American Journal of Operations Management and Information Systems
2025, Vol. 10, No. 1, pp. 13-37
https://doi.org/10.11648/j.ajomis.20251001.12

Review Article

A Systematic Review, Comparison and Juxtaposition of the (Dual Fitting and Factor Revealing LP Technique), (the ESMVERE CPLEX Algorithms) and the (Data Envelopment Analysis and TOPSIS Technique)

Emmanuel Siyawo Mvere*

Mechanical and Production Engineering Department, University of Mauritius, Moka, Mauritius

Abstract

The objective of the present paper is to compare three location science methods. Two of the solutions discussed are practical real-world case studies, while the other method is used to reveal the approximation factor of an algorithm. We conduct an in-depth analysis of the originality, similarities, and differences of the three methods to unravel clarity on the conditions where each method may be applied, or where one method might be preferred over another. (1) The Data Envelopment Analysis (DEA)-Technique for Order Performance (preference) by Similarity to Ideal Solution (TOPSIS) technique is usable in many other areas of management science besides location analysis, however it is used in this case to guide the decision of selecting the best locations among alternatives. Hierarchical groups DEA super-efficiency and group TOPSIS technique is applied on mobile money agents' locations in Harare Zimbabwe. (2) The ESMVERE algorithms introduce novel reformulations of the Uncapacitated Facility Location (UFL) and k-median problems which replace traditional distance matrices with Global Position System (GPS) based Euclidean Distance calculations. They are platform dependent solutions based on the CPLEX Optimization programming language (Opl). The solutions are used to solve a real-world problem which involve the near optimal siting of Hazardous Waste, Used Lead Acid Battery (ULAB), collection facilities in the Republic of Mauritius. (3) The Dual fitting with factor-revealing Linear Program (LP) technique is a location science approximation algorithm solution, used to analyze greedy algorithms that select facilities and connect the clients based on the dual solution information. The thrust of this review is to analyze the complexity, compare, contrast, and discuss the three location science techniques so as to classify them based on their purpose and area of application. Hence this review analyzes three papers and summarizes the main contributions of the three papers.

Keywords

DEA, TOPSIS, ESMVERE CPLEX UFL Problem Solver Algorithm, Dual Fitting, Factor Revealing LP

*Corresponding author: emmanuel.mvere@umail.uom.ac.mu (Emmanuel Siyawo Mvere)

Received: 4 June 2025; **Accepted:** 17 June 2025; **Published:** 1 August 2025

1. Introduction

The choice of the location of mobile banking facilities, Muvingi, Peer [3] or mobile fast-food outlets which involve the "preference" of the decision maker, against the choice of the location of mobile clinics needed to administer vaccines or medication during a pandemic, also against the choice of the selection of locations of facilities needed to control and safeguard the whole population of a country against contamination from the hazardous waste of lead poisoning Mvere [1, 2] is the main contrast of this discussion [1-3]. These real-world problems and solutions are contrasted against theoretical computer science techniques that guide facility location algorithm development, Jain, Mahdian [4] and Mahdian, Spielman [37] to determine their relationship, and to see why it may be extremely complex to apply some of the approximation algorithm techniques such as the Dual Fitting with approximation factor revealing LP technique to real-world problems [37]. This technique is used to reveal the approximation factor of an algorithm.

DEA measures relative performance of a set of producers or Decision-Making Units (DMUs) where the presence of multiple inputs and outputs makes comparisons difficult [5]. Hierarchical groups DEA super-efficiency (SE) and group TOPSIS technique is applied on mobile money agents' locations in Harare Zimbabwe [3]. It is used to make an optimal location analysis decision that incorporates multiple criteria such as, distance from the Central Business District (CBD), road density, zoning tariff, number of shops, number of mobile phone users, mobile transfer service users and population. It is a complex and time-consuming option, because it involves a lot of data gathering to process the decisions efficiently. However, to a greater extent, out of the three main location science techniques being reviewed in this paper [3, 4, 6] it is the least complex in terms of the complexity, or level of hardness of the calculations performed.

DEA in general is an LP based technique solved in polynomial (P) time for a single model [47]. TOPSIS is a deterministic, ranking -based method that also runs in P time [46]. However, in more advanced formulations such as the Hierarchical groups DEA, it "often" becomes, non-deterministic polynomial time hard (NP-hard). The UFL problem and the k-median problem are already proved NP-hard, which implies that if P≠NP there is no algorithm known, which can find an optimal solution in P time [1, 2, 4]. The level of hardness of the k-median problem is 1+2/e, proved by Jain, Mahdian [38]. The level of hardness of the UFL problem is 1.463, proven by Guha, Khuller [39], it was proven NP-hard even in metric spaces where triangle inequality holds.

Group TOPSIS remains polynomial in standard form but can approach NP-hardness in extended frameworks [46]. The formulas for Muvingi, Peer [3] are quite easy to understand, they merely involve an understanding of what's required at each stage, and an understanding of what each symbol represents so as to substitute them carefully and effectively with the data for calculation Figures 8, 9 and 10. These calculations compared to the lemmas of Jain, Mahdian [4] and Mahdian, Spielman [37] really bring out the differences in complexity, such as the complex calculations on Figures 1 and 2. The solution in Muvingi, Peer [3] involves a lot of data input, thus it may not be as user friendly as the solutions in Mvere [2] which simply involve the input of 3 types of data from the UFL problem which are GPS location coordinates of potential facility locations and client locations and the opening costs of the facilities. The solution in Muvingi, Peer [3] may also again not be as user friendly as the solutions in Mvere [1] which involve the input of 2 types of data from the k-median problem, which are GPS location coordinates of potential facility locations and client locations of the facilities. Thus Mvere [1, 2, 6] solves a more complex problem with less input arriving at a calculated solution faster.

However, to a greater extent, in many cases that involve marketing and business value, the DEA and TOPSIS method could be a better choice and closer to a more accurate solution over all others, because it involves all other non-financial criteria that may be involved in the selection of a facility location. The choice of the preferable facility location depends on multiple criteria such as social, economic, environmental, demographic and many other subjective factors that are quantified to objective data in the decision-making process. This results in an all-encompassing decision which ranks all the alternatives or potential location sites based on the evaluation of a set of chosen criteria, resulting in a ranked list of all the potential sites.

The Dual fitting and factor revealing LP technique and the ESMVERE algorithms are primarily focused on minimizing the distances on an instance plane of just clients and facilities. The decision for the best choice of location is based on the calculation of the UFL problem or the k-median problem, both of which aim to cover a set of client locations, with a set of optimal facility locations, calculated from a set of potential facility locations. The main difference between the two being that the UFL problem also involves the facility opening costs while the k-median problem does not involve the facility opening costs, it just covers a set of clients with a set of open facilities, Figure 14.

The two problems answer the question of which set of facility locations should be opened from the larger set of potential facility locations, to ensure that all the clients receive service from an open facility near them at a minimum cost throughout the plane, for all the clients in an instance. ESMVERE algorithms go a step further by using GPS coordinate input for real world solutions. They also embed the Euclidean distance formulae in CPLEX Optimization Programming Language (Opl) as a single unified algorithm, which contrasts the traditional distance matrices used as input to solve these problems for the past 60 years [24, 25].

Comparing the three methods in terms of originality, all

three are based on original conceptualization [3, 4, 6]. However, while Muvingi, Peer [3] credibly extends the existing DEA and TOPSIS formulae, Jain, Mahdian [4] and Mvere [1, 2, 6] are developed on entirely new and original solutions to the UFL and k-median problems, that were clearly not in existence before they developed them, this could be more challenging.

"Some researchers have written custom software that directly structures and solves location models [4, 6, 43, 44], relying on general optimization solvers. While custom software allows developers to specify their models, it comes with obvious shortcomings. It is challenging to develop new software, which requires specific technical skills—software design, tackling data input, model implementation and integration with solvers. "[42].

Dual fitting involves constructing a feasible dual solution that is close to the optimal dual solution of a relaxed version of the original primal problem, using a relaxation of the constraints. This feasible dual solution is then used as a certificate, to guide the construction of a solution to the original primal problem [40]. The methods solve a combinatorial problem which is complex to solve efficiently using the rule of thumb. They are focused on eliminating outliers and achieving the median, optimum distribution for all service centers of clients on a bipartite graph.

With DEA and TOPSIS the starting point is the choice of a set of potential sites (alternatives Chen, Li [5]), of a given number. The quest is then to sort the potential sites and rank them according to a set of criteria from best to worst. Muvingi, Peer [3] uses 16 district and 49 suburb locations in the city of Harare Zimbabwe. This method is clerical and involves a lot of data gathering which may limit its optimum efficiency to only small size problems. Gathering efficient criterion data for, for example a 100 alternative locations appear exhausting, clerical and time consuming [3].

The choice of using district locations in Muvingi, Peer [3], is similar to the instance creation process of the ESMVERE algorithms, Mvere [1, 2, 6] where one must choose what constitutes a potential facility location or a client location. Mvere [1, 2, 6] use 156 gas stations GPS coordinates, as potential facility locations, while the client location coordinates are represented by the centroids of 144 districts locations. The two sets of GPS coordinate data cover the whole island country of Mauritius [1, 2]. These are the only sets of data required which are easier to gather efficiently. The opening cost of a potential facility location in Mvere [1, 2, 6], which is a main required input for the calculation of the UFL problem data, was collected from the potential facility location owners. The gas station owners decided how much they would charge to place a ULAB collection facility on their property.

In addition to the neat final solution quality Mvere [2], the ESMVERE CPLEX UFL algorithm showed superior computational efficiency as problem scale grew. Comparative analyses with other leading methods, including the Iterated Local Search (ILS), Multi-Start Descending Algorithm with Screening (MDAS), and hybrid heuristics (HYB), revealed that the ESMVERE CPLEX UFL problem solver algorithm consistently required lower average CPU time across all examined problem instances [2]. This efficiency gain is particularly important in large-scale applications, where computational resources and time constraints play a critical role.

The Dual fitting and factor revealing LP technique was tested using randomly generated instances which vary in sizes from (20 facilities to 50 clients) to (150 facilities to 400 clients [37]. These instances were generated on a 10000 x 10000 grid. The Internet Topology generator software, GT-ITM and the OR library were also used to generate test instances [37]. The OR-Library is a library of test data sets for several operations research problems. Using this method on a real-world problem would mean at least the construction of a distance matrix of 20 facilities and 50 clients [37]. That would mean 20 times 50 which is equal to 1000 pairs of distances in a distance matrix. It would mean measuring 1000 distances in the real world between potential facility locations and clients before the problem can be solved efficiently with this technique. For 156 potential facility locations and 144 clients' locations in Mvere [1, 2], it would mean measuring 22464 distances accurately to solve for a 1.861 approximate solution, Jain, Mahdian [4] this would be a tedious data gathering process.

The algorithms in Mvere [1, 2], are exact algorithms. An exact algorithm is a computational method that solves an optimization problem to its optimal solution, without any approximation or error [8]. Exact algorithms can be computationally expensive, impractical and time consuming for large-scale problems because they aim to find the optimal solution [9]. However, the inclusion of the Euclidean distance formulae to the UFL and k-median solution in Mvere [1, 2, 6] has reduced computational complexity and made it usable even in large cases. "Sometimes, researchers develop specialized exact algorithms that exploit specific characteristics of the problem or its instances to achieve better performance than a general-purpose algorithm [13]". These specialized exact algorithms made it possible for Mvere [1, 2] to solve the NP-Hard, UFL and k-median problems efficiently in small instances.

The algorithms in Mvere [1, 2] explore the entire solution space, which is extremely efficient in small cases of 7 facilities and 125 clients [2]. However, for larger cases there is a need to purchase better versions of the software [14]. The limitation on size meant that the island of Mauritius was divided into 9 regions and the selection of the best potential facility locations was calculated in series using the same client locations of each region with different facility locations. This calculation would select the optimum facilities to open in each region. This gave a near optimum result in Mvere [1, 2] compared to the optimum solution where the calculation was performed at once using the better version of the software [14]. In most cases the same median facilities would be

opened because the core concept remains the same, that each client connects to a facility which is near it. The difference was observed, with the optimal solution which performs the calculation at once for the whole island, that some regional borderline clients would connect to closer facilities in another region. Thus, the optimum solution had a lower value which went further in lowering the costs of the solution [2].

The approximation factor revealing LP technique on the other hand encodes the problem of finding the worst possible instance of a UFL problem as a linear program [4]. This provides an LP family, one for each value of the number of cities or clients. Jain, Mahdian [4] explains that the best value for the approximation factor (γ) is then the supremum (or the least upper bound) of the optimal solutions to these worst-case LP's. Jain, Mahdian [4] were unable; however, to explicitly measure the supremum, they get a solution which is feasible, to the dual of each of the worst-case LP's. They conjecture that it is possible to determine an upper bound on the duals' objective function value this is also an upper bound on the optimal γ. In the first algorithm, this upper bound is 1.861 [4].

There are situations where exact algorithms cannot be used these are cases that require Approximation algorithms [9]. Approximation algorithms are used instead of exact algorithms when an exact solution is computationally infeasible, too slow, or when a near-optimal solution is sufficient [13]. When finding an optimal solution is intractable, approximation algorithms offer a trade-off between speed and accuracy, finding solutions within a certain approximation factor of the optimal solution [13]. The most common approach to overcome this difficulty, known as approximation algorithms, is by relaxing the requirement of finding optimal solutions [10]. Instead, one can only hope to efficiently find a good approximate solution, one that is not optimal, but is within a small factor away from optimal [11]. Many of these problems in optimization turn out to be computationally intractable, so a rational strategy is to settle for approximation algorithms, that is, algorithms that run in polynomial time and always have known near-optimal workable solutions [12].

2. Main Contents

2.1. Greedy Facility Location Algorithms Analyzed Using Dual Fitting with Factor-revealing LP

We analyze the first algorithm of Jain, Mahdian [4] and Mahdian, Spielman [37] which has an approximation factor of 1.861, to attempt to solve a real-world case of sitting ULAB collection facilities Mvere [1, 2].

Jain, Vazirani [7] proposed a 3 and 6 approximation algorithm for the metric UFL problem and the metric k-median problem, respectively. Their algorithm is an extension of the primary-dual scheme to deal with a primary-dual pair of LPs that are not a covering-packing pair for the UFL metric problem and the k-median metric problem.

To this day, the algorithm they suggest forms the key core basis for all of the primal dual methods. The basic idea behind the algorithm is that each city j continues to boost its dual variable, αj, until it is linked to an open facility, other primary and dual variables simply respond to this transition, trying to preserve feasibility or satisfy complementary conditions of slackness [7].

