Non-linear Integer Programming Model and Algorithms for Connected p-facility Location Problem

Jianming ZHU
College of Engineering and Information Technology, University of Chinese Academy of Sciences, Beijing 100049, China
E-mail: jmzhu@ucas.ac.cn

Abstract In this paper, a new location analysis method is presented. Given a connected graph $G = (V, E)$ with nonnegative edge cost $c_e$ for each edge $e \in E$, $d_{ij}$ is the cost of the shortest path between vertices $i$ and $j$ in the graph. The Connected p-facility Location Problem (CpLP) is to choose $p$ vertices from $V$ so as to minimize the total cost of shortest path of pair-wise of these $p$ vertices. This problem is proved to be NP-hard and non-linear integer programming is formulated. Then, two algorithms are designed for solving the CpLP. One is a heuristic algorithm based on classical maximum clique approach, while the second one is genetic algorithm. Finally, computational results show the comparison between these two algorithms.

Keywords connected location; maximum clique; heuristic algorithm; genetic algorithm

1 Introduction

Given a connected graph $G = (V, E)$ with nonnegative edge cost $c_e$ for each edge $e \in E$, $d_{ij}$ is the cost of the shortest path between vertices $i$ and $j$ in the graph. The Connected p-facility Location Problem (CpLP) is to choose $p$ vertices from $V$ such that the total shortest path cost of each pair of these $p$ vertices is minimized.

Facility Location Problem (FL) is a classical issue which has been studied since Weber Location Problem in 1909. The problem looks for the best locations for a set of facilities that must satisfy requests of service coming from a given set of customers. We use the term “facility” here in its broadest sense. That is, it is meant to include entities such as factories, warehouse, retail outlets, hospitals, electronic switching centers and emergency warning sirens. The ubiquity of location decision-making has led to a strong interest in location analysis and modeling within the operations research and management science community. There are many facility location models, such as covering[1, 2], p-center[3, 4], p-dispersion[5] and p-median[3].

Cooper[6] formulated the Facility Location-Allocation (FLA) that provides a valuable method in deciding where to place facilities coupled with determining how to assign demand to the located facilities in order to utilize resources effectively. Another interesting location analysis method is to combine the vehicle routing problem with facility location, which is formulated to Location-Routing Problem (LRP). Many variations of LRP exist, such as deterministic LRP[7], dynamic LRP[8] and planar LRP[9]. Much more work about LRP can be obtained from[10].

Received December 30, 2013, accepted February 27, 2014
Supported by the National Natural Science Foundation of China (Grant No. 91324012, 71001099, 91024031)
To optimize transportation between facilities, Gupta\cite{11} introduce \textit{Connected Facility Location Problem} (ConFL) which corresponds to the above mentioned facilities. The facilities need to be installed and connected with each other and customer nodes need to be assigned to them. The ConFL problem consists of finding an assignment of each customer to exactly one facility and connecting these facilities via a Steiner tree. Furthermore, the ConFL is transformed into the minimum Steiner arborescence problem and solved by an exact branch-and-cut method. Ten different integer programming formulations for ConFL have been presented by \cite{12}. Several relevant special ConFLs are considered in some recent works, such as capacity constrained ConFL\cite{13,14,15}, hop constrained ConFL\cite{16,17}.

In this paper, a new location analysis method is presented by paying more attention to the sum distance/cost of pair-wise of these selected facilities. This problem comes from the real situation when dealing with natural disasters. In order to relieve the damage of large-scale disasters, many countries pay attention to emergency management issues. Mitigation and preparation are carefully considered in the one hand, while quick and efficient response is done on the other hand. Relief materials are very important in the whole process of disaster response. Huge demands are supplied from all over the country, sometimes outside the country from international support. Distribution center plays an important role in reserving, distributing, and transporting relief goods in order to enhance the ability of material organization. Huge demand for materials, especially for large-scale disasters, need to be transported from all other centers. The objective is to minimize the total transportation cost or time between each pair of the selected distribution centers.

