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

A 1-factorization is a partition of the edge set of a graph into perfect matchings. The concept of 1-factorization is of great interest due to its applications in modeling sports tournaments. Two 1-factorizations are said to be isomorphic (belong to the same isomorphism class) if there exists a bijection between their sets of vertices that transforms one into the other. The non-isomorphic 1-factorization search space is a graph in which each isomorphism class is represented by a vertex and each edge that connects the vertices $\mathcal{F}_a$ and $\mathcal{F}_b$ corresponds to a move in a neighborhood structure, which from a 1-factorization isomorphic to $\mathcal{F}_a$ generates a 1-factorization isomorphic to $\mathcal{F}_b$. An invariant of a 1-factorization is a property that depends only on its structure such that isomorphic 1-factorizations are guaranteed to have equal invariant values. An invariant is complete when any two non-isomorphic 1-factorizations have distinct invariant values. This dissertation reviews seven invariants used to distinguish non-isomorphic 1-factorizations of K_{2n} (complete graph with an even number of vertices). Additionally, considering that the invariants available in the literature are not complete, we propose two new ones, denoted lantern profiles and even-size bichromatic chains. The invariants are compared regarding their sizes and calculation time complexity. Furthermore, we conduct computational experiments to assess their ability to distinguish non-isomorphic 1-factorizations. To accomplish that we use the sets of non-isomorphic 1-factorizations of K_{10} and K_{12}, as well as the sets of non-isomorphic perfect 1-factorizations of K_{14} and K_{16}. We also consider algorithmic and computational aspects for exploring the generalized partial team swap (GPTS) neighborhood, a neighborhood structure for round-robin sports scheduling problems recently proposed in the literature. In this regard, we present algorithms for systematically exploring the GPTS neighborhood. Furthermore, a discussion is presented on how this neighborhood structure increases the connectivity of the search space defined by non-isomorphic 1-factorizations of K_{2n} (for $8 \leq 2n \leq 12$) when compared to other neighborhood structures. Finally, preliminary computational experiments were conducted to evaluate the performance of the GPTS neighborhood, having the Weighted Carry-Over Effects Value Minimization Problem as a case study.

Keywords: 1-factorizations, Graph Theory, Isomorphism, Invariants, Edge coloring, Neighborhoods, Round-robin tournaments

DSC | 045 | 2023

Invariants and Neighborhood Structures for 1-factorizations of complete graphs

Saulo Antonio de Lima Matos

UFBA

Invariants and Neighborhood Structures for 1-factorizations of complete graphs

Saulo Antonio de Lima Matos

Tese de Doutorado

Universidade Federal da Bahia

Programa de Pós-Graduação em
Ciência da Computação

Outubro | 2023

Universidade Federal da Bahia
Instituto de Computação

Programa de Pós-Graduação em Ciência da Computação

INVARIANTS AND NEIGHBORHOOD STRUCTURES FOR 1-FACTORIZATIONS OF COMPLETE GRAPHS

Saulo Antonio de Lima Matos

TESE DE DOUTORADO

Salvador
03 de outubro de 2023

SAULO ANTONIO DE LIMA MATOS

INVARIANTS AND NEIGHBORHOOD STRUCTURES FOR 1-FACTORIZATIONS OF COMPLETE GRAPHS

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

Orientador: Prof. Dr. Rafael Augusto de Melo
Co-orientador: Prof. Dr. Tiago de Oliveira Januario

Salvador
03 de outubro de 2023

Ficha catalográfica elaborada pela Biblioteca Universitária de
Ciências e Tecnologias Prof. Omar Catunda, SIBI – UFBA.

M433 Matos, Saulo Antonio de Lima
Invariants and Neighborhood Structures for 1-factorizations of complete graphs. / Saulo Antonio de Lima Matos. – Salvador, 2023.
55 il.

Orientador: Prof. Dr. Rafael Augusto de Melo
Co-orientador: Prof. Dr. Tiago de Oliveira Januario

Tese (Doutorado) – Universidade Federal da Bahia, Instituto de Computação, 2023.

1. Teoria dos Grafos. 2. Isomorfismo. I. Melo, Rafael Augusto de. II. Januario, Tiago de Oliveira. III. Universidade Federal da Bahia. IV. Título.

CDU – 519.17

TERMO DE APROVAÇÃO

SAULO ANTONIO DE LIMA MATOS

INVARIANTS AND NEIGHBORHOOD STRUCTURES FOR 1-FACTORIZATIONS OF COMPLETE GRAPHS

Esta Tese de Doutorado foi julgada adequada à obtenção do título de Doutor em Ciência da Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia.

