GRASP Metaheuristic for Multiple Allocation $\boldsymbol{p}$-Hub Location Problem

Ali Shobeiri

A Thesis
in
The Department
of
Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science in Industrial Engineering

Concordia University

Montreal, Quebec, Canada

Faculty of Engineering and Computer Science
Department of Mechanical and Industrial Engineering

July 2015

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Ali Shobeiri**

Entitled: **GRASP Metaheuristic for Multiple Allocation p-Hub Location Problem**

And submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science in Industrial Engineering

Complied with regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Dr. Rajamohan Ganesan Chair

Dr. Onur Kuzgunkaya Examiner

Dr. Satyaveer S. Chauhan Examiner

Dr. Ivan Contreras Supervisor

Dr. Navneet Vidyarthi Supervisor

Approved by

Chair of Department or Graduate Program Director

2015

Dean of Faculty of Arts and Science

ABSTRACT

Hub Location Problems (HLPs), belonging to the field of location theory, have been area of much research over the past two decades. This is due, in large measure, to the applications of hub and spoke networks in practice. Among the most classical versions of HLPs are p-hub location problems (p-HLPs), p-hub location problems are one of the most well studied variants of hub location literature. The primary goal of these models is to allocate p hub facilities in a hub and spoke network so as to concentrate flows (demands) to benefit from economies of scale in cost of transportation. The application of p-hub networks extends beyond the field of telecommunication and includes air freight systems, postal delivery systems and airline industries and several transportation related systems. p-HLPs constitute a challenging class of HLPs and are known to be NP-hard. Several solution approaches have been developed from exact solutions using integer programming techniques to the development of metaheuristics. Even though metaheuristic algorithms cannot guarantee optimality, given complexity of large scale HLPs, they are being used for solving these problems. In this thesis, we focus on the multiple allocation uncapacitated p-hub location problem. Four solution algorithms will be proposed to this problem for solving the Australian Postal (AP) data instances. We start with a very simple algorithm and continue with more complicated one in order to present an efficient high quality feasible solution and to assess the impact of the quality of initial feasible solution on local improvement phase. Computational results from the different algorithms were compared to exact solutions to track the efficiency of the proposed algorithms.

ACKNOWLEDGMENTS

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. I am deeply indebted to my supervisor Dr. Ivan Contreras. My appreciation extends beyond academic settings as he has been more than an advisor for me. I am also grateful to my co-supervisor Dr. Navneet Vidyarthi from John Molson School of Business, Concordia University.

I wish to express my deepest appreciation to my parent for their support during the special time of my studying.

I also wish to thank my friends and family for their constant moral support.

Dedicated to my father

Table of Contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1: Introduction

The problem of locating facilities in a manner so that they effectively serve a set of clients has been the subject of much research. The study of location theory formally began in early 1920's when Alfred Weber considered how to position a single warehouse so as to minimize the total distance between it and several customers (Kuhn, 1955). Following this initial investigation, location theory was driven by a few applications which inspired researchers from a range of fields. Location theory gained a renewed interest in 1964 with a publication by Hakimi (1964), who sought to locate switching centers in a communication network and police stations in a highway system. To do so, Hakimi (1964) considered the more general problem of locating one or more facilities on a network so as to minimize the total distance between customers and their closest facility or to minimize the maximum distance. Facility location is a critical aspect of strategic planning for a broad spectrum of public and private firms.

Hub location research became an important area of location theory over the past two decades. This is because of the frequent employment of hub and spoke networks in modern transportation and telecommunication systems. These systems serve demand for travel or communication between many origins and many destinations, where economies of scale exist in the cost for such travel or communications (Campbell et al. 1996). The key feature of these systems is in the way demand is routed; rather than routing every demand with a direct link from its origin and destination points, demand is routed via specific subset of links namely called hub and spoke network (as shown in Figure 1).

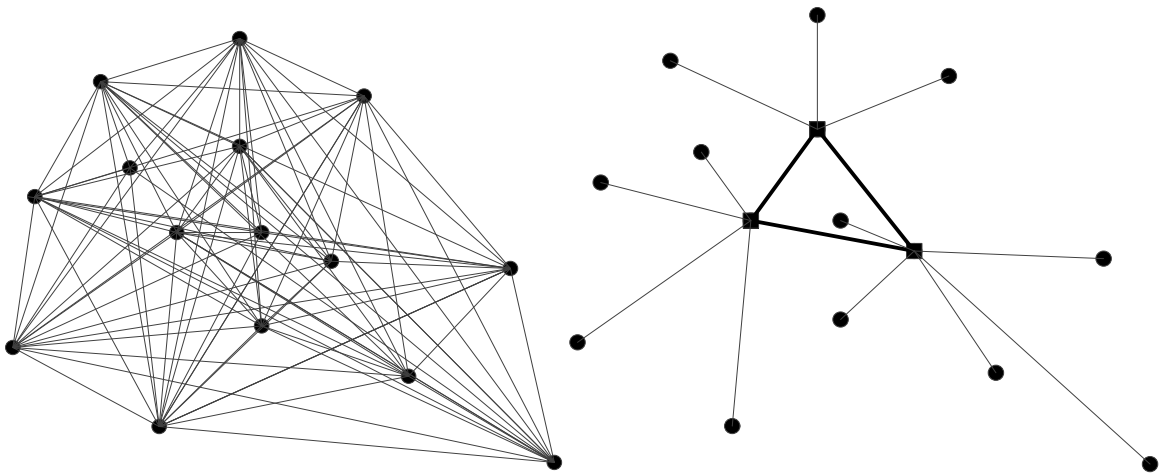

Figure 1.Routed Network with hub (right) and without hub (left)

The use of fewer links concentrates flows and allows economies of scale to be exploited. Given this, hub location problems involve locating hub facilities and designing hub networks (Campbell et al., 2002). The locations of the hubs as well as the paths for sending the flows between the origin-destination pairs are the most important decisions to this problem. Once designed, such a network allows a large set of origins and destinations to be connected with a relatively few links, via central hub facilities. In short, HLPs consist of locating hubs on a network so as to minimize the total flow cost (Contreras et al., 2011a).

Transportation applications of hub location models include air passenger travel, air freight travel, express shipments, large trucking systems, postal operations and rapid transit systems (Campbell & O'Kelly, 2012). Due to their multiple applications, beginning with the pioneering work of O'Kelly (1986), these problems have received an increasing attention in literature. Solution methods have been developed for several variants of HLPs, such as uncapacitated hub location, p-hub location, p-hub center, and hub covering (Campbell & O'Kelly, 2012). For each of these classes of problems, there exist several variants arising from various assumptions, such as hub capacities or a specific topological structure to the hub-

and-spoke network. The reader is referred to Campbell and O'Kelly (2012) and Alumur and Kara (2008) and a recent survey on HLPs by Zanjirani Farahani et al. (2014) and Contreras (2015).

Among all these classes of HLPs, the p-hub median problem and its variants have been comprehensively studied and addressed in recent research on HLPs. The p-hub median problem is a fundamental discrete hub facility location and hub network design problem analogous to the p-median problem (Campbell 1996). The solution of p-hub median problems is a (connected) network in which $p(p-1)/2$ (undirected) hub arcs connect all hub pairs, and the remaining access arcs connect nodes to hubs. A growing body of research has addressed both single allocation hub median problems, in which each non-hub node is incident with exactly one access arc, and multiple allocation hub median problems, in which non-hub nodes may be incident with more than one access arc(as shown in Figure 2).

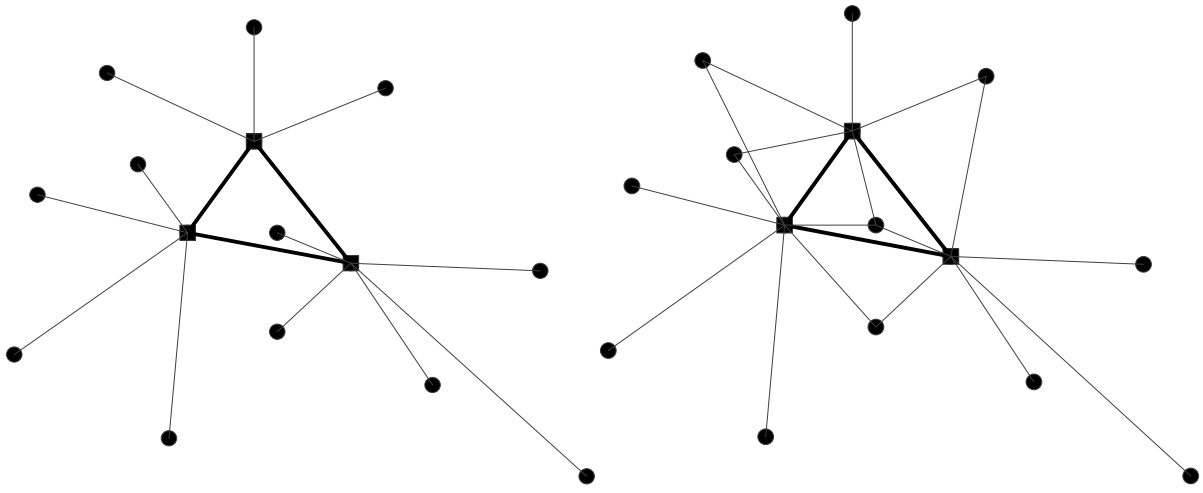

Figure 2.A schematic comparison of Multiple Allocation and Single Allocation Hub Networks. Single Allocation Hub-and-spoke Network (Left), Multiple Allocation Hub-and-spoke Network(right).

Several variants of p-Hub Location Problems (p-HLPs) have been studied in literature. The difference comes from single and multiple allocations of non-hub

nodes to hubs and also the capacity of hub nodes if considered. One of the assumptions in hub location problems is to encourage concentration of flows between all hubs by providing a discounted unit flow cost; the discount factor α, $0 < \alpha < 1$ that has been applied to the transportation cost of the flows between any pair of hubs.

HLPs are among challenging classes of NP-Hard combinatorial optimization problems combining decisions on location and network design. This is a difficult class of problems in operation research and many people have developed different formulations and solution algorithms.

