Ann Oper Res
DOI 10.1007/s10479-017-2517-0

Managing facility disruption in hub-and-spoke networks: formulations and efficient solution methods

Nader Azizi[1]

Abstract Hub disruption result in substantially higher transportation cost and customer dissatisfaction. In this study, first a mathematical model to design hub-and-spoke networks under hub failure is presented. For a fast and inexpensive recovery, the proposed model constructs networks in which every single demand point will have a backup hub to be served from in case of disruption. The problem is formulated as a mixed integer quadratic program in a way that could be linearized without significantly increasing the number of variables. To further ease the model' computational burden, indicator constraints are employed in the linearized model. The resulting formulation produced optimal solutions for small and some medium size instances. To tackle large problems, three efficient particle swarm optimisation-based metaheuristics which incorporate efficient solution representation, short-term memory and special crossover operator are proposed. We present the results for two scenarios relating to high and low probabilities of hub failures and provide managerial insight. The computational results, using problem instances with various sizes taken from CAB and TR datasets, confirm the effectiveness and efficiency of the proposed problem formulation and our new solution techniques.

Keywords Hub-and-spoke · Reliability · Hub failure · Particle swarm optimization · Indicator constraints

1 Introduction

The classical hub location problem aims to find suitable locations for hubs, allocate the existing demand to these facilities, and direct the flow between origin-destination pairs at minimum cost. In this class of locational problems it is common to assume there is a link between every hub pair, there is no direct path between non hub nodes, and there is economies

✉ Nader Azizi
n.azizi@kent.ac.uk

[1] Kent Business School, University of Kent, Canterbury CT2 7PE, UK

Published online: 17 May 2017

of scale for using the inter-hub connections (Alumur and Kara 2008). Hub location problems are categorised into two distinctive groups namely single and multiple allocation problems. In the former, all incoming and outgoing traffic to and from every node is transferred via a single hub, while in the latter each node is allowed to receive and send flow through more than one hub. The focus of this paper is on the Uncapacitated Single Allocation p-Hub Location problem where the number of hub facilities to open is predetermined. This problem is also known as the Single Allocation p-Hub Median problem.

Early research on hub-and-spoke systems focused on developing efficient mathematical formulations and solution techniques for this paradigm. O'Kelly (1986, 1987) presented the first mathematical model for the single allocation p-hub median problem. Another pioneer of the hub location research, Campbell (1994), developed a linear integer formulation for the problem. Based on the idea of multicommodity flow, Ernst and Krishnamoorthy (1996) proposed a new set of formulations for both single and multiple allocation cases. A widely used formulation in the literature for single and multiple allocation p-hub median problem is the work of Skorin-Kapov et al. (1996). The MIP models proposed by Skorin-Kapov et al. (1996) yield to tight linear relaxation.

Over the years, a number of solution techniques e.g., approximation and exact methods have been developed and successfully applied to solve instances of the hub location-allocation problem. Examples of such methods include Simulated Annealing (Ernst and Krishnamoorthy 1999), Genetic Algorithm (Kratica et al. 2007), Hybrid GA and Tabu Search (Abdinnour-Helm 1998), Lagrangian Relaxation (Elhedhli and Wu 2010; Contreras et al. 2009), Benders Decomposition (Contreras et al. 2012), Branch and Bound (Ernst and Krishnamoorthy 1998), and Branch and Cut (Yaman and Carello 2005) among others.

More recent studies focus on the extension of the classical version and aim to develop more realistic hub location-allocation problems such as models with congestion (e.g., Elhedhli and Wu 2010; Azizi et al. 2017), flow dependent economies of scale (Camargo et al. 2009; Campbell et al. 2005), and competitive hub location models (Eiselt and Marianov 2009). For an overview of hub location problem the reader may refer to Campbell and O'Kelly (2012) and Contreras (2015).

In traditional approaches to network design, supply/facility disruption is often presumed to be a rare event. These approaches are mainly concerned with the location of the facilities and the allocation of demand to these facilities in such a way to minimise the total network cost. In practice, however, it is likely that over the course of time one or even more than one facility become disrupted. The cause of disruption may vary from severe weather condition and natural disasters to labour dispute and man-made sabotage. While a set of disruption causes (e.g., earth quake) may occur less frequently, a large number of other causes are likely to strike a supply chain network. For instance, to deal with frequent supply disruption Wall-Mart uses an emergency operations centre. The centre is responsible to prepare for and mitigate the effects of man-made and/or natural disasters (Snyder et al. 2016).

Research have shown that even small disruption in supply may have devastating impacts. In 1998, strikes at two General Motors parts plants resulted in closure of 100 other parts plants and then 26 assembly plants. In the context of locational problems, facility disruption result in excessive transportation cost as customers initially served by these facilities now must be served by other operational facilities in the network (Snyder and Daskin 2005).

Azad et al. (2012) proposed a capacitated supply chain network design model under random disruptions. The formulation aims to determine the locations and types of distribution centres and allocate customers to each opened facility. Azad et al. (2012) considered partially

disrupted facilities which can operate with a fraction of their initial capacities. The authors considered a "general supply chain network" which is different from the topology of the hub-and-spoke system. In a hub-and-spoke network the flow is processed at least in one facility, is bidirectional and there are inter-hub flow interactions.

Hub location problems with reliability consideration have received very limited attention in the literature (Kim and O'Kelly 2009). Kim and O'Kelly (2009) formulated two p-hub location problems in telecommunication network with reliability consideration. The first model, p-hub maximum reliability, maximises the expected network flow. The second model, p-hub mandatory dispersion, specifies the optimal hub locations to increase network reliability and prevents excessive concentration of interacting flows in particular hub facilities. Kim and O'Kelly (2009) investigated both single and multiple allocation cases without considering backup hubs and rerouting the flows.

An et al. (2015) proposed a mixed integer nonlinear formulation for reliable hub-and-spoke problem. The authors successfully solved the formulation for problem instances up to 25 nodes using branch and bound and Lagrangian methods. An et al. (2015) compared the number of passengers served in a classical and that in a reliable network. Using a numerical example, they show that a reliable network can transport more passengers by its regular routes than a network with classical configuration. They also showed that the configuration of classical and reliable networks may not be identical. An et al. (2015) assumed (a) the flow transported in the network is symmetrical and (b) in designing alternative routes a multiple allocation is used (regardless of single allocation or multiple allocation structure used to determine regular routes). Furthermore, in the proposed model by An et al. (2015) flow transported between a pair of nodes which are allocated to the same hub facility will be rerouted through a single backup hub i.e., restricting a pair of demand points initially assigned to the same hub to be rerouted via a single backup hub. Tran et al. (2017) proposed a mixed integer nonlinear model to find the optimal location of predetermined number of hubs. They linearize the model using a specialized flow network called a probability lattice and used a tabu search algorithm to solve problem instances up to 25 nodes with symmetrical flows. Tran et al. (2017) used a level-set approach to allocate spokes to hubs in a specific order. This method of allocation though effective may prevent the model to be extended by e.g., incorporating capacity constraints and/or adapting multiple allocation scheme. Azizi et al. (2016) also proposed a mixed integer nonlinear formulation to design hub-and-spoke network taking into account the probability of hub failure and re-routing cost. They solved a number of relatively small problem instances to optimality and proposed an evolutionary algorithm to solve large instances. Similar to An et al. (2015), the authors showed that configuration of networks with and without hub failure consideration may not be the same. Azizi et al. (2016) consider a backup facility for each hub and assumed that once a facility becomes unavailable all demand points initially served by the disrupted facility is served by one of the operating facilities in the network. Reallocating the affected nodes to only one of the operating facilities in the network though effective in cases where demand points need to communicate via a single hub, may not lead to optimal solution in other cases.

In this paper, we present models that simultaneously minimises the day-to-day network operations cost and the expected transportation cost that incurs when one of the hub facilities experience short term disruption. In doing so, our proposed models seek to find the optimal locations for the hub facilities, allocate the demand to these hubs and find the least expensive backup facility for each demand point in the network. The resulting network(s) is expected to satisfy the demand with a minimum additional cost in the event of supply disruption. Our models intend to provide robust solutions (i.e., networks) that perform well even when part of the network fails. Handling instances with and without symmetrical flow, reallocating

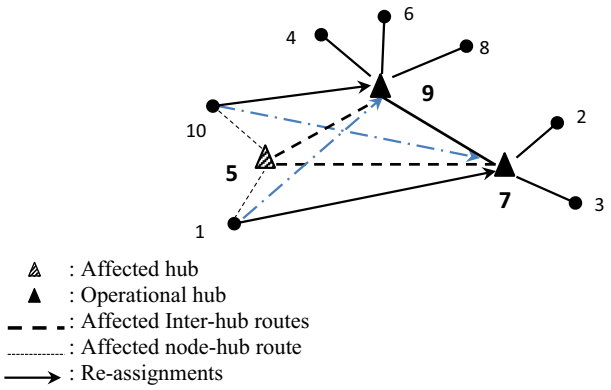


Fig. 1 Reallocation of demand in a typical network when one facility (hub 5) fails to perform normal operations

the affected demand without any restriction to all operating hubs and offering solutions under a range of hub failure probabilities are other important characteristics of the proposed formulation.

To further elaborate our approach, consider the hub-and-spoke system with 10 nodes and 3 hubs presented in Fig. 1. In such a network if any of the three hub facilities is disrupted, either one or the two other operating hubs in the network could be used as backup facility (ies) to satisfy the demand initially served by the disrupted facility. For instance, if hub 5 is disrupted then to maintain network operations, demand point 1 could be reallocated to either hub 9 or 7. Similarly, demand point 10 could also be reassigned to one of the two operating facilities as depicted in Fig. 1.

