

The Maximum Induced Happy Subgraph Problem: A Path-based Branch-and-Cut Approach

Rafael Schneider

Programa de Engenharia de Sistemas e Computação
Federal University of Rio de Janeiro
rschneider@cos.ufrj.br

Ana Flavia Uzeda Macambira

Departamento de Estatística
Federal University of Paraíba
anaflavia@de.ufpb.br

Luidi Simonetti

Programa de Engenharia de Sistemas e Computação
Federal University of Rio de Janeiro
luidi@cos.ufrj.br

Pedro Henrique González

Programa de Engenharia de Sistemas e Computação
Federal University of Rio de Janeiro
pegonzalez@cos.ufrj.br

ABSTRACT

This work addresses the Maximum Happy Induced Subgraph Problem (MHIS), a vertex-coloring variant that seeks to color connected components with a single color. In this problem, a vertex is considered happy if it is colored and all of its neighbors are also colored with the same color. We propose a new mathematical formulation based on paths between pre-colored vertices with distinct colors and develop a *Branch-and-Cut* algorithm that dynamically inserts cuts based on the detection of violated paths. Computational experiments show that the proposed approach is effective on small instances and significantly outperforms the original model on larger and more complex graphs, reinforcing the efficacy of the formulation and method developed in the context of combinatorial optimization.

KEYWORDS. Graph Coloring. Maximum Happy Induced Subgraph (MHIS). Branch-and-cut.

OD - Discrete Optimization, TAG - Graph Theory and Related Algorithms

1. Introduction

The Vertex Coloring Problem (VCP) is a classical challenge in Graph Theory, in which adjacent vertices must receive different colors, aiming to minimize the total number of colors used. Proper coloring is used to represent situations in which different entities, such as tasks or people, have conflicts among themselves and need to be allocated to a limited number of resources without overlap, with real-world applications in various areas [Ahmed, 2012], including school timetable scheduling [Santos and Souza, 2007], sports game scheduling [Miyashiro and Matsui, 2005], radio frequency assignment [Wang et al., 2009], and even seating planning in cinemas and theaters [Kheiri et al., 2021].

Research on variants of the classical coloring problem has become increasingly relevant as modern networks and systems grow more complex [Jensen and Toft, 2011]. While the traditional version aims to separate adjacent vertices by assigning distinct colors, many real applications require the opposite: grouping vertices that share common features, relationships, or interests, as in community detection in social networks [Li and Peng, 2011] or clustering techniques that group objects based on conceptual similarity [Everitt et al., 2011]. In these scenarios, vertex coloring can help identify homogeneous groups, following the homophily principle, which states that nearby vertices in a network tend to share similar attributes [Li and Peng, 2011].

With growing interest in coloring variants, the concept of a vertex's "happiness" emerged, leading to the Maximum Happy Vertices problem (MaxHV) [Zhang and Li, 2015]. In MaxHV, a vertex is happy if it and all its neighbors share the same color, allowing one to measure the homogeneity of a coloring on the graph. Given an initial coloring on some vertices, the goal is to color the remaining vertices to maximize the number of happy vertices. This problem has been extensively studied, focusing on its parameterized complexity and the development of specialized algorithms for restricted graph classes such as trees and bounded-treewidth graphs [Aravind et al., 2016; Agrawal, 2018].

Building on this, the Maximum Happy Induced Subgraph problem (MHIS) [Lewis et al., 2021] focuses on coloring connected components formed by vertices of the same color, seeking to consistently extend an existing partial coloring. To enable this, MHIS adopts a more flexible definition of happiness compared to MaxHV. While in MaxHV a vertex is happy only if it and all its neighbors are colored the same, in MHIS a vertex is happy if it is colored and has no neighbors of a different color, allowing some neighbors to remain uncolored. This flexibility permits identifying cohesive groups without requiring a full coloring of the graph. MHIS is NP-hard due to its equivalence to the Minimum Node Multiway Cut problem (MNMC) [Lewis et al., 2021]. Upper and lower bounds have been established to understand the space of feasible solutions, and both exact and heuristic algorithms have been implemented to explore their behavior on various instances [Lewis et al., 2021].

In this work, we propose a new mathematical formulation for MHIS based on simple paths and present a specialized *Branch-and-Cut* algorithm [Wolsey, 2020] to improve computational performance, especially for larger and more challenging instances. Our objective is to overcome the limitations of existing methods and provide a robust and efficient solution for this challenging problem. The results of extensive computational experiments, discussed in later sections, validate the effectiveness of the proposed method.