Sohn and Park (1998) prove that the single allocation problem is NP-hard for three or more hubs. As even the most basic hub location problems are NP-hard, it is hardly surprising that many of the solution techniques suggested for the various models are heuristics. This includes several metaheuristics such as Greedy heuristics, Tabu Search, Simulated Annealing, Genetic Algorithm and GRASP (Greedy Randomized Adaptive Search Procedures). The effectiveness of these methods depends on their ability to avoid entrapment at local optimality, and exploit the basic structure of the problem, such as a network or a natural ordering among its components. Building on these notions, various heuristic search techniques have been developed that have demonstrably improved our ability to obtain good solutions to difficult combinatorial optimization problems. Consequently, these problems are challenging and many people have been focusing on the development of an efficient formulation both exact and approximate solution methodology. In this thesis we will focus and study the multiple allocation uncapacitated p-hub location problem.

The contribution of this thesis is in the development of a GRASP metaheuristic for finding good quality feasible solutions for this specific problem. In addition, we compare different constructive algorithms that can be later embedded into local improvement methods and report the computational result. Meaning that, we compare the results on the performance of each constructive heuristic algorithm in order to measure the efficiency of the proposed Greedy function.

We have four constructive algorithms; they differentiate the fact that some of them have preprocessing steps which are removing some candidate nodes that seem not beneficial. Eventually, the computational results prove the efficiency of the algorithm and reach optimal solutions for most of the instances. Given this, the prime focus is to obtain an optimal solution to all considered instances (up to 250 nodes or more) within reasonable CPU times.

This thesis is organized as follows. In Chapter 2, a literature review of HLPs is presented. In Chapter 3, we define and describe the problem of this thesis and present a mathematical formulation and assumptions considered for multiple allocation uncapacitated p-hub location problem. In Chapter 4, solution algorithms for this problem are presented. We introduce a local improvement technique for our heuristic algorithms, and we next present our GRASP metaheuristic. The results of computational experiments and an analysis of the proposed GRASP metaheuristic are presented in Chapter 5. Finally, in Chapter 6, concluding remarks and future research avenues are provided.

CHAPTER 2: Preliminaries

The HLPs consist of locating a set of hubs and assigning a of the origins-destinations pairs to the selected hubs. HLPs were originally introduced by O'Kelly (1986a) together with real life examples. These systems serve demand for travel or communication between many origins and destinations, where economies of scale exist in the cost for such travel or communications (Campbell et al. 2002). Hub and spoke networks rather than routing every demand with a direct link from its origin to destination points, use a set of fewer links to route these demands. The use of fewer links concentrates flows and allows economies of scale to be exploited.

A vast literature has focused on developing good formulations for these classes of HLPs. To represent a wide range of HLPs, operations research practitioners have developed a number of mathematical programming formulations and models. Different objective functions have been proposed to make such models amenable to numerous applications. The first integer programming formulation proposed for HLPs is a quadratic model (O'Kelly, 1987). Quite a while, the literature focused on the linearization of the quadratic model proposed (Aykin, 1995; Campbell, 1996; Ernst and Krishnamoorthy 1996; O'Kelly et al. 1996b; Skorin-Kapov et al. 1996). In addition to the integer programming formulation, two heuristic approaches are presented in O'Kelly (1987). The first one, which is also called as 'nearest hub allocation rule', basically investigates allocating each node to the nearest hub while the later investigates the idea of assigning each non-hub node to either its first or second nearest hub. These heuristics are generally quite effective in providing good upper bounds during complete enumeration of hub locations.

We next review some relevant literature on hub location problems classified into four groups: hub location problem with fixed costs, p-hub center problem, hub covering problem and p-hub median problem.

2.1. p-Hub Median Problem

The p-hub Location problem is a fundamental discrete hub facility location and hub network design problem (Campbell 1996). The selection of p hub nodes and assigning the remaining origin/destination nodes to these hubs are the most challenging decisions in p-hub median problems where the objective to p-hub median problems is to minimize the total transportation cost of routing commodities through the network. Several variants of the p-HLPs arise from single and multiple allocations and capacity constraints. The solution of a p-hub median problem is a (connected) network in which $p(p-1)/2$ (undirected) hub arcs connect all hub pairs, and the remaining access arcs connect nodes to hubs. A growing body of research has addressed both single allocation hub median problems, in which each non-hub node is incident with exactly one access arc, and multiple allocation hub median problems in which non-hub nodes may be incident with more than one access arc (Campbell et al., 2002).

Campbell (1994b) provides the first linear integer programming formulation for the p-hub median problem together with mathematical formulations for the hub location problem with fixed costs, the p-hub center and the hub covering problem. O'Kelly (1986b) provides the first quadratic integer programming formulation for the p-hub median problems. Skorin-Kapov et al. (1996) present new formulations for both single and multiple allocation p-hub median problems with tighter LP relaxations. Similar to all classes of hub location problems, one of the assumptions

in p-hub location problems is also to encourage concentration of flows between all hubs by providing a discounted unit flow cost; the discount factor α, $0 < \alpha < 1$ that has been applied to the transportation cost of the flows between any pair of hubs.

Several efficient solution methodologies have been proposed for both single and multiple allocation p-hub location problems. The first approximate algorithm for the p-hub median problem was proposed by O'Kelly (1986b). He develops an enumerative based heuristic that searches all possibilities of *p* hub selection and uses nearest hub for assignment of non-hubs to hub nodes. Klincewicz (1991) develops exchange heuristics for the single allocation p-hub median problem. These heuristics are compared with a clustering heuristic and heuristics developed in O'Kelly (1986b).

There are also several other heuristics such as tabu search and for single and multiple allocation p-HLPs that outperform the earlier heuristics Klincewicz (1992). Skorin-Kapov and Skorin-Kapov (1994) propose another tabu search that outperforms the previous heuristics in terms of the incumbent value but is weaker in terms of computational time. Several other heuristics have also been developed to obtain good quality solutions to larger instances of p-hub median problems. Ernst and Krishnamoorthy (1996) develop a simulated annealing metaheuristic for p-HLPs that outperforms the tabu search presented in Skorin-Kapov and Skorin-Kapov (1994). This work is one of the earliest successful attempts in obtaining good quality solutions to p-hub median problems. Later, Pirkul and Schilling (1998) use Lagrangian relaxation for obtaining better lower bounds to p-hub median problems. They present a subgradient algorithm to obtain lower bounds and good quality feasible solutions.

Note that in multiple allocation classical HLPs, the allocation decisions are trivial once the location of hubs are fixed. That is, each pair of nodes sends flows via the shortest path in the given hub network. This idea was first presented and employed in Ernst and Krishnamoorthy (1998a). Sasaki et al. (1999) consider the 1-stop multiple allocation p-hub median problem which is a special case of the p-HLPs where they allow using at most one hub in routing flows from origin to destination points. They formulate the model as a p-hub median problem and propose two solution algorithms, a branch and bound and a greedy type algorithm. Milanovic (2010) propose a new evolutionary based algorithm for uncapacitated multiple allocation p-HLPs. In another work, Garcia et al. (2012) propose new formulations and a branch and cut algorithm for this problem. Kratica (2013) develops an electromagnetism-like metaheuristic for the uncapacitated multiple allocation p-HLP. Some other authors also study the allocation strategies in HLPs (Yaman, 2011). Peiro et al. (2014) study r-allocation p-HLPs where the number of hub nodes to be assigned to each nonhub does not exceed r in routing of commodities.

2.2. Hub Location Problem with Fixed Costs

The p-hub median problem aims to minimize only the transportation costs and does not take fixed cost of opening hub facilities into consideration. However, these fixed setup costs might be included in the objective function by defining a decision variable that represents the decision of opening hub facilities. In p-hub median problems, the number of hubs to open is fixed and given. In the hub location problems with fixed costs, however, the number of hubs to be established is not specified in advance and is a decision to the model. Therefore, the model will decide the number of hubs to open, which nodes to choose as hubs, and the

allocation of the non-hub nodes to the selected hubs such that the total transportation and setup cost is minimized. O'Kelly (1992) introduces the single allocation version of this problem to the literature and develops a quadratic integer programming formulation. As previously mentioned, Campbell (1994b) provides the first linear programming formulations for both single and multiple allocation types of the problem as well as capacitated and uncapacitated versions. In the capacitated version of this problem, the capacity restrictions are on the inbounding flow carried by each hub.

2.3. p-Hub Center Problem

The p-hub center problem is modeled as minimax optimization problem and its objective might be either minimizing the maximum cost or the maximum travelling time between any origin destination pair. The center problems have important applications such as locating emergency service facilities and vehicles. When the objective of the p-hub center problem might be to minimize the maximum travelling time between each origin destination pair, the decisions of the problem are the locations of p hubs and the assignment of other nodes to these hubs so that the maximum travelling time between origin-destination pairs is minimized. The first formulation for the p-hub center problem is proposed by Campbell (1994b). Although the original formulation is quadratic, a linearization of this model is also presented in the paper. Kara and Tansel (2000) study the p-hub center problem and provide three different linearization of the formulation in Campbell (1994b). They include a new formulation for the p-hub center problem and the linearization of this formulation outperforms all the linearization of the previous model of Campbell (1994b).

There are only a few exact and approximate algorithms for p-hub center problem among which are tabu search for the single case Pamuk and Sipil (2001) together with a demonstrative computational experiment. In a subsequent work, Ernst et al. (2009) present mixed integer programming formulations for problems with single and multiple allocation variants and propose a branch-and-bound approach for solving the multiple allocation case. Kara and Tansel (2001) analyze an interesting aspect of p-hub center problem by considering some operational-level constraints, in a plane scheduling set, where planes cannot leave until all planes arriving at the hub have arrived and call it latest arrival hub location problem. This work was later questioned by Wagner (2004) where he shows this "new" model is the same as the classical model that ignores the transient times (or waiting times). Later, Yaman et al. (2007) extend the latest arrival hub location problem by allowing stopovers between non-hubs and hubs meaning a route from a non-hub to a hub may include a visit to another non-hub node. Campbell et al. (2007) considers the single and multiple allocation of p-hub center problem and illustrates that several special cases of these problems can be solve in polynomial time.

