ABSTRACT

The orienteering problem is a route optimization problem which consists in finding a simple cycle that maximizes the total collected profit subject to a maximum distance limitation. In the last few decades, the occurrence of this problem in real-life applications has boosted the development of many heuristic algorithms to solve it. However, during the same period, not much research has been devoted to the field of exact algorithms for the orienteering problem. The aim of this work is to develop an exact method which is able to obtain optimality certification in a wider set of instances than with previous methods, or to improve the lower and upper bounds in its disability.

We propose a revisited version of the branch-and-cut algorithm for the orienteering problem which includes new contributions in the separation algorithms of inequalities stemming from the cycle problem, in the separation loop, in the variables pricing, and in the calculation of the lower and upper bounds of the problem. Our proposal is compared to three state-of-the-art algorithms on 258 benchmark instances with up to 7397 nodes. The computational experiments show the relevance of the designed components where 18 new optima, 76 new best-known solutions and 85 new upper-bound values were obtained.

Keywords: orienteering problem · branch-and-cut · large problems

1 Introduction

The Orienteering Problem (OP) is a well-known routing problem proposed in the 80s, see [Tsiligirides, 1984] and [Golden et al., 1987]. Given a weighted complete graph with vertex profits and a constant \( d_0 \), the goal is to find the cycle which, with a length lower than or equal to \( d_0 \), maximizes the sum of profit of the visited vertices. In addition to the length constraint, every feasible cycle solution must visit a given depot vertex.

The OP can be seen as a combination of the Knapsack Problem (KP) and the Travelling Salesperson Problem (TSP). Given a set of items with an assigned weight and profit and a constant \( w_0 \), the goal in KP is to find the subset of items which, with a total weight lower than or equal to \( w_0 \), maximizes the sum of the profit of subset items. In the KP, the feasibility of a subset is checked in linear time. In the OP, however, the feasibility of a solution is checked by solving a TSP-decision problem. A subset of vertices is feasible if there exists a cycle (Hamiltonian in the subgraph obtained by the vertices) whose length does not exceed \( d_0 \), finding such a cycle is an NP-complete problem. This simple but non-trivial combination of two NP-hard problems makes the OP an interesting problem to study.
The occurrence of the OP in many real-life applications, such as logistics and tourism, has boosted the emergence of many variants and algorithms to solve the problem over the last decades. A survey of OP variants, approaches, and applications can be seen in [Vansteenwegen and Gunawan, 2019]. In this paper, we have focused on solving the classical problem through an exact algorithm.

Some exact algorithms have been proposed for the OP, see [Laporte and Martello, 1990], [Ramesh et al., 1992], [Leifer and Rosenwein, 1993], and [Gendreau et al., 1998]. The most competitive approach thus far was proposed by [Fischetti et al., 1998] two decades ago. To our knowledge, no exact algorithm for the classical OP has been published afterwards. The first exact algorithm, a Branch-and-Bound (B&B) algorithm, was published in [Laporte and Martello, 1990] where bounds for the problem were provided based on the Knapsack relaxation of the OP. In [Ramesh et al., 1992], new bounds for the B&B were obtained by Lagrangian relaxation. In [Leifer and Rosenwein, 1994] a Branch-and-Cut (B&C) algorithm was proposed, which included logical, connectivity, and cover cuts for the first time. In [Gendreau et al., 1998] a B&C was proposed for a variant of the OP which considers multiple depot nodes. The B&C approach in [Fischetti et al., 1998] outperformed the previous ones in middle-sized OP instances by considering column generation, new valid inequalities (cycle cover and path inequalities), problem-specific separation algorithms, and an efficient primal heuristic. In the last two decades, authors dealing with exact approaches have focused on solving variants of the problem, such as Team OP ([Boussier et al., 2007], [Poggi et al., 2010], [Dang et al., 2013], [Keshikaran et al., 2015], [Bianchessi et al., 2018]), Arc OP ([Archetti et al., 2016]), Team Arc OP ([Archetti et al., 2014], [Riera-Ledesma and Salazar-Gonzalez, 2017]) and Probabilistic OP ([Angelelli et al., 2017]).

Recent results for large-sized instances of the OP presented in [Kobeaga et al., 2018] and [Santini, 2019], have shown that the state-of-the-art B&C algorithm in [Fischetti et al., 1998] does not obtain satisfactory results when the number of nodes is larger than 400. In half of the large-sized instances, the named B&C algorithm does not produce any output. In many of the other half of instances, the returned solution value is far from the values obtained by the heuristic algorithms. The motivation of this work is to develop a B&C algorithm which is able to improve the values of the best-known lower and upper bounds in the literature, and if possible, to obtain optimality certifications in a wider set of instances than the state-of-the-art B&C algorithm. In our view, there is room for improvement for B&C algorithms in the case of OP, mainly if we consider that some of the successful techniques developed for the TSP, such as shrinking and efficient pricing, have not yet been adapted for the OP. This paper is an attempt to combine the adaptation of some of those techniques with our OP specific contributions.

In this work, we have developed and adapted techniques to scale the B&C algorithm to large OP problems. Our main contributions in this paper are the following:

- Develop a new joint separation algorithm for Subcycle Elimination Constraints and Connectivity Constraints, which efficiently uses the shrinking techniques in [Kobeaga et al., 2020] to speed up the algorithm by reducing the adverse effects of the shrinking for Connectivity Constraints (Section 4.2.1).
- Design blossom separation algorithms for Cycle Problems, which generalize the [Padberg and Hong, 1980] and [Grötschel and Holland, 1991] heuristics, providing a considerable improvement in the solution quality and running time (Section 4.2.2).
- Design an efficient variable pricing procedure for the OP inspired by the one developed in [Applegate et al., 2007] for the TSP. It enables repetitive calculations to be avoided and the exact calculation of the reduced cost of some variables to be skipped (Section 4.3).
- Conceive a three-level separation loop for the OP, which treats the cuts according to their relevance while giving the same chance to the cuts with complementary incidence. This significantly reduces the running time of the B&C algorithm (Section 4.4).
- Devise a combination of two alternative primal heuristics, a greedy one ([Fischetti et al., 1998]) in the separation loop and a metaheuristic based one ([Kobeaga et al., 2018]) at the beginning of branch nodes. The new combination boosts the quality of the obtained solutions in large-sized problems (Section 4.5).
- Formulate the computation of the global upper bound in the branching phase for the OP, which enables the upper bound obtained in the branching root node to be updated (Section 4.6).

The computational experiments presented in this paper show the importance of the proposed techniques and components for the B&C algorithm for the OP. We compare our B&C algorithm with three state-of-the-art algorithms, two heuristics ([Kobeaga et al., 2018] and [Santini, 2019]) and one exact ([Fischetti et al., 1998]), on 258 benchmark instances with up to 7397 nodes. In comparison to the literature, the revisited B&C obtains:

- 180 optimum values, from which 18 are new.
We denote by $R$ where the objective function (2a) is to maximize the total collected profit. The constraint (2b) limits the total cycle length. The degree equations (2c), together with the logical constraints (2e) and the integrality constraints (2i), ensure

The OP goal is to determine a simple cycle that maximizes the sum of the scores of the visited vertices, such that it

The rest of the paper is organized as follows. In Section 2 the 0-1 Integer Linear model of the OP is introduced. In Section 3 we present the valid inequalities for the OP. In Section 4 we detail the proposed B&C algorithm for the problem. In Section 5 the results of the computational experiments are shown. The detailed experimental results can be found in the appendices.

2 OP Modelling and Polyhedral Considerations

The OP can be defined by a 5-tuple $G = \langle V, d, s, 1, d_0 \rangle$, where $G = K_n = \langle V, E \rangle$ is a complete graph with vertex set $V$ and edge set $E$; $d = \langle d_e \rangle$ where $d_e$ is the positive distance value (time or weight) associated with each edge of $G$; $s = \langle s_v \rangle$, where $s_v$ is a positive value that represents the score (profit) of vertex $v \in V$; $1 \in V$ is a vertex selected as the depot; and $d_0$ is a positive value that limits the cycle length.

Let us define the following sets:

\[
\begin{align*}
(Q : W) &:= \{[u, v] \in E : u \in Q, v \in W\} &Q, W &\subset V \quad (1a) \\
\delta(Q) &:= (Q : V - Q) &Q &\subset V \quad (1b) \\
E(Q) &:= (Q : Q) &Q &\subset V \quad (1c) \\
V(T) &:= \{v \in V : T \cap \{v : v \neq 0\}\} &T &\subset E \quad (1d)
\end{align*}
\]

where $(Q : W)$ are the edges connecting $Q$ and $W$, $\delta(Q)$ is the set of edges in the coboundary of $Q$ also known as the star-set of $Q$, $E(Q)$ is the set of edges between the vertices of $Q$, and $V(T)$ is the set of vertices incident with an edge set $T$. For simplicity, we sometimes denote $\{e\}$ and $\{v\}$ by $e$ and $v$, respectively, e.g., $\delta(v)$ and $V(e)$.

We denote by $R^V$, and $R^E$, the space of real vectors whose components are indexed by elements of $V$, and $E$ respectively. In the model of OP, two types of decision variables are used, $y = (y_v) \in R^V$ and $x = (x_e) \in R^E$, associated with the nodes and edges of $G$, respectively, where:

\[
y_v = \begin{cases} 
1 & \text{if node } v \text{ is visited} \\
0 & \text{otherwise}
\end{cases} \quad x_e = \begin{cases} 
1 & \text{if edge } e \text{ is traversed} \\
0 & \text{otherwise}
\end{cases}
\]

For $(y, x) \in R^{V \times E}$, $S \subset V$ and $T \subset E$, we define $y(S) = \sum_{v \in S} y_v$ and $x(T) = \sum_{e \in T} x_e$.

The OP goal is to determine a simple cycle that maximizes the sum of the scores of the visited vertices, such that it contains the depot node $1 \in V$ and whose length is equal to or lower than the distance limitation, $d_0$. Then, the OP can be formulated as the following 0-1 Integer Linear model:

\[
\begin{align*}
\max & \quad \sum_{v \in V} s_v y_v \quad (2a) \\
\text{s.t.} & \quad \sum_{e \in E} d_e x_e \leq d_0, \quad (2b) \\
& \quad x(\delta(v)) - 2y_v = 0, \quad v \in V, \quad (2c) \\
& \quad x(\delta(H)) - 2y_l - 2y_r \geq -2, \quad l \in H \subset V, r \in V - H, \quad (2d) \\
& \quad 3 \leq |H| \leq |V| - 3, \quad (2e) \\
& \quad y_v - x_e \geq 0, \quad v \in V, e \in \delta(v), \quad (2f) \\
& \quad 0 \leq y_v \leq 1, \quad v \in V, \quad (2g) \\
& \quad 0 \leq x_e \leq 1, \quad e \in E, \quad (2h) \\
& \quad y_1 = 1, \quad (2i) \\
& \quad x_e \in Z \quad e \in E \quad (2j)
\end{align*}
\]

where the objective function (2a) is to maximize the total collected profit. The constraint (2b) limits the total cycle length. The degree equations (2c), together with the logical constraints (2e) and the integrality constraints (2j), ensure
that the visited vertices have exactly two incident edges and the unvisited vertices none. The Subcycle Elimination Constraints (SEC) \((2d)\) ensure that only one connected cycle exists. Throughout the paper, we use the notation \(\{H, l, r\}\) for the SEC defined by the set \(H \subset V\) and the vertices \(l \in H\) and \(r \notin H\). The constraints \((2g)\) and \((2i)\) impose that the edge variables are 0-1, consequently, considering these together with the Logical Constraints \((2e)\) and the bounds \((2g)\), the vertex variables are also 0-1. The constraint \((2h)\) defines the depot condition.

As mentioned in the introduction, the OP can be seen as a combination of the TSP-decision and the KP problems. Particularly, the OP is a Cycle Problem (CP) where the solutions, which are cycles, need to satisfy a certain length constraint. This relation with the two classical optimization problems is useful when identifying the valid inequalities and their respective separation algorithms for OP. Let us show how the solution space of OP is related to those well-known problems. The OP Polytope \((P_{OP})\) of the complete graph \(K_n\) is defined by:

\[
P_{OP} := \text{conv}\{(y, x) \in \mathbb{R}^V \times E : (y, x) \text{ satisfies } (2h), (2g), (2d), (2e), (2g), (2h), (2i)\}
\]  

(3)

The Knapsack Polytope \((P_{KP})\), see [Balas, 1975], is a well-studied polytope closely related to the \(P_{OP}\):

\[
P_{KP} := \text{conv}\{x \in \mathbb{R}^E : x \text{ satisfies } (2b), (2g), (2i)\}
\]  

(4)

Since the solutions of the OP are cycles, the Cycle Polytope \((P_{CP})\), plays a crucial role when solving the OP with B&C. Based on [Bauer, 1997], the \(P_{CP}\) can be characterized as:

\[
P_{CP} := \text{conv}\{(y, x) \in \mathbb{R}^V \times E : (y, x) \text{ satisfies } (2c), (2d), x(E) \geq 3, (2f), (2g), (2i)\}
\]  

(5)

We have the following relationship:

\[
P_{OP} \subset P_{CP} \cap (\mathbb{R}^V \times P_{KP}) \cap \{(y, x) \in \mathbb{R}^V \times E : y_1 = 1\}
\]  

(6)

Consequently, the potential valid inequalities for the OP are those which are valid for \(P_{CP}\) and the \(P_{KP}\). However, the \(P_{OP}\) and the intersected polytopes in the relationship \((6)\) are not equal and alternative valid inequalities are needed to deal with the OP. An example of a point in \(P_{CP} \cap (\mathbb{R}^V \times P_{KP}) \cap \{(y, x) : y_1 = 1\}\) but not in \(P_{OP}\) is given in Figure 2 of [Fischetti et al., 1998].

