ACTA TECHNICA CORVINIENSIS – Bulletin of Engineering
Tome IV (Year 2011). ISSN 2067-3809

[1.] Yasel José COSTA SALAS, [2.] Norge COELLO MACHADO,
[3.] Elke GLISTAU, [4.] Carlos MACHADO OSES

A NEW ALGORITHM FOR FACILITY LOCATION PROBLEM BASED ON DYNAMIC MESH OPTIMIZATION

ABSTRACT:
This paper proposes Dynamic Mesh Optimization for the classical Facility Location Problem, we introduce this meta-heuristic which is a technique of evolutionary computation. A set of nodes that represent potential location solutions conform a mesh; it grows and moves dynamically throughout the search space. The algorithm performance has been compared with data set from literature. Computational results confirm the efficiency of the strategy we propose.

KEYWORDS:
Algoritm, location problem, Dynamic Mesh Optimization

INTRODUCTION

One of the most important decisions in the logistical planning is to establish where the locations have to be (whether factories, warehouse, markets, etc). The Facility Location Problem (FLP) has been widely studied by different authors, often specialists from Operation Research and Logistic areas. This kind of problem is a well-known NP-Hard combinatorial optimization problem which is encountered frequently in decision making process, beside in logistics system.

In FLP there is a set of locations at which we may build a facility (such as a warehouse), where the cost of building dependents of each location; furthermore, there is a set of client locations (such as stores, markets) that require to be serviced by a facility, and if a client at location j is assigned to a facility at location i, a cost of c_{ij} is incurred that is proportional to the distance between i and j. The objective is to determine a set of locations at which to open facilities so as to minimize the total facility and assignment costs.

An abundant literature on facility location problem is available. Beside, there are several type of them, such as uncapacitated facility location problem introduced by [4], [1] and capacitated facility location problem (CFLP) reported in [3] and [5]. In this paper we focus in the CFLP.

Moreover, various researches have shown the effective use of meta-heuristic in CFLP [10], [6]. This paper proposes to examine the capacitated facility location problem based on DMO, which is classifying as evolutionary computation techniques. Multiple types of nodes are generated in order to conform a mesh, which dynamically expands itself and moves across the search space. This meta-heuristic was created by [7], however all work deals with the optimization process in continuous approach; we modify the algorithm for optimization process in discrete context, such as CFLP.

The paper is structured as follows: In Section 2 is formulated the capacitated facility location problem, description of meta-heuristic and the algorithm steps are defined at Section 3. Computational results and the algorithm performance can be found in Section 4. Conclusions and future researches are outlined in Section 5.

PROBLEM DESCRIPTION

The CFLP is define on a graph G (V, E) where |V| = n, vertices (customer to meet) and "E" indicates the Euclidian distance by which the vertices are connected "V". The decision variable can be described as Xij = (0, 1): where (0) that vertex "j" is not assigned to the facility "i" and (1) otherwise. There is a set M(i) which represents the number of arcs that affect the vertex "i". In addition to each arcs poses a d(i,j) representing the minimum distances between "i" and "j". It is expressed therefore an integer value m_i, which represents nodes, allocated to an installation "i", where i = (1...k). For the capacitated facility location problem are established usually the following constrains:

- $\sum_{i=1}^{k} m_i = n$: *All customers have to be allocated.*
- $\forall i : \sum_{j=1}^{n} X_{ij} = m_i$: *"k" facilities have to cover all customers.*
- $\forall j : \sum_{i=1}^{k} X_{ij} = 1$: *Each customer has just one allocated facility.*

Nodes Generation in DMO

The dynamic nature of our proposal manifests in the generation of (I) the initial mesh; (II) intermediate nodes oriented toward the local optima; (III) intermediate nodes in the direction of the global optimum and (IV) nodes aiming at expanding the dimensions of the current mesh.

The model gives rise to the following parameters: (I) Ni → size of the initial mesh, (II) N → maximum size of the mesh across each cycle (Ni < N) and (III) M → number of cycles.

During the mesh expansion in each cycle, a weight w is defined using expression (1) as in [8], [9].

$$w = (w_0 - 1.4) \times \frac{M - j}{M + 0.4} \qquad (1)$$

DYNAMIC MESH OPTIMIZATION AS A META-HEURISTIC

The main idea of the DMO method is the creation and representation of a mesh in points according to the N-dimensional space wherein the optimization of $f(x_1, x_2, \dots, x_n)$ is performed.

