International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 12 Issue: 1
***Article Received:** 27 August 2023 **Revised:** 12 December 2023 **Accepted:** 3 January 2024*

How Good Is Local Search for Capacitated Facility Location Problem: An Experimental Study

Manisha Bansal[1], Seema Aggarwal[2], Geeta Aggarwal[3*]
[1]Indraprastha College for Women, Delhi, India.
[2]Miranda House, Delhi, India.
[3*]PGDAV College, Delhi, India.
mbansal@ip.du.ac.in[1]
geeta.gupta@pgdav.du.ac.in[3] *Corresponding Author
seema.aggarwal@mirandahouse.ac.in[2]

Abstract
Facility location problems have been widely studied since 1960's. These problems are known to be strongly NP-hard. In capacitated variant of the problem, a capacity constraint is associated with each facility. Capacitated facility location problem (CFLP) instances can be solved exactly using existing MILP solvers but only for small instance sizes. As the size of the problem instance increases beyond few hundred facilities and few hundred clients, it becomes prohibitive to solve these instances exactly. For large problem instances, therefore, other solution methods are used. One approach is to use heuristic methods. These methods usually give good solutions in reasonable time but they do not provide any guarantee about the quality of the solution. Somewhere between these two extremes exist another class of algorithms called approximation algorithms. They also provide only suboptimal solutions to the problem, like heuristic algorithms, in polynomial time. How ever they guarantee worst case upper bounds on the cost of the solution. So, a solution obtained using an approximation algorithm is guaranteed to have its cost between the optimal cost and the upper bound. We present experimental studies done with a local search based approximation algorithm for CFLP given by Bansal *et al.* [1] to show that this algorithm performs well in practice.

Keywords: Facility Location Problem, Local Search, Approximation Algorithm, Experimental Study

1 Introduction

Many organisations need to take decisions regarding placement of various kinds of facilities so that the customers/users having demands for those facilities can be served efficiently. Efficiency might be in terms of time needed to serve a demand, distance a customer has to travel to fulfill its demand or some other measure of cost. Examples are of wide range. A facility might be a supermarket store which needs to be located at strategic locations so that cost of establishing the desired infrastructure is not too much and the location should be such which can be accessed by many customers/clients. This is possible if the neighborhood of the location is densely populated. Another example could be the selection of base stations for wireless services. The network operator company would consider locations which maximizes their revenue. A government organisation might have to decide the locations for schools, hospitals, and other such utilities so that a large number of citizens can be benefited from it. All these are examples of facility location problem. What is common among all these examples is that a location for a facility needs to be identified (set of potential locations may be fixed). The goal is to minimize the cost incurred (if any) in setting up facility at the selected location with an objective to meet the demand of customers in best possible manner. Bhattacharya *et al.* [2] consider different factors that are important for selecting the facility location for different types of warehouses.

Now consider a specific example in which there is a need to set up wired LANs to satisfy the connectivity needs of an institution. Let us assume that the switches are facilities which facilitate connections among the computers connected through that switch. Each switch has a limited number of slots available which restricts the number of computers that can be connected through it. Since the facilities (switches) have capacities (number of slots) that put a constraint on the number of clients (computers) it can serve, it makes an instance of *capacitated facility location problem (CFLP)*.

More formally, in capacitated facility location, each facility $i \in F$ *has* a capacity u_i specifying the maximum amount of demand it can serve. There are two variants of this problem: CFLP with unsplittable demands (all the demand of a client must be served by the same facility) and CFLP with splittable demands (demand of a client can be split and assigned to multiple open facilities). The first variant is even hard to approximate unless P = NP, as shown by Bateni and Hajiaghayi [3]. When capacities of all the facilities are same, the problem is known as *uniform capacitated facility location problem (UCFLP)*. Rightly so,

***Article Received:** 27 August 2023 **Revised:** 12 December 2023 **Accepted:** 3 January 2024*

when capacities are not necessarily the same, it is called *non-uniform capacitated facility location problem* or just *capacitated facility location problem*.

