PERFORMANCE COMPARISON OF CBO AND ECBO FOR LOCATION FINDING PROBLEMS

A. Kaveh and M. Ghobadi
Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran 16, Iran

ABSTRACT

The p-median problem is one of the discrete optimization problem in location theory which aims to satisfy total demand with minimum cost. A high-level algorithmic approach can be specialized to solve optimization problem. In recent years, meta-heuristic methods have been applied to support the solution of Combinatorial Optimization Problems (COP). Collision Bodies Optimization algorithm (CBO) and Enhanced Colliding Bodies Optimization (ECBO) are two recently developed continuous optimization algorithms which have been applied to some structural optimization problems. The main goal of this paper is to provide a useful comparison between capabilities of these two algorithms in solving p-median problems. Comparison of the obtained results shows the validity and robustness of these two new meta-heuristic algorithms for p-median problem.

Keywords: p median problem, CBO, ECBO.

Received: 21 November 2015; Accepted: 12 January 2016

1. INTRODUCTION

Most of the public and private companies have the problem of finding appropriate locations for their facilities. Government agencies need to determine locations of offices and other public services such as schools, hospitals, fire stations, ambulance bases, military bases, radar installations, waste disposal facilities and so on. Industrial companies must locate the fabrication and assembly plants as well as warehouses. In these cases, the success or failure of facilities depends in part on the locations chosen for these facilities [1]. Such problems are known as location-allocation problems.

Most of the studies on location finding problems are classified into four categories:

1. Corresponding author: Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran 16, Iran
2. E-mail address: alikaveh@iust.ac.ir (A. Kaveh)

In this paper a comparative study is performed for two meta-heuristics consisting of CBO and ECBO algorithms for solving p-median problem.

2. THE P-MEDIAN PROBLEM

The p-median problem is one of a large class of location problems in both capacitated and incapacitated conditions. The aim of p-median problem is to locate p facilities among n demand points and allocates the demand points to the facilities. The objective is to minimize the total demand-weighted distance between the demand points and the facilities. In the early 20th century, Alfred Weber presented the same problem with the addition of weights on each of the three points to simulate customer demand. The following formulation of the p-median problem is due to ReVelle and Swain [3].

\[
F = \min \sum_i^n \sum_j^n w_i d_{ij} x_{ij} \\
\text{s.t.} \sum_j^n x_{ij} = 1 \quad \forall j \\
- x_{ij} \leq y_j \quad \forall i, j \\
\sum_i^n y_j = p
\]

where
- \(n\) = total number of demand points,
- \(x_{ij}\) = 1 if a facility is located at point \(i\), 0 otherwise
- \(y_j\) = 1 if a facility is located at point \(j\), 0 otherwise
- \(w_i\) = demand at point \(i\)
- \(d_{ij}\) = travel distance between points \(i\) and \(j\)
- \(p\) = number of facilities to be located

Hakimi [4] developed a method for finding medians on a network or graph and showed that the absolute median of a graph \(G\) is always located at a vertex of a graph. Thus, to find the optimum location for a switching center in a communication network, one must only search the vertices of the graph of a network and he proved that the p-median problem is an NP-hard problem [5].

Mathematical and exact methods were the earliest techniques proposed for solving allocation problems. However, as the problem size increases, the computational time of exact methods increases exponentially. In contrast to exact methods, heuristic algorithms generally have acceptable time and memory requirements, but do not guarantee optimal solution [6]. Over the past three decades, there has been a considerable increase in the amount of solution methods. Meta-heuristics provide a general framework to build heuristics for combinatorial and global optimization problems. Meta-heuristics have many distinctive features that make them as suitable techniques, especially when these are combined with other optimization methods [7].

These have been the subject of intensive research since Kirkpatrick, Gellatt and Vecchi [8] proposed Simulated Annealing (SA) as a general scheme for building heuristics, capable of escaping the local optimums. In 1996 the basic SA heuristic for p-median problem (PMP) has been proposed by Murray and Church. After that several Tabu Search and Genetic search methods have been proposed for solving PMP [9, 10, 11].
In 1992, Dorigo developed the ant colony optimization (ACO), Levanova and Loresh [12] applied the hybrid of ACO and SA. In 2000, Glover applied the Scatter search for PMP which this algorithm was proposed by himself in 1977. Charged System Search (CSS) was used by Kaveh and Sharafi in 2008 [13].


