A method for huge scale maximum covering facility location problems with an application to water well placement in West Darfur

Valentijn F Stienen[a], Britt W J R van Veggel[b], Dick den Hertog[b], Marije Jochemsen[c], Joris C Wagenaar[a]
[a]Department of Econometrics and Operations Research, Tilburg University
[b]Amsterdam Business School, University of Amsterdam
[c]510 - Red Cross Netherlands

Abstract. In the analytical context of addressing humanitarian issues, huge-scale facility location problems play an important role. Whether it is about finding locations for schools, hospitals, or water wells, these problems often involve many demand locations and many potential facility locations. Therefore, these problems are often intractable, often impossible to solve due to large memory requirements. This paper proposes a decomposition approach to efficiently solve huge-scale problems in an exact way by clustering the demand points, solving subproblems for each cluster independently, and combining solutions of subproblems through binary integer programming. Clustering takes advantage of the spatial separation of demand locations to address subproblems independently, guaranteeing optimality. Optionally, heuristic subclustering can be used to further divide clusters. When subclustering is used, bounds on optimality gaps are provided, offering quality guarantees about obtained solutions. We demonstrate the effectiveness of the proposed approach with a case study, which is done in collaboration with 510, involving the placement of water wells in West Darfur (Sudan). Solutions for capacitated and uncapacitated problems are obtained within reasonable computation times while providing optimality guarantees where appropriate.

Key words: Facility location · Maximum covering problems · Huge-scale optimization · Water infrastructure planning · Decomposition methods

1. Introduction

Huge-scale facility location problems play an important role when addressing humanitarian challenges from an analytical perspective. In these challenges, facilities often refer to essential institutions, including hospitals, water wells, markets, and schools. Oftentimes, the objective of such challenges is to have a maximum *coverage* of the population, ensuring that a maximum number of people live within the vicinity of at least one facility. Optimizing coverage for placement of new facilities is essential to reach different targets of the Sustainable Development Goals of the United Nations.

Huge-scale facility location problems with a coverage objective can typically be characterized by a *large number of people that we want to be covered* by facilities, and a *large set of potential facility locations*. Moreover, each facility may have a certain maximum capacity, meaning that it can not cover more than a certain number of people. Now, given a budget limiting the number of facilities we can open, we want to determine the optimal locations for installing these facilities to maximize overall coverage. To find these optimal locations of facilities, we typically choose from a large set of potential facility locations. Such a

large set of potential facility locations may arise from uncertainty regarding where these facilities can be feasibly installed. After the optimization, decision-makers can determine whether an optimal location is indeed suitable or if an alternative location should be found, rather than initially examining the feasibility of any location in the region. This approach avoids potential time wastage on confirming the feasibility for locations that are not optimal. However, this can result in huge-scale facility location problems, having many potential facility locations and many people that we want to be covered by a facility. Moreover, we typically can not consider whole population clusters to be covered, as coverage distances are often small. For instance, aiming for coverage within 500 meters means that coverage must be assessed at the household level, which frequently leads to huge-scale facility location problems. In this paper, locations where people live are referred to as demand locations.

Oftentimes, *coverage curves* can give insight in the trade-off between the number of facilities that can be established (the budget) and the coverage that can be obtained. For each budget of facilities, we compute the maximum coverage that can be obtained with this budget. These coverage numbers are then plotted to form a curve, which we refer to as the *coverage curve*. Based on this curve, one can choose an appropriate combination of a budget and coverage, which corresponds to a spatial configuration of facilities. To create coverage curves, we need to solve many huge-scale facility problems, each with a different budget constraint. This underscores the importance of having an efficient approach to address these problems.

Traditionally, these problems have been addressed by formulating them as Mixed-Integer Optimization Problems (MIO) and subsequently solving them with the aid of solvers (*e.g.,* see Farahani et al. (2019)). When dealing with smaller instances, solving these problems is straightforward, especially if we do not consider capacity constraints (or we assume capacity to be infinite). In this case, we can formulate the facility location problem as a maximum covering problem in which we do not need to assign individual demand locations to facilities. This significantly reduces the computation time and the number of variables required in the facility location problem. However, the complexity of the problem increases significantly once capacity constraints are incorporated. In this case, we need to know which (parts of) demand locations are assigned to which (open) facilities.

In this paper we propose a (decomposition) method to solve huge-scale uncapacitated and capacitated facility location problems that are not solvable with the traditional methods. In particular, we propose an approach that efficiently creates a coverage curve for such huge-scale, uncapacitated and capacitated, facility location problems. To demonstrate the proposed approach, we apply it to a case study to increasing the number of people that have access to clean and safe drinking water in West Darfur, Sudan. This is done in collaboration with 510, an initiative of the Netherlands Red Cross, that focuses on using data and digital to enhance humanitarian aid and disaster response (Red Cross 510 (2022)). The context is more elaborately explained in subsection 4.1.

Literature overview

Next, we examine the literature relevant to both solving huge-scale facility location problems and improving access to drinking water. Many literature exists in the area of using facility location problems in the humanitarian setting. For an extensive literature review of facility location problems in a humanitarian setting, we refer to Boonmee et al. (2017). They distinguish four different facility location problems in a humanitarian setting: deterministic, dynamic, stochastic and robust facility location problems. More recently, Dönmez et al. (2021) present an overview of the research done on facility location problems under uncertainty in a humanitarian setting. They review the literature from different perspectives, including the criteria to optimize and the solution method used.

In this paper, our primary focus lies on maximum covering location models Church and ReVelle (1974). These models are designed to maximize population coverage while limiting the number of (new) facilities to be placed. It is important to note that this represents a specific version of the facility location problem with the objective of maximizing the total coverage of all the (open) facilities. Covering problems like these are crucial in the humanitarian sector, as people often require access to essential necessities such as hospitals, markets, or drinking water points.

As illustrated in Megiddo et al. (1983), the facility location problem is NP-hard. Therefore, heuristic methods have become a popular approach for solving maximum covering location problems. Many heuristics have been developed specifically for the uncapacitated maximum covering location problem. These include genetic algorithms (Zarandi et al. (2011)), local search methods (Rodriguez et al. (2012)), approaches based on Lagrangian relaxation (Galvão and ReVelle (1996)), and techniques using simulated annealing (Li et al. (2011)). Additionally, multiple papers examine the applicability of Greedy Randomized Adaptive Search Procedure (GRASP) Resende and Ribeiro (1998) in this context. One of the first applications of GRASP to the maximum covering facility location problem is the work of Resende (1998). In Máximo et al. (2017) and Máximo and Nascimento (2019) it is shown that the application of GRASP with path-relinking can effectively solve problem instances as large as: (# demand locations $\times$ # potential facility locations) 8,000 $\times$ 8,000. Recently, Antonissen et al. (2023) introduce a heuristic that is based on GRASP, local search, and path-relinking. They demonstrate the performance of their method on an instance size of 400,000 $\times$ 350,000. Moreover, various decomposition techniques have been introduced in the literature. For instance, Senne et al. (2010) propose a cluster-based decomposition method to enhance the upper bounds for solving maximum covering location problems. This strategy involves breaking down the original problem into smaller subproblems, each representing a cluster, thus allowing for more efficient solutions using exact methods and reduced computational times. Another example is the work described in Cordeau et al. (2019). They utilize Benders decomposition to demonstrate its ability to handle problem instances expanding up to dimensions of 40,000,000 $\times$ 100, where they do require the number of potential facility locations to very

small in comparison to the number of demand locations.

It is worth noting that these methods typically do not incorporate capacity constraints. For instance, the method developed in Antonissen et al. (2023) is unable to incorporate capacities for the facilites, and has an exponential running time with regards to the number of wells to be placed (see Appendix 5). Methods that incorporate capacity are often considering much smaller maximum covering location problems. The capacitated maximum covering location problem was first considered in Pirkul and Schilling (1989), Pirkul and Schilling (1991), and Haghani (1996). More recently, papers have considered slightly larger problems (*e.g.,* see ElKady and Abdelsalam (2016), Vatsa and Jayaswal (2021), Atta et al. (2022)), however, these dimensions remain insufficient for the dimensions involved in our case study. They do not consider more than 1,000 demand locations and potential facility locations. These works prioritize additional complexities, such as fuzzy coverage areas or server uncertainty, over addressing huge-scale capacitated maximum covering location problems. In this paper, we introduce a method capable of addressing both huge uncapacitated *and* capacitated facility location problems.

To demonstrate our proposed methodology, we conduct a case study in which we aim to optimize water well accessibility in West-Darfur, Sudan. Many descriptive studies have been performed analyzing populations access to drinking water. For instance, Deshpande et al. (2020) map geographical inequalities between access to drinking water to lower- and middle income countries using a Bayesian geostatistical model. Additionally, Ajisegiri et al. (2019) examines access to drinking water in Nigeria, considering seven distinct indicators. Regarding prescriptive analytics, the literature is much more scarce. An example of an optimization model, designed to find the optimal distribution of water resources, can be found in the work of Zhai et al. (2023). They develop a decentralized water project optimization model to enhance water access equity in rural areas. Moreover, they introduce a stochastic model to inform water project decisions in politically unstable regions, emphasizing the consideration of potential war-related impacts. Where they focus on minimizing the population-weighted total travel distance, we are interested in the trade-off between a budget of wells and the maximum coverage we can achieve. Moreover, they distinguish between various water projects such as pumps and wells, but in a smaller setting. Instead, we focus on a more holistic view of where to place facilities, given a large region, such as a province, or a country.

Contributions

This paper makes several contributions to the field of huge-scale facility location optimization. We propose a new decomposition approach that is able to solve huge-scale, maximum covering, uncapacitated and capacitated, facility location problems. In particular it is able to find (parts of) the corresponding coverage curves efficiently. This approach leverages spatial clustering of demand points to partition huge maximum

covering facility location problems into independent subproblems defined on each cluster. This clustering method still guarantees to find the optimal solution for the original unclustered facility location problem. If, for computational reasons, the problems needs to be partitioned into even smaller subproblems, we propose a subclustering heuristic. As this is a heuristic approach, we cannot guarantee optimality anymore. Instead, we provide upper bounds on the optimality gaps of the obtained solutions. After (sub)clustering, we obtain feasible solutions for the original unclustered problem, by formulating a binary integer programming model to effectively integrate solutions from the subproblems. These methodological contributions are demonstrated through a case study, which is done in collaboration with 510. The case study is about finding optimal locations of water wells in in West Darfur (Sudan) containing over 750,000 people. The case study shows that the proposed approach can efficiently compute coverage curves for the uncapacitated and capacitated huge-scale facility location problems, that were previously intractable using traditional solution techniques. We also show that the provided upper bounds on the optimality gap are worst case scenarios that are likely to be much lower in practice. Moreover, this facility location optimization solution can enhance the Red Cross's decision-making process for borehole location, identifying the most strategic and efficient locations for water points. The identification of these locations can reduce time-consuming processes of selecting potential borehole locations while ensuring an informed response, and ultimately contributing to a maximal impact of their humanitarian efforts. Finally, we do note that all code is publicly available at `https://github.com/valentijnstienen/WaterwellAccess`.

