**	**2.2**	**2.0**	**2.3**	**0.0(5)**	**0.0**
			60	2.0	76.1	0.0(5)	0.0	5.1	2.0	6.1	0.0(5)	0.0	5.7	**2.0**	**6.0**	**0.0(5)**	**0.0**
			70	2.0	355.5	0.0(5)	0.0	3.4	2.1	215.1	0.0(5)	0.0	3.8	**2.0**	**65.4**	**0.0(5)**	**0.0**
	1	1	50	10.7	1798.6	2019.5(1)	18.4	0.5	11.0	1796.0	1.0(1)	0.4	0.5	**10.9**	**1768.8**	**0.8(1)**	**0.3**
			60	12.3	1815.1	9023.2(0)	30.3	1.7	12.3	1719.2	2.2(1)	0.5	1.7	**12.3**	**1728.1**	**2.2(2)**	**0.4**
			70	13.2	1815.4	11005.2(0)	30.0	1.8	13.7	1803.5	1.9(1)	0.1	1.8	**13.7**	**1734.1**	**2.3(2)**	**0.6**
		6	50	2.1	456.6	0.5(4)	0.0	3.9	2.1	529.4	0.5(4)	0.0	4.1	**2.1**	**412.0**	**0.8(4)**	**0.2**
			60	**2.2**	**1076.9**	**0.1(3)**	**0.0**	**3.0**	2.2	1274.0	0.9(2)	0.1	3.1	2.2	1323.6	0.8(2)	0.1
			70	**4.1**	**1480.6**	**1.4(2)**	**0.0**	**3.8**	4.1	1806.9	5.3(0)	1.9	3.9	4.1	1655.6	5.5(1)	1.9
		12	50	2.0	4.2	0.0(5)	0.0	1.7	2.0	2.3	0.0(5)	0.0	2.2	**1.9**	**2.2**	**0.0(5)**	**0.0**
			60	2.0	263.1	0.0(5)	0.0	5.3	2.0	7.0	0.0(5)	0.0	5.6	**2.0**	**6.2**	**0.0(5)**	**0.0**
			70	2.1	780.9	0.2(4)	0.0	3.2	2.0	380.8	0.2(4)	0.0	3.6	**2.0**	**335.9**	**0.0(5)**	**0.0**
	5	1	50	10.8	1530.5	2019.3(2)	18.5	0.0	**10.9**	**367.1**	**0.0(5)**	**0.0**	**0.0**	11.1	506.9	0.0(5)	0.0
			60	12.4	1816.0	7167.1(0)	31.3	0.0	12.5	971.9	5.8(4)	4.5	0.0	**12.2**	**776.0**	**0.0(5)**	**0.0**
			70	13.2	1817.0	6993.(0)	31.4	0.0	**13.6**	**985.4**	**0.0(5)**	**0.0**	**0.0**	13.6	1057.8	0.0(5)	0.0
		6	50	2.1	30.9	0.0(5)	0.0	0.2	**2.1**	**5.3**	**0.0(5)**	**0.0**	**0.2**	2.1	7.7	0.0(5)	0.0
			60	2.3	102.7	0.0(5)	0.0	0.4	**2.2**	**44.7**	**0.0(5)**	**0.0**	**0.4**	2.2	48.6	0.0(5)	0.0
			70	**4.0**	**281.8**	**0.0(5)**	**0.0**	**0.4**	4.1	1484.1	0.4(1)	0.1	0.4	4.1	1451.1	0.2(1)	0.0
		12	50	2.0	3.6	0.0(5)	0.0	0.6	**2.0**	**2.2**	**0.0(5)**	**0.0**	**0.7**	**2.0**	**2.2**	**0.0(5)**	**0.0**
			60	2.0	24.3	0.0(5)	0.0	1.5	2.0	4.4	0.0(5)	0.0	1.6	**2.0**	**4.3**	**0.0(5)**	**0.0**
			70	2.0	52.5	0.0(5)	0.0	0.7	2.0	10.3	0.0(5)	0.0	0.7	**2.0**	**10.0**	**0.0(5)**	**0.0**

Table 4.4. *Performance of formulations (Flow-Cov), (Path-Cov), and (Path-Cov) + VI on graph100 and graph120.*

Analyzing the results shown in Table 4.4, we can conclude that the difficulty of solving the instances is highly dependent on the parameters (B, R and p). It can be observed that the instances become more difficult as B and p decrease and R increases. As shown in Table 4.4 and Figure 4.4, the performance of these formulations is quite similar. An interesting observation is that they complement each other. In other words, there are instances in which formulation (Flow-Cov) did not find the optimal solution within the time limit but (Path-Cov) + VI did and vice versa. Nevertheless, (Path-Cov) + VI found the optimal solution in more instances than (Path-Cov) and all the instances solved to optimality by (Path-Cov) were also solved to optimality by (Path-Cov) + VI. It can be appreciated that the average preprocessing time is lower than fourteen seconds in all cases. Furthermore, the linear relaxation of the formulations ($G^t_{LP}\%$) is quite good, being on average 2.1% for (Flow-Cov) and 2.3% for (Path-Cov) and (Path-Cov) + VI.

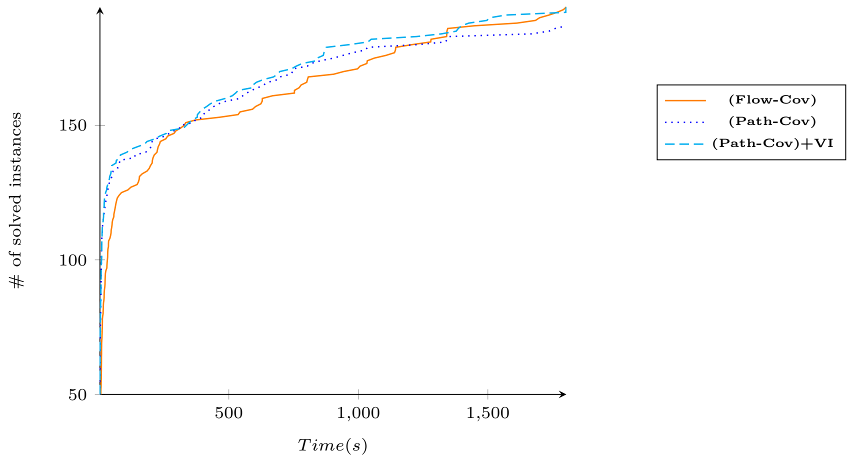


Figure 4.4. *Performance profile graph of #solved instances using (Flow-Cov), (Path-Cov), and (Path-Cov) + VI formulations on graph100 and graph120.*

In view of the results reported in this subsection, we conclude that the best formulations for solving Up-MCLP on complete graphs are (Flow-Cov) and (Path-Cov) + VI. In the next subsection, sparser graphs will be analyzed.

4.6.4 Results on sparse graphs

The aim of this subsection is to compare the proposed formulations on sparse graphs. In particular, we used the uncapacited p-median datasets from the OR-Library.

Table 4.5 has the same structure as Table 4.3. In this table, we reported the results of formulations (Flow-Cov), (Path), (Path) + VI, (Path-Cov), and (Path-Cov) + VI on the smallest pmed datasets (pmed1-pmed5), named pmeds. These networks contain 100 nodes and 195.2 edges on average (the smallest have 190 and the largest 198). The provided results are the average over the five datasets where the other parameters were randomly generated following the procedure described in Subsection 4.6.1.

Similarly to complete graphs, it can be seen in Table 4.5 that the difficulty of the instances is highly parameter-dependent. In addition, it is shown that the preprocessing time is small (less than two seconds in all instances). Furthermore, the number of instances solved to optimality by (Flow-Cov), (Path-Cov), and (Path-Cov) + VI is considerably higher than the ones solved by (Path) and (Path) + VI. Observe that the average of the linear relaxation gaps of (Flow-Cov) (3.1%), (Path-Cov) and (Path-Cov) + VI (4.0%) are better than the linear relaxation gap of (Path) and (Path) + VI (47.5%).

				(Flow-Cov)					(Path)				(Path) + VI			(Path-Cov)					(Path-Cov) + VI			
Data	$B\%$	p	$R\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{total}	$G\%$	$G^t_{BS}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$
pmeds, $\|V\| = 100, \|E\| = 195.2$	0.5	1	50	1.4	26.1	0.0(5)	0.0	4.3	918.8	17.5(4)	2.3	87.5	474.8	23.1(4)	4.5	**1.4**	**20.5**	**0.0(5)**	**0.0**	**4.3**	1.4	24.2	0.0(5)	0.0
			60	**1.5**	**15.5**	**0.0(5)**	**0.0**	**2.1**	1764.7	39.5(1)	8.7	55.3	1168.2	31.9(3)	7.8	1.5	38.7	0.0(5)	0.0	2.1	1.5	25.7	0.0(5)	0.0
			70	**1.5**	**32.6**	**0.0(5)**	**0.0**	**4.0**	1801.9	44.0(0)	8.9	35.8	1580.2	75.0(1)	16.9	1.5	991.7	1.5(3)	0.0	4.1	1.5	572.9	0.3(4)	0.0
		5	50	**1.2**	**1.4**	**0.0(5)**	**0.0**	**1.1**	14.5	0.0(5)	0.0	75.8	13.5	0.0(5)	0.0	**1.2**	**1.4**	**0.0(5)**	**0.0**	**2.7**	**1.2**	**1.4**	**0.0(5)**	**0.0**
			60	**1.2**	**1.8**	**0.0(5)**	**0.0**	**1.3**	575.0	0.0(5)	0.0	57.9	589.1	0.0(5)	0.0	1.2	2.1	0.0(5)	0.0	3.4	1.2	2.3	0.0(5)	0.0
			70	1.2	3.9	0.0(5)	0.0	1.7	803.4	3.2(3)	0.0	38.8	979.5	2.5(3)	0.0	1.2	3.9	0.0(5)	0.0	2.6	**1.2**	**3.7**	**0.0(5)**	**0.0**
		10	50	1.2	1.2	0.0(5)	0.0	0.1	0.8	0.0(5)	0.0	42.8	**0.7**	**0.0(5)**	**0.0**	1.1	1.2	0.0(5)	0.0	3.7	1.1	1.2	0.0(5)	0.0
			60	**1.2**	**1.2**	**0.0(5)**	**0.0**	**0.7**	1.6	0.0(5)	0.0	46.9	1.8	0.0(5)	0.0	1.1	1.3	0.0(5)	0.0	2.9	1.1	1.3	0.0(5)	0.0
			70	**1.2**	**1.4**	**0.0(5)**	**0.0**	**1.6**	9.2	0.0(5)	0.0	35.4	28.2	0.0(5)	0.0	1.2	1.5	0.0(5)	0.0	3.0	1.2	1.6	0.0(5)	0.0
	1	1	50	1.7	570.6	0.0(5)	0.0	7.2	1496.5	31.6(1)	2.7	77.1	1137.1	83.1(3)	13.1	1.7	202.6	0.0(5)	0.0	7.3	**1.6**	**68.7**	**0.0(5)**	**0.0**
			60	1.7	1460.1	2.4(2)	0.0	7.7	1800.6	40.7(0)	4.9	49.6	1488.5	38.2(1)	6.7	1.7	960.6	3.1(4)	1.3	7.8	**1.8**	**850.1**	**3.1(4)**	**1.3**
			70	**1.8**	**425.1**	**0.0(5)**	**0.0**	**4.8**	1801.2	29.3(0)	5.7	30.1	1800.7	349.5(0)	51.1	1.8	1425.3	2.7(2)	0.0	4.9	1.8	1513.8	3.8(1)	0.0
		5	50	1.2	1.8	0.0(5)	0.0	2.8	84.1	0.0(5)	0.0	75.7	77.7	0.0(5)	0.0	1.2	1.8	0.0(5)	0.0	4.9	**1.2**	**1.7**	**0.0(5)**	**0.0**
			60	**1.2**	**3.4**	**0.0(5)**	**0.0**	**5.0**	918.9	0.7(4)	0.1	55.8	1116.2	3.9(2)	0.0	1.2	22.7	0.0(5)	0.0	6.6	1.3	18.2	0.0(5)	0.0
			70	1.3	62.1	0.0(5)	0.0	5.2	1718.3	8.3(1)	0.2	36.3	1602.7	9.6(1)	0.3	1.3	76.6	0.0(5)	0.0	5.3	**1.3**	**39.9**	**0.0(5)**	**0.0**
		10	50	1.1	1.2	0.0(5)	0.0	1.5	0.8	0.0(5)	0.0	43.1	**0.7**	**0.0(5)**	**0.0**	1.1	1.2	0.0(5)	0.0	4.4	1.1	1.2	0.0(5)	0.0
			60	**1.2**	**1.3**	**0.0(5)**	**0.0**	**1.7**	2.6	0.0(5)	0.0	46.3	2.4	0.0(5)	0.0	1.2	1.9	0.0(5)	0.0	4.2	1.2	1.7	0.0(5)	0.0
			70	**1.2**	**1.7**	**0.0(5)**	**0.0**	**3.2**	18.0	0.0(5)	0.0	34.6	25.3	0.0(5)	0.0	1.2	2.6	0.0(5)	0.0	5.6	1.2	2.5	0.0(5)	0.0
	5	1	50	1.5	1015.6	0.0(5)	0.0	3.0	1288.4	24.(2)	2.7	55.1	1155.6	25.7(2)	3.2	**1.5**	**64.8**	**0.0(5)**	**0.0**	**3.0**	1.5	85.5	0.0(5)	0.0
			60	1.6	1407.8	3.4(2)	0.5	4.5	1800.6	25.1(0)	4.8	33.7	1690.5	27.4(1)	6.2	**1.6**	**869.4**	**2.5(3)**	**0.1**	**4.6**	1.6	1140.9	2.5(2)	0.1
			70	1.6	1205.0	1.8(2)	0.0	2.7	1801.7	22.6(0)	5.7	19.3	1758.5	268.6(1)	35.2	1.6	1077.4	1.8(3)	0.0	2.8	**1.7**	**946.9**	**1.9(3)**	**0.0**
		5	50	**1.2**	**2.3**	**0.0(5)**	**0.0**	**4.1**	102.8	0.0(5)	0.0	66.8	132.1	0.0(5)	0.0	1.2	3.0	0.0(5)	0.0	4.2	1.2	2.9	0.0(5)	0.0
			60	**1.2**	**5.2**	**0.0(5)**	**0.0**	**2.7**	686.2	0.3(4)	0.0	46.2	738.7	0.0(5)	0.0	1.2	21.0	0.0(5)	0.0	2.7	1.2	17.6	0.0(5)	0.0
			70	1.3	207.3	0.0(5)	0.0	3.0	1418.4	2.8(3)	0.1	28.8	1458.9	4.2(2)	0.0	1.3	48.5	0.0(5)	0.0	3.0	**1.3**	**48.1**	**0.0(5)**	**0.0**
		10	50	1.2	1.2	0.0(5)	0.0	0.8	1.0	0.0(5)	0.0	38.4	**0.8**	**0.0(5)**	**0.0**	1.1	1.3	0.0(5)	0.0	1.1	1.2	1.3	0.0(5)	0.0
			60	**1.2**	**1.5**	**0.0(5)**	**0.0**	**3.1**	4.3	0.0(5)	0.0	40.7	4.6	0.0(5)	0.0	1.2	1.9	0.0(5)	0.0	4.0	1.2	1.8	0.0(5)	0.0
			70	**1.2**	**2.2**	**0.0(5)**	**0.0**	**2.9**	61.8	0.0(5)	0.0	29.4	38.3	0.0(5)	0.0	1.2	4.1	0.0(5)	0.0	3.0	1.2	3.9	0.0(5)	0.0

Table 4.5. *Performance of formulations (Flow-Cov), (Path), (Path) + VI, (Path-Cov), and (Path-Cov) + VI on pmeds.*

In Table 4.6, we report the result obtained on datasets of larger size, named pmedb. More concretely, the table shows the results of formulations (Flow-Cov), (Path-Cov), and (Path-Cov) + VI on larger pmed datasets (pmed6-pmed10). These networks contain 200 nodes and 777.8 edges on average (the smallest have 774 and the largest 785). Note that the results of (Path), (Path) + VI are not reported because very few instances were solved to optimality. In Figure 4.5, the performance profile of the number of solved instances using these formulations is depicted.

				(Flow-Cov)					(Path-Cov)					(Path-Cov) + VI			
Data	$B\%$	p	$R\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$	$G^t_{LP}\%$	t_{st}	t_{total}	$G\%$	$G^t_{BS}\%$
pmedb, $\lvert V\rvert = 200, \lvert E\rvert = 777.8$	0.5	1	50	12.5	1815.3	5881.2(0)	18.7	9.1	12.4	1812.9	9.3(0)	0.1	9.9	12.3	1812.7	8.9(0)	0.0
			60	18.3	1821.2	6383.5(0)	33.1	7.4	18.5	1819.1	7.5(0)	0.0	8.0	18.4	1818.7	7.7(0)	0.1
			70	36.3	1839.9	13701.3(0)	24.8	5.8	37.6	1848.8	5.8(0)	0.0	6.4	37.1	1837.5	6.1(0)	0.3
		10	50	9.3	22.4	0.0(5)	0.0	2.1	**9.2**	**14.8**	**0.0(5)**	**0.0**	**4.3**	9.1	15.3	0.0(5)	0.0
			60	9.5	1451.5	1.2(2)	0.0	4.4	9.3	1150.8	1.1(2)	0.2	6.5	**9.3**	**975.3**	**1.3(3)**	**0.3**
			70	10.0	1811.3	2.8(0)	0.6	3.9	9.8	1815.2	2.7(0)	1.0	4.4	9.8	1811.3	2.6(0)	1.0
		20	50	**8.9**	**9.1**	**0.0(5)**	**0.0**	**1.8**	8.9	9.2	0.0(5)	0.0	3.6	**8.8**	**9.1**	**0.0(5)**	**0.0**
			60	9.0	45.4	0.0(5)	0.0	2.5	**9.0**	**13.3**	**0.0(5)**	**0.0**	**3.8**	8.9	14.4	0.0(5)	0.0
			70	9.4	785.8	0.1(4)	0.1	2.7	**9.2**	**617.7**	**0.4(4)**	**0.0**	**5.2**	9.2	771.0	0.3(3)	0.0
	1	1	50	11.2	1814.1	6101.7(0)	23.3	8.2	11.2	1811.5	8.4(0)	0.0	8.4	11.1	1811.5	8.4(0)	0.0
			60	16.2	1819.9	13905.8(0)	25.2	7.4	16.3	1819.4	7.5(0)	0.0	7.6	16.1	1816.8	7.5(0)	0.0
			70	33.7	1836.7	17155.8(0)	26.3	5.1	33.7	1834.1	5.4(0)	0.2	5.3	33.5	1834.1	5.5(0)	0.4
		10	50	9.3	299.3	0.0(5)	0.0	3.7	9.0	57.5	0.0(5)	0.0	4.7	**9.0**	**53.7**	**0.0(5)**	**0.0**
			60	9.4	1810.0	3.2(0)	0.1	5.5	9.5	1566.1	2.2(1)	0.5	5.8	**9.3**	**1526.1**	**1.7(1)**	**0.2**
			70	9.5	1811.1	3.9(0)	0.7	3.8	9.6	1811.5	2.7(0)	0.4	3.8	9.5	1810.6	4.1(0)	1.9
		20	50	9.2	9.5	0.0(5)	0.0	3.2	9.1	9.4	0.0(5)	0.0	4.2	**8.9**	**9.3**	**0.0(5)**	**0.0**
			60	9.0	141.0	0.0(5)	0.0	3.6	9.3	23.2	0.0(5)	0.0	4.5	**9.0**	**20.7**	**0.0(5)**	**0.0**
			70	9.3	1613.6	1.0(1)	0.1	3.2	9.2	1158.1	0.5(3)	0.0	4.1	**9.1**	**959.8**	**0.3(4)**	**0.0**
	5	1	50	11.2	1546.2	6102.5(2)	25.3	0.0	**11.2**	**181.1**	**0.0(5)**	**0.0**	**0.0**	11.0	185.9	0.0(5)	0.0
			60	16.6	1742.8	6551.2(1)	24.6	0.1	16.0	917.2	0.1(3)	0.0	0.1	**16.0**	**803.1**	**0.1(4)**	**0.0**
			70	33.2	1837.5	11417.1(0)	29.4	0.2	32.6	853.6	0.2(4)	0.0	0.2	**32.6**	**769.9**	**0.2(4)**	**0.0**
		10	50	9.1	342.8	0.0(5)	0.0	1.7	9.2	34.8	0.0(5)	0.0	1.7	**9.0**	**30.0**	**0.0(5)**	**0.0**
			60	9.6	1527.2	1.1(1)	0.2	1.7	9.2	696.4	0.0(5)	0.0	1.7	**9.2**	**434.5**	**0.0(5)**	**0.0**
			70	9.5	1811.4	1.2(0)	0.8	0.5	9.5	733.6	0.2(4)	0.0	0.5	**9.4**	**687.4**	**0.5(4)**	**0.3**
		20	50	**9.0**	**9.4**	**0.0(5)**	**0.0**	**1.2**	8.9	9.6	0.0(5)	0.0	1.3	8.8	9.5	0.0(5)	0.0
			60	9.2	44.5	0.0(5)	0.0	1.3	9.0	17.1	0.0(5)	0.0	1.4	**8.9**	**15.5**	**0.0(5)**	**0.0**
			70	9.3	1324.6	0.4(2)	0.0	0.8	9.1	730.6	0.0(4)	0.0	0.8	**9.3**	**493.2**	**0.1(4)**	**0.0**

Table 4.6. *Performance of formulations (Flow-Cov), (Path), (Path) + VI, (Path-Cov), and (Path-Cov) + VI on pmedb.*

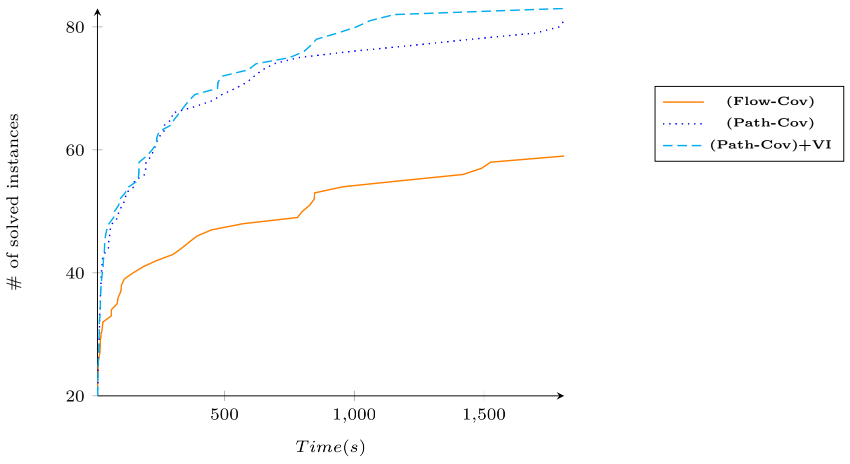


Figure 4.5. *Performance profile graph of #solved instances using (Flow-Cov), (Path-Cov), and (Path-Cov) + VI formulations on pmedb dataset.*

As can be appreciated in Table 4.6 and Figure 4.5, (Path-Cov) + VI outperforms (Flow-Cov) and (Path-Cov). Observe that although the gap of the linear relaxation of (Flow-Cov) (3.4%) is smaller than the gap of (Path-Cov) and (Path-Cov) + VI (4%), the latter is the one that solves to optimality within the time limit the largest number of instances.

