Integer Programming Formulations for Minimum Deficiency Interval Coloring

Merve Bodur\textsuperscript{*1} and James R. Luedtke\textsuperscript{†2}

\textsuperscript{1}School of Industrial and Systems Engineering, Georgia Institute of Technology
\textsuperscript{2}Department of Industrial and Systems Engineering, University of Wisconsin-Madison

July 21, 2016

Abstract

A proper edge-coloring of a given undirected graph with natural numbers identified with colors is an interval (or consecutive) coloring if the colors of edges incident to each vertex form an interval of consecutive integers. Not all graphs admit such an edge-coloring and the problem of deciding whether a graph is interval colorable is NP-complete. For a graph that is not interval colorable, determining a graph invariant called the (minimum) deficiency is a widely used approach. Deficiency is a measure of how close the graph is to have an interval coloring and can be described as the minimum number of pendant edges whose attachment makes the graph interval colorable. The majority of the studies in the literature either derive bounds on the deficiency of general graphs, or calculate the deficiency of graphs belonging to some special graph classes. In this work, we formulate the Minimum Deficiency Problem, to find the exact deficiency value for any given graph, as an integer program, and further enhance the formulation by introducing a family of valid inequalities. Then, we solve our model via a branch-and-cut algorithm. Our computational study on a large set of random graphs illustrates the strength of our formulation and the efficiency of the proposed approach.

Keywords. Interval (consecutive) edge-coloring; minimum deficiency problem; integer programming; cutting planes; column generation

1 Introduction

Coloring problems are an extensively studied class of problems in graph theory. They arise in many applications such as scheduling and frequency allocation [29]. Besides their practical importance, they have many theoretical implications as they are useful in defining graph subclasses and provide graph invariants such as vertex chromatic number and edge chromatic number.

A proper edge-coloring in a graph is an assignment of colors to edges such that no two adjacent edges have the same color. When colors are labeled with the natural numbers, an interval coloring is a proper edge-coloring in which for each vertex of the graph, the colors assigned to edges incident to the vertex form an interval of consecutive integers. If there exists an interval coloring for a graph, the graph is called interval colorable. Simple examples of interval colorable graphs are even cycles. However, not all graphs

\textsuperscript{*}merve.bodur@gatech.edu
\textsuperscript{†}jim.luedtke@wisc.edu
are interval colorable (e.g., odd cycles). For those graphs that are not interval colorable, a natural approach is to define a graph invariant to measure how close the graph is to admit an interval coloring. This invariant is called the (minimum) deficiency of a graph and thus the problem of finding that invariant is called the Minimum Deficiency Problem (MinDef). The deficiency of a graph can be defined as the minimum number of pendant edges whose attachment to the graph makes it interval colorable, where a pendant edge is an edge attached to an existing vertex of a graph whose second endpoint is a new vertex of degree one. A more formal definition of deficiency is provided in Section 2.

There are several extensions of interval coloring. For cyclically interval coloring, a proper edge-coloring using up to a given number of colors with the following property is desired: for each vertex, either the set of colors used on edges incident to the vertex or the set of colors not used on edges incident to the vertex forms an interval of integers [25, 27]. One-sided interval coloring is designed for bipartite graphs with the aim of finding a proper edge-coloring where the “interval property” is required to be satisfied only for the vertices in one partite vertex set [26]. In some problems, the interval property is replaced with “no more than one gap property” [38]. These extensions are out of the scope of this paper, although our approach can be easily extended to such problems.

The interval coloring problem has potential applications in task scheduling, especially in constructing timetables with “compactness requirements” [4, 29]. In particular, it is useful for scheduling problems without waiting or idle times. For instance, a given set of job interviews between some firms and candidates can be modeled as a bipartite graph, with one partite vertex set representing the firms, the other representing to candidates, and edges corresponding to the interviews. The schedule (timetable) may in this case be interpreted as an edge-coloring of the graph with natural numbers corresponding to assigned time slots. Then, an interval coloring of the graph provides a schedule where neither firms nor candidates wait between their meetings. Giaro and Kubale [15] consider this problem in the case of open, flow and mixed shops. The first case is equivalent to interval coloring, while the other two requires some additional restrictions on interval coloring. If the waiting times and idle times are not prohibited but their number has to be minimized, the problem is then equivalent to (MinDef).

To the best of our knowledge, (MinDef) has been addressed from integer programming (IP) point of view only by Altinakar et al. [1], who introduce three variants of a natural IP formulation of the problem. They empirically test the efficacy of these formulations, and compare them with a natural constraint programming (CP) model. They observe that CP outperforms the IP models. However, these methods can only solve very small instances (with less than 10 vertices) to optimality.

In this paper, we start with an IP formulation of (MinDef) provided in [1], which is an edge-based formulation (i.e., decision variables correspond to the edges). We prove that its linear programming (LP) relaxation bound is always zero. Then, we propose an alternative edge-based IP formulation with a stronger LP relaxation. As the matching-based formulations usually provide stronger relaxations than the edge-based formulations for the regular edge-coloring problem, we improve this model further by deriving some matching-based valid inequalities. We suggest a branch-and-cut algorithm to solve the resulting model. In addition, we provide a matching-based formulation of (MinDef) which can be solved via a column generation based algorithm. We conduct a numerical study on a large set of randomly generated graphs to compare the performance of our IP formulations, together with the CP formulation given in [1]. We find that our branch-and-cut algorithm performs significantly better than the other approaches (especially for dense graphs which usually have high deficiencies), finding good feasible solutions and providing very strong relaxation bounds, thus proving the optimality of the solutions significantly faster.

The remainder of this paper is organized as follows. In Section 2 we review relevant definitions. In Section 3 we briefly review related literature. We present the IP formulations and the solution algorithms in
Section 4 We discuss the insights gained through our numerical results in Section 5 and conclude the paper with some remarks in Section 6.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with vertex set $V$ and edge set $E$. We assume that $G$ is finite, simple and connected. We denote the number of vertices in $G$, the maximum degree of $G$, and the chromatic index of $G$ by $n$, $\Delta$, and $\chi'$, respectively. For a vertex $v \in V$, we use $d(v)$ and $N(v)$ to represent the degree of $v$ and the neighborhood of $v$ (i.e., the set of vertices adjacent to $v$), respectively. We start with two alternative definitions of the term “deficiency”.

Definition 1. The deficiency of graph $G$ is the minimum number of pendant edges whose attachment to $G$ makes an interval colorable supergraph of $G$, where a pendant edge is an edge with one endpoint in $G$ and the other one is a new vertex of degree one in the supergraph.

Figure 1 presents a supergraph of the complete graph of order three ($K_3$) obtained by attaching one pendant edge and an interval edge-coloring of it. Solid edges belong to the original graph $K_3$, while the dashed edge is a pendant edge. The color numbers are given next to the edges. Note that $K_3$ is originally not interval colorable but after being augmented by one pendant edge, it becomes interval colorable. Therefore, the deficiency of $K_3$ is one.

![Figure 1: An interval coloring of $K_3$ augmented by one pendant edge.](image1.png)

The second interpretation of the deficiency requires several more definitions:

Definition 2. Given a finite subset $A$ of $\mathbb{N}$, the deficiency of $A$, denoted by def$(A)$, is the number of integers in the interval $[\min(A), \max(A)]$ not belonging to $A$, i.e.,

$$\text{def}(A) = \max(A) - \min(A) - |A| + 1,$$

where $\min$ and $\max$ operators return the smallest and the largest elements in the set, respectively.

Note that if def$(A) = 0$, $A$ is an interval.

Definition 3. Let $c : E \rightarrow \mathbb{N}$ be a proper edge-coloring of $G$ and let $v \in V$. The deficiency of coloring $c$ at vertex $v$, denoted by def$(G, c, v)$, is the deficiency of the set of colors assigned to edges incident to $v$ under $c$.

Definition 4. Let $c : E \rightarrow \mathbb{N}$ be a proper edge-coloring of $G$. The deficiency of coloring $c$ is the sum of deficiencies of all vertices in $G$ under $c$, that is,

$$\text{def}(G, c) = \sum_{v \in V} \text{def}(G, c, v).$$
Definition 5. The deficiency of graph $G$ is the minimum of the deficiencies of all possible proper edge-colorings of $G$, i.e.,

$$
def(G) = \min_{c \in C} \text{def}(G, c),$$

where $C$ represents the set of all proper edge-colorings of $G$.

The equivalence of the two definitions of the deficiency, i.e., Definition 1 and Definition 5, is given in [18].

Lastly, we note that an edge-coloring $c$ of $G$ using $J$ different colors is called a $J$-edge-coloring of $G$. Alternatively, the number of colors, $J$, is referred to as the span of $c$. If there exists such an edge-coloring with deficiency zero, then $G$ is called interval $J$-colorable. The minimum (maximum) span is defined as the minimum (maximum) of the spans among all interval colorings of $G$.

3 Literature Review

The edge-coloring problem is first studied by Tait [43] in 1880 to prove the well-known “Four-Color Conjecture”. Vizing [45] proves that any simple graph can be colored with $\Delta + 1$ colors, while König [28] shows for bipartite graphs that $\Delta$ colors are sufficient. Nemhauser and Park [33] introduce the first IP formulation for the edge-coloring problem which is a “matching-based formulation”. Later, Lee and Leung [30] present a “multi-matching formulation” for the edge-coloring problem.

The interval coloring problem was first introduced by Asratian and Kamalian [5] in 1987. The first addressed question about the problem is its complexity. Asratian and Kamalian [5] show that deciding whether a regular graph is interval colorable is NP-complete. In [42], Sevastjanov proves that it is also NP-complete to decide if a given bipartite graph admits an interval coloring. Giaro and Cubale [13] strengthen this result by showing that the problem of deciding interval $\Delta$-colorability of a bipartite graph is easy if $\Delta \leq 4$ and becomes NP-complete for $\Delta \geq 5$.