The problem variant with splittable demands can be formulated as the following mixed integer linear program (MILP), wherein x_{ij} variables are allowed to be non- integral to capture the splittable nature of demands. f_i denotes the cost of opening a facility $i \in F$ and c_{ij} *denotes* the connection cost between a facility i and a client $j \in C$. d_j denotes the demand of a client j. Connection costs satisfy metric property.

$$\begin{aligned} \min \quad & \sum_{i\in F} f_i y_i + \sum_{i\in F}\sum_{j\in C} c_{ij} d_j x_{ij} \\ \text{s.t.} \quad & \\ & x_{ij} \le y_i \qquad \forall\, i \in F, j \in C \\ & \sum_{i\in F} x_{ij} = 1 \qquad \forall\, j \in C \\ & \sum_{j\in C} d_j x_{ij} \le u_i y_i \qquad \forall\, i \in F \\ & x_{ij} \ge 0 \qquad \forall\, i \in F, j \in C \\ & y_i \in \{0,1\} \qquad \forall\, i \in F \end{aligned}$$

A feasible solution to the above MILP is given by a set $F' \subseteq F$ and an assignment of the clients to the facilities in F' which obeys the capacity constraints where F' is the set of facilities i for which $y_i = 1$. Note that in CFLP clients cannot always be assigned to the nearest open facilities as it may lead to violation of capacity constraints. Once F' is known, best assignment of clients can be found in polynomial time by solving an assignment problem. Thus, any solution to CFLP is completely defined by the set of open facilities.

Capacitated facility location problem instances can be solved exactly using existing MIP solvers but only for small instance sizes. As the size of the problem instance increases beyond few hundred facilities and few hundred clients, it becomes prohibitive to solve these instances exactly. For large problem instances, therefore, other solution methods are used. One approach is to use heuristic methods. These methods usually give good solutions in reasonable time but they do not provide any guarantees about the quality of the solution. Many heuristics techniques have been applied to solve this problem in suboptimal manner since 1960's [4–8].

Somewhere between these two extremes exist another class of algorithms called approximation algorithms. They also provide only suboptimal solutions to the problem like heuristic algorithms in polynomial time. However, they provide worst case upper bounds on the cost of solution. So a solution obtained using an approximation algorithm is guaranteed to have its cost between the optimal cost and the upper bound. Bansal *et al.* [1] presented a local search based approximation algorithm for this problem. We'll call it *5-approx* in the rest of the paper. We present the experimental studies done on 5-approx to show that it performs well in practice. The rest of the paper is organised as follows: in section 2 we give a brief account of the heuristics used for CFLP. In section 3 we briefly discuss 5-approx and then provide the experimental studies done on the algorithm.

2 Heuristics used for CFLP

Various heuristic methods have been used to solve CFLP. Kuehn and Hamburger [4] gave a *local search heuristic* which is also known as Add-Drop-Interchange heuristic. This is the earliest heuristic method used for CFLP. Given an instance, an initial feasible solution is tried for improvements by considering certain local search moves whose number is polynomial in the size of the problem. In the year 1998 Korupolu *et al.* [9] showed that this heuristic is an 8-factor approximation algorithm for the problem when capacities are all uniform. Various other heuristics, based on local search, have been designed in late 1990's and later for CFLP. All these have been proved to have good approximation guarantees.

Another heuristic approach that has been popular for solving CFLP are *Lagrangean heuristics*. Such heuristics involve firstly solving a Lagrangean relaxation

LP relaxation of the MILP formulation of the problem is first solved. Problem Variables are then rounded to 0 or 1 suitably with some probability. This procedure provides a set of open facilities say S. Set of facilities in S together with assignments of clients to facilities in S is the solution for the problem obtained with this heuristic. Barahona *et al.* [12] give one such heuristic for the problem.

Of all the heuristic techniques used for CFLP, local search seems to be the most promising since we are able to get approximation factors using this technique. Lagrangean heuristics, though known to give good solutions in reasonable time, do not provide any guarantees about the quality of the solution. Local search algorithms are the only known algorithms for CFLP which provide good quality solutions theoretically as well as practically.

3 Local search for CFLP

A local search procedure can be described as follows:

1. Consider an initial feasible solution say S with cost $C(S)$.
2. Transform S into S' by making small changes to S. S' must also be feasible
3. If $C(S') < C(S)$ then replace S by S' and repeat from step 2.
4. If no such S' can be found then stop.

In this section, we briefly present the local search approximation algorithm given by Bansal *et al.* [1]. The algorithm provides a 5 approximation factor for the capacitated facility location problem. For a given set of facilities $S \subseteq F$, the optimal assignment of clients to the facilities in S can be done by solving a mincost flow

ISSN: 2321-8169 Volume: 12 Issue: 1
***Article Received:** 27 August 2023 **Revised:** 12 December 2023 **Accepted:** 3 January 2024*

problem. Therefore, we only need to determine a good subset $S \subseteq F$ of facilities.

