




The capacitated single-allocation p -hub location routing problem: a Lagrangian relaxation and a hyper-heuristic approach

Kassem Danach¹ · Shahin Gelareh² · Rahimeh Neamatian Monemi³ 

Received: 28 May 2017 / Accepted: 10 April 2019
© The Author(s) 2019

Abstract

A variant of the hub location routing problem studied in this work, which is the problem of locating a set of hub nodes, is establishing the hub-level network and allocating the spoke nodes to the hub nodes. As a particular property of this problem, each cluster of spoke nodes allocated to a hub constitutes a directed route that starts from the hub, visits all the spokes in the same cluster, and terminates to the same hub. We propose a hybrid of hyper-heuristic and a relax-and-cut solution method, which includes cooperation among several low-level heuristics governed and controlled by a learning mechanism. This hybridization provides a mechanism in which the obtained dual information through the Lagrangian relaxation (bundle) method being utilized to guide the local searches for constructing/improving feasible solutions. Several classes of valid inequalities as well as efficient separation routings are also proposed for being used within the relax-and-cut approach. Our extensive computational experiments confirm the efficiency of this solution method in terms of quality as well as computational time.

Keywords Hub location problem · Hyper-heuristic · Meta-heuristic · Reinforcement learning

✉ Rahimeh Neamatian Monemi
r.n.monemi@gmail.com

¹ Faculty of Economics and Business Administration, Islamic University of Lebanon, Beirut, Lebanon

² Département de Réseaux et Télécommunications, IUT de Bethune, Université Artois, 62400 Béthune, France

³ IT Innovation Centre, University of Southampton, Southampton, UK

1 Introduction

Hub-and-spoke operation is now an almost dominant style of operation both in telecommunication and transportation. The hubs serve as consolidating, switching, and sorting centers. In transportation and logistics, the hub-and-spoke structures are present in almost all modes of transport (cross-docks in national level, major airports, and container ports in continental and global levels). Hubs are facilities at which arriving flows of commodities in smaller volume originated from the spokes are consolidated, sorted and re-distributed (repartitioned) in a larger volume to be transported on fewer highly utilized links and sent to another hub that is either the final destination or is a transshipment point where the spoke final destination is allocated to. The main motivation for deploying such a flow network structure is in exploiting economies of scale (in terms of time and/or cost) in transporting higher volumes on fewer significantly more efficient corridors (connecting hubs).

Given a graph $G(V, A)$ where V is the set of nodes as origins and destinations and A is the set of all possible arcs, the Hub Location Problem (HLP) seeks a subset of nodes in V as hub nodes. Subsequently, each spoke (non-hub) node in V is allocated to one or more hubs. In classical models, two main classes are distinguished Alumur and Kara (2008): (1) *Single Allocation*: wherein each spoke node is assigned to exactly one hub node, and (2) *Multiple Allocation*: offering the possibility of allocating a spoke node to more than one hub node. Moreover, one may also distinguish between *capacitated* and *un-capacitated* variants. Some authors also distinguish between the cases where cardinality of the set of selected hub nodes is defined exogenously and when it is an endogenous part of the problem. This topic has been one of the very active areas of research in the past 2 decades.

1.1 Problem description

Given a graph G , a fixed integer p , as the number of hub nodes to be installed, Γ as the minimum number of spoke nodes to be allocated to each hub node, C as the capacity of each feeder route, w_{ij} as the demand of flow to be transferred/transported from i to j and t_{ij} as the cost of traveling between every two nodes i and j , which is assumed, in this study, as the time needed to travel between nodes i and j . We also introduce d_i as the load of transporter after leaving node i on a spoke-level route (the load at arrival minus the total demand of i from every other node plus the total supply of i to every other node). We seek to construct from the scratch a hub-and-spoke network, which consists of a complete hub-level subgraph with p hub nodes and p spoke-level directed cyclic routes, each of which composed of a hub node and all the spoke nodes allocated to it. The hub-level network is a complete subgraph where pair of connected hubs have a bidirectional link. The objective function of our problem is minimizing the total transit time composed of transportation time plus transshipment time in the whole network, provided that the total volume of flow on a spoke-level arc does not violate the capacity of the transporter on it, given a homogeneous fleet of transporters.

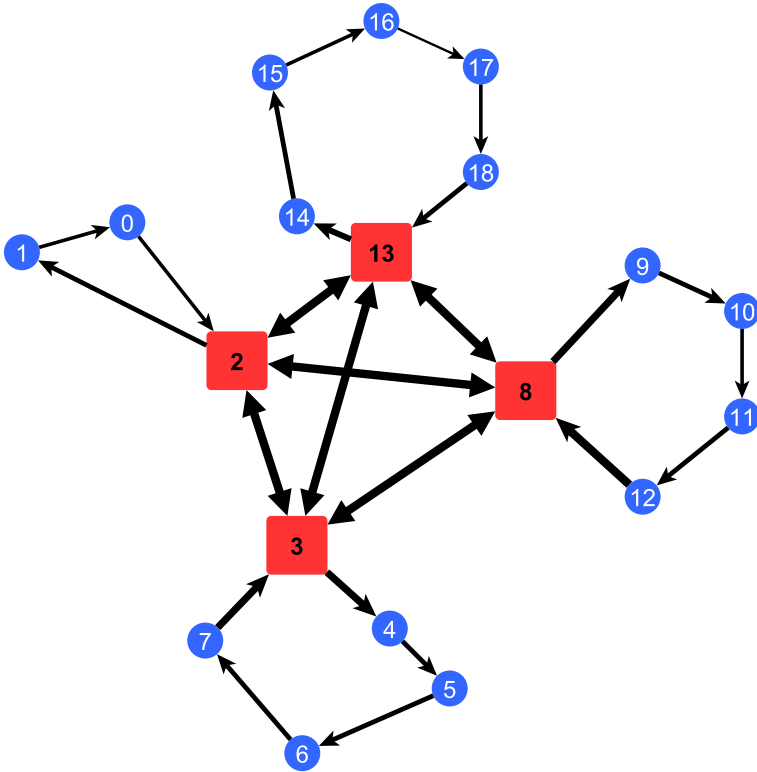


Fig. 1 A solution to a network with 4 hubs and 19 nodes

This problem is a special case of the *Bounded Cardinality Capacitated Hub Routing Problem (BCCHRP)* proposed in Gelareh et al. (2015) in which the lower and upper bounds on the number of hub nodes coincide. We refer to this problem as *Capacitated Single-Allocation p -Hub Location Routing Problem (CSApHLRP)*.

Figure 1 represents a solution for an instance with $|V| = n = 19$ nodes and $p = 4$ fully interconnected hub nodes. Nodes 2, 3, 8, and 13 represent hubs, while all the remaining nodes are spoke nodes. One observes that each hub together with spoke nodes allocated to it forms a direct cycle. In this solution, *four* cycles are defined as follows: (i) $2 \rightarrow 1 \rightarrow 0 \rightarrow 2$, (ii) $3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 3$, (iii) $8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 8$ and, (iv) $13 \rightarrow 14 \rightarrow 15 \rightarrow 16 \rightarrow 17 \rightarrow 18 \rightarrow 13$.

1.2 Literature review

In this work, we are focusing on a problem that fits within the category of capacitated single-allocation hub location problems (CSAHLp). In the following, we review some of the related works in the literature with a particular emphasis on the single-allocation scheme and exogenous number of hubs in modeling and (meta)

heuristic approaches in solution methods. The references brought here are sorted by publication year.

Rabbani and Kazemi (2015) proposed a genetic algorithm and a simulated annealing for solving an un-capacitated multiple allocation p -hub center problem.

Simulated annealing (SA) and iterated local search (ILS) were proposed by Zarandi et al. (2015) to solve single-allocation hub median problem.

The un-capacitated p -Hub location problem wherein each spoke node should be connected to exactly r hubs (where r is a predefined fixed number) is studied by Peiró et al. (2014). The authors proposed a GRASP algorithm to solve the problem.

Rodríguez-Martín et al. (2014) proposed a Mixed Integer Programming (MIP) formulation and a branch-and-cut algorithm for the hub location and routing problem. In this study, the single-allocation scheme is considered where the capacity constraint is defined in terms of number of spokes per route, and the number of hub nodes is considered as a fixed parameter.

A Multiple Ant Colony Optimization (MACO) algorithm is proposed by Ting and Chen (2013) to solve the capacitated single-allocation hub location problem. In this study, the problem is decomposed into two subproblems: (1) the facility location problem and (2) the multiple depot vehicle routing problem. The two subproblems are treated together within MACO where collaboration of colonies is operated by interchanging information through pheromone renewing the chosen location and customer assignment.

Gelareh et al. (2013) proposed an MIP formulation for the hub-and-spoke network design problem and fleet deployment of liner shipping, to minimize weighted sum of transit times while considering fixed deployment costs. The proposed MIP is based on a 4-index formulation of HLPs and allocated spokes forming directed cycles (the so-called *string*) to hubs. Note that, in this work, the fleet of vessels defining the capacity of strings are to be determined from a given heterogenous fleet.

Another VNS is proposed by Jarboui et al. (2013) for the un-capacitated single-allocation hub location problem.

Kratica et al. (2012) proposed a genetic algorithm (GA) for solving an un-capacitated multiple allocation hub location problem. In this study, binary encoding and genetic operators are employed, while the run time is optimized using a caching technique.

Ilić et al. (2010) investigate the Un-capacitated single-allocation p -hub median problem. In this work, a Variable Neighborhood Search (VNS) is considered where a variable neighborhood descent (VND)-based local search using three different neighborhood structures is proposed.

Another simulated annealing for single-allocation capacitated hub location problem was proposed by Yu et al. (2010), given that routes and hubs were constrained by capacity.

Correia et al. (2009) proposed an MIP formulation for the single-allocation hub location problem where capacity on hubs is not an exogenous value, but it is a variable with discrete choices of value. Several variants of this problem have been studied in the literature, such as capacitated and un-capacitated for the single-allocation problem (Correia et al. 2009), and capacitated flow splitting for the multiple allocation one (Campbell et al. 2007).

Randall (2008) proposed an ant colony meta-heuristic optimization algorithm to solve the capacitated single-allocation hub location problem. In this work, four variations of ant colony optimisation meta-heuristic are developed to explore different construction modeling choices.

A heuristic based on (GA) is proposed by Stanimirovic (2008) to solve uncapacitated multiple allocation p -hub median problem. This GA uses binary representation of the solutions. A mutation operator with frozen bits and a caching technique has been employed to solve the problem.

As a relevant work, Ge et al. (2007) investigate the Un-capacitated Single-Allocation p -Hub Location problem. In this study, the authors aim at locating p hub nodes in the network and allocating non-hub nodes to the hubs under the single-allocation scheme where the main objective is to minimize the transportation cost.

Ebery et al. (2000) proposed an MIP formulation and a heuristic for capacitated multiple allocation hub location problem. They hybrid method that combines evolutionary algorithm (EA) with a branch-and-bound algorithm (B&B) to solve a capacitated multiple allocation hub location problem. In this work, the proposed EA selects the set of hubs, while B&B continues with the allocation part of the problem by allocating the non-hubs to the located hubs.

A two-phase tabu search algorithm considering location as the first phase and routing as the second phase with short-term memory is investigated by Tuzun and Burke (1999), to solve capacitated hub location problem considering single-allocation scheme.

Abdinnour-Helm (1998) developed a hybrid method based on combination of genetic algorithm and tabu search to solve an un-capacitated hub location problem with single-allocation scheme. A Genetic Algorithm is used to choose hubs, while a Tabu Search is used to assign spokes to the located hubs.

