Finding the root graph through minimum edge deletion

Martine Labbé, Alfredo Marín, Mercedes Pelegrín

Finding the root graph through minimum edge deletion

Martine Labbé[1], Alfredo Marín[2], Mercedes Pelegrín *[2]

Abstract

The line graph of a graph G has one node per each edge of G, two of them being adjacent only when the corresponding edges have a node of G in common. In this work, we consider the problem of finding the minimum number of edges to delete so that the resulting graph is a line graph, which presents an interesting application in haplotyping of diploid organisms. We propose an Integer Linear Programming formulation for this problem. We compare our approach with the only other existing formulation for the problem and explore the possibility of combining both of them. Finally, we present a computational study to compare the different approaches proposed.

Keywords: Line Graphs, Discrete Optimization, Haplotyping.

1. Introduction

Given a graph, its *line graph* is another graph whose vertices are the edges of the original one. Further an edge links two nodes of the line graph if and only if the corresponding edges of the original graph share a node. Although line graph is the term most commonly used, it has received many names in the literature, which are, to our knowledge: *interchange graph* [17], *derived graph* [2] and *dual graph* [18]. A graph is said to be *line-invertible* if it is isomorphic to the line graph of some other graph, called the *root*. Although obtaining the line graph of a given graph is straightforward, doing the reverse is not a trivial task. In fact, root graph reconstruction has attracted the attention of researchers throughout the years [10,12,14].

In genetics, haplotypes codify certain regions of the genome that show a statistically significant variability within a population. It has been observed that such variability plays an important role in human variation and genetic diseases [8,15]. Haplotype phasing, which consists of estimating the haplotypes that produced a current population of genotypes, is a primary problem in the analysis of genetic data. In this context, consistency relations between genotypes that could have been originated from a common ancestor are codified by a graph. Root graph reconstruction is useful here to estimate the original population size, that is, the number of generating haplotypes. However, if all the consistency relations are considered, sometimes reconstruction from the graph is not possible. In other words, the graph encoding consistency

[☆]Department of Computer Sciences, Université Libre de Bruxelles, Belgium.
INRIA, Lille-Nord Europe, France.
[☆☆]Department of Statistics and Operational Research, Universidad de Murcia, Spain.
* currently with Laboratoire d'Informatique de l'X (LIX), École Polytechnique, Palaiseau, France.
Email addresses: mlabbe@ulb.ac.be (Martine Labbé), amarin@um.es (Alfredo Marín), pelegringarcia@lix.polytechnique.fr (Mercedes Pelegrín *)

(a) K_3 (b) $K_{1,3}$

Figure 1: The only two connected graphs having the same line graph, K_3

relations is not line-invertible. In these cases, one needs to disregard some of these relations, that is, to delete some of the edges of the consistency graph. A combinatorial problem then arises, namely which edges to remove so that the graph is disrupted as little as possible.

In this paper we study the problem of identifying a set of edges of minimum cardinality that have to be deleted from a graph so that it becomes line-invertible. In general, when π is a property of a graph, we can define the corresponding *edge deletion problem* as follows: "find the minimum number of edges whose deletion produces a graph satisfying π". For several common properties, including that of being line-invertible, edge deletion problems are NP-hard [20].

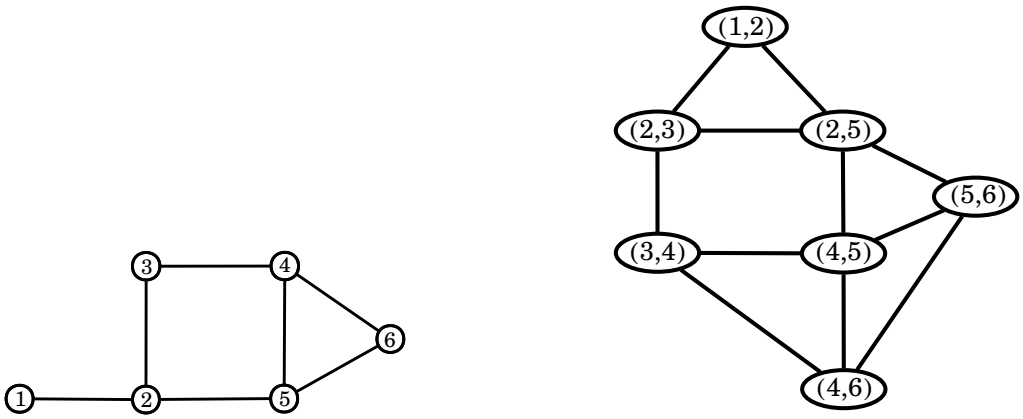
The edge-deletion problem for the property of being line-invertible, EDPL from now on, can be seen as a variant of graph coloring. This observation was used by Halldórsson et al. [7] to derive an ILP formulation which provides a certain node coloring of the graph resulting from the minimum edge deletion. To our knowledge, this is the only ILP approach for EDPL that has been developed so far. In this paper, we present an alternative ILP formulation that has a reduced number of variables, which represent color sharing among nodes without explicitly stating which are the colors shared. We present several valid inequalities for our model. Preliminary computational tests showed that dual bounds of our enhanced formulation and the model in [7] were not comparable. In view of these results, we introduce a third approach that combines the previous two formulations and some linking inequalities, which results in a family of tighter formulations. Our computational experiments allow empirical comparison between the different models and ultimately demonstrate the utility of the proposed alternatives.

The paper is organized as follows. In a preliminary section, notation and basic concepts are presented, together with the haplotype phasing problem. Section 3 discusses the link between line-invertible graphs and graph coloring and presents the formulation for EDPL proposed by Halldórsson et al. afterwards. In Section 4 we propose the new formulation for the problem together with some valid inequalities. A "mixed" approach, which combines elements from both mentioned formulations and some linking inequalities, is introduced in Section 5. In Section 6, our computational experience is reported and a comparative analysis of the different formulations is presented. Finally, some conclusions close the work.

2. Preliminaries

2.1. Line graphs

Given a graph $G = (V, E)$, its *line graph*, $L(G) = (E, F)$, has one node corresponding to each edge $e \in E$ and one edge $f = (e_1, e_2) \in F$ for all $e_1, e_2 \in E$, $e_1 \neq e_2$, that share a node, that



(a) G (b) $L(G)$

Figure 2: A graph and its line graph

is, $e_1 = (u, v)$ and $e_2 = (v, w)$ for some $u, v, w \in V$. Conversely, a graph H is *line-invertible* if there exists G such that $H = L(G)$. G is then known as the *root graph* of H. If a line-invertible graph is not connected, its root can be obtained by assembling the root graphs of its connected components. Therefore, when one speaks about the root of a graph H, it is usually assumed that H is connected (the problem is separable otherwise). The root graph of a connected graph, if it exists, is unique except for K_3, which has $K_{1,3}$ and also K_3 itself as roots (see Figure 1) [18].

As an illustrative example, Figure 2 shows a graph and its line graph, whose nodes are labeled with the corresponding edges in the root. Edges that incide in the same node of G produce complete subgraphs in $L(G)$, like the triangle— i.e., complete subgraph of three nodes— in Figure 2b induced by edges $(2, 5)$, $(4, 5)$ and $(5, 6)$, which share node 5 in G. In fact, we can identify a triangle in $L(G)$ for every node of G with degree 3, i.e., nodes 2, 4 and 5.

Krausz [9] proved that a graph H is line-invertible if and only if there is a partition of its edges that induces complete subgraphs such that no node belongs to more than two of the subgraphs. This characterization is strongly connected to the problem of covering the edge set of a graph with a minimum number of complete subgraphs, known as *clique covering* [13]. In the edge partition of Krausz, each subgraph corresponds to a node in the root graph, and a node $e = (u, v) \in H$ can belong at most to those of u and v. But not every node in the root has an associated complete subgraph in H. The following example illustrates Krausz's characterization.

Example 1. *Consider graphs G and $L(G)$ depicted in Figure 2. Edges that incide in the same node of G produce complete subgraphs in $L(G)$, like the triangle in Figure 2b induced by $(2, 5)$, $(4, 5)$ and $(5, 6)$, which share node 5 in G. Furthermore, we can identify a triangle in $L(G)$ for every node of G with degree 3, i.e., nodes 2, 4 and 5. The partition by Krausz would be $F_2 = \{((1, 2), (2, 3)), ((2, 3), (2, 5)), ((1, 2), (2, 5))\}$, $F_3 = \{((2, 3), (3, 4))\}$, $F_4 = \{((4, 5), (4, 6)), ((3, 4), (4, 6)), ((3, 4), (4, 5))\}$, $F_5 = \{((2, 5), (5, 6)), ((2, 5), (4, 5)), ((4, 5), (5, 6))\}$ and $F_6 = \{((4, 6), (5, 6))\}$. These correspond to nodes 2, 3, 4, 5 and 6 of G, respectively. Node 1 is not associated to any of the complete subgraphs because it has degree 1 in G.*

(a) CC graph G_{cc} (b) Krausz partition for G_{cc}

Figure 3: A Clark consistency graph and its Krausz partition

The characterization given by Krausz comes in a natural way from the definition of a line graph, but there are more. Another characterization, given by van Rooij and Wilf [17], uses the concept of *odd triangles*. A triangle in a graph is even if every node of the graph is adjacent to either 0 or 2 nodes of the triangle, and it is odd otherwise. The characterization described by van Rooij and Wilf states that a graph is line-invertible if and only if it does not contain $K_{1,3}$ as subgraph and, if $\{a, b, c\}$ and $\{a, b, d\}$ are odd triangles, then either $c = d$ or c and d are adjacent. Figure 1b illustrates $K_{1,3}$, which is clearly not a line graph by the characterization of Krausz. Beineke [2] found a new characterization through nine forbidden subgraphs. Besides these theoretical results, there exist algorithms to find the root graph in linear time, [10,14].

Since graphs are the media to encode information in diverse domains, problems and notions of Graph Theory usually find application in a whole range of disciplines. Line graphs are not an exception, being relevant in domains other than Graph Theory, such as percolation theory [19], community detection [1,6,11] or computational biology [7], as shown next.

2.2. Haplotype phasing

Haplotypes retain information about regions of diploid organisms genome that feature meaningful differences within a population. For each individual, such differences are summarized in a binary string, 1 codifies the least frequent allele within the population and 0 the most frequent one. Each of these binary strings defines a haplotype, and its entries are usually called sites. Two haplotypes (ancestors) generate a genotype (descendant), codified as a $\{0, 1, 2\}$ string, 0 and 1 encoding homozygous sites and 2 representing heterozygous sites. That is $h_1 \oplus h_2 = g$ where $h_1, h_2 \in \{0, 1\}^n$, $g \in \{0, 1, 2\}^n$ and $1 \oplus 1 = 1$, $0 \oplus 0 = 0$, $1 \oplus 0 = 2$ and $0 \oplus 1 = 2$. Haplotype phasing consists of estimating the haplotypes that resolve a set of genotypes, i.e. a set of ancestors that could produce a current population of genotypes. Two genotypes may have a common ancestor if their $\{0, 1, 2\}$ strings are consistent, i.e., given a site, either some of the strings has code 2 or both strings have the same code (0 or 1).

Consistency relations between genotypes can be represented by a graph called the Clark consistency (CC) graph [5]. Each genotype is represented by a node and adjacent nodes identify consistent genotypes. Figure 3a shows the CC graph for a set of five genotypes. Genotypes 210 and 021 are not adjacent in this graph because of their third sites— the former has a 0 and the latter has a 1, so they cannot have a common ancestor. If the CC graph is line-invertible, finding its root gives an estimation on the number of haplotypes needed to explain the genotypes. Indeed, when one checks for line-invertibility, a complete subgraph corresponds to a common node in the root, while here it is potentially interpreted as a common ancestor for the corresponding individuals. When the CC graph is line-invertible it is said to be *allelable*.

Further details about equivalence between line-invertible graphs and allelable graphs can be found in [7]. The following example illustrates the relation between the two concepts.

Example 2. *Consider Figure 3b, which depicts the same CC graph as Figure 3a. The different line traces give a partition of the edges of the CC graph in three groups satisfying Krausz's characterization of line graphs, which yields a root graph with three nodes. Since nodes/genotypes of the CC graph within each of the three groups are all adjacent/compatible with each other, the three nodes of the root are identified with their potential common ancestors. The number of nodes of the root graph gives in general a lower bound on the cardinality of the generating population of haplotypes. Figure 3b depicts the codes for the haplotypes that would correspond to the three ancestors. An ancestor that generates 202 and 021 must have code 001 and, similarly, 110 is the ancestor of 122 and 210. When it comes to the triangle induced by 102, 122 and 202, it is clear that their ancestor begins with 10. To determine its last site, we need to look at the other just-discovered haplotypes, 001 and 110. The following equations are to be verified:*

$$10x \oplus 001 = 202,$$

$$10x \oplus 110 = 122.$$

The former gives $x = 0$ and the latter $x = 1$. This is simply because we need more haplotypes to generate the current genotypes. For instance, 100, 101, 001 and 110 is a feasible set of ancestors. The minimum number of haplotypes needed to resolve the population is in fact 4. The reason why we get the lower bound 3 is that, when obtaining the root, we assume that every pair of genotypes that are neighbors share an ancestor, while edges in the CC graph only represent that the nodes can potentially, but not necessarily, be twinned.

The assumption that edges of the CC graph indicate twinned genotypes can lead to the impossibility of resolving the current population. Take the example of $K_{1,3}$ depicted on Figure 1b, which we know that is not allelable. The edges here mean that the central node may have an ancestor in common with each of the other three nodes of the graph, which at the same time are incompatible with each other. Since a node only have two ancestors, the central node will share an ancestor with at most two of its three neighbors, i.e., one of the edges is not "used". In general, when the CC graph is not allelable we have edges that do not represent true sharings, i.e., the corresponding genotypes do not have an ancestor in common. One solution to be able to estimate the haplotypes in these cases is to eliminate these false relations within nodes while trying to have the least impact on the graph topology. A combinatorial problem then arises, which is to find the minimum number of edges that we have to delete so that we obtain an allelable graph.