Some restricted graph classes have been shown to be interval colorable, and the minimum and maximum spans are known for some other graph classes. All trees, complete bipartite graphs [4, 24], regular bipartite graphs [5, 6], doubly convex bipartite graphs [4, 22], grids [14], fans graphs with $n > 3$ [18], Mobius ladders [35], $n$-dimensional cubes [36], 2-processor and 3-processor bipartite graphs [17], bipartite cacti [16], outerplanar bipartite graphs [15], $(2, \Delta)$-biregular bipartite graphs [21] [23] and some classes of $(3,4)$-biregular bipartite graphs [3] [39] [46] are interval colorable. Exact values of the minimum and maximum span parameters are proven for some classes of trees, complete bipartite graphs [5] [24] and Mobius ladders [35]. For general graphs, the bounds on these parameters are examined in [1, 6, 18].

The deficiency is first studied by Giaro et al. [18]. Although it is NP-hard to determine the deficiency of a graph in general [13], exact values of the deficiency has been determined for some special families of graphs such as cycles, complete graphs, wheels and broken wheels [18], generalized $\theta$-graphs [12], and Hertz graphs [17]. Bounds on the deficiency are known for other classes of graphs. For instance, Giaro et al. [18] provide a lower bound for $\Delta$-regular graphs with odd $n$, while Giaro et al. [17] and Schwartz [41] derive an upper bound for rosettes and regular graphs, respectively. On the other hand, Petrosyan [37] shows that Eulerian multigraphs with odd $|E|$ have no interval coloring.

Only a few papers are devoted to the development of algorithms to compute the deficiency. Bouchard et al. [8] propose a tabu search algorithm to heuristically solve (MinDef). They also derive some lower bounds on the deficiency and the span of edge-colorings with minimum deficiency. They conduct computational experiments with some random graphs (with up to 1000 vertices) and some families of graphs with known deficiencies. In this body of work, the closest study to ours is [11] where the authors present several IP formulations as well as a CP formulation of (MinDef). They test the performance of their models with some
random graphs (with up to 100 vertices), and a complete collection of connected simple graphs with \(4 \leq n \leq 8\). In their experiments, they use \(K = 3n - 4\) which is a conjectured upper bound on the maximum span. They observe that the CP model is significantly better than the IP models. Later, Altinakar et al. [2] extends this work to improve the CP model by introducing a set of symmetry breaking constraints, based on graph automorphisms. However, all existing algorithms are able to solve the (MinDef) problem optimally only for very small graphs (e.g., when \(n \leq 10\)). Moreover, they usually fail to prove the optimality of solutions with large deficiencies (due to weak lower bounds), and to construct solutions with small deficiencies, which are the drawbacks that our study addresses.

4 Integer Programming Formulations

Let \(K\) be the set of available colors and let \(K = |K|\). Without loss of generality, we assume that \(K = \{0, 1, \ldots, K - 1\}\) and \(K \leq |E|\). Note that the given \(K\) value might be much smaller than \(|E|\) based on the considered application. For instance, in the task scheduling example discussed in the Introduction, the number of job interviews (edges) could be much larger than the number of available time slots (colors). Moreover, we assume that \(K \geq \Delta + 1\) so that the minimum deficiency problem is always feasible, as Vizing’s Theorem [45] guarantees the existence of a proper edge-coloring with \(\Delta + 1\) colors.

We start with the most natural integer programming (IP) formulation of the problem, which is also presented in [1]. We define binary decision variable \(x_{ijk} = 1\) if edge \(\{i, j\} \in E\) is given color \(k \in K\), and 0 otherwise. We introduce decision variables \(s_i\) and \(S_i\) for \(i \in V\) to represent the minimum and maximum color in the set of colors assigned to edges incident to vertex \(i\), respectively. We refer to the \(x\) variables as “edge-coloring variables”, while we call the \(s\) and \(S\) variables “deficiency variables”. Then, the (MinDef) problem can be formulated as follows:

\[
\begin{align*}
\text{(IP1): } \min & \sum_{i \in V} \left( S_i - s_i - d(i) + 1 \right) \\
\text{s.t. } & \sum_{j \in N(i)} x_{ijk} \leq 1, \quad i \in V, \ k \in K, \quad (1a) \\
& \sum_{k \in K} x_{ijk} = 1, \quad \{i, j\} \in E, \quad (1b) \\
& S_i \geq \sum_{k \in K} (k \cdot x_{ijk}), \quad i \in V, \ j \in N(i), \quad (1c) \\
& s_i \leq \sum_{k \in K} (k \cdot x_{ijk}), \quad i \in V, \ j \in N(i), \quad (1d) \\
& S_i - s_i \geq d(i) - 1, \quad i \in V, \quad (1e) \\
& x_{ijk} \in \{0, 1\}, \quad \{i, j\} \in E, \ k \in K, \quad (1f) \\
& 0 \leq s_i \leq K - d(i), \quad i \in V, \quad (1g) \\
& d(i) - 1 \leq S_i \leq K - 1, \quad i \in V. \quad (1h)
\end{align*}
\]

The objective function minimizes the sum of the deficiencies over all vertices, which is the definition of minimum deficiency. Constraints (1a) guarantee that adjacent edges take different colors. Constraints (1b) ensure that every edge takes exactly one color. These two sets of constraints reveal the property of being a proper edge-coloring. Constraints (1c) enforce \(S_i\) to be greater than or equal to the maximum color assigned to the edges incident to vertex \(i\). Similarly, constraints (1d) enforce \(s_i\) to be less than or equal to
the minimum color assigned to the edges incident to vertex $i$. Constraints (1e) are valid because at least 
$d(i)$ colors must be used to color the edges incident to vertex $i$. The remaining constraints provide variable 
bounds. Note that as $x$ variables are binary-valued, the constraints together with the objective function 
force $s$ and $S$ variables to take on integer values. As such, $s$ and $S$ variables can be taken to be continuous 
variables in the formulation.

A well-known difficulty in solving such an “edge-based” formulation is the symmetry with respect to the 
colors. For instance, if we consider a given coloring of the graph, we can get many other feasible solutions 
to the formulation by just changing the labels of the colors. However, they all correspond to the same solution. 
(This issue is further addressed in Section 4.3.) Another major problem with the above formulation is that its 
linear programming (LP) relaxation, which we denote as (LP1), is very weak. In fact, we have the following 
observation.

**Proposition 1.** The optimal objective value of (LP1) is zero.

**Proof.** Let $\hat{x}_{ijk} = 1/K$ for all $\{i, j\} \in E$ and $k \in K$. Also, let

$$\hat{s}_i = \begin{cases} 
(K - 1)/2, & \text{if } K \geq 2d(i) - 1 \\
K - d(i), & \text{otherwise}
\end{cases}$$

and $\hat{S}_i = \hat{s}_i + d(i) - 1$ for all $i \in V$. Then, it is easy to see that $(\hat{x}, \hat{s}, \hat{S})$ is feasible to (LP1). Constraints 
(1a), (1b) and (II) hold at $\hat{x}$ by construction. Note that (1c) and (1d) reduce to $\hat{s}_i \leq (K - 1)/2 \leq \hat{S}_i$, $i \in V$. 
Also, by the assumption $K \geq \Delta + 1$, we have $K \geq d(i) + 1$. Thus, (1c), (1d), (1g) and (1h) are satisfied 
by $(\hat{s}, \hat{S})$. Finally, (1e) is tight at $(\hat{s}, \hat{S})$, so the objective value is zero. As the objective value is always 
nonnegative, $(\hat{x}, \hat{s}, \hat{S})$ is an optimal solution and zero is the optimal objective value of (LP1).

When (IP1) is solved via a commercial solver, the branch-and-bound algorithm might take prohibitively 
long time because weak relaxation bounds lead to a large number of nodes in the branch-and-bound tree. 
Therefore, we next develop and analyze alternative formulations of the problem with tighter LP relaxations.

### 4.1 Improved model of deficiencies

We propose to replace the deficiency variables with a new type of variables to obtain tighter LP relaxations. 
First, for each $i \in V$, we define the following set of intervals:

$$\mathcal{I}^i = \{ [\ell, u] : \ell, u \in \mathbb{Z}, \ 0 \leq \ell, u \leq K - 1, \ u - \ell \geq d(i) - 1 \}.$$ 

Each interval $[\ell, u] \in \mathcal{I}^i$ represents a possible pair of minimum color $\ell$ and maximum color $u$ for the set of 
edges incident to vertex $i$. Also, for each $k \in K$, we represent the set of intervals for vertex $i \in V$ which 
include color $k$ by

$$\mathcal{I}^i(k) = \{ [\ell, u] \in \mathcal{I}^i : k \in [\ell, u] \}.$$ 

Then, we define the binary variables

$$y_{i,[\ell,u]} = \begin{cases} 
1, & \text{if interval } [\ell, u] \text{ is chosen for vertex } i \\
0, & \text{o.w.}
\end{cases}, \quad i \in V, \quad [\ell, u] \in \mathcal{I}^i.$$ 

Using $y$ variables as the deficiency variables, we formulate the problem as:
Constraints (2a) ensure that each edge takes exactly one color, while constraints (2b) choose exactly one interval (from the set of eligible intervals) for each vertex. Then, constraints (2c) guarantee that not only adjacent edges take different colors, but also edges incident to a vertex take colors from the set of colors included in the interval chosen for that vertex. The objective function minimizes the sum of deficiencies of the chosen intervals over all vertices. Therefore, in any optimal solution of this model, for each vertex, the minimum and the maximum color in its chosen interval must be used for an incident edge. The formal proof of (IP2) being a formulation of (MinDef) is provided in the Appendix.