An application of the p -median Problem is addressed by Rolland et al. (1997). They proposed a tabu search algorithm employing short-term and long-term memory technics, as well as a strategic oscillation for random tabu list sizes.

Klincewicz (1992) studied the p -hub location problem, and to avoid local optima in p -hub location problem, they proposed two meta-heuristics: a tabu search and a GRASP algorithm. The objective of this work is to minimize the total costs of sending traffic over the links. In the proposed two algorithms, local search procedure is based on the two-exchange neighborhood structure.

The work presented in this paper has some similarities with the work presented in Rodríguez-Martín et al. (2014). However, our work is distinguished from Rodríguez-Martín et al. (2014), as follows: in Rodríguez-Martín et al. (2014), the objective function is only to minimize the total transportation cost, while in here, we also considered time as the transshipment cost. In the former, the spoke-level routes are undirected, while we consider direction for the network, such that flow on the spoke-level route circulates in only one direction. Finally, in Rodríguez-Martín et al. (2014), the capacity is defined as the maximum number of spokes along a spoke-level route (the number of physical ports on a machine, as a constraint in telecommunication), while we define it as the volume of flow circulating on a spoke-level route that needs to respect the capacity of transporters available on it.

Our work is also distinguished from the work presented in Gelareh et al. (2015), by the fact that we do not consider any upper/lower bound on the number of hubs. Rather, we assume a fixed cardinality (p) as an exogenous parameter.

To the best of our knowledge, so far, the non-exact methods applied to solve variants of hub location problem were either a meta-heuristic algorithm or a classic heuristic algorithm.

Table 1 summarizes some key features in closely related contributions.

1.3 Motivation, scope, and contribution

This work has initially been inspired from an application in the distribution chain of pharmaceutical products. Assume we have a chain of consumers in a number of different pharmacies, while a company is to deliver their demand. Local pharmacies are able to store a small amount of products which are more demanded; however, this usually happens that they receive a new prescription which is not available in their stock. Thus, they need to order it and the maximum delay to deliver the ordered prescription needs to be short (for formal situations, no more than a day). What the pharmacies need per day is usually in a reasonably moderate amount, and consequently, a truck is enough to serve quite a few number of pharmacies in a tour every morning. The aim is at designing a service network meeting the criteria while optimizing the time and resource consumption.

This work contributes to the state-of-the-arts as follows: In addition to presenting a mathematical model and identifying several classes of valid inequalities and the respective separation routines, we propose a tailored relax-and-cut algorithm based on the Lagrangian relaxation technique. The primal/dual information (such as Lagrangian multipliers and reduced cost of spoke-level variables), which are accumulated during the iterations of solving Lagrangian dual, are exploited to guide the heuristic algorithms of our hyper-heuristic framework. The proposed hyper-heuristic algorithm is comprised of a portfolio of low-level heuristics (some of which are using the Lagrangian relaxation information) and a *heuristic selection* method that learns—in the course of process—how to dynamically replace the selected heuristic among the existing heuristics aiming at reaching a higher likelihood of success and improvement. The latter is, in fact, a reinforcement learning method that has been inspired from the concept of *association rules* in the *business data-mining* world. We are unaware of any similar work in the literature and, to the best of our knowledge, this is the first work tackling a variant of hub location problems using a hyper-heuristic approach (that, in particular, exploits Lagrangian relaxation information and data-mining knowledge).

The remainder of this paper is organized as follows: In Sect. 2, we present the mathematical model based on the model presented in Gelareh et al. (2015). Then, Sect. 3 is divided into three parts. In the first part, a Lagrangian relaxation algorithm and several classes of valid inequalities are proposed for the problem. In the second part of this section, a hyper-heuristic solution approach and its components exploiting information of Lagrangian dual are presented. In the third part, the overall algorithm and some separation routines for the identified valid inequalities are proposed

Table 1 A summary of main elements of the relevant contributions in the literature

Work	Allocation scheme	No. hubs	Objective	Capacity	Cycle length	No. vehicles	Solution method
Nagy and Salhi (1998)	Pick-up/delivery routes	Endogenous	Cost	Yes	Yes	Endogenous	MIP + heuristic
Cetiner et al. (2006)	Multiple allocation	Endogenous	Cost + fleet	No	Yes	Endogenous	Heuristic
de Camargo et al. (2013)	Single allocation	Endogenous	Cost	No	Yes	Endogenous	MIP + Benders decomposition
Wasner and Zäpfel (2004)	Multiple allocation	Endogenous	Cost	Yes	Yes	Endogenous	MIP + heuristic
Rodríguez-Martín et al. (2014)	Single allocation	Exogenous	Cost	#. Spoke per route	No	One per route	MIP + branch-and-cut
Gelareh et al. (2013)	Single allocation	Exogenous	Cost + fleet	Yes	Multiple of weeks	Variable	MIP + Lagrangian decomposition
Gelareh et al. (2015)	Single allocation	$q = 3 \leq \dots \leq p$	Time (transit + trans-shipment)	Yes	≥ 2 Spokes	One per route	MIP + branch-and-cut + Benders
Current work	Single allocation	Exogenous	Time (transit + trans-shipment)	Yes	≥ 2 Spokes	One per route	MIP + Lagrangian relaxation + hyper-heuristic

Table 2 Model parameters

w_{ij}	The flow from i to j
t_{ij}	The distance/time on a direct link on the edge (arc) $i - j$
α	The factor of economies of scale (the factor of travel time efficiency over hub edges)
p	The upper bound on the number of hubs (depots)
q	The lower bound on the number of hubs (depots)
Γ	The minimum number of spokes allocated to each hub/depot node
C^v	The capacity of each vehicle for each feeder network
φ^k	The (fixed) average transshipment time at hub k

Table 3 Decision variables

x_{ijkl}	The fraction of flow from i to j traversing inter-hub edge $\{k, l\}$
s_{ijkl}	The fraction of flow from i to j traversing non-hub edge $\{k, l\}$
r_{ij}	1, If the arc (i, j) belongs to a spoke-level route, 0 otherwise
z_{ik}	1, If node i is allocated to node k where k is a hub, 0 otherwise

to be used in a relax-and-cut algorithm. Computational experiments are reported in Sect. 4. Finally, in Sect. 5, we conclude our work and present the possible future work.

2 Mathematical formulation of the CSA p HLRP

A 2-index model (cardinality of the set of decision variables is of $o(n^2)$) is proposed in Gelareh et al. (2015) for the Bounded Cardinality Capacitated Hub Routing Problem (BCCHRP). Bearing in mind the idea of the proposed model for the BCCHRP, here, in this paper, we are going to model the CSA p HLRP. The variables and parameters required to model this problem are introduced in Tables 2 and 3.

(CSA p HLRP)

$$\min \sum_{i,j,k,l} (t_{kl}(s_{ijkl} + \alpha x_{ijkl})) + \sum_{i,j,k,l:(k \neq i \vee l \neq j)} (\varphi^k + \varphi^l)x_{ijkl} \tag{1}$$

s. t.

$$q \leq \sum_k z_{kk} \leq p \tag{2}$$

$$\sum_l z_{kl} = 1 \quad \forall k \in V \tag{3}$$

$$z_{ik} \leq z_{kk} \quad \forall i, k \in V : k \neq i \tag{4}$$

$$\sum_i z_{ik} \geq \Gamma z_{kk} \quad \forall k \in V \tag{5}$$

$$\sum_{j \neq i} r_{ij} = 1 \quad \forall i \in V \tag{6}$$

$$\sum_{j \neq i} r_{ji} = 1 \quad \forall i \in V \tag{7}$$

$$r_{ij} + r_{ji} \leq 2 - z_{ik} - z_{jl} \quad \forall i, j, k, l \in V : j \neq i, \quad k \neq l \tag{8}$$

$$r_{ij} + r_{ji} \leq 1 \quad \forall i, j \in V : j \neq i, \tag{9}$$

$$\sum_{k \neq i} (x_{ijk} + s_{ijk}) = 1, \quad \forall i, j \in V : j \neq i, \tag{10}$$

$$\sum_{l \neq j} (x_{ijl} + s_{ijl}) = 1, \quad \forall i, j \in V : j \neq i, \tag{11}$$

$$\sum_{l \neq i, k} (x_{ijkl} + s_{ijkl}) = \sum_{l \neq j, k} (x_{ijlk} + s_{ijlk}), \quad \forall i, j, k \in V, \quad k \notin \{i, j, \}, \tag{12}$$

$$\sum_{l \neq k} x_{ijkl} \leq z_{kk} \quad \forall i, j, k \in V : j \neq i, \quad k < l \tag{13}$$

$$\sum_{l \neq k} x_{ijlk} \leq z_{kk} \quad \forall i, j, k \in V : j \neq i, \quad k < l \tag{14}$$

$$s_{ijkl} \leq r_{kl} \quad \forall i, j, k, l \in V : l \neq k \tag{15}$$

$$\sum_{ijl: j \neq i} w_{ij} s_{ijkl} \leq C^v, \quad \forall k \in V \tag{16}$$

$$r \in \mathbb{B}^{|V|^2}, \quad z \in \mathbb{B}^{|V| \times |V|}, \quad x_{ijkl}, s_{ijkl} \in \mathbb{R}_{[0,1]}^{|V|^4} \tag{17}$$

The objective function (1) is comprised of two parts; the first accounts for the total transportation time and the second part is the transshipment time. The transportation part is composed of travel time on the hub-level edges as well as the spoke-level arcs. The travel time on the hub-level edges is discounted by α , because the transporters offer faster services. The transshipment time is accounted for the hub nodes along every O-D path.

Constraints (2) restrict the number of installed hubs, while constraints (3) ensure that every node is assigned to one and only one hub. A self allocation at i shows

that i is a hub (i.e., $z_{ii} = 1$). Constraints (4) indicate that a spoke node can only be assigned to a hub node. However, constraints (5) are to make sure that at least Γ nodes are allocated to every designated hub node. Constraints (6), (7) are to guarantee that a single spoke-level arc leaves (arrives at) every spoke node. Constraints (8) are to avoid having spoke-level arcs between spoke nodes which are allocated to different hub nodes. Constraints (9) ensure that a spoke-level arc can exist only in one direction. Constraints (10)–(12) correspond to flow conservation for every O-D pair. Constraints (13) and (14) are to ensure that traversing an edge composed of two hubs is equivalent to traversing a hub edge connecting those two hubs. Constraints (15) guarantee that the flow on each feeder network travel in the direction of existing arcs. Constraints (16) are to ensure that flow traversing every feeder network is constrained to the capacity of the vehicle deployed on it. This is given as a parameter.

To ensure that the above mathematical model represents correctly the real characteristics of the problem, a set of sub-tour elimination constraints will be added to the model (will be explained in detail in proposition Proposition 1 to guarantee that there will be no sub-tour appears in any feasible solution.

3 Solution algorithm

The proposed solution algorithm is a lower/upper bound solution method. In this algorithm, the lower bounds are calculated using a Lagrangian relaxation-based relax-and-cut approach allowing development of reduced costs of the spoke-level design variables as a by-product of iterations of solving Lagrangian dual problem. The choice of this relaxation (from among other possibilities) is based on a trade-off between the quality of bounds and the computational efficiency of solving the Lagrangian dual problem. The primal (approximate primal solution) and dual information (Lagrangian multipliers) are used to guide different heuristic algorithms that are components of our hyper-heuristic framework.

3.1 Lagrangian relaxation

Lagrangian relaxation for solving (mixed) integer programming problems was first proposed in Geoffrion (1974), Geoffrion and Bride (1978) and later in Fisher (1981, 2004). The idea behind this method is to relax *complicating constraints* by penalizing the objective function upon violation of these constraints. The relaxed problem normally has a special structure and can be solved more efficiently comparing the original problem (see Guignard 2003 for a comprehensive survey of the method.).