3. State of the art

Finding the root graph is related to another well-known feature of graphs, the coloring of their nodes or edges. Here, we formally state such relation and see how it has served to formulate EDPL with a mathematical programming model.

In general, a coloring of the nodes or edges of a graph $G = (V, E)$ is a mapping $\varphi : V \to \mathcal{C}$ or $\varphi : E \to \mathcal{C}$ that assigns one color from the set $\mathcal{C}$ to every node or edge in G. It can be extended to assigning more than one color, arising the so-called k-fold coloring or multi-coloring (see

for instance [4]). In most applications of node/edge coloring, adjacent nodes are constrained to have different colors and the same happens for edges that are incident to a common node. However, this will not be the case here.

The following two definitions serve to establish the relation between line-invertible graphs and graph coloring of edges and nodes, which will be ultimately useful to state EDPL in Lemma 1.

Definition 1 (*Line-graph edge coloring*)**.** *Let $G = (V, E)$ be an undirected graph. We will say that $\varphi : E \to \mathcal{C}$ is an edge coloring of a line-graph if the two following conditions are met:*

(E1) Each subset of edges having the same color $c \in \mathcal{C}$, $E_c := \{e \in E : \varphi(e) = c\}$, induces a complete subgraph. That is, $G_c[E_c] := (V_c, E_c)$, with $V_c = \{v \in V : \exists e \in E_c \text{ s.t. } v \in e\}$ is complete.

(E2) For all $v \in V$ there exist at most two colors, c_1 and c_2, such that $v \in V_{c_1} \cap V_{c_2}$.

If there exists an edge coloring for G that satisfies (E1) and (E2) then G is line-invertible. In effect, it is straightforward to see that these conditions produce an edge partition as that proposed by Krausz in [9] to characterize line-invertible graphs. Conversely, if G is line-invertible, one can obtain an edge coloring satisfying (E1) and (E2) just by assigning different colors to the groups of edges in Krausz's partition.

A line-graph edge coloring is equivalent to a vertex coloring as follows.

Definition 2 (*Line-graph node coloring*)**.** *Let $G = (V, E)$ be an undirected graph. We will say that an assignment of two colors to every node of G, $\varphi : V \to \mathcal{C} \times \mathcal{C}$, is a node coloring of a line graph if the following conditions are met:*

(N1) $\varphi = (\varphi_1, \varphi_2)$ and $\varphi_1(v) \neq \varphi_2(v)$ $\forall v \in V$.

(N2) For all $(u, v) \in E$, $|\{\varphi_1(u), \varphi_2(u)\} \cap \{\varphi_1(v), \varphi_2(v)\}| = 1$.

(N3) For all $(u, v) \notin E$, $|\{\varphi_1(u), \varphi_2(u)\} \cap \{\varphi_1(v), \varphi_2(v)\}| = 0$.

It is easy to check that this node coloring is equivalent to the desired edge coloring. Indeed, (N1) states that the colors given to a node are different; (N2) would allow an edge coloring by assigning to every edge (u, v) the color that u and v have in common; finally, (N3) ensures that such edge coloring induces the complete subgraphs of (E1). Conversely, if an edge coloring satisfies (E1) and (E2), a line-graph node coloring is easily derived by assigning each node the colors of the two complete subgraphs to which it belongs.

As a consequence, Definition 2 gives also a characterization of line graphs. This characterization was previously stated in [7, Lemma 2.1], by using a different argument. We next formally state EDPL in terms of the coloring characterization given by Definitions 1 and 2.

Lemma 1. *Let $G = (V, E)$ be an undirected graph. EDPL is equivalent to finding $E' \subseteq E$ such that the induced graph $G[E \setminus E']$ admits a line-graph node/edge coloring and $|E'|$ is minimized.*

Halldórsson et al. [7] presented an ILP formulation to solve EDPL. The formulation consists of finding a node coloring that satisfies (N1), (N2) and (N3), minimizing the number of pairs of nodes u and v, $(u, v) \in E$, that do not verify (N2), i.e., the deleted edges.

The authors took inspiration from Campêlo's representative formulation for the vertex coloring problem [3] in order to eliminate equivalent (symmetric) solutions. In the classic vertex coloring, each node receives one color. The idea of the representative formulation is to identify colors with nodes so that if node i is colored with i it is said to be a representative. An ordering of the nodes is considered so that every node receives the color of one of its preceding representatives or it is a representative itself. Halldórsson et al. explained their idea for symmetry breaking as follows. Each node i "owns" two colors, $2i-1$ and $2i$. Every node i can be assigned either some color(s) owned by some neighbor(s) k such that $k < i$, and/or some of its own colors, $2i-1$ and $2i$. A node will be assigned a color owned by one of its neighbors if and only if this neighbor is using the color in question. This means that k is always the smallest node among those colored with $2k-1$ or $2k$, if any. On the other hand, if one has to draw on colors $2i$ and $2i-1$ to color node i, $2i-1$ will be the first one to be taken and $2i$ will be used only if $2i-1$ has already been assigned to i.

Before presenting the formulation in [7], we define the left-neighborhoods of a node i as $N^-(i) = \{k \in V : k < i,\ (i,k) \in E\}$ and $N^-[i] = N^-(i) \cup \{i\}$, for all $i \in V$. The precedence relation between nodes could be any order within V, but the reader can assume that $V = \{1, \ldots, n\}$ and $<$ is common numerical order. For convenience in the exposition we will use the following notations:

$$\mathcal{C}(i) = \{2k-1, 2k \ :\ k \in N^-[i]\}, \qquad \mathcal{C}(i,j) = \mathcal{C}(i) \cap \mathcal{C}(j),$$

which identify the colors that can be assigned to node i and those that can be shared by i and j, respectively. We will also denote with $\varphi : V \to \mathcal{C} \times \mathcal{C}$ a coloring function, $\varphi = (\varphi_1, \varphi_2)$.

Halldórsson et al. used three families of variables: y to gather the information about the two colors given to each node; s to identify when the two end-nodes of an edge have the same color (they are products of y variables); and d that account for the number of edges deleted. These binary variables are formally defined as follows

$y_{ir} = 1$ iff $r \in \{\varphi_1(i), \varphi_2(i)\}$, $i \in V$, $r \in \mathcal{C}(i)$,
$s_{ijr} = 1$ iff $r \in \{\varphi_1(i), \varphi_2(i)\} \cap \{\varphi_1(j), \varphi_2(j)\}$, $(i,j) \in E$, $r \in \mathcal{C}(i,j)$, and
$d_{ij} = 1$ iff edge (i,j) is deleted, $(i,j) \in E$.

The formulation in [7] reads:

$$
\begin{aligned}
\text{(H)}\ \min \quad & \sum_{(i,j)\in E, i<j} d_{ij} & & \\
\text{s.t.} \quad & \sum_{r\in\mathcal{C}(i)} y_{ir} = 2 & \forall i \in V & (1)\\
& y_{ir} + y_{jr} \le 1 & \forall (i,j) \notin E, \forall r \in \mathcal{C}(i,j) & (2)\\
& \sum_{r\in\mathcal{C}(i,j)} s_{ijr} = 1 - d_{ij} & \forall (i,j) \in E & (3)\\
& y_{ir} + y_{jr} - 1 \le s_{ijr} & \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j) & (4)\\
& s_{ijr} \le y_{ir} & \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j) & (5)\\
& s_{ijr} \le y_{jr} & \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j) & (6)\\
& y_{i,2i} \le y_{i,2i-1} & \forall i \in V & (7)\\
& y_{i,2k-1} \le y_{k,2k-1} & \forall i \in V, \forall k \in N^-(i) & (8)\\
& y_{i,2k} \le y_{k,2k} & \forall i \in V, \forall k \in N^-(i) & (9)\\
& y_{ir}, s_{ijr'}, d_{ij} \in \{0,1\} & \forall i \in V, \forall r \in \mathcal{C}(i), \forall (i,j) \in E, \forall r' \in \mathcal{C}(i,j). &
\end{aligned}
$$

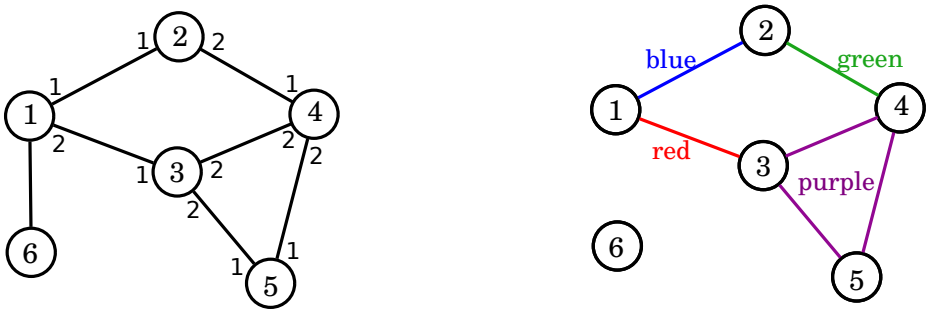
Constraints (1) impose that each node is assigned two colors, and (2) guarantee that non-adjacent nodes are given distinct colors. Constraints (3) say that if two adjacent nodes are assigned different colors the edge must be deleted and, at the same time, guarantee that two adjacent nodes share one color at most. Families (4)-(6) are standard "product constraints" that guarantee that $s_{ijr} = y_{ir}y_{jr}$ for every integer solution. Due to these constraints, $s_{ijr} \in \{0,1\}$ could be relaxed to $s_{ijr} \ge 0$. Finally, constraints (7)-(9) are the symmetry breaking constraints inspired by [3].

4. A new approach

As opposed to model (H), the formulation we propose is not based on a direct translation of Definitions 1 or 2, but a line-graph node/edge coloring can be obtained as a byproduct of our solution. We propose to use only two families of variables: x, with four indices, and d, which are identical to that of (H). With variables x, we will represent those pairs of adjacent nodes that share the same color, distinguishing which function, either φ_1 or φ_2, gives the color to each node. More precisely, our variables are defined as follows:

$$
\begin{aligned}
x_{ijab} &= 1 \text{ if } \varphi_a(i) = \varphi_b(j),\ (i,j) \in E,\ a,b \in \{1,2\}, \text{ and}\\
d_{ij} &= 1 \text{ if } (i,j) \in E \text{ is deleted.}
\end{aligned}
$$

With these variables, colors are not explicitly stated. If for instance $x_{ij21} = 1$ we know that i and j share a color, which corresponds to $\varphi_2(i)$ and $\varphi_1(j)$, but we do not know if such color is blue. Nonetheless, these variables are enough to formulate the problem. Call Γ the sets of triangles in G, $\Gamma = \{\{i,j,k\} \subseteq V :\ i<j<k\ ;\ (i,j),(i,k),(j,k) \in E\}$. The following



(a) Solution to (M') (b) A line-graph edge-coloring for EDPL

Figure 4: A graph and optimal solution with optimal value equal to 1

mathematical program gives a formulation for EDPL:

$$\text{(M')} \quad \min \quad \sum_{(i,j)\in E} d_{ij}$$

$$\text{s.t.} \quad x_{ija1} + x_{ija2} + x_{ika1} + x_{ika2} \leq 1 \quad \forall (i,j),(i,k) \in E, \; j \neq k, (j,k) \notin E, \; a \in \{1,2\} \tag{10}$$

$$x_{ijba} \geq x_{jkac} + x_{ikbc} - 1 \quad \forall \{i,j,k\} \in \Gamma, \; a,b,c \in \{1,2\} \tag{11}$$

$$x_{ikbc} \geq x_{jkac} + x_{ijba} - 1 \quad \forall \{i,j,k\} \in \Gamma, \; a,b,c \in \{1,2\} \tag{12}$$

$$x_{jkac} \geq x_{ikbc} + x_{ijba} - 1 \quad \forall \{i,j,k\} \in \Gamma, \; a,b,c \in \{1,2\} \tag{13}$$

$$\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} = 1 - d_{ij} \quad \forall (i,j) \in E \tag{14}$$

$$x_{ijab}, d_{ij} \in \{0,1\} \quad \forall (i,j) \in E, a,b \in \{1,2\}.$$

Constraints (10) guarantee that two non-adjacent nodes do not share a color. In effect, these constraints read: if i has two non-adjacent neighbors j and k, it may share a color with one of them, but not the same one with the two. Constraints (11)-(13) impose the transitivity of color sharing and are defined for every subset of nodes inducing a triangle. Take, for instance, (11). These constraints are active when the right-hand side is 1, that is, when $x_{jkac} = x_{ikbc} = 1$. If this is the case, then (11) forces that $x_{ijba} = 1$. Note that $x_{jkac} = 1$ means $\varphi_a(j) = \varphi_c(k)$ and $x_{ikbc} = 1$ means $\varphi_b(i) = \varphi_c(k)$, while $x_{ijba} = 1$ just impose what follows by transitivity, $\varphi_b(i) = \varphi_a(j)$. Constraints (12) and (13) are analogous. Finally, (14) ensure that (i,j) is removed if i and j do not share any color and that they can share one color at most.

The following example illustrates the formulation.