In short, the contribution of the paper include:

(i) An investigation into a reliable hub-and-spoke system incorporating heterogeneous probability of hub failure
(ii) A non-linear model purposely designed to simultaneously construct networks with and without symmetrical flow and select backup facilities for every demand possibly affected by a disruption
(iii) An improved linear model with the use of indicator constraints and
(iv) Efficient PSO-based meta-heuristics for large instances.

The remainder of this paper is organised as follow. In Sect. 2, we describe the problem and present non-linear, linear and improved linear formulations. The details of the three proposed PSO based algorithms are presented in Sect. 3 followed by computational results and discussions in Sect. 5. Concluding remarks are presented in Sect. 6.

2 Problem statement and formulation

In the single allocation p-hub median problem, each node is assigned to a particular hub and all incoming and outgoing flow are routed through that hub. The total number of nodes in the network is assumed to be N, each node is considered a potential site and the number of hubs to open is p. A path from origin spoke i to destination spoke j includes three parts: collection from spoke i to the first hub k, transfer between first hub k and the second hub m, and distribution from hub m to destination j. The cost per unit flow along this route, $i \rightarrow k \rightarrow$

$m \rightarrow j$, is calculated as $\chi \times c_{ik} + \alpha \times c_{km} + \delta \times c_{mj}$ where χ and δ are coefficients of collection and distribution respectively and α is the inter-hub discount factor. Let λ_{ij} be the amount of flow to be routed from origin i to destination j, the transportation cost from i to j routed via hubs k and m, C_{ikmj} is then calculated as $C_{ikmj} = \lambda_{ij}\left(\chi \times c_{ik} + \alpha \times c_{km} + \delta \times c_{mj}\right)$.

2.1 Notation

Our model incorporates the following decision variables: hub location and allocation variable z, the route selection x, and the backup selection variable u. These variables are defined as follows:

$z_{ik} = 1$ if node i is assigned to hub k and $=0$ otherwise;
$x_{ikmj} = 1$ if flow from i to j passes through hub k and m and $= 0$ otherwise;
$u_{ikn} = 1$ if n is the backup for node i when hub k in the path from origin i to any destination j is disrupted and $= 0$ otherwise.

We consider location-specific probability of hub failure (heterogeneous). Each hub k (or m) has a probability of failure q_k (q_m) and we assume that only one hub will be disrupted at any given time.

2.2 Model formulation

To formulate the problem, in the event of any hub disruption we recognise four different types of flows namely: (1) flow that is not affected by the disruption (2) flow that uses the disrupted facility as its first hub (3) flow that uses the disrupted facility as the second hub (4) flow that originated from or destined to a hub facility.

The cost of transporting the flow in part of the network which is not affected by disruption could be calculated by subtracting the transportation cost of those routes *affected* by hub failure from the total network transportation cost as follows (Azizi et al. 2016):

$$\sum_l \left(\underbrace{\left(\sum_i \sum_k \sum_m \sum_j C_{ikmj} x_{ikmj} z_{ll} \right)}_{\sum_k R_k z_{ll}} - \left(\underbrace{\sum_i \sum_m \sum_j C_{ilmj} x_{ilmj}}_{R_l} + \underbrace{\sum_i \sum_{k \neq l} \sum_j C_{iklj} x_{iklj}}_{Q_l} \right) \right) q_l$$

$$= \sum_l \left(\underbrace{\left(\sum_k R_k z_{ll} \right)}_{z_{ll} \sum_k R_k} - (R_l + Q_l) \right) q_l = \sum_l \left(\hat{R} z_{ll} - (R_l - Q_l) \right) q_l$$

where

$$\hat{R} = \sum_k R_k$$

Alternatively, this cost could be calculated by adding up the transportation cost in all routes for which the disrupted hub is neither their first nor their second hub. In other words, the total cost of interactions between nodes in the right hand side of the network in Fig. 1 namely nodes 2, 3, 7, 9, 8, 6 and 4 gives the transportation cost in part of the network that is not affected by disruption. This is presented as

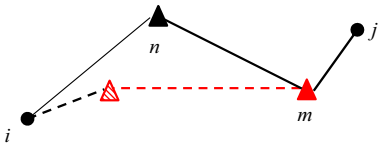


Fig. 2 Rerouting the flow via backup facility n when first hub k is disrupted

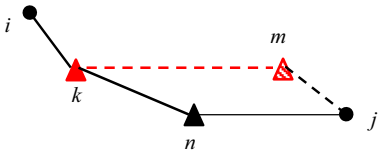


Fig. 3 Rerouting the flow via backup facility n when hub m is disrupted

$$\sum_i \sum_j \sum_l \gamma_{ijl} z_{ll} q_l \tag{1}$$

where

$$\gamma_{ijl} = \sum_{k \neq l} \sum_{m \neq l} C_{ikmj} x_{ikmj} \forall i, j, l \tag{2}$$

Index l represent the assumed disrupted facility. We conducted a computational experiment and found that formulating this cost as a function of R_l and Q_l may result in a weak LP relaxation which is likely to lead to a lengthy computational time. Therefore, in our formulation Eq. (1) is used to calculate the cost of transporting the flow in part of the network which is not affected by disruption.

In case of hub disruption, we assume the affected demand points are served by other facilities in the network. We denote Ω_{ikn} as the rerouting cost of the flow through backup facility n when hub k in the path from origin i to destination j fails. Ω_{ikn} is expressed as

$$\Omega_{ikn} = \sum_{m \neq k} \sum_j C_{inmj} x_{ikmj} \forall i, k, n; i \neq k \tag{3}$$

Figure 2 illustrates how a flow is rerouted via backup facility n when the first hub k is unavailable. The new route to transfer the flow from origin i to destination j is highlighted by solid lines.

Similarly, β_{jmn} is the cost of rerouting the flow through backup n if hub m fails (the first hub for the route initiate at node j), see Fig. 3.

$$\beta_{jmn} = \sum_{k \neq m} \sum_i C_{iknj} x_{ikmj} \forall j, m, n; j \neq m \tag{4}$$

We formulated the problem as a bi-objective optimisation problem and the two components (F1 & F2) of the objective function of t he model are presented as follows:

$$F1 = \sum_i \sum_k \sum_m \sum_j C_{ikmj} x_{ikmj} \tag{5}$$

$$F2 = \sum_i \sum_j \sum_l \gamma_{ijl} z_{ll} q_l + \sum_i \sum_k \sum_n \Omega_{ikn} u_{ikn} q_k$$

$$+\sum_{i}\sum_{k}\sum_{m}\sum_{n}\sum_{j} C_{inmj}x_{ikkj}u_{ikn}u_{jkm}q_k$$
$$+\sum_{j}\sum_{m}\sum_{n}\beta_{jmn}u_{jmn}q_m$$
$$+\sum_{i}\sum_{j}\varphi_{ij}w_{ij}\left(q_i z_{ii}+q_j z_{jj}\right) \tag{6}$$

The costs F1 and F2 refer to the transportation cost in a regular situation and the expected transportation cost resulted from hub failures respectively. Objective F2 consists of five terms. The first term computes the transportation cost of the flow that is not affected by disruption. The second term in F2 calculates the rerouting cost of the flow originated from node i through backup facilities when hub k in the path from origin i to destination j is disrupted and the path includes two hubs; the third term calculates the rerouting cost when the only hub in the path from origin i to destination j is disrupted. In a path with a single hub, origin (i) and destination (j) are initially assigned to the same node (i.e., k). When hub k is disrupted, the two demand points (i and j) would be either allocated to the same or to different backup facilities. The forth term in F2 calculates the rerouting cost of the flow through backup facilities when hub m is disrupted in the path from i to j. In this study it is assumed that when a hub is disrupted, it cannot send or receive any flow to or from other nodes in the network. Therefore, the last term in the objective function F2 (i.e., the fifth term) penalizes the loss of flow/demand in disrupted situations where the source or destination of the flow is a hub. In our study, the penalty cost of losing a unit flow, φ_{ij} , is considered twice as much as the transportation cost of a unit flow between origin i and destination j (Azizi et al. 2016).

The proposed model minimizes the weighted objective function $wF1+(1-w)F2$ where $0 \leq w \leq 1$. The first formulation of the Reliable Single Allocation p-Hub Location problem which is denoted by (RSApHL-I) for short is presented as follows

$$\textit{Minimize } wF1+(1-w)F2 \tag{7}$$

Subject to:

$$\sum_{k} z_{ik}=1 \quad \forall i \tag{8}$$
$$\sum_{k} z_{kk}=p \tag{9}$$
$$z_{ik}\leq z_{kk} \quad \forall i,k \tag{10}$$
$$\sum_{m} x_{ikmj}=z_{ik} \quad \forall i,j,k \tag{11}$$
$$\sum_{k} x_{ikmj}=z_{jm} \quad \forall i,j,m \tag{12}$$
$$\sum_{n\neq k} u_{ikn}=z_{ik} \quad \forall i,k;i\neq k \tag{13}$$
$$u_{ikn}\leq z_{nn} \quad \forall i,k,n;i\neq k \tag{14}$$
$$\gamma_{ijl}=\sum_{k\neq l}\sum_{m\neq l} C_{ikmj}x_{ikmj} \quad \forall i,j,l \tag{15}$$
$$\Omega_{ikn}=\sum_{m\neq k}\sum_{j} C_{inmj}x_{ikmj} \quad \forall i,k,n;i\neq k \tag{16}$$

$$\beta_{jmn} = \sum_{k \neq m} \sum_{i} C_{iknj} x_{ikmj} \quad \forall j, m, n;\ j \neq m \tag{17}$$

$$x_{ikmj}, z_{ik}, u_{ikn} \in \{0, 1\} \quad \forall i, j, k, m, n \tag{18}$$

$$\Omega_{ikn}, \beta_{jmn}, \gamma_{ijl} \geq 0 \quad \forall i, k, m, n, l \tag{19}$$

Constraints (8)–(12) are classical constraints for the single allocation p-hub location problem (Skorin-Kapov et al. 1996). Constraint (8) ensures every node is assigned to only one hub whereas Constraint (9) limits the number of hubs to be opened to "p". Constraint (10) is to ensure node i is assigned to an open hub. Constraints (11) and (12) guarantee that all the traffic between an origin-destination pair has been routed via the hub sub-network. Constraint (13) guarantees each node i has only one backup (n) and it differs from the hub initially assigned to (i.e., k). Constraint (14) ensures the candidate backup n for the affected node i is an open "hub" facility. Equality constraint (15) calculate the transportation cost of the flow that is not routed via hub l. Constraints (16) compute the rerouting cost of flow through backup facility n that is originally transported via the first hub k. Constraints (17) calculate the rerouting cost through backup n that is originally transported through hub m. Constrains (18) and (19) are the standard integrality constraints.