3 Valid Inequalities

In this section, we present valid inequalities for the OP. The straightforward inequalities, as motivated in Section[2] are based on the \(P_{KP}\) (Edge Cover inequalities) and \(P_{CP}\) (Comb inequalities) relaxations of the \(P_{OP}\) and they were mainly proposed in [Fischetti et al., 1998] and [Gendreau et al., 1998]. Additional valid inequalities to those based on \(P_{KP}\) and \(P_{CP}\) have also been proposed in the literature: the Connectivity Constraints in [Leifer and Rosenwein, 1994], the Vertex Cover inequalities in [Gendreau et al., 1998], and the Cycle Cover and the Path inequalities in [Fischetti et al., 1998]. The novelty of this section is an alternative representation of comb inequalities, which is then used for the efficient pricing in Section[4.3].

3.1 Connectivity Constraints

The Connectivity Constraints (CC) are well-known inequalities for the OP, e.g. [Gendreau et al., 1998] and [Leifer and Rosenwein, 1994], and are a particular case of the conditional cuts proposed in [Fischetti et al., 1998]. The CCs exploit the depot constraint \((2h)\). Given a lower bound, LB, of the OP, let \(T\) be a subset of nodes such that \(1 \in T\), \(|T| \geq 2\) and \(\sum_{v \in T} s_v \leq LB\). The inequality defined by \(T\)

\[
x(\delta(T)) \geq 2
\]  

(7)

is valid for the OP. Since \(x(\delta(T)) = x(\delta(V - T))\), the inequality can also be defined for \(T \subset V\) such that \(1 \notin T\) and \(\sum_{v \notin T} s_v \leq LB\). So, it is always possible to assume that \(|T| \leq |V|/2\).

3.2 Comb Inequalities

The comb inequalities were generalized from the TSP to cycle problems in [Bauer, 1997]. A comb is a tuple \(\langle H, \{T_1, \ldots, T_t\}, L, R\rangle\) of three vertex subsets and a family \(T = \{T_1, \ldots, T_t\}\) of vertex subsets such that satisfies the following properties:

i) \(t \geq 3\) and an odd integer
The goal of these cuts is to exclude the paths that due to the length constraint

The maximum length constraint

\[ T_i \cap H \neq \emptyset \text{ and } T_j - H \neq \emptyset \text{ for } i = 1, \ldots, t \]

\[ L = \{ l_i \} \text{ such that } l_i \in T_i \cap H \text{ for } i = 1, \ldots, t \]

\[ R = \{ r_i \} \text{ such that } r_i \in T_i - H \text{ for } i = 1, \ldots, t \]

The set \( H \) is called the handle, the sets in \( T \) are called the teeth, the set \( R \) is called the Root set, and \( L \) is called the Link set. Then, the inequality

\[ x(\delta(H)) + \sum_{j=1}^{t} x(\delta(T_j)) - 2y(R) - 2y(L) \geq 1 - t \]

is facet-defining for \( P_{CP} \), as was shown in [Bauer, 1997], and therefore, a valid inequality for \( OP \). When all the teeth consist of exactly two vertices, the comb inequalities are known as blossom inequalities.

### 3.3 Edge Cover Inequalities

The maximum length constraint [2], which is a capacity constraint for the edge variables, defines a \( KP \) polytope, as explained in Section [2]. For every feasible \((y, x)\), the edge variable, \( x \), belongs to \( P_{KP} \). For the \( OP \), the Edge Cover inequalities are the cover inequalities of the associated \( P_{KP} \) ([Balas, 1975]). These inequalities were first introduced for the \( OP \) in [Leifer and Rosenwein, 1994] and also used in [Fischetti et al., 1998] and [Gendreau et al., 1998]. Let \( F \subset E \) be a subset with \( \sum_{e \in F} d_e > d_0 \), then:

\[ x(F) \leq |F| - 1 \]

defines an Edge Cover inequality for the \( OP \). We assume that \( F \) is a minimal cover, i.e. for every \( F_0 \subseteq F \), we have \( \sum_{e \in F_0} d_e \leq d_0 \).

### 3.4 Cycle Cover Inequalities

Every feasible cycle \( F \subset E \) satisfies the equation \( x(F) = y(V(F)) \). Let \( F \subset E \) be a subset that defines a cycle with \( \sum_{e \in F} d_e > d_0 \), then the inequality

\[ x(F) \leq y(V(F)) - 1 \]

is valid for the \( OP \). These cuts were used in [Fischetti et al., 1998] and [Gendreau et al., 1998].

### 3.5 Vertex Cover Inequalities

Let UB be an upper bound of the \( OP \) and \( Q \subset V \) be a subset with \( \sum_{v \in Q} s_v > UB \), then:

\[ y(Q) \leq |Q| - 1 \]

defines a Vertex Cover inequality for the \( OP \). We assume that \( S \) is a minimal cover. These inequalities were first used for the \( OP \) in [Gendreau et al., 1998].

### 3.6 Path Inequalities

The goal of these cuts is to exclude the paths that due to the length constraint [2] cannot be part of a feasible solution. Let \( P = \{ [i_1, i_2], [i_2, i_3], \ldots, [i_{k-1}, i_k] \} \) be any simple path through \( V(P) = \{ i_1, \ldots, i_k \} \subset V - \{ 1 \} \), and define the vertex set:

\[ W(P) := \{ v \in V - V(P) : d_{1, i_1} + \sum_{e \in P} d_e + d_{i_k, v} + d_{v, 1} \leq d_0 \} \]

Then the following Path inequality

\[ x(P) - y(V(P)) + y_1 + y_k - \sum_{v \in W(P)} x_{i_k, v} \leq 0 \]

is valid for the \( OP \), see [Fischetti et al., 1998].
4 Branch-and-Cut Algorithm

In this section, we present the principal contributions of this paper. These contributions deal with the separation algorithms of inequalities stemming from the cycle problem (SECs and comb inequalities), the design of the separation loop, the pricing of variables for the column generation and the calculation of the lower and upper bounds of the problem. In Figure 1 a flowchart representing a simplified B&C algorithm can be consulted.

Figure 1: Flowchart of the Branch-and-Cut algorithm considered in this work. BRANCH is an oracle which returns an unevaluated node in the branching tree. SEP refers to the separation algorithms. At each action box of the flowchart the subproblem $LP_0$ is updated and solved.

4.1 Initialization

First of all, we obtain an initial heuristic solution. To that aim, we make use of the EA4OP metaheuristic in [Kobeaga et al., 2018] considering a small size population.

Next, we build the initial subproblem, $LP_0$. Given the computational requirements of considering all the variables and constraints that define the OP, an initial subproblem $LP_0$ is built. The $LP_0$ is initialized considering the following subset of constraints and variables:

i) All the vertex variables.

ii) Edges in the 10 nearest neighborhood graph.

iii) Maximum length constraint (2b), degree constraints (2c), and depot constraint (2h).

iv) Variable bounds, (2f) and (2g).
Immediately after the initialization, the edge variables are priced, see Section 4.3. In the rest of the paper, we use the LP0 symbol to refer to any subproblem of the OP, regardless of whether it is the initial one or not.

4.2 Separation Algorithms

In this section, we present the heuristic and exact separation algorithms used to find the violated inequalities. Our contributions are concentrated in the separation algorithms for SECS, CCs and blossom inequalities. Hence, we only give details of these separation algorithms in the section. The details of separation algorithms for the rest of the inequalities (Logical Constraints, Edge Cover, Vertex Cover, Cycle Cover, and Path inequalities) can be found in [Fischetti et al., 1998].

Let \((y^*, x^*)\) be a solution of a particular LP0 problem and define \(V^* = \{ v \in V : y_v^* > 0 \}\) and \(E^* = \{ e \in E : x_e^* > 0 \}\). Then, \(G^* = (V^*, E^*)\) is called the support graph associated with the solution \((y^*, x^*)\).

4.2.1 SECS and CCs

Violated SECs \((14d)\) and CCs \((7)\) are found using a common separation algorithm. This is natural since, in both constraint families, the incidence vector of the arcs, \(x\), in the inequality can be written as the star-set value, \(x(\delta(Q))\) of a subset \(Q\) of vertices. Since \(\delta(Q)\) is the cut associated with \(Q\), the separations of both inequalities are closely related to the minimum cut problem. In [Kobeaga et al., 2020] it was shown that the shrinking techniques substantially speed up the SEC separation algorithms. However, as explained below, the shrinking might also have a negative impact on the finding of violated CCs. In this section, we study how to efficiently use the shrinking to speed up the joint separation algorithm by reducing the adverse effects for CCs.

Given a solution \((y^*, x^*)\) and a subset \(Q\), the subset \(Q\) could generate at the same time a violated SEC and a violated CC for \((y^*, x^*)\). Since the CCs do not depend on the value of the vertices, while the SECs do, the CCs tend to be more violated and more stable, i.e., remain active in subsequent updates of the LP0, than the SECs. Therefore, we treat the CCs with a higher priority.

Although SECS are part of the OP model, in order to control the size of the working LP0, they are included only when required. This strategy is reasonable since there exist polynomial exact separation algorithms for SECS. In contrast, the separation problem for CCs is not known to be polynomial, and it can be modeled as follows:

\[
\begin{align*}
\min & \quad 2 \sum_{v \in V^*} y_v^* z_v - 2 \sum_{v \in V^*} x_{(v,u)}^* z_v z_u \\
\text{s.t.} & \quad \sum_{v \in S} s_v z_v \leq LB \\
& \quad z_1 = 1 \\
& \quad z_v \in \{0, 1\} \quad \forall v \in V
\end{align*}
\]

where \(z = (z_v)\) are binary variables whose values are \(z_v = 1\) if the node \(v\) is selected and 0 otherwise. The problem \((14)\) is a Quadratic Knapsack Problem (QKP) with a fixed variable. Consequently, there exists a violated CC for \((y^*, x^*)\) if and only if the optimal solution of Problem \((14)\) has a value lower than 2. Taking into consideration that repeatedly solving QKPs during the B&C is not viable, the CCs are not separated exactly, but in a heuristic manner take advantage of the SEC separation algorithm. The well-known approaches for the separation of SECS in the TSP, the connected component heuristic and Hong’s approach can be extended to jointly separate the SECS and CCs:

**Connected components heuristic.** The straightforward heuristic to find violated SECS and CCs is to search for the connected components of \(G^*\) using the depth-first-search algorithm. When a connected component contains the depot vertex \(1\) and the sum of the vertices scores in the component is lower than \(LB\), we record the associated CC of the component, otherwise, we record the associated SECS.

**Extended Hong’s approach.** There are two main strategies to exactly separate SEC inequalities in cycle problems, which are extensions of Hong’s approach and the Padberg-Grötschel approach (also known as the Gomory-Hu tree-based approach) for the TSP, see [Kobeaga et al., 2020]. In both approaches, the separation is carried out by solving a sequence of \(|V^*| - 1\) \((s, t)\)-minimum cut problems. On the one hand, in the extended Hong’s approach, the vertex with a higher \(y^*\) value (the depot vertex \(1\)) is fixed to be the source, \(s\), and the sink vertices, \(t\), are chosen from the set \(V^* - \{1\}\). On the other hand, the extended Padberg-Grötschel approach is based on the so-called Gomory-Hu tree (directed and rooted in \(1\)), which is constructed by solving \(|V^*| - 1\) \((s, t)\)-minimum cut problems.
As mentioned above, and as already proposed in the literature, the SEC separation strategies are leveraged to find violated CCs as well. Although the extended Padberg-Grötschel approach obtains a larger number of violated SECs, it is not appropriate to find violated CCs, since the obtained sets do not contain the depot vertex \(1\). Contrarily, the extended Hong’s approach for SECs can be easily adapted to additionally find violated CCs. It can be achieved, by solving at each step of the separation algorithm the \((1, v)\)-minimum cut (useful to find violated SECs) and \((v, 1)\)-minimum cut (useful to find violated CCs) problems. For these reasons, we use the extended Hong’s approach as the base strategy for the joint separation algorithm.

The running time of these SEC separation algorithms can be improved using the shrinking techniques for cycle problems, as was seen in [Kobega et al., 2020]. In this publication, three general shrinking rules (C1, C2, and C3) and three SEC specific shrinking rules (S1, S2, and S3) for cycle problems were presented. However, although the shrinking is a key strategy for efficiently separating the SECs, it might be unfavorable for the separation of CCs. The point is that when the vertices are contracted and grouped, the chance to obtain the subset of vertices with a score sum lower than \(LB\) decreases, consequently, some violated CCs might vanish. Note that, the mentioned shrinking techniques are safe for valid inequalities of the cycle polytope and CCs are not. Therefore, since CCs are important cuts for OP, shrinking might have a negative impact on the performance of the overall B&C algorithm for the OP. One contribution in this paper is to propose strategies to minimize the possible disadvantages of the shrinking (which is important to speed up the separation) in the joint separation algorithm for SECs and CCs.

