Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 346 (2019) 613–624

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

A Branch and Price Algorithm for List Coloring Problem

Mauro Lucci[1,2] Graciela Nasini[1,3] Daniel Severín[1,4]

Depto. de Matemática, FCEIA, Universidad Nacional de Rosario, Argentina
CONICET, Argentina

Abstract

Coloring problems in graphs have been used to model a wide range of real applications. In particular, the List Coloring Problem generalizes the well-known Graph Coloring Problem for which many exact algorithms have been developed. In this work, we present a Branch-and-Price algorithm for the weighted version of the List Coloring Problem, based on the one developed by Mehrotra and Trick (1996) for the Graph Coloring Problem. This version considers non-negative weights associated to each color and it is required to assign a color to each vertex from predetermined lists in such a way the sum of weights of the assigned colors is minimum. Computational experiments show the good performance of our approach, being able to comfortably solve instances whose graphs have up to seventy vertices. These experiences also bring out that the hardness of the instances of the List Coloring Problem does not seem to depend only on quantitative parameters such as the size of the graph, its density, and the size of list of colors, but also on the distribution of colors present in the lists.

Keywords: List Coloring, Branch and Price, Weighted Problem.

1 Introduction

The Graph Coloring Problem (GCP) models a wide range of planning problems such as timetabling, scheduling, electronic bandwidth allocation and sequencing. Some applications impose additional constraints to the GCP, giving rise to known variants such as Equitable Coloring, Precoloring Extension, (γ, μ)-coloring and List Coloring, see e.g. [1,2]. Actually, List Coloring generalizes GCP, Precoloring Extension and (γ, μ)-coloring and has several specific applications such as channel allocation in wireless networks [3]. In many practical situations, colors have different weights (or costs) and it is required to find the coloring of minimum weight instead of minimum cardinality. Particularly, in [4], the design of workdays of drivers

[1] This work is supported by grants ANPCyT PICT-2016-0410 and PID-UNR ING538.

[2] Email: mlucci@fceia.unr.edu.ar

[3] Email: nasini@fceia.unr.edu.ar

[4] Email: daniel@fceia.unr.edu.ar

https://doi.org/10.1016/j.entcs.2019.08.054

in a public transport company is modeled as a Minimum Weighted List Coloring Problem (MWLCP).

Let $G = (V, E)$ be an undirected simple graph and $\mathscr{C}$ be a set of colors. A *coloring of* G is a function $f : V \to \mathscr{C}$ such that $f(u) \neq f(v)$ for every edge (u, v) of G. Given a coloring f of G, the *class color of* $j \in \mathscr{C}$, denoted by $f^{-1}(j)$, is the set of vertices v colored by j, i.e. such that $f(v) = j$. Clearly, each class color is a stable set of G and, therefore, any coloring can be thought as a partition of vertices into stable sets. The *active colors* of a coloring f, denoted by $\mathscr{A}$, are those ones assigned to some vertex, i.e. $\mathscr{A} \doteq \{j \in \mathscr{C} : f^{-1}(j) \neq \emptyset\}$. The GCP consists of finding a coloring of G with minimum cardinality of $\mathscr{A}$. This minimum is called *the chromatic number of* G, denoted by $\chi(G)$, and it is known that obtaining this number is $\mathcal{NP}$-hard for general graphs.

Despite its hardness, the need of obtaining concrete solutions to numerous applications motivated the development of several exact algorithms for GCP: combinatorial branch-and-bound such as DSATUR [5,6], branch-and-cut BC-COL [7] and branch-and-price LPCOLOR [8,9,10].

In the case of BC-COL, an Integer Linear Programming (ILP) compact formulation (*GCP-CF* from now on) based on classic vertex-color assignment variables is used. Instead, LPCOLOR is based on set-covering ILP formulation (*GCP-SC* from now on), where each variable represents a stable set of G. As the number of variables is usually exponential in the size of G, the resolution of the linear relaxation of GCP-SC is addressed by a column generation procedure.

In List Coloring, each vertex $v \in V$ has a preassigned *list* $L(v) \subset \mathscr{C}$ defining those colors that can be assigned to v. Formally, a *list coloring of* G is a coloring f of G with the additional condition that $f(v) \in L(v)$ for all $v \in V$. Note that classic colorings of G are list colorings for which $L(v) = \mathscr{C}$ for all $v \in V$.

