Reactive GRASP for the Capacitated Single Allocation $p$-Hub Median Problem

Ching-Jung TING$^{a}$, Kuo-Rui LU$^{b}$, Kun-Chih WU$^{c}$

$^{a,b,c}$ Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan City, Taiwan 32003, ROC

$^{a}$ E-mail: ietingcj@saturn.yzu.edu.tw
$^{b}$ E-mail: s995428@mail.yzu.edu.tw
$^{c}$ E-mail: s968907@mail.yzu.edu.tw

Abstract: In this paper we consider a capacitated single allocation $p$-hub median problem (CSApHMP) which is to determine the location of $p$ hubs, the allocation of non-hub nodes to hubs in the logistics network. This problem is formulated as an integer programming model with the objective of minimizing the sum of total transportation cost and fixed cost of the selected $p$ hubs. Since the CSApHMP is NP-hard, it is difficult to obtain optimal solution within a reasonable computational time with exact solution approach. We only solve the CSApHMP using Gurobi optimizer for the small sized problems. A reactive greedy randomized adaptive search procedure (GRASP) is proposed to solve the problem. To illustrate the effectiveness of the proposed reactive GRASP algorithm, a comparative study from the benchmark instances is also presented. The experimental results show that the proposed GRASP heuristic can be an effective solution method for the capacitated hub location problem.

Keywords: GRASP, capacitated, $p$-hub median problem, single allocation, hub-and-spoke

1. INTRODUCTION

The hub location problem (HLP) deals with designing hub-and-spoke networks where hub locations and paths for transporting flows between every origin-destination pair of nodes have to be determined. Instead of providing direct links to every origin-destination pair, the hubs serve as consolidation, transshipment or switching points for flows between non-hub nodes. Flows departing from an origin are collected in a hub, transferred between hubs if necessary, and finally distributed to a destination node by consolidating flows from several origins with the same destination. The hub facilities consolidate flows in order to take advantage of economies of scale in transportation cost between hubs. Campbell (1994) presented four types of discrete HLPs: the $p$-hub median problem, the uncapacitated hub location problem, the $p$-hub center problem, and hub covering problem. In this paper it is intended to work with the $p$-hub median problem.

The $p$-hub median problem ($p$HMP) selects $p$ hubs amongst $n$ nodes and assigns non-hub nodes to a hub facility to minimize the cost of routing the flow. In $p$HMP, there are two allocation strategies in most of research: each non-hub node can be allocated either to one hub (single allocation) or to multiple hubs (multiple allocation). The transportation cost for flow between an OD ($i,j$) pair served via hubs $k$ and $m$ includes three cost components: collection (from origin $i$ to hub $k$), transfer (between hubs $k$ and $m$), and distribution (from hub $m$ to destination $j$). Each of these components has a cost coefficient of $\lambda$, $\alpha$, and $\delta$, respectively (O’Kelly, 1987). Due to the inter-hub transfer efficiencies, $\alpha$ is smaller than $\lambda$ and $\delta$. A fixed
cost associated with the establishing a node into a hub has to be paid.

During the past years, $p$HMP has been widely studied in the number of applications in telecommunication, transportation and logistics systems. For example, Takano and Arai (2009) applied the $p$HMP for the Asian hub ports with Los Angeles and Rotterdam in containerized cargo transport. Lin et al. (2012) solved the Chinese H&S air cargo network.

The $p$HMP belongs to the class of $NP$-hard problems (Love et al., 1988). In this paper, we tackle a particular variant of the $p$HMP, known as the capacitated single allocation $p$-hub median problem (CSApHMP). The objective is to minimize the sum of the overall fixed cost of established $p$ hubs and transportation cost in a network with $n$ demand nodes. The CSApHMP is a $NP$-hard problem. Exact solution approach cannot provide solutions for large scale practical hub location problem in a reasonable computational time. The aim of this paper is to propose a greedy randomized adaptive search procedure (GRASP) heuristic with three local search approaches for the CSApHMP. GRASP was preferred to other metaheuristics because of its relatively simple implementation and its small number of control parameters.

The remainder of this paper is organized as follows. In Section 2 we present the previous works in p-hub median problems. Section 3 introduces the mathematical formulations of the problem. The proposed GRASP heuristic is described in Section 4. Computational results of benchmark instances and comparisons with optimal solutions from optimization software Gurobi and Lagrangain relaxation heuristic are reported in Section 5. We conclude the paper and future research directions in Section 6.

2. Previous Works

There are several variants of hub location problems based on particular characteristics of the problems. Klincewicz (1998), Campbell et al. (2002), Alumur and Kara (2008), Campbell and O’Kelly (2012) and Farahani et al. (2013) provided a good survey on different hub location problems. We refer readers to these excellent survey papers therein.

