GENETIC AND HEURISTIC ALGORITHMS FOR FACILITY LOCATION PROBLEM

THESIS

Submitted in partial fulfillment of the requirements

for the award of degree of

M. Tech. (CSA)

Submitted By

Deepanshu Goyal

Roll No.- 601103004

Supervised By

Dr. Mahesh Kumar Sharma
Associate Professor

Mr. Singara Singh
Assistant Professor

School of Mathematics and Computer Applications
THAPAR UNIVERSITY
PATIALA – 147004

JULY 2013

CERTIFICATE

I hereby certify that the work which is presented in the thesis entitled, **"Genetic And Heuristic Algorithms For Facility Location problem"** in partial fulfillment of the requirements for the award of degree of the Master of Technology in **Computer Science and Applications,** submitted in School of mathematics and Computer Applications of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Mahesh Kumar Sharma, Mr. Singara Singh** and refers others researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other university.

Deepanshu Goyal
601103004

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Dr Mahesh Kumar Sharma
Associate Professor

15.7.13
Mr. Singara Singh
Assistant Professor

Countersigned by

Dr. Rajesh Kumar
Head
School of Mathematics and Computer Applications
Thapar University
Patiala

Dr. S.K. Mohapatra
Dean Academic Affairs
Thapar University
Patiala

Acknowledgement

First of all I would like to thank the Almighty, who has always guided me to work on the right path of the life.

This work would not have been possible without the encouragement and able guidance of my supervisors Dr. M.K. Sharma and Mr. Singara Singh. I thank my supervisors for their time, patience, discussions and valuable comments. Their enthusiasm and optimism made this experience both rewarding and enjoyable.

I am equally grateful to Dr. Rajesh Kumar, Associate Professor and Head, School of Mathematics and Computer Applications, for motivation and inspiration that triggered me for the thesis work.

I will be failing in my duty if I don't express my gratitude to Dr. S. K. Mohapatra, Senior Professor and Dean of Academic Affairs Thapar University, for making provisions of infrastructure such as library facilities, computer labs equipped with net facilities, immensely useful for the learners to equip themselves with the latest in the field.

I am also thankful to the entire faculty and staff members of School of Mathematics and Computer Applications Department for their direct-indirect help, cooperation, love and affection, which made my stay at Thapar University memorable.

Last but not least, I would like to thank my parents for their wonderful love and encouragement, without their blessings none of this would have been possible.

ABSTRACT

Facility location problem is a branch of operations research concerning itself with solutions of problem concerning the placement of facilities in order to minimize transportation costs, avoid placing hazardous materials near housing, outperform competitor's facilities, etc. In a simple facility location problem, a single facility is to be placed, with the only optimization criterion being the minimization of the sum of distances from a given set of point sites. More complex problem considered in this discipline includes the placement of multiple facilities, constraints on the locations of facilities, and more complex optimization criteria. It is basically a facilities location problem which includes assignment of clients to the selected number of locations from among the given number of potential locations over a different time periods, for their requirements but with the objectives of minimizing cost for satisfying demands of all clients and minimizing the maximum time needed to fulfill the requirements of all the clients along with some constraints.

The present thesis contains three chapters: The first chapter is introductory in nature in which multi objective optimization, heuristic, genetic algorithm is briefly discussed. In second chapter, a warehouse location problem considered by prakash *et al* [22] is reviewed and a genetic algorithm has been proposed to solve this problem. In third chapter, the concept of a single objective dynamic facility location problem considered by Dias *et al* [4] is introduced to a multi objective warehouse problem and an algorithm proposed by prakash *et al* [22] is modified to find the set of efficient solutions.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER-1

INTRODUCTION AND LITERATURE SURVEY

1.1 Facility location Problem

Facility location is a branch of operation research also known as location analysis that concerning itself with mathematical modeling and find solution of problems that concern the placement of given facilities in order to minimize transportation time, cost, avoid placing hazardous materials near housing, *etc*. The simple facility location problem deal with a single facility is to be placed, with the one optimization criterion being the minimization of the transportation cost from a given set of point sites. Multiple facilities location problem considered more complex problems in this discipline that include placement of multiple facilities to the desired location, constraints on the locations have to be maintained and more complicated optimization criteria. The present thesis deals with a real life scenario of facility location problem which is basically a warehouse location problem which includes assignment of given numbers of ration shops to the selected number of allocation sites from among the given number of potential sites, to fulfill their requirements but with the objectives of minimizing transportation cost for satisfying demands of all allocated ration shops and minimizing the maximum transportation time needed to fulfill the requirements of all the ration shops along with several constraints.

1.2 Multi-Objective Optimization

The mathematical objective model that has more than one objective with several different constraints is known as Multi-Objective model. There are two most important or crucial aspects in the multi-objective that has to be considered: the concepts and terminology that are most important to describe and understand the process and the steps employed to formulate the multi-objective model. There are many factors that play an important role in the understanding and analysis of the multi-objective model.

Multi-objective optimization is defined as:

"A vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions. Hence, the term "optimize" means finding such a solution which would give the values of all the objective functions acceptable to the designer/decision maker."

This model can be formulated as:

Optimize $f(x) = (f_1(x), f_2(x), \ldots\ldots, f_k(x))$

Subject to

$$g_j(x) \leq, =, \geq b_j \qquad , j=1,2,3,\ldots\ldots\ldots,m$$

$$X=(x_1,x_2,\ldots\ldots,x_n)^T \qquad , X \geq 0$$

where $f(x)$ is the objective function that is to be optimized. $f_1(x), f_2(x),\ldots\ldots,f_k(x)$ are K number of different objective functions subject to m constraints. X is a vector consists of decision variables $x_1,x_2,x_3,\ldots\ldots,x_n$.

In multi-objective linear programming, there is more than one objective model and associated a solution technique with it from which one may choose. Some of the previous investigators who recognized this drawback of the single-objective approach suggested a rather brute-force approach to circumvent but not to actually resolve the problem. They have suggested that first make list of all the problem objectives and then pick one of these objectives as the "single objective" in the traditional model performed and consider all the other objectives to be rigid constraints. After that resultant model and to solve it repeat this procedure with next remaining objective as the single objective in the model and at the end pick that solution from all solutions obtained, that "looks most promising" or "best" satisfy all the objectives and constraints. There are few drawbacks to this approach, first, it converts the problem into a possibly huge number of linear programming problems one for each objective which have to be evaluated or solved and second and most crucial is that the resultant solution as achieved at the end may very well not depict that solution that best and satisfies all the objectives. It is obvious that something far more systematic approach is required.

The prioritizing methods or ranking try to ignore the heady problems. Its rather than attempting to find a numerical weight for each objective for the given problem, they simply ask that objectives be ranked according to their perceived priority. Often, most decision makers can do this and, in fact, ranking is an idea that looks inherent too much of decision making. Let's consider one example, in determining raises, a procedure used by many organizations is to first rank their employees in order of their capability, experience, productivity, value to the company, and so on. The basic problem with the ranking approach is how to associate the results of a given solution to the satisfaction of the ranking.

Efficient solution method "circumvents" both the drawbacks of finding weights and that of satisfying the ranking. Efficient solution does this, or at least an attempt to, by generating the set of all efficient solutions those are also called as Pareto optimal solutions or non dominated solutions. Efficient solution approach looks more promising and efficient among the above two that have been discussed.

1.3 Non-Dominated Solutions or Efficient Solutions

A set of solutions is said to be efficient or non-dominated if there exist no solution that is superior to that with respect to at least one objective function but is not inferior to it with respect to any of the objective functions.

If x_1 and x_2 are two solutions, then in this case there can be any of two possibilities-one dominates the other or non-dominates the other. If the problem is of minimization then, without the loss of generality, a solution x_1 dominates x_2 if and only if the following two conditions are satisfied :

$$\forall i \in \{1,2,.....,N_{obj}\}: f_i(x_1) \leq f_i(x_2)$$

$$\exists j \in \{1,2,.....,N_{obj}\}: f_j(x_1) \leq f_j(x_2)$$

where $f(x_1)$ and $f(x_2)$ are the objectives.

If any of the above conditions is violated, then the solution x_1 does not dominate the solution x_2.

But if x_1 dominates the solution x_2 then solution x_1 is called the non-dominated solution within the set $\{x_1, x_2\}$. From the entire set of efficient or non-dominated solutions the decision maker have to select the solution he believed most attractive and promising.

1.4 Heuristics

The word "heuristic" originated from the Greek root meaning to discover. In case of optimization of real life problems, the problems are encountered because of the highly complex nature. The regular algorithms are very often ineffective on these problems but there are, however, other approach that may be used to find a solution to models involving integer and or discrete variables. And, in many cases, such approaches have proven capable of providing acceptable solutions to truly massive size problems. Thus, the pragmatic or human approach can be considered to solve the problems and hence this

shifts the whole paradigm from algorithm-based calculations to the employment of heuristic procedures or heuristic programming.

Heuristics are rules of thumb that are developed through intuition, experience and judgment. In artificial intelligence, a heuristic is a procedure that may lack a proof. It is used when the inter-relationships of the decision variables are not explicitly clear but there is some confidence in understanding the output for certain input. The result of application of heuristics cannot guarantee the optimal solution. A heuristic might help to find solutions which are good, but perhaps not the very best they can be. Obviously the measure of goodness and assessment of a heuristic technique is going to be relative to the domain, and to the specific job that problem solving is going to be applied in that domain. When one or more heuristics are combined with a procedure for deriving a solution from the associated rules, gives a heuristic program. Since the concept is to derive an acceptable solution and not the server-elusive optimal solution, thus the heuristics satisfy certain aspiration criteria as set by the decision maker.

The heuristic approaches are also being used to solve the real world multi-objective problems. These problems always attract us towards the goal programming techniques but of late another approach of the efficient/non-dominated/Pareto optimal solutions has gained importance.

1.5 Tabu Search

Tabu search is a meta-heuristic approach which is associated with a method of seeking optimal solutions. But since Hence, *Tabu* search guides local heuristic search procedures to explore the solution space beyond local optimality. The important characteristics of Tabu search are:

1. It allows moves, which even worsen the value of the objective function.
2. It prevents cycling temporarily, i.e., temporarily revisiting solutions examined recently by using Tabu list. Tabu list contains forbidden moves having been selected recently. The Tabu list changes as move to the next solution whether it is better or worse.
3. It uses incumbent solution which is the best solution obtained so far. The incumbent solution changes to better solution as and when arrive at it.
4. The procedure terminates when revisit an already examined solution.

1.6 Genetic Algorithm

Genetic is the “Evolutionary Computing” was introduced in the 1960s and the concept is developed by I. Rechenberg. Genetic Algorithm (GA) is a family of computational models inspired by development. GA is the searching algorithm that enciphers a possible solution to a specific problem on a simple chromosome like data structure and it applies recombination operators to these structures as to preserve crucial and important information. Usually the GA is often viewed as function optimizer, although the ranges of problem to which GA have to be applied are very large.

From the implementation point of view the GA begins with a initial population of (generally random) chromosomes. As the algorithm precedes one calculate these structures and allocated reproductive opportunities in such a way so that these chromosomes which look to give a better solution for the objective problem are given more chances to `reproduce' next generation and those chromosomes which look poor solution are not allowed to reproduce the next generation.

Generally human new inventions were inspired by nature and our sounding. The neural networks that are the branch of artificial inelegance and widely used, is one example. The GA are the another example. GA can be used in searching by simulating evolution, starting from an initial set of hypotheses or, and generating the next "generations" of solutions for given problem. This algorithm that is widely used in AI was inspired by the way human being evolved into more successful organisms in nature and grows. As the nature teaches us fittest individual in the generation will survive and reproduce the next generation and this phenomena is called natural selection. A chromosome is a complex, long and complicated thread of a DNA that is also called deoxyribonucleic acid.

The length of chromosomes is determined by hereditary factors those particular traits of an individual. Each trait of chromosomes is coded by some combination of DNA i.e. A (Adenine), C (Cytosine), G (Guanine) and T (Thymine). These are similar to an alphabet in a language and these meaningful combinations of the bases produce particular instructions to the cell. The generation will change during reproduction. The chromosomes of the entire living organism exchange randomly from the parents by a process called crossover. Thus we inherited some traits of mother and some traits father that is the natural phenomena. A process that is rare called mutation and it also changes some traits, during copying of chromosomes an error may occur sometime. GA

implements “Optimization Strategies” by simulating evolution of Species through natural selection process.