Given a vector $w \in \mathbb{Z}_+^{\mathscr{C}}$, the MWLCP consists of finding a list coloring of G such that the sum of weights of the active colors, i.e. $\sum_{j \in \mathscr{A}} w_j$, is minimum. As colorings of G can be thought as particular cases of list colorings, MWLCP can be seen as a generalization of a *weighted version* of GCP. However, this weighted version can be trivially reduced to GCP since one can pick the cheapest $\chi(G)$ colors from $\mathscr{C}$ to obtain the coloring of minimum weight.

Clearly, MWLCP is $\mathcal{NP}$-hard in general graphs since it generalizes GCP. However, some known results reveal that MWLCP is indeed *harder* than GCP.

As a first remark, unlike GCP, which is feasible whenever $|\mathscr{C}| \geq \chi(G)$, MWLCP can be infeasible even if $|L(v)| \geq \chi(G)$ for all v. A well-known example is presented in Figure 1 where the available colors for each vertex are enclosed in braces.

Actually, asking whether G has a list coloring is $\mathcal{NP}$-complete even when restricted to instances where the size of lists is at most 3 and G is a cograph or complete bipartite [11]. In contrast, GCP can be solved on cographs and bipartites in linear time.

Another example where these problems differ in difficulty is when graphs are

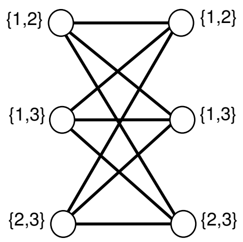

Fig. 1. An infeasible instance

parameterized by treewidth. For a given graph G with fixed treewidth t, there is a dynamic programming algorithm that finds $\chi(G)$ in polynomial time, while asking if G has a list coloring is W[1]-Hard [12].

Formulation GCP-CP (resp. GCP-SC) mentioned above can be extended to the MWLCP in such a way that they coincide when applied to instances of GCP. In this work, we present these *extended* ILP formulations, respectively called MWLCP-CP and MWLCP-SC, and we develop a Branch-and-Price algorithm to solve MWLCP via MWLCP-SC, by following some ideas provided in [8]. However, we have to take into account some additional issues that arise from the difference between GCP and MWLCP. In GCP every vertex can be colored with any element of $\mathscr{C}$ and all colors have the same weight, thus making the colors *indistinguishable* from each other. Clearly, this property is not met for MWLCP forcing us to consider one variable for each stable set and each color in MWLCP-SC. In spite of that, we designed a mechanism to take advantage of those cases where the indistinguishability between colors is partially met. Feasibility is another issue, as mentioned before.

In Section 2, we formally present the ILP formulations MWLCP-CP and MWLCP-SC. In Sections 3 and 4, we explain the main components that define our Branch-and-Price algorithm: the way LP relaxations are solved, how columns are generated and how the node branching is performed. In Section 5, computational experiments are conducted to evaluate the performance of our algorithm over random instances of different sizes and types. Finally, in Section 6, some conclusions are drawn.

2 ILP formulations for MWLCP

Consider an instance of the MWLCP, given by a graph $G = (V, E)$, a set of colors $\mathscr{C}$ with weights $w_j \in \mathbb{Z}_+$ for each $j \in \mathscr{C}$ and lists $L(v) \subset \mathscr{C}$ for each vertex v. W.l.o.g we can assume that G is connected and $\mathscr{C} = \bigcup_{v \in V} L(v)$.

For each color j, let $V_j \doteq \{v \in V : j \in L(v)\}$, G^j be the subgraph of G induced by V_j, I_j be the set of isolated vertices in G^j and $\mathscr{S}(G^j)$ be the set of maximal stable sets of G^j.

The first formulation is based on GCP-CF [7]. We have a binary variable x_{vj}, for each $v \in V$ and $j \in L(v)$, such that its value is 1 if and only if $f(v) = j$, and binary variables y_j such that $y_j = 1$ if j is an active color in f. Then, the MWLCP can be formulated as follows:

$$
\begin{aligned}
\text{(MWLCP-CF)} \quad & \min \sum_{j \in \mathscr{C}} w_j y_j \\
& s.t. \\
& \sum_{j \in L(v)} x_{vj} = 1 && \forall\, v \in V && (1) \\
& x_{uj} + x_{vj} \le y_j && \forall\, j \in \mathscr{C},\ (u,v) \in E(G^j) && (2) \\
& x_{vj} \le y_j && \forall\, j \in \mathscr{C},\ v \in I_j && (3) \\
& x_{vj} \in \{0,1\} && \forall\, v \in V,\ j \in L(v) \\
& y_j \in \{0,1\} && \forall\, j \in \mathscr{C}
\end{aligned}
$$

Constraints (1) ensure that each vertex is assigned a color, (2) impose that no pair of adjacent vertices receive the same color (when that color is used), and (3) control the activation of y_j when v is an isolated vertex in G^j.