2. Facility location problems

In this section, the MIO formulations of the uncapacitated and capacitated maximum covering location problems, adapted from Pirkul and Schilling (1991), are given. Let $\mathcal{I}$ be the set of demand locations, and let $\mathcal{B}$ be the set of potential facility locations. Moreover, let a_i be the number of people in demand location $i \in \mathcal{I}$. In this context, people are considered *covered* when they are located within *covering distance* of the assigned facility. We want to maximize the number of people that is covered. Furthermore, let M be the maximum number of facilities to install, and U_b be the capacity of potential facility location $b \in \mathcal{B}$. Moreover, let N_i be all the facilities that are within covering distance from demand location i. Finally, let v_b be a binary variable that indicates whether potential facility location $b \in \mathcal{B}$ needs to be opened.

Depending on whether we incorporate the capacities of facilities, we need another variable. If we do not include the capacities, we use the binary variable z_i that indicates whether demand location i is covered by an open facility. If we do include the capacities, we use the variable w_{ib} that represents the fraction of the demand a_i that is satisfied by facility b. Below, we show both the uncapacitated and capacitated maximum covering facility location problem:

Constraint (U.1) and (C.1) bound the number of opened facilities to be at most M. Constraint (U.2) ensures

$$\max \quad \sum_{i\in\mathcal{I}} a_i z_i,$$

$$\text{s.t.} \quad \sum_{b\in\mathcal{B}} v_b \le M, \qquad \text{(U.1)}$$

$$z_i \le \sum_{b\in N_i} v_b \qquad \forall i\in\mathcal{I}, \qquad \text{(U.2)}$$

$$v_b, z_i \in \{0,1\}, \quad \forall b\in\mathcal{B},\ i\in\mathcal{I}. \qquad \text{(U.3)}$$

Figure 0 **Uncapacitated maximum covering facility location problem.**

$$\max \quad \sum_{i\in\mathcal{I}}\sum_{b\in\mathcal{B}} a_i w_{ib},$$

$$\text{s.t.} \quad \sum_{b\in\mathcal{B}} v_b \le M, \qquad \text{(C.1)}$$

$$w_{ib} = 0 \qquad \forall i\in\mathcal{I},\ b\notin N_i, \qquad \text{(C.2)}$$

$$\sum_{b\in\mathcal{B}} w_{ib} \le 1 \qquad \forall i\in\mathcal{I}, \qquad \text{(C.3)}$$

$$\sum_{i\in\mathcal{I}} a_i w_{ib} \le U_b v_b \qquad \forall b\in\mathcal{B}, \qquad \text{(C.4)}$$

$$v_b \in \{0,1\},\ w_{ib} \ge 0, \quad \forall i\in\mathcal{I},\ b\in\mathcal{B}. \qquad \text{(C.5)}$$

Figure 0 **Capacitated maximum covering facility location problem.**

that demand location $i \in \mathcal{I}$ can only be covered if there is at least one facility b in the neighborhood of i is opened. Constraint (C.2) assures that open facilities can only cover (parts of) demand locations that are within covering distance from this facility. Constraint (C.3) ensures that we satisfy at most 100% of the people in demand location i. Note that it is possible that a demand location is partly covered from one facility and partly from another facility. Constraint (C.4) is the capacity constraint, it restricts the number of people that are assigned to facility b at its capacity, U_b. Finally, Constraint (U.3) and (C.5) define the domains of the variables. Observe that the number of variables for the capacitated facility problem is much larger than for the uncapacitated version.

To create coverage curves, we need to solve many of these huge-scale facility problems, each with a different budget constraint. Therefore, in this chapter we develop an efficient approach to solve these problems via decomposition.

3. Proposed decomposition approach

In this section, we propose a decomposition approach that is able to solve huge-scale (capacitated) facility location problems in an exact way by clustering the demand locations (Section 3.1), solving subproblems for each cluster independently, and combining solutions of subproblems through binary integer optimization (Section 3.2). Additionally, we propose an optional heuristic subclustering method that can be used to further divide clusters (Section 3.3). When subclustering is used, we also present bounds on optimality gaps, which give insight in the quality guarantees of obtained solutions (Section 3.4).

3.1. Clustering method: DBSCAN

First, we discuss how we use DBSCAN (Ester et al. 1996) to form clusters of demand locations. Subsequently working with these clusters still guarantees that we find optimal solutions. This approach is based on the fact that people can only be covered by a potential facility location if they are within distance D from that potential facility location. Hence, two demand locations may share a facility location if the distance

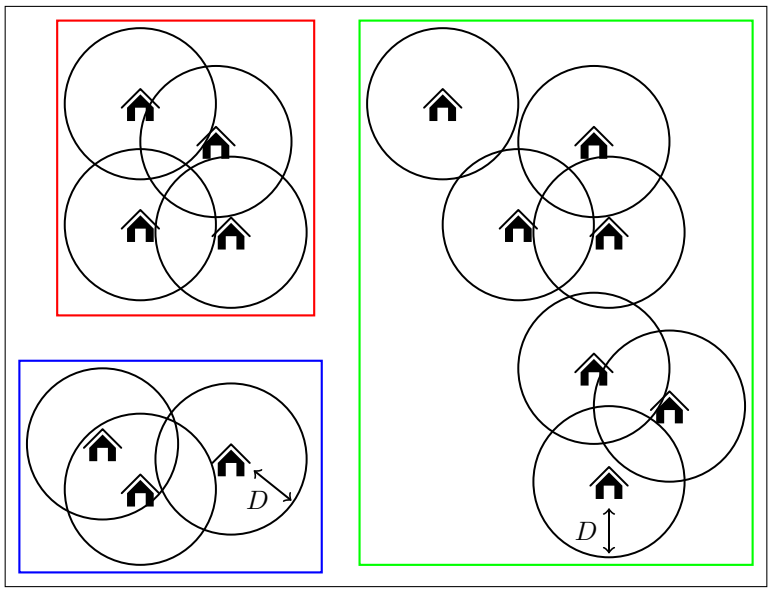

Figure 1 **An illustration how demand locations ( ) can be clustered, such that demand locations in different clusters can not share a facility. Here, D represents the maximum distance a person can be from a facility and still be considered within coverage.**

between these demand locations is at most $2D$. Figure 1 visualizes an example of how demand locations can be clustered in a given situation.

We observe three different clusters, indicated with three different boxes of different colors. Demand locations in each cluster are only *connected* to demand locations in their own cluster. Here, connected means that demand locations are within $2D$ distance from each other; they can potentially share a facility. Demand locations in different clusters cannot share a facility. Therefore, instead of solving the complete facility location problem, we can choose to solve facility location problems for all the individual clusters, and then integrate these solutions into a comprehensive solution for the original (unclustered) situation. This will guarantee the optimal solution.

To find these clusters, we can use a well-known, density-based clustering technique: DBSCAN (Ester et al. 1996). DBSCAN is a density-based clustering algorithm that classifies data points into clusters where each cluster is formed by densely grouped points. This clustering is based on two parameters: ϵ, which defines the radius for neighborhood searches, and $MinPts$, the minimum number of points required to form a cluster. This algorithm can be used to effectively separate regions of high density from noise and outliers, and is often used to identify clusters of arbitrary shapes and sizes. The general computational complexity of DBSCAN is $O(n^2)$, where n is the amount of points to be clustered. More specifically, in our case, we use the DBSCAN algorithm with $MinPts = 1$, as we want to keep all demand locations, and with $\epsilon = 2D$. Note that these settings slightly change the practical complexity of the algorithm. Using these settings, it

means that every point automatically belongs to a cluster, which reduces the amount of checks required to determine whether a point belongs to a cluster. Using DBSCAN in this way results in clusters that contain points that are more than $2D$ distance away from points in other clusters, which is exactly what we need. It is important to note that for a given set of demand location, this clustering procedure always ends up with the same set of clusters.

3.2. Combining cluster coverage curves

The next step of our proposed approach is to determine the coverage curves of all clusters. For each cluster and for each budget of facilities, we solve a (capacitated) maximum covering location problem to determine the corresponding maximum coverage we can get. Then, we combine these coverage curves (of all clusters) into one coverage curve for the original (unclustered) facility location problem. Given a budget (a number of facilities we are allowed to install), we want to determine where to install facilities. This problem can be formulated as a binary integer optimization model.

Let $c \in \mathcal{C}$ be the set of clusters that exist in the problem, and let $j \in \mathcal{J} = \{0, 1, ...\}$ be the set of available numbers of facilities we can allocate to each cluster. Moreover, let k_{cj} be the number of people that are covered in cluster c, when assigning j facilities to this cluster. Additionally, let M be the budget, the number of facilities we are allowed to install. Furthermore, let x_{cj} be a binary variable that indicates whether j facilities are allocated to cluster c. Then, the corresponding binary integer optimization model can be formulated as:

$$
\begin{aligned}
\max \quad & \sum_{c \in \mathcal{C}} \sum_{j \in \mathcal{J}} k_{cj} x_{cj}, & & \\
\text{s.t.} \quad & \sum_{j \in \mathcal{J}} x_{cj} \leq 1 & \forall c \in \mathcal{C}, & \quad \text{(B.1)} \\
& \sum_{c \in \mathcal{C}} \sum_{j \in \mathcal{J}} j \; x_{cj} \leq M, & & \quad \text{(B.2)} \\
& x_{cj} \in \{0, 1\} & \forall c \in \mathcal{C}, \; j \in \mathcal{J}. & \quad \text{(B.3)}
\end{aligned}
$$

In this problem, we maximize the number of people that can be covered in all individual clusters, resulting in a maximum coverage for the unclustered problem. Here, Constraints (B.1) ensure that each cluster can only be assigned a single number of facilities. Constraint (B.2) ensures that the total number of facilities given to each cluster can not exceed our global budget, M. Finally, Constraints (B.3) define the domains of the decision variables. To obtain the coverage curve for the original (unclustered) problem, we have to solve this model for all different values of M.

This problem can be solved as a MIO model, but can also be solved using dynamic programming. The problem can be approached as longest path algorithm on a graph. One would solve the recursive formula

$$V_n(i) = \max_{0 \leq k \leq min(i, M_n)} \quad V_{n-1}(k) + p_{n,k-i}, \tag{1}$$

where $V_n(i)$ is the optimal coverage if i wells are distributed over clusters $0, 1, ..., n$, $p_{n,j}$ is the number of people that are covered when assigning j facilities to cluster n, and M_n is the minimum number of wells needed to reach a coverage of 100% in cluster n.

This recursive formula is implemented with memory. We store the found value for a $V_n(i)$ along with the optimal number of wells for cluster n. To reconstruct the well distribution once we have finished our recursive calculations, we start at the final solution and iterate backwards. To find how many wells are assigned to cluster m, we look at the solution found in $V_m(j)$, where $j = cs - \sum_{p>m} k_p$ and k_p is the optimal number of wells assigned to cluster p.

