Kobeaga, G., Merino, M. & Lozano, J.A. *On solving cycle problems with Branch-and-Cut: extending shrinking and exact subcycle elimination separation algorithms*. **Ann Oper Res** 305, 107–136 (2021). This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/s10479-021-04210-0

On Solving Cycle Problems with Branch-and-Cut: Extending Shrinking and Exact Subcycle Elimination Separation Algorithms

This is a post-peer-review, pre-copyedit version of an article published in:
Annals of Operations Research
The final authenticated version is available online at:
https://doi.org/10.1007/s10479-021-04210-0

Gorka Kobeaga
Basque Center for Applied Mathematics BCAM
gkobeaga@bcamath.org

María Merino
University of the Basque Country UPV/EHU
maria.merino@ehu.eus

Jose A. Lozano
Basque Center for Applied Mathematics BCAM
University of the Basque Country UPV/EHU
jlozano@bcamath.org

August 3, 2021

Abstract

In this paper, we extend techniques developed in the context of the Travelling Salesperson Problem for cycle problems. Particularly, we study the shrinking of support graphs and the exact algorithms for subcycle elimination separation problems. The efficient application of the considered techniques has proved to be essential in the Travelling Salesperson Problem when solving large size problems by Branch-and-Cut, and this has been the motivation behind this work. Regarding the shrinking of support graphs, we prove the validity of the Padberg-Rinaldi general shrinking rules and the Crowder-Padberg subcycle-safe shrinking rules. Concerning the subcycle separation problems, we extend two exact separation algorithms, the Dynamic Hong and the Extended Padberg-Grötschel algorithms, which are shown to be superior to the ones used so far in the literature of cycle problems.

The proposed techniques are empirically tested in 24 subcycle elimination problem instances generated by solving the Orienteering Problem (involving up to 15112 vertices) with Branch-and-Cut. The experiments suggest the relevance of the proposed techniques for cycle problems. The obtained average speedup for the subcycle separation problems in the Orienteering Problem when the proposed techniques are used together is around 50 times in medium-sized instances and around 250 times in large-sized instances.

Keywords cycle problem · branch-and-cut · shrinking · exact separation · subcycle elimination · gomory-hu tree

1 Introduction

The Travelling Salesperson Problem (TSP) has been the source and the testbed of the most important techniques developed for the exact solution of combinational optimization problems. These techniques have been principally developed in the context of the Branch-and-Cut (B&C) algorithm, which combines the Branch-and-Bound (B&B) and the cutting-planes methods, see Applegate et al. (2007) for an historical overview. Eventually, many of these techniques have been successfully adapted to other related problems. However, there are procedures, such as the support graph shrinking and some separation algorithms, that are strongly dependent on the problem peculiarities. As a consequence, these techniques might not have been adapted yet, or there might still be room for further improvements.

As TSP is the most well-known cycle problem, we motivate the goals of this paper focusing on this problem. When a B&B algorithm is used to exactly solve the TSP, which is an Integer Program (IP), the cutting-planes method arises as a natural strategy to handle at least two situations: the exponential number of constraints of the model and the consequences of the linear relaxation of the integer problem. Recall that in a B&B algorithm the branching decisions are made guided by a sequence of Linear Program (LP). These LPs are principally obtained by relaxing the integrality and fixing the variables according to the preceding branching decisions.

Within this approach, the cutting-planes method is required due to the fact that, in order to define a TSP model, an exponential number of constraints in terms of the number of vertices in the TSP is needed, see Padberg Sung (1991). In order to deal with this situation, the exact algorithm is initialized with a subproblem of the LP, let us call this LP_0, that considers a controlled number of constraints. During the algorithm, the excluded constraints are added to LP_0 only if they are required, i.e., if they are violated by the solution of the LP_0. The second reason to consider the cutting-planes method is that since the variables in the linear relaxation of the TSP are considered continuous instead of integers, new families of valid inequalities arise (inequalities that are satisfied by all the cycles), also called cuts, that are not linear combinations of the constraints defining the TSP. Since the number of branch nodes needed to visit by the algorithm is reduced, the cutting-planes are very valuable to decrease the solving time of a B&B algorithm.

Computationally, the most expensive part of the cutting-planes method is to solve the separation problems. Given a solution of the LP_0 and an inequality family, the separation problem for the given family consists of finding either the violated inequalities of the family or a certificate that no violated inequality of the family exists.

The difficulty of efficiently solving the separation problems becomes evident when the number of vertices of the problem increases. It is well known that, in practice, even a polynomial time separation algorithm might turn out to be inefficient for certain families. To mitigate this practical issue, a technique known as shrinking has been exploited in the TSP, see Crowder Padberg (1980); Padberg Rinaldi (1990b); Grötschel Holland (1991). Shrinking consists of safely simplifying, i.e., without losing all the violated inequalities of the family, the support graph generated by the solution of the LP_0. This way, considering that, generally, the separation is harder than the shrinking, the cost of finding the violated inequalities is reduced because the separation is performed in a graph involving a lower number of vertices and edges. In Figure 1, a flowchart of a generic B&C algorithm and the separation algorithm with and without the shrinking.

In the last few decades, many optimization problems have proliferated whose solution is required to be a cycle, but not necessarily Hamiltonian as in the TSP. This is the case for some extensions of the TSP itself, as can be seen in the extensive collection about TSP variants of Gutin Punnen (2007). For instance, the weighted girth problem, consists of finding the minimum cost cycle in a weighted graph, see Coullard Pulleyblank (1989) and Bauer (1997). Cycles are also the solutions of the Generalized TSP (GTSP) where the vertices are labeled in clusters and at least one vertex of each cluster is required to be visited, but not all the vertices, see Fischetti et al. (1995). Other routing problems, which are recently gaining popularity because of their wide range of applications, are the TSP with profits, see Feillet et al. (2005) and Archetti et al. (2014). These problems are the Profitable Tour Problem (PTP), the Orienteering Problem (OP), the Prize Collecting TSP (PCTSP), and their variations. From the TSP with profits, the OP, which consists of finding the cycle that maximizes the collected vertex profits subject to a cycle length constraint, is the one which has been most extensively studied. For a recent book on applications and variants of the OP see Vansteenwegen Gunawan (2019).

This work has three main aims: first, to generalize the shrinking rules (global and subcycle specific) proposed in the literature of the TSP to the case of cycle problems; second, to extend in an effective manner the subcycle exact separation algorithms for cycle problems; and third, to show experimentally the relevance of the proposed shrinking rules and separation algorithms. On the one hand, 6 different shrinking rules for cycle problems are presented in this work, of which three are safe for all the valid inequalities and three are specifically safe for subcycle elimination constraints. On the other hand, we extend two exact separation algorithms proposed in Padberg Grötschel (1985) and Padberg Rinaldi (1990b). We empirically show the contribution of the shrinking and separation strategies in the time reduction and in the generation of violated subcycle elimination constraints. For the experiments, we have used 24 instances of the subcycle separation problem generated in the solution of OP by B&C with up to 15112 number of vertices. The results show that the speedup of using the combination of the proposed shrinking and separation techniques is around 50 times in medium-sized instances and 200 times in large-sized instances.

This paper is structured as follows. In Section 2, we introduce the cycle polytope and other related polytopes used in this work. In Section 3, we study the safe shrinking rules for the cycle polytope. Section 4 includes rules that are particularly safe for Subcycle Elimination Constraints (SEC). In Section 5, two exact separation algorithms of SECs for cycle problems are presented. Finally, in Section 6, we discuss the computational experiments for the different separation algorithms for SECs. Appendices A, B and C of this work are available, which contain pseudocodes of the shrinking

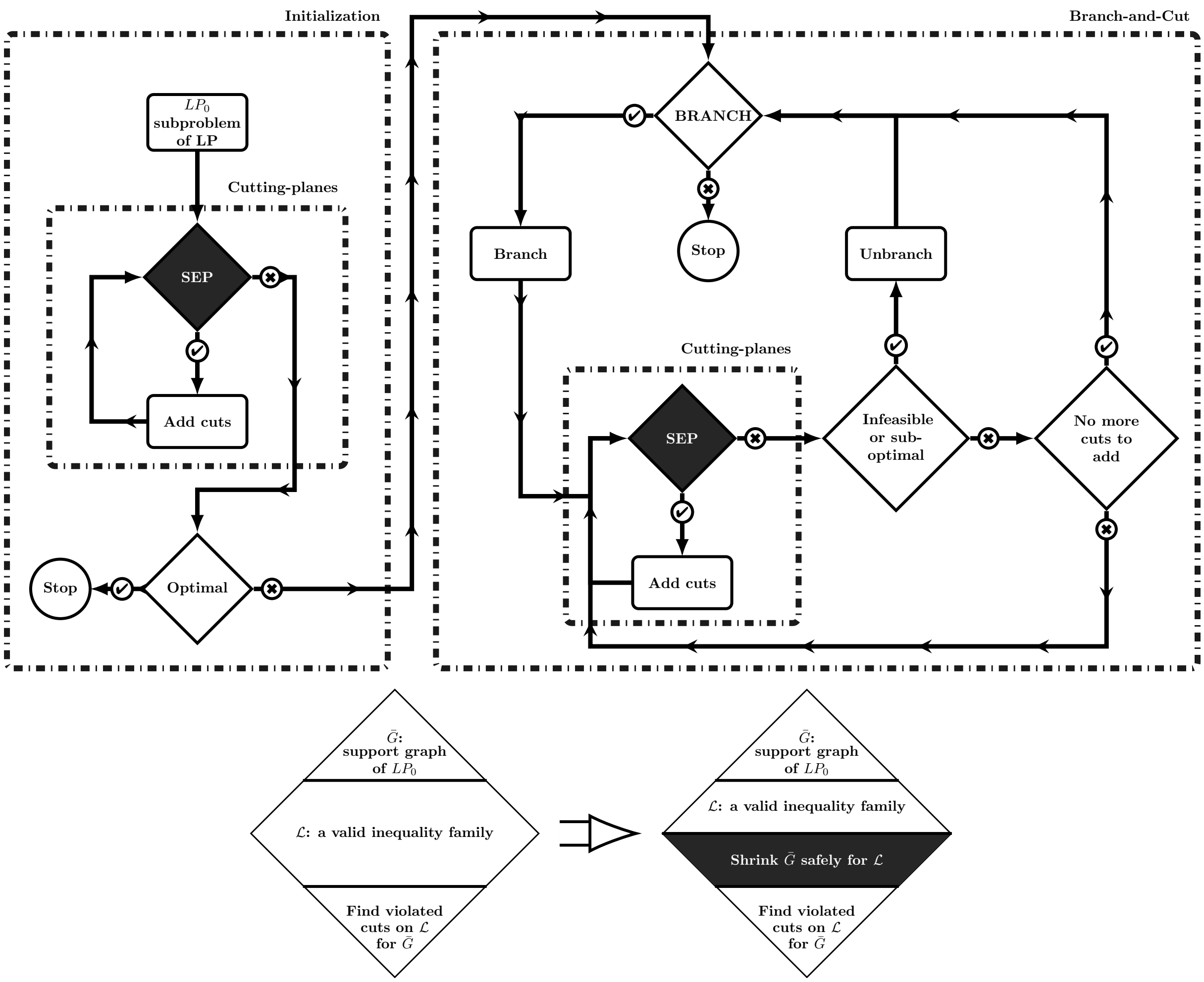


Figure 1: In the top, a flowchart of a generic Branch-and-Cut algorithm. BRANCH is an oracle which returns an unevaluated node in the branching tree. At each action box of the flowchart the subproblem LP_0 is updated and solved. In the bottom, the detailed separation algorithm (SEP) without and with shrinking.

and separation algorithms as well as detailed computational results and figures to illustrate the shrinking techniques. Additionally, we have released the source code of the implementations used for the computational experiments.

2 The Cycle Polytope

Let $G = (V, E)$ be an undirected graph with no loops. Let us define the following sets:

$$(Q : W) := \{[u, v] \in E : u \in Q, v \in W\} \qquad Q, W \subset V \tag{1a}$$
$$\delta(Q) := (Q : V - Q) \qquad Q \subset V \tag{1b}$$
$$E(Q) := (Q : Q) \qquad Q \subset V \tag{1c}$$
$$V(T) := \{v \in V : T \cap (v : V) \neq \emptyset\} \qquad T \subset E \tag{1d}$$
$$N(Q) := V(\delta(Q)) - Q \qquad Q \subset V \tag{1e}$$

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, $V(T)$ is the set of vertices incident with an edge set T, and $N(Q)$ are the neighbour vertices set of Q. For simplicity, we sometimes denote $\{e\}$ and $\{v\}$ by e and v, respectively, e.g., $\delta(v)$ and $V(e)$.

We denote by $\mathbb{R}^V$ and $\mathbb{R}^E$ the space of real vectors whose components are indexed by elements of V and E, respectively. With every subset $T \subset E$ we associate a vector $(y, x)^T = (y^T, x^T)$ called the characteristic vector of T, defined as follows:

$$y_v^T := \begin{cases} 1 & \text{if } v \in V(T) \\ 0 & \text{otherwise} \end{cases} \qquad x_e^T := \begin{cases} 1 & \text{if } e \in T \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

When $y_v^T = 1$, i.e. $v \in V(T)$, we say that the vertex v is visited by the edge set T.

We denote by $\mathcal{C}_G$ the set of (simple) cycles of the graph G. We assume that every cycle $\tau \in \mathcal{C}_G$ is represented as a subset of edges. Then, the cycle polytope P_C^G of the graph G is the convex hull of the characteristic vectors of all the cycles of the graph:

$$P_C^G := conv\{(y,x)^\tau \in \mathbb{R}^{V\times E} : \tau \in \mathcal{C}_G\} \tag{3}$$

By definition, a vector (y,x) belongs to P_C^G if it is a convex combination of cycles of $\mathcal{C}_G$, i.e., $(y,x) \in P_C^G$ if and only if there exists a set of real numbers $\{\lambda_\tau\}_{\tau\in\mathcal{C}_G}$ such that

$$(y,x) = \sum_{\tau\in\mathcal{C}_G} \lambda_\tau (y,x)^\tau \tag{4}$$

$\lambda_\tau \geq 0$ for every $\tau \in \mathcal{C}_G$ and $\sum_{\tau\in\mathcal{C}_G} \lambda_\tau = 1$.

Similarly, we denote by $\mathcal{T}_G$ the set of tours, i.e., Hamiltonian cycles, of the graph G, and by P_{TSP}^G the TSP polytope of the graph G. The P_{TSP}^G is the convex hull of the characteristic vectors of all the tours of the graph:

$$P_{TSP}^G := conv\{(y,x)^\tau \in \mathbb{R}^{V\times E} : \tau \in \mathcal{T}_G\} \tag{5}$$

Note that, $y = 1$ is satisfied by every $(y,x) \in P_{TSP}^G$. Since, the tours form a subset of cycles of G, we have that:

$$P_{TSP}^G \subset P_C^G \tag{6}$$

In order to use Linear Programming based techniques such as the B&C algorithm, the polytope P_C^G must be characterized by means of a system of linear constraints. A complete characterization of the integer points of P_C^G using only edge variables was given in Bauer (1997). In this work, since we find it more convenient to formulate the shrinking rules of Section 3 and Section 4, we consider an equivalent one which uses the vertex and edge variables for the characterization. For $(y,x) \in \mathbb{R}^{V\times E}$, $S \subset V$ and $T \subset E$, we define $y(S) = \sum_{v\in S} y_v$ and $x(T) = \sum_{e\in T} x_e$. Let us consider the following constraints:

$$x(\delta(v)) - 2y_v = 0, \qquad v \in V \tag{7a}$$
$$y_v - x_e \geq 0, \qquad v \in V,\ e \in \delta(v) \tag{7b}$$
$$x(\delta(Q)) - 2y_v - 2y_w \geq -2, \qquad v \in Q \subset V, 3 \leq |Q| \leq |V| - 3,\ w \in V - Q \tag{7c}$$
$$x(E) \geq 3, \tag{7d}$$
$$1 \geq y_v \geq 0, \qquad v \in V \tag{7e}$$
$$x_e \geq 0, \qquad e \in E \tag{7f}$$
$$x_e \in \mathbb{Z} \qquad e \in E \tag{7g}$$

The degree equations (7a) together with the logical constraints (7b) and the integrality constraints (7g) ensure that the visited vertices have exactly two incident edges and the unvisited vertices none. The Subcycle Elimination Constraints (SEC) (7c) ensure that only one connected cycle exists. Throughout the paper, we use the notation $\langle Q, v, w\rangle$ to refer to the SEC defined by the set Q and the vertices $v \in Q$ and $w \notin Q$. In the literature, the SECs have also been called Generalized Subtour Elimination Constraints (GSEC). The inequality (7d) imposes the property that the undirected cycles contain at least 3 edges. The conditions (7e), (7f) and (7g) impose that all the variables are 0-1. Note that the integrality of the y_v variables is ensured by (7a), (7b) and (7g), and the condition $x_e \leq 1$ is ensured by (7b) and (7e). Considering the constraints in (7), the cycle polytope of a graph $G = (V,E)$ can be expressed as follows:

$$P_C^G = conv\{(y,x) \in \mathbb{R}^{V\times E} : (y,x) \text{ satisfies (7a), (7b), (7c), (7d), (7e), (7f), (7g)}\} \tag{8}$$

In some problems, for instance OP and PCTSP, a feasible solution must visit a depot vertex, i.e., $y_d = 1$ for a vertex $d \in V$. In such cases, the family of SECs (7c) that define the cycle polytope can be substituted with the following subfamily:

$$x(\delta(Q)) - 2y_v \geq 0, \qquad v \in Q \subset V, 3 \leq |Q| \leq |V| - 3,\ d \notin Q \tag{9}$$

where each constraint can be represented as $\langle Q, v\rangle$. In a B&C algorithm, where all the constraints of the model are not considered in the LP_0, the only advantage by using this constraint family is that we simplify a vertex in the SEC representation. However, it has one important disadvantage, in the family (9) we might need to consider an SEC with

$|Q| > |V|/2$, while in the family (7c) it can be considered always a SEC such that $|Q| \leq |V|/2$. Therefore, we always consider the family (7c) regardless of whether it is given a depot or not in the cycle problem.

When a B&C is used to solve a cycle problem, the integrality constraints (7g) of the P_C^G are relaxed in order to first seek a solution that satisfies the rest of the constraints. Contrary to this strategy, Pferschy Staněk (2017) have recently considered again relaxing the SEC constraints in the TSP, to first solve the resulting problem to integer optimality with MILP-solvers and then introduce the SECs if required. Despite the improvement of the new MILP-solvers, this approach is still inferior compared to the opposite strategy. As a consequence of the continuous relaxation, a solution (y, x) that satisfies the rest of the constraints of (7) might still not belong to P_C^G. In these cases, instead of directly resorting to the branching phase to tighten the integrality gap, we could check if additional (not dominated by those in (7)) and facet-defining valid inequalities for the P_C^G are violated. The strength of considering additional valid inequalities was shown in the 1970s in the study of the TSP Grötschel Padberg (1979). In Bauer (1997) an extension of the clique trees inequality family (originally defined for the TSP) was given, which includes the so-called comb inequalities, for cycle problems. The shrinking rules proposed in Section 3 are safe for all the valid inequalities for P_C^G.

A polytope that it is closely related to P_C^G is the so-called lower cycle polytope, see Bauer (1997):

$$L_C^G = conv\{P_C^G, (0,0)\} \tag{10}$$

where $(0,0) \in \mathbb{R}^{V \times E}$ is the vector that represents that no vertex and edges of the graph are visited. It is easy to see, that for every graph G, so that it contains at least one cycle, there exist an infinity number of vectors $(y, x) \in L_C$ such that $x(E) < 3$. Hence, the polytope P_C^G is a proper subspace of L_C^G for every graph G that contains at least one cycle. It is crucial to consider the polytope L_C^G to obtain the shrinking results in Section 3.

In a B&C algorithm, it is reasonable to solve the separation problems of the valid inequality families following an order determined by their complexity. This order defines a hierarchy of the inequality families and their closure polytopes. We refer to the closure polytope of an inequality family as the polytope that satisfies all the inequalities of the given family and its preceding families in this hierarchy.

Without considering the variable bounds (7e)-(7f) and the inequality (7d), the simplest inequalities are the degree equations (7a) and the logical constraints (7b). These have, respectively, linear and quadratic exact algorithms in terms of the number of the vertices of G and generally are always included in the LP_0. The closure polytope of the inequalities (7a) and (7b) (the inequality (7d) is excluded to favour the convexity) turns out to be the undirected Assignment Polytope (with loops), P_A^G, which is defined as:

$$P_A^G := \{(y, x) \in \mathbb{R}^{V \times E} : (y, x) \text{ satisfies (7a), (7b), (7e), (7f)}\} \tag{11}$$

Next in the hierarchy comes the SEC family. A straightforward exact separation algorithm for the SECs has $O(|V|^4)$ time complexity (see Section 6.3 for further discussion) and its closure polytope is defined as:

$$P_{SEC}^G := \{(y, x) \in P_A^G : (y, x) \text{ satisfies (7c)}\} \tag{12}$$

Considering the relationship $P_C^G \subset P_{SEC}^G \subset P_A^G$, the underlying purpose of this paper is to effectively determine if a given solution $(y, x) \in P_A^G$ of a LP_0 belongs to P_{SEC}^G, or in case that it does not belong, to provide the violated inequalities.

Throughout the paper, we make use of the following well-known identity repeatedly. Given a graph G, a subset $S \subset V$ and a vector $x \in \mathbb{R}^E$, the identity

$$x(\delta(S)) = \sum_{v \in S} x(\delta(v)) - 2x(E(S)) \tag{13}$$

is always satisfied. In addition, if the vector $(y, x) \in \mathbb{R}^{V \times E}$ satisfies the degree constraints (7a), then the equations

$$x(\delta(S)) = 2y(S) - 2x(E(S)) \quad S \subset V \tag{14}$$

are satisfied by the vector (y, x). Particularly, the identity (14) is satisfied by every vector in P_{TSP}^G, P_C^G, P_{SEC}^G and P_A^G.

3 Shrinking for the Cycle Polytope

In this section, we present three shrinking rules that are safe for the P_C^G, i.e., rules that preserve the existence of violated cycle inequalities in the shrunk graph. In essence, we have generalized for every (simple) cycle problem the results obtained by Padberg Rinaldi (1990b) for Hamiltonian cycle problems. In the following lines, we formalize the concept of safe shrink for P_C^G and we prove the lemmas and the theorem in which shrinking rules for cycle problems are based on. In addition, we show that the three shrinking rules can be consecutively applied for the P_C^G.