Three well-known methods are commonly practiced in the literature for solving the Lagrangian dual problem. The oldest and most well-known one is the *subgradient* method as an iterative method for solving convex minimization problems. The subgradient method was originally proposed in the 60s in the former Soviet Union. Almost at the same time, a very similar method has been proposed in Held and Karp (1971) for solving traveling salesman problem. Later, Lemarechal (1975) proposed the well-known bundle methods as an extension to the simple subgradient. The volume

algorithm was later proposed in Barahona and Anbil (2000) as a method which simultaneously produces a primal feasible solution and a dual bound for the problem. A further analysis of volume algorithm and its relation with bundle method has been studied in Bahiense et al. (2002).

Several variants of the subgradient-based methods have been proposed in the literature. Here, based on some observations from our preliminary computational experiments, we opted to use the well-known bundle method of Frangioni (1995) and the corresponding code (publicly available).

We opted to relax constraints (6), (7), (8), (9), and (15) in the Lagrangian fashion. Let u_{ij}^1, u_{ij}^2 and u_{ij}^4 for all $i, j \in V, i \neq j$, and u_{ijkl}^3 and u_{ijkl}^5 for all $i, j, k, l \in V$, while $i \neq j$ and $k \neq l$ be the Lagrangian multipliers corresponding to (6), (7), (8), (9), and (15) constraints. By doing so, the dual information is going to be retrieved from the coefficients of r_{ij} and z_{ij} variables. A solution to such a Lagrangian relaxation, which is a lower bound to the initial model, partially describes the network design and route structures. The dual information can be used to either construct a partially feasible solution or, in general, to guide some of the heuristics in our hyper-heuristic framework. This is called a *warm start* for a heuristic/huper-heuristic algorithm.

In fact, this relaxation has been chosen among four different scenarios some of which could result in a decomposable subproblem. However, our extensive initial computational experiments find this relaxation as the best trade-off between the quality of bound and computational intractability, among others:

(LRX-CSApHLRP)

$$\begin{aligned}
 \min \quad & \sum_{i,j,k,l} (t_{kl}(s_{ijkl} + \alpha x_{ijkl})) + \sum_{i,j,k,l:(k \neq i \vee l \neq j)} (\varphi^k + \varphi^l)x_{ijkl} \\
 & + \sum_{i,j \neq i} u_{ij}^1 \left(\sum_{j \neq i} r_{ij} - 1 \right) + \sum_{i,j \neq i} u_{ij}^2 \left(\sum_{j \neq i} r_{ji} - 1 \right) \\
 & + \sum_{ijkl:k \neq l, j \neq i} u_{ijkl}^3 (r_{ij} + r_{ji} - 2 + z_{ik} + z_{jl}) + \sum_{i,j \neq i} u_{ij}^4 (r_{ij} + r_{ji} - 1) \\
 & + \sum_{ijkl,l \neq k} u_{ijkl}^5 (s_{ijkl} - r_{kl})
 \end{aligned} \tag{18}$$

s. t.

$$\begin{aligned}
 & (2), (3), (4), (5) \\
 & (10), (11), (12) \\
 & (13), (14), (16) \\
 & r \in \mathbb{B}^{|V|^2}, \quad z \in \mathbb{B}^{|V| \times |V|}, \quad x_{ijkl}, s_{ijkl} \in \mathbb{R}_{[0,1]}^{|V|^4}
 \end{aligned} \tag{19}$$

3.1.1 Valid inequalities for a relax-and-cut

In the proposed Lagrangian relaxation model (LRX-CSA ρ HLRP), all the constraints containing route variables r are relaxed and added to the objective function multiplied by the Lagrangian multipliers. As a consequence, a feasible solution to the resulting network design problem tends to be infeasible (or partially feasible) with respect to the initial model, in many iterations when solving the Lagrangian dual problem. Such infeasibility includes the situation wherein the spoke/feeder nodes allocated to a same hub are forming more than one route; or the cases where a feeder/spoke node allocated to a given hub node lies among the spoke/feeder nodes allocated to one or more other hub nodes.

To circumvent such situations, some valid inequalities are identified and LRX-CSA ρ HLRP is solved by a branch-and-cut process, and the necessary valid inequalities are separated and added upon need. When using a modern MIP solver such as CPLEX, this does not cause efficiency issues, because in each iteration of subgradient algorithm, when LRX-CSA ρ HLRP is solved, only the objective function is being updated and all the separated cuts are retained in the memory (no need to be separated again) if a warm start can be carried out (as is the case in the modern MIP solvers such as CPLEX).

For all $S \subseteq V$, let define $\delta(S)^+ = \{(i, j) | i \in S, j \in V \setminus S\}$, $\delta(S)^- = \{(i, j) | j \in S, i \in V \setminus S\}$ and $\gamma(S) = \{(i, j) | i \in S, j \in S, j \neq i\}$. In the following, we present some valid inequalities for the problem.

Proposition 1 *A kind of Generalized Sub-tour Elimination Constraints (GSECs) ensures that the cut size in every feeder route in a feasible solution is at least equal to 1 in a directed graph (and is equal to two in an undirected graph).*

Let $S \subset V$ and $S' = V \setminus S$:

$$\sum_{k \in S, l \in S'} r_{kl} \geq \sum_{j \in S'} z_{ij}, \quad \forall i \in S. \quad (20)$$

Proof Given $S \subseteq V$. The right-hand side of this inequality represents the total number of spoke nodes in S which are allocated to hubs belonging to S' that is $(V \setminus S)$. Thus, obviously corresponding to every $i \in S$ allocated to hub $j \in S'$, we must have at least one spoke edge, such that one of its end-nodes (i) is in S and the other one (j) is in $V \setminus S$. However, we still might have other spoke nodes in $V \setminus S$ that are connected to i . Thus, these inequalities are valid for the $\mathcal{P}(\text{CSA}\rho\text{HLRP})$.

Proposition 2 *The following inequalities are valid for (CSA ρ HLRP):*

$$\sum_{i, j \in S: j \neq i} (r_{ij} + r_{ji}) \leq |S|. \quad (21)$$

Proof To prove this proposition, let us assume two different cases for every feasible solution:

- (1) Where all the nodes in S are all placed on a single feeder spoke ring. In other word, all nodes in S are allocated to a same hub.
- (2) Where S contains nodes on different feeder rings.

In the first case, a number of arcs between all i and j in S are at most equal to $|S|$ and this number is exactly equal to $|S|$ if S contains all the nodes allocated to the hub. This is the case, where the constraint is binding.

In the second case, if a pair i, j in S is allocated to different hubs, there must not be any direct arc connecting them. Thus, the constraint is valid.

Proposition 3 *The following 2-matching constraints for the main and feeder routes are valid for the CSApHLRP polytope, $\mathcal{P}(\text{CSApHLRP})$:*

$$\sum_{a \in \gamma(H)} r_a + \sum_{a \in (\delta^+(T) \cup \delta^-(T))} r_a \leq |H| + \frac{|\mathcal{T}| - 1}{2}, \tag{22}$$

where for all $H \subset \mathcal{V}$ and all $\mathcal{T} \subset (\delta^+(H) \cup \delta^-(H))$ satisfying the following:

- (i) $|\{i, j\} \cap H| = 1, \forall (i, j) \in \mathcal{T}$,
- (ii) $\{i, j\} \cap \{k, l\} = \emptyset, \forall (i, j) \neq (k, l) \in \mathcal{T}$, and
- (iii) $|\mathcal{T}| \geq 3$ and odd.

$\mathcal{T} \subset \mathcal{V}$ is called *teeth* (with respect to the main route subgraph or the feeder route subgraph, that is, $\mathcal{T}^r, H^r, \mathcal{T}^y$ and H^y) and $H \subset \mathcal{V}$ is known as a *handle*.

Proof Given that, for $a = (i, j)$, we note $r_{ij} = r_a = r(a)$ whenever $r_{ij} = r_{ji}$ (direction of the arc does not make any change). One knows that, in every feasible solution of CSApHLRP, we have the following:

$$2r(\gamma(H)) + r(\delta^+(H)) + r(\delta^-(H)) = \sum_{i \in H} (r(\delta^+(i)) + r(\delta^-(i))),$$

and from constraints bound constraints $r_{ij} \leq 1$, one can obtain the following:

$$\sum_{k \in H} \sum_{l \neq k} (r_{kl} + r_{lk}) \leq 2|H|.$$

Hence

$$\begin{aligned} 2|H| &\geq 2r(\gamma(H)) + r(\delta^+(H)) + r(\delta^-(H)) \\ &= 2r(\gamma(H)) + r(\delta^+(H/\mathcal{T}) + r(\delta^+(\mathcal{T})) + r(\delta^-(H/\mathcal{T})) + r(\delta^-(\mathcal{T}))). \end{aligned} \tag{23}$$

Given that $r(\delta^+(\mathcal{T}) \cup \delta^-(\mathcal{T})) \leq |\mathcal{T}|$ deduced from the bound constraints $r_a \leq 1$, we add this set of constraints to (23) and we obtain the following:

$$\begin{aligned} 2|H| + |\mathcal{T}| &\geq 2r(\gamma(H)) + r(\delta^+(H/\mathcal{T})) + r(\delta^-(H/\mathcal{T})) + 2r(\delta^+(\mathcal{T})) + 2r(\delta^-(\mathcal{T})) \\ &\geq 2r(\gamma(H)) + 2r(\delta^+(\mathcal{T})) + 2r(\delta^-(\mathcal{T})). \end{aligned} \quad (24)$$

Again, given that $|\mathcal{T}|$ is an odd number, by multiplying both sides by $\frac{1}{2}$, we conclude the following:

$$r(\gamma(H)) + r(\delta^+(\mathcal{T})) + r(\delta^-(\mathcal{T})) \leq |H| + \frac{|\mathcal{T}| - 1}{2} \quad (25)$$

and the proof is complete.

Proposition 4 *The following comb inequalities for both main and feeder networks are valid for $\mathcal{P}(\text{CSApHLRP})$:*

$$\sum_{a \in A(H)} r(a) + \sum_{j=1}^t r(A(T_j)) \leq |H| + \sum_{j=1}^t |T_j| - \frac{3t+1}{2}, \quad (26)$$

where for all $S \subset \mathcal{V}$ and all $T_i \subset \mathcal{V}$, $\forall i \in \{1, \dots, t\}$ satisfying the following:

- (i) $H, T_1, T_2, \dots, T_t \subseteq \mathcal{V}$,
- (ii) $T_j \setminus H \neq \emptyset$, $\forall j \in \{1, \dots, t\}$,
- (iii) $T_j \cap H \neq \emptyset$, $\forall j \in \{1, \dots, t\}$,
- (iv) $T_i \cap T_j = \emptyset$, $\forall j \in \{1, \dots, t\}$, and
- (v) $t \geq 3$ and odd.

T_1, T_2, \dots, T_t are called *teeth* (with respect to the main subgraph or the feeder route subgraph, i.e., T_i^r, H_i^r, T_i^y and H^y , $\forall i \in \{1, \dots, t\}$) and $H \subset \mathcal{V}$ is a *handle*.

