The Ring Spur Assignment Problem: New formulation, valid inequalities and a branch-and-cut approach

Rahimeh Neamatian Monemi[a,*], Shahin Gelareh[b]

[a]*Centre for Operational Research, Management Science and Information Systems (CORMSIS), University of Southampton, Southampton, UK*
[b]*Portsmouth Business School, University of Portsmouth, UK*

Abstract

A new mathematical model is proposed for the Ring Spur Assignment Problem (RSAP) that arises in the design of next-generation telecommunication networks. In this problem, every node of the network lies either on a ring among a set of bounded disjoint local rings or is spurred by a single arc to another node on a local ring. A special ring, called a tertiary ring, interconnects the local rings. Our new integer programming model employs only $O(n^2)$ variables and has a stronger LP relaxation. Several classes of valid inequalities and corresponding separation procedures are presented giving rise to an efficient branch-and-cut solution algorithm. We report optimal solutions for all SNDLib instances including those that have not previously been solved to optimality.

Keywords: Location-allocation, branch-and-cut, next-generation telecommunications networks

1. Introduction

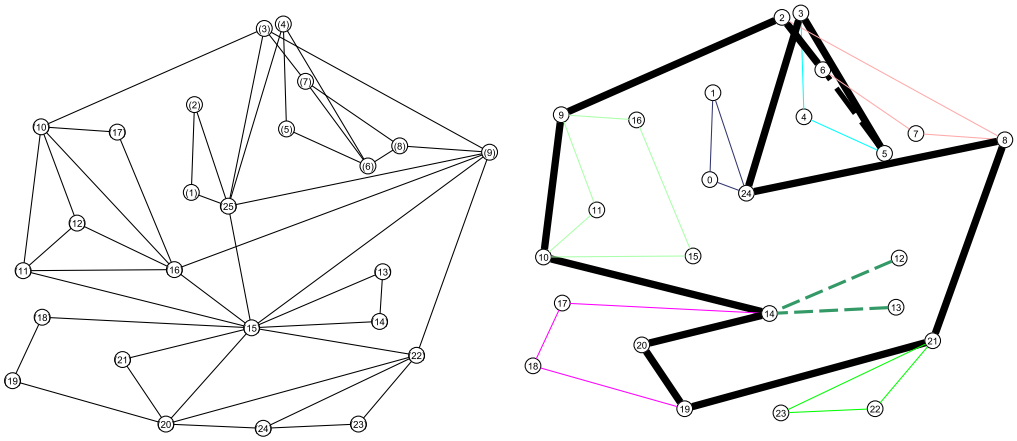
The Ring Spur Assignment Problem introduced in Carroll et al. (2011) arises in the design of next-generation telecommunications networks and can be considered as a location-allocation problem. The nodes in such a network structure either lie on a set of disjoint and bounded local rings or are spurred to other local ring nodes using a single arc. Another special (in terms of technological connections) ring interconnects these local rings and is known as a tertiary ring.

The goal is to design an economical fault-tolerant next-generation network (NGN) for a given telecommunications operator. Such a resilient network is indeed a logical topology for an existing physical infrastructure –that is, we exploit the pre-installed capacity in the physical synchronous digital hierarchy (SDH) (Carroll et al., 2013). This means that if a given node has only one incident edge, or if the residual capacity of a ring is insufficient, the node can only be a spur node connected with a single arc to another node on a local ring. Figure 1b represents an optimal solution to the instance 'France' in Figure 1a from SNDlib (Orlowski et al., 2007).

In Carroll et al. (2011), the authors proposed two formulations for this problem. The constraints in the two models are very similar, and both models use $O(n^3)$ binary variables. While the first model (F1) was rather compact, the second one (F2) was an extended version. The difference between the models lies in the fact that one has four times more constraints than the other one. The authors also showed that there is not much difference between the LP relaxation quality of these two models, but that F1, with considerably fewer constraints, shows better computational behaviour.

Carroll et al. (2013) relying on the formulation F1 in Carroll et al. (2011), identified several classes of valid inequalities and proposed the corresponding separation routines. The authors proposed an efficient branch-and-cut approach capable of solving most of the instances to optimality and in very reasonable computational times. However, this approach faced some difficulties for instances of larger size.

*Corresponding author, r.n.monemi@gmail.com



(a) The original physical network with the existing installation.

(b) The optimal solution where the local ring bounds are set to 8 edges and two nodes 12 and 13 are spurred to local ring node 14.

Figure 1: Illustration of the network of instance 'France' from SNDlib.

Carroll and McGarraghy (2013) decomposed the problem into two integer programming (IP) problems and described a branch-and-cut decomposition heuristic algorithm.

From the network topology perspective, the problem has some similarity with some variants of hub location problems such as (Rodríguez-Martín et al., 2014), the plant-cycle location problem (Labbé et al., 2004), the team-orienteering problem (Fischetti et al., 1998b) and the multiple depots ring star problem (Sundar and Rathinam, 2014).

In earlier works, e.g. Carroll et al. (2013), the authors employed a ring representative node approach to model the local rings and derive the ring bound constraints, resulting in larger initial models. In this work, we use a smaller formulation with significantly fewer decision variables, and separate the ring bound inequalities in the cutting plane phase.

We refer to an undirected connection between two nodes i and j as an *edge*, e, represented by a set $e = \{i, j\}$. A directed connection from node i to node j is referred to as an *arc*, a, represented by an ordered pair $a = (i, j)$.

Let $G(V, E)$ be a graph where V represents the set of nodes and E the set of existing edges . There is at least one edge incident to every node in this underlying graph. We are also given the cost for installing an edge, c_{ij}, and a penalty for installing a spur arc (i, j), which is an instance-dependent constant value, b, that augments the cost to bc_{ij}.

2. Mixed Integer Linear Programming Formulation

In this section, we present our 2-index IP formulation for the Ring Spur Assignment Problem (RSAP2). The parameters are listed in Table 1. It must be noted that b is an instance-dependent penalty parameter calculated independently for every problem instance in such a way that spur arcs are discouraged (see Carroll et al. (2013) and Carroll et al. (2011)).

The decision variables are the following: r_{ij} is 1 if the edge $i - j$ is part of a local ring, 0 otherwise. y_{ij} is 1 if edge $i - j$ is on the tertiary ring, 0 otherwise. We define y_{ii} to be 1 if node i is a tertiary node, 0 otherwise. t_{ij} is 1 if node i is an isolated (spur) node connected (arc) to a node j, 0 otherwise. In addition, t_{ii} is 1 if i is a node that is *not* spurred to any other node. We refer to such a node as *non-isolated* node.

We also introduce $S \subset V$ and $\delta(S) = \{e = \{i, j\} | i \in S, j \in V/\ S\}$ and $\gamma(S) = \{e = \{i, j\} | i, j \in S, j \neq i\}$.

c_{ij}:	the cost associated with installing the connection (i,j),
b:	the penalty cost coefficient applied to a spur edge,
R:	the maximum number of nodes that can lie on a local ring, and
n:	the number of nodes.

Table 1: Model Parameters.