This paper is organized as follows. Section 2 presents a formal definition of MHIS and a brief literature review. Section 3 describes our methodology and algorithmic techniques. Section 4 presents the computational experiments, while Section 5 discusses the conclusions and outlines directions for future work.

2. Definitions and Preliminaries

In this section, we introduce fundamental concepts and notations for the Maximum Happy Induced Subgraph problem (MHIS), which is the focus of this work. We begin with basic definitions on graphs and vertex coloring, including the notion of a partial coloring. We then define the happiness criterion for a vertex in a partial coloring, as proposed in recent literature [Zhang and Li, 2015; Lewis et al., 2021]. Finally, we discuss the structure of happy components and formally state the MHIS objective: to maximize the number of happy vertices given an initial partial coloring.

Let $G = (V, E)$ be a simple graph. For a vertex $u \in V$, its neighborhood is $\Gamma(u) = \{v \in V \mid (u, v) \in E\}$. A *simple path* P in G is a subgraph $P = (V_P, E_P)$, where $V_P = \{v_0, v_1, \dots, v_p\} \subseteq V$ is a set of distinct vertices and $E_P = \{(v_i, v_{i+1}) \in E \mid i = 0, \dots, p-1\}$ is the set of edges connecting them sequentially. An ℓ -coloring of G is a function $c : V \rightarrow C$, where $C = \{1, 2, \dots, \ell\}$ and $c(v)$ denotes the color of vertex v . When ℓ is not relevant, we simply call c a *coloring*. In this work, we consider $\ell > 1$. A *partial coloring* is a function $c : V' \rightarrow C$, with $V' \subsetneq V$, where only vertices in V' are colored and the rest remain uncolored (Figure 1).

The concept of vertex *happiness* was introduced in the Maximum Happy Vertices (MHV) problem [Zhang and Li, 2015]. Here, we focus on MHIS [Lewis et al., 2021], which adapts happiness to partially colored graphs. Given a partial coloring $c : V' \rightarrow C$, a vertex $v \in V'$ is *happy* if for every neighbor $u \in \Gamma(v)$ we have $u \notin V'$ or $c(u) = c(v)$. That is, v is happy if its neighbors either share its color or are uncolored. (Figure 2).

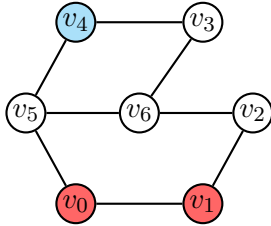


Figure 1: Partial 2-coloring of G .

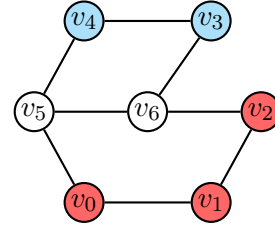


Figure 2: Happy vertex set $\{v_0, v_1, v_2, v_3, v_4\}$.

A *happy component* in G is any connected component $\mathcal{C} \subseteq G$ such that all colored vertices $u, v \in \mathcal{C} \cap V'$ satisfy $c(u) = c(v)$ (Figure 3). The MHIS problem consists in extending the partial coloring $c : V' \rightarrow C$ to a new partial coloring c' on a set $X \subsetneq V$ so that:

- (a) every vertex in the induced subgraph $G[X]$ is happy, and
- (b) $|X|$ is maximized.

The X set thus represents the vertices that will be colored in the final solution, ensuring that all of them belong to happy components. The goal of the problem is to select and color as many of these vertices as possible while ensuring that the resulting coloring does not produce any unhappy vertices. In other words, we aim to maximize the number of happy vertices while minimizing the number of uncolored ones (Figures 4 and 5). A natural constraint of the problem is that $V' \subseteq X$; that is, all previously colored vertices, defined by the partial coloring $c : V' \rightarrow C$, must necessarily be included in X . Equivalently, we choose the largest subset X of vertices to color, ensuring they form happy components and that $V' \subseteq X$.

