Tabu Search for Multi-Period Facility Location: Uncapacitated Problem with an Uncertain Number of Servers

Amit Kumar Vatsa
Diptesh Ghosh

W.P. No. 2014-11-03
November 2014

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD – 380015
INDIA

Tabu Search for Multi-Period Facility Location: Uncapacitated Problem with an Uncertain Number of Servers

Amit Kumar Vatsa
Diptesh Ghosh

Abstract

This paper supersedes the work presented in WP.No.2014-02-06. We study the problem of allocating doctors to primary health centers (PHC). We model the problem as a multi-period uncapacitated facility location problem under uncertainty. The problem is unconventional in that the uncertainty is in the number and period of availability of doctors. This work aims to determine the sequence of opening facilities (assigning doctors to the PHC) over multiple periods so as to cover the maximum demand. We use a minmax regret approach to solve the problem under uncertainty. We present solution techniques using tabu search and compare our solutions with optimal solutions obtained using commercial solvers. We see that one of our tabu search algorithms is faster and yields optimal solutions in the problems we tested on.

Keywords: Facility location, Heuristics, Tabu search

1 Introduction

This study has been motivated by the primary healthcare sector in developing countries. Providing basic health care to its citizens is an important function of any government. Almost all the developing countries have rural populations that are larger than urban populations. Despite this, healthcare facilities are mostly concentrated in the urban regions. Policy makers try to minimize this disparity and make basic health amenities available to everyone in the rural areas as well.

It was suggested in the Alma-Ata declaration of 1978 by the members of World Health Organization (WHO) that there should be one Primary Health Center (PHC), which is essentially a single doctor clinic, for a population of 30,000 in plains and 20,000 for hilly regions. Many developing countries have tried to achieve this goal. However, due to various reasons these targets have not been fulfilled (Walley et al., 2008; Rohde et al., 2008). Moreover, rapid growth in populations in most of these countries require that the targets be revisited. With the economic development in many of these countries and the resulting rise in income, the urban-rural disparity will be huge if basic amenities are not made available at the local level in rural areas. Locating the PHCs in rural areas so as to meet all the local demand is a major challenge for the government. The government tries to establish a network of PHCs so that basic health care facilities are available to the largest possible rural population. In most of the developing countries, availability of doctors in rural areas is a constraint, even if the physical infrastructure exists (Lawn et al., 2008). When doctors become available, it needs to be decided to which among those PHCs without a doctor should the doctor be assigned.

PHCs that are once made operational are rarely closed down due to various political and social reasons. The populations of the villages that PHCs cater to, change over time. Moreover, there is uncertainty in the number and period of availability of doctors. *Ad hoc* assignment of available doctors to PHCs mostly results in a sub-optimal population coverage. Determining the sequence of opening facilities (assigning doctors to the PHC) at the beginning of the planning horizon, helps to plan for the maximum population coverage over the planning horizon. Moreover, it results in transparency in policy making and implementation. We define the uncapacitated PHC location problem (UPHCLP) of determining the sequence in which the PHCs should be assigned with a doctor. In our work we consider a discrete planning horizon consisting of several periods. The number of doctors who become available in each period of the planning horizon is uncertain. Our solution aims to minimize the maximum regret from an *ex post* optimal population coverage, i.e., the population coverage that could have been achieved had the numbers of doctors who will join in each period of the planning horizon been known *a proiri*.

This paper models the UPHCLP and develop a solution methodology to determine the sequence of opening facilities, i.e., assigning doctors to the PHCs. The remainder of this paper is organized as follows. We provide an overview of the existing literature in Section 2. In Section 3, we provide a formulation of the problem. In Section 4, we give some dominance rules to speed up the heuristics and briefly discuss the local search and the tabu search based heuristics. In Section 5, we report our computational experience with commercial solvers and tabu search based heuristics. In the final section we provide a summary of the current work and present some future research directions.

2 Related work

The general facility location problem involves two decisions, a location decision to determine where a facility should be set up, and an allocation decision to determine which customers will be served by a particular facility. Opening of new facilities involves time and capital investment, and it is one of the most important decisions for any institution. Facility location models have been extensively used by the World Bank and various government projects (Antunes & Peeters, 2000, 2001; Brotcorne et al., 2003; Ghaderi & Jabalameli, 2012).

Depending on number of periods in the planning horizon in which location and allocation decisions are to be made, facility location problems can be categorized as single-period or multi-period. Most of the early work was done for the single period case (Drezner & Hamacher, 2001). Subsequently work has focused on the multi-period location problems.

Multi-period facility location problems are important because of two reasons. First, customer demands, transportation/assignment costs and other parameters change over time. Secondly, relocation of facilities involve capital expenditure. In absence of the first characteristic, a single period model can be used to solve the problem and, in the absence of second characteristic a series of disconnected static formulations can be used (Erlenkotter, 1981). Furthermore, in the presence of budget constraints all facilities cannot be opened at once and thus the multi-period problem becomes important.

Multi-period facility location problems have been widely studied after the initial works by Warszawski (1973); Erlenkotter (1981); Van Roy & Erlenkotter (1982). Problems under various constraints have been looked at. Some example of these variants are constraints on location and relocation of facilities (Wesolowsky & Truscott, 1975; Melo et al., 2006), reopening cost different than first time opening cost (Dias et al., 2006, 2007, 2008), and non-zero closing cost (Wesolowsky & Truscott, 1975; Saldanha da Gama & Captivo, 1998; Canel et al., 2001). Arabani & Farahani (2011) provide a survey of the literature in multi-period facility location problem.

The literature on facility location also considers uncertainty in decision making environment. Most of the references in this literature consider uncertainty or incomplete information on the demand at the nodes (Killmer et al., 2001; Averbakh, 2003; Albareda-Sambola et al., 2011; Berman & Wang, 2011). Current et al. (1998) used the minimax regret criteria to decide where to set up the initial set of facilities when the total number of facilities to be set up in the future is uncertain. The problem was presented in the context of p-median problem.

UPHCLP aims to provide maximal population coverage by the network of PHCs and is closely related to the maximal covering location problem (MCLP) with complete and partial/gradual coverage. We present a brief review of literature on these problems.

2.1 Uncapacitated maximal covering location problem with complete coverage

MCLP was introduced by Church & ReVelle (1974). In the MCLP we are given a set I of demand nodes, a set J of candidate locations for facilities, a matrix $[\delta_{ij}]$ of distance between each $i \in I$ and each $j \in J$, a vector (d_i) of demand at each demand node, and a covering distance δ_0. This covering distance is fixed *a priori*, and the facilities can serve only those demand nodes that fall within the covering distance. Let $N_i = \{j \in J : \delta_{ij} \leq \delta_0\}$ be the set of candidate facilities which lie within the covering distance from the demand node i. Each facility has an infinite capacity. We are required to locate p facilities so that the maximum demand at the demand nodes can be met by the facilities. The variables are: $y_j = 1$ if a facility is opened at location j, and 0 otherwise; $x_i = 1$ if the demand node i is within the covering distance of some opened facility, and 0 otherwise. The problem can be formulated as follows:

$$\text{MCLP:} \quad \text{Maximize} \sum_{i \in I} d_i x_i \tag{1}$$

$$s.t. \quad x_i \leq \sum_{j \in N_i} y_j \qquad \forall i \in I \tag{2}$$

$$\sum_{j \in J} y_j = p \tag{3}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \tag{4}$$

$$0 \leq x_i \leq 1 \qquad \forall i \in I \tag{5}$$

The objective in the above formulation maximizes the total demand covered. Constraint set (2) ensures that any demand node will be covered only if a facility within the covering distance from that demand node is opened. Constraint (3) limits the number of facilities to be opened. In the above formulation x_i need not be declared an integer. As y_j are integers and x_i are constrained only by the first constraint in the formulation, x_i will be integers too.

Chung (1986); Schilling et al. (1993) have discussed various applications of the MCLP. Daskin (2000) used MCLP to solve the p-center problem to optimality. An MCLP is solved with different covering distances. When all the demand is met at some least possible covering distance, the p-center problem is solved to optimality. Schilling et al. (1993) reviewed papers on covering problems. Most of the studies on MCLP focus on improving reliability of the system by providing multiple coverage with applications in emergency services like ambulance, fire stations and blood banks (Schilling et al., 1993).

MCLP are NP-Hard but they are known to yield integer friendly solutions. Church & Meadows (1979) used combination of linear programming and branch-and-bound to solve the MCLP. Details on the work and solution methods for MCLP can be seen from the review papers by Schilling et al. (1993); Farahani et al. (2012).

Gunawardane (1982) first introduced the maximal coverage location problem in a dynamic scenario. In the problem setup, location of the facilities and the possible relocation within the planning horizon was considered. The problem is an extension of MCLP, and we use the earlier notation with superscript t to indicate the value of parameters at time period t. The variables in this formulation are: $y_j^t = 1$ if a facility is open at location j at time period t, and 0 otherwise; $x_i^t = 1$ if the demand node i is within the covering distance of some opened facility at time period t, and 0 otherwise. The problem can be formulated as follows:

$$\text{MMCLP:} \quad \text{Maximize } \zeta = \sum_{t \in T} \sum_{i \in I} d_i^t x_i^t \tag{6}$$

$$s.t. \quad x_i^t \le \sum_{j \in N_i} y_j^t \qquad \forall i \in I, t \in T \tag{7}$$

$$y_j^t \ge y_j^{t-1} \qquad \forall j \in J, t > 1 \tag{8}$$

$$\sum_{j \in J} y_j^t = \sum_{t' \le t} p^{t'} \qquad \forall t \in T \tag{9}$$

$$y_j^t \in \{0, 1\} \qquad \forall j \in J, t \in T \tag{10}$$

$$0 \le x_i^t \le 1 \qquad \forall i \in I, t \in T \tag{11}$$

The objective in the above formulation maximizes the total demand covered in the planning horizon. Constraint set (7) ensures that any demand node will be covered only if a facility within the covering distance from that demand node is open. Constraint set (8) guarantees that facility once opened will not be closed in the planning horizon. Constraint set (9) limits the number of facilities to be opened in any period. In the above formulation the variable x_i^t need not be declared as integer. As x_i^t are constrained only by the first constraint in the formulation and y_j^t are integers, hence x_i^t will be integers too.

Gunawardane (1982) solved small problems of the size 10 to 30 demand nodes and 5 periods using the standard solvers. In most of the instances, the LP relaxation yielded integer solutions. Zarandi et al. (2013) is the only other study dealing with the multi-period MCLP. They used simulated annealing to solve problems of size 2500 demand nodes and 200 potential candidate locations.

2.2 Uncapacitated maximal covering location problem with gradual coverage

In many real life situations it is not possible to determine what is the exact coverage distance, or to suggest that demand nodes falling just within the covering distance from a facility are covered, while those just outside the covering distance are not covered. In those situations partial/gradual coverage function has been used by Church & Roberts (1983); Berman et al. (2003); Karasakal & Karasakal (2004); Berman et al. (2010). Let δ_{ij} be the distance between the demand node i and the candidate facility location j. The demand node i is fully covered by the facility j when δ_{ij} falls within a specified minimum covering distance δ_{min}, whereas the demand node is not covered when it lies beyond a specified maximum covering distance δ_{max}. If $\delta_{min} < \delta_{ij} < \delta_{max}$, it is considered partially covered, and the coverage can be a linear, step or any other function of the distance between node i and node j. Let $a_{ij} \in [0, 1]$ be the level of coverage provided by the facility at j to the demand node i.

$$a_{ij} = \begin{cases} 1 & \text{if } \delta_{ij} \le \delta_{min}, \\ f(\delta_{ij}) & \text{if } \delta_{min} < \delta_{ij} < \delta_{max}, \text{ where } 0 < f(\delta_{ij}) < 1, \\ 0 & \text{otherwise} \end{cases}$$

Let $x_{ij} = 1$ if the demand node i is either partially or fully covered by the facility at j, and 0 otherwise. Let N_i be the set of facility sites which cover the demand node i either fully or partially. $y_j = 1$ if a facility is opened at the site j, and 0 otherwise. Karasakal & Karasakal (2004) give the formulation of MCLP with partial coverage as follows:

$$\text{MCLP-P: maximize } \zeta = \sum_i \sum_{j \in N_i} a_{ij} d_i x_{ij} \tag{12}$$

$$s.t. \quad \sum_j y_j = p \tag{13}$$

$$x_{ij} \le y_j \qquad \forall i, j \in N_i \tag{14}$$

$$\sum_{j \in N_i} x_{ij} \le 1 \qquad \forall i \tag{15}$$

$$y_j \in \{0, 1\} \qquad \forall j \tag{16}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N_i \tag{17}$$

The objective in the above formulation maximizes the total demand covered. Constraint (13) limits the number of facilities to be opened. Constraint set (14) ensures that a demand node is assigned to a facility only if the facility is open. Constraint set (15) guarantees that each demand node is assigned to a single facility.

Berman et al. (2010) provide an excellent review of all the work with gradual coverage. MCLP is a special case of MCLP-P when $\delta_{min} = \delta_{max} = \delta_0$; $a_{ij} = 1$ if $\delta_{ij} \le \delta_0$ and 0 otherwise. When the decay of coverage is linear between the distance of δ_{min} and δ_{max}, $a_{ij} = (\delta_{max} - \delta_{ij})/(\delta_{max} - \delta_{min})$. Berman et al. (2010) also show that MCLP-P with a linear decay function generalizes the classical p-median problem. Berman & Wang (2011) studied the problem when demand at the nodes are random variables whose probability distributions are unknown. They used the information on the range of these variables to find the minimax regret location that minimizes the worst-case coverage loss.

A lot of work has been done on problems with minisum objectives and there are many efficient algorithms to solve even the large instances of such problems. However, the problems with minimax structure are relatively difficult to solve. For an example the p-center problem are generally solved by repeatedly solving set covering problems, which are themselves NP hard.

To the best of our knowledge no work has been done that takes into account the uncertainty in number of servers, i.e. in which the number of facilities that will be opened in any period is not known beforehand. In many practical problems this uncertainty exists. For example in the UPHCLP, physical infrastructure for the PHC exist at many locations but due to unavailability of doctors, complete service cannot be provided by these PHCs. Those PHCs where doctors (servers) are not present can only provide very limited healthcare facilities. The number and period of doctors joining rural healthcare in any district is not known *a priori*. When a doctor joins, they are allocated to an unmanned PHCs. The sequence of opening new facilities, i.e. allocating a doctor to a site where physical infrastructure already exists, is pre-decided for a planning horizon. The above mentioned uncertainty in period and number of doctors joining, may lead to such a facility opening sequence being considered, which may be highly suboptimal considering the optimal *ex post* decision, i.e. the optimal decision.

In this work, we are given a set of demand nodes and a set of candidate facility locations. Each facility has infinite capacity however, it can provide complete or partial coverage only to the demand nodes that fall within a specified maximum covering distance. The number of facilities to be opened in various periods of planning horizon is uncertain and gives rise to different scenarios. The objective of the problem is to determine the sequence of opening facilities (staffing PHCs with doctors) that creates the least worst case deviation from the optimal over all scenarios. This minimizes the maximum regret across scenarios.