Jain and Mahdian [4] formalize the methodology used as the dual fitting technique, to examine the set cover problem, Mahdian, Spielman [37], also introduce the concept of using the approximation factor-revealing LP. Using this duo, Jain and Mahdian [4] analyze two greedy algorithms for the metric UFL problem with 1.861 and 1.61 approximation factors for both problems, respectively, and $O(mlogm)$ and $O(n^3)$ runtimes, respectively. When explaining the dual fitting method they imply that the basic algorithm is combinatorial, and is a simple greedy algorithm in the case of the set covering problem, assuming it to be a problem of minimization.

Jain and Mahdian [4] describe the combinatorial algorithm as a primal-dual-type algorithm that allows primal and dual updates iteratively, using the problem's linear programming relaxation and its dual-type algorithm. Although it is not a primal dual algorithm, since the computed dual solution is infeasible, they show that the primal integral solution found by the algorithm is fully paid for by the dual computed solution, meaning that the primal solution's objective function value is bounded by that of the dual solution [4]. The algorithm's main step is to divide the dual by a suitable factor, γ, and to prove that the shrunk dual is feasible, i.e. $fits$ into the instance provided.

On OPT, the shrunk dual is then a lower limit, and γ is the approximation guarantee of the algorithm. The aim is therefore to find the necessary minimum γ, which is to find the worst possible instance of the problem, one in which the dual solution needs to be shrunk as much as possible in order to make it feasible [4]. Jain and Mahdian [4] describe an approximation factor-revealing LP that encodes the problem of finding the worst possible instance as a linear program.

This provides an LP family, one for each value of the number of cities n_c. Jain and Mahdian [4] further explain that the best value for γ is then the supremum of the optimal solutions to these LP's. They were unable however, to explicitly measure the supremum, they get a solution which is feasible, to the dual of each of the LP's. They conjecture that it is possible to determine an upper bound on the duals' objective function value this is also an upper bound on the optimal γ. In the first algorithm, this upper bound is 1.861 and in the second algorithm, 1.61. Jain and Mahdian [4] numerically solves the factor-revealing LP with a large value of n_c to get a closely matched tight example.

The first algorithm of Jain and Mahdian [4], is similar to the greedy set cover algorithm, which iteratively selects the

most cost-effective choice at each stage. Cost effectiveness is defined as a measure of the ratio of the costs incurred to the number of new cities served. To evaluate this algorithm, they used LP-duality and gave an alternate definition that can be seen as a Jain, Vazirani [7] algorithm update.

Algorithm 1 enhances facility location efficiency by balancing opening and connection costs through a structured cost comparison and iterative refinement process. This approach leads to optimized facility placement and city assignment, reducing total operational costs.

The most basic explanation of Jain and Mahdian [4]′s first algorithm is, when a city is connected to an open facility, it withdraws everything it has contributed to the opening costs of other facilities, meaning that the dual pays for the primal solution in full.

Jain and Mahdian [4] summarise their first algorithm as:

Let U be an unconnected cities set.

All cities are unconnected at the beginning, i.e. $U := \mathcal{C}$, and all facilities are unopened.

$$\text{If } U \neq \emptyset$$

Of all the stars, find the most cost-effective one $(i, \mathcal{C}')$, open i if it's not already open, and connect all the cities in $\mathcal{C}'$ to i

$$f_i := 0,\ U := U \setminus \mathcal{C}'$$

A star consisting of one facility and several cities is defined by Jain and Mahdian [4]. The cost of a star is the sum of the facility's opening cost and the connection costs between the facility and all the cities in the star in their algorithm. This is formally clarified as the price of a star $(i,\mathcal{C}')$, where i is a facility, and $\mathcal{C}' \subseteq \mathcal{C}$ is a subset of cities, $f_i + \sum_{(j\in\mathcal{C}')} c_{ij}$. A facility can be selected again after opening in their algorithm, but the opening cost is counted only once and f_i is set to 0 after the first algorithm selects the facility. When connected to an open facility, each city j is deleted from $\mathcal{C}$ and is not taken into account again.

An order is carefully observed with the two algorithms for each facility i, they sort the cities in increasing order of their connection cost to i. Jain and Mahdian [4] determine that a facility and a set containing the first k cities in this order for some k will consist of the most cost-effective star. They describe α_j as the city j's contribution or its share of the overall total expenses. They explain how raising the dual variables in each iteration of the greedy algorithm will help find the most cost-effective star.

They restate Algorithm 1 based on the observation that, the most cost-effective star will be the first star $(i, \mathcal{C}')$ for which $\sum_{(j\in\mathcal{C}')} max(0, \alpha_j - c_{ij}) = f_i$. Dual variables of all unconnected cities are simultaneously increased.

Jain and Mahdian [4] Algorithm 1 restatement:

They add a notion of time, such that each occurrence is related to the time it occurred. The algorithm begins at the moment 0. Each city is initially set to be unconnected ($U := \mathcal{C}$), all facilities are unopened, and α_j is set to 0 for each j.

While $U \neq \emptyset$ increases the time and, simultaneously, $j \in U$ increases the α_j parameter at the same rate for each city, until one of the incidents below takes place (if two events occur at the same time, they are processed in an arbitrary order).

For some j of an unconnected city, and for some i of an open facility, $\alpha_j = c_{ij}$. Connect city j to facility i and delete j from U in this case.

We've got $\sum_{(j\in U)} max(0, \alpha_j - c_{ij}) = f_i$ for some unopened i facility. This ensures that the cities' total contribution is adequate to open the i facility. Open this facility in this case, and j with $\alpha_j \geq c_{ij}$ for each of the unconnected cities, connect j with i, and delete it from U.

Jain and Mahdian [4]give an analysis of algorithm 1 in which they explain that every city's contribution goes into opening one facility at MOST and connecting the city to an open facility. They determine that the total cost of the solution generated by their algorithm is equal to the amount of the $\sum_j \alpha_j$ contributions. Jain and Mahdian [4] explain that α is not a feasible dual solution since they remove a subset of cities in each iteration of the restatement of algorithm 1 and withdraw their input from all facilities.

At the end of a facility, therefore, i, $\sum_j max(\alpha_j - cij, 0)$, can be greater than f_i, thereby violating the corresponding dual constraints. They reiterate that the quest is to find a suitable γ for which α / γ is feasible, $\sum_j \alpha_j/\gamma$ will be the lower limit to the optimum, so at most γ will be the approximation factor for the algorithm [4]. They include a description that describes that if α / γ is a feasible dual, it is trivial if and only if each facility is γ-overtight at most [4].

$$\sum_j max(\alpha_j/\gamma - c_{ij,}, 0) \leq f_i \qquad (1)$$

They conjecture that the goal is to find such a γ, and thus the need is to only consider the cities j for which $\alpha_j \geq \gamma c_{ij}$. Jain and Mahdian [4] give a lemma which captures the metric property: $For\ every\ two\ cities\ j\ ,\ j'\ and\ facility\ i\ ,\ \alpha_j \leq \alpha_{j'} + c_{ij'} + c_{ij}$. This lemma is illustrated in the worked example diagrams City A Algorithm 1 and City B Algorithm 1. They state that when $\alpha_{j'} \geq \alpha_j$, the inequality obviously holds, as on the diagram City B Algorithm 1.

However as on the diagram City A Algorithm 1 assuming $\alpha_j > \alpha_{j'}$. If city j' is connected to facility i' as on the diagram City A Algorithm 1, city A is connected to facility X, fi=10, this facility i' which is fi=10 on the diagram is open at time $\alpha_{j'}$ which is at 5 on the diagram. The contribution α_j which is 8 on the diagram cannot be greater than $c_{i'j}$ which is the distance from client C to facility X, fi=10 on diagram, because in that case city j or city C could be connected to facility X, i' or facility fi= 10 at some time $t = 5 < \alpha_j = 8$.

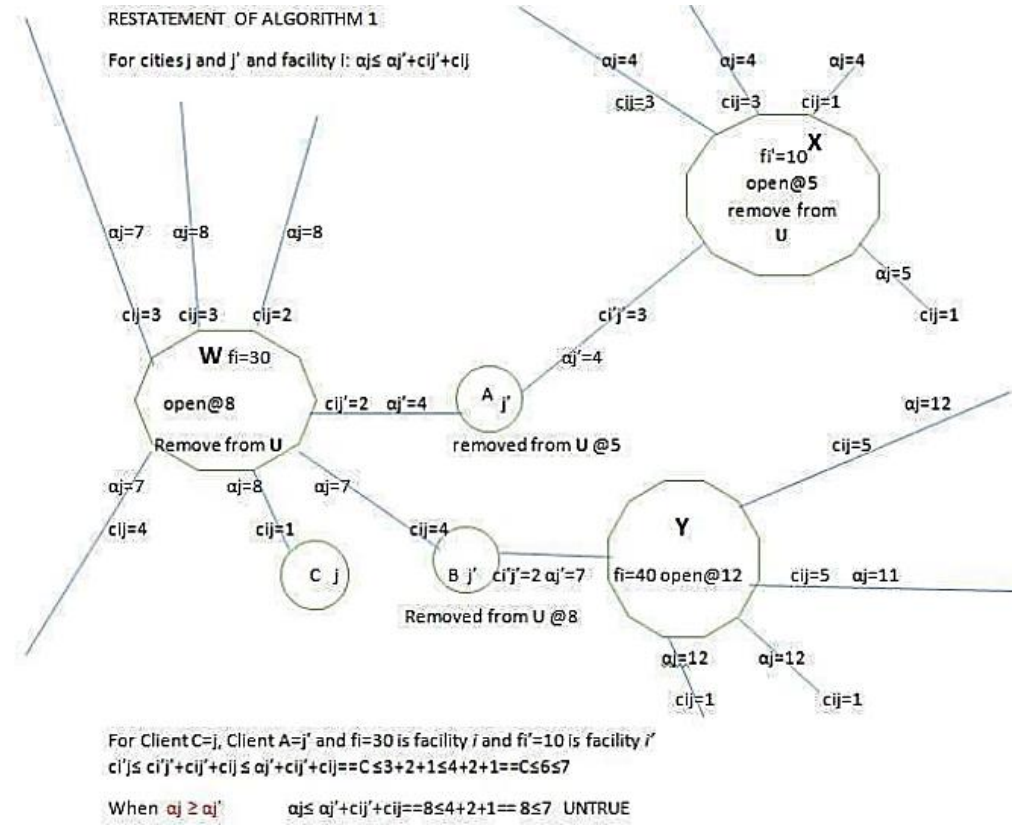


Figure 1. *City A Algorithm 1.*

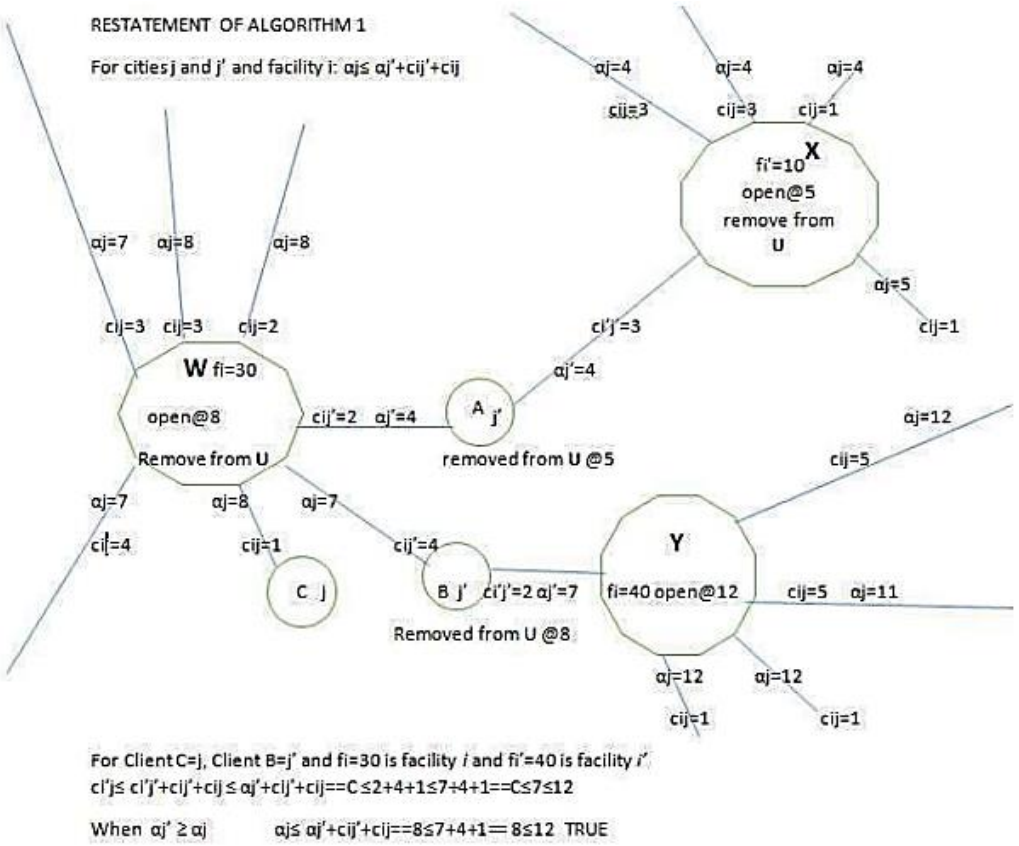


Figure 2. *City B Algorithm 1.*

The inequality calculations on Figure 1 serve as a validation step within the algorithm to determine whether a particular city to facility assignment is feasible or optimal based on cost constraints. The inequality compares the sum of connection costs and parameters for two facilities *i* and *i*' with respect to a city *j*. it checks if the combined cost of assigning city *j* to facility *i'* plus the facility opening cost *fi*' is less than or equal to the cost of assigning the same city to facility *i* plus *fi*. This helps decide if swtching the city's assignment to a different facility reduces the overall cost.

The result "UNTRUE" indicates that the assignment is not cost-effective to change under the given parameters. This step is crucial for the algorithm to iteratively refine facility openings and city assignments to minimize total costs in the location optimization problem.

Algoritm 1 improves facility location efficiency by systematically evaluating and optimizing the assignment of cities to facilities based on combined costs, including fixed facility opening costs and connection costs. It calculates the total cost for assigning cities *j* and *j*' to a facility *i* using the fomular $\alpha_j \leq \alpha_{j'} + c_{ij'} + c_{ij}$ where α_j represents contributions of each city towards opening a facility and c_{ij} represents the connection costs. By comparing these costs across different facilities and cities, the algorithm identifies the most cost-effective assignments, ensuring that facilities with lower total costs are prioritised and opened, while less efficient ones are removed from from the consideration set ***U*.**

The algorithm iteratively opens facilities with the lowest combined costs and removes those that do not improve the overall cost structure, as shown by the removal of facilities W, X and Y from the set *U*. This selective opening and closing process reduces unnecessary facility openings, minimizing fixed costs while maintaining service quality. The cost comparisons, such as $c_{i'j} + c_{ij'} + c_{ij} \leq \alpha_{j'} + c_{ij'} + c_{ij}$ ensure that city assignments are only changed when it leads to a net cost reduction, thus improving allocation efficiency.