The integer-programming model for MHIS [Lewis et al., 2021] uses binary variables x_{ik} , where $x_{ik} = 1$ if vertex i is assigned color k , and $x_{ik} = 0$ otherwise. In addition to the ℓ real colors,

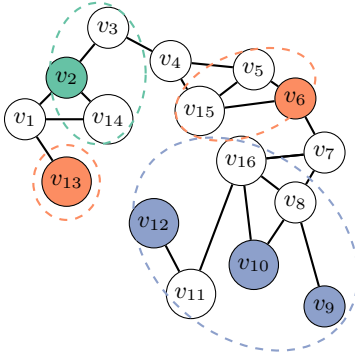


Figure 3: Extension of happy components induced by the partial coloring.

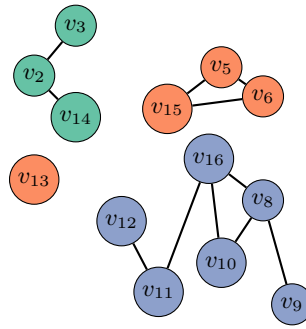


Figure 4: Example of a coloring extended to a valid set $X = V \setminus \{v_1, v_4, v_{19}\}$.

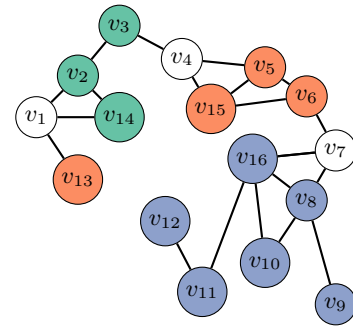


Figure 5: Feasible solution of the MHIS problem, including uncolored vertices.

a fictitious color $\ell + 1$ represents uncolored vertices. The objective is to maximize the number of vertices assigned to real colors:

$$\max \quad |V| - \sum_{i \in V} x_{i\ell+1} \quad (1)$$

$$\text{s.a.} \quad x_{ik} = 1, \quad \forall i \in V', c(i) = k, \quad (2)$$

$$\sum_{k=1}^{\ell+1} x_{ik} = 1, \quad \forall i \in V, \quad (3)$$

$$x_{ik} + x_{jl} \leq 1, \quad \forall (i, j) \in E, \forall k, l \in C, k \neq l, \quad (4)$$

$$x_{ik} \in \{0, 1\}, \quad \forall i \in V, \forall k \in C \cup \{\ell + 1\}. \quad (5)$$

The model constraints ensure the consistency of the color assignment. Constraint set (2) enforces that pre-colored vertices retain their assigned colors. Constraint set (3) guarantees that each vertex is assigned exactly one color, either real or fictitious. Constraint set (4) prevents conflicts between adjacent vertices by ensuring they do not simultaneously receive distinct real colors; this means that for two adjacent vertices to be happy, they must share the same real color. Finally, Constraint set (5) defines the binary nature of the decision variables. The objective function (1) seeks to maximize the number of vertices that are not assigned the fictitious color $\ell + 1$, which is equivalent to maximizing the number of vertices assigned real colors.

3. Methodology

The combinatorial complexity of the Maximum Happy Induced Subgraph (MHIS) problem demands efficient methods to reduce the search space and make the solution tractable, especially on larger instances. In this context, the *Branch-and-Bound* algorithm [Wolsey, 2020] stands out as the primary approach, as it allows an effective exploration of the solution space.

Our contribution's formulation, presented in Section 3.1, reformulates the problem in terms of paths between pre-colored vertices, dispensing with an explicit color representation in the decision variables. However, this approach leads to an exponential number of constraints. To handle this, the *Branch-and-Cut* algorithm emerges as a natural and efficient candidate, since it permits the dynamic incorporation of these constraints only when they are violated during the solution

process. In Section 3.2, we present a separation model whose purpose is to detect and add cuts as solutions are generated, thus ensuring the efficiency of the procedure.

3.1. Path-based Formulation

A MHIS solution consists of maximal connected components where all colored vertices share the same color and are happy. That is, if a vertex $i \in V'$ belongs to a connected component $\mathcal{C}$, then for every colored vertex $j \in \mathcal{C}$, $c(j) = c(i)$. Hence, no connected component may contain two pre-colored vertices $i, j \in V'$ with $c(i) \neq c(j)$. Moreover, any connected component that does not intersect V' may be assigned any color, since there is no coloring restriction on vertices outside V' .

In other words, along any path connecting vertices of different pre-colors, at least one of the internal vertices must remain uncolored in the optimal solution (see Figure 6).

