The Hamiltonian p-median Problem: Polyhedral Results and Branch-and-Cut Algorithm

Michele Barbato*† Luís Gouveia‡

Abstract

In this paper we study the Hamiltonian p-median problem, in which a weighted graph on n vertices is to be partitioned into p simple cycles of minimum total weight. We introduce two new families of valid inequalities for a formulation of the problem in the space of natural edge variables. Each one of the families forbids solutions to the 2-factor relaxation of the problem that have less than p cycles. The inequalities in one of the families are associated with large simple cycles of the underlying graph and generalize known inequalities associated to Hamiltonian cycles. The other family involves inequalities for the case with $p = \lfloor n/3 \rfloor$, associated with edge cuts and multi-cuts whose shores have specific cardinalities. We identify inequalities from both families that define facets of the polytope associated with the problem in the edge variables space. We design branch-and-cut algorithms based on these families of inequalities and on inequalities associated with 2-opt moves removing sub-optimal solutions. Computational experiments on benchmark instances show that our algorithm exhibits a comparable performance with respect to existing exact methods from the literature; moreover our algorithm solves to optimality new instance with up to 400 vertices.

Keywords— Combinatorial optimization; Hamiltonian p-median problem; valid inequality; polyhedral study; branch-and-cut algorithm

1 Introduction

Given a nonnegative integer p and an edge-weighted complete graph over n vertices, the *Hamiltonian p-median problem* (abbreviated to HpMP) is to find a minimum-weight partition of its vertices into p simple cycles. The HpMP has been first introduced by Branco and Coelho in [4] in the context of location-routing problems with real-world applications covering, among others, "schools location, milk stations and depot location for different industrial and commercial purposes". A generic application is illustrated by the assignment of p guards to n objects where it is assumed that each guard cycles among the assigned protection objects. Similarly, the assignment of p maintenance/inspection vehicles to maintain/inspect n machines can also be modelled as the HpMP. Besides its applications to the above location-routing problems, other applications

*corresponding author

†Dipartimento di Informatica "Giovanni Degli Antoni", Università degli Studi di Milano, via Celoria 18, 20133, Milan, Italy. E-mail: michele.barbato@unimi.it

‡Centro de Matemática, Aplicações Fundamentais e Investigação Operacional (CMAF-CIO), Faculdade de Ciências da Universidade de Lisboa, C6 - Piso 4 Lisboa, 1749-016, Portugal. E-mail: legouveia@fc.ul.pt

of the HpMP and its variations have subsequently been found in cutting problems [12] and laser multi-scanners [13].

The HpMP is related to several other combinatorial problems, among which we mention those relevant for our discussion. Firstly, as already noted in all previous studies on the problem, the HpMP for $p = 1$ corresponds to the well-known *Symmetric Traveling Salesman Problem* (abbreviated to STSP), showing that the HpMP is **NP**-hard if p is part of the input. Similarly, when $n = 3p$ the HpMP corresponds to the *triangle packing problem* which is also known to be **NP**-Hard [10]. Instead, if p is not fixed the HpMP reduces to the 2-factor problem (*i.e.,* the problem of determining a minimum weight 2-regular subgraph of a graph G) which is known to be solvable in polynomial time [6].

We point out that the HpMP has appeared in two variants in the literature. In one variant, usually studied in works with formulations using binary edge variables, 2-cycles are not allowed. This may lead to sub-optimal solutions with respect to the other variant, allowing 2-cycle solutions and considered by [4] where the HpMP was initially introduced. The variant in which 2-cycles are allowed is usually modelled with directed models, see *e.g.*, [2], although models using undirected edge variables with values in $\{0, 1, 2\}$ might also be derived, following an approach similar to the one of [3] for a multi-depot routing problem.

The HpMP variant not allowing 2-cycles has been modelled using only edge variables in [17]. The paper [14] proposes valid inequalities defined on the same set of variables, compares several formulations, including formulations only involving edge variables with other formulations using additional variables, and produces the first computational study with formulations using only edge variables. As far as we know, the only polyhedral study on the edge variables formulation has been performed in [17], where several other new valid inequalities are proposed and their strength is discussed. Our paper can be viewed as an extensive follow-up of the studies in [14, 17]. We introduce new inequalities on the edge variables and compare them with existing ones, perform a polyhedral study of the new inequalities in the context of the edge variable formulation, develop and test branch-and-cut algorithms.

Motivations and contributions of the paper. For several routing problems, the fastest resolution methods are those developed from formulations using a minimal set of variables, as, for instance, in the case of the STSP, see, *e.g.,* [16]. Clearly, the performance of such methods strongly depends on the strength of the underlying formulations which, in many cases, involve sets of constraints having exponential size and thus require specialized methods to make them useful from a practical point of view.

Besides the degree constraints for each vertex, the formulations for the HpMP can be viewed as containing two sets of inequalities, one set preventing *more than* p cycles and the other preventing *less than* p cycles. Only a few works have focused on the study of formulations defined in the space of the binary edge variables and on methods devised for such formulations (see, *e.g.,* [14] and [17]). These works describe: *i*) partition-like inequalities that prevent more than p cycles, and that generalize the well known cut constraints from the STSP (enforcing *at most* 1 cycle) and *ii*) families of cycle-like inequalities to prevent less than p cycles. In Section 3, where these inequalities will be formally introduced, we will intuitively argue that the partition inequalities are more "efficient" than the cycle-like inequalities. The greater effectiveness of the constraints forbidding more than p cycles was also observed computationally by previous works on the HpMP, see *e.g.*, [2]. Motivated by this observation, our study will focus on inequalities preventing less than p cycles.

More precisely, we first introduce and discuss the new family of "quasi-Hamiltonian" cycle inequalities which generalizes the "Hamiltonian" cycle inequalities considered in [14, 17]. Next, we provide an additional family of inequalities, designed for the cases with large values of p (which correspond to several unsolved instances from the literature) and that can be viewed as "restricted" cut constraints used in integer-linear programming (ILP) models for the STSP. We will provide sufficient conditions for each of these families to be

facet-inducing for the HpMP polytope, whose extreme points correspond to the HpMP solutions. Finally, we introduce inequalities cutting off sub-optimal solutions, based on 2-opt exchanges. The new sets of inequalities are used to design an effective branch-and-cut algorithm based on the ILP formulation involving only edge variables.

Outline. In Section 2 we revise the literature on the HpMP. In Section 3 we provide an ILP formulation for the HpMP. We report related results from the literature and discuss the relative strength of the inequalities enforcing at least p cycles and of inequalities enforcing at most p cycles in a solution. In Sections 4 and 5 we introduce the new families of valid inequalities for the integer hull of that formulation. In Section 4 we introduce the quasi-Hamiltonian cycle inequalities and also provide sufficient conditions for a subset of the new inequalities to induce facets of the HpMP polytope. In Section 5 we discuss the special cases with $p = \lfloor n/3 \rfloor$. For these cases, we present new valid inequalities associated with cuts and multi-cuts whose shores have specific cardinalities, identify inequalities that are facet-inducing for the HpMP polytope and provide new formulations for the HpMP. In Section 6 we introduce inequalities cutting off sub-optimal solutions. Such inequalities are associated with specific 2-opt exchanges for the HpMP. In Section 7 we describe the separation routines used in our branch-and-cut algorithm to dynamically add all these strengthening inequalities. In Section 8 we describe a branch-and-cut algorithm based on the previous results, we assess its effectiveness with and without the new inequalities and we compare it also to state-of-the-art exact methods for the HpMP.

2 Literature Review

In order to contextualize our contributions, in this section we revise the existing studies on the HpMP. Since in our paper we will consider an exact approach for the HpMP, we especially focus on modelling and exact approaches for the HpMP already appeared in the literature. In particular, we do not detail existing heuristic or approximate algorithms for the HpMP.

The paper [4] introduces for the first time the HpMP in complete weighted digraphs. It presents two formulations for it: a set-partitioning formulation, in which every column is a least-cost cycle spanning a subset of vertices, and a compact ILP formulation based on variables assigning vertices to cycles and variables capturing the precedence relations of vertices within the same cycle. These models are not used to design exact algorithms, while 4 heuristic methods are computationally tested and compared. One of these heuristics is also adapted to the capacitated version of the HpMP formalized in the same paper.

The work of Glaab and Pott [13] initiates the theoretical study of the problem: after proposing an ILP formulation for the HpMP allowing 2-cycles and based on variables assigning each arc to precisely one of the p cycles of a solution, the authors determine the affine hull, the dimension of the convex hull of the integer solutions to such a formulation and show that its nonnegativity constraints induce facets of the underlying polytope.

The paper [25], proposes an alternative ILP formulation for the HpMP where each cycle in a solution is represented as a distinct route in a classical vehicle routing problem with a fictitious depot. The effectiveness of such a formulation has not been tested to date.

The study in [17] focuses on an edge variable formulation for the HpMP (not allowing 2-cycles) and highlights which of its inequalities cannot induce facets for the underlying integer hull. The authors also propose a new family of valid inequalities enforcing at least p cycles in each solution. Finally, they show that the blossom inequalities describing the 2-factor polytope [8] are facet-defining for the integer hull of their formulation.

After these early theoretical works, the research on the HpMP has more recently focused on resolution algorithms. Unless differently stated, the works revised below are for the variant forbidding 2-cycles. In [15] three ILP formulations for the HpMP are compared theoretically and computationally using branch-and-cut algorithms. All formulations involve edge variables and variables assigning vertices to cycles and are enhanced by symmetry-breaking inequalities. The best results in terms of polyhedral comparison and computational results are those based on the formulation enforcing at most p cycles through the so-called "partition" inequalities, recalled also in Section 3 of our paper.

In [14] new ILP formulations are considered and compared, namely: *i*) a formulation based only on edge variables; *ii*) two location-routing models involving variables associating one depot to each cycle on top of the edge variables or of the vertex-cycle assignment variables; *iii*) the set-partitioning formulation of [4] adapted to the variant in which 2-cycles are not allowed. The authors theoretically show that this latter formulation always provides a better lower bound than the location-routing models; however this set-partitioning formulation is not computationally tested, because, as stated in the paper, "it requires column generation". Instead, the computational comparison is performed with branch-and-cut algorithms and the best performance is obtained by using the edge variable formulation.

On top of two heuristic approaches, the paper [9] proposes a branch-and-cut algorithm to solve the HpMP. It is based on a formulation enhancing the one given in [14] and involving edge and vertex-cycle assignment variables. The heuristics provide primal solutions to the branch-and-bound algorithm, and the underlying formulation produces lower bounds of very good quality, thus outperforming the algorithms presented in [14].

A branch-and-price algorithm based on the set-partitioning formulation given in [14] is implemented and tested in [20]. The branch-and-price algorithm considers as a valid column of the model any simple cycle of the graph, relaxing the requirement of it cost minimality (originally present in the set-partitioning formulations of [4, 14]). The resulting algorithm solves instances with up to 318 vertices and outperforms the edge variables formulation of [14] on HpMP instances with values of p larger than $p^\star$, the number of cycles in the 2-factor relaxation of the problem. However it is outperformed by the edge variables formulation when $p < p^\star$.

A branch-and-cut algorithm is presented in [2]. This algorithm is based on a directed model and studies the two variants of the problem, allowing and not allowing 2-cycles. The main idea of the model presented in this work is that it splits the arc variables x_{ij} into three sets depending on whether node i is the depot of a cycle, node j is the depot of a cycle, or none is. This "split" model allows the use of a set of multi-cut inequalities used in an earlier paper by the same authors for a multi-depot routing problem. The branch-and-cut algorithm based on that model solves benchmark instances with up to 171 vertices when 2-cycles are allowed and up to 100 vertices when 2-cycles are forbidden. In the latter case, the algorithm proves the optimality of solutions to benchmark instances previously unsolved and well compares with those of [9, 20].

3 Starting Formulation and Related Known Results

In this section we review an ILP formulation for the HpMP introduced in [17]. Most of notation and terminology used in our paper is standard in combinatorial optimization, hence here we just introduce definitions that are more relevant for the description of the ILP formulation. Let $G = (V, E)$ be a simple undirected complete graph with edge weights $c\colon E \to \mathbb{R}_+$. The *cost* of $F \subseteq E$ is $c(F) := \sum_{f \in F} c_f$ and the same notation is extended to other vectors indexed by the edges of G. All cycles of G considered in this paper are *simple*, that is, they are the 2-regular connected subgraphs of G. A cycle of G is *Hamiltonian* if it spans all vertices of G. Given a partition $S_1, S_2, \ldots, S_m$ of V, we define the *multicut* $\delta(S_1, S_2, \ldots, S_m) =$

$\{\{i,j\} \in E\colon i \in S_k, j \in S_\ell \text{ with } k \neq \ell\}$. For the sake of conciseness we also define $\delta(i) := \delta(\{i\}, V \setminus \{i\})$.

Following [14] and [17], for every fixed nonnegative integer p we model the HpMP in which 2-cycles are not allowed as the following ILP, only involving *edge variables* $x_e \in \{0,1\}$ for every $e \in E$:

$$\min \sum_{e \in E} c_e x_e$$

$$x(\delta(v)) = 2 \quad \forall v \in V \tag{1}$$

$$x(\delta(S_1, S_2 \ldots, S_{p+u})) \geq u+1 \quad \forall \{S_1, \ldots, S_{p+u}\} \text{ partition of } V,\ u \in \left\{1, \ldots, \left\lfloor \frac{n-3p}{3} \right\rfloor\right\} \tag{2}$$

$$x(H) \leq n-p \quad \forall H \text{ Hamiltonian cycle of } G \tag{3}$$

$$x_e \in \{0,1\} \quad \forall e \in E. \tag{4}$$

The solutions to (1)–(4) are precisely the *incidence vectors* of the partitions of V into p cycles $C_1, C_2, \ldots, C_p$, that is, edge $e \in E$ belongs to C_i for some $1 \leq i \leq p$ if and only $x_e = 1$. Indeed, it is well-known that a point $\mathbf{x} \in \{0,1\}^E$ satisfies the *degree constraints* (1) if and only if it is the incidence vector of a partition in cycles of V. Moreover, the two families of inequalities (2) and (3) constrain the number of cycles in such a partition to be p. More precisely, the *partition inequalities* (2) generalize the well-known cut constraints for the STSP (corresponding to the case $p = u = 1$) and forbid cycle partitions with more than p cycles; instead, the *Hamiltonian cycle inequalities* (3) forbid those having less than p cycles by imposing that at least p edges must be removed from any Hamiltonian cycle.

From now on, a *solution* to the HpMP indicates both a point $\mathbf{x}$ satisfying (1)–(4) and its corresponding partition into p cycles. We denote by $\mathcal{X}_p^n$ the *integer hull* of (1)–(4), that is, the convex hull of the solutions to the HpMP.

The strength of inequalities of an ILP formulation is related to the dimension of the corresponding faces in the integer hull of the formulation. In particular, the *facets*, that is, the proper faces of highest dimension, are typically in correspondence with strong, and computationally effective, inequalities. According to [17], a result known from [11] states that the inequalities (2) for $u = 1$ are facet-defining for $\mathcal{X}_p^n$ when $p \geq 1$ and $n \geq 3p$. In contrast, [17] also states that, at least when $p = 2$, the Hamiltonian cycle inequalities (3) cannot induce facets of $\mathcal{X}_p^n$.

We were not able to formally generalize the latter result to every p, thus we conclude by providing an intuitive comparison of the relative strength of the partition inequalities (2) and of the Hamiltonian cycle inequalities (3): namely, we compare the "efficiency" of either inequalities in cutting off infeasible points associated with subgraphs having respectively less or more than p cycles. We start this argument by considering the case $p = 2$. In this case, the infeasible points satisfying the degree constraints (1) and having less than p cycles are the Hamiltonian cycles. Since two distinct Hamiltonian cycles differ in at least 2 edges, the inequality (3) associated with the Hamiltonian cycle H cuts off precisely one infeasible point when $p = 2$, namely H itself. Now, let us consider the infeasible points satisfying the degree constraints and having $p + 1 = 3$ cycles. These points are cut off by the partition inequalities (2) for $u = 1$. Such inequalities are associated to 3 subsets of vertices. The inequality associated to subsets S_1, S_2, S_3 cuts off multiple infeasible points, namely all those corresponding to partitions into 3 cycles C_1, C_2, C_3 such that $V(C_i) = V(S_i)$ for $i = 1, 2, 3$. For a complete graph, the number of these points is equal to $(|S_1|-1)!(|S_2|-1)!(|S_3|-1)!/8$. Observe that it is not restrictive to assume that $|S_i| \geq 3$ for every $i = 1, 2, 3$. Moreover, since $|S_1| + |S_2| + |S_3| = n$, we may additionally assume that $|S_1| \geq \lfloor n/3 \rfloor$. Using then the inequality $m! \geq (m/2)^{m/2}$ valid for every integer m we obtain that the number of solutions cut off by a single partition inequality with $u = 1$ grows

as $(((\lfloor n/3 \rfloor - 1)/2)^{(\lfloor n/3 \rfloor - 1)/2})/2$, which is exponential in n. When $p = 3$, a similar reasoning shows that an inequality (3) associated with the Hamiltonian cycle H cuts off precisely 1 infeasible point corresponding to H and at most $n(n-1)$ corresponding to partitions into at most 2 cycles (namely, those obtained from H by removing two edges and by connecting the residual paths into a single cycle other than H or by closing the residual paths into two cycles). That is, for $p = 3$, a single Hamiltonian cycle inequality cuts off a quadratic number of infeasible points, while a single partition inequality cuts off an exponential number of infeasible points. In general, for every fixed p, a Hamiltonian cycle inequality cuts off $\sim n^{p-1}$ infeasible points, while a partition inequality with $u = 1$ cuts off $\sim (((n/p - 1)/2)^{(n/p-1)/2})/2$ infeasible points, that is, a number which is exponential in n (since p is fixed).

Our explanation of the phenomenon above is that the left-hand side of the Hamiltonian cycle inequalities (3) forces the points violating them to very constrained structures: a point with a high value of the left-hand side in (3) corresponds to partitions in cycles where most of vertices are linked in the same way as they appear in H. This is opposed to the partition inequalities, where the relationship between vertices is only weakly structured.

A first approach to overcome this limitation of the Hamiltonian cycle inequalities is proposed in [14] and consists in generalizing the idea underlying the Hamiltonian cycle inequalities by forbidding subgraphs composed by less than p cycles (and not only by precisely 1 cycle). This leads to the *cycle cover* inequalities associated to any partition of V into disjoint cycles $C_1, C_2, \ldots, C_{p-k}$ with $k \geq 1$:

$$x(C_1) + x(C_2) + \cdots + x(C_{p-k}) \leq n - k - 1. \tag{5}$$

Inequalities (5) have been exploited in the branch-and-cut algorithm of [14] using a heuristic separation (in the same paper it is shown that the separation problem for (5) with $k = 1$ is **NP**-hard). In this paper, we follow a different approach, motivated by the observations above. Namely, in Section 4 we consider inequalities associated to single cycles not necessarily covering all vertices of G. This imposes a less restrictive structure on the infeasible points violating them, and thus, each inequality of the new family cuts off more infeasible points than an inequality in family (3). Moreover, we will be able to show that a single inequality in the new family implies several cycle cover inequalities (5).

4 Quasi-Hamiltonian Cycle Inequalities

Following the motivation given in Section 3, in this section we provide inequalities valid for $\mathcal{X}^n_{\geq p} = \text{conv}\{x \in \{0,1\}^E : x \text{ satisfies (1) and (3)}\}$, that is, the convex hull of incidence vectors of partitions of G into *at least* p cycles. Clearly, $\mathcal{X}^n_{\geq p}$ is a relaxation of $\mathcal{X}^n_p$ hence all inequalities valid for the former are also valid for the latter.

For simplicity, we explain the idea behind the new inequalities by means of examples arising from small values of p. First, let us consider $p = 2$ and let C be a cycle containing all vertices of G except vertices $n-1$ and n. Note that C cannot belong to a partition of V into at least 2 cycles, because it is not possible to get additional cycles using only vertices $n-1$ and n. Then a vertex of $\mathcal{X}^n_{\geq 2}$ corresponds to a HpMP solution with at most $|C| - 2$ edges in C (as otherwise it would contain all vertices of C in the same path). Then inequality $x(C) \leq |C| - 2$ is valid for $\mathcal{X}^n_{\geq 2}$ (hence for $\mathcal{X}^n_2$). Note that: *i*) the latter inequality has the same expression of (3) for $p = 2$, except that the cycle defining its left-hand side does not cover all vertices of G; *ii*) the same argument holds if just one vertex of G does not belong to C, while it does not hold if at least 3 vertices do not belong to C since such vertices are sufficient to produce a cycle. In other words, the cycle defining the new inequality is "quasi-Hamiltonian" since it covers "almost" all vertices of G. When $p = 3$,

the above discussion extends as follows. For a quasi-Hamiltonian cycle C not containing up to two vertices of G, the inequality is $x(C) \leq |C| - 3$, again identical in its form to (3) for $p = 3$. Interestingly, the reasoning extends to the cases of quasi-Hamiltonian cycles excluding more than two vertices. More precisely we obtain the valid inequality $x(C') \leq |C'| - 2$ for every quasi-Hamiltonian cycle C' not containing 3, 4 or 5 vertices. Indeed, up to 5 vertices outside C' are not enough to get a partition of V into at least 3 cycles (including C' itself), hence C' must be disconnected in every feasible solution. As in the case $p = 2$, cycles excluding more than 5 vertices do not yield valid inequalities of this type.

The new inequalities are based on the same reasoning applied to all values of p: 1) the cycles yielding the inequalities need not cover all vertices provided the remaining ones are not enough to complete a feasible solution; 2) the more vertices are excluded from the cycle, the more edges a feasible solution to the HpMP has in common with the cycle, thus yielding a higher right-hand side in the inequality. Formally, the *quasi-Hamiltonian cycle* (QHC) inequality corresponding to a cycle C of G is:

$$x(C) \leq |C| - p + \left\lfloor \frac{n - |C|}{3} \right\rfloor \tag{6}$$

We observe that (6) generalizes the Hamiltonian cycle inequalities (3). The next result formally proves that they are valid:

Proposition 1. *Let C be a cycle of G such that $p - \left\lfloor \frac{n-|C|}{3} \right\rfloor \geq 2$. Then inequality* (6) *associated with C is valid for $\mathcal{X}^n_{\geq p}$ for all $p \geq 2$.*

Proof. Let us take $\mathbf{x} \in \text{vert}(\mathcal{X}^n_{\geq p})$ and let us assume, by contradiction, that $\mathbf{x}(\overline{C}) \geq |\overline{C}| - p + \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor + 1$ for some cycle $\overline{C}$ of G such that $p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor \geq 2$. Let H be the subgraph of G induced by the edges $e \in \overline{C}$ such that $\mathbf{x}_e = 1$, so that $\mathbf{x}(\overline{C}) = |H|$. By our hypothesis, since $|H| \geq |\overline{C}| - p + \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor + 1$ we get that $|\overline{C} \setminus H| = |\overline{C}| - |H| \leq p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor - 1$. The previous inequality implies that H is obtained from $\overline{C}$ by removing at most $p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor - 1$ edges, therefore, since $\overline{C}$ is a cycle, H contains at most $p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor - 1$ connected components. Hence at most $p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor - 1$ cycles containing $V(\overline{C})$ can be obtained by using the edges $e \in E$ such that $\mathbf{x}_e = 1$. Let V' be the set of vertices not contained in any of those cycles. Since $|V'| \leq n - |\overline{C}|$, they can be covered by at most $\left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor$ cycles of G. Finally, the total number of cycles in the subgraph K having $\mathbf{x}$ as incidence vector is the sum of the number of cycles covering the vertices of $\overline{C}$ and the number of cycles covering vertices of V'. The cycles covering vertices in H are at most $p - \left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor - 1$; the cycles covering the vertices of V' are at most $\left\lfloor \frac{n-|\overline{C}|}{3} \right\rfloor$. However K must have at least p cycles and this is a contradiction. □

We now examine and compare the efficiency of QHC inequalities with the efficiency of the Hamiltonian cycle inequalities, respectively (6) and (3), in cutting off infeasible points. We also show that the QHC inequalities (which are associated to a single cycle) imply subsets of the cycle cover inequalities (5) (associated with multiple cycles).

