Pesquisa Operacional (2012) 32(3): 465-496
© 2012 Brazilian Operations Research Society
Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope

ADDRESSING CONGESTION ON SINGLE ALLOCATION HUB-AND-SPOKE NETWORKS

Ricardo Saraiva de Camargo* and Gilberto de Miranda Jr.

Received March 28, 2012 / Accepted August 8, 2012

ABSTRACT. When considering hub-and-spoke networks with single allocation, the absence of alternative routes makes this kind of systems specially vulnerable to congestion effects. In order to improve the design of such networks, congestion costs must be addressed. This article deploys two different techniques for addressing congestion on single allocation hub-and-spoke networks: the Generalized Benders Decomposition and the Outer Approximation method. Both methods are able to solve large scale instances. Computational experiments show how the adoption of advanced solution strategies, such as Pareto-optimal cut generation on the Master Problem branch-and-bound tree, may be decisive. They also demonstrate that the solution effort is not only associated with the size of the instances, but also with their combination of the installation and congestion costs.

Keywords: single allocation hub location problem, Benders decomposition method, outer-approximation algorithm, large scale optimization.

1 INTRODUCTION

Hub-and-spoke networks became an important field of discrete location research. The relevance is largely explained by their widespread use in cargo and passengers transportation and telecommunication systems [5, 16].

In hub-and-spoke networks, direct transportation of flows between pairs of origin-destination nodes is usually extremely costly. As an alternative, flows from different origins but addressed to the same destination can be consolidated at transshipment nodes, known as hubs, prior to be routed, sometimes via other hubs, towards their destinations. Hubs are then responsible for flow aggregation and redistribution. The bundle of flows at the hubs increases the traffic on inter-hub connections, enabling the use of more efficient and higher volume carriers, resulting then in lower per unit transportation costs [47]. Thus economies of scale can be achieved by bulk transportation. Furthermore, hub-and-spoke networks allow lower infrastructure costs and greater overall efficiency of logistics [37].

*Corresponding author

Industrial Engineering Department, Federal University of Minas Gerais, Minas Gerais, MG, Brazil.
E-mails: rcamargo@dep.ufmg.br; miranda@dep.ufmg.br

Generally, in the design of hub-and-spoke networks, there is a connection between every hub pair; no two non-hub nodes can be serviced by a direct link; an origin-destination demand is routed through one or at most two hubs; and the economies of scale at inter-hub connections are represented by a discount factor ($0 \leq \alpha \leq 1$). Usually the main decisions involve the location of hub facilities, the allocation of origin and destination nodes to hubs (formation of the spokes), the establishment of discounted transportation connections and the routing of flows through the network.

Moreover, according to the characteristics considered, different assumptions may be addressed, including: Single or multiple allocation of the non-hub nodes to the installed hubs, the number of hubs to be located may or may not be known beforehand, direct service between non-hub nodes may be enabled, capacity constraints on the amount of traffic a installed hub can handle, consideration of congestion effects at the installed hubs, flow dependent economies of scale on inter-hub connections, and furthermore there may be not a direct connection between every hub pair, implying then an incomplete hub network structure [6] or a network topology where the hubs are connected by means of a spanning tree [18]. A general review of different problems of hub-and-spoke networks can be found at Campbell [14, 15], while a exhaustive survey is present on Campbell *et al.* [16] and Alumur & Kara [5].

One of these problem variants is the single allocation hub location problem (SAHLP) where each non-hub is allocated to a single hub only, a fixed cost is incurred each time a node is selected to be a hub, and the path connecting each pair of origin-destination nodes has one or two hubs present. When there are no installation costs but the number p of hubs to be located is known beforehand, the problem is named as the single allocation p-hub location problem (SApHLP). As both problems are closely related, they share the same mathematical programming formulations, differentiating only by the presence or not of a constraint establishing that p hubs must be located and a term on the objective function summing the fixed cost of the installed hubs.

Furthermore, while the multiple allocation hub location problem is closely related to the facility location problem [40], the SAHLP is more akin to the quadratic assignment problem [46]. Hence it is harder to handle, requiring different approaches regarding solution techniques and proper formulations.

Among the available mixed integer linear programming formulations [3, 14, 24, 28, 46, 55], two are worthy of notice: the models of Skorin-Kapov *et al.* [55] and Ernst & Krishnamoorthy [28]. While the first has the tightest linear programming relaxation (optimality gaps smaller than 1%), yielding integer solution values for the integer variables most of the time at the expense of computer memory and time; the latter presents a good trade-off between formulation size (fewer variables and constraints) and computer effort to solve it.

When solution methods are considered, the SAHLP and the SApHLP do not share the same strategies. Most of the algorithms [2, 28, 29, 38, 39, 46, 51, 54] to solve the SApHLP are based on specialized heuristics and branch-and-bound procedures, while those [1, 17, 21, 53, 56] addressed to the SAHLP rely on meta-heuristics like genetic and simulated annealing algorithms.

Independently of which hub-and-spoke problem is addressed, one of the main overall advantages of such systems is the exploitation of scale economies. However this may induce the formation of networks that tend to overload a small number of hubs, resulting in some inter-hub connections more heavily-utilized than others. This is specially true when single allocations are involved, since the flow leaving from and arriving at a non-hub node goes through only one hub. Hence it is unavoidable to take congestion effects into consideration.

A common way of addressing congestion in SAHLP and SApHLP networks is to limit the amount of traffic a installed hub can handle [7,8,20,30,41,44,52]. Unfortunately, capacity constraints do not mimic the explosive nature of congestion: the more flow a hub attracts, the harder the handling process becomes, the greater the costs. Usually these costs increase extremely rapid due to queuing and delay effects. Hence, elaborate cost functions are needed such as the one employed by Elhedhli & Hu [26].

Elhedhli & Hu [26] are the first authors to consider explicitly the congestion effects on the objective function for the SApHLP. Using a power-law function widely utilized to estimate delay costs in airport applications [34]. They propose a non-linear formulation where these convex cost functions, that increase rapidly as more traffic flows through the installed hubs, are present on the objective function. They linearize their model utilizing a set of infinite piece-wise linear and tangent hyperplanes, and then solve it by means of a Lagrangian relaxation algorithm. They solve only toy instances (up to 25 nodes) with an average optimality gap close to 1%. The obtained solutions have a more balanced overall distribution of flows through the network than the ones attained by disregarding the congestion effects [26].

In this study, instead of linearizing the implied nonlinear formulation, two very efficient algorithms based on the Generalized Benders Decomposition (GBD) method [33] and on the Outer Approximation technique (OA) [23,31,57] are employed to handle the nonlinear SAHLP under congestion. These algorithms are different from the former deployed for the multiple allocation variant proposed at Camargo *et al.* [11]. The main contributions of the present article are the proper derivation of pareto-optimal Benders cuts as devised by Papadakos [49], the exploitation of the special structure of the formulation in order to enable the blending of GBD and OA cuts at the same master program, and the implementation of this strategy in a branch-and-cut framework.

Due to the combination of the best features of each method, the proposed scheme is able to solve large instances to optimality. As far as the authors know, there is no previous report of such large problems solved by the OA technique. Furthermore, a fair comparison of both techniques is presented demonstrating the advantages of one method over the other as a function of some of the instance parameters.

The paper is organized as follows. Section 2 provides general notations and definitions of the nonlinear SAHLP under congestion effects. The GBD and OA algorithms are presented in sections 3 and 4, respectively. Finally, the computational results are shown in section 6, while the final remarks and future research plans are done in section 7.

2 NOTATION AND FORMULATION

In this section, the SAHLP under congestion is formulated as a mixed integer nonlinear program (MINLP) using the model of Skorin-Kapov *et al.* [55] as a starting basis. The formulation requires the following definitions: Let N be the set of demand node locations which exchange flows and let K be the set of candidate nodes to become hubs. Usually $K \subseteq N$, but it is assumed henceforth that all demand nodes are candidates to have a installed hub, implying $K \equiv N$. For all node pairs i and j $(i, j \in N : i \neq j)$, w_{ij} represents the flow demand from origin node i to destination node j which is routed through either one or two installed hubs. Normally $w_{ij} \neq w_{ji}$. Let also $O_i = \sum_{j \in N} w_{ij}$ and $D_i = \sum_{j \in N} w_{ji}$ be the total of demand that is originated from and destined to node $i \in N$, respectively.

Further, let f_k be the fixed installation cost of a hub at node $k \in N$ and let c_{ijkm} be the transportation cost per unit of flow from node i to node j routed via hubs at nodes k and m, that is, the standard transportation cost of route $i - k - m - j$ $(i, j, k, m \in N)$. This transportation cost is the composition of three cost segments: $c_{ijkm} = c_{ik} + \alpha c_{km} + c_{mj}$, where c_{ik} and c_{mj} are the standard transportation cost per unit of flow from node i to hub k and from hub m to node j, and αc_{km} is the discounted standard transportation cost between hubs k and m. The discount factor $0 \leq \alpha \leq 1$ represents the scale economies on the inter-hub connections. If only one hub is present in any given route then $k = m$ and no discount factor is applied for the route $i - k - k - j$.

The MINLP uses flow variables $x_{ijkm} \geq 0$ to represent the fraction of demand w_{ij} $(i, j \in N)$ that is routed through hubs k and m $(k, m \in N)$, in this order; the variables g_k to account for the total flow passing through hub $k \in N$; and the integer variables $z_{ik} \in \{0, 1\}$ to indicate if node $i \in N$ is allocated to hub $k \in N$ $(z_{ik} = 1)$ or not $(z_{ik} = 0)$. When a hub is located at node $k \in N$, then $z_{kk} = 1$; otherwise $z_{kk} = 0$.

The congestion cost function is usually defined as a power-law $\tau_k(g_k) = a\, {g_k}^b$ that increases rapidly as more traffic goes through hub $k \in K$, where the parameters $a > 0$ and $b \geq 1$ are scalars related to the hub features. The function $\tau_k(g_k)$ is increasing on $[0, +\infty)$, proper convex and smooth, and it is normally used to estimate delay costs in airport applications [34], being already used by Elhedhli & Hu [26] for the same problem. In this research, the adopted power law function is designed to consider congestion effects after a given flow threshold Γ_k (usually set to 70% of the hub nominal capacity) is trespassed. Such function can be written as $\tau_k(g_k) = a\, (\max\{0, g_k - \Gamma_k\})^b$ without loss of generality, since Γ_k can be set for any value where the congestion effects start to degrade the network economies of scale.