Most of the studies on hub location problem dealt with uncapacitated cases (Farahani et al., 2013). O’Kelly (1987) presented a quadratic mathematical formulation for the single allocation $p$-hub median problem and proposed two heuristic methods, HEUR1 and HEUR2, to solve the problem. Both heuristics enumerate all possible choices of $p$ hub locations and assign the node to the nearest hub for the former heuristic, while assign the node to the first and second nearest hubs for the latter one. Aykin (1990) formulated the difference in the objective function if a non-hub node is assigned to different hub and defined a procedure to find the optimal allocation of non-hub to a given set of hubs. Campbell (1992) provided a linear integer programming model of the multiple allocation $p$-median problem.

Klincewicz (1991) developed several heuristics based on local improvement considering both the single and double exchange procedures and clustering of nodes. Later Klincewicz (1992) presented a tabu search (TS) and a greedy randomized adaptive search procedure (GRASP) heuristic for the uncapacitated $p$HLP in which they assigned non-hub nodes to the nearest hub. Skorin-Kapov and Skorin-Kapov (1994) computed the results of CAB data set by tabu search and compared with the heuristics of O’Kelly (1987) and TS of Klincewicz (1992). Their results were superior but required longer computational time. Campbell (1994) formulated the HLP as a mixed integer linear programming for four types of discrete hub location problems, which extend the hub location problem to consider more reality situations. Skorin-Kapov et al. (1996) modified Campbell (1994) formulation with tighter mixed integer linear programming relaxations. They tested the uncapacitated multiple allocation $p$-hub location problems (UMApHLP) with the objective function value within 1%
of the optimal solution by CPLEX. However, the results were not guaranteed to obtain all integral solutions.

Campbell (1996) proposed two new heuristics for the single allocation $p$-hub median problem based on the multiple allocation $p$-hub median solutions. In these two heuristics, the allocations are done according to different rules but location decisions are the same. O’Kelly et al. (1996) modified Skorin-Kapov et al. (1996) model by assuming a symmetric flow data and further reducing the problem size. The formulation can find integer solutions most of the time. They also provided the sensitivity analysis of the solution in terms of number of hubs and hub locations to the inter-hub discount factor $\alpha$. Smith et al. (1996) mapped the SApHMP onto a modified Hopfield neural network using O’Kelly’s (1987) quadratic integer programming formulation. The results of a practical postal delivery network (PDN) and the CAB data set demonstrated that the quality of these Hopfield network solutions compares favorably to those obtained using both exact method and simulated annealing.

Ernst and Krishnamoorthy (1996) provided a linear integer programming formulation for uncapacitated single allocation $p$-hub median problem (USApHMP). They developed a branch and bound method and a simulated annealing (SA) algorithm to obtain the upper bounds to improve the general branch and bound method. The SA was comparable with the TS of Skorin-Kapov and Skorin-Kapov (1994) in terms of solution quality and computational time. However, they cannot solve any problem with more than 50 nodes.

Sohn and Park (1997) provided a linear programming formulation for single allocation two-hub median problem and solved the two-hub location problem in polynomial time for instances with 100 nodes. Sohn and Park (1998) presented a further reduction of Skorin-Kapov et al. (1996) formulation for a model with fixed hub locations when the unit transportation cost is symmetric and proportional to the distance. Later Sohn and Park (2000) extended the problem to a three-hub location problem. They showed that this is a NP-hard problem and tested instance by a linear programming relaxation model.

Pirkul and ScHilling (1998) developed a Lagrangian relaxation method based on the Skorin-Kapov et al. (1996) formulation. They used subgradient optimization on the Lagrangian relaxation of the model and provided a cut constraint for one of the subproblems. Ernst and Krishnamoorthy (1998) proposed another branch-and-bound approach which solved shortest-path problems for each origin-destination pair to obtain lower bounds. They solved the largest single allocation problem to date with this algorithm with 100 nodes and with $p = 2$ and 3. Ebery (2001) optimized the new proposed formulas with two or three hubs using CPLEX, and also claimed that such a formula has the potential to solve even larger problems. Abdennour-helm (2001) discussed the solution quality by using the simulated annealing method to solve USApHMP and compared with the results of Skorin-Kapov and Skorin-Kapov (1994) and Campbell (1996).