The second ILP model corresponds to a formulation of colorings as covering of vertices by stable sets, based on GCP-SC. Here, we have a binary variable x_S^j for each $j \in \mathscr{C}$ and each stable set $S \in \mathscr{S}(G^j)$. Observe that, for any color j, its color class is a stable set from G^j and, unlike GCP, no more than one maximal stable set of G^j can be used by the covering. The following ILP model arises naturally:

$$
\begin{aligned}
& \min \sum_{j \in \mathscr{C}} w_j \sum_{S \in \mathscr{S}(G^j)} x_S^j \\
& s.t. \\
& \sum_{j \in L(v)} \sum_{\substack{S \in \mathscr{S}(G^j) \\ v \in S}} x_S^j \ge 1 && \forall\, v \in V && (4) \\
& \sum_{S \in \mathscr{S}(G^j)} x_S^j \le 1 && \forall\, j \in \mathscr{C} && (5) \\
& x_S^j \in \{0,1\} && \forall\, j \in \mathscr{C},\ S \in \mathscr{S}(G^j)
\end{aligned}
$$

Constraints (4) say that, for each vertex v, at least one stable set containing v is selected, while (5) allow to take at most one stable set for each color.

However, this formulation does not properly generalize GCP-SC in the sense that an instance of GCP gives rise to a model with many more variables than GCP-SC. As we have said in the introduction, GCP-SC takes advantage of the indistinguishability between colors. In order to overcome this drawback, we say that two colors $j, k \in \mathscr{C}$ are indistinguishable if $w_j = w_k$ and $G^j = G^k$. Let $\mathscr{C}^k = \{j \in \mathscr{C} : j, k \text{ are indistinguishable}\}$ and $\{\mathscr{C}^k : k \in K\}$ be a partition of $\mathscr{C}$ where $K \subset \mathscr{C}$. In other words, each $k \in K$ is a color that *represents* those ones from $\mathscr{C}^k$.

Now, we can take advantage of the symmetry between those colors and allow the existence of at most $|\mathscr{C}^k|$ stable sets of G^k instead of a single one. Moreover, if $|G^k| \le |\mathscr{C}^k|$, such constraint can be removed since there is an optimal solution verifying this condition. Now, the MWLCP can be re-formulated:

$$
\begin{aligned}
&\text{(MWLCP-SC)} \quad \min \sum_{k\in K} w_k \sum_{S\in\mathscr{S}(G^k)} x_S^k \\
&s.t. \\
&\sum_{k\in K} \sum_{\substack{S\in\mathscr{S}(G^k)\\ v\in S}} x_S^k \ge 1 && \forall\, v \in V \qquad (6)\\
&\sum_{S\in\mathscr{S}(G^k)} x_S^k \le |\mathscr{C}^k| && \forall\, k \in K : |G^k| \ge |\mathscr{C}^k| + 1 \qquad (7)\\
&x_S^k \in \{0,1\} && \forall\, k \in K,\ S \in \mathscr{S}(G^k)
\end{aligned}
$$

Therefore, for instances corresponding to GCP, we have $K = \{1\}$ and $\mathscr{C}^1 = \mathscr{C}$. If, in addition, $|\mathscr{C}| \ge |V(G)|$, constraints (7) are no longer needed and our last formulation turns out to be the same as GCP-SC.

3 Solving LP relaxations of MWLCP-SC

Given an instance $(G, \mathscr{C}, w, L)$ of MWLCP, let $X \doteq \{(S,k) : S \in \mathscr{S}(G^k),\ k \in K\}$. The variables of the linear relaxation of MWLCP-SC are x_S^k for $(S,k) \in X$ and its constraints can be written as $Ax \ge 1$, $Bx \ge -b$ and $x \ge 0$ where, for every variable x_S^k, the corresponding column in A is the characteristic vector of S and the columns in B have a non-zero entry, -1, in the row corresponding to $k \in K$. Moreover, for each $k \in K$, $b_k = |\mathscr{C}^k|$. Note that constraints of the form $x_S^k \le 1$ are not necessary (non-negative weights ensure that at least one optimal solution verifies these conditions).

For any $\hat{X} \subset X$, let $x(\hat{X})$ be the vector of variables x_S^k restricted to $(S,k) \in \hat{X}$ and $LP(\hat{X})$ be the linear relaxation of MWLCP-SC restricted to $x(\hat{X})$. We first give some results concerning the integrality of solutions of the LP relaxation, which will be used in Section 4.

