

PGCOMP - Programa de Pós-Graduação em Ciência da Computação
Universidade Federal da Bahia (UFBA)
Av. Milton Santos, s/n - Ondina
Salvador, BA, Brasil, 40170-110

<https://pgcomp.ufba.br>
pgcomp@ufba.br

Given a graph G , its Grundy number $\Gamma(G)$ defines the worst-case behavior for the well-known and widely used first-fit greedy coloring heuristic. Specifically, $\Gamma(G)$ is the largest k for which a k -coloring can be obtained with the first-fit heuristic. The connected Grundy number $\Gamma_c(G)$ gives the worst-case behavior for the connected first-fit coloring heuristic, that is, one in which each vertex to be colored, except the first, is added adjacent to an already colored vertex. Both problems are NP-hard. In this master's thesis, we present heuristic and exact approaches to the Grundy coloring problem and the connected Grundy coloring problem, which are optimization problems consisting of obtaining the Grundy number and the connected Grundy number, respectively. This study proposes the use of a algorithm Biased Random-Key Genetic Algorithm (BRKGA) and the use of integer programming formulations using a more traditional (standard) approach and a representative one. A new combinatorial upper bound is also proposed that is valid for both problems and an algorithm using dynamic programming for its calculation. The computational experiments show that the new upper bound can improve over a well-established combinatorial bound available in the literature for several instances. The results also evidence that the formulation by representatives has an overall superior performance than the standard formulation, achieving better results for the denser instances, while the latter performs better for the sparser ones to the Grundy coloring problem. However, we show that these types of integer programming formulations are computationally impractical for the connected version. Furthermore, the BRKGA can find high-quality solutions for both problems and can be used with confidence in large instances where the formulations fail for the Grundy coloring problem.

Palavras-chave: combinatorial optimization; graph coloring; Grundy number; BRKGA; worst-case analysis.

Application of biased random-key genetic algorithm and formulations for the Grundy coloring problem and the connected Grundy coloring problem

Mateus Carvalho da Silva

Dissertação de Mestrado

Universidade Federal da Bahia

Programa de Pós-Graduação em
Ciência da Computação

Dezembro | 2023

UFBA



MSC | 172 | 2023

Application of biased random-key genetic algorithm and formulations for the Grundy coloring problem and the connected Grundy coloring problem

Mateus Carvalho da Silva



Universidade Federal da Bahia
Instituto de Computação

Programa de Pós-Graduação em Ciência da Computação

**APPLICATION OF BIASED RANDOM-KEY
GENETIC ALGORITHM AND FORMULATIONS
FOR THE GRUNDY COLORING PROBLEM AND
THE CONNECTED GRUNDY COLORING
PROBLEM**

Mateus Carvalho da Silva

DISSERTAÇÃO DE MESTRADO

Salvador
13 de dezembro de 2023

MATEUS CARVALHO DA SILVA

**APPLICATION OF BIASED RANDOM-KEY GENETIC ALGORITHM AND
FORMULATIONS FOR THE GRUNDY COLORING PROBLEM AND THE
CONNECTED GRUNDY COLORING PROBLEM**

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Rafael Augusto de Melo
Co-orientador: Márcio Costa Santos

Salvador
13 de dezembro de 2023

Sistema de Bibliotecas - UFBA

S586 Silva, Mateus Carvalho.

Application of biased random-key genetic algorithm and formulations for the Grundy coloring problem and the connected Grundy coloring problem / Mateus Carvalho da Silva – Salvador, 2023.

49p.: il.

Orientador: Prof. Dr. Rafael Augusto de Melo.

Co-orientador: Prof. Dr. Márcio Costa Santos.

Dissertação (Mestrado) – Universidade Federal da Bahia, Instituto de Computação, 2023.

1. Combinatorial optimization. 2. Graph coloring. 3. Grundy number. 4. BRKGA. 5, Worst-case analysis.. I. Melo, Rafael Augusto. II. Santos, Márcio Costa. III. Universidade Federal da Bahia. Instituto de Computação. IV. Título.

CDU – 510.5


MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DA BAHIA
INSTITUTO DE COMPUTAÇÃO
PGCOMP - Programa de Pós-Graduação em Ciência da Computação
<http://pgcomp.ufba.br>

“Application of biased random-key genetic algorithm and formulations for the Grundy coloring problem and the connected Grundy coloring problem.”

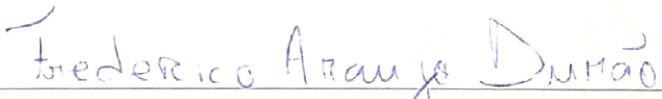
Mateus Carvalho da Silva

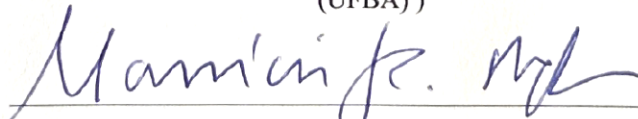
Dissertação apresentada ao
Colegiado do Programa de Pós-Graduação em Ciência
da Computação na Universidade Federal da Bahia,
como requisito parcial para obtenção do Título de
Mestre em Ciência da Computação.

Banca Examinadora


Prof. Dr. Rafael Augusto de Melo
(Orientador - PGCOMP)


Prof. Dr. Pedro Henrique González Silva
(UFRJ)


Prof. Dr. Frederico Araújo Durão
(UFBA)


Prof. Dr. Mauricio Guilherme de Carvalho Resende
(University of Washington)

ACKNOWLEDGEMENTS

First of all, I thank my family for their support, care, and investment in me.

I thank my adviser Rafael Augusto de Melo for all the support and orientation given in the construction and development of this work.

I would like to thank all the friends of the Computational Intelligence and Optimization Research Lab (CInO) whom I had the opportunity to meet and who, by participating in the meetings, enabled me to better understand the concepts presented in this work.

My thanks to Márcio C. Santos, Rodrigo F. Toso e Mauricio G. C. Resende whose suggestions and discussions helped in this process.

I am grateful for the financial support given by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES).

Finally, I thank all who contributed directly or indirectly to the accomplishment of this work.

RESUMO

Dado um grafo G , seu número de Grundy $\Gamma(G)$ define o comportamento de pior caso para a conhecida e amplamente utilizada heurística de coloração gulosa *first-fit*. Mais especificamente, $\Gamma(G)$ é o maior k para o qual uma k -coloração pode ser obtida com a heurística *first-fit*. O número de Grundy conexo $\Gamma_c(G)$ fornece o comportamento do pior caso para a heurística de coloração *first-fit* conexa, ou seja, aquela em que cada vértice a ser colorido, exceto o primeiro, é adicionado adjacente a um vértice já colorido. Ambos os problemas são NP-difíceis. Nesta dissertação, apresentamos abordagens heurísticas e exatas para o problema da coloração de Grundy e o problema de coloração de Grundy conexo, que são problemas de otimização consistindo na obtenção do número de Grundy e do número de Grundy conexo, respectivamente. Nesse estudo é proposto o uso do algoritmo genético de chaves aleatórias viesado (*Biased random-key genetic algorithm* - BRKGA) e do uso de formulações de programação inteira usando uma abordagem mais tradicional (padrão) e uma por representativos. Também é proposto um novo limite superior combinatório que é válido para ambos os problemas e um algoritmo usando programação dinâmica para o seu cálculo. Os experimentos computacionais mostram que o novo limite superior pode melhorar o limite para vários casos em relação a um limite combinatório bem estabelecido disponível na literatura. Os resultados também evidenciam que a formulação por representativos tem um desempenho geral superior que a formulação padrão, alcançando melhores resultados para as instâncias mais densas, enquanto esta última tem melhor desempenho para as mais esparsas para o problema dos números de Grundy. Contudo mostramos que este tipo de formulações com programação inteira são computacionalmente impraticáveis para a versão conexa. Além disso, o BRKGA pode encontrar soluções de alta qualidade para ambos os problemas e pode ser usado com confiança em grandes instâncias onde as formulações falham para o problema da coloração de Grundy.

Palavras-chave: otimização combinatória; coloração de grafos; número de Grundy; BRKGA; análise de pior caso.

ABSTRACT

Given a graph G , its Grundy number $\Gamma(G)$ defines the worst-case behavior for the well-known and widely used first-fit greedy coloring heuristic. Specifically, $\Gamma(G)$ is the largest k for which a k -coloring can be obtained with the first-fit heuristic. The connected Grundy number $\Gamma_c(G)$ gives the worst-case behavior for the connected first-fit coloring heuristic, that is, one in which each vertex to be colored, except the first, is added adjacent to an already colored vertex. Both problems are NP-hard. In this master's thesis, we present heuristic and exact approaches to the Grundy coloring problem and the connected Grundy coloring problem, which are optimization problems consisting of obtaining the Grundy number and the connected Grundy number, respectively. This study proposes the use of an algorithm Biased Random-Key Genetic Algorithm (BRKGA) and the use of integer programming formulations using a more traditional (standard) approach and a representative one. A new combinatorial upper bound is also proposed that is valid for both problems and an algorithm using dynamic programming for its calculation. The computational experiments show that the new upper bound can improve over a well-established combinatorial bound available in the literature for several instances. The results also evidence that the formulation by representatives has an overall superior performance than the standard formulation, achieving better results for the denser instances, while the latter performs better for the sparser ones to the Grundy coloring problem. However, we show that these types of integer programming formulations are computationally impractical for the connected version. Furthermore, the BRKGA can find high-quality solutions for both problems and can be used with confidence in large instances where the formulations fail for the Grundy coloring problem.

Keywords: combinatorial optimization; graph coloring; Grundy number; BRKGA; worst-case analysis.

LIST OF FIGURES

2.1	Coloring examples using $\sigma = (b, d, e, c, f, a)$	4
2.2	Difference in coloring when using a non-connected and connected order with the <i>first-fit</i> heuristic.	4
2.3	Example of possible solutions when coloring a graph using $\chi(G)$ -colors	7
2.4	Example of how works the evolution stage of the BRKGA between two generations	9
2.5	Crossover between two chromosomes in which there is a bias for the inheritance of <i>alleles</i> from the ELITE parent.	10
3.1	Binomial tree T_4 exemplifying the difference between $\Gamma(G)$ and $\chi(G)$ where numbers denote the color of each vertex, and which can be generated using the <i>first-fit</i> heuristic.	13
4.1	Encoded chromosome representing the solution to a graph with 6 vertices where the indices in the first line represent the vertices and the line below contains the key for each of them.	19
4.2	Difference in percent between the bounds $\zeta(G)$ and $\Psi(G)$ for the small instances	23
4.3	Difference in percent between the bounds $\zeta(G)$ and $\Psi(G)$ for the DIMACS instances	23
4.4	Barplot with the percentage of the instances for which the BRKGA at least matched the best solution obtained using any of the IP formulations, separated by $ V $	27
4.5	Barplot with the percentage of the instances for which the BRKGA at least matched the best solution obtained using any of the IP formulations, separated by the value of η	27
4.6	Boxplot summarizing the BRKGA deviations to the best solution considering all the executions	28
4.7	Boxplot summarizing the BRKGA deviations to the best-known solutions, separated by $ V $	29
4.8	Boxplot summarizing the BRKGA deviations to the best-known solutions, separated by η	29
5.1	Comparative boxplots of the deviation between BRKGA for Grundy coloring problem and connected Grundy coloring problem	37

LIST OF TABLES

4.1	Characteristics of the DIMACS instances	21
4.2	Comparison of upper bounds	22
4.3	Results using the formulations for the random graphs	24
4.4	Results using the formulations for the geometric graphs	24
4.5	Results using the formulations for the bipartite graphs	25
4.6	Results using the formulations for the complements of bipartite graphs	26
4.7	Results using the formulations for the DIMACS instances with up to 64 vertices	26
5.1	BRKGA results for the random graphs for the connected Grundy coloring problem	36
5.2	BRKGA results for the geometric graphs for the connected Grundy coloring problem	37
5.3	BRKGA results for the complements of bipartite graphs for the connected Grundy coloring problem	38
5.4	BRKGA results for the <i>DIMACS</i> instances for the connected Grundy coloring problem	39
A.1	BRKGA results for the random graphs for the Grundy coloring problem	47
A.2	BRKGA results for the geometric graphs for the Grundy coloring problem	47
A.3	BRKGA results for the bipartite graphs for the Grundy coloring problem	48
A.4	BRKGA results for the complements of bipartite graphs for the Grundy coloring problem	48
A.5	BRKGA results for the DIMACS instances for the Grundy coloring problem	49

LIST OF ALGORITHMS

1	Decoder-Grundy (G, x)	19
2	Decoder-connected-Grundy (G, c)	33

LIST OF ACRONYMS

- AMDF: Adaptive-maximum degree first.
- BRKGA: Biased random-key genetic algorithm.
- CMDF: Connected maximum-degree first.
- CMinDF: Connected minimum-degree first.
- COP: Combinatorial optimization problem.
- GA: Genetic algorithm.
- IP: Integer programming.
- MDF: Maximum-degree first.
- MinDF: Minimum-degree first.
- RKGA: Random-key genetic algorithm.
- SDL: Smallest-degree last.

LIST OF MAIN NOTATIONS

The list below presents a summary of the main notations and definitions to this master's thesis and which will be used throughout this work, unless otherwise stated.

- G : simple undirected graph.
- V : set of vertices.
- E : set of edges.
- K : set of colors.
- c : color mapping function.
- $c(v)$: color of vertex v .
- $\chi(G)$: chromatic number.
- $N(v)$: *neighborhood* of v .
- $\bar{N}(v)$: *anti-neighborhood* of v .
- $\bar{N}[v]$: *closed anti-neighborhood* of v .
- $d(v)$: degree of v .
- $\Delta(G)$: the largest degree of G .
- $den(G)$: the density of G .
- σ : vertex ordering.
- σ_c : connected vertex ordering.
- $\Gamma(G)$: Grundy number.
- $\Gamma_c(G)$: connected Grundy number.
- p : population size.
- p_e : size of the elite population (%).
- p_m : size of the mutant population (%).
- ρ_e : elite inheritance probability (%).
- n_g : number of generations.

CONTENTS

Chapter 1—Introduction	1
1.1 Motivation	1
1.2 Goals and main contribution	1
1.3 Organization	2
Chapter 2—Definitions	3
2.1 Basic definitions	3
2.2 Greedy criteria	5
2.3 Integer Programming	5
2.3.1 Formulation by representatives	7
2.4 Biased random-key genetic algorithm	8
Chapter 3—Related Works	11
Chapter 4—Grundy coloring problem	15
4.1 A new combinatorial upper bound	15
4.2 Integer programming formulations	16
4.2.1 Standard formulation	16
4.2.2 Formulation by representatives	17
4.3 Biased random-key genetic algorithm	18
4.3.1 Solution encoding	18
4.3.2 Solution decoder	18
4.4 Computational experiments	19
4.4.1 Benchmark instances	20
4.4.2 Tested approaches and parameter settings	20
4.4.3 New upper bound results	22
4.4.4 Results for IP formulations	22
4.4.5 BRKGA results	26
Chapter 5—Connected Grundy coloring problem	30
5.1 Integer programming formulations	30
5.1.1 Standard formulation	30
5.1.2 Formulation by representatives	31
5.2 Biased random-key genetic algorithm	32

5.2.1	Solution encoding	32
5.2.2	Decoder	33
5.3	Computational experiments	33
5.3.1	Benchmark instances	34
5.3.2	Tested approaches and parameter settings	34
5.3.3	IP formulations results	35
5.3.4	BRKGA results	35
5.3.5	Comparison between BRKGA solutions to the problems	37
Chapter 6—Conclusion		40
6.1	Future works	41
References		42
Appendix A—BRKGA results for the Grundy coloring problem		46

INTRODUCTION

1.1 MOTIVATION

Combinatorial optimization problems (COPs) consist of obtaining the best solution (minimum or maximum) in a discrete set of possible solutions. Some COPs can be represented as graph coloring problems and have wide practical applications in the real world, some examples are scheduling,(Leighton, 1979; Gamache, Hertz, & Ouellet, 2007), timetabling (de Werra, 1985; Burke, McCollum, Meisels, Petrovic, & Qu, 2007; Babaei, Karimpour, & Hadidi, 2015), register allocation (Chow & Hennessy, 1990; Smith, Ramsey, & Holloway, 2004), communication networks (Zhu, Dai, & Wang, 2015; Pateromichelakis & Samdanis, 2018), video synopsis (He, Gao, Sang, Qu, & Han, 2017), and railway station design (Jovanović, Pavlović, Belošević, & Milinković, 2020). Several of these problems are NP-hard, and for this reason, it is not known whether there is an algorithm that can optimally solve them efficiently (polynomial time).

When it comes to optimization methods, we can classify them based on their capacity to certify optimality. This classification includes exact methods and heuristics. The main difference between them is that given enough time, an exact method will find the optimal solution, whereas a heuristic is not able to certify that it has found the optimal solution. However, this extra time for the exact methods to prove that they found the optimal solution can be very long. Sometimes this approach could take a long time even to find an initial solution for a problem. Although the heuristic does not guarantee an optimal solution, it can provide you with good solutions in a satisfactory time, so it would be interesting to be able to evaluate how good a heuristic is to solve a problem and even more to be able to analyze its worst case (if the result is close to the worst possible value).

1.2 GOALS AND MAIN CONTRIBUTION

In this master's thesis, our main objective is to study the effectiveness of applying the *biased random-key genetic algorithm* (BRKGA) and formulations to determine the Grundy and connected Grundy numbers. Additionally, we seek to analyze the problem's structure to derive an upper bound based on the vertex neighborhood. Furthermore, we aim to assess the effectiveness of the methods in establishing upper bounds through a computational study for

a large set of instances. We also intend to assess their performance as a worst-case analysis criterion for certain minimization greedy criteria.

Unlike the literature on the problems studied in this work, our approach involves employing both heuristic and exact methods, utilizing two integer programming formulations: standard and by representatives. Additionally, we established the first benchmark for the problems through computational tests with a diverse set of instances, since there are no tests on instances of the approaches proposed in the literature. We also provide a new combinatorial upper bound that is valid for both problems and can be computed in polynomial time using dynamic programming. Finally, we will conduct a comparison of methods that aim to minimize the number of colors, representing the worst-case scenario for them.

1.3 ORGANIZATION

The remainder of this work is organized as follows. Chapter 2 presents some basic definitions and explanations of the problem, greedy criteria, integer programming (IP), and BRKGA metaheuristic. Chapter 3 provides a literature review of related works. Chapter 4 presents a new combinatorial upper bound, two IP formulations, a biased random-key genetic algorithm (BRKGA) metaheuristic applied to the Grundy coloring problem, and computational experiments. Chapter 5 presents the BRKGA metaheuristic applied to the connected Grundy coloring problem, two IP formulations, and computational experiments. Finally, Chapter 6 details the concluding remarks of this master's thesis and future works.