Following this, not all the shrinking strategies for cycle problems described in [Kobega et al., 2020] are adequate for the OP problem. Particularly, we exclude the S2 shrinking rule (which leads to excessively aggressive shrinking strategies and hence to vanish violated CCs in some cases) and only consider the shrinking strategies C1C2 and S1 in the preprocess of the joint separation algorithm. Once entered in the separation algorithm, the shrinking rule S3, which contracts the sink and target of the solved minimum cut, contributes positively to separating both families of constraints since it enables a wider family of subset candidates to be obtained. Hence, the S3 rule is used in combination with the C1C2 and S1 shrinking strategies in the separation algorithm. After the S3 rule is applied, we search for new shrinkable sets using the selected shrinking strategy.

Classically, the candidate subsets for SECs and CCs are obtained by the minimum cut algorithm. However, considering the importance of CCs, we intensify the search for extra candidate subsets for CCs, which is made more efficient by taking advantage of the vertex clustering obtained by the shrinking. We propose new strategies based on the following lemma:

**Lemma 4.1.** Let \((y, x)\) be a vector that satisfies the degree constraints. If \(U\) and \(W\) are subsets of \(V\) such that \(W \subset U\), the following inequality is satisfied:

\[
x(\delta(U - W)) \leq x(\delta(U)) + x(\delta(W))
\]

_Proof._ When \((y, x)\) satisfies the degree constraints, the identity \(x(\delta(T)) = 2y(T) - 2x(E(T))\) is valid for every \(T \subset V\). Replacing the respective expressions in the inequality (15) we obtain:

\[
2y(U - W) - 2x(E(U - W)) \leq 2y(U) - 2x(E(U)) + 2y(W) - 2x(E(W))
\]

Considering the hypothesis \(W \subset U\), we have \(y(U - W) = y(U) - y(W)\).

\[
x(E(U)) - x(E(U - W)) \leq 2y(W) - x(E(W))
\]

Also, if \(Q \subset S\), the equality \(E(S - Q) = E(S) - E(Q) - \delta(Q) \cap E(S)\) holds.

\[
x(E(U)) - x(E(U)) + x(E(W)) + x(\delta(W) \cap E(U)) \leq 2y(W) - x(E(W))
\]

\[
x(\delta(W) \cap E(U)) \leq 2y(W) - 2x(E(W))
\]

This last inequality is satisfied due to \(\delta(Q) \cap E(S) \subset \delta(Q)\), which proves the lemma.

We use the following notation for shrinking. Let \(G = (V, E)\) be the graph and \((\bar{x}, \bar{y})\) the vector obtained by applying a shrinking strategy to \(G^\ast\) and \((y^\ast, x^\ast)\), respectively, and \(\pi : P(V) \to P(V)\) the unshrinking function. Let \(S\) be the subset obtained by the \((\bar{v}, \bar{1})\)-minimum cut (where \(1\) is the contracted vertex that contains the depot vertex \(1\)), so \(1 \in \pi(S)\), and suppose that \(x(\delta(S)) < 2\). Note that, \(x(\delta(S)) = \bar{x}(\delta(S))\), where \(S = \pi(S)\). If \(\sum_{v \in S} s_v \leq LB\), the subset \(S\) defines a violated CC. Otherwise, after each \((\bar{v}, \bar{1})\)-minimum cut problem is solved, and in the case that \(x(\delta(S)) < 2\), we test the following strategies to find candidate subsets for CCs:

i) First, when \(|\pi(1)| > 2\), we check if \(y_1 < 1\) and \(\sum_{v \in \pi(1)} s_v \leq LB\). If this is the case, the subset \(Q = \pi(1)\) defines a violated CC.
We generalize this heuristic for the general cycle problems by applying the Padberg-Hong algorithm by levels. A faster heuristic to find the handle candidates can be designed by omitting some connected components of the Extended Grötschel-Holland heuristic (EGH) TSP, which is known as the odd-component heuristic. In this heuristic for the TSP, the violated blossoms are found by the Extended Padberg-Hong heuristic (EPH). In this heuristic, the search is guided by the greedy algorithm of Kruskal for the Minimum Spanning Tree. At each iteration of the Kruskal algorithm, a new edge is inserted into the tree, and the connected component containing the edge is chosen as a candidate handle. In this work, we consider two alternative approaches to finding candidate handles: the Extended Padberg-Hong heuristic and the Extended Grötschel-Holland heuristic.

**Extended Padberg-Hong heuristic (EPH).** [Padberg and Hong, 1980] proposed a blossom separation heuristic for the TSP, which is known as the odd-component heuristic. In this heuristic for the TSP, the violated blossoms are found by restricting the set of candidate handles to the connected components of the fractional graph \( G^*_1 = (V^*_1, E^*_1) \), where \( E^*_1 = \{ e \in E^* : 0 < x^*_e < 1 \} \) and \( V^*_1 = V(E^*_1) \).

We generalize this heuristic for the general cycle problems by applying the Padberg-Hong algorithm by levels. A level \( \lambda \) is defined by each different value of the set \( \{ y^*_v \}_v \). We call \( L \) the set of distinct levels. Note that, the number of levels, \( |L| \), is bounded by \( |V| \). Associated with a level \( \lambda \) we have the level graph \( G^*_\lambda = (V^*_\lambda, E^*_\lambda) \), where \( E^*_\lambda = \{ e \in E^* : 0 < x^*_e < \lambda \} \) and \( V^*_\lambda = V(E^*_\lambda) \).

A faster heuristic to find the handle candidates can be designed by omitting some connected components of \( G^*_\lambda \). At every level \( \lambda \), we discard the connected components, \( C^*_\lambda \), such that \( y^*_V \neq \lambda \) for all \( v \in C^*_\lambda \). Now, we identify the connected component of vertices with \( y^*_v = \lambda \). So, in total, we search for \( |V^*| \) different connected components of, in the worst case, \( G^*_1 \).

Once we have identified an initial list of candidate handles, the next step is to find the associated teeth for these handles. Let \( H \) be a candidate handle, and define the set of teeth as \( T_H = \{ e \in \delta(H) : x^*_e \geq \lambda \} \). Recall that the teeth of blossoms are edges. Not all the teeth families obtained using this strategy satisfy the comb (blossom) definition. If two teeth overlap in \( v \in H \), then these two teeth are removed from the family of teeth \( T_H \) and the handle is updated as \( H = H \cup \{ v \} \). If, eventually, the list of teeth \( T_H \) consists of an odd number of at least three disjoint teeth, \( (H, T_H, L, R) \) forms a blossom inequality where \( L_1 = T^+_H \cap H \) and \( R_1 = T^+_H - H \). If there is just one tooth i.e., \( T_H = \{ T \} \), we test if \( H \) defines a violated CC. In the case that it does not, then \( H \) alone defines a violated SEC.

**Extended Grötschel-Holland heuristic (EGH).** Another fast heuristic for the TSP was proposed in [Grötschel and Holland, 1991] whose aim was to minimize the influence of small perturbations of \( x^* \) in the separation algorithm. We have adapted this heuristic for the OP using the strategy of levels mentioned above. In this approach, the handles are considered as the vertex sets of the connected components of the graph \( G^*_{\lambda, \epsilon} = (V^*, E^*_{\lambda, \epsilon}) \) where

\[
E^*_{\lambda, \epsilon} = \{ e \in E^* : \epsilon \leq x^*_e \leq (1 - \epsilon)\lambda \}
\]

for a small \( \epsilon, 0 < \epsilon < 1 \). Let \( H \) denote the vertex set of such a component, a candidate handle, and let \( e_1, \ldots, e_t \) be the edges in the set

\[
T_H = \{ e \in \delta(H) \subset E^* : x^*_e > (1 - \epsilon)\lambda \}
\]

in the non-increasing order of \( x^*_e \). If \( t \) is even, then append to \( T_H \) the edge with the highest \( x^*_e \) in

\[
\{ e \in \delta(H) \subset E^* : x^*_e < \epsilon \}
\]

If the edges intersect, the strategy outlined above is followed to obtain a handle \( H \) and a teeth family \( T_H \) that satisfies the blossom definition.

In Figure 2, we illustrate the EPH blossom heuristic for cycle problems. In Figure 2(a) the given support graph is presented, where there are three distinct levels, \( L = \{ 1, 1/2, 1/4 \} \). In Figure 2(b) the candidate handles are presented. Three candidate handles are obtained in level 1: \{1, 2, 3\}, \{5, 6, 7\} and \{10, 11, 12, 13, 14, 15, 16\}. Two
candidate handles are obtained in level 1/2: \{10, 11, 12\} and \{14, 15, 16\}. There are no candidate handles obtained in level 1/4. Next, we check for violated cuts. The star-set of \{10, 11, 12, 13, 14, 15, 16\} is formed by two non-overlapping edges, so it is excluded. The candidates \{5, 6, 7\} and \{10, 11, 12\} define violated blossoms, e.g., \langle\langle10, 11, 12\rangle, \{8, 10\}, \{9, 11\}, \{12, 13\}\rangle, L, R\rangle where \(L = \{10, 11, 12\}\) and \(R = \{8, 9, 13\}\) shown in Figure 2.c). The candidates \{1, 2, 3\} and \{14, 15, 16\} define violated SECs, e.g. \langle\{1, 2, 3\}, 1, 4\rangle and \langle\{14, 15, 16\}, 14, 1\rangle, but first for \{1, 2, 3\} it should be checked whether it defines a violated CC.

### 4.3 Column Generation

During the B&C algorithm, only a subset of edges is included in the working LP\(_0\). At certain points of the algorithm, we need to price the excluded edge variables, and add to the LP\(_0\): 1) to guarantee that the working relaxation is an upper bound of the problem or branched subproblem and 2) to recover, whenever it is possible, a feasible LP\(_0\) after feasibility breaking cuts have been added to the LP\(_0\). Taking into account that usually only a small subset of variables is included in the LP\(_0\), and that the excluded variables could participate in multiple cuts of the LP\(_0\), the pricing phase could constitute a bottleneck of the B&C algorithm. In this section, we develop a technique, inspired by that used in [Applegate et al., 2007], which enables us to avoid repetitive calculations and to skip the exact calculation of the reduced cost of some variables.

Let us call \(L^V\) the family of SECs \(2c\), CC \(7\), and comb \(8\) cuts. In these cuts, the edge variables with non-negative coefficients can be represented as the sum star-set of subsets of vertices. Complementarily, let us call \(L^E\) the family of Logical \(2c\), Edge Cover \(9\), Cycle Cover \(10\) and Path \(13\) cuts. Note that the Vertex Cover \(11\) inequalities do not contribute to the reduced cost of the edge variables. So, in the OP, the reduced cost of an edge variable, \(e = [v, w]\), can be calculated by:

\[
rc_e = -d_e \pi_{d_0} - \pi_v - \pi_w + rc^V_e + rc^E_e
\]

where \(\pi_{d_0}\) is the dual variable of the maximum length constraint \(2b\), \(\pi_v\) and \(\pi_w\) are the dual variables of the degree constraints \(2c\) of \(v\) and \(w\) respectively, and \(rc^V_e\) and \(rc^E_e\) are the contributions made by the cuts in \(L^V\) and \(L^E\), respectively. We will see that the \(rc^E_e\) values can be obtained in linear time in terms of \(|V|\) and \(|L^E|\), and we will reproduce the pricing strategy used in [Applegate et al., 2007] to calculate the \(rc^V_e\) values.

It can be seen that the cost of the calculation of all the \(rc_e\) is \(O(|L^E||V|)\). To that aim, it is sufficient to check that the number of edges with a non-negative coefficient in each cut of \(L^E\) is bounded by \(|V|\). In the case of Logical, Cycle Cover, and Path inequalities, it is derived from the definition of the valid inequality. For Edge Cover inequalities, this bound is obtained in Lemma 4.2.

**Lemma 4.2.** Let \(T \subset E\) denote a subset defining a violated cover inequality. If the degree equations \(2c\) are satisfied by \((y, x) \in \mathbb{R}^V \times E\) then \(|T| \leq |V|\).
We can exclude for exactly pricing the edges that have a negative reduced cost, \(rc_e\), of the reduced costs \(rc_e\), to exclude for exactly pricing the edges that have a non-positive upper bound \(\hat{r}c_e\). For each subset, \(S \in S\), let us call \(\pi_S\) the dual of the subset \(S\) defined as:

\[
\pi_S = \sum_{j=1}^{r} \chi_j(S)\pi_j
\]

where \(\chi_j(S) = 1\) if \(S \in F_j\) and 0 otherwise, and \(\pi_j\) is the dual variable associated with the cut \(j\). Then, the contribution of the cuts in \(L^V\) in the reduced cost of an edge \(e\) can be written as:

\[
rc_e^V = \sum_{S \in F \cap V(e) \neq \emptyset} \pi_S
\]

where \(\pi_S\) is the dual of a subset \(S\). Since, for the edge \(e = [v, w]\), each \(S\) must contain either \(v\) or \(w\), an upper bound, \(\hat{r}c_e\), of \(rc_e^V\) can be obtained by:

\[
\hat{r}c_e^V = \sum_{S \in F, v \in S} \pi_S + \sum_{S \in F, w \in S} \pi_S
\]

which satisfies \(rc_e^V \leq \hat{r}c_e^V\). Therefore, we have the desired upper bound:

\[
\hat{r}c_e = -d_e \pi_{n+1} - \pi_v - \pi_w + rc_e^E + \hat{r}c_e^V
\]

Note that, each edge appears at most twice in a comb inequality, so the calculation of all the \(\hat{r}c_e^V\) has a \(O(M|L^V||V|)\) time complexity where \(M\) is the maximum number of subsets involved in a cut. Therefore, the calculation of all the \(\hat{r}c_e\) has a \(O(M|L^V||V|)\) time complexity. In our B&G, the value of \(M\) is related to the number of teeth in the combs. To ensure a true linear time complexity procedure, one could limit the number of teeth in the combs. However, in practice, the number of teeth tends to be small and it can be assumed that \(M << |V|\).