2.1. A 2-index formulation for the RSAP

Our new mathematical model is presented in the sequel.

$$(RSAP2) \quad \min \sum_{e\in E} c_e y_e + \sum_{e\in E} c_e r_e + \sum_{a=(i,j):\{i,j\}\in E} b\, c_a t_a \tag{1}$$

$$s.\,t.$$

$$\sum_{e\in\delta(k)} y_e = 2y_{kk} \qquad \forall k \in V, \tag{2}$$

$$\sum_{e\in\delta(k)} r_e + 2 \sum_{l\in V: l\neq k, \{k,l\}\in E} t_{kl} = 2 \qquad \forall k \in V, \tag{3}$$

$$t_{ij} + y_{ii} \leq 1 \qquad \forall i, j \in V : j \neq i, \{i,j\} \in E, \tag{4}$$

$$r_e \leq t_{ii} \qquad \forall i \in V, e \in \delta(i), e \in E, \tag{5}$$

$$r_e \leq t_{jj} \qquad \forall j \in V, e \in \delta(i), e \in E, \tag{6}$$

$$y_e \leq t_{ii} \qquad \forall i \in V, e \in \delta(i), e \in E, \tag{7}$$

$$y_e \leq t_{jj} \qquad \forall j \in V, e \in \delta(i), e \in E, \tag{8}$$

$$y_e \leq y_{ii} \qquad \forall i \in V, e \in \delta(i), e \in E, \tag{9}$$

$$y_e \leq y_{jj} \qquad \forall j \in V, e \in \delta(i), e \in E, \tag{10}$$

$$t_{ij} + t_{ii} \leq 1 \qquad \forall i, j \in V : j \neq i, \{i,j\} \in E, \tag{11}$$

$$t_{ij} \leq t_{jj} \qquad \forall i, j \in V : j \neq i, \{i,j\} \in E, \tag{12}$$

$$y_{ii} \leq t_{ii} \qquad \forall i \in V, \tag{13}$$

$$\sum_{j\in V: j\neq i, \{i,j\}\in E} t_{ij} + t_{ii} = 1 \qquad \forall i \in V, \tag{14}$$

$$\sum_{i\in V} y_{ii} \geq 3, \tag{15}$$

$$\sum_{e\in E} y_e \geq 3, \tag{16}$$

$$\sum_{i\in V} y_{ii} \geq \frac{\sum_{i\in V} t_{ii}}{R}, \tag{17}$$

$$r \in \{0,1\}^{|E|}, y \in \{0,1\}^{|E|+|V|}, t \in \{0,1\}^{2|E|+|V|}. \tag{18}$$

The objective function (1) accounts for the total installation costs. It is comprised of the cost of installing tertiary edges and secondary ring edges plus the penalizing cost for installing spur arcs. Constraints (2) make sure that exactly two tertiary edges are incident to every tertiary node. Constraints (3) ensure that a node is either a local ring node, in which case there are two edges incident to this node, or is a node allocated (spurred) to another node, in which case there is only one incident arc. Constraints (4) state that if i is a tertiary node, it cannot be spurred (allocated) to another node, and vice versa. Constraints (5)-(6) make sure that both end-points of a local ring edge are non-spur nodes. Constraints (7)-(8) state that none of the end-points of a tertiary edge can be a spur node, while constraints (9)-(10) ensure that both

end-points of a tertiary edge must be tertiary nodes. For a spur arc (i, j), node j must be a non-spur node while i cannot be a non-spur node. Constraints (11)-(12) guarantee this. A node can be a tertiary node if it is not a spur node, as in constraints (13). Every node is either a non-spur node or is allocated (spurred) to another node. This is stated in constraints (14). Constraints (15)-(17) set a lower bound on the number of tertiary nodes appearing in any feasible solution.

The model is completed once *subtour elimination* constraints and *ring bound* constraints are added to (2)-(18).

2.2. Subtour elimination constraints (SECs)

Subtours must be avoided at two levels: the *tertiary ring level* of the network and the *local ring level*.

2.2.1. Tertiary ring level

Two types of Generalized Subtour Elimination Constraints (GSECs) can be obtained here. The first type ensures that every cut size in a feasible tertiary-level sub-network is at least equal to 2, and the second type breaks two edges from the union of two components, if any.

a) let $S \subset V$ and $S' = V \setminus S$:

$$\sum_{e \in \delta(S)} y_e \geq 2(y_{ii} + y_{jj} - 1), \qquad \forall i \in S, j \in S'. \tag{19}$$

b) let S_a and S_b be two disjoint components of the tertiary-level network (Carroll et al., 2013):

$$\sum_{e \in \gamma(S_a) \cup \gamma(S_b)} y_e \leq |S_a \cup S_b| - 2. \tag{20}$$

2.2.2. Local ring level

In any solution to this problem, e.g. in Figure 1b, the cut size of the graph (after duplicating the spur arcs in the opposite directions), as a whole, is at least equal to 2.

$$\sum_{e=\{i,j\} \in \delta(S): i \in S} \left(r_e + y_e + 2t_{ij}\right) \geq 2, \qquad \forall S \subset V. \tag{21}$$

2.3. Ring bound inequalities

Carroll et al. (2013) defines the ghost ring as follows. We establish the support graph of a fractional solution and identify cycles that are longer than the ring bound, if any. We call these infeasible rings *ghosts*. In such cases, at least two edges must be removed from every such ring.

$$\sum_{e \in \gamma(G_r)} r_e \leq |G_r| - \max\{\lceil |G_r|/R \rceil, 2\}, \qquad \forall \text{ Ghost ring } G_r. \tag{22}$$

The fundamental difference between RSAP and RSAP2 is that in the former the definition of variables allows an explicit presentation of ring bound constraints in the model while in the latter, the bounds are captured by separating and adding valid inequalities (22)in the branch-and-cut process.

A more general form of these inequalities is discussed in the next section.

3. Branch-and-cut Algorithm

A branch-and-cut approach is proposed to solve RSAP instances using the RSAP2 formulation. Our approach is composed of the exploration of a decision tree, a cutting plane phase that improves the linear programming approximation of the problem, a preprocessing phase that may fix some variables, the identification of instance-dependent valid inequalities and a local search to polish the incumbent solution that is either found by visiting an integer feasible node or is proposed by another heuristic. The efficiency of a branch-and-cut approach relies highly on the quality of the initial relaxation, the sharpness of the cuts separated at the cutting plane phase and the efficiency of the separation procedures.

3.1. Initial relaxation

The initial relaxation of the model is composed of constraints (2)-(18) with the exception of constraints (9)-(10). Constraints (9)-(10) are removed (but will be separated during our branch-and-cut) because, beside the numerical instability that they introduce, their presence in the initial relaxation noticeably increases the efforts needed to obtain an integer solution. It has been observed that their link with constraints (13) leads to the production of a highly fractional solution and makes various branching schemes significantly less effective. The result is that there are more nodes to be processed and there is an increase in computational time. Some preliminary computational experiments suggested removing constraints (9)-(10) instead of constraints (13), from the initial relaxation.

3.2. Preprocessing

As in most real-life applications, some instance-dependent structures can be identified, and this information can be exploited to fix some variables or derive instance-dependent valid inequalities. In Figure 2, two situations are identified and explained.

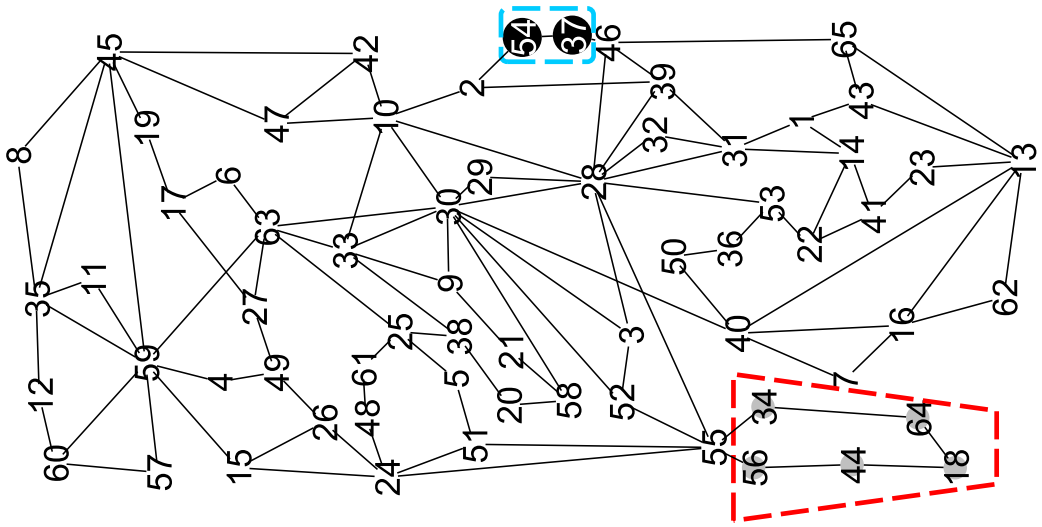


Figure 2: The underlying network structure of 'ta2' from SNDlib.