DEFINITIONS

2.1 BASIC DEFINITIONS

Given a simple undirected graph $G = (V, E)$, where V is the set of vertices and $E = \{uv \mid u, v \in V\}$ and a set of colors K , a *vertex coloring*, or simply *coloring*, is the mapping $c : V \rightarrow K$. A coloring is said a *proper coloring* when $c(u) \neq c(v)$ for all $uv \in E$. A k -coloring is a proper coloring with exactly k colors. Another way to define it would be through *color classes*, in which a k -coloring of graph G is a partition $\{V_1, V_2, \dots, V_k\}$ of V into k independent sets, where these sets are the *color classes*.

The *chromatic number* of a graph G , $\chi(G)$, is the smallest k such that G admits a proper k -coloring. For simplicity, in the remainder of this text, a coloring is defined as proper unless stated otherwise. Furthermore, $N(v)$ denote the *neighborhood* of v which is composed of all vertices adjacent to v in a graph G and $d(v)$ the degree of v , where $d(v) = |N(v)|$; and by $\bar{N}(v)$ the *anti-neighborhood* of v (formed by the vertices that are not adjacent to v). Additionally, let $\bar{N}[v] = \bar{N}(v) \cup \{v\}$ be the *closed anti-neighborhood* of v . Define the largest degree in G as $\Delta(G) = \max\{d(v) \mid v \in V\}$ and the density of a graph as $den(G) = (2 * E) / (|V| * (|V| - 1))$.

Consider the greedy coloring heuristic *first-fit* which assigns to each vertex v the smallest color that is not present in its neighborhood and given a vertex coloring order $\sigma = (v_1, v_2, \dots, v_n)$, note that there is always exist an order σ for which the *first-fit* coloring heuristic gets a $\chi(G)$ -coloring (Benevides et al., 2014). A *Grundy coloring* is a coloring that respects the properties imposed by the *first-fit* heuristic for any order σ . The *Grundy chromatic number* (or simply Grundy number), $\Gamma(G)$, also known as *first-fit chromatic number*, is the largest k such that G admits a Grundy k -coloring, considering an order σ . The *Grundy coloring problem* is defined as an optimization problem that seeks to maximize the number of colors in a *Grundy coloring*.

Figure 2.1 presents two examples of coloring the same graph following the same order $\sigma = (b, d, e, c, f, a)$ in which one is a Grundy coloring and the other is not. The Subfigure 2.1(b) does not represent a Grundy coloring. This is because, when coloring the vertex c , the smallest color not present in its neighborhood was not selected; in this case, it would be 1.

An ordering $\sigma_c = (v_1, \dots, v_n)$ is said to be connected if the subgraph induced by $\{v_1, \dots, v_i\}$ is connected for all $1 \leq i \leq n$. A *connected Grundy coloring* respects the already mentioned properties of the *first-fit coloring* heuristic using a connected ordering σ_c . The *connected Grundy*

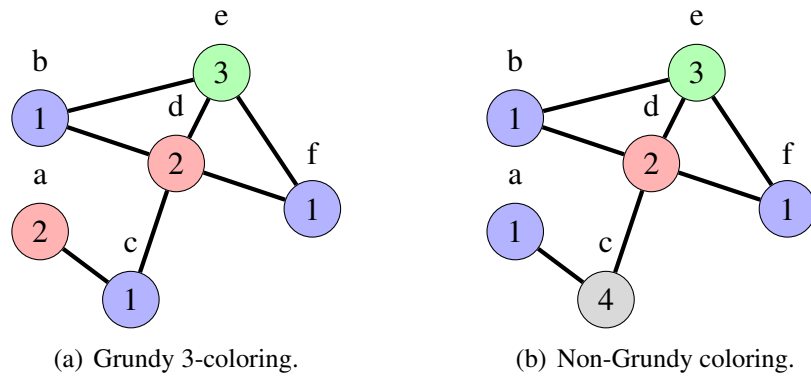


Figure 2.1: Coloring examples using $\sigma = (b, d, e, c, f, a)$

number, $\Gamma_c(G)$, is a more restricted variant where it defines the largest value of k such that G admits a connected k -Grundy-coloring.

Figure 2.2 shows two examples of Grundy colorings, the one on the left 2.2(a) is a non-connected coloring since the order is not connected once the first vertex is selected, and the next one is not a neighbor of the same. It can be noted that there is no way to achieve this coloring in a connected way. This is because for the vertex a receive color 4 and b receive color 3, the vertices $\{d,c,e,f\}$ need to be colored first and this subset does not induce a connected subgraph. The right one 2.2(b) uses a connected order (σ_c) and colors each vertex respecting the *first-fit* heuristic property, therefore the second order generates a connected Grundy coloring.

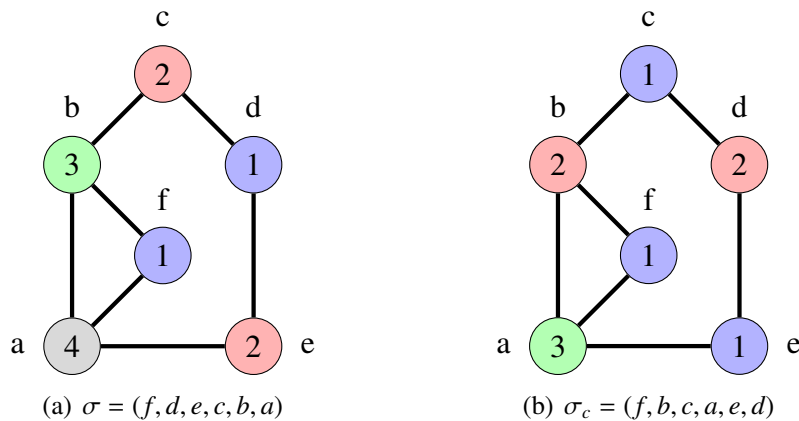


Figure 2.2: Difference in coloring when using a non-connected and connected order with the *first-fit* heuristic.

Given the characteristics of the *first-fit* algorithm, a *Grundy coloring* can be uniquely determined by the order σ of the vertices, since the algorithm establishes that there will always be only one choice for the color (the smallest possible color) for that vertex, depending solely on the color of the neighbors that have already been colored, that is, that precede it in σ . The same is valid for the connected version of the problem with a σ_c sequence.

2.2 GREEDY CRITERIA

Greedy algorithms do not always produce optimal solutions, but for many problems, they do or can produce good solutions. A greedy algorithm consistently opts for the most favorable choice at the current moment (Cormen, Leiserson, Rivest, & Stein, 2009). This choice is based on a greedy criterion that seems promising for the problem. Below we list some well-known greedy criteria in the literature for graph coloring.

- *maximum-degree first* (MDF): Defines a sequence (v_1, \dots, v_n) in which the highest degree vertex is iteratively chosen;
- *adaptive-maximum degree first* (AMDF): Defines a sequence (v_1, \dots, v_n) similar to MDF, with the difference that the degrees of the neighbors of v are decreased by one whenever v is selected;
- *smallest-degree last* (SDL) (Matula & Beck, 1983): Constructs the sequence (v_1, \dots, v_n) based on the gradual removal of the lowest-degree vertex in a subgraph $H \subseteq G$;
- *DSatur* (Brélaz, 1979): Builds a sequence (v_1, \dots, v_n) using an adaptive criterion based on the maximum degree of saturation, where the degree of saturation of a vertex is equal to the number of vertices with different colors that are adjacent to it;
- *connected maximum-degree first* (CMDF): Defines a sequence (v_1, \dots, v_n) in which the highest degree vertex is iteratively chosen that is a neighbor of at least one previously chosen vertex.

Each of these criteria will produce a sequence to which the *first-fit* coloring heuristic can be applied. All these criteria aim to minimize the total number of colors used.

2.3 INTEGER PROGRAMMING

This section focuses on defining what integer programming is and how it can be applied to graph coloring problems and mainly explains the concept of using a formulation by representatives.

"*Integer Programming* is about ways to solve optimization problems with discrete or integer variables" (Wolsey, 2020). A *Mixed Integer Programming* (MIP) can be written as:

$$\max cx + hy \tag{2.1}$$

$$Ax + Gy \leq b, \tag{2.2}$$

$$x \geq 0 \text{ and integer, } y \geq 0 \tag{2.3}$$

Where A is a m by n matrix, G is a m by p , c is a n row-vector, h is a p row-vector, x is a n column-vector of integer variables, and y is a p column-vector of real variables. If all variables are integer, so we can simplify them as:

$$\max cx \tag{2.4}$$

$$Ax \leq b, \tag{2.5}$$

$$x \geq 0 \text{ and integer} \tag{2.6}$$

The first element (2.4) is the objective function, which can be maximization or minimization. The (2.5) represent the set of constraints of a problem, and (2.6) is the integrality constraints. This approach has already proven to be effective in several graph optimization problems, including coloring problems (Melo, Samer, & Urrutia, 2016; Dias, de Freitas, Maculan, & Michelon, 2021). In coloring problems, the classical (standard) method is to use a variable that indicates which color was assigned to a vertex. Let's take the coloring problem where our objective is to determine the chromatic number as an example where we want to minimize the total number of colors used but guarantee a proper coloring. Initially, define a graph $G = (V, E)$ where V is the set of vertices, E is the set of edges and $k = \{1, \dots, |V|\}$ the set of possible colors that can be attributed to a vertex.

To formulate this coloring problem as an integer program, consider the decision variables:

$$x_{kv} = \begin{cases} 1, & \text{if vertex } v \in V \text{ receives color } k \in K_v, \text{ i.e., the } k\text{-th color,} \\ 0, & \text{otherwise;} \end{cases}$$

$$w_k = \begin{cases} 1, & \text{if color } k \in K \text{ is used,} \\ 0, & \text{otherwise.} \end{cases}$$

The coloring problem can thus be formulated as:

$$\min \sum_{k \in K} w_k \tag{2.7}$$

$$x_{uv} + x_{vu} \leq 1, \quad \forall uv \in E \tag{2.8}$$

$$\sum_{k \in K} x_{kv} = 1, \quad \forall v \in V \tag{2.9}$$

$$x_{kv} \leq w_k, \quad \forall k \in K \text{ and } \forall v \in V \tag{2.10}$$

$$x_{kv} \in \{0, 1\}, \quad \forall v \in V, k \in K, \tag{2.11}$$

$$w_k \in \{0, 1\}, \quad \forall k \in K. \tag{2.12}$$

The objective function (2.7) aims to minimize the total number of colors used. Constraint (2.8) guarantees a proper coloring. Constraint (2.9) implies that all vertices must receive exactly one color and (2.10) a vertex v only can receive a color k if the color k was used ($w_k = 1$). Constraints (2.11) and (2.12) guarantee the integrality constraints.

Figure 2.3 illustrates a common problem of coloring problems that occur when using the standard formulation presented previously, which are symmetrical solutions, the same happens in the Grundy coloring problem and connected Grundy coloring problem. There are even more symmetries there if we consider orders because for each of the subfigures, there are more than twenty different vertex orderings that arrive at the same solution if applied the *first-fit* heuristic. In the following section, another way of modeling the same problem will be presented, breaking some symmetries.

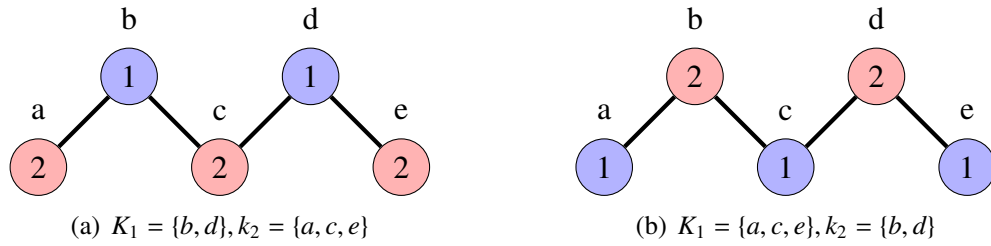


Figure 2.3: Example of possible solutions when coloring a graph using $\chi(G)$ -colors

2.3.1 Formulation by representatives

Formulations by representatives (Correa, 2004; Frota, Maculan, Noronha, & Ribeiro, 2010) have been successfully applied to graph coloring and several other partitioning problems (Campelo, 2005; Bahiense, Frota, Noronha, & Ribeiro, 2014; Melo & Ribeiro, 2015; Melo, Queiroz, & Santos, 2021; Melo, Ribeiro, & Riveaux, 2022). Such formulations come from the insight that one can represent subsets of vertices in the graph by choosing one vertex of the subset to be its representative. To illustrate, let's take the graph in Figure 2.3 and separate the vertices by their color classes, so we have:

- $V_1: \{a, c, e\}$
- $V_2: \{b, d\}$

So if we consider this partition, we can see that they represent the two solutions, just arbitrarily indicating which color each set represents and selecting a vertex from each set to be the representative, in this case, we select the first by alphabetical order. Consequently, symmetry was removed in solutions with the same number of colors that are generated by equal partitioning of the vertices, and symmetry was removed within each set by defining a criterion such as the first element in alphabetical order. This is the main idea behind the use formulation by representatives for coloring problems.

A characteristic of coloring problems that need to guarantee a proper coloring is that there are no two adjacent vertices within the same set V_i because this would imply that both have the same color, therefore the construction always occurs with elements in the anti-neighborhood of the representative.

Rewriting the formulation using the representative idea, we have the following decision variables:

$$X_{vu} = \begin{cases} 1, & \text{if vertex } v \in V \text{ represents the color of vertex } u \in \bar{N}(v), \text{ for } v \leq u \\ 0, & \text{otherwise;} \end{cases}$$

$$X_{vv} = \begin{cases} 1, & \text{if vertex } v \in V \text{ is representative} \\ 0, & \text{otherwise;} \end{cases}$$

Now this coloring problem using representative vertices in the decision variable can thus be formulated as:

$$\min \sum_{v \in V} X_{vv} \quad (2.13)$$

$$X_{vu} + X_{vw} \leq 1, \quad \forall v \in V, u, w \in \bar{N}(v), \text{ s.t. } uw \in E \text{ and } v \leq u < w \quad (2.14)$$

$$\sum_{\substack{v \in \bar{N}[u], \\ v \leq u}} X_{vu} = 1, \quad \forall u \in V, \quad (2.15)$$

$$X_{vu} \in \{0, 1\}, \quad \forall v, u \in V, u \in \bar{N}(v), \text{ s.t. } v \leq u. \quad (2.16)$$

The objective function (2.13) aims to minimize the total number of representative vertices and as each representative vertex refers to a color, consequently it minimizes the total number of colors used. Constraint (2.14) guarantees that if two vertices are neighbors they cannot be represented by the same vertex, i.e., they are not part of the same partition set, so they will have different colors generating a proper coloring. Constraint (2.15) ensures that each vertex is represented by exactly one vertex. The integrality of the variables is in the constraint (2.16). Consequently, in the end, the formulation determines a partition of vertices that represent color classes.

2.4 BIASED RANDOM-KEY GENETIC ALGORITHM

The biased random-key genetic algorithm (BRKGA) was introduced by Gonçalves and Resende (2011) as a general-purpose search metaheuristic for finding good-quality solutions to hard optimization problems. The BRKGA simplifies genetic algorithms in general by making both the representation and the intensification-diversification mechanism problem-independent as follows.

- **Representation:** *Chromosomes* in a BRKGA are represented as a vector of randomly generated real numbers in the interval $[0, 1)$, following the random-key genetic algorithm by Bean (1994). Such random keys or *alleles* define, or *encode*, a single solution to the problem at hand.
- **Intensification:** In the mating process of a BRKGA, which produces the next generation of chromosomes, one parent is always an elite solution (i.e., one with a high fitness value). Furthermore, such a parent has a higher probability of passing its characteristics (defined by its alleles) to the offspring, the other one is a non-elite solution.
- **Diversification:** In every generation, a BRKGA introduces new randomly generated solutions (i.e., mutants) in the population. This prevents premature convergence by allowing the algorithm to escape from local optima regions.

Practitioners are left with the task of calculating the fitness of a chromosome, or *decoding* it. Thus, the main component of a BRKGA implementation is its *decoder*, that is, a deterministic algorithm responsible for mapping a chromosome to a possible solution of the optimization problem at hand. Given a chromosome, the decoder maps it to a solution and computes its objective value. This value is then associated with the chromosome's fitness.

In addition to the decoder, the BRKGA requires and is guided by the following parameters: the population size (p), the size of the elite population (p_e), the size of the mutant population (p_m), the elite inheritance probability (ρ_e), and the number of generations (n_g) or some other stopping criterion (such as maximum time).

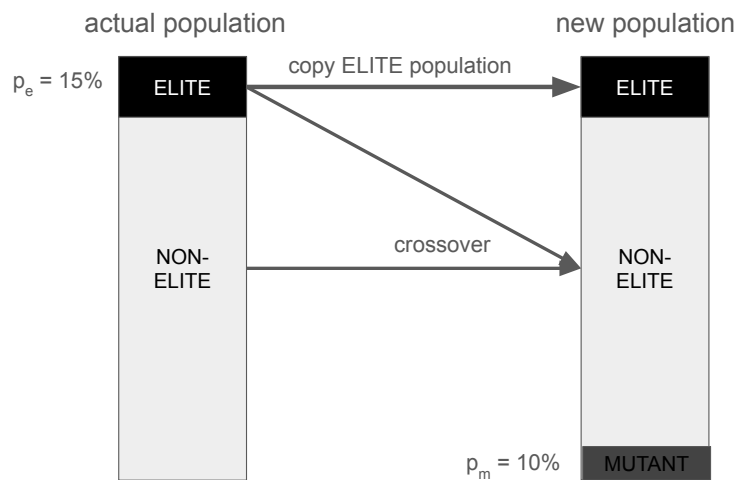


Figure 2.4: Example of how works the evolution stage of the BRKGA between two generations

Figure 2.4 illustrates the evolution between two generations, the entire ELITE population is copied to the next generation, which guarantees non-decreasing (in maximization problems) behavior of the value of the best solution, and the mutant population is created by generating new chromosomes with random keys which is inserted directly into the new population, with no "mutation" in the chromosomes coming from the previous generation.

The crossover (fig. 2.5) always occurs between a chromosome from the ELITE population and a NON-ELITE chromosome, but there is a bias ($\rho_e > 50\%$) so that there is a greater probability of inheriting *alleles* from the ELITE chromosome.