The proposed formulation for RSApHL-I is a Mixed Integer Quadratic Programing (MIQP) and includes $n^2 + 4n^3 + n^4$ variables. In the next section we will show that because of our novel modelling approach linearizing the model using conventional techniques will not significantly increase the number of variables. Our approach is similar to the technique proposed by Chaovalitwongse et al. (2004) to linearize multi-quadratic 0–1 programming problems.

2.3 Linearization

The non-linear terms in the above objective function resulted from the multiplication of binary and non-binary variables of $x, u, z,$ and $\gamma_{ijl}, \Omega_{ikn}, \beta_{jmn}$. To linearize the model, these terms are substituted by three continuous and binary variables y_{ijl}, Γ_{ikn}, ψ_{ikmnj} and ξ_{jmn} as follows.

$$y_{ijl} = \gamma_{ijl} z_{ll} \tag{20}$$

$$\Gamma_{ikn} = \Omega_{ikn} u_{ikn} \quad \forall i, k, n \tag{21}$$

$$\psi_{ikmnj} = x_{ikkj} u_{ikn} u_{jkm} \quad \forall i, k, m, n, j \tag{22}$$

$$\xi_{jmn} = \beta_{jmn} u_{jmn} \quad \forall j, m, n \tag{23}$$

To enforce the above relationships and using the Big-M idea the following sets of constraints are added to the formulations. Here, we refer to Big-Ms as M_1, M_2 and M_3.

$$y_{ijl} \leq M_1 z_{ll} \quad \forall i, j, l \tag{24}$$

$$y_{ijl} \leq \gamma_{ijl} \quad \forall i, j, l \tag{25}$$

$$y_{ijl} \geq \gamma_{ijl} - M_1 (1 - z_{ll}) \quad \forall i, j, l \tag{26}$$

$$\Gamma_{ikn} \leq M_2 u_{ikn} \quad \forall i, k, n \tag{27}$$

$$\Gamma_{ikn} \leq \Omega_{ikn} \quad \forall i, k, n \tag{28}$$

$$\Gamma_{ikn} \geq \Omega_{ikn} - M_2 (1 - u_{ikn}) \quad \forall i, k, n \tag{29}$$

$$\psi_{ikmnj} \leq u_{ikn} \forall i, k, m, n, j \tag{30}$$

$$\psi_{ikmnj} \leq u_{jkm} \quad \forall i, k, m, n, j \tag{31}$$

$$\psi_{ikmnj} \leq x_{ikkj} \quad \forall i,k,m,n,j \tag{32}$$

$$\psi_{ikmnj} \geq x_{ikkj} + u_{ikn} + u_{jkm} - 2 \quad \forall i,k,m,n,j \tag{33}$$

$$\xi_{jmn} \leq M_3 u_{jmn} \quad \forall j,m,n \tag{34}$$

$$\xi_{jmn} \leq \beta_{jmn} \quad \forall j,m,n \tag{35}$$

$$\xi_{jmn} \geq \beta_{jmn} - M_3\left(1 - u_{jmn}\right) \quad \forall j,m,n \tag{36}$$

$$0 \leq y_{ijl} \leq M_1 \quad \forall i,j,l \tag{37}$$

$$0 \leq \psi_{ikmnj} \leq 1 \quad \forall i,k,m,n,j \tag{38}$$

$$0 \leq \Gamma_{ikn} \leq M_2 \quad \forall i,k,n \tag{39}$$

$$0 \leq \xi_{jmn} \leq M_3 \quad \forall j,m,n \tag{40}$$

The resulting linear model for Reliable Single Allocation p-Hub Location problem which we call (RSApHL-II) is presented as follow.

$$Minimize\, wF1 + (1-w)\, F2 \tag{41}$$

where

$$F1 = \sum_i \sum_k \sum_m \sum_j C_{ikmj} x_{ikmj} \tag{42}$$

$$\begin{aligned} F2 = & \sum_i \sum_j \sum_l y_{ijl} q_l \\ & + \sum_i \sum_k \sum_n \Gamma_{ikn} q_k \\ & + \sum_i \sum_k \sum_m \sum_n \sum_j C_{inmj} \psi_{ikmnj} q_k \\ & + \sum_j \sum_m \sum_n \xi_{jmn} q_m \\ & + \sum_i \sum_j \varphi_{ij} w_{ij} \left(q_i z_{ii} + q_j z_{jj}\right) \end{aligned} \tag{43}$$

Subject to:

(8) − (17)

(24) − (40)

$$x_{ikmj}, z_{ik}, u_{ikn} \quad \in \{0,1\} \quad \forall i,j,k,n,m \tag{44}$$

$$\Omega_{ikn}, \beta_{jmn}, y_{ijl} \geq 0 \quad \forall i,k,m,n \tag{45}$$

The number of variables in the above MIP model increases to $n^2+7n^3+n^4+n^5$. A summary of the number of variables and constraints before and after the linearization is given in Table 1.

Table 1 The Number of variables and constraints after linearization

Problem	Number of variables	Number of constraints
RSApHL-I (MIQP)	$n^2+4n^3+n^4$	$1+n+2n^2+7n^3$
RSApHL-II (MIP)	$n^2+7n^3+n^4+n^5$	$1+n+2n^2+16n^3+4n^5$

2.4 Linear model with indicator constraints

In MILP formulations, it is common to model constraints that either hold or are relaxed depending on the value of a binary variable. These constraints are often modelled using the so called *Big-M formulations* to enforce the relationship between a new variable and its corresponding nonlinear term. These Big-M formulations though useful are often blamed for unstable behaviour in MILP model. *Indicator Constraints* have the advantage of preventing the problems associated with Big-M formulations. Similar to Big-M formulations, indicator constraints use a binary variable to turn on or turn off the enforcement of a constraint but without using a big M. Further information about the indicator constraints could be found in the paper by Bonami et al. (2015).

In our linear formulation i.e., RSApHL-II we replaced all the constraints formulated using *Big Ms* by indicator constraints and set $0 \leq x_{ikmj} \leq 1$. For instance, using indicator constraints the following

$$\begin{aligned} y_{ijl} &\leq M_1 z_{ll} \\ y_{ijl} &\leq \gamma_{ijl} \\ y_{ijl} &\geq \gamma_{ijl} - M_1 (1 - z_{ll}) \end{aligned}$$

can be written logically as

$$z_{ll} = 0 \rightarrow y_{ijl} = 0 \tag{46}$$

$$z_{ll} = 1 \rightarrow y_{ijl} = \gamma_{ijl} \tag{47}$$

The new formulation with indicator constraints which we refer to as (RSApHL-III) is given below for the sake of simplicity and convenience.
(41)–(43)
Subject to:
(8)– (17)
(24)– (40)
(44)– (47)

$$u_{ikn} = 0 \rightarrow \Gamma_{ikn} = 0 \tag{48}$$

$$u_{ikn} = 1 \rightarrow \Gamma_{ikn} = \Omega_{ikn} \tag{49}$$

$$u_{jmn} = 0 \rightarrow \xi_{jmn} = 0 \tag{50}$$

$$u_{jmn} = 1 \rightarrow \xi_{jmn} = \beta_{jmn} \tag{51}$$

2.5 Preliminary study

To evaluate and compare the performance of the proposed (linear) formulations with and without indicator constraints, we conducted a limited computational experiment as described in the following. A relatively small problem instance with 10 nodes and 3 hubs is selected from the U.S. Civil Aeronautics Board which is known as CAB dataset (O'Kelly 1987). We generated three instances of the original problem by setting the objective function weight w to 0.3, 0.5 and 0.7. The discount parameter α is set to 0.2; the coefficients of collection χ and distribution cost δ are set to one per unit for all three instances. The problems are solved using CPLEX 12.6 with default options values. The algorithm is run for 5000 s on Intel Core i5 PC with 3.2 GHz processor with 4 GB of RAM. The computational results are presented in Table 2.

Table 2 CPLEX results for three instances of RSApHL-II and RSApHL-III

n[a]	p[b]	α[c]	w[d]	RSApHL-II			% GAP	RSApHL-III		
				Lower bound	Best solution	Time (s)		Lower bound	Best solution	Time (s)
10 (CAB)	3	0.2	0.3	250.05	408.48	5000	39	367.24	367.24[e]	824
			0.5	409.98	409.89[e]	1405	0	409.98	409.98[e]	356
			0.7	452.54	452.54[e]	936	0	452.54	452.54[e]	189

[a] Number of nodes, [b] number of hubs, [c] discount factor, [d] objective weight, [e] optimal solution

The results pertaining to RSApHL-II model show that CPLEX is unable to produce the optimal solution for the problem instance with $w = 0.3$ in 5000 s but manages to locate the optimal solution for the other two instances with $w = 0.5$ and 0.7.

For the RSApHL-III model, results presented in Table 2 clearly show that the same problem instances could be solved more effectively and with significantly less computational efforts when all Big-M formulations are replaced with indicator constraints. All optimal solutions are now discovered while requiring between 20 and 25% of the (computational) times reported for instances of RSApHL-II. This is an interesting result which is achieved by a combination of an effective formulation and the use of the recently made available modelling tool i.e., indicator constraints.

