Optimising the Distribution of Modular Capacitated Services in Smart Cities

Bert Veerman

(2537595)

July 11, 2019

Thesis MSc EOR - Operations Research

Supervisors:

Dr. A.A.N. Ridder *(Vrije Universiteit)*

Dr. F. Phillipson *(TNO)*

Abstract

This thesis focuses on the problem of distributing services over urban areas to satisfy the demand, which is also called the Multi-Service Capacitated Facility Location Problem (MSCFLP). In the smart city context, the demand is spread out over the city. Costs savings can be obtained by combining multiple services. Three different heuristics are proposed: the Extended Pricing Heuristic (EPH), the Extended Linear Relaxation Heuristic (ELRH) and the Extended Sequential Covering Heuristic (ESCH). The heuristics consist of two phases. A feasible solution is found in the first phase, which is optimised by an exact method in the second phase. The exact method is also executed on the whole problem to benchmark the heuristics. In addition, the Multi-Service Modular Capacitated Facility Location Problem (MSMCFLP) is introduced. This problem allows modular capacities which generally leads to large costs benefits. The heuristics and the exact method are performed on nine test instances of both problems. The heuristics can be described as efficient, since good solutions are found in short computation times. Regarding the MSCFLP, the exact method has the best performance, closely followed by the ESCH, EPH and ELRH. Concerning the MSMCFLP, the ESCH even outperforms the exact method due to the stopping criteria. Finally, a lower bound analysis showed that only small improvements can be made by new heuristics in both problems.

Keywords: Smart City Infrastructure Planning, Facility Location, Multi-Service, Modular Capacities.

List of Abbreviations

BILP Binary Integer Linear Programming.

CFLP Capacitated Facility Location Problem.

CH Covering Heuristic.

ELRH Extended Linear Relaxation Heuristic.

EPH Extended Pricing Heuristic.

ESCH Extended Sequential Covering Heuristic.

FLP Facility Location Problem.

ILP Integer Linear Programming.

LP Linear Programming.

MSCFLP Multi-Service Capacitated Facility Location Problem.

MSLSCP Multi-Service Location Set Cover Problem.

MSMCFLP Multi-Service Modular Capacitated Facility Location Problem.

SCH Sequential Covering Heuristic.

SCP Set Covering Problem.

SSCFLP Single-Source Capacitated Facility Location Problem.

SVC Smart Vehicle Communication.

UFLP Uncapacitated Facility Location Problem.

WNP Wireless Network Problem.

Contents

1 Introduction **1**

2 Problem Formulation **3**
2.1 Multi-Service Capacitated Facility Location Problem 3
2.2 Multi-Service Modular Capacitated Facility Location Problem 7
2.3 Multi-Service Location Set Covering Problem 8

3 Literature Review **9**

4 Experimental Design **11**
4.1 Locations 11
4.2 Services 11
4.3 Demand Points 12
4.4 Test Instances 13
4.5 Software and Hardware 13

5 Solution Approach **14**
5.1 Exact Method 14
5.2 Extended Pricing Heuristic 19
5.3 Extended Linear Relaxation Heuristic 20
5.4 Extended Sequential Covering Heuristic 22

6 Computational Results **24**
6.1 MSCFLP Results 24
6.2 MSMCFLP Results - Solution Methods 26
6.3 MSMCFLP Results - Modular Capacities 30
6.4 Lower Bound Analysis 31
6.5 Sensitivity Analysis 33

7 Conclusion **36**

8 Discussion **37**
8.1 Limitations 37
8.2 Suggestions for Further Research 38

A Tables of Computational Results **43**
A.1 MSCFLP 43
A.2 MSMCFLP 44

B Maps of Test Instances **47**

1 Introduction

Worldwide, more and more people move to cities [32]. This trend of urbanisation leads to problems ranging from food security to traffic jams and from air pollution to terrorism. The United Nations emphasises these problems resulting in one of the seventeen Sustainable Development Goals [42] as can be seen in Figure 1. This goal is described as 'make cities and human settlements inclusive, safe, resilient and sustainable'. The government is needed to steer this development into the right direction. For this, data collection is important to identify which problems occur in which areas and under what circumstances. Technological services can be implemented for accurate measurements to prevent these problems from happening. The development of implementing technologies in urban areas is also known as 'smart cities'. The framework that can be used for communication in an intelligent infrastructure is called Internet of Things. According to Zanella et al. [45], a wide variety of devices such as home appliances, vehicles, surveillance cameras and monitoring sensors can be used to provide information to the government, companies and citizens. An efficient distribution of these devices is needed to gather as many data as possible against the lowest costs. In this thesis, it is assumed that one party, say the government, manages these devices to provide the so-called services in the smart city context. The devices are distributed over a set of potential spots such that the demand for the services is covered. The demand for the services can be based on the places where a certain service needs to be provided or data need to be collected. For example, imagine that the municipality of Amsterdam wants to offer Wi-Fi in the city centre. For this, the demand needs to be specified. On some places like squares or stations, a decent Wi-Fi connection is needed. Other places may have less priority, like parks and canals. The distribution of routers over the city centre has to be done in such a way that the demand is covered, while the costs of placing the routers are minimised. Now, imagine the multi-service context in which multiple services need to be provided. Examples of other services are air quality meters, traffic monitoring devices, motion detectors and air alarms. The services have different properties such as range and capacity. For example, many motion detectors should be placed to track every movement on a square. On the other hand, only few air alarms need to be placed to provide alarm signals to an entire city. Costs savings can be obtained by placing devices of different services at the same spot. In addition, there are restrictions on the capacity of the devices, e.g. Wi-Fi routers have a certain bandwidth to serve a limited amount of internet traffic. The problem is in literature also known as the Multi-Service Capacitated Facility Location Problem (MSCFLP) as introduced by Hoekstra [27]. The framework of the problem is to minimise the costs of placing service-providing devices while satisfying restrictions on capacity and range. Therefore, this problem can be used in different applications. An example is the distribution of hospitals over a country [3]. The costs have to be minimised while being able to reach every location within a certain time slot. In addition, the number of beds needs to be in line with the population living in the surrounding area. Other applications of this problem exist in the fields of telecommunication, physics, engineering and economics.

The Dutch organisation for applied scientific research, also known as TNO, conducts research on the subject of smart cities from different perspectives. An example is the development of 5G, which makes the study on smart cities even more interesting. This thesis focuses on the mathematical problem of distributing devices over a city to provide services. Three students have written their thesis before at TNO regarding this subject. Vos [44] studied the uncapacitated version, which focuses on the coverage of the demand without capacity restriction. This is also called the Multi-Service Location Set Cover Problem (MSLSCP). Verhoek [43] extended this by taking into account the capacity restriction and stochastic demand, which means that the future demand is probabilistic. This may be the case for Wi-Fi, since internet traffic is usually predictable but not constant. Hoekstra [27] also included the capacity restriction, this time with fixed demand. The MSCFLP was solved by sequentially distributing the services over the locations by a series of Integer Linear Programming (ILP) problems. Furthermore, she introduced the MSCFLP with Partial Covering in which not all demand had to be covered. Both analysis gave good results, but the computation times remained considerably high.

In this thesis, several heuristics are made for the MSCFLP. Currently, only heuristics exist that sequentially use exact methods, which still results in long computation times. Heuristics are meant to obtain solutions which are not necessarily optimal, but sufficient for reaching their goal in short computation times. The latter is an important property, since the problems are generally large in the smart city context. More precisely, the heuristics will be applied to districts or even entire cities in practice. The proposed heuristics are compared by their computation times and objective values on test instances of different sizes. Usually, there is a trade-off between these two objectives. The performance of the heuristics on the small instances can be compared to the optimal solutions, which can be calculated by exact methods. The performance on the large instances can be compared to the lower bounds provided by the exact methods, since these instances cannot be solved optimally in a reasonable amount of time. The heuristics are based on two approaches. The first approach sequentially executes an algorithm for every service. The second approach executes an algorithm simultaneously for all services. Besides, the Multi-Service Modular Capacitated Facility Location Problem (MSMCFLP) is introduced in this thesis. This problem extends the MSCFLP by allowing modular capacities. In other words, the capacity of services can be increased in discrete steps, which is expected to result in costs benefits. Furthermore, this makes the problem more realistic, since this option usually exists for many services.

This thesis entails the following three contributions to the current research. First of all, the heuristics obtain better solutions to large instances than currently available. The methods used before obtained moderate solutions, even by allowing long computation times. Secondly, the solutions of the heuristics are obtained in shorter computation times. This speed up is interesting due to the large instances in the smart city context. Finally, this thesis introduces the MSMCFLP, which is a relevant extension of the MSCFLP. The heuristics are made in such a way that they can be used for both problems.

This thesis is structured in the following way: Section 2 presents the problem formulations, together with some improvements. Thereafter, the literature review is given in Section 3 and the experimental design is addressed in Section 4. In Section 5, the solution approach is given, which consists of the exact method and the heuristics. The computational results are discussed in Section 6. Finally, the conclusion and discussion are given in Section 7 and 8, respectively.

Figure 1: Sustainable Development Goals defined by the United Nations.

2 Problem Formulation

This section gives an overview of the mathematical formulation of the MSCFLP as in Hoekstra [27]. Besides, the formulation of the newly introduced MSMCFLP is given. Finally, the MSLSCP as in Vos [44] is addressed, since it is used in a heuristic.

2.1 Multi-Service Capacitated Facility Location Problem

First of all, some definitions are needed to accurately state the problem in a mathematical way.

- A demand point requires a certain quantity of one service and is given by coordinates. For example, a home address where Wi-Fi connection of a certain bandwidth is needed.
- A location is a potential spot where services could be provided from. A location is given by coordinates and can for instance be a lamppost.
- An access location is a location that is opened to equip with one or more services. For example, a lamppost that is connected by fiber and has an electricity connection.
- A service access point is a device providing a certain service and can only be placed on access locations. This can for instance be a Wi-Fi router.

The demand points represent the geographical locations where a quantity of a certain service is required. In contrast to Verhoek [43], the demand is taken fixed which means that it is constant over time. Every demand point requires only one service. The demand points can be served by service access points which need to be distributed over the locations. The service access points have a specific capacity, range and costs. Cost benefits can be gained by equipping access locations with multiple service access points of different services, since a location only needs to be opened once. Some mathematical definitions are needed to formulate the problem. Let $\mathcal{L}$ be the set of locations, $\mathcal{F}$ be the set of services and $\mathcal{G}^u$ be the set of demand points for service $u \in \mathcal{F}$. Remark that the sets of demand points for every service are disjoint which is stated mathematically in Equation 1.

$$\mathcal{G}^{u_1} \cap \mathcal{G}^{u_2} = \emptyset, \qquad \forall u_1, u_2 \in \mathcal{F} \text{ with } u_1 \neq u_2. \tag{1}$$

Furthermore, let $\mathcal{G}$ be the set of all demand points, i.e. the union of the sets of the demand points for every service. This is stated mathematically in Equation 2.

$$\mathcal{G} := \bigcup_{u \in \mathcal{F}} \mathcal{G}^u. \tag{2}$$

The parameters of the problem are defined in Table 1.

Table 1: Parameters of the MSCFLP.

Parameter	Range	Description
a_{ij}^u	$\{0,1\}$	$\begin{cases} 1 \text{ if demand point } i \in \mathcal{G}^u \text{ can be served from location } j \in \mathcal{L} \text{ for service } u \in \mathcal{F} \\ 0 \text{ otherwise} \end{cases}$
η^u	$\mathbb{N}^+$	capacity of a service access point of service $u \in \mathcal{F}$
c_j^u	$[0,\infty)$	equipping costs of a service access point of service $u \in \mathcal{F}$ on location $j \in \mathcal{L}$
f_j	$[0,\infty)$	opening costs of location $j \in \mathcal{L}$
d_i^u	$\mathbb{N}^+$	demand of demand point $i \in \mathcal{G}^u$ for service $u \in \mathcal{F}$

The coverage elements a_{ij}^u indicate which demand points can be served from which locations for every service. In this formulation, it is left open how this is employed. An option is to compare the distance between demand points and locations with the range of the corresponding service. Furthermore, it is chosen to take the capacity η^u of a service access point of service u equal for all locations. On the contrary, the equipping and opening costs may vary between locations. The opening costs can be seen as the preparation costs of a location to enable it to equip with service access points of different services. The equipping costs are the costs that have to be paid additionally to provide a specific service from that location. One can imagine that these costs may differ between locations. The demand of a demand point can be expressed in a quantity d_i^u, which is restricted to be a positive integer. Naturally, a demand point can be removed if its demand is equal to zero.

In this problem, three different types of decisions have to be made.

- The selection of access locations.
- The distribution of service access points over the set of access locations.
- The allocation of demand points to service access points.

These decisions can be translated mathematically to decision variables which are presented in Table 2.

Table 2: Decision variables of the MSCFLP.

Variable	**Range**	**Description**
y_j	$\{0,1\}$	$\begin{cases} 1 \text{ if location } j \in \mathcal{L} \text{ is an access location} \\ 0 \text{ otherwise} \end{cases}$
x_j^u	$\{0,1\}$	$\begin{cases} 1 \text{ if location } j \in \mathcal{L} \text{ is equipped with a service access point of service } u \in \mathcal{F} \\ 0 \text{ otherwise} \end{cases}$
s_{ij}^u	$\{0,1,\ldots,d_u^i\}$	quantity of service $u \in \mathcal{F}$ from location $j \in \mathcal{L}$ to demand point $i \in \mathcal{G}^u$

Now, the problem can be expressed as an ILP problem using the parameters and decision variables. This can be seen in Equation 3.

$$\text{Min} \sum_{u\in\mathcal{F}} \sum_{j\in\mathcal{L}} c_j^u x_j^u + \sum_{j\in\mathcal{L}} f_j y_j \tag{3a}$$

subject to

$$x_j^u \leq y_j, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{3b}$$

$$\sum_{i\in\mathcal{G}^u} s_{ij}^u \leq \eta^u x_j^u, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{3c}$$

$$\sum_{j\in\mathcal{L}} s_{ij}^u \geq d_i^u, \qquad \forall i \in \mathcal{G}^u, \forall u \in \mathcal{F}, \tag{3d}$$

$$s_{ij}^u \leq a_{ij}^u M, \qquad \forall i \in \mathcal{G}^u, \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{3e}$$

$$s_{ij}^u \in \mathbb{N}, \qquad \forall i \in \mathcal{G}^u, \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{3f}$$

$$x_j^u \in \{0,1\}, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{3g}$$

$$y_j \in \{0,1\}, \qquad \forall j \in \mathcal{L}, \tag{3h}$$

where M is a number larger than the highest demand $\max\limits_{u \in \mathcal{F}, i \in \mathcal{G}^u} d_i^u$.

The goal is to minimise the total costs (Equation 3a), which consists of the costs regarding the opening and equipping of locations. Constraint 3b ensures that only access locations can be equipped with service access points. The capacity constraint is forced by Constraint 3c. Constraint 3d ensures that all demand is covered. Constraint 3e restricts that service access points can only serve demand points in their corresponding range. The solution space is given by Constraint 3f, 3g and 3h.

The decision variables y_j and x_j^u are binary, whereas the decision variables s_{ij}^u are integer. There are $|\mathcal{L}|(|\mathcal{F}|+1)$ binary variables and $|\mathcal{G}||\mathcal{L}|$ integer variables. The number of constraints is equal to $2|\mathcal{L}||\mathcal{F}| + |\mathcal{G}|(|\mathcal{L}|+1)$.

2.1.1 Alternative Formulation

The formulation above describes the problem well, but consists of a large and symmetric solution space. Symmetric means that there are many feasible solutions with the same objective value. This makes it hard for exact methods to find the global optimum as there is no clear convergence direction. Jeroslow [29] showed that even trivial ILP problems can be unsolvable by exact methods if the symmetry is not removed. Generally, exact methods are more efficient on problems with small and asymmetric solution spaces. Therefore, four improvements are listed below to make the problem more suitable for exact methods.

1. The solution space can be reduced by removing the decision variables s_{ij}^u for which $a_{ij} = 0$. Namely, a service access point cannot serve demand points outside its range. In this way, $\mathcal{L}_i^u$ can be defined as the set of locations for which demand point i requiring service u is within the range. Mathematically, this is stated in Equation 4.

$$\mathcal{L}_i^u := \{j \in \mathcal{L} | a_{ij}^u = 1\}. \tag{4}$$

   Demand points can only be served by their set of locations. Besides, a demand point can be removed if its set of locations is empty, which means that the demand point is not within range of any location.

2. An additional constraint can be added which results in tighter lower bounds as in Cournuéjols et al. [12]. More precisely, the total capacity of the service access points should be larger than the total demand for every service. In fact, this constraint is redundant, since it follows from the capacity and assignment constraint which is proved in Lemma 1. However, it enhances the performance of exact methods.

   Lemma 1. *Equation 5 is a redundant constraint in the formulation of the MSCFLP as given in Equation 3.*

$$\sum_{j \in \mathcal{L}} \eta^u x_j^u \geq \sum_{i \in \mathcal{G}^u} d_i^u, \qquad \forall u \in \mathcal{F}. \tag{5}$$

   Proof. Constraint 3c can be summated over all locations:

$$\sum_{j \in \mathcal{L}} \sum_{i \in \mathcal{G}^u} s_{ij}^u \leq \sum_{j \in \mathcal{L}} \eta^u x_j^u, \qquad \forall u \in \mathcal{F}. \tag{6}$$

In addition, Constraint 3d can be summated over all demand points:

$$\sum_{i\in\mathcal{G}^u}\sum_{j\in\mathcal{L}} s_{ij}^u \geq \sum_{i\in\mathcal{G}^u} d_i^u, \qquad \forall u \in \mathcal{F}. \tag{7}$$

The redundancy follows from combining Equations 6 and 7:

$$\sum_{i\in\mathcal{G}^u} d_i^u \leq \sum_{i\in\mathcal{G}^u}\sum_{j\in\mathcal{L}} s_{ij}^u \leq \sum_{j\in\mathcal{L}} \eta^u x_j^u, \qquad \forall u \in \mathcal{F}. \tag{8}$$

□

3. The formulation of the MSCFLP allows that a demand point can be served by multiple service access points, each serving an unitary fraction of the demand. Note that this can only be the case if the demand is larger than one, since the decision variables s_{ij}^u are restricted to be integer. There may exist services for which this is practically impossible. Therefore, it is decided to use the same formulation as the Single-Source Capacitated Facility Location Problem (SSCFLP) in Guastaroba and Speranza [23]. In this formulation, the connection variables are binary which implies the single-source property and reduces the solution space. This changes the definition of the decision variables s_{ij}^u as can be seen in Equation 9.

$$s_{ij}^u = \begin{cases} 1 \text{ if demand point } i \in \mathcal{G}^u \text{ is served by location } j \in \mathcal{L}, \\ 0 \text{ otherwise.} \end{cases} \tag{9}$$

4. The formulation of the MSCFLP allows that a demand point can be served with more demand than needed. This increases the symmetry in the problem, because there are no connection costs involved. In other words, a service access point which has not reached its capacity can serve more demand to its demand points. This results in many solutions with the same objective value, which may have a negative impact on the computation time of exact methods.