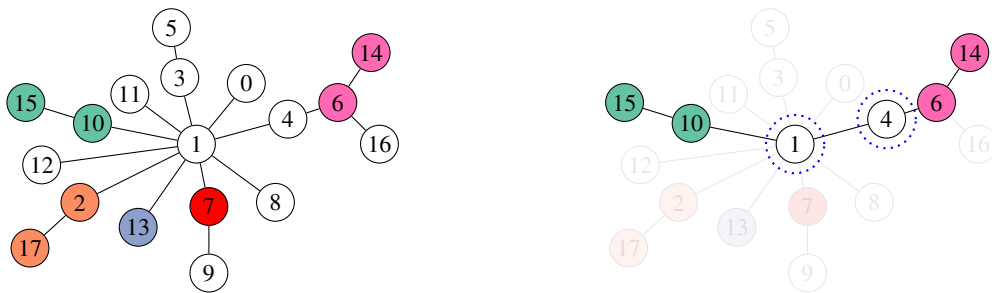


Figure 6: In any MHIS solution, at least one internal vertex (1 or 4) must remain uncolored, otherwise there would be a monochromatic path between differently pre-colored vertices.

Analogously to the original formulation (1)–(5), we leverage the equivalence between happy vertices and colored vertices in feasible solutions. We introduce binary variables h_i , where $h_i = 1$ indicates that vertex i is happy (i.e., colored in the solution).

The core idea is to prevent color conflicts in connected components of the happy subgraph: specifically, to forbid any path that connects two differently pre-colored vertices from being entirely composed of happy vertices. Let $\mathcal{P}$ be the set of all simple paths between $i, j \in V'$ with $c(i) \neq c(j)$. The MHIS can then be formulated as:

$$\max \quad \sum_{i \in V} h_i \quad (6)$$

$$\text{s.a.} \quad h_i = 1, \quad \forall i \in V' \quad (7)$$

$$\sum_{i \in V(p)} h_i \leq |V(p)| - 1, \quad \forall p \in \mathcal{P}, \quad (8)$$

$$h \in \mathbb{B}^{|V|}. \quad (9)$$

Constraint (7) forces all pre-colored vertices to remain happy, while (8) ensures that no entire path $p \in \mathcal{P}$ is chosen, thus forcing at least one internal vertex of each such path to be uncolored.

Although the formulation does not explicitly represent colors in the decision variables, the final coloring can be efficiently recovered from any feasible solution, since each connected component of happy vertices must be monochromatic. This structural property allows us to reconstruct a valid coloring post-optimization without additional complexity. Starting from the pre-colored vertices, we propagate their color hierarchically through their connected components, assigning the

same color to all adjacent happy vertices. Since the propagation respects the neighborhood structure and only traverses happy components, the coloring assignment can be completed in polynomial time using a simple graph traversal.

Because the Constraints set (8) is exponentially large, we adopt a *Branch-and-Cut* approach that generates these constraints dynamically via separation callbacks, as described next.

3.2. Separation Problem

Given a candidate solution $\bar{h} = (\bar{h}_1, \dots, \bar{h}_n)$, the separation problem identifies a violated path constraint. We formulate:

$$\min \sum_{i \in V} (1 - \bar{h}_i) y_i \quad (10)$$

$$\text{s.a.} \quad y \in \mathcal{P} \quad (11)$$

$$y \in \mathbb{B}^{|V|}. \quad (12)$$

Where y_i indicates whether vertex i is on the candidate path, and $\mathcal{P}$ is the set of all simple paths between differently pre-colored vertices. The objective (10) measures the number of “unhappy” vertices along the path, corresponding to $(|V(p)| - 1)$ minus the left-hand side of (8). If the optimal value of the separation problem is less than 1, the corresponding cut is violated.

Since standard algorithms operate on edge weights, we convert vertex happiness values into edge costs. Assign each edge $(i, j) \in E$ the weight

$$c_{ij} = \frac{2 - \bar{h}_i - \bar{h}_j}{2}.$$

To detect violations, we compute the shortest path distance between any two pre-colored vertices $v_1, v_2 \in V'$:

$$d(v_1, v_2) = \sum_{(i,j) \in P} c_{ij} + \frac{1 - \bar{h}_{v_1}}{2} + \frac{1 - \bar{h}_{v_2}}{2},$$

where P is the path found. Since $\bar{h}_{v_1} = \bar{h}_{v_2} = 1$, the endpoint terms vanish. If $d(v_1, v_2) < 1$, we generate the cut $\sum_{i \in V(P)} h_i \leq |V(P)| - 1$ via a callback using Dijkstra’s algorithm [Cormen et al., 2022].

