Heuristic Solutions for the Stochastic r-Allocation p-Hub Median Problem with Non-stop Services*

Juanjo Peiró†[a], Francisco Saldanha-da-Gama[b,c], Ángel Corberán[a], and Rafael Martí[a]

[a]Departament d'Estadística i Investigació Operativa, Universitat de València, Spain
[b]Departamento de Estatística e Investigação Operacional, Universidade de Lisboa, Portugal
[c]Centro de Matemática, Aplicações Fundamentais e Investigação Operacional, Universidade de Lisboa, Portugal

Abstract

In this work we propose a heuristic procedure for a stochastic version of the Uncapacitated r-Allocation p-Hub Median Problem with non-stop services. In particular, we assume that the number of hubs to which a terminal can be allocated, is bounded from above by r. Additionally we consider the possibility of shipping traffic directly between terminals (non-stop services) in case this renders savings in the overall cost. Uncertainty is associated with the traffics to be shipped between nodes and with the transportation costs. If we assume that such uncertainty can be captured by a finite set of scenarios, each of which having some known occurrence probability, it is possible to develop a compact formulation for the deterministic equivalent problem. However, even for small instances of the problem, the model becomes too large to be easily tackled by a general purpose solver. This fact motivates the development of an approximate procedure, whose starting point is the determination of a feasible solution to every (deterministic) single-scenario problem. These solutions are then embedded into a process inspired by Path Relinking: gradually an initial solution to the overall problem is transformed by the incorporation of attributes from some guiding solutions. In our case, the guiding solutions are those found for the single-scenario problems. We report and discuss the results of the computational experiments performed using instances randomly generated for the new problem using the well-known CAB data set.

Keywords and phrases: stochastic hub location, non-stop services, heuristics

*This research was partially developed during a four months visit of Juanjo Peiró to the *Departamento de Estatística e Investigação Operacional, Universidade de Lisboa, Portugal*, thanks to the grant EEBB-I-15-09778 (Ayudas para la movilidad predoctoral para la realización de estancias breves en centros I+D 2015 del Ministerio de Economía y Competitividad de España).

†Corresponding author: juanjo.peiro@uv.es; Universitat de València. Facultat de Ciències Matemàtiques. Carrer del Doctor Moliner, 50. 46100 - Burjassot (València), Spain.

1 Introduction

A hub location problem results from the need to ship traffic from many-to-many nodes in a network. Instead of shipping the traffic directly between nodes, a subset of them is selected for becoming hubs, thus consolidating and distributing the flow. This induces a transportation network that helps making the shipment more efficient and cheaper. For instance, we can take advantage from economies of scale when transporting (large amounts of) traffic between hubs.

Hub location is a topic that has applications in many different areas such as Telecommunication, Logistics, and Transportation. The extensive work that has been developed in this area as well as the applications that have been studied are very well summarized in the recent book chapter by Contreras (2015) and also in the survey papers by Alumur and Kara (2008) and Campbell and O'Kelly (2012). These works and the references there in show that nowadays we can identify several research branches and problem classes in this field. For instance, we find problems in which the number of hubs is exogenously defined while, in others, this is an outcome of the decision making process. Additionally, hubs may be capacitated (when there is some limit for the traffic that can go through them) or uncapacitated. In fact, within the context of hub location, many aspects can be isolated, each of which helping in the characterization of the problem at hand.

One important aspect in a hub location problem regards the allocation pattern for the non-hub nodes (terminals). Traditionally, two patterns have been considered in the literature: single allocation and multiple allocation. In the first case, a terminal is allocated to exactly one hub, whereas in the second a terminal can be allocated to several hubs (without a limit). Recently, these two patterns have been unified under the so-called r-allocation pattern that was introduced by Yaman (2011). In this case, a limit, r, is imposed on the maximum number of hubs to which a terminal can be allocated. In the particular case where $r = 1$ we obtain the single allocation scheme but if we set r large enough (e.g., equal to the number of nodes) then we obtain a multiple allocation problem.

Another important aspect concerning hub location is related with the shipment of the traffic originated at each node. Most of the literature assumes that all traffic must be routed via at least one hub which prevents direct shipments between terminals. However, in some applications (e.g., in Logistics) it may be possible (and even advantageous) to make direct shipments between terminals. This is the case, for instance, if the volume of traffic between the nodes is high and no specialized facility or equipment is required for processing that traffic. Some authors have noticed the practical relevance of considering these "non-stop" services. This is the case with Aykin (1994, 1995a,b), Sung and Jin (2001), Nickel et al. (2001), and Wagner (2007). Nevertheless, the literature capturing this feature is still much scarce.

Finally, one aspect that is increasingly attracting the attention of academicians and practitioners concerns embedding uncertainty in optimization models. This is not a new topic. However, the technological developments we have observed in the past decades (e.g., the huge increase in computing power) made possible to give more relevance to that aspect. This has led to more comprehensive and, from a practical point of view, more relevant models. Hub location has not been ignored in this trend. In fact, several works can already be found in the literature capturing uncertainty in optimization models developed for hub location problems. To the best of the authors' knowledge Marianov and Serra (2003) is the first work dealing with uncertainty within the context of hub location. Sim et al. (2009) introduce the stochastic p-hub center problem. Often, the uncertain parameters are related with the amount of traffic to ship. This is the case, for instance, with the work by Bollapragada et al. (2005). However, other possibilities, as uncertainty in the costs, have been studied (the reader should refer to Contreras et al. 2011a, and Alumur et al. 2012).

Most of the "classical" hub location problems are NP-hard. Accordingly, the same holds for many of their extensions and thus it is not surprising to find much literature presenting heuristic procedures in this field. Along with the first mathematical formulation for the single allocation p-hub median problem, O'Kelly (1987) presented two specially tailored heuristics for obtaining feasible solutions to the problem. Since then, many heuristics have been developed. The uncapacitated single allocation p-hub median problem was further studied by Klincewicz (1992) (tabu search and GRASP), Campbell (1996) (a greedy procedure), Ernst and Krishnamoorthy (1996) (simulated annealing), Kratica et al. (2007) (genetic algorithms), Smith et al. (1996) (neural networks), and Ilić et al. (2010) (variable neighborhood search). The capacitated version of the problem was investigated by Stanimirović (2010), who proposed a genetic algorithm. The unified allocation pattern in the context of this problem, i.e., the uncapacitated r-allocation p-hub median problem, was investigated by Peiró et al. (2014) and Martí et al. (2015). In the former paper, a GRASP procedure was developed whereas the latter was devoted to presenting a scatter search algorithm.

The single allocation p-hub center problem was tackled by Pamuk and Sepil (2001) using tabu search and by Meyer et al. (2009), who developed a 2-phase method based upon ant colony optimization.

The use of (meta)heuristics for approximating the optimal solution to hub location problems goes much beyond the problems for which an exogenous number of hubs, p, is imposed. In fact, the uncapacitated single allocation hub location problem was studied by Abdinnour-Helm and Venkataramanan (1998) (genetic algorithms), who improved the results presented by Abdinnour-Helm (1998) that investigated a hybridization between genetic algorithms and tabu search. Other heuristics for the problem include those developed by Pirkul and Schilling (1998) (lagrangean heuristic), and Cunha and Silva (2007) (genetic algorithms). The multiple allocation version of the problem was investigated by Kratica et al. (2012), who presented a genetic algorithm.

The capacitated single allocation hub location problem was first tackled heuristically by Ernst and Krishnamoorthy (1999) (simulated annealing), and afterwards by Chen (2007) (simulated annealing combined with tabu search), Randall (2008) (ant colony optimization), Silva and Cunha (2009) (tabu search), and Contreras et al. (2009, 2011b) (lagrangean heuristics). The multiple allocation version of the problem was considered by Kratica et al. (2011) (genetic algorithms), and Rodríguez-Martín and Salazar-González (2006) (iterative local search). We note that in these two works, unlike the other works already quoted, the hub level network can be incomplete, i.e., it does not need to be a complete graph. These are problems with (hub level) network design decisions. Also considering incomplete hub networks we find the work by Calık et al. (2009) on a hub covering problem whose optimal solution is approximated using tabu search.

Other works containing heuristics for hub location problems include those by Marianov et al. (1999) on competitive hub location (tabu search was considered), Eiselt and Marianov (2009) (using heuristic concentration—Rosing and ReVelle 1997), and Lüer-Villagra and Marianov (2013) (who developed a genetic algorithm).

The variety of heuristics for hub location problems covered by the literature includes other more specific hub location problems not quoted above. This is the case with the papers by Yaman and Carello (2005) (local search), Marianov and Serra (2003) (tabu search), Sasaki et al. (1999) (greedy approach that generalizes the procedures suggested by Campbell (1996)), and Alumur and Serper (2016) (variable neighborhood search).

As far as stochastic hub location problems are concerned, to the best of the authors' knowledge, the only contribution to the literature so far is the paper by Bollapragada et al. (2005). The authors studied a fixed-wireless network-planning problem with a two phase planning horizon and a different budget for each phase. They considered different hub types (regarding costs and capacities) and assume stochastic demands. A greedy algorithm was proposed for maximizing the expected demand covered.

In the current paper we extend the uncapacitated r-Allocation p-Hub Median Problem and develop an algorithm for finding high-quality feasible solutions. The original problem is extended in two directions: (i) by considering uncertainty in the traffics and in the shipping costs, and (ii) by allowing non-stop services. We assume that uncertainty can be represented by a stochastic random vector with some known probability distribution. This leads to the adoption of a two-stage stochastic modeling framework for the problem: the first-stage decisions refer to the network design (selection of hubs and allocation of terminals to the hubs); the second-stage decision is dependent on how uncertain is revealed and regards the transportation of the traffics through the network.

When the underlying random vector above mentioned has a finite support, it is possible to derive a compact mixed-integer linear programming formulation for the deterministic equivalent problem. Nevertheless, this is a large-scale optimization model even for small instances of the problem, which motivates the development of a heuristic algorithm for obtaining high-quality feasible solutions. The procedure we propose is inspired on the Path Relinking methodology (Glover and Laguna 1997). In particular, it progressively changes an initial solution to the problem by incorporating attributes from a set of guiding solutions. In our case, the guiding solutions are feasible solutions previously obtained to the single-scenario problems (one for each).