In the SAHLP, as each non-hub node is allocated to a single installed hub only, demands w_{ij} and w_{ji} are sent over the same route, enabling then the reduction of the number of variables x_{ijkm} in half [54]. Furthermore, as the outgoing and the incoming flows of a non-hub must go and arrive through the same hub, the cost component of local traffic can be written as $\sum_{k \neq i} (O_i + D_i)\, c_{ik}\, z_{ik}$. Thus enabling the redefinition of the costs c_{ijkm} as

$$c_{ijkm} = \alpha\big(w_{ij} c_{km} + w_{ji} c_{mk}\big), \quad \forall\, i < j,\ k \neq m.$$

These three simple manipulations improve the overall performance of the algorithms here proposed. In the remainder of this paper, for the sake of simplicity in presentation, one must consider $i, j, k, m \in N$ and $i < j$. Hence the implied formulation is given as:

$$\min \sum_k [f_k\, z_{kk} + \tau_k(g_k)] + \sum_{k \neq i} (O_i + D_i)\, c_{ik}\, z_{ik} + \sum_{i<j} \sum_{k \neq m} c_{ijkm}\, x_{ijkm} \qquad (1)$$

$$\text{s. t.: } \sum_k z_{ik} = 1 \qquad \forall\, i \qquad (2)$$

$$\sum_m x_{ijkm} = z_{ik} \qquad \forall\, i < j, k \qquad (3)$$

$$\sum_k x_{ijkm} = z_{jm} \qquad \forall\, i < j, m \qquad (4)$$

$$\sum_i (O_i + D_i) z_{ik} - \sum_{i<j} (w_{ij} + w_{ji}) x_{ijkk} = g_k \qquad \forall\, k \qquad (5)$$

$$z_{ik} \leq z_{kk} \qquad \forall\, i \neq k \qquad (6)$$

$$x_{ijkm} \geq 0 \qquad \forall\, i < j, k, m \qquad (7)$$

$$g_k \geq 0 \qquad \forall\, k \qquad (8)$$

$$z_{ik} \in \{0, 1\} \qquad \forall\, i, k \qquad (9)$$

The objective function (1) minimizes the total cost associated with the demand transportation, the congestion effects and the hub installation costs. Constraints (2) assure that all nodes are allocated to a hub. Constraints (3) guarantee that routes beginning at origin node i, then passing firstly at hub k, and finishing at destination node j will only exist if node i is allocated to hub k. Likewise, constraints (4) guarantee that routes beginning at origin i and passing at hub m just before finishing at destination j will only exist if node j is allocated to hub m. Constraints (5) are responsible for accounting the total hub traffic, avoiding the double computation of the local traffic component. Constraints (6) allow a node i to be allocated to hub k only if hub k is installed, while (7), (8) and (9) are the non-negativity and the integrality constraints of variables x_{ijkm}, g_k and z_{ik}, respectively.

Although there are other formulations for the SAHLP [14, 24, 28], the formulation of Skorin-Kapov *et al.* [55] is chosen because it provides the tightest linear programming bound. This is a key feature, since nonlinear terms tend to weaken any mixed-integer programming formulation, enlarging the integrality gap as the nonlinearities become dominant. The chosen formulation has also a very interesting property: for a fixed feasible vector $\mathbf{z}$, the formulation is decomposable for each $i - j$ pair, recalling that the variables g_k can be recovered by replacing x_{ijkk} by $z_{ik} z_{jk}$.

The total hub flow g_k has three different forms of being assessed in the literature. For some applications, such as the ones regarding public service networks, such as postal, health care, and public security systems, where the most overloading tasks are the sorting and the assembling of flows at the first hub of a given path, Ernst & Krishnamoorthy [30] propose constraints (10).

These constraints assume that the hubs may only be overloaded by the incoming flows originated at the non-hub nodes directly allocated to them.

$$\sum_i O_i z_{ik} = g_k \qquad \forall\, k. \tag{10}$$

For those applications where the aforementioned assumption does not hold, Ernst & Krishnamoorthy [30] further suggest that the total hub flow g_k can be determined by adding the demands destined to the non-hub nodes directly allocated to a given hub, yielding then constraints (11).

$$\sum_i (O_i + D_i) z_{ik} = g_k \qquad \forall\, k. \tag{11}$$

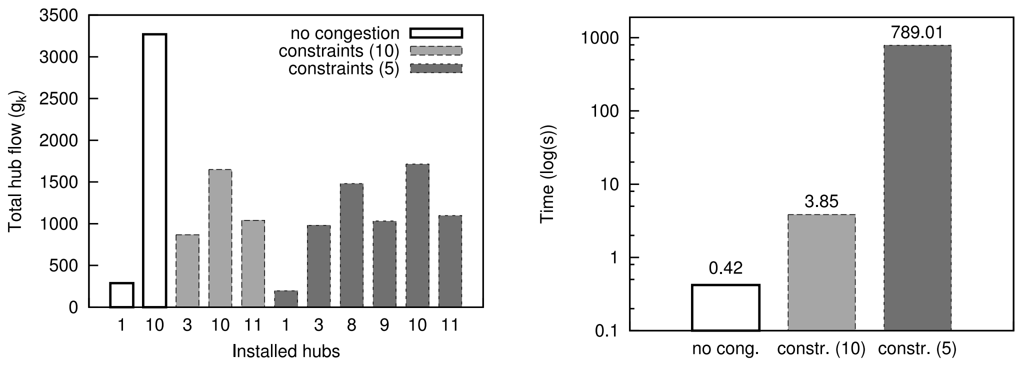
As observed by Labbé *et al.* [41], constraints (11) compute the total hub flow g_k in a wrong way. For those non-hub nodes which are allocated to the same hub, the flows originated from and destined to them are accounted twice: One time on the parcel O_i and another time on the parcel D_i. Labbé *et al.* [41] correct this misconception by proposing constraints (5).

Constraints (5) and (10) induce very different optimal networks, as well as require distinct computational efforts. In order to illustrate these differences, a small experiment was carried out by optimally solving, via CPLEX 12, the instance AP20.2, with 20 nodes and $\alpha = 0.2$, of the well-known Australian Post (AP) standard data set [28,30]. The adopted power-law congestion cost function had parameters $a = 0.01$ and $b = 2$. Figure 2 shows how the network design and the required solution time were affected after constraints (10) were exchanged for (5). The attained values for when no congestion is accounted are also shown. Once again constraints (11) are meaningless since they wrongly compute the total hub flow, therefore they were not considered in this experiment.

In Figure 1(a), the total load (g_k) of each installed hubs is represented by a bar above the hub's index. When congestion effects are disregarded, only two hubs (1 and 10) are installed at the optimal solution. When congestion effects are accounted by computing the total hub flow through constraints (10) or (5), the optimal networks have three (hubs 3, 10 and 11) and six (hubs 1, 3, 8, 9, 10 and 11) hubs, respectively. In other words, they have very distinctive optimal infrastructures.

When the computational efforts are observed, the differences are more pronounced. Figure 1(b) presents, in logarithmic scale, the required solution time for the three considered situations. As can be seen, when constraints (5) are utilized, the required solution time is 204 times greater than when using constraints (10). Constraints (5) greatly hardens the problem, because the term x_{ijkk} is a linearization of the product of $z_{ik}z_{jk}$, which greatly affects the attained lower bounds during the branch-and-bound search.

Constraints (5) and (10) have different design purposes, being suitable for distinct applications. For those applications in which constraints (10) are more adequate, Camargo *et al.* [13] have devised a new technique that hybridizes methods of OA [23,31] and Benders decomposition [9].



(a) Optimal network infra-structure for different forms of accounting the total hub flow.

(b) Computational effort for different forms of accounting the total hub flow.

Figure 1 – Comparison for different forms of accounting the total hub flow for instance AP20.2.

The proposed technique blends OA and classical Benders cuts in a straightforwardly manner, since there is no coupling of the g and x variables on constraints (10). Their algorithm is capable of solving instances up to 200 nodes in reasonable time. Remark that previous articles on the literature solved only small problems up to 25 nodes [26, 27].

Unfortunately, for those practical situations where one needs to account the total hub flow using constraints (5), which are more general, the technique already proposed by Camargo *et al.* [13] can not be directly applied, due to coupling of the g and x variables. Furthermore, as far as the authors know, there are no other studies in the literature that address congestion effects deploying constraints (5).

In the light of the aforementioned reasons, this article proposes a new hybrid OA/Benders algorithm that exploits the structure of the formulation, and optimally solves instances up to 100 nodes. Further, the method clearly outperforms an implemented GBD [33] algorithm, a well known technique for solving MINLPs.

Moreover, whenever two formulations – one tight and large, and another weak and small–are available for a given problem, the proposed scheme can be applied, if the global utilization of the common resources is accounted using the smaller model and is computed independently of the large scale system variables. This kind of manipulation simplifies the handling of the non-linearities, by confining them to a kind of *sandbox*, while uses the large scale system to improve the formulation bounds.

For sake of presentation and understanding of the required concepts of the hybrid technique, the next section is dedicated to the application of the GBD method for the proposed model.

3 GENERALIZED BENDERS DECOMPOSITION

The Benders decomposition algorithm [9] is a partition method for solving mixed-integer linear and non-linear programming problems. In general terms, the algorithm relies on a projection problem manipulation, followed by solution strategies of dualization, outer linearization and relaxation. The complicating variables (integer variables) of the original problem are projected out, resulting into an equivalent model with fewer variables, but many more constraints. When attaining optimality, a large number of these constraints will not be binding, suggesting then a strategy of relaxation that ignores all but a few of these constraints.

Benders proposes a procedure to add these constraints on demand by using two levels of coordination. At a higher level, known as master problem (MP), a relaxed version of the original problem having the set of the integer variables and its associated constraints is responsible for fixing the values of these integer variables and for providing a lower bound (LB) for the problem. At a lower level, known as subproblem (SP), the dual of the original problem with the values of the integer variables temporarily fixed by the MP is responsible for rendering a new cut or a Benders cut to be added to the MP and for generating a upper bound (UB) for the problem. The algorithm iterates, solving the MP and the SP one at a time, until the UB and the LB converge towards an optimal solution, if one exists.

Since the pioneering work of Geoffrion & Graves [32], the Benders decomposition method has been successfully deployed in solving large-scale problems: e.g., the uncapacitated network design problem with undirected arcs [43], the locomotive and car assignment problem [19] and, more recently, the multiple allocation hub location problem [10–12].

In Sections 3.1, 3.2 and 3.3, formal descriptions of the MP, the associated SP, implementation issues and solution strategies are presented.

3.1 Benders master program

Projecting the problem (1)–(9) onto the space of the z variables results into the equivalent problem:

$$\min_{z\in Z} \sum_{k} f_k\, z_{kk} + \sum_{k\neq i} (O_i + D_i)\, c_{ik}\, z_{ik} + \phi(z)$$

where $Z = \{z \in \{0,1\} \mid \text{constraints(2) and (6) hold}\}$ and $\phi(z)$ is the following SP:

$$\phi(z) = \min_{(x,g)\in G} \left\{ \sum_{k} \tau_k(g_k) + \sum_{i<j} \sum_{k\neq m} c_{ijkm}\, x_{ijkm} \right\}$$

being

$$G = \big\{ (x,g) \geq \mathbf{0} \mid \text{constraints (3)–(5) hold} \big\}.$$

Since the constraints defining z are enough to ensure feasibility, the SP is bounded. Further, as $\phi(z)$ has a convex and differentiable objective function and linear constraints, its Karush-Kuhn-Tucker conditions are necessary and sufficient for optimality, hence amenable to dualization techniques. So associating vectors of dual variables u, v and β to constraints (3), (4) and (5), respectively, and because there is no duality gap, $\phi(z)$ can be re-written as:

$$\phi(z) = \max_{u,v,\beta} \left\{ \min_{(x,g)\in G} \left\{ \sum_k \tau_k(g_k) + \sum_{i<j} \sum_{k\neq m} c_{ijkm}\, x_{ijkm} + \sum_{i<j} \sum_{k,m} u_{ijk} x_{ijkm} \right.\right.$$

$$\left. \sum_{i<j} \sum_{k,m} v_{ijm} x_{ijkm} - \sum_{i<j} \sum_k \beta_k (w_{ij} + w_{ji}) x_{ijkk} - \sum_k \beta_k g_k \right\}$$

$$\left. - \sum_{i<j} \sum_k u_{ijk} z_{ik} - \sum_{i<j} \sum_m v_{ijm} z_{jm} + \sum_{i,k} \beta_k (O_i + D_i) z_{ik} \right\}$$