We can exclude for exactly pricing the edges that \(r\hat{c}_e \leq 0\). For those edges that \(r\hat{c}_e > 0\), the exact reduced cost, \(rc_e\), can be calculated by using the upper bound value:

\[
rc_e = \hat{r}c_e - 2 \sum_{S \in F, V(e) \subseteq S} \pi_S
\]

The pricing loop is done in batches. In the first step, a fixed number of \(\hat{r}c_e\) are calculated, the first batch of variables and those with positive values are preselected. In the next step, for those preselected variables, we calculate the exact reduced cost, \(rc_e\), and add to the LP0 the edges whose value is positive. Then, the LP0 is updated. Next, we select
the second batch of variables and we repeat the procedure. When the pricing aims to obtain the upper bound of the branched subproblem, we exit the pricing loop when a whole round of evaluation is performed without introducing a variable to the LP_0. When the pricing aims to recover a feasible LP_0, we exit the pricing loop once a feasible LP_0 is obtained without the need to price all the excluded variables.

### 4.4 Separation Loop

The separation loop to find the violated cuts is accomplished in three subloops. In the inner loop, we consider the separation of logical constraints and the connected components heuristic for SECs and CCs. In the middle loop, we consider the separations of cuts which are related to the cycle essence of the OP, i.e., SECs, CCs, blossoms, and Cycle Cover cuts. In the outer loop, we consider the rest of the cuts, i.e., the Edge Cover, Vertex Cover, and the Path algorithm. When we enter in any of the loops, the first step is to execute the lower level subloops. Then, we start with the first algorithm on the list. If no violated cuts are found we move on to the next algorithm. If violated cuts are found, we first add the cuts and optimize the LP_0. Then, we search for a feasible solution using a primal heuristic and update the LB if needed. We add the associated CC of the heuristic solution if it is violated and then we price the variables. When a new CC cut or a priced edge has been added to the LP_0, the inner loop is repeated. Otherwise, we return to the middle loop.

The middle and outer loops only differ in the considered constraint families. In the middle loop, we consider the separation algorithms in the following order: the extended Padberg-Hong algorithm for blossom, the extended Grötschel-Holland algorithm for blossom, the joint SEC/CC separation algorithm, and Cycle Cover separation algorithm. In the outer loop, we consider the Edge Cover algorithm, the Vertex Cover algorithm, and the Path algorithm. When we enter in any of the loops, the first step is to execute the lower level subloops. Then, we start with the first algorithm on the list. If no violated cuts are found we move on to the next algorithm. If violated cuts are found, we first add the cuts and optimize the LP_0. Then, we search for a feasible solution using a primal heuristic and update the LB if needed. We add the associated CC of the heuristic solution if it is violated and then we price the variables. At this point, we move to the lower level loop and continue with the next separation in the list.

In the separation loop, after adding the violated cuts found in a separation algorithm, we check if any edge variable or constraint can be removed from the LP_0. We remove an edge variable from the LP_0 if, during a number of consecutive evaluations, its associated value, _x_e^∗_, has been zero. We remove a constraint from the LP_0 if during a number of consecutive evaluations its slack has been higher than zero.

### 4.5 Primal Heuristics and Lower Bounds

We use two primal heuristics to obtain feasible solutions from a fractional solution (y^∗, _x^∗_). In the first heuristic, we obtain a single solution, by using the _x^∗_ values related to edges, inspired by the heuristic proposed in [Fischetti et al., 1998]. In the second heuristic, first, we build a population of cycles and then evolve it using the EA4OP metaheuristic, see [Kobeaga et al., 2018]. The cycles in the population are constructed by selecting first the subset of vertices in each cycle using the y^∗ values.

**Path Building primal heuristic (PB).** The PB heuristic was presented in [Fischetti et al., 1998]. First, the edges _e_ ∈ _E^∗_ are sorted in decreasing order of _x_e^∗_, and the ties are randomly broken. The procedure starts with an empty path _T_ = ∅. At each step we select an edge _e_ ∈ _E^∗_ whose _x_e^∗_ has the largest value from the set of edges which have not been considered yet. If the inclusion of _e_ in _T_ does not lead to a vertex with a degree larger than 2, then _T_ = _T_ ∪ { _e_ } otherwise we exclude _e_ and repeat the process. The path building heuristic finishes when the inclusion of _e_ leads to _T_ being a cycle or when there are no edges left to check. If the depot vertex is not in one of the paths in _T_, it is included as a single point path. If _T_ consists of multiple paths, we extend it to a cycle by randomly connecting the extreme vertices (in the original paper the paths were joined using the nearest neighbor heuristic). Since this primal heuristic is fast, it is used in the separation loop.

**Vertex Picking primal heuristic (VP) with the EA4OP metaheuristic.** In the VP heuristic, we first select a collection of vertices in _V^∗_ and then build a random cycle through the selected vertices. Each vertex _v_ is selected according to a Bernoulli distribution with parameter _y_v^∗_. By applying multiple times the VP strategy to obtain feasible solutions
from \((y^*, x^*)\), we build a small population. Then, as explained below, we ensure that the solutions in the population are feasible and improve when it is possible. Once we have a population with feasible solutions, it is evolved using the EA4OP metaheuristic proposed in [Kobeaga et al., 2018]. The EA4OP with VP heuristic is used to find feasible solutions after an edge is branched, as shown in Figure 1.

For solutions obtained by PB and VP heuristics, we improve the route lengths using the Lin-Kernighan heuristic for the TSP, and then first check if it satisfies the constraint (\(2b\)). If it does not, we apply the drop operator which consists in...
deleting vertices from the solution until the cycle satisfies the length constraint. Then we try to improve the solution by the k-d tree based vertex inclusion procedure as explained in [Kobeaga et al., 2018].

4.6 Branching and Upper Bounds

The branching is carried out in a classical way following a depth-first-search, where the edges are branched first to 1 and then to 0. In order to select the edge variable to branch, we use the classical branching strategy: the edge \( e \), with the fractional value closest to 0.5, i.e., the edge that minimizes \( |x^*_e - 0.5| \).

The global upper bound and branch node upper bound are calculated just before pruning a branch. The branch node upper bound, \( UB_N \), is used to verify the pruning, i.e., that \( LB \geq UB_N \). The global upper bound is calculated with two aims: firstly, to use it in Vertex Cover separation, and secondly, to compute the optimality gap when the algorithm finishes due to time limitations.

The global upper bound, \( UB^G \) of OP, is obtained using the dual solution \( \pi^* \) of the solution \((y^*, x^*)\) of the LP:

\[
UB^G = \sum_{i=1}^{c} \pi^*_i b_i + \sum_{v \in V - \{1\}} \sum_{r_{c^*_v}>0} rc^*_v + \sum_{e \in E} rc^*_e
\]  

(23)

where the reduced costs \( rc^*_v \) and \( rc^*_e \) are calculated using the dual variables \( \pi^* \) and \( c \) is the number of constraints.

The upper bound of a branch node, \( UB_N \), can be calculated by subtracting the contributions of the branched edges to \( UB^G \). Let \( B_0, B_1 \subset E \) be the subset of edges branched to 0 and 1, respectively. Then, we obtain \( UB_N \) by:

\[
UB_N = \sum_{i=1}^{c} \pi^*_i b_i + \sum_{v \in V - \{1\}} \sum_{r_{c^*_v}>0} rc^*_v + \sum_{e \in E} rc^*_e - \sum_{e \in B_0} rc^*_e - \sum_{e \in B_1} rc^*_e
\]  

(24)

5 Computational Experiments

In this section, we present the results of the computational experiments. Firstly, we evaluate the new designed components for the revisited B&C algorithm (RB&C); and secondly, we compare the performance of RB&C with state-of-the-art B&C and heuristic algorithms. The software used for the experiments is publicly available on [github.com/bcamath-ds/OPLib](http://github.com/bcamath-ds/OPLib).

The experiments are carried out using well-known instances in the literature. These instances, which are based on the TSPLIB library, were first proposed in [Fischetti et al., 1998] and then extended to larger problems in [Kobeaga et al., 2018]. The instances are split into two groups: medium-sized instances (up to 400 nodes) and large-sized instances (up to 7397 nodes). In total, we consider 258 benchmark instances. They are also classified into three generations (Gen1, Gen2 and Gen3) according to the definition of scores, see [Fischetti et al., 1998]. For all of these three generations, the distance limitation is set as half of the TSP solution value. These instances are publicly available at [github.com/gkobeaga/op-solver](http://github.com/gkobeaga/op-solver).

In order to measure the performance of the algorithms, we compare the quality of the returned best solutions (LB) and the mean running time (in seconds) of the algorithms. In addition, in the case of the B&C algorithms, we also compare the obtained upper bounds (UB). All the experiments for the compared algorithms have been carried out using a 5-hour time limit.

In Table 1, we detail the values of the common parameters for all the simulations of the RB&C algorithm. They were chosen inspired by the parameters used in [Applegate et al., 2007] and our preliminary experiments for the OP.

5.1 Evaluation of Components

In this section, we evaluate the designed components for the RB&C algorithm in Section 4. We have carried out experiments with several alternative configurations of the components. To that aim, a subset of 15 OP instances were selected: 5 TSP instances (pr76, att532, vm1084, r11323 and vm1748, inspired by the subset selected in [Goldberg and Tsiontsios, 2001]) with their respective score generations proposed in [Fischetti et al., 1998]. Then, for each instance and generation, we have executed the different B&C configurations 5 times.
Table 1: Common parameters.

| Parameter         | Value | Description                                      |
|-------------------|-------|--------------------------------------------------|
| ZERO              | $10^{-7}$ | Sensibility of fractional numbers               |
| ADD_CUT_BATCH     | 250   | Maximum number of cuts added to the $LP_0$ at once |
| ADD_MIN_VIOL      | $10^{-6}$ | Minimum violation of a cut to include it in the $LP_0$ |
| SUBLOOP_IMPR      | 1%    | Minimum improvement to repeat the subloops      |
| ADD_SEC_PER_SET   | 50    | Amount of SECs considered for each subset        |
| ADD_PATH_MAX      | 500   | Maximum cuts for Path inequalities separation    |
| ADD_EGH_EPSILON   | 0.3   | Epsilon value for the EGH blossom heuristic      |
| PRICE_MAX_ADD     | 200   | Maximum number of variables added to the $LP_0$  |
| PRICE_RC_THRESH   | $10^{-5}$ | Minimum penalty of a variable to add to the $LP_0$ |
| DEL_DUST_VAR      | $10^{-3}$ | Minimum $y$ value to consider an edge as active |
| DEL_DUST_CUT      | $10^{-3}$ | Maximum slack value to consider a cut as active |
| DEL_MAX_AGE_CUT   | 5     | Consecutive inactivity to delete a cut from the $LP_0$ |
| DEL_MAX_AGE_VAR   | 100   | Consecutive inactivity to delete an edge from the $LP_0$ |
| XHEUR_GREEDY_XMIN | 0.3   | Use arcs larger than this value in PB primal heuristic |
| XHEUR_EA4OP_POP_SIZE | 10  | Population size for EA4OP                        |
| XHEUR_EA4OP_D2D   | 5     | Iterations before checking feasibility in EA4OP  |
| XHEUR_EA4OP_NPAR  | 3     | Number of parents preselected in EA4OP           |

In order to evaluate our contributions, we have chosen a reference configuration, REFERENCE, that incorporates the components proposed in Section 4 and compared it with its alternative configurations. The reference RB&C algorithm considers the following components:

- **SEC/CC separation algorithm** (Section 4.2.1):
  i) SRK=S1S3: Uses shrinking rules S1 and S3.
  ii) CC STRATS: Uses strategies to find extra violated CCs.
- **Blossom separation algorithms** (Section 4.2.2):
  i) EPH BLOSSOM: Uses Extended Padberg-Hong blossom heuristic.
  ii) EGH BLOSSOM: Uses Extended Grötschel-Holland blossom heuristic.
- **Separation algorithms from the literature**:
  i) CYCLE: Uses Cycle Cover inequalities.
  ii) EDGE: Uses Edge Cover inequalities.
  iii) PATH: Uses Path inequalities.
- **Separation Loop strategy**:
  i) SEP=THREE SUBLOOPS: Uses the separation loop strategy presented in Section 4.4
- **Primal heuristics** (Section 4.5):
  i) BRANCH XHEUR=VP + EA4OP: Constructs a small population using VP heuristic and evolves it with EA4OP.
  ii) SEP XHEUR=PB: Constructs a single solution using PB in the separation loop.

The alternative configurations are obtained by modifying a single component in REFERENCE, while the rest of the components remain untouched. These changes to REFERENCE are made by deleting a component(-), adding a new component(+) or replacing a component (COMP=). The tested alternative strategies are the following:

- **SEC/CC separation algorithm**:
  i) -SRK: Does not use any shrinking technique. As a consequence, CC STRATS are not used either.
  ii) SRK=C1C2S3: The shrinking rule S1 is replaced with the rules C1C2.
  iii) -CC STRATS: Does not use strategies to find extra violated CCs.
- **Blossom separation algorithms**: 

15
With respect to the rest of the separation algorithms proposed in the literature for the OP, we include in the reference all the algorithms that do not use the shrinking technique in terms of time, but indeed, the obtained LB values are also worse. In addition, the experiments show that the alternatives decrease the performance of the reference configuration for the RB&C algorithm either in terms of solution quality, upper bound value, or running time. The results grouped by instances are presented in Appendix A.