We start this comparison by considering a cycle C such that $n - 1 \leq |C| \leq n - 2$ and without loss of generality we assume that vertex n (when $|C| = n-1$) or vertices $n-1$ and n (when $|C| = n-2$) do not belong to C. Then, for every $p \geq 2$, the rounded term in (6) is equal to 0 and the QHC inequality associated with C is $x(C) \leq |C| - p$. As in the case of Hamiltonian cycle inequalities, the right-hand side of such inequality is smaller when p is larger. This QHC inequality cuts off several vertices of the 2-factor polytope associated to G that are infeasible for the HpMP. Namely, when $|C| = n - 1$, it cuts off all incidence points of the $n - 1$

Hamiltonian cycles containing all edges of C but one and when $|C| = n-2$, it cuts off all incidence points of the $2(n-2)$ Hamiltonian cycles containing edge $(n-1, n)$ and all edges of C but one. Note that these Hamiltonian cycles are removed by as many Hamiltonian cycle inequalities (3). That is, in terms of cutting off infeasible points of the 2-factor relaxation, a single QHC inequality associated with cycles of length $n-1$ (resp. $n-2$) has the same efficiency as $n-1$ (resp. $2(n-2)$) Hamiltonian cycle inequalities.

We continue this analysis with cycles C of smaller length, namely $n-3 \leq |C| \leq n-5$. In this case, we first observe that the QHC inequalities (6) are not valid when $p = 2$ while they are valid for $p \geq 3$. Since $n-3 \leq |C| \leq n-5$, the rounded term of (6) is equal to 1, hence the QHC inequality corresponding to C is $x(C) \leq |C| - p + 1$ for every $p \geq 3$. A QHC inequality associated with a cycle of length $n-3$ (resp. $n-4$, $n-5$) removes at least $6(n-3)$ (resp. $24(n-4)$ and $120(n-5)$) Hamiltonian cycles obtained by linking all possible paths covering all vertices not in C to all paths obtained from C by removing one edge (note that other infeasible points are also cut off, for example by combining paths on vertices outside C with the pairs of paths obtained from C by removing two edges). This suggests that, relative to the Hamiltonian cycle inequalities, the efficiency of the QHC inequalities grows by decreasing the length of the quasi-Hamiltonian cycles defining the inequalities. An additional property is that when $n-3 \leq |C| \leq n-5$ and $p \geq 3$, the corresponding QHC inequalities imply the 2-cycle cover inequalities (5) associated with cycles C and C' where C' is any cycle covering precisely the vertices not in C. To show this, observe that $|C| + |C'| = n$ and also that the QHC inequality associated with $n-3 \leq |C| \leq n-5$ and $p \geq 3$ is $x(C) \leq |C| - p + 1$. Then summing the latter to the trivial inequality $x(C') \leq |C'|$ we obtain $x(C) + x(C') \leq n - p + 1$ which is precisely inequality (5) associated with the 2 cycles C and C'.

When $p \geq 4$ a similar reasoning does not lead to show that the 3-cycle cover inequalities are implied because at most 5 vertices outside C are not enough to form 2 disjoint cycles. Then, in order to derive such a result for $p \geq 4$ we need to reduce even further the length of the cycles defining the QHC inequality. For example, when $p = 4$ and $n-6 \leq |C| \leq n-8$ we can take any partition into two cycles C' and C'' of the vertices not contained in C and sum, as before, the QHC inequality $x(C) \leq |C| - p + 2$ with the trivial inequality $x(C') + x(C'') \leq |C'| + |C''|$ to obtain the 3-cover cycle inequality $x(C) + x(C') + x(C'') \leq n - p + 2$ associated with C, C' and C''. In fact generalizing the above argument we get the following result whose simple proof is omitted for the sake of brevity:

Proposition 2. *For every $p \geq 3$ and $n \geq 3p$ the QHC inequality* (6) *defined by a cycle C such that $1 \leq \left\lfloor \frac{n-|C|}{3} \right\rfloor \leq p-2$ implies all inequalities* (5) *defined by the partition of V into cycles $C, C_1, C_2, \ldots, C_h$, where $h = \left\lfloor \frac{n-|C|}{3} \right\rfloor$.*

Observe that that the cycle covering inequalities implied by the QHC inequalities, as stated in Proposition 2, are a strict subset of family (5). Moreover, for given C and $h = \left\lfloor \frac{n-|C|}{3} \right\rfloor$ as in Proposition 2 the same reasoning made above yields that the QHC inequality defined by C only implies a weaker form of the cycle cover inequality defined by partitions of G into less than h cycles. For example, if $p = 4$ and $n-6 \leq |C| \leq n-8$ summing the QHC inequality $x(C) \leq |C| - p + 2$ with $x(C') \leq |C'|$ (with C' covering all vertices not in C) yields $x(C) + x(C') \leq n - p + 2$ which is weaker than (5) associated with the partition of G into cycles C and C'.

From the previous discussion we can view the cycles defining the QHC inequalities as organized in ordered blocks, each containing cycles of three consecutive length values. The block on the top contains the cycles of length in the set $\{n, n-1, n-2\}$. Each subsequent block contains the cycles of three lengths not larger than those of the preceding block. In Table 1 we give an example of block view of the QHC inequalities for $n = 20$ and $2 \leq p \leq 5$. From Proposition 2 it follows that the QHC inequalities in lower blocks imply inequalities (5) associated with partitions in a larger number of cycles. While the efficiency analysis intuitively suggests that

Table 1: Blocks of QHC inequalities for $n = 20$: cycle length and right-hand side.

	$\|C\|$	RHS $p = 2$	RHS $p = 3$	RHS $p = 4$	RHS $p = 5$
Block 1	20	18	17	16	15
	19	17	16	15	14
	18	16	15	14	13
Block 2	17	-	15	14	13
	16	-	14	13	12
	15	-	13	12	11
Block 3	14	-	-	12	11
	13	-	-	11	10
	12	-	-	10	9
Block 4	11	-	-	-	9
	10	-	-	-	8
	9	-	-	-	7

the smallest cycle in each blocks yields strong QHC inequalities, we were able to formalize such a result only for the smallest cycles defining a valid QHC inequality, namely cycles C such that $|C| = n - 3p + 4$. Indeed we have:

Proposition 3. *Let $p \geq 3$ and $n \geq 3p+3$. If C is a cycle of odd length $|C| = n-3p+4$, then inequality* (6) *associated with C is facet-defining for $\mathcal{X}_p^n$.*

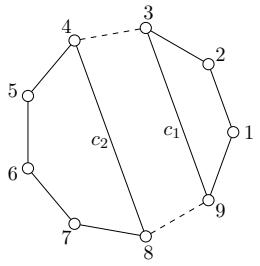
Proof (Sketch). The complete proof is given in B. Here we outline the main argument. When $|C| = n-3p+4$, inequality (6) associated with cycle C is $x(C) \leq |C|-2$. We recall that, since $n \geq 3p+3$, the results reported in [17] ensure that $\dim(\mathcal{X}_p^n) = n(n-1)/2-n$. Then, we construct an affine basis of dimension $n(n-1)/2-n$ for the face induced by $x(C) \leq |C| - 2$. More precisely we construct three sets of points starting from the three types of subgraphs associated with C, of which we give examples in Figure 1. The assumption on the parity of $|C|$ allows us to prove the affine independence of such points. We do this sequentially, by proving the affine independence of a new point from those already considered. This is accomplished by exhibiting a linear inequality satisfied by all points already considered but not by the new one. □

We point out that the result of Proposition 3 only provides sufficient conditions for a subfamily of QHC inequalities to be facet-defining for $\mathcal{X}_p^n$ and by no means it excludes that *other* QHC inequalities can have the same property, at least for some specific values of n and p. In particular, we expect the smallest cycles of each block to be associated with strong QHC inequalities. We leave it open the investigation of a complete characterization of the QHC inequalities that induce facets for $\mathcal{X}_p^n$. Another polyhedral result we can derive is that inequalities (6) strictly tighten the linear relaxation polytope of formulation (1)–(4). For conciseness, the formal derivation of such result is postponed to Proposition 10 of B. In brief, the proof exhibits fractional points belonging to the linear relaxation which violate QHC inequalities associated with cycles of length $n - 3p + 4$.

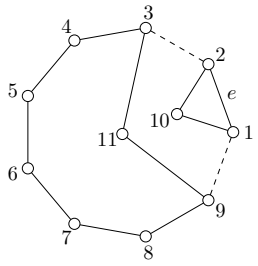
Finally, we point out that, as in the derivation of inequalities (5), the QHC inequalities can be generalized to "quasi-Hamiltonian cycle covers", *i.e.*, to inequalities whose support graph is a partition in cycles of subsets of vertices large enough. Their theoretical analysis and their effective embedding in branch-and-cut algorithms is currently under investigation, therefore we omit this generalization from our paper.

5 The HpMP for Large Values of p

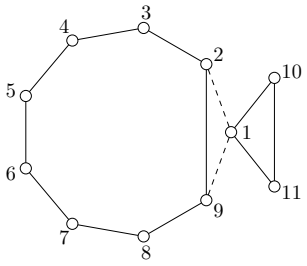
We now focus on the HpMP for large values of p, namely those such that $p = \lfloor n/3 \rfloor$. Several works on the HpMP have reported that these are challenging cases from a computational point of view, see, *e.g.*, [2, 9]. In



(a) $\mathcal{A}^{c_1,c2}$ with $c_1 = \{3, 9\}$ and $c_2 = \{4, 8\}$.



(b) $\mathcal{B}^e$ with $e = \{1, 2\}$.



(c) $\mathcal{C}^1_{10,11}$.

Figure 1: The three subgraphs constructed in the proof of Proposition 3 from a cycle C giving a smallest QHC inequality. Here we consider $n = 14$ and $p = 3$ so that $|C| = 9$. See C for the definition of $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$.

this section we provide new sets of inequalities valid for these cases. They are based on the specific structure of the feasible solutions to the HpMP with $p = \lfloor n/3 \rfloor$ and lead to substantially better computational results than the ones reported in the literature. Moreover, exploiting such inequalities and particular cases of (6) we provide alternative formulations for the corresponding instances of the HpMP.

5.1 Restricted Cut and Multi-Cut Constraints

Inequalities (7) presented in this subsection will be referred to as *restricted cut constraints (RCCs)* since they have the same expression of standard cut constraints for the STSP but restricted to subsets of V of specific cardinality. Their validity stems from the observation that when $n = 3p$, a feasible solution contains only cycles with 3 edges and, when $n = 3p + 1$, the solution consists of $(p - 1)$ cycles with 3 edges and one cycle with 4 edges. The RCCs (7) prevent less than p cycles, and, when $p \geq 3$, it is not difficult to give examples of integer solutions with less than p cycles that are violated by these constraints.

Proposition 4. *Let $p \geq 2$ and let $n = 3p$ or $n = 3p + 1$. Then the inequalities*

$$x(\delta(S)) \geq 2 \tag{7}$$

are valid for $\mathcal{X}^n_p$ for every $S \subseteq V$ such that $2 \leq |S| \leq n - 2$ and for every integer $0 \leq k \leq p$

$$|S| \neq \begin{cases} 3k & \text{if } n = 3p \\ 3k \text{ or } 3k + 1 & \text{if } n = 3p + 1 \end{cases}$$

Proof. If $n = 3p$, every solution is composed uniquely of cycles containing exactly 3 edges. Hence if $S \subseteq V$ is such that $|S|$ is not a multiple of 3, no feasible solution partitions the vertices of S into cycles. Then, by the degree constraints (1), at least two edges of the solution belong to the cut $\delta(S)$. If $n = 3p + 1$, a similar reasoning applies, if we consider that all solutions in this case contain exactly one cycle with 4 edges and the remaining cycles with 3. □

It is well-known that the cut constraints are facet-defining for the STSP polytope. We were able to adapt a proof of this latter result provided in [5, p. 300-301] to the RCCs (7) in the case $n = 3p + 1$, thus obtaining the following result. For the sake of brevity we just sketch the proof here, the complete proof being provided in D.

Proposition 5. *Inequalities* (7) *are facet-defining for* $\mathcal{X}_p^{3p+1}$ *if* $|S| \geq 5$ *and* $p \geq 3$.

Proof (Sketch). We consider a facet $\mathcal{F}'$ of $\mathcal{X}_p^{3p+1}$ containing $\mathcal{F}$. The result follows by transforming the inequality $ax \leq b$ defining $\mathcal{F}'$ into an expression equivalent to (7). Through the degree constraints (1) we get that $a_{s,i} = 0$ for a fixed $s \in S$ and every $i \in V \setminus S$. We extend these identities to $a_e = 0$ for every $e \in \delta(S)$ by exploiting the fact that the solutions depicted in Figure 2a belong to $\mathcal{F}$. Using the construction represented in Figure 2b we are able to provide additional solutions belonging to $\mathcal{F}$ whose structure implies that $a_e = c$ for every $e \in S$ and $a_f = \bar{c}$ for every $f \in E(V \setminus S)$. That is, all points on the facet $\mathcal{F}'$ satisfy the equality

(a) (b)

Figure 2: Constructions used in the proof of Proposition 5. See D for details.

$cx(S) + \bar{c}x(V \setminus S) = c(|S| - 1) + \bar{c}(|V \setminus S| - 1)$. From this equality we get that $\mathcal{F}' = \mathcal{F}$. □

A similar result for the case $n = 3p$ is still open. Indeed, in the proof of Proposition 5 the relation $n = 3p + 1$ is exploited to construct the solutions represented in Figures 2a–2b, while the same construction does not holds when $n = 3p$. We also observe that the assumption $|S| \geq 5$ in Proposition 5 is not restrictive: indeed, when $n = 3p + 1$, the RCCs associated to subsets $S = \{u, v, w, t\}$ of cardinality 4 are implied by the QHC inequalities associated with the three cycles $C_1 = (u, v, w, t, u)$, $C_2 = (u, w, v, t, u)$ and $C_3 = (u, v, t, w, u)$ because summing those inequalities member-wise yields $2x(S) \leq 6$ which is easily seen to be equivalent to $x(\delta(S)) \geq 2$ through the degree constraints (1).

We conclude by observing that when $n = 3p + 2$ we cannot re-adapt the RCCs defined before since each feasible solution admits an empty cut for every possible cardinality of the shores. As an example, let us consider a HpMP instance with $p = 4$ and $n = 14$. Then, $x(\delta(S)) > 0$ is not valid for any subset S such that $|S| \leq 11$ (for subsets with $|S| \geq 12$ the inequality $x(\delta(S)) \geq 2$ is implied by the degree constraints). Indeed, if $|S| \leq 5$ there exists a feasible solution containing a cycle whose vertices are those in S; if $6 \leq |S| \leq 8$ there exists a feasible solution with two cycles partitioning the vertices in S and two additional cycles partitioning the vertices in the complement of S; finally, if $9 \leq |S| \leq 11$, there exists a feasible solution composed of 3 cycles partitioning S and 1 cycle covering the vertices in its complement. Then an extension of the results in this section to the case $n = 3p + 2$ involves the use of more general multi-cut inequalities, as we explain next. Using the same example as before, we note that the inequality $x(\delta(S_1, S_2, S_3)) \geq 2$ with $S_1 = \{1, 2, 3, 4, 5\}$ and $S_2 = \{6, 7, 8, 9, 10\}$ is valid. Indeed, since $n = 14$ and $|S_1| + |S_2| = 10$, there is no feasible solution having precisely one cycle in S_1 and precisely one cycle in S_2 (the remaining 4 vertices are not enough to generate 2 additional cycles), hence at least two edges must cross one of these two shores. In general, we can state the following result (whose proof is postponed to E):

Proposition 6. *Let* $p \geq 2$ *and* $n = 3p + 2$. *Let also* S_1, S_2, S_3 *be three vertex subsets partitioning* G. *Then the following* restricted 3-cut *constraint (R3CC)*

$$x(\delta(S_1, S_2, S_3)) \geq 2 \tag{8}$$

is valid for $\mathcal{X}_p^{3p+2}$ *whenever* $|S_1| \equiv 2 \pmod 3$ *and* $|S_2| \equiv 2 \pmod 3$.

In the experiments performed in Section 8 there is no instance with $n = 3p + 2$ and thus, we have not developed separation algorithms for the inequalities (8) although the development of such an algorithm might be worth investigating.

5.2 Alternative Formulations for the Cases $p = \lfloor n/3 \rfloor$

A natural question is whether the RCCs (7) (when $n = 3p, 3p + 1$) or the R3CC (8) (when $n = 3p + 2$) together with the degree constraints (1) and the integrality constraints (4) suffice to give a valid ILP model for the HpMP with $p = \lfloor n/3 \rfloor$. Unfortunately the answer is negative. As an example when $n = 3p$, consider $n = 9$ and $p = 3$, and an infeasible partition of G into one 3-cycle C_3 and one 6-cycle C_6. Its incidence vector satisfies all constraints (7). Similarly, for $n = 3p + 1$, consider $p = 3$ and $n = 10$. Then, the incidence vector of an infeasible partition of G into a 4-cycle C_4 and a 6-cycle C_6, satisfies constraints (7). Finally, for $n = 3p + 2$, consider $p = 4$ and $n = 14$. Then the infeasible solution given by two 4-cycles and one 6-cycle satisfies all inequalities (8). Motivated by this discussion, and exploiting the same idea behind the validity of the RCCs (7) and their 3-cut version (8), we now describe alternate valid formulations for the HpMP when $p = \lfloor n/3 \rfloor$.

We consider first the case $n = 3p$. Then, as noted in Section 5.1, each feasible solution is composed uniquely of 3-cycles. For this case the smallest cycles yielding valid QHC inequalities have 4 edges, hence the corresponding inequalities (6) are $x(C) \leq 2$ for every 4-cycle C of G. Observe that any integer solution satisfying the degree inequalities (1) with less than p cycles would have at least one cycle, C', with more than 3 edges. But such a solution would violate 4-cycle inequalities (6) associated to any 4-cycle D containing 3 consecutive edges from C'. This argument leads to

Proposition 7. *A valid ILP formulation for the HpMP with* $n = 3p$ *is given by the degree constraints* (1)*, the QHC inequalities* (6) *for 4-cycles and the integrality constraints* (4)*.*

We consider, now, the case $n = 3p + 1$. As noted in Section 5.1, each feasible solution is composed of cycles with 3 edges and a single cycle with 4 edges. Moreover, the smallest cycles associated with QHC inequalities (6) have 5 edges, hence the corresponding inequalities are $x(C) \leq 3$ for every 5-cycle C of G. Consider an instance with $n = 16$ and $p = 5$ and the solution consisting of four disjoint cycles, each with 4 vertices. This integer solution clearly satisfies the degree constraints (1) and the QHC inequalities associated with the smallest cycles, but it is infeasible since it contains less than 5 cycles (and more than one cycle with 4 vertices). Thus, the analogue of Proposition 7 for the case of $n = 3p + 1$ is not valid. However, as the next proposition shows, we obtain a valid formulation if we add the RCCs (7) of the previous Section 5.1.

Proposition 8. *A valid ILP formulation for the HpMP with* $n = 3p+1$ *is given by the degree constraints* (1)*, the QHC inequalities* (6) *associated to 5-cycles, the RCCs* (7) *and the integrality constraints* (4)*.*

Proof. We only need to show that if $\mathbf{x}$ is integer and satisfies (1), (6) and (7) then it is the incidence vector of a solution to the HpMP when $n = 3p + 1$. From (1), (6) and the integrality constraints, $\mathbf{x}$ is the incidence vector of a partition of G into 3- and 4-cycles. Since $n = 3p+1$ the cycle partition corresponding to $\mathbf{x}$ cannot be composed only of 3-cycles and thus, it must contain at least one 4-cycle. Point $\mathbf{x}$ is the incidence vector of a solution to the problem with $n = 3p + 1$ if and only if there is exactly one 4-cycle in such a partition. Assume that there are two distinct 4-cycles C^1 and C^2. Then, by defining $S = V(C^1) \cup V(C^2)$, the RCC (7) $x(\delta(S)) \geq 2$ is valid for $\mathcal{X}_p^{3p+1}$ (because $|S| \neq 3k, 3k + 1$ for all $1 \leq k \leq p$) and would be violated by $\mathbf{x}$, leading to a contradiction. □

Finally, using the R3CCs (8), we get a result for $n = 3p + 2$ analogous to Proposition 8.

Proposition 9. *A valid ILP formulation for the HpMP with $n = 3p+2$ is given by the degree constraints* (1)*, the QHC inequalities* (6) *associated to 6-cycles, the R3CCs* (8) *and the integrality constraints* (4)*.*

Proof (Sketch). A solution to the formulation described in the proposition is a cycle partition of G (due to the degree constraints (1)). Such a partition does not contain cycles of length larger than 6 (due to the QHC inequalities (6) associated to 6-cycles). Next, we show that the partition must contain either *i*) precisely one 5-cycle and $(p-1)$ 3-cycles or *ii*) precisely two 4-cycles and $(p-2)$ 3-cycles. Indeed, in all other cases at least one R3CC (8) must be violated. The full proof of this latter step is provided in E. □

We conclude this section by remarking that the alternative formulations mentioned in the three propositions above do not include inequalities preventing more than p cycles since, when $p = \lfloor n/3 \rfloor$, such inequalities are not needed. In this sense, the cases with $p = \lfloor n/3 \rfloor$ can be viewed as symmetric of the case with $p = 1$ where inequalities preventing less than p cycles are not needed. The study of the $p = 1$ case (that is, the STSP) led to the development of the cut constraints, forbidding more than 1 cycle, which in turn were generalized to the partition inequalities (2), forbidding more than p cycles for $p \geq 1$. With this pattern in mind, we raise the research question of whether the study of the cases with large values of p might lead to alternative effective inequalities forbidding less than p cycles also for smaller values of p (*e.g.*, by generalizing the 3-cut constraints (8) to multi-cut inequalities valid for the HpMP with $p < \lfloor n/3 \rfloor$). This is currently under investigation.

6 Removing Sub-Optimal Solutions: the Restricted 2-Opt Inequalities

In this section we present a family of inequalities for the HpMP which are based on a specific type of 2-opt exchange and remove sub-optimal solutions. In the context of the STSP, the paper [19] explains how to obtain similar inequalities from general local search operators, including 2-opt exchanges.

To obtain similar inequalities for the HpMP we exploit the following remark whose validity is easy to prove.

Remark 1. *Let $S = \{C_1, C_2, \ldots, C_p\}$ be a feasible solution to a given instance of the HpMP whose objective function is denoted c. Let i, j, k, ℓ be four distinct vertices appearing consecutively in this order in cycle C_h for some $h \in \{1, 2, \ldots, p\}$. If $c_{ij} + c_{jk} + c_{k\ell} > c_{ik} + c_{jk} + c_{j\ell}$ then S is not optimal.*

Then, when solving an instance to optimality, we may safely exclude solutions as S in Remark 1. This is accomplished by enforcing the following *restricted 2-opt* inequalities:

$$x_{ij} + x_{jk} + x_{k\ell} \leq 2 \qquad \forall i, j, k, \ell \text{ as in Remark 1.} \tag{9}$$

Indeed, if (9) is violated by a feasible solution S, then S contains in one of its cycles a path (i, j, k, ℓ) and by Remark 1, the cycle containing it can be improved by a 2-opt exchange.

We note that the number of 4-tuples to check for deriving an inequality of type (9) is in the order of $O(n^4)$. Moreover, for each choice of such a 4-tuple there are 12 distinct orderings of its vertices potentially yielding a restricted 2-opt inequality. As a consequence, explicitly including all inequalities (9) in our model is not viable from a computational perspective. Therefore, in our computational experiments, inequalities (9) are added dynamically through a separation procedure described in Section 7. Since, in general, larger families

of inequalities typically overload the LPs solved during a branch-and-bound algorithm we exclude from our discussion inequalities associated to 2-opt exchanges applied to sequences of vertices longer than 4 and we only consider family (9) in our code.

7 Separation Algorithms

In order to assess the effectiveness of the inequalities proposed in previous sections, we will define several valid formulations for the HpMP by using combinations of the proposed inequalities. Since all such formulations involve a large number of constraints (at least cubic in the case of constraints of formulation of Proposition 7 and exponential in the other cases) we have implemented branch-and-cut algorithms. In the following we describe the separation algorithms for the inequalities used in our computational experiment. They always receive in input a (generally non-integer) point $x^\star \in [0,1]^{|E|}$ satisfying the degree constraints (1). Associated with $x^\star$ is its *support graph* $G_{x^\star} = (V, E_{x^\star})$ where $E_{x^\star} = \{e \in E\colon x^\star_e > 0\}$. Unless differently stated, paths in this section have no repeated vertices, that is, we assume a path to be obtained from a simple cycle by removing a single edge; the length of cycles and paths is defined as the number of their edges.

