Node based compact formulations for the Hamiltonian p -median problem

Michele Barbato, Francisco Canas, Luís Gouveia, Pierre Pesneau

Node based compact formulations for the Hamiltonian p-Median Problem

Michele Barbato[1], **Francisco Canas**[2 *], **Luís Gouveia**[2], and **Pierre Pesneau**[3]

[1]Dipartimento di Informatica "Giovanni Degli Antoni", Università degli Studi di Milano, via Celoria 18, 20133, Milan, Italy

[2]Centro de Matemática, Aplicações Fundamentais e Investigação Operacional (CMAF-CIO), Faculdade de Ciências, Universidade de Lisboa, C6 - Piso 4, Lisboa, 1749-016, Portugal

[3]Univ. Bordeaux, CNRS, INRIA, Bordeaux INP, IMB, UMR 5251, F-33400 Talence, France

Abstract

In this paper, we introduce, study and analyse several classes of compact formulations for the symmetric Hamiltonian p-Median Problem (HpMP). Given a positive integer p and a weighted complete undirected graph $G = (V, E)$ with weights on the edges, the HpMP on G is to find a minimum weight set of p elementary cycles partitioning the vertices of G. The advantage of developing compact formulations is that they can be readily used in combination with off-the-shelf optimization software, unlike other types of formulations possibly involving the use of exponentially sized sets of variables or constraints. The main part of the paper focuses on compact formulations for eliminating solutions with less than p cycles. Such formulations are less well known and studied than formulations which prevent solutions with more than p cycles. The proposed formulations are based on a common motivation, that is, the formulations contain variables that assign labels to nodes, and prevent less than p cycles by stating that different depots must have different labels and that nodes in the same cycle must have the same label. We introduce and study aggregated formulations (which consider integer variables that represent the label of the node) and disaggregated formulations (which consider binary variables that assign each node to a given label). The aggregated models are new. The disaggregated formulations are not, although in all of them new enhancements have been included to make them more competitive with the aggregated models. The two main conclusions of this study are: *i*) in the context of compact formulations, it is worth looking at the models with integer node variables, which have a smaller size. Despite their weaker LP relaxation bounds, the fewer variables and constraints lead to faster integer resolution, especially when solving instances with more than 50 nodes; *ii*) the best of our compact models exhibit a performance that, overall, is comparable to that of the best methods known for the HpMP (including branch-and-cut algorithms), solving to optimality instances with up to 226 nodes within 1 hour. This corroborates our message that the knowledge of the inequalities for preventing less than p cycles is much less well understood.

Keywords: Combinatorial optimization; Integer linear programming; Polyhedral theory; Valid inequalities; Hamiltonian p-median problem; Location Routing

1 Introduction

In this paper, we propose, study and analyse a several classes of compact formulations for the symmetric Hamiltonian p-Median Problem (HpMP). Given a positive integer p and a weighted complete undirected graph $G = (V, E)$ with a weight d_{ij} associated to each edge $\{i, j\} \in E$, the HpMP on G is to find a minimum-weight partition in p elementary cycles of the vertices of G (the weight of the p cycles is the sum of the weights of the edges in those cycles). If $p = 1$, the HpMP is the classical symmetric Traveling Salesman Problem (TSP) showing that the HpMP is NP-hard if p is part of the input. If p is not fixed, the

*Corresponding author. E-mail: francisco.m.p.canas@gmail.com

HpMP reduces to the 2-factor problem, that is, the problem of determining a minimum weight 2-regular subgraph of a graph G, which is known to be solvable in polynomial time (see, for instance, Cornuejols and Pulleyblank [9]).

The HpMP was first introduced by Branco and Coelho [6] as an application to real-world problems covering, among others, "schools location, milk stations and depot location for different industrial and commercial purposes". The typical application is illustrated by the assignment of p guards to n objects, where it is assumed that each guard cycles among the assigned objects. A similar application is the assignment of p maintenance/inspection vehicles that need to maintain/inspect n machines. Besides its applications to routing problems, other applications of the HpMP and its variations have subsequently been found in cutting problems (see, for instance, Glaab [12]) and laser multi-scanners (see, for instance, Glaab and Pott [13]).

Two related variants of the HpMP have been studied and discussed in the literature. In one of the variants, 2-cycles (that is, cycles with two nodes) are not allowed. This variant is usually used in studies with formulations using binary edge variables. In the other variant, as in the original work by Branco and Coelho [6], 2-cycles are allowed. This variant is usually modelled with directed arc formulations although models using edge variables with values in $\{0, 1, 2\}$ might also be derived, following an approach similar to the one of Benavent and Martinez-Sykora [5] for a multi-depot routing problem.

This paper focuses on the use of compact formulations to solve the variant not allowing 2-cycles. We include, next, an overview focusing on exact modelling approaches for this variant (we refer to Bektaş *et al.* [3] for references on the variant allowing 2-cycles and also on directed formulations, and to Barbato and Gouveia [1] for a more complete survey involving other approaches). This variant has been modelled using only edge variables by Hupp and Liers [16]. Gollowitzer *et al.* [15] presents three ILP formulations for the HpMP which are compared from a theoretical point of view as well as from a computational point of view. All formulations described in this reference involve edge variables and variables assigning nodes to cycles and are enhanced by symmetry-breaking inequalities. In fact, one such formulation will be mentioned in Section 3.2.2 of this paper. The best results in terms of polyhedral comparison and computational results are those based on the formulation preventing more than p cycles by using an exponential sized set of so-called "partition" inequalities.

Gollowitzer *et al.* [14] propose valid inequalities defined on the set of edge variables, introduce new formulations and present an extensive comparison of the LP relaxations of many formulations known from the literature, including formulations using only edge variables with other formulations using additional variables. This paper also produces the first computational study with formulations using only edge variables. Erdoğan *et al.* [11] propose an enhanced formulation of one given by Gollowitzer *et al.* [14] and a branch-and-cut algorithm based on the new formulation to solve the HpMP. The branch-and-cut algorithm outperforms the algorithms previously presented by Gollowitzer *et al.*[14].

A different approach was presented and tested by Marzouk *et al.* [17]. The authors present a branch-and-price algorithm based on the set-partitioning formulation given by Gollowitzer *et al.* [14]. The B&P algorithm solves instances with up to 318 vertices and outperforms the method of Gollowitzer *et al.* [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 same algorithm when $p < p^\star$.

A branch-and-cut algorithm is presented by Bektaş *et al.* [3] for the two variants. The authors present an extended directed formulation incorporating a set of multi-cut inequalities used in an earlier paper by the same authors for a multi-depot routing problem [2]. The algorithm 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 compares well with those of Erdoğan *et al.* [11] and Marzouk *et al.* [17].

More recently, Barbato and Gouveia [1] have introduced two new families of valid inequalities (for preventing solutions with less than p cycles) for a formulation of the problem in the space of natural edge variables. Sufficient conditions for inequalities in both families to define facets of the associated polytope are also given. A branch-and-cut algorithm based on these families of inequalities is developed. Instances with up to 400 vertices are solved, well comparing with exact methods that are state-of-the-art in the literature.

Content and Contribution of the Paper. Formulations that are polynomial in the number of variables and constraints have the advantage that they can be readily used in combination with off-the-shelf optimization software, unlike other types of formulations possibly involving the use of exponentially sized sets of variables or constraints. While typically requiring less computational time, algorithms that are based on such formulations require the use of specialized methods, such as constraint separation, which may not always be easy to understand, implement or use. Moreover, compact formulations are convenient to solve hard combinatorial problems appearing in subroutines of more complex algorithms (*e.g.*, separation problems in branch-and-cut algorithms).

As pointed out by Gollowitzer *et al.* [14], formulations for this problem can be viewed as composed by three sets of constraints: *i*) the degree constraints on the nodes; *ii*) one set that, together with the first, prevents more than p cycles; *iii*) one set that, together with the first, prevents less than p cycles. More recently, Bektaş *et al.* [3] contextualized this framework in terms of directed models using node variables y_i that indicate whether node i plays the role of a "depot" in a cycle in the solution together with the constraint $\sum_{i \in V} y_i = p$. The main argument from Bektaş *et al.* [3] to justify the use of these variables in the formulations is that in solutions satisfying only the degree constraints and having more than p cycles, at least one of the cycles will have no depots assigned to it. Thus, the cycle(s) with no depots assigned to can be "eliminated" by inequalities similar to the standard sub-tour elimination constraints known from the TSP (or ATSP). Alternatively, in solutions satisfying only the degree constraints and having less than p cycles, at least one cycle will have more than one depot assigned to it. Thus, the cycle(s) with more than one depot assigned to can be "eliminated" by inequalities similar to the "path-elimination" inequalities known from models of variants with multi-depots.

With respect to inequalities of type *ii*), the literature on compact formulations for the ATSP is vast (see, for instance, Roberti and Toth [20] and Öncan *et al.* [18]) and thus, there are plenty of possibilities. However, one observation about this adaptation is needed, namely that the adapted constraints will need to include the y_i variables. In this paper, we study two classes of models for modelling the generic constraints *ii*). We use an adaptation of the Desrochers and Laporte (DL) model [10] and an adaptation of the well known multicommodity flow (MCF) model by Claus [8]. The first model is known to have a weak linear programming (LP) relaxation bound but has a rather good performance to obtain the optimal integer solution when used within a ILP package. The second model has a LP relaxation bound that is considerably stronger than the one given by the DL model. However, due to the large number of variables and constraints, it is less efficient to solve. The reader is referred to Roberti and Toth [20] for the evaluation of the performance of these two models, and other models, to obtain the optimal integer solution when used within a ILP package, for the case of the ATSP. The reason these two models are included in this study is to have an idea of how the LP relaxation bounds for the whole model change when formulations for modelling part *ii*) change. Although we do not expect the MCF based model to be competitive, a (substantial) improvement of the reported LP relaxation bounds might suggest equivalent and alternative solution methods that might later be explored.

The main part of the paper focuses on compact formulations for part *iii*). The main reason is that inequalities which prevent solutions with less than p cycles are less well known than inequalities which prevent solutions with more than p cycles. Every formulation presented in this work for part *iii*) stems from a common motivation, that is, the formulations contain variables that assign labels to nodes, and prevent less than p cycles by stating that different depots must have different labels and that nodes in the same cycle must have the same label. The formulations described and proposed in this study will be described in Section 3.

Due to the focus on inequalities for modelling part *iii*), we will also address a variant of the HpMP. This variant is motivated by some empirical observations taken from the literature, namely that the instances which are more difficult to solve are obtained with large values of p and that, in many cases, inequalities from part *ii*) are not needed in the models to solve instances with large values of p. Thus, to focus our study on inequalities of part *iii*), we also address the variant where the number of cycles is restricted to be greater than or equal to p. We denote this variant by HpMP$_{\geq}$ and observe that it is modelled by considering only the two subsets of constraints *i*) and *iii*).

Our study will start by addressing models for the HpMP$_{\geq}$, which will be differentiated by the constraints used for part *iii*). For the HpMP, the models will be obtained by considering the models for the

HpMP$_{\geq}$ either augmented with the adapted DL constraints, or the adapted MCF constraints, for part *ii*). The formulations will be compared in terms of LP relaxation bound values (both theoretically and computationally) and on the basis of computational time requirements for attaining optimal solutions for a number of benchmark instances. One of the main conclusions of this study is that, in the context of compact formulations, it is worth looking at the more compact models with the node variables. Despite the weaker LP relaxation bounds, the fewer variables and constraints lead to faster computational times, especially when solving instances with more than 50 nodes. This might result from the use of current ILP packages that benefit strongly from built-in enhancements. redcheck again One other conclusion is that, although not outperforming more sophisticated methods known from the literature, the results provided by our compact models are often comparable, especially if p is not very close to 1 or to $\lfloor \frac{n}{3} \rfloor$.

Section 2 describes a base model that will be used in most of the formulations studied in this paper. In Section 3, we address formulations for the HpMP$_{\geq}$, or alternatively, formulations for preventing less than p cycles. We also make a brief discussion on the comparison of the LP relaxations of the models of the different classes discussed before. In Section 4, we consider models for the HpMP that are obtained from models described in Section 3, by augmenting them either with the adapted DL constraints or the adapted MCF constraints, for part *ii*). We also compare these two approaches. In Section 5, we describe the computational experiments, including a description of the instances tested and an evaluation of the results obtained with the models discussed in the previous sections. A final section concludes the paper.

2 Formulations - The BASE model

In this paper we will consider several compact formulations for the HpMP and HpMP$_{\geq}$. They are expressed in extended spaces involving several sets of variables in addition to the edge variables describing the HpMP in the "natural" space. The *integer hull* of such a formulation is the polytope coinciding with the convex hull of its solutions. Projecting the integer hull of a formulation for the HpMP onto the subspace of edge variables we obtain the convex hull of the incidence vectors of the HpMP solutions. For ease of notation, from now on, we say that an inequality is *valid* for the HpMP if it is valid for one of the integer hulls of the formulations presented below. To simplify the notation in the pages that follow, we also denote by $\delta(i) = \{\{i,j\} \in E, \forall j \in V\}$ the set of all edges adjacent to node i. Finally, for a given formulation F, we refer to its LP relaxation as F$_L$, and to its optimal value as v(F).

In this section, we present a base formulation which is contained in most of the formulations for the HpMP that are presented in the pages that follow. This base formulation includes two sets of binary variables: *i*) variables u_{ij} will take value 1 if the edge $\{i,j\} \in E$ is included in one of the p cycles and 0 otherwise, and *ii*) variables y_j will indicate whether node $j \in V$ is a depot ($y_j = 1$) or not ($y_j = 0$) of the cycle node j is in. Consider, also, the following set of constraints:

$$\text{Min.} \quad \sum_{\{i,j\} \in E} d_{ij} u_{ij} \tag{1a}$$

$$\text{s.t.} \quad \sum_{\{i,j\} \in \delta(i)} u_{ij} = 2, \quad \forall i \in V \tag{1b}$$

$$\sum_{j \in V} y_j = p, \tag{1c}$$

$$u_{ij} \in \{0,1\}, \qquad \forall \{i,j\} \in E \tag{1d}$$

$$y_j \in \{0,1\}, \qquad \forall j \in V \tag{1e}$$

The objective function corresponds to minimizing the sum of the weights of the edges which define the p cycles. Equalities (1b) are the usual assignment constraints stating that each node is included in one and only one cycle. Constraints (1d) define the u_{ij} variables as binary. A solution to the formulation described by the u_{ij} variables alone is composed of several disjoint cycles (with at least three nodes each) covering all nodes of the graph. Equality (1c) indicates the number of nodes that play the role of depots of the cycles. Finally, constraints (1e) define the y_j variables as binary. We denote by "BASE" the formulation described above. Observe that for the moment there are no inequalities relating the two different sets of

variables. In the following sections we describe several sets of such constraints, including sets of constraints that prevent solutions from having less than p cycles and sets of constraints that prevent solutions from having more than p cycles. Besides the two sets of variables previously defined, in some of the models to be presented later on, other sets of variables will be used to define the new sets of constraints. Also, in one class of formulations the additional variables will make unnecessary the use of these depot variables.

There are many equivalent solutions on the u_{ij} variables that differ on the choice of the y_i variable to denote the depot of each cycle. In order to reduce the number of equivalent solutions, most of the formulations we study in this work will include inequalities that enforce the following symmetry breaking (SB) strategy: a node can be a depot only if it is the node with the lowest index in the cycle it belongs to. In fact, for some of the formulations presented in this work, it is far from clear how to write valid formulations with the same sets of variables without imposing this property. The SB strategy allows us to tighten the redundant inequality $u_{ij} \leq 1$ into:

$$u_{ij} + y_j \leq 1, \quad \forall \{i,j\} \in E : i < j \tag{2}$$

Proposition 1. *Constraints* (2) *are valid for the HpMP$_{\geq}$.*

Proof. Consider a pair of nodes i, j such that $i < j$. We show that we cannot have both $y_j = 1$ and $u_{ij} = 1$. If $y_j = 1$, then, j must be the node with the lowest index in the cycle it belongs to. Therefore, since $i < j$, nodes i and j cannot be in the same cycle, and thus, these two nodes cannot be adjacent (that is, $u_{ij} = 0$). □

The previous inequality will be added to most of the formulations presented later on. The computational results will also show that the addition of these inequalities is relevant for the computational times.

3 Preventing less than p cycles

As mentioned in the introduction, all the formulations presented in this paper for preventing solutions with less than p cycles are based on a similar motivation: *i*) assigning different labels to the depots and *ii*) guaranteeing that two consecutive nodes in a cycle have the same label. We will use the designation "continuity constraints" for constraints guaranteeing this last condition. These formulations can be viewed in two different ways. The first one depends on the type of variables used to represent the label of the node, either by using integer variables to represent the label of a node (aggregated formulations) or binary variables that assign each node to a given label (disaggregated formulations). The second view depends on the meaning of the label. It can either be the index of the node serving as a depot for a cycle (node-depot assignment) or the index, between 1 and p, of the cycle (node-cycle assignment). The aggregated models are new. The disaggregated formulations are not, although in all of them new enhancements have been included in the models. Since we are modelling the HpMP$_{\geq}$, we may have feasible solutions with more than p cycles. In such situations, we can have nodes from different cycles with the same labels. These situations will be detailed in the proofs of the validity of the models introduced in the next sections.

3.1 Aggregated formulations

In this subsection, we discuss two classes of aggregated models that include integer variables indicating the value of the label of a node and constraints guaranteeing that nodes in the same cycle must have the same label.

We start by presenting a relaxed formulation, denoted by Rel, that can be viewed as a base to build the models presented in this subsection. The formulation Rel is built from the BASE formulation by adding the generic label integer variables z_i, $\forall i \in V$ and the following constraints:

$$z_i \leq z_j + (i-1)(1-u_{ij}), \quad \forall \{i,j\} \in E : i < j \tag{3a}$$
$$z_j \leq z_i + (j-1)(1-u_{ij}), \quad \forall \{i,j\} \in E : i < j \tag{3b}$$
$$1 \leq z_i \leq i \text{ and integer}, \qquad \forall i \in V \tag{3c}$$

Constraints (3a) and (3b) guarantee $z_i = z_j$ for any pair of adjacent nodes i and j. The Rel formulation relaxes the condition i) stated at the introduction of Section 3, namely that depots must have different labels. Thus, it allows infeasible solutions - for instance, for any value of p, any solution featuring a single Hamiltonian cycle is feasible, since $z_i = 1$, $\forall i \in V$ is feasible.

The two classes differ in the interpretation of the label of a node and are obtained by considering different sets of constraints linking the label variables z_i with the depot variables y_i.

3.1.1 Node-depot assignment formulations

In this section we consider the label of a node as the index of the node serving as depot for the cycle. More precisely, we consider the integer variables $s_i \in \mathbb{N}$, $\forall i \in V$, which indicate the index of the depot of the cycle containing node i (more precisely, if node i belongs to a cycle with depot d, then, $s_i = d$). Models using a similar set of variables have been proposed by Burger *et al.* [7] (and later by Bektaş *et al.* [4]) for a multi-depot routing problem. In this section, we show how to adapt these models for the HpMP$_{\geq}$ by adding depot variables. We also introduce a new (as far as we know) enhancement of the "continuity" inequalities which implies some of the constraints of the original model leading to a valid model with fewer constraints. Consider the following set of constraints:

$$s_i \leq s_j + (i-1)(1-u_{ij}), \quad \forall \{i,j\} \in E : i < j \tag{4a}$$
$$s_j \leq s_i + (j-1)(1-u_{ij}), \quad \forall \{i,j\} \in E : i < j \tag{4b}$$
$$s_i \geq (i-1)y_i + 1, \quad \forall i \in V \tag{4c}$$
$$1 \leq s_i \leq i \text{ and integer}, \quad \forall i \in V \tag{4d}$$

Constraints (4a), (4b) and (4d) are constraints (3a), (3b) and (3c) after setting $s_i = z_i$, $\forall i \in V$. Constraints (4c) and (4d) guarantee that $s_i = i$ if node i is a depot (this follows from considering $y_i = 1$ in constraint (4c) for node i). When $y_i = 0$, these two constraints guarantee that $1 \leq s_i \leq i$. This is consistent with the SB strategy mentioned earlier, as it implies only the node with the lowest index in a cycle can be its depot. Observe that although the label variables s_i are interpreted as integer, they can be defined as continuous. This also applies to all the formulations described in this subsection.

We denote by ANDA (for "Aggregated Node to Depot Assignment") the formulation which results from augmenting BASE with constraints (4a)-(4d). Observe that since this formulation is an augmentation of Rel, we have that $v(\text{ANDA}_L) \geq v(\text{Rel}_L)$. Additionally, our computational results show that there exist instances for which $v(\text{ANDA}_L) > v(\text{Rel}_L)$.