3 Problem formulation

When the number of facilities that will be opened in all periods of the planning horizon is known, the facility location problem reduces to MMCLP (complete or gradual coverage). However, when there is uncertainty in the numbers of facilities that can be opened in each period of the planning horizon, the objective changes to minimizing the worst case regret from a solution. A solution to this problem is the facility opening sequence, which needs to be pre-decided for the planning horizon. All possible scenarios through which servers will become available over the planning horizon need to be considered. The regret associated with any solution, for a given scenario is the difference between the *ex post* optimal coverage for that scenario and the coverage obtained with the solution.

For an example consider only three scenarios exist and there are two possible sequences. The demand coverage matrix for both the facility opening sequences under all the scenarios is given in Table 1.

Table 1: A sample coverage matrix

	Sequence 1	Sequence 2	Maximum
Scenario 1	100	180	180
Scenario 2	200	150	200
Scenario 3	50	70	70

A regret matrix as given in Table 2 can be obtained with the coverage matrix of Table 1. In this example, the sequence 2 gives the least worst case regret. Hence, if the objective is to minimize the worst case regret, sequence 2 will be selected.

Table 2: A sample regret matrix

	Sequence 1	Sequence 2
Scenario 1	80	0
Scenario 2	0	50
Scenario 3	20	0
Maximum regret	80	50

We model the problem as follows. There is a set of demand nodes $I = \{1, 2, .., m\}$ and a set of location $J = \{1, 2, .., n\}$ where facilities are to be opened during the planning horizon. Other than a set J of locations where facilities are yet to be opened, we are also given a set J_b of locations where facilities are open before the start of the planning horizon. Let $a_{ij} \in [0, 1]$ be the level of coverage provided by the facility at j to the demand node i. The demand at node i in period t is given by d_i^t.

Let $s \in S$ be the index that represents the future scenario in which new servers will be available at each period of the planning horizon. A scenario $(a_1, a_2, ..., a_{|T|})$ indicates that in the first period, a_1 new servers will be available (a_1 facilities can be opened), a_2 new servers in the second period, likewise and in the last period $a_{|T|}$ new servers will be available. As the problem requires finding the sequence in which n facilities shall be opened, scenarios are generated so as all n facilities are opened by the end of the last period. This is equivalent to generation of all possible scenarios such that between 0 to n facilities are opened by the first $|T| - 1$ periods. This adequately captures the uncertainty in the availability of the servers. One of the concerns while generating scenarios is that some unrealistic scenarios might influence the solution. However, in this problem we find that many such extreme scenarios do not determine the optimal sequence, and we give some scenario dominance rules later in Section 4. To better understand how the scenario are generated consider an example with $n = 2$ and $|T| = 3$. All possible scenarios for this problem size are $(0, 0, 2), (0, 1, 1), (0, 2, 0), (1, 0, 1), (1, 1, 0)$ and $(2, 0, 0)$. With n candidate facilities to be opened in $|T|$ periods, number of scenarios of getting new servers over the planning horizon is $\binom{n+|T|-1}{n} = \frac{(n+|T|-1)!}{n!(|T|-1)!}$. Notice that by the last period all the facilities have been opened, consequently, demand coverage in the last period will be the same for all the scenarios. Hence, we need to consider only the first $|T| - 1$ periods to capture all the uncertainty. We represent the first $|T| - 1$ periods as a set $T_0 = T \backslash \{|T|\}$.

To represent the facility opening sequence, we introduce a matrix $R = [r_{jj'}]$, where $r_{jj'}$ takes a value of 1 if the facility at j has a rank $j' \in \{1, 2, ..., n\}$ of opening in a facility opening sequence, and 0 otherwise. A subscript s is introduced in the parameters to represent the values with the scenario $s \in S$. $y_{js}^t = 1$ if a facility is open at the location j at the time period t when the facility opening scenario is s, and 0 otherwise. $x_{is}^t = 1$ if the demand node i is within the covering distance of some opened facility at the time period t and scenario s, and 0 otherwise. To establish a relation between the variables $r_{jj'}$ and y_{js}^t we can argue that a facility at j is open at any given period, if and only if the rank of facility at j in the facility opening sequence is not more than the total number of facilities opened till that period. Mathematically,

$$y_{js}^t = 1 \iff \sum_{j'} j' r_{jj'} \leq \sum_{t' \leq t} p_s^{t'}$$

$$\text{i.e.,} \quad y_{js}^t = 1 \implies \sum_{j'} j' r_{jj'} \le \sum_{t' \le t} p_s^{t'} \quad \text{and} \quad y_{js}^t = 0 \implies \sum_{j'} j' r_{jj'} \ge \sum_{t' \le t} p_s^{t'} + 1$$

This translates into two big-M type of constraints. To have tight constraints, the big-M values must be as low as possible however, it should be sufficiently large for the problem. Hence, for the two constraints introduced, we use the big-M values as:

$$max(\sum_{j'} j' r_{jj'} - \sum_{t' \le t} p_s^{t'}) = n - 0 = n \ \text{ and } max(\sum_{t' \le t} p_s^{t'} + 1 - \sum_{j'} j' r_{jj'}) = n + 1 - 1 = n \ \text{ respectively}$$

Therefore the constraints are:

$$\sum_{j'} j' r_{jj'} \le \sum_{t' \le t} p_s^{t'} + n(1 - y_{js}^t) \qquad \forall j \in J, t \in T_0, s \in S \tag{18}$$

$$\sum_{j'} j' r_{jj'} \ge \sum_{t' \le t} p_s^{t'} + 1 - n y_{js}^t \qquad \forall j \in J, t \in T_0, s \in S \tag{19}$$

Proposition 1: Constraint set (19) will not be binding at optimality.

Proof. The above two constraints give a bound of y_{js}^t as:

$$(\sum_{t' \le t} p_s^{t'} + 1 - \sum_{j'} j' r_{jj'})/n \le y_{js}^t \le (\sum_{t' \le t} p_s^{t'} + n - \sum_{j'} j' r_{jj'})/n$$

Where, $\sum_{j'} j' r_{jj'}$ is the rank of facility j in a facility opening sequence. The objective ensures that maximum coverage is attained for any given facility opening sequence and scenario. Coverage of any demand node is constrained only by y_{js}^t as can be seen from the constraint set (7) of the MMCLP formulation. Hence, only the upper bound of y_{js}^t will be binding at optimality as the objective guarantees that for any given facility opening sequence and scenario, y_{js}^t attains the maximum possible value. □

Proposition 2: Constraint set (18) does not make the constraint set $\sum_{j} y_{js}^t = \sum_{t' \le t} p_s^{t'} \quad \forall t \in T_0, s \in S$ redundant.

Proof. Constraint set (18) gives a bound of y_{js}^t as:

$$\begin{aligned}
& y_{js}^t \le (\sum_{t' \le t} p_s^{t'} + n - \sum_{j'} j' r_{jj'})/n && \forall j, t, s \\
\implies & \sum_{j} y_{js}^t \le \sum_{j} (\sum_{t' \le t} p_s^{t'} + n - \sum_{j'} j' r_{jj'})/n && \forall t, s \\
& = \sum_{t' \le t} p_s^{t'} + n - \sum_{j} \sum_{j'} j' r_{jj'}/n \\
& = \sum_{t' \le t} p_s^{t'} + n - \sum_{j} (1 r_{j1} + 2 r_{j2} + \ldots + n r_{jn})/n
\end{aligned}$$

$$= \sum_{t' \leq t} p_s^{t'} + n - 1/n\left(\sum_j r_{j1} + 2\sum_j r_{j2} + ... + n\sum_j r_{jn}\right)$$
$$= \sum_{t' \leq t} p_s^{t'} + n - 1/n(1 + 2 + ... + n) \quad [\because \sum_j r_{jj'} = 1 \ \ \forall j']$$
$$= \sum_{t' \leq t} p_s^{t'} + n - (n+1)/2 = \sum_{t' \leq t} p_s^{t'} + (n-1)/2$$

Number of candidate facilities $n > 1$, hence constraint set $\sum_j y_{js}^t = \sum_{t' \leq t} p^{t'_s} \quad \forall t \in T_0, s \in S$ gives a tighter upper bound of $\sum_j y_{js}^t$. □

3.1 Uncapacitated problem with complete coverage

Let Π be the set of all possible facility opening sequence and Π_k be one of the facility opening sequences. Let the total demand coverage with Π_k in scenario $s \in S$ by the end of period $|T_0|$ be $\zeta_{\Pi_k,s}$. The maximal demand coverage that can be achieved with scenario $s \in S$ is $\zeta_s^* = \max_{\Pi_k \in \Pi} \zeta_{\Pi_k,s}$, which is obtained by solving the MMCLP with $t \in T_0$ and server availability given by scenario s. Here $a_{ij} = 1$ if the demand node i is within the covering distance from the facility at j, and 0 otherwise. The problem that minimizes the worst case regret can be formulated as:

$$\min_{\Pi_k \in \Pi} \max_{s \in S} (\zeta_s^* - \sum_{t \in T_0} \sum_{i \in I} d_i^t x_{is}^t) \tag{20}$$
$$s.t. \quad x_{is}^t \leq \sum_{j \in J} a_{ij} y_{js}^t + \sum_{j \in J_b} a_{ij} \qquad \forall i \in I, t \in T_0, s \in S \tag{21}$$
$$\sum_{j \in J} y_{js}^t = \sum_{t' \leq t} p_s^{t'} \qquad \forall t \in T_0, s \in S \tag{22}$$
$$\sum_{j'} r_{jj'} = 1 \qquad \forall j \in J \tag{23}$$
$$\sum_{j \in J} r_{jj'} = 1 \qquad \forall j' \in \{1, 2, ..., n\} \tag{24}$$
$$\sum_{j'} j' r_{jj'} \leq \sum_{t' \leq t} p_s^{t'} + n(1 - y_{js}^t) \qquad \forall j \in J, t \in T_0, s \in S \tag{25}$$
$$0 \leq x_{is}^t \leq 1 \qquad \forall i \in I, t \in T_0, s \in S \tag{26}$$
$$y_{js}^t = 1 \qquad \forall j \in J_b, t \in T_0, s \in S \tag{27}$$
$$y_{js}^t \in \{0, 1\} \qquad \forall j \in J, t \in T_0, s \in S \tag{28}$$
$$r_{jj'} \in \{0, 1\} \qquad \forall j \in J, j' \in \{1, 2, ..., n\} \tag{29}$$

In the above formulation constraint set (21) and (22) come from the MMCLP formulation. Constraint set (23) and (24) necessitate that all the facilities get a unique rank in a facility opening sequence. Constraint set (25) relates the variables $r_{jj'}$ and y_{js}^t as explained earlier. The constraint set (8) of the MMCLP formulation is not included in this formulation as it is redundant when the facilities are opened with a pre-defined sequence. Here, the objective is nonlinear, which can be linearized using the standard linearization technique. The resulting linear model for multi-period uncapacitated location problem with server uncertainty (MULPSU) can be formulated as follows:

$$\text{MULPSU:} \quad \text{Minimize} \ \ \theta \tag{30}$$

$$s.t. \quad (21) - (29)$$

$$\theta \geq \zeta_s^* - \sum_{t \in T_0} \sum_{i \in I} d_i^t x_{is}^t \qquad \forall s \in S \tag{31}$$

$$\theta \geq 0 \tag{32}$$

In the above formulation constraint set (31) has been used for making the model linear.

Proposition 3: Assuming the set of locations where facilities are open before the start of the planning horizon $J_b = \phi$, and removing the set of demand nodes which are within the covering distance of facilities in J_b, will not change the solution of MULPSU.

Proof. Let $i \in I_b$ represent the demand nodes which are covered by the facilities opened before the start of the planning horizon. Now, $x_{is}^t = 1 \quad \forall i \in I_b, t \in T_0, s \in S$, hence we have:

$$\zeta_s = \sum_{t \in T_0} \sum_{i \in I} d_i^t x_{is}^t = \sum_{t \in T_0} \sum_{i \in I \setminus I_b} d_i^t x_{is}^t + \sum_{t \in T_0} \sum_{i \in I_b} d_i^t x_{is}^t = \sum_{t \in T_0} \sum_{i \in I \setminus I_b} d_i^t x_{is}^t + \sum_{t \in T_0} \sum_{i \in I_b} d_i^t$$

The second term is a constant, consequently

$$\theta = \max_s (\zeta_s^* - \zeta_s) = \max_s \big(\sum_{t \in T_0} \sum_{i \in I \setminus I_b} d_i^t x_{is}^{t*} - \sum_{t \in T_0} \sum_{i \in I \setminus I_b} d_i^t x_{is}^t \big)$$

Therefore we can remove the set J_b and I_b to solve the MULPSU. □

3.2 Uncapacitated location problem with gradual coverage

This problem generalizes the MULPSU. The demand nodes which lie within a minimum covering distance δ_{min} and a maximum covering distance δ_{max} from a facility at j are considered partially covered by the facility. Here $a_{ij} \in [0, 1]$, which is the level of coverage provided by the facility at j to the demand node i, can have a fractional value. The problem which minimizes the worst case regret can be formulated as:

$$\text{MULPSU-P:} \quad \text{Minimize} \ \ \theta \tag{33}$$

$$s.t. \quad \sum_{j \in J} y_{js}^t = \sum_{t' \leq t} p_s^{t'} \qquad \forall t \in T_0, s \in S \tag{34}$$

$$x_{ijs}^t \leq y_{js}^t \qquad \forall i \in I, j \in N_i \cap J, \forall t \in T_0, s \in S \tag{35}$$

$$\sum_{j \in N_i} x_{ijs}^t \leq 1 \qquad \forall i \in I, t \in T_0, s \in S \tag{36}$$

$$\theta \geq \zeta_s^* - \sum_{i \in I} \sum_{j \in N_i} \sum_{t \in T_0} a_{ij} d_i^t x_{ijs}^t \qquad \forall s \in S \tag{37}$$

$$\sum_{j'} r_{jj'} = 1 \qquad \forall j \in J \tag{38}$$

$$\sum_{j \in J} r_{jj'} = 1 \qquad \forall j' \in \{1, 2, ..., n\} \tag{39}$$

$$\sum_{j'} j' r_{jj'} \leq \sum_{t' \leq t} p_s^{t'} + n(1 - y_{js}^t) \qquad \forall j \in J, t \in T_0, s \in S \tag{40}$$

$$x_{ijs}^t \in \{0, 1\} \qquad \forall i \in I, j \in N_i, t \in T_0, s \in S \tag{41}$$

$$\theta \geq 0 \tag{42}$$

$$y_{js}^t \in \{0, 1\} \qquad \forall j \in J, t \in T_0, s \in S \tag{43}$$

$$r_{jj'} \in \{0, 1\} \qquad \forall j \in J, j' \in \{1, 2, ..., n\} \tag{44}$$

In the above formulation the objective is to minimize across all facility opening sequence, the maximum regret θ associated with any scenario. Constraint (34), (35) and (36) come from the MMCLP-P formulation. Constraint (37) ensures that θ takes the highest value of regret across scenarios. Constraint set (38) and (39) necessitate that all the facilities get a unique rank in a facility opening sequence. Constraint set (40) relates the variables $r_{j'j}$ and y_{js}^t as explained earlier as in the MULPSU formulation.

Proposition 4: The variables x_{ijs}^t can be relaxed as continuous variable in the MULPSU-P formulation.