Table 1 contains the set of all the cities on the map instance. The map has 16 cities on the first column to the left, On the second column each city is labeled after the facility it is connected to, its nearest facility, either facility W, facility Y or facility X. The city length is the distance between the city and its nearest facility. After a facility opens the city connects to the facility, this is the time it is removed from **U**, the set of connected facilities.

Table 1. *Restatement of algorithm 1. [4]*

U set of cities from Figure 1	city Label	city length	The time its removed from U
1	W1	1	8
2	W2	2 city A	5
3	W3	2	8
4	W4	3	8
5	W5	3	8
6	W6	4	8
7	W7 city B	4	8
8	Y1	1	12
9	Y2	1	12
10	Y3	2 city B	8
11	Y4	5	12
12	Y5	5	11
12	X1	1	5
13	X2	1	5
14	X3	3	5
15	X4	3	5
16	X5 city A	3	5

Table 2. *Facility W fi=30.*

	t=α_j	W1	W2	W3	W4	W5	W6	W7	fi contributions
$\sum_{dj}$	c_{ij}	1	2	2	3	3	4	4	=19
$\sum \alpha_j$	time	α_{j1}	α_{j2}	α_{j3}	α_{j4}	α_{j5}	α_{j6}	α_{j7}	
$\sum Contributions$	0	0	0	0	0	0	0	0	
0	1	1	1	1	1	1	1	1	0
1	2	1	1	1	1	1	1	1	1
4	3	1	1	1	1	1	1	1	3
9	4	1	1	1	1	1	1	1	5
15	5	1	0removed	1	1	1	1	1	6
21	6	1	0	1	1	1	1	1	6
27	7	1	0	1	1	1	1	1	6
open 30	@8	1	0	1	1	-	-	-	3
α_js	α_js	8	4	8	8	7	7	7	

Table 2 shows the contribution of each city in terms of αjs towards the opening of the Facility W. The αj increases as time in seconds, while each connected city contributes towards the opening cost off the facility W. Each city contributes towards covering its distance its distance (cm) first, bold font numbers, when it reaches the facility in distance (cm), then it starts contributing towards the opening of the facility (30 units). When all the contributions from all the cities connected to the facility are equivalent to the opening cost of the facility, the facility opens and connects every city at time 8.

City W2 is connected to another facility from the start of the time facility X, hence when the other facility X opens it connects to it. At time five its removed from U hence it stops contributing towards facility W opening

Table 3. *Facility X fi=10.*

	t=α_j	X1	X2	X3	X4	X5	fi contributions
$\sum_{dj}$	c_{ij}	1	1	3	3	3	=11
$\sum \alpha_j$	time	α_{j1}	α_{j2}	α_{j3}	α_{j4}	α_{j5}	
$\sum Contributions$	0	0	0	0	0	0	
0	1	1	1	1	1	1	0
2	2	1	1	1	1	1	2
4	3	1	1	1	1	1	2
9	4	1	1	1	1	1	5
Open 10	@5	1	-	-	-	-	1
α_js	α_js	5	4	4	4	4	

Table 3 shows the contribution of each city in terms of αjs towards the opening of the facility X. The αj increases as time in seconds, while each connected city contributes towards the opening cost of the Facility X. Each city

contributes towards covering its distance (cm) first, bold font, when it reaches the facility in distance (cm), then it starts contributing towards the opening of the facility (10 units). When all the contributions from all the cities connected to the facility are equivalent to the opening cost of the facility, the facility opens and connects every city at time 5.

Table 4. Facility Y fi=40.

	t=α_j	Y1	Y2	Y3	Y4	Y5	fi contributions
$\sum_{dj}$	c_{ij}	1	1	2	5	5	=14
$\sum \alpha_j$	time	α_{j1}	α_{j2}	α_{j3}	α_{j4}	α_{j5}	
$\sum Contributions$	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0
2	2	1	1	1	1	1	2
5	3	1	1	1	1	1	3
8	4	1	1	1	1	1	3
11	5	1	1	1	1	1	3
16	6	1	1	1	1	1	5
21	7	1	1	1	1	1	5
25	8	1	1	0removed	1	1	4
29	9	1	1	0	1	1	4
33	10	1	1	0	1	1	4
37	11	1	1	0	1	1	4
Open 40	@12	1	1	0	1	-	3
α_js	α_js	12	12	7	12	11	

Table 4 shows the contributions of each city in terms of αjs towards the opening of the Facility Y. The αj increases as time in seconds, while each connected city contributes towards the opening cost of the Facility Y. Each city contributes towards covering its distance (cm) first, bold font, when it reaches the facility in distance (cm), then it starts contributing towards the opening of the facility (40 units). When all the contributions from all the cities connected to the facility are equivalent to the opening cost of the facility, the facility opens and connects every city at time 12.

Jain and Mahdian [4] give another lemma which shows how the sum of the α_js of every facility must always be below the opening cost. They state this as, during the algorithm, the total contribution provided to a facility at any time is no more than its cost. $For\ every\ city\ j\ and\ facility\ i\ \sum_{l=j}^{k} max(\alpha_j - c_{il}) \leq fi$. The tables W, X, Y of the worked example diagram Algorithm 1 city B are an example of this. For example Facility W, $\sum_{j=W1}^{W7} max(49-19) \leq 40$, For Facility X, $\sum_{j=X1}^{X5} max(21-11) \leq 10$ and For Facility Y, $\sum_{j=Y1}^{Y5} max(54-14) \leq 40$.

In this lemma Jain and Mahdian [4] give an example of a $\alpha_k \leq \alpha_j$ for which the α_k algorithm stops increasing as soon as the k city has a tight edge to a facility which is fully paid for. This means that when a facility opens the algorithm stops increasing the α_js thus there is with many facilities at least one α_k which does not grow to the size of the α_j. Facility X is an example of this, with city X1 being the α_j @5 and cities X2, X3, X4, X5 being the α_ks which stop growing at 4.

Jain and Mahdian [4] state that the goal is to find the minimum γ for which $\sum_{j=1}^{k} (\alpha_j/\gamma - c_{ij}) \leq f_i$, hence the maximum of the ratio $\frac{\sum_{j=1}^{k} \alpha_j}{f+\sum_{j=1}^{k} d_j}$ is sought.

They go on to state that zk the maximum value of the objective function of the following program (2) is equal to the Linear Program (7) which they call the factor revealing LP below.

$$z_k = maximize \frac{\sum_{j=1}^{k} \alpha_j}{f+\sum_{j=1}^{k} d_j} \quad (2)$$

Subject to the constraints:

$$\alpha_j \leq \alpha_j + 1 \; \forall j \in \{1, \ldots\ldots\ldots, k-1\} \quad (3)$$

To note this is explained by Facility X where there is a k which can be the city larger than all the other cities explained by city XI being the k and cities X2, X3, X4, X5 being the ascending α_js 1,....., k-1.

$$\alpha_j \leq \alpha_l + d_j + d_l \; \forall j, l \in \{1, \ldots\ldots\ldots, k\} \quad (4)$$

$$\sum_{l=j}^{k} max(\alpha_j - d_l, 0) \leq f \; \forall j \in \{1, \ldots\ldots\ldots, k\} \quad (5)$$

$$\alpha_j, d_j, f \geq 0 \; \forall j \in \{1, \ldots\ldots\ldots, k\} \quad (6)$$

The Factor Revealing LP

$$z_k = maximize \sum_{j=1}^{k} \alpha_j \quad (7)$$

Subject to the constraints:

$$f + \sum_{j=1}^{k} d_j \leq 1 \quad (8)$$

$$\alpha_j \leq \alpha_j + 1 \; \forall j \in \{1, \ldots\ldots\ldots, k-1\} \quad (9)$$

$$\alpha_j \leq \alpha_l + d_j + d_l \; \forall j, l \in \{1, \ldots\ldots\ldots, k\} \quad (10)$$

$$x_{jl} \geq \alpha_j - d_l \; \forall j \in \{1, \ldots\ldots\ldots, k\} \quad (11)$$

$$\sum_{l=j}^{k} x_{jl} \leq f \; \forall j \in \{1, \ldots\ldots\ldots, k\} \quad (12)$$

$$\alpha_j, d_j, f \geq 0 \; \forall j \in \{1, \ldots\ldots\ldots, k\} \quad (13)$$

They state that (Primal):

1) zk is the optimal solution to the approximation factor-revealing LP.
2) zk is a lower bound on the value of γ.
3) zk is the minimum γ for which $\sum_{j=1}^{k} (\alpha_j/\gamma - c_{ij}) \leq f_i$.
4) zk is the maximum value of the objective function of the ratio $\frac{\sum_{j=1}^{k} \alpha_j}{f+\sum_{j=1}^{k} d_j}$.

At optimal, primal and dual solutions are equal thus:

They state that (Dual):

1) A solution to the dual of the approximation actor-revealing LP is needed to prove an upper bound on γ.
2) An upper bound on the objective function values of the duals of the factor-revealing LP is an upper bound on γ.
3) It is possible to solve the dual of the factor-revealing LP to prove a close to optimal upper bound on the value of zk.

The suprenum being the least upper bound, The suprenum of (zk) the optimal solution to the approximatimation factor-revealing LP $\sup_{k\geq 1}\{zk\}$ is the best value of γ. Therefore $\gamma =: \sup_{k\geq 1}\{zk\}$.

Jain and Mahdian [4] give a lemma for every $k \geq 1$, $zk \leq 1.861$.

They multiply the inequalities of the factor-revealing LP to give an inequality of the form:

$$\sum_{j=1}^{k} \alpha_j/\gamma - \sum_{j=1}^{k} c_{ij} \leq f \quad (14)$$

$$\sum_{j=1}^{k} \alpha_j - \gamma \sum_{j=1}^{k} c_{ij} \leq \gamma f \quad (15)$$

$$\sum_{j=1}^{k} \alpha_j - 1.861 \sum_{j=1}^{k} d_j \leq 1.861 f \quad (16)$$

They suggest that finding the right multipliers for the above inequalities is equivalent to obtaining *a feasible solution to the dual* of the factor-revealing LP. They add three constants $p1$, $p2$ and θ_j to prove the bound on γ for algorithm 1.

$$\sum_{i=j}^{k} \theta_j \leq 1.861 \leq \gamma \quad (17)$$

$$\sum_{j=1}^{k} \alpha_j == \sum_{j=1}^{p1k} \sum_{i=j}^{p2k} \theta_j(\alpha_j - d_i) \quad (18)$$

$$-1.861 \sum_{j=1}^{k} d_j == \sum_{j=p1k+1}^{k} \sum_{i=j}^{k} \theta_j(\alpha_j - d_i) \quad (19)$$

$$\left(\sum_{i=j}^{k} \theta_j\right) f == 1.861 f \quad (20)$$

$$coeff_A[\alpha_j] = \begin{cases} (p2k - j + 1)\theta_j & \text{if j} \leq \text{p1k} \\ (k - j + 1)\theta_j & \text{if j} > \text{p1k} \end{cases} \quad (21)$$

$$coeff_A[-d_j] = \begin{cases} \sum_{i=1}^{j} \theta_i & \text{if j} \leq \text{p2k} \\ \sum_{i=p1k+1}^{j} \theta_i & \text{if j} > \text{p2k} \end{cases} \quad (22)$$

From equation (18), (19) and (20) the inequalities (16) and (28) are compared. Inequality (28) extends inequality (16) by adding the three constants $p1$, $p2$ and θ_j. They denote that the left hand side of inequality (28) by A and furthermore propose that the sum of the coefficients of α_j's in A is at least k. Thus θ_j's are needed to satisfy the inequality:

$$\sum_{j=1}^{p1k} (p2k - j + 1)\theta_j + \sum_{j=p1k+1}^{k} (k - j + 1)\theta_j \geq k \quad (23)$$

They propose inequality (23) which guarantees that the average coefficient of α_j's in A is at least one. An example of this is the best way to describe the concept: If k is 20, the number of cities being 20.

$$\sum_{j=1}^{20} \alpha_j \quad (24)$$

The total sum of α_j's could be 23 therefore

$$\sum_{j=1}^{20} \alpha_j = 23 \quad (25)$$

Dividing 23 the total sum of α_j's by their number 20 then the average coefficient of α_j's is 1.5.

Jain and Mahdian [4] propose that for: $lj \geq j$

$$\sum_{i=j}^{lj} (\alpha_j - d_i) \leq f \quad (26)$$

$$lj = \begin{cases} p2k & \text{ifj} \leq \text{p1k} \\ k & \text{ifj} > \text{p1k} \end{cases} \quad (27)$$

$$\sum_{j=1}^{p1k} \sum_{i=j}^{p2k} \theta_j(\alpha_j - d_i) + \sum_{j=p1k+1}^{k} \sum_{i=j}^{k} \theta_j(\alpha_j - d_i) \leq \left(\sum_{i=j}^{k} \theta_j\right) f \quad (28)$$

They prove that:

$$1.861 == \gamma \quad (29)$$

in the worst case.

It should be possible, from the look of things by simple analysis, to use an instance created, for the reverse supply chain network design of ULABs in Mauritius, Mvere [1, 2] to achieve a solution guaranteed by γ. Since γ guarantees that the greedy algorithm Jain and Mahdian [4] solves the UFL problem, (1.861) close to an optimal solution. However, this is not the purpose of this algorithm, and achieving its use in the real world would be extremly difficult and complex. An attempt at this would mean:

1) The instance created for siting ULABs in Mauritius Mvere [1, 2] has to be created.
2) To create this instance there is a need to develop a Distance matrix of 156 facilities and 144 clients
3) This *n* x *m* would be 22464 distances that have to be measured for each pair of distances between a client and a facility
4) The instance created for siting ULABs in Mauritius [1, 2] are encoded as a family of linear programs.
5) The same instance is recreated into a family of instances with different number of n_c or cities and clients.
6) Solve these family of LPs created, to get their optimal solutions.
7) Obtain the suprenum of these LPs and compare it to the already known (γ).
8) In each case, with different number of cities n_c, we have a usable optimal solution guranteed by (γ).
9) This is because (γ) is the supremum (or the least upper bound) of the optimal solutions to these worst case LP's.