The mesh endures an expansion process toward the most promising regions of the search space but, at the same time, becomes finer in those areas where there exists points that constitute local ends of the function. The dynamic nature of the mesh is given by the fact that its size (number of nodes) and configuration both change over time. When it comes to the feature selection problem, nodes can be depicted as binary vectors $n(x_1, x_2, \dots, x_n)$ of N components, one per attribute, with the component n_i = 1 if the i-th attribute is being considered as part of the solution or zero otherwise. In each cycle, the mesh is created with an initial number of nodes. Subsequently, new nodes are generated until an upper boundary in the number of nodes is reached. The mesh at the next cycle is comprised of the fittest nodes of the mesh in the former iteration. Along the search process, the node carrying the best value of the objective (evaluation) function so far is recorded, so n_g denotes the global end attained up to now by the search algorithm.

In the case of the facility location problem, the quality and evaluation function at the same time is displayed by expression (2), which is formulated in the classical FLP.

$$Eval(n) = \sum_{i}^{k} \sum_{j}^{n} d_{ij} \cdot X_{ij} \qquad (2)$$

THE DMO-FLP ALGORITHM

STEP 1. Generate the initial mesh for each cycle: At the beginning of the algorithm's execution, the initial mesh (binary values represented by decision variable) will be made up of Ni randomly generated nodes while in the remaining iterations, the initial mesh is built upon the selection of the best (in terms of evaluation measure) Ni nodes of the mesh in the preceding cycle.

STEP 2. Node generation toward local optima: The aim of this step is to come up with new nodes settled in the direction of the local optima found by the algorithm. For each node n, its K-nearest neighbor nodes are computed (the Hamming distance is a suitable option for the FLP). If none of the neighbors surpasses n in fitness function value, then n is said to be a local optimum and no nodes are begotten out of it in this step. Conversely, suppose that node n_e is "better" than n and the rest of its neighbors. In this case, a new node arises somewhere between n and n_e.

The proximity of the newly generated node n^* to the current node n or to the local optimum n_e is contingent upon a factor r which is calculated based on the evaluation function values both at nodes n and n_e. Each component of n takes either the value of n_i or ne_i according to a rule involving a stochastic value. The threshold r determining how every component n_i^* is set is calculated by expression (3).*

$$r = 1 - 0.5 \frac{Eval(ne)}{Eval(n)} \qquad (3)$$

f(n, ne, r) : For each component n_i: If Random() < r then $n_i^* = ne_i$ otherwise $n_i^* = n_i$

Notice from (4) that the lower the ratio between Eval(n) and Eval(ne), the more likely it is that ni takes the value of the i-th component of the local optimum.*

STEP 3. Node generation toward global optimum: Here the idea is the same as in the previous step but now r is computed differently and a function g is introduced. Needless to say that n_g represents the global optimum found thus far by the algorithm in each cycle, see expression (4).

$$r = 1 - 0.5 \frac{Eval(ng)}{Eval(n)} \qquad (4)$$

g(n, ng, r) : For each component n_i: If Random() < r then $n_i^* = ng_i$ otherwise $n_i^* = n_i$

STEP 4. Mesh expansion: In this step, the mesh is stretched from its outer nodes using function h, i.e. using nodes located at the boundary of the initial mesh in each cycle. The weight w depicted in (1) assures that the expansion declines all over the search process (i.e., a bigger expansion is achieved at the early cycles and it fades out as the algorithm progresses). To determine which nodes lie in the outskirts of the mesh, those having the lowest and greatest norm are picked. Remark that, in this step, as many outer nodes as needed are selected so as to fill out the maximum mesh size N. The rules governing this sort of node generation can be found next:

For each node nl in the lower boundary (those with lower norm):
h(nl, w) : For each component n_i: If Random() < w then $n_i^* = 0$ otherwise $n_i^* = nl_i$
For each node nu in the upper boundary (those with greater norm):
h(nu, w) : For each component n_i: If Random() < w then $n_i^* = 1$ otherwise $n_i^* = nu_i$
In the context of facility location, the norm of a node (vector) is the number of its components set to 1. Finally, Algorithm 1 outlines the workflow of the DMO approach. It is also worth remarking that no direct search algorithm guarantees to find the global optimum no matter how refined the heuristic search might be.

