Two Phase Genetic Approach to Single Source Capacitated Facility Location Problem

Sheetal Taneja[i], Monika Bisla[ii]

ABSTRACT

With rise in urbanization, problem of optimally allocating facilities to satisfy the needs of customers has gained importance. Most of the solutions proposed in literature have modelled the Single Source Capacitated Facility Location Problem (SSCFLP) as an optimization problem minimizing the cost. We have proposed a two phase solution to the problem. First phase is a pre-processing step which reduces the search space. Second phase applies genetic algoithm to obtain a solution for the problem. The proposed algorithm is tested on benchmark datasets taken from Delmaire et al., and the search space was found to converge to a nearly optimal result in most of the cases. Also, proposed solution with pre-processing phase is found to achieve effective results in comparison to approach without pre-processing step, both with respect to the solution obtained and number of generations.

KEYWORDS

Genetic Algorithm, Single Source Capacitated Facility Location Problem

I. INTRODUCTION

With rise in urbanization, problem of optimally allocating facilities to satisfy the needs of customers has gained importance. The problem is categorized as a spatial decision problem [1], which requires identifying the location and allocation of facilities. This problem has found applications in various domains such as telecommunication networks, distributed systems. The problem is categorized into two types on the basis of finiteness of the available capacity of facility involved, namely capacitated and uncapacitated. In capacitated facility location problem, each facility is assumed to have a finite capacity, and thus can serve the demand of finite number of customers. Whereas in uncapacitated facility location problem, each facility is assumed to have infinite capacity, and hence can serve the demand of infinite number of customers. Again the problem can be further categorized as single source and multi-source. In single source facility location problem, customers' demand can be satisfied by only single source, and hence each customer is assigned to only one facility. However, in multi-source facility location problem, a customer can avail the services of multiple facilities to meet his/her demand. In most real world scenarios, capacity of facility being finite, Single Source Capacitated Facility Location Problem (SSCFLP) becomes important. This problem is NP-hard combinatorial optimization problem [2].

The challenge in this problem is to determine which facilities should be opened so that the total demand of all the customers taken together is satisfied. Opening up a facility involves fixed cost of setting up the facility, and the variable cost which denotes the travel cost incurred by the customers to avail the facility. Thus, minimizing cost is also a challenge. In other words, the optimization issue in SSCFLP is to find minimum number of facilities so that the total demand of all the customers taken together is satisfied while the total operational cost of all the facilities concerned is minimized.

Most of the work in literature has focused on Single Source Capacitated Facility Location Problem (SSCFLP) as an optimization problem minimizing the total operational cost. We have proposed two phase genetic approach for solving it:

First Phase- It is necessarily a pre-processing step to reduce the search space.
Second Phase- It employs genetic algorithm for finding a solution.

Results demonstrate that our approach is able to generate nearly optimal result in respective of the objective of minimum cost in most of the cases. Also, proposed solution with pre-processing phase is found to achieve effective results in comparison to approach without pre-processing step, both with respect to solution obtained and number of generations.

The remaining paper is organized as follows: in section 2, we discuss the related work, section 3 formally define the problem and describe the strategy proposed for solving the problem. Section 4 deals with the experiments performed on various data sets and the results obtained. Finally, section 5 gives the conclusion and section 6 gives the scope for further work.

II. RELATED WORK

Several approaches for solving SSCFLP have been put forward in literature. Most of the work is based on Lagrangian heuristics where main difference lies in the constraints that are relaxed. Klincewicz and Luss [3] relaxed capacity constraint and solved the resulting uncapacitated problem using dual ascent algorithm. Pirkul [4], Barcelo´ and Casanovas [5], and Sridharan [6] relaxed customer assignment constraint and solved the problem by finding solution to a number of knapsack problems. Pirkul's solutions were the best fesible solutions among them. Beasley [7] relaxed both customer assignment and capacity constraint and obtained good quality

[i] *Dyal Singh College, University of Delhi, New Delhi, India, Email: sheetaltaneja1@gmail.com*
[ii] *Microsoft India R&D Ltd, Email: monikabisla91@gmail.com*

Copyright ©IJICCT, Vol II, Issue I (Jan-Jun2014): ISSN 2347-7202

solutions for different location problems, and Agar and Salhi [8] improved Beasleys algorithm by proposing its modification based on two-phase approach for solving assignment problem.