Based on the results presented in this subsection, we conclude that the best formulation for solving Up-MCLP on pmedian graphs was (Path-Cov) + VI. Furthermore, the results included in this subsection show the usefulness of including the valid inequalities discussed in the chapter.

4.7 Concluding remarks

In this chapter, we have tackled an interesting problem: the upgrading maximal covering location problem with edge length modifications, Up-MCLP. As far as we know, it is the first time that this problem is discussed in the literature.

Since we were dealing with a new problem, we proposed three different mixed-integer formulations to model the situation from various perspectives. Moreover, we developed an effective preprocessing phase, which fixed many variables and reduced the size of the problem considerably. Then, for each formulation, we provided several sets of valid inequalities. These constraints allowed us to strengthen the formulations and to reduce the symmetries contained in the problem, shortening the time to solve the formulations. The performance of the three formulations and the improvement provided by the preprocessing phase and the valid inequalities can be appreciated in the computational results included in the chapter.

We believe that this work is an encouraging starting point that opens up many opportunities for further research. For example, interesting and similar problems could be obtained by modifying the covering criterion, the location criterion, and the upgrading assumptions. Observe that the results included in this chapter are presented in Baldomero-Naranjo et al. (2021a). In addition, we are working on theoretical results that show the complexity of several particular cases of this problem in Baldomero-Naranjo et al. (2021c).

In the next chapter we also deal with the MCLP. However, the initial hypotheses are very different. We assume that the facility can be located anywhere along the network and that the demand is continuous (distributed along the edges) and uncertain with only a known interval estimation. To face this situation of total uncertainty, we propose a model to find the location that minimizes the maximal regret.

5 Minmax Regret Maximal Covering Location Problems with Edge Demands

This chapter addresses a version of the single-facility Maximal Covering Location Problem on a network where the demand is: *i)* distributed along the edges and *ii)* uncertain with only a known interval estimation. To deal with this problem, we propose a minmax regret model where the service facility can be located anywhere along the network. This problem is called Minmax Regret Maximal Covering Location Problem with demand distributed along the edges (MMR-EMCLP). Furthermore, we present two polynomial algorithms for finding the location that minimizes the maximal regret assuming that the demand realization is an unknown constant or linear function on each edge. We also include two illustrative examples as well as a computational study for the unknown constant demand case to illustrate the potential and limits of the proposed methodology.

5.1 Introduction

In this chapter, we continue our research on the maximal covering location problem (MCLP). In particular, we focus on a version of the single-facility MCLP on a network where the demand is distributed along the edges and uncertain with only a known interval estimation. As see in the previous chapter, we can find many different variants of this problem in the literature, some of them in continuous space and others on networks or in discrete spaces (see Plastria (2002); García and Marín (2019); Berman and Krass (2002) for a detailed survey of each case). Most of the classical models discussed in the literature assume that the demand is represented by a finite set of points. However, recently, an increasing body of literature in facility location addresses problems where the demand is continuously distributed over a region (polygon, straight lines, edges, etc.), see Murray and Tong (2007); Murray et al. (2010).

In the case of network problems (the subject of study of this chapter), there are several applications where the assumption that the demand only occurs at the nodes of the network is not realistic; e.g. for the location of emergency facilities (see Li et al. (2011)), for the planning of garbage collection, meter reading, or mail delivery, or situations where the coverage areas are extremely distance-dependent such as the location of Automated External Defibrillators (AED), bus stops, Automated Teller Machines (ATM), etc. Similarly, assuming that the service facilities can only be located at the nodes of the network, the so-called node-restricted version, also leads to unrealistic models for some situations, see ReVelle et al. (1976); Church and Meadows (1979). For example, in the case of locating bus stops, this assumption would mean that they can only be placed at the corner of streets. Thus, assuming that the demand is concentrated at nodes and/or service facilities can only be located at nodes may yield models that are far from the real situation under study, providing solutions that are of little use, see Current and Schilling (1990) for further details.

In the following, we review the main models in the literature related to our problem. The MCLP where the demand is assumed to exist only at nodes is solved in Church and ReVelle (1974). They proposed an integer programming formulation to solve the multi-facility node-restricted problem. García and Marín (2019) and Marín et al. (2018) provided an overview of models for the discrete covering location problem. Recently, Cordeau et al. (2019) proposed an effective decomposition approach for the MCLP based on a branch-and-Benders-cut formulation to solve realistic cases where the number of customers is much larger than the number of potential facility locations. To solve the continuous version of the problem, i.e., facilities can be located not only at the nodes but also along the edges, Church and Meadows (1979) proposed a Finite Dominating Set (FDS). An FDS for a location problem is a finite set of points that is guaranteed to contain an optimal solution for the problem. If such a set can be found, then the continuous optimization problem can equivalently be formulated as a discrete optimization problem and solved applying techniques of discrete optimization. This technique is widely applied in the literature to solve these kinds of problems, see for example Nickel and Puerto (1999); Kalcsics et al. (2002, 2003); Berman et al. (2009).

The MCLP where the demand is distributed along the edges (edge demand) and the facilities can be located anywhere on the graph (continuous location space) was first proposed by Berman et al. (2016). They found a FDS for the case of known demand functions and presented an exact approach to solve the single facility problem for constant and piecewise linear demand functions (unlike this chapter, the actual demand functions are assumed to be known a priori). Extensions of this model are also analyzed, as for instance the obnoxious version, the multi-facility case, and the conditional problem (when all but one facility have already been located). Moreover, they proposed a heuristic for the multi-facility case. Under the same framework (edge demand and continuous location space), a minimum set-covering problem (locating the minimum number of facilities to cover all the demand points) is studied by Fröhlich et al. (2020). Furthermore, a stochastic version of the MCLP is analyzed by Blanquero et al. (2016). In this paper, the expected demand covered is maximized and the problem is formulated as a Mixed Integer Nonlinear Program (MINLP) that is solved using a branch-and-bound algorithm comprising a combinatorial part (in which edges of the network are chosen to contain facilities) and a continuous global optimization part (once the edges are chosen, the optimal locations are found on those edges).

In this chapter, we consider the MCLP under this framework on a network, i.e., edge demand and continuous location space. However, unlike the aforementioned papers, the actual intensity of the demand along each edge is unknown. Instead, the only available knowledge is an upper and lower bound of this intensity at each point of the network. We are not aware of any other paper dealing simultaneously with edge demand, a continuous location space, and uncertainty in the intensity of the demand for the MCLP on networks (or, in fact, for any location problem on networks). From a practical point of view, assuming that demand along an edge is known corresponds to an ideal but usually unrealistic scenario. Since demand is uncertain in nature and varies from one day to another, or even within a day, we will treat demand as being unknown. Nevertheless, we usually have a good estimation of the minimal or maximal demand along the edge, so that we can at least assume demand to lie within a known range.

Concerning real-life applications, the MCLP has been applied in several fields such as the location of emergency facilities, health care, natural disaster rescue, ecology, signal-transmission facilities, bike-sharing, police resources, see Adenso-Díaz and Rodríguez (1997); Bonn et al. (2002); Chung (1986); Daskin and Dean (2004); Muren et al. (2020); Wright et al. (2006); Yang et al. (2020). More concretely, our model can be used to locate AED, ATM, bus stops, automated parcel lockers, or bicycle parking racks, among others. In these applications the network represents a city, where the edges are the streets and the nodes are the intersections between them. Moreover, these services can be located anywhere along the network because they do not require sophisticated infrastructures or spaces. We consider that the demand is the potential amount of users for that service. Also, a client would be considered as a covered client if the distance between him/her and the service is lower than or equal to a fixed radius. In these applications, the exact demand of these services is usually unknown but we can estimate the minimal and maximal values of the demand taking into account the distribution of clients along the streets. The uncertainty in demand can stem from the uncertainty about the need for or the attractiveness of the facility's service, but can also be due to a variation of the actually present population in the street over the course of the day. For example, in a residential area, the number of inhabitants during the working hours would be smaller than in the evenings.

In the face of this situation of total uncertainty in the demand, we propose to use concepts from robust optimization to identify suitable facility locations. More concretely, we aim at minimizing the worst-case coverage loss, i.e., minimize the maximal regret. This criterion was first introduced in Savage (1951). A review of minmax regret optimization as well as a discussion on its potential applications is contained in the book by Kouvelis and Yu (1997b). This criterion has been successfully applied to many location problems on networks, see Assunção et al. (2017); Averbakh (2001); Averbakh and Berman (2000); Averbakh and Lebedev (2004); Conde (2019); Conde et al. (2018); Kouvelis et al. (1993); Puerto et al. (2009), among others. In case of the MCLP, this criterion is used by Coco et al. (2018) for a discrete problem with a finite set of demand points and a finite set of potential locations for facilities. Berman and Wang (2011) consider the minmax regret objective for the gradual covering location problem on networks, also assuming that demand only occurs at nodes, but that facilities can be located anywhere along the network. The gradual covering location problem is a generalized version of MCLP where the coverage area is divided in two sectors, one where the demand of nodes within it is completely covered and the other one where the demand is partially covered. An alternative way of considering uncertainty for the discrete MCLP is to use a fuzzy framework, as shown in Arana-Jiménez et al. (2020).

Our goal in this chapter is to solve the minmax regret single-facility MCLP on a network with three main features: edge demand, uncertainty in the intensity of the demand, and a continuous location space. We analyze two cases: i) the realization of the demand intensity is unknown, but constant on an edge and bounded (from above and below) by two known constant functions; and ii) the demand intensity is given by an unknown linear function on each edge and it is bounded by two known linear functions (see López-de-los-Mozos et al. (2013); Averbakh et al. (2018); Kouvelis and Yu (1997a) for a different way of using uncertain demand intensities at nodes given by linear functions for minmax regret location problems).

In the aforementioned examples of locating AED, ATM, bus stops, automated parcel lockers, or bicycle parking racks, the uncertainty in the demand comes from the fact that the number of inhabitants (potential clients) during the working day is not known. Looking at cities, the type of buildings that one can find along a street are often similar. For example, in the suburbs it is often all single-family homes or low-storey apartment buildings along a street; in more central areas, high-storey apartment buildings or even high rises. If the type of buildings is fairly uniform, we can assume that the lower and upper bounds on the demand are also fairly uniform, i.e., each of them can be modeled by a constant function.

For such situations, our aim is to find a location such that the worst-case regret for the location is minimal. This assumption makes sense for AED locations, because the health service does not know who is going to need an AED or where, but they are interested in providing this service for all the inhabitants of the city as fast as possible. Similarly, for the location of ATMs, the banks are interested in providing their services to the maximal number of people and avoid losing potential clients.

Regarding linear demand along the edge, this could be the case in the situation of locating bus stops, automated parcel lockers, or bicycle parking racks, where the demand along the streets can vary since either the presence of potential users (tourists, workers, students, etc.) increases when approaching the city center, the business area, or the university campus, or the type of buildings along a street changes gradually, e.g., from single-family homes on one end to row houses on the other end. A suitable way to model such increases could be through a linear function, since, in most cases, these increases are gradual when approaching to those areas. Accordingly, also the lower and upper bounds on the demand intensity could be linear. Maybe not in a strictly linear fashion, e.g., in the case of a change of type of building, but as long as the change is not too dramatic, this is a fair assumption. In case of a dramatic change, e.g., from single-family homes to high rises, we would split the edge at the point where the change occurs and treat the two sub-edges separately, reflecting the change also in the lower and upper bound.

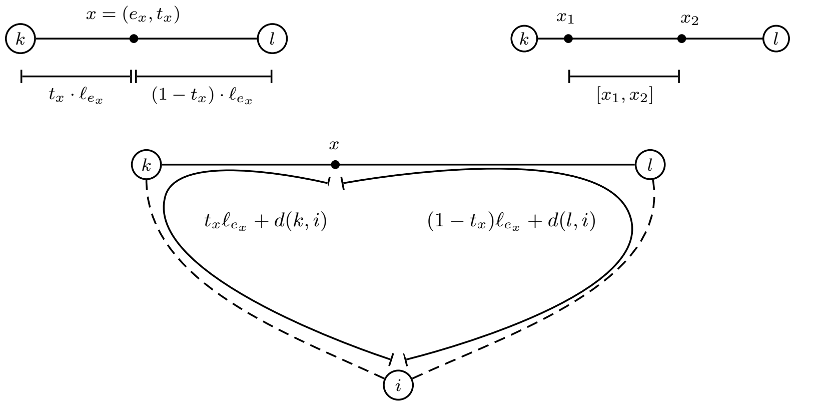
Alternative applications can be found if the network represents air ducts of a building (conduits used in heating, ventilation, and air conditioning). For example, one application is to locate an aerosol dispenser in order to disinfect the conduits, see Frazier (1990). This fits to a MCLP, since aerosols are only effective within a certain distance from their source. Moreover, the demand represents the presence of bacteria, viruses, and fungi along the conduits that should be killed. The actual intensity of demand along the conduits is unknown, but can be estimated by room occupancy rates, uses and types of buildings (healthcare facilities, offices, restaurants, care homes, etc.), and proximity to air inlets and outlets, among others. Since the presence of germs increases when approaching to air inlets and outlets, the demand can be approximated by linear functions.

The remainder of the chapter is organized as follows. In Section 5.2, we describe the problem. In Section 5.3, we present a polynomial time algorithm to solve the unknown constant demand case and we include a computational study to illustrate the potential and limits of the proposed methodology. In Section 5.4, we develop a polynomial time algorithm to solve the unknown linear demand case. In both cases, an example illustrating the proposed methodology is shown. In Section 5.5, we present our conclusions and an outlook to future research. Finally, there are two appendices. In the first one, we include some known results about the lower envelope of Jordan arcs that will be used to analyze the complexity of the algorithms proposed in the chapter. In the second one, we derive the representation of some functions used in the examples.

5.2 Definitions and Problem Description

Let $G = (V, E)$ be an undirected graph with node set $V = \{1, \dots, n\}$ and edge set E, where $|E| = m$. Every edge $e = [k, l] \in E$, $k, l \in V$, has a positive length ℓ_e and is assumed to be rectifiable. G will also denote the continuum set of all points of the network. In addition, in a slight abuse of notation, we use e_x to represent an edge that contains point x. For an edge $e_x = [k, l] \in E$, a point $x \in e_x$ is also represented as $x = (e_x, t_x)$, $0 \leq t_x \leq 1$, where t_x is the relative distance of x from k with respect to ℓ_{e_x}, i.e., the distance from x to node k is $t_x \ell_{e_x}$ and to node l is $(1-t_x)\ell_{e_x}$. Then, for a node $i \in V$, the distance from x to i is defined as $d(x, i) = \min\{t_x \cdot \ell_{e_x} + d(k, i), \, (1 - t_x) \cdot \ell_{e_x} + d(l, i)\}$, where $d(k, i)$ and $d(l, i)$ are the lengths of the shortest paths connecting k with i and l with i, respectively. The distance $d(x, y)$ between two points $x, y \in G$ is defined analogously. Let $(e, [t_1, t_2]) = [x_1, x_2] = \{x \in e = [k, l] : t_1\ell_e \leq d(x, k) \leq t_2\ell_e, (1-t_2)\ell_e \leq d(x, l) \leq (1-t_1)\ell_e\}$ be a closed subedge of e, where $x_1 = (e, t_1)$ and $x_2 = (e, t_2)$. Besides, we are given a fixed coverage radius $R > 0$. See Figure 5.1 for an illustration of a point x on e_x (top left-hand side), a subedge $[x_1, x_2]$ (top right-hand side), and of $d(x, i)$ (at the bottom).

Finally, for each edge $e = [k, l] \in E$ we are given two non-negative continuous functions $lb_e : [0, 1] \to \mathbb{R}_0^+$ and $ub_e : [0, 1] \to \mathbb{R}_0^+$ that specify the minimal and maximal demand along the edge. A specific *demand realization* or *scenario* on e is denoted by the continuous function $w_e : [0, 1] \to \mathbb{R}_0^+$



$x = (e_x, t_x)$
k
l
$t_x \cdot \ell_{e_x}$
$(1 - t_x) \cdot \ell_{e_x}$
x_1
x_2
k
l
$[x_1, x_2]$
x
k
l
$t_x\ell_{e_x} + d(k, i)$
$(1 - t_x)\ell_{e_x} + d(l, i)$
i

Figure 5.1. *Illustration for a point on an edge, a subedge, and $d(x, i)$.*

where $lb_e(t) \leq w_e(t) \leq ub_e(t)$, $t \in [0, 1]$. For short, we sometimes write $lb_e \leq w_e \leq ub_e$ for the latter condition. We define by $lb = (lb_e)_{e \in E}$ the vector of lower bound functions on the network; analogously, we define $ub = (ub_e)_{e \in E}$ and $w = (w_e)_{e \in E}$. Abusing notation, we will abbreviate the corresponding condition for the demand realizations by $lb \leq w \leq ub$.

Let $x \in G$ be a point on the network. We say that a point $z \in G$ is *covered* by a facility at x, if $d(x, z) \leq R$. Let $C(x) := \{z \in G \mid d(x, z) \leq R\}$ be the *coverage area* of x, i.e., the whole set of points of G covered by x and let $C_e(x) := \{z \in e \mid d(x, z) \leq R\}$ be the coverage area of x on e, i.e., the set of points of e covered by x. The total amount of covered demand on an edge $e \in E$ by a facility at x for a specific demand realization w is given by

$$g_e(x, w_e) \;=\; \int_{y=(e,t) \in C_e(x)} w_e(t)\, dt, \tag{5.1}$$

and the total amount of covered demand on the entire network by

$$g(x, w) \;=\; \sum_{e \in E} g_e(x, w_e)\,. \tag{5.2}$$

The *worst-case* or *maximal regret* of choosing x over all possible demand realizations is defined as

$$r(x) \;:=\; \max_{lb \leq w \leq ub} \left(\max_{y \in G} g(y, w) \;-\; g(x, w) \right). \tag{5.3}$$

The realization w and the point y maximizing the right-hand side of the expression above are called the *worst-case realization* and the *worst-case alternative*, respectively, for x. The *Minmax Regret Maximal Covering Location Problem* *with Edge Demand (MMR-EMCLP)* can now be formulated as follows:

$$r^* := \min_{x\in G} \max_{lb\leq w\leq ub} \left(\max_{y\in G} g(y,w) - g(x,w) \right). \tag{5.4}$$

Next, we will recall several definitions and results that were given in Berman et al. (2016) for the case without uncertainty. Let $x = (e_x, t_x)$ with $e_x = [k,l] \in E$ and $t_x \in [0,1]$. For an arbitrary edge $e = [i,j] \in E$, $e \neq e_x$, let

$$\begin{aligned} s_e^+(x) &= \min\left\{1, \max\left\{0, \frac{R - d(x,i)}{\ell_e}\right\}\right\} \quad \text{and} \\ s_e^-(x) &= \max\left\{0, \min\left\{1, 1 - \frac{R - d(x,j)}{\ell_e}\right\}\right\}. \end{aligned}$$

If $s_e^+(x) \in (0,1)$, then $(e, s_e^+(x))$ is the point on e for which the shortest path from this point to x via the end node i has a length of exactly R. Moreover, all points on the subedge $(e, [0, s_e^+(x)])$ are covered by x. The same interpretation holds for $s_e^-(x)$ with respect to node j. See the picture on the left-hand side in Figure 5.2. For $e = e_x$ we define

$$s_{e_x}^+(x) = \max\left\{0, \frac{d(x,k) - R}{\ell_{e_x}}\right\} \text{ and } s_{e_x}^-(x) = \min\left\{1, 1 - \frac{d(x,l) - R}{\ell_{e_x}}\right\}.$$

Hence, all demand points on the subedge $(e_x, [s_{e_x}^+(x), s_{e_x}^-(x)])$ of e_x will be covered by a facility at x. See the picture on the right-hand side in Figure 5.2 for an illustration. In the following we call $s_e^+(\cdot)$ and $s_e^-(\cdot)$ the *edge coverage functions.*

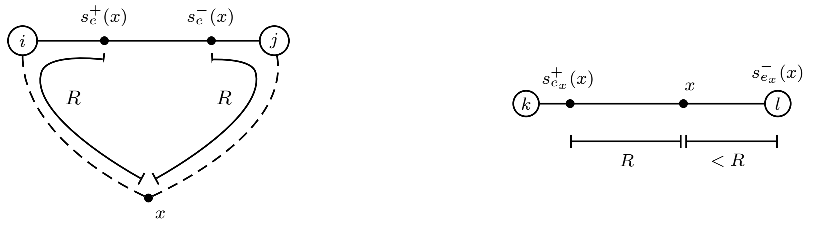


Figure 5.2. *Illustration of the edge coverage points $s_e^+(x)$ and $s_e^-(x)$ (Berman et al. (2016)).*

In the following lemma, we sum up the calculation of the coverage area of x:

Lemma 5.1 (Berman et al. (2016)). *For $e \in E \setminus \{e_x\}$, we have*

$$C_e(x) = \begin{cases} e, & \text{if } s_e^-(x) \leq s_e^+(x), \\ (e, [0, s_e^+(x)] \cup [s_e^-(x), 1]), & \text{if } 0 < s_e^+(x) < s_e^-(x) < 1, \\ (e, [0, s_e^+(x)]), & \text{if } 0 < s_e^+(x) < 1 = s_e^-(x), \\ (e, [s_e^-(x), 1]), & \text{if } s_e^+(x) = 0 < s_e^-(x) < 1, \\ \emptyset, & \text{if } s_e^+(x) = 0,\ s_e^-(x) = 1, \end{cases}$$

and $C_{e_x}(x) = (e_x, [s_{e_x}^+(x), s_{e_x}^-(x)])$.