Since the supremum is the least upper bound and with the help of variable $\eta \geq 0$, the whole problem (1)–(9) is then equivalent to following MP:

$$\min_{z\in Z} \sum_k f_k\, z_{kk} + \sum_{k\neq i} (O_i + D_i)\, c_{ik}\, z_{ik} + \eta \tag{12}$$

$$\text{s. t.:}\quad \eta \geq \min_{(x,g)\in G} \left\{ \sum_k \tau_k(g_k) + \sum_{i<j} \sum_{k\neq m} c_{ijkm}\, x_{ijkm} + \sum_{i<j} \sum_{k,m} u_{ijk} x_{ijkm} \right.$$

$$\left. \sum_{i<j} \sum_{k,m} v_{ijm} x_{ijkm} - \sum_{i<j} \sum_k \beta_k (w_{ij} + w_{ji}) x_{ijkk} - \sum_k \beta_k g_k \right\}$$

$$- \sum_{i<j} \sum_k u_{ijk} z_{ik} - \sum_{i<j} \sum_m v_{ijm} z_{jm} + \sum_{i,k} \beta_k (O_i + D_i) z_{ik} \qquad \forall\, u, v, \beta \tag{13}$$

$$\eta \geq 0 \tag{14}$$

Because a large number of the constraints of the MP (12)–(14) will not be binding when optimality is attained, the GBD algorithm solves the MP through a strategy of relaxation that ignores all but a few of the constraints (13). These constraints are then added, via a iterated procedure, to the MP as needed. Thus for a given iteration t, where $z = z^t$ and after the solution of the associated SP and the recovery of the optimal values of u^t, v^t and β^t, the optimal value of $\phi(z^t)$ is given by:

$$\phi(z^t) = \min_{(x,g)\in G} \left\{ \sum_k \tau_k(g_k) + \sum_{i<j} \sum_{k\neq m} c_{ijkm}\, x_{ijkm} + \sum_{i<j} \sum_{k,m} u^t_{ijk} x_{ijkm} \right.$$

$$\left. \sum_{i<j} \sum_{k,m} v^t_{ijm} x_{ijkm} - \sum_{i<j} \sum_k \beta^t_k (w_{ij} + w_{ji}) x_{ijkk} - \sum_k \beta^t_k g_k \right\}$$

$$- \sum_{i<j} \sum_k u^t_{ijk} z^t_{ik} - \sum_{i<j} \sum_m v^t_{ijm} z^t_{jm} + \sum_{i,k} \beta^t_k (O_i + D_i) z^t_{ik} \tag{15}$$

Further, constraints (13) can be rewritten for iteration t in the form:

$$\eta \geq \min_{(x,g)\in G} \Bigg\{ \sum_k \tau_k(g_k) + \sum_{i<j}\sum_{k\neq m} c_{ijkm}\, x_{ijkm} + \sum_{i<j}\sum_{k,m} u^t_{ijk} x_{ijkm}$$
$$\sum_{i<j}\sum_{k,m} v^t_{ijm} x_{ijkm} - \sum_{i<j}\sum_{k} \beta^t_k (w_{ij} + w_{ji}) x_{ijkk} - \sum_k \beta^t_k g_k \Bigg\}$$
$$- \sum_{i<j}\sum_{k} u^t_{ijk} z_{ik} - \sum_{i<j}\sum_{m} v^t_{ijm} z_{jm} + \sum_{i,k} \beta^t_k (O_i + D_i) z_{ik} \tag{16}$$

Therefore, by eliminating the minimum in (16) by using (15), the relaxed Benders master program (RMP) is stated as:

$$\min_{z\in Z} \sum_k f_k\, z_{kk} + \sum_{k\neq i} (O_i + D_i)\, c_{ik}\, z_{ik} + \eta \tag{17}$$

$$\text{s. t.:}\quad \eta \geq \phi(z^t) - \sum_{i<j}\sum_{k} u^t_{ijk}(z_{ik} - z^t_{ik}) - \sum_{i<j}\sum_{m} v^t_{ijm}(z_{jm} - z^t_{jm})$$
$$+ \sum_{i,k} \beta^t_k (O_i + D_i)(z_{ik} - z^t_{ik}) \qquad \forall\, t = 1, \ldots, T \tag{18}$$

$$\eta \geq 0 \tag{19}$$

where T is the maximum number of iterations in order to attaining the optimal solution. In the next section, the resolution of SP $\phi(z)$ is presented.

3.2 Benders subproblem

For a hub structure z^t fixed by the MP (17)–(19) at iteration t, the SP $\phi(z^t)$ is given by:

$$\min \sum_k \tau_k(g_k) + \sum_{i<j}\sum_{k\neq m} c_{ijkm}\, x_{ijkm} \tag{20}$$

$$\text{s. t.:}\quad g_k + \sum_{i<j} (w_{ij} + w_{ji}) x_{ijkk} = \sum_i (O_i + D_i) z^t_{ik} \qquad \forall\, k \tag{21}$$

$$\sum_m x_{ijkm} = z^t_{ik} \qquad \forall i < j,\ \forall\, k \tag{22}$$

$$\sum_k x_{ijkm} = z^t_{jm} \qquad \forall i < j,\ \forall\, m \tag{23}$$

$$x_{ijkm} \geq 0 \qquad \forall i < j,\ \forall k, m \tag{24}$$

$$g_k \geq 0 \qquad \forall k \tag{25}$$

In the SP (20)–(25), there are two sets of variables, x and g, coupled by the constraints (21). After dualizing these constraints by the associated dual multipliers β, a decomposable problem is implied:

$$d(\beta) = \min \sum_k (\tau_k(g_k) - \beta_k\, g_k) - \sum_{i<j} \sum_k \beta_k (w_{ij} + w_{ji}) x_{ijkk} + \sum_{i<j} \sum_{k \neq m} c_{ijkm}\, x_{ijkm} + \sum_{i,k} \beta_k (O_i + D_i) z^t_{ik}$$

subject to constraints (22)–(25).

The problem $d(\beta)$ is decomposable in two smaller SPs and a constant: A linear problem, $d_L(\beta)$, having only the x_{ijkm} variables; a non-linear problem $d_{NL}(\beta)$, having just the g_k variables; and a fixed term. Further, the SP $d_{NL}(\beta)$ is convex and differentiable, being therefore its Karush-Kuhn-Tucker conditions necessary and sufficient for optimality.

Hence for each k, given a structure z^t fixed by the RMP at iteration t, the optimal solution of β^t minimizes $d_{NL}(\beta)$ or:

$$\beta^t_k = \tau'_k(g^t_k) \qquad \forall\, k \qquad (26)$$

In order to determine the optimal value of β^t_k, the value of g^t_k has to be computed first. As the variables x_{ijkm} can also be stated as the product of $z_{ik}\, z_{jm}$ for any RMP solution z^t, the optimal value of g^t_k can be obtained by rewriting equation (21) in the following way:

$$g^t_k = \sum_i (O_i + D_i) z^t_{ik} - \sum_{i<j} (w_{ij} + w_{ji}) z^t_{ik} z^t_{jk} \qquad \forall\, k \qquad (27)$$

This small and apparently innocuous detail makes it possible to easily calculate the value of g^t, avoiding therefore the use of non-linear programming methods for evaluating $\phi(z^t)$. Moreover, this equation (27) has an additional advantage since it also allows the decomposition of the SP $d_L(\beta)$ in smaller problems, one for each $i - j$ pair. Henceforth the optimal values of u^t and v^t can now be computed by solving the dual linear programming of these smaller problems:

$$\max \sum_k u_{ijk}\, z^t_{ik} + \sum_m v_{ijm}\, z^t_{jm} \qquad (28)$$

$$\text{s. t.:} \quad u_{ijk} + v_{ijm} \leq c_{ijkm} \qquad \forall\, k \neq m \qquad (29)$$

$$u_{ijk} + v_{ijk} \leq -\beta_k (w_{ij} + w_{ji}) \qquad \forall\, k \qquad (30)$$

$$u_{ijk} \in \mathbb{R} \qquad \forall\, k \qquad (31)$$

$$v_{ijm} \in \mathbb{R} \qquad \forall\, m \qquad (32)$$

3.3 Implementation and solution strategies

The efficiency of GBD algorithm depends mainly on the number of iterations required to attain global convergence. This number is intimately related to the quality of the Benders cuts assembled. Strong cuts usually mean fewer iterations. In order to have strong cuts, the solution of the SP has to be judiciously done, since the Benders algorithm is very sensitive to the selection of the dual variables. If care is not taken, a poor behavior of the algorithm may be expected.

Magnanti & Wong [42] propose a methodology for tackling this issue and hence accelerate the Benders algorithm. They notice that when the dual SP has multiple optimal solutions different Benders cuts can be generated. Instead of adding them all to the MP, they propose a SP to be solved, very similar to (28)–(32), in order to find the strongest cut, which dominates all the other cuts in the sense of pareto-optimality. Magnanti & Wong define the strength of a cut compared to another cut according to the following definition:

Definition 1. *A cut is said to be pareto-optimal if it is not dominated by any other cut. Let* $U = \{(u, v) : \text{satisfying constraints } (29) - (32)\}$ *be the set of feasible values for the dual variables* u *and* v*. A Benders cut* (18) *corresponding to* $(u^1, v^1) \in U$ *dominates that corresponding to* $(u^2, v^2) \in U$*, if:*

$$\sum_{i<j}\sum_{k} u^1_{ijk}\, z_{ik} + \sum_{i<j}\sum_{m} v^1_{ijm}\, z_{jm} \geq \sum_{i<j}\sum_{k} u^2_{ijk}\, z_{ik} + \sum_{i<j}\sum_{m} v^2_{ijm}\, z_{jm}, \quad \forall\, z \in Z$$

with strict inequality for at least one point $z \in Z$*. Similarly, it is said that* (u^1, v^1) *dominates* (u^2, v^2) *and it is called pareto-optimal.*

In order to build up the pareto-optimal cut generation SP, they use the notion of core points:

Definition 2. *A point* $z^0 \in Z$ *is a core point if it belongs to the relative interior of the convex hull of* Z *or* $z^0 \in ri(Z^c)$*, where* $ri(*)$ *and* Z^c *are the relative interior and convex hull of set* Z*, respectively.*

In the original algorithm of Magnanti & Wong, they have two different SPs to solve at each iteration: One associated to the current MP solution z^t, SP (28)–(32), and another related to a initial fixed given core point z^0. This additional SP differs slightly from the original SP, having the form:

$$\max \sum_{k} u_{ijk}\, z^0_{ik} + \sum_{m} v_{ijm}\, z^0_{jm} \tag{33}$$

$$\text{s.t.:} \sum_{k} u_{ijk}\, z^t_{ik} + \sum_{m} v_{ijm}\, z^t_{jm} = \bar{d}^{ij}_L(\beta, u, v) \tag{34}$$

$$\text{constraints } (29) - (32) \tag{35}$$

where $\bar{d}^{ij}_L(\beta, u, v)$ is the optimal value of the objective function of the SP associated with the MP z^t. This Magnanti & Wong SP (33)–(35) generates pareto-optimal cuts that speed up the convergence of the method. In order to achieve a good performance, one must assess the computational burden of finding these pareto-optimal cuts faced to the number of iterations that they might save.

More recently, Papadakos [49] shows that it is possible to deal with a different version of the Magnanti & Wong SP that decreases the computational effort for cut generation by disregarding constraint (34). As a drawback, this enhancement only works if a different core point z^0 is used at each iteration [49].