The development of (meta)heuristics for stochastic combinatorial optimization problems is not a new topic as we can observe in the survey paper by Bianchi et al. (2009). Nevertheless, we can also conclude that most of the work was developed on stochastic traveling salesman problems, on stochastic vehicle routing, and on stochastic scheduling. Within the context of facility location, no much work can be found. The paper by Albareda-Sambola et al. (2013) is a good exception. The authors introduced a so-called fix-and-relax-coordination procedure for a multi-period location–allocation problem under uncertainty. This is a specialization of the fix-and-relax heuristic (Dillenberger et al. 1994, Escudero and Salmerón 2005) embedding a branch-and-fix coordination two-stage solution algorithm (Alonso-Ayuso et al. 2003).

The new methodology we propose in this work, represents a contribution to the development of heuristic approaches for stochastic hub location problems that can be easily extended to other stochastic discrete optimization problems. The remainder of the paper is organized as follows: in Section 2 we present a mathematical model for the deterministic version of the problem we are investigating. Afterwards, we extend the model to a setting in which the traffics and shipping costs are stochastic. In Section 3 we introduce the new heuristic procedure. In particular, Section 3.1 is devoted to developing a heuristic to the deterministic (single-scenario) version of the problem. Finally, in Section 4, we present the computational tests performed in order to assess the quality of the new heuristic. The paper ends with some conclusions and an outlook of the work done.

2 The Uncapacitated r-Allocation p-Hub Median Problem with non-stop services

In this section we start by summarizing the most important results concerning the (deterministic) Uncapacitated r-Allocation p-Hub Median Problem introduced by Yaman (2011), as well as the special case in which non-stop services are allowed (Sung and Jin 2001). Then, we introduce a stochastic version, which, as far as we know, is the first time it has been considered in the Literature.

2.1 Deterministic model

Consider a network $G = (V, E)$ with a set of demand nodes V and a set of edges E, and let t_{ij} be the amount of traffic to be sent between each pair of nodes i and j. In the Uncapacitated r-Allocation p-Hub Median Problem (UrApHMP), traffic t_{ij} has to be routed along a path $i \to k \to l \to j$, where nodes k and $l \in V$ are used as intermediate points for this transportation. The UrApHMP consists of choosing a set H of p nodes that are used as intermediate transfer points between any pair of nodes in G, allocating each terminal to at most r of the p hubs, and such that the total transportation cost is minimized. The traffic to and from each terminal can be routed via one or several hubs among the ones to which the terminal is allocated. Each terminal The nodes in H are commonly called *hub nodes* or, simply, *hubs*, while all the other nodes in the network are known as *terminal nodes* or *terminals*.

The UrApHMP was first introduced by Yaman (2011), who also pointed out the possibility of considering non-stop services between origin–destination pairs. Non-stop services, as considered by Aykin (1994, 1995a,b), Sung and Jin (2001), Nickel et al. (2001), and Wagner (2007), refers to the possibility of sending traffics between any pair of nodes using a direct edge, at a given cost, rather than going through their corresponding hubs. The direct transportation via such a direct edge is called a non-stop service.

The problem we study in this paper is the UrApHMP with non-stop services, where fixed assignment costs of terminals to hubs are also considered. We call this problem the Uncapacitated r-Allocation p-Hub Median Problem with non-stop services (UrApHMP-NSS).

In order to formulate the problem as a mixed-integer linear optimization problem, we define the following variables. Given a node $k \in V$, $z_{kk} = 1$ if node k is set to be a hub and $z_{kk} = 0$, and otherwise. Given a non-hub node $i \in V$, $z_{ik} = 1$ if node i is assigned to node k and $z_{ik} = 0$ otherwise. Moreover, we define x_{ijkl} as the proportion of the traffic t_{ij} from node i to node j that travels along the path $i \to k \to l \to j$, where k and l are the nodes that will be used as hubs. Finally, for two nodes $i, j \in V$, $y_{ij} = 1$ if t_{ij} is routed on a non-stop service, and $y_{ij} = 0$ otherwise. The UrApHMP-NSS, similarly to the formulation proposed in Yaman (2011), can be formulated as follows.

$$
\begin{aligned}
\min \quad & \sum_{i,k \in V, i \neq k} a_{ik} z_{ik} + \sum_{i,j,k,l \in V} t_{ij}(\chi c_{ik} + \alpha c_{kl} + \delta c_{lj}) x_{ijkl} \\
& + \sum_{i,j \in V} (t_{ij} d_{ij} + b_{ij}) y_{ij} && (1) \\
\text{s. t.} \quad & \sum_{k \in V} z_{kk} = p && (2) \\
& \sum_{k \in V} z_{ik} \leq r, \qquad \forall i, k \in V && (3) \\
& z_{ik} \leq z_{kk}, \qquad \forall i, k \in V && (4) \\
& \sum_{l \in V} x_{ijkl} \leq z_{ik}, \qquad \forall i, j, k \in V && (5) \\
& \sum_{k \in V} x_{ijkl} \leq z_{jl}, \qquad \forall i, j, l \in V && (6) \\
& \sum_{k \in V} \sum_{l \in V} x_{ijkl} + y_{ij} = 1, \qquad \forall i, j \in V : t_{ij} > 0 && (7) \\
& x_{ijkl} \geq 0, \qquad \forall i, j, k, l \in V && (8) \\
& y_{ij} \in \{0, 1\}, \qquad \forall i, j \in V && (9) \\
& z_{ik} \in \{0, 1\}, \qquad \forall i, k \in V. && (10)
\end{aligned}
$$

The objective function (1) represents the total cost. It consists of the allocation cost of terminals to hubs and the transportation cost of the traffics. The first term refers to the assignment cost, a_{ik}, of each node $i \in V$ to a particular hub $k \in V$, regardless of the amount of traffic node i sends or receives through k. The second term represents the cost of transporting all traffics t_{ij} through the hubs, where χ, α, and δ are unit rates for collection (origin-hub), transfer (hub-hub) and distribution (hub-destination), respectively. In this term, c_{ik}, c_{kl}, and c_{lj} denote the cost of shipping all the traffic t_{ij} via the edges (i, k), (k, l), and (l, j), respectively. The last term of the objective function describes the cost of transporting traffics using non-stop services. In this term, d_{ij} denotes the cost of shipping the traffic t_{ij} using a non-stop service. Note that there is also a fixed cost, b_{ij}, associated with the use of this direct edge between i and j.

Constraint (2) imposes to use exactly p nodes as hubs, while constraints (3) restrict any node to be assigned to at most r of the p hubs. Constraints (4) assure that if node k is not a hub, node i cannot be assigned to it. Moreover, constraints (5) and (6) guarantee that if nodes i and j are not assigned to hubs k and l, no traffic can be sent between i and j through those hubs. Constraints (7) ensure that all traffic of the network is routed, either using the hub connections or non-stop services. Finally, the variables domains are stated in constraints (8)–(10).

2.2 A two-stage stochastic model

We introduce next a stochastic version of the above problem. In particular we assume that demands and transportation costs are not known in advance but can be captured by a probability distribution. This is motivated by the following observations: (i) a hub location problem has embedded a network design problem (locating the hubs and allocating the terminals to the hubs) whose related decisions often have a long-lasting effect. Hence, these are intrinsically strategic decisions typically made before having perfect information about the future. On the other hand, the decisions about transportation of the traffics are often operational decisions that can be made just in time, when precise information is available, i.e., after uncertainty is revealed. (ii) What is more, transportation costs are often related to the price of resources like electricity or oil, and therefore they are quite difficult to predict. However, the actual transportation costs incurred will depend on how the network design decisions were made since the latter condition the former.

The above observations also motivate the use of a two-stage stochastic programming optimization model in which the here-and-now decisions (first-stage) concern the network design and the recourse decisions (second-stage) regard the transportation of the traffics. The latter are called "recourse decisions" because they are made in such a way that the best response is given (depending on the occurring scenario) to the setting defined by the first-stage decisions. This modeling framework is not new within the context of hub location. In the works by Contreras et al. (2011a) and Alumur et al. (2012) we can observe stochastic hub location problems with the network design decisions separated from the transportation decisions.

We call scenario a complete realization of all the uncertain parameters. The number of possible scenarios can be either finite or infinite, depending on the supports of the random variables involved in the problem. In fact, if, for every $i, j \in V$, the traffic t_{ij} is assumed to be a random variable, the same happening with the costs c_{ij} and d_{ij}, $(i,j) \in E$, then the random vector underlying the problem is $\boldsymbol{\xi} = \left[[t_{ij}]_{i,j\in V}, [c_{ij}]_{(i,j)\in E}, [d_{ij}]_{(i,j\in E)}\right]$. Each realization of this random vector is a scenario. We assume that it is possible to compute or estimate accurately the probability associated with each scenario (the reader can refer to Alumur et al. 2012, for a deeper discussion on this issue).

We can introduce a stochastic version of the UrApHMP-NSS, as follows:

$$\begin{aligned} \min \quad & \sum_{i,k\in V, i\neq k} a_{ik} z_{ik} + \mathcal{Q}(\mathbf{z}) && (11) \\ \text{s.t.} \quad & (2) - (4), (10), \end{aligned}$$

where $\mathcal{Q}(\mathbf{z}) = E_{\boldsymbol{\xi}}[Q(\mathbf{z}, \boldsymbol{\xi})]$ is the mathematical expectation with respect to ξ, and

$$Q(\mathbf{z}, \boldsymbol{\xi}) = \quad \min \sum_{i,j,k,l\in V} t_{ij}(\chi c_{ik} + \alpha c_{kl} + \delta c_{lj}) x_{ijkl}$$

$$+\sum_{i,j\in V}(t_{ij}d_{ij}+b_{ij})y_{ij} \tag{12}$$

$$\text{s.t.} \quad \sum_{l\in V} x_{ijkl} \leq z_{ik}, \qquad \forall i,j,k\in V \tag{13}$$

$$\sum_{k\in V} x_{ijkl} \leq z_{jl}, \qquad \forall i,j,l\in V \tag{14}$$

$$\sum_{k\in V}\sum_{l\in V} x_{ijkl} + y_{ij} = 1, \quad \forall i,j\in V : t_{ij} > 0 \tag{15}$$

$$x_{ijkl} \geq 0, \qquad \forall i,j,k,l\in V \tag{16}$$

$$y_{ij} \in \{0,1\}, \qquad \forall i,j\in V. \tag{17}$$

If the support, say Ξ, of the random vector $\boldsymbol{\xi}$ is finite, we can index the scenarios in the set $S=\{1,\dots,|\Xi|\}$. Accordingly, we can also index the stochastic parameters and the second-stage decision as follows: for $s\in S$, t_{ijs} is the traffic to be sent from i to j under scenario s, c_{iks}, c_{kls}, and c_{ljs} correspond to the cost of shipping all the traffic t_{ijs} via the edges (i,k), (k,l), and (l,j), respectively, under scenario s, d_{ijs} denotes the cost of shipping the traffic t_{ijs} using a non-stop service under scenario s, x_{ijkls} is the the proportion of the traffic t_{ij} from node i to node j that travels along the path $i\to k\to l\to j$, where k and l are the nodes that will be used as hubs, and y_{ijs} is a binary variable equal to 1 if a non-stop service is used in scenario s for shipping the traffic t_{ijs} and 0 otherwise.

