Computers and Operations Research 123 (2020) 104986

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/caor

A branch-cut-and-price algorithm for the traveling salesperson problem with hotel selection

Luiz Henrique Barbosa, Eduardo Uchoa *

Universidade Federal Fluminense, Departamento de Engenharia de Produção Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil

ARTICLE INFO

Article history:
Received 28 June 2019
Revised 29 April 2020
Accepted 29 April 2020
Available online 11 May 2020

Keywords:
Column generation
Cut separation
Routing

ABSTRACT

The Traveling Salesperson Problem with Hotel Selection (TSPHS) is a realistic extension of the classic Traveling Salesperson Problem recently introduced to the literature. In the TSPHS, there is a time limit that restricts the visits that can be performed in a single day. Therefore, several days may be necessary to visit all clients. The salesperson has to spend the night in one of the available hotels. Previous works focus mainly on metaheuristics and on MIP formulations. This work presents a sophisticated exact algorithm for the TSPHS, a Branch-Cut-and-Price (BCP) algorithm that includes and adapts several features found in state-of-the-art algorithms for vehicle routing. In that algorithm, columns correspond to possible salesperson day trips; subtour elimination cuts, 2-path cuts, and limited-memory subset row cuts are separated. Computational results show that many medium-sized instances, having up to 75 clients and 20 hotels, can be solved to optimality, as well as some larger instances from the literature, with up to 225 clients.

1. Introduction

The Traveling Salesperson Problem with Hotel Selection (TSPHS), introduced by Vansteenwegen et al. (2012), is a generalization of the classic TSP where, as expected in many actual situations, the salesperson has a limited time in each day for visiting the clients and must stay at a hotel overnight. A TSPHS solution, a tour, is composed of a connected sequence of trips, each one corresponding to a salesperson's workday. Each trip should start at a hotel, then visit a subset of the clients and end at a hotel. The maximum duration of a trip, including traveling time and client service time, should respect a given limit. The tour must visit all clients exactly once, but hotels can be visited any number of times. Finally, the sequence of trips must start and finish at the origin hotel, which may represent the salesperson's residence.

The TSPHS can be directly applied to solve real-world problems other than the application suggested by its name. An example of such application is planning long routes for truck drivers. Many countries have strict regulations on how many hours per day a truck driver can work. Therefore, when planning these routes, one must take those limits into account. Another natural application could be to plan routes for electric vehicles that have limited autonomy in areas where few charging stations are available. When the problem was initially introduced in Vansteenwegen et al. (2012), one of the main motivations was to address a problem faced by a geographical mapping company that needed to plan mapping tasks that could span multiple days.

More formally, the TSPHS is defined as follows. Let $G = (V, A)$ be a complete directed graph with vertex-set $V = H \cup C$, where $H = \{0, \ldots, m-1\}$ represents a set of m *hotels* and $C = \{m, \ldots, m+n-1\}$ represents a set of n *clients*. Vertex 0 represents the *origin hotel*. Each arc $a \in A$ has a *travel time* c_a. Each client $i \in C$ has a *service time* s_i. A *trip* is an elementary path in G in which the first and last nodes are hotels, and all other nodes in between (if any) are clients. Given a trip t, let $V(t)$ and $A(t)$ be the sets of vertices and arcs in t, respectively. The duration of a trip t, given by $\sum_{a \in A(t)} c_a + \sum_{i \in (V(t) \cap C)} s_i$, should not exceed a given *time limit* L. The *cost* of a trip t is given by $c(t) = \sum_{a \in A(t)} c_a$, just the sum of the traveling times. A *tour* $S = (t_1, \ldots, t_K)$ is a sequence of $K \geqslant 1$ trips such that: (i) t_1 starts at 0; (ii) t_j ends at the same hotel where t_{j+1} starts, for $j = 1, \ldots, K-1$; (iii) t_K ends at 0; (iv) each client $i \in C$ belongs to exactly one of the trips in S. The origin hotel is the only hotel that must be visited, and it is valid to visit hotels consecutively in a tour, in cases where there are trips that do not visit any clients. The TSPHS consists of finding a tour S with the lowest possible number of trips that minimizes the total trip cost. In other words, it first minimizes the number of trips, and secondly, their cost. Note that a TSPHS instance can be infeasible if L is too small for its traveling and service times.

* Corresponding author.
E-mail address: uchoa@producao.uff.br (E. Uchoa).

https://doi.org/10.1016/j.cor.2020.104986
0305-0548/

The classic TSP can be modeled as a TSPHS with a single hotel and $L = \infty$, therefore the TSPHS is also an NP-hard problem. Actually, TSPHS tours can be much more topolologically complex than TSP tours. This happens because some hotels may be visited more than once, creating subtours. Fig. 1 shows an example of a TSPHS tour, for an instance with $m = 8$ and $n = 10$. Hotels are represented in squares; clients are represented as circled nodes. The tour is $S = (\mathbf{0}, 8, 9, \mathbf{4}, \mathbf{5}, 11, \mathbf{5}, 10, \mathbf{6}, 12, 13, \mathbf{6}, 14, 15, \mathbf{3}, 16, \mathbf{2}, 17, \mathbf{0})$.

1.1. Literature review and related problems

Vansteenwegen et al. (2012) presented TSPHS for the first time, proposed an Iterated Local Search (ILS) algorithm for its solution and a MIP formulation for the problem. In that MIP, the number of trips is fixed at q and iteratively increased until it finds a feasible solution. Miller-Tucker-Zemlin-like constraints (Miller et al., 1960) are used to prevent subtours within a trip. Furthermore, they also proposed four sets of benchmark instances, based on existing Vehicle Routing Problem (VRP) and TSP instances, and compared results from the ILS to those of the MIP formulation solved by CPLEX. Only instances with up to four trips, two hotels, and 40 clients could be solved to optimality.

Castro et al. (2013) presented a memetic algorithm with an embedded tabu search, adapting well-known neighborhood operators for the TSP and the VRP, and a different MIP formulation. The memetic algorithm consistently obtained optimal solutions for the instances where the optimal was known and improved the best-known solutions for several other instances. The MIP formulation used a Big-M to prioritize minimizing the total number of trips over the total duration of the tour. The Miller-Tucker-Zemlin subtour elimination constraints were also replaced by the much more efficient Dantzig-Fulkerson-Johnson constraints. Although the focus of that work was on the memetic algorithm, a cutting-plane algorithm over the new formulation was able to compute some additional optimal solutions for instances with up to 40 clients.

Castro et al. (2015) obtained better results for the benchmark instances with a heuristic algorithm based on Iterated Local Search and Variable Neighborhood Descent meta-heuristics. The initial solution is constructed based on an order-first split-second strategy and is then improved upon by using several different neighborhoods, some of which had been used in previous papers. This new algorithm was also significantly faster than the previous ones. Moreover, a new mathematical formulation based on set partitioning was introduced, well suited for a column generation algorithm. However, they did not present an exact algorithm based on that formulation.

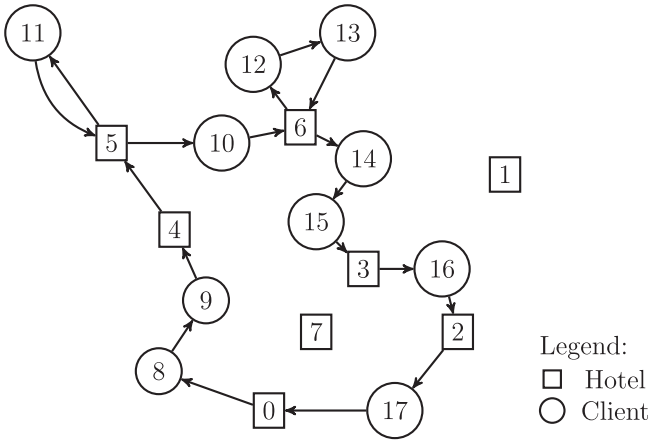

Fig. 1. Example of a TSPHS tour.

Finally, a Variable Neighborhood Search algorithm was presented in Sousa et al. (2015). The VNS algorithm used the same order-first, split-second strategy to generate initial solutions and well-known neighborhoods to refine it, as well as a perturbation operator on the clients. This approach was successful in obtaining improvements over the results in Castro et al. (2013) and Castro et al. (2015).

Several problems are closely related to the TSPHS. The Black and White Traveling Salesman Problem (Ghiani et al., 2006) is a variation of the TSP where the vertex set is partitioned into black and white vertices. There is a constraint on the number of consecutive white vertices that can be visited, as well as the total length traversed between two black vertices. Aside from the constraint on the number of consecutive white vertices visited, the main difference from the TSPHS is that in the BWTSP, every vertex must be visited exactly once, including the black vertices, whereas in the TSPHS hotels can be visited any number of times. It has applications in airline scheduling and telecommunications.

The Orienteering Problem with Hotel Selection (OPHS) introduced by Divsalar et al. (2013) is also similar to the TSPHS. The vertex set consists of clients and hotels, and each client can be visited at most once. The goal is to visit the set of clients that maximizes the total collected score while respecting a daily budget constraint. Similarly to the TSPHS, hotels must be selected to stay in between trips. The OPHS differs from the TSPHS as it must choose which clients to visit given a fixed number of days, where the goal of the TSPHS is to visit all clients, minimizing the number of days and the tour length.

Vansteenwegen et al. (2012) discusses the similarities between the TSPHS and other routing problems. For a thorough review of routing problems with intermediate stops, we refer to Schiffer et al. (2019).

2. The proposed algorithm

The proposed approach uses a formulation based on set partitioning to solve the TSPHS. Although Castro et al. (2015) provides a set partitioning formulation, it is different from the one used in this work. The formulation is then solved using a branch-cut-and-price algorithm that generates both columns and cuts.

2.1. Mathematical formulation

Let T be the set of all valid trips. For each $t \in T$, define a non-negative variable λ_t and coefficients a_{ij}^t indicating how many times arc (i,j) appears in trip t. For $S \subset V, \delta^-(S)$ and $\delta^+(S)$ are the subsets of the arcs in $G = (V,A)$ that enter and leave S, respectively. The formulation for the TSPHS given below has an exponentially large number of variables and constraints and leads to a branch-cut-and-price algorithm.

$$\text{minimize} \sum_{t \in T} \left(\sum_{a \in A} c_a a_a^t \right) \lambda_t \tag{1}$$

subject to

$$\sum_{t \in T} \left(\sum_{a \in \delta^-(\{j\})} a_a^t \right) \lambda_t = 1 \quad \forall j \in C \tag{2}$$

$$\sum_{t \in T} \left(\sum_{a \in \delta^+(\{0\})} a_a^t \right) \lambda_t \geqslant 1 \tag{3}$$

$$\sum_{t \in T} \left(\sum_{a \in \delta^-(\{h\})} a_a^t - \sum_{a \in \delta^+(\{h\})} a_a^t \right) \lambda_t = 0 \quad \forall h \in H \tag{4}$$

$$\sum_{t\in T}\left(\sum_{a\in\delta^-(S)} a_a^t\right)\lambda_t \geqslant 1 \quad \forall S\subset V: 0\notin S, S\cap C\neq\varnothing \tag{5}$$

$$\sum_{t\in T}\left(\sum_{a\in\delta^+(H)} a_a^t\right)\lambda_t = q \tag{6}$$

$$\lambda_t\in\{0,1\} \quad \forall t\in T \tag{7}$$

In this formulation, the number of trips q is fixed and obtained by a procedure described in Section 2.2. The objective function then aims to minimize the total tour duration. Constraints (2) define that every client must be visited exactly once by determining that they have one incoming arc. Constraint (3) requires that hotel 0 be visited at least once. Consider the constraint (4) corresponding to a hotel h; trips that end at a hotel h have coefficient 1, trips that start at h have coefficient -1 and trips that both start and end at h have coefficient 0. Therefore, constraints (4) ensure that the same number of trips start and end at each hotel. Constraints (5) ensure overall trip connectivity by enforcing the existence of a path from hotel 0 to every client. Finally, constraint (6) fix the total number of trips and constraints (7) define decision variables λ_t as binary. Note that isolated cycles formed by arcs joining only hotels are not forbidden, but those cycles would not appear in any optimal solution.

The formulation (1)–(7) presented above is for the generic case where costs between nodes can be asymmetric. For the symmetric case, i.e., where $c_{ij}=c_{ji}, \forall (i,j)\in A$, we define a different formulation that allows us to increase the efficiency of the algorithm used for pricing trips. In this formulation, the problem is defined over a non-directed complete graph $G=(V,E)$ and decision variables $b_h : h\in H$ are introduced to assist in ensuring that every hotel has an even number of incident edges. Coefficients $a_{ij}^t : (i,j)\in E$ must also be redefined so that they indicate how many times an edge (i,j) is navigated in trip t. Finally, we also define $\delta(S) : S\subset V$ as the subsets of edges in $G=(V,E)$ with one endpoint in S and $E(S)$ as the subset of edges with both endpoints in S.

$$\text{minimize}\sum_{t\in T}\left(\sum_{e\in E} c_e a_e^t\right)\lambda_t \tag{8}$$

subject to

$$\sum_{t\in T}\left(\sum_{e\in\delta(\{j\})} a_e^t\right)\lambda_t = 2 \quad \forall j\in C \tag{9}$$

$$\sum_{t\in T}\left(\sum_{e\in\delta(\{0\})} a_e^t\right)\lambda_t \geqslant 2 \tag{10}$$

$$\sum_{t\in T}\left(\sum_{e\in\delta(\{h\})} a_e^t\right)\lambda_t = 2b_h \quad \forall h\in H \tag{11}$$

$$\sum_{t\in T}\left(\sum_{e\in\delta(S)} a_e^t\right)\lambda_t \geqslant 2 \quad \forall S\subset V: 0\notin S, S\cap C\neq\varnothing \tag{12}$$

$$\sum_{t\in T}\left(\sum_{e\in\delta(H)\cup E(H)} a_e^t\right)\lambda_t = 2q \tag{13}$$

$$\lambda_t\in\{0,1\} \quad \forall t\in T \tag{14}$$

$$b_h\in\mathbb{Z}^+ \quad \forall h\in H \tag{15}$$

Constraints (9) define that every client should be visited exactly once by determining that they have two adjacent edges. Constraint (10) requires that hotel 0 be visited at least once. Constraints (11) guarantee that each hotel has an even number of adjacent edges. Constraints (12) are similar to constraints (5) and enforce overall trip connectivity by requiring the existence of two paths from hotel 0 to every client. Finally, constraint (13) determines the total number of trips and constraints (14) and (15) define decision variables λ_t as binary and b_h as non-negative integers, respectively.

A solution to the formulation (8)–(15) can also be expressed in terms of edge-based decision variables $x_e : e\in E$ by applying the following expression:

$$x_e = \sum_{t\in T} a_e^t\lambda_t \quad \forall e\in E \tag{16}$$

In fact, the integrality constraint (14) expressed over λ variables can be replaced by constraints (17) and (18).

$$x_e\in\{0,1,2\} \quad \forall e\in\delta(H) \tag{17}$$

$$x_e\in\{0,1\} \quad \forall e\in E\setminus\delta(H) \tag{18}$$

Given that it is possible for a trip to begin and end at the same hotel, while visiting only a single client, constraint (17) allows edges adjacent to hotels to be traversed twice. Any constraint expressed in terms of x variables can be transformed to the λ variables using (16). It can be observed that all the constraints in the set partitioning formulation can be seen as arising from such a transformation. For example, constraints (9) are equivalent to $\sum_{e\in\delta(\{j\})} x_e = 2, \forall j\in C$.

2.1.1. The pricing subproblem

The pricing subproblem is responsible for generating valid trips to be added to the master problem defined by the linear relaxation of the formulations in Section 2.1. Every trip must be a valid path that starts and ends at one of the available hotels and possibly visit one or more clients in between, respecting the constraint that prevents the trip length from exceeding the daily time limit L.

This problem can be modeled as a Shortest Path Problem with Resource Constraints (SPPRC), where the constraint is set on the total duration of the trip, also an NP-hard problem in itself (see Desaulniers et al., 2005). Since the trip can start and end at any available hotel, the pricing subproblem uses an auxiliary graph derived from the original problem graph. Given a graph $G'=(N'',A'')$, the pricing consists in finding the shortest path between nodes s and t, placeholders for the source and sink nodes of the shortest path problem. Sets $H''=\{0'',\ldots,(m-1)''\}$ and $H''=\{0'',\ldots,(m-1)''\}$ will be copies of the original set of hotels H and represent the set of available hotels for the salesman to start and end a trip, respectively. Consequently, set N'' will be defined as $\{s,t\}\cup H''\cup H''\cup C$, and the set of arcs $A''=\bigcup_{k=1}^{6}A_k''$, where $A_k'', k=1,\ldots,6$ is defined below.

$$\begin{aligned}
A_1'' &= \{(i,j): i=s, j\in H''\}\\
A_2'' &= \{(i,j): i\in H'', j\in H''\}\setminus\{(i',j''): i'\in H', j''\in H'', i=j\}\\
A_3' &= \{(i,j): i\in H', j\in C\}\\
A_4' &= \{(i,j): \{i,j\}\subset C\}\\
A_5' &= \{(i,j): i\in C, j\in H''\}\\
A_6' &= \{(i,j): i=H'', j=t\}
\end{aligned}$$

Fig. 2 shows an example of an auxiliary graph for the pricing subproblem for an instance with two hotels and three clients. Note that the definition of A' excludes arcs where both endpoints are at the same node, preventing trips that begin and end at the same hotel without visiting any clients.

The reduced cost $\bar{c}_{ij}$ of an arc variable $a=(i,j)\in A'$ for the symmetric case, based on formulation (8)–(15), are calculated as follows. Let $\pi_i, i\in C, \alpha, \beta_h, h\in H$ and $\omega_S, S\subset V: 0\notin S, S\cap C\neq\varnothing$ denote the duals associated with constraints ()()()(9)–(11) and (12) and v be the dual value associated with constraint (13). Define also $f(i,j)$ as a function that returns 1 if $(i,j)\in\delta(0)$ or 0 otherwise.

$$\bar{c}_{ij} = 0 \forall (i,j) \in A'_1 \cup A'_6$$

$$\bar{c}_{ij} = c_{ij} - \beta_i - \beta_j - \sum_{\substack{S \subset V: 0 \notin S, S \cap C \neq \emptyset \\ (i,j) \in \delta(S)}} \omega_S - f(i,j)\alpha - 2\nu \forall (i,j) \in A'_2$$

$$\bar{c}_{ij} = c_{ij} - \beta_i - \pi_j - \sum_{\substack{S \subset V: 0 \notin S, S \cap C \neq \emptyset \\ (i,j) \in \delta(S)}} \omega_S - f(i,j)\alpha - \nu \forall (i,j) \in A'_3$$

$$\bar{c}_{ij} = c_{ij} - \pi_i - \pi_j - \sum_{\substack{S \subset V: 0 \notin S, S \cap C \neq \emptyset \\ (i,j) \in \delta(S)}} \omega_S \forall (i,j) \in A'_4$$

$$\bar{c}_{ij} = c_{ij} - \pi_i - \beta_j - \sum_{\substack{S \subset V: 0 \notin S, S \cap C \neq \emptyset \\ (i,j) \in \delta(S)}} \omega_S - f(i,j)\alpha - \nu \forall (i,j) \in A'_5$$

Let $c(P)$ be the total cost of path P, given by the sum of the arcs traversed in such path and $s(P)$ the sum of the service times of the clients visited in P. The pricing subproblem now consists of finding an s-t path over G' that minimizes $\bar{c}(P)$, the total reduced cost of path P respecting the time limit L, and is described by (19).

$$\text{minimize}\{\bar{c}(P) : P \in \mathcal{P}_{s-t}(G'), c(P) + s(P) \leqslant L\} \tag{19}$$

As described in detail in Section 2.4, the pricing subproblem is solved through a labeling algorithm.

2.2. The branch-cut-and-price algorithm for the TSPHS

The algorithm optionally takes as input upper bound parameters q_{UB} and l_{UB}, respectively, the total number of trips and the total cost of a known solution, possibly obtained through a heuristic algorithm. It starts by obtaining a lower bound on the TSPHS tour length, l_{LB}. That is accomplished by first solving a classic TSP over the set of points defined by the clients and the origin hotel, using travel times as distances. Note that if travel times do not satisfy the triangular inequality, the distances should be defined as the value of the shortest travel time between two points, possibly passing by hotels in $H \setminus \{0\}$. If $l_{LB} + \sum_{i \in V} s_i$ is less than the daily time limit L, then the TSPHS can be completed in one day, and the TSP solution is valid and optimal for the TSPHS. Otherwise, we calculate a lower bound q_{LB} on the number of trips by dividing $l_{LB} + \sum_{i \in V} s_i$ by L and rounding up to the next integer.

The algorithm initializes $q = q_{LB}$, and runs the Branch-Cut-and-Price (BCP) algorithm iteratively, incrementing q until it finds a solution. The total cost of the heuristic solution, l_{UB}, can only be used by the branch-and-bound to prune the branching tree when $q = q_{UB}$, as it is possible to have a solution with fewer trips, but greater length. If $q \neq q_{UB}$, a loose upper bound is taken by multiplying q by L.

This iterative strategy proved to be more efficient than using a Big-M approach, as it becomes easier to use the upper bound available from a known solution and makes the branch-and-bound tree more balanced. This way, the BCP algorithm iteratively increases q to search solutions with that number of trips until it finds one. The pseudo-code in Algorithm 1 describes the procedure.

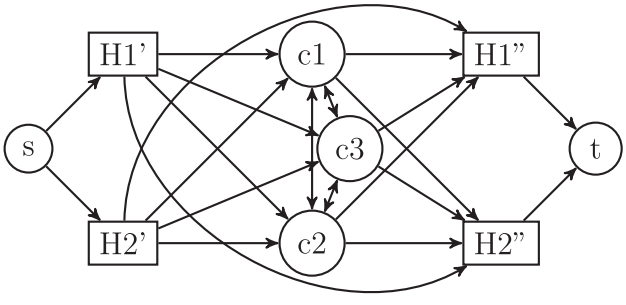

Fig. 2. Auxiliary graph for the pricing subproblem.

Algorithm 1 Solve-TSPHS(q_{UB}, l_{UB})

$S_{TSP} = \text{Solve-TSP}(C \cup \{0\})$
$l_{LB} = c(S_{TSP})$
if $l_{LB} + \sum_{i \in V} s_i \leq L$ **then return** S_{TSP}
$S \leftarrow \emptyset$
$q = q_{LB} = \lceil (l_{LB} + \sum_{i \in V} s_i)/L \rceil$
while $S = \emptyset$ **do**
  $ub \leftarrow q \times L$
  if $q = q_{UB}$ **then** $ub \leftarrow l_{UB}$
  $S \leftarrow \text{BranchCutAndPrice}(q, ub)$
  $q \leftarrow q + 1$
return S

2.3. Cuts

Our algorithm uses three families of cuts. Following the classification provided in Fukasawa et al. (2006), two of these cuts are considered robust, i.e., they do not change the pricing subproblem. Those robust cuts can be defined over the x variables. They are translated to λ variables using (16). The third family of cuts, lm-SRCs, can only be defined directly over the λ variables. It is non-robust, demanding more careful management.

The cuts are added in the following order:

1. SECs
2. 2-Path Cuts
3. lm-SRCs

Let $\mathcal{F}$ denote the set of families of cuts described above. Algorithm 2 describes the procedure in which cuts are separated and added to the problem. After obtaining a solution from the CG procedure, the algorithm attempts to separate cuts for each of the families. A family of cuts is only processed when no cuts of previous families can be separated. When cuts are successfully separated, the algorithm executes the CG procedure again.

Note that adding a cut from a particular family can result in a solution that violates constraints from families that have already been processed. Therefore, after the CG procedure finishes, at every iteration, the algorithm attempts to separate cuts starting back from the first family of cuts. When no more cuts from any of the families can be separated, the algorithm then returns the solution S.

Algorithm 2 ColumnAndCutGeneration(RMP)

```
S ← ∅
repeat
    addedCuts ← false
    S ← StandardColumnGeneration(RMP)
    foreach f ∈ F (in order) do
        C ← SeparateCuts(f, S)
        if C ≠ ∅ then
            Add cuts C to RMP
            addedCuts ← true
            break
until ¬addedCuts
return S
```

This approach allows us to prioritize families of cuts for which the separation procedure is faster and that have less impact on the pricing subproblem, as will be described later. While the SECs are essential for the problem formulation, this algorithm uses the other families of cuts with the objective of strengthening the formulation, contributing to raising the lower bounds obtained and thus reducing the size of the branch-and-bound tree. In that context, such an iterative approach also enables us to more easily assess the impact that each family of cuts has in raising the lower bound.

The *ColumnAndCutGeneration* algorithm is embedded in a branch-and-bound framework. Because the proposed algorithm combines both column and cut generation, it is common to classify it as a *branch-cut-and-price* algorithm.

2.3.1. Subtour elimination constraints

Constraints (12) are SECs used to ensure that there is a path from every client to the origin hotel. Given that the pricing subproblem yields near elementary paths, as described in Section 2.4, it is rare to find subtours among a set of clients that violates the SECs. It is possible, however, to obtain a set of acyclic trips that, together, form a subtour disconnected from a portion of the graph, and SECs eliminate such cases. Also, note that the constraints are defined over the set of clients C as opposed to the entire set of vertices because, with the exception of the origin hotel, the salesman is not required to visit every hotel. Separation is performed using the *minimum cut* algorithm, a procedure similar to the one commonly used for separating SECs in classic TSPs (see Applegate et al., 2007).

2.3.2. 2-Path cuts

2-Path cuts originally introduced by Kohl et al. (1999) were designed specifically for the Vehicle Routing Problem with Time Windows. In that context, let S be a set of clients that cannot be serviced in a single route due to time window incompatibility or lack of capacity. The 2-path cut over S requires the solution to have at least two routes to service all clients in S.

In the TSPHS, to assert that a single trip cannot service a set of clients S, it is necessary to ensure that no valid trip can start at a hotel, visit all clients in S and return to a hotel, regardless of the order in which clients are visited. The separation of this cut is done by first enumerating several candidate sets S such that $\sum_{e\in\delta(S)} x_e < 4$ using the labeling algorithm described in Section 2.4.4. For each candidate S, an instance of the TSP having S plus a root node r as points is solved. The distance between r and a client $i \in S$ is defined as the shortest travel time between a hotel and i; the distances between pairs of clients in S are defined by their travel times. If the TSP tour distance plus $\sum_{i\in S} s_i$ exceeds L, cut (20) is added.

$$\min_{e\in\delta(S)} x_e \geqslant 4 \tag{20}$$

2.3.3. Limited memory subset row cuts

A family of cuts known as the Subset Row Cuts (SRC) was defined in Jepsen et al. (2008) over route variables on the master problem for the VRP. Let α_i^t denote the number of times client i is visited on trip t, such that $\alpha_i^t = \frac{1}{2}\sum_{(i,j)\in\delta(i)} a_{ij}^t$. Given a set $N \subseteq C$ and a multiplier $p, 0 < p < 1$, a (N, p)-SRC could be defined in the TSPHS as:

$$\sum_{t\in T} \lfloor p \sum_{i\in N} \alpha_i^t \rfloor \lambda_t \leqslant \lfloor p|N| \rfloor \tag{21}$$

Note that (21) is valid as it can be obtained through a Chv'atal-Gomory rounding of the corresponding constraints in (9). SRCs are non-robust, i.e., the effect of their dual variables cannot be included in the arc reduced costs, and thus, these cuts need to be explicitly treated in the SPPRC pricing subproblem. As the algorithm adds more cuts, the subproblem complexity increases, and so they need to be carefully managed to prevent the pricing from becoming intractable.

A very useful family of SRC cuts is the 3 Subset Row Cut (3-SRC), which is obtained when $|N| = 3$ and $p = 1/2$. They have been used in Baldacci et al. (2011) and Contardo and Martinelli (2014). Each 3-SRC requires an additional dimension on the label in the pricing algorithm to indicate the parity of the number of visits made to nodes in N. For that reason, only a relatively small number of 3-SRCs could be separated in these papers to keep the pricing tractable.

More recently, Pecin (2014) introduced the limited-memory (N, M, p)-Subset Row Cuts (lm-SRC). The lm-SRC requires an additional set $M, N \subseteq M \subseteq C$ and can be defined as:

$$\sum_{t\in T} \alpha(N, M, p, t)\lambda_t \leqslant \lfloor p|M| \rfloor \tag{22}$$

Function $\alpha(N, M, p, t)$ is defined by Algorithm 3.

Algorithm 3 function $\alpha(N, M, p, t)$

```
coeff ← 0, state ← 0
foreach vertex i in trip t (in order) do
    if i ∉ M then state ← 0
    else if i ∈ N then
        state ← state + p
        if state ≥ 1 then
            coeff ← coeff + 1
            state ← state − 1
return coeff
```

Observe that when $M = C$, the lm-SRC becomes identical to an SRC. However, if M is a proper subset of C, then the lm-SRC is a weakening of the SRC, as the coefficient returned by the α function will be less than or equal to that of the regular SRC. This happens because every time the trip leaves M, the variable *state* is reset to zero, potentially decreasing the returned coefficient. The labeling algorithm executes that procedural function to compute the state of each partial path. Each label must have an additional dimension to store those states. The advantage of the lm-SRC over the classic SRC is its reduced impact on the labeling algorithm when $|M| \ll |C|$. The reasons for this reduction are explained in more detail in Section 2.4.

To minimize the impact on the labeling algorithm, M should be as small as possible. Therefore, the separation of such cuts is done by first analyzing the selected columns and identifying a violated (N, p)-SRC, and then using Algorithm 4 to identify the minimal memory set M for which the lm-SRC cut would still be violated.

Algorithm 4 Calculate minimal M

$M \leftarrow N$
foreach *trip* t *that* $\lambda_t > 0$ **and** $\lfloor p \sum_{i \in N} \alpha_i^r \rfloor > 0$ **do**
  $state \leftarrow 0; Aux \leftarrow \emptyset$
  foreach *vertex* $i \in t$ *(in order)* **do**
    if $i \in N$ **then**
      $state \leftarrow state + p$
      if $state \geq 1$ **then**
        $M \leftarrow M \cup Aux, Aux \leftarrow \emptyset, state \leftarrow state - 1$
    else if $state > 0$ **then**
      $Aux \leftarrow Aux \cup i$
return M

2.4. Column generation with ng-path relaxation

The Shortest Path Problem with Resource Constraints commonly appears as pricing subproblems in column generation algorithms for routing and scheduling problems Desaulniers et al. (2005). When only elementary paths are permitted, i.e., paths that do not have subtours, the problem is commonly referred to as the Elementary Shortest Path Problem with Resource Constraints (ESPPRC). Since the ESPPRC is strongly NP-hard and usually also hard to solve in practice, it is common to relax the path elementarity constraint to obtain a problem that can be solved by pseudo-polynomial algorithms, such as dynamic programming labeling algorithms. Eventually, the BCP solution converges towards an elementary path due to the client degree and integrality constraints. However, the additional columns corresponding to non-elementary paths yield weaker lower bounds for the root node, leading to larger branch-and-bound trees. A common trade-off is allowing non-elementary paths with k-cycle elimination, that is, preventing cycles of sizes k and smaller from being formed (see Irnich and Villeneuve, 2006). Recently, a more efficient approach to the path elementarity relaxation, known as *ng*-routes, has been proposed by Baldacci et al. (2011).