The problem formulation with the four improvements mentioned above can be found in Equation 10.

$$\text{Min} \sum_{u\in\mathcal{F}}\sum_{j\in\mathcal{L}} c_j^u x_j^u + \sum_{j\in\mathcal{L}} f_j y_j \tag{10a}$$

subject to

$$x_j^u \leq y_j, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{10b}$$

$$\sum_{i\in\mathcal{G}^u} d_i^u s_{ij}^u \leq \eta^u x_j^u, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{10c}$$

$$\sum_{j\in\mathcal{L}} s_{ij}^u = 1, \qquad \forall i \in \mathcal{G}^u, \forall u \in \mathcal{F}, \tag{10d}$$

$$\sum_{j\in\mathcal{L}} \eta^u x_j^u \geq \sum_{i\in\mathcal{G}^u} d_i^u, \qquad \forall u \in \mathcal{F}, \tag{10e}$$

$$s_{ij}^u \in \{0,1\}, \qquad \forall i \in \mathcal{G}^u, \forall u \in \mathcal{F}, \forall j \in \mathcal{L}_i^u, \tag{10f}$$

$$x_j^u \in \{0,1\}, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{10g}$$

$$y_j \in \{0,1\}, \qquad \forall j \in \mathcal{L}. \tag{10h}$$

The following improvements can be observed by comparing to the original formulation (Equation 3). The improvements are numbered similarly as they are explained above.

1. The decision variables s_{ij}^u in Equation 10f are only defined when $a_{ij}^u = 1$ and they are restricted to be binary. Therefore, the range constraint in Equation 3e is removed.

2. Constraint 10e is added.

3. The capacity constraint in Equation 10c contains the demand parameter d_i^u, since the decision variables s_{ij}^u are now binary. Besides, the right hand side of Constraint 10d is equal to one.

4. The assignment constraint in Equation 10d contains the equality sign to encounter the symmetry.

The alternative formulation changes the problem from an ILP to a Binary Integer Linear Programming (BILP) problem in which all decision variables are binary. Observations have shown that this improves the performance of exact methods, since the solution space is reduced. The number of binary variables is equal to $\sum_{u\in\mathcal{F}}\sum_{i\in\mathcal{G}^u}\mathcal{L}_i^u + |\mathcal{L}|(|\mathcal{F}|+1)$. The number of constraints is equal to $(2|\mathcal{L}|+1)|\mathcal{F}|+|\mathcal{G}|$.

2.2 Multi-Service Modular Capacitated Facility Location Problem

In the MSCFLP, an access location can only be equipped with one service access point of every service. A service access point may have more demand in its range than it is able to serve due to capacity restrictions. Other access locations are needed for the service access points that serve the remaining demand points. Allowing multiple service access points on one access location, is in literature also known as modular capacities. Note that the practical application of this is left open in this thesis. It can be interpreted as increasing the capacity of a service access point in discrete steps. In other words, multiple services access points on one access location are equivalent to one more expensive service access point having the same capacity. By all means, allowing modular capacities is expected to give significant costs benefits, since less access locations are needed. These benefits will mainly arise in dense areas, because locations in these areas have more demand points in their range than capacity available. Therefore, a new problem is introduced: the Multi-Service Modular Capacitated Facility Location Problem (MSMCFLP).

First of all, the following definition is needed. Let κ_j^u be the maximum number of service access point of service u on location j. The problem formulation is similar to the alternative formulation of the MSCFLP as in Equation 10 in which Constraint 10b and 10g are replaced by Constraint 11a and 11b, respectively.

$$x_j^u \leq \kappa_j^u y_j, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{11a}$$

$$x_j^u \in \{0, 1, \dots, \kappa_j^u\}, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}. \tag{11b}$$

Important to state is that the objective function stays the same, because it is assumed that the equipping costs are linear in the number of service access points that are placed on one location. In other words, for every service access point of service u on location j, the same equipping costs c_j^u have to be paid. Besides, the maximum number of service access point $\kappa_j^u \in \mathbb{N}^+$ can be determined based on practical reasons. This setup is very general to allow different sizes of locations and service access points. For example, a location can be equipped with many small service access points, but only a few large ones. Finally, the MSMCFLP is equal to the MSCFLP if $\kappa_j^u = 1, \forall j \in \mathcal{L}, \forall u \in \mathcal{F}$.

2.3 Multi-Service Location Set Covering Problem

The MSLSCP is a simplified version of the MSCFLP. More precisely, it is the MSCFLP without capacity restriction. This means that every demand point only has to be in the range of at least one service access point of the corresponding service. It can also be seen as the MSCFLP with non-binding capacity constraints, i.e. the capacity parameter η^u is equal to infinity for all services. An example is the air alarm service: every home in the range of an air alarm can receive the alarm signal, regardless of the number of homes in the range of the air alarm. In other words, there is no capacity restriction involved. The MSLSCP can be rewritten without the decision variables s^u_{ij}, because there is no need to keep track of the individual connections. The assignment constraint can be rewritten by making use of the coverage elements a^u_{ij} as can be seen in Equation 12.

$$\text{Min} \sum_{u \in \mathcal{F}} \sum_{j \in \mathcal{L}} c^u_j x^u_j + \sum_{j \in \mathcal{L}} f_j y_j \tag{12a}$$

subject to

$$x^u_j \leq y_j, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{12b}$$

$$\sum_{j \in \mathcal{L}} a^u_{ij} x^u_j \geq 1, \qquad \forall i \in \mathcal{G}^u, \forall u \in \mathcal{F}, \tag{12c}$$

$$x^u_j \in \{0, 1\}, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}, \tag{12d}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in \mathcal{L}. \tag{12e}$$

Note that the problem for one service is equal to the Set Covering Problem (SCP) as in Farahani et al. [18].

3 Literature Review

The MSCFLP is introduced by Hoekstra [27] and is not studied elsewhere. Most studies focus on one service in the Capacitated Facility Location Problem (CFLP), optionally with single-source constraints (SSCFLP). Besides, the Multi-Service Location Set Cover Problem (MSLSCP) is studied, because it is used in a heuristic. Finally, some background information is given about the *branch-and-cut* method, which is the exact method that is used in this thesis.

Facility location problems (FLP) are extensively studied in literature. In these problems, the placement of facilities is optimised to minimise the costs while serving all customers. Generally, the costs consist of the sum of the placement costs of facilities and the transportation costs between facilities and customers. However, many variations of the costs structure exist. An example is the minimax version in which the maximum distance between the facilities and customers is minimised. The problems can be divided into two categories: the capacitated (CFLP) [36] and the uncapacitated (UFLP) [17] problem. Both problems are known to be NP-hard, which means that it is impossible to solve these problems in polynomial time. This can be proved by a reduction from Vertex Cover [24]. The capacitated problem puts an additional restriction on the capacity of the facilities. Both the UFLP and the CFLP allow that a customer is served by multiple facilities, all serving a fraction of the demand. Many extensions of the CFLP exist. An example is the Modular Capacitated Facility Location Problem in which different facilities types can be chosen [13]. Another example that allows more flexibility is the Multi-Product Capacitated Facility Location Problem with General production and building Costs as in Montoya et al. [35]. In this problem, the placement of facilities for multiple products is optimised while also taking into account the production costs of the facilities.

The SSCFLP is an extensively studied version of the CFLP. In literature, this is also known as the Capacitated Concentrator Location Problem. It adds the restriction that customers can be served from only one facility. Because this is an extension of the CFLP, the problem is NP-hard. Therefore, many heuristics have been proposed in literature. They differ from each other in computation time and objective value. Most heuristics are Lagrangian in which a constraint is relaxed and placed in the objective function. Violations of the corresponding constraint are penalised by the Lagrange multiplier. The relaxed problem is easier to solve and forms an approximation to the original problem. The difficulty lies in the updating of the Lagrange multiplier, which is usually done by *subgradient optimisation*. Klincewicz and Luss [31] relaxed the capacity constraint which resulted in the UFLP. This problem is solved by a *dual ascent* algorithm whereafter the solution is made feasible by an *add* heuristic. Pirkul [37] relaxed the assignment constraint and solved the remaining problem by multiple knapsack problems. Both constraints are relaxed by Beasley [8], whose algorithm was later improved by Agar and Salhi [2]. Besides, different approaches have been researched. Hindi and Pieńkosz [25] found efficient solutions of large scale problems by applying Lagrangian Relaxation with *restricted neighbourhood search*. Gadegaard et al. [20] proposed an improved *cut-and-solve* algorithm that consists of three phases. A *cutting planes* algorithm for the lower bound, a *two-level branching* heuristic for the upper bound and optionally a *cut-and-solve* framework to close the optimality gap. A *repeated matching* heuristic was found by Rönnqvist et al. [38], which often produced better results than Lagrangian heuristics. Also metaheuristics have been proposed. Filho and Galvão [19] made a *tabu search* heuristic which was shown to be competitive to other solution methods available. Ho [26] extended this approach with an *iterated tabu search* heuristic to avoid convergence to local optima. Other examples are the *reactive GRASP* heuristic of Delmaire et al. [16], the *kernel search* framework of Guastaroba and Speranza [23] and the *scatter search* algorithm of Contreras and Diaz [11]. Furthermore, Ahuja et al. [4] applied Very Large Scale Neighbourhood search procedures for solving large problems. This procedure was later improved by Tran et al. [41] resulting in better solutions in shorter computation times.

The MSCFLP extends the SSCFLP by taking multiple products into account. Until now, the problem is only used in the smart city context in which products are called services. Different services access points have to be distributed over the city such that the demand for all services is satisfied. Costs savings can be obtained by placing multiple service access points of different services on the same access location. Hoekstra [27] extended this problem by allowing Partial Covering in which not all demand had to be covered. A penalty is given to the unserved demand points. Both problems were solved by an exact method and two heuristics. The best results were obtained by a heuristic that sequentially solved the problem for one service by an exact method with costs updating. This is based on the fact that opening costs only have to be paid once for every location. After a location is opened, only the equipping costs need to be paid for the subsequent services. It is important to notice that the MSCFLP is not a generalisation of the CFLP or vice versa, since there are no connection costs involved. On the other hand, the MSCFLP takes multiple services into account which extends the CFLP. The MSCFLP can be seen as a generalisation of the SSCFLP without connection costs.

The MSLSCP is a simplification of the MSCFLP, as explained in Section 2.3. Vos [44] showed that exact methods need long computation times to solve the large instances. Therefore, two greedy heuristics and one Lagrangian heuristic were found. The greedy heuristics were made from the perspectives of the demand points and locations. The Lagrangian heuristic is based on the fact that the problem for one service boils down to an SCP. Garey and Johnson [21] proved that this problem is NP-hard. As a result, the SCP is an extensively studied problem in literature, for which many heuristics exist. An example is the Lagrangian heuristic of Beasley [7], which is shown to be very efficient. The Lagrange multiplier λ is updated by *subgradient optimisation*. However, the computation times are still high for large instances. Ceria et al. [10] showed that the computation times can be reduced by defining the core problem. This boils down to only regarding the sets with the lowest costs, while keeping all items included in a minimum number of sets. Vos [44] showed that this combination of Lagrangian Relaxation and defining the core problem performed very well on the MSLSCP.

Next to heuristics, a wide range of exact methods exists to solve ILP problems like the FLP. Heuristics usually find good or near-optimal solutions in a reasonable amount of time. On the other hand, exact methods by definition find the optimal solution, but an excessive amount of time may be needed. In this thesis, the *branch-and-cut* method is considered. This is a combination of the *branch-and-bound* and *cutting planes* algorithm. Separately, the *branch-and-bound* method needs long computation times to solve an ILP problem. On the other hand, it is usually impossible to solve an ILP problem by only using *cutting planes*, according to Mitchell [34]. However, the combination of both algorithms results in an efficient method. Next to solving the problem, the *branch-and-cut* algorithm provides bounds. If the algorithm is terminated before finding the optimal solution, an indication of the maximum distance to the optimal solution can be deducted from the bounds. In this way, the performance of the heuristics can be examined. A more thorough explanation of the *branch-and-cut* algorithm is given in Section 5.1.2.

4 Experimental Design

The same experimental design as in Hoekstra [27] is used, which is presented in this section. First of all, the set-up of the locations, services and demand points is explained. Thereafter, an overview of the test instances is given. Realistic data is used to give a proper estimation of the costs benefits that can be gained. Finally, the software and hardware that are used to conduct the experiments are described.

4.1 Locations

Lampposts are taken as locations that can be equipped with services. This choice is made on the following grounds.

- There are many lampposts which induces a large set of options.
- Lampposts are well spread over a city which enhances the coverability.
- Lampposts already have an electricity connection, which make them practically suitable to equip with service access points.
- The position of lampposts is well documented. This makes it easy to come up with test instances based on real data.

These grounds do not hold for other options, e.g. traffic lights or bus stops. Both are only present at busy roads which counteract the coverabilitiy. Another options is to build new locations, without using the existing infrastructure. The advantage of this is that the new locations can be placed on tactical spots, although high costs are involved. Besides, using the existing infrastructure is more sustainable, since no new locations need to be built. The coordinates of the locations are obtained via Dataplatform. The opening costs are the costs of connecting the lampposts to the internet or another medium to transfer information. They can differ from each other, since some lampposts may be more difficult to connect than others. On the other hand, the differences between the opening costs are bounded by the triangle inequality [39], since the costs of connecting lamppost I cannot be higher than the costs of connecting lamppost II plus connecting the two lampposts. Therefore, it is chosen to take the opening costs randomly between $4,000$ and $5,000$, i.e. $f_j \sim [4,000, 5,000]$. Moreover, this has practical advantages for the computation times of the solution methods, since it reduces the symmetry of the problem. The random generator is restored after every run to ensure that the same opening costs are taken for every run.

4.2 Services

In the smart city context, a wide range of services can be required. In the experimental design of this thesis, three different services are considered: Wi-Fi, Smart Vehicle Communication (SVC) and Alarm. The Wi-Fi service provides wireless internet access in the city. This can be relevant for tourism or may replace the existing fiber network in the long run. The SVC units communicate with vehicles on the road to optimise the traffic flow, which can decrease traffic jams and the associated air pollution. Besides, these units are needed to enable the usage of self-driving cars. Finally, the Alarm service can give alarm signals to inform citizens about disasters et cetera. In the Netherlands, this already exists in the form of air alarms. However, the distribution of these services can be done simultaneously to gain costs benefits. The three services differ from each other in range, costs and capacity. An overview of these properties can be found in Table 3.

Table 3: Overview of the parameters of the services.

Number u	Name	Range	Costs c_j^u	Capacity η^u
1	Wi-Fi	100	300	30
2	SVC	200	350	15
3	Alarm	300	150	∞

Note that the equipping costs of a service access point are taken equal for all locations. The range is defined in metres and it is assumed that the coverage area of every service is circular. This allows us to determine the coverage elements a_{ij}^u based on the coordinates (longitude and latitude) of the locations and demand points. The haversine method is used to calculate the distance between two points on a sphere, in this case two sets of coordinates on the globe. This method takes the curvature of the globe into account, which is more precise than the Euclidean distance. However, this effect is expected to be negligible on the scale of a city. Note that the height of a demand point or location is not taken into account. For example, it is assumed that there is no difference between a Wi-Fi demand point on the first and the tenth floor of a building. Furthermore, the assumption of a circular coverage area is not true for all services, e.g. the Wi-Fi connection is dependent on the infrastructure. However, these simplifications are needed to make the problem seizeable. Different ranges and capacities are chosen in such a way that different amounts of service access points are needed to serve the demand. The Wi-Fi service has the smallest range of hundred metres and has a capacity of thirty. This means that one service access point can serve up to thirty demand points with demand equal to one within its range. The SVC service has a larger range, but a smaller capacity. The Alarm service has the largest range and an infinite capacity, since every demand point within the range can receive the alarm signal.

For simplicity, it is chosen to take $\kappa_j^u = \kappa$ the same for every location and service in the MSMCFLP. This means that every access location can be equipped with κ service access points of every service. The parameter κ is also called the modularity parameter, since it determines the possibilities of the modular capacities. The results are generated for $\kappa = 3, 5, 10$. Note that the MSMCFLP with $\kappa = 1$ is equal to the MSCFLP for which results are generated as well.

4.3 Demand Points

The demand points indicate the geographical locations that require a specific amount of a certain service. They are spread over the city, since services can be required anywhere. However, the density of the demand points can vary. For example, less services are generally required in parks than in shopping districts. Every service has its own set of demand points and every demand point requires only one service. The coordinates of the demand points are based on real data. All demand points are located inside the region boundary that is based on the lampposts. This minimises the number of remote demand points that are not within the range of any location. Home addresses are taken as Wi-Fi demand points, all with demand equal to one. The SVC demand points are based on the roads. The coordinates of these points lie ten to hundred metres away from each other, such that the curvature of the road is represented. The demand of these points can vary between one, two and three, depending on the traffic density of the road. Demand points have a demand equal to three on highways, a demand equal to two on regional roads and a demand equal to one on all other roads. The Alarm demand points are equally spread inside the region boundary, such that the entire area is covered. All Alarm demand points have a demand equal to one.

4.4 Test Instances

Nine subareas of Amsterdam are taken as test instances, which differ from each other in size and density. It is tried to include subareas with different aspects, such as highways, parks and canals. This can be seen in the geographical maps of the instances given in Appendix B. The sizes of the instances can be found in Table 4. Instances 1-7 are referred to as small, whereas instance 8 and 9 are referred to as large.

Table 4: Overview of the sizes of the test instances.

		# Demand points			
Instance	**# Locations**	**Wi-Fi**	**SVC**	**Alarm**	**Total**
1	33	47	3	9	59
2	77	73	8	15	96
3	99	260	9	13	282
4	102	462	8	15	485
5	400	111	20	25	156
6	782	21	93	104	218
7	516	1,241	42	46	1,329
8	6,079	8,106	397	326	8,829
9	6,981	10,122	528	431	11,081

The number of locations is relatively high compared to the number of demand points, taking into account the defined capacities in Table 3. For example, one Wi-Fi access point can serve thirty demand points, which implies that many locations will not be opened. However, this is only the case for dense instances in which the demand points and locations are close to each other. It is expected that relatively more locations are opened in sparse instances, because the service access points need to cover a larger area in that case.

Some demand points are discarded as they are not within the range of any location. Two Wi-Fi and one SVC demand point are removed from instance 6. In addition, four Wi-Fi and five SVC demand points are removed from instance 9. However, these amounts are relatively small compared to the total number of demand points in these instances.

4.5 Software and Hardware

The experiments are programmed in the programming language MATLAB (version R2018a), which is widely used in universities and business. The program is published by Mathworks, a corporation that is specialised in mathematical computing software. MATLAB includes many different functions for matrix multiplication, the implementation of algorithms, visualisation and the creation of interfaces. The optimisation is carried out by the commercial solver CPLEX ILOG CPLEX Optimisation studio (version 12.8.0), also known as CPLEX. This is a commercial solver that continues to be developed by IBM. CPLEX uses flexible and comprehensive methods to solve, among other things, Linear Programming (LP) and Integer Linear Programming (ILP) problems. A student license for both programs is obtained via the Vrije Universiteit. The experiments are executed on a DELL laptop with an Intel(R) Core(TM) i7-4810MQ CPU 2.80 GHz and an installed RAM of 16.0 GB.

5 Solution Approach

In this section, the different methods to obtain solutions are presented. First of all, the exact method is discussed. Thereafter, three heuristics are given: the Extended Pricing Heuristic (EPH), the Extended Linear Relaxation Heuristic (ELRH) and the Extended Sequential Covering Heuristic (ESCH). The first two heuristics execute an algorithm simultaneously for all services, whereas the latter one does this sequentially for every service. The three heuristics consist of two phases. Firstly, a feasible solution is found which is optimised by an exact method in the second phase. The two main reasons for this approach are stated below.

- Large instances are considered in this thesis. Generally speaking, solving the whole problem instantly by an exact method takes excessive amounts of time. The first phase can be seen as a warm-up to reduce the size of the problem in the second phase.
- Using heuristics usually goes along with sub-optimality. This effect is tried to be reduced by using an exact method in the second phase.