Next, we show that the LP relaxation of (IP2), denoted as (LP2), provides tighter bounds than (LP1).

**Proposition 2.** Let \( \nu^*_1 \) and \( \nu^*_2 \) be the optimal objective values of (LP1) and (LP2), respectively. Then, \( \nu^*_1 \leq \nu^*_2 \). Moreover, the inequality can be strict.

**Proof.** As the problem is assumed to be feasible, \( \nu^*_2 \) is finite and always nonnegative by the definition of \( y \) variables. Then, as \( \nu^*_1 = 0 \) by Proposition 1, we have \( \nu^*_1 \leq \nu^*_2 \). Moreover, this inequality can be strict. In Figure 2 we provide an example for which \( \nu^*_1 = 0 \), while \( \nu^*_2 = 1 \) when \( K = 5 \).

![Figure 2: The complete bipartite graph \( K_{3,4} \).]
model. Next, in Section 4.2.2, we introduce the so-called matching variables to obtain a matching-based formulation of (IP1). Finally, we describe a column generation based algorithm to solve the LP relaxation of the matching-based formulation.

4.2.1 Valid inequalities

A matching is defined as a set of edges that have pairwise no common end-points. We denote the set of all matchings in the graph $G$ by $M$. Let “conv” denote the convex hull operator. The matching polytope of $G$ is the set defined as

$$P_M(G) = \text{conv}\{\chi^m \in \mathbb{R}^{|E|} : m \in M\},$$

where $\chi^m$ is the incidence vector of matching $m \in M$, i.e., $\chi^m_e = 1$ if $e \in m$ and 0 otherwise for $e \in E$. Let $\delta(i)$ denote the set of edges incident to vertex $i \in V$. The fractional matching polytope of $G$ is the set

$$P_{FM} = \{x \in \mathbb{R}^{|E|} : \sum_{e \in \delta(i)} x_e \leq 1, i \in V\}.$$

It is well-known that any vector $x$ of the matching polytope satisfies the blossom inequalities

$$\sum_{e \in E(S)} x_e \leq \frac{|S| - 1}{2}, \quad S \subseteq V, \ |S| \text{ odd},$$

where $E(S) := \{(i, j) \in E : i, j \in S\}$. Moreover, adding the blossom inequalities to the fractional matching polytope is sufficient to describe the matching polytope.

**Theorem 3 ([11]).** Let $P_M(G)$ be the matching polytope of graph $G$ as defined in (3). Then,

$$P_M(G) = \{x \in \mathbb{R}^{|E|} : \sum_{e \in \delta(i)} x_e \leq 1, \ i \in V, \ \text{and} \ \sum_{e \in E(S)} x_e \leq (|S| - 1)/2, \ S \subseteq V \text{ s.t. } |S| \text{ odd}\}.$$

We make the following observation about the feasible solutions of (IP2). As the set of edges taking the same color must form a matching in any feasible solution, for any fixed $k \in K$, the vector consisting of the variables $\{x_{ijk}\}_{(i,j) \in E}$ must belong to the matching polytope. Therefore, for each color $k \in K$, we may add the blossom inequalities to (IP2), and leading to the following enhanced formulation:

(IP2-B): \[
\begin{align*}
\min & \sum_{i \in V} \sum_{[\ell,u] \in \mathcal{I}^i} (u - \ell - d(i) + 1) y_{i,[\ell,u]} \\
\text{s.t.} & \quad (2a) - (2e), \\
& \sum_{(i,j) \in E(S)} x_{ijk} \leq \frac{|S| - 1}{2}, \quad S \subseteq V, \ |S| \text{ odd}, \ k \in K.
\end{align*}
\]

We denote the LP relaxation of (IP2-B) by (LP2-B).

**Proposition 4.** Let $\nu^*_2$ and $\nu^*_2B$ be the optimal objective values of (LP2) and (LP2-B), respectively. Then, $\nu^*_2 \leq \nu^*_2B$. Moreover, the inequality can be strict.

**Proof.** As (IP2-B) is obtained by adding some valid inequalities to (IP2), $\nu^*_2 \leq \nu^*_2B$ trivially holds. The smallest example showing that this inequality can be strict is the complete graph with three vertices, i.e., a triangle. For instance, for the triangle with $K = 3$, we obtain $\nu^*_2 = 0$, whereas $\nu^*_2B = 1$. \qed
For the triangle example used in the above proof, among all three blossom inequalities, it is actually sufficient to add only the one corresponding to \( S = V \) and \( k = 1 \) to (LP2) to get an optimal integer solution. This observation holds in general. Even though there are exponentially many blossom inequalities, most of them will not be binding in an optimal solution. For large instances, it is computationally impossible to add all blossom inequalities to the formulation a priori. Therefore, such inequalities should instead be added to the formulation as needed during the branch-and-bound algorithm to cut off optimal solutions of the LP relaxation. This yields a branch-and-cut algorithm, where a separation problem can be solved to identify violated blossom inequalities whenever a fractional LP solution is obtained at the nodes of the branch-and-bound tree. At any fractional node, either violated inequalities are added to the LP to cut off the current solution and the LP is re-solved, or branching is performed (i.e., no violated inequalities are found). More implementation details about our branch-and-cut algorithm are provided in Section 5.

Blossom inequalities can be separated in polynomial time. The first separation algorithm is devised by Padberg and Rao [34], whose running time is improved for dense graph in [19]. Later, a simpler and faster polynomial algorithm is proposed in [31]. In our implementation, we choose an alternative use of Padberg and Rao’s algorithm. Consider the perfect matching polytope of \( G \) which can be fully described as

\[
P_{PM}(G) = \{ x \in \mathbb{R}^{[E]}_+ : \sum_{e \in \delta(i)} x_e = 1, \ i \in V, \ and \ \sum_{e \in \delta(S)} x_e \geq 1, \ S \subseteq V \ s.t. |S| \ odd \},
\]

where \( \delta(S) := \{(i, j) \in E : i \in S, \ j \notin S \} \). The last set of inequalities are called the odd-set inequalities. Padberg and Rao [34] show that the odd-set inequalities can be separated in strongly polynomial time according to the following steps. (i) Construct a Gomory-Hu tree \( T \) based on the given fractional solution to be separated. (ii) For each edge \( e \) in the Gomory-Hu tree, check whether it is odd (even), where an edge \( e \) is odd (even) in the tree if both components of \( T - e \) have an odd (even) number of vertices. (iii) Find a minimum capacity odd edge \( \hat{e} \) in the Gomory-Hu tree. If its capacity is less than one, return the odd-set inequality constructed by using the set of vertices in one component of \( T - \hat{e} \) as the set \( S \). This procedure returns a most violated odd-set inequality if there exists any. Then, blossom inequalities can be separated for the matching polytope of \( G \) by separating the odd-set inequalities for the perfect matching polytope of a larger graph \( \tilde{G} = (\tilde{V}, \tilde{E}) \), constructed as follows. First creating a copy \( G' = (V', E') \) of \( G = (V, E) \), then let \( \tilde{V} = V \cup V' \) and \( \tilde{E} = E \cup E' \cup \{(i, i') : i \in V\} \). This procedure works by using the reduction between the two polytopes [40]. If we take \( \tilde{x} \in P_M(G) \) and construct

\[
\tilde{x}_{\hat{e}} = \begin{cases} \hat{x}_e, & \text{if } \hat{e} = e \in E \\ \hat{x}_e, & \text{if } \hat{e} = e' \in E' \\ 1 - \sum_{e \notin \delta(i)} \hat{x}_e, & \text{if } \hat{e} = (i, i') \end{cases}, \ \hat{e} \in \tilde{E}.
\]

then \( \tilde{x} \in P_{PM}(\tilde{G}) \). Using this relation, a blossom inequality which is valid for \( P_M(G) \) can be converted to an odd-set inequality which is valid for \( P_{PM}(\tilde{G}) \), and vice versa.

### 4.2.2 Dantzig-Wolfe reformulation

Next, we propose a matching-based formulation of (MinDef). We first define parameters

\[
a_{m,e} = \begin{cases} 1, & \text{if } e \in m \\ 0, & \text{o.w.} \end{cases}, \ m \in \mathcal{M}, \ e \in E,
\]
and
\[ b_{m,i} = \begin{cases} 1, & \text{if } \exists e = \{i,j\} \in E \text{ s.t. } e \in m, \ m \in \mathcal{M}, \ i \in V. \\ 0, & \text{o.w.} \end{cases} \]

Note that these parameters have the following relationship:
\[ b_{m,i} = \sum_{e = \{i,j\}; j \in N(i)} a_{m,e}, \ m \in \mathcal{M}, \ i \in V. \tag{5} \]

Next, we introduce the binary decision variables, referred to as the matching variables,
\[ x_{k,m} = \begin{cases} 1, & \text{if } m \text{ is chosen and given color } k \\ 0, & \text{o.w.} \end{cases}, \ k \in \mathcal{K}, \ m \in \mathcal{M}, \]

and replace the edge-coloring variables with the matching variables in (IP2) using the relation
\[ x_{ijk} = \sum_{m \in \mathcal{M}} a_{m,e} x_{k,m}, \ e = \{i,j\} \in E, \ k \in \mathcal{K}. \]

We thus obtain the following formulation of (MinDef):
\[
\text{(IP3): min } \sum_{i \in V} \sum_{[\ell,u] \in \mathcal{T}^i} (u - \ell - d(i) + 1) y_{i,[\ell,u]} \\
\text{s.t. } \sum_{[\ell,u] \in \mathcal{T}^i} y_{i,[\ell,u]} = 1, \ i \in V, \tag{6a} \\
\sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} a_{m,e} x_{k,m} = 1, \ e \in E, \tag{6b} \\
\sum_{m \in \mathcal{M}} x_{k,m} \leq 1, \ k \in \mathcal{K}, \tag{6c} \\
\sum_{m \in \mathcal{M}} b_{m,i} x_{k,m} \leq \sum_{[\ell,u] \in \mathcal{T}^i(k)} y_{i,[\ell,u]}, \ i \in V, \ k \in \mathcal{K}, \tag{6d} \\
x_{k,m} \in \{0,1\}, \ k \in \mathcal{K}, m \in \mathcal{M}, \tag{6e} \\
y_{i,[\ell,u]} \in \{0,1\}, \ i \in V, [\ell,u] \in \mathcal{T}^i. \tag{6f}
\]

(IP3) partitions the edges into disjoint matchings, assigns different colors to these matchings and colors all the edges in a matching by the matching’s color. Constraints (6b) enforce that each edge is covered by exactly one matching, which takes exactly one color, while constraints (6c) assign each color to at most one matching. Therefore, constraints (6d) guarantee that the edges incident to a vertex take different colors. The formal proof that (IP3) is a formulation of (MinDef) is provided in the Appendix. Note that as the edge-coloring variables are written as a convex combination of the matching variables, this formulation can be seen as the Dantzig-Wolfe reformulation of the problem (see the survey [44] for details).