Table 2: Results of the alternative configurations for RB&C. In bold, the values of the alternatives that are worse than those obtained by the reference configuration.

| Strategy                  | Gen1          |         |         |         | Gen2          |         |         |         | Gen3          |         |         |
|---------------------------|---------------|---------|---------|---------|---------------|---------|---------|---------|---------------|---------|---------|
|                           | LB  | UB  | Time   | LB  | UB  | Time   | LB  | UB  | Time   | LB  | UB  | Time |
| reference                 | 0.05 | 0.00 | 262.06 | 0.05 | 0.04 | 23.11  | 0.02 | 0.01 | 44.02  |
| -SRK                      | 0.13 | 0.09 | 532.37 | 0.10 | 0.04 | 25.86  | 0.02 | 0.02 | 134.74 |
| SRK=C1C2S3                | 0.02 | 0.00 | 88.32  | 0.09 | 0.04 | 31.72  | 0.01 | 0.01 | 79.81  |
| -CC STRATS                | 0.02 | 0.00 | 115.91 | 0.04 | 0.01 | 21.85  | 0.01 | 0.01 | 449.90 |
| -EPH BLOSSOM              | 0.09 | 0.15 | 208.65 | 0.12 | 0.15 | 33.04  | 0.10 | 0.22 | 199.79 |
| -EGH BLOSSOM              | 0.02 | 0.00 | 290.71 | 0.04 | 0.04 | 26.18  | 0.03 | 0.01 | 91.83  |
| +FST BLOSSOM              | 0.00 | 0.00 | 345.32 | 0.04 | 0.00 | 26.43  | 0.04 | 0.00 | 66.54  |
| -CYCLE COVER              | 0.11 | 0.00 | 137.73 | 0.13 | 0.04 | 30.04  | 0.05 | 0.01 | 35.50  |
| -PATH                     | 0.06 | 0.00 | 124.79 | 0.02 | 0.04 | 25.60  | 0.03 | 0.01 | 48.18  |
| +VERTEX COVER             | 0.05 | 0.00 | 61.10  | 0.03 | 0.03 | 22.33  | 0.03 | 0.01 | 104.82 |
| SEP=TWO SUBLOOPS          | 0.05 | 0.00 | 315.34 | 0.06 | 0.04 | 17.05  | 0.03 | 0.01 | 164.44 |
| BRANCH XHEUR=PB           | 0.08 | 0.00 | 179.14 | 0.12 | 0.01 | 2.37   | 0.04 | 0.01 | 62.74  |
| BRANCH XHEUR=VP-EA4OP     | 0.12 | 0.00 | 222.40 | 0.07 | 0.04 | 7.17   | 0.01 | 0.01 | 158.65 |

The results show that the alternatives decrease the performance of the reference configuration for the RB&C algorithm either in terms of solution quality, upper bound value, or running time. The experiments restate the importance of the shrinking techniques for the SEC/CC separation algorithm, as can be seen in the results for -SRK. It is not only worse not using the shrinking in terms of time, but indeed, the obtained LB values are also worse. In addition, the results suggest that the S1 shrinking technique, which is considered in the reference, might be preferable to the C1C2 technique. Regarding the CC STRATS, the results for Gen3 suggest that not considering the strategies to find extra violated CCs might have a negative impact on the running time of the algorithm.

Next, looking at the separation algorithms for blossoms, the results show that the EPH heuristic is crucial in the RB&C, particularly, if we focus on the obtained LB and UB values. From the table, we can also extract that the EGH heuristic improves the running time of the B&C algorithm. Alternatively, although the FST blossom heuristic might improve the quality of the solutions, it reports worse running times.

With respect to the rest of the separation algorithms proposed in the literature for the OP, we include in the reference all but Vertex Cover inequalities. This way, the RB&C uses the same families of cuts as in [Fischetti et al., 1998], which enables us to evaluate the contributions in this paper in a better way.

Finally, the experiments show that the VP primal heuristic plays an important role in obtaining better LB values, particularly for large problems, as can be seen in the detailed results in Appendix A. However, solving the VP primal
heuristic in the branch node is more costly than PB primal heuristic, hence the running time of the RB&C is worsened in the smallest instances. Similarly, by using the EA4OP to improve the results by VP heuristic, the obtained LB values are improved in large problems at the expense of worsening the running time in the smallest instances.

5.2 Comparison with state-of-the-art Algorithms

The proposed reference RB&C has been compared with the state-of-the-art B&C algorithm in [Fischetti et al., 1998] (FST) and two state-of-the-art heuristics, [Kobeaga et al., 2018] (EA4OP) and [Santini, 2019] (ALNS). The detailed results can be found in Appendix B.

Three notes before moving on to the discussion. First, the FST code reports the running times using one trailing digit while the rest of the algorithms report the times using two trailing digits. In order to make use of the reported times in the literature of the FST, we round the obtained times by the RB&C to one trailing digit when we compare it with the FST algorithm. Secondly, the FST returns a false optimum for pa561 in Gen1. We assume that this is a consequence of the rounding sensibility and we accept as valid the rest of the reported optima by FST. Thirdly, eight instances (rat99, rat195, tsp225, pa561, rat575, rat783, nrw1379, and fnl4461) of Gen3 have been excluded for the comparison of the RB&C with the EA4OP and the ALNS, due to an issue in the generation of scores of the instances used by those algorithms. Since the results of the current comparison are clear enough, we have discarded rerunning the experiments with the updated scores.

In Table 3 it can be seen that the RB&C algorithm is able to obtain the best-known solutions value in all the medium-sized instances. Moving on to large-sized instances, the superiority of the RB&C algorithm compared to the FST approach becomes evident. While the FST algorithm fails to output a solution in almost half of the instances (mainly because of running out of memory), the RB&C algorithm returns a solution for every instance. Moreover, it obtains the best-known solution in significantly more instances than FST (245 against 170) and UB (249 against 173) values. Even more, it obtains more optimality certifications (180 against 165).

In Table 4 it can be seen that the RB&C algorithm is able to obtain the best-known solutions value in all the medium-sized instances. Moving on to large-sized instances, the superiority of the RB&C algorithm compared to the FST approach becomes evident. While the FST algorithm fails to output a solution in almost half of the instances (mainly because of running out of memory), the RB&C algorithm returns a solution for every instance. Moreover, it obtains the best-known solution in significantly more instances than FST (245 against 170) and UB (249 against 173) values. Even more, it obtains more optimality certifications (180 against 165).
In Table 4, we compare the quality of the solutions and running times, restricted to those instances in which FST actually returns a solution. We particularly focus on the number of solutions (optimality certifications, best-known solutions and upper bounds) that are new in the literature, i.e., values not obtained by the rest of the algorithms. Thus, for the lower-bound values, we also take into account the results obtained by the EA4OP and ALNS heuristics. Additionally, we show the number of instances in which the considered B&C algorithms are faster than the competitor. When we restrict the considered instances to the instances where the FST obtains a feasible solution, the RB&C outperforms the results of the FST. While the FST obtains 1 new best-known solution (not obtained by any other algorithm) and 9 new UB values, the RB&C obtains 31 LB and 34 UB new values. In the same set of instances, the FST obtains 3 optimality certifications that the RB&C is not able to obtain, while the RB&C obtains 7 optimality certifications that the FST is unable to obtain. Moreover, it turns out that the RB&C is faster than the FST in 100 instances while the FST is faster than the RB&C in 63 instances.

Next, we compare the RB&C algorithm against state-of-the-art algorithms in terms of solution quality, running time, and Pareto efficiency. In Table 5 and Table 6, the algorithms are compared pairwise and instance-by-instance for medium-sized and large-sized instances respectively. The aim is to measure the number of instances where an algorithm is simultaneously as least as fast as the opponent and obtains a better quality solution.

Table 5 shows that the RB&C algorithm is competitive in medium-sized instances. Compared to the ALNS heuristic and FST algorithm, it obtains better Pareto efficiency results in the three generations. Comparing it to EA4OP, the Pareto efficiency is lower because the heuristic is a faster algorithm. Nevertheless, the RB&C obtains much better quality solutions.

Table 6 shows that RB&C is the best performing algorithm in large-sized instances. Particularly, it behaves better than the FST algorithm, obtaining the best quality and time solutions in most of the instances, hence obtaining better Pareto results. The ALNS algorithm is able to return some solutions with better quality or running time, however, overall, the RB&C performs better in large-sized instances. The EA4OP metaheuristic is faster than the B&C but, in general, obtains worse quality solutions.

Finally, in Table 7, we summarize the new best-known results obtained in the experiments. The RB&C algorithm obtains 18 new optimality certifications, 76 new best-known solution values and 85 new upper-bound values.

6 Conclusions and Future Work

We have presented a revisited version of the B&C algorithm for the OP that brings multiple contributions together. We have proposed a joint separation algorithm for SECs and CCs, which efficiently uses the shrinking technique for cycle problems by reducing the adverse effects of the shrinking for CCs. We have developed two blossom heuristics for cycle problems which generalize the well-known approaches in the literature of the TSP. We have designed an efficient variable pricing procedure for the OP which enables us to avoid repetitive calculations and to skip the exact calculation
Table 6: Comparison in large-sized instances against state-of-the-art algorithms in terms of quality, time and Pareto efficiency.

|        | Gen1 | Gen2 | Gen3 |
|--------|------|------|------|
|        | EA4OP| tie  | RB&C | EA4OP| tie  | RB&C | EA4OP| tie  | RB&C |
| Quality| 1    | 0    | 40   | 5    | 0    | 36   | 3    | 0    | 33   |
| Time   | 39   | 0    | 2    | 40   | 1    | 0    | 35   | 1    | 0    |
| Pareto | 1    | 0    | 2    | 5    | 0    | 1    | 3    | 0    | 1    |

|        | ALNS | tie  | RB&C | ALNS | tie  | RB&C | ALNS | tie  | RB&C |
|--------|------|------|------|------|------|------|------|------|------|
| Quality| 2    | 2    | 37   | 4    | 1    | 36   | 4    | 0    | 32   |
| Time   | 6    | 11   | 24   | 13   | 25   | 3    | 13   | 19   | 4    |
| Pareto | 4    | 0    | 34   | 5    | 0    | 24   | 4    | 0    | 20   |

|        | FST  | tie  | RB&C | FST  | tie  | RB&C | FST  | tie  | RB&C |
|--------|------|------|------|------|------|------|------|------|------|
| Quality| 0    | 13   | 28   | 0    | 9    | 32   | 3    | 11   | 27   |
| Time   | 1    | 5    | 35   | 8    | 13   | 20   | 5    | 17   | 19   |
| Pareto | 1    | 1    | 39   | 8    | 0    | 33   | 7    | 2    | 32   |

Table 7: New best-known optimum, lower bound and upper bound values.

|        | OPT | LB | UB |
|--------|-----|----|----|
| Gen1   | 12  | 25 | 28 |
| Gen2   | 2   | 27 | 28 |
| Gen3   | 4   | 24 | 29 |
| All    | 18  | 76 | 85 |

of the reduced cost of some variables. We have proposed a separation loop for the OP that takes into consideration the different contributions and separation costs of the valid inequalities. We have used alternative primal heuristics, one of which is based on a metaheuristic, and a mechanism to update the global upper bound during the branching phase to tighten the lower and upper bounds for the cases when the algorithm finishes without an optimality certification.

The experiments have shown that the RB&C algorithm for OP is a more efficient approach than the state-of-the-art B&C algorithm. The introduced algorithm has increased the number of solved problems, obtained better running times in more instances, succeeded in returning new optimality certifications, new best known solutions, and new upper-bound values for large problems. Additionally, it has been shown that the RB&C algorithm obtains better quality solutions than the state-of-the-art heuristics for the OP within the 5-hour running time limit.

Nevertheless, there are many research lines that remain open after this work. One of the most demanding aspects to improve in the presented approach is the implementation of advanced branching techniques. The use of more general cuts, such as combs and clique trees, and the development of their respective separation algorithms for cycle problems might help to improve the performance of the RB&C algorithm. All these future contributions might help to solve the remaining instances until optimality, but we can anticipate it will not be an easy challenge. Implementing the contributions in this paper to other cycle problems which are different from the OP will definitely help to comprehend their importance in the context of cycle problems with a more general view.

Acknowledgements The first and second authors are partially supported by the project PID2019-104933GB-I00 (Spanish Ministry of Science and Innovation). The first and third authors are partially supported by the projects BERC 2018-2021 (Basque Government) and by SEV-2017-0718 (Spanish Ministry of Economy and Competitiveness). The first author is also supported by the grant BES-2015-072036 (Spanish Ministry of Economy and Competitiveness) and project ELKARTEK (Basque Government). The second author is supported by IT-1252-19 (Basque Government) and GIU17/011 (University of the Basque Country). The third author is also supported by IT-1244-19 (Basque Government) and TIN2016-78365R (Spanish Ministry of Science and Innovation). We gratefully acknowledge the authors of the TSP solver Concorde for making their code available to the public, since it has been the working basis of our implementations. We also thank Prof. J.J. Salazar-Gonzalez who provided us with the codes used in [Fischetti et al., 1999].
References

[Angelelli et al., 2017] Angelelli, E., Archetti, C., Filippi, C., and Vindigni, M. (2017). The probabilistic orienteering problem. Computers & Operations Research, 81:269–281.

[Applegate et al., 2007] 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.