For solving SSCFLP, Delmaire et. al. [10] proposed various heuristics based on evolutionary algorithms, GRASP (Greedy Randomized Adaptive Search Procedure), simulated annealing, and tabu search; and tested the proposed approach on the data sets used by Barcelo et. al. [9] in the his Lagrangian relaxation approach and found considerably better results. Delmaire et. al. [11] proposed Reactive GRASP (RGRASP) and tabu search based heuristics and two hybrid approaches that combine the two. The approach aimed at strengthening the local search and introducing the diversification mechanism and performed better than Delmaire et. al. [10] both with respect to quality of solution, and deviation from the value of best known solution.

Various other heuristics have also been proposed in literature. Ronnqvist¨ et. al. [12] proposed repeated matching heuristics and results obtained are shown to be the same and often better than the ones obtained using Lagrangian heuristics. Ahuja et. al. [13] proposed multi-exchange heuristics in which very Large Scale Neighborhood (VLSN) search algorithm is proposed. Contreras and D´ıaz [14] proposed a scatter search approach that also uses GRASP and tabu search, and provides an upper bound for the solution. Chen and Ting [15] proposed a hybrid algorithm which combines Lagrangian heuristic and Ant Colony System and found results competitive with other algorithms.

Some decomposition based approaches have also been proposed for SSCFLP. Neebe and Rao [16] formulated the problem as a set partitioning problem and introduced branch-and-bound algorithm for solving it. However, it can work only for small and medium size problems. Holmberg et. al. [17] incorporated repeated matching algorithm, Lagrangian heuristics, and branch-and-bound approach for solving SSCFLP. Diaz and Fernandez´ [18] proposed branch-and-price algorithm which incorporated column generation routine for finding lower and upper bound for the problem, and found satisfactory results. Yang et. al. [19] proposed a cut-and-solve approach which is a special case of branch-and-cut. This approach aims to provide solution to two types of problem (corresponds to two nodes) at each level, one is sparse problem whose solution provides upper bound and another is dense problem whose solution provides lower bound. Avella et. al. [20] proposed an approach that is based on generation of cutting planes for solving SSCFLP and reported its efficiency on large data sets.

A few genetic approaches have also been proposed. Jaramillo et. al. [21] and Cortinhal and Captivo [22] applied genetic approach (evolutionary approach) on the problem of SSCFLP which dealt with the single objective function of minimizing operational cost. They found genetic algorithms to be unsuitable for SSCFLP, however found its effectiveness only for uncapacitated version of the problem. Julstrom [24] used two permutation codings of plant locations and customers respectively, and found effectiveness of evolutionary algorithm based on selection and mutation operator. Harris et. al. [23] proposed algorithm that combines evolutionary multi-objective algorithm with Lagrangian heuristics, to solve SSCFLP which aims to minimize two objectives. First objective aims to minimize cost and the second objective is applicable in a special setting minimizing CO_2 emissions from transport. Our solution competes with already proposed approaches as it con-siderably reduces the search space and achieves comparative results by applying genetic based approach.

III. Problem Definition And Proposed Strategy

Let us consider a problem involving m customers and n facilities. Let a_i denotes the demand of i^{th} customer, b_j denotes the capacity of the j^{th} facility, f_j denotes the fixed establishment cost of j^{th} facility, and c_{ij} denotes the cost of assigning i^{th} customer to j^{th} facility. We define the binary decision variables:

$$y_i = \begin{cases} 0 & \text{otherwise:} \\ 1 & \text{if facility i is open;} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if customer i is assigned to facility j ;} \\ 0 & \text{otherwise:} \end{cases}$$