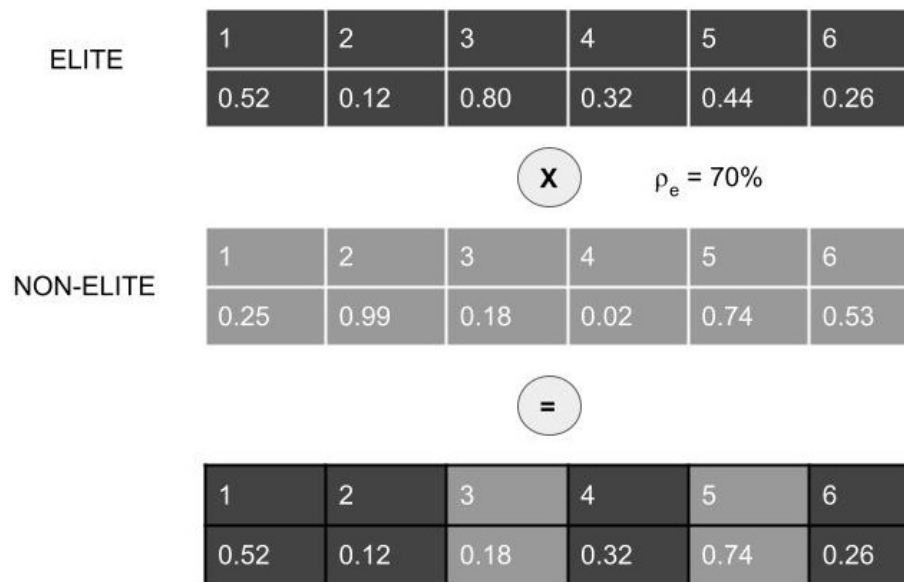


Figure 2.5: Crossover between two chromosomes in which there is a bias for the inheritance of *alleles* from the ELITE parent.

RELATED WORKS

The concept of the Grundy number was first studied by Grundy (1939) in the context of game theory and digraphs (Berge, 1973), where $\Gamma(G)$ was related to the impartiality of a given game state. This concept was formally introduced into graph theory by Christen and Selkow (1979) when studying properties of perfect colorings in which some relationships were established between the Grundy number $\Gamma(G)$, the largest clique of a graph $\mu(G)$, the chromatic number $\chi(G)$, and the *ochromatic* number $\psi(G)$. The last one, the *chromatic number* was proposed independently by Simmons (1983). Later it was demonstrated that *ochromatic number* and Grundy number are equivalent in Erdős, Hare, Hedetniemi, and Laskar (1987).

The *connected Grundy number*, $\Gamma_c(G)$, is a more restricted variant where it defines the largest value of k such that G admits a connected k -Grundy-coloring. It should be noted that contrary to what happens with greedy colorings, there are graphs that do not have a connected greedy coloring with $\chi(G)$ colors (Benevides et al., 2014).

Studies on the Grundy number have focused especially on its complexity for specific graph classes. Despite this, some algorithms have already been proposed to solve the problem, such as a linear time algorithm to find a Grundy coloring in trees (Hedetniemi, Hedetniemi, & Beyer, 1982), this result being extended in (Telle & Proskurowski, 1997) who proposed a dynamic programming algorithm with complexity $k^{O(w)} * 2^{O(wk)} * n = O(n^{3w^2})$, where w is the (*treewidth*). Bonnet, Foucaud, Kim, and Sikora (2018) presented an exact exponential time algorithm $O(2.443^n)$ to the Grundy number problem but left open the question whether there is an $O(c^n)$ exact algorithm for the connected problem, with c a constant. However, there are no computational results from the application of such proposed algorithms in any instance.

When it comes to computational complexity, determining whether $\Gamma(G) \geq k$ is NP-complete for graphs in general (Goyal & Vishwanathan, 1997). Furthermore, the problem is NP-complete even for the complement of bipartite graphs (Zaker, 2005, 2006) and for bipartite graphs (Havet & Sampaio, 2013). However, it can be determined in polynomial time whether the Grundy number is greater than k , as long as k is fixed (Zaker, 2006), on the other hand, it is NP-complete to obtain the connected Grundy number for any $k \geq 7$ (Bonnet et al., 2018). Effantin and Kheddouci (2007) conduct a study on the Grundy number for the following classes of graphs: stable, complete, path, cycle of order n , complete bipartite graphs in $n + p$ vertices, and the cartesian product of two graphs.

The last class of graphs mentioned was the focus of the article in which they related the behavior of the Grundy number for a graph G and the Grundy number for the cartesian product between G and some other graph H . In particular, when assuming these graphs as bipartite, path, complete, and a cycle of order m . Finally, the case of the cartesian product between multiple path graphs was studied. Effantin and Kheddouci (2007) brought another interesting contribution on this article was an algorithm on how to generate all graphs G with the minimum number of edges such that $\Gamma(G) = k$, which can be very useful as a method to generate instances for computational experiments where the optimum is already known, facilitating the evaluation of methods for computing the Grundy number.

A sequence of r different vertices (u_1, \dots, u_r) of G is denoted a *feasible Grundy sequence* if, for $1 \leq i \leq r$, the degree of g_i in $G - \{g_{i+1}, \dots, g_r\}$ is at least $i - 1$. An available upper bound for the Grundy number in G , denoted the *stair factor* $\zeta(G)$, is defined as the size of its maximum cardinality feasible Grundy (Shi et al., 2005).

Note that, in addition to its theoretical aspects, the Grundy number has valuable applicability, as it offers a quality performance comparison (worst-case scenario) for the greedy coloring heuristic *first-fit* that is widely used (Gyárfás & Lehel, 1988; Al-Omari & Sabri, 2006; Ehmsen, Favrholt, Kohrt, & Mihai, 2010). It can also be used with the same comparative purpose for other heuristics that aim to minimize the number of colors. Also, Effantin and Kheddouci (2007) exemplified two practical applications for the Grundy number problem, related to scheduling problems and multiprocessor architecture. Just think of a system in which a process P_i can only be carried out if the processes P_1, P_2, \dots, P_{i-1} have been carried out. Therefore, calculating the Grundy number would answer how many processes can be loaded in this architecture and how many times we need to load processes in the architecture to be able to compute P_n .

Another concept adjacent to the Grundy number is partial Grundy coloring (Shi et al., 2005). The basic idea when coloring vertices is to separate them into color classes $\{V_1, V_2, \dots, V_k\}$, in which each vertex appears in only one class, introducing the concept of *Grundy vertex* which would be a vertex $v \in V_i$ where it is neighboring at least one vertex V_j for all $j < i$. In partial coloring, for each color class there is at least one *Grundy vertex*. Then the partial Grundy number ($\partial \Gamma(G)$) is the largest k such that the graph accepts a partial k -coloring. Shi et al. (2005) showed that $\partial \Gamma(G) \leq \Gamma(G)$ and $\partial \Gamma(G) \leq \zeta(G) \leq \Delta(G) + 1$, which leads to the following extended result:

$$\chi(G) \leq \partial \Gamma(G) \leq \Gamma(G) \leq \zeta(G) \leq \Delta(G) + 1.$$

It should be noticed, however, that the difference $\Gamma(G) - \chi(G)$ can be arbitrarily large (Bonnet et al., 2018). Figure 3.1 exemplifies how the difference can grow arbitrarily, just take two binomial trees T_{k-1} making the root of one a child of the root of the other, and the root of the latter is declared as the root to generate a graph with $\Gamma(T_k) = k$ and $\chi(T_k) = 2$ (Aboulker, Bonnet, Kim, & Sikora, 2023).

In this paper, it was shown that as long as a graph has a sufficiently large *girth*, then there is a partial Grundy coloring for any viable Grundy sequence. Furthermore, if the waist of the graph is greater than 8, then $\partial \Gamma(G) = \zeta(G)$, and also presenting a linear time algorithm for obtaining the Grundy number of a tree. Finally, obtaining the partial Grundy number is NP-hard, as shown in an article through a reduction to the problem of whether the graph is 3-colorable (Shi et al., 2005).

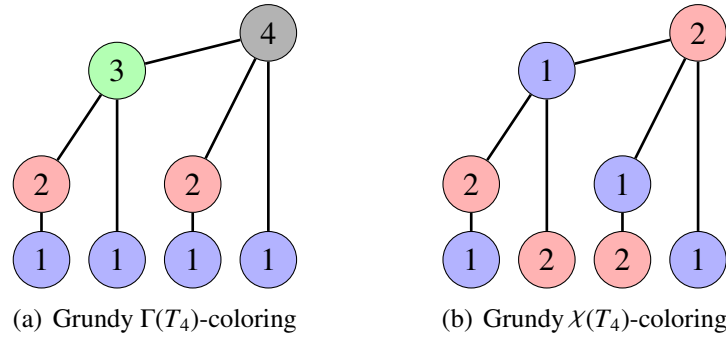


Figure 3.1: Binomial tree T_4 exemplifying the difference between $\Gamma(G)$ and $\chi(G)$ where numbers denote the color of each vertex, and which can be generated using the *first-fit* heuristic.

Connected colorings were also studied in Mota, Rocha, and Silva (2020), in which case they investigated when $\chi_c(G) = \chi(G)$ in H -free graph, being an H -free graph when a graph G does not contain a copy of H as an induced subgraph. The theoretical results involve knowing when one assumes that H belongs to a specific class of graphs, the problem of deciding whether $\chi(G) \leq k$ becomes trivial and when it is still NP-complete, the same for the decision problem $\chi_c(G) \leq k$. Note that here we are dealing with the minimization problem.

Bonamy et al. (2021) did a study on connected edge coloring, in which colors are assigned to the edges, always indicating the smallest color available for that edge. The minimum number of colors is called the chromatic index $\chi'(G)$ and the maximum is called the Grundy index $\Gamma'(G)$, with $\chi'_c(G)$ being defined as the smallest number of colors for a connected coloring of the edges. It has been proven that it is NP-hard to determine whether $\chi'_c(G) > \chi'(G)$, but it is trivial in the case of bipartite graphs resulting in $\chi'_c(G) = \chi'(G)$.

Furthermore, we observed that other problems are somehow related to Grundy coloring. For example, Campêlo and Severín (2021) studied a more generic concept called the Grundy dominant number. A legal dominant sequence of a graph is an ordered dominant set of vertices where each element dominates at least one other not dominated by its predecessors in the sequence, and the size of the largest sequence is called the Grundy dominant number.

This is a particular case of the Grundy coverage problem in hypergraphs, an edge coverage problem, as previously mentioned, is called Grundy's dominant number and also has a version called Grundy's total domination number (Campêlo & Severín, 2021). This problem originates from a problem in the area of game theory, as well as the original problem, in which two players have opposing interests, being exemplified as a problem in which companies compete for service concessions, in which the government wants to maximize the number of companies that offer the service and once the company serves a point A , it needs to provide the service to A 's neighbors.

Campêlo and Severín (2021) was one of the few articles that dealt with a variant of Grundy's problem and that proposed an algorithmic approach, using *tabu search* and integer programming and that also provided the results of computational experiments on a set of instances already used in the literature, as well as a real instance.

Masih and Zaker (2021) studied the relationship between the Grundy number and the b -chromatic number, these two elements are important parameters for graph coloring problems and

have been studied separately, and in this article, some relationships between them are established. The same authors even established the relationship $\Gamma(G) - \log \Gamma(G) \leq b(G)$, as long as the graph has a sufficiently large girth (Masih & Zaker, 2022). Some other minor findings are more specific to some specific classes of graphs.

Another related concept is that of graphs *hard-to-color* (Janczewski, Kubale, Manuszewski, & Piwakowski, 2001), in which a graph is said to be *hard-to-color* for a given algorithm if every implementation of that algorithm results in a non-optimal coloration. In this situation we are talking about minimization algorithms in which their result is $A(G) > \chi(G)$, in the worst case being $A(G) = \Gamma(G)$.

GRUNDY COLORING PROBLEM

This chapter presents the study carried out for the Grundy coloring problem. A new combinatorial limit for the Grundy coloring problem is proposed in Section 4.1 as well as an efficient way to compute it using dynamic programming. Two integer programming formulations are presented in Section 4.2. The standard formulation incorporates the new proposed limit as well as the *stair-factor* limit in an approach that aims to reduce the total number of variables and constraints to be evaluated in relation to a form that does not take this into account. The other formulation is by representatives that exploits the advantages compared to the standard approach mentioned in Section 2.3.1. A metaheuristic approach using the BRKGA for the problem is presented in Section 4.3. Finally, computational experiments with the analysis of the results are presented in Section 4.4.

4.1 A NEW COMBINATORIAL UPPER BOUND

Consider the values $\psi(v, k)$ for $v \in V$ and $k \in \{1, \dots, \Delta(G) + 1\}$ recursively as

$$\psi(v, k) = \begin{cases} \max\{l \mid \exists(u_1, \dots, u_{l-1}) \subseteq N(v) \text{ such that } \psi(u_i, k-1) \geq i, \forall i, 1 \leq i \leq l-1\}, & \text{if } k \geq 2; \\ 1, & \text{otherwise.} \end{cases}$$

Define the *connected degree sequence value* $\Psi(G)$ as:

$$\Psi(G) = \max_{u \in V} \{\psi(u, \Delta(G) + 1)\}.$$

Proposition 1. $\Gamma(G) \leq \Psi(G) \leq \Delta(G) + 1$.

Proof. Consider any feasible Grundy coloring c using k colors. Notice that any vertex can receive color 1. On the other hand, for a vertex v to receive color k , all the other colors in $\{1, \dots, k-1\}$ have to be already used for its neighbors. Recursively, each neighbor receiving color $i \in \{2, \dots, k-1\}$ must have at least one neighbor with each of the colors in $\{1, \dots, i-1\}$. Therefore, $c(v) \leq \psi(v, c(v))$. In addition, as ψ is a nondecreasing function of k , it follows that $\psi(v, c(v)) \leq \psi(v, \Delta(G) + 1)$. Hence, $\Gamma(G) \leq \Psi(G)$. \blacksquare

Proposition 2. $\Psi(G)$ can be calculated in polynomial time.

Proof. We provide a simple $O(|V|\Delta(G)^2)$ dynamic programming-based algorithm for calculating $\Psi(G)$. We remark that $\psi(v, k)$ can be calculated in linear time $O(\Delta(G))$ using dynamic programming for each pair (v, k) if the values for $\psi(u, k - 1)$ are known for every $u \in V$. To see how, assume that the vertices in $N(v)$ are in nondecreasing order based on the values $\psi(u, k - 1)$, as such an ordering can be performed in $O(\Delta(G))$ using counting sort. Define $M(j)$ to be the largest l such that there is a sequence $(0, 1, \dots, l)$ in the vertices indexed from 0 to j , where index 0 corresponds to a dummy vertex and $0 \leq j \leq |N(v)|$. Besides, let $P(j)$ be the last element in such a sequence. Define $M(0) = P(0) = 0$. Thus, if $d(v_j) \leq M(j - 1)$, we set $M(j) = M(j - 1)$ and $P(j) = P(j - 1)$. Otherwise, we set $M(j) = M(j - 1) + 1$ and $P(j) = j$. Finally, $\Psi(G) = \max_{u \in V} \{\psi(u, \Delta(G) + 1)\}$. Thus, the $O(|V|\Delta(G))$ elements of ψ can be calculated in $O(|\Delta(G)|)$ each, implying a total running time of $O(|V|\Delta(G)^2)$. \blacksquare

4.2 INTEGER PROGRAMMING FORMULATIONS

Integer programming (IP) has shown to be effective for tackling several graph optimization problems, including graph coloring and variants (Melo et al., 2016; Furini, Malaguti, & Santini, 2018; de Freitas, Dias, Maculan, & Szwarcfiter, 2021; Dias et al., 2021; Marzo, Melo, Ribeiro, & Santos, 2022; Melo & Ribeiro, 2022, 2023). In this section, we formulate the Grundy coloring problem as integer programs. Section 4.2.1 presents a formulation applying a standard methodology in integer programming for coloring problems, while Section 4.2.2 describes a formulation employing a methodology known as representatives. In what follows, for the sake of simplicity, denote the set of vertices by $V = \{1, \dots, n\}$. Besides, define the sequence of available colors as $K = \{1, \dots, \min(\zeta(G), \Psi(G))\}$, and $K_v = \{k' \in K \mid k' \leq \min(\zeta(G), \psi(v, \Delta(G) + 1))\}$ the sequence of possible colors for the vertex v . Finally, define $V_k = \{v \in V \mid k \in K_v\}$ as the set of vertices that can receive the k -th color.

4.2.1 Standard formulation

To formulate the Grundy coloring problem as an integer program, consider the decision variables:

$$x_{kv} = \begin{cases} 1, & \text{if vertex } v \in V \text{ receives color } k \in K_v, \text{ i.e., the } k\text{-th color,} \\ 0, & \text{otherwise;} \end{cases}$$

$$w_k = \begin{cases} 1, & \text{if color } k \in K \text{ is used,} \\ 0, & \text{otherwise.} \end{cases}$$

The Grundy coloring problem can thus be formulated as:

$$\max \sum_{k \in K} w_k \tag{4.1}$$

$$x_{ku} + x_{kv} \leq w_k, \quad \forall k \in K_v \cap K_u, uv \in E, \tag{4.2}$$

$$x_{kv} \leq w_k, \quad \forall k \in K_v, v \in V, \text{ s.t. } |N(v)| = 0, \tag{4.3}$$

$$\sum_{k \in K_v} x_{kv} = 1, \quad \forall v \in V, \quad (4.4)$$

$$w_k \leq \sum_{v \in V_k} x_{kv}, \quad \forall k \in K, \quad (4.5)$$

$$x_{k'v} \leq \sum_{u \in N(v) \cap V_k} x_{ku}, \quad \forall v \in V, k, k' \in K_v, \text{ with } k < k', \quad (4.6)$$

$$w_{k'} \leq w_k, \quad \forall k, k' \in K, \text{ with } k < k', \quad (4.7)$$

$$x_{kv} \in \{0, 1\}, \quad \forall v \in V, k \in K_v, \quad (4.8)$$

$$w_k \in \{0, 1\}, \quad \forall k \in K. \quad (4.9)$$

The objective function (4.1) maximizes the number of used colors. Constraints (4.2)-(4.3) ensure that adjacent vertices do not receive the same color and that a vertex can only receive a color when that color is used. Constraints (4.4) establish that each vertex receives exactly one color. Constraints (4.5) determine that w_k is only set to one if color k is used for at least one vertex. Constraints (4.6) guarantee the Grundy property, i.e., that a color is only used for a vertex if each of the previous colors was used for at least one of its neighbors. Constraints (4.7) impose an order on the used colors. Constraints (4.8)-(4.9) define the integrality requirements of the variables.

4.2.2 Formulation by representatives

In the following, we describe a formulation by representatives for the Grundy coloring problem. Besides the variables representing the subsets of vertices, we use a second set of variables to represent the relative order between the colors (representatives) to ensure the Grundy property.