All solution methods can be applied to both the MSCFLP and the MSMCFLP. The methods are explained considering the MSCFLP. The extensions to MSMCFLP are given at the end of each subsection. The computational results will follow in Section 6.

5.1 Exact Method

The exact method serves as a benchmark to the heuristics and is applied to the alternative formulation (Equation 10). The exact method obtains optimal solutions to the ILP problems. However, the computation times are dependent on the problem size. Usually, small problems can be solved quickly, while larger problems may need an excessive amount of time to obtain the optimal solution. Therefore, stopping criteria are determined which are discussed in Subsection 5.1.3. As mentioned before, the exact method that is used in this thesis is CPLEX. In general, this method consists of two steps: preprocessing and a *branch-and-cut* algorithm, which are explained in Subsection 5.1.1 and 5.1.2, respectively.

5.1.1 Preprocessing

The preprocessing is applied to enhance the performance of the *branch-and-cut* algorithm afterwards. CPLEX automatically preprocesses the problem in the following ways.

- Eliminate redundancy: It can be the case that a constraint is more restrictive than other ones. The less restrictive constraints are called redundant and can be removed, since they have no effect on the problem.
- Simplify constraints: Constraints are usually implemented from a practical perspective. Take for example Constraint 10c, which is formulated from the perspective that capacity has to be taken into account. However, this is not necessarily the most efficient formulation. In some cases, decision variables appear multiple times in a constraint or coefficients can be scaled down.
- Reduce problem size: This one is less intuitive. The problem consists of an objective function, constraint matrix and boundaries for the decision variables. Rows and columns in the constraint matrix, which present constraints and variables in the problem, may be removed through substitution. This is a mathematical technique to simplify matrices.

Important to remark is that there are many options users can specify to perform the preprocessing methods given above. After the preprocessing, the *branch-and-cut* algorithm is executed.

5.1.2 Branch-and-cut

Branch-and-cut is a combinatorial optimisation method for solving ILP problems. It combines the *branch-and-bound* and *cutting planes* algorithm. Firstly, the *branch-and-bound* method is addressed, since it forms the body of the *branch-and-cut* algorithm. Thereafter, the *cutting planes* method is explained and finally the combination of both algorithms is given. Without loss of generality, a minimisation problem is regarded.

The *branch-and-bound* algorithm tries to classify the solution space, i.e. the possible values that the decision variables can attain. Usually, it costs an excessive amount of time to simply check all possible solutions. Therefore, the algorithm partitions the solution space, which is also called branching. Some areas could be interesting to dive into and others can be excluded, because they have no potential. For this process, bounds are used to decide whether solution areas are interesting or not. More precisely, the property is used that the problem without integrality constraints, also called the LP relaxation, gives better solutions than the original problem. This follows trivially from the fact that relaxations contain less constraints by definition. The solution to the LP relaxation can be computed efficiently using the *Simplex* algorithm. If in a certain solution area, the LP relaxation gives a worse solution than the best known feasible solution, this solution area surely does not contain an improving feasible solution. The algorithm will proceed until all solution areas are classified. To state this mathematically, some definitions are needed. Let S be the set of all active problems, i.e. problems with solution areas that are not yet investigated. Besides, let x^* be the current best feasible solution, also known as the incumbent, and z^* the corresponding objective value. The incumbent can be seen as the upper bound for the optimal solution, since it is a feasible solution but a smaller one may exist. Finally, let x be the solution of an LP relaxation and z the corresponding objective value. A generalised overview of a *branch-and-cut* algorithm on an ILP problem with minimisation objective as in Johnson et al. [30] can be found in Algorithm 1. For the *branch-and-bound* algorithm, line 10 can be skipped.

Algorithm 1: Overview of a *branch-and-cut* algorithm on a minimisation ILP problem.

1 Set $S = \{P0\}$;
2 Set $z^* = \infty, x^* = []$;
3 **while** $S \neq \emptyset$ **do**
4   Select a problem $P \in S$ and update $S = S \setminus P$;
5   Solve the LP relaxation of P;
6   **if** $z \leq z^*$ **then**
7   **if** x *is integer* **then**
8   Update $z^* = z, x^* = x$;
9   **else**
10   Add cutting planes and return to line 5;
11   Branch problem P into multiple problems and add them to S;
12   **end**
13   **end**
14 **end**

Note: Omitting line 10 gives the *branch-and-bound* algorithm.

The algorithm starts with the whole problem $P0$ and initialises the incumbent and the corresponding objective value. Optionally, these variables can be initialised with a specified starting solution. Next, the algorithm selects an active problem P and solves the LP relaxation. The algorithm checks whether the objective value z of the solution of the LP relaxation is lower than the incumbent. If this is not the case, the algorithm takes another problem from the set of active problems S, because this solution area surely does not contain an improving incumbent. If this is the case, the algorithm checks whether the solution x of the LP relaxation is integer to eventually update the incumbent. Otherwise, the algorithm branches the problem into multiple problems and adds them to the set of active problems. An example of a *branch-and-bound* procedure is given in Figure 2. The overview is called a tree and one specific problem is called a leaf. Removing a leaf from the active set is also called pruning.

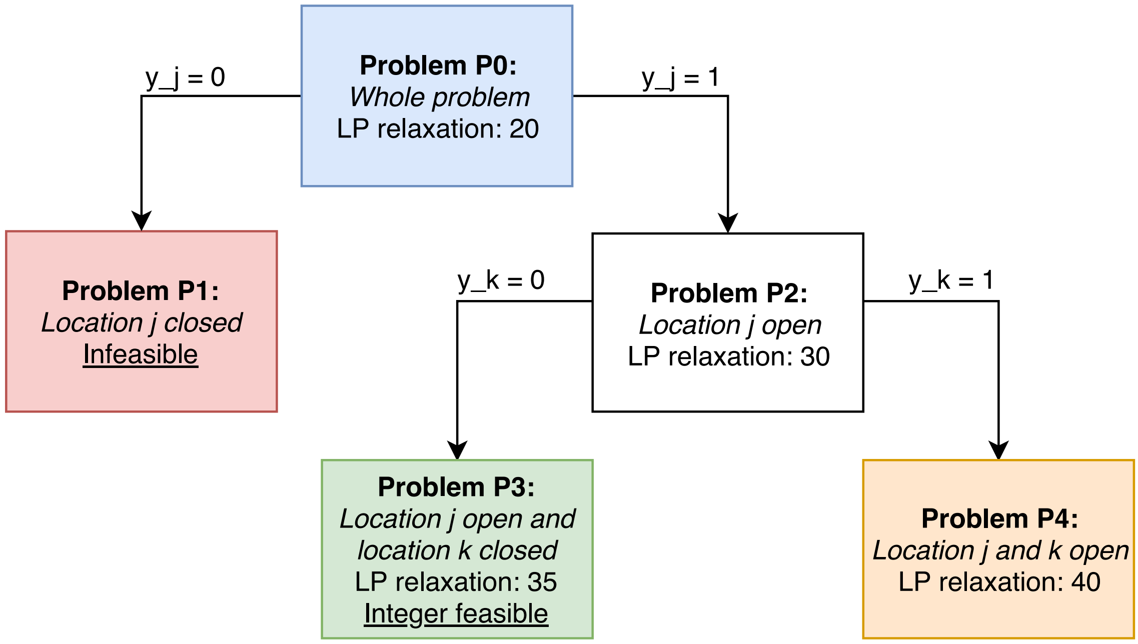


Figure 2: Example of a *branch-and-bound* procedure.

The procedure starts with the whole problem $P0$ and solves the LP relaxation with an objective value of 20. The solution is fractional, so the algorithm decides to branch on the variable y_j which indicates if location j is open ($y_j = 0$) or closed ($y_j = 1$). This results in two problems: $P1$ and $P2$. Problem $P1$ has no feasible solution, since location j is closed. This can be caused by a demand point that can only be served by location j. If location j is closed, this demand point cannot be served anymore. The algorithm prunes this leaf and continues with problem $P2$. Trivially, the solution of the LP relaxation of problem $P2$ is higher than the one of problem $P0$, since there is the additional restriction that location j is open. The solution is still fractional, so the algorithm continues branching. It decides to branch on the variable y_k which indicates if location k is open. Again, the solutions of the LP relaxations in both leaves are calculated. The solution of problem $P3$ is integer feasible with an objective value equal to 35. As a result, the incumbent is updated. Problem $P4$ gives a solution of 40 which allows the algorithm to prune the leaf, since the solution of the LP relaxation is higher than the incumbent. The algorithm cannot continue, because the set of active problems is empty. All leaves having a lower solution to the LP relaxation than the incumbent are already branched. Therefore, the incumbent is the optimal solution. Usually, more iterations are needed, but this procedure serves as an example to the working of the *branch-and-bound* algorithm.

The bottleneck of this procedure is that there is a small probability that the LP relaxation gives a feasible solution. For this reason, *cutting planes*, also known as cuts, can be added to the problem.

Cuts are constraints that eliminate infeasible solutions. Cuts can either be local or global [34]. Local cuts are only applied to the current leaf and the leaves following from it and global cuts yield for the entire tree. A wide range of methods exists to create cuts. A graphical example of the working of this algorithm can be found in Figure 3.

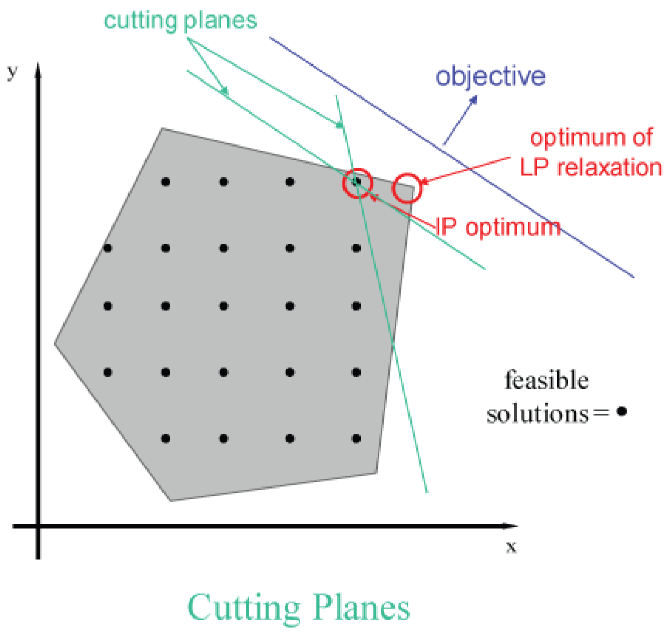


Figure 3: Graphical example of the *cutting planes* algorithm.
Source: `http://www.gurobi.com/resources/getting-started/mip-basics`

The grey polyhedron indicates the solution space of the LP relaxation and the constraints form the boundaries. The black dots represent the feasible integral solutions. The solution of the LP relaxation is a corner point of the polyhedron [14]. Unfortunately, the corner points are usually infeasible solutions, since they do not coincide with the black dots. Therefore, a part of the polyhedron with only fractional solutions is cut off by two cuts such that the new corner point is integral. However, many cuts could be needed to make sure that every corner point is integral. Therefore, only a predetermined number of cuts can be applied to large problems.

The *cutting planes* algorithm is integrated in the *branch-and-bound* algorithm in the following way [34]. If the solution of the LP relaxation is fractional, it will add cuts to the problem and solve the LP relaxation again. This is executed in line 10 of Algorithm 1. After the cuts are added, two cases are possible.

- The solution of the LP relaxation with cuts is integral. The algorithm compares the solution to the incumbent. If the solution is lower, the incumbent is updated. Furthermore, the algorithm stops in this leaf, because branching from this leaf will not give better results.

- The solution of the LP relaxation with cuts is still fractional. By all means, the cuts have not decreased the objective value of the solution, since constraints are added. The algorithm compares the solution to the incumbent and only continues branching from this leaf if the solution is lower than the incumbent. Otherwise, this leaf is pruned as there cannot be a better feasible solution in this solution area.

As said before, this is a brief explanation about the *branch-and-cut* algorithm. Many different methods exist for cut generation, e.g. *mixed integer rounding cuts*. A *mixed integer rounding cut* is generated by rounding the coefficients of the integer decision variables and the right hand size of the constraint matrix as in Marchand et al. [33]. Other examples are *lift-and-project cuts* [5], *Gomory fractional cuts* [22] and *zero-half cuts* [9]. In addition, many different branching strategies exist of which an overview can be found in Achterberg et al. [1]. An example is *maximal infeasible branching* which branches on the decision variable in the solution of the LP relaxation with the fractional part closest to a half. The opposite strategy also exists and is called *minimal infeasible branching*. Other strategies are *strong branching* and *pseudo costs branching*. The advantage of CPLEX is that it identifies the properties of the problem and applies the most suitable methods for branching and cut generation. Another important feature of CPLEX is that it keeps track of the lower bound. For minimisation problems, this value is smaller than or equal to the optimal solution. The lower bound can be determined based on the solutions of the LP relaxations. The gap between the incumbent and the lower bound is also called the optimality gap. It is an indication of the maximum improvement that can be achieved. This is interesting for large problems, since there is generally not sufficient time to wait for convergence to the optimal solution. Consequently, stopping criteria have to be determined which are explained below.

5.1.3 Stopping Criteria

It is important to specify the stopping criteria, since it is not always possible to obtain the optimal solution. The exact method is used separately and as part of the heuristics that are explained in the next subsections. In both cases, it is decided to terminate the solution if the optimality gap is below 0.01%, which is calculated as in Equation 13.

$$\text{Optimality gap } = \frac{\text{Objective value - Lower bound}}{\text{Objective value}} \cdot 100\%. \tag{13}$$

Besides, the computation time is bounded. For the separate use of the exact method, the maximum computation time is set equal to one hour for the small instances (1-7) and twelve hours for the large instances (8 and 9) in order to give the exact method sufficient time to optimise. For the exact method in the heuristics, the maximum computation time is set equal to five minutes. The reason for this is the importance of the computation time on the performance and user-friendliness of the heuristics. A sensitivity analysis on the maximum computation time of the heuristics is given in Section 6.5.

5.2 Extended Pricing Heuristic

This heuristic is developed from the perspective of service access points. It greedily equips locations with service access points for all services simultaneously. The heuristic consists of the following two phases:

1. Find a feasible solution to the MSCFLP using the *Pricing Heuristic*.
2. Solve the MSCFLP using the exact method on the restricted problem with only the intermediate access locations.

The solution of the first phase is also referred to as the intermediate solution, whereas the solution of the second phase is the final solution. In the first phase, the EPH finds a feasible solution to the whole problem in a greedy way by the *Pricing Heuristic*. In the second phase, the alternative formulation (Equation 10) is applied to the restricted problem. This problem contains all the demand points and only the intermediate access locations, i.e. the access locations in the solution of the first phase. Reducing the whole problem to the restricted problem in the second phase decreases the computation time of the exact method. Note that the second phase always gives a feasible solution, since the solution of the first phase is already feasible. On the other hand, the *Pricing Heuristic* is not expected to give the best results in the first phase by its greedy nature. The second phase gives optimal solutions to the restricted problem by the exact method. However, this can result in long computation times, especially for large instances.

The *Pricing Heuristic* iteratively equips access locations with 'cheap' service access points till all demand points are served. An overview of the algorithm can be found in Algorithm 2.

Algorithm 2: Pricing Heuristic

```
while Not every demand point is served do
    Calculate price of service access points;
    Equip access location with cheapest service access point;
    Order demand points in range of selected service access point based on reachability;
    while Capacity of selected service access point is not exceeded do
        Serve demand points by selected service access point;
    end
    Update costs, demand points, service access points;
end
```

The price of a service access point of service u on location j is calculated as in Equation 14.

$$P_j^u = \frac{\text{number of unserved demand points of service } u \text{ in range of location } j}{\text{costs of equipping location } j \text{ with service } u}. \tag{14}$$

The reachability of a demand point is calculated as the number of possible service access points that can still serve the demand point. The last step in the algorithm consists of updating the variables. The costs are updated, since the opening costs only have to be paid once for every location. This means that after this step only equipping costs have to be paid at this location. The served demand points are left out in the next iteration.

This heuristic can be used for the MSCFLP and the MSMCFLP, since only one service access point is added in each iteration. If location j is already equipped with κ_j^u service access points of service u, no more service access points of that service can be placed on location j. In the second phase, the corresponding ILP formulation can be used to optimise the number of service access points and access locations in the final solution.

5.3 Extended Linear Relaxation Heuristic

This heuristic is developed from the perspective that the integrality constraints are difficult to satisfy. It uses the property that LP problems can efficiently be solved using the *Simplex* method. The problem without integrality constraints is also called the LP relaxation. This heuristic considers all services simultaneously and consists of the following two phases:

1. Solve the LP relaxation of the MSCFLP.
2. Solve the MSCFLP using the exact method on the restricted problem with only the intermediate access locations.

The first phase solves the LP relaxation which usually results in a fractional solution. Therefore, the intermediate access locations are defined as the locations $j \in \mathcal{L}$ with $y_j > 0$. The number of intermediate access locations is high if the opening costs are the same for every location. Namely, the solution of the LP relaxation contains many decision variables $y_j \in [0, \epsilon]$ for small ϵ. Consequently, the computation time of the exact method in the second phase increases, because the problem size is still large. On the contrary, when the opening costs have variation, the *Simplex* method identifies the cheap locations which decreases the number of intermediate access locations significantly. Therefore, the opening costs have variation as explained in Subsection 4.1. The second phase optimises the final number of access locations and service access points as in the EPH.

Note that also this heuristic can be used for the MSCFLP and MSMCFLP. The LP relaxation and ILP formulation are different, but the method stays the same.

5.3.1 Reducing the Solution Space in the Second Phase

The solution space of the ELRH in the second phase can be reduced by only selecting the intermediate access locations with $y_j > \theta$, $\theta \in [0, 1]$. This will decrease the computation time for all instances. Intuitively, those locations are more important than the locations with low values for y_j. However, this procedure for $\theta > 0$ can lead to infeasibility, since not all access locations of the first phase are present in the second phase. Besides, the objective value may be higher, since only a subset of the locations is regarded. Moreover, it takes time to check the feasibility for different values of θ. To encounter the problem of infeasibility, Algorithm 3 can be used.

Algorithm 3: Find optimal theta

Solve the LP relaxation;
for $\theta = \theta_0$ *to* 0 *with stepsize* ϵ **do**
    Determine intermediate access locations with $y_j > \theta$;
    if *Feasible solution to restricted problem* **then**
        break;
    end
end
$\theta^* = \theta$

The algorithm starts with solving the LP relaxation of the problem. Then, it loops over the values of θ starting from θ_0 to 0 with step size ϵ. In each iteration, it determines the intermediate access locations for the corresponding value of θ and checks whether a feasible solution to the restricted problem exists. The algorithm stops if there is a feasible solution, because that would be the highest value of θ for which a feasible solution exists. The highest value of θ means the least number of intermediate access locations, which results in a smaller solution space of the restricted problem in the second phase. In most cases, this will decrease the computation time in the second phase. In each iteration, there are more intermediate access locations which increases the probability of the presence of a feasible solution. Note that a feasible solution for $\theta = 0$ always exists by taking $y_j = 1$ for all intermediate access locations, i.e. all locations with $y_j > 0$ after the first phase. In this thesis, θ_0 and ϵ are taken equal to 0.10 and 0.01, respectively. The results of the algorithm on the nine instances of the MSCFLP can be found in Table 5.

Table 5: Optimal values of θ with the corresponding number of intermediate access locations of the ELRH on the MSCFLP.