We observe that in the two cited works on the multi-depot routing problem, more elaborate upper bounding constraints on the s_i variables were also necessary for the validity of the model. In the case of the HpMP$_{\geq}$, similar constraints, such as $s_i \leq i - 1 + y_i$, $\forall i \in V$, could have been added to the model. However, as shown in the next result, these constraints are not needed to obtain a valid model for the HpMP$_{\geq}$ (this follows from the SB strategy imposing that $s_i \leq i$). Also, although not redundant in the LP relaxation of the model, these constraints only contribute for very small improvements of the LP relaxation bounds. For this reason, they were not included in the ANDA model and will not be further considered in the remainder of the text.

Proposition 2. *The formulation ANDA is valid for the HpMP$_{\geq}$.*

Proof. Consider a feasible solution for the HpMP$_{\geq}$ and consider an assignment of values, 0 or 1, to the u_{ij} variables corresponding to this solution. It is easy to choose an assignment of values for the remaining two sets of variables that satisfy (4a)-(4d). To see this, consider the cycles in the feasible solution to be sorted by ascending order of the indices of their nodes with lowest indices. Considering d_k the node with the lowest index in cycle k such that $1 \leq k \leq p$, set $y_{d_k} = 1$. For every node i, set $s_i = d_k$ if node i is in some cycle k such that $1 \leq k \leq p$, and set $s_i = 1$ if node i is in some cycle k such that $k > p$. The resulting solution is feasible for the ANDA model.

To see the converse (that is, that any partition of the graph in less than p cycles violates some of these constraints), observe that a feasible solution for BASE is composed of k disjoint cycles that cover all nodes of the graph. Constraints (4a) and (4b) imply that

$$s_{i_j} = s_{i_l}, \forall j, l \in \{1, \ldots, m\}$$

for the nodes in any cycle $C = \{i_1, \ldots, i_m\}$ of this solution.

Assume that $k < p$. In such a solution, there will be at least one cycle with at least two depots (this follows from (1c)), e.g., nodes a and b such that $a < b$. As mentioned before, $s_a = s_b$ must hold. However, since nodes a and b are depots, we also have that $y_a = y_b = 1$. Constraints (4c) and (4d) imply $s_b = b > a = s_a$, leading to $s_b > s_a$. But this contradicts what was observed before, namely that $s_a = s_b$. Thus, any solution with less than p cycles is not feasible for the ANDA model. □

The validity of constraints (2) raises the question of knowing whether the term $(1 - u_{ij})$ in constraints (4a) and/or constraints (4b) can be replaced by $(1 - u_{ij} - y_j)$. In fact, while it is not apparent how to enhance constraints (4b), we can provide the following stronger inequality in the case of (4a):

$$s_i \leq s_j + (i-1)(1-u_{ij}) - (j-1)y_j, \quad \forall \{i,j\} \in E : i < j \tag{4e}$$

The next result proves the validity of constraints (4e) and that they are a strengthened version of (4a). The result also states that in the presence of constraints (4d), constraints (4c) are a particular case of (4e):

Proposition 3. *Constraints* (4e) *are valid for the HpMP$_{\geq}$ and:*

- *Constraints* $y_j \geq 0$, $\forall j \in V$ *and* (4e) *imply* (4a)*;*
- *Constraints* (4d) *and* (4e) *imply* (4c)*.*

Proof. To prove the validity of constraints (4e), consider some edge $\{i,j\} \in E$. We assume $y_j = 1$ (otherwise the inequality would correspond to the original valid inequality). The SB strategy implies $u_{ij} = 0$ if $y_j = 1$ and $i < j$, and constraints (4d) and (4c) imply $s_j = j$ if $y_j = 1$. Therefore, the corresponding constraint (4e) becomes $s_i \leq i$, which is implied by constraints (4d). Thus, constraints (4e) are valid for the HpMP$_{\geq}$.

Regarding the two implications, the first is trivial. To prove the second result, consider the constraint (4e) for a given node j and node $i = 1$. Since $s_1 = 1$ (which is implied by constraints (4d)), the resulting constraint becomes (4c) for the same node j. □

We denote by SANDA the formulation which results from replacing constraints (4a) and (4c) in ANDA with constraints (4e). Proposition 3 leads to:

Proposition 4. *The formulation SANDA is valid for the HpMP$_{\geq}$ and* $v(SANDA_L) \geq v(ANDA_L)$*.*

Finally, we also study the effect of adding constraints (2) to the previously defined models. We denote by ANDA$^+$ and SANDA$^+$ the models which result from augmenting the models ANDA and SANDA, respectively, with these constraints. The following proposition is an obvious consequence of the definition of these models:

Proposition 5.

- $v(ANDA^+{}_L) \geq v(ANDA_L)$*;*
- $v(SANDA^+{}_L) \geq v(SANDA_L)$*.*

The computational results reported in Section 5 indicate that the addition of constraints (2) also seems to have a greater impact on the LP relaxation bounds and on the computational times than the addition of constraints (4e). The main conclusion is that the most competitive model from this class of models is the SANDA$^+$ model.

3.1.2 Node-cycle assignment formulations

In this section, we consider the label of a node as the index of the cycle that node is in. More precisely, we consider the integer variables $v_i \in \mathbb{N}$, $\forall i \in V$ such that, if node i belongs to the k-th cycle, then,

$v_i = k$, $\forall k = 1, \ldots, p$. As noted before, to the best of our knowledge, formulations using such variables are new. Consider the following set of constraints:

$$v_i \leq v_j + (i-1)(1-u_{ij}), \qquad \forall \{i,j\} \in E : i < j \tag{5a}$$

$$v_j \leq v_i + (j-1)(1-u_{ij}), \qquad \forall \{i,j\} \in E : i < j \tag{5b}$$

$$v_i \geq \sum_{j=1}^{i} y_j - (i-2)(1-y_i), \quad \forall i \in V \tag{5c}$$

$$v_i \leq \sum_{j=1}^{i} y_j, \qquad \forall i \in V \tag{5d}$$

$$1 \leq v_i \leq i \text{ and integer}, \qquad \forall i \in V \tag{5e}$$

Constraints (5a), (5b) and (5e) are constraints (3a), (3b) and (3c) after setting $v_i = z_i$, $\forall i \in V$. Constraints (5c) and (5d) relate the v_i variables with the y_i variables. When $y_i = 0$, constraints (5c) are redundant. When $y_i = 1$, these constraints, together with (5d), guarantee that $v_i = \sum_{j=1}^{i} y_j$, that is, the cycle number of a given node i equals the number of depot nodes with an index less than or equal to i. Constraints (5e) set lower and upper bounds on the v_i variables. Observe that constraints (1c) and (5d) also imply that $v_i \leq p$, $\forall i \in V$, which is consistent with the definition of the v_i variables given before. The equality $v_i = \sum_{j=1}^{i} y_j$ when $y_i = 1$ is consistent with the assumption that the binary node variables y_i are interpreted as indicating whether node i is the depot of a cycle *and* is the node of smallest index in the cycle. Observe also that constraints (5c) and (5d) enforce an additional SB strategy, namely that the cycles must be sorted by ascending order of the indices of their depots. Finally, although the new variables v_i are defined as integer, they can be defined as continuous without altering the validity of the model. This applies to all the formulations described in this subsection.

We denote by WANCA (for "Weak Aggregated Node to Cycle Assignment") the formulation which results from augmenting BASE with constraints (5a)-(5e).

Proposition 6. *The formulation WANCA is valid for the HpMP$_{\geq}$.*

Proof. Consider a feasible solution for the HpMP$_{\geq}$ and consider an assignment of values, 0 or 1, to the u_{ij} variables corresponding to this solution. It is easy to choose an assignment of values for the remaining two sets of variables that satisfy (5a)-(5e). To see this, consider the cycles in the feasible solution to be sorted by ascending order of the indices of their nodes with lowest indices. Let d_k be the node with the lowest index in cycle k, $1 \leq k \leq p$, and set $y_{d_k} = 1$. For every node i, set $v_i = k$ if node i is in some cycle k, $1 \leq k \leq p$. In the case that the solution has more than p cycles, set, for instance, $v_i = 1$ if node i is in some cycle k such that $k > p$. The resulting solution is feasible for the WANCA model.

To see the converse (that is, that any partition of the graph in less than p cycles violates some of these constraints), observe that a feasible solution for BASE is composed of k disjoint cycles that cover all nodes of the graph. We begin by observing that for any pair of adjacent nodes i and j, constraints (5a) and (5b) imply $v_i = v_j$, which, in turn, implies that

$$v_{i_k} = v_{i_l}, \forall k, l \in \{1, \ldots, m\}$$

for the nodes in any cycle $C = \{i_1, \ldots, i_m\}$ of this solution.

Assume that $k < p$. In such a solution, there will be at least one cycle with at least two depots (this follows from (1c)), e.g., nodes a and b such that $a < b$. As mentioned before, $v_a = v_b$ must hold. However, since nodes a and b are depots, we also have that $y_a = y_b = 1$. Constraints (5c) and (5d) imply $v_b = \sum_{j=1}^{b} y_j \geq 1 + \sum_{j=1}^{a} y_j = 1 + v_a$ leading to $v_b \geq v_a + 1$. But this contradicts what was observed before, namely that $v_a = v_b$. Thus, some of constraints (5a)-(5e) are violated by any solution featuring less than p cycles. □

In contrast to what happens with the ANDA model, the LP relaxation bound of the WANCA model does not improve on the LP relaxation bound of the original BASE model.

Proposition 7. $v(BASE_L) = v(WANCA_L)$.

Proof. As WANCA is an augmentation of BASE, $v(\text{WANCA}_L) \geq v(\text{BASE}_L)$ clearly holds. Therefore, it is only necessary to prove $v(\text{WANCA}_L) \leq v(\text{BASE}_L)$. This can be proven by showing that for any feasible solution for BASE_L, there is also a feasible solution for WANCA_L with the same cost. Consider a feasible solution (u^b, y^b) for BASE_L. Observe that for the remainder of the proof, the value of the variables y^b are irrelevant. Consider now a solution (u^w, y^w, v^w) for WANCA_L such that $u^w = u^b$, $y_1 = 1$, $y_2 = 0$, $y_i = 0.5$, $\forall i \in V : 3 \leq i \leq 2p$, $y_i = 0$, $\forall i \in V : i > 2p$ and $v_i = 1$, $\forall i \in V$. It is easy to see that this solution is feasible for WANCA_L and has the same cost as the original solution. □

This result shows that the linking constraints (5c) and (5d) are effective only on the integer solutions. A more detailed comparison between the two classes of aggregated models can be found in Section 3.1.3.

As mentioned earlier, constraints (1c) and (5d) imply $v_i \leq p$, $\forall i \in V$. These upper bounds indicate that the constraints (5a), (5b), (5c) and (5e) can be "enhanced", as follows:

$$v_i \leq v_j + \min\{p-1, i-1\}(1-u_{ij}), \qquad \forall \{i,j\} \in E : i < j \tag{6a}$$

$$v_j \leq v_i + \min\{p-1, j-1\}(1-u_{ij}), \qquad \forall \{i,j\} \in E : i < j \tag{6b}$$

$$v_i \geq \sum_{j=1}^{i} y_j - \min\{p-1, i-2\}(1-y_i), \quad \forall i \in V \tag{6c}$$

$$1 \leq v_i \leq \min\{i, p\} \text{ and integer}, \qquad \forall i \in V \tag{6d}$$

Constraints (6a), (6b), (6c) and (6d) are enhanced versions of (and can replace) constraints (5a), (5b), (5c) and (5e), respectively. We denote by ANCA (for "Aggregated Node to Cycle Assignment") the formulation which results from augmenting the BASE formulation with constraints (5d) and (6a)-(6d), and omit the proof of the validity of ANCA as it is very similar to that of the validity of WANCA.

Proposition 8. *The formulation ANCA is valid for the* $\text{HpMP}_{\geq}$.

The enhancement obtained by using the "min" coefficients in every constraint as well as by restricting the value of the variables would suggest that the ANCA model would lead to promising results with respect to LP relaxation bounds, at least when compared with WANCA. Quite surprisingly, there are no improvements for the benchmark instances considered in the computational experiment. On the other hand, there are improvements for other randomly generated instances. This, combined with the fact that ANCA is a strengthening of WANCA, proves that $v(\text{ANCA}_L) \geq v(\text{WANCA}_L)$ and that there exist instances for which the inequality is strict.

The validity of constraints (2) also leads to a similar question made in the context of the ANDA model, namely of knowing whether constraints (6a) and/or (6b) can be enhanced. And once again, while it is not apparent how to enhance constraints (6b), we can present an enhanced version of constraints (6a):

$$v_i \leq v_j + \min\{p-1, i-1\}(1-u_{ij}) - \min\{p, i\}y_j, \quad \forall \{i,j\} \in E : i < j \tag{6e}$$

Proposition 9. *Constraints* (6e) *are valid for the* $\text{HpMP}_{\geq}$. *Constraints* $y_j \geq 0$, $\forall j \in V$ *and* (6e) *imply* (6a).

Proof. To prove the validity of constraints (6e), consider a pair of nodes i, j such that $i < j$ and we assume that $y_j = 1$ (otherwise the inequality corresponds to the original valid inequality). It has already been shown (in the proof of Proposition 1) that $y_j = 1$ implies $u_{ij} = 0$ if $i < j$. Therefore, the corresponding constraint (6e) becomes $v_i \leq v_j - 1$. But this inequality is implied by constraints (6c) and (5d) which state that $v_j = \sum_{l=1}^{j} y_l \geq \sum_{l=1}^{i} y_l + 1 \geq v_i + 1$ when $y_j = 1$, proving these constraints are valid. The proof of the implication is trivial. □

We observe, however, that constraints (6a) cannot be "enhanced" by using the coefficient of variable y_j from SANDA (i.e, adding the term "$-\min\{p, j-1\}y_j$" to the previous constraints). The reason for this is that the value of the coefficient $(j-1)$ is too high, conflicting with constraints (6c) and/or (5d), leading to a not valid inequality. To see this, consider $i < j - 1 \leq p$. Then, the "inequality" becomes $v_i \leq v_j + (i-1)(1-u_{ij}) - (j-1)y_j$. Consider also that j is a depot, that is, $y_j = 1$. In this case, due

to constraints (6c) and (5d), we have that $v_j = \sum_{k=1}^{j} y_k$. Finally, observe that $u_{ij} = 0$ since $i < j$ and j is a depot. Then the "inequality" reduces to $v_i + (j - i) \leq \sum_{k=1}^{j} y_k$ which is not necessarily valid when $1 < j - i$. Observe, however, that when $i = j - 1$, the inequality is valid, since we obtain the corresponding constraint (6e).

We denote by SANCA ("S" for "Strengthened") the formulation which results from replacing constraints (6a) in ANCA with constraints (6e). Proposition (9) leads to:

Proposition 10. *The formulation SANCA is valid for the HpMP$_{\geq}$ and $v(SANCA_L) \geq v(ANCA_L)$.*

The two models described in this subsection can also be enhanced by adding constraints (2) as was the case of the ANDA and SANDA models. We denote by ANCA$^+$ and SANCA$^+$ the models which result from augmenting ANCA and SANCA respectively with constraints (2). The following proposition is an obvious consequence of the definition of these models:

Proposition 11.
- $v(ANCA^+{}_L) \geq v(ANCA_L)$;
- $v(SANCA^+{}_L) \geq v(SANCA_L)$.

The computational results reported in Section 5 indicate that the addition of constraints (2) and (6e) has a positive impact on the LP relaxation bounds and on the computational times obtained with these models, and unlike what happened with the aggregated node-depot models, the impact of augmenting the ANCA model with constraints (6e) is comparable to that of augmenting the same model with constraints (2). As a result of this, the main conclusion to be drawn from the computational testing of these models is that formulation SANCA$^+$ is the most competitive aggregated node-cycle assignment model. However, regarding computational times, the computational results indicate that the aggregated node-depot formulations are more competitive and we try to provide an explanation for this behaviour in the next paragraphs.

To conclude this section on aggregated node-cycle assignment formulations, we observe that we can obtain a more compact and still valid model (although with a weaker LP relaxation) by removing constraints (6c) and (5d) from the SANCA model. Although more compact, the fact that it has a weaker LP relaxation may explain why, according to some preliminary computational testing, this model is not as competitive as the SANCA model for solving instances of the HpMP$_{\geq}$. Thus, we omit details and the proof of its validity in this paper.

3.1.3 Comparing the Aggregated Node-Depot and Node-Cycle Assignment formulations

In this section we emphasize the differences between the aggregated node-depot and node-cycle assignment formulations, which can explain the computational results reported in Section 5. We begin the comparison with models ANDA and WANCA. The ANDA and WANCA models differ only in the constraints linking the label variables with the depot variables. As pointed before, the set of linking constraints used in the node-cycle model does not lead to an improvement with respect to the LP relaxation of the BASE model.

With respect to comparing the ANCA model with the ANDA model, although ANDA sometimes provides slightly better LP relaxation bounds, the differences are minimal. In fact, we did not find any instance where ANCA has a strictly better LP relaxation bound than ANDA, although, in our computational experiments, we have instances for which the model ANDA augmented with constraints (5d) and (6a)-(6d) of the ANCA model resulted in LP relaxation bounds which were strictly better than those obtained with the original ANDA model. This shows that the LP relaxation of the ANDA model does not dominate that of the ANCA model. Also, for obtaining the optimal solutions, the CPU times produced by ANDA are, in general, better than the CPU times produced by ANCA, as reported in Section 5. To explain this behaviour, we recall one property of each class of models, namely that for the ANDA model we have that $s_i = i$ if node i is a depot, and with respect to the ANCA (and also the WANCA) model, we have that $v_i = \sum_{j=1}^{i} y_j$ if node i is a depot. One immediate consequence of this is that in the node-depot models, the label of a node depends only on the choice of the depot of the cycle while, in node-cycle models, the label depends on the number of depots previously chosen. This might explain the different behaviour of the two classes of models for solving instances of the problem. In particular, in a branch-and-bound algorithm, when a y_i variable is fixed to 1, the value of the corresponding v_i variable in the node-cycle models might

still remain dependent on other y_j variables for which the values have not been fixed, whereas the value of the corresponding s_i variable in the node-depot models always becomes a constant (since $y_i = 1$ implies $s_i = i$). In other words, the benefits of using constraints (4c) seem to outweigh the negatives of not being able to lower the coefficients on constraints (3a) and (3b).

With respect to comparing the model SANDA with the model SANCA, we begin by observing that one of the main differences between these models is in the lifted constraints (4e) and (6e). Consider, again, the "generic" label variables z_i, $\forall i \in V$ introduced at the beginning of this section and the relaxed lifted constraint:

$$z_i \leq z_j + (i-1)(1-u_{ij}) - iy_j, \quad \forall \{i,j\} \in E : i < j \tag{7}$$

The lifted inequality from the SANDA model is obtained from (7) by setting $s_i = z_i$, $\forall i \in V$ and adding "$-(j-i-1)y_j$" to the right-hand side of constraints (7), whereas the lifted inequality from the SANCA model can be obtained from (7) by setting $v_i = z_i$, $\forall i \in V$ and lowering the coefficients in constraints (7) to obtain (6e). This argument, where we start with a relaxed inequality and provide different enhancements to obtain two different valid inequalities, suggests that the strengthened ND model and the strengthened NC models have non-related LP relaxations (as confirmed by the computational results). The results also indicate that, empirically, we gain more by adding the term "$-(j-i)y_j$" to the right-hand side of (7) and not reducing the coefficients (and thus, being allowed to keep the original interpretation of the s_i variables) than by doing the coefficient reduction.

3.2 Disaggregated formulations

In this section we discuss two classes of disaggregated models that include binary node variables that assign nodes to cycles. The two classes differ in the interpretation of the node assignment variables. The models discussed in the two next subsections are built from models known from the literature. We emphasize, however, that in both cases a reduced and valid version of the main model are proposed for the first time. The reduced models contain half of the "continuity" constraints of the main models, and while the removal of these constraints results in slightly lower LP relaxation bounds, the lower number of constraints means these models result in lower computational times when compared with their "complete" counterparts. In spite of this, however, these reduced models are still not competitive with the aggregated models presented in Section 3.1. In one class of the models, a new set of inequalities is also proposed for the first time. The new inequalities lead to substantial improvements on the LP relaxation bounds as well as substantial reductions on the CPU times to obtain the optimal solution.