This improved formulation will be used to provide when possible lower and upper bounds for benchmarking purposes. As will be shown in our computational results, the above improved model formulation could only be used to solve relatively small to medium size problem instances to optimality.

Due to the limitation in solving the above model formulation to optimality for realistically sized problem instances with CPLEX, one way forward is to design an efficient metaheuristic. In this study, we developed three metaheuristics based on the well-known evolutionary algorithm of Particle Swarm Optimisation (PSO). These algorithms are discussed in the following section.

3 Particle swarm optimization

Particle swarm optimization (PSO) is one of the most efficient nature-inspired optimization algorithms. It was proposed by Kennedy and Eberhart (1995) about two decades ago. The algorithm was developed based on the motion of a flock of birds searching for foods and aimed to mainly optimise continuous non-linear optimisation problems.

In PSO based algorithms, at the beginning of the evolutionary process a set of particles which is called a *swarm* is generated randomly. Each member of a swarm is called a *particle*. During the search process, particles are flown through a hyperspace. A particle's status on the space is characterized by two elements namely *position* and *velocity*. Particles may change their position in the search space similar to flying birds searching for food in the sky. While searching the hyperspace, a particle adjusts its (new) velocity according to its own best experience, the best experience of all particles in the swarm and its previous velocity. The particle's new velocity and current position are then used to determine its new position.

A large portion of the published research works on PSO is dedicated to continuous optimization problems whereas there is a lack of research dealing with its discrete counterpart

namely combinatorial optimization problems (Liu et al. 2008). To the best of our knowledge, our proposed algorithms is the first PSO-based optimisation technique to be developed for solving this interesting though challenging class of hub location problem.

In this paper two variations of a basic PSO and a hybrid PSO algorithm that incorporate crossover and memory are proposed. The two versions which we refer to as PSO-v1 and PSO-v2, differ in terms of solutions representation and the mechanism by which a continuous particle position is transformed into one or more discrete solution(s). The hybrid PSO algorithm on the other hand, is an extension of PSO-v2 which incorporates a short-term memory and a crossover operator, both of which are in our knowledge, new ingredients that are successfully embedded for the first time into the PSO search methodology.

3.1 PSO-v1 algorithm

In PSO-v1 algorithm, each particle is represented by a $(1 \times n)$ array in which n represent the number of nodes. We define the position of a particle by

$$p_{kl} = \left[a_{kl}^1, a_{kl}^2, a_{kl}^3, \ldots, a_{kl}^n\right]$$

With $a_{kl}^{(i)}$ being its ith position value. We represent a group of m particles (i.e., a population of particles) at each iteration l by

$$S^l = [p_{1l}, p_{2l}, p_{3l}, \ldots, p_{ml}]$$

Particle position PSO-v1 algorithm starts by generating *random* continuous position values for the m particles in the population in the range

$$[1,\ (p + 1 - \varepsilon)]$$

where p is the number of hubs and $0 < \varepsilon < 1$; in this study, we assume $\varepsilon = 0.01$.

Figure 4 illustrates a typical particle position for a problem with 10 nodes and 3 hubs which is populated with continuous random numbers in the range of [1, 3.99].

Particle velocity In PSO-based algorithms, particles fly to new areas in the search space using a *velocity* function. In this study, the initial velocities are generated randomly according to the following expression:

$$v_{k0} = v_{min} + (v_{max} - v_{min}) \times U\,\{0, 1\} \tag{52}$$

v_{k0} is the initial velocity of particle p_{k0} with v_{min} and v_{max} being the pre-defined range of velocity values. We assume v_{min} is zero and we define v_{max} as a function of the number of nodes (n) and hubs (p) in the problem

$$v_{max} = (p \times n) \times \theta \tag{53}$$

where θ is a small value between $0 < \theta < 1$; here we assume $\theta = 0.1$

3.1	2.95	1.3	3.2	2.7	3.6	1.6	3.3	3.4	2.1

Fig. 4 Example of a particle presentation in PSO-v1 for a problem with 10 nodes and 3 hubs

3	2	**1**	3	2	3	**1**	3	3	2

Fig. 5 Particle position array in Fig. 5 after disregarding decimal points

Updating particle velocity and position At iteration l the kth particle velocity is updated using the following formula

$$v_{k,l+1} = \omega v_{kl} + C_1 r_1 \left(Pbest_{kl} - p_{kl}\right) + C_2 r_2 \left(Pbest_{gbest,l} - p_{kl}\right); k = 1, \ldots, n \quad (54)$$

where C_1 and C_2 are acceleration coefficients; ω the inertia factor; and r_1 and r_2 are two independent random numbers uniformly distributed in the range [0 , 1].

Once all the n particles new velocities are determined, the corresponding particles' positions are then updated by adding the new particles velocities to their current positions as follow

$$p_{k,l+1} = Max\left[\left(p_{kl} + v_{k,l+1}\right), p_{max}\right] \quad (55)$$

Also it is common practice in PSO algorithms to control the search by limiting the magnitude of particle's elements. In this study we set

$$p_{max} = p + 1 - \varepsilon \quad (56)$$

Transforming a continuous particle position into one or more discrete solutions Due to the continuous nature of the particle position in PSO, standard encoding schemes cannot be directly adopted for discrete location problems such as hub and spoke systems. One procedure to transform a continuous particle position into a discrete solution in PSO-v1 is described via the following example. Consider the particle' position presented in Fig. 5. As mentioned earlier, the number of columns/elements in a particle position array corresponds to the number of nodes in the problem (in this example the number of elements are 10). The integer part of the largest element of the array refers to the number of hubs which in the above example is 3.

The one dimensional array with continuous values presented in Fig. 4 could be encoded as an array with integer values by taking its integer part $[a_n]$ where a_n represent the value of each element in the position array. The corresponding particle position array with integer values for the example in Fig. 4 is presented in Fig. 5.

All elements in the particle position that have the same value constitute a cluster. For example, the following three clusters $\{3, 7\}$, $\{2, 5, 10\}$ and $\{1, 4, 6, 8, 9\}$ are presented in Fig. 6a.

As each member of a cluster (i.e., a node) is a potential site for a hub facility, a discrete feasible solution can be generated by connecting the selected hubs. For example, a possible configuration is given in Fig. 6b where nodes 3, 2, and 4 corresponding to clusters 1, 2 and 3 are chosen. Considering the combination of potential hub facilities in each cluster node, a quite large number of feasible solutions could be constructed from one single particle position. In the example presented in Fig. 6a, a total of 30 ($3 \times 2 \times 5$) solutions could be generated. In general, if n_i refers to the number of nodes in cluster i, the total number of configurations become $\prod_{i=1}^{p} n_i$.

Clearly, as the number of hubs and/nodes increases the number of solutions/possibilities could increase significantly. Examining all possible solutions produced by changing hub facilities in each cluster could be, particularly, for relatively large instances computationally expensive. Therefore, it is more practical to either randomly select one solution among all

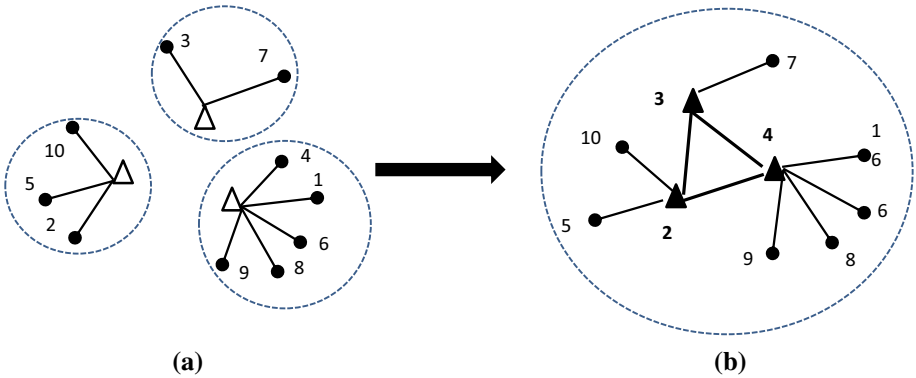


Fig. 6 **a** Three cluster nodes associated with particle array in Fig. 6, **b** a typical solution generated from the three cluster nodes

Start
i=1, initialise K_0 *(say* K_0=0.5)
Do while $i < SwarmSize$
create a *one-dimensional* particle position array with continuous values $\left[a_{kl}^{i}\right]$
Take the integer part of the particle elements (i.e., $Int(a_{kl}^{i})$)
Identify the *p* cluster nodes in the particle position array
Generate *K* solutions from $\prod_{i=1}^{p} n_i$ potential solutions; $K = K_0 \prod_{i=1}^{p} n_i$
Evaluate the cost for each configuration and select the solution with the least cost.
Apply a perturbation to increase diversification
$i \leftarrow i + 1$
Loop
End

Fig. 7 Algorithmic steps to convert a continuous particle position into a discrete solution (PSO-v1)

possible solutions or generate a fraction of potential solutions and select the best one with the lowest total cost.

We set up a limited number of experiments using instances with 10, 15, 20, and 25 nodes and concluded that examining a fraction of the potential solutions (e.g., k_0) and selecting the best one among the subgroup could provide good overall best results.

In our proposed PSO-v1 algorithm, once a discrete solution is found, a small perturbation (e.g., reassigning a node to another hub) is also applied to improve the diversification in particles population. The main steps to transfer a continuous particle position into a discrete solution are given in Fig. 7 with *Swarmsize* representing the population size.

Backup selection PSO-v1 encoding scheme transfers a continuous particle position to a discrete solution to a single allocation median hub location problem. Although the selection process of backup facilities could also be a part of the solution representation, to retain the simplicity of our solution representation we opted to treat this decision separately. Backup facilities for the existing demand points could be chosen (1) randomly (2) by examining all possibilities or (3) based on proximity to a potential backup facility. Examining all combinations (i.e., complete enumeration) will provide the best solution as it guarantees the selection of the most economically suitable backup but it is computationally expensive even for small problem instances. We therefore compared the two other options using a number of test

problems and concluded that in large instances selecting nearby backups to demand nodes outperforms the random backup selection strategy. For the small to medium size problem instances, random backup selection may result in better quality solutions.