Proof. Let an optimal solution of MULPSU-P has some x_{ijs}^t as fraction, for a scenario s and time t. Now $\sum_{j \in N_i} x_{ijs}^t \leq 1$ and the objective ensures that the variables x_{ijs}^t take the maximum possible values. This means that for the scenario s and time t, more than one facilities have been allotted to the demand node i. This indicates that either all of those facilities fall within δ_{min}, or are equidistant from the node i. As the facilities do not have capacity limits, making $x_{ijs}^t = 1$ for one of those facilities and zero for others will not make the problem infeasible and will result in the same objective function value. Hence we can construct a feasible solution which has the same objective function value, from the fractional solution of variable x_{ijs}^t. □

Notice that proposition 3 which was valid for MULPSU, does not hold for the MULPSU-P. Hence to solve the MULPSU-P, we cannot ignore the set of facilities which are open before the start of planning horizon. This is because the facilities which are open before the start of planning horizon might provide partial coverage to some demand nodes. Those demand nodes which are only partially covered cannot be removed from the problem as indicated in proposition 3.

4 Solution methodology

It is important to note at the onset that a greedy or *ad hoc* method will not give the optimal solution in most problem instances because of the following proposition:

Proposition 5: A facility opening sequence $\Pi_1 = (\pi_1, \pi_2, ..., \pi_n)$ will give zero regret for all scenarios i.e. will be the best sequence irrespective of the scenario if and only if all the following conditions hold true:

$$\begin{aligned} \bar{d}^t_{\pi_1} &\geq \bar{d}^t_{\pi'_1} && \forall t \in T_0, \pi'_1 \in J : \pi'_1 \neq \pi_1 \\ \bar{d}_{\pi_1 \cup \pi_2} &\geq \bar{d}_{\pi'_1 \cup \pi'_2} && \forall t \in T_0, (\pi'_1, \pi'_2) \in J : \{\pi'_1, \pi'_2\} \neq \{\pi_1, \pi_2\} \\ &\vdots \\ \bar{d}^t_{\pi_1 \cup \pi_2 \cup ... \cup \pi_{n-1}} &\geq \bar{d}^t_{\pi'_1 \cup \pi'_2 \cup ... \cup \pi'_{n-1}} && \forall t \in T_0, (\pi'_1, \pi'_2, ..., \pi'_{n-1}) \in J : \{\pi'_1, \pi'_2, ..., \pi'_{n-1}\} \neq \{\pi_1, \pi_2, ..., \pi_{n-1}\} \end{aligned}$$

Where, $\bar{d}^t_{\pi_1 \cup \pi_2 \cup ... \cup \pi_i}$ represents the total demand covered by the set of facilities $\{\pi_1, \pi_2, ..., \pi_i\}$.

Proof. We first prove by contradiction that if all the above conditions are met, the facility opening sequence $\Pi_1 = (\pi_1, \pi_2, ..., \pi_n)$ will give zero regret i.e. the highest coverage for all scenarios. Let there be $s = (a_1, a_2, .., a_t, .., a_{|T|}) \in S$, for which another sequence $\Pi_2 = (\pi'_1, \pi'_2, ..., \pi'_n)$ gives more demand coverage even though all the above conditions are met. Then there must be atleast one period for which demand covered in that period with Π_1 is less than the demand coverage with Π_2, i.e.

$$\exists t \in T : \bar{d}^t_{\pi_1 \cup \pi_2 \cup ... first\ (a_1+a_2+..a_t)\ facilities\ in\ \Pi_1} < \bar{d}^t_{\pi'_1 \cup \pi'_2 \cup ... first\ (a_1+a_2+..a_t)\ facilities\ in\ \Pi_2}$$

This contradicts the assumption that all the relations in the proposition are satisfied.

We now prove that if any of the above conditions are not satisfied, facility opening sequence Π_1 will not give zero regret i.e. the highest coverage for all the scenarios. Let us assume that there exists $t \in T_0, l \in \{1, 2.., n-1\} : \bar{d}^t_{\pi_1 \cup \pi_2 \cup ... \cup \pi_l} < \bar{d}^t_{\pi'_1 \cup \pi'_2 \cup ... \cup \pi'_l}$. If we can find a scenario in which Π_2 gives higher coverage than Π_1 our proof is done. Note that for the scenario when l new servers are available in period t and rest of the $n - l$ servers become available in the last period, Π_2 gives a higher coverage. Hence for any facility to give zero regret in all the scenarios, all the above conditions need to be satisfied. □

The numbers of possible facility opening sequences when n is 5, 10 and 15 are 120, 3628800, and 1307674368000 respectively. Moreover, the number of server availability scenarios are 126, 1001 and 3876 for $|T| = 5$ and $n = 5$, 10 and 15 respectively. For any given scenario the problem reduces to a MMCLP, which is the multi-period version of the MCLP. MMCLP is NP-hard since its single period version, MCLP is known to be NP-Hard (Drezner & Hamacher, 2001). As MULPSU requires solving an NP hard problem multiple times, exorbitant computational effort will be needed to solve the problem to optimality.

4.1 Dominance rules

Here we present some dominance rules to reduce the search space. We provide the formal proofs of these rules in Appendix A.

Proposition 6: Let $\Pi_1 = (\pi_1, \pi_2, .., \pi_{i-1}, \pi_i, ..\pi_j, \pi_{j+1}..\pi_n)$ be a sequence in which facilities are to be opened, and let $\Pi_2 = (\pi_1, \pi_2, .., \pi_{i-1}, \pi_j, ..\pi_i, \pi_{j+1}..\pi_n)$ be a sequence obtained by switching the positions of π_i and π_j in Π_1. Let $\bar{d}^t_{\pi_1 \cup \pi_2 \cup ... \cup \pi_i}$ represent the total demand covered by the set of facilities $(\pi_1, \pi_2, ..., \pi_i)$.

If all the inequalities below are satisfied, facility opening sequence Π_1 will dominate Π_2, or in other words, regret associated with Π_1 will be not be higher than that associated with Π_2 for any corresponding scenario.

$$\begin{aligned}
\bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_i} &\geq \bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_j} && \forall t \in T_0\\
\bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_i\cup\pi_{i+1}} &\geq \bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_j\cup\pi_{i+1}} && \forall t \in T_0\\
&\vdots\\
\bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_i\cup\pi_{i+1}\cup\ldots\cup\pi_{j-1}} &\geq \bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_j\cup\pi_{i+1}\cup\ldots\cup\pi_{j-1}} && \forall t \in T_0
\end{aligned}$$

The above set of inequalities in general form can be written as:

$$\bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_i} \geq \bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_j} \qquad \forall t \in T_0 \tag{45}$$

$$\bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_i\ldots\cup\pi_l} \geq \bar{d}^t_{\pi_1\cup\pi_2\cup\ldots\cup\pi_j\ldots\cup\pi_l} \qquad \forall t \in T_0, l \in \{i+1, i+2, ..., j-1\} \tag{46}$$

Notice that in both Π_1 and Π_2, first $i-1$ facilities are the same. Moreover, if j or more facilities are open, the same set of facilities in both Π_1 and Π_2 will be open. Hence, the above proposition does not include these conditions as they are trivial equality relations.

Computation of left hand and right hand side in each inequality of the above proposition takes $O(nm)$ time. Moreover, there are $O(n)$ such inequalities for each $t \in T_0$. Therefore, the dominance calculation will take $O(n^2m|T_0|)$ time for each sequence. In comparison, if a sequence is found to be dominated the time saved will be $O(nm|T_0||S|)$, which will be $O(n^5m|T_0|)$ when $|T|$ is 5 periods i.e. $|T_0| = 4$.

Proposition 7: Those server availability scenarios where all n servers become available in the same period will have zero regret for any facility opening sequence.

Proposition 8a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ...\pi_n)$, the regret associated with the server availability scenario $s_1 = (0, a_2, a_3, .., a_{|T|}) \in S$ will not be greater than that associated with the scenario $s_2 = (1, a_2 - 1, a_3, .., a_{|T|}) \in S$ if:

$$\bar{d}^1_{\pi_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{47}$$

where, $a_2 > 0$ and $\bar{d}^1_{\pi_1}$ is the demand coverage by π_1 in the first period of the planning horizon.

Proposition 8b: When $a_2 = 0$ in scenario $s_1 = (0, a_2, a_3, .., a_{|T|})$ and i be the first period with a non-zero a_i. For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, .., \pi_n)$, regret associated with the scenario $s_1 = (0, .., a_i, .., a_{|T|})$ will not be greater than that associated with the scenario $s_2 = (1, .., a_i - 1, .., a_{|T|})$ if:

$$\bar{d}^1_{\pi_1} + \bar{d}^2_{\pi_1} + .. + \bar{d}^{i-1}_{\pi_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{48}$$

Proposition 9a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret associated with the scenario $s_1 = (a_1, a_2, .., a_{|T|-1}, 0)$ will not be greater than that associated with the scenario $s_2 = (a_1, a_2, .., a_{|T|-1} - 1, 1)$

if:

$$\bar{d}_J^{|T|-1} - \bar{d}_{J\setminus\pi_n}^{|T|-1} \geq \zeta_{s_1}^* - \zeta_{s_2}^* \tag{49}$$

where, $a_{|T|-1} > 0$.

Proposition 9b: When $a_{|T|-1} = 0$ in scenario $s_1 = (a_1, .., a_{|T|-1}, 0)$ and i be the last period with a non-zero a_i. For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret associated with the scenario $s_1 = (a_1, .., a_i, ., 0)$ will not be greater than that associated with the scenario $s_2(a_1, .., a_i - 1, .., 1)$ if:

$$(\bar{d}_J^{|T|-1} - \bar{d}_{J\setminus\pi_n}^{|T|-1}) + (\bar{d}_J^{|T|-2} - \bar{d}_{J\setminus\pi_n}^{|T|-2}) + ... + (\bar{d}_J^{i} - \bar{d}_{J\setminus\pi_n}^{i}) \geq \zeta_{s_1}^* - \zeta_{s_2}^* \tag{50}$$

All possible facility opening sequence need not be checked to establish the dominance of the scenarios. One of the n facilities will be the last facility or first facility and thus checking for these n facilities in the the left hand side of inequality in proposition 8 or 9 will establish the dominance. This dominance calculation will take $O(n^2m|T|)$ time for each scenario and if a scenario is found to be dominated the time saved will be $O(n^3m|T||Iter.|)$, where $|Iter.|)$ is the total number of iterations in the neighborhood search. Number of scenarios that have no new servers available in first or the last period is $2 \times \binom{n+|T|-2}{n} - \binom{n+|T|-3}{n}$. For example, with $n = 10$ and $|T| = 5$, number of such scenarios is 506 (out of 1001 total possible scenarios).

Proposition 10a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret associated with the scenario $s_1 = (1, a_2, a_3, .., a_{|T|})$ will not be greater than that associated with the scenario $s_2 = (2, a_2 - 1, a_3, .., a_{|T|})$ if:

$$\bar{d}_{\pi_1\cup\pi_2}^{1} - \bar{d}_{\pi_1}^{1} \leq \zeta_{s_2}^* - \zeta_{s_1}^* \tag{51}$$

where, $a_2 > 0$.

Proposition 10b: When $a_2 = 0$ in scenario $(1, a_2, a_3, .., a_{|T|})$ and i be the first period with a non-zero a_i (other than period 1). For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret associated with the scenario $s_1 = (1, .., a_i, .., a_{|T|})$ will not be greater than that associated with the scenario $s_2 = (2, .., a_i - 1, .., a_{|T|})$ if:

$$(\bar{d}_{\pi_1\cup\pi_2}^{1} - \bar{d}_{\pi_1}^{1}) + (\bar{d}_{\pi_1\cup\pi_2}^{2} - \bar{d}_{\pi_1}^{2}) + .. + (\bar{d}_{\pi_1\cup\pi_2}^{i-1} - \bar{d}_{\pi_1}^{i-1}) \leq \zeta_{s_2}^* - \zeta_{s_1}^* \tag{52}$$

Proposition 11a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret in the scenario $s_1 = (a_1, a_2, .., a_{|T|-1}, 1)$ will not be greater than that associated with the scenario $s_2 = (a_1, a_2, .., a_{|T|-1} - 1, 2)$ if:

$$\bar{d}_{J\setminus\pi_n}^{|T|-1} - \bar{d}_{J\setminus(\pi_n,\pi_{n-1})}^{|T|-1} \geq \zeta_{s_1}^* - \zeta_{s_2}^* \tag{53}$$

where, $a_{|T|-1} > 0$.

Proposition 11b: When $a_{|T|-1} = 0$ in scenario $(a_1, .., a_{|T|-1}, 1)$ and i be the last period with a non-zero a_i(other than period $|T|$). For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret in the scenario $s_1 = (a_1, .., a_i, .., 1)$ will not be greater than that in scenario $s_2 = (a_1, .., a_i - 1, .., 2)$ if:

$$(\bar{d}^{|T|-1}_{J\setminus\pi_n} - \bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})}) + (\bar{d}^{|T|-2}_{J\setminus\pi_n} - \bar{d}^{|T|-2}_{J\setminus(\pi_n,\pi_{n-1})}) + ... + (\bar{d}^{i}_{J\setminus\pi_n} - \bar{d}^{i}_{J\setminus(\pi_n,\pi_{n-1})}) \geq \zeta^*_{s_1} - \zeta^*_{s_2} \tag{54}$$

It can be seen from the proposition 10 and 11 that all possible facility opening sequence need not be checked to establish the dominance of the scenario. One of the n facilities will be the first (last) facility and the second (second last) facility will be one of the remaining $n-1$ facilities. Thus checking for these $n(n-1)$ possibilities in the the left hand side of inequality in the proposition 10 or 11 will establish the dominance. This dominance calculation will take $O(n^3m|T|)$ time for each scenario and if a scenario is found to be dominated the time saved will be $O(n^3m|T||Iter.|)$. Number of scenarios that have one new server available in first or the last period (excluding the scenarios that have no new server available in first or last period) is $2 \times \binom{n+|T|-3}{n-1} - \binom{n+|T|-5}{n-2} - 2 \times \binom{n+|T|-4}{n-1}$. For example, with $n = 10$ and $|T| = 5$, number of such scenarios is 285 (out of 1001 total possible scenarios).

Proposition 12a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret in the scenario $s_1 = (2, a_2, a_3, .., a_{|T|})$ will not be greater than that in scenario $s_2 = (3, a_2 - 1, a_3, .., a_{|T|})$ if:

$$\bar{d}^{1}_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^{1}_{\pi_1\cup\pi_2} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{55}$$

where, $a_2 > 0$.

Proposition 12b: When $a_2 = 0$ in scenario $(2, a_2, a_3, .., a_{|T|})$ and i be the first period with a non-zero a_i (other than period 1). For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret in the scenario $s_1 = (2, .., a_i, .., a_{|T|})$ will not be greater than that in scenario $s_2 = (3, .., a_i - 1, .., a_{|T|})$ if:

$$(\bar{d}^{1}_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^{1}_{\pi_1\cup\pi_2}) + (\bar{d}^{2}_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^{2}_{\pi_1\cup\pi_2}) + .. + (\bar{d}^{i-1}_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^{i-1}_{\pi_1\cup\pi_2}) \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{56}$$

Proposition 13a: For any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ..., \pi_n)$, regret in the scenario $s_1 = (a_1, a_2, .., a_{|T|-1}, 2)$ will not be greater than that in scenario $s_2 = (a_1, a_2, .., a_{|T|-1} - 1, 3)$ if:

$$\bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} - \bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})} \geq \zeta^*_{s_1} - \zeta^*_{s_2} \tag{57}$$

where, $a_{|T|-1} > 0$.