The SSCFLP can be formulated as given below:

$$\text{M inimize } F_1(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} c_{ij} + y_j \sum_{j=1}^{n} t_j \tag{1}$$

subject to

$$\sum_{i=1}^{m} x_{ij} a_i \quad b_j \ \text{ for } j = 1 ; 2 ; : : : ; n \tag{2}$$

$$\sum_{j}^{n} x_{ij} \quad 1 \text{ for } i = 1 ; 2 ; : : : ; m \tag{3}$$

The objective function $F_1(x)$ as also specified by Delmaire et. al. [11] describes the total operational cost. Also, while constraint (eq2) ensures that the total customer demand served by a facility does not exceed its capacity, constraint (eq3) ensures that each customer is assigned to at most one facility. In the latter constraint, there might be an unassigned customer since a customer demand may exceed capacity of any of the available facilities.

In the proposed strategy, we have used a pre-processing step to reduce the search space. The results obtained from the pre-processing step have been used to generate the initial population P_0. We then apply genetic algorithm to obtain the solution for the problem.

The flowchart in Figure 1 gives an overview of the proposed strategy comprising of first phase based on pre-processing and second genetic based phase.

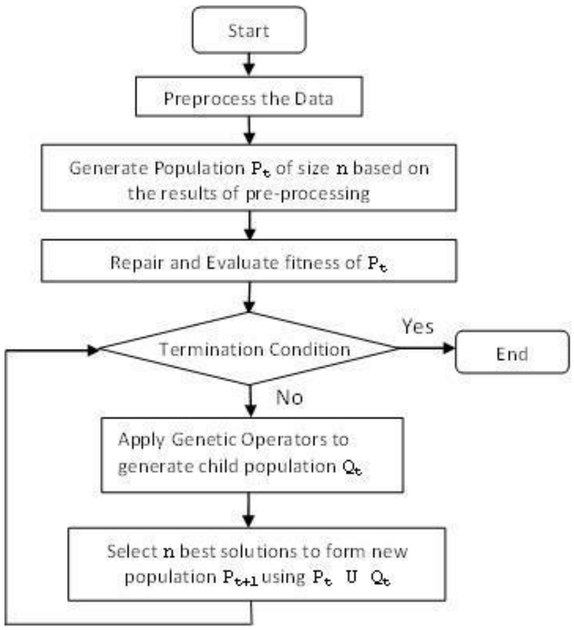


Fig. 1: Flowchart for proposed Algorithm

A. Phase 1: Pre-Processing

According to problem definition, a customer can be assigned to any one of the opened facilities which can satisfy its demand. Therefore, in a problem involving m customers and n facilities, search space is of the order of n^m. This makes the problem intractable for even moderate values of n and m. As the problem involves a variable cost component which depends on the distance between the customer and the source(facility), an optimal solution will generally assign customer to a facility which lies in close proximity(neighborhood). Thus, it makes sense if one can identify a set of neighborhood facilities for each of the given customers. This may reduce search space significantly as optimal solutions will generally have assignment of facilities from this set.

In proposed Pre-Processing phase, we try to identify the lower bound (minimum number of facilities) and upper bound (maximum number of facilities) that may be required to satisfy the total demand of all the customers. This helps us to establish the worst case and the best case bound for a given problem. This is achieved by computing the cumulative capacity of the facilities. One can compute the minimum number of facilities, say f_{min} that can satisfy the total demand, say T of all the customers by taking cumulative sum of capacities sorted in descending order of their capacity values. Similarly, maximum number of facilities, say f_{max} in worst case can be computed by taking cumulative sum of capacities arranged in ascending order. Assuming uniform distribution of customers to facility, a facility will have minimum m/f_{max} number of customers denoted by say C_{min}, where m denotes total number of customers. We identify group of C_{min} closest customers for j^{th} facility and compute its mean. Subsequent groups of C_{min} closest customers are only retained if the difference between the means of subsequent group and already added groups is less than the overall standard deviation, say std_j for the given facility j. This helps us to retain the closest set of customers in which the standard deviation(std_j) is less than overall standard deviation for the facility j. It has been experimentally observed that in the most optimal scenarios, the assignment of customers to the given facility is generally from this set. The above notion is inverted to identify the potential facilities for each customer. If the pre-processing step results in a scenario where no potential facility could be identified for a customer, then we randomly assign any f_{min} number of facilities to it. Potential set of facilities obtained in this pre-processing phase is used in genetic algorithm of second phase to reduce the search space. Outline of pre-processing step is given below:

Algorithm 1: Pre-Processing Step

Input:

1) m: number of customers, n: number of facilities
2) a: demand vector of customers
3) b: capacity vector of facilities
4) c: cost matrix conataining cost of assigning each customer to each facility

Output:

1) Set of potential facilities for each customer

```
Set total demand of customers, say T = Σ_{i=1}^{m} a_i.;
temp =b;
Arrange values in vector temp in ascending order.;
Find lowest k1 such that Σ_{j=1}^{k1} temp_j   T .Let f_max denotes k1.;
Find largest k2 such that Σ_{j=n}^{k2} temp_j   T . Let f_min denotes n  k2.;
C_min = m/f_max ;
for each facility j = 1 : n do
    d_j [1 : m] = c(j; :);
    Sort d_j in ascending order;
    Calculate standard deviation, say std_j of d_j .;
    Divide the cost vector d_j into groups of C_min.;
    Calculate mean of each group.;
    Add subsequent groups of C_min if the difference between
    the mean of subsequent group and already added groups is
    less than std_j . ;
end
Determine potential facilities that can be assigned for each customer;
If there is any customer without any facility assigned, randomly assign f_min facilities to it;
```

Customer Facility	Customer 1	Customer 2	Customer 3	Cutomer 4	Cutomer 5	Customer 6
Facility 1	10	4	7	18	20	23
Facility 2	8	10	15	10	13	6
Facility 3	24	18	6	5	7	8

TABLE I: Distance Cost Matrix

Given three facilities, consider the problem of allocating 6 customers. Tables I, II, and III give matrices denoting distance cost, customers' demand, and capacity and fixed cost of facilities respectively.

Customer	Customer 1	Customer 2	Customer 3	Cutomer 4	Cutomer 5	Customer 6
Demand	5	4	7	6	4	2

TABLE II: Customer Demand Matrix

Facility	Capacity	Fixed Cost
Facility 1	4	2
Facility 2	11	4
Facility 3	17	5

TABLE III: Capacity and Fixed Cost of Facility

Preprocessing Steps applied on above input tables I, II, and III with n = 3 and m = 6

1) Total Demand T = 28
2) temp = [4; 11; 17]
3) Sorted Capacity Values, temp = [4; 11; 17]
4) f_{max} = 3
5) f_{min} = 2
6) C_{min} = 6=3 = 2
7) a) For Facility 1
   Sorted Cost Vector = [4; 7; 10; 18; 20; 23]
   std = 7:7115
   First Group : [4; 7] , mean = 5:5
   Selected (* C_{min} is 2)
   Second Group: [10; 18], mean = 14
   Not Selected (* (14 5:5) > 7:7115)
   <u>Result</u> : Facility 1 is assigned to Customers 2 and 3.

   b) For Facility 2
   Sorted Cost Vector = [6; 8; 10; 10; 13; 15]
   std = 3:2660
   First Group : [6; 8] , mean = 7
   Selected (* C_{min} is 2)
   Second Group: [10; 10], mean = 10
   Selected (* (10 7) < 3:2660)
   Third Group: [13; 15], mean = 14
   Not Selected (* (14 8:5) > 3:2660)
   <u>Result</u> : Facility 2 is assigned to Customers 1, 2, 4 and 6.

   c) For Facility 3
   Sorted Cost Vector = [5; 6; 7; 8; 18; 24]
   std = 7:7889
   First Group : [5; 6] , mean = 5:5
   Selected (* C_{min} is 2) Second Group: [7; 8], mean = 7:5
   Selected (* (7:5 5:5) < 7:7889)
   Third Group: [18; 24], mean = 21
   Not Selected (* (21 6:5) > 7:7889)
   <u>Result</u> : Facility 3 is assigned to Customers 3, 4, 5 and 6.
8) Possible Facilities for each customer are:
   Customer 1: Facility 2
   Customer 2: Facility 1, Facility 2
   Customer 3: Facility 1, Facility 3
   Customer 4: Facility 2, Facility 3
   Customer 5: Facility 3
   Customer 6: Facility 2, Facility 3