This methodology enables us to calculate a trade-off curve very easily as well. Since all $V_n(i)$'s are calculated and stored, we can use these to calculate $V_{n'}(j)$ for all $j \leq i$ and generate a curve that shows the trade-off between number of wells and coverage. This is significantly faster than using an optimization algorithm to calculate coordinates of the curve individually.

3.3. Subclustering: cluster size reduction

It might happen that after clustering some of the corresponding (capacitated) maximum covering location problems are too large to solve. In this case, we need to create even smaller clusters that can be solved. In other words, we need to create clusters of demand locations *within* the clusters found by DBSCAN. We will refer to this procedure as *subclustering*. For example, in the most right cluster of Figure 1, we observe that the top four and the bottom three demand locations could likely be separated in two different subclusters while still guaranteeing to find the optimal solution for the original (unclustered) situation. The only way in which the obtained integrated solution would not be optimal is if it is optimal to install a facility in the small area overlapping the two connecting circles. Such a facility can only serve these two demand locations, which makes it even less likely to be an optimal location. Our approach of subclustering clusters obtained from the DBSCAN algorithm is based on this observation.

It is important to note that there are many ways to split a cluster in subclusters. We could, for instance, start with a cluster of 1,000 demand locations and end up with 5 subclusters of sizes 10, 10, 10, 10, and 960. This is, however, typically not what we would like to end up with. We would still need to solve a facility location problem with almost 1,000 demand locations. Note that this could, for instance, happen if we apply (sub)clustering techniques such as K-means, or if we would again apply DBSCAN, but now with a smaller neighborhood parameter. Instead, we would rather have five subclusters of size 200. In short, we would like to obtain subclusters that have at most T demand locations per subcluster. Note that, typically, T depends on the computational resources that are available. The larger T, the better the heuristic solutions will be.

Now, we could find a way to end up with a set of subclusters with all approximately T demand locations. This means that the amount of subclusters would be the total amount of demand locations in the original cluster divided by T. Setting up an optimization model for this is possible, but can require a lot of computation time for large clusters. Even creating a good heuristic that splits the cluster in this amount of subclusters (or any other amount of subclusters that is larger than 2) is typically computationally intensive. Creating fast algorithms for such amounts of subclusters is therefore an opportunity for future research.

For simplicity and for computational time reasons, we, therefore, decide to come up with a fast heuristic procedure that sequentially splits a cluster in **two** subclusters. So, assuming that we sequentially split a cluster in two, in the previous example, after one iteration, we would have two subclusters of size 500, after the next iteration, we would have 4 clusters of 250, etc. Note that this does mean that it could happen that our subclusters are (much) smaller than the threshold T, for instance when before the last iteration, we had subclusters of size $T + 1$. We would need to do one more iteration leading to subclusters of size almost equal to $T/2$.

Next, we discuss more specifically *how* we split a cluster in two subclusters. Also here, there is a tradeoff between the time it takes to do this subclustering and the quality of the obtained solution. One option is, for instance, to find the two points that are furthest away from each other. Then, sort all points according to their distance to one of the extreme points. Then add half of the sorted list to a cluster that includes this extreme point. The other points, together with the other extreme point, form the other cluster. Another alternative is to solve an exact minimum cut problem that divides the demand locations in two similar-sized clusters, while minimizing the number of *cut* connections between demand locations. This gives the best results, but it may take a long time to actually solve this optimization problem.

In this research, we propose a method that sequentially splits a cluster in half. We first find the two points that are the furthest away from each other. Both these points now form their own cluster. Next, we start with any of these clusters, which we will refer to as the current cluster. We compute the distances from all the unclustered points to any of the points in the current cluster, and in the other cluster. Then we subtract the distances to the other cluster from the distances to the current cluster. Now, the point that has the maximum value is the point that we add to the *other* cluster. Note that this is, at first, the point that is furthest away from the current cluster. Since, we also subtract the distance to the other cluster, we prefer points that are close to the other cluster. So a point that is far away from the current cluster and close to the other cluster will be added to the other cluster. Then, we turn to the other cluster and do the same thing, in the end, adding a point to the current cluster. We do this until there are no more unclustered points. This algorithm is summarized in Algorithm 1. Here, P represents the set of demand locations that need to be subclustered.

Note that this algorithm results in two equal-sized subclusters if the number of people in the original cluster

Algorithm 1 Subclustering Algorithm

procedure SUBCLUSTERPOINTS(P)
  Find the points $p_0 \in P$ and $p_1 \in P$ that are furthest apart.
  Initialize clusters: $N[0] = \{p_0\}$ and $N[1] = \{p_1\}$, where $N[0]$ ($N[1]$) contains all $p \in P$ that are currently assigned to cluster 0 (1), respectively.
  $P = P \backslash \{p_1, p_2\}$ ▷ Remove points p_1 and p_2 from P
  `turn` = 0 ▷ Indicating cluster 0's turn
  while $P \neq \emptyset$ **do**
    $D_{\texttt{turn}}$, $D_{\texttt{not turn}}$: vectors containing the shortest distances from each point $p \in P$ to the nearest point within the `turn` (`not turn`) cluster, respectively.
    $S = D_{\texttt{turn}} - D_{\texttt{not turn}}$ ▷ Compute distance scores for each $p \in P$
    $p_{\text{max}} \leftarrow \text{argmax}_{p \in P} S_p$

    $N[\texttt{not turn}] = N[\texttt{not turn}] \cup \{p_{\text{max}}\}$ ▷ Add point p_{max} to the cluster
    $P = P \backslash \{p_{\text{max}}\}$ ▷ Remove point p_{max} from P

    `turn` $\leftarrow$ `not turn` ▷ Toggle `turn` to switch *playing* clusters
  end while
  return N[0] and N[1]
end procedure

is even. If the number of people in the original cluster is odd, we will have one subcluster with one more demand location. Moreover, the complexity of this algorithm is $O((|P|)^2)$. The main factors contributing to this complexity include the loop iterating through the number of points in P and computing the distance vectors $D_{\texttt{turn}}$, $D_{\texttt{not turn}}$ in each iteration. This means that when dealing with very large clusters that require subclustering, the computation time could become substantial. In such cases, it might be worth exploring alternative subclustering methods to improve efficiency. This is, however, beyond the scope of this research. In our case, because we use DBSCAN for the initial clustering step, it is unlikely we will encounter large clusters requiring subclustering.

To summarize, this (sub)clustering method creates two clusters with an equal number of demand locations from a single larger cluster. In practice, one keeps subclustering clusters until the number of demand locations for each of the subclusters is less than or equal to T.

3.4. Upper bounds on optimality gap

Subclustering may lead to suboptimal solutions. In this section we derive an upper bound for the optimality gap. It is important to note that for the subclustering, we do not assume that subclustering is done in any particular way. The only thing we assume for computing this upper bound is that each demand location now contains an additional attribute that indicates to which subcluster it belongs. We start with deriving an upper bound on the optimality gap of a specific cluster that has demand points from different subclusters.

Given a specific cluster, let $\mathcal{L}$ be the set of all potential facility locations in this cluster, and let $\mathcal{D}$ be the set of all demand locations in this cluster. Let $\mathcal{SC}$ be the set of the different subclusters that are present in this cluster. Then, let $d_s \subseteq D$ be the set of demand locations that belong to subcluster s for all $s \in \mathcal{SC}$. So, $\mathcal{D} = \bigcup_{s \in \mathcal{SC}} d_s$.

Definition (solution): $L \subseteq \mathcal{L}$ is a *solution* that consists of a set of potential facility locations that are opened.

Definition (coverage): c(l, d) denotes the amount of people in location set $d \subseteq \mathcal{D}$ that are covered using solution l.

Next, we define three different sets of solutions. Given a specific budget, let L^* be the solution that maximizes the coverage when we do not use subclustering. This means that, in this situation, we can cover $c(L^*, \mathcal{D})$ people. Note that we typically do not know this number. Moreover, let L^*_s for $s \in \mathcal{SC}$ be the set of facility locations that are obtained when *splitting* solution L^* (which is obtained without subclustering) as follows. Each facility $l \in L^*$ is allocated to the set L^*_s corresponding to subcluster s which covers the highest number of people from facility l. Note that, $L^* = \bigcup_{s \in \mathcal{SC}} L^*_s$. Finally, let L^s be the solution that maximizes the coverage in subcluster s given the amount of facilities in L^*_s. In other words, we are redistributing the facilities in L^*_s to maximize the coverage in this subcluster. Note that this means that $c(L^s, d_s) \geq c(L^*_s, d_s),\ \forall s \in \mathcal{SC}$.

To clarify these three different solutions, consider the example situation sketched in Figure 2. This represents a situation in which a cluster consists of demand locations (◯) that belong to two different subclusters. Now, suppose the optimal solution of the whole cluster with a budget of 5 facilities is shown in Figure 2, where the triangles represent the optimal facilities. In other words, $L^* = \{\triangle_1, \triangle_2, \triangle_3, \triangle_4, \triangle_5\}$. Now, suppose we want to find the solutions L^*_1 and L^*_2. We know that all facilities that only cover people from a single subcluster, say s, will be in the corresponding solution L^*_s. There is, however, one facility that covers people from both subclusters, $\triangle_3$. Facility $\triangle_3$ will be assigned to the solution corresponding to either subcluster 1 or subcluster 2, based on which subcluster it covers more people from. Assume that this is the demand location from subcluster 2. Then $L^*_1 = \{\triangle_1, \triangle_2\}$ and $L^*_2 = \{\triangle_3, \triangle_4, \triangle_5\}$. Note that indeed $L^* = L^*_1 \cup L^*_2$. Now, the solution L^1 (and L^2) can be obtained by maximizing the coverage for subcluster 1 (and 2), given the respective budgets of 2 and 3 facilities. Solutions L^1 and L^2 are visualized in the right plot of Figure 2.

Recall that the proposed algorithm suggests to use a solution that is formed on the basis of optimal solutions of the individual subclusters. Let L^{**} be this solution that is proposed. Note that it is not necessarily equal to $L^1 \cup L^2$, as the budgets per subcluster may differ. Therefore, we do note that $c(L^{**}, \mathcal{D}) \geq \sum_{s \in \mathcal{SC}} c(L^s, d_s)$. In this section, we want to derive a low upper bound for the optimality gap $c(L^*, \mathcal{D}) - c(L^{**}, \mathcal{D})$.