3.2.1 Node-depot assignment formulations

In this section we return to the interpretation of Section 3.1.1 and consider the label of a node as the index of the node serving as depot for the cycle. More precisely, we consider the binary node-depot assignment variables ${s_i}^d$, $\forall i, d \in V : d \leq i$, indicating if node d is the depot of the cycle node i is in. Observe that these variables are defined only for $d \leq i$ due to the SB strategy, and that variables ${s_d}^d$ indicate if node d is a depot. The models discussed in this subsection is built upon models known from the literature (see Gollowitzer *et al.* [14], Erdoğan *et al.* [11] and Bektaş *et al.* [3], this last reference describes a model for the directed case). As with the ANDA models, for solutions with more than p cycles, we will have nodes from more than one cycle assigned to the same depot d since the constraints of the model will limit the number of depots to p. This will be clarified in the proof of Proposition 12. The following system prevents

less than p cycles:

$$\sum_{d \in V: d \leq i} {s_i}^d = 1, \qquad \forall i \in V \tag{8a}$$

$${s_i}^d \leq {s_d}^d \qquad \forall i, d \in V : d \leq i \tag{8b}$$

$${s_i}^d \leq {s_j}^d + 1 - u_{ij}, \quad \forall \{i,j\} \in E : i < j, \forall d \in V : d \leq i \tag{8c}$$

$${s_j}^d \leq {s_i}^d + 1 - u_{ij}, \quad \forall \{i,j\} \in E : i < j, \forall d \in V : d \leq i \tag{8d}$$

$${s_i}^i = y_i, \qquad \forall i \in V \tag{8e}$$

$${s_i}^d \in \{0,1\} \qquad \forall i, d \in V : d \leq i \tag{8f}$$

Constraints (8a) indicate that any node i must be assigned to exactly one depot and (8b) indicates that if node j is in the cycle with node d as the depot, then, node d must be a depot. Constraints (8c) and (8d) are similar to the continuity inequalities presented in the other models and indicate that two adjacent nodes must be assigned to the same depot. Constraints (8e) relate the node-depot assignment variables with the depot variables defined in the BASE model. Constraints (8f) define the new variables as binary - however, it is easy to see that they can be defined as continuous without altering the validity of the model.

Erdoğan *et al.* [11] present an exponentially sized generalization of the continuity constraints (8c) and (8d) which consider subsets of depots, instead of single node sets, and according to the results in Erdoğan *et al.* [11], these constraints appear to be effective in solving instances of the problem. However, since the focus of this paper is on compact formulations, we do not elaborate further on these generalized constraints.

We denote by DNDA (for "Disaggregated Node-Depot Assignment") the formulation which results from augmenting BASE with constraints (8a)-(8f). Although the proof of the validity of the DNDA model for the HpMP$_{\geq}$ can be taken and adapted from the literature, we present a proof here, since it allows an easier proof of the reduced model.

Proposition 12. *The formulation DNDA is valid for the HpMP$_{\geq}$.*

Proof. Consider a feasible solution for the HpMP$_{\geq}$ and consider an assignment of values, 0 or 1, to the u_{ij} variables corresponding to this solution. It is easy to choose an assignment of values for the remaining two sets of variables that satisfy (8a)-(8f). To see this, consider the cycles in the feasible solution to be sorted by ascending order of the indices of their nodes with lowest indices. Considering d_k the node with the lowest index in cycle k such that $1 \leq k \leq p$, set ${s_{d_k}}^{d_k} = 1$. For every node i, set ${s_i}^{d_k} = 1$ if node i is in some cycle k such that $1 \leq k \leq p$, and for all the nodes i in some cycle k such that $k > p$ we set, for instance, ${s_i}^1 = 1$. The resulting solution is feasible for the DNDA model.

To see that partitions of the graph in less than p cycles are not feasible, we consider a solution with less than p cycles, and show it is not feasible for the aforementioned model. We begin by observing that for any pair of adjacent nodes i and j and any depot d, constraints (8c) and (8d) imply ${s_i}^d = {s_j}^d$. This, in turn, means that for any cycle with nodes $\{i_1, \ldots, i_m\}$, constraints (8c) and (8d) imply the following:

$${s_{i_r}}^d = {s_{i_l}}^d, \forall r, l \in \{1, \ldots, m\}, \forall d \in V \tag{8g}$$

To see that this model prevents solutions with less than p cycles, start by considering a solution with less than p cycles. Under this assumption, constraints (1c) and (8e) imply one of the cycles will include at least two depots - for instance, nodes a and b. This means ${s_a}^a = {s_b}^b = 1$. However, according to what was previously observed, as a consequence of ${s_a}^a = 1$, ${s_i}^a = 1$ for every node i in the cycle to which node a belongs must hold, including node b. However, if ${s_b}^a = {s_b}^b = 1$, constraint (8a) for node b is violated. Therefore, some of constraints (8a)-(8f) are violated by any solution featuring less than p cycles, and the proposition holds. □

We present next two classes of valid inequalities that strengthen the LP relaxation of the DNDA model. The first class was presented in [11] and the second is new. The following constraints state that if some node j is a depot, there must be two other nodes assigned to node j (in other words, each cycle must have at least three nodes):

$$\sum_{j \in V: j > i} {s_j}^i \geq 2{s_i}^i, \quad \forall i \in V \tag{8h}$$

As mentioned in [11], these constraints are relevant for the reported computational results. The second class is an enhancement of constraints (2) and a consequence of the SB strategy:

$$u_{ij} + \sum_{d=i+1}^{j} {s_j}^d \leq 1, \quad \forall \{i,j\} \in E : i < j \tag{8i}$$

These constraints state that if some edge $\{i,j\}$ belongs to one of the cycles, then, the depot of that cycle cannot be any node with an index greater than i.

Proposition 13. *Constraints* (8i) *are valid for the HpMP*$_{\geq}$.

Proof. We prove the validity of these constraints by showing that, for any pair of nodes i,j such that $i<j$, $\sum_{d=i+1}^{j} {s_j}^d = 1$ implies $u_{ij} = 0$. Assume that $\sum_{d=i+1}^{j} {s_j}^d = 1$. In this case, the depot of the cycle to which node j belongs is some node d such that $d > i$. Therefore, i cannot belong to that cycle, which implies that nodes i and j cannot be adjacent, and therefore, $u_{ij} = 0$ must hold. □

We denote by DNDA^{+} the model which results from augmenting DNDA with constraints (8h) and (8i). The previous propositions show that:

Proposition 14. *The formulation DNDA*$^{+}$ *is valid for the HpMP*$_{\geq}$ *and* $v(DNDA^{+}{}_L) \geq v(DNDA_L)$.

Our computational results will show that the addition of these constraints improves the LP relaxation bound of the DNDA model and that, in several cases, the solution times become substantially smaller. However, the disadvantage of the disaggregated models, namely of the "node-depot assignment" models, is the large number of variables and constraints, which makes it difficult for solvers to solve large instances using these models. In order to potentially improve the computational times obtained with this class of models, we show next that we can define a valid model, DNDA- (DNDA$^{+}_{-}$), which is obtained by removing constraints (8d) from DNDA (DNDA^{+}). We now show DNDA- and DNDA$^{+}_{-}$ are valid formulations for the HpMP$_{\geq}$:

Proposition 15. *The formulations DNDA- and DNDA*$^{+}_{-}$ *are valid for the HpMP*$_{\geq}$.

Proof. The proof is done only for the DNDA- model since a similar proof holds for the DNDA$^{+}_{-}$ model. We prove the validity of the DNDA- model by showing that for any cycle with nodes $\{i_1, \ldots, i_m\}$ in any feasible solution for this model, the equality (8g) still holds, as the rest of the proof would be the same as that of Proposition 12.

To prove that (8g) still holds, as in the proof of Proposition 12, we only need to prove the following equality holds for any pair of adjacent nodes i,j such that $i<j$:

$${s_i}^d = {s_j}^d, \forall d \in V : d \leq i,j$$

To prove the equality above, we consider a feasible solution for DNDA- and a pair of adjacent nodes i and j (we assume $i<j$). Since these nodes are adjacent, constraints (8c) imply ${s_i}^d \leq {s_j}^d$, $\forall d \in V : d \leq i,j$. Observe that constraints (8a) and (8f) also imply that there exists exactly one node, d', such that ${s_i}^{d'} = 1$. Since i and j are adjacent, constraints (8f) and (8c) imply that ${s_j}^{d'} = 1$. Constraints (8a) for nodes i and j imply $\sum_{a \in V : a \neq d'} {s_i}^a = \sum_{a \in V : a \neq d'} {s_j}^a = 0$, which, combined with $0 \leq {s_l}^a, \forall l,a \in V$, implies ${s_i}^a = {s_j}^a = 0$, $\forall a \in V : a \neq d'$. Therefore, ${s_i}^d = {s_j}^d$, $\forall d \in V : d \leq i,j$ holds. □

The definition of these models leads to the following result:

Proposition 16. $v(DNDA^{+}{}_L) \geq v(DNDA^{+}_{-}{}_L)$ *and* $v(DNDA^{+}_{-}{}_L) \geq v(DNDA\text{-}_L)$.

The results from our computational results indicate that in terms of computational times, the models DNDA$^{+}_{-}$ and DNDA^{+} are the more competitive models, showing that the addition of constraints (8h) and (8i) is relevant, independently of using the complete or the reduced model.

3.2.2 Node-cycle assignment formulations

Unlike any other model presented in this study, the models discussed in this subsection do not require the y_i variables to guarantee the requirement of the p cycles. Thus, they might be viewed as not following the paradigm of the three previous classes of models. The "complete" model discussed in this section is from [15] (see, also, [14]). However, the variant of the model using only half of the so-called "continuity" constraints is new. The model includes the binary variables v_i^k, $\forall i \in V$, $\forall k = 1, \ldots, p : k \leq i$, indicating whether node i is in cycle k. Observe that these variables are only defined for $k \leq i$, since, in order to reduce the number of equivalent solutions, we follow Gollowitzer *et al.* [15] and consider that a node i cannot be in a cycle k such that $i < k$. However, despite reducing the number of equivalent solutions, this strategy does not eliminate symmetries entirely, as, for some cycle $k \in \{1, \ldots, p-1\}$, the node with the lowest index of cycle k can have an index greater than the index of the node with the lowest index of cycle $k+1$. Similarly to what happened in the definition of the variables in the previous class of models, for solutions with more than p cycles, we will have more than one cycle assigned to the same index k. This will be clarified in the proof of Proposition 17. Consider the following model to prevent less than p cycles:

$$\sum_{k=1}^{\min\{i,p\}} v_i^k = 1, \qquad \forall i \in V \tag{9a}$$

$$v_i^k \leq v_j^k + 1 - u_{ij}, \quad \forall \{i,j\} \in E : i < j, \forall k = 1, \ldots, p : k \leq i \tag{9b}$$

$$v_j^k \leq v_i^k + 1 - u_{ij}, \quad \forall \{i,j\} \in E : i < j, \forall k = 1, \ldots, p : k \leq i \tag{9c}$$

$$\sum_{i \in V : i \geq k} v_i^k \geq 3, \qquad \forall k = 1, \ldots, p \tag{9d}$$

$$v_i^k \in \{0,1\}, \qquad \forall i \in V, \forall k = 1, \ldots, p \tag{9e}$$

Constraints (9a) guarantee that each node is assigned to exactly one cycle. Constraints (9b) and (9c) guarantee that two adjacent nodes are in the same cycle. Constraints (9d) guarantee that each cycle has at least three nodes. Observe that a right-hand side value of 1 would suffice to obtain a valid formulation. Indeed, our computational results show that the stronger right-hand side leads to improvements on the LP bounds - however, these modifications do not drastically affect the behaviour of these formulations when compared with the other models presented in this paper. These constraints have a similar meaning to constraints (8h) presented in Section 3.2.1 for the DNDA models. Finally, constraints (9e) define the v_i^k variables as binary. For the DNDA models, we have observed that the ${s_i}^d$ variables, $\forall i, d \in V : d \neq i$, can be defined as continuous. However, for the validity of the DNCA models, the v_i^k variables need to be defined as binary. As noted above, the previous model does not require the y_i variables to guarantee the requirement of the p cycles. This requirement is guaranteed by the range of the variation of the index k in the variables v_i^k.

We denote by DNCA (for "Disaggregated Node-Cycle Assignment") the formulation which results from augmenting BASE with constraints (9a)-(9e) and removing constraints (1c) and (1e) and the y_i variables. The proof of the validity of the DNCA model for the HpMP$_{\geq}$ can be adapted from the one given in Gollowitzer *et al.* [15]. However, as was the case of the DNDA model, we present a proof here, since it allows an easier proof of the reduced model.

Proposition 17. *The formulation DNCA is valid for the HpMP$_{\geq}$.*

Proof. Consider a feasible solution for the HpMP$_{\geq}$ and consider an assignment of values, 0 or 1, to the u_{ij} variables corresponding to this solution. It is easy to choose an assignment of values for the remaining two sets of variables that satisfy (9a)-(9e). To see this, consider the cycles in the feasible solution to be sorted by ascending order of the indices of their nodes with lowest indices. For every node i, set $v_i^k = 1$ if node i is in some cycle k such that $1 \leq k \leq p$, and for all the nodes i in some cycle k such that $k > p$ we set, for instance, $v_i^1 = 1$. The resulting solution is feasible for the DNCA model.

To prove the converse we begin by observing that for any pair of adjacent nodes i and j and any cycle k, constraints (9b) and (9c) imply $v_i^k = v_j^k$. This, in turn, means that for any cycle with nodes $\{i_1, \ldots, i_m\}$,

constraints (9b) and (9c) imply the following:

$$v_{i_r}^k = v_{i_l}^k,\ \forall r, l \in \{1, \ldots, m\},\ \forall k = 1, \ldots, p \tag{9f}$$

We consider, now, a solution, S, with fewer than p cycles, and show that it is not feasible for the DNCA model. Constraints (9d) imply that for each $k = 1, ..., p$ there exist some nodes i such that each $v_i^k = 1$. But since the index k ranges from 1 to p and the solution S has fewer than p cycles, at least one of these cycles must have one node, say i, such that $v_i^r = 1$ for a given r, and another node, say j, such that $v_i^s = 1$ for a given s, $r \neq s$. But this contradicts (9c) since these constraints imply that $s = r$. Therefore, some of constraints (9a)-(9e) are violated by any solution featuring less than p cycles, and the proposition holds. □

Similarly to what was done for model DNDA, we can also define a new model, DNCA-, which is obtained from DNCA by removing constraints (9c). We now show that DNCA- is a valid formulation for the HpMP$_{\geq}$:

Proposition 18. *The formulation DNCA- is valid for the HpMP$_{\geq}$.*

Proof. We prove the validity of the DNCA- model by showing that for any cycle with nodes $\{i_1, \ldots, i_m\}$ in any feasible solution for this model, the equality (9f) still holds, as the rest of the proof would be the same as that of Proposition 17.

To prove that (9f) still holds, as in the proof of Proposition 17, we only need to prove the following equality holds for any pair of adjacent nodes i, j such that $i < j$:

$$v_i^k = v_j^k,\ \forall k = 1, \ldots, p : k \leq i, j$$

To prove the equality above, we consider a feasible solution for DNCA- and a pair of adjacent nodes i and j (we assume $i < j$). Since these nodes are adjacent, constraints (9b) imply $v_i^k \leq v_j^k,\ \forall k = 1, \ldots, p : k \leq i, j$. Observe that constraints (9a) and (9e) also imply that there exists exactly one cycle index, k', such that $v_i^{k'} = 1$. Since i and j are adjacent, constraints (9e) and (9b) imply that $v_j^{k'} = 1$. Constraints (9a) for nodes i and j imply $\sum_{q=1:q\neq k'}^{\min\{i,p\}} v_i^q = \sum_{q=1:q\neq k'}^{\min\{j,p\}} v_j^q = 0$, which, combined with $0 \leq v_l^q,\ \forall l \in V, q = 1, \ldots, p$, implies $v_i^q = v_j^q = 0,\ \forall q = 1, \ldots, p : q \neq k$. Therefore, $v_i^k = v_j^k,\ \forall k = 1, \ldots, p \in V : k \leq i, j$ holds. □

The definition of the DNCA and DNCA- models results in the following:

Proposition 19. $v(DNCA_L) \geq v(DNCA\text{-}_L)$.

The computational results indicate that, although not considerably worse, the node-cycle models are not as good as the node-depot models. In terms of computational times, these results are interesting since the node-cycle models have much fewer variables than the node-depot models. We give an explanation in the computational results section.

3.3 Comparing aggregated formulations with disaggregated formulations

In this subsection we make a brief comparison on the behaviour of the models in each of the four classes just described. As pointed out (and confirmed in the computational results section) the aggregated models have worse LP relaxation bounds than the disaggregated models, but are strongly preferred for obtaining the optimal integer solutions. In order to give some insight in the relationship between the models, observe that the variables of the node-depot (node-cycle) aggregated models can be related with the variables of the node-depot (node-cycle) disaggregated models as follows:

$$s_i = \sum_{d=1}^{i} d s_i^d, \qquad \forall i \in V \tag{10}$$

$$v_i = \sum_{k=1}^{\min\{i,p\}} k v_i^k, \quad \forall i \in V \tag{11}$$

Observe that by using (10) (respectively (11)), we can add any constraint of the aggregated node-depot (respectively node-cycle) assignment models to the disaggregated node-depot (respectively node-cycle) models. We can, for instance, consider two augmented disaggregated models - the first is obtained from DNDA by adding constraints (4a) and (4b) (where the s_i variables are replaced according to (10)), and the second is obtained from DNCA by adding constraints (6a) and (6b) (where the v_i variables are replaced according to (11)). Some preliminary computational testing shows there are instances where these augmented disaggregated models have strictly better LP relaxation bounds than their non-augmented counterparts, showing some of the constraints found in the aggregated models are not implied by the disaggregated models. However, while, in terms of their LP relaxations, these augmented disaggregated models are always at least as good as the corresponding aggregated models, they are not competitive with their aggregated counterparts regarding computational times (due to having many more variables and constraints). A similar situation, although with better LP bounds, arises when the disaggregated models are augmented with the lifted constraints (4e) or (6e).

Finally, the relationship between the two sets of variables, aggregated and disaggregated, may also suggest new inequalities to enhance the LP relaxations of the aggregated models. For instance, in Sections 3.2.1 and 3.2.2, it was stated that the addition of constraints (8h) helps in speeding up computational times and that the enhanced right-hand side in constraints (9d) results in improvements to the LP relaxation bounds. The following inequalities can be derived from those constraints:

$$\sum_{i \in V} s_i \geq \sum_{i \in V} 3iy_i + n - 3p \tag{12}$$

$$\sum_{i \in V} v_i \geq \sum_{k=1}^{p} 3k + n - 3p \tag{13}$$

These constraints are a consequence of the fact that it is assumed that each cycle must include at least three nodes, and are valid for the aggregated node-depot and node-cycle assignment models respectively. The next two propositions show how to derive these inequalities from the disaggregated models and the linking constraints (10) and (11), also indirectly proving their validity.

Proposition 20. *Constraints* (1c), (8a), (8e), (8h) *and* (10) *imply constraints* (12).

Proof. To derive constraints (12) from constraints of the DNDA models, we begin by observing that constraints (8h) and (8e) imply:

$$\sum_{d \in V} \sum_{i \in V} (d-1) {s_i}^d \geq \sum_{d \in V} 3(d-1)y_d$$

Rearranging the last term and using (1c), we get:

$$\sum_{d \in V} 3(d-1)y_d = \sum_{d \in V} 3dy_d - \sum_{d \in V} 3y_d = \sum_{d \in V} 3dy_d - 3\sum_{d \in V} y_d = \sum_{d \in V} 3dy_d - 3p$$

This leads to:

$$\sum_{d \in V} \sum_{i \in V} (d-1) {s_i}^d \geq \sum_{d \in V} 3dy_d - 3p$$

Observe also that constraints (8a) imply:

$$\sum_{i \in V} \sum_{d \in V} {s_i}^d = \sum_{d \in V} \sum_{i \in V} {s_i}^d = n$$

The previous two inequalities lead to:

$$\sum_{d \in V} \sum_{i \in V} (d-1) {s_i}^d + \sum_{d \in V} \sum_{i \in V} {s_i}^d = \sum_{i \in V} \sum_{d \in V} d{s_i}^d \geq \sum_{d \in V} 3dy_d + n - 3p$$

Finally, by using (10), we obtain (12). □

Proposition 21. *Constraints* (9a), (9d) *and* (11) *imply constraints* (13).

