Heuristics for a Hub Location-Routing Problem

Mauro Cardoso Lopes
Institute of Computing, University of Campinas, Avenida Albert Einstein, 1251, Campinas-SP, 13083-852, Brazil

Carlos Eduardo de Andrade
Advanced Technology Department, AT&T Labs Research, 200 South Laurel Avenue, Middletown, New Jersey 07748

Thiago Alves de Queiroz
Institute of Mathematics and Technology, Federal University of Goiás, Avenida Dr. Lamartine Pinto de Avelar 1120, Catalão-GO, 75704-020, Brazil

Mauricio G. C. Resende
Mathematical Optimization and Planning Group, Amazon.com, Inc., 333 Boren Avenue North, Seattle, Washington 98109

Flávio Keidi Miyazawa
Institute of Computing, University of Campinas, Avenida Albert Einstein, 1251, Campinas-SP, 13083-852, Brazil

We investigate a variant of the many-to-many hub location-routing problem which consists in partitioning the set of nodes of a graph into routes containing exactly one hub each, and determining an extra route interconnecting all hubs. A variable neighborhood descent with neighborhood structures based on remove/add, swap and exchange moves nested with routing and location operations is used as a local search procedure in a multistart algorithm. We also consider a sequential version of this local search in the multistart. In addition, a biased random-key genetic algorithm working with a local search routine, which also considers routing and location operations, is applied to the problem. To compare the heuristic solutions, we develop an integer programming formulation which is solved with a branch-and-cut algorithm. Capacity and path elimination constraints are added in a cutting plane fashion. The separation algorithms are based on the computation of min-cut trees and on the connected components of a support graph. Computational experiments were conducted on several benchmark instances of routing problems and show that the heuristics are effective on medium to large-sized instances, while the branch-and-cut algorithm solves small to medium sized problems to optimality. These algorithms were also compared with a commercial hybrid solver showing that the heuristics are quite competitive.
 NETWORKS, Vol. 68(1), 54–90 2016

Received October 2014; accepted May 2016
Correspondence to: C. E. de Andrade; e-mail: cea@research.att.com
DOI 10.1002/net.21685
Published online 29 June 2016 in Wiley Online Library (wileyonlinelibrary.com).

Keywords: Hub location-routing problem; heuristics; variable neighborhood descent; biased random-key genetic algorithm; integer formulation

1. INTRODUCTION

In freight transportation or telecommunications, hub nodes are special facilities that serve as connections between origins and destinations, these also called non-hub nodes, in which goods or electronic data from origins are consolidated in hubs and switched to their respective destinations. When a destination is served with goods of different origins (many-to-one case), or an origin has to serve different destinations (one-to-many case), or many origins have to serve many destinations (many-to-many case), hub networks offer some benefits. In these situations, routes from origins to hubs are used to transport goods to be consolidated and switched by interhub edges (hub-to-hub connections) to final hubs, and finally routed from these final hubs to the destinations they serve. Figure 1 depicts the case in which goods from the origins n_1, n_2, n_9, and h_6 are routed (a blue arrow indicates the flow between nodes) to their respective hubs (except for h_6, which is itself a hub) and then, using an interhub route, they are transported to hub h_5, which serves the destination n_4.

Recently, [14] presented a survey on the advances obtained for hub location problems with a detailed description of the characteristics and features that have been considered for them. Moreover, the author emphasizes how important are these problems when dealing with the design of transportation systems and telecommunication networks. In this article,

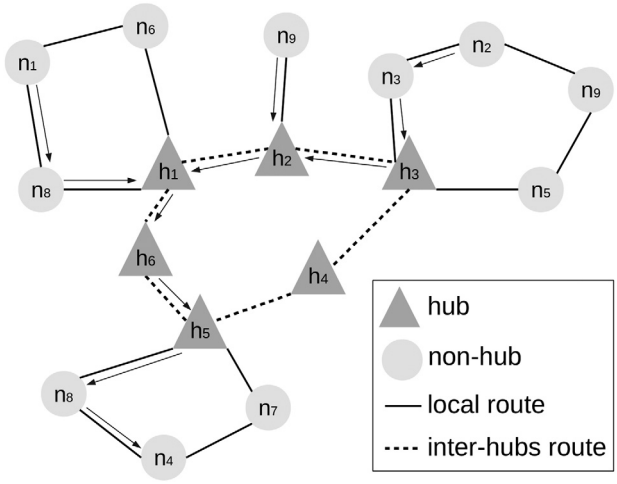


FIG. 1. Example of a many-to-one case in a ring hub network.

we investigate the many-to-many case of a hub location problem with the following assumptions:

- any node is a candidate to be a hub in the network;
- the number of required hub nodes is initially given;
- each non-hub node is served by exactly one hub and must be in only one local route, and therefore, pickup and delivery happen at the same time;
- for each hub there exists at most one local route starting and ending at the hub that serves one or more non-hub nodes;
- hubs are unlimited in capacity, but any local route is limited in length by a given maximum value;
- there is no hub installation cost;
- the hubs must be in an interhub route forming a ring of hubs;
- a discount factor $0 \leq \alpha \leq 1$ is applied when using the edges of the interhub route (e.g., when there is a bulk transportation [57]).

The objective is to minimize the total cost given by the interhub transportation route and the local routes connecting non-hub nodes to hub nodes.

On the perspective of location-routing problems [45], we have a many-to-many location-routing problem that seeks a subset of p nodes (the hub nodes) organized in an interhub route such that from each hub, there is at most one local route containing non-hub nodes that it serves. The objective is to minimize the total cost incurred by all routes. We address a variant of the many-to-many hub location-routing problem (MMLRP) that was proposed by Nagy and Salhi [55] and which has many applications, for example, freight industry, transportation of parcels, including passengers, mail, and beverages [37, 41], as well as in telecommunications [72]. Conversely, if there is no interhub route, the problem reduces to a variant of the plant-cycle location problem [38] in which any node is a candidate to be a plant. This problem arises in the design of Global System for Mobile (GSM) networks.

We call the problem addressed here the *Many-to-Many p-Location-Hamiltonian Cycle Problem* (MMpLHP), due to the fact that all nodes are covered by a set of exactly p routes (some of which may be empty) and the limitation that exactly one Hamiltonian cycle connects the hub nodes. Note that MMpLHP is $\mathcal{NP}$-hard since it contains two $\mathcal{NP}$-hard subproblems, namely the facility location problem and the traveling salesman problem [22].

Applications of the MMpLHP arise in transportation systems, particularly in public and urban transportation of passengers. Subways and trains use the interhub route, so each hub is a platform that consolidates passengers from many origins and allows their delivery to many destinations. Additionally, some of these platforms are served by bus lines (local routes) allowing passengers to reach their final destinations. Observe that some platforms may not be served by bus lines depending on whether they are located in the city center or a suburban region. Besides, bus lines may contain one or more stops, although it is commonly more than one. This problem can also be found in the design of distributed data networks where hubs are related, for instance, to switches, concentrators, or routers.

1.1. Literature Review of the MMLRP

Introduced by Nagy and Salhi [55], the MMLRP aims to minimize the total cost incurred by all routes, while locating hubs and assigning non-hub nodes to hubs subject to the following constraints:

- all hubs must be linked, that is, there is an edge between any pair of hubs—complete hub network;
- pickup and delivery may not happen at the same time, so a non-hub node may be visited more than once;
- vehicles are capacitated and routes are limited by a travel time limit;
- hubs can be in more than one route;
- there is no discount factor α.

The MMLRP was tackled by Nagy and Salhi [55] with a nested heuristic that solves a location problem using an add/drop/shift method enhanced with a tabu search routine and determine routes from these hubs by solving a multidepot vehicle routing problem heuristically. They also presented an integer programming formulation and discussed how it can be applied to other related problems. However, no computational results are presented. Instead, the authors limited themselves to solving a single illustrative example with their heuristic.

The taxonomy of location-routing problems, presented in [45], only considers four papers on the MMLRP, including the preliminary study of [55]. Bruns *et al.* [10] considered the MMLRP without solving the routing problem explicitly. Instead, they proposed an approximative approach that reduces the original problem to the plant location problem. This strategy was used by those authors to provide solutions for the Swiss public postal service under different scenarios. Wasner and Zaäpfel [73] considered a real case study of the Austrian postal delivery service modeled by a nonlinear formulation which was solved using a parallel strategy with different local search routines combined with feedback loops. Their strategy first determines the number and location of the hubs and assigns postal zones to hubs so that hub

costs can be obtained. Next, pickup and delivery routes are determined for the hubs. Finally, the total solution cost is computed. Feedback loops are used to decrease the solution cost by reassigning postal zones to hubs, combining routes, and changing the number and locations of hubs. Also, a central hub is considered through which all traffic between hubs must be routed, and pickup and delivery occur simultaneously. Karaoglan *et al*. [36] considered the location-routing problem with simultaneous pickup and delivery constraints in which hub-to-hub routes are not allowed. In Çetiner et al. [12], the MMLRP was solved with a two-stage procedure. Hub locations are determined as well as allocation of non-hub nodes to hubs and then routes are computed accordingly. The procedure alternates between the hub update stage and the route update stage to reduce the overall costs. Experiments with the Turkish postal delivery system and instances from the literature showed the competitiveness of their algorithm. Camargo *et al*. [11] proposed a large-scale integer programming formulation for the MMLRP which combines models of the single allocation hub location and traveling salesman problems. Their formulation uses four-index integer variables combined with five-index continuous variables, but once integer variables are fixed, two simple subproblems result. This characteristic allowed Benders decomposition to be applied so that instances with up to 100 nodes were solved, although Camargo *et al*. [11] reported results for only one instance with 100 nodes, which required approximately 46 h of processing to solve. Moreover, the results focused on instances with up to 50 nodes and for different values of α were solved in at most 2h of computing time. They concluded that as the value of α increases, the number of hubs decreases, and solution time increases.

A branch-and-cut algorithm was proposed by Rodríguez-Martín *et al*. [65] for a variant of the MMLRP in which there is a fixed number of hubs to locate, at most one local route per hub, and each of these routes is constrained to have at most a given number of non-hub nodes. Their integer programming model considers five types of variables related to: hub location; hubs without local routes; local routes with one, two, or more non-hub nodes; flow between nodes; and routes with three or more edges. Families of valid inequalities and separation algorithms for connectivity, subtour elimination, and capacity constraints were proposed so that their branch-and-cut algorithm was capable of solving to optimality instances with up to 50 nodes within a time limit of two hours. Lopes *et al*. [44] addressed the MMpLHP, for which they developed a local search heuristic (exploring three different neighborhoods) and an integer programming model.

Note that it is very costly to link directly all pairs of hubs. Therefore, a complete hub network may be inefficient for this type of problem. With this in mind, Labbé and Yaman [39] considered a star hub network with applications in telecommunications. They proposed two integer programming models and performed a polyhedral analysis indicating that some constraints are facet inequalities. Furthermore, they derived a heuristic based on Lagrangian relaxation which successfully computed optimal solutions for 104 instances. Yaman and Elloumi [75] also considered a star hub network, while Contreras et al. [15] investigated a tree structure. Rieck *et al*. [64] investigated a variant found in the timber-trade industry that considers a general hub network where hubs are limited to a set of potential locations, in which pickup and delivery are not necessarily simultaneous, and where non-hub nodes are categorized as supply and delivery points. Their mixed-integer programming formulation, strengthened with general and instance dependent inequalities, solved instances to optimality whose total number of nodes were at most 17. To tackle larger instances, two heuristics, a fix-and-optimize procedure, which uses the formulation, and a genetic algorithm were proposed and compared.

For other variants of location-routing problems, the reader can follow the recent classification performed in [45] and also the surveys by [16, 49] and [56]. Andrade *et al*. [1] introduced a variant called the k-Interconnected multidepot multitraveling salesmen problem, with applications in telecommunications. They developed a biased random-key genetic algorithm using local search procedures that combine swap moves and subroute optimization. Rahmani *et al*. [59] proposed local search methods to the two-echelon variant with pickup and delivery encountered in the distribution of shoes.

Similarly, variants of hub location problems were studied in: [31], which considered intermodal hubs (i.e., fully interconnected facilities) and presented a heuristic combining branch-and-bound, Lagrangian relaxation, and linear programming relaxation; [60], which proposed an integer formulation for the variant with multilevel capacities when locating hubs and allocating non-hubs; [63], which considered inventory services at hubs and congestion on interhub route, for which the authors proposed a mixed integer nonlinear mathematical model; [20], which investigated the influence of the set-up cost of hubs by a series of numerical tests and real world considerations; and [23], which proposed a mixed integer nonlinear programming model for a variant with application in third party logistics.

1.2. Literature Review of Variable Neighborhood Search

The Variable Neighborhood Search (VNS) was proposed in [53] as a new systematized search heuristic. VNS explores different neighborhood structures with a local search procedure to obtain a global solution for all explored neighborhoods. In general, an initial random solution is obtained from a neighborhood structure (*shake* phase) and then possible improved with a local search routine. The local search is commonly implemented using a best improvement method. A survey of VNS can be found in [30].

Algorithm 1 briefly describes a basic VNS. Let x be a solution and $\mathcal{F}(x)$ be its value, and let $\mathcal{N} = \{N_1, \ldots, N_\kappa\}$ be a set of κ different neighborhoods. The algorithm starts with a random solution. In lines 1.4–1.11, the algorithm first computes a random solution x' that is next used as input on the local search phase to get a possible improved solution x''. If x'' is better than the current best solution x, the

Algorithm 1 Basic VNS scheme

```
1.1  Let x be an initial random solution.
1.2  repeat
1.3      imp ← 1.
1.4      while imp ≤ κ do
1.5          x' ← shake(x, N_imp).
1.6          x'' ← localSearch(x').
1.7          if F(x'') < F(x) then
1.8              x ← x''.
1.9              c ← 1.
1.10         else
1.11             imp ← imp + 1.
1.12 until a stopping criterion is not reached
1.13 return solution x.
```

algorithm continues with the current c neighborhood after update x to x'', otherwise, the algorithm considers the next neighborhood structure. The stopping criterion may be maximum time, number of iterations, or some other quality or constraint measurement.

A simple variant of VNS is the Variable Neighborhood Descent (VND) [53], which can be summarized by the following steps given an initial solution x and $imp \leftarrow 1$: (i) find the best solution x' in $N_{\text{imp}}(x)$; (ii) update x by x' if x' is better than it and then go to step (i) with imp $\leftarrow$ 1, otherwise consider the next neighborhood (imp $\leftarrow$ imp + 1) and repeat the steps (i) and (ii) if $imp \leq \kappa$. It is very common to use VND as a local search method, as it can get a solution that traversed all neighborhoods, instead one limited to a single (or even few) neighborhood structures.

VNS-based algorithms were used in some papers that addressed hub networks. Ilić *et al*. [33] solved a p-hub median problem, which was first introduced by [57], while a survey of the p-median problem can be found in [61]. Ilić *et al*. [33] proposed two general VNS variants differing in how the VND was used in the local search. Their VND was composed of three neighborhood structures based on allocating non-hub nodes, alternating hubs, and locating new hubs. The first VND consists in a sequential application of the neighborhoods, while the second one is a nested version. Computational experiments over large instances illustrated the good behavior of these two algorithms.

Applications in location, routing, or location-routing problems using VNS were developed by several authors. Mladenović *et al*. [51] solved a variant of the traveling salesman problem (TSP) using a two-level VNS, in which a VNS is used inside another one, allowing the quick computation of feasible solutions. Their approach outperformed a branch-and-cut method as well as a tabu search algorithm. Another variant of the TSP, now with one-commodity pickup-and-delivery constraints, was solved with a VNS in [52]. Salhi *et al*. [66] considered a VNS-based algorithm to solve the multidepot routing problem with heterogeneous vehicles. Their approach combined VNS with a sweep method, a 2-opt procedure, Dijkstra's algorithm, and a diversification strategy. The local search in this VNS has six operators consisting of insertion, swap, and 2-opt moves. Salhi *et al*. [66] improved the best known solution of 23 out of 26 instances from the literature. Pacheco *et al*. [58] addressed a vehicle routing problem that emerged in a Spanish bakery. Their VNS algorithm uses memory in the shaking step and may accept a bad solution.

Mladenović et al. [50] addressed the p-center problem using a tabu search and a VNS, both working on a node substitution neighborhood structure. The authors related that the VNS algorithm was better than the tabu search in face of the computational experiments. A hybridization of a VNS with a tabu search for the vehicle routing problem with multiple time windows was proposed in Belhaiza *et al*. [7]. Their algorithm accepts infeasible solutions that violate time windows, capacity, or maximum duration constraints, but penalize such infeasibilities. The local search routine considers two phases of improvement, one in single-routes and the other in multiroutes. The tabu list contains a list of solution structures recently explored. Jarboui *et al*. [35] considered themselves the first ones to present VNS algorithms for the location-routing problem, although we found a VNS based algorithm for the two-echelon variant, previously proposed by Schwengerer *et al*. [68], which generalizes the location-routing problem. Jarboui *et al*. [35] considered five neighborhood structures, four of which were designed to deal with routes, and one to deal with location and routing decisions simultaneously. Such structures are based on insertion, removal, or swap moves applied on the same route or two different routes. The algorithm accepts solutions that exceed the depot capacity, but penalizes them.

1.3. Biased Random-Key Genetic Algorithms

To search for good solutions for the MMpLHP, we implemented a biased random-key genetic algorithm (BRKGA) [28]. Our choice was mainly grounded on recent successes with classical hard combinatorial optimization problems such as routing [1] covering [62], packing [29] scheduling [26], and clustering [2].

BRKGA was introduced by Gonçalves and Almeida [19, 27] and has two key features that distinguish it from traditional genetic algorithms [24]: (i) A standardized chromosome encoding that uses a vector with t uniformly drawn random keys (*alleles*) over the interval $[0, 1]$ [5]; (ii) A well-defined evolutionary process which uses parameterized uniform crossover [70] and substitutes the application of the mutation operator on existing chromosomes with newly introduced *mutants*—defined as t-long vectors of (uniformly drawn) random keys—for exploration.

The most interesting and important characteristic of BRKGA is the crossover between individuals. As opposed to traditional genetic algorithms, BRKGA performs crossover without caring about the feasibility of the solutions generated for the new individuals. This is possible because of the standard chromosome encoding that is responsible for

guaranteeing the feasibility of the decoding function. The decoding function or *decoder* $d : [0,1]^t \rightarrow \mathcal{S}$ maps the real vector with a valid solution (chromosome of size t) in the solution space $\mathcal{S}$. In some problems where feasibility is hard to achieve, the decoder may generate invalid solutions, making use of a penalty factor.

Algorithm 2 summarizes a typical BRKGA framework. Basically, we generate *pop* chromosomes as initial individuals[1] using vectors with t uniformly drawn random keys over the interval $[0, 1]$. In each iteration, a problem-specific *decoder* extracts the fitness of the chromosomes. To build a new generation, we copy the pop_e best individuals (the *elite* set), add pop_μ random chromosomes (the *mutants*) and generate, by applying the crossover operator, $\text{pop} - \text{pop}_e - \text{pop}_\mu$ offspring. Crossover is done between a random individual from the elite set and an individual from the remainder of the population: an offspring is generated by *mating*, where we take each allele from the elite parent with probability ρ_e or from the other parent with probability $1 - \rho_e$. With $\rho_e = 0.5$, the standard uniform crossover occurs. With $\rho_e > 0.5$ by definition, intensification happens at two levels: when parents are selected, because one is drawn from the elite set, and when offspring are conceived, because their alleles are inherited from the elite parent with greater probability. Diversification happens with the introduction of mutants at each generation, as they are vectors with t uniformly drawn random keys. The usual mutation operators on individual genes are not employed by BRKGA. Observe that the above scheme prevents infeasibility since, by definition, the resulting chromosomes—both offspring and mutants—are always vectors of random keys over $[0, 1]$. One characteristic of BRKGAs is the biased selection of keys in the crossover process which enables the offspring to inherit characteristics of the elite parent with controlled probability. In [28], several experiments corroborate with this observation.

A common approach to genetic algorithms is the island model [74], where several populations are evolved independently and exchange their best individuals every given number of generations. This improves the variability of individuals, usually speeding up convergence, and reduces the risk that the algorithm will get stuck in local optima. Note that it is not necessary that this process be done in parallel in the sense of using several parallel machines or CPUs. It is straightforward to adapt the BRKGA framework: π separate populations are created such that they are evolved simultaneously applying the evolutionary process in lines 3–8 of Algorithm 2 to each population. In this case, we will have $\text{Pop}_1, \ldots, \text{Pop}_\pi$ populations, $\text{Eli}_1, \ldots, \text{Eli}_\pi$ elite sets, and $Q_1, \ldots, Q_\pi$ "next generation" pools. The individual exchanges occur when a given threshold is reached, for instance, at every δ generations. For each population Pop_i, the η best individuals are copied from other populations $\text{Pop}_{j \neq i}$ and replace the $\eta(\pi - 1)$ worst individuals in Pop_i.

In conclusion, the parameters that must be specified beforehand are the size of the chromosomes t, the size of the population pop, the size of elite set pop_e, the number of mutants pop_μ introduced at each generation, and the inheritance probability ρ_e. If using parallel populations, we must set the number of populations π and the generation threshold δ to exchange the η best individuals. Advice for the parameter setup can be found in [28].

[1] The pair formed by a chromosome and its fitness is called individual.

Algorithm 2 BRKGA scheme

```
2.1 Generate the initial population Pop.
2.2 while a stopping criterion is not reached do
2.3     Decode each chromosome of Pop and extract
        its solution and fitness.
2.4     Sort the population Pop in non-increasing order
        of fitness. Consider the top pop_e individuals as
        the elite group Eli.
2.5     Copy Eli to the next generation Q, unaltered.
2.6     Add pop_μ randomly-generated new
        chromosomes (mutants) to Q.
2.7     Generate pop−pop_e−pop_μ chromosomes
        (offspring) by parameterized crossover,
        selecting a random parent from Eli and
        another from Pop \ Eli. Add them to Q.
2.8     Pop ← Q.
2.9 return best individual found.
```

1.4. Our Contribution

One can note that the literature of heuristic methods for the many-to-many location-routing problem is sparse. Most of the research was done using integer programming models. To the best of our knowledge, there are neither VND nor BRKGA algorithms to solve the MMpLHP specifically.

In this article, we propose three heuristics to solve the MMpLHP. The first heuristic is a BRKGA that implements a local search based on the Lin-Kernighan heuristic for the TSP. The second is a multistart algorithm that uses a VND as local search, which has neighborhoods defined with remove/add, swap, and exchange moves. We also propose a simple local search with multistart that comes as extension of the previous heuristic. Our choice was mainly grounded on recent successes of those heuristics and variants on classical hard combinatorial optimization. To validate our results, we propose an exact algorithm based on integer programming and valid inequalities that solves small and medium size instances. We tested several instances of TSPLIB, using 12 different scenarios. Our results are also compared with a commercial hybrid solver, namely LocalSolver [34].

