A Column Generation Approach for Graph Coloring Using Maximal Cliques

Felipe Bartelt

***Abstract*— This work investigates the use of column generation to compute lower bounds for the chromatic number of undirected graphs. By modeling color classes as maximal cliques in the complement graph, we reformulate the graph coloring problem and solve its linear relaxation iteratively. New columns are introduced through a pricing subproblem based on dual values from the restricted master problem. The proposed method is tested on five benchmark instances from the OR-Library, yielding tight lower bounds in most cases.**

***Index Terms*— graph coloring, column generation, chromatic number, linear relaxation, complement graph, combinatorial optimization.**

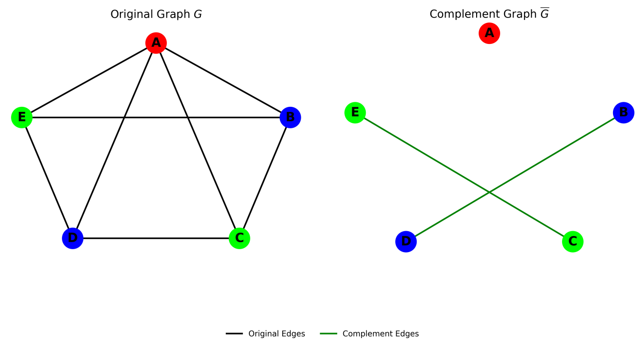

Fig. 1: Graph coloring example

I. Introduction

GRAPH coloring, a central problem in combinatorial optimization, seeks to assign colors to the vertices of a graph such that no two adjacent vertices share the same color. This constraint captures pairwise conflicts inherent in diverse applications, including scheduling of conflicting tasks [1], frequency assignment in wireless networks [2], compiler register allocation [3], and puzzle resolution (such as Sudoku; [4]). The chromatic number—the minimum number of colors required—is NP-hard to compute [5], motivating the development of bounding techniques and relaxations for practical computation.

In this work, we employ the column generation formulation for graph coloring introduced in [6], but adopt a different representation using the complement graph $\overline{G}$. Specifically, we model color classes as maximal cliques in $\overline{G}$ rather than independent sets in G—an equivalent formulation that we find to be more conducive to visualizing conflict relationships. Although this reformulation yields an equivalent integer programming model, the clique-based interpretation in $\overline{G}$ offers topological clarity that proves useful in our analysis. Our goal is not to solve the integer program to optimality, but rather to compute *lower bounds* on the chromatic number by solving the linear programming relaxation of the column generation model.

II. Problem Formulation

Let $G = (V, E)$ be an undirected graph, where V denotes the set of vertices with $|V| = n$, and E the set of edges with $|E| = m$. A *coloring* of G is an assignment of labels (colors) to the vertices such that no two adjacent vertices share the same color. A *minimum coloring* is one that uses the fewest number of colors, and the smallest such number is known as the *chromatic number*, denoted $\chi(G)$.

There are various ways to formulate the graph coloring problem. In this work, we rely on a formulation based on cliques of the *complement graph*. The complement of a graph $G = (V, E)$, denoted $\overline{G} = (V, \overline{E})$, is a graph with the same vertex set as G, but where edges are defined as

$$\overline{E} = \{(u, v) \in V \times V \mid u \neq v,\ (u, v) \notin E\},$$

so that $\overline{G}$ contains exactly the edges that are absent in G.

A *clique* in a graph $G = (V, E)$ is a subset of vertices $C \subseteq V$ such that every pair of distinct vertices in C is adjacent. A *maximal clique* is a clique that is not properly contained in any larger clique; that is, adding any vertex to it would violate the clique property.

As illustrated in Fig. 1, identifying color classes as independent sets in G (as done in [6]) is equivalent to representing them as maximal cliques in $\overline{G}$. We adopt the latter perspective, as it yields a more natural and intuitive interpretation of the problem structure for the purposes of our formulation and analysis.

Let $\mathcal{C}$ denote the set of all maximal cliques in the complement graph $\overline{G}$. For each clique $C \in \mathcal{C}$, let $\lambda_C \in \mathbb{B}$ be a binary variable indicating whether a color class is assigned to the vertices in C. The problem of determining the chromatic number $\chi(G)$ of a graph G can then be formulated as the following integer program, referred to as the integer master problem (IMP):

$$\begin{aligned} \min \sum_{C \in \mathcal{C}} \lambda_C & \\ \sum_{\{C \in \mathcal{C} \mid v \in C\}} \lambda_C \geq 1 \ \forall v \in V, & \qquad (1) \\ \lambda_C \in \mathbb{B} \ \forall C \in \mathcal{C}. & \end{aligned}$$