For each customer $i \in C$, let $NG(i) \subset C$ be a subset of customers that have a relationship with i such that $i \in NG(i)$. In the case of the TSPHS, we define this relationship by the set of customers $j \in C$ that are closest to i, i.e., customers with the smallest $c_{ij} + s_j$. These are called the ng-sets and denote the set of customers that i can remember. For instance, when building a path P, upon reaching customer i, the set $\Pi(P)$ contains the memory of the visited customers that belong to $NG(i)$. While expanding path P from node i to j, if node j belongs to $\Pi(P)$, that expansion is not allowed. However, if $j \notin \Pi(P)$, the expansion is permitted, and $\Pi(P)$ is updated so that it remembers only the clients that belong to $NG(j)$. Clearly, if a node belongs to $NG(i)$ but does not belong to $NG(j)$, it can be revisited after passing through j, thus forming a cycle. The visited client memory set $\Pi(P)$ can for a path P, such that $C(P)$ is the set of customers on path P and i_p is the last customer on that path, can be defined as follows:

$$\Pi(P) = \left\{ i_k \in C(P) \setminus \{i_p\} : i_k \in \bigcap_{s=k+1}^{p} NG(i_s) \right\} \cup \{i_p\} \tag{23}$$

The size of the *ng*-set is an important factor in determining the quality of the solutions: the larger the set, the less likely it is for cycles to form, and the higher the bounds yielded by the SPPRC. However, increasing the size of the *ng*-set also increases the complexity of the algorithm: since $\Pi(P)$ would likely be larger, more partial paths would need to be evaluated in the labeling algorithm. Experiments in Poggi and Uchoaoggi (2014) with *ng*-sets of different sizes and observed that sets of size 8 yield lower bounds comparable to 5-cycle elimination, only in significantly less time. We experimented with sets of sizes 8, 12, and 16 in our algorithm.

2.4.1. The pricing algorithm

Labeling algorithms differ from the traditional dynamic programming algorithms because they only store (as labels) the reachable states, i.e., those that represent feasible partial paths. More importantly, while traditional dynamic programming only performs dominance over identical states, labeling algorithms perform dominance between labels corresponding to non-identical states. A label L' representing a path $P(L')$ is dominated, and therefore eliminated, if there is another label L representing a path $P(L)$ that ends in the same vertex and is not more costly and does not use more resources than $P(L')$. Labeling algorithms start from a single label representing a null path at the source vertex, initially marked as non-extended. At each step, a non-extended label L is extended to additional labels corresponding to the possible ways of adding a single arc to $P(L)$. Dominance may be used to eliminate labels, avoiding future extensions. If there are no non-extended labels, the algorithm stops, and the optimal solutions are found among the labels representing paths ending in the sink vertex. Labeling algorithms are classified as being either label-setting or label-correcting. The defining property of a label-setting algorithm is that it only extends labels that can never be dominated by another label created after that extension. The proposed labeling algorithm, which considers *ng*-paths and lm 3-SRCs, is label-correcting.

For a given *ng*-feasible partial path P, a label $L(P) = (\bar{c}(P), q(P), \Pi(P), S(P), pred(P))$ is a data structure that stores the accumulated reduced cost of path P, the total time spent, the list of nodes forbidden as immediate extensions of P, the set of states for 3-SRC cuts and a pointer to the predecessor label so that at the end of the algorithm it is possible to reconstruct the path. The total time available in a day, L, is divided into R time intervals. Labels are stored in bi-dimensional buckets $B(i, r)$, indexed by the node $i \in N'$ and the time interval that corresponds to the total time spent for that label. We define $d_{ij} = c_{ij} + \frac{s_i}{2} + \frac{s_j}{2} : (i, j) \in A$ as the amount of time consumed when navigating arc (i, j). That definition implies that $d_{ij} = d_{ji}$ if the travel times are symmetric. When the length of the intervals $l = L/R$ is greater than $\min_{(i,j) \in A}\{d_{ij}\}$, it is possible for a path to leave a node landing in a bucket that has already been processed and expanded, so that it is necessary to re-visit certain buckets and distinguish nodes that have already been processed

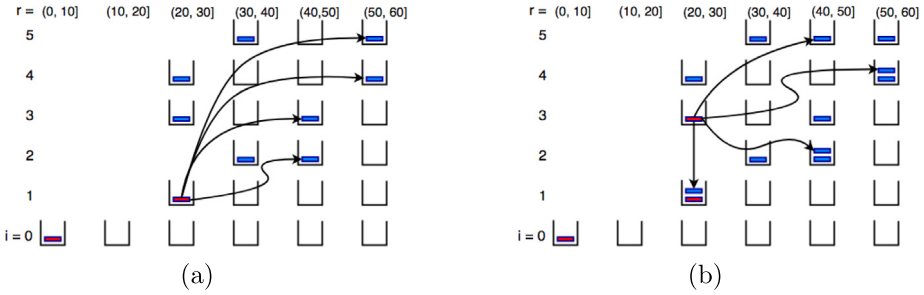

Fig. 3. Label-correcting algorithm with ranges of resource consumption.

from those that have not. Fig. 3 illustrates that situation. In the example, suppose $d_{0,1} = 20, d_{0,3} = 21$ and $d_{3,1} = 3$. The initial label in bucket $B(0,(0,10])$ has already been expanded into labels in buckets $B(1,(20,30])$ and $B(3,(20,30])$. The algorithm proceeds to first expanding labels in $B(1,(20,30])$ (Fig. 3. (a)). When labels in $B(3,(20,30])$ are expanded (Fig. 3. (b)), a new label appears in bucket $B(1,(20,30])$. As this new label may dominate labels already expanded, the resulting algorithm is label-correcting. Anyway, bucket $B(1,(20,30])$ needs to be processed again, the new labels must be expanded.

We define two different pools inside each bucket: $\mathcal{L}_{B(i,r)}$ and $\mathcal{U}_{B(i,r)}$ which correspond, respectively, to the pools of processed and unprocessed labels of bucket $B(i,r)$. They are both kept sorted by cost for efficiency reasons. We also define two auxiliary operations used in our labeling algorithm. The Pop operation returns the unprocessed label with minimal cost in $\mathcal{U}_{B(i,r)}$ and moves it to $\mathcal{L}_{B(i,r)}$. The Push operation is responsible for adding a new label to the bucket and is described by Algorithm 5.

Algorithm 5 Push(i, r, L)

```
insertLabel ← true
foreach L' ∈ L_{B(i,r)} : c(L') ≤ c(L) do
    if L' dominates L then
        insertLabel ← false; break
if insertLabel then
    foreach L' ∈ U(B(i,r)) do
        if L dominates L' then U_{B(i,r)} ← U_{B(i,r)} \ {L'}
        else if L' dominates L then
            insertLabel ← false; break
if insertLabel then
    U_{B(i,r)} ← U_{B(i,r)} ∪ {L}
```

The concept of dominance plays an important role in the efficiency of the labeling algorithm. It allows the algorithm to refrain from testing label extensions that are known in advance not to be optimal. A label $L(P_1)$ dominates a label $L(P_2)$ iff $\bar{c}(P1) \leqslant \bar{c}(P2), q(P1) \leqslant q(P2)$ and $\Pi(P1) \subseteq \Pi(P2)$. In other words, $L(P_1)$ dominates $L(P_2)$ if its cost is lower or equal to that of P_2, it did not spent more time than P_2 and all possible extensions of P_2 are also possible in P_1. The candidate label is compared against all labels in $\mathcal{L}_{B(i,r)}$ for dominance. If any of the processed labels dominate the candidate label, it is pruned. The procedure then inspects labels in $\mathcal{U}_{B(i,r)}$ and prunes the candidate label if it is dominated by an existing label or removes existing labels that are dominated by the candidate label. Finally, if the candidate label is not dominated by any labels in either $\mathcal{L}_{B(i,r)}$ or $\mathcal{U}_{B(i,r)}$, it is pushed to the pool of unprocessed labels. The pseudo-code for the labeling algorithm is shown in Algorithm 6.

Algorithm 6 Labeling algorithm for the pricing subproblem

```
U_{B(i,r)} ← ∅, L_{B(i,r)} ← ∅, ∀i ∈ N', r = 0, ..., R
U_{B(0,0)} ← {(0, 0, ∅, ∅, nil)}
for r = 0 to R do
    while ∃i ∈ N' : U_{B(i,r)} ≠ ∅ do
        foreach i ∈ N' : U_{B(i,r)} ≠ ∅ do
            while U_{B(i,r)} ≠ ∅ do
                L_1 = (c̄_1, i, q, Π_1, S_1, _) ← Pop(i, r)
                foreach j ∈ N' : (i, j) ∈ A' and x_ij is not fixed to 0 do
                    if j ∉ Π_1 and q + d_ij ≤ L then
                        c̄_2 ← c̄_1 + c̄_ij
                        S_2 ← S_1
                        for s = 1 to n_s do
                            if j ∉ M(s) then S_2[s] ← 0
                            else if j ∈ C(s) then
                                S_2[s] ← S_2[s] + p(s)
                                if S_2[s] ≥ 1 then
                                    c̄_2 ← c̄_2 − σ_s
                                    S_2[s] ← S_2[s] − 1
                        Π_2 ← (Π_1 ∩ NG(j)) ∪ {j}
                        q_2 ← q + d_ij
                        L_2 ← (c̄_2, j, q_2, Π_2, S_2, pointer to L_1)
                        r_2 ← ⌊(q_2)/l⌋
                        Push(j, r_2, L_2)
return S
```

The pricing algorithm goes through each time interval r and node i, attempting to expand each of the labels in that bucket to every node $j \in N'$. Labels can only be expanded to a node j if j is not in the list of visited clients. If the expansion is permitted, a candidate label L_2 is created, and its cost is taken by adding the originating label's cost with the reduced cost of arc (i,j). The set of 3-SRCs states is iterated so that they can be updated and have their duals deducted from the reduced cost as necessary in a similar sub-procedure to Algorithm 3. The set of visited clients is updated so that it contains node j and all visited nodes from L_1 that are also part of the ng-set of j. The time interval r_2 is calculated for the candidate label by taking the floor of the total distance at that point divided by the bucket length l. The label is then pushed to the pool of unprocessed labels.

Note that due to our usage of 3-SRCs, the dominance rule must be modified to take into account potential penalties that we might incur when that label is further expanded. For instance, consider

two labels L_1 and L_2 such that L_1 dominates L_2 according to the previously stated criteria, but L_1 has cuts in a state that would make its path incur in penalties if expanded to node k, while L_2 would not. In this case, it would be possible for L_2 to have a lower cost than that of L_1 if they were both expanded to k, and therefore, L_1 no longer dominates L_2. The dominance rule must be then modified to take that into account (see Jepsen et al., 2008):

Condition 1. A label $L(P_1)$ dominates another label $L(P_2)$ if $\bar{c}(P_1) \leqslant \bar{c}(P_2) + \sum_{1 \leqslant s \leqslant n_s : S(P_1)[s] > S(P_2)[s]} \sigma_s$, and every completion for P_2 is also a valid completion for P_1, where S is the set of states for the 3-SRC cuts and σ_s is the dual of the s^{th} 3-SRC cut.

It is now possible the explain the reasons for the advantage of lm-SRCs over SRCs. The main point for the improved algorithm efficiency is related to the dominance rule stated in Condition 1. With a traditional SRC, all nodes need to be aware of the existence of that cut, weakening the dominance rule in all buckets. As more SRCs are separated, this leads to an exponential proliferation of non-dominated labels. In lm-SRCs with a small memory set M, the dominance is only weakened in buckets of $M(s)$, allowing for more cuts to be separated before the exponential proliferation is observed (see Pecin, 2014).

Finally, a very simple heuristic is also implemented in the algorithm to allow the pricing to run faster. In the heuristic labeling algorithm, all labels within the same bucket are considered to have equivalent remaining capacity for dominance verification. That allows for fewer labels to be kept in the buckets, significantly reducing the amount of time it takes to run the algorithm. The exact labeling algorithm only needs to be run when the heuristic version returns no columns.

2.4.2. Optimizing for symmetric cases

As the bucket index increases and the algorithm gets closer to the capacity limit of a single trip, the number of labels grows considerably, resulting in longer iterations. A common approach to mitigate this problem is to have a bi-directional labeling algorithm (see Righini and Salani, 2006), where we have forward and backward labeling algorithms that generate labels each up to a certain point and a concatenation step to connect labels generated by the two labeling directions. When the graph is symmetric, doing the forward labeling up until the middle bucket index suffices, i.e., up to $\lceil \frac{R}{2} \rceil$, as the second half of the labels that would have been generated by the backward labeling algorithm is a reflection of the first half.

After running the forward labeling, partial paths generated by the labeling algorithm then need to be connected by the Algorithm 7. For each arc (i,j), the algorithm connects two labels of partial paths, one ending in node i and the other in node j. The algorithm attempts to connect labels L_1 for bucket $B(i, r_i)$ for every r_i where it is possible for an arc (i,j) to end in a time interval greater than the maximum time interval processed by the partial forward labeling algorithm, i.e., $\lceil \frac{R}{2} \rceil$. Since the buckets are indexed based on continuous time intervals, the bucket index r_j to which we attempt to connect label L_1 depends on the total capacity consumed by that label. Therefore, for each label L_1, labels L_2 in bucket $B(j, r_j)$ will be inspected, for every r_j starting at the smallest time interval at which it is possible for arc (i,j) to land when starting from interval r_i up to R. The smallest time interval r_j can be given by $\lfloor (q + d_{ij})/l \rfloor$.

Only the forward labeling is run, so the target interval is must be converted to its corresponding index on the forward labeling bucket, which can be done by the expression $R - r$, where r is the target index. That way, time interval R would correspond to index 0.

Labels L_1 and L_2 are connected iff the total distance of the resulting connection does not exceed the daily time limit, there are no common neighbors in the list of visited clients and the reduced cost of the resulting path is negative, after discounting eventual penalties from the lm-SRC cuts incurred while connecting the two labels.

Algorithm 7 Connecting partial paths

```
Run Partial Labeling
foreach (i, j) ∈ A' : x_ij is not fixed to 0 do
    for r_i = ⌊(⌈R/2⌉ × l − d_ij)/l⌋ to ⌈R/2⌉ do
        foreach L_1 = (c̄_1, i, q, Π_1, -) ∈ L_{B(i,r_i)} do
            for r_j = 0 to R − ⌊(q + d_ij)/l⌋ do
                foreach L_2 = (c̄_2, j, p, Π_2, -) ∈ L_{B(j,r_j)} do
                    c_r ← c̄_1 + c̄_2 + c̄_ij
                    if c_r ≥ 0 then break
                    if q + p + d_ij + ≤ L and Π_1 ∩ Π_2 = ∅ then
                        for s = 1, ..., nS do
                            if S_1[s] + S_2[s] ≥ 1 then c_r ← c_r − σ_s
                        if c_r < 0 then Save(pointer to L_1, pointer to L_2)
```

2.4.3. Variable fixing by reduced costs

The variable fixing procedure used in this work is similar to the one proposed in Irnich et al. (2010). For each edge $e \in E$, it seeks to eliminate variable x_e by proving that no integral solution that improves upon the current best-known solution includes such edge.

A full run of the labeling algorithm described in Section 2.4.1 is executed after the pricing. For each $x_e : e = (i,j) \in E$, the generated labels for i for each $r = 0, \ldots, R$ are tentatively concatenated to each label generated for j. If no valid paths with reduced cost smaller than the integrality gap can be obtained by concatenating those labels, then the x_e variable is fixed to 0 and is no longer processed in subsequent iterations. This procedure has great potential for reducing the number of arcs processed by the labeling algorithm and, consequently, increasing the speed of the pricing step.

2.4.4. The labeling procedure for separating 2-path cuts

Section 2.3.2 provides an overview into how 2-Path cuts are separated. Now we explain in detail the heuristic enumeration procedure that generates candidate sets $S \subset C$. Such procedure is structurally similar to the algorithm used for solving the pricing subproblem, described in Section 2.4.1. The modifications are the following.

- Only arcs corresponding to edges where $x^*_e > 0$ are considered.
- The objective function is defined over arc distances, instead of reduced costs. The daily time limit constraint is now defined over an interval of $(L, 2L]$, since only paths that exceed L are interesting, and it is extremely rare to find violated 2-Path constraints for paths of length greater than $2L$. As a consequence, a path is only extended to a hotel if the resulting distance lies in that interval.
- Only elementary paths are explored to reduce the number of false positives, where a path violates the daily time limit due to revisited clients. This is done by setting $NG(j) = C$ for each $j \in C$, i.e., $\Pi(P)$ now represents all clients in P.

- Let $\bar{\delta}^*(P)$ be the sum of the fractional values of the edges incident to the client nodes of P, i.e., $\sum_{e \in \delta(P \cap C)} x^*_e$. A label $L(P)$ where $\bar{\delta}^*(P) \geqslant 4$ is discarded. This is done because it was experimentally verified that extensions P' of P are not likely to have $\bar{\delta}^*(P') < 4$.
- Finally, the dominance criterion is the following. A label $L(P_1)$ dominates a label $L(P_2)$ iff $\Pi(P_1) = \Pi(P_2)$ and $d(P_1) \leqslant d(P_2)$.

Fig. 4 depicts part of a possible fractional solution, showing the paths corresponding to three different λ variables, all of them with a value of 0.5. Of course, all of those paths, (4, 8, 6, 7, 0), (2, 8, 6, 5, 0) and (3, 5, 7, 1), respect the time limit L. The heuristic labeling heuristic starts at hotel 4, and, using only arcs corresponding to edges where $x^*_e > 0$, finds the path $P = (4, 8, 6, 5, 7, 1)$ that exceeds L and such that $\bar{\delta}^*(P) = 2$. So, $S = \{8, 6, 5, 1\}$ is a candidate set. After verifying (by solving a TSP) that it is not possible to visit all customers in S on a single trip for all possible starting and ending hotels, the violated 2-Path cut is added.

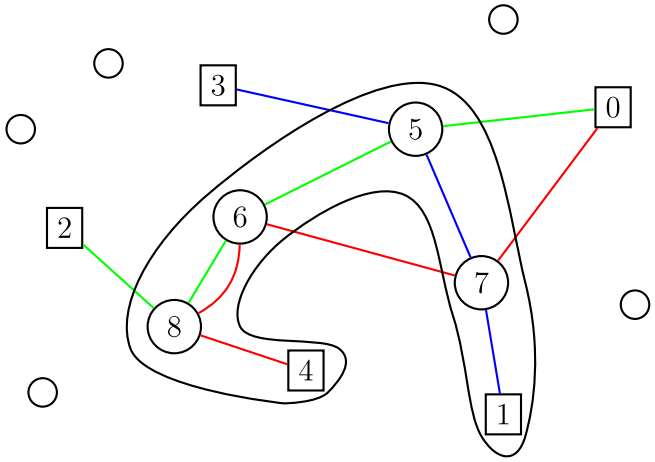

Fig. 4. Section of a sample fractional TSPHS solution.

2.5. Branching rules

Branching is done first on the number of times a hotel is used, i.e., on the $b_h : h \in H$ decision variables. Let b^*_h be a value in a solution for hotel $h \in H$. If b^*_h is fractional, the current node is branched into two subtrees:

$$b_h \leqslant \lfloor b^*_h \rfloor \quad \text{or} \quad b_h \geqslant \lceil b^*_h \rceil$$

If all hotels have integer values, branching is then done on the edges of the graph, namely decision variables $x_e : e \in E$. For every edge $e \in E$, the value of x_e is inspected. In the event of a fractional value x^*_e for edge e, the node is then further branched into the following subtrees:

$$x_e \leqslant \lfloor x^*_e \rfloor \quad \text{or} \quad x_e \geqslant \lceil x^*_e \rceil$$

The constraints in the resulting subproblems are transformed to the set partitioning based formulation as described in Expression (16) so that they can be enforced in the pricing subproblem when calculating the arc reduced costs.

When branching on b_h, the variable that has the most fractional value is selected. For branching on edges, the branching criteria selects the edge with the most fractional value weighed by the distance, i.e., the edge for which the following expression returns the largest value: $\max\{\lceil x^*_e \rceil - x^*_e, x^*_e - \lfloor x^*_e \rfloor\} \times c_e$.

3. Computational experiments

We perform computational experiments on benchmark instances from the literature and also on new instances proposed in this work. The algorithm was implemented in C++ using the Microsoft Visual Studio 2015 compiler, and the experiments where run on a Windows environment with IBM CPLEX 12.7 and an Intel Core i7 processor with 2.4 GHz and 16 GB of RAM, enforcing a time limit of five hours for every instance. The Concorde TSP Solver (Applegate et al., 2007) was used to solve the TSP at the beginning of the algorithm, to obtain the initial lower bounds. For the instances from the literature, the best-known solution was used as an upper bound.

3.1. Existing benchmark instances

Four sets of instances exist in the literature, proposed by Vansteenwegen et al. (2012). The instances in Set 1 were adapted from well-known benchmark instances for vehicle routing problems. Instances c101, c201, r101, r201, rc101 and rc201 were based on VRPTW instances Solomon (1987), while instances pr01 through pr10 came from multi-depot VRPTW instances (Cordeau et al., 1997). The TSPHS version of these instances does not consider the time windows, but the closing time of the starting hotel is used as the time limit L for the trips. Five hotels were added to the instances on the same locations as customers 1, 11, 21, 31, and 41. This way, these customers still need to be visited exactly once, but they can also be used as hotels.

Set 2 was obtained by taking the instances from Set 1 and limiting the number of clients to 10, 15, 30, and 40 clients. Aside from the depot, a single extra hotel was placed at the location of Customer 1. These are relatively small instances, for which most solutions include only one or two trips.

The third set of instances is derived from 16 well-known benchmark instances of the classic TSP and were created by adding hotels along the optimal TSP tour in intervals that respect L; in such a way, the TSP solution is likely to also be optimal for the TSPHS. Each TSP instance generates three TSPHS instances, one with 3, one with 5 and one with 10 extra hotels, besides the origin hotel that is always in the same location as the first client. Finally, Set 4 is derived from 15 of the TSP instances used in Set 3, but the hotels are located at the same coordinates of 10 of the customers: 1, 6, 11, 16, 21, 25, 31, 35, 41.

Experiments were executed for each of the sets of instances available in the literature using the best-known solution from the literature as an initial upper bound and using *ng*-sets of sizes 8 and 12. The overall performance of the proposed BCP algorithm over the sets of instances proposed in Vansteenwegen et al. (2012) is shown in Table 1. Column *Set* identifies the instance set, *Avg Clients* is its average number of clients and *hotels* is the number of hotels in each set. Column *# Inst* is the number of instances, and *# Prev Opt* shows the number of instances that had already been solved to optimality before this work. The results from the computational experiments are grouped according to the size of the *ng*-set. Columns *# Opt* and *# Feas* are the number of instances where optimal solutions and feasible solutions not proven to be optimal are found, respectively. In the remaining instances, no solution could be found within the time limit. Column *Avg Time* is the average BCP time for the instances solved to optimality. The previous best exact results were obtained by Castro et al. (2013) using Gurobi 4.6 MIP solver, receiving the value of the best-known heuristic solution as an initial upper bound and with a time limit of six hours. Only instances of Set 2 could be solved to optimality before this work.

Table 1
Summary of results for instances from literature.

Set	Avg Clients	Hotels	# Inst	# Prev Opt	ng 8			ng 12		
					# Opt	# Feas	Avg Time	# Opt	# Feas	Avg Time
1	146	6	16	0	1	4	1,615.0	1	1	988.0
2 w/ 10 clients	10	2	13	13	13	0	0.001	13	0	0.001
2 w/ 15 clients	15	2	13	13	13	0	0.2	13	0	2,432.5
2 w/ 30 clients	30	2	13	11	13	0	45.8	13	0	387.2
2 w/ 40 clients	40	2	13	10	13	0	275.3	13	0	382.1
3 w/ 3 extra hotels	189	4	16	0	12	0	365.1	12	0	1,184.4
3 w/ 5 extra hotels	189	6	16	0	13	0	628.7	13	0	1,192.1
3 w/ 10 extra hotels	189	11	16	0	12	0	2,536.9	11	1	1,340.7
4	189	10	15	0	5	1	2,445.2	5	1	2,172.6
Total					*95*	*5*	*642.3*	*94*	*3*	*1,052.3*

- Set 1 is difficult for the proposed BCP algorithm; an optimal solution was found for only one instance. Actually, for several instances, it was not possible to even solve the root node within the time limit. The difficulty with those instances, besides their large number of clients, is that they have very large values for L, leading to many clients per trip. The long trips degrade the performance of the labeling algorithm used in the pricing subproblem. The best-known solutions for those instances have an average of 40 clients per trip.
- The BCP algorithm could solve all instances of Set 2. Instances for which the TSPHS solution has a single trip are solved just by solving the corresponding TSP problem and verifying that the tour length and the sum of the service times of all clients are less than the given time limit. For instances with 30 and 40 customers, the BCP algorithm shows a significant advantage over the MIP by Castro et al. (2013), solving even the open instances in reasonable times.
- For Set 3, instances of up to 225 clients could be solved to optimality. For most instances, the algorithm was able to obtain a good lower bound, allowing the variable fixing procedure to eliminate a significant number of variables and reducing the amount of branching that was needed. However, for the larger instances, those with 280, 442, and 1002 customers, it was not possible to solve even the root node. Those big instances lead to excessively large times spent in the pricing subproblem, which reaches seconds, and even minutes, for each iteration.
- Finally, for Set 4, the BCP algorithm obtained optimal solutions for five instances of up to 100 clients. For one of those instances, the optimal solution was better than the previously best-known solution.
- Using *ng*-sets of size 8 yielded better results. Although the larger *ng*-set yields better lower bounds, it has a significant impact on the pricing algorithm, which limits the number of Subset Row Cuts that can be added.

3.2. Proposed benchmark instances

The existing benchmark instances have characteristics that limit our ability to test the proposed BCP more extensively. We can mention the following issues:

- Most instances in Set 1 permit an unrealistic large number of customers per trip, reaching 100 clients per trip. Our BCP clearly performs poorly in those conditions, and therefore, there is not much to be tested.
- The instances in Set 2 are small, and our BCP easily solved most of them. Moreover, most of the optimal solutions use only one or two trips. So, those instances are not exercising the possibilities of complex topologies in TSPHS solutions.
- Set 3 is very artificial. As mentioned before, it was designed in such a way that the optimal TSP solution is probably also the optimal TSPHS solution.

For these reasons, we decided to create a new set of instances. In those new instances, clients and hotels are randomly positioned on a 100 by 100 grid, and edge costs are obtained by taking the euclidian distances rounded to one decimal place. We created 10 instances for each combination of number of clients ($|C|$) in $\{50, 75, 100\}$, number of hotels ($|H|$) in {5,10,20} and daily time limit (L) in $\{100, 150, 200\}$. For all instances, every client has a service time of 10 units. However, we realized that almost all instances with $|H| = 5$ and $L = 100$ were infeasible, so we dropped the corresponding combinations. Therefore, our experiments included a total of 240 instances. We remark that a few of those 240 instances are still infeasible. We detect infeasibility during pre-processing by eliminating any hotel for which there is no feasible path to the origin hotel. Then, for each client, we identify their closest hotel and check whether making a trip from that hotel to the client and back respects the daily limit. If for any client it does not, the instance is deemed infeasible.

The instances are named according to their defining parameters. For example, h05_c50_l150_01 is the first instance (numbered 01) with $|C| = 50, |H| = 5$ and $L = 100$. The whole group of 10 instances is referred as h05_c50_l150. Another important information is that instances with the same number were created "incrementally". This means that h10_c50_l150_01 is created from h05_c50_l150_01 by adding 5 more hotels; h05_c100_l150_01 is created from h05_c50_l150_01 by adding 50 more customers; and so on. Moreover, instance h05_c50_l200_01 only differs from h05_c50_l150_01 by the value of L.

Since no heuristics were available to generate an initial solution, the BCP algorithm was run without any external upper bound. The lack of an initial upper bound negatively affects the algorithm's performance, sometimes forcing it to go deep in the branch-and-bound tree to find the first integer solution.

Table 2 presents a summary of the experiments executed in the new instances. Aside from the columns already described in Section 3.1 for Table 1, column # Inf shows the number of instances proved to be infeasible in each group. Experiments were executed with *ng*-sets of sizes 8, 12 and 16. The results are discussed below:

Table 2
Summary of results for proposed instances.

Set	Avg Clients	Hotels	# Inst	# Inf	ng 8			ng 12			ng 16		
					# Opt	# Feas	Avg Time	# Opt	# Feas	Avg Time	# Opt	# Feas	Avg Time
h05_c50_l150	50	5	10	4	6	0	280.7	6	0	275.8	6	0	291.5
h05_c50_l200	50	5	10	0	8	2	1,684.2	8	2	2,085.4	7	3	2,313.0
h10_c50_l100	50	10	10	5	5	0	1,337.0	5	0	1,470.8	5	0	1,514.2
h10_c50_l150	50	10	10	0	8	2	1,709.9	8	2	1,934.8	9	1	1,942.0
h10_c50_l200	50	10	10	0	10	0	1,808.6	10	0	1,323.6	9	0	3,491.7
h20_c50_l100	50	20	10	0	8	1	3,265.4	8	1	3,197.8	8	1	3,923.5
h20_c50_l150	50	20	10	0	10	0	1,554.2	10	0	1,448.9	10	0	1,960.6
h20_c50_l200	50	20	10	0	7	3	1,158.6	8	2	4,187.6	10	0	4,196.5
Total					62	8	1,667.5	63	7	2,031.4	64	5	2,615.2
h05_c75_l150	75	5	10	1	5	2	5,006.8	5	2	1,314.8	6	2	1,738.0
h05_c75_l200	75	5	10	0	3	6	1,023.3	4	5	2,691.8	2	6	8,756.0
h10_c75_l100	75	10	10	3	4	3	3,063.0	7	0	4,666.7	5	2	3,026.4
h10_c75_l150	75	10	10	0	4	4	7,040.2	9	1	6,172.7	4	5	10,719.2
h10_c75_l200	75	10	10	0	2	4	1,356.5	5	2	6,413.8	1	2	10,840.0
h20_c75_l100	75	20	10	0	2	5	3,022.5	5	1	9,058.6	1	5	342.0
h20_c75_l150	75	20	10	0	2	7	5,269.0	3	5	5,222.0	0	6	-
h20_c75_l200	75	20	10	0	4	5	6,549.2	3	5	3,634.7	3	6	7,236.3
Total					26	36	4,385.0	41	21	5,109.6	22	34	5,401.8
h05_c100_l150	100	5	10	0	1	5	110.0	2	6	3,472.0	3	2	2,386.0
h05_c100_l200	100	5	10	0	1	5	15,490.0	0	5	–	1	3	9,735.0
h10_c100_l100	100	10	10	6	0	0	–	0	1	–	0	1	-
h10_c100_l150	100	10	10	0	0	6	–	1	7	7,532.0	1	2	14,804.0
h10_c100_l200	100	10	10	0	0	4	–	0	3	–	0	2	-
h20_c100_l100	100	20	10	0	1	4	10,870.0	2	4	7,903.5	2	5	5,858.0
h20_c100_l150	100	20	10	0	0	5	–	0	6	–	0	1	-
h20_c100_l200	100	20	10	0	0	1	–	0	3	–	0	1	-
Total					3	30	8,823.3	5	35	6,056.6	7	17	6,201.9
Total					*91*	*74*	*2,679.8*	*109*	*63*	*3,373.9*	*93*	*56*	*3,544.4*

- Overall, using *ng*-sets of size 12 yields better results: 109 optimal solutions, as opposed to 91 obtained from using *ng*-8 or 93 with *ng*-16.
- The BCP performance for instances with 50 clients is very good. Even disregarding the 9 instances shown to be infeasible, 66 out of the remaining 71 instances (93%) were solved to optimality, across all *ng* sizes tested. For four instances (6%), feasible solutions could be found. The method failed in finding solutions in only one instance. With 50 clients, the performance with five hotels (88% optimal), 10 hotels (96% optimal), or 20 hotels (93% optimal) does not seem to vary significantly. Concerning the time limit value, no significant differences in performance were found for instances with $L = 100, L = 150$ or $L = 200$ (87%, 96% and 93% optimal, respectively.