B. Phase 2: Genetic Algorithm

Second phase of the proposed approach employs Genetic Algorithm. It converges to a solution starting from an initial random population of size N. Each individual in the population is represented by a chromosome having m genes representing the m customers in the problem. Each of the genes can take any value from the potential set of facilities obtained from pre-processing step. Thus, the complete chromosome denotes the assignment of facilities to all the customers. To cater for the rare scenario in which the pre-processing step may fail to identify the the potential set of facilities that may contain the optimal assignment, we have relaxed 10% of the initial population to explore the entire search space by taking up gene value from any of the available facilities.

Outline of Genetic Algorithm is given below:

Algorithm 2: Genetic Algorithm

Input:
1) m: number of customers, n: number of facilities
2) a: demand vector of customers
3) b: capacity vector of facilities
4) c: cost matrix containing cost of assigning each customer to each facility
5) NumGen: total number of generations

Output:
1) A facility assigned to each customer

Generate initial population P_t at t = 0 such that:;
- 90% of the population have random assignment to the customers from the facilities chosen in pre-processing step.
- Remaining population have random assignment to the customers considering entire search space.

while t < NumGenerations do
  Repair P_t for any constraint violations and calculate Fitness using a, b and c.;
  Perform tournament selection on P_t to get new P_t.;
  The population, P_t, is used to create child population Q_t (say), of size N by using genetic operators crossover and mutation, and repaired if needed.;
  P_t and Q_t are combined to get a population of size 2N, denoted by K_t (say), i.e. $K_t = P_t [Q_t$.;
  Select N best solutions with respect to cost objective to form new population P_{t+1} using K_t.;
  Increment the value of generation counter t.;
end

Having generated a popultaion of N chromosomes, each chromosome in the initial population is tested for capacity constraint violation and repaired if needed. Repairing process identifies the facility j whose capacity is insufficient to satisfy the total demand of all customers assigned to it. It then chooses a customer assigned to facility j whose demand and exceeding demand (difference of total demand and capacity) has minimum gap and reassign it to any randomly chosen potential facility obtained from first phase. This is followed until facility j has sufficient capacity to satify the demand of all customers assigned to it. However, a scenario may arise when the demand of customer to be reassigned is greater than the left over capacity of all potential facilities. To cater for this situation, facility that causes minimum cost addition is chosen from the set of all available facilities and is assigned to the customer. In this case, if none of the N facilities has the potential to satisfy the demand, reinitialization of few genes (1=5 m) of chromosomes are done, and the chromosome thus obtained is considered for repairing.

Next step of the algorithm computes the fitness of each chromosome with respect to objective function F_1. Chromo-somes are then selected from the parent population using tournament selection. From the selected chromosomes, N=2 pairs are formed randomly which undergo crossover process to generate N offsprings. Two point crossover has been used for generating child population because it allows solution space to be searched more thoroughly as suggested by Kaya et. al. [25]. Each chromosome in the child population is then mutated with probability p_m. Thus, number of genes mutated in an offspring chromosome are p_m noOfGenes. Mutated chromosome is retained only if it represents a better solution than the parent chromosome; otherwise the parent chromosome is retained for the next generation to preserve elitism. It should be noted that child population obtained after crossover and mutation is also repaired if needed. The next generation population of size N is then obtained by selecting best solutions with respect to cost objective from the union of current child and parent population. It should be noted that steps 3 to 8 shown in algorithm 2 are performed repeatedly until the termination condition (number of generations) is met.

Solution	Customer 1	Customer 2	Customer 3	Cutomer 4	Cutomer 5	Customer 6
Facility	2	1	3	3	3	2

TABLE IV: Solution Cost: 47

Tables IV gives the resultant chromosome obtained after Phase 2. It may be noted that optimal assignment for each customer is from the potential set of facilities obtained in pre-processing phase. The resultant chromosome represents a solution with the minimum cost.