The remainder of this paper is organized as follows. In Section 2, we will formulate a mathematic model for CpLP in detail and present the NP-hardness proof for this problem. Two efficient algorithms are designed for solving this problem in Section 3. The first one is based on procedure for maximum clique problem, while the other one is genetic algorithm. Computational results are shown in Section 4. In Section 5, we draw conclusions with further researches discussion.

2 Model formulation and complexity analysis

In this section, the mathematical model for CpLP will be formulated. And complexity of this problem will be analyzed.

2.1 Basic model

Let $G = (V, E)$ be a connected graph with nonnegative edge cost $c_e$ for each edge $e \in E$. Parameters and decision variables are defined below.

\textit{Parameters}

$p$ : the number of vertices need to be selected.

$d_{ij}$ : the cost of the shortest path between vertices $i$ and $j$ in the graph $G = (V, E)$.

\textit{Decision variables}

$y_i = \begin{cases} 
1, & \text{vertex } i \text{ is selected,} \\
0, & \text{otherwise,} 
\end{cases}$
Now the non-linear integer programming formulation is presented below.

\[
\min z = \frac{1}{2} \sum_{i \in V} \sum_{j \in V} d_{ij} y_i y_j \tag{1}
\]

subject to

\[
\sum_{i \in V} y_i = p \tag{2}
\]

\[y_i = \{0, 1\}, \quad \forall i \in V \tag{3}\]

2.2 NP-hardness proof

By reduction from maximum clique problem, we will prove the problem is NP-hard. Given an undirected graph \(G\), a clique is a subgraph of \(G\) in which all the pairs of vertices are adjacent. Two vertices are said to be adjacent if they are connected by an edge. The maximum clique problem, which is one of the most important NP-hard problems in discrete mathematics and theoretical computer science, is to find a clique with maximum number of vertices.

The above definition of maximum clique problem is called the optimization version. In order to perform the proof, let’s introduce the recognition version of the maximum clique problem.

Given an undirected graph \(G\) and integer \(p\). Whether \(G\) has a clique which has at least \(p\) vertices?

Then, we define recognition version of Connected p-facility Location Problem as follows.

Given a connected graph \(G = (V, E)\) with nonnegative edge cost \(c_e\) for each edge \(e \in E\), \(d_{ij}\) is the cost of the shortest path between vertices \(i\) and \(j\) in the graph. Also, given an integer \(p\) and nonnegative number \(w\). Whether \(G\) has a subset with \(p\) vertices whose sum cost of each pair of these \(p\) vertices is at most \(w\)?

Since the recognition version of a combinatorial problem has the same complexity as optimization version, we only need to prove the recognition version of CpLP to be NP-hard.

**Theorem 1** The recognition version of CpLP is NP-hard even when all edge with unit cost.

**Proof** For any instance of recognition version of maximum clique problem, we will construct an instance of CpLP.

For an instance \(I_{\text{MCP}}\) of maximum clique problem with undirected graph \(G = (V, E)\) and integer \(p\), let’s construct an instance \(I_{\text{CpLP}}\). Define a weighted graph \(G = (V, E)\) with \(c_e = 1\) for any \(e \in E\), integer \(p\) and \(w = \frac{p(p-1)}{2}\). We will show \(I_{\text{CpLP}}\) has a subset with \(p\) vertices whose sum shortest path cost of each pair of these \(p\) vertices is at most \(\frac{p(p-1)}{2}\) if and only if \(I_{\text{MCP}}\) has a clique which has at least \(p\) vertices. At first, if \(I_{\text{MCP}}\) has a clique \(C\) which has at least \(p\) vertices, the cost of shortest path between each pair of vertices in \(C\) equals 1, then the sum cost equals \(\frac{p(p-1)}{2}\). On the other hand, if \(I_{\text{CpLP}}\) has a subset with \(p\) vertices whose sum shortest path cost of each pair of these \(p\) vertices is at most \(\frac{p(p-1)}{2}\), any pair of vertices in this subset is adjacent in \(G\), then this subset is a clique of \(G\) with \(p\) vertices. This is the end of the proof.