Lemma 3.1 *Let x^* be an optimal solution of $LP(X)$ and let $\tilde{X} = \{(S,k) \in X : |S| \ge 2\}$. If $x^*(\tilde{X})$ is integral then $LP(X)$ has an optimal integer solution.*

Proof. Consider the partition $\{X^0, X^+\}$ of X where $X^0 \doteq \{(S,k) : x_S^{*k} = 0\}$ and $X^+ \doteq X \setminus X^0$. Clearly, $x^*(X^+)$ is an optimal solution for $LP(X^+)$. Since $x^*(\tilde{X})$ is integral, variables from $\tilde{X} \cap X^+$ are set to 1. Then, by replacing occurrences of those variables by 1 in $LP(X^+)$, we obtain a linear program LP' over variables $X^+ \setminus \tilde{X}$, such that $x^*(X^+ \setminus \tilde{X})$ is an optimal solution for LP'. Observe that the stable sets represented by variables in $X^+ \setminus \tilde{X}$ are singletons.

Let A' and B' be the submatrices of A and B corresponding to constraints (6) and (7) in LP', respectively. Since columns of A' are characteristic vectors of singleton stable sets, they have one positive entry equal to 1. Similarly, columns of B' have at most one negative entry equal to -1, thus implying that the matrix of coefficients of LP' is totally unimodular. Therefore, there exist integer optimal solutions for LP' and also $LP(X)$. □

Note that when every G^j is complete, stable sets are singletons and $\tilde{X} = \emptyset$ in the statement of the previous lemma. Then, we obtain:

Corollary 3.2 *If, for all $j \in \mathscr{C}$, G^j is a complete graph, then $LP(X)$ has an integral optimal solution.*

Actually, Corollary 3.2 can be also derived from a reduction of MWLCP to the Minimum Weighted Perfect Matching Problem on Bipartite Graphs, described below.

Consider an instance $(G, \mathscr{C}, w, L)$ of MWLCP such that G^j is a complete graph for all $j \in \mathscr{C}$. Observe that each color can be assigned to at most one vertex. Then, we can assume that $|V(G)| \leq \mathscr{C}$ (otherwise, MWLCP would be infeasible). Let Z be a set of dummy elements such that $|Z| = |\mathscr{C}| - |V(G)|$.

Now, consider the bipartite graph $\tilde{G}$ with vertex partition defined by $V(G) \cup Z$ and $\mathscr{C}$, and such that for all $v \in V(G)$ and $j \in \mathscr{C}$, (v, j) is an edge of $\tilde{G}$ if and only if $j \in L(v)$. The weight of (v, j) is w_j. We also consider edges (z, j) for all $z \in Z$ and $j \in \mathscr{C}$ with weight zero. Clearly, if f is a list coloring of G, there exists a perfect matching of $\tilde{G}$ containing edges $(v, f(v))$ for all $v \in V(G)$ and edges $(z, j(z))$ for all $z \in Z$ and some $j(z)$ taken from the non-active colors in f. The weight of this perfect matching coincides with the weight of the list coloring. Conversely, for any perfect matching M of $\tilde{G}$, edges $(v, j) \in M$ for every $v \in V(G)$ define a list coloring of G with the same weight that M. Moreover, $\tilde{G}$ has a perfect matching if and only if MWLCP is feasible.

In practice, this problem can be addressed much faster via the Hungarian Algorithm than by solving the LP relaxation.

3.1 Column generation

Let $\hat{X} \subset X$, $x^*(\hat{X})$ be an optimal solution of $LP(\hat{X})$, and (π^*, γ^*) be the optimal dual solution of $LP(\hat{X})$, where π and γ are the dual variables corresponding to constraints (6) and (7), respectively (for those $k \in K$ such that $|G^k| \leq |\mathscr{C}^k|$, we assume $\gamma^*_k = 0$). Note that, for any $(S, k) \in X$, the reduced cost of x^k_S is $c^k_S \doteq \sum_{v \in S} \pi^*_v - (w_k + \gamma^*_k)$.

We recall that the x^* obtained from $x^*(\hat{X})$ by padding zeros, i.e. $x^*(X \setminus \hat{X}) = \mathbf{0}$, is feasible for $LP(X)$ but not necessarily optimal. In order to know that, we have to decide if there exists $(S, k) \in X$ such that $c^k_S > 0$ or, equivalently, to solve the following auxiliary problem:

$$\text{(Aux)} \quad \exists\ (S, k) \in X \textit{ such that } \textstyle\sum_{v \in S} \pi^*_v > w_k + \gamma^*_k?$$

Clearly, (Aux) is equivalent to solve, for each $k \in K$, the *Maximum Weighted Stable Set Problem* (MWSSP) on G^k with weight π^*_v for each $v \in V(G^k)$.