We denote the LP relaxation of (IP3) by (LP3), and represent its optimal objective value by \( \nu_3^* \).

Proposition 5. (LP3) and (LP2-B) are equivalent, that is, \( \nu_3^* = \nu_2^{*B} \).

Proof. It immediately follows from Edmond’s matching theorem, Theorem [3]. \qed

10
This result combined with \textbf{Proposition 2} and \textbf{Proposition 4} implies that
\[ \nu_1^* \leq \nu_2^* \leq \nu_3^* = \nu_2^B. \]

Therefore, (IP3) has the strongest LP relaxation among the formulations presented so far. Even though (LP3) is tighter than (LP2) in general, we show that they have the same strength for bipartite graphs. Therefore, the blossom inequalities are not necessary for bipartite graphs, as (LP2) would also provide strong relaxation bounds. We first provide a known result about matchings which will be used in the proof of our claim.

\textbf{Theorem 6} ([40]). If $G$ is bipartite, then $P_M(G) = P_{FM}(G)$.

The fact that the matching polytope and the fractional matching polytope of a bipartite graph coincide leads to the following result.

\textbf{Proposition 7}. If the graph $G$ is bipartite, then (LP3) and (LP2) are equivalent, that is, $\nu_3^* = \nu_2^*$.  

\textbf{Proof}. As we already know from \textbf{Proposition 4} and \textbf{Proposition 5} that $\nu_2^* \leq \nu_3^* = \nu_2^B$, it is sufficient to show that $\nu_2^B \leq \nu_2^*$. Let $(\hat{x}, \hat{y})$ be an optimal solution of (LP2). For $k \in \mathcal{K}$, we define the vector $\tilde{x}_k \in \mathbb{R}^{|E|}$ such that $\tilde{x}_e^{jk} = \hat{x}_{ijk}$ for all $e = \{i,j\} \in E$. As $\hat{x}_k \in P_{FM}(G)$ and $G$ is bipartite, by \textbf{Theorem 6}, we have $\hat{x}_k \in P_M(G)$. Then, \textbf{Theorem 3} implies that $\hat{x}$ satisfies all blossom inequalities. Therefore, $(\hat{x}, \hat{y})$ is a feasible solution of (LP2-B), which shows that $\nu_2^B \leq \nu_2^*$. \hfill \qed

The number of $x$ variables in (IP3) grows exponentially in the size of the graph, and thus for moderate size graphs, it is not possible to enumerate all matchings in the graph, and hence explicitly construct the formulation (IP3), in a reasonable amount of time. However, most of these variables will take value zero in an optimal solution. Therefore, a possible solution approach is to apply a branch-and-price algorithm \cite{7}. The first requirement of branch-and-price is to solve the LP relaxation of the model via column generation \cite{9}, which we explain next.

Column generation solves an LP by iteratively adding the variables of the model. In order to start the column generation procedure, an initial feasible solution of the LP must be provided. In other words, a set of initial columns are given in the beginning. This problem is referred to as the (restricted) master problem or (restricted) master LP. At every step of the column generation algorithm, the current master LP is solved, its optimal dual solution values are passed to the so-called pricing problem, new columns are generated and added to the master LP, which is then re-solved. This loop is repeated until no more columns are generated.

We first explain how to generate new columns for (LP3), then suggest a way of getting an initial feasible solution. For convenience, and without loss of generality, we ignore the upper bounds on the $x$ and $y$ variables in the relaxation of constraints (6c) and (6d) as they are implied by the other constraints in (LP3). We denote the dual variables associated with the constraints (6a), (6b), (6c) and (6d) of (LP3) by $\theta$, $\gamma$, $\delta$ and $\Omega$, respectively. Then, the dual of (LP3) is as follows:

\[
\begin{align*}
\max & \quad \sum_{i \in V} \theta_i + \sum_{e \in E} \gamma_e + \sum_{k \in \mathcal{K}} \delta_k \\
\text{s.t.} \quad & \delta_k + \sum_{e = \{i,j\} \in E} a_{m,e} (\gamma_e + \Omega_{i,k} + \Omega_{j,k}) \leq 0, \quad m \in \mathcal{M}, \ k \in \mathcal{K}, \\
& \theta_i - \sum_{k \in \{\ell, \ldots, u\}} \Omega_{i,k} \leq u - \ell - d(i) + 1, \quad i \in V, \ [\ell, u] \in \mathcal{T}, \\
& \Omega_{i,k} \leq 0, \quad i \in V, \ k \in \mathcal{K}, \\
& \delta_k \leq 0, \quad k \in \mathcal{K}.
\end{align*}
\]
Let \((\theta^*, \gamma^*, \delta^*, \Omega^*)\) be an optimal dual solution of the current LP relaxation at any step of the column generation and let \(c_{e,k}^* := \gamma_{e,k}^* + \Omega_{i,k}^* + \Omega_{j,k}^*\), for \(e = \{i, j\} \in E, k \in K\). The pricing problem is separable over \(k\), thus for each \(k \in K\) we have the following pricing problem:

\[
v_k^* := \max_{e \in E} \sum_{e \in E} c_{e,k}^* w_e \quad \text{s.t.} \quad \{e \in E : w_e = 1\} \in M,
\]

\(w \in \{0, 1\}^{|E|}\),

where the decision variable \(w_e = 1\) if edge \(e \in E\) is chosen, and 0 otherwise. Observe that this is a **Maximum Weight Matching** problem. Therefore, it is polynomially solvable, e.g., it can be efficiently solved by Edmond’s algorithm [11]. Let \(w^*\) be an optimal solution of the pricing problem for \(k \in K\). If \(v_k^* + \delta_k^* > 0\), then a new column (with negative reduced cost) is found. Specifically, a new column corresponding to the variable \(x_{k,m}\), where \(m = \{e \in E : w_e^* = 1\}\) is added to the master LP. In the case that more than one column is found, different strategies can be used to determine the columns to be added as long as at least one column is added. Also, one can pick a specific (static or dynamic) order to solve the pricing problems for \(k \in K\), and stop whenever a desired number of columns are found. In our implementation, at every iteration, we solve the pricing problems for all \(k \in K\) and add all of the columns found that have negative reduced cost. When \(v_k^* + \delta_k^* \leq 0\) for all \(k \in K\), an optimal solution of the current master LP is also optimal for (LP3).

In order to get a feasible solution of (LP3) to initialize the column generation procedure, we propose solving the LP relaxation of the **Minimum Cardinality Edge-Coloring** problem, which is

\[
(\text{EC}): \quad v^* := \min_{m \in M} \sum_{m \in M} x_m \quad \text{s.t.} \quad \sum_{m \in M} a_{m,e} x_m = 1, \quad e \in E, \quad x \in \mathbb{R}^{|M|}.
\]

As (EC) has exponentially many variables, we solve this model via column generation as well. Let \(\pi\) denote the dual variables and let \(\pi^*\) be an optimal dual solution at the current iteration. The pricing problem is again a maximum weight matching problem, which is equivalent to (8) with edge weights \(\pi^*\) in the objective. We find an initial feasible solution for (EC) (i.e., a set of matchings) as follows. First, we find a maximum cardinality matching (via Edmond’s blossom shrinking algorithm). Then, we remove these edges from the graph, find a maximum cardinality matching in the remaining graph, and repeat until all of the edges are removed. Suppose that we obtain \(x^*\) as an optimal solution of (EC), with objective value \(v^*\). Let \(M^* := \{m \in M : x_m^* > 0\}\). Then, we can initialize (LP3) with columns corresponding to \(x_{k,m}\) for all \(m \in M^*, k \in K\). The initial LP formed for (LP3) from this procedure is always feasible when \(K \geq \Delta + 1\), which is proven in the Appendix.

In order to solve (IP3) to optimality, column generation should be combined with branch-and-bound, which results in a branch-and-price algorithm. Specifically, in branch-and-price, a branch-and-bound tree search is done where a predetermined branching rule is performed on the integer variables in the problem. The search is initialized with the master LP containing no branching restrictions, which is formed by the columns in a given feasible LP relaxation solution. At each node of the branch-and-bound tree, the master LP, augmented by branching constraints, is solved via column generation. If the optimal master LP solution
does not satisfy integrality constraints, then branching is performed. As we do not include the branch-and-price algorithm in our computational experiments, we do not provide further details of the algorithm and refer the reader to [7]. However, we note some difficulties in deriving a branch-and-price algorithm. The most important issue is that deriving an appropriate branching scheme in a column generation context can be non-trivial. The conventional integer programming branching techniques may not be effective because fixing variables can destroy the structure of the pricing problem ([7]). Therefore, it is crucial to use special (problem-specific) branching schemes to preserve the pricing problem structure and to yield balanced search trees. For many other practical issues such as stabilization of the column generation, early termination of the master LP, adaptation of primal heuristics and preprocessing techniques, and column management, we refer the reader to [44].