4 PSO-v2 algorithm

In this variant PSO-v2, a continuous particle position is directly encoded/mapped to only one discrete solution. The main difference(s) between this version and the earlier one are the solution representations and the mechanisms through which a continuous particle position is encoded into a solution.

Particle position and representation Here, a particle is represented by a matrix of size $2 \times n$ where n stands for the number of nodes. Entities in the first row of the matrix represent hub facilities. Therefore, in a problem with p hubs, we deal with only the first p elements in the first row of the matrix and the remainders are used to rank the entire elements later. For instance, see Fig. 8 for a problem with 10 nodes and 3 hubs.

Entities in the second row of the matrix denote the allocation of each node to one of the hubs specified in the first row.

To generate an initial particle position, the first row of the matrix is populated with continuous numbers generated using the following equation

$$p_{k0} = p_{min} + (p_{max} - p_{min}) \times U\{0, 1\} \tag{57}$$

we assume $p_{min} = 0$ and $p_{max} = p \times n$.

Similar to PSO-v1 position array, the second row of the matrix is filled with randomly generated continuous numbers in the range of $[1, p + 1 - \varepsilon]$ where $0 < \varepsilon < 1$.

Particle velocity Similar to that in PSO-v1, the initial values of particles' velocities, v_{k0} , are generated using Eq. (52) with v_{max} and v_{min} defined as in PSO-V1 (Eq. 53). Particles velocities and positions are updated in each iteration using Eqs. (54) and (44) respectively.

Transforming a continuous particle position into a single discrete solution To transfer a continuous particle position into a discrete solution, first all entities in the first row of the particle position matrix are ranked. Then p elements of the first row are considered as potential hub locations. For instance, in the particle position presented in Fig. 8 the smallest number is the fifth element "0.7" and is ranked "1"; the third element "1.1" is ranked second (i.e., "2"); and the first element "1.6" is ranked third (i.e., "3") etc. After ranking the numbers in the first row of the particle position matrix, the three potential hub facilities (p assume to be 3) are located as node 3 (first hub), 5(second hub) and 2(third hub).

To determine the allocation of non-hub nodes to the hubs in the first row, we consider the integer parts of the elements in the second row in Fig. 8.

1.6	**2.5**	**1.1**	3.6	0.7	5.6	2.1	7.1	3.4	4.2
3.1	2.95	1.3	3.2	2.7	3.6	1.6	3.3	3.4	2.1

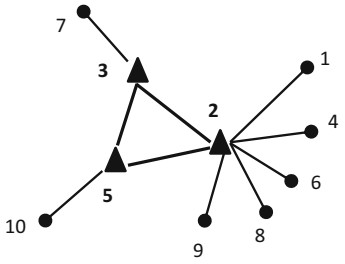
Fig. 8 Example of particle representation in PSO-v2: a problem with 10 nodes and 3 hubs

3	**5**	**2**	7	1	9	4	10	6	8
3	**2**	1	3	2	3	1	3	3	2

Fig. 9 Transferred continuous particle position in Fig. 8 to a discrete solution

Fig. 10 Repaired particle position in its corresponding network

3	**5**	**2**	7	1	9	4	10	6	8
3	**3**	1	3	2	3	1	3	3	2



Start
i=1
Do while *i* < *SwarmSize*
Take a *two-dimensional* particle position matrix with continuous values
Rank all elements in the first row
Consider the first *p* elements in the first row as potential hub locations
Take the integer parts of the numbers in the second row
Evaluate the solution
If infeasible, repair the solution

$i \leftarrow i + 1$
Loop
End

Fig. 11 Steps to convert a continuous particle position into a discrete solution: PSO-v2

For instance, in Fig. 9 the third and the seventh elements of the second row are identical and have the value of "1" which means nodes 3 and 7 are allocated to the first hub which is hub "3".

Repair algorithm During the encoding process of PSO-v2 occasionally an infeasible solution may be generated as a given hub may not be necessary allocated to itself. In such situation, a simple repair algorithm to amend the solution and change it into a feasible one is used. For instance, in Fig. 9, the second node (node "2") has been assigned to the second hub (hub "5") instead of being allocated to itself (i.e., node "2") which is the third hub in the first row. To repair this solution, the value of the second element (of the second row) is simply replaced by value "3". The resulting repaired solution is given in Fig. 10.

The transfer of a continuous particle position into a discrete solution is presented in Fig. 11.

4.1 Hybrid PSO algorithm

Our preliminary results for relatively small problem instances show that both variants (i.e., PSO-v1 and PSO-v2) perform well in finding good quality solutions to the RSApHL-III.

Nevertheless in the majority of cases the same solutions are obtained by PSO-v2 requiring relatively shorter computational times.

Research have shown that PSO-based optimization algorithms are highly randomised and, in general, are susceptible to early convergence (i.e., could be trapped into a local optima too early in the search). To overcome this drawback and hence improve the performance of the PSO-2 algorithm, two new features are imbedded into the algorithm to make our hybrid PSO which we refer to H-PSO for short. (a) The first feature is a *memory* which is setup by collecting a number of *global best* particle positions at the end of each iteration. The size of the memory is assumed to be the same as that of the PSO swarm size e.g., 20. (b) The second feature is a *crossover* operator aiming to improve the quality of the solutions in the PSO population as well as diversifying the search. Once a memory is completed, solutions stored in the memory are combined using a special type of crossover operator. To perform the crossover operation, two parent particles are selected each time randomly to produce an offspring particle. The fitness of the newly generated solution is then compared with those in the PSO population; the offspring replaces the first inferior particle found in the population. The offspring particle is ignored if it cannot replace any of the particles in the population. Upon completion of the new population the memory is reset before the next iteration of the algorithm begins. The crossover operator is described next.

The proposed crossover operator The solution representation used in H-PSO is the same as that in PSO-v2 (2 dimensional matrix). Classical crossover operators are often designed for one-dimensional arrays and could not be directly applied here. Therefore, we propose a special type of crossover that combines the classical "*single point*" with what we name a "*constructive*" crossover operator.

To produce a new offspring, a template 2-dimentional matrix for the offspring solution is first constructed. Next, two parents are marked randomly from the current population. Then, one of the two parents is selected randomly and the first element on the "first row" of its matrix is transferred to the same place on the offspring particle template matrix. The second element of the offspring template is taken from the other parent. These steps are repeated until the first row of the offspring array is populated with elements transferred from the two parents.

Upon completion of the first row of the offspring particle matrix, a classical single-point crossover is applied to the second row of the parents' particles to populate the second row of the offspring particle matrix. In a classical single-point crossover operation, one point is selected on the second row of the parent particle matrix. One of the parents is then selected randomly and all elements up to that point on its array are transferred into the offspring matrix. The rest of the locations on the offspring matrix (i.e., second row) are filled by the elements beyond the crossover point on the other parent chromosome. The steps of the proposed crossover operator are presented in Fig. 12.

5 Computational results and analysis

We tested our proposed PSO-based heuristics on 126 benchmark problems and solved our formulation for 54 of these instances using CPLEX. These instances are derived from U.S. Civil Aeronautics Board (CAB) (O'Kelly 1987) and Turkish Postal System (TR) datasets (Çetiner 2003). The problem instances are generated by setting the number of nodes N to 10, 15, 20, 25 and 55; the number of open hubs "p" to 3 and 5; the discount factor α to 0.2, 0.4

Single point-constructive crossover operator
i=1
While $i < SwarmSize+1$ **Do**
create a template matrix for an offspring particle
select two particles from the current swarm
j=1
select one of the two particles
while j<*p*+1 **Do**
Transfer item *j* from the first row in the particle matrix to the same place in the offspring particle matrix
$j \leftarrow j + 1$
select the other particle
End while
Apply the classical single-point crossover operation to the second rows of the two particles and complete the offspring matrix
$i \leftarrow i + 1$
End while

Fig. 12 The proposed crossover operator for H-PSO

Table 3 The computational time for all test problems

Nodes	10		15		20		25		55	
Hubs	3	5	3	5	3	5	3	5	3	5
CPU (s)	10	10	20	20	20	30	40	60	90	120

and 0.8; and the objective weight w to 0.3, 0.5, and 0.7. The coefficients of collection χ and distribution cost δ are set to 1 per unit for all test problems ($\chi = \delta = 1$). The PSO parameters are as follows. The swarm size is set to 25 for test problems with 10 nodes and 20 for the rest of instances. C1 and C2 are set to 2 and the inertia factor, ω, to 1 for all test problems. The computational times are presented in Table 3. Each test problem is run 10 times and the best results are reported. The MIP version of RSApHL-III model is coded in AIMMS and solved using CPLEX 12.6 with CPLEX options set to their default values. All algorithms are run on an Intel Core i5 PC with 3.2 GHz processor with 4 GB of RAM.

In this section we analyse two scenarios reflecting real life applications. In the first one we consider networks that are quite highly susceptible to hub disruptions. Whereas in the second, we attempt to address other networks in which hub disruption is less likely to occur. The former scenario includes a network in which hubs are assumed to be, for instance, distribution centres. In these types of networks the probabilities of hub failures are expected to be relatively high. In the second scenario, such as hub-and-spoke networks in airline industry, the chance of hub failure is believed to be relatively low though extremely costly if that happens.

For the case with high probability of hub failure, the probabilities, q_i, for all nodes in the network $i = 1 \ldots n$ are generated randomly from U [0.1, 0.3]. For the other case, we use the low probabilities of hub failures as given in the paper by An et al. (2015).