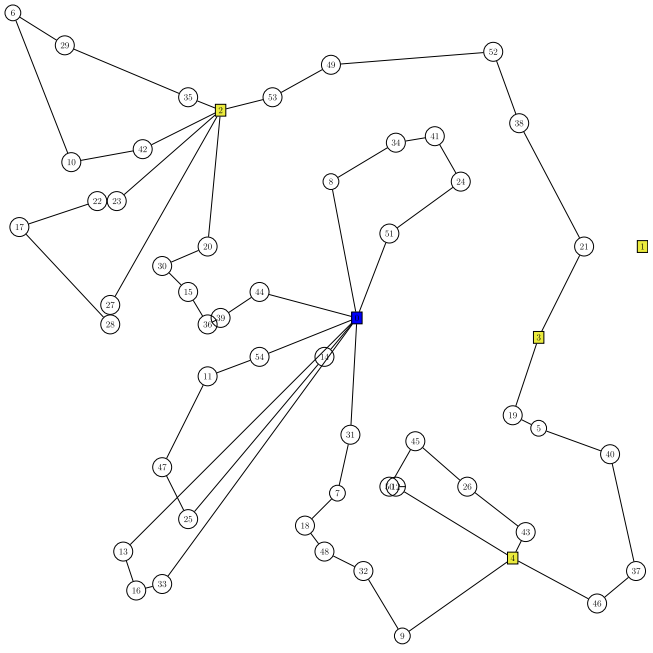

Fig. 5. Optimal solution for instance h05_c50_l150_05.

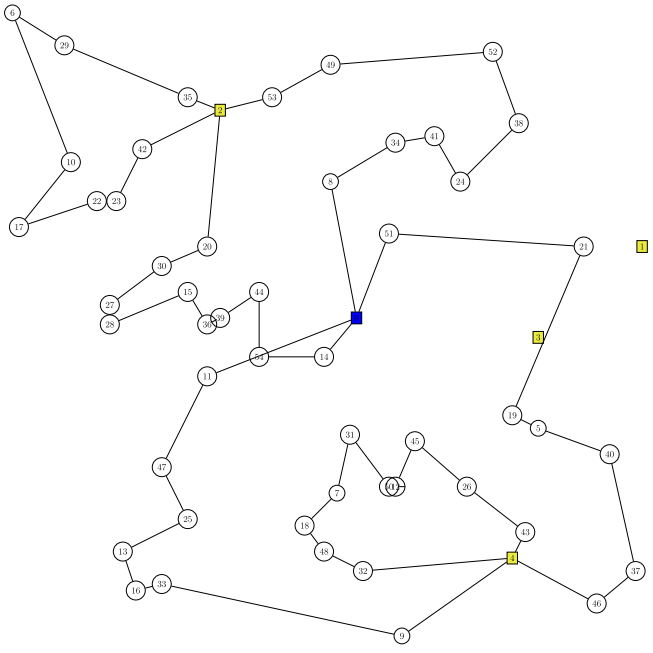

Fig. 6. Optimal solution for instance h05_c50_l200_05.

- The BCP performance for instances with 75 clients is reasonable. Disregarding the four instances shown to be infeasible, 45 out of the remaining 76 instances (59%) could be solved to optimality. For 24 instances (31%), feasible solutions could be found. The method failed on finding solution for seven instances (9%). Performance on instances with five hotels (58% optimal) was lower than those with 10 hotels (78% optimal). For instances with 20 hotels (43% optimal), however, the performance again drops significantly. As expected, instances with $L = 200$ (50% optimal) are harder than instances with $L = 100$ (71% optimal) or $L = 150$ (62% optimal). Larger values of L make the labeling algorithm used in the pricing subproblem less efficient.
- Instances with 100 clients can be too large for the BCP algorithm. Disregarding the 6 instances shown to be infeasible, only eight out of the remaining 74 instances (11%) could be solved to optimality. For 41 instances (56%), feasible solutions could be found. The method failed in finding solutions in 25 instances.

Figs. 5 and 6 shows the optimal solution of instances h05_c50_l150 and h05_c50_l200. The blue square represents the origin hotel, the yellow squares represent the other four hotels, and the circles represent the clients. As happens in almost all optimal solutions found for the new set, the depicted solutions have a quite complex topology, with some hotels visited several times, and other hotels not visited at all.

In those experiments, 2-Path cuts showed to be effective in most of the instances. However, the most significant impact on increasing the lower bounds and reducing the size of the branch-and-bound tree was brought by the 3-SRCs, which worked well for nearly all instances, some of which could even be solved at the root node. The variable fixing procedure was also crucial in reducing the total runtime of the pricing algorithm, enabling the separation of more SRCs. Detailed (instance by instance) results for all the experiments in this paper is available in the appendix.

4. Concluding remarks

The proposed BCP algorithm for the TSPHS was successful in obtaining exact solutions and improving some best-known solutions for instances in the literature with 225 customers. Furthermore, we also proposed a new set of instances designed to systematically assess how the variations in the number of clients, number of hotels, and daily time limit affect algorithm performance. Extensive experiments with those new instances showed that the BCP could solve many instances with up to 75 clients and 20 hotels, even in the absence of an external heuristic upper bound. These results demonstrate the effectiveness of the techniques used in the algorithm, adapted from recent vehicle routing works. They also motivate further studies of exact and heuristic algorithms for the TSPHS.

CRediT authorship contribution statement

Luiz Henrique Barbosa: Conceptualization, Methodology, Software, Investigation, Writing - original draft. **Eduardo Uchoa:** Conceptualization, Methodology, Investigation, Supervision, Writing - review & editing.

Acknowledgements

This study was financed in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) grant 313601/2018-6 (Produtividade 1B), and by Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ), grant E-26/202.887/2017 (Cientista do Nosso Estado).

Appendix A. Detailed results for TSPHS instances

This appendix presents detailed computational results for each of the tested instances. All solutions are presented as ordered pairs where the first element is the number of trips and the second, the tour length.

A.1. Instances from the literature

Tables 3–19 correspond to the experiments with the existing instances from the literature. For all those tables, the first three columns identify the instance, the number of clients, and the best-

Table 3
Results for Set 1 with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(9, 595.6)	8	2	–	–	562.5	576.0	578.6	578.6	8	21	0	0.5	0	504	–	18000
c201	(3, 560.0)	3	0	–	–	541.8	548.5	548.5	550.3	20	7	0	81.6	13	236	(**3**, 559.7)	18000
pr01	(2, 859.2)	2	0	–	–	821.7	858.3	858.3	858.3	32	3	110	90.5	6	4	(**2**, **859.2**)	1615
pr02	(3, 1,257.3)	3	0	–	–	1,224.4	1,242.2	1,242.2	1,242.2	15	0	10	77.2	2	117	–	18000
pr03	(4, 1,612.1)	4	0	–	–	1,560.8	1,572.5	1,572.5	1,572.5	12	0	0	65.9	0	99	–	18000
pr04	(5, 1,719.4)	5	1	–	–	–	–	–	–	0	0	0	0.0	0	0	–	0
pr05	(5, 1,835.7)	5	0	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr06	(7, 2,130.1)	6	1	–	–	–	–	–	–	0	0	0	0.0	0	0	–	0
pr07	(3, 1,062.3)	3	0	–	–	980.5	1,049.7	1,049.7	1,051.0	11	0	10	86.7	6	290	(**3**, 1,061.6)	18000
pr08	(4, 1,472.0)	4	0	–	–	1,427.9	1,435.9	1,435.9	1,435.9	10	0	0	57.3	3	69	–	18000
pr09	(5, 1,844.3)	5	1	–	–	–	–	–	–	0	0	0	0.0	0	0	–	0
pr10	(7, 2,243.5)	6	4	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
r101	(8, 704.6)	8	0	–	–	672.4	672.4	672.4	675.6	3	3	40	35.7	17	175	(**8**, 702.5)	18000
r201	(2, 643.4)	2	0	–	–	625.0	634.2	634.5	634.5	25	3	0	82.8	0	13	–	18000
rc101	(8, 674.1)	7	0	7	746	658.4	658.4	658.4	661.5	1	3	100	73.7	50	195	(**8**, 672.6)	18000
rc201	(2, 642.7)	2	0	–	–	616.3	616.3	620.8	620.8	0	0	0	0.0	0	0	–	18000
Average Gap						*4.3*	*1.9*	*1.9*	*1.7*								

Table 4
Results for Set 1 with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(9, 595.6)	8	4	–	–	575.2	582.2	585.5	585.5	11	31	0	0.0	0	64	–	18000
c201	(3, 560.0)	3	0	–	–	542.4	549.0	550.6	550.6	20	6	0	81.8	0	4	–	18000
pr01	(2, 859.2)	2	4	–	–	822.8	858.3	858.3	858.3	26	0	130	90.9	0	1	(**2**, **859.2**)	988
pr02	(3, 1,257.3)	3	0	–	–	1,225.2	1,242.2	1,242.2	1,242.2	19	0	0	78.5	0	11	–	18000
pr03	(4, 1,612.1)	4	0	–	–	1,561.7	1,572.6	1,572.6	1,572.6	11	0	0	65.6	0	5	–	18000
pr04	(5, 1,719.4)	5	4	–	–	1,652.9	1,652.9	1,652.9	1,652.9	0	0	0	0.0	0	0	–	18000
pr05	(5, 1,835.7)	5	4	–	–	–	–	–	–	0	0	0	0.0	0	0	–	3
pr06	(7, 2,130.1)	6	17	–	–	–	–	–	–	0	0	0	0.0	0	0	–	16
pr07	(3, 1,062.3)	3	10	–	–	982.0	1,049.7	1,049.7	1,049.7	11	0	0	86.0	0	173	–	18000
pr08	(4, 1,472.0)	4	0	–	–	1,428.2	1,436.1	1,436.1	1,436.1	10	0	0	57.4	0	5	–	18000
pr09	(5, 1,844.3)	5	0	–	–	1,793.2	1,793.2	1,793.2	1,793.2	0	0	0	0.0	0	0	–	18000
pr10	(7, 2,243.5)	6	75	–	–	–	–	–	–	0	0	0	0.0	0	0	–	74
r101	(8, 704.6)	8	0	–	–	672.9	673.0	673.0	673.0	2	0	0	36.1	0	597	–	18000
r201	(2, 643.4)	2	0	–	–	625.1	634.4	634.4	634.4	0	0	0	0.0	0	0	–	18000
rc101	(8, 674.1)	7	0	7	2178	658.4	658.6	658.6	659.8	2	0	20	74.2	36	397	(**8**, 673.1)	18000
rc201	(2, 642.7)	2	0	–	–	616.9	616.9	616.9	616.9	0	0	0	0.0	0	0	–	18000
Average Gap						*3.2*	*1.1*	*1.1*	*1.0*								

Table 5
Results for Set 2 with 10 clients with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(**1**, **55.1**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **55.1**)	0
pr01	(**1**, **335.6**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **335.6**)	0
pr02	(**1**, **552.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **552.9**)	0
pr03	(**1**, **437.3**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **437.3**)	0
pr04	(**1**, **376.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **376.4**)	0
pr05	(**1**, **354.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **354.9**)	0
pr06	(**1**, **486.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **486.4**)	0
pr07	(**1**, **565.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **565.2**)	0
pr08	(**1**, **482.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **482.4**)	0
pr09	(**1**, **532.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **532.5**)	0
pr10	(**1**, **349.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **349.5**)	0
r101	(**2**, **172.8**)	2	0	–	–	172.7	172.8	172.8	172.8	1	0	0	60.7	0	0	(**2**, **172.8**)	0
rc101	(**1**, **137.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **137.5**)	0
Average Gap						*0.1*	*0.0*	*0.0*	*0.0*								

Table 6
Results for Set 2 with 10 clients with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(**1**, **55.1**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **55.1**)	0
pr01	(**1**, **335.6**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **335.6**)	0
pr02	(**1**, **552.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **552.9**)	0
pr03	(**1**, **437.3**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **437.3**)	0
pr04	(**1**, **376.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **376.4**)	0
pr05	(**1**, **354.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **354.9**)	0
pr06	(**1**, **486.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **486.4**)	0
pr07	(**1**, **565.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **565.2**)	0
pr08	(**1**, **482.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **482.4**)	0
pr09	(**1**, **532.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **532.5**)	0
pr10	(**1**, **349.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **349.5**)	0
r101	(**2**, **172.8**)	2	0	–	–	172.7	172.8	172.8	172.8	1	0	0	60.7	0	0	(**2**, **172.8**)	0
rc101	(**1**, **137.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **137.5**)	0
Average Gap						*0.1*	*0.0*	*0.0*	*0.0*								

Table 7
Results for Set 2 with 15 clients with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(**2**, **102.2**)	2	0	–	–	99.1	102.2	102.2	102.2	3	1	30	72.6	0	0	(**2**, **102.2**)	1
pr01	(**1**, **444.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **444.4**)	0
pr02	(**1**, **576.6**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **576.6**)	0
pr03	(**1**, **451.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **451.9**)	0
pr04	(**1**, **507.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **507.4**)	0
pr05	(**1**, **379.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **379.2**)	0
pr06	(**1**, **520.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **520.2**)	0
pr07	(**1**, **614.3**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **614.3**)	0
pr08	(**1**, **508.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **508.2**)	0
pr09	(**1**, **591.7**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **591.7**)	0
pr10	(**1**, **456.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **456.9**)	0
r101	(**2**, **229.8**)	2	0	–	–	229.8	229.8	229.8	229.8	0	1	0	72.6	0	0	(**2**, **229.8**)	0
rc101	(**2**, **153.2**)	2	0	–	–	127.1	153.2	153.2	153.2	6	0	0	72.6	0	0	(**2**, **153.2**)	1
Average Gap						*6.7*	*0.0*	*0.0*	*0.0*								

Table 8
Results for Set 2 with 15 clients with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(**2**, **102.2**)	2	0	–	–	99.1	102.2	102.2	102.2	3	1	30	72.6	0	0	(**2**, **102.2**)	1
pr01	(**1**, **444.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **444.4**)	0
pr02	(**1**, **576.6**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **576.6**)	0
pr03	(**1**, **451.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **451.9**)	0
pr04	(**1**, **507.4**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **507.4**)	0
pr05	(**1**, **379.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **379.2**)	0
pr06	(**1**, **520.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **520.2**)	0
pr07	(**1**, **614.3**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **614.3**)	0
pr08	(**1**, **508.2**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **508.2**)	0
pr09	(**1**, **591.7**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **591.7**)	0
pr10	(**1**, **456.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **456.9**)	0
r101	(**2**, **229.8**)	2	0	–	–	229.8	229.8	229.8	229.8	0	1	0	72.6	0	0	(**2**, **229.8**)	0
rc101	(**2**, **153.2**)	2	0	–	–	127.1	153.2	153.2	153.2	6	0	0	72.6	0	0	(**2**, **153.2**)	1
Average Gap						*6.7*	*0.0*	*0.0*	*0.0*								

Table 9
Results for Set 2 with 30 clients with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(3, 163.2)	3	0	–	–	146.9	159.4	159.4	161.6	3	8	390	60.5	0	24	(**3**, **163.2**)	280
pr01	(**1**, **649.8**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **649.8**)	0
pr02	(**2**, **734.3**)	2	0	–	–	709.9	712.5	712.5	721.8	2	0	40	65.6	0	8	(**2**, **734.3**)	110
pr03	(**1**, **631.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **631.5**)	0
pr04	(**2**, **730.6**)	2	0	–	–	679.7	730.6	730.6	730.6	9	0	0	86.6	0	0	(**2**, **730.6**)	28
pr05	(**1**, **500.7**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **500.7**)	0
pr06	(**2**, **693.2**)	2	0	–	–	657.0	679.2	679.2	679.2	3	1	80	70.2	0	1	(**2**, **693.2**)	30
pr07	(**2**, **742.4**)	2	0	–	–	666.0	738.2	738.2	742.4	6	0	60	86.5	0	0	(**2**, **742.4**)	21
pr08	(**2**, **634.2**)	2	0	–	–	606.5	634.2	634.2	634.2	8	0	0	84.7	0	0	(**2**, **634.2**)	118
pr09	(**2**, **743.4**)	2	0	–	–	743.4	743.4	743.4	743.4	0	0	0	85.7	0	0	(**2**, **743.4**)	4
pr10	(**1**, **594.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **594.9**)	0
r101	(**3**, **355.2**)	3	0	–	–	349.6	350.0	353.4	355.2	1	7	60	83.8	0	0	(**3**, **355.2**)	1
rc101	(3, 405.5)	3	0	–	–	399.6	403.4	403.4	405.5	3	0	60	82.6	0	0	(**3**, **405.5**)	4
Average Gap						*4.8*	*1.1*	*1.0*	*0.5*								

Table 10
Results for Set 2 with 30 clients with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(3, 163.2)	3	0	–	–	147.5	159.9	160.1	161.3	2	10	180	61.0	0	26	(**3**, **163.2**)	458
pr01	(**1**, **649.8**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **649.8**)	0
pr02	(**2**, **734.3**)	2	0	–	–	710.7	713.4	713.4	713.4	1	0	20	55.3	54	1396	(**2**, **734.3**)	4115
pr03	(**1**, **631.5**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **631.5**)	0
pr04	(**2**, **730.6**)	2	0	–	–	679.8	730.6	730.6	730.6	9	1	20	86.0	0	0	(**2**, **730.6**)	130
pr05	(**1**, **500.7**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **500.7**)	0
pr06	(**2**, **693.2**)	2	0	–	–	657.0	679.2	679.2	679.2	3	1	20	68.0	0	1	(**2**, **693.2**)	93
pr07	(**2**, **742.4**)	2	0	–	–	666.0	738.2	738.2	742.4	6	0	40	87.1	0	0	(**2**, **742.4**)	62
pr08	(**2**, **634.2**)	2	0	–	–	606.5	634.2	634.2	634.2	8	0	0	86.8	0	0	(**2**, **634.2**)	147
pr09	(**2**, **743.4**)	2	0	–	–	743.4	743.4	743.4	743.4	0	0	0	86.1	0	0	(**2**, **743.4**)	19
pr10	(**1**, **594.9**)	1	0	–	–	–	–	–	–	0	0	0	0.0	0	0	(**1**, **594.9**)	0
r101	(**3**, **355.2**)	3	0	–	–	350.8	351.2	353.5	355.2	2	10	60	84.7	0	0	(**3**, **355.2**)	2
rc101	(3, 405.5)	3	0	–	–	400.0	403.4	403.4	405.5	2	0	30	82.0	0	0	(**3**, **405.5**)	8
Average Gap						*4.7*	*1.0*	*0.9*	*0.7*								

Table 11
Results for Set 2 with 40 clients with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(4, 266.1)	4	0	–	–	247.9	261.6	261.8	263.4	6	8	240	60.1	0	150	(**4**, **266.1**)	2516
pr01	(**2**, **722.5**)	2	0	–	–	714.5	722.5	722.5	722.5	4	0	0	89.8	0	0	(**2**, **722.5**)	39
pr02	(**2**, **867.9**)	2	0	–	–	860.9	864.2	867.9	867.9	2	19	0	88.6	0	0	(**2**, **867.9**)	23
pr03	(**2**, **843.4**)	2	0	–	–	836.2	836.5	836.5	843.4	4	0	30	89.7	0	0	(**2**, **843.4**)	30
pr04	(**2**, **769.5**)	2	0	–	–	760.8	769.5	769.5	769.5	3	0	0	89.9	0	0	(**2**, **769.5**)	28
pr05	(**2**, **668.7**)	2	0	–	–	659.9	666.6	666.6	668.7	8	0	20	89.4	0	0	(**2**, **668.7**)	48
pr06	(**2**, **752.9**)	2	0	–	–	740.3	744.0	744.0	752.9	4	0	150	88.8	0	0	(**2**, **752.9**)	74
pr07	(**2**, **840.0**)	2	0	–	–	763.4	829.1	829.1	834.3	9	1	150	81.3	0	48	(**2**, **840.0**)	665
pr08	(**2**, **713.2**)	2	0	–	–	712.4	713.2	713.2	713.2	1	1	0	89.9	0	0	(**2**, **713.2**)	24
pr09	(**2**, **823.4**)	2	0	–	–	808.2	812.5	812.5	823.4	4	0	30	89.6	0	0	(**2**, **823.4**)	47
pr10	(**2**, **720.4**)	2	0	–	–	705.1	717.5	717.5	720.4	2	0	40	89.2	0	0	(**2**, **720.4**)	52
r101	(4, 462.8)	4	0	–	–	456.5	456.5	457.2	458.4	3	9	90	82.6	0	20	(**4**, **462.8**)	21
rc101	(4, 450.3)	4	0	–	–	444.9	447.2	447.8	450.3	2	1	60	86.9	0	0	(**4**, **450.3**)	12
Average Gap						*2.3*	*0.7*	*0.7*	*0.2*								

Table 12
Results for Set 2 with 40 clients with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
c101	(4, 266.1)	4	0	–	–	252.0	262.3	262.3	263.0	5	6	120	58.8	0	152	(**4**, **266.1**)	2231
pr01	(**2**, **722.5**)	2	0	–	–	716.7	722.5	722.5	722.5	2	0	0	89.7	0	0	(**2**, **722.5**)	344
pr02	(**2**, **867.9**)	2	0	–	–	862.7	865.7	865.7	867.9	2	0	10	88.9	0	0	(**2**, **867.9**)	92
pr03	(**2**, **843.4**)	2	0	–	–	836.8	837.3	837.3	843.4	3	2	30	89.3	0	0	(**2**, **843.4**)	160
pr04	(**2**, **769.5**)	2	0	–	–	760.8	769.5	769.5	769.5	1	0	0	89.0	0	0	(**2**, **769.5**)	216
pr05	(**2**, **668.7**)	2	0	–	–	660.0	666.6	666.6	668.7	5	0	30	89.7	0	0	(**2**, **668.7**)	436
pr06	(**2**, **752.9**)	2	0	–	–	740.7	744.0	744.0	749.2	3	1	50	85.1	0	2	(**2**, **752.9**)	219
pr07	(**2**, **840.0**)	2	0	–	–	763.4	829.3	829.3	833.1	8	2	70	81.5	0	66	(**2**, **840.0**)	784
pr08	(**2**, **713.2**)	2	0	–	–	712.4	713.2	713.2	713.2	1	0	0	88.6	0	0	(**2**, **713.2**)	143
pr09	(**2**, **823.4**)	2	0	–	–	808.7	812.5	812.5	823.4	4	0	40	89.4	0	0	(**2**, **823.4**)	99
pr10	(**2**, **720.4**)	2	0	–	–	706.1	718.2	718.2	720.4	2	0	40	89.6	0	0	(**2**, **720.4**)	188
r101	(4, 462.8)	4	0	–	–	456.5	456.5	457.2	458.4	2	6	120	81.8	0	18	(**4**, **462.8**)	22
rc101	(4, 450.3)	4	0	–	–	445.6	447.2	447.2	450.3	2	0	90	85.8	0	0	(**4**, **450.3**)	33
Average Gap						*2.1*	*0.7*	*0.7*	*0.3*								

Table 13
Results for Set 3 with 3 extra hotels with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(5, 2,591.0)	4	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(4, 7,542.0)	4	0	–	–	7,542.0	7,542.0	7,542.0	7,542.0	0	0	0	90.9	0	0	(**4**, **7,542.0**)	84
ch150	(4, 6,528.0)	4	0	–	–	6,514.7	6,528.0	6,528.0	6,528.0	3	2	20	97.2	0	0	(**4**, **6,528.0**)	1026
eil101	(4, 629.0)	4	0	–	–	627.9	628.0	628.0	629.0	18	2	60	95.6	0	0	(**4**, **629.0**)	323
eil51	(4, 426.0)	4	0	–	–	423.9	426.0	426.0	426.0	3	5	80	91.0	0	0	(**4**, **426.0**)	14
eil76	(4, 538.0)	4	0	–	–	537.9	538.0	538.0	538.0	4	2	10	94.5	0	0	(**4**, **538.0**)	93
kroa100	(4, 21,282.0)	4	0	–	–	20,824.1	21,131.2	21,131.2	21,282.0	15	9	20	95.5	0	0	(**4**, **21,282.0**)	420
kroc100	(4, 20,749.0)	4	0	–	–	20,749.0	20,749.0	20,749.0	20,749.0	0	0	0	95.1	0	0	(**4**, **20,749.0**)	240
krod100	(4, 21,294.0)	4	0	–	–	21,236.0	21,254.0	21,254.0	21,294.0	5	2	50	95.5	0	0	(**4**, **21,294.0**)	229
lin105	(4, 14,379.0)	4	0	–	–	14,379.0	14,379.0	14,379.0	14,379.0	0	2	20	95.5	0	0	(**4**, **14,379.0**)	596
pcb442	(4, 50,778.0)	4	13	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(4, 259,045.0)	4	10	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(4, 108,159.0)	4	0	–	–	100,026.0	105,243.0	105,243.0	106,650.0	13	0	120	77.6	0	17	(**4**, **108,159.0**)	1086
rd100	(4, 7,910.0)	4	0	–	–	7,880.4	7,910.0	7,910.0	7,910.0	2	0	0	95.5	0	0	(**4**, **7,910.0**)	235
st70	(4, 675.0)	4	0	–	–	671.3	675.0	675.0	675.0	3	0	0	93.6	0	0	(**4**, **675.0**)	35
tsp225	(4, 3,916.0)	0	18000	–	–	–	–	–	–	–	–	–	0	0	0	–	18000
Average Gap						*1.0*	*0.3*	*0.3*	*0.1*								

Table 14
Results for Set 3 with 3 extra hotels with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(5, 2,591.0)	4	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(4, 7,542.0)	4	0	–	–	7,542.0	7,542.0	7,542.0	7,542.0	0	0	0	91.0	0	0	(**4**, **7,542.0**)	106
ch150	(4, 6,528.0)	4	3	–	–	6,528.0	6,528.0	6,528.0	6,528.0	0	1	10	96.9	0	0	(**4**, **6,528.0**)	1886
eil101	(4, 629.0)	4	0	–	–	628.0	628.0	628.0	628.0	0	0	0	94.6	0	1	(**4**, **629.0**)	445
eil51	(4, 426.0)	4	0	–	–	423.9	426.0	426.0	426.0	3	1	20	91.2	0	0	(**4**, **426.0**)	28
eil76	(4, 538.0)	4	0	–	–	538.0	538.0	538.0	538.0	1	2	10	93.9	0	0	(**4**, **538.0**)	185
kroa100	(4, 21,282.0)	4	0	–	–	20,883.0	21,138.9	21,138.9	21,282.0	14	3	40	95.3	0	0	(**4**, **21,282.0**)	729
kroc100	(4, 20,749.0)	4	0	–	–	20,749.0	20,749.0	20,749.0	20,749.0	0	0	0	95.2	0	0	(**4**, **20,749.0**)	541
krod100	(4, 21,294.0)	4	0	–	–	21,267.8	21,267.8	21,267.8	21,294.0	0	1	20	95.6	0	0	(**4**, **21,294.0**)	482
lin105	(4, 14,379.0)	4	0	–	–	14,379.0	14,379.0	14,379.0	14,379.0	0	0	0	94.9	0	0	(**4**, **14,379.0**)	1982
pcb442	(4, 50,778.0)	4	13	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(4, 259,045.0)	4	10	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(4, 108,159.0)	4	0	–	–	100,062.0	105,257.0	105,257.0	105,257.0	2	0	0	70.4	0	127	(**4**, **108,159.0**)	7276
rd100	(4, 7,910.0)	4	0	–	–	7,885.4	7,910.0	7,910.0	7,910.0	2	1	10	95.3	0	0	(**4**, **7,910.0**)	486
st70	(4, 675.0)	4	0	–	–	671.4	675.0	675.0	675.0	4	0	0	93.1	0	0	(**4**, **675.0**)	67
tsp225	(4, 3,916.0)	4	1	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
Average Gap						*0.9*	*0.3*	*0.3*	*0.2*								

Table 15
Results for Set 3 with 5 extra hotels with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(7, 2,646.0)	6	3	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(6, 7,542.0)	6	0	–	–	7,226.5	7,542.0	7,542.0	7,542.0	6	4	90	91.7	0	0	(**6**, **7,542.0**)	16
ch150	(6, 6,528.0)	6	0	–	–	6,473.3	6,515.0	6,515.0	6,528.0	20	2	240	97.2	0	0	(**6**, **6,528.0**)	1149
eil101	(6, 629.0)	6	0	–	–	627.7	628.0	628.0	629.0	4	6	100	95.9	0	0	(**6**, **629.0**)	91
eil51	(6, 426.0)	6	0	–	–	417.2	425.9	426.0	426.0	16	11	90	91.4	0	0	(**6**, **426.0**)	7
eil76	(6, 538.0)	6	0	–	–	532.5	538.0	538.0	538.0	12	7	99	93.9	0	0	(**6**, **538.0**)	46
kroa100	(6, 21,282.0)	6	0	–	–	21,114.0	21,268.5	21,268.5	21,268.5	8	4	20	95.1	0	4	(**6**, **21,282.0**)	120
kroc100	(6, 20,749.0)	6	0	–	–	20,617.8	20,722.7	20,722.7	20,749.0	2	1	80	95.7	0	0	(**6**, **20,749.0**)	103
krod100	(6, 21,294.0)	6	0	–	–	21,294.0	21,294.0	21,294.0	21,294.0	0	0	0	95.1	0	0	(**6**, **21,294.0**)	66
lin105	(6, 14,379.0)	6	0	–	–	14,077.9	14,379.0	14,379.0	14,379.0	6	0	0	95.3	0	0	(**6**, **14,379.0**)	171
pcb442	(6, 50,778.0)	6	9	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(6, 259,045.0)	6	7	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(6, 108,159.0)	6	0	–	–	104,808.0	106,158.0	106,191.0	108,159.0	18	27	1040	94.0	0	0	(**6**, **108,159.0**)	287
rd100	(6, 7,910.0)	6	0	–	–	7,903.0	7,910.0	7,910.0	7,910.0	4	3	60	95.6	0	1	(**6**, **7,910.0**)	141
st70	(6, 675.0)	6	0	–	–	656.8	674.2	674.3	675.0	18	3	88	93.3	0	0	(**6**, **675.0**)	94
tsp225	(6, 3,916.0)	6	2	–	–	3,878.9	3,895.3	3,895.3	3,901.0	22	0	10	95.3	0	8	(**6**, **3,916.0**)	5882
Average Gap						*1.4*	*0.2*	*0.2*	*0.0*								