Since the CpLP is NP-hard, it is hard to solve this problem exactly especially for large-scale
instances. A lower bound for CpLP is presented as follows for comparison of our algorithms.

\[ \text{lowerbound} = \text{the sum } d_{ij} \text{ of minimum } \frac{p(p-1)}{2} \text{pairs of vertices.} \]

3 Algorithms

In this section, two algorithms will be presented. The first one is a heuristic algorithm based on classical maximum clique approach, while the second one is genetic algorithm.

3.1 Algorithm based on maximum clique procedure

Since the maximum clique problem is also NP-hard, we design a heuristic polynomial algorithm for the maximum clique problem at the beginning. Given an undirected graph \( G = (V, E) \), let \( G(U) \) be the induced graph for any subset \( U \subseteq V \). \( d_{G(U)}(v) \) is the degree of vertex \( v \in U \) in graph \( G(U) \) and \( N_{G(U)}(v) \) is the neighbor of vertex \( v \in U \) in graph \( G(U) \). Now, we present the maximum clique procedure (MCP).

**Procedure MCP** \((G = (V, E))\)

**Input** a graph \( G = (V, E) \)

**Output** a clique \( C \) and its size \( NumC \)

1. \( C = \emptyset, U = V \);
2. **while** \( U \neq \emptyset \)
3. \( u = \max\{v|d_{G(U)}(v), v \in U\} \)
4. \( C = C \cup \{u\} \)
5. \( U = N_{G(U)}(u) \setminus \{u\} \)
6. **endwhile**
7. \( NumC = \text{size}(C) \)
8. return \( C \) and \( NumC \)

There are much more research on maximum clique problem. One standard approach is based on the branch-and-bound method, such as [19–22]. Our procedure MCP is a simple greedy method which can find a clique within a very short time. Next, we will present our algorithm for solving the CpLP based on procedure MCP. The algorithm begins with an empty graph, then executes the procedure MCP while adding the edges.

**Algorithm based on maximum clique procedure**

**Input** a connected graph \( G = (V, E) \) with nonnegative edge cost \( c_e \) for each edge \( e \in E \), and an integer \( p \)

**Output** a subset \( C \) of \( V \) with \( p \) vertices

1. Calculate the shortest path \( d_{ij} \) for every pair of vertices \( i \) and \( j \) using Floyd-Warshall algorithm;
2. \( E' = \emptyset, \) NumC = 0;
3. \( \text{visited}(ij) = 0, \forall i, j \in V; \)
4. \hspace{1cm} \textbf{while} NumC < p
5. \hspace{1cm} \varepsilon = \min\{ (i, j)\mid d_{ij} \text{ and } \text{visited}(ij) = 0, i, j \in V \};
6. \hspace{1cm} E' = E' \cup \{\varepsilon\};
7. \hspace{1cm} \left[C, \text{NumC}\right] = \text{procedure MCP}(G = (V, E'));
8. \hspace{1cm} \text{visited}(\varepsilon) = 1;
9. \hspace{1cm} \textbf{ endwhile}
10. \hspace{1cm} \textbf{while} NumC > p
11. \hspace{1cm} v = \max\{i\mid \sum_{j \in C} d_{ij}, i \in C\};
12. \hspace{1cm} C = C \setminus \{v\};
13. \hspace{1cm} \text{NumC} = \text{NumC} - 1;
14. \hspace{1cm} \textbf{ endwhile}
15. \hspace{1cm} \text{return } C

3.2 Genetic algorithm for CpLP

The solution is coded as a vector of \( V \) dimensions, and \( V \) is vertex set. Each vector is a chromosome, and each gene is binary corresponding to variable \( y_i \). The fitness function of a chromosome is defined as follows.

\[
\text{fitness}(y) = M \left/ \left( \frac{1}{2} \sum_{i \in V} \sum_{j \in V} d_{ij} y_i y_j + M' \times \left| p - \sum_{i \in V} y_i \right| + 1 \right) \right. \hspace{1cm} (*)
\]

\( M \): constant.

\( M' \): penalty factor for the number of uncovered nodes.

\( P_c \): the probability of crossover operation.