The cost of the solution S is denoted by $c(S) = c_f(S) + c_s(S)$, where $c_f(S)$ is the facility cost and $c_s(S)$ is the service cost of the solution S.

Bansal *et al.* [1] suggested a local search algorithm to find a good approximate solution for the problem. Starting with a feasible solution S the following operations are performed to improve the solution if possible.

- **add(s):** $S \leftarrow S \cup s$, $s \notin S$. In this operation a facility s which is not in the current solution S is added if its addition improves the cost of the solution.
- **mopen(t, T):** $S \leftarrow (S \cup t) \setminus T$, $t \notin S, T \subseteq S$. In this operation a facility $t \notin S$ is opened and a subset of facilities $T \subseteq S$ is closed.
- **mclose(s, T):** $S \leftarrow (S \cup T) \setminus s, s \in S, T \subseteq F \setminus S$. In this operation a facility $s \in S$ is closed and a subset of facilities T (disjoint from S) is opened.
- **mmulti(r, R, t, T):** $S \leftarrow (S \cup R \cup t) \setminus (r \cup T), r \in S \setminus T, T \subseteq S \setminus r, R \subseteq F \setminus (S \cup t), t \notin S \cup R$. This operation is essentially a combination of a $\mathtt{mclose}(r, R)$ and $\mathtt{mopen}(t, T)$ with the added provision that clients served by r may be assigned to facility t.

S is locally optimal if none of the four operations improve the cost of the solution and at this point the algorithm stops. Polynomial running time can be ensured at the expense of an additive ϵ in the approximation factor by doing a local search operation only if the cost reduces by more than a $1-\epsilon/5n$ factor, for $\epsilon > 0$.

In section 2 we discussed various heuristic approaches for CFLP of which local search heuristic looks good. Next, we present experimental studies for 5-approx on data sets used in earlier studies of heuristics for CFLP, both benchmark instances [5] and random instances [7] [12]. 5approx provides solutions which are within (1 + 0.15) factor of the optimal solution for all the instances tested.

4 Data sets used

We performed experiments on benchmark instances as well as random instances which have been used in earlier studies.

Benchmark data sets

These data sets essentially include the standard data sets given in Akinc *et al.* [5], for the capacitated warehouse location problem. The benchmark instances tested for solution quality are taken from OR library. These instances are such that each facility has same capacity and facility cost.

Random data sets used

We performed experiments on three types of random problem instances: *type A*, *type B* and *type C*. To construct problem instances of type A, we used the procedure as in [7] [12] which is as follows:

1. For a problem instance of size $n \times m$, where n is the number of facilities and m is the number of clients, points are generated uniformly at random in a unit square to represent this many facilities and clients.
2. Compute euclidean distances between every point representing a facility say i and every point representing a client say j and multiply these distances by 10.
3. Demands for each client are generated from interval [5,35] uniformly at random i.e. from $U[5, 35]$.
4. s_j is generated from the interval [10,160] uniformly at random.
5. Capacity for a facility i is generated from interval [10,160] uniformly at random.
6. Facility costs are computed to reflect economies of scale using the formula
   $f_i = U[0, 90] + U[100, 110]\sqrt{s_j}$

Problem instances of type B are constructed by modifying step 2 of the above procedure. The euclidean distances computed are multiplied by 100 and for type C instances these distances are multiplied by 1000. For the problem instances of type A, facility cost component of a solution dominates the cost of the solution. For problem instances of type C, it is the service cost component that dominates the cost. Type B instances are somewhere in between the two types of instances. We observe that for a problem of a given size, instances of type A take largest amount of time to reach local optimal as compared to type B or type C instances.

Experiments

We computed optimal solution using LINGO 13 optimization software from LINDO Systems Inc. We give the % error i.e. percentage by which the locally optimal solution differs from the optimal solution, thereby giving the quality of the solution produced by the algorithm.

Table 1 provides the results for the benchmark instances. Results for these instances are within 1+ 0.10 factor of the optimal solution.

Next, we give our computational experience for *very small and small* instances.

1. Very small data instances are of sizes 50 × 50, 100 × 100 and 200 × 200.
2. Small data sets are of sizes 400 × 400 and 500 × 500.