Let us introduce the following notation. Given a graph $G = (V, E)$, the vector $(y, x) \in \mathbb{R}^{V \times E}$ and a subset $S \subset V$, we denote by $G[S] = (V[S], E[S])$ the graph obtained by shrinking the set S into a single vertex $s \notin V$, where the resulting set of vertices and edges are as follows:

$$V[S] = (V - S) \cup \{s\} \tag{15a}$$

$$E[S] = E(V - S) \cup \{[s, v] : v \in V - S, x(S : v) > 0\} \tag{15b}$$

and by $(y[S], x[S]) \in \mathbb{R}^{V[S] \times E[S]}$ we denote the vector with components

$$x[S]([u, v]) = x_{[u,v]} \qquad \forall [u, v] \in E \cap E[S] \tag{16a}$$

$$x[S]([s, v]) = x(S : v) \qquad \forall v \in V - S \tag{16b}$$

$$y[S](v) = y_v \qquad \forall v \in V \cap V[S] \tag{16c}$$

$$y[S](s) = x(\delta(S))/2 \tag{16d}$$

Let $Q \subset V$ be a subset of vertices, we denote with $Q[S]$ the subset derived by shrinking S

$$Q[S] = \begin{cases} (Q - S) \cup \{s\} & \text{if } S \cap Q \neq \emptyset \\ Q & \text{otherwise} \end{cases} \tag{17}$$

which has the following associated values:

$$y[S](Q[S]) = \begin{cases} y(Q) - y(Q \cap S) + \dfrac{x(\delta(S))}{2} & \text{if } S \cap Q \neq \emptyset \\ y(Q) & \text{otherwise} \end{cases} \tag{18a}$$

$$x[S](\delta(Q[S])) = \begin{cases} x(\delta(S \cup Q)) & \text{if } S \cap Q \neq \emptyset \\ x(\delta(Q)) & \text{otherwise} \end{cases} \tag{18b}$$

$$x[S](E(Q[S])) = \quad x(E(Q)) - x(E(Q \cap S)) \tag{18c}$$

Based on the definition given in Padberg Rinaldi (1990b) for safe shrinking for the P^G_{TSP}, an analogue definition can be formulated for safe shrinking for the P^G_C.

Definition 3.1. *Given a vector $(y, x) \notin P^G_C$, a set $S \subset V$ is safe to shrink if $(y[S], x[S]) \notin P^{G[S]}_C$.*

Note that the definition does not assume a one-to-one correspondence between the violated inequalities of (y, x) and $(y[S], x[S])$ (e.g. different violated cuts for (y, x) might overlap to the same violated cut for $(y[S], x[S])$). When a set S is safe to shrink for a given (y, x), it is also said that S is shrinkable for (y, x).

The definition of shrinkable set does not provide a practical tool for finding them. Hence, the first goal is to give a set of rules of shrinking for P^G_C, which are obtained in Theorem 3.6. The strategy used in Padberg Rinaldi (1990b) to obtain the shrinking rules for tours cannot be applied directly for simple cycles, because it relies on the fact that the tours visit every vertex in the graph. So, first we need to obtain the following lemma.

Lemma 3.2. *Let $(y, x) \in L^G_C$ be a vector. Suppose that $\{Q, \{u\}, \{v\}\}$ is a partition of V such that $x_{[u,v]} = x(u : Q) = x(v : Q) > 0$. Then any cycle τ of $\mathcal{C}^G$ that has a positive coefficient in the convex combination of (y, x), $\lambda_\tau > 0$, fulfills one of the following cases:*

(i) $V(\tau) \subset Q$

(ii) $|\tau \cap (u : Q)| = |\tau \cap (v : Q)| = |\tau \cap [u, v]| = 1$

Proof. Let $\mathcal{C}_{uv}$ denote the subset of cycles in $\mathcal{C}$ that visits the edge $[u, v]$ and has a positive value, $\lambda_\tau > 0$. Note that since $(y, x) \in L^G_C$, then $x_{[u,v]} \leq y_v$ and $x_{[u,v]} \leq y_u$. So, in order to satisfy the degree equations, every cycle τ in $\mathcal{C}_{uv}$ must contain at least an edge in $(u : Q)$ and $(v : Q)$. Moreover, since τ is a simple cycle, every $\tau \in \mathcal{C}_{uv}$ crosses exactly once $(u : Q)$ and $(v : Q)$. Now, let us see that if τ does not belong to $\mathcal{C}_{uv}$ and $\lambda_\tau > 0$, then τ is contained in Q. Consider the following inequality:

$$x_{[u,v]} = \sum_{\zeta \in \mathcal{C}_{uv}} \lambda_\zeta x^\zeta_{[u,v]} = \sum_{\zeta \in \mathcal{C}_{uv}} \lambda_\zeta = \sum_{\zeta \in \mathcal{C}_{uv}} \sum_{e \in (u:Q)} \lambda_\zeta x^\zeta_e \leq \tag{19a}$$

$$\sum_{\zeta \in \mathcal{C}_{uv}} \sum_{e \in (u:Q)} \lambda_\zeta x^\zeta_e + \sum_{\zeta \notin \mathcal{C}_{uv}} \sum_{e \in (u:Q)} \lambda_\zeta x^\zeta_e = x(u : Q) \tag{19b}$$

Since $x_{[u,v]} = x(u : Q)$, we have that $x^\tau_e = 0$ for every $e \in (u : Q)$. Similarly, we obtain that $x^\tau_e = 0$ for every $e \in (v : Q)$. Therefore, τ is contained in Q.

□

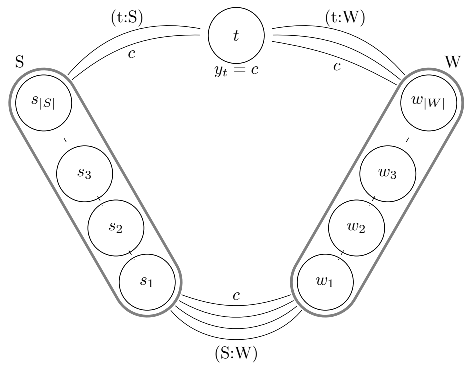


Figure 2: Illustration of the scenario in Lemma 3.3.

The next result generalizes the main theorem of shrinking in Padberg Rinaldi (1990b). The principal idea is to use a constant, c, to extend the rules of the original paper (where $\forall v \in V$ satisfies $y_v = 1$) for vertices that have fractional value. We also need an additional hypothesis about the vector $(y[W], x[W])$ obtained by shrinking the subset W, the "complement" of S, which is not required for the TSP because it is trivially satisfied by Hamiltonian cycles.

Lemma 3.3. *Given a vector $(y, x) \notin P^G_C$, let $\{S, W, \{t\}\}$ be a partition of V with $2 \leq |S|$ and c be a constant where $0 < c \leq 1$ such that:*

(i) $y_v = c \, \forall v \in S \cup \{t\}$

(ii) $x(E(S)) = c \cdot (|S| - 1)$

(iii) $x(t : S) = c$

(iv) $(y[W], x[W]) \in L^{G[W]}_C$

(v) No cycle in the convex combination of $(y[W], x[W])$ is contained in S

Then it is safe to shrink S for (y, x).

Proof. Based on the hypotheses i), ii) and iii) of the lemma and the identity (14) we obtain that $x(S : W) = c$ and $x(t : W) = c$, as illustrated in Figure 2.

Suppose for contradiction that S is not shrinkable, so $(y[S], x[S]) \in P^{G[S]}_C$. Since $x_{[s,t]} = x(s : W) = x(t : W)$, based on Lemma 3.2, the vector $(y[S], x[S])$ can be written as:

$$(y[S], x[S]) = \sum_{\zeta \in \mathcal{W}_s} \alpha_\zeta (y, x)^\zeta + \sum_{\zeta \in \mathcal{W}_0} \alpha^0_\zeta (y, x)^\zeta \tag{20}$$

where $\mathcal{W}_s$ is the set of cycles visiting the shrunk vertex s having $\alpha_\zeta > 0$ and $\mathcal{W}_0$ is the set of cycles contained in W having $\alpha^0_\zeta > 0$. Note that $\mathcal{W}_0$ might be an empty set. The coefficients satisfy $\sum_{\zeta \in \mathcal{W}_s} \alpha_\zeta + \sum_{\zeta \in \mathcal{W}_0} \alpha^0_\zeta = 1$.

By hypothesis the vector $(y[W], x[W])$ belongs to $L^{G[W]}_C$, so $(y[W], x[W])$ can be written as a convex combination of cycles of $\mathcal{C}_{G[W]}$ and the vector $(0, 0)$. Because of the Lemma 3.2 and by the hypothesis v) the vector $(y[W], x[W])$ can be written as:

$$(y[W], x[W]) = \sum_{\eta \in \mathcal{S}_w} \beta_\eta (y, x)^\eta + \beta_{(0,0)}(0, 0) \tag{21}$$

where $\mathcal{S}_w$ is the set of cycles visiting w (the vertex to which W is contracted to) having $\beta_\eta > 0$, $\beta_{(0,0)} \geq 0$ and $\sum_{\eta \in \mathcal{S}_w} \beta_\eta + \beta_{(0,0)} = 1$.

Now, considering $x(t:s) = x(t:w) = c$ we have that:

$$c = \sum_{\zeta \in \mathcal{W}_s} \alpha_\zeta = \sum_{\eta \in \mathcal{S}_w} \beta_\eta \tag{22}$$

and from the fact that the coefficients sum up to one, we have that:

$$1 - c = \sum_{\eta \in \mathcal{W}_0} \alpha^0_\eta = \beta_{(0,0)} \tag{23}$$

To prove the lemma we follow the "patch-and-weight" strategy used in Padberg Rinaldi (1990b) for the P^G_{TSP} whose goal is to reconstruct the cycles and coefficients of the convex combination of the vector (y, x). According to the vertices in W, we can partition $\mathcal{W}_s$ into $|W|$ pairwise disjoint subsets (some of them which be empty). For $j \in \{1, \ldots, |W|\}$ let us call $\mathcal{W}^j_s$ the subset of cycles in $\mathcal{W}_s$ containing the edge $[s, w_j]$, and denote by $\zeta^j_1, \ldots, \zeta^j_{k_j}$ the cycles of $\mathcal{W}^j_s$ and by $\beta^j_1, \ldots, \beta^j_{k_j}$ their coefficients in the convex combination. In the same way, we can partition $\mathcal{S}_w$ into $|S|$ subsets calling $\mathcal{S}^i_w$ the subset of cycles in $\mathcal{S}_w$ containing the edge $[s_i, w]$. We denote by $\eta^i_1, \ldots, \eta^i_{h_i}$ the cycles of $\mathcal{S}^i_w$ and by $\alpha^i_1, \ldots, \alpha^i_{h_i}$ their coefficients in the convex combination.

The cycles of the convex combination of (y, x) are constructed in two steps. In the first step, $|\mathcal{S}_w|$ copies of each cycle in $\mathcal{W}_s$ are created. With this goal, for each $j \in \{1, \ldots, |W|\}$ and for each $l \in \{1, \ldots, k_j\}$, create $|S|$ copies of the cycle ζ^j_l, and denote them by $\{\tau^{ij}_l\}$ for $i \in \{1, \ldots, |S|\}$. Then, for each $j \in \{1, \ldots, |W|\}$, for each $l \in \{1, \ldots, k_j\}$ and for each $i \in \{1, \ldots, |S|\}$ create h_i copies of τ^{ij}_l, and denote them by $\{\tau^{ij}_{ml}\}$ for $m \in \{1, \ldots, h_i\}$. At this point we have $|\mathcal{W}_s| \cdot |\mathcal{S}_w|$ cycles that belong to $G[S]$. In the second step, these cycles of $G[S]$ are extended to cycles of G. To that end, consider each cycle τ^{ij}_{ml} and remove the edges $[t, s]$ and $[s, w_j]$ and join the resulting path with the path in $G[W]$ obtained from the cycle η^i_m by removing the edges $[w, t]$ and $[s_i, w]$, and add the edge $[s_i, w_j]$ to obtain the extension of τ^{ij}_{ml} to G.

The coefficients of the constructed τ^{ij}_{ml} cycles are defined in the following way:

$$\lambda^{ij}_{ml} = \frac{x_{[s_j, w_i]} \cdot \alpha^j_l \cdot \beta^i_m}{\sum_{r=1}^{k_j} \alpha^j_r \cdot \sum_{r=1}^{h_i} \beta^i_r} \tag{24}$$

where $i \in \{1, \ldots, |S|\}$, $j \in \{1, \ldots, |W|\}$, $m \in \{1, \ldots, h_i\}$ and $l \in \{1, \ldots, k_j\}$. It can be verified that the coefficients defined this way sum c in total:

$$\sum_{i,j,m,l} \lambda^{ij}_{ml} = \sum_{i,j} x_{[s_j, w_i]} \sum_{m,l} \frac{\alpha^j_l \cdot \beta^i_m}{\sum_{r=1}^{k_j} \alpha^j_r \cdot \sum_{r=1}^{h_i} \beta^i_r} = \sum_{i,j} x_{[s_j, w_i]} = x(S:W) = c \tag{25}$$

Then the vector (y, x) can be obtained as a convex combination of the cycles in $\mathcal{W}_0$ and $\{\tau^{ij}_{ml}\}$ with coefficients $\{\alpha^0_\zeta\}$ and $\{\lambda^{ij}_{ml}\}$, respectively. We conclude $(y, x) \in P^G_C$ which is a contradiction.

□

The lemma gives a sufficient condition for a set to be shrinkable, but still it is not practical. The next theorem gives three practical scenarios to make use of Lemma 3.3. Beforehand, let us obtain a useful result for L^G_C. Consider the undirected version of the Assignment Polytope (without loops) P^1_A defined as:

$$P^1_A := \{(y, x) \in \mathbb{R}^{V \times E} : (y, x) \text{ satisfies (7a), (7b), (7f), } y = 1\} \tag{26}$$

It is a well-known result of the literature that $P^G_{TSP} = P^1_A$ for $3 \leq |V| \leq 5$ (see Grötschel Padberg (1979)). This relationship is the key to obtaining the shrinking rules for the P^G_{TSP} in Padberg Rinaldi (1990b). So, we would like to obtain a similar result for L^G_C and P^G_A. However, $L^G_C \neq P_A$ when $4 \leq |V|$, as shown in the counterexample of Figure 3. The vector defined in the figure belongs to P^G_A, but it does not belong to L^G_C, because it cannot be expressed as a convex combination of cycles.

Nevertheless, we have the following lemma which is enough to prove Theorem 3.6.

Lemma 3.4. *Let $G = (V, E)$ be a graph and c be a constant such that $3 \leq |V| \leq 5$ and $0 < c \leq 1$. If $(y, x) \in P_A$ such that $y_v = c$ for all $v \in V$, then $(y, x) \in L_C$.*

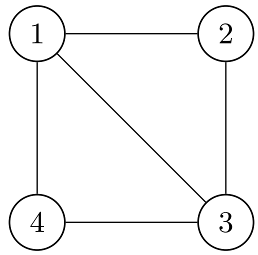


Figure 3: An example of a solution that belongs to P_A^G but not to L_C^G when $|V| = 4$ (it can be easily extended for $|V| \geq 4$ by means of subdivisions). All the edges in the figure have value $\frac{1}{2}$. The values of the vertices satisfy the degree equations.

Proof. It is straightforward that if $(y, x) \in P_A$ such that $y_v = c$ for all $v \in V$, then $\frac{1}{c}(y, x) \in P_A^1$. By the classical result in Grötschel Padberg (1979), since $3 \leq |V| \leq 5$, the equality $P_A^1 = P_{TSP}$ is satisfied. Since P_{TSP}^G is contained in L_C^G, the vector $\frac{1}{c}(y, x)$ belongs to L_C^G. Then, since both $(0, 0)$ and $\frac{1}{c}(y, x)$ belong to L_C^G, which is convex, and $0 \leq c \leq 1$ we have that $(y, x) \in L_C$.

□

Lemma 3.5. *Given a graph G such that $|V| = 5$, a vector $(y, x) \in L_C^G$ and $0 \leq c \leq 1$, suppose that $\lambda_{(0,0)} = 1 - c$. Let $\{S, \{t\}, \{w\}\}$ be a partition of V such that $x_{[t,w]} = x(t : S) = x(w : S) = c$, then every cycle τ in $\mathcal{C}^G$ such that $\lambda_\tau > 0$ is not contained in S.*

Proof. Since $\{S, \{t\}, \{w\}\}$ is a partition of V, we have that $|S| = 3$ and $|V - S| = 2$. Hence, every cycle in $\mathcal{C}^G$ has vertices in S. According to the number of visited vertices of S, we can partition $\mathcal{C}^G$ into 3 subsets $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$. Furthermore, the set $\mathcal{C}_3$ can be partitioned into two subsets, $\mathcal{C}_3^{in}$ and $\mathcal{C}_3^{out}$, determined by whether the cycles are fully contained in S or not. Since (y, x) belongs to L_C^G, there is a convex combination of cycles of $\mathcal{C}^G$ whose coefficients satisfy

$$\sum_{\tau \in \mathcal{C}_1} \lambda_\tau^1 + \sum_{\tau \in \mathcal{C}_2} \lambda_\tau^2 + \sum_{\tau \in \mathcal{C}_3^{out}} \lambda_\tau^{3o} + \sum_{\tau \in \mathcal{C}_3^{in}} \lambda_\tau^{3i} + \lambda_{(0,0)} = 1 \tag{27}$$

Since the cycles in $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3^{out}$ have edges in $(t : S)$ and $(w : S)$, by the Lemma 3.2, each cycle has exactly one edge in the mentioned edge sets. Now, consider the hypothesis that $x(t : S) = c$ (or $x(w : S) = c$), so the coefficients also satisfy the following identity:

$$\sum_{\tau \in \mathcal{C}_1} \lambda_\tau^1 + \sum_{\tau \in \mathcal{C}_2} \lambda_\tau^2 + \sum_{\tau \in \mathcal{C}_3^{out}} \lambda_\tau^{3o} = c \tag{28}$$

By hypothesis, we have that $\lambda_{(0,0)} = 1 - c$ and by (27) and (28), we obtain that $\lambda_\tau^{3i} = 0$ for all $\tau \in \mathcal{C}_3^{in}$, which means that every cycle in $\mathcal{C}^G$ contained in S has null coefficient.

□

Theorem 3.6 (Rules C1, C2 and C3)**.** *Given a vector $(y, x) \notin P_C^G$, let $S \subset V$ with $2 \leq |S| \leq 3$, $t \in V - S$ and $0 < c \leq 1$ be such that:*

(i) $y_v = c \; \forall v \in S \cup \{t\}$

(ii) $x(E(S)) = c \cdot (|S| - 1)$

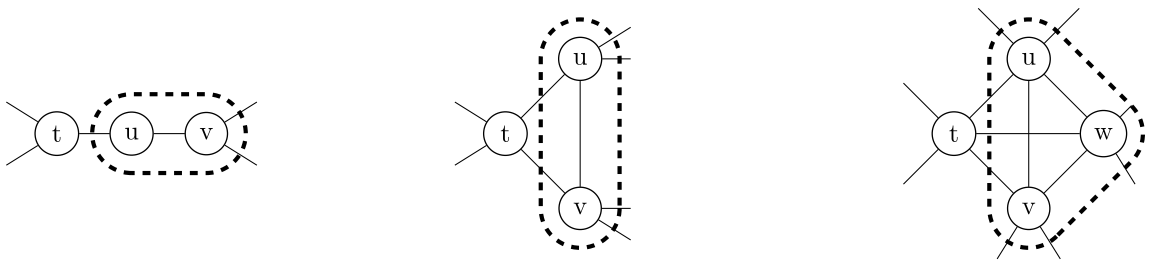
(iii) $x(t : S) = c$

Then it is safe to shrink S for (y, x).

Proof. Let $W = V - (S \cup \{t\})$ be a subset of V. If the hypotheses are satisfied, note that W is non-empty. Since $2 \leq |S| \leq 3$, we have that $4 \leq |V[W]| \leq 5$. Notice that, $y_v = c$ for all the vertices of $V[W]$ and $(y[W], x[W]) \in P_A^{G[W]}$. Under these hypotheses, by Lemma 3.4, the vector $(y[W], x[W])$ belongs to $L_C^{G[W]}$. When $|S| = 2$, it does not exist any cycle contained in S. When $|S| = 3$, as a consequence of Lemma 3.5, we have that it does not exist a cycle in the convex combination of $(y[W], x[W])$ contained in S. Therefore, the hypotheses of Lemma 3.3 are satisfied and S is shrinkable.

□

From Theorem 3.6, three shrinking rules can be derived, which are summarized in Figure 4: the rules C1 and C2 correspond to the case $|S| = 2$ and the rule C3 to $|S| = 3$.



$y_u = y_v = y_t = c$
$x_{[u,v]} = c$
$x_{[t,u]} = c$
Rule C1

$y_u = y_v = y_t = c$
$x_{[u,v]} = c$
$x_{[t,u]} + x_{[t,v]} = c$
Rule C2

$y_u = y_v = y_w = y_t = c$
$x_{[u,v]} + x_{[u,w]} + x_{[v,w]} = c$
$x_{[t,u]} + x_{[t,v]} + x_{[t,w]} = c$
Rule C3

Figure 4: Illustration of the three shrinking rules derived from the Theorem 3.6

It is easy to see that rule C2 dominates the rule C1, in fact it is just a particular case of it. The reason to split them, is that the cost of checking C1 is lower than the cost of C2. By contrast, rule C3 is not dominated by the rules C1 and C2. In Figure 5, an example is given of a vector $(y, x) \in P_A$ in which rule C3 can be applied but not C1 and C2. For instance, if we consider $S = \{1, 2, 3\}$, $W = \{4, 5, 6\}$ and $t = 7$, then S is shrinkable by rule C3. Since the vertices and edges have different values, there is no shrinkable set that can be identified by rule C1 or C2.

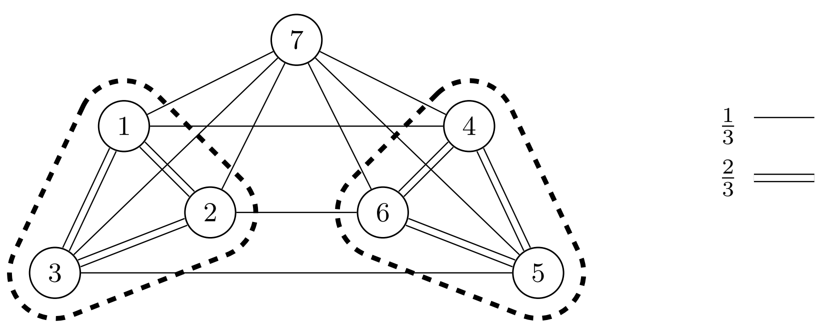


Figure 5: Example of a pair G and $(y, x) \in P_A$ where rule C3 can be applied but not rules C1 nor C2. The values of the edges are the ones detailed in the legend and all the vertices have value 1.

A useful property of the rules derived from Theorem 3.6 is that the value of the vertices is inherited in the shrunk graphs.

Lemma 3.7. *Under the hypotheses of Theorem 3.6, $y[S](v[S]) = y_v$ for all $v \in V$.*

Proof. For every $v \in V - S$, we have $y[S](v[S]) = y_v$ by definition. Since $2y_s = x(\delta(S)) = 2y_v$ for $v \in S$ we obtain the result of the lemma.

□

In the preprocess of separation algorithms, it is desirable to perform multiple consecutive safe shrinkings. For that aim, we need to analyse what happens with the hypotheses of Theorem 3.6 after the contraction of a shrinkable set. More precisely, we need to see when the shrunk vector belongs to P_A^G.