		# Intermediate access locations	
Instance	$\boldsymbol{\theta^*}$	$\boldsymbol{\theta = 0}$	$\boldsymbol{\theta = \theta^*}$
1	0.10	3	3
2	0.06	6	5
3	0.06	12	12
4	0.10	19	17
5	0.06	12	12
6	0.06	39	37
7	0.06	50	50
8	0.03	418	418
9	0.00	530	530

Overall, θ^* is decreasing in the size of the instances. This can be explained by the fact that large instances have a higher probability of having an intermediate access location with $y_j \leq \theta^*$, which cannot be omitted in the second phase. In addition, Table 5 shows that, even for instances with high values of θ^*, only few intermediate access locations are omitted. This does not decrease the solution space significantly. A more specific analysis of the value of θ on the number of intermediate access locations is done on instance 9 which can be seen in Figure 4.

As expected, the blue line is decreasing in θ. Higher values of θ imply less access locations. The orange line represent the number of access locations in the final solution, which is equal to 401. The two lines meet each other at $\theta = 0.26$, which means that there are 401 intermediate access locations with $y_j > 0.26$. It would be perfect if those locations are able to form a feasible solution in the second phase. Unfortunately, this is not the case. More specifically, no intermediate access locations can be omitted for instance 9, since $\theta^* = 0$. To conclude, determining θ^* takes time and has no significant effect on the solution space. Therefore, θ is taken equal to zero in the rest of this thesis.

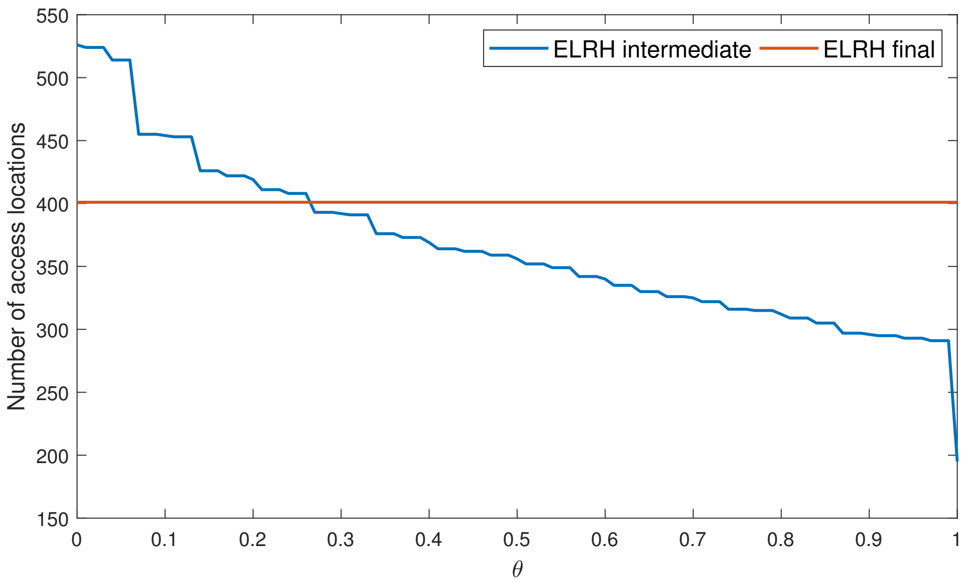


Figure 4: The blue line represents the number of intermediate access locations with $y_j > \theta$ for different values of θ on instance 9. The orange line represents the number of access locations in the final solution.

5.4 Extended Sequential Covering Heuristic

This heuristic is developed from the perspective that the capacity restriction is difficult to satisfy. By relaxing this restriction, the problem is equal to the MSLSCP for which efficient solutions can be obtained. The ESCH consists of the following two phases:

1. Find a feasible solution to the MSCFLP using the Sequential Covering Heuristic (SCH).

2. Solve the MSCFLP using the exact method on the restricted problem with only the intermediate access locations.

In the first phase, the SCH finds a feasible solution by sequentially executing the Covering Heuristic (CH) for the different services. The outline of the SCH can be found in Algorithm 4. The second phase optimises the solution of the first phase by an exact method in the same way as the EPH and ELRH.

Algorithm 4: Sequential Covering Heuristic

Costs of locations are equal to opening costs plus equipping costs;
Order the services increasingly based on their range;
for $u \in \mathcal{F}$ **do**
    Execute the CH for service u;
    Update costs;
end

Note that this algorithm sequentially considers the problem for only one service which means that locations in iteration u represent service access points of service u in the whole problem. In this way, the costs of the locations for the first service are equal to the opening costs plus the equipping costs. For the subsequent services, the costs of the already opened locations are equal to the equipping costs only. The algorithm can execute the services in different orders. Hoekstra

[27] showed for another algorithm that the best solutions were obtained by ordering the services increasingly based on their range. The intuition for this is that the service with the smallest range needs the most service access points. The service access points of the services with a larger range can be placed on the already opened locations of the previous iterations.

The CH starts with finding a feasible solution to the SCP to cover the demand points. The solution consists of a set of locations. Then, those locations serve as many demand points as possible in their corresponding range. Every demand point is in the range of a location, since the locations form a covering. However, not all demand points can necessarily be served due to capacity restrictions. It could be the case that the demand in the range of a location exceeds the capacity. After all the locations in the covering have served as many demand as their range and capacity allows, the served demand points and locations in the covering are removed and a covering of locations is found again on the residual problem. This procedure is repeated until all demand points are served. An overview of this procedure can be found in Algorithm 5. To obtain solutions to the SCP's, the algorithm of Vos [44] is used. This algorithm is a combination of Lagrangian Relaxation and a method to define the core problem.

Algorithm 5: Covering Heuristic

```
while Unserved demand points exist do
    Find feasible solution to SCP;
    Order locations of SCP solution increasingly based on demand in their range;
    for Locations of SCP solution do
        Order demand points in range of location based on number of times covered in SCP
          solution and reachability;
        while Capacity of location is not exceeded do
            Serve demand points by location;
        end
        Update locations and demand points;
    end
end
```

Important to notice is that some demand points can be covered multiple times in an SCP solution, which means they can be served by multiple locations. Intuitively, the location with the least demand in its range starts serving demand points so that some demand of the locations with more demand is already served. In this way, it is expected that the least number of coverings are needed. Furthermore, the demand points in the range of a location are ordered by the number of times covered in the SCP solution and the reachability, which is the number of possible connections to all remaining locations. This minimises the probability that a demand point cannot be served by any location.

The ESCH can be used for both the MSCFLP and the MSMCFLP. In the first phase, the capacity of a location in the CH can be multiplied by κ_j^u, which is defined as the maximum number of service access points of service u on location j. Furthermore, the corresponding ILP formulation can be used in the second phase.

6 Computational Results

In this section, the computational results of the solution methods from Section 5 are discussed. The solution methods consist of the exact method and three heuristics: EPH, ELRH and ESCH. The exact method serves as a benchmark to the heuristics. The results are conducted based on the experimental design of Section 4 which contains nine test instances. First of all, the results of the solution methods on the MSCFLP are addressed. Thereafter, the results on the MSMCFLP with $\kappa = 3, 5, 10$ are given. The analysis of the results is based on the solution methods and the modular capacities, which can be found in Subsection 6.2 and 6.3, respectively. The first analysis focuses on the effects of the solution methods on the results given the different values of the modularity parameter κ. The second analysis focuses on the effects of the modular capacities on the results given the solution methods. Furthermore, a lower bound analysis is given. This gives an overall indication of the performance of the solution methods and the potential that is left. Finally, the results of a sensitivity analysis on the maximum computation time are presented.

6.1 MSCFLP Results

In this subsection, the results of the solution methods on the MSCFLP are addressed. Table 6 gives a summarised overview of the computational results. An extensive overview of the computational results can be found in Table 15 in Appendix A.1. Based on these tables, many results can be described which are organised as follows: total problem, first phase and second phase. The total problem consists of the objective value and total computation time as can be seen in the table below. The first and second phase give a more thorough analysis based on Table 15.

Table 6: Summarised overview of the computational results of the solution methods on the MSCFLP. The objective values of the heuristics comprise the relative differences to the exact method. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3.

	Exact		EPH		ELRH		ESCH	
No.	**Obj.**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**
1	9,139	0.1	2.0	0.0	0.0	0.1	2.0	0.2
2	18,266	20.4	0.9	0.0	23.5	0.2	0.1	0.2
3	39,717	3.0	0.6	0.3	10.7	0.7	0.3	0.3
4	70,869	3.1	7.2	0.3	0.0	0.4	4.0	0.7
5	31,788	3,600.5†	0.7	0.1	12.1	0.3	12.5	0.1
6	90,574	3,600.6†	22.6	0.1	20.5	0.4	4.7	0.2
7	190,865	3,600.2†	2.2	2.0	1.7	2.8	2.2	4.7
8	1,346,700	43,202.1†	6.4	315.7†	4.4	324.9†	2.6	384.4†
9	1,699,594	43,205.4†	6.1	326.4†	4.4	327.1†	2.9	514.4†

The solution represents a distribution of service access points over the locations such that all demand is satisfied. Take for example instance 1, Table 15 shows that the solution of the exact method contains two access locations equipped with two Wi-Fi, one SVC and one Alarm service access point. A graphical overview of this solution can be found in Figure 5. The locations marked with a star represent the access locations. One can see that the access location on the left is equipped with a Wi-Fi, SVC and Alarm service access point, since it forms the centre of their ranges. An additional access location on the right is needed for the second Wi-Fi service access point to satisfy the demand.

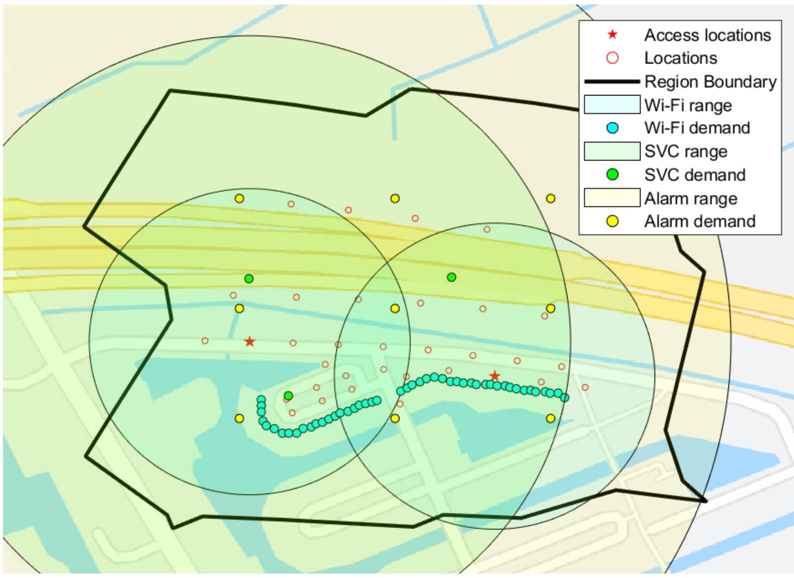


Figure 5: Graphical overview of the solution of the exact method on the MSCFLP for instance 1.

Total Problem

The best solution entails the lowest objective value as a minimisation problem is regarded. The exact method has the lowest objective value for all instances, which follows the expectation. Namely, the exact method solves the problem optimally if sufficient time is given. The heuristics differ in solution quality. For most instances, the objective values of the EPH and ESCH are close to the exact method. The ELRH generally has higher objective values. However, it has optimal solutions to instance 1 and 4, since the objective values are the same as the exact method. On average, the objective values of the EPH, the ELRH and the ESCH are respectively 5.4%, 8.6% and 3.5% above the exact method. This means that the ESCH is on average the best heuristic in terms of objective value. However, the EPH and ELRH have lower objective values for several instances.

The computation times in Table 6 are calculated as the sum of the computation times of the first and second phase in Table 15. Some computation times are marked with a dagger. This means that the solution method is terminated for this instance by the computation time limit as given in the Subsection 5.1.3. It is clear that the computation times of the exact method are much higher than the heuristics. However, more attention is given to the computation times of the heuristics, since the exact method only serves as a benchmark. The computation times of the heuristics are low, especially for the small instances. The large instances need five to ten minutes which is still short compared to the exact method. The ESCH generally needs more computation time which is explained below.

First Phase

The exact method consists of only one phase which is denoted as the second phase in the table. Therefore, this part only considers the first phase of the heuristics. The EPH and ELRH show relatively short computation times, whereas the ESCH needs longer computation times, especially for the large instances. The EPH has a greedy nature and the ELRH solves an LP relaxation in the first phase. Both can be executed quickly. The ESCH iteratively solves multiple SCP's by Lagrangian Relaxation, which obviously takes more time.

As said before, the solution of the first phase is indicated with the preposition intermediate. Only the intermediate access locations are passed to the restricted problem. In most instances, the number of intermediate access locations is the lowest for the ESCH followed by the EPH and the ELRH. Important to notice is that this is not necessarily beneficial. The number of intermediate access locations affects the size of the restricted problem in the second phase. Having more intermediate access locations gives more possible solutions to the restricted problem. On the other hand, the restricted problem is easier to optimise with less intermediate access locations. This can be observed by the optimality gaps in the second phase, which is addressed below.

Second Phase

This part contains the results of the exact method and the second phase of the heuristics. This includes the computation times, optimality gaps and the number of access locations and service access points in the final solution. Note that the final solution is by definition the solution of the second phase. The functioning of the stopping criteria can be concluded from the computation times. The second phases of the heuristics are terminated in instance 8 and 9. Apparently, the restricted problems in the second phase of the small instances are solvable within the maximum computation time. The exact method is terminated in instances 5-9. On the other hand, the first four instances are solved optimally by the exact method. The optimality gaps are calculated as stated in Equation 13. It is important to emphasise the relation between the computation time and optimality gap. The solution method is terminated if either the optimality gap equals zero or the maximum computation time is reached. This means that positive gaps exist, if and only if, the maximum computation time is reached. In general, the optimality gaps are low which means that sufficient computation time is given. The only outlier is the gap of the exact method in instance 5, which is caused by the symmetry of the problem.

The number of access locations and service access points are almost the same in the first five instances for all solution methods. To shorten the sentences, the term equipment is used to indicate the access locations and service access points. The small differences in the objective values are caused by the variations in the opening costs. Evidently, the exact method selects the cheapest access locations as it has the lowest objective values. In particular, the ELRH needs more equipment in the final solution, which results in higher objective values as observed before. The number of access locations is the most important indicator for the objective value, since the opening costs are higher than the equipping costs. As a result, the order of the solution methods based on the number of access locations is similar to the order based on the objective values. On the contrary, this order does not always apply to the number of service access points. An example is instance 8, in which the exact method needs more Alarm service access points than the heuristics. Apparently, it can be more beneficial to select cheap access locations than to minimise the number of service access points. Note that this is possible, since the difference in the opening costs can be equal to $5,000 - 4,000 = 1,000$, which is higher than the equipping costs of a service access point. Obviously, the choice of the parameters affects the structure of the solutions.

6.2 MSMCFLP Results - Solution Methods

In this subsection, the main focus is on the effects of the solution methods on the results of the MSMCFLP. The different values of the modularity parameter κ are taken as fixed. The analysis is organised in the same way as the analysis of the results on the MSCFLP. The summarised overviews of the computational results on the MSMCFLP with $\kappa = 3, 5, 10$ are given in Table 7, 8 and 9, respectively. The extensive overviews can be found in Table 16, 17 and 18 in Appendix A.2.

Table 7: Summarised overview of the computational results of the solution methods on the MSM-CFLP with $\kappa = 3$. The objective values of the heuristics comprise the relative differences to the exact method. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3.

	Exact		EPH		ELRH		ESCH	
No.	**Obj.**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**
1	5,298	0.3	0.0	0.0	72.5	0.0	0.0	0.0
2	14,226	6.7	2.1	0.0	30.1	0.0	0.0	0.0
3	19,528	21.1	22.0	0.1	21.6	0.3	0.0	0.0
4	30,787	41.9	15.0	0.1	24.1	0.3	2.6	0.3
5	23,974	115.0	16.5	0.0	16.3	0.3	15.2	0.0
6	86,335	552.6	23.8	0.2	21.8	0.3	5.2	0.2
7	108,113	3,600.3†	3.5	4.0	9.1	96.5	-6.9	1.5
8	915,121	43,211.6†	-9.6	316.9†	-5.7	315.6†	-15.1	61.5
9	1,123,715	43,209.8†	-5.9	325.4†	-1.8	313.2†	-15.8	422.8†

Table 8: Summarised overview of the computational results of the solution methods on the MSM-CFLP with $\kappa = 5$. The objective values of the heuristics comprise the relative differences to the exact method. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3.

	Exact		EPH		ELRH		ESCH	
No.	**Obj.**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**
1	5,298	0.3	0.0	0.0	72.5	0.1	0.0	0.0
2	14,226	8.7	2.1	0.0	30.1	0.1	0.0	0.0
3	19,528	37.8	0.1	0.1	21.6	0.9	0.0	0.1
4	27,017	189.7	16.1	0.1	27.0	0.2	2.0	0.3
5	23,974	122.1	16.5	0.0	16.3	0.3	15.2	0.0
6	86,335	711.3	23.8	0.1	21.8	0.5	5.2	0.2
7	123,908	3,605.1†	-9.9	1.8	-4.8	19.8	-25.0	1.3
8	903,732	43,218.8†	-10.8	316.3†	-6.0	313.3†	-20.1	345.3†
9	1,188,758	43,208.0†	-13.1	326.9†	-8.6	319.7†	-25.7	143.9

Table 9: Summarised overview of the computational results of the solution methods on the MSM-CFLP with $\kappa = 10$. The objective values of the heuristics comprise the relative differences to the exact method. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3.

	Exact		EPH		ELRH		ESCH	
No.	**Obj.**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**	**Obj. (%)**	**Time (s)**
1	5,298	0.3	0.0	0.0	72.5	0.1	0.0	0.0
2	14,226	8.9	2.1	0.0	0.0	0.4	0.0	0.0
3	19,528	29.9	0.1	0.1	21.6	5.1	0.0	0.1
4	27,017	137.0	1.1	0.0	15.4	4.4	2.0	0.3
5	23,974	123.3	16.5	0.1	16.3	0.9	15.2	0.0
6	86,335	1,025.2	23.8	0.1	4.3	3.3	5.2	0.2
7	120,241	3,602.4†	-13.5	0.3	-17.1	19.4	-22.7	1.1
8	919,707	43,219.7†	-12.6	317.2†	-17.2	313.5†	-22.3	51.3
9	1,408,962	43,208.2†	-28.2	330.8†	-29.5	314.3†	-37.8	419.7†

First of all, an example of the modular capacities is shown. For comparison to the MSCFLP, the solution of the exact method on the MSMCFLP with $\kappa = 3$ for instance 1 is taken. Table 16 shows that the optimal solution contains one access location which is equipped with two Wi-Fi, one SVC and one Alarm service access point. This is in contrast to the solution of the MSCFLP in which two access locations are needed. Apparently, the solution of the MSMCFLP utilises the possibility to equip an access locations with multiple service access points of the same service. This involves a cost reduction of $9,139 - 5,298 = 3,841$. A graphical overview of the solution can be found in Figure 6. There is only one access location which is the centre of the ranges of all service access points. Note that the Wi-Fi range is constructed by two Wi-Fi service access points which means there is sufficient capacity available.