First, we define the notion of *total loss*:

Definition ((total) loss): Let L be a solution for the entire cluster. The *total loss* of solution L is the difference between $c(L, \mathcal{D})$ and $\sum_{s \in \mathcal{SC}} c(L_s, d_s)$. We can compute this total loss by summing the individual

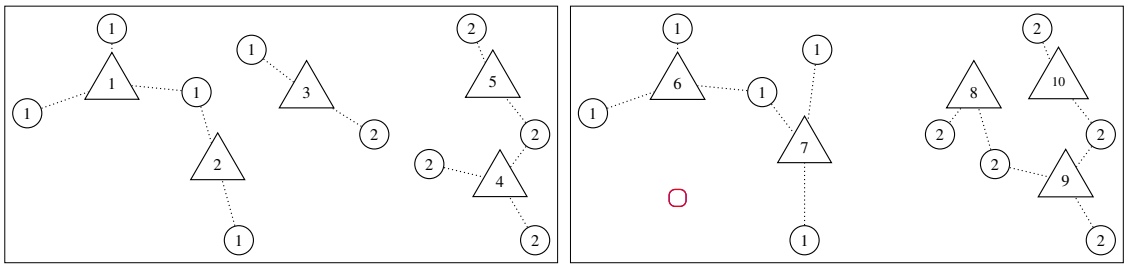

Figure 2 **Illustration of different solutions, where a $\triangle$ represents a facility and a $\bigcirc$ represents a demand location. A dotted line ($\cdots$) means that the facility covers (part of) the people living in that demand location. Left: Optimal solution for the situation without subclustering, $L^* = \{\triangle_1, \triangle_2, \triangle_3, \triangle_4, \triangle_5\}$. Consequently, $L_1^* = \{\triangle_1, \triangle_2\}$ and $L_2^* = \{\triangle_3, \triangle_4, \triangle_5\}$. Right: Optimal situations per subcluster (blue and red box) given the budgets that correspond to the sizes of L_1^* and L_2^*. Thus, $L^1 = \{\triangle_6, \triangle_7\}$ and $L^2 = \{\triangle_8, \triangle_9, \triangle_{10}\}$.**

losses of all $l \in L$, where such an individual loss of facility l, $loss(l)$, is the maximum amount of people that are not covered at facility location l, due to subclustering. In mathematical terms:

$$total\_loss(L) = \sum_{l \in L} loss(l), \text{ where } loss(l) = \sum_{s=1}^{|\mathcal{SC}|-1} u_{(s)}^l,$$

where $u_{(i)}^l$ represents the i^{th} smallest element of the values $\{c(l, d_1), c(l, d_2), ..., c(l, d_{|\mathcal{SC}|}\}$.

To illustrate this definition a little more, consider again the previous example. Here the optimal solution of the entire cluster, L^*, is equal to $\{\triangle_1, \triangle_2, \triangle_3, \triangle_4, \triangle_5\}$. Moreover, recall that $L_1^* = \{\triangle_1, \triangle_2\}$ and $L_2^* = \{\triangle_3, \triangle_4, \triangle_5\}$. When we apply subclustering and then combine the obtained individual solutions, one possible solution is $L_1^* \cup L_2^*$, with a minimal coverage of $c(L_1^*, d_1) + c(L_2^*, d_2)$. It is important to note that in this scenario, the facilities in L_1^* only cover demand locations in d_1, and those in L_2^* cover only demand locations in d_2, as solutions of subclusters do not allow facilities to cover people from different subclusters. Note that the term *minimal* coverage is used because the actual coverage could potentially be higher if people in subcluster 1 are still covered by $\triangle_3$. Now, the difference in coverage between this solution and the solution that is obtained when we do not use subclustering is the number of people that, in the situation without using subclustering, are assigned to a facility that is not in their corresponding splitted solution, L_s^*. In this case, this is the number of people in subcluster 1 who would have been covered by $\triangle_3$, as $\triangle_3 \notin L_1^*$. This gap represents the *loss* of $\triangle_3$ under the subclustered solution. Note that only facilities like $\triangle_3$, which in the non-subclustered solution, cover people from more than one subcluster, have a positive loss. In this specific example the *total loss* is therefore equal to the loss of $\triangle_3$.

Let us generalize this example a little bit. For now, we represent each facility $l \in L^*$ which covers people from different subclusters with a pie chart. Each color in the pie chart represents a subcluster, and the number inside each slice of the circle chart represents the number of people that are covered by this facility. An example situation is visualized in Figure 3.

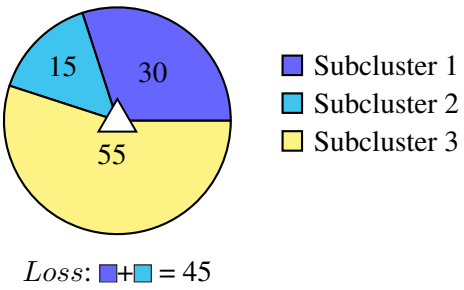

Figure 3 **An illustration of a potential facility location that may cover people in different subclusters. When first subclustering, this facility can only cover people from a single subcluster. Maximizing the coverage, this is the yellow subcluster. Therefore, in this case, the loss of this facility would be 45. Since there is only one facility with a positive loss, the total loss of the original solution would also be 45 people.**

When we apply subclustering and then combine the solutions, one possible solution is $L_1^* \cup L_2^* \cup L_3^*$, with a minimal coverage of $c(L_1^*, d_1) + c(L_2^*, d_2) + c(L_3^*, d_3)$. In this scenario, the facility shown in Figure 3 only covers demand locations in d_3, as solutions of subclusters do not allow facilities to cover people from different subclusters. Therefore, the *loss* of this facility consists of the people that were originally covered by this facility, but not necessarily anymore after subclustering, the 30 people from subcluster 1 and the 15 people from subcluster 2.

This reasoning generalizes to multiple facility locations. Consider the situation in Figure 4, where we assume that there are now three potential facility locations that can cover people from different subclusters in a specific solution.

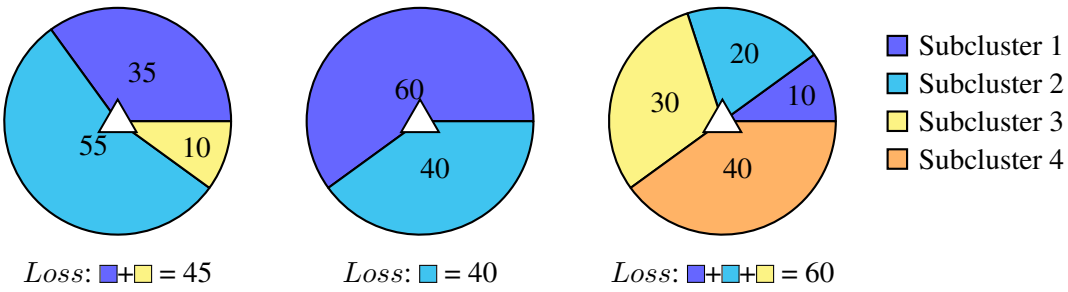

Figure 4 **An illustration of multiple potential facility locations that may cover people in different subclusters. When first subclustering, each facility can only cover people from a single subcluster. Maximizing the coverage, these are the cyan, blue, and orange subcluster, respectively. Therefore, in this case, the total loss of the solution would be $45 + 40 + 60 = 145$ people.**

In this case, the total loss of the original solution for the entire cluster is the sum of the individual losses ($= 45 + 40 + 60 = 145$) people.

Using the definitions introduced above, we state the main theorem that gives an upper bound on the gap between the optimal solution of a cluster and the solution consisting of the union of optimal solutions for subclusters:

Theorem (optimality gap):

$$c(L^*, \mathcal{D}) - c(L^{**}, \mathcal{D}) \leq \max_{U \subseteq \mathcal{L}} \{total\_loss(U)\},$$

where the maximization problem can be solved efficiently, *i.e.*, it is the optimal objective value of the following linear optimization problem. Let a_d represent the number of people at demand location d, and let $p_{ds} = 0$ for $d \in d_s$, and $p_{ds} = a_d$ for $d \notin d_s$, $\forall d \in \mathcal{D}, s \in \mathcal{SC}$. Then, using decision variables t_l and y_{dl} for all $l \in \mathcal{L}^{REL}, d \in \mathcal{D}^{REL}$, the maximization problem can be formulated as follows:

$$\max \left\{ \sum_{l \in \mathcal{L}^{REL}} t_l \;\middle|\; \begin{array}{l} \sum_{l \in \mathcal{L}^{REL}} y_{dl} \leq 1 \quad \forall d \in \mathcal{D}^{REL}, \\ \sum_{d \in \mathcal{D}^{REL}} p_{ds} y_{dl} \geq t_l \quad \forall s \in \mathcal{SC},\ l \in \mathcal{L}^{REL}, \\ 0 \leq y_{dl} \leq 1 \quad \forall d \in \mathcal{D}^{REL},\ l \in \mathcal{L}^{REL} \end{array} \right\}.$$

B ased on the definitions introduced above, we can write the following:

$$\begin{aligned} c(L^*, \mathcal{D}) &= \sum_{s \in \mathcal{SC}} c(L^*_s, d_s) + total\_loss(L^*), \\ &\leq \sum_{s \in \mathcal{SC}} c(L^*_s, d_s) + \max_{U \subseteq \mathcal{L}} \{total\_loss(U)\} \\ &\leq \sum_{s \in \mathcal{SC}} c(L^s, d_s) + \max_{U \subseteq \mathcal{L}} \{total\_loss(U)\} \\ &\leq c(L^{**}, \mathcal{D}) + \max_{U \subseteq \mathcal{L}} \{total\_loss(U)\}. \end{aligned}$$

Note that $total\_loss(L^*) = \sum_{l \in L^*} loss(l)$. Note that the first equality follows from the definition of the total loss. The first inequality is true because L^* is a specific solution with a corresponding total loss. It is part of the feasible region of the proposed maximization problem. The second and third inequality follow readily from the definitions introduced before. Now, we can rewrite the last inequality as:

$$c(L^*, \mathcal{D}) - c(L^{**}, \mathcal{D}) \leq \max_{U \subseteq \mathcal{L}} \{total\_loss(U)\},$$

which proves the first part of the theorem. Next, we show how to find this quantity $\max_{U} \{total\_loss(U)\}$. In other words, we are looking for a solution, a set of open facilities and an assignment of people to these

open facilities, such that we maximize the total loss. Since the loss is zero if a facility covers only people from a single subcluster, we are only interested in opening facilities that can potentially cover people from different subclusters. As a consequence, we are only interested in the people that can be covered by such a facility location. Mathematically, we define the following two, *relevant*, sets:

$$\mathcal{L}^{REL} = \{l \in \mathcal{L} \mid l \text{ can cover people in at least 2 subclusters}\} \subseteq \mathcal{L},$$
$$\mathcal{D}^{REL} = \{d \in \mathcal{D} \mid d \text{ can be covered by a facility in } \mathcal{L}^{REL}\} \subseteq \mathcal{D}.$$