Lemma 3.8. *Let S be a shrinkable set for $(y,x) \in P_A^G$ obtained from Theorem 3.6 using the $\{S, W, \{t\}\}$ partition. Then, $(y[S], x[S])$ satisfies the degree equations and the logical constraints associated with every edge in $E(W) \cup (t : V)$. In addition, we have either*

i) $(y[S], x[S]) \in P_A^{G[S]}$, *or*

ii) $\exists w \in W$ *such that* $y_w < y_s$ *and* $y_w < x_{[w,s]} \le y_s$

Proof. From the definition of the shrunk vector, it is clear that $(y[S], x[S])$ satisfies the degree equations. Since $v \in S$ satisfies $y_v \le 1$, $y_s = y_v$ also satisfies $y_s \le 1$. Moreover, $x_{[t,s]} = y_s = y_t$. If $x(w : S) \le y_w$ for all $w \in W$ then $(y[S], x[S])$ satisfies the logical constraints and $(y[S], x[S]) \in P_A^{G[S]}$. If the previous is not true, there exists a vertex $w \in W$ such that $x(w : S) > y_w$ and $y_w < y_s$ (because by hypothesis $(y, x) \in P_A^G$). Therefore, the logical constraint $x_{[w,s]} \le y_w$ is violated for $(y[S], x[S])$ by a vertex $w \in W$ such that $y_w < y_s$.

□

There are two scenarios where the shrunk vector always belongs to P_A^G. First, when all the vertices of V have the same y value, as is the case when $(y, x) \in P_{TSP}^G$, and secondly, when only rule C1 is applied. The next theorem shows that if $(y, x) \in P_A^G$, it is possible to shrink a subset S obtained by the rules of Theorem 3.6 and continue with further safe shrinkings regardless of whether or not $(y[S], x[S])$ belongs to $P_A^{G[S]}$.

Theorem 3.9. *Given a vector $(y, x) \in P_A^G$, it is safe to consecutively apply the shrinking rules derived from Theorem 3.6.*

Proof. Let S be a subset obtained from Theorem 3.6 such that $(y[S], x[S]) \notin P_A^{G[S]}$. By Lemma 3.8 we know that the only violated logical constraints of $(y[S], x[S])$ consist of edges whose vertices, s and $v \in W$, have different values $y_v < y_s$. Notice that in the proof of Theorem 3.6 the hypothesis that the logical constraints are satisfied is used twice. First in Lemma 3.2, which is applied for vertices having the same value. Secondly in Theorem 3.6, where it is assumed $(y[N], x[N]) \in P_A^{G[N]}$ for a given subset N of $V[S]$. In order to see that this last hypothesis is always satisfied by every shrinkable set candidate, let us suppose that $\{M, N, \{r\}\}$ is a partition of $V[S]$ that satisfies hypotheses i), ii) and iii) of Theorem 3.6. Then there are two possible cases: $v \in M \cup \{r\}$ and $s \in N$, or vice versa. The hypothesis $(y[N], x[N]) \in P_A^{G[N]}$ is satisfied in both cases, because $x_{[n,v]} \le y_n = y_u$ for $u \in M \cup \{r\}$.

□

Another interesting scenario occurs when there is at least a vertex $v \in V$ satisfying $y_v = 1$, as happens in the context of cycle problems with depot. In all these problems, the case ii) of Lemma 3.8 has a special meaning as shown in Theorem 3.11.

Lemma 3.10. *If $(y, x) \in \mathbb{R}^{V \times E}$ satisfies the degree equations* (7a) *and $u, v \in V$ are two vertices such that $x_{[u,v]} > y_u$ then $x(\delta(\{u, v\})) < 2y_v$.*

Proof. As (y, x) satisfies the degree equations:

$$2y_u < 2x_{[u,v]} = 2y_u + 2y_v - x(\delta(\{u, v\})) \tag{29}$$

□

Theorem 3.11. *Given a vector $(y, x) \in P_A^G$, let $O = \{v \in V : y_v = 1\}$ be the subset of vertices with value equal to one and S be a shrinkable set for (y, x) obtained from Theorem 3.6 such that $O - S \neq \emptyset$. Then, we have either*

i) $(y[S], x[S]) \in P_A^{G[S]}$, *or*

ii) $\exists w \in V - S$ *such that, for every* $u \in S$ *and* $v \in O - S$, *the SEC* $\langle S \cup \{w\}, u, v \rangle$ *is violated by* (y, x).

Proof. Note that, in the case ii) of Lemma 3.8, the vertex $w \in V - S$ cannot be contained in O because $y_w < 1$. Now, as a consequence of Lemma 3.10 we can rewrite the second case.

□

4 Safe Shrinking Rules for the Subcycle Closure Polytope

Depending on the inequality, more aggressive contractions can be employed as a preprocess of separation algorithms. In the TSP, for the subtour separation problem, Crowder Padberg (1980) introduced subtour specific shrinking rules to simplify the support graphs before proceeding with the separation algorithms. With the aim of motivating the concepts in the subcycle-safe shrinking procedure, let us prove the following result.

Lemma 4.1. *Given a vector* $(y, x) \in P_A^G$ *and an edge* $e \in E$*, let* $S = V(e)$ *be the subset associated with the edge* e*. If* $(y[S], x[S]) \in P_{SEC}^{G[S]}$*, then either*

i) $(y, x) \in P_{SEC}^G$*, or*

ii) every violated SEC $\langle Q, r, t\rangle$ *for* (y, x) *satisfies* $S \cap Q \neq \emptyset$ *and* $S - Q \neq \emptyset$

Proof. Let $e = [u, v]$ be the given edge and $\langle Q, r, t\rangle$ be a SEC for (y, x) such that $S \subset Q$ (or $S \subset V - Q$). On the one hand, since $(y, x) \in P_A^G$, we have $y[S](u[S]) \geq y_u$ and $y[S](v[S]) \geq y_v$. On the other hand, $x[S](\delta(Q[S])) = x(\delta(Q))$ by definition. Then the SEC $\langle Q[S], r[S], t[S]\rangle$ for $(y[S], x[S])$, is at least as violated as $\langle Q, r, t\rangle$ for (y, x). So if $(y[S], x[S]) \in P_{SEC}^{G[S]}$, and $(y, x) \notin P_{SEC}^G$, the only violated SECs for (y, x) are associated with subsets that separate u and v.

□

Recall that we want to search the violated SECs for a vector $(y, x) \in P_A^G$, which has been obtained from the LP_0 subproblem. Let us assume that we have defined a first shrinking rule that contracts edges by avoiding the scenario ii) of Lemma 4.1. So if $(y, x) \notin P_{SEC}^G$, as a consequence of the lemma, $(y, x) \notin P_{SEC}^{G[S]}$. In this case, the vector $(y[S], x[S])$ does not belong to the closure of SECs because either there exists violated logical constraints, SECs or both. Let us suppose that we have a second shrinking rule that identifies (and saves) the violated logicals and "fixes" them. Repeatedly applying the second rule, we will eventually reach a vector that satisfies the logical constraints. Now, we are in a similar situation to the starting point, so we can try with the first rule again and so on. This is the main idea exploited in the subcycle-safe shrinking process.

Definition 4.2. *Given a vector* $(y, x) \in \mathbb{R}^{V \times E}$ *that satisfies the degree equations, a set* $S = \{u, v\} \subset V$ *is subcycle-safe to shrink if at least one of the following conditions is satisfied:*

i) $(y[S], x[S]) \notin P_{SEC}^{G[S]}$*, or*

ii) if there exist violated logical constraints for (y, x)*, these are associated with the edge* $[u, v]$

Note that the second condition does not require the existence of violated logical constraints for (y, x), which enables the subcycle-safe shrinkable set definition for vectors (y, x) in P_{SEC}^G to be used. Furthermore, this condition means: if we have already found a violated constraint, we should not worry if later the shrinking the vector is projected to the subcycle closure polytope, since we have already achieved the goal of the separation problem.

In some sense, from Theorem 4.4 we derive the first shrinking rule of the motivation above and from Theorem 4.5 the second shrinking rule. The condition that avoids the case ii) of the Lemma 4.3 is the hypothesis $x_{[u,v]} \geq \max\{y_u, y_v\}$ in the theorems. Actually, the hypothesis that $(y, x) \in P_A^G$ of the first rule can be replaced with the hypothesis that all the logical constraints associated with vertices u and v (excluding the one with $[u, v]$) are satisfied, which is a consequence of the hypothesis $x_{[u,v]} \geq \max\{y_u, y_v\}$. Let us address the next lemma as an intermediate step.

Lemma 4.3. *Given a vector* $(y, x) \in \mathbb{R}^{V \times E}$ *that satisfies the degree equations, let* $S = \{u, v\} \subset V$ *be a subset such that* $x_{[u,v]} \geq \max\{y_u, y_v\}$*. Then, if* $(y, x) \notin P_A^G$*, at least one of the following conditions is satisfied:*

i) $(y[S], x[S]) \notin P_A^{G[S]}$*, or*

ii) if there exist violated logical constraints for (y, x)*, these are associated with the edge* $[u, v]$

Proof. On the one hand, since $x(\{u, v\} : w) \geq x_{[u,w]}$ and $x(\{u, v\} : w) \geq x_{[v,w]}$ for all $w \in V - \{u, v\}$, every violated logical constraint for (y, x) associated with the vertices in $V - \{u, v\}$ can be adapted to violated constraints for $(y[S], x[S])$. On the other hand, since $x_{[u,v]} \geq \max\{y_u, y_v\}$ and the degree equations are satisfied, we have that $x_{[u,w]} \leq y_u$ and $x_{[v,w]} \leq y_v$ for all $w \subset V - \{u, v\}$. Therefore, if $(y[S], x[S]) \in P_A^{G[S]}$, the only possible violated logical constraints associated with the vertices of S correspond with the edge $[u, v]$.

□

The SEC inequalities (7c) are defined for sets, Q, such that $3 \leq |Q| \leq |V| - 3$. However, if $\langle Q, u, v \rangle$ violates for (y, x) the inequality of (7c) but $|Q| = 2$ or $|Q| = |V| - 2$, then a violated logical constraint can be identified and therefore we also know that $(y, x) \notin P^G_{SEC}$. For instance, if $\langle \{u, w\}, u, v \rangle$ does not satisfy the inequality (7c), then $y_w < x_{uw}$ is a violated constraint. In the following proofs, the term violated SEC, embracing the cases $|Q[S]| = 2$ and $|Q[S]| = |V[S]| - 2$, refers to its associated violated logical constraint when required.

Theorem 4.4 (Rule S1)**.** *Given a vector $(y, x) \in \mathbb{R}^{V \times E}$ that satisfies the degree equations, let $u, v \in V$ be two vertices such that $x_{[u,v]} = y_u = y_v = c$. If there exists a vertex $w \in V - \{u, v\}$ such that $y_w \geq c$, then it is subcycle-safe to shrink $S = \{u, v\}$.*

Proof. Assume the vector (y, x) belongs to P^G_A, i.e., only violated SECs exists for (y, x), otherwise the theorem is satisfied by Lemma 4.3. Let $\langle Q, r, t \rangle$ be a violated SEC for (y, x), and without loss of generality, suppose that $S \cap Q \neq \emptyset$. The goal is to see that for a violated SEC for (y, x), there is a violated SEC for $(y[S], x[S])$.

First, let us suppose that $S \subset Q$, where $x[S](\delta(Q[S])) = x(\delta(Q))$ is satisfied by definition. The only case that is needed to check is when $r \in S$. Without loss of generality, suppose that $r = v$. By hypothesis $y_u = x_{[u,v]}$, so $2y_v = x(\delta(S)) = 2y[S](v)$ and $\langle Q[S], y[S](s), y[S](r) \rangle$ define the desired SEC for $(y[S], x[S])$.

$$x[S](\delta(Q[S])) = x(\delta(Q)) < 2y_v + 2y_t - 2 = 2y[S](s) + 2y[S](t) - 2 \tag{30}$$

Next, let us analyze the case $S \cap Q \neq \emptyset$ and $Q - S \neq \emptyset$. Without loss of generality, suppose that $u \in Q$ and $v, w \in V - Q$. The subcase that requires a special attention is when $r = u$ and $t = v$. Note that, since (y, x) satisfies the degree equations and, also by hypothesis, $y_v = x_{[u,v]}$, we have that $x(v : V - Q) \leq x(v : Q)$, and therefore:

$$\begin{aligned} x[S](\delta(Q[S])) &= x(\delta(Q \cup S)) && \text{(31a)} \\ &= x(\delta(Q)) + x(\delta(v)) - 2x(v : Q) && \text{(31b)} \\ &= x(\delta(Q)) + x(v : V - Q) - x(v : Q) \leq x(\delta(Q)) && \text{(31c)} \\ &< 2y_r + 2y_v - 2 = 2y_w + 2y_t - 2 = 2y[S](r) + 2y[S](w) - 2 && \text{(31d)} \end{aligned}$$

Hence, there also exists a violated SEC (or logical constraint) for $(y[S], x[S])$ and the set S is subcycle-safe to shrink.

□

Clearly, the shrinking rule S1 dominates the rules C1 and C2 of Theorem 3.6. For every scenario where rules C1 or C2 can be applied, rule S1 is also applicable, since the existence of $w \in V - \{u, v\}$ is determined by the vertex $t \in V - \{u, v\}$ in Theorem 3.6. Moreover, rule C3 should not be combined with rule S1, since might exist vertices with the same y value whose connecting edge has a greater value in the shrunk graph obtained by S1.

Theorem 4.5 (Rule S2)**.** *Given a vector $(y, x) \in \mathbb{R}^{V \times E}$ that satisfies the degree equations, let $u, v \in V$ be two vertices such that $x_{[u,v]} > \max\{y_u, y_v\}$ then it is subcycle-safe to shrink $S = \{u, v\}$.*

Proof. The theorem is a direct consequence of Lemma 4.3.

□

Note that, if $(y, x) \in P^G_A$ and S is a shrinkable set obtained from Theorem 3.6, then by Lemma 3.8 we have that $x_e \leq \max\{y_u, y_v\}$ for every $e = [u, v] \in E[S]$. Hence, it only makes sense to use the rule S2 in combination with the rule S1.

If a subcycle-safe rule is applied, we know that all the SECs have not vanished. However, new violated SECs for $(y[S], x[S])$ might have appeared, which cannot be adapted to a violated one for (y, x). This situation would lead to identifying unnecessary cuts for (y, x) and therefore to slowing down the separation algorithm (the cut generation part). It is reasonable to ask when the violated SECs for $(y[S], x[S])$ can be transformed to violated SECs for (y, x) and when not. Let us define the mapping by $\pi_S : \mathcal{P}(V[S]) \to \mathcal{P}(V)$

$$\pi_S(Q) = \begin{cases} Q - \{s\} \cup S & \text{if } s \in Q \\ Q & \text{otherwise} \end{cases} \tag{32}$$

For a given S, the inverse, π_S^{-1}, of the mapping π_S is the set shrinking defined in (17), i.e., $\pi_S^{-1}(Q) = Q[S]$. We have that $Q = \pi_S^{-1}(\pi_S(Q))$ for all $Q \subset V[S]$ and $Q \subset \pi_S(\pi_S^{-1}(Q))$ for all $Q \subset V$. An important property of the mapping

π_S, by the definition (18c), is that $x(\delta(\pi_S(Q))) = x[S](\delta(Q))$ for all $Q \subset V[S]$. In some cases, we will need to refer to the set obtained by unshrinking completely the contracted sets, where multiple shrinking might have been performed, e.g., $G[S_1][S_2]$. In such cases, we simplify the notation and denote $\pi(Q)$, e.g., $\pi(Q) = \pi_{S_1}(\pi_{S_2}(Q))$.

When an inequality family is targeted in a separation problem, knowing the representation of such inequalities, as is the case for the SECs, is very valuable to study how an inequality is transformed when shrinking and unshrinking a set. Moreover, since $x(\delta(\pi_S(Q))) = x[S](\delta(Q))$ for all $Q \subset V[S]$, understanding the relationship between y and $y[S]$ values is the key point to see how the violated SEC inequalities behave under the different shrinking rules.

Lemma 4.6. *Given a vector $(y, x) \in \mathbb{R}^{V \times E}$ that satisfies the degree equations and a subset $S = \{u, v\}$ of V. The following holds:*

i) $y[S](v[S]) > y_v$ *if* $x_{[u,v]} < y_u$

ii) $y[S](v[S]) < y_v$ *if* $x_{[u,v]} > y_u$

iii) $y[S](v[S]) = y_v$ *if* $x_{[u,v]} = y_u$

Proof. It is a consequence of the definition of $y[S]$ and the identity (14).

□

Lemma 4.7. *Under the hypotheses of Theorem 4.4, $y[S](v[S]) = y_v$ for all $v \in V$.*

Proof. For every $v \in V - S$, we have $y[S](v[S]) = y_v$ by definition. For $u, v \in S$, since $y_u = y_v = x_{[u,v]}$, we obtain the equality by Lemma 4.6.

□

Lemma 4.8. *Let G be an undirected graph, $(y, x) \in \mathbb{R}^{V \times E}$ be a vector and a vertex subset $S \subset V$. Suppose that $y[S](u) \leq y(v)$ for all $u \in V[S]$ and $v \in \pi_S(u)$. Then, for each SEC for $(y[S], x[S])$ there exists at least one SEC as violated as it for (y, x).*

Proof. Note that, if $r \in Q$ and $t \notin Q$ then $u \in \pi_S(Q)$ and $v \notin \pi_S(Q)$ for all $u \in \pi_S(r)$ and $v \in \pi_S(t)$. Let $\langle Q, r, t\rangle$ be a SEC inequality violated by $(y[S], x[S])$. Therefore, the SEC inequality $\langle \pi_S(Q), u, v\rangle$ is violated by (y, x) where $u \in \pi_S(r)$ and $v \in \pi_S(t)$.

$$x(\delta(\pi_S(Q))) - 2y_u - 2y_v \leq x[S](\delta(Q)) - 2y[S](r) - 2y[S](t) \quad u \in \pi_S(r) \text{ and } v \in \pi_S(t) \tag{33}$$

□

Corollary 4.9. *Let G be an undirected graph and $(y, x) \in \mathbb{R}^{V \times E}$ be a vector. If S is a shrinkable subset obtained by rules C1, C2, C3 or S1, then $(y, x) \notin P_{SEC}^{G}$ if and only if $(y[S], x[S]) \notin P_{SEC}^{G[S]}$.*

Proof. It is a consequence of Lemma 3.7 and Lemma 4.7.

□

When rule S2 is applied, as a consequence of Lemma 4.6, some vertices of the shrunk graph will have lower values than the original ones. Although, by the definition of subcycle-safe shrinking, all the violated SECs for (y, x) are not vanished, we might lose some of them in the shrinking process. However, it could be interesting to identify and save those excluded violated SECs if possible. For that aim we consider a vector $m[S] \in \mathbb{R}^{V[S]}$ defined as $m[S](v) = \max\{y_u : u \in \pi_S(v)\}$. It is clear that if only the rules of Theorem 3.6 and the rule S1 are applied, $m[S](v) = y[S](v)$ for all $v \in V[S]$. Considering the vector $m[S]$, we evaluate a SEC $\langle Q, u, v\rangle$ for a given vector $(y[S], x[S])$ by the expression

$$x[S](\delta(Q)) - 2m[S](u) - 2m[S](v) \geq -2 \tag{34}$$

and only if this is violated, we save the SEC $\langle Q, u, v\rangle$ for (y, x).

5 Exact Separation Algorithms for SECs

In this section, we present two exact separation algorithms for SECs in cycle problems. Given a vector $(y, x) \in P_A^G$, an algorithm which finds violated SECs for (x, y) is called a separation algorithm. A separation algorithm is called exact if it always finds violated inequalities when they exist, otherwise it is called heuristic. Let $\bar{G} = (\bar{V}, \bar{E})$ be the support graph of the given vector (y, x) where

$$\bar{V} := \{v \in V : y_v > 0\} \tag{35a}$$

$$\bar{E} := \{e \in E : x_e > 0\} \tag{35b}$$

Before delving into the separation algorithms in depth, we need to make an observation which has important consequences for SEC separation problems in cycle problems. In the TSP, the y values are fixed to 1, so the constraints in the family (7c) only depend on the star-set value of subsets of vertices. For this reason, the SEC separation problem for the TSP is closely related with the minimum cut problem, particularly, the most violated SEC for (y, x) is in correspondence with the global minimum cut of $\bar{G}$. However, in cycle problems in general, the SECs $\langle C, v, d\rangle$ obtained from the global minimum cut of $\bar{G}$, $x(C : V - C)$, might not be violated, although other violated SECs for (y, x) can exist. This scenario is shown in the example in Figure 6. The global minimum cut in the figure is obtained by $C = \{4\}$ and because $|C| < 3$, by definition (7c), there is no violated SEC inequality of type $\langle C, v, u\rangle$ (or equivalently of type $\langle V - C, v, u\rangle$). However, the SECs $\langle\{2, 3, 8\}, 2, 6\rangle$ (or $\langle\{1, 4, 5, 6, 7, 9\}, 6, 2\rangle$), $\langle\{2, 3, 4, 8\}, 2, 6\rangle$ (or $\langle\{1, 5, 6, 7, 9\}, 6, 2\rangle$) and $\langle\{2, 3, 4, 5, 8\}, 2, 6\rangle$ (or $\langle\{1, 6, 7, 9\}, 6, 2\rangle$) are violated for the vector (y, x) represented in Figure 6.

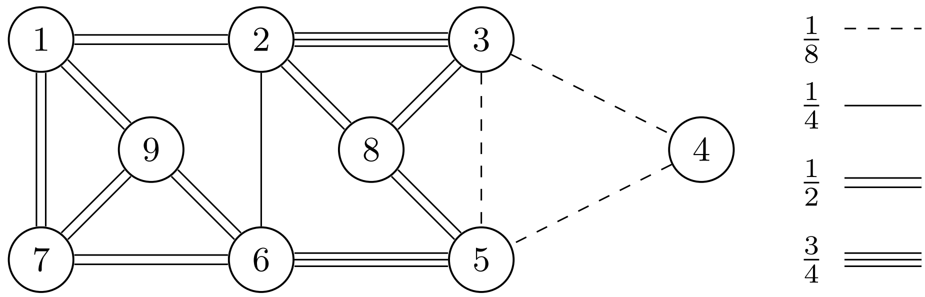


Figure 6: An example of a vector (y, x) where the associated SEC with the global minimum cut of the support graph is not violated, while violated SECs for the vector exist. The edge values of the vector (y, x) are detailed in the legend, while the vertex values are derived by the degree equations.

The straightforward exact algorithm to find violated SECs for (y, x), consists of solving $\binom{|\bar{V}|}{2}$ number of (s, t)-minimum cuts problems on $\bar{G}$, one for each pair of different vertices, and then evaluating the associated inequality (7c) using the y values of the pair of vertices. When using the push-relabel algorithm in Goldberg Tarjan (1988) with highest-level selection and global relabeling heuristics to solve the (s, t)-minimum cut problems (or better said, to solve its dual: the (s, t)-maximum flow problems), the straightforward exact strategy has a $O(|\bar{V}|^4\sqrt{|\bar{E}|})$ time complexity. Note that for cycle problems in general, the algorithm in Hao Orlin (1992) cannot be used to find the most violated SEC. Although this algorithm solves the global minimum cut in $O(|\bar{V}|^2\sqrt{|\bar{E}|})$ steps, which might be very useful, particularly for the TSP, in a general cycle problem the global minimum cut might not correspond with a violated SEC as shown above.

The proposed separation algorithms in this paper, the Dynamic Hong's algorithm and the Extended Padberg-Grötschel algorithm, are two exact algorithms for cycle problems that run in $O(|\bar{V}|^3\sqrt{|\bar{E}|})$. They are motivated by two observations made in Fischetti et al. (1997). First, for a given pair of different vertices $u, v \in V$, the most violated SEC, $\langle Q, u, v\rangle$, corresponds to the subset Q such that $(Q : V - Q)$ is a (u, v)-minimum cut. Secondly, for a given subset Q, the most violated SEC, $\langle Q, u, v\rangle$, corresponds to the vertices $u = \arg\max\{y_w : w \in Q\}$ and $v = \arg\max\{y_w : w \in V - Q\}$. The next two algorithms exploit these two observations, in order to guarantee that the most violated SEC for (y, x) is identified.