Even though, as pointed out by Mercier *et al.* [45], there are not practical methods to find good core points, Papadakos [49] proves that given a valid initial core point $z^0 \in ri(Z^c)$ and $z \in Z$ then any convex combination of z^0 and z is also a core point. This successful approximation scheme is given by:

$$z^0_{ik} = \lambda\, z^0_{ik} + (1-\lambda)\, z^t_{ik} \qquad \forall\, i, k, t$$

where $0 < \lambda < 1$. As also observed by Papadakos [49] and Mercier *et al.* [45], $\lambda = 1/2$ gives the best results empirically.

Furthermore, the selected starting core point that demonstrated the best overall performance during the computer experiments is taken as:

$$z^0_{kk} = 0.5 \qquad \forall k \tag{36}$$

$$z^0_{ik} = 0.5/(n-1) \qquad \forall i \neq k \tag{37}$$

where $n = |N|$.

Proposition 1. *The point described by* (36) *and* (37) *is a valid core point, i.e.* $z^0 \in ri(Z^c)$.

Proof. In order to proof Proposition 1, it is necessary to show that z^0 can be expressed as convex combination of at least two integer feasible solutions. A point z is a convex combination of several other points z^r, $r = 1, \ldots, m$, if:

$$z = \sum_{r=1}^{m} \lambda^r z^r$$

$$\sum_{r=1}^{m} \lambda^r = 1$$

$$\lambda^r \geq 0 \qquad \forall\, r = 1, \ldots, m$$

It is possible to verify that z^0 respects the convex combination definition by making z^1 equal to the solution where all nodes are hubs and $z^r, r = 2, \ldots, (n+1)$, equal to the n possible solutions having a single installed hub, and establishing $\lambda^1 = 0.5 - 0.5/(n-1)$ and $\lambda^r = 0.5/(n-1)$, $r = 2, \ldots, (n+1)$. □

A sketch of the implemented algorithm is detailed in Algorithm 1 where UB, LB, ε, θ^*_{RMP} and ϑ^* are the upper bound, lower bound, stopping precision, the objective function optimal value of the RMP, and the objective function value of the current solution, respectively.

Algorithm 1: Generalized Benders Decomposition

```
1  Set $UB = +\infty$, $LB = -\infty$, $t = 1$, $z^0$.
2  If $(UB - LB) < \varepsilon$, then stop. Terminate, a near optimal solution has been obtained.
3  Calculate the values of $\beta^t$ using equation (26) and the values of $g^t$ using $z^0$.
4  Solve the subproblem $d_L(\beta)$ for $z^0$.
5  Add a new Benders cut to the RMP using (18) and $z^0$ in the place of $z^t$.
6  Solve the RMP (17)–(19), obtaining $\theta^*_{RMP}$ and the optimal values for the integer variables $z^t$.
7  Set $LB = \theta^*_{RMP}$ and update $z^t$ in subproblems $d_L(\beta)$ and $d_{NL}(\beta)$.
8  Calculate the values of $\beta^t$ using equation (26) and the values of $g^t$.
9  Solve the subproblem $d_L(\beta)$.
10 Compute the optimal value of $\phi(z^t)$ and set:
   $\vartheta^* = \phi(z^t) + \sum_k f_k\, z^t_{kk} + \sum_{k \neq i} (O_i + D_i)\, c_{ik}\, z^t_{ik}$.
11 Add a new Benders cut to the RMP using (18).
12 Update core point $z^0$.
13 If $\vartheta^* < UB$, then set $UB = \vartheta^*$.
14 Increment $t$ and go to **2**.
```

Usually, the solution time of the MP (line **6**) is usually much higher than the SP because of the integrality constraints. A common strategy to short it is to reduce the number of MP solved by embedding the pareto-optimal cuts generation procedure in a standard *Branch-And-Cut* framework, Algorithm 2.

Before starting the branch-and-cut, a cut pool Ω is set up. Some cuts can be added early in the tree to avoid the exploration of too many infeasible branch-and-bound nodes. However, adding too many unnecessary cuts can slow down the LP relaxation at each node. A good starting cut pool is obtained by solving the LP relaxation of the MP and adding five to ten cuts. In Algorithm 2, solving a node $\rho \in \Omega$ means solving the LP relaxation of MP, augmented with branching constraints of ρ and the cuts in pool Ω. The adopted implementation algorithm has been carried in the *Branch-And-Cut* framework of CPLEX 9.1 using the standard callback classes.

Algorithm 2: Branch-and-Cut Generalized Benders Decomposition

```
1  Set $\Omega = \{\rho\}$, where $\rho$ has no branching constraints.
2  If $\Omega$ is empty, then stop.
3  Select $\rho^* \in \Omega$.
4  $\Omega = \Omega \setminus \rho^*$.
5  Solve $\rho^*$, obtaining $z^*$, $\eta^*$.
6  Calculate the values of $\beta^*$ using equation (26) and the values of $g^*$ using $z^*$.
7  Solve the subproblem $d_L(\beta^*)$ for $z^*$.
8  Compute the optimal value of $\phi(z^*)$.
9  If $\eta^* > \delta\, \phi(z^*)$, then add a new Benders cut to the RMP using (18).
10 Branch, yielding in nodes $\rho'$ and $\rho''$, $\Omega = \Omega \cup \{\rho', \rho''\}$.
11 Go to **2**.
```

where δ is a scalar parameter set to 1.10. Line **9** only allows the inclusion of cuts that might improve the overall lower bound of the algorithm.

Although the GBD algorithm solves the SAHLP under congestion successfully, see the computational results Section 6, an specialized OA procedure is also presented in the next section. Both techniques are very competitive and clearly outperformed CPLEX 9.1 on the tests carried out.

4 THE OUTER APPROXIMATION METHOD

The OA method is a simple but effective technique based on a cutting plane approach for solving MINLPs [23, 31, 57]. The OA technique has been applied on structural optimization of flowsheet processes [35], on the synthesis of heat exchanger networks [4, 50], on general engineering processes [35] and on logistics applications [22, 36]. A general survey of the technique can be found at Grossmann & Kravanja [35].

The method is a coordination technique between a MP and a SP akin in essence to the GBD. Nevertheless the OA MP is written on the space of all the problem variables, opposing to the GBD projection onto the space of the complicating ones. This feature enhances the strength of the associated cutting planes and ensures the convergence in fewer iterations than the observed in GBD. As a drawback, the solution of the MP requires a greater computational effort worsening as the problem size increases.

In order to understand the development of the OA technique for the SAHLP under congestion, a general overview of the method is required. Given a MINLP in its most basic algebraic representation, where x and z are the sets of continuous and discrete variables, respectively,

$$f : \mathbb{R}^{q\times s} \mapsto \mathbb{R} \text{ and } g : \mathbb{R}^{q\times s} \mapsto \mathbb{R}^m$$

are two continuously differentiable functions, and Z and $\mathbf{X}$ are polyhedral sets:

$$\min f(z,x) \tag{38}$$

$$\text{s. t.: } g_j(z,x) \leq 0 \qquad \forall\, j = 1,\ldots,m \tag{39}$$

$$z \in Z,\ z \in \mathbb{Z}^q \tag{40}$$

$$x \in \mathbf{X} \tag{41}$$

It is possible to reduce this problem to a pure nonlinear program by choosing a fixed vector $z = z^h,\ z^h \in Z$, for some iteration h, yielding the following nonlinear SP:

$$\min f(z^h,x) \tag{42}$$

$$\text{s. t.: } g_j(z^h,x) \leq 0 \qquad \forall\, j = 1,\ldots,m \tag{43}$$

$$x \in \mathbf{X} \tag{44}$$

When solved, the above SP (42)–(44) permits to infer the gradient of the functions $f(z, x)$ and $g_j(z, x),\ \forall j$ at (z^h, x^h). If no further feasibility constraints are required, then a straightforward manipulation enables the problem (38)–(41) to be equivalent to a mixed integer program (MIP):

$$\min \xi \tag{45}$$

$$\text{s. t.: } \xi \geq f(z^h, x^h) + \nabla f(z^h, x^h)^T \begin{pmatrix} z - z^h \\ x - x^h \end{pmatrix} \qquad \forall\, h = 1, \ldots, H \tag{46}$$

$$0 \geq g(z^h, x^h) + \nabla g(z^h, x^h)^T \begin{pmatrix} z - z^h \\ x - x^h \end{pmatrix} \qquad \forall\, h = 1, \ldots, H \tag{47}$$

$$z \in Z,\ z \in \mathbb{Z}^q \tag{48}$$

$$x \in \mathbf{X} \tag{49}$$

$$\xi \in \mathbb{R} \tag{50}$$

where H is the maximum number of iterations and ξ is an objective function variable under-estimator.

Problem (45)–(50) is known as the OA MP. Constraints (46) and (47) are responsible for performing the OA of the objective function and the feasible region, respectively. When functions $g(z, x)$ are proper convex and a constraint qualification holds for every solution of (42)–(44), then constraints (47) are necessary and sufficient to outer approximate the feasible region.

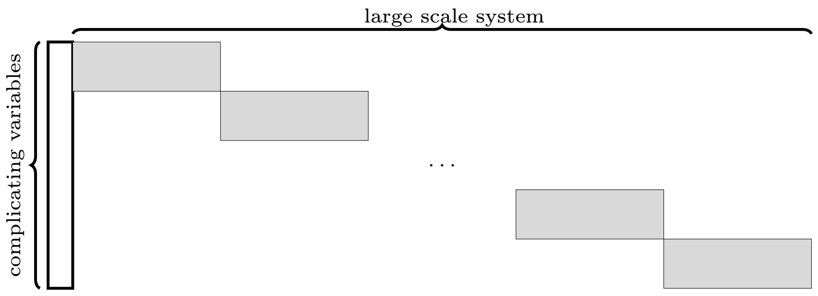
5 THE HYBRID OA/BENDERS TECHNIQUE

The LBs attained by the OA technique are greater or equal to the ones obtained by the GBD, resulting then in fewer iterations for convergence [23]. However, in order to have these better LBs, the OA's RMP has the number of variables and constraints greater than the GBD's RMP. Consequently, the size of the largest instance that the OA technique is capable of solving is much smaller than the largest of the GBD.

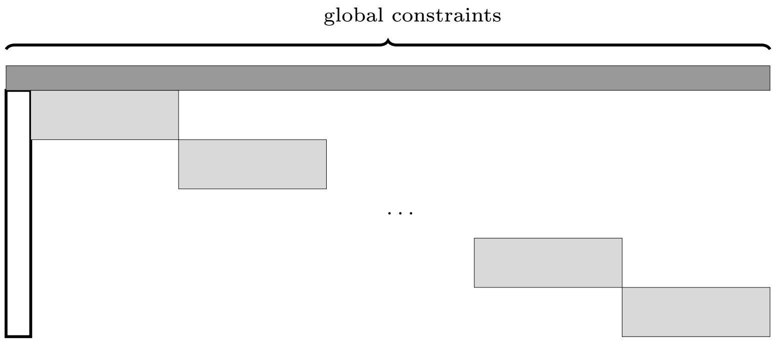
One way of circumventing the limitations of the OA's RMP in handling large size instances is to pay attention to its constraint matrix structure. Sometimes, when the RMP is a MILP, it can be efficiently solved by means of a Benders decomposition algorithm. This is specially true when the constraints of the problem being solved have a stair-case structure, see Figure 2(a), like the ones of location-transportation problems.

Unfortunately, this is not the case when capacity limits or congestion control are enforced by constraints that measure the level of utilization of the installed infra-structure. Normally, these constraints destroy the stair-case structure, see Figure 2(b), usually deprecating the performance of employed algorithms.