Separation of the partition inequalities (2). The generic separation problem for inequalities (2) is **NP**-hard, as shown in [14] through a reduction from the minimum k-cut problem. Hence for the separation of inequalities (2) we resort to the same algorithm devised in [14]. Given $x^\star \in [0,1]^{|E|}$, the algorithm constructs the support graph $G_{x^\star}$. Next it computes the connected components S_1, $S_2, \ldots, S_c$ of $G_{x^\star}$. If $c \geq p+1$ then the sets S_1, $S_2, \ldots, S_c$ define a violated inequality (2). The construction of graph $G_{x^\star}$ takes $O(|V|^2)$-time, while computing the connected components takes $O(|V|)$-time by depth-first search. We point out that, for generic fractional points $x^\star \in [0,1]^{|E|}$ the above algorithm is heuristic, although it is exact when $x^\star \in \{0,1\}^{|E|}$.

Separation of the QHC inequalities (6). We are not aware of the complexity of separating generic QHC inequalities (6), although we conjecture it is **NP**-hard. We describe a separation algorithm for the inequalities (6), that is, inequalities of the form $x(C) \leq |C| - p + \left\lfloor \frac{n-|C|}{3} \right\rfloor$ where C is a cycle of length at least $n-3p+4$. Since Proposition 3 indicates that QHC inequalities associated to smallest cycles are strong the separation algorithm takes this into consideration. We consider separately the cases $x^\star \in \{0,1\}^{|E|}$ and $x^\star \notin \{0,1\}^{|E|}$.

Separation for $x^\star \in \{0,1\}$. In this case, the integrality of $x^\star$ implies that it is the incidence vector of a cycle partition $C_1, C_2, \ldots, C_\ell$ of G. If $\ell \geq p$ the algorithm terminates since, by Proposition 1, the QHC inequalities are valid for $\mathcal{X}^n_{\geq p}$, hence none of them is violated. Otherwise, we have $\ell < p$, and we first check if each of these ℓ cycles alone leads to a violated QHC inequality. If one violated QHC inequality is found in this way the separation stops; otherwise we merge subsets of the ℓ cycles into a single cycle until a violated QHC inequality is found. The existence of such an inequality can be shown theoretically by exploiting $\ell < p$. From the last property it follows that this separation algorithm is exact. Furthermore, since the QHC inequalities together with the inequalities (1), (2) and (4) are sufficient to define a valid formulation for the HpMP, the correctness of the above procedure, together with the correctness of the separation of inequalities (2) on integer points, guarantees that our branch-and-cut algorithms separating (2) and (6) always yields an optimal solution to the HpMP. The details on cycle merging is provided in A.2.

Separation for $x^\star \notin \{0,1\}^{|E|}$. We start by determining the connected components $K_1, K_2, \ldots, K_\ell$ of the support graph $G_{x^\star}$. If $\ell \geq p$, the procedure stops, reporting no violated cut. Otherwise, we look for a minimal subset of components whose total vertex number is at least $n-3p+4$. Subsequently, a STSP is heuristically

solved on the subgraph of $G_{x^\star}$ induced by those vertices and we check whether the resulting cycle C defines a violated QHC inequality. Additionally, if $|C| > n - 3p + 4$, we search for other violated inequalities by modifying the cycle C according to 2 rules: 1) we sequentially remove vertices guaranteeing that the resulting cycles still define a valid QHC inequality which, if violated, is added to the pool of violated inequalities; 2) as soon as a violated QHC inequality cannot be produced by the process in step 1) we construct cycles of length $n - 3p + 4$ from C by considering all its sub-paths of consecutive $n - 3p + 4$ vertices and linking their endpoints. Finally, the above procedure is repeated for larger subsets of vertices obtained by merging additional connected components from the set $K_1, K_2, \ldots, K_\ell$. All details are provided in A.3.

Separation of RCCs (7). We separate the RCCs (7) for HpMP instances with $n = 3p, 3p+1$ as follows. When $x^\star \in \{0,1\}^{|E|}$ we use the following exact separation algorithm which runs in $O(n^2)$ time. First we retrieve the cycles $C_1, C_2, \ldots, C_\ell$ composing the graph $G_{x^\star}$. We collect all cycles C such that $|C| \equiv 1 \pmod 3$ in a set K_1 and those such that $|C| \equiv 2 \pmod 3$ in a set K_2. In the case $n = 3p$, if $|K_1| = |K_2| = 0$ no RCC is violated by $x^\star$. Otherwise all RCCs $\delta(V(C)) \geq 2$ for $C \in K_1 \cup K_2$ and all RCCs $\delta(V(C_1 \cup C_2)) \geq 2$ for $C_1, C_2 \in K_1$ are added to the pool of violated cuts. In the case $n = 3p + 1$, no RCC is violated by $x^\star$ if $|K_2| = 0$ and $|K_1| = 1$. Otherwise, all RCCs $\delta(V(C)) \geq 2$ for $C \in K_2$ and all RCCs $\delta(V(C_1 \cup C_2)) \geq 2$ for $C_1, C_2 \in K_1$ are added to the pool of violated cuts.

When $x^\star \notin \{0,1\}^{|E|}$, a min-weight cut $\delta(S)$ in the weighted support graph $(G_{x^\star}, x^\star)$ with S satisfying the cardinality condition of Proposition 4 can be found in polynomial-time, using techniques from submodular minimization under congruency constraints, see [22] and the discussion in [21, Sect. 3.1.2]. However, in our code we resort to the following simpler heuristic algorithm that works well in practice: we run the exact separation algorithm for the STSP cut constraints implemented in CONCORDE [1] and only add those satisfying the definition given in Proposition 4.

Separation of QHC inequalities (6) for the cases $n = 3p$ and $n = 3p + 1$. In this subsection we describe the approaches used in our branch-and-cut algorithm to separate the QHC inequalities (6) for 4-cycles and 5-cycles, respectively valid for the cases $n = 3p$ and $n = 3p + 1$. We first explain the separation algorithms for $n = 3p$. Given $x^\star \in \{0,1\}^E$, that is, when $x^\star$ is integer, we determine all cycles composing $G_{x^\star}$. Let $C = (v_0, v_1, \ldots, v_k, v_0)$ be one such cycle having length at least 4 and for every $i = 0, 1, \ldots, k$ consider the cycles $C_i = (v_i, v_{i+1}, v_{i+2}, v_{i+3}, v_i)$, with indices taken modulo $k + 1$: each cycle C_i yields a violated QHC inequality (6) associated to a 4-cycle. All these inequalities are added to the pool of violated cuts.

When $x^\star \notin \{0,1\}^E$, that is, when $x^\star$ is non-integer, we observe that a 4-cycle C_4 yielding a violated QHC inequality necessarily contains two consecutive edges e and f of $G_{x^\star}$ such that $x^\star_e + x^\star_f > 1$ (as otherwise, summing twice all edges of C_4 gives $2x^\star(C_4) \leq 4$ which is the QHC associated to C_4 and hence the latter cannot be violated). Therefore for every vertex $v \in V$ we first search for the two distinct vertices $u, w \in V \setminus \{v\}$ such that $x^\star_{\{u,v\}} + x^\star_{\{v,w\}}$ is maximized. If the latter value is greater than 1, vertex v potentially belongs to a 4-cycle yielding a violated QHC inequality. In this case, for every vertex $t \in V \setminus \{u, v, w\}$ we define the two 3-paths $P^1_{v,t} = (t, u, v, w)$ and $P^2_{v,t} = (u, v, w, t)$. If

$$x^\star(P^1_{v,t}) > 2 \text{ or } x^\star(P^2_{v,t}) > 2 \tag{10}$$

we add to the pool of violated cuts the QHC inequality $x(C_{v,t}) \leq 2$ where $C_{v,t} = (t, u, v, w, t)$.

The case $n = 3p + 1$ is treated by considering 5-cycles and 4-paths in place of the 4-cycles and 3-paths of the above procedure, and by setting to 3 the right-hand-side of condition (10). We point out that, while the

separation algorithm described above is exact for $x^\star \in \{0,1\}^E$, it is heuristic for $x^\star \notin \{0,1\}^E$. Indeed, when $n = 3p$ (resp. $n = 3p+1$) there could be violated QHC inequalities associated with 4-cycles (resp. 5-cycles) even if condition (10) is not satisfied (*e.g.*, taking $x^\star$ of value 0.6 identically on all edges of such cycles). Although this procedure is heuristic, the computational results presented later on indicate that it works well in practice. Furthermore, in preliminary tests, we have considered the inclusion of all violated 4- and 5-cycle inequalities. The obtained results indicate that adding QHC inequalities whose underlying cycles satisfy that condition leads to slightly better computational times, hence we maintain that approach.

Separation of the restricted 2-opt inequalities (9). Inequalities (9) can be separated in $O(n^4)$ time for both fractional and integer solutions by exhaustive enumeration of all vertices i, j, k, ℓ indexing (9). Preliminary experiments highlighted that such a procedure is too time consuming. Therefore, we use the following heuristic separation. Consider a threshold $\tau \in [0,1[$, solution $x^\star$ and all connected components $K_1, K_2, \ldots, K_m$ of the graph $G_{x^\star}(\tau)$ obtained from $G_{x^\star}$ by removing all edges e such that $x^\star_e < \tau$. Next, iteratively for $i = 1, 2, \ldots, m$, we run the enumerative algorithm only on the vertices of K_i. For each possible configuration of four vertices $i, j, k, \ell \in K_i$ we check whether $c_{ij} + c_{jk} + c_{kl} > c_{ik} + c_{jk} + c_{jl}$ and $x^\star_{ij} + x^\star_{jk} + x^\star_{kl} > 2$; if this is the case, then $x^\star$ violates inequality (9) corresponding to i, j, k, ℓ and is subsequently added to the model. In the computational results presented below we choose $\tau = 0.7$ and we separate (9) only on non-integer points. The chosen value for τ reduces the density of the graph where paths of 4 vertices are detected; moreover, it guarantees that every path of 4 vertices of $G_{x^\star}(0.7)$ verifying the cost condition of Remark 1 violates the corresponding inequality (9), because the right-hand side of this inequality is 2.

8 Experimental Study

In this section we report the computational results obtained from HpMP formulations defined by using several combinations of the inequalities proposed in the previous sections. Our main algorithm is a branch-and-cut algorithm employing all the valid inequalities described above and, from now on, denoted $\mathcal{F}$ (which stands for "full", since it exploits all ideas presented in the paper). Algorithm $\mathcal{F}$ starts with the 2-factor relaxation involving only the degree and integrality constraints (1) and (4) and dynamically adds the following inequalities, exploiting the separation routines described in Section 7:

- partition inequalities (2)
- QHC inequalities (6)
- restricted 2-opt inequalities (9)
- RCCs (7); these inequalities are used only when $n = 3p, 3p+1$
- QHC inequalities defined by 4-cycles (resp. 5-cycles); these inequalities are used when $n = 3p$ (resp. $n = 3p+1$).

The QHC inequalities associated with 4- and 5-cycles are separated by the specialized routine presented in the previous section. Moreover, the restricted 2-opt inequalities (9) are added *globally*, that is, once separated they are present in the LPs of all branch-and-cut nodes; all other families of cuts are added *locally*, that is, they are present only in sub-trees rooted at the branch-and-cut node where they have been separated. At each node of the enumeration tree a primal heuristic described below is also run.

We analyse the computational effectiveness of $\mathcal{F}$ in three steps: in Section 8.1 we compare $\mathcal{F}$ to state-of-the-art algorithms appeared in other studies on the HpMP; in Section 8.2 we compare $\mathcal{F}$ and the algorithms from the literature with the alternative formulations for the HpMP with $n = 3p, 3p + 1$ (see Proposition 7 and Proposition 8); finally, in Section 8.3 we study the impact of the valid inequalities on the performance of $\mathcal{F}$.

Before presenting the results, we describe the instances used in tests, the primal heuristic used in our algorithms and some additional implementation details.

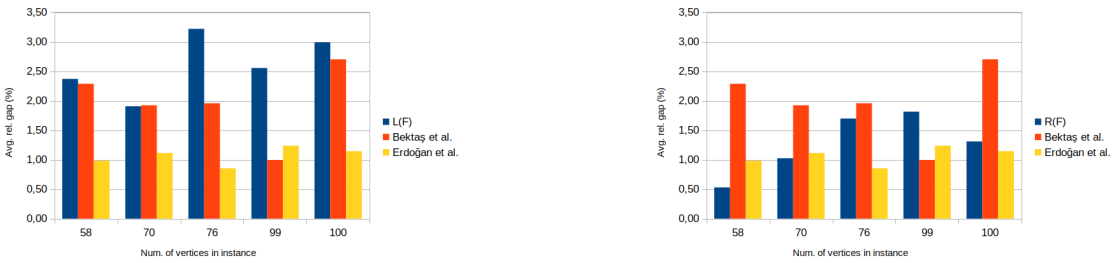
Instances. We consider a set of benchmark HpMP instances already used in [2] and [9] to test their exact algorithms. These instances were obtained from 24 TSPLIB instances [23, 24] with $21 \leq n \leq 159$. The paper [9] classifies them into "small" ones ($21 \leq n \leq 52$) and "large" ones ($n > 52$) and we follow such a classification. The number of vertices in an instance corresponds to the numerical value at the end of the instance name. From each one of these TSPLIB instances, 5 HpMP instances were created by considering $p \in \{\lfloor \frac{n}{10} \rfloor, \lfloor \frac{n}{7} \rfloor, \lfloor \frac{n}{5} \rfloor, \lfloor \frac{n}{4} \rfloor, \lfloor \frac{n}{3} \rfloor\}$. The only exceptions to this rule are the HpMP instances generated from `dantzig42` and `u159` of the TSPLIB: for these two instances, the papers [2, 9] use non-standard values of p; instead, for our tests we generate HpMP instances also from them in the standard way described above. We extend the benchmark by considering 7 additional TSPLIB instances with $124 \leq n \leq 400$, that is, instances `pr124`, `brg180`, `rat195`, `d198`, `pr226`, `gil262` and `rd400` and, for each of them, we have created 5 HpMP instances using the same values of p as above. Therefore, in the tests reported below we consider a total of 155 HpMP instances.

Primal heuristic. For our primal heuristic we follow an idea presented in [14]. Namely, given the edge-weighted graph corresponding to a fractional solution we first compute a minimum-weight spanning tree, next we remove the $p - 1$ edges of largest weight thus obtaining p connected components; finally a STSP heuristic is run on each component to determine p cycles. These latter are refined by executing the 2-opt exchanges on the sequences of 4 consecutive vertices, so to satisfy the condition given in Remark 1. The whole process terminates in polynomial time. A detailed description of the heuristic is provided in A.1.

Other implementation details. The parts of our branch-and-cut algorithms that do not directly include CONCORDE are implemented in `C++`. We compiled the code with `g++ 7.4.0` on a Ubuntu 18.04 machine, equipped with a Intel(R) Core(TM) i7-6700K CPU (4.00 gigahertz) and 32 gigabyte of RAM. The search trees generated by the branch-and-cut algorithms are managed via IBM CPLEX 12.6.3 [18]. We also use the standard setting of CPLEX for heuristic procedures and for the addition of general-purpose cuts. In all our tests we use the "strong branching" strategy of CPLEX (CPLEX parameter `VarSel` set to 3) and the "best bound" tradeoff of CPLEX for the MIP emphasis (CPLEX parameter `MIPEmphasis` set to 3). When not differently specified, all other CPLEX parameters are set to their default values. In particular, due to the presence of CPLEX callbacks for the implementation of cut separation, the branch-and-cut algorithms used in our experiments are run in sequential mode on a single thread.

8.1 Comparison with the Literature

We compare $\mathcal{F}$ with the two best branch-and-cut algorithms appeared in the literature, namely those of [2, 9]. We do not consider in our comparison the branch-and-price algorithm of [20] whose performance is comparable to the branch-and-cut of [2] (see the discussion therein). When comparing with the literature, we consider both the quality of lower bounds (both LP bound and root node relaxation values) obtained from $\mathcal{F}$ and the



(a) (b)

Figure 3: Average relative optimality gaps, depending on the number of vertices in the instances. Comparison of literature LP bounds with the LP bound of our formulation (Fig. 3a) and with root node bound of algorithm $\mathcal{F}$ (Fig. 3b).

performance of $\mathcal{F}$ in obtaining integer optimality. We point out that we did not run the literature algorithms, hence their performance is taken from the tables of results given in the respective papers. Therefore, we point out that the results from [2] and [9] were obtained on machines slower than the one used in our experiments, namely on machine equipped with an Intel Core i7-4790 3.6 gigahertz processor and 8 gigabyte of RAM in the former case, and on a Lenovo T440p laptop with an i7 2.50 gigahertz CPU and 8 gigabyte RAM in the latter case; moreover, the algorithm of [2] was run with a CPU time limit of 3 hours (while ours and the algorithm of [9] have a CPU time limit of 1 hour).

Quality of lower bounds. We consider two lower bounding approaches related to formulation $\mathcal{F}$:

- *LP bound* of $\mathcal{F}$, obtained from the linear relaxation of the full formulation used in algorithm $\mathcal{F}$;
- *root node bound* of $\mathcal{F}$, corresponding to algorithm $\mathcal{F}$ stopped at the root node (*i.e.*, CPLEX parameter `NodeLim` is set to 1).

We denote the LP bound of $\mathcal{F}$ by $\mathcal{L}(\mathcal{F})$ and its root node bound by $\mathcal{R}(\mathcal{F})$. The difference between the two is that in the second case the lower bound value benefits from CPLEX generic cuts and other features related with the integrality of variables. In both approaches the inequalities are separated using the routines described in Section 7. We point out that, in general, such separation routines are heuristic, hence the results only approximate the real value of the lower bound and root node bound associated with the formulation and the valid inequalities underlying $\mathcal{F}$.

We test both lower bounding approaches on the 55 "large" instances also considered in both Bektaş *et al.* [2] and Erdoğan *et al.* [9] and having $58 \leq n \leq 100$. We begin with the qualitative comparison of the LP bound of our formulation with the LP bounds of the models presented in [2, 9], provided in the bar diagram of Figure 3a.

The diagram is constructed as follows. On the x-axis we report the number n of vertices in the HpMP instances. For a given n, we depict 3 vertical bars, corresponding to the 3 algorithms considered in the comparison (blue column for $\mathcal{L}(\mathcal{F})$, red column for the LP bound of the formulation in [2] and yellow column for the one of the formulation in [9]). The height of the bar of each algorithm is the value of the average relative optimality gap (in percentage) of the corresponding lower bound. The average is computed over all instances having the same number of vertices. The gap on a single instance is computed as $100(UB^\star - LB)/UB^\star$, where $UB^\star$ is the best upper-bound known from the literature and our tests on exact resolution (see below). From this construction smaller bars correspond to better performance. Then it is clear that, for every instance size, the best performance corresponds to the formulation of [9]. The second best performance corresponds to the

LP bounds obtained by [2]. For almost every instance size, $\mathcal{L}(\mathcal{F})$ is weaker than the other approaches. In an instance-wise comparison (provided for completeness in F.1) the exceptions correspond to HpMP instances with $n = 3p, 3p + 1$.

Subsequently, we consider $\mathcal{R}(\mathcal{F})$. An instance-wise comparison with other lower bounding approaches is also reported in F.1. From it we first report that, as expected from the presence of CPLEX cuts, $\mathcal{R}(\mathcal{F})$ is never weaker than $\mathcal{L}(\mathcal{F})$; moreover, $\mathcal{R}(\mathcal{F})$ is comparable with the LP bounds of the literature formulations. Similarly to Figure 3a the results are represented by the bar diagram of Figure 3b where, now, the blue bar corresponds to $\mathcal{R}(\mathcal{F})$. We see that for $n = 58$ it outperforms both LP bounds from the literature; for the instances with $n = 76, 99$ $\mathcal{R}(\mathcal{F})$ and the LP bound of the formulation of [9] give the best results, while the one of [2] is almost always outperformed; finally, for instances with $n = 70, 100$ (which are the majority in the considered dataset) the best results are obtained by the formulation of [9] and $\mathcal{R}(\mathcal{F})$ while the LP bound of the formulation in [2] is outperformed. The latter results highlight that using CPLEX cuts is central to obtain strong lower bounds with our full formulation.

Exact resolution of small instances from the literature. We now analyse the performance of algorithm $\mathcal{F}$ when solving HpMP instances to integer optimality. In Table 2 we consider the 30 "small" benchmark instances also considered in both [2] and [9], and having $n \leq 52$. For each instance we highlight the value $p^\star$ indicating the number of cycles of its 2-factor relaxation. In the table we report the CPU time in seconds needed by each algorithm to reach the optimality or, if the time limit is reached, the relative optimality gap $100 \cdot \frac{UB-LB}{UB}$, where UB and LB are the best upper- and lower-bound obtained by the algorithm upon termination.

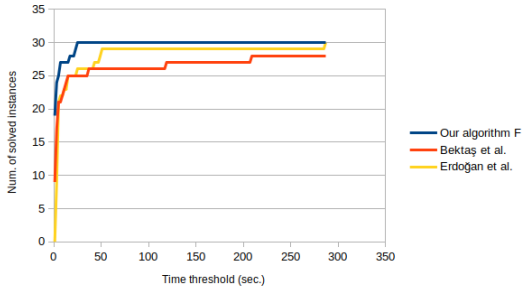
From Table 2 we see that all three algorithms are effective in solving all the HpMP instances considered in this experiment, with the unique exception of `att48` with $p = 16$ which is unsolved by the algorithm of [2]. Table 2 also shows that the performance of all three algorithms slightly degrades in HpMP instances with $p \geq p^\star$, with the algorithms of [2] and [9] being more stable when $p = \lfloor n/4 \rfloor$ and our algorithm $\mathcal{F}$ being more stable when $p = \lfloor n/3 \rfloor$. This latter outcome is expected given the use of the RCCs and of the 4- or 5-cycle QHC inequalities in our algorithm.

The results in Table 2 indicate that our algorithm $\mathcal{F}$ outperforms both algorithms from the literature. In particular, the algorithm of [2] turns out to be the slowest one, especially on HpMP instances having $p = \lfloor n/3 \rfloor$ in which case the CPU time needed to reach optimality is two or three orders of magnitude larger than that of $\mathcal{F}$. The algorithm of [9] is more effective than the one of [2] but it is never faster than ours, except in the HpMP instances `gr48` and `eil51` with $p = 12$.

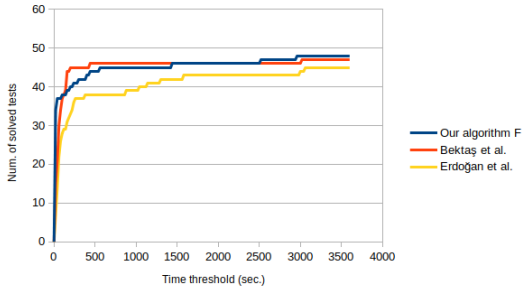
Computing the average on the subset of instances solved to optimality by *all* three considered algorithms, we see that $\mathcal{F}$ takes one third of the time needed by the algorithm of [9] to reach optimality and 1/25 of the time needed by the algorithm of [2]. We explain these very positive results by the effectiveness of our strenghtening cuts along with the quadratic size of the variable set used in our formulation (the formulations used in [2] and [9] are "extended" as they use additional variables).

In Figure 4a we give a qualitative overview of the relative effectiveness of the three considered algorithms through performance profiles: the x-axis reports the CPU time in seconds and the corresponding point of each profile on the y-axis is the number of solved instances within that CPU time; hence upper profiles correspond to better performance. Figure 4a shows that our algorithm solves all small benchmark instances to optimality in less than 30 seconds, while literature algorithms are slower.

For completeness, we report that the algorithm of [9] was also tested on 25 smaller benchmark instances with $21 \leq n \leq 29$, not considered in [2]. A comparison of $\mathcal{F}$ and the algorithm of [9] on the whole set of instances with $21 \leq n \leq 52$ is provided in F.2. Here we just summarize the results: both algorithms solve all



(a) Performance profile of algorithm $\mathcal{F}$ and those of [2, 9] (small instances).



(b) Performance profile of algorithm $\mathcal{F}$ and those of [2, 9] (large instances).

Figure 4: Comparison of $\mathcal{F}$ with the branch-and-cut algorithm algorithm from [2, 9] on benchmark instances.

instances to optimality in comparable CPU times, except when $p = \lfloor n/3 \rfloor$ in which case our algorithm $\mathcal{F}$ is faster. Due to this latter difference, in average our algorithm takes 1/10 of the time needed by the algorithm of [9] to reach optimality.