Proof. To derive constraints (13) from constraints of the DNCA models, we begin by observing constraints (9d) imply:

$$\sum_{k=1}^{p}\sum_{i\in V}(k-1)v_i^k \geq \sum_{k=1}^{p}3(k-1) = \sum_{k=1}^{p}3k-3p$$

Observe also that constraints (9a) imply:

$$\sum_{i\in V}\sum_{k=1}^{p}v_i^k = \sum_{k=1}^{p}\sum_{i\in V}v_i^k = n$$

The previous two inequalities lead to:

$$\sum_{k=1}^{p}\sum_{i\in V}(k-1)v_i^k + \sum_{k=1}^{p}\sum_{i\in V}v_i^k = \sum_{k=1}^{p}\sum_{i\in V}kv_i^k \geq \sum_{k=1}^{p}3k+n-3p$$

Finally, by using (11), we obtain (13). □

However, while constraints (8h) and the enhanced right-hand side in constraints (9d) appear to have a positive impact on the computational times or on the LP relaxation bounds, some preliminary computational results suggest that constraints (12) and (13) result in minor improvements to the LP relaxation bounds ((13) results in slightly greater improvements than (12)) and no considerable improvements to the computational times when added to the aggregated node-depot and node-cycle assignment models.

On the other hand, these negative results do not prevent further study in the relation between aggregated and disaggregated models.

4 Preventing more than p cycles

In this section we address the HpMP and present constraints to prevent solutions with more than p cycles. It is far from clear how to write compact sets of such constraints without "directing" the graph, that is, for edge $\{i,j\}$ we also distinguish whether the edge is used in the direction from i to j (arc (i,j) is used) or is used the direction from j to i (arc (j,i) is used). Thus, we introduce the binary variables $x_{ij} \in \{0,1\}$, $\forall i,j \in V$, which indicate whether arc (i,j) is in the solution, and add, to the BASE model, the sets of constraints:

$$u_{ij} = x_{ij} + x_{ji}, \quad \forall\{i,j\} \in E \tag{14a}$$

$$\sum_{j\in V:j\neq i} x_{ij} = 1, \quad \forall i \in V \tag{14b}$$

$$\sum_{j\in V:j\neq i} x_{ji} = 1, \quad \forall i \in V \tag{14c}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i,j \in V : \{i,j\} \in E \tag{14d}$$

Constraints (14b) and (14c) are usually included in MILP formulations for the ATSP. Observe that by using (14a) we can remove the u_{ij} variables from the models and obtain "pure" directed models. In fact, in the computational testing for the HpMP, pure directed models where used. Observe also that the constraints $x_{ij} + x_{ji} \leq 1$, $\forall\{i,j\} \in E$ (that result from (14a)) were not added to the directed models since they are implied by the continuity constraints of the models.

We present, in the next two subsections, adaptations of two models known from the Asymmetric TSP to prevent solutions with more than p cycles. As far as we know, these adaptations are new and, as noted in the introduction, we describe two models with substantially different LP strengths in order to evaluate the contribution of this subsystem to the model for the HpMP.

4.1 The adapted Desrochers and Laporte formulation

Besides the y_j and x_{ij} variables already introduced for the directed BASE model, the formulation presented in this subsection also uses non-negative variables u_i, $\forall i \in V$, that can be interpreted as indicating the position of a node in the cycle it is in (assuming an orientation in the cycle). Consider the following inequalities:

$$u_j \geq u_i + 1 - M(1 - x_{ij} + y_j) + (M-2)x_{ji}, \quad \forall i, j \in V,\ i \neq j \tag{15a}$$

$$(1 - y_j) \leq u_j \leq (M-1)(1-y_j), \quad \forall j \in V \tag{15b}$$

In inequalities (15a) and (15b), the value M equals $n-3(p-1)$, which is the size of the largest cycle in any feasible solution. Inequalities (15b) indicate that a node i is a depot ($y_i = 1$) if and only if its position, in the cycle, is zero ($u_i = 0$) (observe that as a consequence, we have $u_j \geq 1$ for any node j that is not a depot). The requirement that $y_i = 1$ if and only if $u_i = 0$ is not needed for the validity of the model, but it helps the explanation of the inequalities (15a) in the case when $y_i = 1$.

The inequalities (15a) are an adaptation of the well-known Desrochers and Laporte (DL) inequalities presented in [10] and guarantee that the position of two consecutive nodes in a cycle must increase by at least 1, except if node j is a depot. The adaptation includes the y_j variable in one of the terms of the inequality to guarantee this last condition. For a pair of nodes i, j such that $x_{ij} = 1$, when $y_j = y_i = 0$, these inequalities have the standard interpretation, that is, the combined effect of the two inequalities for pairs i, j and j, i (with i and j not being depots) states that $u_j = u_i + 1$. If one of those nodes is a depot, however, the effect of the two constraints for this pair of nodes changes. Assuming again that $x_{ij} = 1$, if $y_i = 1$, the two inequalities for pairs i, j and j, i state that $1 \leq u_j \leq M + 1$, and if $y_j = 1$, the two inequalities state that $-1 \leq u_i \leq M - 1$. In any case, we see that a cycle containing a depot is not forbidden.

These constraints, added to the BASE model, eliminate solutions with more than p cycles. To see this, consider a feasible solution with more than p cycles. If there are more than p cycles, at least one of these cycles does not have a depot. Using (15a) in a circular fashion along the arcs of the cycle we obtain a contradiction.

To conclude we make the following observations: *i*) Although we have "interpreted" the u_i variables as indicating the position of a node in the cycle, they need not be defined as integer (and are not). Also, as the model is defined now, we do not guarantee that the first non-depot node in the cycle has position equal to 1. Thus, for a cycle (j, q, r) with j being the depot, we may have $u_j = 0$, $u_q = 3$ and $u_r = 4$.

With respect to the HpMP, we maintain the same designation of the models that have been used for the HpMP$_{\geq}$, noting that the models for the HpMP are obtained from the models for the HpMP$_{\geq}$ by replacing the edge variables u_{ij} with the arc variables x_{ij} (according to equality (14a)), by replacing constraints (1b) and (1d) with constraints (14b), (14c) and (14d), and by adding the u_i variables and constraints (15a) and (15b).

4.2 The adapted multi-commodity flow formulation

Instead of the adapted DL constraints, we can use flows to prevent more than p cycles. There are several ways to derive flow based models for preventing more than p cycles. In this section, we present an adaptation of the well known multicommodity flow (MCF) model by Claus [8] for the ATSP. As noted in the introduction, for the ATSP, MCF based models are known to provide substantially better LP relaxation bounds than the ones provided by the DL model. Thus, although we do not expect the MCF based model to be competitive (in regards to computational times) with the DL based model, their inclusion in this study serves to have an idea of how the LP relaxation bounds for the whole model change when models for the subsystem preventing more than p cycles change.

Besides the variables x_{ij} and y_j, consider the variables g^q_{ij}, for all arcs $(i, j) \in A$ and for all nodes $q \in V : q \neq j$, indicating whether arc (i, j) and node q are in one of the cycles and arc (i, j) is in the path from node q to the depot of the cycle. Clearly, the variables g^q_{ij} are not defined for $q = j$.

Consider the following constraints:

$$\sum_{i \in V} g^q_{qi} + y_q = 1, \qquad \forall q \in V \tag{16a}$$

$$\sum_{i \in V} g^q_{ji} + y_j \leq 1, \qquad \forall q, j \in V : q \neq j \tag{16b}$$

$$\sum_{i \in V} g^q_{ij} - \sum_{i \in V} g^q_{ji} \leq y_j, \quad \forall q, j \in V : q \neq j \tag{16c}$$

$$g^q_{ij} \leq x_{ij}, \qquad \forall (i,j) \in A, \forall q \in V : q \neq j \tag{16d}$$

$$g^q_{ij} \in \{0,1\} \qquad \forall (i,j) \in A, \forall q \in V : q \neq j \tag{16e}$$

Constraints (16a) state that for any node q, either the node is the depot of a cycle or it is the origin of flow in some cycle (that is, there is a path from node q to the depot of the same cycle). Constraints (16b) state that a given node j can only be the depot of a cycle if it is not in the path (more precisely, there is no arc leaving node j) from any other node q to the depot of the cycle it belongs to. Constraints (16c) state that if there exists an arc coming into node j in the path from a node q, then either node j is the depot of a cycle or there exists an arc leaving node j in the path from node q. Constraints (16d) are the standard constraints linking the g^q_{ij} variables with the x_{ij} variables. Finally, constraints (16e) define the g^q_{ij} variables as binary.

To see that these constraints, when added to the BASE model, eliminate solutions with more than p cycles, observe again that as pointed out before, if there are more than p cycles, at least one of these cycles does not have a depot. Consider some node q in that cycle. The corresponding constraint (16a) guarantees that node q is the origin of a flow (since it is not a depot). By constraints (16c), that flow will traverse the entire cycle without being "absorbed" by any depot, and assuming node r is the node which precedes node q in that cycle, then, $\sum_{i \in V} g^q_{ir} = 1 = \sum_{i \in V} g^q_{ri}$ holds. If node r precedes node q, then, $x_{rq} = 1$, and, by constraints (14b), $x_{ri} = 0$, $\forall i \in V \setminus \{q\}$, also holds. But constraints (16d) imply $g^q_{ri} \leq x_{ri} = 0 \; \forall i \in V \setminus \{q\}$, and $g^q_{rq} = 1$ cannot hold since the variable g^q_{rq} is not defined. We therefore arrive at the conclusion that $\sum_{i \in V} g^q_{ri} = 0$ must hold, but this contradicts what was observed, since $\sum_{i \in V} g^q_{ri} = 1$ should hold. This contradiction is a consequence of considering a cycle with no depots, and therefore, these constraints eliminate solutions with more than p cycles.

5 Computational Results

In this section, the numerical results from our computational experiment are presented. Section 5.1 introduces the benchmark instances as well as the hardware and software configurations used in this computational experiment. Section 5.2 includes an analysis of the computational results obtained with all the models presented in this work for the HpMP$_{\geq}$, and has two subsections. In Section 5.2.1, we analyse the results obtained with the aggregated models for the HpMP$_{\geq}$, and in Section 5.2.2, we analyse the results obtained with the disaggregated models for the HpMP$_{\geq}$. Finally, Section 5.3 includes an overview of the results obtained for the HpMP. Regarding this last section, we observe that only a few models for the HpMP were tested, and these were selected based on the results reported in Section 5.2 for the HpMP$_{\geq}$.

5.1 Hardware / Software Configurations and Test Instances

For this computational experiment, we use a subset of the set of symmetric instances for the TSP from TSPLIB [19]. This subset includes 20 instances - gr24, fri26, bayg29, swiss42, **att48**, gr48, hk48, **eil51**, **berlin52**, brazil58, **st70**, **eil76**, **pr76**, **rat99**, **kroA100**, **kroB100**, **kroC100**, **kroD100**, **kroE100** and **rd100** - with the number of nodes ranging from 24 to 100. The instances whose names are bold have their edge weight functions modified to the euclidean distance (without rounding to integer values).

All tests were run on a computer with an Intel® Core™ i7-4790 CPU, 8GB of DDR3-1600 RAM running Windows 10 Pro, version 21H2, within which CPLEX 20.1.0 Concert Technology for `C++` was used. All CPLEX parameters are set to their default values, with the exception of the time limit, which

is set to one hour - if an instance is not solved within the time limit, an optimal value is not obtained. Instead, an interval is obtained, to which the optimal value is guaranteed to belong.

5.2 Results for the HpMP$_{\geq}$

In this subsection we analyse the results for the HpMP$_{\geq}$ taken from the four classes of models. We compare the performance of the aggregated models in Section 5.2.1 and of the disaggregated models in Section 5.2.2. In this last subsection we also compare the performance of aggregated versus disaggregated models.

For each one of the instances described before, each model for the HpMP$_{\geq}$ is tested for three different values of p, namely $p_1^* = \lfloor \frac{n}{4} \rfloor$, $p_2^* = \lceil \frac{p_1^* + p_3^*}{2} \rceil$ and $p_3^* = \lfloor \frac{n}{3} \rfloor$. For this testing, p_1^* and p_3^* are considered as they are the two largest values of p considered in the benchmark instances for the HpMP, and p_2^* is an intermediate value. Additionally, p_3^* is also of interest, as it is the only value of p for which the constraints used to prevent more than p cycles are not needed. We observe that this section (and Section 5.3) only includes the average data - the detailed results can be found in the appendix.

5.2.1 Results for Aggregated Models

In this subsection we compare all the aggregated models defined in Section 3.1. We compare the models within each class (node-depot and node-cycle assignment models), and also between the different classes. We also evaluate the impact of augmenting these formulations with constraints (4e), (6e) and (2).

Table 1 presents the average LP and RN gaps as well as the CPU times to obtain the optimal solutions of the 10 models previously described, four node-depot models and six node-cycle models, and Figure 1 includes two charts which, for a certain time threshold t, indicate how many instances were solved using the corresponding model in up to t seconds. The LP gap can be calculated as follows: $LP = 100 \times \frac{z^* - z_l}{z^*}$, where LP is the LP gap, z^* is the optimal value of that instance and z_l is the LP relaxation bound value. The RN (for "Root Node") gap can be calculated using a similar equation, replacing z_l with z_r, where z_r is the lower bound obtained by limiting CPLEX to the root node of the branch-and-bound tree. In most cases, z_r is considerably larger than z_l since CPLEX may also add generic cuts which improve this lower bound and which are not considered when calculating z_l. In Table 1, each row corresponds to one aggregated formulation presented in Section 3.1. For each row, the first three columns have the average LP gaps obtained with the corresponding model for all instances such that $p = p_1^*$, $p = p_2^*$ and $p = p_3^*$, respectively. The next three columns have the average RN gaps obtained with the corresponding model for all instances, and the last three columns indicate the average CPU times obtained with the corresponding model for all instances. Next to each average CPU time entry we include the number of tests for which either the time limit was reached or CPLEX ran out of memory.

	Average LP Gaps (%)			Average RN Gaps (%)			Average CPU Times (s)		
	p_1^*	p_2^*	p_3^*	p_1^*	p_2^*	p_3^*	p_1^*	p_2^*	p_3^*
ANDA	3.14%	4.86%	9.61%	2.06%	3.06%	7.01%	76 (0)	296 (1)	2161 (11)
SANDA	3.11%	4.75%	9.34%	2.05%	3.07%	6.85%	58 (0)	374 (0)	2191 (11)
ANDA$^+$	3.10%	4.72%	9.28%	1.87%	2.71%	6.14%	38 (0)	219 (0)	2203 (10)
SANDA$^+$	**3.09%**	**4.70%**	**9.19%**	**1.84%**	**2.57%**	**5.92%**	**35 (0)**	**124 (0)**	2186 (10)
ANCA	3.14%	4.86%	9.62%	2.49%	4.08%	8.21%	946 (3)	1332 (7)	2550 (13)
SANCA	3.13%	4.80%	9.46%	2.30%	3.55%	7.53%	283 (1)	868 (3)	2285 (12)
ANCA$^+$	3.12%	4.77%	9.37%	2.16%	3.16%	6.76%	234 (1)	846 (4)	2465 (13)
SANCA$^+$	3.12%	4.76%	9.32%	2.23%	3.15%	6.93%	127 (0)	662 (3)	**2157 (10)**
BASE	3.14%	4.86%	9.62%	N/A	N/A	N/A	N/A	N/A	N/A

Table 1: Average LP and RN gaps and CPU times obtained with each aggregated model for the HpMP$_{\geq}$, for each value of p

With respect to the aggregated node-depot assignment formulations, the computational results reported in Table 1 and Figure 1 indicate that the models ANDA$^+$ and SANDA$^+$ lead to lower computational times when compared with the corresponding non-enhanced models (although the difference is not very

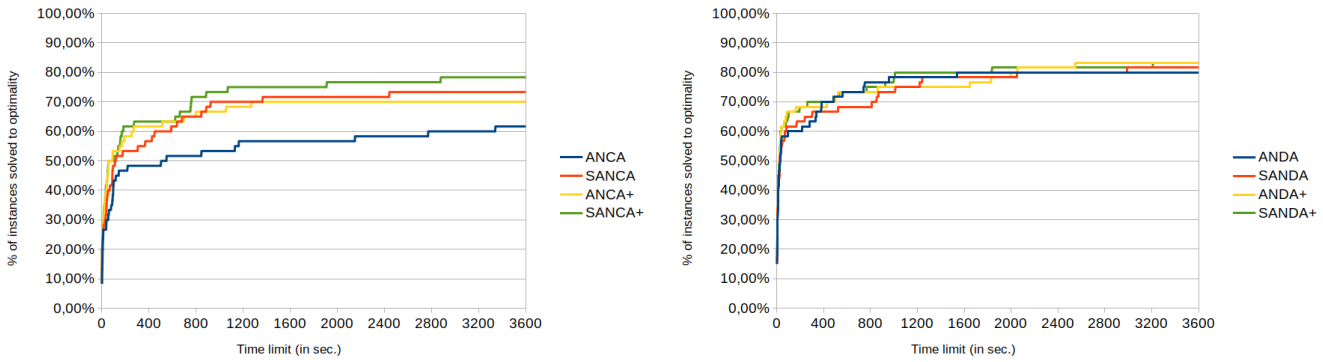

Figure 1: Number of solved instances of the benchmark set within fixed time thresholds for aggregated models for the HpMP$_{\geq}$

substantial). Clearly, the LP relaxation bounds of ANDA$^+$ and SANDA$^+$ are also always better than those of ANDA and SANDA respectively (this is obvious considering the definition of these models, although the difference is not very substantial).

A similar observation holds when we move from ANDA to SANDA and from ANDA$^+$ to SANDA$^+$ and the main conclusion is that the most competitive model from this class of models is the SANDA$^+$ model. The addition of constraints (2) also seems to have a greater impact on the LP relaxation bounds and on the computational times than the addition of constraints (4e) (observe that the average gaps of ANDA$^+$ are slightly lower than those of SANDA, and the same applies to the average computational times). Finally, we also observe that, for p_3^*, all the average times are very similar. This is due to the fact that, for many instances with $p = p_3^*$, all models reach the time limit of one hour, which reduces the differences between the average times of all the models.

With respect to the aggregated node-cycle assignment formulations, these computational results indicate that although only small improvements are obtained in the LP relaxation bounds when adding constraints (2), (as was the case of the aggregated node-depot assignment models), the results also show that the addition of these constraints appears to be effective in the resolution of the instances (once again, similarly to what happened with the aggregated node-depot models). Model ANCA seems to benefit the most with the addition of constraints (2).

The addition of the lifted constraints (6e) only leads to very small improvements of the LP relaxation bounds. However, similarly to what happened with the addition of constraints (2), the CPU times to obtain the optimal solution appear to improve with the inclusion of these lifted constraints. These constraints also seem to result in greater improvements to the computational times obtained with the aggregated node-cycle assignment models than constraints (4e) do for the aggregated node-depot assignment models.

As previously mentioned, the linking constraints (6c) and (5d), although not needed to define a valid model, help reduce the computational times. In addition to this, as mentioned earlier, constraints (6e) and (2) also further improve the computational times obtained with these models.

Despite resulting in considerably better computational times, SANCA$^+$ (the aggregated node-cycle assignment model with the lowest average computational times) is still not competitive with SANDA$^+$ (the aggregated node-depot assignment model with the lowest average computational times). We have already given one explanation for this behaviour in Section 3.1.3. One additional explanation for the differences in average CPU times between the aggregated node-cycle and node-depot assignment formulations might be related with the differences in average RN gaps between one class of formulations and the other. While the LP gaps between all the aggregated models are quite similar, the aggregated node-depot models seem to have better average RN gaps than their node-cycle counterparts. In other words, CPLEX appears to be able to generate stronger cuts from the node-depot formulations. This can also explain the lower computational times obtained with the aggregated node-depot formulations.

Finally, we have also provided the bounds given by the BASE model and observe that the LP relaxation bounds obtained with BASE are not often considerably worse than those obtained with the proposed

aggregated models. In fact, for all the instances in the benchmark set, the LP relaxation bounds given by ANCA are not better than those obtained with the BASE model. This was already pointed out in Section 3.1.3, where we have also stated that some testing with randomly generated instances shows that there are instances for which $v(\text{ANCA}_L) > v(\text{BASE}_L)$. The small differences between the LP gaps obtained with BASE and with the other aggregated formulations suggests that further improvements for these aggregated models may be a topic of interest for future work.