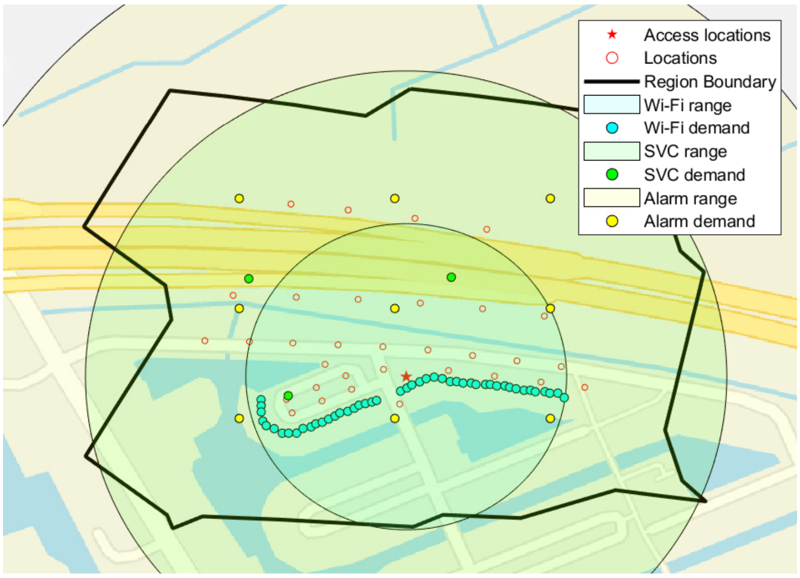


Figure 6: Graphical overview of the solution of the exact method on the MSMCFLP with $\kappa = 3$ for instance 1.

Total Problem

The tables show comparable behaviour for the different values of κ. The exact method always has the lowest objective values for the first six instances. The ESCH has the same objective values for the first three instances. Slightly higher objective values are obtained by the EPH. In general, the ELRH has the highest objective values, especially for the first instance. The lowest objective values of instance 7-9 are obtained by the ESCH. The average deviations of the objective values of the heuristics to the exact method are given in Table 10.

Table 10: Overview of the average deviations (%) of the objective values of the heuristics to the exact method.

	MSCFLP	MSMCFLP		
Heuristic	$\kappa = 1$	$\kappa = 3$	$\kappa = 5$	$\kappa = 10$
EPH	5.4	7.5	2.8	-1.2
ELRH	8.6	20.9	18.9	7.4
ESCH	3.5	-1.6	-5.4	-6.7

Concerning the MSMCFLP, one can see that the ESCH has on average lower objective values than the exact method. This is mainly caused by the good performance of the ESCH on instances 7-9. Although the exact method has the best solutions to the MSCFLP, the ESCH outperforms the exact method on the MSMCFLP. Concerning the MSMCFLP, the solution methods can be ordered in the following way based on the objective values: ESCH, exact method, EPH, ELRH.

The computation times show the same pattern for the different values of κ. The exact method optimally solves the first six instances before the computation time limit. The last three instances are terminated according to their computation time limits. The heuristics need short computation times for the first seven instances. Instance 7 is an exception, since the heuristics need short computation times, whereas the exact method shows difficulties solving this problem. The heuristics need more time for instance 8 and 9, which are terminated by the computation time limit in most cases.

First Phase

As said before, the exact method contains only one phase which is denoted as the second phase. Therefore, no computational results of the exact method exist for the first phase. The computational results of the heuristics show the same behaviour as for the MSCFLP. The computation times of the first phase are negligible for the first seven instances. The computation times of instance 8 and 9 are still relatively low. The ESCH needs more time than the EPH and ELRH, since it executes a more comprehensive algorithm in the first phase. The number of intermediate access locations shows the same pattern as in the MSCFLP. The ESCH has the least number of intermediate access locations, followed by the EPH and ELRH.

Second Phase

The optimality gaps are only positive when the solution methods are terminated by the computation time limit. The exact method solves the first six instances optimally for all values of κ. However, the gaps are high in the last three instances. Evidently, the exact method needs more time to find the optimal solution in these instances. The heuristics have small gaps in instance 8 and 9. This means that sufficient time is given to obtain a good solution to the restricted problem in the second phase of the heuristics. The size of the gaps is in proportion to the number of intermediate access locations, since this determines the size of the restricted problem in the second phase. Generally speaking, having less intermediate access locations results in smaller optimality gaps in the second phase.

The exact method consists of the optimal number of access locations and service access points in the final solutions of the first six instances, since these instances are solved optimally. The objective values of the solution methods are close to each other. Trivially, this also yields for the number of access locations and service access points. Again, the ESCH only needs slightly more equipment than the exact method. More equipment is generally needed for the EPH and ELRH. The opposite is true for instance 7-9 in which the ESCH generally needs the least number of access locations and service access points. As explained before, the same pattern was observed in the objective values.

6.3 MSMCFLP Results - Modular Capacities

This part focuses on the effects of the modular capacities on the computational results of the MSMCFLP. The modular capacities are determined by the value of the modularity parameter κ, which is taken equal to three, five and ten in this thesis. The MSCFLP is considered as the MSMCFLP with $\kappa = 1$ in this analysis.

Total Problem

As explained before, allowing modular capacities enlarges the solution space. Therefore, it is expected that the solutions improve as κ increases. This is true for the objective values of the heuristics, which decrease as κ increases for all instances. Especially, the step from the MSCFLP ($\kappa = 1$) to MSMCFLP with $\kappa = 3$ stipulates this. Take for example instance 1 in which the objective value is almost halved, since the number of access locations is reduced from two to one. The remaining steps to higher values of κ have less effect on the objective value. Apparently, only few access locations exist that are equipped with more than three service access points of the same service. In other words, the marginal effect of an additional service access point is decreasing in κ. For instance 1, the ELRH does not take advantage of the modular capacities, since the objective value stays constant. Evidently, not all demand points are located within the range of one intermediate access location. Therefore, two access locations are needed in the final solution. On the other hand, the exact method, EPH and ESCH make use of this opportunity. The objective value in instance 5 and 6 merely decreases for higher values of κ. Apparently, the sparse instances take less advantage of the modular capacities, because one service access point is usually sufficient to serve the demand points in its range. It is remarkable that the objective values of the exact method increase for $\kappa = 5, 10$ in instance 7-9. This is caused by the large optimality gaps in the second phase as explained below.

First Phase

In general, the computation times of the first phase are not affected by the value of κ. An exception is the ESCH for which the computation times of the first phase are lower in the MSMCFLP. This is caused by the construction of the algorithm as described in Subsection 5.4. It iteratively computes coverings to assign demand points to service access points. Less iterations are needed when multiple service access points are allowed on one access location.

For the EPH and ESCH, the number of intermediate access locations decreases as κ increases. The heuristics accommodate for the modular capacities as explained in Subsection 5.2 and 5.4, respectively. Concerning the ELRH, the number of intermediate access locations increases for $\kappa = 10$. As described in Subsection 5.3, the ELRH solves the LP relaxation in the first phase. Constraint 11a becomes less restricting for higher values of κ. The decision variables of the service access locations x_j^u are allowed to be ten times higher as the decision variables of the locations y_j. For example, a location j with $y_j = 0.1$ can be equipped with one service access point of every service in the LP relaxation. This results in many access locations in the solution of the LP relaxation. As a result, the computation times of the second phase are higher, which is addressed below.

Second Phase

The ELRH needs longer computation times in the second phase as κ increases, which can be explained by the increased number of intermediate access locations. The opposite is happening for the ESCH, which has shorter computation times in the second phase for higher values of κ. For this heuristic, the number of intermediate access locations is small, which makes the optimisation in the second phase easier. Consequently, this results in smaller optimality gaps. The value of κ also affects the computation times of the exact method. Instances 5 and 6 need shorter computation times for higher values of κ, which is caused by the sparseness of these instances. On the contrary, longer computation time are needed for instance 3 and 4, which is caused by the enlarged solution space. The optimality gaps of the exact method are larger for higher values of κ due to the enlarged solution space. More time is needed to find better solutions. Concerning the heuristics, the optimality gaps decrease for higher values of κ. The exception is the ELRH for $\kappa = 10$. The reason for this behaviour is the size of the restricted problem, which is caused by the large number of intermediate access locations as explained above.

As expected, the number of access locations in the final solution decreases when κ increases. This results in lower objective values as mentioned before. The exact method needs more access locations, which is caused by the large optimality gaps. The number of service access points slightly increases for higher values of κ. Equipping one access location with multiple service access points only decreases the number of access locations. More service access points are needed, since there are less access locations to place them on. It may be beneficial to save an access location while using more service access points. The large instances underline this theory. To illustrate this, the final solutions of the ESCH in instance 9 for $\kappa = 3$ and $\kappa = 10$ are compared. Sixteen access locations less are needed in the final solution for $\kappa = 10$. On the other hand, three Wi-Fi, three SVC and one Alarm service access point are additionally needed compared to the final solution for $\kappa = 3$.

6.4 Lower Bound Analysis

The lower bound indicates the lowest value that the objective value can possibly attain. The exact method iteratively updates the lower bound to tighten the gap between the lower bound and the current best feasible solution, also known as the incumbent. If the lower bound equals the incumbent, it is proven that the optimal solution is found. In the large instances, the results have shown that the optimal solution is not found. However, the lower bound gives an indication of the best possible objective value. In this way, it can be estimated how much room for improvement is left. The lower bound based on the exact method is also referred to as exact lower bound. The exact lower bounds can be found in Table 11.

Table 11: Overview of the exact lower bounds on the MSCFLP and MSMCFLP with different values of κ.

	MSCFLP	MSMCFLP		
No.	$\boldsymbol{\kappa = 1}$	$\boldsymbol{\kappa = 3}$	$\boldsymbol{\kappa = 5}$	$\boldsymbol{\kappa = 10}$
1	9,139	5,298	5,298	5,298
2	18,266	14,226	14,226	14,226
3	39,717	19,528	19,528	19,528
4	70,869	30,787	27,017	27,017
5	29,534	23,974	23,974	23,974
6	88,219	86,335	86,335	86,335
7	187,468	90,415	86,066	84,602
8	1,320,843	630,244	492,444	454,243
9	1,652,515	768,172	619,343	527,093

One can see that the exact lower bounds decrease when κ increases. This is caused in a direct and indirect way by the larger solution space for higher values of κ. A larger solution space directly decreases the lower bounds, since the optimal solution may be lower. Besides, a larger solution space affects the size of the optimality gaps. This can be seen in instance 7-9 on the MSMCFLP with $\kappa = 3, 5, 10$ in Table 16, 17 and 18, respectively. The large optimality gaps induce the lower bounds to be even lower. The gaps can be reduced by allowing longer computation times, which is practically not possible. On the other hand, lower bounds can be obtained by using the properties of the problem. More specifically, the minimum number of service access points and access locations can be argued. This is based on the following two methods:

- Coverage method: It can be determined how many service access points and access locations are needed to cover all demand points without taking into account the capacity restriction. This results in the MSLSCP. Note that the modularity parameter κ can be omitted, since the capacity restriction is removed. Efficient solutions of the MSLSCP can be obtained by using the method of Vos [44]. The solution contains a set of access locations and service access points which forms a minimum for the solution of the MSMCFLP.
- Capacity method: It can be determined how many service access points are needed to satisfy all demand without taking into account the coverage restriction. This problem can be referred to as the MSMCFLP in which all demand points can be served from every location. The number of service access points can be calculated by dividing the demand by the capacity of the corresponding service as stated in Equation 15.

  $$\text{Minimum number of service access points of service } u = \frac{\sum_{i \in \mathcal{G}^u} d_i^u}{\eta^u}. \tag{15}$$

  The number of access locations is equal to the maximum of the number of service access points divided by κ, since κ service access points of each service are allowed on every access location.

The minimum number of service access points is equal to the maximum of both methods, since the lower bound is preferred to be as high as possible. The minimum costs of the service access points can be calculated by multiplying the minimum number of service access points by the equipping costs of the corresponding service. The minimum costs of the access locations are more difficult to calculate, since the opening costs differ from each other. It can be calculated in the same manner as the minimum costs of the service access points by determining the minimum number of access locations. This can be multiplied by the lowest opening costs of the corresponding instance. On the other hand, the minimum costs can be directly obtained by calculating $\sum_{j \in \mathcal{L}} f_j y_j$, based on the MSLSCP solution. The minimum costs of the access locations are equal to the maximum of these two approaches. The lower bound is calculated as the sum of the minimum costs of service access points and access locations. This lower bound is referred to as argued lower bound. It is important to state that the argued lower bounds are not irrefutable, since the solutions to the MSLSCP in the coverage method are computed heuristically. However, they give a good indication of the potential improvement of the solutions. The argued lower bounds can be found in Table 12.

Table 12: Overview of the argued lower bounds of instance 7-9 on the MSCFLP and MSMCFLP.

No.	**MSCFLP**	**MSMCFLP**
7	185,225	92,068
8	1,194,631	701,395
9	1,494,949	865,055

Note that the MSMCFLP is not split for the different values of κ, since these results are identical. The reason for this is that the coverage method plays a dominant role in the MSMCFLP. As said before, the κ does not affect the coverage method which implies the identical results. On the contrary, the capacity method is restricting in the MSCFLP, since modular capacities are not allowed in this problem.

The argued lower bounds can be compared to the exact lower bounds as in Table 11. Concerning the MSCFLP, the argued lower bounds are lower than the exact lower bounds, since the optimality gaps are low in this problem. On the contrary, the argued lower bounds of the MSMCFLP are much higher than the exact lower bounds. This is due to the large optimality gaps of the exact method in this problem. The maximum of the exact and argued lower bound is defined as the implied lower bound. This can be compared to the objective values of the heuristics to acquire an indication of the performance. The implied optimality gap can be calculated to measure the difference between the objective values and the implied lower bounds.

Table 13: Overview of the average implied optimality gaps (%) of the solution methods on instance 7-9. The implied optimality gaps are calculated as in Equation 13. The lower bounds of the MSCFLP are obtained by the exact method and the lower bounds of the MSMCFLP by argumentation.

	MSCFLP	MSMCFLP		
Solution method	$\boldsymbol{\kappa = 1}$	$\boldsymbol{\kappa = 3}$	$\boldsymbol{\kappa = 5}$	$\boldsymbol{\kappa = 10}$
Exact	2.2	20.4	25.1	28.6
EPH	6.7	17.0	15.6	12.9
ELRH	5.5	20.8	19.9	9.4
ESCH	4.6	9.0	2.0	1.4

Regarding the MSCFLP, the table shows that the exact method slightly differs from the lower bound, followed by the ESCH, ELRH and EPH. However, the exact method has high gaps for the MSMCFLP. The ESCH has the best performance for the MSMCFLP, whereas the EPH and ELRH show comparable optimality gaps. To conclude, only small improvements can be made regarding the MSCFLP. The exact method entails the best solutions, followed closely by the ESCH. Concerning the MSMCFLP, the solutions of the exact method can be improved by circa 20% to 30%. The solutions of the EPH and ELRH can be improved by circa 10% to 20%. The ESCH has almost optimal results. However, the implied optimality gaps of the MSMCFLP are not irrefutable, since they are obtained by the heuristically computed argued lower bounds. Either new heuristics or longer computation times are needed to further improve the solutions.

6.5 Sensitivity Analysis

Solution methods usually have a trade-off between computation time and objective value. Longer computation times result in better solutions, but this is not always necessary. In practice, good solutions in short times may be preferable to optimal solutions in long times. For this reason, a sensitivity analysis is performed on the maximum computation time of the second phase of the heuristics. In the solution approach in Section 5, the maximum computation time is set equal to five minutes. To keep it concise, the analysis is only performed on the MSCFLP. The computational results on the MSCFLP in Table 15 show that positive optimality gaps exist in instance 8 and 9. Therefore, the analysis is only performed on these two instances for the MSCFLP.

The experiment is conducted for the three heuristics and is set up in the following way. First of all, the first phase of the heuristic is executed. Thereafter, the second phase is performed with the maximum computation time ranging from one minute to ten minutes. Every maximum computation time entails an optimality gap, which can be found in Table 14.

Table 14: Overview of the optimality gaps (%) of the second phase of the heuristics for different computation time limits. The considered problems are instance 8 and 9 of the MSCFLP.

	Instance 8			Instance 9		
# Minutes	**EPH**	**ELRH**	**ESCH**	**EPH**	**ELRH**	**ESCH**
1	0.92	1.48	0.53	1.26	1.66	1.30
2	0.51	1.15	0.40	1.02	1.66	0.11
3	0.45	0.70	0.39	0.67	1.36	0.05
4	0.43	0.64	0.37	0.67	1.36	0.05
5	0.42	0.64	0.34	0.67	1.16	0.02
6	0.42	0.64	0.34	0.67	1.06	0.02
7	0.39	0.64	0.33	0.67	1.06	0.02
8	0.37	0.64	0.31	0.67	1.01	0.02
9	0.37	0.64	0.31	0.60	1.01	0.02
10	0.37	0.62	0.31	0.60	1.01	0.02

The observant reader may point out that the optimality gaps after five minutes slightly differ from the gaps stated in Table 15. It is assumed that this is caused by the minor differences in the technical performance of the laptop. On average, the gaps of instance 8 are lower than the gaps of instance 9. This is caused by the size of the problem. The exact method generally needs more time on larger problems, since they contain larger solution spaces. Figures 7 and 8 graphically show the development of the optimality gaps as function of the maximum computation times for instance 8 and 9, respectively.

Trivially, the optimality gaps are decreasing in the maximum computation time. As stated in Subsection 6.1, the smallest gaps are obtained by the ESCH, followed by the EPH and the ELRH. This pattern stays the same for all maximum computation times, since the lines are not crossing each other. One can see that the largest improvement is usually made in the first minutes. The best example is the ESCH on instance 9, where the gap drops from 1.30% to 0.11% in the second minute. Thereafter, only minor improvements are made. In other words, the marginal effect of an additional minute is decreasing over time. The lines in Figure 7 converge more than the lines in Figure 8. Apparently, more comparable solutions are found in the second phase of the heuristics for smaller problems.

It is difficult to state an overall conclusion, since users have different priorities for the objective value and computation time. However, it can be seen that only minor improvements are made after four minutes for instance 8. This boundary is approximately five minutes for instance 9. It can be concluded that the choice of five minutes in the solution approach in Section 5 is substantiated. However, more comprehensive stopping criteria may be more suitable when the algorithm is applied to larger instances. These criteria can be based on the size of the problem or number of iterations without improvement.

Figure 7: Graphical representation of the optimality gaps as function of the maximum computation time for instance 8 on the MSCFLP.