Table 16
Results for Set 3 with 5 extra hotels with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(7, 2,646.0)	6	3	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(6, 7,542.0)	6	0	–	–	7,250.0	7,542.0	7,542.0	7,542.0	4	0	0	91.7	0	0	(**6**, **7,542.0**)	55
ch150	(6, 6,528.0)	6	0	–	–	6,480.9	6,518.3	6,518.3	6,528.0	19	1	40	97.2	0	0	(**6**, **6,528.0**)	1068
eil101	(6, 629.0)	6	0	–	–	628.4	629.0	629.0	629.0	3	2	0	95.8	0	0	(**6**, **629.0**)	110
eil51	(6, 426.0)	6	0	–	–	417.6	426.0	426.0	426.0	14	2	30	91.3	0	0	(**6**, **426.0**)	5
eil76	(6, 538.0)	6	0	–	–	532.6	538.0	538.0	538.0	10	4	60	93.7	0	0	(**6**, **538.0**)	63
kroa100	(6, 21,282.0)	6	0	–	–	21,114.0	21,268.5	21,268.5	21,268.5	8	15	80	94.9	0	4	(**6**, **21,282.0**)	166
kroc100	(6, 20,749.0)	6	14	–	–	20,617.8	20,722.7	20,722.7	20,749.0	3	5	120	95.6	0	0	(**6**, **20,749.0**)	155
krod100	(6, 21,294.0)	6	0	–	–	21,294.0	21,294.0	21,294.0	21,294.0	0	0	0	95.5	0	0	(**6**, **21,294.0**)	155
lin105	(6, 14,379.0)	6	0	–	–	14,082.0	14,379.0	14,379.0	14,379.0	9	2	40	95.6	0	0	(**6**, **14,379.0**)	566
pcb442	(6, 50,778.0)	6	9	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(6, 259,045.0)	6	7	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(6, 108,159.0)	6	0	–	–	104,859.0	106,166.0	106,166.0	108,159.0	14	50	1580	94.2	0	0	(**6**, **108,159.0**)	279
rd100	(6, 7,910.0)	6	0	–	–	7,903.0	7,910.0	7,910.0	7,910.0	1	0	0	95.8	0	0	(**6**, **7,910.0**)	165
st70	(6, 675.0)	6	0	–	–	657.3	674.2	674.3	675.0	22	4	60	93.1	0	0	(**6**, **675.0**)	163
tsp225	(6, 3,916.0)	6	1	–	–	3,883.8	3,898.0	3,898.0	3,898.0	15	0	0	94.2	0	24	(**6**, **3,916.0**)	12547
Average Gap						*1.4*	*0.2*	*0.2*	*0.0*								

Table 17
Results for Set 3 with 10 extra hotels with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(11, 2,613.0)	11	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(8, 7,864.0)	7	0	7	16	7,548.5	7,634.5	7,653.9	7,687.5	16	23	150	74.0	0	880	(**8**, **7,801.0**)	6613
ch150	(11, 6,528.0)	10	1	10	4872	6,528.0	6,528.0	6,528.0	6,528.0	0	5	40	96.9	0	50	(**11**, **6,528.0**)	4977
eil101	(11, 629.0)	10	0	10	185	626.5	627.5	628.0	628.0	2	26	174	94.8	0	2	(**11**, **629.0**)	242
eil51	(10, 426.0)	10	0	–	–	421.0	426.0	426.0	426.0	7	3	48	91.1	0	0	(**10**, **426.0**)	1
eil76	(11, 538.0)	10	0	10	4	534.7	538.0	538.0	538.0	5	5	60	93.9	0	0	(**11**, **538.0**)	8
kroa100	(11, 21,282.0)	10	0	10	7	21,164.0	21,228.0	21,228.0	21,282.0	6	4	150	95.5	0	0	(**11**, **21,282.0**)	22
kroc100	(11, 20,749.0)	10	0	10	14	20,502.0	20,749.0	20,749.0	20,749.0	7	5	68	95.3	0	0	(**11**, **20,749.0**)	28
krod100	(11, 21,294.0)	10	0	10	8	20,797.5	21,294.0	21,294.0	21,294.0	13	5	150	95.6	0	0	(**11**, **21,294.0**)	26
lin105	(10, 14,379.0)	8	0	8	42	13,634.4	14,507.3	14,507.3	14,507.3	34	5	20	37.3	158	750	–	18000
pcb442	(11, 51,774.0)	11	12	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(11, 259,045.0)	11	9	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(11, 108,159.0)	10	0	10	7	106,473.0	106,574.0	106,574.0	106,971.0	5	70	334	86.7	312	1622	(**11**, **108,159.0**)	16468
rd100	(10, 7,910.0)	9	0	9	29	7,717.9	7,910.0	7,910.0	7,910.0	12	2	30	95.7	0	0	(**10**, **7,910.0**)	45
st70	(10, 675.0)	10	0	–	–	661.0	675.0	675.0	675.0	7	0	0	93.6	0	0	(**10**, **675.0**)	4
tsp225	(11, 3,916.0)	10	3	10	1213	3,833.3	3,906.4	3,906.4	3,916.0	39	28	110	98.1	0	0	(**11**, **3,916.0**)	2009
Average Gap						*1.5*	*0.4*	*0.3*	*0.2*								

Table 18
Results for Set 3 with 10 extra hotels with ng sets of size 12.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(11, 2,613.0)	11	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(8, 7,864.0)	7	0	7	41	7,627.8	7,658.3	7,658.4	7,704.0	11	35	125	71.5	0	7154	(**8**, 7,864.0)	18000
ch150	(11, 6,528.0)	10	0	10	9715	6,528.0	6,528.0	6,528.0	6,528.0	0	1	20	96.8	0	188	(**11**, **6,528.0**)	9815
eil101	(11, 629.0)	10	0	10	119	626.5	627.5	627.5	628.0	1	5	260	94.7	0	2	(**11**, **629.0**)	152
eil51	(10, 426.0)	10	0	–	–	421.0	426.0	426.0	426.0	7	6	85	90.9	0	0	(**10**, **426.0**)	1
eil76	(11, 538.0)	10	0	10	4	535.4	538.0	538.0	538.0	1	4	60	94.5	0	0	(**11**, **538.0**)	7
kroa100	(11, 21,282.0)	10	0	10	9	21,196.4	21,282.0	21,282.0	21,282.0	8	4	90	95.6	0	0	(**11**, **21,282.0**)	17
kroc100	(11, 20,749.0)	10	0	10	16	20,502.0	20,749.0	20,749.0	20,749.0	7	7	128	95.7	0	2	(**11**, **20,749.0**)	35
krod100	(11, 21,294.0)	10	0	10	9	20,865.2	21,294.0	21,294.0	21,294.0	12	0	0	95.5	0	0	(**11**, **21,294.0**)	16
lin105	(10, 14,379.0)	8	0	8	159	13,687.7	14,515.8	14,519.1	14,519.1	35	12	0	37.4	0	1303	–	18000
pcb442	(11, 51,774.0)	11	12	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(11, 259,045.0)	11	9	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(11, 108,159.0)	10	0	10	9	106,473.0	106,574.0	106,574.0	106,971.0	3	14	150	86.3	114	559	(**11**, **108,159.0**)	2182
rd100	(10, 7,910.0)	9	0	9	41	7,718.0	7,910.0	7,910.0	7,910.0	11	3	30	95.7	0	0	(**10**, **7,910.0**)	56
st70	(10, 675.0)	10	0	–	–	661.0	675.0	675.0	675.0	9	4	60	93.6	0	0	(**10**, **675.0**)	5
tsp225	(11, 3,916.0)	10	1	10	1585	3,843.5	3,910.6	3,910.6	3,916.0	33	94	90	98.2	0	0	(**11**, **3,916.0**)	2462
Average Gap						*1.3*	*0.3*	*0.3*	*0.2*								

Table 19
Results for Set 4 with ng sets of size 8.

Instance	Best-Known Solution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(6, 2,646.0)	5	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(7, 8,642.0)	5	0	5–6	5546	8,514.6	8,533.4	8,533.6	8,550.5	11	15	150	75.5	0	714	(**7**, **8,585.0**)	6486
ch150	(6, 6,647.0)	5	1	5	574	6,233.1	6,490.1	6,490.1	6,490.1	35	0	20	74.7	0	24	–	18000
eil101	(6, 634.0)	5	0	5	77	599.0	627.5	627.5	630.2	41	34	825	91.2	0	228	–	18000
eil51	(6, 429.0)	5	0	5	8	392.5	424.0	424.0	425.7	34	11	516	87.9	0	84	(**6**, **429.0**)	646
eil76	(6, 539.0)	5	0	5	58	528.4	537.5	537.5	538.2	17	29	1334	93.7	0	28	(**6**, **539.0**)	862
kroa100	(6, 22,343.0)	5	0	5	99	21,008.8	21,306.2	21,306.2	21,320.0	14	2	20	56.1	0	164	(**6**, 22,321.0)	18000
kroc100	(6, 20,933.0)	5	0	5	105	19,923.0	20,697.9	20,789.0	20,933.0	29	28	300	96.0	0	0	(**6**, **20,933.0**)	387
krod100	(6, 21,464.0)	5	0	5	143	20,878.6	21,253.0	21,284.3	21,428.4	24	25	2489	94.0	0	12	(**6**, **21,464.0**)	3845
pcb442	(6, 54,339.0)	5	15	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(7, 290,110.0)	5	7	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(6, 118,061.0)	5	0	5	50	108,331.0	111,872.0	111,976.0	111,976.0	29	4	0	30.5	0	381	–	18000
rd100	(6, 8,244.0)	5	0	5	73	7,817.9	8,044.9	8,045.8	8,070.8	20	4	200	74.8	0	3491	–	18000
st70	(6, 723.0)	5	0	5	18	653.4	696.2	696.2	706.9	41	2	660	70.3	0	200	–	18000
tsp225	(6, 4,502.0)	5	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
Average Gap						*5.3*	*2.0*	*2.0*	*1.6*								

Table 20
Results for Set 4 with ng sets of size 12.

Instance	Best-KnownSolution	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
		Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
a280	(6, 2,646.0)	5	2	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
berlin52	(7, 8,642.0)	5	0	5–6	7112	8,523.6	8,535.8	8,535.8	8,542.0	4	8	37	74.1	0	670	(**7**, **8,585.0**)	7592
ch150	(6, 6,647.0)	5	0	5	1446	6,237.7	6,490.1	6,490.1	6,490.1	32	0	0	75.4	0	367	–	18000
eil101	(6, 634.0)	5	0	5	239	599.1	627.5	627.5	629.9	35	15	410	90.0	0	141	–	18000
eil51	(6, 429.0)	5	0	5	11	392.5	424.0	424.0	425.7	31	25	1222	87.7	0	116	(**6**, **429.0**)	551
eil76	(6, 539.0)	5	0	5	119	528.6	537.6	537.6	538.2	16	23	1003	93.9	0	32	(**6**, **539.0**)	525
kroa100	(6, 22,343.0)	5	0	5	170	21,038.3	21,306.2	21,306.2	21,306.2	12	0	0	56.4	0	145	–	18000
kroc100	(6, 20,933.0)	5	0	5	185	19,923.0	20,698.0	20,698.0	20,933.0	24	38	340	95.8	0	0	(**6**, **20,933.0**)	476
krod100	(6, 21,464.0)	5	0	5	265	20,888.1	21,269.0	21,319.2	21,428.6	21	35	1692	94.2	0	8	(**6**, **21,464.0**)	1719
pcb442	(6, 54,339.0)	5	15	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr1002	(7, 290,110.0)	5	7	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
pr76	(6, 118,061.0)	5	0	5	71	108,538.0	112,085.0	112,085.0	112,085.0	28	1	0	31.3	0	504	–	18000
rd100	(6, 8,244.0)	5	0	5	298	7,848.9	8,050.2	8,053.8	8,056.7	20	1	20	75.6	0	158	–	18000
st70	(6, 723.0)	5	0	5	33	654.2	700.1	700.1	703.9	30	1	340	64.0	0	245	(**6**, 718.0)	18000
tsp225	(6, 4,502.0)	5	1	–	–	–	–	–	–	0	0	0	0.0	0	0	–	18000
Average Gap						*5.2*	*2.0*	*1.9*	*1.6*								

Table 21
Results for proposed instances with 5 hotels, 50 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c50_l150_01	*infeasible*															
h05_c50_l150_02	8	0	8–9	1	801.1	811.2	818.9	822.2	10	10	109	9.7	0	6	(**10**, **823.8**)	8
h05_c50_l150_03	7	0	7–9	0	851.1	851.9	872.9	877.5	11	18	361	21.0	46	178	(**10**, **899.6**)	490
h05_c50_l150_04	8	0	8–10	0	978.4	987.1	988.8	988.8	5	7	0	17.8	2	2	(**11**, **988.8**)	9
h05_c50_l150_05	8	0	8–9	0	796.2	802.1	810.0	815.3	3	35	90	11.0	4	36	(**10**, **816.4**)	76
h05_c50_l150_06	*infeasible*															
h05_c50_l150_07	8	0	8–12	0	1,283.1	1,290.2	1,296.2	1,306.8	11	15	179	22.7	0	1070	(**13**, **1,325.0**)	1100
h05_c50_l150_08	*infeasible*															
h05_c50_l150_09	7	0	7–8	0	722.2	722.2	722.2	722.2	0	0	0	12.8	0	0	(**9**, **722.2**)	1
h05_c50_l150_10	*infeasible*															
Average Gap					*2.5*	*1.9*	*1.1*	*0.7*								
h05_c50_l200_01	6	0	6	1	830.2	831.1	832.4	843.9	4	16	240	30.5	8	444	(**7**, **862.3**)	1966
h05_c50_l200_02	6	0	6	7	656.1	663.4	664.1	667.6	9	9	482	0.1	60	424	(**7**, **675.4**)	1602
h05_c50_l200_03	6	0	6	28	654.8	661.0	666.1	668.0	11	10	30	1.6	0	0	(**7**, **668.0**)	30
h05_c50_l200_04	6	0	6	2	724.8	733.9	739.7	751.8	13	19	570	1.8	26	2010	(**7**, **758.1**)	9252
h05_c50_l200_05	6	0	–	–	635.0	649.1	654.1	666.3	13	13	300	42.2	2	4	(**6**, **677.1**)	135
h05_c50_l200_06	6	0	6–7	4	869.5	872.0	885.8	893.4	6	21	450	5.3	2	76	(**8**, **898.3**)	385
h05_c50_l200_07	6	0	6	2	774.5	787.8	788.1	793.2	15	5	518	2.9	0	1953	(**7**, 846.9)	18000
h05_c50_l200_08	6	0	6	2	753.9	759.8	759.8	770.4	9	3	390	1.2	242	3879	(**7**, 810.0)	18000
h05_c50_l200_09	6	0	6	17	652.7	652.7	656.9	663.8	6	7	150	0.0	0	2	(**7**, **664.3**)	23
h05_c50_l200_10	6	0	6–7	1	876.8	895.9	897.3	911.7	7	8	390	5.5	0	10	(**8**, **914.6**)	81
Average Gap					*4.4*	*3.4*	*2.9*	*1.8*								

Table 22
Results for proposed instances with 5 hotels, 50 clients with ng sets of size 12

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c50_l150_01	*infeasible*															
h05_c50_l150_02	8	0	8–9	2	801.1	811.2	818.9	822.2	10	10	68	9.7	0	6	(**10**, **823.8**)	9
h05_c50_l150_03	7	0	7–9	0	851.1	851.9	872.9	877.5	8	27	284	21.0	50	182	(**10**, **899.6**)	359
h05_c50_l150_04	8	0	8–10	3	978.4	987.1	988.8	988.8	5	2	0	17.8	2	2	(**11**, **988.8**)	14
h05_c50_l150_05	8	0	8–9	1	796.5	802.4	810.9	815.3	3	41	60	11.0	4	40	(**10**, **816.4**)	77
h05_c50_l150_06	*infeasible*															
h05_c50_l150_07	8	0	8–12	1	1,285.3	1,291.1	1,297.4	1,306.8	10	17	170	22.7	0	1078	(**13**, **1,325.0**)	1194
h05_c50_l150_08	*infeasible*															
h05_c50_l150_09	7	0	7–8	1	722.2	722.2	722.2	722.2	0	0	0	12.8	0	0	(**9**, **722.2**)	2
h05_c50_l150_10	*infeasible*															
Average Gap					*2.4*	*1.9*	*1.1*	*0.7*								
h05_c50_l200_01	6	0	6	5	831.9	832.1	833.3	843.9	3	16	300	32.1	8	440	(**7**, **862.3**)	2388
h05_c50_l200_02	6	0	6	12	656.3	663.4	664.1	667.4	9	11	240	0.1	58	458	(**7**, **675.4**)	1950
h05_c50_l200_03	6	0	6	50	654.8	661.0	666.1	668.0	10	11	30	1.6	0	0	(**7**, **668.0**)	52
h05_c50_l200_04	6	0	6	12	725.1	734.0	740.1	751.4	11	17	300	1.8	26	2014	(**7**, **758.1**)	11483
h05_c50_l200_05	6	0	–	–	637.2	651.0	654.4	665.3	12	29	210	46.4	2	4	(**6**, **677.1**)	160
h05_c50_l200_06	6	0	6–7	15	869.5	872.0	886.0	893.4	6	25	405	5.3	2	76	(**8**, **898.3**)	489
h05_c50_l200_07	6	0	6	10	774.5	788.3	788.3	792.7	11	9	180	2.8	0	2052	(**7**, 845.9)	18000
h05_c50_l200_08	6	0	6	7	754.4	759.8	759.8	770.3	9	2	330	1.5	404	3605	(**7**, 810.5)	18000
h05_c50_l200_09	6	0	6	30	652.8	652.8	657.1	663.8	7	8	120	0.0	0	2	(**7**, **664.3**)	50
h05_c50_l200_10	6	0	6–7	6	877.5	897.4	900.0	911.7	5	8	420	5.5	0	10	(**8**, **914.6**)	111
Average Gap					*4.3*	*3.3*	*2.8*	*1.8*								

Table 23
Results for proposed instances with 5 hotels, 50 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound			Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c50_l150_01	*infeasible*															
h05_c50_l150_02	8	0	8–9	4	801.1	811.2	818.9	822.2	10	10	79	9.7	0	6	(**10**, **823.8**)	12
h05_c50_l150_03	7	0	7–9	4	851.1	851.9	873.0	877.5	8	17	232	21.0	44	174	(**10**, **899.6**)	342
h05_c50_l150_04	8	0	8–10	10	978.4	987.1	988.8	988.8	5	2	0	17.8	2	2	(**11**, **988.8**)	29
h05_c50_l150_05	8	0	8–9	6	797.0	803.0	810.5	815.3	4	37	60	11.0	4	40	(**10**, **816.4**)	100
h05_c50_l150_06	*infeasible*															
h05_c50_l150_07	8	0	8–12	6	1,285.3	1,291.1	1,297.4	1,306.8	11	14	165	22.7	0	1086	(**13**, **1,325.0**)	1261
h05_c50_l150_08	*infeasible*															
h05_c50_l150_09	7	0	7–8	4	722.2	722.2	722.2	722.2	0	0	0	12.8	0	0	(**9**, **722.2**)	5
h05_c50_l150_10	*infeasible*															
Average Gap					*2.4*	*1.9*	*1.1*	*0.7*								
h05_c50_l200_01	6	0	6	17	832.6	833.2	834.3	843.9	3	12	240	32.9	12	450	(**7**, **862.3**)	3420
h05_c50_l200_02	6	0	6	35	656.3	663.6	664.1	666.8	8	9	90	0.1	132	1576	(**7**, **675.4**)	10665
h05_c50_l200_03	6	0	6	95	654.8	661.0	666.1	668.0	10	10	30	1.6	0	0	(**7**, **668.0**)	100
h05_c50_l200_04	6	0	6	48	725.1	734.2	740.4	749.1	10	19	90	1.8	134	1871	(7, 819.8)	18000
h05_c50_l200_05	6	0	–	–	640.3	651.7	655.5	665.1	12	18	150	49.5	2	20	(**6**, **677.1**)	624
h05_c50_l200_06	6	0	6–7	52	869.5	872.3	886.0	892.8	6	22	180	5.3	6	136	(**8**, **898.3**)	1126
h05_c50_l200_07	6	0	6	29	774.5	789.5	789.5	792.6	12	2	120	2.7	4	2048	(7, 851.9)	18000
h05_c50_l200_08	6	0	6	18	754.4	759.8	759.8	769.5	8	0	120	1.5	142	4751	(7, 811.9)	18000
h05_c50_l200_09	6	0	6	51	652.8	652.8	657.1	663.8	4	7	90	0.0	0	2	(**7**, **664.3**)	94
h05_c50_l200_10	6	0	6–7	25	877.6	897.4	900.0	911.6	5	8	390	5.5	0	10	(**8**, **914.6**)	162
Average Gap					*4.2*	*3.2*	*2.8*	*1.9*								

known solution, presented in bold for instances that had already been solved to optimality (all but five instances in Set 2, no instance from the other sets).

- The columns under *TSP LB* show the lower bound obtained on the number of trips by solving a TSP at the beginning of the algorithm, as described in Section 2.2, and the time spent on that.
- columns under *Infeas Trips* show the time spent by the BCP algorithm trying to find solutions with a number of trips q for which the problem was infeasible. If more than a single value of q was proved infeasible, a range with the number of days tested that were infeasible is shown. For example, consider the instance berlin52 of Set 4. The TSP LB on the number of trips is 5. The BCP algorithm spent a total of 5546 s for proving infeasibility with $q = 5$ and $q = 6$ using ng-8. The remaining 10 columns refer to the BCP run with the last tested value of q.
- The columns under *Root Lower Bounds* show the relaxation lower bound obtained at the root node of the BB tree before adding any cuts, after adding SECs, after adding 2-Path cuts and after adding the lm-SRCs. As described in Section 2.3, families of cuts are processed in order. The values shown in these columns are the last lower bound obtained before separating cuts from the subsequent family. In the case of lm-SRCs, which is the last family of cuts to be processed, the value in that column is the final root node bound.
- The columns under *Root Num Cuts* show the number of cuts of each family separated in the root node.
- Column *% Fixed* shows the percentage of the arc variables that were fixed by reduced cost at the end of the root node.
- Columns *Hotels* and *Edges* shows the number of processed nodes in the BB tree as a result of branching on the number of hotels and on the number of times an edge is traversed, until the algorithm finished with an optimal solution or reached the time limit.
- Column *Solution* shows the value of the best solution found by the BCP algorithm. Optimal solutions are presented in bold. Note that it is possible to prove that only the number of trips in the obtained solution is optimal if the BCP proved smaller values of q are infeasible. Improvements over the best-known solution are underlined.
- Finally, *Total Time* gives the time in seconds spent in the overall algorithm.

Furthermore, the last row in each table described the average percent gap from the relaxation lower bound to the best-known solution.

The results for Set 1 are presented in Table 3 and 4. This set proved to be very difficult for the proposed BCP algorithm. It was possible to obtain the optimal solution only for instance pr48 and improve best-known solutions for instances c201, pr07, r101, and rc101. For instances pr04, pr05, pr06, pr09, and pr10, it was not possible to even solve the root node of the BB tree. The main difficulty with these instances was the large number of clients per trip, which degrades the performance of the labeling algorithm used in the pricing subproblem. For the instances that could not be solved, the best-known solutions have an average of 40 clients per trip. When the TSP lower bound on the number of trips is smaller than the number of tips in the best-known solution, the tour length cannot be used as an upper bound, making the variable fixing procedure ineffective, as can be observed in instance c101.

Tables 5–12 show the results for Set 2 using 10, 15, 30 and 40 customers. Instances for which the TSPHS solution had a single trip were solved just by solving the corresponding TSP problem and verifying that the tour length and the sum of the service times of

Table 24
Results for proposed instances with 10 hotels, 50 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c50_l100_01	*infeasible*															
h10_c50_l100_02	11	0	11–14	2	793.2	810.9	820.8	822.9	16	22	90	45.6	774	4562	(**15**, **834.8**)	1897
h10_c50_l100_03	11	0	11–14	77	756.9	774.5	788.2	791.5	14	11	10	53.3	64	0	(**15**, **797.0**)	82
h10_c50_l100_04	*infeasible*															
h10_c50_l100_05	11	0	11–14	1	801.0	816.8	828.1	834.4	22	16	96	44.5	852	7978	(**15**, **851.1**)	3625
h10_c50_l100_06	*infeasible*															
h10_c50_l100_07	11	0	11–15	1	888.7	904.2	917.2	925.9	25	14	80	40.8	152	1908	(**16**, **934.5**)	1077
h10_c50_l100_08	*infeasible*															
h10_c50_l100_09	11	0	11–13	0	749.4	791.6	793.4	799.1	26	13	57	50.0	0	8	(**14**, **800.1**)	4
h10_c50_l100_10	*infeasible*															
Average Gap					*5.4*	*2.8*	*1.6*	*1.0*								
h10_c50_l150_01	7	0	7–8	0	770.9	785.6	787.4	797.3	18	21	506	39.1	30	8	(**9**, **826.2**)	173
h10_c50_l150_02	8	0	8	375	608.9	633.0	635.2	638.6	22	18	243	1.3	52	1466	(**9**, **644.0**)	2208
h10_c50_l150_03	7	0	7	1	592.4	611.8	616.8	618.0	25	19	61	13.1	42	12	(**8**, **644.3**)	277
h10_c50_l150_04	8	3	8	4	657.9	688.4	688.8	693.0	28	4	245	7.4	254	7255	(**9**, 795.5)	18000
h10_c50_l150_05	8	0	–	–	605.1	633.5	641.9	659.3	25	33	683	49.4	8	176	(**8**, **684.5**)	937
h10_c50_l150_06	8	0	8–9	3	786.0	804.7	818.0	822.2	37	11	76	10.9	6	120	(**10**, **827.0**)	86
h10_c50_l150_07	7	4	–	–	626.0	656.3	664.9	667.2	29	16	90	7.1	220	10311	(**9**, 741.9)	18000
h10_c50_l150_08	7	0	7	1	607.1	637.0	637.5	639.5	26	13	161	23.0	4	504	(**8**, **672.6**)	878
h10_c50_l150_09	7	3	–	–	593.4	605.6	612.2	617.6	22	14	171	23.7	966	4930	(**8**, **664.3**)	9114
h10_c50_l150_10	8	0	8	0	745.7	759.8	764.6	774.3	17	5	90	33.3	0	0	(**9**, **774.3**)	6
Average Gap					*8.6*	*5.4*	*4.7*	*3.9*								
h10_c50_l200_01	6	0	–	–	610.2	624.5	627.6	639.8	27	4	450	20.4	36	368	(**6**, **652.9**)	8044
h10_c50_l200_02	6	0	–	–	576.5	599.3	599.5	603.7	36	5	300	0.2	0	398	(**6**, **617.9**)	3157
h10_c50_l200_03	6	0	–	–	539.1	554.6	555.8	557.4	31	8	60	0.0	0	0	(**6**, **557.4**)	16
h10_c50_l200_04	6	0	6	246	608.0	648.1	652.8	662.0	39	9	563	0.2	0	12	(**7**, **663.6**)	308
h10_c50_l200_05	6	0	–	–	555.4	598.6	601.7	601.7	22	12	0	0.2	0	0	(**6**, **601.7**)	7
h10_c50_l200_06	6	0	–	–	640.7	646.1	648.4	650.4	26	13	240	32.4	2	50	(**6**, **658.4**)	498
h10_c50_l200_07	6	0	–	–	580.7	612.1	613.0	615.2	24	5	168	1.9	0	12	(**6**, **619.5**)	139
h10_c50_l200_08	6	0	–	–	545.5	562.1	562.1	564.0	17	0	60	0.1	8	92	(**6**, **577.0**)	279
h10_c50_l200_09	6	0	–	–	554.6	568.6	568.6	572.7	22	0	140	0.1	208	2280	(**6**, **585.9**)	5577
h10_c50_l200_10	6	0	6	8	629.4	652.4	665.2	669.0	15	16	150	1.4	2	12	(**7**, **672.0**)	61
Average Gap					*5.9*	*2.2*	*1.8*	*1.1*								

Table 25
Results for proposed instances with 10 hotels, 50 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c50_l100_01	*infeasible*															
h10_c50_l100_02	11	0	11–14	1	793.2	810.9	820.8	822.9	16	22	85	45.6	796	4720	(**15**, **834.8**)	2232
h10_c50_l100_03	11	0	11–14	51	756.9	775.3	788.2	791.5	16	10	10	53.3	64	0	(**15**, **797.0**)	54
h10_c50_l100_04	*infeasible*															
h10_c50_l100_05	11	0	11–14	0	801.3	816.8	828.1	834.4	19	18	79	44.5	856	7948	(**15**, **851.1**)	3801
h10_c50_l100_06	*infeasible*															
h10_c50_l100_07	11	0	11–15	2	888.7	904.2	917.2	925.9	25	16	87	40.8	152	2050	(**16**, **934.5**)	1259
h10_c50_l100_08	*infeasible*															
h10_c50_l100_09	11	0	11–13	0	749.4	791.6	793.4	799.1	28	13	57	50.0	0	10	(**14**, **800.1**)	8
h10_c50_l100_10	*infeasible*															
Average Gap					*5.4*	*2.8*	*1.6*	*1.0*								
h10_c50_l150_01	7	0	7–8	2	772.0	786.9	788.3	797.3	15	17	367	40.0	32	8	(**9**, **826.2**)	211
h10_c50_l150_02	8	0	8	441	609.2	633.2	635.2	638.6	25	18	210	1.3	54	1490	(**9**, **644.0**)	2512
h10_c50_l150_03	7	0	7	2	592.4	611.8	616.9	618.0	22	13	57	13.1	60	34	(**8**, **644.3**)	447
h10_c50_l150_04	8	0	8	2	657.9	688.4	688.8	693.0	30	4	171	7.4	258	5301	(**9**, 795.5)	18000
h10_c50_l150_05	8	0	–	–	605.1	633.5	641.9	659.3	24	36	642	47.2	8	170	(**8**, **684.5**)	996
h10_c50_l150_06	8	0	8–9	8	786.0	804.9	818.0	822.2	31	9	75	10.9	6	120	(**10**, **827.0**)	104
h10_c50_l150_07	8	0	8	8	626.0	656.7	665.3	667.2	30	13	65	7.1	344	7252	(**9**, 721.0)	18000
h10_c50_l150_08	7	0	7	2	607.1	637.1	637.6	639.5	20	13	127	22.9	4	508	(**8**, **672.6**)	1181
h10_c50_l150_09	7	0	7	2	594.5	607.5	613.0	617.6	18	13	155	25.4	948	3324	(**8**, **664.3**)	10021
h10_c50_l150_10	8	0	8	1	746.4	761.3	765.6	774.3	22	4	60	30.1	0	0	(**9**, **774.3**)	6
Average Gap					*8.5*	*5.3*	*4.7*	*3.9*								
h10_c50_l200_01	6	0	–	–	613.8	625.4	628.0	639.4	25	2	270	18.1	10	6	(**6**, **652.9**)	339
h10_c50_l200_02	6	0	–	–	577.6	600.0	600.0	603.2	31	7	120	0.5	0	252	(**6**, **617.9**)	2184
h10_c50_l200_03	6	0	–	–	539.1	554.6	556.9	557.4	25	9	30	0.0	0	0	(**6**, **557.4**)	17
h10_c50_l200_04	6	0	6	313	608.1	648.1	652.8	661.6	33	9	180	0.2	0	38	(**7**, **663.6**)	467
h10_c50_l200_05	6	0	–	–	559.2	600.0	601.7	601.7	26	10	0	0.3	0	0	(**6**, **601.7**)	13
h10_c50_l200_06	6	0	–	–	641.3	646.1	648.9	650.4	29	12	180	34.3	2	60	(**6**, **658.4**)	640
h10_c50_l200_07	6	0	–	–	581.7	612.8	613.3	615.2	25	5	123	2.7	0	38	(**6**, **619.5**)	478
h10_c50_l200_08	6	0	–	–	545.6	562.4	562.4	564.0	12	0	30	0.1	8	94	(**6**, **577.0**)	387
h10_c50_l200_09	6	0	–	–	555.3	568.6	568.6	572.7	23	0	120	0.1	208	2494	(**6**, **585.9**)	8602
h10_c50_l200_10	6	0	6	15	630.8	652.5	666.1	669.0	16	14	90	1.4	2	12	(**7**, **672.0**)	109
Average Gap					*5.7*	*2.2*	*1.7*	*1.2*								