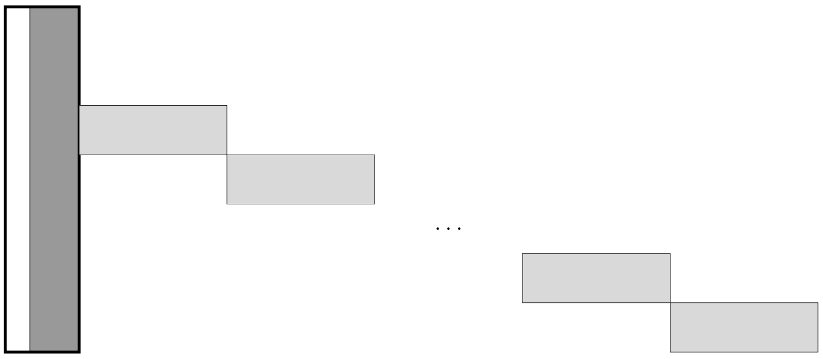
Nevertheless, whenever the MINLPs can be reformulated by adding new variables in order to separate the non-linearities from the large scale system, regaining thus a stair-case structure, see Figure 2(c), an hybrid strategy combining OA/Benders cuts can be efficiently used.



(a) Stair-case structure of typical location-transportation problems.



(b) Stair-case structure destroyed by global constraints imposed on common resources.



(c) Stair-case structure regained due to problem manipulation. The bold rectangle contains the new master problem variables.

Figure 2 – MINLP structures.

This strategy may allow the parallel solution of the SPs of OA and Benders methods. In other words, both OA and Benders cuts can be added to the RMP at the same time. Moreover, assuming that number of additional variables is much smaller than the number of variables of the large-scale system, the required solution time of the RMP might be shortened when compared with the standalone application of OA or GBD, because it has better bounds than the GBD's RMP and it has a smaller size than the OA's RMP.

In the case of formulation (1)–(9), the objective function is separable on the linear and nonlinear terms. Hence applying the OA technique only requires the replacement of $\tau_k(g_k)$ by ξ_k for each k on the objective function and the addition of constraints (46) in the form (52). These are responsible for the OA of function $\tau_k(g_k)$. The equivalent formulation of the OA MP can then be given as:

$$\min \sum_k [f_k\, z_{kk} + \xi_k] + \sum_{k \neq i} (O_i + D_i)\, c_{ik}\, z_{ik} + \sum_{i<j} \sum_{k \neq m} c_{ijkm}\, x_{ijkm} \tag{51}$$

$$\text{s.t.:}\quad (2) - (9)$$

$$\xi_k \geq \tau_k(g_k^h) + \beta_k^h (g_k - g_k^h) \qquad \forall\, k,\ h = 1, \ldots, H \tag{52}$$

$$\xi_k \geq 0 \qquad \forall\, k \tag{53}$$

Constraints (47) are not present in formulation (51)–(53), because the coupling constraints (5) are linear, making unnecessary thus to perform a OA of the feasible region. Furthermore, as expected, this formulation is still too large to be efficiently solved. The large size of x variables set might represent a computer burden during the solution of the OA MP. One way of addressing this drawback is to project these variables out provided that some manipulations are carried out. The idea here is to enable the solution of the OA MP by means of a Benders decomposition algorithm.

Observing the role of variables x_{ijkk} on constraints (5), it is possible to replace it using additional variables $y_{ijk} \geq 0$ and some coupling constraints on y, z and x. The equation (5) responsible for computing g_k variables is then reformulated as follows:

$$\sum_i (O_i + D_i) z_{ik} - \sum_{i<j} (w_{ij} + w_{ji}) y_{ijk} = g_k \qquad \forall\, k \tag{54}$$

where the values of y_{ijk} can be computed using the following coupling constraints:

$$y_{ijk} \geq z_{ik} + z_{jk} - 1 \qquad \forall\, i < j, k$$

$$y_{ijk} \leq z_{ik} \qquad \forall\, i < j, k$$

$$y_{ijk} \leq z_{jk} \qquad \forall\, i < j, k$$

$$y_{ijk} \geq 0 \qquad \forall\, i < j, k$$

Moreover, in order to avoid the degradation of linear programming bounds, constraints (3) and (4) are now rewritten as:

$$y_{ijk} + \sum_m x_{ijkm} = z_{ik} \qquad \forall\, i < j, k$$

$$y_{ijm} + \sum_k x_{ijkm} = z_{jm} \qquad \forall\, i < j, m$$

These manipulations allow the decomposition of the OA MP using the standard scheme of Benders decomposition. It is now possible to project the x variables out, making the OA MP more manageable. For fixed values of y^t and z^t at some iteration t, the Benders primal SP is given by:

$$\min \sum_{i<j} \sum_{k \neq m} c_{ijkm}\ x_{ijkm}$$

s. t.:

$$\sum_m x_{ijkm} = \left(z^t_{ik} - y^t_{ijk}\right) \qquad \forall i < j,\ k \tag{55}$$

$$\sum_k x_{ijkm} = \left(z^t_{jm} - y^t_{ijm}\right) \qquad \forall i < j,\ m \tag{56}$$

$$x_{ijkm} \geq 0 \qquad \forall i < j,\ \forall k, m$$

Associating the dual variables $u \in \mathbb{R}$ and $v \in \mathbb{R}$ to constraints (55) and (56), respectively, the following dual Benders SP is obtained for a given $i - j$ pair:

$$\max \sum_k u_{ijk}\ \left(z^t_{ik} - y^t_{ijk}\right) + \sum_m v_{ijm}\ \left(z^t_{jm} - y^t_{ijm}\right) \tag{57}$$

$$\text{s. t.: } u_{ijk} + v_{ijm} \leq c_{ijkm} \qquad \forall k \neq m \tag{58}$$

$$u_{ijk} + v_{ijk} \leq 0 \qquad \forall\, k \tag{59}$$

$$u_{ijk} \in \mathbb{R} \qquad \forall\, k \tag{60}$$

$$v_{ijm} \in \mathbb{R} \qquad \forall\, m \tag{61}$$

Using (57) and the help of an auxiliary variable $\eta \geq 0$, Benders cuts (64) can be derived, where u^t_{ijk} and v^t_{ijm} are the optimal values of the dual variables of iteration t. Once again, no further feasibility constraints are required, being the dual Benders SP always bounded. Moreover, it is also possible to eliminate the variables g_k from the OA MP using equations (54). Therefore the resulting reformulated OA MP is now written as:

$$\min \sum_k [f_k\ z_{kk} + \xi_k] + \sum_{k \neq i} (O_i + D_i)\ c_{ik}\ z_{ik} + \eta \tag{62}$$

$$\text{s. t.: } \sum_k z_{ik} = 1 \qquad \forall\, i \tag{63}$$

$$\eta \geq \sum_{i<j} \sum_k u^t_{ijk}(z_{ik} - y_{ijk}) - \sum_{i<j} \sum_m v^t_{ijm}(z_{jm} - y_{ijm}) \qquad \forall\, t = 1, \ldots, T \tag{64}$$

$$\xi_k + \sum_{i<j} \beta^h_k (w_{ij} + w_{ji}) y_{ijk}$$

$$- \sum_i \beta^h_k (O_i + D_i) z_{ik} \geq [\tau_k(g^h_k) - \beta^h_k g^h_k] \qquad \forall\, k,\ h = 1, \ldots, H \tag{65}$$

$$z_{ik} \leq z_{kk} \qquad \forall\, i \neq k \tag{66}$$

$$y_{ijk} \geq z_{ik} + z_{jk} - 1 \qquad \forall\, i < j, k \qquad (67)$$

$$y_{ijk} \leq z_{ik} \qquad \forall\, i < j, k \qquad (68)$$

$$y_{ijk} \leq z_{jk} \qquad \forall\, i < j, k \qquad (69)$$

$$y_{ijk} \geq 0 \qquad \forall\, i < j, k \qquad (70)$$

$$\eta \geq 0 \qquad (71)$$

$$\xi_k \geq 0 \qquad \forall\, k \qquad (72)$$

$$z_{ik} \in \{0, 1\} \qquad \forall\, i, k \qquad (73)$$

In formulation (62)–(73), $\tau_k(g_k^h)$ is calculated using equation (54) for the pair (z^h, y^h), while the quantity $\beta_k^h\ g_k^h$ is computed recalling equation (26). It is important to remark that the final form of the OA MP enables the parallel solution of SP involving the pair $(g,\ \beta)$ and the $(u,\ v)$ variables. Whenever a new solution (z, y) is available, new Benders cuts (64) or new OA cuts (65) can be added to the MP in any order.

Moreover, all the results concerning pareto-optimal cuts and core points can be reused here, since a starting core point on the y space can be readily obtained by making $y_{ijk}^0 = z_{ik}^0\ z_{jm}^0$. But even when using pareto-optimal cuts, the solution of (62)–(73) does not imply the generation of cuts as strong as those obtained by the classical OA MP, i.e. without projecting out the x variables. However, it provides a good trade-off between the strength of the cuts and the computational effort on solving the MP.

A sketch of the implemented algorithm is detailed in Algorithm 3, where θ_{OAMP}^* and Φ_{OASP}^* are the objective function optimal value of the OA MP and the OA SP, respectively.

At lines **3** and **13** of Algorithm 3, the OA MP is solved after relaxing and imposing the integrality constraints (73), respectively. In lines **3** through **12**, Benders and OA cuts are added to the OA MP while the difference between the under-estimator variable and the values of the SP objective function are greater than a threshold δ.

In a similar way to Algorithm 1, the OA MP resolution time is usually much larger than the SP time, due to the integrality constraints. Therefore, the generation of Benders and OA cuts is embedded in a standard branch-and-cut framework, akin to Algorithm 2. The next section presents computation experiments comparing the deployed solution strategies.

6 COMPUTATIONAL EXPERIMENTS

In order to assess the performance of the proposed algorithms and the impact of the single allocation feature of this class of problems under hub congestion, three sets of computational experiments were designed.

The first one demonstrates how the hub infrastructure changes when congestion costs are considered and that, depending on how the congestion cost function parameters are set, it is possible to solve to optimality large instances by using the GBD algorithm (Algorithm 1).

Algorithm 3: Outer Approximation Method

```
1  Set $UB = +\infty$, $LB = -\infty$, $t = 1$, $h = 1$, $z^0$ and $y^0$.
2  If $(UB - LB) < \varepsilon$, then stop. Terminate, a near optimal solution has been obtained.
3  Solve the OA MP (62)–(72), obtaining $\theta^*_{OAMP}$ and the optimal values for the variables $z^t$, $y^t$.
4  Set $LB = \theta^*_{OAMP}$.
5  Update core point $z^0$ and $y^0$
6  Use $z^0$ and $y^0$ instead of $z^t$ and $y^t$, respectively, in subproblem (57)–(61).
7  Solve the subproblem (57)–(61) using core points $z^0$ and $y^0$.
8  Add a new Benders cut to the OA MP using (64).
9  Calculate the values of $\beta^h$ using equation (26) and the values of $g^h$.
10 Add a OA cut to the OA MP using (65)
11 Increment $t$ and $h$.
12 If $(\eta^* - \Phi^*_{OASP}) > \delta$ then goto **3**
13 Solve the OA MP (62)–(73), obtaining $\theta^*_{OAMP}$ and the optimal values for the variables $z^h$, $y^h$.
14 Set $LB = \theta^*_{OAMP}$.
15 Solve the subproblem (57)–(61).
16 Add a new Benders cut to the OA MP using (64).
17 Update core point $z^0$ and $y^0$
18 Solve the subproblem (57)–(61).
19 Add a new Benders cut to the OA MP using (64).
20 Calculate the values of $\beta^h$ using equation (26) and the values of $g^h$.
21 Add a OA cut to the OA MP using (65)
22 $\vartheta^* = \sum_k \left[ f_k z^h_{kk} + \tau_k(g^h_k) \right] + \sum_{k \neq i} (O_i + D_i)\, c_{ik}\, z^h_{ik} + \sum_{i<j} \sum_{k \neq m} c_{ijkm}\, z^h_{ik}\, z^h_{jm}$
23 If $\vartheta^* < UB$, then set $UB = \vartheta^*$.
24 Increment $t$ and $h$ and go to **2**
```

The second set verifies how a given instance becomes harder to solve as the congestion and the installation costs vary. It infers that the instances become harder to solve when the two effects are combined. Hence being important to deploy a more elaborate GBD: Algorithm 2. This experiment also shows that the computational burden is not only explained by the instance size, but by the parameters settings too.