\( P_m \): the probability of mutation operation.

\( NP \): size of the population.

\( NG \): the generation of GA.

\( |p - \sum_{i \in V} y_i| \) is absolute value function. The crossover and mutation are simple binary valued and roulette wheel selection is applied. Then, the genetic algorithm is shown below.
Genetic algorithm for CpLP

Step 1. Generate the initial population. And calculate the fitness of each chromosome according to (*).

Step 2. (Selection) If generation is more than $NG$, stop and output the best chromosome; else, applying roulette wheel selection method to generate the next population.

Step 3. (Crossover) For two selected chromosomes in the population, applying simple binary valued crossover method to generate two chromosomes according to $P_c$.

Step 4. (Mutation) For any chromosome, randomly choose a gene to mutate according to $P_m$. Go to Step 2.

4 Experimental results

In this section, random method is introduced to generate the basic connected graphs. Vertices are placed randomly on a $400 \times 400$ square. Two vertices are adjacent when their Euclidean distance is no more than a predetermined radius. Two such graphs are shown as Figure 1.

![Figure 1 Two randomly generated graph with 100 vertices and radius is 80 and 150](image)

We have implemented the algorithm based on maximum clique procedure and genetic algorithm in Matlab 7.1 and carried out computational experiments to evaluate them. The computer used has a Inter(R) Core(TM) i5 2.30GHz CPU and Windows 7 operating system.

Parameters for genetic algorithm are set as follows. $NP = 100, NG = 500, P_c = 0.6, P_m = 0.1, M = 1000, M' = 1000000$.

In the first group of instances, we will analyse the performance of our presented algorithms when the radius is fixed and number of vertices increases. $p = 10$ facilities need to be chosen from the vertex set. Radius is 100, and the number of vertices is increasing from 40 to 1000. In this group of instances, sparse or dense graphs are also considered. Experimental results are shown in Table 1, from which we can draw the following conclusions.

1) The objective value obtained from algorithm based on Procedure MCP is better than that of genetic algorithm. According to further analysis by comparing with the lower bound of CpLP, the produced value of algorithm based on Procedure MCP is about 3 to 7 times of lower bound while the objective value of genetic algorithm is about 5 to $2.1 \times 10^5$ times of lower bound. Figure 2 shows the comparison among these values.

2) The solution output from genetic algorithm is not stability especially for large scale instances. When the number of vertices goes to 1000, there are 445 vertices been chosen. At
the same time, the objective value becomes higher $2.0 \times 10^7$, which is a very bad solution.

3) When the number of vertices is lower than 100, both of these two algorithms can find better solution.

4) The computational CPU time of algorithm based on Procedure MCP is much faster than that of genetic algorithm under the current parameters.

<table>
<thead>
<tr>
<th>Number of vertices</th>
<th>Number of edges</th>
<th>Genetic algorithm</th>
<th>Algorithm based on procedure MCP</th>
<th>Lowerbound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of facilities</td>
<td>Cost (seconds)</td>
<td>Cputime (seconds)</td>
<td>Number of facilities</td>
</tr>
<tr>
<td>40</td>
<td>127</td>
<td>6.38E+03</td>
<td>5.632</td>
<td>10</td>
</tr>
<tr>
<td>50</td>
<td>198</td>
<td>7.44E+03</td>
<td>7.317</td>
<td>10</td>
</tr>
<tr>
<td>60</td>
<td>284</td>
<td>7.74E+03</td>
<td>9.095</td>
<td>10</td>
</tr>
<tr>
<td>70</td>
<td>422</td>
<td>1.04E+03</td>
<td>11.201</td>
<td>10</td>
</tr>
<tr>
<td>80</td>
<td>509</td>
<td>8.32E+03</td>
<td>13.447</td>
<td>10</td>
</tr>
<tr>
<td>90</td>
<td>644</td>
<td>6.42E+03</td>
<td>16.552</td>
<td>10</td>
</tr>
<tr>
<td>100</td>
<td>808</td>
<td>7.78E+03</td>
<td>18.861</td>
<td>10</td>
</tr>
<tr>
<td>110</td>
<td>983</td>
<td>7.19E+03</td>
<td>21.84</td>
<td>10</td>
</tr>
<tr>
<td>120</td>
<td>1206</td>
<td>8.37E+03</td>
<td>25.07</td>
<td>10</td>
</tr>
<tr>
<td>130</td>
<td>1455</td>
<td>9.93E+03</td>
<td>28.813</td>
<td>10</td>
</tr>
<tr>
<td>140</td>
<td>1609</td>
<td>9.28E+03</td>
<td>32.604</td>
<td>10</td>
</tr>
<tr>
<td>150</td>
<td>1855</td>
<td>1.07E+04</td>
<td>36.707</td>
<td>10</td>
</tr>
<tr>
<td>200</td>
<td>3097</td>
<td>1.98E+05</td>
<td>61.605</td>
<td>10</td>
</tr>
<tr>
<td>300</td>
<td>6987</td>
<td>9.46E+05</td>
<td>132.257</td>
<td>10</td>
</tr>
<tr>
<td>500</td>
<td>19392</td>
<td>4.88E+06</td>
<td>389.096</td>
<td>10</td>
</tr>
<tr>
<td>1000</td>
<td>80342</td>
<td>2.00E+07</td>
<td>1837.231</td>
<td>10</td>
</tr>
</tbody>
</table>