2.4. Hub Covering Problem

In covering problems, some cost or time parameters are restricted to a specified value due to the resource limitations or for customer satisfaction (Campbell et al., 2002). Some variations of the hub covering problem might be minimizing the total cost under the restriction of the travelling time for any origin-destination pair, or minimizing the number of facilities opened by restricting the travelling cost of each origin-destination pair. The objective of the hub covering problem might be to minimize the number of hubs to open so that the total transportation cost is within

a specific value. Then the model needs to decide the number and location of the hubs together with the allocation of non-hub nodes to these selected hubs. Moreover, the objective of the hub covering problem might also be to minimize the total cost as well. When we consider the cargo delivery systems, the firms might want to establish a network structure that will enable service for each origin-destination pair in certain time period, say 24 hours, with minimum total cost (cost of transportation and operating hubs).The first Mixed Integer Linear Programming formulation (MIP) for the hub covering problem is developed by Campbell (1994b), which mainly studies the hub set-covering problem and the maximal hub-covering problem.

Campbell (1994b) defines coverage based on several criteria. Let i and j to be origin-destination points and k and m to be hubs. By his proposal, origin-destination pair is covered if (i) the cost of routing a commodity from i to j through k and m does not exceed a pre-specified value, (ii) the cost of each link in the described path does not exceed a pre-specified value, and (iii) each access links (i,k) and (m,j) are less than a separate specific values. After presentation of hub set and hub maximal covering problem with single and multiple allocation by Campbell (1994b), Kara and Tansel (2003) and Wagner (2008) provide MIP formulation to p-hub covering problem.

The first category of hub covering problems is hub set covering problem. This problem is very similar to the well-known set covering problem but only differs in terms of the hub network topology. This problem is formulated like p-hub median problems while the number of hubs to be located is not known in advance. In fact, the hub set-covering problem tries to locate hubs to cover all demand such that the set-up cost of hub facilities is minimized. The maximal hub-covering problem,

however, maximizes the demand covered with a given number of hubs to locate. That is, it strives to locate a given number of facilities to best meet the (weighted) demands. Unlike first version of hub covering problems, the number of hubs to be located is known to maximal hub-covering model and the objective is again to maximize transportation demand covered. Note also that, in the hub set covering formulation, because all of the demands must be met (covered) regardless, the relative weight of the demands generated by the existing facilities are inconsequential, whereas in the maximal hub covering objective some existing demands may be left unmet (uncovered), meaning the designed network might not be, and most often is not, able to provide service to all demand points. This could be one of the weaknesses of this model and might also be one of the reasons this model have not attracted a significant attention in literature.

2.5. Metaheuristic Algorithms

Most of the discrete optimization problems cannot be solved to optimality for realistically sized instances. In computer science, metaheuristic designates a computational method that optimizes a problem by iteratively trying to improve a candidate solution with respect to a given measure of quality namely the objective function value. Metaheuristics do not guarantee to find an optimal solution. The effectiveness of these methods depends upon their ability to adapt to a particular realization, avoid entrapment at local optima, and exploit the basic structure of the problem, such as a network or a natural ordering among its components. Building on these notions, various heuristic search techniques have been developed that have demonstrably improved our ability to obtain good solutions to difficult

combinatorial optimization problems. Among the most promising and successful algorithms are Genetic Algorithm (GA), Simulated Annealing (SA), Tabu Search (TS), Scatter Search (SS), Path Relinking (PR), and Greedy Randomized Adaptive Search Procedure (GRASP).

A Greedy Randomized Adaptive Search Procedure (GRASP) is a multi-start process in which each iteration includes two phases, (i) a greedy randomized construction phase and (ii) a local search procedure. During the construction, a feasible solution is generated, and in the local search procedure provides a local optimum for the neighborhood of the constructed solution (Feo and Resende, 1995). Finally, at each iteration, the best overall solution (incumbent solution) will be kept if it improves objective function (as shown in following pseudo code).

Algorithm 1. Grasp

while ($k \leq maxitr$ *and other stopping criteria is not satisfied*){
    Construct a feasible solution
    Improve the obtained solution
}***endwhile***

This procedure is repeated for *maxitr* times (*maxitr* is large enough). Repeated applications of a construction procedure yield diverse starting solutions for the local search. In the construction phase, a feasible solution is iteratively constructed, by adding one element at a time. The basic GRASP construction phase is similar to the semi-greedy heuristic proposed independently by Hart and Shogan (1987).

The choice of the next element to be added, at each construction phase, is determined by ordering all candidate elements in a candidate list C with respect to a greedy function $g: C \rightarrow R$. The greedy function g measures the benefit of

selecting that particular element. The GRASP heuristic is called adaptive for the benefits associated with every element that is updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. One of the main drawbacks of the deterministic greedy procedure is that it suffers from lack of diversity at construction phase as its tends to construct feasible solutions that yield one local optimal. To avoid this local optimality, a random element has been added to enable larger area of feasible region to be explored.

The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list that is not necessarily the top candidate. The list of best candidates is called the Restricted Candidate List (*RCL)*. The *RCL* is chosen based on the value of ω at each iteration. This selection technique allows different solutions to be obtained at each iteration of GRASP algorithm. The parameter ω controls the amount of greediness and randomness in the algorithm where $\omega = 0$ corresponds to deterministic greedy construction procedure, while $\omega = 1$ produces random construction.

CHAPTER 3: Uncapacitated Multiple Allocation p-Hub Location Problem

In this chapter we will provide the formal definition of the problem. Moreover, we provide the mathematical formulation for the problem.

3.1. Problem definition

Let $G = (N, A)$ to be an undirected graph where $N = \{1,2, \dots, n\}$ corresponds to set of nodes and $A = \{1, 2, \dots, a\}$ represents the set of arcs in the network. Considering the general setup of the hub location problems, *n* points (origins and destinations), the flow W_{ij} and the per unit transportation cost d_{ij} from origin i to destination j, and the discount factor α for hub-to-hub transportation are given. The unit transportation cost from origin i to destination j via hubs k and l is denoted by $(d_{ik} + \alpha d_{km} + d_{mj})$. Then the total transportation cost from origin i to destination j via hubs k and m is:

$$F_{ijkm} = W_{ij} \times (d_{ik} + \alpha d_{km} + d_{jm}).$$

The multiple allocation p-hub median problem consists of locating p hubs. In this problem, the hub level network is complete and that no non-hub nodes cannot be connected directly.

3.2. MIP Formulation

We next present a MIP formulation for the multiple allocation uncapacitated p-hub problem. We introduce our first set of binary routing variables X_{ijkm} to be 1 if and only if flow from node i to node j is routed by hubs k and m and the second binary location variables Z_k to be 1 if and only if node $k \in$ N is a hub node. Using these

binary decision variables, the Multiple Allocation p-Hub Location Problem can be formulated as (Campbell, 1994b):

$$\textbf{minimize} \qquad \sum_{i\epsilon N}\sum_{j\epsilon N}\sum_{k\epsilon N}\sum_{m\epsilon N} F_{ijkm}X_{ijkm} \tag{1}$$

$$subject\ to: \quad \sum_{k\epsilon N} Z_k = p \tag{2}$$

$$\sum_{k\epsilon N}\sum_{m\epsilon N} X_{ijkm} = 1 \quad \forall \quad i,j \ \in N \tag{3}$$

$$\sum_{m\epsilon N} X_{ijkm} \leq Z_k \quad \forall \quad i,j,k \ \in N \tag{4}$$

$$\sum_{k\epsilon N} X_{ijkm} \leq Z_m \quad \forall \quad i,j,m \ \in N \tag{5}$$

$$X_{ijkm} \geq 0 \quad \forall \quad i,j,k,m \ \in N \tag{6}$$

$$Z_k \in \{0,1\} \quad \forall \ k \ \in N \tag{7}$$

The objective (1) is to minimize the total cost of routing flows between origins and destinations via hub nodes and/or hub arcs. Constraints (2) imposes that there are p hub nodes allowed to be established in the network. Constraint (3) is to guarantee a path through at least one or at most two hubs. Constraint (4) and (5) are making sure that no flow is routed via non hub nodes and non-hub arcs. Finally, constraints (6) and (7) are non-negativity and binary constraints. The X_{ijkm} variables, even though have a binary interpretation, can be defined as nonnegative continuous variables and the formulation will enforce them to take a binary value. This is a consequence of the fact that there are no capacity constraints on the hubs.

p-Hub median problems belong to the class of NP-Hard combinatorial optimization problems combining location and network design decisions. The computational hurdle imposed by complex hub location formulations has limited most research in this area to small to medium size problems. As even the most basic hub median problems are NP-hard, many of the solution techniques suggested for HLPs are heuristics.

CHAPTER 4: Solution Algorithms for MApHLP

4.1 Constructive Heuristics

In this chapter we present 4 different constructive heuristic algorithms that will provide an initial feasible solution for this problem that later would be improved by using local improvement technics. Each of these algorithms is based on different procedure. One of these solutions is deterministic, two of them are randomized and one of them is adaptive randomized.

4.1.1 Fully Randomized

In this algorithm we observe the results by choosing candidates in fully randomized form. To do so, all hub candidates were chosen completely random. Meaning p nodes are chosen randomly and they assigned as hubs. At this stage for every i as an origin node and j as a destination node, we have to find the best routing. The rout from i has to pass either one hub or maximum two hubs and then links to j. The reason that the rout has to pass maximum two nodes is based on the triangular inequality that would be satisfied by having our rout passing maximum two hub nodes. Figure 3 illustrates the above fact where i, j are origin and destination nodes, and k, l, m are hub nodes.