Table 26
Results for proposed instances with 10 hotels, 50 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c50_l100_01	*infeasible*															
h10_c50_l100_02	11	0	11–14	1	793.2	810.9	820.8	822.9	16	22	87	45.6	802	4748	(**15**, **834.8**)	2228
h10_c50_l100_03	11	0	11–14	55	756.9	775.3	788.2	791.5	14	11	10	53.3	64	0	(**15**, **797.0**)	59
h10_c50_l100_04	*infeasible*															
h10_c50_l100_05	11	0	11–14	0	801.3	816.8	828.1	834.4	19	16	90	44.5	864	7950	(**15**, **851.1**)	3829
h10_c50_l100_06	*infeasible*															
h10_c50_l100_07	11	0	11–15	2	888.7	904.2	917.2	925.9	25	14	76	40.8	168	2312	(**16**, **934.5**)	1448
h10_c50_l100_08	*infeasible*															
h10_c50_l100_09	11	0	11–13	0	749.4	791.6	793.4	799.1	25	13	68	50.0	0	10	(**14**, **800.1**)	7
h10_c50_l100_10	*infeasible*															
Average Gap					*5.4*	*2.8*	*1.6*	*1.0*								
h10_c50_l150_01	7	0	7–8	7	772.0	787.8	788.6	797.3	15	18	318	39.7	28	8	(**9**, **826.2**)	224
h10_c50_l150_02	8	0	8	452	609.2	633.2	635.2	638.6	24	18	153	1.3	56	1476	(**9**, **644.0**)	2534
h10_c50_l150_03	7	0	7	5	592.4	611.8	616.9	618.0	22	15	57	13.2	64	28	(**8**, **644.3**)	456
h10_c50_l150_04	8	0	8	7	657.9	688.7	688.9	693.0	29	4	144	7.4	254	5267	(**9**, 795.5)	18000
h10_c50_l150_05	8	0	–	–	605.1	633.5	642.0	659.3	24	34	565	47.5	8	176	(**8**, **684.5**)	1124
h10_c50_l150_06	8	0	8–9	13	786.0	804.9	818.0	822.2	30	11	60	10.9	6	116	(**10**, **827.0**)	124
h10_c50_l150_07	8	0	8	12	626.0	657.0	665.8	667.2	30	13	35	7.1	10	396	(**9**, **676.5**)	352
h10_c50_l150_08	7	0	7	6	607.1	637.1	637.6	639.5	31	19	160	23.1	4	508	(**8**, **672.6**)	1599
h10_c50_l150_09	7	0	7	7	594.5	607.5	613.0	617.6	18	12	143	25.5	936	3276	(**8**, **664.3**)	11057
h10_c50_l150_10	8	0	8	2	746.4	761.3	765.6	774.3	16	3	60	29.8	0	0	(**9**, **774.3**)	8
Average Gap					*8.5*	*5.3*	*4.6*	*3.9*								
h10_c50_l200_01	6	0	–	–	614.7	626.0	628.8	638.0	21	2	120	21.3	26	448	(**6**, **652.9**)	11779
h10_c50_l200_02	6	0	–	–	578.2	600.0	600.0	601.6	30	0	30	0.3	0	174	(**6**, **617.9**)	1762
h10_c50_l200_03	6	0	–	–	539.1	554.6	555.8	557.4	26	4	60	0.0	0	0	(**6**, **557.4**)	36
h10_c50_l200_04	6	0	6	851	608.1	648.1	652.8	656.1	25	8	60	0.2	26	1394	(**7**, **663.6**)	7703
h10_c50_l200_05	6	0	–	–	559.2	600.0	601.7	601.7	25	10	0	0.2	0	0	(**6**, **601.7**)	31
h10_c50_l200_06	6	0	–	–	641.3	646.2	649.0	650.3	26	12	120	35.7	2	226	(**6**, **658.4**)	8266
h10_c50_l200_07	6	0	–	–	581.7	612.8	613.3	614.8	25	5	30	2.4	0	24	(**6**, **619.5**)	702
h10_c50_l200_08	6	0	–	–	545.6	562.4	562.4	564.0	12	0	30	0.1	8	100	(**6**, **577.0**)	970
h10_c50_l200_09	6	0	–	–	555.3	568.6	568.6	572.7	26	0	120	0.1	250	683	–	18000
h10_c50_l200_10	6	0	6	41	630.8	652.5	666.1	669.0	15	16	95	1.4	2	14	(**7**, **672.0**)	176
Average Gap					*5.6*	*2.1*	*1.7*	*1.3*								

Table 27
Results for proposed instances with 20 hotels, 50 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c50_l100_01	11	0	11–12	28	568.1	606.6	608.0	611.3	57	10	7	34.5	156	230	(**13**, **615.7**)	297
h20_c50_l100_02	12	5	–	–	621.3	651.5	654.1	657.2	75	22	190	32.4	664	12481	(**13**, 726.1)	18000
h20_c50_l100_03	11	0	11	0	564.5	606.1	609.0	623.8	81	27	219	43.4	268	2	(**12**, **656.2**)	874
h20_c50_l100_04	12	0	12	0	633.1	691.7	698.4	701.0	59	16	88	35.2	134	2166	(**13**, **720.0**)	3354
h20_c50_l100_05	11	0	11–12	0	627.5	675.1	683.4	691.4	50	18	131	30.8	10	11509	–	18000
h20_c50_l100_06	11	0	11–12	0	628.0	669.8	671.7	675.3	53	27	198	35.6	1170	7106	(**13**, **690.2**)	6955
h20_c50_l100_07	11	0	11–14	0	822.2	850.1	863.3	872.7	77	12	154	37.8	44	2744	(**15**, **889.1**)	2388
h20_c50_l100_08	11	0	11–12	8	576.8	630.3	641.0	645.5	56	15	73	34.3	112	3448	(**13**, **664.1**)	2972
h20_c50_l100_09	11	0	11–14	0	814.5	839.1	841.1	843.0	34	6	11	43.2	76	690	(**15**, **849.5**)	438
h20_c50_l100_10	11	0	11–13	0	704.8	737.5	737.5	737.5	14	0	0	42.0	1798	11134	(**14**, **770.8**)	8845
Average Gap					*9.6*	*4.1*	*3.5*	*2.9*								
h20_c50_l150_01	7	0	7	1	518.6	565.0	565.7	569.5	66	7	402	0.5	232	736	(**8**, **583.1**)	3530
h20_c50_l150_02	8	0	–	–	551.2	590.5	591.2	592.8	57	6	94	0.6	40	4	(**8**, **597.3**)	126
h20_c50_l150_03	7	0	7	2	495.8	541.8	547.5	547.5	56	5	0	1.6	178	552	(**8**, **552.4**)	4878
h20_c50_l150_04	8	0	8	891	571.0	624.6	624.8	626.5	56	10	326	2.0	208	510	(**9**, **630.1**)	2923
h20_c50_l150_05	8	0	–	–	550.5	603.7	608.9	614.8	46	14	450	4.1	10	200	(**8**, **615.8**)	634
h20_c50_l150_06	8	0	–	–	544.2	579.4	580.0	586.6	61	10	176	2.6	136	100	(**8**, **601.8**)	1042
h20_c50_l150_07	8	0	8	62	596.0	643.2	651.4	654.5	62	14	133	5.3	210	114	(**9**, **660.9**)	561
h20_c50_l150_08	7	0	7	1	521.8	567.8	568.3	570.0	58	11	69	1.5	0	584	(**8**, **586.3**)	1013
h20_c50_l150_09	7	0	7	1	608.2	638.5	638.9	644.2	56	10	380	37.2	0	194	(**8**, **670.4**)	825
h20_c50_l150_10	8	0	–	–	578.0	609.4	609.9	613.7	39	7	90	16.8	0	0	(**8**, **613.7**)	10
Average Gap					*9.4*	*2.4*	*2.0*	*1.5*								
h20_c50_l200_01	6	0	–	–	500.8	548.2	548.5	551.0	70	3	360	0.0	64	4	(**6**, **558.7**)	941
h20_c50_l200_02	6	0	–	–	539.5	579.0	579.5	581.8	65	3	210	0.0	38	0	(**6**, **583.7**)	427
h20_c50_l200_03	6	0	–	–	483.8	527.2	527.4	531.6	77	8	180	0.0	12	0	(**6**, **539.3**)	222
h20_c50_l200_04	5	17	–	–	570.8	623.3	623.3	626.6	61	4	270	7.7	0	819	(**6**, 661.8)	18000
h20_c50_l200_05	6	0	–	–	523.9	580.5	583.2	587.0	53	11	210	0.1	0	172	(**6**, **592.5**)	1222
h20_c50_l200_06	5	60	–	–	532.0	567.6	571.3	575.6	58	9	240	0.3	4	1021	(**6**, 617.4)	18000
h20_c50_l200_07	6	0	–	–	563.3	606.4	607.1	608.5	46	10	30	1.5	0	10	(**6**, **612.6**)	47
h20_c50_l200_08	6	0	–	–	506.3	546.9	547.0	551.1	70	8	420	0.0	36	586	(**6**, **554.3**)	2785
h20_c50_l200_09	6	0	–	–	556.1	582.0	583.6	589.6	86	10	570	0.1	0	716	(**6**, **602.2**)	2466
h20_c50_l200_10	5	13	–	–	559.9	590.0	590.3	596.7	51	17	270	0.5	32	1421	(**6**, 656.7)	18000
Average Gap					*9.3*	*2.2*	*2.0*	*1.4*								

Table 28
Results for proposed instances with 20 hotels, 50 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c50_l100_01	11	0	11–12	28	568.7	606.6	608.0	611.3	45	10	7	34.5	152	238	(**13**, **615.7**)	277
h20_c50_l100_02	11	0	11–12	3	621.3	651.5	654.1	657.2	85	21	116	32.4	488	9410	(**13**, 726.1)	18000
h20_c50_l100_03	11	0	11	0	564.5	606.1	609.0	623.8	81	26	193	49.0	252	2	(**12**, **656.2**)	434
h20_c50_l100_04	12	0	12	0	633.1	691.7	698.5	701.0	59	16	71	35.2	138	2154	(**13**, **720.0**)	3343
h20_c50_l100_05	11	0	11–12	0	627.5	675.1	683.4	691.4	55	19	142	30.8	10	8723	–	18000
h20_c50_l100_06	11	0	11–12	0	628.0	669.9	671.7	675.3	55	26	206	35.6	1178	7114	(**13**, **690.2**)	6869
h20_c50_l100_07	11	0	11–14	0	822.2	850.1	863.3	872.7	70	12	111	37.8	40	2740	(**15**, **889.1**)	2316
h20_c50_l100_08	11	0	11–12	10	576.8	630.3	641.0	645.5	64	15	66	34.3	100	3328	(**13**, **664.1**)	2824
h20_c50_l100_09	11	0	11–14	0	814.5	839.1	841.1	843.0	36	6	11	43.2	76	670	(**15**, **849.5**)	401
h20_c50_l100_10	11	0	11–13	0	704.8	737.5	737.5	737.5	14	0	0	42.0	1806	11488	(**14**, **770.8**)	9118
Average Gap					*9.6*	*4.1*	*3.5*	*2.9*								
h20_c50_l150_01	7	0	7	3	518.8	565.0	565.8	569.5	77	8	320	0.5	226	734	(**8**, **583.1**)	3622
h20_c50_l150_02	8	0	–	–	552.2	591.1	591.2	592.8	51	5	107	0.6	40	4	(**8**, **597.3**)	128
h20_c50_l150_03	7	0	7	3	495.8	541.8	547.5	547.5	49	5	0	1.6	218	560	(**8**, **552.4**)	3238
h20_c50_l150_04	8	0	8	1031	571.0	624.7	624.9	626.5	65	9	320	2.0	220	486	(**9**, **630.1**)	3130
h20_c50_l150_05	8	0	–	–	551.1	603.7	608.9	614.8	45	19	402	4.4	10	196	(**8**, **615.8**)	708
h20_c50_l150_06	8	0	–	–	544.2	579.5	580.0	586.6	63	10	161	2.7	134	94	(**8**, **601.8**)	1046
h20_c50_l150_07	8	0	8	75	596.0	643.4	651.8	654.5	66	13	150	5.3	208	152	(**9**, **660.9**)	615
h20_c50_l150_08	7	0	7	3	521.8	568.0	568.3	570.0	53	14	108	1.5	0	584	(**8**, **586.3**)	1106
h20_c50_l150_09	7	0	7	2	608.2	638.5	638.9	644.2	63	12	317	38.7	0	194	(**8**, **670.4**)	884
h20_c50_l150_10	8	0	–	–	578.1	609.4	609.9	613.7	34	7	90	17.4	0	0	(**8**, **613.7**)	12
Average Gap					*9.4*	*2.4*	*2.0*	*1.5*								
h20_c50_l200_01	6	0	–	–	501.0	548.4	548.5	550.8	68	3	90	0.0	74	8	(**6**, **558.7**)	1184
h20_c50_l200_02	6	0	–	–	539.5	579.1	579.5	580.9	78	3	90	0.0	40	10	(**6**, **583.7**)	676
h20_c50_l200_03	6	0	–	–	483.8	527.3	527.5	530.5	63	5	60	0.0	48	248	(**6**, **539.3**)	3406
h20_c50_l200_04	6	0	–	–	571.3	623.5	623.5	626.4	55	0	120	8.0	0	1003	(**6**, 661.8)	18000
h20_c50_l200_05	6	0	–	–	524.5	580.8	583.1	586.0	47	13	90	0.1	0	1108	(**6**, **592.5**)	16423
h20_c50_l200_06	6	0	–	–	532.0	568.3	571.5	575.3	59	8	120	0.1	4	1846	(**6**, 603.3)	18000
h20_c50_l200_07	6	0	–	–	563.3	606.4	607.2	608.5	53	5	30	2.0	0	10	(**6**, **612.6**)	64
h20_c50_l200_08	6	0	–	–	506.3	546.9	547.0	550.6	60	8	150	0.0	2	476	(**6**, **554.3**)	4405
h20_c50_l200_09	6	0	–	–	556.1	582.0	583.6	589.1	61	8	270	0.2	0	726	(**6**, **602.2**)	2771
h20_c50_l200_10	6	0	–	–	560.1	590.1	590.3	594.6	52	9	90	0.4	0	318	(**6**, **607.1**)	4572
Average Gap					*9.3*	*2.2*	*2.0*	*1.5*								

Table 29
Results for proposed instances with 20 hotels, 50 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c50_l100_01	11	0	11–12	28	568.7	606.6	608.0	611.3	55	10	7	34.5	96	136	(**13**, **615.7**)	165
h20_c50_l100_02	11	0	11–12	2	621.3	651.5	654.1	657.2	73	21	112	32.4	654	10422	(**13**, 695.5)	18000
h20_c50_l100_03	11	0	11	0	564.5	606.1	609.0	623.8	74	26	185	49.7	272	2	(**12**, **656.2**)	543
h20_c50_l100_04	12	0	12	0	633.1	691.7	698.5	701.0	65	15	67	35.2	134	2144	(**13**, **720.0**)	3617
h20_c50_l100_05	11	0	11–12	0	627.5	675.1	683.4	691.4	55	17	110	30.8	10	9684	–	18000
h20_c50_l100_06	11	0	11–12	0	628.0	669.9	671.7	675.3	55	24	193	35.6	1162	7102	(**13**, **690.2**)	7237
h20_c50_l100_07	11	0	11–14	0	822.2	850.1	863.3	872.7	74	12	122	37.8	44	2746	(**15**, **889.1**)	2386
h20_c50_l100_08	11	0	11–12	10	576.8	630.3	641.0	645.5	58	15	67	34.3	106	3272	(**13**, **664.1**)	2775
h20_c50_l100_09	11	0	11–14	0	814.5	839.1	841.1	843.0	37	6	11	43.2	82	738	(**15**, **849.5**)	421
h20_c50_l100_10	11	0	11–13	0	704.8	737.5	737.5	737.5	14	0	0	42.0	2984	16696	(**14**, **770.8**)	14244
Average Gap					*9.6*	*4.1*	*3.5*	*2.9*								
h20_c50_l150_01	7	0	7	9	518.8	565.0	565.8	569.5	84	8	386	0.5	230	742	(**8**, **583.1**)	4399
h20_c50_l150_02	8	0	–	–	552.2	591.1	591.2	592.8	57	5	108	0.6	40	2	(**8**, **597.3**)	213
h20_c50_l150_03	7	0	7	14	495.8	541.8	547.5	547.5	49	5	0	1.6	204	566	(**8**, **552.4**)	4635
h20_c50_l150_04	8	0	8	1655	571.0	624.9	625.3	626.5	53	7	228	2.0	208	362	(**9**, **630.1**)	4716
h20_c50_l150_05	8	0	–	–	551.1	603.7	608.9	614.8	46	19	389	4.1	10	148	(**8**, **615.8**)	761
h20_c50_l150_06	8	0	–	–	544.2	579.5	580.0	586.6	64	8	120	2.6	138	96	(**8**, **601.8**)	1432
h20_c50_l150_07	8	0	8	100	596.5	643.8	652.2	654.5	58	10	120	5.3	158	92	(**9**, **660.9**)	580
h20_c50_l150_08	7	0	7	8	521.8	568.0	568.3	570.0	52	14	102	1.5	0	596	(**8**, **586.3**)	1440
h20_c50_l150_09	7	0	7	6	608.2	638.5	638.9	644.2	56	12	278	37.0	0	200	(**8**, **670.4**)	1415
h20_c50_l150_10	8	0	–	–	578.1	609.4	609.9	613.7	34	7	90	18.4	0	0	(**8**, **613.7**)	15
Average Gap					*9.4*	*2.4*	*2.0*	*1.5*								
h20_c50_l200_01	6	0	–	–	501.0	548.4	548.5	550.1	68	3	30	0.0	60	44	(**6**, **558.7**)	2776
h20_c50_l200_02	6	0	–	–	539.5	579.1	579.5	579.9	51	3	30	0.0	56	28	(**6**, **583.7**)	1588
h20_c50_l200_03	6	0	–	–	483.9	527.3	527.5	527.5	50	4	0	0.0	86	130	(**6**, **539.3**)	4000
h20_c50_l200_04	6	0	–	–	571.4	623.5	623.5	624.2	56	1	30	8.4	164	210	(**6**, **646.3**)	8761
h20_c50_l200_05	6	0	–	–	526.0	580.8	583.1	584.6	48	12	30	0.1	0	888	(**6**, **592.5**)	13646
h20_c50_l200_06	6	0	–	–	534.0	568.8	571.7	573.4	48	7	30	0.1	96	112	(**6**, **585.5**)	2212
h20_c50_l200_07	6	0	–	–	563.3	606.4	607.2	608.4	52	7	30	1.1	0	10	(**6**, **612.6**)	148
h20_c50_l200_08	6	0	–	–	506.3	546.9	547.0	549.6	61	8	45	0.0	40	722	(**6**, **554.3**)	5248
h20_c50_l200_09	6	0	–	–	556.1	582.0	583.6	588.2	63	10	120	0.1	0	546	(**6**, **602.2**)	3101
h20_c50_l200_10	6	0	–	–	560.1	590.1	590.3	594.1	49	11	60	0.4	20	0	(**6**, **607.1**)	485
Average Gap					*9.2*	*2.2*	*2.0*	*1.7*								

Table 30
Results for proposed instances with 5 hotels, 75 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c75_l150_01	10	0	10–13	1	1,222.7	1,222.7	1,234.1	1,247.9	4	36	450	20.7	0	0	(**14**, **1,247.9**)	43
h05_c75_l150_02	10	0	10–12	4	1,023.4	1,033.7	1,044.2	1,057.2	5	30	872	15.7	24	618	–	18000
h05_c75_l150_03	10	0	10–12	9	987.6	987.6	1,007.4	1,014.3	1	22	300	14.0	40	569	–	18000
h05_c75_l150_04	10	0	10–12	4	998.1	998.7	1,005.7	1,012.6	4	46	686	12.9	60	1302	(**13**, **1,016.7**)	13355
h05_c75_l150_05	10	0	10–11	4	880.5	883.9	889.8	906.1	10	44	660	9.3	58	394	(**12**, **916.8**)	5119
h05_c75_l150_06	10	0	10–13	8	1,147.6	1,147.6	1,165.4	1,173.1	0	16	105	17.3	4	5817	(**14**, 1,229.3)	18000
h05_c75_l150_07	10	2	10–13	2	1,205.3	1,208.1	1,213.9	1,234.3	9	26	750	19.4	488	392	(**14**, **1,236.6**)	6367
h05_c75_l150_08	10	0	10–12	10	994.0	998.3	1,000.6	1,011.5	2	22	554	13.2	24	1271	(**13**, 1,085.6)	18000
h05_c75_l150_09	10	1	10–14	6	1,298.0	1,301.4	1,308.8	1,316.9	7	15	274	22.0	8	6	(**15**, **1,322.9**)	150
h05_c75_l150_10	*infeasible*															
Average Gap					*3.5*	*3.3*	*2.5*	*1.4*								
h05_c75_l200_01	8	0	8	10	893.9	897.6	908.1	920.5	6	37	180	1.5	36	200	(**9**, **927.9**)	1814
h05_c75_l200_02	8	0	8	11	831.9	846.3	848.0	856.9	8	32	180	1.0	32	314	(**9**, 999.5)	18000
h05_c75_l200_03	7	0	7–8	47	796.2	796.2	801.0	806.3	2	30	180	0.6	13	902	(**9**, 872.2)	18000
h05_c75_l200_04	8	2	8	11	823.2	826.5	830.6	839.9	6	28	270	1.2	12	36	(**9**, **850.7**)	761
h05_c75_l200_05	8	0	8	14333	770.2	770.2	781.6	792.1	12	38	180	0.1	81	770	(**9**, 838.3)	18000
h05_c75_l200_06	8	0	8	14	877.9	878.3	884.6	894.5	7	40	210	0.3	14	8	(**9**, **902.0**)	495
h05_c75_l200_07	8	0	8	10	898.0	900.4	904.5	917.0	9	23	150	1.0	4	739	(**9**, 959.8)	18000
h05_c75_l200_08	7	0	7–8	195	797.0	802.9	804.5	816.1	11	37	120	0.6	8	527	(**9**, 936.5)	18000
h05_c75_l200_09	8	0	8	11	884.0	885.1	885.8	896.0	4	8	90	1.7	5	882	–	18000
h05_c75_l200_10	7	0	7–9	27	1,069.6	1,079.2	1,089.3	1,094.8	11	27	90	2.1	7	639	(**10**, 1,188.0)	18000
Average Gap					*4.4*	*4.0*	*3.4*	*2.3*								

Table 31
Results for proposed instances with 5 hotels, 75 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c75_l150_01	10	0	10–13	5	1,223.3	1,223.3	1,234.2	1,247.9	4	37	270	20.7	0	0	(**14**, **1,247.9**)	28
h05_c75_l150_02	10	0	10–12	8	1,023.7	1,034.5	1,045.1	1,057.2	4	35	685	15.7	14	1058	–	18000
h05_c75_l150_03	10	0	10–12	10	989.3	989.3	1,010.1	1,014.3	0	22	190	14.0	42	817	–	18000
h05_c75_l150_04	10	0	10–12	8	1,000.5	1,000.5	1,006.2	1,012.6	4	34	340	12.9	24	230	(**13**, **1,016.7**)	1175
h05_c75_l150_05	10	0	10–11	8	882.2	885.6	891.2	906.1	9	32	600	9.3	68	378	(**12**, **916.8**)	2259
h05_c75_l150_06	10	0	10–13	14	1,150.0	1,150.0	1,165.4	1,173.1	0	18	123	17.3	4	7847	(**14**, 1,229.3)	18000
h05_c75_l150_07	10	0	10–13	10	1,210.1	1,211.8	1,212.6	1,234.3	10	30	480	19.0	72	352	(**14**, **1,236.6**)	2910
h05_c75_l150_08	10	0	10–12	17	994.2	998.6	1,000.6	1,011.5	2	21	292	13.2	100	1987	(**13**, 1,095.1)	18000
h05_c75_l150_09	10	0	10–14	12	1,304.1	1,307.9	1,310.9	1,316.9	5	9	148	22.0	10	24	(**15**, **1,322.9**)	202
h05_c75_l150_10	*infeasible*															
Average Gap					*3.4*	*3.1*	*2.5*	*1.4*								
h05_c75_l200_01	8	0	8	17	897.8	902.1	909.7	919.6	4	22	120	1.6	28	391	(**9**, 1,007.3)	18000
h05_c75_l200_02	8	0	8	27	833.7	846.6	848.5	855.3	9	32	90	1.0	44	618	(**9**, 875.5)	18000
h05_c75_l200_03	7	0	7–8	75	797.1	797.1	803.4	805.8	0	26	60	0.6	91	611	(**9**, 856.6)	18000
h05_c75_l200_04	8	0	8	28	823.5	826.5	830.6	836.4	4	20	120	1.2	18	66	(**9**, **850.7**)	1716
h05_c75_l200_05	8	0	–	–	772.7	773.1	781.5	788.9	5	37	60	24.2	85	969	–	18000
h05_c75_l200_06	8	0	8	32	878.2	878.5	885.3	886.2	2	25	30	0.3	78	324	(**9**, **902.0**)	6844
h05_c75_l200_07	8	0	8	26	901.9	903.2	906.5	911.8	4	20	90	0.9	26	605	(**9**, 986.2)	18000
h05_c75_l200_08	7	0	7–8	180	802.0	806.6	808.4	809.6	5	32	30	0.6	14	22	(**9**, **816.1**)	732
h05_c75_l200_09	8	0	8	33	884.5	885.8	886.4	894.2	5	10	60	1.7	16	50	(**9**, **910.2**)	1475
h05_c75_l200_10	7	0	7–9	64	1,071.1	1,080.8	1,089.7	1,096.0	11	15	75	2.1	3	475	(**10**, 1,210.7)	18000
Average Gap					*4.2*	*3.8*	*3.3*	*2.7*								

Table 32
Results for proposed instances with 5 hotels, 75 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound			Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c75_l150_01	10	0	10–13	14	1,224.6	1,224.8	1,235.9	1,247.9	6	38	210	20.7	0	0	(**14**, **1,247.9**)	38
h05_c75_l150_02	10	0	10–12	26	1,023.9	1,034.7	1,045.1	1,057.2	5	35	648	15.7	151	2585	(**13**, 1,103.3)	18000
h05_c75_l150_03	10	0	10–12	26	989.5	989.5	1,010.5	1,014.3	1	33	237	14.0	40	701	–	18000
h05_c75_l150_04	10	0	10–12	27	1,000.6	1,000.6	1,006.3	1,012.6	4	29	430	12.9	32	298	(**13**, **1,016.7**)	1731
h05_c75_l150_05	10	0	10–11	23	884.5	886.6	891.6	906.1	9	32	390	9.3	38	336	(**12**, **916.8**)	1581
h05_c75_l150_06	10	0	10–13	42	1,150.0	1,150.0	1,165.5	1,173.1	0	16	119	17.3	6	7312	(**14**, 1,229.3)	18000
h05_c75_l150_07	10	0	10–13	28	1,211.8	1,213.8	1,215.3	1,234.3	6	14	391	19.0	70	478	(**14**, **1,236.6**)	5977
h05_c75_l150_08	10	0	10–12	57	994.2	998.6	1,000.6	1,011.5	2	20	303	13.2	26	64	(**13**, **1,017.1**)	935
h05_c75_l150_09	10	0	10–14	39	1,304.1	1,307.9	1,310.9	1,316.9	5	9	148	22.0	8	4	(**15**, **1,322.9**)	166
h05_c75_l150_10	*infeasible*															
Average Gap					*3.3*	*3.0*	*2.4*	*1.4*								
h05_c75_l200_01	8	0	8	56	898.1	902.1	910.5	916.2	6	22	30	1.6	44	250	(**9**, **927.9**)	11348
h05_c75_l200_02	8	0	8	99	833.7	846.8	848.6	849.3	6	23	30	1.0	4	684	–	18000
h05_c75_l200_03	7	0	7–8	238	798.3	798.3	803.5	804.5	0	14	30	0.6	15	1160	(**9**, 828.6)	18000
h05_c75_l200_04	8	0	8	119	823.5	827.3	831.2	835.1	4	27	45	1.2	82	415	(**9**, 882.3)	18000
h05_c75_l200_05	8	0	–	–	773.2	773.5	782.1	786.5	4	37	30	16.5	49	626	–	18000
h05_c75_l200_06	8	0	8	119	879.2	879.6	885.5	885.5	2	24	0	0.3	28	218	(**9**, **902.0**)	6164
h05_c75_l200_07	8	0	8	118	905.2	906.4	909.0	909.0	3	12	0	1.0	4	297	(**9**, 1,006.6)	18000
h05_c75_l200_08	7	0	7–8	975	802.0	806.6	808.4	808.4	5	32	0	0.6	22	336	(**9**, 894.1)	18000
h05_c75_l200_09	8	0	8	157	884.5	885.8	886.5	886.5	3	9	0	1.7	19	326	(**9**, 1,002.7)	18000
h05_c75_l200_10	7	0	7–9	269	1,072.7	1,081.6	1,090.3	1,090.3	10	18	0	2.1	16	1216	(**10**, 1,139.7)	18000
Average Gap					*4.1*	*3.7*	*3.2*	*3.0*								

all clients were less than the given time limit. That includes most instances with 10 and 15 clients and 4 instances of the set with 30 clients. For instances with 30 and 40 customers, the BCP algorithm starts to show its advantage over the MIP formulations used in the literature, solving all open instances.