5.2.2 Results for Disaggregated Models

In this subsection we start by comparing the disaggregated models, node-depot with node-cycle, and then compare the aggregated models with the disaggregated ones. Table 2 follows the same format as Table 1, and Figure 2 follows the same format as Figure 1:

	Average LP Gaps (%)			Average RN Gaps (%)			Average CPU Times (s)		
	p_1^*	p_2^*	p_3^*	p_1^*	p_2^*	p_3^*	p_1^*	p_2^*	p_3^*
DNDA-	3.01%	4.61%	9.23%	2.67%	4.06%	8.42%	1772 (9)	2182 (9)	2909 (16)
DNDA	2.96%	4.53%	9.14%	2.69%	4.06%	8.39%	2030 (10)	2630 (13)	2918 (16)
DNDA$^{+}_{-}$	2.59%	3.55%	6.63%	2.09%	2.88%	5.78%	1576 (8)	1720 (8)	**2083 (11)**
DNDA^{+}	**2.29%**	**3.07%**	**5.86%**	**1.76%**	**2.37%**	**5.00%**	**1551 (8)**	**1541 (8)**	2145 (10)
DNCA-	3.14%	4.78%	9.09%	2.53%	3.97%	7.98%	1841 (9)	2752 (15)	2884 (16)
DNCA	3.12%	4.69%	8.72%	2.36%	3.76%	7.63%	2035 (9)	2825 (15)	2885 (16)

Table 2: Average LP and RN gaps and CPU times obtained with each disaggregated model for the HpMP$_{\geq}$, for each value of p

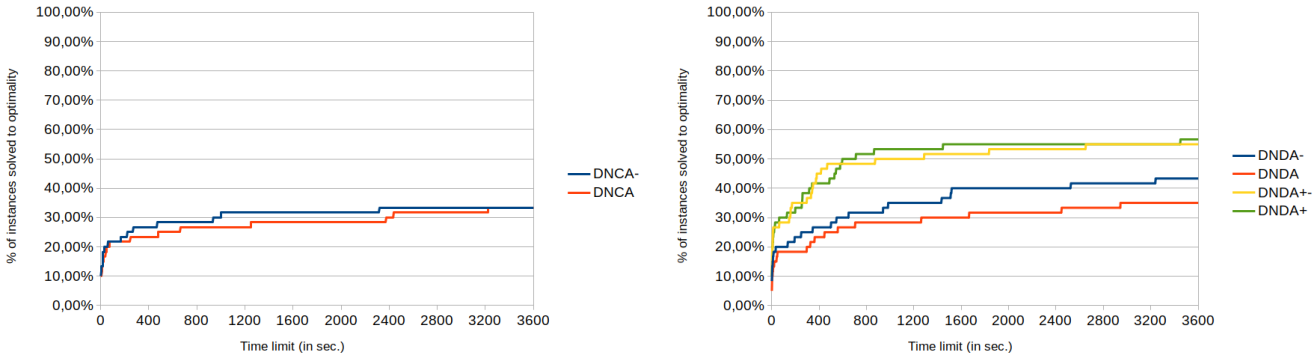

Figure 2: Number of solved instances of the benchmark set within fixed time thresholds for disaggregated models for the HpMP$_{\geq}$

The results from our computational testing show that the LP relaxation bounds of the DNDA- model are only slightly worse than those of the DNDA model, but this difference is much larger when comparing DNDA$^{+}_{-}$ and DNDA^{+}. This may explain why the CPU times obtained with DNDA- are considerably better than those obtained with DNDA and why this does not happen when comparing DNDA$^{+}_{-}$ with DNDA^{+}. As a conclusion, the models DNDA$^{+}_{-}$ and DNDA^{+} are the most competitive models, showing that the addition of constraints (8h) and (8i) is relevant, independently of using the complete or the reduced model.

The results also indicate that for the node-cycle case, the reduced model is not as advantageous as in the node-depot case. Also, and although not considerably worse, the node-cycle models are not as good as the node-depot models. In terms of computational times, these results are interesting since the node-cycle models have fewer variables and constraints than the node-depot models. One explanation for this behaviour is that in the DNCA models, the v_i^k variables need to be defined as binary. This is in contrast with the DNDA models, where only the ${s_d}^d$ variables need to be defined as binary. That is, the DNDA models are still valid if we relax the definition of the ${s_i}^d$, $\forall i, d \in V : d \leq i$ variables to be continuous.

Additionally, unlike all the models of the other classes and as mentioned in the beginning of Section 3.2.2, the DNCA models still allow many equivalent solutions, despite the fact that some of these equivalent solutions are eliminated by only defining the v_i^k variables for $k \leq i$. These symmetries can have a negative impact on the computational times obtained with these models.

We also observe that when solving with the DNDA model, CPLEX has run out of memory for all instances with $n \geq 99$ and $p = p_2^*$ or $p = p_3^*$. A similar situation happens with the DNDA- model for all instances with $n \geq 99$ and $p = p_3^*$. This is likely to be a consequence of the larger number of constraints and variables included in these disaggregated node-depot assignment models. However, the same does not happen when solving models DNDA$^+$- and DNDA$^+$, as these rarely resulted in CPLEX running out of memory, most likely due to the improvements to the LP relaxation bounds. Unlike what happened with models DNDA- and DNDA, and similarly to models DNDA$^+$- and DNDA$^+$, models DNCA- and DNCA also did not result in CPLEX running out of memory. This is likely to be a consequence of the lower number of variables and constraints these models have when compared with their node-depot counterparts.

We can also compare the results of the disaggregated models with the results from the aggregated models reported in the previous subsection. As noted in the introduction, and although providing better LP relaxation bounds (the model DNDA$^+$ provides substantially better LP bounds), for obtaining the optimal solutions the disaggregated models are not competitive and require considerably more CPU time when compared with the aggregated models. This difference between the CPU times obtained with the aggregated and disaggregated models might also be explained by the differences in RN gaps, as the average RN gaps obtained with the aggregated models are very similar to (if not lower than) the average RN gaps obtained with their disaggregated counterparts. The DNDA$^+$- and DNDA$^+$ formulations are the only exceptions, since they have the best average RN gaps out of all the models tested in this work.

Finally, comparing with the LP gaps obtained with BASE, we observe that the LP bounds produced by DNDA$^+$ and DNDA$^+$- are considerably better, in contrast to what happens with the other models, either aggregated or disaggregated. This suggests that a more complete study on the relationship between disaggregated and aggregated models, similarly to what has been described in Section 3.3, may lead to improvements on the aggregated formulations.

5.3 Results for the HpMP

In this subsection we analyse the results obtained for the HpMP. Based on the results for the HpMP$_{\geq}$, we only compare the results from a few of the aggregated models, namely the models SANDA$^+$ and SANCA$^+$.

For each one of the instances described in Section 5.1, we follow the literature and we test the models for the values $p_1 = \lfloor \frac{n}{10} \rfloor$, $p_2 = \lfloor \frac{n}{7} \rfloor$, $p_3 = \lfloor \frac{n}{5} \rfloor$, $p_4 = \lfloor \frac{n}{4} \rfloor$ and $p_5 = \lfloor \frac{n}{3} \rfloor$. Tables 3 (which includes average LP and RN gaps) and 4 (which includes average CPU times) follow the same format as Tables 1 and 2.

	Average LP Gaps (%)					Average RN Gaps (%)				
	p_1	p_2	p_3	p_4	p_5	p_1	p_2	p_3	p_4	p_5
SANDA$^+$	1.52%	1.17%	**1.83%**	**3.11%**	**9.19%**	**0.94%**	0.64%	**1.21%**	**2.25%**	**6.75%**
SANCA$^+$	1.52%	1.17%	1.84%	3.13%	9.32%	0.96%	**0.63%**	1.33%	2.43%	7.5%

Table 3: Average LP and RN gaps obtained with SANDA$^+$ and SANCA$^+$, for each value of p

	Average CPU Times (s)				
	p_1	p_2	p_3	p_4	p_5
SANDA$^+$	**417 (2)**	**22 (0)**	**70 (0)**	**247 (1)**	**2047 (10)**
SANCA$^+$	461 (2)	33 (0)	303 (0)	462 (1)	2183 (11)

Table 4: Average CPU times obtained with SANDA$^+$ and SANCA$^+$, for each value of p

The main conclusion from these results is that the SANDA$^+$ formulation results in the best CPU times and has, on average, marginally better LP relaxation bounds than SANCA$^+$. Regarding the LP relaxation bounds, there are no differences for the smallest values of p, and the difference between the average gaps

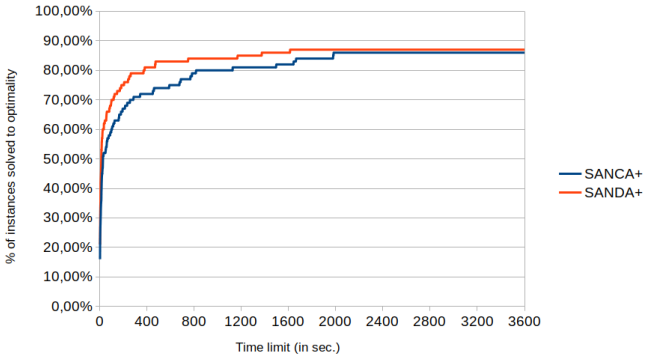

Figure 3: Number of solved instances of the benchmark set within fixed time thresholds for aggregated models for the HpMP

is the largest for p_5 (and even then, it is not substantial, at only 0.13% on average). Regarding average computational times, these follow a trend similar to the LP gap averages (that is, the differences between the CPU times grow with p). These also make it clear that SANDA$^+$ is the most competitive of these two models - even though there are some instances where the computational times obtained with SANCA$^+$ are lower than those obtained with SANDA$^+$, those differences are typically very minor (for more detailed results, we forward the reader to the appendix). Finally, an interesting observation arises when comparing the models for the HpMP with the models for the HpMP$_{\geq}$ - for all the instances with $p = p_4$ and $p = p_5$, these models produce the same LP relaxation bounds and optimal values, with the single exception of instance fri26 with $p = p_4$. In other words, for the vast majority of instances such that $p \geq p_4$, the adapted DL constraints (or any other constraints used to prevent more than p cycles) are not required, as the optimal solution will include p cycles regardless of whether such constraints are included. This behaviour was pointed out in the introduction of this paper and was used as a motivation to focus our study on the HpMP$_{\geq}$. Additionally, observe that the optimal values for all instances of the HpMP with $p = p_5$ must coincide with the optimal values of the corresponding instances of the HpMP$_{\geq}$ with $p = p_3^*$, as in this case, there can never be solutions with more than p_5 cycles. We remark, however, that the RN gaps are better for the HpMP$_{\geq}$ but these improvements are not propagated for the overall computational times.

Although we have made a study on compact formulations for the HpMP, we observe that the average gaps reported in Table 3, for the smaller values of p (p_1 to p_3), are comparable with the best LP relaxation bounds known from the literature (including the bounds associated to the models with exponentially sized set of inequalities reported by Erdoğan *et al.* [11], Bektaş *et al.* [3] and Barbato and Gouveia [1]). This also justifies the focus of this paper in studying these models for large values of p. The reported average gaps are, however, much worse for the biggest values of p, notably when compared with the bounds in the recent work by Barbato and Gouveia [1], where valid inequalities especially tailored for large values of p have been developed. As noted in the introduction, despite having a good performance to obtain the optimal integer solution when used within a ILP package, the adapted DL model is known to result in weak LP relaxation bound. Thus, we have also modelled the system preventing solutions with more than p cycles with the adapted MCF system (instead of the adapted DL model) described in Section 4.2. Since the HpMP model with the adapted MCF constraints contains many more variables than the HpMP model with the adapted DL constraints, we do not expect the former to be competitive with the latter to obtain the optimal solutions. However, we have compared the LP relaxation bounds with the adapted DL and with the adapted MCF systems using SANDA to prevent less than p cycles, in order to see whether some insight can be obtained with this experiment. A few computational results show that the differences are minimal, even for the smallest value of p for which the models for the HpMP were tested, that is $p = \lfloor \frac{n}{10} \rfloor$. In order to focus our analysis, we have also extended this comparison for $p = 2$, $p = 3$ and $p = 4$, and we have only performed these tests for instances with $n \leq 76$. These results are summarized in Table 5.

Looking at Table 5, we observe that the difference between the average gaps of both models is lower for larger values of p, and while it is substantial for $p = 2$, it is not so much so for $p = 4$. Also, while

	Average LP Gaps (%)			
	$p = 2$	$p = 3$	$p = 4$	$p = \lfloor \frac{n}{10} \rfloor$
DL / SANDA	2.91%	2.10%	1.74%	1.47%
MCF / SANDA	**1.08%**	**1.45%**	**1.46%**	**1.07%**

Table 5: Average LP gaps obtained with models for the HpMP featuring the adapted DL and MCF constraints, for each value of p and for $n \leq 76$

there is some difference between the gap averages for $p = \lfloor \frac{n}{10} \rfloor$, we observe that these averages are skewed by the test results obtained for the instances with less than 30 nodes, since, for such instances, $\lfloor \frac{n}{10} \rfloor = 2$. Indeed, if we calculate the average gaps for $p = \lfloor \frac{n}{10} \rfloor$ only for instances with more than 40 nodes (and up to 76 nodes), we see that the average LP gaps are now 1.36% and 1.29% for DL and MCF respectively. Considering what was said before, the results for $p = \lfloor \frac{n}{10} \rfloor$ also indicate that, for the values of p associated to the usual benchmark instances of the HpMP, the choice of the model for the system to prevent solutions with more than p cycles should be based more on computational issues. The results also indicate that for small values of p, we observe gains in the LP relaxation bounds when using MCF in place of DL. These results are consistent with what is known from the ATSP and indicate that if if the study is focused in small values of $p : p > 1$ then alternative models should be investigated.

5.4 Comparison with branch-and-cut methods from the literature

In this subsection, we compare the results obtained for the HpMP with the best formulation considered in this work (SANDA$^+$) with the results obtained with other (more sophisticated) branch-and-cut algorithms, namely, those presented by Erdoğan *et al.* [11], Bektaş *et al.* [3] and Barbato & Gouveia [1]. To allow for a more detailed comparison with the algorithm presented by Barbato & Gouveia [1], the formulation SANDA$^+$ for the HpMP was also tested on the instances considered in [1] with up to 400 nodes.

One remark is in order - the specifications of the computers in which these models were tested differ, and therefore, some of the differences in computational times may be explained by these specification differences. Regardless, we believe most of those differences to be explained by factors other than the specifications of the computers, since those specifications (in particular, regarding the CPU) are similar, with the exception of the CPU used to test Erdoğan *et al.*'s [11] algorithm, which is slower than the CPUs used to test the other algorithms.

The average data considered in this comparison can be found in Tables 6 and 7. In Table 6, only instances with node counts between 42 and 100 were considered, since those are the instances for which every algorithm was tested. More detailed results can be found in the Appendix D.

p	Erdoğan *et al.*	Bektaş *et al.*	Barbato & Gouveia	SANDA$^+$
p_1	304.27 (0)	39.47 (0)	**0.83 (0)**	490.11 (2)
p_2	320.06 (1)	31.18 (0)	**24.09 (0)**	26.06 (0)
p_3	490.93 (2)	**40.94 (0)**	235.23 (1)	81.92 (0)
p_4	273.24 (1)	**60.29 (0)**	513.47 (2)	290.33 (1)
p_5	1655.14 (6)	2632.35 (9)	**1266.72 (4)**	2407.19 (10)

Table 6: Average CPU times obtained with three algorithms from the literature and SANDA$^+$, for instances with node counts between 42 and 100

The results of Table 6 indicate that the compact formulation is not competitive for p_1 and p_5. However, SANDA$^+$ is quite competitive for p_2, p_3 and, to a lesser extent, p_4. We also see that the algorithm proposed by Bektaş *et al.* [3] results in the lowest computational times for p_3 and p_4, whereas the algorithm proposed by Barbato & Gouveia [1] results in the lowest average computational times for every other value of p.

p	Barbato & Gouveia	SANDA$^+$
p_1	**381.29 (2)**	1109.24 (7)
p_2	**316.59 (2)**	622.2 (4)
p_3	685.4 (5)	**474.41 (3)**
p_4	1123.55 (8)	**628.16 (4)**
p_5	**1403.12 (8)**	2449.44 (17)

Table 7: Average CPU times obtained with algorithm F from [1] and SANDA$^+$ for all instances for which both algorithms were tested

As for Table 7, it shows a clear trend: Barbato & Gouveia's [1] algorithm is substantially better if $p = p_1$ (that is, if it is close to 1) or if $p = p_5$ (that is, if it is equal or close to $\lfloor \frac{n}{3} \rfloor$), but that is not the case for the other three values of p - indeed, for p_3 and p_4, SANDA$^+$ even results in lower CPU times (on average). We observe, however, that for the instances with more than 226 nodes, SANDA$^+$ is not competitive with Barbato & Gouveia's [1] algorithm.

In conclusion, the results obtained with the SANDA$^+$ formulation are comparable with the results obtained with the more sophisticated branch-and-cut algorithms presented by Erdoğan *et al.* [11], Bektaş *et al.* [3] and Barbato & Gouveia [1] for p_2, p_3 and p_4. Indeed, and as it can be seen in Appendix D, SANDA$^+$ was even able to solve some instances in record time - for instance, kroC100 and u159 with $p = p_4$. It was also able to find new optimal solutions, for instance, for u159 with $p = p_3$ or d198 and pr226 with $p = p_4$.

6 Conclusions

In this paper, we have introduced, studied and analysed several classes of compact formulations for the symmetric Hamiltonian p-Median Problem (HpMP). We have focused our study on compact formulations for eliminating solutions with less than p cycles since such formulations are less well known and studied than formulations which prevent solutions with more than p cycles. The advantage of such formulations is that they can be readily used in combination with off-the-shelf optimization software, unlike other types of formulations possibly involving the use of exponentially sized sets of variables or constraints which usually require the use of specialized methods, such as constraint separation, which may not always be easy to understand, implement or use. The proposed formulations are based on a common motivation, that is, they contain variables that assign labels to nodes, and prevent less than p cycles by stating that different depots must have different labels and that nodes in the same cycle must have the same label. We introduce and study aggregated formulations (which consider integer variables that represent the label of the node) and disaggregated formulations (which consider binary variables that assign each node to a given label). The aggregated models are new. The disaggregated formulations are not, although in all of them new enhancements have been included to make them more competitive with the aggregated models.

The main conclusion of this study is that, in the context of compact formulations, it is worth looking at the new and more compact models with the node variables. Despite the weaker LP relaxation bounds, the fewer variables and constraints lead to faster computational times, especially when solving instances with more than 50 nodes, and manage to be competitive with more sophisticated branch-and-cut algorithms from the literature for instances with up to around 200 nodes (unless p is very close to 1 or $\lfloor \frac{n}{3} \rfloor$). The statements made at the end of sections 5.2.1 and 5.2.2 suggest that further improvements for these aggregated models may be a topic of future research combined with a deeper understanding of the relationship between disaggregated and aggregated models.

7 Acknowledgements

L. Gouveia and F. Canas were supported by Portuguese National Funding from FCT - Foundation for Science and Technology, under projects UIDB/04561/2020 and UIDP/04561/2020.

References

[1] M. Barbato and L. Gouveia. The hamiltonian p-median problem: Polyhedral results and branch-and-cut algorithm. *Working paper*, 2022. Submitted (Currently in Optimization Online: https://optimization-online.org/?p=21544).

[2] T. Bektaş, L. Gouveia, and D. Santos. New path elimination constraints for multi-depot routing problems. *Networks*, 70(3):246–261, 2017.

[3] 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.

[4] T. Bektaş, L. Gouveia, and D. Santos. Compact formulations for multi-depot routing problems: Theoretical and computational comparisons. *Computers & Operations Research*, 124:105084, 2020.

[5] E. Benavent and A. Martinez-Sykora. Multi-depot multiple tsp: A polyhedral study and computational results. *Annals of Operations Research*, 207, 08 2013.

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

[7] M. Burger, Z. Su, and B. De Schutter. A node current-based 2-index formulation for the fixed-destination multi-depot travelling salesman problem. *European Journal of Operational Research*, 265(2):463–477, 2018.

[8] A. Claus. A new formulation for the travelling salesman problem. *SIAM Journal on Algebraic Discrete Methods*, 5(1):21–25, 1984.

[9] G. Cornuejols and W. R. Pulleyblank. *Perfect triangle-free 2-matchings*, pages 1–7. Springer Berlin Heidelberg, Berlin, Heidelberg, 1980.

[10] M. Desrochers and G. Laporte. Improvements and extensions to the miller-tucker-zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27–36, 1991.

[11] 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.

[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 J. Pahl, T. Reiners, and S. Voß, editors, *Network Optimization*, pages 385–394. Springer Berlin Heidelberg, 2011.

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

[17] 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.

[18] T. Öncan, İ. K. Altınel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36(3):637–654, 2009.

[19] G. Reinelt. TSPLIB - a traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

[20] R. Roberti and P. Toth. Models and algorithms for the asymmetric traveling salesman problem: an experimental comparison. *EURO Journal on Transportation and Logistics*, 1(1-2):113–133, 2012.