4.3 Symmetry breaking

A natural way to remove symmetry in a problem is to add some symmetry breaking inequalities. There are two commonly used approaches to manage symmetry breaking inequalities: generating dynamic symmetry breaking inequalities during the solution process; and adding static symmetry breaking inequalities to the initial formulation (explicitly or implicitly), cutting some of the symmetric solutions while keeping at least one optimal solution ([32]). In order to reduce the symmetry in our problem, we use the second approach. For each \( k \in \mathcal{K} \), we introduce a new binary variable \( w_k \) which takes value 1 if color \( k \) is used, and 0 otherwise. Then, we enforce the use of only consecutive colors starting from 0. In other words, we do not use color \( k + 1 \) unless color \( k \) is used. Also, as we need to use at least \( \Delta \) different colors, we can fix the first \( \Delta \) of \( w \) variables to 1.

In order to have a fair comparison, we apply these symmetry breaking improvements to all of the formulations that we have presented. In the first model, (IP1), we replace (1a) with

\[
\sum_{j \in N(i)} x_{ijk} \leq w_k, \quad i \in V, \; k \in \mathcal{K},
\]

and also add the following constraints:

\[
\begin{align*}
w_k &= 1, \quad k = 0, \ldots, \Delta - 1, \\
w_k &\leq w_{k-1}, \quad k = \Delta, \ldots, K - 1, \\
w_k &\in \{0, 1\}, \quad k \in \mathcal{K}.
\end{align*}
\]

(10)

(11)

(12)

For the models (IP2) and (IP2-B), we add not only (10)-(12) but also

\[
\sum_{[\ell,u] \in \mathcal{I}(k)} y_{i,[\ell,u]} \leq w_k, \quad i \in V, \; k \in \mathcal{K}.
\]

Lastly, for the (MC) model, we replace (6c) with

\[
\sum_{m \in \mathcal{M}} x_{km} \begin{cases} = 1, & k = 0, \ldots, \Delta - 1, \\ \leq 1, & k = \Delta, \\ \leq \sum_{m \in \mathcal{M}} x_{k-1,m}, & k = \Delta + 1, \ldots, K - 1. \end{cases}
\]

(13)

Note that these modifications do not change the pricing problem given in [5], but affects the criteria used to
detect the new columns. Now, a new column would be added to the master LP if \( v^*_k + \tilde{\delta}^*_k > 0 \), where \( v^*_k \)
denotes the optimal value of the pricing problem for \( k \in \mathcal{K} \) as before, whereas the value \( \tilde{\delta}^*_k \) is calculated
using optimal values \( \delta^* \) of the dual variables associated with (13) as
\[
\tilde{\delta}^*_k = \begin{cases} 
\delta^*_k, & \text{if } k = K - 1 \text{ or } k < \Delta \\
\delta^*_k - \delta^*_{k+1}, & \text{o.w.}
\end{cases}, \quad k \in \mathcal{K}.
\]
Also note that the feasible solution provided in Proposition 10 remains feasible for (LP3) after the addition
of the symmetry breaking constraints.

In order to eliminate more symmetry, we consider fixing the color of one edge. As a candidate, we
choose an edge \( \{i, j\} \) with maximum \( d(i) + d(j) \) value. However, as it is not possible to fix an arbitrary
color for the chosen edge, we restrict the colors allowed for that edge to the set \( \{0, 1, \ldots, \lfloor (K - 1)/2 \rfloor \} \).

5 Computational Experiments

Test instances. We perform our numerical experiments on a test data set consisting of randomly generated
problem instances for which the expected edge density of the graph (measured as \( D = \frac{2|E|}{n(n-1)} \)) takes values
0.2, 0.5 and 0.8. As stated in [8], which we also observe in our numerical experiments, graphs with odd
number of vertices are considered as more challenging because they typically have larger deficiencies than
similar ones with even number of vertices. Therefore, we analyze more instances with odd number of
vertices. In particular, we consider graphs with the number of vertices \( n \in \{11, 15, 19, 20, 23, 27, 30, 31\} \).
For each \( (D, n) \) combination, we generate five different graphs. In order to obtain larger deficiencies, and
thus more challenging instances, we use \( K = \Delta + 1 \) in some experiments.

The deficiency of our test instances (obtained by solving the problems to optimality without time limit) are given in Table 1, where the values in the curly brackets correspond to the five different instances generated for a fixed \( (D, n) \) combination. Note instances with higher density have higher deficiencies. Also, as mentioned before, the instances with even \( n \) have lower deficiencies; in particular they are all interval colorable.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( D = 0.2 )</th>
<th>( D = 0.5 )</th>
<th>( D = 0.8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 1, 0}</td>
<td>{2, 5, 1, 1}</td>
</tr>
<tr>
<td>15</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 6, 1, 3}</td>
</tr>
<tr>
<td>19</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 0, 0}</td>
<td>{8, 3, 5, 7}</td>
</tr>
<tr>
<td>20</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 0, 0}</td>
<td>{8, 0, 0, 0}</td>
</tr>
<tr>
<td>23</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 1, 0, 0}</td>
<td>{8, 12, 7, 3}</td>
</tr>
<tr>
<td>27</td>
<td>{0, 0, 0, 0}</td>
<td>{1, 1, 0, 0}</td>
<td>{9, 8, 8, 9}</td>
</tr>
<tr>
<td>30</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 0, 0, 0}</td>
</tr>
<tr>
<td>31</td>
<td>{0, 0, 0, 0}</td>
<td>{0, 3, 0, 0}</td>
<td>{9, 15, 11, 12}</td>
</tr>
</tbody>
</table>

Table 1: Deficiency of the instances when \( K = \Delta + 1 \)

Constraint programming formulation. Constraint programming (CP) is also widely used to model graph
coloring problems, especially for vertex-coloring. As noted in [20], the combinatorial structure of coloring
problems makes CP approaches often efficient and competitive with respect to the IP ones; however standard
CP approaches lack efficient mechanisms to compute tight lower bounds and to guide the search towards
the optimal solution. Therefore, we also compare our IP models with a CP model.
We introduce decision variables $z_{ij}$ to represent the color given to edge $\{i, j\} \in E$, and decision variables $\eta_i$ to denote the deficiency of vertex $i \in V$. Also, we again use decision variables $s_i$ and $S_i$ to represent the minimum and maximum color in the set of colors assigned to edges incident to vertex $i \in V$, respectively. Then, a natural constraint programming formulation for (MinDef) problem follows as:

\[(CP): \min \sum_{i \in V} \eta_i \]
\[\text{s.t.} \quad \text{All Diff} (z_{ij}), \quad i \in V, \quad (14a)\]
\[S_i = \max_{j \in N(i)} (z_{ij}), \quad i \in V, \quad (14b)\]
\[s_i = \min_{j \in N(i)} (z_{ij}), \quad i \in V, \quad (14c)\]
\[\eta_i = S_i - s_i + 1 - d(i), \quad i \in V, \quad (14d)\]
\[z_{ij} \in \{0, \ldots, K - 1\}, \quad \{i, j\} \in E, \quad (14e)\]
\[n_i \in \{0, \ldots, K - d(i)\}, \quad i \in V, \quad (14f)\]
\[s_i \in \{0, \ldots, K - d(i)\}, \quad i \in V, \quad (14g)\]
\[S_i \in \{d(i) - 1, \ldots, K - 1\}, \quad i \in V. \quad (14h)\]

The objective function minimizes the sum of the deficiencies over all vertices. Constraints (14a) use “All Different” predicate to enforce that adjacent edges take different colors. Constraints (14b), (14c) and (14d) are necessary to define $S$, $s$ and $\eta$ variables, respectively. The remaining constraints provide variable bounds. We note that the above formulation is also used in [1]. As in the IP models, we also introduce the following symmetry breaking constraints:

\[|\{i, j\} \in E : z_{ij} = k| \geq 1, \quad k = 0, \ldots, \Delta - 1, \quad (15)\]
\[z_{i^*, j^*} \leq \lfloor (K - 1)/2 \rfloor, \quad \text{where } \{i^*, j^*\} \text{ is an edge in } E \text{ with maximum } d(i) + d(j) \text{ value.} \]

Implementation details. We implement all algorithms in C++ using IBM ILOG CPLEX 12.4 for solving all LPs and IPs, IBM ILOG CP OPTIMIZER 12.4 for solving CPs, and LEMON Graph Library 1.2.3 [10] for seeking maximum weight matchings and constructing Gomory-Hu trees. We run all experiments using a single thread on a Linux workstation with 3.16 GHz Intel Xeon CPUs and 8 GB memory. For all runs, we impose a solution time limit of one hour.