Example 3. *Consider the graph depicted in Figure 4a. Every edge (i,j) is labeled with the indices a, b of the corresponding x_{ijab}-variable that takes value 1 in a feasible solution of (M'). For instance, according to the figure, $x_{1211} = 1$ and $x_{1321} = 1$. This means that colors $\varphi_1(1)$ and $\varphi_1(2)$, given to nodes 1 and 2 respectively, coincide and so do $\varphi_2(1)$ and $\varphi_1(3)$, which were given to nodes 1 and 3. Constraints (10) when $i = 1$ and $j = 6$ forbid node 1 to share color $\varphi_1(1)$ with*

*node 6 when $k = 2$, since $(2, 6) \notin E$. Similarly, when $k = 3$, constraints (10) avoid that $\varphi_2(1)$ coincides with some of the colors given to node 6. As a result, $d_{16} = 1$ in the depicted solution. Moreover, in this example, 1 is the optimum of the problem. This is, indeed, due to fact that 1 has three neighbors that induce an independent set in the graph (i.e., none of them are adjacent to each other). On the other hand, color sharing between nodes 3, 4 and 5 is consistent thanks to constraints (11)-(13). Note that, even if not explicitly stated, this feasible solution yields a line-graph node coloring. To obtain this coloring, we start from one node, say 1, and suppose that $\varphi_1(1)$ =*blue *and $\varphi_2(1)$ =*red*. Then, since $x_{1211} = 1$, node 2 has color blue, $\varphi_1(2)$ =*blue, and another color, say green, $\varphi_2(2)$ =*green*. On the other hand, since $x_{1321} = 1$, $\varphi_1(3)$ =*red, and we can assume $\varphi_2(3)$ =*purple*. Now, $x_{2421} = 1$ and $x_{3422} = 1$ force $\varphi_1(4)$ =*green *and $\varphi_2(4)$ =*purple*. Finally, $x_{4521} = 1$ and $x_{3521} = 1$ imply that $\varphi_1(5)$ =*purple*. Note that $\varphi_2(5)$ is not imposed by any of the neighbors of node 5 and then we can assign to it any new color, for instance, white (in fact this color is not needed if we think about edge-coloring). Figure 4b illustrates the resulting coloring and edge deletion.*

In general, one can always obtain a line-graph coloring for a graph from a feasible solution to (M'). Algorithm 1 states a procedure to derive such a coloring. Proposition 1 ensures the correctness of the algoritm. Let us suppose that $(\bar{x}, \bar{d})$ is a feasible solution to (M') over a graph G that, when used as input of Algorithm 1, produces a node coloring $\bar{\varphi} = (\bar{\varphi}_1, \bar{\varphi}_2)$. The following lemmas state some properties of $\bar{\varphi}$ that will be useful in the proof of Proposition 1.

Lemma 2. *If $\bar{\varphi}_a(i) = \bar{\varphi}_b(j)$ then $(i, j) \in E$. Moreover, $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$.*

Proof. If $\bar{\varphi}_a(i) = \bar{\varphi}_b(j)$ they have been set in the same call to the procedure *explore* of Algorithm 1, say *explore*(k, c). If $k = i$ or $k = j$, it has to be $c = a$ or $c = b$, respectively. Due to the algorithm construction, this would imply $(i, j) \in E$ and also $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$. Otherwise, if $k \neq i$ and $k \neq j$, due to the steps in the procedure *explore*, we can assume there exist edges $(i, k) \in E$ and $(j, k) \in E$ such that $\bar{x}_{kica} = 1$ and $\bar{x}_{kjcb} = 1$ without loss of generality (the proof will be analogous if we suppose $\bar{x}_{ikac} = 1$ or $\bar{x}_{jkbc} = 1$). Constraints (10) then imply that $(i, j) \in E$ and (13) yield $\bar{x}_{ijab} = 1$.

□

Lemma 3. *Every $\bar{\varphi}_1(i)$ and $\bar{\varphi}_2(i)$ are modified by Algorithm 1 only once, for all $i \in V$. In particular, if $\bar{\varphi}_a(i)$ is modified in a call to the procedure* explore*, then, before such assignment is made, it was $\bar{\varphi}_a(i) = 0$.*

Proof. Let $i \in V$ and $a \in \{1, 2\}$ be a node and an index of the coloring function $\bar{\varphi}$. Consider the call to *explore*(i, a) inside Algorithm 1. We distinguish two cases.

1. If $\bar{\varphi}_a(i) = 0$, all the neighbors j of i such that $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$ modify their color function $\bar{\varphi}_b(j)$. Consequently, $\bar{\varphi}_a(i)$ cannot be modified in subsequent calls to *explore* for any of these pairs (j, b).

2. Otherwise, if $\bar{\varphi}_a(i) > 0$, *explore*(i, a) does not modify any value of the coloring function $\bar{\varphi}$. In this case, $\bar{\varphi}_a(i)$ must have been modified in a call to *explore*(i', a') for some $i' \in V$ and $a' \in \{1, 2\}$ such that $(i, i') \in E$ and $\bar{x}_{ii'aa'} = 1$ or $\bar{x}_{i'ia'a} = 1$. We fix then i' and suppose that *explore*(i', a') is the call in which $\bar{\varphi}_a(i)$ was modified for the first time. We have to prove that this is the only call to *explore* that modifies $\bar{\varphi}_a(i)$. To do so, let us

Algorithm 1 Line-graph node coloring from a feasible solution to (M')

Input $G = (V, E)$: an undirected graph.
$(\bar{x}, \bar{d})$: feasible solution to (M') over graph G.
Output $\bar{\varphi} = (\bar{\varphi}_1, \bar{\varphi}_2)$: a node coloring for G.

Step 1 Let *col* be the color counter, with initial value $col := 0$
Let $\bar{\varphi}(i) := (0, 0)\ \forall i \in V$
Step 2 **For all** $i \in V$ **do**:
For all $a \in \{1, 2\}$ **do**:
explore (i, a)
End-For all
End-For all
Step 3 Return $\bar{\varphi}$.

Procedure explore(i, a)
If $\bar{\varphi}_a(i) = 0$ **then**:
Let $col := col + 1$, $\bar{\varphi}_a(i) := col$
For all $j \in V$ such that $(i, j) \in E$ and $b \in \{1, 2\}$ **do**:
If $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$ * **then** let $\bar{\varphi}_b(j) := col$ **End-If**
End-For all
End-If
End-Procedure

* Note that only one of the two variables, either $x_{ijab} = 1$ or $x_{jiba} = 1$, exists in the model (E is a set of unordered pairs of nodes)

consider any other call *explore*(i'', a''), with $i'' \in V$ and $a'' \in \{1,2\}$ such that $(i,i'') \in E$. If $\bar{\varphi}_a(i)$ is modified within this call, it has to be $\bar{x}_{ii''aa''} = 1$ or $\bar{x}_{i''ia''a} = 1$, which implies $(i',i'') \in E$ due to constraints (10). Moreover, transitivity constraints (11)-(13) applied to the triangle $\{i,i',i''\}$ yield $\bar{x}_{i'i''a'a''} = 1$ or $\bar{x}_{i''i'a''a'} = 1$. But this means that $\bar{\varphi}_{a''}(i'')$ must have been modified also in the call to *explore*(i', a'). Therefore, when *explore*(i'', a'') is executed (this happens later on from the call to *explore*(i', a')), $\bar{\varphi}_{a''}(i'') > 0$ and the call returns without modifying any color function.

□

Proposition 1. *Let $(\bar{x},\bar{d})$ be a feasible solution to (M'). The coloring $\bar{\varphi} = (\bar{\varphi}_1,\bar{\varphi}_2)$ built with Algorithm 1 satisfies properties (N1)-(N3) for the graph $G = (V,E')$ where $E' := E \setminus \{(i,j) : \bar{d}_{ij} = 1\}$.*

Proof.

(N1) If $\bar{\varphi}_1(i) = \bar{\varphi}_2(i) = col$ for some $i \in V$ then col must have been assigned to $\bar{\varphi}_1(i)$ and $\bar{\varphi}_2(i)$ in the same call to procedure *explore*. By the algorithm's construction, this would be only possible if there exists i' such that $(i,i') \in E$ and $\bar{x}_{ii'1a} = \bar{x}_{ii'2a}$ (or $\bar{x}_{i'ia1} = \bar{x}_{i'ia2}$), which contradicts (14).

(N2) Let $(i,j) \in E'$. Due to (14), there exist $a,b \in \{1,2\}$ such that $\bar{x}_{ijab} = 1$. We will prove the following two statements: (i) i and j share at least one color and (ii) i and j cannot have the same two colors.

(i) If $\bar{\varphi}_a(i) = 0$ when *explore*(i, a) is executed, then the algorithm will assign the same color to $\bar{\varphi}_a(i)$ and $\bar{\varphi}_b(j)$. The same happens if $\bar{\varphi}_b(j) = 0$ when (j, b) is explored. Moreover, Lemma 3 guarantees that these colors will not be modified later on by the algorithm. Therefore, let us suppose that $\bar{\varphi}_a(i) > 0$, resp. $\bar{\varphi}_b(j) > 0$, when *explore*(i, a), resp. *explore*(j, b), is executed. Then, $\bar{\varphi}_a(i)$ was modified in a call to the procedure *explore*(i',a') for i' adjacent to i and $a' \in \{1,2\}$ such that $\bar{x}_{i'ia'a} = 1$ or $\bar{x}_{ii'aa'} = 1$. Similarly, $\bar{\varphi}_b(j)$ was modified in a call to *explore*(j',b') for j' adjacent to j and $b' \in \{1,2\}$ such that $\bar{x}_{j'jb'b} = 1$ or $\bar{x}_{jj'bb'} = 1$. Due to constraints (10), since i shares a with both i' and j, it has to be $(i',j) \in E$. Similarly, since j shares b with both i and j', $(i,j') \in E$. Moreover, transitivity constraints (11)-(13) applied on the triangles $\{i',i,j\}$ and $\{i,j,j'\}$ imply that $\bar{x}_{i'ja'b} = 1$ (or $\bar{x}_{j'iba'} = 1$) and $\bar{x}_{ij'ab'} = 1$ (or $\bar{x}_{j'ib'a} = 1$), respectively. This means that $\bar{\varphi}_b(j)$ was modified during the call to *explore*(i',a') and that $\bar{\varphi}_a(i)$ was modified in *explore*(j', b'). But due to Lemma 3, every color assignment is modified only once, which yields $i' = j'$, $a' = b'$ and thus $\bar{\varphi}_a(i) = \bar{\varphi}_b(j)$.

(ii) If i and j have the same two colors, due to Lemma 2, it will be $\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} = 2$, which contradicts (14).

(N3) Let (i,j) be such that $(i,j) \notin E'$ and suppose (proof by contradiction) that i and j share a color, i.e., $\bar{\varphi}_a(i) = \bar{\varphi}_b(j) = col$ for some color col and $a,b \in \{1,2\}$. Due to Lemma 2, $(i,j) \in E$ and either $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$. Since $(i,j) \notin E'$ by hypothesis, it has to be

$\bar{d}_{ij} = 1$. This implies, from constraints (14), that $\sum_{a=1}^{2}\sum_{b=1}^{2} \bar{x}_{ijab} = 0$, which contradicts the fact that either $\bar{x}_{ijab} = 1$ or $\bar{x}_{jiba} = 1$.

□

Note that variables d_{ij} can be eliminated from (M'). In effect, using constraints (14), d_{ij} can be removed from the objective, which will be equivalent to maximizing the sum $\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab}$. The new objective would stand for the number of edges that remain in the graph, instead of the number of deleted edges. After that, replacing (14) with $\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \leq 1$ will complete the transformation, obtaining an equivalent formulation without variables d_{ij}, (M). Symmetry breaking constraints can also be added to (M), but we will leave the analysis for Section 4.1.

$$
\begin{aligned}
\text{(M)}\ \max \quad & \sum_{(i,j)\in E}\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \\
\text{s.t.} \quad & (10) - (13) \\
& \sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \leq 1 \qquad \forall (i,j) \in E \qquad (15) \\
& x_{ijab} \in \{0,1\} \qquad \forall (i,j) \in E, a, b \in \{1,2\}.
\end{aligned}
$$

The formulation can still be improved, for instance, by adding symmetry breaking constraints. But before analyzing this and other enhancements in the next section, we shall compare the size of (H) and (M).

Solutions to (H) provide a line-graph node coloring for EDPL, while variables of (M) only account for the way colors are shared in such a coloring. Because of this fact, (M) has fewer variables than (H). If one denotes by t the number of triangles in G, $t = |\Gamma|$, variables in each formulation can be easily compared, as follows. Since color candidates for one node are its two colors and the two colors of its neighbors, (H) has $2n + 2|E|$ y-variables. A similar reasoning serves to count the members of the family of s-variables, which are $2|E| + 2t$ in total. On the one hand, note that for each edge (i, j) with $i < j$ there exist $s_{ij,2i-1}$ and $s_{ij,2i}$, which accounts for $2|E|$ variables of the family. Other than $2i - 1$ and $2i$, nodes i and j could share one of the two colors of a common neighbor, say node k, with $k < i$. That is, $s_{ij,2k-1}$ and $s_{ij,2k}$ exist for each $\{k, i, j\} \in \Gamma$, which accounts for the rest of the variables in the family, $2t$ in total. Finally, we avoid counting d-variables in (H) since they can be eliminated by using a similar argument than that used for (M'). In summary, (H) has $2n + 4|E| + 2t$ variables, while (M) only uses $4|E|$ variables.

As for the number of constraints, let t' be the number of triplets $\{i, j, k\}$ such that $(i, j), (i, k) \in E$ and $(j, k) \notin E$. Regarding (H), it is straightforward that there are n constraints in (1). The second family, (2), is defined in terms of triplets $\{k, i, j\}$ such that $(k, i), (k, j) \in E$ and $(i, j) \notin E$. In effect, $2k-1$ and $2k$ stand for the only possible values of r in (2) whenever $k < i$ and $k < j$. Using this observation, one can check that there are at most $2t'$ constraints in (2). Family (3) has $|E|$ members; (4)- (6) are defined for each valid subscript ijr of variables s, so they account for $6|E| + 6t$ constraints; finally, (7), (8) and (9) gather n, $|E|$ and $|E|$ constraints respectively. Consequently, formulation (H) has $n + 2t' + |E| + 6|E| + 6t + n + 2|E| = 2n + 2t' + 6t + 9|E|$

constraints at most (or $n + 2t' + 6t + 7|E|$ if symmetry breaking constraints are excluded). On the other hand, formulation (M) has $2t'$ constraints of type (10). Constraints in (11) account for 2^3 (the different combinations of a, b and c) times t, and the same happens for (12) and (13). Finally, (15) is composed by $|E|$ constraints. In sum, (M) has $2t'+24t+|E|$ constraints in total. It is not straightforward to analyze which formulation, (M) or (H), has fewer constraints. Since they are dependent on the structure of the graph G in question, both situations can occur. Consider for instance a star, that is, a graph consisting of $n + 1$ nodes, one of them adjacent to the rest, which makes $|E| = n$. In this case $t = 0$ and $t' = \binom{|E|}{2}$. This implies that (M) has $2t' + 24t + |E| = n^2$ constraints, while (H) has $n + 2t' + 6t + 7|E| = n^2 + 7n$ if the center of the star corresponds to node 1. Conversely, if one takes the wheel with $n + 1$ nodes, which is made of a cycle of n nodes plus a central node adjacent to the rest, things change. In this case, $|E| = 2n$ and $t = n$. Formulation (M) has then $2t' + 24t + |E| = 2t' + 26n$ constraints, while (H) has $n + 2t' + 6t + 7|E| = 2t' + 21n$ at most.