Consider the decision variables defined as follows:

$$X_{vu} = \begin{cases} 1, & \text{if vertex } v \in V \text{ represents the color of vertex } u \in \bar{N}(v), \text{ for } v \leq u; \\ 0, & \text{otherwise;} \end{cases}$$

$$X_{vv} = \begin{cases} 1, & \text{if vertex } v \in V \text{ is a representative,} \\ 0, & \text{otherwise;} \end{cases}$$

$$y_{vu} = \begin{cases} 1, & \text{if vertices } v, u \in V \text{ are representatives and the color of } v \text{ precedes the} \\ & \text{color of } u, \text{ for } v \neq u, \\ 0, & \text{otherwise.} \end{cases}$$

Additionally, let ϕ_v be a potential variable associated with each vertex $v \in V$. The potential variables ensure the solution is acyclic by defining that the potential of a vertex increases as the number of vertices preceding it in the sequence grows. The problem can be cast as the following IP formulation by representatives:

$$\max \sum_{v \in V} X_{vv} \quad (4.10)$$

$$X_{uv} + X_{uw} \leq X_{uu}, \quad \forall u \in V, v, w \in \bar{N}(u), \text{ s.t. } vw \in E \text{ and } u \leq v < w, \quad (4.11)$$

$$X_{uv} \leq X_{uu}, \quad \forall u \in V, v \in \bar{N}(u), \text{ s.t. } N(v) \cap \bar{N}(u) = \emptyset \text{ and } u \leq v, \quad (4.12)$$

$$\sum_{\substack{v \in \bar{N}[u], \\ v \leq u}} X_{vu} = 1, \quad \forall u \in V, \quad (4.13)$$

$$X_{uv} \leq \sum_{\substack{w \in N(v) \cap \bar{N}[p], \\ w \geq p}} X_{pw} + 1 - y_{pu}, \quad \forall u, p \in V, v \in \bar{N}[u], \text{ s.t. } p \neq u \text{ and } u \leq v, \quad (4.14)$$

$$y_{vu} + y_{uv} \geq X_{uu} + X_{vv} - 1, \quad \forall u, v \in V, \text{ s.t. } u < v, \quad (4.15)$$

$$y_{uv} + y_{vu} \leq X_{uu}, \quad \forall u, v \in V, \text{ s.t. } u \neq v, \quad (4.16)$$

$$\phi_u - \phi_v + 1 \leq |V|(1 - y_{uv}), \quad \forall u, v \in V, \text{ s.t. } u \neq v, \quad (4.17)$$

$$0 \leq \phi_v \leq |V| - 1, \quad \forall v \in V, \quad (4.18)$$

$$X_{uv} \in \{0, 1\}, \quad \forall u \in V, v \in \bar{N}[u], \text{ s.t. } u \leq v, \quad (4.19)$$

$$y_{uv} \in \{0, 1\}, \quad \forall u, v \in V, \text{ s.t. } u \neq v, \quad (4.20)$$

$$\phi_v \in \mathbb{R}_+, \quad \forall v \in V. \quad (4.21)$$

The objective function (4.10) maximizes the number of representative vertices, which determine the colors. Constraints (4.11)-(4.12) guarantee that adjacent vertices do not receive the same color (do not have the same representative) and that a vertex can only represent others if it is a representative. Constraints (4.13) ensure that every vertex has a color. Constraints (4.14) force a vertex v receiving color u to have at least one neighbor w that receives the color represented by a vertex p whenever the color represented by p precedes that of u . Constraints (4.15)-(4.16) determine an order between two vertices if and only if both are representatives. Constraints (4.17)-(4.18) guarantee the order between the representative vertices. Constraints (4.17) implies that if a representative vertex precedes another one, then it must have a smaller potential. Constraints (4.18) define the potential range of a vertex. Constraints (4.19)-(4.20) define the integrality requirements on the variables. Constraints (4.21) restrain the domain of the potential variables.

4.3 BIASED RANDOM-KEY GENETIC ALGORITHM

This section presents the chromosome encoding and the decoder algorithm used in BRKGA for the Grundy coloring problem.

4.3.1 Solution encoding

Each solution is encoded as a vector x of random keys of length $\Upsilon = |V|$, where the i -th random key corresponds to the i -th vertex of G . So, in the example 4.1 the first vertex has key 0.25, the second one has the key 0.90, and so on.

4.3.2 Solution decoder

The decoder defines the vertex coloring order: the i -th random key determines the priority of the i -th vertex to be selected. The decoder sorts the vector of random keys in non-increasing order, thus inducing a coloring order for the nodes of the graph. A simple function is then

V	1	2	3	4	5	6
key	0.25	0.90	0.12	0.88	0.55	0.34

Figure 4.1: Encoded chromosome representing the solution to a graph with 6 vertices where the indices in the first line represent the vertices and the line below contains the key for each of them.

performed on each selected key to color the vertex associated with that key with the smallest possible color based on the *first-fit* algorithm. By design, the resulting coloring is a Grundy coloring.

Algorithm 1 implements the decoder, using the vector of random keys x to construct the Grundy coloring for G . The line 1 initializes the vector that stores the selected colors for each vertex. The loop at lines 2-3 is responsible for coloring the vertex with the highest key at each iteration. The procedure *Color-Vertex* at line 3 implements the *first-fit* algorithm that chooses the lowest-index color possible for vertex v that has not yet been selected for any of its neighbors. The line 4 returns the total number of colors used, i.e., the fitness of the Grundy coloring constructed from x for the graph G .

Algorithm 1: Decoder-Grundy (G, x)

```

1  $colors \leftarrow \{0, \dots, 0\};$ 
2 foreach  $v \in V$  in non increasing order of their keys  $x_v$  do
3    $colors[v] \leftarrow \text{Color-Vertex}(G, v, colors);$ 
4 return  $\max_{v \in V} colors[v];$ 

```

The running time of Algorithm 1 can be determined as follows. Notice that sorting the vertices according to their keys can be done in $O(|V| \log |V|)$. The complexity of the loop in lines 2-3 corresponds to the cost of traversing the graph's adjacency list, which runs in $O(|V| + |E|)$. Retrieving the maximum value in $colors$ (line 4) can be computed in $O(|V|)$. Therefore, Algorithm 1 can be implemented to run in $O(|V| \log |V| + |E|)$ time.

4.4 COMPUTATIONAL EXPERIMENTS

All the experiments were executed on a machine running Ubuntu x86-64 GNU/Linux, with an Intel Core i7-10700 Octa-Core 2.90 GHz processor and 16Gb of RAM. The formulations

were implemented in Julia and solved with Gurobi 10.0.1. The BRKGA was developed in C++ using the BRKGA API (Toso & Resende, 2015; Toso, 2018).

4.4.1 Benchmark instances

The benchmark set comprises graphs that were already used in the literature for other coloring problems (Melo et al., 2021). It contains (a) random graphs, (b) geometric graphs, (c) bipartite graphs, (d) the complements of bipartite graphs, and (e) instances from the second *DIMACS Implementation Challenge* (Trick et al., 2015).

Instances (a)-(c) were created with the graph generator *ggen* (Morgenstern, n.d.) by Melo et al. (2021). They have $|V| \in \{50, 60, 70, 80\}$ and were generated with $\eta \in \{0.2, 0.4, 0.6, 0.8\}$ as the probability of having an edge (for random and bipartite graphs) or the existence of an edge if the Euclidean distance between vertices is less than or equal to η (for geometric graphs). The instances (d) correspond to the complements of the bipartite graphs defined in (c). To control for the randomness of the instance generator, there are five instances for each instance group, where a group corresponds to a combination of graph class, number of vertices, and Euclidean distance (or probability for random and bipartite graphs). The groups are identified by $C_|V|_ \eta$, where C represents the graph class: random (rand), geometric (geo), bipartite (bip), and complement of bipartite (cbip). The results are aggregated for each instance group and presented as the average among its five instances. Notice that there are 320 of such instances, 80 for each graph class. From now on, we denote these by small instances.

Instances (e) are a subset of instances from the Second DIMACS Implementation Challenge with up to 500 vertices. This set consists of 42 instances with $|V| \in [28, 500]$. Their characteristics (number of vertices and density) are shown in Table 4.1. These instances are widely used in the literature, especially for coloring and maximum clique problems (Avanthay, Hertz, & Zufferey, 2003; Lü & Hao, 2010; Moalic & Gondran, 2018; Nogueira, Pinheiro, & Subramanian, 2018; San Segundo, Coniglio, Furini, & Ljubić, 2019; Melo et al., 2021).

4.4.2 Tested approaches and parameter settings

The following approaches were considered in the computational experiments:

- The new combinatorial upper bound proposed in Section 4.1 ($\Psi(G)$);
- The standard formulation described in Section 4.2.1 (std);
- The formulation by representatives detailed in Section 4.2.2 (rep);
- The BRKGA described in Section 4.3 (BRKGA);
- The greedy criterion *minimum-degree first* (MinDF), that defines a coloring sequence (v_1, \dots, v_n) prioritizing the vertices with lower degree.

The goal of MinDF is to generate a sequence that maximizes the number of colors used by giving priority to the lowest-degree vertices. Thus, the greedy criterion tries to force vertices with lower degrees to receive the lowest colors, attempting to allow vertices with higher degrees to receive colors with higher indices. Additionally, some widely-used and well-established

Table 4.1: Characteristics of the DIMACS instances

instance	$ V $	$den(G)$	instance	$ V $	$den(G)$
johnson8-2-4	28	0.55	c-fat200-5	200	0.42
johnson8-4-4	70	0.78	zeroin.i.1	211	0.18
mann_a9	45	0.92	zeroin.i.2	206	0.15
hamming6-2	64	0.90	zeroin.i.3	206	0.16
hamming6-4	64	0.34	dsjc250.1	250	0.10
c125.9	125	0.89	r250.1	250	0.02
dsjc125.1	125	0.09	hamming8-2	256	0.96
dsjc125.5	125	0.50	hamming8-4	256	0.63
dsjc125.9	125	0.89	fpsol2.i.2	451	0.08
r125.1	125	0.02	fpsol2.i.3	425	0.09
r125.1c	125	0.96	le450_5a	450	0.05
r125.5	125	0.49	le450_5b	450	0.05
keller4	147	0.64	le450_5c	450	0.09
mulsol.i.1	197	0.20	le450_5d	450	0.09
mulsol.i.2	188	0.22	le450_15a	450	0.08
mulsol.i.3	184	0.23	le450_15b	450	0.08
mulsol.i.4	185	0.23	le450_25a	450	0.08
mulsol.i.5	186	0.23	le450_25b	450	0.08
brock200_2	200	0.49	dsjr500.1	500	0.02
c-fat200-1	200	0.07	c-fat500-1	500	0.03
c-fat200-2	200	0.16	c-fat500-2	500	0.07

greedy criteria for the coloring problem (that aims to minimize the number of colors) were used as baselines: *MDF*, *AMDF*, *SDL*, *DSatur*.

The Gurobi solver was set with the default configurations and a single thread. A time limit of 3600 seconds (1 hour) was given for each formulation to solve each of the instances. A warm start (i.e., initial feasible solution) was given to the formulations, provided by the best solution achieved using any of the greedy heuristics (MinDF, MDF, AMDF, SDL, DSatur).

The settings for the BRKGA were defined as follows. All executions were performed using a single thread. A time limit of 300 seconds (five minutes) was defined as the stopping criterion instead of the number of generations. Though this choice makes it harder to reproduce, it facilitates the control of the total run time and benefits efficient decoders. The parameters were defined based on preliminary tests considering a subset of 28 instances. The instances were chosen to be a representative sample, ensuring that at least 5 instances were randomly selected from each class and varying the densities and number of vertices. The test considered the following possible settings: $p = \{1 \times |V|, 2 \times |V|, 3 \times |V|\}$, $p_e = \{5\%, 15\%, 30\%\}$, $p_m = \{5\%, 10\%, 30\%\}$ and $\rho_e = \{60\%, 70\%, 90\%\}$. In total, 81 configurations were tested for which the BRKGA was run with five different seeds. The considered values were based on how the metaheuristic converges over the generations by varying each of the parameters (Gonçalves & Resende, 2011). Finally, the configuration selected for the overall experiments was $(p, p_e, p_m, \rho_e) = (2 \times |V|, 30\%, 10\%, 60\%)$.

4.4.3 New upper bound results

In this section, we summarize the results achieved using the new upper bound $\Psi(G)$ when compared to the trivial upper bound $\Delta(G) + 1$ and $\zeta(G)$. Table 4.2 shows, for each instance class (first column), the percentage of the instances for which $\Psi(G)$ strictly improves over $\Delta(G) + 1$ (second column), strictly improves over $\zeta(G)$ (third column), and at least matches $\zeta(G)$. We remark that the DIMACS instances do not exactly represent a graph class, but we consider them as one only for presentation purposes. The results indicate that the new upper bound is able to improve over the best available combinatorial bound for several instances (19.0%). Most of the improvements are concentrated on the bipartite graphs (63.7%), followed by the DIMACS instances (19.0%) and the random graphs (11.3%).

Table 4.2: Comparison of upper bounds

Class	$\Psi(G) < \Delta(G) + 1$	$\Psi(G) < \zeta(G)$	$\Psi(G) \leq \zeta(G)$
Random	60.0%	11.3%	33.7%
Geometric	32.5%	1.2%	15.0%
Bipartite	73.7%	63.7%	95.0%
Complement of bipartite	22.5%	0.0%	5.0%
DIMACS	54.7%	19.0%	30.0%
Total	48.0%	19.0%	36.3%

Figure 4.2 visually compares the bounds $\Psi(G)$ and $\zeta(G)$ for all instances with the exception of the DIMACS instances. Its y -axis represents the percentage difference between the bounds, with $diff = \frac{100 \times (\Psi(G) - \zeta(G))}{\Psi(G)}$ and its x -axis provides the instances in non-decreasing order of their $diff$ values. Notice that negative values of $diff$ indicate that $\Psi(G)$ improves over $\zeta(G)$, and positive values indicate the opposite. We can see that $\zeta(G)$ provides a better bound for a larger number of instances. However, the negative values on the left side of the graph show that $\Psi(G)$ can significantly improve the bounds provided by $\zeta(G)$ for a subset of the instances, and these improvements can be as large as 60%.

Figure 4.3 provides the same information as Figure 4.2, but now for the DIMACS instances. It shows that the bounds provided by $\zeta(G)$ and $\Psi(G)$ are the same for several instances and that $\zeta(G)$ achieves better bounds than $\Psi(G)$ for a more significant number of instances. We can observe that the improvements achieved by $\Psi(G)$ over $\zeta(G)$ can reach values as high as 13%.

4.4.4 Results for IP formulations

Tables 4.3-4.6 summarize the results using the formulations for the benchmark instances (a)-(d), which have up to 80 vertices. In these tables, the first column represents the instance group so that each row corresponds to the average values over its five instances. The second column (UB) provides the best upper bound, considering $\zeta(G)$ and $\Psi(G)$. The third (h) indicates the average value of the initial solution value provided for the formulation. In what follows, for each of the formulations, the columns indicate the average of the best solutions found (best), the average execution time (time), the average optimality *gap* in percentage, and the number of instances solved to optimality (#opt). The last two rows of the tables indicate the average of the

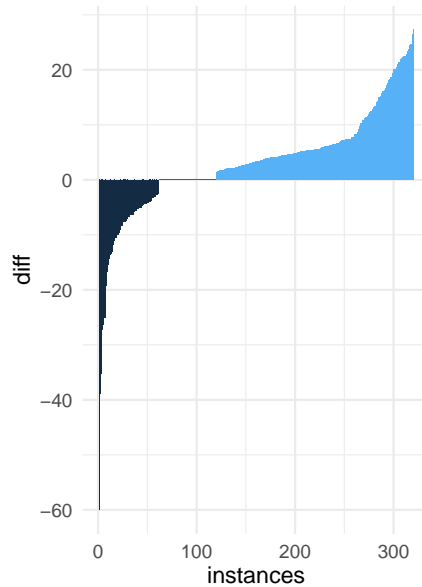


Figure 4.2: Difference in percent between the bounds $\zeta(G)$ and $\Psi(G)$ for the small instances

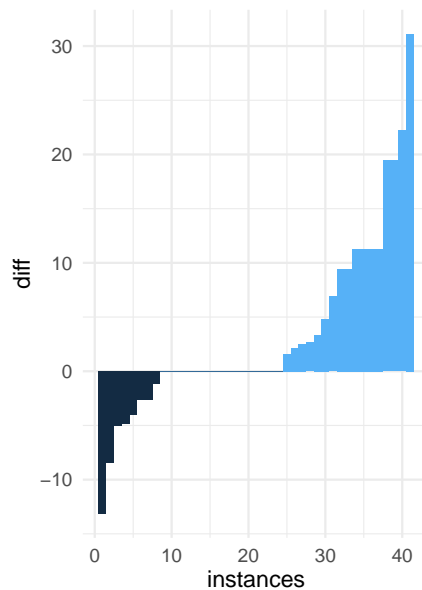


Figure 4.3: Difference in percent between the bounds $\zeta(G)$ and $\Psi(G)$ for the DIMACS instances

values across all rows and the total number of instances solved to optimality. The largest values in the columns best are highlighted in bold.

Table 4.3 shows the results for the random graphs. It shows that the formulations managed to improve the initial solutions for all the instance groups. In addition, on average, the formulation by representatives reached better solutions, finding five optimal solutions for the group *rand_50_0.8* and two for *rand_60_0.8*. However, the average gap in the standard formulation was smaller. It is interesting to notice the pattern in this table, in which the standard formulation

Table 4.3: Results using the formulations for the random graphs

group	ub	h	std				rep			
			best	time	gap	#opt	best	time	gap	#opt
rand_50_0.2	17.0	7.6	10.8	3600.0	33.7	0	10.4	3600.0	123.6	0
rand_50_0.4	26.0	11.8	15.8	3600.0	57.3	0	15.4	3600.0	84.6	0
rand_50_0.6	35.4	16.6	22.6	3600.0	59.4	0	22.6	3600.0	33.7	0
rand_50_0.8	43.4	23.8	29.6	3600.0	52.9	0	31.6	146.5	0.0	5
rand_60_0.2	18.4	8.6	12.0	3600.0	45.3	0	11.6	3600.0	156.3	0
rand_60_0.4	31.2	12.8	18.8	3600.0	67.0	0	17.8	3600.0	97.2	0
rand_60_0.6	42.4	19.0	25.0	3600.0	71.2	0	26.2	3600.0	44.4	0
rand_60_0.8	52.0	27.6	34.2	3600.0	57.9	0	36.8	2754.8	2.2	2
rand_70_0.2	21.2	9.2	12.6	3600.0	58.7	0	12.0	3600.0	194.0	0
rand_70_0.4	36.8	15.8	20.2	3600.0	82.3	0	19.8	3600.1	111.2	0
rand_70_0.6	48.6	21.0	27.4	3600.0	80.4	0	28.6	3600.0	57.4	0
rand_70_0.8	60.6	30.8	37.6	3600.3	68.2	0	40.0	3600.0	12.5	0
rand_80_0.2	24.4	11.2	13.6	3600.0	73.6	0	12.8	3600.0	219.7	0
rand_80_0.4	40.6	16.0	21.0	3600.0	92.6	0	20.2	3600.0	137.7	0
rand_80_0.6	55.6	23.8	29.8	3600.1	90.2	0	30.6	3600.0	69.4	0
rand_80_0.8	68.6	32.6	40.8	3600.2	74.6	0	44.8	3600.0	16.1	0
Mean	38.8	18.0	23.3	3600.0	66.5		23.8	3331.1	85.0	
Total						0				7

performs the best for the values of probability $\rho \in \{0.2, 0.4\}$, while the formulation by representatives achieves the best results for $p \in \{0.6, 0.8\}$. A possible explanation for this is as follows. If we observe the formulation by representatives (Section 4.2.2), the denser a graph, the smaller the anti-neighborhood of a vertex, then it will have a reduced number of variables X and fewer constraints. This ends up reducing the number of possible choices.