This has so far proved to be a misconception, an asumption that since the algorithm is well formulated in theory, it can be easily attempted to solve a real world case. Many theoratical science approximation algorithms such as Jain and Mahdian [4] have the most complex transfer from science research to real world application. Approximation algorithms in general can be adapted to specific needs and constraints, they can be used to find multiple solutions, allowing for greater flexibility in decision-making, especially when dealing with real-world problems where constraints and objectives are often "approximately" formulated [20]. However, not all approximation algorithms are usable practically in real world problems. Some involve complex data structures, or sophisticated algorithmic techniques, leading to difficult implementation issues or improved running time performance (over exact algorithms) only on impractically large inputs [21].

In consultation with the Author of [17], who is an emminent expert and scolar in the field of theoratical computer science, they revealed that they had not, at the time of writing, implemented their approximation algorithm of [17] in real world practice, and they were not very familiar with the application side of the two problems, the UFL and the k-median problem. "If you want to implement algorithms for the two problems, you may try to start with the primal-dual algorithms [4] since they do not require you to solve a linear programming" [17].

The primal-dual approach in optimization does not always require solving a linear program directly to find a solution [18]. It's a powerful technique that constructs feasible primal and dual solutions iteratively, often leading to algorithms where analysis is derived from the primal-dual interaction [18]. It focuses on iteratively improving feasible solutions and using the duality relationship to gain insights into the problem, reducing the duality gap, and eventually finding the optimal solution [19]. Primal-dual algorithms, avoid explicitly solving a (LP) problem [19]. Instead, they construct approximate solutions to the problem and feasible solutions to the dual of an LP relaxation simultaneously [22]. This approach leverages the duality relationship between the primal and dual problems, allowing for a more efficient and insightful solution process [22].

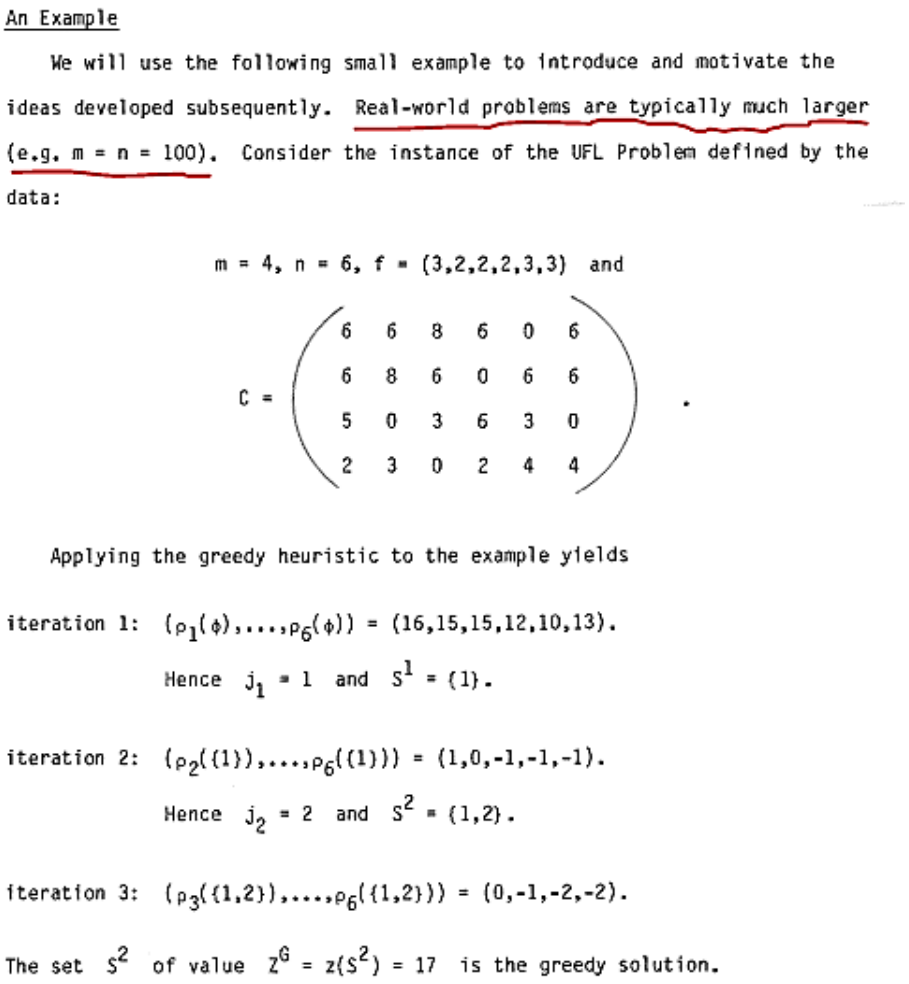
2.2. The Near Optimal Siting of Hazardous Waste (Used Lead Acid Battery (ULAB)) Collection Facilities in the Republic of Mauritius Using the [2] ESMVERE CPLEX UFL Problem Solver Algorithm, [1] ESMVERE CPLEX k-median Problem Solver Algorithm [6]

Mvere [1, 2] explores reverse supply chain network design problems in the automotive industry of the republic of Mauritius, specifically focusing on the challenge of recycling ULABs. The authors examine logistical issues related to the UFL and the k-median location problems [6]. Thus Mvere [1, 2] deals with the important problem of improving the safety and effectiveness of the collection, processing, and disposal of one of the most hazardous parts of the vehicles, the ULAB

[6].

Mvere [1, 2] presents a mathematical framework that consists of two facility location problems the UFL, and the k-median problems as alternative approaches to finding the best location sites for the collection facilities of the ULAB batteries Mvere [1, 2, 6] present an innovative approach to the siting of hazardous waste collection facilities in Mauritius using the novel ESMVERE CPLEX UFL problem solver algorithm, and the ESMVERE CPLEX k-median problem solver algorithms [6]. Both papers address a significant gap in the literature by applying UFL and k-median problem-solving techniques in a real-world setting, using the actual data from the Mauritius statistical office [6]. Mvere [1, 2] ,illustrated a step-by-step instance creation process which familiarizes the reader with how to solve a realistic real-world problem. They clearly demonstrated the algorithm solution to the real-world data from Mauritius using the ArcGIS mapping [6]. The solutions are also visible when they are input on google earth [6].

The primary contribution of Mvere [1, 2] is the removal of the distance matrix as input in the calculation for the UFL problem and the k-median problem, replacing the distance matrix with the GPS input and Euclidean distance formulae [6]. This simplified the data input and reduced computational complexity for the algorithm making it easy, usable, and convenient as a solution [6].



An Example

We will use the following small example to introduce and motivate the ideas developed subsequently. Real-world problems are typically much larger (e.g. m = n = 100). Consider the instance of the UFL Problem defined by the data:

$m = 4,\ n = 6,\ f = (3,2,2,2,3,3)$ and

$$C = \begin{pmatrix} 6 & 6 & 8 & 6 & 0 & 6 \\ 6 & 8 & 6 & 0 & 6 & 6 \\ 5 & 0 & 3 & 6 & 3 & 0 \\ 2 & 3 & 0 & 2 & 4 & 4 \end{pmatrix}.$$

Applying the greedy heuristic to the example yields

iteration 1: $(\rho_1(\phi),\ldots,\rho_6(\phi)) = (16,15,15,12,10,13)$.

Hence $j_1 = 1$ and $S^1 = \{1\}$.

iteration 2: $(\rho_2(\{1\}),\ldots,\rho_6(\{1\})) = (1,0,-1,-1,-1)$.

Hence $j_2 = 2$ and $S^2 = \{1,2\}$.

iteration 3: $(\rho_3(\{1,2\}),\ldots,\rho_6(\{1,2\})) = (0,-1,-2,-2)$.

The set S^2 of value $Z^G = z(S^2) = 17$ is the greedy solution.

Figure 3. *UFL Matrix solution. Source. Cornuejols, Nemhauser [29]*

Since their introduction over 60 years ago by Balinsky [24] and Hakimi [25] both the UFL and k-median problems have been solved using the distance matrix. Jain, Mahdian [4], Cacioppi, Watson [26], Mistry [27], Mahdian, Spielman [37], are examples of the CPLEX Opl software algorithms that solve the UFL problem using the input of the distance matrix Srinivasan [28] is a p-median example of how the problems were solved [6]. Cornuejols, Nemhauser [29] is an example of how the UFL problem was solved, as seen on Figure 3.

Thus Mvere [1, 2] introduce a solution that does not need the input of a distance matrix but uses GPS based Euclidean distance calculations, while, to the best of our knowledge, all known prior solutions used the input of a distance matrix in the calculation of the UFL problem and the k-median problem [6].

It would be incredibly difficult to use this solution in a real-world scenario, which could explain why there are not so many real-world examples [6]. All known prior innovations to the UFL and k-median problem solutions were centered on improvements and innovations around the matrix [28, 29]. Thus Mvere [1, 2] proposes a solution which is a novel breakthrough spanning over 60 years of work in this area [24, 25]. Cacioppi, Watson [26], use the distance matrix below (30), in a small nine warehouse example [6]. Even in their short example, to ensure that the distance of each point to the other remaining eight points is combinatorically considered, is a big combinatorial task [1, 2, 6, 26]. This becomes even more complex when you must gather the data between all possible combinations of distances, for example if there are 156 facility locations and 144 client locations m x n = 22464 location distances to measure and place in a distance matrix. This is somewhat easy when it is randomly generated by software, however, in real world practice this is extremely complex. Below is a distance matrix of only 9 x 9 locations [1, 2, 6].

Cacioppi, Watson [26] Distance Matrix (30)

$$Distances\ [[0,720,790,297,283,296,461,796,996]$$
$$[720,0,884,555,722,461,685,245,1099]$$
$$[790,884,0,976,614,667,371,645,219]$$
$$[297,555,976,0,531,359,602,715,1217]$$
$$[283,722,614,531,0,263,286,629,721]$$
$$[296,461,667,359,263,0,288,479,907]$$
$$[461,685,371,602,286,288,0,448,589]$$
$$[769,245,645,715,629,479,448,0,867]$$
$$[996,1099,219,1217,721,907,589,867,0]];$$

This would be impractical in many cases in the real world where there might be hundreds or thousands of location points, and the distances that make each pair of distances [1, 2, 6]. Thus, a new model is developed that does not need the input of the distance matrix but uses the formulae for the Euclidean distance between two points and the input of actual GPS coordinate location points to solve the UFL and the k-median problem in Mvere [1, 2, 6]. This solution is user friendly and convenient.

The formula for the Euclidean distance between two points is:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad (31)$$

The UFL, k-median CPLEX Opl Euclidean distance formulae developed for this study by Mvere [1, 2, 6]:

$$function\ getDistance\ (ulabcity1, ulabfacilitycity1)$$

{

$$return\ opl.sqrt(opl.pow(ulabcity1.x - ulabfacility1.x, 2) + opl.pow(ulabcity1.y - ulabfacility1.y, 2))$$

}

The new method solves the UFL and the k-median problem by making a choice between facility locations and client locations, without considering the distance between all the points as a distance matrix [6]. The newly developed method only requires the GPS coordinate location points [1, 2, 6]. This means that it opens the best out of n potential facility locations and assigns only m clients to these open facilities per calculation [1, 2, 6].

A combination of the CPLEX solutions to the Travelling Salesman Problem (TSP) by Caceres [30] and a CPLEX solution to the p-median algorithm by Cacioppi, Watson [26], was used to develop a new and unique software invention which solves the UFL and k-median problem by Mvere [1 ,2 ,6]. The works of Cacioppi, Watson [26] and Caceres [30] were thoroughly analyzed and reviewed.

Combining these two solutions made it possible to develop 3 entirely new CPLEX models which solve the UFL problem, k-UFL problem and k-median problem [6]. The sub tour elimination decision variables of the TSP problem were removed [30]. Arrays of the facility locations and opening costs were added in Mvere [1, 2, 6], these enabled the costs of the edges to be computed from each facility to each client. The changes have developed entirely new software solutions in Mvere [6] after careful consideration and study.

Definition of parameters

a) i is used to denote a (ulab collection facility).

b) j is used to denote a (ulab city) it is a residential area with a population of clients that want to dispose ulabs in Mauritius.

c) x_{ij} is a decision variable which denotes whether (ulab collection facility) i is connected to (ulab city) j or not. x_{ij} is a binary variable, which becomes 1 if (ulab collection facility) i is connected to (ulab city) j or 0 if (ulab collection facility) i is not connected. Being connected means the (ulab city) can dispose its ulabs at the open (ulab collection facility).

d) y_i is a decision variable which denotes whether (ulab collection facility) i is open or not. y_i is a binary variable, which becomes 1 if (ulab collection facility) i is open to receive disposed ulabs from (ulab cities), or 0 if (ulab collection facility) i is not open to receive ulabs.

e) c_{ij} denotes the cost of the distance from (ulab city) j to (ulab collection facility) i. The distance determines the cost which has to be minimized. The distance is only a significant cost if the (ulab collection facility) i is open and connected to (ulab city) j, that is if x_{ij} is 1.

f) $\mathcal{F}$ is the larger set of (ulab collection facilities). The goal is to open the best set of (ulab collection facilities), which serve all the (ulab cities) at a minimum cost. The best set of (ulab collection facilities) is chosen and opened from the larger set of (ulab collection facilities).

g) $\mathcal{C}$ is the larger set of (ulab cities). All the (ulab cities) have to be served and Connected to a (ulab collection facility). All the (ulab cities) have to be connected to an open (ulab collection facilities), Unlike and direct op-

posite of (ulab collection facilities, where there is a choice to select the optimum located, open them, and let the rest remain closed.

h) $\mathcal{F}'$ is the best set of (ulab collection facilities) opened. This is when the best set of (ulab collection facilities) is chosen and opened from the larger set of potential (ulab collection facilities).

i) f_i the costs of opening (ulab collection facilities) i. The facility opening cost is one of the variables that determines the final value of the objective function. The objective is to minimize the sum of the (ulab collection facilities) opening costs.

The metric UFL problem is formulated as an integer program to satisfy the constraints:

$$Minimize \sum_{i\in\mathcal{F}} f_i y_i + \sum_{(i\in\mathcal{F}, j\in\mathcal{C})} c_{ij} x_{ij}. \quad (32)$$

Subject to the constraints:

$$\sum_{(i\in\mathcal{F})} x_{ij} = 1 \; \forall j \in \mathcal{C} \quad (33)$$

$$x_{ij} \leq y_i \; \forall i \in \mathcal{F}, j \in \mathcal{C} \quad (34)$$

$$x_{ij} \in 0{,}1 \; \forall i \in \mathcal{F}, j \in \mathcal{C} \quad (35)$$

$$y_i \in 0{,}1 \; \forall i \in \mathcal{F} \quad (36)$$

The mathematical formulation of the UFL problem is then converted to the CPLEX Opl language. The model formulation is translated into the Opl language using the IBM ILOG CPLEX Optimization Studio IDE 20.1.0. Exactly as the TSP model by Caceres [30] described in the literature the CPLEX Optimization Engine requires a model of the form:

1. Size or Range
2. Input Data Sources
3. Decision Variables
4. Decision Expressions
5. Objective Function
6. Constraints

1) Size or Range

The Opl language is used to declare two integer variables n is for the (ulab collection facilities) and m is for the (ulab cities). The Opl language is used to create two range variables $facilities$ and $cities$ which contain all the (ulab collection facilities) and (ulab cities) on the plane.