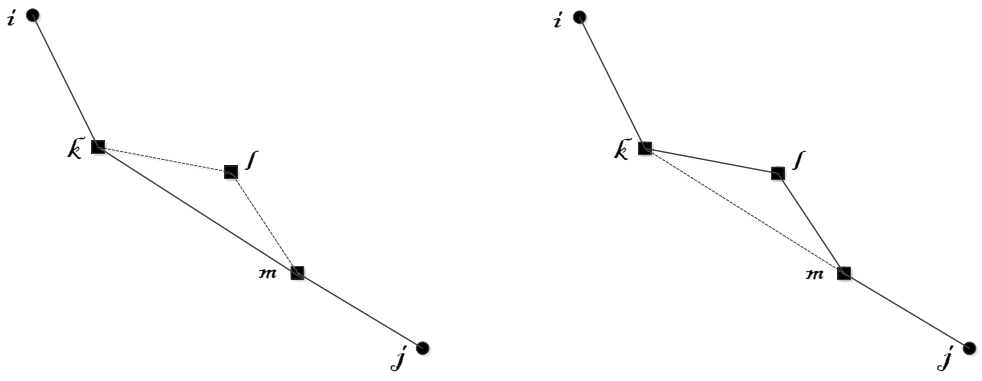

Figure 3. Straight vs. triangle connection

Consequently, connections are made and the shortest path is chosen for every i and j. Finally, the objective function is calculated.

Let $\mathrm{H_s}$ represent the set of open hubs, $\mathrm{E_s}$denote the set of open hub-arcs and $\mathrm{S} = (\mathrm{H_s}, \mathrm{E_s})$. We define V as the set of nodes that is equal to N and updates according to the algorithm criteria in every iteration. Using these notations we present the pseudo code for fully randomized algorithm as follow:

Algorithm 2. Construction phase of Fully Randomized heuristic to MApHLP

Initialization
    $\boldsymbol{H_s} = \emptyset$, $RCL = \emptyset$, *UB=0, V=N;*
while($|H_s| \neq p$ ***) do***
    $RCL = V$
    Select randomly $k \in RCL$
    $H_s = H_s \cup \{k\}$
    $V = V - \{k\}$
end while
if ($|H_s| = p$)
    Construct E_s
    Solve routing sub problem
end if
Evaluate objective function valueUB

4.1.2 Greedy Deterministic

In order to proceed with Greedy Deterministic algorithm, we should define our greedy function. The greedy function is based on the amount of flow that can be concentrated at every potential hub node which is originated (or with destination) from a set of nodes that are within a predetermined radius from the hub node. Campbell et al. (2005) address the importance of the demand pattern and spatial distribution of the hub nodes in the design of hub-and-spoke network obtained from p-hub median problems. Given this, one can realize that the spatial distribution of nodes in a hub and spoke network should be of major consideration.

We employ the standard deviation with respect to the distance of direct connections between all nodes. In short, given direct connection distances between each pair of nodes we calculate the associated standard deviation and use its value as the radius of our circular sector. In other words, each node i is a center of a circle. The radius of the circle is equal to STD and the area created by the circle is called circular sector.

Let $D = \{d_{ij} | \, i, j \in N, i < j\}$ represent the set of distance values between each pair of nodes where $|D| = \frac{|N|(|N|-1)}{2}$. Let STD denote the standard deviation of the distance parameters in set D. Following shows how the standard deviation, STD, is calculated:

d_{ij} = disdtance between node i and node j.

$\bar{d} = \frac{2\sum_{i<j} d_{ij}}{n(n-1)} \quad for\ i\ and\ j\ \in N, i < j$

STD = $\sqrt{\frac{\sum_{i<j}(d_{ij} - \bar{d})^2}{n-1}} \quad for\ i\ and\ j\ \in N, i < j$

Let C_i to be the set of nodes that are within a distance of $i \in N$, $C_i = \{j \in N | d_{ij} \leq STD\}$ and $g(i)$to be greedy function the total amount of flow originated from or arrived to some nodes $j \in C_i$. If we consider $\sum W_{ij}$as total flow amount for node i, then we define greedy function as follow:

$$g(i) = \sum W_{ij} \quad for\ j\ \in C_i$$

Through the construction phase of Greedy Deterministic, for each node $i \in N$, we calculate the total flow in a particular circle with radius of STD. A schematic illustration of the above described is shown in Figure 4. Density of nodes& flow accumulation in the section is the main idea to look for a potential hub

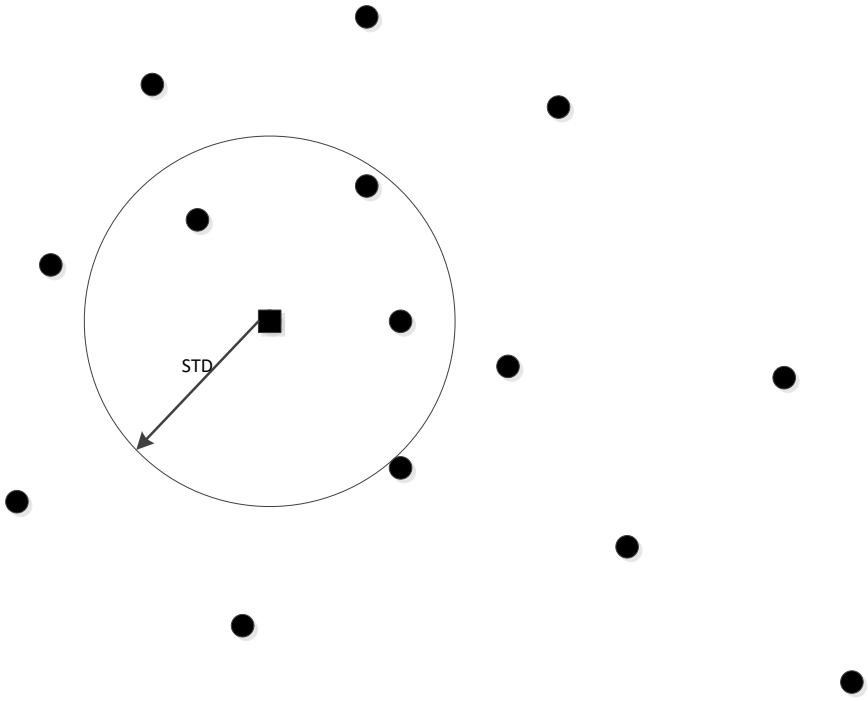

Figure 4. Density of nodes& flow accumulation in the section is the main idea to look for a potential hub

Algorithm 3 describes the pseudo code for Greedy Deterministic algorithm:

Algorithm 3. Construction phase of Greedy Deterministic heuristic to MApHLP

Initialization
$\boldsymbol{H_s} = \emptyset,\ RCL = \emptyset$, UB=0, V=N;
while($|H_s| \neq p$ ***) do***
$Evaluate\ g(i) for\ all\ i \in V$
$RCL = \{i \in V | \ g(i) = g_{max}\ \}$
$Select\ k\ \in RCL$
$H_s = H_s\ \cup \{k\}$
$V = V - \{k\}$
end while
if ($|H_s| = p$***)***
Construct E_s
Solve routing sub problem
end if
Evaluate objective function valueUB

Where $g_{max} = \max\{g(i) | \ i\ \in V\ \}$.

4.1.3 Greedy Randomized

In this part we provide Greedy Randomized algorithm. In this algorithm we choose the hub node randomly and we apply the greedy function to it. It is apparent to the first introduced algorithm, fully randomize, with the difference of removing some candidate nodes that seem not beneficial. Here, greedy function is responsible for removing those candidates. Meaning a hub candidate is chosen completely random. For this particular candidate i, we evaluate $g(i)$ in order to identify C_i that are associated nodes in the circular sector. We continue this process until p nodes are chosen. Like previous algorithms, when the p-hubs are defined, connections would be made and the objective function is calculated.

There is one more thing in this algorithm that we should take into account. In some cases, it could be possible that the cardinality of the complementary of the set C_i is a null set while we have not reach the quantity of the p. Meaning we need to choose another non-hub node and make it as a hub node but there isn't any available node to choose from. To prevent happening this scenario, we verify at every iteration if the remaining non hub node in V are equal or greater than $p - |H_s|$. If this case happened, we have to stop removing associated node in the circular sector and denote the remaining node as a candidate.

Algorithm 4. Construction phase of Greedy Randomized to MApHLP

Initialization
    $\boldsymbol{H_s} = \emptyset,\ RCL = \emptyset$, *UB=0, V=N;*
while($|H_s| \neq p$ *or* $V \neq \emptyset$ ***) do***
    $RCL = V$
    Select randomly $k \in RCL$
    $H_s = H_s \cup \{k\}$
    $V = V - C_k$
end while
if ($|H_s| = p$***)***
    Construct E_s
    Solve routing sub problem

end if
Evaluate objective function valueUB

4.1.4 Greedy Randomized adaptive Search

We now present our Greedy Randomized Adaptive Search heuristics for the Uncapacitated Multiple Allocation p-Hub Location Problem. We have already defined a greedy function based on the amount of flow in the circular sector.

Let $g_{min} = \min \{g(i) | \, i \in V\}$, the restricted candidate list can then be stated as following:

$$RCL = \{i \in V | g(i) \geq g_{min} + \omega(g_{max} - g_{min})\}$$

where $\omega \in [0,1]$ is the probabilistic parameter that controls the level of greediness of randomness used during the constructive phase.

After sorting related flow amounts belonging the nodes in the sector, the restricted candidate list will be generated based on greedy randomness parameter, $\omega \in [0,1]$, and then we randomly select one node from the defined candidate list. In next step, we remove all the nodes in the circular sector associated with the selected hub node and then calculate the greedy function for the remained nodes. This process will be repeated for p times until p hub nodes are selected (as shown in

Figure 5-Figure 7). Similar to what we had in the section 4.1.3 for each iteration, it should be verified that the remaining nodes are equal or greater than $p - |H_s|$.