4.1. Valid inequalities

After studying the structure of (M), we have derived several valid inequalities that we present here in different groups. Even if colors do not appear explicitly in our formulation, (M) still has symmetric solutions, since color positions of a node are interchangeable. The first family of inequalities tries to avoid these symmetric solutions. The second one represents the fact that nodes of independent sets cannot have the same color. Another family of inequalities considers the case in which a deletion of an edge produces an independent set of 3 nodes. Finally, the last one is related to the transitivity of color sharing. We will denote $N(i)$ the neighborhood of node i, $N(i) = \{j \in V : (i, j) \in E\}$, and $N[i]$ its closed neighborhood, $N[i] = N(i) \cup \{i\}$.

4.1.1. Symmetry breaking inequalities

Variables x_{ijab} produce symmetric solutions by their very nature: we can always swap colors positions of a node and update the variables to be consistent with the change without altering the objective value. One way of forbidding such a change is to enforce the position of the colors for some nodes. More specifically, given a node i, we pay attention to its neighbor with smallest index, j_1^i. Since i has two colors to share with its neighbors, we can assume w.l.o.g. that j_1^i is not the one sharing $\varphi_2(i)$. That is, the following variables can be set to zero:

$$x_{ij_1^i 21} = x_{ij_1^i 22} = 0 \qquad \forall i, j_1^i := \min\{j : j \in N(i)\}. \tag{16}$$

Nevertheless, if (i, j_1^i) is deleted, $x_{ij_1^i ab} = 0\ \forall a, b \in \{1, 2\}$ and the previous equalities are unnecessary. We can consider now the second neighbor of i with smallest index, say j_2^i. Similarly as before, we can assume that j_2^i does not share $\varphi_2(i)$ if (i, j_1^i) is deleted:

$$x_{ij_2^i 21} + x_{ij_2^i 22} \leq x_{ij_1^i 11} + x_{ij_1^i 12} \qquad \forall i, j_2^i := \min\{j : j_2^i \neq j_1^i,\ (i, j) \in E\}. \tag{17}$$

If $x_{ij_1^i 11} = 1$ or $x_{ij_1^i 12} = 1$ (17) trivially stands. Otherwise, $x_{ij_1^i 11} + x_{ij_1^i 12} = 0$, which together with (16) implies that i and j_1^i do not share any color. Since i and j_1^i are adjacent, this means that the edge will be deleted. Then we can assume that i does not share $\varphi_2(i)$ with j_2^i, i.e., $x_{ij_2^i 21} + x_{ij_2^i 22} = 0$ and hence (17) is valid. We can continue the process with the third, fourth...

neighbor of i with smallest index. The following set of inequalities gathers all the cases in a compact way:

$$x_{ij21} + x_{ij22} \leq \sum_{k<j:(i,k)\in E} (x_{ik11} + x_{ik12}) \qquad \forall (i,j) \in E. \tag{18}$$

These inequalities read: "if i does not share color $\varphi_1(i)$ with any neighbor $k < j$ then it does not share $\varphi_2(i)$ with j either". Inequalities (16) and (17) are (18) with $j = j_1^i$ and j_2^i respectively. Note that in the former case the summation on the right-hand side of (18) becomes empty and the inequality then reads $x_{ij_1^i 21} + x_{ij_1^i 22} \leq 0$.

4.1.2. Independent set inequalities

Because of the definition of the problem, any two non-adjacent nodes do not share a color. This means that, given a color and two non-adjacent nodes, one of them at most is given the color. More generally, given a color and an *independent set* (recall that this is a set of nodes that are non-adjacent to each other), one of the nodes in the set at most will have the color.

Since variables of (M) are only defined for adjacent nodes, if two non-adjacent nodes were to share a color, this would happen through transitivity. That is, if $(j,k) \notin E$ and i sharing its color in position a with both j and k existed, then j and k would have the same color. This is precisely what (10) forbid. To generalize (10), we need to look at independent sets inside the neighborhood of i, $N(i)$:

$$\sum_{j\in S} (x_{ija1} + x_{ija2}) \leq 1 \qquad \forall i, \forall S \subseteq N(i), S \in \mathcal{P}_G, \forall a \in \{1,2\}, \tag{19}$$

where $\mathcal{P}_G$ is the collection of all the independent sets of G. These are set-packing constraints — left hand side is a linear combination of the x-variables with coefficients 0 or 1 and the coefficient on the right-hand side is 1— that will be stronger with the size of S. Node i and nodes in S induce a star, in which i is the internal node. When $|S| = 3$ we will refer to this induced subgraph, which is, in fact, $K_{1,3}$, as *fork*.

4.1.3. Quasi-fork inequalities

Edge deletions can yield substructures that were not originally in graph G, like new independent sets. If this occurred, inequalities (19) would be valid for the nodes in question. In this section we adapt valid inequalities (19) to cover such cases.

For all $\{i,j,k\} \in \Gamma$ and $\ell \in N(i)$, $\ell \notin N[j]$ and $\ell \notin N[k]$, the following inequalities hold:

$$\sum_{a=1}^{2}\sum_{b=1}^{2} (x_{ijab} + x_{ikab} + x_{i\ell ab} - x_{jkab}) \leq 2, \tag{20}$$

$$x_{ija1} + x_{ija2} + x_{ika1} + x_{ika2} + x_{i\ell a1} + x_{i\ell a2} - \sum_{b=1}^{2}\sum_{c=1}^{2} x_{jkbc} \leq 1, \quad \forall a \in \{1,2\}. \tag{21}$$

Nodes 1, 3, 4 and 5 of the graph on Figure 4a illustrate the situation that we are considering. Note that, if $(4,5)$ is deleted, these nodes induce a fork. In general, we will call the subgraph induced by the triangle $\{i,j,k\}$ and ℓ quasi-fork. Inequalities (20) and (21) are active when $\sum_{a=1}^{2}\sum_{b=1}^{2} x_{jkab} = 0$, that is, when (j,k) is removed. If this were the case, taking $S = \{j,k,\ell\}$, we have that (21) would be (19), and (20) would be the sum over a of (19).

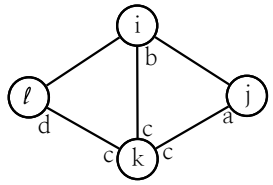


Figure 5: Subgraph of Inequalities (22)

4.1.4. Adjacent triangles inequalities

These inequalities are inspired by transitivity inequalities (11)-(13) of (M), but here we consider two triangles instead of just one. Suppose that $\{i, j, k\}$ and $\{i, \ell, k\}$ are two elements in Γ such that $(j, \ell) \notin E$. We say that they are adjacent triangles because they share edge (i, k) (see Figure 5). The following inequalities are valid for (M):

$$x_{ijba} + x_{i\ell bd} \geq x_{ikbc} + x_{jkac} + x_{k\ell cd} - 1, \quad \forall a, b, c, d \in \{1, 2\}, \tag{22}$$

$$x_{jkab} + x_{k\ell bd} \geq x_{ikcb} + x_{ijca} + x_{i\ell cd} - 1, \quad \forall a, b, c, d \in \{1, 2\}. \tag{23}$$

Inequalities (22) are illustrated in Figure 5. Since $(j, \ell) \notin E$, each instance of (22) has right-hand side at most 1. This bound is attained either when $x_{ikbc} + x_{jkac} = 2$ or when $x_{ikbc} + x_{k\ell cd} = 2$. Due to transitivity, this would imply either that $x_{ijba} = 1$ in the first case or $x_{i\ell bd} = 1$ in the second. Then, if the right-hand side of (22) is 1 it has to be $x_{ijba} + x_{i\ell bd} \geq 1$ and thus the inequality is valid. Inequalities (23) are analogous with the difference that i and k are turned around.

5. An alternative approach

After doing some computational experiments, we realized that none of the linear relaxations of (M) and (H) is stronger than the other, even after including inequalities of Section 4.1.1 in (M) or/and inequalities (19) with $|S| = 3$. That is, if v(HLP) and v(MLP) are respectively the optimal values of the corresponding formulations after relaxing the integrality of the variables, we have v(HLP)$<$ v(MLP) for some instances and v(HLP)$>$ v(MLP) for some others. In view of this, we propose a family of formulations, (HM), which combines variables y and s of (H) with variables x of (M). Our aim is to produce formulations with stronger linear relaxations and, with this, to be able to reduce the computational times. Actually, we propose three different versions, (HMa), (HMb) and (HMc), the last two being the result of removing some constraints from (HMa).

Before presenting the family (HM), we introduce some constraints that establish links between the variables from (H) and (M). The aim of these linking constraints is to enhance the family (HM).

5.1. Linking constraints

Variables x distinguish between the two colors given to a node, i.e., between the components φ_1 and φ_2 of $\varphi = (\varphi_1, \varphi_2)$. Hence, to be able to state the links between x-variables in (M) and y-variables in (H), we need to establish which color is given by the first and second component of φ. We will assume that if a node i is given colors r_1 and r_2 with $r_1 < r_2$ then $\varphi_1(i) = r_1$ and $\varphi_2(i) = r_2$. In terms of x variables, if $(i, j) \in E$, $y_{i,r_1} = y_{i,r_2} = 1$ and $y_{j,r_1} = y_{j,r_3} = 1$, $r_2 \neq r_3$, (i and j share color r_1) this means that:

(i) $x_{ij11} = 1$ iff $r_1 < r_2$ and $r_1 < r_3$,

(ii) $x_{ij22} = 1$ iff $r_1 > r_2$ and $r_1 > r_3$,

(iii) $x_{ij12} = 1$ iff $r_1 < r_2$ and $r_1 > r_3$,

(iv) $x_{ij21} = 1$ iff $r_1 > r_2$ and $r_1 < r_3$.

Based on this assumption, we develop a set of linking constraints between the variables of (H) and (M). Namely, we devise four families of linking constraints that ensure that the four statements above are satisfied.

To begin with, the following constraints correspond to (i) and (ii)

$$s_{ijr} - \sum_{\substack{r' \in \mathcal{C}(i) \\ r' < r}} y_{ir'} - \sum_{\substack{r' \in \mathcal{C}(j) \\ r' < r}} y_{jr'} \leq x_{ij11} \quad \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j) \tag{24}$$

$$s_{ijr} - \sum_{\substack{r' \in \mathcal{C}(i) \\ r' > r}} y_{ir'} - \sum_{\substack{r' \in \mathcal{C}(j) \\ r' > r}} y_{jr'} \leq x_{ij22} \quad \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j). \tag{25}$$

A constraint in (24) or (25) is active when its left-hand side is 1. In the case of (24), this occurs when nodes i and j share color r ($s_{ijr} = 1$) and neither i nor j is assigned a color smaller than r ($\sum_{r' \in \mathcal{C}(i), r' < r} y_{ir'} = 0$ and $\sum_{r' \in \mathcal{C}(j), r' < r} y_{jr'} = 0$). This would mean that i and j are sharing a color, $\varphi_1(i) = \varphi_1(j)$, that is, we are in case (i). If this happens, (24) force $x_{ij11} = 1$. A similar argument proves that constraints (25) stand for case (ii).

A slightly different argument serves to model (iii) and (iv) with the following mathematical constraints,

$$\sum_{u \in \mathcal{C}(i,j)} s_{iju} - \sum_{\substack{r' \in \mathcal{C}(i) \\ r' < r}} y_{ir'} - \sum_{\substack{r' \in \mathcal{C}(j) \\ r' > r}} y_{jr'} \leq x_{ij12} \quad \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j) \tag{26}$$

$$\sum_{u \in \mathcal{C}(i,j)} s_{iju} - \sum_{\substack{r' \in \mathcal{C}(i) \\ r' > r}} y_{ir'} - \sum_{\substack{r' \in \mathcal{C}(j) \\ r' < r}} y_{jr'} \leq x_{ij21} \quad \forall (i,j) \in E, \forall r \in \mathcal{C}(i,j). \tag{27}$$

Again, these constraints are active when their left-hand sides are 1. Take for instance (26). If i and j share a color, then one of the terms of the first sum on the left-hand side will be one. If, for some color r, the colors given to i are greater than or equal to r ($\sum_{r' \in \mathcal{C}(i), r' < r} y_{ir'} = 0$) and those given to j are smaller than or equal to r ($\sum_{r' \in \mathcal{C}(j), r' > r} y_{jr'} = 0$), then i and j share color r. When this happens, r coincides with $\varphi_1(i)$ and $\varphi_2(j)$, that is, we are in case (iii). In this case, (26) force $x_{ij12} = 1$. A similar reasoning proves that (27) serve to model case (iv).

There are other constraints that can be included to model relations coming from the semantic meaning of the decision variables. These relations can be explicitly written in the mathematical model, with the aim of ultimately producing a stronger polyhedron.