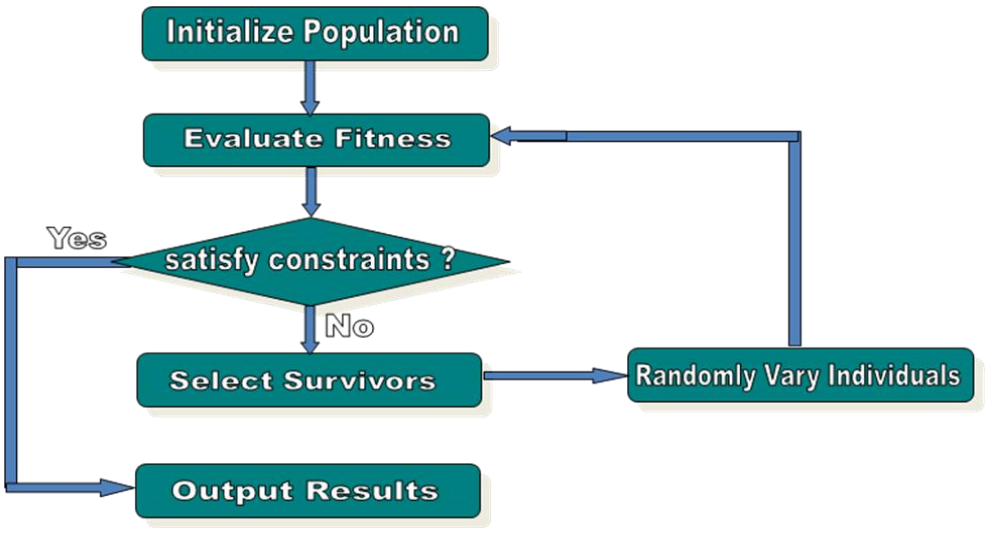

Figure 1.1: Flow chart for GA.

The above figure is showing the working of the GA. The GA first randomly generates the initial population or it can be given by user. The initial population is encoded in to binary form and then it evaluates the fitness function. The fitness function will decide fitness value for objective the function. If it is a minimization problem then, the minimum value of the fitness function will be taken as best solution and if the objective function is to be maximized then, the maximum value will be sleeted as the best solution. After applying crossover and mutation and get the new next generation of p

GA starts with initial population of a random string that represents decision or design variables. The initial population is first encoded in binary form that is the combination of 0 and 1after that population is then operated by three main operators; reproduction, crossover and mutation to create a new set of population that is also called the next generation. GA can be viewed as trying to minimize the fitness function that can be done by calculating several solution vectors. These operators are to create new population by selection, combination or alteration of the current population that has shown to be most promising solutions, that is why these operators are used. The new population is again evaluated and tested till termination take place. If the termination criterion is not satisfied then the current population is iteratively operated by the above operators and evaluated.

The above steps of the GA are repeated until the termination criterion is satisfied. After applying the one cycle of the above discussed operations and the subsequent evaluation procedure is known as a generation in Genetic terminology. All the operators that are used by the GA are described in the following steps.

The first set of individual of the generation, those will generate the first new generation is called *Initial Population*. Represent each individual in the form of string of binary, called *encoding* in GA and by using the binary representation of population it will be more easy to apply genetic operator on those representation so that the next generation to produce. The process of representing the solution in the form of a string that conveys the necessary information.

To provide the population diversity and the selection individuals from the initial population is the responsibility of the selection process in GA and these are the two most important factors in the genetic search . There is the tradeoff between selective pressure and diversity but they are strongly related to each other, since an increase in the selective pressure decreases the population diversity and if increase the diversity decreases the selective pressure. If the population becomes too homogeneous or there is less diversity in the population mutation will almost be the only factor that provides diversity or variation in the population. Therefore, it is very crucial to make the correct choice when GA determining a selection process.

When selecting individuals for both parents reproducing the same generation then selection process will become very important to apply. There are some processes available for selection method and they all inherited to natural selection process of the nature. Nature follows fittest to survive theory and fittest individual is more likely to reproduce healthy generation. It has to be explained how the selective pressure convergences the algorithm before discussing those methods of selection. Selection process replicates the most successful solutions found in a population at a rate proportional to their relative quality

One of the most important genetic operators is crossover [9]. Crossover implements a reproduction between two organisms from the parents (taken from the initial population). Before implementing it understand the working of this operator that works on a pair of solutions, recombines them in certain way generating one or more new individuals in the

generation and that inherited some of the characteristics of their parents and in that way the characteristic are passed on to the future generations.

In GA, the functionality of the crossover depends on the data representation. There are numbers of crossover different crossover operators have been introduced. To demonstrate the working of simple crossover operators is illustrated in figure 1.2. The example shows the working of Simple crossover that is used the binary representation of the generation, even though it will not be used further in this thesis.

Example: In Simple Crossover, the procedure starts with two parent solutions P1 and P2.

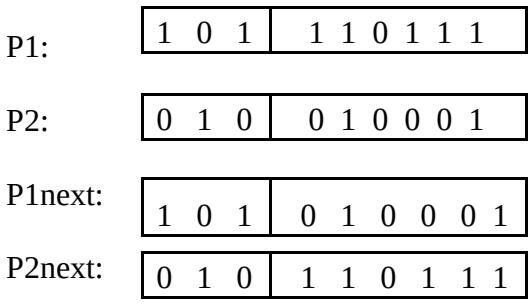

Figure 1.2: Illustration of Simple Crossover

The line between P1 and P2 demonstrates the cut in which the crossover value is randomly generated that is 3. After the crossover operation the new generation is represented by P1next and P2next respectively.

Crossover is the process in which two chromosomes those are represented by the strings of binary values combine their genetic material (bits) to produce a new string for the population and possesses both their characteristics.

1. Two strings are picked from the mating pool at random to cross over.
2. The method chosen depends on the Encoding Method.

GA contains another operator that is known as mutation. Mutation is applied to a single solution with a certain probability. Mutation makes small random changes in the solution and is responsible for diversity in the generation. These random changes will gradually change some new characteristics to the population. These new characteristics could not be developed by the crossover process.. The simple example of the mutation operator is shown in the figure 1.3.

A: 1 1 0 **0** 1 1 0

B: 1 1 0 **1** 1 1 0

Figure 1.3: Illustration of a simple mutation.

In the shown example, A represents the binary form of the population before apply the mutation process and B represents the binary form of the population after the mutation process. In the process of mutation bit are generated randomly in the same way as generated in the crossover thus, bit 4 has been changed from 0 to 1 in the string A. Mutation location or position randomly generated that is 4, this mutation is called 1 bit mutation. The binary data string A represents a parent solution. Randomly, the fourth bit has been chosen to be mutated. The resulting string is stored in B.

1.7 Literature Survey

Wesolowsky [28] was one of the first investigators who studied the dynamic location problem. He generalizes the Weber problem in his paper, considering the existence of several time periods. Wesolowsky and Truscott [29] describe a discrete location problem considering a fixed number of open facilities in each time period in his paper. They describe a new and efficient method that is known as resolution method based on dynamic programming.

Fong and Srinivasan [15, 20] studied the problem of determining a schedule of capacity expansions of facilities over a planning phase by determining size, the location, and timing of construction of facilities at potential site. They presented a heuristic approach that tries to improve a feasible solution by exchanging capacities between pairs of area.

Erlenkotter [6] purposed the two main characteristics emphasis the consideration of a dynamic location problem in which the assignment costs change frequently during the planning phase and there must be significant costs for relocating facilities to the potential sites. If the first characteristic is absent then the given problem can be formulated as a SPLP. If the second characteristic is absent then, a set of disconnected SPLP can be considered one for each period of the planning phase.

Van Roy and Erlenkotter [27] described the dynamic simple plant location problem without capacity constraints, considering that a facility can only be open in the starting of a time period and they assumed remains open until and unless the end of the planning phase. If there are open facilities at the starting of the planning phase, the existing facilities can be closed at the termination of a time period and these facilities remains closed until the termination of the planning phase. The investigators describe a branch-and-bound approach that uses a dual ascent procedure and present computational results that shows the efficiency of the method.

Laporte and Dejax [18] presented the study that represents dynamic location-routing problem and they present two different solution procedures to solve the problem. Jacobsen [13] described several multi period capacitated location methods and models. Shulman [24] presented a dynamic location problem with capacity constraints. The author considers a fixed number of possible facilities with maximum capacities that accepting more than one facility can be allocated at the same site but in different time periods thus; increasing the existing capacity in one location.

Dias *et al.* [4] formulated a dynamic location problem with opening, closing and reopening of facilities and primal-dual heuristic is proposed that computes both upper and lower limits to its optimal solution. They consider the possibility of reconfiguring any location more than once over the planning horizon. A primal-dual heuristic was developed and tested over a set of randomly generated test problems. The result obtained are quite good, both in terms of the quality of lower and upper bounds calculated as in terms of the computational time spent by the heuristic. A branch and bound procedure that enables to optimize the problem is also described and tested over the same set of randomly generated problems.

Considerable attention has been devoted to the un-capacitated facility location problem (UFLP) in mathematical programming. Krarup and Pruzan [16] presented study on simple plant location problem. Efroymson and Ray [5] have proposed a formulation of the un-capacitated plant location problem with several constraints and they used branch and bound approach to solve it. Erlenkotter [7] has developed a dual-based branch and bound algorithm for the problem and this algorithm has been widely accepted as an efficient known approach. Thus, one of the main results has been the development of linear programming-based branch and bound algorithms. The most standard study in this

area is the algorithm given by Erlenkotter [7], that is a branch and bound algorithm based on dual descent which is an efficient approach to solve the un-capacitated facility location problem.

Korkel [14] showed in his study the exact solution for large-scale simple plant location problem that modify a primal-dual version of Erlenkotter [5] algorithm. The approximation algorithms for the un-capacitated facility location have also been studied by many authors. Beasley [2] has proposed Lagrangian heuristics for location problems and his results indicated that his proposed procedure is robust and efficient for solving different location problems. Simao and Thizy [25] have been developed a dual simplex algorithm for the canonical representation of UFLP. Conn and Cornuejols [3] proposed a theory of projection also exploiting a dual formulation. Gao and Robinson [8] presented a study of general model, dual-based and branch & bound procedure to find optimal solutions for several UFLP. They claimed that their proposed solution approach effectively solves realistically sized UFLP.

Al-Sultan and Al-Fawzan [1] add a concept of Tabu search to the UFLP in his study. The best neighbor which is not selected, Tabu is selected as the next state if it improves the objective. The main step is executed for a number of times and the algorithm also uses a sophisticated and effective heuristic as the beginning point of the Tabu search.

Ignizio and Cavalier [12] have solved and formulated the problem of allocating upto a fixed number of sites from among a given number of potential sites for allocating facilities at them and clustering ration shops to the selected sites in such a way that each ration shop is assigned to a unique selected site. The problem has only a single objective to minimize and that is the sum of distances from each ration shop to his/her assigned site.

Prakash *et al.* [22] consider the selection of warehouse sites for clustering ration shops to them with two objectives through a heuristic algorithm incorporating tabu search. The problem is to selects upto a fixed numbers of sites from among a given number of potential warehouse sites for assigned ration shops to them with several constraints and two objectives those have to be minimized. One of the constraints is that each ration shop must be assigned to a unique site, which is selected for allocating a warehouse at it and however there is no restriction on the number of ration shops to be assigned to a warehouse at the selected site. Other constraint is that the setup cost of the warehouses

should not exceed a certain budgetary amount that is given. The two objectives are to minimize the total cost and duration of meeting requirements of all the ration shops from their assigned warehouses at the allocated sites. The list of all efficient solutions of this problem has been found through solving a sequence of prioritized bicriterion problems by a heuristic approach. Each of the prioritized bicriterion problem is solved using a heuristic approach consisting of a combination of add and drop rules. Combining incorporation of tabu search with heuristic leads to better output than those obtained earlier without incorporation tabu search. The incorporation of tabu search allows search for global optimal solution in a longer or wider region so that it increasing the possibility of reaching at the global optimal solution or a solution very close to it and thus leading to better outputs.

Tabu search that is associated with a heuristic optimize algorithm seeking global optimal solution. This is because most of the heuristic algorithms select moves in the neighborhood from the current solution which improve the value of the objective function. This produce the result at times into getting stuck at the local optimum and terminating the algorithm at the reaching at the local optimum instead of the global optimum solution. The most important characteristics of the tabu search are explained as:

(i) Tabu search allows moves, which even worsen the value of the objective function in order to get out of the local optimum solution.

(ii) Tabu temporarily prevents cycling for example, revisiting the solutions already examined recently by using a tabu list. The tabu list contains all the forbidden moves which prohibit revisiting the solutions have been examined recently that increase their efficiency. The tabu list changes as move to the next solution no matter whether it is worse or best

(iii) Tabu uses the concept of an incumbent solution which is the best solution or result gotten so far. The incumbent solution changes to a better from the current solution as arrive to it.

(iv) Algorithm terminates when either revisit an already examined solution point or not able to find a new current position.