Proof For $S \subset \mathcal{V}$, in every feasible solution of CSApHLRP we have the following:

$$r(\gamma(H)) = |H| - \frac{\delta^+(S) + \delta^-(S)}{2}, \quad (27)$$

and for every $T_j \subset \mathcal{V}$, $j \in \{1, \dots, t\}$, we have the following:

$$r(\gamma(T_j)) = |T_j| - \frac{\delta^+(T_j) + \delta^-(T_j)}{2}. \quad (28)$$

Furthermore, from (27) and (28), one yields the following:

$$\begin{aligned} r(\gamma(H)) + \sum_i r(\gamma(T_i)) &= |H| + \sum_j |T_j| \\ &\quad - \frac{1}{2} \left((\delta^+(H) + \delta^-(H)) + \sum_j (\delta^+(T_j) + \delta^-(T_j)) \right). \end{aligned} \quad (29)$$

Let $\delta_i^+(H) \cup \delta_i^-(H)$ denote the cut set associated with arcs having one end-point in $H \cap T_i$ and another end-point outside H . We know that $\delta^+(H) + \delta^-(H) \geq \sum_i (\delta_i^+(H) + \delta_i^-(H))$.

It can be easily shown that $(\delta_i^+(H) + \delta_i^-(H)) + (\delta^+(T_i) + \delta^-(T_i)) \geq 3$. We also know that $\delta^+(H) + \delta^-(H)$ and $\delta^+(T_i) + \delta^-(T_i)$ are even numbers and, therefore, $\delta^+(H) + \delta^-(H) + \sum_i (\delta^+(T_i) + \delta^-(T_i)) \geq 3t + 1$. By substituting in (29), the proof is complete.

3.2 Hyper-heuristic for CSApHLRP

Hyper-heuristics may be seen as a generalization of (meta-)heuristics and an easier way for categorize a large body of literature of heuristics and meta-heuristics that was rather difficult to classify previously. A significant part of the optimization literature is devoted to the use of a given Meta-heuristic (or a hybrid scheme) to solve the problem and often a comparison among a bunch of such methods is presented in different articles. According to Burke et al. (2009), hyper-heuristic is a search methodology or learning mechanism to select or generate heuristics to solve a specific combinatorial problem, while, in this work, we deal with the class of *heuristic selection* hyper-heuristics (the notion of *heuristics to choose heuristics in the context of combinatorial optimisation*). Cowling et al. (2001) focuses on the strategies for systematic alternation among different low-level heuristics in different steps of the search (e.g., supported by learning mechanisms). This kind of hyper-heuristic is considered as a high-level approach that, given a problem and a number of low-level heuristics, can select and apply a suitable low-level heuristic at each iteration (Burke et al. 2003).

Using a combination of an *operator (low-level heuristic) selection vector* and a *solution acceptance criterion*, a hyper-heuristic must offer a 'good' solution (when exists), within an unseen low-level problem domain (Marshall et al. 2015). Figure 2 depicts a general structure of a hyper-heuristic.

It must be noted that in a meta-heuristic, one conducts a search in the space of feasible solutions, while, in a hyper-heuristic, the search is carried out over the space of heuristic algorithms (Burke et al. 2003). Our goal here is to select the best series (not necessarily a sequence) of (meta-)heuristics to be applied to obtain high-quality solution(s).

Hyper-heuristics have been successfully applied to solve various problems in routing and scheduling (see Danach et al. 2015a, b; Monemi et al. 2015; Marshall et al. 2014; Garrido and Riff 2010; Garrido and Castro 2012), scheduling (see Ahmed et al. 2015; Zheng et al. 2015; Aron et al. 2015), and bin packing (see Sim et al. 2015; Beyaz et al. 2015; Ross et al. 2002) among others.

Özcan et al. (2012) categorizes such methods as in the following: (i) *Purely Random* selection that consists of randomly selecting a low-level heuristic from among a set of existing heuristics (see also Özcan and Kheiri 2012), (ii) *Random Descent* selection that randomly selects a low-level-heuristic and applies it until the solution in hand is improved (see Kendall and Mohamad 2004 as an example

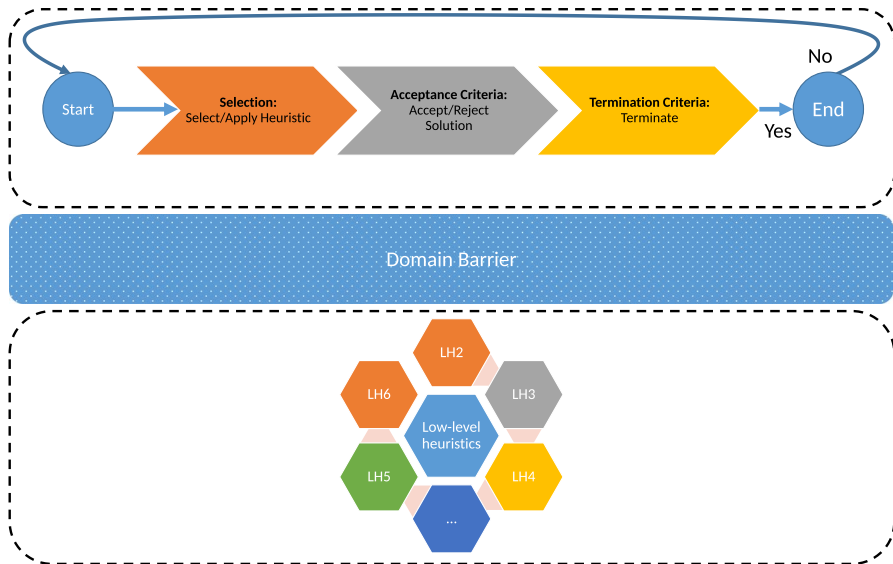


Fig. 2 The general scheme of a selection-type hyper-heuristic framework

of such a selection strategy). (iii) *Random Permutation Descent* selection that randomly selects a set of low-level-heuristics, and applies them in a circular manner. It then applies the first heuristic until a worsening move is hit. The next heuristics are applied on a cyclic manner (see Özcan and Kheiri 2012; Cowling et al. 2001; Burke and Soubeiga 2003). (iv) *Greedy* selection applies all low-level heuristics separately at each iteration and chooses the one that generates the best cost (see Cowling et al. 2002, 2001; Özcan and Kheiri 2012 as examples of this strategy). (v) *Peckish* selection follows the same strategy as the greedy selection, but a subset of low-level heuristics is considered instead of all low-level heuristics (see Cowling and Chakhlevitch 2003, 2007) and, (vi) *Reinforcement Learning* heuristic selection is based on a reward value for each heuristic. Such type of reward that represents its historical performance has been considered by Özcan et al. (2012), Drake et al. (2015), and Aron et al. (2015).

Our approach falls within the category of *reinforcement learning* methods and has been inspired from the concept of *association rules* in the *business data-mining* world (Aguinis et al. 2012). By the definition of Burke et al. (2010), our approach: (1) is a heuristic-selection type (as opposed to heuristic-generation type), (2) applies an online learning, and (3) is a hybrid construction–perturbation heuristic.

The overall flow of algorithm is described in the following: first, an initial solution is constructed using a constructive heuristic. Then, through a learning process, the hyper-heuristic proposes a series of perturbation, constructive, and/or improvement heuristics to improve the solution.

The first heuristic, which is a constructive one, optionally exploits the Lagrangian dual information to construct the initial (probably partial) solution. In the next step, such a solution can be partially destroyed and reconstructed,

perturbed-and-improved, just-improved, or any other combination. A potential improvement made by a transition from the first constructive heuristic to another heuristic is *evaluated*.

An acceptance criterion then decides on the transition from the current state (heuristic) to another one.

Here, we use a simulated annealing move acceptance scheme, where the decision regarding the transition (move) to be accepted depends on quality of the solution obtained from applying heuristics (scale of improvement) and the elapsed CPU time. All improved moves are always accepted, and the acceptance of worsening ones follows Metropolis criterion Kirkpatrick (1984):

$$\exp(-\Delta f/T) > R(0, 1), \quad (30)$$

where Δf is the difference between the post-transition and the current objective function values, T is a *synthetic temperature* and $R(0, 1)$ is a uniform random number between 0 and 1. T represents the global temperature. When T is larger, many non-improving or even deteriorating transitions (moves) can be accepted, while, when it tends to zero, $\exp(-\Delta f/T)$ become closer to zero, too. Therefore, for sufficiently small values of T , only improving moves are accepted.

The algorithm starts with solving Lagrangian relaxation LRX-CSA p HLRP using bundle algorithm (Frangioni 1995). The output (when stopping criteria of bundle are met) is in the form of a feasible solution with respect to the kept constraints (at least the p hubs are identified) as well as a set of coefficients of variables in the objective function. Such coefficients contain dual information.

3.2.1 Low-level heuristics

A rather large set of heuristics categorized in three classes, *constructive*, *improvement*, and *perturbation* are used within our hyper-heuristic framework. In the following, we describe every heuristic with sufficient details.

Constructive heuristics They build a solution from the scratch, based on several predefined rules. In this study, two constructive algorithms are implemented—CS1 and CS2.

- CS 1) That is outlined in Algorithm 1, performs as in the following: it selects p hubs that have the highest sum of supply and demand. Then, it allocates at least Γ nearest spokes to every hub (by considering capacity, e.g., based on the volume of demand/supply arriving/departing at/from the hub for the spoke on route). Finally, it incrementally constructs each feeder route by sequencing the spokes allocated to a given hub taking into consideration the capacity constraint.
- CS 2) Is distinguished from the first one, i.e., CS1, in the way, it selects the set of p hub candidates. CS2 uses the dual information of the Lagrangian relaxation (i.e., the coefficient of z_{ik} in LRX-CSA p HLRP) as well as some statistics collected about the frequency that a given node appeared as a hub during the iterations of bundle and the historical memory of hyper-heuristic process, if any. It allocates at least Γ spoke nodes to each hub node and at the end allocates remaining spokes. To construct the spoke-level route and sequencing the hubs and its allocated

spokes along such a route, we start from a hub and add the arcs incrementally. Let l be the last node already added to the partial spoke-level route and \mathcal{S}_L be the set of remaining spokes to put on the route, unless \mathcal{S}_L is a singleton, the next candidate spoke node, which we call it $R_Nearest$, to be added to the existing partial route is $i^* = \operatorname{argmin}_{i \in \mathcal{S}_L} \{\alpha_1 t_{s_l s_c} + \alpha_2 r_{s_l s_c}\}$, $0 \leq \alpha_1 + \alpha_2 \leq 1$, $\alpha_1, \alpha_2 \geq 0$. Here, $t_{s_l s_c}$ represents the travel time between the last spoke s_l in the cycle of the feeder route associate to the hub and the candidate spoke s_c . Similarly, $r_{s_l s_c}$ represents the travel time between the last spoke s_l in the cycle of the feeder route associate to the hub and the candidate spoke s_c , and $\tilde{r}_{s_l s_c}$ corresponds to the coefficient of $r_{s_l s_c}$ in LRX-CSA p HLRP.