Despite the MWSSP being $\mathcal{NP}$-hard, the enumerative routine given in [9] shows to be reasonably fast when it is called by the branch-and-price algorithm. In addition, it is not always necessary to solve the MWSSP to optimality. One can stop the optimization as soon as a set $S \in \mathscr{S}(G^k)$ with weight greater than the threshold $T^k \doteq w_k + \gamma_k^*$ is found.

Moreover, if we have $G^k = G^\ell$ and $T^k > T^\ell$ for some $k \neq \ell \in K$, there is no need to run the enumerative routine for G^ℓ since the stable set found for G^k can be reused for G^ℓ.

Although finding one column is enough for the column generation process, preliminary experiments have shown that incorporating many columns at the same time decreases the number of iterations of the process and, thus, the overall time. In our implementation, we search for (if exists) one entering variable per $k \in K$.

Observe that the column generation process needs to start with a set $\hat{X} \subset X$ for which $LP(\hat{X})$ is feasible. We recall that, unlike GCP, to know if an instance of MWLCP is feasible is an $\mathcal{NP}$-complete problem and the best algorithm that finds a list coloring, whenever it exists, is $O(2^n)n^{O(1)}$ [13]. In the following section we propose an initialization procedure.

3.2 *Initializing the linear relaxations*

Our approach consists in extending the solution space by creating a dummy color per vertex. Given an instance $(G, \mathscr{C}, w, L)$ of MWLCP we generate an extended instance $(G, \mathscr{C}', w', L')$ where $\mathscr{C}' = \mathscr{C} \cup \{d_v : v \in V(G)\}$, $w'_j = w_j$ for all $j \in \mathscr{C}$ and $w'_{d_v} = M$ for all $v \in V(G)$, where M is a *big number* (e.g. $M = 1 + \sum_{j \in \mathscr{C}} w_j$). In addition, $L'(v) = L(v) \cup \{d_v\}$ for all $v \in V(G)$.

Now, the linear relaxation obtained from the instance $(G, \mathscr{C}', w', L')$ is trivially feasible since $x^{d_v}_{\{v\}} = 1$ for all $v \in V(G)$ is a feasible solution which allows us to start our column generation process. That is, $\hat{X} = \{(\{v\}, d_v) : v \in V(G)\}$.

It is expected that dummy colors will no longer be used as optimization proceeds. In fact, if some dummy color is still active in the optimal solution of the relaxation associated to the extended instance $(G, \mathscr{C}', w', L')$, then $(G, \mathscr{C}, w, L)$ is infeasible.

Preliminary experiments have revealed that starting with dummy colors is better than a feasible initial coloring, when the latter can be generated (there are heuristics algorithms that deliver list colorings, such as k-Greedy-List [14]). We speculate the cause of this may be that the initial columns yielded by such colorings are of poor quality, thus making the LP solver to perform more iterations.

Since our column generation process is embeeded in a Branch-and-Price framework, the subproblems we have to solve for each node of the B&B tree should correspond to new instances of the MWLCP. To achieve that purpose, we need to devise a *robust branching rule*. In the next section we present such a rule, adapting the one first proposed in [15] and used in [8] for the GCP.

4 A robust branching rule

The idea behind the rule given in [8] is to pick two non-adjacent vertices u and v and divide the space of solutions between those ones satisfying $f(u) = f(v)$ (both share the same color) and those others satisfying $f(u) \neq f(v)$. Finding the optimal coloring in the latter case is equivalent to solve GCP on a graph obtained by adding edge (u, v) to G. In the former case, one has to solve GCP on a graph obtained from G by collapsing vertices u and v into a single one (that is, connecting u to every vertex from $N_G(u) \cup N_G(v)$ and removing v, where $N_G(x)$ is the open neighborhood of x in G).

We follow the same strategy, with the additional condition that vertices u and v must satisfy $L(u) \cap L(v) \neq \emptyset$. Indeed, finding the optimal list coloring f of $(G, \mathscr{C}, w, L)$ with $f(u) \neq f(v)$ is equivalent to solve MWLCP for the instance $(G_1, \mathscr{C}, w, L)$ where $G_1 = (V(G), E(G) \cup \{u, v\})$, whereas finding the optimal f with $f(u) = f(v)$ is equivalent to solve MWLCP for the instance $(G_2, \mathscr{C}, w, L_2)$ where $G_2 = G \setminus \{v\}$ and u is connected to every vertex from $N_G(v)$, $L_2(z) = L(z)$ for all $z \in V(G_2) \setminus \{u\}$ and $L_2(u) = L(u) \cap L(v)$.