For instance, if i and j do not share a color, that is, $\sum_{a=1}^{2} \sum_{b=1}^{2} x_{ijab} = 0$, then node j can not receive any of the colors of i, $2i - 1$ or $2i$. This can be expressed with the following constraints:

$$y_{j,2i-1} + y_{j,2i} \leq \sum_{a=1}^{2} \sum_{b=1}^{2} x_{ijab} \quad \forall j > i : (i,j) \in E. \tag{28}$$

On the other hand, if $y_{i,2i} = 0$ then color $2i-1$ is the greatest color that can be assigned to i. Thus, if $x_{ij21} = x_{ij22} = 0$ as well, node j can be colored neither with $2i-1$ nor with $2i$:

$$y_{j,2i-1} + y_{j,2i} \leq x_{ij21} + x_{ij22} + y_{i,2i} \quad \forall j > i : (i,j) \in E. \tag{29}$$

Following a similar idea and taking into account that $2i$ is the greatest color that can be assigned to i, we can devise the following constraints,

$$y_{j,2i} \leq x_{ij21} + x_{ij22} \quad \forall j > i : (i,j) \in E. \tag{30}$$

Recalling that symmetry breaking constraints in (H) impose that $y_{i,2i-1} = 1$ if $y_{i,2i} = 1$, we observe that if i and j share a color and i receives color $2i$ then j receives $2i-1$ or $2i$:

$$y_{i,2i} + \sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \leq 1 + y_{j,2i-1} + y_{j,2i} \quad \forall j > i : (i,j) \in E. \tag{31}$$

Finally, the following equation gathers "incompatible" variables in set packing constraints,

$$y_{j,2i} + y_{j,2j} + x_{ij11} + x_{ij12} \leq 1 \quad \forall j > i : (i,j) \in E. \tag{32}$$

5.2. A family of mixed formulations

The first formulation in family (HM), named (HMa), consists of some constraints from (H) and (M) plus the linking constraints above described. The formulation is

$$\begin{aligned}
\text{(HMa)}\ \max \quad & \sum_{(i,j)\in E}\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} & & \\
\text{s.t.} \quad & (1),(2),(4)-(9), & & \\
& (18),(24)-(32) & & \\
& \sum_{r\in\mathcal{C}(i,j)} s_{ijr} = \sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} & \forall (i,j) \in E & (33)\\
& y_{j,2j} + \sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \leq 1 & \forall j > i : (i,j) \in E & (34)\\
& \sum_{j\in S}(x_{ija1} + x_{ija2}) \leq 1 & \forall i, \forall S \subseteq N(i), S \in \mathcal{P}_G, |S| \in \{2,3\} & \\
& & \forall a \in \{1,2\} & (35)\\
& y_{ir}, s_{ijr'} \in \{0,1\} & \forall i \in V, \forall r \in \mathcal{C}(i), \forall (i,j) \in E, \forall r' \in \mathcal{C}(i,j) & \\
& x_{ijab} \in \{0,1\} & \forall (i,j) \in E, \forall a,b \in \{1,2\}. &
\end{aligned}$$

Constraints (1), (2) and (4)-(9) integrate (H) in the formulation except for the fact that (3) are missing. These last constraints are replaced by (33), which are the result of combining (3) and (14).

Constraints (34) are a stronger version of (15) of (M), since one variable, $y_{j,2j}$, has been added to the left-hand side. Note that, if $y_{j,2j} = 1$, j is colored with $2j$ and $2j-1$. Since $j > i$,

i and j will not share any color in this case, so $\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} = 0$. Constraints (35) are (19) when $|S| = 2$ or $|S| = 3$, which in particular include (10). If $|S_1| = 2$, $|S_2| = 3$ and $S_1 \subseteq S_2$, we would only consider (35) for S_2. Symmetry breaking constraints (18) of formulation (M) are also included in (HMa).

The reader may have noticed that constraints (11)-(13) of (M) have not been included in (HMa). Nevertheless, the formulation is valid since it somehow includes (H) —more precisely it includes the modification of (H) that results from eliminating variables d. There is room for wondering whether a solution of (HMa) is feasible for (M), though. In Proposition 2, we will show that every feasible solution of (HMa) satisfies constraints (11)-(13). The following lemma gives a partial result that will be crucial in the proof of Proposition 2.

Lemma 4. *Let $\{i, j, k\} \in \Gamma$ be a triangle in G and let $(\bar{y}, \bar{s}, \bar{x})$ be a feasible solution of (HMa). If $\bar{x}$ is such that one of the nodes in $\{i, j, k\}$ shares the same color with the other two then there exists $r \in \mathcal{C}$ such that $\bar{y}_{ir} = \bar{y}_{jr} = \bar{y}_{kr} = 1$.*

Proof. Suppose for instance that $\bar{x}_{jkac} = \bar{x}_{ikbc} = 1$, where $a, b, c \in \{1, 2\}$. That is, node k shares its color in position c with both i and j. Recalling that nodes in the triangle are ordered, i.e., $i < j < k$, different assumptions can be made other than k sharing a color with i and j. Namely, it could be that $\bar{x}_{ijab} = \bar{x}_{jkbc} = 1$ or $\bar{x}_{ijab} = \bar{x}_{ikac} = 1$. We will prove the lemma when $\bar{x}_{jkac} = \bar{x}_{ikbc} = 1$ and overlook the other cases, which can be proven likewise.

Due to (33), $\exists r_1, r_2$ such that $\bar{s}_{ikr_1} = \bar{s}_{jkr_2} = 1$. Because of (5) and (6), we have that $\bar{y}_{ir_1} = \bar{y}_{kr_1} = \bar{y}_{jr_2} = \bar{y}_{kr_2} = 1$. For contradiction, suppose that $r_1 \neq r_2$. We distinguish the following cases:

1. $c = 1$ and $r_1 < r_2$.
   Take (25) applied to indices j, k, r_2:

   $$\bar{s}_{jkr_2} - \sum_{r'>r_2} \bar{y}_{jr'} - \sum_{r'>r_2} \bar{y}_{kr'} \leq \bar{x}_{jk22}.$$

   Since $\bar{s}_{jkr_2} = 1$, $\sum_{r'>r_2} \bar{y}_{kr'} = 0$ because $r_1 < r_2$ and $\bar{x}_{jk22} = 0$ because $c = 1$, the inequality above only stands if $\sum_{r'>r_2} \bar{y}_{jr'} = 1$. If we now consider (26) with same indices j, k, r_2:

   $$\sum_{u} \bar{s}_{jku} \leq \bar{x}_{jk12} + \sum_{r'<r_2} \bar{y}_{jr'} + \sum_{r'>r_2} \bar{y}_{kr'},$$

   it implies $\bar{x}_{jk12} = 1$, which is a contradiction.

2. $c = 1$ and $r_2 < r_1$.
   The proof is analogous to the previous case just by taking (25) and (26) with indices i, k, r_1 instead of j, k, r_2.

3. $c = 2$ and $r_1 < r_2$.
   Take (27) applied to indices i, k, r_1:

   $$\sum_{u} \bar{s}_{iku} \leq \bar{x}_{ik21} + \sum_{r'>r_1} \bar{y}_{ir'} + \sum_{r'<r_1} \bar{y}_{kr'}.$$

Since $\sum_u \bar{s}_{iku} = 1$, $\bar{x}_{ik21} = 0$ due to $c = 2$ and $\sum_{r'<r_1} \bar{y}_{kr'} = 0$ because $r_1 < r_2$, the inequality above implies $1 = \sum_{r'>r_1} \bar{y}_{ir'}$. Now take (24) again with indices i, k, r_1:

$$\bar{s}_{ikr_1} - \sum_{r'<r_1} \bar{y}_{ir'} - \sum_{r'<r_1} \bar{y}_{kr'} \leq \bar{x}_{ik11},$$

where $\bar{s}_{ikr_1} = 1$ and the rest terms on the left-hand side are zero, which yields $\bar{x}_{ik11} = 1$. This contradicts initial assumption $c = 2$.

4. $c = 2$ and $r_2 < r_1$.
   The proof is analogous to the previous case just by taking (27) and (24) with indices j, k, r_2 instead of i, k, r_1.

All the cases lead to a contradiction and hence we have found $r := r_1 = r_2$ as desired. □

Proposition 2. *Let $(\bar{y}, \bar{s}, \bar{x})$ be a feasible solution to (HMa). Then $\bar{x}$ satisfy transitivity constraints (11)-(13).*

Proof. Let $\{i, j, k\} \in \Gamma$ be a triangle in G. We have to prove that the coloring of nodes i, j, k encoded by $\bar{x}$ is transitive. That is, if, according to the solution, one of the nodes in the triangle $\{i, j, k\}$ shares the same color with the other two, we have to prove that the latter share that color as well. Lemma 4 ensures that $\bar{y}_{ir} = \bar{y}_{jr} = \bar{y}_{kr} = 1$ for some color r.

Given that $\bar{y}_{ir} = \bar{y}_{jr} = \bar{y}_{kr} = 1$, $\bar{s}_{ijr} = \bar{s}_{jkr} = \bar{s}_{ikr} = 1$ follows from (4). Now, we can define $Y^+_{ir} = \sum_{r'>r} \bar{y}_{ir'}$ and $Y^-_{ir} = \sum_{r'<r} \bar{y}_{ir'}$, for all $i \in V, r \in \mathcal{C}(i)$. Constraints (24)- (27) for the nodes of our triangle $\{i, j, k\}$ are then:

$$\begin{array}{llll}
1 \leq & x_{ij11} + Y^-_{ir} + Y^-_{jr} & 1 \leq & x_{ik11} + Y^-_{ir} + Y^-_{kr} \quad 1 \leq \; x_{jk11} + Y^-_{jr} + Y^-_{kr} \\
1 \leq & x_{ij22} + Y^+_{ir} + Y^+_{jr} & 1 \leq & x_{ik22} + Y^+_{ir} + Y^+_{kr} \quad 1 \leq \; x_{jk22} + Y^+_{jr} + Y^+_{kr} \\
1 \leq & x_{ij12} + Y^-_{ir} + Y^+_{jr} & 1 \leq & x_{ik12} + Y^-_{ir} + Y^+_{kr} \quad 1 \leq \; x_{jk12} + Y^-_{jr} + Y^+_{kr} \\
1 \leq & x_{ij21} + Y^+_{ir} + Y^-_{jr} & 1 \leq & x_{ik21} + Y^+_{ir} + Y^-_{kr} \quad 1 \leq \; x_{jk21} + Y^+_{jr} + Y^-_{kr}
\end{array}$$

Since, from (1), $\sum_{r \in \mathcal{C}(i)} \bar{y}_{ir} = 2$, it is straightforward that either $Y^+_{ir} = 1$ (and $Y^-_{ir} = 0$) or $Y^-_{ir} = 1$ (and $Y^+_{ir} = 0$), and the same is true for j and k. Consider the group of rewritten constraints for i, j, which are on the first column. In every solution, only one of the four constraints has null Y-terms, which will indicate which of the x_{ij}-variables takes value one. The same happens for the second and third columns of constraints. It is easy to check that any of the nine possibilities regarding sums $Y^{+,-}_{ir}, Y^{+,-}_{jr}, Y^{+,-}_{kr}$ yield x-variables that respect transitivity in the triangle $\{i, j, k\}$. For instance, suppose that $Y^+_{ir} = 1$, $Y^-_{jr} = 1$ and $Y^-_{kr} = 1$. Then, $\bar{x}_{ij12} = 1$, $\bar{x}_{ik12} = 1$ and $\bar{x}_{jk22} = 1$ yield respectively from the fourth constraints in the first and second columns and the second constraint in the third column. These values respect transitivity constraints (11)-(13). □

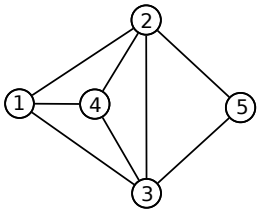


Figure 6: Graph of Remark 1

Remark 1. *Proposition 2 is not true for fractional solutions to (HMa). Consider for instance the graph depicted in Figure 6. A feasible solution to (HMa) is:*

$$
\begin{array}{llllll}
x_{1211}=1.0 & x_{2411}=0.5 & s_{132}=0.5 & s_{341}=0.5 & y_{31}=0.5 & y_{53}=1.0 \\
x_{1311}=0.5 & x_{2521}=1.0 & s_{141}=0.5 & s_{342}=0.5 & y_{32}=0.5 & y_{59}=1.0, \\
x_{1321}=0.5 & x_{3411}=0.5 & s_{142}=0.5 & s_{353}=0.5 & y_{33}=0.5 & \\
x_{1411}=0.5 & x_{3422}=0.5 & s_{231}=0.5 & y_{11}=1.0 & y_{35}=0.5 & \\
x_{1421}=0.5 & x_{3521}=0.5 & s_{233}=0.5 & y_{12}=1.0 & y_{41}=0.5 & \\
x_{2311}=0.5 & s_{121}=1.0 & s_{241}=0.5 & y_{21}=1.0 & y_{42}=0.5 & \\
x_{2321}=0.5 & s_{131}=0.5 & s_{253}=1.0 & y_{23}=1.0 & y_{47}=1.0 &
\end{array}
$$

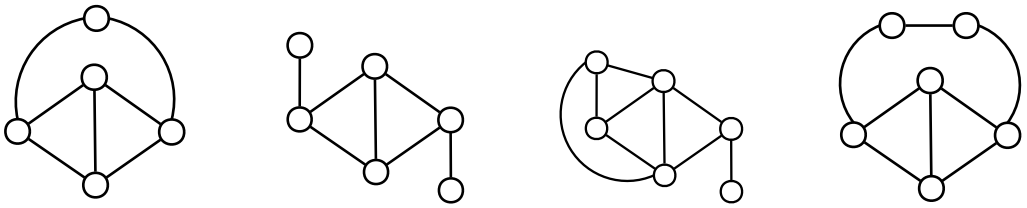
which violates $x_{3511} \geq x_{2521} + x_{2321} - 1$.

Previous remark implies that including (11)-(13) in (HMa) could produce a formulation with stronger LP bounds than that of (HMa). Nevertheless, computational experience shows that the LP bound of (HMa) is usually not affected when (11)-(13) are included to the formulation. When this is not the case, the improvement of the bound with respect to plain (HMa) is insignificant.