Algorithm 1 CS1

```

1: procedure CONSTRUCTIVE1( $\mathcal{N}, p, \Gamma$ )
2:   Hubs = ChooseHubAccordingDemand()
3:   getFeeders(Hubs)
4:   for  $i = 1, \dots, p$  ( $p = |\mathcal{H}|$ ) do
5:      $j=0$ 
6:     while  $j < \Gamma$  do
7:        $f = R\_Nearest(\text{Hub}[i], \text{feeders}, \text{tabu})$ 
8:        $\text{temp TempTabu} \leftarrow f$ 
9:       if CapacityConstraint(AssignFeederToHub(Hub[i],  $f$ )) then
10:         AssignFeederToHub(Hub[i],  $f$ )
11:          $\text{Tabu} \leftarrow f$ 
12:          $j = j + 1$ 
13:       end if
14:     end while
15:      $\text{TempTabu} = \text{Tabu}$ 
16:   end for
17:    $\text{TempTabu} = \text{Tabu}$ 
18:   for  $i = 1, \dots, p$  ( $p = |\mathcal{H}|$ ) do
19:      $j=0$ 
20:     while true do
21:        $f = R\_Nearest(\text{Hub}[i], \text{feeders}, \text{tabu})$ 
22:       if  $f = \text{Null}$  then Break;
23:     end if
24:      $\text{TempTabu} \leftarrow f$ 
25:     if CapacityConstraint(AssignFeederToHub(Hub[i],  $f$ )) then
26:       AssignFeederToHub(Hub[i],  $f$ )
27:        $\text{Tabu} \leftarrow f$ 
28:        $j = j + 1$ 
29:     end if
30:   end while
31:    $\text{TempTabu} = \text{Tabu}$ 
32: end for return Hubs
33: end procedure

```

Improvement These set of heuristics start from a complete solution and apply some moves to improve the objective value. The following improvement heuristics are proposed:

- IMP 1) *Rotate* Given a route (a hub and a sequence of spokes along the route), this heuristic rotates (clock-wise or counter clock-wise) the string. Such rotation of size k will designate the k th spoke node after the current hub as the new hub and re-establish the current hub as a spoke. This heuristic is useful when the order of nodes on the route is rather correct, but the choice of hub is the cause of infeasibility (with respect to the capacity) or sub-optimality.
- IMP 2) *Re-order* It re-sequences the nodes (hubs and spokes) to find a better order of nodes along the route. While re-ordering, a sequence of $\{H, S_1, S_2, \dots, S_m\}$ may, for example, turn to $\{S_3, S_1, H, S_m, \dots, S_2\}$, wherein S_3 becomes the new hub and H turns to a spoke node. This heuristic takes into the capacity on the route and the total volume of demand and supply of nodes, such that, by some intelligent re-sequencings, either a feasible solution is obtained or an already feasible solution is improved.
- IMP 3) *Re-allocate* It removes an allocated spoke from a feeder network that has the maximum number of feeder nodes, and insert it into another feeder network with sufficient residual capacity. This node can be a hub or spoke in its original route and can be equivalently hub or spoke in the destination route. If it becomes a hub at the destination then the hub at the destination turns to spoke on the route. Normally, such nodes can be the one in an *irreducible subset* of the nodes on the route that are the cause of infeasibility or the nodes that are far from each other, but lie on the same feeder-level arc (imposing high transportation time).
- IMP 4) *SW1ffImp* It randomly swaps two allocated spoke nodes from two different feeder networks, if and only if the solution improves the incumbent. This heuristic can be seen as a kind of search that tries to diversify and sample from different parts of the search space.
- IMP 5) *SW2ffImp* It randomly swaps three spoke nodes belonging to three different feeder networks, if and only if the solution improves over the incumbent.

Perturbation These heuristics start from a complete solution and do some diversifications by injecting some noises. The following heuristics have shown to be effective:

- PRT 1) *SW1ff* It randomly swaps two spokes allocated to the two different hubs (see Algorithm 2). At the end, it either returns the feasible solution with the maximum Hamming distance (with respect to the binary design variables) from the input solution or one of the solutions among the p farthest ones.
- PRT 2) *SW2ff* It randomly swaps three spokes belonging to three different routes. It follows the same principle as in *SW1ff*, except that a very limited number ($n^2 < |\cdot| \ll n^3$) of samples are examined as a complete 3-opt is very expensive.
- PRT 3) *SW1hf* It swaps a random feeder and its corresponding hub. The hub becomes a spoke and the spoke one becomes a hub.
- PRT 4) *SW2hf* It is a fast local search that swaps a random feeder and a random hub, which should not be the hub to which it is allocated.

- PRT 5) *SWhh* It is a fast local search swapping two hubs. This can be done by taking into account the supply/demand of the entering hub that can violate the capacity (if we seek generating feasible solutions from perturbation).
- PRT 6) *Insert/delete* It removes an allocated spoke from a route, and insert it to another one either with respect to the residual capacity available on another route or even randomly.

Algorithm 2 SW1ff

```

procedure SW1FF(hub)
  tempHub = hub
  rdmHub = Random(p)
  rdmFeeder1 = Random(SizeOf(hub[rdmHub]))
  rdmFeeder2 = Random(SizeOf(hub[rdmHub]))
  while ( dordmFeeder1 == rdmFeeder2)
    rdmFeeder2 = Random(SizeOf(hub[rdmHub]))
  end while
  tempHub ← Swap(Feeder[rdmFeeder1],Feeder[rdmFeeder2])
  if CapacityConstraint(tempHub) then
    Return tempHub
  end ifreturn hub
end procedure

```

It must be emphasized that the *perturbation* heuristics aim at introducing some diversifications. Therefore, we do not expect a perturbed solution to be a feasible one. Rather, we hope that it can make a jump and help exploring unvisited regions in the solution space.

3.2.2 Learning algorithm

As mentioned earlier, several heuristic-selection methods are introduced in the literature. In this study, we work with a reinforcement learning mechanism to select the heuristics to be applied during the hyper-heuristic process.

In reinforcement learning, we are dealing with an *agent* that learns through some interaction with a so-called *dynamic environment*. Such interaction are usually realized through some trial-and-error efforts. At each point in time, t , an agent is in a given state $s_t \in S$ and can choose an action a_t —possibly from among several— to make a transition to another state $s_{t+1} \in S$. In this work, we deal with the sets S and A of bounded cardinality, i.e., $card(|A|) + card(|S|) < card(\mathbb{R}) = \aleph$. More precisely, we work with a finite number of transition that can be made.

When an action is taken, a numerical reward, r_{t+1} is attributed to this action. Such a reward is a feedback from the environment that depends on the quality of action taken by the agent. An agent that wants to maximize the total reward

attributed to it has to be able to exploit the experiences which it has obtained so far (memory) and also be able to explore other parts of the environment and identify better actions Sutton and Barto (1998). In general, there are two types of reinforcement learning:

- (a) *Online learning* heuristic rewards are updated online (i.e., dynamically) throughout the search according to the performance of every heuristic.
- (b) *Off-line learning* heuristic rewards are retrieved from a database, which stores the historical performance of every heuristic (often on similar class of problems) in the past.

Here, learning mechanism, which is presented in Algorithm 3, is inspired from a data-mining technique called Association Rules (AR) guided by a Tabu Search (TS). When looking at the principles of AR, one can observe that AR is actually a kind of reinforcement learning that helps in respecting the dependency order of heuristic applications. The AR learning mechanism is introduced by Agrawal et al. (1993). This is a technique in data mining that has attracted many attention. Extracting interesting correlation, frequent patterns, associations, or casual structures from group of items in the transaction databases or data repositories are the purpose of AR technique. Traditionally, AR has been widely used in Market Basket Analysis (Aguinis et al. 2012) to find how items purchased by customers are related.

In association analysis, there are two sets of items (called *itemsets*): (1) the *antecedent* (the *if* part), and (2) the *consequent* (the *then* part). Moreover, an association rule has two more values that express the degree of uncertainty about the rules. The first value is called the *support* for the rule, which is defined as the proportion of task-relevant data transactions for which the pattern is true. The second one is known as the *confidence* of the rule and is a measure of *certainty* or *trustworthiness* associated with each discovered pattern.

Here, our goal is to find some relationships between the different implemented heuristics, to find the best series of heuristics to be applied. The numerical reward is not going to be assigned to a single heuristic but to a series of them. Association Rule heuristic-selection method deals with two main variables: *support* and *confidence*. These variables are used to measure the performance of a series of heuristics. If a heuristic does not have the required support at a certain time, or if it does not have enough confidence, the Tabu Search meta-heuristic method prevents it from being selected.

The relevant mapping are explained in the following:

- (a) A heuristic series (H_m, \dots, H_n) weight (rewards) is equal to the negative summation (for minimization case) of the objective function value divided by the number of times this heuristic has been applied.
- (b) A support of an H_s is equal to the summation of the heuristic series (H_1, \dots, H_s) rewards in the precedent transactions divided by the summation of all the heuristic series rewards.

- (c) The confidence of a suggested H_s in such series (H_m, \dots, H_n) is equal to the support of H_s divided by the support of item set (H_m, \dots, H_n) .

Tabu list T_l : Given our time limit, we initially set $T_l = \emptyset$, and at each iteration, T_l is updated as follows: if the elapsed CPU time is less than a certain milestone (e.g., the time limit divided by a given number $V1$), heuristics are chosen randomly to establish a historical heuristic series performance. Otherwise, if the elapsed CPU time is greater or equal to the milestone $V1$, T_l will contain all heuristics for each series except those for which the support value is greater than a threshold th_s . In the case that the elapsed time is greater than the second milestone $V2 > V1$, T_l will contain all heuristics for each series except those for which the confidence value is greater than a threshold th_c . It must be noted that the tabu list is updated in accordance with the changes of quality of each heuristic series.

Algorithm 3 Choose Heuristic Method

```

procedure CHOOSEHEURISTIC(Elapsed_Time)
  if (Elapsed_time <  $V1$ ) then
    Return Random(NbOfHeuristic)
  end if
  if (Elapsed_Time >  $V1$  and Elapsed_Time <  $V2$ ) then
    while TRUE do
      Suggested_Heuristic_ID = Random(Nb_Of_Heuristic)
      S = Support(Suggested_Heuristic_ID)
      if ( $S > th_s$ ) then
        Return Suggested_Heuristic_ID
      end if
    end while
  end if
  if (Elapsed_Time >  $V2$ ) then
    while TRUE do
      Suggested_Heuristic_ID = Random(Nb_Of_Heuristic)
      C = Confidence(Suggested_Heuristic_ID)
      if ( $C > th_c$ ) then
        Return Suggested_Heuristic_ID
      end if
    end while
  end if
end procedure

```

3.3 Algorithm

Solving Lagrangian dual The corresponding Lagrangian dual problem is solved using the bundle method of Frangioni (1995).

Tree management and branching rules Our effort is put into quickly improving the lower bound, and accordingly, we process the nodes following a breadth-first scheme. Moreover, our extensive computational experiments reveal that the solution

method is rather robust to the branching decisions, and therefore, no particularly promising branching scheme is identified.

Cut management We add cuts mainly during the solution process to find the optimal solution to the MIP problem. Some cuts are added to almost all the nodes of the branch-and-bound tree, while others are added only at certain nodes. All such cuts are added if they meet a threshold of violation, ϵ . All the globally valid cuts are added as global cuts. This means that, once added, they remain in the LP until the end of the branch-and-bound process. No cut pool is maintained.

Separation of valid inequalities (20) If a node i is allocated to a hub node j , we can send one unit of flow from a dummy node p to the hub node where i is allocated to. If less than one flow arrives at that hub, a cut is identified.

Separation of valid inequalities (21) At any integer node where an inequality of form (21) is violated, every sub-path of length $|S| + 1$ represents such a valid inequality and is violated by 1. This separation can be done in $O(n^2)$.

Separation of valid inequalities (22) Initial computational experiments show that Padberg and Rao (1982) outperforms Fischetti et al. (1998) which is a heuristic mechanism.

Separation of valid inequalities (26) Separation based on block decomposition (as implemented for TSP in Concorde) is shown to be more efficient than that of Letchford and Lodi (2002)—even though a polynomial time.

4 Computational experiments

We have generated our instances based on the well-known Australian Post (AP) data set from the OR-Library. The transshipment times are generated randomly for some real values within $[2, 5]$ for the given time unit. The existing capacities of the original data (see Ernst and Krishnamoorthy 1999) are not used, because they do not always result in feasible solutions as the problem structures are different. We assume that our fleet of vehicles is homogenous and, therefore, has a unique capacity size. Let $O_i = \sum_j W_{ij}$ and $D_i = \sum_j W_{ji}$ and let p be the number of hubs. The capacity is generated as $C = \frac{\sum_i O_i + \sum_i D_i}{p}$.