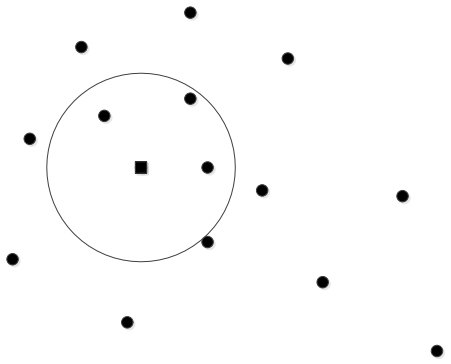

Figure 5. Elimination of associated nodes and assigning a new hub

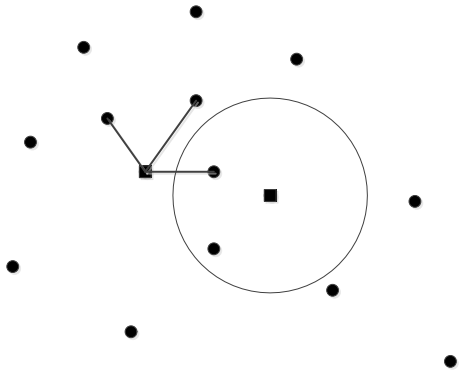

Figure 6. Assigning new hub

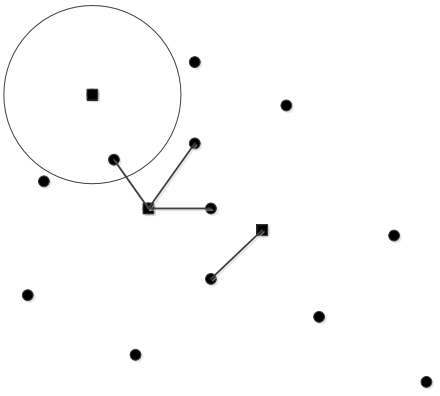

Figure 7. Iterations continue for p times

The final step in the construction phase will be assignment of each non-hub node based on the objective function. That is to find the best route originated from every single nun hub node to all distant node passing at most two hub nodes. The pseudo code of our construction phase is presented in Algorithm 5.

Algorithm 5. Construction phase of Greedy Randomized adaptive Search heuristic to MApHLP

Initialization
    $\boldsymbol{H_s} = \emptyset,\ RCL = \emptyset$, *UB=0, V=N;*
while($|H_s| \neq p$ *or* $V \neq \emptyset$ ***) do***
    $Evaluate\ g(i) for\ all\ i \in V$
    $RCL = \{i \in V |\ g(i) \geq g_{min} + \omega(g_{max}\text{-}g_{min})\}$
    Select randomly $k \in RCL$
    $H_s = H_s \cup \{k\}$
    $V = V - C_k$
end while
if ($|H_s| = p$***)***
    Construct E_s
    Solve routing sub problem
end if
Evaluate objective function valueUB

4.2 Local Improvement Method

The local search phase tries to improve an initial feasible solution by means of opening and closing new hubs and reassigning non-hub nodes to the hub facilities. This means that after the construction phase, we close an open hub node and replace it by a closed hub node. We then reassign some non-hub nodes to the new hub to construct a feasible solution. If one solution improves the network transportation cost, it would be updated as the best solution found and the algorithm continues to further find better solution in the next iterations. Once this procedure is examined, given all possible substitutions, the greedy parameters will change and the construction phase will find another starting feasible solution.

Our node-shift neighborhood search closes an open hub node $i \in H_s$ and its joint hub arcs $A_i = \{e | e \in E_s, i \in e\}$ and opens a new non-hub node $j \in N \backslash H_s$ and the hub arc set $A_j = \{e = (e1, e2) | e \in E \backslash E_s, j \in e, |\{e1, e2\} \cap H_s| = 1\}$.

Our node-shift neighborhood can be described as:

$$N_{node-shift}(S) = \{S' = (H'_s, E'_s) | H'_s = H_s \cup \{j\} \backslash \{i\}, \exists! i \in H_s, \exists! j \notin H_s\}.$$

We explore these neighborhoods by using first improvement strategy that arbitrary selecting a current hub node and closing it, and selecting an open hub node and closing it.

The neighborhood search algorithm is presented in Algorithm 6. Local Search

Algorithm 6. Local Search

Initialization
    terminate = false
while($terminate = false$***) do***
    explore $N_{node-shift}$
    if*(solution not improved in* $N_{node-shift}$***) do***
        terminate = true
    end if
end while

In chapter 5 we present the computational results of our constructive algorithms with the local search.

4.3 GRASP

Our GRASP metaheuristic is a multi-start algorithm. We start with setting the value of $\omega = 0$ and iteratively increasing its value by 0.05. The value of ω is set to 0 when it reaches its highest value. This procedure stops after 100 iterations. The overall scheme for our Grasp Algorithm is presented in Algorithm 7.

Algorithm 7. GRASP metaheuristic to MApHLP

Initialization
    $\omega = 0, \quad t = 0$
while($t \leq maxitr$***) do***
    Call Construction phase
    Call Local search
    $\omega = \omega + 0.05$
    if *(*$\omega = 1$*)* ***do***
        $\omega = 0$
    end if
end while

CHAPTER 5: Computational Results

In this chapter, we detail our computational results. We firstly present our data, software and hardware details and then provide a computational experiments to demonstrate the efficiency of our GRASP metaheuristic.

The data set used in this study is driven from Australian postal data that is available at the OR library which can be downloaded at *mscmga.ms.ic.ac.uk/jeb/orlib/phubinfo.html*. This data set is frequently used in hub location literature and consists of the Euclidean distances d_{ij} and positive flow w_{ij} between each pair of nodes $(i, j) \in A$ in the network. The algorithms were coded in $\mathrm{C}++$, and run on PC with a Pentium® Dual-Core CPU E5500 processor running at a 2.80 GHz with 4 GB of RAM under Windows 7 environment.

We run a series of computational experiments on a benchmark instances ranging from 20 to 200 nodes. The data set consists of *six* sets of instances each of size $|N| = 20,\ 50, 75, 100, 150$ and 200. The value of p is set to 3, 5 and 8. The discount factor α is also chosen as $\alpha = 0.2, 0.5$ and 0.8. The number of different combinations results in a total of 54 instances.

The remaining of this chapter is organized as follows. We first, for comparison purposes, detail the computational results from an adaptation Benders decomposition proposed for p-Hub median problem in literature (Contreras et al. 2011c) and then detail the performance of fully randomized, greedy deterministic and greedy randomized, and finally the computational performance of our GRASP metaheuristic. The percentage deviation of obtained upperbound from optimal solution obtained by Benders decomposition algorithm is calculated as following:

$$100(Upperbound - Lowerbound)/Upperbound$$

To verify the efficiency of proposed algorithms, the global optimal results to p-median problem obtained with a Benders decomposition algorithm are shown in Tables 1 to 3**.** These results are later discussed in Section 5.2, together with some comparisons on our GRASP metaheuristic results. Through the presentation of our computational results, second, third and fourth columns in tables represent number of nodes in the network, number of hubs and the value of discount factor α respectivly. The fifth column represents the value of optimal solution to these instances in Table 1-3. And finally the last column reports the CPU time taken to solve these instances to optimality.

Table 1. Global optimal solutios to MApHLP (small size instances)

#	*N*	*P*	*α*	*Global Optimal (BD)*	*Time*
1	20	3	0.2	57142.47	0.34
2	20	3	0.5	64754.02	0.17
3	20	3	0.8	68857.72	0.25
4	20	5	0.2	44506.65	1.19
5	20	5	0.5	55097.16	1.04
6	20	5	0.8	61936.78	0.55
7	20	8	0.2	32740.84	4.48
8	20	8	0.5	45400.03	2.54
9	20	8	0.8	55675.66	1.33
10	50	3	0.2	60920.18	1.65
11	50	3	0.5	67767.86	1.09
12	50	3	0.8	71770.74	0.86
13	50	5	0.2	49576.2	4.98
14	50	5	0.5	58625.15	4.04
15	50	5	0.8	65049.65	4.55
16	50	8	0.2	40946.25	362.5
17	50	8	0.5	52083.52	253.66
18	50	8	0.8	60572.55	115.13

Table 2. Global optimal solutios to MApHLP (medium size instances)

#	*N*	*P*	*α*	*Global Optimal (BD)*	*Time*
19	75	3	0.2	61863.71	5.27
20	75	3	0.5	68682.21	3.88
21	75	3	0.8	72533.67	3.07
22	75	5	0.2	50696.2	17.15
23	75	5	0.5	59714.46	15.24
24	75	5	0.8	65809.72	13.9
25	75	8	0.2	42425.19	448.89
26	75	8	0.5	53477.88	391.08
27	75	8	0.8	61641.59	370.32
28	100	3	0.2	61818.58	14.51
29	100	3	0.5	68562.71	10.52
30	100	3	0.8	72518.02	8.63
31	100	5	0.2	51157.74	58.58
32	100	5	0.5	60185.38	53.04
33	100	5	0.8	66177.81	38.8
34	100	8	0.2	42775.18	1540.6
35	100	8	0.5	53859.01	732.4
36	100	8	0.8	61961.79	537.19

Table 3. Global optimal solutios to MApHLP (large size instances)

#	*N*	*P*	*α*	*Global Optimal (BD)*	*Time*
37	150	3	0.2	61962.82	92.54
38	150	3	0.5	68854.63	52.43
39	150	3	0.8	72779.86	34.38
40	150	5	0.2	51444.08	650.49
41	150	5	0.5	60472.72	448.85
42	150	5	0.8	66407.3	188.51
43	150	8	0.2	43212.87	73755.4
44	150	8	0.5	54213.92	32312.68
45	150	8	0.8	62180.31	3414.24
46	200	3	0.2	62515.21	715.03
47	200	3	0.5	69334.46	281.89
48	200	3	0.8	73198.94	138.69
*49	200	5	0.2	52365.93	74041.89
*50	200	5	0.5	61353.77	72699.55
51	200	5	0.8	67015.23	3441.95
*52	200	8	0.2	43771.78	72346.23
*53	200	8	0.5	55069.05	73317.53
*54	200	8	0.8	62865.93	72136.69

(*: results for the instances marked with * are not proven global optimal)

5.2 A Comparison for Several Constructive Algorithms

In this section we compare the results obtained in algorithms presented in Chapter 4 with the optimal solutions from a Benders decomposition in terms of computational time in obtaining upper bounds with heuristics and how far their value functions are from the global optimal results. We use $\%\mathrm{DEV}$ that would give us the gap from the global optimal from Benders decomposition algorithm in section 5.1.1. The $\%\mathrm{DEV}$ is calculated as follows:

$$\%\mathrm{DEV} = \frac{(UB - OPT)}{OPT} \times 100$$

In order to have the name of solution algorithms in brief, we make S1, S2, S3 and S4. These names are associated as follows:

S1: "*fully randomized*"

S2: "*greedy deterministic*"

S3: "*greedy randomized*"

S4: "*greedy randomized adaptive*"

First we present the results from the constructive phase for each algorithm including CPU time of calculation and in next section we show the results that are given by the solutions with local search. This way we have good perspective of how effective are the constructive phases and assess the impact of the described neighborhood search.