A detailed discussion about heuristic approach and the methods based on this approach incorporating tabu search can be found in the works of Ignizio and Cavalier [12] and Rardin [23].

The study presented by Kratica *et al.* [17] the simple plant location problem (SPLP) is considered. They proposed a GA to solve this problem. They have concluded that by using the purposed algorithm it is possible to solve SPLP with more than 1000 facility sites and their assigned ration shops. All the computational results are demonstrated and compared to existing dual based algorithms.

According to Dwivedi *et al.* [21] developed a flexible method for solving the travelling salesman problem (TSP) using GA. They used the TSP as the domain in their problem.TSP has considered being NP-complete and it is the standard example of such problems. There are many approaches had been many developed to solve this problem using classical methods such as integer programming and by very well known study of graph theory algorithms with different success. There paper gives a solution which includes a GA implementation to give a maximal approximation of the problem as can with the minimum cost. They have used the GA crossover as a main operator to solve the problem of travelling salesman. There was lot of attempt to discover an efficient and appropriate crossover operator. They have been presented a strategy that find the nearly optimized solution to these NP-complete type of problems, using the purposed crossover technique for GA that generates most appropriate solution to the TSP. In their paper the efficiency of their crossover operator is compared as against some existing crossover operators. The main work proposed in there paper intends to compare the efficiency of the new crossover operator with some existing ones.

Gulsen and Smith [10] proposed a hierarchical GA, a framework for calculating closed solution form functions that uses multi-variant data sets. In which the hierarchy begins with an upper GA that searches for appropriate functional forms given a user defined set of candidate independent variables and the primitives. In which each functional form is coded as a tree structure. Where coefficients, functional and primitives are linked and then the functional forms are sent to the second part to the data set and the chosen error metric. To ignore undue complication of the functional form that is identified by the upper GA, to calculate fitness they used the penalty function. Due to the computational effort required for this sequential optimization of each candidate function, this system has been implemented on a Cray supercomputer.

Maric **[**19**]** in this paper he proposed a new evolutionary method for solving the multi-level uncapacitated facility location problem (MLUFLP). They used the binary encoding

scheme with appropriate objective function that contains dynamic programming approach for finding sites to located facilities on each level to satisfy areas' required demands. First they perform experiments on the modified standard single level facility location problem. GA has found all known optimal or the best solutions for smaller dimension instances. The optimal solution found by total enumeration and CPLEX solver. They conclude that all optimal/best known solutions were found by GA for a single-level variant of the problem.

According to Tohyama *et al.* [26] UFLP is a fundamental optimization problem that involves the selection of different locations at which facilities providing the same service are to be allocated. It has been shown that the UFLP is NP-hard problem. It has usually been thought that there is no chance to finding a polynomial time algorithm for which an optimal solution can always found. They proposed a GA for solving the UFLP in this paper. In the UFLP, According to the ratio of the cost of facility allocated and the cost to ration shops of the facility, the number of facility allocation can be roughly estimated in this UFLP. Thus the partial solution spaces that are supposed to contain a good solution can be predicted to some extent that can be on the basis of the classification index. The Mutation operator of GA with the operation that searches the solution space that is supposed to contain a good solution, the proposed approach can search the whole space of solutions efficiently. They compared effectiveness of their method by using a numerical experiment with existing methods.

1.8 Present Work

In this study, a warehouse location problem considered by Prakash *et al.* [22] is reviewed and a genetic algorithm has been proposed to solve this problem. Also the concept of a single objective dynamic facility location problem considered by Dias *et al.* [4] is introduced to a multi objective warehouse problem and an algorithm proposed by Prakash *et al.* [22] is modified to find the set of efficient solutions.

CHAPTER-2

WAREHOUSE LOCATION PROBLEM AND GENETIC ALGORITHM

2.1 Introduction

Ignizio and Cavalier [12] have formulated and solved the problem of selecting upto a fixed number of sites from among a given number of potential sites for locating warehouses at them and clustering customers to the selected sites in such a way that each customer is assigned to a unique selected site. The problem has a single objective to minimize the sum of distances from each customer to his/her assigned site. Prakash *et al.*. [22] consider the selection of warehouse sites for clustering ration shops to them with two objectives through a heuristic algorithm incorporating tabu search. The problem selects up to a fixed number of sites from among a given number of potential sites for clustering ration shops to them subject to several constraints with two objectives. One of the constraints is that each ration shop should be clustered to a unique site which is selected for locating a warehouse at it: however there is no restriction on the number of ration shops to be clustered to a warehouse at the selected site. Another constraint is that the setup cost of the warehouses should not exceed a certain budgetary amount. The two objectives are to minimize the total cost and duration of meeting requirements of all the ration shops from their assigned warehouses to the selected sites. The set of efficient solutions of this problem has been found through solving using a heuristic algorithm consisting of a combination of add and drop rules. The incorporation of Tabu search leads to better results than those obtained earlier without its incorporation.

In this chapter, A GA has been proposed to find out the efficient solutions for a problem of selection of warehouse sites for clustering ration shops to them with two objectives Prakash *et al.* [22].

2.2 Problem Formulation

The problem of selecting upto fixed number of sites from among a given number of potential warehouse site for clustering a given number of ration shops to them subject to several constraint with two objectives (Prakash *et al.* [22]) is considered and reproduce here for ready reference. Suppose that there are *M* ration shops, *N* potential warehouse

sites and *K* be the maximum number of sites, which can be selected from among the *N* potential sites for locating warehouses at them. The *M* ration shops are to be clustered upto *K* sites in such a way that each shop is assigned to a unique site, which is selected for locating a warehouse. There is no restriction on the number of ration shops to be clustered to a selected site. Let c_{ij} and t_{ij} *(i=1,2,…,M; j=1,2,…,N)* units be the cost and time, respectively of meeting requirements of ration shop *i* from site *j*. *Let* c_j units be the setup cost of setting up a warehouse at the site *j* and *B* units be the budgetary amount allocated for setting up warehouses. Let x_{ij} be the binary variable assuming value 0 or 1 according as ration shop *i* is not assigned or assigned to site *j*, and y_j be the variable assuming the value 0 or 1 according as potential site *j* is not selected or selected for locating a warehouse. Thus, the problem seeks to select upto a fixed number of sites from among a given number of potential sites for locating warehouses at them with two objectives subject to several constraints.

The two objective functions which are to be minimized are:

$$C = \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} x_{ij} \qquad (2.1)$$

$$T = \max\{t_{ij} x_{ij} : i = 1,2,\ldots,M;\ j = 1,2,\ldots,N\} \qquad (2.2)$$

And Constraints of the problem are

$$\sum_{j=1}^{N} y_j \leq K \qquad (2.3)$$

$$\sum_{j=1}^{N} c_i y_j \leq B \qquad (2.4)$$

$$\sum_{j=1}^{N} x_{ij} = 1 \ \{i = 1,2,\ldots, M\} \qquad (2.5)$$

$$x_{ij} - y_j \leq 0 \{i = 1,2,\ldots, M;\ j = 1,2,\ldots, N\} \qquad (2.6)$$

$$x_{ij}, y_j = 0 \text{ or } 1 \{i = 1,2,\ldots, M;\ j = 1,2,\ldots, N\} \qquad (2.7)$$

Where above symbols denotes the following:

M = Number of ration shops

N = Potential warehouse sites

K = No of sites , which can be selected from among the N potential sites for locating warehouse at them.

B = Budgetary amount allocated for the setup of warehouses.

c_{ij} & t_{ij} (i=1,2,…,M ; j=1,2,…,N) = Units be the cost and time respectively of meeting requirements of ration shop i from site j.

c_j = setup cost of a warehouse at the site j.

x_{ij} = variable assuming value 0 or 1 according as ration shop i is not assigned or assigned to site j.

y_j = variable assuming values 0 or 1 according as the site j is not selected or selected for locating a warehouse at it.

C & T = denote the total cost and duration respectively of meeting requirements of all the ration shops from their assigned warehouses

2.3 Solution Procedure

The problem of uncapacitated warehouse location problem is solved using GA with heuristic approach. In the problem there are N potential locations available for the construction of warehouse and M ration shops that are allocated to the k warehouses. Therefore, K warehouse locations have to be selected from the given potential warehouse locations. These K warehouse locations are to be selected in such a way that total cost and time of meeting the requirements of the ration shops are minimized. At first, take a random initial population containing 2n to 8n individuals where n is the number of the variable. Each individual in the population represent a set of potential warehouse locations. Each individual in the population will represent K locations on which warehouses are to be structured. There are many ways for generating initial random population, for *e.g.*, with use of a random number table, flipping a coin deciding which side to be taken as 1, *etc*. After taking the initial population, next is to calculate the total cost and time of meeting the requirements of the ration shops for each individual in the population. To calculate cost and time of the individual in the population, following approach is used.

Procedure to obtain 1st efficient solution

An individual is taken into the consideration only if, the budgetary constraint is not violated for *K* sites which is contained by an individual. First, search the minimum unit cost, c_{ij} for the first ration shop among the *K* sites taken as the one of the individuals in the population. For this, the unit cost of the first site is considered as the minimum unit cost and then compared with remaining unit cost of selected sites. In this way, the minimum unit cost for the first ration shop is calculated and this is assign to the site which has minimum unit cost, $c_{ij.}$ In case of tie, select the site which has less unit time. Moreover the minimum unit time is same for two or more, any site can be selected. To assign the second ration shop, again find the minimum unit cost, c_{ij} for the sites. Allocate the second ration shop to the site which has minimum unit cost. If the minimum unit cost is same, allocate ration shop to a site with minimum unit time. If the unit time is also same, allocate the ration shop to any one of the sites. Repeat the same procedure for the remaining ration shops. While doing this, it is important that each ration shop is assigned to a unique warehouse site which is selected for locating a warehouse.

Once all the ration shops are allocated to their concerned warehouses, calculate the sum of these unit costs i.e. total cost. Concurrently, calculate the maximum time within which all the requirements of the ration shops from their concerned warehouses can be fulfilled. To calculate the maximum time, first store all unit time, t_{ij} corresponding to the selected site at which the ration shops are allocated, then retrieve the maximum unit time from them. Repeat the above procedure, for all the individuals in the initial population. Incumbent solution is the best solution obtained so far. Evaluate the best solution from among the total cost *C* and time *T* of the individuals in the population, taking the cost as the prioritized function over time function. Consider this solution as the incumbent solution. Represent the each individual in the initial population into its binary form by converting each site of the individual separately in its binary form. The maximum value that a site can take goes upto N, this number can be represented in the 2^n-1 binary digits. Therefore, all the sites can be representated within 2^n-1 binary digits. Hence each site in an individual is represented, separately, in its binary form, for *e.g.*, if an individual contain a sites set of 1, 2 and 3. Each site 1,2,3 are separately represented by 001,010 and 011 respectively. These three binary representations are then combining to make one binary representation i.e. 001010011. Repeat the above procedure for all the individuals

in the population and represent each set of sites i.e. individuals in its nine digits binary form.

Select the two parents on which crossover and mutation operators are to be applied. Selection of the two parents depends upon the value of the fitness function. As the problem is of minimizing, hence two individuals giving minimum value of the fitness function are taken as parents. Fitness function is evaluated by ratio of total cost of selected site and sum of total cost. Compare the value of fitness function of each individual in the population and select the individual which has minimum value as one of the parents, again compare the fitness value of each individual in the population and select the second minimum from the population as the second parent. Once the parents are decided, crossover is then applied on these parents. Crossover is applied to the binary representation of the potential warehouse sites. In crossover, binary bits are exchanged between the two parent sites. Single point crossover is performed on the two parents and to perform single point crossover, crossover site is generated randomly. Crossover site is the position in the binary array of the parents, from which exchange of bits take place and goes upto the entire length of the array, for *e.g.*, two best parents be 001010011 and 110011101 and crossover site is 4. Therefore all the bits after 4^{th} position will be exchanged between the two parents, i.e. 001011101 and 110010011. Crossover will be done only if it does not produce any undesirable results. A check is performed on the crossover operator, if it produces any invalid code then crossover is not performed on the two parents.

Next, mutation operator is applied. In mutation, one bit is reversed in both the parents so as to produce new generations. If the bit is one then it is reversed to zero and if bit is zero then it is reversed to one. To perform mutation, mutation site is randomly generated which is the position in the binary strings where mutation is performed, for *e.g.* after crossover, two binary strings be 001011101 and 110010011 and mutation site generated randomly be 7. Then output after mutations be 001011111 and 110010001. Hence these two new binary strings will replace the old binary strings and become the new generations. But before performing mutation on the two best individuals in the population, a check is performed. This check ensures that if after mutation, any undesirable results are generated then mutation is not performed. Mutation is performed only if it is ensured that it will not generate any invalid codes. By following the above

procedure the new generation is generated. Store the best solution of the new generation and compare it with the incumbent solution. If the incumbent solution is better than there is no change in incumbent solution. If the best solution obtained in the new generation is better than incumbent solution then this solution become the incumbent solution. Repeat the above procedure for all the generations till the desired first efficient solution is obtained.