Using this, we classify all edges $E \setminus \{e_x\}$ as covered, uncovered, and partially covered as follows:

$$\begin{aligned} E^c(x) &= \{e \in E \setminus \{e_x\} \mid s_e^-(x) \leq s_e^+(x)\}, \\ E^u(x) &= \{e \in E \setminus \{e_x\} \mid s_e^+(x) = 0 \text{ and } s_e^-(x) = 1\}, \\ E^p(x) &= E \setminus \{E^c(x) \cup E^u(x) \cup \{e_x\}\}. \end{aligned}$$

Edge e_x will either be fully or partially covered. The total coverage can now be written as

$$\begin{aligned} g(x, w) &= \int_{s_{e_x}^+(x)}^{s_{e_x}^-(x)} w_{e_x}(u)\, du + \sum_{e \in E^c(x)} \int_0^1 w_e(u)\, du \\ &\quad + \sum_{e \in E^p(x)} \left(\int_0^{s_e^+(x)} w_e(u)\, du + \int_{s_e^-(x)}^1 w_e(u)\, du \right). \end{aligned} \tag{5.5}$$

5.2.1 Singularity points

In the current subsection, we analyze the singularity points on a network that will be useful to analyze the behavior of the objective function for the problem under study. A point $x = (e_x, t_x)$, $e_x = [k, l] \in E$ is called a bottleneck point of node i, if $t_x \cdot \ell_{e_x} + d(k, i) = (1 - t_x) \cdot \ell_{e_x} + d(l, i)$ with $0 < t_x < 1$. We denote by BP_{e_x} and BP the set of all bottleneck points on edge e_x and all bottleneck points of the network, respectively.

In the following, we denote by $NP_{e_x} = \{x \in e_x \mid \exists\, i \in V : d(x, i) = R\}$ the set of all network intersect points (NIPs) on an edge e_x and by $NP = \bigcup_{e_x \in E} NP_{e_x}$ the set of all NIPs on the network (Church and Meadows (1979)).

The bottleneck points and network intersect points determine the breakpoints of the edge coverage functions, as stated in the following result.

Lemma 5.2 (Berman et al. (2016)). *The edge coverage functions $s_e^+(x)$ and $s_e^-(x)$, $e \in E$, are continuous and piecewise linear functions over $x \in e_x$ with a constant number of pieces. Breakpoints of these two functions are either* *bottleneck points or network intersect points (associated with some of the two endnodes of e).*

A point $x \in G$ is called *exact coverage point* if $0 < s_e^+(x) = s_e^-(x) < 1$ for some $e \in E$ and there exists $\epsilon_1, \epsilon_2 \geq 0$, $\epsilon_1 + \epsilon_2 > 0$, such that all points $x' \in (x-\epsilon_1, x) \cup (x, x+\epsilon_2)$, no longer completely cover e (Berman et al. (2016)). We denote by EP_{e_x} the set of all exact coverage points on an edge e_x and by $EP = \bigcup_{e_x \in E} EP_{e_x}$ the set of all exact coverage points on the network.

Finally, for $e_x = [k,l] \in E$, we define the set of *partition points*

$$PP_{e_x} := \{k,l\} \cup NP_{e_x} \cup BP_{e_x} \cup EP_{e_x} \qquad \text{and} \qquad PP := \bigcup_{e_x \in E} PP_{e_x} .$$

For two consecutive partition points z^1, $z^2 \in PP_{e_x}$, we call the subedge $[z^1, z^2] \subseteq e_x$ a *partition edge.* Concerning the cardinality of PP, let $e, e_x \in E$, such that $e \neq e_x$. Each node of the network induces at most one bottleneck point and at most two network intersect points on e_x. Similarly, we compute the number of exact coverage points. By Lemma 5.2, $s_e^+(x)$ and $s_e^-(x)$, $e \in E$, are continuous and piecewise linear functions over $x \in e_x$ with a constant number of pieces. Hence, there is a constant number of solutions on e_x of $s_e^+(x) = s_e^-(x)$, i.e., there is a constant number of exact coverage points. Therefore, there are at most $\mathcal{O}(m)$ partition points on each $e_x \in E$ and $\mathcal{O}(m^2)$ on the whole network. With the above definitions, we can formulate the following result:

Proposition 5.1 (Berman et al. (2016)). *Let $e_x \in E$ and $z^1, z^2 \in PP_{e_x}$ be two consecutive partition points on e_x. Moreover, let w be a non-negative continuous demand function.*

1. *The sets $E^c(x)$, $E^u(x)$, and $E^p(x)$ are identical for all $x \in [z^1, z^2]$.*
2. *The edge coverage functions $s_e^+(x)$ and $s_e^-(x)$ have a unique linear representation in x over $[z^1, z^2]$ for each partially covered edge $e \in E^p(x)$.*
3. *All functions $g_e(x, w_e)$, $e \in E$, have a unique representation in x over $[z^1, z^2]$.*

A summary of the notation introduced in this section can be seen in Table 5.1.

5.3 The Unknown Constant Demand Case

In this section, we solve the single facility (MMR-EMCLP) with unknown constant demand realizations $w_e(t)$, i.e., we assume that the demand realization along the edges is an unknown but constant function bounded by known

BP_e, BP	the set of all bottleneck points on edge e and on G, respectively.
$c_e(x)$	the parts per unit of coverage function of e.
$C_e(x), C(x)$	the coverage area of x on e and on G, respectively.
$d(x, y)$	distance between two points $x, y \in G$.
$e = [k, l]$	edge, where $k, l \in V$.
e_x	edge that contains point x.
E	the set of edges.
$E^c(x), E^u(x), E^p(x)$	the set of edges that are covered, uncovered and partially covered by x.
EP_e, EP	the set of all exact coverage points on edge e and on G, respectively.
$G = (V, E)$	network with node set V and edge set E and the continuum set of points of it.
IC_e, IC	the set of all identical coverage points on edge e and on G, respectively.
ℓ_e	length of edge $e \in E$.
lb_e, ub_e	the functions that specify the minimal and maximal demand over edge e.
m	number of edges.
n	number of nodes.
NP_e, NP	the set of all network intersect points on edge e and on G, respectively.
PP_e, PP	the set of all partition points on edge e and on G, respectively.
$r(x)$	the maximal regret of location x over all possible demand realizations.
$r_e(x, y), r(x, y)$	the maximal regret for x with respect to y on an edge e and on G, respectively.
R	the coverage radius.
$s_e^+(x), s_e^-(x)$	the edge coverage functions of e.
$x = (e_x, t_x)$	point in edge $e_x = [k, l]$ where t_x is the relative distance of x from k with respect to l_{e_x}.
$[x_1, x_2] = (e, [t_1, t_2])$	subedge of e, where $x_1 = (e, t_1)$, $x_2 = (e, t_2)$, and $0 \leq t_1 < t_2 \leq 1$.
V	the set of nodes.
w_e	demand realization on edge e.

Table 5.1. *Notation used in the chapter.*

lower and upper constant functions. We will derive theoretical properties of the solution with the objective of providing an exact algorithm to solve the problem in polynomial time.

We first look at the covered demand. Slightly abusing notation, we denote by ub_e, lb_e, and w_e the constant value of the lower and upper bound function and the demand realization, respectively, on e. Let $x = (e_x, t_x)$ be given with

$e_x = [k, l] \in E$ and $t_x \in [0, 1]$. Using (5.5), we obtain

$$g(x, w) = w_{e_x}(s^-_{e_x}(x) - s^+_{e_x}(x)) + \sum_{e \in E^c(x)} w_e + \sum_{e \in E^p(x)} w_e \left(1 - (s^-_e(x) - s^+_e(x))\right). \tag{5.6}$$

By Proposition 5.1, we obtain that the function $g(x, w)$ is linear in x over each partition edge of e_x and it allows us to get the following result.

Proposition 5.2 (Berman et al. (2016)). *PP is a finite dominating set for the single facility maximal covering location problem with constant demand on each edge.*

Remark 5.3.1. *The finite dominating set does not depend on the actual realization w.*

From Proposition 5.2 and Observation 5.3.1, we immediately obtain the following result:

Lemma 5.3. *The maximal regret of choosing x over all possible constant demand realizations on each edge is* $r(x) = \max\limits_{lb \le w \le ub} \left(\max\limits_{y \in PP} g(y, w) - g(x, w) \right)$.

In the following, we denote

$$c_e(x) := \begin{cases} 1, & \text{if } e \in E^c(x), \\ 1 - (s^-_e(x) - s^+_e(x)), & \text{if } e \in E^p(x), \\ 0, & \text{if } e \in E^u(x), \\ s^-_{e_x}(x) - s^+_{e_x}(x), & \text{if } e = e_x, \end{cases}$$

as the *parts per unit of coverage* of the respective edge. Observe that $0 \le c_e(x) \le 1$ and $g(x, w) = \sum_{e \in E} w_e\, c_e(x)$. For a given partition point $y \in PP$, the maximal regret for x with respect to y on an edge $e \in E$ can be computed as:

$$r_e(x, y) := \max_{lb_e \le w_e \le ub_e} g_e(y, w_e) - g_e(x, w_e). \tag{5.7}$$

Therefore, the maximal regret for x with respect to y on the entire network is given by:

$$\begin{aligned} r(x, y) &:= \max_{lb \le w \le ub} (g(y, w) - g(x, w)) = \max_{lb \le w \le ub} \sum_{e \in E} (g_e(y, w_e) - g_e(x, w_e)) \\ &= \sum_{e \in E} \max_{lb_e \le w_e \le ub_e} (g_e(y, w_e) - g_e(x, w_e)) = \sum_{e \in E} r_e(x, y) \\ &= \sum_{e \in E} \max_{lb_e \le w_e \le ub_e} w_e\,(c_e(y) - c_e(x)) \end{aligned}$$

$$= \sum_{e \in E: c_e(y) \geq c_e(x)} ub_e \, (c_e(y) - c_e(x)) + \sum_{e \in E: c_e(y) < c_e(x)} lb_e \, (c_e(y) - c_e(x)). \tag{5.8}$$

Theorem 5.1. *The maximal regret of $x \in G$ is given by*

$$r(x) \;=\; \max_{y \in PP} \sum_{e \in E} \left(g_e(y, w_e^{x,y}) - g_e(x, w_e^{x,y})\right) \,, \tag{5.9}$$

where

$$w_e^{x,y} \;=\; \begin{cases} ub_e, & \text{if } c_e(y) \geq c_e(x), \\ lb_e, & \text{otherwise,} \end{cases}$$

is the worst-case demand realization with respect to x and y.

Proof:

Let y^* and w^* be the worst-case alternative and worst-case demand realization, respectively, for $x \in G$, see expression (5.3), i.e., $r(x) = g(y^*, w^*) - g(x, w^*)$. From Proposition 5.2 and (5.8) we can assume without loss of generality that $y^* \in PP$ and $w^* = w^{x,y^*}$, respectively. Thus,

$$r(x) \;=\; \max_{y \in PP} \left(g(y, w^{x,y}) - g(x, w^{x,y})\right) \,.$$

$\square$

Having established an easy way to compute the maximal regret of x, the only open question is how to optimize x over e_x. Let $[z^1, z^2]$ be a partition edge of e_x and $x \in \operatorname{int}([z^1, z^2])$. Moreover, let $y \in PP$ be given. From Proposition 5.1 we know that the edge coverage functions are linear over $[z^1, z^2]$ and that the sets $E^p(x)$, $E^c(x)$, and $E^u(x)$ are identical for all $x \in [z^1, z^2]$. What, however, can change over $[z^1, z^2]$ is the demand realization $w^{x,y}$ that yields the maximal regret for x and y. In that case, there must exist an edge $e \in E$ and a point $z \in [z^1, z^2]$ such that z provides the exact same coverage for e as y, i.e., $c_e(z) = c_e(y)$. To see that, we next look at the four possible cases (in the remaining cases, i.e., $e \in E^c(z) \cap E^c(y)$ and $e \in E^u(z) \cap E^u(y)$, $r_e(z, y) = 0$ for any $z \in [z^1, z^2]$):
If $e \in E^p(z) \cap E^p(y)$, or $z \in e$ and $y \in e$, then:

$$c_e(z) \;=\; c_e(y) \quad \Leftrightarrow \quad s_e^-(z) - s_e^+(z) \;=\; s_e^-(y) - s_e^+(y).$$

If $e \in E^p(z)$ and $y \in e$, then:

$$c_e(z) \;=\; c_e(y) \quad \Leftrightarrow \quad 1 - (s_e^-(z) - s_e^+(z)) \;=\; s_e^-(y) - s_e^+(y).$$

Finally, if $z \in e$ and $e \in E^p(y)$, then:

$$c_e(z) \;=\; c_e(y) \quad \Leftrightarrow \quad s_e^-(z) - s_e^+(z) \;=\; 1 - (s_e^-(y) - s_e^+(y)).$$

As the edge coverage functions, $s_e^+(z)$ and $s_e^-(z)$, for $z \in [z^1, z^2]$ are linear functions, the equation $c_e(z) = c_e(y)$ will have either at most one solution or a subinterval contained in $[z^1, z^2]$ as solution. In the former case, if a solution exists it will be called *identical coverage point* with respect to y and e and in the later case, without loss of generality, the extremes of this subinterval will be called the identical coverage points of $[z^1, z^2]$ with respect to y and e. Furthermore, in both cases the solution is independent of w. We define $IC_{e_x}(y)$ as the set of all identical coverage points of G induced by y and e_x. Moreover, we define $IC = \bigcup_{e_x \in E, y \in PP} IC_{e_x}(y)$.

Theorem 5.2. *Let z^1, z^2 be two consecutive elements of $PP \cup IC$ on an edge $e_x \in E$ and $y \in PP$. Then*

1. *$g(y, w^{x,y}) - g(x, w^{x,y})$ is linear in x over $[z^1, z^2]$.*
2. *$r(x)$ is piecewise linear and convex in x over $[z^1, z^2]$.*

Proof:

Let z^1, z^2 be two consecutive elements of $PP \cup IC$ and let $x' \in \text{int}([z^1, z^2])$. Moreover, let $y \in PP$ and $w^{x',y}$ the corresponding worst-case demand realization. By definition, we can assume without loss of generality that $w^{x',y} = w^{x,y}$ for all $x \in [z^1, z^2]$. From Proposition 5.1 we know that the edge coverage functions are linear over $[z^1, z^2]$ and that the sets $E^p(x)$, $E^c(x)$, and $E^u(x)$ are identical for all $x \in [z^1, z^2]$. Hence, $g(y, w^{x,y}) - g(x, w^{x,y})$ is linear in x over $[z^1, z^2]$. The second result then follows immediately from the previous statement and Theorem 5.1. $\square$

Remark 5.3.2. *Theorem 5.2 gives us a partition in the domain of the function $r(x)$ where it is convex. The optimal solution of the problem is the minimum of this function, which can be computed as the minimum of the minima in each subedge. However, we propose an alternative strategy for computing the optimal solution of this problem whose complexity is smaller than that of this mentioned procedure.*

Theorem 5.3. *For a given $e \in E$ and $y \in PP$, $r_e(x, y)$ is a piecewise linear function for $x \in e_x$ with a constant number of pieces and $r(x, y)$ is a piecewise linear function with $\mathcal{O}(m)$ number of pieces for $x \in e_x$.*

Proof:

Let $e_x, e \in E$ and $y \in PP$. By Lemma 5.2, we know that the edge coverage functions $s_e^+(x)$, and $s_e^-(x)$ are piecewise linear functions over $e_x \in E$ with a constant number of breakpoints for $x \in e_x$. Hence, by definition, for each $e \in E$, $c_e(x)$ is also a piecewise linear function with a constant number of breakpoints on e_x. Thus using (5.8) and the fact that the number of identical coverage points on e_x for an edge $e \in E$ is constant, by Theorem 5.1 there is a constant number of possible worst-case demand realization, $w_e^{x,y}$ and then, $r_e(x, y)$ is a piecewise linear function with a constant number of pieces. $\square$

Theorems 5.1 and 5.3 give rise to Algorithm 5.1 to find an optimal solution for the (MMR-EMCLP) with unknown constant demand realizations.

Algorithm 5.1: Optimal algorithm for the single facility (MMR-EMCLP) with unknown constant demand

```
   Input: Network $G = (V, E)$; lower and upper bounds $lb_e$, $ub_e$,
          respectively for $e \in E$; coverage radius $R > 0$.
   Output: Optimal solution $x^*$.
 2 Determine the set $PP$ of partition points.
 4 foreach $e_x \in E$ do
 6     foreach $y \in PP$ do
 8         Derive the representation of $r_e(x, y)$, for each $e \in E, x \in e_x$.
10         Obtain the representation of $r(x, y)$, for $x \in e_x$.
12     Obtain the upper envelope of $r(x, y)$ for $y \in PP$, i.e., obtain $r(x)$
         for $x \in e_x$.
14     Find the minimum $x'_{e_x}$ of $r(x)$ over $e_x$.
16     if $r(x'_{e_x}) < r(x^*)$ then set $x^* := x'_{e_x}$, $r(x^*) = r(x'_{e_x})$.
18 return $x^*$.
```

For computing the complexity of the Algorithm 5.1 some technical results are needed about the complexity of computing an upper envelope of Jordan arcs. This complexity is expressed in terms of $\lambda_s(n)$, the maximum length of a Davenport-Schinzel sequence of order s on n symbols, see Sharir and Agarwal (1995). The results used in the following proof can be found in the Appendix.

Theorem 5.4. *The single facility (MMR-EMCLP) with unknown constant demand realizations on each edge can be solved exactly in $\mathcal{O}(m\lambda_3(m^3))$ time using Algorithm 5.1.*

Proof:

As the time to compute each partition point is constant, Step 1 requires $\mathcal{O}(m^2)$ time (Berman et al. (2016)).

Since there are m edges, $r(x, y)$ has m addends, each one being a piecewise linear function with a constant number of pieces for each $y \in PP$ and

$e \in E$, (Theorem 5.3). Step 4 takes constant time for each $e \in E$ and fixed $y \in PP$. Besides Step 5 generates a function $r(x, y)$ piecewise linear with $\mathcal{O}(m)$ breakpoints.

Step 6 obtains the upper envelope of $\mathcal{O}(m^3)$ line segments, i.e., $\mathcal{O}(m)$ line segments for each $y \in PP$. Since they are line segments, we have $s = 1$. Therefore, this step takes $\mathcal{O}(\lambda_2(m^3)\log m) = \mathcal{O}(m^3 \log m)$ time (Theorem 5.9). In addition, the complexity of computing the minimum in Step 7 is dominated by the complexity of the upper envelope. Hence, this step takes $\mathcal{O}(\lambda_3(m^3))$ time and Step 8 takes constant time.

Step 4 is executed once for each $e_x \in E, y \in PP$ and $e \in E$, i.e., $\mathcal{O}(m^4)$ times. Moreover Step 5 is executed once for each $e_x \in E, y \in PP$, i.e., $\mathcal{O}(m^3)$. Similarly, Step 6-8 are executed once for each $e_x \in E$, i.e., $\mathcal{O}(m)$ times. Thus, the overall complexity of the algorithm is $\mathcal{O}(m\lambda_3(m^3))$. □

Remark 5.3.3. *Recall that $\lambda_3(m) = \Theta(m\alpha(m))$ where $\alpha(m)$ is the functional inverse of the Ackermann's function. A weaker but simpler upper bound is $\lambda_3(m) = \mathcal{O}(m \log m)$, see Sharir and Agarwal (1995) for further details.*

Example 5.1. *Consider the network depicted in Figure 5.3. For each edge $e \in E$, its length is printed next to the edge. Let $R = 1$, $lb_{[1,2]} = 3$, $ub_{[1,2]} = 15$, $lb_{[2,3]} = 1$, $ub_{[2,3]} = 7$, $lb_{[1,3]} = 2$, and $ub_{[1,3]} = 8$. The set of partition points is given by $PP = V \cup \{([1,3], 1/3), ([1,3], 2/3), ([2,3], 1/2)\}$. The three partition points not included in V, indicated as dots in the figure, are the bottleneck point $([1,3], 2/3)$ and the three network intersect points $(([1,3], 1/3), ([1,3], 2/3)$, and $([2,3], 1/2)$.*

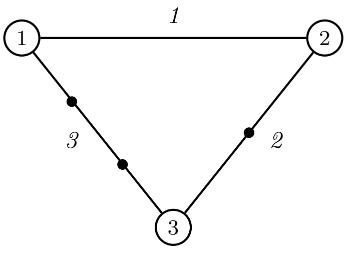


Figure 5.3. *Network with edge lengths.*

First, we derive the representation of the edge coverage functions and the parts per unit coverage functions. Their expressions can be found in Appendix 5.6.2. Next, for each edge e_x, we compute the maximal regret for $x \in e_x$

with respect to $y \in PP$:

$$r(x,y) = \sum_{e \in E : c_e(y) \geq c_e(x)} ub_e \left(c_e(y) - c_e(x)\right) + \sum_{e \in E : c_e(y) < c_e(x)} lb_e \left(c_e(y) - c_e(x)\right).$$

In Figure 5.4, functions $r(x_1, y)$, for $(x_1, y) \in [1,2] \times PP$, are depicted. The upper envelope, $r(x)$, is represented as a dotted line. Note that $r(x_1, y_3)$, is equal to $r(x_1, y_4)$, for $y_3 = ([1,3], 1/3)$ and $y_4 = ([1,3], 2/3)$. The reason of it is that the parts per unit of coverage functions are equal. As can be observed, the minimum is found in $x'_{[1,2]} = \left([1,2], \frac{2}{3}\right)$ and the regret is $r(x'_{[1,2]}) = \frac{13}{9}$.