Exact resolution of large instances from the literature. We now perform a similar analysis on the 55 "large" instances of the benchmark set already considered in the literature, namely instances having $58 \leq n \leq 100$. The CPU times of the algorithm $\mathcal{F}$ and those of [2] and [9] are reported in Table 3. Such a table is constructed in a similar manner as Table 2. Since the considered formulations fail to reach optimality in several large HpMP instances, Table 3 additionally reports the following values in the last three lines: number of instances solved by each algorithm; average CPU time needed by each algorithm to reach optimality, computed on the subset of HpMP instances solved by *all* algorithms; average relative optimality gap of each algorithm computed using the same formula of Table 2 on the subset of HpMP instances unsolved by *all* algorithms.

A first observation is that, as for small benchmark instances, large HpMP instances become more difficult for all considered algorithms when $p^\star \geq p$ and especially if $p = \lfloor n/3 \rfloor$. On these latter cases, our algorithm in general performs better than the algorithm known from the literature, especially on instances with $n \geq 99$.

A second observation is that our algorithm $\mathcal{F}$ is the best one in average: the average CPU time needed by $\mathcal{F}$ to reach optimality is roughly 1/3 of the average CPU time needed by the algorithm of [2] and 1/5 of that of the algorithm in [9]. In terms of optimality gap we see that, in average, our algorithm and the one of [9] yield the best (and comparable) performances.

An instance-wise analysis also shows the effectiveness of our algorithm: in Table 3 the boldface value of each line corresponds to the algorithm yielding a performance considerably better than other algorithms and we see that $\mathcal{F}$ is often the best one throughout the whole set of HpMP instances. It solves to optimality 48 out of 55 HpMP instances, that is, 3 more than the algorithm of [9] and the same amount of the algorithm of [2]; however, we point out that instance `st70` with $p = 23$ was solved by the algorithm of [2] in more than 1 hour. Moreover, instances `kroA100` and `kroC100` with $p = 33$ solved by our algorithm were previously unsolved in the literature. Resuming, on large instances, algorithm $\mathcal{F}$ is very effective especially when $p \in \{\lfloor n/10 \rfloor, \lfloor n/7 \rfloor, \lfloor n/3 \rfloor\}$: when $p \in \{\lfloor n/10 \rfloor, \lfloor n/7 \rfloor\}$ the partition inequalities forbidding more than p cycles are more relevant and we have an effective separation routine for such inequalities thus obtaining a good performance overall; when $p = \lfloor n/3 \rfloor$, our restricted cut constraints turn out to be very useful; moreover, our algorithm $\mathcal{F}$ benefits from a specialized separation routine of the quasi-Hamiltonian cycle inequalities.

We finally provide a qualitative overview of the results of Table 3 through the performance profiles of Figure 4b, constructed as Figure 4a. From Figure 4b we see that our algorithm $\mathcal{F}$ solves to optimality the largest amount of HpMP instances within very short time thresholds (less than 100 seconds); increasing the CPU time threshold the algorithm of [2] and ours become comparable and more effective than the algorithm of [9].

Exact resolution of new benchmark instances. Finally, we report on the performance of our algorithm $\mathcal{F}$ on the set of the new benchmark instances. We also include the results obtained on the HpMP instances based on `u159` of the TSPLIB since, in constrast to [2, 9], we test them with the standard values of $p \in \{15, 22, 31, 39, 53\}$. Results are presented in Table 4 and they include the final upper- and lower-bound upon termination (columns "Final UB" and "Final LB" respectively) and the computational performance of the algorithm in terms of CPU time needed to reach optimality or of relative optimality gap if the instance is not solved within the time limit (last column of the table). To improve the readability of the table, boldface values in the last column indicate the HpMP instances solved to optimality. The results show that the most difficult HpMP instances are those based on the TSPLIB instance `brg180` as none of them is solved to optimality and all of them end with a relative optimality gap greater than the 80%. At the same time, algorithm $\mathcal{F}$ solves to optimality at least one HpMP instance for all instances with $n \geq 195$. Overall, algorithm $\mathcal{F}$ solves 17 out of the 40 considered HpMP instances with an average CPU time of resolution of ~ 2222 seconds and an average relative optimality gap of $\sim 21\%$.

8.2 Tests with Large Values of p

In this subsection we evaluate the impact of the RCCs and of the QHC inequalities associated with 4- and 5-cycles on instances with $n = 3p, 3p + 1$.

We start by briefly discussing root node bound results on 11 HpMP instances with $58 \leq n \leq 100$. We consider the root node bounds obtained from (i) the "full" formulation $\mathcal{F}$, (ii) the version of $\mathcal{F}$ obtained by not performing the separation of the QHC inequalities (6) associated to 4- or 5-cycles, (iii) the version of $\mathcal{F}$ obtained by not performing the separation of the RCCs (7) and (iv) the starting formulation (1)–(4). Our results indicate that the latter version always gives the worst lower bounds. In 6 of the considered instances the version separating only the RCCs yields the best root node bound. We explain this by the heuristic nature of our separation routines. In the remaining 5 tested instances, the full formulation $\mathcal{F}$ yields the best root node bound values, thus benefiting from the combination of RCCs and QHC inequalities associated with 4- and 5-cycles. Overall we conclude that the RCCs have the strongest impact on the root node bound, but their combination with the QHC inequalities associated with 4- and 5-cycles may lead to improvements. The tables reporting the detailed results of this experiment are provided in F.3.

Tests on instances with $n = 3p$. Subsequently, we move to the analysis of algorithms solving the HpMP to integer optimality. We begin with the 13 HpMP instances in which $n = 3p$, 8 of which were considered in [9] while the remaining 5 have been introduced in our paper (namely, `dantzig42` and `u159` with $p = n/3$, and the instances based on `brg180`, `rat195`, `d198`). We compare the performance of branch-and-cut algorithms based on HpMP formulations obtained by using several combinations of the inequalities presented in our paper. Namely, we consider:

- the basic "alternative" formulation $\mathcal{A}$ of Proposition 7, given by (1), (4), the restricted 2-opt inequalities (9) and (6) for 4-cycles (separated with the specialized routine);
- the formulation $\mathcal{A}^+$, obtained from $\mathcal{A}$ by additionally separating the RCCs (7);

Table 2: Comparison of our branch-and-cut algorithm $\mathcal{F}$ with those of [2, 9] (small instances).

Instance	$p^\star$	p	$\mathcal{F}$	Bektaş *et al.* [2]	Erdoğan *et al.* [9]
swiss42	7	4	**0.02**	1.00	1.37
		6	**0.06**	1.00	1.70
		8	**0.04**	1.00	1.56
		10	**0.07**	1.00	2.02
		14	**0.19**	1.00	1.12
att48	5	4	**0.01**	1.00	3.73
		6	**0.03**	1.00	3.41
		9	**0.33**	2.00	3.99
		12	3.36	4.00	3.99
		16	**1.28**	5.83%	285.90
gr48	6	4	**0.08**	3.00	2.82
		6	**0.02**	2.00	1.76
		9	**6.30**	13.00	13.70
		12	22.95	8.00	**4.91**
		16	**1.28**	208.00	24.25
hk48	6	4	**0.10**	2.00	3.48
		6	0.00	0.00	2.88
		9	**0.32**	2.00	3.05
		12	2.25	4.00	3.41
		16	**0.19**	118.00	10.04
eil51	3	5	**0.08**	4.00	4.58
		7	**0.72**	5.00	6.88
		10	15.70	**10.00**	41.32
		12	24.05	**14.00**	**14.41**
		17	**1.48**	1701.00	50.96
berlin52	7	5	**0.19**	2.00	3.66
		7	**0.00**	1.00	2.57
		10	**0.18**	2.00	4.43
		13	**1.93**	2.00	4.68
		17	**5.22**	36.00	48.81
Average CPU			3.01	74.14	9.50

Table 3: Comparison of our branch-and-cut algorithm $\mathcal{F}$ with those of [2, 9] (large instances).

Instance	$p^\star$	p	$\mathcal{F}$	Bektaş *et al.* [2]	Erdoğan *et al.* [9]
brazil58	12	5	**0.73**	12.00	78.90
		8	**0.14**	6.00	36.95
		11	**0.01**	5.00	5.14
		14	**0.99**	2.00	4.72
		19	**0.76**	71.00	31.13
st70	12	7	**0.41**	5.00	18.11
		10	**0.14**	4.00	12.56
		14	**0.08**	3.00	8.66
		17	**0.71**	9.00	11.16
		23	**192.41**	3739.00	1137.77
eil76	4	7	**0.3**	10.00	20.97
		10	**2.14**	38.00	18.60
		15	89.3	**58.00**	207.04
		19	1.87%	**133.00**	371.35
		25	3.12%	3.66%	**1025.73**
pr76	8	7	**0.14**	5.00	25.29
		10	8.85	8.00	224.40
		15	224.94	**34.00**	0.70%
		19	287.79	**7.00**	45.62
		25	38.56	**12.00**	867.49
rat99	8	9	0.72	12.00	90.16
		14	387.77	**23.00**	2.55%
		19	1.90%	**43.00**	2.19%
		24	7.24%	**44.00**	1.17%
		33	**16.11**	3003.00	2.85%
kroA100	13	10	**3.29**	151.00	2993.41
		14	**0.74**	95.00	40.47
		20	**5.46**	30.00	57.24
		25	435.69	**51.00**	77.87
		33	**2504.77**	8.87%	2.43%
kroB100	20	10	**2.84**	145.00	1575.86
		14	**1.67**	143.00	1292.72
		20	**0.39**	75.00	114.70
		25	**3.12**	16.00	34.89
		33	4.46%	11.07%	2.23%
kroC100	13	10	**2.35**	98.00	93.61
		14	**6.58**	67.00	77.78
		20	**23.12**	55.00	229.62
		25	543.1	436.00	**197.60**
		33	**2942.86**	9.35%	4.03%
kroD100	14	10	**0.14**	48.00	50.50
		14	**0.08**	42.00	46.87
		20	**14.88**	197.00	254.33
		25	155.73	160.00	154.50
		33	1.62%	3.21%	1.02%
kroE100	12	10	**0.09**	25.00	28.92
		14	**0.42**	37.00	28.45
		20	**4.11**	65.00	51.43
		25	**38.75**	92.00	62.60
		33	**1429.21**	1.64%	3054.13
rd100	14	10	**2.66**	147.00	177.19
		14	**0.15**	57.00	42.96
		20	**13.76**	101.00	149.61
		25	**8.43**	42.00	51.30
		33	0.29%	3.56%	1.66%
Tests solved			48/55	48/55	45/55
Average CPU			45.04	151.93	255.45
Average Opt. Gap			2.12%	5.94%	1.64%

Table 4: Performance of our branch-and-cut algorithm $\mathcal{F}$ on the set of new benchmark instances.

Instance	$p^\star$	p	Final UB	Final LB	CPU Time/Opt. Gap
pr124	23	12	52479.47	52479.47	**32.64**
		17	52210.02	52210.02	**255.94**
		24	51487.56	51487.56	**2.22**
		31	51830.28	51830.28	**5.26**
		41	57672.38	57672.38	**221.92**
u159	20	15	41238.46	41238.46	**2.77**
		22	41208.78	41208.78	**16.54**
		31	41849.60	41789.29	0.14%
		39	43377.73	42189.09	2.74%
		53	47320.58	47320.58	**129.42**
brg180	15	18	10910.00	1852.79	83.02%
		25	99410.00	1923.10	98.07%
		36	190040.00	2049.62	98.92%
		45	312280.00	2090.00	99.33%
		60	88790.00	2739.34	96.91%
rat195	7	19	2326.51	2326.51	**2978.33**
		27	2493.27	2337.33	6.25%
		39	3034.79	2360.93	22.20%
		48	3524.14	2388.50	32.22%
		65	2678.97	2678.97	**1595.81**
d198	32	19	12100.43	11997.56	0.85%
		28	11910.72	11910.72	**1.45**
		39	11927.47	11927.47	**103.62**
		49	12332.60	12005.21	2.65%
		66	17046.55	16615.30	2.53%
pr226	43	22	60693.96	58030.75	4.39%
		32	58033.26	57433.26	1.03%
		45	57177.55	57177.55	**0.95**
		56	108704.75	57177.55	47.40%
		75	240399.60	72872.37	69.69%
gil262	30	26	2260.32	2260.32	**26.86**
		37	2263.31	2263.31	**167.72**
		52	2283.96	2277.62	0.28%
		65	2996.28	2284.07	23.77%
		87	3597.48	2426.12	32.56%
rd400	51	40	14675.53	14675.53	**39.96**
		57	14684.15	14684.15	**496.84**
		80	17587.96	14717.44	16.32%
		100	24026.88	14765.27	38.55%
		133	31933.63	15576.99	51.22%
Tests solved					17/40
Average CPU					2221.96
Average Opt. Gap					20.78%

Table 5: CPU times and optimality gaps of various HpMP formulations on instances with $n = 3p$.

Instance	$\mathcal{A}$	$\mathcal{A}^+$	$\mathcal{Q}$	$\mathcal{F}$
gr21	0.00	0.00	0.01	0.00
gr24	0.01	0.01	0.01	0.49
dantzig42	0.75	0.53	0.55	0.70
swiss42	0.02	0.02	0.03	0.06
att48	6.52	**1.06**	5.93	1.27
gr48	0.84	0.84	0.62	0.83
hk48	0.12	0.18	0.19	0.19
eil51	1.23	1.73	1.53	1.52
rat99	25.75	13.43	14.48	16.05
u159	489.10	**49.07**	443.08	129.24
brg180	97.77%	97.68%	97.90%	**96.91%**
rat195	**1314.25**	2633.75	1972.04	1591.79
d198	39.46%	**0.71%**	9.52%	2.53%
Instances solved	11/13	11/13	11/13	11/13
Average CPU	167.14	245.51	221.68	**158.38**
Average Opt. Gap	68.62%	**49.20%**	61.85%	49.72%

Table 6: Comparison of our algorithms based on alternative formulations with those from literature for instances with $n = 3p$.

Instance	$\mathcal{A}$	$\mathcal{A}^+$	Bektaş *et al.* [2]	Erdoğan *et al.* [9]
swiss42	0.02	0.02	1.00	1.12
att48	6.52	**1.06**	5.83%	285.9
gr48	**0.84**	**0.84**	208.00	24.25
hk48	**0.12**	**0.18**	118.00	10.04
eil51	**1.23**	**1.73**	1701.00	50.96
rat99	25.75	**13.43**	3003.00	2.85%
Tests solved	6/6	6/6	5/6	5/6
Average CPU	0.55	0.69	507.00	21.59

- the formulation $\mathcal{Q}$ based on the generic QHC inequalities, given by (1), (2), (4), the restricted 2-opt inequalities (9) and (6) for all valid cycles;
- the "full" formulation $\mathcal{F}$.

In Table 5 we report the optimality gaps and the CPU times of these 4 formulations on the considered instances. Each column of the table corresponds to a version of the algorithm based on one of the 4 formulations above (indicated in the header of the table). The reported values are CPU times in seconds, except when the instance is not solved to optimality, in which case we report the relative optimality gap in percentage (computed as in Table 2 and Table 3 of Section 8.1). We also indicate the total number of solved instances, the average CPU times (computed on the subset of instances solved by *all* algorithms) and the average optimality gap (computed on the subset of instances unsolved by *all* algorithms). Full information, including the best upper- and lower-bounds obtained with each formulation is provided in Table 14 of F.3.

All four algorithms solve to optimality the same 11 instances. As indicated in Table 5 the best average CPU time is obtained by using formulations $\mathcal{A}$ and $\mathcal{F}$, while the best average optimality gap is obtained by using formulation $\mathcal{A}^+$. However all performances are similar, probably due to the fact that most of benchmark instances with $n = 3p$ are of small size and hence easily solved by all four versions of our algorithm. In fact, the few performance differences arise only on the new larger instances: in this case the formulations $\mathcal{A}$ and $\mathcal{A}^+$ are the best overall, as they are much faster than other versions in solving respectively instances `rat195` and `u159`. Moreover, the algorithm based on formulation $\mathcal{A}^+$ reports an optimality gap close to 0 on the largest instance `d198` considered in these experiment.

We then compare the algorithms based on formulations $\mathcal{A}$ and $\mathcal{A}^+$ with those of [2] and [9]. We restrict the comparison to the subset of 6 benchmark instances also considered in both these works. The comparison is presented in Table 6. Also in this case the average CPU time reported in the last line of the table is computed on the subset of instances solved by *all* algorithms considered in the table. The effectiveness of our algorithms based on formulations $\mathcal{A}$ and $\mathcal{A}^+$ is clear from the results reported in that table: while our algorithms solve to optimality all the 6 instances, both literature algorithms fail on instance `rat99`; moreover, we reduce the average CPU time needed to reach optimality by three orders of magnitude with respect to the algorithm of [2] and by two orders of magnitude with respect to the algorithm of [9]. We also point out that similar conclusions would be obtained by comparing the algorithms based on formulations $\mathcal{Q}$ and $\mathcal{F}$ with the literature algorithms.

Table 7: CPU times and optimality gaps of various HpMP formulations on instances with $n = 3p + 1$.

Instance	$\mathcal{Q}$	$\mathcal{A}^+$	$\mathcal{F}$
brazil58	5.80	**0.94**	**0.73**
st70	832.68	247.32	**190.58**
eil76	4.92%	**2.61%**	3.11%
pr76	122.80	**17.62**	38.44
kroA100	1.82%	**2112.57**	2457.73
kroB100	5.20%	**0.88%**	4.46%
kroC100	8.06%	3.50%	**2906.00**
kroD100	3.00%	**0.90%**	1.62%
kroE100	0.40%	**880.03**	1431.66
rd100	1.83%	**2164.67**	0.29%
pr124	1.12%	**66.79**	220.31
pr226	68.85%	**52.63%**	69.69%
gil262	46.30%	**25.51%**	32.56%
rd400	**51.20%**	54.57%	51.22%
Instances solved	3/14	7/14	7/14
Average CPU ($\mathcal{A}^+, \mathcal{F}$)		**554.21**	723.24
Average Opt. Gap ($\mathcal{A}^+, \mathcal{F}$)		**27.24%**	32.21%

Table 8: Comparison of our algorithms based on alternative formulations with those from literature for instances with $n = 3p + 1$.

Instance	Bektaş *et al.* [2]	Erdoğan *et al.* [9]	$\mathcal{A}^+$
brazil58	71.00	31.13	**0.94**
st70	3739.00	1137.77	**247.32**
eil76	3.66%	**1025.73**	2.61%
pr76	**12.00**	867.49	17.62
kroA100	8.87%	2.43%	**2112.57**
kroB100	11.07%	2.23%	**0.88%**
kroC100	9.35%	4.03%	**3.50%**
kroD100	3.21%	1.02%	**0.90%**
kroE100	1.64%	3054.13	**880.03**
rd100	3.56%	1.66%	**2164.67**
Tests solved	3/10	5/10	6/10

Tests on instances with $n = 3p+1$. We now report on a similar study for the 14 benchmark instances satisfying $n = 3p + 1$, 10 of which were tested also in the literature, while 4 are new (namely, pr124, pr226, gil262, rd400). In this case, formulations $\mathcal{A}^+$ and $\mathcal{F}$ are modified by separating the corresponding version of the RCCs (7) and the QHC inequalities (6) associated with 5-cycles; also, formulation $\mathcal{A}$ updated with the QHC inequalities associated to 5-cycles is not considered in this part because it is not valid for the HpMP with $n = 3p + 1$, see Proposition 8. Hence in Table 7 we compare the three algorithms based on the formulations $\mathcal{A}^+$, $\mathcal{Q}$ and $\mathcal{F}$.

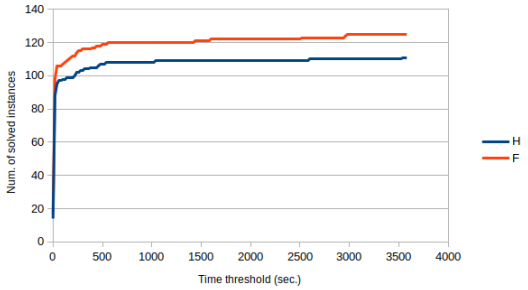
Results are computed as in Table 5, with the exception of the average CPU times and optimality gaps: in this case the averages are computed by considering the subset of instances respectively solved and unsolved by both algorithms based on $\mathcal{A}^+$ and $\mathcal{F}$.

Indeed, from Table 7 we see that the use of formulation $\mathcal{Q}$ yields the worst performance, with only 3 out of the 14 tested instances solved to optimality, hence we exclude such a formulation from average comparison; instead, using formulation $\mathcal{A}^+$ and $\mathcal{F}$ allows the resolution of 7 instances to optimality, although these sets do not coincide (the algorithm based on $\mathcal{A}^+$ fails on kroC100, while the one based on $\mathcal{F}$ fails on rd100). Moreover, the algorithm using $\mathcal{A}^+$ yields the best performance overall, being in average ~ 200 seconds faster than the algorithm using $\mathcal{F}$ and reporting a better average optimality gap. In Table 15 of F.3 we provide also the best upper- and lower-bounds reported by the three algorithms above upon termination.

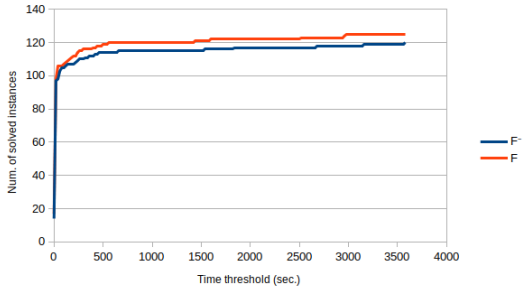
We next compare our best algorithm based on formulation $\mathcal{A}^+$ with those proposed in [2] and [9]. The comparison is reported in Table 8. The results indicate that our algorithm yields the best performance by solving more instances than the algorithms of [2] and [9] in shorter time (the only exception is eil76, solved by the algorithm of [9] and unsolved by ours). On the instances unsolved by at least two of the algorithms considered in Table 8 ours always reports the smallest optimality gap.

8.3 Impact of Valid Inequalities

In the previous Section 8.2 we have analyzed the impact of the RCCs (7) and of the QHC inequalities associated to the 4- and 5-cycles in the cases with $n = 3p, 3p + 1$. In this section we discuss the impact of the QHC inequalities (6) and of the restricted 2-opt inequalities (9) on resolution of the HpMP instances for generic values of p. More precisely we consider the following three algorithms for the HpMP:



(a) Performance profiles of algorithms $\mathcal{H}$ and $\mathcal{F}$: number of solved instances of the whole benchmark within fixed time thresholds.



(b) Performance profiles of algorithms $\mathcal{F}^-$ and $\mathcal{F}$: number of solved instances of the whole benchmark within fixed time thresholds.

Figure 5: Comparison of several versions of $\mathcal{F}$ with and without valid inequalities.

- the "full" algorithm $\mathcal{F}$;
- algorithm $\mathcal{F}^-$ obtained from $\mathcal{F}$ by excluding the restricted 2-opt inequalities (9);
- algorithm $\mathcal{H}$ obtained from $\mathcal{F}$ by imposing that only QHC inequalities associated to Hamiltonian cycles are added during the branch-and-cut process. Such inequalities are separated by modifying the routine for the general QHC inequalities as follows: if the current solution $x^\star$ is integer, we merge *all* cycles composing the graph $G_{x^\star}$ associated to $x^\star$; if $x^\star$ is not integer, we heuristically solve a STSP on the weighted *complete* graph on vertex set V (that is, here, unlike the separation of general QHC inequalities, edges $e \in E$ with $x^\star_e = 0$ are also considered), and the resulting QHC inequality associated to the STSP solution is checked for violation.

We begin by comparing $\mathcal{H}$ with $\mathcal{F}$ thus obtaining an indication of the impact of the general QHC inequalities on the performance of the HpMP formulation given in Section 3. A qualitative comparison is provided in the performance profile of Fig. 5a, reporting the number of instances (y-axis) solved within a given time threshold in seconds (x-axis). We see that $\mathcal{F}$ clearly dominates $\mathcal{H}$; furthermore, while the profile of $\mathcal{H}$ flattens out around 1000 seconds, the profile of $\mathcal{F}$ is steeper and increases with more CPU time available.

A synthetic quantitative comparison is also provided in the Table 9, reporting the number of HpMP instances of the entire benchmark set solved to optimality by $\mathcal{H}$ and $\mathcal{F}$ within the CPU time limit of 1 hour, the average relative optimality gap, computed on the subset of instances unsolved by both algorithms, and the average CPU time needed to reach optimality, computed on the subset of instances solved by both algorithms: out of the 155 instances composing the whole benchmark set, $\mathcal{F}$ solves 13 instances more than $\mathcal{H}$ and it is ~ 20 seconds faster in average than $\mathcal{H}$ on the subset of instances solved by both algorithms while reporting a slightly lower average optimality gap on the subset of instances unsolved by both algorithms.