Proposition 13b: When $a_{|T|-1} = 0$ in scenario $(a_1, .., a_{|T|-1}, 2)$ and i be the last period with a non-zero a_i(other than period $|T|$), for any facility opening sequence $\Pi_k = (\pi_1, \pi_2, ...\pi_n)$, regret in the scenario

$s_1 = (a_1, .., a_i, .., 1)$ will not be greater than that in scenario $s_2 = (a_1, .., a_i - 1, ., 2)$ if:

$$(\bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} - \bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})}) + (\bar{d}^{|T|-2}_{J\setminus(\pi_n,\pi_{n-1})} - \bar{d}^{|T|-2}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})}) + ... + (\bar{d}^{i}_{J\setminus(\pi_n,\pi_{n-1})} - \bar{d}^{i}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})}) \geq \zeta^*_{s_1} - \zeta^*_{s_2} \quad (58)$$

It can be seen from the proposition 12 and 13 that all possible facility opening sequence need not be checked to establish the dominance of the scenario. One of the n facilities will be the first (last) facility, the second (second last) and the third (third last) facility will be one of the remaining $n-1$ and $n-2$ facilities respectively. Thus checking for these $n(n-1)(n-2)$ possibilities in the the left hand side of inequality in the proposition 12 or 13 will establish the dominance. This dominance calculation will take $O(n^4 m|T|)$ time for each scenario and if a scenario is found to be dominated the time saved will be $O(n^3 m|T||Iter.|)$. Number of scenarios that have two new servers available in first or the last period (excluding the scenarios that have no/one new server available in first or last period) is $2 \times \binom{n+|T|-3}{n-1} - \binom{n+|T|-5}{n-2} - 2 \times \binom{n+|T|-4}{n-1} - 2 \times \binom{n+|T|-6}{n-3}$. For example, with $n = 10$ and $|T| = 5$, number of such scenarios is 140 (out of 1001 total possible scenarios).

In a similar fashion we can have dominance rules for three or higher number of servers available in the first (last) period. However, we have not used those relations because of two reasons. Firstly, the computational time to check for dominance condition will be higher and as a result there will not be much saving in the time even if a scenario is found to be dominated. Secondly, the number of scenarios that have not been covered in the earlier propositions will be few. For example, with $n = 10$ and $|T| = 5$, a total of 931 out of 1001 all possible scenarios are covered in the earlier propositions. Using these propositions we have reduced our search space and we can use the search methods given in the next subsection to find a good solution.

4.2 Neighborhood search based methods

These methods start with an initial solution as the current solution and then check the neighborhood for a better solution. The initial solution can be generated using a greedy heuristics. In this problem we have generated the initial solution by arranging the facility in non-increasing order of demand covered over the planning horizon. We have used the swap neighborhood structure. In this neighborhood, a move from one solution to the other is executed by interchanging two distinct elements in the permutation (Taillard, 1990).

Local search (LS): It is a basic neighborhood search heuristics. It starts with a given initial solution as the current solution and checks its neighborhood for a better solution. If such a solution exists, the best neighbor will be selected as the current solution for the next iteration. If the neighborhood of the current solution does not contain any solution better than it, LS returns the current solution and terminates. This method may give a local optima as a solution and does not guarantee globally optimal solutions to most combinatorial problems. However, for many problems it returns relatively good quality solutions.

Tabu search (TS): It is one of the most effective improvements in the local search and was introduced by Glover (1989, 1990). TS starts with a given initial solution as the current solution. It performs a search on the neighbors of the current solution and selects the next solution keeping the search history in mind. This technique allows it to overcome the local optima which local search cannot escape. A tabu list is maintained which prohibits certain moves from the current solution. A tabu tenure specifies the number of iterations a given move will be in the tabu list. It has been shown in the literature that random tabu tenure gives a better performance (Taillard, 1991, 1995). Another important feature of the TS is the aspiration criterion, which ensures that moves which are exceptionally promising are not ignored due to their tabu status.

5 Computational experiments

The procedures were coded in C++ (Visual Studio 2010). IBM ILOG CPLEX 12.4 was used as the IP solver. The procedures were implemented on an personal computer with Intel Core i5 (3.30 GHz) processors; 4 GB RAM; and windows 64-bit operating system. In our implementation the tabu tenure was a random number between 5 and 10. This tabu tenure was first suggested by Gendreau et al. (1999). It reduces the probability of cycling and the search getting stuck at a local minima. We used a simple aspiration criterion: a move is said to satisfy the aspiration criterion if it results in a solution with a regret lower than that of the best solution found thus far. A move satisfying the aspiration criteria can be selected, overruling its tabu status. We conducted some experiments to find out the minimum number of iterations after which the solution does not improve further. In most of the problem instances the solution stabilized before 100 iterations. For a few difficult instances nearly 200 iterations were needed. One such instance with $n = 15$, and $m = 500$ where 174 iterations were needed for solution to stabilize is given in Figure 1. Based on the these findings we allowed our tabu search implementations to run for 250 iterations.

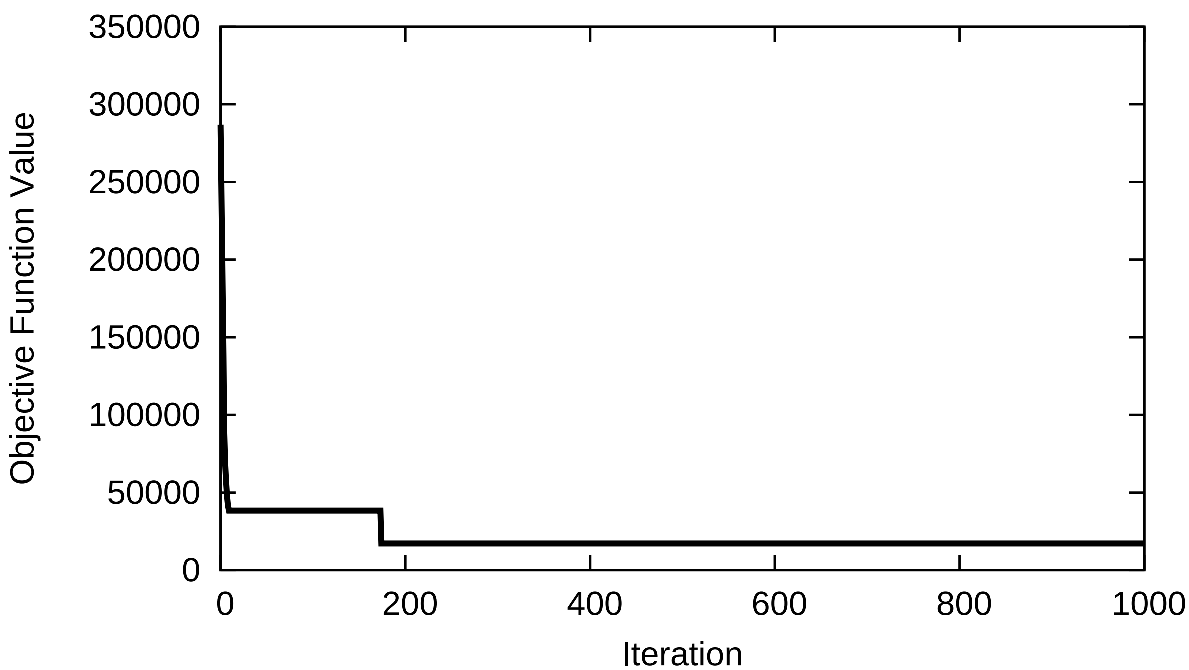

Figure 1: Example of change in objective function value with TS iterations

Problems of size 5, 10, 15 and 20 facilities with 100, 200, 300, 400 and 500 demand nodes were solved for $|T| = 5$ periods for the uncapacitated problem with complete coverage. For the uncapacitated problem with gradual coverage, we conducted the experiments with size 5, 10 and 15 facilities.

We generated abscissa and ordinates of all the demand nodes as random numbers from the uniform distribution [0, 100]. The first period demand at each demand node i was selected from a uniform distribution [50, 1500]. The demand at a node i changes in each period of the planning horizon with a growth rate g_i, which is assumed to remain constant over the planning horizon. Hence, the demand at a node i in period t of the planning horizon is given by $d_i^t = (d_i^{t-1})g_i$. The annual demand growth rate for each demand node has been generated from the uniform distribution [-0.04, 0.10]. This growth rate has been assumed to remain constant over the planning horizon. Covering distance of 20 units was used for the the problem size of 5, 10 and 15 facilities. For the problem size of 20 facilities we used a covering distance of 15 units. For all the problem sizes we assume that there are no open facilities before the start of the planning horizon. The problem instance is generated by calculating the distance between all pairs of demand node i and candidate facility location j. Subsequently we determine a_{ij} i.e. the level of coverage provided by the facility at j to the demand node i. We then generate a set S of all server availability scenarios such that over $|T|$ periods, n servers become available.

We conducted our experiments with two implementations of the tabu search (TS-I and TS-II). In the first implementation (TS-I) we did not use any of the dominance rules. The second implementation (TS-II) included all the dominance rules given in Section 4. In TS-II a set of non-dominated scenarios $S_0 \subset S$, as per proposition 8 through 13, is found out. All the dominated scenarios $S \backslash S_0$ are removed from further calculations in the algorithm. The pseudocodes for TS-II implementation is given in Algorithm 1. In TS-I instead of the set S_0, we consider the set S for calculation in step 17 and 25 of Algorithm 1. Note that in our implementations, the maximum demand coverage for each scenario was computed using CPLEX.

For each of problem sizes we report the summary of 10 instances which do not give zero regret solutions. Notice that such instances are more difficult to solve compared to the instances which output zero regret solutions, because once a feasible solution (upper bound) of zero is found there can be no further improvement and the search terminates. We present the summary of the performance of the solution methods on these problems in Tables 3 through 9, more details has been provided in Appendix B. Tables 3 through 6 pertain to the problem with complete coverage and tables 7 through 9 pertain to the problem with gradual coverage. We report the average and the maximum times with different methods for the 10 instances not giving zero regret solutions. Time taken to find the maximal coverage for all the scenarios using exact method (CPLEX 12.4) is reported first. Then we report the time taken using CPLEX (excluding the time taken to find maximal coverage for all scenarios). For this exact method we compare the time taken without the scenario dominance rules (CP-I) and time taken when scenario dominance rules are included in the problem formulation (CP-II). Then we report the solution times for the two TS implementations TS-I and TS-II. The column Opt# report the number of instances out of the 10 instances with non-zero regret, the TS implementation could find an optimal solution (CP-I or CP-II solution) .

For the problem size of $n = 15$ and $m = 100$, 200, 300, 400 and 500 in MULPSU, we found the best lower and upper bound found using CP-II at the end of one hour. We also found the best lower and upper bound for the problem size of $n = 10$ and $m = 100$, 200, 300, 400 and 500 with MULPSU-P. This can be used to compare the quality of the solution returned by the exact method at the end of one hour with the heuristic

Algorithm 1 Tabu search (TS-II implementation)

Input: A problem instance, tabu tenure parameter t, and the total no. of iterations k

Output: A facility opening sequence with minimax regret of demand coverage, and the maximum regret associated with this sequence

Code

1: find the maximal demand coverage for each scenario $s \in S$, by solving the multi-period maximal coverage location problem (complete or gradual coverage);
2: using dominance rules find a set of scenarios $S_0 \subset S$ which are not dominated;
3: generate initial facility opening sequence sequence Π (decreasing order of the total demand covered over the planning horizon);
4: find coverage and regret with Π for each scenario $s \in S_0$;
5: set $\Pi^* \leftarrow \Pi$ and $minimax\_reg^* \leftarrow$ maximum regret with Π across scenarios;
6: **for** iterations from 1 to k **do**
7:     **if** $minimax\_reg^* == 0$ **then**
8:         break;
9:     **end if**
10:     generate all neighbors Π_N of Π;
11:     **for** each neighbor Π_n of Π **do**
12:         **if** Π dominates Π_n **then**
13:             go to next neighbor
14:         **else**
15:             **if** Π_n is a non-tabu neighbor Π_n **then**
16:                 set $nontabu\_regret\_neigh \leftarrow \infty$;
17:                 find coverage and regret with Π_n for each scenario $s \in S_0$;
18:                 set $tmp\_regret \leftarrow$ maximum regret with Π_n across scenarios;
19:                 **if** $tmp\_regret < nontabu\_regret\_neigh$ **then**
20:                     set $\Pi^{non-tabu\_nbr} \leftarrow \Pi_n$;
21:                     set $nontabu\_regret\_neigh \leftarrow tmp\_regret$;
22:                 **end if**
23:             **else**
24:                 set $tabu\_regret\_neigh \leftarrow \infty$;
25:                 find coverage and regret with Π_n for each scenario $s \in S_0$;
26:                 set $tmp\_regret \leftarrow$ maximum regret with Π_n across scenarios;
27:                 **if** $tmp\_regret < tabu\_regret\_neigh$ **then**
28:                     set $\Pi^{tabu\_nbr} \leftarrow \Pi_n$ and $tabu\_regret\_neigh \leftarrow tmp\_regret$;
29:                 **end if**
30:             **end if**
31:         **end if**
32:     **end for**
33:     **if** there are no undominated non-tabu neighbors of Π **then**
34:         find the best dominated non-tabu neighbor using step 15 to 22 for each neighbor Π_n of Π;
35:     **end if**
36:     select the best neighbor Π^{nbr} keeping a check on the aspiration criterion;
37:     set $max\_regret\_neigh \leftarrow$ maximum regret with Π^{nbr} across scenarios;
38:     update the tabu list using tabu tenure t selected randomly between an upper and a lower bound;
39:     **if** $max\_regret\_neigh < minimax\_reg^*$ **then**
40:         set $\Pi^* \leftarrow \Pi^{nbr}$ and $minimax\_reg^* \leftarrow max\_regret\_neigh$;
41:     **end if**
42:     set $\Pi \leftarrow \Pi^{nbr}$ for the next iteration;
43: **end for**
44: output Π^* and $minimax\_reg^*$;

solution. We observe that for all the instances, the best feasible solution found by CPLEX at the end of one hour is not better than the solution given by TS-II implementation.