Using this new notation and representing by π_s the probability associated with scenario $s\in S$, we can present the so-called extensive form of the deterministic equivalent:

$$\begin{aligned}
\min \quad & \sum_{i,k\in V, i\neq k} a_{ik}z_{ik} \\
& + \sum_{s\in S}\pi_s \left[\sum_{i,j,k,l\in V} t_{ijs}(\chi c_{iks}+\alpha c_{kls}+\delta c_{ljs})x_{ijkls}\right. \\
& \left. + \sum_{i,j\in V}(t_{ijs}d_{ijs}+b_{ijs})y_{ijs}\right]
\end{aligned} \tag{18}$$

$$\text{s.t.} \quad (2)-(4),(10)$$

$$\sum_{l\in V} x_{ijkls} \leq z_{ik}, \qquad \forall i,j,k\in V,\ s\in S \tag{19}$$

$$\sum_{k\in V} x_{ijkls} \leq z_{jl}, \qquad \forall i,j,l\in V,\ s\in S \tag{20}$$

$$\sum_{k\in V}\sum_{l\in V} x_{ijkls} + y_{ijs} = 1, \qquad \forall i,j\in V,\ s\in S \tag{21}$$

$$x_{ijkls} \geq 0, \qquad \forall i,j,k,l\in V,\ s\in S \tag{22}$$

$$y_{ijs} \in \{0,1\}, \qquad \forall i,j\in V,\ s\in S. \tag{23}$$

The above model will be denoted by $\mathcal{P}$. We note that the non-anticipativity principle is implicit in this model since the choice to be made for the z-variables should will result the same no matter the occurring scenario. In other words, the challenge here is to select a set of hubs and assign the terminals to these hubs in a way that performs well in every possible situation (scenario).

2.3 Making a decision under uncertainty—A minmax regret model

The stochastic model just presented lies on the assumption that the probabilities π_s are known. If this is not the case, then, alternatives are necessary for formulating the problem. One possibility explored within the context of hub location by Alumur et al. (2012) is to consider a min-max regret model. We can propose the same type of model for the UrApHMP-NSS under uncertainty.

We first notice that model (2)-(4), (10), (18)–(23) can be solved for a subset of scenarios and, in particular, the model can be solved for a single scenario $s \in S$ by setting $\pi_s = 1$. The resulting solution is the optimal solution for scenario s whose value we can denote by $\mathcal{V}_s$.

After having computed the values $\mathcal{V}_s$, $s \in S$ we can compute the so-called regret of some solution $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ with respect to a scenario s. This is done as follows:

$$R_s = \sum_{i,k \in V, i \neq k} a_{ik} z_{ik} + \left[\sum_{i,j,k,l \in V} t_{ijs}(\chi c_{iks} + \alpha c_{kls} + \delta c_{ljs}) x_{ijkls} + \sum_{i,j \in V} (t_{ijs} d_{ijs} + b_{ijs}) y_{ijs} \right] - \mathcal{V}_s, \quad s \in S. \quad (24)$$

The problem consists of finding the solution $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ that minimizes the maximum regret we can observe according to:

$$\min \left\{ \max_{s \in S} R_s \right\} \quad (25)$$

$$\text{s.t.} \quad (2)-(4), (10), (19)-(23)$$

$$R_s = \sum_{i,k \in V, i \neq k} a_{ik} z_{ik} + \left[\sum_{i,j,k,l \in V} t_{ijs}(\chi c_{iks} + \alpha c_{kls} + \delta c_{ljs}) x_{ijkls} + \sum_{i,j \in V} (t_{ijs} d_{ijs} + b_{ijs}) y_{ijs} \right] - \mathcal{V}_s, \qquad s \in S. \quad (26)$$

3 A Greedy Attributive Scenario based Constructive Method

In this section we present an algorithm for the Stochastic UrApHMP-NSS. Since problem $\mathcal{P}$ is a mixed-integer linear program with a large number of binary variables even for small instances, it is a very hard task to solve it to optimality Hence, we propose a heuristic algorithm for finding high-quality feasible solutions to the problem. The idea is to iteratively build solutions to $\mathcal{P}$ using the solutions of single-scenario problems.

For each scenario $s \in S$, let us denote by $\mathcal{P}_s$ the problem associated with s:

$$\min \sum_{i,k \in V, i \neq k} a_{ik} z_{ik} + \sum_{i,j,k,l \in V} t_{ijs}(\chi c_{iks} + \alpha c_{kls} + \delta c_{ljs}) x_{ijkls}$$
$$+ \sum_{i,j \in V} (t_{ijs} d_{ijs} + b_{ijs}) y_{ijs} \tag{27}$$

$$\text{s.t.} \quad (2) - (4), (10)$$
$$\sum_{l \in V} x_{ijkls} \leq z_{ik}, \qquad \forall i, j, k \in V \tag{28}$$
$$\sum_{k \in V} x_{ijkls} \leq z_{jl}, \qquad \forall i, j, l \in V \tag{29}$$
$$\sum_{k \in V} \sum_{l \in V} x_{ijkls} + y_{ijs} = 1, \qquad \forall i, j \in V \tag{30}$$
$$x_{ijkls} \geq 0, \qquad \forall i, j, k, l \in V \tag{31}$$
$$y_{ijs} \in \{0, 1\}, \qquad \forall i, j \in V. \tag{32}$$

The solutions for problems $\mathcal{P}_s$, $s \in S$, may render different network designs, i.e., distinct hubs selected as well as distinct allocations of terminals. Even, the number of hubs to which a terminal is assigned to may be different for one scenario to another. Moreover, problems $\mathcal{P}_s$ are also $NP-hard$ (they have the classical multiple allocation hub location problem as a particular case) and thus they can hardly be solved to optimality even using a specially tailored algorithm. Accordingly, we can also resort to heuristics in order to find good feasible solutions.

The algorithm we proposed next is based upon the idea that good solutions for the single-scenario problems $\mathcal{P}_s$, $s \in S$, may contain information about the good attributes of a good solution (possibly optimal) to $\mathcal{P}$. Furthermore, a well-known feature of the optimal solutions to stochastic programming problems is that they represent a trade-off between the solutions for the different scenarios (for the different realizations of the uncertainty).

These facts motivate our method, which aims at combining the information provided by the single-scenario problems to obtain a solution for $\mathcal{P}$.

The full procedure, whose pseudo-code is shown in Algorithm 1, is performed until a previously defined number of iterations or time limit is reached. The final solution is the best found throughout the process.

3.1 A heuristic for the UrApHMP-NSS.

In this section we describe the heuristic method devised to obtain feasible solutions to a problem $\mathcal{P}_s$, $s \in S$, which turns out to be a heuristic for the UrApHMP-NSS. It consists of a constructive phase followed by a local search phase.

3.1.1 Constructive procedure

A solution to the UrApHMP-NSS is fully determined by (i) selecting the p hubs, (ii) allocating each terminal to at most r hubs, and (iii) transporting the traffics. Each of this components can be looked at as a subproblem. The constructive

Algorithm 1: Main loop of the algorithm to solve $\mathcal{P}$

Input: G, t_{max}, $itermax$

1 Initialize $\beta^* \leftarrow +\infty$ and $\beta_s \leftarrow \infty, \forall s \in S$
2 **while** t_{max} is not reached **do**
3   **foreach** $s \in S$ **do**
4   value of $(z, x, y)^s \leftarrow +\infty$
5   **for** $iter \leftarrow 1$ **to** $itermax$ **do**
6   Construct $(z, x, y)^s_{iter}$
7             $\mathsf{LS}_{\mathsf{change}}((z, x, y)^s_{iter})$
8             $\mathsf{LS}_{\mathsf{reduce}}((z, x, y)^s_{iter})$
9   // Update the incumbent solution
10   **if** value of $(z, x, y)^s_{iter} <$ value of $(z, x, y)^s$ **then**
11                 $(z, x, y)^s \leftarrow (z, x, y)^s_{iter}$
12   update β_s;
13   Obtain $(z, x, y)^{\mathcal{P}}$ from $(z, x, y)^s$, for all $s \in S$;
14   **if** value of $(z, x, y)^{\mathcal{P}} < \beta^*$ **then**
15         $\beta^* \leftarrow$ value of $\mathcal{P}$
16         $(z, x, y)^* \leftarrow (z, x, y)^{\mathcal{P}}$

Output: $(z, x, y)^*$

procedure (denoted by Construct $(z, x, y)^s_{iter}$ in Algorithm 1, line 6) is outlined in Algorithm 2.

Selecting p hubs

In order to determine the set H containing the p nodes that will be selected as hubs (lines 1 to 17 in Algorithm 2) we start by selecting q ($q < p$) nodes; afterwards we select the remaining $p - q$ nodes.

Initially, we compute the values

$$T_i = \sum_{j \in V} (t_{ij} + t_{ji}), \forall i \in V,$$

representing the total traffic originated and destined to each node $i \in V$. Defining $T = \sum_{i \in V} T_i$ we can compute the weights $w_i = \dfrac{T_i}{T} \in [0, 1]$ and such that $\sum_{i \in V} w_i = 1$. One node, say h, is randomly selected according to the values (probabilities) w_i and we set $H \leftarrow \{h\}$.

For selecting the following $q - 1$ hubs, we update T as $T = \sum_{i \in V \setminus H} T_i$ and the weights $w_i, \forall i \in V \setminus H$, accordingly, and proceed as before until we get $|H| = q$. This way of selecting the first q hubs gives advantage to the nodes having the largest amount of traffics.

The remaining $p - q$ hubs are now selected using a different criterion: for each $h \in H$, we compute $c^h_{\max} = \max_{i \in V \setminus H} \{\frac{c_{hi} + c_{ih}}{2}\}$ and $c^h_{\min} = \min_{i \in V \setminus H} \{\frac{c_{hi} + c_{ih}}{2}\}$, and define a threshold $\mathcal{T}$ as

$$\mathcal{T} = \frac{\sum_{h \in H} \left(c^h_{\max} + c^h_{\min}\right)}{|H|}.$$

Algorithm 2: Construct $(z, x, y)^s_{iter}$.