Only considering these *relevant* sets significantly reduces the scale of the maximization problem we are trying to solve. Regarding this maximization problem, let a_d be the amount of people that belong to demand location d. Then, we define, for each demand location $d \in \mathcal{D}$, and for each subcluster $s \in \mathcal{SC}$, p_{ds} as follows:

$$p_{ds} = \begin{cases} 0 & d \in d_s \\ a_d & d \notin d_s \end{cases}$$

This parameter indicates the maximum loss we can get when we do not cover demand location d from the subcluster it belongs to. Moreover, let y_{dl} be a variable that indicates how much percent of demand location d is assigned to potential facility location l. Note that we can only assign people to facilities if they are in their vicinity. Otherwise, we set y_{dl} equal to 0. Now, suppose that we have an assignment of people to potential facility locations (this can be obtained from the variable y_{dl}). For each open facility location l, we choose to cover the people from the subcluster that results in the maximum coverage (or minimum loss). Here, we refer back to Figures 3 and 4, highlighting the selection of the subcluster that yields the largest coverage. Mathematically, choosing to cover the people of subcluster s would result in a (hypothetical) loss for this facility location l of $\sum_{d \in \mathcal{D}^{REL}} p_{ds} y_{dl}$. Therefore, the (actual) loss that corresponds to potential facility location l, t_l, can be defined as

$$t_l = \min_s \left\{ \sum_{d \in \mathcal{D}^{REL}} p_{ds} y_{dl} \right\}.$$

The maximization model in mathematical terms is shown below:

$$\begin{aligned}
\max \quad & \sum_{l \in \mathcal{L}^{REL}} t_l, \\
\text{s.t.} \quad & \sum_{l \in \mathcal{L}^{REL}} y_{dl} \leq 1 && \forall d \in \mathcal{D}^{REL}, && \text{(C.1)} \\
& \sum_{d \in \mathcal{D}^{REL}} p_{ds} y_{dl} \geq t_l && \forall s \in \mathcal{SC},\ l \in \mathcal{L}^{REL}, && \text{(C.2)} \\
& 0 \leq y_{dl} \leq 1 && \forall d \in \mathcal{D}^{REL},\ l \in \mathcal{L}^{REL}. && \text{(C.3)}
\end{aligned}$$

Constraint (C.1) ensures that every individual can be assigned to only one potential facility location. This

ensures that we do not double count people in losses of different potential facility locations. Constraint (C.2) defines t_j, the loss of potential facility location j. Finally, the objective maximizes the sum of the losses of each of the potential facility locations. In short, we assign people to potential facility locations in such a way that we maximize the total loss.

Finally, computing and summing the maximum losses for every cluster will give us a maximum loss for the entire problem. If we divide this amount of people by the total amount of people in the problem, we obtain the maximum gap between the optimal and the found coverage in percentages. Note that we are able to find better (lower) bounds for when there are less potential facility locations opened (*e.g.*, due to a budget). This results in a bound for each individual budget. However, this will take more time to compute, especially for situations with a lot of subclusters. Therefore, we propose to only calculate this bound without using a budget.

4. Case study West Darfur

This section presents a case study conducted in collaboration with 510. We start by describing the data and any preprocessing steps performed (Section 4.2). In Section 4.3, we discuss the results of solving the huge-scale facility location problems faced by 510, using the proposed decomposition approach. In Section 4.4, we analyze the optimality guarantees for the solutions we obtained. Finally, in Section 4.5, we perform analyses on the runtime of the whole procedure.

4.1. Context of case study

This is done in collaboration with 510, an initiative of the Netherlands Red Cross, that focuses on using data and digital to enhance humanitarian aid and disaster response (Red Cross 510 (2022)).

Within their thematic area Water and Landscape, 510 aims to create insights in water related data, ultimately guiding response to reduce negative impacts on people's lives. The lack of access to clean drinking water is a challenge that millions of people worldwide are facing World Health Organization (2023). Insufficient infrastructure, contamination of water sources, and compromised water quality are issues that many communities need to deal with. Moreover, the effects of climate change, both in the form of drought and floods, increases water scarcity and water contamination, which catalyses the impact on already vulnerable populations Andrade et al. (2018). This lack of clean drinking water contributes to the spread of waterborne diseases, conflict, and hampers economic development Levy and Sidel (2011). The sixth goal of the United Nations Sustainable Development Goals aims to ensure availability and sustainable management of water and sanitation for all. Target 6.1 states it aims to achieve universal and equitable access to safe and affordable drinking water for all by 2030, and is indicated by the proportion of population that safely uses managed drinking water services (United Nations (2023)). Ensuring the optimal coverage of new water facilities is essential to reach this target.

For this paper, we focus on the state of West Darfur, Sudan, which has faced continuous challenges of food-insecurity, degradation of water resources, and conflict Unicef (2022). The Netherlands Red Cross, being one of their long-term partners of the Sudanese Red Crescent, has been involved in a project in this area, addressing challenges in, amongst others, food-insecurity, water and sanitation, and natural resource management. One of the goals in the field of water and sanitation, is better access to clean water. In vulnerable areas like these, waterpoints are vital for ensuring such access to clean drinking water Lapworth et al. (2020). Establishing reliable and conveniently located waterpoints helps communities in West Darfur to secure a sustainable and safe supply of drinking water. Therefore, a budget has been secured within the project to drill new water wells. It is essential to place new water wells strategically to increase their impact on the population coverage. Therefore, 510 has been tasked to select potential locations for water wells, maximizing the number of households that live in the vicinity of clean drinking water. Within this paper, we refer to a borehole as a water well, since the term borehole is not commonly used outside the field of WASH humanitarian aid and might confuse the layman reader. Boreholes are deep holes drilled into the ground to access groundwater, and are often equipped with a handpump Lapworth et al. (2020).

4.2. Data and preparation

In this case study, individuals are considered *covered* when they are assigned to an operational water well located within a 500-meter radius, as measured by Euclidean distance. In this case study we use data from WorldPop (Bondarenko et al. 2020). In total, there are 762,887 people spread out over 89,379 demand locations. The Red Cross is specifically interested in the *coverage curves* for this area, such that they can examine the relation between installing additional water wells and the added coverage this will cause. Actual solutions, spatial configurations of water wells, can then be extracted, thus maximizing coverage while staying within the limits of a predetermined budget for water well placements.

To find optimal locations for water wells, we first need to find a set of potential water well locations in the area. Since there is no information about which locations are suitable for a water well, the idea is to generate many potential water well locations, distributed uniformly across the designated area. Then, we determine which locations are optimal, and then experts on the ground can see whether it is actually possible to build water wells at the proposed (optimal) locations. If some of the locations turn out to be infeasible, we do the optimization again, where we now exclude the infeasible locations. This is done until all proposed optimal locations are feasible. The consensus is that this is more efficient than first investigating which locations are potentially good candidates for a water well, as there are many potential water well locations.

Note that generating potential water well locations is done for each cluster. For clusters of size one and two, the potential water well location is at the demand location or in between the two demand locations, respectively. For any cluster consisting of more that two demand locations, we generate potential water well

locations on a grid of size 250 meters. This procedure leads to a large number of potential water well locations. However, if we are considering the uncapacitated case, we can exclude certain potential water well locations from consideration as they can never be part of an optimal solution. To be more precise, we do not consider potential water well locations that are outperformed by others, which cover the same household locations and possibly more. Note that, in the uncapacitated setting, it is never optimal to install a water well in such a location, because there exists another location that covers all the same demand locations and more. This reduction of potential water well locations we need to consider simplifies the solution process for the uncapacitated version significantly. Note that in the capacitated setting, we cannot remove these locations. For instance, consider the situation in which all potential dominating points are already at their full capacity, and we still have budget for an additional facility to cover other demand locations. Then, we have to use a potential facility location that is dominated by an (already opened) dominating facility location.

In Figure **??**, we illustrate creating the grid of potential water well locations, and the corresponding dominating points, for an example cluster.

Note that when we only use the dominating points, this leads to a smaller sized problem. In Table 1, we report the size of the problem when using the all potential water well locations, and when using only the dominating points.

# Demand locations	# Potential water well locations	
	Only dominating	*All*
89,379	38,267	158,272

Table 1 Final problem sizes of the facility location problem.

It is important to recall that we can only reduce the amount of potential facility locations for the uncapacitated setting. This means that we can only make the uncapacitated facility location problem easier to solve, while the capacitated version is typically more difficult to solve (due to its larger amount of variables). In the remainder of this chapter, we use for the uncapacitated facility location problem, only the dominating points and for the capacitated version, all potential water well locations.

4.3. Application of the decomposition approach

The first step in applying the proposed decomposition approach is clustering the population dataset, using the DBSCAN algorithm. Note that this clustering procedure still guarantees that we find the optimal solution to the original problem after integrating the results of the subproblems. Using the DCSCAN algorithm, we get a set of 887 different clusters. These clusters are visualized in Figure 5. In Table 2, we show the number of household locations, and the number of potential water well locations for the largest 5 clusters in terms of the number of household locations they contain. Recall that for the uncapacitated

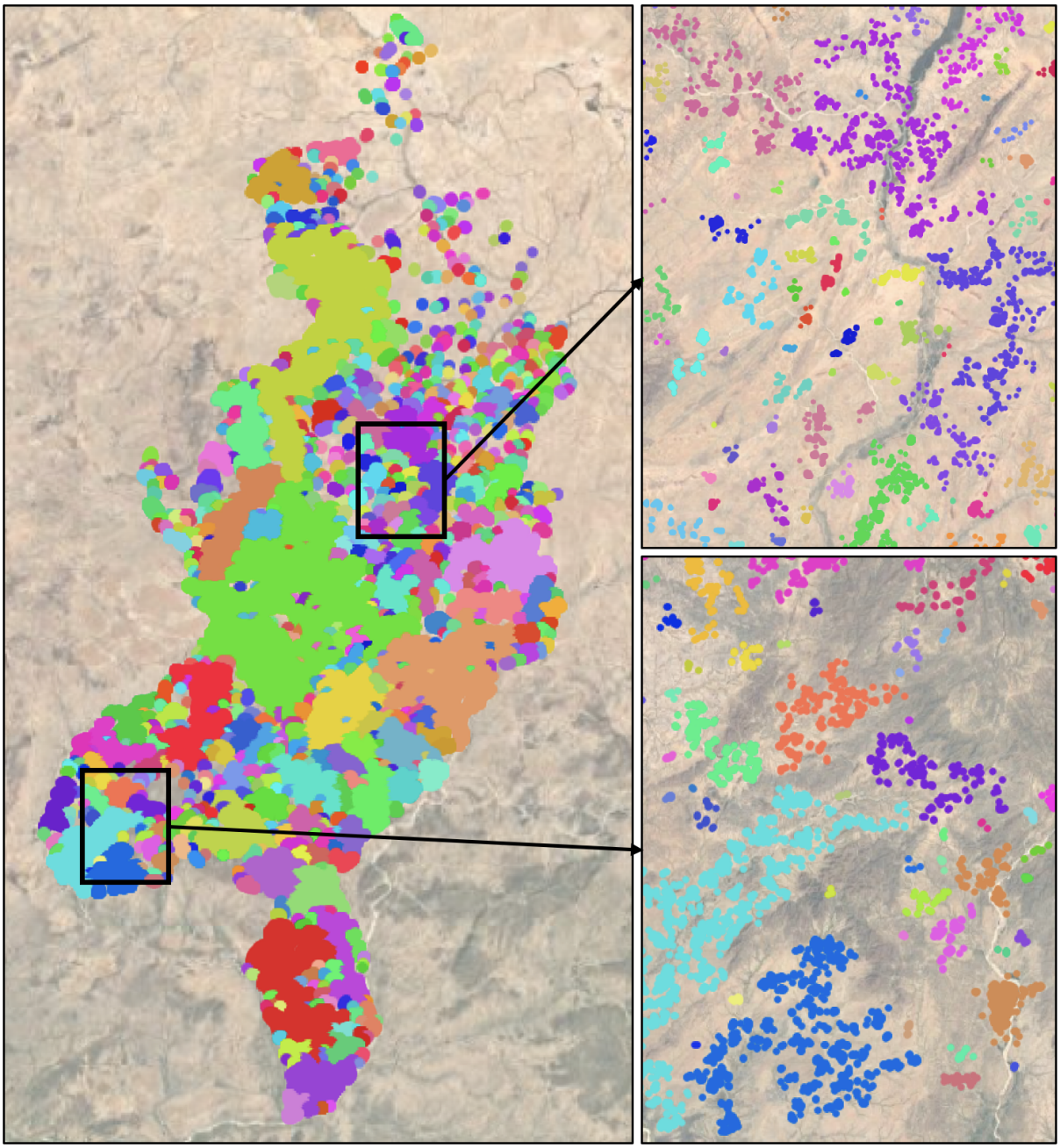

Figure 5 **The results of the first round of clustering, using the DBSCAN algorithm with $\epsilon = 1000$ and the minimum number of points per cluster equal to 1. Each color represents a different cluster.**

facility location problem, we can remove the dominated locations, but for the capacitated version we can not (if we want to guarantee finding the optimal solution).