Algorithm 1 The DMO meta-heuristic

```
Randomly generate Ni nodes to build the initial mesh
Evaluate all the mesh nodes
Repeat
   for each node n in the mesh do
      Find its K-nearest neighbors
      n_best ← the best of its neighbors
      if n_best is better than n then
         Generate a new node by using function f
      end if
   end for
   for each initial node in the current mesh do
      Generate a new node by using function g
   end for
   Repeat
      Select the most outward node of the mesh
      Generate a new node by using function h
   until MeshSize = N
   Select the best Ni nodes of the current mesh and
set up the next mesh
until CurrentIteration = M
```

COMPUTATIONAL RESULTS

In this section some computational results are presented in order to evaluate the performance of the algorithm described in Section 3. Algorithm runs have been carried out on a personal computer equipped with a Intel Pentium dual-core processor 1.6 GHz and 1 GB of ram memory. The FLP-DMO was coded Java 1.5.0.
The configuration of the DMO-FLP has been defined as follows: a mesh with 40 nodes is used, 15 of them regarded as initial nodes, therefore is necessary to generate 25 nodes per cycle according with Algorithm 1. Finally 100 iterations were executed.

Table 1: Numerical results for DMO compared to GA.

	DMO		GA	
Instances	bestf	Avg.	Bestf	Avg.
6Cap10	2882.2	2974.6	2796.1	2984.3
10Cap10	3029.3	3102.7	2998.3	3042.2
12Cap20	2225.1	2386.5	2227.3	2324.6
16Cap30	2032.1	2234.4	2002.7	2103.4
20Cap40	1824.0	1975.8	1796.1	1854.8

The algorithm is tested into five different problems from literature. These problems are derived form a benchmark datasets found in [2].

In Table 1, the best found solution (bestf) and run time (RT) are reported for two algorithms: DMO and Genetic Algorithm (GA). For Generic algorithm were fixed a group of parameters figured in Table 2.

Table 2: GA parameters

Parameters	Value
Population Size	300
Crossover rate	0.90
Mutation rate	0.10
Number of Runs	20

Starting from figures of Table 1 we obtained non significant differences between these algorithms, due to results of Wilconxon coefficient as statistic test.

CONCLUSIONS & FUTURE RESEARCH

In this paper, Dynamic Mesh Optimization is presented as evolutionary algorithm. Moreover, the performance of DMO in Facility Location Problem outcomes relevant for datasets found in literature. According with numerical and statistic test, we can conclude that DMO can solve CFLP in similar way than Genetic Algorithm.
Future researches would focus to combine Dynamic Mesh Optimization with Local Search strategy in order to improve the solution quality. Beside we have to consider sensitivity analysis of the fixed parameters in the algorithm.

REFERENCES

[1.] Kuehn and M. J. Hamburger. (1963). A heuristic program for locating warehouses. Management Sci., 9:643–666.
[2.] FLP-LIB. Benchmarks for CFLP. Available in http://www.cflp-bmks06.ac.uk/
[3.] M. Korupolu, C. Plaxton, and R. Rajaraman. (2000). Analysis of a local search heuristic for facility location problems. Journal of Algorithms 37:146-188.
[4.] M. L. Balinksi. (1966). On finding integer solutions to linear programs. In Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems, pages 225–248.
[5.] M. Mahdian, Y. Ye, and J. Zhang. (2002). Improved approximation algorithms for metric facility location problems. In Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization, pages 229-242.
[6.] Mladenovic, N., Brimberg, J., Hansen, P., Moreno, J. (2007). The p-median problem: A survey of meta-heuristic approaches. European Journal of Operational Research, 3, 927-939.
[7.] Puris, A. (2009). Design and analysis of populational meta-heuristic. PhD thesis. Central University of “Las Villas”. Santa Clara.
[8.] Wang, X., Yang, J., Jensen, R., Liu, X. (2006). Rough Set Feature Selection and Rule Induction for Prediction of Malignancy Degree in Brain Glioma. Computer methods and Programs in Biomedicine 83, 147-156.

[9.] Wang, X., Yang, J., Teng, X., Xia, W., Jensen, R. (2007). Feature Selection Based on Rough Sets and Particle Swarm Optimization. Pattern Recognition Letters 28, 459-471.

[10.] Wu, L., Zhang, X., Zhang, J. (2006). Capacitated facility location problem with general setup cost. Computer & Operations Research, 33, 1226-1241.

AUTHORS & AFFILIATION

[1.] Yasel José COSTA SALAS
[2.] Norge COELLO MACHADO
[3.] Elke GLISTAU
[4.] Carlos MACHADO OSES

[1, 2, 4] FAKULTÄT VON INDUSTRIAL INGENIEUR UND TOURISMUS. UCLV. CUBA
[3] FAKULTÄT VON MASCHINENBAU. ILM. UNIVERSITÄT MAGDEBURG, DEUTSCHLAND

ACTA TECHNICA CORVINIENSIS – BULLETIN of ENGINEERING
ISSN: 2067-3809 [CD-Rom, online]