Salvador, 03 de OUTUBRO de 2023

Prof. Dr. Rafael Augusto de Melo
Universidade Federal da Bahia

Tiago

Prof. Dr. Tiago de Oliveira Januario
Universidade Federal da Bahia

Prof. Dr. Sebastián Alberto Urrutia
Molde University College

Vinicius Fernandes dos Santos

Prof. Dr. Vinicius Fernandes dos Santos
Universidade Federal de Minas Gerais

Marcio Costa Santos

Prof. Dr. Marcio Costa Santos
Universidade Federal de Minas Gerais

Prof. Dr. Celso da Cruz Carneiro Ribeiro
Universidade Federal da Bahia

ACKNOWLEDGEMENTS

Postgraduate Program in Computer Science (PGCOMP) at Federal University of Bahia (UFBA) and Bahia State Research Support Foundation (FAPESB) for the scholarship. My advisors, Rafael Melo and Tiago Januario, for their dedication and support.

RESUMO

Uma 1-fatoração é uma partição do conjunto de arestas de um grafo em emparelhamentos perfeitos. O conceito de 1-fatoração é de grande interesse devido às suas aplicações na modelagem de torneios esportivos. Duas 1-fatorações são ditas isomorfas (pertencem a mesma classe de isomorfismo) se existir uma bijeção entre seus conjuntos de vértices que transforme uma na outra. O espaço de busca de 1-fatorações não isomorfas é um grafo em que cada classe de isomorfismo é representada por um vértice e cada aresta que conecta os vértices $\mathcal{F}_a$ e $\mathcal{F}_b$ corresponde a um movimento em uma estrutura de vizinhança, que a partir de uma 1-fatoração isomorfa a $\mathcal{F}_a$ gera uma 1-fatoração isomorfa a $\mathcal{F}_b$. Uma invariante de uma 1-fatoração é uma propriedade que depende apenas de sua estrutura, de modo que 1-fatorações isomorfas possuem valores de invariantes iguais. Uma invariante é completa quando quaisquer duas 1-fatorações não isomorfas têm valores invariantes distintos. Essa tese analisa sete invariantes utilizadas para distinguir 1-fatorações não isomorfas de K_{2n} (grafos completos com quantidade par de vértices). Considerando que as invariantes disponíveis na literatura não são completas, propomos duas novas invariantes, denominadas *lantern profiles* e *even-size bichromatic chains*. As invariantes são comparadas quanto aos seus tamanhos e à complexidade computacional do seu cálculo. Além disso, realizamos experimentos computacionais para avaliar suas capacidades de distinguir 1-fatorações não isomorfas. Para tal, utilizamos os conjuntos de 1-fatorações não isomorfas de K_{10} e K_{12}, bem como os conjuntos de 1-fatorações perfeitas não isomorfas de K_{14} e K_{16}. Também consideramos aspectos algorítmicos e computacionais para explorar a vizinhança *generalized partial team swap* (GPTS), uma estrutura de vizinhança para problemas de planejamento de tabelas de torneios *round-robin* recentemente proposta na literatura. Nesse sentido, apresentamos algoritmos para explorar sistematicamente a vizinhança GPTS. Além disso, é apresentada uma discussão sobre como esta estrutura de vizinhança aumenta a conectividade do espaço de busca definido por 1-fatorações não isomorfas de K_{2n} (para $8 \leq 2n \leq 12$) quando comparada a outras estruturas de vizinhança. Por fim, experimentos computacionais preliminares foram conduzidos para avaliar o desempenho da vizinhança GPTS, utilizando como estudo de caso o *Weighted Carry-Over Effects Value Minimization Problem.*

Palavras-chave: 1-Fatorações, Teoria dos Grafos, Isomorfismo, Invariantes, Coloração de arestas, Vizinhanças, Torneios *round-robin*

ABSTRACT

A 1-factorization is a partition of the edge set of a graph into perfect matchings. The concept of 1-factorization is of great interest due to its applications in modeling sports tournaments. Two 1-factorizations are said to be isomorphic (belong to the same isomorphism class) if there exists a bijection between their sets of vertices that transforms one into the other. The non-isomorphic 1-factorization search space is a graph in which each isomorphism class is represented by a vertex and each edge that connects the vertices $\mathcal{F}_a$ and $\mathcal{F}_b$ corresponds to a move in a neighborhood structure, which from a 1-factorization isomorphic to $\mathcal{F}_a$ generates a 1-factorization isomorphic to $\mathcal{F}_b$. An invariant of a 1-factorization is a property that depends only on its structure such that isomorphic 1-factorizations are guaranteed to have equal invariant values. An invariant is complete when any two non-isomorphic 1-factorizations have distinct invariant values. This dissertation reviews seven invariants used to distinguish non-isomorphic 1-factorizations of K_{2n} (complete graph with an even number of vertices). Additionally, considering that the invariants available in the literature are not complete, we propose two new ones, denoted lantern profiles and even-size bichromatic chains. The invariants are compared regarding their sizes and calculation time complexity. Furthermore, we conduct computational experiments to assess their ability to distinguish non-isomorphic 1-factorizations. To accomplish that we use the sets of non-isomorphic 1-factorizations of K_{10} and K_{12}, as well as the sets of non-isomorphic perfect 1-factorizations of K_{14} and K_{16}. We also consider algorithmic and computational aspects for exploring the generalized partial team swap (GPTS) neighborhood, a neighborhood structure for round-robin sports scheduling problems recently proposed in the literature. In this regard, we present algorithms for systematically exploring the GPTS neighborhood. Furthermore, a discussion is presented on how this neighborhood structure increases the connectivity of the search space defined by non-isomorphic 1-factorizations of K_{2n} (for $8 \leq 2n \leq 12$) when compared to other neighborhood structures. Finally, preliminary computational experiments were conducted to evaluate the performance of the GPTS neighborhood, having the Weighted Carry-Over Effects Value Minimization Problem as a case study.