Input: q, λ

// Choose the first q hubs
foreach $i \in V$ **do**
  $T_i \leftarrow \sum_{j \in V}(t_{ij} + t_{ji})$
$H \leftarrow \emptyset$
while $|H| < q$ **do**
  $T \leftarrow \sum_{i \in V \setminus H} T_i$
  $w_i \leftarrow T_i/T.\ i \in V \setminus H$
  select $h \in V \setminus H$ in $V \setminus H$ according to the discrete probability distribution induced by the values w_i, $i \in V \setminus H$
  $H \leftarrow H \cup \{h\}$
// Choose the remaining $p - q$ hubs
while $|H| < p$ **do**
  foreach $h \in H$ **do**
    $c^h_{\min} \leftarrow \min_{i \in V \setminus H}\{(c_{hi} + c_{ih})/2\}$ $c^h_{\max} \leftarrow \max_{i \in V \setminus H}\{(c_{hi} + c_{ih})/2\}$
  $\mathcal{T} \leftarrow \lambda\ \frac{\sum_{h \in H}\left(c^h_{\max} + c^h_{\min}\right)}{|H|}$
  $D \leftarrow \left\{ i \in V \setminus H \mid \frac{\sum_{h \in H}(c_{ih} + c_{hi})}{|H|} \leq \mathcal{T} \right\}$
  select $h \in D$ according to a uniform distribution in $V \setminus H$
  $H \leftarrow H \cup \{h\}$
// Assign each terminal to a single hub
foreach $i \in V \setminus H$ **do**
  $bestMeasure(i) \leftarrow +\infty$;
  foreach $h \in H$ **do**
    Compute $Measure(i, h)$
    if $Measure(i,h) \leq bestMeasure(i)$ **then**
      $bestMeasure(i) \leftarrow Measure(i, h)$
      $h^* \leftarrow h$;
  $H_i \leftarrow \{h^*\}$
// Increase of the terminals assignments
foreach $i \in V \setminus H$ **do**
  repeat
    Compute $c(T_i)$
    Let $h^* = \arg\min_{h \in H \setminus H_i}\{a_{ih}\}$
    Compute $c'(T_i)$
    if $c'(T_i) + a_{ih^*} \leq c(T_i)$ **then**
      $H_i \leftarrow H_i \cup h^*$
    else
      continue $\leftarrow$ FALSE;
  until $(|H_i| = r)$ or (continue is FALSE) ;
// Traffic transportation
Solve optimally the routing problem to obtain $(z, x, y)^s_{iter}$

Output: $(z, x, y)^s_{iter}$

Now, the set of terminals with average transporting cost to and from the hubs less than or equal to the computed threshold is defined:

$$D = \left\{ i \in V \setminus H : \frac{\sum_{h \in H}(c_{ih} + c_{hi})}{|H|} \leq \mathcal{T} \right\}.$$

Set D would correspond to those nodes that, if they were selected as hubs, the transportation costs between them and the hubs already in H would be small. This is motivated by the fact that most of the traffics of the network will traverse the arcs among the hubs; thus the inter-hubs transportation costs are desirably small.

Once set D is defined, a first node h is selected at random from this set and inserted into H. If $|H| < p$, $c^h_{\max}$ and $c^h_{\min}$ are updated for all $h \in H$, $\mathcal{T}$ and D to choose another hub.

Although q has been thought as a search parameter, we think that for the values of p we consider in our computational experiments ($3 \leq p \leq 5$), a fixed value of 2 for q is reasonable, and this is the value we have used in all the experiments.

Allocating terminals to hubs

We are interested in finding a good feasible solution as earliest as possible. This is guaranteed by allocating each terminal to (at least) a single hub. Accordingly, in order to find an assignment of terminals to hubs, we assume as a starting point that all the outgoing/ingoing traffics from/to a terminal are transported through a single hub.

With this purpose, we considered three different measures to help making a decision about the allocation allocating of a terminal i to a hub h (*Measure(i,h)* in Algorithm 2).

The first measure is simply defined by the assignment cost a_{ih} of terminal i to hub h; the second one takes into account the unitary transportation cost. It is defined as follows:

$$\chi c_{ih} + \delta c_{hi} + \alpha \left(\frac{\sum_{\ell \in H \setminus \{h\}} c_{h\ell}}{|H| - 1} \right). \tag{33}$$

In the above expression, the third term is motivated by the fact that it is expected that most of the traffic will be routed through the hub subnetwork. In particular, we are considering the average unitary cost between hub h and all the other hubs. A drawback of measure (33) is that is does not take into account the cost associated with non-stop services.

The third measure attempted, overcomes this issue; it captures in a single value the promising features of the previous measures:

$$a_{ih} + \chi c_{ih} \sum_{j \in V} t_{ij} + \delta c_{hi} \sum_{j \in V} t_{ji} + T_i \alpha \left(\frac{\sum_{\ell \in H \setminus \{h\}} c_{k\ell}}{|H| - 1} \right). \tag{34}$$

Each terminal i is allocated to the hub k^* yielding the lowest value of the adopted measure. At this point, we have $|H_i| = 1, \forall i \in V \setminus H$.

After the initial allocation of terminals is performed, we check whether it compensates to increase the number of hubs to which each terminal is allocated.

For every terminal $i \in V \setminus H$, define $c(T_i)$ as the cost associated with the transportation of all the traffics to and from i via H_i and through the hub network induced by H. It is evident that is convenient allocating i to another hub $h^* \in H \setminus H_i$ if $c'(T_i) + a_{ih^*} \leq c(T_i)$, where $c'(T_i)$ represents the cost of transporting all the traffics to and from i in the network defined by H, where $H_i := H_i \cup h^*$ and all the other subsets $H_j, \forall j \neq i$, have not been modified. Since computing all the possibilities is too much time consuming, not all hubs $h^* \in H \setminus H_i$ are tried, only the hub h^* for which $a_{ih^*} = \min_{h \in H \setminus H_i}\{a_{ih}\}$. If $c'(T_i) + a_{ih^*} \leq c(T_i)$, we set $H_i := H_i \cup h^*$ and repeat the procedure for the same terminal i, while $|H_i| \leq r$; otherwise, we proceed with another terminal j.

The whole allocation procedure is summarized in lines 18 to 37 in Algorithm 2.

Traffics transportation

Once the set of hubs is known, as well as the allocation of terminals to hubs, the problem of finding the optimal route for the traffics among any pair of nodes is solved by computing the shortest paths and considering non-stop services as well. Note that for a given pair of nodes i and j, the optimal route for the traffic t_{ij} may be different to the one associated with traffic t_{ji}.

When the optimal routes for sending the traffics have been computed, we have a feasible solution to the UrApHMP-NSS, or, taking into account the context of the stochastic problem, to a single-scenario problem $\mathcal{P}_s$, $s \in S$.

3.1.2 Improving the solution

Two local search procedures are devised for improving the solution obtained using the constructive algorithm. They correspond to lines 7 and 8 in Algorithm 1, and are based on changing the subsets H_i, $i \in V \setminus H$, as well as on reducing their size, for some terminals i. These two procedures (denoted by $\mathsf{LS}_{\mathsf{change}}$ and $\mathsf{LS}_{\mathsf{reduce}}$ in Algorithm 2), are described next.

Changing the assignments of terminals to hubs (LS_{change})

Consider a terminal $i \in V \setminus H$ as well as the set of hubs, H_i, to which it is currently allocated. The procedure $\mathsf{LS}_{\mathsf{change}}$ iteratively explores the possibility of replacing one hub $\ell \in H_i$ by another hub $\ell' \in H \setminus H_i$. It should be noticed that in order to check whether such a move "improves" the current solution, we need to recompute the cost associated with the transportation of all the traffics involving node i, and not just those transported through hub ℓ. We start by computing

$$\mathcal{R}(i) = \frac{\frac{1}{|H_i|}\sum_{k \in H_i} a_{ik}}{T_i}, \quad i \in V \setminus H.$$

Note that the numerator of $\mathcal{R}(i)$ is the average allocation cost of i to its associated hubs. Hence, $\mathcal{R}(i)$ is a ratio between that cost and all the traffic to and from i. Therefore, we obtain a unitary traffic average cost involving terminal i.

The values of $\mathcal{R}(i)$ are now sorted non-increasingly. This induces a sequence for nodes in $V \setminus H$ that we denote by $(i_1, \ldots, i_{|V \setminus H|})$. The improvement procedure takes the terminals iteratively, according to this sequence. For each terminal $i \in V \setminus H$, all the possible pairs (ℓ, ℓ'), with $\ell \in H_i$ and $\ell' \in H \setminus H_i$ are tested. The pseudo-code associated with this procedure is detailed in Algorithm 3.

Algorithm 3: $\mathsf{LS}_{\mathsf{change}}$

```
Input: (z, x, y)^s_iter
continue ← TRUE;
while continue is TRUE do
    continue ← FALSE
    compute R(i), ∀i ∈ V \ H
    find a sequence (i_1, ..., i_|V\H|) induced by sorting the values R(i)
    (i ∈ V \ H) non-increasingly
    foreach j = 1, ..., |V \ H| do
        foreach ℓ ∈ H_{i_j} do
            foreach ℓ' ∈ H \ H_{i_j} do
                if (cost using ℓ') < (cost using ℓ) then
                    replace ℓ by ℓ' in H_{i_j}
                    continue ← TRUE
Output: (z, x, y)^s_iter
```

Reducing the number of allocations to hubs (LS_{reduce})

This procedure, which is detailed in Algorithm 4, aims at reducing the cardinality of some subsets H_i, $|H_i| \geq 2$, $i \in V \setminus H$, in case we conclude that this is advantageous from a cost perspective. In order to accomplish this, we compute

$$\mathcal{R}'(i) = \frac{\sum_{k \in H_i} a_{ik}}{T_i}$$

for all terminals $i \in V \setminus H$, which is the average allocation cost of i per traffic unit. As before, a sequence of terminals is induced by sorting the values $\mathcal{R}'(i)$ non-increasingly. The terminals are them analyzed according to this sequence.

For a terminal $i \in V \setminus H$ such that $|H_i| \geq 2$ we check the possibility of decreasing $|H_i|$ by choosing the hub $\ell \in H_i$ that appears the least in the paths associated with traffics T_i (that we denote by $Paths(i)$). If the total cost decreases by removing the allocation of node i to hub ℓ, then we do so. In order to check whether the costs decreases, we only need to recompute the paths in Paths(i^*) that make use of hub ℓ. Note that removing ℓ from H_i increases the transportation cost but decreases the total cost by $a_{i\ell}$. This process, that attempts to remove one allocation, is performed for all terminals in $V \setminus H$. Afterwards, $\mathcal{R}'(i)$ is recomputed for the terminals i for which the cardinality of H_i has been reduced and the procedure restarts but considering only such terminals. The process continues until no decrease in the cost can be achieved.