5.1 Scenario I: the case of high hub failure probability

5.1.1 CAB dataset

In CAB dataset the flow is transported equally in both directions (*i.e.,* symmetrical flow). In such networks, the size of the problem can be reduced by calculating the transportation cost

Table 4 A summary of the computational results and comparisons with CPLEX-CAB dataset

	PSO-v1		PSO-V2		H-PSO		CPLEX			Gap (%)
	ATNC[a]	Time[b]	ATNC	Time	ATNC	Time	ALB[c]	AUB[d]	Time	
10 Nodes	466.48	3.00	465.20	0.35	465.05	0.34	464.99	464.99	372.5	0.0
15 Nodes	840.92	12.40	804.94	8.60	801.78	7.50	533.34	850.15	1809.24	36
20 Nodes	759.50	12.51	726.01	13.16	708.83	12.85	–	–	–	–
25 Noes	878.21	20.07	833.90	29.85	815.58	21.52	–	–	–	–

[a] Average total network cost, [b] solution time, [c] average lower bound/LP bound, [d] average upper bound/best solution

in just one direction and then doubling the cost to present the total cost of transporting the flow. This setting has been adopted by a number of studies and, to some extent, reduces the computational time required to solve problems with symmetrical flow.

We begin our analysis by evaluating the computational performance of the proposed PSO algorithms and the CPLEX using 18 small benchmark problems with 10 nodes and 3 and 5 hubs. A summary of the (average) results is presented in the first row of Table 4.

CPLEX solved all 18 benchmark problems in a reasonable computational times with 17 within 1–11 min and the other requiring just 18 min with a large portion of this time being spent in closing the last 5% gap. The average computational time to solve all 18 problems is 372.5 s. Nevertheless these are still significantly high when compared against the time required by any of the PSO-based algorithms.

The results in Table 4 further show that the three PSO-based algorithms performed well though H-PSO found more optimal solutions than the other two algorithms (i.e., PSO-v1 and PSO-v2). For test problems with 10 nodes, the optimal locations of hubs and allocations of non-hubs to the selected hub facilities as well as backup facilities for each node are presented in Table 5. For example, in this table the optimal value of the objective function for the problem with 3 hubs, discount factor of 0.2, and the objective weight of 0.3 is 367.24; the locations of the hubs are 4 (Chicago), 2 (Baltimore) and 7 (Dallas-FW); the backup facility for nodes 1, 5, 6, 9 is 2 (Baltimore) and for nodes 3, 8 and 10 is 4 (Chicago) (Fig. 13).

For problem instances with 15 nodes and 3 and 5 hubs, CPLEX provides lower and upper bounds for 15 instances and finds only lower bounds for the remaining 3. A comparison of the average costs and the average computational times in Table 4 indicate that both H-PSO and PSO-v2 perform better than PSO-v1. Similar to the results for test problems with 10 nodes, no significant difference between the results provided by H-PSO and PSO-v2 is observed.

In test problems with 15 nodes, our proposed PSO based algorithms improved the upper bounds for the 18 instances with H-PSO algorithm achieving improvement up to 20%.

CPLEX did not provide any solution for larger instances with 20 and 25 nodes due to the lack of sufficient memory. A summary of the computational results for these 36 large problems are also presented in Table 4. Here, the three PSO algorithms could be clearly ranked with H-PSO on top in terms of both solutions quality and the computational times followed by PSO-v2 and PSO-v1.

Further observations and analysis Incorporating the probability of hub failure in decision making process of the hub location and allocation problem is a proactive strategy aimed at reducing the excessive financial and related non-financial costs when one or more facilities become unavailable. However, managers are less likely to use the resulting network with

Table 5 Detailed results for problems with 10 nodes

n[a]	p[b]	α[c]	w[d]	Hub locations	Backup hubs	TNC[e]	Time (s)	Gap (%)
10 (CAB)	3	0.2	0.3	4224447747	2040220424	367.24	183.8	0.0
			0.5	4224447747	2040220424	409.89	75.2	0.0
			0.7	4224447747	2040220424	452.54	57.5	0.0
		0.4	0.3	4224447747	2040220424	401.54	256.8	0.0
			0.5	4224447747	2040220424	457.13	138.8	0.0
			0.7	4224447747	2040220424	512.70	70.6	0.0
		0.8	0.3	4224447447	2040220724	469.07	286.4	0.0
			0.5	4224447447	2040220724	550.10	233.4	0.0
			0.7	4994497497	9440940704	620.35	118.2	0.0
	5	0.2	0.3	1224447847	0040120021	375.22	556.2	0.0
			0.5	1224447847	0040120021	368.98	571.6	0.0
			0.7	1224997797	0090440401	362.31	560.3	0.0
		0.4	0.3	1224447847	0040220021	432.53	638.1	0.0
			0.5	1224447847	0040220021	437.50	536.9	0.0
			0.7	1224447847	0040120021	442.46	585.5	0.0
		0.8	0.3	1224447847	0040220021	547.03	1063.4	0.0
			0.5	1224447847	0040120021	574.43	665.3	0.0
			0.7	1994997897	0140440001	588.85	107.3	0.0
		Average				464.99	372.5	0.0

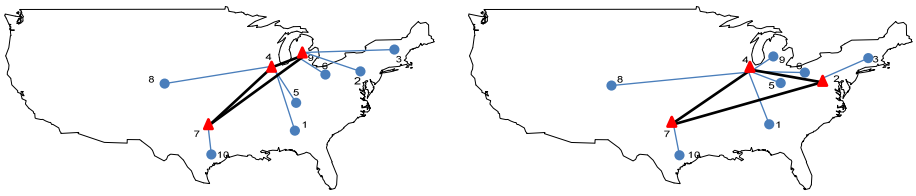
[a] Number of nodes, [b] number of hubs, [c] discount factor, [d] objective weight, [e] total network cost

1 Atlanta	6 Cleveland	11 Kansas City	16 New Orleans	21 St. Louis
2 Baltimore	7 Dallas-Fw	12 Los Angeles	17 New York	22 San Francisco
3 Boston	8 Denver	13 Memphis	18 Philadelphia	23 Seattle
4 Chicago	9 Detroit	14 Miami	19 Phoenix	24 Tampa
5 Cincinnati	10 Houston	15 Minneapolis	20 Pittsburgh	25 Washington

Fig. 13 Name of the cities in CAB dataset

reliability consideration if a significant increase in the regular transportation cost incurs. To investigate this we consider a problem with 10 nodes and 3 hubs and empirically analyse the effect of hub failure consideration on the regular network transportation cost. The benchmark problem is taken from the CAB dataset and generated by setting the objective function weight w and the discount parameter of α to 0.3 and 0.8 respectively. The optimal network configuration for this problem without hub failure consideration is depicted in Fig. 14a with its corresponding transportation cost being 716.98 units. The resulting network when the possibility of hub failure (between 10 and 30%) is taken into consideration is depicted in Fig. 14b.

The network presented in Fig. 14b is based on our approach and realistically reflects the real world. The (unweighted) regular transportation cost of the network in Fig. 14b is 752.65 unit which is 4.7% higher than that of the network provided by the classical model. This finding suggests that using our proposed model that takes to account facility unavailability may not result in a significant increase in regular transportation cost while providing a more robust configuration in case of hub failure. Furthermore, as illustrated in Fig. 14 the topologies of the two networks with and without hub failure consideration are different. This observation



(a) **(b)**

Fig. 14 **a** Optimal network to classical single allocation hub location problem, **b** optimal network to proposed RSApHLP-III

Table 6 Computational results and comparisons with CPLEX-TR dataset

	PSO-v1		PSO-V2		H-PSO		CPLEX			Gap (%)
	ATNC[a]	Time[b]	ATNC	Time	ATNC	Time	ALB[c]	AUB[d]	Time	
10 Nodes	404.19	3.80	401.79	0.20	401.79	0.30	401.78	401.78	280.12	0.0
25 Nodes	645.87	22.29	592.33	25.81	563.79	18.33	–	–	–	–
55 Nodes	705.50	38.51	672.68	65.96	662.32	54.54	–	–	–	–

[a] Average total network cost, [b] solution time, [c] average lower bounds/LP bound, [d] average upper bounds/best solution

suggests that solving a classical hub location problem to optimality and then deciding on the backup facilities is a simple but an ill-informed strategy that will lead to an inferior solution. This is an important strategic decision that ought not to be taken lightly.

5.1.2 TR dataset

The proposed formulation and algorithms have been also tested on 54 benchmark problems derived from TR dataset with non-symmetrical flow ranging from 10 to 55 nodes. The average costs and computational times over 18 problem instances with 10 nodes and 3 and 5 hubs are presented in the first row of Tables 6. Similar to the CAB dataset, CPLEX solved all benchmark problems with 10 nodes in reasonable computational times. It appears that, in general, CPLEX has less difficulty in solving instances from TR dataset as the average computational time required (280 sec) is about 25% lower than that used for CAB dataset (373 sec). This could be due to the dataset's structure and properties.

Further examination of the results in Table 6 reveals that all three proposed PSO algorithms performed well in solving small to medium size instances and for the larger problems H-PSO outperforms the other two. This is the same performance pattern as the one observed earlier from the CAB dataset.