Table 16 of F.4 reports the detailed instance-wise comparison of $\mathcal{H}$ and $\mathcal{F}$, omitted here for the sake of brevity. The main conclusions are the following: in general, algorithm $\mathcal{F}$ is faster than $\mathcal{H}$; in 26 cases with sizes of instances ranging from 22 to 400 vertices and mostly on instances having $n = 3p+1$, algorithm $\mathcal{F}$ at least halves the time needed to reach optimality with respect to $\mathcal{H}$, whereas the symmetric phenomenon of $\mathcal{H}$ at least halving the CPU times of $\mathcal{F}$ occurs only in 1 case (on instance `dantzig42` with $p = 10$); there is only one HpMP instance solved by algorithm $\mathcal{H}$ and not by $\mathcal{F}$, namely `u159` with $p = 31$; this exception appears to be instance-dependent, even though our log files indicate that the few instances for which $\mathcal{H}$ is faster than $\mathcal{F}$ are explained by considerably fewer violated inequalities added by $\mathcal{H}$ during the branch-and-cut process, and thus, to faster resolution of the intermediate LPs; all 13 HpMP instances solved by $\mathcal{F}$ and not by $\mathcal{H}$ have

Table 9: Comparison of the performance of algorithms $\mathcal{H}$ and $\mathcal{F}$.

	$\mathcal{H}$	$\mathcal{F}$
Solved instances	111/155	124/155
Avg. Opt. Gap (unsolved by both)	33.16%	29.36%
Avg. CPU Time (solved by both)	75.38 sec.	58.98 sec.

Table 10: Comparison of the performance of algorithms $\mathcal{F}^-$ and $\mathcal{F}$.

	$\mathcal{F}^-$	$\mathcal{F}$
Solved instances	119/155	124/155
Avg. Opt. Gap (unsolved by both)	29.12%	28.39%
Avg. CPU Time (solved by both)	82.35 sec.	67.81 sec.

$n = 3p$ or $n = 3p + 1$ indicating that the QHC inequalities (6) associated with 4- or 5-cycles make algorithm $\mathcal{F}$ effective on those instances (recall that the RCCs are used in both $\mathcal{H}$ and $\mathcal{F}$).

Next, we perform a similar comparison between algorithms $\mathcal{F}^-$ and $\mathcal{F}$, thus getting an indication of the impact of the restricted 2-opt inequalities (9). Results are synthetically summarized in the performance profiles of Fig. 5b and Table 10, constructed as those of the comparison between $\mathcal{H}$ and $\mathcal{F}$. The performance profiles show that both algorithms exhibit similar behaviours, with $\mathcal{F}$ only slightly dominating $\mathcal{F}^-$. From the table we see that $\mathcal{F}$ solves 5 instances more than $\mathcal{F}^-$ and takes ~ 15 seconds of CPU time less in average to reach optimality (on the subset of instances solved by both algorithms); on the subset of instances unsolved by both algorithms, the average relative optimality gap is essentially identical. The instance-wise comparison of $\mathcal{F}^-$ and $\mathcal{F}$ is reported in Table 17 of F.4. Again algorithm $\mathcal{F}$ is generally faster than $\mathcal{F}^-$. There are 19 cases in which algorithm $\mathcal{F}$ at least halves the CPU times of $\mathcal{F}'$, while the opposite situation occurs 8 times; there is no instance solved only by $\mathcal{F}^-$ and of the 5 instances solved only by $\mathcal{F}$ just one has $n = 3p + 1$, namely, `kroC100` with $p = 33$.

From the two comparisons above we conclude that the QHC inequalities have a strong impact on the HpMP formulation of Section 3; this is especially true when considering the difficult HpMP instances having $n = 3p, 3p + 1$, for which we separate the QHC inequalities associated with the smallest cycles by using a dedicated separation routine; the restricted 2-opt inequalities are also useful, although their impact does not seem to be as decisive.

9 Conclusion

In this paper we have studied the HpMP, a routing problem in which a weighted graph on n vertices must be partitioned into precisely p cycles of minimum total weight. We have focused on polyhedral and computational aspects of an ILP formulation only involving natural edge variables, since for several network design problems, the more efficient algorithms are those developed from formulations with a minimal set of variables (as for example the symmetric TSP).

Our study has started from the observation that an ILP based on edge variables had already been presented in the literature and included two types of inequalities, namely those enforcing at most p cycles and those enforcing at least p cycles in a solution to the problem. We have provided an intuitive theoretical argument showing that the former set of inequalities is more effective than the latter one in removing solutions to the 2-factor relaxation that are infeasible for the HpMP. Therefore we have developed new families of inequalities for these cases. The first one is the family of the quasi-Hamiltonian cycle inequalities and includes as special case the Hamiltonian cycle inequalities of the starting ILP. Their inclusion strictly tightens the linear relaxation of the starting formulation. Moreover, we have provided sufficient conditions for a subset of the quasi-Hamiltonian cycle inequalities to induce facets of the integer hull of the formulation.

Subsequently, we have considered the cases of the HpMP in which $p = \lfloor n/3 \rfloor$ and which are the most difficult to solve in practice, as shown by several previous works in the literature. For the cases with $n = 3p, 3p + 1$, we have introduced the restricted cut constraints, very similar to the cut constraints of the

symmetric TSP, except for additional requirements on the cardinality of the underlying cut shores; we have provided sufficient conditions for the restricted cut constraints to be facet-inducing for the integer hull of the considered formulation. For the case $n = 3p+2$ we have introduced the restricted 3-cut constraints, obtained in a similar manner. We have also observed that combining suitable cases of the quasi-Hamiltonian cycle inequalities and with these additional constraints we obtain alternative valid formulations for the HpMP in which $p = \lfloor n/3 \rfloor$.

Finally, we have introduced inequalities cutting off sub-optimal solutions and corresponding to specific 2-opt exchanges for the HpMP. They are inspired from the general framework of local search inequalities presented in [19].

A branch-and-cut algorithm involving all inequalities introduced in our paper has been compared on benchmark instances to state-of-the-art branch-and-cut algorithms. When solving to integer optimality it exhibits a very good performance. For small instances it outperforms state-of-the-art algorithms from the literature, probably due the difference in size of the respective underlying formulations (the existing algorithms relies on formulations in extended spaces). For larger instances our algorithm exhibits, in general, a good performance, always comparable or better than the performance of state-of-the-art algorithms for the HpMP. Given such good results, we have extended the benchmark by including 40 new instances with up to 400 vertices. Our algorithm solved 17 of them to optimality within 1 hour of computation (boldface values in Table 4). The largest instances solved to optimality have 400 vertices.

Our study has also led to some interesting research questions that we would like to investigate as a future work. The first one is to give sufficient and necessary conditions for the quasi-Hamiltonian cycle inequalities and the restricted cut and multi-cut constraints to be facet-inducing. The second research question concerns the generalization of the principle behind the quasi-Hamiltonian cycle inequalities to their cycle covering counterpart, with an emphasis on their polyhedral properties and on their effective usage in branch-and-cut algorithms. Finally, we would like to investigate generalizations of the restricted (multi-)cut constraints to cases other than those with $p = \lfloor n/3 \rfloor$ and their usage in alternative formulations for the HpMP.

References

[1] D. L. Applegate, R. E. Bixby, V. Chvátal, W. Cook, D. G. Espinoza, M. Goycoolea, and K. Helsgaun. Concorde TSP Solver. `https://www.math.uwaterloo.ca/tsp/concorde/index.html`.

[2] T. Bektaş, L. Gouveia, and D. Santos. Revisiting the Hamiltonian p-median problem: A new formulation on directed graphs and a branch-and-cut algorithm. *European Journal of Operational Research*, 2018.

[3] E. Benavent and A. Martínez. Multi-depot multiple TSP: a polyhedral study and computational results. *Annals of Operations Research*, 207(1):7–25, 2013.

[4] I. Branco and J. D. Coelho. The Hamiltonian p-median problem. *European Journal of Operational Research*, 47(1):86–95, 1990.

[5] M. Conforti, G. Cornuéjols, and G. Zambelli. Integer programming, volume 271 of Graduate Texts in Mathematics, 2014.

[6] G. Cornuéjols and W. R. Pulleyblank. Perfect triangle-free 2-matchings. In V.J. Rayward-Smith, editor, *Combinatorial Optimization II*, pages 1–7. Springer, 1980.

[7] B. Dezső, A. Jüttner, and P. Kovács. LEMON–an open source `C++` graph template library. *Electronic Notes in Theoretical Computer Science*, 264(5):23–45, 2011.

[8] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards B*, 69(125-130):55–56, 1965.

[9] G. Erdoğan, G. Laporte, and A. M. R. Chía. Exact and heuristic algorithms for the Hamiltonian p-median problem. *European Journal of Operational Research*, 253(2):280–289, 2016.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[11] H. Glaab. *Eine Variante des Traveling-Salesman-Problems mit mehreren Handlungsreisenden: Modelle, Algorithmen und Anwendung*. Shaker, 2000.

[12] H. Glaab. A new variant of a vehicle routing problem: Lower and upper bounds. *European Journal of Operational Research*, 139(3):557–577, 2002.

[13] H. Glaab and A. Pott. The Hamiltonian p–median problem. *The Electronic Journal of Combinatorics*, 7(1):R42, 2000.

[14] S. Gollowitzer, L. Gouveia, G. Laporte, D. L. Pereira, and A. Wojciechowski. A comparison of several models for the Hamiltonian p-median problem. *Networks*, 63(4):350–363, 2014.

[15] S. Gollowitzer, D. L. Pereira, and A. Wojciechowski. New models for and numerical tests of the Hamiltonian p-median problem. In *International Conference on Network Optimization*, pages 385–394. Springer, 2011.

[16] G. Gutin and A. P. Punnen. *The traveling salesman problem and its variations*, volume 12. Springer Science & Business Media, 2006.

[17] L. Hupp and F. Liers. A polyhedral study of the Hamiltonian p-median problem. *Electronic Notes in Discrete Mathematics*, 41:213–220, 2013.

[18] IBM. ILOG CPLEX 12.6 User Manual. `http://public.dhe.ibm.com/software/products/Decision_Optimization/docs/pdf/usrcplex.pdf`. Accessed: 2016-11-01.

[19] G. Lancia, F. Rinaldi, and P. Serafini. Local search inequalities. *Discrete Optimization*, 16:76–89, 2015.

[20] A. M. Marzouk, E. Moreno-Centeno, and H. Üster. A branch-and-price algorithm for solving the Hamiltonian p-median problem. *INFORMS Journal on Computing*, 28(4):674–686, 2016.

[21] M. Nägele. *Efficient Methods for Congruency-Constrained Optimization*. PhD thesis, ETH Zurich, 2021.

[22] M. Nägele, B. Sudakov, and R. Zenklusen. Submodular minimization under congruency constraints. *Combinatorica*, 39(6):1351–1386, 2019.

[23] G. Reinelt. TSPLIB—A traveling salesman problem library. *ORSA journal on computing*, 3(4):376–384, 1991.

[24] G. Reinhelt. TSPLIB: a library of sample instances for the TSP (and related problems) from various sources and of various types. *URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/*, 1995.

[25] M. Zohrehbandian. A new formulation of the Hamiltonian p-median problem. *Applied Mathematical Sciences*, 1(8):355–361, 2007.

A Algorithmic Details

Some details on the algorithms used as sub-routines in our branch-and-cut algorithms have been omitted in the main paper for the sake of brevity. In this section we provide more detailed descriptions.

A.1 Primal heuristic

In our primal heuristic we obtain a solution to the HpMP starting from the current fractional solution $x^\star$ by following the same idea presented in [14].

Firstly, we construct a weighted complete graph $\tilde{G} = (V, E)$ with weights $w_e = 1 - x^\star_e$ for every $e \in E$. Subsequently, we determine a minimum-weight spanning tree $\tilde{T}$ of $\tilde{G}$ by means of Kruskal's algorithm. We remove from $\tilde{T}$ the $p-1$ edges of largest weight, thus obtaining a forest of p trees, $T_1, T_2, \ldots, T_p$. Some of these trees may contain less than three vertices. In this case we move vertices from *large* trees (*i.e.,* with at least 4 vertices) to *small* trees (*i.e.,* with less than 3 vertices) as follows: we move from a large tree K_1 to a small tree K_2 the vertex v that maximizes the quantity $w(v : K_2) - w(v : K_1)$. Every time a vertex is moved, the list of large and small trees is updated and the process is repeated on the updated list, until all trees have at least three vertices. As before, we let $T_1, T_2, \ldots, T_p$ indicate the trees obtained after the cardinality adjustment described above. For every $i = 1, 2, \ldots, p$ we run a nearest neighbor method for the STSP on the complete subgraph $\tilde{G}[T_i]$, thus obtaining a set of p disjoint cycles Each one of these cycles is modified in order to ensure the feasibility of the resulting solution with respect to the restricted 2-opt inequalities (9), by executing the 2-opt exchanges associated to the sequences of 4 consecutive vertices in the cycle, so to satisfy the condition given in Remark 1. The resulting set of p cycles is our feasible solution.

In our code we use the COIN-OR library LEMON [7] to run both Kruskal algorithm and the nearest neighbor method on graph $\tilde{G}$. The whole primal heuristic has $O(n^2)$ time complexity.

A.2 Separation of QHC inequalities for $x^\star \in \{0, 1\}$.

Let $C_1, C_2, \ldots, C_\ell$ be the cycle partition of G corresponding to solution $x^\star$. If $\ell \geq p$ the algorithm terminates since, by Proposition 1, the QHC inequalities are valid for $\mathcal{X}^n_{\geq p}$, hence none of them is violated. Otherwise, we have $\ell < p$, and we first check if each of these ℓ cycles alone leads to a violated QHC inequality. All violated inequalities found in this way are added to the current formulation and, if at least one violated inequality has been added, the separation algorithm terminates. If no single cycle in the partition yields a violated QHC inequality we attempt to merge them to find QHC inequalities associated to larger ones. First, we sort them by non-decreasing order of their length. Without loss of generality, let the ordered list of cycles be $C_1, C_2, \ldots, C_\ell$ such that $|C_i| \leq |C_{i+1}|$ for every $i = 1, 2, \ldots, \ell - 1$. We find the smallest t such that $K = \bigcup_{i=1}^t C_i$ satisfies $p - \left\lfloor \frac{n-|K|}{3} \right\rfloor \geq 2$ and $|K| - t - 1 > |K| - p + \left\lfloor \frac{n-|K|}{3} \right\rfloor$. The latter condition guarantees that merging $C_1, C_2, \ldots, C_t$ into a single cycle C (in the way explained below) yields the violated QHC inequality $x(C) \leq |C| - p + \left\lfloor \frac{n-|C|}{3} \right\rfloor$. Note also that taking $t = \ell$ always satisfies the two previous conditions because in that case $|K| = n$ and $\ell < p$. Hence the required minimum certainly exists. The cycles are merged as follows. For each $i = 1, 2, \ldots, \ell$ we remove from each C_i the edge having largest objective coefficient. Then the resulting paths are linked in the order given by the sorting step, that is, the path resulting from C_1 is linked to the path resulting from C_2, this latter is linked to the path resulting from C_3, and so on, until linking back the path resulting from C_ℓ to the pending endpoint of the one resulting from C_1.

We point out that the algorithm just described to separate the QHC inequalities from an integer solution is an exact separation algorithm. Furthermore, since the QHC inequalities together with the inequalities (1), (2)

and (4) are sufficient to define a valid formulation for the HpMP, the correctness of the above procedure together with the correctness of the separation of inequalities (2), guarantees that our branch-and-cut algorithms separating (2) and (6) always yields an optimal solution to the HpMP.

A.3 Separation of QHC inequalities for $x^\star \notin \{0,1\}^{|E|}$.

We start by determining the connected components $K_1, K_2, \ldots, K_\ell$ of the support graph $G_{x^\star}$. If $\ell \geq p$, the procedure stops, reporting no violated cut. Otherwise, the components are sorted by non-decreasing order of their size (vertex cardinality). Without loss of generality, let us assume that $|K_i| \leq |K_{i+1}|$ for all $i = 1, 2, \ldots, \ell - 1$.

Next, we find the minimum $t \leq \ell$ such that $|K_1| + |K_2| + \cdots + |K_t| \geq n - 3p + 4$. Then a STSP is solved on the subgraph of $G_{x^\star}$ induced by $V(K_1) \cup \cdots \cup V(K_t)$ with edge weights $w_e = 1 - x^\star_e$ (we take the complement of the fractional solution values so that when computing a minimum-weight Hamiltonian cycle C in the subgraph we maximize the evaluation of $x^\star$ on the left-hand side of the QHC inequality associated with C). In our implementation we have solved the STSP heuristically and have used the nearest neighborhood heuristic provided by the COIN-OR library LEMON [7]. We avoid running the heuristic separately on each component to limit the computational burden of the procedure. If the cycle C corresponding to the STSP solution obtained in the previous step yields a violated inequality, this inequality is added to the pool of violated cuts.

Additionally, if $|C| > n - 3p + 4$, we search for other violated inequalities by modifying the cycle C according to the two following 2 rules:

1. Remove vertices from C obtaining, sequentially, cycles with one vertex less than the previous cycle in the sequence (with the idea of reaching a cycle of smallest length). At each step we check that the current cycle has cardinality at least $n - 3p + 5$, so that the vertex removal still yields a valid QHC inequality. If such preliminary condition holds, we consider all triples of vertices u, v and w appearing in sequence in C. We determine the inequality associated with cycle C/v, where the latter is obtained from cycle C by removing edges (u, v) and (v, w) and adding edge (u, w) such that:
   - $\left\lfloor \frac{n-|C|}{3} \right\rfloor = \left\lfloor \frac{n-|C/v|}{3} \right\rfloor$
   - $x^\star_{\{uv\}} + x^\star_{\{vw\}} < 1$

   As soon as we are not able to produce new violated QHC inequalities with the method given above, we proceed to rule 2 described below.

2. Break C into cycles of smallest length: we consider all sub-paths of consecutive $n-3p+4$ vertices of C and we link the endpoints of each such paths so to get a family of cycles each of cardinality $n-3p+4$; if any of the cycles obtained in this way leads to a violated inequality, it is added to the pool of violated constraints.

The above procedure finds candidate cycles associated to violated QHC inequalities, starting from the first t connected components of $G_{x^\star}$. The same procedure is then repeated by increasing the value of t by 1, that is, by considering one additional connected component at a time, until all of them have been added. Note that in this last case, the procedure starts by searching for Hamiltonian cycle inequalities (*i.e.,* those having a support graph covering all vertices).

B Inequalities (6) Tighten the Linear Relaxation of (1)–(4)

In Section 4 we observed that the family of QHC inequalities (6) strictly tightens the linear relaxation of (1)–(4). This is formally stated and proven in the following proposition.

Proposition 10. *Let $p \geq 2$, $n \geq 3p+1$ and let*

$$R_{\geq p} = \{x \in [0,1]^{|E|} : x \text{ satisfies } (1),(2),(3)\},$$
$$R^{+}_{\geq p} = \{x \in [0,1]^{|E|} : x \text{ satisfies } (1),(2),(6)\}.$$

Then $R^{+}_{\geq p} \subset R_{\geq p}$.

Proof. The weak inclusion in the result holds because family (6) includes inequalities (3). We now prove that the inclusion is strict by constructing a fractional point of $\mathrm{R}_{\geq p}$ which is cut off by an inequality of family (6). Let us fix $(p-2)$ cycles $C_1, C_2, \ldots, C_{p-2}$ of G such that $|C_i| = 3$ for $i = 1, 2, \ldots, p-2$ and let the vertices of G not contained in such cycles be numbered from 1 to $m = n - 3p + 6$. We consider the subgraph F of G given by the union of $C_1, C_2, \ldots, C_{p-2}$ with the subgraph F' induced by the following edge set:

$$\{\{i, i+1\}, \{1, m\}, \{1, m-1\}, \{2, m-1\}, \{2, m\}\{1, 3\}, \{m-2, m\} : i = 1, 2, \ldots, m-1\}\,.$$

Note that F' is a cycle of G with five chords (see Figure 6 for an illustration with $m = 10$ and $p = 5$).

Let us assign a weight of 1 to all edges of F except to those five chords and to edges $\{1,2\}$, $\{2,3\}$, $\{1,m\}$, $\{m-1,m\}$, $\{m-1,m-2\}$ which are assigned a weight of $1/2$. All other edges of G are given a weight of 0.

The point $\mathbf{x} \in [0,1]^{|E|}$ whose coordinate $\mathbf{x}_e$ has a value corresponding to the weight of edge e, belongs to $\mathrm{R}_{\geq p}$. First, it is easily checked that $\mathbf{x}$ satisfies the degree inequalities (1). Moreover, it is a $\frac{1}{2}$-combination of two partitions of G into $p-1$ cycles, hence it satisfies all constraints (2); more precisely, both such partitions contain cycles $C_1, C_2, \ldots, C_{p-1}$ and the first one additionally contains cycle $(1, 2, \ldots, m-1, m, 1)$ while the second partition contains cycle $(1, 3, 4, \ldots, m-2, m, 2, m-1, 1)$.

Now we show that $\mathbf{x}$ satisfies all inequalities of type (3) thus getting $\mathbf{x} \in \mathrm{R}_{\geq p}$. Indeed, let H be a Hamiltonian cycle defining an inequality (3). Then H can intersect cycles $C_1, C_2, \ldots, C_{p-2}$ in at most $2(p-2)$ edges; moreover, the subpath of F' linking vertex 3 with vertex $m-2$ through edges of weight 1 has length $n-3p+1$ and every subpath of F' linking the vertices in $\{1, 2, 3, m-2, m-1, m\}$ through edges of weight $1/2$ has total weight $5/2$. Therefore the intersection of H with F has a total weight of at most $n-p-1/2$, that is, $\mathbf{x}(H) \leq n-p-1/2 < n-p$.

Finally, we prove that $\mathbf{x} \notin R^{+}_{\geq p}$. Indeed, let us consider the QHC inequality $x(C) \leq |C|-2$ associated with the cycle $C = (1, 3, 4, \ldots, m-2, m, 1)$ of length $n-3p+4$. Since $|C|-2 = n-3p+2$ and $\mathbf{x}(C) = n-3p+5/2$ we have that $\mathbf{x}$ violates the considered QHC inequality, hence it does not belong to $R^{+}_{\geq p}$. □

C Proof of Proposition 3

By our hypothesis, C is an odd cycle of G of length $n-3p+4$ and the corresponding QHCC inequality is simply:

$$x(C) \leq |C| - 2. \tag{11}$$

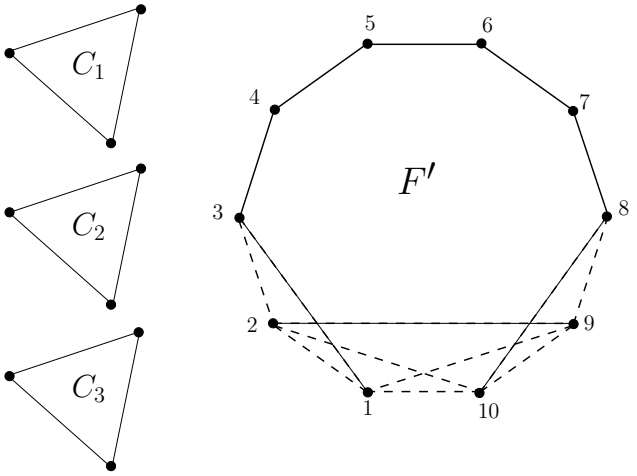


Figure 6: Illustration of the subgraph F and of the fractional point $\mathbf{x}$ defined in the proof of Proposition 10 for $p = 5$ and $m = 10$. All edges in the graph have a weight of 1 except the dashed ones which have a weight of 1/2.

Moreover, since $n \geq 3p+3$, we have $\dim(\mathcal{X}_p^n) = \frac{n(n-1)}{2} - n$ by a result of [17] so to prove that (11) defines a facet of $\mathcal{X}_p^n$ we will exhibit a set $\mathcal{F}$ of $\dim(\mathcal{X}_p^n)$ affinely independent points of $\mathcal{X}_p^n$ satisfying (11) with equality. This latter requirement implies that $\mathcal{F}$ will contain the incidence points of HpMP solutions excluding exactly two edges of cycle S. For a clear description of such solutions we give a few definitions.