Very small and small instances are important to us for two reasons, firstly because for these instances we computed optimal solution/lower bound using LINGO optimization software and therefore for these instances we give the % error i.e. percentage by which the locally optimal solution

differs from the optimal solution/lower bound, thereby giving the quality of the solution produced by the algorithm. Secondly, because they are small/ very small, which means whatever we do with them, we can see results coming quickly in front of us to make important observations.
The solution obtained by the algorithm 5-approx is not dependent upon the way we construct an initial feasible solution, as far as approximation guarantee of the algorithm is concerned. We observed that certain ways of constructing an initial feasible solution turned out to be quicker and better for some particular type of problem instances. Further experiments can be done to study the effect of initial feasible solution on the final solution. These are only preliminary experiments with greater emphasis on the quality of the solutions obtained in practice then on the response time, though it was observed that response times are also reasonably good and comparable to existing heuristic algorithms.
The first step of a local search algorithm is to build an initial feasible solution. For CFLP, for a feasible solution, we need a set of facilities which are sufficient to satisfy the total demand. An assignment of clients to these facilities can be easily done by solving a min cost flow. For the selection of facilities for the initial feasible solution, we tried choosing them randomly or by picking them from the sorted order, where sorting was done based on f_i/u_i values. Tables 2 - 4 show the comparative study of the solutions obtained when a) initial feasible solution was picked randomly vs b) initial feasible solution was picked from the sorted order of the facilities.
Based on the results obtained from these experiments, for the rest of the experiments, we computed the initial feasible solution using first few facilities from the sorted order that would satisfy the total demand. This choice gives us the advantage in terms of both response time as well as solution quality for type A instances. And these are the instances which seem most time-consuming instances.

Experiments on very small instances

Very small instances are important to us because they are very small, which means whatever we do with them, we can see results coming quickly in front of us to make important observations. It is from the initial experiments on these instances that we were able to decide a uniform way of constructing an initial feasible solution for all types of instances.
Together with small instances, we have a set of 25 instances in each group of type A, B and C instances to give us a good idea about the quality of solutions obtained using 5-approx. We report these results in tables 5-7 w.r.t type A, B and C instances. For all the type A instances considered, solution obtained by 5-approx is within (1+0.08)-factor of the optimal solution. For type B instances error is a bit more, maximum is 14.49%, as compared to that with type A instances. Results obtained for type C instances are also quite encouraging, which are within (1+0.08)-factor of the optimal. Another important observation we make from these experiments is that type A instances are most time consuming. As compared to type B instances these instances take up to 10-times more time when comparing two same sized instances of these two types. Time taken by type C instances is very very small and is of the order of less than 10 milli-seconds even for 500 x 500 sized instances, which is just a small fraction of time needed to solve instances of type A and type B.
In our experiments on small data sets, we also observed that a close operation is a more time-consuming operation as compared to an open operation. We therefore tried to reduce the number of close operations over all the iterations, to reduce the running time of the procedure. We tried to apply the following criteria to consider a facility for mclose operation. If it is at most half full, then only we considered it, otherwise not. It turned out that with this type of filtering of facilities, the reduction in time of the procedure is up to 50% to 60% for type A instances, 5% to 10% for type B instances and no remarkable difference for type C instances. Reduction in response time for type A instances is always welcome for us as they are the ones which consume more time of all the lot. And to argue why this reduction happened more for type A instances it is important to observe that at any point of time an intermediate solution (not a locally optimal solution) will have a greater number of facilities which are more than half full (due to their high facility cost to service cost ratio). Therefore, with the heuristic suggested for selecting a facility for close operation, we could reduce the number of close operations performed for type A instances. The same argument answers the question why the reduction is not so much for type B instances and no reduction for type C instances.

From these experiments we can see that this local search procedure is good in practice as well i.e. although the approximation factor of this algorithm is 5, but in our experiments, we can see that cost of our solution is never greater than (1 + 0.15) times the cost of optimal solution. Considering heuristic for add operation we get a locally optimal solution in reasonable time for small instances.

5 Conclusion

From the experimental study done in this paper, we can see that local search algorithm for capacitated facility location problem gives good results in practice. In particular, 5-approx is good in practice i.e. although the approximation factor of this algorithm is 5, but in our experiments, we can see that cost of our solution is never greater than (1 + 0.15) times the cost of the optimal solution.
Further studies can be performed to try out a) different ways of selecting an initial feasible solution b) additional heuristics that can provide improved results with better response times and c) better and effective implementation.