Note that, in a finite number of applications of the previous rule, we get instances where all graphs G^k are complete. At that point, the optimal solutions of their LP relaxations are integers by Corollary 3.2.

In addition, due to the intersection performed to obtain $L_2(u)$, it is likely to reach subproblems whose instances have a vertex with a singleton list. Since such vertex is forced to be colored with the unique available color for it, the instance can be preprocessed according to the following result:

Lemma 4.1 *Let $(G, \mathscr{C}, w, L)$ be an instance of the MWLCP such that $L(u) = \{j\}$ for some $u \in V(G)$ and $j \in \mathscr{C}$. Solving $(G, \mathscr{C}, w, L)$ is equivalent to solve MWLCP for an instance $(G', \mathscr{C}, w, L')$ where $G' = G \setminus \{u\}$, $L'(z) = L(z) \setminus \{j\}$ for all $z \in N(u)$ and $L'(z) = L(z)$ for all $z \in V(G') \setminus N(u)$.*

Proof. If f' is an optimal list coloring of $(G', \mathscr{C}, w, L')$ then $f(u) = j$, $f(z) = f'(z)$ for all $z \in V(G')$ is an optimal list coloring of $(G, \mathscr{C}, w, L)$. Instead, if $(G', \mathscr{C}, w, L')$ is infeasible, $(G, \mathscr{C}, w, L)$ is also infeasible. □

It only remains to specify a criterion for selecting vertices u and v, which we address below.

4.1 Branching variable selection strategy

In [8], a simple but efficient rule is proposed. We explain it briefly in terms of our formulation for the case $K = \{1\}$ (superscripts are omitted). Assume that x^* is a non-integer optimal solution of the LP relaxation. The authors first search for the *most fractional variable*, i.e. an $x^*_{S_1}$ that minimizes $|x^*_{S_1} - \frac{1}{2}|$, and pick $u \in S_1$. Since $x^*_{S_1} \notin \{0, 1\}$ and (6) holds for u, there exists another S_2 such that $x^*_{S_2} > 0$ and $u \in S_2$. Then, they search for the first vertex v such that $v \in S_1 \Delta S_2$ and, due to the fact that stable sets are different from each other, such v exists and it is not adjacent to u.

In our case, we search for the most fractional $x^{*k}_{S_1}$ satisfying $|S_1| \geq 2$ and we pick $u \in S_1$. Lemma 3.1 guarantees the existence of such stable set (otherwise, the solution would be integral). Now we need a pair (S_2, ℓ) such that $S_1 \neq S_2$, $u \in S_2$, and $x^{*\ell}_{S_2} > 0$. If such (S_2, ℓ) exists, we pick $v \in S_2 \setminus S_1$. Otherwise, we pick $v \in S_1 \setminus \{u\}$. In this way, we can assert that u and v are non-adjacent and satisfy the pre-condition $L(u) \cap L(v) \neq \emptyset$ since they must have colors k or ℓ in their lists.

5 Computational results

This section is devoted to analyze the performance of the proposed Branch-and-Price algorithm over randomly generated instances, with number of vertices $n \in \{50, 60, 70\}$ and edge probability $p \in \{0.25, 0.50, 0.75\}$.

Cardinality of $\mathscr{C}$ and distribution of colors in the lists are ruled by two parameters $c \in \{0.5, 1.0, 1.5\}$ and $q \in \{0.25, 0.50, 0.75\}$. The former is used to set $\mathscr{C} = \{1, \ldots, \lfloor cn \rfloor\}$. The latter, which we refer to as *membership-to-list probability*, is the probability that a color $j \in \mathscr{C}$ belongs to $L(v)$, for each v. Weights are set to 1 for all colors. Random numbers are yielded by an uniform distribution.

The experiment consists of comparing two approaches, which we call CPLEX-CF and BP-SC. In the former, the ILP solver of CPLEX (with default parameters) solve the MWLCP-CF formulation. The latter is our Branch-and-Price algorithm, which was manually implemented in `C++` and uses CPLEX only for the resolution of LP relaxations. In order to speed up the generation of subproblems, we copy and perform minimum changes to the data structure where the instance lies. This saves time since it is not necessary to re-compute the partition of the set of colors nor the subgraphs G^k among them from scratch. Lemma 4.1 is applied during this stage. Then, subproblems are solved in a depth-first search fashion. Regarding the pricing routine, we traverse graphs G^k according to the value T^k in decreasing order, in order to avoid solving the MWSSP over the same graph more than once. On the other hand, our current implementation does not use the Hungarian Algorithm for the relaxations related to Corollary 3.2. This feature will be incorporated in a future version.