In order to solve (IP2-B), we embed the generation of blossom inequalities within a branch-and-bound algorithm, leading to a branch-and-cut algorithm. However, before starting the branch-and-cut algorithm, we first solve the LP relaxation of (IP2-B) via a cutting-plane algorithm. At every iteration, we solve the LP, generate the blossom cuts among which we add the ones violated by the current LP solution to the LP. For each available color, we only add the most violated blossom cut, if there exists any. The cutting-plane algorithm stops when no more violated blossom cuts are found. In order to limit the number of cuts added at this phase, once the LP solve is done, we remove all the cuts that are not tight at the optimal LP solution. The purpose of this first phase implementation is that CPLEX can generate its own cuts based on the constraints in the given model formulation, so it can generate more and/or stronger cuts by using more information about the problem. Then, in the second phase, we apply the branch-and-cut algorithm where the addition of blossom cuts is implemented within a UserConstraintCallback in CPLEX to cut off fractional solutions.
5.1 Comparison of alternative approaches

We first compare four different formulations, namely (IP1), (IP2), (IP2-B) and (CP), in terms of optimality gaps obtained at the end of the given time limit. In Figure 3, we present the (frequency) histograms of the absolute optimality gaps (i.e., the difference of the best upper and lower bounds) obtained by the model formulations for different density values. Note that for each density level, there are 40 instances in total. Some gap intervals are presented in a combined fashion in order to have a more compact histogram. Also, if no upper bound is found within the time limit of one hour (i.e., no feasible solution), then the absolute gap is denoted by “∞”.

![Histograms of absolute optimality gaps for different density levels]

(a) $D = 0.2$

(b) $D = 0.5$

(c) $D = 0.8$

Figure 3: Absolute optimality gaps of the models for three density levels

We observe that the problem difficulty increases with $D$, in the sense that we fail more often in solving instances to optimality and end up with larger optimality gaps as $D$ increases. As such, all the formulations perform well for the instances with $D = 0.2$, whereas (IP2-B) clearly outperforms the others for the cases with $D = 0.5$ and $D = 0.8$, solving all of the instances except one. This is mainly because the instances with larger densities have larger deficiencies (see Table 1), and for such instances the strength of the relaxation bounds become more important.

We next compare the performance of the formulations in terms of solution times. In Figure 4, we plot the empirical cumulative distribution function, which we refer to as the CDF plot, of the solution times (in seconds) on logarithmic scale. That is, after sorting the logarithms of the solution times in increasing order, for each $t \in \{1, 2, \ldots, 40\}$ we plot a data point $(T, t)$ where $T$ is the sum of the first $t$ time values.

We find that (CP) is significantly faster than the other methods for the instances with $D = 0.2$. On the
other hand, when $D = 0.8$, (IP2-B) not only solves many more instances but also provides a significant reduction in the solution time. For the middle density level, (IP2-B) is either comparable with (CP) or performs much better. We also observe that all of the instances that (CP) can solve within the time limit have zero deficiencies.

In Table 2, we provide some statistics about the number of nodes in the branch-and-bound tree for our IP models, including the percentage of the instances solved at the root node, the median of the number of nodes explored in the tree, and the percentage of the instances where more than 1000 nodes are processed, in the columns labeled as “At Root”, “Median” and “≥ 1000”, respectively. We see that (IP2-B) solves more than 85% of the instances at the root node, which shows the strength of its LP relaxation. Overall, (IP2-B) explores very few nodes in the branch-and-bound tree, whereas (IP1) and (IP2) usually lead to large branch-and-bound trees. As (IP2-B) performs significantly better than (IP1) and (IP2), we do not consider (IP1) and (IP2) in the remaining experiments.

From these analyses, it appears that (CP) is able to find good feasible solutions quickly, but it has difficulties in proving the optimality. On the other hand, (IP2-B) provides very strong relaxation bounds, thus performs better for the instances with larger deficiencies. The solution times and the number of branch-and-bound nodes provide some evidence on the ability of (IP2-B) to find good feasible solutions. In order to get a better idea about upper bound generation, we look at the time that the last (i.e., the best integer
Table 2: The number of node statistics for the IP formulations

<table>
<thead>
<tr>
<th>$D$</th>
<th>Model</th>
<th>At Root</th>
<th>Median</th>
<th>$\geq 1000$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>(IP1)</td>
<td>10.0</td>
<td>2316</td>
<td>57.5</td>
</tr>
<tr>
<td></td>
<td>(IP2)</td>
<td>42.5</td>
<td>12</td>
<td>20.0</td>
</tr>
<tr>
<td></td>
<td>(IP2-B)</td>
<td>85.0</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>0.5</td>
<td>(IP1)</td>
<td>2.5</td>
<td>448837</td>
<td>92.5</td>
</tr>
<tr>
<td></td>
<td>(IP2)</td>
<td>12.5</td>
<td>148611</td>
<td>80.0</td>
</tr>
<tr>
<td></td>
<td>(IP2-B)</td>
<td>90.0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.8</td>
<td>(IP1)</td>
<td>0.0</td>
<td>325550</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>(IP2)</td>
<td>0.0</td>
<td>277278</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>(IP2-B)</td>
<td>87.5</td>
<td>0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

feasible solution) is found. Figure 5 shows the CDF plot of the last feasible solution times (in seconds) on logarithmic scale for (IP2-B) and (CP), for $D = 0.5$ and $D = 0.8$. Note that as all of the instances with $D = 0.2$ have zero deficiency and both models solve all those instances in an hour, the last solution time is equal to the solution time, see Figure 4 (a).

![Figure 5](image.png)

Figure 5: Times when the last feasible solution is found by the models for $D = 0.5$ and $D = 0.8$

We see that (CP) is especially useful for obtaining integer feasible solutions for the low density cases. In fact, more careful analyses on the individual results lead to the conclusion that (CP) is able to discover good feasible solutions only for the instances with very small deficiencies. This explains the dominating performance of (CP) on all instances with $D = 0.2$ and on half of the instances with $D = 0.5$. For the other cases, (IP2-B) finds good solutions significantly faster.

5.2 Impact of the number of allowed colors

We next investigate how the deficiencies change as we increase the number of allowed colors, $K$. For this aim, we pick the instances from our test set that are not interval colorable when $K = \Delta + 1$, and use (IP2-B) and (CP) to find the deficiency values of these instances for some increased values of $K$. Note that the number of instances (out of 40) that are interval colorable are 40, 33 and 11 for $D = 0.2$, $D = 0.5$ and $D = 0.8$, respectively.
We consider increasing $K$ gradually: We first solve a problem with $K = \Delta + 1$ colors (with the time limit of one hour), then increase $K$ by the upper bound obtained on the deficiency when $K = \Delta + 1$. This helps us to use some information from the previous solve. Specifically, we do the following:

- For (CP): We solve the problem with $K = \Delta + 1$. Let “def$_{CP}$” denote the deficiency of the best solution found within the time limit. Then, we solve the problem with $K = \Delta + 1 + \text{def}_{CP}$, where we provide the best feasible solution of the previous solve as a starting solution to the CP solver.
- For (IP2-B): We solve the problem with $K = \Delta + 1$. Let “def$_{IP2-B}$” denote the deficiency of the best solution found within the time limit. Then, we solve the problem with $K = \Delta + 1 + \text{def}_{IP2-B}$. As in the CP case, we provide the best solution found in the previous solve as an initial solution to the IP solver. In addition, we reuse some blossom cuts generated in the previous solve to tighten the model.

We choose to reuse some of the cuts that are generated when solving the LP relaxation of the previous model; the ones that are tight at the optimal LP solution.

Table 3 includes all seven instances with $D = 0.5$ that are not interval colorable with $K = \Delta + 1$ colors. The columns labeled as “LB” and “UB” correspond to the lower and upper bound values on the deficiency reported by (IP2-B), respectively, while the “CP” column refers to the upper bounds on the deficiency found by (CP). The solution times in seconds are given in the last two columns. For each instance, in the “K” column, the first value is equal to $\Delta + 1$, whereas the second line provides “$\Delta + 1 + \text{def}_{IP2-B}, \Delta + 1 + \text{def}_{CP}$” (a single value is given if they are equal). The bounds for the instances that could not be solved within the time limit are shown in bold.

| n | $|E|$ | $\Delta$ | K | LB | UB | CP | IP Time | CP Time |
|---|---|---|---|---|---|---|---|---|
| graph_D05_1 | 11 | 30 | 7 | 8 | 1 | 1 | 0 | 0 |
| | | | 9 | 0 | 0 | 0 | 0 | 1 |
| graph_D05_2 | 23 | 117 | 13 | 14 | 1 | 1 | 3 | 3600 |
| | | | 15 | 0 | 0 | 4 | 13 | |
| graph_D05_3 | 27 | 174 | 16 | 17 | 1 | 1 | 34 | 3600 |
| | | | 18 | 0 | 0 | 38 | 3600 | |
| graph_D05_4 | 27 | 177 | 17 | 18 | 0 | 2 | 3600 | 3600 |
| | | | 20,19 | 0 | 0 | 493 | 3600 | |
| graph_D05_5 | 27 | 177 | 18 | 19 | 1 | 1 | 58 | 3600 |
| | | | 20,22 | 0 | 0 | 99 | 3600 | |
| graph_D05_6 | 31 | 232 | 19 | 20 | 3 | 3 | 335 | 3600 |
| | | | 23,25 | 0 | 3 | 3600 | 3600 | |
| graph_D05_7 | 31 | 247 | 20 | 21 | 1 | 1 | 165 | 3600 |
| | | | 22,25 | 0 | 0 | 513 | 3600 | |

Table 3: Impact of $K$ on deficiency for instances with $D = 0.5$

We observe that all the instances except one (which could not be solved) become interval colorable after adding only a few colors. Moreover, (IP2-B) is able to find an interval coloring for those instances in a reasonable amount of time, while (CP) fails in the majority of them.