A Test Results (HpMP$_{\geq}$)

This appendix includes the detailed test results obtained using the models proposed in this paper for the HpMP$_{\geq}$. A "N/A" entry in one of these tables indicates CPLEX ran out of memory while performing the corresponding test. We also recall that each test has a time limit of 1h, and that if that time limit is reached in a test, the corresponding entry in the table containing the CPU times is the gap, in %, between the obtained lower and upper bounds. Finally, for each row, the best result is in bold, unless all results are equal, in which case no result is in bold.

Instance	p	Model							
		ANDA	SANDA	ANDA$^+$	SANDA$^+$	ANCA	SANCA	ANCA$^+$	SANCA$^+$
gr24	p_1^*	3.28%	3.27%	3.27%	**3.27%**	3.28%	3.28%	3.28%	3.28%
	p_2^*	4.93%	4.82%	4.82%	**4.80%**	4.93%	4.87%	4.84%	4.83%
	p_3^*	7.02%	6.50%	6.51%	**6.40%**	7.02%	6.73%	6.58%	6.51%
fri26	p_1^*	0.34%	0.34%	**0.34%**	**0.34%**	0.34%	0.34%	0.34%	0.34%
	p_2^*	0.34%	0.33%	0.31%	0.31%	0.34%	0.32%	0.31%	**0.31%**
	p_3^*	0.56%	0.48%	0.27%	**0.24%**	0.56%	0.38%	0.34%	0.33%
bayg29	p_1^*	4.45%	**4.43%**	4.44%	**4.43%**	4.45%	4.45%	4.45%	4.45%
	p_2^*	5.85%	5.81%	5.81%	**5.80%**	5.85%	5.82%	5.84%	5.82%
	p_3^*	7.75%	7.66%	7.67%	**7.63%**	7.76%	7.70%	7.71%	7.67%
swiss42	p_1^*	1.90%	1.90%	1.90%	1.90%	1.90%	1.90%	1.90%	1.90%
	p_2^*	3.30%	3.24%	3.29%	**3.24%**	3.30%	3.30%	3.30%	3.30%
	p_3^*	6.00%	5.61%	5.68%	**5.38%**	6.00%	5.89%	5.87%	5.65%
att48	p_1^*	3.28%	3.27%	3.28%	**3.27%**	3.28%	3.28%	3.28%	3.28%
	p_2^*	5.97%	5.89%	5.90%	**5.85%**	5.97%	5.94%	5.93%	5.92%
	p_3^*	14.57%	14.27%	14.28%	**14.21%**	14.57%	14.45%	14.37%	14.29%
gr48	p_1^*	4.83%	4.83%	4.83%	4.83%	4.83%	4.83%	4.83%	4.83%
	p_2^*	6.86%	**6.85%**	6.85%	**6.85%**	6.86%	6.86%	6.86%	6.86%
	p_3^*	12.42%	12.28%	12.30%	**12.27%**	12.42%	12.41%	12.41%	12.38%
hk48	p_1^*	2.21%	**2.21%**	**2.21%**	**2.21%**	2.21%	2.21%	2.21%	2.21%
	p_2^*	3.47%	3.47%	3.47%	**3.47%**	3.47%	3.47%	3.47%	3.47%
	p_3^*	8.33%	8.29%	8.30%	**8.27%**	8.33%	8.31%	8.31%	8.29%
eil51	p_1^*	4.06%	3.96%	3.96%	**3.95%**	4.10%	4.02%	4.01%	3.97%
	p_2^*	6.01%	5.86%	5.87%	5.85%	6.11%	5.93%	5.86%	**5.80%**
	p_3^*	11.53%	11.34%	11.34%	11.30%	11.67%	11.39%	11.30%	**11.24%**
berlin52	p_1^*	1.81%	1.81%	1.81%	**1.81%**	1.81%	1.81%	1.81%	1.81%
	p_2^*	3.54%	3.53%	3.53%	**3.52%**	3.54%	3.54%	3.54%	3.54%
	p_3^*	8.13%	8.05%	8.07%	**8.00%**	8.13%	8.11%	8.10%	8.07%
brazil58	p_1^*	1.53%	1.53%	1.53%	**1.53%**	1.53%	1.53%	1.53%	1.53%
	p_2^*	4.35%	4.32%	4.32%	**4.31%**	4.35%	4.34%	4.33%	4.31%
	p_3^*	7.68%	7.56%	7.58%	**7.51%**	7.68%	7.62%	7.59%	7.57%
st70	p_1^*	1.21%	1.21%	1.21%	**1.20%**	1.21%	1.21%	1.21%	1.21%
	p_2^*	2.58%	2.55%	2.55%	**2.54%**	2.58%	2.57%	2.57%	2.56%
	p_3^*	9.50%	9.34%	9.26%	**9.21%**	9.50%	9.47%	9.37%	9.34%
eil76	p_1^*	4.12%	4.09%	4.09%	**4.09%**	4.12%	4.12%	4.12%	4.12%
	p_2^*	5.66%	5.58%	5.58%	**5.57%**	5.66%	5.64%	5.63%	5.61%
	p_3^*	10.12%	9.99%	9.99%	**9.98%**	10.14%	10.04%	10.00%	9.98%
pr76	p_1^*	5.25%	5.02%	4.92%	**4.89%**	5.25%	5.09%	5.08%	5.05%
	p_2^*	6.59%	6.22%	6.02%	**5.91%**	6.61%	6.33%	6.29%	6.26%
	p_3^*	10.03%	9.39%	9.17%	**8.96%**	10.07%	9.63%	9.53%	9.47%
rat99	p_1^*	5.47%	5.34%	5.20%	**5.15%**	5.47%	5.38%	5.29%	5.27%
	p_2^*	8.23%	7.74%	7.33%	**7.21%**	8.23%	7.87%	7.55%	7.50%
	p_3^*	12.37%	11.36%	10.63%	**10.43%**	12.37%	11.56%	10.90%	10.77%
kroA100	p_1^*	4.43%	**4.42%**	4.42%	**4.42%**	4.43%	4.43%	4.43%	4.43%
	p_2^*	6.70%	6.56%	6.56%	**6.54%**	6.70%	6.66%	6.66%	6.65%
	p_3^*	13.10%	12.66%	12.69%	**12.60%**	13.10%	12.94%	12.89%	12.84%
kroB100	p_1^*	2.17%	2.16%	2.16%	**2.15%**	2.17%	2.17%	2.17%	2.17%
	p_2^*	3.60%	3.50%	3.50%	**3.49%**	3.60%	3.56%	3.55%	3.53%
	p_3^*	11.29%	10.93%	10.93%	**10.83%**	11.29%	11.11%	10.99%	10.97%
kroC100	p_1^*	3.55%	3.51%	3.51%	**3.50%**	3.55%	3.54%	3.54%	3.54%
	p_2^*	5.83%	5.72%	5.71%	**5.68%**	5.83%	5.79%	5.78%	5.77%
	p_3^*	12.29%	12.07%	12.06%	**11.99%**	12.30%	12.20%	12.12%	12.07%
kroD100	p_1^*	3.48%	3.46%	3.46%	**3.46%**	3.48%	3.47%	3.47%	3.47%
	p_2^*	5.19%	5.10%	5.10%	**5.07%**	5.19%	5.16%	5.15%	5.14%
	p_3^*	10.28%	10.08%	10.04%	**9.96%**	10.29%	10.20%	10.12%	10.11%
kroE100	p_1^*	2.63%	2.61%	2.61%	**2.60%**	2.63%	2.62%	2.61%	2.61%
	p_2^*	3.59%	3.48%	3.47%	**3.45%**	3.59%	3.53%	3.51%	3.49%
	p_3^*	9.50%	9.22%	9.14%	**9.07%**	9.50%	9.35%	9.22%	9.19%
rd100	p_1^*	2.90%	2.90%	2.90%	2.90%	2.90%	2.90%	2.90%	2.90%
	p_2^*	4.52%	4.50%	4.50%	**4.50%**	4.52%	4.52%	4.52%	4.52%
	p_3^*	9.77%	9.67%	9.67%	**9.65%**	9.77%	9.74%	9.72%	9.72%

Table 8: LP gaps obtained with aggregated models for the HpMP$_{\geq}$

Instance	p	Model							
		ANDA	SANDA	ANDA$^+$	SANDA$^+$	ANCA	SANCA	ANCA$^+$	SANCA$^+$
gr24	p_1^*	2.8%	2.95%	2.76%	2.87%	3.08%	2.6%	2.51%	**2.41%**
	p_2^*	3.83%	3.51%	3.79%	3.64%	4.1%	4.00%	3.85%	**3.24%**
	p_3^*	5.42%	4.56%	**4.41%**	4.42%	4.94%	4.99%	4.91%	4.56%
fri26	p_1^*	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	p_2^*	**0.00%**	**0.00%**	**0.00%**	0.31%	**0.00%**	**0.00%**	**0.00%**	0.31%
	p_3^*	0.1%	**0.00%**	**0.00%**	**0.00%**	0.23%	**0.00%**	**0.00%**	**0.00%**
bayg29	p_1^*	4.1%	3.9%	3.96%	3.79%	4.19%	4.11%	3.98%	**3.46%**
	p_2^*	4.93%	4.99%	4.5%	**4.45%**	5.37%	5.2%	4.76%	4.84%
	p_3^*	6.16%	5.95%	4.12%	**4.11%**	6.45%	6.18%	5.02%	5.8%
swiss42	p_1^*	0.81%	0.94%	**0.65%**	**0.65%**	0.97%	0.88%	0.92%	0.74%
	p_2^*	1.96%	1.96%	1.78%	1.73%	2.31%	2.00%	**1.49%**	2.07%
	p_3^*	3.88%	3.00%	2.61%	**2.06%**	4.72%	3.81%	2.71%	2.8%
att48	p_1^*	2.32%	2.36%	2.24%	**1.89%**	2.59%	2.33%	2.4%	2.5%
	p_2^*	4.11%	**4.09%**	4.31%	4.17%	5.07%	4.88%	**4.09%**	4.31%
	p_3^*	12.06%	12.44%	11.8%	**10.41%**	12.92%	13.1%	11.4%	12.36%
gr48	p_1^*	3.7%	3.45%	3.42%	**2.84%**	4.32%	3.97%	3.53%	4.28%
	p_2^*	4.75%	5.00%	4.28%	**3.97%**	6.15%	4.98%	4.66%	4.08%
	p_3^*	9.62%	8.91%	8.45%	8.36%	11.18%	8.79%	**8.13%**	8.8%
hk48	p_1^*	1.64%	1.68%	1.21%	**1.07%**	2.21%	2.05%	2.09%	2.05%
	p_2^*	2.3%	2.19%	1.99%	**1.64%**	3.47%	2.87%	2.13%	2.48%
	p_3^*	6.41%	6.42%	6.33%	**5.3%**	7.66%	6.82%	5.88%	5.79%
eil51	p_1^*	2.89%	3.07%	2.85%	**2.45%**	3.44%	3.42%	3.42%	3.33%
	p_2^*	4.07%	3.95%	3.67%	**3.08%**	5.24%	5.08%	4.48%	4.75%
	p_3^*	8.89%	8.22%	7.61%	**7.33%**	10.01%	9.29%	9.33%	8.61%
berlin52	p_1^*	1.55%	1.42%	1.19%	**1.12%**	1.8%	1.73%	1.68%	1.76%
	p_2^*	2.22%	2.24%	2.05%	**1.62%**	3.53%	3.24%	2.56%	3.27%
	p_3^*	6.18%	6.05%	6.55%	**4.83%**	8.12%	6.82%	6.37%	6.65%
brazil58	p_1^*	0.7%	**0.47%**	0.5%	0.63%	0.7%	0.68%	0.64%	0.53%
	p_2^*	2.96%	2.91%	2.37%	**2.33%**	3.54%	3.00%	2.76%	2.87%
	p_3^*	5.69%	5.98%	**4.25%**	4.51%	6.99%	6.27%	5.63%	5.06%
st70	p_1^*	0.83%	0.81%	**0.79%**	0.8%	0.96%	0.86%	0.87%	0.86%
	p_2^*	1.67%	2.01%	**1.45%**	1.47%	2.33%	1.97%	1.75%	1.84%
	p_3^*	7.6%	7.73%	**6.93%**	7.1%	8.38%	8.81%	7.58%	7.43%
eil76	p_1^*	3.49%	3.38%	3.31%	**3.09%**	3.76%	3.75%	3.67%	3.67%
	p_2^*	4.13%	4.2%	4.21%	**3.68%**	5.27%	5.07%	5.04%	4.54%
	p_3^*	8.17%	7.76%	7.69%	**7.22%**	9.13%	9.08%	8.79%	8.3%
pr76	p_1^*	2.48%	2.66%	**2.18%**	2.61%	3.34%	3.3%	2.23%	2.73%
	p_2^*	3.47%	2.85%	2.41%	**2.37%**	4.12%	3.41%	2.74%	3.25%
	p_3^*	5.51%	4.83%	**4.76%**	**4.76%**	7.29%	6.31%	4.79%	5.86%
rat99	p_1^*	3.15%	3.53%	**2.54%**	3.55%	5.12%	3.4%	3.69%	3.77%
	p_2^*	4.47%	4.21%	3.2%	**2.71%**	7.89%	5.09%	4.66%	4.03%
	p_3^*	6.5%	6.32%	5.74%	**4.94%**	12.04%	8.27%	6.34%	8.08%
kroA100	p_1^*	2.46%	2.5%	2.47%	**2.3%**	3.51%	2.98%	2.76%	3.31%
	p_2^*	3.7%	3.86%	**3.08%**	3.47%	5.81%	4.69%	4.19%	3.7%
	p_3^*	9.06%	9.74%	7.7%	**7.5%**	11.35%	9.82%	9.19%	9.75%
kroB100	p_1^*	0.89%	0.83%	0.9%	**0.59%**	1.25%	1.22%	1.15%	1.29%
	p_2^*	1.4%	1.43%	**0.95%**	0.97%	2.51%	2.04%	1.76%	1.64%
	p_3^*	7.94%	8.03%	**7.16%**	7.31%	9.09%	8.91%	8.64%	8.08%
kroC100	p_1^*	2.75%	2.48%	**2.32%**	2.33%	3.00%	2.97%	2.66%	2.69%
	p_2^*	3.97%	4.15%	3.47%	**2.88%**	5.21%	4.62%	4.22%	3.78%
	p_3^*	9.34%	9.52%	**7.79%**	8.58%	10.22%	10.16%	9.11%	9.28%
kroD100	p_1^*	1.85%	1.9%	**1.84%**	1.88%	2.19%	2.13%	1.95%	1.98%
	p_2^*	3.09%	3.12%	3.05%	**2.81%**	3.73%	3.32%	3.28%	3.2%
	p_3^*	7.16%	7.44%	6.57%	**6.52%**	8.17%	7.75%	7.17%	7.31%
kroE100	p_1^*	1.74%	2.04%	**1.45%**	1.6%	2.00%	2.25%	2.02%	2.13%
	p_2^*	1.93%	2.16%	**1.45%**	1.73%	2.97%	2.5%	2.15%	2.08%
	p_3^*	7.1%	6.66%	**5.99%**	6.51%	7.71%	7.37%	6.59%	6.66%
rd100	p_1^*	1.18%	**0.72%**	0.89%	0.84%	1.52%	1.37%	1.05%	1.26%
	p_2^*	2.25%	2.5%	**2.21%**	2.4%	3.04%	2.97%	2.7%	2.81%
	p_3^*	7.31%	7.41%	**6.32%**	6.57%	7.56%	8.04%	7.64%	7.39%

Table 9: RN gaps obtained with aggregated models for the HpMP$_{\geq}$

Instance	p	Model							
		ANDA	SANDA	ANDA$^+$	SANDA$^+$	ANCA	SANCA	ANCA$^+$	SANCA$^+$
gr24	p_1^*	**0.2**	**0.2**	**0.2**	**0.2**	0.5	0.3	0.3	**0.2**
	p_2^*	0.4	0.6	0.3	**0.2**	1	0.4	0.3	0.5
	p_3^*	**0.3**	0.4	**0.3**	0.4	2	1.4	1.6	0.5
fri26	p_1^*	**0**	0.1	0.1	0.1	0.1	**0**	0.1	0.1
	p_2^*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	p_3^*	**0.1**	**0.1**	**0.1**	**0.1**	0.2	**0.1**	**0.1**	**0.1**
bayg29	p_1^*	1.1	1.1	**1**	1.1	2.4	2.4	2	2.3
	p_2^*	2.2	2	2.3	**1.7**	5.9	2.9	2.9	2.3
	p_3^*	3	3.2	4.1	**2.4**	8.8	4.4	4.3	4.4
swiss42	p_1^*	0.5	**0.4**	0.6	0.7	1.4	1.2	1.6	0.8
	p_2^*	**0.8**	1	0.9	0.9	5.1	1.1	2.6	2.4
	p_3^*	36.8	**8.8**	12.6	9.3	60.3	41.3	13.4	15.9
att48	p_1^*	2.1	2.3	1.9	**0.9**	12.9	11.4	6.4	7.7
	p_2^*	15.3	**14.5**	19.8	22.2	144.2	89.8	54.2	49
	p_3^*	3.92%	7.07%	**3.82%**	7.71%	N/A	6.48%	5.80%	7.08%
hk48	p_1^*	13.4	**6**	6.6	8.9	40.3	38.9	18	28.5
	p_2^*	**19**	26.5	23.2	24.9	92.1	52.9	56.7	41.8
	p_3^*	**338.9**	853.2	862.6	927.6	2150.7	924.3	514.2	2879.6
gr48	p_1^*	1.9	2.3	1.6	**1.4**	6.2	2	5.9	5.5
	p_2^*	5.2	**4.7**	7.8	6	37.8	17.4	14.2	11.6
	p_3^*	216.1	303.6	**161.5**	195	846.3	306.2	697.8	275
eil51	p_1^*	11.9	13.4	**10**	13.2	97.4	33.4	56.5	16.9
	p_2^*	40.8	24.6	21.1	**19.3**	119.9	95.1	50.5	48
	p_3^*	N/A	1.40%	2054.7	999.7	3.23%	3.69%	64.47%	**757.5**
berlin52	p_1^*	2.7	**1.4**	**1.4**	1.5	8.3	9.1	7.8	8.3
	p_2^*	10	12.8	**8.8**	11.7	100.6	27	29.2	14.5
	p_3^*	751.5	811.8	**426**	533.9	1130	639.3	1271.4	662.7
brazil58	p_1^*	0.9	0.9	1	**0.6**	6.6	2.8	1.5	1.6
	p_2^*	9.9	13.2	**7.8**	11.9	89.3	90.9	43.2	33.7
	p_3^*	742.4	1013.9	2549.5	77.24%	81.93%	**589.1**	79.04%	624.3
st70	p_1^*	1.4	1.3	1.3	**0.9**	11.8	5.2	1.6	2.8
	p_2^*	11.3	5.6	4.8	**4.6**	80.8	69.9	19.8	35
	p_3^*	1.74%	4.18%	2.61%	**3210.3**	79.74%	76.46%	75.94%	76.00%
eil76	p_1^*	955.3	869.9	523.4	**494.7**	0.66%	0.33%	0.56%	1070.3
	p_2^*	2.00%	1242.9	1829.7	**1010.7**	2.87%	2442.9	1.56%	0.79%
	p_3^*	70.91%	**46.55%**	67.80%	66.21%	73.02%	72.15%	71.44%	68.17%
pr76	p_1^*	16.9	12.2	11.7	**7.2**	97.4	47.6	24.6	29.9
	p_2^*	38.2	25.4	**16.3**	23.5	219.9	121	41.4	132.6
	p_3^*	1540	**1219.7**	1997.8	1838.2	2.97%	18.69%	2.17%	1911.7
rat99	p_1^*	384.8	84.9	**38.1**	40.3	1.22%	369.8	272.6	183.2
	p_2^*	379	240.7	**75.3**	91.2	1.59%	888.9	799	156.8
	p_3^*	2.66%	4.19%	4.52%	**1.53%**	13.91%	5.91%	2.27%	5.03%
kroA100	p_1^*	**28.6**	34.8	31.5	28.9	61.05%	177.5	170.5	169.1
	p_2^*	563.6	524.4	87.4	**77.6**	83.24%	1.32%	2.99%	0.56%
	p_3^*	86.24%	86.94%	**84.81%**	87.01%	86.87%	87.29%	86.42%	86.71%
kroB100	p_1^*	7.2	**5.8**	30.2	7.7	55.9	40.3	31.8	20
	p_2^*	27	66.2	23.5	**12.7**	550.1	449.9	153.3	160
	p_3^*	83.70%	84.99%	84.08%	84.43%	84.83%	83.60%	85.11%	**83.47%**
kroC100	p_1^*	30.4	**24.7**	28.2	28.1	2772.4	681.8	36.4	139.8
	p_2^*	279.5	2987.7	1648.7	**261.1**	24.25%	1.08%	1056.2	762.4
	p_3^*	**84.52%**	86.53%	86.86%	86.70%	88.35%	86.66%	87.38%	87.29%
kroD100	p_1^*	34.7	33.3	**22.3**	30.7	1163.3	426.9	91.5	753.3
	p_2^*	486.9	2051.2	**62.3**	764.8	76.98%	80.79%	1.76%	0.32%
	p_3^*	86.14%	85.21%	85.42%	85.58%	**83.60%**	85.59%	86.10%	84.13%
kroE100	p_1^*	**13.6**	47.9	28.8	20	503.2	112.2	92.4	55.7
	p_2^*	93.1	168.4	34.7	**30.8**	82.43%	845.9	191.6	100.3
	p_3^*	86.88%	86.52%	86.20%	**84.94%**	86.23%	86.56%	87.37%	86.91%
rd100	p_1^*	19.2	22.2	12.1	**11.8**	3343.8	89.2	253.5	52.8
	p_2^*	333.1	**65.6**	503	100.9	77.89%	1366.1	0.73%	887.4
	p_3^*	84.11%	84.09%	83.29%	82.35%	84.49%	84.13%	**82.27%**	**82.27%**