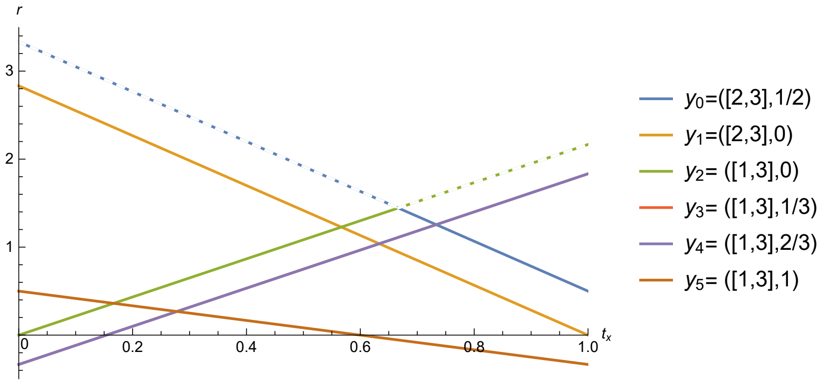


Figure 5.4. *Representation of $r(x_1, y)$, $(x_1, y) \in [1,2] \times PP$.*

Similarly, functions $r(x_2, y)$, for $(x_2, y) \in [2,3] \times PP$, are depicted in Figure 5.5. The upper envelope, $r(x)$, is represented as a dotted line. As before, observe that $r(x_2, y_3)$, is equal to $r(x_2, y_4)$, for $y_3 = ([1,3], 1/3)$ and $y_4 = ([1,3], 2/3)$. Over this edge, $[2,3]$, the minimum is found in $x'_{[2,3]} = ([2,3], 0)$ and the regret is $r(x'_{[2,3]}) = \frac{13}{6}$.

Finally, functions $r(x_3, y)$, for $(x_3, y) \in [1,3] \times PP$, are represented in Figure 5.6. The upper envelope, $r(x)$, is depicted as a dotted line. As before, note that $r(x_3, y_3)$, is equal to $r(x_3, y_4)$, for $y_3 = ([1,3], 1/3)$ and $y_4 = ([1,3], 2/3)$. Over this edge, $[1,3]$, the minimum is $x'_{[1,3]} = ([1,3], 0)$ and the regret is $r(x'_{[1,3]}) = \frac{10}{3}$.

Therefore, the optimal location of the facility with the objective of minimizing the maximal regret is $x^\star = \left([1,2], \frac{2}{3}\right)$ and the regret is $r(x^\star) = \frac{13}{9}$.

To highlight the usefulness of our max-regret approach, we also compute the optimal location for a deterministic version of the problem. To that end, we assume that the demand is known and equal to the mean of the upper and lower bound functions over each edge, i.e., $w_e = 0.5(ub_e + lb_e)$. Afterwards, we determined the optimal solution x^*_D for this deterministic version using the

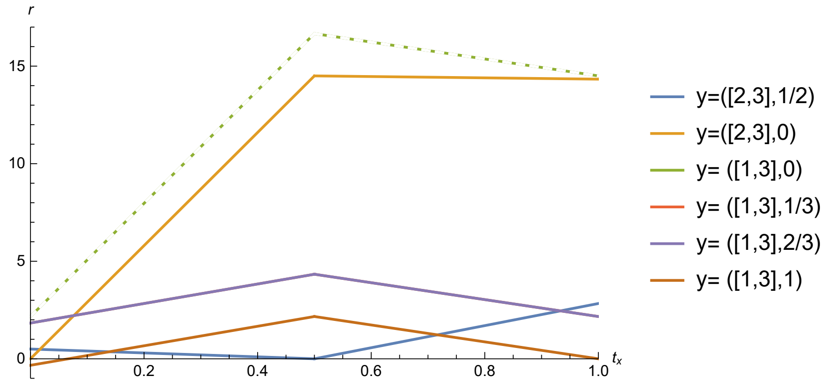


Figure 5.5. *Representation of* $r(x_2, y)$, $(x_2, y) \in [2, 3] \times PP$.

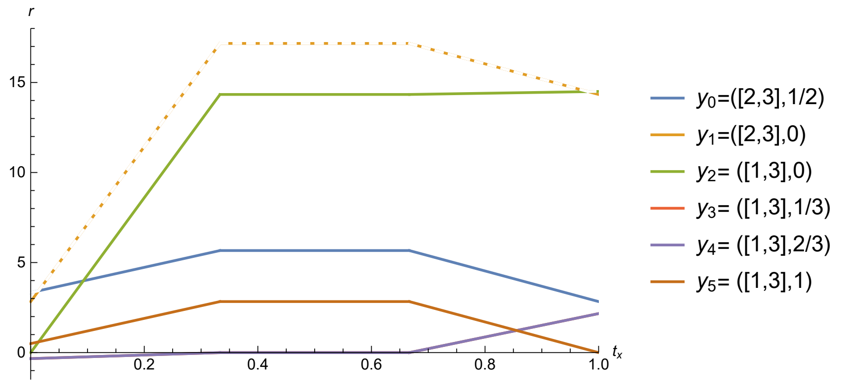


Figure 5.6. *Representation of* $r(x_3, y)$, $(x_3, y) \in [1, 3] \times PP$.

algorithm in Berman et al. (2016). Finally, we computed the maximal regret of this solution, i.e., $r(x^*_D)$*. The optimal solution of the deterministic problem is vertex 2, whose maximal regret is* $r(x^*_D) = \frac{13}{6}$*. Moreover, we also calculated the amount of covered demand in the deterministic version of the problem for the solution* x^*_D *as well as for the optimal solution* $x^\star$ *of the max-regret problem. The covered demand (the objective value of the deterministic model) is 11 and 10.8889, respectively. Therefore, while the amount of covered demand for both locations is almost identical, the maximal regret of the minmax-regret solution is significantly lower than the one for the deterministic solution. The solutions of both models, their maximal regret, and the covered demand (assuming that the demand is deterministic) are given in Table 5.2.*

Model	*Optimal solution*	*Maximal Regret*	*Covered Demand*
(MMR-EMCLP)	$([1,2],\frac{2}{3})$	$\frac{13}{9}$	*10.8889*
Berman et al. (2016)	$([1,2],1)$	$\frac{13}{6}$	*11*

Table 5.2. *Solutions for Example 5.1.*

5.3.1 Computational experiments

In this section we present computational experiments for Algorithm 5.1 for randomly generated as well as real-world data sets. Starting with the former, for a given number of vertices $n \in \{40, 60, 80, 100\}$ and a given edge density $p \in \{0.1, 0.2, 0.3\}$ (density as the percentage of the edges of a complete graph), we randomly generated five connected graphs (in total 60 graphs). For each graph, we first drew the edge lengths from a uniform distribution over $[1, 20]$. Afterwards, we computed the distance matrix and checked that the graph did not violate the triangle inequality (if this happened, we replaced the corresponding edge lengths with the shortest path distances and recalculated the distance matrix; we repeated this process until the triangle inequality was satisfied for all edges). In the next step, we randomly generated the values lb_e and ub_e for each edge. To that end, we fixed a value $UB \in \{10, 50, 100\}$ and drew lb_e and ub_e from a uniform distribution over $[0, UB/2]$ and $[UB/2, UB]$, respectively. Finally, the coverage radius $R \in \{0.1 \cdot d^{max}, 0.2 \cdot d^{max}, 0.3 \cdot d^{max}\}$ is taken as a fixed percentage of the diameter $d^{max} = \max_{i,j\in V} d(i,j)$ of the graph. As a result, for a given combination of n and p, i.e., a given graph, we have nine different instances. The two real-world data sets are based on different street graphs from a German city, with the edge lengths being the length of the street in meters. As we did not have access to actual demand data, we generated upper and lower bounds in the same way as for the randomly generated graphs. Analogously, we chose the coverage radius $R \in \{0.1 \cdot d^{max}, 0.2 \cdot d^{max}, 0.3 \cdot d^{max}\}$.

We compare our algorithm with two alternative versions. The first alternative is the node-restricted (MMR-EMCLP), i.e., we replace $\min_{x\in G}$ by $\min_{x\in V}$ in (5.4). The motivation for the second alternative was to get an idea about the value of the stochastic solution for this setting. To that end, we first replaced the unknown constant demand on an edge e by a constant known demand value $w_e = (ub_e + lb_e)/2$. Afterwards, we determined the optimal solution for this deterministic problem using the algorithm in Berman et al. (2016). Finally, we computed the maximal regret of this solution. In the following, we call these two variants simply the *node-restricted* and the *deterministic* algorithm, and our method the *max-regret* algorithm. The corresponding problems and optimal solutions are denoted analogously. All algorithms were implemented in C++ and run on a Windows 10 Laptop with a i5-8350U CPU with 1.9 GHz and 8 GB RAM.

5.3.1 Random graphs

We start with the randomly generated graphs. We first analyse the effect of varying the edge densities. The results are given in Table 5.3. The first two columns specify the number of nodes and the edge density. The next column shows the average total run time of the max-regret algorithm. The following three columns are for the node-restricted algorithm. The first shows the average relative percentage deviation of the maximal regret of the node-restricted solution with respect to the maximal regret of max-regret solution. The next two columns present the maximum relative percentage deviation and the total run time. The following three columns show exactly the same information, but this time for the deterministic algorithm. All averages are taken over the 45 different instances that have the same edge density and number of nodes (five different graphs per n and p, and nine different instances per graph).

		Max regret	Node-restricted			Deterministic		
Nodes	Density	Time	Avg Dev	Max Dev	Time	Avg Dev	Max Dev	Time
40	0.1	2.10	0.8%	7.6%	0.05	21.8%	318.2%	0.03
40	0.2	16.77	0.6%	13.9%	0.18	6.0%	47.0%	0.09
40	0.3	45.66	4.4%	84.3%	0.35	8.3%	84.3%	0.20
60	0.1	30.62	2.3%	31.2%	0.35	11.1%	63.7%	0.18
60	0.2	160.34	0.0%	0.1%	1.15	3.8%	56.2%	0.43
60	0.3	328.37	0.0%	0.0%	2.14	0.7%	15.3%	0.83
80	0.1	244.91	0.0%	0.1%	1.90	2.0%	24.1%	0.49
80	0.2	886.53	3.8%	32.5%	4.83	9.7%	32.5%	1.33
80	0.3	2022.59	0.2%	4.2%	8.15	1.7%	25.1%	2.43
100	0.1	1694.93	0.5%	18.7%	4.12	3.7%	43.4%	1.00
100	0.2	3203.70	0.5%	6.1%	11.51	1.3%	25.1%	2.91
100	0.3	7122.23	0.4%	8.2%	21.96	0.7%	8.2%	5.58

Table 5.3. *Comparison for different edge densities for the random graphs.*

As expected, the run time increases with an increasing number of nodes and edge density (the latter is due to an increase in the number of partition and identical coverage points, see Table 5.4 for more details). While the average percentage deviation for the node-restricted algorithm is quite small, the maximal percentage deviation for an instance can be very large, indicating significantly inferior solutions; and even more so for the deterministic algorithm. In general, the average percentage deviation for the deterministic problems seems to be decreasing with an increase in the edge density p. This becomes more apparent once we average for each edge density p over the number of nodes, obtaining average deviations of 9.63%, 5.19%, and 2.84% for $p = 0.1, 0.2$, and

0.3, respectively. Concerning the node-restricted problem, the opposite seems to be the case. Averaging the average percentage deviations over the number of nodes, we obtain percentages of 0.89%, 1.23%, and 1.23%. Finally, if we average the average percentage deviations for each number of nodes over the edge densities, we can observe that the averages generally decrease with an increasing number of nodes. For the node-restricted algorithm (deterministic algorithm), we obtain averages of 1.9%, 0.8%, 1.3%, and 0.4% (12.0%, 5.2%, 4.5%, and 1.9%).

In Table 5.4 we present the average number of partition points, $\#PP$, and identical coverage points, $\#ICP$ (which are not at the same time also partition points), as well as the average times to calculate them.

Nodes	Density	#PP	Time	#ICP	Time
40	0.1	983.7	0.0	1018.1	0.6
40	0.2	2119.7	0.0	1274.4	5.3
40	0.3	2773.5	0.0	1670.9	16.3
60	0.1	2735.1	0.0	1672.5	9.8
60	0.2	4603.5	0.0	2375.3	53.6
60	0.3	5824.7	0.0	2374.3	100.9
80	0.1	4971.6	0.0	2712.2	81.6
80	0.2	8066.8	0.0	3567.7	347.4
80	0.3	10089.6	0.0	4228.7	713.8
100	0.1	7622.7	0.0	3924.7	237.6
100	0.2	11770.1	0.0	4979.5	906.8
100	0.3	14640.46	0.0	6222.7	2423.6

Table 5.4. *Statistics for the number for partition and identical coverage points.*

Next, we turn to analyzing the effect of varying coverage radii. The results are given in Table 5.5, where $\%R$ denotes the percentage of the diameter of the graph that constitutes the coverage radius R, e.g., $\%R = 0.1$ means that $R = 0.1 \cdot d^{max}$. All other columns are identical to Table 5.3.

		Max regret	Node-restricted			Deterministic		
Nodes	%R	Time	Avg Dev	Max Dev	Time	Avg Dev	Max Dev	Time
40	0.1	10.06	0.7%	13.9%	0.15	23.2%	318.2%	0.08
40	0.2	18.28	0.5%	7.6%	0.18	5.9%	73.5%	0.12
40	0.3	36.18	4.6%	84.3%	0.24	6.9%	84.3%	0.13
60	0.1	90.29	0.0%	1.4%	0.99	4.3%	63.7%	0.41
60	0.2	157.42	2.2%	31.2%	1.17	5.9%	45.6%	0.49
60	0.3	271.63	0.0%	0.6%	1.49	5.4%	56.2%	0.54
80	0.1	544.87	0.2%	4.2%	4.49	5.8%	28.4%	1.23
80	0.2	940.82	2.3%	32.5%	4.80	4.6%	32.5%	1.37
80	0.3	1668.34	1.5%	19.6%	5.60	3.0%	19.6%	1.66
100	0.1	1669.05	0.2%	2.9%	11.01	3.6%	43.4%	2.79
100	0.2	4053.70	0.4%	6.1%	11.44	0.8%	6.8%	2.99
100	0.3	6298.12	0.8%	18.7%	15.15	1.3%	18.9%	3.72

Table 5.5. *Comparison for different coverage radii for the random graphs.*

As expected, the run time increases with an increasing coverage radius. While, again, the average percentage deviations for the node-restricted algorithm are quite small (in contrast to the deterministic algorithm), the maximal percentage deviations can be very large. Concerning varying coverage radii, the individual averages do not show a consistent trend. However, if we average the average percentage deviations for each coverage radius over the number of nodes, we can again observe opposing trends for the deterministic and the node-restricted algorithm. For the former, the averages strictly decrease with an increasing coverage radius (9.22%, 4.29%, and 4.15%), while for the latter they strictly increase (0.27%, 1.36%, and 1.72%).

Finally, we turn to the effect of varying UB for the generation of the upper and lower bounds on the edges. The results are given in Table 5.6, where UB denotes the value of the parameter used for generating the upper and lower bounds on the demand on an edge. All other columns are the same as before. We can make very similar observations as for the previous two comparisons.

		Max regret	Node-restricted			Deterministic		
Nodes	UB	Time	Avg Dev	Max Dev	Time	Avg Dev	Max Dev	Time
40	10	21.5	2.0%	53.9%	0.2	9.1%	140.2%	0.1
40	50	21.4	2.2%	84.3%	0.2	12.1%	251.1%	0.1
40	100	21.4	1.6%	50.2%	0.2	14.8%	318.2%	0.1
60	10	171.8	0.8%	26.4%	1.2	7.1%	63.7%	0.4
60	50	171.5	0.6%	28.1%	1.1	4.0%	56.2%	0.4
60	100	175.9	0.8%	31.2%	1.2	4.5%	46.4%	0.5
80	10	1045.1	1.5%	29.1%	5.0	4.4%	29.1%	1.4
80	50	1041.3	1.1%	25.8%	4.9	4.5%	28.0%	1.4
80	100	1067.4	1.3%	32.5%	4.8	4.5%	32.5%	1.4
100	10	3148.2	0.4%	8.2%	12.7	2.8%	43.4%	3.2
100	50	3255.5	0.3%	5.7%	12.5	1.0%	19.9%	3.1
100	100	3172.8	0.6%	18.7%	12.3	1.8%	30.2%	3.1

Table 5.6. *Comparison for different values for UB for the random graphs.*

A striking difference, however, is that the maximal percentage deviations for the node-restricted algorithm are very high for all combinations of n and UB. Moreover, no consistent trend in the average percentage deviations can be observed with respect to increasing values of UB. This time, also averaging over the number of nodes for each value of UB does not reveal anything (for the deterministic algorithm, we obtain averages of 5.85%, 5.4%, and 6.41% for $UB = 10, 50$, and 100, respectively, and for the node-restricted problem we have 1.19%, 1.06%, and 1.1%). One might have expected that larger values of UB would mean a larger "range of uncertainty" resulting in higher percentage deviations. But this is only evident for the deterministic algorithm for the 40-node instances. Observe that while the range of uncertainty is much larger in absolute values, it does not change in relative values.

5.3.2 Real-world graphs

The two real-world data sets have 132 edges and 213 edges (with 106 nodes and 143 nodes, respectively). We carry out an analogous analysis as for the random data sets, starting with the coverage radii. The results are shown in Table 5.7, where the values are just averaged over UB now.

		Max regret	Node-restricted			Deterministic		
Edges	%R	Time	Avg Dev	Max Dev	Time	Avg Dev	Max Dev	Time
132	0.1	26.0	0.1%	1.2%	0.2	34.1%	72.6%	0.1
132	0.2	106.6	1.1%	3.4%	0.4	8.9%	32.7%	0.1
132	0.3	196.8	1.1%	2.9%	0.5	25.1%	45.0%	0.1
213	0.1	536.5	1.6%	7.4%	1.3	31.3%	61.2%	0.2
213	0.2	1652.2	3.6%	8.4%	2.1	12.7%	20.3%	0.4
213	0.3	2954.9	12.4%	25.3%	2.9	80.0%	108.8%	0.5

Table 5.7. *Comparison for different coverage radii for the street graphs.*

We can make similar observations concerning the run times. For the node-restricted algorithm, there seems to be a trend with respect to increasing values of R resulting in increasing average deviations, but more tests would be required to verify this observation.

In Table 5.8 we present the number of partition points and identical coverage points (which are not at the same time also partition point), as well as the average times to calculate them. As UB does not affect the number of points, the values are not averages but the actual numbers for both street graphs.

Edges	%R	#PP	Time	#ICP	Time
132	0.1	1451	0.0	15251	0.2
132	0.2	1975	0.0	33460	1.8
132	0.3	2481	0.0	48034	5.7
213	0.1	3936	0.0	63138	8.8
213	0.2	6035	0.0	110884	41.6
213	0.3	7008	0.0	128882	70.0

Table 5.8. *Statistics for the number for partition and identical coverage points for the street graphs.*

Finally, we turn to the effect of varying UB for the generation of the upper and lower bounds on the edges. The results are given in Table 5.9, where the values are just averaged over $\%R$.

		Max regret	Node-restricted			Deterministic		
Edges	UB	Time	Avg Dev	Max Dev	Time	Avg Dev	Max Dev	Time
132	10	101.3	1.0%	3.4%	0.3	30.3%	71.1%	0.1
132	50	108.7	0.7%	3.3%	0.3	23.7%	72.6%	0.1
132	100	119.3	0.7%	2.8%	0.4	14.2%	37.3%	0.1
213	10	1607.8	5.7%	25.3%	2.2	33.3%	94.1%	0.4
213	50	1941.1	5.0%	23.9%	2.1	43.4%	99.6%	0.3
213	100	1594.7	6.9%	23.9%	2.0	47.3%	108.8%	0.4

Table 5.9. *Comparison for different values for UB for the street graphs.*

As for the random data sets, no clear trend in the percentage deviations can be observed with respect to varying values of UB. The maximum deviations again underline that the node-restricted and the deterministic algorithm may produce significantly inferior solutions.

In conclusion, the obtained results show the advantages of using our model when we want to minimize the maximal regret. These advantages have significant consequences on the service to be located as well as on the city's performance. For example, if a defibrillator has to be placed in the city with 213 edges (where UB=50) the obtained results means that in the worst case scenario for the demand (taking into account different coverage radii), the solution of our model leaves on average 5.0% (43.4%) less of the uncovered population by the node-restricted (deterministic) solution. These results are even more remarkable if we consider the particular value of the radius that provided the maximum difference, where the solution of our model leaves 23.9% (99.6%) less of the uncovered population by node-restricted (deterministic) solution.

5.4 The Unknown Linear Demand Case

In this section, we consider the case of an unknown linear demand realization bounded by known linear lower and upper bound functions. Let $lb_e(t) = a_e^{lb} + b_e^{lb} \cdot t$, $ub_e(t) = a_e^{ub} + b_e^{ub} \cdot t$, and $w_e(t) = a_e^w + b_e^w \cdot t$. Unfortunately, Proposition 5.2 and Observation 5.3.1 no longer hold for non-constant demand functions, although we can still easily compute the optimal solution of $\max_{x \in G} g(x, w)$ for a given w in $\mathcal{O}(m^2 \log m)$ time (Berman et al. (2016)). What, however, still works is to discretize the domain w over which we optimize to find the worst-case demand realization.