3. OPTIMIZATION ALGORITHM

Evolutionary computation uses iterative process, such as growth or development in a population. This population is then selected in a guided random search using parallel processing to achieve the desired end. A local search procedure looks for the best solution near another solution by repeatedly making small changes to the current solution. This procedure is continued until no further solution can be found. The convergence properties of meta-heuristics are closely related to the random sequence applied on their operators during a run. In particular, when starting some optimizations with different random numbers, experience shows that the results may be very close but not equal, and require also different numbers of generations to reach the same optimal value [7].

A meta-heuristic can be successful on a given optimization problem if a balance between the exploration (diversification) and the exploitation (intensification) can be achieved. Exploitation is needed to identify parts of the search space with high quality solutions. Exploitation is important to intensify the search in some promising areas of the accumulated search experience. The main differences between the existing meta-heuristics concern the particular way in which they try to achieve this balance [18]. Also defining a neighborhood structure is the most important aspect of algorithms. By using an efficient neighborhood structure, a problem can be solved with higher accuracy in less computational time.

CBO and ECBO are two recently developed meta-heuristic algorithms developed by Kaveh & Mahdavi [19] and Kaveh & Ilchi Ghazaan [20], respectively.

3.1 CBO algorithm

The basic idea of the theory of colliding bodies optimization (CBO) is that the total momentum before the collision to be the same as the total momentum after the collision [19]. CBO is a simple algorithm and it depends on no internal parameter. In this algorithm, each body is a candidate solution to the problem which is characterized by a mass and velocity. In the initialization phase of the CBO, the positions and velocities of all individuals are randomly initialized (Eq. (1) & Eq. (2)). In the second step, objective function is evaluated and masses are defined (Eq. (3)). At each iteration, a particle CB, adjusts its position $X_i$ and velocity $V_i$ according to the previous position and the velocity after the collision (Eq. (4) & Eq. (6)).

The optimization is repeated until termination criteria, specified as the maximum number of iterations, is satisfied.
A. Kaveh and M. Ghobadi

\[ x_i^0 = x_{\text{min}} + \text{rand}(x_{\text{max}} - x_{\text{min}}), \quad i = 1, 2, \ldots, n, \]  
(1)

\[ v_i = x_{\text{i} - \frac{n}{2}} - x_i, \quad i = \frac{n}{2} + 1, \ldots, n \]  
(2)

\[ m_k = \frac{1}{\text{fit}(k)}, \quad k = 1, 2, \ldots, n \]  
(3)

\[ v_i^k = \frac{m_{\text{i} - \frac{n}{2}}^k}{m_{\text{i} - \frac{n}{2}} + m_i^k}, \quad i = \frac{n}{2} + 1, \ldots, n \]  
(4)

\[ \varepsilon = 1 - \frac{\text{iter}}{\text{iter}_{\text{max}}} \]  
(5)

\[ x_i^{\text{new}} = x_{\text{i} - \frac{n}{2}} + \text{rand} \circ v_i, \quad i = \frac{n}{2} + 1, \ldots, n \]  
(6)

### 3.2. ECBO algorithm

In order to improve the CBO to get faster and more reliable solutions, Enhanced Colliding Bodies Optimization (ECBO) was developed by Kaveh and Ichi Ghazaan [20]. ECBO uses memory to save a number of historically best CBs and also utilizes a mechanism to escape from local optima. The solution vectors saved in CM are added to the population, and the same numbers of current worst CBs are deleted. Then the values of \( X_i \) and \( V_i \) are evaluated before and after the collision. Finally, CBs are sorted according to their masses in a decreasing order. In ECBO, a parameter like \( \text{Pro} \) within \((0, 1)\) is introduced which specifies whether a component of each CB must be changed or not. For each colliding body \( \text{Pro} \) is compared with \( \text{rand} \, i \) \((i = 1, 2, \ldots, n)\) which is a random number uniformly distributed within \((0, 1)\). If \( \text{rand} \, i < \text{Pro} \), one dimension of the \( i \)th CB is selected randomly and its value is regenerated and the termination condition is checked [21]. CBO and ECBO and some recently developed metaheuristic algorithms can be found in Kaveh [22]. Pseudo code of teh ECBO is illustrated in Fig. 1.