In the following sections, we present algorithms and computational experiments to solve the MMpLHP. Section 3 describes the proposed BRKGA and Section 4 the multistart with VND and pure local search. Section 5 presents an integer programming formulation and separation algorithms to detect constraint violations. Computational experiments

with instances adapted from the literature are reported in Section 6, while conclusions and directions for future work are given in Section 7.

2. DEFINITIONS

Let $G = (V, A)$ be an undirected loopless complete simple graph with node set V and edge set A, where each $e \in A$ has a weight or cost $c_e \in \mathbb{R}^+$ (note that the edge $\{i,j\} \in A$ if and only if $i \neq j$). Without loss of generality, consider $V = \{1, 2, \ldots, n\}$, where $n = |V|$. Consider also $V[B]$ and $E[B]$, respectively, as subsets of nodes and edges induced by structure B. A route $\mathcal{C}$ of size t is a subgraph of G such that nodes $V[\mathcal{C}]$ form a sequence of distinct nodes $v_1, \ldots, v_j$ and edge $(v_i, v_{(i \bmod j)+1}) \in E[\mathcal{C}]$ for all $1 \leq i \leq j$. Note that, by this definition, a route of size two is a simple edge which is called *return trip* route [6]. We also consider a route of size 1 (without edges) and call it an *empty route*.

In the remainder of the article, we consider an instance $\mathcal{I}$ of the MMpLHP given by: the number p of required hub nodes; an undirected complete simple graph $G = (V, A)$ as defined above (each $v \in V$ is a hub candidate); a discount factor $\alpha \in [0, 1]$, for those edges connecting hubs; and a maximum length C allowed for any local route, that is, each of these routes may have at most C nodes, including the hub, which means C–1 customers are served.

A feasible solution x of the MMpLHP consists of a set of routes given by $\mathcal{R} = \{r_1, r_2, \ldots, r_p\}$. We represent this set by a tuple $t_x = \langle r_1 = t_x(1), r_2 = t_x(2), \ldots, r_p = t_x(p)\rangle$, with each r_k, for $k = 1, 2, \ldots, p$, being a sequence of nodes forming a local route where the first node of each r_k corresponds to its hub. It follows that the interhub route is obtained by taking the first node of each route in t_x. Figure 2 illustrates a solution of the MMpLHP. The cost of solution x is given by

$$\mathcal{F}(x) = \alpha \left[\left(\sum_{k=1}^{p-1} c_{\{r_k(1), r_{k+1}(1)\}} \right) + c_{\{r_p(1), r_1(1)\}} \right] + \sum_{k=1}^{p} \left[\left(\sum_{s=1}^{|r_k|-1} c_{\{r_k(s), r_k(s+1)\}} \right) + c_{\{r_k(|r_k|), r_k(1)\}} \right]. \quad (1)$$

In this version, we do not consider costs to open the hubs.

With respect to the interhub route, one can address three special cases, all related to parameter p. For $p = 2$, the interhub route is formed by one edge as a "bridge" between two local routes. For $p = 1$ or $p = n$, we have a single route (a local route or an interhub route, respectively). If the capacity $C \geq n$, the problem reduces to the Traveling Salesman Problem. In fact, using this reduction, one can easily argue that MMpLHP is $\mathcal{NP}$-hard.

Lemma 1. *Given an instance $\mathcal{I}$ of the TSP, let TSP($\mathcal{I}$) be an optimal solution value for the TSP, and MMpLHP($\mathcal{I}$) be an optimal solution value for MMpLHP such that $p = 1$, $\alpha = 1$, and $C \geq |V[\mathcal{I}]|$. Then TSP($\mathcal{I}$) = MMpLHP($\mathcal{I}$).*

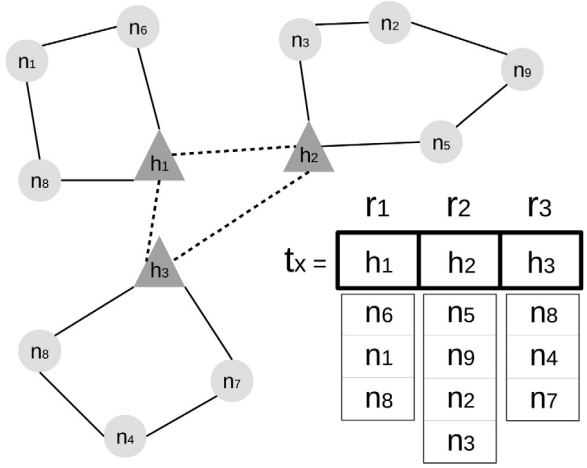


FIG. 2. Representation of a given solution.

Note that the maximum capacity of the local routes and the number of hubs are closely related. A low capacity requires a larger number of short routes to cover all the nodes in a valid solution. Therefore, feasible instances have the number of hubs and the capacity of the local routes related according to $C \geq \lceil n/p \rceil$.

3. BRKGA FOR THE MMPLHP

To use a BRKGA, we need to represent a solution with a chromosome and implement a decoder using this representation. The decoding phase is the only connection between the BRKGA and the problem itself. The implementation of the decoder needs to maintain the compatibility of the solution extraction and the genetic operators. We seek an adequate representation of a solution by a chromosome. In this phase, it is also possible to intensify the search applying a local search procedure on the decoded solution. Note that finding a better solution in the neighborhood implies adjusting the values of the genes (alleles) to reflect this new solution. The fitness of a decoded chromosome is the cost of the routes found. In following sections, we describe these particularities.

3.1. Representation

In most BRKGA implementations, a chromosome is a real vector $\mathbf{v}' \in [0, 1]^t$ following the procedure given in Bean [5]. Here, we use an integer vector $\mathbf{v} \in \mathbb{N}^t$. The projection of $\mathbf{v}' \to \mathbf{v}$ is done with the function $f(\mathbf{v}') = \lceil \beta \mathbf{v}' \rceil$ with $\beta > 1$ sufficiently large.

A chromosome is a $(p + n)$-vector $\mathbf{v}$ of integers. The first genes $\mathbf{v}_1, \ldots, \mathbf{v}_p \in \{1, \ldots, C\}^p$ represent the size of the local routes. The remaining genes $\mathbf{v}_{p+1}, \ldots, \mathbf{v}_{p+n} \in \mathbb{N}^n$ represent the position of a node in a local route.

To extract a solution from this chromosome, we sort the alleles $\mathbf{v}_{p+1}, \ldots, \mathbf{v}_{p+n}$ in nonincreasing order, generating a permutation π of the nodes. Nodes $\pi_1, \ldots, \pi_p$ represent the hubs. Each hub π_i, for $i = 1, \ldots, p$, has a local route formed by nodes $\pi_{p+o(i)+j}$, for $j = 1, \ldots, \mathbf{v}_i$ and $\mathbf{v}_i \geq 2$, such that

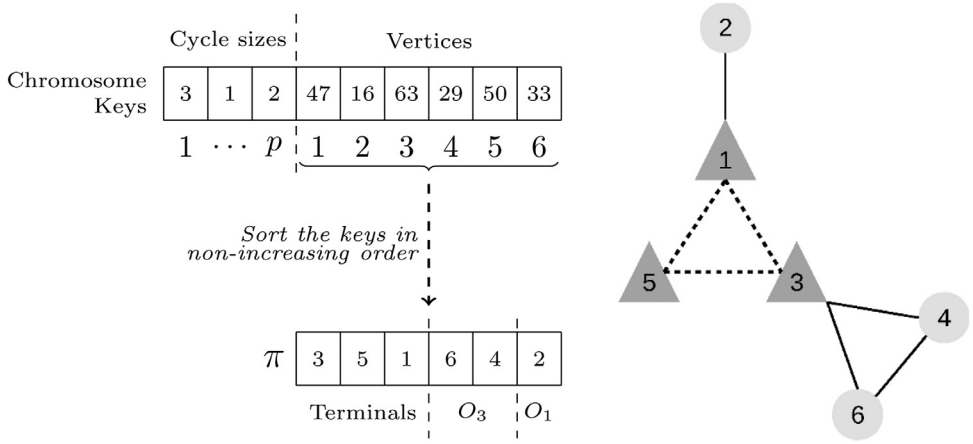


FIG. 3. Example of solution extraction.

$$o(i) = \begin{cases} o(i-1) + \mathbf{v}_{i-1} - 1 & \text{if } i > 1, \\ 0 & \text{otherwise.} \end{cases}$$

The function $o(i)$ indicates the offset in the alignment of routes in the permutation π. Note that $\mathbf{v}_i = 1$ indicates hub i is alone in an empty route.

For instance, consider the chromosome of Figure 3, where $n = C = 6$ and $p = 3$. To extract a solution, we sort the indices representing the nodes, using their corresponding keys and generate the permutation π. The first hub is the node 3 and it leads the local route r_1: $3 \to 6 \to 4$. The second local route r_2 is an empty route formed only by node 5. The local route r_3 is a return trip formed by node 1 as hub and node 2.

The advantage of this representation is that any modification in the values of keys $\mathbf{v}_{p+1}, \ldots, \mathbf{v}_{p+n}$ will not destroy the feasibility of the solution extracted from the chromosome. This approach was used in several papers, such as [67, 69]. Conversely, arbitrary modifications of keys $\mathbf{v}_1, \ldots, \mathbf{v}_p$ can result in invalid route sizes. In this case, it is necessary to use the repair procedure described in Section 3.2.

Note that distinct chromosomes may decode to the same solution either by rotation of the position of the alleles or by different key values in the same order. However, this is unlikely to occur because of the range of the integers used in practice.

3.2. *Decoding a Solution*

In BRKGAs, the decoder extracts a solution from the chromosomes as well as tries to improve the solution with some local search procedure. Using the representation of the previous section requires that the sizes of the local routes are correct, that is, the sum of the first p keys must equal the number of nodes. Algorithm 3 increments or decrements the size of each route based on the difference between the sum of the key values and the number of nodes. Note that the modifications are done among randomly selected routes with the objective of preserving the "structures" inherited from the parents. Therefore, we avoid deep changes in a particular route. The complexity of this repair procedure is $O(|pC - n|)$.

Algorithm 3 adjustRouteSizes

Input: a vector $\mathbf{v} \in \mathbb{N}^{p+n}$, non-negative integers n, p, and C.
Output: modified $\mathbf{v}$.

3.1 Let $B = \{1, \ldots, p\}$.
3.2 $diff \leftarrow \sum_{i \in B} \mathbf{v}_i - n$.
3.3 **if** $diff < 0$ **then**
3.4 $\quad sign \leftarrow -1$.
3.5 **else** $sign \leftarrow 1$.
3.6 **while** $diff \neq 0$ **do**
3.7 $\quad$ Choose i from B uniformly at random.
3.8 $\quad$ **if** ($\mathbf{v}_i > 2$ **and** $sign < 0$) **or** ($\mathbf{v}_i < C$ **and** $sign > 0$) **then**
3.9 $\quad\quad \mathbf{v}_i \leftarrow \mathbf{v}_i + sign$.
3.10 $\quad\quad diff \leftarrow diff + sign$.
3.11 $\quad$ **else**
3.12 $\quad\quad B \leftarrow B \setminus \{i\}$.

After the repair phase, the decoder proceeds with solution extraction as described in Section 3.1. At this point, it is possible (although rare) that two or more keys have identical values. This could generate different solutions with different values from the same chromosome. To avoid this situation, the decoder modifies the keys to have different values and, therefore, guarantees a unique solution. However, note that this modification must be done only on the keys corresponding to the node ordering. The keys corresponding to the route size are kept untouched. Algorithm 4 summarizes the entire process, including the local search procedures described in next section.

3.2.1. Local Improvements. Local improvement procedures have been shown to help convergence of genetic algorithms. Potentially, each chromosome can be in the

Algorithm 4 Decoder

Input: a vector $\mathbf{v} \in \mathbb{N}^{p+n}$, non-negative integers n, p, and C.
Output: modified $\mathbf{v}$, the *fitness* of $\mathbf{v}$

4.1 `adjustRouteSizes` $(\mathbf{v}, p, n, C)$.
4.2 Sort the alleles $\mathbf{v}_{p+1}, \ldots, \mathbf{v}_{p+n}$ in non-decreasing order, generating a vector $\mathbf{s}$ of sorted keys and a permutation π of the nodes.
4.3 Keeping the order, modify the values of $\mathbf{s}$ to ensure they are all different.
4.4 Extract the set of routes $\mathcal{R}$ using π and $\mathbf{s}$, and consider I the inter-hub route.
4.5 **foreach** $r \in (\mathcal{R} \cup \{I\})$ **do**
4.6 LK (r).
4.7 **foreach** $r \in \mathcal{R}$ **do**
4.8 $k \leftarrow$ `findBestTerminal` (r).
4.9 Consider k the new hub of r.
4.10 LK (I); $j \leftarrow 1$.
4.11 **foreach** $r \in \mathcal{R}$ **do**
4.12 $\mathbf{v}_{p+r(1)} \leftarrow \mathbf{s}_j$; $j \leftarrow j + 1$.
4.13 **foreach** $k \in r$ *such that* $k \neq r(1)$ **do**
4.14 $\mathbf{v}_{p+k} \leftarrow \mathbf{s}_j$; $j \leftarrow j + 1$.
4.15 Let x be a solution obtained from $\mathcal{R}$.
4.16 **return** $\mathbf{v}$ *and* $\mathcal{F}(x)$.

neighborhood of a different local minimum. In such a case, local search would be able to reach these minima [40].

To perform local improvements, we explore two different neighborhoods. The first is the well-known "edge exchange" neighborhood obtained from 2-opt and 3-opt movements (it is also known as the "λ-opt" neighborhood). Lin and Kernighan [42] generalize these methods resulting in one of the best heuristics for the Traveling Salesman Problem. Their procedure is adaptive and at each step decides dynamically what type of λ-opt should be applied. In [48], the Lin-Kernighan heuristic was improved using, at the start of each iteration, a "guess" or "kick" tour instead of only doing restarts. The *double-bridge move*, cheaper than a 4-exchange move, was also proposed by Martin *et al*. [48]. This heuristic is known as the *Chained Lin-Kernighan procedure* (LK). In [32], several refinements of LK are described. A sequential 5-opt step, restricted to a neighborhood of five α-nearness members, is used. This is computed using 1-trees on a modified weight distance matrix.

To explore the "edge exchange" neighborhood, we apply the LK heuristic to each local route as well as the interhub route but with a limited number of iterations. One can note that an exchange in positions of nodes in a route implies that the key values of the nodes must also be swapped. In particular, the modification in the positions of hubs in the interhub route must be done carefully so that the remaining structure of the chromosome is not destroyed. Lines 4.6 and 4.7 of Algorithm 4 summarize this procedure.

The second neighborhood "rotates" the local routes looking for new hub nodes capable of reducing the cost of the solution. Let r_i be a local route such that $r_i(1)$ is its hub. Let $r_{i-1}(1)$ and $r_{i+1}(1)$ be, respectively, the predecessor and the successor hubs of $r_i(1)$ in the interhub route. Rotating cycle r_i means finding a node $k = \text{argmin}_{k' \in r_i} \left\{ c_{(r_{i-1}(1),k')} + c_{(k',r_{i+1}(1))} \right\}$, that is, finding a node to be the new hub that minimizes the cost of the interhub route. We call this procedure findBestTerminal. In Section 5, we define a similar neighborhood structure called N_3, described by Algorithm 7. In this neighborhood, the local routes are kept intact but the hub and, consequently, the cost of the interhub route is changed. In fact, the node exchange can result in a new interhub route to which is applied the LK procedure. Lines 4.8–4.10 of Algorithm 4 summarize this procedure.

It is important to note that all these modifications on the solution imply that the keys of the corresponding chromosome must be modified to reflect the new solution. Lines 4.12–4.15 copy back the key values from the ordered vector $\mathbf{s}$ to reflect these modifications.

3.3. Initial Population

The common approach to initialize the population of a BRKGA is to generate the chromosomes with uniformly drawn random keys over the interval $[0, 1]$. This results in highly heterogeneous individuals that may slow down the convergence of the algorithm. Conversely, these individuals may lie in neighborhoods of different local minima and help avoid premature convergence of the algorithm. In an attempt to speed up the search, we use a fast heuristic to generate some good initial solutions and then complete the initial population with random chromosomes. An advantage of using such individuals in the starting population is that they are probably closer to a good solution than most random individuals.

The heuristic is based on a k-means clustering algorithm that partitions the nodes in k clusters such that each node belongs to the cluster with the nearest mean [43]. This is a well-known algorithm, used in fields such as machine learning and computational geometry. Each cluster generated by k-means is considered as a local route and the hub is the node closest to the mean of the cluster. The nodes are taken in arbitrary order as the cycles will be optimized in the decoding procedure. Although the cost of this solution is usually not very good, it carries some geometric information about the distribution of the nodes and it is likely that some parts of these clusters will be part of the final solution. One can note that this approach can be used only in geometric instances. In Barreto *et al*. [4], a similar approach is used to solve the Capacitated Location-Routing Problem.

The encoding of the chromosome from this solution is done in the following way: a vector $\mathbf{s}'$ is generated with n random keys, sorted in nonincreasing order. The first p positions of the chromosome are associated with the size of the

clusters taken in some arbitrary but fixed order, say $r_1, \ldots, r_p$. Using this same order, we associate each hub $r_i(1)$ with a key of $\mathbf{s}'$. For each cluster r_i we associate the nodes of this cluster with the remaining keys of $\mathbf{s}'$ in the order they appear (see Fig. 3). This ensures compatibility and uniformity with the BRKGA framework.

4. A VNS-BASED HEURISTIC

The proposed heuristic is said to be a VNS-based heuristic, because it is based on the main core of Algorithm 1, which is described at lines 1.3–1.4, and explores different neighborhood structures (in a sense similar to VND) during its local search phase. First, the heuristic constructs a random initial solution instead to consider the *shake* phase, and then it applies a consistent local search procedure to improve it.

Algorithm 5 depicts the main steps of this heuristic, which we refer to as M-VND. Observe that M-VND represents lines 1.3–1.4 of Algorithm 1, except for the *shake* phase that is now substituted by a random procedure, which constructs an initial solution every time M-VND loops instead to obtain a random solution from a neighborhood structure, and for *MAX* that is not related with the number of neighborhood structures but now with a maximum number of nonimprovements that the best solution can have. In fact M-VND can be seen as a multistart algorithm that uses a VND-based method as its local search.

Algorithm 5 M-VND

Input : Instance I; maximum number of iterations MAX.
Output: Solution for the MMpLHP.

```
5.1 imp ← 1.
5.2 while imp ≤ MAX do
5.3     x ← construct an initial random solution for I.
5.4     x̄ ← LocalSearch(I, x).
5.5     if F(x̄) < F(x) then
5.6         imp ← 0; x ← x̄.
5.7     else
5.8         imp ← imp + 1.
5.9 return x.
```

4.1. Random Solutions

A random solution x is generated by first locating p hubs, chosen at random in V, and attributing them to $r_1(1), \ldots, r_p(1)$. Next, the remaining nodes in V are allocated to $r_1, \ldots, r_p$. For each non-hub node a hub is randomly selected. Then the non-hub node is linked by an edge to the last non-hub node in the route associated with the selected hub, or directly to the hub if no such route exists.

More precisely, the procedure selects both $i \in V \setminus \{r_1, \ldots, r_p\}$ and $k \in \{1, 2, \ldots, p\}$ at random and appends i to r_k if $|r_k| \leq C$. Otherwise, the algorithm randomly selects $s \in \{1, 2, \ldots, p\} \setminus \{k\}$ until it reaches the first r_s such that $|r_s| < C$. These steps are repeated until all nodes in V be allocated to some route.

4.2. A VND-Based Algorithm

VND is a particular variant of the VNS algorithm, which is commonly adopted as local search within other algorithms. The VND considers κ distinct neighborhood structures, denoted by $N_1, N_2, \ldots, N_\kappa$. Each neighborhood structure is characterized by a move (or even a combination of moves) and contains all solutions (neighbors of x) that can be obtained by applying such a move (one at a time on a given solution x). A move consists of operations applied on x to obtain a new solution x'. Example of moves in location routing problems (see [35] for more details) are related to add, remove or swap elements of position in x.

The neighborhood structures are generally considered in a systematic way, that is, the search starts exploring neighborhood N_1 until no further improvement is possible. Next, the search continues with N_2. If an improved solution could be determined in N_2, then the VND returns to N_1 to explore the neighborhood of this new solution until the last neighborhood. Otherwise, it continues with N_3, and so on. Observe that, after it explores the N_κ neighborhoods with no improvement, the solution is local optimum with respect to all neighborhoods [35].

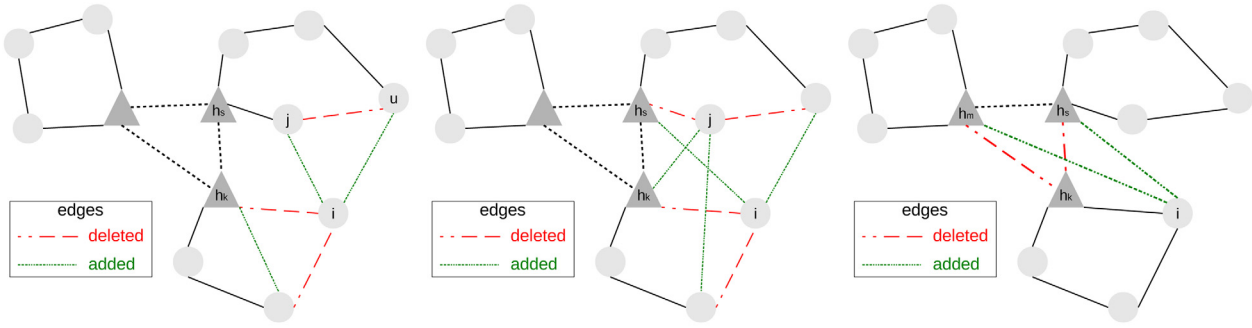
In the sequel, three neighborhood structures N_1, N_2, and N_3 are proposed for the problem under consideration and they aim to explore different ways to locate hubs and allocate non-hub nodes to hubs. We assume that the pair of routes (r_k, r_s) is different from (r_s, r_k), that is, the order is taken into account.

Neighborhood N_1 of a solution x is characterized by a move with operations of remove and insert over local routes, which consists in removing a node from a local route r_k and inserting it into another local route r_s. In other words, given a pair of local routes (r_k, r_s), $i \in r_k$ and an edge $\{j, u\}$ of route r_s, remove i from r_k and add it to r_s so edge $\{j, u\}$ now becomes edges $\{j, i\}$ and $\{i, u\}$. Figure 4a illustrates this move.