$int\ n = \ldots;$ // is the number of (ulab collection facilities)

$int\ m = \ldots;$ // is the number of (ulab cities)

$range\ ulabfacilities = 1..n;$ //the range of all the facilities from 1 up to *n*.

$range\ ulabcities = 1..m;$ //the range of all the cities from 1 up to *m*.

$int\ fi[ulabfacilities] = \ldots;$ //this is an array of the opening costs of all (ulab collection facilities)

In the CPLEX data file //the opening costs of all (ulab collection facilities) are given as a matrix:

$fi = [30, 20, 50, 20, 40, 60, 10];$

2) Input Data Sources

The tuple $Location$ is created for this model, this contains the float variables x and y. When it is called, this method divides the plane into x and y coordinates. It gives the location of every (ulab collection facility) and (ulab city). It also gives the starting and ending point of every edge, in terms of its x and y coordinates.

$tuple\ Location\ \{float\ x; float\ y;\}$

The tuple $edge$ is created, this contains the integer variables i and j. When it is called, this method connects all the points on the plane into edges, and every edge has beginning point (ulab city) j and ending point (ulab collection facility) i. Thus, the tuple $edge$ is called to create the edges.

$tuple\ edge\ \{int\ i; int\ j;\}$

A set of edges named $edges$ is created, which defines the character of what an edge is, and defines its boundaries, while placing all the edges on the plane into a set. The edge is described as less than the variable i and greater than the variable j, i is found in the $rangeulabfacilities = 1\ldots n$; and j is within the $rangeulabcities = 1\ldots m;$.

$setof\ (edge)\ edges = \{< i, >$
$j \mid i\ in\ ulabfacilities, j\ in\ ulabcities\ \};$

A float variable c is created, this is an array variable. c is an array of all the edges from the variable $edges$ which contains the set of all the edges. Square braces [] are used to denote an array and in this case are used to describe the variable c as an array which contains the set of all edges $edges$.

$float\ c[edges];$

The tuple $Location$ is called to create an array $ulabcityLocation$ which contains the range $ulabcities$. This array places x and y coordinates in every city. The array stores the x and y coordinates of every city.

$Location\ ulabcityLocation\ [ulabcities];$

The tuple $Location$ is called to create an array $ulabfacilityLocation$ which contains the range $ulabfacilities$. This array places x and y coordinates on every facility. The array stores the x and y coordinates of every facility.

$Location\ ulabfacilityLocation\ [ulabfacilities];$

The generated data is preprocessed before the problem is computed. To preprocess the data, a code block $execute\{\}$ is created. In the code block, a $functiongetDistance()$ is declared, this function calculates and returns the Euclidean 2-dimensional distance between a facility and a city. It returns this value based on their x and y coordinates. The formula for the Euclidean distance between two points is the same as on formular (31)

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

calculated by subtracting the x coordinates of the two points and the result squared is added to the subtraction of the two y coordinates of the two points. The difference of the subtrac-

tion of the two points were the x coordinates are subtracting each other and the y coordinates are subtracting each other is squared and then it is added. The square root of this difference is the length of the distance. This general common formula is translated into the Opl Language in the function $getDistance()$ illustrated in the code below.

In the code block $execute\{\}$, a for loop which randomly generates the x and y coordinates of every ulabcity are created as well as a for loop which randomly generates the x and y coordinates of every ulabfacility. Finally in the same code block $execute\{\}$, a for loop with a variable e is constructed. e represents every edge in the set of $edges$. The loop goes to the array c which is an array which stores all the edges from the variable $edges$ which contains the set of all the edges, it then uses the $functiongetDistance()$ to calculate the size of every edge in terms of the x and y coordinates of its starting point the ulabcity$j$ and its ending point the ulabfacility i.

$execute\{$
$function\ getDistance\ (ulabcity1, ulabfacilitycity1)\ \{$
$return\ opl.sqrt(opl.pow(ulabcity1.x - ulabfacility1.x, 2) + opl.pow(ulabcity1.y - ulabfacility1.y, 2))\ \}$
$for(var\ i\ in ulabfacilities)\ \{$
$ulabfacilityLocation[i].x;$
$ulabfacilityLocation[i].y;$
$\}$
$for(var\ j\ in ulabcities)\ \{$
$ulabcityLocation[j].x;$
$ulabcityLocation[j].y;$
$\}$
$for(var\ e\ in\ edges)\{$
$c[e] = getDistance(facilityLocation[e.i], cityLocation[e.j])$
$\}$

The code block $execute\{\}$ above, which is written in the Opl Language, contains one function and three for loops. The function $getDistance()$ returns the Euclidean 2-dimensional distance between two points.

The first for loop in the code block $execute\{\}$ is created by declaring a variable i. The variable i loops through every (ulab collection facility), in the range $ulabfacilities$. It places x and y coordinates. The code $ulabfacilityLocation[i].x;$ searches for an x coordinate from the array $ulabfacilityLocation$ in the data file.dat it loops through all the (ulab collection facilities) in the array. The code $facilityLocation[i].y;$ searches for an y coordinate from the array $ulabfacilityLocation$ looping through every (ulab collection facility) in the array.

The second for loop in the code block $execute\{\}$ is created by declaring a variable j. The variable j loops through every (ulab city), in the range $ulabcities$. It places x and y coordinates. The code $ulabcityLocation[j].x;$ searches for an x coordinate from the array $ulabcityLocation$ in the data file.dat it loops through all the (ulabcities) in the array. The code $ulabcityLocation[j].y;$ searches for an y coordinate from the array $ulabcityLocation$ looping through every (ulab city) in the array.

The third for loop in the code block $execute\{\}$ is created by declaring a variable e. The variable e loops through every edge in the set of edges. The function $getDistance()$ then calculates the distance between the x and y coordinates of the i and the x and y coordinates of the j of each edge. Each edge has beginning point j and ending point i. The function $getDistance()$ returns this distance for each edge in the array of edges $c[e]$. This just gives the length of every edge on the plane.

3) Decision Variables

There are two main decision variables in the UFL problem x_{ij}, which is declared in CPLEX Opl in the model as a Boolean decision variable

$dvar\ boolean\ x[edge];$

, then there is y_i which is also declared in the model as a Boolean decision variable

$dvar\ boolean\ y[facilities];$

The two variables stand for the solution set of facilities and edges. The variable x is multiplied to an element of an array []. The array that the variable x is multiplied to, is an array of the tuple $edge$. The elements of the tuple $edge$ are the contents of the array $[edge]$. Thus, the tuple $edge$ contains integer variables i's and j's. Therefore x connects i's and j's in an array to create the edges.

4) Decision Expressions

Decision variables and decision expressions are used to create the actual Opl model. Caceres [30] translate the mathematical formulation of the model decision expression into the Opl language.

$$minimize \sum_{i\in\mathcal{F}} f_i y_i + \sum_{(i\in\mathcal{F}, j\in\mathcal{C})} c_{ij} x_{ij}. \quad (37)$$

The two decision expressions have been placed above and below, to compare and juxtapose the translation so that other models can also be analyzed and translated into the Opl language model.

$dexpr\ float\ OpeningCost = sum(i\ in\ ulabfacilities)\ fi[i] * y[i];$

$dexpr\ float\ TotalDistance = sum(e\ in\ edges)\ c[e] * x[e];$

5) Objective Function

The objective function is that of the UFL problem in that it is to minimize the total sum of opening costs, and the total distance traveled from (ulabcity) to (ulabfacility) location.

$Minimize\ OpeningCost + TotalDistance;$

6) Constraints

The constraints are translated from the mathematical formulation of the UFL problem into Opl as a code block $subjectto\{\}$. In the constraint code block $subjectto\{\}$, the first constraint is placed as a for loop.

$subject\ to\ \{$

$forall(j\ in\ ulabcities: j >= 1)$
$sum\ (i\ in\ ulabfacilities)\ x[< i,j >] == 1;$
$i\ in\ ulabfacilities: i >= 1, j\ in\ ulabcities: j >= 1$
$sum(i\ in\ ulabfacilities)\ x[< i,j >] == 1;\ x[< i,j >] <= y[i];$
}

7) The Data

The CPLEX data file contains the data values which are entered manually. The values of n the number of (ulabfacilities), m the number of ulabcities. It also contains the values of the two arrays, $ulabcityLocation$ and $ulabfacilityLocation$ which host the x and y coordinates of ulabcity locations and ulab collection facility locations. It also contains the values of the array fi in which is entered the opening cost values of the ulab collection facilities.

8) The Input File

The location input is within an array of x and y coordinates *ulabcitylocation* and *ulabfacilityLocation:* [<x y> <x y>] can be replaced by GPS Coordinates for example: [< -20.172544 57.4883317 > <-20.165595 57.492625>]. The input coordinates are easier and more convenient to input for the calculation of the UFL problem compared with a distance matrix of the same coordinate points. This makes the algorithm solution user friendly, and usable in a real-world case.

Li [36] shows an example problem where Walmart tasks the k-median problem, Mvere [1], an American retail giant, to select the best locations to site their retail stores. The aim was to site their 50 stores in Washington state so that the average travelling time for people to their nearest Walmart shop is minimized [36]. This is an example of a real-world case where the average travelling time, distance, is minimized. However, the decision is only concentrated on financial terms, monetized terms [1, 2]. There are other criteria such as competition with other retail stores within the vicinity, the presence of other buildings, residential areas, road density, population density and demographics and other factors that are not put into consideration with location models like Mvere [1, 2] and Jain, Mahdian [4] which may affect the performance of the Walmart store when it is located on any optimally selected facility location by the UFL or k-median problem.

The DEA and TOPSIS method may be used in Location analysis to make an informed decision in such a case as suggested by Li [36] which includes many other criteria. Each of the 50 locations is turned into a DMU and listed as an alternative, its performance as a potential location is evaluated according to the multiple criteria. This involves a lot of data gathering. It also involves a lot of subjective choices and preferences of the decision makers and the community.

```
/*********************************************
 * OPL 20.1.0.0 Data
 * Author: Emmanuel Siyawo Mvere
 * Creation Date: OCT 28, 2022 at 6:58: PM
 *********************************************/

n=7;

m=21;

ulabcityLocation=[<0.1 6.4> <0.2 0.9> <0.75 2.1> <1 6.9> <2.44 5.6> <3.1 3.4>
 <3.45 0.9> <4.25 5.6> <5.15 2.2> <5.6 0.95> <6.1 6.95> <7.55 6.9> <7.9 4.4>
 <9.2 1> <9.35 3.5> <9.65 4.4><10.55  5.7><12.3 2.25> <12.4 4.45> <12.55 7>
 <12.65 1>];

ulabfacilityLocation=[<0.7 3.8> <2.2 1.2> <4.3 7.3> <5.8 4.75> <7.3 1.3> <10.2
 2.5><10.8 7.3> ];

fi=[30, 20, 50, 20, 40, 60, 10];
```

Figure 4. *Input file. Source Mvere [1, 2, 6]*

2.3. Hierarchical Groups DEA Super-efficiency and Group TOPSIS Technique: Application on Mobile Money Agents' Locations

Multi Criteria Decision Analysis (MCDA) is the umbrella term for methods that deal with multiple criteria, while DEA and TOPSIS are specific tools within that framework. They all contribute to the process of making informed decisions when there are multiple factors to consider [15]. While DEA is a non-parametric method used to evaluate the relative efficiency of DMUs, which are entities that convert inputs into outputs [33]. TOPSIS ranks alternatives based on their distance to both an ideal solution and a non-ideal solution [16]. In location analysis, DEA and TOPSIS were used by Muvingi, Peer [3. 32], Chen, Li [5] and Zangeneh, Akram[31].

Hierarchical groups DEA super-efficiency refers to a method used in (DEA) to assess the efficiency of DMUs that have a hierarchical structure, where some DMUs are part of larger groups, for example districts and suburbs [3].

It extends the concept of super-efficiency, which is used to

rank and compare DMUs that are already efficient in the traditional DEA analysis [3]. Super-efficiency methods are used to rank and compare efficient units. DEA determines a "best-practice frontier" by identifying the most efficient DMUs and then compares other DMUs against this frontier [33, 35]. Traditionally, DEA identifies efficient DMUs, meaning those that are operating on the best-practice frontier, and inefficient DMUs, those below the frontier [33]. DEA arises from situations where the goal is to determine the productive efficiency of a system by comparing how well the system converts inputs into outputs, while MCDA models have arisen from the need to analyze a set of alternatives according to conflicting criteria [5]

Instead of the traditional cost-benefit analysis in which all criteria are converted to a monetized term as a common comparison ground like the UFL and k-median problems, this framework incorporates non-monetized considerations into the management planning process, thereby improving the project appraisal process [5]. Both quantitative criteria that can be measured in numerical values objectively and qualitative criteria that can only be gauged subjectively are integrated together to generate comprehensive evaluations for all alternatives. Muvingi, Peer [3] categorize the preferable locations of 16 districts and 49 suburbs as the alternatives. The alternatives are ranked according to criteria such as Distance from CBD (D), Road density (RD), Zoning tariff (ZT), Number of shops (NS), Number of mobile phone users (NMPU), Mobile transfer service users (MTSU) and Population (P) as seen on Figure 5.

Identifying the set of alternatives, $A = \{a^1, a^2, a^n\}$ and the set of criteria, $C = \{c^1, c^2, c^q\}$. Step (1) also provides a direct physical measurement as the consequence of alternative a^i on criterion c_j for every i = 1,..., n and j = 1,..., q, denoted by m^i_j, representing the (i, j)-entry of an n × q matrix, called the information (or performance) matrix [5]. The matrix on Figure 7 is transposed compared to the one on Figure 6. The Matrix on Figure 7 shows a Hierarchy of district and suburb alternatives to the left and criterion listed on the top.