We need to keep in mind that the number of variables needed in a capacitated facility location can potentially be equal to the number of household locations multiplied with the number of potential water well locations. On the other hand, many of the variables can be set to zero, when they household locations

Cluster order	Household locations	Potential water well locations	
		Only dominating	*All*
0	18,636	7,325	22,309
1	6,917	3,646	13,610
2	4,066	1,893	8,455
3	3,771	2,042	6,740
4	2,651	702	2,129

Table 2 Characteristics of the clusters. For the number of potential water well locations, we distinguish between not removing the dominated locations and keeping the dominated locations

are not in the vicinity of potential water well location. We observe that especially for the first few clusters, still many variables need to be incorporated, for which we do not have the computational resources. Therefore, we subcluster some of the larger clusters.

When using subclustering, we need to specify the maximum allowed number of household points within a cluster. Evidently, the higher this number, the less subclusters will be created. In Table 3, we show how many subclusters are created for different maximum cluster sizes. Moreover, we show how many clusters we, in the end, have (this incorporates all subclustering).

Cluster order	Maximum cluster size			
	10,000	5,000	3,000	1,000
0	2	4	8	32
1	0	2	4	8
2	0	0	2	8
3	0	0	2	4
4	0	0	0	4
Total number of clusters	888	891	899	954

Table 3 Characteristics of the number of subclusters for the largest clusters (in terms of number of household locations).

In this case study, we decide to use for the uncapacitated a version clusters of at most 3,000 household locations, and for the capacitated version, we want a maximum of 1,000 household locations per clusters. Recall that if we do not need to incorporate capacity, we can solve the maximum covering problem instead. This requires significantly fewer variables and, therefore, we can use larger clusters for the uncapacitated version. After applying subclustering we thus have 899 clusters for the uncapacitated setting, and 954 clusters for the capacitated setting.

Next, we create a coverage curve for each of the clusters. In other words, we determine, for each cluster, the maximum number of people we can cover, given a number of water wells we are allowed to install (our budget). As an example, Figure 6 shows the coverage curves for the first 10 clusters in the capacitated setting. Note that these clusters are the result of the subclustering procedure. This, in turn, means that all the clusters shown are subclusters derived from the initial Cluster 0, as we subclustered this specific cluster into 32 distinct subclusters. Next, as described in Section 3.2, these coverage curves are integrated

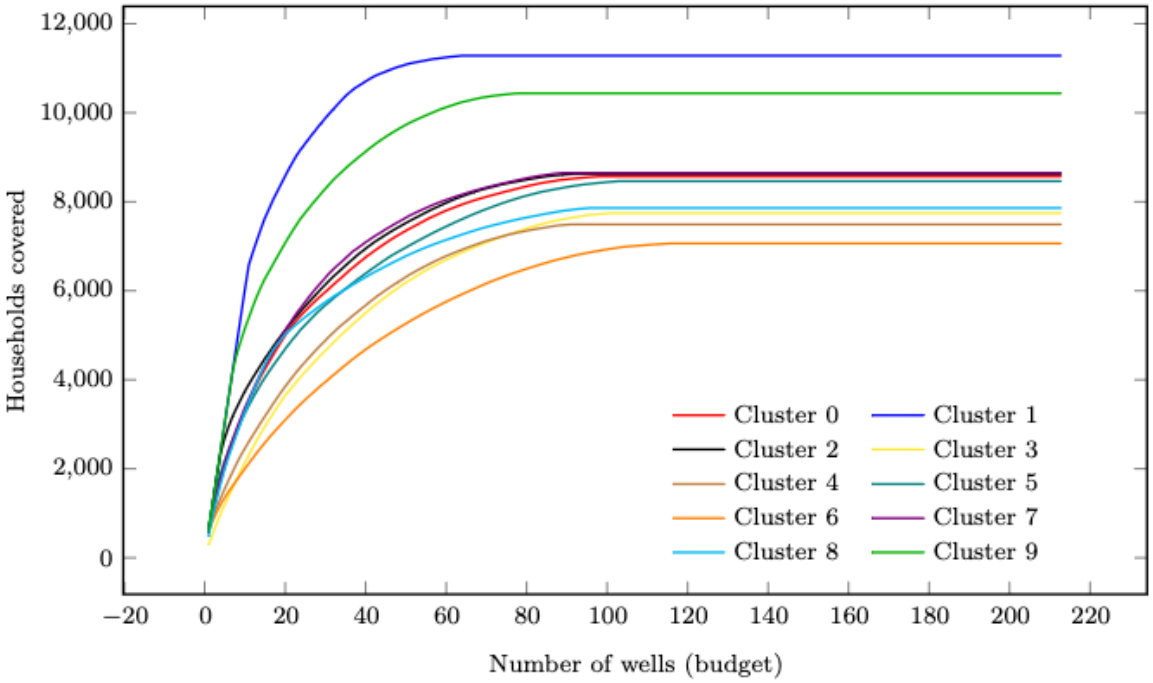

Figure 6 **Coverage curves for the first 10 (out of 954) clusters of the capacitated maximum covering problem, with a maximum cluster size of 1,000 demand locations. Note that these clusters are the result of subclustering, meaning they are all derived from the initial Cluster 0, as this cluster was subclustered into 32 subclusters.**

to determine the coverage curve for the overall facility location problem. In Figure 7, we show for both the capacitated and uncapacitated facility location problem, these overall coverage curves. Moreover, we also show, in Figure 8, the number of people that are served extra compared to the solution with one well less.

We observe that in Figure 7 both curves look very similar. There is, however, a difference at the start, where the blue line (capacitated version) is a straight line with slope of 600 people ($\approx 0.08\%$ coverage), due to the fact that one water well can only serve 600 people. This effect is more clearly visible in the right figure, where we plotted the number of water wells that is added compared to the solution with one less water well. We see that for the line that corresponds to the capacitated version, in the beginning, we keep

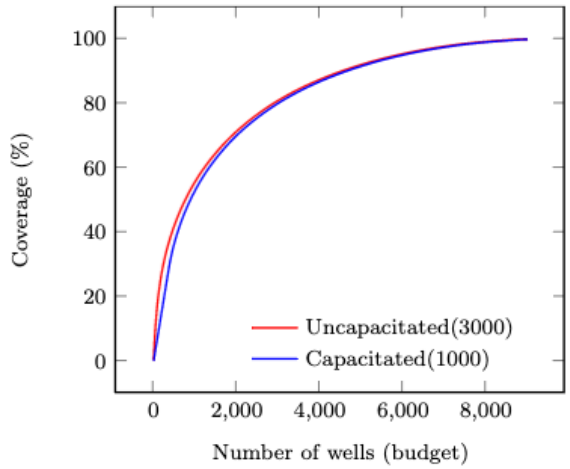

Figure 7 **Coverage curves for the uncapacitated and capacitated facility location problem.**

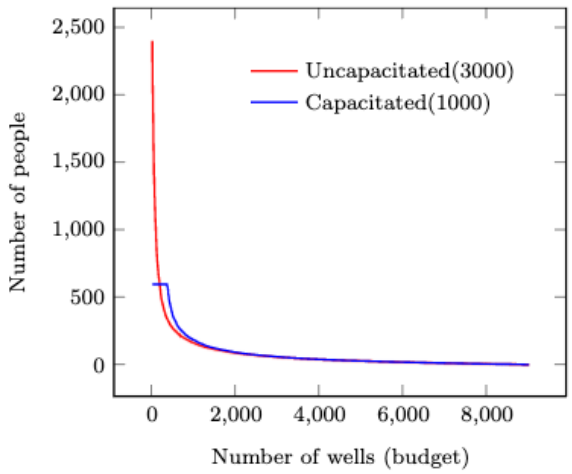

Figure 8 **Added number of people when increasing the budget with one well.**

adding wells that reach their maximum capacity (600 people). On the contrary, the line that corresponds to the uncapacitated version starts much higher, which means that, in this case, more than the 600 people per water well are covered.

Recall that subclustering means that we cannot guarantee optimality anymore. In the next section, we perform experiments to investigate the impact of this subclustering. Specifically, we compute upper bounds on the gap between the obtained solutions and the optimal solution of the original (unclustered) situation.

4.4. Optimality gap performance experiments

Next, we compute upper bounds on the optimality gaps for the (un)capacitated facility location problems discussed in this case study. Note that if we do not subcluster at all, this gap is zero. As we generate more subclusters, problems can be solved more quickly and may incorporate more complexity, such as including capacity considerations. However, as shown in Table 4 this comes at the cost of an increase on the upper bound of the the optimality gap, which may yield suboptimal solutions.

	Maximum cluster size			
	10,000	5,000	3,000	1,000
Total number of clusters	888	891	899	954
Upper bound	0.8%	1.4%	1.5%	5.4%

Table 4 **Upper bounds for maximum numbers of household locations per cluster (subclustering setting).**

Note that in our case, we used for the uncapacitated model a maximum of 3,000 household locations per

cluster, resulting in a gap of at most 1.5%. For our capacitated version, we used a maximum of 1,000 household locations per cluster, resulting in a gap of at most 5.4%.

Next, we would like to show that the worst case bounds we provide tend to overestimate actual optimality gaps. We perform several experiments that show that the actual gap is much smaller than the proposed upper bound. To illustrate this, we use cluster 1, 2 and 3 in the uncapacitated setting. We compare the actual and the upper bound for these three different clusters. Note that we do not include cluster 0 (the largest in terms of household locations), as we can not obtain an exact solution for this cluster. Similarly, for the capacitated version, we are not able to obtain an exact solution. Without this optimal solution, we can not make a comparison with the proposed upper bounds. Note that comparing even smaller clusters is feasible, however, this may result in subclusters containing very few household locations, requiring fewer and fewer water wells. This, in turn, may distort the performance evaluation, as this is typically not what happens in practice. Therefore, we do the comparison with cluster 1, 2 and 3 for the uncapacitated case. We do expect similar performance for the capacitated versions.