Algorithm 4: $\mathsf{LS}_{\mathsf{reduce}}$

Input: $(z, x, y)^s_{iter}$

continue $\leftarrow$ TRUE;
while continue *is* TRUE **do**
    continue $\leftarrow$ FALSE
    Compute $\mathcal{R}'(i), \forall i \in V \setminus H : |H_i| \geq 2$
    find a sequence $(i_1, \ldots, i_L)$ induced by sorting non-increasingly the values $\mathcal{R}'(i)$, $i \in V \setminus H : |H_i| \geq 2$
    foreach $j = 1, \ldots, L$ **do**
        Select $\ell \in H_{i_j}$ as the hub used less times in Paths(i_j)
        Compute $Cost$ as the cost of routing traffics T_{i_j} using H_{i_j}
        Compute $\overline{Cost}$ as the cost of routing traffics T_{i_j} using $H_{i_j} \setminus \{\ell\}$
        if $\overline{Cost} < Cost + a_{i_j \ell}$ **then**
            $H_{i_j} \leftarrow H_{i_j} \setminus \{\ell\}$
            continue $\leftarrow$ TRUE

Output: $(z, x, y)^s_{iter}$

3.2 Constructing a feasible solution to the stochastic problem

In this section we introduce a mechanism that allows the combination of attributes from solutions to the single-scenario problems in order to build a feasible solution to the overall problem $\mathcal{P}$

We denote by $\overline{\beta} = (\beta_1, \ldots, \beta_{|S|})$ the vector containing the value of the best solution found for the single-scenario problems, $\mathcal{P}_s$, $s \in S$. Additionally, let z^s_{ik} be the values of the z-variables in the solution found for $\mathcal{P}_s$, $s \in S$.

Furthermore, consider the $|V| \times |V|$ matrix, denoted by $\mathcal{U}$, whose generic entry is $u_{ij} = \pi_1 z^1_{ij} + \pi_2 z^2_{ij} + \ldots + \pi_{|S|} z^{|S|}_{ij} = \sum_s \pi_s z^s_{ij}$, $i, j \in V$. Let U_j be the j-th column of $\mathcal{U}$ ($j \in V$). Taking into account that $z_{ij} = 1$ if terminal i is assigned to hub j, and u_{ij} contains the same information averaged across the different scenarios, when we examine matrix $\mathcal{U}$ we have to consider that each row i represents a terminal node and each column j a potential hub.

In order to quantify the "attractiveness" of a node to become a hub in the solution for the overall problem $\mathcal{P}$, we compute the marginal vector, $\overline{u}$, resulting from summing all rows of $\mathcal{U}$, i.e., a vector whose generic component is $\overline{u}_j = \sum_{i \in V} u_{ij}$, $j \in V$.

The starting point for building a feasible solution to $\mathcal{U}$ is to randomly select a scenario s^* according to the probabilities $\pi_1, \ldots, \pi_{|S|}$. Then, a process inspired on the Path Relinking methodology (Glover and Laguna 1997) that gradually transforms an initiating solution by incorporating to it attributes of some guiding solutions is devised. In particular, we propose here to consider the attributes of the best solutions obtained in each scenario averaged according to their probability. This information is contained in matrix $\mathcal{U}$. The steps of the process can be summarized as follows:

(i) Consider the set of p hubs, H_{s^*}, in the feasible solution obtained for $\mathcal{P}_{s^*}$.

(ii) Find the p indices associated with the larges values of $\overline{u}_j$ for $j \in V \setminus H_{s^*}$.

(iii) Denote by J=$(j_1, ..., j_p)$ a sequence of the indices found in (iii) resulting from sorting the corresponding values of $\overline{u}_j$ non-increasigly.

(iv) Set $\ell = j_1$.

(v) For each $h \in H_{s^*}$,

(a) denote by $H'_{s^*}(h)$ the set of hubs resulting from replacing h by ℓ;

(b) for each $k \in H'_{s^*}(h)$ and for each $i \in V$, if $u_{ik} > 0$ set $z_{ik} = 1$, i.e., allocate node i to hub k.

(c) For each node $i \in V$ check whether the allocations set for the terminal are feasible. If not, repair them. Three cases can be distinguished:

c1. terminal i has been assigned to more than r hubs (i.e., $|H_i| > r$). In this case, the $|H_i| - r$ assignments of i to the hubs in $H'_{s^*}(h)$ having the smallest values u_{ij} are discarded by setting the corresponding z-variables to 0.

c2. terminal i has not been assigned to any hub (i.e., $|H_i| = 0$). This may be the case of a node i assigned only to hub h. The assignment of i to the existing hubs is evaluated at each scenario using the measure (34) described in Section 3.1.1, and the number of times a hub is identified as the best for i is calculated. The hub k appearing most is the one selected for allocating i to.

c3. Node i is a hub and was allocated to other hubs. In this case, all the assignments of i to other hubs are removed.

(d) consider $H'_{s^*}(h)$ as the set of operating hubs together with the allocations z_{ik} resulting from (b) and (c). For each scenario $s \in S$ do the following:

1. compute the cost under scenario s, if we implement such feasible network design for that particular scenario. Denote that cost by $\hat{c}_h^s$.
2. compute the value $r_h^s = \frac{\hat{c}_h^s - \beta_s}{\beta_s}$. This value represents a sort of relative "regret" if the network design implemented in scenario s is the one induced by $H'_{s^*}(h)$ instead of the best network design known so far for that scenario. (Recall that the incumbent upper bound on the optimal value for scenario s is β_s.).
3. Compute $r_h^{\max} = \max_{s \in S}\{r_h^s\}$ as the maximum relative "regret" across all scenarios if we take the network design induced by $H'_{s^*}(h)$ with the allocations resulting from (b) and (c).

(vi) p sets of p nodes result from (v), since ℓ is replacing in turn every hub $h \in H_{s^*}$. The next step is to compare those sets of hubs and decide for one of them. To do so, we use the values $r_h^{\max}$ computed in (v)(d)(3.). In particular, we select the hub set $H_{s^*}(h')$ such that

$$h' \in \arg\min_{h \in H_{s^*}} \{r_h^{\max}\}.$$

(vii) Setting the network design induced by $H_{s^*}(h')$ as explained above for the overall problem $\mathcal{P}$, we can now easily solve the resulting transportation problem and eventually get a complete feasible solution to the stochastic problem. Accordingly, we should update the best incumbent solution for $\mathcal{P}$ if the cost found is smaller than the cost of the incumbent (in case some already exists).

(viii) This mechanism proceeds now by setting $\ell = j_2$ and analyzing the $p-1$ ways of replacing a column in $H_{s^*} \setminus \{j_1\}$ by ℓ.

(ix) When, finally, we set $\ell = j_p$ there is only one possible replacement in $H_{s^*} \setminus \{j_1, \dots, j_{p-1}\}$ which is the only one attempted.

(x) The best solution for $\mathcal{P}$ is updated each time a new solution with less cost is obtained.

We note that the above mechanism can (and should) be repeated several times since the starting point is randomized.

The steps (i)–(x) above, define the heuristic procedure we propose for obtaining feasible solutions to the stochastic UrApHMP-NSS.

Note that if in step (v)(d) we find a negative "regret" for some scenario, this means that we have found a feasible solution under that scenario better than the best one known so far. In this case, we should update the corresponding value of β_s in accordance to the new upper bound found.

It is worth noticing that the above presented scheme is quite flexible. In fact, it is modular in the sense that some parts can simply be replaced without the need of changing the global structure. For instance, if a different approach is considered for tackling the single-scenario problems, the above structure can be adopted exactly as presented.

4 Computational experiments

In this section we describe the characteristics of the instances tested and the computational results obtained with the above proposed algorithm.

4.1 Test instances

Since the Stochastic UrApHMP-NSS has been introduced in this paper, there are no benchmark instances available. Hence, we have generated a set of instances in order to evaluate the behavior of our new method.

In order to generate instances for the Stochastic UrApHMP-NSS we have followed a similar reasoning as the one proposed by Alumur et al. (2012) taking as starting point the CAB25 data set introduced by O'Kelly (1987).

From the original CAB data file, which contains information on distances and traffics between 25 major cities in the USA, we have generated a total of 74 instances with:

- $|V| \in \{15, 20, 25\}$;
- $p \in \{3, 4, 5\}$;
- $r \in \{2, \dots, p-1\}$.

As it is often done when using the CAB data, the traffics are scaled (by dividing them by the total traffic) so that the total demand is equal to 1. The values for parameters χ and δ, representing the unit rates for collection (origin-hub) and distribution (hub-destination), are set to 1. Parameter α (the unit rate for transfer hub-hub) takes values in the set $\{0.2, 0.4, 0.6, 0.8, 1.0\}$. The other parameters are generated according to the following:

- From the traffics τ_{ij} in the original CAB instances, we have generated traffics t_{ij} for each scenario as follows:
  - (i) a number w is randomly generated in the interval $[1, 100]$;
  - (ii) if $w \leq 66$, t_{ij} is randomly selected in the interval $[0.01\tau_{ij}, 5\tau_{ij}]$; otherwise, t_{ij} is randomly selected in $]5\tau_{ij}, 10\tau_{ij}]$.
- The assignment costs, a_{ik}, $i, k \in V$ have been randomly generated in the interval $[10 \log \sum_{j \in V} t_{ij}, 20 \log \sum_{j \in V} t_{ij}]$.
- For each scenario, the costs c_{ij}, have been randomly generated in the interval $[0.5\gamma_{ij}, 1.5\gamma_{ij}]$, where γ_{ij} $(i, j \in V)$ represent the original costs.
- The non-stop transportation costs d_{ij} have been randomly selected in the interval $[0.2(\chi + \alpha + \delta)\gamma_{ij}, 0.8(\chi + \alpha + \delta)\gamma_{ij}]$.
- The costs b_{ij} have been randomly chosen in $[10 \log(t_{ij} + t_{ji}), 20 \log(t_{ij} + t_{ji})]$.

Nine scenarios have been considered for all the instances, and the probabilities π_s associated with the scenarios have been randomly generated.

4.2 Computational results

In this section we report the computational results obtained with the method we proposed in Section 3 for solving the Stochastic UrApHMP-NSS. The procedure has been implemented in C using GCC 4.8.4 with optimization flag -O3. The results of the proposed method reported in this section have been obtained with an Intel core i7–3770 at 3.40GHz using a single thread and 16GB of RAM, under Ubuntu 14.04.03 GNU/Linux – 64 bits operating system, while those corresponding to CPLEX have been obtained with a SGI Altix UV 1000 system with 20 processors (out of 64) at 2.67 GHz, 120 cores, and 120GB of RAM.