We noticed during the computational tests that running times of (HMa) can be significantly reduced with little detriment in the LP bound, by removing some constraints. After a few tests, we decided to consider as well the two following modifications of the initial combined formulation.

$$
\begin{aligned}
\text{(HMb)}\ \max \quad & \sum_{(i,j)\in E}\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \\
\text{s.t.} \quad & (1),(2),(4)-(9),(18),(33)-(35),(28-32) \\
& y_{i,r}, s_{ijr'}, x_{ijab} \in \{0,1\} \qquad \forall i \in V, \forall r \in \mathcal{C}_i, \forall (i,j) \in E, \\
& \qquad\qquad\qquad\qquad\qquad\quad \forall r' \in \mathcal{C}(i,j), \forall a,b \in \{1,2\}
\end{aligned}
$$

$$
\begin{aligned}
\text{(HMc)}\ \max \quad & \sum_{(i,j)\in E}\sum_{a=1}^{2}\sum_{b=1}^{2} x_{ijab} \\
\text{s.t.} \quad & (1),(2),(4)-(9),(18),(33)-(35),(30) \\
& y_{i,r}, s_{ijr'}, x_{ijab} \in \{0,1\} \qquad \forall i \in V, \forall r \in \mathcal{C}_i, \forall (i,j) \in E, \\
& \qquad\qquad\qquad\qquad\qquad\quad \forall r' \in \mathcal{C}(i,j), \forall a,b \in \{1,2\}.
\end{aligned}
$$



(a) T_1 (b) T_2 (c) T_3 (d) T_4

Figure 7: Four graphs containing a pair of adjacent triangles, both being odd

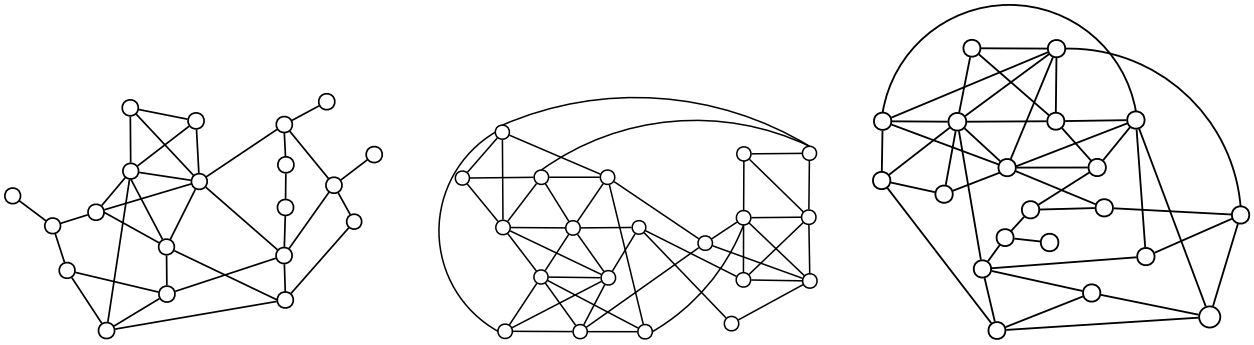


Figure 8: Three forbidden graphs of 20 nodes each, containing overlapping $K_{1,3}$, T_1, T_2, T_3 and T_4

Note that, due to the deletion of constraints, we can not guarantee that an optimal solution of (HMb) or (HMc) is going to be feasible for (M). Nevertheless, we can ensure that variables y codify the optimal solution. This is because the formulation that results after removing variables d from (H) is embedded in both (HMb) and (HMc), and is given by constraints (1), (2), (4)-(9), (33) and (34). Thus, in the case of (HMb) and (HMc), variables x are just a tool to produce a tighter and therefore more efficient formulation, as the computational experience reported in the next section proves.

6. Computational experiments

The aim of our computational study is twofold. First, we would like to study the effect of the valid inequalities of Section 4.1 on (M), in order to determine whether it is useful to add them (or a subset of them) to the model. Second, we will compare the resulting best configurations for (M) with (H), (HMa), (HMb) and (HMc).

The processor used for the tests was an Intel core i7-6700k CPU at 4.0 GHz $\times$ 8 with 16 GB of RAM memory. The solver was CPLEX v12.6.3 64-bit under operating system Linux Ubuntu 16.04. The testbed consisted of four families of instances, which we have generated ourselves except for the last:

- Forbidden graphs: these graphs have been generated by combining some of the forbidden subgraphs in [2] several times and randomly adding more edges. Namely, we consider $K_{1,3}$ (see Figure 1b) and the four graphs depicted in Figure 7, $T_1, \ldots, T_4$, which violate the characterization of line graphs given in [17]. We replicate each of them up to eight times independently, and link the copies with density 50, i.e. two nodes of different copies are linked with probability 0.5. We also generate graphs by linking $K_{1,3}$ and T_1, replicating

	Fork ineq.	Sim. breaking	Ineq. (22) and (23)
(M)	No		No
(M1)	No		Yes
(M2)	No	(18)	No
(M3)	No	(18)	Yes
(M4)	No	(16), (17)	No
(M5)	No	(16), (17)	Yes
(M6)	Yes		No
(M7)	Yes		Yes
(M8)	Yes	(18)	No
(M9)	Yes	(18)	Yes
(M10)	Yes	(16), (17)	No
(M11)	Yes	(16), (17)	Yes

Table 3: Configurations used in the preliminary study

the resulting graph two or three times and combining the copies as before. Finally, the testbed is completed with the three bigger graphs of Figure 8, which contain many overlapping forbidden structures, and graphs resulting from combining them up to three times.

- Random graphs: graphs randomly generated with different number of nodes and densities. Namely, we generated graphs of densities 75, 60 and 50 with 15, 20 and 30 nodes, respectively; graphs with density 30, having 40 and 50 nodes; density 20 with number of nodes 50, 60 and 70; density 15 with 80, 90, 120 and 150 nodes; three graphs with density 10, having 100, 150 and 175 nodes; finally, one graph with density 5 and 200 nodes.

- Perturbed line graphs: we use a graph as starting point, generate its line graph and then we randomly add edges to it, until a maximum number of extra edges is reached. A node is first randomly fixed and edges are added between it and up to three other nodes, which potentially increases the number of subgraphs $K_{1,3}$ contained by the instance. We use the resulting graph as problem input. With this procedure we obtain a bound on the optimal value (the number of edges added). The graphs used as starting point were randomly generated with the following characteristics: three graphs having densities 80, 75 and 70, with 15, 20 and 20 nodes respectively; two with density 40 having 30 and 40 nodes; two with density 30 having 30 and 50 nodes; one graph with density 25 and 40 nodes; four graphs with densities 20 and 40, 50, 60 having 70 nodes each; three with densities 15 and 60, 70 having 80 nodes; four that have density 10 with 70, 90,100 and 120 nodes; and, finally, two graphs with density 5 having 100 and 120 nodes. The resulting line graphs were added between 20 and 360 edges.

- HapMap graphs: these instances have been provided by Hálldorsson and were used in [7]. They codify information from a sample of 77 individuals taken from the HapMap populations [16]. Namely, the instances used are: ASW_100_{0,2}, ASW_200_8, CHB_100_{5,8}, CHB_200_{0,3}, GIH_100_{5,8}, GIH_200_{0,8}, JPT_200_{0,1,2,3}, LWK_100_{2,5,9} and TSI_200_{0,1,2,3}.

File	n	m	den.%	OPT	LP OPT												
					H	M	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11
Forb1	20	100	53	43	11.5	0.0	0.0	8.5	8.5	8.5	8.5	32.5	32.5	**34.6**	**34.6**	34.4	34.4
Forb2	18	87	57	33	11.3	0.0	0.0	8.0	8.0	8.0	8.0	26.8	26.8	**27.7**	**27.7**	**27.7**	**27.7**
Forb3	20	110	58	43	12.0	0.0	0.0	9.0	9.0	9.0	9.0	35.5	35.5	**37.2**	**37.2**	**37.2**	**37.2**
Forb4	32	145	29	61	40.0	0.0	0.0	14.5	14.5	14.5	14.5	47.9	47.9	**52.0**	**52.0**	51.6	51.6
Forb5	21	116	55	45	18.4	0.0	0.0	8.8	8.9	8.8	8.9	37.2	37.2	**38.9**	**38.9**	38.8	38.8
Rand1	100	498	10	326	**284.0**	0.0	0.0	47.2	47.2	47.2	47.2	166.0	166.0	197.0	197.0	197.0	197.0
Rand2	20	122	64	42	13.4	0.0	0.0	9.5	9.5	9.5	9.5	34.5	34.5	**35.3**	**35.3**	35.2	35.2
Rand3	40	254	33	148	92.2	0.0	0.0	19.0	19.0	19.0	19.0	84.7	84.7	**96.7**	**96.7**	**96.7**	**96.7**
Rand4	50	256	21	150	**106.0**	0.0	0.0	23.5	23.5	23.5	23.5	85.3	85.3	100.0	100.0	100.0	100.0
Rand5	60	367	21	228	**169.0**	0.0	0.0	28.5	28.5	28.5	28.5	122.0	122.0	141.0	141.0	141.0	141.0
Pert1	86	957	26	45	41.0	0.0	0.0	23.0	25.3	23.0	25.3	**45.0**	**45.0**	**45.0**	**45.0**	**45.0**	**45.0**
Pert2	143	2004	20	72	**72.0**	0.0	0.0	36.0	40.4	36.0	40.2	**72.0**	**72.0**	**72.0**	**72.0**	**72.0**	**72.0**
Pert3	137	1851	20	69	**69.0**	0.0	0.0	34.0	38.6	34.0	38.5	**69.0**	**69.0**	**69.0**	**69.0**	**69.0**	**69.0**
Pert4	191	2461	14	102	99.0	0.0	0.0	48.0	49.2	48.0	49.2	**102.0**	**102.0**	**102.0**	**102.0**	**102.0**	**102.0**
Pert5	271	2550	7	136	135.0	0.0	0.0	74.2	74.8	74.2	74.8	**136.0**	**136.0**	**136.0**	**136.0**	**136.0**	**136.0**
Hap1	77	313	11	161	**114.0**	0.0	0.0	22.2	22.2	22.2	22.2	98.7	98.7	107.0	107.0	107.0	107.0
Hap2	77	253	9	130	83.0	0.0	0.0	21.0	21.0	21.0	21.0	83.5	83.5	**90.8**	**90.8**	90.6	90.6
Hap3	77	357	12	188	99.0	0.0	0.0	22.5	22.5	22.5	22.5	118.0	118.0	**128.0**	**128.0**	**128.0**	**128.0**
Hap4	77	338	12	146	91.0	0.0	0.0	23.7	23.7	23.7	23.7	107.0	107.0	**113.0**	**113.0**	112.0	112.0
Hap5	77	308	11	152	102.0	0.0	0.0	27.8	27.8	27.8	27.8	102.0	102.0	**112.0**	**112.0**	111.0	111.0

Table 4: Preliminary study results

6.1. Preliminary study

In the first part of our computational study, we have used (M) with different combinations of the inequalities of Section 4.1:

- Either without (19) or with "fork inequalities" (19) with $|S| = 3$,
- Three possibilities regarding symmetry breaking: without constraints, with (16) and (17) or with (18),
- Either with valid inequalities (22) and (23) or without them.

After combining the options above, we obtain twelve configurations, which we call (M), (M1),. . ., (M11) (see Table 3). We perform a preliminary study in order to find the best configurations among the twelve. Here we use five instances from each of the four databases, which makes a total of 20 instances. We fix a time limit of 300 seconds, disable CPLEX dynamic search to force CPLEX to use traditional branch-and-cut strategy, and execute each experiment with and without CPLEX cuts.

Table 4 summarizes some results of this preliminary study. In the first columns we can see the name of the instances (File), the number of nodes (n), the number of edges (m), the density as a percentage (den.%) and the optimal value for EDPL (OPT). The following columns show the optimal value of the linear relaxation of H, M, M1,. . .,M11. As we mentioned in the previous section, there are instances for which H has the best LP bound, while our formulations are better in this respect for some others (best LP bounds are in bold).

Table 5 gathers the running times obtained in this preliminary study when CPLEX cuts are activated (default) and when we disable them. Entries displaying "TL" stand for those experiments in which the time limit was reached without finding the optimum. It can be observed that the formulations we propose obtain satisfactory results in comparison to (H), except for the third data set. The configurations that obtained the best running times were (M4), (M8) and (M10).