To that end, we first characterize the feasible region $lb \le w \le ub$ in terms of $a_e^{lb}, b_e^{lb}, a_e^{ub}$, and b_e^{ub}:

$$lb \le w \le ub \qquad \Leftrightarrow \qquad \begin{aligned} a_e^w &\ge lb_e(0) = a_e^{lb}, \\ a_e^w &\le ub_e(0) = a_e^{ub}, \\ a_e^w + b_e^w, &\le ub_e(1) = a_e^{ub} + b_e^{ub}, \\ a_e^w + b_e^w &\ge lb_e(1) = a_e^{lb} + b_e^{lb}, \\ a_e^w &\ge 0, b_e^w \in \mathbb{R}. \end{aligned} \tag{5.10}$$

We denote by F_e the feasible set of points (a_e^w, b_e^w) satisfying the system of inequalities on the right hand side of (5.10). F_e is a parallelogram in the a_e^w/b_e^w-space whose left and right sides are vertical and whose upper and lower sides are diagonal with slope -1. See the sketch on the left-hand side in Figure 5.7.

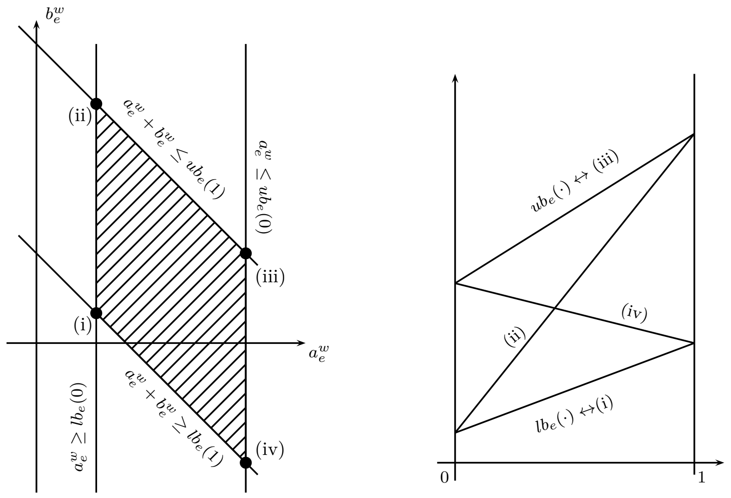

b_e^w
a_e^w
(i)
(ii)
(iii)
(iv)
$a_e^w + b_e^w \leq ub_e(1)$
$a_e^w \leq ub_e(0)$
$a_e^w + b_e^w \geq lb_e(1)$
$a_e^w \geq lb_e(0)$
$ub_e(\cdot) \leftrightarrow$ (iii)
(iv)
(ii)
$lb_e(\cdot) \leftrightarrow$(i)
0
1

Figure 5.7. *Sketch of the feasible region F_e (left-hand side) and the demand realisations corresponding to the four corners of F_e (right-hand side).*

The four corners of the parallelogram are

(i) (a_e^{lb}, b_e^{lb}),
(ii) $(a_e^{lb}, a_e^{ub} + b_e^{ub} - a_e^{lb})$,
(iii) (a_e^{ub}, b_e^{ub}),
(iv) $(a_e^{ub}, a_e^{lb} + b_e^{lb} - a_e^{ub})$.

The first and third coincide with $w_e(\cdot) = lb_e(\cdot)$ and $w_e(\cdot) = ub_e(\cdot)$, respectively. For the second and fourth, $w_e(\cdot)$ crosses diagonally between the lower and upper bound function. For example for the second point, $w_e(\cdot)$ starts at $lb_e(0)$ and ends at $ub_e(1)$. See the sketch on the right-hand side in Figure 5.7.

Proposition 5.3. *The worst-case demand realization for a fixed $x, y \in G$ and $e \in E$ can be obtained by solving the following linear program:*

$$r_e(x,y) = \max \ a_e^w \left(c_e(y) - c_e(x)\right) + \frac{1}{2} b_e^w \left(\bar{c}_e(y) - \bar{c}_e(x)\right), \qquad (5.11)$$
$$s.t. \ (5.10)$$

where

$$\bar{c}_e(x) = \begin{cases} 1, & \textit{if } e \in E^c(x), \\ 1 - ((s_e^-(x))^2 - (s_e^+(x))^2), & \textit{if } e \in E^p(x), \\ 0, & \textit{if } e \in E^u(x), \\ (s_{e_x}^-(x))^2 - (s_{e_x}^+(x))^2, & \textit{if } e = e_x. \end{cases}$$

Proof:

Let $x = (e_x, t_x) \in G$, $y = (e_y, t_y) \in G$, and $e \in E$ be given. Next, we will derive the expression of the maximal regret for the different cases. We distinguish between the situations that define the different pieces of the functions c_e and $\bar{c}_e$. We first consider the case that $e \in E^p(x) \cap E^p(y)$ and $x, y \notin e$ (Case 1.1). Then, the maximal regret with respect to x and y on e can be computed as

$$\begin{aligned} r_e(x,y) &= \max_{lb_e \le w_e \le ub_e} (g_e(y,w_e) - g_e(x,w_e)) \\ &= \max_{lb_e \le w_e \le ub_e} \int_0^{s_e^+(y)} w_e(u)\,du + \int_{s_e^-(y)}^1 w_e(u)\,du \\ &\qquad - \int_0^{s_e^+(x)} w_e(u)\,du - \int_{s_e^-(x)}^1 w_e(u)\,du \\ &= \max_{lb_e \le w_e \le ub_e} \int_{s_e^+(x)}^{s_e^-(x)} w_e(u)\,du - \int_{s_e^+(y)}^{s_e^-(y)} w_e(u)\,du \\ &= \max_{lb_e \le w_e \le ub_e} \left[a_e^w \cdot u + \frac{1}{2} b_e^w \cdot u^2\right]_{s_e^+(x)}^{s_e^-(x)} - \left[a_e^w \cdot u + \frac{1}{2} b_e^w \cdot u^2\right]_{s_e^+(y)}^{s_e^-(y)} \\ &= \max_{lb_e \le w_e \le ub_e} a_e^w \left(s_e^-(x) - s_e^+(x) - s_e^-(y) + s_e^+(y)\right) \\ &\qquad + \frac{1}{2} b_e^w \left((s_e^-(x))^2 - (s_e^+(x))^2 - (s_e^-(y))^2 + (s_e^+(y))^2\right) \\ &= \max_{lb_e \le w_e \le ub_e} a_e^w \left(c_e(y) - c_e(x)\right) + \frac{1}{2} b_e^w \left(\bar{c}_e(y) - \bar{c}_e(x)\right). \end{aligned} \tag{5.12}$$

So far we assumed that $x, y \notin e$. Now, we assume that $y \notin e$ but $x \in e$ (Case 1.2), then we obtain

$$\begin{aligned} r_e(x,y) &= \max_{lb_e \le w_e \le ub_e} \int_0^{s_e^+(y)} w_e(u)\,du + \int_{s_e^-(y)}^1 w_e(u)\,du - \int_{s_e^+(x)}^{s_e^-(x)} w_e(u)\,du \\ &= \max_{lb_e \le w_e \le ub_e} \left[a_e^w \cdot u + \frac{1}{2} b_e^w \cdot u^2\right]_0^{s_e^+(y)} + \left[a_e^w \cdot u + \frac{1}{2} b_e^w \cdot u^2\right]_{s_e^-(y)}^1 \\ &\qquad - \left[a_e^w \cdot u + \frac{1}{2} b_e^w \cdot u^2\right]_{s_e^+(x)}^{s_e^-(x)} \\ &= \max_{lb_e \le w_e \le ub_e} a_e^w \left(s_e^+(y) + 1 - s_e^-(y) - s_e^-(x) + s_e^+(x)\right) \end{aligned}$$

$$+\ \frac{1}{2}b_e^w\left((s_e^+(y))^2+1-(s_e^-(y))^2-(s_e^-(x))^2+(s_e^+(x))^2\right)$$
$$=\max_{lb_e\le w_e\le ub_e}\ a_e^w\left(c_e(y)-c_e(x)\right)+\ \frac{1}{2}b_e^w\left(\bar{c}_e(y)-\bar{c}_e(x)\right). \tag{5.13}$$

Analogously, the same expression of $r_e(x,y)$ can be obtained in the remaining two subcases: $x\notin e, y\in e$ (Case 1.3) and $x,y\in e$ (Case 1.4). Next, we analyze the cases where $e\notin E^p(x)\cap E^p(y)$. If $e\in E^p(y)\cap E^c(x)$ and $x,y\notin e$ (Case 2.1), then

$$\begin{aligned}
r_e(x,y)&=\max_{lb_e\le w_e\le ub_e}\int_0^{s_e^+(y)}w_e(u)\,du+\int_{s_e^-(y)}^1 w_e(u)\,du\ -\int_0^1 w_e(u)\,du\\
&=\max_{lb_e\le w_e\le ub_e}\left[a_e^w\cdot u+\frac{1}{2}b_e^w\cdot u^2\right]_0^{s_e^+(y)}+\left[a_e^w\cdot u+\frac{1}{2}b_e^w\cdot u^2\right]_{s_e^-(y)}^1\\
&\qquad-\left[a_e^w\cdot u+\frac{1}{2}b_e^w\cdot u^2\right]_0^1\\
&=\max_{lb_e\le w_e\le ub_e}\ a_e^w\left(s_e^+(y)+1-s_e^-(y)-1\right)+\ \frac{1}{2}b_e^w\left((s_e^+(y))^2+1-(s_e^-(y))^2-1\right)\\
&=\max_{lb_e\le w_e\le ub_e}\ a_e^w\left(c_e(y)-1\right)+\ \frac{1}{2}b_e^w\left(\bar{c}_e(y)-1\right)\\
&=\max_{lb_e\le w_e\le ub_e}\ a_e^w\left(c_e(y)-c_e(x)\right)+\ \frac{1}{2}b_e^w\left(\bar{c}_e(y)-\bar{c}_e(x)\right).
\end{aligned}\tag{5.14}$$

The remaining cases can be proven analogously. Therefore, we have proven that $r_e(x,y)$ can be obtained by solving a linear program. □

Since $r_e(x,y)$ can be reduced to solve a linear program with feasible region F_e, then, at least one of the four corners of F_e is optimal. In the following, we present the conditions to identify which corner of F_e will yield the maximal regret.

Theorem 5.5. *An optimal solution of* (5.11), (a_e^{w*},b_e^{w*}), *is given by the first column of Table 5.10 whenever the conditions of columns 2-4 are fulfilled.*

(a_e^{w*},b_e^{w*})	*Conditions*		
	$c_e(y)-c_e(x)$	$\bar{c}_e(y)-\bar{c}_e(x)$	$(\bar{c}_e(y)-\bar{c}_e(x))-2(c_e(y)-c_e(x))$
(a_e^{lb},b_e^{lb})	≤ 0	≤ 0	≥ 0
$(a_e^{lb},a_e^{ub}+b_e^{ub}-a_e^{lb})$	≥ 0	≥ 0	≥ 0
	≤ 0	≥ 0	–
(a_e^{ub},b_e^{ub})	≥ 0	≥ 0	≤ 0
$(a_e^{ub},a_e^{lb}+b_e^{lb}-a_e^{ub})$	≤ 0	≤ 0	≤ 0
	≥ 0	≤ 0	–

Table 5.10. *Worst-case demand realization; optimal solution of* (5.11).

Proof:

The objective function in (5.11) is linear in a_e^w and b_e^w. Hence, at least one of the four corners of F_e will be optimal. Thus, if $c_e(y)-c_e(x)\ge 0$ and $\bar{c}_e(y)-\bar{c}_e(x)\le 0$, an optimal solution is $(a_e^{ub},a_e^{lb}+b_e^{lb}-a_e^{ub})$, i.e., corner (iv), since we want to make a_e^w as large as possible and b_e^w as small as possible to maximize the regret. Similarly, if $c_e(y)-c_e(x)\le 0$ and $\bar{c}_e(y)-\bar{c}_e(x)\ge 0$, an optimal solution is $(a_e^{lb},a_e^{ub}+b_e^{ub}-a_e^{lb})$, i.e., corner (ii), since we want to make

a_e^w as small as possible and b_e^w as large as possible to maximize the regret. However, if $c_e(y) - c_e(x) > 0$ and $\bar{c}_e(y) - \bar{c}_e(x) > 0$, or $c_e(y) - c_e(x) < 0$ and $\bar{c}_e(y) - \bar{c}_e(x) < 0$, obtaining an optimal solution is not straightforward. Let us consider the case where $c_e(y) - c_e(x) > 0$ and $\bar{c}_e(y) - \bar{c}_e(x) > 0$, the other case can be analyzed analogously. In this case, the two candidate points to be an optimal solution of (5.11) will be $(a_e^{lb}, a_e^{ub} + b_e^{ub} - a_e^{lb})$ and (a_e^{ub}, b_e^{ub}), i.e., corners (ii) and (iii). Thus, (a_e^{ub}, b_e^{ub}) will be an optimal solution of (5.11) if the following inequality holds:

$$a_e^{lb}(c_e(y)-c_e(x))+\frac{1}{2}(a_e^{ub}+b_e^{ub}-a_e^{lb})(\bar{c}_e(y)-\bar{c}_e(x))\leq a_e^{ub}(c_e(y)-c_e(x))+\frac{1}{2}b_e^{ub}(\bar{c}_e(y)-\bar{c}_e(x)),$$

or equivalently:

$$\frac{1}{2}(a_e^{ub} - a_e^{lb})(\bar{c}_e(y) - \bar{c}_e(x)) \leq (a_e^{ub} - a_e^{lb})(c_e(y) - c_e(x)).$$

By hypothesis $a_e^{ub} - a_e^{lb} \geq 0$. If $a_e^{ub} - a_e^{lb} = 0$, corners (ii) and (iii) coincide, therefore we can assume without loss of generality that $a_e^{ub} - a_e^{lb} > 0$. Thus,

$$\frac{1}{2}(\bar{c}_e(y) - \bar{c}_e(x)) \leq c_e(y) - c_e(x).$$

It means that if $c_e(y) - c_e(x) > 0$, $\bar{c}_e(y) - \bar{c}_e(x) > 0$, and $\bar{c}_e(y) - \bar{c}_e(x) \leq 2(c_e(y) - c_e(x))$, then an optimal solution of (5.11) is (a_e^{ub}, b_e^{ub}), otherwise an optimal solution will be $(a_e^{lb}, a_e^{ub} + b_e^{ub} - a_e^{lb})$. Therefore, we obtain the conditions described in Table 5.10 and the result follows. $\square$

Theorem 5.6. *For a given $e \in E, x \in e_x \in E$, and $y \in e_y \in E$, the conditions described in Theorem 5.5 generate a subdivision of $e_x \times e_y$ with respect to e with a constant number of cells, such that the representation of $r_e(x, y)$ over each cell is a quadratic function.*

Proof:

Let $x \in e_x \in E$ and let $y \in e_y \in E$. From Lemma 5.2 and Proposition 5.1 we know that the edge coverage functions, $s_e^+(x), s_e^-(x)$ $(s_e^+(y), s_e^-(y))$ are piecewise linear functions with a constant number of pieces over $e_x \in E$ $(e_y \in E)$. Thus, for each $e \in E$, the expressions $c_e(x)$ and $\bar{c}_e(x)$ $(c_e(y)$ and $\bar{c}_e(y))$ have a constant number of explicit representations for any $x \in e_x$ $(y \in e_y)$. The breakpoints of the edge coverage functions for a given edge $e \in E$ correspond to either bottleneck points or network intersect points (Lemma 5.2). Therefore, we add the corresponding vertical and horizontal lines induced by these constant number of partition points to the subdivision. Moreover, by Theorem 5.5 we can identify the corresponding worst-case demand realizations, i.e., w_e^* in each cell. Therefore, a constant number of algebraic curves are derived from the conditions defined in Table 5.10. The arrangement inside the square $e_x \times e_y$ of these constant number of planar algebraic curves for a given $e \in E$, i.e., the vertical and horizontal lines induced by the breakpoints of the edge coverage functions and the conditions of Table 5.10, generates a subdivision with a constant number of cells because each pair of curves intersects in a constant number of points. Furthermore, within each cell of this subdivision there is a common worst-case demand realization, w_e^*, i.e., within each cell of this subdivision $r_e(x, y)$ has a unique representation as a quadratic function. □

In the following, we will compute the representation of the maximal regret in the network depicted in Figure 5.3 for a pair of edges. Moreover, we will represent the generated subdivision over the square composed by these two edges.

Example 5.2. *Consider the network depicted in Figure 5.3. For each edge $e \in E$, its length is printed next to the edge. Let $R = 1$, $lb_{[1,2]}(t) = 3 - 3t$, $ub_{[1,2]}(t) = 15 + 7t$, $lb_{[2,3]}(t) = 3t$, $ub_{[2,3]}(t) = 7 + 3t$, $lb_{[1,3]}(t) = 2 + 3t$, and $ub_{[1,3]}(t) = 8 + 10t$. The set of partition points is given by $PP = V \cup \{([1,3], 1/3), ([1,3], 2/3), ([2,3], 1/2)\}$. The three partition points not included in V (indicated as dots in the figure) are the bottleneck point $([1,3], 2/3)$ and the three network intersect points $(([1,3], 1/3), ([1,3], 2/3), ([2,3], 1/2)$.*

The network of this example is the one used in Example 5.1. Therefore, the edge coverage functions are identical. As stated before, the representation of these functions can be found in Appendix 5.6.2.

Next, we denote $F_{[1,2]}, F_{[2,3]}$, and $F_{[1,3]}$ the feasible region $lb \leq w \leq ub$ in terms of $a_e^{lb}, b_e^{lb}, a_e^{ub}$, and b_e^{ub} for $e \in \{[1,2], [2,3], [1,3]\}$ respectively, i.e., the feasible set of points satisfying the system of inequalities of the right hand side of (5.10).

Let $x = ([1,2], t_x) \in [1,2]$, i.e., $0 \leq t_x \leq 1$, and $y = ([2,3], t_y) \in [2,3]$, i.e., $0 \leq t_y \leq 1$. In the following, we compute the worst-case demand realization in $(x, y) \in [1,2] \times [2,3]$ for a fixed edge $[1,2] \in E$ applying Proposition 5.3:

$$r_{[1,2]}(x,y) = \begin{cases} \max\limits_{(a^w_{[1,2]},\, b^w_{[1,2]}) \in F_{[1,2]}} -2t_y a^w_{[1,2]} - 2t_y^2 b^w_{[1,2]}, & \text{if } 0 \leq t_y \leq \frac{1}{2}, \\ \max\limits_{(a^w_{[1,2]},\, b^w_{[1,2]}) \in F_{[1,2]}} -a^w_{[1,2]} - \dfrac{1}{2} b^w_{[1,2]}, & \text{if } \frac{1}{2} \leq t_y \leq 1. \end{cases}$$

First, we analyze the sign of the coefficients of $a^w_{[1,2]}$ and $b^w_{[1,2]}$ in both pieces. We observe that there is no change of sign in them, therefore the worst-case demand realization is constant over each piece, i.e., there are only two cells generated by the horizontal line induced by the breakpoint of $r_{[1,2]}(x,y)$. From Proposition 5.5, we obtain that the optimal solution is corner (i) for both *pieces, thus:*

$$r_{[1,2]}(x,y)=\begin{cases}-6t_y+6t_y^2, & \text{if } 0\le t_y\le \frac{1}{2},\\ -\frac{3}{2}, & \text{if } \frac{1}{2}\le t_y\le 1.\end{cases}$$

Figure 5.8 shows the subdivision generated in the square $[1,2]\times[2,3]$*, in each cell the expression of the maximal regret as a unique representation.*

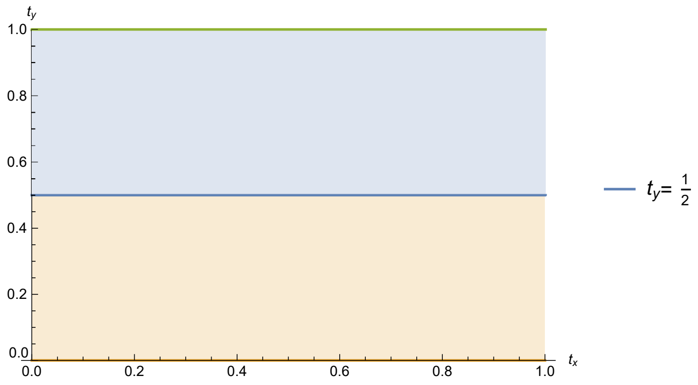


Figure 5.8. *Cells for* $r_{[1,2]}(x,y)$*,* $x=([1,2],t_x)$ *and* $y=([2,3],t_y)$*.*

Hereunder, we compute the worst-case demand realization in $(x,y)\in[1,2]\times[2,3]$ *for a fixed edge* $[2,3]\in E$ *applying Proposition 5.3:*

$$r_{[2,3]}(x,y)=\begin{cases}\max\limits_{\left(a^w_{[2,3]},\,b^w_{[2,3]}\right)\in F_{[2,3]}} \frac{1}{2}(1-t_x+2t_y)a^w_{[2,3]} + \frac{1}{8}\left(-t_x^2+(1+2t_y)^2\right)b^w_{[2,3]}, & \text{if } 0\le t_y\le\frac{1}{2},\\ \max\limits_{\left(a^w_{[2,3]},\,b^w_{[2,3]}\right)\in F_{[2,3]}} \frac{1}{2}(3-2t_y-t_x)a^w_{[2,3]} + \frac{1}{2}\left(1-\frac{1}{4}\left((2t_y-1)^2+t_x^2\right)\right)b^w_{[2,3]}, & \text{if } \frac{1}{2}\le t_y\le 1.\end{cases}$$

Since the coefficients of $a^w_{[2,3]}$ *and* $b^w_{[2,3]}$ *have a non-negative value for both definitions, from Proposition 5.5, we obtain that the changes in* $r_{[2,3]}(x,y)$ *are determined by the sign of* $(\bar{c}_{[2,3]}(y)-\bar{c}_{[2,3]}(x))-2(c_{[2,3]}(y)-c_{[2,3]}(x))$*. For* $0\le t_y\le\frac{1}{2}$*, this sign is also constant, so for this case new cells are not defined. However, for* $\frac{1}{2}\le t_y\le 1$*, the sign is not constant, then we set* $\left(1-\frac{1}{4}\left((2t_y-1)^2+t_x^2\right)\right)-(3-2t_y-t_x)=0$ *and solve for* $t_y\in\left[\frac{1}{2},1\right]$*. We get the parametric curve* $(t_x,\frac{3}{2}-\frac{1}{2}\sqrt{4t_x-t_x^2})$*, for* $2-\sqrt{3}\le t_x\le 1$*, this curve determines the change of the sign of* $(\bar{c}_{[2,3]}(y)-\bar{c}_{[2,3]}(x))-2(c_{[2,3]}(y)-c_{[2,3]}(x))$ *and as a consequence, the change of definition of* $r_{[2,3]}(x,y)$*.*

Figure 5.9 shows the subdivision generated by the horizontal line induced by the breakpoint of $r_{[2,3]}(x,y)$ *and the change of the sign of* $(\bar{c}_{[2,3]}(y)-\bar{c}_{[2,3]}(x))-2(c_{[2,3]}(y)-c_{[2,3]}(x))$*. In each cell, the representation of* $r_{[2,3]}(x,y)$ *is unique: in the green area (upper left part) it is* $r_{[2,3]}(x,y)=\frac{7}{2}(3-2t_y-$

$t_x) + \frac{3}{2}\left(1 - \frac{1}{4}((2t_y-1)^2 + t_x^2)\right)$; i.e., corner (iii) is the optimal solution, in the yellow one (upper right part) it is $r_{[2,3]}(x,y) = 5\left(1 - \frac{1}{4}((2t_y-1)^2 + t_x^2)\right)$; i.e., corner (ii) is the optimal solution, and in the blue one (lower part) it is $r_{[2,3]}(x,y) = \frac{7}{2}(1 - t_x + 2t_y) + \frac{3}{2}\left(-t_x^2 + (1+2t_y)^2\right)$; i.e., corner (iii) is the optimal solution.