IV. Experimentation

The proposed algorithm has been coded in MATLAB 2010R. It is tested on benchmark data sets taken from Delmaire et. al. [11]. These data sets alongwith optimal solution can be obtained from webpage[1].

We have investigated six data sets, each pertaining to 20 customers and 10 facilities. Each data set contains the following information: variable cost of assigning customer i to facility j, customers' demand, fixed cost of setting up facilities and the capacity of each facility. As mentioned in Barcelo [26], these data set are generated from a uniform distribution.

Experiments were conducted on these data sets with and without pre-processing step taking 0.1 as the mutation rate. Table V shows the comparison of results thus obtained. It may be noted that incorporating pre-processing phase before applying genetic algorithm leads to significant improvement with respect to cost minimization objective. Also, the number of generations taken for results to converge are less for proposed algorithm with pre-processing phase.

Problem	Without pre-processing		With pre-processing	
Data Set	Cost	Number of Generations	Cost	Number of Generations
P1	2117	89	2035	59
P2	4425	63	4374	20
P3	6199	65	6098	30
P4	7837	54	7313	30
P5	4638	89	4567	31
P6	2318	79	2269	27

TABLE V: Results with and without pre-processing phase

Table VI shows comparison of the result obtained by proposed algorithm with that of optimal results[1] given for the problem. The table lists cost and the facilities used for optimal solution and the solution obtained with proposed approach. It may be noted that our algorithm generates a solution which are comparable to the optimal result in most of the cases. However, for data sets P 6, proposed approach is able to attain optimal result.

Problem	Optimal Solution		Proposed Algorithm	
Data Set	Cost	Facilities Used	Cost	Facilities Used
P1	2014	2,3,4,5,7,8,9	2035	1,2,4,5,7,8,9,10
P2	4251	1,4,6,7,8,9,10	4374	1,3,4,7,8,9,10
P3	6051	1,3,4,5,7,9,10	6098	1,2,3,5,7,8,9,10
P4	7168	1,4,5,6,7,8,10	7313	1,3,4,6,7,8,10
P5	4551	1,2,4,5,6,7,9	4567	1,2,5,6,7,8,9
P6	2269	2,3,4,5,7,9,10	2269	2,3,4,5,7,9,10

TABLE VI: Result of optimal solution and proposed algorithm

The proposed approach is applied on each data set for several times. For each data set, boxplot plotted in Figure 2 shows the median and number of generations taken in converging to result for different runs.

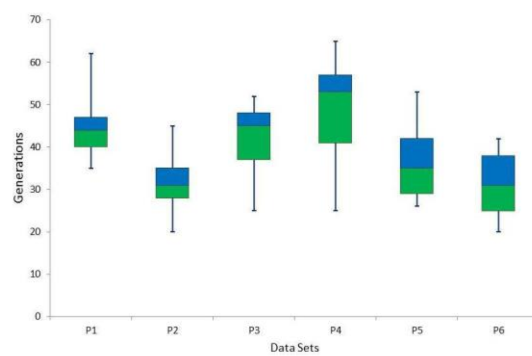
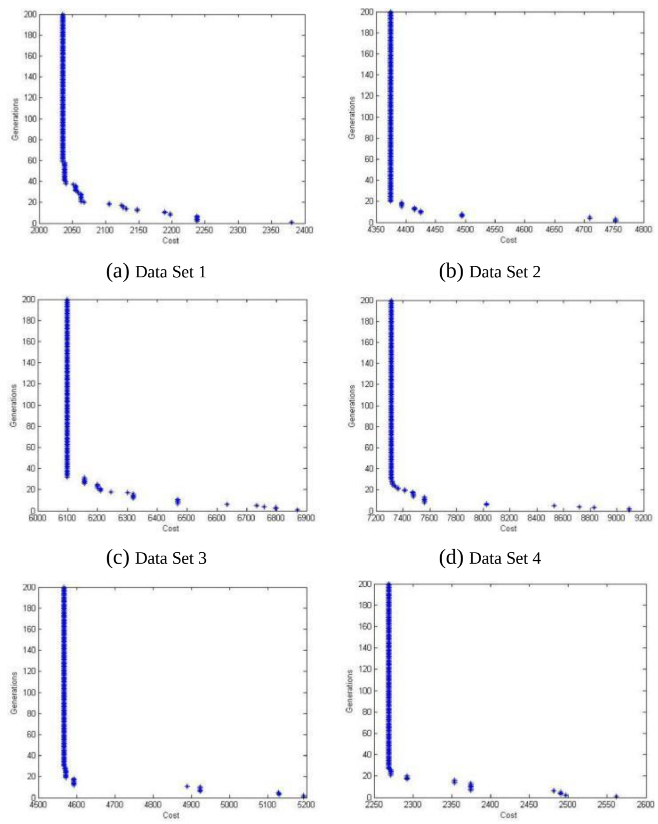