The results for Set 3 are displayed in Tables 13–18. The previous best known solution for those instances almost always coincide with the optimal TSP solution for their sets of clients, but the values in those columns are not displayed in bold, as this is not a proof of TSPHS optimality. For example, the optimal tour length found for instance berlin52 with 10 extra hotels is 7,864, which is greater than the TSP length of 7,542 for the same points. In fact, Vansteenwegen et al. (2012) and Castro et al. (2013) also found the solution (8, 7,864), while Castro et al. (2015) and Sousa et al. (2015) found (9, 7,542).

Finally, for Set 4, the BCP algorithm obtained optimal solutions for 5 instances, one of those optimal solutions (berlin52) improve upon the previously best-known solution. For instance kroa100, the best solution found also improves upon the previous best. Note that instance lin105 is missing from the results in Tables 19 and 20. A solution (18, 50,930) was initially reported by Vansteenwegen et al. (2012), but it was removed from subsequent works on the TSPHS. After inspecting that instance, we verified that it is actually infeasible.

By comparing the results in the tables with *ng*-8 to those with *ng*-12, it is possible to see how initially the lower bounds obtained by using *ng*-12 are better. However, after applying SRCs, they tend to be lower than those obtained by using *ng*-8. For the majority of the instances that couldn't be solved, pricing was the biggest challenge. By increasing the *ng*-set size, the performance of the pricing algorithm tends to degrade, which explains why using *ng*-12 yields one less optimal result than *ng*-8. For this reason, we did not attempt to experiment with larger *ng*-sets.

A.2. New instances

Tables 21–47 correspond to the experiments with the newly proposed instances. The columns in those tables are similar to those from previous tables; the only exception is that the number of clients is already encoded in the instance name and that no previous best-known solutions exist.

Results for instances with 50 clients are shown in Tables 21–29. The majority of the instances can be solved to optimality, and using *ng*-12 allowed us to obtain one other optimal result than using *ng*-8 or *ng*-16. Even though for some instances the gap between the TSP lower bound and the number of trips in the optimal solution is significant, the algorithm ruled out infeasible values of q quickly. Another interesting observation is that this gap is larger when the time limit is tighter—a larger limit requires fewer trips and fewer visits to hotels, allowing the TSPHS tour to resemble the TSP tour.

Tables 30 and 38 show the results for instances with 75 clients. Although both 2-Path and 3-SRC were effective in increasing the lower bounds, in general, there was still a large number of branch-and-bound nodes to be solved. With 20 hotels, it became more difficult to obtain optimal solutions for the proposed instances with 75 clients. In these instances, using *ng*-12 presents a significant advantage over other *ng*-set sizes.

Finally, results for instances with 100 clients are displayed in Tables 39 and 47. With 100 clients, it becomes much harder to find solutions without a proper upper bound, although the algorithm was able to find three optimal solutions using *ng*-8, five with *ng*-12 and seven with *ng*-16.

Appendix B. Solutions for TSPHS instances

Pictures for some of the TSPHS instances solved by the TSPHS algorithm are available in this appendix. Hotels are represented as square nodes, while clients are shown as circles.

Table 33
Results for proposed instances with 10 hotels, 75 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c75_l100_01	15	0	15–18	14	938.4	947.2	953.8	956.7	15	23	120	46.9	737	7462	(**19**, 1,026.6)	18000
h10_c75_l100_02	15	0	15–18	0	974.6	996.2	1,006.0	1,007.9	21	21	45	51.5	186	248	(**19**, **1,017.5**)	1322
h10_c75_l100_03	14	0	14–17	0	907.0	920.4	931.9	943.0	6	60	360	44.4	148	90	(**18**, **977.4**)	860
h10_c75_l100_04	15	0	15–17	1	915.4	917.9	933.1	935.2	13	34	60	42.6	738	1110	(**18**, **978.9**)	8661
h10_c75_l100_05	15	0	15–19	1	1,045.9	1,062.7	1,072.3	1,079.8	19	31	123	49.7	372	3739	(**20**, 1,111.0)	18000
h10_c75_l100_06	15	0	15–20	2	1,168.4	1,169.5	1,172.6	1,174.9	8	11	29	47.5	66	3701	(**21**, 1,234.4)	18000
h10_c75_l100_07	15	0	15–18	0	1,018.4	1,020.2	1,023.7	1,039.3	13	35	380	51.7	242	258	(**19**, **1,058.8**)	1409
h10_c75_l100_08	*infeasible*															
h10_c75_l100_09	*infeasible*															
h10_c75_l100_10	*infeasible*															
Average Gap					*4.5*	*3.6*	*2.8*	*2.2*								
h10_c75_l150_01	10	0	10	4	745.2	756.2	764.7	772.2	32	29	480	4.3	14	691	–	18000
h10_c75_l150_02	10	0	10	5	760.6	775.7	781.1	791.1	18	33	930	3.6	7	468	–	18000
h10_c75_l150_03	10	0	10	4	722.2	741.2	748.6	762.5	24	33	571	3.2	74	532	(**11**, 849.8)	18000
h10_c75_l150_04	10	0	10	4	759.0	768.6	771.4	782.0	17	20	550	3.0	64	400	(**11**, **803.1**)	7733
h10_c75_l150_05	10	0	10	4	787.7	798.2	801.5	812.1	36	27	963	13.6	92	44	(**11**, **830.5**)	3206
h10_c75_l150_06	10	0	10–11	700	829.0	831.2	836.9	840.2	16	17	239	6.4	31	2277	(**12**, 932.4)	18000
h10_c75_l150_07	10	0	10	4	765.1	767.9	768.0	774.0	15	9	240	4.6	32	1121	(**11**, 862.2)	18000
h10_c75_l150_08	10	1	10–11	34	829.5	835.9	840.5	854.7	24	20	533	5.2	328	1566	(**12**, **865.4**)	12487
h10_c75_l150_09	10	0	10–12	160	964.6	966.3	972.6	981.6	12	24	182	13.9	33	1569	(**13**, 1,039.3)	18000
h10_c75_l150_10	10	0	10–11	5	912.9	919.7	922.0	937.9	20	26	510	8.9	86	154	(**12**, **946.6**)	4735
Average Gap					*5.1*	*4.1*	*3.6*	*2.4*								
h10_c75_l200_01	8	0	–	–	698.3	713.0	717.3	726.1	33	8	150	0.0	8	349	(**8**, 820.3)	18000
h10_c75_l200_02	8	0	–	–	702.1	718.5	719.7	726.9	33	21	120	0.0	10	772	–	18000
h10_c75_l200_03	7	0	7	16	656.4	679.3	680.5	682.4	21	10	30	0.0	76	40	(**8**, **701.6**)	2007
h10_c75_l200_04	8	0	–	–	722.5	730.5	731.7	736.4	24	15	150	0.0	11	637	–	18000
h10_c75_l200_05	8	0	–	–	717.5	728.1	730.9	736.3	17	17	90	0.0	86	390	(**8**, 816.3)	18000
h10_c75_l200_06	8	0	–	–	741.3	742.4	752.0	758.9	12	17	90	3.0	7	705	–	18000
h10_c75_l200_07	8	0	–	–	712.6	714.8	714.8	723.6	25	13	210	0.1	22	8	(**8**, **728.1**)	706
h10_c75_l200_08	7	0	7	19	713.9	720.4	720.6	734.1	29	18	195	0.0	10	680	–	18000
h10_c75_l200_09	8	0	8	590	779.4	779.8	787.2	791.1	14	12	90	0.9	28	682	(**9**, 904.0)	18000
h10_c75_l200_10	7	0	7	15	732.4	745.7	747.4	757.8	26	37	120	1.7	5	557	(**8**, 808.8)	18000
Average Gap					*6.2*	*4.7*	*4.4*	*3.6*								

Table 34
Results for proposed instances with 10 hotels, 75 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c75_l100_01	15	0	15–18	6	938.5	947.6	953.8	956.7	13	23	77	46.9	1346	7292	(**19**, **974.4**)	12803
h10_c75_l100_02	15	0	15–18	1	974.6	996.2	1,006.0	1,007.9	18	20	45	51.5	178	246	(**19**, **1,017.5**)	742
h10_c75_l100_03	14	0	14–17	0	907.4	920.9	932.7	943.0	7	37	222	44.4	170	104	(**18**, **977.4**)	530
h10_c75_l100_04	15	0	15–17	0	915.4	917.9	933.1	935.2	13	34	60	42.6	700	1070	(**18**, **978.9**)	4684
h10_c75_l100_05	15	0	15–19	0	1,045.9	1,062.7	1,072.3	1,079.8	19	33	135	49.7	606	4014	(**20**, **1,099.2**)	11515
h10_c75_l100_06	15	0	15–20	2	1,168.4	1,169.5	1,172.6	1,174.9	6	11	29	47.5	176	1324	(**21**, **1,185.3**)	1465
h10_c75_l100_07	15	0	15–18	0	1,018.4	1,020.2	1,023.7	1,039.3	13	38	357	51.7	260	300	(**19**, **1,058.8**)	928
h10_c75_l100_08	*infeasible*															
h10_c75_l100_09	*infeasible*															
h10_c75_l100_10	*infeasible*															
Average Gap					*4.5*	*3.6*	*2.8*	*2.2*								
h10_c75_l150_01	10	0	10	7	745.2	756.3	765.0	771.7	37	29	360	4.3	104	64	(**11**, **784.2**)	2277
h10_c75_l150_02	10	0	10	9	761.0	776.1	783.2	790.9	22	26	540	4.0	308	622	(**11**, **821.8**)	11119
h10_c75_l150_03	10	0	10	8	722.4	741.2	748.6	762.5	23	43	450	3.2	130	854	(**11**, 828.1)	18000
h10_c75_l150_04	10	0	10	9	759.0	768.6	771.4	781.8	14	27	390	3.0	48	6	(**11**, **803.1**)	648
h10_c75_l150_05	10	0	10	7	788.0	798.3	801.9	812.1	29	23	670	13.9	88	64	(**11**, **830.5**)	3443
h10_c75_l150_06	10	0	10–11	525	831.2	833.6	836.9	840.2	13	18	228	6.4	320	5034	(**12**, **854.1**)	17581
h10_c75_l150_07	10	0	10	6	765.7	768.4	768.4	774.0	18	7	150	4.6	164	628	(**11**, **791.9**)	7246
h10_c75_l150_08	10	0	10–11	32	830.4	836.6	841.5	854.7	24	20	510	5.2	214	700	(**12**, **865.4**)	4989
h10_c75_l150_09	10	0	10–12	104	965.1	966.4	973.1	981.6	17	23	210	13.9	176	1114	(**13**, **993.6**)	5347
h10_c75_l150_10	10	0	10–11	13	915.0	921.9	924.2	936.7	20	19	210	8.9	64	154	(**12**, **946.6**)	2904
Average Gap					*5.0*	*4.0*	*3.5*	*2.4*								
h10_c75_l200_01	8	0	–	–	698.3	713.1	717.3	722.0	29	6	60	0.0	25	390	(**8**, 820.3)	18000
h10_c75_l200_02	8	0	–	–	702.7	719.3	719.8	725.1	25	13	45	0.0	72	574	(**8**, 748.7)	18000
h10_c75_l200_03	7	0	7	37	657.0	679.4	680.5	682.4	19	11	30	0.0	134	234	(**8**, **701.6**)	6749
h10_c75_l200_04	8	0	–	–	722.5	730.5	731.7	732.8	20	11	30	0.0	19	558	–	18000
h10_c75_l200_05	8	0	–	–	717.5	728.1	730.9	732.8	13	13	30	0.0	22	26	(**8**, **747.5**)	1299
h10_c75_l200_06	8	0	–	–	742.6	743.6	753.3	756.5	14	20	30	3.6	12	730	–	18000
h10_c75_l200_07	8	0	–	–	713.1	715.5	715.9	719.9	19	9	60	0.1	8	2	(**8**, **728.1**)	276
h10_c75_l200_08	7	0	7	50	718.1	723.9	724.8	728.6	20	16	30	0.1	9	625	–	18000
h10_c75_l200_09	8	0	8	1014	781.7	782.7	788.2	789.5	10	11	30	0.9	86	180	(**9**, **794.6**)	8092
h10_c75_l200_10	7	0	7	44	733.3	746.6	748.1	752.0	21	39	45	0.9	94	276	(**8**, **779.8**)	15653
Average Gap					*6.1*	*4.6*	*4.4*	*4.0*								

Table 35
Results for proposed instances with 10 hotels, 75 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c75_l100_01	15	1	15–18	12	938.5	947.6	953.8	956.7	12	24	60	46.9	867	5486	(**19**, 996.7)	18000
h10_c75_l100_02	15	0	15–18	2	974.6	996.2	1,006.0	1,007.9	18	20	45	51.5	200	256	(**19**, **1,017.5**)	1289
h10_c75_l100_03	14	0	14–17	2	907.4	920.9	932.7	943.0	7	40	189	44.4	164	100	(**18**, **977.4**)	1009
h10_c75_l100_04	15	0	15–17	2	915.4	917.9	933.1	935.2	13	33	60	42.6	698	1106	(**18**, **978.9**)	8929
h10_c75_l100_05	15	0	15–19	1	1,045.9	1,062.7	1,072.3	1,079.8	19	25	125	49.7	237	3401	(**20**, 1,153.1)	18000
h10_c75_l100_06	15	0	15–20	4	1,168.4	1,169.5	1,172.6	1,174.9	6	11	29	47.5	164	1324	(**21**, **1,185.3**)	2361
h10_c75_l100_07	15	0	15–18	1	1,018.4	1,020.2	1,023.7	1,039.3	13	32	259	51.7	270	298	(**19**, **1,058.8**)	1544
h10_c75_l100_08	*infeasible*															
h10_c75_l100_09	*infeasible*															
h10_c75_l100_10	*infeasible*															
Average Gap					*4.5*	*3.6*	*2.8*	*2.2*								
h10_c75_l150_01	10	1	10	38	745.6	756.4	764.3	769.4	27	28	150	4.3	118	322	(**11**, **784.2**)	11946
h10_c75_l150_02	10	1	10	46	761.0	776.1	783.2	790.2	18	24	240	3.9	37	639	–	18000
h10_c75_l150_03	10	0	10	37	722.7	742.2	748.6	762.3	23	39	270	3.2	90	558	(**11**, 814.1)	18000
h10_c75_l150_04	10	0	10	45	759.0	768.6	771.4	781.3	14	24	240	3.0	69	942	(**11**, 803.1)	18000
h10_c75_l150_05	10	0	10	35	788.4	798.3	802.1	811.5	28	16	360	15.1	84	64	(**11**, **830.5**)	3344
h10_c75_l150_06	10	0	10–11	1009	831.2	833.6	836.9	840.2	16	18	210	6.4	82	2375	(**12**, 871.3)	18000
h10_c75_l150_07	10	0	10	35	765.7	768.4	768.4	774.0	17	8	162	4.6	180	484	(**11**, **791.9**)	10955
h10_c75_l150_08	10	0	10–11	141	830.4	836.8	841.9	854.5	29	23	257	5.2	286	1850	(**12**, **865.4**)	16632
h10_c75_l150_09	10	0	10–12	344	965.1	966.4	973.1	981.6	15	22	165	13.9	35	1599	(**13**, 1,039.3)	18000
h10_c75_l150_10	10	0	10–11	68	915.0	921.9	924.7	935.9	15	24	150	8.9	30	817	(**12**, 970.5)	18000
Average Gap					*5.0*	*4.0*	*3.5*	*2.5*								
h10_c75_l200_01	8	1	–	–	698.3	713.2	717.3	717.3	25	5	0	0.0	6	1415	–	18000
h10_c75_l200_02	8	0	–	–	703.1	719.3	719.8	719.8	16	5	0	0.0	5	669	–	18000
h10_c75_l200_03	7	0	7	193	657.0	679.4	680.5	680.5	17	10	0	0.0	92	296	(**8**, **701.6**)	10840
h10_c75_l200_04	8	0	–	–	723.6	730.5	731.8	731.8	19	13	0	0.0	29	434	(**8**, 785.1)	18000
h10_c75_l200_05	8	0	–	–	717.6	728.2	730.9	730.9	14	15	0	0.0	30	554	(**8**, 823.0)	18000
h10_c75_l200_06	8	0	–	–	742.6	743.8	753.4	753.4	10	18	0	3.5	7	693	–	18000
h10_c75_l200_07	8	1	–	–	714.6	716.7	717.1	717.1	10	3	0	0.2	7	426	–	18000
h10_c75_l200_08	7	0	7	340	718.1	724.0	724.8	724.8	18	18	0	0.1	7	469	–	18000
h10_c75_l200_09	8	0	8	2776	781.8	783.1	788.2	788.2	10	11	0	0.9	36	640	–	18000
h10_c75_l200_10	7	0	7	225	733.7	747.1	749.5	749.5	26	19	0	1.0	5	830	–	18000
Average Gap					*6.1*	*4.6*	*4.3*	*4.3*								

Table 36
Results for proposed instances with 20 hotels, 75 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c75_l100_01	15	0	15–16	1	819.0	849.3	853.5	862.1	51	14	150	36.9	30	0	(**17**, **872.0**)	199
h20_c75_l100_02	15	0	15–17	1	878.5	898.3	905.7	908.5	61	15	124	42.7	1067	2211	(**18**, 956.8)	18000
h20_c75_l100_03	14	0	14–16	1	796.3	819.5	827.4	831.8	38	26	107	35.8	1555	1318	(**17**, 868.6)	18000
h20_c75_l100_04	15	0	15–16	1	804.0	820.4	840.7	848.6	59	27	339	33.9	182	4148	(**17**, 883.3)	18000
h20_c75_l100_05	15	0	15–16	27	719.0	756.4	761.5	764.4	54	27	88	34.4	14	2254	–	18000
h20_c75_l100_06	15	0	15–17	1	870.4	885.9	889.9	896.0	38	18	287	34.3	768	1844	(**18**, 946.1)	18000
h20_c75_l100_07	15	2	15–16	1	806.2	819.7	823.0	828.8	36	39	151	41.0	2402	1180	(**17**, 861.9)	18000
h20_c75_l100_08	14	0	14–16	1	803.3	830.4	839.5	845.7	66	23	389	40.0	664	1056	–	18000
h20_c75_l100_09	15	0	15–16	1	777.3	790.4	795.6	801.1	41	26	164	41.9	942	650	(**17**, **816.1**)	5846
h20_c75_l100_10	14	0	14–16	1	786.2	821.7	830.6	841.1	58	18	251	31.5	630	1202	–	18000
Average Gap					*6.7*	*4.6*	*3.8*	*3.1*								
h20_c75_l150_01	10	0	10	8	699.8	734.9	738.7	744.6	66	17	300	1.7	12	761	(**11**, 822.2)	18000
h20_c75_l150_02	10	0	10	6	715.6	741.6	747.9	756.9	75	26	450	2.2	10	707	–	18000
h20_c75_l150_03	10	0	10	41	654.3	686.0	692.2	701.2	49	37	210	1.9	505	1014	(**11**, 726.0)	18000
h20_c75_l150_04	10	0	10	6	709.2	734.7	740.9	743.6	57	13	488	1.6	30	806	(**11**, 832.9)	18000
h20_c75_l150_05	10	0	10	31	668.6	696.0	701.1	704.9	81	29	300	0.3	83	2485	(**11**, 739.6)	18000
h20_c75_l150_06	10	1	10	6	736.5	748.8	758.8	764.3	54	15	402	2.6	976	574	(**11**, 787.0)	18000
h20_c75_l150_07	10	0	10	8	705.2	719.5	720.9	725.9	74	13	596	1.3	25	1089	(**11**, 819.3)	18000
h20_c75_l150_08	10	0	10	793	658.7	685.8	688.9	695.4	50	17	138	1.7	138	66	(**11**, **697.4**)	3343
h20_c75_l150_09	10	0	10	27	677.9	695.1	701.6	708.4	47	38	371	2.0	228	76	(**11**, **727.4**)	7195
h20_c75_l150_10	10	0	10	97	658.2	693.5	698.5	702.4	74	20	240	0.9	15	1104	(**11**, 802.3)	18000
Average Gap					*10.6*	*7.4*	*6.7*	*5.9*								
h20_c75_l200_01	8	1	–	–	665.1	702.1	707.0	709.2	66	7	120	0.0	385	116	(**8**, 725.0)	18000
h20_c75_l200_02	8	0	–	–	677.0	702.5	704.2	710.4	86	15	90	0.0	299	20	(**8**, 726.9)	18000
h20_c75_l200_03	7	0	7	23	628.3	652.6	655.2	655.7	51	4	60	0.0	72	18	(**8**, **662.1**)	1597
h20_c75_l200_04	8	0	–	–	701.7	716.9	719.6	721.8	61	11	120	0.0	108	8	(**8**, **731.1**)	3982
h20_c75_l200_05	8	0	–	–	655.5	681.3	681.5	685.3	37	6	90	0.0	13	381	(**8**, 802.9)	18000
h20_c75_l200_06	8	0	–	–	694.3	710.2	713.1	714.2	43	24	90	0.0	480	48	(**8**, **732.2**)	16398
h20_c75_l200_07	8	0	–	–	674.1	688.8	688.8	692.2	68	11	180	0.0	223	5957	–	18000
h20_c75_l200_08	7	1	7	22	625.0	651.3	653.2	658.2	60	17	120	0.0	110	46	(**8**, **670.6**)	4220
h20_c75_l200_09	8	0	–	–	649.2	664.0	669.1	673.2	47	21	90	0.0	6	311	(**8**, 798.7)	18000
h20_c75_l200_10	7	0	7	25	615.5	657.9	660.6	663.6	60	14	105	0.0	23	516	(**8**, 685.6)	18000
Average Gap					*7.7*	*4.3*	*4.0*	*3.5*								

Table 37
Results for proposed instances with 20 hotels, 75 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c75_l100_01	15	0	15–16	1	819.0	850.2	853.6	862.1	48	11	89	36.3	44	0	(**17**, **872.0**)	181
h20_c75_l100_02	15	2	15–17	1	879.6	898.3	905.7	908.5	62	17	103	42.7	1945	4291	(**18**, 956.8)	18000
h20_c75_l100_03	14	0	14–16	1	796.4	819.5	827.9	831.8	36	28	61	35.8	1552	1324	(**17**, **868.6**)	9383
h20_c75_l100_04	15	0	15–16	1	804.0	820.4	840.7	848.6	61	29	324	33.9	983	738	–	18000
h20_c75_l100_05	15	0	15–16	20	719.0	756.4	761.5	764.4	55	27	83	34.4	14	4038	–	18000
h20_c75_l100_06	15	0	15–17	1	870.7	885.9	889.9	896.0	35	16	194	34.3	2576	2772	(**18**, **910.0**)	16594
h20_c75_l100_07	15	0	15–16	1	806.7	820.0	823.5	828.8	31	30	141	41.0	3674	1414	(**17**, **861.9**)	13771
h20_c75_l100_08	14	0	14–16	1	803.3	830.4	839.5	845.7	69	24	406	40.0	1520	1332	–	18000
h20_c75_l100_09	15	0	15–16	1	777.5	790.4	795.9	801.1	39	24	107	41.9	1238	936	(**17**, **816.1**)	5364
h20_c75_l100_10	14	0	14–16	1	786.9	822.1	831.5	841.1	46	16	227	31.5	1130	2337	–	18000
Average Gap					*6.7*	*4.6*	*3.7*	*3.1*								
h20_c75_l150_01	10	0	10	14	699.8	734.9	738.8	743.2	66	15	120	1.7	28	1163	(**11**, 853.3)	18000
h20_c75_l150_02	10	0	10	12	717.0	742.8	749.0	756.4	64	24	210	2.2	10	705	–	18000
h20_c75_l150_03	10	0	10	37	655.0	687.7	693.4	700.9	43	45	180	1.9	956	506	(**11**, **719.8**)	9632
h20_c75_l150_04	10	0	10	11	709.3	734.8	741.0	742.9	61	15	150	1.6	18	620	–	18000
h20_c75_l150_05	10	0	10	29	668.6	696.0	701.2	704.9	69	26	240	0.3	311	1720	(**11**, 739.3)	18000
h20_c75_l150_06	10	0	10	14	737.4	748.8	758.8	764.2	45	15	210	2.6	1166	1276	(**11**, 787.0)	18000
h20_c75_l150_07	10	0	10	14	705.2	719.7	721.2	725.8	68	11	270	1.3	31	1681	(**11**, 819.2)	18000
h20_c75_l150_08	10	0	10	739	659.5	685.8	688.6	695.4	51	16	90	1.7	164	336	(**11**, **697.4**)	3689
h20_c75_l150_09	10	0	10	21	678.1	695.1	701.6	708.3	56	27	210	2.0	204	42	(**11**, **727.4**)	2345
h20_c75_l150_10	10	0	10	108	658.2	693.8	698.5	702.3	67	20	150	0.9	21	1985	(**11**, 777.4)	18000
Average Gap					*10.6*	*7.3*	*6.6*	*6.0*								
h20_c75_l200_01	8	0	–	–	665.1	702.1	707.0	707.4	66	6	30	0.0	33	621	(**8**, 737.8)	18000
h20_c75_l200_02	8	0	–	–	679.0	702.9	706.0	707.2	71	16	30	0.0	11	597	(**8**, 757.4)	18000
h20_c75_l200_03	7	0	7	43	628.3	652.6	655.2	655.5	49	2	30	0.0	106	28	(**8**, **662.1**)	1488
h20_c75_l200_04	8	0	–	–	701.8	716.9	719.6	721.1	66	11	60	0.0	17	351	(**8**, 783.6)	18000
h20_c75_l200_05	8	0	–	–	655.5	681.4	681.6	683.6	30	4	30	0.0	34	138	(**8**, **690.2**)	4756
h20_c75_l200_06	8	0	–	–	694.3	710.5	713.1	713.6	43	20	30	0.0	7	739	–	18000
h20_c75_l200_07	8	0	–	–	674.1	689.0	689.0	691.0	52	5	75	0.0	7	550	–	18000
h20_c75_l200_08	7	0	7	55	625.7	651.6	653.5	655.8	52	16	30	0.0	104	92	(**8**, **670.6**)	4660
h20_c75_l200_09	8	0	–	–	650.9	666.6	668.6	671.7	53	13	30	0.0	57	270	(**8**, 729.3)	18000
h20_c75_l200_10	7	0	7	75	615.7	658.4	661.1	661.5	60	13	30	0.0	7	513	(**8**, 707.5)	18000
Average Gap					*7.7*	*4.2*	*3.9*	*3.7*								

Table 38
Results for proposed instances with 20 hotels, 75 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c75_l100_01	15	1	15–16	4	819.0	850.2	853.6	862.1	47	11	86	36.3	44	0	(**17**, **872.0**)	342
h20_c75_l100_02	15	1	15–17	2	879.6	898.3	905.7	908.5	62	17	113	42.7	1056	2138	(**18**, 956.8)	18000
h20_c75_l100_03	14	1	14–16	5	796.8	819.5	827.9	831.8	40	29	83	35.8	1305	1274	(**17**, 868.6)	18000
h20_c75_l100_04	15	0	15–16	4	804.0	820.4	840.7	848.6	54	30	273	33.9	866	665	–	18000
h20_c75_l100_05	15	0	15–16	32	719.0	756.5	761.5	764.4	57	26	96	34.4	14	2226	–	18000
h20_c75_l100_06	15	0	15–17	6	870.7	885.9	889.9	896.0	33	16	184	34.3	589	1702	(**18**, 968.2)	18000
h20_c75_l100_07	15	0	15–16	3	806.7	820.0	823.5	828.8	34	30	149	41.0	1923	1140	(**17**, 871.7)	18000
h20_c75_l100_08	14	0	14–16	9	803.3	830.4	839.5	845.7	73	25	374	40.0	535	1040	–	18000
h20_c75_l100_09	15	0	15–16	4	777.5	790.4	795.9	801.1	41	23	101	41.9	1402	1544	(**17**, 817.2)	18000
h20_c75_l100_10	14	0	14–16	6	786.9	822.1	831.5	841.1	49	16	236	31.5	579	1163	–	18000
Average Gap					*6.7*	*4.6*	*3.7*	*3.1*								
h20_c75_l150_01	10	0	10	47	699.8	734.9	738.8	742.1	63	16	60	1.7	10	477	–	18000
h20_c75_l150_02	10	1	10	43	717.0	742.8	748.9	754.3	59	24	90	2.2	12	523	–	18000
h20_c75_l150_03	10	0	10	96	655.0	687.7	693.4	701.3	49	45	180	1.9	10	1067	(**11**, 794.2)	18000
h20_c75_l150_04	10	0	10	53	709.3	734.8	741.0	742.4	56	15	90	1.6	16	626	(**11**, 826.5)	18000
h20_c75_l150_05	10	0	10	68	668.8	696.0	701.2	704.7	75	21	180	0.3	1033	529	(**11**, 739.6)	18000
h20_c75_l150_06	10	0	10	50	737.5	748.8	758.8	763.7	45	14	90	2.6	365	909	(**11**, 796.0)	18000
h20_c75_l150_07	10	0	10	48	705.3	719.8	721.2	725.4	60	11	120	1.3	18	584	(**11**, 819.2)	18000
h20_c75_l150_08	10	0	10	1124	659.5	685.8	688.6	695.4	50	14	90	1.7	27	629	(**11**, 839.9)	18000
h20_c75_l150_09	10	0	10	70	678.1	695.1	701.6	708.3	53	26	210	2.0	15	582	–	18000
h20_c75_l150_10	10	0	10	238	658.3	693.9	698.6	701.5	56	15	90	0.9	9	768	–	18000
Average Gap					*10.6*	*7.3*	*6.6*	*6.0*								
h20_c75_l200_01	8	1	–	–	665.9	702.1	707.0	707.0	68	5	0	0.0	9	371	(**8**, 757.3)	18000
h20_c75_l200_02	8	0	–	–	679.0	702.9	704.8	704.8	74	13	0	0.0	7	535	(**8**, 802.6)	18000
h20_c75_l200_03	7	0	7	192	628.3	652.9	655.5	655.5	41	2	0	0.0	66	34	(**8**, **662.1**)	2224
h20_c75_l200_04	8	0	–	–	701.8	716.9	719.6	719.6	48	7	0	0.0	7	586	(**8**, 777.0)	18000
h20_c75_l200_05	8	0	–	–	655.5	681.4	681.6	681.6	30	4	0	0.0	10	16	(**8**, **690.2**)	1493
h20_c75_l200_06	8	0	–	–	694.3	710.5	713.4	713.4	42	20	0	0.0	9	1031	(**8**, 769.8)	18000
h20_c75_l200_07	8	0	–	–	674.1	689.0	689.0	689.0	42	2	0	0.0	11	327	(**8**, 791.6)	18000
h20_c75_l200_08	7	0	7	316	625.7	651.6	655.1	655.1	47	16	0	0.0	258	118	(**8**, **670.6**)	17992
h20_c75_l200_09	8	0	–	–	650.9	666.6	668.6	668.6	47	7	0	0.0	8	354	(**8**, 809.4)	18000
h20_c75_l200_10	7	0	7	360	615.7	658.4	661.1	661.1	51	12	0	0.0	7	1212	–	18000
Average Gap					*7.7*	*4.2*	*3.9*	*3.9*								