Table 3: Computational results for MULPSU with $n = 5$, and $|T| = 5$

	MMCLPs	CPLEX		TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*68.33% of the solutions were zero regret solutions*)							
Avg.	0.29	0.53	0.19	0.70	10	0.16	10
Max.	0.35	0.78	0.36	0.72		0.19	
$m = 200$ (*71.67% of the solutions were zero regret solutions*)							
Avg.	0.42	0.96	0.37	1.35	10	0.32	10
Max.	0.58	1.37	0.87	1.38		0.46	
$m = 300$ (*78.33% of the solutions were zero regret solutions*)							
Avg.	0.51	1.36	0.51	2.02	10	0.52	10
Max.	0.64	2.54	1.31	2.11		0.75	
$m = 400$ (*70.0% of the solutions were zero regret solutions*)							
Avg.	0.68	2.44	0.88	2.69	10	0.66	10
Max.	0.95	4.51	1.45	2.75		1.03	
$m = 500$ (*76.67% of the solutions were zero regret solutions*)							
Avg.	0.81	2.51	1.17	3.33	10	0.90	10
Max.	1.09	3.95	2.89	3.43		1.37	

Table 4: Computational results for MULPSU with $n = 10$, and $|T| = 5$

	MMCLPs	CPLEX		TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*13.33% of the solutions were zero regret solutions*)							
Avg.	4.90	158.59	37.85	35.30	10	5.42	10
Max.	6.16	331.94	67.91	37.62		9.25	
$m = 200$ (*37.5% of the solutions were zero regret solutions*)							
Avg.	8.07	399.47	64.16	69.57	10	8.82	10
Max.	12.10	1981.92	131.17	73.23		14.11	
$m = 300$ (*20.0% of the solutions were zero regret solutions*)							
Avg.	8.46	527.01	156.12	100.03	9	13.67	10
Max.	12.02	2302.27	273.40	102.80		25.77	
$m = 400$ (*13.33% of the solutions were zero regret solutions*)							
Avg.	10.31	637.45	335.26	133.40	8	19.77	10
Max.	14.34	1481.53	1193.70	137.01		27.80	
$m = 500$ (*33.33% of the solutions were zero regret solutions*)							
Avg.	23.67	2619.96	1059.83	175.39	10	30.37	10
Max.	56.92	11581.27	3786.10	181.96		54.26	

It can be seen from the tables 4, 5 and 8 that TS-I could not give the optimal or the best solution for a few problems, while the tabu search implementation TS-II gave the optimal solution for all the instances which CPLEX could solve in reasonable time. It so happens because with the dominance rule on facility opening sequence, a set of non-optimal solutions is cut off form the consideration set while doing the neighborhood search. Consequently, when an incumbent solution is better than all its neighbors, a dominated non-tabu move might be carried out in TS-I. This sometime entraps the search into a local minima. TS-II prevents such moves and reduces chances of getting stuck into a local minima.

For the complete coverage problem CPLEX could solve most of the problems upto the size of 15 facilities and 200 demand nodes in 24 hours, while for the gradual coverage problem, size of 10 facilities and 100

Table 5: Computational results for MULPSU with $n = 15$, and $|T| = 5$

	MMCLPs	CPLEX		TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	22.04	36469.57	27277.72	431.69	8	60.77	10
Max.	28.40	171646.02	143033.60	455.89		157.18	
$m = 200$ (*7.14% of the solutions were zero regret solutions*)							
Avg.	34.03	-	13343.96	734.82	10	83.59	10
Max.	56.01	-	60755.85	769.04		164.20	
$m = 300$ (*6.67% of the solutions were zero regret solutions*)							
Avg.	43.01	-	-	1090.20	-	136.41	-
Max.	49.31	-	-	1157.80		466.20	
$m = 400$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	59.78	-	-	1429.45	-	160.85	-
Max.	93.34	-	-	1493.53		491.77	
$m = 500$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	76.29	-	-	1791.11	-	226.48	-
Max.	154.97	-	-	1832.42		476.77	

Table 6: Computational results for MULPSU with $n = 20$, and $|T| = 5$

	MMCLPs	CPLEX		TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	51.21	-	-	2581.69	-	137.73	-
Max.	58.27	-	-	2970.99		345.96	
$m = 200$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	77.69	-	-	5099.88	-	382.99	-
Max.	121.64	-	-	5324.05		1803.33	
$m = 300$ (*9.09% of the solutions were zero regret solutions*)							
Avg.	111.64	-	-	7386.42	-	318.03	-
Max.	140.71	-	-	7561.08		629.08	
$m = 400$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	119.83	-	-	9682.58	-	669.64	-
Max.	141.36	-	-	9967.05		2546.68	
$m = 500$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	164.75	-	-	12921.81	-	797.65	-
Max.	197.84	-	-	16072.58		1402.69	

Table 7: Computational results for MULPSU-P with $n = 5$, and $|T| = 5$

	MMCLP-Ps	**CPLEX**		**TS-I**		**TS-II**	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*66.67% of the solutions were zero regret solutions*)							
Avg.	0.59	5.21	2.01	1.52	10	0.42	10
Max.	0.80	9.44	3.65	1.58		0.58	
$m = 200$ (*67.74% of the solutions were zero regret solutions*)							
Avg.	1.49	16.78	7.83	2.89	10	0.91	10
Max.	2.09	34.90	19.33	3.07		1.20	
$m = 300$ (*68.75% of the solutions were zero regret solutions*)							
Avg.	1.78	18.66	7.68	4.36	10	1.35	10
Max.	3.37	46.71	27.96	4.74		1.97	
$m = 400$ (*62.96% of the solutions were zero regret solutions*)							
Avg.	4.60	44.12	21.29	5.91	10	2.17	10
Max.	7.85	84.33	52.99	6.23		2.61	
$m = 500$ (*73.68% of the solutions were zero regret solutions*)							
Avg.	4.62	53.91	16.07	7.54	10	2.31	10
Max.	12.42	147.03	27.99	7.77		2.98	

Table 8: Computational results for MULPSU-P with $n = 10$, and $|T| = 5$

	MMCLP-Ps	**CPLEX**		**TS-I**		**TS-II**	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt#	CPU (s)	Opt#
$m = 100$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	28.98	22652.95	11172.37	69.45	10	13.77	10
Max.	35.15	101723.11	53292.26	73.59		22.03	
$m = 200$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	42.99	-	8931.43	139.62	9	26.12	10
Max.	69.62	-	27674.67	148.14		36.19	
$m = 300$ (*9.09% of the solutions were zero regret solutions*)							
Avg.	92.54	-	-	224.56	-	37.43	-
Max.	123.08	-	-	257.39		51.78	
$m = 400$ (*16.67% of the solutions were zero regret solutions*)							
Avg.	167.19	-	-	302.20	-	55.52	-
Max.	231.14	-	-	312.54		70.03	
$m = 500$ (*23.08% of the solutions were zero regret solutions*)							
Avg.	241.29	-	-	393.92	-	84.02	-
Max.	309.98	-	-	413.80		144.16	

Table 9: Computational results for MULPSU-P with $n = 15$, and $|T| = 5$

	MMCLP-Ps	CPLEX		TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	CPU (s)	Opt #	CPU (s)	Opt #
$m = 100$ (*9.09% of the solutions were zero regret solutions*)							
Avg.	125.80	-	-	830.69	-	79.24	-
Max.	174.51	-	-	934.99		123.30	
$m = 200$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	379.93	-	-	1714.40	-	160.03	-
Max.	739.90	-	-	1884.22		302.92	
$m = 300$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	675.93	-	-	2489.32	-	235.83	-
Max.	996.94	-	-	2693.89		374.07	
$m = 400$ (*0.0% of the solutions were zero regret solutions*)							
Avg.	975.75	-	-	3419.59	-	425.92	-
Max.	1361.29	-	-	3670.19		601.84	
$m = 500$ (*9.09% of the solutions were zero regret solutions*)							
Avg.	1356.31	-	-	4357.58	-	394.59	-
Max.	1693.78	-	-	4664.50		491.16	

demand nodes could be solved using CPLEX in that time. Higher sized problems could not be solved due to memory limitations. CPLEX took much larger amount of time compared to the time taken by the TS implementations to solve the larger instances of the problem. Among the TS implementations, TS-II took significantly lower time than TS-I, while it was able to find the best solution for all the instances. We also notice that the quality of feasible solution given by the exact solver in one hour for the large instances (which could not be solved in one hour by CP-II) is much worse than the solution obtained by TS-II implementation in most cases.

6 Conclusions

In this paper we have provided a formulation as well as solution method for the multi-period facility location problem with an uncertain number of servers, and the facility opening sequence as the decision variable. We looked at two variants of the problem. First is the problem with complete coverage where all the demand nodes falling just within the covering distance from a facility are considered covered, while those just outside the covering distance are not covered. In the other variant we generalized the problem by allowing partial coverage between a minimum and a maximum covering distance. Partial or gradual covering problems have been studied by Church & Roberts (1983); Berman et al. (2003); Karasakal & Karasakal (2004); Berman et al. (2010). For both the above mentioned problem variants we evaluate the performance of commercial solver and two implementations of tabu search with a swap neighborhood structure. The best customized tabu search algorithm was able to solve instances of practical size which CPLEX could not solve. Furthermore, it gave the optimal solution for all instances which could be solved using CPLEX and took significantly lesser time.

We have used tabu search based meta-heuristics to solve the multi-period facility location problem with an uncertain number of servers. Other exact and approximation methods can be further explored to solve a similar problem. Another extension of this work will be to consider the problem with capacity limitations on facilities. Furthermore, if the probability of different scenarios can be ascertained beforehand, other decision making criterion instead of minimax regret can be studied. Some examples are maximizing the expected coverage over all scenarios, minimizing the expected regret or α-reliable p-minimax regret.

Appendix A

Proposition 6:

Proof. Assume that Π_1 does not dominate Π_2 even though all the above relationships are satisfied. Then there must exist a scenario of server availability $s = (a_1, a_2, ..a_{|T|}) \in S$ for which regret with Π_1 is more than regret with Π_2, or in other words demand coverage with Π_1 is less than the demand coverage with Π_2. If this hold then there must be atleast one period for which demand covered in that period with Π_1 is less than the demand coverage with Π_2, i.e.

$$\exists t \in T : \bar{d}^t_{\pi_1 \cup \pi_2 \cup ... first\ (a_1+a_2+..a_t)\ facilities\ in\ \Pi_1} < \bar{d}^t_{\pi_1 \cup \pi_2 \cup ... first\ (a_1+a_2+..a_t)\ facilities\ in\ \Pi_2}$$

This contradicts the assumption that all the relations in the proposition are satisfied. Hence, Π_1 will dominate Π_2 if all the above set of relationships are satisfied. □

Proposition 7:

Proof. It is easy to see that when all n servers become available in the same period, the facility opening sequence becomes immaterial, i.e. all facility opening sequence will provide the same demand coverage. □

Proposition 8a:

Proof. Notice that $\bar{d}^1_{\pi_1}$ is the additional demand covered when the first facility of the facility opening sequence Π_k is opened in the first period instead of the second period. Hence, $\bar{d}^1_{\pi_1} = \zeta_{\Pi_k, s_2} - \zeta_{\Pi_k, s_1}$ and we have:

$$\bar{d}^1_{\pi_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1}$$

$$\Rightarrow \zeta_{\Pi_k, s_2} - \zeta_{\Pi_k, s_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{A.1}$$

$$\Rightarrow \zeta^*_{s_1} - \zeta_{\Pi_k, s_1} \leq \zeta^*_{s_2} - \zeta_{\Pi_k, s_2} \tag{A.2}$$

or, Regret associated with the scenario $s_1 = (0, a_2, a_3, .., a_{|T|}) \leq$ Regret associated with the scenario $s_2 = (1, a_2 - 1, a_3, .., a_{|T|})$ □

Proposition 8b:

Proof. The left hand side of the above inequality is the coverage by the first facility of the facility opening sequence $\Pi_k \in \Pi$, in the first period through $i-1$ periods of the planning horizon. Hence, it is the additional demand covered when the first facility of the facility opening sequence Π_k is opened in the first period instead of the i^{th} period. The proof for this is similar to the earlier one. Note that in this case

$$\zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1} = \bar{d}^1_{\pi_1} + \bar{d}^2_{\pi_1} + .. + \bar{d}^{i-1}_{\pi_1} \tag{A.3}$$

□

Proposition 9a:

Proof. The left hand side of the above inequality is the unique coverage by the last facility of the facility opening sequence $\Pi_k \in \Pi$, in the second last period of the planning horizon. Hence, $\bar{d}^{|T|-1}_J - \bar{d}^{|T|-1}_{J\setminus\pi_n} = \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2}$, and we have:

$$\bar{d}^{|T|-1}_J - \bar{d}^{|T|-1}_{J\setminus\pi_n} \geq \zeta^*_{s_1} - \zeta^*_{s_2}$$
$$\Rightarrow \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta^*_{s_2} \tag{A.4}$$
$$\Rightarrow \zeta^*_{s_2} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta_{\Pi_k,s_1} \tag{A.5}$$

or, Regret associated with the scenario $s_1 = (a_1, a_2, .., a_{|T|-1} - 1, 1) \geq$ Regret associated with the scenario $s_2 = (a_1, a_2, .., a_{|T|-1}, 0)$ □

Proposition 9b:

Proof. The left hand side of the above inequality is the unique demand covered by the last facility of any facility opening sequence $\Pi_k \in \Pi$, in the i^{th} period through $(T-1)^{th}$ periods of the planning horizon. Hence, it is the additional demand covered when the last facility of the facility opening sequence Π_k is opened in the i^{th} period instead of the last period. The proof for this is similar to the earlier one. Note that in this case the the left hand side of inequality is given by $\zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1}$. □

Proposition 10a:

Proof. The left hand side of the above inequality is the unique coverage by the second facility of any facility opening sequence $\Pi_k \in \Pi$, in the first period of the planning horizon. Hence, it is the additional demand covered when the second facility of the facility opening sequence Π_k is opened in the first period instead of the second period. Therefore, $\bar{d}^1_{\pi_1\cup\pi_2} - \bar{d}^1_{\pi_1} = \zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1}$ and we have:

$$\bar{d}^1_{\pi_1\cup\pi_2} - \bar{d}^1_{\pi_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1}$$
$$\Rightarrow \zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \tag{A.6}$$
$$\Rightarrow \zeta^*_{s_1} - \zeta_{\Pi_k,s_1} \leq \zeta^*_{s_2} - \zeta_{\Pi_k,s_2} \tag{A.7}$$

or, Regret associated with the scenario $s_1 = (1, a_2, a_3, .., a_{|T|}) \leq$ Regret associated with the scenario $s_2 = (2, a_2 - 1, a_3, .., a_{|T|})$ □

Proposition 10b:

Proof. The left hand side of the above inequality is the unique coverage by the second facility of any facility opening sequence $\Pi_k \in \Pi$, in the first period through $i-1$ periods of the planning horizon. Hence, it is the additional demand covered when the second facility of the facility opening sequence Π_k is opened in the first period instead of the i^{th} period. The proof for this is similar to the earlier one. □

Proposition 11a:

Proof. The left hand side of the above inequality is the unique coverage by the second last facility of any facility opening sequence $\Pi_k \in \Pi$, in the second last period of the planning horizon. Hence, $\bar{d}^{|T|-1}_{J\setminus\pi_n} - \bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} = \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2}$ and we have:

$$\begin{aligned} &\bar{d}^{|T|-1}_{J\setminus\pi_n} - \bar{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} \geq \zeta^*_{s_1} - \zeta^*_{s_2} \\ &\Rightarrow \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta^*_{s_2} \qquad \text{(A.8)} \\ &\Rightarrow \zeta^*_{s_2} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta_{\Pi_k,s_1} \qquad \text{(A.9)} \end{aligned}$$

or, Regret associated with the scenario $s_2 = (a_1, a_2, .., a_{|T|-1} - 1, 2) \geq$ Regret associated with the scenario $s_1 = (a_1, a_2, .., a_{|T|-1}, 1)$ □

Proposition 11b:

Proof. The left hand side of the above inequality is the unique demand covered by the second last facility of any facility opening sequence $\Pi_k \in \Pi$, in the i^{th} period through $(T-1)^{th}$ periods of the planning horizon. The proof for this is similar to the earlier one. □