Table 4.4: Results using the formulations for the geometric graphs

group	ub	h	std				rep			
			best	time	gap	#opt	best	time	gap	#opt
geo_50_0.2	10.6	7.2	9.0	34.7	0.0	5	9.0	3600.0	91.5	0
geo_50_0.4	25.4	18.0	22.2	3600.0	19.8	0	22.2	3600.0	29.4	0
geo_50_0.6	35.2	26.6	31.0	3600.0	41.1	0	31.2	3434.3	7.5	1
geo_50_0.8	41.0	34.8	37.6	3600.0	32.6	0	38.2	642.6	0.0	5
geo_60_0.2	12.2	7.8	10.2	730.1	1.8	4	10.0	3600.0	140.9	0
geo_60_0.4	31.2	19.4	26.6	3600.0	23.9	0	26.4	3600.0	35.1	0
geo_60_0.6	42.6	32.6	37.6	3600.0	44.1	0	37.8	3600.0	8.2	0
geo_60_0.8	50.8	44.6	47.2	3600.0	26.3	0	47.6	57.2	0.0	5
geo_70_0.2	14.4	9.6	12.6	1367.6	1.4	4	12.2	3600.0	146.7	0
geo_70_0.4	33.8	21.0	28.4	3600.0	27.5	0	27.4	3600.0	52.9	0
geo_70_0.6	48.0	36.4	41.8	3600.0	44.5	0	42.0	3600.0	14.1	0
geo_70_0.8	58.4	48.6	53.4	3600.1	31.5	0	54.4	935.4	0.3	4
geo_80_0.2	15.0	10.4	12.8	2010.9	3.4	3	11.8	3600.0	190.6	0
geo_80_0.4	38.0	24.4	32.2	3600.0	28.5	0	31.6	3600.1	53.7	0
geo_80_0.6	55.8	40.6	47.6	3600.0	50.0	0	48.2	3600.0	15.1	0
geo_80_0.8	68.4	58.2	61.6	3600.1	29.9	0	63.2	206.9	0.0	5
Mean	36.6	27.5	31.9	2959.0	25.4		32.0	2804.8	49.1	
Total						16				20

Table 4.4 provides the results for the geometric graphs. It indicates that the geometric instances seem to be easier to solve, with the standard formulation and that of representatives being able to prove optimality of, respectively, 16 and 20 instances. Again, the formulation by representatives achieved better results for most instances. However, on average, the performances of the two formulations were very close. Another interesting point is that the pattern found for random instances is repeated for the geometric instances with 60 or more vertices. That is, the formulation by representatives has better results when the Euclidean distance is 0.6 and 0.8, and the standard formulation in the others. It should be noticed that, together, the formulations solved 36 out of the 80 instances.

Table 4.5: Results using the formulations for the bipartite graphs

group	ub	h	std				rep			
			best	time	gap	#opt	best	time	gap	#opt
bip_50_0.2	9.8	4.8	8.2	636.1	0.0	5	7.2	3600.0	142.1	0
bip_50_0.4	15.8	4.0	10.8	3600.0	33.5	0	10.0	3600.0	130.0	0
bip_50_0.6	20.4	2.8	13.8	3600.0	40.5	0	13.8	3600.0	82.7	0
bip_50_0.8	21.0	2.0	16.8	3600.0	19.3	0	17.0	3137.9	21.5	1
bip_60_0.2	11.6	4.2	9.2	2369.3	4.4	3	7.8	3600.0	198.2	0
bip_60_0.4	17.2	2.4	11.6	3600.0	36.5	0	11.0	3600.0	158.2	0
bip_60_0.6	23.2	3.4	14.8	3600.0	48.5	0	14.4	3600.0	111.1	0
bip_60_0.8	27.4	2.2	19.8	3600.0	35.1	0	19.6	3600.0	47.8	0
bip_70_0.2	12.4	5.0	9.4	2882.5	16.9	1	8.6	3600.1	230.3	0
bip_70_0.4	18.6	3.4	12.6	3600.0	41.3	0	11.6	3600.0	192.0	0
bip_70_0.6	25.4	2.8	16.4	3600.0	50.1	0	15.8	3600.0	130.3	0
bip_70_0.8	29.6	2.0	21.0	3600.0	37.7	0	21.2	3600.0	65.0	0
bip_80_0.2	14.4	5.4	10.0	3600.0	30.0	0	8.8	3600.3	272.5	0
bip_80_0.4	23.0	4.2	13.6	3600.0	64.7	0	11.6	3600.0	247.3	0
bip_80_0.6	29.4	2.4	18.4	3600.0	56.8	0	17.6	3600.1	145.7	0
bip_80_0.8	35.4	2.4	22.8	3600.0	51.7	0	22.8	3600.0	87.9	0
Mean	20.9	3.3	14.3	3292.9	35.4		13.6	3571.6	141.4	
Total						9				0

Table 4.5 shows the results for the bipartite graphs. It evidences a clear predominance of the standard formulation as it could prove the optimality for nine instances and had an average gap approximately four times smaller. It is worth mentioning that this set of bipartite graphs is not very dense. On the other hand, the results for the complements of these graphs, which are available in Table 4.6, show the advantage of the formulation by representatives for denser graphs. They show that this formulation achieved optimality for 14 instances and obtained an average gap of 11.2%, being approximately three times smaller than that of the standard formulation.

Table 4.7 presents the results of the formulations for the DIMACS instances with up to 64 vertices. Unlike Tables 4.3-4.6, each of its rows corresponds to a single instance. Thus, although the columns have the same meaning as the previous ones, each row no longer provides the average values but that found for the specific instance. It can be noticed that both formulations arrive at the same result in four of the five cases, but the formulation by representatives proves the optimality for two of these cases. We remark that the experiments performed showed that the formulations started facing difficulties with instances having more than 80 vertices with our

Table 4.6: Results using the formulations for the complements of bipartite graphs

group	ub	h	std				rep			
			best	time	gap	#opt	best	time	gap	#opt
cbip_50_0.2	46.4	30.0	34.0	3600.0	42.4	0	34.6	7.5	0.0	5
cbip_50_0.4	43.0	26.6	30.4	3600.0	45.4	0	31.2	3600.0	5.1	0
cbip_50_0.6	38.6	26.6	29.0	3600.0	35.9	0	29.2	3600.0	13.0	0
cbip_50_0.8	35.2	29.6	30.4	3600.0	21.4	0	30.4	3600.0	13.3	0
cbip_60_0.2	56.0	35.6	39.4	3600.1	47.6	0	41.4	36.3	0.0	5
cbip_60_0.4	51.6	34.2	37.0	3600.0	46.1	0	37.8	3120.4	7.1	1
cbip_60_0.6	46.4	32.6	35.0	3600.0	38.4	0	35.2	3600.0	15.0	0
cbip_60_0.8	41.0	33.0	34.4	3600.0	22.3	0	34.4	3600.0	18.6	0
cbip_70_0.2	65.2	40.8	45.8	3600.1	47.7	0	47.2	2189.1	1.7	2
cbip_70_0.4	60.0	41.4	43.8	3600.1	44.4	0	44.4	3600.0	9.7	0
cbip_70_0.6	54.8	39.8	41.2	3600.0	38.6	0	42.0	3600.0	15.0	0
cbip_70_0.8	48.4	40.4	41.0	3600.0	23.1	0	41.0	3600.0	17.1	0
cbip_80_0.2	72.6	44.0	50.0	3600.1	50.4	0	51.4	3366.8	4.2	1
cbip_80_0.4	68.6	42.2	45.8	3600.1	54.6	0	47.2	3600.0	16.4	0
cbip_80_0.6	61.8	43.8	45.6	3600.1	40.6	0	46.0	3600.0	18.4	0
cbip_80_0.8	54.2	43.6	44.6	3600.1	25.6	0	44.2	3600.0	23.4	0
Mean	52.7	36.1	39.2	3600.0	39.0		39.8	3020.0	11.2	
Total						0				14

available computational resources. In such cases, the processes in execution were killed by the operating system due to the excessive use of memory required by the solver. For this reason, we do not provide the results for these instances.

Table 4.7: Results using the formulations for the DIMACS instances with up to 64 vertices

instance	ub	h	std			rep		
			best	time	gap	best	time	gap
johnson8-2-4	16.0	7.0	12.0	3600.0	33.3	12.0	46.8	0.0
johnson8-4-4	54.0	21.0	26.0	3600.0	107.6	29.0	3600.0	41.3
maan_a9	42.0	21.0	21.0	3600.0	100.0	21.0	0.1	0.0
hamming6-2	58.0	37.0	40.0	3600.0	10.0	40.0	3600.0	10.0
hamming6-4	23.0	9.0	13.0	3600.0	153.0	13.0	3600.0	153.0

4.4.5 BRKGA results

In this section, we discuss the results achieved using the BRKGA. The tables summarizing the results for all the benchmark instances are available in Appendix A. The plots in Figures 4.4 and 4.5 show the percentage of the instances that BRKGA found a solution that at least matches the one obtained using any of the formulations for each graph class, separated by the number of vertices and the density of the graph, respectively. The plots show that the BRKGA has more difficulty with the bipartite instances. The BRKGA at least matched all the best results for the five DIMACS instances for which the solver was able to finish the execution (those appearing in Table 4.7), finding strictly better results for two of them.

Figure 4.4 shows that, for almost all the graph classes, BRKGA achieved better performances for the instances with 50 and 60 vertices. Figure 4.5 shows that changing the density of the graph

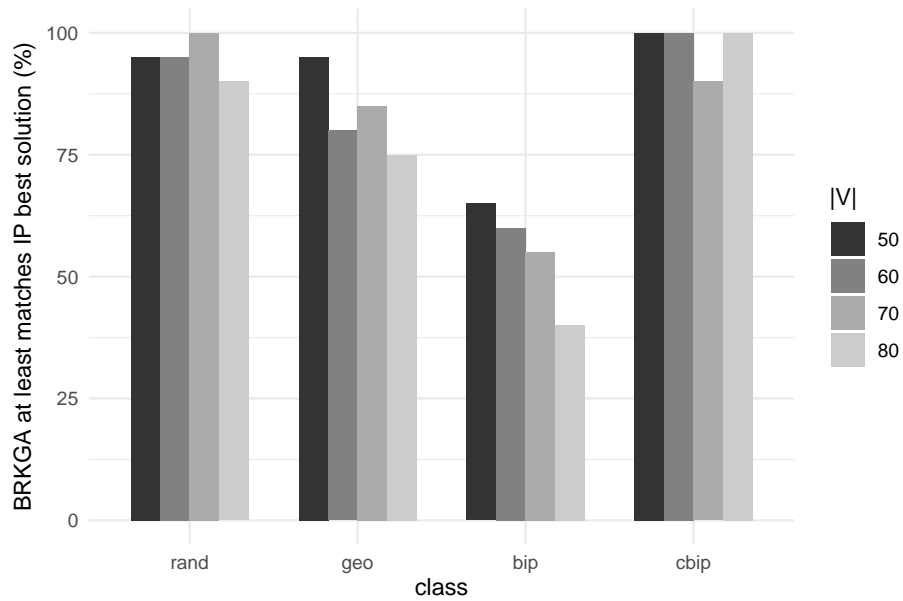


Figure 4.4: Barplot with the percentage of the instances for which the BRKGA at least matched the best solution obtained using any of the IP formulations, separated by $|V|$

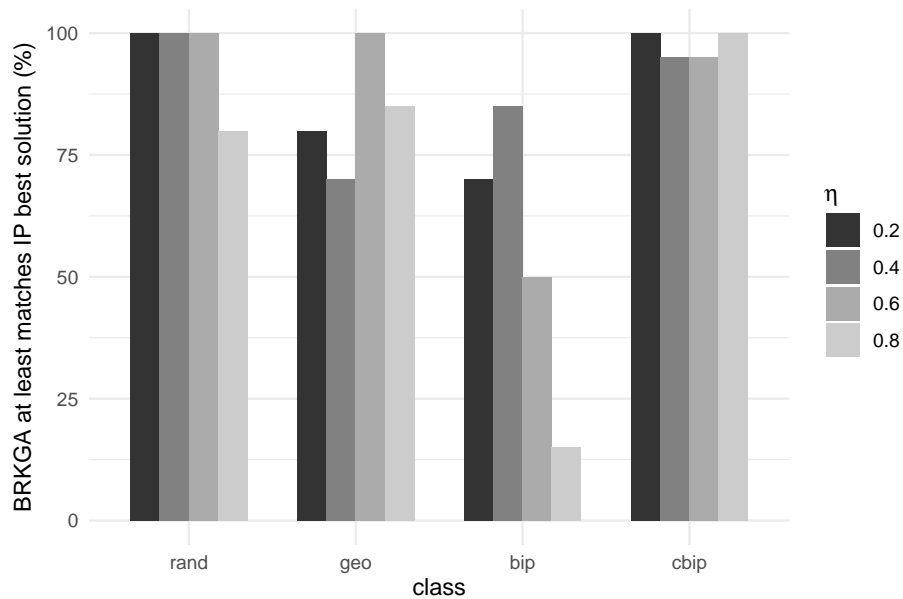


Figure 4.5: Barplot with the percentage of the instances for which the BRKGA at least matched the best solution obtained using any of the IP formulations, separated by the value of η

does not significantly affect the BRKGA's ability to find better solutions than the formulations for the random and complement of bipartite groups. This is an interesting behavior, considering the specificities of the proposed formulations that may perform better depending on the density of the graph (as it was observed for the formulation by representatives).

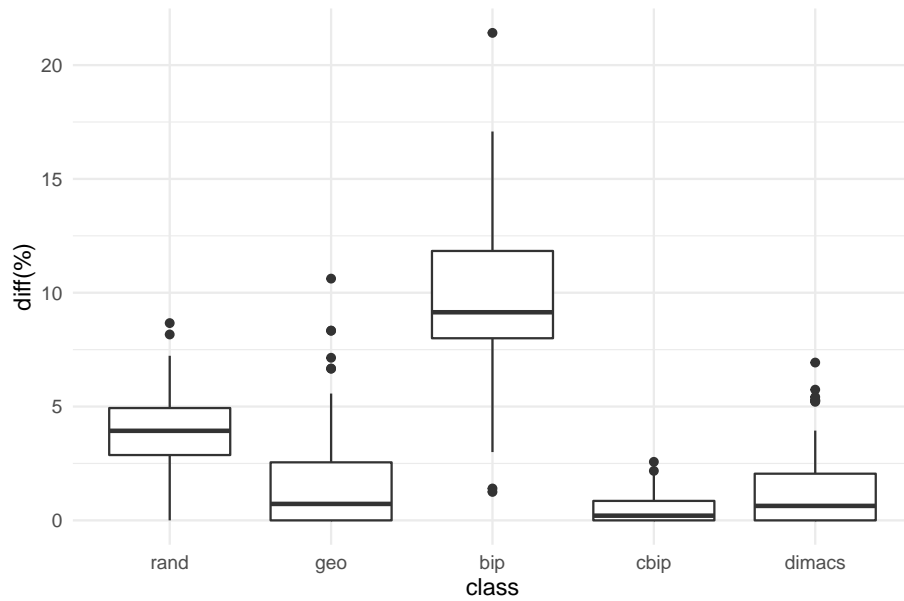


Figure 4.6: Boxplot summarizing the BRKGA deviations to the best solution considering all the executions

The plot in Figure 4.6 shows the deviations of the solutions found in all the runs of the BRKGA to the best-known solution for the corresponding instance. Figures 4.7 and 4.8 provide similar information, but separated by $|V|$ and η , respectively. It can be noticed that the mean deviations are close to zero for the complement of bipartite, geometric, and DIMACS graphs. For the random graphs, the mean deviation is slightly higher but still below 5%. This shows how the BRKGA recurrently converges to the best solutions or close to them. This pattern is only broken for bipartite instances where the standard formulation stands out probably by reducing the number of possible solutions by including the limits $\zeta(G)$ and $\Psi(G)$ in the formulation. It is noteworthy that there are very few outliers for each graph class, reinforcing the convergence consistency of the BRKGA.

It should be noticed that there is a subtle pattern. Namely, when the number of vertices increases, there is an increase in the mean deviations for almost all cases, which can be seen in Figure 4.7. The boxes in the graph also move upwards with the increase in the number of vertices for the set of random, geometric, and bipartite graphs. That is, the concentration of deviations increases with the increase in the size of the graph. There are few DIMACS instances with less than 80 vertices, thus, it is reasonable to have a high amplitude. However, it is interesting that small deviations and amplitudes are observed for the other instances with different characteristics. However, in Figure 4.8, the opposite occurs when it comes to the density of the graph. By increasing the density of the graph, the mean deviations decrease within the same graph class in almost all cases. For random and geometric graphs, there is a pattern in which when the density of the graphs increases, there is simultaneously a greater concentration of deviation values (smaller boxes) and a decrease in deviation values.