The experiment is carried out by a desktop computer equipped with an AMD Phenom II X4 3.4 GHz (a single thread is used), 3.6 GB of memory, Ubuntu 16.04 operating system, GCC 5.4.0 and IBM ILOG CPLEX 12.7 as the integer and linear programming solver. A time limit of 1 hour is imposed on solving each instance.

Results are summarized in Table 1. In each row, averages over 5 instances generated with the same combination of values (n, p, c, q) are reported: averages of CPU time (in seconds) for CPLEX-CF and BP-SC with best times highlighted, and average of number of nodes explored by BP-SC. Only instances solved to optimality within the time limit are considered in the averages. For those cases when at least one of the 5 instances is not solved, the number of solved ones is reported in brackets. If none of them is solved, a symbol "–" is displayed.

As we can see from the table, BP-SC achieves a remarkable performance, being able to solve to optimality 98% of the instances (397 of 405). In particular, all

			$q = 0.25$			$q = 0.50$			$q = 0.75$		
			SC	SC	CF	SC	SC	CF	SC	SC	CF
n	p	c	nodes	time	time	nodes	time	time	nodes	time	time
50	0.25	0.5	47	0.3	0.3	233	8.5	22.0	55	2.9	1235.9
		1.0	475	5.4	1.2	792	33.6	660.3	53	4.2	–
		1.5	88	1.3	4.2	87	3.3	454.8	56	5.2	–
	0.50	0.5	46	0.4	0.6	159	4.2	400.9	50	2.1	–
		1.0	57	0.7	2.5	238	8.1	507.8	45	2.9	–
		1.5	211	3.6	6.2	63	2.5	623.4	37	3.5	–
	0.75	0.5	39	0.3	0.5	26	0.5	27.0	16	0.6	–
		1.0	22	0.2	0.4	23	0.6	53.1	13	0.8	–
		1.5	30	0.4	0.8	19	0.7	64.1	15	1.3	–
60	0.25	0.5	107	1.4	0.8	230	8.6	106.0	895	143.7	–
		1.0	627	14.1	8.0	95	7.1	789.9	184	28.6	–
		1.5	261	8.7	30.1	631	102.3	-	726	177.6	–
	0.50	0.5	43	0.7	1.6	132	4.9	596.5(2)	487	49.2	–
		1.0	249	5.5	25.9	71	4.1	-	457	68.6	–
		1.5	113	3.4	34.3	121	10.9	-	408	86.1	–
	0.75	0.5	51	0.6	0.9	33	1.3	1686.6(3)	35	2.3	–
		1.0	43	0.8	3.1	34	1.8	-	40	3.9	–
		1.5	32	0.8	2.9	45	2.5	-	33	4.7	–
70	0.25	0.5	8467	223.6	4.6	388	55.8	1154.8(3)	71	19.1	–
		1.0	2669	68.5	88.7	3167	590.0(2)	–	95	32.6	–
		1.5	4814	281.7(3)	894.0	1853	230.5(3)	–	101	43.0	–
	0.50	0.5	523	14.7	82.3	2333	188.5	–	166	34.1	–
		1.0	1506	59.3	632.6(4)	200	14.9	–	153	50.4	–
		1.5	661	26.1	1703.6	1506	11.0	–	161	71.3	–
	0.75	0.5	59	1.2	4.3	68	3.2	–	45	5.0	–
		1.0	64	1.7	30.7	40	3.3	–	45	8.3	–
		1.5	77	2.6	94.5	52	4.9	–	46	11.8	–

Table 1
Results on random graphs (average of 5 instances per row).

instances with 50 and 60 vertices can be solved in a couple of minutes. The same behavior is observed when increasing the number of vertices to 70 and considering an edge probability of 0.5 or 0.75 (medium to high density). In contrast, instances with edge probabilities of 0.25 (low density) are occasionally harder to solve, except when membership-to-list probability is high.

Note that the number of explored nodes is low, pointing out that the linear relaxations provide tight bounds and the node selection strategy is good enough, but also the resolution of each relaxation is very time consuming.

Regarding CPLEX-CF, we notice that it solves half of the instances within the time limit (202 of 405). One of the disadvantages of this formulation is the weak lower bound provided by the value of the LP relaxation. In fact, LP relaxations of BP-SC are 174% larger (in average) than those from CPLEX-CF, and this gap is accentuated as the size of the instance grows in any parameter (n, p, c or q). The peak is reached on instances with $n = 70$, $p = 0.75$, $c = 1.5$ and $q = 0.75$ where LP relaxations of BP-SC are 604% larger.