Fig. 2: Boxplot showing median and number of generations taken in converging to result for different runs

Graphs for result obtained using two-phase proposed approach are given in Figure 3. These graphs represent solution obtained and its convergence for data sets 1 to

[1] http://www-eio.upc.es/ elena/sscplp/

6. It may be noted that each graph depicts the number of generations and cost. With increase in number of generations, operational cost associated with the facility tends to decrease.



(a) Data Set 1
(b) Data Set 2
(c) Data Set 3
(d) Data Set 4
(e) Data Set 5
(f) Data Set 6

Fig. 3: Results for each data set

Table VII compares the result obtained with other solutions obtained by other techniques as given in Delmaire et. al. [11]. It may be noted that our algorithm is able to attain comparable results for the given six data sets.

Problem Set	Proposed Algorithm	GRASP	RGRASP
1	2035	2014	2014
2	4374	4289	4269
3	6098	6061	6051
4	7313	7168	7168
5	4567	4567	4551
6	2269	2269	2269

TABLE VII: Comparison of result obtained with other algorithms

V. Conclusion

The two-phase approach proposed for single source ca-pacitated facility location problem is able to find effcective results with respect to cost minimization objective. The pre-processing step that we have introduced helps in reducing the search space significantly, thereby leading to fast convergence to a solution in second phase based on genetic algorithm. Proposed algorithm achieves nearly optimal result in most cases. Also, results obtained with pre-processing phase are found to be better than the results obtained without pre-processing phase.

VI. Future Scope

The proposed approach may be incorporated to find so-lutions for large data sets. Also, one may explore parallel implementation of the algorithm while dealing with large data sets.

References

[1] Chakhar, S., & Mousseau, V. (2004). ”Towards a ty-pology of spatial decision problems”. ANNALES DU LAMSADE N 2 Juin 2004, 125.

[2] M. R. Garey and D. S. Johnson (1979).” Comput-ers and Intractability: A guide to the theory of NP-completeness”. W. H. Freeman and co., New York

[3] Klincewicz, J. G., and Luss, H. (1986). ”A Lagrangian relaxation heuristic for capacitated facility location with single-source constraints”. Journal of the Operational Research Society, 495-500.

[4] Pirkul, H. (1987). ”Efficient algorithms for the capaci-tated concentrator location problem”. Computers & Op-erations Research, 14(3), 197-208.

[5] Barcelo,´ J., & Casanovas, J. (1984). ”A heuristic La-grangean algorithm for the capacitated plant location problem”. European Journal of Operational Research, 15(2), 212-226.

[6] Sridharan, R. (1993). ”A Lagrangian heuristic for the capacitated plant location problem with single source constraints”. European Journal of Operational Research, 66(3), 305-312.

[7] Beasley, J. E. (1993). ”Lagrangean heuristics for location problems”. European Journal Of Operational Research, 65(3), 383-399.

[8] Agar, M. C., & Salhi, S. (1998). ”Lagrangean heuristics applied to a variety of large capacitated plant location problems”. Journal of the Operational Research Society, 49(10), 1072-1084.