Finally, when the combination of the congestion and the installation cost parameters are stressed, a further approach is required. Thus a comparison of the OA Algorithm 3 and of the GBD Algorithm 2 is presented on the final set of experiments.

In all the experiments, the well-known Australian Post (AP) standard data set introduced by [28,30] is used. The names of the instances are coded as $APn.\alpha$, where n is the number of nodes and $\alpha = \{2, 4, 6, 8\}$ represents the selected discount factors 0.2, 0.4, 0.6 and 0.8, respectively. These instances have sizes ranging from 10 to 200 nodes, Euclidean distances to represent the transportation costs, and installation costs for the first 50 nodes only.

As $K \equiv N$ is considered in the experiments, hub fixed costs were generated for all instances using a Gaussian distribution with average equal to f_o and variance set to 40% to mimic how different installation costs vary in realistic problems. Here f_o represents the scaled difference in objective value between a scenario in which a virtual hub is located in the center of mass and a scenario in which all nodes are hubs, as introduced by [25]. Further, as done by [25], the nodes with higher flows are selected to have the higher fixed costs, hardening in general the selection of potential hubs. For alternative articles considering this test bed and the methodological generation of hub set-up costs refer to [10–12,25,26,48].

The nominal capacity of each hub k was generated by taking the total demand of the node $(O_k + D_k)$ added to a random fraction (U[15%, 50%]) of the total demand, recalling that the hub total traffic does not drop linearly with the number of installed hubs, equation (5). The algorithms were implemented in $C++$ using the IBM CPLEX 11 Concert Technology under a Linux operating system. The experiments were run on a regular PC desktop with a Intel Core 2 Quad $3.2GHz$ processor and $8Gb$ of RAM, and had 72, 000 seconds as a time limit.

In the first set of experiments, the parameters a and b were set to 0.0001 and 2, respectively. The obtained results are presented in Table 1. The entry *LP* plots the number of cycles where the GBD MP was solved disregarding the integrality constraints (hot-start cycles). The entry *Integer* shows the number of additional integer cycles needed to prove optimality. Columns *Solution Time* and *Installed Hubs* show the total computational time required and the number of installed hubs.

An interesting feature of this class of hub-and-spoke problems can be observed in Table 1 when compared to the results reported in Camargo *et al.* [11]. The single allocation systems are much more affected by congestion effects than the multiple ones. This is expected, since, in the present case, there are no alternative paths for a given origin destination pair once the network is designed. This enhanced sensitivity can be deduced from Table 1 where an addition of 1.35 hubs on average was required to handle the congestion effects even for very small congestion cost parameters.

In this sense, the economies of scale play a fundamental role in the network design, once they allow the affordance of the congestion extra costs prior to installing additional infrastructure. Further, the computational burden to prove optimality grows as the efficiency of the economies of scale in counterbalancing the congestion effects is reduced. Figure 3 shows this trend by plotting the log scale of the solution time of instances AP40, AP50 and AP70 under congestion.

Observing the rows with small economies of scale, APn.8, the required number of additional hubs to manage the congestion costs is significant, raising the following question: If the network is unable to provide large discounts on inter-hub connections due to the lack of transportation technology or other environmental reasons, is it a good strategy to design such networks using the single allocation approach? Future research may then address the economies of scale as a function of the total flow on the inter-hub connections and the associated impact on the network design under hub congestion.

Table 1 – Obtained results for Algorithm 1.

	No congestion costs				Accounting congestion costs			
Instance	Benders Cycles		Time	Installed	Benders Cycles		Time	Installed
	LP	Integer	[s]	Hubs	LP	Integer	[s]	Hubs
AP10.2	5	1	0.05	2	5	58	6.96	4
AP10.4	5	1	0.05	2	5	113	24.14	4
AP10.6	5	1	0.05	3	5	36	2.96	4
AP10.8	5	2	0.07	4	5	14	0.56	6
AP20.2	5	3	0.08	3	5	19	6.96	4
AP20.4	5	1	0.36	2	5	26	10.91	4
AP20.6	5	2	0.59	4	5	13	3.46	5
AP20.8	5	3	1.16	3	5	29	11.71	5
AP30.2	5	1	2.5	3	5	10	13.2	4
AP30.4	5	2	2.87	2	5	45	122.42	4
AP30.6	5	2	2.94	2	7	9	13.22	3
AP30.8	5	4	3.38	2	5	13	20.62	4
AP40.2	5	1	4.93	4	5	8	20.19	5
AP40.4	5	2	5.35	3	5	16	49.8	4
AP40.6	5	2	6.01	3	5	13	36.8	4
AP40.8	5	3	7.68	3	5	21	81.02	4
AP50.2	10	1	33.78	3	10	9	82.07	4
AP50.4	10	1	35.04	3	10	19	170.05	3
AP50.6	10	1	19.24	2	10	71	1155.74	4
AP50.8	5	5	31.07	2	5	95	2548.9	4
AP70.2	5	1	66.65	3	5	1	67.9	4
AP70.4	5	1	69.73	3	5	13	312.58	4
AP70.6	5	2	68.24	3	5	25	753.45	4
AP70.8	5	3	63.93	3	5	44	1301.2	5
AP100.2	5	1	213.28	3	5	6	528.97	4
AP100.4	5	2	268.02	3	5	9	748.88	5
AP100.6	5	4	538.37	2	5	35	2134.25	4
AP100.8	10	3	608.2	2	10	60	17957.64	3
AP150.2	5	1	2151.49	3	5	14	9591.41	4
AP150.4	5	1	2728.8	3	5	37	37795.5	4
AP200.2	5	1	13183.19	4	5	10	46230.17	5

For the second set of experiments, the instance AP10.2 was selected in order to show how a given instance becomes hard to solve as the congestion costs are increased. The parameter b was set to 2, while the congestion cost function parameter a was scaled from 0.0001 to 100. Instances with low congestion cost parameter a tend to have low solution times. When the congestion costs are increased, the computer effort is augmented. However, if the congestion costs are very high, the computational burden to solve the associated instance is not worsened in the same manner.

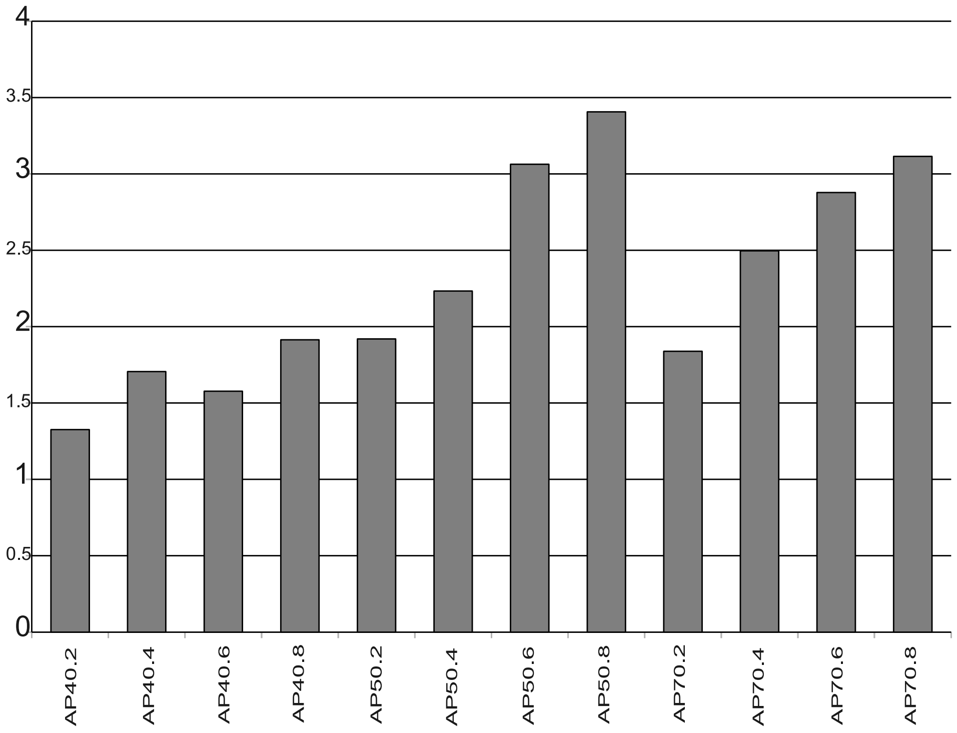


Figure 3 – Computer burden trend (log(Time[s])).

This can be seen in Figure 4 as the computational effort stabilizes after a given threshold of the parameter a is trespassed. In other words, all the economies of scale were preserved, at the cost of an additional investment on the network infrastructure.

This phenomenon suggests that this class of problems has its inherent difficulty associated to other components than the congestion cost parameters, such as the hub installation costs. In order to investigate the role of these costs, the same instance AP10.2 was used, but having the parameters a and b fixed to 0.001 and 2, respectively. A scaling factor multiplying the installation costs was adopted ranging from 0.5 to 20.

As happened in the later case, Figure 5 shows that after a given threshold of the increased hub installation costs, the computing time to prove the instance optimality is stabilized. Hence from Figures 4 and 5, is possible to infer that the problem under study becomes harder to solve if a given instance presents high hub installation costs and also a very aggressive congestion cost function.

In the next experiments, for fixed parameters $a = 0.001$ and $b = 2$, a comparison of computing times and number of integer cycles to attain optimality of Algorithm 1 and 2 is depicted in the fourth, fifth, seventh and eighth columns, respectively, of Table 2. Observe that different installation scaling factors were adopted, as indicated in the second column, making these instances harder to solve than the ones present in Table 1.