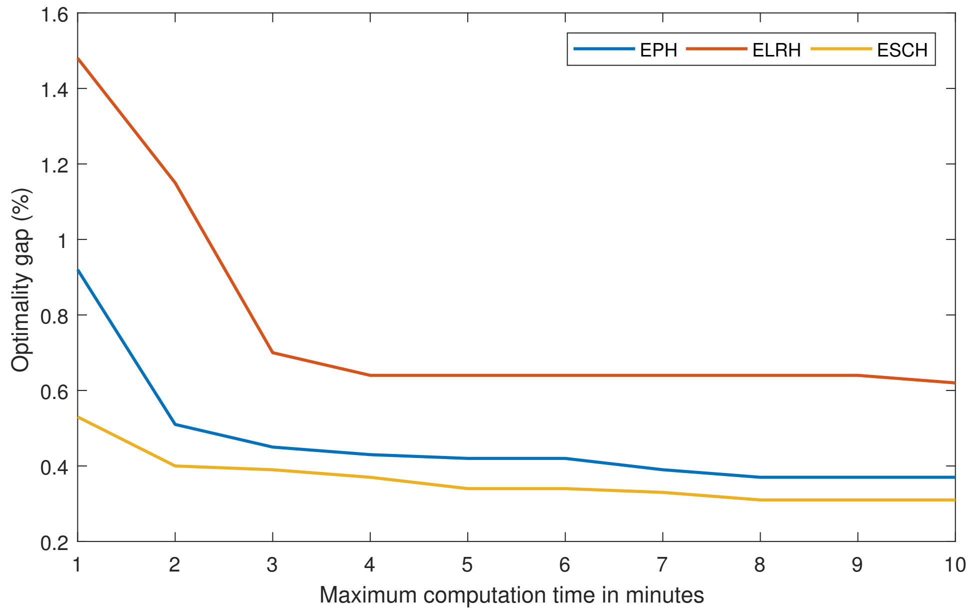
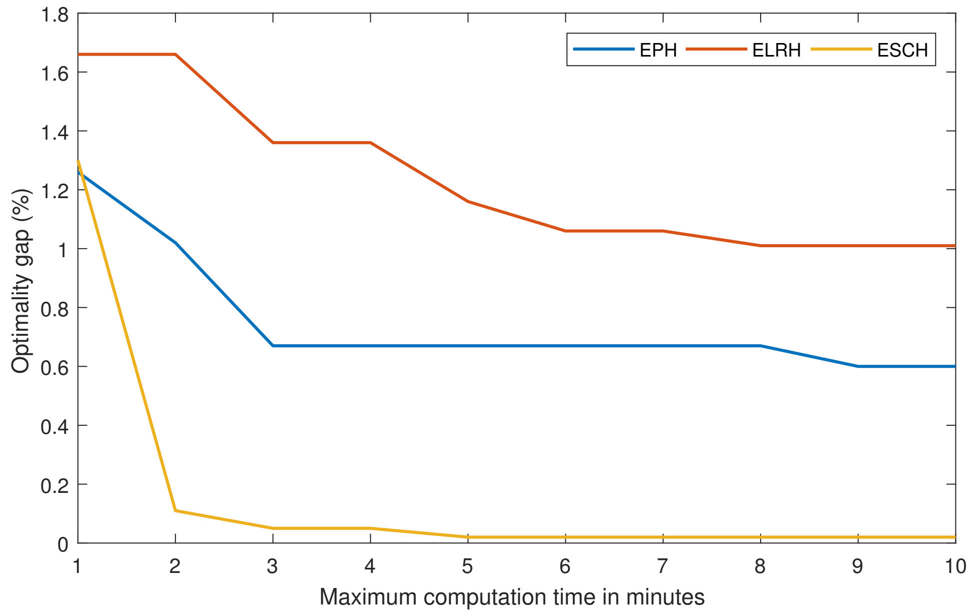


Figure 8: Graphical representation of the optimality gaps as function of the maximum computation time for instance 9 on the MSCFLP.



7 Conclusion

This thesis considers the problem of distributing services over urban areas to satisfy the demand. In the smart city context, the demand is spread out over the city. The demand can be covered by equipping an access location with a service access point of the corresponding service. Costs savings can be gained by combining service access points of different services on the same access location. Three heuristics are introduced, each consisting of two phases. In the first phase, a feasible solution is found which is optimised in the second phase by an exact method. The heuristics are benchmarked to the exact method on the whole problem. For this, nine test instances are used that differ in size and density. First of all, the MSCFLP is regarded which allows only one service access point of every service on an access location. Besides, the MSMCFLP is introduced which allows modular capacities. In other words, multiple service access points of every service are allowed on an access location. The modularity parameter κ is taken equal to three, five and ten.

The computational results on the MSCFLP show that the exact method has the lowest objective values, followed by the ESCH, EPH and ELRH. However, this involves long computation times for the exact method, since the performance of the exact method is mainly dependent on the size of the instances. On the other hand, the heuristics have similar computation times of at most ten minutes, even for large instances. The exact method has relatively small optimality gaps, but even smaller optimality gaps exist for the heuristics. The optimality gaps of the heuristics are affected by the number of access locations in the solution of the first phase, which form the restricted problem in the second phase. The objective values are mainly affected by the number of access locations in the final solution, since the opening costs are much higher than the equipping costs. Comparable behaviour of the solution methods is observed on the MSMCFLP. In general, the objective values decrease as κ increases due to the enlarged solution space. This improvement in objective value is mainly present for small values of κ. The more service access points on one access location are allowed, the smaller is the marginal effect of an additional service access point. The low objective values are caused by the decreased number of access locations. The number of service access points is comparable for all values of κ. The heuristics show similar computation times and smaller optimality gaps compared to the MSCFLP. On the contrary, the exact method has larger optimality gaps for the large instances. Consequently, the ESCH outperforms the exact method on the MSMCFLP.

A lower bound analysis is conducted to indicate the solution quality and the potential that is left for new heuristics. Lower bounds are obtained by the exact method and argumentation. The exact lower bounds are higher for the MSCFLP, whereas the argued lower bounds are higher for the MSMCFLP. The difference between the objective values and lower bounds are small for both problems. However, the argued lower bounds are not irrefutable, since they are computed heuristically. Summarised, almost optimal solutions exist for both problems. Only little room is left for improvements in terms of objective value. Of course, minimisation of the computation time can always be pursued. A sensitivity analysis is performed to measure the effect of the maximum computation time on the objective values of the heuristics on the MSCFLP. The analysis showed that the marginal effect of additional computation time decreases over time. The maximum computation times of five minutes in the second phase of the heuristics is shown to be substantiated.

To conclude, good solutions in short times are found for the MSCFLP and MSMCFLP. In terms of solution quality, the best heuristic is the ESCH, followed by the EPH and ELRH. For the MSMCFLP, the ESCH even outperforms the exact method. Besides, it can be concluded that allowing modular capacities generally leads to large costs benefits.

8 Discussion

In this section, the limitations of this thesis and suggestions for further research are discussed.

8.1 Limitations

This thesis contains two limitations: the circular coverage area of the service access points and the variation of the heuristics. An attempt is made to find more heuristics based on Lagrangian Relaxation. Besides, the dual formulation is derived and the literature is researched for metaheuristics. Unfortunately, these approaches did not succeed.

8.1.1 Circular Coverage Area

It is assumed that the coverage areas of the service access points are circular. This is generally not true for the Wi-Fi service which signal is dependent on the infrastructure. Besides, the height of the demand points is not taken into account. Currently, only the haversine distance between demand points and locations plays a role in determining the coverage elements a_{ij}^u. A solution for this is to check which demand points can be reached from every location. Practically, this can be done by a propagation grid, which indicates the real ranges of service access points on a map. In this way, the coverage elements a_{ij}^u realistically represent the possible connections between locations and demand points.

8.1.2 Lagrangian Relaxation

An attempt is made to employ Lagrangian Relaxation in a heuristic for the SSCFLP, which can be executed sequentially to solve the MSCFLP. Different articles have been studied, of which the relaxation of the assignment constraint seemed to be the most promising [37, 25, 6]. The relaxation of the capacity constraint has weaker performance [37] and may be viable [15]. The relaxation of the assignment constraint results in multiple knapsack problems for which efficient methods exist. In literature, the Lagrangian heuristics solve the SSCFLP containing connection costs c_{ij}, which are not present in the formulation of the MSCFLP. An extension could be to include connection costs c_{ij}^u based on the distance between locations and demand points. Take for example the Wi-Fi service, the intuition would be that the connection is better when the distance is smaller. An objective of the algorithm would be to minimise the distances. If locations serve demand points outside their range, high connection costs are involved such that the algorithm tries to avoid this. The mathematical formulation of the connection costs can be found in Equation 16.

$$c_{ij}^u = \begin{cases} \alpha^u \pi_{ij} & \text{if } a_{ij}^u = 1, \\ C & \text{if } a_{ij}^u = 0. \end{cases} \tag{16}$$

The distance in meters between demand point $i \in \mathcal{G}^u$ and location $j \in \mathcal{L}$ is given by π_{ij}. A factor α^u can be used to scale the distance such that the connection costs are proportional to the opening and equipping costs. The constant C should be larger than the maximum of the opening and equipping costs such that it is always better to open an additional location than to serve a demand point from a location with $a_{ij}^u = 0$. As said before, this approach did not succeed. This was mainly caused by the difficulty of the *subgradient optimisation*, i.e. the method for updating the Lagrange multipliers. Besides, the symmetry of the problem gave several complications.

8.1.3 Dual Formulation

Problems can be considered from different perspectives. The dual formulation approaches the problem from the opposite perspective as the primal formulation. In some cases, the dual formulation is easier to solve than the primal one. Besides, the solution of the dual formulation forms a lower bound for minimisation problems. This is also called weak duality [40]. However, experiments have shown that these lower bounds are less restricting than the obtained lower bounds in Subsection 6.4. For LP problems, strong duality can be proven which means that an optimal solution of the primal is also an optimal solution of the dual. Unfortunately, this does not yield for ILP problems.

8.1.4 Metaheuristics

Metaheuristics can be applied to different kinds of problems due to their flexibility. For this, the parameters can be adjusted to the problem. However, incorporating problem-specific behaviour in heuristics may be more beneficial. Several metaheuristics exist to solve the SSCFLP, which are stated in Section 3. It is investigated whether their performance is sufficient for the smart city context. This means that large problems can be solved within minutes.

In most articles [19, 38, 26, 16, 11], the metaheuristics are only tested on small instances. The instances of Delmaire et al. [16] and Holmberg et al. [28] are often used as a benchmark. The first set contains 57 instances, in which the number of locations range from 10 to 30 and the number of demand points range from 20 to 90. The second set includes 71 instances with more demand points. The metaheuristics still need tens of seconds or minutes to solve the problems. It is clear that these metaheuristics cannot be used for large problems. Nevertheless, three articles [4, 23, 41] have been found that solve larger instances containing 1,000 locations and demand points. However, these metaheuristics need computation times of 5,000 to 10,000 seconds. For comparison, this thesis contains instances up to 6,981 locations and 11,081 demand points for three different services which are solved within minutes. To conclude, even the best metaheuristics available are not suited for the large problems in the smart city context.

8.2 Suggestions for Further Research

There are many suggestions for further research which are stated below.

- Test instances can be made of entire cities. Consequently, the results are better scalable to the smart city context. However, the computation times will increase due to the larger solution space. For this, the city can be divided in subareas, which are separately solved. An algorithm can be made to make the division and combine the separate solutions. Intermediate costs updating can be used to avoid inefficiency.

- The demand of the test instances can be made more realistic. In this thesis, home addresses are considered as Wi-Fi demand points. However, Wi-Fi may also be needed on streets and in parks. Besides, the SVC demand points are currently not representing all the roads. Finally, the Alarm demand points are equally spread over the area. It can be the case that the Alarm service access points cover all demand points, but not the entire area. More demand points can be added to avoid this.

- More services can be added to make the problem more realistic. In the smart city context, many services may exist that need different connections, like fiber or ether. Groups of services can be made to distinguish between these different connections. This extension makes the problem more difficult, since the optimisation has to be executed over multiple groups.

- A restriction on the combination of service access points of different services on one access location can be included. An example of this can be found in Equation 17.

$$\sum_{u \in \mathcal{F}} \gamma^u x_j^u \leq \tau_j y_j, \qquad \forall j \in \mathcal{L}. \tag{17}$$

  The parameters can be chosen based on practical reasons. For example, either a large service or two small services can be placed on an access location. In that case, the γ^u of the large service should be twice as large as the one of the small service. Furthermore, note that for consistency reasons the following should hold:

$$\left\lfloor \frac{\tau_j}{\gamma^u} \right\rfloor \leq \kappa_j^u, \qquad \forall j \in \mathcal{L}, \forall u \in \mathcal{F}. \tag{18}$$

  By this construction, Constraint 11a becomes redundant and can be removed. A sensitivity analysis could be performed on the parameters to obtain the effect on the objective value.

- Synergies may be obtained by combining the results of this thesis and the previous ones. Every thesis extends the problem in a certain way. All these extensions can be combined to make the problem more realistic. The demand can be modelled stochastically as in Verhoek [43] and the problem may allow the Partial Covering of Hoekstra [27]. Moreover, the modular capacities introduced in this thesis and the Wireless Network Problem (WNP) of Vos [44] may be used. The WNP can be seen as the final phase in which the lampposts are connected to an existing network.

- This thesis focuses on the theoretical objective values and computation times of the solution. However, attention can also be given to the robustness, which is a measure for the practical feasibility of solutions. For example, the effects of a defect lamppost can be measured. Simulations can be conducted to measure the number of unserved demand points in case of defect lampposts. This phenomenon can be incorporated in the objective function to maximise the robustness.

References

[1] T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *Operations Research Letters*, 2005. ISSN 01676377. doi: 10.1016/j.orl.2004.04.002.

[2] M. C. Agar and S. Salhi. Lagrangean heuristics applied to a variety of large capacitated plant location problems. *Journal of the Operational Research Society*, 1998. ISSN 14769360. doi: 10.1057/palgrave.jors.2600621.

[3] A. Ahmadi-Javid, P. Seyedi, and S. S. Syam. A survey of healthcare facility location, 2017. ISSN 03050548.

[4] R. K. Ahuja, J. B. Orlin, S. Pallottino, M. Scaparra, and M. G. Scutellà. A Multi-Exchange Heuristic for the Single Source Capacitated Facility Location Problem. *SSRN Electronic Journal*, 2005. doi: 10.2139/ssrn.337621.

[5] E. Balas, S. Ceria, and G. Cornuéjols. Mixed 0-1 Programming by Lift-and-Project in a Branch-and-Cut Framework. *Management Science*, 2008. ISSN 0025-1909. doi: 10.1287/mnsc.42.9.1229.

[6] J. Barceló and J. Casanovas. A heuristic lagrangean algorithm for the capacitated plant location problem. *European Journal of Operations Research*, 15:212–226, 1984.

[7] J. E. Beasley. A lagrangian heuristic for set-covering problems. *Naval Research Logistics*, 37(1):151–164, 1990.

[8] J. E. Beasley. Lagrangean heuristics for location problems. *European Journal of Operational Research*, 1993. ISSN 03772217. doi: 10.1016/0377-2217(93)90118-7.

[9] A. Caprara and M. Fischetti. {0, 1/2}-Chvátal-Gomory cuts. *Mathematical Programming, Series B*, 1996. ISSN 00255610. doi: 10.1007/BF02592196.

[10] S. Ceria, P. Nobili, and A. Sassano. A Lagrangian-based heuristic for large-scale set covering problems. *Mathematical Programming*, 81:215–228, 1998.

[11] I. A. Contreras and J. A. Díaz. Scatter search for the single source capacitated facility location problem. *Annals of Operations Research*, 2008. ISSN 02545330. doi: 10.1007/s10479-007-0193-1.

[12] G. Cornuéjols, R. Sridharan, and J. M. Thizy. A comparison of heuristics and relaxations for the Capacitated Plant Location Problem. *European Journal of Operational Research*, 50: 280–297, 1991.

[13] I. Correia and M. E. Captivo. A Lagrangean Heuristic for a Modular Capacitated Location Problem. *Annals of Operations Research*, 122:141–161, 2003.

[14] G. B. Dantzig. *Linear Programming 2: Theory and Extensions*. Springer, 2002. ISBN 978-0691080000. doi: 10.1287/opre.50.1.42.17798.

[15] K. Darby-Downman and H. Lewis. Lagrangian Relaxation and the Single-Source Capacitated Facility-Location Problem. *Journal of Operations Research Society*, 39(11):1035–1040, 1988.

[16] H. Delmaire, J. A. Díaz, E. Fernández, and M. Ortega. Reactive Grasp And Tabu Search Based Heuristics For The Single Source Capacitated Plant Location Problem. *INFOR: Information Systems and Operational Research*, 1999. ISSN 0315-5986. doi: 10.1080/03155986.1999.11732381.

[17] R. Dileep. Simple Methods for Uncapacitated Facility Location/Allocation Problems. *Journal of Operations Management*, 1(4), 1981.

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

[19] V. Filho and R. Galvão. A tabu search heuristic for the concentrator location problem. *Location Science*, 6:189–209, 1998.

[20] S. L. Gadegaard, A. Klose, and L. R. Nielsen. An improved cut-and-solve algorithm for the single-source capacitated facility location problem. *EURO Journal on Computational Optimization*, 6(1):1–27, 2018. ISSN 21924414. doi: 10.1007/s13675-017-0084-4.

[21] M. R. Garey and D. S. Johnson. *Computers and intractability : a guide to the theory of NP-completeness*. W.H. Freeman & Co., 1979.

[22] R. E. Gomory and W. J. Baumol. Integer Programming and Pricing. *Econometrica*, 2006. ISSN 00129682. doi: 10.2307/1910130.

[23] G. Guastaroba and M. G. Speranza. A heuristic for BILP problems: The Single Source Capacitated Facility Location Problem. *European Journal of Operational Research*, 2014. ISSN 03772217. doi: 10.1016/j.ejor.2014.04.007.

[24] S. Guha and S. Khuller. Greedy Strikes Back: Improved Facility Location Algorithms. *Journal of Algorithms*, 31:228–248, 1999.

[25] K. S. Hindi and K. Pieńkosz. Efficient solution of large scale, single-source, capacitated plant location problems. *Journal of the Operational Research Society*, 50:268–274, 1999.

[26] S. C. Ho. An iterated tabu search heuristic for the Single Source Capacitated Facility Location Problem. *Applied Soft Computing*, 27:169–178, 2015.

[27] G. Hoekstra. Location Assignment of Capacitated Services in Smart Cities. Master's thesis, University of Groningen, 2018.

[28] K. Holmberg, M. Rönnqvist, and D. Yuan. An exact algorithm for the capacitated facility location problems with single sourcing. *European Journal of Operational Research*, 1999. ISSN 03772217. doi: 10.1016/S0377-2217(98)00008-3.

[29] R. G. Jeroslow. Trivial integer programs unsolvable by branch-and-bound. *Mathematical Programming*, 1974. ISSN 0025-5610. doi: 10.1007/bf01580225.

[30] D. S. Johnson, C. H. Papadimitriou, and K. Steiglitz. Combinatorial Optimization: Algorithms and Complexity. *The American Mathematical Monthly*, 2006. ISSN 00029890. doi: 10.2307/2322374.

[31] J. G. Klincewicz and H. Luss. A Lagrangian Relaxation Heuristic for Capacitated Facility Location with Single-Source Constraints. *Journal of Operational Research society*, 37:495–500, 1986.

[32] B. Maheshwari, V. P. Singh, and B. Thoradeniya. *Balanced Urban Development: Options and Strategies for Liveable Cities*. Springer, 2016. ISBN 978-3-319-28110-0. doi: 10.1007/978-3-319-28112-4.

[33] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 2002. ISSN 0166218X. doi: 10.1016/S0166-218X(01)00348-1.

[34] J. E. Mitchell. Branch-and-Cut Algorithms for Combinatorial Optimization Problems. In *Handbook of applied optimization*. Oxford University Press, 1999.

[35] A. Montoya, M. C. Vélez-Gallego, and J. G. Villegas. Multi-product capacitated facility location problem with general production and building costs. *Netnomics*, 17:47–70, 2016.

[36] R. M. Nauss. An Improved Algorithm for the Capacitated Facility Location Problem. *Journal of the Operational Research Society*, 19(12):1195–1201, 1978.

[37] H. Pirkul. Efficient Algorithms for the capacitated concentrator problem. *Computer Operations Research*, 14:197–208, 1987.

[38] M. Rönnqvist, S. Tragantalerngsak, and J. Holt. A repeated matching heuristic for the single-source capacitated facility location problem. *European Journal of Operational Research*, 116:51–68, 1999.

[39] W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill Education, 1976. ISBN 978-0070542358.

[40] H. Tijms. *Operationele Analyse*. Epsilon Uitgeverij, 2010.

[41] T. H. Tran, M. P. Scaparra, and J. R. O'Hanley. A hypergraph multi-exchange heuristic for the single-source capacitated facility location problem. *European Journal of Operational Research*, 2017. ISSN 03772217. doi: 10.1016/j.ejor.2017.04.032.

[42] UN. The Sustainable Development Goals Report. Technical report, United Nations, 2018.