Table 4. Algorithms Comparison (small size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1	S2	S3	S4	S1	S2	S3	S4
1	20	3	0.2	3.91%	12.40%	3.91%	3.91%	0.39	0.01	0.38	0.43
2	20	3	0.5	8.07%	10.13%	3.66%	4.07%	0.4	0	0.4	0.37
3	20	3	0.8	3.61%	9.30%	3.32%	3.32%	0.53	0	0.39	0.39
4	20	5	0.2	13.42%	20.86%	8.34%	8.01%	0.43	0.01	0.44	0.82
5	20	5	0.5	7.39%	15.75%	7.29%	6.62%	0.47	0.01	0.42	0.44
6	20	5	0.8	5.62%	13.29%	5.28%	7.15%	0.41	0	0.45	0.42
7	20	8	0.2	41.74%	41.74%	41.74%	41.74%	0.58	0.01	0.53	0.51
8	20	8	0.5	29.13%	29.13%	29.13%	29.13%	0.54	0.01	0.52	0.56
9	20	8	0.8	20.09%	20.09%	20.09%	20.09%	0.57	0.02	0.65	0.64
10	50	3	0.2	9.22%	25.28%	4.53%	4.53%	1.18	0.02	1.18	1.08
11	50	3	0.5	8.85%	21.69%	5.61%	6.48%	1.35	0.02	1.13	1.3
12	50	3	0.8	8.62%	18.85%	5.33%	5.27%	1.12	0.01	1.3	1.28
13	50	5	0.2	20.85%	33.37%	18.36%	15.13%	1.38	0.01	1.22	1.23
14	50	5	0.5	15.28%	28.65%	12.33%	14.93%	1.61	0.03	1.31	1.21
15	50	5	0.8	8.25%	23.79%	9.89%	6.76%	1.49	0.01	1.53	1.19
16	50	8	0.2	43.91%	43.91%	43.91%	43.91%	1.76	0.05	1.62	1.62
17	50	8	0.5	35.18%	35.18%	35.18%	35.18%	1.71	0.05	1.67	1.7
18	50	8	0.8	27.84%	27.84%	27.84%	27.84%	1.72	0.03	1.65	1.9
Average:				17.28%	23.96%	15.88%	15.78%	0.98	0.02	0.93	0.95

In Table 4, column S1 reports the performance of fully randomized algorithm for some small size instances. The smallest gap obtained is equal to 3.61% for N=20 at α=0.2 and p=8, while the largest gap that is 43.91% corresponds to N=50, p=8 and α = 0.2. Clear enough, this results are not promising even for small instances.

S2 details the performance of greedy deterministic randomized algorithm for our small size instances. The smallest gap obtained is equal to 9.30% for N=20 at p=3 and α=0.8 while the largest gap corresponds to $N = 50$, p=8 and α = 0.2.

S3 shows the performance of greedy deterministic algorithm for our small size instances. The smallest gap obtained is equal to 3.32% for N=20 at α=0.8 and p=3,

while the largest gap 43.91% corresponds to N=50, p=8 and α = 0.2. The largest time is also 108.94 (sec) for N=50 for p=8.

S4 reports the performance of GRASP constructive algorithm for small size instances. The smallest gap is 3.32% for N=20 at α=0.8 and p=3, and the largest gap is equal to 43.91% at N=50, p=8 and α = 0.2 where the average of %Dev shows 0.10% improvement compared with best result driven from the other greedy algorithms.

Table 5. Algorithms Comparison (medium size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1	S2	S3	S4	S1	S2	S3	S4
19	75	3	0.2	12.39%	22.75%	9.12%	9.12%	2.55	0.02	2.36	2.16
20	75	3	0.5	7.80%	18.36%	2.49%	7.38%	2.46	0.02	2.19	2.79
21	75	3	0.8	8.55%	16.20%	4.04%	2.41%	2.26	0.03	2.19	2.61
22	75	5	0.2	12.60%	31.79%	12.14%	13.32%	2.73	0.02	2.42	2.37
23	75	5	0.5	15.64%	25.56%	9.82%	11.94%	2.61	0.02	2.41	2.57
24	75	5	0.8	9.71%	21.66%	8.62%	10.78%	2.66	0.02	2.43	2.86
25	75	8	0.2	20.44%	41.66%	41.66%	41.66%	2.94	0.06	2.98	3.27
26	75	8	0.5	20.53%	32.87%	32.87%	32.87%	3.67	0.07	3	3.36
27	75	8	0.8	14.32%	26.55%	26.55%	26.55%	3.02	0.06	3.01	3.32
28	100	3	0.2	6.79%	24.95%	2.37%	3.43%	4.28	0.05	3.77	3.64
29	100	3	0.5	8.08%	21.35%	5.58%	4.42%	4.37	0.04	3.77	3.67
30	100	3	0.8	6.78%	18.71%	3.64%	2.90%	3.99	0.05	3.79	4.15
31	100	5	0.2	18.96%	32.81%	11.23%	12.44%	4.32	0.04	4.18	4.59
32	100	5	0.5	18.20%	27.88%	10.33%	15.68%	4.41	0.06	4.18	4.91
33	100	5	0.8	12.85%	23.87%	7.01%	8.52%	4.28	0.04	4.72	4.49
34	100	8	0.2	20.56%	41.56%	19.87%	14.95%	5.14	0.06	4.96	5.29
35	100	8	0.5	17.07%	34.12%	17.16%	16.33%	5.43	0.07	5.06	6.01
36	100	8	0.8	5.53%	28.04%	3.38%	4.01%	5.75	0.05	4.91	5.44
Average:				13.16%	26.05%	12.66%	13.26%	3.72	0.04	3.46	3.75

Table 5, column S1 details the performance of fully randomized algorithm for our medium size instances. The smallest gap obtained is equal to 5.53% for N=100 at

α=0.8 and p=8, while the largest gap corresponds to N=100, p=8 and α = 0.2. Note that, these results again, demonstrate the fully randomness trait in the behavior of algorithm as the quality of solution is not improved. The largest time is also 5.75 seconds for N=100 given 100 iterations of randomly selecting p=8 hubs.

The results presented in column S2 details the performance of greedy deterministic algorithm for our medium size instances. The smallest gap obtained is equal to 16.20% for N=75 at α=0.8 and p=3, while the largest gap 41.66% is obtained at N=75, p=8 and α = 0.2.

Column S3 reports the performance of greedy randomized algorithm for medium size instances. The smallest gap obtained is equal to 2.37% for N=100 at α=0.2 and p=3, while the largest gap 41.66% corresponds to N=75, p=8 and α = 0.2.

The performance of GRASP algorithm for medium size instances is shown in S4. The smallest gap is 2.41% for $N = 75$ at $\alpha = 0.8$ and $p = 3$, and the largest gap is equal to 41.66% at $N = 75$, $p = 8$ and $\alpha = 0.2$.

Table 6. Algorithms Comparison (large size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1	S2	S3	S4	S1	S2	S3	S4
37	150	3	0.2	10.04%	22.85%	7.31%	3.92%	9.15	0.08	8.74	9.1
38	150	3	0.5	10.72%	18.06%	2.65%	5.20%	10.72	0.08	8.47	9.35
39	150	3	0.8	6.28%	15.62%	2.64%	2.36%	9.18	0.11	9.13	8.13
40	150	5	0.2	16.70%	28.21%	11.73%	0.76%	11.21	0.08	9.87	9.23
41	150	5	0.5	15.19%	22.85%	10.99%	9.27%	11.57	0.09	9.5	9.47
42	150	5	0.8	9.79%	19.20%	8.02%	10.64%	10.39	0.1	9.63	9.72
43	150	8	0.2	25.57%	35.62%	21.46%	17.81%	12.82	0.1	11.9	11.2
44	150	8	0.5	21.87%	28.21%	13.84%	14.20%	13.21	0.1	11.65	12.48
45	150	8	0.8	15.53%	22.98%	10.87%	12.94%	11.96	0.1	14.85	11.72
46	200	3	0.2	14.89%	18.89%	3.18%	4.84%	19.86	0.14	16.65	16.51
47	200	3	0.5	10.73%	15.85%	3.70%	4.25%	20.16	0.16	18.5	15.47
48	200	3	0.8	8.74%	13.59%	5.15%	5.80%	17.47	0.15	16.85	16.03
49	200	5	0.2	16.84%	24.96%	13.92%	12.77%	21.43	0.2	19.83	17.16
50	200	5	0.5	14.25%	21.06%	11.21%	13.37%	20.13	0.18	18.68	18.91
51	200	5	0.8	12.93%	17.78%	7.65%	7.35%	20.05	0.16	18.45	17.02
52	200	8	0.2	21.78%	34.77%	18.64%	16.75%	23.83	0.2	21.51	20.52
53	200	8	0.5	19.24%	27.43%	16.02%	15.15%	24.55	0.19	21.26	22.17
54	200	8	0.8	15.00%	22.01%	12.00%	11.17%	23.35	0.22	21.54	19.84
Average:				14.78%	22.77%	10.05%	9.36%	16.17	0.14	14.83	14.11

Table 6 reports the performance of fully randomized algorithm for our large size instances.

For fully randomized algorithm that is S1, the smallest gap obtained is equal to 6.28% for $N = 150$ at α=0.8 and p=3, while the largest gap that is 25.57% corresponds to N=150, p=8 and α = 0.2.

The performance of greedy deterministic algorithm for our large size instances is written in S2. The smallest gap obtained is equal to 13.59 % for N=200 at α=0.8 and p=3, while the largest gap 34.77% corresponds to N=200, p=8 and α = 0.2. The largest time is also 271.22 (sec) for N=200 given 100 iterations of randomly selecting p=8 hubs.