First, we define $I = \{1, 2, \ldots, |C|\}$ and $N = \{|C|+1, \ldots, n\}$. Without loss of generality we number the vertices of C consecutively, by using the indices in I and we number the indices of $V \setminus V(C)$ using the indices in N. Next, we define two *consecutive chords* of C to be any choice of pairs (i_1, j_1) and (i_2, j_2) of indices of I, such that $\{i_1, i_2\}$ and $\{j_1, j_2\}$ are edges of C. Let also $\mathfrak{C}$ be the set of ordered pairs of consecutive chords (c_1, c_2) where $c_1 = (i_1, j_1)$ and $c_2 = (i_2, j_2)$, with i_1 and j_1 belonging to the odd path obtained from C after removing edges $\{i_1, i_2\}$ and $\{j_1, j_2\}$.

Given $(c_1, c_2) \in \mathfrak{C}$, we define $\mathcal{A}^{c_1,c_2} \subseteq E(C)$ as the subset containing the edges $\{i_1, j_1\}$, $\{i_2, j_2\}$ and all the edges of C except $\{i_1, i_2\}$ and $\{j_1, j_2\}$; similarly, given the edge $e = \{i_1, j_1\}$ of C and the chord $c = \{i_2, j_2\}$ of C with i_2 (resp. j_2) the vertex of C adjacent to i_1 (resp. j_1) , we define $\mathcal{B}^{e,c} \subseteq E(C)$ as the subset containing all the edges of C except $\{i_1, i_2\}$ and $\{j_1, j_2\}$, and containing two edges linking vertex $|C|+1$ to i_1 and j_1 and two edges linking vertex $|C|+2$ to i_2 and j_2.

Finally, for $i \in I$ and $j, k \in N$, we define $\mathcal{C}_{j,k}^i \subseteq E$ as the subset of edges containing $\{i, j\}$, $\{i, k\}$, $\{j, k\}$ and all edges of C except those incident to i. An example of the three kinds of edge subsets is given in Figure 1 of the main paper.

Note that all subsets of types defined above contain two cycles. Hence, if T is an edge subset of one of the three types above and P is an edge subset yielding a partition of $V \setminus V(T)$ into $(p-2)$ cycles, the set of edges $T \cup P$ is a HpMP solution, whose incidence vector will be denoted (T, P). Note that, since T contains at most two vertices of $V \setminus V(C)$ and $|V \setminus V(C)| = 3p-4$ it is always possible to partition $V \setminus V(T)$ into $(p-2)$ cycles.

Now, recall that $d := \dim(\mathcal{X}_{p-2}^{3p-4}) = \frac{(3p-4)(3p-5)}{2} - (3p-4)$, since $3p-4 > 3(p-2)+1$. Therefore, there exist $d+1$ edge subsets partitioning $V \setminus V(C)$ into $(p-2)$ cycles such that their incidence points $S_1, S_2, \ldots, S_{d+1}$ are affinely independent.

Next, we observe that the incidence vectors of the solutions in $\mathcal{A} = \{\mathcal{A}^{c_1,c_2} \colon (c_1, c_2) \in \mathfrak{C}\}$ are affinely

independent points. To prove this we define a *chord-path* to be a sequence $(d_1, d_2, \ldots, d_k)$ of chords, such that $(d_i, d_{i+1}) \in \mathfrak{C}$. Note that, since C is odd, a chord-path $(d_1, d_2, \ldots, d_M)$ is maximal (with respect to the inclusion in a longer chord-path) if $M = \frac{|C|-3}{2}$. Moreover, every chord belongs to exactly one maximal chord-path. Now, let us consider two consecutive chords c_1 and c_2 and a maximal chord-path $(d_1, d_2, \ldots, d_M)$ such that $d_j = c_1$ and $d_{j+1} = c_2$ for some $j \in \{1, 2, \ldots M-1\}$. Then the equation $\sum_{\ell=1}^{j}(-1)^\ell x_{d_\ell} + \sum_{\ell=j+1}^{M}(-1)^{\ell-1} x_{d_\ell} = 0$ is violated by the incidence vector of $\mathcal{A}^{c_1,c_2}$ whereas it is satisfied by all other incidence vectors of subsets of $\mathcal{A}$. This shows that the incidence vectors of the elements of $\mathcal{A}$ are affinely independent.

Let $\overline{A} \in \mathcal{A}$ be fixed; we define $\mathcal{F}' = \{(A, S_1) \colon \forall A \in \mathcal{A}\}$ and $\mathcal{F}'' = \{(\overline{A}, S_j) \colon 2 \le j \le d+1\}$. By the affine independence of the incidence vectors of $S_1, S_2, \ldots, S_{d+1}$ and those of the elements of $\mathcal{A}$, we have that $\mathcal{F}^1 := \mathcal{F}' \cup \mathcal{F}''$ contains $|\mathcal{A}| + d$ affinely independent incidence vectors of HpMP solutions.

We extend $\mathcal{F}^1$ to a larger set of affinely independent points. In order to do so, we first need to introduce a one-to-one correspondence between the maximal chord-paths and the vertices of C. Given $i \in V(C)$ (or, equivalently, $i \in I$), let us call d^i the chord that links the two vertices adjacent to i in C. The one-to-one correspondence between maximal chord-paths and vertices of C now follows since there exists exactly one maximal chord-path $(d^i_1, d^i_2, \ldots, d^i_M)$ such that $d^i_M \equiv d^i$. With the notation just introduced, we define the following expressions:

$$E^i(x) = \sum_{\ell=1}^{M}(-1)^\ell x_{d^i_\ell} \qquad \forall i \in I.$$

For every $\mathcal{A}^{c_1,c_2}$ there exists a unique $\bar{\imath} \in I$ such that c_1 and c_2 belong to the maximal chord-path associated with $\bar{\imath}$. Then $E^{\bar{\imath}}(\mathcal{A}^{c_1,c_2}) = 0$ since x_{c_1} and x_{c_2} have opposite sign in $E^{\bar{\imath}}(x)$. Since c_1 and c_2 do not belong to any other chord-path, $E^i(\mathcal{A}^{c_1,c_2}) = 0$ also for every $i \neq \bar{\imath}$. Then all points in $\mathcal{F}^1$ satisfy the equations

$$(-1)^M x_{\{i,|C|+1\}} - E^i(x) = 0 \qquad \forall i \in I. \tag{12}$$

For every $j, k \in N$ let us denote $P_{j,k}$ an arbitrary subset of edges partitioning the vertices in $N \setminus \{j,k\}$ into $(p-2)$ cycles. For every $i \in I$, we define

$$\mathcal{C}^i = \left\{(\mathcal{C}^i_{|C|+1,k}, P_{|C|+1,k}) \colon k \in N \setminus \{|C|+1\}\right\}.$$

All points in $\bigcup_{i\in I}\mathcal{C}^i$ also satisfy equation (12). Moreover, for every $i \in I$ and $k \in N \setminus \{|C|+1\}$, the point $(\mathcal{C}^i_{|C|+1,k}, P_{|C|+1,k})$ is not affinely dependent on the other points of $\mathcal{F}''' := \mathcal{F}^1 \cup \bigcup_{i\in I}\mathcal{C}^i$ since it is the only one to violate the equation $x_{\{i,k\}} = 0$.

Subsequently, let $\mathcal{F}^2 = \mathcal{F}''' \cup \{(\mathcal{C}^i_{|C|+2,|C|+3}, P_{|C|+2,|C|+3}) \colon i \in I\}$. The set $\mathcal{F}^2$ is an affinely independent set because, given $i \in I$, the point $(\mathcal{C}^i_{|C|+2,|C|+3}, P_{|C|+2,|C|+3})$ is the only one to violate equation (12) for index i.

At this point we show that $|\mathcal{F}^2| = \dim(\mathcal{X}^n_p) - |C|$. First, we point out that, by the construction above, $|\mathcal{F}^2| - |\mathcal{F}^1| = |C|(n - |C|)$. Then we show that

$$|\mathcal{F}^1| = |\mathcal{A}| + d = \frac{|C|(|C|-5)}{2} + \frac{(n-|C|)(n-|C|-1)}{2} - (n-|C|).$$

We only need to prove that $|\mathcal{A}| = \frac{|C|(|C|-5)}{2}$ since, by $\dim(\mathcal{X}^m_{p-2}) = \frac{m(m-1)}{2} - m$, we get that $d =$

$\frac{(n-|C|)(n-|C|-1)}{2} - (n-|C|)$.

Indeed, recall that each maximal chord-path $(d_1^i, d_2^i, \ldots, d_M^i)$ uniquely corresponds to vertex $i \in I$ and has $M = \frac{|C|-3}{2}$. Moreover, each maximal chord-path originates $M-1$ elements of $\mathfrak{C}$ (by taking (d_j^i, d_{j+1}^i) for $1 \leq j \leq M-1$) and each element of $\mathfrak{C}$ belongs to exactly one maximal chord-path. Therefore we have

$$|\mathcal{A}| = |\mathfrak{C}| = |C|(\frac{|C|-3}{2} - 1) = \frac{|C|(|C|-5)}{2}.$$

From $|C| = n - 3p + 4$, it is standard calculation to show that

$$\begin{aligned} |\mathcal{F}^2| &= \frac{|C|(|C|-5)}{2} + \frac{(n-|C|)(n-|C|-1)}{2} - (n-|C|) + |C|(n-|C|) \\ &= \frac{n(n-3)}{2} - |C| = \dim(\mathcal{X}_p^n) - |C| \end{aligned} .$$

The above calculation implies that it suffices to extend $\mathcal{F}^2$ with additional $|C|$ affinely independent points. We start by extending $\mathcal{F}^2$ iteratively as follows. At iteration j let F^j be the set of affinely independent points added to $\mathcal{F}^2$ until iteration $j-1$. We will find an equation satisfied by all points in $\mathcal{F}^2 \cup F^j$ and a feasible solution H^j to the HpMP whose incidence vector violates such an equation but satisfies the initial QHC inequality with equality. This shows that the new feasible solution cannot be affinely dependent on the points in $\mathcal{F}^2 \cup F^j$. Iteration j concludes by setting $F^{j+1} = F^j \cup \{H^j\}$.

First, let us observe that all points in $\mathcal{F}^2$ satisfy:

$$(-1)^M x(i : V \setminus V(C)) - 2E^i(x) = 0 \qquad \forall\, i \in I.$$

Denoting $L^i(x)$ the left-hand-side of the above equation, this implies that all points in $\mathcal{F}^2$ also satisfy the equations:

$$\sum_{i=1}^{m} (-1)^{i+1} L^i(x) = 0 \qquad \forall\, 3 \leq m \leq |C|. \tag{13}$$

Now for each $m = 3, 4, \ldots, |C|$ we consider the edge $c_1^m = (m-1, m)$ and the chord $c_2^m = (m-2, m+1)$ (here we assume $|C|+1 \equiv 1$). Then we extend $\mathcal{F}^2$ by adding (in the order given by the indices) the incidence vectors of the HpMP solutions $B^m = (\mathcal{B}^{c_1^m, c_2^m}, P_{|C|+1,|C|+2})$. For every $m \in \{3, 4, \ldots, |C|\}$, point B^m violates equation (13) associated with m while satisfying the ℓ-th equation (13) for $m+1 \leq \ell \leq |C|$ (the case $\ell = |C|$ holds because $|C|$ is odd).

To finish the proof we need to add two other points. We consider two distinct cases.

Case $\frac{|C|-3}{2}$ odd. In this case all points added so far satisfy equation:

$$\sum_{k=0}^{(|C|-1)/2} (-1)^k L^{2k+1}(x) + \sum_{k=1}^{(|C|-1)/2} (-1)^{k+1} L^{2k}(x) = 0.$$

The latter is violated by the incidence vector of the solution given by cycles $(1, 2, 3, 4, 5, |C|+1)$, $(6, 7, \ldots |C|, |C|+2)$ and $(p-2)$ cycles partitioning $N \setminus \{|C|+1, |C|+2\}$. Now this new solution and all those previously added

satisfy the following equation:

$$\sum_{k=1}^{(|C|-1)/4} L^{4k-1}(x) - \sum_{k=1}^{(|C|-1)/4} L^{4k}(x) = 0,$$

which is violated by the incidence vector of the solution consisting of cycles $(1, 2, |C|+1)$, $(3, 4, \ldots, |C|, |C|+2)$ and $(p-2)$ cycles partitioning the vertices in $N \setminus \{|C|+1, |C|+2\}$.

Case $\frac{|C|-3}{2}$ even. In this case all points added so far satisfy equation:

$$\sum_{k=1}^{(|C|-1)/2} (-1)^k L^{2k}(x) + \sum_{k=0}^{(|C|-1)/2} (-1)^k L^{2k+1}(x) = 0.$$

Hence, we further extend our set of affinely independent points by adding the incidence vector of the solution consisting of cycles $(|C|, 1, 2, |C|+1)$, $(3, 4, \ldots, |C|-1, |C|+2)$ and $(p-2)$ cycles of $N \setminus \{|C|+1, |C|+2\}$. This latter point and all previously added points also satisfy equation

$$\sum_{k=0}^{(|C|-3)/4} L^{2+4k}(x) - \sum_{k=0}^{(|C|-3)/4} L^{3+4k}(x) = 0.$$

This latter is violated by the incidence vector of the solution given by cycles $(1, |C|, |C|+1)$, $(2, 3, \ldots, |C|-1, |C|+2)$ and $(p-2)$ cycles of $N \setminus \{|C|+1, |C|+2\}$.

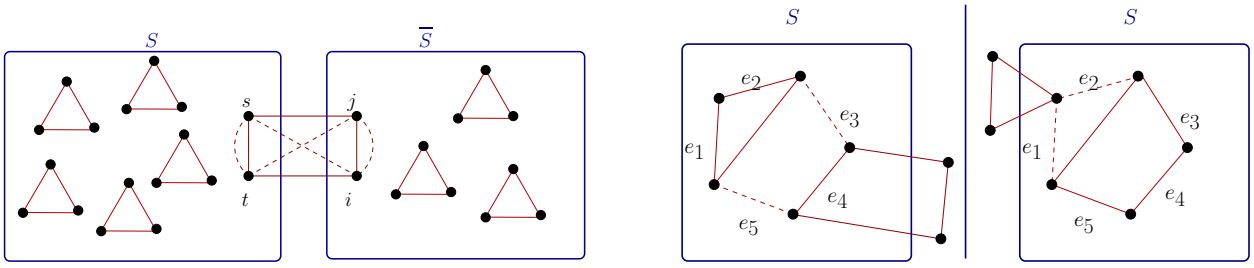
D Proof of Proposition 5

Let us consider a subset S such that $|S| = 3k+2$ for some integer $1 \leq k \leq p-1$. Let $\mathcal{F} = \{x \in \mathcal{X}_p^{3p+1} \colon x(\delta(S)) = 2\}$ be the face induced by inequality (7) associated with S and let $\mathcal{F}'$ be a facet containing $\mathcal{F}$, defined by an inequality $ax \leq b$ valid for $\mathcal{X}_p^{3p+1}$. We show that $\mathcal{F} = \mathcal{F}'$, the inclusion $\mathcal{F} \subseteq \mathcal{F}'$ holding by definition of $\mathcal{F}'$.

Using the same technique provided in [5, p. 300–301] we now show that (without loss of generality) $a_e = 0$ for all $e \in \delta(S)$. First we select a vertex $s \in S$ and using the degree constraints on the vertices in $V \setminus S$ we get the identity $\sum_{i \in V \setminus S} a_{\{s,i\}} x(\delta(i)) = 2 \sum_{i \in V \setminus S} a_{\{s,i\}}$. Subtracting the latter identity from $ax \leq b$ we can suppose without loss of generality that $a_{\{s,i\}} = 0$ for all $i \in V \setminus S$.

Let us now consider $t \in S \setminus \{s\}$. Since $|S| = 3k+2$ with $1 \leq k \leq p-1$ we can construct a solution to the HpMP with $n = 3p+1$ as follows. First we select distinct $i, j \in V \setminus S$; then we subdivide $S \setminus \{s,t\}$ in $(|S|-2)/3$ cycles, $V \setminus (S \cup \{i,j\})$ in $(|V \setminus S|-2)/3$ cycles and we add the cycle $C = \{\{s,t\}, \{t,i\}, \{i,j\}, \{s,j\}\}$. Note that by considering $C' = \{\{s,t\}, \{t,j\}, \{i,j\}, \{s,i\}\}$ we obtain another feasible solution to the HpMP with $n = 3p+1$ (see Figure 7a for an illustration). Moreover, letting $\mathbf{x}$ and $\mathbf{x}'$ be the incidence vectors of the above-mentioned solutions we have that $\mathbf{x}(\delta(S)) = 2 = \mathbf{x}'(\delta(S))$, that is $\mathbf{x}, \mathbf{x}' \in \mathcal{F} \subseteq \mathcal{F}'$. Hence $a\mathbf{x} = b = a\mathbf{x}'$ which in turn implies $a_{\{s,i\}} + a_{\{t,j\}} = a_{\{s,j\}} + a_{\{t,i\}}$. Since also $a_{\{s,i\}} = a_{\{s,j\}} = 0$ we conclude that $a_{\{t,j\}} = a_{\{t,i\}} = \lambda_t$ for some real value λ_t and $t \in S$.

Subtracting for every $t \in S$ the identities $\sum_{t \in S} \lambda_t x(\delta(t)) = 2 \sum_{t \in S} \lambda_t$ from $ax \leq b$ we get that, without loss of generality, $a_e = 0$ for $e \in \delta(S)$. Again following the same idea of [5], we show that $a_e = c$ for all $e \in E(S)$, where c is a constant value. More precisely we show that, given a cycle C of length 5, of $G[S] = (V(S), E(S))$ then $a_e = a_{e'}$ for all $e, e' \in C$. Then the result will follow by covering $E(S)$ with a set of



(a) Cycle C' is drawn with dashed edges. It is obtained from C (drawn with solid edges) by switching the two edges of $\delta(S)$.

(b) Illustration of the two solutions constructed from cycle C in the second step of the proof.

Figure 7

cycles of length 5 such that each cycle contains at least one edge of another cycle. Note that since $|S| = 3k+2$ and $k \geq 1$ the graph $G[S]$ contains at least 5 vertices so there exists at least a 5-cycle C contained in $G[S]$. Let $e_1, e_2, \ldots, e_5$ be the edges of C such that e_i and e_{i+1} have a common endpoint (we assume that $e_6 \equiv e_1$).

We construct a solution H to the HpMP as follows (see Figure 7b for an illustration). We subdivide the vertices in $S \setminus V(C)$ in $k-1$ cycles of length 3. Then we consider the cycle of length 3 containing e_1 and e_2 and a cycle of length 4 containing e_4 and two edges linking the endpoints of e_4 to two distinct vertices of $V \setminus S$. Finally we subdivide all other vertices of $V \setminus S$ into cycles of length 3. Denoting $\mathbf{x}$ the incidence vector of such a solution we have $\mathbf{x}(\delta(S)) = 2$, hence $a\mathbf{x} = b$ since $\mathbf{x} \in \mathcal{F} \subseteq \mathcal{F}'$.

Let us now consider another solution H' constructed as follows. We subdivide all vertices of $S \setminus V(C)$ in cycles of length 3. Then we consider the cycle of length 4 containing e_3, e_4 and e_5 and a cycle of length 3 containing two edges linking the common point of e_1 and e_2 to two distinct vertices in $V \setminus S$. Finally we subdivide all remaining vertices in cycles of length 3. Denoting by $\mathbf{x}'$ the incidence vector of this latter solution we get, as above, that $a\mathbf{x}' = b$. Note that, by construction, $(H \triangle H') \setminus \delta(S) = \{e_1, e_2, e_3, e_5\}$ and from $a_e = 0$ for all $e \in \delta(S)$, the identity $a\mathbf{x} = b = a\mathbf{x}'$ implies the identity $a_{e_1} + a_{e_2} = a_{e_3} + a_{e_5}$.

We can now construct similar identities involving the edges of cycle C by "rotating" the edges in the construction of the above two solutions along the cycle C. We get the following system of four identities:

$$\begin{aligned} a_{e_1} + a_{e_2} &= a_{e_3} + a_{e_5} \\ a_{e_1} + a_{e_4} &= a_{e_2} + a_{e_3} \\ a_{e_2} + a_{e_5} &= a_{e_3} + a_{e_4} \\ a_{e_3} + a_{e_1} &= a_{e_4} + a_{e_5}. \end{aligned}$$

This system yields $a_e = c$ for all $e \in C$ and some constant $c \in \mathbb{R}$. We can repeat the constructions above on the edges of $E(V \setminus S)$, since $p \geq 3$ and $|V \setminus S| = 3\ell + 2$ for some integer $\ell \geq 1$. In this case we get $a_e = \bar{c}$ for all $e \in E(V \setminus S)$. Hence the inequality defining $\mathcal{F}'$ can be rewritten as $cx(S) + \bar{c}x(V \setminus S) \leq b$. Since any point $\mathbf{x} \in \mathcal{F} \subseteq \mathcal{F}'$ satisfies the previous expression with equality and since $\mathbf{x} \in \mathrm{vert}(\mathcal{F})$ only if $\mathbf{x}(S) = |S| - 1$ and $\mathbf{x}(V \setminus S) = |V \setminus S| - 1$, we get that $b = c(|S| - 1) + \bar{c}(|V \setminus S| - 1)$.

In particular we have:

$$c\mathbf{x}'(S) + \bar{c}\mathbf{x}'(V \setminus S) = c(|S| - 1) + \bar{c}(|V \setminus S| - 1) \qquad \forall \mathbf{x}' \in \mathcal{F}' \tag{14}$$

Since $\mathcal{F}'$ is a proper face of $\mathcal{X}_p^{3p+1}$ at least one of c and $\bar{c}$ is nonzero. Using the degree constraints (1) it is

immediate to get that

$$x(V \setminus S) = |V \setminus S| - |S| + x(S) \tag{15}$$

is an implicit equality for $\mathcal{X}_p^{3p+1}$. Such an equality coincides with the expression in (14) if $c = -\bar{c}$. Then $c \neq -\bar{c}$: since $\mathcal{F}'$ is a proper face of $\mathcal{X}_p^{3p+1}$ there must exist a point of $\mathcal{X}_p^{3p+1} \setminus \mathcal{F}'$ violating (14), hence (14) cannot be an implicit equality. Now, using (15) casts (14) into $(c+\bar{c})\mathbf{x}'(S) = (c+\bar{c})(|S|-1)$. From $c+\bar{c} \neq 0$ we get $\mathbf{x}'(S) = |S|-1$ for every $\mathbf{x}' \in \mathcal{F}'$. Since $x(S) \leq |S|-1$ defines $\mathcal{F}$ we obtain that $\mathcal{F}' = \mathcal{F}$.

E Proofs of Proposition 6 and Proposition 9

Proof of Proposition 6. Since $|S_1| \equiv 2 \pmod 3$ and $|S_2| \equiv 2 \pmod 3$ there exist $k_1, k_2 \in \mathbb{Z}_+$ such that $|S_1| = 3k_1 + 2$ and $|S_2| = 3k_2 + 2$. Since S_1, S_2 and S_3 partition V and $n = 3p+2$ there is also $k_3 \in \mathbb{Z}_+$ such that $|S_3| = 3k_3 + 1$ and $k_1 + k_2 + k_3 = p - 1$. Let us assume, by contradiction, that there exists a feasible solution to the HpMP with $n = 3p + 2$ having no edge in the 3-cut $\delta(S_1, S_2, S_3)$. Then each shore S_i for $i = 1, 2, 3$ is partitioned in cycles in such a solution. However, S_1 and S_2 can contain at most k_1 and k_2 cycles respectively, because $|S_1| = 3k_1 + 2$ and $|S_2| = 3k_2 + 2$. Moreover, S_3 can contain at most k_3 cycles. Hence the solution has at most $p - 1$ cycles, a contradiction to its feasibility. It follows that all feasible solutions to the HpMP with $n = 3p + 2$ has at least one edge in the 3-cut $\delta(S_1, S_2, S_3)$. Since such edge belongs to a cycle, this latter has at least another edge in the 3-cut. Hence $x(\delta(S_1, S_2, S_3)) \geq 2$ is valid for the $\mathcal{X}_p^{3p+2}$.