Lemma 1. *Let i, j and k be three consecutively connected nodes each of which has a degree of 2 in the initial network structure. In any feasible solution of RSAP2, $r_{ij}, r_{jk}, t_{ii}, t_{jj}$ and t_{kk} are at their upper bound. Therefore, they can be fixed at 1. Moreover, if any of the three nodes becomes a tertiary node, then the other two will also have to become tertiary nodes. This result gives rise to the following inequalities:*

$$\sum_{e \in E'} y_e + y_{jj} + y_{kk} \geq 4 y_{ii} \qquad \forall \{i, j, k\} \subseteq V, E' = \{\{i, j\}, \{j, k\}\} \subseteq E \tag{23}$$

$$\sum_{e\in E'} y_e + y_{ii} + y_{kk} \geq 4y_{jj} \qquad \forall\{i,j,k\} \subseteq V, E' = \{\{i,j\},\{j,k\}\} \subseteq E \tag{24}$$

$$\sum_{e\in E'} y_e + y_{ii} + y_{jj} \geq 4y_{kk} \qquad \forall\{i,j,k\} \subseteq V, E' = \{\{i,j\},\{j,k\}\} \subseteq E \tag{25}$$

An example of such case can be observed in Figure 2 for the grey-filled nodes in the south-east of the figure (nodes 18, 34, 44, 56 and 64). Of these five nodes, none can exclusively be a tertiary node because: 1) for every tertiary node, two tertiary edges are incident; and 2) the other two vertices of such incident edges must also be tertiary nodes (i.e., node 55 must also become a tertiary node). Now, we have some tertiary nodes (i.e. nodes 18, 34, 44, 56 and 64) that are not serving any local ring (unless a complete local ring 55 – 34 – 64 – 18 – 44 – 56 – 55 coincides with the tertiary edges, in which case the other nodes cannot form a feasible local ring), and no isolated node is allocated to them as the underlying network structure does not permit this. Therefore, some of the other nodes may be spurred to node 55, but the rest (which can be greater than R in number) cannot form one single local ring passing through the tertiary node 55. It is also not possible to have a second tertiary ring, as the tertiary level graph must be a connected one.

Lemma 2. *Let i and j be two consecutively connected nodes each of which has a degree of 2 in the initial network structure. Let also $i-k$ and $j-l$ be the unique edges incident to i and j, respectively. In any feasible solution of RSAP2, we have:*

$$\frac{1}{2}(t_{ik} + t_{jl}) + \frac{1}{3}\sum_{e\in E'} r_e \geq 1, \qquad \forall\{i,j,k,l\} \subseteq V,\ E' = \{\{k,i\},\{j,l\},\{i,j\}\} \subseteq E. \tag{26}$$

An example of such a case can be observed in Figure 2 for the black-filled nodes on the northern part of the figure (nodes 37 and 54). Either node 54 is spurred at node 2 and node 37 is spurred at node 46, or they form a sequence of 2 – 54 – 37 – 46 edges on a local ring, as nodes 54 and 37 cannot be exclusively tertiary nodes with neither a local ring passing through them nor any isolated node spurred at them.

3.3. Local search

To the best of our knowledge, no heuristic algorithm has so far been proposed for this problem. Our aim here is to use the feasibility pump heuristic (Fischetti et al., 2005) to find a feasible solution during B&B and to propose a way to polish (if possible) and improve such solutions on the basis of the information from the problem instance. This procedure is required to be very inexpensive, while being effective. Because it will be called on every candidate integer solution (visited during the branch-and-bound) before accepting it as an incumbent, it must therefore not be too time-consuming. In what follows, we propose a set of neighbourhoods that can be explored to find solutions that are possibly better than the incumbent. Here, we use neighbourhoods of size $O(|V||E|)$. Let us assume that the network structure in Figure 3 represents the current incumbent of a branch-and-bound process. In this figure, the dashed (blue) edges represent the local ring edges, the solid (red) edges represent the tertiary edges, the dotted (green) links are the spur arcs (from an isolated node to a ring node) and the initial edges in the underlying network are the fine grey dashed lines in the background.

Neighbourhood of tertiary edges ($\mathcal{N}_1$). Let i be a non-isolated node (e.g. node 32, surrounded by a dashed (blue) star) that is connected to two other tertiary nodes k and l using both tertiary edges and local ring edges, with at least two overlapping tertiary and local ring edges (on 32 – 43 or on 32 – 3). One can verify the possibility of directly connecting the two tertiary nodes k and l (by a tertiary edge 43 – 3) instead of the edge $i-k$ (32-43) and $i-l$ (32-3), to obtain a feasible solution that one would hope would have a smaller objective value, if the triangle inequality holds.

Figure 4, depicts a neighbouring feasible solution when $\mathcal{N}_1$ is applied to the nodes 6, 15 and 32 (surrounded by the dashed (blue) star) in Figure 3.

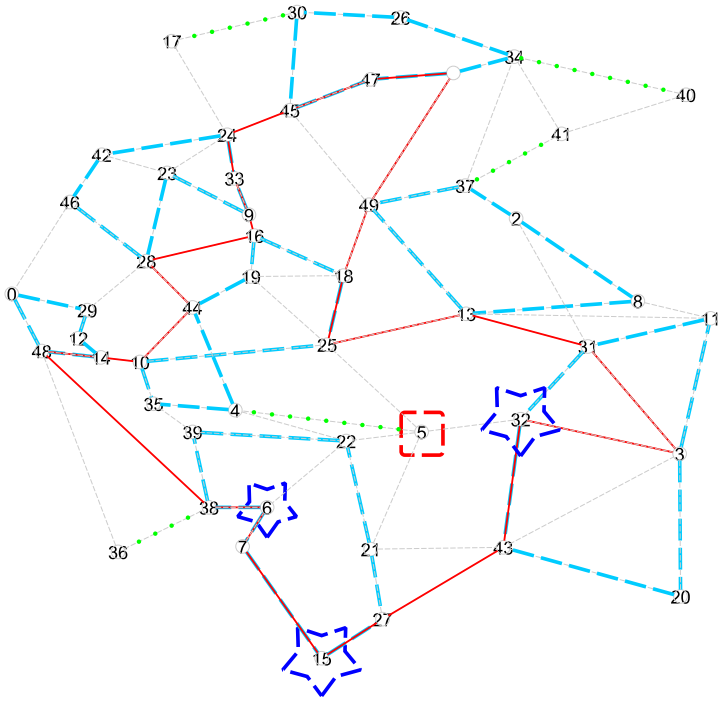


Figure 3: An incumbent found during the tree search for the problem instance 'Germany' from SNDlib.

Neighbourhood of spur arcs ($\mathcal{N}_2$). For an isolated node i spurred at node k in a solution s, the neighboring solutions with respect to this neighborhood are those that are different from s only by allocation of i to a different non-isolated node.

In Figure 3, node 5 (surrounded by a dashed (red) rectangle) is spurred at node 4. A neighbouring solution in this neighbourhood is a solution obtained by examining possible (with respect to the initial edges of the underlying network represented by the dashed (grey) edges) alternative allocations that are feasible and, it is hoped, cheaper.

One can observe that allocating node 5 to node 22, as in Figure 5, might be cheaper (if Euclidian distance holds), while it also preserves the feasibility of the solution with respect to the length of the secondary rings. Should this be true, the improvement can be significant, as the values of the spur arc penalty in Carroll et al. (2013) for the SNDlib problem instances is b, $3 \leq b \leq 17$, times more expensive than the cost of normal edges.

Neighbourhood for merging a spur node in a secondary ring ($\mathcal{N}_3$). If in a node i in solution s is spurred at a non-isolated node k, a neighboring solution with respect to this neighborhood is a solution that (if the underlying initial network allows) inserts i next to k on the route passing through k.

In Figure 3, given that the cost of a spur arc is b times more expensive than the cost of a normal edge, inserting the spur node between two other non-spur nodes of a local ring may, perhaps, improve the solution. Figure 6 depicts such a solution obtained from the solution in Figure 3.

3.4. Valid inequalities

Some other classes of valid inequalities can be presented for the RSAP2 polytope, $\mathcal{P}(RSAP2)$.