Figure 2  Comparison of objective value among lower bound, algorithm based on Procedure MCP and genetic algorithm
In the second group of instances, we will analyse the performance of our presented algorithms when the number of vertices is fixed and the radius is different. Also, \( p = 10 \) facilities need to be chosen from the vertex set. There are 100 vertices randomly placed on the square. The radius changes from 60 to 570. Obviously, the graph will be complete when radius is larger than \( 400 \sqrt{2} \). Experimental results are shown in Table 2, from which we can draw the following conclusions.

### Table 2: Computational results for fixed number of vertices

<table>
<thead>
<tr>
<th>Radius of edges</th>
<th>Number of facilities</th>
<th>Genetic algorithm</th>
<th>Algorithm based on procedure MCP</th>
<th>Lowerbound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Number of edges</td>
<td>Cost (seconds)</td>
<td>Number of edges</td>
</tr>
<tr>
<td>60</td>
<td>368</td>
<td>10</td>
<td>1.16E+04 18.767</td>
<td>10</td>
</tr>
<tr>
<td>70</td>
<td>422</td>
<td>10</td>
<td>8.31E+03 18.798</td>
<td>10</td>
</tr>
<tr>
<td>80</td>
<td>567</td>
<td>10</td>
<td>1.01E+04 18.798</td>
<td>10</td>
</tr>
<tr>
<td>90</td>
<td>645</td>
<td>10</td>
<td>966E+03 18.798</td>
<td>10</td>
</tr>
<tr>
<td>100</td>
<td>861</td>
<td>10</td>
<td>9.90E+03 18.908</td>
<td>10</td>
</tr>
<tr>
<td>110</td>
<td>980</td>
<td>10</td>
<td>8.37E+03 18.907</td>
<td>10</td>
</tr>
<tr>
<td>120</td>
<td>1147</td>
<td>10</td>
<td>7.28E+03 18.814</td>
<td>10</td>
</tr>
<tr>
<td>130</td>
<td>1205</td>
<td>10</td>
<td>9.13E+03 18.829</td>
<td>10</td>
</tr>
<tr>
<td>140</td>
<td>1611</td>
<td>10</td>
<td>7.41E+03 18.938</td>
<td>10</td>
</tr>
<tr>
<td>150</td>
<td>1574</td>
<td>10</td>
<td>8.23E+03 18.829</td>
<td>10</td>
</tr>
<tr>
<td>200</td>
<td>2387</td>
<td>10</td>
<td>7.73E+03 18.783</td>
<td>10</td>
</tr>
<tr>
<td>250</td>
<td>3135</td>
<td>10</td>
<td>8.41E+03 18.783</td>
<td>10</td>
</tr>
<tr>
<td>350</td>
<td>4562</td>
<td>10</td>
<td>9.46E+03 18.907</td>
<td>10</td>
</tr>
<tr>
<td>570</td>
<td>4950</td>
<td>10</td>
<td>8.40E+03 18.845</td>
<td>10</td>
</tr>
</tbody>
</table>

1) The performance of algorithm based on Procedure MCP is better than that of genetic algorithm not only for the objective value but also for computational CPU time.