Proposition 12a:

Proof. The left hand side of the above inequality is the unique coverage by the third facility of any facility opening sequence $\Pi_k \in \Pi$, in the first period of the planning horizon. Hence $\bar{d}^1_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^1_{\pi_1\cup\pi_2} = \zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1}$ and we have:

$$\begin{aligned} &\bar{d}^1_{\pi_1\cup\pi_2\cup\pi_3} - \bar{d}^1_{\pi_1\cup\pi_2} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \\ &\Rightarrow \zeta_{\Pi_k,s_2} - \zeta_{\Pi_k,s_1} \leq \zeta^*_{s_2} - \zeta^*_{s_1} \qquad \text{(A.10)} \\ &\Rightarrow \zeta^*_{s_1} - \zeta_{\Pi_k,s_1} \leq \zeta^*_{s_2} - \zeta_{\Pi_k,s_2} \qquad \text{(A.11)} \end{aligned}$$

or, Regret in scenario $s_1 = (2, a_2, a_3, .., a_{|T|}) \leq$ Regret in the scenario $s_2 = (3, a_2 - 1, a_3, .., a_{|T|})$ □

Proposition 12b:

Proof. The left hand side of the above inequality is the unique coverage by the third facility of any facility opening sequence $\Pi_k \in \Pi$ in the first period through $i-1$ periods of the planning horizon. The proof for this is similar to the earlier one. □

Proposition 13a:

Proof. The left hand side of the above inequality is the unique coverage by the third last facility of any facility opening sequence $\Pi_k \in \Pi$ in the second last period of the planning horizon. Hence $\vec{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} - \vec{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})} = \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2}$ and we have:

$$\vec{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1})} - \vec{d}^{|T|-1}_{J\setminus(\pi_n,\pi_{n-1},\pi_{n-2})} \geq \zeta^*_{s_1} - \zeta^*_{s_2}$$

$$\Rightarrow \zeta_{\Pi_k,s_1} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta^*_{s_2} \quad (A.12)$$

$$\Rightarrow \zeta^*_{s_2} - \zeta_{\Pi_k,s_2} \geq \zeta^*_{s_1} - \zeta_{\Pi_k,s_1} \quad (A.13)$$

or, Regret in the scenario $s_2 = (a_1, a_2, .., a_{|T|-1} - 1, 3) \geq$ Regret in scenario $s_1 = (a_1, a_2, .., a_{|T|-1}, 2)$ □

Proposition 13b:

Proof. The left hand side of the above inequality is the unique demand covered by the third last facility of any facility opening sequence $\Pi_k \in \Pi$, in the i^{th} period through $(|T|-1)^{th}$ periods of the planning horizon. The proof for this is similar to the earlier one. □

Appendix B

Table B.1: Computational results for MULPSU with $n = 5$, $m = 100$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.32	0.50	0.18	16.6	0.67	0.0	0.15	0.0
2	0.24	0.43	0.14	687.0	0.72	0.0	0.18	0.0
3	0.27	0.38	0.11	472.2	0.71	0.0	0.14	0.0
4	0.27	0.48	0.13	45.6	0.70	0.0	0.14	0.0
5	0.35	0.78	0.36	1015.5	0.70	0.0	0.18	0.0
6	0.30	0.55	0.21	203.0	0.71	0.0	0.19	0.0
7	0.30	0.55	0.17	16.6	0.71	0.0	0.15	0.0
8	0.28	0.53	0.28	2248.1	0.70	0.0	0.17	0.0
9	0.30	0.48	0.16	13.0	0.70	0.0	0.15	0.0
10	0.30	0.58	0.19	133.2	0.70	0.0	0.15	0.0
Avg.	0.29	0.53	0.19		0.70		0.16	
Max.	0.35	0.78	0.36		0.72		0.19	

68.33% of the solutions were zero regret solutions for this problem size

Table B.2: Computational results for MULPSU with $n = 5$, $m = 200$ and $|T| = 5$

Instance	**MMCLPs**	**CPLEX**			**TS-I**		**TS-II**	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.36	0.69	0.22	115.0	1.34	0.0	0.31	0.0
2	0.38	0.87	0.27	159.0	1.31	0.0	0.23	0.0
3	0.47	1.03	0.44	6772.5	1.28	0.0	0.31	0.0
4	0.38	0.87	0.30	4.0	1.37	0.0	0.30	0.0
5	0.48	1.37	0.87	6011.5	1.38	0.0	0.46	0.0
6	0.36	0.72	0.25	576.8	1.37	0.0	0.31	0.0
7	0.33	0.56	0.17	136.0	1.36	0.0	0.28	0.0
8	0.58	1.36	0.34	3585.0	1.35	0.0	0.40	0.0
9	0.41	1.20	0.50	474.2	1.38	0.0	0.33	0.0
10	0.44	0.92	0.30	391.9	1.35	0.0	0.25	0.0
Avg.	0.42	0.96	0.37		1.35		0.32	
Max.	0.58	1.37	0.87		1.38		0.46	

71.67% of the solutions were zero regret solutions for this problem size

Table B.3: Computational results for MULPSU with $n = 5$, $m = 300$ and $|T| = 5$

Instance	**MMCLPs**	**CPLEX**			**TS-I**		**TS-II**	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.55	1.42	0.44	852.5	2.10	0.0	0.38	0.0
3	0.48	1.00	0.28	130.0	2.11	0.0	0.51	0.0
4	0.53	1.54	0.75	2449.3	2.10	0.0	0.75	0.0
10	0.44	1.05	0.34	730.9	2.07	0.0	0.44	0.0
12	0.56	1.33	0.39	157.1	2.03	0.0	0.38	0.0
20	0.39	0.87	0.25	106.0	1.95	0.0	0.47	0.0
23	0.41	0.94	0.30	30.0	1.88	0.0	0.47	0.0
25	0.52	1.15	0.33	94.9	1.91	0.0	0.51	0.0
30	0.56	1.73	0.69	452.1	2.03	0.0	0.61	0.0
36	0.64	2.54	1.31	5509.5	2.03	0.0	0.69	0.0
Avg.	0.51	1.36	0.51		2.02		0.52	
Max.	0.64	2.54	1.31		2.11		0.75	

78.33% of the solutions were zero regret solutions for this problem size

Table B.4: Computational results for MULPSU with $n = 5$, $m = 400$ and $|T| = 5$

Instance	**MMCLPs**	**CPLEX**			**TS-I**		**TS-II**	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.70	2.36	1.01	9968.8	2.65	0.0	0.53	0.0
2	0.62	1.78	0.64	863.7	2.64	0.0	0.54	0.0
3	0.81	2.64	1.08	3478.1	2.75	0.0	0.77	0.0
4	0.52	1.33	0.44	1476.0	2.71	0.0	0.65	0.0
5	0.50	1.22	0.37	449.3	2.68	0.0	0.58	0.0
6	0.73	1.97	1.45	18969.7	2.66	0.0	1.03	0.0
7	0.67	2.96	1.40	5742.5	2.69	0.0	0.66	0.0
8	0.50	1.47	0.44	349.1	2.71	0.0	0.60	0.0
9	0.76	4.51	0.73	353.9	2.72	0.0	0.66	0.0
10	0.95	4.18	1.20	3575.2	2.66	0.0	0.60	0.0
Avg.	0.68	2.44	0.88		2.69		0.66	
Max.	0.95	4.51	1.45		2.75		1.03	

70.0% of the solutions were zero regret solutions for this problem size

Table B.5: Computational results for MULPSU with $n = 5$, $m = 500$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.95	3.48	2.89	10055.0	3.26	0.0	1.02	0.0
2	1.09	3.87	1.64	2432.2	3.30	0.0	1.06	0.0
3	0.72	2.15	0.86	441.0	3.37	0.0	0.94	0.0
4	0.64	1.75	0.52	107.8	3.34	0.0	0.76	0.0
5	0.59	1.39	0.44	178.0	3.20	0.0	0.78	0.0
6	0.97	3.95	1.55	20461.3	3.34	0.0	0.75	0.0
7	0.87	2.23	1.03	408.0	3.34	0.0	0.82	0.0
8	0.94	2.53	1.59	1884.1	3.31	0.0	1.37	0.0
9	0.62	1.47	0.44	6.0	3.40	0.0	0.72	0.0
10	0.73	2.26	0.81	275.0	3.43	0.0	0.76	0.0
Avg.	0.81	2.51	1.17		3.33		0.90	
Max.	1.09	3.95	2.89		3.43		1.37	

76.67% of the solutions were zero regret solutions for this problem size

Table B.6: Computational results for MULPSU with $n = 10$, $m = 100$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	3.67	34.14	13.95	159.6	34.73	0.0	9.25	0.0
2	4.11	66.67	20.29	1532.4	34.11	0.0	4.74	0.0
3	4.51	331.94	21.60	3822.3	35.05	0.0	2.38	0.0
4	4.15	185.84	55.31	3643.9	34.63	0.0	2.63	0.0
5	5.95	263.73	67.91	6197.2	35.65	0.0	5.48	0.0
6	6.16	92.96	57.95	2507.9	35.58	0.0	3.56	0.0
7	4.67	215.57	55.60	2911.0	37.62	0.0	5.91	0.0
8	5.32	139.12	44.51	2442.2	34.97	0.0	9.17	0.0
9	5.45	137.90	22.20	249.7	33.85	0.0	8.50	0.0
10	5.03	118.07	19.19	5065.4	36.81	0.0	2.61	0.0
Avg.	4.90	158.59	37.85		35.30		5.42	
Max.	6.16	331.94	67.91		37.62		9.25	

13.33% of the solutions were zero regret solutions for this problem size

Table B.7: Computational results for MULPSU with $n = 10$, $m = 200$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	11.41	428.07	100.46	4673.2	68.73	0.0	7.76	0.0
2	8.32	261.08	39.98	7509.0	69.61	0.0	10.14	0.0
3	7.17	108.50	131.17	19828.6	73.23	0.0	11.37	0.0
4	5.24	115.04	41.64	115.7	70.94	0.0	14.11	0.0
5	5.74	122.69	48.12	8758.8	69.27	0.0	4.86	0.0
6	12.10	1981.92	66.55	13179.4	70.57	0.0	6.90	0.0
7	6.19	186.49	57.25	1536.8	68.38	0.0	9.31	0.0
8	6.70	430.22	59.59	5849.1	68.10	0.0	5.29	0.0
9	6.85	85.62	25.04	36.0	68.11	0.0	12.52	0.0
10	11.01	275.05	71.82	212.4	68.78	0.0	5.94	0.0
Avg.	8.07	399.47	64.16		69.57		8.82	
Max.	12.10	1981.92	131.17		73.23		14.11	

37.50% of the solutions were zero regret solutions for this problem size

Table B.8: Computational results for MULPSU with $n = 10$, $m = 300$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	8.49	275.70	219.65	8557.7	101.09	0.0	10.57	0.0
2	7.87	343.57	190.84	5174.2	98.33	0.0	7.12	0.0
3	8.40	254.94	268.08	7986.4	102.11	0.0	25.77	0.0
4	6.79	136.10	78.29	255.9	102.80	0.0	22.25	0.0
5	8.20	2302.27	273.40	21763.9	101.00	4473.3	11.23	0.0
6	8.83	702.33	169.55	4618.1	99.45	0.0	12.22	0.0
7	12.02	374.08	119.63	863.1	97.58	0.0	9.55	0.0
8	7.79	71.12	18.64	13.0	101.75	0.0	19.79	0.0
9	6.53	457.43	83.75	5917.8	96.89	0.0	11.01	0.0
10	9.73	352.52	139.34	3627.6	99.31	0.0	7.14	0.0
Avg.	8.46	527.01	156.12		100.03		13.67	
Max.	12.02	2302.27	273.40		102.80		25.77	

20.0% of the solutions were zero regret solutions for this problem size

Table B.9: Computational results for MULPSU with $n = 10$, $m = 400$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	9.16	475.14	139.19	10688.6	132.68	0.0	19.16	0.0
2	7.70	123.37	41.99	374.6	134.85	0.0	27.64	0.0
3	8.11	132.78	46.48	396.9	134.71	0.0	24.28	0.0
4	8.59	345.42	185.66	10033.1	135.28	0.0	11.36	0.0
5	14.34	1127.47	841.57	20531.7	132.77	0.0	13.89	0.0
6	9.81	1222.05	1193.70	14744.4	137.01	6597.6	16.68	0.0
7	10.86	439.29	109.90	13783.7	131.87	0.0	18.41	0.0
8	12.71	603.44	243.70	7549.6	134.92	0.0	20.13	0.0
9	11.63	1481.53	412.19	15144.2	132.22	2653.8	18.32	0.0
10	10.17	423.98	138.22	2192.6	127.72	0.0	27.80	0.0
Avg.	10.31	637.45	335.26		133.40		19.77	
Max.	14.34	1481.53	1193.70		137.01		27.80	

13.33% of the solutions were zero regret solutions for this problem size

Table B.10: Computational results for MULPSU with $n = 10$, $m = 500$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	20.43	1091.96	550.78	1966.1	176.91	0.0	14.48	0.0
2	17.38	5225.95	1013.19	9289.5	172.32	0.0	21.59	0.0
3	12.80	608.88	230.60	5094.7	175.49	0.0	14.61	0.0
4	19.05	680.13	328.40	7181.4	176.87	0.0	54.26	0.0
5	16.87	1795.82	298.39	9362.9	177.89	0.0	47.94	0.0
6	16.79	886.58	580.06	2680.2	174.14	0.0	28.58	0.0
7	22.09	2008.01	217.66	3969.9	174.84	0.0	32.71	0.0
8	26.33	11581.27	3786.10	14057.9	167.74	0.0	26.07	0.0
9	28.08	1837.01	3366.29	14543.8	181.96	0.0	26.66	0.0
10	56.92	484.03	226.85	42.0	175.74	0.0	36.78	0.0
Avg.	23.67	2619.96	1059.83		175.39		30.37	
Max.	56.92	11581.27	3786.10		181.96		54.26	

33.33% of the solutions were zero regret solutions for this problem size

Table B.11: Computational results for MULPSU with $n = 15$, $m = 100$ and $|T| = 5$

Instance	MMCLPs	CPLEX			TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap	UB	LB
1	20.07	2054.8	398.7	159.6	432.3	5132.5	15.2	0.0	-	-
2	16.69	1266.0	285.9	73.7	455.9	0.0	101.3	0.0	-	-
3	21.97	6200.4	2045.8	6075.3	411.7	0.0	32.4	0.0	-	-
4	19.93	2355.6	483.5	18.8	439.9	0.0	125.8	0.0	-	-
5	21.12	70710.4	3093.7	3221.5	414.1	0.0	15.6	0.0	-	-
6	27.39	4785.8	3515.2	5471.8	443.0	0.0	76.6	0.0	-	-
7	25.83	102270.2	118323.8	7631.9	425.2	0.0	31.6	0.0	99190.9	5268.3
8	28.40	171646.0	143033.6	4106.3	429.1	0.0	15.8	0.0	4717.9	3152.8
9	18.27	1596.6	626.5	3293.7	443.9	4060.2	36.2	0.0	-	-
10	20.70	1810.0	970.5	249.7	421.7	0.0	157.2	0.0	-	-
Avg.	22.04	36469.6	27277.7		431.7		60.8			
Max.	28.40	171646.0	143033.6		455.9		157.2			