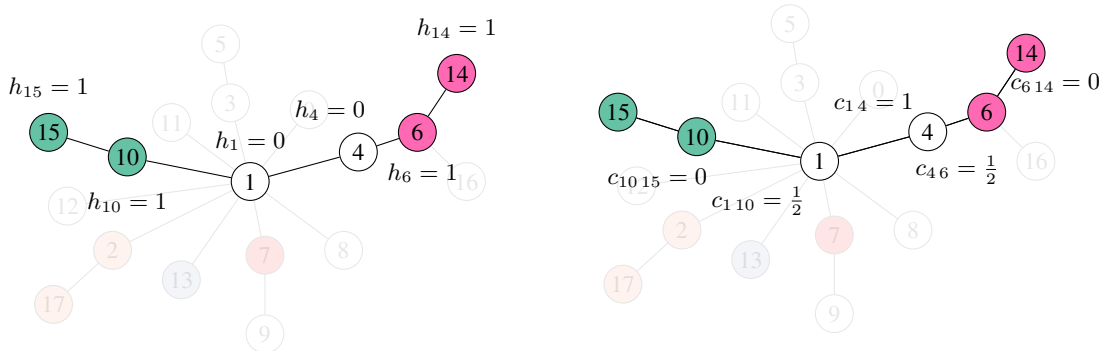


Figure 7: Transformation of weights into edge costs along the path where $d(v_{15}, v_{14}) = 2$.

4. Computational Experiments

The computational experiments were performed on a machine with an Intel® Core™ i7-5500U CPU @ 2.40 GHz, with 2 physical cores, 4 threads, and 4 MB cache. The system has 16 GB of RAM. The implementation was done in Python v3.10.14 and solved with IBM ILOG CPLEX® (version 22.11). The NetworkX library was used for graph handling.

The remainder of the section is organized as follows: Section 4.1 describes the generation of the instances used in the experiments, detailing the different network topologies. Section 4.2 presents and discusses the experimental results, offering a performance analysis and validating the effectiveness of the proposed methodology.

4.1. Instances

The instances used in this work were generated to reproduce the experimental universe adopted in the study that introduced the MHIS problem, ensuring a fair comparison between the evaluated methods. We directly employed the tools and instance-generation code made publicly available by the authors [Lewis et al., 2021]. Three graph topologies were considered: random graphs, d -regular graphs, and scale-free graphs. Random graphs were generated from a set of n isolated vertices, connecting each pair with a fixed probability p , yielding a binomial degree distribution. d -regular graphs were constructed via the Steger-Wormald random method [Steger and Wormald, 1999], ensuring every vertex has degree exactly d . For scale-free graphs, we used the Barabási-Albert model [Barabási, 2009], with parameter q determining the number of new edges added with each inserted vertex under preferential attachment.

After generating each graph, the pre-colored vertices were selected uniformly at random, guaranteeing that each of the k colors is used at least once and, whenever possible, that adjacent vertices do not share the same color. The instance sets cover different combinations of k , graph density, and proportion of pre-colored vertices.

4.2. Results and Discussion

We compare two exact methods. The first is Branch-and-Bound (B&B MHIS), applied to the original model (1)-(5). The second is Branch-and-Cut (B&C-Paths), based on our path-based formulation, using a separation problem for cut generation. Both methods were given a 900s time limit. Performance metrics are: `Sol_Val` (best solution value found), `GAP` (percentage gap between the best primal solution and the best dual bound), and `Time (s)` (total runtime in seconds). When optimality was not proven, the recorded time is 900s.

A total of 852 instances were evaluated, generated from combinations of the number of vertices n , pre-color count k , graph type, and pre-color proportion. Of these, 12 instances were not fully solved by at least one method within the time limit.

Results are organized into two parts: in Section 4.2.1, we discuss the unsolved instances; in Section 4.2.2, we present the solved instances. Table 1 shows the unsolved instances, where at least one method failed to produce an integer solution within 900s. Table 2 reports results for the instances solved by both methods.

4.2.1. Unsolved Instances

The unsolved instances comprise two graph types with distinct connectivity patterns: d -regular graphs (all vertices have degree d) and scale-free graphs (heterogeneous degree distribution). In Table 1, the “Parameter” column indicates d for regular graphs and q (edges per new vertex) for scale-free graphs.