5.1 Dynamic Hong's Exact Separation Algorithm

The Hong's exact approach, which emerged in the context of the TSP, consists of solving only $|\bar{V}| - 1$ number of (s,t)-minimum cut problems, by fixing a random vertex, s, as the source of all the minimum cut problems, at the expense of possibly losing a subset of violated cuts, see Hong (1972).

This exact approach can be extended for cycle problems, by selecting s as a vertex of $\bar{V}$ with maximum y value. Based on the second observation in Fischetti et al. (1997), an s selected this way will belong to the most violated SEC corresponding to every subset Q. However, since to define a SEC we need to select another vertex in $\bar{V} - \{s\}$, based on the first observation, we consider for each $t \in \bar{V} - \{s\}$ the subset Q such that $(Q : V - Q)$ is a (s,t)-minimum cut. This shows that the extension of the Hong's approach for cycle problems is also an exact separation algorithm.

Let us suppose that the vertices $\bar{V} = \{\bar{v}_1, \ldots, \bar{v}_{|\bar{V}|}\}$ are ordered decreasingly by y and define the source $s_i = \bar{v}_1$ and the sink $t_i = \bar{v}_{i+1}$ for all $i \in \{1, \ldots, |\bar{V}| - 1\}$. In Fischetti et al. (1998) and Bérubé et al. (2009), after each (s_i,t_i)-minimum cut, $(Q : V - Q)$, they increase the weight of the edge $[s_i, t_i]$ by $2 - x(\delta(Q))$, in order to prevent collecting the same SEC in subsequent iterations. A disadvantage of this strategy is that the degree equations are not satisfied anymore. In Theorem 5.2 we achieve the same objective by shrinking the set $\{s_i, t_i\}$, with the extra feature of reducing the size of the graph for the following iterations.

The underlying idea of Theorem 5.2 comes from the shrinking rule for minimum cut problems, Theorem 3.3, in Padberg Rinaldi (1990a). This theorem says that the edges having a value greater than or equal to the upper bound of the minimum cut can be contracted. However, this rule is not safe for SECs in cycle problems. For instance, based on Theorem 3.3, in Figure 6 we would shrink the set $\{2, 6\}$ because the value of the edge $[2, 6]$ is equal to the global minimum cut value $x(C : V - C)$. However, because all the violated SECs in the figure consider the vertices 2 and 6 as disjoint ones, it is not safe to shrink the set $\{2, 6\}$.

Lemma 5.1. *Given a vector $(y, x) \in \mathbb{R}^{V \times E}$ that satisfies the degree constraints and four vertices $u, v, u^{'}, v^{'} \in \bar{V}$ such that $y_u + y_v \geq y_{u'} + y_{v'}$, let $(Q : \bar{V} - Q)$ be a (u, v)-minimum cut and $(Q^{'} : \bar{V} - Q^{'})$ be a $(u^{'}, v^{'})$-minimum cut in $\bar{G}$. If $\langle Q^{'}, u^{'}, v^{'} \rangle$ is a strictly more violated SEC than $\langle Q, u, v \rangle$, then both u, v vertices belong either to $Q^{'}$ or $\bar{V} - Q^{'}$.*

Proof. Suppose that $\langle Q^{'}, u^{'}, v^{'} \rangle$ is a strictly more violated SEC than $\langle Q, u, v \rangle$, then:

$$x(\delta(Q)) - 2y_u - 2y_v + 2 > x(\delta(S)) - 2y_{u'} - 2y_{v'} + 2 \tag{36a}$$
$$x(\delta(Q)) > x(\delta(S)) + 2y_u + 2y_v - 2y_{u'} - 2y_{v'} \tag{36b}$$
$$x(\delta(Q)) > x(\delta(S)) \tag{36c}$$

Since $x(\delta(Q)) = x(Q : \bar{V} - Q)$ is the value of the (u, v)-minimum cut and $x(\delta(Q^{'}))$ is strictly smaller than it, then both u and v belong either to $Q^{'}$ or $V - Q^{'}$.

□

Theorem 5.2 (Rule S3)**.** *Given a vector $(y, x) \in \mathbb{R}^{V \times E}$ satisfying the degree equations, consider $u, v \in \bar{V}$ such that $\min\{y_u, y_v\} \geq y_w$ for all $w \in \bar{V} - \{u, v\}$. Then, after solving the (u, v)-minimum cut problem and collecting, if any, the associated violated SECs, it is subcycle-safe to shrink $S = \{u, v\}$.*

Proof. The theorem is a direct consequence of Lemma 5.1.

□

The dynamic Hong's algorithm is based on Theorem 5.2, and it takes its name because the source, s, for the (s,t)-minimum cut problems might not be the same as in the classical approach. The algorithm works as follows: suppose that the vertices of $\bar{V}$ are ordered decreasingly by y, and set for the first minimum cut problem $s_1 = \bar{v}_1$ and $t_1 = \bar{v}_2$. Next, we solve the (s_1,t_1)-minimum cut problem, evaluate the obtained SEC candidates and, thereafter, shrink $\{s_1, t_1\}$. To proceed with the subsequent iteration, we need to know if the ordering of the vertices has changed after the $\{s_1, t_1\}$ shrinking, so we consider the Lemma 4.6. When the logical constraint $x_{[s_1,t_1]} \leq y_{s_1}$ is satisfied, we have that $y[\{s_1, t_1\}](s_1[\{s_1, t_1\}]) \geq y_{t_1} \geq y_v$ for all $v \in \bar{V} - \{s_1, t_1\}$, and, hence, the vertex $s_1[\{s_1, t_1\}]$ will be "again" the source of the subsequent minimum cut problem. However, when $x_{[s_1,t_1]} > y_{s_1}$, it might happen that $y[\{s_1, t_1\}](s_1[\{s_1, t_1\}]) \leq y_v$ for some $v \in \bar{V} - \{s_1, t_1\}$. In this situation, after shrinking the set $\{s_1, t_1\}$, we will need to reorder the vertices of $\bar{V}[\{s_1, t_1\}]$ decreasingly by y (rearrange $s_1[\{s_1, t_1\}]$ in the set $\bar{V}$). So now, to proceed, we set as s_2 and t_2, the first two vertices of $\bar{V}[\{s_1, t_1\}]$, continue by solving the (s_2,t_2)-minimum cut problem, evaluating the possible violated SECs and shrinking $\{s_2, t_2\}$, and so on.

5.2 Extended Padberg-Grötschel Exact Separation Algorithm

Padberg Grötschel (1985), showed a different exact separation algorithm for SECs in the TSP, whose key component is the multitermal flow algorithm proposed in Gomory Hu (1961). A multitermal flow algorithm is solved, in turn, using the so-called Gomory-Hu tree, which can be constructed solving a $|\bar{V}| - 1$ number of (s, t)-minimum cut problems.

In Fischetti et al. (1997) it was mentioned that an analogue approach to the one given for the TSP might be used for the SECs in the cycle problems, but no details were given to illustrate how this approach should be extended. However, note that the adaptation of the Padberg-Grötschel approach for cycle problems is not trivial. The algorithm in Padberg Grötschel (1985) for the TSP relies on the correspondence between the most violated subtour elimination constraint for (y, x) and the global minimum cut of $\bar{G}$, which is not always the case in general cycle problems (this might not even be violated while other exist).

In cycle problems, Gomory-Hu trees were used to find violated SECs in Bauer et al. (2002) for the Cardinality Constrained Cycle Problem (CCCP) and in Jepsen et al. (2014) for the Capacited Profitable Tour Problem (CPTP). Nevertheless, in absence of details of the approach used to identify the violated SECs, we understand that in both papers the selected inequality corresponds with the global minimum cut. Therefore, these separation algorithms for SECs should be considered as heuristics. As far as we know, an exact extension for the Padberg-Grötschel separation algorithm for SECs in cycle problems has not been detailed in the literature.

In order to extend the separation algorithm for cycle problems, we need to construct a Gomory-Hu tree, $T = (\bar{V}, A_T)$, of the support graph $\bar{G}$ with weights (y, x). However, unlike in the original approach, the tree T has to be constructed as a directed rooted tree, where the root is set as a vertex of $\bar{V}$ with maximum y value. Let us denote by $\Delta(v)$ the set of descendant vertices of $v \in \bar{V}$ and by r the root of the tree T. We consider that every vertex is descendant of itself, i.e., $v \in \Delta(v)$. Suppose that the arcs of A_T are in the descendant orientation, and call h_e the head vertex of an arc a. Given $a \in A_T$, we define

$$u_a = \arg\max\{y_v : v \in \Delta(h_a)\} \tag{37a}$$
$$v_a = \arg\max\{y_v : v \in \bar{V} - \Delta(h_a)\} \tag{37b}$$

which identifies the vertices, u_a and v_a, with the maximum y value for each of the two connected components of the graph $(\bar{V}, A_T - \{a\})$. Note that, from the way that we have chosen the root, we can assume that $v_a = r$. Then, once the directed rooted Gomory-Hu tree is constructed, the violated SECs are collected in $O(\bar{V})$ computational time. With that aim, we check for each arc $a \in A_T$ ($|A_T| < |\bar{V}|$) if the inequality $w_a - 2y_{u_a} - 2y_r \geq -2$ is violated, being w_a the weight of the arc a in the Gomory-Hu tree T representing the (s, t)-minimum cut for the two extreme vertices of the arc a. If this happens, the violated SEC is defined by $\langle \Delta(h_a), u_a, r \rangle$.

Note that this can be done efficiently because the u_a vertices of the arcs can be updated without an extra computational overhead. At every step of the Gomory-Hu algorithm, when a new arc is added to the tree, the descendant vertices are identified, which can be grasped to update the u_a vertices. Also, with a proper implementation of the Gomory-Hu algorithm, it is possible to maintain the subset that contains the selected r as the root of the subsequent trees. For more details, see the pseudocode in the Appendix A.

In a similar way to the extension of Hong's approach, it can be shown that the extension of Padberg-Grötschel is exact for cycle problems. In this case, the root vertex r plays the role of s, whereas each arc $a \in A_T$ identifies simultaneously a vertex in $V - \{r\}$, $t = h_a$, and its associated (s, t)-minimum cut. Furthermore, it goes one step beyond, based on the second observation, it considers u_a instead of h_a. Hence, the number of violated cuts found by the extension of the classical Hong's approach is dominated by the extension of the Padberg-Grötschel approach.

According to our experiments in Section 6, the Extended Padberg-Grötschel approach consumes a much lower computational time than the Extended Hong approach, although both approaches have the same worst case running time complexity. This happens because the subsequent (s, t)-minimum cut problems are solved in subgraphs of $\bar{G}$ in the Gomory-Hu tree based approach. When the problem size increases, the time needed for the shrinking and unshrinking operations during the Gomory-Hu tree construction is insignificant compared to the time needed to solve the (s, t)-minimum cut problems. Therefore, in addition to potentially finding more violated SECs, the Extended Padberg-Grötschel is a faster exact separation algorithm than the Extended Hong's Algorithm.

In Figure 7, we illustrate the Extended Padberg-Grötschel approach to find the violated SECs for the vector (y, x) defined in Figure 6. The weight w_a of each $a \in A_T$ in the tree is detailed above the arcs, and the y values of the vertices u_a and v_a are detailed inside a box, at the top and at the bottom respectively, near the head vertex of the arc. Two violated SECs are identified $\langle \{2, 3, 4, 5, 8\}, 2, 6 \rangle$ and $\langle \{2, 3, 8\}, 2, 6 \rangle$. Note that, if in this particular tree, the vertex 2 is chosen to be the root, only the violated SEC $\langle \{1, 6, 7, 9\}, 6, 2 \rangle$ (equivalent to $\langle \{2, 3, 8\}, 2, 6 \rangle$) is collected, which shows that the exact algorithm is sensible to the directed rooted Gomory-Hu tree construction.

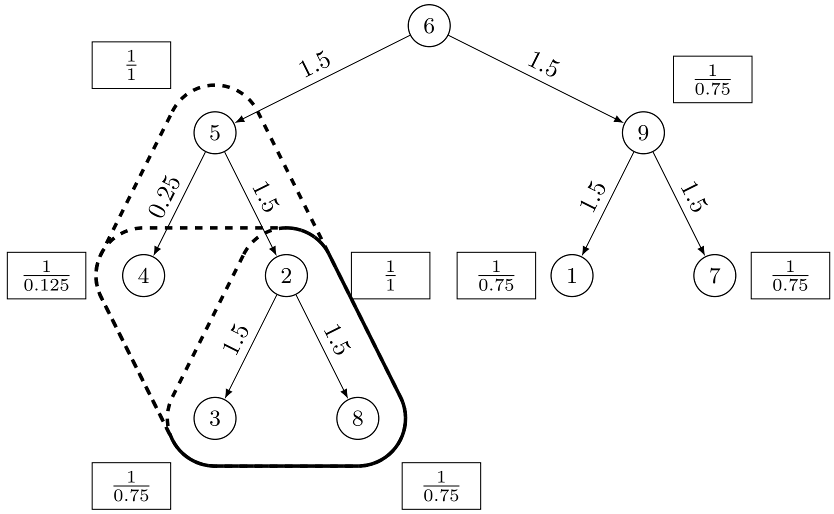


Figure 7: An example of the directed rooted Gomory-Hu tree for the SEC separation problem of Figure 6. The u_a (below) and v_a (above) values are detailed in the boxes. The arc weights are detailed next to the arcs.

Although, the detailed approach until now always finds violated inequalities when they exist, extra violated SECs can be collected using a more exhaustive search whose cost is $O(|\bar{V}|^2)$. Observe that $x(\delta(\Delta(h_a) \cup \Delta(h_f))) \leq w_a + w_f$ for every $a, f \in A_T$. Then, we can define $y_{u(e,f)} = \max\{y_{u_a}, y_{u_f}\}$ and check if $w_a + w_f - 2y_{u(a,f)} - 2y_r < -2$ for each pair arcs of A_T. This way, the violated SEC $\langle\{2,3,4,8\}, 2, 6\rangle$ in Figure 7 can be identified. We have not made use of this kind of extra SECs in our experiments.

6 Computational Results for Shrinking and Separation Algorithms for SECs

In this section we describe the results of the computational experiments for the shrinking and the exact separation algorithms for SECs. These experiments have been designed with two goals in mind. First, to show the importance of the shrinking technique for cycle problems, and second, to evaluate the performance of different combination of shrinking and separation algorithms for SECs.

The computational study of this section is inspired by two studies for the minimum cut algorithms: Jünger et al. (2000) and Goldberg Tsioutsiouliklis (2001). In both papers, the minimum cut algorithms are tested in instances originated, among others, from the solution of the TSP by a B&C algorithm. Note that, as explained in Section 5, the global minimum cut algorithms tested in these papers are not suitable for our purpose.

Jünger et al. (2000) studied the performance of different algorithms in combination with the shrinking rules defined for the minimum cut problems in Padberg Rinaldi (1990a). Similarly, in this paper, we show the performance of the combination of shrinking rules and separation algorithms for SECs in cycle problems. Goldberg Tsioutsiouliklis (2001) compared different Gomory-Hu tree building strategies: Gusfield (1990) implementation and three variants of the classical implementation. It was shown, for the SEC separation problem in the TSP, that the classical Gomory-Hu building based strategies outperform Gusfield's implementation, whereas they have not obtained significant differences among the variants of the classical implementation. The directed rooted Gomory-Hu tree algorithm presented in Section 5 can be considered within the class of classical implementations.

6.1 Benchmark Instances

The cycle problems could have a very large variety of origins, where the cycle constraints might be combined with additional constraints (e.g., a limit in the length of the cycle) and different objective functions (e.g., maximizing the profits and/or minimizing the length). These different natures of the cycle problems might vary the results obtained by each proposed strategy. However, we assume that in general terms the behaviour of the strategies for SECs is similar for all the cycle problems. So, instead of presenting an extensive comparison for different cycle problems, we focus our experiments on a well-known cycle problem, the Orienteering Problem (OP).

With the purpose of evaluating our shrinking and separation algorithms for SECs, we have built the SEC separation instances by obtaining vectors $(y, x) \notin P_C^G$ during a B&C algorithm for the OP. The OP instances are constructed based on the TSPLIB instances in Reinelt (1991) following the approach in Fischetti et al. (1998). Particularly, we have chosen the TSPLIB instances selected in Goldberg Tsioutsiouliklis (2001): pr76, att532, vm1084, rl1323, vm1748, rl5934, usa13509, d15112. Based on these 8 TSP instances, we have constructed 24 OP instances following the approach in the OP literature. The depot vertex is considered to be the first vertex of the TSPLIB instance, the maximum cycle length in the OP is set as half of the TSP value of the instance (values reported in Applegate et al. (2007)) and the profits of the vertices are generated in three different ways: Gen1, all the vertices have equal profit; Gen2, the scores are generated pseudorandomly; and Gen3, the vertices which are further from the depot vertex have a greater profit. Once the OP instances have been constructed, the SEC separation instances are generated by considering the first support graph during a B&C algorithm for the OP which satisfies the degree constraints, the logical constraints and the connectivity. We have classified the instances into two equal-sized groups: Medium, instances whose original OP problem has less than 1500 vertices, and Large, the rest of the instances. All the used OP instances and SEC separation problem instances are available at Kobeaga (2020).

6.2 Shrinking Strategies for SECs

Relying on the results of Section 3 and Section 4, we have considered 5 different shrinking strategies for SECs. We have named the obtained strategies, by concatenating the names of the involved rules: C1, C1C2, C1C2C3, S1, S1S2. The pseudocodes of these strategies are detailed in Appendix A.

In each strategy, each involved rule is applied exhaustively. For instance, for the rule C1, the hypotheses of Theorem 3.6 are checked for every possible set $S \subset \bar{V}$ and vertex $t \in \bar{V} - S$. Moreover, when a shrinkable set S is found and shrunk, new shrinkable sets might appear in the graph obtained after applying the shrinking. In order to handle these scenarios, we make use of a heap set, $H \subset \bar{V}$, which stores all the vertices that need to be checked to see whether they belong to a candidate S. For that, first, the set H is initialized considering all the vertices of $\bar{V}$. During the search procedure, whenever the heap set H is not empty, we draw one of its vertex, v, and consider it as contained in S. Then, we find neighbour vertices of v that, if they incorporate to S, might make S shrinkable. If a shrinkable set S is found, first we remove the vertices in the set S from H, and then we shrink the graph $\bar{G}$ and the vectors (y, x) and m (remember that $m_v = \max\{y_u : u \in \pi(v)\}$ for $v \in \bar{V}$). Immediately thereafter, we add the newly created vertex s and its neighbours to the heap H. Additionally, when the support graph has vertices with value one, we check if violated SECs exist as suggested by Lemma 3.10 and Theorem 3.11.

6.3 Exact Separation Algorithms for SECs

We study the performance of four exact separation algorithms for SECs:

i) Algorithm EH: Extended Hong's algorithm.

ii) Algorithm DH: Dynamic Hong's algorithm.

iii) Algorithm DHI: Dynamic Hong's algorithm with internal shrinking.

iv) Algorithm EPG: Extended Padberg-Grötschel algorithm.

The Algorithm EH is the Hong separation algorithm extended for cycle problems in Fischetti et al. (1997). The Algorithm DH refers to the Dynamic Hong separation algorithm explained in Section 5, i.e., after each minimum cut, we shrink the source and sink vertices based on rule S3. In Algorithm DHI, in analogy to the approach used in Applegate et al. (2007) for the TSP, inside the DH separation algorithm, after shrinking the source and the sink vertices, we apply the given shrinking strategy to the newly obtained graph. The Algorithm EPG refers to the extended Padberg-Grötschel algorithm explained in Section 5.

When a violated SEC, $\langle Q, u, v\rangle$, is found, we save in a repository only the Q set of the violated SEC. During the whole separation procedure each Q set is saved only once to avoid generating unnecessary cuts. Moreover, if $|Q| > |\bar{V}|/2$, we save $\bar{V} - Q$ instead of Q in order to decrease memory resource requirements. Once the separation algorithm is completed, we generate the SEC cuts from the saved Q sets in the following way: we consider for candidate vertices, u and v, the vertices with maximum y value inside Q, $M(Q) = \{u \in Q : y_u \geq y_v \ \forall v \in Q\}$, and outside Q, $M(\bar{V} - Q) = \{u \in \bar{V} - Q : y_u \geq y_v \ \forall v \in \bar{V} - Q\}$. Since the amount of generated SECs might be huge (producing memory problems), we consider only k_{in} and k_{out} randomly selected vertices from $M(Q)$ and $M(\bar{V} - Q)$, respectively. Note that in a cycle problem with depot, we have either $d \in M(Q)$ or $d \in M(\bar{V} - Q)$ for every Q, so it would be sufficient to select the depot instead of the randomly selected vertices. In other words, in these problems, it is enough to consider $u = d$ and $k_{in} = 1$ if $d \in M(Q)$ and $v = d$ and $k_{out} = 1$ otherwise. However, with the aim of obtaining

		Preprocess		Separation			
		Graph Size		Speedup			
Size	Shrinking	$\%\lvert\bar{V}\rvert$	$\%\lvert\bar{E}\rvert$	EH	DH	DHI	EPG
Medium	NO	100.00	100.00	1	9	9	9
	C1	42.55	50.61	6	29	23	19
	C1C2	39.73	46.40	7	32	27	20
	C1C2C3	39.73	46.40	7	33	25	20
	S1	22.88	26.43	16	57	51	28
	S1S2	21.26	24.53	17	60	53	27
Large	NO	100.00	100.00	1	15	15	16
	C1	30.45	37.88	17	107	74	139
	C1C2	27.95	34.10	20	122	86	151
	C1C2C3	27.95	34.10	20	121	80	150
	S1	16.15	19.91	44	221	203	215
	S1S2	14.34	17.43	53	252	227	225

Table 1: Average speedup of the proposed algorithms using the Algorithm EH with no shrinking preprocess as a baseline.

insights about the SEC generation process in general cases, in the experiments, we have ignored that the OP is a cycle problem with depot.

The pseudocodes of the considered shrinking and separation strategies can be found in Appendix A and the source code of the implementation used for the experiments is publicly available at Kobeaga (2020).

6.4 Results

For the experiments, we have run 10 times each combination of shrinking and separation strategies with two objectives in mind: evaluate the influence of the random choices during the algorithm (ties are broken randomly when ordering $\bar{V}$; source and sink vertices are selected randomly in the Gomory-Hu tree construction) and obtain a better approximation of the running times. We have divided the process of finding the violated cuts into three parts: (1) the preprocess, which considers the shrinking carried out before the separation, (2) the separation, which consists of finding the Q sets that define violated cuts, and (3) the generation of the violated SEC from the Q sets. Since the SEC generation is closely related to the obtained Q sets in the previous parts, and it is independent of the considered shrinking and separation strategies, we have limited the discussion of results to the preprocess and the separation parts.

The computational results are summarized in two tables. In Table 1, we present the information about the graph simplification and the relative time needed by each combination of strategies compared to the reference strategy (Algorithm EH with NO shrinking). In Table 2, we show the absolute values (on average) about the collected Q sets and the time needed (in milliseconds) by each combination of strategies. Although these tables give a general picture of the behaviour of the strategies, we consider that the results reflect what happens instance by instance. The detailed results of the experiments can be found in Appendix B.