For cluster 1, 2 and 3, in the uncapacitated setting, we compute the coverage curve in an exact way, and we create the coverage curve using the proposed decomposition approach, in which we first perform subclustering. We distinguish 3 different subclustering settings, whether we split the cluster in 2, 4, or 8 subclusters. Next, we subtract these solution from each other to find the *actual gap* between the optimal solutions and the solutions obtained via the proposed decomposition approach. Next to this actual gap, we also compute the upper bounds for the correspoding solutions that are based on the proposed decomposition approach. The results are shown in Figure 9.

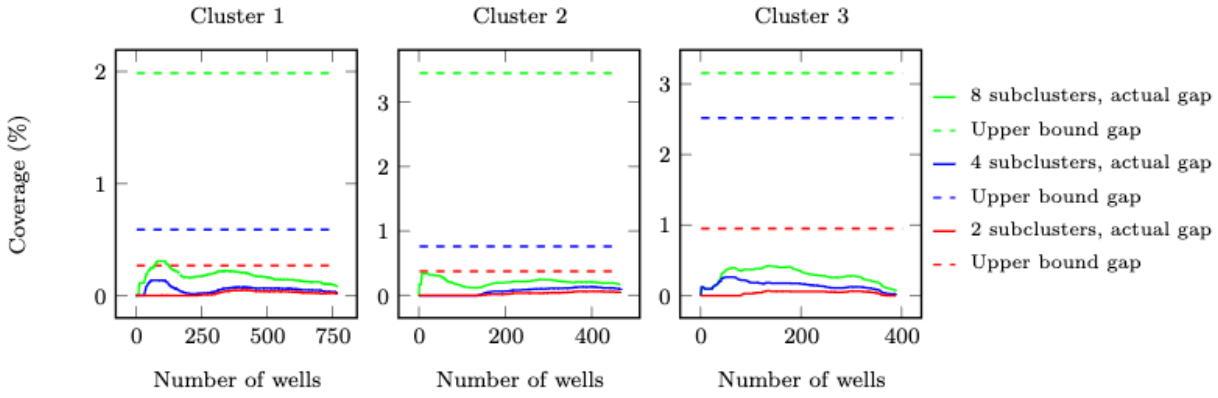

Figure 9 **Optimality gap analyses using clusters 1, 2 and 3, in the uncapacitated setting.**

First, we indeed observe, as expected, that the situation in which we obtain more subclusters leads to a higher upper bound on the gap, and also to a higher actual gap. However, we do see that the actual gaps are

very small compared to their respective upper bounds. Note that this makes sense, as it is not very likely that the precise boundaries between subclusters play an important role when optimizing the unclustered facility location problem. When we construct the worst case scenario, we extensively leverage this region. Therefore, we may end up with highly inefficient solutions. In short, the worst case bounds we offer provide a guarantee, but in practice, the actual gap is likely to be much smaller.

4.5. Computation time

Finally, we discuss the computation time of the experiments. As the proposed algorithm will be used for practical decision-making in uncertain environments, where it is essential to execute, verify, and fine-tune many scenarios, the computation time of the algorithms is especially important. Long waiting periods between runs are impractical and could diminish the likelihood of implementation. Note that it is worth mentioning that with the proposed approach, we are now able to solve very large (capacitated) facility location problems, as we can create smaller and smaller subproblems. Note that it depends on available computational resources how many subproblems are required. The larger the maximum size of subclusters, the lower the upper bound on the optimality gap, and the better the heuristic solutions will be.

The experiments are run on a PC with 24 CPUs that have a clock rate of 2.9 GHz and with 201 GB RAM. We do this analysis for all individual parts of the proposed procedure: the clustering part, creating the coverage curves, and combining the cluster solutions. We show the results for both the uncapacitated setting, and for the capacitated version.

The first step of the proposed procedure is to cluster the demand points using the DBSCAN algorithm. In our case, this algorithm takes less than a second. Secondly, we discuss the time it takes to create the set of potential water well locations. Here, we differentiate between the situation where we also want to remove all the dominated potential locations (which can be done in the uncapacitated setting) and the case in which we want to keep these locations (when considering the capacitated version). Note that in this time, we also find out which potential water well locations are in the vicinity of which household location(s), as this is part of the information of the potential water well locations. The computation times of both these operations are tabulated in Table 5.

We observe that it takes relatively long to filter out the dominated points. However, the result is that we are potentially able to solve more complicated facility location problems, which is especially useful when trying to solve huge-scale facility location problems.

Next, we discuss the subclustering part. We compare the computation times for both the case in which we use 3,000 (used for the uncapacitated setting), and 1,000 (used for the capacitated setting) as maximum number of household locations per cluster. The results are shown in Table 6.

Cluster order	Find potential water well locations Remove dominated	 Not remove dominated
0	147.5	0.4
1	48.4	0.3
2	16.8	0.2
3	13.9	0.1
4	1.5	0.1
Total runtime	276.3	19.0

Table 5 Computation times (in minutes) of determining the potential water well locations.

Cluster order	Subclustering Uncapacitated	 Capacitated
0	13.9 (8)	14.6 (32)
1	1.4 (4)	1.5 (8)
2	0.3 (2)	0.5 (8)
3	0.3 (2)	0.4 (4)
4	- (0)	0.2 (4)
Total runtime	15.9	17.9

Table 6 Computation times (in minutes) of the subclustering part of the approach. For clarity, we show the number of subclsuters that is created per cluster between parentheses.

We conclude that cluster 0 is a real outlier regarding the time. This mainly has to do with the fact that we need 32 subclusters to satisfy the maximum cluster size restriction.

The part that takes the largest amount of computation time is creating the coverage curves of all individual clusters. We do, however, have the option to adjust the MIP gap when solving the optimization problems. The MIP gap is an upper bound on the actual MIP gap of the final solution. The MIP solver will terminate when the current lower bound can only be improved by less than *MIP gap* percent. In the results shown when discussing the case study, we used a MIP gap of 0.0001 (or 0.01%). For comparison, we also ran experiments when use a MIP gap of 0.001 (or 0.1%). We do note that the results obtained with a MIP gap of 0.001 are very similar to the results obtained with a MIP gap of 0.0001. In Table 7, we show the computation times of the largest clusters, distinguishing between the two different MIP gaps. Recall that for the uncapacitated facility location problem, we now with 899 subclusters. For the capacitated version, we deal with 955 subclusters.

Cluster order \ MIP GAP	Uncapacitated 0.001	Uncapacitated 0.0001	Capacitated 0.001	Capacitated 0.0001
0	100.7	175.5	29.6	29.8
1	1.6	3.7	45.7	817.1
2	0.3	0.4	34.2	41.0
3	15.4	57.6	20.7	21.9
4	0.7	4.0	24.4	25.2
Total runtime (in hours)	147.0 ($\approx$ 2.5h)	298.8 ($\approx$ 5.0h)	1,085.4 ($\approx$ 18.1h)	2,020.9 ($\approx$ 33.7h)

Table 7 Computation times (in minutes) of creating the coverage curves.

We indeed observe that this part takes quite some time. We also see that reducing the MIP gap to 0.001, we reduce the computation time significantly, which is mainly caused by the large reduction in computation time of cluster 1. More specifically, for cluster 1, it is relatively easy to find a solution within a maximum margin of 0.1% from the optimal solution. Using an even smaller maximum MIP gap is possible, but it comes at the cost of increased computation time. Note that the MIP gap relates to the number of people that are covered. As an illustration, in this case, cluster 1 consists of 11,291 individuals. By setting a gap of 0.001, we could potentially cover an additional 11.291 people in the worst-case scenario. Alternatively, with a gap of 0.0001, the worst-case scenario might yield coverage for an additional 1.1291 people. Therefore, we observe that having even smaller MIP gaps will most likely not be of practical relevance.

Although reducing the time it takes to compute these coverage curves is beyond the scope of this research, we do want to mention a few ways in which this can be achieved. One way to reduce this computation time is to only compute the starting point of these coverage curves. One could argue, for instance, that it will not be a solution to build a water well for less than 50 people. There is a moment in which allowing for one additional well only increases the number of covered people by less than 50. Incorporating such rules means that we can stop computing the coverage curve if the marginal increase in people covered drops below a certain threshold. Note that this assumes that the coverage curve is concave, but this is not necessarily always the case. It will, however, still be a good approximation. Another similar way is to minimize the number of times we need to find the coverage that corresponds to a given budget. In essence, each time we find a point of the coverage curve we are solving an optimization problem. There are ways to reduce this number of optimization problems (*e.g.,* see Ten Eikelder and van Amerongen (2023)). Finally, as already mentioned before, we used a MIP gap of 0.01%. We could increase this number to find slightly sub-optimal solutions for the coverage curves, but with a significantly reduced computation time.

Once we have all coverage curves of all the clusters, we need to combine these coverage curves using the dynamic programming method for the original (unclustered) situation. The dynamic programming method allows us to solve this for different numbers of available water wells (budget), so that we again get a complete coverage curve. Note that this approach works for the capacitated and uncapacitated setting. The capacities are already incorporated when determining the coverage curves for each cluster. The computation times of solving the binary integer programming problems are shown in Table 8.

	Uncapacitated	Capacitated
Avg. runtime per given budget	0.016	0.009
Total runtime (whole coverage curve)	143.4	86.4

Table 8 Computation times (in minutes) of the binary integer programming problems.

We observe that solving one binary integer programming problem takes almost no time. However, to create a complete coverage curve, we need to solve this problem almost 10,000 times, which quickly adds up to a more significant total computation time. Note that we may want to only obtain parts of the coverage curve, which will reduce the computation time required significantly. Moreover, we could obtain an approximation of the coverage curve by only solving the binary integer programming problem for budgets that are multiples of 10, 100, etc.

Finally, we discuss the time it takes to compute the upper bounds for the optimality gaps. The results are shown in Table 9.

	Provide upper bound	
Cluster order	Uncapacitated	Capacitated
0	22.7 (8)	61.4 (32)
1	2.0 (4)	2.1 (8)
2	1.0 (2)	1.2 (8)
3	0.5 (2)	0.7 (4)
4	- (0)	1.9 (4)
Total runtime	26.2	71.8

Table 9 Computation times (in minutes) of determining the upper bounds for the optimality gaps. For clarity, we show the number of subclusters that is created per cluster between parentheses.

We observe that determining the upper bound takes slightly more time than the subclustering part itself.

Especially for cluster 0, we see a significant amount of computation time required, compared to the other clusters. This makes sense, as we have to solve a relatively large optimization problem for cluster 0. There are many potential water well locations that may cover people from different subclusters, as there are 32 subclusters created from this cluster. Subsequently, there are many people that could be covered by such potential water well locations.