Table 10: CPU times (s) to reach optimality with aggregated models for the HpMP$_{\geq}$

Instance	p	Model					
		DNDA-	DNDA	DNDA$^{+}_{-}$	DNDA^{+}	DNCA-	DNCA
gr24	p_1^*	2.98%	2.89%	2.59%	**2.04%**	3.26%	3.26%
	p_2^*	4.34%	4.25%	3.39%	**2.71%**	4.51%	4.50%
	p_3^*	6.11%	6.04%	3.75%	**2.45%**	4.64%	3.75%
fri26	p_1^*	0.33%	0.33%	**0.31%**	**0.31%**	0.34%	0.34%
	p_2^*	0.30%	0.30%	0.28%	**0.28%**	0.34%	0.34%
	p_3^*	0.38%	0.38%	0.00%	**0.00%**	0.56%	0.55%
bayg29	p_1^*	4.32%	4.16%	4.03%	**3.62%**	4.43%	4.35%
	p_2^*	5.62%	5.41%	4.83%	**4.13%**	5.77%	5.58%
	p_3^*	7.39%	7.16%	5.45%	**4.27%**	7.50%	7.07%
swiss42	p_1^*	1.83%	1.76%	1.76%	**1.60%**	1.90%	1.90%
	p_2^*	3.12%	3.01%	2.60%	**2.34%**	3.30%	3.26%
	p_3^*	5.66%	5.49%	3.10%	**2.64%**	5.45%	4.95%
att48	p_1^*	3.16%	3.09%	2.79%	**2.41%**	3.24%	3.22%
	p_2^*	5.75%	5.65%	4.78%	**4.01%**	5.80%	5.56%
	p_3^*	14.23%	14.14%	10.56%	**9.96%**	13.02%	11.73%
hk48	p_1^*	4.62%	4.59%	4.20%	**3.59%**	4.83%	4.83%
	p_2^*	6.50%	6.45%	5.40%	**4.58%**	6.80%	6.77%
	p_3^*	11.88%	11.82%	8.58%	**7.25%**	11.37%	10.64%
gr48	p_1^*	2.13%	2.12%	1.55%	**1.26%**	2.21%	2.21%
	p_2^*	3.30%	3.29%	2.07%	**1.68%**	3.39%	3.38%
	p_3^*	8.03%	8.03%	5.41%	**4.85%**	7.70%	7.67%
eil51	p_1^*	3.83%	3.70%	3.42%	**2.60%**	4.00%	3.88%
	p_2^*	5.67%	5.48%	3.96%	**2.91%**	5.75%	5.44%
	p_3^*	11.08%	10.89%	7.11%	**5.74%**	10.47%	9.81%
berlin52	p_1^*	1.73%	1.69%	1.31%	**1.14%**	1.81%	1.81%
	p_2^*	3.35%	3.30%	2.42%	**2.13%**	3.53%	3.50%
	p_3^*	7.82%	7.78%	5.40%	**4.72%**	7.82%	7.75%
brazil58	p_1^*	1.48%	1.42%	1.35%	**1.17%**	1.52%	1.52%
	p_2^*	4.19%	4.08%	3.40%	**2.83%**	4.30%	4.24%
	p_3^*	7.40%	7.30%	4.87%	**3.90%**	7.35%	6.97%
st70	p_1^*	1.16%	1.16%	1.11%	**1.10%**	1.21%	1.21%
	p_2^*	2.44%	2.41%	2.13%	**1.92%**	2.58%	2.57%
	p_3^*	9.22%	9.18%	7.89%	**7.50%**	9.33%	9.28%
eil76	p_1^*	3.94%	3.85%	3.38%	**2.86%**	4.08%	4.04%
	p_2^*	5.40%	5.30%	4.10%	**3.42%**	5.58%	5.50%
	p_3^*	9.78%	9.67%	6.95%	**5.57%**	9.95%	9.63%
pr76	p_1^*	5.05%	4.99%	4.04%	**4.02%**	5.23%	5.23%
	p_2^*	6.29%	6.22%	4.44%	**4.41%**	6.57%	6.48%
	p_3^*	9.63%	9.55%	6.77%	**6.61%**	9.97%	9.63%
rat99	p_1^*	5.34%	5.33%	4.04%	**4.00%**	5.47%	5.47%
	p_2^*	7.95%	7.94%	5.45%	**5.34%**	8.22%	8.22%
	p_3^*	11.93%	11.92%	7.87%	**7.74%**	12.25%	12.24%
kroA100	p_1^*	4.25%	4.21%	3.68%	**3.08%**	4.43%	4.39%
	p_2^*	6.38%	6.29%	5.07%	**4.11%**	6.65%	6.39%
	p_3^*	12.62%	12.48%	9.72%	**8.15%**	12.58%	11.97%
kroB100	p_1^*	2.02%	1.95%	1.50%	**1.27%**	2.16%	2.14%
	p_2^*	3.31%	3.19%	2.01%	**1.64%**	3.51%	3.38%
	p_3^*	10.80%	10.67%	8.32%	**7.58%**	10.69%	10.36%
kroC100	p_1^*	3.40%	3.38%	3.05%	**2.81%**	3.52%	3.50%
	p_2^*	5.60%	5.57%	4.62%	**4.30%**	5.75%	5.72%
	p_3^*	11.95%	11.91%	9.80%	**8.99%**	12.06%	11.93%
kroD100	p_1^*	3.34%	3.27%	3.01%	**2.83%**	3.47%	3.44%
	p_2^*	4.95%	4.87%	4.11%	**3.62%**	5.16%	5.07%
	p_3^*	9.94%	9.85%	7.50%	**6.82%**	10.09%	9.77%
kroE100	p_1^*	2.53%	2.49%	2.22%	**1.90%**	2.62%	2.59%
	p_2^*	3.39%	3.31%	2.45%	**2.00%**	3.56%	3.51%
	p_3^*	9.17%	9.05%	6.75%	**6.16%**	9.43%	9.08%
rd100	p_1^*	2.81%	2.78%	2.59%	**2.24%**	2.90%	2.89%
	p_2^*	4.35%	4.31%	3.54%	**2.98%**	4.51%	4.48%
	p_3^*	9.49%	9.46%	6.87%	**6.27%**	9.59%	9.54%

Table 11: LP gaps obtained with disaggregated models for the HpMP$_{\geq}$

Instance	p	Model					
		DNDA-	DNDA	DNDA$^{+}_{-}$	DNDA^{+}	DNCA-	DNCA
gr24	p_1^*	2.62%	2.55%	2.24%	**1.85%**	2.64%	0.00%
	p_2^*	3.53%	3.57%	3.15%	2.34%	1.61%	**0.00%**
	p_3^*	5.67%	5.48%	2.49%	**0.00%**	3.50%	1.66%
fri26	p_1^*	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	p_2^*	**0.00%**	**0.00%**	0.28%	**0.00%**	**0.00%**	**0.00%**
	p_3^*	**0.00%**	**0.00%**	**0.00%**	**0.00%**	0.11%	0.11%
bayg29	p_1^*	3.86%	3.81%	3.56%	**3.16%**	4.03%	3.82%
	p_2^*	5.13%	5.02%	3.90%	**3.27%**	5.23%	4.75%
	p_3^*	6.67%	6.40%	4.63%	**3.82%**	6.42%	5.85%
swiss42	p_1^*	1.46%	1.47%	0.93%	**0.83%**	0.93%	0.87%
	p_2^*	2.36%	2.27%	**1.65%**	1.67%	2.29%	2.21%
	p_3^*	4.48%	3.91%	2.06%	**2.03%**	4.00%	3.70%
att48	p_1^*	2.38%	2.95%	2.37%	**2.04%**	2.78%	2.40%
	p_2^*	4.93%	5.16%	4.13%	**3.68%**	4.77%	4.32%
	p_3^*	13.60%	13.36%	9.81%	**9.42%**	11.39%	10.72%
gr48	p_1^*	4.17%	4.21%	3.63%	**2.82%**	4.05%	4.04%
	p_2^*	5.93%	5.77%	4.74%	**3.66%**	6.43%	6.25%
	p_3^*	11.01%	10.94%	7.18%	**6.22%**	10.25%	9.72%
hk48	p_1^*	2.04%	2.03%	1.17%	**1.05%**	2.20%	2.19%
	p_2^*	2.98%	3.04%	1.82%	**1.39%**	3.33%	3.29%
	p_3^*	7.29%	7.61%	4.75%	**3.80%**	7.49%	7.29%
eil51	p_1^*	3.40%	3.45%	2.75%	**2.40%**	3.49%	3.39%
	p_2^*	4.87%	4.95%	2.98%	**2.27%**	5.15%	4.77%
	p_3^*	9.95%	10.16%	6.17%	**5.07%**	8.51%	8.29%
berlin52	p_1^*	1.60%	1.55%	1.12%	**0.86%**	1.81%	1.78%
	p_2^*	2.98%	3.03%	2.26%	**1.86%**	3.38%	3.36%
	p_3^*	7.15%	7.22%	4.83%	**4.24%**	7.00%	6.35%
brazil58	p_1^*	0.94%	1.04%	1.17%	0.89%	**0.80%**	0.83%
	p_2^*	3.56%	3.56%	2.46%	**1.84%**	3.07%	3.36%
	p_3^*	6.65%	6.73%	3.10%	**2.62%**	5.39%	5.40%
st70	p_1^*	0.92%	0.95%	0.83%	**0.73%**	0.96%	0.95%
	p_2^*	2.25%	2.33%	1.61%	**1.38%**	2.29%	2.27%
	p_3^*	8.62%	8.93%	7.57%	**7.00%**	8.67%	8.31%
eil76	p_1^*	3.50%	3.45%	3.10%	**2.57%**	3.99%	3.79%
	p_2^*	4.82%	4.66%	3.57%	**3.15%**	5.26%	5.14%
	p_3^*	8.95%	8.75%	6.33%	**5.07%**	9.30%	9.11%
pr76	p_1^*	4.76%	4.74%	2.52%	**2.42%**	3.55%	3.83%
	p_2^*	5.89%	5.88%	**2.91%**	2.99%	4.96%	4.68%
	p_3^*	8.93%	8.87%	5.70%	**5.48%**	8.73%	8.49%
rat99	p_1^*	5.17%	5.17%	2.65%	**2.58%**	5.15%	5.12%
	p_2^*	7.41%	7.50%	3.53%	**3.34%**	7.99%	7.89%
	p_3^*	11.06%	11.06%	6.17%	**5.96%**	11.95%	11.91%
kroA100	p_1^*	3.76%	3.72%	3.20%	**2.51%**	3.51%	3.50%
	p_2^*	5.48%	5.43%	4.30%	**3.11%**	5.64%	5.47%
	p_3^*	11.21%	11.18%	8.90%	**7.43%**	11.32%	11.18%
kroB100	p_1^*	1.76%	1.69%	1.14%	**0.71%**	1.67%	1.40%
	p_2^*	2.73%	2.60%	1.66%	**1.01%**	2.73%	2.51%
	p_3^*	9.82%	9.65%	7.87%	**7.07%**	9.34%	9.28%
kroC100	p_1^*	3.05%	3.17%	2.83%	**2.28%**	3.10%	3.06%
	p_2^*	5.06%	5.18%	4.06%	**3.67%**	5.04%	5.09%
	p_3^*	11.34%	11.40%	8.62%	**7.78%**	10.82%	10.49%
kroD100	p_1^*	3.05%	3.00%	2.56%	**2.26%**	2.63%	2.61%
	p_2^*	4.46%	4.36%	3.51%	**2.89%**	4.13%	3.97%
	p_3^*	8.85%	9.03%	7.10%	**6.16%**	8.46%	8.05%
kroE100	p_1^*	2.37%	2.25%	2.13%	**1.66%**	2.04%	2.37%
	p_2^*	2.96%	2.87%	1.95%	**1.55%**	3.01%	3.17%
	p_3^*	8.30%	8.28%	5.74%	**4.98%**	8.77%	8.62%
rd100	p_1^*	2.63%	2.60%	2.03%	1.68%	1.35%	**1.23%**
	p_2^*	3.97%	4.02%	3.18%	**2.37%**	3.12%	2.71%
	p_3^*	8.76%	8.77%	6.64%	**5.81%**	8.29%	8.01%

Table 12: RN gaps obtained with disaggregated models for the HpMP$_{\geq}$

Instance	p	Model					
		DNDA-	DNDA	DNDA$^{+}_{-}$	DNDA^{+}	DNCA-	DNCA
gr24	p_1^*	1	1.3	0.4	0.4	**0.3**	0.4
	p_2^*	0.9	2.4	**0.4**	0.5	**0.4**	**0.4**
	p_3^*	3.2	7.1	**0.3**	**0.3**	0.4	0.4
fri26	p_1^*	**0.1**	0.2	**0.1**	0.2	**0.1**	**0.1**
	p_2^*	**0.1**	0.2	**0.1**	**0.1**	**0.1**	**0.1**
	p_3^*	**0.1**	0.3	**0.1**	**0.1**	0.2	0.2
bayg29	p_1^*	7.8	12.7	**2.5**	5.9	6.4	9.7
	p_2^*	15.9	21	3.7	**3.6**	20.7	16.7
	p_3^*	33.9	42.9	**3.4**	4.4	17.1	25.1
swiss42	p_1^*	3.1	3.4	**2.5**	3.2	2.7	10.5
	p_2^*	4.9	48.6	**3.4**	8.4	19.6	39.7
	p_3^*	547.2	704.5	**8.8**	28.5	60.7	76.9
att48	p_1^*	248.3	362.2	**158.1**	259.1	221.2	250
	p_2^*	939.7	2447.5	**332.3**	529.4	0.6%	0.57%
	p_3^*	50.75%	62.18%	0.79%	**3450.7**	6.25%	6.66%
hk48	p_1^*	648.8	1666.9	**296.5**	340.9	2316.7	3222.5
	p_2^*	1511.5	1.00%	**342.9**	487.9	2.55%	4.92%
	p_3^*	N/A	67.52%	**468.6**	595.1	4.83%	4.9%
gr48	p_1^*	135.9	295.3	12.7	**3.6**	271.7	662.6
	p_2^*	500.3	327	63.9	**63**	1.35%	1.18%
	p_3^*	4.22%	58.35%	**171.6**	253.2	3.75%	2.98%
eil51	p_1^*	1434.7	0.92%	**379.6**	865	936.8	2373.7
	p_2^*	2525.2	2.74%	417.3	**198.7**	2.16%	2.15%
	p_3^*	58.32%	63.13%	**873.9**	1445	5.94%	8.47%
berlin52	p_1^*	193.4	445.3	9.5	**4.4**	167.7	1249
	p_2^*	1518.4	2943.4	**147.3**	258.7	1.92%	0.66%
	p_3^*	45.16%	60.69%	**373**	543.6	2.6%	5.1%
brazil58	p_1^*	10.2	557.3	**8.2**	14.8	29.1	51.4
	p_2^*	982.4	68.25%	349.2	**21.2**	1000.5	2435.4
	p_3^*	68.02%	77.14%	**159.9**	577.6	1.86%	2.05%
st70	p_1^*	346.8	1262.6	**6.1**	10.8	469.5	479.1
	p_2^*	3240.8	0.57%	1286.6	**129.6**	2.67%	1.45%
	p_3^*	81.00%	N/A	**3.57%**	5.55%	8.14%	10.58%
eil76	p_1^*	73.57%	11.5%	**2.94%**	4.64%	8.46%	6.85%
	p_2^*	5.47%	78.85%	**2.35%**	3.19%	19.33%	15.2%
	p_3^*	77.92%	N/A	9.7%	**5.5%**	21.36%	26.2%
pr76	p_1^*	1.89%	5.51%	**1835.4**	1.24%	4.51%	6.2%
	p_2^*	73.32%	80.8%	**2651.3**	1.66%	11.7%	21.19%
	p_3^*	80.2%	**10.76%**	N/A	13.2%	42.72%	82.54%
rat99	p_1^*	84.47%	84.89%	4.36%	**2.79%**	22.46%	80.74%
	p_2^*	84.67%	N/A	5.13%	**4.42%**	79.19%	84.51%
	p_3^*	N/A	N/A	15.21%	**14.86%**	84.48%	85.07%
kroA100	p_1^*	88.05%	87.94%	8.18%	**3.05%**	5.47%	17.9%
	p_2^*	87.41%	N/A	23.65%	85.41%	**7.71%**	30.33%
	p_3^*	N/A	N/A	**29.38%**	87.4%	44.3%	88.4%
kroB100	p_1^*	2.08%	8.12%	3.45%	**710.4**	0.5%	2.82%
	p_2^*	87.03%	N/A	0.7%	**316.4**	26.38%	12.96%
	p_3^*	N/A	N/A	**28.17%**	86.49%	29.44%	29.56%
kroC100	p_1^*	71.19%	12.71%	5.69%	**2.88%**	4.91%	85.56%
	p_2^*	87.38%	N/A	**8.99%**	87.6%	17.01%	85.56%
	p_3^*	N/A	N/A	**28.69%**	87.91%	47.28%	43.13%
kroD100	p_1^*	**2.4%**	86.62%	2.55%	7.35%	4.17%	86.46%
	p_2^*	86.08%	N/A	4.14%	**3.29%**	29.72%	84.89%
	p_3^*	N/A	N/A	28.63%	42.38%	**24.52%**	87.22%
kroE100	p_1^*	88.04%	88.62%	6.91%	4.6%	**3.1%**	86.62%
	p_2^*	87.78%	N/A	9.78%	**6.95%**	10.72%	31.68%
	p_3^*	N/A	N/A	**7.83%**	87.88%	22.05%	88.06%
rd100	p_1^*	7.67%	87.53%	3.84%	3.68%	**2.75%**	86.87%
	p_2^*	85.42%	N/A	5.53%	**3.8%**	28.8%	85.18%
	p_3^*	N/A	N/A	**25.67%**	85.84%	35.56%	86.16%

Table 13: CPU times (s) to reach optimality with disaggregated models for the HpMP$_{\geq}$

B Test Results (HpMP)

This appendix includes the detailed test results obtained using the models proposed in this paper for the HpMP. A "N/A" entry in one of these tables indicates CPLEX ran out of memory while performing the corresponding test. We also recall that each test has a time limit of 1h, and that if that time limit is reached in a test, the corresponding entry in the table containing the CPU times is the gap, in %, between the obtained lower and upper bounds. Finally, for each row, the best result is in bold, unless all results are equal, in which case no result is in bold.