In Table 1 it can be seen that the graph is contracted considerably by means of the shrinking, especially in large problems. The largest contractions are achieved with strategy S1S2. An interesting point of the results is that with the rules derived from Theorem 3.6 (C1,C2,C3) the support graph is simplified significantly, which encourages us to apply the shrinking preprocess for other valid inequalities, such as combs. Note that, rule C3 does not contract the graph more than what is already achieved by the combination of rules C2 and C3, see Section 6.5 for the discussion concerning this result.

Regarding the speedup up obtained by the shrinking strategies, the results are clear and show the importance of performing the shrinking preprocess before the separation algorithms. If we observe the column related to Algorithm EH in Table 1, the speedup obtained by each shrinking strategy is meaningful. In Medium instances, on average, the speedup is about 6 times for the least aggressive strategy (C1), and 17 times in Large instances. By means of the most aggressive strategy (S1S2) the speedup on average is 17 for Medium-sized instances and 53 in Large-sized instances.

With respect to the time needed, the separation algorithms, Algorithm DH and Algorithm EPG, are both faster than the commonly used Algorithm EH, which shows the relevance of the detailed exact separation algorithms in Section 5. If we compare Algorithm DH and Algorithm EPG, without considering any shrinking strategy, the speedups on average are similar (9 and 9 times, respectively) and Algorithm EPG in larger instances (15 and 16 times, respectively). The table also suggests, based on the results of Algorithm DH and Algorithm DHI, that it might not be convenient in the Dynamic Hong's separation algorithm to internally carry out extra shrinking procedures.

Taking into account jointly the shrinking and separation strategies, the largest speedups are obtained when rules S1 and S2 are combined in the preprocess and, after that, alternatives to the standard Hong separation algorithms are used. In terms of running time, the Algorithm DH with the S1S2 shrinking preprocess obtains the best results in the experiments, with an average speedup of 60 in Medium-sized instances and 252 in Large-sized instances. The results obtained by Algorithm EPG with the S1S2 preprocess strategy are also very good, especially in Large-sized instances with an average speedup of 225.

		Preprocess		Separation							
		All		EH		DH		DHI		EPG	
Size	Shrinking	#Q	Time	#Q	Time	#Q	Time	#Q	Time	#Q	Time
Medium	NO	0.0	0.5	83.8	211.6	79.9	17.1	79.9	17.1	438.2	16.3
	C1	0.0	0.8	27.8	30.2	58.4	5.2	58.4	6.5	149.0	7.8
	C1C2	5.5	0.8	31.6	25.2	59.4	4.6	59.4	5.5	139.7	7.3
	C1C2C3	5.5	0.9	31.6	25.5	59.4	4.5	59.4	5.9	139.8	7.4
	S1	29.3	0.9	43.4	10.2	63.1	2.6	63.1	2.9	101.3	5.3
	S1S2	35.1	0.9	48.8	9.5	69.0	2.5	69.9	2.8	98.3	5.3
Large	NO	0.0	9.9	679.4	26578.2	372.6	2140.0	372.6	2140.0	3395.1	1828.8
	C1	0.0	22.5	154.2	1513.4	266.8	203.7	266.8	295.8	756.6	146.7
	C1C2	17.0	22.8	166.8	1320.0	271.7	179.3	271.7	257.2	717.9	135.2
	C1C2C3	16.8	23.2	166.6	1321.0	271.5	181.0	271.5	277.1	717.8	136.2
	S1	169.2	25.1	225.4	515.4	287.0	95.4	287.0	103.8	507.1	94.7
	S1S2	248.8	25.3	293.1	427.2	372.2	83.5	374.3	91.5	528.0	91.1

Table 2: On average, the number of Q sets found and the time needed by strategy and size.

Apart from the running time, an aspect to consider when making a choice about the separation algorithm is the number of violated cuts found. As we have already mentioned, in the cycle problems, the number of collected violated SECs is closely related with the Q sets obtained by the separation algorithms. Therefore, we have measured the obtained amount of Q sets instead of the number of violated SECs. In Table 2, the average number of Q sets and time of each combination of strategies is shown.

The first aspect to note is that, by means of the shrinking preprocess, which is considerably faster than the exact separation procedure, we are able to find violated SECs in many instances (via Theorem 3.11 and Lemma 3.10). These violated SECs might be enough for the separation goal and, in practice, we could skip the exact separation algorithm if violated inequalities are found in the preprocess. In the separation process, in general, the largest amount of Q sets are obtained by Algorithm EPG, as was anticipated theoretically in Section 5. Note that, the quantity of obtained Q sets is sensitive to the randomness of the shrinking and separation strategies (it can be concluded because $\#Q$ is not always an integer).

In the view of these results, the S1S2 shrinking strategy is the best choice to use as the preprocess of SEC separation algorithms. Bearing in mind both the time and the obtained amount of Q sets, either Algorithm DH or Algorithm EPG might be a good choice as the separation algorithm. However, it is not clear from these results which of the two exact approaches should be used in practice. It probably depends on the nature and the size of the cycle problem under consideration.

6.5 Discussion

Finally, we would like to open a discussion about the following concerns as a consequence of the computational results. It might be helpful, to look at the detailed computational results in Appendix B. to understand the motivation behind the discussion below.

In Figure 5, an example of a vector $(y, x) \in P_A^G$ was shown where rule C3 can be applied but rules C1 nor C2 cannot. However, in the experiments, although rule C3 has been applied in some instances, we have not obtained any situation in which rule C3 was able to simplify the support graph more than with the rest of the rules.

An open question is then to explain why rule C3 does not improve the results obtained by means of the rules C1 and C2. We believe that this is related with the planarity property of the support graphs, which is satisfied in the considered instances. Note that the graph in the example of Figure 5 is not planar because the complete graph of 5 vertices, K_5, is a subgraph of it.

Conjecture 6.1. *Given a graph G, let $(y, x) \in P_A^G$ be a vector. If the support graph $\bar{G}$ of (y, x) is planar, then the combination of the rules $C1$ and $C2$ dominate the rule $C3$.*

Note that the rules C1, C2, and C3 induce a contraction of an edge (a sequence of contractions for C3), which is a closed operation in planar graphs. Therefore, if $\bar{G}$ is planar then $\bar{G}[S]$ is also planar for every subset S obtained from these rules. While working with the OP, we have empirically seen that in geometrical instances the support graph obtained within a B&C is planar most of the time.

Another interesting fact that can be extracted from the experiments is that the number of vertices and edges in the shrunk graph (the final result) is independent of the ordering of the considered rules and the shrinkable sets. This suggests the idea that the obtained shrunk graphs are isomorphic.

Conjecture 6.2. *Given a graph G, let $(y, x) \in P_A^G$ be a vector and SRK $\in \{$C1, C1C2, C1C2C3, S1, S1S2$\}$ be a fixed shrinking strategy, then the graphs obtained by applying SRK to (y, x) are isomorphic.*

If the conjecture is true, the complexity of the separation algorithm carried out in the shrunk graph does not depend on the different implementations of a shrinking strategy. As a consequence, in the future, we might focus on identifying the implementations of the shrinking strategies that might obtain the largest amount of Q sets, especially for the preprocess, e.g., by reordering the vertices in the heap.

7 Conclusions and Future Work

In this paper, for cycle problems, we have successfully generalized the global (C1, C2 and C3) and SEC specific (S1, S2 and S3) shrinking rules proposed in the literature of the TSP. The obtained computational results for the shrinking in the OP are remarkable and, hence, very promising for other cycle problems. The results clearly show that the shrinking technique considerably improves the running time of the separation algorithm for SECs. This opens the possibility to investigate in two directions in cycle problems: (1) studying the shrinking for other valid cycle inequalities of the OP (e.g., combs) and (2) evaluating for other cycle problems the shrinking technique in SEC separation problems.

Part of the paper focuses on exact SEC separation algorithms for cycle problems. We have extended from the TSP two exact algorithms (Algorithm DH and Algorithm EPG). The proposed separation algorithms were shown to be more efficient in the OP than the exact algorithm used so far in the literature (the adaptation of the classical Hong's approach). The importance of the detailed extension of the Padberg-Grötschel approach, Algorithm EPG, lies in the fact that in cycle problems, in general, the global minimum cut of a support graph might not generate a violated SEC, while violated SECs in the same graph exist. An example is given where this claim is shown, which implies that the adaptions of the Padberg-Grötschel approach used so far in the literature of cycle problems should be viewed as heuristic separation algorithms. Therefore, this might be the first exact extension of the Padberg-Grötschel approach in the literature for cycle problems.

Acknowledgements 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 the project ELKARTEK (Basque Government). The second author is supported by IT-1252-19 (Basque Government) and PPG17/32 and GIU17/011 (University of the Basque Country). The third author is also supported by IT1244-19 (Basque Government) and TIN2016-78365R (Spanish Ministry of Science and Innovation). This article has been partially written while the first author was at Operational Research Group at University of Brescia as a visiting PhD student under the supervision of Prof. Speranza. He wishes to sincerely thank the OR group and the university for their warm hospitality and an excellent atmosphere. 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.

References

Applegate DL, Bixby RE, Chvatal V, Cook WJ (2007) *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)* (Princeton, NJ, USA: Princeton University Press), ISBN 0691129932, 9780691129938.

Archetti C, Speranza MG, Vigo D (2014) Vehicle routing problems with profits. *Vehicle Routing: Problems, methods, and applications*, chapter 10, 273–297 (MOS-SIAM Series on Optimization).

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

Bauer P, Linderoth J, Savelsbergh M (2002) A branch and cut approach to the cardinality constrained circuit problem. *Mathematical Programming* 91:307–348.

Bérubé JF, Gendreau M, Potvin JY (2009) A branch-and-cut algorithm for the undirected prize collecting traveling salesman problem. *Networks* 54:56–67.

Coullard CR, Pulleyblank WR (1989) On cycle cones and polyhedra. *Linear Algebra and its Applications* 114-115:613 – 640, ISSN 0024-3795.

Crowder H, Padberg MW (1980) Solving large-scale symmetric travelling salesman problems to optimality. *Management Science* 26(5):495–509.

Feillet D, Dejax P, Gendreau M (2005) Traveling salesman problems with profits. *Transportation Science* 39(2):188–205, ISSN 1526-5447.

Fischetti M, Salazar-González JJ, Toth P (1995) The symmetric generalized traveling salesman polytope. *Networks* 26(2):113–123.

Fischetti M, Salazar-González JJ, Toth P (1997) A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research* 45:378–394.

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

Goldberg AV, Tarjan RE (1988) A new approach to the maximum-flow problem. *J. ACM* 35(4):921–940, ISSN 0004-5411.

Goldberg AV, Tsioutsiouliklis K (2001) Cut tree algorithms: An experimental study. *Journal of Algorithms* 38(1):51 – 83, ISSN 0196-6774.

Gomory R, Hu T (1961) Multiterminal network flows. *Journal of The Society for Industrial and Applied Mathematics* 9.

Grötschel M, Holland O (1991) Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming* 51(1):141–202, ISSN 1436-4646.

Grötschel M, Padberg M (1979) On the symmetric travelling salesman problem i: Inequalities. *Mathematical Programming* 16(1):265–280, ISSN 1436-4646.

Gusfield D (1990) Very simple methods for all pairs network flow analysis. *SIAM Journal on Computing* 19(1):143–155.

Gutin G, Punnen AP (2007) *The Traveling Salesman Problem and Its Variations (Combinatorial Optimization)* (Springer).

Hao J, Orlin JB (1992) A faster algorithm for finding the minimum cut in a graph. *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, 165–174, SODA '92 (Society for Industrial and Applied Mathematics), ISBN 089791466X.

Hong S (1972) *A Linear Programming Approach for the Traveling Salesman Problem*. Ph.D. Thesis (Baltimore, Maryland, USA: Johns Hopkins University).

Jepsen MK, Petersen B, Spoorendonk S, Pisinger D (2014) A branch-and-cut algorithm for the capacitated profitable tour problem. *Discrete Optimization* 14:78 – 96, ISSN 1572-5286.

Jünger M, Rinaldi G, Thienel S (2000) Practical performance of efficient minimum cut algorithms. *Algorithmica* 26:172–195, ISSN 1432-0541.

Kobeaga G (2020) gkobeaga/cpsrksec: Preprint version. `http://dx.doi.org/10.5281/zenodo.3873356`.

Padberg M, Grötschel M (1985) Polyhedral computations. E L Lawler, J K Lenstra, A H G Rinnooy Kan, D B Shimoys, eds., *The Traveling Salesman Problem*, 307–360 (Chichester, UK: John Wiley & Sons).

Padberg M, Rinaldi G (1990a) An efficient algorithm for the minimum capacity cut problem. *Mathematical Programming* 47(1):19–36, ISSN 1436-4646.

Padberg M, Rinaldi G (1990b) Facet identification for the symmetric traveling salesman polytope. *Mathematical Programming* 47(1):219–257, ISSN 1436-4646.

Padberg M, Sung TY (1991) An analytical comparison of different formulations of the travelling salesman problem. *Mathematical Programming* 52(1):315–357, ISSN 1436-4646.

Pferschy U, Staněk R (2017) Generating subtour elimination constraints for the tsp from pure integer solutions. *Central European Journal of Operations Research* 25:231–260, ISSN 1613-9178.

Reinelt G (1991) Tsplib - a traveling salesman problem library. *ORSA Journal on Computing* 3(4):376–384.

Vansteenwegen P, Gunawan A (2019) *State-of-the-Art Solution Techniques for OP and TOP*, 41–66 (Cham: Springer International Publishing), ISBN 978-3-030-29746-6.

Appendices

A Pseudocodes of the Shrinking and Separation Strategies

In this appendix, we detail the pseudocodes of the shrinking and separation strategies used in the computational experiments for Section 6. These strategies are combinations of the shrinking rules proposed in Section 3 and Section 4, and the exact separation algorithms proposed in Section 5.

The pseudocodes should be considered as illustrations of the implementations of strategies whose aim is to help the reader to understand how the strategies work. The source code in C of the computational implementations is available at Kobeaga (2020). In Table A.1, we detail the meaning of the symbols used in the pseudocodes.

Symbol		Meaning
$G = (V, E)$		Input graph of the cycle problem
$\bar{G} = (\bar{V}, \bar{E})$		Support graph
(y, x)	$\in P_A^G$	A solution of the LP_0
m	$\in \mathbb{R}_+^{\bar{V}}$	A vector where $m_v = \max\{y_u : u \in \pi(v)\}$
H	$\subset \bar{V}$	Heap: vertices remaining to check
S	$\subset \bar{V}$	A subset candidate for the shrinking
Q	$\subset V$	A subset of V
$\bar{Q}$	$\subset \bar{V}$	A subset of $\bar{V}$
$\mathcal{Q}$	$\subset \mathcal{P}(V)$	List of Q sets of V
$\mathcal{L}$		List of violated SECs
D	$\subset \bar{V}$	Set of fixed vertices. In a cycle problem with depot: $D = \{d\}$
O	$\subset \bar{V}$	Set of vertices with value one
$(k_{in} \times k_{out})$	$\in \mathbb{N}_+ \times \mathbb{N}_+$	Maximum vertices (inside and outside) considered when generating the violated SECs from the Q sets
$T = (V, A_T)$		A directed rooted tree
$parent$	$V \to V$	Successive parent of each v in the tree
$child$	$V \to V$	Successive children of each v in the tree
w	$\in \mathbb{R}_+^{A_T}$	Weights of the arcs of the Gomory-Hu tree
$G^* = (V^*, E^*)$		Generic graph used in the Gomory-Hu tree construction

Table A.1: A summary of the symbols used in the pseudocodes

A.1 Shrinking Strategies

The shrinking strategies are combinations of the shrinking rules of Section 3 and Section 4. In total, 5 different shrinking strategies for SECs are obtained: C1, C1C2, C1C2C3, S1 and S1S2. The SHRINK/UPDATE procedure refers to a process performed every time a set is shrunk.

Algorithm SHRINK/UPDATE: Shrink graph and vectors. Save Q sets. Update heap.

input : $\bar{G}$, (y, x), m, H, S and $\mathcal{Q}$
output : $\bar{G}$, (y, x), m, H, s and $\mathcal{Q}$

1 $\bar{G} \leftarrow \bar{G}[S]$;
2 $(y, x) \leftarrow (y[S], x[S])$;
3 $m \leftarrow m[S]$;
4 $H \leftarrow H[S]$;
5 $O \leftarrow \{v \in \bar{V} : m_v \geq 1\}$;
6 **for** $n \in N(s)$ **do**
7 **if** $y_n < x_{[n,s]}$ **then**
8 **for** $r \in O$ **do**
9 **if** $r \neq s$ **then**
10 **if** $\langle \{s, n\}, s, r \rangle$ *violates* (34) **then**
11 $Q \leftarrow \{\pi(\{s, n\})\}$;
12 **if** $|Q| > |V|/2$ **then**
13 $Q \leftarrow V - Q$;
14 **end**
15 $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{Q\}$;
16 **goto** line 20;
17 **end**
18 **end**
19 **end**
20 **end**
21 $H \leftarrow H \cup \{n\}$;
22 **end**

Algorithm C1: Shrinking: Rule C1

input : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$
output : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$

1 **while** $|H| \neq \emptyset$ **do**
2 Select a vertex $u \in H$;
3 $H \leftarrow H - \{u\}$;
4 $c \leftarrow y_u$;
5 **for** $v \in N(u)$ **do**
6 **if** $y_v = c$ *and* $x_{[u,v]} = c$ **then**
7 **for** $t \in N(v) - \{u\}$ **do**
8 **if** $y_t = c$ *and* $x_{[v,t]} = c$ **then**
9 $S \leftarrow \{u, v\}$;
10 SHRINK/UPDATE ($\bar{G}, (y, x), m, H, S, \mathcal{Q}$);
11 **goto** line 15;
12 **end**
13 **end**
14 **end**
15 **end**
16 **end**

Algorithm C1C2: Shrinking: Rule C1 and Rule C2

input : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$
output : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$

1 **while** $|H| \neq \emptyset$ **do**
2   Select a vertex $u \in H$;
3     $H \leftarrow H - \{u\}$;
4     $c \leftarrow y_u$;
5   **for** $v \in N(u)$ **do**
6   **if** $y_v = c$ *and* $x_{[u,v]} = c$ **then**
7   **for** $t \in N(v) - \{u\}$ **do**
8   **if** $y_t = c$ *and* $x_{[u,t]} + x_{[v,t]} = c$ **then**
9                     $S \leftarrow \{u, v\}$;
10   SHRINK/UPDATE ($\bar{G}, (y, x), m, H, S, \mathcal{Q}$);
11   **goto** line 15;
12   **end**
13   **end**
14   **end**
15   **end**
16 **end**

Algorithm C1C2C3: Shrinking: Rule C1, C2 and C3

input : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$
output : $\bar{G}$, (y, x), m, H and $\mathcal{Q}$

1 **while** $|H| \neq \emptyset$ **do**
2   Select a vertex $u \in H$;
3     $H \leftarrow H - \{u\}$;
4     $c \leftarrow y_u$;
5   **for** $v \in N(u)$ **do**
6   **if** $y_v = c$ *and* $x_{[u,v]} = c$ **then**
7   **for** $t \in N(v) - \{u\}$ **do**
8   **if** $y_t = c$ *and* $x_{[u,t]} + x_{[v,t]} = c$ **then**
9                     $S \leftarrow \{u, v\}$;
10   SHRINK/UPDATE ($\bar{G}, (y, x), m, H, S, \mathcal{Q}$);
11   **goto** line 26;
12   **end**
13   **end**
14   **for** $w \in N(v) - \{u\}$ **do**
15   **if** $x_{[u,t]} + x_{[u,w]} + x_{[v,w]} = 2c$ **then**
16   **for** $t \in N(w) - \{v, u\}$ **do**
17   **if** $y_t = c$ *and* $x_{[u,t]} + x_{[v,t]} = c$ **then**
18                             $S \leftarrow \{u, v, w\}$;
19   SHRINK/UPDATE ($\bar{G}, (y, x), m, H, S, \mathcal{Q}$);
20   **goto** line 26;
21   **end**
22   **end**
23   **end**
24   **end**
25   **end**
26   **end**
27 **end**

Algorithm S1: Shrinking: Rule S1

```
input  : Ḡ, (y, x), m, H and Q
output : Ḡ, (y, x), m, H and Q
while |H| ≠ ∅ do
    Select a vertex u ∈ H;
    H ← H − {u};
    c ← y_u;
    for v ∈ N(u) do
        if y_v = c and x_[u,v] = c then
            if ∃w ∈ V̄ − {u, v} such that y_w ≥ c then
                S ← {u, v};
                SHRINK/UPDATE (Ḡ, (y, x), m, H, S, Q);
                goto line 13;
            end
        end
    end
end
```

Algorithm S1S2: Shrinking: Rule S1 and S2

```
input  : Ḡ, (y, x), m, H, D and Q
output : Ḡ, (y, x), m, H, D and Q
while |H| ≠ ∅ do
    Select a vertex u ∈ H;
    H ← H − {u};
    c ← y_u;
    for v ∈ N(u) do
        if y_v = c and x_[u,v] = c then
            if ∃w ∈ V̄ − {u, v} such that y_w ≥ c then
                S ← {u, v};
                SHRINK/UPDATE (Ḡ, (y, x), m, H, S, Q);
                goto line 17;
            end
        else if x_[u,v] > y_u and x_[u,v] > y_v then
            S ← {u, v};
            SHRINK/UPDATE (Ḡ, (y, x), m, H, S, Q);
            goto line 17;
        end
    end
end
```

A.2 Exact SEC Separation Strategies

The exact separation strategies detailed in this appendix refer to the separation algorithms used for the experiments in Section 6. We assume that the vertex set $\bar{V} = \{v_1, \ldots, v_{|\bar{V}|}\}$ is an ordered set. The CUTGEN algorithm is the procedure detailed in Section 6 to generate the most violated SECs corresponding to set Q given the parameter $(k_{in}, k_{out}) \in \mathbb{N}_+ \times \mathbb{N}_+$. The vector (k_{in}, k_{out}) represents the maximum amount of vertices that are considered inside and outside Q. Note that, CUTGEN is defined to select, for each inside vertex, a number of k_{out} different random outside vertices to maximize the randomness of the obtained violated SECs.