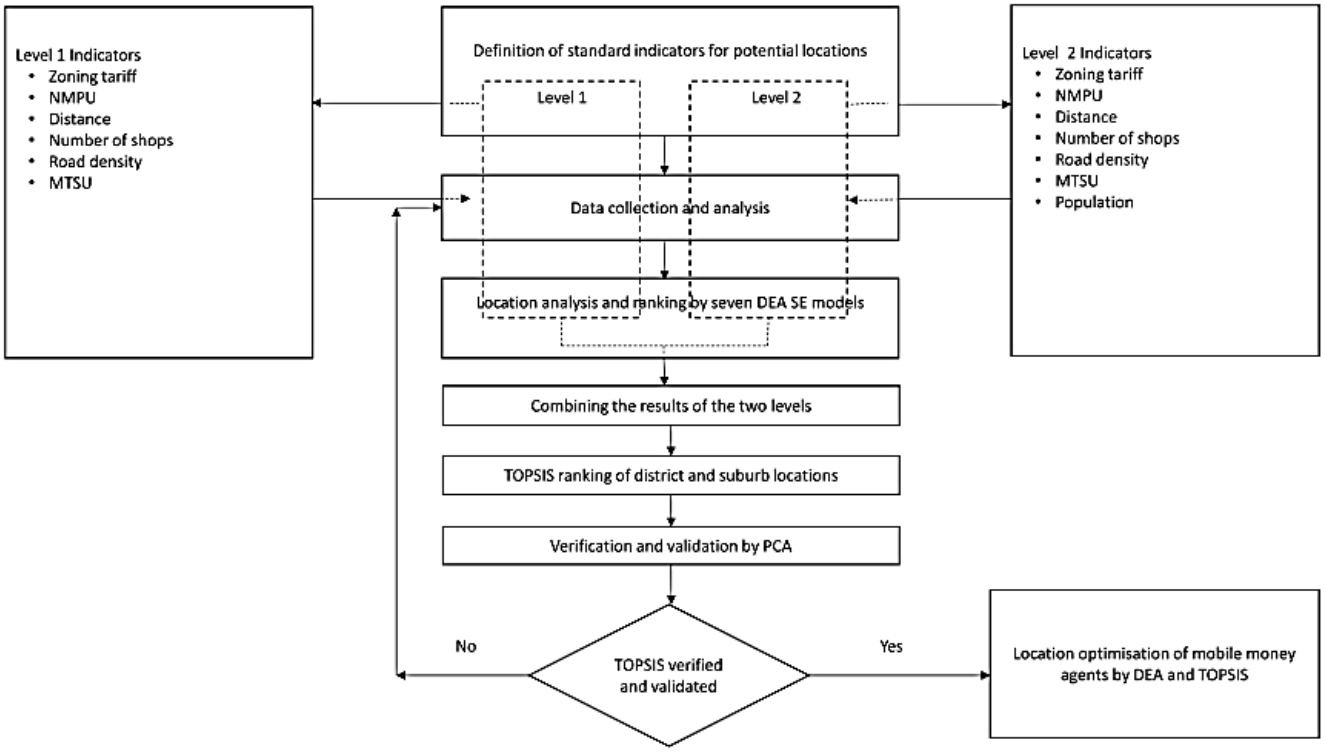


Figure 5. *Integrated model for the locations of mobile money agents. Source: Muvingi, Peer [3]*

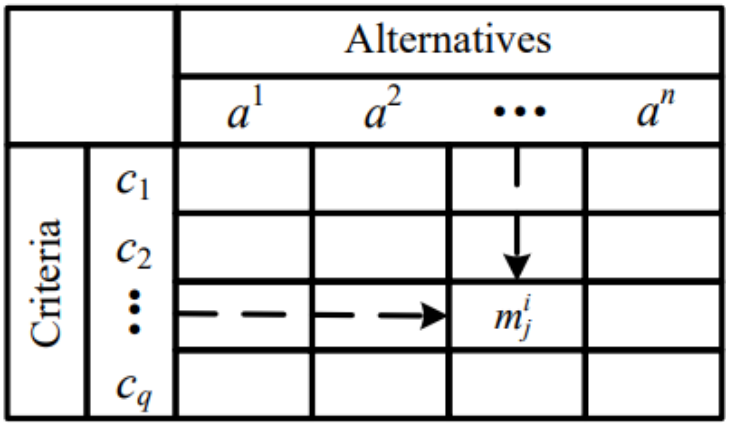


Figure 6. *DEA performance matrix Source: Chen, Li [5].*

Data for level 1 analysis.

District	Suburb	ZT	NMPU	D	NS	RD	MTSU
		Input	Input	Output	Output	Output	Output
Borrowdale	Borrowdale Brook	40.00	3746.00	18.90	2.00	25.30	967.00
	Borrowdale	32.17	4491.00	16.00	18.00	67.91	1567.00
	Greystone Park	32.17	3626.00	16.00	9.00	40.78	2175.00
	Philadelphia	40.00	2935.00	14.50	4.00	13.96	909.00
Budiriro	Ward 33	4.00	46311.00	15.38	47.00	90.66	26265.00
	Ward 43	4.00	45572.00	15.65	24.00	174.83	22212.00
CBD	Belvedere	16.34	8576.00	4.20	26.00	83.05	4059.00
	Monavale	12.56	6447.00	4.90	2.00	12.56	3439.00
	Eastlea	18.06	5365.00	6.30	30.00	47.61	2099.00
	Milton Park	46.26	3195.00	1.70	13.00	46.26	2606.00
Glen Norah	Ward 27	5.00	30928.00	13.98	45.00	50.55	14509.00
	Ward 28	5.00	8184.00	14.70	22.00	45.29	7116.00
Glen View	Ward 30	5.00	47709.00	14.79	26.00	65.21	26151.00
	Ward 31	5.00	18819.00	14.70	61.00	25.42	7976.00
	Ward 32	5.00	20506.00	15.90	33.00	51.71	10649.00
Greendale	Greengrove	21.45	4224.00	12.00	4.00	7.65	2816.00
	Greendale	21.45	4172.00	10.20	37.00	30.83	2185.00
	Glen Lorne	32.17	4326.00	17.50	2.00	16.81	2349.00
Hatcliffe	Hatcliffe	4.00	28481.00	19.62	15.00	152.16	13274.00
Hatfield	Msasa Park	16.06	5329.00	13.20	13.00	13.69	3475.00

Figure 7. *Hierarchical DEA performance matrix. Source: Muvingi, Peer [3]*

In Muvingi, Peer [3], there were three main objectives:

1) Ranking potential suburb and district locations for Mobile Money Agents (MMAs) using seven different SE approaches [3].
2) Adjust the SE ratings of the suburb locations, termed level 1 DMUs, with the SE ratings of the district locations, termed level 2 DMUs, in which they are grouped [3].
3) Combine the seven-suburb aggregated SE (SASE) ratings with the TOPSIS method to get a universal ranking of the suburbs and district locations [3].

Muvingi, Peer [3] sought to incorporate slacks variables directly into the efficiency analysis of DMUs with a hierarchical group structure composed of the suburb and district locations of MMAs. Seven hierarchical DEA SE models that directly incorporated slack variables were developed and applied in the ranking of suburb and district locations [3]. The suburb locations were arranged and grouped in the district locations Noura, Jahanshahloo [34] shows an example of slacks-based measure (SBM) of efficiency of the DMUs,

Noura, Jahanshahloo [34] deal with *n* DMUs with the input and output matrices

$X=(x_{ij})\in R^{m\times n}$ and $Y=(y_{rj})\in R^{s\times n}$, respectively.

They assume that the data set is nonnegative, i.e., $X\geq 0$ and $Y\geq 0$ the production possibility set ρ is defined as [34]:

$$p=\{(x,y)\,|x\geq X\lambda, y\leq Y\lambda, \lambda\geq 0\}\ \text{[34]} \qquad (38)$$

where λ is a non-negative vector in R^n.

A study consider an expression for describing a certain $DMU_o=(x_o, y_o)$ as[34]:

$$x_o=X\lambda+s^-, \qquad (39)$$

$$y_o=Y\lambda-s^+. \qquad (40)$$

With $\lambda\geq 0$, $s^-\geq 0$ and $s^+\geq 0$. The vector $s^-\in R^m$ and $s^+\in R^s$ indicate the input excess and output shortfall of this expression, respectively, and are called slacks.

From the condition $X\geq 0$ and $\lambda\geq 0$, it holds that

$$x_o \geq s^-.$$

Using s^- and s^+, Noura, Jahanshahloo [34] define an index ρ as follows:

$$\rho = \frac{1-\frac{1}{m}\sum_{i=1}^{m} s_i^-/x_{io}}{1+\frac{1}{s}\sum_{r=1}^{s} s_r^+/y_{ro}}. \text{ [34]} \qquad (41)$$

It can be verified that ρ satisfies the properties (i) units invariant and (ii) monotone decreasing in input/output slacks.

From (84), it holds that:

$0 < \rho \leq 1$.

In an effort to estimate the efficiency of (x_o, y_o), they formulate the following fractional program SBM in λ, s^- and s^+.

$$\begin{aligned} \text{Min} \quad & \rho = \frac{1-\frac{1}{m}\sum_{i=1}^{m} s_i^-/x_{io}}{1+\frac{1}{s}\sum_{r=1}^{s} s_r^+/y_{ro}} \\ \text{subject to} \quad & \\ & x_o = X\lambda + s^- \\ & y_o = Y\lambda + s^+ \\ & \lambda \geq 0, s^- \geq 0, s^+ \geq 0. \end{aligned} \quad \text{[34]} \qquad (42)$$

SBM can be transformed into a linear program using the Charnes–Cooper transformation in a comparable way to the CCR model.

The analysis of the level 1 and 2 SE ratings is derived from the seven models implemented for DMUs previously assumed to be homogeneous and not arranged in a group structure [3]. To ensure that those models are applicable to the SE evaluation of DMUs in a two-level hierarchical group structure, additional constraints were added:

$$\begin{aligned} & \delta_o^*(2) = \min \delta_o(2), \\ \text{s.t.} \quad & \delta_o(2)x_{io}^1(2) = \sum_{j=1,j\neq o}^{n} x_{ij}^1(2)\lambda_j + s_i^{-1}, \quad i = 1,\dots,m_1, \\ & \delta_o(2)x_{lo}^2(2) = \sum_{j=1,j\neq o}^{n} x_{lj}^2(2)\lambda_j + s_l^{-2}, \quad l = 1,\dots,m_2, \\ & y_{ro}^1(2) = \sum_{j=1,j\neq o}^{n} y_{rj}^1(2)\lambda_j - s_r^{+1}, \quad r = 1,\dots,s, \\ & \lambda_j, s_i^{-1}, s_l^{-2}, s_r^{+1} \geq 0, \quad \forall j, i, l, r, \end{aligned}$$

Figure 8. *DSE-1 ratings. Source: Muvingi, Peer [3].*

DSE-1 ratings [3].

A study introduced an improved Super SBM model that can handle negative data, which is an improvement from the traditional super SBM models. They propose a non-radial SE model capable of ranking efficient DMUs [3].

$$\begin{aligned} & \delta_o^*(2) = \min \frac{\frac{1}{(m_1+m_2)}\left(\sum_{i=1}^{m_1} x_i^{*1}(2)/x_{io}^1(2) + \sum_{l=1}^{m_2} x_l^{*2}(2)/x_{lo}^2(2)\right)}{\frac{1}{s}\sum_{r=1}^{s} y_r^{*1}(2)/y_{ro}^1(2)} \\ \text{s.t.} \quad & x_i^{*1}(2) = \sum_{j=1,j\neq o}^{n} x_{ij}^1(2)\lambda_j + s_i^{-1}, \quad i = 1,\dots,m_1, \\ & x_l^{*2}(2) = \sum_{j=1,j\neq o}^{n} x_{lj}^2(2)\lambda_j + s_l^{-2}, \quad l = 1,\dots,m_2, \\ & y_r^{*1}(2) = \sum_{j=1,j\neq o}^{n} y_{rj}^1(2)\lambda_j - s_r^{+1}, \quad r = 1,\dots,s, \\ & x_i^{*1}(2) \geq x_{io}^1(2), \quad i = 1,\dots,m_1, \\ & x_l^{*2}(2) \geq x_{lo}^2(2), \quad l = 1,\dots,m_2, \\ & y_r^{*1}(2) \leq y_{ro}^1(2), \quad r = 1,\dots,s, \\ & y_r^{*1}(2), \lambda_j, s_i^{-1}, s_l^{-2}, s_r^{+1} \geq 0, \quad \forall j, i, l, r. \end{aligned}$$

Figure 9. *DSE-2 ratings. Source: Muvingi, Peer [3]*

Muvingi, Peer [3] consider a modified slacks based SE measure in the presence of negative data. The SE methodology, for example that of Bajec, Tuljak [35]. is crafted to manage data uncertainties, while the multi-level nature of mobile money agent networks in Muvingi, Peer [3] can be somewhat complex.

$$\begin{aligned} & \delta_o^*(2) = \min 1 + t\gamma + (1-t)\left(\sum_{i=1}^{m_1}\frac{s_{1i}^{+1}}{x_{io}^1(2)} + \sum_{l=1}^{m_2}\frac{s_{1l}^{+2}}{x_{lo}^2(2)} + \sum_{r=1}^{s}\frac{s_{2r}^{-1}}{y_{ro}^1(2)}\right) \\ \text{s.t.} \quad & \sum_{j=1,j\neq o}^{n} x_{ij}^1(2)\lambda_j + s_{1i}^{-1} - s_{1i}^{+1} = x_{io}^1(2), \quad i = 1,\dots,m_1, \\ & \sum_{j=1,j\neq o}^{n} x_{lj}^2(2)\lambda_j + s_{1l}^{-2} - s_{1l}^{+2} = x_{lo}^2(2), \quad l = 1,\dots,m_2, \\ & \sum_{j=1,j\neq o}^{n} y_{rj}^1(2)\lambda_j + s_{2r}^{-1} - s_{2r}^{+1} = y_{ro}^1, \quad r = 1,\dots,s, \\ & s_{1i}^{+1} \leq \gamma, \quad i = 1,\dots,m_1, \\ & s_{1l}^{+2} \leq \gamma, \quad l = 1,\dots,m_2, \\ & s_{2r}^{-} \leq \gamma, \quad r = 1,\dots,s, \\ & \lambda_j, s_{1i}^{\pm 1}, s_{1l}^{\pm 2}, s_{2r}^{\pm 1} \geq 0, \quad \forall j, i, l, r. \end{aligned}$$

Figure 10. *DSE-5 ratings. Source: Muvingi, Peer [3].*

The Muvingi, Peer [3] study implemented the TOPSIS method to obtain a clear ranking of all the suburb locations through the incorporation of all the seven DSE ratings [3]. Their ranking results revealed that most of the highly-rated suburb areas are located in low density areas, while the opposite was observed in the TOPSIS ranking of district locations [3]. They conjectured that a complete ranking of the DMUs through TOPSIS was achieved for all district and suburb locations [3]. The adoption of the TOPSIS method provided rankings derived from the ratings of seven diversified SE approaches, which ensured that the merits and demerits of each DEA SE model were considered in the final ranking of DMUs [3]. The TOPSIS method refined the ranking by comparing each agent's performance against an ideal solution, thereby overcoming the limitations of traditional

DEA in distinguishing between efficient units.