Table 39
Results for proposed instances with 5 hotels, 100 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c100_l150_01	12	0	12–15	19	1,282.0	1,282.0	1,290.9	1,306.8	1	40	390	21.2	38	738	(**16**, 1,313.4)	18000
h05_c100_l150_02	13	0	13–16	390	1,323.4	1,324.0	1,327.6	1,343.2	8	31	456	15.7	14	354	(**17**, 1,443.6)	18000
h05_c100_l150_03	12	0	12–20	22	2,011.0	2,012.9	2,041.9	2,058.8	6	31	720	28.8	131	795	–	18000
h05_c100_l150_04	12	0	12–16	9	1,441.5	1,441.5	1,452.4	1,491.4	0	36	990	48.6	60	398	–	18000
h05_c100_l150_05	12	0	12–18	60	1,605.5	1,616.2	1,625.8	1,637.5	4	51	630	20.7	10	886	(**19**, 1,653.0)	18000
h05_c100_l150_06	12	2	12–17	800	1,433.7	1,433.7	1,440.0	1,460.0	2	40	861	18.4	72	1507	(**18**, 1,480.9)	18000
h05_c100_l150_07	12	0	12–16	494	1,312.1	1,312.1	1,323.6	1,333.9	6	52	540	17.5	20	560	(**17**, 1,367.0)	18000
h05_c100_l150_08	12	0	12–15	61	1,167.0	1,169.7	1,185.1	1,196.3	1	71	150	14.0	2	0	(**16**, **1,196.8**)	110
h05_c100_l150_09	12	0	12–18	18	1,702.0	1,703.1	1,717.7	1,736.5	6	43	480	24.8	83	754	–	18000
h05_c100_l150_10	12	0	12–18	19	1,738.9	1,743.8	1,746.1	1,765.6	8	23	810	31.4	38	825	–	18000
Average Gap					*3.2*	*3.1*	*2.3*	*1.3*								
h05_c100_l200_01	9	0	9–10	60	984.2	984.5	988.6	994.8	3	14	90	0.5	4	1185	–	18000
h05_c100_l200_02	10	0	10	36	1,038.2	1,040.9	1,042.0	1,047.6	11	13	60	1.1	43	226	(**11**, 1,090.3)	18000
h05_c100_l200_03	9	0	9–11	51	1,238.3	1,243.9	1,244.4	1,246.7	5	11	90	1.9	38	290	(**12**, **1,266.1**)	15490
h05_c100_l200_04	9	0	9–10	47	1,073.1	1,073.1	1,074.7	1,079.2	0	8	60	2.4	11	242	(**11**, 1,137.4)	18000
h05_c100_l200_05	9	0	9–10	47	1,141.3	1,143.6	1,145.0	1,153.1	3	13	90	44.1	75	976	–	18000
h05_c100_l200_06	9	1	9–10	51	1,047.2	1,047.2	1,050.6	1,056.9	0	35	90	0.4	18	1007	–	18000
h05_c100_l200_07	9	0	9–10	58	1,029.4	1,029.4	1,034.3	1,038.0	3	13	60	0.4	5	986	–	18000
h05_c100_l200_08	9	0	9–10	121	949.2	952.1	957.7	961.7	6	26	30	0.1	13	431	(**11**, 1,154.1)	18000
h05_c100_l200_09	9	0	9–11	79	1,145.3	1,147.1	1,163.5	1,169.7	7	33	60	2.9	13	500	(**12**, 1,243.4)	18000
h05_c100_l200_10	9	0	9–11	106	1,162.5	1,166.9	1,170.8	1,180.3	7	42	60	2.3	16	626	(**12**, 1,222.5)	18000
Average Gap					*8.0*	*7.8*	*7.4*	*6.9*								

Table 40
Results for proposed instances with 5 hotels, 100 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c100_l150_01	12	0	12–15	42	1,284.1	1,284.1	1,291.9	1,306.0	1	37	210	17.6	48	520	(**16**, **1,309.3**)	6811
h05_c100_l150_02	13	0	13–16	359	1,324.3	1,324.9	1,331.6	1,342.9	7	27	180	15.7	34	810	(**17**, 1,387.4)	18000
h05_c100_l150_03	12	0	12–20	50	2,011.8	2,013.5	2,043.4	2,058.7	6	35	420	29.0	74	608	(**21**, 2,095.7)	18000
h05_c100_l150_04	12	0	12–16	23	1,443.2	1,443.2	1,455.2	1,489.6	0	28	570	52.8	94	477	–	18000
h05_c100_l150_05	12	0	12–18	67	1,606.1	1,616.3	1,626.1	1,637.0	4	44	450	20.7	165	1145	(**19**, 1,650.7)	18000
h05_c100_l150_06	12	0	12–17	398	1,433.8	1,433.8	1,441.1	1,459.8	1	41	420	18.4	44	2115	(**18**, 1,485.9)	18000
h05_c100_l150_07	12	0	12–16	672	1,312.5	1,312.5	1,323.7	1,333.1	5	53	330	17.5	72	645	(**17**, 1,455.7)	18000
h05_c100_l150_08	12	0	12–15	89	1,168.5	1,171.1	1,191.2	1,196.3	1	49	90	14.0	2	0	(**16**, **1,196.8**)	133
h05_c100_l150_09	12	0	12–18	54	1,705.2	1,705.6	1,718.3	1,736.0	5	34	270	24.8	34	490	(**19**, 1,797.1)	18000
h05_c100_l150_10	12	0	12–18	47	1,743.9	1,748.2	1,749.9	1,764.1	9	13	240	25.7	3	1184	–	18000
Average Gap					*3.1*	*3.0*	*2.2*	*1.3*								
h05_c100_l200_01	9	0	9–10	150	987.9	988.0	991.4	992.4	2	12	30	0.5	4	574	(**11**, 1,109.0)	18000
h05_c100_l200_02	10	0	10	106	1,039.8	1,043.0	1,043.7	1,047.5	13	11	30	1.2	30	210	(**11**, 1,108.0)	18000
h05_c100_l200_03	9	0	9–11	117	1,238.9	1,244.3	1,244.7	1,246.0	4	15	30	1.9	48	268	(**12**, 1,266.1)	18000
h05_c100_l200_04	9	0	9–10	133	1,076.2	1,076.2	1,078.4	1,080.1	0	12	30	2.8	28	1011	–	18000
h05_c100_l200_05	9	0	9–10	113	1,142.1	1,144.7	1,146.6	1,148.0	3	17	30	35.7	83	940	–	18000
h05_c100_l200_06	9	0	9–10	129	1,048.5	1,048.5	1,052.4	1,053.8	0	19	30	0.4	16	862	–	18000
h05_c100_l200_07	9	0	9–10	117	1,031.9	1,032.0	1,037.3	1,037.3	2	15	0	0.4	3	823	–	18000
h05_c100_l200_08	9	0	9–10	215	949.4	952.3	958.2	958.2	4	21	0	0.1	6	802	–	18000
h05_c100_l200_09	9	0	9–11	218	1,145.8	1,147.2	1,164.0	1,164.3	5	26	30	2.9	4	12237	(**12**, 1,243.3)	18000
h05_c100_l200_10	9	0	9–11	222	1,165.0	1,169.8	1,172.8	1,177.0	7	19	30	2.3	131	202	(**12**, 1,299.0)	18000
Average Gap					*7.8*	*7.6*	*7.3*	*7.1*								

Table 41
Results for proposed instances with 5 hotels, 100 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h05_c100_l150_01	12	0	12–15	119	1,284.4	1,284.4	1,292.1	1,300.4	1	30	90	17.6	20	514	(**16**, **1,309.3**)	6200
h05_c100_l150_02	13	0	13–16	446	1,324.5	1,325.0	1,331.6	1,341.5	6	27	90	15.7	18	6	(**17**, **1,348.2**)	693
h05_c100_l150_03	12	0	12–20	122	2,013.7	2,015.5	2,044.1	2,056.9	5	24	300	29.4	12	870	–	18000
h05_c100_l150_04	12	0	12–16	50	1,447.2	1,447.2	1,459.4	1,483.0	0	27	240	35.8	158	714	–	18000
h05_c100_l150_05	12	0	12–18	152	1,606.3	1,616.4	1,626.2	1,635.8	4	45	270	20.7	4	3864	(**19**, 1,705.2)	18000
h05_c100_l150_06	12	0	12–17	819	1,434.2	1,434.2	1,441.3	1,457.8	1	30	150	18.4	35	1312	(**18**, 1,579.7)	18000
h05_c100_l150_07	12	0	12–16	1027	1,313.0	1,313.0	1,323.7	1,331.3	1	50	150	17.5	60	819	–	18000
h05_c100_l150_08	12	0	12–15	186	1,169.5	1,171.6	1,191.2	1,196.3	2	50	90	14.0	2	0	(**16**, **1,196.8**)	265
h05_c100_l150_09	12	0	12–18	162	1,705.7	1,705.8	1,718.3	1,731.8	4	30	120	24.8	5	915	–	18000
h05_c100_l150_10	12	0	12–18	119	1,745.5	1,749.6	1,752.9	1,763.0	8	12	180	26.9	3	1193	–	18000
Average Gap					*3.1*	*3.0*	*2.1*	*1.5*								
h05_c100_l200_01	9	0	9–10	534	987.9	988.2	991.6	991.6	2	14	0	0.5	4	522	(**11**, 1,159.8)	18000
h05_c100_l200_02	10	0	10	408	1,040.0	1,043.2	1,043.9	1,043.9	10	9	0	1.2	41	739	–	18000
h05_c100_l200_03	9	0	9–11	428	1,238.9	1,244.3	1,244.7	1,244.7	4	11	0	1.9	50	166	(**12**, **1,266.1**)	9735
h05_c100_l200_04	9	0	9–10	560	1,076.2	1,076.2	1,078.6	1,078.6	0	8	0	2.9	4	346	(**11**, 1,142.9)	18000
h05_c100_l200_05	9	0	9–10	411	1,144.5	1,147.1	1,149.1	1,149.1	1	14	0	37.9	6	946	–	18000
h05_c100_l200_06	9	0	9–10	493	1,048.5	1,048.5	1,052.6	1,052.6	0	20	0	0.4	4	574	(**11**, 1,154.5)	18000
h05_c100_l200_07	9	0	9–10	507	1,032.7	1,032.7	1,038.2	1,038.2	1	25	0	0.4	3	759	–	18000
h05_c100_l200_08	9	0	9–10	893	949.4	952.3	958.2	958.2	4	17	0	0.1	8	1495	–	18000
h05_c100_l200_09	9	0	9–11	1049	1,146.0	1,147.3	1,164.0	1,164.0	5	18	0	2.9	4	944	–	18000
h05_c100_l200_10	9	0	9–11	1404	1,165.4	1,170.6	1,173.6	1,173.6	6	17	0	2.3	10	1102	–	18000
Average Gap					*7.8*	*7.6*	*7.2*	*7.2*								

Table 42
Results for proposed instances with 10 hotels, 100 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c100_l100_01	18	0	18–24	2	1,282.9	1,293.0	1,307.4	1,317.8	18	35	220	49.9	39	804	–	18000
h10_c100_l100_02	*infeasible*															
h10_c100_l100_03	*infeasible*															
h10_c100_l100_04	*infeasible*															
h10_c100_l100_05	*infeasible*															
h10_c100_l100_06	18	0	18–24	2	1,335.5	1,342.4	1,352.5	1,367.4	11	48	456	53.1	72	787	–	18000
h10_c100_l100_07	18	0	18–29	19	1,759.2	1,765.1	1,787.9	1,796.4	26	40	345	62.1	50	1539	–	18000
h10_c100_l100_08	18	0	18–24	10	1,246.8	1,253.9	1,287.5	1,290.4	17	30	87	49.6	8	709	–	18000
h10_c100_l100_09	*infeasible*															
h10_c100_l100_10	*infeasible*															
Average Gap					*10.6*	*10.1*	*7.7*	*7.5*								
h10_c100_l150_01	12	0	12–13	18	943.7	949.0	954.5	961.9	14	39	330	6.0	65	539	(**14**, 993.0)	18000
h10_c100_l150_02	13	2	13–14	19	1,057.8	1,060.4	1,060.7	1,068.2	9	24	240	9.0	8	578	–	18000
h10_c100_l150_03	12	0	12–14	18	1,131.1	1,133.4	1,135.6	1,149.9	8	17	420	12.4	18	401	(**15**, 1,221.4)	18000
h10_c100_l150_04	12	2	12–15	29	1,186.2	1,195.3	1,197.5	1,225.5	13	27	450	9.4	6	739	–	18000
h10_c100_l150_05	12	1	12–15	24	1,184.2	1,205.6	1,210.3	1,221.0	25	36	330	9.2	14	352	(**16**, 1,283.2)	18000
h10_c100_l150_06	12	0	12–13	22	946.6	962.1	965.1	985.9	14	25	300	6.3	18	233	(**14**, 1,054.3)	18000
h10_c100_l150_07	12	1	12–14	23	1,055.7	1,066.7	1,074.7	1,082.1	21	24	300	10.8	30	1525	(**15**, 1,096.7)	18000
h10_c100_l150_08	12	0	12–13	20	926.6	929.9	939.0	960.1	9	46	240	5.1	8	734	–	18000
h10_c100_l150_09	12	0	12–14	25	1,104.1	1,120.8	1,129.5	1,139.3	16	33	270	12.1	8	642	–	18000
h10_c100_l150_10	12	0	12–14	568	1,026.8	1,032.0	1,035.4	1,049.7	12	27	210	8.6	27	446	(**15**, 1,095.1)	18000
Average Gap					*6.1*	*5.4*	*5.0*	*3.7*								
h10_c100_l200_01	9	0	9	54	843.3	854.3	856.4	862.3	19	16	75	0.0	7	844	–	18000
h10_c100_l200_02	10	3	–	–	900.9	906.1	907.1	914.8	12	19	90	5.8	184	2056	–	18000
h10_c100_l200_03	9	3	9–10	844	948.2	953.3	963.0	965.8	11	14	30	0.1	171	2443	–	18000
h10_c100_l200_04	9	3	9–10	193	953.9	962.9	963.8	967.8	17	15	60	0.1	6	7201	(**11**, 1,075.5)	18000
h10_c100_l200_05	9	0	9–10	10015	931.3	944.3	944.4	948.8	20	4	60	0.4	240	291	(**11**, 1,040.2)	18000
h10_c100_l200_06	9	1	9	52	841.8	859.7	864.7	867.9	12	35	45	0.0	28	245	(**10**, 908.6)	18000
h10_c100_l200_07	9	0	9	49	904.1	910.6	914.8	927.8	19	34	75	17.3	87	601	–	18000
h10_c100_l200_08	9	0	9	50	852.2	858.0	858.1	859.0	14	2	30	0.0	8	879	–	18000
h10_c100_l200_09	9	0	9	46	900.1	915.1	921.1	928.9	15	20	60	16.4	285	675	–	18000
h10_c100_l200_10	9	0	9	42	876.6	879.5	879.7	885.1	12	10	60	1.4	7	208	(**10**, 926.3)	18000
Average Gap					*6.7*	*5.8*	*5.5*	*5.0*								

Table 43
Results for proposed instances with 10 hotels, 100 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c100_l100_01	18	0	18–24	2	1,282.9	1,293.4	1,307.7	1,317.8	16	38	210	49.9	97	1469	–	18000
h10_c100_l100_02	*infeasible*															
h10_c100_l100_03	*infeasible*															
h10_c100_l100_04	*infeasible*															
h10_c100_l100_05	*infeasible*															
h10_c100_l100_06	18	0	18–24	2	1,337.0	1,344.0	1,353.5	1,367.4	13	38	391	53.1	98	1565	–	18000
h10_c100_l100_07	18	0	18–29	14	1,759.2	1,765.2	1,789.1	1,796.4	24	41	223	62.1	30	2699	–	18000
h10_c100_l100_08	18	0	18–24	8	1,246.8	1,253.9	1,287.5	1,290.4	19	31	82	49.6	32	2121	(**25**, 1,394.8)	18000
h10_c100_l100_09	*infeasible*															
h10_c100_l100_10	*infeasible*															
Average Gap					*10.6*	*10.1*	*7.7*	*7.5*								
h10_c100_l150_01	12	0	12–13	32	944.0	949.7	955.0	961.6	13	25	120	6.0	102	739	(**14**, 993.0)	18000
h10_c100_l150_02	13	0	13–14	38	1,057.8	1,061.0	1,061.2	1,063.9	9	20	120	9.0	349	232	(**15**, 1,090.6)	18000
h10_c100_l150_03	12	0	12–14	31	1,132.9	1,136.1	1,137.5	1,146.7	6	18	180	15.0	44	530	(**15**, 1,202.3)	18000
h10_c100_l150_04	12	0	12–15	30	1,191.1	1,201.0	1,202.3	1,222.4	13	27	210	9.4	32	635	(**16**, 1,294.5)	18000
h10_c100_l150_05	12	0	12–15	44	1,184.8	1,207.0	1,212.5	1,221.5	26	36	240	9.2	250	268	(**16**, **1,237.5**)	7532
h10_c100_l150_06	12	0	12–13	39	946.7	962.2	965.3	981.5	13	20	120	6.3	34	612	(**14**, 1,028.6)	18000
h10_c100_l150_07	12	0	12–14	47	1,056.3	1,068.1	1,074.9	1,081.4	20	27	180	10.8	7	655	–	18000
h10_c100_l150_08	12	0	12–13	36	928.4	931.3	939.9	956.0	8	44	90	5.1	58	386	(**14**, 1,036.3)	18000
h10_c100_l150_09	12	0	12–14	50	1,105.6	1,124.0	1,131.7	1,138.0	12	37	150	12.3	12	608	–	18000
h10_c100_l150_10	12	0	12–14	410	1,026.8	1,032.0	1,035.4	1,048.6	12	23	180	8.6	50	983	(**15**, 1,171.6)	18000
Average Gap					*6.0*	*5.2*	*4.9*	*3.9*								
h10_c100_l200_01	9	0	9	112	845.7	856.9	858.9	858.9	15	7	30	0.0	7	904	–	18000
h10_c100_l200_02	10	0	–	–	902.3	906.9	910.3	911.1	13	38	30	6.2	38	183	(**10**, 950.4)	18000
h10_c100_l200_03	9	0	9–10	367	948.4	953.5	965.2	966.1	12	11	30	0.1	16	841	–	18000
h10_c100_l200_04	9	0	9–10	178	954.3	963.8	964.5	965.8	14	6	30	0.1	7	1122	(**11**, 1,110.8)	18000
h10_c100_l200_05	9	0	9–10	13551	931.8	944.9	945.0	946.2	17	3	30	0.4	291	249	(**11**, 1,094.1)	18000
h10_c100_l200_06	9	0	9	139	844.1	859.9	865.4	865.4	10	28	0	0.0	7	816	–	18000
h10_c100_l200_07	9	0	9	102	909.5	914.8	918.7	918.7	12	23	30	9.9	96	603	–	18000
h10_c100_l200_08	9	0	9	143	853.2	859.4	859.4	859.4	13	1	0	0.0	8	267	–	18000
h10_c100_l200_09	9	0	9	137	900.6	916.6	922.7	922.7	16	8	0	20.0	101	1319	–	18000
h10_c100_l200_10	9	0	9	95	876.9	879.5	879.7	879.7	8	5	0	1.4	4	2124	–	18000
Average Gap					*6.6*	*5.7*	*5.4*	*5.3*								

Table 44
Results for proposed instances with 10 hotels, 100 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h10_c100_l100_01	18	0	18–24	10	1,282.9	1,293.4	1,307.7	1,317.8	15	32	160	49.9	63	877	–	18000
h10_c100_l100_02	*infeasible*															
h10_c100_l100_03	*infeasible*															
h10_c100_l100_04	*infeasible*															
h10_c100_l100_05	*infeasible*															
h10_c100_l100_06	18	0	18–24	13	1,337.0	1,344.0	1,353.5	1,367.4	11	38	343	53.1	80	877	–	18000
h10_c100_l100_07	18	0	18–29	33	1,759.2	1,765.2	1,789.1	1,796.4	25	44	192	62.1	57	1364	–	18000
h10_c100_l100_08	18	0	18–24	25	1,246.8	1,253.9	1,287.6	1,290.4	17	29	114	49.6	26	1089	(**25**, 1,394.8)	18000
h10_c100_l100_09	*infeasible*															
h10_c100_l100_10	*infeasible*															
Average Gap					*10.6*	*10.1*	*7.7*	*7.5*								
h10_c100_l150_01	12	0	12–13	159	944.0	949.7	955.0	959.5	13	25	30	6.0	19	741	–	18000
h10_c100_l150_02	13	0	13–14	169	1,057.9	1,061.1	1,061.3	1,062.2	9	11	30	9.0	8	853	–	18000
h10_c100_l150_03	12	0	12–14	127	1,133.1	1,136.3	1,137.5	1,146.0	5	21	120	15.5	23	279	(**15**, 1,221.4)	18000
h10_c100_l150_04	12	1	12–15	125	1,197.0	1,206.4	1,207.6	1,221.4	12	23	90	9.4	15	959	–	18000
h10_c100_l150_05	12	1	12–15	183	1,185.0	1,207.1	1,212.6	1,218.6	22	34	90	9.2	10	259	(**16**, 1,324.6)	18000
h10_c100_l150_06	12	0	12–13	221	946.7	962.2	965.7	973.0	8	19	30	6.3	8	733	–	18000
h10_c100_l150_07	12	1	12–14	222	1,056.3	1,068.3	1,074.9	1,080.6	19	20	90	10.8	118	798	(**15**, **1,096.7**)	14804
h10_c100_l150_08	12	0	12–13	173	930.1	931.8	940.8	953.5	10	37	60	5.1	8	507	–	18000
h10_c100_l150_09	12	0	12–14	222	1,105.6	1,124.0	1,131.7	1,134.2	11	29	30	12.0	8	679	–	18000
h10_c100_l150_10	12	1	12–14	2593	1,028.7	1,033.9	1,037.1	1,038.2	6	14	30	8.6	91	1196	–	18000
Average Gap					*5.9*	*5.2*	*4.8*	*4.2*								
h10_c100_l200_01	9	0	9	605	845.7	856.9	859.1	859.1	15	9	0	0.0	10	400	–	18000
h10_c100_l200_02	10	0	–	–	902.6	907.5	910.6	910.6	9	32	0	6.4	17	284	–	18000
h10_c100_l200_03	9	0	9–10	2293	949.2	953.9	965.3	965.3	9	10	0	0.1	60	269	(**11**, 1,011.8)	18000
h10_c100_l200_04	9	1	9–10	884	954.3	963.8	964.7	964.7	13	8	0	0.1	12	424	(**11**, 1,010.1)	18000
h10_c100_l200_05	9	0	9	370	932.3	945.9	945.9	945.9	18	1	0	0.4	136	310	–	18000
h10_c100_l200_06	9	1	9	806	844.2	860.0	865.5	865.5	10	31	0	0.0	7	432	–	18000
h10_c100_l200_07	9	0	9	535	911.1	914.9	918.9	918.9	11	20	0	11.0	29	317	–	18000
h10_c100_l200_08	9	0	9	796	853.2	859.4	859.4	859.4	13	2	0	0.0	8	431	–	18000
h10_c100_l200_09	9	1	9	714	900.6	916.6	922.8	922.8	16	7	0	19.7	64	130	–	18000
h10_c100_l200_10	9	0	9	604	877.1	879.8	880.0	880.0	10	7	0	1.5	20	633	–	18000
Average Gap					*6.6*	*5.7*	*5.3*	*5.3*								

Table 45
Results for proposed instances with 20 hotels, 100 clients with ng sets of size 8.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c100_l100_01	18	0	18–20	3	913.2	935.0	950.0	951.6	37	28	95	37.2	572	154	(**21**, **971.8**)	10870
h20_c100_l100_02	19	2	19–22	2148	1,056.4	1,067.4	1,072.3	1,081.6	38	39	838	42.4	89	875	(**23**, 1,221.8)	18000
h20_c100_l100_03	18	0	18–21	5	1,005.2	1,039.7	1,045.2	1,053.4	48	22	386	38.1	47	836	(**22**, 1,123.4)	18000
h20_c100_l100_04	18	0	18–21	3	1,024.4	1,044.4	1,054.1	1,070.0	47	46	458	42.3	295	234	–	18000
h20_c100_l100_05	18	0	18–23	3	1,209.4	1,226.8	1,240.5	1,250.6	61	32	303	39.7	646	46	(**24**, 1,299.9)	18000
h20_c100_l100_06	18	0	18–21	24	1,005.4	1,020.3	1,044.4	1,055.1	36	52	560	41.7	119	447	–	18000
h20_c100_l100_07	18	2	18–24	3	1,316.4	1,329.4	1,342.2	1,353.7	55	38	524	46.5	111	572	–	18000
h20_c100_l100_08	18	0	18–21	28	973.0	995.6	1,006.6	1,019.0	58	44	655	38.1	53	344	–	18000
h20_c100_l100_09	18	0	18–21	74	970.5	984.6	993.8	1,001.1	25	21	171	37.6	63	1113	(**22**, 1,066.9)	18000
h20_c100_l100_10	18	0	18–21	39	982.5	1,006.6	1,011.0	1,015.3	41	28	90	35.8	17	833	–	18000
Average Gap					*7.9*	*6.1*	*5.1*	*4.5*								
h20_c100_l150_01	12	0	12	16	807.5	828.4	832.2	839.9	57	32	120	2.3	26	301	(**13**, 908.0)	18000
h20_c100_l150_02	13	2	13	5495	864.8	882.1	882.1	889.2	48	21	180	2.9	188	450	–	18000
h20_c100_l150_03	12	0	12–13	1155	841.9	878.4	880.0	885.8	51	13	270	2.7	32	633	–	18000
h20_c100_l150_04	12	0	12	14	843.2	874.1	874.5	885.8	54	16	210	38.7	695	16	–	18000
h20_c100_l150_05	12	0	12–13	136	848.5	883.9	887.8	892.8	62	18	210	3.9	23	456	(**14**, 951.7)	18000
h20_c100_l150_06	12	2	12	20	824.0	839.5	851.7	864.5	37	28	180	5.4	157	172	(**13**, 902.1)	18000
h20_c100_l150_07	12	0	12–13	23	913.0	929.1	940.0	950.4	52	43	150	6.1	11	645	–	18000
h20_c100_l150_08	12	0	12	19	834.2	849.1	851.7	855.6	53	22	120	5.7	450	8	–	18000
h20_c100_l150_09	12	0	12	16	811.7	838.3	842.0	847.8	38	35	150	3.4	221	178	(**13**, 873.8)	18000
h20_c100_l150_10	12	0	12	18	829.4	846.9	848.8	859.5	48	21	180	10.0	94	544	(**13**, 912.1)	18000
Average Gap					*8.4*	*5.7*	*5.3*	*4.4*								
h20_c100_l200_01	9	0	9	56	770.4	791.1	796.4	799.2	58	12	30	0.0	9	716	–	18000
h20_c100_l200_02	10	2	–	–	805.4	837.1	838.8	840.3	60	16	30	0.0	9	630	–	18000
h20_c100_l200_03	9	0	9	51	793.3	828.8	829.5	831.6	41	9	45	0.0	8	571	–	18000
h20_c100_l200_04	9	0	9	57	795.4	825.1	826.5	832.5	55	23	45	0.0	9	674	–	18000
h20_c100_l200_05	9	0	9	56	791.3	817.7	824.4	826.4	58	19	60	0.1	0	2232	–	18000
h20_c100_l200_06	9	3	9	68	769.1	790.1	795.7	797.0	43	20	30	0.0	9	3068	–	18000
h20_c100_l200_07	9	0	9	59	824.5	841.3	844.3	850.7	54	36	45	0.0	8	736	–	18000
h20_c100_l200_08	9	0	9	65	792.8	813.3	815.3	816.1	46	12	30	0.0	9	3368	–	18000
h20_c100_l200_09	9	0	9	63	755.6	779.9	780.8	788.4	48	15	60	0.0	8	2554	–	18000
h20_c100_l200_10	9	0	9	61	769.0	793.6	793.7	796.0	44	12	60	0.0	8	214	(**10**, 893.2)	18000
Average Gap					*13.5*	*11.1*	*10.8*	*10.6*								

Table 46
Results for proposed instances with 20 hotels, 100 clients with ng sets of size 12.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c100_l100_01	18	0	18–20	5	913.4	935.0	950.0	951.6	37	25	85	37.2	574	152	(**21**, **971.8**)	5655
h20_c100_l100_02	19	0	19–22	995	1,056.4	1,067.4	1,072.4	1,081.6	37	40	497	42.4	97	1996	(**23**, 1,220.2)	18000
h20_c100_l100_03	18	0	18–21	5	1,005.2	1,039.7	1,045.2	1,053.4	40	19	252	38.1	609	2426	(**22**, 1,100.4)	18000
h20_c100_l100_04	18	0	18–21	6	1,024.4	1,044.5	1,054.3	1,070.0	58	52	415	42.3	788	206	–	18000
h20_c100_l100_05	18	0	18–23	5	1,209.6	1,226.8	1,240.8	1,250.6	52	26	214	39.7	1512	62	(**24**, 1,299.9)	18000
h20_c100_l100_06	18	0	18–21	14	1,005.7	1,020.7	1,044.4	1,055.1	37	46	383	41.7	479	1530	(**22**, 1,116.1)	18000
h20_c100_l100_07	18	0	18–24	9	1,316.5	1,329.4	1,342.4	1,353.7	58	41	425	46.5	109	1346	–	18000
h20_c100_l100_08	18	0	18–21	18	973.0	995.6	1,006.7	1,019.0	59	43	617	38.1	38	602	–	18000
h20_c100_l100_09	18	0	18–21	38	971.0	985.0	995.0	1,001.1	25	19	143	37.6	13	1311	–	18000
h20_c100_l100_10	18	0	18–21	22	982.6	1,006.6	1,011.0	1,015.3	42	27	60	35.8	274	818	(**22**, **1,023.7**)	10152
Average Gap					*7.9*	*6.1*	*5.1*	*4.5*								
h20_c100_l150_01	12	0	12	29	808.9	829.0	833.2	838.0	44	28	60	2.6	146	477	(**13**, 879.5)	18000
h20_c100_l150_02	13	0	13	5220	866.3	882.4	882.5	887.6	39	15	90	2.9	176	496	(**14**, 951.4)	18000
h20_c100_l150_03	12	0	12–13	768	843.5	878.4	880.7	884.6	42	9	90	2.7	46	820	(**14**, 932.7)	18000
h20_c100_l150_04	12	0	12	27	844.7	876.1	877.1	883.1	52	11	90	37.0	623	82	–	18000
h20_c100_l150_05	12	0	12–13	120	848.7	883.9	888.4	890.9	58	12	30	3.9	149	492	(**14**, 935.0)	18000
h20_c100_l150_06	12	0	12	28	825.5	840.6	852.4	857.3	34	15	30	6.9	318	402	–	18000
h20_c100_l150_07	12	0	12–13	47	914.3	930.5	940.9	946.0	49	39	60	6.1	11	752	–	18000
h20_c100_l150_08	12	0	12	35	834.2	849.3	852.9	855.7	56	34	90	5.9	588	52	–	18000
h20_c100_l150_09	12	0	12	30	812.2	839.2	842.8	846.7	39	43	120	4.1	386	216	(**13**, 873.8)	18000
h20_c100_l150_10	12	0	12	30	830.1	849.3	850.6	858.9	39	18	105	10.2	268	430	(**13**, 888.7)	18000
Average Gap					*8.3*	*5.6*	*5.2*	*4.7*								
h20_c100_l200_01	9	0	9	155	771.0	792.3	795.9	795.9	50	14	0	0.0	9	2386	–	18000
h20_c100_l200_02	10	0	–	–	806.3	837.1	838.9	838.9	62	16	0	0.0	9	512	–	18000
h20_c100_l200_03	9	0	9	117	793.5	828.8	829.5	829.5	42	8	0	0.0	8	3424	–	18000
h20_c100_l200_04	9	0	9	145	797.0	825.7	826.6	826.6	48	16	0	0.0	9	2503	–	18000
h20_c100_l200_05	9	0	9	114	792.5	819.2	824.9	824.9	53	12	0	0.1	9	1971	–	18000
h20_c100_l200_06	9	0	9	152	769.5	790.1	796.0	796.0	38	22	0	0.0	11	420	(**10**, 918.5)	18000
h20_c100_l200_07	9	0	9	154	824.7	841.6	844.5	844.5	46	21	0	0.0	8	2583	–	18000
h20_c100_l200_08	9	0	9	161	793.2	813.8	815.1	815.1	47	4	0	0.0	11	435	(**10**, 884.7)	18000
h20_c100_l200_09	9	0	9	171	755.6	780.0	781.1	781.1	39	4	0	0.0	8	2042	–	18000
h20_c100_l200_10	9	0	9	144	769.0	793.9	794.0	794.0	37	6	0	0.0	8	11629	(**10**, 928.0)	18000
Average Gap					*13.5*	*11.0*	*10.8*	*10.8*								

Table 47
Results for proposed instances with 20 hotels, 100 clients with ng sets of size 16.