2) Both of the two algorithms are stability.

3) The objective values and computational CPU times of the presented algorithms do not vary too much when radius grows larger.

4) The CpLP is not sensitive for the radius when the number of vertices is fixed.

In the third group of instances, we will analyse the performance of our presented algorithms when the number of vertices and radius are all fixed while number of facilities changes. There are 100 vertices randomly placed on the square and radius is 100. The number of facilities increases from 5 to 50. Table 3 shows the computational results.

1) The algorithm based on Procedure MCP can produce better solution no matter number of facilities is small or large and the algorithm is more stability. The objective value is
about 2 to 5 times of the lower bound. The genetic algorithm can also produce better solution only when the number of facilities is large, while it is not a better algorithm for small number of facilities.

2) The computational CPU time of algorithm based on Procedure MCP becomes much longer when the number of facilities increases. While the genetic algorithm is not sensitive for the facility number too much, it is because the coding method is up to total number of vertices.

Table 3 Computational results for fixed connectivity radius and number of vertices

<table>
<thead>
<tr>
<th>p</th>
<th>Number of edges</th>
<th>Genetic algorithm</th>
<th>Algorithm based on procedure MCP</th>
<th>Lowerbound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Cost (seconds)</td>
<td>Cputime (seconds)</td>
<td>Cost/low</td>
</tr>
<tr>
<td>5</td>
<td>617</td>
<td>2.11E+03</td>
<td>18.845</td>
<td>37.79</td>
</tr>
<tr>
<td>10</td>
<td>593</td>
<td>1.91E+04</td>
<td>18.892</td>
<td>11.62</td>
</tr>
<tr>
<td>15</td>
<td>563</td>
<td>1.91E+04</td>
<td>18.892</td>
<td>11.62</td>
</tr>
<tr>
<td>20</td>
<td>537</td>
<td>4.19E+04</td>
<td>18.845</td>
<td>9.16</td>
</tr>
<tr>
<td>25</td>
<td>538</td>
<td>6.55E+04</td>
<td>18.829</td>
<td>7.89</td>
</tr>
<tr>
<td>30</td>
<td>558</td>
<td>1.01E+05</td>
<td>18.798</td>
<td>7.39</td>
</tr>
<tr>
<td>35</td>
<td>613</td>
<td>1.27E+05</td>
<td>18.923</td>
<td>5.75</td>
</tr>
<tr>
<td>40</td>
<td>592</td>
<td>1.50E+05</td>
<td>18.845</td>
<td>4.36</td>
</tr>
<tr>
<td>45</td>
<td>556</td>
<td>1.99E+05</td>
<td>18.829</td>
<td>3.89</td>
</tr>
<tr>
<td>50</td>
<td>539</td>
<td>2.56E+05</td>
<td>18.798</td>
<td>3.55</td>
</tr>
</tbody>
</table>

From the above groups of instances, the algorithm based on Procedure MCP always produces better solution. Furthermore, other better procedure for maximum clique problem can improve the performance of our presented algorithm. On the other hand, genetic algorithm can also produce better solution if increase the size of the population and the generation of GA.

5 Conclusion

In this paper, we present a new location analysis method by considering the distance between each pair of selected vertices, which is called connected p-facilities location problem (CpLP). This problem is proved to be NP-hard and mathematical model is formulated. For solving CpLP, we present two algorithms. One is based on maximum clique procedure, while the other one is genetic method. From the computational results of three groups of instances, these two algorithms can produce better solutions by comparing with the lower bound of CpLP.

For further researches, new coding methods can improve the performance of genetic algorithm. Another direction is to design new algorithm based on relaxing the mathematical model of CpLP.

References