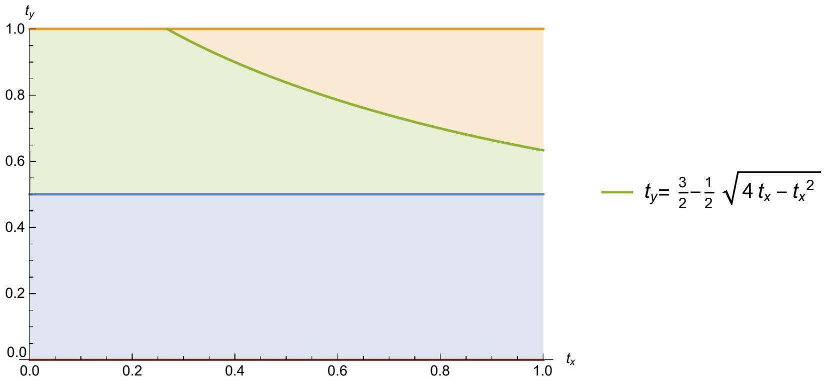


Figure 5.9. *Cells for $r_{[2,3]}(x,y)$, $x = ([1,2],t_x)$ and $y = ([2,3],t_y)$.*

Next, we compute the worst-case demand realization in $(x,y) \in [1,2]\times[2,3]$ for a fixed edge $[1,3] \in E$ applying Proposition 5.3:

$$r_{[1,3]}(x,y) = \begin{cases} \max\limits_{\left(a^w_{[1,3]},\, b^w_{[1,3]}\right)\in F_{[1,3]}} \begin{array}{l} \frac{1}{3}(-1+t_x)a^w_{[1,3]} \\ -\frac{1}{18}(1-t_x)^2 b^w_{[1,3]}, \end{array} & \text{if } 0 \le t_y \le \frac{1}{2}, \\ \max\limits_{\left(a^w_{[1,3]},\, b^w_{[1,3]}\right)\in F_{[1,3]}} \begin{array}{l} \frac{1}{3}(-2+t_x+2t_y)a^w_{[1,3]} \\ +\frac{1}{2}\left(1 - \frac{1}{9}\left((4-2t_y)^2+(1-t_x)^2\right)\right) b^w_{[1,3]}, \end{array} & \text{if } \frac{1}{2} \le t_y \le 1. \end{cases}$$

From Proposition 5.5, we obtain that the changes of definition of $r_{[1,3]}(x,y)$ are determined by the sign of $c_{[1,3]}(y) - c_{[1,3]}(x)$, $\bar{c}_{[1,3]}(y) - \bar{c}_{[1,3]}(x)$, and $(\bar{c}_{[1,3]}(y) - \bar{c}_{[1,3]}(x)) - 2(c_{[1,3]}(y) - c_{[1,3]}(x))$. In the Figure 5.10, the cells generated by the horizontal line induced by the breakpoint of $r_{[1,3]}(x,y)$ and the curves $c_{[1,3]}(y) - c_{[1,3]}(x) = 0$, $\bar{c}_{[1,3]}(y) - \bar{c}_{[1,3]}(x) = 0$, and $(\bar{c}_{[1,3]}(y) - \bar{c}_{[1,3]}(x)) - 2(c_{[1,3]}(y) - c_{[1,3]}(x)) = 0$ are depicted. In each area, the representation of $r_{[1,3]}(x,y)$ is unique: in the pink area (lower part) it is $r_{[1,3]}(x,y) = \frac{2}{3}(-1+t_x) - \frac{1}{6}(1-t_x)^2$; i.e., the corner (i) is the optimal solution, in the orange area (middle lower left part) it is $r_{[1,3]}(x,y) = \frac{2}{3}(t_x + 2(t_y-1)) + \frac{3}{2}\left(1 - \frac{1}{9}((4-2t_y)^2 + (1-t_x)^2)\right)$; i.e., the corner (i) is the optimal solution, in the blue and lavender area (upper left and middle lower right part respectively) it is $r_{[1,3]}(x,y) = \frac{2}{3}(t_x + 2(t_y-1)) + 8\left(1 - \frac{1}{9}((4-2t_y)^2 + (1-t_x)^2)\right)$, i.e., the corner (ii) is the optimal solution, and in the green (upper right part) *area it is* $r_{[1,3]}(x,y) = \frac{8}{3}(t_x + 2(t_y-1)) + 5\left(1 - \frac{1}{9}((4-2t_y)^2 + (1-t_x)^2)\right)$; *i.e., the corner (iii) is the optimal solution. It is worth noting that although the expression* $t_x = 1 - \frac{t_y}{2}$ *implies a change in the sign of* $c_{[1,3]}(y) - c_{[1,3]}(x)$, $r_{[1,3]}(x,y)$ *does not change, because the optimal corner is the same. For this reason, the blue and the lavender cells would be considered as one in the following steps.*

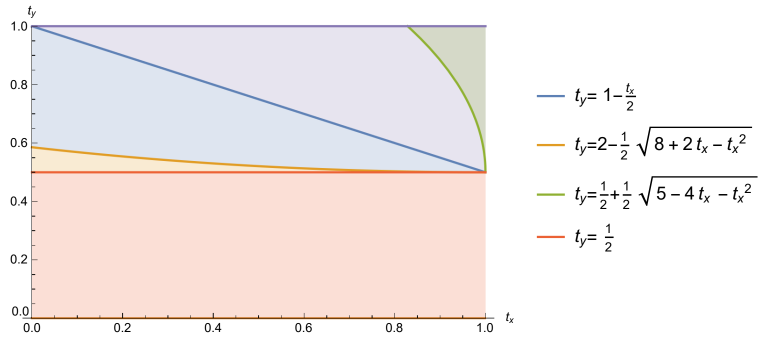


Figure 5.10. *Cells for* $r_{[1,3]}(x,y)$, $x = ([1,2], t_x)$ *and* $y = ([2,3], t_y)$.

Combining all the above subdivisions, we obtain a finer subdivision shown in Figure 5.11. The function $r(x,y)$ *has a unique representation in each cell as a quadratic function which is obtained as the sum of the corresponding* $r_e(x,y)$, *for each* $e \in E$.

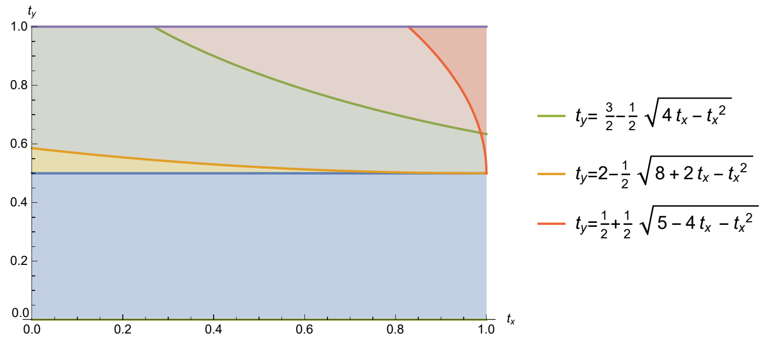


Figure 5.11. *Cells for* $r(x,y)$, $x = ([1,2], t_x)$ *and* $y = ([2,3], t_y)$.

5.4.1 Resolution method

In the current subsection, a polynomial time algorithm is developed for solving the single facility (MMR-EMCLP) with unknown linear demand realizations. We start explaining the solution methods before analyzing its complexity. The idea of the procedure is to compute a subdivision over the square $e_x \times e_y$ for $e_x, e_y \in E$ where for any $x \in e_x$, $\max_{y \in e_y} r(x,y)$ is achieved at the boundaries of the cells of this subdivision. We call $\mathcal{C}_{e_x e_y}$ the sets of arcs defining this subdivision.

Theorem 5.7. *The single facility (MMR-EMCLP) with unknown linear demand realizations can be solved to optimality in polynomial time.*

Proof:

Let $x \in e_x \in E$ and $y \in e_y \in E$. For a given $e \in E$, we derive a subdivision over the square $e_x \times e_y$ with a constant number of cells, such that, within each cell of the subdivision $r_e(x,y)$ has a unique representation (Theorem 5.6). Indeed, within each cell of the arrangement generated by the vertical and horizontal lines induced by the breakpoints of the edge coverage functions and the algebraic curves defined on Table 5.10, $r_e(x,y)$ has a unique representation as a quadratic function for each $e \in E$. Since $r(x,y)$ is the sum of $r_e(x,y)$ for all $e \in E$, the intersections of the cells generated for each $e \in E$ provide a finer subdivision in the square $e_x \times e_y$ such that, within the new cells, $r(x,y)$ has a unique representation as a quadratic function. Since $r(x,y)$ is quadratic, for a fixed x the maximum of $r(x,y)$; i.e., $\max_{y \in e_y} r(x,y)$ can be found on the boundary of the cells previously defined or, if the function is concave inside a cell, in the intersection of the curve $\frac{\partial r}{\partial y}(x,y) = 0$ (for a fixed x) with the cell. Let $\mathcal{C}_{e_x e_y}$ be the set of the previous arcs (arcs defining the boundary of a cell and the arc $\frac{\partial r}{\partial y}(x,y) = 0$ in the cells where $r(x,y)$ is concave) parametrized as $(x, y(x))$ in the square $e_x \times e_y$. The upper envelope, $h_{e_x e_y}(x)$, of $r(x, y(x))$ for all $(x, y(x)) \in \mathcal{C}_{e_x e_y}$ represents the optimal $y \in e_y$ for each $x \in e_x$ for the worst-case demand realization (a similar idea of computing the upper envelopes over parametrized curves was used in López-de-los-Mozos et al. (2013)); i.e.,

$$h_{e_x e_y}(x) \;=\; \max_{lb \le w \le ub} \left(\max_{y \in e_y} g(y,w) \;-\; g(x,w) \right), \text{ for } x \in e_x. \tag{5.15}$$

Repeating this procedure for each $e_y \in E$, the upper envelope, $h_{e_x}(x)$, of $r(x, y(x))$ for all $(x, y(x)) \in \bigcup_{e_y \in E} \mathcal{C}_{e_x e_y}$ determines the maximum regret of choosing $x \in e_x$ over the optimal location with respect to w and $y \in G$, i.e.,

$$h_{e_x}(x) \;=\; \max_{lb \le w \le ub} \left(\max_{y \in G} g(y,w) \;-\; g(x,w) \right), \text{ for } x \in e_x. \tag{5.16}$$

Let $x'_{e_x} \in e_x$ be the minimum of $h_{e_x}(x)$ for $x \in e_x$. It determines the optimal location of the minmax regret problem in e_x; i.e.,

$$h_{e_x}(x'_{e_x}) = \min_{x \in e_x} \max_{lb \leq w \leq ub} \left(\max_{y \in G} g(y,w) - g(x,w) \right). \tag{5.17}$$

This procedure should be repeated for each $e_x \in E$. The optimal solution, x^*, is $x^* = \min\limits_{e_x \in E} h_{e_x}(x'_{e_x})$. Computing the upper envelope of these arcs and finding the minimum of this upper envelope can be done in polynomial time, therefore the result follows. The procedure presented in this proof is summed up in Algorithm 5.2. □

Algorithm 5.2: Optimal algorithm for the single facility (MMR-EMCLP) with unknown linear demand realizations.

```
Input: Network $G = (V, E)$; lower and upper bounds
       $lb_e(t) = a_e^{lb} + b_e^{lb} \cdot t$ and $ub_e(t) = a_e^{ub} + b_e^{ub} \cdot t$, respectively, for
       $e \in E$; coverage radius $R > 0$.
Output: Optimal solution $x^*$.
2  foreach $e_x \in E$ do
4      foreach $e_y \in E$ do
6          Compute the subdivision generated by the arcs in $\mathcal{C}_{e_x e_y}$ :
7          i) For each edge $e$, the vertical and horizontal lines induced by
             the breakpoints of the edge coverage functions and the curves
             defining the conditions in Table 5.10.
8          ii) For any cell where $r(x,y)$ is concave in $y \in e_y$ for a fixed
             $x \in e_x$, the intersection of the curve $\frac{\partial r}{\partial y}(x,y) = 0$ with that
             cell.
10     Obtain the upper envelope, $h_{e_x}(x)$, of $r(x, y(x))$ of the arcs
         contained in $\bigcup_{e_y \in E} \mathcal{C}_{e_x e_y}$. ;
12     Find the minimum $x'_{e_x}$ of $h_{e_x}(x)$ over $e_x$.
14     if $h_{e_x}(x'_{e_x}) < r(x^*)$ then set $x^* := x'_{e_x}$, $r(x^*) = h_{e_x}(x'_{e_x})$.;
16 return $x^*$.;
```

For computing the complexity of solving the single facility (MMR-EMCLP) with unknown linear demand realizations on each edge by applying Algorithm 5.2, the following results are needed.

Lemma 5.4. *The set of arcs included in $\mathcal{C}_{e_x e_y}$ of the form $\frac{\partial r}{\partial y}(x,y) = 0$ will be constant functions.*

Proof:

By definition, the function $r(x,y)$ does not have an xy-term. Therefore, $\frac{\partial r}{\partial y}(x,y)$ does not depend on x. Thus, $\frac{\partial r}{\partial x \partial y}(x,y) = 0$. $\square$

In Algorithm 5.2, we are computing the upper envelope of $r(x, y(x))$, where $(x, y(x))$ is the parametrization of an algebraic arc of degree two, i.e., these curves may contain one square root and polynomials of degree two. Then, the intersection of two arcs could be transformed into a polynomial equation of degree eight, as a consequence they intersect at most eight times. Therefore, the complexity of the upper envelope of m of these arcs is $\mathcal{O}(\lambda_{10}(m))$ and it can be computed in $\mathcal{O}(\lambda_9(m)\log m)$, see Theorem 5.9 in the Appendix 5.6.1.

Furthermore, the arrangement of m planar algebraic curves inside a specified square should be computed during the algorithm, this can be done in $\mathcal{O}(m^2)$ time, see Keyser et al. (2000) for further details. Besides, this arrangement has complexity $\mathcal{O}(m^2)$ as stated in Halperin and Sharir (2017).

Theorem 5.8. *The single facility (MMR-EMCLP) with unknown linear demand realizations can be solved exactly in $\mathcal{O}(m\ \lambda_{10}(m^3))$ time using Algorithm 5.2.*

Proof:

Since $r(x,y) = \sum_{e \in E} r_e(x,y)$ and $r_e(x,y)$ has a constant number of representations as quadratic functions over $x \in e_x$ and $y \in e_y$, for each $e \in E$ (Theorem 5.6), therefore $r(x,y)$ has $\mathcal{O}(m)$ representations as quadratic functions over $x \in e_x, y \in e_y$. In the following, we will see the complexity of computing the expression of $r(x,y)$. The expression of $r_e(x,y)$ is determined by the vertical and horizontal lines induced by the breakpoints of the edge coverage functions and the conditions of Table 5.10, i.e., for each $e \in E$ a constant number of algebraic curves subdivides the square $e_x \times e_y$. Thus, the square $e_x \times e_y$ is divided in $\mathcal{O}(m^2)$ cells by the $\mathcal{O}(m)$ algebraic curves previously obtained for all $e \in E$, this arrangement can be constructed in $\mathcal{O}(m^2)$ time (Keyser et al. (2000)). In a cell, we evaluate an interior point in order to apply Proposition 5.5 and identify (a_e^w, b_e^w), for each $e \in E$, it is done in $\mathcal{O}(m)$ time. If we move to a neighbor cell (a cell that shares an arc boundary) only a constant number of addends changes, then we compute $r(x,y)$ in this new cell in constant time, hence we obtain $r(x,y)$ for all cells in $\mathcal{O}(m^2)$ time just moving by adjacent cells in the square $e_x \times e_y$. Furthermore, Step 3 includes in $\mathcal{C}_{e_x e_y}$ $\mathcal{O}(m^2)$ arcs of cells boundaries and computes the derivative of $\mathcal{O}(m^2)$ functions; it takes $\mathcal{O}(m^2)$ time. Step 4 obtain the upper envelope of $\mathcal{O}(m^3)$

arcs; it takes $\mathcal{O}(\lambda_9(m^3)\log m)$ (Theorem 5.9). In addition, in Step 5, the minimum of an upper envelope is found, the complexity of it is the complexity of the upper envelope; i.e, $\mathcal{O}(\lambda_{10}(m^3))$ (Theorem 5.9). Finally, Step 6 takes constant time.

Therefore, since Step 4 and Step 5 are executed once for each $e_x \in E$; i.e., $\mathcal{O}(m)$ times, the overall complexity of the algorithm is $\mathcal{O}(m\ \lambda_{10}(m^3))$. $\square$

Remark 5.4.1. *The complexity of Algorithm 5.2 is upper bounded by $\mathcal{O}(m^4 \log^* m)$, see Appendix 5.6.1.*

For illustrating the resolution method proposed in Algorithm 5.2, we compute an iteration of the mentioned algorithm to solve (MMR-EMCLP) with unknown linear demand realizations in the network depicted in Figure 5.3.

Example 5.2 (cont.) *In Example 5.2, we computed the expression of the maximal regret of choosing $x \in [1,2]$ over $y \in [2,3]$, i.e., we computed the worst-case demand realization. Now, the objective is to identify for each $x \in [1,2]$ the worst-case alternative for $y \in G$.*

First, we consider the case $y \in [2,3]$. Since $r(x,y)$ is a quadratic function for a fixed $x \in [1,2]$, the $\max_{y\in[2,3]} r(x,y)$ will be found in the boundaries of the cells depicted in Figure 5.11, or in the curve $\frac{\partial r}{\partial y}(x,y) = 0$ in the cells that $r(x,y)$ is concave. In Figure 5.12, these arcs are depicted.

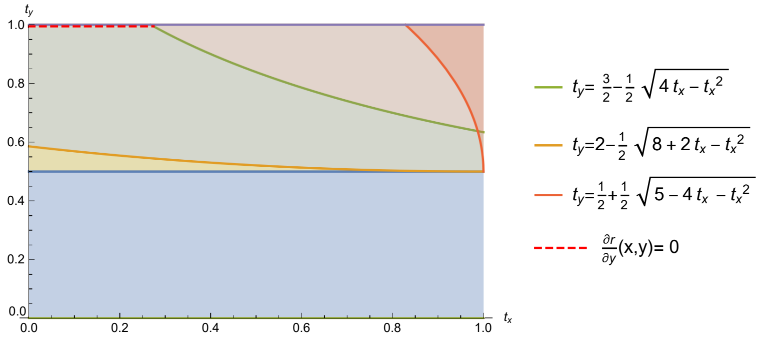


Figure 5.12. *Cells for $r(x,y)$, $x = ([1,2], t_x)$ and $y = ([2,3], t_y)$.*

The following step is to obtain $r(x, y(x))$ of the boundaries of the cells previously defined and the arcs $\frac{\partial r}{\partial y}(x,y) = 0$ whenever $r(x,y(x))$ is concave. In Figure 5.13 is depicted $r(x,y(x))$ of the previous mentioned arcs and the upper envelope is represented as a dotted function in the mentioned figure. Note that in the general algorithm there is no need to compute the upper envelope in this *step, but it has been computed here in order to illustrate the algorithm and simplify the following graphic.*

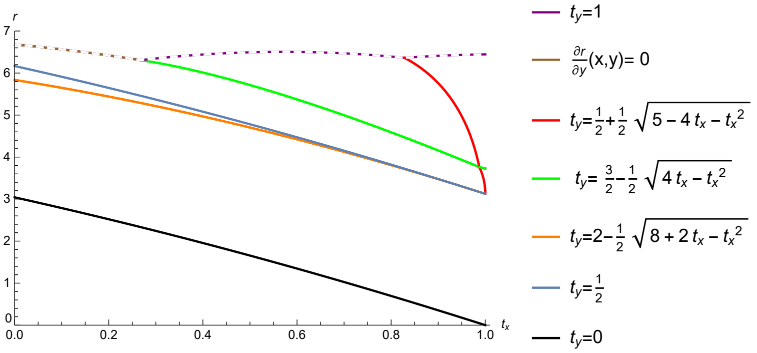


Figure 5.13. $r(x, y(x))$, $x = ([1, 2], t_x)$ *and* $y = ([2, 3], t_y)$.