Procedure to obtain 2nd efficient solution

The second efficient solution is obtained by replacing the time equal to or greater than time obtained in first efficient solution with arbitrarily large value. Then same procedure is followed as described above and subsequent efficient solutions are obtained.

2.4 Algorithm

Input: Cost matrix and time matrix

Output: Efficient Solutions

1. Evaluate initial population, P(0) randomly.
2. Represent initial population, P(0) in its binary form .
3. While (no more efficient solutions are possible) do
    - 3.1. While(P(k) < maximum no. of population) do
        - For(i=0; i<number of individuals in population; i++)
            - Calculate total cost, C(k)(i) and time, T(k)(i) for the population, P(k).
    - 3.2 Store the best solution obtained so far in incumbent solution.
    - 3.3. Retrieve two best individuals from the C(k) and Take it as P1 and P2.
    - 3.4 Apply genetic operators to strings P1 and P2.
        - Crossover: Crossover site is generated randomly, and then bits of P1 and P2 are exchanged from the crossover site.
        - Mutation: Mutation site is generated randomly, and then that is reversed in both P1 and P2. Only one bit is changed at a time.
    - 3.5 k=k+1.
    - 3.6 Go to step 3.1.
4. Replace the entries of cost matrix and time matrix with very large value.
5. Go to Step 3.
6. End.

2.5 Case Study

In the Case Study, example of warehouse location problem is considered. In it, there are N = 7 potential warehouse locations, k = 3 warehouses are to be structured, M = 5 ration shops that are to be allocated to three warehouse. B = 1400000.

In the table below, rows correspond to ration shops and columns correspond to sites.

Table 2.1: Data Input for 1st Efficient Solution[22]

Ration Shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	40	30	50	120	180	170	150
	8	2	11	6	10	9	7
2	70	80	130	130	170	140	20
	6	6	9	7	8	10	11
3	80	20	200	70	140	130	60
	8	9	5	8	11	12	10
4	60	10	70	80	30	160	210
	11	6	13	11	9	8	6
5	90	100	20	60	40	50	220
	10	13	8	12	6	6	14
SetUpCost©	100000	300000	700000	800000	400000	200000	500000

On the above problem, GA is applied as described in the solution procedure. The step by step output of the above is shown below with help of snaps.

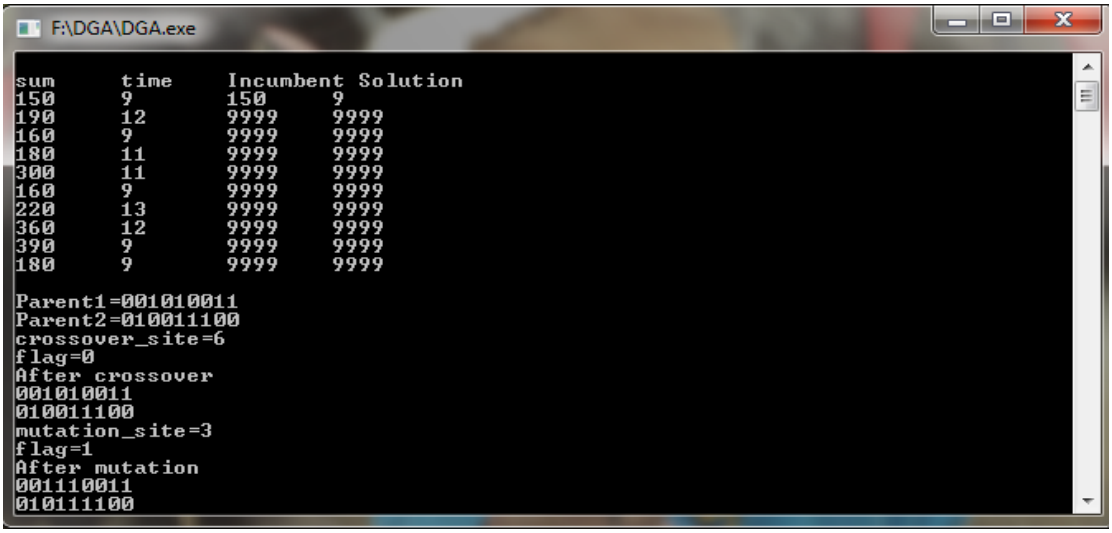

Figure 2.1: 1st Iteration of GA

In the figure 2.1, sum and time array with incumbent solution is shown. It is a snapshot of the processing performed on the initial population. The sum array contains the total cost C for each individual in the population. Each individual contains the three potential warehouse sites. The time array contains the maximum time corresponding to the each individual in the array. After this, two minimum cost in the sum array is found and mutation and crossover is applied. If the flag is 1 then only crossover and mutation is applied.

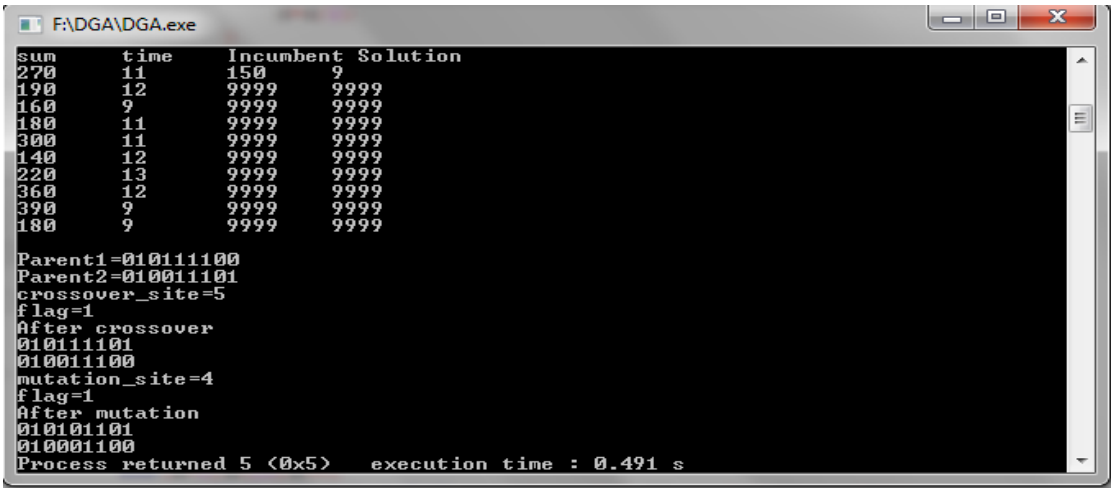

Figure 2.2: 2nd iteration of GA

In the figure 2.2, second iteration is shown. Sum and time array now contains the cost C and time T of the new population, i.e. the next generation, evolved after applying crossover and mutation on the previous population. In this snapshot, crossover site comes to be 5. As flag is 1, it denotes that the crossover is performed.

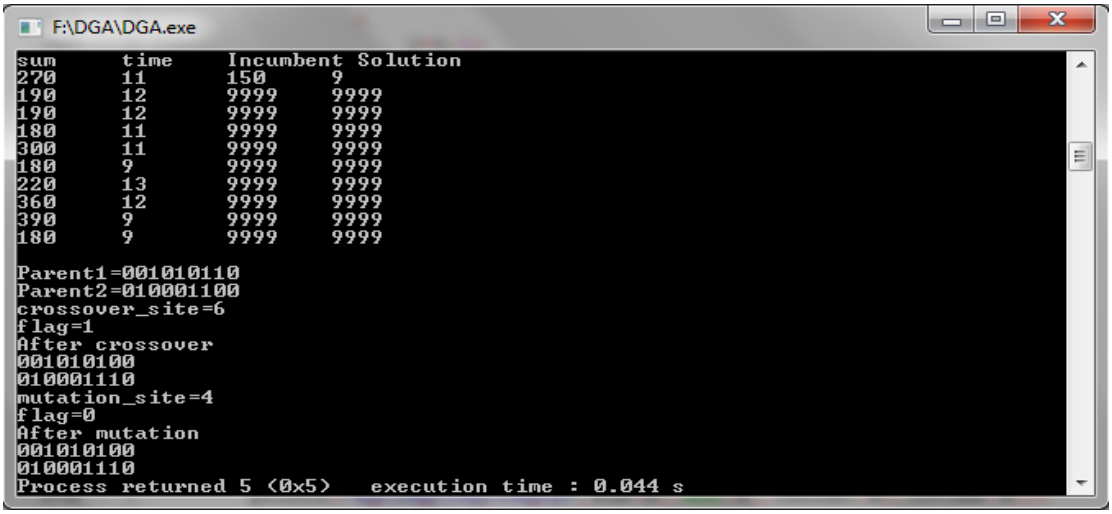

Figure 2.3: 3rd iteration of GA

In the figure 2.3, third generation of the problem is shown. In this, Incumbent solution shows the best result obtained so far. As combination (150, 9) is obtained in the first generation, then it is compared with results of the new generations but no solution dominates this solution so it remains unchanged. Other notations are same as above.

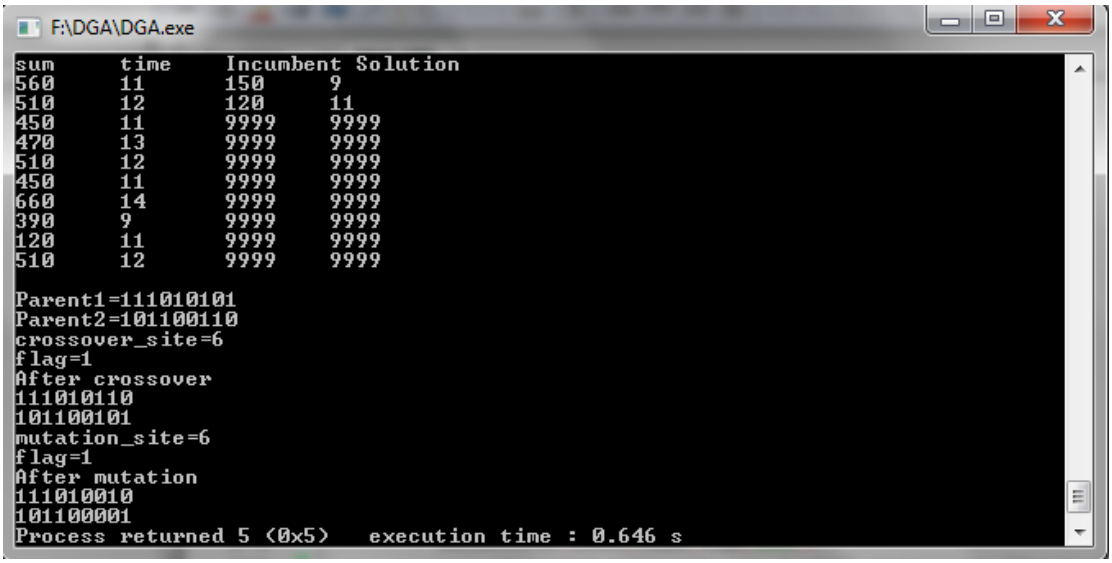

Figure 2.4: Incumbent solution of GA

In this figure 2.4, number of incumbent solution changes. Following the above procedure explained above, desired result is produced. New solution entered in the incumbent solution as sum array contains the cost which is less than cost of incumbent solution but time is more as compared to the time of incumbent solution so both of these solutions become the incumbent solutions.

2nd Efficient solution and subsequent ones

According to the 1st efficient solution, entries in the cost matrix and time matrix is changed. Time greater than or equal to time obtained in 1st efficient solution is replaced by very large value and corresponding cost is also replaced with very large value. Here, all entries t>=9 are blocked.

Table 2.2: Input Data for 2nd efficient solution[22]

Ration Shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	40	30	M	120	M	M	150
	8	2	M	6	M	M	7
2	70	80	M	130	170	M	M
	6	6	M	7	8	M	M
3	80	M	200	70	M	M	M
	8	M	5	8	M	M	M
4	M	10	M	M	M	160	210
	M	6	M	M	M	8	6
5	M	M	20	M	40	50	M
	M	M	8	M	6	6	M
SetUpCost©	100000	300000	700000	800000	400000	200000	500000

Apply GA for above data will allow to obtain the 2nd efficient solution. In the above table, all the time and cost entries related to time greater than 9 i.e. t >= 9 are replaced with very high value. This is done so that, in next efficient solution, time will be less than as compared to the previous efficient solution. Then GA is employed to find the efficient solution. GA is applied in the same way as it is applied for obtaining 1st efficient solution. Proceeding in the same manner subsequent efficient solutions are generated.