Algorithm EH: Extended Hong's exact separation algorithm

input : $\bar{G}$, (y, x), D and (k_{in}, k_{out})
output : A list $\mathcal{L}$ of violated SECs

$\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y; $m \leftarrow y$;
$H \leftarrow \bar{V}$;
Apply shrinking strategy $(\bar{G}, (y, x), m, H, D, \mathcal{Q})$;
while $|\bar{V}| > 1$ **do**
  $Q \leftarrow (v_1, v_2)$-minimum cut in the graph $\bar{G}$;
  if $\langle Q, v_1, v_2 \rangle$ *violates* (34) **then**
    if $|Q| > |V|/2$ **then**
      $Q \leftarrow V - Q$;
    end
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\pi(Q)\}$;
  end
end
$\mathcal{L} \leftarrow$ CUTGEN $(\bar{G}, (y, x), D, \mathcal{Q}, (k_{in}, k_{out}))$;

Algorithm DH: Dynamic Hong's exact separation algorithm

input : $\bar{G}$, (y, x), D and (k_{in}, k_{out})
output : A list $\mathcal{L}$ of violated SECs

$\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y;
$m \leftarrow y$;
$H \leftarrow \bar{V}$;
Apply shrinking strategy ($\bar{G}, (y, x), m, H, \mathcal{Q}$);
while $|\bar{V}| > 1$ **do**
    $Q \leftarrow (v_1, v_2)$-minimum cut in the graph $\bar{G}$;
    if $\langle Q, v_1, v_2 \rangle$ *violates* (34) **then**
        if $|Q| > |V|/2$ **then**
            $Q \leftarrow V - Q$;
        end
        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\pi(Q)\}$;
    end
    if $x_{[v_1, v_2]} > y_{v_2}$ **then**
        reorder $\leftarrow 1$;
    else
        reorder $\leftarrow 0$;
    end
    $S \leftarrow \{v_1, v_2\}$;
    $\bar{G} \leftarrow \bar{G}[S]$;
    $(y, x) \leftarrow (y[S], x[S])$;
    $m \leftarrow m[S]$;
    if *reorder* **then**
        $\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y;
    end
end
$\mathcal{L} \leftarrow$ CUTGEN ($\bar{G}, (y, x), D, \mathcal{Q}, (k_{in}, k_{out})$);

Algorithm DHI: Dynamic Hong with extra shrinking separation algorithm

input : $\bar{G}$, (y, x), D and (k_{in}, k_{out})
output : A family $\mathcal{Q}$ of violated SECs
$\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y;
$m \leftarrow y$;
$H \leftarrow \bar{V}$;
Apply shrinking strategy ($\bar{G}, (y, x), m, H, \mathcal{Q}$);
while $|\bar{V}| > 1$ **do**
    $Q \leftarrow (v_1, v_2)$-minimum cut in the graph $\bar{G}$;
    if $\langle Q, v_1, v_2 \rangle$ *violates* (34) **then**
        if $|Q| > |V|/2$ **then**
            $Q \leftarrow V - Q$;
        end
        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\pi(Q)\}$;
    end
    if $x_{[v_1, v_2]} > y_{v_2}$ **then**
        reorder $\leftarrow 1$;
    else
        reorder $\leftarrow 0$;
    end
    $S \leftarrow \{v_1, v_2\}$;
    SHRINK/UPDATE ($\bar{G}, (y, x), m, H, S, \mathcal{Q}$);
    Apply shrinking strategy ($\bar{G}, (y, x), m, H, \mathcal{Q}$);
    if *reorder* **then**
        $\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y;
    end
end
$\mathcal{L} \leftarrow$ CUTGEN ($\bar{G}, (y, x), D, \mathcal{Q}, (k_{in}, k_{out})$);

Algorithm EPG: Extended Padberg-Grötschel exact separation algorithm

input : $\bar{G}$, (y, x), D and (k_{in}, k_{out})
output : A family $\mathcal{Q}$ of violated SECs
$\bar{V} \leftarrow$ sort $\bar{V}$ decreasingly by y;
$m \leftarrow y$;
Apply shrinking strategy ($\bar{G}, (y, x), m, H, \mathcal{Q}$);
$(T, w, u) \leftarrow$GHTREE ($\bar{G}, (y, x), v_1$);
for $a \in A_T$ **do**
    $Q \leftarrow d_a$;
    if $w_a - 2 \cdot u_a - 2 \cdot v_a < 2$ **then**
        if $|Q| > |V|/2$ **then**
            $Q \leftarrow V - Q$;
        end
        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\pi(Q)\}$;
    end
end
$\mathcal{L} \leftarrow$ CUTGEN ($\bar{G}, (y, x), D, \mathcal{Q}, (k_{in}, k_{out})$);

Algorithm CUTGEN: SEC generation

input : $\bar{G}$, (y, x), D, $\mathcal{Q}$, (k_{in}, k_{out})
output : A family $\mathcal{L}$ of violated SECs

for $Q \in \mathcal{Q}$ **do**
  if $D \cap Q = \emptyset$ **then**
    $M_{in} \leftarrow \{v \in Q : y_v \geq y_u \ \forall u \in Q\}$;
    $S_{in} \leftarrow$ randomly select k_{in} vertices from M_{in};
  else
    $S_{in} \leftarrow$ a vertex in $D \cap Q$;
  end
  if $D - Q = \emptyset$ **then**
    $M_{out} \leftarrow \{v \in \bar{V} - Q : y_v \geq y_u \ \forall u \in \bar{V} - Q\}$;
  else
    $S_{out} \leftarrow$ a vertex in $D - Q$;
  end
  for $u \in S_{in}$ **do**
    if $D - Q = \emptyset$ **then**
      $S_{out} \leftarrow$ randomly select k_{out} vertices from M_{out};
    end
    for $v \in S_{out}$ **do**
      Add the violated SEC $\langle Q, u, v \rangle$ to $\mathcal{L}$;
    end
  end
end
$\mathcal{L} \leftarrow$ CUTGEN $(\bar{G}, (y, x), D, \mathcal{Q}, (k_{in}, k_{out}))$;

A.3 Directed Rooted Gomory-Hu Tree

As was explained in Section 5, the key for an efficient extension of the Padberg-Grötschel exact separation algorithm is the construction of the directed rooted Gomory-Hu tree, which is detailed in the following pseudocodes. The novelty is the ADD-ARC/REORDER-TREE procedure, where we show how the Gomory-Hu construction must be adapted to evaluate the u_v values ($u_v = \arg\max\{y_u : u \in \Delta(v)\}$) and reorder the tree in order to maintain a given vertex in the top of the tree.

Algorithm GHTREE: Rooted directed Gomory-Hu tree

input : $\bar{G}$, (y, x), r
output : T, w, u: a rooted directed weighted tree

$T \leftarrow (V, \emptyset)$;
for $v \in V$ **do**
  $u_v = m_v = \arg\max\{y_w : w \in \pi(v) \in \bar{G}\}$;
end
$G^* \leftarrow \bar{G}$ and consider $|\pi(v)| = 1$ for every $v \in V^*$;
$(T, w, u) \leftarrow$ GHTREE-RECURSIVE$(G^*, (y, x), r, T, w, u)$;

Algorithm GHTREE-RECURSIVE: Recursive operator to build the Gomory-Hu tree

input : G^*, (y, x), r, T, w, u
output : T, w, u

$C \leftarrow \{v \in V^* : |\pi(v)| = 1\}$;
if $|C| > 1$ **then**
  $(a, b) \leftarrow$ randomly select two different vertices from C;
  $(A : B) \leftarrow (a, b)$-minimum cut in G^*;
  $(T, w, u, r_a, r_b) \leftarrow$ ADD-ARC/REORDER-TREE$(T, (y, x), m, u, r, A, B)$;
  $(T, w, u) \leftarrow$ GHTREE-RECURSIVE$(G^*[B], (y[B], x[B]), r_a, T, w, u)$;
  $(T, w, u) \leftarrow$ GHTREE-RECURSIVE$(G^*[A], (y[A], x[A]), r_b, T, w, u)$;
end

Algorithm ADD-ARC/REORDER-TREE: Add arc and reorder the tree

```
input  : T, (y, x), m, u, r, A, B
output : T, w, u, r_a, r_b
if r ∈ A then
    r_a ← r;
    r_b ← b;
    if parent(r) ∈ A or parent(r) = ∅ then
        e = (r, b);
    else
        e = (b, r);
        f = (p(r), r);
        g = (p(r), b);
        w_g ← w_f;
        A_T = A_T − {f} ∪ {g};
        m_r = max{m_r, m_b};
    end
    u_r = m_r;
    u_b = m_b;
    for c ∈ child(r) do
        if c ∈ A then
            u_r = max{u_r, u_c};
        else
            A_T = A_T − {(r, c)} ∪ {(a, c)};
            u_b = max{u_b, u_c};
        end
    end
else
    r_a ← a;
    r_b ← r;
    if parent(r) ∈ B or parent(r) = ∅ then
        e = (r, a);
    else
        e = (a, r);
        f = (p(r), r);
        g = (p(r), a);
        w_g ← w_f;
        A_T = A_T − {f} ∪ {g};
    end
    u_r = m_r;
    u_a = m_a;
    for c ∈ child(r) do
        if c ∈ B then
            u_r = max{u_r, u_c};
        else
            A_T = A_T − {(r, c)} ∪ {(a, c)};
            u_a = max{u_a, u_c};
        end
    end
end
A_T = A_T ∪ {e};
w_e ← x(A : B);
```

B Detailed Computational Results

In this section, we show the computational results obtained in each considered SEC instance. For each instance, we present three tables: two are related with the shrinking processes and one is related with separation and SEC generation processes. In addition, the results are separated into three groups (Gen1, Gen2 and Gen3). These groups represent the generation strategy proposed in (Fischetti et al., 1998) to build the OP vertex scores which are then used to obtain the support graphs.

In tables Table B.1, Table B.3, . . . and Table B.15, we report the details of the shrinking preprocess. One can see, below the support graph and shrunk graph columns, the size of the given support graph and the size of the shrunk support graph for each shrinking strategy. In the preprocess columns, we show the number of Q sets obtained and the time (in milliseconds) needed by each shrinking preprocess. As can be seen, the shrinking is very fast, needing very few dozens of millisecond to be accomplished in the larger instances. An interesting point of these tables is that within the shrinking preprocess we are already able to obtain Q sets that correspond with violated SECs. In particular, the largest amount of Q sets are obtained with the shrinking strategy S1S2.

In tables Table B.2, Table B.4, . . . and Table B.16, we report the number of times a rule is applied by each shrinking strategy. Regarding the Conjecture 1 in the discussion of the computational experiments of the main paper, it can be seen that Rule C3 is rarely applied in the shrinking preprocess. Moreover, the strategy C1C2C3 does not provide further contractions of the support graph and, in all the compared instances, the obtained final shrunk graphs have the same amount of vertices and edges as with strategy C1C2.

The extra column in these tables represents how many extra vertices are contracted in the internal shrinking process of Algorithm DHI, i.e, Extra is increased by one if rule C1, C2 or S1 is applied and by two if rule C3 is applied. The results show that this extra shrinking is rarely achieved.

In tables Table B.17, Table B.18, . . . and Table B.24 ,we report the details about the separation process and SEC generation. We can see that EPG approach always obtains more violated SECs than Algorithm EH as suggested theoretically in the main paper. Moreover, without using the shrinking preprocess, the EPG algorithm is always faster than Algorithm EH except for the smallest instance pr76.

Regarding the SEC generation process, we compare two strategies 1×1 and 10×10, which refer to the amount of vertices considered inside and outside Q sets when generating the violated SECs. What we see is that, in medium-sized instances, the generation of violated SECs is the most time-consuming part (see the results regarding Algorithm EPG), but in large-sized, this difference is shortened. Nevertheless, it is likely that most of the generated violated cuts by 10×10 (around half a million of different violated SECs were obtained in large-sized instances by EPG) are useless and counterproductive to consider them, in practice, for a B&C.

pr76

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\lvert V \rvert$	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	#Q	Time	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	#Q	Time	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	$\lvert \bar{V} \rvert$	$\lvert \bar{E} \rvert$	#Q	Time
NO	76	65	71	65	71	0	0.05	50	59	50	59	0	0.06	54	63	54	63	0	0.05
C1	76	65	71	26	32	0	0.10	50	59	20	29	0	0.08	54	63	24	33	0	0.09
C1C2	76	65	71	26	32	0	0.09	50	59	20	29	0	0.08	54	63	24	33	0	0.09
C1C2C3	76	65	71	26	32	0	0.10	50	59	20	29	0	0.09	54	63	24	33	0	0.10
S1	76	65	71	14	15	4	0.10	50	59	9	13	0	0.08	54	63	13	18	0	0.09
S1S2	76	65	71	11	12	4	0.10	50	59	9	13	0	0.08	54	63	13	18	0	0.09

Table B.1: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in pr76.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	39	0	0	0	0	75	0	30	0	0	0	0	53	0	30	0	0	0	0	65	0
C1C2	39	0	0	0	0	75	0	30	0	0	0	0	53	0	30	0	0	0	0	65	0
C1C2C3	39	0	0	0	0	75	0	30	0	0	0	0	53	0	30	0	0	0	0	65	0
S1	0	0	0	51	0	73	0	0	0	0	41	0	62	0	0	0	0	41	0	63	0
S1S2	0	0	0	53	1	72	0	0	0	0	41	0	62	0	0	0	0	41	0	63	0

Table B.2: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in pr76. The extra column is particular of DHI separation strategy (during separation).

att532

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	532	458	528	458	528	0	0.33	413	503	413	503	0	0.29	412	512	412	512	0	0.32
C1	532	458	528	166	236	0	0.62	413	503	212	302	0	0.51	412	512	240	340	0	0.50
C1C2	532	458	528	142	196	3	0.67	413	503	200	279	4	0.53	412	512	221	305	6	0.54
C1C2C3	532	458	528	142	196	4	0.70	413	503	200	279	4	0.59	412	512	221	305	6	0.58
S1	532	458	528	77	111	15	0.74	413	503	119	164	29	0.63	412	512	135	185	24	0.59
S1S2	532	458	528	73	105	19	0.69	413	503	109	148	31	0.65	412	512	129	179	25	0.63

Table B.3: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in att532.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0
C1	292	0	0	0	0	531	0.0	201	0	0	0	0	462	0.0	172	0	0	0	0	466	0.0
C1C2	302	14	0	0	0	534	0.0	205	8	0	0	0	469	0.0	178	13	0	0	0	470	0.0
C1C2C3	302	10	2	0	0	528	0.0	205	8	0	0	0	469	0.0	178	13	0	0	0	470	0.0
S1	0	0	0	381	0	513	0.0	0	0	0	294	0	470	0.0	0	0	0	277	0	467	0.0
S1S2	0	0	0	381	4	508	0.5	0	0	0	296	8	469	0.0	0	0	0	278	5	470	0.2

Table B.4: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in att532. The extra column is particular of DHI separation strategy (during separation).

vm1084

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	1084	861	980	861	980	0	0.66	863	1012	863	1012	0	0.68	785	917	785	917	0	0.63
C1	1084	861	980	297	416	0	1.16	863	1012	377	526	0	1.18	785	917	297	429	0	1.08
C1C2	1084	861	980	260	354	8	1.20	863	1012	341	465	7	1.17	785	917	267	378	7	1.11
C1C2C3	1084	861	980	260	354	8	1.30	863	1012	341	465	6	1.27	785	917	267	378	7	1.22
S1	1084	861	980	147	202	40	1.30	863	1012	213	289	45	1.30	785	917	160	233	34	1.14
S1S2	1084	861	980	134	180	48	1.39	863	1012	200	272	53	1.37	785	917	146	210	42	1.26

Table B.5: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in vm1084.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	564	0	0	0	0	945	0	486	0	0	0	0	970	0	488	0	0	0	0	881	0
C1C2	582	19	0	0	0	980	0	502	20	0	0	0	971	0	500	18	0	0	0	887	0
C1C2C3	582	19	0	0	0	980	0	502	18	1	0	0	968	0	500	16	1	0	0	882	0
S1	0	0	0	714	0	962	0	0	0	0	650	0	975	0	0	0	0	625	0	876	0
S1S2	0	0	0	716	11	950	0	0	0	0	654	9	964	0	0	0	0	627	12	870	0

Table B.6: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in vm1084. The extra column is particular of DHI separation strategy (during separation).

rl1323

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	1323	1011	1165	1011	1165	0	0.83	933	1073	933	1073	0	0.76	956	1124	956	1124	0	0.77
C1	1323	1011	1165	421	575	0	1.39	933	1073	375	515	0	1.32	956	1124	406	574	0	1.32
C1C2	1323	1011	1165	401	538	10	1.44	933	1073	335	445	12	1.36	956	1124	382	529	9	1.35
C1C2C3	1323	1011	1165	401	538	10	1.48	933	1073	335	445	12	1.46	956	1124	382	529	9	1.50
S1	1323	1011	1165	248	331	46	1.58	933	1073	209	276	59	1.46	956	1124	225	303	56	1.53
S1S2	1323	1011	1165	237	317	50	1.58	933	1073	200	262	66	1.50	956	1124	194	257	83	1.59

Table B.7: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in rl1323.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0
C1	590	0	0	0	0	1149	0.0	558	0	0	0	0	1045	0.0	550	0	0	0	0	1093	0.0
C1C2	598	12	0	0	0	1141	0.0	579	19	0	0	0	1048	0.0	559	15	0	0	0	1091	0.0
C1C2C3	598	12	0	0	0	1141	0.0	578	18	1	0	0	1044	0.0	559	13	1	0	0	1086	0.0
S1	0	0	0	763	0	1148	0.0	0	0	0	724	0	1055	0.0	0	0	0	731	0	1092	0.0
S1S2	0	0	0	764	10	1141	0.0	0	0	0	726	7	1050	0.5	0	0	0	738	24	1069	0.0

Table B.8: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in rl1323. The extra column is particular of DHI separation strategy (during separation).

vm1748

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	1748	1490	1756	1490	1756	0	1.32	1487	1837	1487	1837	0	1.36	1361	1586	1361	1586	0	1.21
C1	1748	1490	1756	642	908	0	2.39	1487	1837	808	1158	0	2.33	1361	1586	515	740	0	2.18
C1C2	1748	1490	1756	596	823	18	2.49	1487	1837	727	1005	32	2.47	1361	1586	480	680	6	2.28
C1C2C3	1748	1490	1756	596	823	18	2.72	1487	1837	727	1005	32	2.64	1361	1586	480	680	6	2.42
S1	1748	1490	1756	374	513	76	2.87	1487	1837	487	675	106	2.88	1361	1586	284	411	48	2.51
S1S2	1748	1490	1756	337	462	87	2.86	1487	1837	455	630	121	2.81	1361	1586	249	358	72	2.63

Table B.9: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in vm1748.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0
C1	848	0	0	0	0	1653	0.0	679	0	0	0	0	1659	0.0	846	0	0	0	0	1518	0.0
C1C2	866	28	0	0	0	1676	0.0	711	49	0	0	0	1690	0.0	859	22	0	0	0	1533	0.0
C1C2C3	866	28	0	0	0	1676	0.0	711	43	3	0	0	1680	0.0	859	20	1	0	0	1529	0.0
S1	0	0	0	1116	0	1692	0.0	0	0	0	1000	0	1729	0.0	0	0	0	1077	0	1525	0.0
S1S2	0	0	0	1123	30	1684	0.0	0	0	0	1003	29	1720	0.2	0	0	0	1081	31	1501	0.8

Table B.10: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in vm1748. The extra column is particular of DHI separation strategy (during separation).

rl5934

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	5934	4303	4871	4303	4871	0	5.67	4101	4651	4101	4651	0	5.31	3970	4424	3970	4424	0	5.43
C1	5934	4303	4871	1477	2045	0	11.10	4101	4651	1533	2083	0	10.06	3970	4424	1127	1581	0	10.44
C1C2	5934	4303	4871	1312	1754	54	11.50	4101	4651	1415	1873	44	10.45	3970	4424	1014	1381	42	10.60
C1C2C3	5934	4303	4871	1312	1754	54	11.61	4101	4651	1415	1873	43	10.71	3970	4424	1014	1381	41	10.57
S1	5934	4303	4871	800	1067	255	12.32	4101	4651	877	1123	266	11.64	3970	4424	624	848	189	10.99
S1S2	5934	4303	4871	750	990	300	12.57	4101	4651	800	1026	296	11.83	3970	4424	560	756	237	11.25

Table B.11: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in rl5934.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	2826	0	0	0	0	4864	0	2568	0	0	0	0	4598	0	2843	0	0	0	0	4389	0
C1C2	2904	87	0	0	0	4854	0	2622	64	0	0	0	4603	0	2897	59	0	0	0	4374	0
C1C2C3	2904	85	1	0	0	4852	0	2620	58	4	0	0	4585	0	2896	54	3	0	0	4363	0
S1	0	0	0	3503	0	4790	0	0	0	0	3224	0	4563	0	0	0	0	3346	0	4361	0
S1S2	0	0	0	3511	42	4757	0	0	0	0	3247	54	4542	0	0	0	0	3355	55	4324	0

Table B.12: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in rl5934. The extra column is particular of DHI separation strategy (during separation).

usa13509

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	13509	9084	9990	9084	9990	0	18.58	8015	8735	8015	8735	0	14.58	7245	7992	7245	7992	0	12.73
C1	13509	9084	9990	2017	2923	0	44.98	8015	8735	1723	2443	0	35.07	7245	7992	1644	2391	0	28.85
C1C2	13509	9084	9990	1891	2725	1	44.86	8015	8735	1623	2284	1	35.70	7245	7992	1490	2148	0	29.28
C1C2C3	13509	9084	9990	1891	2725	1	45.12	8015	8735	1623	2284	1	35.42	7245	7992	1490	2148	0	30.03
S1	13509	9084	9990	882	1347	213	49.39	8015	8735	846	1249	196	39.33	7245	7992	718	1107	146	32.58
S1S2	13509	9084	9990	705	1046	381	50.35	8015	8735	717	1027	349	39.16	7245	7992	587	885	255	32.45

Table B.13: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in usa13509.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	7067	0	0	0	0	10005	0	6292	0	0	0	0	8751	0	5601	0	0	0	0	7971	0
C1C2	7123	70	0	0	0	9969	0	6334	58	0	0	0	8722	0	5666	89	0	0	0	7936	0
C1C2C3	7123	64	3	0	0	9960	0	6334	56	1	0	0	8718	0	5666	83	3	0	0	7926	0
S1	0	0	0	8202	0	9753	0	0	0	0	7169	0	8628	0	0	0	0	6527	0	7771	0
S1S2	0	0	0	8223	156	9610	0	0	0	0	7179	119	8503	0	0	0	0	6541	117	7670	0

Table B.14: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in usa13509. The extra column is particular of DHI separation strategy (during separation).

d15112

Shrinking: Preprocess and Extra

		Gen1						Gen2						Gen3					
		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess		Support graph		Shrunk graph		Preprocess	
Shrinking	$\|V\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time	$\|\bar{V}\|$	$\|\bar{E}\|$	$\|\bar{V}\|$	$\|\bar{E}\|$	#Q	Time
NO	15112	9075	9866	9075	9866	0	18.73	7682	8322	7682	8322	0	14.25	9393	10378	9393	10378	0	19.63
C1	15112	9075	9866	1793	2584	0	43.70	7682	8322	1600	2240	0	32.45	9393	10378	2170	3155	0	46.91
C1C2	15112	9075	9866	1656	2373	0	44.42	7682	8322	1510	2097	1	32.28	9393	10378	1994	2876	5	47.70
C1C2C3	15112	9075	9866	1656	2373	0	45.28	7682	8322	1510	2097	1	32.70	9393	10378	1994	2876	5	48.93
S1	15112	9075	9866	785	1203	176	48.52	7682	8322	809	1164	197	34.90	9393	10378	941	1460	163	53.01
S1S2	15112	9075	9866	658	977	305	48.12	7682	8322	690	964	297	36.06	9393	10378	796	1213	285	53.44

Table B.15: Graph sizes, number of obtained Q sets and running time of the preprocess by shrinking strategy and OP instance generation in d15112.