Instance	Trips LB		Infeas Trips		Relax Lower Bound				Relax Num Cuts			% Fixed	Branching		Solution	Total Time
	Num	Time	Num	Time	No Cuts	SEC	2-Path	lm-SRC	SEC	2-Path	lm-SRC		Hotel	Edge		
h20_c100_l100_01	18	0	18–20	19	913.4	935.0	950.0	951.6	41	25	85	37.2	568	160	(**21**, **971.8**)	11438
h20_c100_l100_02	19	0	19–22	2101	1,056.4	1,067.4	1,072.4	1,081.6	38	42	409	42.4	89	861	(**23**, 1,187.1)	18000
h20_c100_l100_03	18	0	18–21	16	1,005.2	1,039.7	1,045.2	1,053.4	43	19	260	38.1	182	1328	(**22**, 1,100.4)	18000
h20_c100_l100_04	18	1	18–21	19	1,024.4	1,044.5	1,054.3	1,070.0	52	53	392	42.3	326	238	–	18000
h20_c100_l100_05	18	1	18–23	19	1,209.6	1,226.8	1,240.8	1,250.6	54	23	195	39.7	762	46	(**24**, 1,299.9)	18000
h20_c100_l100_06	18	0	18–21	31	1,005.7	1,020.7	1,044.4	1,055.1	43	44	384	41.7	53	853	(**22**, 1,125.5)	18000
h20_c100_l100_07	18	2	18–24	29	1,316.5	1,329.4	1,342.4	1,353.7	58	39	430	46.5	95	688	–	18000
h20_c100_l100_08	18	0	18–21	49	973.0	995.7	1,006.7	1,019.0	61	43	468	38.1	29	456	–	18000
h20_c100_l100_09	18	1	18–21	75	971.0	985.0	995.0	1,001.1	27	19	141	37.6	34	1859	(**22**, 1,069.0)	18000
h20_c100_l100_10	18	0	18–21	32	982.6	1,006.6	1,011.0	1,015.3	43	27	60	35.8	94	0	(**22**, **1,023.7**)	278
Average Gap					*7.9*	*6.1*	*5.1*	*4.5*								
h20_c100_l150_01	12	1	12	131	808.9	829.1	833.8	833.8	42	25	0	1.4	69	608	–	18000
h20_c100_l150_02	13	0	–	–	866.3	882.4	882.5	882.5	38	5	0	2.9	430	2	–	18000
h20_c100_l150_03	12	0	12–13	3022	843.5	878.4	880.8	884.6	45	9	60	2.7	81	618	–	18000
h20_c100_l150_04	12	0	12	106	844.8	876.1	877.1	879.7	50	9	30	27.9	453	46	–	18000
h20_c100_l150_05	12	0	12–13	402	848.7	883.9	888.4	890.4	58	12	30	3.9	14	344	(**14**, 974.4)	18000
h20_c100_l150_06	12	0	12	122	825.5	840.6	852.4	852.4	40	17	0	6.8	216	442	–	18000
h20_c100_l150_07	12	0	12–13	233	914.3	930.5	941.1	943.2	52	36	30	6.1	14	480	–	18000
h20_c100_l150_08	12	0	12	155	834.2	849.3	852.9	852.9	45	21	0	6.0	265	58	–	18000
h20_c100_l150_09	12	1	12	118	812.2	839.3	842.8	844.4	41	35	30	4.1	113	358	–	18000
h20_c100_l150_10	12	0	12	109	830.1	849.3	850.6	852.9	32	12	30	5.4	85	587	–	18000
Average Gap					*8.3*	*5.6*	*5.2*	*5.0*								
h20_c100_l200_01	9	0	9	1020	771.2	792.3	795.9	795.9	57	14	0	0.0	9	317	–	18000
h20_c100_l200_02	10	1	–	–	806.3	837.1	838.9	838.9	56	16	0	0.0	9	189	–	18000
h20_c100_l200_03	9	0	9	603	793.6	828.8	829.5	829.5	42	8	0	0.0	8	520	–	18000
h20_c100_l200_04	9	0	9	999	797.0	825.7	826.6	826.6	42	16	0	0.0	9	360	–	18000
h20_c100_l200_05	9	1	9	723	792.5	819.3	825.0	825.0	51	12	0	0.1	30	349	–	18000
h20_c100_l200_06	9	0	9	1132	769.5	790.1	796.2	796.2	42	23	0	0.0	9	328	–	18000
h20_c100_l200_07	9	1	9	1233	825.1	841.8	844.7	844.7	41	16	0	0.0	10	150	–	18000
h20_c100_l200_08	9	0	9	1265	793.2	813.8	815.1	815.1	52	4	0	0.0	9	377	(**10**, 956.4)	18000
h20_c100_l200_09	9	0	9	1215	755.7	780.2	783.1	783.1	39	8	0	0.0	8	240	–	18000
h20_c100_l200_10	9	0	9	1018	769.0	794.0	794.0	794.0	42	2	0	0.0	8	332	–	18000
Average Gap					*13.5*	*11.0*	*10.8*	*10.8*								

See Figs. 7–13.

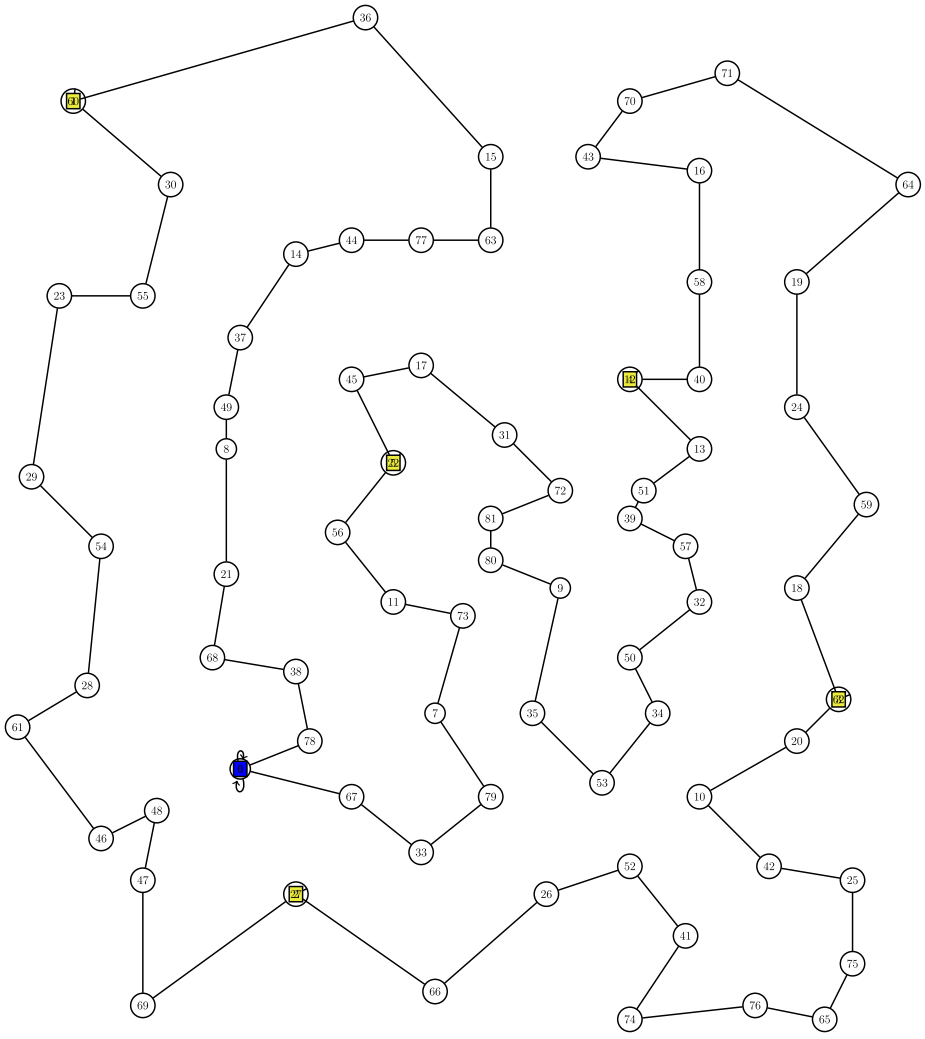

Fig. 7. Optimal solution for instance eil76.s5 (Set 3 with 5 hotels).

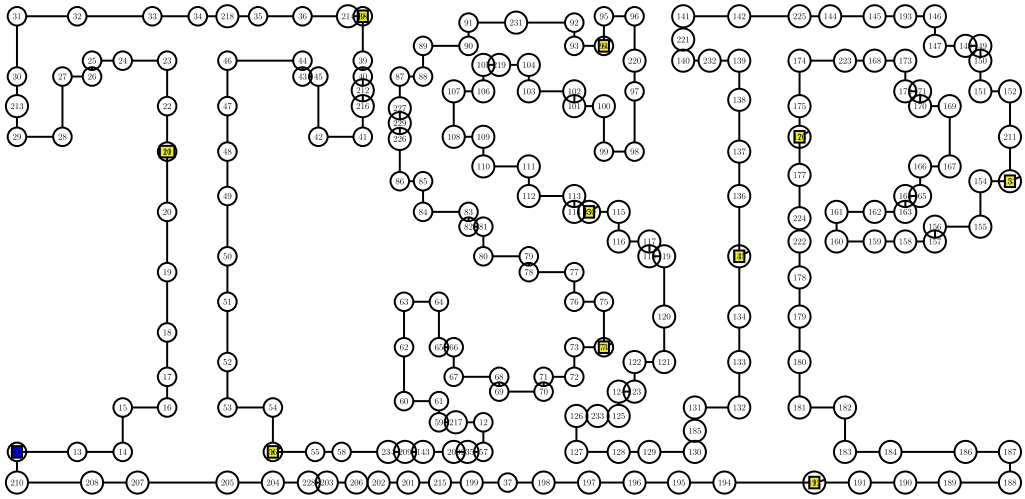

Fig. 8. Optimal solution for instance tsp225.s10 (Set 3 with 10 hotels).

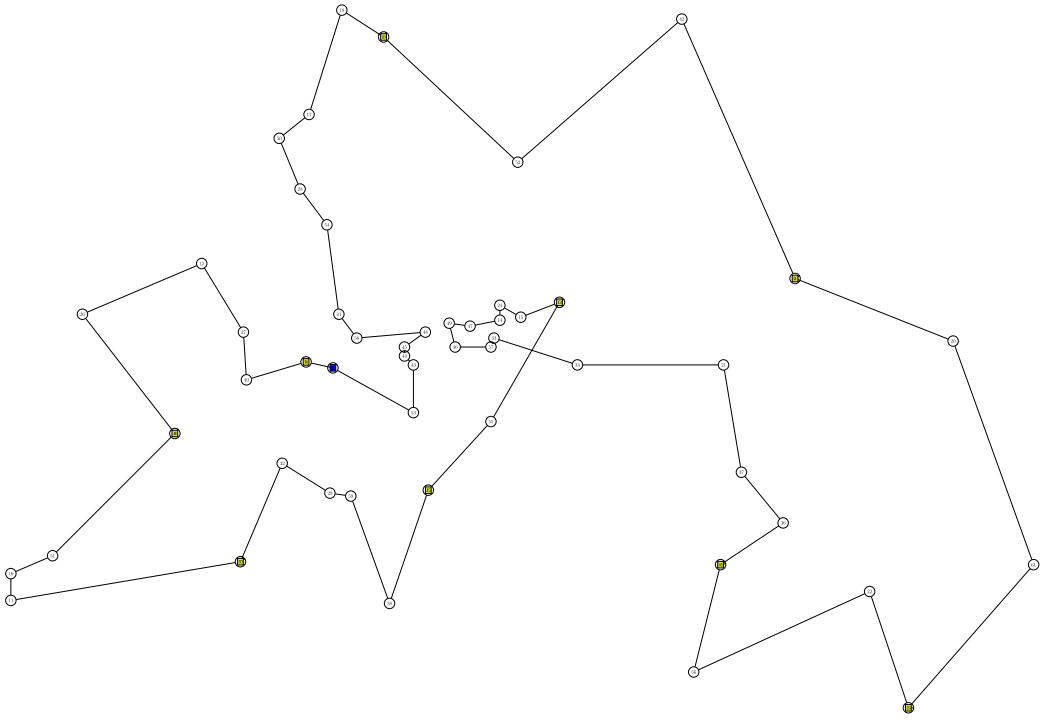

Fig. 9. Solution (8, 7,864) for instance berlin52.s10 (Set 3 with 10 hotels).

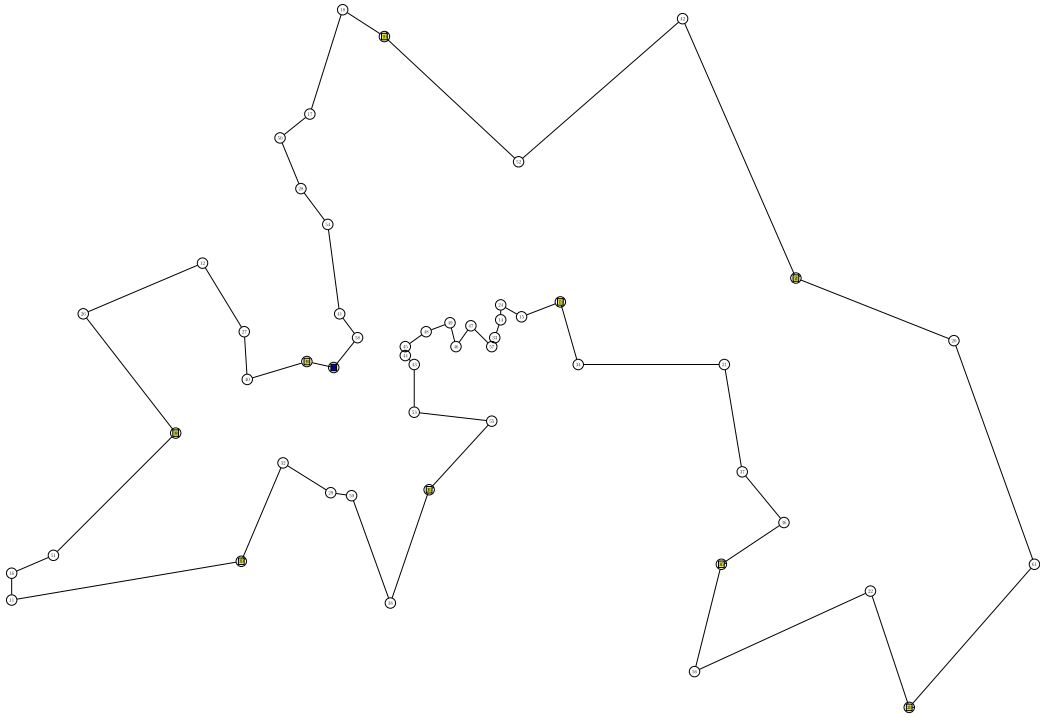

Fig. 10. Solution (9, 7,542) for instance berlin52.s10 (Set 3 with 10 hotels).

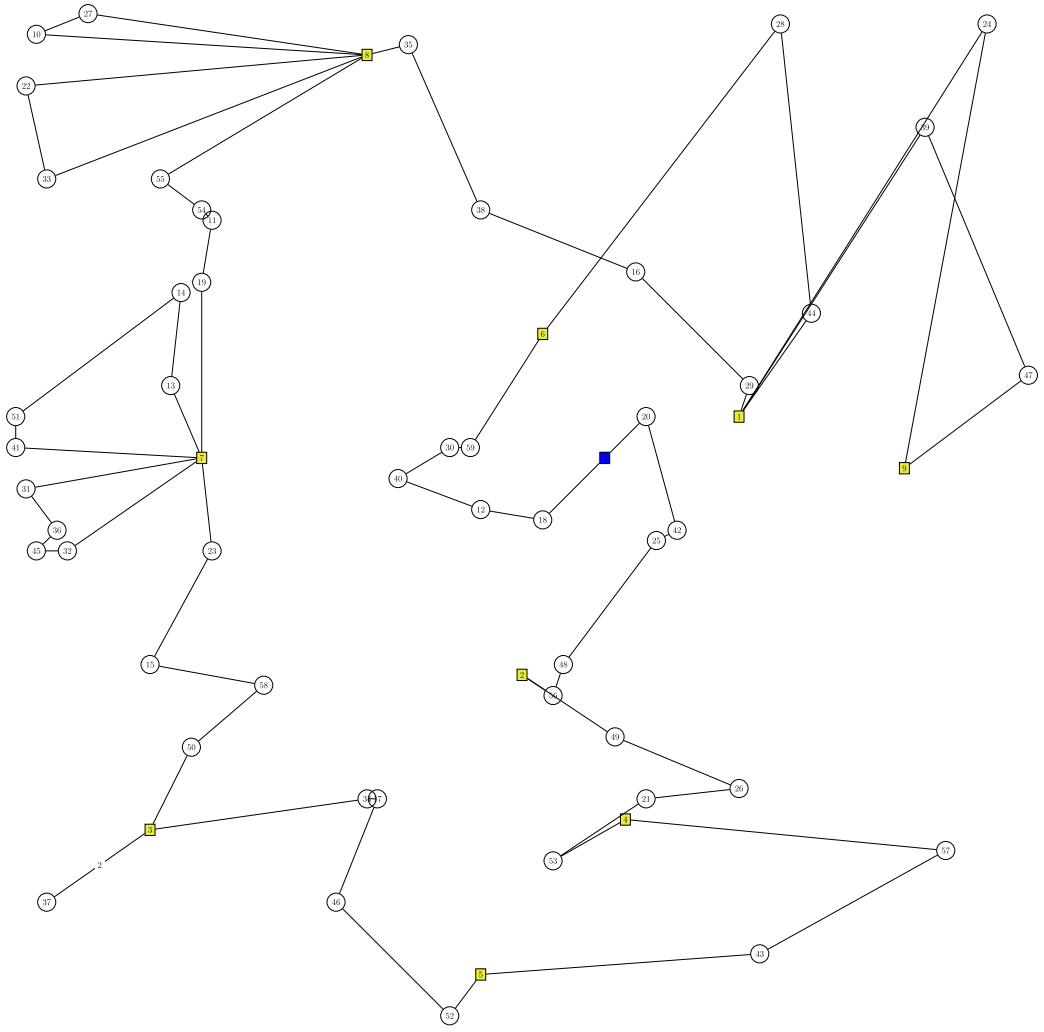

Fig. 11. Optimal solution for instance h10_c50_l100_07.

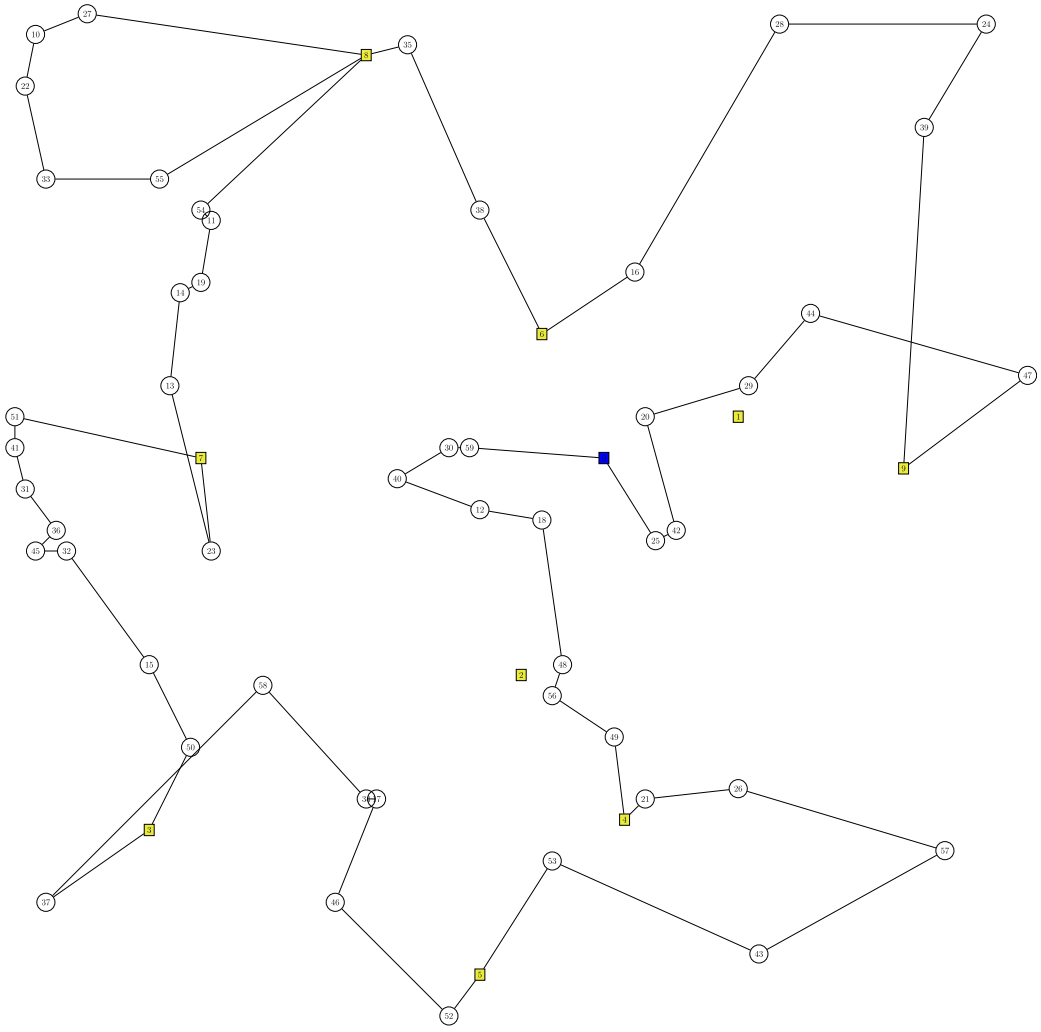

Fig. 12. Optimal solution for instance h10_c50_l150_07.

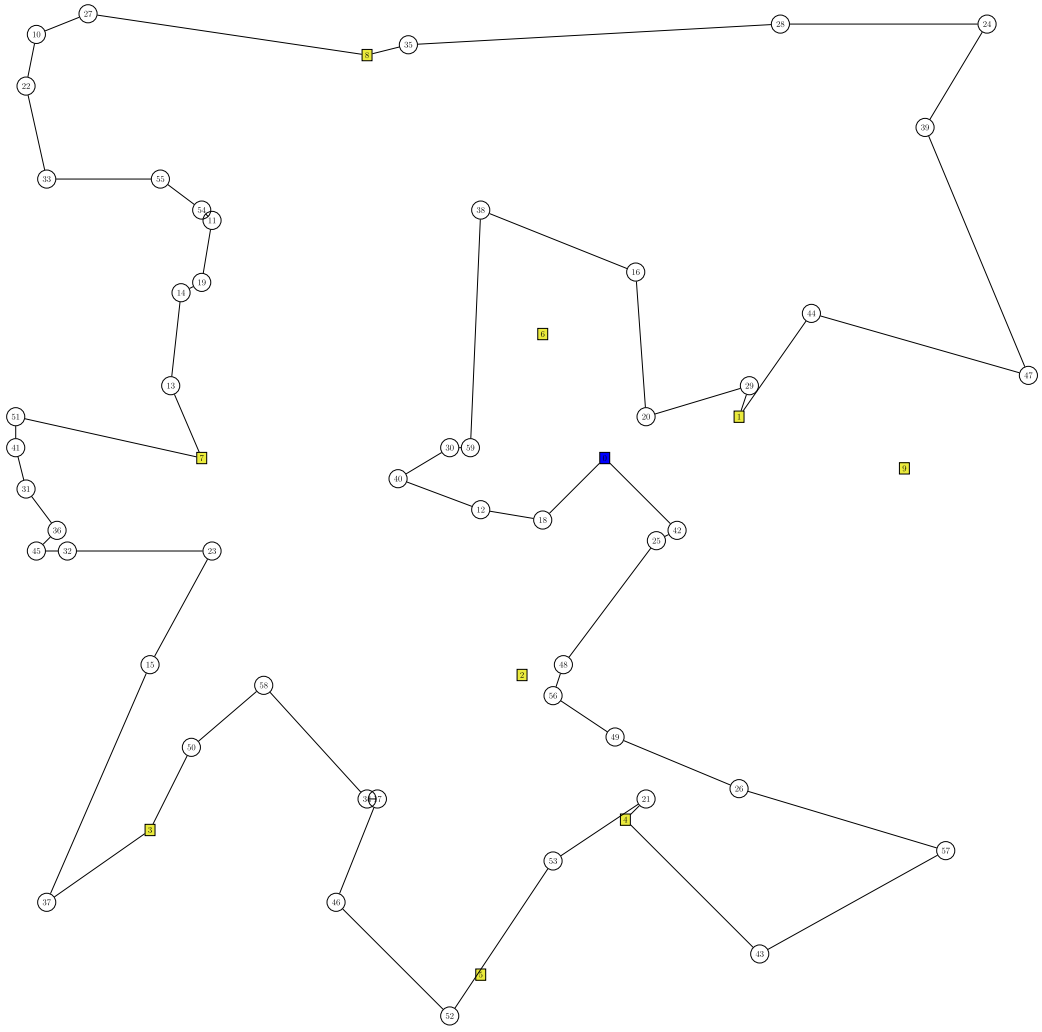

Fig. 13. Optimal solution for instance h10_c50_l200_07.

References

Applegate, D.L., Bixby, R.E., Chvátal, V., Cook, W.J., 2007. The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics). Princeton University Press, Princeton, NJ.

Baldacci, R., Mingozzi, A., Roberti, R., 2011. New route relaxation and pricing strategies for the vehicle routing problem. Oper. Res. 59 (5), 1269–1283.

Castro, M., Sörensen, K., Vansteenwegen, P., Goos, P., 2013. A memetic algorithm for the travelling salesperson problem with hotel selection. Comput. Oper. Res. 40, 1716–1728.

Castro, M., Sörensen, K., Vansteenwegen, P., Goos, P., 2015. A fast metaheuristic for the travelling salesperson problem with hotel selection. 4OR-Q 13, 15–34.

Contardo, C., Martinelli, R., 2014. A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. Discrete Optimiz. 12, 129–146.

Cordeau, J.-F., Gendreau, M., Laporte, G., 1997. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 30, 105–119.

Desaulniers, G., Desrosiers, J., Solomon, M., 2005. Column Generation. Springer.

Divsalar, A., Vansteenwegen, P., Cattrysse, D., 2013. A variable neighborhood search method for the orienteering problem with hotel selection. Int. J. Prod. Econ. 145, 150–160.

Fukasawa, R., Longo, H., Lysgaard, J., Poggi de Aragão, M., Reis, M., Uchoa, E., Werneck, R., 2006. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. Math. Program. 116 (3), 491–511.

Ghiani, G., Laporte, G., Semet, F., 2006. The black and white traveling salesman problem. Oper. Res. 54, 366–378.

Irnich, S., Desaulniers, G., Desrosiers, J., Hadjar, A., 2010. Path-reduced costs for eliminating arcs in routing and scheduling. INFORMS J. Comput. 22 (2), 297–313.

Irnich, S., Villeneuve, D., 2006. The shortest-path problem with resource constraints and k-cycle elimination. INFORMS J. Comput. 18 (3), 391–406.

Jepsen, M., Petersen, B., Spoorendonk, S., Pisinger, D., 2008. Subset-row inequalities applied to the vehicle-routing problem with time windows. Oper. Res. 56 (2), 497–511.

Kohl, N., Desrosiers, J., Madsen, O., Solomon, M., Soumis, F., 1999. 2-path cuts for the vehicle routing problem with time windows. Transp. Sci. 33 (1), 101–116.

Miller, C., Tucker, A., Zemlin, R., 1960. Integer programming formulation of traveling salesman problems. J. ACM 7 (4), 326–329.

Pecin, D.G., 2014. Exact algorithms for the capacitated vehicle routing problem (Ph. D. thesis). Pontifícia Universidade Católica do Rio de Janeiro.

Poggi, M., Uchoa, E., 2014. New exact algorithms for the capacitated vehicle routing problem. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications. SIAM, Philadelphia, Ch. 3, pp. 59–86. .

Righini, G., Salani, M., 2006. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optimiz. 3 (3), 255–273.

Schiffer, M., Schneider, M., Walther, G., Laporte, G., 2019. Vehicle routing and location routing with intermediate stops: a review. Transp. Sci. 53. .

Solomon, M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper. Res. 35, 254–266.

Sousa, M.M., Ochi, L.S., Coelho, I.M., Gonçalves, L.B., 2015. A variable neighborhood search heuristic for the traveling salesman problem with hotel selection. In: Computing Conference (CLEI), 2015 Latin American. IEEE, pp. 1–12. .

Vansteenwegen, P., Souffriau, W., Sörensen, K., 2012. The travelling salesperson problem with hotel selection. J. Oper. Res. Soc. 63, 207–217.