		LP Gaps (%)		RN Gaps (%)				LP Gaps (%)		RN Gaps (%)	
Instance	p	SANDA$^+$	SANCA$^+$	SANDA$^+$	SANCA$^+$	Instance	p	SANDA$^+$	SANCA$^+$	SANDA$^+$	SANCA$^+$
gr24	p_1	1.09%	1.09%	0.89%	0.89%	st70	p_1	1.52%	1.52%	1.33%	**1.27%**
	p_2	0.20%	0.20%	0.00%	0.00%		p_2	0.64%	0.64%	0.44%	0.44%
	p_3	0.20%	0.20%	0.00%	0.00%		p_3	0.38%	0.38%	**0.12%**	0.13%
	p_4	**3.27%**	3.28%	**2.91%**	3.09%		p_4	**1.20%**	1.21%	0.91%	0.91%
	p_5	**6.40%**	6.51%	5.43%	**5.27%**		p_5	**9.21%**	9.34%	**7.91%**	8.16%
fri26	p_1	3.40%	3.40%	3.07%	3.07%	eil76	p_1	0.41%	0.41%	0.28%	0.28%
	p_2	2.55%	2.55%	2.21%	2.21%		p_2	0.79%	0.79%	0.66%	0.66%
	p_3	1.46%	1.46%	1.12%	1.12%		p_3	2.07%	2.07%	**1.8%**	1.9%
	p_4	**0.67%**	0.68%	0.34%	0.34%		p_4	**4.09%**	4.12%	**3.48%**	3.85%
	p_5	**0.24%**	0.33%	0.00%	0.00%		p_5	**9.98%**	9.98%	**8.32%**	9.15%
bayg29	p_1	1.02%	1.02%	0.9%	0.9%	pr76	p_1	**2.36%**	2.37%	0.47%	0.47%
	p_2	0.19%	0.19%	0.00%	0.00%		p_2	**2.67%**	2.74%	**0.73%**	1.16%
	p_3	0.58%	0.58%	**0.33%**	0.41%		p_3	**4.34%**	4.43%	**2.13%**	3.13%
	p_4	**4.43%**	4.45%	**3.89%**	4.17%		p_4	**4.89%**	5.05%	**2.03%**	2.36%
	p_5	**7.63%**	7.67%	**5.65%**	6.33%		p_5	**8.96%**	9.47%	**4.74%**	6.48%
swiss42	p_1	1.42%	1.42%	0.49%	0.49%	rat99	p_1	0.46%	0.46%	0.2%	**0.16%**
	p_2	1.34%	1.34%	0.41%	0.41%		p_2	1.68%	1.68%	**1.41%**	1.43%
	p_3	1.34%	1.34%	**0.4%**	0.43%		p_3	**3.27%**	3.33%	**2.79%**	2.93%
	p_4	1.90%	1.90%	0.94%	**0.86%**		p_4	**5.15%**	5.27%	**3.2%**	4.23%
	p_5	**5.38%**	5.65%	3.25%	**2.19%**		p_5	**10.43%**	10.77%	**5.79%**	7.77%
att48	p_1	0.73%	0.73%	0.34%	0.34%	kroA100	p_1	2.61%	2.61%	**1.68%**	1.79%
	p_2	0.52%	0.52%	0.13%	0.13%		p_2	1.31%	1.31%	0.36%	0.36%
	p_3	1.63%	1.63%	**1.16%**	1.22%		p_3	2.46%	2.46%	**1.39%**	1.61%
	p_4	**3.27%**	3.28%	**2.46%**	2.77%		p_4	**4.42%**	4.43%	**3.15%**	3.41%
	p_5	**14.21%**	14.29%	**10.92%**	12.32%		p_5	**12.60%**	12.84%	**9.32%**	10.9%
gr48	p_1	1.49%	1.49%	0.9%	0.9%	kroB100	p_1	2.34%	2.34%	**1.61%**	1.79%
	p_2	0.75%	0.75%	**0.24%**	0.31%		p_2	2.06%	2.06%	1.5%	**1.37%**
	p_3	3.19%	3.19%	**2.44%**	2.75%		p_3	1.57%	1.57%	**0.97%**	0.98%
	p_4	4.83%	4.83%	**4.06%**	4.34%		p_4	**2.15%**	2.17%	1.42%	**1.27%**
	p_5	**12.27%**	12.38%	**7.93%**	10.02%		p_5	**10.83%**	10.97%	**8.98%**	9.06%
hk48	p_1	0.66%	0.66%	0.66%	0.66%	kroC100	p_1	1.10%	1.10%	0.65%	**0.61%**
	p_2	0.00%	0.00%	0.00%	0.00%		p_2	1.18%	1.18%	**0.69%**	0.79%
	p_3	0.84%	0.84%	0.84%	0.84%		p_3	**2.14%**	2.14%	**1.72%**	1.76%
	p_4	**2.21%**	2.21%	**1.54%**	2.21%		p_4	**3.50%**	3.54%	**2.73%**	3.02%
	p_5	**8.27%**	8.29%	**5.83%**	6.47%		p_5	**11.99%**	12.07%	**9.59%**	10.16%
eil51	p_1	0.86%	0.86%	0.53%	**0.31%**	kroD100	p_1	1.58%	1.58%	0.43%	**0.42%**
	p_2	**1.09%**	1.10%	0.99%	**0.83%**		p_2	1.56%	1.56%	0.68%	**0.39%**
	p_3	**3.11%**	3.15%	2.87%	**2.82%**		p_3	**2.47%**	2.47%	**1.31%**	1.38%
	p_4	**3.95%**	3.97%	**3.48%**	3.51%		p_4	**3.46%**	3.47%	**2.17%**	2.25%
	p_5	11.30%	**11.24%**	8.83%	**8.75%**		p_5	**9.96%**	10.11%	**7.53%**	8.03%
berlin52	p_1	0.22%	0.22%	0.21%	0.21%	kroE100	p_1	0.71%	0.71%	**0.3%**	0.37%
	p_2	0.01%	0.01%	0.00%	0.00%		p_2	0.76%	0.76%	0.56%	0.56%
	p_3	0.56%	0.56%	0.54%	0.54%		p_3	1.52%	1.52%	1.11%	**1.04%**
	p_4	**1.81%**	1.81%	**1.69%**	1.81%		p_4	**2.60%**	2.61%	**1.98%**	2.01%
	p_5	**8.00%**	8.07%	**5.47%**	7.58%		p_5	**9.07%**	9.19%	**7.04%**	7.77%
brazil58	p_1	3.90%	3.90%	3.09%	3.09%	rd100	p_1	2.49%	2.49%	**0.69%**	1.08%
	p_2	1.85%	1.85%	1.01%	1.01%		p_2	2.18%	2.18%	0.71%	**0.6%**
	p_3	0.87%	0.87%	**0.03%**	0.28%		p_3	2.67%	2.67%	**1.1%**	1.35%
	p_4	**1.53%**	1.53%	**0.69%**	0.7%		p_4	2.90%	2.90%	1.9%	**1.43%**
	p_5	**7.51%**	7.57%	**5.04%**	5.51%		p_5	**9.65%**	9.72%	**7.34%**	8.06%

Table 14: LP and RN Gaps (%) to reach optimality with aggregated models for the HpMP

Instance	p	Model SANDA$^+$	Model SANCA$^+$	Instance	p	Model SANDA$^+$	Model SANCA$^+$
gr24	p_1	0.5	**0.3**	st70	p_1	58.3	**30.1**
	p_2	0.1	0.1		p_2	**9.4**	18.9
	p_3	0.1	0.1		p_3	**2.1**	2.3
	p_4	**1**	1.7		p_4	**6**	14
	p_5	**1.2**	1.6		p_5	**3.09%**	4.38%
fri26	p_1	0.8	**0.4**	eil76	p_1	**7.3**	57.9
	p_2	0.9	**0.8**		p_2	**17.3**	63.9
	p_3	**0.3**	0.4		p_3	**370**	685.3
	p_4	**0.2**	0.3		p_4	1.30%	**1.02%**
	p_5	0.1	0.1		p_5	19.01%	**7.33%**
bayg29	p_1	**0.3**	0.4	pr76	p_1	**2.9**	16.9
	p_2	0.2	0.2		p_2	**10.5**	26.6
	p_3	0.3	0.3		p_3	**379.4**	1979
	p_4	**5.7**	6.8		p_4	**55.2**	88.2
	p_5	**11.3**	12.4		p_5	1372.7	**1125.7**
swiss42	p_1	2.3	**1.2**	rat99	p_1	5.5	**3.6**
	p_2	**0.7**	1		p_2	**249.1**	285.3
	p_3	**0.6**	1.1		p_3	**170.6**	1642.3
	p_4	**1.6**	3.2		p_4	**239.5**	588.1
	p_5	**5.9**	8.8		p_5	**2.14%**	26.34%
att48	p_1	**0.8**	1.3	kroA100	p_1	0.39%	**0.37%**
	p_2	**0.5**	0.8		p_2	**5.5**	25
	p_3	**5**	7.3		p_3	**23.1**	192.4
	p_4	**12.6**	16.2		p_4	**471.5**	1663.2
	p_5	**2.13%**	2.73%		p_5	29.41%	**15.72%**
gr48	p_1	**5.3**	5.6	kroB100	p_1	**748.4**	767.8
	p_2	**0.5**	0.9		p_2	**95.2**	104.4
	p_3	**41.6**	48.7		p_3	**1.9**	9.6
	p_4	**21.8**	75.7		p_4	**15.4**	96.6
	p_5	**261.3**	341.7		p_5	**6.89%**	14.01%
hk48	p_1	**3.8**	11.1	kroC100	p_1	**99**	160.8
	p_2	**0.3**	0.5		p_2	**8.1**	25.9
	p_3	**3.5**	7.2		p_3	**116.9**	675.3
	p_4	**12.8**	13.6		p_4	**146.4**	254.4
	p_5	**33.8**	124.6		p_5	**50.20%**	100.00%
eil51	p_1	**0.8**	1.1	kroD100	p_1	**3.3**	25.3
	p_2	**2.6**	14.1		p_2	6.7	**5.9**
	p_3	**79.3**	163.8		p_3	**124.7**	231.4
	p_4	85.5	**58.8**		p_4	**205**	1495.3
	p_5	**1612.6**	1.50%		p_5	**11.01%**	28.24%
berlin52	p_1	**2.1**	2.2	kroE100	p_1	**10.7**	114.3
	p_2	**0.3**	0.6		p_2	**4.6**	12.1
	p_3	**2.5**	12.4		p_3	**15.2**	178.2
	p_4	**11.5**	22.8		p_4	**33.9**	449.3
	p_5	**1167.7**	1981.5		p_5	**3.14%**	100.00%
brazil58	p_1	1.05%	**0.92%**	rd100	p_1	**181.3**	815.7
	p_2	**20.4**	22.6		p_2	**11.2**	51.9
	p_3	**1**	2.2		p_3	**55.1**	213.8
	p_4	**1.9**	4.7		p_4	**14.8**	780.8
	p_5	468.3	**458.7**		p_5	**5.85%**	17.86%

Table 15: CPU times (s) to reach optimality with aggregated models for the HpMP

C Test Results (DL and MCF comparison)

This appendix includes the detailed test results obtained using the adapted DL and MCF formulations, with SANDA used to prevent less than p cycles. Only instances with $n \leq 76$ were considered, and only the LP gaps were calculated. One last observation is in order: we were so far unable to determine the optimal values of instances brazil58 (with $p = 4$) and pr76 (with $p = 2$). However, we have obtained intervals to which the optimal values are guaranteed to belong. They are $[21898.1533, 22211]$ and $[104500, 106362.1561]$. To calculate the gaps for these two instances, we have used the upper bounds of the corresponding intervals. Even if these upper bounds are not guaranteed to be the optimal values, the differences in the average gaps should not be substantial, as all other gaps were calculated using the corresponding optimal values.

		Model	
Instance	p	DL / SANDA	MCF / SANDA
gr24	$2 = p_1$	1.09%	**0.15%**
	3	0.2%	**0.04%**
	4	0.2%	0.2%
fri26	$2 = p_1$	3.4%	**0.71%**
	3	2.55%	**1.44%**
	4	2.76%	**2.43%**
bayg29	$2 = p_1$	1.02%	**0.13%**
	3	0.13%	0.13%
	4	0.19%	0.19%
swiss42	2	3.23%	**0.92%**
	3	2.84%	**2.00%**
	$4 = p_1$	1.42%	1.42%
att48	2	3.24%	**2.15%**
	3	1.43%	**1.03%**
	$4 = p_1$	0.73%	**0.68%**
gr48	2	2.81%	**2.00%**
	3	2.05%	**1.76%**
	$4 = p_1$	1.49%	**1.47%**
hk48	2	1.43%	**0.6%**
	3	0.76%	**0.67%**
	$4 = p_1$	0.66%	**0.65%**
eil51	2	1.42%	**1.09%**
	3	0.55%	0.55%
	4	0.66%	0.66%
	p_1	0.86%	0.86%
berlin52	2	0.58%	**0.00%**
	3	0.04%	**0.00%**
	4	0.27%	**0.25%**
	p_1	0.22%	0.22%
brazil58	2	7.12%	**0.78%**
	3	6.76%	**5.17%**
	4	5.92%	**4.95%**
	p_1	3.9%	**3.6%**
st70	2	5.15%	**0.44%**
	3	4.33%	**1.42%**
	4	3.55%	**1.55%**
	p_1	1.52%	**1.26%**
eil76	2	0.45%	**0.35%**
	3	0.22%	**0.18%**
	4	0.13%	0.13%
	p_1	0.41%	0.41%
pr76	2	6.93%	**4.69%**
	3	5.47%	**4.44%**
	4	4.66%	**4.42%**
	p_1	2.37%	2.37%

Table 16: LP gaps (%) obtained with models for the HpMP including the adapted DL and MCF constraints

D Test Results (Comparison with literature algorithms for the HpMP)

This appendix includes the computational times obtained in many instances using the SANDA$^+$ formulation and three other algorithms from the literature for the HpMP.

	p	Erdoğan *et al.*	Bektaş *et al.*	Barbato & Gouveia	SANDA$^+$
gr24	2	0.31	N/A	**0**	0.45
	3	0.25	N/A	**0**	0.11
	4	0.27	N/A	**0**	0.1
	6	0.51	N/A	**0.23**	0.98
	8	**0.24**	N/A	0.29	1.23
fri26	2	0.41	N/A	**0**	0.75
	3	0.31	N/A	**0.01**	0.89
	5	0.44	N/A	**0**	0.3
	6	0.37	N/A	**0**	0.2
	8	0.21	N/A	**0**	0.15
bayg29	2	0.56	N/A	**0**	0.32
	4	0.5	N/A	**0**	0.15
	5	0.53	N/A	**0.01**	0.35
	7	2.15	N/A	**1.35**	5.69
	9	**1.73**	N/A	2.44	11.3
swiss42	4	1.37	1	**0.02**	2.29
	6	1.7	1	**0.06**	0.73
	8	1.56	1	**0.04**	0.59
	10	2.02	1	**0.07**	1.63
	14	1.12	1	**0.19**	5.93
att48	4	3.73	1	**0.01**	0.76
	6	3.41	1	**0.03**	0.49
	9	3.99	2	**0.33**	5.01
	12	3.99	4	**3.36**	12.62
	16	285.9	N/A	**1.28**	2.13%
gr48	4	2.82	3	**0.08**	5.32
	6	1.76	2	**0.02**	0.51
	9	13.7	13	**6.3**	41.57
	12	**4.91**	8	22.95	21.84
	16	24.25	208	**1.28**	261.29
hk48	4	3.48	2	**0.1**	3.85
	6	2.88	**0**	**0**	0.34
	9	3.05	2	**0.32**	3.53
	12	3.41	4	**2.25**	12.82
	16	10.04	118	**0.19**	33.79
eil51	5	4.58	4	**0.08**	0.79
	7	6.88	5	**0.72**	2.62
	10	41.32	**10**	15.7	79.34
	12	14.41	**14**	24.05	85.54
	17	50.96	1701	**1.48**	1612.63
berlin52	5	3.66	2	**0.19**	2.12
	7	2.57	1	**0**	0.29
	10	4.43	2	**0.18**	2.54
	13	4.68	2	**1.93**	11.47
	17	48.81	36	**5.22**	1167.73

Table 17: CPU times obtained with three algorithms from the literature and SANDA$^+$ (part 1)

	p	Erdoğan *et al.*	Bektaş *et al.*	Barbato & Gouveia	SANDA$^+$
brazil58	5	78.9	12	**0.73**	1.05%
	8	36.95	6	**0.14**	20.44
	11	5.14	5	**0.01**	0.96
	14	4.72	2	**0.99**	1.94
	19	31.13	71	**0.76**	468.27
st70	7	18.11	5	**0.41**	58.34
	10	12.56	4	**0.14**	9.38
	14	8.66	3	**0.08**	2.1
	17	11.16	9	**0.71**	6
	23	1137.77	N/A	**192.41**	3.09%
eil76	7	20.97	10	**0.3**	7.25
	10	18.6	38	**2.14**	17.25
	15	207.04	**58**	89.3	370.04
	19	371.35	**133**	1.87%	1.30%
	25	**1025.73**	N/A	3.12%	19.01%
pr76	7	25.29	5	**0.14**	2.93
	10	224.4	**8**	8.85	10.54
	15	0.70%	**34**	224.94	379.36
	19	45.62	**7**	287.79	55.23
	25	867.49	**12**	38.56	1372.67
rat99	9	90.16	12	**0.72**	5.49
	14	2.55%	**23**	387.77	249.09
	19	2.19%	**43**	1.90%	170.65
	24	1.17%	**44**	7.24%	239.46
	33	2.85%	3003	**16.11**	2.14%
kroA100	10	2993.41	151	**3.29**	0.39%
	14	40.47	95	**0.74**	5.52
	20	57.24	30	**5.46**	23.13
	25	77.87	**51**	435.69	471.45
	33	2.43%	N/A	**2504.77**	29.41%
kroB100	10	1575.86	145	**2.84**	748.38
	14	1292.72	143	**1.67**	95.18
	20	114.7	75	**0.39**	1.87
	25	34.89	16	**3.12**	15.43
	33	**2.23%**	N/A	4.46%	6.89%
kroC100	10	93.61	98	**2.35**	98.97
	14	77.78	67	**6.58**	8.13
	20	229.62	55	**23.12**	116.93
	25	197.6	436	543.1	**146.43**
	33	4.03%	N/A	**2942.86**	50.20%
kroD100	10	50.5	48	**0.14**	3.27
	14	46.87	42	**0.08**	6.71
	20	254.33	197	**14.88**	124.69
	25	**154.5**	160	155.73	205.02
	33	**1.02%**	N/A	1.62%	11.01%

Table 18: CPU times obtained with three algorithms from the literature and SANDA$^+$ (part 2)

	p	Erdoğan *et al.*	Bektaş *et al.*	Barbato & Gouveia	SANDA$^+$
kroE100	10	28.92	25	**0.09**	10.73
	14	28.45	37	**0.42**	4.59
	20	51.43	65	**4.11**	15.21
	25	62.6	92	38.75	**33.9**
	33	3054.13	N/A	**1429.21**	3.14%
rd100	10	177.19	147	**2.66**	181.31
	14	42.96	57	**0.15**	11.2
	20	149.61	101	**13.76**	55.12
	25	51.3	42	**8.43**	14.82
	33	1.66%	N/A	**0.29%**	5.85%
pr124	12	N/A	N/A	**32.64**	3.58%
	17	N/A	N/A	**255.94**	0.63%
	24	N/A	N/A	**2.22**	7.97
	31	N/A	N/A	**5.26**	23.43
	41	N/A	N/A	**221.92**	22.81%
u159	15	N/A	N/A	**2.77**	160.65
	22	N/A	N/A	**16.54**	21.83
	31	N/A	N/A	0.14%	**324.73**
	39	N/A	N/A	2.74%	**480.53**
	53	N/A	N/A	**129.42**	100.00%
rat195	19	N/A	N/A	**2978.33**	3455.48
	27	N/A	N/A	6.25%	**0.72%**
	39	N/A	N/A	22.20%	**1.19%**
	48	N/A	N/A	32.22%	**6.89%**
	65	N/A	N/A	**1595.81**	100.00%
d198	19	N/A	N/A	**0.85%**	3.13%
	28	N/A	N/A	**1.45**	255.76
	39	N/A	N/A	**103.62**	139.54
	49	N/A	N/A	2.65%	**644.58**
	66	N/A	N/A	**2.53%**	100.00%
pr226	22	N/A	N/A	**4.39%**	6.78%
	32	N/A	N/A	**1.03%**	1.47%
	45	N/A	N/A	**0.95**	143.52
	56	N/A	N/A	47.40%	**69.24**
	75	N/A	N/A	**69.69%**	100.00%
gil262	26	N/A	N/A	**26.86**	4.39%
	37	N/A	N/A	**167.72**	1677.69
	52	N/A	N/A	**0.28%**	13.21%
	65	N/A	N/A	**23.77%**	100.00%
	87	N/A	N/A	**32.56%**	100.00%
rd400	40	N/A	N/A	**39.96**	100.00%
	57	N/A	N/A	**496.84**	1.12%
	80	N/A	N/A	**16.32%**	100.00%
	100	N/A	N/A	**38.55%**	100.00%
	133	N/A	N/A	**51.22%**	100.00%

Table 19: CPU times obtained with three algorithms from the literature and SANDA$^+$ (part 3)