Proposition 1. *The following inequalities are deduced from the inequalities* (3) *and are valid for* $\mathcal{P}(RSAP2)$*:*

$$\sum_{e \in \delta(k)} r_e = 2t_{kk} \qquad \forall k \in V \tag{27}$$

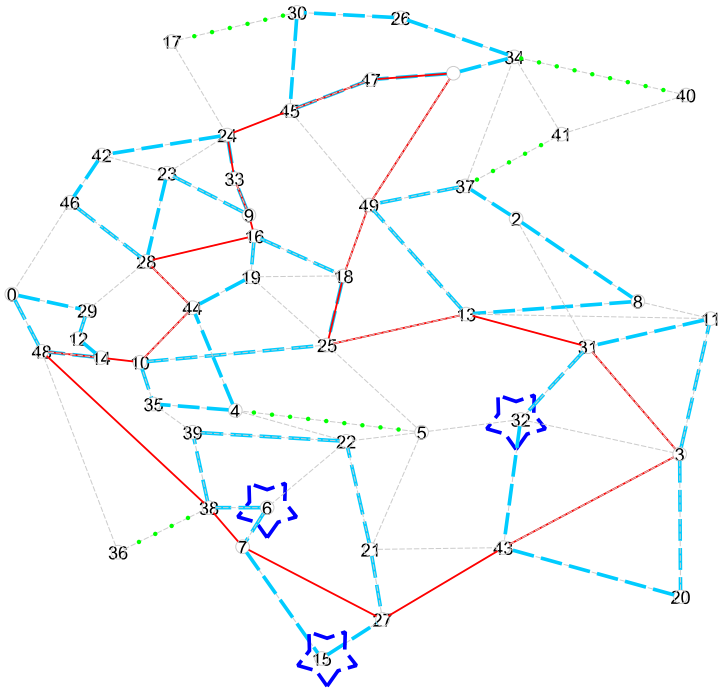


Figure 4: A feasible solution obtained by applying $\mathcal{N}_1$ to the solution in Figure 3. The edges 7 – 38 replaces 6 – 38 and 6 – 7; the edge 7 – 27 replaces 7 – 15 and 15 – 27; and 3 – 43 replaces 3 – 32 and 32 – 43.

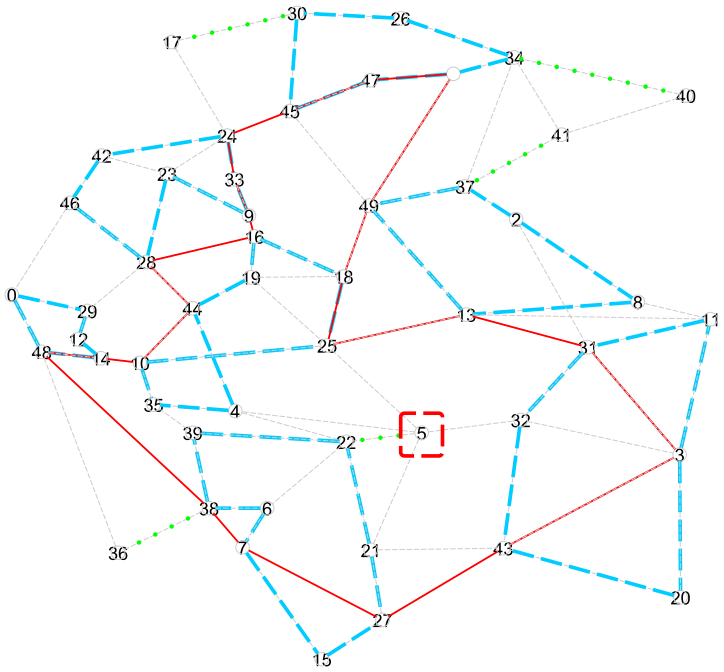


Figure 5: A feasible solution obtained by applying $\mathcal{N}_2$ to the solution in Figure 3. The isolated node 5 is now spurred at node 22.

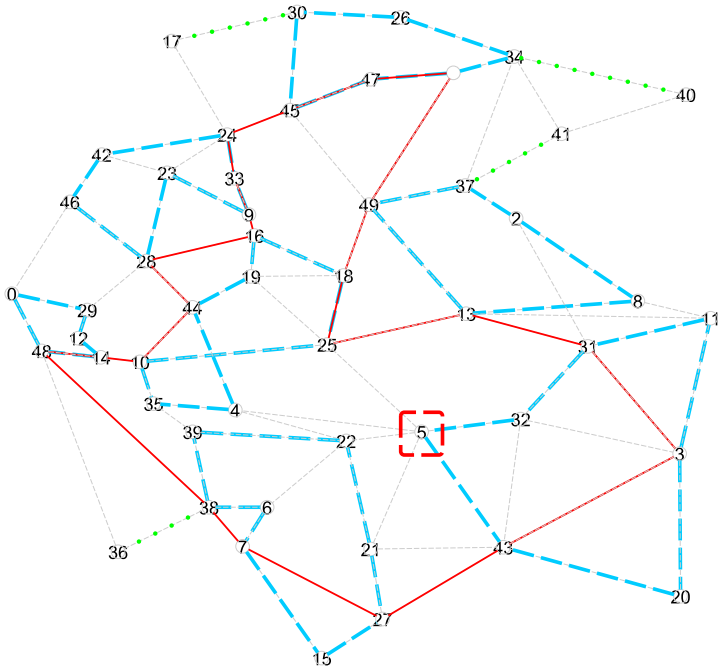


Figure 6: A feasible solution obtained by applying $\mathcal{N}_3$ to the solution in Figure 3. The edge 32 – 43 has been broken and node 5 is inserted between nodes 43 and 32.

Proof. These inequalities are easily derived by substituting (14) in (3). □

Proposition 2. *The following inequalities proposed in Rodríguez-Martín et al. (2014) are valid for $\mathcal{P}(RSAP2)$:*

$$\sum_{e\in\gamma(S)} r_e - \sum_{i\in S} y_{ii} \leq |S| - \lceil \frac{|S|}{R} \rceil \tag{28}$$

for all $S \subset V$. These inequalities are similar to the ghost inequalities (22). The difference is that while constraints (22) address the set of edges of long rings (the fractional ring solutions with more than the ring bound number of edges), the valid inequalities (28) are valid for any subset of edges.

Proposition 3. *The following inequalities are valid for $\mathcal{P}(RSAP2)$:*

$$\sum_{e\in\delta(S)} r_e \geq 2\left(\frac{\sum_{i\in S} t_{ii}}{R} - \sum_{i\in S} y_{ii}\right) \tag{29}$$

for all $S \subset V$.

Proof. See Appendix A. □

Proposition 4. *The valid inequalities (29) can be strengthened and written as*

$$\sum_{e\in\delta(S)} r_e \geq 2\left(\frac{\sum_{i\in S} t_{ii} + \sum_{e\in\delta(S)} r_e/2}{R} - \sum_{i\in S} y_{ii}\right) \tag{30}$$

for all $S \subset V$. This strengthening gives a better lower approximation of the number of edges, leaving a set $S \subset V$.