Keywords: 1-factorizations, Graph Theory, Isomorphism, Invariants, Edge coloring, Neighborhoods, Round-robin tournaments

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ABBREVIATIONS

P1Fs	*Perfect 1-factorizations*	43
PRS	*Partial Round Swap*	49
PTS	*Partial Team Swap*	49
GPTS	*Generalized Partial Team Swap*	49
SRR	*Single Round-Robin*	29
WCOEV	*Weighted Carry-Over Effects Value*	49
HC	*Hamiltonian Cycle*	32

Chapter 1

INTRODUCTION

1.1 MOTIVATION

A 1-factorization is a partition of the edge set of a graph into perfect matchings Two 1-factorizations are said to be isomorphic if there exists a bijection between their sets of vertices that transforms one into the other. The concept of 1-factorization is of great interest due to its applications in modeling sports tournaments. Several sports tournaments involving $2n$ teams are organized as *Single Round-Robin* (SRR) tournaments. It is natural to model such a tournament as a 1-factorization of K_{2n} (complete graph with an even number of vertices), with each vertex representing a team and each edge representing the game between the teams corresponding to its endpoints. Therefore, advances in the 1-factorization research area can influence the planning of sporting events in the future.

To tackle optimization problems in the context of SRR, search procedures (such as local search and explicit or implicit enumeration) are often used to explore the 1-factorizations of K_{2n}. Different neighborhood structures have been used in local search procedures for SRR tournament scheduling problems. The *Generalized Partial Team Swap* (GPTS) neighborhood was introduced by Januario et al. (2016). As pointed out by Ribeiro, Urrutia and de Werra (2023), due to the size and complex structure of GPTS neighborhood, it might be hard to design algorithms that systematically explore this neighborhood. Considering this and taking into account the lack of study regarding algorithmic and computational aspects of the GPTS neighborhood, it is necessary to design strategies to explore this neighborhood.

It is known that these neighborhood structures used in local search procedures may get stuck into a portion of the search space corresponding to 1-factorizations with the same structure (Costa; Urrutia; Ribeiro, 2012; Januario; Urrutia, 2015). Thus, identifying the isomorphism between two 1-factorizations becomes an important matter. Invariants can be used to distinguish between non-isomorphic 1-factorizations of K_{2n}. An invariant is complete when any two non-isomorphic 1-factorizations have distinct invariant values. There are several invariants for 1-factorizations of K_{2n} described in the literature, such as cycle profiles (Gelling, 1973), tricolor vectors (Griggs; Rosa, 1996), divisions (Mendelsohn;

Rosa, 1985), and trains (Dinitz; Wallis, 1991). Given that the invariants available in the literature are not complete and the research area related to the search for invariants to increase the capacity of distinction remains open, new ones can be proposed.

1.2 OBJECTIVES AND CONTRIBUTIONS

The main objectives of this dissertation are to propose new invariants for 1-factorizations of K_{2n} and present algorithms for systematically exploring the GPTS neighborhood structure, proposed by Januario et al. (2016).

In order to achieve the first objective, two new invariants will be proposed, and their capacities to distinguishing non-isomorphic 1-factorizations of K_{2n} will be analyzed. Furthermore, it is necessary to compare these invariants with other invariants available in the literature. For that purpose, we aim to evaluate the invariants' distinguish capacity on different sets of non-isomorphic 1-factorizations of K_{2n}. Besides, different combinations of the invariants must be analyzed to assess their complementarity.

Regarding the second objective, it is necessary to propose and develop efficient strategies for obtaining neighbors in this neighborhood. Additionally, it is important to investigate the connectivity of the GPTS neighborhood structure. This investigation will address whether the GPTS neighborhood structure increases the connectivity of the search space defined by non-isomorphic 1-factorizations of K_{2n} when compared to other neighborhood structures.

The first contribution of this dissertation is a review of invariants used to distinguish non-isomorphic 1-factorizations of K_{2n}. Additionally, considering that these invariants are not complete, we propose two new ones, denoted *lantern profiles* and *even-size bichromatic chains*. The invariants are compared regarding their sizes and calculation time complexity. Furthermore, we conduct computational experiments to assess their ability to distinguish non-isomorphic 1-factorizations. To accomplish that we use the sets of non-isomorphic 1-factorizations of K_{10} and K_{12}, as well as subsets of the non-isomorphic 1-factorizations of K_{14} and K_{16}. Moreover, computational results show that four of the invariants (including the two proposed invariants) have shown to be much stronger than the others when considering the sets of non-isomorphic 1-factorizations of K_{12} tested. Last but not least, the combinations of invariants were also evaluated, showing a complementarity in their distinguishing abilities.