Observe that for identifying the worst-case alternative the previous procedure should be repeated for each $y \in e_y \in E$, *i.e., obtain the representation of* $r(x, y)$, *derive the subdivision of the square* $e_x \times e_y$, *compute* $r(x, y(x))$ *of the boundaries of the cells, and calculate the upper envelope. The upper envelopes obtained for each* $y \in e_y \in E$ *are depicted in Figure 5.14. It worth mentioning that in the general algorithm, in this step we will compute the upper envelope of all the arcs previously obtained for each* $e_y \in E$ *instead of computing the upper envelope of the upper envelopes obtained for each edge, but in the example we did that in order to simplify the graphics. As can be appreciated, the minimum value of* r *is* 6.4836, *where* $x^*_{[1,2]} = ([1, 2], 0.1572)$; *these values were rounded to four decimal places.*

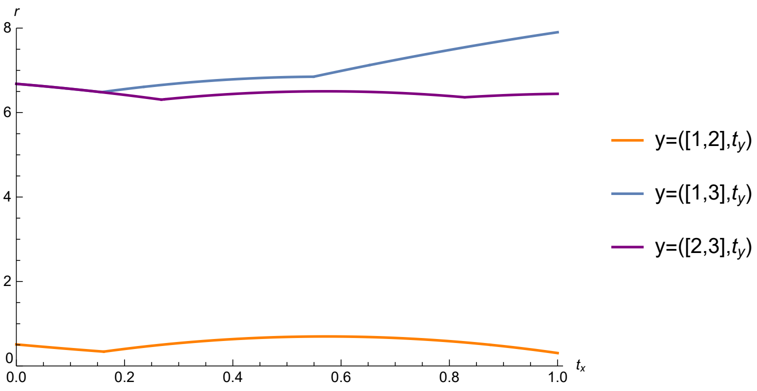


Figure 5.14. *Upper envelope of* $r(x, y(x))$, $x = ([1, 2], t_x)$.

This procedure should be repeated for each $x \in e_x \in E$. The local minima over edge $[2, 3]$ is vertex 2, where the value of r is 7.9023, while the local minima over edge $[1, 3]$ is the point $([1, 3], 0.0533)$, where the value of r is 6.3055; these values were rounded to four decimal places. The optimal location is the x where the minimum value of $r(x^*)$ is found, i.e., the point $([1, 3], 0.0533)$.

To highlight the usefulness of our max-regret approach, we also calculate the optimal location for a deterministic equivalent of the problem. We solve the problem assuming that the demand is known and it is equal to the mean of the upper and lower bound functions over each edge, i.e., $0.5(ub_e + lb_e)$. Following the procedure described in Berman et al. (2016), the optimal solution is vertex 2, resulting in a maximal regret of $\frac{569}{72} \approx 7.9028$. Therefore, the difference between both optimal solutions concerning the max-regret value is significant. The solutions of both models, the maximal regret of them and the covered demand assuming that the demand is deterministic are depicted in Table 5.11.

Model	*Optimal solution*	*Maximal Regret*	*Covered Demand*
(MMR-EMCLP)	$([1, 3], 0.0533)$	6.3055	*10.6858*
Berman et al. (2016)	$([1, 2], 1)$	$\frac{569}{72}$	*12.1250*

Table 5.11. *Solutions for Example 5.2.*

5.5 Concluding remarks

In this chapter, we studied the single-facility Minmax Regret Maximal Covering Location Problem (MMR-EMCLP) on a network where the demand is unknown and distributed along the edges and the facility can be located anywhere on the network. We presented two polynomial time algorithms for solving the cases where the realization demands are unknown constant or linear functions.

As far as we know, this is the first study that applies the minmax regret criterion on a maximal covering location problem on a network where the demand is distributed along the edges. Our results show that the problem is solvable in polynomial time (where the demand realization are constant or linear functions) although the majority of polynomially solvable combinatorial optimization problems become NP-hard in the minmax regret version, as stated in Kouvelis and Yu (1997b).

There are several potential avenues for future research. Firstly, solving the single facility location problem for other kind of demand realization functions. Secondly, considering the multi-facility location version of the problem.

Finally, another interesting open question is how to formulate the problem under a different criterion of coverage as e.g. the gradual covering, the cooperative covering model or the variable radius model, see Berman et al. (2010) for more details of these criteria.

Observe that the results presented in this chapter are published in Baldomero-Naranjo et al. (2021b).

5.6 Appendix

5.6.1 Technical Notes

In this appendix, we include some results about $\lambda_s(n)$, the maximum length of a Davenport-Schinzel sequence of order s on n symbols. We use these results for computing the complexity of the algorithms proposed in the chapter.

Theorem 5.9 (Sharir and Agarwal (1995, Theorem 6.5)). *Given a set of m (unbounded $x-$monotone) Jordan arcs with at most s intersections between any pair of arcs, its lower envelope has an $\mathcal{O}(\lambda_{s+2}(m))$ complexity, and it can be computed in $\mathcal{O}(\lambda_{s+1}(m)\log m)$ time.*

Observe that $\lambda_1(m) = m$, $\lambda_2(m) = 2m-1$, $\lambda_3(m) = \Theta(m\alpha(m))$, and $\lambda_4(m) = \Theta(m2^{\alpha(m)})$, where $\alpha(m)$ is the functional inverse of the Ackermann's function which grows very slowly. However, the problem of estimating $\lambda_s(m)$ for $s > 4$ is more complicated. For any constant s, it is well-known the bound $\lambda_s(m) = \mathcal{O}(m\log^* m)$. Recall that $\log^* m$ is the minimum number of times q such that q consecutive applications of the log operator will map m to a value smaller than 1, i.e., $\overbrace{\log\ldots\log m}^{(q)} \leq 1$. Actually, $\log^* m$ is the smallest height of an exponential "tower" of 2's, $2^{2^{2^{\cdots}}}$ which is $\geq m$ (nothing changes if 2 is replaced by another base $b > 1$). Observe that, $\log^* m$ is much smaller than $\log m$ and it can be considered almost constant for "practical" values of m, see Sharir and Agarwal (1995) for further improvement of these bounds.

5.6.2 Representation of the coverage functions of the examples

This appendix contains the representation of the edge coverage functions and the parts per unit of coverage functions of the network depicted in Figure 5.3. These functions are used in Example 5.1 and 5.2.

Starting with edge $[1,2]$, for $x_1 = ([1,2],t_1)$ the edge coverage functions for all $e \in E$ are given by

$$s^+_{[1,2]}(x_1) = 0, \qquad\qquad s^-_{[1,2]}(x_1) = 1,$$

$$s^+_{[2,3]}(x_1) = \frac{t_1}{2}, \qquad s^-_{[2,3]}(x_1) = 1,$$
$$s^+_{[1,3]}(x_1) = \frac{1-t_1}{3}, \qquad s^-_{[1,3]}(x_1) = 1.$$

Therefore, the parts per unit of coverage functions are:

$$\begin{aligned} c_{[1,2]}(x_1) &= 1, \\ c_{[2,3]}(x_1) &= \frac{t_1}{2}, \\ c_{[1,3]}(x_1) &= \frac{1-t_1}{3}. \end{aligned}$$

Likewise, for $x_2 = ([2,3], t_2)$ the edge coverage functions for all $e \in E$ are given by

$$s^+_{[1,2]}(x_2) = 0, \qquad s^-_{[1,2]}(x_2) = \begin{cases} 2t_2, & \text{if } t_2 \le \frac{1}{2}, \\ 1, & \text{otherwise,} \end{cases}$$

$$s^+_{[2,3]}(x_2) = \begin{cases} \frac{2t_2-1}{2}, & \text{if } t_2 \ge \frac{1}{2}, \\ 0, & \text{otherwise,} \end{cases} \qquad s^-_{[2,3]}(x_2) = \begin{cases} \frac{1+2t_2}{2}, & \text{if } t_2 \le \frac{1}{2}, \\ 1, & \text{otherwise,} \end{cases}$$

$$s^+_{[1,3]}(x_2) = 0, \qquad s^-_{[1,3]}(x_2) = \begin{cases} \frac{4-2t_2}{3}, & \text{if } t_2 \ge \frac{1}{2}, \\ 1, & \text{otherwise.} \end{cases}$$

Thus, the parts per unit of coverage functions are:

$$c_{[1,2]}(x_2) = \begin{cases} 1-2t_2, & \text{if } 0 \le t_2 \le \frac{1}{2}, \\ 0, & \text{if } \frac{1}{2} \le t_2 \le 1, \end{cases}$$

$$c_{[2,3]}(x_2) = \begin{cases} \frac{1}{2}+t_2, & \text{if } 0 \le t_2 \le \frac{1}{2}, \\ \frac{3}{2}-t_2, & \text{if } \frac{1}{2} \le t_2 \le 1, \end{cases}$$

$$c_{[1,3]}(x_2) = \begin{cases} 0, & \text{if } 0 \le t_2 \le \frac{1}{2}, \\ \frac{-1+2t_2}{3}, & \text{if } \frac{1}{2} \le t_2 \le 1. \end{cases}$$

In addition, for $x_3 = ([1,3], t_3)$ the edge coverage functions for all $e \in E$ are given by

$$s^+_{[1,2]}(x_3) = \begin{cases} 1-3t_3, & \text{if } t_3 \le \frac{1}{3}, \\ 0, & \text{otherwise,} \end{cases} \qquad s^-_{[1,2]}(x_3) = 1,$$

$$s^+_{[2,3]}(x_3) = 0, \qquad s^-_{[2,3]}(x_3) = \begin{cases} \frac{4-3t_3}{2}, & \text{if } t_3 \ge \frac{2}{3}, \\ 1, & \text{otherwise,} \end{cases}$$

$$s^+_{[1,3]}(x_2) = \begin{cases} \frac{3t_3-1}{3} & \text{if } t_3 \ge \frac{1}{3}, \\ 0 & \text{otherwise,} \end{cases} \qquad s^-_{[1,3]}(x_3) = \begin{cases} \frac{3t_3+1}{3}, & \text{if } t_3 \le \frac{2}{3}, \\ 1, & \text{otherwise.} \end{cases}$$

Hence, the parts per unit of coverage functions are:

$$c_{[1,2]}(x_3) = \begin{cases} 1 - 3t_3, & \text{if } 0 \leq t_3 \leq \frac{1}{3}, \\ 0, & \text{if } \frac{1}{3} \leq t_3 \leq 1, \end{cases}$$

$$c_{[2,3]}(x_3) = \begin{cases} 0, & \text{if } 0 \leq t_3 \leq \frac{2}{3}, \\ -1 + \frac{3t_3}{2}, & \text{if } \frac{2}{3} \leq t_3 \leq 1, \end{cases}$$

$$c_{[1,3]}(x_3) = \begin{cases} \frac{1}{3} + t_3, & \text{if } 0 \leq t_3 \leq \frac{1}{3}, \\ \frac{2}{3}, & \text{if } \frac{1}{3} \leq t_3 \leq \frac{2}{3}, \\ \frac{4}{3} - t_3, & \text{if } \frac{2}{3} \leq t_3 \leq 1. \end{cases}$$

6 Conclusions

This PhD dissertation has studied four different problems that can be categorized into two main areas: Classification (Chapters 2 and 3) and Location (Chapters 4 and 5). In each chapter, different Mathematical Optimization tools have been designed to analyze and solve the proposed problems. The methodologies and solution approaches followed to address these problems have been developed considering the intrinsic properties of each problem.

First, mixed integer formulations have been derived to represent each of the problems. A deep analysis of the different proposed models has been carried out to provide characterizations of the solution set and to strengthen the initial formulations by including valid inequalities and tightening the values of the big M parameters. The analysis has derived in the development of exact and heuristic approaches to solve each of the formulations. Furthermore, we have exploited the specify properties of the solution space to develop a preprocessing phase to fix some variables and remove several constraints. Besides, we have proved that some particular cases of one of the studied problem are solvable in polynomial time. To find the exact solution of these cases, we have developed polynomial algorithms and provided the upper bounds of their complexity. Finally, all the proposed resolution procedures have been tested on several synthetic and real-world datasets.

At the end of each chapter we have detailed the conclusions for each problem and the lines of future work that arise from them. In the following discussion, this question will be examined from a general perspective.

In this PhD dissertation, classification problems have been approached using SVM. There is extensive literature showing the efficiency of this tool. However, this technique has weaknesses too, a couple of them have been analyzed in this PhD dissertation. In particular, we have addressed the exact resolution of SVM-based models that avoid the excessive influence of outliers

(Chapter 2). We have provided efficient exact solution approaches that outperforms the state-of-the-art methods. Moreover, we have developed a new classifier based on SVM that limits the outliers influence and performs feature selection simultaneously (Chapter 3). Besides introducing a new model, exact and heuristic algorithms have been designed to solve them. After the analysis of these models, there are many questions to continue working on. On the one hand, an interesting line of research is to generalize the presented techniques and models to any ℓ_p-norm and to include in these models kernel functions. On the other hand, it would be interesting to extend our work on classification models to SVM-based regression models, known as Support Vector Regression (SVR). These are only a few of the many possibilities for future work that arise directly from this PhD dissertation, since classification models are widely used in several areas of knowledge and, therefore, the mathematical procedures to solve them are in continuous development.

On the other hand, the maximal covering location problem in networks has been deeply studied. This problem has been solved under different assumptions to describe the reality of diverse applications. In particular, the upgrading version of this problem has been analyzed (Chapter 4). It assumes that the length of the edges of the network can be shorten (within a budget) at the same time that the services are located. Therefore, the solutions obtained are considerably better than those achieved if the decisions were made independently. As far as we know, it is the first time this problem is addressed in the literature. For this reason, several formulations have been proposed to model the problem from various perspectives. We believe that this work could be an encouraging starting point to address the upgrading version of other classical location problems. This is especially interesting since the development of the upgrading version of location problems is a growing area, in which there are many open questions.

Finally, we have addressed an uncertain version of the MCLP where the demand is distributed along the edges and the facilities can be located anywhere on the network (Chapter 5). From a practical point of view, assuming that demand along an edge is known corresponds to an ideal but usually unrealistic scenario, therefore we have treated demands as being unknown. To the best of our knowledge, this is the first time that this version of the MCLP is discussed in the literature. Observe that although the majority of polynomial solvable combinatorial optimization problems become NP-hard in the minmax regret version, we have proved that the problem where the demands of the edges are represented as constant and linear functions is solvable in polynomial time. Furthermore, we have proposed polynomial algorithms to solve these cases. Following this line of research, we believe that it is really interesting to solve this problem for other demands functions as piecewise linear, polynomial, etc. Besides, the applied methodology could be used to solve other location problems with different objective functions.

In conclusion, a wide range of mathematical optimization techniques have been studied and implemented throughout this PhD dissertation to solve open problems in Operations Research. The presented results not only address open questions in the area, but also provide starting points for future lines of research.

Bibliography

B. Adenso-Díaz and F. Rodríguez. A simple search heuristic for the MCLP: Application to the location of ambulance bases in a rural region. *Omega*, 25 (2):181 – 187, 1997.

E. Afrashteh, B. Alizadeh, and F. Baroughi. Optimal approaches for upgrading selective obnoxious p-median location problems on tree networks. *Annals of Operations Research*, 289(2):153–172, 2020.

H. Alazzam, A. Sharieh, and K. E. Sabri. A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. *Expert Systems with Applications*, 148:113249, 2020.

B. Alizadeh and R. Etemad. Linear time optimal approaches for reverse obnoxious center location problems on networks. *Optimization*, 65(11):2025–2036, 2016.

U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750, 1999.

E. Álvarez-Miranda and M. Sinnl. Lagrangian and branch-and-cut approaches for upgrading spanning tree problems. *Computers & Operations Research*, 83:13 – 27, 2017.

E. Angelelli, R. Mansini, and M. G. Speranza. Kernel search: A general heuristic for the multi-dimensional knapsack problem. *Computers & Operations Research*, 37(11):2017 – 2026, 2010.

M. Arana-Jiménez, V. Blanco, and E. Fernández. On the fuzzy maximal covering location problem. *European Journal of Operational Research*, 283 (2):692 – 705, 2020.

T. W. Archibald and S. E. Marshall. Review of mathematical programming applications in water resource management under uncertainty. *Environmental Modeling & Assessment*, 23(6):753–777, 2018.

L. Assunção, T. F. Noronha, A. C. Santos, and R. Andrade. A linear programming based heuristic framework for min-max regret combinatorial optimization problems with interval costs. *Computers & Operations Research*, 81:51 – 66, 2017.

P. Avella, M. Boccia, and I. Vasilyev. Computational experience with general cutting planes for the set covering problem. *Operations Research Letters*, 37 (1):16–20, 2009.

I. Averbakh. On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming*, 90(2):263–272, 2001.

I. Averbakh and O. Berman. Minmax regret median location on a network under uncertainty. *INFORMS Journal on Computing*, 12(2):104–110, 2000.

I. Averbakh and V. Lebedev. Interval data minmax regret network optimization problems. *Discrete Applied Mathematics*, 138(3):289–301, 2004.

I. Averbakh, O. Berman, D. Krass, J. Kalcsics, and S. Nickel. Cooperative covering problems on networks. *Networks*, 63(4):334–349, 2014.

I. Averbakh, O. Berman, and M. Leal. Improved complexity results for the robust mean absolute deviation problem on networks with linear vertex weights. *Discrete Applied Mathematics*, 239:193–199, 2018.

H. Aytug. Feature selection for support vector machines using Generalized Benders Decomposition. *European Journal of Operational Research*, 244(1): 210 – 218, 2015.

M. Baldomero-Naranjo, L. I. Martínez-Merino, and A. M. Rodríguez-Chía. Tightening big Ms in Integer Programming Formulations for Support Vector Machines with Ramp Loss. *European Journal of Operational Research*, 286: 84–100, 2020.

M. Baldomero-Naranjo, J. Kalcsics, A. Marín, and A. M. Rodríguez-Chía. Upgrading edges in the maximal covering location problem. *Manuscript submitted for publication*, 2021a.

M. Baldomero-Naranjo, J. Kalcsics, and A. M. Rodríguez-Chía. Minmax regret maximal covering location problems with edge demands. *Computers & Operations Research*, 130:105181, 2021b.

M. Baldomero-Naranjo, J. Kalcsics, and A. M. Rodríguez-Chía. On the complexity of the upgrading version of the maximal covering location problem. *Manuscript*, 2021c.

M. Baldomero-Naranjo, L. I. Martínez-Merino, and A. M. Rodríguez-Chía. A robust SVM-based approach with feature selection and outliers detection for classification problems. *Expert Systems with Applications*, 178:115017, 2021d.

M. Balinski. Integer programming: methods, uses, computation. *Management Science*, 12:253–313, 1965.

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

B. D. Barkana, I. Saricicek, and B. Yildirim. Performance analysis of descriptive statistical features in retinal vessel segmentation via fuzzy logic, ann, svm, and classifier fusion. *Knowledge-Based Systems*, 118:165–176, 2017.

F. Baroughi Bonab, R. E. Burkard, and E. Gassner. Inverse p-median problems with variable edge lengths. *Mathematical Methods of Operations Research*, 73(2):263–280, 2011.

J. E. Beasley. OR-Library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072, 1990.

P. Belotti, P. Bonami, M. Fischetti, A. Lodi, M. Monaci, A. Nogales-Gómez, and D. Salvagnin. On handling indicator constraints in mixed integer programming. *Computational Optimization and Applications*, 65(3):545–566, 2016.

O. Berman and D. Krass. The generalized maximal covering location problem. *Computers & Operations Research*, 29(6):563 – 581, 2002.

O. Berman and J. Wang. The minmax regret gradual covering location problem on a network with incomplete information of demand weights. *European Journal of Operational Research*, 208(3):233 – 238, 2011.

O. Berman, J. Kalcsics, D. Krass, and S. Nickel. The ordered gradual covering location problem on a network. *Discrete Applied Mathematics*, 157(18): 3689–3707, 2009.

O. Berman, Z. Drezner, and D. Krass. Generalized coverage: New developments in location models. *Computers & Operations Research*, 37:1675–1687, 2010.

O. Berman, J. Kalcsics, and D. Krass. On covering location problems on networks with edge demand. *Computers & Operations Research*, 74:214–227, 2016.

D. P. Bertsekas. *Nonlinear programming.* Athena scientific Belmont, 1999.

V. Blanco and A. Marín. Upgrading nodes in tree-shaped hub location. *Computers & Operations Research*, 102:75–90, 2019.

V. Blanco, A. Japón, and J. Puerto. A mathematical programming approach to binary supervised classification with label noise. *Preprint, arXiv:2004.10170v1*, 2020a.

V. Blanco, A. Japón, and J. Puerto. Optimal arrangements of hyperplanes for SVM-based multiclass classification. *Advances in Data Analysis and Classification*, 14:175–199, 2020b.

V. Blanco, J. Puerto, and A. M. Rodríguez-Chía. On ℓ_p-Support Vector Machines and Multidimensional Kernels. *Journal of Machine Learning Research*, 21(14):1–29, 2020c.

R. Blanquero, E. Carrizosa, and B. G. Tóth. Maximal covering location problems on networks with regional demand. *Omega*, 64:77–85, 2016.

A. Bonn, A. S. L. Rodrigues, and K. J. Gaston. Threatened and endemic species: are they good indicators of patterns of biodiversity on a national scale? *Ecology Letters*, 5(6):733–741, 2002.

J. P. Brooks. Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, 59(2):467–479, 2011.

R. E. Burkard, Y. Lin, and J. Zhang. Weight reduction problems with certain bottleneck objectives. *European Journal of Operational Research*, 153(1): 191–199, 2004a.

R. E. Burkard, C. Pleschiutschnig, and J. Zhang. Inverse median problems. *Discrete Optimization*, 1(1):23–39, 2004b.

R. E. Burkard, E. Gassner, and J. Hatzl. A linear time algorithm for the reverse 1-median problem on a cycle. *Networks*, 48(1):16–23, 2006.

R. E. Burkard, E. Gassner, and J. Hatzl. Reverse 2-median problem on trees. *Discrete Applied Mathematics*, 156(11):1963–1976, 2008.

E. Byvatov and G. Schneider. Support vector machine applications in bioinformatics. *Appl Bioinformatics*, 2(2):67–77, 2003.

E. Carrizosa and D. Romero-Morales. Supervised classification and mathematical optimization. *Computers and Operations Research*, 40(1):150–165, 2013.

E. Carrizosa, A. Nogales-Gómez, and D. Romero-Morales. Heuristic approaches for support vector machines with the ramp loss. *Optimization Letters*, 8(3):1125–1135, 2014.