Proof. See the proof in Rodríguez-Martín et al. (2014). □

Proposition 5. *The following 2-matching constraints for both tertiary and local ring levels are valid for* $\mathcal{P}(RSAP2)$*:*

$$\sum_{e\in\gamma(H)} y_e + \sum_{e\in\mathcal{T}} y_e \leq \sum_{i\in H} y_{ii} + \lfloor\frac{|\mathcal{T}|}{2}\rfloor \tag{31}$$

$$\sum_{e\in\gamma(H)} r_e + \sum_{e\in\mathcal{T}} r_e \leq \sum_{i\in H} t_{ii} + \lfloor\frac{|\mathcal{T}|}{2}\rfloor \tag{32}$$

for all $H \subset V$ *and all* $\mathcal{T} \subset \delta(H)$ *satisfying,*
i) $|\{i,j\} \cap H| = 1, \quad \forall\{i,j\} \in \mathcal{T}$,
ii) $|\{i,j\} \cap \{k,l\}| = \emptyset, \quad \{i,j\} \neq \{k,l\} \in \mathcal{T}$, *and*
iii) $|\mathcal{T}| \geq 3$ *and odd.*
where $H \subset V$ *is a handle and* $\mathcal{T} \subset \delta(H)$ *are called teeth (with respect to the tertiary subgraph or the local ring subgraph, i.e.,* $\mathcal{T}^r, H^r, \mathcal{T}^y$ *and* H^y*).*

Proof. See Appendix A. □

Lemma 3. *For* $|\mathcal{T}| = 1$*, the valid inequalities* (31) *(*(32)*) are reduced to the constraints* (9) *and* (10) *(*(5)-(6)*).*

Proof. Consider (31) where $|\mathcal{T}| = 1$. Hence, $\gamma(H) = \emptyset$ and, therefore, $|\gamma(H)| = 0$. The proof is then complete. □

Proposition 6. *The following comb inequalities for both tertiary and local ring levels are valid for* $\mathcal{P}(RSAP2)$*:*

$$\sum_{e\in\gamma(H)} r_e + \sum_{j=1}^{t} \sum_{e\in\gamma(T_j)} r_e \leq \sum_{i\in H} t_{ii} + \sum_{j=1}^{t} |T_j| - \frac{3t+1}{2}, \tag{33}$$

$$\sum_{e\in\gamma(H)} y_e + \sum_{j=1}^{t} \sum_{e\in\gamma(T_j)} y_e \leq \sum_{i\in H} y_{ii} + \sum_{j=1}^{t} |T_j| - \frac{3t+1}{2}, \tag{34}$$

for all $H \subset V$ *and all* $T_i \subset V$, $\forall i \in \{1,\ldots,t\}$ *satisfying,*
i) $H, T_1, T_2, \ldots, T_t \subseteq V$,
ii) $T_j \setminus H \neq \emptyset, \quad \forall j \in \{1,\ldots,t\}$,
ii) $T_j \cap H \neq \emptyset, \quad \forall j \in \{1,\ldots,t\}$,
ii) $T_i \cap T_j = \emptyset, \quad \forall j \in \{1,\ldots,t\}$, *and*
iii) $t \geq 3$ *and odd.*

Proof. See Appendix A. □

Proposition 7. *The following odd-hole inequalities are valid for* $\mathcal{P}(RSAP2)$*.*

$$t_{ij} + t_{jk} + t_{ki} \leq 1, \qquad \forall\{i,j,k\} \subseteq V, \{i,j\}, \{j,k\}, \{k,i\} \in E \tag{35}$$

Proposition 8. *The following moving sub-path inequalities are valid for* $\mathcal{P}(RSAP2)$*. Let* r^* *be an integer partial solution and a violated inequality of* (22) *in the form of* $P = (v_1, v_2, \ldots, v_m, v_1) : m > R$ *exist. We can identify some minimal subsets of variables (corresponding to a sequence of edges) in such a path for which the ring bound capacity is violated. The following valid inequalities are violated by such a solution:*

$$\sum_{e=\{i,i+1\}:\ i\in\{i_1,\ldots,i_1+R\}} r_e \leq R, \qquad \forall i_1 \in \{v_1, v_2, \ldots, v_1+R, \ldots, v_m\} : i_1 + R \leq m-1 \tag{36}$$

where i *is selected as a rolling sequence basis.*

Proof. As the local rings are mutually exclusive, any (sub)path (a sequence of connected local ring edges) that has more than R local ring edges (which can appear, as the ring bound constraints are not present in the initial relaxation) must be broken down. Therefore, from every consecutive $R+1$ local ring edges, only R may be retained. □

Proposition 9. *The following inequalities are valid for the RSAP2 polytope.*

1. *Neither a local ring edge nor a tertiary edge can overlap with a spur arc (in either direction).*

$$y_e + t_{ij} + t_{ji} \leq 1, \qquad \forall e = \{i,j\} \in E, \tag{37}$$
$$r_e + t_{ij} + t_{ji} \leq 1, \qquad \forall e = \{i,j\} \in E. \tag{38}$$

2. *Both vertices of a local ring (tertiary ring) edge are connected to other local rings (tertiary ring) edges.*

$$r_e \leq \sum_{e' \in \delta(i) \setminus \{e\}} r_{e'}, \qquad \forall e = \{i,j\} \in E, \tag{39}$$
$$r_e \leq \sum_{e' \in \delta(j) \setminus \{e\}} r_{e'}, \qquad \forall e = \{i,j\} \in E, \tag{40}$$
$$y_e \leq \sum_{e' \in \delta(i) \setminus \{e\}} y_{e'}, \qquad \forall e = \{i,j\} \in E, \tag{41}$$
$$y_e \leq \sum_{e' \in \delta(j) \setminus \{e\}} y_{e'}, \qquad \forall e = \{i,j\} \in E. \tag{42}$$

3. *From a node i, either two local ring (tertiary ring) edges are encompassed, or a spur arc leaves.*

$$\frac{1}{2} \sum_{e \in \delta(i)} y_e + \sum_{j \neq i : \{i,j\} \in E} t_{ij} \leq 1, \qquad \forall i \in V, \tag{43}$$
$$\frac{1}{2} \sum_{e \in \delta(i)} r_e + \sum_{j \neq i : \{i,j\} \in E} t_{ij} \leq 1, \qquad \forall i \in V. \tag{44}$$

4. *No isolated triangle, whether exclusively in the space of y or exclusively in the space of r, can exist in any feasible solution. At least one local ring (tertiary) edge must be encompassed from the nodes of a triangle of tertiary (local ring) edges.*

$$\sum_{e \in \gamma(S)} r_e - \frac{1}{2} \sum_{e \in \delta(S)} y_e \leq 2, \qquad \forall S = \{i,j,k\} \subseteq V, \tag{45}$$
$$\sum_{e \in \gamma(S)} y_e - \frac{1}{2} \sum_{e \in \delta(S)} r_e \leq 2, \qquad \forall S = \{i,j,k\} \subseteq V, \tag{46}$$
$$\sum_{e \in \gamma(S)} r_e - \sum_{i \in S} y_{ii} \leq 2, \qquad \forall S = \{i,j,k\} \subseteq V. \tag{47}$$

5. *The following inequalities are valid for the polytope of RSAP2.*

$$r_e \leq t_{jj} - t_{ij}, \qquad \forall e = \{i,j\} \in E, \tag{48}$$
$$r_e \leq t_{ii} - t_{ji}, \qquad \forall e = \{i,j\} \in E, \tag{49}$$
$$y_e \leq t_{jj} - t_{ij}, \qquad \forall e = \{i,j\} \in E, \tag{50}$$
$$y_e \leq t_{ii} - t_{ji}, \qquad \forall e = \{i,j\} \in E. \tag{51}$$

Moreover, (48)-(49) *dominate* (5)-(6) *and* (50)-(51) *dominate* (7)-(8). *In addition,* (37)-(38) *are also dominated by* (48)-(51).

3.5. Cutting plane

Separation of valid inequalities (19). For a given fractional solution (t^*, y^*, r^*), we establish a support graph $G' = (V', E')$ where V' is composed of all nodes in the solution for which $y^*_{ii} > 0$ and all $e \in E$ for which $y_e > 0$ weighted by $y^*_e > 0$. We then use the max-flow algorithm of Edmonds and Karp (1972) between every pair of nodes, say i and j. If the cut size is less than $2(y^*_{ii} + y^*_{jj} - 1)$, we add a cut.

Separation of valid inequalities (20). We establish a support graph $G'' = (V'', E'')$ where V'' is composed of all nodes in the solution for which $y^*_{ii} > 0$ and E'' is composed of all $e \in E$ for which $y^*_e > 0$. We then identify the components in this graph and for every pair of components a and b we separate the corresponding cut. Although one can also separate such inequalities for the fractional solution with 0 min-cut, to avoid computational overheads and the separation of too many similar cuts (observed in initial tests), these constraints are added only in the case of integer solutions. The connected components on an undirected graph are identified using Depth-first search (DFS) approach and with the complexity $O(V''+E'')$.