	Gen1							Gen2							Gen3						
	Preprocess						DHI	Preprocess						DHI	Preprocess						DHI
Shrinking	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra	C1	C2	C3	S1	S2	H	Extra
NO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	7282	0	0	0	0	9933	0	6082	0	0	0	0	8382	0	7223	0	0	0	0	10350	0
C1C2	7345	74	0	0	0	9864	0	6120	52	0	0	0	8357	0	7299	100	0	0	0	10305	0
C1C2C3	7345	70	2	0	0	9858	0	6120	52	0	0	0	8357	0	7299	98	1	0	0	10302	0
S1	0	0	0	8290	0	9693	0	0	0	0	6873	0	8270	0	0	0	0	8452	0	10118	0
S1S2	0	0	0	8301	116	9586	0	0	0	0	6893	99	8165	0	0	0	0	8468	129	10025	0

Table B.16: Number of applications of each rule in the preprocess by shrinking strategy and OP instance generation in d15112. The extra column is particular of DHI separation strategy (during separation).

pr76

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	46	0.4	46	0.5	3030	1.5	16	0.6	16	0.7	860	1.1	37	0.3	37	0.4	2750	1.4
	C1	11	0.2	11	0.2	610	0.5	10	0.2	10	0.2	390	0.5	10	0.2	10	0.2	620	0.4
	C1C2	11	0.2	11	0.2	610	0.5	10	0.2	10	0.2	390	0.5	10	0.2	10	0.2	620	0.5
	C1C2C3	11	0.2	11	0.2	610	0.5	10	0.2	10	0.3	390	0.4	10	0.2	10	0.3	620	0.5
	S1	8	0.1	8	0.2	420	0.3	2	0.1	2	0.1	60	0.1	6	0.1	6	0.1	290	0.3
	S1S2	6	0.1	6	0.2	340	0.3	2	0.1	2	0.1	60	0.2	6	0.1	6	0.1	290	0.2
DH	NO	9	0.2	9	0.2	220	0.3	8	0.3	8	0.3	170	0.4	11	0.2	11	0.2	510	0.4
	C1	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.2	6	0.2	6	0.2	190	0.3
	C1C2	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.3	6	0.1	6	0.2	190	0.2
	C1C2C3	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.2	6	0.2	6	0.2	190	0.3
	S1	7	0.1	7	0.2	390	0.3	3	0.1	3	0.1	80	0.1	4	0.1	4	0.1	110	0.2
	S1S2	7	0.1	7	0.1	370	0.3	3	0.1	3	0.1	80	0.2	4	0.1	4	0.1	110	0.2
DHI	NO	9	0.2	9	0.2	220	0.3	8	0.3	8	0.3	170	0.4	11	0.2	11	0.2	510	0.4
	C1	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.3	6	0.2	6	0.2	190	0.3
	C1C2	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.3	6	0.2	6	0.2	190	0.3
	C1C2C3	6	0.2	5	0.2	220	0.3	7	0.2	7	0.2	230	0.3	6	0.2	6	0.2	190	0.3
	S1	7	0.1	7	0.2	390	0.3	3	0.1	3	0.1	80	0.1	4	0.1	4	0.1	110	0.2
	S1S2	7	0.1	7	0.1	370	0.3	3	0.1	3	0.1	80	0.1	4	0.1	4	0.2	110	0.2
EPG	NO	48	0.6	48	0.7	3090	2.0	20	0.8	20	0.8	960	1.2	39	0.5	39	0.6	2910	1.4
	C1	14	0.4	14	0.4	680	0.7	10	0.3	10	0.3	390	0.5	12	0.3	12	0.4	620	0.6
	C1C2	14	0.3	14	0.4	680	0.6	10	0.3	10	0.3	390	0.5	12	0.3	12	0.4	620	0.6
	C1C2C3	14	0.4	14	0.4	680	0.6	10	0.3	10	0.4	390	0.5	12	0.3	12	0.4	620	0.5
	S1	10	0.2	10	0.3	540	0.4	3	0.2	3	0.2	80	0.2	6	0.2	6	0.3	370	0.3
	S1S2	7	0.2	7	0.3	440	0.3	3	0.2	3	0.2	80	0.2	6	0.2	6	0.2	370	0.3

Table B.17: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in pr76.

att532

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	94	65.3	94	67.8	8550	67.4	32	54.2	32	53.9	2630	56.0	30	65.6	30	65.9	2750	67.4
	C1	43	5.4	43	6.1	3560	7.8	17	17.0	17	17.4	1470	17.6	21	20.4	21	20.6	1850	21.7
	C1C2	40	4.4	40	5.3	3250	6.0	21	14.1	21	14.1	1570	15.2	27	17.0	27	17.2	2080	18.3
	C1C2C3	41	4.9	41	5.4	3350	7.3	21	14.6	21	14.9	1570	15.2	27	16.7	27	17.5	2080	17.5
	S1	27	3.6	27	4.2	1820	4.3	40	7.2	40	8.0	2490	8.4	38	7.6	38	8.1	2160	8.8
	S1S2	31	3.2	31	3.7	2220	4.2	48	5.3	48	5.9	3240	7.2	39	6.7	39	6.6	2190	8.5
DH	NO	52	5.6	51	6.1	3280	7.8	56	8.2	55	8.6	2500	9.5	54	11.3	54	11.5	2420	12.7
	C1	34	2.2	34	2.4	1950	3.8	41	3.9	40	4.0	1940	5.4	44	5.6	44	6.1	2140	6.8
	C1C2	34	1.9	34	2.4	2060	3.1	44	3.4	43	4.1	2110	4.5	46	4.8	46	5.2	2320	6.3
	C1C2C3	35	1.9	35	2.0	2160	3.5	44	3.7	43	4.0	2110	5.5	46	4.5	46	4.8	2320	6.1
	S1	29	1.6	29	1.8	1970	2.8	59	1.9	58	2.5	3290	3.7	53	2.4	53	2.6	2970	4.3
	S1S2	31	1.3	31	1.6	2170	2.5	63	2.0	62	2.6	3600	4.5	53	2.3	53	2.8	2890	4.3
DHI	NO	52	5.6	51	6.1	3280	7.8	56	8.2	55	8.6	2500	9.5	54	11.3	54	11.5	2420	12.7
	C1	34	2.9	34	3.5	1950	3.7	41	4.6	40	5.3	1940	5.6	44	5.9	44	6.0	2140	7.4
	C1C2	34	2.3	34	2.7	2060	3.6	44	3.6	43	4.3	2110	4.6	46	5.0	46	5.8	2320	5.8
	C1C2C3	35	2.4	35	2.5	2160	4.3	44	4.5	43	5.0	2110	6.1	46	5.8	46	6.2	2320	7.3
	S1	29	1.4	29	1.7	1970	2.5	59	2.3	58	2.8	3290	4.7	53	2.4	53	2.9	2970	4.4
	S1S2	33	1.6	33	1.8	2290	2.9	63	1.9	62	2.5	3600	4.1	57	2.6	53	3.4	2890	4.6
EPG	NO	349	9.4	349	11.6	30780	25.7	283	8.7	283	11.3	23110	19.6	288	8.2	288	9.6	24040	21.6
	C1	97	4.2	97	5.3	7550	7.5	122	5.0	122	5.8	8930	10.3	145	5.9	145	7.4	11400	11.5
	C1C2	84	4.1	84	5.0	6510	7.4	118	4.8	118	6.1	8600	9.6	137	5.4	137	6.5	10620	11.3
	C1C2C3	85	3.9	85	4.4	6640	7.6	118	4.6	118	5.3	8600	9.2	137	5.4	137	6.8	10620	10.7
	S1	53	3.3	53	3.9	4020	5.4	93	3.9	93	5.0	6370	6.8	99	3.7	99	4.4	6970	7.2
	S1S2	54	3.4	54	3.7	4040	5.7	88	3.5	88	4.0	5870	6.5	93	4.2	93	5.0	6550	7.7

Table B.18: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in att532.

vm1084

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	102	538.5	102	539.8	8350	545.3	156	211.2	156	212.3	14470	224.9	116	291.9	116	293.6	10280	303.0
	C1	40	42.9	40	44.3	3070	45.0	32	47.6	32	47.8	2540	49.8	33	36.5	33	36.9	2410	39.0
	C1C2	44	30.6	44	30.8	3110	34.1	37	40.4	37	40.2	2580	43.2	38	30.7	38	31.8	2700	32.8
	C1C2C3	44	31.3	44	31.2	3110	34.8	36	41.0	36	41.2	2540	43.4	38	31.2	38	32.1	2700	33.5
	S1	56	11.4	56	11.9	3630	14.7	63	21.0	63	21.2	3620	24.1	53	12.7	53	13.7	3450	15.3
	S1S2	63	11.0	63	11.9	4220	14.6	71	20.4	71	20.4	4290	24.1	61	11.3	61	12.5	4180	14.5
DH	NO	80	30.3	80	31.9	4260	32.8	115	25.2	115	26.5	5380	29.3	73	18.7	72	19.8	3970	21.2
	C1	57	8.2	57	9.2	3030	10.4	89	7.1	89	8.8	4080	10.2	61	5.8	58	7.0	3030	8.2
	C1C2	60	6.5	60	7.5	3320	8.9	88	6.4	88	7.8	3980	10.3	62	5.4	59	6.2	3260	8.4
	C1C2C3	60	6.7	60	7.8	3320	8.9	87	6.0	87	7.3	3940	9.1	62	5.4	59	6.5	3260	8.1
	S1	69	3.4	67	4.7	4250	7.1	96	4.0	96	6.1	5090	8.0	74	2.7	73	3.8	4690	6.3
	S1S2	81	3.2	79	4.9	5250	7.1	108	3.4	108	5.7	6070	7.4	82	3.0	81	4.6	5360	7.4
DHI	NO	80	30.3	80	31.9	4260	32.8	115	25.2	115	26.5	5380	29.3	73	18.7	72	19.8	3970	21.2
	C1	57	10.0	57	11.0	3030	12.2	89	9.8	89	12.2	4080	12.2	61	7.7	58	8.7	3030	10.2
	C1C2	60	7.8	60	8.9	3320	10.2	88	8.3	88	9.7	3980	11.6	62	6.9	59	7.4	3260	10.1
	C1C2C3	60	8.3	60	9.2	3320	10.8	87	8.4	87	9.2	3940	11.9	62	7.3	59	8.8	3260	9.3
	S1	69	3.8	67	4.9	4250	7.6	96	4.0	96	5.4	5090	8.3	74	3.3	73	4.6	4690	7.3
	S1S2	81	3.7	79	5.3	5250	7.7	108	4.1	108	6.1	6070	8.9	82	3.0	81	4.8	5360	7.0
EPG	NO	690	24.0	690	32.5	62170	73.3	652	30.4	652	38.5	56770	81.7	493	23.5	493	28.7	44000	59.2
	C1	186	9.9	186	12.3	14770	21.9	222	14.1	222	16.6	16850	31.3	170	10.3	170	13.1	13660	20.7
	C1C2	167	9.4	167	11.6	13470	20.5	203	13.7	203	16.7	14900	28.5	154	9.1	154	11.8	12370	18.0
	C1C2C3	167	9.4	167	11.5	13470	20.7	202	13.8	202	16.5	14860	29.5	155	10.2	155	13.5	12490	19.0
	S1	117	7.6	117	9.9	9070	14.3	156	9.7	156	12.2	10890	20.3	113	6.1	113	7.7	8510	12.6
	S1S2	118	7.6	118	9.5	9270	14.8	153	9.2	153	11.6	10790	18.6	110	7.1	110	9.2	8240	13.2

Table B.19: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in vm1084.

rl1323

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	100	582.4	100	582.4	8930	589.9	27	441.1	27	442.9	1660	441.2	250	288.0	250	291.2	20440	307.4
	C1	28	83.7	28	83.6	2630	87.1	26	58.0	26	58.1	1710	59.9	62	50.2	62	51.3	4030	53.4
	C1C2	36	76.5	36	77.2	2880	79.3	36	43.2	36	43.1	1930	45.7	69	44.9	69	45.1	4260	49.4
	C1C2C3	36	77.2	36	78.5	2880	79.4	36	43.0	36	43.5	1930	45.0	69	45.9	69	46.8	4260	49.6
	S1	67	24.4	67	25.6	4650	27.6	72	22.0	72	22.0	4450	26.2	89	12.4	89	14.1	5130	15.6
	S1S2	70	23.0	70	24.4	4910	26.0	77	21.8	77	23.6	4970	24.5	111	11.2	111	13.0	6730	15.8
DH	NO	190	38.1	185	40.4	7330	44.2	173	32.1	170	34.0	7580	38.8	138	34.8	138	36.3	5450	39.3
	C1	131	11.3	127	13.3	6280	16.2	114	8.4	111	9.9	5530	13.4	111	9.1	111	10.7	4970	13.5
	C1C2	133	11.0	130	13.7	6480	15.5	113	7.5	110	9.5	5670	12.3	114	8.3	114	10.5	5290	12.4
	C1C2C3	133	10.1	130	12.6	6480	14.6	113	7.5	110	9.5	5660	12.1	114	8.0	114	10.3	5290	11.7
	S1	129	6.0	127	8.8	7570	11.5	120	4.4	118	7.0	7550	9.9	114	4.7	114	7.3	5870	9.3
	S1S2	130	5.5	128	8.2	7650	11.1	127	4.7	125	7.0	8110	11.4	139	3.9	139	6.8	7860	9.4
DHI	NO	190	38.1	185	40.4	7330	44.2	173	32.1	170	34.0	7580	38.8	138	34.8	138	36.3	5450	39.3
	C1	131	14.5	127	16.3	6280	19.1	114	11.4	111	13.2	5530	15.8	111	11.0	111	12.3	4970	15.1
	C1C2	133	13.7	130	16.0	6480	18.1	113	8.4	110	10.2	5670	12.6	114	9.3	114	10.9	5290	13.4
	C1C2C3	133	13.8	130	16.4	6480	17.9	113	9.5	110	12.0	5660	13.2	114	10.4	114	12.2	5290	14.5
	S1	129	6.8	127	9.4	7570	12.2	120	5.3	118	7.8	7550	11.2	114	5.3	114	7.9	5870	10.0
	S1S2	130	6.2	128	8.9	7650	11.1	132	5.5	125	7.2	8110	12.2	139	4.5	139	7.4	7860	9.9
EPG	NO	828	32.4	828	44.5	70130	84.7	803	29.9	803	41.2	68310	79.9	765	27.7	765	38.4	59420	70.0
	C1	280	15.1	280	19.9	21700	30.5	250	14.6	250	18.9	19280	28.2	280	13.5	280	17.7	18020	27.5
	C1C2	272	14.3	272	18.4	20960	29.1	233	13.2	233	16.8	17690	26.1	272	12.6	272	16.5	17370	25.4
	C1C2C3	272	14.1	272	18.2	20960	28.9	233	13.5	233	17.0	17680	26.5	272	13.3	272	17.4	17370	25.7
	S1	194	11.0	194	14.5	14790	20.8	175	9.1	175	12.7	13170	17.9	197	9.0	197	12.5	11890	17.1
	S1S2	189	10.3	189	13.1	14260	21.5	175	9.5	175	12.0	13270	19.4	184	8.5	184	11.4	12310	16.4

Table B.20: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in rl1323.

vm1748

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	130	2198.8	130	2197.0	11200	2222.3	40	1068.1	40	1070.7	2970	1071.0	188	1140.2	188	1144.0	15540	1176.5
	C1	47	175.9	47	176.9	3250	181.9	28	227.5	28	228.1	1930	231.3	82	138.3	82	140.6	5740	147.2
	C1C2	63	152.6	63	155.1	3660	157.6	60	175.3	60	175.7	3000	181.2	86	114.5	86	116.0	5820	123.7
	C1C2C3	63	155.5	63	156.5	3660	161.9	60	175.1	60	176.4	3000	179.8	86	116.0	86	116.7	5820	126.0
	S1	112	95.9	112	98.9	6230	102.7	128	80.4	128	83.1	6030	87.0	82	32.5	82	34.1	4250	38.1
	S1S2	112	62.3	112	65.9	5980	67.2	143	70.2	143	73.6	7200	77.1	102	25.5	102	27.3	6070	32.6
DH	NO	186	78.2	185	82.5	7430	86.5	218	86.3	217	91.1	9940	95.5	143	63.4	143	66.3	6670	69.9
	C1	156	24.0	154	27.9	6770	30.6	185	50.5	184	54.8	8740	59.2	118	17.2	118	19.4	5300	23.0
	C1C2	164	20.6	162	24.2	7400	28.6	196	36.4	195	40.0	9580	47.0	118	14.1	118	17.3	5350	19.1
	C1C2C3	164	20.6	162	24.4	7400	28.7	197	37.2	196	40.9	9630	47.8	118	13.9	118	17.0	5350	19.0
	S1	175	12.6	173	16.8	8940	20.9	220	16.8	220	21.1	11170	28.1	113	9.1	112	11.9	5270	14.4
	S1S2	193	11.0	191	15.6	10340	20.1	238	15.4	238	21.0	12570	26.0	138	8.4	137	11.6	7440	15.1
DHI	NO	186	78.2	185	82.5	7430	86.5	218	86.3	217	91.1	9940	95.5	143	63.4	143	66.3	6670	69.9
	C1	156	32.5	154	35.6	6770	40.1	185	67.2	184	70.8	8740	76.2	118	21.6	118	24.5	5300	26.7
	C1C2	164	27.7	162	30.9	7400	36.2	196	50.8	195	55.8	9580	59.6	118	18.4	118	20.2	5350	25.2
	C1C2C3	164	29.4	162	32.7	7400	37.5	197	52.3	196	56.3	9630	62.2	118	20.0	118	22.2	5350	25.9
	S1	175	13.4	173	17.0	8940	21.9	220	18.6	220	24.3	11170	27.8	113	9.2	112	12.2	5270	14.0
	S1S2	193	12.6	191	16.6	10340	22.1	257	22.4	238	27.3	12570	34.4	144	10.5	137	13.8	7440	16.8
EPG	NO	1217	95.3	1217	120.5	109120	227.6	1140	64.2	1140	88.5	101750	161.5	1038	61.1	1038	81.1	92340	176.9
	C1	449	23.8	449	33.5	36440	64.3	513	26.8	513	38.0	41990	69.7	318	18.7	318	25.7	23820	45.5
	C1C2	426	22.4	426	32.2	33510	59.8	479	25.9	479	36.4	37390	63.3	304	19.0	304	25.8	22540	44.3
	C1C2C3	426	22.5	426	31.5	33510	60.6	478	24.8	478	36.2	37300	61.4	304	18.2	304	24.2	22540	45.2
	S1	312	21.5	312	29.1	23320	48.2	372	18.0	372	25.5	26540	43.9	207	13.3	207	17.6	14150	30.7
	S1S2	287	16.3	287	22.6	20970	40.0	355	18.2	355	26.8	24860	41.7	200	12.8	200	17.0	14280	27.8

Table B.21: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in vm1748.

rl5934

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	21	7923.7	21	7914.3	1760	7942.1	60	12227.3	60	12243.7	5060	12233.7	307	7466.6	307	7486.2	28100	7530.7
	C1	19	991.6	19	997.7	1630	995.7	27	700.9	27	705.3	2410	707.9	57	877.0	57	879.1	3650	888.1
	C1C2	73	818.8	73	826.3	3510	829.4	71	622.5	71	629.4	4400	635.0	95	721.4	95	725.4	5300	737.8
	C1C2C3	73	816.8	73	825.1	3510	826.7	70	619.6	70	626.9	4370	631.1	94	722.7	94	729.4	5260	735.3
	S1	270	361.5	270	379.7	16660	393.6	285	528.9	285	548.7	17730	562.3	222	331.0	222	347.2	12170	354.4
	S1S2	315	334.5	315	356.5	20630	372.8	315	451.3	315	475.5	19890	485.0	270	275.3	270	293.9	16270	304.2
DH	NO	611	664.2	602	706.9	28970	728.0	665	575.3	660	623.0	30590	644.1	407	580.5	402	609.6	19770	621.7
	C1	415	126.3	403	153.6	22810	173.0	444	108.8	438	141.1	24660	158.3	291	72.5	287	92.5	15110	103.7
	C1C2	435	106.7	423	137.6	23640	155.2	460	97.6	454	131.2	25640	149.9	306	61.5	302	80.8	16060	94.3
	C1C2C3	435	106.3	423	137.5	23640	154.8	459	99.0	453	133.0	25650	149.7	304	61.6	300	81.1	15920	93.8
	S1	514	60.8	503	97.3	32830	117.6	519	60.8	514	97.9	33250	120.2	336	45.5	333	68.4	18080	79.2
	S1S2	549	58.1	538	95.5	35930	120.9	544	55.2	539	96.0	34370	115.7	378	40.4	375	66.8	21840	81.0
DHI	NO	611	664.2	602	706.9	28970	728.0	665	575.3	660	623.0	30590	644.1	407	580.5	402	609.6	19770	621.7
	C1	415	179.8	403	209.3	22810	227.2	444	188.8	438	220.5	24660	239.5	291	102.5	287	122.0	15110	132.6
	C1C2	435	149.1	423	179.8	23640	196.3	460	167.4	454	201.2	25640	218.1	306	83.6	302	105.4	16060	115.4
	C1C2C3	435	153.6	423	184.1	23640	203.5	459	173.0	453	205.3	25650	223.4	304	86.5	300	107.2	15920	119.2
	S1	514	67.7	503	104.4	32830	124.3	519	71.7	514	109.7	33250	130.4	336	49.7	333	71.7	18080	84.6
	S1S2	549	67.9	538	104.8	35930	129.3	544	66.5	539	105.9	34370	126.9	378	43.8	375	70.0	21840	82.8
EPG	NO	3661	734.5	3661	991.4	329310	1343.9	3434	777.9	3434	1044.9	303080	1370.5	3293	880.3	3293	1101.5	294840	1500.7
	C1	1011	103.6	1011	177.1	83650	282.0	1022	96.1	1022	172.9	83060	275.9	724	84.8	724	132.1	53500	202.8
	C1C2	927	91.8	927	159.9	75220	257.5	967	89.9	967	160.2	77540	253.4	673	78.3	673	123.5	48880	185.3
	C1C2C3	927	90.5	927	159.0	75220	258.2	964	87.3	964	159.4	77290	249.3	677	76.5	677	124.7	49080	184.3
	S1	753	62.7	753	116.9	58590	185.1	786	70.7	786	130.9	60470	205.3	536	54.2	536	90.3	36190	130.5
	S1S2	743	57.6	743	113.0	57470	176.9	746	68.1	746	123.7	56770	194.0	532	52.5	532	88.2	36470	127.7

Table B.22: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in rl5934.