Regarding the GPTS neighborhood structure, the main contribution is to present algorithms to systematically explore this neighborhood. Additionally, this dissertation presents two strategies for obtaining neighbors in the GPTS neighborhood. An additional contribution is a study on algorithmic and computational aspects of the GPTS neighborhood. Furthermore, this work provides a discussion on how this neighborhood structure increases the connectivity of the search space defined by non-isomorphic 1-factorizations of K_{2n} (for $8 \leq 2n \leq 12$) when compared to other neighborhood structures. We show that the GPTS is able to directly connect more pairs of non-isomorphic 1- factorizations of both K_8 and K_{10}. Additionally, the GPTS is the only one capable of connecting the non-isomorphic 1-factorization search space of K_{12}. Finally, we present preliminary computational experiments that were conducted to evaluate the performance of the GPTS, having the *Weighted Carry-Over Effects Value* (WCOEV) Minimization Problem as a case study.

1.3 ELEMENTS OF GRAPH THEORY

This subsection describes the definitions of graph theory that are being used in this work. Let $G = (V, E)$ be a simple and undirected graph with vertex set V and edge set $E \subset V \times V$. If $V' \subseteq V$ and $E' \subseteq E$, then $G' = (V', E')$ is a *subgraph* of $G = (V, E)$. Two vertices u, v joined by an edge $e = uv$ are said to be *adjacent*, or *neighbors*, and that edge is said to be *incident* to u and v. The set of *neighbors* of a vertex v in G is denoted by $N(v)$, also called the *neighborhood* of v. The *degree* $d(v)$ of a vertex v is the number of edges incident to it. The number $\Delta(G) = max\{d(v) : v \in V\}$ is the *maximum degree* of a vertex in G.

A *chain* in a simple graph is a sequence of adjacent edges that allow linking two vertices in the graph. A graph is *connected* if, for any pair of vertices, there is a chain linking them. The *length* k of a chain is given by its number of edges. An *even-size chain* is a chain of even length. A chain of length k is called a *k-chain*. If $P = (v_1, \ldots, v_{k-1})$ is a chain and $k \geq 3$, then $C = P \cdot (v_1) = (v_1, \ldots, v_{k-1}, v_1)$ is called a *cycle*. A cycle of length k is called a *k-cycle*. A *Hamiltonian Cycle* (HC) is a cycle going through every vertex of the graph.

Two graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$ are *isomorphic* if there exists a bijective function φ from V_G to V_H such that $xy \in E_G$ if and only if $\varphi(x)\varphi(y) \in E_H$. Informally, graphs G and H are isomorphic if it is possible to obtain H from G (and vice versa) just by renaming its vertices.

A 1-factorization of G is a partition of E into perfect matchings. Each one of these matchings is called a *1-factor* of the 1-factorization. Figure 1.1 depicts a 1-factorization of K_4. Notice that not every graph has a 1-factorization. For instance, graphs with an odd number of vertices do not have perfect matchings and, thus, cannot have 1-factorizations. Complete graphs K_{2n} do have 1-factorizations. The union of any two 1-factors of a given 1-factorization is a 2-regular graph consisting of a set of even-size cycles. A 1-factorization is said to be *perfect* if the union of any two of its 1-factors forms a HC.

Two 1-factorizations $\mathcal{F} = \{F_1, \ldots, F_k\}$ and $\mathcal{H} = \{H_1, \ldots, H_k\}$ of G are called *isomorphic* if there exists a bijective function φ from the node set V of G onto itself such that $\{F_1^\varphi, \ldots, F_k^\varphi\} = \{H_1, \ldots, H_k\}$, where F_i^φ is the set of all edges $\varphi(x)\varphi(y)$ and xy is an edge in F_i. We say that two isomorphic 1-factorizations belong to the same *isomorphism class* (or equivalence class).

A *proper edge coloring* of a graph G is a mapping $c : E \rightarrow \mathcal{C}$ so that $c(e_i) \neq c(e_j)$ for any adjacent edges e_i and e_j, where the elements of $\mathcal{C}$ are the available colors. A *k-coloring* is a proper edge coloring using k colors. Notice that a 1-factorization provides a proper edge coloring of a graph by associating a color to each 1-factor. This is illustrated in Figure 1.1 for a 1-factorization of K_4, in which the edges colored with red, green, and blue, belong, respectively, to the 1-factors F_1, F_2, and F_3. A color α is incident to vertices u and v if the edge $e = uv$ has that color. A *bichromatic chain* is a chain in which its edges are alternately colored with two different colors.