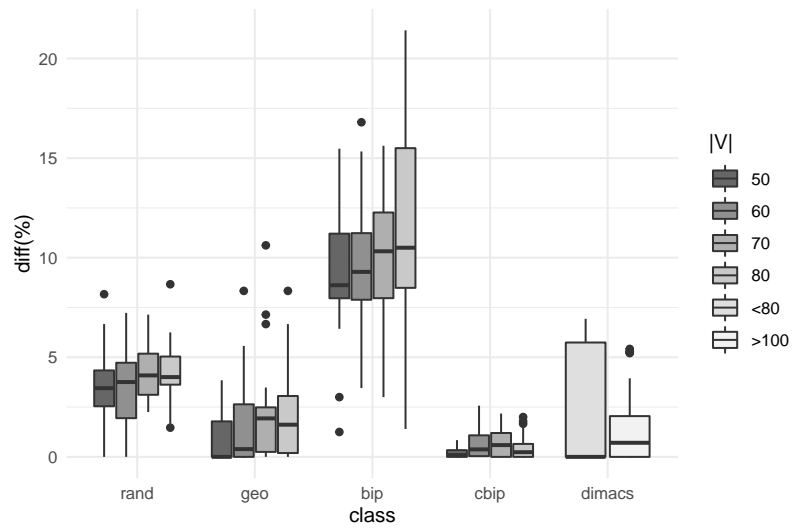


Figure 4.7: Boxplot summarizing the BRKGA deviations to the best-known solutions, separated by $|V|$

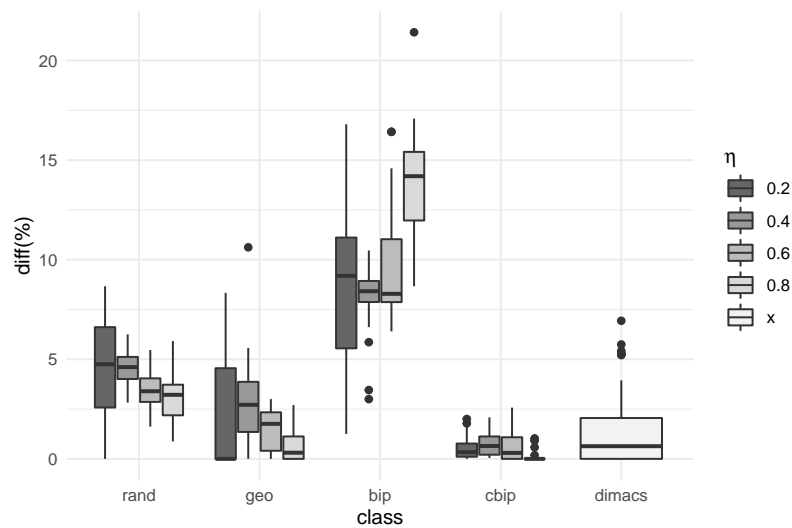


Figure 4.8: Boxplot summarizing the BRKGA deviations to the best-known solutions, separated by η

CONNECTED GRUNDY COLORING PROBLEM

This chapter focuses on the connected variant of the problem in which we propose two integer programming formulations in Section 5.1. The standard formulation incorporates the new proposed limit (Section 4.1), the *stair-factor* limit and takes into account a temporal factor in the coloring order aiming to reduce the total number of variables and constraints to be evaluated in relation for a form that doesn't take this into account. The application of the BRKGA for the problem in Section 5.2. Computational experiments with analysis of the results and a comparison between the BRKGA results of the Grundy coloring problem with its connected version in Section 5.3.

5.1 INTEGER PROGRAMMING FORMULATIONS

In this section, we formulate the connected Grundy coloring problem as an integer programs.

5.1.1 Standard formulation

In what follows, denote the set of vertices by $V = \{1, \dots, n\}$. Furthermore, let $T = \{1, \dots, |V|\}$ denote periods, representing the order in which the vertices are colored. Besides, define the sequence of available colors as $K = \{1, \dots, \Psi(G)\}$, $K_v = \{k' \in K \mid k' \leq \psi(v, \Delta(G) + 1)\}$ the sequences of possible colors to the vertex v , and $K_{vt} = \{k' \in K \mid k' \leq \min(t, \psi(v, \Delta(G) + 1))\}$ the sequence of possible colors for the vertex v in time t . Finally, define $V_k = \{v \in V \mid k \in K_v\}$ as the set of vertices that can receive the k -th color.

To formulate the connected Grundy coloring problem as an integer program, consider the decision variables:

$$z_{vkt} = \begin{cases} 1, & \text{if vertex } v \in V \text{ receives color } k \in K_{vt} \text{ in time } t \in T, \\ 0, & \text{otherwise.} \end{cases}$$

$$w_k = \begin{cases} 1, & \text{if color } k \in K \text{ is used,} \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the connected Grundy coloring problem can be cast as:

$$\max \sum_{k \in K} w_k \quad (5.1)$$

$$\sum_{\substack{t \in T, \\ t \geq k}} z_{ukt} + \sum_{\substack{t \in T, \\ t \geq k}} z_{vkt} \leq w_k, \quad \forall k \in K_v \cap K_u, uv \in E, \quad (5.2)$$

$$\sum_{t \in T} \sum_{k \in K_{vt}} z_{vkt} = 1, \quad \forall v \in V, \quad (5.3)$$

$$w_k \leq \sum_{v \in V} \sum_{\substack{t \in T, \\ t \geq k}} z_{vkt}, \quad \forall k \in K, \quad (5.4)$$

$$\sum_{t' \in \{k', \dots, t\}} z_{vk't'} \leq \sum_{\substack{u \in N(v), \\ u \in V_k}} \sum_{t' \in \{k, \dots, t-1\}} z_{ukt'}, \quad \forall v \in V, k, k' \in K_v, t \in T \setminus \{1\}, \text{ with } k < k', \quad (5.5)$$

$$\sum_{v \in V} \sum_{k \in K_{vt}} z_{vkt} = 1, \quad \forall t \in T, \quad (5.6)$$

$$\sum_{k \in K_{vt}} z_{vkt} \leq \sum_{u \in N(v)} \sum_{t'=1}^{t-1} \sum_{k \in K_{ut'}} z_{ukt'}, \quad \forall v \in V, t \in T \setminus \{1\}, \quad (5.7)$$

$$w_k \in \{0, 1\}, \quad \forall k \in K. \quad (5.8)$$

$$z_{vkt} \in \{0, 1\}, \quad \forall v \in V, t \in T, k \in K_{vt}. \quad (5.9)$$

Constraints (5.2) ensure that adjacent vertices do not receive the same color. Constraints (5.3) guarantee that each vertex receives exactly one color in a single period. Constraints (5.4) determine that w_k is only set to one if color k is used. Constraints (5.5) guarantee the Grundy property. Notice that they imply that if a vertex $v \in V$ receives a color in periods one up to t , all the colors with lower index need to be used in the neighborhood of v in periods one up to $t - 1$. Constraints (5.6) establish that a single vertex receives a color in each period. Constraints (5.7) ensure that the coloring is connected. Constraints (5.9) define integrality requirements on the variables.

5.1.2 Formulation by representatives

In the following, we describe the formulation by representatives for the connected Grundy coloring problem. Consider the following decision variables:

$$Z_{vut} = \begin{cases} 1, & \text{if vertex } u \in V \text{ is represented by vertex } v \in \bar{N}[u] \text{ in time } t \in T, \text{ for } v \leq u, \\ 0, & \text{otherwise;} \end{cases}$$

$$y_{vu} = \begin{cases} 1, & \text{if vertices } v, u \in V \text{ are representatives and the color of } v \text{ precedes that of } u, \text{ for } v \neq u, \\ 0, & \text{otherwise.} \end{cases}$$

An IP formulation by representatives for the connected Grundy coloring problem can be defined as:

$$\max \sum_{v \in V} \sum_{t \in T} Z_{vvt} \quad (5.10)$$

$$\sum_{t \in T} Z_{uvt} + \sum_{t \in T} Z_{uwt} \leq \sum_{t \in T} Z_{uut}, \quad \forall u \in V, v, w \in \bar{N}[u], \text{ s.t. } vw \in E \text{ and } u \leq v < w, \quad (5.11)$$

$$\sum_{t \in T} Z_{uvt} \leq \sum_{t \in T} Z_{uut}, \quad \forall u \in V, v \in \bar{N}(u), \text{ s.t. } N(v) \cap \bar{N}(u) = \emptyset \text{ and } u < v, \quad (5.12)$$

$$\sum_{\substack{v \in \bar{N}[u], \\ v \leq u}} \sum_{t \in T} Z_{vut} = 1, \quad \forall u \in V, \quad (5.13)$$

$$\sum_{t' \in \{1, \dots, t\}} Z_{uv't'} \leq \sum_{\substack{w \in N(v) \cap \bar{N}[p], \\ p \leq w}} \sum_{t' \in \{1, \dots, t-1\}} Z_{pwt'} + 1 - y_{pu}, \quad \forall u, p \in V, v \in \bar{N}[u],$$

$$t \in T \setminus \{1\}, \text{ s.t. } p \neq u \text{ and } u \leq v, \quad (5.14)$$

$$\sum_{\substack{u \in \bar{N}[v], \\ u \leq v}} Z_{uvt} \leq \sum_{v' \in N(v)} \sum_{\substack{u \in \bar{N}[v'], \\ u \leq v'}} \sum_{t=1}^{t-1} Z_{uv't}, \quad \forall v \in V, t \in T \setminus \{1\}, \quad (5.15)$$

$$y_{vu} + y_{uv} \geq \sum_{t \in T} Z_{uut} + \sum_{t \in T} Z_{vvt} - 1, \quad \forall u, v \in V, \text{ s.t. } u < v, \quad (5.16)$$

$$y_{uv} + y_{vu} \leq \sum_{t \in T} Z_{uut}, \quad \forall u, v \in V, \text{ s.t. } u \neq v, \quad (5.17)$$

$$Z_{uvt} \in \{0, 1\}, \quad \forall u \in V, v \in \bar{N}[u], t \in T, \text{ s.t. } u \leq v, \quad (5.18)$$

$$y_{uv} \in \{0, 1\}, \quad \forall u, v \in V, \text{ s.t. } u \neq v. \quad (5.19)$$

The objective function (5.10) maximizes the number of representative vertices. Constraints (5.11) ensure that adjacent vertices do not receive the same color. Constraints (5.12) indicate that a vertex can only represent another one if the former is a representative. Constraints (5.13) guarantee that every vertex receives a color in a single period. Constraints (5.14) imply the Grundy property. They establish that if $p, u \in V$ are representatives, p precedes u , and u represents $v \in V$, then p has to represent one of the neighbors of v before u can represent v . Constraints (5.15) ensure that the coloring is connected. Constraints (5.16)-(5.17) ensure an order between two vertices if and only if they are both representatives. Constraints (5.18)-(5.19) determine the integrality of the variables.

5.2 BIASED RANDOM-KEY GENETIC ALGORITHM

This section presents the encoding and decoder used in BRKGA for the connected Grundy coloring problem.

5.2.1 Solution encoding

The solution is encoded in the same way as presented in section 4.3.1, a vector x of random keys with length $\Upsilon = |V|$, with the i -th key representing the i -th vertex in V .

5.2.2 Decoder

The decoder defines the vertex coloring order considering the i -th key as the priority of the i -th vertex to be selected while respecting the connectivity constraint of the problem. Algorithm 2 describes the decoder, which receives a graph G and a vector of random keys c as parameters. The algorithm utilizes a priority queue Q based on the random keys c_v . The auxiliary procedure $\text{Enqueue}(Q, v)$ inserts the vertex v into the priority queue Q according to its key c_v . The $\text{Dequeue}(Q)$ method removes and returns the element with the highest key from Q .

In Algorithm 2, the line 1 initializes the color of all vertices to -1. The vertex with the highest key is added to the priority queue Q (lines 2-3). The loop from lines 4-10 repeats the process of selecting the vertex with the highest key from the priority queue (line 5) and coloring it with the lowest possible color following the first-fit algorithm with the auxiliary procedure Color-Vertex (line 6). Color-Vertex sets the color of a vertex as the lowest-index color that has not yet been used for any of its neighbors. Subsequently, its neighbors that have not yet been added to the priority queue are inserted into Q (lines 7-10). Line 11 returns the vector of colors and the total number of used colors (fitness).

Algorithm 2: Decoder-connected-Grundy (G, c)

```

1 colors ← {-1, ..., -1};
2 v ← vertex with highest key value cv;
3 Enqueue(Q, v);
4 while Q ≠ ∅ do
5   v ← Dequeue(Q);
6   colors[v] ← Color-Vertex(G, v, colors);
7   foreach u in N(v) do
8     if colors[u] = -1 then
9       colors[u] = 0;
10      Enqueue(Q, v);
11 return colors, maxv∈V colors[v];

```

Proposition 3. *Algorithm 2 can be implemented to run in $O(|V| \log |V| + |E|)$.*

Proof. First, note that each vertex $v \in V$ enters and exits the priority queue Q exactly once. Therefore, all operations related to Q are performed in $O(|V| \log |V|)$. The loop from lines 4-10 involves traversing the graph's adjacency list to determine the color and enqueue the neighbors of each vertex. Thus, excluding the operations related to the priority queue, which have already been accounted for in the computational cost, the remaining operations can be performed in $O(|V| + |E|)$. Consequently, Algorithm 2 can be implemented to run in $O(|V| \log |V| + |E|)$. ■

5.3 COMPUTATIONAL EXPERIMENTS

All the experiments were executed on a machine running Ubuntu x86-64 GNU/Linux, with an Intel Core i7-10700 Octa-Core 2.90 Ghz processor and 16Gb of RAM. The formulations

were implemented in Julia and solved with Gurobi 10.0.1. The BRKGA was developed in C++ using the BRKGA API (Toso & Resende, 2015; Toso, 2018).

5.3.1 Benchmark instances

The tests were carried out with the same set of instances used in 4.4.1, except for bipartite graphs, since the connected problem is trivial in the case of bipartite graphs. It is necessary to point out that not all instances of this set are connected. Then these instances were connected using an algorithm that tries not to change the connected Grundy number of the original instance. The vertex with the highest degree with the lowest index of each connected component is selected and a path is created between these vertices from the lowest index to the one with the highest index between them, making the instance connected.

5.3.2 Tested approaches and parameter settings

The following approaches were considered in the computational experiments:

- the standard formulation described in Section 5.1.1 (std);
- the formulation by representatives detailed in Section 5.1.2 (rep);
- the BRKGA explained in Section 5.2 (BRKGA);
- the greedy criterion *connected minimum-degree first* (CMinDF), that defines a connected coloring sequence (v_1, \dots, v_n) prioritizing the vertices with lower degree as described in 4.4.2, but using a adaptative BFS-based approach.

Additionally, connected versions of some widely-used and well-established greedy criteria for the coloring problem (that aims to minimize the number of colors) were used as baselines:

- *connected maximum-degree first* (CMDF, 2.2)
- *DSatur* (section 2.2).

The solver was set with the default configurations and a single thread. A time limit of 3600 seconds (1 hour) was given for each formulation to solve each of the instances. A warm start (i.e., initial feasible solution) was given to the formulations, provided by the best solution achieved using any of the greedy heuristics (CMinDF, CMDF, DSatur).

The settings for the BRKGA were defined as follows. All executions were performed using a single thread. A time limit of 300 seconds (five minutes) was defined as the stopping criterion instead of the number of generations. This choice makes it easier to control the total runtime and benefits efficient decoders. The parameters were defined based on preliminary tests considering a subset of 28 instances and the following possible settings: $p = \{1 \times |V|, 2 \times |V|, 3 \times |V|\}$, $p_e = \{5\%, 15\%, 30\%\}$, $p_m = \{5\%, 10\%, 30\%\}$ e $\rho_e = \{60\%, 70\%, 90\%\}$. In total, 81 configurations were tested for which the BRKGA was run with five different seeds. The considered values were based on how the metaheuristic converges over the generations by varying each of the parameters (Gonçalves & Resende, 2011). Finally, the configuration selected for the overall experiments was $(p, p_e, p_m, \rho_e) = (3 \times |V|, 15\%, 10\%, 60\%)$.

5.3.3 IP formulations results

The connected Grundy coloring problem proved challenging for both models. The first tests with the model using the default configurations and single thread did not generate good results, only in 12% of the instances it managed to find a solution, in the remaining cases the process was killed due to memory overflow. The 12% were in general the smallest instances of 50 vertices and the instances *johnson8-2-4* and *mann_a9*, in addition to few vertices these instances are also either extremely dense or extremely sparse. Only 5 geometric and 5 random instances with 60 vertices and 1 geometric instance with 70 vertices managed to resolve the first node, with a density of less than 0.2. In other words, only in scenarios where each formulation has an advantage (graph density at the extremes) was it possible to obtain any results, even then only for the smallest instances overall.

Other attempts were made using multiple threads and changing the initial root relaxation method, but not showing any significant signs of improvement. Within these 12% of instances that had an initial solution found, only a quarter of them did the solver manage to solve more than one node, and within this subset it was tested increasing the timeout to 10.800 seconds (3h), but this led to a few more processes be killed by memory overflow.

5.3.4 BRKGA results

In this section, we discuss the results achieved using the BRKGA. Tables 5.1-5.4 summarize the results of the experiments with the BRKGA. In Tables 5.1-5.3, the first column represents an instance group, so all the values in each row correspond to the average over five instances. The next three columns provide the average mean, maximum, and *time to best* (ttb) considering the 50 independent runs for each of the instances. The fifth and sixth columns provide the average best result found by CMinDF and its average deviation from the best solution achieved by the BRKGA. The last two columns show the average best result achieved with any of the other greedy criteria (column best) and its average deviation from the best solution achieved by the BRKGA. The columns diff are defined as $diff = \frac{100 * (BRKGA_{max} - best)}{BRKGA_{max}}$. Table 5.4 follows the same structure, but each row represents a single instance, implying that the values are not averaged over five instances.

The results for the random instances (table 5.1) show that BRKGA achieved a diff of at least 23% for all groups compared to CMinDF, and even better compared to other heuristics with a *diff* of at least 35%, taking less than 100 seconds to converge in all cases and less than a minute in 37.5% of the groups.

Table 5.2 shows that BRKGA had a different performance in geometric instances. In two groups it took less than 0.1 to converge, which may indicate that it was stucked in a local optimum. The same can be said for groups *geo50_0.8* and *geo70_0.2*, which converged in less than 7 seconds and with a standard deviation equal to 0. At best, it may have converged to the global optimum, but there is no way to confirm this at the moment. Even so, it was possible to achieve a percentage difference of at least $\pm 15\%$ in 75% of the groups compared to CMinDF. A *diff* of at least 17% for all cases compared to other heuristics, with a $diff > 30\%$ in 43.75% of the groups.