Pérez et al. (2007) presented a hybrid algorithm that merges the variable neighborhood search (VNS) and the path-relinking (PR) paradigms. Both VNS and PR use systematic neighborhood-based strategies to explore the feasible region and yield adequate results even with large-sized problems. Their computational results showed that the proposed hybrid algorithm provided an efficient alternative for solving the $p$-hub median problem. Kratica et al. (2007) constructed two genetic algorithms (GAs) for the USApHMP. The numerical experiments showed that the GAs can solve the problem with up to 200 nodes and 20 hubs. Takano and Arai (2009) presented a genetic algorithm for the hub-and-spoke problem (GAHP) for the liner shipping with shuttle service. The GAHP was first validated by CAB instances and an example of the H&S network with shuttle services with 18 ports was also analyzed.
To the best of our knowledge, there are just a few articles in the literature dealing with capacitated single allocation p-hub median problems. Aykın (1994) studied the capacitated multiple allocation hub location problem in determining airline hub locations. Two heuristics and a lower bound based on Lagrangian relaxation were proposed. Ernst and Krishnamoorthy (1999) extended the Skorin-Kapov et al. (1996) formulation to the capacitated case and also proposed a mixed integer programming formulation. They proposed a simulated annealing and a random descent to obtain upper bounds. Ebery et al. (2000) presented formulations and shortest path based solution approaches for the capacitated multiple allocation hub location problem. Both CAB and AP data set were tested.

Marin (2005) presented a new formulation for the multiple allocation capacitated hub location problem based on the same idea used in Ebery et al. (2000) but an OD pair can be split into several routes. Labbé et al. (2005) studied a CSAHLp where only the operation costs associated with the flow were considered. A branch-and-cut algorithm was proposed to solve the problem up to 50 nodes. Pérez et al. (2005) proposed a GRASP-path relinking (GRASP-PR) for the CSApHMP. The greedy evaluation function has two phases: a location phase and an allocation phase. In the location phase, one hub is added at each time, while the allocation phase allocates a node to the nearest hub. Computational experiments showed that GRASP-PR provided better solutions than by either GRASP or PR individually on large scale instances. da Graça Costa et al. (2008) considered the CSAHLp using bi-criteria approach. They presented two models on the second objective, the first minimizes the time to processing flows, while the second minimizes the maximum service time at the hubs.

Chen (2008) developed a simulated annealing algorithm with three levels to resolve the CSAHLp. Computational results showed that the proposed heuristic outperformed the SA by Ernst and Krishnamoorthy (1999). Randall (2008) developed four variations of the ant colony optimization metaheuristic that explores different construction modeling choices. Contreras et al. (2009) proposed a Lagrangean relaxation to obtain tight upper and lower bounds on CSAHLp. Computational experiments on benchmark instances and new generated large size instances showed that the LR obtained or improved the best known solution. Stanimirović (2010) proposed a genetic algorithm for solving the CSApHMP that is to minimize the total transportation cost. The GA can reach all optimal solution for instances up to 50 nodes. Contreras et al. (2011) solved capacitated single allocation hub location problems with up to 200 nodes using a branch-and-price algorithm that integrates lower bounds from Lagrangian relaxation. Lu and Ting (2013) proposed a Lagrangian relaxation heuristic to solve the CSApHMP. The LR heuristic tested AP benchmark instances up to 50 nodes. Stanojević et al. (2015) proposed a hybrid method that is consisted of an evolutionary algorithm and a parallel branch-and-bound method (EA-BnB) for solving the CSAHLp. The hybrid algorithm was tested on the standard AP data sets with up to 300 nodes and compared to other heuristic approaches.

3. MATHEMATICAL FORMULATION

The CSApHMP considers a network of n demand nodes and p hubs must be located. Each non-hub node is allocated to a single hub that does not violate the capacity. The flow between any pair of nodes (i, j) must be routed through either one or at most two hubs k and l, in which node i is allocated to hub k and node j is allocated to hub l. It is assumed that direct transportation between i and j is not possible. The cost of transport a unit of flow along the path i-k-l-j is computed as $C_{ijkl}$. Some research considered fixed cost of the selected hubs
(O’kelly et al., 1996; Contreras et al., 2009), while others only considered the transportation cost (Campbell, 1996; Labbé et al., 2005). In this paper, the objective is to minimize the sum of flow transportation cost and fixed cost of locating facilities.

3.1 Notations

The following notations are used throughout the paper.