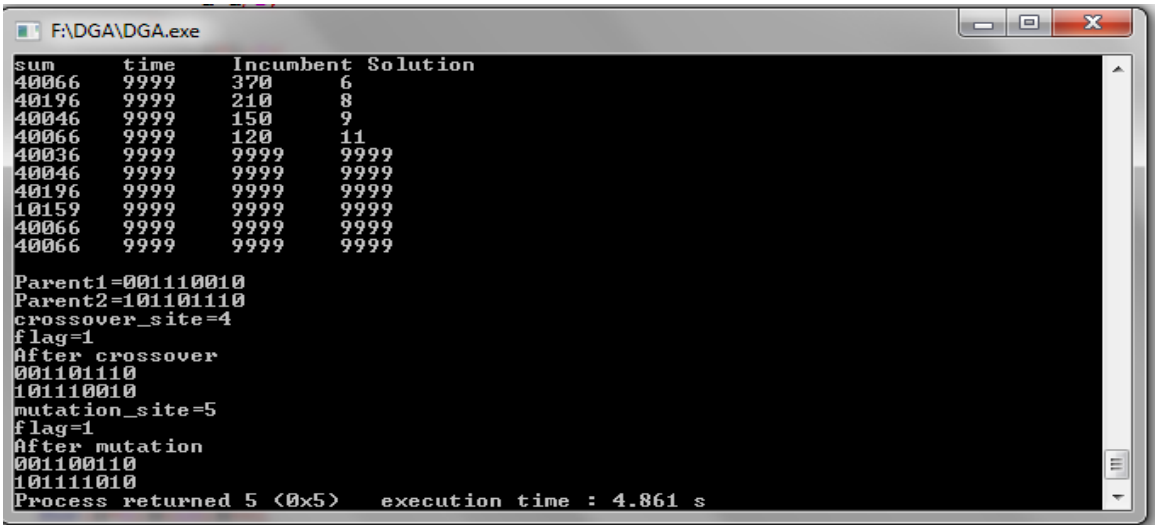

Figure 2.5: Efficient Solutions

Table 2.3: Efficient solutions

Variable at level 1	Total Cost of meeting requirements	Duration of meeting requirements
x_{12}, x_{22}, x_{33}, x_{42}, x_{55}, y_2, y_3, y_5	360	6
x_{12}, x_{21}, x_{31}, x_{42}, x_{53}, y_1, y_2, y_3	210	8
X_{12}, x_{21}, x_{32}, x_{42}, x_{53}, y_1 y_2, y_3	150	9
X_{12}, x_{27}, x_{32}, x_{42}, x_{55}, y_2, y_5, y_7	120	11

2.6 Conclusion

Prakash *et al.* [22] developed a heuristic algorithm incorporating tabu search scheme and applied to solve a numerical problem of selecting warehouse site and clustering ration shops to them and obtained four efficient solutions. In this chapter, a GA has been proposed for this warehouse site location problem and same numerical problem has been solved. It has been observed that the algorithm is quite simple and the same set of efficient solutions is obtained.

CHAPTER-3

EFFICIENT SOLUTIONS FOR UNCAPACITATED FACILITY LOCATION PROBLEM

3.1 Introduction

Dias *et al.* [4] consider a single objective dynamic facility location problem with the possibility of opening, closing and reopening a facility more than once during a planning horizon. The differentiation between the opening and the reopening is that it allows the differentiation of the corresponding fixed costs. There are several situations where these costs are clearly different (for instance, if the facilities have already been acquired or, in case of locating obnoxious facilities, if studies of environmental impact have already been done). The model proposed also considers the existence of operating and closing costs (that most of the times cannot be ignored). It is also possible to consider the existence of open facilities at the beginning of the planning horizon.

The idea based on this work is extended to a multi objective warehouse location problem considered in second chapter.

In the proposed problem, it has been considered that the warehouse is available for a long period but warehouse location is not available for this period. Hence the time is divided into smaller time periods and warehouse is allocated and re-allocated to the warehouse locations according to their availability in a particular time period. If a potential warehouse location is available in a time period but it is not available in next time period then warehouse is re-allocated to some other location available in next time period.

In this chapter the concept of Dias *et al.* [4] for a single objective dynamic facility location problem is introduced to multi-objective warehouse location problem with opening of new warehouse in different time periods. The algorithm proposed by Prakash *et al.* [22] is modified to find the set of efficient solutions.

3.2 Problem Formulation

Let us consider n ration shops, m potential warehouse sites, T number of periods and K be the number of sites, which can be selected from among the m potential sites for locating warehouses at them in T different time periods. The n ration shops are to be clustered upto K sites in such a way that each shop is assigned to a unique site, which is selected

for locating a warehouse. There is no restriction on the number of ration shops to be clustered to a selected site.

The two objective function which are to be minimized are:

Minimize

$$\sum_{t=1}^{T}\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}^{t} x_{ij}^{t} + \sum_{t=1}^{T}\sum_{i=1}^{m} FA_i^t y_i^t \qquad (3.1)$$

$$\max\{\sum_{t=1}^{T}\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij}^{t} x_{ij}^{t}\} \qquad (3.2)$$

Subject to

$$\sum_{i=1}^{m} y_i^t \le K \qquad t = 1,....., T \qquad (3.3)$$

$$\sum_{i=1}^{m} x_{ij}^{t} = 1 \qquad \forall\ j \in J,\ t = 1,....., T \qquad (3.4)$$

$$x_{ij}^{t} - y_i^t \le 0 \quad i \in I,\ j \in J,\ t = 1,....., T \qquad (3.5)$$

$$\sum_{\tau=1}^{t}\sum_{\varepsilon=t}^{T} (a_{i\tau}^{\varepsilon}) - y_i^t \ge 0 \quad \forall i \in I, t = 1,....., T \qquad (3.6)$$

$$\sum_{\tau=1}^{t}\sum_{\varepsilon=t}^{T} a_{i\tau}^{\varepsilon} \le 1 \quad \forall i \in I,\ t = 1,....., T \qquad (3.7)$$

$$a_{it}^{\varepsilon},\ x_{ij}^{t},\ y_i^t \in \{0,1\}\ \forall i \in I,\ j \in J,\ t = 1,....., T,\ \varepsilon \ge t \qquad (3.8)$$

Let us define the following notations:

J *{1,…….,n}* set of indexes corresponding to the ration shops' locations.

I *{1,…….,m}* set of indexes corresponding to warehouses' possible locations.

T Total number of time periods.

t *{1 ,……., T}* set of indexes corresponding to number of time periods.

FA_i^t Fixed cost of opening a warehouse *i* at the beginning of period *t*.

c_{ij}^t Unit cost of meeting requirements of ration shop *j* to warehouse *i* in period *t*.

d_{ij}^t Unit time of meeting requirements of ration shop *j* to warehouse *i* in period *t*.

K Number of sites to be selected from given potential warehouse sites.

Let us define the following variables:

$$x_{ij}^{t} \quad \begin{cases} 1 & \text{if client } j \text{ is assigned to a facility } i \text{ at the begining of period } t. \\ 0 & \text{otherwise} \end{cases}$$

$$a_{it}^{\varepsilon} \quad \begin{cases} 1 & \text{if facility } i \text{ is opened at the begining of period } t \text{ and stays} \\ & \text{open until the end of period } \varepsilon. \\ 0 & \text{otherwise} \end{cases}$$

$$y_{i}^{t} \quad \begin{cases} 1 & \text{if facility } i \text{ is selected for time period } t. \\ 0 & \text{otherwise} \end{cases}$$

Constraint (3.3) guarantees that, in each time period up to *K* number of warehouses can be selected. Constraint (3.4) ensures that each ration shop is assigned to exactly one warehouse in every time period. Constraint (3.5) ensures that each ration shop is assigned to a warehouse from among the selected warehouses. Constraint (3.6) guarantees that, in every time period, warehouses that are selected are from the warehouses that are available in that time period. Constraint (3.7) guarantees that, in every time period, only one warehouse can be opened in each location.

3.3 Solution Procedure

The problem formulated is a binary integer of nonlinear problem because the variables x_{ij}^{t} , a_{it}^{ε}, y_{i}^{t} are binary integers and the objective function given by equation (3.2) is nonlinear. The list of all efficient solutions can be obtained through an iterative heuristic approach incorporating tabu search requiring solutions of the prioritized bicriterion problems.

The procedures to obtain the first, second, and subsequent efficient solutions are explained as:

Procedure for 1st efficient solution

The first efficient solution $(X^{(1)},Y^{(1)})$ of the above problem given by equations (3.1) to (3.7) is the optimal solution of the problem where the time and the total cost to fulfill the requirements of the ration shops from their allocated warehouses locations are minimized with the first and the second priorities respectively with constraints that are represented by the equations (3.3) to (3.7). The heuristic approach is proposed to solve the problem is the extended and modified of the heuristic approach that was proposed Ignizio [11] to

solve a partial covering problem with a single objective. The proposed approach consists of two phases to solve the problem. The first phase, a solution of the first prioritized bicriterion problem can be calculated using the add rule of Tabu. The solution obtained in the first phase is improved successively by using a combination of add and drop rules of tabu search this step is applied in to second phase. No sites are selected in the first phase. First determine the sites that are available for the first time period and subsequent periods with the help of concerned variable for the given problem. Take the cost and time of meeting bulk requirements as infinity for the sites not available in a period. Select the site wherefrom the total cost of meeting the requirements of all the ration shops is minimized. When there is a tie, select the site wherefrom the maximum time among the times of meeting requirements of ration shop is also minimized and if there is a tie again, we can select arbitrarily any site among the tied ones. Determine the time period up to which first site selected is available. Now select the second site in combination with the already selected first site in such way that total cost of meeting the warehouses are minimized. Calculate the cost for the time periods for which both the sites are available. Assign ration shop to the location where from cost of meeting the warehouses are minimum among the selected locations.

In the case of a tie, select the location which in combination with the already selected location also minimizes the maximum time among the times of meeting warehouses of each ration shop from its assigned location. Then select a third location, which in combination with the two locations already selected minimizes the total cost of meeting warehouses of all ration shops. Calculate the cost for the time periods for which all three sites are available.

The process of selecting and allocating locations using add rule is terminated if at most K locations have been selected. Determine the least time period up to which all of the K locations are available and also determine the locations which are available in next time period locations which are not available from next time period. Select only those number of new sites which are not available in next time periods by following the same procedure as defined above. Continue this process until locations are selected for all time periods. This leads to a solution of the 1st prioritized bicriterion problem of the formulated problem which is termed as its iteration 0.

The second phase is different to the first in which call the drop and add rules with tabu search for more improvement of the obtained solution. Due to incorporation of tabu search allows search for global optimal solution in a wider region so that increase the possibility of reaching at the global optimal solution or very close to it and thus leading to better results. The incumbent solution will always contain the best solution. The incumbent solution for the first efficient solution $(X^{(1)}, Y^{(1)})$ is the best solution obtained so far. Obtained the first iterative solution $(X^{(1)}_{(1)}, Y^{(1)}_{(1)})$ from the 0^{th} iterative $(X^{(1)}_{(0)}, Y^{(1)}_{(0)})$ solution call the drop and add rules with tabu search as follows.

In it form two separate tabu lists for drop and add rules respectively. One list, the tabu list for drop rule consists of last two selected sites of first time period appearing in the order of their selection. The two locations included in the tabu list will be prohibited from drop from the 0^{th} iterative solution and the drop rule will drop only the first selected site.

Another list that is also known as tabu list for the add rule will include first location selected, in the first time period, in the 0^{th} iterative solution. The location in the tabu list of the add rule will be prohibited from being selected in the next iterative solution. The add rule with its associated tabu list has to be implemented after calling the drop rule with its tabu list resulting into the first iterative solution. After obtaining the first iterative solution, compare it with the incumbent solution and if the current solution is better than the incumbent solution overwrite incumbent solution with current solution otherwise incumbent solution remain unchanged because in that case incumbent solution is the best solution.

Now to find second iterative solution $(X^{(1)}_{(2)}, Y^{(1)}_{(2)})$ proceed in the same manner as explain above but with the changes in the tabu list. Now the tabu list for the drop rule used for obtaining second iterative solution that will now include last two locations of the first time period obtained in the first iterative solution and will appear in the order of their selection. The tabu list for the add rule used for obtaining second iterative solution will now include first location of the first time period obtained in the first iterative solution and if the current solution is better than the incumbent solution then overwrite take place that overwrite incumbent solution with current solution else incumbent solution remain unchanged because in that case incumbent solution is the best solution. Now for the third

iterative solution and subsequent solutions are obtained in the same way as described above. Change the incumbent solution to the current iterative solution if it is better than the incumbent solution else remains unchanged.