DSE ratings and rankings.

District	DSE-1	DSE-2	DSE-3	DSE-4	DSE-5	DSE-6	DSE-7
Borrowdale	1.1528 (9)	1.0346 (7)	1.0971 (9)	1.0971 (9)	1.1665 (8)	1.0440 (13)	1.3144 (8)
Budiriro	1.5007 (4)	1.0547 (5)	1.2285 (6)	1.2285 (6)	1.5282 (4)	1.4613 (7)	1.5486 (4)
CBD	1.0491 (14)	1.0152 (10)	1.0229 (14)	1.0229 (12)	1.0621 (14)	1.1336 (10)	1.0915 (11)
Glen Norah	1.1449 (10)	1.0291 (9)	1.1036 (8)	1.1036 (7)	1.1623 (9)	1.5899 (5)	1.3790 (7)
Glen View	1.0717 (13)	0.9715 (13)	1.0351 (13)	1.0200 (13)	1.1002 (12)	1.0860 (11)	1.0715 (12)
Greendale	1.3192 (6)	1.1532 (4)	1.2439 (5)	1.2439 (5)	1.5264 (5)	1.5136 (6)	1.4698 (6)
Hatcliffe	2.5495 (1)	1.2505 (1)	2.1460(1)	2.1460 (1)	1.8898 (1)	2.2466 (3)	2.5096 (1)
Hatfield	1.5968 (3)	1.1785 (3)	1.3455 (3)	1.3455 (3)	1.6015 (3)	1.2830 (8)	2.0868 (2)
Highfield	0.9933 (15)	0.9487 (14)	1.0000 (15)	0.9854 (15)	1.0000 (15)	1.0000 (15)	1.0000 (12)
Highlands	1.3654 (5)	1.0373 (6)	1.3281 (4)	1.3281 (4)	1.2676 (6)	1.0717 (12)	1.2829 (9)
Kuwadzana	0.9697 (16)	0.9395 (15)	1.0000 (15)	0.9895 (14)	1.0000 (15)	1.0000 (15)	1.5078 (5)
Mabvuku	1.6654 (2)	1.2162 (2)	1.5159 (2)	1.5159 (2)	1.7731 (2)	2.3339 (2)	1.2586 (10)
Marlborough	1.1612 (8)	0.8388 (16)	1.0537 (11)	0.8847 (16)	1.1391 (11)	1.0360 (14)	1.0000 (12)
Mbare	1.1742 (7)	1.0034 (11)	1.1077 (7)	1.1021 (8)	1.1918 (7)	1.2722 (9)	1.8302 (3)
Warren Park	1.1007 (11)	1.0309 (8)	1.0624 (10)	1.0624 (10)	1.1619 (10)	2.0827 (4)	1.0000 (12)
Waterfalls	1.0933 (12)	0.9802 (12)	1.0385 (12)	1.0385 (11)	1.0866 (13)	2.3706 (1)	1.0000 (12)

Figure 11. *DSE ratings and rankings. Source: Muvingi, Peer [3].*

TOPSIS Ranking of suburb locations.

Suburb	A_1^+	A_1^-	$D_{\rho_1}^-$	$D_{\rho_1}^+$	C_{ρ_1}	Rank
Greendale	0.0034	0.0732	0.2706	0.0580	0.8235	1
Hatfield	0.0083	0.0904	0.3007	0.0912	0.7673	2
Waterfalls	0.0182	0.0437	0.2090	0.1349	0.6077	3
Ward 20	0.0230	0.0297	0.1725	0.1518	0.5319	4
Highlands	0.0416	0.0130	0.1140	0.2041	0.3583	5
Ward 19	0.0559	0.0075	0.0866	0.2363	0.2681	6
Newlands	0.0567	0.0070	0.0835	0.2382	0.2596	7
Logan Park	0.0851	0.0103	0.1013	0.2918	0.2578	8
Ward 21	0.0759	0.0041	0.0637	0.2755	0.1878	9
Ward 28	0.0722	0.0030	0.0544	0.2688	0.1684	10
Ward 11	0.0732	0.0022	0.0472	0.2705	0.1487	11
Ward 26	0.0755	0.0020	0.0443	0.2748	0.1389	12
Rhodesville	0.0854	0.0020	0.0452	0.2923	0.1339	13
Ward 33	0.0790	0.0013	0.0366	0.2811	0.1151	14
Ward 43	0.0790	0.0013	0.0363	0.2812	0.1145	15
Ward 4	0.0812	0.0011	0.0332	0.2849	0.1044	16
Hatcliffe	0.0825	0.0010	0.0314	0.2872	0.0985	17
Westlea	0.0826	0.0010	0.0312	0.2874	0.0978	18
Msasa Park	0.0846	0.0008	0.0282	0.2909	0.0885	19
Borrowdale	0.0850	0.0007	0.0270	0.2915	0.0847	20
Ward 37	0.0854	0.0007	0.0262	0.2922	0.0824	21
Warren Park D	0.0909	0.0007	0.0263	0.3014	0.0804	22
Kambuzuma	0.0864	0.0007	0.0256	0.2939	0.0800	23
Ward 31	0.0856	0.0006	0.0248	0.2925	0.0782	24

Figure 12. *TOPSIS rankings. Source: Muvingi, Peer. [3]*

Muvingi, Peer [3] used the Principal Component Analysis (PCA) method and the Spearman correlation analysis to verify and validate the TOPSIS results. They found the accuracy of the district location TOPSIS ranking approach to be moderate, at the rate of 61 percent on the observed suburb locations. Mvere [6, 23] used the ESMVERE-R-hamming k-median clustering method to infer opinions of whether there would be a need to introduce a new collection system for ULABs to which 55 percent of the respondents confirmed the need for a new ULAB collection system [23]. However, the study, Muvingi, Peer [3], contributed to the literature focused on comparing TOPSIS and PCA, as most of the work is on the comparison of DEA and PCA. While Mvere [6, 23] introduced an original, completely new and innovative way of analyzing questionnaire data.

The main contribution of the study Muvingi, Peer [3] includes:

1) It provides insight to bank and Mobile Money Operators (MMOs) regulators into how the characteristics of locations are related to the use of mobile money, represented by MTSU.
2) Regulators can use the efficiency analysis of the loca-

tions in the study as a proxy measure for potential financial inclusion in a given location.

3) The study provided a bench-marking analysis of potential MMA locations, allowing banks to strategically locate their branches depending on the influence of MMAs locations on their operations.
4) The location performance results guide MMOs for MMA network scaling.

3. Conclusion

In situations which demand urgent coverage of the entire population indiscriminately, for example situations where there is a national disaster or a disease outbreak, to cover the entire population, the UFL problem or the k-median problem would be a better choice technique. One of the most complex decisions in such a real-world case would be to determine what constitutes a facility, how much a facility would cost to build and what would represent a client [41]. The information would determine how many potential facility locations sites and client locations sites would be in an instance, thus the size of an instance.

In such situations, the ESMVERE CPLEX algorithms would prove to be better solutions because of them being user friendly on the input and speedy in how they process the data. Lead poisoning is a danger, although it is a subtle and long-term danger it is still of paramount importance that every community has safe disposal facilities near them. Thus, the disposal of such hazardous waste would not involve preference or competition in a normal scenario.

This acutely differs from situations where the goal is to attract certain types of customers. This situation requires more information about the potential site. Mobile money services and banking services in general involve preferences. The preference of the decision makers in selecting places that maximize their value and minimize their costs, places that are away from competition, and places that are convenient and easily accessible to their customers. On the other hand, the customer usually has a choice in selecting a service from a particular location. They may not always seek a facility which is near them, but they may choose to seek their services in places that are less congested, easier to reach, where the mobile phone network responds faster etc. These are conditions where the DEA and TOPSIS technique would be an ideal solution.

In the case of Walmart [37], for 50 stores sitting in Washington state, the ESMVERE algorithms k-median version, can be more useful in the decision because of speediness to a solution which covers the whole state. However, in such a case, to a greater extent, the Hierarchical groups DEA and group TOPSIS technique will afford more information which could result in better decisions. This is a situation in which the two techniques could be complimentary and used together.

The ESMVERE CPLEX k-median problem solver algorithm is used to determine the optimum or near optimum locations which minimize the average travel time of every customer in the state of Washington to a Walmart shop of maybe about 50 locations. Then the Hierarchical groups DEA and group TOPSIS technique will determine and rank the best out of those 50 locations based on relevant criteria. This could mean ranking the performance of different values of k such as k= 40 or k=50 or k=60 to determine the best value for k which covers the entire population [40]. The two techniques used in conjunction could mean a more rigorous and time demanding data gathering process but better decision making.

The dual fitting and factor revealing LP technique remains complex to implement in such a real-world case as that of Mvere [1, 2] because of the input of a distance matrix. To measure 22464 location distances accurately would be tedious, this solution compared to the input of 156 GPS Coordinate facility locations and 144 GPS coordinate client locations which proves the usefulness of the new solution invented by Mvere [1, 2]. Although the algorithm Jain and Mahdian [4] on the Tables 1, 2, 3, 4 and Figures 1 and 2, is well formulated in theory, more needs to be done to achieve its use to solve real world problems. Future work in the area might mean reformulation of the algorithm to adopt the methodology of [1, 2] which include the Euclidean Distance formula and GPS coordinate input,

While Muvingi, Peer [3] offers a hybrid evaluation framework that integrates hierarchical groups DEA with group TOPSIS to assess the efficiency of mobile money agent's locations in Zimbabwe. The detailed methodology and empirical application within the Zimbabwean context offer valuable insights for evaluating service delivery in digital financial systems.

The ESMVERE CPLEX UFL problem solver algorithms can extends to a broader range of facility location challenges across different industries and sectors. The algorithm's robust scalability and solution quality make it suitable for various UFL location problems where decision-makers must identify optimal facility sites to minimize costs or maximize service efficiency without capacity constraints.

Beyond hazardous waste management, the ESMVERE UFL algorithm [2] can address applications such as:

1) Health Care Facility Location Optimizing the placement of clinics, hospitals, or emergency response centers to improve accessibility and reduce response times in urban or rural areas.
2) Retail and Distribution Centers Determining the best locations for warehouses or retail outlets to balance logistics costs and market coverage efficiently.
3) Public Service Infrastructure Siting facilities like schools, fire stations, or public offices where service coverage and proximity to populations are critical.
4) Telecommunications Networks Facilitating the positioning of network hubs or base stations to ensure optimal coverage with minimal deployment costs.

These potential applications leverage the algorithms capability to handle large problem instances, with complex cost

structures and constraints, enabling practical deployment in varied operational contexts. The balanced integration of sophisticated optimization techniques within the ESMVERE framework enables the algorithms to maintain scalability without compromising on solution accuracy.

In conclusion, this systematic review successfully highlights the differences and similarities among three significant location science methods. Each method offers different real world applications, as demonstrated through practical case studies in mobile banking and hazardous waste management. To enhance the practical utility of these methods, further exploration of their implications is essential. Future research should focus on integrating these techniques with emerging technologies to improve decision making processes in various sectors

Abbreviations

CBD	Central Business District
DEA	Data Envelopment Analysis
DMU	Decision-Making Units
GPS	Global Positioning System
HYB	Hybrid Heuristics
ILS	Iterated Local Search
MCDA	Multi Criteria Decision Analysis
MDAS	Multi-Start Descending Algorithm with Screening
MMA	Mobile Money Agents
MMO	Mobile Money Operators
Np-hard	Non-Deterministic Polynomial Time Hard
Opl	Optimization Programming Language
P	Polynomial Time
PCA	Principal Component Analysis
TOPSIS	Technique for Order Performance (preference) by Similarity to Ideal Solution
SASE	Suburb Aggregated Super Efficiency
SE	Super Efficiency
SBM	Slacks Based Measure
UFL	Uncapacitated Facility Location
ULAB	Used Lead Acid Battery

Acknowledgments

I would like to thank my parents, My Mum who has made all my research efforts possible., My whole family, my darling, my colleagues. The people of Zimbabwe, the people of Mauritius, the people of Africa, and all the people of the world. I would like to thank My lord and Savior JESUS CHRIST for turning my dream into a reality.

Author Contributions

Emmanuel Siyawo Mvere is the sole author. The author read and approved the final manuscript.

Conflicts of Interest

The authors declare no conflicts of interest.

Appendix

The ESMVERE CPLEX UFL Problem Solver Model

The ESMVERE CPLEX UFL problem solver algorithm is a software invention that solves the UFL problem. It is a tool that aids, and guides management in making sound decisions on where to locate facilities. It aids management in the decision of where the best places are, lowest cost places, the minimum distance places to locate facilities considering that there are opening costs and the Euclidean distances, direct line distances of GPS coordinate points?

The CPLEX Opl Code

```
int n =…; //number of facilities
int m = ⋯; //number of clients declared in data files
range ulabfacilities = 1..n;
range ulabcities = 1..m; int fi[ulabfacilities] =…;
tuple Location {float x; float y; }
tuple edge {int i; int j; }
setof (edge) edges = {< i, >
j | i in ulabfacilities, j in ulabcities };//defines an edge
float c[edges]; //An Array of each edge on the plane measured separately
Location ulabcityLocation [ulabcities];
Location ulabfacilityLocation [ulabfacilities];
execute {
  function getDistance (ulabcity1, ulabfacilitycity1) {
 //Euclidean distance between each facility and client
  return opl.sqrt(opl.pow(ulabcity1.x −
ulabfacility1.x, 2) + opl.pow(ulabcity1.y
−ulabfacility1.y, 2)) }
for(var i inulabfacilities) {
ulabfacilityLocation[i].x;
ulabfacilityLocation[i].y;
}
for(var j inulabcities) {
ulabcityLocation[j].x;
ulabcityLocation[j].y;
}
  for(var e in edges){ //distance between each edge "heuristic"
  c[e] =
getDistance(facilityLocation[e.i], cityLocation[e.j])
  } }
  dexpr float OpeningCost =
sum(i in ulabfacilities) fi[i] * y[i];
  dexpr float TotalDistance = sum(e in edges) c[e] *
x[e];
  Minimize OpeningCost + TotalDistance; //objective
```

function

$subject\ to$ { //constraints from p-median [26]

$forall(j\ in\ ulabcities: j >= 1)$

$sum\ (i\ in\ ulabfacilities)\ x[< i,j >] == 1;$

$forall(i\ in\ ulabfacilities, j\ in\ ulabcities)$

$x[< i,j >] <= y[i];$

$sum(i\ in\ ulabfacilities)$

}

```
1/*********************************************
2 * OPL 20.1.0.0 Data
3 * Author: Emmanuel Siyawo Mvere
4 * Creation Date: OCT 28, 2022 at 6:58: PM
5 *********************************************/
6
7 n=7;
8
9 m=21;
10
11 ulabcityLocation=[<0.1 6.4> <0.2 0.9> <0.75 2.1> <1 6.9> <2.44 5.6> <3.1 3.4>
  <3.45 0.9> <4.25 5.6> <5.15 2.2> <5.6 0.95> <6.1 6.95> <7.55 6.9> <7.9 4.4>
  <9.2 1> <9.35 3.5> <9.65 4.4><10.55  5.7><12.3 2.25> <12.4 4.45> <12.55 7>
  <12.65 1>];
12
13 ulabfacilityLocation=[<0.7 3.8> <2.2 1.2> <4.3 7.3> <5.8 4.75> <7.3 1.3> <10.2
  2.5><10.8 7.3> ];
14
15 fi=[30, 20, 50, 20, 40, 60, 10];
```

Figure 13. *Input File. Input file. Source Mvere [1, 2, 6]*

Input File

In this case the input are x and y coordinates [<x y> <x y>] can be replaced by

GPS Coordinates [< -20.172544 57.4883317 > <-20.165595 57.492625>] compare with a distance matrix of the same coordinate points.