Separation of valid inequalities (21). We establish a support graph $G^\dagger = (V^\dagger, E^\dagger)$ where $V^\dagger = V$ and all $e = \{i, j\} \in E$ for which $y^*_e + r^*_e + t^*_{ij} + t^*_{ji} > 0$ weighted by $\max\{y^*_e, r^*_e, (t^*_{ij} + t^*_{ji})\}$. We then use the max-flow algorithm of Edmonds and Karp (1972) between every pair of nodes, say i and j and identify S and $V^\dagger / S$. Wherever the max-flow is less than ϵ we add the cut for the given S and $V^\dagger / S$.

Separation of valid inequalities (22). For a solution (t^*, y^*, r^*), we establish a support graph $G^\ddagger = (V^\ddagger, E^\ddagger)$ where $V^\ddagger$ is composed of all $e \in E$ for which $r^*_e > 0$. We then enumerate all the cycles in such a graph using Tiernan's method (see Tiernan (1970)) and examine every cycle to identify the violated cuts. When enumerating all the cycles, one may also examine the violation of the inequalities (36).

Separation of valid inequalities (30). We can rewrite (30) as $(R-1)\sum_{e\in\delta(S)} r_e + 2R\sum_{i\in S} y_{ii} \geq 2\sum_{i\in S} t_{ii}$ or $(R-1)\sum_{e\in\delta(S)} r_e + 2R\sum_{i\in S} y_{ii} + 2\sum_{i\notin S} t_{ii} \geq 2\sum_{i\in V} t_{ii}$. These can now be separated in the same way as the SEC in the traveling salesman problem. We establish a support graph $G^\S = (V^\S, E^\S)$ where $V^\S = V \cup \{s, t\}$ where s and t are two dummy nodes and $E^\S$ is composed of the following edges: 1) all edges $e \in E$ with capacity $R-1$ for all $r_e > 0$, 2) the edges $\{s, i\}$, $\forall i \in V$ with capacity $2Ry_{ii}$, and 3) the edges $\{i, t\}$, $\forall i \in V$ with capacity $2t_{ii}$. A set S where $s \in S, t \notin S$ represents a set that can potentially give a violated valid inequality (30). The complexity of this separation is equivalent to the complexity of an s-t min-cut algorithm called on $G^\S$.

Separation of valid inequalities (31) *and* (32). While Fischetti et al. (1998a) and other researchers proposed several heuristic separation algorithms for such constraints, our initial computational experiments confirm that for this problem such techniques are not sufficiently successful in identifying the 2-matching structures (in particular in the tertiary level subgraph). Therefore, we use the method of Padberg and Rao (1982) for separating such valid inequalities.

Separation of valid inequalities (33) *and* (34). Letchford and Lodi (2002) proposed a polynomial-time algorithm for separating such inequalities. However, based on some initial computational experiments, for this problem and the set of our instances, block decomposition proves to be more efficient. Therefore, we opt to find combs based on block decomposition as used for the classical TSP implemented in Concorde (Applegate et al., 2007) (it can be downloaded at Cook's).

Separation of valid inequalities (36). At any integer node where an inequality of form (22) is violated, every sub-path of length $R+1$ represents such a valid inequality and is violated by 1. This separation can be done in $O(n^2)$.

3.6. Algorithm

The preprocessing is done before the branch-and-cut process starts. As the constraints (9)-(10) and (19)-(22) are part of the description of the polytope, they are separated each time a node LP is solved. The valid inequalities (31) and (32) are separated every $K1$ nodes, if fractional, and the valid inequalities (33) and (34) are separated at every $K2 > K1$ nodes, if fractional. We examine the violation of valid inequalities (30) each time a max-flow algorithm is invoked for separating other valid inequalities. Every integer node and every $K1$ nodes, we examine whether there are valid inequalities from among (37)-(51) that are violated.

Each time a feasible solution is found in the branch-and-bound tree, we try to improve it by searching within the proposed neighbourhoods, and update the primal bound.

4. Computational Results

The branch-and-cut is coded in C++ and run on a personal computer with an Intel Core i7 CPU at 3.4 GHz and 16 GB of RAM. We use CPLEX 12.6.3 as an MIP solver. The tolerance ϵ is set to 1e-6 and the max-flow problems are solved using the Boost implementation of Edmonds and Karp (1972) algorithm for directed graphs (with the undirected graphs being converted to directed ones in an appropriate way). In order to carry out our computational experiments, we use the instances from SNDlib (exactly those instances used in Carroll et al. (2013)). The parameter b is calculated in the same manner as in Carroll et al. (2013) and is such that establishing spur arcs is not encouraged. The b values for all SNDlib instances are outlined in Table 2.

Problem name	Size (m, n)	Penalty weight b
dfn-bwin	10,45	3
pdh	11,34	4
di-yuan	11,42	16
dfn-gwin	11,47	6
polska	12,18	3
atlanta	15,22	17
newyork	16,49	7
ta1	24,51	7
france	25,45	10
janos-us	26,42	4
norway	27,51	6
sun	27,51	15
cost266	37,57	13
giul39	39,86	6
pioro40	40,89	9
germany	50,88	7
ta2	65,108	20

Table 2: The parameter b used in computational experiments.

We use CPLEX as a tree manager and a modern branch-and-cut solver. We deactivate and limit the number of generated cuts to 0. In particular, we also avoid the Gomory cuts being generated, as our preliminary computational experiments showed that they reduce the performance of branch-and-bound by producing highly fractional nodes and hindering convergence. All presolving, primal reduction, dual reduction, etc. are deactivated, but the feasibility pump heuristic of CPLEX is used to find good quality solutions. The parameter $K1$ is set to 300 and $K2$ is set to 500.

In Table 3, we report our computational experiments. In the first column, 'B&B-Type' indicates whether the emphasis in CPLEX has been placed on proving optimality through moving the best bound value or on generating more feasible solutions during the solution process. The 'Instance' column reports the instance

name. In the third column, 'TimeRoot', we report the time spent on solving the LP relaxation, and this is followed by the LP objective value and the LP status, in the columns 'LPobj' and 'LPStatus', respectively. In the 'TimeIP' column, the time elapsed for solving the IP formulation is shown, and in the 'Nnodes' column, the number of processed nodes is reported. The columns 'Gap' and 'IPStatus' report the integrality gap and the CPLEX status upon termination. In the 'IPobj' column, the IP objective function upon termination is shown, and in 'bestObj' the best lower bound achieved by CPLEX is reported. Finally, in the last column, 'NUserCuts', we report the number of added user cuts. This number includes cuts of all the aforementioned types that are added during the branch-and-cut process.

Table 3: Branch-and-cut summary, small problems $R = 8$.