[Archetti et al., 2016] Archetti, C., Corberán, A., Plana, I., Sanchis, J. M., and Speranza, M. G. (2016). A branch-and-cut algorithm for the orienteering arc routing problem. Computers & Operations Research, 66:95 – 104.

[Archetti et al., 2014] Archetti, C., Speranza, M. G., Corberán, A., Sanchis, J. M., and Plana, I. (2014). The team orienteering arc routing problem. Transportation Science, 48(3):442–457.

[Balas, 1975] Balas, E. (1975). Facets of the knapsack polytope. Mathematical Programming, 8(1):146–164.

[Bauer, 1997] Bauer, P. (1997). The circuit polytope: Facets. Mathematics of Operations Research, 22(1):110–145.

[Bianchessi et al., 2018] Bianchessi, N., Mansini, R., and Speranza, M. G. (2018). A branch-and-cut algorithm for the team orienteering problem. International Transactions in Operational Research, 25(2):627–635.

[Boussier et al., 2007] Boussier, S., Feillet, D., and Gendreau, M. (2007). An exact algorithm for team orienteering problems. 4OR quarterly journal of the Belgian, French and Italian Operations Research Societies, 5:211–230.

[Dang et al., 2013] Dang, D.-C., El-Hajj, R., and Moukrim, A. (2013). A branch-and-cut algorithm for solving the team orienteering problem. In Gomes, C. and Sellmann, M., editors, Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, pages 332–339, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

[Gendreau et al., 1998] Gendreau, M., Laporte, G., and Semet, F. (1998). A branch-and-cut algorithm for the undirected selective traveling salesman problem. Networks, 32:263–273.

[Goldberg and Tsioutsiouliklis, 2001] Goldberg, A. V. and Tsioutsiouliklis, K. (2001). Cut tree algorithms: An experimental study. Journal of Algorithms, 38(1):51 – 83.

[Golden et al., 1987] Golden, B. L., Levy, L., and Vohra, R. (1987). The orienteering problem. Naval Research Logistics, 34:307–318.

[Grötschel and Holland, 1991] Grötschel, M. and Holland, O. (1991). Solution of large-scale symmetric travelling salesman problems. Mathematical Programming, 51(1):141–202.

[Keshkar et al., 2015] Keshkar, M., Ziarati, K., Bettinelli, A., and Vigo, D. (2015). Enhanced exact solution methods for the team orienteering problem. International Journal of Production Research, ahead-of-print:1–11.

[Kobeaga et al., 2018] Kobeaga, G., Merino, M., and Lozano, J. A. (2018). An efficient evolutionary algorithm for the orienteering problem. Computers & Operations Research, 90:42 – 59.

[Kobeaga et al., 2020] Kobeaga, G., Merino, M., and Lozano, J. A. (2020). On solving cycle problems with branch-and-cut: Extending shrinking and exact subcycle elimination separation algorithms. arXiv:2004.14574.

[Laporte and Martello, 1990] Laporte, G. and Martello, S. (1990). The selective travelling salesman problem. Discrete Applied Mathematics, 26(2):193–207.

[Leifer and Rosenwein, 1994] Leifer, A. C. and Rosenwein, M. B. (1994). Strong linear programming relaxations for the orienteering problem. European Journal of Operational Research, 73(3):517–523.

[Padberg and Hong, 1980] Padberg, M. and Hong, S. (1980). On the symmetric travelling salesman problem: A computational study, pages 78–107. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Poggi et al., 2010] Poggi, M., Viana, H., and Uchoa, E. (2010). The Team Orienteering Problem: Formulations and Branch-Cut and Price. In Erlebach, T. and Lübbekece, M., editors, 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS’10), volume 14 of OpenAccess Series in Informatics (OASIcs), pages 142–155, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Ramesh et al., 1992] Ramesh, R., Yoon, Y.-S., and Karwan, M. H. (1992). An optimal algorithm for the orienteering tour problem. ORSA Journal on Computing, 4(2):155–165.

[Riera-Ledesma and Salazar-González, 2017] Riera-Ledesma, J. and Salazar-González, J. J. (2017). Solving the team orienteering arc routing problem with a column generation approach. European Journal of Operational Research, 262(1):14 – 27.
[Santini, 2019] Santini, A. (2019). An adaptive large neighbourhood search algorithm for the orienteering problem. Expert Systems with Applications, 123:154 – 167.

[Tsiligirides, 1984] Tsiligirides, T. (1984). Heuristic methods applied to orienteering. Journal of the Operational Research Society, 35:797–809.

[Vansteenwegen and Gunawan, 2019] Vansteenwegen, P. and Gunawan, A. (2019). State-of-the-Art Solution Techniques for OP and TOP, pages 41–66. Springer International Publishing, Cham.
Appendices

A Configuration of Components: Detailed Results

In this section, we show the detailed results of the alternative RB&C configurations by instances and generations. Each configuration has been executed five times with a 5-hour execution time limit. We show the obtained results of the configuration in terms of lower-bound values, LB, upper-bound values, UB, and time (in seconds) performance, Time. For the LB and UB, the obtained best value for each configuration (the maximum for LB and the minimum for the UB) is presented in the Best column. Regarding the Time, the Mean column shows the meantime of the five executions. The Gap column represents the relative distance to best-known value (higher Best value in the case of LB, and lower Best in the case of UB and Mean in the case of Time, respectively).
| Strategy         | Gen1       | Gen2       | Gen3       |
|------------------|------------|------------|------------|
|                  | LB | UB | Time       |LB | UB | Time       |LB | UB | Time       |
| REFERENCE        | 49 | 0  | 0.04       | 2708 | 0 | 2708       | 0  | 1.13 | 90.94     | 2430 | 0 | 2430       | 0  | 1.03 | 39.55     |
| - SRK            | 49 | 0  | 0.04       | 119.35 | 2708 | 0 | 2708       | 1.21 | 104.70 | 2430 | 0 | 2430       | 1.06 | 42.60 | 21.84     |
| - SRK=CC2S3      | 49 | 0  | 0.04       | 111.83 | 2708 | 0 | 2708       | 1.38 | 133.98 | 2430 | 0 | 2430       | 0.90  | 21.84 | 21.84     |
| - CC STRATS      | 49 | 0  | 0.04       | 124.73 | 2708 | 0 | 2708       | 1.13 | 90.57  | 2430 | 0 | 2430       | 1.04  | 39.74 | 21.84     |
| - EPH BLOSSOM    | 49 | 0  | 0.03       | 64.52  | 2708 | 0 | 2708       | 1.44 | 143.58 | 2430 | 0 | 2430       | 0.74  | 40.00 | 30.56     |
| + FST BLOSSOM    | 49 | 0  | 0.09       | 398.92 | 2708 | 0 | 2708       | 1.32 | 123.29 | 2430 | 0 | 2430       | 0.85  | 47.06 | 14.47     |
| - EDGE COVER     | 49 | 0  | 0.03       | 83.87  | 2708 | 0 | 2708       | 1.33 | 125.56 | 2430 | 0 | 2430       | 1.76  | 136.91 | 34.99     |
| - CYCLE COVER    | 49 | 0  | 0.05       | 174.19 | 2708 | 0 | 2708       | 1.20 | 103.38 | 2430 | 0 | 2430       | 0.79  | 30.70 | 7.02      |
| - PATH           | 49 | 0  | 0.04       | 116.13 | 2708 | 0 | 2708       | 1.39 | 135.40 | 2430 | 0 | 2430       | 0.79  | 30.70 | 7.02      |
| + VERTEX COVER   | 49 | 0  | 0.04       | 104.30 | 2708 | 0 | 2708       | 1.11 | 87.02  | 2430 | 0 | 2430       | 0.98  | 31.94 | 31.94     |
| SEP: TWO SUBLOOPS| 49 | 0  | 0.07       | 266.67 | 2708 | 0 | 2708       | 0.95 | 60.62  | 2430 | 0 | 2430       | 1.00  | 34.99 | 34.99     |

**Table 8: pr76.**

| Strategy         | Gen1       | Gen2       | Gen3       |
|------------------|------------|------------|------------|
|                  | LB | UB | Time       |LB | UB | Time       |LB | UB | Time       |
| REFERENCE        | 363 | 0.00 | 363 | 0.00       | 359.51 | 1031.58 | 19633 | 0.06 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 166.80 | 29.99     |
| - SRK            | 363 | 0.00 | 363 | 0.00       | 643.50 | 1925.50 | 19635 | 0.05 | 19800       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 219.86 | 71.34     |
| - SRK=CC2S3      | 363 | 0.00 | 363 | 0.00       | 120.89 | 280.53 | 19634 | 0.05 | 19800       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 284.01 | 121.34 | 121.34     |
| - CC STRATS      | 363 | 0.00 | 363 | 0.00       | 118.09 | 271.70 | 19633 | 0.06 | 19802       | 0.02 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 2696.49 | 2001.40 |
| - EPH BLOSSOM    | 363 | 0.00 | 363 | 0.00       | 31.77  | 0.00    | 19643 | 0.01 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 316.47 | 146.63     |
| + FST BLOSSOM    | 363 | 0.00 | 363 | 0.00       | 420.83 | 1224.61 | 19634 | 0.05 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 252.53 | 96.80      |
| - EDGE COVER     | 363 | 0.00 | 363 | 0.00       | 110.11 | 246.59 | 19642 | 0.01 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 210.91 | 64.36      |
| - CYCLE COVER    | 363 | 0.00 | 363 | 0.00       | 252.18 | 693.77 | 19629 | 0.08 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 212.04 | 65.25      |
| - PATH           | 363 | 0.00 | 363 | 0.00       | 81.69  | 157.14 | 19637 | 0.04 | 19799       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 305.62 | 138.18     |
| + VERTEX COVER   | 363 | 0.00 | 363 | 0.00       | 300.17 | 844.81 | 19631 | 0.07 | 19801       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 146.51 | 14.18      |
| SEP: TWO SUBLOOPS| 363 | 0.00 | 363 | 0.00       | 190.93 | 560.97 | 19611 | 0.17 | 19800       | 0.01 | 18000.00 | 0.00       | 15498 | 0.00 | 15498       | 0.00       | 194.63 | 51.68      |

**Table 9: att532.**
A revisited branch-and-cut algorithm for large-scale orienteering problems

Table 10: vm1084.

| Strategy         | Gen1 | Gen2 | Gen3 |
|------------------|------|------|------|
| REFERENCE        | LB   | UB   | Time|
|                  | Best | Gap  | Best | Gap  |
|                  | 777  | 0.00 | 777  | 0.00 |
| LB               | 40770| 0.02 | 40954| 0.02 |
| UB               | 18000| 0.00 | 18000| 0.00 |

Table 11: r11323.

| Strategy         | Gen1 | Gen2 | Gen3 |
|------------------|------|------|------|
| REFERENCE        | LB   | UB   | Time|
|                  | Best | Gap  | Best | Gap  |
|                  | 814  | 0.00 | 814  | 0.00 |
| LB               | 43377| 0.00 | 43454| 0.18 |
| UB               | 18000| 24.62| 18000| 24.62 |


| Strategy                  | Gen1 | Gen2 | Gen3 |
|---------------------------|------|------|------|
|                          | LB   | UB   | Time | LB   | UB   | Time | LB   | UB   | Time |
| Best | Gap  | Best | Gap  | Mean | Gap  | Best | Gap  | Best | Gap  | Mean | Gap  |
| REFERENCE                 | 1276 | 0.23 | 1282 | 0.00 | 18000| 0   | 68013| 0.16 | 68305| 0.01 | 18000| 0   |
| - SRK                     | 1271 | 0.63 | 1282 | 0.00 | 18000| 0   | 67812| 0.45 | 68306| 0.01 | 18000| 0   |
| - SRK=CV2S3              | 1278 | 0.08 | 1282 | 0.00 | 18000| 0   | 67863| 0.38 | 68306| 0.01 | 18000| 0   |
| - CC STRATS              | 1278 | 0.08 | 1282 | 0.00 | 18000| 0   | 68016| 0.15 | 68304| 0.01 | 18000| 0   |
| - EPH BLOSSOM            | 1273 | 0.47 | 1284 | 0.16 | 18000| 0   | 67735| 0.57 | 68460| 0.23 | 18000| 0   |
| - EGH BLOSSOM            | 1278 | 0.08 | 1282 | 0.00 | 18000| 0   | 68029| 0.14 | 68311| 0.02 | 18000| 0   |
| + FST BLOSSOM            | 1279 | 0.00 | 1282 | 0.00 | 18000| 0   | 67986| 0.20 | 68300| 0.00 | 18000| 0   |
| - EDGE COVER             | 1272 | 0.55 | 1282 | 0.00 | 18000| 0   | 67877| 0.36 | 68306| 0.01 | 18000| 0   |
| - CYCLE COVER            | 1275 | 0.31 | 1282 | 0.00 | 18000| 0   | 68055| 0.10 | 68302| 0.00 | 18000| 0   |
| - PATH                   | 1274 | 0.39 | 1282 | 0.00 | 18000| 0   | 67831| 0.43 | 68309| 0.01 | 18000| 0   |
| + VERTEX COVER           | 1276 | 0.23 | 1282 | 0.00 | 18000| 0   | 68032| 0.13 | 68300| 0.00 | 18000| 0   |
| SEP: TWO SUBLOOPS        | 1276 | 0.23 | 1282 | 0.00 | 18000| 0   | 67967| 0.23 | 68314| 0.02 | 18000| 0   |
| BRANCH HEUR-PB           | 1274 | 0.39 | 1282 | 0.00 | 18000| 0   | 67830| 0.43 | 68300| 0.00 | 18000| 0   |
| BRANCH HEUR-VF - EA4OP   | 1278 | 0.08 | 1282 | 0.00 | 18000| 0   | 67981| 0.21 | 68307| 0.01 | 18000| 0   |

Table 12: vm1748.
B Comparison with state-of-the-art Algorithms: Detailed Results

In this appendix, we detail the experimental results for the four algorithms (FST B&C, EA4OP, ALNS and RB&C). Table 13 shows the results for medium-sized instances of generation 1, Table 14 for large-sized instances of generation 1, Table 15 for medium-sized instances of generation 2, Table 16 for large-sized instances of generation 2, Table 17 for medium-sized instances of generation 3 and Table 18 for large-sized instances of generation 3.