The second neighborhood, denoted by N_2 correspond to swap moves over local routes. That is, for a given pair of local routes (r_k, r_s) and nodes $i \in r_k$ and $j \in r_s$, i is moved to r_s and j to r_k, where Figure 4b illustrates this example. It is important mentioning that both N_1 and N_2 are only applied on non-hub nodes.

The last neighborhood structure, denoted by N_3, is concerned with new locations for hub nodes by applying an exchange move. In other words, for a given route r_k and node $i \in r_k$, consider now node i its hub with edges updated accordingly. The aim is to search for the node that would decrease the solution cost the most if converted to a hub. Figure 4c shows when node i becomes the new hub of route r_k.

Accordingly to Mladenović and Hansen [53], after defined the neighborhood structures, they have to be evaluated in order to find the best solution on them for an input solution x. Therefore, it is easy (not computationally speaking) evaluate



(a) Remove/add move of N_1. (b) Swap move of N_2. (c) Exchange move of N_3.

FIG. 4. Moves of N_1, N_2, and N_3. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

each neighborhood just by applying its move considering all possibilities, namely do a local search to find a local optima solution in the given neighborhood. For example, in the case of N_1, it has to check the remove/add move for each pair of local routes (r_k, r_s) in solution x, each node $i \in r_k$ and each edge $\{j, u\}$ of route r_s, so the one that improves x at its maximum (that is, resulting on the minimum $\mathcal{F}()$) is indeed chosen. The same remains valid for N_2 and N_3.

The strategy used here to evaluate the neighborhood structures consists of doing a local search as explained in the last paragraph, however instead of getting the move that results on the local optima solution for the respective neighborhood, all operations that can indeed improve the input solution x (even by a small amount) are stored in a list. Therefore, we select an operation from this list uniformly at random and apply it after all. Algorithm 6 describes the procedure that explores neighborhood N_1 in this way. Note that each successful remove/add move (i.e., an operation that results in a decrease of $\mathcal{F}(\bar{x})$) is stored in a list L.

For the neighborhood structure N_2, it is sufficient to consider Algorithm 6 substituting the remove/add move to the corresponding swap move at lines 6.4 (it becomes node $j \in r_s$) and 6.5 (opr consists in swapping i and j). Conversely, Algorithm 7 describes the procedure to explore neighborhood N_3. With this neighborhood, new solutions characterized by different hubs are considered.

Algorithm 6 Procedure to explore neighborhood N_1

Input : Instance I; solution x.
Output: Solution $\bar{x}$.
6.1 $\bar{x} \leftarrow x$; $L \leftarrow \emptyset$.
6.2 **foreach** *pair of routes* $\{r_k, r_s\} \in \bar{x}$ **do**
6.3 **foreach** *node* $i \in r_k$ **do**
6.4 **foreach** *edge* $\{j, u\} \in r_s$ **do**
6.5 $opr \leftarrow$ *remove* i of r_k and $\{j, u\}$ of r_s; *add* edges $\{j, i\}$ and $\{i, u\}$ in r_s.
6.6 Apply opr to $\bar{x}$.
6.7 **if** $\mathcal{F}(\bar{x}) < \mathcal{F}(x)$ **then** $L \leftarrow L \cup \{opr\}$.
6.8 Undo opr in $\bar{x}$.
6.9 $opr \leftarrow$ select uniformly at random one element of L.
6.10 Apply opr to $\bar{x}$.
6.11 **return** $\bar{x}$.

Algorithm 7 Procedure to explore neighborhood N_3

Input : Instance I; solution x.
Output: Solution $\bar{x}$.
7.1 $\bar{x} \leftarrow x$; $L \leftarrow \emptyset$.
7.2 **foreach** *route* $r_k \in \bar{x}$ **do**
7.3 Let h_k be the hub of r_k.
7.4 **foreach** *node* $i \in r_k$ *with* $i \neq h_k$ **do**
7.5 Let $e_1 = \{h_m, h_k\}$ and $e_2 = \{h_s, h_k\}$ be the edges of the inter-hub route incident to h_k.
7.6 $opr \leftarrow$ *exchange* edges e_1 and e_2 by $\{h_m, i\}$ and $\{h_s, i\}$ in $\bar{x}$.
7.7 Apply opr to $\bar{x}$.
7.8 **if** $\mathcal{F}(\bar{x}) < \mathcal{F}(x)$ **then** $L \leftarrow L \cup \{opr\}$.
7.9 Undo opr in $\bar{x}$.
7.10 $opr \leftarrow$ select uniformly at random one element of L.
7.11 Apply opr to $\bar{x}$.
7.12 **return** $\bar{x}$.

The order in which each neighborhood is explored can substantially affect the performance of the search, since the first structures may be explored more than the last ones. We consider a deterministic order exploring N_1, then N_2, and finally N_3. Both N_1 and N_2 work over the non-hub nodes, while N_3 observes only hub locations. Therefore, the proposed VND cycles through these neighborhoods and returns a solution that traversed all neighborhoods and it is expected to be the local optima among all of them. However, note that due to the stochastic choice related with the list L during neighborhoods exploration, this VND is not a best improvement method (or even a first improvement one) at all.

To intensify the search, once the solution can no longer be improved, we nest the cycles through the neighborhoods with the Chained Lin-Kernighan heuristic (LK) which is

applied in each route of the current solution. Therefore, if the solution is improved with LK, the neighborhoods are explored once again. Otherwise, we nest these steps with a best-improvement version of the procedure that explores N_3, denoted by $bestN_3$. Algorithm 8 outlines the Nested-VND procedure and is adopted as the local search routine in M-VND (at line 5.4). Note that the main core of a basic VND is outlined at lines $8.5 - -8.11$

Algorithm 8 Nested-VND

```
      Input  : Instance I; solution x.
      Output: Solution x̄.
8.1   x̄ ← x.
8.2   repeat
8.3       out ← false.
8.4       repeat
8.5           ip ← 1; in ← false.
8.6           while ip ≤ 3 do
8.7               x̄ ← Procedure to explore N_ip(I, x̄).
8.8               if F(x̄) < F(x) then
8.9                   ip ← 1; x ← x̄.
8.10              else
8.11                  ip ← ip + 1.
8.12          x̄ ← LK(I, x̄).
8.13          if F(x̄) < F(x) then
8.14              in ← true; x ← x̄.
8.15      until in = true
8.16      x̄ ← bestN3(I, x̄).
8.17      if F(x̄) < F(x) then
8.18          out ← true; x ← x̄.
8.19  until out = true
8.20  return x̄.
```

As LK improves the routes within a solution x, reorganizing node sequences according to λ-moves, it is natural to attempt to improve hub locations as well. The purpose of $bestN_3$ is to cycle through the interhub route of x until no improvement in $\mathcal{F}(x)$ be possible. It starts in route r_k, with $k = 1$ at the very beginning, trying each node $i \in r_k$ as hub. If there is such an $i \in r_k$ that improves $\mathcal{F}(x)$, then $bestN_3$ restarts in route r_s, with $s = k + 1$; otherwise, it continues analyzing the next route. Starting at k, it stops after analyzing all routes and concluding there is no new hub node i in any of them capable of improving x.

4.3. A Reduced Version of the Nested-VND

The Nested-VND algorithm may require high computational effort due to intensive restarts during the neighborhood explorations. Therefore, we consider a simple variant in which the neighborhood structures are sequentially explored but without returning immediately to the first structure when any of them can improve the solution. In other words, the algorithm returns to the first structure after applying all structures consecutively and only if improvements in the solution are found. The LK and $bestN_3$ are still used but also in a consecutive way, even no improvement on the solution occurs. Algorithm 9 assembles the ideas stated thus far in a reduced Consecutive nested Neighborhood Search (CNS). Let M-CNS be the M-VND algorithm in which the local search is now performed with the CNS.

Algorithm 9 CNS

```
      Input  : Instance I; solution x.
      Output: Solution x̄.
9.1   x̄ ← x.
9.2   repeat
9.3       out ← false.
9.4       repeat
9.5           in ← false.
9.6           x̄ ← Procedure to explore N1(I, x̄).
9.7           x̄ ← Procedure to explore N2(I, x̄).
9.8           x̄ ← Procedure to explore N3(I, x̄).
9.9           if F(x̄) < F(x) then
9.10              in ← true; x ← x̄.
9.11      until in = true
9.12      x̄ ← LK(I, x̄); x̄ ← bestN3(I, x̄).
9.13      if F(x̄) < F(x) then
9.14          out ← true; x ← x̄.
9.15  until out = true
9.16  return x̄.
```

5. INTEGER PROGRAMMING FORMULATION

We propose an integer programming model for the MMpLHP that uses binary variables. Variable $x_e = 1$ indicates that edge e links two hubs. Variable $z_e = 1$ considers e to be present in the solution but involving at least one non-hub. Variable $w_e = 1$ is used to represent a local route connecting a hub to only one non-hub node modeling local routes that serve only a single non-hub, namely *return trips* according to Belenguer *et al.* [6]. Also, variable $y_i = 1$ assumes a hub located at node i, that is, i is a hub node.

The following notation is used. For $S \subseteq V$, let $\delta(S)$ be the set of edges with one extremity in S and the other in $S \setminus V$, and the notation $\delta(\{i\})$ is simplified to $\delta(i)$. Consider $E[S]$ as the set of edges with both extremities in S. We denote by $\mathcal{P}_{ij}$ the set of all paths connecting node i to node j. For $P \in \mathcal{P}_{ij}$, $V[P]$ denotes the set of nodes in P.

Formulation (2) describes the integer programming model whose objective function aims to minimize the overall cost of locating p hubs and making the interhub and local routes. Constraints (2.i) ensure the number of hubs opened must be equal to p. Constraints (2.ii) are degree constraints for nodes assuring that each one has at most two incident edges, since some hubs (station ones) do not have any local route. Similarly, constraints (2.iii) are the degree constraints for

hub nodes considering the interhub route. Constraints (2.*iv*) impose that the edges cannot be simultaneously in the interhub and any other local route, including return trips. To avoid unconnected solutions, constraints (2.*v*) ensure that for any subset S of nodes, the number of edges (regardless of being in the interhub or in a local route) in $\delta(S)$ must be at least two. The same applies to constraints (2.*vi*), but now only observing hubs that are open and the edges in the interhub route.

We have path elimination constraints in (2.*vii*) that prevent routes starting at a hub and finishing at another one. In other words, suppose there is a path $P \in \mathcal{P}_{ij}$ in which i and j are hubs, and the remaining nodes are non-hub nodes. It follows that $\sum_{e\in E[P]} z_e + \sum_{k\in V[P]} y_k = |E[P]| + 2 = |V[P]| + 1 > |V[P]|$. Conversely, suppose there is a path $P \in \mathcal{P}_{ij}$ for which inequality $\sum_{e\in E[P]} z_e + \sum_{k\in V[P]} y_k \geq |V[P]| + 1$ (**) holds. Without loss of generality, we consider such a P of minimum length. Note that i and j are necessarily hubs, otherwise we can remove them and get a shorter path than P which also violates the inequality in (2.*vii*). Besides that, all the remaining nodes in P must be non-hub nodes. For example, suppose that one of these remaining nodes is a hub. Let us name it h, so $y_h = 1$. As P is the shortest path in P_{ij}, it follows that the sub-paths connecting i to h, P_{ih}, and h to j, P_{hj}, also satisfy the inequality in (2.*vii*), that is, $\sum_{e\in E[P_{ih}]} z_e + \sum_{k\in V[P_{ih}]} y_k \leq |V[P_{ih}]|$ and $\sum_{e\in E[P_{hj}]} z_e + \sum_{k\in V[P_{hj}]} y_k \leq |V[P_{hj}]|$ hold. Adding these inequalities, we get $\sum_{e\in E[P]} z_e + \sum_{k\in V[P]} y_k + y_h \leq |V[P_{ih}]| + 1$, which contradicts (**).

Capacity constraints are given in (2.*viii*) and impose that the number of incident edges in any subset of nodes must be at least twice the number of routes required to serve their demand, considering those nodes as non-hub nodes. From the rounded capacity inequalities in [54], we derive $\sum_{e\in\delta(S)}(z_e + 2w_e) \geq 2\left\lceil \frac{|S|}{C-1} \right\rceil$ for a given $S \subset V$, since each route can serve at most C–1 non-hub nodes. Note that the number of hubs in S is not known during optimization, therefore we need to consider $\sum_{i\in S}(1 - y_i)$ instead of $|S|$, consequently the ceiling function must be removed. Moreover, observe that for each hub in S, we can subtract two incident edges on S obtaining the term $-2\sum_{i\in S} y_i$.

Constraints (2.*ix*) and (2.*x*) assure that a return trip is composed by a non-hub node connected to some hub. And, integrality constraints of the variables are given in (2.*xi*) and (2.*xii*).

$$\min \quad \sum_{e\in A} c_e(\alpha\, x_e + z_e + 2w_e)$$

subject to :

$$(i) \quad \sum_{e\in V} y_e = p$$

$$(ii) \quad \sum_{e\in\delta(i)} (z_e + 2w_e) \leq 2, \qquad \forall i \in V$$

$$(iii) \quad \sum_{e\in\delta(i)} x_e = 2y_i, \qquad \forall i \in V$$

$$(iv) \quad x_e + z_e + w_e \leq 1, \qquad \forall e \in A$$

$$(v) \quad \sum_{e\in\delta(S)} (x_e + z_e + 2w_e) \geq 2, \qquad \forall S \subset V : |S| \geq 1$$

$$(vi) \quad \sum_{e\in\delta(S)} x_e \geq 2(y_i + y_j - 1), \qquad \forall i,j \in V, \forall S \subset V : |S \cap \{i,j\}| = 1$$

$$(vii) \quad \sum_{e\in E[P]} z_e + \sum_{k\in V[P]} y_k \leq |V[P]|, \qquad \forall i,j \in V : i \neq j, \forall P \in \mathcal{P}_{ij}$$

$$(viii) \quad \sum_{e\in\delta(S)} (z_e + 2w_e) \geq 2\left(\frac{\sum_{i\in S}(1 - y_i)}{C-1} - \sum_{i\in S} y_i\right), \qquad \forall S \subset V : |S| \geq 2$$

$$(ix) \quad w_e \leq y_i + y_j, \qquad \forall e = \{i,j\} \in A$$

$$(x) \quad z_e + w_e + y_i + y_j \leq 2, \qquad \forall e = \{i,j\} \in A$$

$$(xi) \quad x_e, z_e, w_e \in \{0,1\}, \qquad \forall e \in A$$

$$(xii) \quad y_i \in \{0,1\}, \qquad \forall i \in V. \qquad (2)$$

5.1. *Separation Algorithms*

The number of constraints (2.*v*) − (2.*viii*) may be very large in practice. Therefore, separation routines are used to search for such violated inequalities and they are added into the formulation in a cutting plane fashion.

First of all, let $(\bar{x}, \bar{y}, \bar{z}, \bar{w})$ be the fractional optimal solution of the current node of the branch-and-bound tree during the optimization. For constraints (2.*v*) and (2.*vi*), we used the implementation of the algorithm in [17] to compute the Gomory-Hu tree [25] which contains s–t minimum cuts, defined as $MC_{s,t}$, for $s,t \in V$. In the case of (2.*v*), the graph used by the Gomory-Hu algorithm is $\bar{G} = (V,A) = G$, but now its edges have weight defined according to the values of $\bar{x}_e$, $\bar{z}_e$, and $\bar{w}_e$. For constraints (2.*vi*), we must observe only the values of $\bar{x}_e$.

Therefore, after the Gomory-Hu tree under $\bar{G}$ be computed, we check the violation of (2.*v*) observing all edges $\{s,t\} \in A$. In other words, for all $\{s,t\} \in A$, compute $\dot{v}_{\{s,t\}} = 2(\bar{y}_s + \bar{y}_t - 1) - \sum_{e\in MC_{s,t}}(\bar{x}_e + \bar{z}_e + 2\bar{w}_e)$. Let $\{i,j\}$ be the edge which attains $\max_{\{r,s\}\in A}\dot{v}_{\{r,s\}} > 0$. If $\{i,j\}$ exists, we insert the inequality for $\delta(S) = \left\{e \in MC_{s,t}\right\}$. The same steps are performed for constraints (2.*vi*).

For path constraints (2.*vii*), the algorithm considers the graph $\bar{G}' = (V, \bar{A}')$, where $\bar{A}' = \{e \in A | \bar{x}_e + \bar{z}_e > 0\}$. From the hubs in $\bar{G}'$, the algorithm takes hub i with the maximum value of $\bar{y}_i$ (ties are broken randomly), and performs a search enumerating all paths starting from hub i. The first path that violates such a constraint is then considered and the algorithm

stops. It is worth mentioning that we first check the violation of these constraints considering the following procedure, so the previous algorithm is applied only if the next one fails. For each nonvisited hub i in $\bar{G}'' = (V, \bar{A}'')$ in which $\bar{A}'' = \{e \in A | \bar{z}_e > 0\}$, the procedure performs a breadth-first search to find another hub j, so that there is a path connecting i with j in $\bar{G}''$. Then, we insert an inequality to avoid such a path. Observe that we do not need to consider variables w_e (return trips) within path constraints, since paths involving return trips are eliminated with constraints (2.*ix*) and (2.*x*), so here we are only interested in paths with at least two non-hub nodes.

Capacity constraints (2.*viii*) cannot be separated in polynomial time, as they generalize the rounded capacity inequalities, which require solving an $\mathcal{NP}$-hard problem [54]. We adapted the heuristic procedure in [47] that searches for a violated rounded capacity constraint, but it does not guarantee success. In our case, we consider the graph $\bar{G} = (V, \bar{A})$ such that $\bar{A} = \{e \in A | \bar{x}_e > 0\}$. Let $S_1, S_2, \ldots, S_p$ be the connected components of $\bar{G}$, so we check the violation of (2.*viii*) for each of these components and also for $V \setminus S_1, V \setminus S_2, \ldots, V \setminus S_p$.

Another class of inequalities inserted at the beginning of the optimization is represented in inequalities (3). This class guarantees that any edge in E is in the interhub route only if both its incident nodes are hubs. Although these inequalities are not necessary to obtain a valid solution, they enable a tighter formulation. The running time cost of inserting and dealing with these inequalities is almost negligible.

$$\begin{aligned} x_{ij} &\leq y_i, \\ x_{ij} &\leq y_j, \qquad \forall \{i,j\} \in A. \end{aligned} \tag{3}$$

6. COMPUTATIONAL STUDY

6.1. Environment

All algorithms were implemented in the C++ programming language and the experiments occurred on identical machines with quad-core Intel Xeon E5530 2.4 GHz CPUs and 32 GBytes of RAM running GNU/Linux. For the chained Lin-Kernighan, we used an implementation available on Concorde *TSP* Solver [3]. For the integer linear program, Formulation (2) is solved with a branch-and-cut algorithm that uses the framework provided by the Gurobi Optimizer version 5.6.2. We call this algorithm Exact. We also tested the commercial solver LocalSolver version 6.0. This solver consists in a collection of techniques using a hybrid neighborhood search approach. According to its documentation, LocalSolver combines local search techniques, constraint propagation and inference techniques, linear and mixed-integer programming techniques, as well as nonlinear programming techniques. This algorithm is denoted by LS.

6.2. Instances

We consider 77 instances adapted from the TSPLIB repository for the TSP, for which we used three scenarios of values of p and C observing the total number of nodes n. We consider that the 28 instances with less than 100 nodes are *small* instances, and the 49 instances within 100 and 1000 nodes are *large* instances. The value of α is set to $\{0.2, 0.4, 0.6, 0.8\}$ in accordance with Camargo *et al*. [11]. The aim is to stress the algorithms in face of different types of networks that may arise in practical applications, as in transportation problems or when designing and planning optical fiber networks.

We used the following scenarios:

- Scenario *ST*: $p = \lceil 0.2n \rceil$ and $C = \left\lceil \frac{n}{p} \right\rceil$: this scenario has tight local routes that force every hub to have a nonempty local route associated with it;
- Scenario *SL*: p as in *ST* and $C = \left\lfloor 1.8 \left\lceil \frac{n}{p} \right\rceil \right\rfloor$: this scenario allows large local routes and hubs may have no local route associated with them;
- Scenario *SQ*: $p = C = \lceil \sqrt{n} \rceil$: this scenario allows large local routes and hubs may have no local route associated with them.

6.3. Algorithm Setup

For Exact, we set the y variables as the highest priority variables for branching. After some preliminary tests, once hubs are located, the problem is simplified. The separation algorithms are applied whenever a feasible solution or the corresponding node relaxation solution is found. First, we check the violation of constraints (2.*v*) and (2.*vi*). Then, if necessary, constraints (2.*vii*) and (2.*viii*) are checked. After all, when no such a violated constraint is found, the separation procedure proposed by [47] is executed to detect rounded capacity inequalities as first defined for the capacitated vehicle routing problem. However, in our case, we use our adapted version to check the violation of capacity inequalities as in (2.*viii*).

To tune the parameters of the three heuristics, we use the *iterated racing* procedure [9]. This method consists in sampling configurations from a particular distribution, evaluating them using either the Friedman test or the t-test, and refining the sampling distribution with repeated applications of F-Race. We use the irace package [46], implemented in R, for parameter tuning. For each heuristic, we use a budget of 4000 experiments in the tuning procedure, where each experiment was limited to 1 h.

Following the values suggested by irace for the BRKGA. The population size was set to $p = 1500$, the elite size to $p_e = \lceil 0.20p \rceil$, and the number of mutants to $p_m = \lfloor 0.10p \rfloor$. The probability of inheriting each allele from the elite parent was $\rho_e = 0.80$. We used the island model [74] with three independent and concurrent populations where every 100 generations each population exports its best solution to the other populations. After 500 generations without improvement, all populations are reset to vectors of random keys. The kick type used by LK was set to find four closest 4-swap moves, and the max number of 4-swaps without progress (stallcount) was set to 1000. We use 4 simultaneous cores for decoding. Conversely, the maximum number of iterations *MAX* used as input for the M-VND and M-CNS is set to 30,000, that is, such algorithm stops when 30,000 consecutive iterations do not improve the solution.