Parameters

- \( A_k \): fixed cost of locating a hub at node \( k \)
- \( C_{ijkl} \): the transportation cost of a unit of flow from origin \( i \) to destination \( j \) routed via hubs \( k \) and \( l \), \( C_{ijkl} = \lambda d_{ik} + \alpha d_{kl} + \delta d_{jl} \)
- \( d_{ij} \): the distance between nodes \( i \) and \( j \)
- \( N \): set of nodes
- \( p \): number of required hubs
- \( Q_k \): capacity of node \( k \)
- \( w_{ij} \): the flow between nodes \( i \) and \( j \)
- \( \alpha \): the unit flow costs for transfer
- \( \lambda \): the unit flow costs for collection
- \( \delta \): the unit flow costs for distribution

Decision variables

- \( X_{ijkl} \): if the flow from node \( i \) to \( j \) routed via hubs \( k \) and \( l \)
- \( Y_{ik} \): if node \( i \) is allocated to hub \( k \)

3.2 The Model

In this paper, we revised the mathematical formulation for the uncapacitated \( p \)HMP presented by Campbell (1996). The mixed integer formulation is as follows.

\[
\text{Min} \quad \sum_{i \in N} \sum_{j \in N} \sum_{k \in N} \sum_{l \in N} w_{ij} C_{ijkl} X_{ijkl} + \sum_{k \in N} A_k Y_{ik} \quad (1)
\]

\[
\text{S.T.} \quad \sum_{k \in N} Y_{ik} = p \quad (2)
\]

\[
Y_{ik} \leq Y_{kk} \quad \forall i, k \in N \quad (3)
\]

\[
\sum_{k \in N} \sum_{l \in N} X_{ijkl} = 1 \quad \forall i, j \in N \quad (4)
\]

\[
\sum_{l \in N} X_{ijkl} = Y_{ik} \quad \forall i, j, k \in N \quad (5)
\]

\[
\sum_{k \in N} X_{ijkl} = Y_{jl} \quad \forall i, j, l \in N \quad (6)
\]

\[
\sum_{i \in N} \sum_{j \in N} w_{ij} Y_{ik} \leq Q_k Y_{kk} \quad \forall k \in N \quad (7)
\]

\[
Y_{ik} \in \{0, 1\} \quad \forall i, k \in N \quad (8)
\]

\[
X_{ijkl} \in \{0, 1\} \quad \forall i, j, k, l \in N \quad (9)
\]

The objective function (1) minimizes the sum of transportation cost of flows and the
fixed cost of established facilities. Constraint (2) stipulates that exactly $p$ hubs are chosen. Constraint (3) ensures that node $i$ can be allocated to hub $k$ only if $k$ is selected as a hub. Constraint (4) states that every node is allocated to exactly one hub. Constraints (5) and (6) ensure that flow from origin $i$ to designation $j$ cannot be allocated to a hub pair $k$ and $l$ via path $i$-$k$-$l$-$j$ unless node $i$ is allocated to hub $k$ and $j$ is allocated to hub $l$. Constraint (7) ensures that all the assigned demand to an opened facility must less than or equal to the capacity. Constraints (8) and (9) are binary integrality constraints.

4. GRASP

GRASP is a multi-start or iterative metaheuristic and is a memoryless procedure for combinatorial optimization problems. Each iteration of the GRASP is consisted of two phases, construction and local search (Feo and Resende, 1989). In construction phase a feasible solution is produced, one element at a time. If the solution is not feasible, a repair mechanism is applied to achieve feasibility or it is discarded. Once a feasible solution is obtained, a local optimum in the neighborhood of the constructed solution is sought in local search phase. The best overall solution is kept as the result. The advantage of GRASP is that few parameters need to be set and tuned. Therefore development can focus on implementing efficient data structures to assure quick GRASP iterations. We refer the reader to the excellent surveys by Festa and Resende (2009a) and Resende and Ribeiro (2014).

GRASP has been applied in scheduling, routing, logic, partitioning, location, graph theory, assignment, manufacturing, transportation, telecommunications, biology and related fields, automatic drawing, power systems, and VLSI design (Festa and Resende, 2009b). The general GRASP procedures are as follows.

Set parameter

While the termination criteria are not satisfied do

Construct greedy random solution(s)
Local search(s)
Solution updates(s, Best_s)

Stop

Output best solution(Best_s)

For the construction phase, a restricted candidate list (RCL) is created. The RCL is made up of the elements with the smallest incremental cost of the greedy function. This list can be limited either by the number of elements (cardinality-based) or by their quality (value-based). In the former case, it consists of pre-determined number of elements with the best incremental costs. The latter case is associated with a control parameter $\gamma$ ($\gamma \in [0, 1]$). The RCL is formed by all feasible elements which can be inserted into the partial solution without destroying feasibility and whose quality is better than the threshold as eq. (10).

$$RCL = \{k \mid g_k < g^*(1+\gamma)\} \quad (10)$$

where $g_k$ is the greedy function value of element $k$, $g^*$ is the best solution so far. The case $\gamma = 0$ corresponds to a pure greedy algorithm, while $\gamma = 1$ represents a random construction. $\gamma$ is recommended as 0.2 which leads to good solutions (Resende and Ribeiro, 2014). The one element is chosen from the RCL, not necessary the one with best greedy function value.