File	Running time with/without CPLEX cuts																							
	H		M		M1		M2		M3		M4		M5		M6		M7		M8		M9		M10		M11	
Forb1	8.7	60.8	10.4	11.0	175.1	285.4	**3.1**	**5.8**	63.6	132.6	7.9	8.3	118.4	210.7	14.7	7.0	TL	287.0	5.7	**5.2**	70.5	75.5	5.5	7.0	104.1	260.1
Forb2	4.0	5.0	6.9	4.4	123.4	102.1	**2.7**	**1.8**	23.3	31.6	5.8	**2.2**	38.4	38.3	7.0	3.5	127.6	85.7	**2.4**	**2.1**	31.3	23.6	4.8	**2.3**	48.6	31.3
Forb3	15.5	63.2	14.8	10.4	TL	TL	9.6	7.6	160.5	201.9	**8.8**	**6.0**	120.1	119.1	13.7	11.4	TL	TL	10.4	**6.4**	96.3	197.3	9.5	9.4	122.2	203.6
Forb4	1.6	1.9	5.5	4.2	89.7	58.4	2.6	**0.5**	12.0	12.3	**0.8**	**0.5**	12.1	7.2	2.9	2.7	34.6	52.0	2.9	1.4	12.9	19.9	1.6	2.0	10.4	16.6
Forb5	11.5	17.3	13.1	9.5	242.4	277.3	4.3	7.2	39.1	75.3	**3.4**	**3.8**	52.5	229.9	11.3	6.2	265.1	240.8	4.2	7.1	22.9	61.7	4.1	**4.1**	25.0	118.4
Rand1	**1.2**	119.8	2.5	6.9	5.8	19.4	1.5	7.7	3.5	19.3	**1.4**	3.4	2.2	3.9	**1.4**	6.7	4.7	5.0	2.0	**2.4**	4.0	11.3	**1.1**	5.4	2.0	2.8
Rand2	66.3	175.6	235.8	65.8	TL	TL	69.1	89.9	TL	TL	175.6	52.4	TL	TL	181.9	**40.8**	TL	TL	**44.5**	55.8	TL	TL	TL	51.0	TL	TL
Rand3	36.6	301.4	48.2	206.3	TL	TL	49.4	141.5	TL	TL	**29.7**	TL	TL	TL	56.9	132.1	TL	TL	75.5	131.3	TL	TL	38.3	**72.5**	TL	TL
Rand4	**1.3**	21.1	**1.4**	4.0	8.9	29.1	**1.2**	1.1	4.7	5.5	1.2	**0.3**	2.7	1.1	1.4	**0.7**	5.0	12.5	0.9	**0.5**	2.8	1.1	1.1	**0.4**	5.1	5.2
Rand5	8.8	301.1	16.6	170.7	TL	TL	9.5	61.5	34.4	TL	6.7	71.4	50.6	TL	32.2	34.8	TL	TL	**5.3**	**11.7**	37.2	228.8	8.8	14.7	46.5	TL
Pert1	**4.4**	**8.3**	TL	TL	TL	TL	16.5	16.0	100.1	99.8	13.4	13.3	89.8	92.4	TL	TL	TL	TL	17.0	16.0	94.3	108.1	14.0	13.7	88.7	91.2
Pert2	**12.4**	**16.4**	TL	TL	TL	TL	108.4	109.8	TL	TL	86.0	86.4	TL	TL	TL	TL	TL	TL	109.4	103.6	TL	TL	89.1	89.0	TL	TL
Pert3	**3.4**	**3.3**	TL	TL	TL	TL	79.4	80.8	TL	TL	66.7	66.9	TL	TL	TL	TL	TL	TL	82.9	82.9	TL	TL	69.1	69.2	TL	TL
Pert4	**2.7**	**8.6**	TL	TL	TL	TL	135.8	132.2	TL	TL	126.6	127.3	TL	TL	TL	TL	TL	TL	142.1	140.9	TL	TL	118.5	118.9	TL	TL
Pert5	**1.8**	**3.3**	TL	TL	TL	TL	81.1	80.9	162.5	163.4	63.6	63.2	138.9	144.8	TL	TL	TL	TL	86.9	85.2	166.8	166.6	66.0	66.9	139.9	139.7
Hap1	**10.0**	280.1	44.5	50.6	TL	TL	18.8	172.2	TL	TL	36.4	TL	TL	TL	43.8	69.1	TL	TL	29.8	**33.6**	TL	TL	46.3	95.2	TL	TL
Hap2	4.2	22.7	8.7	9.2	245.6	226.5	**3.5**	8.3	26.8	50.5	4.3	7.4	68.4	168.5	**3.8**	8.0	20.2	209.5	**3.2**	7.4	64.1	35.4	4.0	**5.9**	20.4	116.9
Hap3	**29.7**	301.7	123.8	136.4	TL	TL	47.0	92.7	TL	TL	TL	299.2	TL	TL	116.6	157.5	TL	TL	39.9	**69.0**	TL	TL	TL	TL	TL	TL
Hap4	13.7	120.2	27.6	59.0	TL	TL	**10.6**	38.4	227.9	TL	50.4	133.8	TL	TL	89.1	51.2	TL	TL	13.9	**19.5**	95.2	TL	35.5	167.2	TL	TL
Hap5	**6.5**	184.3	18.8	25.0	TL	TL	**6.4**	21.6	160.3	TL	10.3	31.7	247.8	TL	19.8	33.5	TL	TL	10.0	**13.2**	283.7	TL	9.9	**12.8**	129.2	TL

Table 5: Running times in seconds of preliminary study

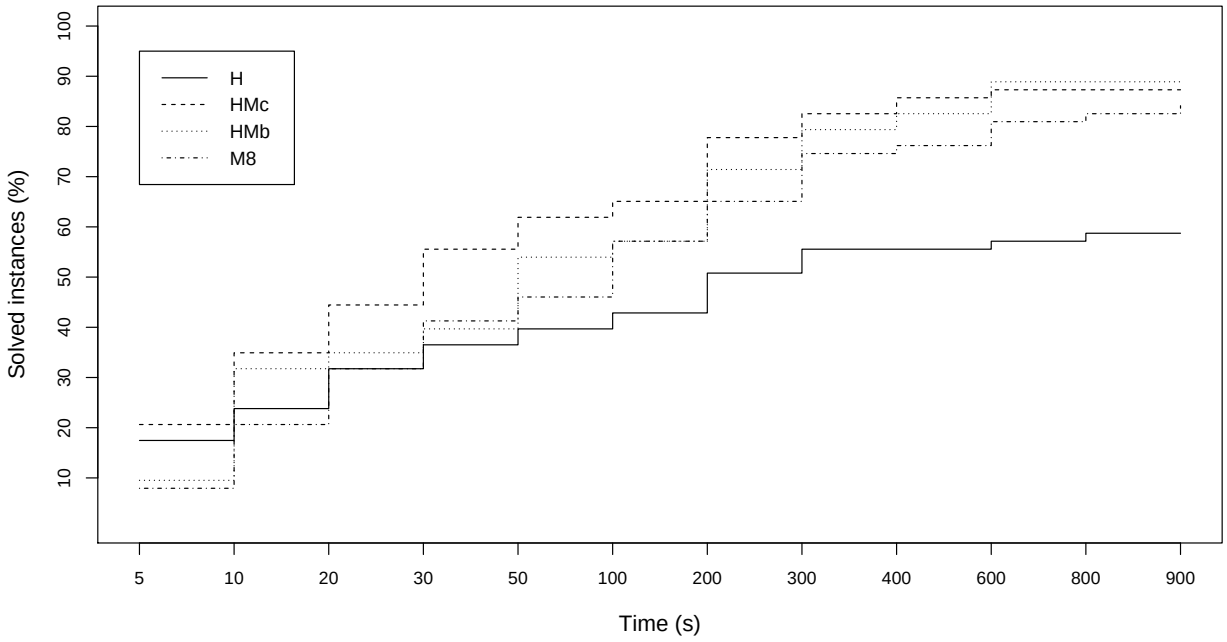


Figure 9: Percentage of solved instances after the seconds shown on the abscissa axis (this axis is not scaled)

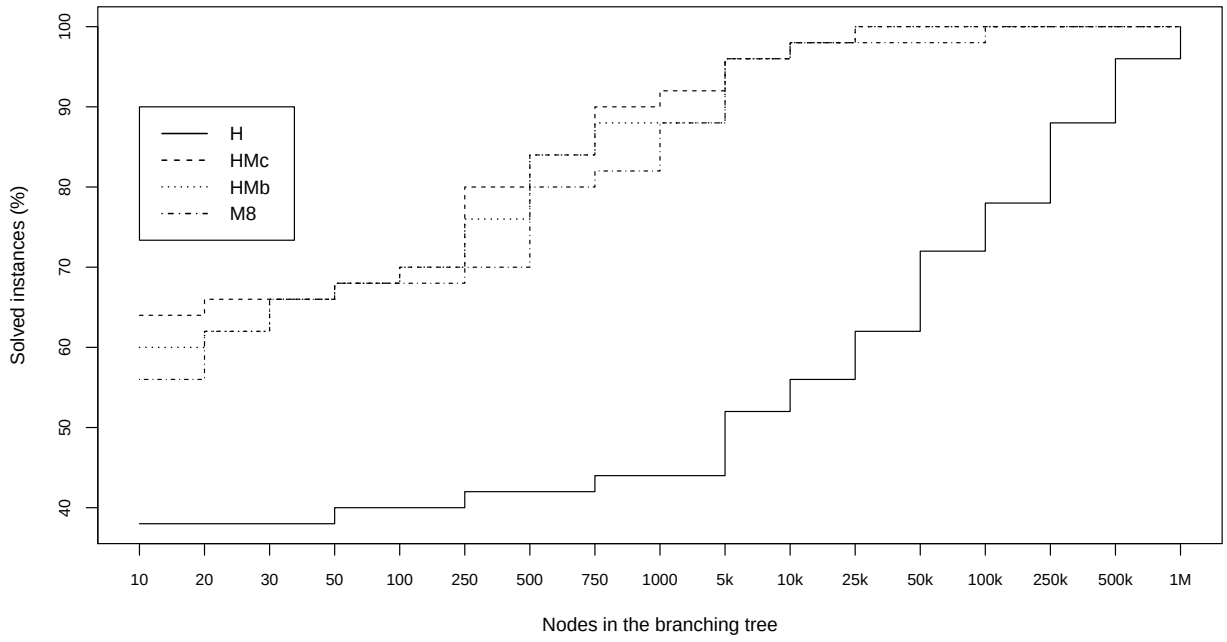


Figure 10: Percentage of solved instances after exploring as many nodes as on the abscissa axis (this axis is not scaled)

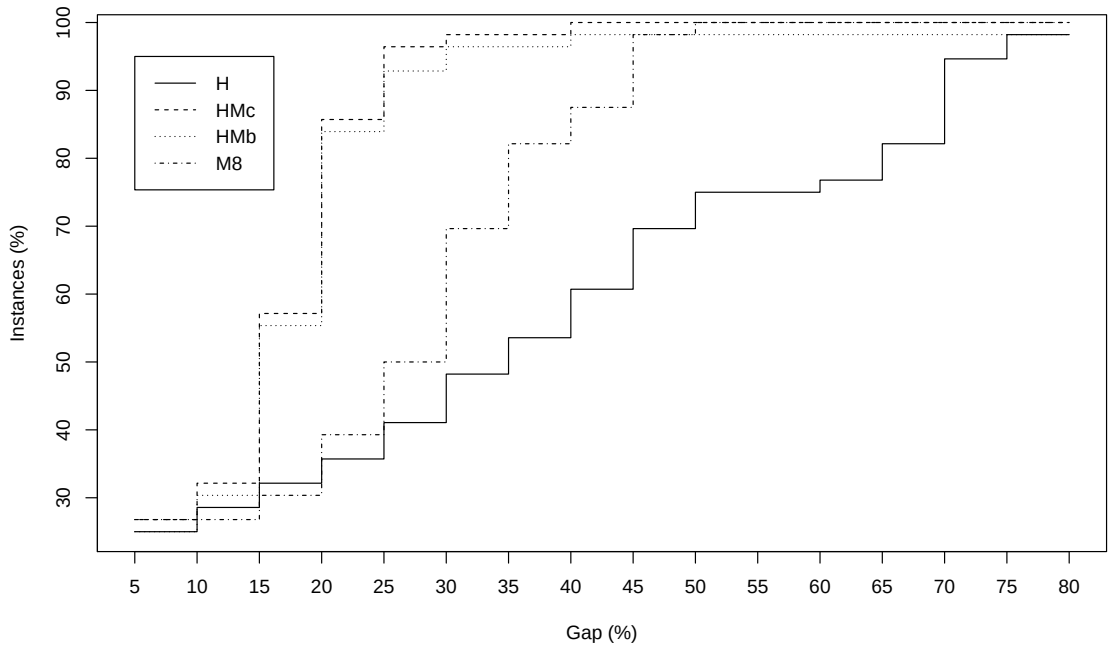


Figure 11: Percentage of instances with a gap smaller than or equal to the gap shown on the abscissa axis

6.2. Main computational study

The aim of the main study is to compare Halldórsson's formulation (H), the three winning variants of the preliminary study, (M4), (M8) and (M10), and mixed formulations (HMa), (HMb) and (HMc). For the experiments, we set a time limit of 900 seconds and disable CPLEX dynamic search and cuts. We use a testbed of 82 files: 23 forbidden graphs, 16 random graphs, 21 perturbed line graphs and 22 HapMap graphs. For the sake of clarity, all the diagrams and charts reported here refer to (H) and the best configurations found among the six proposed formulations tested in this section, which turn out to be (M8), (HMb) and (HMc).

Figure 9 shows the percentage of instances (ordinate axis) that were solved after a certain number of seconds (abscissa axis). Overall, (HMc) obtains the highest rates. (H) barely solved 60% of the instances within the time limit, while (M8), (HMb) and (HMc) could solve between 80% and 90%. The biggest step that separates (HMc) from the other models on the chart is placed between 30 and 50 seconds, being (HMc) able to solve more than half of the instances after 50 seconds. Figure 10 illustrates a similar comparison but in terms of the nodes of the branching tree explored instead of the running time. Note that this chart only considers instances that all the compared models could solve (the comparison is pointless otherwise). The difference on the overall performance of (H) and the rest of the models is huge. While (H) solves less than half of the instances after exploring 5000 nodes, (HMc) solves more than 90%. Up to 5000 nodes (HMc) is the overall winner, and (M8) and (HMb) reach its performance from 5000 nodes on. Finally, (H) needs more than $500,000$ nodes to solve all the instances. The last chart, depicted on Figure 11, shows the percentage of instances (ordinate axis) that has an LP gap smaller than or equal to a certain limit (abscissa axis). The LP gap varies depending on

the formulation used and is expressed as a percentage. It is given by the distance (in absolute terms) between the optimal value of EDPL and the optimal value of the linear relaxation of the formulation. To calculate the gap, we always take the objective "number of edges deleted", that is, we transform the objective values obtained with the formulations where variables d were deleted, which stand for "number of remaining edges". According to the figure, mixed models (HMb) and (HMc) show the best overall trends, as they were meant to. (M8) also shows a good performance; the number of instances within a given gap grows rapidly when the gap limit increases by steps of five units. The chart shows that (HMc) has the tightest LP gap, which is smaller than or equal to 45% for all the instances.