As seen in Table 3 for graph_D05_6, none of the methods made any improvement on the upper bound, and the lower bound is zero. For $D = 0.8$, we do not report such “no improvement” instances. Table 4 illustrates the results for 18 of 29 instances with $D = 0.8$ that are not interval colorable when $K = \Delta + 1$.
Table 4: Impact of $K$ on deficiency for instances with $D = 0.8$

(IP2-B) proves that nine instances (the unhighlighted ones in the table) become interval colorable after allowing $\text{def}_{IP2-B}$ more colors. Note that $\text{def}_{IP2-B}$ is exact in all of the instances. Moreover, (IP2-B) proves that in three instances (the dark highlighted ones in the table), $\text{def}_{IP2-B}$ many additional colors are not sufficient to obtain an interval coloring. The light highlighted instances in the table could not be solved to optimality, although their upper bounds have been improved by either of the two methods. Finally, we remark that (IP2-B) performs significantly better than (CP) in terms of solvability and especially finding good feasible solutions. For the instances that (IP2-B) could not solve within the time limit, especially for the ones that do not appear in Table 4, we find that the most (or all) of the time limit has been spent in solving the LP relaxation due to our aggressive approach on the blossom cut generation.
5.3 Comparison of LP relaxations

In our final experiment, we compare the computational performance of solving the LP relaxations of (IP2-B) and (IP3). Although the optimal values of (LP2-B) and (LP3) are equal, the solution times of these models might affect the decision of which model to implement. We present the results of this experiment for \( K = \Delta + 1 \) and \( K = \Delta + 1 + \text{def} \) in Table 5, where def is the deficiency of the former case. As the number of instances that are not interval colorable when \( K = \Delta + 1 + \text{def} \) is very few for \( D = 0.2 \) and \( D = 0.5 \), we only consider the instances with \( D = 0.8 \). We provide the result only for three values of \( n \), which involve the instances whose LP relaxations take the longest time to solve.

(LP2-B) is solved via a cutting plane algorithm, while column generation is used to solve (LP3). As such, for the former, we report the number of cuts added to the LP (“Cuts”), the number of times the LP is solved (“Iters”) and the total time spent in seconds (“Time”), whereas for the latter, we report the number of columns added to the master problem (“Cols”), the number of times the master LP is solved (“Iters”) and the total time spent in seconds (“Time”). All the values correspond to the averages, i.e., each row of the table shows the averages over five instances.

<table>
<thead>
<tr>
<th>( K )</th>
<th>( D )</th>
<th>( n )</th>
<th>(LP2-B)</th>
<th></th>
<th>(LP3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cuts</td>
<td>Iters</td>
<td>Time</td>
</tr>
<tr>
<td>( \Delta + 1 )</td>
<td>0.2</td>
<td>23</td>
<td>26.0</td>
<td>4.0</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
<td>30.6</td>
<td>5.0</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td></td>
<td>26.2</td>
<td>4.2</td>
<td>0.3</td>
</tr>
<tr>
<td>( \Delta + 1 )</td>
<td>0.5</td>
<td>23</td>
<td>295.2</td>
<td>12.4</td>
<td>3.4</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
<td>722.8</td>
<td>20.6</td>
<td>22.4</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td></td>
<td>1094.4</td>
<td>27.0</td>
<td>73.2</td>
</tr>
<tr>
<td>( \Delta + 1 )</td>
<td>0.8</td>
<td>23</td>
<td>1121.4</td>
<td>20.4</td>
<td>31.9</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
<td>1571.0</td>
<td>25.0</td>
<td>104.7</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td></td>
<td>2357.2</td>
<td>30.6</td>
<td>348.4</td>
</tr>
<tr>
<td>( \Delta + 1 + \text{def} )</td>
<td>0.8</td>
<td>23</td>
<td>884.4</td>
<td>40.6</td>
<td>161.3</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
<td>992.6</td>
<td>39.4</td>
<td>337.5</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td></td>
<td>1615.2</td>
<td>61.2</td>
<td>1688.6</td>
</tr>
</tbody>
</table>

Table 5: Some statistics about solving the LP relaxation

These results indicate that the solution times for (LP2-B) and (LP3) are comparable. Although it appears that (LP3) might be more efficient for larger and denser instances, the time savings are not very significant for our test instances. Therefore, we do not implement a branch-and-price algorithm to test (IP3) in our numerical experiments.

6 Concluding Remarks

In this paper, we present exact solution algorithms via IP techniques to solve the problem of finding the minimum deficiency for general graphs. Starting with a natural IP formulation of the problem, we devise new formulations with tighter LP relaxations. We present a cutting plane algorithm and an alternative column generation algorithm to solve the LP relaxation of the resulting model. In order to solve the problem to optimality, we incorporate the cutting plane algorithm into a branch-and-bound tree and obtain a branch-
and-cut algorithm. Our computational results on a set of random instances indicate that our branch-and-cut algorithm can solve the instances of medium size efficiently. In particular, we find that the algorithm overcomes the two major drawbacks of the existing methods from the literature, namely finding good feasible solutions and providing strong relaxation bounds, thus proving the optimality of the solutions quickly. These improvements can be especially useful for testing the existing (or new) conjectures about the deficiency or the span of the edge-colorings with minimum deficiency, hence contributing to theoretical research in graph theory as well.

We perform the majority of our experiments using $K = \Delta + 1$ in order to work with instances having larger deficiencies. We find that almost all of the instances become interval colorable after the addition of a few more colors. Therefore, when $K$ is large, we propose a framework where we start with $K = \Delta + 1$ and gradually increase its value. This enables us to re-use some information from previous solves such as cutting planes and feasible solutions as warm starts.

In our numerical experiments, we observe that the cutting plane algorithm is comparable with the column generation algorithm in solving the LP relaxations. As such, we prefer not to incorporate the column generation into our final algorithm. However, we also recognize that the column generation might be more efficient for larger and denser instances, which merits further research. On the other hand, although our IP formulation significantly outperforms the CP formulation, a comparison with an improved CP model is also a subject of future research. Moreover, we next plan to explore some hybrid methods, such as developing a branch-and-cut-and-price algorithm and incorporating good lower bounds obtained via IP within a CP model.

Acknowledgments. We are grateful to Tinaz Ekim and Z. Caner Taskin for providing helpful discussions.

References

Appendix

Proposition 8. \( (IP2) \) is a formulation of \((\text{MinDef})\) problem.

Proof. Let \( c : E \rightarrow K \) be a proper edge-coloring of graph \( G = (V, E) \), where \( K = \{0, 1, \ldots, K - 1\} \).

First, we need to show that there is a feasible solution of \((IP2)\) whose objective function value is less than or equal to the deficiency of \( c \), which is

\[
def(G, c) = \sum_{i \in V} (\max_{j \in N(i)} c(i, j) - \min_{j \in N(i)} c(i, j) - d(i) + 1).
\]

For each \( i \in V \), we define

\[
\ell^*(i) = \min_{j \in N(i)} c(i, j) \quad \text{and} \quad u^*(i) = \max_{j \in N(i)} c(i, j).
\]

Also, we let

\[
x^*_{ijk} = \begin{cases} 
1, & \text{if } c(i, j) = k, \\
0, & \text{o.w.}, \quad \{i, j\} \in E, \ k \in K,
\end{cases}
\]

and

\[
y^*_{i, [\ell, u]} = \begin{cases} 
1, & \text{if } \ell = \ell^*(i) \text{ and } u = u^*(i), \\
0, & \text{o.w.}, \quad i \in V, \ [\ell, u] \in T^i.
\end{cases}
\]
At \((x^*, y^*)\), all constraints in (IP2), except (2c), are trivially satisfied by the definitions of \(x^*\) and \(y^*\). Now, we confirm that (2c) also holds at \((x^*, y^*)\). Let \(i \in V\) and \(k \in K\). Then, \(\sum_{j \in N(i)} x^*_{ijk} = 1\) if \(c(i, j) = k\) for some \(j \in N(i)\). In that case, we have \(\ell^*(i) \leq k \leq u^*(i)\), which means that \([\ell^*(i), u^*(i)] \in I^k(k)\). Hence, \(\sum_{[\ell, u] \in I^k(k)} y^*_{i, [\ell, u]} = 1\). Otherwise, as \(\sum_{j \in N(i)} x^*_{ijk} = 0\), the constraint is also satisfied. Therefore, \((x^*, y^*)\) is a feasible solution to (IP2), whose objective value is equal to

\[
\sum_{i \in V} \sum_{[\ell, u] \in I^k} (u - \ell - d(i) + 1) y^*_{i, [\ell, u]} = \sum_{i \in V} (u^*(i) - \ell^*(i) - d(i) + 1) = \text{def}(G, c).
\]

For the reverse direction, we need to show that any feasible solution of (IP2) corresponds to a proper edge-coloring of \(G\) with deficiency less than or equal to the objective function value. Let \((\hat{x}, \hat{y})\) be a feasible solution of (IP2). We define \(\hat{c} : E \rightarrow K\) as

\[
\hat{c}(i, j) = \sum_{k \in K} k \hat{x}_{ijk}, \quad \{i, j\} \in E.
\]

From (2a), we guarantee that each edge takes exactly one color from \(K\) in \(\hat{c}\). For \(i \in V\) and \(k \in K\), as we have

\[
\sum_{j \in N(i)} \hat{x}_{ijk} \leq \sum_{[\ell, u] \in I^k(k)} \hat{y}_{i, [\ell, u]} \leq \sum_{[\ell, u] \in I^k} \hat{y}_{i, [\ell, u]} = 1,
\]

\(\hat{c}\) is a proper edge-coloring. Next, for each \(i \in V\), let \(\hat{\ell}(i), \hat{u}(i) \in K\) such that \(\hat{y}_{i, [\hat{\ell}(i), \hat{u}(i)]} = 1\). Then, the objective function value at \((\hat{x}, \hat{y})\) is \(\sum_{i \in V} (\hat{u}(i) - \hat{\ell}(i) - d(i) + 1)\). We know that \(\hat{\ell}(i) \leq \hat{c}(i, j) \leq \hat{u}(i)\) for all \(i \in V, j \in N(i)\). Therefore, as \(\max_{j \in N(i)} \hat{c}(i, j) \leq \hat{u}(i)\) and \(\min_{j \in N(i)} \hat{c}(i, j) \geq \hat{\ell}(i)\), \(\text{def}(G, \hat{c})\) is smaller than or equal to the objective value of \((\hat{x}, \hat{y})\), which completes the proof. □