5. Conclusions and recommendations

In this paper, we propose a new decomposition approach that is able to solve huge-scale (capacitated) facility location problems that were previously intractable in an exact way by clustering demand locations. We solve subproblems for each cluster separately, and combine solutions of subproblems through binary integer optimization. Additionally, we propose an optional heuristic subclustering method that can be used to further divide clusters. We demonstrate the effectiveness and applicability of the proposed approach by performing a case study in collaboration with 510, on where to install new water wells in West Darfur (Sudan), such that the total coverage is maximized. Using the proposed decomposition approach we show that we can efficiently obtain solutions that are guaranteed to have a small optimality gap. The number of clusters we choose to use directly influences the memory and computation time needed to solve the problem. A large number of clusters results in a decreased memory requirement to store the relevant variables and a reduced computation time to find the coverage curves.

A key insight from this chapter is that when addressing huge-scale maximum coverage facility location problems that involve covering a population, leveraging natural population clusters through density-based clustering techniques (*e.g.*, DBSCAN) can provide significant computational advantages.

Potential lines for further research are, for instance, improving the subclustering method. This is especially important if many subclusters need to be created. For instance, if, after the first clustering procedure using DBSCAN, many clusters remain that need to be subclustered. Or if limited computational resources are available. In this case, a smaller subcluster threshold needs to be set, which means that many subclusters are likely to be needed. A large amount of subclusters means that we have many splits of clusters, which might slightly deteriorate the quality of the proposed solution. Note that in our case study we need a relatively small amount of subclusters, 12 in the uncapacitated setting and 67 in the capacitated setting, where in the uncapacitated setting we end up with 899 (sub)clusters and in the capacitated setting with 954 (sub)clusters (see also Table 2). Therefore, in our case, using other subclustering procedures will likely not have a significant effect on the results.

One way to improve this clustering algorithm is by allowing clusters to have slightly different sizes. Currently, our approach relies on a heuristic, which involves the alternating addition of demand locations to subclusters. By design, this algorithm results in equal-sized clusters. As an example, towards the end of the

current subclustering process, we might incorporate a more greedy approach, where we continuously add demand locations to their nearest subcluster without the need for alternation. It is worth noting that using such a heuristic might involve additional subclustering if larger clusters remain unsuitable in size. Alternatively, we could develop a local search algorithm to improve clusters after the subclustering step. Another option is to consider formulating the problem of subclustering as a Mixed Integer Optimization problem (MIO), such as a minimum cut problem. However, we need to keep in mind that such approaches likely require more computation time. In short, there is a trade-off between the complexity and computation time of subclustering, and the performance benefits it presents.

References

Ajisegiri, B., Andres, L. A., Bhatt, S., Dasgupta, B., Echenique, J. A., Gething, P. W., Zabludovsky, J. G., and Joseph, G. (2019). Geo-spatial modeling of access to water and sanitation in Nigeria. *Journal of Water, Sanitation and Hygiene for Development*, 9(2):258–280.

Andrade, L., O'Dwyer, J., O'Neill, E., and Hynds, P. (2018). Surface water flooding, groundwater contamination, and enteric disease in developed countries: A scoping review of connections and consequences. *Environmental pollution*, 236:540–549.

Antonissen, J., Krishnakumari, P. K., Theulen, F., Gromicho, J., Den Hertog, D., Kaiser, K., and Kant, G. (2023). Optimizing Geospatial Accessibility to Healthcare Services in Low- and Middle-Income Countries.

Atta, S., Mahapatra, P. R. S., and Mukhopadhyay, A. (2022). Solving a new variant of the capacitated maximal covering location problem with fuzzy coverage area using metaheuristic approaches. *Computers & Industrial Engineering*, 170:108315.

Bondarenko, M., Kerr, D., Sorichetta, A., and Tatem, A. (2020). Census/projection-disaggregated gridded population datasets for 51 countries across sub-saharan africa in 2020 using building footprints. `https://www.worldpop.org/sdi/publications/10044`.

Boonmee, C., Arimura, M., and Asada, T. (2017). Facility location optimization model for emergency humanitarian logistics. *International Journal of Disaster Risk Reduction*, 24:485–498.

Church, R. and ReVelle, C. (1974). The maximal covering location problem. In *Papers of the Regional Science Association*, volume 32, pages 101–118. Springer-Verlag.

Cordeau, J.-F., Furini, F., and Ljubić, I. (2019). Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research*, 275(3):882–896.

Deshpande, A., Miller-Petrie, M. K., Lindstedt, P. A., Baumann, M. M., Johnson, K. B., Blacker, B. F., Abbastabar, H., Abd-Allah, F., Abdelalim, A., Abdollahpour, I., et al. (2020). Mapping geographical inequalities in access to drinking water and sanitation facilities in low-income and middle-income countries, 2000–17. *The Lancet Global Health*, 8(9):e1162–e1185.

Dönmez, Z., Kara, B. Y., Karsu, Ö., and Saldanha-da Gama, F. (2021). Humanitarian facility location under uncertainty: critical review and future prospects. *Omega*, 102:102393.

ElKady, S. K. and Abdelsalam, H. M. (2016). A modified particle swarm optimization algorithm for solving capacitated maximal covering location problem in healthcare systems. *Applications of intelligent optimization in Biology and Medicine: current trends and open problems*, 96:117–133.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

Farahani, R. Z., Fallah, S., Ruiz, R., Hosseini, S., and Asgari, N. (2019). Or models in urban service facility location: A critical review of applications and future developments. *European Journal of Operational Research*, 276(1):1–27.

Galvão, R. D. and ReVelle, C. (1996). A Lagrangean heuristic for the maximal covering location problem. *European Journal of Operational Research*, 88(1):114–123.

Haghani, A. (1996). Capacitated maximum covering location models: Formulations and solution procedures. *Journal of Advanced Transportation*, 30(3):101–136.

Lapworth, D., MacDonald, A., Kebede, S., Owor, M., Chavula, G., Fallas, H., Wilson, P., Ward, J., Lark, M., Okullo, J., et al. (2020). Drinking water quality from rural handpump-boreholes in africa. *Environmental Research Letters*, 15(6):064020.

Levy, B. S. and Sidel, V. W. (2011). Water rights and water fights: preventing and resolving conflicts before they boil over.

Li, X., Zhao, Z., Zhu, X., and Wyatt, T. (2011). Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research*, 74:281–310.

Máximo, V. R. and Nascimento, M. C. (2019). Intensification, learning and diversification in a hybrid metaheuristic: an efficient unification. *Journal of Heuristics*, 25(4):539–564.

Máximo, V. R., Nascimento, M. C., and Carvalho, A. C. (2017). Intelligent-guided adaptive search for the maximum covering location problem. *Computers & Operations Research*, 78:129–137.

Megiddo, N., Zemel, E., and Hakimi, S. L. (1983). The maximum coverage location problem. *SIAM Journal on Algebraic Discrete Methods*, 4(2):253–261.

Pirkul, H. and Schilling, D. (1989). The capacitated maximal covering location problem with backup service. *Annals of Operations Research*, 18(1):141–154.

Pirkul, H. and Schilling, D. A. (1991). The maximal covering location problem with capacities on total workload. *Management Science*, 37(2):233–248.

Red Cross 510 (2022). Improving humanitarian action with data & digital. `https://www.510.global`. Accessed: 2023-10-24.

Resende, M. G. (1998). Computing approximate solutions of the maximum covering problem with grasp. *Journal of Heuristics*, 4(2):161–177.

Resende, M. G. and Ribeiro, C. (1998). Greedy randomized adaptive search procedures (GRASP). *AT&T Labs Research Technical Report*, 98(1):1–11.

Rodriguez, F. J., Blum, C., Lozano, M., and García-Martínez, C. (2012). Iterated greedy algorithms for the maximal covering location problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 172–181. Springer.

Senne, E. L., Pereira, M. A., and Lorena, L. A. (2010). A decomposition heuristic for the maximal covering location problem. *Advances in Operations Research*, 2010.

Ten Eikelder, S. and van Amerongen, J. (2023). Resource allocation problems with expensive function evaluations. *European Journal of Operational Research*, 306(3):1170–1185.

Unicef (2022). State profile: West darfur. `https://www.unicef.org/sudan/documents/state-profile-west-darfur`.

United Nations (2023). Sustainable development goal 6: Ensure availability and sustainable management of water and sanitation for all. `https://sdgs.un.org/goals/goal6#targets_and_indicators`.

Vatsa, A. K. and Jayaswal, S. (2021). Capacitated multi-period maximal covering location problem with server uncertainty. *European Journal of Operational Research*, 289(3):1107–1126.

World Health Organization (2023). Drinking-water. `https://www.who.int/news-room/fact-sheets/detail/drinking-water`.

Zarandi, M. H., Davari, S., and Sisakht, S. A. (2011). The large scale maximal covering location problem. *Scientia Iranica*, 18(6):1564–1570.

Zhai, C., Bretthauer, K., Mejia, J., and Pedraza-Martinez, A. (2023). Improving drinking water access and equity in rural Sub-Saharan Africa. *Production and Operations Management*.

Appendix A: Applying GRASP with path-relinking

Figure 10 shows the results of the application of the GRASP with path-relinking alrogithm as formulated in Antonissen et al. (2023) to West-Darfur. While solutions can be generated for different budgets, the running time does increase exponentially with the amount of facilities to place.

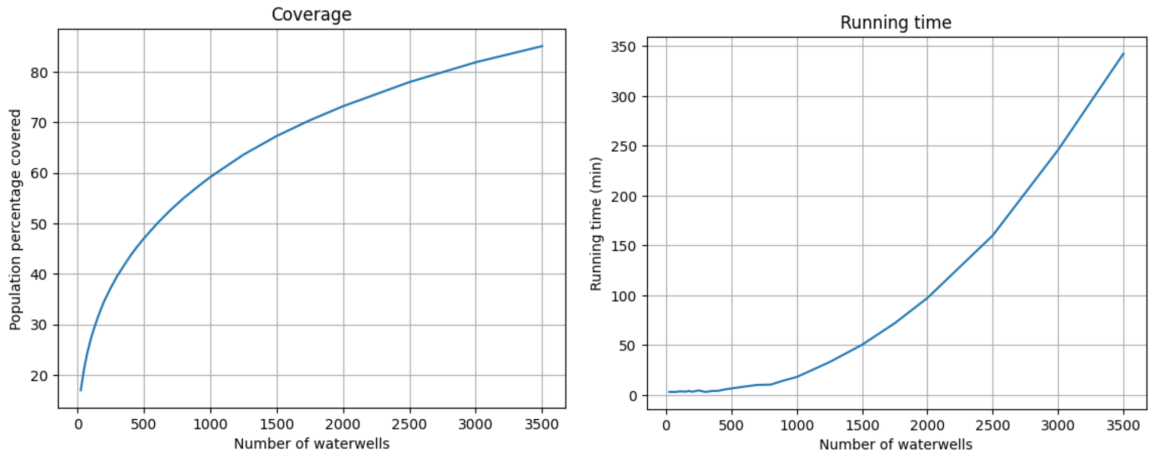

Figure 10 (left) Curve showing the trade-off between the coverage and the number of wells to place, (right) showing the trade-off between running time and number of wells to place.