The smallest gap for greedy randomized is equal to 2.64% and corresponds to N=150 at α=0.8 and p=3, while the largest gap 18.64% corresponds to N=200, p=8 and α = 0.2. The largest time is also 8904.21 (sec) for N=200 at p=8.

The results presented in column S4 details the performance of the GRASP algorithm for our large size instances. The smallest gap obtained is equal to 0.76% for N=150 at α=0.2 and p=5, while the largest gap 17.81% is obtained at N=150, p=8 and α = 0.2.

Observe that the quality of solutions obtained at construction phase is significantly improved in greedy randomized and GRASP. This shows that the greedy function in our study, namely consideration of standard deviation and the evaluation of candidate list based on the GRASP scheme can hold promise to better quality solutions.

5.3 A Comparison for Several Constructive with Local Search Algorithms

We now evaluate the performance of each algorithm when our neighborhood search is added to the search procedure of construction phases of each algorithm over the instances presented in section 5.2. Following 3 tables are the results in small, medium and large size instances.

Table 7. Algorithms Comparison with Local Search (small size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1L	S2L	S3L	S4L	S1L	S2L	S3L	S4L
1	20	3	0.2	0.00%	0.00%	0.00%	0.00%	0.97	0.07	0.99	0.93
2	20	3	0.5	0.00%	0.00%	0.00%	0.00%	1.03	0.06	0.97	1.05
3	20	3	0.8	0.00%	0.00%	0.00%	0.00%	1.08	0.03	0.97	1.11
4	20	5	0.2	0.00%	0.00%	0.00%	0.00%	2.42	0.07	2.65	2.95
5	20	5	0.5	0.00%	0.06%	0.00%	0.00%	2.52	0.11	3.22	2.37
6	20	5	0.8	0.00%	0.57%	0.00%	0.00%	2.66	0.08	2.4	2.78
7	20	8	0.2	0.00%	0.00%	0.00%	0.00%	7.59	0.23	7.06	7.56
8	20	8	0.5	0.00%	0.00%	0.00%	0.00%	8.56	0.24	8.89	7.55
9	20	8	0.8	0.00%	0.00%	0.00%	0.00%	8.1	0.24	7.61	7.49
10	50	3	0.2	0.00%	0.00%	0.00%	0.00%	8.3	0.3	11.54	8.2
11	50	3	0.5	0.00%	0.00%	0.00%	0.00%	10.3	0.26	11.45	8.35
12	50	3	0.8	0.00%	0.00%	0.00%	0.00%	9.06	0.23	9.39	8.2
13	50	5	0.2	0.00%	0.10%	0.00%	0.00%	32.17	0.84	35.02	29.77
14	50	5	0.5	0.00%	0.00%	0.00%	0.00%	45.7	0.84	31.38	29.3
15	50	5	0.8	0.00%	0.00%	0.00%	0.00%	32.45	0.83	34.96	32.29
16	50	8	0.2	0.00%	0.00%	0.00%	0.00%	107.54	3.17	108.94	108.04
17	50	8	0.5	0.00%	0.00%	0.00%	0.00%	107.59	3.19	106.88	137.17
18	50	8	0.8	0.00%	0.00%	0.00%	0.00%	106.91	3.12	106.77	120.78
Average:				0.00%	0.04%	0.00%	0.00%	22.50	0.77	27.28	28.66

Table 7 reports the performance of GRASP metaheuristic algorithm for small size instances. The results presented in this table are good in our opinion. The S1L, S3L and S4L algorithms find the optimal solution of instances where the S2L is unable to find the optimal solution for all instances.

Table 8. Algorithms Comparison with Local Search (medium size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1L	S2L	S3L	S4L	S1L	S2L	S3L	S4L
19	75	3	0.2	0.00%	0.00%	0.00%	0.00%	27.67	0.78	28.84	30.43
20	75	3	0.5	0.00%	0.00%	0.00%	0.00%	26.86	0.74	27.98	27.36
21	75	3	0.8	0.00%	0.00%	0.00%	0.00%	26.35	0.75	29.36	25.17
22	75	5	0.2	0.00%	0.92%	0.00%	0.00%	98.79	2.78	119.03	94.71
23	75	5	0.5	0.00%	0.74%	0.00%	0.00%	120.84	2.81	121.14	94.36
24	75	5	0.8	0.00%	0.46%	0.00%	0.00%	96.61	3.65	107.76	117.07
25	75	8	0.2	0.00%	0.00%	0.00%	0.00%	371.54	13.36	360.27	356.73
26	75	8	0.5	0.00%	0.00%	0.00%	0.00%	470.2	13.89	379.88	354.84
27	75	8	0.8	0.03%	0.00%	0.00%	0.00%	405.13	10.59	356.49	354.46
28	100	3	0.2	0.00%	1.15%	0.00%	0.00%	57.25	2.13	88.06	58.95
29	100	3	0.5	0.00%	0.76%	0.00%	0.00%	71.67	1.71	63.91	58.56
30	100	3	0.8	0.00%	0.15%	0.00%	0.00%	65.15	1.69	66.98	58.03
31	100	5	0.2	0.00%	0.91%	0.00%	0.00%	221.55	6.48	234.99	221.31
32	100	5	0.5	0.00%	0.31%	0.01%	0.00%	219.43	8.05	226.81	272.54
33	100	5	0.8	0.00%	0.34%	0.00%	0.00%	218.43	6.31	223.22	215.4
34	100	8	0.2	0.17%	1.83%	0.85%	0.00%	1067.42	24.85	850.88	1026.36
35	100	8	0.5	0.13%	1.34%	0.51%	0.00%	1066.06	24.83	957.77	895.76
36	100	8	0.8	0.01%	0.82%	0.34%	0.00%	901.05	24.48	867.89	855.64
Average:				0.02%	0.54%	0.10%	0.00%	307.33	8.33	283.96	284.32

Table 8 presents the performance of GRASP metaheuristic algorithm for medium size instances. The results presented in this table are good in our opinion. The S4L algorithm reaches the optimal solution of instances for all medium size instances. The other algorithms, however, fail to find the optimal solution to some of instances. The performance of GRASP algorithm for medium size instances again proves its efficiency in obtaining optimal solutions where our deviation is 0.00%.

Table 9. Algorithms Comparison with Local Search (large size instances)

Instances				% DEV				CPU Time			
Number	*N*	*P*	*α*	S1L	S2L	S3L	S4L	S1L	S2L	S3L	S4L
37	150	3	0.2	0.00%	3.66%	0.00%	0.00%	191.22	5.62	207.15	189.72
38	150	3	0.5	0.00%	2.05%	0.00%	0.00%	235.92	5.6	213.16	188.3
39	150	3	0.8	0.00%	1.01%	0.00%	0.00%	241.78	6.31	192.48	187.74
40	150	5	0.2	0.00%	0.76%	0.08%	0.00%	847.79	22.1	942.29	730.79
41	150	5	0.5	0.00%	0.11%	0.00%	0.00%	790	22.37	789.49	737.06
42	150	5	0.8	0.00%	0.35%	0.00%	0.00%	912.59	21.47	1023.21	726.81
43	150	8	0.2	0.09%	1.12%	0.05%	0.00%	3552.09	83.97	2974.89	3168.07
44	150	8	0.5	0.06%	0.75%	0.00%	0.00%	2870.95	84.61	3240.38	3502.63
45	150	8	0.8	0.00%	0.69%	0.00%	0.00%	3096.1	83.31	3047.79	2967.18
46	200	3	0.2	0.00%	0.00%	0.00%	0.00%	457.08	17.09	493.02	441.43
47	200	3	0.5	0.00%	0.00%	0.00%	0.00%	532.88	13.07	517.26	460.53
48	200	3	0.8	0.00%	0.00%	0.00%	0.00%	473.56	13.79	507.35	462.47
49	200	5	0.2	-0.01%	-0.01%	-0.01%	-0.01%	1954.99	58.29	1960.99	1795.94
50	200	5	0.5	0.00%	0.17%	0.00%	0.00%	1964.38	53.56	1748.07	1788.88
51	200	5	0.8	0.00%	0.32%	0.00%	0.00%	1922.98	64.52	1752.54	1696.7
52	200	8	0.2	0.00%	1.70%	0.68%	0.00%	6818.31	254.52	7167.32	6715.59
53	200	8	0.5	0.14%	0.86%	0.10%	0.00%	7111.83	206.3	7238.41	6833.39
54	200	8	0.8	0.00%	0.63%	0.03%	0.00%	7053.67	271.22	8904.21	7183.48
Average:				0.02%	0.79%	0.05%	0.00%	2279.34	71.54	2384.45	2209.82

Table 9 presents the performance of algorithms coupled with a neighborhood search procedure for large size instances. The results presented in this table are very good, in our opinion. The S4L algorithm finds the optimal solution to instances where the other three algorithms are unable to find the optimal solution to most of instances. That is, the gap between our incumbent and the optimal solution obtained by Bender algorithm is equal to 0.00%.

5.4 Computational summary

In this section we briefly describe the results presented in Tables 4 to 9. Clear enough, the results on the average DEV% explains that in fully random algorithm

without any neighborhood search procedure, the average gap is 15.07%. Note that the average gaps obtained from Greedy deterministic is 24.66% that is worse than average since the there is no randomness in searching other feasible regions. The performance of greedy randomize is, however, better as it benefits from our greedy function. The performance of our GRASP metaheuristic is significantly better than that of other heuristics. The key idea in GRASP is that it expands its neighborhood search as it benefits from a greedy function that selects potential hub location and, finally, the neighborhood search always obtains the optimal solution of the instances.

To have a slightly different point of view, the following two tables showing the results that categorized based of quantity of instances that we had done our test (20, 50, 75, 100, 150 and 200). Table 10 is the result of constructive phase and table 11 shows the result of constructive phase with the local search.