In the Best column, we show the global best-known lower and upper-bound values. For each algorithm, we detail the best LB, the goodness gap GGap, the best UB, and the meantime (in seconds). The GGap represents the relative distance between the algorithm’s best LB and the global best-known LB. For the RB&C algorithm we also detail the optimality gap OGap which represents the relative distance between the obtained LB and UB by RB&C.

For each algorithm, generation and size, we have calculated the average gap and running time over the instances where a feasible solution was obtained by the algorithm. In those instances where the time limit was reached, a running time of 5 hours has been used. These averages are shown in the last row of the tables. The symbols in the tables mean the following:

- *: best-known solution achieved
- -: not comparable result
- .: the code finished unexpectedly
| Instance | Best | FST | EA4OP | ALNS | RB&C |
|----------|------|-----|-------|------|------|
| n        | LB   | UB  | LB    | GGap | UB   |LGap | Time |
| an48     | 31   | 31  | 31    | 0.00 | 31   | 0.25 | 6.77 |
| gr48     | 31   | 31  | 31    | 0.00 | 31   | 0.24 | 7.20 |
| bk48     | 30   | 30  | 30    | 0.00 | 30   | 0.24 | 9.51 |
| c651     | 29   | 29  | 29    | 0.00 | 29   | 0.30 | 9.42 |
| berlin52 | 37   | 37  | 37    | 0.00 | 37   | 0.10 | 11.3 |
| brasil58 | 46   | 46  | 46    | 0.00 | 46   | 0.10 | 15.99|
| st70     | 43   | 43  | 43    | 0.10 | 43   | 0.32 | 15.99|
| est76    | 47   | 47  | 47    | 0.10 | 47   | 0.33 | 20.51|
| pr76     | 49   | 49  | 49    | 0.10 | 49   | 0.40 | 16.16|
| gr90     | 64   | 64  | 64    | 0.10 | 64   | 1.44 | 20.31|
| rat99    | 52   | 52  | 52    | 0.40 | 52   | 0.66 | 27.75|
| kroA100  | 56   | 56  | 56    | 0.40 | 55   | 0.34 | 34.75|
| kroB100  | 58   | 58  | 58    | 0.56 | 57   | 0.63 | 34.06|
| kroC100  | 56   | 56  | 56    | 0.40 | 56   | 0.48 | 34.32|
| kroD100  | 59   | 59  | 59    | 0.10 | 58   | 0.65 | 34.61|
| kroE100  | 57   | 57  | 57    | 1.59 | 57   | 0.30 | 32.26|
| rd100    | 61   | 61  | 61    | 0.20 | 61   | 0.74 | 29.49|
| eli101   | 64   | 64  | 64    | 0.10 | 64   | 0.79 | 31.73|
| lin105   | 66   | 66  | 66    | 0.30 | 66   | 1.42 | 32.11|
| pr107    | 54   | 54  | 54    | 0.30 | 54   | 0.93 | 78.46|
| gr120    | 75   | 75  | 75    | 0.10 | 74   | 1.32 | 29.58|
| pr124    | 75   | 75  | 75    | 0.30 | 75   | 1.11 | 49.64|
| bier127  | 103  | 103 | 103   | 0.30 | 103  | 1.18 | 40.84|
| pr136    | 71   | 71  | 71    | 1.40 | 71   | 0.96 | 29.97|
| gr137    | 81   | 81  | 81    | 1.50 | 80   | 3.70 | 39.21|
| pr144    | 77   | 77  | 77    | 1.30 | 77   | 2.61 | 87.82|
| kroA150  | 86   | 86  | 86    | 1.75 | 86   | 1.17 | 82.79|
| kroB150  | 87   | 87  | 87    | 1.20 | 86   | 1.10 | 81.64|
| pr152    | 77   | 77  | 77    | 1.40 | 77   | 3.64 | 91.38|
| u159     | 93   | 93  | 93    | 3.40 | 92   | 1.08 | 99.63|
| rat195   | 102  | 102 | 102   | 2.60 | 99   | 2.74 | 195.57|
| d198     | 123  | 123 | 123   | 3.20 | 123  | 1.23 | 65.57|
| kroA200  | 117  | 117 | 117   | 1.20 | 117  | 1.74 | 114.75|
| kroB200  | 119  | 119 | 119   | 1.40 | 119  | 1.67 | 86.58|
| gr202    | 145  | 145 | 145   | 1.70 | 145  | 6.89 | 187.56|
| ts225    | 124  | 124 | 124   | 1.28 | 124  | 2.79 | 180.00|
| tsp225   | 129  | 129 | 129   | 0.94 | 127  | 2.29 | 198.47|
| pr226    | 126  | 126 | 126   | 1.66 | 126  | 6.61 | 181.94|
| gr229    | 176  | 176 | 176   | 0.90 | 176  | 8.31 | 108.27|
| gi262    | 158  | 158 | 158   | 0.90 | 156  | 2.83 | 240.02|
| pr264    | 132  | 132 | 132   | 2.12 | 132  | 5.62 | 314.29|
| a280     | 147  | 147 | 147   | 1.60 | 143  | 3.72 | 239.06|
| pr299    | 162  | 162 | 162   | 1.11 | 162  | 1.23 | 410.90|
| lin318   | 205  | 205 | 205   | 2.20 | 202  | 1.46 | 294.23|
| rd400    | 239  | 239 | 239   | 3.70 | 234  | 6.59 | 422.56|

**Table 13: Generation 1, n ≤ 400**
Table 14: Generation 1, \( n > 400 \)

| Instance | Best | UB  | LB  | GGap | UB  | Time | LB  | GGap | UB  | OGap | Time |
|----------|------|-----|-----|------|-----|------|-----|------|-----|------|------|
| fl417    | 228  | 230 |     |      |     |      |     |      |     |      |      |
| gr431    | 350  | 350 |     |      |     |      |     |      |     |      |      |
| pr439    | 313  | 313 |     |      |     |      |     |      |     |      |      |
| pc442    | 251  | 251 |     |      |     |      |     |      |     |      |      |
| d493     | 320  | 320 |     |      |     |      |     |      |     |      |      |
| at532    | 363  | 363 |     |      |     |      |     |      |     |      |      |
| at53     | 425  | 425 |     |      |     |      |     |      |     |      |      |
| pa561    | 357  | 357 |     |      |     |      |     |      |     |      |      |
| u574     | 354  | 354 |     |      |     |      |     |      |     |      |      |
| ra575    | 322  | 322 |     |      |     |      |     |      |     |      |      |
| ps54     | 435  | 435 |     |      |     |      |     |      |     |      |      |
| d657     | 366  | 366 |     |      |     |      |     |      |     |      |      |
| gr666    | 506  | 506 |     |      |     |      |     |      |     |      |      |
| a724     | 439  | 439 |     |      |     |      |     |      |     |      |      |
| rat783   | 438  | 438 |     |      |     |      |     |      |     |      |      |
| ds9000   | 656  | 656 |     |      |     |      |     |      |     |      |      |
| p1002    | 606  | 606 |     |      |     |      |     |      |     |      |      |
| u1060    | 660  | 660 |     |      |     |      |     |      |     |      |      |
| vm1064   | 777  | 777 |     |      |     |      |     |      |     |      |      |
| pcb1173  | 675  | 675 |     |      |     |      |     |      |     |      |      |
| d1291    | 715  | 715 |     |      |     |      |     |      |     |      |      |
| r1304    | 802  | 802 |     |      |     |      |     |      |     |      |      |
| r1323    | 814  | 814 | 811 | 0.37 | 846 | 18000.00 | 782 | 3.93 | 89.68 | 785 | 3.56 | 14585.10 |
| nw1379   | 815  | 817 |     |      |     |      |     |      |     |      |      |
| fl1400   | 1048 | 1084 |     |      |     |      |     |      |     |      |      |
| u1432    | 754  | 764 |     |      |     |      |     |      |     |      |      |
| h57     | 897  | 900 |     |      |     |      |     |      |     |      |      |
| d1655    | 922  | 924 |     |      |     |      |     |      |     |      |      |
| vm1748   | 1276 | 1282 |     |      |     |      |     |      |     |      |      |
| r1817    | 983  | 983 |     |      |     |      |     |      |     |      |      |
| r1889    | 1226 | 1226 |     |      |     |      |     |      |     |      |      |
| d2103    | 1200 | 1200 |     |      |     |      |     |      |     |      |      |
| u2152    | 1151 | 1151 |     |      |     |      |     |      |     |      |      |
| u2319    | 1170 | 1171 |     |      |     |      |     |      |     |      |      |
| p2392    | 1316 | 1415 |     |      |     |      |     |      |     |      |      |
| pcb3038  | 1727 | 1730 |     |      |     |      |     |      |     |      |      |
| r3795    | 1965 | 2249 |     |      |     |      |     |      |     |      |      |
| fn441    | 2541 | 2570 |     |      |     |      |     |      |     |      |      |
| t5915    | 3593 | 3786 |     |      |     |      |     |      |     |      |      |
| t5934    | 3632 | 3752 |     |      |     |      |     |      |     |      |      |
| pla7397  | 5289 | 5657 |     |      |     |      |     |      |     |      |      |
| n > 400  | 435  | 435 | 7433.41 | 4.32 | 990.82 | 3.12 | 11802.68 | 0.11 | 1.49 | 10387.02 |
### Table 15: Generation 2, $n \leq 400$

| Instance | Best | FST | EA4OP | ALNS | RB&C |
|----------|------|-----|-------|------|------|
| at48 | 1717 | 1717 | * | 1717 | * | 6.77 | * | 0.04 |
| gr48 | 1761 | 1761 | * | 1783 | * | 7.87 | * | 1.32 |
| bk48 | 1614 | 1614 | * | 1614 | * | 7.19 | * | 0.10 |
| el51 | 1674 | 1674 | 1674 | * | 1674 | * | 10.13 | * | 0.96 |
| berlin52 | 1897 | 1897 | 1897 | * | 1897 | * | 10.74 | * | 3.23 |
| bra58 | 2220 | 2220 | 2220 | * | 2220 | * | 12.32 | * | 0.46 |
| st70 | 2286 | 2286 | 2286 | * | 2286 | * | 21.65 | * | 1.77 |
| el76 | 2550 | 2550 | 2550 | * | 2550 | * | 16.06 | * | 0.62 |
| pr76 | 2708 | 2708 | 2708 | * | 2708 | * | 19.48 | * | 1.46 |
| gr96 | 3396 | 3396 | 3396 | * | 3396 | * | 31.98 | * | 9.50 |
| rat99 | 2944 | 2944 | 2944 | * | 2944 | * | 32.08 | * | 3.25 |
| kn40 | 3212 | 3212 | 3212 | * | 3212 | * | 32.85 | * | 0.70 |
| kn50 | 3241 | 3241 | 3241 | * | 3241 | * | 48.39 | * | 13.28 |
| kroC100 | 2944 | 2944 | 2944 | * | 2944 | * | 39.27 | * | 2.22 |
| kroD100 | 3307 | 3307 | 3307 | * | 3307 | * | 30.52 | * | 3.62 |
| eil100 | 3359 | 3359 | 3359 | * | 3359 | * | 30.80 | * | 11.31 |
|eil101 | 3359 | 3359 | 3359 | * | 3359 | * | 30.80 | * | 0.36 |
| lind105 | 3359 | 3359 | 3359 | * | 3359 | * | 30.80 | * | 4.15 |
| pr107 | 2667 | 2667 | 2667 | * | 2667 | * | 69.67 | * | 0.20 |
| gr120 | 4371 | 4371 | 4371 | * | 4371 | * | 40.41 | * | 5.57 |
| pr124 | 3917 | 3917 | 3917 | * | 3917 | * | 55.25 | * | 1.07 |
| bier127 | 5383 | 5383 | 5383 | * | 5383 | * | 23.01 | * | 0.96 |
| pr136 | 4309 | 4309 | 4309 | * | 4309 | * | 35.63 | * | 1.25 |
| gr137 | 4286 | 4286 | 4286 | * | 4286 | * | 63.80 | * | 10.65 |
| pr144 | 4003 | 4003 | 4003 | * | 4003 | * | 100.20 | * | 32.23 |
| kroA150 | 4918 | 4918 | 4918 | * | 4918 | * | 80.06 | * | 60.43 |
| kroB150 | 4869 | 4869 | 4869 | * | 4869 | * | 61.96 | * | 16.94 |
| pr152 | 4279 | 4279 | 4279 | * | 4279 | * | 67.41 | * | 1.85 |
| u159 | 4960 | 4960 | 4960 | * | 4960 | * | 109.59 | * | 14.96 |
| rat195 | 5791 | 5791 | 5791 | * | 5791 | * | 263.23 | * | 46.09 |
| kroA200 | 6547 | 6547 | 6547 | * | 6547 | * | 116.11 | * | 16.18 |
| kroB200 | 6419 | 6419 | 6419 | * | 6419 | * | 624.00 | * | 20.62 |
| gr200 | 7889 | 7889 | 7889 | * | 7889 | * | 188.27 | * | 139.90 |
| n225 | 6834 | 6834 | 6834 | * | 6834 | * | 94.00 | * | 95.22 |
| tsp225 | 6987 | 6987 | 6987 | * | 6987 | * | 299.73 | * | 54.09 |
| pr226 | 6662 | 6662 | 6662 | * | 6662 | * | 201.68 | * | 2894.81 |
| pr229 | 9177 | 9177 | 9177 | * | 9177 | * | 1379.55 | * | 16.67 |
| pbl26 | 8321 | 8321 | 8321 | * | 8321 | * | 847.41 | * | 64.63 |
| pr264 | 6654 | 6654 | 6654 | * | 6654 | * | 487.41 | * | 0.62 |
| a260 | 8428 | 8428 | 8428 | * | 8428 | * | 487.41 | * | 0.62 |
| p269 | 9182 | 9182 | 9182 | * | 9182 | * | 487.41 | * | 0.62 |
| lin318 | 10923 | 10923 | 10923 | * | 10923 | * | 370.64 | * | 367.53 |
| rd400 | 13652 | 13652 | 13652 | * | 13652 | * | 1174.91 | * | 769.66 |