All experiments were performed on an Intel 2.54 GHz core i5 CPU and 4 Gb RAM on a machine with Windows 7 as OS. All instances are named in a format ni_pj_k , where i indicates the number of nodes, j indicates the number of hubs, and k indicates the factor of economies of scale.

Our initial computational experiment suggested that valid inequalities (20) and (21) be separated only at integer nodes rather than fractional solutions. However, valid inequalities (22) and (26) can only be separated at fractional nodes and at every 100 nodes, if fractional.

The termination criteria are chosen in such a way to avoid non-promising computational efforts and premature convergence. The proposed termination criterion sets a global time limit $T_{limit} = \max \{6000, n^2 \times p \times 250\}$ m.s. for the overall computation. In addition, we set a second termination criterion for terminating the whole algorithm once $10 \times n \times p$ non-improving iterations have been observed.

In construction phase, α_1 and α_2 were set to 0.5. This means that LR r -coefficients and the travel time are equally important in influencing the choice of arcs to be established. $V1$ is set to $T_{limit}/3$, while $V2$ is set to $2T_{limit}/3$. th_S and th_C are set to a tolerance of 15% from the best objective function value.

When doing iterations of bundle algorithm, the subproblem was solved using CPLEX 12.7.1, the time limit was set to 3,600 seconds, and the number of iterations was set to 100. ϵ is set to $1e-6$.

While a fair comparison between a heuristic algorithm and an exact decomposition is rather non-trivial, in Table 4, we report the results of our computational experiments with hyper-heuristic (HH), with and without using results of LR, next to the results of Bender decomposition from Gelareh et al. (2015). There are, in total, 24 instances ranging from 10 nodes with 3 hubs to 20 nodes with 6 hubs. The first column reports the instance name and the second (resp. seventh) one indicates the computational times of Bender Decomposition method (resp. HH.). The best solution found by HH without LR (resp. Bender Decomposition) is reported in the fifth (resp. third) column. The fourth column reports the status at termination when applying Bender decomposition method. The gap between HH without LR results (resp. LR with HH represented by LR-HH) and Benders Decompositions are reported in the seventh (resp. ten) column. The last column indicates the gap between the best objective function value taken by HH with and without using LR results.

One observes that the relative gaps between the solution reported by Benders decomposition and those of HH never exceeded 0.08% if HH uses LR results; otherwise, it does not exceed 0.32%. In this table, the negative gaps indicate that such HH solutions are better than the feasible solution obtained by applying the Benders decomposition method in Gelareh et al. (2015). When compared to the computational times elapsed before obtaining such high-quality solutions, it is confirmed that the proposed HH framework is capable of obtaining high-quality solutions in very reasonable computational times, especially within LR results.

The gap between HH with and without LR results reaches 2.37% in instance n15_p3_0.8, which proves the additional value of using the LR variable coefficient into the process of HH. LR variable coefficient shown in the objective function appears very well during the process of HH. In construction phases (resp. in perturbation phase), all routes between any pairs have the coefficient greater than a fixed number such as ι (resp. ν) are excluded. Furthermore, during the improvement phases, pairs with small coefficients are favored to be connected.

Table 5 reports the total number of iterations every heuristic has worked on each instance. Rotate improvement heuristic has worked the most in all instances. Next, it comes to the constructive heuristics CS1 and CS2, which have more or less similar number of iterations for many instances. This may indicate that the information collected in the course of Lagrangian optimization in LR, the statistics collected during HH process, and the construction by taking into account demand (i.e., CS1) when choosing the hubs, have some similarities. Among the perturbation heuristics, SW2hf contributes in the most number of iterations.

Table 4 Quality comparison between benders decomposition and the proposed hyper-heuristic algorithms with and without LR

Instance	Bender Decomposition			Hyper-heuristic without LR			HH with LR			Gap (%) between HHs
	Ex. Time (s)	Obj. Val.	CplexStatus	Obj. Val.	HH Ex. Time (s)	Gap(%) with BD	Obj. Val.	HH Ex. Time (s)	Gap (%) with BD	
n10_p3_0.7	17	3235.99	Optimal	3235.99	4.075	0.00	3235.99	3.11	0.00	0.00
n10_p3_0.8	14	3315.81	Optimal	3315.81	6.127	0.00	3315.81	5.01	0.00	0.00
n10_p3_0.9	14	3395.63	Optimal	3395.63	6.064	0.00	3395.63	5.01	0.00	0.00
n15_p3_0.7	929	11,120.2	Optimal	10,990.48296	5.003	-1.18	10,990.48	4.10	-1.18	0.00
n15_p3_0.8	1395	11,255.74	Optimal	11,194.96	6.109	-0.54	10,929.41	6.10	-2.99	2.37
n15_p3_0.9	398	11,244.2	Optimal	10,421.16	6.058	-7.90	10,421.16	5.11	-7.90	0.00
n15_p4_0.7	1073	10,683.44	Optimal	8532.44	6.06	-25.21	8198.31	6.08	-30.31	3.92
n15_p4_0.8	883	10,170.06	Optimal	7966.52	5.076	-27.66	7966.52	4.10	-27.66	0.00
n15_p4_0.9	899	9067.25	Optimal	8949.29	6.089	-1.32	8949.29	6.11	-1.32	0.00
n15_p5_0.7	626	6899.27	Optimal	6921.75	6.068	0.32	6905.00	5.10	0.08	0.24
n15_p5_0.8	591	7143.31	Optimal	6921.75	5.064	-3.20	6921.75	4.19	-3.20	0.00
n15_p5_0.9	1615	7387.35	Optimal	7387.35	5.028	0.00	7387.35	5.00	0.00	0.00
n20_p3_0.7	15,307	24,814.09	Optimal	23,380.95	6.06	-6.13	23,380.95	5.09	-6.13	0.00
n20_p3_0.8	-	-	Failed	22,411.66225	6.044	-	22,411.66225	5.10	-	0.00
n20_p3_0.9	28,673	24,935.81	Optimal	23,316.54777	6.045	-6.94	23,316.54777	5.11	-6.94	0.00
n20_p4_0.7	-	-	Failed	19,382.47479	8.11	-	19,382.47479	7.99	-	0.00
n20_p4_0.8	14,432	21,284.45	Optimal	17,519.51325	7.077	-21.49	17,519.51325	7.00	-21.49	0.00
n20_p4_0.9	15,767	21,585.4	Optimal	21,585.4	8.129	0.00	21,585.40	7.01	0.00	0.00
n20_p5_0.7	6470	16,933.63	Optimal	16,933.63	9.26	0.00	16,933.63	8.21	0.00	0.00
n20_p5_0.8	19,572	17,624.7	Optimal	16,365.60	10.076	-7.69	16,256.34	8.07	-8.42	0.67
n20_p5_0.9	19,051	17,441.26	Optimal	17,441.26	10.146	0.00	17,441.26	8.24	0.00	0.00
n20_p6_0.7	27,531	15,738.56	Optimal	14,397.56	12.145	-9.31	14,397.56	11.13	-9.31	0.00
n20_p6_0.8	46,695	16,922.59	AbortUser	14,495.69	11.233	-16.74	14,440.86	9.06	-17.19	0.38

Table 4 (continued)

Instance	Bender Decomposition		Hyper-heuristic without LR			HH with LR		Gap (%) between HHs		
	Ex. Time (s)	Obj. Val.	CplexStatus	Obj. Val.	HH Ex. Time (s)	Gap(%) with BD	Obj. Val.		HH Ex. Time (s)	Gap (%) with BD
n20_p6_0.9	98,991	17,274.12	AbortUser	14,894.40	12.252	-15.98	14,662.78	11.19	-17.81	1.56

We have applied the proposed HH framework on a wider range of problem instance sizes. Table 7 reports the statistics of applying the proposed HH on a test-bed composed of instances of up to 100 nodes.

In Table 6, initial objective values of the HH with and without LR results are reported. In general, initial objective function using LR results is better than the results of this approach without LR. The average relative gap between the initial and the best found objective function value, for the instances of Table 6 without using LR results, is 21.45% (resp. 19.36% using LR results), while this becomes 26.66% for the instances with greater than 20 nodes (see Table 7). This is, on one hand, due to a much larger and spread solution space in larger instances, which, in turn, increases the probability of finding better solutions during the search and, on the other hand, may show that the HH is capable of benefiting from such a much larger search space to better improve the initial solutions. Another pessimistic view might question the performance of construction heuristics and quality of initial solution constructed by them.

To ensure the quality of solutions, we reserve only information about the allocation resulted from LR HH and ignore all spoke routes information. The problem has some similarity with the pick-up delivery vehicle routing problem with multiple depots. Each hub is considered as depot, and the objective is to construct a feeder network for each hub from its allocated spokes. We inject all allocation information into solver, CPLEX, which finds feeder networks. All constraints related to the number of hubs and allocations is ignored. This process, which uses some data resulted from a heuristic approach and completes the solution using a solver, is referred to as *Matheuristic*. Matheuristic optimization algorithms (Bartolini and Mingozzi 2009) are a combination between (meta-)heuristic approaches and other different techniques of mathematical programming.

The results of matheuristic are reported in Table 8 together with the relative gap between the solution of matheuristic and LR HH objective value, the execution time, and the Cplex status at termination. One observes that—within a time limit of 1 h—as the instance size increases, the LR HH is by far more efficient.

In Table 9, we report the results of applying iterative method for solving Lagrangian relaxation proposed in Sect. 3.1. The bundle method is known for its higher precision and faster convergence compared to the simpler versions of subgradient method. We have initialized the Lagrangian multipliers with the corresponding dual values of the LP relaxation, wherever possible (problem size).

In Table 9, we have reported the CPU time as well as the best bound obtained for the given amount of CPU time. LR was able to converge for the problem until size of 20 nodes in reasonable time for several instances.

5 Conclusion and future work

A hyper-heuristic approach is proposed for solving instances of a variant of Hub Location Routing Problems (CSA p HLRP). The problem has been inspired from an application in the distribution chain of pharmaceutical product. While exact methods have shown a very limited success when dealing with instances

Table 5 Statistic of applying different heuristics in solving every instances

Instance	Insert/delete	Re-allocate	Re-order	Rotate	SW1ffmp	SW2ffmp	SW2hf	SW1ff	SW2ff	CS1	CS2	SW/hh	SW/hf
n10_q3_0.7	3605	3568	3613	14,331	3550	3641	3621	3570	3491	3542	3542	3553	3572
n10_q3_0.8	3864	3899	3900	15,481	3861	3794	3898	3815	3846	3731	3769	3884	3843
n10_q3_0.9	3770	3895	3853	15,165	3856	3786	3897	3796	3706	3821	3751	3868	3838
n15_q3_0.7	2513	2520	2458	9969	2470	2533	2533	2461	2427	2490	2464	2481	2500
n15_q3_0.8	2471	2534	2437	9938	2505	2551	2497	2458	2421	2443	2460	2501	2439
n15_q3_0.9	2466	2515	2482	9876	2394	2561	2483	2467	2436	2428	2437	2427	2461
n15_q4_0.7	2571	2519	2532	10,129	2549	2621	2518	2469	2490	2494	2522	2550	2528
n15_q4_0.8	2381	2388	2348	9263	2348	2423	2412	2287	2252	2328	2334	2341	2296
n15_q4_0.9	2451	2509	2427	9824	2474	2529	2502	2391	2426	2483	2422	2454	2472
n15_q5_0.7	2561	2577	2562	10,079	2511	2518	2554	2463	2474	2511	2561	2591	2491
n15_q5_0.8	2468	2418	2495	9898	2485	2576	2495	2401	2459	2467	2460	2494	2427
n15_q5_0.9	2483	2479	2469	9917	2514	2579	2478	2407	2449	2405	2442	2505	2443
n20_q3_0.7	1055	1072	1021	4148	1056	1058	1108	1013	1021	1046	1058	1036	1011
n20_q3_0.8	1195	1127	1105	4582	1160	1235	1181	1087	1115	1177	1179	1198	1093
n20_q3_0.9	1164	1124	1112	4602	1181	1194	1161	1126	1133	1118	1152	1158	1122
n20_q4_0.7	1642	1603	1601	6323	1612	1690	1682	1532	1569	1642	1625	1610	1605
n20_q4_0.8	1584	1601	1515	6242	1516	1578	1612	1530	1553	1602	1542	1531	1547
n20_q4_0.9	1672	1614	1581	6444	1500	1624	1693	1552	1578	1608	1587	1637	1554
n20_q5_0.7	1973	1956	1931	7716	1923	2041	1933	1884	1854	1939	1936	1978	1956
n20_q5_0.8	1974	1951	1926	7638	1902	2034	1962	1861	1831	1926	1934	1944	1968
n20_q5_0.9	1928	1973	1995	7687	1889	1951	1914	1928	1882	1892	1942	1942	1881
n20_q6_0.7	2211	2226	2233	8769	2193	2253	2214	2161	2115	2139	2181	2241	2123
n20_q6_0.8	2187	2156	2152	8650	2155	2241	2225	2154	2104	2162	2164	2189	2136
n20_q6_0.9	2156	2159	2172	8707	2148	2196	2172	2137	2147	2184	2123	2157	2067