Prais and Ribeiro (2000) showed that using a single fixed value $\gamma$ might not able to find a high-quality solution. An alternative is to use a different value which is chosen uniformly at random in the interval $[0, 1]$. They proposed another alternative, called reactive GRASP, in
which the RCL parameter $\gamma$ is self-adjusted according to the solution qualities previously obtained. In this paper, a reactive GRASP is used. Figure 1 shows the framework for the GRASP.

4.1 Construction Phase

To obtain a solution for the CASpHMP we first locate the $p$ hubs and then determine the assignment of each non-hub nodes to one of the $p$ hubs. Let $m$ be the fixed number of allowed values and $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ the list of these control values. The $\gamma$ is randomly selected in $\Gamma$ before adding one element and that the probabilities of choosing $\gamma_i$ are progressively modified based on the average solution values.

The probabilities of taking $\gamma_i$ are initialized to $1/m$ (uniform distribution). During the search process, the number of solution built with each value $\gamma_i$ is recorded and the corresponding solution values $b_i$. Let $f^*$ be the best solution value found till now. $q_i$ is the ratio of best solution value $f^*$ to the average solution value $b_i$. For each iteration, the probability $p_i$ is updated to favor the values yielding the best average solution values based on eqs. (11) and (12).

\[
q_i = \frac{f^*}{b_i}, \quad \forall i = 1, \ldots, m \tag{11}
\]

\[
p_i = \frac{q_i}{\sum_{j=1}^{m} q_j}, \quad \forall i = 1, \ldots, m \tag{12}
\]

Once the control parameter is chosen, the RCL is determined based on eq. (10). After some iterations, the reactive GRASP tends to use the control parameter that has provided the best average solution values.

Let $k$ be a candidate location for a hub. For any node $j$ that could be assigned to $k$ we need to consider that all the flow from $j$ to any other node $i$, will be routed through $k$. The greedy function we will use involves summing the cost changes due to each node $j$ that would be reassigned to $k$, if $k$ were added as a hub and the fixed cost of node $k$ as eq. (13). (For locating the first hub, when $K$ is empty, the greedy function is computed by assuming the initial assignment cost to be a large number.)

\[
g_k = \sum_{j \in N} \sum_{i \in K, h \in K} \min(d_{jh}, d_{jh})w_j + A_k \tag{13}
\]

Detailed procedures of the construction phase of each iteration are as follows.

**Step 1** Set $p_1 = 1/m$, $e = 1$.

**Step 2** Draw $i \in \{1, 2, \ldots, m\}$ with probability $p_i$, and the $\gamma_i$.

**Step 3** Compute the greedy function value of each candidate location ($g_i$) based on eq. (13) and the best greedy function value $g^*$.

**Step 4** RCL = \{ $k \mid g_k \leq g^* (1 + \gamma_i)$ \}.

**Step 5** Based on the greedy function values in RCL, compute $o_i = 1/g_i$, the probability of selecting $i s_i$ in RCL is computed as

\[
s_i = \frac{o_i}{\sum_{j \in RCL} o_j} \tag{14}
\]

**Step 6** Generate a probability in $[0, 1]$ based on roulette wheel selection of eq. (14) as a
new hub.

Step 7  \( e = e + 1, \) if \( e \leq p \), go to step 3; otherwise, stop.

Note that once a solution is found in an iteration, the control parameter \( \Gamma \) and the probabilities of selecting each control parameter \( p_i \) must be updated for next iteration.

![Figure 1. The GRASP flowchart for solving the CSapHMP](image)

4.2 Local Search Phase

Since the solution found by the construction phase is not necessary a local optimum, local search methods can be applied to improve it. The local search procedure of the GRASP algorithm by Klincewicz (1992) is based on the 2-exchange. We propose three types of moves: node shift, node swap, and hub swap. Node shift moves are intended to change the assignment of a random selected node to another hub. Node swap moves are to swap two nodes to their hubs. Hub swap is to swap a hub node with a non-hub node.
4.3 Solution Representation

To represent a solution, we need to specify the locations of $p$ hubs, the assignment of nodes to a hub. In particular, $S = \{Y, X\}$, where $Y = \{y_1, y_2, \ldots, y_p\} \subseteq N$ is the set of $p$ hub locations, $X = \{x_i \mid i \in N\}$ is the assignment of nodes to the selected hubs. Figure 2 shows a solution for 10 nodes with 3 hub location. The selected hubs are nodes 1, 3, and 4 while the assignment for the nodes are 1, 4, 3, 4, 4, 3, 1, 3, 3, 1, respectively.