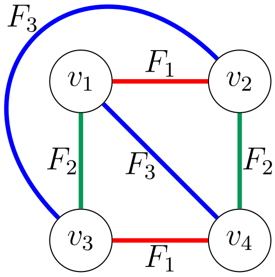


Figure 1.1: A 1-factorization $\mathcal{F} = \{F_1, F_2, F_3\}$ of K_4, with $F_1 = \{v_1v_2, v_3v_4\}$, $F_2 = \{v_1v_3, v_2v_4\}$, and $F_3 = \{v_1v_4, v_2v_3\}$.

The concept of isomorphic 1-factorizations can also be extended to *isomorphic colorings*, therefore two colorings are said to be isomorphic if, and only if, their associated 1-factorizations are isomorphic. In the remainder of this work, for the sake of simplicity, we denote proper edge coloring by simply edge coloring or coloring, unless stated otherwise.

1.3.1 1-factorizations and 1-factorization isomorphism

This subsection reviews some of the literature on 1-factorization isomorphism.

Different works are concerned with the computation of the largest possible set of non-isomorphic 1-factorizations of complete graphs. It is known that this task has a high computational cost since the number of non-isomorphic 1-factorizations of K_{2n} increases very fast with n. Dinitz, Garnick and McKay (1994) state that the computation of all non-isomorphic 1-factorizations of K_{12} would require over 160 MIPS-years of CPU time (which is equivalent to 160 years on a computer running at 1 million instructions per second) on a single computer.

Regarding the number of 1-factorizations, there is a unique 1-factorization of K_4, six of K_6, and $6,240$ of K_8 (Wallis, 2007). There are $1,255,566,720$ 1-factorizations of K_{10} (Gelling, 1973), $252,282,619,805,368,320$ of K_{12} (Dinitz; Garnick; McKay, 1994), and $98,758,655,816,833,727,741,338,583,040$ of K_{14} (Kaski; Östergård, 2009). Nevertheless, the number of isomorphism classes of 1-factorizations for such complete graphs can be much smaller. There is a unique isomorphism class of 1-factorizations for K_4 and K_6, six for K_8 (Dickson; Safford, 1906), 396 for K_{10} (Gelling; Odeh, 1974), $526,915,620$ for K_{12} (Dinitz; Garnick; McKay, 1994), and $1,132,835,421,602,062,347$ for K_{14} (Kaski; Östergård, 2009).

Among the studies on the number of non-isomorphic *Perfect 1-factorizations* (P1Fs), the ones conducted in Petrenyuk and Petrenyuk (1980), Dinitz and Garnick (1996), Meszka (2020), and Gill and Wanless (2020) stand out for the graphs K_{12}, K_{14}, and K_{16}. The reported results indicate that the numbers of isomorphism classes of P1Fs are 5 for K_{12}, 23 for K_{14}, and $3,155$ for K_{16}. In addition, for $4 \leq 2n \leq 10$ there is a unique perfect 1-factorization of K_{2n}. A recent survey on P1Fs can be found in Rosa (2019).

A survey (Mendelsohn; Rosa, 1985) and an entire book (Wallis, 1997) were devoted exclusively to 1-factorizations.

1.4 TEXT ORGANIZATION

The remainder of this dissertation is organized as follows. Chapter 2 reviews and details seven invariants available in the literature and proposes two new ones For all the nine invariants presented, we analyzed their size and computational complexity. Furthermore, we conduct computational experiments to assess their ability to distinguish non-isomorphic 1-factorizations. Chapter 3 discusses the algorithmic and computational aspects of the GPTS neighborhood and presents algorithms to systematically explore the GPTS neighborhood. This chapter also presents some computational experimental analyses that were carried out in order to compare the GPTS neighborhood structure to other neighborhood structures. Chapter 4 presents some concluding remarks and discusses possible research directions.

Chapter

2

A TUTORIAL ON INVARIANTS FOR 1-FACTORIZATIONS OF K_{2n}: DESCRIPTION AND COMPUTATION

Invariants are widely used in situations in which one wants to quickly verify whether two structures are non-isomorphic. In particular, invariants are needed to speed up classification algorithms. For instance, if the invariant values of two objects are different, then no further tests are needed to determine that they are structurally different. An invariant of a 1-factorization is a property that depends only on its structure such that isomorphic 1-factorizations are guaranteed to have equal invariant values. As such, non-isomorphic 1-factorizations may or may not have different invariant values. An invariant is complete when any two non-isomorphic 1-factorizations have distinct invariant values.

This chapter reviews seven invariants used to distinguish non-isomorphic 1-factorizations of K_{2n} (complete graphs with an even number of vertices). Additionally, considering that the invariants available in the literature are not complete, we propose two new ones, denoted lantern profiles and even-size bichromatic chains. The invariants are compared regarding their sizes and calculation time complexity. Furthermore, we conduct computational experiments to assess their ability to distinguish non-isomorphic 1-factorizations. To accomplish that we use the sets of non-isomorphic 1-factorizations of K_{10} and K_{12}, as well as the sets of non-isomorphic *Perfect 1-factorizations* (P1Fs) of K_{14} and K_{16}. Moreover, the experiments evaluate the combination of some of the invariants.