The process will terminate of obtaining newer iterative solutions when revisit an already visited solution or now it is not possible to obtain a new solution and the first efficient solution $(X^{(1)}, Y^{(1)})$ has obtained from the incumbent solution when this algorithm terminates.

Procedure for 2nd efficient solution

Now, the second efficient solution that is represented by $(X^{(2)}, Y^{(2)})$ of the formulated problem is obtained by solving the problem obtained from the first bicriterion problem after replacing all the cost and time whose time is the same or greater than the time obtained in the first efficient solution. The problem thus designated as second prioritized bicriterion problem. In the next step the second prioritized bicriterion problem is solved in the same way as the process solved the problem for the first prioritized bicriterion problem and similarly the subsequent efficient solution is obtained in the same way as the described above or done in the first phase. This process of calculating the efficient solution is terminated when there are all infinity or any of the constraints is violated indicating there is no further efficient solution possible.

3.4 Algorithm

Notations used in Algorithms

min_t : It denotes the time period for which locations are selected.

num_ration shop : Number of ration shops in the warehouse location problem.

setup_cost1 : It is the setup cost of the all the warehouses available in the first time period.

setup_cost2 : It is the setup cost of the all the warehouses available in the second time period

setup_cost3 : It is the setup cost of the all the warehouses available in the third time period.

setup_cost4 : It is the setup cost of the all the warehouses available in the fourth time period.

end_matrix : It is the matrix which denotes the time period up to which potential warehouse locations are available.

cost_matrix1 : It is a matrix representing unit cost of meeting all the requirements of the ration shops from the potential warehouse locations in the first time period.

cost_matrix2 : It is a matrix representing unit cost of meeting all the requirements of the ration shops from the potential warehouse locations in the second time period.

cost_matrix3 : It is a matrix representing unit cost of meeting all the requirements of the ration shops from the potential warehouse locations in the third time period.

cost_matrix4 : It is a matrix representing unit cost of meeting all the requirements of the ration shops from the potential warehouse locations in the fourth time period.

num_warehouse : Number of potential warehouse locations in the problem.

time_matrix1 : It is a matrix representing unit time of meeting all the requirements of the ration shops from the potential warehouse locations in the first time period.

time_matrix2 : It is a matrix representing unit time of meeting all the requirements of the ration shops from the potential warehouse locations in the second time period.

time_matrix3 : It is a matrix representing unit time of meeting all the requirements of the ration shops from the potential warehouse locations in the third time period.

time_matrix4 : It is a matrix representing unit time of meeting all the requirements of the ration shops from the potential warehouse locations in the fourth time period.

Algorithm selectWarehouseSite

Input: smallest_index3, flag, smallest_index2, num_warehouse, smallest_index, min_t, end_matrix, setup_cost1, setup_cost2, setup_cost3, setup_cost4, num_ration shop, cost_matrix1, cost_matrix2, cost_matrix3, cost_matrix4, time_matrix1, time_matrix2, time_matrix3, time_matrix4

Output: t3, smallest3, smallest_index3

1. for each I $\in$ num_warehouse do

   n $\leftarrow$ 1 , t $\leftarrow$ 0

1.1 if I indicates already selected site then

1.1.1 Make sum of that warehouse site infinite.

sum(i) = Inf

1.1.2 Go to next iteration.

1.2 if (select first site) then

1.2.1 Update sum for each warehouse site for particular time period.

else (select second or third warehouse site)

1.2.2 Add setup cost of selected sites and new site for particular time period.

1.3 for each K ∈ end_matrix do

1.3.1 If (k>end_matrix of selected sites) then break the loop.

1.3.2 for each j ∈ num_ration do

1.3.2.1 Calculate the minimum cost from the selected sites and new site for a concerned time period.

1.3.2.2 Calculate corresponding time period.

end

1.3.3 select the site with minimum sum.

End

Algorithm Calsum1

Input: min_t, num_ration shop, setup_cost1, setup_cost2, setup_cost3, setup_cost4, end_matrix, cost_matrix1, cost_matrix2, cost_matrix3, cost_matrix4, num_warehouse

Output: sum_cost1

1. for each I ∈ num_warehouse

1.1 Initialize sum_cost1 with very large value.

1.2 Update sum_cost1 according to the min_t.

1.3 Add unit cost to the sum_cost1 according to its num_warehouse and num _ ration.

3.5 Case Study

Consider an example of warehouse location problem in which there are m=7 potential locations, K=3 warehouses, n=5 ration shops and T=4 time periods. In the table below, rows 1-5 correspond to ration shops and columns 1-7 correspond to potential warehouses sites. Upper and lower entries of cell (i, j) depicts the units of costs and time respectively of meeting the requirement of ration shop from potential warehouse location. Four different tables denote the data taken in four time periods. For each time period unit and time respectively of meeting the requirement of ration shop from potential warehouse location is changed. On applying the procedure discussed above, obtain the efficient solutions of the problem and then incorporate the tabu search approach to iterate the elusive global optimum, while considering the moves which even worsen the values of the objective function.

Table 3.1: Data for 1st period in the problem

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	40	30	50	120	180	140	150
	8	2	11	6	10	9	7
2	70	80	130	130	170	140	20
	6	6	9	7	8	10	11
3	80	200	200	70	140	70	60
	8	9	5	8	11	12	10
4	60	10	70	80	30	160	210
	11	6	13	11	9	8	6
5	90	100	20	60	40	50	220
	10	13	8	12	6	6	14
SetUpCost©	100000	300000	700000	800000	400000	200000	500000

Table 3.2: Data for 2nd period in the problem

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	50	40	40	120	180	170	120
	9	3	11	6	10	9	7
2	80	80	100	150	200	140	200
	6	6	7	6	11	10	5
3	80	90	20	20	30	60	60
	11	10	11	8	9	13	10
4	50	90	40	50	140	40	220
	6	10	6	6	11	4	14
5	10	130	70	160	70	80	210
	6	12	8	8	13	11	6
SetUpCost©	100000	400000	600000	800000	600000	500000	600000

Table 3.3: Data for 3rd period in the problem

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	20	160	20	100	90	80	160
	8	8	11	7	10	6	8
2	80	170	120	60	40	130	60
	11	9	6	12	10	12	12
3	80	50	100	50	70	210	60
	6	6	6	9	13	6	10
4	90	180	40	140	200	70	80
	10	10	11	11	11	5	11
5	230	140	120	240	30	20	100
	11	10	7	3	9	11	13
SetUpCost©	600000	800000	500000	700000	400000	600000	100000

Table 3.4: Data for 4th period in the problem

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	140	140	170	120	120	50	60
	10	10	9	7	6	6	12
2	40	20	90	40	160	30	160
	11	11	10	11	8	9	8
3	10	80	10	80	80	60	200
	6	6	6	6	11	10	5
4	50	20	60	180	220	210	150
	9	11	12	11	14	6	9
5	180	20	100	200	30	40	220
	10	8	7	11	12	3	13
SetUpCost©	400000	100000	500000	800000	700000	600000	500000

Table 3.5: Site Availability Matrix for all time periods

Warehouse Site	Time Peroid							
	I		II		III		IV	
1	a_{11}^{1}	0	a_{12}^{1}	-1	a_{13}^{1}	-1	a_{14}^{1}	-1
	a_{11}^{2}	0	a_{12}^{2}	0	a_{13}^{2}	-1	a_{14}^{2}	-1
	a_{11}^{3}	1	a_{12}^{3}	0	a_{13}^{3}	0	a_{14}^{3}	-1
	a_{11}^{4}	0	a_{12}^{4}	0	a_{13}^{4}	0	a_{14}^{4}	0
2	a_{21}^{1}	0	a_{22}^{1}	-1	a_{23}^{1}	-1	a_{24}^{1}	-1
	a_{21}^{2}	1	a_{22}^{2}	0	a_{23}^{2}	-1	a_{24}^{2}	-1
	a_{21}^{3}	0	a_{22}^{3}	0	a_{23}^{3}	0	a_{24}^{3}	-1
	a_{21}^{4}	0	a_{22}^{4}	0	a_{23}^{4}	0	a_{24}^{4}	0
3	a_{31}^{1}	0	a_{32}^{1}	-1	a_{33}^{1}	-1	a_{34}^{1}	-1
	a_{31}^{2}	0	a_{32}^{2}	0	a_{33}^{2}	-1	a_{34}^{2}	-1
	a_{31}^{3}	1	a_{32}^{3}	0	a_{33}^{3}	0	a_{34}^{3}	-1
	a_{31}^{4}	0	a_{32}^{4}	0	a_{33}^{4}	0	a_{34}^{4}	0
4	a_{41}^{1}	0	a_{42}^{1}	-1	a_{43}^{1}	-1	a_{44}^{1}	-1
	a_{41}^{2}	0	a_{42}^{2}	0	a_{43}^{2}	-1	a_{44}^{2}	-1
	a_{41}^{3}	0	a_{42}^{3}	1	a_{43}^{3}	0	a_{44}^{3}	-1
	a_{41}^{4}	0	a_{42}^{4}	0	a_{43}^{4}	0	a_{44}^{4}	0
5	a_{51}^{1}	0	a_{52}^{1}	-1	a_{53}^{1}	-1	a_{54}^{1}	-1
	a_{51}^{2}	0	a_{52}^{2}	0	a_{53}^{2}	-1	a_{54}^{2}	-1
	a_{51}^{3}	0	a_{52}^{3}	0	a_{53}^{3}	0	a_{54}^{3}	-1
	a_{51}^{4}	1	a_{52}^{4}	0	a_{53}^{4}	0	a_{54}^{4}	0
6	a_{61}^{1}	0	a_{62}^{1}	-1	a_{63}^{1}	-1	a_{64}^{1}	-1
	a_{61}^{2}	0	a_{62}^{2}	0	a_{63}^{2}	-1	a_{64}^{2}	-1
	a_{61}^{3}	0	a_{62}^{3}	0	a_{63}^{3}	0	a_{64}^{3}	-1
	a_{61}^{4}	1	a_{62}^{4}	0	a_{63}^{4}	0	a_{64}^{4}	0
7	a_{71}^{1}	0	a_{72}^{1}	-1	a_{73}^{1}	-1	a_{74}^{1}	-1
	a_{71}^{2}	0	a_{72}^{2}	0	a_{73}^{2}	-1	a_{74}^{2}	-1
	a_{71}^{3}	0	a_{72}^{3}	0	a_{73}^{3}	0	a_{74}^{3}	-1
	a_{71}^{4}	1	a_{72}^{4}	0	a_{73}^{4}	0	a_{74}^{4}	0

First Non-dominated Solution

Step 1. Add the entire costs column wise for each warehouse for the time periods corresponding warehouse is available and the least of the total costs thus selected.

Table 3.6: Total cost and time for one site

		Warehouse potential sites						
		1	2	3	4	5	6	7
Sum of costs in single column		1110	850	1140	Inf	2310	1950	2720
Set up cost		100000	300000	700000	Inf	400000	200000	500000
Total cost		101110	300850	701140	Inf	402310	201950	502720
Corresp.Time		11	13	13	Inf	14	13	14

Step 2. Selection of two warehouse potential sites at a time.

Table 3.7: Total cost and time for two sites

		Warehouse potential sites					
		(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)
Sum of costs in two column		540	800	Inf	730	880	840
Set up cost		400000	800000	Inf	500000	300000	600000
Total cost		400540	800800	Inf	500730	300880	600840
Corresp.Time		11	11	Inf	13	13	13

As the pair first and sixth pair got the least value so these two sites are selected.

Step 3. Selection of three warehouse potential sites at a time.

Table 3.8: Total cost and time for three sites

		Warehouse potential sites				
		(1,6,2)	(1,6,3)	(1,6,4)	(1,6,5)	(1,6,7)
Sum of costs in three column		460	690	Inf	680	720
Set up cost		600000	1000000	Inf	700000	800000
Total cost		600460	1000690	Inf	700680	800720
Corresp.Time		13	12	Inf	13	13

Triplet (1, 6, 2) is giving the least value so these three sites .i.e. 1,6 and 2 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 4. Selection of warehouse potential sites for third time period.

Table 3.9: Total cost and time of three sites for third time period

		Warehouse potential sites			
		(1,6,3)	(1,6,4)	(1,6,5)	(1,6,7)
Sum of costs in three column		240	220	220	230
Set up cost		500000	700000	400000	100000
Total cost		500240	700220	400220	100230
Corresp.Time		11	12	13	12

As the triplet(1,6,7) gives the minimum cost so sites 1 , 6, and 7 are selected for time period III.

Step 5. Selection of warehouse potential sites for fourth time period.