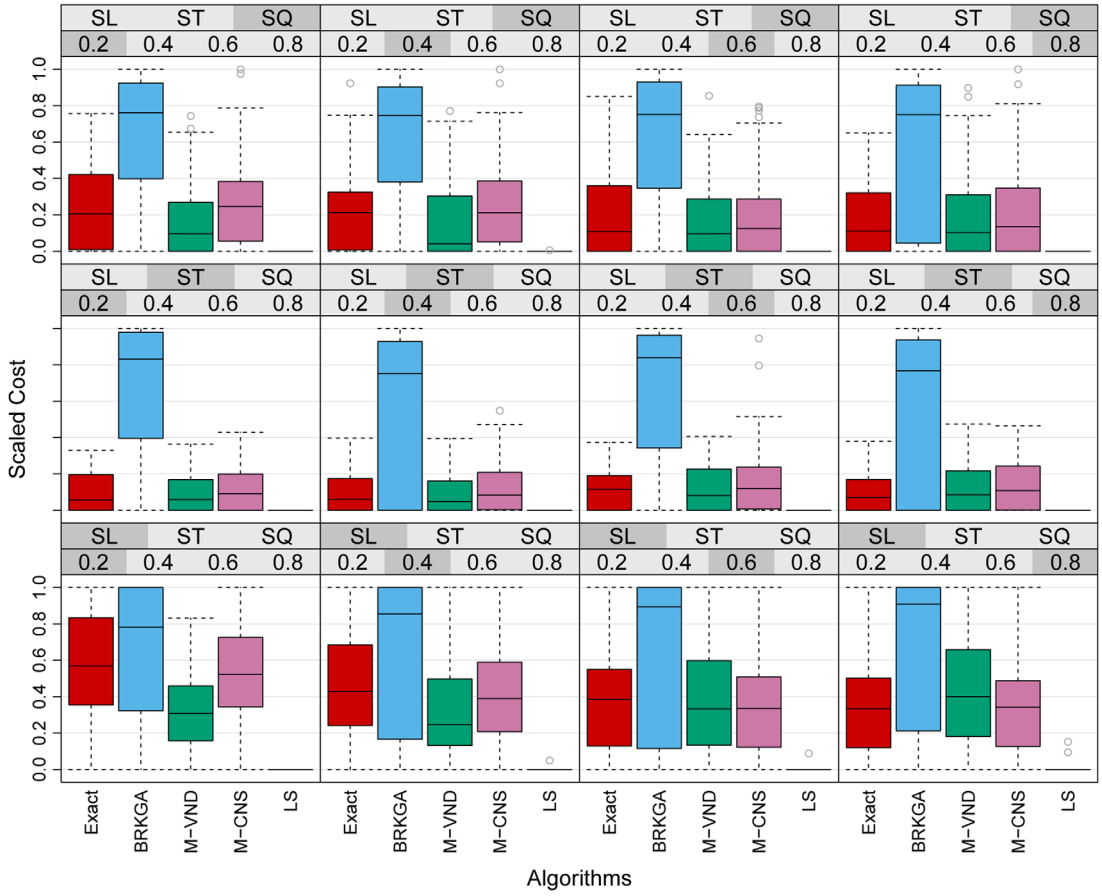


FIG. 5. Dispersion of costs for each algorithm for each scenario and α value for small instances. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The LocalSolver routine is used with the default parameters, with the maximum threads set to four, as in the other algorithms. Besides, for each algorithm the time limit was set to 3600 s to solve each instance, and the solution computed with the heuristic is used as initial upper bound by Exact.

6.4. *Results*

To compare the algorithms with respect to solution cost, it is necessary to scale the results since each instance can have very different values and even different orders of magnitude. For each instance $\mathcal{I}$, let $\chi_{\mathcal{I}}$ be the set of costs of the solutions found for $\mathcal{I}$, and $D_{\mathcal{I}} = \max(\chi_{\mathcal{I}}) - \min(\chi_{\mathcal{I}})$. The scaling is done by the simple transformation

$$\chi'_{\mathcal{I}} = \begin{cases} (x - \min(\chi_{\mathcal{I}}))/D_{\mathcal{I}} & \forall x \in \chi_{\mathcal{I}} \text{ and } D_{\mathcal{I}} > 0, \\ 1 & \text{otherwise.} \end{cases}$$

where $\chi'_{\mathcal{I}}$ is the set of scaled costs. Note that all values are scaled to the range $[0, 1]$.

Using this scaling process, Figures 5 and 6 shows the distribution of costs for each algorithm. In Figure 5, the distribution of costs for small instances (with less than 100 nodes) are depicted. In Figure 6, the distribution of costs for large instances (within 100 and 1000 nodes) are depicted. Note that in the last figure, the Exact algorithm results are not show since this algorithm was not able to produce better solutions than the incumbent generated by the heuristic. The box plots show the location of the first quartile, the cost median and the third quartile. The whiskers extend to the most extreme revenue no more than 1.5 times the length of the box. The dots are the outliers. Each plot corresponds to a scenario and α value as indicated by the gray strips. As expected, LocalSolver presented very good results excelling in all scenarios. We believe that the variability and specialization of the heuristics called by the LocalSolver procedure were able to detect and solve this problem very well. One can note that the M-VND and the M-CNS presented very good results in such that their medians are below the medians from BRKGA in all cases. Comparing with the Exact algorithm, M-VND obtained better results in the most cases. The surprise was the BRKGA that did not present the expected results even if compared to the Exact algorithm, particularly on *ST* scenario. We believe that the learning mechanism was not effective for this problem since the uniform-like crossover of BRKGA may introduce undesired mutations [71]. Although BRKGA did not present the expected results, it was able to find some good solutions as the whiskers and outliers show. One can also note that the algorithms presented larger variance in the *SL* scenario than others, suggesting that *SL* may be harder to solve than other

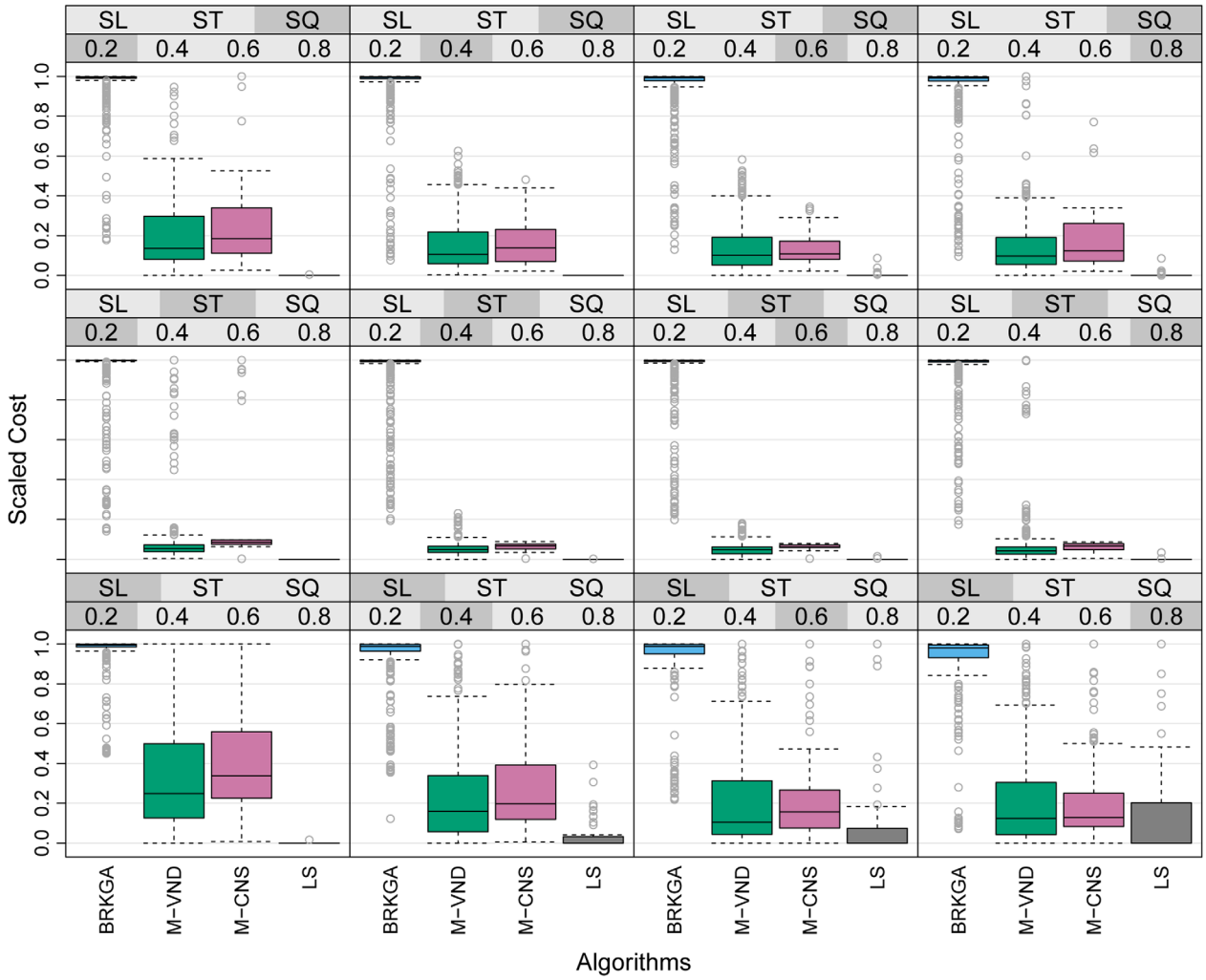


FIG. 6. Dispersion of costs for each algorithm for each scenario and α value for large instances. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

scenarios. Camargo *et al*. [11] related that different values of α may present differences in the hardness of the instance, although we could not detected in our experiments.

The box plots help shape our intuition that our approaches obtained better results than those of previous methods. To confirm our conclusions, we tested the normality of these distributions using the Shapiro–Wilk test and applied the Mann–Whitney–Wilcoxon U-test, considered more effective than the t-test for distributions sufficiently far from normal and for sufficiently large sample sizes [13, 21]. For all tests, we assume a confidence interval of 99%. For almost all distributions (76% of them), the p-values are above 0.05 indicating that we cannot reject the normality hypotheses. Since one can find non-normal distributions, we applied the U-test which assumes as null hypothesis that the location statistics are equal in both distributions. As several statistical tests were performed, we used a p-value correction procedure based on false discovery rate (FDR) to minimize the number of false positives (Type I error) as indicated by Benjamini and Hochberg [8].

Table 1 shows U test results for each pair of algorithms at 99% of confidence level. The structure of this table is as follows. Each row and column is indexed by one algorithm. Each element in the diagonal (bold) is the median of the corresponding algorithm. The upper-right diagonal elements are the differences in location statistics for each pair of algorithms. A positive difference indicates that the "row algorithm" has its location statistics higher (worse) than the "column algorithm," and a negative difference is the opposite. The bottom-left diagonal elements are the p-values of each test. We omitted all $p < 0.01$ values, that indicate that the difference is statistically significant for those pairs. We also omitted confidence intervals as for all tests the values lie in these intervals and they are very narrow. The Exact algorithm obtained good results, significantly better than BRKGA and significantly worse than M-VND. With respect to M-CNS, the difference was not significant since $p > 0.01$. M-VND was significantly better than M-CNS although the difference in location statistics was small. Confirming the box plot observations, BRKGA was significantly worse than the other

TABLE 1. Difference in median location for cost distributions for all instances, using a confidence interval of 99%. The main diagonal in bold shows the median of the results of each algorithm.

	Exact	BRKGA	M-VND	M-CNS	LS
Exact	**0.18**	−0.68	0.03	0.00	0.17
BRKGA		**0.98**	0.79	0.69	0.98
M-VND			**0.08**	−0.05	0.08
M-CNS	0.31			**0.18**	0.17
LS					**0.02**

TABLE 2. Algorithm performance considering the best found solutions.

Inst.	Alg.	Best solutions		Prop. diff.	
		% Best	% Run	%	σ
Small	Exact	24.11	24.11	4.07	3.17
	BRKGA	29.46	35.25	12.54	11.68
	M-VND	28.27	32.24	4.21	2.64
	M-CNS	24.11	22.88	4.23	2.82
	LS	98.51	98.51	0.40	0.28
Large	BRKGA	0.00	0.00	262.45	1032.94
	M-VND	12.50	1.26	10.69	8.68
	M-CNS	86.97	70.32	13.23	13.26
	LS	86.97	86.97	9.03	14.59

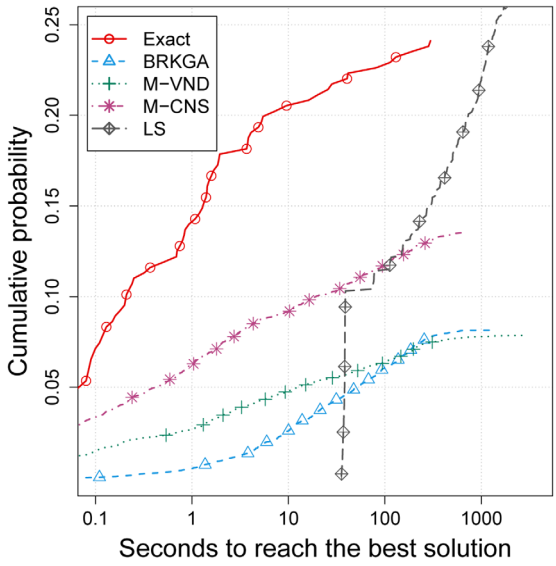


FIG. 7. Running time distributions to optimal assignment of routes. The identification marks correspond to 2.0% of the points plotted for each algorithm. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

algorithms. We can also note that *LS* was significantly better than the others algorithms.

Table 2 reports the performance of the algorithms. The first column is the class of the instances for which the results are shown. The second column is the name of the algorithm. The first group describes the percentages of best solutions found of each algorithm. Column "% Best" shows a percentage of the number of best solutions found, and column "% Run" shows a percentage of the number of runs on which the algorithm found a best solution. The two columns under label "Prop. diff." show, respectively, the average of the proportional difference between the best solution value and the achieved value (%), and its corresponding standard deviation (σ). For small instances (less than 100 nodes), Exact found few best solutions while the other solutions it produced did not vary too much with respect to quality. Note that both "% Opt" and "% Run" have the same value since a single run was performed per instance and scenario. The heuristics performed well, finding more than 27% of the best solutions on average. Note that BRKGA found more best solutions than M-VND and M-CNS in a larger percentage of runs. Conversely, the gap between the nonbest solutions that BRKGA found is much larger than for the other algorithms, as it can be see visually on the box plots and in the fifth and sixth columns of Table 2. Therefore, although BRKGA can find good solutions, its results might vary a lot. BRKGA was the only algorithm that was able to find the best solution to instances eil51, scenario SL, and $\alpha = \{0.6, 0.8\}$; and swiss42, scenario SL, and $\alpha = 0.8$. Only M-VND found best solutions for swiss42, scenario SL, and $\alpha = 0.4$, and pr76, scenario SQ, $\alpha = 0.4$. Only LS was able to find the best solutions on 184 (54.76%) of 336 small instances, considering all scenarios and α variations.

For large instances, best solutions were found in few iterations. M-CNS and LS found most best solutions. Indeed, both algorithms performed similarly in the same instances. One can refer to the Supporting Information for more details. Both M-VND and BRKGA found very few of best solutions. BRKGA presented a large variation in the solution quality. For the instances brg180 and p654, BRKGA obtained results particularly bad which are more the one order magnitude worse than the best results obtained for all scenarios and discount factors. Although M-VND found few best solutions, it was the sole algorithm to find the best solutions for 27 (12.5%) of 576 large instances, considering all scenarios and α's. LS presented very good results in both small and large instances. We argue again that these results are due to the hybrid algorithmic nature of LS.

Figure 7 shows performance profiles [18] for all algorithms. In performance profiles, the abscissa shows the time needed to reach a target solution value (in log scale), while the ordinate shows the cumulative probability to reach a target solution value for the given time in the abscissa. Each algorithm is characterized by a different performance profile curve made up of (time, cumulative probability) pairs, one for each execution of the algorithm on a particular instance. Runs that took over 3600 s are not shown in the figure. Therefore, the percentage of runs that concluded within the time limit can be seen as the intersection of the profile with the right-hand side of the figure.

The target values are the best solution for each instance and scenario. The solid red line with white circles represents the empirical cumulative probability of the Exact algorithm. The BRKGA is represented by the dashed blue line with triangles,

M-VND by the gree dotted line with crosses, M-CNS by the dashed mangenta line with asterisks, and LS by the grey loosy dashed line with diamonds. First, observe that LS produces results much later than the other algorithms even Exact. Indeed, LS uses the most of this time pre-processing the instance. LS found its first best result in 35 s (instance ulysses22, scenario SL, alpha 0.2). After that, LS shows a steep slope finding about 11% of the best results in 100 s and 22% in 1000 s. The Exact algorithm presents the fastest converge to good solutions with about 23% in 100 s. Among our heuristic approches, M-CNS presents the best convergence with 11% in 100 s. M-VND and BRKGA have similar convengence curves with only about 6% of best results in 100 s.

7. CONCLUDING REMARKS

We present three heuristics and an integer programming formulation with separation algorithms to solve a variant of the many-to-many hub location-routing problem, namely the MMpLHP. The first heuristic is a biased random-key genetic algorithm with a local search routine, while the others are a combination of multistart with variable neighborhood structures. The heuristics allows the computation of optimal solution for several instances and also provided a good cutoff when solving the integer linear formulation by the branch-and-cut algorithm.

Comparing those heuristics, that one based on variable neighborhood descent is the best, returning better solutions than the others and in a less computation time. It seems that the procedure used to codify/decodify the solution in the BRKGA does not provided good solutions. Future researches will focus on how represent chromosomes and decodify random-keys in a solution structure.

Although LocalSolver had computed good solutions, it finished its execution only after to reach the given time limit, that is, for all the instances its runtime was of 1 h, while the other algorithms finished much before for a lot of instances. Moreover, we noted that LocalSolver presented some difficulty for finding good solutions in low running times.

Observing the Exact algorithm, we note that there is room for improvements by considering new separation algorithms and valid inequalities based on those from the capacitated vehicle routing problem.

ACKNOWLEDGMENTS

The authors would like to thank CNPq, FAPESP, and FAPEG for financial support. This work of Carlos E. Andrade was done when he was a Ph.D. student at University of Campinas. This work of Mauricio G. C. Resende was done when he was employed at AT&T Labs Research.

REFERENCES

[1] C. E. Andrade, F.K. Miyazawa, and M.G.C. Resende, Evolutionary algorithm for the k-interconnected multi-depot multi-traveling salesmen problem, Proc 15th Ann Conference on Genetic and Evolutionary Computation, GECCO'13. ACM, New York, NY, 2013, pp. 463–470.

[2] C.E. Andrade, M.G.C. Resende, H.J. Karloff, and F.K. Miyazawa, Evolutionary Algorithms for Overlapping Correlation Clustering, Proc 2014 Conference on Genetic and Evolutionary Computation, GECCO'14, ACM, New York, NY, 2014, pp. 405–412.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, Concorde User Documentation, Concorde TSP Solver at http://www.math.uwaterloo.ca/tsp/concorde, Accessed on June 15th, 2015.

[4] S. Barreto, C. Ferreira, J. Paixão, and B.S. Santos, Using clustering analysis in a capacitated location-routing problem, Eur J Oper Res 179 (2007), 968–977.

[5] J.C. Bean, Genetic algorithms and random keys for sequencing and optimization, ORSA J Comput 2 (1994), 154–160.

[6] J.-M. Belenguer, E. Benavent, C. Prins, C. Prodhon, and R. W. Calvo, A branch-and-cut method for the capacitated location-routing problem, Comput Oper Res 38 (2011), 931–941.

[7] S. Belhaiza, P. Hansen, and G. Laporte, A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows, Comput Oper Res 52 Part B (2014), 269–281.

[8] Y. Benjamini and Y. Hochberg, Controlling the false discovery rate: A practical and powerful approach to multiple testing, J R Stat Soc Series B Methodol 57 (1995), 289–300.

[9] M. Birattari, Z. Yuan, P. Balaprakash, and T. Stützle, "F-Race and iterated F-Race: An overview," Experimental methods for the analysis of optimization algorithms, Springer Berlin Heidelberg, 2010, pp. 311–336.

[10] A. Bruns, A. Klose, and P. Stähly, Restructuring of Swiss parcel delivery services, OR Spektrum 22 (2000), 285–302.

[11] R.S. Camargo, G. Miranda, and A. Lokketangen. A new formulation and an exact approach for the many-to-many hub location-routing problem. Appl Math Model 37 (2013), 7465–7480.

[12] S. Çetiner, C. Sepil, and H. Süral, Hubbing and postal routing in postal delivery systems. Ann Oper Res 181 (2010), 109–124.

[13] W.J. Conover, Practical nonparametric statistics, 2nd edition, Wiley, 1980.

[14] I. Contreras, "Hub location problems," Location Science, G. Laporte, S. Nickel, and F. Saldanha da Gama (Editors), Springer International Publishing, Switzerland, 2015, pp. 311–344.

[15] I. Contreras, E. Fernández, and A. Marín, Tight bounds from a path based formulation for the tree of hub location problem. Comput Oper Res 36 (2009), 3117–3127.

[16] A. Corberán, E. Mota, and J.J. Salazar, Some recent contributions to routing and location problems. Networks 42 (2003), 109–113.

[17] B. Dezs, A. Jüttner, and P. Kovács. LEMON – an open source C++ graph template library. Electron Notes Theor Comput Sci 264 (2011), 23–45.

[18] E.D. Dolan and J.J. Moré, Benchmarking optimization software with performance profiles. Math Program 91 (2002), 201–213.

[19] M. Ericsson, M. G. C. Resende and P. M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. J Comb Optim 6 (2002), 299–333.

[20] C.-H. Fang, S.-X. Li, and Y.-F. Wu, "Multiple allocation hub location problem with flow-dependent set-up cost." Proc 6th Intl Asia Conference on Industrial Engineering and Management Innovation: Core Theory and Applications of Industrial Engineering (volume 1), E. Qi (Editor), Atlantis Press, Paris, 2016, pp. 671–678.

[21] M.P. Fay and M.A. Proschan, Wilcoxon-Mann-Whitney or t-test?, On assumptions for hypothesis tests and multiple interpretations of decision rules. Stat Surv 4 (2010), 1–39.

[22] M.R. Garey and D.S. Johnson, Computers and intractability: A guide to the theory of $\mathcal{NP}$-completeness. Freeman, San Francisco, 1979.

[23] N. Ghaffari-Nasab, M. Ghazanfari, and E. Teimoury, Hub-and-spoke logistics network design for third party logistics service providers. Int J Manage Sci Eng Manage 11 (2016), 49–61.

[24] D.E. Goldberg, Genetic algorithms in search, optimization and machine learning, 1st edition, Addison-Wesley Professional, Boston, MA, 1989.

[25] R.E. Gomory and T.C. Hu, Multi-terminal network flow, J Soc Ind Appl Math 9 (1961), 551–570.

[26] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende, A random key based genetic algorithm for the resource constrained project scheduling problems. Comput Oper Res 36 (2009), 92–109.

[27] J.F. Gonçalves and J. Almeida, A hybrid genetic algorithm for assembly line balancing. J Heuristics 8 (2002), 629–642.

[28] J.F. Gonçalves and M.G.C. Resende, Biased random-key genetic algorithms for combinatorial optimization, J Heuristics 17 (2011), 487–525.

[29] J.F. Gonçalves and M.G.C. Resende, A parallel multi-population genetic algorithm for a constrained two-dimensional orthogonal packing problem. J Comb Optim 22 (2011), 180–201.