5.2 Scenario II: the case of low hub failure probability

In some applications of the hub-and-spoke systems it is common to assume a low probability of hub failure. To examine how the proposed formulation behaves under these circumstances and to assess what impact this change might have on the final solutions, we tested the same problem instances from the CAB dataset but with low probability of hub failures instead. As

Table 7 A summary of the computational results and comparisons -CAB dataset

	High probability of hub failure					Low probability of hub failure				
	H-PSO		CPLEX			H-PSO		CPLEX		
	ATNC[a]	Time[b]	AUB[c]	Time	Gap (%)	ATNC	Time	AUB	Time	Gap (%)
10 Nodes	465.05	0.34	464.99	372.5	0.0	303.02	2.59	303.02	126.59	0.0
15 Nodes	801.78	7.50	883.25	915.60	37.1	487.58	11.11	485.40	882.56	7.18
20 Nodes	708.83	12.85	–	–	–	483.42	15.33	–	–	–
25 Noes	815.58	21.52	–	–	–	564.49	32.29	–	–	–

[a] Average total network cost, [b] solution time, [c] average upper bounds/best solution

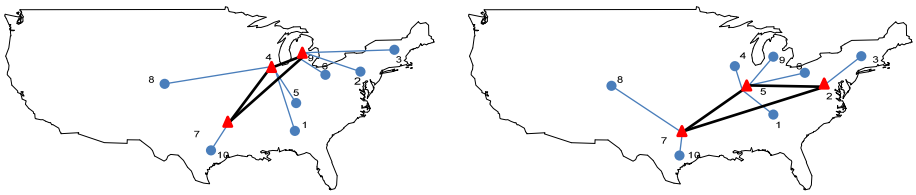


Fig. 15 **a** Optimal network to classical single allocation hub location problem, **b** optimal network to proposed reliable single allocation hub location problem with low probabilities of hub failure

mentioned earlier, in this study we use the low probabilities of hub failures as given in the paper by An et al. (2015).

To be consistent with the analysis conducted in the first scenario, all other parameters used in the formulation and the PSO algorithm are the same.

Furthermore, as H-PSO proved to be more powerful than the other two proposed algorithms from now on we only discuss the results provided by this algorithm. We begin with the small instances with 10 nodes and 3 and 5 hubs from the CAB dataset.

For convenience, the average cost of the optimal/best solutions for all problem instances tested from CAB dataset with low and high probabilities of hub failures are also presented in Table 7. From the computational point of view, it is clear that instances with low probabilities of hub failures could be solved with the proposed formulation (i.e., RSApHLP-III) much faster than that with high probabilities. For instance, the average computational times to solve all 18 benchmark problems with 10 nodes and 3 and 5 hubs are 127 and 373 s respectively.

A further examination of the results show that, in the majority of the cases in this scenario the locations of the hubs, allocations of demand to these hubs and the backup facilities differ from those in the previous scenario. To further explore this issue we repeated the experiment that resulted in two networks depicted in Fig. 14 but using low probabilities of failures, see Fig. 15. It is worth mentioning that the two network configurations provided by RSApHL-III with high (Fig. 14b) and low (Fig. 15b) probabilities of hub failure differ both in terms of the selected hubs and the allocations of the demand to these hubs. As expected the regular transportation cost of the network in Fig. 15a is slightly higher than that of the network of Fig. 15b with hub failure consideration (i.e., 716.98 vs. 767.84). However, the expected transportation cost of the network provided by our formulation (Fig. 15b) is 35% (52.5 vs. 80.4) lower than that provided by the classical model (Fig. 15a). This observation clearly

Table 8 Detailed results for problems with 20 and 25 nodes—CAB dataset

n[a]	p[b]	α[c]	w[d]	TNC[e]	Time (s)	Hubs	Backups
20	3	0.2	0.3	306.76	19.42	20, 7, 8	7, 7, 7, 7, 8, 7, 0, 0, 8, 8, 20, 7, 7, 7, 8, 8, 7, 7, 7, 0
			0.5	443.86	10.91	4, 18, 12	18, 4, 4, 0, 18, 18, 12, 12, 18, 18, 18, 0, 18, 4, 18, 18, 4, 0, 4, 18
			0.7	558.99	5.85	4, 17, 12	17, 4, 4, 0, 17, 17, 12, 12, 17, 12, 17, 0, 17, 4, 17, 17, 0, 4, 4, 4
		0.4	0.3	329.38	3.80	5, 18, 8	18, 5, 5, 18, 0, 18, 8, 0, 18, 8, 8, 5, 18, 18, 8, 8, 5, 0, 5, 18
			0.5	446.41	15.26	4, 18, 12	18, 4, 4, 0, 12, 18, 12, 18, 18, 12, 18, 0, 18, 4, 12, 12, 4, 0, 18, 4
			0.7	652.82	8.47	5, 17, 12	17, 5, 5, 17, 0, 17, 12, 12, 17, 12, 17, 0, 17, 17, 17, 17, 0, 5, 5, 17
		0.8	0.3	379.05	16.72	20, 7, 8	7, 7, 7, 7, 8, 7, 0, 0, 8, 20, 8, 7, 8, 7, 7, 20, 7, 7, 7, 0
			0.5	596.22	19.86	20, 4, 11	4, 4, 4, 0, 11, 11, 4, 4, 4, 4, 0, 4, 4, 11, 11, 20, 4, 11, 4, 0
			0.7	800.75	15.76	20, 4, 11	4, 4, 4, 0, 4, 4, 4, 4, 4, 4, 0, 4, 11, 4, 11, 11, 4, 4, 4, 0
	5	0.2	0.3	268.90	6.23	4, 2, 7, 19, 14	2, 0, 4, 0, 4, 4, 0, 19, 2, 4, 4, 7, 7, 0, 7, 14, 4, 4, 0, 4
			0.5	362.65	23.35	4, 18, 16, 12, 14	14, 4, 14, 0, 18, 16, 18, 16, 18, 16, 14, 0, 14, 0, 18, 0, 4, 0, 14, 18
			0.7	448.84	21.10	20, 4, 17, 7, 12	17, 17, 20, 0, 4, 4, 0, 12, 20, 17, 20, 0, 12, 4, 17, 20, 0, 20, 4, 0
		0.4	0.3	321.04	25.80	1, 20, 7, 8, 14	0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 7, 7, 7, 0, 8, 1, 1, 1, 7, 0
			0.5	465.64	21.29	20, 17, 4, 11, 17	14, 14, 20, 0, 17, 17, 17, 4, 20, 20, 0, 4, 20, 0, 4, 20, 0, 20, 4, 0
			0.7	561.16	9.33	5, 17, 4, 12, 17	4, 14, 5, 0, 0, 17, 4, 14, 4, 4, 4, 0, 12, 0, 4, 12, 0, 5, 5, 17
		0.8	0.3	393.51	7.00	21, 15, 8, 7, 20	2, 0, 2, 15, 2, 2, 0, 0, 2, 8, 15, 7, 20, 2, 0, 7, 2, 2, 7, 0
			0.5	593.36	26.58	1, 20, 7, 8, 15	0, 1, 1, 15, 7, 1, 0, 0, 15, 20, 1, 1, 7, 1, 0, 1, 1, 1, 7, 0
			0.7	772.22	19.16	20, 4, 7, 12, 15	15, 4, 15, 0, 15, 4, 0, 15, 4, 15, 20, 0, 15, 4, 0, 4, 4, 15, 12, 0

Table 8 continued

n[a]	p[b]	α[c]	w[d]	TNC[e]	Time (s)	Hubs	Backups
25	3	0.2	0.3	348.35	11.59	13, 8, 18	18, 18, 13, 18, 18, 18, 8, 0, 18, 8, 8, 13, 0, 18, 8, 8, 13, 0, 13, 13, 8, 13, 13, 18, 13
			0.5	491.23	38.95	5, 18, 12	12, 18, 5, 18, 0, 18, 12, 18, 18, 12, 18, 0, 12, 18, 12, 5, 5, 0, 18, 12, 18, 5, 5, 18, 18
			0.7	635.83	15.30	5, 17, 22	17, 5, 5, 17, 0, 17, 17, 22, 17, 17, 17, 5, 17, 17, 17, 17, 0, 5, 22, 17, 17, 0, 5, 17, 17
		0.4	0.3	379.10	3.14	5, 18, 8	18, 5, 5, 18, 0, 18, 8, 0, 18, 8, 8, 5, 18, 18, 8, 8, 5, 0, 5, 18, 8, 5, 8, 5, 5
			0.5	579.34	20.78	4, 2, 12	4, 0, 4, 0, 4, 4, 2, 12, 2, 4, 4, 0, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4
			0.7	704.09	29.45	5, 18, 12	18, 5, 5, 18, 0, 18, 12, 12, 18, 18, 18, 0, 18, 18, 18, 18, 5, 0, 12, 18, 18, 5, 5, 18, 5
		0.8	0.3	429.57	34.18	20, 8, 15	15, 15, 15, 15, 15, 15, 8, 0, 15, 8, 15, 15, 15, 15, 0, 15, 15, 15, 15, 0, 15, 15, 8, 15, 15
			0.5	668.43	26.47	20, 8, 15	15, 15, 15, 15, 15, 15, 8, 0, 15, 8, 15, 15, 15, 15, 0, 15, 15, 15, 15, 0, 15, 15, 15, 15, 15
			0.7	921.65	29.03	5, 25, 12	25, 5, 5, 25, 0, 25, 12, 25, 12, 5, 25, 0, 25, 25, 25, 25, 5, 5, 25, 25, 25, 25, 5, 25, 0
	5	0.2	0.3	343.23	45.18	1, 5, 17, 11, 12	0, 1, 5, 11, 0, 1, 1, 12, 17, 17, 0, 0, 1, 11, 5, 11, 0, 5, 5, 1, 5, 11, 11, 17, 17
			0.5	459.70	50.06	9, 14, 18, 21, 12	21, 18, 9, 9, 9, 9, 9, 12, 0, 21, 9, 0, 9, 0, 9, 14, 9, 0, 12, 18, 0, 21, 12, 14, 9
			0.7	569.32	28.19	4, 17, 7, 12, 14	14, 17, 4, 0, 17, 17, 0, 7, 17, 7, 7, 0, 7, 0, 7, 7, 0, 17, 12, 17, 7, 7, 12, 14, 4
		0.4	0.3	392.07	47.76	20, 7, 12, 14, 18	14, 18, 18, 18, 18, 18, 0, 7, 18, 7, 7, 0, 7, 0, 7, 7, 20, 0, 7, 0, 7, 7, 12, 14, 18
			0.5	529.29	44.31	5, 17, 8, 12, 14	14, 5, 5, 17, 0, 17, 8, 0, 17, 8, 8, 0, 14, 0, 8, 14, 0, 17, 12, 17, 8, 8, 8, 14, 17
			0.7	655.40	41.14	4, 17, 12, 14, 25	25, 25, 25, 0, 25, 25, 14, 12, 25, 14, 25, 0, 25, 0, 25, 14, 0, 25, 12, 25, 25, 4, 4, 14, 0
		0.8	0.3	471.56	34.87	5, 2, 7, 8, 20	20, 0, 20, 20, 0, 20, 0, 0, 20, 7, 7, 8, 7, 2, 8, 7, 20, 2, 7, 0, 7, 7, 8, 2, 2
			0.5	693.74	42.53	20, 4, 8, 11, 19	4, 4, 4, 0, 4, 4, 11, 0, 4, 11, 0, 19, 11, 4, 4, 11, 4, 4, 0, 0, 11, 19, 19, 4, 4
			0.7	888.99	38.24	20, 4, 8, 12, 19	4, 4, 4, 0, 4, 4, 8, 0, 4, 8, 4, 0, 4, 4, 8, 4, 4, 4, 0, 0, 4, 19, 8, 4, 4
	Average			523.96	23.81		