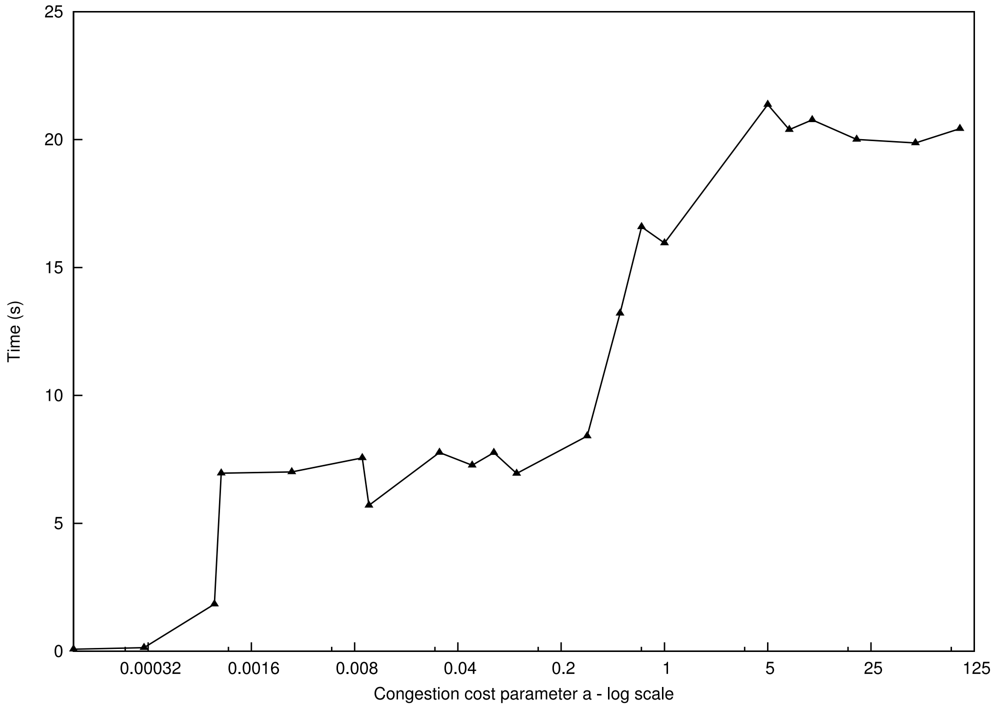


Figure 4 – Computer effort *versus* congestion costs.

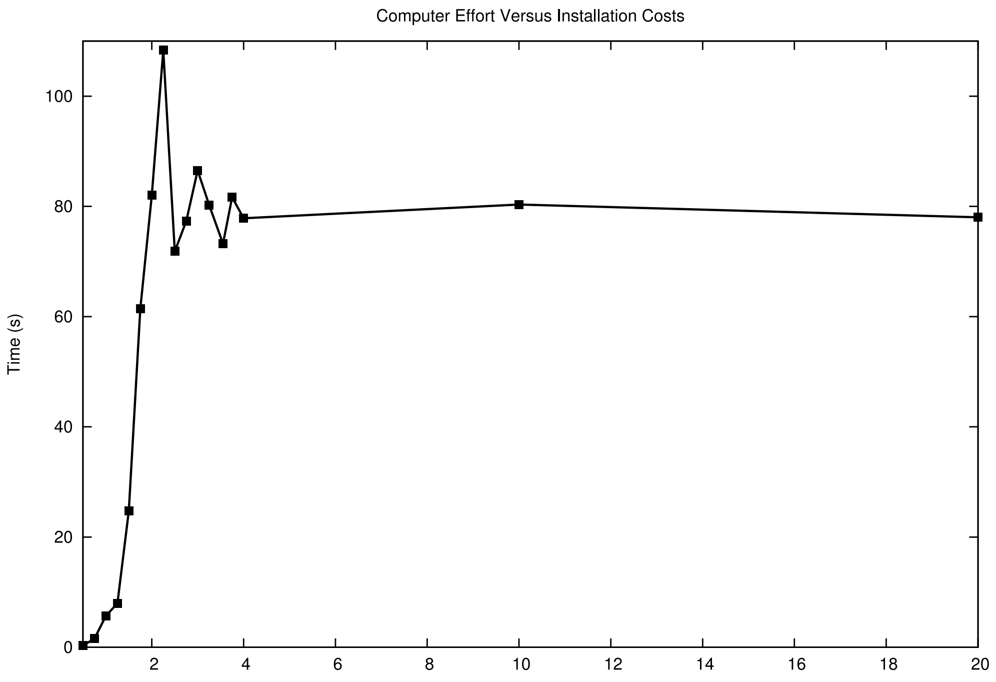


Figure 5 – Computer effort *versus* installation costs.

Algorithm 1 requires 7 times more computing time and 11 times more integer cycles than Algorithm 2, on average. It is also interesting to remark that there are computing times in Table 2 that are comparable to the ones associated with instances of 150 and 200 nodes of Table 1. Definitely indicating that the complexity of this problem it is not only a matter of instance size.

Table 2 – Algorithm 1 *versus* Algorithm 2.

Instance	f_k scaling factor	Algorithm 1 Benders Cycles LP	Algorithm 1 Benders Cycles Integer	Algorithm 1 Time [s]	Algorithm 2 Benders Cycles LP	Algorithm 2 Benders Cycles Integer	Algorithm 2 Time [s]
AP10.2	1.5	5	83	25.18	5	11	14.81
AP10.4	1.5	5	131	90.74	5	12	29.02
AP10.6	1.2	5	87	34.88	5	8	10.61
AP10.8	1.2	5	212	310.04	5	14	38.35
AP20.2	3.5	5	199	4486.89	5	13	415.82
AP20.4	1	5	169	1369.07	5	14	310.28
AP20.6	0.8	5	226	1470.8	5	23	315.33
AP20.8	0.6	5	304	5209.17	5	16	232.86
AP30.2	2.5	5	152	3090.3	5	20	1770.87
AP30.4	2	10	205	11005.81	10	12	2539.44
AP30.6	1.5	10	206	23195.05	10	12	1401.31
AP30.8	1.5	10	209	33624.85	10	11	2874.71
AP40.2	3	10	154	8599.01	10	13	3547.57
AP40.4	2	10	179	10456.33	10	28	1976.87
AP40.6	1.5	10	136	4202.79	10	18	1725.92
AP40.8	1.5	10	160	30842.53	10	14	1405.04
AP50.2	6	10	92	6693.18	10	14	3499.35
AP50.4	5	10	92	9828.01	10	14	1602.16
AP50.6	3	10	100	6030.94	10	18	2946.36
AP50.8	2	10	161	49098.57	10	16	5899.16

Furthermore, after observing the evolution of the upper and lower bounds of both algorithms during the solution process of instance AP40.6 of Table 2 (refer to Figure 6), the main advantage of Algorithm 2 over Algorithm 1 is its ability to reduce the tail-off effect. In order to close 1% of optimality gap, Algorithm 2 takes close to a 1000 seconds or 42% of the total time, against 3, 600 seconds or 87% of Algorithm 1. An akin behavior occurs on all the other instances.

The tail-off effect is more pronounced if the instances combine the two characteristics pointed above, large hub installation and an aggressive congestion costs, making the deployment of Algorithm 2 an interesting solution approach. However, when this combination is stressed-out even more, a different strategy is made necessary, as demonstrated in the third and final set of experiments.

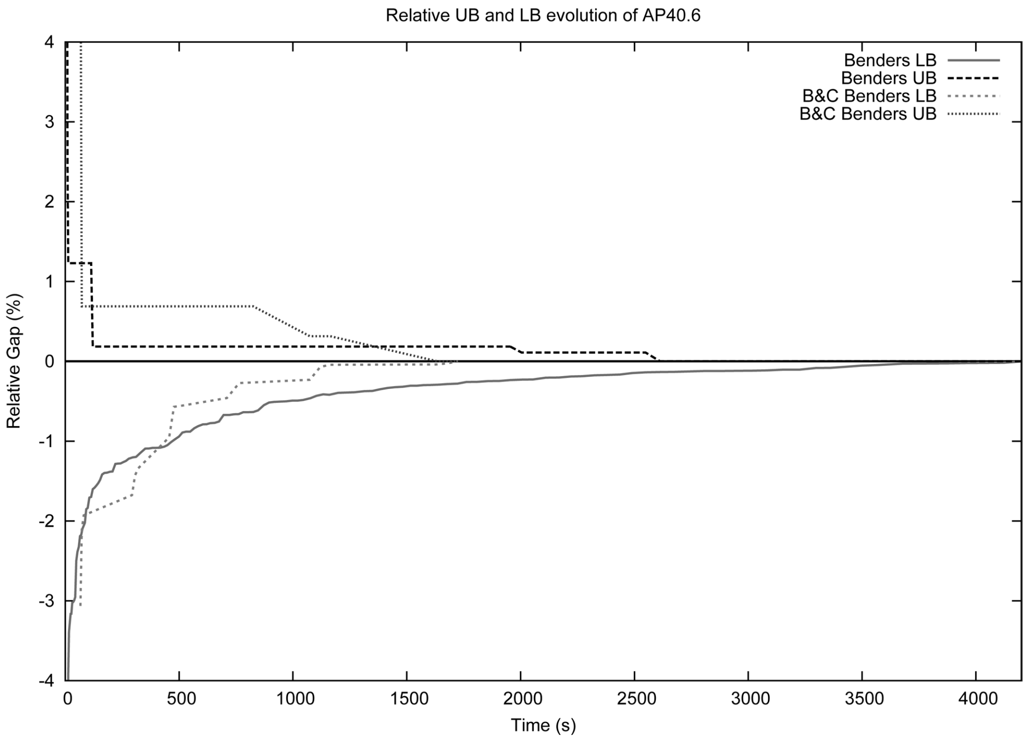


Figure 6 – Convergence curve for Algorithms 1 and 2.

In these final experiments, the congestion cost parameter is set to 0.01 and the installation cost scaling factors were adopted, as indicated in the second column of Table 3. Recall that in the first two sets of experiments, parameter a was set to 0.0001 and 0.001, respectively. In other words, the combined effect of having high congestion and installation costs makes these instances even harder to solve.

Table 3 presents the comparison results of Algorithms 2 and 3. The Algorithm 3 or the OA technique demonstrates a superior performance on these very hard instances, being 3.3 times faster and taking 3 times less integer cycles on average to prove optimality than the Algorithm 3. Notice that Algorithm 2 fails to prove optimality in four instances, ending with an average optimality gap close to 1% (instances AP70 and AP100).

Although, on one hand, Algorithm 3 demonstrates a remarkable performance tackling hard instances, on the other hand, the solution effort for solving each OA MP is still very high, even after projecting the x variables out to reduce its dimension. In this sense, this technique is preferable only if large optimality gaps of the first integer cycles of Algorithm 2 are detected as can be seen in Table 4.

In order to clarify the former statement, a direct comparison of the three implemented algorithms is presented in Table 4. In this experiment, instances AP10.2 and AP20.4 were selected, being

Table 3 – Algorithm 2 *versus* Algorithm 3.

Instance	f_k scaling factor	Algorithm 2			Algorithm 3		
		Main Cycles		Time	Main Cycles		Time
		LP	Integer	[s]	LP	Integer	[s]
AP10.2	3	5	11	328.67	3	5	98.64
AP10.8	2.5	5	16	568.08	3	4	110.81
AP20.2	2.5	5	16	2283.76	3	6	418.01
AP20.8	2.5	5	11	3108.63	2	3	1330.92
AP30.2	2.5	5	20	7976.71	2	5	4237.58
AP30.8	3	5	8	6389.15	2	6	2612.24
AP40.2	3.5	5	18	17789.13	2	6	4220.5
AP40.8	2.5	5	11	8734.13	2	5	1563.89
AP50.2	5	5	12	29335.13	2	4	8345.67
AP50.8	4	5	17	48764.24	2	6	25678.30
AP70.2	7	5	12	> 72000	2	4	33345.67
AP70.8	6	5	17	> 72000	2	6	21236.13
AP100.2	10	5	12	> 72000	2	4	30041.07
AP100.8	8	5	17	> 72000	2	6	27849.51

the hub installation scaling factor and congestion parameter a varied according to the second and third columns, respectively.

The boldfaced entries in Table 4 represents the minimum computational effort for the instances solved. The deployment of Algorithm 3 is more interesting than the others when the optimality gaps of the first integer cycles of Algorithms 1 and 2 are greater than 40% on average for the tested problems.