[30] P. Hansen, N. Mladenovic, and J.A.M. Perez, Variable neighborhood search: methods and applications. Ann Oper Res 175 (2010), 367–407.

[31] Y. He, T. Wu, C. Zhang, and Z. Liang, An improved MIP heuristic for the intermodal hub location problem. Omega 57 Part B (2015), 203–211.

[32] K. Helsgaun, An effective implementation of the Lin-Kernighan traveling salesman heuristic. Eur J Oper Res 126 (2000), 106–130.

[33] A. Ilić, D. Urošević, J. Brimberg, and N. Mladenović. A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem. Eur J Oper Res 206 (2010), 289–300.

[34] Innovation 24. LocalSolver. Available at: http://www.localsolver.com, 2016. Accessed on May 18th, 2016.

[35] B. Jarboui, H. Derbel, S. Hanafi, and N. Mladenovic, Variable neighborhood search for location routing. Comput Oper Res 40 (2013), 47–57.

[36] I. Karaoglan, F. Altiparmak, I. Kara, and B. Dengiz, The location-routing problem with simultaneous pickup, delivery: Formulations, a heuristic approach, Omega 40 (2012), 465–477.

[37] M.J. Kuby and R.G. Gray, The hub network design problem with stopovers and feeders: The case of federal express, TOP 27 (1993), 1–12.

[38] M. Labbé, I. Rodríguez-Martín, and J.J. Salazar-González, A branch-and-cut algorithm for the plant-cycle location problem. J Oper Res Soc 55 (2004), 513–520.

[39] M. Labbé and H. Yaman. Solving the hub location problem in a star-star network. Networks 51 (2008), 19–33.

[40] D. Levine, Application of a hybrid genetic algorithm to airline crew scheduling, Comput Oper Res 23 (1996), 547–558.

[41] C.C. Lin, J.Y. Lin, and Y.C. Chen, The capacitated p-hub median problem with integral constraints: An application to a Chinese air cargo network. Appl Math Model 36 (2012), 2777–2787.

[42] S. Lin and B.W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. Oper Res 21 (1973), 498–516.

[43] S.P. Lloyd. Least squares quantization in pcm. IEEE Trans Inform Theory 28 (1982), 129–137.

[44] C.M. Lopes, A.T. de Queiroz, E.C. de Andrade, and F. Miyazawa, Solving a variant of the hub location-routing problem. LISS 2014, Proc 4th Intl Conf on Logistics, Informatics and Service Science, Z. Zhang, M. Z. Shen, J. Zhang, and R. Zhang (Editors), Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 395–400.

[45] R.B. Lopes, C. Ferreira, B.S. Santos, and S. Barreto, A taxonomical analysism, current methods and objectives on location-routing problems. Int Trans Oper Res 20 (2013), 795–822.

[46] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Birattari, The irace package, Iterated Race for Automatic Algorithm Configuration. techreport TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium. Available at: http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2011-004.pdf, 2011. Accessed on June 15th, 2015.

[47] J. Lysgaard, A.N. Letchford and R.W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. Math Program Series A and B 100 (2004), 423–445.

[48] O. Martin, S.W. Otto and E.W. Felten. Large-step Markov chains for the TSP incorporating local search heuristics. Oper Res Lett 11 (1992), 219–224.

[49] H. Min, V. Jayaramanb, and R. Srivastava, Combined location-routing problems: A synthesis and future research directions. Eur J Oper Res 108 (1998), 1–15.

[50] N. Mladenović, M. Labbé and P. Hansen. Solving the p-center problem with tabu search and variable neighborhood search, Networks 42 (2003), 48–64.

[51] N. Mladenović, R. Todosijević, and D. Urošević. Two level General variable neighborhood search for Attractive traveling salesman problem. Comp Oper Res 52 Part B (2014), 341–348.

[52] N. Mladenović, D. Urošević, S. Hanafi, and A. Ilić, A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem, Eur J Oper Res 220 (2012), 270–285.

[53] N. Mladenović and P. Hansen, Variable neighborhood search, Comp Oper Res 24 (1997), 1097–1100.

[54] D. Naddef and G. Rinaldi, Branch-and-cut algorithms for the capacitated VRP, P. Toth and D. Vigo (Editors), The Vehicle Routing Problem. SIAM Monographs Disc Math Appl 9 (2001), 53–84.

[55] G. Nagy and S. Salhi, The many-to-many location-routing problem. TOP 6 (1998), 261–275.

[56] G. Nagy and S. Salhi, Location-routing: Issues, models and methods. Eur J Oper Res 177 (2007), 649–672.

[57] M.E. O'Kelly. A quadratic integer program for the location of interacting hub facilities. Eur J Oper Res 32 (1987), 393–404.

[58] J. Pacheco, I. García, and A. Álvarez. Enhancing variable neighborhood search by adding memory: Application to a real logistic problem. Knowledge-Based Syst 62 (2014), 28–37.

[59] Y. Rahmani, W.R. Cherif-Khettaf, and A. Oulamara, A local search approach for the two–echelon multi-products location–routing problem with pickup and delivery, IFAC-PapersOnLine, 48 (2015), 193–199.

[60] S. Rastani, M. Setak, and H. Karimi. Capacity selection for hubs and hub links in hub location problems. Int J Manage Sci Eng Manage 11 (2016), 123–133.

[61] J. Reese. Solution methods for the *p*-median problem: An annotated bibliography. Networks 48 (2006), 125–142.

[62] M.G.C. Resende, R.F. Toso, J.F. Gonçalves, and R.M.A. Silva, A biased random-key genetic algorithm for the Steiner triple covering problem. Optim Lett 6 (2012), 605–619.

[63] S.A.S. Rezazad, M. Aminnayeri, and H. Karimi, A congested double path approach for a hub location-allocation problem with service level at hubs. IJE Trans A: Basics 28 (2015), 1049–1058.

[64] J. Rieck, C. Ehrenberg, and J. Zimmermann. Many-to-many location-routing with inter-hub transport and multi-commodity pickup-and-delivery. Eur J Oper Res 236 (2014), 863–878.

[65] I. Rodríguez-Martín, J.-J. Salazar-González, and H. Yaman. A branch-and-cut algorithm for the hub location and routing problem. Comput Oper Res 50 (2014), 161–174.

[66] S. Salhi, A. Imran, and N. A. Wassan, The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation. Comput Oper Res 52 Part B (2014), 315–325.

[67] F. Samanlioglu, M.B. Kurz, W.G. Ferrell, and S. Tangudu, A hybrid random-key genetic algorithm for a symmetric travelling salesman problem. Int J Oper Res 2 (2006), 47–63.

[68] M. Schwengerer, S. Pirkwieser, and G.R. Raidl, A variable neighborhood search approach for the two-echelon location-routing problem. Evolution Comp Comb Optim 7245 (2012), 13–24.

[69] L.V. Snyder and M.S. Daskin, A random-key genetic algorithm for the generalized traveling salesman problem. Eur J Oper Res 174 (2006), 38–53.

[70] W.M. Spears and K.A. DeJong, On the virtues of parameterized uniform crossover. Proc 4th Intl Conf on Genetic Algorithms, 1991, pp. 230–236.

[71] R. Tinós, D. Whitley, and G. Ochoa. Generalized asymmetric partition crossover (GAPX) for the asymmetric TSP. Proc 2014 Conference on Genetic and Evolutionary Computation, GECCO '14. ACM, New York, NY, 2014, pp. 501–508.

[72] Z. Wang, C. Lin, and C.K. Chan, Demonstration of a single-fiber self-healing CWDM metro access ring network with unidirectional OADM, Photonics Technol Lett 18 (2006), 163–165.

[73] M. Wasner and G. Zaäpfel, An integrated multi-depot hublocation vehicle routing model for network planning of parcel service. Int J Prod Econom 90 (2004), 403–419.

[74] D. Whitley, S. Rana and R.B. Heckendorn. The island model genetic algorithm: On separability, population size and convergence. J Comp Inform Technol 7 (1998), 33–47.

[75] H. Yaman and S. Elloumi, Star *p*-hub center problem and star p-hub median problem with bounded path lengths. Comput Oper Res 39 (2012), 2725–2732.

SUPPLEMENTARY MATERIAL

A Best results for small instances

This section presents the best results obtained for small instances. The tables format is the following: the first column and second columns are the instance name and the best solution cost obtained for this instance, respectively. The columns named % show the percentage of the cost from the best solution obtained by the algorithm that names the column. A low percentage indicates that the obtained solution is closer to the best. A star ($\star$) indicates that the algorithm found the best solution. The columns t show the time in seconds to obtain that solution.

TABLE 3. Best results for ST scenario and $\alpha = 0.2$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	12245.20	0.03	11	7.25	152	0.02	113	⋆	338	⋆	597
bayg29	1807.40	⋆	1	⋆	108	⋆	0	⋆	0	⋆	11
bays29	2293.80	⋆	1	0.89	8	⋆	0	⋆	0	⋆	11
berlin52	8564.80	3.51	4	11.26	1014	⋆	114	⋆	334	⋆	11
brazil58	26192.40	1.43	24	16.60	1307	0.05	63	0.46	519	⋆	3819
burma14	3680.20	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	797.00	1.53	9	11.09	314	0.07	2	0.17	39	⋆	980
eil51	481.60	2.20	8	7.18	296	0.33	55	1.16	132	⋆	890
eil76	623.00	2.69	42	12.93	1409	1.15	3	2.31	115	⋆	22
fri26	956.60	⋆	0	⋆	33	⋆	0	⋆	0	⋆	11
gr17	1997.20	⋆	0	⋆	3	⋆	0	⋆	0	⋆	11
gr21	3074.00	⋆	0	⋆	2	⋆	12	⋆	8	⋆	11
gr24	1469.60	⋆	0	⋆	1	⋆	13	⋆	9	⋆	11
gr48	5667.60	1.50	74	3.00	195	⋆	35	0.29	71	⋆	146
gr96	64400.80	5.40	401	13.78	3266	3.13	603	3.31	543	⋆	270
hk48	13492.20	0.32	45	6.18	1022	⋆	0	⋆	15	⋆	90
kroA100	26281.40	6.92	204	44.50	2172	5.53	110	6.63	927	⋆	3616
kroB100	27320.00	6.15	89	22.60	3016	5.09	103	6.67	367	⋆	3932
kroC100	26484.60	8.29	163	17.78	3567	6.79	976	6.06	651	⋆	292
kroD100	26432.40	7.54	81	17.98	3245	5.49	653	3.59	317	⋆	1464
kroE100	27486.40	9.34	366	24.17	3196	3.13	279	3.49	191	⋆	247
pr76	121663.80	6.03	41	23.04	2438	2.09	51	4.18	148	⋆	1735
rat99	1505.00	4.15	186	11.97	3352	3.80	186	3.96	289	⋆	552
rd100	9779.60	4.52	258	13.85	2687	3.55	54	4.09	63	⋆	1487
st70	790.40	3.18	44	13.71	1098	3.31	112	0.40	254	⋆	304
swiss42	1475.80	0.06	25	3.03	401	0.06	465	0.06	40	⋆	22
ulysses16	7766.20	⋆	0	⋆	1	⋆	41	⋆	10	⋆	11
ulysses22	7165.60	⋆	0	⋆	102	⋆	132	⋆	13	⋆	11

TABLE 4. Best results for ST scenario and $\alpha = 0.4$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	13222.40	0.07	103	4.50	796	⋆	101	0.02	130	⋆	2321
bayg29	1916.00	⋆	0	⋆	12	⋆	0	⋆	0	⋆	11
bays29	2429.80	⋆	0	⋆	15	⋆	0	⋆	0	⋆	11
berlin52	9154.60	2.13	124	4.99	513	⋆	88	0.08	197	⋆	236
brazil58	28735.80	1.92	166	8.95	3149	0.21	349	0.44	242	⋆	3008
burma14	3891.40	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	867.00	1.20	10	7.68	302	⋆	96	0.32	220	⋆	326
eil51	509.20	1.53	11	8.05	665	0.62	68	1.29	131	⋆	642
eil76	658.60	3.55	36	10.41	1442	1.60	768	1.85	164	⋆	304
fri26	1038.20	0.19	0	⋆	47	⋆	0	⋆	7	⋆	11
gr17	2139.80	⋆	0	⋆	1	⋆	0	⋆	0	⋆	11
gr21	3314.40	⋆	0	⋆	2	⋆	42	⋆	12	⋆	11
gr24	1542.20	⋆	0	⋆	24	⋆	75	⋆	14	⋆	11
gr48	6031.20	0.76	106	3.70	317	⋆	53	⋆	13	⋆	45
gr96	68699.00	5.67	271	14.06	1751	3.26	209	5.43	931	⋆	2907
hk48	14358.40	⋆	24	1.61	716	⋆	13	⋆	23	⋆	33
kroA100	28326.40	8.19	64	41.92	3575	5.83	138	6.43	529	⋆	281
kroB100	29295.80	6.44	330	16.16	2726	5.04	239	5.04	199	⋆	518
kroC100	28320.60	4.80	110	18.53	3229	2.63	941	4.16	987	⋆	1769
kroD100	28024.80	7.16	342	27.70	3357	4.67	198	6.64	457	⋆	3132
kroE100	29431.20	5.00	326	18.02	3544	5.91	1409	4.23	689	⋆	123
pr76	130948.00	5.16	37	18.38	3139	0.11	476	2.27	298	⋆	1600
rat99	1583.80	6.45	138	11.68	2825	4.21	40	3.75	152	⋆	1926
rd100	10403.40	3.61	311	16.62	3258	4.49	574	4.45	334	⋆	1352
st70	840.80	3.30	8	9.44	1468	1.87	241	2.64	31	⋆	259
swiss42	1568.60	0.40	4	4.36	420	0.40	41	0.40	32	⋆	90
ulysses16	8073.40	⋆	0	⋆	0	⋆	10	⋆	7	⋆	11
ulysses22	7544.20	⋆	0	⋆	7	⋆	13	⋆	9	⋆	11

TABLE 5. Best results for ST scenario and $\alpha = 0.6$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	14182.80	0.47	6	1.20	571	⋆	87	0.11	57	⋆	676
bayg29	2014.00	⋆	0	⋆	11	⋆	0	⋆	0	⋆	11
bays29	2545.20	⋆	0	⋆	55	⋆	0	⋆	0	⋆	11
berlin52	9744.40	1.51	99	6.69	1537	⋆	3	0.17	91	⋆	416
brazil58	31113.20	0.57	75	17.31	2860	0.21	324	⋆	309	⋆	101
burma14	4102.60	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	936.60	1.15	37	4.07	541	⋆	7	⋆	27	⋆	11
eil51	536.80	1.30	67	6.25	376	1.00	43	0.44	41	⋆	811
eil76	687.80	3.80	26	9.33	1179	2.61	248	3.05	14	⋆	664
fri26	1119.80	0.26	8	⋆	22	⋆	0	⋆	31	⋆	11
gr17	2270.20	⋆	0	⋆	6	⋆	0	⋆	0	⋆	11
gr21	3518.60	⋆	1	⋆	3	⋆	42	⋆	12	⋆	11
gr24	1614.80	⋆	0	⋆	67	⋆	106	⋆	13	⋆	22
gr48	6394.80	0.56	10	1.52	705	⋆	12	0.10	36	⋆	912
gr96	73446.00	4.42	201	13.12	2633	2.69	10	3.46	508	⋆	630
hk48	15224.60	⋆	87	0.48	691	⋆	22	⋆	8	⋆	45
kroA100	30105.00	7.48	81	87.19	0	6.32	1430	3.46	183	⋆	90
kroB100	31161.20	4.66	228	18.15	2420	3.75	498	2.95	735	⋆	2061
kroC100	29997.80	7.64	125	20.91	3443	5.79	111	5.37	778	⋆	1847
kroD100	30000.20	8.91	249	25.52	2188	4.93	201	5.59	172	⋆	2681
kroE100	31178.20	5.06	82	17.06	2409	3.36	868	3.83	2	⋆	946
pr76	139383.00	3.75	77	14.93	3243	1.19	274	3.01	164	⋆	2174
rat99	1683.80	5.92	52	5.28	3258	3.59	353	1.06	91	⋆	552
rd100	11027.80	8.06	35	10.04	2523	4.70	470	3.36	86	⋆	969
st70	891.20	3.97	30	11.75	358	2.71	197	0.11	377	⋆	11
swiss42	1661.20	0.55	4	3.49	412	0.55	38	0.55	10	⋆	78
ulysses16	8344.60	⋆	0	⋆	0	⋆	38	⋆	10	⋆	11
ulysses22	7849.80	⋆	0	⋆	6	⋆	63	⋆	13	⋆	11

TABLE 6. Best results for ST scenario and $\alpha = 0.8$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	15115.00	⋆	91	4.90	552	⋆	63	⋆	46	⋆	338
bayg29	2112.00	⋆	0	⋆	16	⋆	0	⋆	0	⋆	11
bays29	2654.40	⋆	0	⋆	12	⋆	0	⋆	0	⋆	11
berlin52	10321.40	0.35	179	4.79	859	0.12	4	0.02	182	⋆	2704
brazil58	33009.20	0.76	87	11.58	872	0.18	175	0.07	203	⋆	608
burma14	4270.80	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	997.20	0.36	17	1.72	651	0.36	118	⋆	32	⋆	45
eil51	561.40	0.74	38	2.99	372	0.74	20	⋆	153	⋆	326
eil76	730.80	2.51	96	6.89	1366	0.13	95	0.76	431	⋆	540
fri26	1201.40	0.33	1	⋆	25	⋆	0	⋆	10	⋆	11
gr17	2392.60	⋆	0	⋆	6	⋆	0	⋆	0	⋆	11
gr21	3715.80	⋆	0	⋆	2	⋆	32	⋆	12	⋆	11
gr24	1676.60	⋆	0	⋆	2	⋆	53	⋆	23	⋆	11
gr48	6747.80	0.53	17	1.46	387	⋆	38	⋆	14	⋆	169
gr96	77671.60	5.76	204	13.13	2385	2.59	499	3.10	133	⋆	1307
hk48	15995.00	0.06	61	2.03	759	⋆	1	⋆	96	⋆	90
kroA100	31850.60	7.87	392	81.20	400	4.45	254	6.37	217	⋆	3301
kroB100	33131.00	6.31	262	10.31	2672	4.42	209	3.57	218	⋆	1025
kroC100	31866.20	7.23	144	16.60	1991	4.94	48	4.98	699	⋆	1138
kroD100	31753.80	7.58	392	24.45	2849	4.01	611	4.69	304	⋆	1284
kroE100	33088.60	2.71	141	17.18	2454	3.93	1575	3.59	260	⋆	338
pr76	148434.80	2.24	74	10.71	3287	1.57	50	0.79	105	⋆	2907
rat99	1794.80	1.77	355	4.69	2907	1.39	8	2.75	74	⋆	315
rd100	11553.40	7.08	94	11.52	3238	3.96	135	4.88	34	⋆	2433
st70	941.60	2.33	92	7.83	810	0.78	232	1.23	275	⋆	1273
swiss42	1753.60	0.24	15	1.96	411	0.24	36	0.24	13	⋆	22
ulysses16	8615.80	⋆	0	⋆	0	⋆	33	⋆	10	⋆	11
ulysses22	8155.40	⋆	0	⋆	6	⋆	64	⋆	13	⋆	11

TABLE 7. Best results for SL scenario and $\alpha = 0.2$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	9775.20	3.18	142	5.14	390	0.65	307	1.88	189	⋆	2749
bayg29	1490.60	3.87	21	1.28	18	⋆	5	1.65	16	⋆	135
bays29	1817.20	3.58	50	1.87	93	0.15	7	3.58	41	⋆	101
berlin52	6700.80	8.91	123	6.95	572	1.86	93	6.36	74	⋆	3346
brazil58	20243.00	11.20	98	6.68	2379	3.91	699	7.86	454	⋆	259
burma14	2832.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	649.20	3.20	68	3.91	136	1.66	25	2.83	46	⋆	2974
eil51	408.00	4.80	165	3.62	323	0.49	258	3.92	201	⋆	3695
eil76	508.40	7.59	151	12.62	1147	4.40	815	6.56	189	⋆	2636
fri26	800.80	4.87	17	⋆	8	⋆	13	0.69	30	⋆	11
gr17	1661.80	3.88	0	⋆	0	⋆	0	3.88	0	⋆	11
gr21	2402.00	0.91	17	⋆	1	⋆	39	⋆	21	⋆	169
gr24	1163.60	0.37	5	⋆	6	⋆	40	⋆	18	⋆	33
gr48	4674.60	5.05	2	4.98	450	2.55	57	3.83	93	⋆	3380
gr96	51884.20	13.63	37	11.79	0	6.17	1548	12.69	618	⋆	1047
hk48	10630.20	6.19	7	2.42	688	2.99	527	4.24	363	⋆	1915
kroA100	21377.20	6.75	9	10.11	0	5.93	687	4.48	736	⋆	4056
kroB100	21969.80	8.49	220	23.33	0	4.44	1970	4.30	305	⋆	405
kroC100	21802.40	3.88	314	12.25	0	3.06	9	2.31	1418	⋆	1611
kroD100	20576.40	8.14	199	43.80	0	5.92	1494	5.66	324	⋆	191
kroE100	22455.00	6.44	333	9.27	0	3.31	601	3.81	214	⋆	3819
pr76	100117.60	15.66	24	18.08	994	3.34	823	12.76	42	⋆	597
rat99	1245.00	5.25	314	34.61	465	3.59	856	2.47	451	⋆	2264
rd100	8040.80	5.79	357	39.59	0	3.39	1454	2.39	6	⋆	3414
st70	645.80	7.52	20	7.71	829	7.52	360	7.52	361	⋆	2129
swiss42	1198.60	8.76	12	4.80	392	3.63	183	6.47	50	⋆	1442
ulysses16	4433.80	0.74	4	⋆	0	⋆	24	1.00	10	⋆	11
ulysses22	4114.40	21.79	3	⋆	2	⋆	66	1.20	35	⋆	11