Figures 12 and 13 depict some statistical information about the running times of the experiments. Bars on Figure 12 show the average running times for each model grouped by data set. Although we observe that (H) has by far the tallest bars in general, this is not the case for the perturbed line graphs data set, for which it clearly has the smallest running time. As regards the other data sets, (HMc) seems to be a good alternative. On the other hand, Figure 13 is a box-and-whisker diagram, which is again grouped by data sets. The bottom and top of the boxes are the 25th and 75th percentile of the running times obtained, and the band inside them is the median. The end of the whiskers are:

$$\text{upper whisker} = \min(\max(rt), Q_3 + 1.5 \cdot IQR)$$

$$\text{lower whisker} = \max(\min(rt), Q_1 - 1.5 \cdot IQR)$$

where rt is the sample of running times and $IQR = Q_3 - Q_1$ is the box length. Finally, dots on the chart are outliers, which fall out of the whiskers limits. As opposed to the average, this kind of diagram allows us to observe the spread of values. For example, (H) presents the largest interquartile range for forbidden and HapMap graphs, which is almost equal to its overall range. This indicates that running times are highly dispersed between 0 and the time limit; the fact that Q_3 is nearly 900 reveals that around 25% of the instances were not solved within the time limit. In the case of random graphs, such percentage increases to 50% at least. Moreover, (H) solved the first 25% of the instances within 550 seconds (Q_1), as opposed to the rest of the formulations which only took few seconds. Conversely, (H) is really effective solving perturbed line graphs. Its tight box-and-whisker shows that it solved all the instances of this dataset within few seconds. As for (M8), it features highly dispersed running times for forbidden and random graphs. Although (M8) is rather fast for half of the complete testbed (see the illustration of the median for each dataset, which remains low), it is never the alternative with lowest measure Q_1, median or Q_3. In general, (M8), (HMb) and (HMc) present medians closest to the bottom of their boxes. This reveals an asymmetry in the distribution of the running times, which is denser for smaller values.

As a conclusion, we can say that formulation (H) is faster for the third group of instances, which are easier to solve as shown by Figure 13. The contribution of the proposed approach remains then at the resolution of the hardest instances. Among the formulations proposed, (HMb) and (HMc) seem the best alternatives for these datasets. The reason why (H) performs particularly well for perturbed line graphs might be the structure of the target graph, $G = (V, E)$. In these instances, G is built as follows. We start from a graph, obtain its line graph and finally add some edges to the latter. The resulting graph, G, has a strong structure where nodes (edges of the original graph) are clustered according to the end-nodes that they shared,

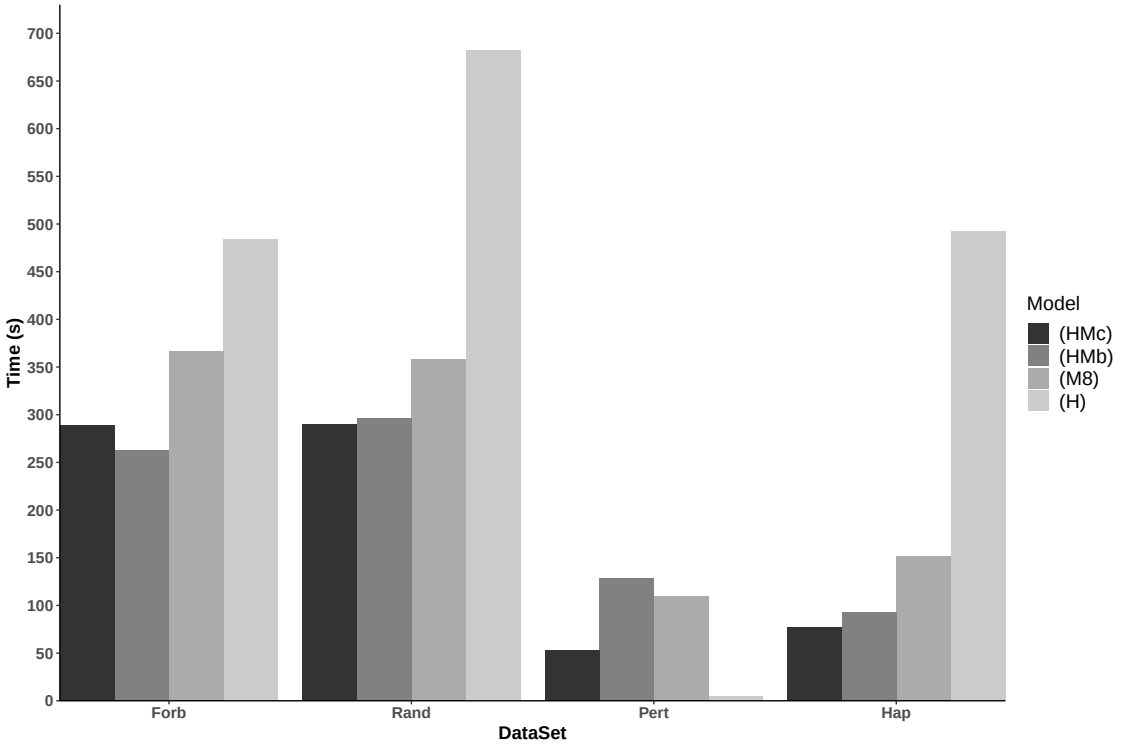


Figure 12: Average running time for the different models and data sets

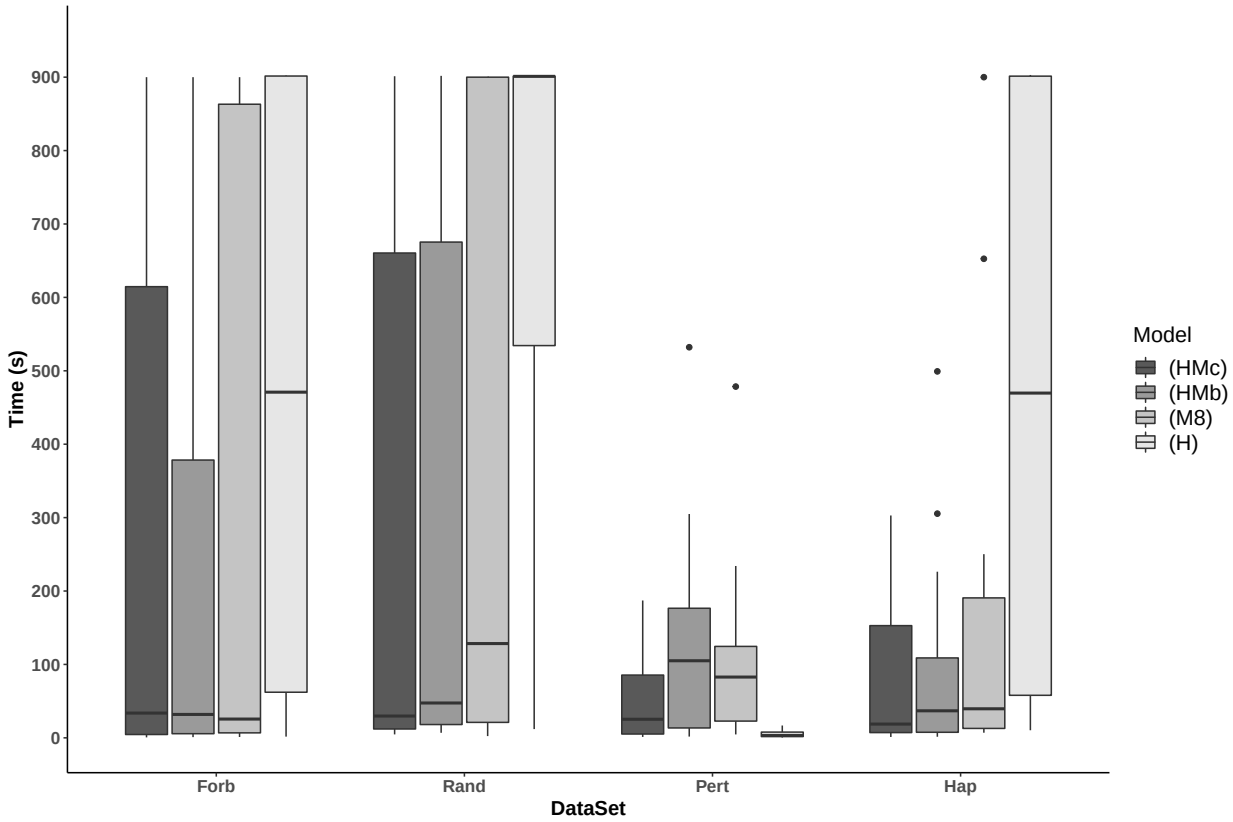


Figure 13: Box-and-whisker diagrams representing the running time for the different models and data sets

which is not disrupted after adding some edges. This structure is inherited by the variables and constraints of (H), which are defined for each edge of G (except for constraints (2)). Conversely, constraints of (M) are defined in terms of two structures: triangles and triplets $\{i, j, k\}$ such that $(i, j), (i, k) \in E$ and $(j, k) \notin E$. This might lead to a model with no structure and therefore harder to decompose and solve. In other words, (H) allows to exploit the structure of L(G) better than (M) in this case. Regarding the other datasets, average running times on Figure 12 suggested the use of (HMc), but Figure 13 shows that the running times of (HMb) are in general less dispersed.

7. Conclusions

We show that EDPL can be seen as a variant of a coloring problem and propose a new ILP formulation based on the color sharing of adjacent nodes rather than on explicitly coloring them. Valid inequalities and symmetry breaking constraints are devised for the model. We present a second family of formulations that combine elements from our new approach and the only existing ILP formulation for EDPL [7]. All the models are tested and compared in a computational study, for which we use real-case and synthetic data sets, which we generated ourselves using different criteria. We conclude that combining the existing and proposed formulations is the best alternative when it comes to hard instances, significantly reducing the computational time, the LP gap and the number of nodes of the branching tree, being (HMb) the preferred one. We also notice that Halldórsson's approach [7] is particularly fast when solving the instances of one of the data sets.

The practical application of EDPL to haplotype phasing makes the work presented here have a potential use in the process of discovering relevant items in population genetics such as the effective population size.

Acknowledgements

The authors acknowledge that research reported here was supported by Spanish *Ministerio de Economía y Competitividad*, project MTM2015-65915-R, and *Ministerio de Educación, Cultura y Deporte* PhD grant FPU15/05883. The authors sincerely thank Prof. Bjarni Vilhjálmur Halldórsson for sharing some of the benchmarks used in the computational study.

[1] Ahn, Y.Y., Bagrow, J.P., Lehmann, S. (2010). Link communities reveal multiscale complexity in networks. Nature 466, 761-764.

[2] Beineke, L.W. (1970). Characterizations of derived graphs. Journal of Combinatorial Theory 9, 129-135.

[3] Campêlo, M., Campos, V.A., Corrêa, R.C. (2007). On the asymmetric representatives formulation for the vertex coloring problem. Discrete Applied Mathematics 156, 1097-1111.

[4] Campêlo, M., Moura, P.F.S., Santos, M.C. (2013). On the representatives k-fold coloring polytope. Electronic Notes in Discrete Mathematics 44, 239-244.

[5] Clark, A.G. (1990). Inference of haplotypes from PCR-amplified samples of diploid populations. Molecular Biology and Evolution 7, 111-122.

[6] Evans, T.S., Lambiotte, R. (2009). Line graphs, link partitions, and overlapping communities. Physical Review E 80, 016105.

[7] Halldórsson, B.V., Blokh, D., Sharan, R. (2013). Estimating population size via line graph reconstruction. Algorithms for Molecular Biology, 8-17.

[8] Hoehe, M.R., Kopke, K., Wendel, B., Rohde, K., Flachmeier, C., Kidd, K.K., Berrettini, W.H., Church, G.M. (2000). Sequence variability and candidate gene analysis in complex disease: Association of μ opioid receptor gene variation with substance dependence. Human Molecular Genetics 9, 2895-2908.

[9] Krausz, J. (1943). Démonstration nouvelle d'un théorème de Whitney sur les réseaux. Mat. Fiz. Lapok 50, 75-85.

[10] Lehot, P.G.H. (1974). An optimal algorithm to detect a line graph and output its root graph. J. ACM 21, 569-575.

[11] Manka-Krason, A., Mwijage, A., Kulakowski, K. (2010). Clustering in random line graphs. Computer Physics Communications 181, 118-121.

[12] Naor, J., Novick, M. B. (1990). An efficient reconstruction of a graph from its line graph in parallel. Journal of algorithms 11, 132-143.

[13] Pullman, N. J. (1983). Clique coverings of graphs-a survey. In Combinatorial Mathematics X, 72-85. Springer, Berlin, Heidelberg.

[14] Roussopoulos, N.D. (1973). A $\max\{m, n\}$ algorithm for determining the graph H from its line graph G. Information Processing Letters 2, 108-112.

[15] Terwilliger, J.D., Weiss, K.M. (1998). Linkage disequilibrium mapping of complex disease: Fantasy and reality? Current Opinions Biotechnology 9, 579-594.

[16] The international HapMap Consortium (2010). Integrating common and rare genetic variation in diverse human populations. Nature 467, 52-58.

[17] van Rooij, A.C.M., Wilf, H.S. (1965). The interchange graph of a finite graph. Acta Mathematica Academiae Scientiarum Hungarica 16, 263-269.

[18] Whitney, H. (1992). Congruent graphs and the connectivity of graphs. Hassler Whitney Collected Papers. Birkhäuser Boston, 61-79.

[19] Wierman, J.C., Naor, D.P., Smalletz, J. (2007). Incorporating variability into an approximation formula for bond percolation thresholds of planar periodic lattices. Physical Review E 75, 011114.

[20] Yannakakis, M. (1978). Node and edge deletion NP-complete problems. In Proceedings of the 10^{th} Annual ACM Symposium on Theory of Computing, 253-264. New York.