2.1 BASIC DEFINITIONS

A mapping $I : G \to \mathbb{R}^m$ that extracts properties from a graph and maps them to an m-dimensional vector is an *invariant* if it assigns equal values to isomorphic graphs, and we say that this invariant has *size m*. Given the graphs G_1 and G_2, if G_1 is isomorphic to G_2, then $I(G_1) = I(G_2)$. In case $I(G_1) = I(G_2)$ if only if G_1 and G_2 are isomorphic, then the invariant is said to be *complete*. In other words, if G_1 and G_2 are not isomorphic and I is a complete invariant, then $I(G_1) \neq I(G_2)$. Consider the two graphs illustrated in

Figure 2.1. Both have the same number of vertices and edges. Thus, a simple invariant that considers only these properties cannot distinguish between them. However, when considering the sorted degree distributions at their vertices, we can define a new invariant capable of distinguishing such graphs. It is much more efficient to test for isomorphism by checking the invariant values instead of checking for all possible isomorphism functions.

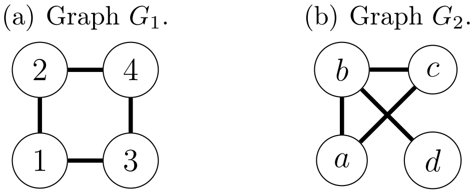


Figure 2.1: The graph G_1 (on the left) and the graph G_2 (on the right) have the same number of vertices and edges. Therefore, an invariant based on those values would not be able to distinguish them. By considering an invariant based on their sorted degree distributions, i.e., $I(G_1) = (2, 2, 2, 2)$ and $I(G_2) = (1, 2, 2, 3)$, we can assert that they are not isomorphic since their invariant values are different.

Two isomorphic 1-factorizations belong to the same isomorphism class (or equivalence class). For instance, there are 6, 240 distinct 1-factorizations of K_8, but they can be classified into only six isomorphism classes. Figure 2.2 illustrates two isomorphic 1-factorizations of K_8. Table 2.1 presents six non-isomorphic 1-factorizations of K_8, each representing its isomorphism class.

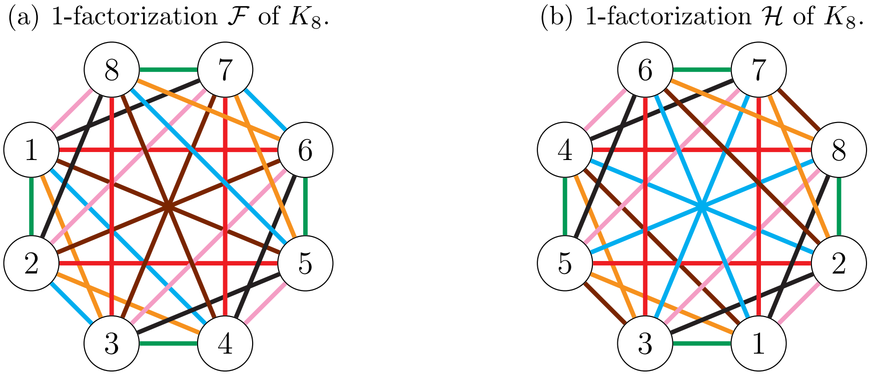

(a) 1-factorization $\mathcal{F}$ of K_8.
(b) 1-factorization $\mathcal{H}$ of K_8.
8
7
1
6
2
5
3
4
6
7
4
8
5
2
3
1

Figure 2.2: Examples of isomorphic 1-factorizations of K_8. The seven 1-factors of each 1-factorization are characterized by different colors. $\mathcal{H}$ (on the right) can be obtained from $\mathcal{F}$ (on the left) with the function φ defined as follows: $\varphi(1) = 4$, $\varphi(2) = 5$, $\varphi(3) = 3$, $\varphi(4) = 1$, $\varphi(5) = 2$, $\varphi(6) = 8$, $\varphi(7) = 7$, and $\varphi(8) = 6$.

The concept of invariant can be extended to 1-factorizations. A mapping $I_f : \mathcal{F} \to \mathbb{R}^m$ that extracts properties of a 1-factorization and maps them to an m-dimensional vector is a *1-factorization invariant* if it assigns equal values to isomorphic 1-factorizations. This means that, given the 1-factorizations $\mathcal{F}_1$ and $\mathcal{F}_2$, if $\mathcal{F}_1$ is isomorphic to $\mathcal{F}_2$, then $I_f(\mathcal{F}_1) = I_f(\mathcal{F}_2)$. In case $I_f(\mathcal{F}_1) = I_f(\mathcal{F}_2)$ only if $\mathcal{F}_1$ and $\mathcal{F}_2$ are isomorphic, then the invariant is said to be *complete*. The *strength* of an invariant is defined as the number of isomorphism classes identified in a set of non-isomorphic 1-factorizations, which is determined by the quantity of distinct invariant values for the non-isomorphic 1-factorizations in this set.