B&B-Type	Instance	TimeRoot	LPobj	LPStatus	TimeIP	Nnodes	Gap	IPStatus	IPobj	bestObj	NUserCuts
bb	dfn-bwin	0.01	105607.20	Optimal	0.17	16	0.00	Optimal	105809.60	105809.60	34
f	dfn-bwin	0.00	105607.20	Optimal	0.20	30	0.00	Optimal	105809.60	105809.60	34
bb	pdh	0.00	1261235.00	Optimal	0.17	15	0.00	Optimal	1355139.00	1355139.00	33
f	pdh	0.02	1261235.00	Optimal	0.13	23	0.00	Optimal	1355139.00	1355139.00	19
bb	dfn-gwin	0.01	15092.00	Optimal	0.14	3	0.00	Optimal	15724.00	15724.00	21
f	dfn-gwin	0.01	15092.00	Optimal	0.12	4	0.00	Optimal	15724.00	15724.00	18
bb	polska	0.02	3085.00	Optimal	0.14	6	0.00	Optimal	3487.00	3487.00	30
f	polska	0.00	3085.00	Optimal	0.39	24	0.00	Optimal	3487.00	3487.00	38
bb	atlanta	0.00	34184500.00	Optimal	0.13	3	0.00	Optimal	55452500.00	55452500.00	26
f	atlanta	0.01	34184500.00	Optimal	0.12	4	0.00	Optimal	55452500.00	55452500.00	27
bb	newyork	0.00	1345200.00	Optimal	1.95	874	0.00	Optimal	1512400.00	1512400.00	342
f	newyork	0.02	1345200.00	Optimal	3.04	1377	0.00	Optimal	1512400.00	1512400.00	558
bb	ta1	0.02	8166968.28	Optimal	2.45	451	0.00	Optimal	11410168.68	11410168.68	181
f	ta1	0.02	8166968.28	Optimal	3.26	757	0.00	Optimal	11410168.68	11410168.68	211
bb	france	0.01	14600.00	Optimal	0.44	7	0.00	Optimal	20800.00	20800.00	49
f	france	0.02	14600.00	Optimal	0.59	38	0.00	Optimal	20800.00	20800.00	81
bb	janos-us	0.01	12278.00	Optimal	10.25	1558	0.00	Optimal	16672.00	16672.00	398
f	janos-us	0.01	12278.00	Optimal	14.02	3429	0.00	Optimal	16672.00	16672.00	434
bb	norway	0.02	434575.00	Optimal	139.75	15446	0.01	OptimalTol	596070.00	596010.83	2365
f	norway	0.01	434575.00	Optimal	93.72	14275	0.00	Optimal	596070.00	596070.00	2345
bb	sun	0.01	434.57	Optimal	92.77	11071	0.01	OptimalTol	694.99	694.94	1780
f	sun	0.01	434.57	Optimal	76.92	12143	0.00	Optimal	694.99	694.99	1893
bb	cost266	0.01	7066350.00	Optimal	9.22	612	0.01	OptimalTol	12262590.00	12261378.21	246
f	cost266	0.03	7066350.00	Optimal	27.39	2141	0.00	Optimal	12262590.00	12262590.00	487
bb	di-yuan	0.02	370500.00	Optimal	0.16	0	0.00	Optimal	412300.00	412300.00	17
f	di-yuan	0.02	370500.00	Optimal	0.17	0	0.00	Optimal	412300.00	412300.00	17

In Table 3, except for 'Norway', which is solved in almost 93 seconds, all instances are solved in less than 30 seconds.

It is not trivial to perform a fair comparison between this model and the model in Carroll et al. (2013), in terms of computational behaviour. The main reason is that we do not use the same MIP solver (CPLEX vs. XpressMP, which have significant differences in their implementation and existing functionalities). Even in situations where we solve the problem with many few nodes (e.g. in 'ta1', 200 nodes using our approach and around 1,300 in theirs; in the case of 'France,' 49 vs. more than 1,000 nodes; or in 'cost266' with 250 vs. 2,700), it is rather hard to judge. However, our primary computational experiments show that when both codes run on the same hardware, this combination of model and the branch-and-cut approach we have devised is, in general, superior to the one in Carroll et al. (2013) with respect to both the computational time and the number of instances that are solved to optimality.

In Table 4, we present the optimal solutions for those instances that had not been solved to optimality prior to our work, and that we solve to optimality for the first time.

Table 4: Branch-and-cut summary, large problems, $R = 8$.

Instance	TimeRoot	LPobj	LPStatus	TimeIP	Nnodes	Gap	IPStatus	IPobj	bestObj	NUserCuts
germany	0.02	321942.50	Optimal	3017.57	70584	0.01	OptimalTol	549150.00	549096.00	10906
giul39	0.01	703.25	Optimal	14612.41	274544	0.00	Optimal	931.00	931.00	27485
pioro40	0.01	5890.50	Optimal	165585.47	2789260	0.00	Optimal	9159.00	9159.00	33112
ta2	0.03	24432722.12	Optimal	806.18	11785	0.01	OptimalTol	73782803.54	73776170.38	4199

In Carroll et al. (2013), within a time limit of 3 hours: for 'giul39', the authors report a feasible solution with an integrality gap of 10.65 per cent; for 'pioro40', a solution with a gap of 23.44 percent; for 'Germany', one with a gap of 0.51 per cent, while the process run into memory issues after around 8,000 seconds, and for 'ta2' a solution with a gap of '2.46'. However, as our proposed model is of smaller size, we can continue to run our branch-and-cut algorithm even beyond this time limit.

We obtain an optimal solution for 'ta2' in around 800 seconds, for 'Germany' in less than an hour of computation and for 'giul39' in around 4 hours. For 'pioro40', we reach a gap of around 14 per cent within the first 3 hours, and obtain an optimal solution in 46 hours (around two days of computation). The optimal solution for 'pioro40' is depicted in Figure 7.

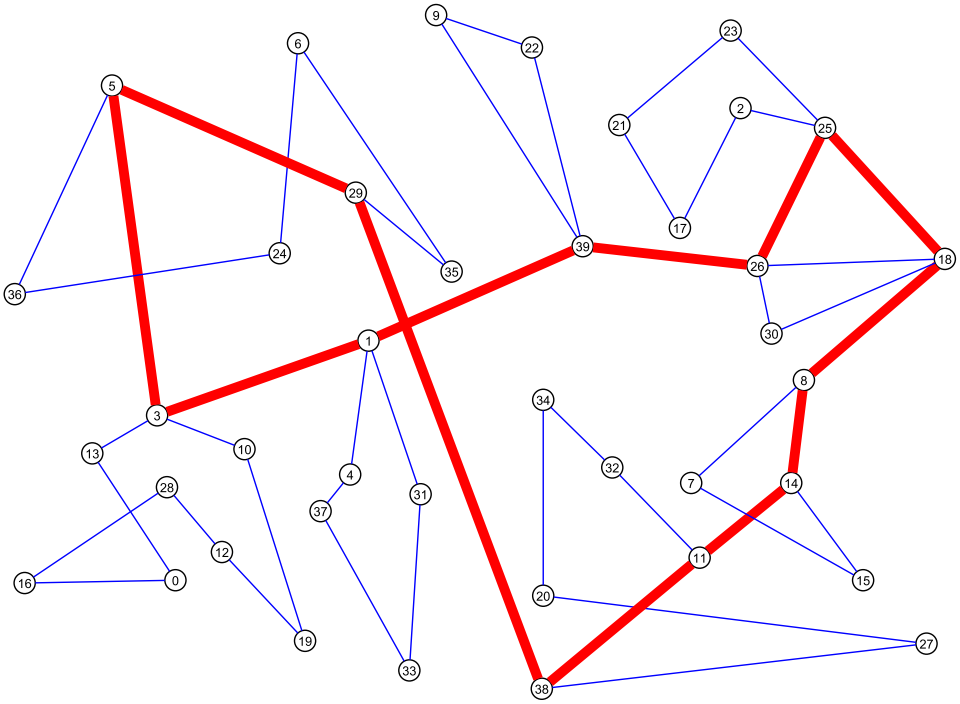


Figure 7: The optimal solution of 'pioro40' with objective value 9159.00.

In fact, we can say that the feasible solutions for 'ta2' and 'Germany' reported in Carroll et al. (2013) are indeed optimal solutions. However, for 'giul39', we find the optimal solution, which is different from the best-known feasible solution reported previously.

Table 5 reports the statistics on the number of cuts of each type being separated during the branch-and-cut process. This table is restricted to the previously unsolved instances. The first column represents the instance name, the second column, 'Poly', reports the number of cuts separated from among (37)-(51). The third one, 'Ghost', represents the total number of Ghost cuts and inequalities (36). The next columns, 'SEC_Y' and 'Comp_Y', report the number of cuts of (19)-(20). We report the number of separated inequalities (21) in the 'SEC_RY' column. The 2-matching constraints in the tertiary level and the secondary ring level are reported in '2M_Y' (i.e. (31)) and '2M_R' (i.e. (32)), respectively.

As no violated inequality from among (28), (30), (33) and (34) has been identified, we do not report them in this table.

It must be noted that sometimes the value in the column 'NUserCuts' does not equal the sum of valid inequalities from the different classes because CPLEX verifies whether a given cut is parallel to another one before adding it to the model. If it finds the cut to be parallel, its internal mechanism ignores the cut.