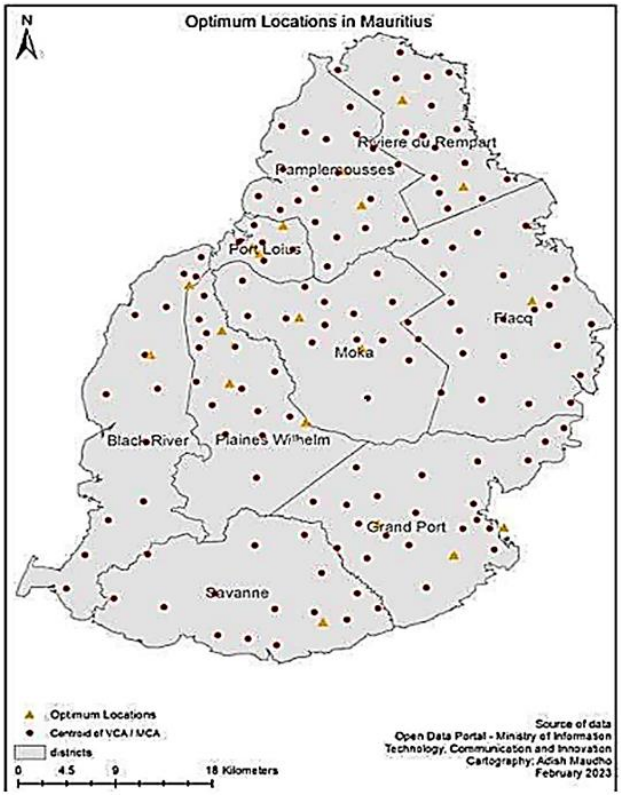


Figure 14. *k-median Optimum Solution. Source Mvere [1, 6], Maudho [45].*

Figure 14 shows the final near optimum solution which hosts nineteen facilities.

References

[1] E. S. Mvere. (2025) The Near Optimal Siting of Hazardous Waste (Used Lead Acid Battery (ULAB)) Collection Facilities in The Republic of Mauritius Using the Esmvere Cplex k-median solver algorithm. In K. Madhu (Eds,), Advances in Numerical Analysis and Applications. Vol 1, pp, 16-46, ISBN: 978-81-958975-6-8.

[2] E. S. Mvere.(2025) The Near Optimal Siting of Hazardous Waste (Used Lead Acid Battery (ULAB)) Collection Facilities in The Republic of Mauritius Using the ESMVERE Cplex UFL problem solver algorithm. In K. Madhu (Eds,), Advances in Numerical Analysis and Applications. Vol 1, pp, 47-80, ISBN: 978-81-958975-6-8.

[3] Muvingi, J., Peer, A. A. I., Hosseinzadeh Lotfi, F., & Ozsoy, V. S. (2023). Hierarchical groups DEA super-efficiency and group TOPSIS technique: Application on mobile money agents' locations, Expert Systems with Applications, Volume 234, 0957-4174, https://doi.org/10.1016/j.eswa.2023.121033

[4] Jain, K., Mahdian, M., Markakis, E., Saberi, A. & Vazirani, V. V. (2003). Greedy facility location algorithms analyzed using dual fitting with factor-revealing Lp. J. ACM.

[5] Chen, Y., Li, K. W., Xu, H.(2009) A DEA-TOPSIS method for multiple criteria decision analysis in emergency management. J. Syst. Sci. Syst. Eng. 18, 489–507. https://doi.org/10.1007/s11518-009-5120-3

[6] Mvere, E. S (2025). A Systematic Review Of The Esmvere Algorithms, Which Are Contributions In The Field Of Uncapacitated Facility Location Problems And K - Median Problems *International Journal of Science Academic Research*, Vol. 06 No. 2, 2025, pp. 9352-9362

[7] Jain, K. & Vazirani, V. (1999). Primal-dual approximation algorithms for metric facility location and k-median problems. in 'Proceedings of the 40th Annual Symposium on Foundations of Computer Science'. FOCS '99. p. 2.

[8] Faiz, S. & Krichen, S. (2012) Geographical Information Systems and Spatial Optimization Vol 1, pp, 176, ISBN 9780429072079 CRC Press

[9] Harder, H. D. (2023) Exact Algorithm or Heuristic? https://medium.com/data-science/exact-algorithm-or-heuristic-20d59d7fb359 *pp. 2025–05–19*

[10] Li, S. (2014). Approximation Algorithms for Network Routing and Facility Location Problems. PhD thesis. Princeton University.

[11] Svensson, O. (2009). Approximability of Some Classical Graph and Scheduling Problems. PhD thesis. University of Lugano.

[12] Swamy, C. (2004). Approximation Algorithms for Clustering Problems. PhD thesis. Cornell University.

[13] Woeginger, G. J. (2003). Exact Algorithms for NP-Hard Problems: A Survey. In: Jünger, M., Reinelt, G., Rinaldi, G. (eds) Combinatorial Optimization — Eureka, You Shrink!. Lecture Notes in Computer Science, vol 2570. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-36478-1_17

[14] IBM (2025) Decision Optimization at a glance https://www.g2.com/products/ibm-decision-optimization/pricing pp. 2025–02–25

[15] Charles, V., Aparicio, J., Zhu, J.(2019) The curse of dimensionality of decision-making units: A simple approach to increase the discriminatory power of data envelopment analysis, *European Journal of Operational Research, Volume 279, Issue 3*, Pages 929-940, ISSN 0377-2217,

[16] Ramdania, D. R., Manaf, K., Junaedi, F. R., Fathonih, A., Hadiana, A,(2020) TOPSIS Method on Selection of New Employees' Acceptance,"6th International Conference on Wireless and Telematics (ICWT), Yogyakarta, pp. 1-4, https://doi.org/10.1109/ICWT50448.2020.9243658

[17] Li, S.(2013) A 1.488 approximation algorithm for the Uncapacitated facility location problem, Information and Computation, Vo 222, 2013, Pages 45-58, https://doi.org/10.1016/j.ic.2012.01.007

[18] Nguyen, K. T.,(2019) Primal-Dual Approaches in Online Algorithms, Algorithmic Game Theory and Online Learning. Operations Research [math. OC]. Université Paris Sorbonne. fftel-02192531f.

[19] Ebrahimnejad, A.,(2012) A primal-dual method for solving linear programming problems with fuzzy cost coefficients based on linear ranking functions and its applications," *International Journal of Industrial and Systems Engineering, Inderscience Enterprises Ltd*, vol. 12(2), pages 119-140.

[20] Chu, C., Antonio, J., (1999) Approximation Algorithms to Solve Real-Life Multicriteria Cutting Stock Problems *Operations Research* 47: 4, 495-508.

[21] Wikipedia contributors. (2025, April 25). Approximation algorithm. In Wikipedia, The Free Encyclopedia. Retrieved 23: 05, May 25, 2025, from https://en.wikipedia.org/w/index.php?title=Approximation_algorithm&oldid=1287316159

[22] Maria, G. L., Hua, W., Henry, W.,(2009). A Stable Primal-Dual Approach for Linear Programming under Nondegeneracy Assumptions. *Computational Optimization and Applications*. 44. 213-247. https://doi.org/10.1007/s10589-007-9157-2

[23] Mvere, E. S (2025) An Analysis of the Used Lead Acid Battery (ULAB) Collection Network Mauritius Using the ESMVERE-R-Hamming-k-median Clustering Method., *International Journal of Science Academic Research*, Vol. 06 No. 1, 2025.

[24] Balinski, M. (1966). On finding integer solutions to linear programs In: Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems, pp. 225-248.

[25] Hakimi, S. L., (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. Operations Research, 12(3), pp. 450-459.

[26] Cacioppi, P.& Watson, M. (2014). A Deep Dive into Strategic Network Design Programming: OPL CPLEX Edition. Northwestern University Press.

[27] Mistry, S (2019) Optimization-Facility-Location [source code]. https://github.com/samarthmistry/Optimization-Facility-Location/blob/master/UFL.cpp

[28] Srinivasan, G. (2012). Mod-08 Lec-32 Location problems -- p median problem, Fixed charge problem. https://www.youtube.com/watch?v=82OUSjOM2bg pp. 2025–05–26.

[29] Cornuejols, G., Nemhauser, G & Wolsey, L (1984). The Uncapacitated Facility Location Problem. Cornell University

[30] Caceres, H. (2014). CPLEX Seminar Getting started with CPLEX Studio (part 2). https://www.youtube.com/watch?v=URHDTzzDJak pp. 2025–05–26.

[31] Zangeneh, M., Akram, A., Nielsen, P., Keyhani, A., & Banaeian, N. (2015). A solution approach for agricultural service center location problems using TOPSIS, DEA and SAW techniques. In P. Golinska, & A. Kawa (Eds.), *Technology management for sustainable production and logistics* (pp. 25–56). Springer, Berlin, Heidelberg, http://dx.doi.org/10.1007/978-3-642-33935-6_2

[32] Muvingi, J., Peer, A. A. I., Hosseinzadeh Lotfi, F., & Ozsoy, V. S. (2022). Hierarchical groups DEA cross-efficiency and TOPSIS technique: An application on mobile money agents locations. *International Journal of Mathematics in Operational Research*, *21*(2), 171–199. http://dx.doi.org/10.1504/IJMOR.2020.10037455

[33] Labijak-Kowalska, A., Kadziński, M., & Mrozek, W. (2023). Robust Additive Value-Based Efficiency Analysis with a Hierarchical Structure of Inputs and Outputs. *Applied Sciences*, *13*(11), 6406. https://doi.org/10.3390/app13116406

[34] Noura, A. A.,. Jahanshahloo, H. F., G. R., Rashidi, S. F. (2011) Super-efficiency in DEA by effectiveness of each unit in society, *Applied Mathematics Letters, Volume 24, Issue 5, Pages 623-626, ISSN 0893-9659.*

[35] Bajec, P., Tuljak-Suban, D., & Zalokar, E. (2021). A Distance-Based AHP-DEA Super-Efficiency Approach for Selecting an Electric Bike Sharing System Provider: One Step Closer to Sustainability and a Win–Win Effect for All Target Groups. *Sustainability*, *13*(2), 549. https://doi.org/10.3390/su13020549

[36] Li, S.(2013) Approximation Algorithms for Facility Location Problems and Network Routing Problems. https://www.youtube.com/watch?v=0HHo7PWHcuQ pp. 2025–05–26.

[37] Mahdian, M., and Spielman, D. A. (2004). Facility location and the analysis of algorithms through factor-revealing programs. Ph. D. Dissertation. Massachusetts Institute of Technology, USA. Order Number: AAI0806252.

[38] Jain, K., Mahdian, M. and Saberi, A. (2002). A new greedy approach for facility location problems. *in* 'Proceedings of the thirty-fourth

[39] Guha, S. and Khuller, S. (1998). Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms* p. pp. 649–657.

[40] Ahmadian, S. (2017). Approximation Algorithms for Clustering and Facility Location Problems. PhD thesis. University of Waterloo.

[41] Peidro, D., Martin, X., Panadero, & Juan, A. (2024). Solving the Uncapacitated facility location problem under uncertainty: a hybrid tabu search with path-relinking simheuristic approach

[42] Chen, H., Murray, A. T. & Jiang, R. Open-source approaches for location cover models: capabilities and efficiency. *J Geogr Syst* 23, 361–380 (2021). https://doi.org/10.1007/s10109-021-00350-w

[43] Dam, T., Ta, A. T., and Mai, T.(2003) Robust maximum capture facility location under random utility maximization models, *European Journal of Operational Research,* Vol 310, Issue 3, pp 1128-1150.

[44] Tong D., Murray, A. T.(2009) Maximizing coverage of spatial demand for service. Pap Reg Sci 88(1): 85– 97. https://doi.org/10.1111/j.1435-5957.2008.00168.x

[45] Maudho, A. (2023). ArcGIS Cartography Open Data Portal, Ministry Of Information Technology Communication and Innovation Mauritius.

[46] Hamdani, and Wardoyo, R. (2016). The complexity calculation for group decision making using TOPSIS algorithm. In *AIP conference proceedings* (Vol. 1755, No. 1, p. 070007). AIP Publishing LLC.

[47] Muvingi, J., Peer, A. A. I., Jablonsk ý, J., Azizi. H., and Lotfi, H. F., (2024). Hierarchical fuzzy DEA model with double frontiers combined with TOPSIS technique: application on mobile money agents locations, *OPSEARCH*, Springer; Operational Research Society of India, vol. 61(3), pages 1154-1191, September.

Biography

Emmanuel Siyawo Mvere earned his MPhil in Operational Research, in the Department of Mechanical and Production Engineering, Faculty of Engineering of the University of Mauritius. He earned his MSc Applied Statistics with Operational Research in the Department of Applied Mathematical Sciences, School of Innovative Technologies and Engineering, of the University of Technology Mauritius. He earned his BTech (honours) Degree in Electronic Commerce in the School of Business and Management Sciences of the Harare Institute of Technology, Zimbabwe. He earned his Certificate in Applied Information Technology specializing in Network Engineering in the School of Technology, University of Zimbabwe, Inventor of software solutions: The ESMVERE CPLEX UFL problem solver algorithm, The ESMVERE CPLEX k-median problem solver algorithm, The ESMVERE CPLEX k-UFL problem solver algorithm and the ESMVERE-R-Hamming-k-median clustering data analysis method, among many others. Recently published by invite, 2 book chapters in an editorial book: Advances in Numerical Analysis and Applications.