Since at least one method did not find a primal solution, the standard GAP cannot be computed. We instead report the relative GAP based on the dual bound UB returned by the solver and the trivial solution z_{trivial} (pre-colored vertices only, see Constraints (2) and (7)):

$$\text{Relative GAP (\%)} = \frac{|UB - z_{\text{trivial}}| \times 100}{z_{\text{trivial}}}.$$

Values marked with an asterisk refer to the trivial solution. Relative GAPs marked with an asterisk are computed according to this alternative approach.

Table 1: Unsolved Instances: B&B MHIS e B&C-Paths

n	k	Instance Type	Parameter	B&B MHIS		B&C - Paths	
				Sol_Val	GAP	Sol_Val	GAP
500	3	d-regular	5	257	25.38%	100 *	211% *
500	4	d-regular	4	277	31.9%	100 *	240% *
500	10	d-regular	5	296	9.99%	100 *	223.1% *
500	15	d-regular	6	260	9.94%	100 *	184% *
500	20	d-regular	8	50 *	548,72% *	234	37.61%
500	30	d-regular	7	50 *	612,07% *	248	38.10%
500	40	d-regular	4	100 *	261,34% *	100 *	235,5% *
500	40	d-regular	7	100 *	149,45% *	201	19.40%
500	50	d-regular	8	50 *	531,42% *	225	37.89%
500	50	d-regular	8	100 *	111,86 *	191	8.64%
500	30	livre de escala	7	50 *	501,40% *	237	24.26%
500	40	livre de escala	3	50 *	768,28% *	376	8.51%

All instances presented in Table 1 have 500 vertices and could not be solved to optimality by any of the methods. For this reason, execution times are omitted from the table, as all runs reached the time limit of 900s.

Overall, the B&C-Paths method demonstrated greater robustness, obtaining higher-quality solutions in most cases. However, in the four instances with the smallest number of colors - $k = 3$, $k = 4$, $k = 10$, and $k = 15$, all on d -regular graphs, the B&B MHIS method was able to find feasible integer solutions, with GAPs ranging from 9.94% to 25.48%, while B&C-Paths failed to find any integer solution, resulting in relative GAPs above 200%. It is worth noting that, despite the small number of colors, these instances have 100 pre-colored vertices (indicated in the `Sol_Val` column as the trivial solution), which increases the complexity of the path structure between distinct pre-colored vertices.

On the other hand, in instances with higher values of k , the efficiency of the cutting-plane approach becomes more evident. The larger number of pre-colored vertices favors the generation of relevant violated paths, accelerating the algorithm's progress. In these cases, the B&C-Paths method showed a clear advantage, solving 7 instances for which the B&B MHIS was unable to find any feasible integer solution. A striking example is the instance on a scale-free graph with $k = 40$, where B&B MHIS remained at the trivial solution, resulting in a relative GAP of 768.28%. In contrast, B&C-Paths obtained a significantly better solution, with a GAP of only 8.51%.

Additionally, in the 4-regular instance with $n = 500$ and $k = 40$, where neither method was able to find a feasible integer solution, the B&C-Paths method stood out by achieving a smaller relative GAP of 235.5%. This was due to a better dual bound (UB), which was lower than that obtained by the B&B MHIS method, which resulted in a relative GAP of 261.34%.

These results demonstrate how the performance of the methods is strongly influenced by the instance structure and the value of k , highlighting the superiority of the path-based formulation in scenarios with a higher density of pre-colored vertices and more complex structures, where its ability to generate effective cuts provides a significant advantage.

4.2.2. Solved Instances

Results for the solved instances are grouped by (n, k) . The column # indicates the number of instances in each group. For each group, we report the average of each metric. `Opt (%)` is the

percentage of instances solved to proven optimality. Boldface marks the best method per metric; ties are left unhighlighted.