[43] M. Verhoek. Optimising the placement of access points for smart city services with stochastic demand. Master's thesis, University of Groningen, 2017.

[44] T. J. C. Vos. Using Lampposts to Provide Urban Areas with Multiple Services. Master's thesis, Erasmus School of Economics, 2016.

[45] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 2014. ISSN 23274662. doi: 10.1109/JIOT.2014.2306328.

A Tables of Computational Results

This section contains the tables of the computational results of the solution methods. First of all, the computational results of the MSCFLP are given in Subsection A.1. Thereafter, the results of the MSMCFLP with the different values of κ are given in Subsection A.2.

A.1 MSCFLP

Table 15: Overview of the computational results of the solution methods on the MSCFLP. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3. AL indicates the number of access locations, whereas Wi-Fi, SVC and Alarm indicate the number of service access points of the corresponding service.

			First phase		Second phase					
	No.	Objective	Time (s)	AL	Time (s)	Gap (%)	AL	Wi-Fi	SVC	Alarm
Exact	1	9,139	-	-	0.1	0.00	2	2	1	1
	2	18,266	-	-	20.4	0.00	4	4	2	1
	3	39,717	-	-	3.0	0.00	9	9	1	1
	4	70,869	-	-	3.1	0.00	16	16	2	1
	5	31,788	-	-	3,600.5	7.09	7	7	3	2
	6	90,574	-	-	3,600.6	2.60	19	11	18	11
	7†	190,865	-	-	3,600.2	1.78	43	43	6	4
	8†	1,346,700	-	-	43,202.1	1.92	303	298	57	34
	9†	1,699,594	-	-	43,205.4	2.77	382	362	77	44
EPH	1	9,326	0.0	2	0.0	0.00	2	2	1	1
	2	18,433	0.0	4	0.0	0.00	4	4	2	1
	3	39,949	0.0	12	0.3	0.00	9	9	1	1
	4	75,949	0.0	17	0.3	0.00	16	16	2	1
	5	32,020	0.0	9	0.1	0.00	7	7	3	2
	6	111,042	0.0	27	0.1	0.00	24	12	19	11
	7	195,113	0.1	48	1.9	0.00	43	43	7	4
	8†	1,433,498	15.4	362	300.3	0.41	315	305	59	32
	9†	1,803,252	25.9	457	300.5	0.64	398	370	79	42
ELRH	1	9,139	0.1	3	0.0	0.00	2	2	1	1
	2	22,557	0.1	6	0.1	0.00	5	5	2	1
	3	43,965	0.2	12	0.5	0.00	10	10	1	1
	4	70,869	0.1	19	0.3	0.00	16	16	2	1
	5	35,625	0.2	12	0.1	0.00	8	7	3	2
	6	109,134	0.2	38	0.2	0.00	24	12	18	12
	7	194,034	0.5	50	2.3	0.00	44	44	6	5
	8†	1,406,526	24.4	417	300.5	0.63	318	311	58	31
	9†	1,775,179	26.8	526	300.3	1.36	401	375	79	46
ESCH	1	9,326	0.2	2	0.0	0.00	2	2	1	1
	2	18,279	0.1	4	0.1	0.00	4	4	2	1
	3	39,842	0.2	9	0.1	0.00	9	9	1	1
	4	73,720	0.4	17	0.3	0.00	16	16	2	1
	5	35,772	0.1	8	0.0	0.00	8	7	3	2
	6	94,850	0.1	20	0.1	0.00	20	10	17	14
	7	195,050	1.7	46	3.0	0.00	43	43	6	4
	8†	1,381,724	84.0	329	300.4	0.37	306	297	59	30
	9†	1,748,761	213.6	407	300.8	0.05	389	366	76	43

A.2 MSMCFLP

Table 16: Overview of the computational results of the solution methods on the MSMCFLP with $\kappa = 3$. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3. AL indicates the number of access locations, whereas Wi-Fi, SVC and Alarm indicate the number of service access points of the corresponding service.

			First phase		Second phase					
	No.	**Objective**	**Time (s)**	**AL**	**Time (s)**	**Gap (%)**	**AL**	**Wi-Fi**	**SVC**	**Alarm**
Exact	1	5,298	-	-	0.3	0.00	1	2	1	1
	2	14,226	-	-	6.7	0.00	3	4	2	1
	3	19,528	-	-	21.1	0.00	4	9	1	1
	4	30,787	-	-	41.9	0.00	6	16	2	2
	5	23,974	-	-	115.0	0.00	5	7	3	2
	6	86,335	-	-	552.6	0.00	18	11	16	12
	7†	108,113	-	-	3,600.3	16.37	22	45	7	4
	8†	915,121	-	-	43,211.6	31.13	190	331	65	45
	9†	1,123,715	-	-	43,209.8	31.64	237	375	86	52
EPH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,521	0.0	3	0.0	0.00	3	4	2	1
	3	23,829	0.0	7	0.1	0.00	5	10	1	1
	4	35,404	0.0	8	0.1	0.00	7	17	2	1
	5	27,935	0.0	6	0.0	0.00	6	7	3	2
	6	106,868	0.0	26	0.2	0.00	23	12	19	11
	7	111,870	0.1	27	3.9	0.00	23	45	7	5
	8†	827,050	16.6	201	300.3	0.15	171	309	60	30
	9†	1,056,926	25.0	255	300.4	0.20	219	372	78	41
ELRH	1	9,139	0.0	3	0.0	0.00	2	2	1	1
	2	18,503	0.0	5	0.0	0.00	4	5	2	1
	3	23,741	0.1	10	0.2	0.00	5	10	1	1
	4	38,200	0.1	12	0.2	0.00	8	16	2	1
	5	27,870	0.2	12	0.1	0.00	6	8	3	2
	6	105,134	0.2	38	0.1	0.00	23	12	18	12
	7	117,915	0.4	39	96.1	0.00	25	46	7	5
	8†	862,747	15.3	320	300.3	0.40	183	318	60	31
	9†	1,103,859	12.7	395	300.5	0.13	235	378	80	44
ESCH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,226	0.0	3	0.0	0.00	3	4	2	1
	3	19,528	0.0	4	0.0	0.00	4	9	1	1
	4	31,582	0.2	7	0.1	0.00	6	16	2	2
	5	27,629	0.0	6	0.0	0.00	6	7	3	2
	6	90,839	0.1	19	0.1	0.00	19	10	17	14
	7	100,650	1.1	22	0.4	0.00	20	45	7	4
	8	776,769	47.0	165	14.5	0.00	158	303	60	30
	9†	946,693	122.5	200	300.3	0.07	192	368	79	43

Table 17: Overview of the computational results of the solution methods on the MSMCFLP with $\kappa = 5$. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3. AL indicates the number of access locations, whereas Wi-Fi, SVC and Alarm indicate the number of service access points of the corresponding service.

			First phase		Second phase					
	No.	**Objective**	**Time (s)**	**AL**	**Time (s)**	**Gap (%)**	**AL**	**Wi-Fi**	**SVC**	**Alarm**
Exact	1	5,298	-	-	0.3	0.00	1	2	1	1
	2	14,226	-	-	8.7	0.00	3	4	2	1
	3	19,528	-	-	37.8	0.00	4	9	1	1
	4	27,017	-	-	189.7	0.00	5	16	2	2
	5	23,974	-	-	122.1	0.00	5	7	3	2
	6	86,335	-	-	711.3	0.00	18	11	16	12
	7†	123,908	-	-	3,605.1	30.54	25	44	8	5
	8†	903,732	-	-	43,218.8	45.51	188	309	67	39
	9†	1,188,758	-	-	43,208.0	47.90	249	382	93	55
EPH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,521	0.0	3	0.0	0.00	3	4	2	1
	3	19,548	0.0	5	0.1	0.00	4	9	1	1
	4	31,361	0.0	7	0.1	0.00	6	16	2	1
	5	27,935	0.0	6	0.0	0.00	6	7	3	2
	6	106,868	0.0	26	0.1	0.00	23	12	19	11
	7	111,625	0.1	24	1.7	0.00	23	45	7	5
	8†	806,518	16.1	184	300.2	0.10	165	312	62	30
	9†	1,033,239	26.6	223	300.3	0.05	213	372	81	42
ELRH	1	9,139	0.0	3	0.1	0.00	2	2	1	1
	2	18,503	0.1	5	0.0	0.00	4	5	2	1
	3	23,741	0.1	10	0.8	0.00	5	10	1	1
	4	34,300	0.1	12	0.1	0.00	7	16	2	2
	5	27,870	0.2	12	0.1	0.00	6	8	3	2
	6	105,134	0.3	37	0.2	0.00	23	12	18	12
	7	117,915	0.5	38	19.3	0.00	25	46	7	5
	8†	849,274	13.0	319	300.3	0.07	180	314	60	30
	9†	1,087,098	19.3	389	300.4	0.13	231	378	79	43
ESCH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,226	0.0	3	0.0	0.00	3	4	2	1
	3	19,528	0.0	4	0.1	0.00	4	9	1	1
	4	27,563	0.2	5	0.1	0.00	5	16	2	2
	5	27,629	0.0	6	0.0	0.00	6	7	3	2
	6	90,839	0.1	19	0.1	0.00	19	10	17	14
	7	92,968	1.1	18	0.2	0.00	18	45	8	4
	8†	722,300	45.1	145	300.2	0.04	144	308	60	30
	9	883,783	120.4	179	23.5	0.00	176	370	81	44

Table 18: Overview of the computational results of the solution methods on the MSMCFLP with $\kappa = 10$. Instances marked with a † are terminated according to the stopping criteria as explained in Subsection 5.1.3. AL indicates the number of access locations, whereas Wi-Fi, SVC and Alarm indicate the number of service access points of the corresponding service.

			First phase		Second phase					
	No.	**Objective**	**Time (s)**	**AL**	**Time (s)**	**Gap (%)**	**AL**	**Wi-Fi**	**SVC**	**Alarm**
Exact	1	5,298	-	-	0.3	0.00	1	2	1	1
	2	14,226	-	-	8.9	0.00	3	4	2	1
	3	19,528	-	-	29.9	0.00	4	9	1	1
	4	27,017	-	-	137.0	0.00	5	16	2	2
	5	23,974	-	-	123.3	0.00	5	7	3	2
	6	86,335	-	-	1,025.2	0.00	18	11	16	12
	7†	120,241	-	-	3,602.4	29.64	25	45	7	5
	8†	919,707	-	-	43,219.7	50.61	193	310	67	38
	9†	1,408,962	-	-	43,208.2	62.59	301	387	97	63
EPH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,521	0.0	3	0.0	0.00	3	4	2	1
	3	19,548	0.0	5	0.1	0.00	4	9	1	1
	4	27,306	0.0	5	0.0	0.00	5	16	2	1
	5	27,935	0.0	6	0.1	0.00	6	7	3	2
	6	106,868	0.0	26	0.1	0.00	23	12	19	11
	7	104,061	0.1	21	0.2	0.00	21	44	7	5
	8†	803,564	17.0	175	300.2	0.08	165	309	60	32
	9†	1,011,979	30.6	211	300.2	0.09	208	374	80	43
ELRH	1	9,139	0.0	3	0.1	0.00	2	2	1	1
	2	14,226	0.1	19	0.3	0.00	3	4	2	1
	3	23,741	0.1	21	5.0	0.00	5	10	1	1
	4	31,188	0.1	27	4.3	0.00	6	16	2	2
	5	27,870	0.2	35	0.7	0.00	6	8	3	2
	6	90,072	0.3	134	3.0	0.00	19	10	17	13
	7	99,652	0.7	79	18.7	0.00	20	46	7	5
	8†	761,230	12.9	605	300.6	0.69	154	314	61	32
	9†	992,688	13.8	763	300.5	2.59	203	382	80	47
ESCH	1	5,298	0.0	1	0.0	0.00	1	2	1	1
	2	14,226	0.0	3	0.0	0.00	3	4	2	1
	3	19,528	0.0	4	0.1	0.00	4	9	1	1
	4	27,563	0.2	5	0.1	0.00	5	16	2	2
	5	27,629	0.0	6	0.0	0.00	6	7	3	2
	6	90,839	0.1	19	0.1	0.00	19	10	17	14
	7	92,968	1.0	18	0.1	0.00	18	45	8	4
	8	714,246	44.0	143	7.3	0.11	142	308	60	30
	9†	876,355	119.5	175	300.2	0.09	174	371	82	44

B Maps of Test Instances

In this section, the maps of the test instances are given. The locations are indicated by the red open circles. The cyan, green and yellow filled circles indicate the demand points of Wi-Fi, SVC and Alarm, respectively.

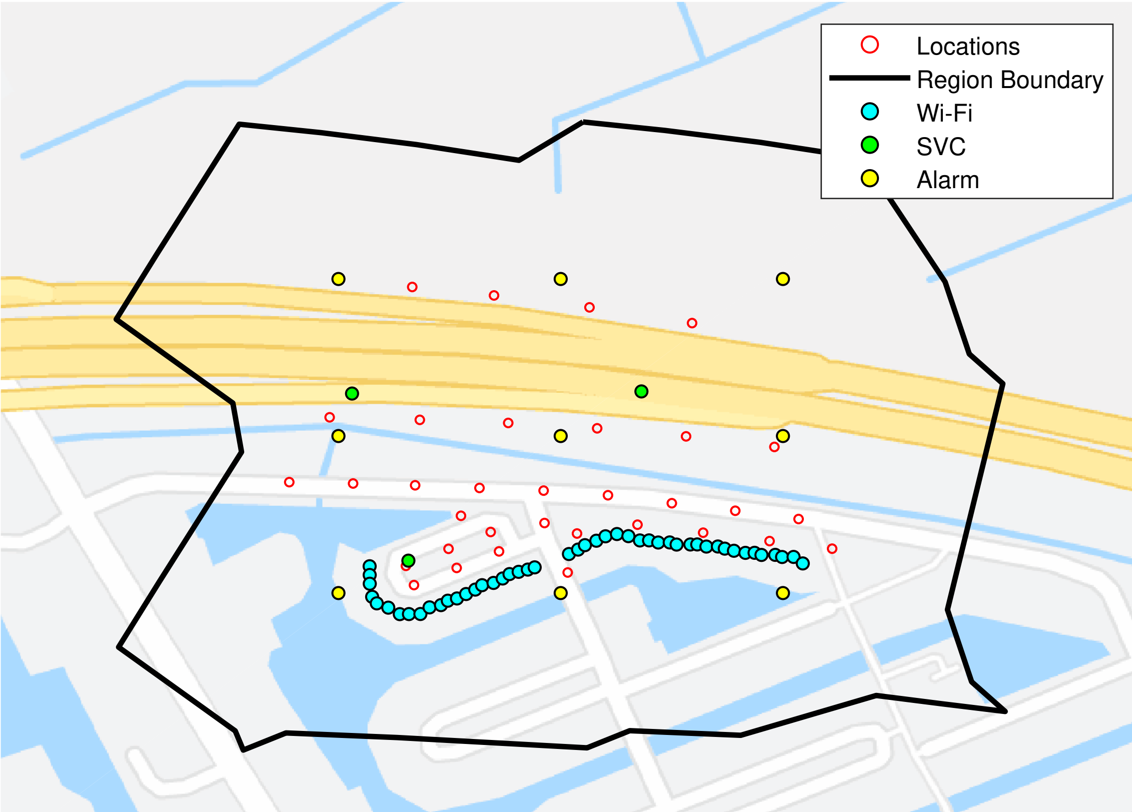


Figure 9: Map of instance 1.

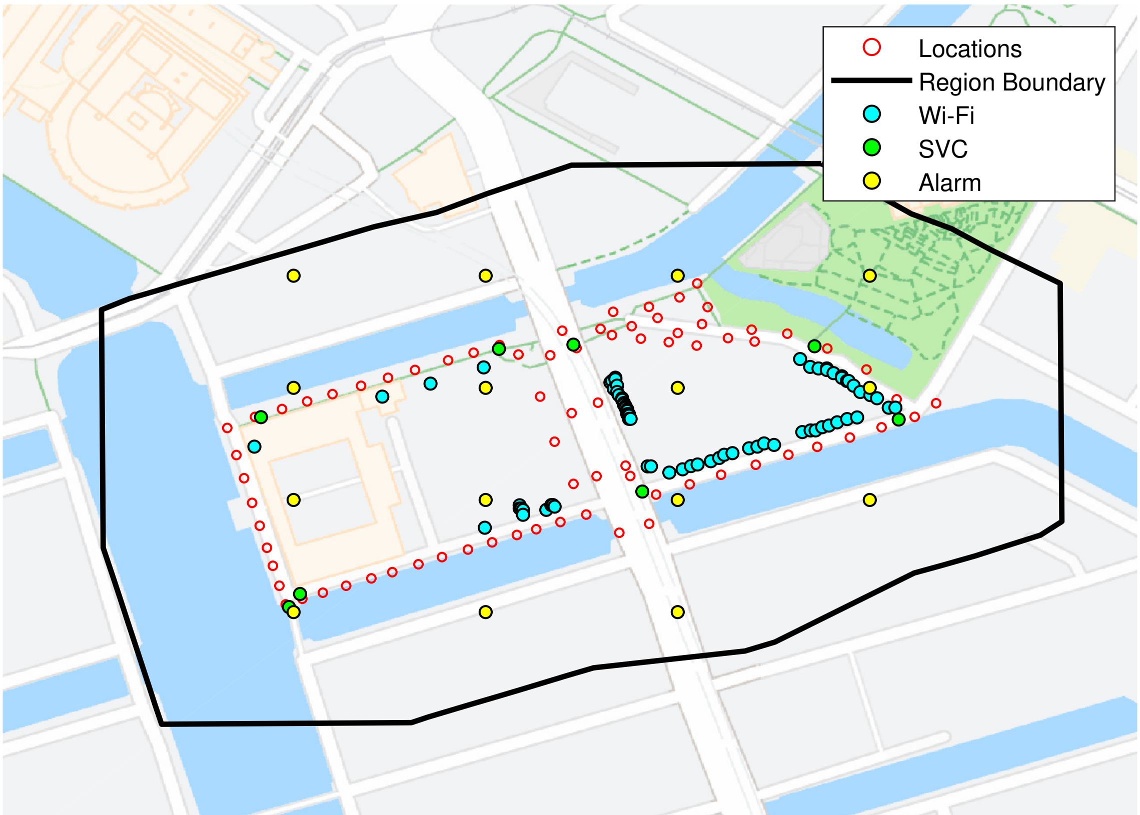


Figure 10: Map of instance 2.