TABLE 8. Best results for SL scenario and $\alpha = 0.4$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	10582.60	2.35	103	0.97	479	1.86	781	1.82	137	⋆	3481
bayg29	1607.40	1.24	11	⋆	57	⋆	22	⋆	43	⋆	157
bays29	2013.80	1.23	6	⋆	23	⋆	41	1.23	10	⋆	214
berlin52	7340.40	7.63	192	6.92	1115	2.24	843	4.64	275	⋆	3470
brazil58	23287.40	8.19	85	11.01	1950	1.13	340	6.20	562	⋆	2276
burma14	3136.40	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	713.80	2.18	21	2.41	646	0.72	60	1.40	6	⋆	2388
eil51	441.00	1.63	133	0.22	159	1.40	409	0.36	386	⋆	743
eil76	559.40	4.04	8	7.32	947	3.11	514	2.00	766	⋆	270
fri26	896.80	3.97	41	0.29	49	⋆	23	⋆	4	⋆	11
gr17	1889.60	1.86	1	⋆	0	⋆	0	⋆	0	⋆	11
gr21	2677.00	2.54	5	⋆	3	⋆	38	⋆	21	⋆	67
gr24	1256.00	⋆	13	4.95	34	⋆	39	⋆	30	⋆	135
gr48	5078.40	3.17	153	4.34	377	1.52	31	2.59	242	⋆	2478
gr96	58584.00	9.71	258	8.31	0	3.54	1451	6.56	6	⋆	1498
hk48	11529.40	3.39	92	3.95	167	2.89	377	1.65	160	⋆	146
kroA100	23564.40	0.37	126	8.44	0	2.02	918	2.27	233	⋆	1611
kroB100	24093.40	5.58	228	20.36	0	3.34	1448	3.38	357	⋆	2873
kroC100	23757.80	3.58	268	10.85	0	4.12	1979	2.17	599	⋆	112
kroD100	22555.80	5.52	370	39.92	342	4.84	1673	4.25	471	⋆	1915
kroE100	24157.20	7.51	67	9.28	380	4.91	1279	3.01	727	⋆	146
pr76	113871.40	6.56	165	11.50	1448	4.02	558	6.79	503	⋆	3447
rat99	1325.00	4.99	190	34.50	0	3.44	1587	3.20	56	⋆	169
rd100	8570.00	4.61	141	39.24	0	5.10	817	3.74	1276	⋆	3740
st70	701.80	5.15	220	6.41	1213	5.52	313	5.15	618	⋆	3166
swiss42	1329.00	3.25	18	4.06	103	⋆	86	2.90	141	0.28	2005
ulysses16	5381.20	6.47	1	⋆	0	⋆	8	⋆	10	⋆	11
ulysses22	5085.80	7.51	41	⋆	1	⋆	66	⋆	10	⋆	11

TABLE 9. Best results for SL scenario and $\alpha = 0.6$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	11295.80	3.07	169	3.50	249	0.77	185	1.23	56	⋆	1926
bayg29	1686.60	⋆	0	⋆	4	⋆	28	⋆	27	⋆	78
bays29	2119.40	0.69	5	⋆	68	⋆	5	⋆	28	⋆	169
berlin52	7979.00	4.45	205	6.10	894	2.08	155	2.83	183	⋆	1476
brazil58	26094.40	6.78	104	14.19	956	1.13	287	4.34	426	⋆	969
burma14	3370.20	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	759.40	2.02	56	1.79	372	1.92	415	1.23	172	⋆	754
eil51	458.60	1.83	93	⋆	206	1.78	146	1.65	255	0.48	2647
eil76	596.00	2.48	286	18.55	0	1.84	344	0.93	176	⋆	2726
fri26	986.60	1.82	34	0.12	7	⋆	0	0.12	45	⋆	11
gr17	2066.60	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
gr21	2952.00	1.78	1	0.04	1	⋆	59	⋆	23	⋆	22
gr24	1334.20	0.13	14	2.59	63	⋆	39	⋆	19	⋆	56
gr48	5423.00	2.03	87	1.29	102	1.62	178	0.27	116	⋆	2140
gr96	63384.40	4.91	60	8.71	0	4.64	1235	5.08	565	⋆	1577
hk48	12221.60	3.44	2	2.40	265	2.53	642	0.57	28	⋆	1002
kroA100	24852.80	4.62	258	10.93	0	4.99	1726	3.46	382	⋆	3616
kroB100	25839.00	5.53	365	19.56	436	4.67	2220	4.02	1080	⋆	811
kroC100	25525.00	5.26	17	7.35	1335	3.80	1728	1.46	199	⋆	281
kroD100	24319.20	7.95	274	38.18	386	4.38	1236	2.80	2664	⋆	4033
kroE100	25693.40	6.92	26	10.80	0	5.81	72	4.81	56	⋆	1104
pr76	124700.00	3.89	291	8.40	1180	2.44	158	1.87	213	⋆	11
rat99	1411.60	4.54	210	32.65	1	3.79	926	2.36	477	⋆	642
rd100	9302.20	3.68	357	34.88	345	3.73	225	0.95	478	⋆	180
st70	752.80	4.72	383	5.95	1186	3.71	51	2.39	322	⋆	1915
swiss42	1432.20	2.59	53	1.14	396	0.60	116	0.68	14	⋆	1273
ulysses16	6295.80	⋆	1	⋆	0	⋆	59	⋆	38	⋆	11
ulysses22	6057.20	4.25	38	⋆	2	⋆	9	⋆	58	⋆	11

TABLE 10. Best results for SL scenario and $\alpha = 0.8$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
bayg29	1745.80	1.71	2	⋆	4	⋆	27	⋆	6	⋆	11
bays29	2197.00	⋆	42	⋆	14	⋆	55	⋆	8	⋆	56
berlin52	8526.20	3.97	83	0.55	2100	0.58	169	2.50	171	⋆	2174
brazil58	28961.40	3.59	1	7.14	1401	1.27	1701	1.15	328	⋆	890
burma14	3468.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	800.00	2.75	80	0.12	403	1.97	95	1.47	81	⋆	1036
eil51	473.80	2.23	145	⋆	197	3.33	567	0.92	14	0.84	1847
eil76	618.80	3.07	250	21.88	0	3.97	478	1.81	783	⋆	1340
fri26	1047.20	0.34	22	0.42	8	⋆	1	⋆	22	⋆	135
gr17	2128.80	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
gr21	3095.40	⋆	9	⋆	0	⋆	30	⋆	23	⋆	11
gr24	1398.00	⋆	8	2.60	1	⋆	47	⋆	19	⋆	56
gr48	5630.40	3.58	125	3.14	421	3.53	1	2.12	349	⋆	2952
gr96	67994.60	4.99	9	9.36	0	3.90	1046	4.93	947	⋆	3740
hk48	12938.20	2.64	65	2.22	524	2.77	269	0.39	32	⋆	2749
kroA100	26401.20	3.76	194	12.06	0	5.94	3403	3.14	657	⋆	67
kroB100	27375.60	4.95	366	19.83	0	6.02	910	3.99	712	⋆	2963
kroC100	26524.20	5.04	32	12.46	1	5.98	480	4.75	49	⋆	3109
kroD100	25872.60	5.42	309	38.63	0	5.59	161	4.20	316	⋆	2400
kroE100	26936.60	8.23	112	12.76	0	7.13	1541	5.96	1236	⋆	608
pr76	129003.00	5.88	334	17.32	0	4.69	1058	4.22	264	⋆	3064
rat99	1489.60	5.27	18	33.15	0	3.82	237	2.84	276	⋆	202
rd100	9765.20	4.08	271	36.71	0	2.97	894	2.87	1002	⋆	1273
st70	797.40	5.11	45	6.27	1466	3.46	263	3.28	278	⋆	3143
swiss42	1485.60	0.71	21	⋆	230	1.50	211	1.66	132	0.29	2997
ulysses16	6885.00	⋆	0	⋆	0	⋆	8	⋆	6	⋆	11
ulysses22	7028.60	1.58	1	1.58	1	⋆	56	⋆	58	⋆	11

TABLE 11. Best results for SQ scenario and $\alpha = 0.2$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	11822.00	0.24	38	5.83	560	0.01	3	⋆	54	⋆	518
bayg29	1687.80	0.91	15	0.91	83	⋆	1	0.02	16	⋆	33
bays29	2108.40	0.73	21	⋆	110	⋆	6	⋆	44	⋆	45
berlin52	7956.20	3.71	93	7.90	583	0.17	235	1.08	50	⋆	1960
brazil58	25898.40	5.27	102	12.41	1909	⋆	59	⋆	38	⋆	45
burma14	3480.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	749.00	0.16	81	8.46	379	⋆	38	0.02	92	⋆	11
eil51	457.20	1.31	21	5.86	192	0.91	84	0.83	108	⋆	2580
eil76	587.80	2.51	68	4.79	952	1.15	308	1.22	103	⋆	2298
fri26	906.20	0.77	3	⋆	34	⋆	0	⋆	3	⋆	11
gr17	1737.20	1.61	2	⋆	0	⋆	0	⋆	0	⋆	11
gr21	3074.00	⋆	2	⋆	3	⋆	12	⋆	12	⋆	11
gr24	1469.60	⋆	0	⋆	1	⋆	13	⋆	10	⋆	11
gr48	5683.20	⋆	34	3.32	725	⋆	9	⋆	6	⋆	22
gr96	64649.40	4.55	145	21.46	2227	1.49	632	4.60	375	⋆	2253
hk48	13022.80	0.04	71	1.10	352	⋆	20	⋆	40	⋆	11
kroA100	26266.80	9.19	339	3.69	3149	5.02	1194	2.47	801	⋆	3695
kroB100	26494.20	9.25	64	14.07	3600	5.71	1038	4.13	1238	⋆	1487
kroC100	25858.60	12.07	390	12.60	1499	4.83	211	4.67	609	⋆	2794
kroD100	25150.80	11.69	31	11.78	3266	5.39	410	6.12	968	⋆	315
kroE100	26999.80	10.03	223	10.26	3486	2.79	484	3.42	1077	⋆	1036
pr76	122930.20	3.59	195	11.24	1395	1.56	170	3.69	351	⋆	1938
rat99	1409.00	6.08	175	6.71	1892	3.36	92	3.05	1073	⋆	202
rd100	9587.60	8.89	289	9.37	1703	2.97	1061	4.38	1470	⋆	78
st70	743.60	4.38	13	6.34	396	4.38	360	4.38	186	⋆	202
swiss42	1425.20	1.60	36	6.21	280	0.50	170	1.47	141	⋆	33
ulysses16	7766.20	⋆	0	⋆	1	⋆	10	⋆	7	⋆	11
ulysses22	7165.60	⋆	0	⋆	20	⋆	13	⋆	9	⋆	11

TABLE 12. Best results for SQ scenario and $\alpha = 0.4$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	12629.80	0.84	31	4.80	329	⋆	10	0.03	13	⋆	33
bayg29	1816.60	0.25	2	0.25	22	⋆	4	⋆	5	⋆	202
bays29	2264.80	1.26	2	1.08	34	⋆	2	⋆	20	⋆	11
berlin52	8419.80	2.68	55	9.92	571	0.25	9	0.64	42	⋆	45
brazil58	28157.80	2.75	75	12.34	1624	⋆	199	0.10	76	⋆	597
burma14	3690.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	815.00	2.16	21	5.49	185	⋆	28	0.04	52	⋆	67
eil51	481.20	0.33	81	1.45	443	0.08	67	⋆	156	⋆	1374
eil76	615.40	1.59	20	3.25	1047	0.09	138	0.74	258	⋆	3301
fri26	988.40	0.04	12	⋆	73	⋆	0	⋆	7	⋆	11
gr17	2009.40	0.23	2	⋆	2	⋆	0	⋆	0	⋆	11
gr21	3314.40	⋆	0	⋆	7	⋆	12	⋆	10	⋆	11
gr24	1542.20	⋆	0	⋆	3	⋆	13	⋆	9	⋆	11
gr48	5971.40	⋆	12	1.85	415	⋆	3	⋆	3	⋆	56
gr96	68702.60	6.19	291	12.12	1972	0.53	1130	0.28	1259	⋆	2073
hk48	13719.60	0.01	47	1.96	746	⋆	14	0.06	60	⋆	45
kroA100	27816.00	3.53	25	2.45	2471	3.30	2061	3.86	2484	⋆	3380
kroB100	27893.40	6.43	388	7.39	2048	4.79	159	6.25	1977	⋆	1352
kroC100	27439.40	7.59	317	11.79	2574	4.44	752	4.57	129	⋆	3808
kroD100	26246.40	11.50	214	11.81	1242	6.50	892	5.25	629	⋆	1430
kroE100	27914.40	6.30	96	13.70	2072	4.43	142	6.74	27	⋆	1183
pr76	128720.00	3.66	7	9.52	2322	⋆	129	3.05	35	0.09	2850
rat99	1496.80	3.42	27	4.47	2230	2.79	1584	1.96	295	⋆	3121
rd100	9922.00	10.36	404	8.84	2386	3.97	725	4.99	861	⋆	1036
st70	781.00	3.22	39	3.81	673	3.22	243	3.22	157	⋆	2929
swiss42	1526.40	0.26	22	4.97	170	⋆	34	0.07	123	⋆	90
ulysses16	8073.40	⋆	0	⋆	0	⋆	10	⋆	7	⋆	11
ulysses22	7544.20	⋆	0	⋆	9	⋆	13	⋆	9	⋆	11

TABLE 13. Best results for SQ scenario and $\alpha = 0.6$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	13436.20	0.88	49	2.68	617	⋆	156	⋆	77	⋆	270
bayg29	1905.80	⋆	1	⋆	65	⋆	0	⋆	5	⋆	11
bays29	2392.60	⋆	0	⋆	6	⋆	0	⋆	0	⋆	11
berlin52	8843.20	0.63	87	7.92	591	0.22	200	0.89	214	⋆	552
brazil58	30253.00	0.72	36	15.47	868	⋆	81	⋆	103	⋆	315
burma14	3900.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	881.00	0.61	14	1.93	271	⋆	5	0.13	2	⋆	11
eil51	497.20	1.85	4	⋆	166	0.92	31	⋆	36	⋆	2636
eil76	639.20	2.72	31	4.38	923	0.62	269	0.15	170	⋆	878
fri26	1062.60	⋆	0	⋆	9	⋆	0	⋆	0	⋆	11
gr17	2183.80	⋆	0	⋆	17	⋆	0	⋆	0	⋆	11
gr21	3518.60	⋆	1	⋆	7	⋆	12	⋆	8	⋆	11
gr24	1614.80	⋆	0	⋆	6	⋆	13	⋆	10	⋆	22
gr48	6259.60	⋆	2	2.80	349	⋆	1	⋆	42	⋆	11
gr96	72394.20	3.64	300	9.24	1369	0.51	852	0.28	1854	⋆	2501
hk48	14416.40	0.46	5	1.20	387	⋆	9	⋆	8	⋆	236
kroA100	29098.80	8.75	236	3.86	2237	3.66	330	5.14	58	⋆	1025
kroB100	29291.80	4.84	364	8.82	3390	4.23	261	3.44	628	⋆	3797
kroC100	28703.60	3.18	130	4.51	2836	3.57	1409	3.74	221	⋆	1002
kroD100	28073.60	6.99	378	5.39	2722	2.22	54	4.28	804	⋆	3774
kroE100	29548.40	6.65	63	7.93	3598	3.81	758	3.41	825	⋆	3076
pr76	133609.40	5.10	36	7.17	1282	0.37	266	2.28	38	⋆	822
rat99	1547.20	5.68	376	5.48	2424	1.88	806	2.33	397	⋆	2591
rd100	10486.60	8.00	209	6.73	2823	2.47	1153	2.42	1915	⋆	315
st70	820.00	3.22	172	5.22	748	3.22	306	3.22	262	⋆	2253
swiss42	1606.60	0.97	23	2.49	385	0.11	100	0.11	62	⋆	383
ulysses16	8344.60	⋆	0	⋆	0	⋆	10	⋆	7	⋆	11
ulysses22	7849.80	⋆	0	⋆	2	⋆	13	⋆	9	⋆	11

TABLE 14. Best results for SQ scenario and $\alpha = 0.8$.

Instance	Best	Exact		BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t	%	t
att48	14198.80	0.36	44	1.68	876	⋆	3	⋆	37	⋆	56
bayg29	1990.40	⋆	4	⋆	83	⋆	0	⋆	1	⋆	22
bays29	2490.00	⋆	6	⋆	102	⋆	8	⋆	0	⋆	101
berlin52	9266.60	1.70	48	8.92	640	0.21	24	0.21	48	⋆	112
brazil58	31401.00	1.12	115	8.76	761	0.53	62	⋆	197	⋆	33
burma14	4110.00	⋆	0	⋆	0	⋆	0	⋆	0	⋆	11
dantzig42	930.00	0.15	12	⋆	259	⋆	9	0.15	32	⋆	112
eil51	514.60	0.97	64	2.06	121	0.73	100	⋆	149	⋆	1859
eil76	660.00	3.63	0	3.60	1107	0.36	396	1.12	299	⋆	687
fri26	1106.80	⋆	0	⋆	7	⋆	0	⋆	0	⋆	11
gr17	2313.00	⋆	1	⋆	0	⋆	0	⋆	0	⋆	11
gr21	3715.80	⋆	0	⋆	2	⋆	12	⋆	8	⋆	11
gr24	1676.60	⋆	0	⋆	2	⋆	16	⋆	10	⋆	11
gr48	6519.40	⋆	57	4.46	278	⋆	3	⋆	5	⋆	101
gr96	75334.40	2.98	173	10.78	1522	0.18	1006	1.59	124	⋆	214
hk48	15088.20	0.46	15	1.61	121	0.01	18	0.05	6	⋆	67
kroA100	30115.80	4.84	342	5.92	2112	2.69	757	2.97	688	⋆	157
kroB100	30956.80	7.44	315	4.49	3480	4.25	1568	3.93	1639	⋆	416
kroC100	29752.00	5.77	21	11.98	2114	3.29	799	4.71	983	⋆	1971
kroD100	29356.00	9.10	109	7.67	2263	3.30	199	3.49	296	⋆	2016
kroE100	30912.40	7.89	116	8.66	3584	3.07	365	2.31	48	⋆	1036
pr76	139551.80	2.66	92	8.35	1458	0.94	133	2.90	128	⋆	1949
rat99	1631.40	1.49	247	5.02	2524	0.09	88	0.69	779	⋆	856
rd100	10796.40	7.35	351	6.31	1707	3.14	609	5.41	386	⋆	552
st70	860.00	2.41	78	4.32	649	2.41	245	2.41	182	⋆	360
swiss42	1668.00	1.15	6	2.67	263	1.12	111	⋆	106	⋆	338
ulysses16	8615.80	⋆	0	⋆	0	⋆	10	⋆	7	⋆	11
ulysses22	8155.40	⋆	0	⋆	4	⋆	13	⋆	9	⋆	11

B Best results for the large instances

This section presents the best results obtained for large instances. The tables format is the same of Appendix A. Note that the results for the exact algorithm are not shown since it was not able to improve the incumbent solution from the initial heuristic.

TABLE 15. Best results for ST scenario and $\alpha = 0.2$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3456.60	212.09	4	14.85	306	⋆	3662	⋆	3662
ali535	284620.20	348.22	13	21.11	344	⋆	3887	⋆	3887
att532	38018.40	126.00	3591	10.46	2947	⋆	1713	⋆	1713
bier127	141586.20	107.49	2	4.02	323	⋆	3741	⋆	3741
brg180	3010.00	10754.68	2405	42.26	48	⋆	2254	⋆	2254
ch130	7592.80	152.58	2	5.95	919	⋆	1758	⋆	1758
ch150	8563.80	119.83	2	5.43	781	⋆	3369	⋆	3369
d198	18703.00	182.00	3502	4.58	3542	⋆	2344	⋆	2344
d493	45693.20	222.70	10	12.30	2217	⋆	4045	⋆	4045
d657	68762.40	343.40	14	14.90	1845	⋆	3820	⋆	3820
eil101	731.60	14.65	3498	2.49	66	⋆	1780	⋆	1780
fl417	16529.00	954.98	21	12.26	3142	⋆	3572	⋆	3572
gil262	3134.60	195.73	4	6.81	2077	⋆	3606	⋆	3606
gr120	8395.80	22.69	3601	4.29	371	⋆	3831	⋆	3831
gr137	85267.80	168.78	640	3.43	456	⋆	406	⋆	406
gr202	48234.40	134.34	4	6.56	2009	⋆	1296	⋆	1296
gr229	170164.00	218.20	4	7.39	1920	⋆	2783	⋆	2783
gr431	222808.00	224.28	11	8.31	1396	⋆	3256	⋆	3256
gr666	426102.60	284.97	14	7.22	2264	⋆	3741	⋆	3741
kroA150	33138.40	202.66	3	9.41	643	⋆	203	⋆	203
kroA200	37508.20	193.74	4	12.73	3288	⋆	2039	⋆	2039
kroB150	32536.20	177.43	3498	6.56	1742	⋆	1296	⋆	1296
kroB200	38415.40	260.89	3	11.71	1597	⋆	2975	⋆	2975
lin105	18788.80	136.86	487	6.34	12	⋆	3301	⋆	3301
lin318	54942.80	321.50	5	14.81	731	⋆	3752	⋆	3752
p654	53781.20	1219.11	40	9.03	898	⋆	3234	⋆	3234
pa561	3738.20	271.30	10	21.70	1536	⋆	1296	⋆	1296
pcb442	69558.20	247.02	7	13.04	1227	⋆	3606	⋆	3606
pr107	43390.40	32.38	3588	1.84	518	⋆	124	⋆	124
pr124	76691.20	59.14	3561	5.41	123	⋆	654	⋆	654
pr136	114093.60	74.00	3570	2.77	1050	⋆	3876	⋆	3876
pr144	64519.80	342.55	2	15.28	597	⋆	113	⋆	113
pr152	77103.20	293.30	4	6.95	1226	⋆	845	⋆	845
pr226	96057.00	595.17	4	16.17	542	⋆	3876	⋆	3876
pr264	64883.80	460.27	3430	8.79	314	⋆	3561	⋆	3561
pr299	64681.20	313.97	5	15.03	672	⋆	4000	⋆	4000
pr439	152004.80	315.12	9	13.75	3582	⋆	2828	⋆	2828
rat195	3118.80	180.34	3	6.11	270	⋆	755	⋆	755
rat575	9303.40	304.19	10	19.03	3253	⋆	3831	⋆	3831
rat783	12688.20	309.25	15	14.22	1931	⋆	2377	⋆	2377
rd400	20643.60	222.85	6	17.96	196	⋆	3335	⋆	3335
si175	23189.40	43.26	3548	4.69	1414	⋆	2039	⋆	2039
si535	52952.80	123.89	3612	12.34	2808	⋆	2096	⋆	2096
ts225	189215.00	197.11	4	13.63	893	⋆	2028	⋆	2028
tsp225	4905.80	184.75	4	11.29	2579	⋆	417	⋆	417
u159	54892.00	183.13	3	9.24	377	⋆	3110	⋆	3110
u574	51889.80	323.94	11	17.73	42	⋆	3189	⋆	3189
u724	61544.60	384.44	14	11.60	2546	⋆	3797	⋆	3797