A relevant component in the heuristic procedure described in Section 3.2 is the algorithm proposed for the deterministic (single-scenario) problems. For this reason, we start by analyzing the quality of the feasible solutions obtained by that algorithm. The information can be found in Table 1, where each row contains average results for the 9 single-scenario instances generated for each combination of n, p, r, α.

In this table, the column headed with "CPLEX" contains the average CPU time (seconds) required to solve the instances to optimality using this solver. We then observe 3 groups of columns (4 columns each). The first group contains the results (percentage deviation w.r.t. optimum and CPU time when 100 or 1000 iterations were performed) for the feasible solutions provided by Algorithm 2. The second and third groups of columns contain the results obtained when

				CPLEX	Construct				Construct + $\mathsf{LS}_{\mathsf{change}}$				Construct + $\mathsf{LS}_{\mathsf{change}}$ + $\mathsf{LS}_{\mathsf{reduce}}$			
				Opt.	100 iter		1000 iter		100 iter		1000 iter		100 iter		1000 iter	
n	p	r	α	CPU	Dev(%)	CPU	Dev(%)	CPU	Dev(%)	CPU	Dev(%)	CPU	Dev(%)	CPU	Dev(%)	CPU
15	4	2	1.00	2.17	56.1	0.00	53.6	0.03	53.6	0.02	50.9	0.18	4.2	0.02	1.5	0.19
15	4	2	0.80	2.32	56.8	0.00	52.6	0.03	55.1	0.02	51.4	0.17	3.8	0.02	1.7	0.18
15	4	2	0.60	2.22	57.6	0.00	51.2	0.03	56.4	0.02	50.2	0.17	3.7	0.02	1.6	0.17
15	4	2	0.40	1.36	58.3	0.00	51.1	0.03	56.8	0.02	50.1	0.16	4.1	0.02	1.7	0.15
15	4	2	0.20	1.22	59.6	0.00	53.5	0.03	58.9	0.01	52.9	0.15	4.2	0.01	1.3	0.15
20	4	2	1.00	56.68	39.1	0.00	36.1	0.05	33.1	0.04	30.9	0.40	4.5	0.04	2.2	0.39
20	4	2	0.80	64.02	39.3	0.01	36.4	0.05	34.6	0.04	31.7	0.37	3.3	0.04	1.6	0.36
20	4	2	0.60	44.91	42.0	0.01	37.0	0.05	38.3	0.04	34.1	0.35	4.1	0.03	1.4	0.34
20	4	2	0.40	26.33	44.0	0.01	36.7	0.05	41.7	0.03	35.4	0.31	3.8	0.03	1.2	0.31
20	4	2	0.20	12.13	44.4	0.01	38.2	0.05	42.7	0.03	37.1	0.28	3.6	0.03	0.8	0.28
20	5	2	1.00	66.94	36.5	0.01	34.4	0.05	29.9	0.05	28.6	0.52	3.7	0.05	2.3	0.50
20	5	2	0.80	64.94	37.8	0.01	34.4	0.05	32.7	0.05	30.1	0.49	3.4	0.05	1.4	0.48
20	5	2	0.60	36.77	41.7	0.01	37.9	0.05	37.5	0.05	33.8	0.45	4.7	0.05	1.8	0.44
20	5	2	0.40	16.69	46.0	0.00	39.5	0.05	43.5	0.04	37.1	0.41	5.8	0.04	2.1	0.40
20	5	2	0.20	8.75	49.7	0.00	42.1	0.05	48.6	0.04	41.3	0.37	6.1	0.03	2.1	0.35
25	5	2	1.00	701.98	40.7	0.01	37.0	0.07	34.3	0.09	32.6	0.87	4.6	0.09	2.6	0.86
25	5	2	0.80	454.14	43.5	0.01	40.8	0.07	38.4	0.08	36.5	0.81	4.2	0.08	2.9	0.82
25	5	2	0.60	116.36	47.0	0.01	43.1	0.07	43.6	0.08	40.3	0.76	4.3	0.08	2.8	0.76
25	5	2	0.40	56.85	50.2	0.01	46.4	0.07	47.8	0.07	44.7	0.69	3.9	0.07	2.3	0.69
25	5	2	0.20	28.37	54.4	0.01	49.8	0.07	53.1	0.06	49.0	0.63	4.6	0.06	2.9	0.62
Average				88.26	47.2	0.01	42.6	0.05	44.0	0.04	39.9	0.43	4.2	0.04	1.9	0.42

Table 1: Comparison of the heuristic for the UrApHMP-NSS with CPLEX on some CAB instances.

Algorithm 3 (second group) and Algorithm 3 followed by 4 (third group) were performed for improving the initial feasible solution.

Observing Table 1 we conclude that the complete procedure (Algorithms 2–4) is a very efficient tool for obtaining high-quality solutions to the (deterministic) UrApHMP-NSS. In particular, we observe that running the whole procedure 1000 times renders extremely sharp upper bounds at the expenses of a negligible increase in the computational effort. The results obtained indicate that Algorithm 1 is a good element to put at the core of the overall procedure for the stochastic problem.

			CPLEX	Heur 1000		Heur 2000	
n	p	# inst	CPU	Dev (%)	CPU	Dev (%)	CPU
15	3	5	271.5	1.0	9.5	1.0	18.8
15	4	10	65.3	3.2	18.2	2.2	36.1
15	5	15	44.7	1.9	29.6	1.3	58.9
20	3	5	16261.2	1.6	18.8	1.2	37.2
20	4	10	10479.9	2.2	37.0	1.4	72.7
20	5	15	10080.7	2.9	61.9	1.8	123.2
25	5	14⋆	129615.7	3.9	104.3	3.8	207.2
Summary		74	29116.4	2.6	47.7	2.0	94.6

Table 2: Computational results on the 74 CAB instances.

Table 2 reports the results obtained using the procedure developed in Section 3.2 for the Stochastic UrApHMP-NSS. The information presented in each row corresponds to the average values obtained for all the instances with the characteristics described. First two columns show the number of nodes of the group of instances and the number of hubs to be open, respectively. As mentioned before, one instance was generated for each value of p, $r \in \{2, \ldots, p-1\}$, and $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. The number of instances in each group is shown in column "# inst".

For $n = 25$ and $p = 5$ we report results for 14 out of the 15 CAB instances considered. This is marked with "⋆". The omitted instance (corresponding to $r = 4$ and $\alpha = 0.6$), has been removed from the table since CPLEX was unable of providing even a lower bound after 5 days of computing time.

All the instances reported have been solved to optimality with CPLEX and heuristically with our algorithm.

The CPU time (seconds) required by CPLEX as well as by two alternatives for our method (1000 and 2000 iterations in total) and the average percentage deviation with respect to the optimal solution are reported in columns "CPU" and "Dev (%)".

The results obtained give strong evidence to the high efficiency of the new heuristic proposed. In particular, we can observe an average deviation of 2.6% after 1000 runs of the procedure in less than 50 seconds (on average), while the average deviation reduces to 2% when 2000 runs are performed which is accomplished in less that 100 seconds. Additionally, it is worth noticing that for 1000 runs, a deviation smaller than 1% was obtained for 15 out of the 74 instances, and a maximum deviation of 7.1% was observed; when 2000 runs were performed, the method led to a deviation smaller than 1% in 23 out of the 74 instances, while the maximum deviation obtained was 5.5%. Note that the average CPU time required by CPLEX for solving the instances in the last group is approximately 36 hours, while in some of the instances it takes up to 5 days to find the optimal solution with the SGI Altix UV 1000 system. The detailed results (for each instance tested) are reported in Table 4 in the Appendix.

One important element in the heuristic algorithm proposed in Section 3.2 for the Stochastic UrApHMP-NSS concerns the vector $\beta_1, \ldots, \beta_{|S|}$ that contains the best known upper bound for each of the $|S|$ single-scenario problems. During the process, the changes performed in the solutions to the stochastic problem may allow finding better solutions for some single-scenario problems, which leads to changes in the above vector. In Figure 1 we represent the evolution of the 9 βs, each one for one scenario, considered in the instance with $n = 25$, $p = 5$, $r = 2$, and $\alpha = 1.00$. In this figure, we also represent the evolution of the value of the best known solution to the stochastic problem. The Figure shows that changing the values of the β is something that the process takes advantage from. Moreover, at least for this instance, we can observe larger improvements in an earlier stage of the process, which also indicates that the overall procedure seems to be effective not only when it comes to finding a good solution to the stochastic problem, but also in terms of sharpening the best upper bounds known for the single-scenario problems.

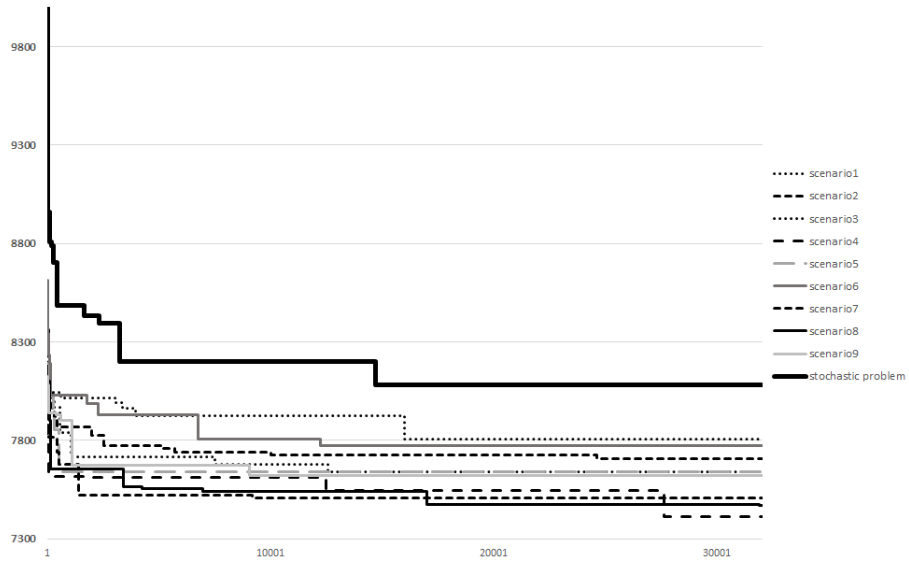

Figure 1: Beta evolution

Finally, we conclude the analysis of the results by evaluating the relevance of considering the stochastic modeling framework proposed for the UrApHMP-NSS instead of using a simplified deterministic model. We accomplish this analysis by computing the so-called Expected Value of Perfect Information (EVPI) that quantifies the amount that the decision maker would be willing to pay to access the perfect future information. A high EVPI indicates that the decision maker perceives as quite relevant having access to the perfect information which indicates that uncertainty may be a relevant factor in the problem.