**Proposition 9.** (IP3) is a formulation of (MinDef) problem.

**Proof.** Let \(c : E \rightarrow K\) be a proper edge-coloring of graph \(G = (V, E)\), where \(K = \{0, 1, \ldots, K - 1\}\). First, we need to show that there is a feasible solution of (IP3) whose objective function value is less than or equal to the deficiency of \(c\), which is

\[
\text{def}(G, c) = \sum_{i \in V} \left( \max_{j \in N(i)} c(i, j) - \min_{j \in N(i)} c(i, j) - d(i) + 1 \right).
\]

For each \(i \in V\), let

\[
u_i^* = \max_{j \in N(i)} c(i, j) \quad \text{and} \quad \ell_i^* = \min_{j \in N(i)} c(i, j)
\]

and define

\[
y^*_{i, [\ell, u]} = \begin{cases} 1, & \text{if } \ell = \ell_i^* \text{ and } u = u_i^* \\ 0, & \text{o.w.} \end{cases}
\]

then it is easy to see that (6a) and (6b) are satisfied. Moreover, the objective function value of (IP3) is equal to \(\text{def}(G, c)\). Now, for each \(k \in K\), let \(E_k = \{ e \in E : c(e) = k \}\). Note that \(E_k\) might be empty for some \(k \in K\). Then, \(E_0, \ldots, E_{K-1}\) is a partition of \(E\), where \(E_k\) is a matching for each \(k \in K\). If we define

\[
x^*_{k, m} = \begin{cases} 1, & \text{if } m = E_k \text{ and } E_k \neq \emptyset \\ 0, & \text{o.w.} \end{cases}, \quad k \in K, \quad m \in M,
\]

then (6c) is satisfied. Next, we check the remaining constraints to show that \((x^*, y^*)\) is feasible to (IP3):

[25]
(6b): For \( e \in E \), \( \sum_{m \in M} \sum_{k \in K} a_{m,e}x_{k,m} = \sum_{k \in K} b_{E_k,e}x_{k,E_k} = x_{c(e),E_{c(e)}} = 1 \).

(6c): For \( k \in K \), \( \sum_{m \in M} x_{k,m} = x_{k,E_k} \leq 1 \).

(6d): For \( i \in V \), \( k \in K \), \( \sum_{m \in M} b_{m,i}x_{k,m} = b_{E_k,i}x_{k,E_k} = \begin{cases} 1, & \text{if } \exists \{i,j\} \in E \text{ s.t. } c(i,j) = k \\ 0, & \text{o.w.} \end{cases} \)

The constraint is satisfied in both cases as

## \[ b_{E_k,i}x_{k,E_k} = 1 \Rightarrow \ell_i^* \leq k \leq u_i^* \Rightarrow [\ell_i^*, u_i^*] \in \mathcal{I}^i(k) \Rightarrow \sum_{[\ell, u] \in \mathcal{I}^i} y_{i, [\ell, u]}^* = 1. \]

Therefore, \((x^*, y^*)\) is a feasible solution of (IP3) with objective function value \( \text{def}(G, c) \).

For the reverse direction, we need to show that any feasible solution of (IP3) corresponds to a proper edge-coloring of \( G \) with deficiency less than or equal to the objective function value. Let \((\hat{x}, \hat{y})\) be a feasible solution of (IP3). Letting

## \[ \hat{\delta}_{e,k} := \sum_{m \in M} a_{m,e} \hat{x}_{k,m}, \ e \in E, \ k \in K, \]

we define \( \hat{c} : E \rightarrow K \) as

## \[ \hat{c}(e) = \sum_{k \in K} k \hat{\delta}_{e,k}, \ e \in E. \]

(6b) implies that for any \( e \in E \), we have \( \sum_{k \in K} \hat{\delta}_{e,k} = 1 \). Then, as \( \hat{\delta}_{e,k} \) values are nonnegative integers, this means that each edge \( e \in E \) takes exactly one color, which is \( \hat{c}(e) \). This also shows that edge colors are actually taken from the set \( K \). Next, we show that \( \hat{c} \) is a proper edge-coloring. Let \( i \in V \) and \( k \in K \). Then, we have

## \[ \sum_{j \in N(i)} \hat{\delta}_{i,j,k} = \sum_{e=\{i,j\}: j \in N(i)} \sum_{m \in M} a_{m,e} \hat{x}_{k,m} = \sum_{m \in M} \hat{x}_{k,m} b_{m,i} \leq \sum_{[\ell, u] \in \mathcal{I}^i} y_{i, [\ell, u]} \leq \sum_{[\ell, u] \in \mathcal{I}^i} y_{i, [\ell, u]} \leq 1, \]

which means that each color can be used at most once for the set of edges incident to a vertex. Finally, letting

## \[ \hat{u}_i = \max_{j \in N(i)} \hat{c}(i, j) \text{ and } \hat{\ell}_i = \min_{j \in N(i)} \hat{c}(i, j) \]

for all \( i \in V \), we consider the deficiency of \( \hat{c} \):

## \[ \text{def}(G, \hat{c}) = \sum_{i \in V} (\hat{u}_i - \hat{\ell}_i - d(i) + 1). \]

For any \( i \in V \), (6d) for \( k = \hat{u}_i \) and for \( k = \hat{\ell}_i \) imply that \( \hat{y}_{i, [\ell, u]} = 1 \) for some \( [\ell, u] \in \mathcal{I}^i \) with \( \ell \leq \hat{u}_i \leq u \) and \( \ell \leq \hat{\ell}_i \leq u \), respectively. This combined with (6a) shows that \( \hat{y}_{i, [\ell, u]} = 1 \) for the interval \( [\ell, u] \in \mathcal{I}^i \) with \( u \geq \hat{u}_i \) and \( \ell \leq \hat{\ell}_i \). Hence, the objective function value of the solution \((\hat{x}, \hat{y})\) is greater than or equal to \( \text{def}(G, \hat{c}) \).

\[ \square \]

**Proposition 10.** Assume that \( K \geq \Delta + 1 \) and \( \Delta > 0 \). Let \( x^* \in \mathbb{R}^{\#M}_+ \) and \( v^* \) be an optimal solution and
the optimal value of (EC), respectively. Then, \((\hat{x}, \hat{y})\) with
\[
\hat{x}_{k,m} = \begin{cases} 
  x^*/v^*, & \text{if } k < \Delta \\
  (v^* - \Delta)x^*/v^*, & \text{if } k = \Delta \\
  0, & \text{otherwise}
\end{cases}, \quad k \in K, \ m \in M
\]

and
\[
\hat{y}_{i,\ell,u} = \begin{cases} 
  1, & \text{if } \ell = 0 \text{ and } u = K - 1 \\
  0, & \text{otherwise}
\end{cases}, \quad i \in V, \ [\ell,u] \in I^i
\]
is a feasible solution to (LP3).

Proof. Vizing’s Theorem \cite{45} implies that \(v^* \leq \Delta + 1\). Also, we have \(v^* = \sum_{m \in M} x^*_m\). Therefore, the bound constraints of (LP3) are satisfied by \((\hat{x}, \hat{y})\). Now, we show that the rest of the constraints in (LP3) also hold at \((\hat{x}, \hat{y})\):

\textbf{(6a):} For any \(i \in V\), only \(\hat{y}_{i,0,K-1} = 1\). Therefore, \(\sum_{[\ell,u] \in \mathcal{I}^i} \hat{y}_{i,\ell,u} = 1\), \(i \in V\).

\textbf{(6b):} Let \(e \in E\). As we have, for each \(m \in M\),
\[
\sum_{k \in K} \hat{x}_{k,m} = \sum_{k < \Delta} \hat{x}_{k,m} + \hat{x}_{\Delta,m} + \sum_{k > \Delta} \hat{x}_{k,m} = \Delta x^*_m/v^* + (v^* - \Delta)x^*_m/v^* = x^*_m,
\]
we obtain
\[
\sum_{m \in M} \sum_{k \in K} a_{m,e} \hat{x}_{k,m} = \sum_{m \in M} a_{m,e} \sum_{k \in K} \hat{x}_{k,m} = \sum_{m \in M} a_{m,e} x^*_m = 1.
\]

\textbf{(6c):} For any \(k < \Delta\),
\[
\sum_{m \in M} \hat{x}_{k,m} = \frac{1}{v^*} \sum_{m \in M} x^*_m = \frac{1}{v^*} v^* = 1.
\]
For \(k = \Delta\),
\[
\sum_{m \in M} \hat{x}_{\Delta,m} = \frac{v^* - \Delta}{v^*} \sum_{m \in M} x^*_m = \frac{v^* - \Delta}{v^*} v^* = v^* - \Delta \leq \Delta + 1 - \Delta = 1.
\]
For \(k > \Delta\), we have \(\hat{x}_{k,m} = 0\), \(m \in M\) so the rest of this type of constraints is also satisfied.

\textbf{(6d):} For any \(i \in V, k \in K\),
\[
\sum_{m \in M} b_{m,i} \hat{x}_{k,m} \leq \sum_{m \in M} \hat{x}_{k,m} = \begin{cases} 
  \sum_{m \in M} x^*_m/v^* = 1, & \text{if } k < \Delta \\
  \left(\frac{1}{v^*} - \Delta\right), & \text{if } k = \Delta \\
  0, & \text{otherwise}
\end{cases} \leq 1 = \sum_{[\ell,u] \in \mathcal{I}^i(k)} \hat{y}_{i,\ell,u}.
\]