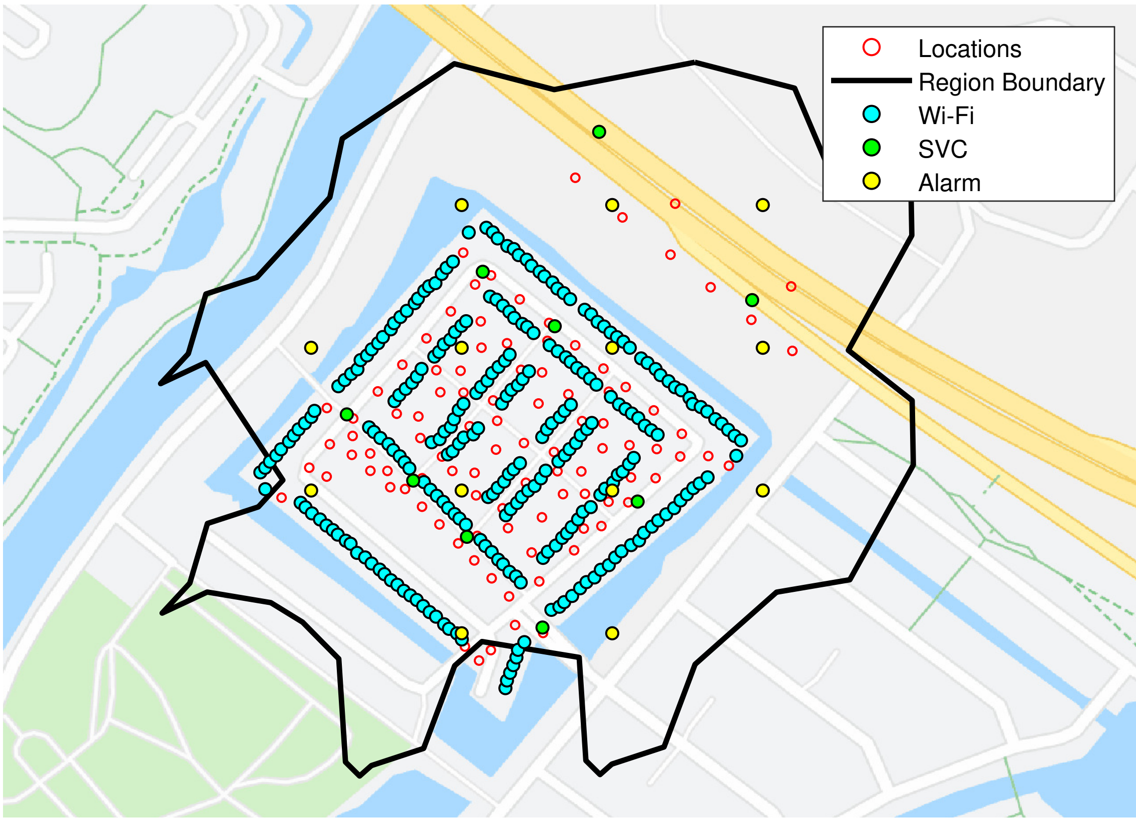


Figure 11: Map of instance 3.

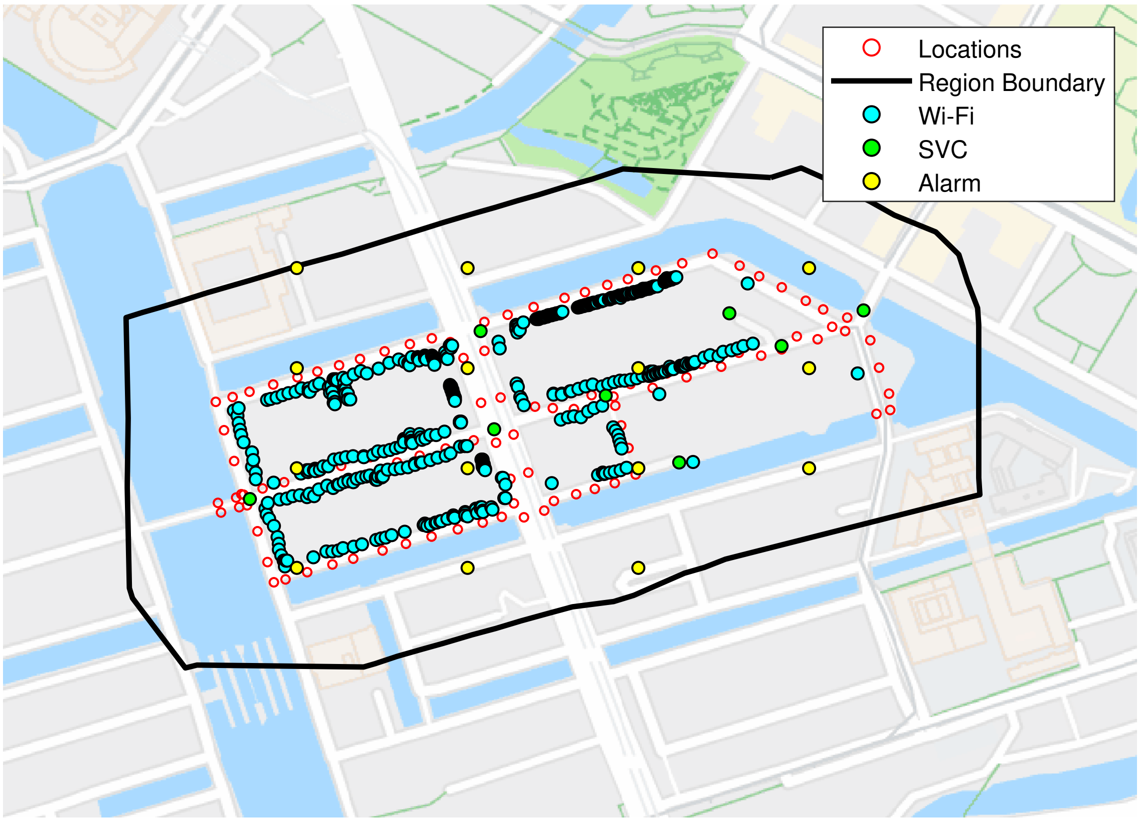


Figure 12: Map of instance 4.

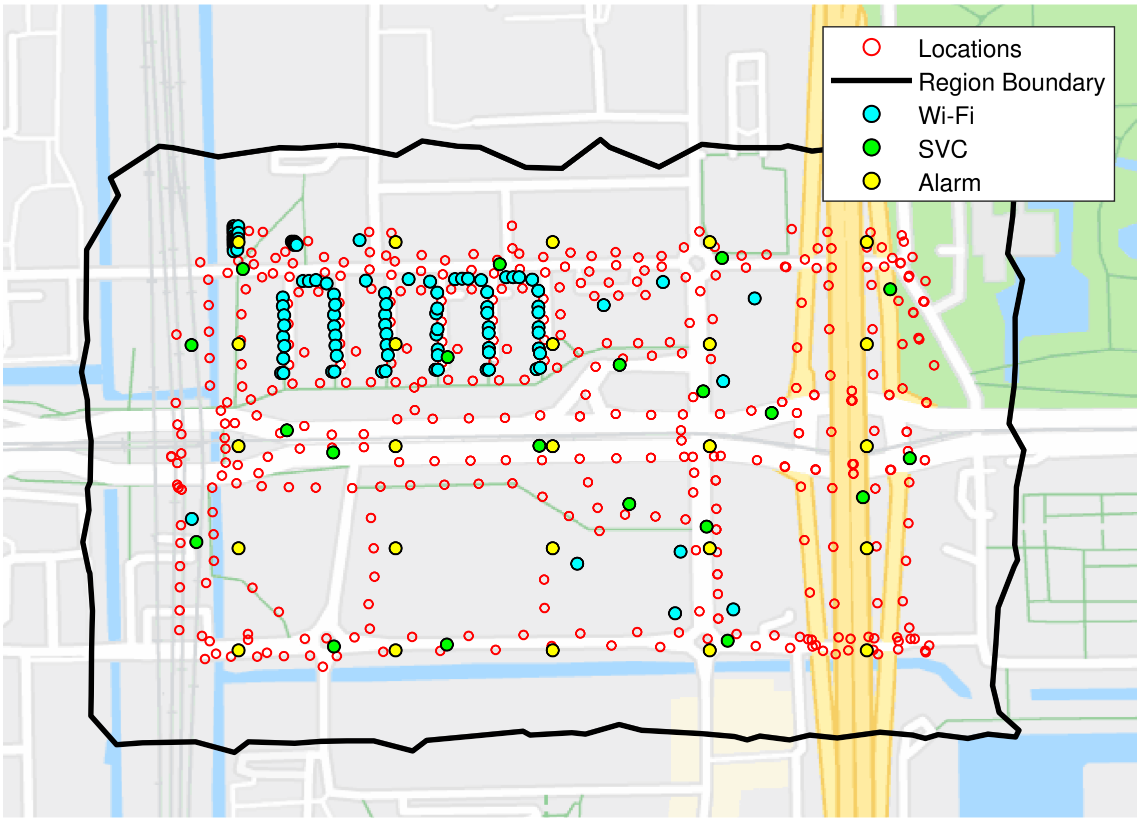


Figure 13: Map of instance 5.

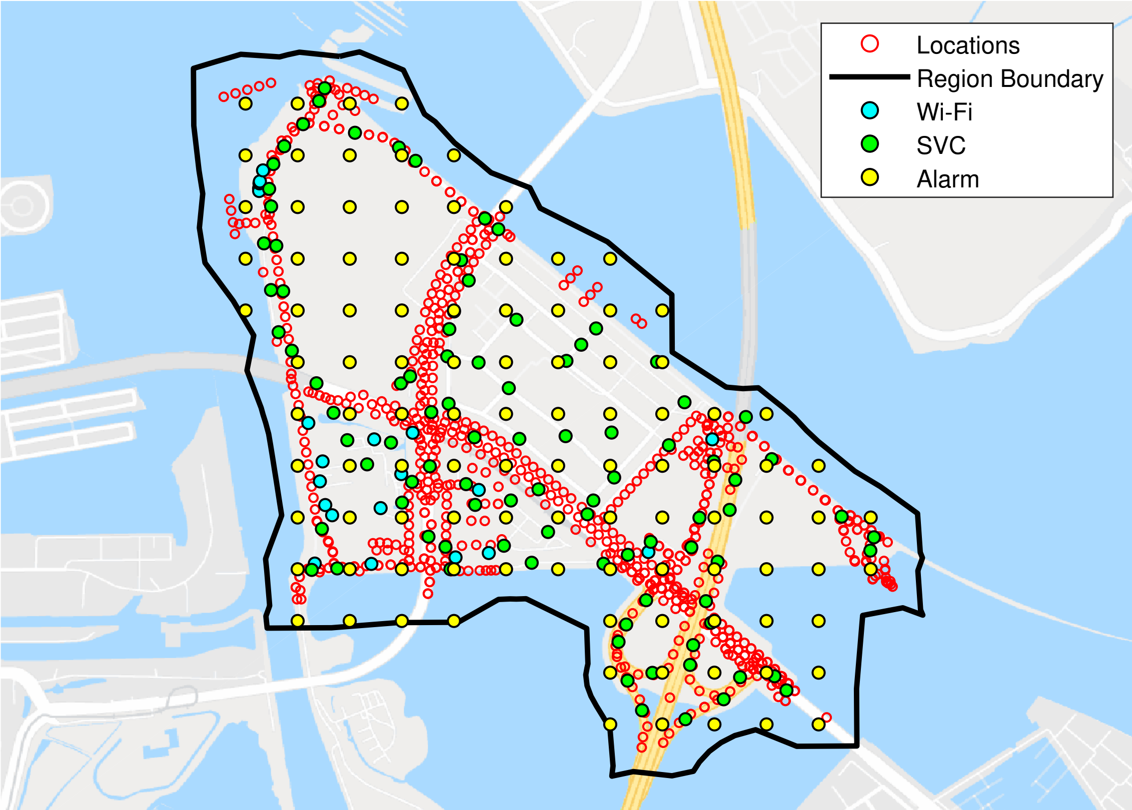


Figure 14: Map of instance 6.

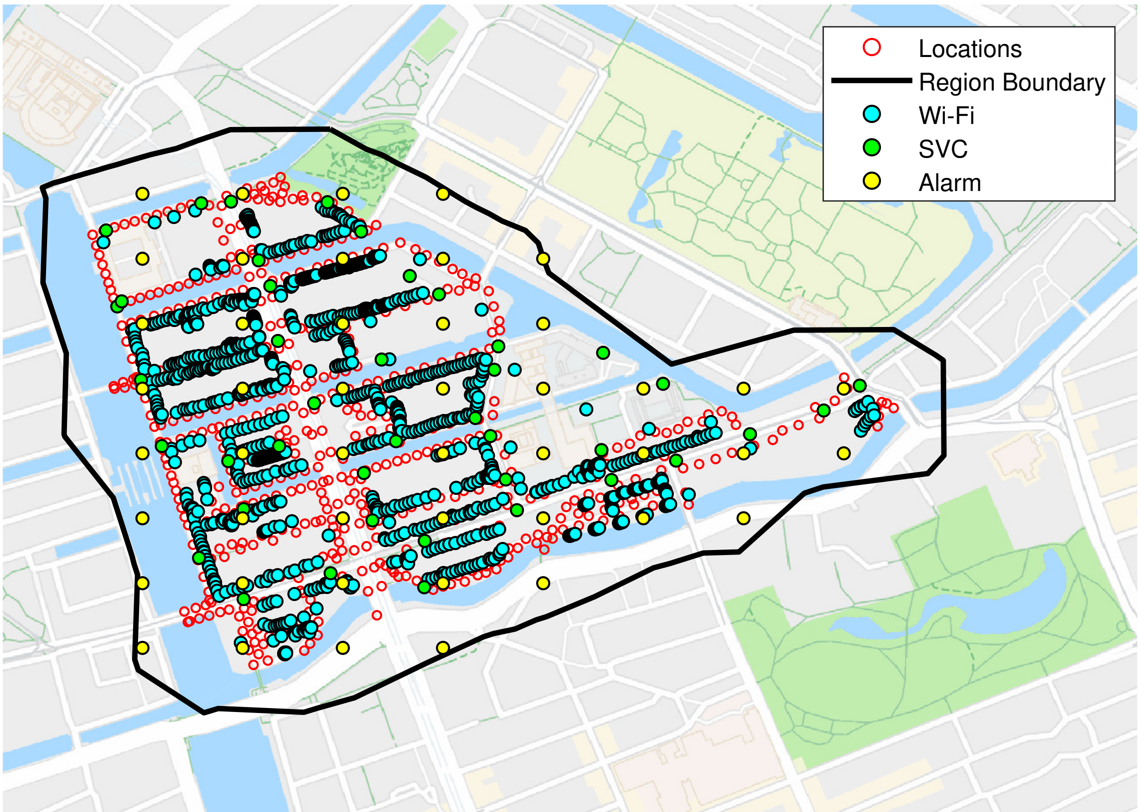


Figure 15: Map of instance 7.

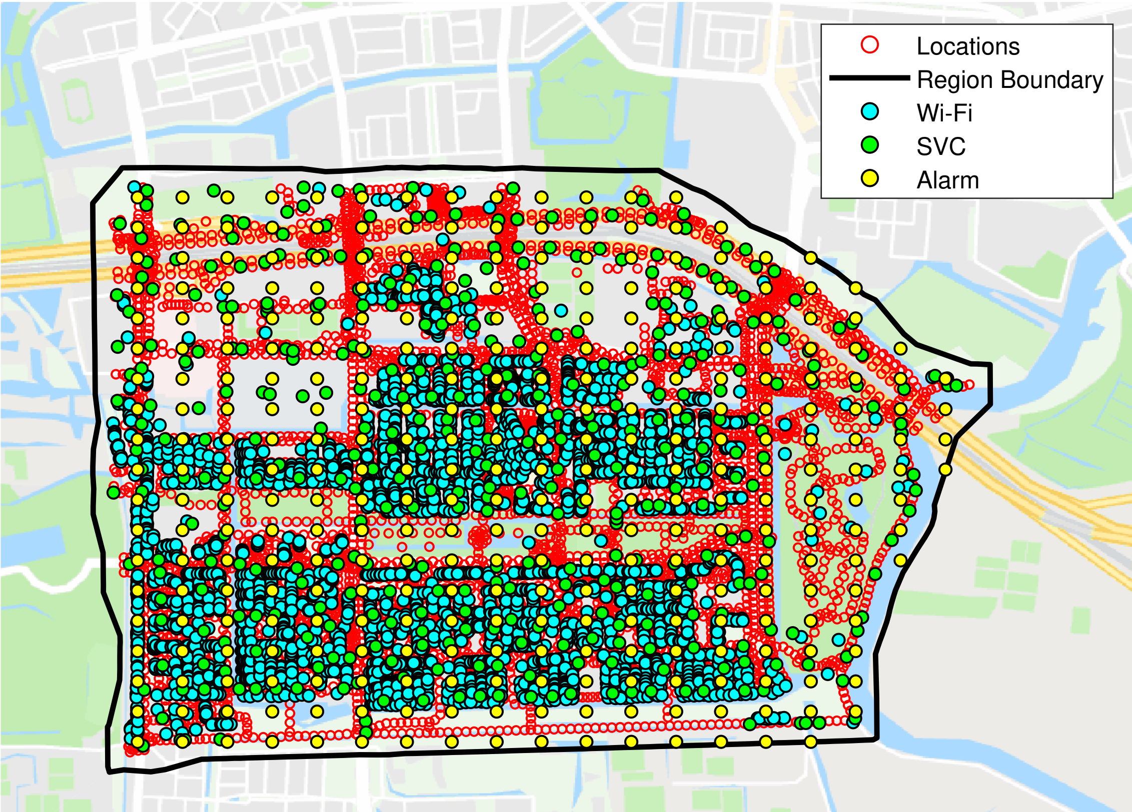


Figure 16: Map of instance 8.

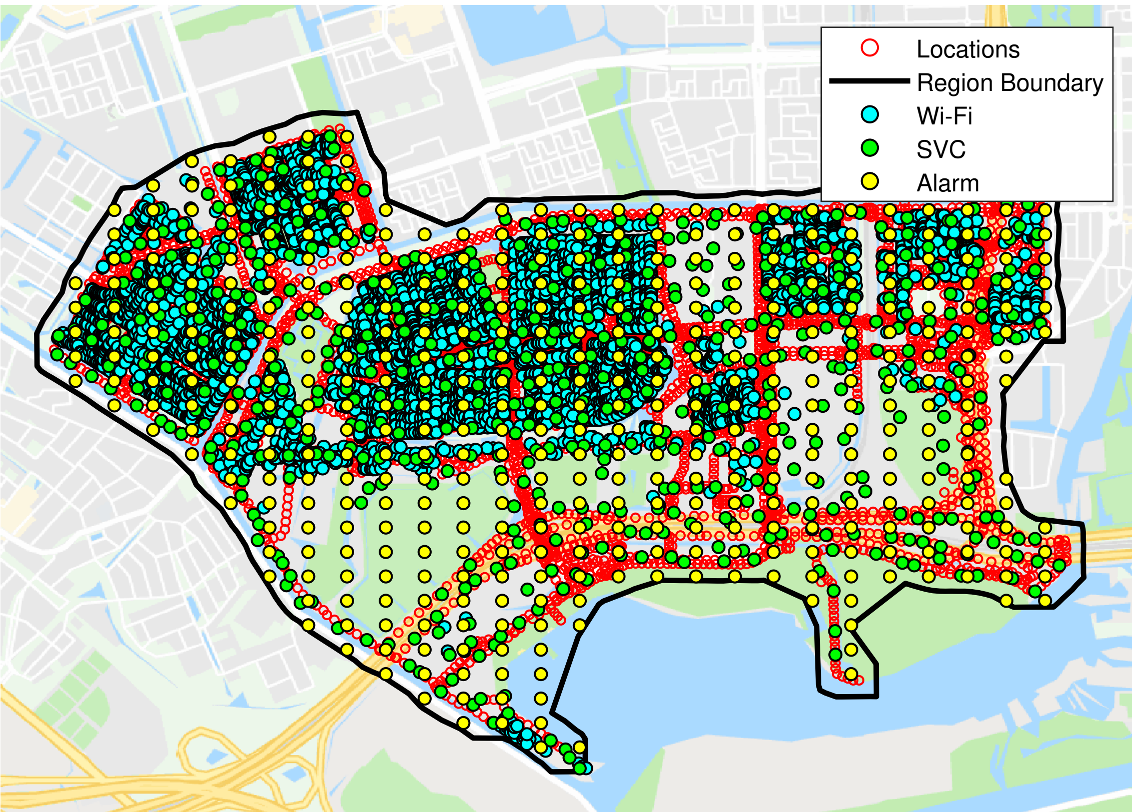


Figure 17: Map of instance 9.