References

[1] Bansal, M., Garg, N., Gupta, N.: A 5-approximation for capacitated facility location. In: Proceedings of 20th Annual European Symposium on Algorithms (ESA), Ljubljana, Slovenia, pp. 133–144. Springer,

(2012). https://doi.org/10.1007/ 978-3-642-33090-2 13 . http://dx.doi.org/10.1007/978-3-642-33090-2 13

[2] Bhattacharya, C., Saurabh, S., Sanyal, M., Hudnurkar, M.: Chapter 12. Ware- housing and Facility Location in E-Commerce, pp. 227–253. https://doi.org/10. 1142/9789811245992 0012 . https://www.worldscientific.com/doi/abs/10.1142/ 9789811245992 0012

[3] Bateni, M., Hajiaghayi, M.: Assignment problem in content distribution networks: Unsplittable hard-capacitated facility location. ACM Transactions on Algorithms **8**(3), 20 (2012)

[4] Kuehn, A.A., Hamburger, M.J.: A heuristic program for locating warehouses. Journal of Management Science **9**(4), 643–666 (1963)

[5] Akinc, U., Khumawala, B.M.: An efficient branch and bound algorithm for the capacitated warehouse location problem. Journal of Management Science **23**(6), 585–594 (1977)

[6] Baker, B.M.: Linear relaxations of the capacitated warehouse location problem. Journal of the Operational Research Society, 475–479 (1982)

[7] Cornuejols, G., Sridharan, R., Thizy, J.-M.: A comparison of heuristics and relaxations for the capacitated plant location problem. European Journal of Operational Research **50**(3), 280–297 (1991)

[8] Avella, P., Boccia, M., Sforza, A., Vasil´ıev, I.: An effective heuristic for large- scale capacitated facility location problems. Journal of Heuristics **15**(6), 597–615 (2009) https://doi.org/10.1007/s10732-008-9078-y

[9] Korupolu, M.R., Plaxton, C.G., Rajaraman, R.: Analysis of a local search heuristic for facility location problems. Journal of Algorithms **37**(1), 146–188 (2000) https://doi.org/10.1006/jagm.2000.1100

[10] Sinha, P.: Observations on some heuristic methods for the capacitated facility location problem. Journal of OPSEARCH **49**(1), 86–93 (2012)

[11] Alenezy, E.J.: Solving capacitated facility location problem using lagrangian decomposition and volume algorithm. Advances in Operations Research **2020**, 1–7 (2020)

[12] Barahona, F., Chudak, F.A.: Near-optimal solutions to large-scale facility location problems. Journal of Discrete Optimization **2**(1), 35–50 (2005)

Table 1 Experimental study for benchmark instances

Name	Opt	Lope(E6)	%Err
cap41	1040444.38	1141640	9.73
cap42	1098000.45	1204460	9.70
cap43	1153000.45	1265050	9.72
cap44	1235500.45	1355050	9.68
cap51	1025208.23	1047680	2.19
cap61	932615.75	951221	1.99
cap62	977799.40	1002100	2.49
cap63	1014062.05	1047100	3.26
cap64	1045650.25	1084350	3.70
cap71	932615.75	932616	0.00
cap72	977799.40	977799	0.00
cap73	1010641.45	1013930	0.33
cap74	1034976.98	1054290	1.87
cap81	838499.29	881307	5.11
cap82	910889.56	982353	7.85
cap83	975889.56	1046920	7.28
cap84	1069369.53	1131970	5.85
cap91	796648.44	817983	2.68
cap92	855733.50	882198	3.09
cap94	946051.33	976724	3.24
cap101	796648.44	802787	0.77
cap102	854704.20	861309	0.77
cap103	893782.11	899466	0.64
cap104	928941.75	961035	3.45
cap111	826124.71	839149	1.58
cap112	901377.21	908636	0.81
cap113	970567.75	987668	1.76
cap114	1063356.49	1089910	2.50
cap121	793439.56	804608	1.41
cap122	852524.63	866622	1.65
cap123	895302.33	910780	1.73
cap131	793439.56	797992	0.57
cap132	851495.33	856221	0.55
capa8	19240822.45	20013500	4.02
capa10	18438046.54	18749500	1.69
capa12	17765201.95	18599300	4.70
capa14	17160439.01	18120300	5.59
capb5	13656379.58	14406100	5.49
capb6	13361927.45	14102400	5.54
capb7	13198556.43	14049700	6.45
capb8	13082516.5	13767500	5.24
capc5	11646596.97	12242400	5.12
capc575	11570340.29	11957400	3.35
capc65	11518743.74	11788200	2.34
capc725	11505767.39	11771200	2.31