Table 5: Numerical result

CutType/Instance	Poly	Ghost	SEC_Y	SEC_RY	2M_Y	2M_R	Comp_Y
germany	175	2848	8228	38294	0	0	27
giul39	171	19372	22015	73843	83	63	11
pioro40	147	9040	812	0	462	606	993
ta2	202	628	2757	8338	0	2	4123

5. Conclusions

In this paper, we have proposed an integer programming model for the Ring Spur Assignment Problem (RSAP) that was introduced in Carroll et al. (2013). The original model uses $O(n^3)$ (about $n^3 + 4n^2$) binary variables while our new model uses only $O(n^2)$ (about $3n^2$) binary variables. The proposed model is composed of an exponential number of inequalities and requires a branch-and-cut algorithm to solve the instances. We have proposed some variable fixing, an incumbent polishing strategy within some neighbourhood structures, and some instance-dependent preprocessing and inequality deduction techniques. We have also identified several classes of valid inequalities for this problem, and proposed efficient separation routines to separate the ones that are violated. The computational results confirm that the proposed model can be solved efficiently using this branch-and-cut framework. For some instances with no known optimal solutions so far, our approach is capable of proving optimality. Further research directions include polyhedral analysis, such as identifying the dimension of polytope, proving the facetness of some of the valid inequalities, and tightening (identifying further) the valid inequalities using techniques such as lift-and-project etc.

Acknowledgement

The authors would like to give their warm thanks to Paula Carroll from University College of Dublin for sharing elements facilitating the comparison between the models.

References

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

Carroll, P., Fortz, B., Labbé, M., and McGarraghy, S. (2011). Improved formulations for the ring spur assignment problem. In *Network optimization*, pages 24–36. Springer Berlin Heidelberg.

Carroll, P., Fortz, B., Labbé, M., and McGarraghy, S. (2013). A branch-and-cut algorithm for the ring spur assignment problem. *Networks*, 61(2):89–103.

Carroll, P. and McGarraghy, S. (2013). A decomposition algorithm for the ring spur assignment problem. *International Transactions in Operational Research*, 20(1):119–139.

Cook's, W. (2006). ://www.math.uwaterloo.ca/tsp/. Accessed: 2017-04-25.

Edmonds, J. and Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2):248–264.

Fischetti, M., Glover, F., and Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, 104(1):91–104.

Fischetti, M., Gonzalez, J. J. S., and Toth, P. (1998a). Solving the orienteering problem through branch-and-cut. *INFORMS J. on Computing*, 10(2):133–148.

Fischetti, M., Salazar-González, J.-J., and Toth, P. (1998b). Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10:133–148.

Labbé, M., Rodríguez-Martin, I., and Salazar-Gonzalez, J. (2004). A branch-and-cut algorithm for the plant-cycle location problem. *Journal of the Operational Research Society*, 55(5):513–520.

Letchford, A. N. and Lodi, A. (2002). *Integer Programming and Combinatorial Optimization: 9th International IPCO Conference Cambridge, MA, USA, May 27–29, 2002 Proceedings*, chapter Polynomial-Time Separation of Simple Comb Inequalities, pages 93–108. Springer Berlin Heidelberg, Berlin, Heidelberg.

Orlowski, S., Pióro, M., Tomaszewski, A., and Wessäly, R. (2007). Sndlib 1.0 - survivable network design library. In *3rd International Network Optimization Conference INOC 2007, Spa, Belgium*.

Padberg, M. W. and Rao, M. R. (1982). Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7(1):67–80.

Rodríguez-Martín, I., Salazar-González, J. J., and Yaman, H. (2014). A branch-and-cut algorithm for the hub location and routing problem. *Computers & OR*, 50:161–174.

Sundar, K. and Rathinam, S. (2014). Multiple depot ring star problem: A polyhedral study and exact algorithm. *arXiv preprint arXiv:1407.5080*.

Tiernan, J. C. (1970). An efficient search algorithm to find the elementary circuits of a graph. *Commun. ACM*, 13(12):722–726.

Appendix A. Proofs of Propositions

Proof of Proposition 2

Proof. It is easy to verify that in the absence of tertiary nodes in the set S, the number of edges in the set S is bounded to:

$$\sum_{i\in S} t_{ii} - \lceil \frac{\sum_{i\in S} t_{ii}}{R} \rceil, \tag{A.1}$$

and therefore,

$$\sum_{i\in S} t_{ii} - \lceil \frac{\sum_{i\in S} t_{ii}}{R} \rceil \leq |S| - \lceil \frac{|S|}{R} \rceil. \tag{A.2}$$

For any additional tertiary node i inside the set at most one more edge is added. The proof is complete. □

Proof of Proposition 3

Proof. In every feasible solution of RSAP2 we have,

$$\sum_{e\in\gamma(S)} r_e - \sum_{i\in S} y_{ii} \leq \sum_{i\in S} t_{ii} - \lceil \frac{\sum_{i\in S} t_{ii}}{R} \rceil,$$

and,

$$\sum_{e\in\gamma(S)} r_e = \frac{1}{2} \sum_{i\in S} \sum_{e\in\delta(i)} r_e - \frac{1}{2} \sum_{e\in\delta(S)} r_e,$$

and from (27) and after some algebra we obtain:

$$\sum_{e\in\delta(S)} r_e \geq 2\left(\lceil \frac{\sum_{i\in S} t_{ii}}{R} \rceil - \sum_{i\in S} y_{ii} \right) \geq 2\left(\frac{\sum_{i\in S} t_{ii}}{R} - \sum_{i\in S} y_{ii} \right) \tag{A.3}$$

and the proof is complete. □

Proof of Proposition 5

Proof. In every feasible solution of RSAP2 we have,

$$2\sum_{e\in\gamma(H)} r_e + \sum_{e\in\delta(H)} r_e = \sum_{i\in H} \sum_{e\in\delta(i)} r_e$$

and from the constraints (27) one obtains:

$$\sum_{k\in H} \sum_{l\neq k} r_{kl} = 2\sum_{k\in H} t_{kk}.$$

Hence,

$$2\sum_{k\in H} t_{kk} = 2\sum_{e\in\gamma(H)} r_e + \sum_{e\in\delta(H)} r_e = 2\sum_{e\in\gamma(H)} r_e + \sum_{e\in\delta(H)} r_e \setminus \mathcal{T})) + r(\mathcal{T}) \tag{A.4}$$

given that $r(\mathcal{T}) \leq |\mathcal{T}|$ deduced from the bound constraints $x_e \leq 1$, we add this set of constraints to (A.4) and we obtain:

$$2\sum_{k\in H} t_{kk} + |\mathcal{T}| \geq 2\sum_{e\in\gamma(H)} r_e + \sum_{e\in\delta(H)\setminus\mathcal{T}} r_e + 2\sum_{e\in\mathcal{T}} r_e \geq 2\sum_{e\in\gamma(H)} r_e + 2\sum_{e\in\mathcal{T}} r_e \tag{A.5}$$

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

$$\sum_{e\in\gamma(H)} r_e + \sum_{e\in\mathcal{T}} r_e \leq \sum_{k\in H} t_{kk} + \lfloor \frac{|\mathcal{T}|}{2} \rfloor. \tag{A.6}$$

The proof of validity of (31) follows a similar way.

□

Proof of Proposition 6

Proof. For $S \subset V$, in every feasible solution of RSAP2 we have,

$$\sum_{e\in\gamma(H)} r_e = \sum_{i\in H} t_{ii} - \frac{\delta(S)}{2} \tag{A.7}$$

and for every $T_j \subset V,\ j \in \{1,\ldots,t\}$ we have,

$$\sum_{e\in\gamma(T_j)} r_e = \sum_{i\in T_j} t_{ii} - \frac{\delta(T_j)}{2} \tag{A.8}$$

and from (A.7) and (A.8), one yields:

$$\sum_{e\in\gamma(H)} r_e + \sum_i \sum_{e\in\gamma(T_i)} r_e = \sum_{i\in H} t_{ii} + \sum_j \sum_{i\in T_j} t_{ii} - \frac{1}{2}\left(\delta(H) + \sum_j \delta(T_j)\right). \tag{A.9}$$

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

It can be easily shown that $\delta_i(H) + \delta(T_i) \geq 3$. We also know that $\delta(H)$ and $\delta(T_i)$ are even and therefore, $\delta(H) + \sum_i \delta(T_i) \geq 3t + 1$. By substituting in (A.9) the proof is complete.

The proof of validity of (34) follows a similar way. □