Proof of Proposition 9. Let $\mathbf{x}$ be an integer point satisfying the degree constraints (1), the QHC inequalities (6) associated to 6-cycles. Then $\mathbf{x}$ is the incidence vector of a partition in cycles of G such that all cycles have length between 3 and 5. The result will follow by showing that for such partition only the following cases may hold: *i*) either it contains precisely one 5-cycle and $(p-1)$ 3-cycles or *ii*) it contains precisely two 4-cycles and $(p-2)$ 3-cycles. First there exists at most one 5-cycle, as otherwise denoting by C^1 and C^2 two 5-cycles of the partition the R3CC associated with the subsets $V(C^1), V(C^2)$ and $V \setminus (V(C^1) \cup V(C^2))$ will be violated. If there exists precisely one 5-cycle C^1 in the partition then there cannot exist a 4-cycle: otherwise, there should be at least two 4-cycles C^2 and C^3 (since $n = 3p + 2$) and thus the R3CC associated with $V(C^1)$, $V(C^2) \cup V(C^3)$ and their complement would be violated. Hence if the partition contains precisely one 5-cycle then it contains $(p-2)$ 3-cycles and case *i*) holds. Now, assume there is no 5-cycle. Since $n = 3p + 2$, there exist at least two 4-cycles. If, by contradiction, there are three C^1, C^2 and C^3 then there must exist a 4th 4-cycle C^4 (because $n = 3p + 2$). Again the R3CC associated with $V(C^1) \cup V(C^2)$, $V(C^3) \cup V(C^4)$ and their complement would be violated. Then the partition contains precisely two 4-cycles and $(p-2)$ 3-cycles, showing that *ii*) holds.

F Additional Computational Results

In this section we provide computational results extending those of Section 8.

F.1 Comparison of lower bounds

In Table 11 we provide an instance-wise comparison between the LP bound and the root node bound obtained with our formulation and algorithm $\mathcal{F}$ (defined in Section 8) and the LP bounds of the formulations presented in [2, 9]. We compare on benchmark instances considered in those works and having $58 \leq n \leq 100$. For each

lower bounding approach we report the lower bound value (column "Value") and the relative optimality gap (column "Gap (%)") computed with respect to the best known upper bound (column $UB^\star$) of the instances. In particular, lower values in this column "Gap (%)" indicate that the corresponding algorithm performs better than other algorithms in terms of lower bounds.

From the table it is apparent that, overall, the worse lower bound is the one corresponding to the LP bound of our formulation (column $\mathcal{L}(\mathcal{F})$). The latter yields good lower bounds only on instances with $n = 3p, 3p + 1$. However, when CPLEX cuts are allowed (column $\mathcal{R}(\mathcal{F})$, corresponding to the root node bound) we get more competitive performance, The same results are displayed in a more qualitative form in Figure 3a and Figure 3b of the main paper.

F.2 Comparison with Erdoğan *et al.* algorithm on small instances

In Table 12 we report a comparison of our algorithm $\mathcal{F}$ (defined in Section 8) with that of [9] on small benchmark instances, that is those having $21 \leq n \leq 52$. Both algorithms reach optimality in all considered tests. Table 12 reports the final upper- and lower-bound and the CPU time needed to obtain them; the last line of the table is the average CPU time computed on the whole set of tests.

As also explained in Section 8 our algorithm is 10 times faster than the algorithm of [9] in average, especially due to its effectiveness when $p = \lfloor n/3 \rfloor$.

Table 11: Comparison of lower bounds obtained from algorithm $\mathcal{F}$ with those reported in [2, 9].

				$\mathcal{L}(\mathcal{F})$		$\mathcal{R}(\mathcal{F})$		Bektaş *et al.* [2]		Erdoğan *et al.* [9]	
Instance	$p^\star$	p	$UB^\star$	Value	Gap (%)	Value	Gap (%)	Value	Gap	Value	Gap (%)
brazil58	12	5	21744	20978.00	3.52	21419.97	1.49	21001	3.42	21170.75	2.64
		8	21289	20896.00	1.85	21219.52	0.33	20904	1.81	21081.5	0.97
		11	21080	20896.50	0.87	21080.00	0.00	20902.4	0.84	21080	0.00
		14	21221	20922.16	1.41	21110.00	0.52	21023.2	0.93	21221	0.00
		19	22635	21683.86	4.20	22564.81	0.31	21631.7	4.43	22340.99	1.30
st70	12	7	638.22	630.18	1.26	634.32	0.61	631.417	1.07	633.29	0.77
		10	632.54	628.50	0.64	632.54	0.00	628.559	0.63	630.63	0.30
		14	630.9	628.50	0.38	630.90	0.00	628.708	0.35	630.9	0.00
		17	636.19	628.85	1.15	632.81	0.53	630.017	0.97	635.51	0.11
		23	694.49	652.17	6.09	666.89	3.97	648.67	6.60	664.05	4.38
eil76	4	7	542.95	540.73	0.41	542.40	0.10	541.493	0.27	542.73	0.04
		10	545.02	541.49	0.65	543.39	0.30	543.297	0.32	544.51	0.09
		15	552.15	543.15	1.63	545.20	1.26	547.805	0.79	549.16	0.54
		19	563.95	545.70	3.24	547.15	2.98	553.543	1.85	557.28	1.18
		25	601.71	559.50	7.01	572.08	4.92	572.847	4.80	587.99	2.28
pr76	8	7	101401.33	99016.25	2.35	101401.33	0.00	99028.9	2.34	101091.86	0.31
		10	101779.42	99147.01	2.59	101111.76	0.66	99263	2.47	101165.57	0.60
		15	103822.35	99396.01	4.26	101593.51	2.15	100667	3.04	102513.09	1.26
		19	104481.75	99757.49	4.52	102193.66	2.19	102559	1.84	104036.62	0.43
		25	110073.94	103999.29	5.52	107409.85	2.42	108023	1.86	108095.83	1.80
rat99	8	9	1209.09	1203.54	0.46	1208.44	0.05	1203.98	0.42	1208.05	0.09
		14	1224.1	1204.51	1.60	1209.03	1.23	1214.61	0.78	1216.87	0.59
		19	1245.16	1207.40	3.03	1212.71	2.61	1235.13	0.81	1235.64	0.76
		24	1273.23	1217.81	4.35	1223.43	3.91	1263.17	0.79	1257.16	1.26
		33	1373.37	1327.70	3.33	1355.93	1.27	1343.4	2.18	1325.59	3.48
kroA100	13	10	19900.87	19442.48	2.30	19671.99	1.15	19570.2	1.66	19556.81	1.73
		14	19637.52	19380.71	1.31	19589.29	0.25	19380.7	1.31	19568.28	0.35
		20	19868.64	19408.92	2.31	19658.80	1.06	19523.3	1.74	19777.63	0.46
		25	20279.51	19536.47	3.66	19841.17	2.16	19815.7	2.29	20226.3	0.26
		33	22303.23	20573.92	7.75	21537.38	3.43	20542.7	7.89	21445.62	3.85
kroB100	20	10	20823.12	20368.64	2.18	20663.42	0.77	20444.8	1.82	20495.53	1.57
		14	20762.88	20335.99	2.06	20656.41	0.51	20396.3	1.77	20495.1	1.29
		20	20660.05	20339.98	1.55	20660.05	0.00	20414	1.19	20475.75	0.89
		25	20786.92	20398.39	1.87	20682.21	0.50	20581.7	0.99	20737.89	0.24
		33	22923.42	21387.41	6.70	21765.76	5.05	21413.9	6.59	21992.6	4.06
kroC100	13	10	19923.3	19703.26	1.10	19898.68	0.12	19703.3	1.10	19841.12	0.41
		14	19938.84	19707.93	1.16	19867.11	0.36	19725.6	1.07	19855.85	0.42
		20	20135	19734.27	1.99	19922.20	1.06	19853.1	1.40	20013.27	0.60
		25	20427.96	19804.85	3.05	19976.21	2.21	20033.8	1.93	20305.76	0.60
		33	22465.73	20680.17	7.95	21558.71	4.04	20286	9.70	21371.77	4.87
kroD100	14	10	20270.57	19951.26	1.58	20270.57	0.00	19957	1.55	20226.34	0.22
		14	20267.23	19951.26	1.56	20267.23	0.00	19962.9	1.50	20200.88	0.33
		20	20457	19984.10	2.31	20299.03	0.77	20021.2	2.13	20352.89	0.51
		25	20671.19	20058.04	2.97	20378.07	1.42	20156.9	2.49	20575.72	0.46
		33	22238.56	20672.12	7.04	21362.41	3.94	20669.6	7.06	21533.97	3.17
kroE100	12	10	20766.43	20629.87	0.66	20766.43	0.00	20651	0.56	20766.43	0.00
		14	20777.69	20618.94	0.76	20764.91	0.06	20641	0.66	20760.78	0.08
		20	20937.39	20633.66	1.45	20829.81	0.51	20715	1.06	20924.83	0.06
		25	21174.94	20704.65	2.22	20908.98	1.26	20891.6	1.34	21110.75	0.30
		33	22782.98	21431.11	5.93	21895.20	3.90	21485.3	5.70	22157.75	2.74
rd100	14	10	7524.08	7338.36	2.47	7510.14	0.19	7338.77	2.46	7489.73	0.46
		14	7500.44	7336.96	2.18	7500.44	0.00	7336.96	2.18	7479.44	0.28
		20	7537.98	7337.44	2.66	7503.73	0.45	7354.05	2.44	7507.01	0.41
		25	7555.83	7351.47	2.70	7516.50	0.52	7419.3	1.81	7550.19	0.07
		33	8131.25	7624.92	6.23	7839.75	3.58	7670.45	5.67	7837.31	3.61

Table 12: Comparison of our algorithm $\mathcal{F}$ with that of [9] on the whole set of "small" benchmark instances.

			$\mathcal{F}$			Erdoğan *et al.* [9]		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
gr21	1	2	2773.00	2773.00	0.02	2773.00	2773.00	0.49
		3	2774.00	2774.00	0.02	2774.00	2774.00	0.34
		4	2757.00	2757.00	0.00	2757.00	2757.00	0.19
		5	2832.00	2832.00	0.04	2832.00	2832.00	0.46
		7	3043.00	3043.00	0.00	3043.00	3043.00	0.45
ulysses22	5	2	68.33	68.33	0.00	68.33	68.33	0.39
		3	66.43	66.43	0.01	66.43	66.43	0.38
		4	64.23	64.23	0.00	64.23	64.23	0.19
		5	63.08	63.08	0.00	63.08	63.08	0.16
		7	65.08	65.08	0.01	65.08	65.08	0.18
gr24	3	2	1238.00	1238.00	0.00	1238.00	1238.00	0.31
		3	1227.00	1227.00	0.00	1227.00	1227.00	0.25
		4	1227.00	1227.00	0.00	1227.00	1227.00	0.27
		6	1266.00	1266.00	0.23	1266.00	1266.00	0.51
		8	1317.00	1317.00	0.29	1317.00	1317.00	0.24
fri26	7	2	911.00	911.00	0.00	911.00	911.00	0.41
		3	903.00	903.00	0.01	903.00	903.00	0.31
		5	893.00	893.00	0.00	893.00	893.00	0.44
		6	886.00	886.00	0.00	886.00	886.00	0.37
		8	885.00	885.00	0.00	885.00	885.00	0.21
bayg29	3	2	1562.00	1562.00	0.00	1562.00	1562.00	0.56
		4	1549.00	1549.00	0.00	1549.00	1549.00	0.5
		5	1555.00	1555.00	0.01	1555.00	1555.00	0.53
		7	1618.00	1618.00	1.35	1618.00	1618.00	2.15
		9	1676.00	1676.00	2.44	1676.00	1676.00	1.73
swiss42	7	4	1232.00	1232.00	0.02	1232.00	1232.00	1.37
		6	1231.00	1231.00	0.06	1231.00	1231.00	1.7
		8	1231.00	1231.00	0.04	1231.00	1231.00	1.56
		10	1238.00	1238.00	0.07	1238.00	1238.00	2.02
		14	1292.00	1292.00	0.19	1292.00	1292.00	1.12
att48	5	4	31903.30	31903.30	0.01	31903.30	31903.30	3.73
		6	31836.12	31836.12	0.03	31836.12	31836.12	3.41
		9	32195.53	32195.53	0.33	32195.53	32195.53	3.99
		12	32742.91	32742.91	3.36	32742.91	32742.91	3.99
		16	37068.82	37068.82	1.28	37068.82	37068.82	285.9
gr48	6	4	4841.00	4841.00	0.08	4841.00	4841.00	2.82
		6	4805.00	4805.00	0.02	4805.00	4805.00	1.76
		9	4926.00	4926.00	6.30	4926.00	4926.00	13.7
		12	5011.00	5011.00	22.95	5011.00	5011.00	4.91
		16	5445.00	5445.00	1.28	5445.00	5445.00	24.25
hk48	6	4	11271.00	11271.00	0.10	11271.00	11271.00	3.48
		6	11197.00	11197.00	0.00	11197.00	11197.00	2.88
		9	11292.00	11292.00	0.32	11292.00	11292.00	3.05
		12	11450.00	11450.00	2.25	11450.00	11450.00	3.41
		16	12215.00	12215.00	0.19	12215.00	12215.00	10.04
eil51	3	5	422.32	422.32	0.08	422.32	422.32	4.58
		7	424.36	424.36	0.72	424.36	424.36	6.88
		10	432.49	432.49	15.70	432.49	432.49	41.32
		12	436.59	436.59	24.05	436.59	436.59	14.41
		17	473.98	473.98	1.48	473.98	473.98	50.96
berlin52	7	5	7182.23	7182.23	0.19	7182.23	7182.23	3.66
		7	7167.20	7167.20	0.00	7167.20	7167.20	2.57
		10	7206.70	7206.70	0.18	7206.70	7206.70	4.43
		13	7298.63	7298.63	1.93	7298.63	7298.63	4.68
		17	7800.77	7800.77	5.22	7800.77	7800.77	48.81
Average CPU					1.69			10.43

F.3 Results for instances with $n = 3p$ and $n = 3p + 1$

In this section we give the full table of results of our tests presented in Section 8.2. In Table 13 we consider the following four root node bounds:

- $\mathcal{R}(\mathcal{F})$: obtained from the full formulation $\mathcal{F}$, defined in the main paper
- $\mathcal{R}(\mathcal{C})$: obtained from $\mathcal{F}$ by not performing the explicit separation of QHC inequalities (6) associated with 4- and 5-cycles (although the generic separation of the QHC inequalities is maintained).
- $\mathcal{R}(\mathcal{Q})$: obtained from $\mathcal{F}$ by not performing the separation of RCCs (7). Note that, in this case, both the separations of generic QHC inequalities (6) and of the special cases associated with 4- and 5-cycles is performed.
- $\mathcal{R}(\mathcal{H})$: corresponding to the root node bound of formulation (1)–(4).

Results are reported respectively in columns $\mathcal{R}(\mathcal{F})$, $\mathcal{R}(\mathcal{C})$, $\mathcal{R}(\mathcal{Q})$ and $\mathcal{R}(\mathcal{H})$ of Table 13. In boldface we highlight the best results in the table.

Table 13: Comparison of root node bounds obtained from four different HpMP formulations.

Instance	$\mathcal{R}(\mathcal{F})$	$\mathcal{R}(\mathcal{C})$	$\mathcal{R}(\mathcal{Q})$	$\mathcal{R}(\mathcal{H})$
brazil58	**22564.81**	22496.51	22079.29	21277.38
st70	666.89	**668.20**	663.97	638.71
eil76	**572.08**	571.68	570.98	553.97
pr76	107409.85	**107854.21**	106734.76	103025.37
rat99	1355.93	**1358.41**	1358.09	1248.75
kroA100	**21537.38**	21458.08	21160.16	20159.31
kroB100	21765.76	**21925.54**	21492.64	20935.25
kroC100	**21558.71**	21527.56	21167.68	20084.28
kroD100	21362.41	**21371.25**	21214.67	20523.19
kroE100	21895.20	**21952.22**	21743.21	21069.04
rd100	**7839.75**	7823.80	7793.25	7558.55

Next, we consider the comparison of the four exact algorithms based on the formulations $\mathcal{A}$, $\mathcal{A}^+$, $\mathcal{Q}$ and $\mathcal{F}$, defined in the main paper. On top of the results already given in Section 8.2, in Table 14 we report for each algorithm the best upper- and lower-bounds found at termination (columns "UB" and "LB", respectively). As in Section 8.2, the last lines of Table 14 reports the number of instances solved by each algorithm, followed by the average values of CPU time and relative optimality gap obtained at termination; these two averages are computed on the subset of instances respectively solved and unsolved by all algorithms within the time limit.

Table 14: Instances with $n = 3p$. CPU times, relative optimality gap and best upper- and lower-bounds obtained upon termination by the algorithms based on $\mathcal{A}$, $\mathcal{A}^+$, $\mathcal{Q}$ and $\mathcal{F}$ (see main paper for definitions).

	$\mathcal{A}$			$\mathcal{A}^+$			$\mathcal{Q}$			$\mathcal{F}$		
Instance	UB	LB	CPU	UB	LB	CPU	UB	LB	CPU	UB	LB	CPU
gr21	3043.00	3043.00	0.00	3043.00	3043.00	0.00	3043.00	3043.00	0.01	3043.00	3043.00	0.00
gr24	1317.00	1317.00	0.01	1317.00	1317.00	0.01	1317.00	1317.00	0.01	1317.00	1317.00	0.49
dantzig42	796.00	796.00	0.75	796.00	796.00	0.53	796.00	796.00	0.55	796.00	796.00	0.70
swiss42	1292.00	1292.00	0.02	1292.00	1292.00	0.02	1292.00	1292.00	0.03	1292.00	1292.00	0.06
att48	37068.82	37068.82	6.52	37068.82	37068.82	**1.06**	37068.82	37068.82	5.93	37068.82	37068.82	1.27
gr48	5445.00	5445.00	0.84	5445.00	5445.00	0.84	5445.00	5445.00	0.62	5445.00	5445.00	0.83
hk48	12215.00	12215.00	0.12	12215.00	12215.00	0.18	12215.00	12215.00	0.19	12215.00	12215.00	0.19
eil51	473.98	473.98	1.23	473.98	473.98	1.73	473.98	473.98	1.53	473.98	473.98	1.52
rat99	1373.37	1373.37	25.75	1373.37	1373.37	**13.43**	1373.37	1373.37	14.48	1373.37	1373.37	16.05
u159	47320.58	47320.58	489.10	47320.58	47320.58	**49.07**	47320.58	47320.58	443.08	47320.58	47320.58	129.24
brg180	123090.00	2739.07	97.77%	118130.00	2740.51	97.68%	130570.00	2738.65	97.90%	88790.00	2739.35	96.91%
rat195	2678.97	2678.97	**1314.25**	2678.97	2678.97	2633.75	2678.97	2678.97	1972.04	2678.97	2678.97	1591.79
d198	22641.76	13708.29	39.46%	16772.92	16653.22	0.71%	19073.20	14152.06	25.80%	17046.55	16615.36	2.53%
Instances solved			11/13			11/13			11/13			11/13
Average CPU			167.14			245.51			221.68			158.38
Average Opt. Gap			68.62%			49.20%			61.85%			49.72%

A similar table is constructed using the results of the algorithms based on $\mathcal{A}^+$, $\mathcal{Q}$ and $\mathcal{F}$ on the instances having $n = 3p + 1$. The results are reported in Table 15. The only difference with respect to Table 14 is that here we compute the average CPU time and relative optimality gap on the set of instances respectively solved and unsolved by both $\mathcal{A}^+$ and $\mathcal{F}$ versions within the time limit (since the algorithm relying on $\mathcal{Q}$ only solves 3 instances, while both those using $\mathcal{A}^+$ and $\mathcal{F}$ solve 7 instances).

Table 15: Instances with $n = 3p + 1$. CPU times, relative optimality gap and best upper- and lower-bounds obtained upon termination by the algorithms based on $\mathcal{A}^+$, $\mathcal{Q}$ and $\mathcal{F}$ (see main paper for definitions).

	$\mathcal{Q}$			$\mathcal{A}^+$			$\mathcal{F}$		
Instance	UB	LB	CPU	UB	LB	CPU	UB	LB	CPU
brazil58	22635.00	22635.00	5.8	22635.00	22635.00	**0.94**	22635.00	22635.00	**0.73**
st70	694.49	694.49	832.68	694.49	694.49	247.32	694.49	694.49	**190.58**
eil76	619.44	588.98	4.92%	610.56	594.63	**2.61%**	613.72	594.63	3.11%
pr76	110073.94	110073.94	122.8	110073.94	110073.94	**17.62**	110073.94	110073.94	38.44
kroA100	22411.17	22002.76	1.82%	22303.23	22303.23	**2112.57**	22303.23	22303.23	2457.73
kroB100	23606.71	22379.07	5.20%	22923.42	22722.79	**0.88%**	23679.36	22624.06	4.46%
kroC100	23829.68	21908.79	8.06%	23141.65	22331.75	3.50%	22465.73	22465.73	**2906**
kroD100	22554.05	21877.82	3.00%	22248.32	22047.70	**0.90%**	22402.88	22039.55	1.62%
kroE100	22782.98	22692.38	0.40%	22782.98	22782.98	**880.03**	22782.98	22782.98	1431.66
rd100	8198.57	8048.44	1.83%	8131.25	8131.25	**2164.67**	8149.65	8126.14	0.29%
pr124	57852.33	57201.93	1.12%	57672.38	57672.38	**66.79**	57672.38	57672.38	220.31
pr226	208085.23	64812.90	68.85%	156843.48	74302.02	**52.63%**	240399.60	72873.28	69.69%
gil262	4482.12	2406.85	46.30%	3274.99	2439.54	**25.51%**	3597.48	2426.11	32.56%
rd400	31864.65	15548.53	**51.20%**	34236.43	15553.04	54.57%	31933.63	15577.00	**51.22%**
Instances solved			3/14			7/14			7/14
Average CPU ($\mathcal{A}^+, \mathcal{F}$)			2539.75			554.21			723.24
Average Opt. Gap ($\mathcal{A}^+, \mathcal{F}$)			35.30%			27.24%			32.21%

F.4 Impact of QHC and restricted 2-opt inequalities

In this section, we provide the full-detailed comparison between algorithms $\mathcal{F}$ and $\mathcal{H}$ and algorithms $\mathcal{F}$ and $\mathcal{F}^-$, defined in Section 8.3.

The results of these two comparisons are reported in Table 16 and Table 17 respectively. Both tables are constructed in the same way indicating the best upper- and lower- bounds obtained by each algorithm (columns UB and LB) and the CPU time needed by each algorithm to reach optimality (column CPU); when a HpMP instance is unsolved within the time limit of 1 hour, we report the relative optimality gap obtained at the termination.

In each table, we use the boldface to indicate the HpMP instances solved by one algorithm and not by the other within the time limit. As explained in Section 8.3, there is only 1 instance solved by $\mathcal{H}$ and not by $\mathcal{F}$ while there are 15 HpMP instances solved by $\mathcal{F}$ and not by $\mathcal{H}$; all these latter have $n = 3p, 3p+1$, showing the usefulness of the QHC inequalities associated to 4- and 5-cycles. The restricted 2-opt inequalities have a more limited impact, as shown by the fact that just 5 instances are solved only by $\mathcal{F}$ and not by $\mathcal{F}^-$.

CPU times marked with an asterisk indicate that the corresponding algorithm reached optimality in less than half of the time needed by the competing algorithm. From Table 16 we see that this happens only 1 time for $\mathcal{H}$ (on instance `dantzig42` with $p = 10$) and 26 times for $\mathcal{F}$ (mostly on instances with $n = 3p, 3p+1$); from Table 17 we see that this happens 8 times for $\mathcal{F}^-$ and 19 times for $\mathcal{F}$.

Finally, the last line of each table contains the average CPU time of each algorithm computed by considering also unsolved HpMP instances (for which we use a CPU time value of 3600 seconds). Both tables show that $\mathcal{F}$ outperforms the other two variants, being ~ 200 and ~ 120 seconds faster on average than $\mathcal{H}$ and $\mathcal{F}^-$, respectively.

Table 16: Detailed comparison of the performance of algorithms $\mathcal{H}$ and $\mathcal{F}$, see Section 8.3: best upper- and lower-bounds obtained by the algorithms and CPU times to reach optimality (relative optimality gap for instances not solved within 1 CPU hour).