---

**Pseudo code of Enhanced Colliding Bodies Optimization**

- Initial location is created randomly
- The value of objective function and mass are evaluated
- While stop criteria is not attained (like max iteration)
  - for each CBs
    - Calculate CBs velocity before collision according equation (2)
    - Calculate CBs velocity after collision according equation (4)
    - Update CBs position according equation (6)
  - If \( \text{rand} \, i < \text{Pro} \)
    - One dimension of the \( i \)th CB is selected randomly and regenerate
  - End if
  - End for
- End while
- End

---

Figure 1. Pseudo code of enhanced colliding bodies optimization [21]
4. NUMERICAL EXAMPLE

In this section, three examples are studied. The first example is a quadrangular FE mesh with 1936 nodes (44×44). The second and third examples are rectangular and H-shaped forms of Kaveh and Mahdavi [16]. A weighted incidence graph is used to transform the connectivity properties of finite element models into those of graphs. In all of these examples the weights of all the nodes and edges are taken as unity. In the second stage after establishing adjacency matrix of the graph, the graph is partitioned into p subdomains by use of the p median concept and metaheuristic algorithms. Three optimization algorithms, CBO, ECBO and PSO are applied for decomposing the meshes by p medians and the results are compared in Table 1.

**Example 1:** A quadrangular FE mesh with 44×44=1936 nodes and its decomposition into 5-10-20 subdomains are shown in Fig. 2. The comparisons of convergence rates for the three algorithms for each instance are made in Table 1 and Fig. 3.

**Example 2:** A rectangular FE mesh with 760 nodes and four internal perforations is shown Fig. 4. The comparisons of convergence rates for the three algorithms for each instance are made in Table 1 and Fig. 5.

**Example 3:** A H-shaped FE mesh is considered as shown in Fig. 6, and decomposed into 5–10 subdomains with medians. The comparisons of convergence rates for the three algorithms for each instance are made in Table 1 and Fig. 7.

![Figure 2. A quadrangular mesh divided into 5 and 10 subdomains by the ECBO algorithm](image-url)
Table 1: The minimum cost and CPU time for the considered numerical examples

<table>
<thead>
<tr>
<th>Number of medians</th>
<th>CBO</th>
<th></th>
<th>ECBO</th>
<th></th>
<th>PSO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min cost</td>
<td>CPU time(s)</td>
<td>min cost</td>
<td>CPU time(s)</td>
<td>min cost</td>
<td>CPU time(s)</td>
</tr>
<tr>
<td>Example 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=5</td>
<td>19048</td>
<td>0.86639</td>
<td>18912</td>
<td>0.79748</td>
<td>19423</td>
<td>0.98038</td>
</tr>
<tr>
<td>p=10</td>
<td>13344</td>
<td>1.09393</td>
<td>13216</td>
<td>1.230409</td>
<td>13872</td>
<td>1.6339</td>
</tr>
<tr>
<td>p=20</td>
<td>9368</td>
<td>2.9727</td>
<td>9232</td>
<td>3.1651</td>
<td>9448</td>
<td>6.2567</td>
</tr>
<tr>
<td>Example 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=5</td>
<td>3711</td>
<td>0.41211</td>
<td>3623</td>
<td>0.53425</td>
<td>3960</td>
<td>3.8188</td>
</tr>
<tr>
<td>p=10</td>
<td>2680</td>
<td>1.26562</td>
<td>2575</td>
<td>1.38088</td>
<td>2744</td>
<td>4.498625</td>
</tr>
<tr>
<td>Example 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=5</td>
<td>4374</td>
<td>1.201909</td>
<td>4114</td>
<td>2.187340</td>
<td>4224</td>
<td>2.568828</td>
</tr>
<tr>
<td>p=10</td>
<td>3002</td>
<td>2.691113</td>
<td>2930</td>
<td>2.758156</td>
<td>3076</td>
<td>2.275321</td>
</tr>
</tbody>
</table>

Figure 3. The history of convergence for \( p = 5 \) and \( p = 10 \) in a quadrangular mesh

Figure 4. A rectangular mesh divided into \( p \) subdomains by the ECBO algorithm
Figure 5. The history of the convergence for $p = 10$ in a rectangular mesh with four perforations.

Figure 6. The H-shaped mesh divided into $p$ subdomains by the ECBO.

Figure 7. The history of convergence for $p = 5$ and $p = 10$ for the H-shaped mesh.
5. DISCUSSION AND CONCLUSIONS

In this study the CBO and ECBO algorithms are applied to finding medians of different finite element models. Though the CBO has been successfully implemented in partitioning and the speed of convergence is better than ECBO (Fig. 8), however, the reliability of the ECBO in min cost and solution accuracy is preferable (see Fig. 3, Fig. 5 and Fig. 7). By ECBO, more runs have a similar answers and this shows a better convergence around near optimal solution within a reasonable time. This confirms that the memory of ECBO can help the CBO to escape from local minima.

![Comparison of times for partitioning](image1)

![Comparison of times for partitioning](image2)

Figure 8. Comparison of the computational time for the CBO and ECBO

REFERENCES