* Average
Table 16: Generation 2, $n > 400$

| Instance | FST | Best | GAAP | ALNS | RB&C |
|----------|-----|------|------|------|------|
| b417     | 11933 | 12294 | 11894 | 0.33 | 12294 | 1800.00 |
| gr31     | 18318 | 18318 | 18318 | * | 18318 | 969.50 |
| pe439    | 16171 | 16171 | 16171 | * | 16171 | 1298.30 |
| ph442    | 14484 | 14484 | 14484 | * | 14484 | 6259.10 |
| d495     | 16695 | 16695 | 16695 | 1.57 | 17.15 | 6231.42 |
| at532    | 19635 | 19635 | 19598 | 0.19 | 19980 | 1800.00 |
| al535    | 21954 | 21954 | 21954 | * | 2099.70 | 1800.00 |
| pr561    | 19576 | 19576 | 19576 | * | 19276 | 1817.10 |
| u574     | 19351 | 19351 | 19351 | * | 19351 | 612.50 |
| rat57    | 18251 | 18251 | 18251 | * | 18251 | 931.10 |
| pr654    | 17900 | 21566 | 17160 | 4.13 | 21566 | 1800.00 |
| d657     | 21503 | 21503 | 21503 | * | 21503 | 682.40 |
| gr666    | 26514 | 26569 | . . | . . | 26569 | 1800.00 |
| u724     | 24223 | 24223 | 24223 | * | 24223 | 3830.50 |
| rat783   | 25474 | 25474 | 25474 | . . | 25474 | 1800.00 |
| dq100    | 35835 | 35915 | 35772 | 0.18 | 35917 | 1800.00 |
| pr100    | 33030 | 33092 | 27066 | 18.06 | . . | 1800.00 |
| u1060    | 36151 | 36271 | 36110 | 2.88 | 77.78 | 1800.00 |
| vm1084   | 40777 | 40952 | 40687 | 0.22 | 40954 | 1800.00 |
| pch1173  | 37035 | 37100 | . . | . . | 37100 | 1800.00 |
| d1291    | 37778 | 37854 | . . | . . | 37854 | 1800.00 |
| rli1304  | 42275 | 42359 | . . | . . | 42359 | 1800.00 |
| rli1323  | 43377 | 43450 | 43347 | 0.07 | 43450 | 1800.00 |
| nrw1379  | 46676 | 46787 | . . | . . | 46787 | 1800.00 |
| fl1400   | 56992 | 64298 | 53222 | 6.12 | 64296 | 1800.00 |
| ui1472   | 46946 | 47018 | . . | . . | 47018 | 1800.00 |
| fl1577   | 45505 | 50154 | . . | . . | 50154 | 1800.00 |
| di655    | 49319 | 53083 | . . | . . | 53083 | 1800.00 |
| vmi1748  | 68042 | 68303 | . . | . . | 68303 | 1800.00 |
| ui1817   | 54245 | 54554 | . . | . . | 54554 | 1800.00 |
| rl1889   | 63308 | 64425 | 52047 | 17.79 | . . | 1800.00 |
| d2103    | 63426 | 63426 | . . | . . | 63426 | 1800.00 |
| u2152    | 64649 | 64775 | 58375 | 16.51 | . . | 1800.00 |
| u2319    | 80914 | 81139 | 72790 | 10.04 | . . | 1800.00 |
| pr2392   | 72843 | 78237 | 64577 | 11.35 | . . | 1800.00 |
| pch3038  | 97902 | 97995 | 83951 | 14.25 | . . | 1800.00 |
| fl3795   | 103397 | 142895 | . . | . . | 142895 | 1800.00 |
| fnl4461  | 147109 | 150189 | . . | . . | 150189 | 1800.00 |
| fli591   | 184424 | 197729 | . . | . . | 197729 | 1800.00 |
| fli5934  | 187034 | 196805 | . . | . . | 196805 | 1800.00 |
| plaze797 | 281977 | 297246 | . . | . . | 297246 | 1800.00 |

| Average | 4.51 | 11464.10 | 3.13 | 10093.37 | 2.87 | 12827.93 | 0.40 | 3.06 | 15369.91 |
Table 17: Generation 3, $n \leq 400$

| Instance | Best | FST | EAOIP | ALNS | RB&O |
|----------|------|-----|-------|------|-------|
| at48    | 1049 | 1049 | 1049  | 1049 | 1049  |
| gr48    | 1480 | 1480 | 1480  | 1480 | 1480  |
| bk48    | 1764 | 1764 | 1764  | 1764 | 1764  |
| eil51   | 1399 | 1399 | 1399  | 1399 | 1399  |
| berlin52| 1036 | 1036 | 1036  | 1036 | 1036  |
| braz58  | 1702 | 1702 | 1702  | 1702 | 1702  |
| st70    | 2108 | 2108 | 2108  | 2108 | 2108  |
| eil76   | 2467 | 2467 | 2467  | 2467 | 2467  |
| pr76    | 2430 | 2430 | 2430  | 2430 | 2430  |
| gr96    | 3170 | 3170 | 3170  | 3170 | 3170  |
| rat99   | 2908 | 2908 | 2908  | 2908 | 2908  |
| kroA100| 3211 | 3211 | 3211  | 3211 | 3211  |
| kroB100| 2804 | 2804 | 2804  | 2804 | 2804  |
| kroC100| 3155 | 3155 | 3155  | 3155 | 3155  |
| kroD100| 3167 | 3167 | 3167  | 3167 | 3167  |
| kroE100| 3049 | 3049 | 3049  | 3049 | 3049  |
| rd100  | 2926 | 2926 | 2926  | 2926 | 2926  |
| eil101 | 3345 | 3345 | 3345  | 3345 | 3345  |
| lin105 | 2986 | 2986 | 2986  | 2986 | 2986  |
| pr107  | 1877 | 1877 | 1877  | 1877 | 1877  |
| pr120  | 3779 | 3779 | 3779  | 3779 | 3779  |
| pr124  | 3557 | 3557 | 3557  | 3557 | 3557  |
| bier127| 2365 | 2365 | 2365  | 2365 | 2365  |
| pe136  | 4390 | 4390 | 4390  | 4390 | 4390  |
| pr137  | 3954 | 3954 | 3954  | 3954 | 3954  |
| pr144  | 3745 | 3745 | 3745  | 3745 | 3745  |
| kroA150| 5039 | 5039 | 5039  | 5039 | 5039  |
| kroB150| 5314 | 5314 | 5314  | 5314 | 5314  |
| pe152  | 3905 | 3905 | 3905  | 3905 | 3905  |
| u159   | 5272 | 5272 | 5272  | 5272 | 5272  |
| rat195 | 6195 | 6195 | 6195  | 6195 | 6195  |
| d198   | 6320 | 6320 | 6320  | 6320 | 6320  |
| kroA200| 6123 | 6123 | 6123  | 6123 | 6123  |
| kroB200| 6266 | 6266 | 6266  | 6266 | 6266  |
| pr202  | 8616 | 8616 | 8616  | 8616 | 8616  |
| pr225  | 7575 | 7575 | 7575  | 7575 | 7575  |
| tsp225 | 7740 | 7740 | 7740  | 7740 | 7740  |
| pr226  | 6993 | 6993 | 6993  | 6993 | 6993  |
| pr229  | 6328 | 6328 | 6328  | 6328 | 6328  |
| kro262 | 9246 | 9246 | 9246  | 9246 | 9246  |
| pr264  | 8137 | 8137 | 8137  | 8137 | 8137  |
| a280   | 9774 | 9774 | 9774  | 9774 | 9774  |
| pe299  | 10343| 10343| 10343 | 10343| 10343 |
| lin318 | 10368| 10368| 10368 | 10368| 10368 |
| rd400  | 13223| 13223| 13223 | 13223| 13223|

**Average:** * 149.66 0.85 2.35 0.55 180.55 * 272.43
Table 18: Generation 3, $n > 400$

| Instance | Best | FST | EE4OP | ALNS | RB&C |
|----------|------|-----|-------|------|------|
| B417     | 14220| 14220| 14220| 14186| 14220| 14186 | 14186 | 14186 |
| gr411    | 10911| 10911| 10911| 10817| 10907| 10907 | 10907 | 10907 |
| pe459    | 15176| 15176| 15176| 15097| 15080| 15080 | 15080 | 15080 |
| pb442    | 14819| 14819| 14819| 14522| 14695| 14695 | 14695 | 14695 |
| d493     | 25188| 25188| 25188| 24981| 24849| 24849 | 24849 | 24849 |
| at532    | 15498| 15498| 15498| 15342| 15335| 15335 | 15335 | 15335 |
| all533   | 9414 | 9414 | 9414 | 9328 | 9308 | 9308  | 9308  | 9308  |
| ns561    | 14819| 14819| 14819| 14819| 14819| 14819 | 14819 | 14819 |
| uv574    | 20064| 20064| 20064| 19691| 19841| 19841 | 19841 | 19841 |
| ra757    | 20109| 20109| 20109| 20109| 20109| 20109 | 20109 | 20109 |
| pe654    | 24492| 24492| 24492| 24130| 24227| 24227 | 24227 | 24227 |
| d657     | 24562| 24562| 24562| 23772| 23829| 23829 | 23829 | 23829 |
| gr666    | 17023| 17048| 17048| 16902| 16709| 16709 | 16709 | 16709 |
| r724     | 28348| 28348| 28348| 27932| 28033| 28033 | 28033 | 28033 |
| rat783   | 27566| 27566| 27566| 27566| 27566| 27566 | 27566 | 27566 |
| s3100    | 31544| 31544| 31544| 30943| 31040| 31040 | 31040 | 31040 |
| pr1002   | 39526| 39526| 39526| 38762| 38502| 38502 | 38502 | 38502 |
| u1060    | 37492| 37492| 37492| 36570| 36570| 36570 | 36570 | 36570 |
| vmi1084  | 37669| 37669| 37669| 37508| 37518| 37518 | 37518 | 37518 |
| pch1173  | 41257| 41257| 41257| 40069| 40153| 40153 | 40153 | 40153 |
| d1291    | 42153| 42153| 42153| 38132| 39199| 39199 | 39199 | 39199 |
| r1304    | 44881| 44881| 44881| 41214| 41679| 41679 | 41679 | 41679 |
| r1323    | 47213| 47384| 44458| 46661| 45500| 45500 | 45500 | 45500 |
| nrw1379  | 42920| 42920| 42920| 42920| 42920| 42920 | 42920 | 42920 |
| fl1400   | 57470| 59491| 54792| 57226| 57740| 57740 | 57740 | 57740 |
| ui1472   | 47778| 47895| 47778| 46657| 47242| 47242 | 47242 | 47242 |
| fl1577   | 45935| 48809| 45935| 45935| 45935| 45935 | 45935 | 45935 |
| di655    | 62048| 62945| 51168| 58728| 60956| 60956 | 60956 | 60956 |
| vmi1748  | 71885| 72010| 69797| 70958| 71244| 71244 | 71244 | 71244 |
| u1817    | 63639| 67670| 52186| 63639| 63806| 63806 | 63806 | 63806 |
| rl1889   | 70065| 71106| 43374| 68422| 68096| 68096 | 68096 | 68096 |
| d2103    | 82877| 82973| 76035| 73333| 81081| 81081 | 81081 | 81081 |
| u2152    | 74007| 78066| 52091| 73400| 72733| 72733 | 72733 | 72733 |
| r2139    | 79351| 81619| 79351| 78113| 79130| 79130 | 79130 | 79130 |
| pr2392   | 85409| 90261| 60225| 84094| 85084| 85084 | 85084 | 85084 |
| pch3038  | 106928|112006| 96356|104667| 105337|105337|105337|105337|
| fl3795   | 97307|116792| 50927| 97707| 95580| 95580 | 95580 | 95580 |
| fn4461   | 146995|152562| 146995| 149336| 180184| 180184| 180184| 180184|
| fl5915   | 203695|217366| 203695|199336| 203695| 203695| 203695| 203695|
| fl5934   | 212021|229405| 212021|207385| 203667| 203667| 203667| 203667|
| pl7397   | 322245|334885| 322245|120744| 312645| 312645| 312645| 312645|

Average: 6.78 13749.78 1.80 1168.44 1.47 12837.26 0.34 2.24 14843.74