Instead of solving the integer problem directly, we consider its linear relaxation, which we refer to as the master problem (MP):

$$\begin{aligned} \min \sum_{C\in\mathcal{C}} \lambda_C & \\ \sum_{\{C\in\mathcal{C} \mid v\in C\}} \lambda_C \geq 1 \ \forall v \in V, & \\ 0 \leq \lambda_C \leq 1 \ \forall C \in \mathcal{C}. & \end{aligned} \tag{2}$$

It is known that the number of maximal cliques in a graph with $n \geq 2$ vertices can be as large as $3^{\frac{n}{3}}$ [7]. This exponential bound makes solving the MP directly intractable for even moderately sized graphs. However, the problem can be approached using a *column generation* method [8], which operates on a much smaller subset of variables. Let $\widehat{\mathcal{C}} \subset \mathcal{C}$ be such that $|\widehat{\mathcal{C}}| \ll |\mathcal{C}|$. The resulting restricted master problem (RMP) is given by:

$$\begin{aligned} \min \sum_{C\in\widehat{\mathcal{C}}} \lambda_C & \\ \sum_{\{C\in\widehat{\mathcal{C}} \mid v\in C\}} \lambda_C \geq 1 \ \forall v \in V, & \\ 0 \leq \lambda_C \leq 1 \ \forall C \in \widehat{\mathcal{C}}. & \end{aligned} \tag{3}$$

To determine whether the current RMP can be improved by introducing a new column, we solve a separation subproblem (SP), commonly known as the *pricing problem*:

$$\begin{aligned} \max \sum_{v\in V} \pi_v y_v & \\ y_v + y_w \leq 1 \ \forall (v,w) \in E, & \\ y_v \in \mathbb{B} \ \forall v \in V, & \end{aligned} \tag{4}$$

where the dual variables π_v correspond to the vertex constraints in the RMP, and are obtained after solving it to optimality. The binary variables y_v indicate whether vertex v is included in the proposed new clique. The constraint $y_v + y_w \leq 1$ for all $(v, w) \in E$ ensures that the selected set of vertices forms a clique in the complement graph.

If the objective value of this pricing problem exceeds 1, the corresponding set $\{v \in V \mid y_v = 1\}$ defines a clique that can be added to $\widehat{\mathcal{C}}$ to improve the RMP solution [6]. If the objective value is less than or equal to 1, then no improving column exists, and the current solution of the RMP coincides with the optimal solution of the full MP. Furthermore, if the variables λ_C are integer-valued in the final solution, then the result is also optimal for the original IMP; otherwise, enforcing integrality becomes necessary to obtain a valid coloring.

III. Implementation

The problem formulation requires an initial set of maximal cliques. To this end, we employ a greedy algorithm, presented in Algorithm 1, which ensures that each vertex in the graph is included in at least one clique.

To generate new columns—that is, new cliques—at each iteration, we use the solution obtained from the pricing problem (4). Specifically, the set $Y = \{v \in V \mid y_v = 1\}$ identifies a clique in the complement graph $\overline{G}$, which is then added to the RMP.

Algorithm 1 Greedy Maximal Clique Cover in $\overline{G}$

Input: Complement graph $\overline{G} = (V, \overline{E})$
Output: A set of maximal cliques $\widehat{\mathcal{C}}$ such that every $v \in V$ appears in at least one clique

$\widehat{\mathcal{C}} \leftarrow \varnothing$ ▷ Set of maximal cliques
$W \leftarrow V$ ▷ Vertices yet to be covered
while $W \neq \varnothing$ **do**
  Choose an arbitrary $v \in W$
  $C \leftarrow \{v\}$
  $N \leftarrow \{u \in W \mid (v,u) \in \overline{E}\}$
  while $N \neq \varnothing$ **do**
    for all $u \in N$ **do**
      $s(u) \leftarrow \big|\{w \in N \mid (u,w) \in \overline{E}\}\big|$
    end for
    $u^\star \leftarrow \arg\max_{u\in N} s(u)$
    $C \leftarrow C \cup \{u^\star\}$
    $N \leftarrow N \cap \{w \in W \mid (u^\star, w) \in \overline{E}\}$
  end while
  $\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} \cup \{C\}$
  $W \leftarrow W \setminus C$
end while
return $\widehat{\mathcal{C}}$

Only one column is generated per iteration. As a result, the current subset of maximal cliques is updated iteratively as

$$\widehat{\mathcal{C}}_{k+1} = \widehat{\mathcal{C}}_k \cup \{Y\},$$

where k denotes the current iteration index.

IV. Results

We evaluate the proposed method on five graph coloring instances from the OR Library[1]: `gcol1.txt` through `gcol5.txt`. These instances, detailed in Table 6 of [9], each consist of 100 vertices, while differing in edge counts: 2487, 2487, 2482, 2503, and 2450, respectively.