TABLE 16. Best results for ST scenario and $\alpha = 0.4$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3770.20	192.45	5	13.67	2329	⋆	2332	⋆	2332
ali535	319622.60	305.86	14	15.03	2580	⋆	4056	⋆	4056
att532	41466.40	317.67	11	7.83	628	⋆	1994	⋆	1994
bier127	149479.00	103.01	3	3.62	1182	⋆	2456	⋆	2456
brg180	3138.00	9745.06	2710	40.28	1218	⋆	1138	⋆	1138
ch130	8138.60	143.02	2	8.66	513	⋆	2085	⋆	2085
ch150	9053.60	114.37	2	5.87	413	⋆	800	⋆	800
d198	20303.00	270.49	3376	5.00	2406	⋆	4000	⋆	4000
d493	49251.20	205.24	9	8.93	454	⋆	3470	⋆	3470
d657	72890.40	324.36	13	13.62	1823	⋆	4056	⋆	4056
eil101	767.60	13.31	2532	1.64	2	⋆	2941	⋆	2941
fl417	18638.00	844.29	22	5.33	266	⋆	1713	⋆	1713
gil262	3457.00	174.51	4	4.59	3211	⋆	2490	⋆	2490
gr120	8882.00	18.51	3269	5.93	709	⋆	383	⋆	383
gr137	91550.40	142.77	2040	4.32	665	⋆	383	⋆	383
gr202	50338.20	129.81	4	5.67	1083	⋆	2377	⋆	2377
gr229	181780.80	204.26	4	7.58	3386	⋆	2006	⋆	2006
gr431	238486.00	209.17	10	6.49	1866	⋆	3223	⋆	3223
gr666	461821.20	260.89	16	5.76	2947	⋆	3527	⋆	3527
kroA150	35036.00	158.61	2959	9.51	712	⋆	2806	⋆	2806
kroA200	40385.60	179.60	3	10.26	40	⋆	293	⋆	293
kroB150	34810.60	140.77	3540	8.20	106	⋆	169	⋆	169
kroB200	41277.20	242.41	3	6.93	2077	⋆	2535	⋆	2535
lin105	20322.60	49.64	3577	2.98	468	⋆	79	⋆	79
lin318	61670.00	282.20	6	10.10	2958	⋆	3583	⋆	3583
p654	63213.00	1029.16	40	⋆	3047	3.13	3054	3.13	3054
pa561	4022.00	250.50	10	17.53	181	⋆	2783	⋆	2783
pcb442	74307.60	230.88	7	11.46	2568	⋆	3617	⋆	3617
pr107	48845.60	29.84	3585	1.22	1158	⋆	214	⋆	214
pr124	84036.00	40.77	3459	6.07	149	⋆	327	⋆	327
pr136	122812.60	26.85	3578	1.73	902	⋆	1701	⋆	1701
pr144	72286.40	307.36	2	13.08	1184	⋆	2806	⋆	2806
pr152	87625.00	257.61	3	4.23	934	⋆	3617	⋆	3617
pr226	107159.80	533.04	5	15.60	1441	⋆	3606	⋆	3606
pr264	74489.60	405.43	3192	1.53	1491	⋆	270	⋆	270
pr299	67673.60	302.87	5	14.41	2165	⋆	1217	⋆	1217
pr439	169286.60	279.76	10	8.94	2866	⋆	3910	⋆	3910
rat195	3304.80	169.92	3	4.25	961	⋆	2423	⋆	2423
rat575	10202.00	274.21	10	11.20	1812	⋆	3234	⋆	3234
rat783	13815.20	281.34	15	8.15	261	⋆	1532	⋆	1532
rd400	22061.60	208.33	6	14.93	873	⋆	1465	⋆	1465
si175	24230.40	32.78	3541	5.04	1638	⋆	1104	⋆	1104
si535	56353.80	111.46	3588	10.60	393	⋆	2321	⋆	2321
ts225	204668.80	181.50	3	11.92	3182	⋆	372	⋆	372
tsp225	5197.00	174.52	1050	10.06	3039	⋆	2434	⋆	2434
u159	59531.20	168.25	2	5.92	346	⋆	1037	⋆	1037
u574	54591.40	309.45	10	15.61	84	⋆	3842	⋆	3842
u724	64157.00	370.47	14	12.36	3442	⋆	3031	⋆	3031

TABLE 17. Best results for ST scenario and $\alpha = 0.6$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	4027.00	179.71	4	10.22	2638	⋆	3752	⋆	3752
ali535	333331.20	295.37	14	13.62	2420	⋆	1262	⋆	1262
att532	44580.00	294.05	11	2.54	1347	⋆	3403	⋆	3403
bier127	158014.20	98.18	3	3.37	591	⋆	710	⋆	710
brg180	3290.00	9886.14	3545	38.24	2595	⋆	2490	⋆	2490
ch130	8637.20	135.94	2	6.28	20	⋆	3865	⋆	3865
ch150	9687.80	106.30	2	3.52	1456	⋆	1172	⋆	1172
d198	22122.20	265.87	3553	4.33	1158	⋆	3606	⋆	3606
d493	52318.80	192.47	10	6.74	1217	⋆	3346	⋆	3346
d657	80512.40	289.76	12	7.58	2435	⋆	4056	⋆	4056
eil101	811.80	11.58	1636	2.73	202	⋆	2952	⋆	2952
fl417	20634.20	760.81	21	⋆	1670	12.10	2479	12.10	2479
gil262	3554.80	173.37	4	8.01	2169	⋆	1690	⋆	1690
gr120	9369.40	17.21	2474	3.85	486	⋆	473	⋆	473
gr137	96483.40	152.52	1325	4.87	81	⋆	1183	⋆	1183
gr202	53354.20	122.60	4	5.77	1270	⋆	3887	⋆	3887
gr229	191158.80	194.59	4	7.28	2458	⋆	4000	⋆	4000
gr431	260866.00	188.33	10	⋆	1404	2.99	4056	2.99	4056
gr666	505546.00	234.84	14	⋆	2093	1.19	3211	1.19	3211
kroA150	36724.60	185.35	3	11.17	885	⋆	2806	⋆	2806
kroA200	42756.80	170.51	3	10.98	2588	⋆	1566	⋆	1566
kroB150	36976.80	129.15	3601	8.03	217	⋆	2434	⋆	2434
kroB200	43821.80	210.28	3539	7.76	1830	⋆	2344	⋆	2344
lin105	21813.20	118.82	1128	4.03	219	⋆	1577	⋆	1577
lin318	66046.20	263.13	6	9.21	510	⋆	361	⋆	361
p654	63811.00	1025.41	40	1.50	708	⋆	3403	⋆	3403
pa561	4317.80	231.91	10	13.86	1717	⋆	2569	⋆	2569
pcb442	78893.20	217.44	7	9.61	2829	⋆	3606	⋆	3606
pr107	54267.00	36.19	3575	1.64	226	⋆	3820	⋆	3820
pr124	91520.20	36.46	3553	5.70	397	⋆	406	⋆	406
pr136	129916.00	25.53	3590	3.26	74	⋆	507	⋆	507
pr144	79953.20	279.68	3	9.29	213	⋆	1532	⋆	1532
pr152	97926.00	230.38	3	3.10	1079	⋆	687	⋆	687
pr226	125358.20	449.60	5	7.93	2128	⋆	428	⋆	428
pr264	76182.40	406.65	3590	6.42	3207	⋆	2648	⋆	2648
pr299	76571.60	262.09	5	7.32	546	⋆	1927	⋆	1927
pr439	177292.20	269.37	10	10.66	1171	⋆	3820	⋆	3820
rat195	3410.20	167.15	3	6.57	1626	⋆	1724	⋆	1724
rat575	10667.80	262.69	10	12.06	2669	⋆	3808	⋆	3808
rat783	15062.20	254.99	15	5.45	603	⋆	2073	⋆	2073
rd400	24118.80	187.77	7	10.57	2787	⋆	2017	⋆	2017
si175	25374.80	33.73	3555	4.14	2833	⋆	3977	⋆	3977
si535	58693.20	108.84	3574	10.55	976	⋆	3470	⋆	3470
ts225	221182.40	166.81	3	10.74	2499	⋆	2085	⋆	2085
tsp225	5671.20	157.86	3	5.89	1773	⋆	856	⋆	856
u159	63488.60	158.24	2	5.64	515	⋆	834	⋆	834
u574	60074.40	278.04	11	9.60	1488	⋆	3358	⋆	3358
u724	73406.00	316.25	14	4.21	1135	⋆	3561	⋆	3561

TABLE 18. Best results for ST scenario and $\alpha = 0.8$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
ali535	367866.20	264.48	13	9.00	514	⋆	2366	⋆	2366
att532	45182.60	294.43	10	10.15	3064	⋆	4056	⋆	4056
bier127	165915.40	94.57	2	2.31	32	⋆	1386	⋆	1386
brg180	3402.00	9428.28	3186	38.62	3158	⋆	282	⋆	282
ch130	9162.40	69.41	3594	5.50	339	⋆	3730	⋆	3730
ch150	9932.60	106.86	2	6.47	1114	⋆	518	⋆	518
d198	23604.60	-	-	5.41	11	⋆	1724	⋆	1724
d493	55891.00	178.86	10	4.03	2484	⋆	3301	⋆	3301
d657	88187.40	260.69	13	1.58	2518	⋆	496	⋆	496
eil101	847.00	14.57	2288	2.86	239	⋆	2524	⋆	2524
fl417	22734.00	688.42	21	⋆	178	3.11	3910	3.11	3910
gil262	3799.20	161.48	4	3.16	433	⋆	2130	⋆	2130
gr120	9802.60	16.11	3521	4.91	646	⋆	1442	⋆	1442
gr137	103978.00	128.11	1937	2.53	565	⋆	338	⋆	338
gr202	55956.60	116.60	4	4.08	1423	⋆	4034	⋆	4034
gr229	210525.60	172.82	5	1.58	1171	⋆	2783	⋆	2783
gr431	274111.80	179.86	10	2.43	2326	⋆	2648	⋆	2648
gr666	507272.00	239.18	14	⋆	1925	8.21	1307	8.21	1307
kroA150	39587.60	170.39	4	7.51	626	⋆	2163	⋆	2163
kroA200	45931.00	158.15	4	8.20	1383	⋆	845	⋆	845
kroB150	38839.00	142.19	3247	8.09	127	⋆	2208	⋆	2208
kroB200	45657.60	221.47	3	9.38	1414	⋆	3549	⋆	3549
lin105	23174.20	33.90	3538	3.79	325	⋆	608	⋆	608
lin318	70673.40	244.92	5	7.41	871	⋆	2366	⋆	2366
p654	72361.80	898.39	40	⋆	370	31.26	3741	31.26	3741
pa561	4515.80	222.09	10	13.47	2168	⋆	3403	⋆	3403
pcb442	89040.80	186.41	7	1.63	3110	⋆	3076	⋆	3076
pr107	59982.20	14.91	3204	0.14	316	⋆	3290	⋆	3290
pr124	98152.60	50.66	3562	6.56	829	⋆	1476	⋆	1476
pr136	135582.60	25.70	3448	3.88	438	⋆	2287	⋆	2287
pr144	87646.20	256.65	3	13.95	370	⋆	2321	⋆	2321
pr152	106650.80	212.60	3	3.47	630	⋆	879	⋆	879
pr226	124526.00	461.68	1382	13.35	1953	⋆	3662	⋆	3662
pr264	81327.20	381.90	3569	7.59	1499	⋆	2986	⋆	2986
pr299	83953.60	235.90	5	2.97	1107	⋆	2445	⋆	2445
pr439	190654.40	249.83	9	9.79	131	⋆	3121	⋆	3121
rat195	3618.40	153.24	1246	4.14	1794	⋆	1183	⋆	1183
rat575	11253.60	248.75	10	9.69	2030	⋆	2873	⋆	2873
rat783	15643.40	246.73	15	3.59	84	⋆	3966	⋆	3966
rd400	25887.80	173.40	6	5.40	2250	⋆	3583	⋆	3583
si175	26395.40	28.77	3547	3.30	1021	⋆	4000	⋆	4000
si535	62298.80	99.52	3562	8.20	392	⋆	1825	⋆	1825
ts225	231025.00	161.57	4	10.21	566	⋆	1825	⋆	1825
tsp225	5895.20	153.42	3	7.74	2449	⋆	1003	⋆	1003
u159	67702.20	148.48	3	4.83	1099	⋆	146	⋆	146
u574	64653.00	256.64	11	7.99	1084	⋆	2051	⋆	2051
u724	75214.40	311.10	14	6.09	1948	⋆	3459	⋆	3459

TABLE 19. Best results for SL scenario and $\alpha = 0.2$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	2923.80	14.48	4	7.00	2941	⋆	1927	⋆	1927
ali535	208952.40	106.82	13	15.79	2634	⋆	1330	⋆	1330
att532	30515.80	35.93	11	8.42	1896	⋆	3628	⋆	3628
bier127	95908.00	58.31	2	15.91	287	⋆	1758	⋆	1758
brg180	1826.00	5028.26	2530	37.68	1919	⋆	3662	⋆	3662
ch130	6101.20	6.62	1	6.53	772	⋆	3256	⋆	3256
ch150	6759.20	10.95	2	6.72	1523	⋆	3403	⋆	3403
d198	11870.60	-	-	8.05	2085	⋆	1499	⋆	1499
d493	34607.80	9.56	10	10.92	3066	⋆	1262	⋆	1262
d657	55580.20	84.90	12	7.05	3340	⋆	3448	⋆	3448
eil101	611.40	26.56	1	4.58	228	⋆	3177	⋆	3177
fl417	10079.40	299.01	20	21.55	1859	⋆	3549	⋆	3549
gil262	2526.80	19.82	3	8.96	1805	⋆	1724	⋆	1724
gr120	6801.60	26.19	1	2.56	205	⋆	2704	⋆	2704
gr137	68186.20	42.17	2	8.12	1462	⋆	270	⋆	270
gr202	32794.20	67.55	4	19.97	717	⋆	2479	⋆	2479
gr229	130170.40	52.93	5	10.81	2977	⋆	3955	⋆	3955
gr431	169197.60	37.47	10	12.48	3058	⋆	1183	⋆	1183
gr666	310176.60	65.57	13	13.61	2264	⋆	1093	⋆	1093
kroA150	27066.40	16.10	2	3.66	3539	⋆	868	⋆	868
kroA200	31202.80	13.38	4	4.06	361	⋆	642	⋆	642
kroB150	26558.00	7.05	2	5.06	1815	⋆	2411	⋆	2411
kroB200	30726.40	72.50	3	7.03	2419	⋆	2062	⋆	2062
lin105	13665.80	46.68	1	9.47	484	⋆	135	⋆	135
lin318	43504.00	87.26	4	14.52	3392	⋆	3403	⋆	3403
p654	37606.80	271.88	39	⋆	2175	4.50	3989	4.50	3989
pa561	3172.60	63.09	9	14.92	1646	⋆	3797	⋆	3797
pcb442	60690.40	35.15	6	2.66	427	⋆	3324	⋆	3324
pr107	31473.80	143.55	3473	6.79	1948	⋆	3583	⋆	3583
pr124	58246.00	189.60	2	4.20	359	⋆	68	⋆	68
pr136	88301.20	24.85	2	5.88	1654	⋆	1769	⋆	1769
pr144	42582.00	55.16	2	27.72	1627	⋆	2794	⋆	2794
pr152	55053.00	150.65	3	0.57	229	⋆	1285	⋆	1285
pr226	70846.60	414.35	4	14.44	1275	⋆	901	⋆	901
pr264	44709.00	191.26	9	13.03	3234	⋆	1600	⋆	1600
pr299	53916.80	120.57	4	7.58	456	⋆	3346	⋆	3346
pr439	110594.20	28.81	8	12.81	1351	⋆	744	⋆	744
rat195	2588.40	8.63	4	2.08	714	⋆	2772	⋆	2772
rat575	8054.40	18.62	9	1.50	457	⋆	1724	⋆	1724
rat783	10889.60	3.23	13	2.52	2418	⋆	3662	⋆	3662
rd400	16930.80	35.37	5	7.67	461	⋆	3944	⋆	3944
si175	20523.80	44.28	2571	5.60	3278	⋆	2682	⋆	2682
si535	46078.00	118.09	3418	15.09	963	⋆	3718	⋆	3718
ts225	165665.80	3.83	3	4.14	134	⋆	2073	⋆	2073
tsp225	4161.00	41.95	3	3.85	1995	⋆	451	⋆	451
u159	44265.00	13.93	3	6.60	2358	⋆	1848	⋆	1848
u574	43541.60	40.02	10	2.64	3243	⋆	3380	⋆	3380
u724	50707.20	26.84	12	2.05	3338	⋆	3651	⋆	3651

TABLE 20. Best results for SL scenario and $\alpha = 0.4$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3284.60	9.07	3	1.78	3529	⋆	3628	⋆	3628
ali535	258673.00	75.44	13	⋆	1308	3.13	1656	3.13	1656
att532	35795.80	22.69	10	⋆	1599	4.53	3842	4.53	3842
bier127	113243.40	46.27	2	10.32	365	⋆	755	⋆	755
brg180	2012.00	4455.96	2896	37.38	437	⋆	800	⋆	800
ch130	6738.40	5.26	1	3.74	2184	⋆	2163	⋆	2163
ch150	7515.00	7.49	2	3.33	909	⋆	394	⋆	394
d198	13960.40	327.32	2870	9.00	2947	⋆	3673	⋆	3673
d493	39454.80	5.03	9	5.67	2530	⋆	4011	⋆	4011
d657	62664.40	71.75	11	1.93	777	⋆	2986	⋆	2986
eil101	665.20	22.91	1	2.44	245	⋆	3707	⋆	3707
fl417	13018.60	221.33	20	⋆	3640	23.20	2107	23.20	2107
gil262	2825.40	15.01	3	4.65	3346	⋆	4056	⋆	4056
gr120	7366.40	24.00	1	3.79	1465	⋆	631	⋆	631
gr137	76258.80	35.55	2	4.79	558	⋆	2310	⋆	2310
gr202	40334.40	44.98	4	6.37	2168	⋆	3301	⋆	3301
gr229	151476.20	38.61	4	2.52	731	⋆	1544	⋆	1544
gr431	201398.00	23.27	10	⋆	2418	3.91	1949	3.91	1949
gr666	366943.00	47.05	14	3.99	1255	⋆	2794	⋆	2794
kroA150	29863.00	12.66	2	3.75	81	⋆	293	⋆	293
kroA200	32867.40	16.36	3	6.21	2675	⋆	1589	⋆	1589
kroB150	28845.80	6.56	2	4.83	874	⋆	3944	⋆	3944
kroB200	33747.40	64.91	2	6.07	183	⋆	3775	⋆	3775
lin105	15571.60	38.76	1	4.93	374	⋆	113	⋆	113
lin318	48511.20	76.43	4	8.11	2685	⋆	4034	⋆	4034
p654	39513.60	265.44	40	⋆	3011	35.81	3425	35.81	3425
pa561	3509.80	53.92	9	10.79	3474	⋆	2073	⋆	2073
pcb442	66692.80	29.62	6	⋆	3432	4.02	3054	4.02	3054
pr107	37348.40	28.25	3525	4.16	125	⋆	3820	⋆	3820
pr124	63537.40	177.35	1	5.48	2392	⋆	2772	⋆	2772
pr136	96195.60	22.86	2	5.02	1694	⋆	1037	⋆	1037
pr144	52045.40	43.92	2	21.75	154	⋆	2997	⋆	2997
pr152	64842.20	128.04	3	⋆	2193	0.32	766	0.32	766
pr226	82831.60	353.71	3	⋆	266	8.82	3459	8.82	3459
pr264	55675.20	144.29	10	⋆	2769	13.20	1713	13.20	1713
pr299	60185.80	105.50	4	⋆	106	0.18	2332	0.18	2332
pr439	138673.00	11.32	8	⋆	1183	2.13	1792	2.13	1792
rat195	2658.80	12.67	3	4.68	1713	⋆	1487	⋆	1487
rat575	8541.80	18.21	8	⋆	2819	5.71	2794	5.71	2794
rat783	11811.60	1.60	14	1.30	2387	⋆	3369	⋆	3369
rd400	18778.00	29.02	5	⋆	1660	4.76	1499	4.76	1499
si175	21762.20	40.25	2844	4.37	1424	⋆	3549	⋆	3549
si535	48914.00	109.87	2676	13.11	3450	⋆	2783	⋆	2783
ts225	178639.40	3.77	3	3.83	1508	⋆	2930	⋆	2930
tsp225	4441.60	40.27	3	5.11	1186	⋆	1104	⋆	1104
u159	49570.80	10.33	2	3.02	682	⋆	2524	⋆	2524
u574	47989.60	34.33	9	⋆	376	3.65	2231	3.65	2231
u724	54727.40	24.04	12	⋆	1307	9.75	1792	9.75	1792

TABLE 21. Best results for SL scenario and $\alpha = 0.6$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3513.60	8.77	3	0.82	2206	⋆	3876	⋆	3876
ali535	281629.00	68.92	13	⋆	2987	6.28	3538	6.28	3538
att532	37828.60	22.69	11	⋆	559	20.49	1825	20.49	1825
bier127	130058.20	37.98	2	4.65	2833	⋆	3099	⋆	3099
brg180	2214.00	4204.61	3286	32.43	1581	⋆	2186	⋆	2186
ch130	7209.60	6.54	1	5.01	930	⋆	2569	⋆	2569
ch150	8034.40	7.36	2	3.35	853	⋆	3786	⋆	3786
d198	15618.20	296.08	2782	7.29	2617	⋆	2783	⋆	2783
d493	44250.40	1.58	11	1.07	1796	⋆	3685	⋆	3685
d657	68588.60	63.95	11	⋆	1831	3.77	3899	3.77	3899
eil101	702.40	22.64	1	3.30	1329	⋆	1194	⋆	1194
fl417	14912.60	191.35	21	⋆	2022	32.65	1927	32.65	1927
gil262	3089.80	11.84	3	1.48	2142	⋆	3538	⋆	3538
gr120	7690.40	25.93	2	5.67	594	⋆	710	⋆	710
gr137	82113.40	33.71	2	4.09	2109	⋆	2930	⋆	2930
gr202	45912.40	35.27	4	3.12	2792	⋆	3132	⋆	3132
gr229	166602.00	32.89	4	⋆	3063	0.19	2772	0.19	2772
gr431	220850.00	19.26	10	⋆	2569	20.02	2332	20.02	2332
gr666	403739.00	40.10	13	⋆	3590	7.86	2197	7.86	2197
kroA150	31331.40	14.46	2	4.89	2702	⋆	3425	⋆	3425
kroA200	37626.20	9.26	2	⋆	1520	4.10	3831	4.10	3831
kroB150	31269.80	5.68	2	3.13	2257	⋆	1668	⋆	1668
kroB200	37723.80	54.64	3	0.92	1876	⋆	507	⋆	507
lin105	16608.00	39.52	1	8.01	101	⋆	203	⋆	203
lin318	56195.40	59.73	5	2.71	1142	⋆	3268	⋆	3268
p654	44616.80	233.81	39	⋆	2747	87.78	2490	87.78	2490
pa561	3880.40	44.85	9	4.96	2439	⋆	3797	⋆	3797
pcb442	70896.40	28.16	6	⋆	166	4.44	4045	4.44	4045
pr107	42894.00	45.76	3596	3.16	1780	⋆	777	⋆	777
pr124	68683.80	167.43	2	7.49	1028	⋆	2197	⋆	2197
pr136	102193.20	23.86	2	6.99	1471	⋆	1285	⋆	1285
pr144	60725.40	37.89	2	13.39	63	⋆	3076	⋆	3076
pr152	72642.20	117.17	3	4.42	16	⋆	721	⋆	721
pr226	101579.00	281.50	3	⋆	2796	8.71	3234	8.71	3234
pr264	59509.20	138.27	9	3.00	742	0.53	3583	0.53	3583
pr299	65943.00	94.66	4	⋆	2734	0.58	3763	0.58	3763
pr439	148275.80	12.14	9	1.12	2007	⋆	4023	⋆	4023
rat195	2904.20	9.48	3	0.99	1495	⋆	4023	⋆	4023
rat575	9301.20	14.28	8	⋆	2828	4.19	1285	4.19	1285
rat783	12629.80	0.91	13	⋆	1244	4.18	2659	4.18	2659
rd400	20670.80	23.57	5	⋆	2785	1.15	3639	1.15	3639
si175	23051.60	36.56	3062	3.56	1505	⋆	2761	⋆	2761
si535	52562.80	100.09	3250	9.90	105	⋆	3515	⋆	3515
ts225	196707.60	1.36	3	1.13	1089	⋆	4000	⋆	4000
tsp225	4838.60	35.40	3	2.08	1582	⋆	3110	⋆	3110
u159	54022.80	9.43	2	2.51	2121	⋆	1408	⋆	1408
u574	51394.80	32.22	10	⋆	2528	5.90	1408	5.90	1408
u724	59431.40	20.30	13	⋆	3336	3.86	2456	3.86	2456