The EVPI is obtained as follows:

- First, the Wait-and-See value is computed according to $WS = \sum_{s\in S} \pi_s \mathcal{V}_s$, where $\mathcal{V}_s$ is the optimal value of problem $\mathcal{P}_s$.
- Second, the stochastic problem $\mathcal{P}$ is solved to optimality. Let $\mathcal{V}$ its value.
- Finally the EVPI is computed as: EVPI $= \mathcal{V} - WS$.

n	p	r	α	Wait-and-see optimal value	Optimal value of the Stochastic UrApHMP-NSS	Relative EVPI
15	4	2	0.2	2386.3	2393.1	0.28%
15	4	2	0.4	2533.1	2542.0	0.35%
15	4	2	0.6	2678.4	2690.8	0.46%
15	4	2	0.8	2795.4	2839.6	1.56%
15	4	2	1.0	2878.4	2957.6	2.68%
20	4	2	0.2	5078.3	5135.8	1.12%
20	4	2	0.4	5584.6	5705.3	2.12%
20	4	2	0.6	6050.5	6214.0	2.63%
20	4	2	0.8	6430.6	6654.4	3.36%
20	4	2	1.0	6672.3	7037.4	5.19%
20	5	2	0.2	4458.4	4499.1	0.91%
20	5	2	0.4	5053.2	5126.4	1.43%
20	5	2	0.6	5619.3	5753.8	2.34%
20	5	2	0.8	6077.5	6304.2	3.60%
20	5	2	1.0	6366.5	6696.2	4.92%
25	5	2	0.2	5256.0	5313.3	1.08%
25	5	2	0.4	5877.9	5961.2	1.40%
25	5	2	0.6	6443.6	6569.8	1.92%
25	5	2	0.8	6974.6	7159.2	2.58%
25	5	2	1.0	7469.8	7735.8	3.44%
Average				5134.2	5264.5	2.17%

Table 3: Exact Expected Value of the Perfect Information for some CAB instances.

A more informative measure is the relative EVPI: $100\frac{\mathcal{V}-WS}{\mathcal{V}}$, since the corresponding result (in percentage) ignores the magnitude of the values involved in the problem. The results obtained for the test instances we have considered can be observed in Table 3.

A first aspect emerging from this table is the increase of the relative EVPI with α. This is not surprising since small values of α induce small inter-hub costs, which makes the impact of the decisions associated with the hubs smaller and thus the relevance of the uncertainty also decreases.

Overall, we observe percentages that are always positive and ranging up to 5.19%. Taking into account that we are working with fairly small instances (up to 25 nodes) this values show that considering stochasticity in our problem may be of great relevance.

5 Conclusions

In this paper we have investigated a stochastic version of the r-Allocation p-Hub Median Problem with Non-stop Services. We have started by extending the already existing r-Allocation p-Hub Median Problem in order to capture non-stop services. Afterwards we have developed a stochastic programming modeling framework for the problem. Due to the difficulty in solving the problem to optimality, we have derived a heuristic approach for the new stochastic problem. A side contribution is the development of a heuristic approach for the deterministic (single-scenario) version of the problem. We have performed computational tests using instances generated from the well-known CAB data set. The results show the effectiveness of our new heuristic for obtaining high-quality feasible solutions to the problem with a small CPU time.

One important aspect of our heuristic is its modularity. For instance, in case a different algorithm is devised for finding feasible solutions to the single-scenario problems, the methodology described in Section 3.2 is still valid. Another possibility is to change the attribute matrix $\mathcal{U}$. Again, the procedure would be valid as presented.

The high quality of the results obtained in this work encourages the application of the same type of heuristic methodology to other stochastic hub location problems and even to consider more comprehensive models from a practical point of view.

Acknowledgements

This work was supported by the Spanish Ministerio de Economía y Competitividad and Fondo Europeo de Desarrollo Regional (FEDER) (projects TIN-2015-65460-C02-01(MINECO/FEDER), MTM-2015-68097-P(MINECO/FEDER), PhD. grant BES-2013-064245, and short term visit funding EEBB-I-15-09778) and by the Generalitat Valenciana (project Prometeo 2013/049). Francisco Saldanha-da-Gama was supported by the Portuguese Science Foundation (FCT—Fundação para a Ciência e Tecnologia) under the project UID/MAT/04561/2013 (CMAF-CIO/FCUL). This support is gratefully acknowledged.

References

S. Abdinnour-Helm. A hybrid heuristic for the uncapacitated hub location problem. *European Journal of Operatiomal Research*, 106:489–499, 1998.

S. Abdinnour-Helm and M.A. Venkataramanan. Solution approaches to hub location problems. *Annals of Operations Research*, 78:31–50, 1998.

M. Albareda-Sambola, A. Alonso-Ayuso, L.F. Escudero, E. Fernández, and C. Pizarro. Fix-and-relax-coordination for a multi-period location–allocation problem under uncertainty. *Computers & Operations Research*, 40:2878–2892, 2013.

A. Alonso-Ayuso, L.F. Escudero, and M.T. Ortuño. BFC, a branch-and-fix coordination algorithmic framework for solving stochastic 0–1 programs. *European Journal of Operational Research*, 151:503–519, 2003.

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

S. Alumur and E.Z. Serper. The design of capacitated intermodal hub networks with different vehicle types. *Transportation Research B*, 86:51–65, 2016.

S. Alumur, S. Nickel, and F. Saldanha-da-Gama. Hub location under uncertainty. *Transportation Research B*, 46:529–543, 2012.

T. Aykin. Lagrangian relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, 79:501–523, 1994.

T. Aykin. The hub location and routing problem. *European Journal of Operational Research*, 83:200–219, 1995a.

T. Aykin. Networking policies for hub-and-spoke systems with application to the air transportation system. *Transportation Science*, 29:201–221, 1995b.

L. Bianchi, M. Dorigo, L.M. Gambardella, and W.J. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, 8:239–287, 2009.

R. Bollapragada, J. Camm, U.S. Rao, and J. Wu. A two-phase greedy algorithm to locate and allocate hubs for fixed-wireless broadband access. *Operations Research Letters*, 33:134–142, 2005.

H. Calık, S.A. Alumur, B.Y. Kara, and O.E. Karasan. A tabu-based heuristic for the hub covering problem over incomplete hub networks. *Computers & Operations Research*, 36(12):3088–3096, 2009.

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

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

J. Chen. A hybrid heuristic for the uncapacitated single allocation hub location problem. *Omega*, 35:211–220, 2007.

I. Contreras. Hub location problems. In G. Laporte, S. Nickel, and F. Saldanha-da-Gama, editors, *Location Science*, pages 311–344. Springer, 2015.

I. Contreras, J.A. Díaz, and E. Fernández. Lagrangean relaxation for the capacitated hub location problem with single assignment. *OR Spectrum*, 31: 483–505, 2009.

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

I. Contreras, J.A. Díaz, and E. Fernández. Branch and price for large-scale capacitated hub location problems with single assignment. *INFORMS Journal in Computing*, 23:41–55, 2011b.

C.B. Cunha and M.R. Silva. A genetic algorithm for the problem of configuring a hub-and-spoke network for a LTL trucking company in Brazil. *European Journal of Operational Research*, 179:747–758, 2007.

Ch. Dillenberger, L.F. Escudero, A. Wollensak, and W. Zang. On practical resource allocation for production planning and scheduling with period overlapping setups. *European Journal of Operational Research*, 75:275–286, 1994.

H.A. Eiselt and V. Marianov. A conditional p-hub location problem with attraction functions. *Computers & Operations Research*, 36:3128–3135, 2009.

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

A.T. Ernst and M. Krishnamoorthy. Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, 86: 141–159, 1999.

L.F. Escudero and J. Salmerón. Fix-and-relax partitioning. An algorithmic framework for large-scale resource constrained project selection and scheduling. *Annals of Operations Research*, 140:163–188, 2005.

F. Glover and M. Laguna. *Tabu search.* Kluwer, Norwell, MA, 1997.

A. Ilić, D. Urošević, J. Brimberg, and N. Mladenović. A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, 206:289–300, 2010.

J. Klincewicz. Avoiding local optima in the p-hub location problem using tabu search and GRASP. *Annals of Operations Research*, 40:283–302, 1992.

J. Kratica, Z. Stanimirović, D. Tošić, and V. Filipović. Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, 182:15–28, 2007.

J. Kratica, M. Milanović, Z. Z. Stanimirović, and D. Tošić. An evolutionary-based approach for solving a capacitated hub location problem. *Applied Soft Computing*, 11:1858–1866, 2011.

J. Kratica, Z. Stanimirović, D. Tošić, and V. Filipović. Genetic algorithm for solving uncapacitated multiple allocation hub location problem. *Computing and Informatics*, 24:415–426, 2012.

A. Lüer-Villagra and V. Marianov. A competitive hub location and pricing problem. *European Journal of Operational Research*, 231:734–744, 2013.

V. Marianov and D. Serra. Location models for airline hubs behaving as M/D/c queues. *Computers & Operations Research*, 30:983–1003, 2003.

V. Marianov, D. Serra, and C. ReVelle. Location of hubs in a competitive environment. *European Journal of Operational Research*, 114:363–371, 1999.

R. Martí, Á. Corberán, and J. Peiró. Scatter search for an uncapacitated p-hub median problem. *Computers & Operations Research*, 58:53–66, 2015.

T. Meyer, A. T. Ernst, and M. Krishnamoorthy. A 2-phase algorithm for solving the single allocation p-hub center problem. *Computers & Operations Research*, 36(12):3143–3151, 2009.

S. Nickel, A. Schobel, and T. Sonneborn. Hub location problems in urban traffic networks. In J. Niittymaki and M. Pursula, editors, *Mathematics Methods and Optimization in Transportation Systems*, pages 1–12. Kluwer Academic Publishers, 2001.

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

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

J. Peiró, Á. Corberán, and R. Martí. GRASP for the uncapacitated r-allocation p-hub median problem. *Computers & Operations Research*, 43:50–60, 2014.

H. Pirkul and D.A. Schilling. An efficient procedure for designing single allocation hub and spoke systems. *Management Science*, 44:S235–S242, 1998.