\begin{table}[h]
\centering
\begin{tabular}{cccccccccccc}
\hline
$y_1$ & $y_2$ & $y_3$ & $x_1$ & $x_2$ & $x_3$ & $x_4$ & $x_5$ & $x_6$ & $x_7$ & $x_8$ & $x_9$ & $x_{10}$ \\
\hline
1   & 3   & 4   & 1   & 4   & 3   & 4   & 4   & 3   & 1   & 3   & 3   & 1   \\
\hline
\end{tabular}
\caption{A solution representation for 10 nodes and 3 hubs}
\end{table}

5. COMPUTATIONAL EXPERIMENTS

In this section we present results for the reactive GRASP heuristic that is described in previous section. The solutions are compared against the results obtained by using Gurobi 4.5.2 optimizer and Lagrangian relaxation (LR) by Lu and Ting (2013). All tests were carried out on a PC of Intel Core2 Duo 3.0GHz CPU with 2 GB RAM, running under Windows 7 operations systems. The reactive GRASP was coded in Microsoft Visual Studio 2010 C++ and tested on four sets of AP instances in OR-Library taken from Beasley (1996) which are available at http://people.brunel.ac.uk/~mastjjb/jeb/info.html. We set the computational time for Gurobi optimizer for 2 hours. After a preliminary experiment, we set the control parameter set $\Gamma = \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Every instance is run for 20 times.

The AP data set is derived from a real-world hub location problem for Australia Post (AP) delivery system by Ernst and Krishnamoorthy (1996). The smaller size AP instances up to fifty nodes with the sized $n = 10, 20, 25, 40$, and 50. These instances include capacities and fixed costs on nodes which were described in Ernst and Krishnamoorthy (1999). They used a combination of two types of fixed cost, tight (T) and loose (L), and two possibilities of capacities of the hubs, tight (T) and loose (L) for each problem size. Instances with fixed costs of type T, nodes with larger flows have higher fixed costs. This makes it more difficult to solve. For every problem size the four instances correspond to one of the four possible combinations, LL, LT, TL, and TT. The number of hubs $p$ in tested instances is up to 5.

Tables 1 to 4 show the results of Gurobi optimizer, Lagrangean relaxation, and our reactive GRASP approach for all instances. Columns 1 and 2 are number of nodes $n$ and number of hubs $p$. Columns 3 and 4 are the optimal solutions and CPU times of Gurobi. Columns 5 to 8 are the gaps and CPU times for LR and GRASP, respectively. The Gurobi can only solve the instance up to 25 nodes ($n \leq 25$). For the instances with more than 25 nodes, the Gurobi optimizer cannot find the optimal solution within 2 hours (marked as a dash “-”). For instances with $n > 25$, only results for CSAHLR are available. We present the results from Ernst and Krishnamoorthy (1999) for comparison. The solution quality is evaluated as a gap which is the relative deviation from the optimal solution obtained by Gurobi or best known solution from literature as eq. (15).

$$\text{gap} = \frac{\text{solution} - \text{Opt.}}{\text{Opt.}} \times 100\% \quad (15)$$

For the medium size with more 25 nodes in tight capacity, the number of hubs must be at least 3. We notice that as the number of hubs $p$ increases for a given number of nodes $n$, the
CPU time increases.

Table 1 provides the results for the LL instances. We observe that our GRASP can obtain the optimal solutions for all instances, while LR cannot obtain the optimal solution in one instance. The computational time decreases as the number of hubs increases.

<table>
<thead>
<tr>
<th>n</th>
<th>p</th>
<th>Gurobi Opt. time</th>
<th>LR gap time</th>
<th>GRASP gap time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>230008.5 1.14</td>
<td>25.28 0.00</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>224250.1 1.21</td>
<td>2.32 0.00</td>
<td>0.12</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>229172.6 1.32</td>
<td>2.31 0.00</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>239292.3 1.21</td>
<td>1.28 0.00</td>
<td>0.05</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>234691.0 18.13</td>
<td>79.23 0.00</td>
<td>0.36</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>239444.2 20.56</td>
<td>35.54 0.00</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>251939.7 20.91</td>
<td>38.80 0.00</td>
<td>0.32</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>266745.2 19.00</td>
<td>15.18 0.00</td>
<td>0.29</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>238978.0 87.19</td>
<td>190.37 0.00</td>
<td>0.58</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>242437.2 88.64</td>
<td>132.29 0.00</td>
<td>0.43</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>252716.6 123.20</td>
<td>64.91 0.00</td>
<td>0.48</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>263518.3 59.97</td>
<td>74.68 0.00</td>
<td>0.55</td>
</tr>
<tr>
<td>40</td>
<td>2</td>
<td>241955.71 a</td>
<td>18.97 0.00</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>- b</td>
<td>17.10 0.00</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-</td>
<td>326.07 - 1.31</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-</td>
<td>22.36 - 1.24</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>238520.59 -</td>
<td>564.40 0.00</td>
<td>0.76</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-</td>
<td>530.10 - 1.68</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-</td>
<td>34.06 - 1.79</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-</td>
<td>37.78 - 2.14</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>0.02 110.65 0.00 0.70</td>
<td></td>
</tr>
</tbody>
</table>

a: Solution provided by Ernst and Krishnamoorthy (1999).
b: Gurobi cannot solve the instance within 2 hours.