There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.10: Total cost and time of three sites for fourth time period

		Warehouse potential sites
		(5,6,7)
Sum of costs in three column		320
Set up cost		700000
Total cost		700320
Corresp.Time		12

Therefore,

Total cost =14,01,010.
Max Time=13.

Now ,Tabu search will be applied. In it , Iterations are performed to improve upon the solution with respect to the first objective function.

Iteration No .1 : The locations selected for setting up of warehouses are 1, 6 and 2. Thus, the Tabu lists are given by

Table 3.11: Tabu search for 1st iteration

Tabu list for drop	Tabu list for add
$\{y_6, y_2\}$	$\{y_1\}$

Hence, the solutions at this stage are

Step 1.

Table 3.12: Total cost and time for three sites in 1st iteration

		Warehouse potential sites			
		(6,2,3)	(6,2,4)	(6,2,5)	(6,2,7)
Sum of costs in three column		560	Inf	490	470
Set up cost		1200000	Inf	900000	1000000
Total cost		1200560	Inf	900490	1000470
Corresp.Time		12	Inf	13	13

Triplet (6,2,5) is giving the least value so these three sites .i.e. 6 , 2 and 5 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 2.

Selection of warehouse potential sites for remaining time periods. Since sites 6 and 5 is available for third time period. Therefore find only one more site.

Table 3.13: Total cost and time for three sites in 3rd and 4th time period in 1st iteration

		Warehouse potential sites			
		(6,5,1)	(6,5,3)	(6,5,4)	(6,5,7)
Sum of costs in three column		220	190	260	590
Set up cost		600000	500000	700000	100000
Total cost		600220	500190	700260	100590
Corresp.Time		13	13	11	12

As triplets (6,7,5) has minimum cost therefore sites 6,7 and 5 is selected for third and fourth time period.

Total cost = 1001080.

Max Time=13.

Iteration No.2: The Tabu lists are given by

Table 3.14: Tabu search for 2nd iteration

Tabu list for drop	Tabu list for add
$\{y_2, y_5\}$	$\{y_6\}$

Hence, the solution at this stage are

Step 1

Table 3.15: Total cost and time for three sites in 1st and 2nd time period in 2nd iteration

		Warehouse potential sites			
		(2,5,1)	(2,5,3)	(2,5,4)	(2,5,7)
Sum of costs in three column		440	530	Inf	470
Set up cost		800000	1400000	Inf	1200000
Total cost		800440	1400530	Inf	1200470
Corresp.Time		9	11	Inf	13

Triplet (2,5,1) is giving the least value so these three sites .i.e. 2 , 5 and 1 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 2.

Selection of warehouse potential sites for remaining time periods. Since sites 5 and 1 is available for third time period. Therefore find only one more site.

Table 3.16: Total cost and time for three sites in 3rd time period in 2nd iteration

		Warehouse potential sites			
		(5,1,3)	(5,1,4)	(5,1,6)	(5,1,7)
Sum of costs in three column		200	230	220	230
Set up cost		500000	700000	600000	100000
Total cost		500200	700230	600220	100230
Corresp.Time		13	10	13	11

Triplet (5,1,7) is giving the least value so these three sites .i.e. 5 , 1 and 7 are selected.

Step 3.

Selection of warehouse potential sites for fourth time period.

There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.17: Total cost and time for three sites in 4^{th} time period in 2^{nd} iteration

		Warehouse potential sites
		(5,6,7)
Sum of costs in three column		320
Set up cost		600000
Total cost		600320
Corresp.Time		12

Total cost =15,00,990.

Max Time=12.

Iteration No .3 : Thus, the Tabu lists are given by

Table 3.18: Tabu search for 3^{rd} iteration

Tabu list for drop	Tabu list for add
$\{y_5, y_1\}$	$\{y_2\}$

Hence, the solution at this stage are

Step 1.

Table 3.19: Total cost and time for three sites in 1^{st} and 2^{nd} time period in 3^{rd} iteration

		Warehouse potential sites			
		(5,1,3)	(5,1,4)	(5,1,6)	(5,1,7)
Sum of costs in three column		630	Inf	680	640
Set up cost		1200000	Inf	700000	1000000
Total cost		1200630	Inf	700680	1000640
Corresp.Time		13	Inf	13	11

Combination of sites 5, 1, and 6 gives the least cost. So these sites are selected for three time periods.

Step 2.

Selection of warehouse potential sites for fourth time period. There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.20: Total cost and time for three sites in 3rd and 4th time period in 3rd iteration

		Warehouse potential sites
		(5,6,7)
Sum of costs in three column		320
Set up cost		500000
Total cost		500320
Corresp.Time		12

Total cost = 12,01,000

Max Time=13

Iteration No .4 : Thus, the Tabu lists are given by

Table 3.21: Tabu search for 4th iteration

Tabu list for drop	Tabu list for add
$\{y_1, y_6\}$	$\{y_5\}$

Hence, the solution at this stage are

Step 1.

Table 3.22: Total cost and time for three sites in 1st and 2nd time period in 4th iteration

		Warehouse potential sites				
		(1,6,2)	(1,6,3)	(1,6,4)	(1,6,5)	(1,6,7)
Sum of costs in three column		460	690	Inf	680	720
Set up cost		600000	1000000	Inf	700000	800000
Total cost		600460	1000690	Inf	700680	800720
Corresp.Time		13	12	Inf	13	13

Since least cost is of (1,6,2).So sites selected are 1,6 and 2.

But I had already get this solution. So I will stop at this point.

Efficient Solutions :

Table 3.23: Efficient Solutions

Efficient Solution	Variables xij's and yj's at level 1				Total cost of meeting requirements	Duration of meeting requirements
	I	II	III	IV	$C(X^1)$	$T(X^1)$
(X^1,Y^1)	$x_{12},x_{22},x_{36},x_{42},x_{55},y_6,y_2,y_5$	$x_{12},x_{22},x_{35},x_{46},x_{55},y_6,y_2,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_5,y_6,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{55},y_5,y_6,y_7$	10,01,080	13

Change t_{ij}'s by M(large positive number),t_{ij}'s ≥ 13

Table 3.24: Data Representation after removing time ≥ 13 for 1st time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	40	30	50	120	180	140	150
	8	2	11	6	10	9	7
2	70	80	130	130	170	140	20
	6	6	9	7	8	10	11
3	80	200	200	70	140	70	60
	8	9	5	8	11	12	10
4	60	10	**M**	80	30	160	210
	11	6	**M**	11	9	8	6
5	90	**M**	20	60	40	50	**M**
	10	**M**	8	12	6	6	**M**
SetUpCost©	100000	300000	700000	800000	400000	200000	500000

Table 3.25: Data Representation after removing time ≥ 13 for 2nd time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	50	40	40	120	180	170	120
	9	3	11	6	10	9	7
2	80	80	100	150	200	140	200
	6	6	7	6	11	10	5
3	80	90	20	20	30	**M**	60
	11	10	11	8	9	**M**	10
4	50	90	40	50	140	40	**M**
	6	10	6	6	11	4	**M**
5	10	130	70	160	**M**	80	210
	6	12	8	8	**M**	11	6
SetUpCost©	100000	400000	600000	800000	600000	500000	600000

Table 3.26: Data Representation after removing time ≥ 13 for 3rd time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	20	160	20	100	90	80	160
	8	8	11	7	10	6	8
2	80	170	120	60	40	130	60
	11	9	6	12	10	12	12
3	80	50	100	50	**M**	210	60
	6	6	6	9	**M**	6	10
4	90	180	40	140	200	70	80
	10	10	11	11	11	5	11
5	230	140	120	240	30	20	**M**
	11	10	7	3	9	11	**M**
SetUpCost©	600000	800000	500000	700000	400000	600000	100000

Table 3.27: Data Representation after removing time ≥ 13 for 4th time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	140	140	170	120	120	50	60
	10	10	9	7	6	6	12
2	40	20	90	40	160	30	160
	11	11	10	11	8	9	8
3	10	80	10	80	80	60	200
	6	6	6	6	11	10	5
4	50	20	60	180	**M**	210	150
	9	11	12	11	**M**	6	9
5	180	20	100	200	30	40	**M**
	10	8	7	11	12	3	**M**
SetUpCost©	400000	100000	500000	800000	700000	600000	500000

2nd non dominated solution :

Step 1. Add the entire costs *column wise* and the least of the total costs thus selected.

Table 3.28: Total cost and time of single site for 2nd non dominated solution

		Warehouse potential sites						
		1	2	3	4	5	6	7
Sum of costs in single column		1110	M	M	Inf	M	M	M
Set up cost		100000	300000	700000	Inf	400000	200000	500000
Total cost		101110	M	M	Inf	M	M	M
Corresp.Time		11	M	M	Inf	M	M	M

As the least cost is of first site so first site is selected.

Step 2. Selection of two warehouse potential sites at a time.

Table 3.29: Total cost and time of second site for 2nd non dominated solution

		Warehouse potential sites					
		(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)
Sum of costs in two column		540	800	Inf	740	880	970
Set up cost		400000	800000	Inf	500000	300000	600000
Total cost		400540	800800	Inf	500740	300880	600970
Corresp.Time		11	11	Inf	11	12	12

As the pair sixth got the least value so these two sites i.e. 1 and 6 are selected.

Step 3. Selection of three facility potential sites at a time.

Table 3.30: Total cost and time of third site for 1st and 2nd time period

		Warehouse potential sites				
		(1,6,2)	(1,6,3)	(1,6,4)	(1,6,5)	(1,6,7)
Sum of costs in three column		460	690	Inf	690	700
Set up cost		600000	1000000	Inf	700000	800000
Total cost		600460	1000690	Inf	700690	800700
Corresp.Time		12	12	Inf	12	12

Triplet (1, 6, 2) is giving the least value so these three sites .i.e. 1,6 and

2 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 4. Selection of warehouse potential sites for third time period.

Table 3.31: Total cost and time of third site for 3rd time period

		Warehouse potential sites			
		(1,6,3)	(1,6,4)	(1,6,5)	(1,6,7)
Sum of costs in three column		240	220	230	230
Set up cost		500000	700000	400000	100000
Total cost		500240	700220	400230	100230
Corresp.Time		11	12	11	12

As the triplet(1,6,7) gives the minimum cost so sites 1 , 6 and 7 are selected for time period III.

Step 5. Selection of warehouse potential sites for fourth time period.

There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.32: Total cost and time of third site for 4th time period

		Warehouse potential sites
		(5,6,7)
Sum of costs in three column		320
Set up cost		700000
Total cost		700320
Corresp.Time		12

Therefore

Total Cost = 14,01,010.

Max Time = 12.

Iteration No .1 : The locations selected for setting up of warehouses are 1, 2 and 5. Thus, the Tabu lists are given

Table 3.33: Tabu Search for 1st iteration

Tabu list for drop	Tabu list for add
$\{y_2, y_6\}$	$\{y_1\}$

Hence, the solutions at this stage are

Step 1.

Table 3.34: Total cost and time for three sites for 1st and 2nd time period in 1st iteration

		Warehouse potential sites			
		(2,6,3)	(2,6,4)	(2,6,5)	(2,6,7)
Sum of costs in three column		460	Inf	500	470
Set up cost		1200000	Inf	900000	1000000
Total cost		1200460	Inf	900500	1000470
Corresp.Time		12	Inf	12	11

Triplet (2,6,5) is giving the least value so these three sites .i.e. 2 , 6 and 5 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 2.

Selection of warehouse potential sites for remaining time periods . Since sites 6 and 5 is available for remaining time period . Therefore find only one more site.

Table 3.35: Total cost and time for three sites for 3rd and 4th time period in 1st iteration

		Warehouse potential sites			
		(6,5,1)	(6,5,3)	(6,5,4)	(6,5,7)
Sum of costs in three column		230	220	260	590
Set up cost		600000	500000	700000	100000
Total cost		600230	500180	700260	100590
Corresp.Time		11	11	11	11

As triplets (6,5,7) has minimum cost therefore sites 6,5 and 7 is selected for third and fourth time periods.

Total cost = 10,01,090.

Max Time=12

Iteration No .2 :. Thus, the Tabu lists given are

Table 3.36: Tabu Search for 2nd iteration

Tabu list for drop	Tabu list for add
$\{y_6, y_5\}$	$\{y_2\}$

Hence, the solutions at this stage are

Step 1.

Table 3.37: Total cost and time for three sites for 3rd time period in 2nd iteration

		Warehouse potential sites			
		(6,5,1)	(6,5,3)	(6,5,4)	(6,5,7)
Sum of costs in three column		700	860	Inf	1290
Set up cost		700000	1300000	Inf	1100000
Total cost		700630	1300860	Inf	1101290
Corresp.Time		12	12	Inf	12

As only sites available are 6 , 5 and 1 because all others pairs have very large time period . So, these three sites are selected for three time periods.