Although CPLEX-CF outperforms BP-SC in 5% of the cases, these ones usually have low density graphs and lists with a small amount of colors, making MWLCP-CF smaller in number of variables and constraints. In the case of BP-SC, the lower the density of the graph, the harder the resolution of the MWSSP on graphs G^k (whose tend to be sparse).

Last but not least, both approaches evidence perturbations in the performance not only when varying number of vertices and density but also when changing the structure of the lists of colors. In the case of BP-SC, this behavior seems to be unpredictable. Even when we fix the number of vertices, edge probability and membership-to-list probability, the performance does not always get worse as the number of colors increases.

6 Conclusions

In this work, we present a Branch-and-Price algorithm to solve the weighted version of the List Coloring Problem, which has many applications and generalizes several other coloring problems. In order to achieve that, we propose a column generation process and a robust branching scheme based on the ones given in [8] for the GCP. In particular, when restricting to instances coming from GCP, our approach behaves just like LPCOLOR. However, other aspects should have been taken into account, such as the feasibility and the lack of indistinguishability between the colors. As far as we are concerned, this is the first exact algorithm based on ILP for this problem.

The computational experiments show that our algorithm has a remarkable performance, being able to solve to optimality randomly generated instances up to 70 vertices and within a time limit of 1 hour. Our results also expose that the hardness of the instances does not seem to depend only on the generation parameters (n, p, c and q), but also on the distribution of colors present in the lists. In this respect, we believe that further research still remains to be done. Particularly, it should be studied how the structure of graphs G^k, i.e. size, density and overlap (number of vertices they share each other), influence the performance of the Branch-and-Price algorithm. A better understanding of tough instances can help to detect improvement opportunities in our algorithm.

In future works, we plan to evaluate the performance of our algorithm over other types of instances. For example, by adding different weights to colors, varying the number of indistinguishable colors, using instances coming from applications (see e.g. [4]) and other coloring problems (e.g. Precoloring Extension, (γ, μ)-coloring) or by creating instances from benchmark graphs such as the ones from COLORLIB (DIMACS).

References

[1] Kubale, M., *Graph Colorings*, American Mathematical Society (2004).

[2] Bonomo, F., G. Durán, J. Marenco, *Exploring the complexity boundary between coloring and list-coloring*, Ann. Oper. Res. **169** (2009), 3–16.

[3] Wang, W., and X. Liu, *List-coloring based channel allocation for open-spectrum wireless networks*, IEEE 62nd Vehicular Technology Conference, Dallas, TX, USA (2005), 690–694.

[4] Lucci, M., G. Nasini, and D. Severín, *Planning the Workday of Bus Drivers by a Graph List-Coloring Model*, Electronic Journal SADIO **17** (2018), 77–91.

[5] Brélaz, D., *New Methods to Color the Vertices of a Graph*, Commun. ACM **22** (1979), 251–256.

[6] San Segundo, P., *A new DSATUR-based algorithm for exact vertex coloring*, Comp. Oper. Res. **39** (2012), 1724–1733.

[7] Méndez-Díaz, I., and P. Zabala, *A cutting plane algorithm for graph coloring*, Discr. Appl. Math. **156** (2008), 159–179.

[8] Mehrotra, A., and M. Trick, *A Column Generation Approach for Graph Coloring*, INFORMS J. Comput. **8** (1996), 331–437.

[9] Held, S., E. Sewell, and W. Cook, *Safe Lower Bounds For Graph Coloring*, Lect. Notes Comput. Sc. **6655** (2011), 261–273.

[10] Malaguti, E., M. Monaci, and P. Toth, *An exact approach for the Vertex Coloring Problem*, Discr. Optim. **8** (2011), 174–190.

[11] Golovach, P., M. Johnson, D. Paulusma, and J. Song, *A Survey on the Computational Complexity of Coloring Graphs with Forbidden Subgraphs*, J. Graph Theory **84** (2017), 331–363.

[12] Cygan, M., F. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer International Publishing (2015).

[13] Bjorklund, A., T. Husfeldt, and Mikko Koivisto, *Set partitioning via inclusion-exclusion*, SIAM Journal Comput. **39** (2009), 546–563.

[14] Achlioptas, D., and M. Molloy, *The analysis of a list-coloring algorithm on a random graph*, 38th Annual Symposium on Foundations of Computer Science, Miami, FL, USA (1997), 204–212.

[15] Zykov, A. A., *On some properties of linear complexes*, Matematicheskij Sbornik **24** (1949), 163–188.