Table 6 Quality comparison between the initial objective value taken by the proposed hyper-heuristic algorithms with and without LR

Instance	HH initial Obj. Val.	LR HH initial Obj. Val	Gap
n10_q3_0.7	4113.93	4113.93	0.00
n10_q3_0.8	3732.68	3732.68	0.00
n10_q3_0.9	4281.89	4281.89	0.00
n15_q3_0.7	11,967.42	11,967.42	0.00
n15_q3_0.8	14,148.48	12,075.56	14.66
n15_q3_0.9	14,267.24	14,267.24	0.00
n15_q4_0.7	12,287.22	11,194.78	8.89
n15_q4_0.8	12,460.91	11,373.32	8.73
n15_q4_0.9	11,551.86	11,551.86	0.00
n15_q5_0.7	7942.10	7942.10	0.00
n15_q5_0.8	9341.05	8160.33	12.64
n15_q5_0.9	7583.26	7583.26	0.00
n20_q3_0.7	30,247.17	30,093.41	0.51
n20_q3_0.8	28,804.76	28,804.76	0.00
n20_q3_0.9	29,559.33	29,559.33	0.00
n20_q4_0.7	28,564.87	26,512.51	7.18
n20_q4_0.8	29,230.58	23,481.05	19.67
n20_q4_0.9	24,569.19	24,569.19	0.00
n20_q5_0.7	22,584.80	22,584.80	0.00
n20_q5_0.8	24,364.62	24,364.62	0.00
n20_q5_0.9	21,354.40	21,354.40	0.00
n20_q6_0.7	15,949.97	15,949.97	0.00
n20_q6_0.8	17,950.91	17,950.91	0.00
n20_q6_0.9	18,522.41	18,522.41	0.00

of problem and the instances of moderate size remain still very intractable, one has to resort to non-exact (heuristic-based) methods for obtaining solutions to the moderate-to-large size instances.

We proposed the first hyper-heuristic for this class of problems. The proposed hyper-heuristic consists of several low-level heuristics guided by means of a learning method, which is inspired from a topic in business data-mining world known as *Association Rules*. The hyper-heuristic benefits from the primal/dual information of a Lagrangian relaxation of the problem (relax-and-cut, more precisely). We have identified several classes of valid inequalities and the corresponding separation routines to separate such inequalities and use them within the relax-and-cut framework.

Our numerical results show that, in the case of instances for which an optimal solution or a solution with known quality is known, the proposed method is very competitive. In all such cases, the solutions are obtained in a very reasonable CPU time. Therefore, it is expected that for larger size instances for which finding an optimal solution (with general-purpose solvers or even some decompositions) needs an

Table 7 HH results for large instances of p -hub location problem

instance	HH initial Obj. Val.	LR HH initial Obj. Val.	HH execution time (s)
n30_p3_0.7	105,144.53	75,757.20	21.08
n30_p3_0.8	114,888.19	76,278.81	21.05
n30_p3_0.9	104,911.22	72,447.12	21.02
n30_p5_0.7	69,112.47	53,366.71	35.09
n30_p5_0.8	78,602.46	54,171.46	35.03
n30_p5_0.9	73,163.24	54,901.29	35.03
n40_p3_0.7	202,842.00	164,427.72	31.63
n40_p3_0.8	212,202.17	170,884.40	31.54
n40_p3_0.9	206,603.16	178,590.38	31.53
n40_p5_0.8	184,960.11	132,972.45	52.58
n40_p5_0.9	189,692.00	117,903.04	52.54
n40_p6_0.7	140,120.51	106,682.03	63.13
n50_p3_0.7	370,022.47	283,335.66	36.09
n50_p3_0.8	357,808.84	290,623.44	36.12
n50_p3_0.9	374,514.09	225,264.35	36.19
n50_p5_0.7	291,375.19	208,124.35	60.10
n50_p5_0.8	287,717.05	208,912.96	60.50
n50_p5_0.9	288,860.51	232,705.22	60.09
n70_p3_0.7	808,406.46	638,821.53	51.28
n70_p3_0.8	805,167.64	617,315.20	51.07
n70_p3_0.9	880,752.26	617,315.20	51.14
n70_p5_0.7	789,712.18	484,787.83	85.14
n70_p5_0.8	693,946.36	542,097.71	85.14
n70_p5_0.9	717,687.40	551,465.33	85.16
n80_p3_0.7	1,129,687.50	905,846.24	61.64
n80_p3_0.8	1,128,080.39	919,430.31	61.74
n80_p3_0.9	1,172,538.60	853,701.87	61.73
n80_p5_0.7	902,766.77	676,355.62	102.69
n80_p5_0.8	908,105.30	713,127.24	102.84
n80_p5_0.9	949,987.94	723,792.60	102.77
n90_p3_0.7	1,430,318.33	1,265,909.60	66.10
n90_p3_0.8	1,594,559.93	1,186,128.92	66.29
n90_p3_0.9	1,407,958.75	1,166,788.83	66.38
n90_p5_0.7	1,085,625.40	959,702.73	110.30
n90_p5_0.8	1,288,459.47	971,146.25	110.34
n90_p5_0.9	1,126,831.55	988,784.52	110.15
n100_p3_0.7	1,679,715.79	1,484,907.58	76.69
n100_p3_0.8	1,692,753.82	1,491,646.31	76.96
n100_p3_0.9	2,061,323.94	1,512,299.50	76.88
n100_p5_0.7	1,839,058.26	1,251,859.24	127.96
n100_p5_0.8	1,499,370.20	1,291,282.92	127.77
n100_p5_0.9	1,450,410.23	1,275,010.77	127.75

Table 7 (continued)

instance	HH initial Obj. Val.	LR HH initial Obj. Val.	HH execution time (s)
n100_p10_0.7	1,095,903.40	980,333.22	255.55
n100_p10_0.8	1,185,961.32	967,479.92	255.37
n100_p10_0.9	1,143,466.25	952,694.14	255.30

Table 8 A comparison between results of matheuristic with that of LR HH

Instance	Status	Matheuristic time (s)	Matheuristic OF	LR HH Obj. val.	GAP (%) LR HH and Math.
n10_q3_0.7	Optimal	0.3	3235.991	3235.99	0
n15_q3_0.7	Optimal	13.82	10,990.48	10,990.48	0
n15_q3_0.8	Optimal	40.3	10,902.53	10,929.41	0.25
n15_q3_0.9	Optimal	16.97	10,276.65	10,421.16	1.39
n15_q4_0.7	Optimal	1.76	8198.31	8198.31	0
n15_q4_0.8	Optimal	11.56	7966.52	7966.52	0
n15_q4_0.9	Optimal	2	8945.09	8949.29	0.05
n20_q3_0.7	Feasible	3605.57	23,213.17	23,380.95	0.72
n20_q3_0.8	Feasible	3605.7	22,411.66	22,411.66	0
n20_q3_0.9	Feasible	3605.73	23,316.55	23,316.55	0
n20_q4_0.7	Optimal	3675.79	19,382.47	19,382.47	0
n20_q4_0.8	Optimal	484.4	17,519.51	17,519.51	0
n20_q5_0.8	Optimal	7304.8	15,824.26	16,256.34	2.66
n20_q6_0.7	Optimal	12.45	14,397.56	14,397.56	0
n20_q6_0.8	Optimal	26.89	14,321.04	14,440.86	0.83
n20_q6_0.9	Optimal	33.88	14,662.78	14,662.78	0
n30_p3_0.7	Feasible	3672.64	80,871.04	75,757.2	- 6.75
n30_p3_0.8	Feasible	4118.88	81,845.67	76,278.82	- 7.3
n30_p3_0.9	Feasible	7756.06	78,998.28	73,284.34	- 7.8
n40_p3_0.7	Feasible	3770.47	292,132.1	164,427.72	- 77.67
n40_p3_0.8	Feasible	3787.6	352,933.8	170,884.4	- 106.53
n40_p3_0.9	Feasible	3796.32	316,753.5	178,590.38	- 77.36

effort beyond capacity of current hardware resources, the proposed framework that is equipped with advances learning mechanisms performs very well.

In our future work, we will add further aspects of real-life application to the model and consider solving problem taking into consideration a heterogeneous fleet of trucks/vessels with different route capacities, inter-route transshipment, etc.

Table 9 Lagrangian relaxation results

Instance	Time (s)	BestLB
n10 p3 0.9	228.177	2384.16
n10 p3 0.8	231.55	2319.98
n10 p3 0.7	219.116	2239.29
n15 p3 0.9	1665.9	5785.28
n15 p3 0.8	1704.89	5732.76
n15 p3 0.7	1744.7	5580.06
n15 p4 0.9	1901.39	5593.87
n15 p4 0.8	2020.9	5418.13
n15 p4 0.7	1937.78	5253.88
n15 p5 0.9	2164.69	5332.59
n15 p5 0.8	2246.72	5134.72
n15 p5 0.7	2185.9	4992.37
n20 p3 0.9	10,818	10,239.6
n20 p3 0.8	10,693	10,066.4
n20 p3 0.7	10,240	10,014.9
n20 p4 0.9	12,879	9933.42
n20 p4 0.8	13,407	9636.57
n20 p4 0.7	12,408	9426.4
n20 p5 0.9	22,194	9586.66
n20 p5 0.8	19,660	9389.53
n20 p5 0.7	15,495	9188.13
n20 p6 0.9	33,290	9366.45
n20 p6 0.8	23,062	8991.41
n20 p6 0.7	17,112	8652.79

Acknowledgements The BENMIP solver development has been supported by the PGM0, the Gaspard Monge Programme for Optimization and Operational Research, in the framework of the BENMIP project.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Abdinnour-Helm S (1998) A hybrid heuristic for the un-capacitated hub location problem. *Eur J Oper Res* 106(23):489–499
- Agrawal R, Imieliński T, Swami A (1993) Mining association rules between sets of items in large databases. In: *ACM SIGMOD record*, ACM, Vol 22, pp 207–216
- Aguinis H, Forcum LE, Joo H (2012) Using market basket analysis in management research. *J Manag* 39(7):1799–1824. <https://doi.org/10.1177/0149206312466147>
- Ahmed LN, Özcan E, Kheiri A (2015) Solving high school timetabling problems worldwide using selection hyper-heuristics. *Exp Syst Appl* 42(13):5463–5471
- Alumur S, Kara BY (2008) Network hub location problems: the state of the art. *Eur J Oper Res* 190(1):1–21