			$\mathcal{H}$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
`gr21`	1	2	2773.00	2773.00	0.03	2773.00	2773.00	0.02
		3	2774.00	2774.00	0.04	2774.00	2774.00	0.02
		4	2757.00	2757.00	0.00	2757.00	2757.00	0.00
		5	2832.00	2832.00	0.06	2832.00	2832.00	0.04
		7	3043.00	3043.00	0.58	3043.00	3043.00	0.00*
`ulysses22`	5	2	68.33	68.33	0.01	68.33	68.33	0.00*
		3	66.43	66.43	0.01	66.43	66.43	0.01
		4	64.23	64.23	0.00	64.23	64.23	0.00
		5	63.08	63.08	0.00	63.08	63.08	0.00
		7	65.08	65.08	0.08	65.08	65.08	0.01*
`gr24`	3	2	1238.00	1238.00	0.00	1238.00	1238.00	0.00
		3	1227.00	1227.00	0.00	1227.00	1227.00	0.00
		4	1227.00	1227.00	0.00	1227.00	1227.00	0.00
		6	1266.00	1266.00	0.27	1266.00	1266.00	0.23
		8	1317.00	1317.00	1.04	1317.00	1317.00	0.29*

continued on next page

			$\mathcal{H}$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
fri26	7	2	911.00	911.00	0.00	911.00	911.00	0.00
		3	903.00	903.00	0.01	903.00	903.00	0.01
		5	893.00	893.00	0.00	893.00	893.00	0.00
		6	886.00	886.00	0.00	886.00	886.00	0.00
		8	885.00	885.00	0.00	885.00	885.00	0.00
bayg29	3	2	1562.00	1562.00	0.00	1562.00	1562.00	0.00
		4	1549.00	1549.00	0.00	1549.00	1549.00	0.00
		5	1555.00	1555.00	0.02	1555.00	1555.00	0.01
		7	1618.00	1618.00	1.88	1618.00	1618.00	1.35
		9	1676.00	1676.00	10.68	1676.00	1676.00	2.44*
dantzig42	8	4	648.00	648.00	0.01	648.00	648.00	0.01
		6	647.00	647.00	0.01	647.00	647.00	0.01
		8	646.00	646.00	0.00	646.00	646.00	0.00
		10	654.00	654.00	0.20*	654.00	654.00	0.55
		14	797.00	784.80	1.53%	796.00	796.00	**0.62***
swiss42	7	4	1232.00	1232.00	0.02	1232.00	1232.00	0.02
		6	1231.00	1231.00	0.07	1231.00	1231.00	0.06
		8	1231.00	1231.00	0.31	1231.00	1231.00	0.04*
		10	1238.00	1238.00	0.50	1238.00	1238.00	0.07*
		14	1292.00	1292.00	1.68	1292.00	1292.00	0.19*
att48	5	4	31903.30	31903.30	0.01	31903.30	31903.30	0.01
		6	31836.12	31836.12	0.03	31836.12	31836.12	0.03
		9	32195.53	32195.53	0.46	32195.53	32195.53	0.33
		12	32742.91	32742.91	6.25	32742.91	32742.91	3.36
		16	38004.35	35956.02	5.39%	37068.82	37068.82	**1.28***
gr48	6	4	4841.00	4841.00	0.08	4841.00	4841.00	0.08
		6	4805.00	4805.00	0.02	4805.00	4805.00	0.02
		9	4926.00	4926.00	7.86	4926.00	4926.00	6.30
		12	5011.00	5011.00	21.68	5011.00	5011.00	22.95
		16	5469.00	5332.85	2.49%	5445.00	5445.00	**1.28***
hk48	6	4	11271.00	11271.00	0.10	11271.00	11271.00	0.10
		6	11197.00	11197.00	0.00	11197.00	11197.00	0.00
		9	11292.00	11292.00	0.36	11292.00	11292.00	0.32
		12	11450.00	11450.00	2.80	11450.00	11450.00	2.25
		16	12215.00	12215.00	478.22	12215.00	12215.00	0.19*
eil51	3	5	422.32	422.32	0.12	422.32	422.32	0.08
		7	424.36	424.36	0.46	424.36	424.36	0.72
		10	432.49	432.49	16.23	432.49	432.49	15.70
		12	436.59	436.59	37.04	436.59	436.59	24.05
		17	475.56	462.64	2.72%	473.98	473.98	**1.48***

continued on next page

			$\mathcal{H}$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
berlin52	7	5	7182.23	7182.23	0.19	7182.23	7182.23	0.19
		7	7167.20	7167.20	0.01	7167.20	7167.20	0.00*
		10	7206.70	7206.70	0.17	7206.70	7206.70	0.18
		13	7298.63	7298.63	2.48	7298.63	7298.63	1.93
		17	7800.77	7725.20	0.97%	7800.77	7800.77	**5.22***
brazil58	12	5	21744.00	21744.00	1.06	21744.00	21744.00	0.73
		8	21289.00	21289.00	0.14	21289.00	21289.00	0.14
		11	21080.00	21080.00	0.01	21080.00	21080.00	0.01
		14	21221.00	21221.00	1.74	21221.00	21221.00	0.99
		19	22649.00	22597.68	0.23%	22635.00	22635.00	**0.76***
st70	12	7	638.22	638.22	0.45	638.22	638.22	0.41
		10	632.54	632.54	0.15	632.54	632.54	0.14
		14	630.90	630.90	0.07	630.90	630.90	0.08
		17	636.19	636.19	0.52	636.19	636.19	0.71
		23	695.54	682.98	1.81%	694.49	694.49	**192.41***
eil76	4	7	542.95	542.95	0.48	542.95	542.95	0.30
		10	545.02	545.02	1.63	545.02	545.02	2.14
		15	552.15	552.15	89.65	552.15	552.15	89.30
		19	566.24	560.26	1.06%	571.05	560.36	1.87%
		25	629.91	571.42	9.28%	613.72	594.59	3.12%
pr76	8	7	101401.33	101401.33	0.14	101401.33	101401.33	0.14
		10	101779.42	101779.42	7.62	101779.42	101779.42	8.85
		15	103663.31	103663.31	301.38	103663.31	103663.31	224.94
		19	104481.75	104481.75	277.29	104481.75	104481.75	287.79
		25	110073.94	108805.51	1.15%	110073.94	110073.94	**38.56***
rat99	8	9	1209.09	1209.09	0.83	1209.09	1209.09	0.72
		14	1224.10	1224.10	456.73	1224.10	1224.10	387.77
		19	1257.05	1239.68	1.38%	1263.26	1239.29	1.90%
		24	1327.10	1251.40	5.70%	1348.19	1250.53	7.24%
		33	1635.48	1290.65	21.08%	1373.37	1373.37	**16.11***
kroA100	13	10	19900.87	19900.87	3.49	19900.87	19900.87	3.29
		14	19637.52	19637.52	0.72	19637.52	19637.52	0.74
		20	19868.64	19868.64	5.99	19868.64	19868.64	5.46
		25	20279.51	20279.51	228.19	20279.51	20279.51	435.69
		33	23610.92	21507.89	8.91%	22303.23	22303.23	**2504.77**
kroB100	20	10	20823.12	20823.12	3.13	20823.12	20823.12	2.84
		14	20762.88	20762.88	1.55	20762.88	20762.88	1.67
		20	20660.05	20660.05	0.27	20660.05	20660.05	0.39
		25	20786.92	20786.92	2.23	20786.92	20786.92	3.12
		33	25145.99	21919.69	12.83%	23679.36	22624.32	4.46%

continued on next page

			$\mathcal{H}$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
kroC100	13	10	19923.30	19923.30	2.36	19923.30	19923.30	2.35
		14	19938.84	19938.84	8.94	19938.84	19938.84	6.58
		20	20135.00	20135.00	27.87	20135.00	20135.00	23.12
		25	20427.96	20427.96	526.17	20427.96	20427.96	543.10
		33	27482.73	21456.99	21.93%	22465.73	22465.73	**2942.86**
kroD100	14	10	20270.57	20270.57	0.14	20270.57	20270.57	0.14
		14	20267.23	20267.23	0.08	20267.23	20267.23	0.08
		20	20457.00	20457.00	12.09	20457.00	20457.00	14.88
		25	20671.19	20671.19	134.82	20671.19	20671.19	155.73
		33	23781.09	21476.72	9.69%	22402.88	22039.19	1.62%
kroE100	12	10	20766.43	20766.43	0.09	20766.43	20766.43	0.09
		14	20777.69	20777.69	0.37	20777.69	20777.69	0.42
		20	20937.39	20937.39	4.60	20937.39	20937.39	4.11
		25	21174.94	21174.94	51.71	21174.94	21174.94	38.75
		33	24005.54	22023.75	8.26%	22782.98	22782.98	**1429.21***
rd100	14	10	7524.08	7524.08	2.73	7524.08	7524.08	2.66
		14	7500.44	7500.44	0.14	7500.44	7500.44	0.15
		20	7537.98	7537.98	20.03	7537.98	7537.98	13.76
		25	7555.83	7555.83	10.27	7555.83	7555.83	8.43
		33	8655.13	7835.20	9.47%	8149.65	8125.84	0.29%
pr124	23	12	52479.47	52479.47	49.82	52479.47	52479.47	32.64
		17	52210.02	52210.02	364.29	52210.02	52210.02	255.94
		24	51487.56	51487.56	1.30	51487.56	51487.56	2.22
		31	51830.28	51830.28	11.04	51830.28	51830.28	5.26*
		41	76304.28	56143.68	26.42%	57672.38	57672.38	**221.92***
u159	20	15	41238.46	41238.46	2.83	41238.46	41238.46	2.77
		22	41208.78	41208.78	21.49	41208.78	41208.78	16.54
		31	41805.27	41805.27	**2592.02**	41849.60	41789.29	0.14%
		39	43542.80	42195.83	3.09%	43377.73	42189.09	2.74%
		53	88242.71	43892.96	50.26%	47320.58	47320.58	**129.42***
brg180	15	18	10910.00	1860.00	82.95%	10910.00	1852.79	83.02%
		25	106920.00	1920.84	98.20%	99410.00	1923.10	98.07%
		36	230610.00	2048.27	99.11%	190040.00	2049.62	98.92%
		45	285850.00	2117.94	99.26%	312280.00	2090.00	99.33%
		60	341710.00	2250.00	99.34%	88790.00	2739.34	96.91%
rat195	7	19	2326.51	2326.51	3522.47	2326.51	2326.51	2978.33
		27	2461.25	2337.61	5.02%	2493.27	2337.33	6.25%
		39	2984.12	2361.83	20.85%	3034.79	2360.93	22.20%
		48	3626.13	2389.50	34.10%	3524.14	2388.50	32.22%
		65	4612.05	2478.16	46.27%	2678.97	2678.97	**1595.81***

continued on next page

			$\mathcal{H}$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
d198	32	19	12092.19	11997.02	0.79%	12100.43	11997.56	0.85%
		28	11910.72	11910.72	1.61	11910.72	11910.72	1.45
		39	11927.47	11927.47	211.87	11927.47	11927.47	103.62*
		49	12424.48	12006.69	3.36%	12332.60	12005.21	2.65%
		66	30609.80	15887.78	48.10%	17046.55	16615.30	2.53%
pr226	43	22	60693.96	58012.96	4.42%	60693.96	58030.75	4.39%
		32	58033.26	57433.26	1.03%	58033.26	57433.26	1.03%
		45	57177.55	57177.55	0.82	57177.55	57177.55	0.95
		56	145316.96	57177.55	60.65%	108704.75	57177.55	47.40%
		75	223824.06	71888.63	67.88%	240399.60	72872.37	69.69%
gil262	30	26	2260.32	2260.32	27.10	2260.32	2260.32	26.86
		37	2263.31	2263.31	224.29	2263.31	2263.31	167.72
		52	2329.06	2277.13	2.23%	2283.96	2277.62	0.28%
		65	2752.05	2287.24	16.89%	2996.28	2284.07	23.77%
		87	5531.13	2347.68	57.56%	3597.48	2426.12	32.56%
rd400	51	40	14675.53	14675.53	39.29	14675.53	14675.53	39.96
		57	14684.15	14684.15	1033.38	14684.15	14684.15	496.84*
		80	17363.46	14717.95	15.24%	17587.96	14717.44	16.32%
		100	21442.39	14768.20	31.13%	24026.88	14765.27	38.55%
		133	38842.13	15108.42	61.10%	31933.63	15576.99	51.22%
Average CPU					1092.15			797.22

Table 17: Detailed comparison of the performance of algorithms $\mathcal{F}^-$ and $\mathcal{F}$ (see Section 8.3: best upper- and lower-bounds obtained by the algorithms and CPU times to reach optimality (relative optimality gap for instances not solved within 1 CPU hour).

			$\mathcal{F}^-$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
gr21	1	2	2773.00	2773.00	0.03	2773.00	2773.00	0.02
		3	2774.00	2774.00	0.02	2774.00	2774.00	0.02
		4	2757.00	2757.00	0.00	2757.00	2757.00	0.00
		5	2832.00	2832.00	0.04	2832.00	2832.00	0.04
		7	3043.00	3043.00	0.00	3043.00	3043.00	0.00
ulysses22	5	2	68.33	68.33	0.01	68.33	68.33	0.00*
		3	66.43	66.43	0.01	66.43	66.43	0.01
		4	64.23	64.23	0.00	64.23	64.23	0.00
		5	63.08	63.08	0.00	63.08	63.08	0.00
		7	65.08	65.08	0.01	65.08	65.08	0.01
gr24	3	2	1238.00	1238.00	0.00	1238.00	1238.00	0.00
		3	1227.00	1227.00	0.00	1227.00	1227.00	0.00
		4	1227.00	1227.00	0.00	1227.00	1227.00	0.00
		6	1266.00	1266.00	0.22	1266.00	1266.00	0.23
		8	1317.00	1317.00	0.41	1317.00	1317.00	0.29
fri26	7	2	911.00	911.00	0.00	911.00	911.00	0.00
		3	903.00	903.00	0.01	903.00	903.00	0.01
		5	893.00	893.00	0.01	893.00	893.00	0.00*
		6	886.00	886.00	0.00	886.00	886.00	0.00
		8	885.00	885.00	0.00	885.00	885.00	0.00
bayg29	3	2	1562.00	1562.00	0.00	1562.00	1562.00	0.00
		4	1549.00	1549.00	0.00	1549.00	1549.00	0.00
		5	1555.00	1555.00	0.01	1555.00	1555.00	0.01
		7	1618.00	1618.00	1.56	1618.00	1618.00	1.35
		9	1676.00	1676.00	3.09	1676.00	1676.00	2.44
dantzig42	8	4	648.00	648.00	0.01	648.00	648.00	0.01
		6	647.00	647.00	0.01	647.00	647.00	0.01
		8	646.00	646.00	0.00	646.00	646.00	0.00
		10	654.00	654.00	0.73	654.00	654.00	0.55
		14	796.00	796.00	0.50	796.00	796.00	0.62
swiss42	7	4	1232.00	1232.00	0.02	1232.00	1232.00	0.02
		6	1231.00	1231.00	0.05	1231.00	1231.00	0.06
		8	1231.00	1231.00	0.05	1231.00	1231.00	0.04
		10	1238.00	1238.00	0.07	1238.00	1238.00	0.07
		14	1292.00	1292.00	0.30	1292.00	1292.00	0.19
att48	5	4	31903.30	31903.30	0.01	31903.30	31903.30	0.01
		6	31836.12	31836.12	0.03	31836.12	31836.12	0.03

continued on next page

			$\mathcal{F}^-$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
		9	32195.53	32195.53	0.29	32195.53	32195.53	0.33
		12	32742.91	32742.91	4.90	32742.91	32742.91	3.36
		16	37068.82	37068.82	1.21	37068.82	37068.82	1.28
gr48	6	4	4841.00	4841.00	0.07	4841.00	4841.00	0.08
		6	4805.00	4805.00	0.02	4805.00	4805.00	0.02
		9	4926.00	4926.00	8.52	4926.00	4926.00	6.30
		12	5011.00	5011.00	47.08	5011.00	5011.00	22.95*
		16	5445.00	5445.00	1.14	5445.00	5445.00	1.28
hk48	6	4	11271.00	11271.00	0.08	11271.00	11271.00	0.10
		6	11197.00	11197.00	0.00	11197.00	11197.00	0.00
		9	11292.00	11292.00	0.30	11292.00	11292.00	0.32
		12	11450.00	11450.00	2.80	11450.00	11450.00	2.25
		16	12215.00	12215.00	0.10	12215.00	12215.00	0.19
eil51	3	5	422.32	422.32	0.08	422.32	422.32	0.08
		7	424.36	424.36	0.48	424.36	424.36	0.72
		10	432.49	432.49	17.22	432.49	432.49	15.70
		12	436.59	436.59	46.96	436.59	436.59	24.05
		17	473.98	473.98	1.47	473.98	473.98	1.48
berlin52	7	5	7182.23	7182.23	0.10	7182.23	7182.23	0.19
		7	7167.20	7167.20	0.01	7167.20	7167.20	0.00*
		10	7206.70	7206.70	0.26	7206.70	7206.70	0.18
		13	7298.63	7298.63	4.10	7298.63	7298.63	1.93*
		17	7800.77	7800.77	4.36	7800.77	7800.77	5.22
brazil58	12	5	21744.00	21744.00	1.09	21744.00	21744.00	0.73
		8	21289.00	21289.00	0.13	21289.00	21289.00	0.14
		11	21080.00	21080.00	0.01	21080.00	21080.00	0.01
		14	21221.00	21221.00	1.86	21221.00	21221.00	0.99
		19	22635.00	22635.00	0.56	22635.00	22635.00	0.76
st70	12	7	638.22	638.22	0.36	638.22	638.22	0.41
		10	632.54	632.54	0.14	632.54	632.54	0.14
		14	630.90	630.90	0.08	630.90	630.90	0.08
		17	636.19	636.19	0.50	636.19	636.19	0.71
		23	694.49	694.49	641.80	694.49	694.49	192.41*
eil76	4	7	542.95	542.95	0.60	542.95	542.95	0.30
		10	545.02	545.02	1.55	545.02	545.02	2.14
		15	552.15	552.15	139.25	552.15	552.15	89.30
		19	567.56	559.24	1.47%	571.05	560.36	1.87%
		25	606.83	593.84	2.14%	613.72	594.59	3.12%
pr76	8	7	101401.33	101401.33	0.10	101401.33	101401.33	0.14
		10	101779.42	101779.42	8.28	101779.42	101779.42	8.85

continued on next page

			$\mathcal{F}^-$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
		15	103663.31	103607.48	0.05%	103663.31	103663.31	**224.94***
		19	104481.75	104258.14	0.21%	104481.75	104481.75	**287.79***
		25	110073.94	110073.94	45.61	110073.94	110073.94	38.56
rat99	8	9	1209.09	1209.09	0.33*	1209.09	1209.09	0.72
		14	1224.10	1224.10	405.88	1224.10	1224.10	387.77
		19	1258.74	1236.25	1.79%	1263.26	1239.29	1.90%
		24	1339.47	1248.32	6.80%	1348.19	1250.53	7.24%
		33	1373.37	1373.37	16.72	1373.37	1373.37	16.11
kroA100	13	10	19900.87	19900.87	2.41	19900.87	19900.87	3.29
		14	19637.52	19637.52	1.49	19637.52	19637.52	0.74*
		20	19868.64	19868.64	43.94	19868.64	19868.64	5.46*
		25	20305.61	20204.73	0.50%	20279.51	20279.51	**435.69***
		33	22303.23	22303.23	1830.95	22303.23	22303.23	2504.77
kroB100	20	10	20823.12	20823.12	2.17	20823.12	20823.12	2.84
		14	20762.88	20762.88	0.92	20762.88	20762.88	1.67
		20	20660.05	20660.05	0.33	20660.05	20660.05	0.39
		25	20786.92	20786.92	4.26	20786.92	20786.92	3.12
		33	23363.77	22551.30	3.48%	23679.36	22624.32	4.46%
kroC100	13	10	19923.30	19923.30	0.55*	19923.30	19923.30	2.35
		14	19938.84	19938.84	6.84	19938.84	19938.84	6.58
		20	20135.00	20135.00	309.03	20135.00	20135.00	23.12*
		25	20450.95	20348.40	0.50%	20427.96	20427.96	**543.10***
		33	23489.28	22265.96	5.21%	22465.73	22465.73	**2942.86**
kroD100	14	10	20270.57	20270.57	0.13	20270.57	20270.57	0.14
		14	20267.23	20267.23	0.08	20267.23	20267.23	0.08
		20	20457.00	20457.00	248.82	20457.00	20457.00	14.88*
		25	20671.19	20670.97	3577.82	20671.19	20671.19	155.73*
		33	22550.84	21982.67	2.52%	22402.88	22039.19	1.62%
kroE100	12	10	20766.43	20766.43	0.08	20766.43	20766.43	0.09
		14	20777.69	20777.69	0.35	20777.69	20777.69	0.42
		20	20937.39	20937.39	9.67	20937.39	20937.39	4.11*
		25	21174.94	21174.94	357.32	21174.94	21174.94	38.75*
		33	22782.98	22782.98	1537.96	22782.98	22782.98	1429.21
rd100	14	10	7524.08	7524.08	0.56*	7524.08	7524.08	2.66
		14	7500.44	7500.44	0.10	7500.44	7500.44	0.15
		20	7537.98	7537.98	63.73	7537.98	7537.98	13.76*
		25	7555.83	7555.83	15.37	7555.83	7555.83	8.43
		33	8152.26	8094.33	0.71%	8149.65	8125.84	0.29%
pr124	23	12	52479.47	52479.47	18.75	52479.47	52479.47	32.64
		17	52210.02	52210.02	225.29	52210.02	52210.02	255.94

continued on next page

			$\mathcal{F}^-$			$\mathcal{F}$		
Instance	$p^\star$	p	UB	LB	CPU	UB	LB	CPU
		24	51487.56	51487.56	2.12	51487.56	51487.56	2.22
		31	51830.28	51830.28	34.68	51830.28	51830.28	5.26*
		41	57672.38	57672.38	58.55*	57672.38	57672.38	221.92
u159	20	15	41238.46	41238.46	0.54*	41238.46	41238.46	2.77
		22	41208.78	41208.78	4.01*	41208.78	41208.78	16.54
		31	42116.86	41715.78	0.95%	41849.60	41789.29	0.14%
		39	49046.51	42022.02	14.32%	43377.73	42189.09	2.74%
		53	47320.58	47320.58	70.03	47320.58	47320.58	129.42
brg180	15	18	10910.00	1852.91	83.02%	10910.00	1852.79	83.02%
		25	99410.00	1923.11	98.07%	99410.00	1923.10	98.07%
		36	190040.00	2049.67	98.92%	190040.00	2049.62	98.92%
		45	312280.00	2090.00	99.33%	312280.00	2090.00	99.33%
		60	88790.00	2739.36	96.91%	88790.00	2739.34	96.91%
rat195	7	19	2326.51	2326.51	3159.13	2326.51	2326.51	2978.33
		27	2503.00	2336.92	6.64%	2493.27	2337.33	6.25%
		39	2875.20	2359.16	17.95%	3034.79	2360.93	22.20%
		48	3508.39	2386.15	31.99%	3524.14	2388.50	32.22%
		65	2678.97	2678.97	2662.16	2678.97	2678.97	1595.81
d198	32	19	12168.77	12000.27	1.38%	12100.43	11997.56	0.85%
		28	11910.72	11910.72	1.27	11910.72	11910.72	1.45
		39	11927.47	11927.47	103.80	11927.47	11927.47	103.62
		49	12866.97	11999.62	6.74%	12332.60	12005.21	2.65%
		66	17268.55	16656.34	3.55%	17046.55	16615.30	2.53%
pr226	43	22	60693.96	58007.83	4.43%	60693.96	58030.75	4.39%
		32	58033.26	57433.26	1.03%	58033.26	57433.26	1.03%
		45	57177.55	57177.55	1.27	57177.55	57177.55	0.95
		56	108704.75	57177.55	47.40%	108704.75	57177.55	47.40%
		75	188842.01	66540.16	64.76%	240399.60	72872.37	69.69%
gil262	30	26	2260.32	2260.32	4.43*	2260.32	2260.32	26.86
		37	2263.31	2263.31	214.71	2263.31	2263.31	167.72
		52	2298.48	2273.91	1.07%	2283.96	2277.62	0.28%
		65	3317.88	2280.15	31.28%	2996.28	2284.07	23.77%
		87	3789.76	2442.95	35.54%	3597.48	2426.12	32.56%
rd400	51	40	14675.53	14675.53	17.46*	14675.53	14675.53	39.96
		57	14684.15	14684.15	447.53	14684.15	14684.15	496.84
		80	17150.78	14711.37	14.22%	17587.96	14717.44	16.32%
		100	26414.50	14751.99	44.15%	24026.88	14765.27	38.55%
		133	31783.48	15590.57	50.95%	31933.63	15576.99	51.22%
Average CPU					919.06			797.22