Step 2.

Selection of warehouse potential sites for fourth time period.

There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.38: Total cost and time for three sites for 4th time period in 2nd iteration

		Warehouse potential sites
		(5,6,7)
Sum of costs in three column		320
Set up cost		500000
Total cost		500320
Corresp.Time		12

Total Cost = 12,00,950.

Max Time = 12.

Iteration No .3 :. Thus, the Tabu lists given are

Table 3.39: Tabu Search for 3rd iteration

Tabu list for drop	Tabu list for add
$\{y_5, y_1\}$	$\{y_6\}$

Hence, the solutions at this stage are

Step 1.

Table 3.40: Total cost and time for three sites for 1st and 2nd time period in 3rd iteration

		Warehouse potential sites			
		(5,1,2)	(5,1,3)	(5,1,4)	(5,1,7)
Sum of costs in three column		440	640	Inf	640
Set up cost		800000	1200000	Inf	1000000
Total cost		800440	1200640	Inf	1000640
Corresp.Time		9	11	Inf	11

Triplet (5,1,2) is giving the least value so these three sites .i.e. 5 , 1 and

2 are selected. But as site 2 is available for only two time period therefore these sites are selected for only two time periods.

I have to choose another triplet for remaining time periods.

Step 2

Selection of warehouse potential sites for remaining time periods . Since sites 3 and 7 is available for third time period . Therefore find only one more site.

Table 3.41: Total cost and time for three sites for 3rd time period in 3rd iteration

		Warehouse potential sites			
		(5,1,3)	(5,1,4)	(5,1,6)	(5,1,7)
Sum of costs in three column		210	230	230	230
Set up cost		500000	700000	600000	100000
Total cost		500210	700230	600230	100230
Corresp.Time		11	10	11	11

As triplets (5,1,7) has minimum cost therefore sites 5,1 and 7 is selected for third time period.

Step 3.

Selection of warehouse potential sites for fourth time period.

There are only three sites available in fourth period i.e. 5 , 6 , and 7 .

Table 3.42: Total cost and time for three sites for 4th time period in 3rd iteration

		Warehouse potential sites
		(5,6,7)
Sum of costs in Three column		320
Set up cost		600000
Total cost		600320
Corresp.Time		12

Total cost = 15,00,900.

Max Time=12

Iteration No .4 :. Thus, the Tabu lists given are

Table 3.43: Tabu Search for 4th iteration

Tabu list for drop	Tabu list for add
$\{y_1,y_2\}$	$\{y_5\}$

Hence, the solutions at this stage are

Step 1.

Table 3.44: Total cost and time for three sites for 1st and 2nd time period in 4th iteration

		Warehouse potential sites			
		(1,2,3)	(1,2,4)	(1,2,6)	(1,2,7)
Sum of costs in three column		400	Inf	480	450
Set up cost		1100000	Inf	700000	900000
Total cost		900400	Inf	700480	900450
Corresp.Time		11	Inf	12	11

Triplet (1,2,6) is giving the least value so these three sites .i.e. 1 , 2 and 6 are selected .Since 1, 2 and 6 is the solution where started from, so in order to stop the recycling, stop the iteration process.1

Efficient Solutions :

Table 3.45: Efficient Solutions

Efficient Solution	Variables xij's and yj's at level 1				Total cost of meeting requirements	Duration of meeting requirements
	I	II	III	IV	$C(X^1)$	$T(X^1)$
(X^1,Y^1)	$x_{12},x_{22},x_{36},x_{42},x_{55},y_6,y_2,y_5$	$x_{12},x_{21},x_{35},x_{41},x_{51},y_1,y_2,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_5,y_6,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{55},y_5,y_6,y_7$	10,01,080	13
	I	II	III	IV	$C(X^2)$	$T(X^2)$
(X^2,Y^2)	$x_{12},x_{22},x_{36},x_{42},x_{55},y_2,y_6,y_5$	$x_{12},x_{22},x_{35},x_{46},x_{56},y_2,y_6,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_6,y_5,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{55},y_6,y_5,y_7$	10,01,090	12

Change t_{ij}'s by M(large positive number),t_{ij}'s ≥ 12

Table 3.46: Data Representation for time ≥ 12 for 1st time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	40	30	50	120	180	140	150
	8	2	11	6	10	9	7
2	70	80	130	130	170	140	20
	6	6	9	7	8	10	11
3	80	200	200	70	140	**M**	60
	8	9	5	8	11	**M**	10
4	60	10	**M**	80	30	160	210
	11	6	**M**	11	9	8	6
5	90	**M**	20	**M**	40	50	**M**
	10	**M**	8	**M**	6	6	**M**
SetUpCost©	100000	300000	700000	800000	400000	200000	500000

Table 3.47: Data Representation for time ≥ 12 for 2nd time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	50	40	40	120	180	170	120
	9	3	11	6	10	9	7
2	80	80	100	150	200	140	200
	6	6	7	6	11	10	5
3	80	90	20	20	30	**M**	60
	11	10	11	8	9	**M**	10
4	50	90	40	50	140	40	**M**
	6	10	6	6	11	4	**M**
5	10	**M**	70	160	**M**	80	210
	6	**M**	8	8	**M**	11	6
SetUpCost©	100000	400000	600000	800000	600000	500000	600000

Table 3.48: Data Representation for time ≥ 12 for 3rd time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	20	160	20	100	90	80	160
	8	8	11	7	10	6	8
2	80	170	120	**M**	40	**M**	**M**
	11	9	6	**M**	10	**M**	**M**
3	80	50	100	50	**M**	210	60
	6	6	6	9	**M**	6	10
4	90	180	40	140	200	70	80
	10	10	11	11	11	5	11
5	230	140	120	240	30	20	**M**
	11	10	7	3	9	11	**M**
SetUpCost©	600000	800000	500000	700000	400000	600000	100000

Table 3.49: Data Representation for time ≥ 12 for 4th time period

Ration shops	Warehouse Potential Sites						
	1	2	3	4	5	6	7
1	140	140	170	120	120	50	**M**
	10	10	9	7	6	6	**M**
2	40	20	90	40	160	30	160
	11	11	10	11	8	9	8
3	10	80	10	80	80	60	200
	6	6	6	6	11	10	5
4	50	20	**M**	180	**M**	210	150
	9	11	**M**	11	**M**	6	9
5	180	20	100	200	**M**	40	**M**
	10	8	7	11	**M**	3	**M**
SetUpCost©	400000	100000	500000	800000	700000	600000	500000

3rd non dominated solution :

To obtain the 3rd non dominated solution, proceed in the same manner as defined above to obtain the incumbent solution

Efficient Solutions :

Table 3.50: Efficient Solutions

Efficient Solution	Variables xij's and yj's at level 1				Total cost of meeting requirements	Duration of meeting requirements
	I	II	III	IV	$C(X^1)$	$T(X^1)$
(X^1,Y^1)	$x_{12},x_{22},x_{36},x_{42},x_{55},y_6,y_2,y_5$	$x_{12},x_{21},x_{35},x_{41},x_{51},y_1,y_2,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_5,y_6,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{55},y_5,y_6,y_7$	10,01,080	13
	I	II	III	IV	$C(X^2)$	$T(X^2)$
(X^2,Y^2)	$x_{12},x_{22},x_{36},x_{42},x_{55},y_2,y_6,y_5$	$x_{12},x_{22},x_{35},x_{46},x_{56},y_2,y_6,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_6,y_5,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{55},y_6,y_5,y_7$	10,01,090	12
	I	II	III	IV	$C(X^3)$	$T(X^3)$
(X^3,Y^3)	$x_{12},x_{22},x_{35},x_{42},x_{55},y_6,y_2,y_5$	$x_{12},x_{22},x_{35},x_{46},x_{56},y_6,y_2,y_5$	$x_{16},x_{25},x_{37},x_{46},x_{56},y_6,y_5,y_7$	$x_{16},x_{26},x_{36},x_{47},x_{56},y_6,y_5,y_7$	10,01,170	11

3.6 Conclusion

In this chapter, concept of single objective dynamic facility location problem is introduced to uncapacitated multi-objective warehouse location problem with opening of new warehouses in different time periods. Algorithm proposed by Prakash *et al.* [22] is modified to find set of efficient solutions. The three efficient solutions are obtained for this problem.

REFERENCES

[1] Al-Sultan K S, Al-Fawzan M A. A tabu search approach to the uncapacited facility location problem. Annals of Operations Research 1999; 86: 91-103.

[2] Beasley J E. Lagrangian heuristics for location problems. European Journal of Operational Research 1993; 65: 383–399.

[3] Conn A R, Cornuejols G. A projection method for the uncapacitated facility location problem. Mathematical Programming 1990; 46: 273–298.

[4] Dias J, Captivo M.E, Climaco Joao. Efficient primal-dual heuristic for a dynamic location problem. Computer and Operation Research 2007; 34:1800-1823.

[5] Efroymson M.A, Ray T.L. A branch and bound algorithm for plant location. Operations Research 1966; 14: 361–368.

[6] Erlenkotter D. A comparative study of approaches to dynamic location problems. European Journal of Operational Research 1981; 6:133–43.

[7] Erlenkotter D. A dual-based procedure for uncapacitated facility location. Operations Research 1978; 14: 361–368.

[8] Gao L L, Robinson E P Jr. Uncapacitated facility location: General solution procedure and computational experience. European Journal of Operational Research 1994; 76: 410–427.

[9] Grefenstette J J. Optimization of control parameters for genetic algorithms. IEEE-SMC 1986; 16: 122-128.

[10] Gulsen M, Smith A E. A hierarchical genetic algorithm for system identification and curve fitting with a supercomputer implementation. Evolutionary Algorithms 1999; 111: 111-137.

[11] Ignizio JP. "A Heuristic Solution to Generalized Covering Problems", Ph.D. Dissertation, Virginia Polytechnic Institute, Blacksburg, 1971.

[12] Ignizio J P, Cavalier T M. Linear Programming. Prentice-Hall, Englewood Cliffs, New Jersey 1994; 457–505.

[13] Jacobsen S. Multiperiod capacitated location models. In: Mirchandani PB, Francis RL, editors. Discrete location theory. New York: Wiley Interscience; 1990. p. 173–207.

[14] Körkel M. On the exact solution of large-scale simple plant location problems. European Journal of Operational Research 1989; 39: 157–173.

[15] Krarup J, Pruzan P. Ingredients of locational analysis in discrete location. In: Mirchandani PB, Francis RL, editors. Discrete location theory. New York: Wiley Interscience; 1990. p. 1–54.

[16] Krarup J, Pruzan P. The simple plant location problem: Survey and Synthesis. European Journal of Operational Research 1983; 12: 36–81.

[17] Kratica J, Tosic D, Filipovic V and Ljubic I. Solving the simple plant location problem by genetic algorithm. RAIRO Operations Research 2001; 35: 127-142.

[18] Laporte G, Dejax P. Dynamic location-routing problems. Journal of the Operational Research Society 1989; 40:471–82.

[19] Maric M. An efficient genetic algorithm for solving multi-level uncapacitated facility location problem. Computing and Informatics 2010; 29: 183-201.

[20] Morris J G. On the extent to which certain fixed-charged depot location problems can be solved by LP. Journal of the Operational Research Society 1978; 29:71–6.

[21] Phogat A, Dwivedi V, Chauhan T, Saxena S, Agarwal P. Travelling salesman problem using genetic algorithm. IJCA journal 2012; 1: 25-30.

[22] Prakash S, Sharma M K, Singh A. Selection of warehouse sites for clustering ration shops to them with two objectives through a heuristic algorithm incorporating tabu search. OPSEARCH 2009; 46: 449-460.

[23] Rardin R L. Optimization in Operations Research. Pearson Education (Singapore) Pte. Ltd., Indian Branch, New Delhi 1998; 15–16: 670–705.

[24] Shulman A. An algorithm for solving dynamic capacitated plant location problems with discrete expansion sizes. Operations Research 1991; 39:423–36.

[25] Simao H P, Thizy J M. A dual simplex algorithm for the canonical representation of the un-capacitated facility location problem. Operations Research Letters 1989; 8: 279–286.

[26] Tohyama H, Ida K, Matsueda J. A genetic algorithm for uncapacitated facility location problem. Electronics and Communications in Japan 2011; 94.

[27] Van Roy T, Erlenkotter D. A dual-based procedure for dynamic facility location. Management Science 1982; 28:1091–105.

[28] Wesolowsky GO. Dynamic facility location. Management Science 1973; 19:1241–8.

[29] Wesolowsky G, Truscott W. The Multiperiod location-allocation problem with relocation of warehouses. Management Science 1975; 22:57–65.