Table 5.3 shows the results for the complements of the bipartite graph, which is a set of

group	BRKGA			CMinDF		heuristic	
	max	mean	tbt	best	diff(%)	best	diff(%)
rand_50_0.2	11.4	11.1	47.3	7.8	31.6	6.2	45.6
rand_50_0.4	17.0	16.1	53.9	11.6	31.8	9.8	42.4
rand_50_0.6	23.4	22.4	77.8	16.4	29.9	14.2	39.3
rand_50_0.8	31.0	30.6	74.6	23.8	23.2	20.0	35.5
rand_60_0.2	12.4	12.0	51.6	8.4	32.3	7.2	41.9
rand_60_0.4	19.6	18.6	57.8	13.2	32.7	11.2	42.9
rand_60_0.6	26.4	25.4	80.8	18.8	28.8	16.6	37.1
rand_60_0.8	36.0	35.3	74.1	27.6	23.3	23.2	35.6
rand_70_0.2	13.6	13.1	35.7	10.0	26.5	7.8	42.6
rand_70_0.4	21.6	20.5	72.4	15.2	29.6	13.0	39.8
rand_70_0.6	29.0	27.7	84.9	21.2	26.9	17.6	39.3
rand_70_0.8	40.2	39.0	84.0	30.8	23.4	25.6	36.3
rand_80_0.2	14.8	14.1	39.8	10.0	32.4	8.4	43.2
rand_80_0.4	23.2	22.0	74.8	16.6	28.4	13.4	42.2
rand_80_0.6	32.4	30.9	86.0	23.4	27.8	20.2	37.7
rand_80_0.8	44.0	42.7	93.8	32.0	27.3	28.2	35.9

Table 5.1: BRKGA results for the random graphs for the connected Grundy coloring problem

very dense graphs. The first thing to point out is that in all cases it took less than 21 seconds on average, which may indicate difficulty in finding the global optimum, having a worse result than the heuristics in the group *cbip70_0.8*. However, if you compare it with the table A.4 of the results of the formulations for the less restricted version of the problem, you can note that in most cases it reaches the same value including the optimal ones, which shows that in general BRKGA performs very well for this type of instance.

Table 5.4 shows the results for the set of DIMACS instances, which, as already mentioned, is a very heterogeneous set. 15 of 42 instances converged in less than 20 seconds, which may indicate a difficulty in finding a local optimum, however, if we look at *mann_a9* and *johnson8-2-4* as examples, we see that the value found was the same as the optimum proven by the models for the Grundy number. There is no very significant difference in the other 13 cases compared to the model or BRKGA for the unconnected version, which indicates that they are good quality solutions that converged quickly to solutions close to the global optimum. Furthermore, it managed to establish a percentage difference of at least 15% in 76.1% of instances compared to *CMinDF* and in 83.3% compared to other heuristics, although in one case (*mann_a9*) there was no difference and in another 9 the difference was less than 10%.

group	BRKGA			CMinDF		heuristic	
	max	mean	ttb	best	diff(%)	best	diff(%)
geo_50_0.2	8.2	8.2	0.0	7.0	14.6	6.8	17.1
geo_50_0.4	22.2	22.0	39.9	17.8	19.8	15.4	30.6
geo_50_0.6	31.2	30.8	35.6	26.0	16.7	23.6	24.4
geo_50_0.8	37.8	37.8	6.8	35.0	7.4	30.4	19.6
geo_60_0.2	9.8	9.7	17.6	7.4	24.5	6.8	30.6
geo_60_0.4	26.6	26.0	51.3	18.6	30.1	16.4	38.3
geo_60_0.6	38.0	37.3	42.6	32.8	13.7	27.6	27.4
geo_60_0.8	47.4	47.3	17.4	44.4	6.3	37.8	20.3
geo_70_0.2	12.0	11.9	5.7	9.4	21.7	8.2	31.7
geo_70_0.4	27.6	26.7	47.0	21.4	22.5	18.0	34.8
geo_70_0.6	42.0	41.0	62.9	35.6	15.2	29.2	30.5
geo_70_0.8	54.2	53.6	30.8	49.2	9.2	44.6	17.7
geo_80_0.2	12.4	12.4	0.1	9.2	25.4	8.8	29.4
geo_80_0.4	32.2	31.1	62.7	23.6	26.7	20.6	36.0
geo_80_0.6	47.8	47.2	50.5	39.6	17.2	33.8	29.3
geo_80_0.8	62.4	62.3	20.0	57.6	7.7	50.6	18.9

Table 5.2: BRKGA results for the geometric graphs for the connected Grundy coloring problem

5.3.5 Comparison between BRKGA solutions to the problems

In this section, we will report the deviations between the solutions obtained by BRKGA for the two variants of the problem: the Grundy coloring problem and the connected Grundy coloring problem. For this, the best values found by BRKGA in the 50 executions for each of the instances are considered. Denote by max_{PCG} and max_{PCGC} the largest values found by BRKGA for a given instance of the Grundy coloring problem and the connected Grundy coloring problem, respectively. The deviation for an instance is given by $100 - 100 * \frac{max_{PCGC}}{max_{PCG}}$.

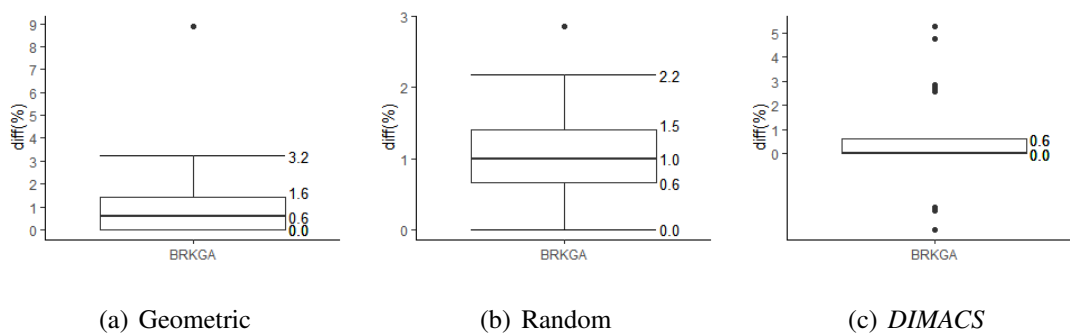


Figure 5.1: Comparative boxplots of the deviation between BRKGA for Grundy coloring problem and connected Grundy coloring problem

Figure 5.1 shows the *boxplots* with the deviations for each of the sets of instances. It can be

group	BRKGA			CMinDF		heuristic	
	max	mean	ttb	best	diff(%)	best	diff(%)
cbip_50_0.2	34.6	34.5	13.7	29.6	14.5	27.0	22.0
cbip_50_0.4	31.2	31.1	5.5	27.2	13.0	26.4	15.5
cbip_50_0.6	29.0	29.0	1.8	26.6	8.3	26.2	9.7
cbip_50_0.8	30.4	30.4	4.3	29.6	2.6	29.6	2.6
cbip_60_0.2	41.0	40.6	9.8	35.0	14.6	32.0	22.0
cbip_60_0.4	38.0	37.7	4.0	34.4	9.5	33.8	11.1
cbip_60_0.6	36.0	35.8	2.2	33.0	8.3	32.4	10.0
cbip_60_0.8	35.0	35.0	13.1	33.0	5.7	32.8	6.3
cbip_70_0.2	47.2	46.8	11.4	40.4	14.4	38.6	18.2
cbip_70_0.4	44.0	43.4	5.7	41.4	5.9	41.4	5.9
cbip_70_0.6	41.8	41.7	16.9	39.8	4.8	39.8	4.8
cbip_70_0.8	39.0	38.8	2.4	40.4	-3.6	40.4	-3.6
cbip_80_0.2	51.6	51.1	7.3	44.8	13.2	40.2	22.1
cbip_80_0.4	47.5	47.1	13.5	42.4	10.7	41.2	13.3
cbip_80_0.6	46.7	46.7	0.3	43.8	6.1	43.8	6.1
cbip_80_0.8	44.8	44.7	20.9	43.6	2.7	43.6	2.7

Table 5.3: BRKGA results for the complements of bipartite graphs for the connected Grundy coloring problem

noted that the deviations for the 3 groups of instances are small, mostly close to 1%. The *boxplot* 5.1(a) shows that for these instances you get almost the same result when trying to solve both problems using this approach. 5.1(b) shows that the additional restrictiveness already makes it a little more difficult for instances of random graphs, which is why it presents larger deviations than in geometric ones in most cases. And finally, 5.1(c) is the one that presented the most irregular results, having a large number of *outliers* and with a small interquartile range close to the 0.0%, indicating that in most cases the two reached the same maximum. The subfigure 5.1(c) also indicates that BRKGA certainly did not find the optimal solutions for some instances, given that in certain cases solutions for the related variant (which is more restricted) had larger values than those for the problem of the original Grundy coloring.

instance	BRKGA			CMinDF		heuristic	
	max	mean	ttb	best	diff(%)	best	diff(%)
brock200_2	46.0	44.2	168.9	37.0	19.6	33.0	28.3
c-fat200-1	18.0	18.0	0.2	16.0	11.1	14.0	22.2
c-fat200-2	34.0	34.0	1.1	24.0	29.4	24.0	29.4
c-fat200-5	87.0	86.6	3.3	85.0	2.3	72.0	17.2
c-fat500-1	20.0	20.0	0.8	14.0	30.0	17.0	15.0
c-fat500-2	38.0	38.0	17.7	26.0	31.6	32.0	15.8
c125.9	77.0	76.0	73.4	62.0	19.5	54.0	29.9
dsjc125.1	13.0	12.4	13.9	8.0	38.5	7.0	46.2
dsjc125.5	36.0	35.4	61.1	28.0	22.2	23.0	36.1
dsjc125.9	77.0	75.6	48.1	62.0	19.5	57.0	26.0
dsjc250.1	18.0	18.0	71.6	14.0	22.2	11.0	38.9
dsjr500.1	20.0	20.0	9.9	16.0	20.0	14.0	30.0
fpsol2.i.2	40.0	39.9	35.6	34.0	15.0	30.0	25.0
fpsol2.i.3	40.0	39.8	33.7	34.0	15.0	30.0	25.0
hamming6-2	40.0	40.0	0.2	37.0	7.5	32.0	20.0
hamming6-4	15.0	13.7	58.4	9.0	40.0	8.0	46.7
hamming8-2	160.0	160.0	172.1	135.0	15.6	128.0	20.0
hamming8-4	39.0	38.2	52.6	27.0	30.8	33.0	15.4
johnson8-2-4	12.0	12.0	4.6	6.0	50.0	8.0	33.3
johnson8-4-4	30.0	28.9	120.3	19.0	36.7	21.0	30.0
keller4	48.0	44.4	125.5	28.0	41.7	39.0	18.8
le450_15a	32.0	31.2	168.6	26.0	18.8	19.0	40.6
le450_15b	33.0	31.5	155.6	25.0	24.2	19.0	42.4
le450_25a	44.0	43.2	106.0	35.0	20.5	25.0	43.2
le450_25b	44.0	42.4	128.7	33.0	25.0	25.0	43.2
le450_5a	18.0	18.0	91.6	14.0	22.2	11.0	38.9
le450_5b	18.0	18.0	113.4	14.0	22.2	11.0	38.9
le450_5c	23.0	22.0	134.8	18.0	21.7	9.0	60.9
le450_5d	22.0	21.8	154.0	17.0	22.7	12.0	45.5
mann_a9	21.0	21.0	0.0	20.0	4.8	21.0	0.0
mulsol.i.1	52.0	52.0	5.5	50.0	3.8	49.0	5.8
mulsol.i.2	34.0	33.4	34.2	32.0	5.9	31.0	8.8
mulsol.i.3	34.0	33.4	47.0	32.0	5.9	31.0	8.8
mulsol.i.4	34.0	33.4	39.6	32.0	5.9	31.0	8.8
mulsol.i.5	34.0	34.0	0.4	31.0	8.8	31.0	8.8
r125.1	7.0	7.0	0.0	5.0	28.5	5.0	28.5
r125.1c	62.0	62.0	0.9	51.0	17.7	47.0	24.2
r125.5	65.0	63.7	41.1	52.0	20.0	40.0	38.5
r250.1	12.0	11.1	10.5	10.0	16.7	8.0	33.3
zeroin.i.1	53.0	52.8	89.3	50.0	5.7	49.0	7.5
zeroin.i.2	36.0	34.9	69.6	31.0	13.9	30.0	16.7
zeroin.i.3	37.0	35.1	63.6	31.0	16.2	30.0	18.9

Table 5.4: BRKGA results for the *DIMACS* instances for the connected Grundy coloring problem

CONCLUSION

This chapter summarizes the main contributions of this master thesis considering the Grundy coloring problem and its connected version, whose optimal solution provides the Grundy chromatic number and the connected Grundy chromatic number, respectively. We proposed a new combinatorial upper bound that is valid for both problems, and two integer programming formulations, and a biased random-key genetic algorithm (BRKGA) for each problem. To our knowledge, the integer programming formulations and the BRKGA stand out as the first optimization methods to tackle the problems for general graphs.

The computational experiments showed that the new combinatorial upper bound improves over a well-established bound available in the literature for 19.0% of the tested instances. The experiments also indicated that the formulation by representatives achieved an overall better performance than the standard formulation. The formulation by representatives achieved better results for the denser instances, while the standard formulation performed slightly better for the sparser ones. The Grundy coloring problem proved to be challenging for the formulations, even for the instances with 50 vertices. The formulations, despite being able to improve the results from the initial solution provided, failed to prove optimality for most of the instances within the time limit of 3600 seconds. Furthermore, they ended up with an optimality *gap* greater than 50% in 35% of the instances for the standard formulation and in 44% of the cases for the formulation by representatives.

On the other hand, it was observed that the formulations proposed for the connected problem represent a computationally impractical approach in their current state. This observation further underscores how the mere addition of a connectivity constraint significantly increases the complexity of the problem.

The experiments performed with the BRKGA indicate that this metaheuristic can find good-quality solutions within short computational times, achieving robustness regarding the variations between the qualities of the solutions obtained in multiple executions. Furthermore, the BRKGA found systematically better solutions than a proposed greedy algorithm to maximize the number of colors in a *first-fit* coloring, and in the great majority of the cases, solutions that at least match those obtained by the formulations. The same happened with applying the BRKGA to obtain $\Gamma_c(G)$, emphasizing that there was almost no significant difference between the values obtained

for both problems, indicating that it is possible for these cases to use even the results of the formulations or BRKGA for the Grundy number problem as a reference for a tight upper bound for the optimum over the connected problem.

Preliminary results of this master's thesis were presented at the the Brazilian Operations Research Symposium in 2023 (Carvalho, Melo, Santos, Toso, & Resende, 2023a, 2023b).

6.1 FUTURE WORKS

Ideas for future work on the Grundy coloring problem and its connected version include exploring the application of other metaheuristics, which can be used in conjunction with *tabu search* to aid in the search and try to avoid the search having problems with many visits to symmetric solutions.

Another possibility is the study of valid inequalities to check whether their incorporation would allow improving the bounds in the Grundy coloring problem. The computational experiments so far indicate that it would be better to carry out the work in the connected version with metaheuristics. Something that still needs to be analyzed, but that could be a way of simplifying in some cases, are the implications of cutting edges in a graph as a possibility of decomposing the graph to solve each component separately and have an approximate value to the connected Grundy number.

REFERENCES

- Aboulker, P., Bonnet, É., Kim, E. J., & Sikora, F. (2023). Grundy coloring and friends, half-graphs, bicliques. *Algorithmica*, 85(1), 1–28.
- Al-Omari, H., & Sabri, K. E. (2006). New graph coloring algorithms. *American Journal of Mathematics and Statistics*, 2(4), 739–741.
- Avanthay, C., Hertz, A., & Zufferey, N. (2003). A variable neighborhood search for graph coloring. *European Journal of Operational Research*, 151(2), 379–388.
- Babaei, H., Karimpour, J., & Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86, 43–59.
- Bahiense, L., Frota, Y. A. M., Noronha, T. F., & Ribeiro, C. C. (2014). A branch-and-cut algorithm for the equitable coloring problem using a formulation by representatives. *Discrete Applied Mathematics*, 164, 34–46.
- Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA journal on computing*, 6(2), 154–160.
- Benevides, F., Campos, V., Dourado, M., Griffiths, S., Morris, R., Sampaio, L., & Silva, A. (2014). Connected greedy colourings. In *Latin American Symposium on Theoretical Informatics* (pp. 433–441).
- Berge, C. (1973). *Graphs and hypergraphs*. North-Holland Publishing Co.
- Bonamy, M., Groenland, C., Muller, C., Narboni, J., Pekárek, J., & Wesolek, A. (2021). A note on connected greedy edge colouring. *Discrete Applied Mathematics*, 304, 129–136.
- Bonnet, É., Foucaud, F., Kim, E. J., & Sikora, F. (2018). Complexity of Grundy coloring and its variants. *Discrete Applied Mathematics*, 243, 99–114.
- Brélaž, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM*, 22(4), 251–256.
- Burke, E. K., McCollum, B., Meisels, A., Petrovic, S., & Qu, R. (2007). A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research*, 176(1), 177–192.
- Campêlo, M., Campos, V., & Corrêa, R. (2005). On the asymmetric representatives formulation for the vertex coloring problem. *Electronic Notes in Discrete Mathematics*, 19, 337–343.
- Campêlo, M., & Severín, D. (2021). An integer programming approach for solving a generalized version of the Grundy domination number. *Discrete Applied Mathematics*, 301, 26–48.
- Carvalho, M., Melo, R., Santos, M. C., Toso, R. F., & Resende, M. G. (2023a). Algoritmos genéticos de chaves aleatórias enviesadas para o problema da coloração de Grundy. In *Proceedings of the LV Brazilian Operations Research Symposium*. São José dos Campos: Galoá.
- Carvalho, M., Melo, R., Santos, M. C., Toso, R. F., & Resende, M. G. (2023b). Formulações de programação inteira para o problema da coloração de Grundy. In *Proceedings of the LV Brazilian Operations Research Symposium*. São José dos Campos: Galoá.