Table 2: Solved Instances: B&B MHIS e B&C-Paths

n	k	#	B&B MHIS				B&C-Paths			
			Sol_Val	Opt(%)	GAP	Time(s)	Sol_Val	Opt(%)	GAP	Time(s)
20	2	33	11.76	100%	0%	0.00	11.76	100%	0%	0.01
	3	27	9.63	100%	0%	0.00	9.63	100%	0%	0.01
	4	26	8.58	100%	0%	0.00	8.58	100%	0%	0.01
	5	15	8.27	100%	0%	0.00	8.27	100%	0%	0.01
	7	1	9.0	100%	0%	0.00	9.0	100%	0%	0.02
50	2	32	31.66	100%	0%	0.00	31.66	100%	0%	0.04
	3	39	26.36	100%	0%	0.02	26.36	100%	0%	0.07
	4	29	24.45	100%	0%	0.06	24.45	100%	0%	0.09
	5	27	22.7	100%	0%	0.07	22.7	100%	0%	0.09
	7	20	24.95	100%	0%	0.04	24.95	100%	0%	0.09
	10	17	22.65	100%	0%	0.13	22.65	100%	0%	0.08
100	2	38	59.45	100%	0%	0.01	59.45	100%	0%	0.31
	3	27	52.22	100%	0%	0.07	52.22	100%	0%	0.37
	4	33	51.12	100%	0%	0.20	51.12	100%	0%	0.70
	5	23	49.61	100%	0%	2.69	49.61	100%	0%	2.28
	7	26	48.92	100%	0%	31.87	48.92	100%	0%	7.23
	10	22	50.23	86.36%	2.46%	125.57	50.86	100%	0%	30.78
	20	17	47.35	100%	0%	5.20	47.35	100%	0%	1.05
200	2	39	133.59	100%	0%	0.01	133.59	100%	0%	1.71
	3	20	112.75	95%	0.32%	45.03	112.9	100%	0%	21.19
	4	16	124.0	93.75%	0.39%	59.08	124.06	100%	0%	20.91
	5	20	112.6	85%	1.69%	161.07	113.1	90%	0.83%	128.52
	7	18	115.33	94.44%	0.71%	63.05	115.39	94.44%	0.27%	56.37
	10	18	121.5	83.33%	4.72%	185.67	123.5	88.89%	1.40%	104.08
	15	18	115.5	77.78%	1.64%	205.92	116.17	100%	0%	46.51
	20	22	107.41	86.36%	5%	158.08	109.0	95.45%	1.36%	67.02
	30	9	118.67	77.78%	3.68%	205.43	119.11	77.78%	1.65%	203.32
500	2	42	311.88	100%	0%	0.04	311.88	100%	0%	27.87
	3	22	277.59	86.36%	0.18%	122.99	277.64	100%	0%	38.60
	4	23	252.52	73.91%	4.61%	238.93	253.78	73.91%	3.34%	254.22
	5	15	241.93	86.67%	5.76%	174.21	245.27	86.67%	3.54%	149.87
	7	23	286.39	60.87%	7.63%	388.79	291.7	73.91%	3.73%	278.12
	10	13	275.23	46.15%	13.96%	491.53	284.23	76.92%	5.99%	362.02

We observe that the methods perform equivalently on the smallest instances ($n = 20$ and $n = 50$), achieving optimality in all cases with runtimes below 0.13s. In these instances, B&B MHIS was slightly faster on average, but this difference is negligible and attributable to the simplicity of both the instances and the original formulation.

From instances with 100 vertices onward, the advantage of B&C-Paths becomes clear. For example, in the group $n = 100$, $k = 10$, B&B MHIS failed to solve 16.64% of the instances within 900s. Nonetheless, its average GAP remained low (2.46%), and its average solution value (50.23) was close to that of B&C-Paths (50.86). However, B&B MHIS required an average of 125.57s, whereas B&C-Paths solved them in only 30.78s on average, demonstrating a significant scalability improvement.

In the groups with $n = 200$ and $n = 500$, the superiority of B&C-Paths is even more pronounced. Except for $n = 200$, $k = 2$ (where B&B MHIS was faster), in all other 200-vertex combinations, B&C-Paths not only solved more instances to optimality but also did so in less time.

For instance, for $n = 200$, $k = 50$, B&B MHIS solved 75% of instances with an average of 231s, while B&C-Paths proved optimality in just 5.83s.

On the largest instances ($n = 500$), the limitations of B&B MHIS become evident in both optimality proof and runtime. Of the 11 groups of this size, B&B MHIS achieved 100% optimality only for $k = 2$. In contrast, B&C-Paths reached 100% for both $k = 2$ and $k = 3$, indicating its greater robustness.

Regarding the average GAP, B&B MHIS exceeded 5% in 6 of 11 groups and stayed below 1% only for $k = 2$ and $k = 3$. B&C-Paths exceeded 5% in only 3 groups and maintained GAP below 1% not only in the first two classes (where GAP was zero) but also in the last two ($k = 40$ and $k = 50$), further demonstrating the efficiency and scalability of the path-based formulation on more structured instances.