0.0% of the solutions were zero regret solutions for this problem size

Table B.12: Computational results for MULPSU with $n = 15$, $m = 200$ and $|T| = 5$

Instance	MMCLPs	CPLEX		TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CP-II (s)	Solution	CPU (s)	Solution	CPU (s)	Solution	UB	LB
1	48.2	60755.9	13335.4	699.5	13335.4	60.0	13335.4	132904.0	10559.9
2	33.6	1491.6	7509.0	737.1	7509.0	46.1	7509.0	-	-
3	30.7	33042.6	4833.0	731.6	4833.0	55.5	4833.0	4833.0	3885.9
4	27.2	5258.4	9274.4	769.0	9274.4	62.0	9274.4	166950.0	9274.4
5	30.2	1428.5	3608.8	734.0	3608.8	121.7	3608.8	-	-
6	26.5	10357.9	904.0	749.4	904.0	77.0	904.0	904.0	760.8
7	29.9	2545.6	3215.5	690.7	3215.5	47.1	3215.5	-	-
8	56.0	*	*	757.7	6263.6	115.5	6263.6	93780.1	0.0
9	30.2	3632.4	16059.5	758.1	16059.5	86.8	16059.5	16059.5	11948.5
10	27.7	1582.7	1853.6	721.1	1853.6	164.2	1853.6	-	-
Avg.	34.0	13344.0		734.8		83.6			
Max.	56.0	60755.9		769.0		164.2			

7.14% of the solutions were zero regret solutions for this problem size
** : Terminated due to insufficient memory*

Table B.13: Computational results for MULPSU with $n = 15$, $m = 300$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution	UB	LB
1	41.1	1095.9	863.5	249.5	863.5	863.5	863.5
2	43.6	1092.8	7991.5	47.6	7991.5	120294.0	0.0
3	42.7	1128.8	18405.0	100.4	18405.0	18405.0	10682.8
4	38.4	1097.3	1582.7	100.9	1582.7	1582.7	1582.7
5	47.7	1144.7	20642.3	83.4	20642.3	206240.0	0.0
6	48.3	1157.8	13364.7	91.4	13364.7	132970.0	0.0
7	41.8	1121.5	7406.8	46.7	7406.8	7406.8	7406.8
8	49.3	994.7	22592.7	73.4	19254.6	64386.2	0.0
9	44.1	1057.4	9712.3	104.7	9712.3	149622.0	0.0
10	33.155	1011.126	876.6	466.200	876.6	876.6	876.6
Avg.	43.0	1090.2		136.4			
Max.	49.3	1157.8		466.2			

6.67% of the solutions were zero regret solutions for this problem size

Table B.14: Computational results for MULPSU with $n = 15$, $m = 400$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution	UB	LB
1	51.8	1410.4	10688.6	175.2	7520.3	242875.0	0.0
2	64.1	1493.5	17916.3	81.5	17916.3	299084.0	0.0
3	51.4	1450.7	4162.9	137.7	4162.9	232404.0	0.0
4	41.3	1437.6	12103.4	110.1	12103.4	12103.4	12103.4
5	60.1	1399.0	26243.2	154.0	18194.5	263782.0	0.0
6	61.7	1444.6	27138.1	126.1	27138.1	313474.0	0.0
7	54.3	1429.5	617.0	491.8	617.0	617.0	617.0
8	66.0	1379.2	12120.2	63.6	12120.2	12120.2	7742.0
9	93.3	1435.3	14726.4	201.2	12313.0	213073.0	0.0
10	53.8	1414.8	15144.2	67.3	15144.2	242291.0	0.0
Avg.	59.8	1429.5		160.8			
Max.	93.3	1493.5		491.8			

0.0% of the solutions were zero regret solutions for this problem size

Table B.15: Computational results for MULPSU with $n = 15$, $m = 500$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution	UB	LB
1	64.4	1814.8	29318.7	125.0	20806.6	297428.0	0.0
2	79.0	1816.1	17766.4	103.1	17766.4	326766.0	0.0
3	71.4	1764.0	15251.8	104.8	12126.9	398029.0	0.0
4	46.0	1729.2	4972.4	136.0	4972.4	4972.4	4972.4
5	73.0	1768.2	17206.1	120.0	17206.1	452067.0	0.0
6	64.4	1809.6	9248.3	467.1	9248.3	336854.0	0.0
7	60.5	1814.9	732.0	359.8	732.0	732.0	732.0
8	59.6	1832.4	14151.6	205.1	4748.5	541377.0	0.0
9	89.6	1745.5	41226.1	476.8	8087.1	257911.0	0.0
10	155.0	1816.4	34245.4	167.0	22524.7	359558.0	0.0
Avg.	76.3	1791.1		226.5			
Max.	155.0	1832.4		476.8			

0.0% of the solutions were zero regret solutions for this problem size

Table B.16: Computational results for MULPSU with $n = 20$, $m = 100$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	50.5	174.9	0.0	17.4	0.0
2	48.5	2959.8	1517.3	112.3	1517.3
3	51.4	2795.3	3036.7	58.6	3036.7
4	52.1	2759.8	8496.9	181.5	5742.8
5	52.5	2829.7	4312.9	115.8	4312.9
6	46.9	2908.6	181.6	148.1	181.6
7	52.7	2752.7	6408.5	207.9	6408.5
8	58.3	2782.2	1767.2	346.0	1767.2
9	45.6	2971.0	3363.8	113.0	3363.8
10	53.7	2883.0	2730.9	76.9	2730.9
Avg.	51.2	2581.7		137.7	
Max.	58.3	2971.0		346.0	

0.0% of the solutions were zero regret solutions for this problem size

Table B.17: Computational results for MULPSU with $n = 20$, $m = 200$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	84.8	5020.7	4991.5	218.3	3492.3
2	76.5	4877.2	3222.5	166.4	3222.5
3	67.5	5105.5	360.0	1803.3	360.0
4	63.0	5105.3	541.2	401.0	541.2
5	73.7	5051.8	1806.3	106.2	1806.3
6	66.7	5237.0	808.7	372.8	808.7
7	73.2	5203.1	2464.9	179.0	2464.9
8	72.6	5001.9	561.2	193.8	561.2
9	77.2	5072.3	5813.0	152.2	5813.0
10	121.6	5324.0	420.7	236.8	420.7
Avg.	77.7	5099.9		383.0	
Max.	121.6	5324.0		1803.3	

0.0% of the solutions were zero regret solutions for this problem size

Table B.18: Computational results for MULPSU with $n = 20$, $m = 300$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	135.9	7270.4	5679.0	243.5	5679.0
2	105.8	7175.9	7633.0	424.0	7633.0
3	98.4	7394.5	3723.2	257.9	3723.2
4	110.2	7426.5	16877.7	365.0	10221.8
5	100.4	7370.6	4659.3	300.4	4659.3
6	92.8	7561.1	12304.8	629.1	9978.9
7	140.7	7411.8	3906.0	136.2	3906.0
8	117.4	7371.1	6518.0	231.6	6518.0
9	121.3	7403.1	11025.5	268.3	11025.5
10	93.5	7479.2	4940.2	324.3	4940.2
Avg.	111.6	7386.4		318.0	
Max.	140.7	7561.1		629.1	

9.09% of the solutions were zero regret solutions for this problem size

Table B.19: Computational results for MULPSU with $n = 20$, $m = 400$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	119.0	9098.3	2257.3	337.3	2257.3
2	141.4	9507.9	8364.5	396.2	8364.5
3	113.1	9967.0	5297.3	398.6	5297.3
4	103.8	9551.5	8180.6	613.9	8180.6
5	141.1	9687.4	8351.3	224.0	8351.3
6	117.7	9683.0	5257.1	527.3	5257.1
7	110.4	9950.0	13696.3	573.7	13696.3
8	109.1	9679.0	20711.8	2546.7	13.0
9	124.1	9886.1	5021.5	654.4	5021.5
10	118.7	9815.7	18366.6	424.4	13158.8
Avg.	119.8	9682.6		669.6	
Max.	141.4	9967.0		2546.7	

0.0% of the solutions were zero regret solutions for this problem size

Table B.20: Computational results for MULPSU with $n = 20$, $m = 500$ and $|T| = 5$

Instance	MMCLPs	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	181.1	12009.3	16880	1027.3	16067
2	197.8	12046.2	19299	1342.0	19299
3	178.0	11787.1	10163	426.2	10163
4	124.1	11929.0	5262.4	1402.7	5262.4
5	195.1	12362.4	6761.3	501.8	6761.3
6	188.0	12557.4	14324	823.3	14324
7	125.4	14151.3	9511	582.9	5918.8
8	157.2	16072.6	10377	899.0	10377
9	157.5	12744.5	9723.9	485.3	9723.9
10	143.2	13558.5	4167.9	486.1	4167.9
Avg.	164.7	12921.8		797.7	
Max.	197.8	16072.6		1402.7	

0.0% of the solutions were zero regret solutions for this problem size

Table B.21: Computational results for MULPSU-P with $n = 5$, $m = 100$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	0.61	4.56	1.58	31.8	1.55	0.0	0.44	0.0
2	0.53	3.62	1.00	459.1	1.56	0.0	0.39	0.0
3	0.67	6.61	1.89	61.9	1.56	0.0	0.34	0.0
4	0.55	4.03	0.91	598.3	1.56	0.0	0.36	0.0
5	0.72	8.53	3.65	45.0	1.58	0.0	0.45	0.0
6	0.52	3.90	2.31	540.5	1.53	0.0	0.41	0.0
7	0.42	2.45	0.73	152.9	1.55	0.0	0.38	0.0
8	0.80	9.44	3.26	1147.0	1.47	0.0	0.58	0.0
9	0.50	3.92	1.69	334.4	1.45	0.0	0.36	0.0
10	0.62	5.01	3.12	509.9	1.44	0.0	0.53	0.0
Avg.	0.59	5.21	2.01		1.52		0.42	
Max.	0.80	9.44	3.65		1.58		0.58	

66.67% of the solutions were zero regret solutions for this problem size

Table B.22: Computational results for MULPSU-P with $n = 5$, $m = 200$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	1.17	12.92	2.54	19.1	2.90	0.0	0.81	0.0
2	0.66	6.23	1.78	457.9	2.79	0.0	0.75	0.0
3	1.36	13.17	5.37	3079.5	3.03	0.0	0.98	0.0
4	1.47	15.24	5.21	11446.5	3.04	0.0	0.75	0.0
5	1.83	17.08	16.13	812.1	3.03	0.0	0.95	0.0
6	1.98	34.90	19.33	4662.6	2.70	0.0	0.98	0.0
7	1.05	7.44	1.73	178.1	2.76	0.0	0.87	0.0
8	2.09	29.75	10.59	426.9	2.79	0.0	0.87	0.0
9	1.34	12.00	2.98	4373.3	3.07	0.0	0.91	0.0
10	2.00	19.13	12.64	4677.4	2.79	0.0	1.20	0.0
Avg.	1.49	16.78	7.83		2.89		0.91	
Max.	2.09	34.90	19.33		3.07		1.20	

67.74% of the solutions were zero regret solutions for this problem size

Table B.23: Computational results for MULPSU-P with $n = 5$, $m = 300$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	3.37	46.71	27.96	351.0	4.45	0.0	1.69	0.0
2	1.56	14.38	4.84	525.4	4.17	0.0	1.31	0.0
3	1.36	13.49	5.51	410.2	4.74	0.0	1.19	0.0
4	1.61	12.90	3.45	124.0	4.62	0.0	1.12	0.0
5	2.22	18.21	6.99	8184.4	4.59	0.0	1.44	0.0
6	0.81	5.62	1.97	14.3	4.28	0.0	1.17	0.0
7	2.39	27.91	13.10	8749.5	4.35	0.0	1.97	0.0
8	1.36	13.26	6.44	591.6	4.12	0.0	1.23	0.0
9	2.09	21.28	4.31	374.9	4.13	0.0	1.23	0.0
10	1.01	12.89	2.23	286.1	4.20	0.0	1.12	0.0
Avg.	1.78	18.66	7.68		4.36		1.35	
Max.	3.37	46.71	27.96		4.74		1.97	

68.75% of the solutions were zero regret solutions for this problem size

Table B.24: Computational results for MULPSU-P with $n = 5$, $m = 400$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	6.04	54.12	25.97	10634.6	5.73	0.0	1.69	0.0
2	4.60	38.02	13.67	2347.2	5.98	0.0	1.87	0.0
3	7.85	84.33	40.55	15026.5	6.10	0.0	2.61	0.0
4	4.71	53.51	52.99	14828.1	5.73	0.0	2.57	0.0
5	2.34	26.11	3.62	363.4	5.77	0.0	1.73	0.0
6	5.04	31.12	21.86	8114.8	6.23	0.0	2.12	0.0
7	4.20	63.99	6.86	90.9	6.13	0.0	1.86	0.0
8	3.73	34.41	14.74	24859.0	5.87	0.0	2.56	0.0
9	5.79	34.96	18.97	5993.2	5.63	0.0	2.48	0.0
10	1.75	20.64	13.68	12168.5	5.91	0.0	2.26	0.0
Avg.	4.60	44.12	21.29		5.91		2.17	
Max.	7.85	84.33	52.99		6.23		2.61	

62.96% of the solutions were zero regret solutions for this problem size

Table B.25: Computational results for MULPSU-P with $n = 5$, $m = 500$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap
1	3.32	54.69	8.77	189.2	7.60	0.0	2.03	0.0
2	1.54	15.12	3.01	261.5	7.54	0.0	2.06	0.0
3	4.21	34.66	11.23	68.5	7.66	0.0	2.56	0.0
4	2.53	35.19	16.40	15543.7	7.52	0.0	2.17	0.0
5	6.75	63.12	24.24	29176.3	7.50	0.0	2.59	0.0
6	2.89	25.23	11.29	10454.2	6.88	0.0	2.79	0.0
7	4.34	40.50	22.95	12974.8	7.75	0.0	2.98	0.0
8	2.79	42.74	6.93	37.5	7.71	0.0	2.03	0.0
9	5.38	147.03	27.99	24700.7	7.46	0.0	1.61	0.0
10	12.42	80.82	27.86	23865.5	7.77	0.0	2.26	0.0
Avg.	4.62	53.91	16.07		7.54		2.31	
Max.	12.42	147.03	27.99		7.77		2.98	

73.68% of the solutions were zero regret solutions for this problem size

Table B.26: Computational results for MULPSU-P with $n = 10$, $m = 100$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX			TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CP-I (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap	UB	LB
1	26.6	3687.1	327.2	280.6	70.5	0.0	9.6	0.0	-	-
2	19.7	1273.1	570.6	7.3	73.6	0.0	17.5	0.0	-	-
3	28.2	8143.0	670.1	3908.7	68.7	0.0	5.5	0.0	-	-
4	27.0	20423.5	1572.2	3262.5	71.6	0.0	8.1	0.0	-	-
5	34.2	7328.3	9368.5	1920.6	63.7	0.0	10.6	0.0	1920.6	1687.8
6	31.7	101723.1	53292.3	10066.3	71.1	0.0	18.2	0.0	86901.1	4551.1
7	30.3	8475.3	5890.8	8496.6	68.4	0.0	12.8	0.0	86255.4	4943.7
8	35.1	8538.1	2892.3	5663.0	70.3	0.0	22.0	0.0	-	-
9	23.4	1675.9	662.2	10.0	70.1	0.0	19.7	0.0	-	-
10	33.5	65262.1	36477.5	5097.5	66.5	0.0	13.7	0.0	54123.9	2101.2
Avg.	29.0	22652.9	11172.4		69.5		13.8			
Max.	35.1	101723.1	53292.3		73.6		22.0			