Table 2.1: Six non-isomorphic 1-factorizations of K_8. Each 8-digit block represents a 1-factor. For instance, 12345678 represents the 1-factor $\{12, 34, 56, 78\}$.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7
$\mathcal{F}_0$	12345678	13245768	14235867	15263748	16253847	17283546	18273645
$\mathcal{F}_1$	12345678	13245768	14235867	15263748	16253847	17283645	18273546
$\mathcal{F}_2$	12345678	13245768	14235867	15273648	16283745	17253846	18263547
$\mathcal{F}_3$	12345678	13245768	14235867	15273846	16283745	17253648	18263547
$\mathcal{F}_4$	12345678	13245768	14253867	15273648	16283745	17234658	18263547
$\mathcal{F}_5$	12345678	13254768	14273658	15283746	16234857	17263845	18243567

2.2 CONTRIBUTIONS AND ORGANIZATION

In this chapter, we concentrate on ways to distinguish between non-isomorphic 1-factorizations of K_{2n}. First, we review the invariants available in the literature. Second, we propose two new invariants. Last but not least, computational results are performed to evaluate the strength of the described invariants. The tests analyze their capacity to distinguish non-isomorphic 1-factorizations. For that purpose, we consider the sets of 1-factorizations of K_{10} and K_{12}, as well as the sets of P1Fs of K_{14} and K_{16}. Different combinations of the invariants are also analyzed to evaluate their complementarity. Finally, we considered randomly generated 1-factorizations of K_{16} and K_{20}.

The remainder of this chapter is organized as follows. Section 2.3 reviews the related literature. Section 2.4 details seven invariants for 1-factorizations from the literature. Section 2.5 proposes two new invariants: lantern profiles and even-size bichromatic chains. Section 2.6 summarizes the computational experiments. Section 2.7 discusses some concluding remarks.

2.3 RELATED WORKS

This section reviews some of the literature on graph isomorphism and invariants.

Graph isomorphism as a computational problem first appeared in the chemistry literature of the 1950s as the problem of matching a molecular graph against a database of such graphs (Grohe; Schweitzer, 2020). The question of whether graph isomorphism is solvable in polynomial time remains open. However, polynomial algorithms are known for testing the isomorphism of many classes of graphs. Additionally, it is claimed that the general graph isomorphism problem can be solved in quasipolynomial time (Babai, 2015; Babai, 2016; Helfgott; Bajpai; Dona, 2017). Subgraph isomorphism, on the other hand, is long known to be NP-complete (Garey; Johnson, 1979). Although an efficient algorithm for the graph isomorphism problem is not known, there are software available that can be executed in low computational times in practice for certain graphs. McKay and Piperno (2014) lists various software for isomorphism testing.

There are several invariants for graphs (Brigham; Dutton, 1985), some of which are trivial as the number of vertices and edges. Other examples are the maximum, minimum, and average degrees, connectivity, chromatic number, chromatic index, and the existence of a cycle. Although these invariants are not complete, based on them, we can construct invariants that can be used to distinguish non-isomorphic 1-factorizations of K_{2n}.

2.4 EXISTING INVARIANTS FOR 1-FACTORIZATIONS

In this section, we detail seven invariants for 1-factorizations described in the literature: cycle profiles (Gelling, 1973), tricolor vectors (Griggs; Rosa, 1996), divisions (Wallis, 1973; Wallis, 1997), trains (Dinitz; Wallis, 1991), and the three invariants proposed in Gill and Wanless (2020), denoted by trains-path, row-cycles, and row-cycles-per-row. To illustrate how each invariant works, we will use the six non-isomorphic 1-factorizations of K_8 presented in Table 2.1.

2.4.1 Cycle profiles

Given a 1-factorization $\mathcal{F}$, let $c(v, F_i, F_j)$ be the cycle size containing the vertex $v \in V$ in the subgraph formed by the two 1-factors F_i and F_j. Moreover, let $c_k(v) = |\{(i, j) : c(v, F_i, F_j) = k, 1 \leq i < j \leq 2n-1\}|$ be the number of k-cycles containing the vertex $v \in V$ considering any pair of distinct 1-factors. Given a 1-factorization $\mathcal{F}$ and an integer k, $4 \leq k \leq 2n$, the *cycle profiles* invariant (Gelling, 1973) is defined as the sorted sequence $(c_k(v_{\pi_k(1)}), c_k(v_{\pi_k(2)}), \ldots, c_k(v_{\pi_k(2n)}))$, where $v_{\pi_k(\ell)}$ is the vertex participating in the ℓ^{th} largest number of k-cycles. In this work, we consider the cycle profiles invariant, which we denote by $I_f^{cp}(\mathcal{F})$, to be the vector formed from the concatenation of the sequences for all the relevant values of k. Consequently, if two 1-factorizations have distinct sequences, they are not isomorphic. Notice that this invariant has size $\Theta(n^2)$. Furthermore, observe that cycle profiles does not distinguish between P1Fs since for any vertex and pair of 1-factors, $k = 2n$ is the only possible value.