[a] Number of nodes, [b] number of hubs, [c] discount factor, [d] objective weight, [e] total network cost

shows that even with very low probabilities of hub failure a significant cost saving could be achieved by considering hub failure. This is an interesting finding as one may expect less saving in expected transportation cost as the probabilities of hub failure decreases.

Table 7 also summarises the average costs, computational times and Gaps for instances with 15, 20, and 25 nodes and 3 and 5 hubs. For test problems with 15 nodes, CPLEX is first run for a limited time of 20 min and the results are recorded. The computational times are then extended to 2.5 hours to see whether the quality of the final solutions can possibly be improved. Similar to what we observed in small sized problem instances, our proposed formulation with low probabilities of hub failures is solved with less computational effort. For the test problems with 15 nodes the average optimality gap based on 20 min computing times, is about 8.75% which is significantly lower than that recorded in previous scenario i.e., 36%. Extending the computational times to 2.5 hours, CPLEX found optimal solutions for two test problems with 3 hubs. For the other instances, longer computational times either improved the best upper bounds or in some cases significantly reduced the optimality gaps through tightening the lower bounds.

H-PSO also solved two instances to optimality and improved the upper bounds for a number of those problem instances. The quality of the solutions provided by the H-PSO are clearly comparable with those provided by CPLEX in extended computational times. This observation demonstrates the efficiency and stability of our proposed hybrid PSO algorithm.

A total of 36 larger instances with 20 and 25 nodes are also solved using our hybrid PSO (H-PSO) algorithm. A summary of the results and detail description of each solution for these classes of benchmark problems are presented in Tables 7, 8 respectively. In Table 8 we report the total network cost, the hub locations, demand allocation and the selected backup facilities. For instance, for the 20 nodes instance with 3 hubs, discount factor 0.2, and the objective function weight of 0.3, the cost of the best solution found is 306.76 units; the selected hub locations are 20 (Pittsburgh), 7 (Dallas-FW), and 8 (Denver); the backup facility for nodes 1–4, 6, 12–14, and 17–19 is 7 (Dallas-FW); for nodes 5,9–10, 15 and 16 is hub 8 (Denver); and for node 11 it is hub 20 (Pittsburgh).

6 Conclusion

In this study, a mixed integer quadratic programming (MIQP) formulation is proposed for the reliable single allocation hub location problem with heterogeneous probability of hub failure. The model assigned a backup facility to every demand point in the network for a fast and a low-cost recovery after hub failure. The objective function of the model minimises the weighted transportation cost in regular situation and the expected cost resulted from hub failures.

The proposed formulation is developed in such way to maintain the size of the problem after linearization. To improve the computational efficiency of the resulting MILP model, all constraints initially formulated using Big Ms are substituted with logical constraints known as Indicator Constraints. Using the proposed formulation, we solved all small instances and some medium size problem instances to optimality and provided lower and/or upper bounds for other problems. To tackle large instances, three novel algorithms based on PSO namely PSO-v1, PSO-v2 and H-PSO are proposed. In PSO-v1 and PSO-v2, we investigated the effect of two different solution representations and the mechanisms through which a continuous particle is transferred to a discrete solution. It was found that PSO-v2 that maps a continuous particle to a single discrete solution performs better than PSO-v1. The performance of

the PSO-v2 is then further improved by introducing two new interesting features namely a crossover operator and a memory to keep track of good solutions that are explored during the search. To our knowledge, this is the first time the hybridisation of PSO with those attributes is examined.

We presented and discussed the computational results for two scenarios. In the first scenario it is assumed that hub disruption is more likely to occur and therefore the selected probabilities of hub failures are set relatively high. Whereas in the second a hub disruption is assumed to be an event with a low probability of occurrence. Our numerical results show that even with low levels of probabilities of hub failures the regular transportation cost of the reliable network will not increase significantly and the amount of cost saving remains considerable. This finding, however, is based on experiments involving relatively small instances. Further research required to examine optimal solutions to large problem instances.

In this study the single allocation version of the hub location problem is considered, however the proposed model and our metaheuristic could be easily adopted to address the multiple allocation cases. Other venues of future research including related problems that incorporate capacity constraints and fixed costs are also worth examining.

References

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

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

An, Y., Zhang, Y., & Zeng, B. (2015). The reliable hub-and-spoke design problem: Models and algorithms. *Transportation Research Part B*, *77*, 103–122.

Azad, N., Saharidis, G. K. D., Davoudpour, H., Malekly, H., & Yektamaram, S. A. (2012). Strategies for protecting supply chain networks against facility and transportation disruptions: an improved Benders decomposition approach. *Annals of Operations Research*, *210*, 125–163.

Azizi, N., Chauhan, S., & Vidyarthid, N. (2017). Modelling and analysis of hub-and-spoke networks under stochastic demand and congestion. *Annals of Operations Research* (forthcoming).

Azizi, N., Chauhan, S., Salhi, S., & Vidyarthid, N. (2016). The impact of hub failure in hub-and-spoke networks: Mathematical formulations and solution techniques. *Computers and Operations Research*, *65*, 174–188.

Bonami, P., Lodi, A., Tramontani, A., & Wiese, S. (2015). On mathematical programming with indicator constraints. *Mathematical Programming*, *151*, 191–223.

Camargo, R. S., de Miranda, G, Jr., & Luna, H. P. (2009). Benders decomposition for the hub location problems with economies of scale. *Transportation Science*, *43*(1), 86–97.

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

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

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

Çetiner, S. (2003). An iterative hub location and routing problem for postal delivery systems. Master's thesis, Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey.

Chaovalitwongse, W., Pardalos, P., & Prokopyev, O. A. (2004). A new linearization technique multi-quadratic 0–1 programming problems. *Operations Research Letters*, *32*, 517–522.

Contreras, I., Diaz, J. A., & Marin, A. (2009). Lagrangean relaxation for the capacitated hub single assignment. *OR Spectrum*, *31*(3), 485–505.

Contreras, I., Cordeau, J. F., & Laporte, G. (2012). Exact solution of large-scale hub location problems with multiple capacity levels. *Transportation Science*, *46*, 439–459.
Contreras, I. (2015). Hub location problems. In G. Laporte, S. Nickel, & F. Saldanha-da-Gama (Eds.), *Location science*. Berlin: Springer.
Eiselt, H. A., & Marianov, V. (2009). A conditional p-hub location problem with attraction functions. *Computers and Operations Research*, *36*(12), 3128–3135.
Elhedhli, S., & Wu, H. (2010). A Lagrangean heuristic for hub-and-spoke system design with capacity selection and congestion. *INFORMS Journal of Computing*, *22*(2), 282–296.
Ernst, A. T., & Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, *4*, 139–154.
Ernst, A. T., & Krishnamoorthy, M. (1998). An exact solution approach based on shortest-paths for p-hub median problems. *INFORMS Journal on Computing*, *10*(2), 149–162.
Ernst, A. T., & Krishnamoorthy, M. (1999). Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, *86*, 141–159.
Kennedy, J., & Eberhart R. (1995). Particle swarm optimization. In Proceedings of the IEEE international conference on neural network, Perth, Australia. IEEE Service Centre Piscataway NJ, 1942–1948.
Kim, H., & O'Kelly, M. E. (2009). Reliable p-hub location problems in telecommunication networks. *Geographical Analysis*, *41*, 283–306.
Kratica, J., Stanimirovic, Z., Tosic, D., & Filipovic, V. (2007). Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, *182*(1), 15–28.
Liu, B., Wang, L., & Jin, Y. (2008). An effective hybrid PSO-based algorithm for flow shop scheduling with limited buffers. *Computers and Operations Research*, *35*, 2791–2806.
O'Kelly, M. E. A. (1986). The location of interacting hub facilities. *Transportation Science*, *20*, 92–105.
O'Kelly, M. E. A. (1987). Quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, *32*, 393–404.
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.
Snyder, L. V., Atan, Z., Peng, P., Rong, Y., Schmitt, A., & Sinsoysal, B. (2016). OR/MS models for supply chain disruptions: A review. *IIE Transactions*, *48*(2), 89–109.
Snyder, L. V., & Daskin, M. S. (2005). Reliability models for facility location: The expected failure cost case. *Transportation Science*, *39*, 400–416.
Tran, T. H., O'Hanley, J. R., & Scaparra, M. P. (2017). Reliable hub network design: Formulation and solution techniques. *Transportation Science*, *51*(1), 358–375.
Yaman, H., & Carello, G. (2005). Solving the hub location problem with modular link capacities. *Computers and Operations Research*, *32*, 3227–3245.