usa13509

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	1162	46201.1	1162	46550.1	108550	46915.2	1183	22163.0	1183	22528.4	109110	22903.0	676	33770.8	676	34109.3	60910	34008.7
	C1	95	4798.9	95	4827.7	7930	4857.0	229	1080.0	229	1138.9	19380	1221.3	180	2369.9	180	2410.5	15090	2467.2
	C1C2	94	4277.8	94	4312.3	7760	4320.3	223	984.5	223	1037.1	18770	1124.6	175	2008.6	175	2044.2	14590	2106.0
	C1C2C3	94	4277.8	94	4296.2	7760	4340.0	223	985.4	223	1038.4	18770	1120.6	175	2004.9	175	2047.0	14590	2097.2
	S1	286	1005.5	286	1075.0	19750	1117.2	286	394.0	286	459.9	20560	518.8	223	504.3	223	556.7	15600	593.8
	S1S2	431	782.6	431	872.9	32630	954.6	419	327.7	419	425.3	32410	501.3	314	384.8	314	451.3	23810	503.6
DH	NO	412	4608.6	412	4683.7	25490	4762.3	407	3107.9	407	3247.8	24230	3187.2	306	2828.2	306	2944.0	19650	2866.9
	C1	309	387.7	309	464.5	20750	483.1	271	256.6	271	319.7	17900	338.5	231	264.3	231	312.1	15750	331.9
	C1C2	307	351.5	307	427.7	20550	453.4	271	239.9	271	302.8	18050	329.1	229	235.1	229	284.0	15550	302.4
	C1C2C3	307	357.9	307	436.7	20550	459.0	271	247.3	271	308.6	18050	333.0	229	234.2	229	282.1	15550	305.0
	S1	291	179.8	291	247.8	19630	276.0	283	127.2	283	195.4	19720	216.2	219	126.5	219	178.1	14950	188.6
	S1S2	476	157.9	476	267.0	36010	316.4	446	110.8	446	215.6	34050	255.2	335	101.6	335	174.1	25060	200.8
DHI	NO	412	4608.6	412	4683.7	25490	4762.3	407	3107.9	407	3247.8	24230	3187.2	306	2828.2	306	2944.0	19650	2866.9
	C1	309	567.6	309	640.7	20750	664.7	271	396.0	271	457.5	17900	478.7	231	371.7	231	422.0	15750	439.3
	C1C2	307	507.7	307	587.5	20550	610.7	271	362.1	271	422.3	18050	447.6	229	315.2	229	365.9	15550	382.5
	C1C2C3	307	548.8	307	615.4	20550	652.4	271	398.7	271	466.3	18050	483.4	229	353.1	229	400.4	15550	421.8
	S1	291	192.3	291	260.1	19630	288.8	283	138.7	283	201.5	19720	228.8	219	135.5	219	180.5	14950	198.7
	S1S2	476	166.4	476	281.2	36010	319.9	446	122.3	446	218.9	34050	259.9	335	107.8	335	180.0	25060	206.1
EPG	NO	5367	4061.4	5367	5338.9	504100	6940.9	4430	2618.3	4430	3721.2	409520	4787.3	3283	1818.2	3283	2543.6	306330	3267.8
	C1	978	261.3	978	484.3	85220	742.0	840	178.0	840	374.2	71890	588.5	717	182.0	717	329.1	62610	530.0
	C1C2	949	242.5	949	474.3	82350	708.4	811	170.2	811	364.8	69650	579.1	678	167.9	678	318.9	58930	479.3
	C1C2C3	946	246.0	946	479.4	82020	711.5	814	170.7	814	358.4	69610	574.5	678	167.5	678	316.4	58930	480.8
	S1	588	164.6	588	307.3	48480	449.1	547	117.9	547	244.5	45350	368.7	432	110.1	432	198.6	35650	299.8
	S1S2	674	159.6	674	309.6	56260	482.3	628	114.4	628	260.8	52280	378.4	478	106.3	478	214.5	39740	303.7

Table B.23: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in usa13509.

d15112

Separation and SEC Generation

		Gen1						Gen2						Gen3					
		Separation		SEC Generation				Separation		SEC Generation				Separation		SEC Generation			
		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)		(20 runs)		1x1 (10 runs)		10x10 (10 runs)	
Sep.	Shrinking	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time	#Q	Time	#SEC	Time	#SEC	Time
EH	NO	3284	42274.1	3284	43874.3	307840	43266.4	128	55626.9	128	55295.8	12080	56185.5	974	86877.2	974	87707.5	90920	86895.5
	C1	513	1864.8	513	2010.1	44790	2152.1	447	926.5	447	1048.0	38160	1138.6	127	4010.1	127	4026.4	11000	4095.6
	C1C2	495	1615.2	495	1763.4	43070	1879.5	435	837.7	435	959.8	37010	1026.9	131	3511.6	131	3542.8	11170	3580.4
	C1C2C3	495	1612.2	495	1758.9	43070	1878.9	435	839.7	435	963.4	37010	1037.3	131	3526.1	131	3541.4	11170	3609.6
	S1	348	466.2	348	564.1	28230	614.7	220	1349.8	220	1415.4	15440	1453.0	243	1035.1	243	1097.3	17380	1140.9
	S1S2	434	414.6	434	535.6	35780	591.0	320	1123.7	320	1219.4	24710	1257.3	342	874.3	342	979.0	26500	1015.7
DH	NO	354	4560.4	354	4691.7	23170	4658.4	401	3201.1	401	3341.2	24610	3315.4	361	5325.5	356	5591.6	24550	5295.0
	C1	242	362.2	242	428.9	17340	453.8	270	286.4	270	356.7	19180	383.3	269	487.5	264	569.3	18640	584.5
	C1C2	241	298.1	241	367.9	17290	385.3	266	254.9	266	331.4	18790	352.1	267	435.0	264	516.3	18660	530.1
	C1C2C3	241	300.5	241	373.6	17290	385.8	266	256.3	266	329.1	18790	355.5	267	437.0	264	514.2	18660	530.5
	S1	237	166.5	237	237.1	17340	249.4	278	132.0	278	202.2	20520	234.9	259	207.5	257	285.2	18230	303.9
	S1S2	380	144.1	380	253.8	30060	280.2	401	116.2	401	226.3	31170	263.4	389	182.7	387	301.0	29410	338.2
DHI	NO	354	4560.4	354	4691.7	23170	4658.4	401	3201.1	401	3341.2	24610	3315.4	361	5325.5	356	5591.6	24550	5295.0
	C1	242	519.5	242	581.7	17340	610.6	270	392.3	270	460.2	19180	491.0	269	710.6	264	783.9	18640	816.0
	C1C2	241	433.9	241	499.6	17290	521.6	266	348.1	266	417.6	18790	445.0	267	622.1	264	695.4	18660	724.1
	C1C2C3	241	460.3	241	526.5	17290	550.6	266	373.6	266	453.0	18790	472.9	267	675.2	264	760.0	18660	769.0
	S1	237	179.5	237	247.7	17340	263.5	278	144.6	278	220.5	20520	243.9	259	224.8	257	302.6	18230	322.9
	S1S2	380	150.6	380	250.2	30060	287.3	401	128.8	401	240.7	31170	271.0	389	199.1	387	315.5	29410	352.6
EPG	NO	5070	4634.5	5070	6114.2	479550	7891.6	4803	2714.3	4803	4082.5	447480	5683.1	4005	3486.2	4005	4604.9	375100	5701.0
	C1	809	257.9	809	483.6	72140	721.7	829	230.3	829	472.9	72890	716.8	869	297.0	869	539.5	75080	765.4
	C1C2	782	235.9	782	467.5	69660	662.9	785	213.9	785	428.8	68750	655.3	834	264.4	834	502.7	71410	699.9
	C1C2C3	780	248.8	780	475.8	69130	684.1	785	212.7	785	436.0	68750	650.3	834	268.4	834	500.2	71410	711.7
	S1	485	185.8	485	324.8	41720	458.0	545	155.8	545	309.6	46260	452.3	522	162.3	522	309.4	42500	408.4
	S1S2	555	181.3	555	334.7	47860	473.9	577	146.7	577	303.8	49440	450.9	561	159.9	561	311.2	47250	424.0

Table B.24: Number of obtained Q sets in separation, number of generated SECs when $k_{in} \times k_{out}$ is set to 1×1 and 10×10 and their running times by separation strategy, shrinking strategy and OP instance generation in d15112.

C Figures

In this section, we show some shrunk graphs obtained from the proposed shrinking strategies. The goal is to help the reader to obtain insights about the alternative strategies. We focus on the pr76-Gen1 SEC instance to do so. For each strategy two figures are presented, one preserving the geometry of the original OP instance and other showing the topological representation. In the figures, the vertices and the edges with value 1 are represented in black. The vertices and the edges with value in $[0.5, 1)$ are represented in red. The vertices in white and the edges with dashed style represent those with value in $(0, 0.5)$. The edges in blue and double lined style represent those with value greater than 1. The depot vertex of the OP, the vertex 1, is colored in green.

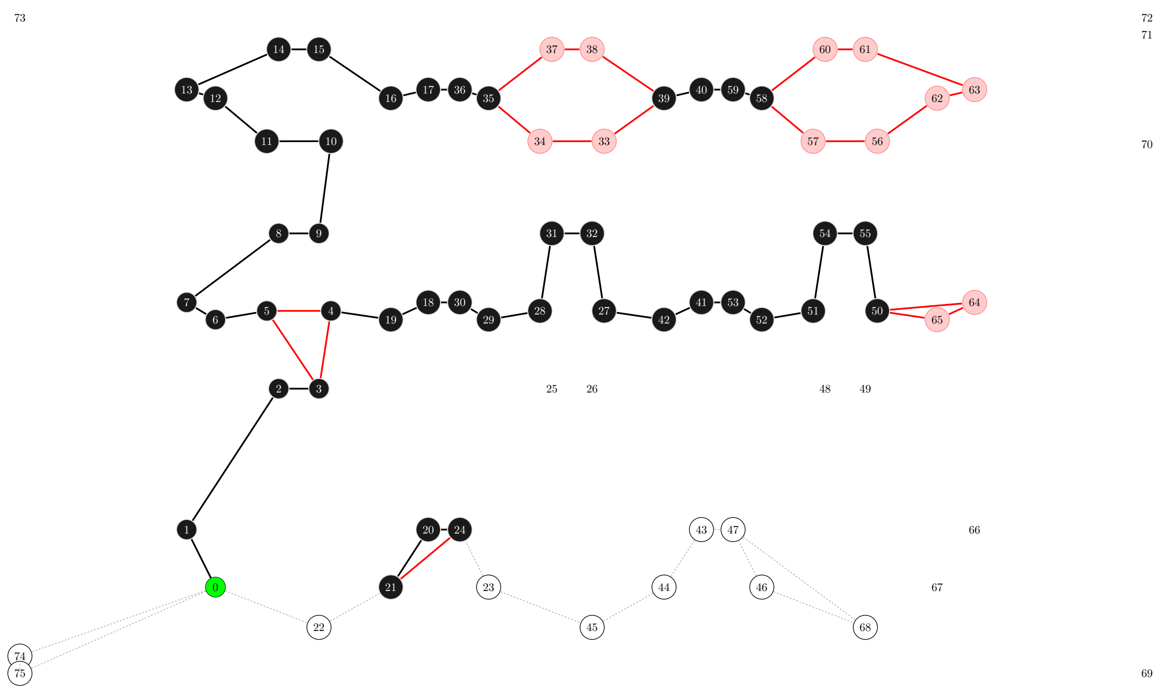


Figure C.1: Support graph of pr76-gen1 SEC instance

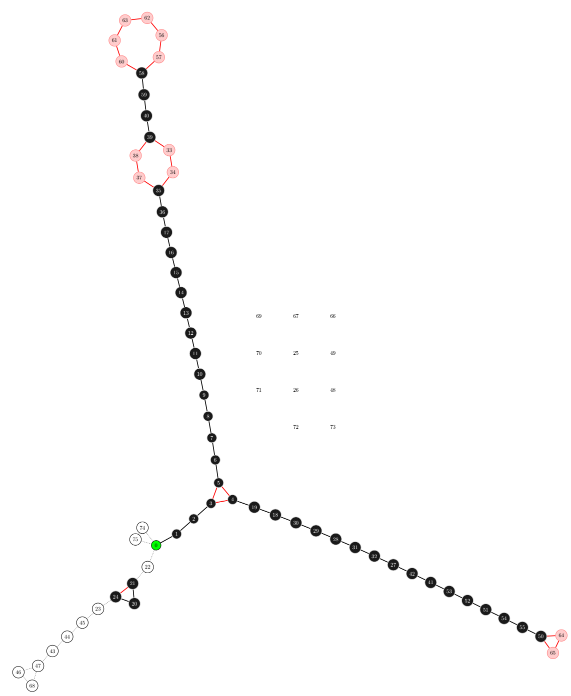


Figure C.2: Topological representation of the pr76-gen1 SEC instance

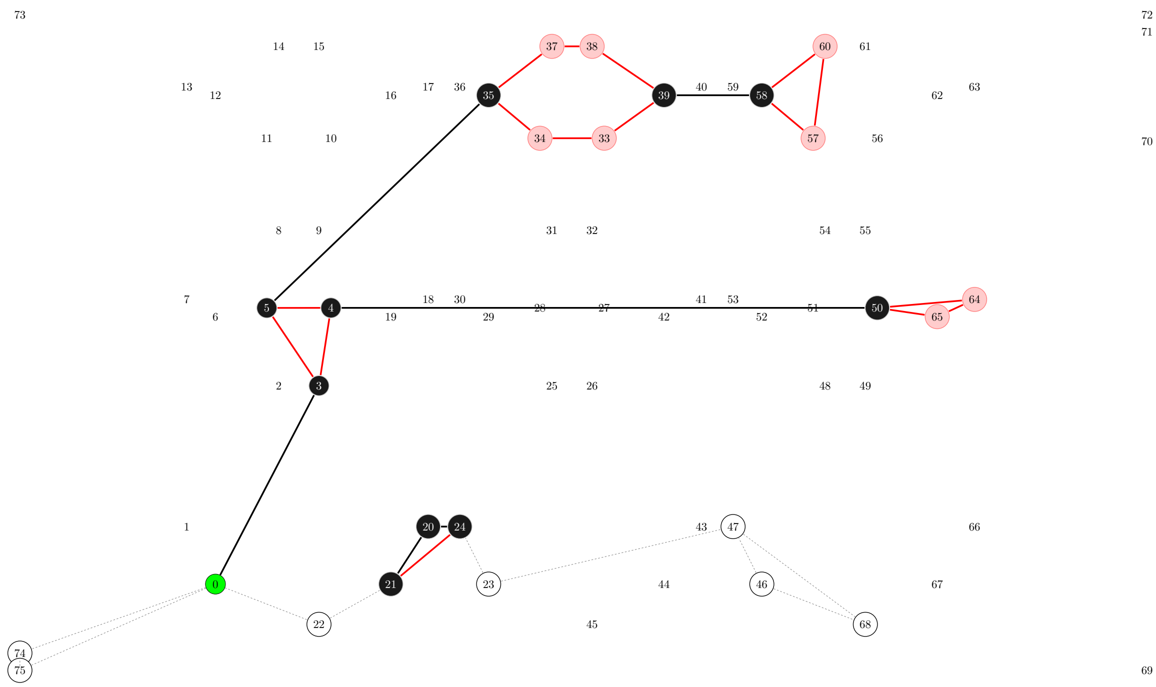


Figure C.3: Resulting graph after C1 shrinking strategy

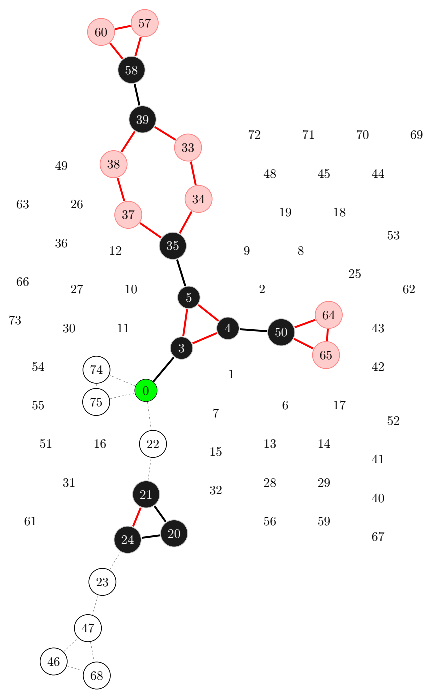


Figure C.4: Topological representation of the graph after C1 shrinking strategy

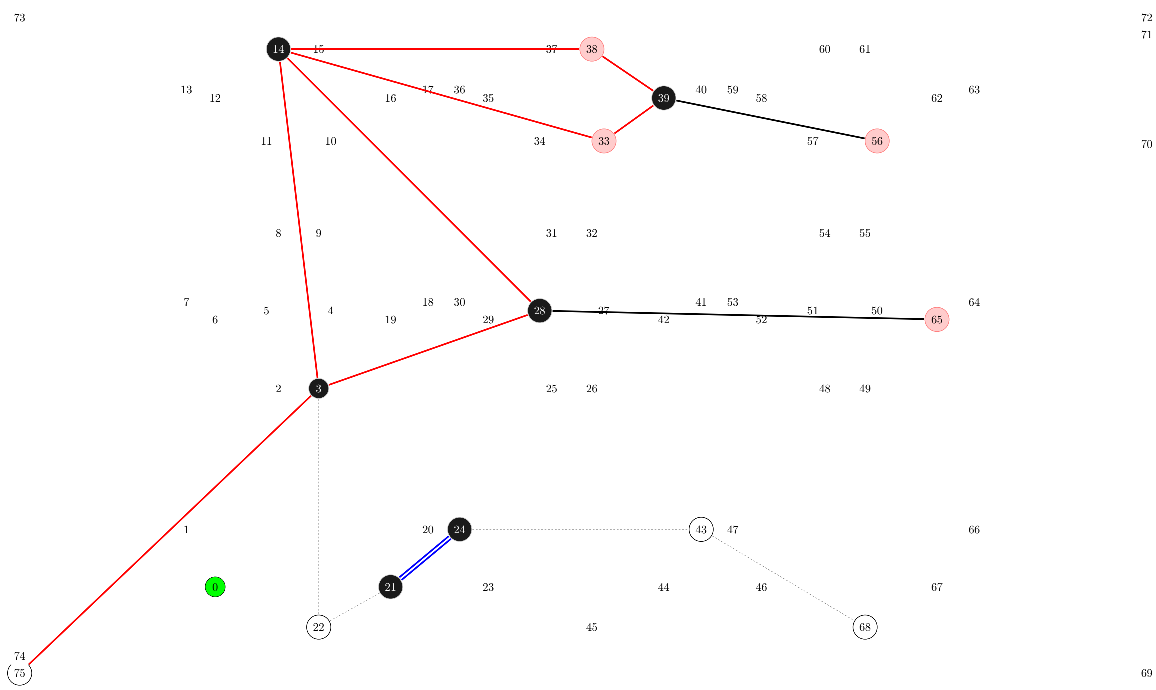


Figure C.5: Resulting graph after S1 shrinking strategy

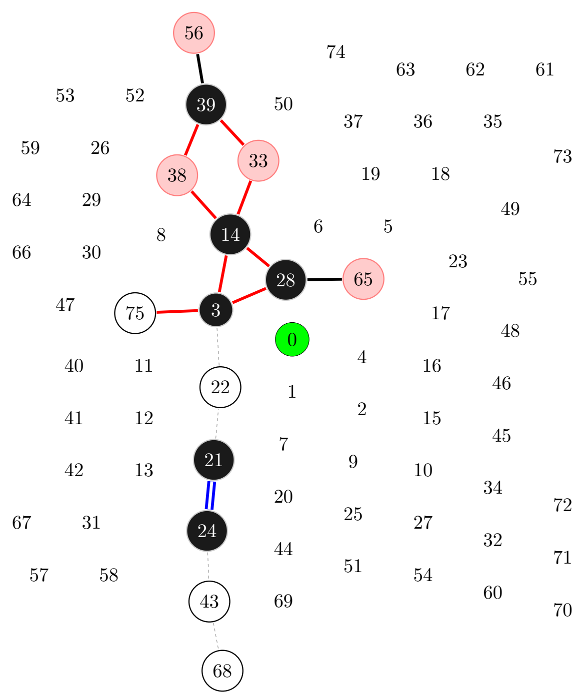


Figure C.6: Topological representation of the graph after S1 shrinking strategy

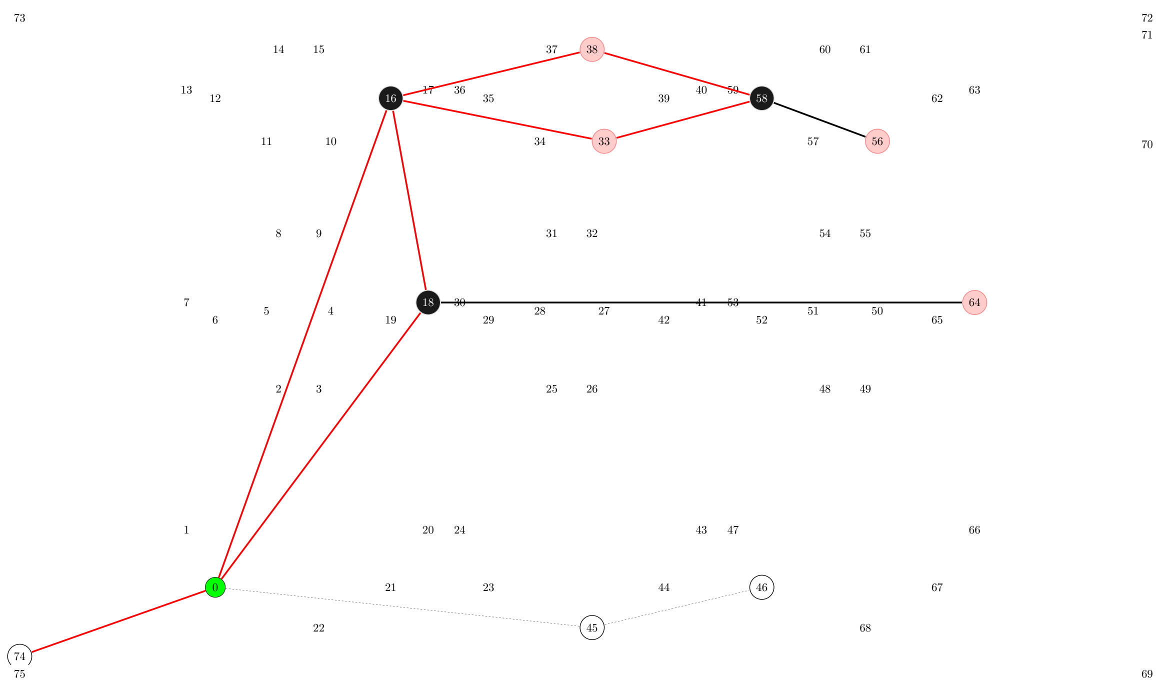


Figure C.7: Resulting graph after S1S2 shrinking strategy

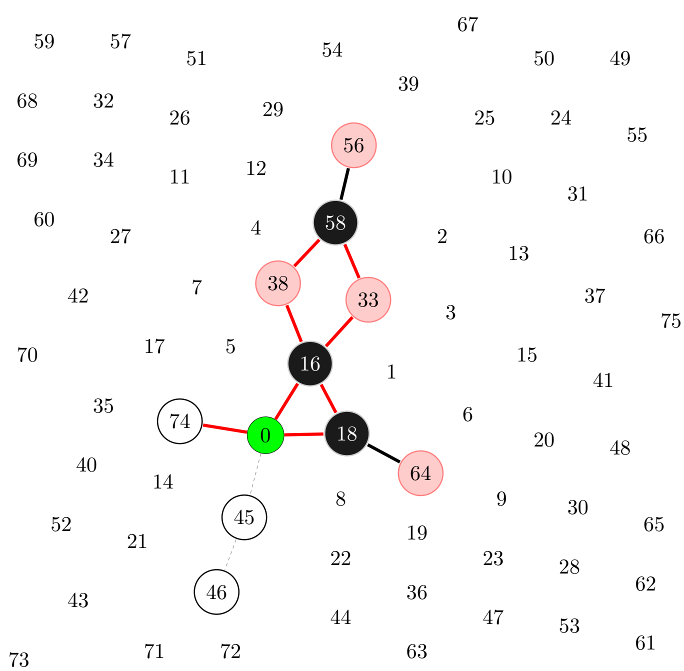


Figure C.8: Topological representation of the graph after S1S2 shrinking strategy