Table 10. % DEV for initial solutions and CPU Time

N	% DEV for initial solutions				CPU Time			
	S1	S2	S3	S4	S1	S2	S3	S4
20	14.78%	19.19%	13.64%	13.78%	0.48	0.01	0.46	0.51
50	19.78%	28.73%	18.11%	17.78%	1.48	0.03	1.40	1.39
75	13.55%	26.38%	16.37%	17.34%	2.77	0.04	2.55	2.81
100	12.76%	28.14%	8.95%	9.19%	4.66	0.05	4.37	4.69
150	14.63%	23.73%	9.95%	8.57%	11.13	0.09	10.42	10.04
200	14.93%	21.82%	10.16%	10.16%	21.20	0.18	19.25	18.18
AVG.	15.07%	24.66%	12.86%	12.86%	6.95	0.07	6.41	6.27

Table 11. % DEV for initial solutions with the local search and CPU Time

N	% DEV for initial solutions + local search				CPU Time			
	S1	S2	S3	S4	S1	S2	S3	S4
20	0.00%	0.07%	0.00%	0.00%	3.88	0.13	3.86	3.75
50	0.00%	0.01%	0.00%	0.00%	51.11	1.42	50.70	53.57
75	0.00%	0.24%	0.00%	0.00%	182.67	5.48	170.08	161.68
100	0.03%	0.85%	0.19%	0.00%	432.00	11.17	397.83	406.95
150	0.02%	1.17%	0.02%	0.00%	1415.38	37.26	1403.43	1377.59
200	0.02%	0.41%	0.09%	0.00%	3143.30	105.82	3365.46	3042.05
AVG.	0.01%	0.46%	0.05%	0.00%	871.39	26.88	898.56	840.93

Table 11 describes the effect of local search procedure on the incumbent solution and the computational times of each algorithm. The average %Dev for our GRASP metaheuristic is again Zero where other algorithms fail to reach optimal solution. This demonstrates the impact of our greedy function in providing diverse and good feasible solutions at construction phase of the GRASP algorithm.

The following table gives us a summary of view of the results that shows it consolidate. Basically the numbers are the same one written in the last row of table 10 and 11. That gives another view for the result throughout of all algorithms that were worked in this thesis.

Table 12. result

Algorithm	%DEV	CPU Time	%DEV & LS	CPU Time
Fully Randomize	15.07%	6.95	0.01%	871.39
Greedy Deterministic	24.66%	0.07	0.46%	26.88
Greedy Randomize	12.86%	6.41	0.05%	898.56
GRASP	12.80%	6.27	0.00%	840.93

CHAPTER 6: Conclusion and Future Research Directions

In this thesis, we developed a GRASP metaheuristic for Multiple Allocation p-Hub Location Problems. We run a series of experiments on a set of benchmark instances with up to 200 nodes to evaluate the effectiveness of the proposed algorithm. We compared the solutions obtained from our algorithm to that of a Benders decomposition known to literature that further demonstrates the capability of our algorithm in finding good quality solutions within a reasonable computational time. An observation from our research highlights the effectiveness of our greedy function on constructive phase of the GRASP algorithm. In addition, future work on metaheuristic solution methods should be considered to efficiently obtain better feasible solutions for larger instances.

Future studies might consider studying the capacitated version of the problem for efficiently obtaining good quality solutions for larger instances. A modification in the construction process of selecting hubs where the standard deviation of distances is updated for the remaining nodes after each selecting a hub node could be of another future work.

References

Alumur, S., Kara, B.Y. (2008). Network hub location problems: The state of the art. *European Journal of Operational Research, 190(1): 1-21.*

Aykin,T. (1995).The hub location and routing problem. *European Journal of Operational Research, 29(3):200-219.*

Campbell, J.F. (1994b). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research, 72:387-405.*

Campbell, J.F. (1996). Hub location and the p-hub median problem. *Operations Research, 44(6): 923-935.*

Campbell, J.F., Ernst, A.T. and Krishnamoorthy, M., (2002). Hub location problems. *In: Drezner, Z., Hamacher, H.W. (Eds.), Facility Location: Applications and Theory. Springer, Heidelberg, pp. 373-408.*

Campbell, J. F., Ernst, A. T. and Krishnamoorthy, M. (2005).Hub arc location problems: Part I - introduction and results. *Management Science, 51(10):1540-1555*.

Campbell, A. M., Lowe, T. J., and Zhang, L. (2007). The p-hub center allocation problem. *European Journal of Operational Research, 176(2):819-835.*

Campbell, J.F., O'Kelly, M.E. (2012).Twenty-five years of hub location research. *Transportation Science, 46(2):153-169*.

Contreras, I. (2015). Hub location problems. In *Location Science*, pp. 311-344, Springer International Publishing.

Contreras, I., Cordeau, J.F. and Laporte, G. (2011a) stochastic uncapacitated hub location. *European Journal of Operational Research, 212: 518-528.*

Contreras, I., Cordeau, J.F. andLaporte, G. (2011c) Benders decomposition for large-scale uncapacitated hub location. *Operations Research, 59(6):1477-1490.*

Ernst, A.T., Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated Single allocation p-hub median problem. *Location Science, 4:139-154.*

Ernst A.T., Krishnamoorthy M. (1998a). Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problem. *European Journal of Operational Research, 104: 100-112.*

Ernst, A. T., Hamacher, H., Jiang, H., Krishnamoorthy, M., and Woeginger, G. (2009). Uncapacitated single and multiple allocation p-hub center problems.*Computers & Operations Research, 36(7):2230-2241.*

Feo T.A., Resende, M.G.C. (1995). A greedy randomized adaptive search procedure. *Journal of global optimization. 6(2):109-133.*

Garcia, S., Landete, M., and Marin, A. (2012). New formulation and a branch-andcut algorithm for the multiple allocation p-hub median problem. *European Journal of Operational Research, 220(1):48-57.*

Glover, F. (1996). Tabu search and adaptive memory programming-advances, applications and challenges. Interfaces in computer science and operations research. Kluwer, 1-75.

Golden, B., Skiscim, C. (1986). Using simulated annealing to solve routing and location problems. *Naval Research Logistics Quarterly. 33:261-279.*

Hakimi, SL. (1964). Optimal locations of switching centers and the absolute centers and medians of a graph. *Operations Research 12(3): 450-459.*

Hart, J.P., Shogan. A.W. (1987). Semi-greedy heuristics: An empirical study. *Operations Research Letters.6:107-114.*

Kara, B.Y., Tansel, B.C. (2000).On the single assignment p-hub center problem. *European Journal of Operational Research, 125(3):648-55.*

Kara, B. Y. and Tansel, B. C. (2001). The latest arrival hub location problem. *Management Science, 47(10):1408-1420.*

Kara, B.Y., Tansel, B.C. (2003). The single-assignment hub covering problem: Models and linearizations. *Journal of the Operational Research Society, 54(1):59-64.*

Kirkpatrick.S., Gclatt, C. D.,Vecchi, Jr. and Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science, 220(4598): 671-680.*

Klincewicz, J.G. (1991). Heuristics for the p-hub location problem*. European Journal of Operational Research. 53:25-37.*

Klincewicz, J.G. (1992). Avoiding local optima in the p-hub location problem using tabu search and grasp. *Annals of Operational Research, 40: 283-302.*

Kratica, J. (2013). An electromagnetism-like metaheuristic for the uncapacitated multiple allocation p-hub median problem. *Computers & Industrial Engineering, 66(4):1015-1024.*

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. Naval Research Logistic Quarterly, 2:83-97.

Michalewicz, Z., Janikow, CZ (1992). A modified genetic algorithm for optimal control problems. *Computers & Mathematics with Applications. 23(12):83-94.*

Milanovic, M. (2010). A new evolutionary based approach for solving the uncapacitatedmultiple allocation p-hub median problem. *In Soft Computing in IndustrialApplications, pages 81- 88. Springer.*

O'Kelly, M.E. (1986a). Activity levels at hub facilities in interacting networks. *Geographical Analysis, 18(4):343-356.*

O'Kelly, M.E. (1986b). The location of interacting hub facilities. *Transportation Science. 20:92-106.*

O'Kelly, M.E. (1987). A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research, 32: 393-404.*

O'Kelly, M.E. (1992). Hub facility location with fixed costs. *Papers in Regional Science, 20(2): 293-306.*

Pamuk, F. S. and Sepil, C. (2001). A solution to the hub center problem via a single-relocation algorithm with tabu search. *IIE Transactions, 33(5):399-411.*

Peiro, J., Corberan, A. and Marti, R. (2014). GRASP for the uncapacitated r-Allocation p-hub median problem. *Computers & Operations Research 43: 50-60.*

Pirkul, H. and Schilling, D. A. (1998). An efficient procedure for designing single allocation hub and spoke systems. *Management Science, 44(12-Part-2):235-242.*

Sasaki, M., Suzuki, A. and Drezner, Z. (1999). On the selection of hub airports for an airline hub-and spoke system. *Computers and Operations Research. 26:1411-1422.*

Skorin-Kapov, D., Skorin-Kapov, J. (1994). On tabu search for the location of interacting hub facilities. *European Journal of Operations Research, 73:502-509.*

Skorin-Kapov, D., Skorin-Kapov, J. and O'Kelly, M. (1996). Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research, 73:501–508*.

Sohn, J., Park, S. (1998). Efficient solution procedure and reduced size formulation for p-HLP. *European Journal of Operational Research, 108:118-126.*

Wagner, B. (2004). A note on "the latest arrival hub location problem". *Management science. 50(12):1751-1752.*

Wagner, B. (2008). Model formulations for hub covering problems. *Journal of the Operational Research Society. 59(7):932-938.*

Yaman, H., Kara, B. Y., and Tansel, B. C. (2007). The latest arrival hub location problem for cargo delivery systems with stopovers. *Transportation Research Part B: Methodological, 41(8):906-919.*

Yaman, H. (2011). Allocation strategies in hub networks. *European Journal of Operational Research, 211(3):442-451.*

Zanjirani-Farahani, R., Hekmatfar, M., Boloori-Arabani, A. and Nikbakhsh, E. (2013). Hub location problems: a review of models, classification, techniques and application. *Computers & Industrial Engineering, 64(4):1096-1109*.

Zanjirani-Farahani, R., Rezapour, S., Drezner, T., & Fallah, S. (2014). Competitive supply chain network design: An overview of classifications, models, solution techniques and applications. *Omega, 45, 92-118.*