Table 2 presents the results for the LT instances. As can be seen, the number of hubs must be at least 3 for instances with \( n > 25 \) in tight capacity. Our GRASP cannot find the optimal solution for \( n > 25 \). The LR cannot obtain 1 optimal solution for instances with \( n \leq 25 \). The GRASP and LR do not show good solution quality for instances with \( n > 25 \) as those for loose type instances in Table 1. The average gaps are 1.14% and 0.05% for LR and GRASP, respectively, which are much higher than the loose capacity instances in Table 1. The computational times for reactive GRAP for both LL and LT instances are almost the same; however, the LR needs more time for LT instances than those for LL as expected.

Table 3 provides the results for the TL instances. From table 3 we observe that our GRASP can obtain the optimal solutions for all instances. LR cannot obtain the optimal solutions in 2 instances. The average gap of those instances with available optimal solution is 0.04% for LR.
Table 2. The results for LT instances

<table>
<thead>
<tr>
<th>n</th>
<th>p</th>
<th>Gurobi</th>
<th>LR</th>
<th>GRASP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Opt.</td>
<td>time</td>
<td>gap</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>256048.6</td>
<td>1.17</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>252973.6</td>
<td>1.05</td>
<td>0.00</td>
<td>137.65</td>
</tr>
<tr>
<td>4</td>
<td>250992.3</td>
<td>1.03</td>
<td>0.00</td>
<td>22.91</td>
</tr>
<tr>
<td>5</td>
<td>261451.2</td>
<td>1.01</td>
<td>0.00</td>
<td>38.59</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>253517.4</td>
<td>639.65</td>
<td>2.99</td>
</tr>
<tr>
<td>3</td>
<td>257247.7</td>
<td>2079.15</td>
<td>0.14</td>
<td>18.39</td>
</tr>
<tr>
<td>4</td>
<td>260678.9</td>
<td>269.24</td>
<td>0.00</td>
<td>280.05</td>
</tr>
<tr>
<td>5</td>
<td>274975.4</td>
<td>291.50</td>
<td>0.00</td>
<td>217.75</td>
</tr>
<tr>
<td>25</td>
<td>3</td>
<td>276372.5</td>
<td>751.32</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>278235.0</td>
<td>271.99</td>
<td>0.00</td>
<td>1862.89</td>
</tr>
<tr>
<td>5</td>
<td>284952.6</td>
<td>73.54</td>
<td>0.00</td>
<td>377.67</td>
</tr>
<tr>
<td>40</td>
<td>3</td>
<td>272218.32</td>
<td>5.36</td>
<td>289.24</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>28.38</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>303.53</td>
</tr>
<tr>
<td>50</td>
<td>3</td>
<td>-</td>
<td>-</td>
<td>665.93</td>
</tr>
<tr>
<td>4</td>
<td>272897.49</td>
<td>6.30</td>
<td>49.98</td>
<td>0.91</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>41.33</td>
</tr>
</tbody>
</table>

Average: 1.14 364.26 0.05 0.67

a: Solution provided by Ernst and Krishnamoorthy (1999).
b: Gurobi cannot solve the instance within 2 hours.

Table 4 presents the results for the TT instances. As can be seen, the number of hubs must be at least 3 for instances with $n > 25$ in tight capacity. Our GRASP cannot find 2 optimal solutions. The LR cannot obtain optimal solutions in 4 instances and does not show good solution quality for instances with $n > 25$ as those loose type instances in Table 1. The average gap for all instances is 1.93% and 0.12% for LR and GRASP, respectively.

In general, we observe that instances with tight capacities and fixed costs are hard to solve. Note that the reactive GRASP is able to obtain optimal solutions in the loose capacity type instances within short computational times. The largest gap for all instances is 1.15% for the tight capacity instances. However, the performance of LR algorithm does not provide good solutions in tight capacity instances as GRASP.