A final comment is made here necessary. For the authors knowledge, there is no report on the literature that OA methods are able to solve such large scale problems. Notice that the instances AP100.2 and AP100.8 have 10,000 integer variables and 49,500,000 continuous variables. Therefore, solving the OA master programs by using the Benders decomposition method is a promising technique, *i.e.* whenever the problem structure is amenable and the GBD is not a good alternative.

7 FINAL REMARKS

Addressing congestion effects by explicitly considering them as costs on the objective function yields a more realistic modeling approach, specially when suitable nonlinear functions are selected. Once the rapid growth of the congestion costs is portrayed, a flow load balancing over the installed infrastructure is induced, leading thus to a superior network design.

This research presents two different techniques for solving large scale instances of single allocation hub location problem under congestion: the GBD technique and the OA method. Both

Table 4 – A direct comparison of the three implemented algorithms.

Instance	Instance Parameters		Algorithm 1	Algorithm 2	Algorithm 3	Initial
	f_k scaling factor	congestion parameter a	Time [s]	Time [s]	Time [s]	optimality gap [%]
AP10.2	1.00	0.0001	**6.98**	10.19	11.19	8.17
	1.00	0.0005	**7.34**	9.03	10.70	11.31
	1.00	0.0010	6.19	**3.14**	5.15	26.42
	1.00	0.0050	11.26	**5.69**	6.48	18.73
	2.00	0.0010	38.27	15.01	**8.22**	54.02
	2.00	0.0100	84.77	59.19	**11.95**	73.02
	3.00	0.1000	249.59	195.94	**86.71**	86.15
AP20.4	1.00	0.0001	**10.91**	14.39	20.48	5.12
	1.00	0.0005	57.15	**33.99**	50.90	19.03
	1.00	0.0010	1199.79	**293.38**	566.50	33.07
	1.00	0.0050	1429.65	**376.77**	569.71	43.05
	2.00	0.0010	46474.47	1393.10	**303.96**	59.93
	2.00	0.0100	>72000	24286.90	**2267.52**	78.84
	3.00	0.1000	>72000	>72000	**43747.74**	85.23

are capable of solving instances up to 100 nodes, being the GBD (Algorithm 1) able to manage larger instances of 150 and 200 nodes.

Finally, the complexity on this class of problems is not only associated with the size of the instances. When an aggressive congestion cost function and large hub installation costs are combined, the computational effort to solve these instances may become remarkable. In these cases, the addition of pareto-optimal cuts on the MP branch-and-bound tree is an interesting way to reduce the tail-off effect, improving the overall performance of both of the portrayed methods: GBD and OA. In general, on one hand the GBD method is more scalable with instance size, on the other hand, the OA technique is less affected by the tail-off effect.

ACKNOWLEDGMENTS

The authors' research has been funded by Conselho Nacional de Pesquisa – CNPq, project 480757/2008-9.

REFERENCES

[1] Abdinnour-Helm S. 1998. A hybrid heuristic for the uncapacitated hub location problem. *European Journal of Operational Research*, **2-3**(106): 489–499.

[2] Abdinnour-Helm S. 2001. Using simulated annealing to solve the p-hub median problem. *International Journal of Physical Distribution & Logistics Management*, **31**(3): 203–220.

[3] Abdinnour-Helm S & Venkataramanan MA. 1998. Solution approaches to hub location problems. *Annals of Operations Research*, **78**: 31–50.

[4] Adjiman CS, Androulakis IP & Floudas CA. 1997. Global optimization of MINLP problems in process synthesis and design. *Computers & Chemical Engineering*, **21**(Suppl. S): S445–S450.

[5] Alumur SA & Kara BY. 2008. Network hub location problems: The state of the art. *European Journal of Operational Research*, **190**: 1–21.

[6] Alumur SA, Kara BY & Karasan OE. 2009. The design of single allocation incomplete hub networks. *Transportation Research Part B*, **43**: 936–951.

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

[8] Aykin T. 1995. Network policies for hub-and-spoke systems with applications to the air transportation system. *Transportation Science*, **29**: 201–221.

[9] Benders JF. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerisch Mathematik*, **4**: 238–252.

[10] Camargo RS, Miranda Jr G de & Luna HP. 2008. Benders decomposition for the uncapacitated multiple allocation hub location problem. *Computers and Operations Research*, **35**: 1047–1064.

[11] Camargo RS, Miranda Jr G de, Ferreira R & Luna HP. 2009. Multiple allocation hub-and-spoke network design under hub congestion. *Computers and Operations Research*, **36**: 3097–3106.

[12] Camargo RS, Miranda Jr G de & Luna HP. 2009. Benders decomposition for hub location problems with economies of scale. *Transportation Science*, **43**: 86–97.

[13] Camargo RS, Miranda Jr G de & Ferreira RPM. 2011. A hybrid outer-approximation/benders decomposition algorithm for the single allocation hub location problem under congestion. *Operations Research Letters*, **39**(5): 329–337.

[14] Campbell JF. 1994. Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, **72**: 387–405.

[15] Campbell JF. 1994. A survey of network hub location. *Studies in Locational Analysis*, **6**: 31–49.

[16] Campbell JF, Ernst AT & Krishnamoorthy M. 2002. Hub location problems. In: Drezner Z and Hamacher HW (editors), Facility Location: Applications and Theory, chapter 12, pages 373–407. Springer, 1st edition.

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

[18] Contreras I, Fernandez E & Marn A. 2010. The tree of hubs location problem. *European Journal of Operational Research*, **202**: 390–400.

[19] Cordeau JF, Soumis F & Desrosiers J. 2001. Simultaneous assignment of locomotives and cars to passenger trains. *Operations Research*, **49**(4): 531–548.

[20] Costa M da G, Captivo ME & Clímaco J. 2008. Capacitated single allocation hub location problema bi-criteria approach. *Computers and Operations Research*, **35**(11): 3671–3695.

[21] Cunha CB & Silva MR. 2007. 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.

[22] DESAI J & SEN S. 2010. A global optimization algorithm for reliable network design. *European Journal of Operational Research*, **200**(1): 1–8.

[23] DURAN MA & GROSSMANN IE. 1986. An outer-approximation algorithm for a claar of mixed-integer nonlinear programs. *Mathematical Programming*, **36**: 307.

[24] EBERY J. 2001. Solving large single allocation p-hub problems with two or three hubs. *European Journal of Operational Research*, **128**(2): 447–458.

[25] EBERY J, KRISHNAMOORTHY M, ERNST A & BOLAND N. 2000. The capacitated multiple allocation hub location problema: Formulations and algorithms. *European Journal of Operational Research*, **120**: 614–631.

[26] ELHEDHLI S & HU FX. 2005. Hub-and-spoke network design with congestion. *Computers and Operations Research*, **32**: 1615–1632.

[27] ELHEDHLI S & WU H. 2010. A lagrangean heuristic for hub-and-spoke system design with capacity selection and congestion. *INFORMS Journal on Computing*, **22**(2): 282–296.

[28] ERNST AT & KRISHNAMOORTHY M. 1996. Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, **4**: 139–154.

[29] ERNST AT & KRISHNAMOORTHY M. 1998. An exact solution approach based on shortest-paths for p-hub median problems. *Journal on Computing*, **10**: 149–162.

[30] ERNST AT & KRISHNAMOORTHY M. 1999. Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, **86**: 141–159.

[31] FLETCHER R & LEYFER S. 1994. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, **66**: 327.

[32] GEOFFRION AM & GRAVES GW. 1974. Multicommodity distribution system design by Benders decomposition. *Management Science*, **20**: 822–844.

[33] GEOFFRION AM. 1972. Generalized Benders decomposition. *Journal of optimization Theory and Applications*, **10**(4): 237–260.

[34] GILLEN D & LEVINSON D. 1999. The full cost of air travel in the california corridor. *Transportation Research Record: Journal of the Transportation Research Board*, **1662**: 1–9.

[35] GROSSMANN IE & KRAVANJA Z. 1995. Mixed-integer nonlinear programming techniques for process systems engineering. *Computers & Chemical Engineering*, **19**: 189–204.

[36] HUANG SM, BATTA R & NAGI R. 2005. Distribution network design: Selection and sizing of congested connections. *Naval Research Logistics*, **52**(8): 701–712.

[37] KARA BY & TANSEL BC. 2003. The latest arrival hub location problem. *Management Science*, **47**: 1408–1420.

[38] KLINCEWICZ JG. 1991. Heuristics for the p-hub location problem. *European Journal of Operational Research*, **53**: 25–37.

[39] KLINCEWICZ JG. 1992. Avoiding local optima in the p-hub location problem using tabu search and grasp. *Annals of Operations Research*, **40**: 283–302.

[40] KLINCEWICZ JG. 1996. A dual algorithm for the uncapacitated hub location problem. *Location Science*, **4**(3): 173–184.

[41] LABBÉ M, YAMAN H & GOURDIN E. 2005. A branch and cut algorithm for hub location problems with single assignment. *Mathematical Programming: Series A*, **102**: 371–405.

[42] MAGNANTI TL & WONG RT. 1981. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, **29**(3): 464–483.

[43] MAGNANTI TL, MIRCHANDANI P & WONG RT. 1986. Tailoring Benders decomposition for uncapacitated network design. *Mathematical Programming Study*, **26**: 112–154.

[44] MARIANOV V & SERRA D. 2003. Location models for airline hubs behaving as M/D/c queues. *Computer and Operations Research*, **30**(7): 983–1003. ISSN 0305-0548.

[45] MERCIER A, CORDEAU JF & SOUMIS F. 2005. Acomputational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers and Operations Research*, **32**(6): 1451–1476.

[46] O'KELLY ME. 1987. A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, **32**: 393–404.

[47] O'KELLY ME. 1998. A geographer's analysis of hub-and-spoke networks. *Journal of Transport Geography*, **3**(6): 171–186.

[48] O'KELLY ME & BRYAN DL. 1998. Hub location with flow economies of scale. *Transportation Research Part B*, **32**(8): 605–616.

[49] PAPADAKOS N. 2008. Practical enhancements to the Magnanti-Wong method. *Operations Research Letters*, **36**: 444–449.

[50] PAULES GE & FLOUDAS CA. 1992. Stochastic-programming in process synthesis – A 2-stage model wiht MINLP recourse for multiperiod heat-integrated distillation sequences. *Computers & Chemical Engineering*, **16**(3): 189–210.

[51] PIRKUL H & SCHILLING DA. 1998. An efficient procedure for designing single allocation hub and spoke systems. *Management Science*, **44**(12): 235–242.

[52] RODRIGUEZ V, ALVAREZ MJ & BARCOS L. 2007. Hub location under capacity constraints. *Transportation Research Part E*, **43**: 495–505.

[53] SILVA MR & CUNHA CB. 2009. New simple and efficient heuristics for the uncapacitated single allocation hub location problem. *Computers and Operations Research*, **36**(12): 3152–3165.

[54] SKORIN-KAPOV D & SKORIN-KAPOV J. 1994. On tabu search for the location of interacting hub facilities. *European Journal of Operational Research*, **73**: 501–508.

[55] SKORIN-KAPOV D, SKORIN-KAPOV J & O'KELLY M. 1996. Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research*, **94**: 582–593.

[56] TOPCUOGLU H, CORUT F, ERMIS M & YILMAZ G. 2005. Solving the uncapacitated hub location problem using genetic algorithms. *Computers and Operations Research*, **32**(4): 967–984.

[57] YUAN X, ZHANG S, PIBOLEAU L & DOMENECH S. 1988. Une methode d'optimisation nonlineare an variables pour la conception de procedes. *Operations Research*, **22**: 331.