J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez. A comprehensive survey on support vector machine classification: applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

W. Chen, H. Yang, P. Feng, H. Ding, and H. Lin. idna4mc: identifying dna n4-methylcytosine sites based on nucleotide chemical properties. *Bioinformatics*, 33(22):3518–3523, 2017.

C.-H. Chung. Recent applications of the maximal covering location planning (M.C.L.P.) model. *The Journal of the Operational Research Society*, 37(8): 735–746, 1986.

R. Church. The planar maximal covering location problem. *Journal of Regional Science*, 24(2):185–201, 1984.

R. Church and M. Meadows. Location modeling utilizing maximum service distance criteria. *Geographical Analysis*, 11:358–373, 1979.

R. Church and C. ReVelle. The maximal covering location problem. *Papers of the Regional Science Association*, 3:101–118, 1974.

A. A. Coco, A. C. Santos, and T. F. Noronha. Formulation and algorithms for the robust maximal covering location problem. *Electronic Notes in Discrete Mathematics*, 64:145 – 154, 2018.

E. Conde. Robust minmax regret combinatorial optimization problems with a resource-dependent uncertainty polyhedron of scenarios. *Computers & Operations Research*, 103:97–108, 2019.

E. Conde, M. Leal, and J. Puerto. A minmax regret version of the time-dependent shortest path problem. *European Journal of Operational Research*, 270(3):968 – 981, 2018.

J.-F. Cordeau, F. Furini, and I. Ljubić. Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research*, 275(3):882 – 896, 2019.

C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20 (3):273–297, 1995.

T. Cura. Use of support vector machines with a parallel local search algorithm for data classification and feature selection. *Expert Systems with Applications*, 145:113133, 2020.

J. Current and D. Schilling. Analysis of errors due to demand data aggregation in the set covering and maximal covering location problems. *Geographical Analysis*, 22(2):116–126, 1990.

G. B. Dantzig. Programming in a linear structure. *Econometrica*, 17:73–74, 1949.

M. Daskin. A maximum expected covering location model: Formulation, properties and heuristic solution. *Transportation Science*, 17:48–70, 1983.

M. S. Daskin and L. K. Dean. Location of health care facilities. In M. B. F. Sainfort and W. Pierskalla, editors, *Handbook of OR/MS in Health Care: A Handbook of Methods and Applications*, pages 43–76. Kluwer, 2004.

M. de Hoon, S. Imoto, J. Nolan, and S. Miyano. Open source clustering software. *Bioinformatics*, 20(9):1453–1454, 2004.

D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1-3):161–190, 2002.

I. Demgensky, H. Noltemeier, and H.-C. Wirth. On the flow cost lowering problem. *European Journal of Operational Research*, 137(2):265–271, 2002.

A. P. Duarte Silva. Optimization approaches to supervised classification. *European Journal of Operational Research*, 261(2):772–788, 2017.

G. Dutta and R. Fourer. A survey of mathematical programming applications in integrated steel plants. *Manufacturing & Service Operations Management*, 3(4):387–400, 2001.

M. Ehrgott, Ç. Güler, H. W. Hamacher, and L. Shao. Mathematical optimization in intensity modulated radiation therapy. *Annals of Operations Research*, 175(1):309–365, 2010.

T. Ensor and S. Cooper. Overcoming barriers to health service access: influencing the demand side. *Health Policy Plan*, 19(2):69–79, 2004.

I. Espejo, A. Marín, and A. M. Rodríguez-Chía. Closest assignment constraints in discrete location problems. *European Journal of Operational Research*, 219(1):49–58, 2012.

R. L. Frazier. Aerosol air and duct treatment apparatus for air conditioning and heating systems, 1990. US Patent 4,903,583.

A. Fredriksson. Location-allocation of public services – Citizen access, transparency and measurement. A method and evidence from Brazil and Sweden. *Socio-Economic Planning Sciences*, 59:1–12, 2017.

N. Fröhlich, A. Maier, and H. W. Hamacher. Covering edges in networks. *Networks*, 75(3):278–290, 2020.

T. Furey, N. Cristianini, N. Duffy, D. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

S. García and A. Marín. Covering location problems. In G. Laporte, S. Nickel, and F. Saldanha da Gama, editors, *Location Science*, pages 99–119. Springer International Publishing, 2019.

E. Gassner. Up-and downgrading the 1-median in a network. Technical Report 2007-6, Graz University of Technology, 2007.

E. Gassner. Up-and downgrading the 1-center in a network. *European Journal of Operational Research*, 198(2):370 – 377, 2009.

E. Gassner. An inverse approach to convex ordered median problems in trees. *Journal of Combinatorial Optimization*, 23(2):261–273, 2012.

M. Gaudioso, E. Gorgone, M. Labbé, and A. M. Rodríguez-Chía. Lagrangian relaxation for SVM feature selection. *Computers & Operations Research*, 87:137 – 145, 2017.

M. Gaudioso, E. Gorgone, and J.-B. Hiriart-Urruty. Feature selection in SVM via polyhedral k-norm. *Optimization Letters*, 14:19–36, 2020.

V. Gavrishchaka and S. Banerjee. Support vector machine as an efficient framework for stock market volatility forecasting. *Computational Management Science*, 3(2):147–160, 2006.

B. Ghaddar and J. Naoum-Sawaya. High dimensional data classification and feature selection using support vector machines. *European Journal of Operational Research*, 265(3):993–1004, 2018.

H. Ghoddusi, G. G. Creamer, and N. Rafizadeh. Machine learning in energy economics and finance: A review. *Energy Economics*, 81:709 – 727, 2019.

S. Gollowitzer and I. Ljubić. MIP models for connected facility location: a theoretical and computational study. *Computers & Operations Research*, 38:435–449, 2011.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439): 531–537, 1999.

L. Gonzalez-Abril, F. Velasco, J. Ortega, and L. Franco. Support vector machines for classification of input vectors with different metrics. *Computers & Mathematics with Applications*, 61(9):2874–2878, 2011.

G. Guastaroba and M. G. Speranza. Kernel search for the capacitated facility location problem. *Journal of Heuristics*, 18(6):877–917, 2012.

G. Guastaroba, M. Savelsbergh, and M. Speranza. Adaptive kernel search: A heuristic for solving mixed integer linear programs. *European Journal of Operational Research*, 263(3):789 – 804, 2017.

J. Guerrero, G. Pajares, M. Montalvo, J. Romeo, and M. Guijarro. Support vector machines for crop/weeds identification in maize fields. *Expert Systems with Applications*, 39(12):11149 – 11155, 2012.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46(1): 389–422, 2002.

I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, editors. *Feature extraction: foundations and applications*, volume 207 of *Series Studies in Fuzziness and Soft Computing*. Springer-Verlag Berlin Heidelberg, 2006.

S. Hakimi. Optimal location of switching centers and the absolute centers and medians of a graph. *Operations Research*, 12:450–459, 1964.

S. Hakimi. Optimal distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research*, 13:462–475, 1965.

D. Halperin and M. Sharir. Arrangements. In J. E. Goodman, J. O'Rourke, and C. D. Tóth, editors, *Handbook of discrete and computational geometry, third edition*, chapter 28, pages 723–762. Chapman and Hall/CRC, 2017.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer, 2009.

C. Heuberger. Inverse combinatorial optimization: A survey on problems, methods, and results. *Journal of Combinatorial Optimization*, 8(3):329–361, 2004.

D. S. Hochbaum, editor. *Approximation Algorithms for NP-hard Problems*. PWS Publishing Co., Boston, MA, USA, 1997. ISBN 0-534-94968-1.

X. Huang, L. Shi, and J. A. Suykens. Ramp loss linear programming support vector machine. *Journal of Machine Learning Research*, 15:2185–2211, 2014.

M. B. Jacoby and M. Holman. Managing medical bills on the brink of bankruptcy. *Yale journal of health policy, law, and ethics*, 10(2):239–289, 291–297, 2010.

A. Jiménez-Cordero, J. M. Morales, and S. Pineda. A novel embedded min-max approach for feature selection in nonlinear support vector machine classification. *European Journal of Operation Research*, 293(1):24–35, 2021.

J. H. Joloudari, H. Saadatfar, A. Dehzangi, and S. Shamshirband. Computer-aided decision-making for predicting liver disease using pso-based optimized svm with feature selection. *Informatics in Medicine Unlocked*, 17:100255, 2019.

J. Kalcsics, S. Nickel, J. Puerto, and A. Tamir. Algorithmic results for ordered median problems defined on networks and the plane. *Operations Research Letters*, 30:149–158, 2002.

J. Kalcsics, S. Nickel, and J. Puerto. Multi-facility ordered median problems: A further analysis. *Networks*, 41(1):1–12, 2003.

J. Kallrath. Mixed integer optimization in the chemical process industry: Experience, potential and future perspectives. *Chemical Engineering Research and Design*, 78(6):809–822, 2000.

L. V. Kantorovich. Mathematical methods of organizing and planning production. *Management science*, 6(4):366–422, 1960.

B. Y. Kara and M.-E. Rancourt. Location problems in humanitarian supply chains. In G. Laporte, S. Nickel, and F. Saldanha da Gama, editors, *Location Science*, pages 611–629. Springer International Publishing, 2019.

M. Karatas and L. Eriskin. The minimal covering location and sizing problem in the presence of gradual cooperative coverage. *European Journal of Operational Research*, 295(3):838–856, 2021.

J. Keyser, T. Culver, D. Manocha, and S. Krishnan. Efficient and exact manipulation of algebraic points and curves. *Computer-Aided Design*, 32(11): 649 – 662, 2000.

C. Kim, F. J. Costello, and K. C. Lee. Integrating qualitative comparative analysis and support vector machine methods to reduce passengers' resistance to biometric e-gates for sustainable airport operations. *Sustainability*, 11(19):5349, 2019.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

P. Kouvelis and G. Yu. Robust 1-median location problems: Dynamic aspects and uncertainty. In P. Kouvelis and G. Yu, editors, *Robust Discrete Optimization and Its Applications. Nonconvex Optimization and Its Applications*, volume 14. Springer, Boston, MA, 1997a.

P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications. Nonconvex Optimization and Its Applications*, volume 14. Springer, Boston, MA, 1997b.

P. Kouvelis, G. L. Vairaktarakis, and G. Yu. *Robust 1-median location on a tree in the presence of demand and transportation cost uncertainty.* Department of Industrial & Systems Engineering, University of Florida, 1993.

G. Kunapuli, K. Bennett, J. Hu, and J.-S. Pang. Classification model selection via bilevel programming. *Optimization Methods and Software*, 23(4):475–489, 2008.

M. Labbé, L. I. Martínez-Merino, and A. M. Rodríguez-Chía. Mixed Integer Linear Programming for Feature Selection in Support Vector Machine. *Discrete Applied Mathematics*, 261:276–304, 2019.

G. Laporte, S. Nickel, and F. Saldanha da Gama. Introduction to location science. In G. Laporte, S. Nickel, and F. Saldanha da Gama, editors, *Location Science*, pages 1–21. Springer International Publishing, 2019.

E. K. Lee and T.-L. Wu. *Classification and Disease Prediction Via Mathematical Programming*, pages 1–50. Springer US, Boston, MA, 2009.

I. G. Lee, Q. Zhang, S. W. Yoon, and D. Won. A mixed integer linear programming support vector machine for cost-effective feature selection. *Knowledge-Based Systems*, 203:106145, 2020.

X. Li, Z. Zhao, X. Zhu, and T. Wyatt. Covering models and optimization techniques for emergency response facility location and planning: A review. *Mathematical Methods of Operations Research*, 74(3):281–310, 2011.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Y. Liu, H. Helen Zhang, C. Park, and J. Ahn. Support vector machines with adaptive lq penalty. *Computational Statistics & Data Analysis*, 51(12):6380–6394, 2007.

M. C. López-de-los-Mozos, J. Puerto, and A. M. Rodríguez-Chía. Robust mean absolute deviation problems on networks with linear vertex weights. *Networks*, 61(1):76–85, 2013.

R. Love, J. Morris, and G. Wesolowsky. *Facilities Location: Models & Methods.* North Holland Publishing Company, New York, 1988.

H. Lukashevich, S. Nowak, and P. Dunker. Using one-class svm outliers detection for verification of collaboratively tagged image training sets. In *2009*

IEEE International Conference on Multimedia and Expo, pages 682–685, 2009.

S. Maldonado, J. Pérez, R. Weber, and M. Labbé. Feature selection for Support Vector Machines via Mixed Integer Linear Programming. *Information Sciences*, 279:163–175, 2014.

S. Maldonado, J. Pérez, and C. Bravo. Cost-based feature selection for support vector machines: An application in credit scoring. *European Journal of Operational Research*, 261(2):656 – 665, 2017.

S. Maldonado, J. López, A. Jiménez-Molina, and H. Lira. Simultaneous feature selection and heterogeneity control for svm classification: An application to mental workload assessment. *Expert Systems with Applications*, 143:112988, 2020.

O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13(3):444–452, 1965.

O. L. Mangasarian. Multisurface method of pattern separation. *IEEE Transactions on Information Theory*, 14(6):801–807, 1968.

O. L. Mangasarian. Mathematical programming in data mining. *Data mining and knowledge discovery*, 1(2):183–201, 1997.

A. Manne. Plant location under economics of scale-decentralization and computation. *Management Science*, 11:213–235, 1964.

A. Marín, L. I. Martínez-Merino, A. M. Rodríguez-Chía, and F. Saldanha-da Gama. Multi-period stochastic covering location problems: Modeling framework and solution approach. *European Journal of Operational Research*, 268 (2):432–449, 2018.

M. Melo, S. Nickel, and F. Saldanha da Gama. Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning. *Computers & Operations Research*, 33:181–208, 2006.

J. Min and Y.-C. Lee. Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications*, 28(4):603–614, 2005.

M. Mrówczyńska, M. Sztubecka, M. Skiba, A. Bazan-Krzywoszańska, and P. Bejga. The use of artificial intelligence as a tool supporting sustainable development local policy. *Sustainability*, 11(15):4199, 2019.

Muren, H. Li, S. K. Mukhopadhyay, J.-J. Wu, L. Zhou, and Z. Du. Balanced maximal covering location problem and its application in bike-sharing. *International Journal of Production Economics*, 223:107513, 2020.

A. T. Murray and D. Tong. Coverage optimization in continuous space facility siting. *International Journal of Geographical Information Science*, 21(7): 757–776, 2007.

A. T. Murray, D. Tong, and K. Kim. Enhancing classic coverage location models. *International Regional Science Review*, 33(2):115–133, 2010.

J. v. Neumann. Discussion of a Maximization problem. *Institute for Advanced Study. Manuscript*, 1947.

K. T. Nguyen and A. R. Sepasian. The inverse 1-center problem on trees with variable edge lengths under Chebyshev norm and Hamming distance. *Journal of Combinatorial Optimization*, 32(3):872–884, 2016.

M. H. Nguyen and F. de la Torre. Optimal feature selection for support vector machines. *Pattern Recognition*, 43:584–591, 2010.

S. Nickel and J. Puerto. A unified approach to network location problems. *Networks*, 34:283–290, 1999.

R. D. Norton and P. B. Hazell. *Mathematical programming for economic analysis in agriculture*. Macmillan, 1986.

D. Paik and S. Sahni. Network upgrading problems. *Networks*, 26(1):45–58, 1995.

F. Plastria. Continuous covering location problems. In Z. Drezner and H. Hamacher, editors, *Facility Location: Applications and Theory*, pages 37–79. Springer, Berlin, 2002.

F. Plastria. Up- and downgrading the Euclidean 1-median problem and knapsack Voronoi diagrams. *Annals of Operations Research*, 246:227–251, 2016.

F. Plastria and E. Carrizosa. Gauge distances and median hyperplanes. *Journal of Optimization Theory and Applications*, 110(1):173–182, 2001.

F. Plastria and E. Carrizosa. Minmax-distance approximation and separation problems: geometrical properties. *Mathematical programming*, 132(1):153–177, 2012.

D. Prokhorov. Ijcnn 2001 neural network competition. *Slide presentation in IJCNN*, 1:97, 2001.

J. Puerto and A. M. Rodríguez-Chía. Modelos de la localización continua. *Boletín de la Sociedad Española de Matemática Aplicada*, 29:89–132, 2004.

J. Puerto, A. M. Rodríguez-Chía, and A. Tamir. Minimax regret single-facility ordered median location problems on networks. *INFORMS Journal on Computing*, 21(1):77–87, 2009.

A. Rais and A. Viana. Operations research in healthcare: a survey. *International Transactions in Operational Research*, 18(1):1–31, 2011.

T. U. Rehman, M. S. Mahmud, Y. K. Chang, J. Jin, and J. Shin. Current and future applications of statistical machine learning algorithms for agricultural machine vision systems. *Computers and Electronics in Agriculture*, 156:585 – 605, 2019.

J. Ren. Ann vs. svm: Which one performs better in classification of mccs in mammogram imaging. *Knowledge-Based Systems*, 26:144–153, 2012.

C. ReVelle, C. Toregas, and L. Falkson. Applications of the location set-covering problem. *Geographical Analysis*, 8(1):65–76, 1976.

C. ReVelle, M. Scholssberg, and J. Williams. Solving the maximal covering location problem with heuristic concentration. *Computers & Operations Research*, 35(2):427–435, 2008.

C. S. ReVelle and R. W. Swain. Central facilities location. *Geographical Analysis*, 2(1):30–42, 1970.

C. M. Salgado, C. Azevedo, H. Proença, and S. M. Vieira. Noise versus outliers. In *Secondary Analysis of Electronic Health Records*, pages 163–183. Springer International Publishing, 2016.

L. J. Savage. The theory of statistical decision. *Journal of the American Statistical Association*, 46(253):55–67, 1951.

M. P. Scaparra and R. L. Church. Location Problems Under Disaster Events. In G. Laporte, S. Nickel, and F. Saldanha da Gama, editors, *Location Science*, pages 631–656. Springer International Publishing, 2019.

A. R. Sepasian. Upgrading the 1-center problem with edge length variables on a tree. *Discrete Optimization*, 29:1–17, 2018.

A. R. Sepasian and F. Rahbarnia. Upgrading p-median problem on a path. *Journal of Mathematical Modelling and Algorithms in Operations Research*, 14(2):145–157, 2015.

M. Shakhsi-Niaei, S. H. Iranmanesh, and S. A. Torabi. A review of mathematical optimization applications in oil-and-gas upstream & midstream management. *International Journal of Energy and Statistics*, 1(02):143–154, 2013.

M. Sharir and P. K. Agarwal. *Davenport-Schinzel sequences and their geometric applications.* Cambridge University Press, 1995.

M. A. Shipp, K. N. Ross, P. Tamayo, A. P. Weng, J. L. Kutok, R. C. Aguiar, M. Gaasenbeek, M. Angelo, M. Reich, G. S. Pinkus, T. S. Ray, M. A. Koval, K. W. Last, A. Norton, T. A. Lister, J. Mesirov, D. S. Neuberg, E. S. Lander, J. C. Aster, and T. R. Golub. Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature medicine*, 8(1):68–74, 2002.

F. Silva and D. Serra. Locating emergency services with different priorities: the priority queuing covering location problem. *Journal of the Operational Research Society*, 59(9):1229–1238, 2008.

C. Toregas, R. Swain, C. ReVelle, and L. Bergman. The location of emergency service facilities. *Operations Research*, 19(6):1363–1373, 1971.

V. Vapnik. *Statistical Learning Theory.* Wiley, 1 edition, 1998.

A. K. Vatsa and S. Jayaswal. Capacitated multi-period maximal covering location problem with server uncertainty. *European Journal of Operational Research*, 289(3):1107–1126, 2021.

H. Wang, B. Zheng, S. W. Yoon, and H. S. Ko. A support vector machine-based ensemble algorithm for breast cancer diagnosis. *European Journal of Operational Research*, 267(2):687 – 699, 2018.

Q. Wang and Y. Bai. An efficient algorithm for reverse 2-median problem on a cycle. *Journal of Networks*, 5(10):1169–1176, 2010.

P. D. Wright, M. J. Liberatore, and R. L. Nydick. A survey of Operations Research Models and Applications in Homeland Security. *Interfaces*, 36(6): 514–529, 2006.

L. Wu, J. Lee, J. Zhang, and Q. Wang. The inverse 1-median problem on tree networks with variable real edge lengths. *Mathematical Problems in Engineering*, 2013:313868, 2013.

G. Xu, Z. Cao, B.-G. Hu, and J. C. Principe. Robust support vector machines based on the rescaled hinge loss function. *Pattern Recognition*, 63:139 – 148, 2017.

P. Yang, Y. Xiao, Y. Zhang, S. Zhou, J. Yang, and Y. Xu. The continuous maximal covering location problem in large-scale natural disaster rescue scenes. *Computers & Industrial Engineering*, page 106608, 2020.

X. Yang and J. Zhang. Inverse center location problem on a tree. *Journal of Systems Science and Complexity*, 21(4):643–656, 2008.

J. Żak and S. Węgliński. The selection of the logistics center location based on mcdm/a methodology. *Transportation Research Procedia*, 3:555–564, 2014. 17th Meeting of the EURO Working Group on Transportation, EWGT2014, 2-4 July 2014, Sevilla, Spain.

H. Zhang, J. Ahn, X. Lin, and C. Park. Gene selection using support vector machines with non-convex penalty. *Bioinformatics*, 22(1):88–95, 2006.

J. Zhang, X. Yang, and M.-C. Cai. Reverse center location problem. In *Algorithms and Computation*, pages 279–294. Springer, 1999.

Q. Zhang, X. Guan, and P. M. Pardalos. Maximum shortest path interdiction problem by upgrading edges on trees under weighted l_1 norm. *Journal of Global Optimization*, 79(4):959–987, 2021.

W. Zhang, T. Yoshida, and X. Tang. Text classification based on multi-word with support vector machine. *Knowledge-Based Systems*, 21(8):879–886, 2008.

Y. Zhang, N. Meratnia, and P. Havinga. Hyperellipsoidal svm-based outlier detection technique for geosensor networks. In N. Trigoni, A. Markham, and S. Nawaz, editors, *GeoSensor Networks*, pages 31–41, Berlin, Heidelberg,

2009. Springer Berlin Heidelberg.

List of Figures

List of Tables