[9] Barcelo, J., Fernandez´ E., and Jornsten Kurt O. (1991). ”Computational results from a new Lagrangean relax-ation algorithm for the capacitated plant location prob-lem.” European Journal of Operational Research, 53(1), 38-45.

[10] Delmaire H, D´ıaz JA, Fernandez´ E, Ortega M. (1997). ”Comparing new heuristics for the pure integer capaci-tated plant location problem. ”. Techn Report Dept Statist and Oper Res Univ Politecn Catalunya, Barcelona, no. DR97/10

[11] Delmaire, H., D´ıaz, J. A., Fernandez,´ E. L. E. N. A., & Ortega, M. A. R. U. J. A. (1999). ”Reactive GRASP and tabu search based heuristics for the single source capacitated plant location problem”. Infor-Information Systems and Operational Research, 37(3), 194-225.

[12] Ronnqvist,¨ M., Tragantalerngsak, S., & Holt, J. (1999). ”A repeated matching heuristic for the single-source capacitated facility location problem”. European Journal of Operational Research, 116(1), 51-68.

[13] Ahuja, R. K., Orlin, J. B., Pallottino, S., Scaparra, M. P., & Scutella, M. G. (2004). ”A multi-exchange heuristic for the single-source capacitated facility location prob-lem”. Management Science, 50(6), 749-760.

[14] Contreras, I. A., & D´ıaz, J. A. (2008). ”Scatter search for the single source capacitated facility location problem”. Annals of Operations Research, 157(1), 73-89.

[15] Chen, C. H., & Ting, C. J. (2008). ”Combining lagrangian heuristic and ant colony system to solve the sin-gle source capacitated facility location problem”. Trans-portation research part E: logistics and transportation review, 44(6), 1099-1122.

[16] Neebe, A. W., & Rao, M. R. (1983). ”An algorithm for the fixed-charge assigning users to sources problem”. Journal of the Operational Research Society, 1107-1113.

[17] Holmberg, K., Ronnqvist,¨ M., & Yuan, D. (1999). ”An exact algorithm for the capacitated facility location problems with single sourcing”. European Journal of Operational Research, 113(3), 544-559.

[18] Diaz, J. A., & Fernandez,´ E. (2002). ”A branch-and-price algorithm for the single source capacitated plant location problem”. Journal of the Operational Research Society, 728-740.

[19] Yang, Z., Chu, F., & Chen, H. (2012). ”A cut-and-solve based algorithm for the single-source capacitated facility location problem”. European Journal of Operational Research, 221(3), 521-532.

[20] Avella, Pasquale, Maurizio Boccia, & Sara Mattia (2013). ”A Branch-and-Cut algorithm for the Single Source Ca-pacitated Facility Location problem”. Advanced Logistics and Transport (ICALT), 2013 International Conference on. IEEE.

[21] Jaramillo, J. H., Bhadury, J., & Batta, R. (2002). ”On the use of genetic algorithms to solve location problems”. Computers & Operations Research, 29(6), 761-779.

[22] Cortinhal, M. J., & Captivo, M. E. (2004). ”Genetic algorithms for the single source capacitated location problem”. In Metaheuristics: computer decision-making ,187-216, Springer US.

[23] Harris, I., Mumford, C. L., & Naim, M. M. (2011, July). ”An evolutionary bi-objective approach to the capacitated facility location problem with cost and CO 2 emissions”. In Proceedings of the 13th annual conference on Genetic and evolutionary computation ,697-704, ACM.

[24] Julstrom, B. A. (2008, July). ”An evolutionary algorithm for some cases of the single-source constrained plant location problem”. In Proceedings of the 10th annual conference on Genetic and evolutionary computation, (pp. 607-608), ACM.

[25] Kaya, Y., & Uyar, M. (2011). ”A novel crossover operator for genetic algorithms: ring crossover”. arXiv preprint arXiv:1105.0355.

[26] Barcelo, J. 1985. ”Computational experiments with location problems using automatic constraint generation”. Working paper RR85/06 FIB, Universitat Politecnica de Catalunya