0.0% of the solutions were zero regret solutions for this problem size

Table B.27: Computational results for MULPSU-P with $n = 10$, $m = 200$ and $|T| = 5$

Instance	MMCLP-Ps	CPLEX		TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CP-II (s)	solution	CPU (s)	gap	CPU (s)	gap	UB	LB
1	45.6	19497.5	11272.2	140.8	0.0	17.8	0.0	11272.2	5933.2
2	69.6	3814.1	2227.8	148.1	0.0	28.3	0.0	2227.8	2000.8
3	51.8	8197.3	11591.5	141.5	0.0	25.8	0.0	155530.0	7243.1
4	24.2	777.1	540.0	147.3	0.0	36.2	0.0	-	-
5	46.7	2353.1	823.3	139.5	0.0	25.7	0.0	-	-
6	47.9	12415.0	6862.4	144.5	0.0	27.0	0.0	176669.0	3999.9
7	22.1	599.6	3079.5	129.4	0.0	22.7	0.0	-	-
8	46.3	27674.7	2901.2	137.5	0.0	32.1	0.0	81697.7	0.0
9	40.7	7930.1	4972.4	131.6	8939.8	21.5	0.0	121496.0	0.0
10	35.0	6055.8	2832.5	135.8	0.0	24.1	0.0	2832.5	1667.0
Avg.	43.0	8931.4		139.6		26.1			
Max.	69.6	27674.7		148.1		36.2			

0.0% of the solutions were zero regret solutions for this problem size

Table B.28: Computational results for MULPSU-P with $n = 10$, $m = 300$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II		CP-II (1 hr)	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution	UB	LB
1	96.00	227.52	30305.1	41.93	13040.9	134513.0	0.0
2	103.96	226.65	11574.0	51.78	11574.0	108453.0	0.0
3	123.08	234.00	18321.9	35.29	17891.7	134564.0	0.0
4	73.26	257.39	16340.6	50.79	16340.6	236152.0	0.0
5	115.82	206.47	11633.1	26.26	11633.1	124797.0	0.0
6	122.94	217.14	10260.9	51.56	10260.9	113046.0	0.0
7	70.51	223.12	12196.8	20.45	12196.8	197000.0	11625.9
8	108.30	213.03	10965.5	25.60	10965.5	82236.6	0.0
9	41.18	220.60	44.9	42.03	44.9	160197.0	0.0
10	70.36	219.74	14723.4	28.58	14723.4	133775.0	0.0
Avg.	92.54	224.56		37.43			
Max.	123.08	257.39		51.78			

9.09% of the solutions were zero regret solutions for this problem size

Table B.29: Computational results for MULPSU-P with $n = 10$, $m = 400$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	103.34	284.40	13199.7	51.91	13199.7
2	148.02	302.10	13922.7	67.85	13922.7
3	136.23	299.19	23551.0	56.19	23551.0
4	156.90	304.05	10634.6	68.80	10634.6
5	219.88	304.33	9475.6	48.05	9475.6
6	192.54	304.50	15026.5	70.03	15026.5
7	144.11	306.61	39187.1	51.27	30696.0
8	231.14	308.66	12520.3	60.87	12520.3
9	176.89	295.62	20226.0	41.56	20226.0
10	162.82	312.54	12807.7	38.65	12807.7
Avg.	167.19	302.20		55.52	
Max.	231.14	312.54		70.03	

16.67% of the solutions were zero regret solutions for this problem size

Table B.30: Computational results for MULPSU-P with $n = 10$, $m = 500$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	255.75	377.78	55221.4	113.01	55221.4
2	286.49	403.03	21287.3	68.03	21287.3
3	213.75	398.37	20978.7	83.95	20978.7
4	225.36	408.13	2168.2	56.19	2168.2
5	242.80	413.80	5692.6	144.16	5692.6
6	191.46	382.08	18005.9	120.08	18005.9
7	232.73	391.53	10452.8	52.71	10452.8
8	220.73	388.84	5205.0	47.95	5205.0
9	233.83	399.68	2334.0	39.77	2334.0
10	309.98	376.02	42478.8	114.36	42478.8
Avg.	241.29	393.92		84.02	
Max.	309.98	413.80		144.16	

23.08% of the solutions were zero regret solutions for this problem size

Table B.31: Computational results for MULPSU-P with $n = 15$, $m = 100$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	85.88	810.42	5162.9	71.21	5162.9
2	128.55	854.27	4025.6	59.17	4025.6
3	142.92	749.34	4336.9	63.40	4336.9
4	82.59	835.47	1478.7	45.26	1478.7
5	124.15	732.61	10212.4	78.42	10212.4
6	174.51	836.68	12137.0	99.22	10131.6
7	147.92	835.61	12321.4	123.30	11693.0
8	147.50	853.87	16028.0	60.97	3226.9
9	124.08	863.61	9443.1	103.30	9443.1
10	99.89	934.99	2686.5	88.13	2686.5
Avg.	125.80	830.69		79.24	
Max.	174.51	934.99		123.30	

9.09% of the solutions were zero regret solutions for this problem size

Table B.32: Computational results for MULPSU-P with $n = 15$, $m = 200$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	502.30	1638.98	21085.2	149.28	21085.2
2	422.29	1827.40	23183.0	195.42	23183.0
3	285.64	1671.15	4635.7	181.97	4635.7
4	193.17	1884.22	6605.7	87.23	6605.7
5	336.53	1737.57	7287.6	137.88	7287.6
6	268.12	1719.45	6434.5	118.13	6434.5
7	337.44	1548.26	11732.4	100.95	11732.4
8	739.90	1717.86	8372.1	302.92	8372.1
9	377.48	1700.91	10936.2	109.18	7480.8
10	336.46	1698.24	21227.4	217.35	2832.5
Avg.	379.93	1714.40		160.03	
Max.	739.90	1884.22		302.92	

0.0% of the solutions were zero regret solutions for this problem size

Table B.33: Computational results for MULPSU-P with $n = 15$, $m = 300$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	844.08	2463.13	16711.6	217.92	16563.4
2	996.94	2463.47	24973.9	312.66	23362.5
3	723.00	2607.80	13450.2	234.61	13450.2
4	618.87	2600.26	19648.8	229.89	19648.8
5	388.80	2401.23	43670.1	121.73	2890.1
6	599.18	2405.60	13262.0	212.25	13262.0
7	903.60	2329.00	35707.6	374.07	13293.4
8	459.31	2693.89	15356.1	218.24	15356.1
9	525.35	2283.49	10250.7	148.62	10250.7
10	700.19	2645.39	18249.3	288.29	18249.3
Avg.	675.93	2489.32		235.83	
Max.	996.94	2693.89		374.07	

0.0% of the solutions were zero regret solutions for this problem size

Table B.34: Computational results for MULPSU-P with $n = 15$, $m = 400$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	905.80	3292.49	18523.9	507.73	13706.7
2	862.07	3589.47	38724.2	601.84	38724.2
3	873.22	3670.19	18023.2	420.87	18023.2
4	858.67	3500.03	22114.1	481.13	22114.1
5	813.59	3369.38	38087.8	351.27	20806.4
6	932.04	3478.80	12505.6	390.33	12505.6
7	1361.29	3325.18	19116.4	378.19	19116.4
8	915.13	3266.16	37372.0	357.05	37372.0
9	1101.79	3477.17	14657.5	486.58	14657.5
10	1133.92	3227.07	20226.0	284.19	20226.0
Avg.	975.75	3419.59		425.92	
Max.	1361.29	3670.19		601.84	

0.0% of the solutions were zero regret solutions for this problem size

Table B.35: Computational results for MULPSU-P with $n = 15$, $m = 500$ and $|T| = 5$

Instance	MMCLP-Ps	TS-I		TS-II	
	CPU (s)	CPU (s)	Solution	CPU (s)	Solution
1	1395.64	4379.83	55221.4	491.16	53670.4
2	1410.20	4664.50	9142.0	475.60	9142.0
3	1471.66	4213.80	13426.1	299.38	13426.1
4	1693.78	4432.52	26155.8	424.53	21214.9
5	1156.74	4473.56	9824.0	452.28	9824.0
6	1205.34	4270.90	52971.8	280.40	33889.3
7	1171.62	4471.51	16802.1	313.46	16802.1
8	1413.57	4153.49	27899.4	426.02	27899.4
9	1517.23	4374.83	19117.5	472.70	19117.5
10	1127.27	4140.86	68280.1	310.38	21040.6
Avg.	1356.31	4357.58		394.59	
Max.	1693.78	4664.50		491.16	

9.09% of the solutions were zero regret solutions for this problem size

References

Albareda-Sambola, M., Fernández, E., & Saldanha-da Gama, F. (2011). The facility location problem with bernoulli demands. *Omega*, *39*(3), 335–345.

Antunes, A., & Peeters, D. (2000). A dynamic optimization model for school network planning. *Socio-Economic Planning Sciences*, *34*(2), 101–120.

Antunes, A., & Peeters, D. (2001). On solving complex multi-period location models using simulated annealing. *European Journal of Operational Research*, *130*(1), 190–201.

Arabani, A., & Farahani, R. (2011). Facility location dynamics: An overview of classifications and applications. *Computers & Industrial Engineering*.

Averbakh, I. (2003). Complexity of robust single facility location problems on networks with uncertain edge lengths. *Discrete Applied Mathematics*, *127*(3), 505–522.

Berman, O., Drezner, Z., & Krass, D. (2010). Generalized coverage: New developments in covering location models. *Computers & Operations Research*, *37*(10), 1675–1687.

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

Berman, O., & Wang, J. (2011). The minmax regret gradual covering location problem on a network with incomplete information of demand weights. *European Journal of Operational Research*, *208*(3), 233–238.

Brotcorne, L., Laporte, G., & Semet, F. (2003). Ambulance location and relocation models. *European journal of operational research*, *147*(3), 451–463.

Canel, C., Khumawala, B., Law, J., & Loh, A. (2001). An algorithm for the capacitated, multi-commodity multi-period facility location problem. *Computers & Operations Research*, *28*(5), 411–427.

Chung, C.-H. (1986). Recent applications of the maximal covering location planning (MCLP) model. *Journal of the Operational Research Society*, 735–746.

Church, R., & Meadows, M. E. (1979). Location modeling utilizing maximum service distance criteria. *Geographical Analysis*, *11*(4), 358–373.

Church, R., & ReVelle, C. S. (1974). The maximal covering location problem. *Papers in regional science*, *32*(1), 101–118.

Church, R., & Roberts, K. L. (1983). Generalized coverage models and public facility location. In *Papers of the regional science association* (Vol. 53, pp. 117–135).

Current, J., Ratick, S., & ReVelle, C. (1998). Dynamic facility location when the total number of facilities is uncertain: A decision analysis approach. *European Journal of Operational Research*, *110*(3), 597–609.

Daskin, M. (2000). A new approach to solving the vertex p-center problem to optimality: Algorithm and computational results. *Communications of the Operations Research Society of Japan*, *45*(9), 428–436.

Dias, J., Captivo, M., & Clímaco, J. (2008). A memetic algorithm for multi-objective dynamic location problems. *Journal of Global Optimization*, *42*(2), 221–253.

Dias, J., Captivo, M. E., & Clímaco, J. (2006). Capacitated dynamic location problems with opening, closure and reopening of facilities. *IMA Journal of Management Mathematics*, *17*(4), 317–348.

Dias, J., Eugénia Captivo, M., & Clímaco, J. (2007). Efficient primal-dual heuristic for a dynamic location problem. *Computers & Operations Research*, *34*(6), 1800–1823.

Drezner, Z., & Hamacher, H. (2001). *Facility location: Applications and theory.* Springer.

Erlenkotter, D. (1981). A comparative study of approaches to dynamic location problems. *European Journal of Operational Research*, *6*(2), 133–143.

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

Gendreau, M., Laporte, G., Musaraganyi, C., & Taillard, É. D. (1999). A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, *26*(12), 1153–1173.

Ghaderi, A., & Jabalameli, M. (2012). Modeling the budget-constrained dynamic uncapacitated facility location-network design problem and solving it via two efficient heuristics: A case study of health care. *Mathematical and Computer Modelling*.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, *1*(3), 190–206.

Glover, F. (1990). Tabu searchpart ii. *ORSA Journal on computing*, *2*(1), 4–32.

Gunawardane, G. (1982). Dynamic versions of set covering type public facility location problems. *European Journal of Operational Research*, *10*(2), 190–195.

Karasakal, O., & Karasakal, E. K. (2004). A maximal covering location model in the presence of partial coverage. *Computers & Operations Research*, *31*(9), 1515–1526.

Killmer, K. A., Anandalingam, G., & Malcolm, S. A. (2001). Siting noxious facilities under uncertainty. *European Journal of Operational Research*, *133*(3), 596–607.

Lawn, J. E., Rohde, J., Rifkin, S., Were, M., Paul, V. K., & Chopra, M. (2008). Alma-ata 30 years on: Revolutionary, relevant, and time to revitalise. *The Lancet*, *372*(9642), 917–927.

Melo, M. T., Nickel, S., & Saldanha da Gama, F. (2006). Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning. *Computers & Operations Research*, *33*(1), 181–208.

Rohde, J., Cousens, S., Chopra, M., Tangcharoensathien, V., Black, R., Bhutta, Z. A., & Lawn, J. E. (2008). 30 years after alma-ata: Has primary health care worked in countries? *The Lancet*, *372*(9642), 950–961.

Saldanha da Gama, F., & Captivo, M. (1998). A heuristic approach for the discrete dynamic location problem. *Location Science*, *6*(1), 211–223.

Schilling, D. A., Jayaraman, V., & Barkhi, R. (1993). A review of covering problems in facility location. *Computers & Operations Research*.

Taillard, E. (1990). Some efficient heuristic methods for the flow shop sequencing problem. *European Journal of Operational Research*, *47*(1), 65–74.

Taillard, E. (1991). Robust taboo search for the quadratic assignment problem. *Parallel computing*, *17*(4), 443–455.

Taillard, E. (1995). Comparison of iterative searches for the quadratic assignment problem. *Location science*, *3*(2), 87–105.

Van Roy, T., & Erlenkotter, D. (1982). A dual-based procedure for dynamic facility location. *Management Science*, *28*(10), 1091–1105.

Walley, J., Lawn, J. E., Tinker, A., De Francisco, A., Chopra, M., Rudan, I., . . . Black, R. E. (2008). Primary health care: Making alma-ata a reality. *The Lancet*, *372*(9642), 1001–1007.

Warszawski, A. (1973). Multi-dimensional location problems. *Operational Research Quarterly*, 165–179.

Wesolowsky, G., & Truscott, W. (1975). The multiperiod location-allocation problem with relocation of facilities. *Management Science*, *22*(1), 57–65.

Zarandi, M. H. F., Davari, S., & Sisakht, S. A. H. (2013). The large-scale dynamic maximal covering location problem. *Mathematical and Computer Modelling*, *57*(3), 710–719.