5. Conclusions and Future Work

We have proposed a new path-based integer formulation for the Maximum Happy Induced Subgraph problem (MHIS), which dispenses with explicit color variables by introducing happiness variables. Based on this model, we developed a *Branch-and-Cut* algorithm that dynamically incorporates valid cuts identified via an efficient separation problem.

Computational results highlight the high performance of the proposed method, especially on challenging instances. For small graphs, both methods yield similar solution quality and runtimes. However, as graph size and complexity increase, the superiority of the *Branch-and-Cut* approach becomes evident: it consistently finds high-quality integer solutions even when the original Branch-and-Bound model fails to produce any feasible solution within the time limit.

Moreover, the path-based formulation enabled solving instances that were intractable for the original model under a *Branch-and-Bound* scheme. The dynamic addition of path cuts was essential to guide the search more effectively, particularly on large and dense graphs. This led to a significantly more scalable and robust solver.

As future work, we plan to explore more efficient separation strategies to accelerate the detection of violated paths and to develop hybrid heuristics that combine exact cuts with local search or metaheuristics, aiming to generate strong initial solutions that further enhance the performance of our *Branch-and-Cut* algorithm.

Acknowledgements

This research was partially supported by the Federal University of Rio de Janeiro (ALV - 23079.227622/2023-70), the Brazilian National Council for Scientific and Technological Development (CNPq, grants 307663/2021-3 and 304680/2023-0), the Carlos Chagas Filho Foundation for Research Support of the State of Rio de Janeiro (FAPERJ, E-26/201.341/2022), and CAPES (Finance Code 001).

References

- Agrawal, A. (2018). On the parameterized complexity of happy vertex coloring. In Brankovic, L., Ryan, J., and Smyth, W. F., editors, *Combinatorial Algorithms*, p. 103–115, Cham. Springer International Publishing. ISBN 978-3-319-78825-8.
- Ahmed, S. (2012). Applications of graph coloring in modern computer science. *International Journal of Computer and Information Technology*, 3(2):1–7.
- Aravind, N. R., Kalyanasundaram, S., and Kare, A. S. (2016). Linear time algorithms for happy vertex coloring problems for trees. In Mäkinen, V., Puglisi, S. J., and Salmela, L., editors, *Combinatorial Algorithms*, p. 281–292, Cham. Springer International Publishing. ISBN 978-3-319-44543-4.

- Barabási, A.-L. (2009). Scale-free networks: a decade and beyond. *science*, 325(5939):412–413.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2022). *Introduction to algorithms*. MIT press.
- Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster analysis*.
- Jensen, T. R. and Toft, B. (2011). *Graph coloring problems*. John Wiley & Sons.
- Kheiri, A., Lewis, R., Thompson, J., and Harper, P. (2021). Constructing operating theatre schedules using partitioned graph colouring techniques. *Health Systems*, 10(4):286–297.
- Lewis, R., Thiruvady, D., and Morgan, K. (2021). The maximum happy induced subgraph problem: Bounds and algorithms. *Computers & Operations Research*, 126:105114. ISSN 0305-0548. URL <https://www.sciencedirect.com/science/article/pii/S0305054820302318>.
- Li, A. and Peng, P. (2011). The small-community phenomenon in networks. *Mathematical Structures in Computer Science*, 22.
- Miyashiro, R. and Matsui, T. (2005). A polynomial-time algorithm to find an equitable home-away assignment. *Operations Research Letters*, 33(3):235–241.
- Santos, H. G. and Souza, M. J. F. (2007). Programação de horários em instituições educacionais: formulações e algoritmos. *XXXIX SBPO-Simpósio Brasileiro de Pesquisa Operacional*, (1): 2827–2882.
- Steger, A. and Wormald, N. C. (1999). Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8(4):377–396.
- Wang, J., Huang, Y., and Jiang, H. (2009). Improved algorithm of spectrum allocation based on graph coloring model in cognitive radio. In *2009 WRI International Conference on Communications and Mobile Computing*, volume 3, p. 353–357. IEEE.
- Wolsey, L. A. (2020). *Integer programming*. John Wiley & Sons.
- Zhang, P. and Li, A. (2015). Algorithmic aspects of homophily of networks. *Theoretical Computer Science*, 593:117–131. ISSN 0304-3975. URL <https://www.sciencedirect.com/science/article/pii/S0304397515005058>.