- Chow, F. C., & Hennessy, J. L. (1990). The priority-based coloring approach to register allocation. *ACM Transactions on Programming Languages and Systems*, 12(4), 501–536.
- Christen, C. A., & Selkow, S. M. (1979). Some perfect coloring properties of graphs. *Journal of Combinatorial Theory, Series B*, 27(1), 49–59.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms, third edition* (3rd ed.). The MIT Press.
- Corrêa, R. C., Campêlo, M., & Frota, Y. A. M. (2004). Cliques, holes and the vertex coloring polytope. *Information Processing Letters*, 89, 159–164.
- de Freitas, R., Dias, B., Maculan, N., & Szwarcfiter, J. (2021). On distance graph coloring problems. *International Transactions in Operational Research*, 28(3), 1213–1241.
- de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2), 151–162.
- Dias, B., de Freitas, R., Maculan, N., & Michelon, P. (2021). Integer and constraint programming approaches for providing optimality to the bandwidth multicoloring problem. *RAIRO: Recherche Opérationnelle*, 55, S1949–S1967.
- Effantin, B., & Kheddouci, H. (2007). Grundy number of graphs. *Discussiones Mathematicae Graph Theory*, 27(1), 5–18.
- Ehmsen, M. R., Favrholt, L. M., Kohrt, J. S., & Mihai, R. (2010). Comparing first-fit and next-fit for online edge coloring. *Theoretical computer science*, 411(16-18), 1734–1741.
- Erdős, P., Hare, W., Hedetniemi, S. T., & Laskar, R. (1987). On the equality of the Grundy and chromatic numbers of a graph. *Journal of Graph Theory*, 11(2), 157–159.
- Frota, Y., Maculan, N., Noronha, T. F., & Ribeiro, C. C. (2010). A branch-and-cut algorithm for partition coloring. *Networks: An International Journal*, 55, 194–204.
- Furini, F., Malaguti, E., & Santini, A. (2018). An exact algorithm for the partition coloring problem. *Computers & Operations Research*, 92, 170–181.
- Gamache, M., Hertz, A., & Ouellet, J. O. (2007). A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding. *Computers & Operations Research*, 34(8), 2384–2395.
- Gonçalves, J. F., & Resende, M. G. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5), 487–525.
- Goyal, N., & Vishwanathan, S. (1997). NP-completeness of undirected Grundy numbering and related problems. *Manuscript, Bombay*.
- Grundy, P. M. (1939). Mathematics and games. *Eureka*, 2, 6–9.
- Gyárfás, A., & Lehel, J. (1988). On-line and first fit colorings of graphs. *Journal of Graph theory*, 12(2), 217–227.
- Havet, F., & Sampaio, L. (2013). On the Grundy and b -chromatic numbers of a graph. *Algorithmica*, 65(4), 885–899.
- He, Y., Gao, C., Sang, N., Qu, Z., & Han, J. (2017). Graph coloring based surveillance video synopsis. *Neurocomputing*, 225, 64–79.
- Hedetniemi, S. M., Hedetniemi, S. T., & Beyer, T. (1982). A linear algorithm for the Grundy (coloring) number of a tree. *Congr. Numer*, 36, 351–363.
- Janczewski, R., Kubale, M., Manuszewski, K., & Piwakowski, K. (2001). The smallest hard-to-color graph for algorithm DSATUR. *Discrete Mathematics*, 236(1), 151–165.

- Jovanović, P., Pavlović, N., Belošević, I., & Milinković, S. (2020). Graph coloring-based approach for railway station design analysis and capacity determination. *European Journal of Operational Research*, 287(1), 348–360.
- Leighton, F. T. (1979). A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, 84(6), 489–506.
- Lü, Z., & Hao, J.-K. (2010). A memetic algorithm for graph coloring. *European Journal of Operational Research*, 203(1), 241–250.
- Marzo, R. G., Melo, R. A., Ribeiro, C. C., & Santos, M. C. (2022). New formulations and branch-and-cut procedures for the longest induced path problem. *Computers & Operations Research*, 139, 105627.
- Masih, Z., & Zaker, M. (2021). On Grundy and b -chromatic number of some families of graphs: A comparative study. *Graphs and Combinatorics*, 37(2), 605–620.
- Masih, Z., & Zaker, M. (2022). Some comparative results concerning the Grundy and b -chromatic number of graphs. *Discrete Applied Mathematics*, 306, 1–6.
- Matula, D. W., & Beck, L. L. (1983). Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, 30(3), 417–427.
- Melo, R. A., Queiroz, M. F., & Santos, M. C. (2021). A matheuristic approach for the b -coloring problem using integer programming and a multi-start multi-greedy randomized metaheuristic. *European Journal of Operational Research*, 295(1), 66–81.
- Melo, R. A., & Ribeiro, C. C. (2015). Improved solutions for the freight consolidation and containerization problem using aggregation and symmetry breaking. *Computers & Industrial Engineering*, 85, 402–413.
- Melo, R. A., & Ribeiro, C. C. (2022). Maximum weighted induced forests and trees: New formulations and a computational comparative review. *International Transactions in Operational Research*, 29(4), 2263–2287.
- Melo, R. A., & Ribeiro, C. C. (2023). MIP formulations for induced graph optimization problems: a tutorial. *International Transactions in Operational Research*, 30(6), 3159–3200.
- Melo, R. A., Ribeiro, C. C., & Riveaux, J. A. (2022). The minimum quasi-clique partitioning problem: Complexity, formulations, and a computational study. *Information Sciences*, 612, 655–674.
- Melo, R. A., Samer, P., & Urrutia, S. (2016). An effective decomposition approach and heuristics to generate spanning trees with a small number of branch vertices. *Computational Optimization and Applications*, 65(3), 821–844.
- Moalic, L., & Gondran, A. (2018). Variations on memetic algorithms for graph coloring problems. *Journal of Heuristics*, 24(1), 1–24.
- Morgenstern, C. (n.d.). *Graph generator ggen*. (Online reference, last accessed on July 25, 2023, <http://iridia.ulb.ac.be/~fmascia/files/ggen.tar.bz2>)
- Mota, E., Rocha, L., & Silva, A. (2020). Connected greedy coloring of H -free graphs. *Discrete Applied Mathematics*, 284, 572–584.
- Nogueira, B., Pinheiro, R. G., & Subramanian, A. (2018). A hybrid iterated local search heuristic for the maximum weight independent set problem. *Optimization Letters*, 12(3), 567–583.
- Pateromichelakis, E., & Samdanis, K. (2018). A graph coloring based inter-slice resource

- management for 5G dynamic TDD RANs. In *2018 IEEE International Conference on Communications (ICC)* (p. 1-6).
- San Segundo, P., Coniglio, S., Furini, F., & Ljubić, I. (2019). A new branch-and-bound algorithm for the maximum edge-weighted clique problem. *European Journal of Operational Research*, *278*(1), 76–90.
- Shi, Z., Goddard, W., Hedetniemi, S. T., Kennedy, K., Laskar, R., & McRae, A. (2005). An algorithm for partial Grundy number on trees. *Discrete Mathematics*, *304*(1-3), 108–116.
- Simmons, G. J. (1983). On the chromatic number of a graph. *Congressus Numerantium*, *40*, 339–366.
- Smith, M. D., Ramsey, N., & Holloway, G. (2004). A generalized algorithm for graph-coloring register allocation. In *Proceedings of the ACM SIGPLAN 2004 Conference on Programming Language Design and Implementation* (pp. 277–288).
- Telle, J. A., & Proskurowski, A. (1997). Algorithms for vertex partitioning problems on partial k -trees. *SIAM Journal on Discrete Mathematics*, *10*(4), 529–550.
- Toso, R. F. (2018). *API for biased random-key genetic algorithms*. <https://github.com/rfrancotoso/brkgaAPI>. (Accessed: 2022-12-05)
- Toso, R. F., & Resende, M. G. C. (2015). A C++ application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*, *30*(1), 81–93.
- Trick, M., Chvatal, V., Cook, B., Johnson, D., McGeoch, C., & Tarjan, B. (2015). *Benchmark instances from the Second DIMACS Implementation Challenge*. (Online reference, last access on July 25, 2023, <http://archive.dimacs.rutgers.edu/pub/challenge/graph/benchmarks/>)
- Wolsey, L. A. (2020). *Integer programming*. John Wiley & Sons.
- Zaker, M. (2005). Grundy chromatic number of the complement of bipartite graphs. *Australasian Journal of Combinatorics*, *31*, 325–330.
- Zaker, M. (2006). Results on the Grundy chromatic number of graphs. *Discrete Mathematics*, *306*(23), 3166–3173.
- Zhu, X., Dai, L., & Wang, Z. (2015). Graph coloring based pilot allocation to mitigate pilot contamination for multi-cell massive MIMO systems. *IEEE Communications Letters*, *19*(10), 1842–1845.

BRKGA RESULTS FOR THE GRUNDY COLORING PROBLEM

Tables A.1-A.5 summarize the results of the experiments with the BRKGA. In Tables A.1-A.4, the first column represents an instance group, so all the values in each row correspond to the average over five instances. The next three columns provide the average mean, maximum, and *time to best* (ttb) considering the 50 independent runs for each of the instances. The fifth and sixth columns provide the average best result found by MinDF and its average deviation from the best solution achieved by the BRKGA. The last two columns show the average best result achieved with any of the formulations (column best) and its average deviation from the best solution achieved by the BRKGA. The columns diff are defined as $diff = \frac{100 * (BRKGA_{max} - best)}{BRKGA_{max}}$. Table A.5 follows the same structure, but each row represents a single instance, implying that the values are not averaged over five instances.

group	BRKGA			MinDF		IP	
	mean	max	ttb	best	diff(%)	best	diff(%)
rand_50_0.2	11.2	11.6	35.8	7.6	34.5	10.8	6.9
rand_50_0.4	16.3	17.0	51.7	11.8	30.6	15.8	7.1
rand_50_0.6	22.6	23.4	81.1	16.6	29.1	22.6	3.4
rand_50_0.8	30.8	31.4	77.9	23.8	24.2	31.6	-0.6
rand_60_0.2	12.1	12.6	46.3	8.6	31.7	12.0	4.8
rand_60_0.4	18.8	19.8	65.2	12.8	35.4	18.8	5.1
rand_60_0.6	25.7	26.4	95.3	19.0	28.0	26.2	0.8
rand_60_0.8	35.9	36.8	78.9	27.6	25.0	36.8	0.0
rand_70_0.2	13.2	14.0	31.5	9.2	34.3	12.6	10.0
rand_70_0.4	20.8	21.8	79.1	15.8	27.5	20.2	7.3
rand_70_0.6	28.1	29.2	83.9	21.0	28.1	28.6	2.1
rand_70_0.8	39.5	40.6	93.9	30.8	24.1	40.0	1.5
rand_80_0.2	14.3	15.0	49.2	11.2	25.3	13.6	9.3
rand_80_0.4	22.3	23.4	66.8	16.0	31.6	21.0	10.3
rand_80_0.6	31.4	32.6	94.2	23.8	27.0	30.6	6.1
rand_80_0.8	43.1	44.6	82.1	32.6	26.9	44.8	-0.4
Mean	24.1	25.0	69.5	19.8	28.9	24.1	4.6

Table A.1: BRKGA results for the random graphs for the Grundy coloring problem

group	BRKGA			MinDF		IP	
	mean	max	ttb	best	diff(%)	best	diff(%)
geo_50_0.2	8.9	9.0	20.0	7.0	22.2	9.0	0.0
geo_50_0.4	22.1	22.2	26.0	18.0	18.9	22.2	0.0
geo_50_0.6	31.0	31.4	31.3	26.6	15.3	31.2	0.6
geo_50_0.8	37.8	38.0	3.9	34.8	8.4	38.2	-0.5
geo_60_0.2	10.0	10.0	5.2	7.8	22.0	10.2	-2.0
geo_60_0.4	26.3	26.6	41.7	19.4	27.1	26.6	0.0
geo_60_0.6	37.5	38.0	39.0	32.6	14.2	37.8	0.5
geo_60_0.8	47.5	47.6	19.8	44.6	6.3	47.6	0.0
geo_70_0.2	12.2	12.4	5.3	9.6	22.6	12.6	-1.6
geo_70_0.4	27.4	28.4	55.9	21.0	26.1	28.4	0.0
geo_70_0.6	41.7	42.4	42.1	36.4	14.2	42.0	0.9
geo_70_0.8	54.0	54.2	36.3	48.6	10.3	54.4	-0.4
geo_80_0.2	12.4	12.4	0.1	10.4	16.1	12.8	-3.2
geo_80_0.4	31.5	32.2	61.0	24.4	24.2	32.2	0.0
geo_80_0.6	47.7	48.4	52.1	40.6	16.1	48.2	0.4
geo_80_0.8	62.7	63.0	24.7	58.2	7.6	63.2	-0.3
Mean	31.9	32.2	29.0	27.5	16.9	32.2	-0.4

Table A.2: BRKGA results for the geometric graphs for the Grundy coloring problem

group	BRKGA			MinDF		IP	
	mean	max	ttb	best	diff(%)	best	diff(%)
bip_50_0.2	7.6	8.0	31.2	4.8	40.0	8.2	-2.5
bip_50_0.4	10.1	11.0	19.6	3.2	70.9	10.8	1.8
bip_50_0.6	12.8	13.4	65.1	2.8	79.1	13.8	-3.0
bip_50_0.8	14.9	16.0	59.4	2.0	87.5	17.0	-6.2
bip_60_0.2	8.2	8.6	19.4	4.2	51.2	9.2	-7.0
bip_60_0.4	10.9	11.6	53.3	2.2	81.0	11.6	0.0
bip_60_0.6	13.9	15.0	82.5	3.4	77.3	14.8	1.3
bip_60_0.8	17.2	18.8	92.7	2.0	89.4	19.8	-5.3
bip_70_0.2	8.6	9.2	27.7	5.0	45.7	9.4	-2.2
bip_70_0.4	11.7	12.4	62.4	2.4	80.6	12.6	-1.6
bip_70_0.6	14.6	15.8	81.9	2.8	82.3	16.4	-3.8
bip_70_0.8	18.4	20.2	110.7	2.0	90.1	21.2	-5.0
bip_80_0.2	9.4	9.8	49.1	5.2	46.9	10.0	-2.0
bip_80_0.4	12.8	13.8	76.3	3.6	73.9	13.6	1.4
bip_80_0.6	16.0	17.0	108.6	2.0	88.2	18.4	-8.2
bip_80_0.8	19.4	21.2	115.7	2.4	88.7	22.8	-7.5
Mean	12.9	13.8	65.9	3.1	73.3	14.3	-3.1

Table A.3: BRKGA results for the bipartite graphs for the Grundy coloring problem

group	BRKGA			MinDF		IP	
	mean	max	ttb	best	diff(%)	best	diff(%)
cbip_50_0.2	34.5	34.6	16.6	30.0	13.3	34.6	0.0
cbip_50_0.4	31.0	31.2	4.8	26.6	14.7	31.2	0.0
cbip_50_0.6	29.2	29.2	3.9	26.4	9.6	29.2	0.0
cbip_50_0.8	30.4	30.4	1.4	29.6	2.6	30.4	0.0
cbip_60_0.2	41.3	41.4	9.2	35.6	14.0	41.4	0.0
cbip_60_0.4	37.8	38.2	11.4	34.2	10.5	37.8	1.0
cbip_60_0.6	35.3	35.8	6.3	32.6	8.9	35.2	1.7
cbip_60_0.8	34.5	34.6	19.3	33.0	4.6	34.4	0.6
cbip_70_0.2	46.8	47.2	9.6	40.8	13.6	47.2	0.0
cbip_70_0.4	44.5	44.8	8.6	41.4	7.6	44.4	0.9
cbip_70_0.6	41.7	41.8	16.6	39.8	4.8	42.0	-0.5
cbip_70_0.8	41.1	41.2	1.4	40.4	1.9	41.0	0.5
cbip_80_0.2	51.1	51.6	3.5	44.0	14.7	51.4	0.4
cbip_80_0.4	47.1	47.4	7.9	42.2	11.0	47.2	0.4
cbip_80_0.6	46.2	46.4	1.1	43.8	5.6	46.0	0.9
cbip_80_0.8	44.7	44.8	19.8	43.6	2.7	44.6	0.4
Mean	39.8	40.0	8.8	36.5	8.7	39.8	0.4

Table A.4: BRKGA results for the complements of bipartite graphs for the Grundy coloring problem

instance	BRKGA			MinDF		IP	
	mean	max	ttb	best	diff(%)	best	diff(%)
johnson8-2-4	12.0	12.0	6.0	7.0	41.6	12.0	0.0
johnson8-4-4	29.2	31.0	116.6	14.0	54.4	29.0	0.6
mann_a9	21.0	21.0	0.0	20.0	4.7	21.0	0.0
hamming6-2	40.0	40.0	0.1	32.0	20.0	40.0	0.0
hamming6-4	13.9	15.0	57.4	8.0	38.4	13.0	13.3
c125.9	76.8	78.0	70.5	61.0	21.7	x	x
dsjc125.1	12.6	13.0	29.3	9.0	30.7	x	x
dsjc125.5	35.9	38.0	47.2	28.0	26.3	x	x
dsjc125.9	76.5	78.0	58.4	61.0	21.7	x	x
r125.1	7.0	7.0	0.0	6.0	14.2	x	x
r125.1c	62.0	62.0	0.7	51.0	17.7	x	x
r125.5	63.8	65.0	41.6	52.0	20.0	x	x
keller4	45.5	48.0	140.3	26.0	45.8	x	x
multsol.i.1	52.0	52.0	4.3	51.0	1.9	x	x
multsol.i.2	33.5	34.0	69.0	33.0	2.9	x	x
multsol.i.3	33.5	34.0	41.5	33.0	2.9	x	x
multsol.i.4	33.5	34.0	65.4	33.0	2.9	x	x
multsol.i.5	34.0	34.0	0.3	33.0	2.9	x	x
brock200_2	44.2	45.0	154.4	38.0	15.5	x	x
c-fat200-1	18.0	18.0	0.0	17.0	5.5	x	x
c-fat200-2	35.0	35.0	1.4	24.0	31.4	x	x
c-fat200-5	86.8	87.0	2.0	86.0	1.1	x	x
zeroin.i.1	52.8	53.0	87.9	50.0	5.6	x	x
zeroin.i.2	35.0	37.0	55.0	31.0	16.2	x	x
zeroin.i.3	35.0	37.0	45.7	31.0	16.2	x	x
dsjc250.1	18.0	18.0	53.5	13.0	27.7	x	x
r250.1	12.0	12.0	16.5	10.0	16.6	x	x
hamming8-2	159.6	161.0	199.8	128.0	20.4	x	x
hamming8-4	38.2	39.0	80.4	32.0	17.9	x	x
fpsol2.i.2	39.9	40.0	36.7	34.0	15.0	x	x
fpsol2.i.3	40.0	40.0	48.6	34.0	15.0	x	x
le450_5a	18.0	19.0	119.3	14.0	26.3	x	x
le450_5b	17.9	18.0	112.1	15.0	16.6	x	x
le450_5c	22.0	23.0	158.1	17.0	26.0	x	x
le450_5d	21.9	22.0	171.9	17.0	22.7	x	x
le450_15a	31.2	32.0	157.5	26.0	18.7	x	x
le450_15b	31.8	32.0	167.8	26.0	18.7	x	x
le450_25a	43.3	44.0	135.8	34.0	20.9	x	x
le450_25b	42.1	43.0	124.0	34.0	20.9	x	x
dsjr500.1	19.9	20.0	7.0	16.0	20.0	x	x
c-fat500-1	21.0	21.0	4.1	14.0	33.3	x	x
c-fat500-2	39.0	39.0	39.7	26.0	33.3	x	x
Mean	38.2	38.8	64.9	31.5	19.8	*	*

Table A.5: BRKGA results for the DIMACS instances for the Grundy coloring problem