All experiments were executed on a machine running ArchLinux with an Intel Core i5-10300H CPU (4.50 GHz, single-threaded mode). The environment was configured with `Python 3.10.18`, `CPLEX 22.1.1.0`, and the `networkx` library[2] for graph operations such as computing the complement graph and neighbor sets. CPLEX was run with its default configuration, which may include multithreading.

Results are summarized in Table I, where $\widehat{\chi}(G)$ denotes the chromatic number estimate obtained via linear relaxation. For the first three instances, the algorithm consistently yields lower bounds near the known chromatic number of 16. In the fourth instance, the bound remains close but taking its floor slightly underestimates the chromatic number. In the fifth instance, the bound exceeds the true chromatic number, resulting in a rounded value of 17, which is incorrect. We conjecture that this overestimation arises from the influence of the initial clique

[1] https://people.brunel.ac.uk/~mastjjb/jeb/orlib/colourinfo.html
[2] https://networkx.org

TABLE I: Performance of the column generation algorithm for each test instance.

Instance	Initial Columns	Iterations	RMP Time[s]	SP Time [s]	Total Time [s]	$\widehat{\chi}(G)$	$\chi(G)$ [9]
`gcol1`	38	46	0.0207	3.7979	3.8826	15.9720	16
`gcol2`	39	44	0.0187	3.7893	3.8624	15.8518	16
`gcol3`	44	48	0.0218	4.5616	4.6434	15.9382	16
`gcol4`	34	64	0.0303	6.3346	6.4425	15.1906	16
`gcol5`	32	31	0.0121	1.4675	1.5244	16.5628	16

set $\widehat{\mathcal{C}}_0$, which may bias the solution toward suboptimal color class usage.

For all instances, the pricing subproblem (SP) terminated early with the message "102: An optimal solution within the tolerance defined by the relative or absolute MIP gap has been found." When this occurs, the column generation procedure is halted, and the current objective value is reported. None of the instances achieved convergence in the reduced cost test, i.e., the value $1 - \sum_{v \in V} \pi_v y_v$ remained below the threshold $\varepsilon = -10^{-6}$ throughout all iterations.

V. Conclusion

In this work, we proposed a column generation approach to estimate the chromatic number of undirected graphs using maximal cliques from the complement graph. By formulating the problem as a linear relaxation of an integer master problem, we aimed to obtain strong lower bounds without solving the full integer program.

We evaluated the method on five benchmark instances from the literature. For three of these, the algorithm produced tight lower bounds that closely matched the known chromatic number. In one instance, the bound was reasonably accurate, while in the final case, the bound exceeded the true chromatic number—highlighting a potential sensitivity of the method to the initial set of cliques.

This observation suggests several directions for future research. Improving the strategy for generating initial columns may enhance the reliability of the lower bounds. Furthermore, extending the method to allow multiple column insertions per iteration may lead to faster convergence and improved estimates.

References

[1] G. Laporte and S. Desroches, "Examination timetabling by computer," *Computers & Operations Research*, vol. 11, pp. 351–360, jan 1984.

[2] W. K. Hale, "Frequency Assignment: Theory and Applications," *Proceedings of the IEEE*, vol. 68, no. 12, pp. 1497–1514, 1980.

[3] G. J. Chaitin, M. A. Auslander, A. K. Chandra, J. Cocke, M. E. Hopkins, and P. W. Markstein, "Register allocation via coloring," *Computer Languages*, vol. 6, pp. 47–57, jan 1981.

[4] R. Vidyashree, B. T. Jeevika, and N. Guruprasad, "Graph Coloring Approach to Solve Sudoku Problems," *Proceedings of the 18th INDIAcom; 2024 11th International Conference on Computing for Sustainable Global Development, INDIACom 2024*, pp. 529–533, 2024.

[5] R. M. Karp, "Reducibility among combinatorial problems," in *50 Years of Integer Programming 1958-2008: from the Early Years to the State-of-the-Art*, pp. 219–241, Springer, 2009.

[6] A. Mehrotra and M. A. Trick, "A Column Generation Approach for Graph Coloring," *INFORMS Journal on Computing*, vol. 8, pp. 344–354, nov 1996.

[7] J. W. Moon and L. Moser, "On cliques in graphs," *Israel Journal of Mathematics*, vol. 3, pp. 23–28, mar 1965.

[8] J. Desrosiers, M. Lübbecke, G. Desaulniers, and J. B. Gauthier, "Branch-and-price," Les Cahiers du GERAD G-2024-36, Groupe d'études et de recherche en analyse des décisions, GERAD, Montréal QC H3T 2A7, Canada, Oct. 2024.

[9] C. Fleurent and J. A. Ferland, "Genetic and hybrid algorithms for graph coloring," *Annals of Operations Research*, vol. 63, no. 3, pp. 437–461, 1996.