TABLE 22. Best results for SL scenario and $\alpha = 0.8$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3758.80	7.84	4	0.87	3612	7.00	1566	7.00	1566
ali535	300239.00	65.61	13	⋆	3218	7.42	2794	7.42	2794
att532	40058.00	21.90	10	⋆	3492	4.35	3527	4.35	3527
bier127	142623.40	35.50	3	4.49	476	⋆	845	⋆	845
brg180	2418.00	3563.85	1905	30.11	1042	⋆	3955	⋆	3955
ch130	7806.60	5.92	1	1.80	2247	⋆	4000	⋆	4000
ch150	8208.20	11.93	2	5.05	2795	⋆	3346	⋆	3346
d198	18069.40	-	-	5.68	444	⋆	282	⋆	282
d493	47047.40	3.28	9	⋆	1657	5.19	3820	5.19	3820
d657	72682.00	61.74	11	⋆	1582	8.65	3741	8.65	3741
eil101	724.80	24.89	1	4.44	513	⋆	2310	⋆	2310
fl417	16785.40	168.49	21	⋆	3478	49.14	3527	49.14	3527
gil262	3145.00	16.92	4	8.10	955	⋆	3797	⋆	3797
gr120	8405.00	21.73	2	3.19	1864	⋆	1994	⋆	1994
gr137	88333.40	31.57	2	3.87	2070	⋆	2321	⋆	2321
gr202	47843.80	37.57	4	4.25	1020	⋆	3313	⋆	3313
gr229	172526.20	34.49	4	3.42	3563	⋆	1555	⋆	1555
gr431	233964.00	19.59	10	⋆	393	11.07	1758	11.07	1758
gr666	426340.00	38.96	13	⋆	225	15.43	3268	15.43	3268
kroA150	33179.60	14.77	2	5.61	1907	⋆	4056	⋆	4056
kroA200	39815.60	10.57	4	1.22	296	⋆	2851	⋆	2851
kroB150	34650.00	2.02	3	0.49	2115	⋆	3132	⋆	3132
kroB200	38374.40	59.00	4	6.42	3488	⋆	4011	⋆	4011
lin105	17806.20	38.91	1	5.44	1624	⋆	3673	⋆	3673
lin318	58379.00	60.95	5	2.77	1366	⋆	3820	⋆	3820
p654	47804.80	221.05	38	⋆	2515	68.94	3054	68.94	3054
pa561	4101.20	42.30	8	5.57	2479	⋆	2434	⋆	2434
pcb442	73521.60	29.52	6	⋆	2067	6.46	4056	6.46	4056
pr107	47471.00	51.49	3561	4.67	98	⋆	34	⋆	34
pr124	74591.40	154.41	1022	7.58	2460	⋆	2377	⋆	2377
pr136	108442.60	23.63	2	7.00	2278	⋆	2220	⋆	2220
pr144	70560.00	31.01	2	9.94	306	⋆	180	⋆	180
pr152	85602.60	95.81	3	⋆	2544	2.17	473	2.17	473
pr226	114311.00	248.79	3	⋆	786	1.78	2242	1.78	2242
pr264	67050.20	120.10	10	⋆	3257	8.22	3065	8.22	3065
pr299	71211.00	86.85	4	0.15	454	2.76	3515	2.76	3515
pr439	158893.00	11.77	8	⋆	1861	8.84	3730	8.84	3730
rat195	3018.60	10.99	3	1.97	1711	⋆	3380	⋆	3380
rat575	9758.60	14.58	9	⋆	238	8.56	3775	8.56	3775
rat783	13383.20	0.73	13	⋆	330	10.22	3966	10.22	3966
rd400	21557.60	25.08	5	2.36	1699	⋆	3977	⋆	3977
si175	23976.60	35.37	3465	3.83	2387	⋆	2152	⋆	2152
si535	55732.80	92.95	2924	8.06	1259	⋆	3989	⋆	3989
ts225	205304.20	3.78	3	1.12	2381	⋆	1814	⋆	1814
tsp225	5247.60	31.06	3	0.26	975	⋆	2355	⋆	2355
u159	58544.80	7.92	2	1.01	2544	⋆	1837	⋆	1837
u574	55187.80	29.34	9	⋆	2431	11.24	3403	11.24	3403
u724	62743.60	19.74	12	⋆	3116	9.75	2175	9.75	2175

TABLE 23. Best results for SQ scenario and $\alpha = 0.2$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3143.20	75.69	3	17.27	2785	⋆	1825	⋆	1825
ali535	289911.20	60.86	15	22.79	2363	⋆	2648	⋆	2648
att532	34821.40	130.16	12	14.53	3659	⋆	1532	⋆	1532
bier127	127988.00	52.87	3433	2.91	784	⋆	1769	⋆	1769
brg180	2014.00	1181.23	2766	38.23	1431	⋆	3628	⋆	3628
ch130	6982.20	59.55	1	3.92	114	⋆	3876	⋆	3876
ch150	7546.00	79.55	3280	5.25	2177	⋆	23	⋆	23
d198	15136.60	-	-	8.93	1515	⋆	3561	⋆	3561
d493	41027.60	75.15	11	14.94	2607	⋆	3966	⋆	3966
d657	62898.20	75.93	7	20.60	802	⋆	3966	⋆	3966
eil101	674.40	32.06	1	1.57	249	⋆	2276	⋆	2276
fl417	16925.20	316.54	14	37.25	2214	⋆	3651	⋆	3651
gil262	2861.00	86.68	2	9.93	529	⋆	3820	⋆	3820
gr120	8015.80	11.98	2598	4.96	446	⋆	146	⋆	146
gr137	79413.60	75.21	2	4.47	1059	⋆	2344	⋆	2344
gr202	44226.40	73.85	3	6.78	2545	⋆	1938	⋆	1938
gr229	156722.00	46.42	5	4.23	1230	⋆	3685	⋆	3685
gr431	239108.00	57.97	14	0.78	2118	⋆	2896	⋆	2896
gr666	426963.00	53.45	13	18.61	1821	⋆	1465	⋆	1465
kroA150	30998.60	75.56	2515	3.96	1967	⋆	665	⋆	665
kroA200	34689.80	85.11	3	9.08	3015	⋆	2197	⋆	2197
kroB150	29736.00	107.15	1609	5.28	49	⋆	3211	⋆	3211
kroB200	34347.00	131.19	2	4.55	457	⋆	293	⋆	293
lin105	16267.00	30.12	3533	5.97	644	⋆	237	⋆	237
lin318	53637.80	96.91	4	19.26	163	⋆	2220	⋆	2220
p654	61824.00	319.45	36	29.59	515	⋆	3854	⋆	3854
pa561	3296.80	90.14	7	40.62	3530	⋆	1825	⋆	1825
pcb442	63173.00	93.07	4	7.12	605	⋆	2693	⋆	2693
pr107	40108.80	29.76	3597	⋆	1455	0.56	4034	0.56	4034
pr124	65516.60	108.60	1	6.63	1945	⋆	676	⋆	676
pr136	103656.20	103.92	3432	6.13	529	⋆	2411	⋆	2411
pr144	82880.20	92.29	1	23.10	2233	⋆	3617	⋆	3617
pr152	75053.40	192.70	2	11.71	510	⋆	1048	⋆	1048
pr226	91213.00	291.42	2889	25.21	2966	⋆	2017	⋆	2017
pr264	54190.00	309.98	8	15.09	319	⋆	823	⋆	823
pr299	61484.60	159.89	5	7.61	2020	⋆	1983	⋆	1983
pr439	147258.40	99.84	8	30.32	631	⋆	2930	⋆	2930
rat195	2709.00	43.48	3	11.89	2413	⋆	3515	⋆	3515
rat575	9050.20	42.73	7	19.97	1466	⋆	935	⋆	935
rat783	12423.80	70.46	10	29.88	1595	⋆	2062	⋆	2062
rd400	19631.20	77.81	5	29.22	3581	⋆	992	⋆	992
si175	22339.40	25.41	3553	3.82	799	⋆	1397	⋆	1397
si535	50628.40	63.38	1061	17.63	2286	⋆	2287	⋆	2287
ts225	173576.20	99.80	2	18.27	3194	⋆	2839	⋆	2839
tsp225	4826.00	95.35	3	16.33	632	⋆	3741	⋆	3741
u159	50997.40	94.93	2	2.75	936	⋆	1758	⋆	1758
u574	48388.60	114.65	7	32.42	550	⋆	2479	⋆	2479
u724	55834.20	102.60	9	25.26	2685	⋆	4000	⋆	4000

TABLE 24. Best results for SQ scenario and $\alpha = 0.4$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3512.00	60.83	3	6.96	1344	⋆	2637	⋆	2637
ali535	327228.20	45.03	16	7.37	871	⋆	4056	⋆	4056
att532	40069.60	103.11	11	4.00	1057	⋆	1577	⋆	1577
bier127	138626.80	29.78	3601	0.65	1996	⋆	1544	⋆	1544
brg180	2216.00	602.80	3330	28.79	2392	⋆	597	⋆	597
ch130	7490.60	53.60	1	2.06	1596	⋆	766	⋆	766
ch150	7951.60	73.35	3504	1.08	2543	⋆	1735	⋆	1735
d198	16180.80	199.74	3464	7.09	245	⋆	2569	⋆	2569
d493	46096.40	58.68	10	6.02	809	⋆	1882	⋆	1882
d657	68857.80	62.94	7	10.06	3709	⋆	3121	⋆	3121
eil101	701.40	31.19	1	1.91	754	⋆	496	⋆	496
fl417	19709.60	264.57	15	13.35	3579	⋆	3177	⋆	3177
gil262	2922.60	87.11	3	12.10	2252	⋆	3842	⋆	3842
gr120	8494.80	4.80	2684	2.38	501	⋆	2907	⋆	2907
gr137	83729.60	70.96	2	4.05	1651	⋆	1431	⋆	1431
gr202	46247.20	68.54	1946	2.63	2555	⋆	1949	⋆	1949
gr229	161743.60	46.07	4	2.43	734	⋆	3414	⋆	3414
gr431	234096.80	64.23	14	2.23	2993	⋆	3065	⋆	3065
gr666	466335.20	43.15	13	5.12	1032	⋆	2614	⋆	2614
kroA150	32055.00	77.01	3079	2.48	1906	⋆	800	⋆	800
kroA200	36549.20	80.51	2	6.18	2119	⋆	3820	⋆	3820
kroB150	32209.60	94.15	3552	0.93	1433	⋆	3054	⋆	3054
kroB200	36219.40	124.28	2	6.61	2175	⋆	992	⋆	992
lin105	17504.00	25.38	3567	5.43	514	⋆	23	⋆	23
lin318	58112.00	85.15	3	16.41	873	⋆	924	⋆	924
p654	69385.40	278.53	36	18.36	1280	⋆	3054	⋆	3054
pa561	3532.60	80.10	7	33.89	1173	⋆	3504	⋆	3504
pcb442	70651.80	75.51	4	2.78	371	⋆	1206	⋆	1206
pr107	44820.20	19.04	3464	0.50	684	⋆	597	⋆	597
pr124	73344.60	76.54	701	3.81	822	⋆	68	⋆	68
pr136	111919.80	33.16	3542	0.76	453	⋆	2287	⋆	2287
pr144	90482.60	82.31	1	19.30	1901	⋆	45	⋆	45
pr152	83091.60	173.24	2	12.12	1017	⋆	3527	⋆	3527
pr226	98141.60	291.11	2	25.41	2238	⋆	2197	⋆	2197
pr264	59156.20	278.23	3289	8.19	2303	⋆	2839	⋆	2839
pr299	66417.20	144.46	5	4.04	3015	⋆	3392	⋆	3392
pr439	170010.80	76.51	7	14.47	571	⋆	3369	⋆	3369
rat195	2891.80	37.84	1432	8.54	1118	⋆	3335	⋆	3335
rat575	9336.00	40.81	7	18.82	2430	⋆	2558	⋆	2558
rat783	13107.60	63.64	10	21.68	2937	⋆	3459	⋆	3459
rd400	20713.40	71.35	4	23.56	1933	⋆	3515	⋆	3515
si175	22813.20	16.33	3561	4.33	2048	⋆	1239	⋆	1239
si535	51727.60	60.55	1288	16.14	2294	⋆	1476	⋆	1476
ts225	182243.80	94.35	2	16.84	1817	⋆	1566	⋆	1566
tsp225	5071.60	85.30	3229	11.49	3116	⋆	1386	⋆	1386
u159	54056.00	89.03	2	1.93	1551	⋆	1758	⋆	1758
u574	54956.00	91.28	8	16.43	1080	⋆	1217	⋆	1217
u724	59486.40	92.64	9	22.61	1797	⋆	1656	⋆	1656

TABLE 25. Best results for SL scenario and $\alpha = 0.6$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3706.40	55.80	3	3.76	2912	⋆	3076	⋆	3076
ali535	354255.00	36.63	16	⋆	3255	3.39	1825	3.39	1825
att532	42349.00	94.87	11	⋆	3408	1.57	2186	1.57	2186
bier127	141942.00	38.74	3588	⋆	2067	0.60	2366	0.60	2366
brg180	2292.00	889.44	2896	26.88	1519	⋆	2118	⋆	2118
ch130	7709.20	53.98	1	1.34	1294	⋆	124	⋆	124
ch150	8048.80	78.79	2942	5.18	1031	⋆	1487	⋆	1487
d198	18005.60	175.48	2666	0.79	2125	⋆	1048	⋆	1048
d493	47109.00	58.04	10	4.72	3605	⋆	1893	⋆	1893
d657	76151.20	49.27	8	3.33	2519	⋆	2445	⋆	2445
eil101	721.20	9.04	2158	3.05	410	⋆	4011	⋆	4011
fl417	18235.20	301.59	15	41.39	385	⋆	1476	⋆	1476
gil262	3057.60	83.05	2	6.99	1708	⋆	2400	⋆	2400
gr120	8650.20	8.32	3053	1.16	33	⋆	79	⋆	79
gr137	88321.80	65.90	1039	2.83	1532	⋆	496	⋆	496
gr202	48008.20	65.22	3	⋆	3526	0.53	2546	0.53	2546
gr229	172805.00	40.65	4	⋆	1831	0.18	4011	0.18	4011
gr431	242846.00	61.12	14	⋆	1366	2.37	1961	2.37	1961
gr666	508873.80	33.58	13	4.15	540	⋆	3944	⋆	3944
kroA150	34372.80	64.92	3588	0.88	1813	⋆	1780	⋆	1780
kroA200	38106.40	77.76	2	4.08	2252	⋆	541	⋆	541
kroB150	33240.00	84.16	3097	3.33	2754	⋆	2242	⋆	2242
kroB200	37749.80	120.07	3	6.45	1967	⋆	2254	⋆	2254
lin105	18641.00	11.06	3270	3.98	1223	⋆	1307	⋆	1307
lin318	60626.00	81.02	3	11.44	3470	⋆	4056	⋆	4056
p654	80777.80	229.25	34	9.77	3576	⋆	3177	⋆	3177
pa561	3731.00	73.07	7	29.73	2004	⋆	3268	⋆	3268
pcb442	69350.40	81.73	4	5.04	438	⋆	1769	⋆	1769
pr107	49372.40	16.07	3415	1.17	322	⋆	2152	⋆	2152
pr124	79055.80	84.85	1	0.61	2256	⋆	2839	⋆	2839
pr136	117773.40	49.69	3520	0.43	941	⋆	2715	⋆	2715
pr144	94012.00	81.45	2	16.01	672	⋆	1848	⋆	1848
pr152	92238.20	154.11	2	8.13	1067	⋆	1983	⋆	1983
pr226	116068.80	238.78	2	9.64	2861	⋆	3606	⋆	3606
pr264	69912.00	229.64	8	4.77	3302	⋆	3042	⋆	3042
pr299	69348.00	137.58	5	1.82	278	⋆	2986	⋆	2986
pr439	156643.60	95.55	8	22.08	1915	⋆	3752	⋆	3752
rat195	2927.80	39.72	3	6.19	1663	⋆	3437	⋆	3437
rat575	9930.20	34.76	8	12.18	223	⋆	3087	⋆	3087
rat783	14615.40	48.62	10	13.57	1888	⋆	4056	⋆	4056
rd400	21520.60	67.59	4	21.89	1304	⋆	1566	⋆	1566
si175	23374.20	13.37	3553	3.30	590	⋆	349	⋆	349
si535	53815.00	54.58	3205	11.26	286	⋆	1983	⋆	1983
ts225	192156.00	88.17	2	11.13	3384	⋆	146	⋆	146
tsp225	5228.20	87.62	3	10.92	2353	⋆	2952	⋆	2952
u159	56757.20	84.91	2	2.06	2548	⋆	766	⋆	766
u574	56628.80	88.12	7	15.29	668	⋆	4045	⋆	4045
u724	64214.60	80.62	9	16.79	1782	⋆	2242	⋆	2242

TABLE 26. Best results for SL scenario and $\alpha = 0.8$.

Instance	Best	BRKGA		M-VND		M-CNS		LS	
		%	t	%	t	%	t	%	t
a280	3713.40	58.91	3	8.00	822	⋆	4000	⋆	4000
ali535	350398.00	40.74	15	⋆	2389	3.72	2434	3.72	2434
att532	44071.00	90.42	12	⋆	1218	1.27	1600	1.27	1600
bier127	146626.40	16.35	3405	0.05	2474	⋆	3324	⋆	3324
brg180	2334.00	867.10	2303	28.96	1934	⋆	1611	⋆	1611
ch130	7974.00	48.75	820	2.15	1223	⋆	3482	⋆	3482
ch150	8322.60	91.07	626	5.31	2534	⋆	192	⋆	192
d198	18431.20	-	-	5.74	550	⋆	1758	⋆	1758
d493	47699.20	58.90	10	4.78	1171	⋆	3989	⋆	3989
d657	76983.00	49.73	7	4.71	3064	⋆	1679	⋆	1679
eil101	750.80	3.33	980	1.47	295	⋆	789	⋆	789
fl417	26095.00	185.72	14	⋆	2644	2.43	3425	2.43	3425
gil262	3157.60	81.38	2	8.54	2669	⋆	2107	⋆	2107
gr120	8941.80	9.29	2530	3.71	16	⋆	79	⋆	79
gr137	93473.20	61.71	2	0.60	1014	⋆	687	⋆	687
gr202	49358.40	61.36	1397	1.26	1607	⋆	2975	⋆	2975
gr229	175181.20	42.62	5	1.69	1079	⋆	3718	⋆	3718
gr431	228030.20	74.81	14	7.86	3493	⋆	3696	⋆	3696
gr666	484812.00	42.87	13	7.33	908	⋆	3775	⋆	3775
kroA150	34932.80	58.33	3509	5.03	2034	⋆	2276	⋆	2276
kroA200	39973.60	73.86	3	2.42	208	⋆	1679	⋆	1679
kroB150	34017.40	86.29	2906	5.57	569	⋆	3538	⋆	3538
kroB200	40721.40	108.54	2	⋆	3231	2.62	2625	2.62	2625
lin105	19619.00	23.71	2722	0.89	1887	⋆	3346	⋆	3346
lin318	65772.40	69.92	3	7.13	2665	⋆	3031	⋆	3031
p654	90123.20	198.36	34	⋆	3802	3.13	3020	3.13	3020
pa561	4090.20	60.42	7	19.46	2352	⋆	3899	⋆	3899
pcb442	75617.40	69.22	4	2.61	1118	⋆	1972	⋆	1972
pr107	54121.40	14.35	3233	1.41	847	⋆	3189	⋆	3189
pr124	82108.60	88.34	1	5.41	2962	⋆	777	⋆	777
pr136	121172.00	58.47	3553	⋆	2982	0.90	79	0.90	79
pr144	98767.20	78.44	2	15.00	3577	⋆	3662	⋆	3662
pr152	96253.00	151.15	2	15.51	411	⋆	3414	⋆	3414
pr226	132307.40	204.31	2	6.59	2304	⋆	3786	⋆	3786
pr264	70925.40	230.75	8	2.95	421	⋆	4056	⋆	4056
pr299	66825.20	150.49	5	10.15	479	⋆	777	⋆	777
pr439	181194.00	72.13	7	9.20	1209	⋆	2355	⋆	2355
rat195	3083.40	35.96	3	6.31	899	⋆	3955	⋆	3955
rat575	10899.20	24.82	7	⋆	862	0.42	1352	0.42	1352
rat783	15081.20	46.01	10	6.33	398	⋆	4056	⋆	4056
rd400	21844.00	67.79	4	22.16	449	⋆	3166	⋆	3166
si175	23804.00	24.54	3506	3.86	72	⋆	1431	⋆	1431
si535	54353.20	53.54	2618	12.73	102	⋆	3538	⋆	3538
ts225	191523.00	92.72	2	13.31	276	⋆	417	⋆	417
tsp225	5606.40	78.44	2	5.49	3001	⋆	3831	⋆	3831
u159	58916.60	82.84	2	⋆	1678	0.03	2592	0.03	2592
u574	62333.20	73.29	8	5.51	1629	⋆	1927	⋆	1927
u724	69170.80	70.24	9	8.81	158	⋆	2794	⋆	2794