Table 2 Type A very small instances

Comparison on the basis of feasible solution			
	random	sorted	%change in quality
ndat10_50	11754.4	11260.7	4.38
ndat11_50	12697.2	11492.8	10.48
ndat12_50	12450.9	12108.6	2.83
ndat13_50	11475.3	11049.9	3.85
ndat14_50	12247.1	11370.5	7.71
ndat10_100	21859.9	21020	4.00
ndat11_100	23021.5	22048.6	4.41
ndat12_100	23355.5	23752.4	-1.67
ndat13_100	22218.6	21471.1	3.48
ndat14_100	23265.7	20936.9	11.12
ndat10_200	38520	38097	1.11
ndat11_200	42353.6	40417.5	4.79
ndat12_200	41132.6	43354.7	-5.13
ndat13_200	44911.2	39863.9	12.66
ndat14_200	41502.2	41502.2	0.00

Table 3 Type B vey small instances

Comparison on the basis of feasible solution			
	random	sorted	%change in quality
ndat20_50	23469.4	23741.7	-1.15
ndat21_50	21646.6	21646.6	0.00
ndat22_50	26904.5	26355.2	2.08
ndat23_50	24148.4	25546.6	-5.47
ndat24_50	26265.5	25612.9	2.55
ndat20_100	40239.8	39542	1.76
ndat21_100	38088.2	37775.5	0.83
ndat22_100	40405.1	39452.8	2.41
ndat23_100	37996.8	39724.8	-4.35
ndat24_100	40515.7	41790	-3.05
ndat20_200	71630.4	70097	2.19
ndat21_200	70545.6	70709.2	-0.23
ndat22_200	70930.5	69075.1	2.69
ndat23_200	62655.2	63194.8	-0.85
ndat24_200	65234.9	65234.9	0.00

Table 4 Type C very small instances

Comparison on the basis of feasible solution			
	random	sorted	%change in quality
ndat30_50	93907.9	94853.4	-1.00
ndat31_50	105099	103909	1.15
ndat32_50	113520	113990	-0.41
ndat33_50	95907.2	96613.6	-0.73
ndat34_50	100376	102246	-1.83
ndat30_100	171245	170938	0.18
ndat31_100	186467	186759	-0.16
ndat32_100	155773	155434	0.22
ndat33_100	148527	145819	1.86
ndat34_100	146903	145560	0.92
ndat30_200	273419	273942	-0.19
ndat31_200	233455	233589	-0.06
ndat32_200	236750	237987	-0.52
ndat33_200	229990	228224	0.77
ndat34_200	232098	233099	-0.43

Table 5 Experimental study for small instances of type A

5-approx(with all operations)			
	OPT	LOPT	%error
ndat10_400	72550.7	75115	0.04
ndat11_400	77213.3	81921.5	0.06
ndat12_400	76096.9	79464.5	0.04
ndat13_400	73771.6	78455.1	0.06
ndat14_400	74186.6	76537.9	0.03
ndat10_500	93634.4	96764.9	0.03
ndat11_500	93436.7	96349.7	0.03
ndat12_500	92652	95336.8	0.03
ndat13_500	93466.4	97976.7	0.05
ndat14_500	93763.5	98437	0.05

Table 6 Experimental study for small instances of type B

	OPT	LOPT	%error
ndat20_400	113557	126168	0.11
ndat21_400	111682	123627	0.11
ndat22_400	113991	124994	0.10
ndat23_400	112429	126174	0.12
ndat24_400	113587	126194	0.11
ndat20_500	139620	154165	0.10
ndat21_500	135965	149348	0.10
ndat22_500	135371	149066	0.10
ndat23_500	138221	155477	0.12
ndat24_500	133515	149595	0.12

Table 7 Experimental study for small instances of type C

5-approx(with all operations)			
	OPT	LOPT	%error
ndat30_400	368469	395064	0.07
ndat31_400	369409	398159	0.08
ndat32_400	366481	387862	0.06
ndat33_400	381740	407525	0.07
ndat34_400	355443	377673	0.06
ndat30_500	421953	453102	0.07
ndat31_500	405522	438945	0.08
ndat32_500	420826	449195	0.07
ndat33_500	433168	456736	0.05
ndat34_500	430161	454706	0.06