M. Randall. Solution approaches for the capacitated single allocation hub location problem using ant colony optimization. *Computational Optimization and Applications*, 39:239–261, 2008.

I. Rodríguez-Martín and J.J. Salazar-González. An iterated local search heuristic for a capacitated hub location problem. *Lecture Notes in Computer Science*, 4030:70–81, 2006.

K.E. Rosing and C.S. ReVelle. Heuristic concentration: Two stage solution construction. *European Journal of Operational Research*, 97:75–86, 1997.

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

M.R. Silva and C.B. Cunha. New simple and efficient heuristics for the uncapacitated single allocation hub location problem. *Computers & Operations Research*, 36:3152–3165, 2009.

T. Sim, T.J. Lowe, and B.W. Thomas. The stochastic p-hub center problem with service-level constraints. *Computers & Operations Research*, 36(12): 3166–3177, 2009.

K. Smith, M. Krishnamoorthy, and M. Palaniswami. Neural versus traditional approaches to the location of interacting hub facilities. *Location Science*, 4: 155–171, 1996.

Z. Stanimirović. A genetic algorithm approach for the capacitated single allocation p-hub median problem. *Computing and Informatics*, 29:117–132, 2010.

C.S. Sung and H.W. Jin. Dual-based approach for a hub network design problem under non-restrictive policy. *European Journal of Operational Research*, 132: 88–105, 2001.

B. Wagner. An exact solution procedure for a cluster hub location problem. *European Journal of Operational Research*, 178:391–401, 2007.

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

H. Yaman and G. Carello. Solving the hub location problem with modular link capacities. *Computers & Operations Research*, 32:3227–3245, 2005.

Appendix

In this appendix we detail the results reported in Table 2. In particular, for each combination n, p, r, α we report the results obtained using CPLEX to solve the instance to optimality and also the results obtained after performing the new heuristic proposed considering 1000 and 2000 runs. The information is gathered in Table 4.

Table 4: Detailed results for the CAB instances.

				CPLEX		Heur 1000			Heur 2000		
n	p	r	α	Value	CPU (scs)	Value	Dev (%)	CPU (scs)	Value	Dev (%)	CPU (scs)
			0.2	2782.5	50.6	2842.3	2.1	8.5	2842.3	2.1	16.9
			0.4	2920.2	90.4	2962.9	1.5	9.0	2958.0	1.3	17.8
15	3	2	0.6	3057.8	362.6	3057.9	0.0	9.6	3057.9	0.0	18.9
			0.8	3187.1	668.4	3187.1	0.0	10.0	3187.1	0.0	19.8
			1.0	3233.6	185.7	3278.8	1.4	10.5	3278.8	1.4	20.6
			0.2	2393.1	36.4	2514.4	5.1	12.9	2410.8	0.7	25.6
			0.4	2542.0	44.3	2542.0	0.0	13.9	2542.0	0.0	29.4
15	4	2	0.6	2690.8	77.0	2754.7	2.4	15.0	2754.7	2.4	29.5
			0.8	2839.6	44.9	2941.2	3.6	15.8	2845.6	0.2	31.3
			1.0	2957.6	187.0	3077.2	4.0	16.6	3048.8	3.1	32.8
			0.2	2393.1	35.7	2522.8	5.4	19.1	2522.8	5.4	38.0
			0.4	2542.0	41.6	2542.0	0.0	20.5	2542.0	0.0	40.9
15	4	3	0.6	2690.8	30.9	2807.5	4.3	21.7	2786.3	3.5	43.1
			0.8	2839.6	35.2	2961.7	4.3	23.5	2961.7	4.3	44.7
			1.0	2957.6	119.8	3029.9	2.4	23.2	3029.9	2.4	46.0
			0.2	2119.8	29.8	2128.7	0.4	15.9	2128.7	0.4	31.9
			0.4	2274.8	29.3	2274.8	0.0	17.3	2274.8	0.0	34.3
15	5	2	0.6	2429.8	27.8	2459.7	1.2	20.8	2459.7	1.2	51.4
			0.8	2584.8	30.6	2584.8	0.0	20.2	2584.8	0.0	41.3
			1.0	2733.8	61.4	2760.6	1.0	21.1	2736.2	0.1	42.0
			0.2	2119.8	32.8	2194.8	3.5	29.4	2161.2	2.0	56.7

Continued on next page

Table 4 – *Detailed results for the CAB instances — continued from previous page*

				CPLEX		Heur 1000			Heur 2000		
n	p	r	α	Value	CPU (scs)	Value	Dev (%)	CPU (scs)	Value	Dev (%)	CPU (scs)
			0.4	2274.8	38.2	2365.3	4.0	31.5	2274.8	0.0	61.3
15	5	3	0.6	2429.8	69.5	2459.7	1.2	34.2	2459.7	1.2	65.7
			0.8	2584.8	41.0	2627.6	1.7	35.0	2627.6	1.7	69.3
			1.0	2733.8	106.4	2814.9	3.0	36.2	2794.6	2.2	71.6
			0.2	2119.8	29.5	2179.4	2.8	32.8	2162.4	2.0	64.0
			0.4	2274.8	24.6	2325.5	2.2	34.7	2325.1	2.2	68.9
15	5	4	0.6	2429.8	47.4	2510.6	3.3	37.1	2474.1	1.8	72.6
			0.8	2584.8	36.2	2635.7	2.0	38.7	2635.7	2.0	75.0
			1.0	2733.8	66.0	2805.1	2.6	38.8	2796.1	2.3	76.9
			0.2	5970.6	90.0	5998.07	0.5	16.3	5998.07	0.5	32.2
			0.4	6353.9	481.7	6495.68	2.2	17.7	6495.68	2.2	35.1
20	3	2	0.6	6737.2	4558.3	6737.15	0.0	19.1	6737.15	0.0	37.5
			0.8	7120.4	51470.9	7250.2	1.8	20.1	7120.44	0.0	39.8
			1.0	7435.7	24705.1	7690.44	3.4	21.0	7690.44	3.4	41.5
			0.2	5135.8	276.0	5135.81	0.0	25.6	5135.81	0.0	50.3
			0.4	5705.3	762.1	5740.23	0.6	28.3	5740.23	0.6	56.0
20	4	2	0.6	6214.0	4745.1	6350.84	2.2	31.4	6242.07	0.5	61.0
			0.8	6654.4	18544.3	6801.44	2.2	33.5	6729.13	1.1	65.1
			1.0	7037.4	29172.6	7165.38	1.8	35.1	7165.38	1.8	69.0
			0.2	5135.8	199.9	5135.81	0.0	37.6	5135.81	0.0	74.1
			0.4	5705.3	1229.2	5771.89	1.2	40.9	5771.89	1.2	80.7
20	4	3	0.6	6214.0	5750.7	6377.65	2.6	43.4	6248.07	0.5	86.0

Continued on next page

Table 4 – *Detailed results for the CAB instances — continued from previous page*

				CPLEX		Heur 1000			Heur 2000		
n	p	r	α	Value	CPU (scs)	Value	Dev (%)	CPU (scs)	Value	Dev (%)	CPU (scs)
			0.8	6654.4	24547.4	6959.13	4.6	46.2	6863.92	3.1	90.8
			1.0	7037.4	19571.9	7492.79	6.5	48.2	7398.64	5.1	93.8
			0.2	4499.1	93.1	4648.9	3.3	32.5	4499.1	0.0	64.7
			0.4	5126.4	98.3	5234.8	2.1	36.8	5126.4	0.0	72.8
20	5	2	0.6	5753.8	2443.2	5826.4	1.3	40.3	5826.4	1.3	80.9
			0.8	6304.2	16982.8	6381.2	1.2	43.6	6381.2	1.2	88.6
			1.0	6696.2	34889.4	6947.0	3.7	45.9	6903.6	3.1	91.3
			0.2	4499.1	72.1	4507.8	0.2	59.3	4507.8	0.2	117.4
			0.4	5126.4	78.8	5260.6	2.6	65.5	5260.6	2.6	129.8
20	5	3	0.6	5753.8	2875.6	5826.4	1.3	71.0	5818.3	1.1	141.4
			0.8	6304.2	19990.1	6471.3	2.7	75.0	6381.2	1.2	151.3
			1.0	6696.2	33709.4	7028.3	5.0	78.7	6894.7	3.0	156.4
			0.2	4499.1	168.0	4507.8	0.2	66.2	4507.8	0.2	134.5
			0.4	5126.4	211.0	5305.3	3.5	72.0	5214.5	1.7	142.9
20	5	4	0.6	5753.8	1214.8	6019.3	4.6	78.1	5959.0	3.6	151.7
			0.8	6304.2	25392.8	6623.8	5.1	81.0	6483.2	2.8	159.0
			1.0	6696.2	12991.4	7119.9	6.3	83.5	7039.9	5.1	164.7
			0.2	5313.3	11385.8	5528.21	4.0	56.4	5528.21	4.0	111.7
			0.4	5961.2	17156.0	6140.71	3.0	63.2	6140.71	3.0	125.2
25	5	2	0.6	6568.8	124321.0	6796.78	3.5	68.7	6796.78	3.5	136.4
			0.8	7159.2	221453.3	7365.45	2.9	74.2	7365.45	2.9	147.3
			1.0	7735.8	415802.1	8122.09	5.0	78.5	8099.9	4.7	156.3

Continued on next page

Table 4 – *Detailed results for the CAB instances — continued from previous page*

				CPLEX		Heur 1000			Heur 2000		
n	p	r	α	Value	CPU (scs)	Value	Dev (%)	CPU (scs)	Value	Dev (%)	CPU (scs)
			0.2	5313.3	30176.6	5505.8	3.6	102.5	5505.8	3.6	203.9
			0.4	5961.2	54987.0	6076.01	1.9	112.7	6076.01	1.9	226.9
25	5	3	0.6	6568.8	25439.3	6917.3	5.3	121.0	6908.77	5.2	240.4
			0.8	7159.2	137703.5	7436.9	3.9	129.0	7436.9	3.9	255.2
			1.0	7735.8	306435.3	8040.74	3.9	135.1	8040.74	3.9	267.7
			0.2	5313.3	7166.7	5470.89	3.0	114.6	5470.89	3.0	227.7
			0.4	5961.2	33398.2	6113.35	2.6	123.6	6113.35	2.6	246.1
25	5	4	0.8	7159.2	217706.7	7539.48	5.3	136.5	7539.48	5.3	272.5
			1.0	7735.8	211488.1	8287.62	7.1	144.3	8162.45	5.5	282.9
Average				**4734.5**	**29116.4**	**4868.94**	**2.6**	**47.7**	**4839.54**	**2.0**	**94.6**