- Aron R, Chana I, Abraham A (2015) A hyper-heuristic approach for resource provisioning-based scheduling in grid environment. *J Supercomput* 71(4):1427–1450
- Bahiense L, Maculan N, Sagastizábal CA (2002) The volume algorithm revisited: relation with bundle methods. *Math Program* 94(1):41–69
- Barahona F, Anbil R (2000) The volume algorithm: producing primal solutions with a subgradient method. *Math Program* 87(3):385–399
- Bartolini E, Mingozzi A (2009) Algorithms for the non-bifurcated network design problem. *J Heuristics* 15(3):259–281
- Beyaz M, Dokeroglu T, Cosar A (2015) Robust hyper-heuristic algorithms for the offline oriented/non-oriented 2d bin packing problems. *Appl Soft Comput* 36:236–245
- Burke E, Kendall G, Newall J, Hart E, Ross P, Schulenburg S (2003) Hyper-heuristics: an emerging direction in modern search technology. In: *Handbook of meta-heuristics*, Springer, pp 457–474
- Burke E, Soubeiga E (2003) Scheduling nurses using a tabu-search hyper-heuristic. In: *Proceedings of the 1st multidisciplinary international conference on scheduling: theory and applications (MISTA 2003)*, Nottingham, pp 180–197
- Burke EK, Hyde M, Kendall G, Ochoa G, Özcan E, Qu R (2009) A survey of hyper-heuristics. *Computer Science Technical Report No. NOTTCS-TR-SUB-0906241418-2747*, School of Computer Science and Information Technology, University of Nottingham, Nottingham
- Burke EK, Hyde M, Kendall G, Ochoa G, Özcan E, Woodward JR (2010) A classification of hyper-heuristic approaches. In: *Handbook of meta-heuristics*. Springer, New York, pp 449–468
- Campbell A, Lowe T, Zhang L (2007) The p -hub center allocation problem. *Eur J Oper Res* 176(2):819–835
- Cetiner S, Sepil C, Sural H (2006) Hubbing and routing in postal delivery systems. Tech. rep., Industrial Engineering Department, Middle East Technical University, Ankara
- Correia I, Nickel S, Saldanha-da Gama F (2009) Single-allocation hub location problems with capacity choices. *Fraunhofer-Institut für Techno-und Wirtschaftsmathematik, Fraunhofer (ITWM)*
- Cowling P, Chakhlevitch K (2003) Hyper-heuristics for managing a large collection of low level heuristics to schedule personnel. In: *Evolutionary computation, 2003, CEC'03. The 2003 Congress on, IEEE, Vol 2*, pp 1214–1221
- Cowling P, Kendall G, Soubeiga E (2001) A hyper-heuristic approach to scheduling a sales summit. In: *Practice and theory of automated timetabling III*. Springer, New York, pp 176–190
- Cowling P, Kendall G, Soubeiga E (2002) Hyper-heuristics: a robust optimisation method applied to nurse scheduling. In: *parallel problem solving from nature-PPSN VII*. Springer, New York, pp 851–860
- Cowling PI, Chakhlevitch K (2007) Using a large set of low level heuristics in a hyper-heuristic approach to personnel scheduling. In: *Evolutionary scheduling*, Springer, New York, pp 543–576
- Danach K, Hassani JA-H, Khalil W, Gelareh S (2015a) Routing heterogeneous mobile hospital with different patients priorities: hyper-heuristic approach. In: *Digital information and communication technology and its applications (DICTAP), 2015 5th international conference on, IEEE*, pp 155–158
- Danach K, Khalil W, Gelareh S (2015b) Multiple strings planing problem in maritime service network: Hyper-heuristic approach. In: *Technological advances in electrical, electronics and computer engineering (TAECE), 2015 3rd international conference on, IEEE*, pp 85–88
- de Camargo RS, de Miranda G, Løkketangen A (2013) A new formulation and an exact approach for the many-to-many hub location-routing problem. *Appl Math Modell* 37(12):7465–7480
- Drake JH, Özcan E, Burke EK (2015) A modified choice function hyper-heuristic controlling unary and binary operators. In: *Proceedings of the IEEE congress on evolutionary computation (CEC 2015)*
- Ebery J, Krishnamoorthy M, Ernst A, Boland N (2000) The capacitated multiple allocation hub location problem: formulations and algorithms. *Eur J Oper Res* 120:614–631
- Ernst A, Krishnamoorthy M (1999) Solution algorithms for the capacitated single allocation hub location problem. *Ann Oper Res* 86:141–159
- Fischetti M, Gonzalez JJS, Toth P (1998) Solving the orienteering problem through branch-and-cut. *INFORMS J Comput* 10(2):133–148
- Fisher ML (1981) The lagrangian relaxation method for solving integer programming problems. *Manag Sci* 27(1):1–18
- Fisher ML (2004) The lagrangian relaxation method for solving integer programming problems. *Manag Sci* 50(12-supplement):1861–1871
- Frangioni A (1995) Solving semidefinite quadratic problems within nonsmooth optimization algorithms. Technical report

- Garrido P, Castro C (2012) A flexible and adaptive hyper-heuristic approach for (dynamic) capacitated vehicle routing problems. *Fund Inf* 119(1):29–60
- Garrido P, Riff MC (2010) Dvrp: a hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic. *J Heuristics* 16(6):795–834
- Ge D, Ye Y, Zhang J (2007) The fixed-hub single allocation problem: a geometric rounding approach. <http://www.stanford.edu/~yye/reviseHub.pdf>
- Gelareh S, Maculan N, Mahey P, Monemi RN (2013) Hub-and-spoke network design and fleet deployment for string planning of liner shipping. *Appl Math Modell* 37(5):3307–3321
- Gelareh S, Monemi RN, Semet F, (2015) Capacitated bounded cardinality hub routing problem: model and solution algorithm. Technical report [arXiv:1705.07985](https://arxiv.org/abs/1705.07985)
- Geoffrion A, Bride RN (1978) Lagrangean relaxation applied to capacitated facility location problems. *AIIE Trans* 10(1):40–47
- Geoffrion AM (1974) Lagrangian relaxation and its uses in integer programming. *Math Program Stud* 2:82–114
- Guignard M (2003) Lagrangian relaxation. *TOP* 11(2):151–228
- Held M, Karp R (1971) The traveling-salesman problem and minimum spanning trees: part II. *Math Program* 1(1):6–25
- Ilić A, Urošević D, Brimberg J, Mladenović N (2010) A general variable neighborhood search for solving the un-capacitated single allocation p-hub median problem. *Eur J Oper Res* 206(2):289–300
- Jarboui B, Derbel H, Hanafi S, Mladenović N (2013) Variable neighborhood search for location routing. *Comput Oper Res* 40(1):47–57
- Kendall G, Mohamad M, (2004) Channel assignment in cellular communication using a great deluge hyper-heuristic. In: *Networks, 2004 (ICON 2004)*. Proceedings of 12th IEEE international conference on, IEEE, Vol 2, pp 769–773
- Kirkpatrick S (1984) Optimization by simulated annealing: quantitative studies. *J Stat Phys* 34(5–6):975–986
- Klincewicz J (1992) Avoiding local optima in the p-hub location problem using tabu search and grasp. *Ann Oper Res* 40:283–302
- Kratka J, Stanimirović Z, Tošić D, Filipović V (2012) Genetic algorithm for solving un-capacitated multiple allocation hub location problem. *Comput Inf* 24(4):415–426
- Lemarechal C (1975) An extension of Davidon methods to non differentiable problems. *Nondifferentiable optimization*. Springer, New York, pp 95–109
- Letchford AN, Lodi A (2002) Integer programming and combinatorial optimization. In: *Proceedings of 9th International IPCO conference*, Ch. polynomial-time separation of simple comb inequalities, Springer, Berlin, May 27–29, 2002, pp 93–108
- Marshall RJ, Johnston M, Zhang M (2014) Developing a hyper-heuristic using grammatical evolution and the capacitated vehicle routing problem. In: *Simulated evolution and learning*, Springer, New York, pp 668–679
- Marshall RJ, Johnston M, Zhang M (2015) Hyper-heuristic operator selection and acceptance criteria. In: *Evolutionary computation in combinatorial optimization*, Springer, New York, pp 99–113
- Monemi RN, Danach K, Khalil W, Gelareh S, Lima FC, Aloise DJ (2015) Solution methods for scheduling of heterogeneous parallel machines applied to the workover rig problem. *Exp Syst Appl* 42(9):4493–4505
- Nagy G, Salhi S (1998) The many-to-many location-routing problem. *Top* 6(2):261–275
- Özcan E, Kheiri A (2012) A hyper-heuristic based on random gradient, greedy and dominance. In: *Computer and information sciences II*. Springer, New York, pp 557–563
- Özcan E, Mısırlı M, Ochoa G, Burke EK (2012) A reinforcement learning: great-deluge hyper-heuristic. *Modeling, analysis, and applications in meta-heuristic computing: advancements and trends*, p 34
- Padberg MW, Rao MR (1982) Odd minimum cut-sets and b-matchings. *Math Oper Res* 7(1):67–80
- Peiró J, Corberán Á, Martí R (2014) Grasp for the un-capacitated r -allocation p -hub median problem. *Comput Oper Res* 43:50–60
- Rabbani M, Kazemi S (2015) Solving un-capacitated multiple allocation p-hub center problem by dijkstra's algorithm-based genetic algorithm and simulated annealing. *Int J Ind Eng Comput* 6(3):405–418
- Randall M (2008) Solution approaches for the capacitated single allocation hub location problem using ant colony optimisation. *Comput Optim Appl* 39(2):239–261
- Rodríguez-Martín I, Salazar-González J-J, Yaman H (2014) A branch-and-cut algorithm for the hub location and routing problem. *Comput Oper Res* 50:161–174

- Rolland E, Schilling DA, Current JR (1997) An efficient tabu search procedure for the p -median problem. *Eur J Oper Res* 96(2):329–342
- Ross P, Schulenburg S, Marin-Blázquez JG, Hart E (2002) Hyper-heuristics: learning to combine simple heuristics in bin-packing problems. In: GECCO, pp 942–948
- Sim K, Hart E, Paechter B (2015) A lifelong learning hyper-heuristic method for bin packing. *Evolut Comput* 23(1):37–67
- Stanimirovic Z (2008) An efficient genetic algorithm for the un-capacitated multiple allocation p -hub median problem. *Control Cybern* 37:669–692
- Sutton RS, Barto AG (1998) Reinforcement learning: an introduction, vol 1. MIT Press, Cambridge
- Ting C-J, Chen C-H (2013) A multiple ant colony optimization algorithm for the capacitated location routing problem. *Int J Product Econ* 141(1):34–44
- Tuzun D, Burke LI (1999) A two-phase tabu search approach to the location routing problem. *Eur J Oper Res* 116(1):87–99
- Wasner M, Zäpfel G (2004) An integrated multi-depot hub-location vehicle routing model for network planning of parcel service. *Int J Product Econ* 90(3):403–419
- Yu VF, Shih-Wei L, Wenyih L, Ching-Jung T (2010) A simulated annealing heuristic for the capacitated location routing problem. *Comput Ind Eng* 58(2):288–299
- Zarandi MF, Davari S, Sisakht SH (2015) An empirical comparison of simulated annealing and iterated local search for the hierarchical single allocation hub median location problem. *Sci Iran Trans E Ind Eng* 22(3):1203
- Zheng Y-J, Zhang M-X, Ling H-F, Chen S-Y (2015) Emergency railway transportation planning using a hyper-heuristic approach. *Intell Transp Syst IEEE Trans* 16(1):321–329

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.