5. CONCLUSION

In this paper, we tackle the capacitated single allocation $p$-hub median problem (CSApHMP). The problem has various practical real world applications in hub-and-spoke network design. We propose a reactive GRASP heuristic with three local search approaches to solve the problem. Four sets of benchmark instances from well-known AP hub data set are tested for our reactive GRASP heuristic. The results are also compared with the Gurobi optimizer and the LR approach.

Our reactive GRASP algorithm is able to obtain optimal solutions for instances $n \leq 25$. However, the performance is not as good for $n > 25$ instances. The present study shows that
Table 3. Test results for TL instances.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Gurobi</th>
<th>LR</th>
<th>GRASP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>gap</td>
<td>time</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>264544.0</td>
<td>0.00</td>
<td>12.59</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>263399.9</td>
<td>0.00</td>
<td>2.44</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>269074.3</td>
<td>0.00</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>281327.3</td>
<td>0.00</td>
<td>0.30</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>271128.2</td>
<td>0.00</td>
<td>54.09</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>281304.8</td>
<td>0.43</td>
<td>29.51</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>295223.9</td>
<td>0.00</td>
<td>22.98</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>310122.8</td>
<td>0.00</td>
<td>3.50</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>310317.6</td>
<td>0.00</td>
<td>113.42</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>328092.6</td>
<td>0.00</td>
<td>90.56</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>347416.8</td>
<td>0.00</td>
<td>74.59</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>368288.6</td>
<td>0.07</td>
<td>31.63</td>
</tr>
<tr>
<td>40</td>
<td>2</td>
<td>298919.01</td>
<td>0.00</td>
<td>231.11</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>323092.6</td>
<td>0.00</td>
<td>179.23</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>347416.8</td>
<td>0.00</td>
<td>116.99</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>368288.6</td>
<td>0.07</td>
<td>133.75</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>319015.77</td>
<td>0.00</td>
<td>381.79</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>330.78</td>
<td>0.00</td>
<td>330.78</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>399.26</td>
<td>0.00</td>
<td>399.26</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>344.76</td>
<td>0.00</td>
<td>344.76</td>
</tr>
</tbody>
</table>

Average: 0.04, 127.70, 0.00, 0.84

---

the GRASP is an effective method to solve CSApHMP. Problems of practical size are larger than any instances solved in this paper. Hence, future work can apply the GRASP algorithm for larger instances of the real world problem and compare to other heuristic approaches. The GRASP could also be used to solve the capacitated multiple allocation $p$-hub median problem (CMApHMP). We could also hybrid GRASP metaheuristic with path relinking to improve the solution quality, especially for the tight capacity instances. Other research direction might to apply GRASP metaheuristic for the multiple capacity levels selection of the CSApHMP.

REFERENCES


### Table 4. Test results for TT instances

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>264544.0</td>
<td>1.01</td>
<td>0.00</td>
<td>2.22</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2633 99.9</td>
<td>1.12</td>
<td>0.00</td>
<td>2.25</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>269074.3</td>
<td>1.19</td>
<td>0.00</td>
<td>0.48</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>281327.3</td>
<td>1.06</td>
<td>0.00</td>
<td>0.28</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>329068.6</td>
<td>1849.93</td>
<td>1.12</td>
<td>0.00</td>
<td>2.25</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.03</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>296035.4</td>
<td>24.96</td>
<td>0.00</td>
<td>0.48</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>306296.3</td>
<td>14.88</td>
<td>0.00</td>
<td>0.28</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>325568.5</td>
<td>14.57</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>3</td>
<td>348369.1</td>
<td>1442.95</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>369576.8</td>
<td>1364.41</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>391996.5</td>
<td>357.94</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>40</td>
<td>3</td>
<td>354874.10$^a$</td>
<td>-</td>
<td>4.66</td>
<td>190.96</td>
<td>1.15</td>
<td>1.16</td>
<td>1.12</td>
<td>1.16</td>
<td>0.03</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>- $^b$</td>
<td>-</td>
<td>-</td>
<td>12.93</td>
<td>-</td>
<td>1.20</td>
<td>-</td>
<td>1.20</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>147.19</td>
<td>-</td>
<td>1.59</td>
<td>-</td>
<td>1.59</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>3</td>
<td>417440.99</td>
<td>-</td>
<td>10.11</td>
<td>28.55</td>
<td>0.86</td>
<td>2.45</td>
<td>2.33</td>
<td>2.45</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>315.37</td>
<td>-</td>
<td>1.53</td>
<td>-</td>
<td>1.53</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>24.09</td>
<td>-</td>
<td>2.33</td>
<td>-</td>
<td>2.33</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Average: 1.93  | 63.25  | 0.12  | 0.74  |

$^a$: Solution provided by Ernst and Krishnamoorthy (1999).

$^b$: Gurobi cannot solve the instance within 2 hours.

---