Table 2.2 shows that cycle profiles is a complete invariant for the 1-factorizations of K_8, as the values for the six considered non-isomorphic 1-factorizations are pairwise different. Notice that for the graph K_8, the size of any cycle formed by two 1-factors will be either four or eight. Thus, the choice for the parameter k is restricted to $k \in \{4, 8\}$. Moreover, each vertex belongs to the same number of cycles of size k in every 1-factorization of K_8 for both values of k.

Proposition 2.1. *The cycle profiles invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. Consider that each 1-factor is represented by adjacency lists. A traversal on the union of each pair of distinct 1-factors (F_i, F_j), with $i < j$, can thus be performed in $O(n)$ to obtain all the disjoint cycles. After obtaining each cycle, it can be traversed, and the

Table 2.2: Cycle profiles invariant values for the six non-isomorphic 1-factorizations of K_8. The value of $c_k(v)$ is the same for every $v \in V(K_8)$.

	4-cycles: sorted $c_4(v)$ values								8-cycles: sorted $c_8(v)$ values							
$I_f^{cp}(\mathcal{F}_0)$	21	21	21	21	21	21	21	21	0	0	0	0	0	0	0	0
$I_f^{cp}(\mathcal{F}_1)$	13	13	13	13	13	13	13	13	8	8	8	8	8	8	8	8
$I_f^{cp}(\mathcal{F}_2)$	7	7	7	7	7	7	7	7	14	14	14	14	14	14	14	14
$I_f^{cp}(\mathcal{F}_3)$	9	9	9	9	9	9	9	9	12	12	12	12	12	12	12	12
$I_f^{cp}(\mathcal{F}_4)$	3	3	3	3	3	3	3	3	18	18	18	18	18	18	18	18
$I_f^{cp}(\mathcal{F}_5)$	0	0	0	0	0	0	0	0	21	21	21	21	21	21	21	21

value of $c(v, F_i, F_j)$ can be set for every vertex v contained in it, which can be done in $O(n)$ for all the cycles. Thus, we have $O(n^2) \times O(n)$, resulting in $O(n^3)$. ▮

2.4.2 Tricolor vectors

Given a 1-factorization $\mathcal{F}$ and three distinct vertices, the edges between them belong to exactly three distinct 1-factors. Denote this property by $\mathcal{T}(u, v, w) = \{F_i, F_j, F_k\}$ where u, v, and w are the vertices and F_i, F_j, and F_k are the corresponding 1-factors. Let $N(F_i, F_j, F_k) = |\{(u, v, w) : 1 \le u < v < w \le 2n, \mathcal{T}(u, v, w) = \{F_i, F_j, F_k\}\}|$ be the number of unordered triples of distinct vertices $\{u, v, w\}$ such that $\mathcal{T}(u, v, w) = \{F_i, F_j, F_k\}$. The *tricolor vectors* invariant (Griggs; Rosa, 1996) of a 1-factorization, which we denote by $I_f^{tv}(\mathcal{F})$, is defined as the vector corresponding to the sequence $(\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_{2n})$, where $\mathcal{T}_q$ is the number of triples $\{F_i, F_j, F_k\}$ such that $N(F_i, F_j, F_k) = q$. The first element $\mathcal{T}_0$ is called the *tricolor number* (Wallis, 1997). Notice that the maximum number of times a given set of three 1-factors may be the image of the function $\mathcal{T}$ is limited by the number of vertices $2n$. To see this observe that, in the 3-regular graph formed by three specific 1-factors, a given vertex may be part of at most $\binom{3}{2} = 3$ triangles. The result follows since each triangle is composed of three distinct vertices. Notice that this invariant has size $\Theta(n)$.

Table 2.3 presents each one of the tricolor vectors of the six non-isomorphic 1-factorizations of K_8. Observe that tricolor vectors is complete for the 1-factorizations of K_8.

Table 2.3: Tricolor vectors invariant for for the six non-isomorphic 1-factorizations of K_8.

	$\mathcal{T}_0$	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_3$	$\mathcal{T}_4$	$\mathcal{T}_5$	$\mathcal{T}_6$	$\mathcal{T}_7$	$\mathcal{T}_8$
$I_f^{tv}(\mathcal{F}_0)$	28	0	0	0	0	0	0	0	7
$I_f^{tv}(\mathcal{F}_1)$	24	0	0	0	8	0	0	0	3
$I_f^{tv}(\mathcal{F}_2)$	18	0	8	0	8	0	0	0	1
$I_f^{tv}(\mathcal{F}_3)$	22	0	0	0	12	0	0	0	1
$I_f^{tv}(\mathcal{F}_4)$	9	8	12	0	6	0	0	0	0
$I_f^{tv}(\mathcal{F}_5)$	0	14	21	0	0	0	0	0	0

Proposition 2.2. *The tricolor vectors invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. Consider the 1-factorization represented as an adjacency matrix with each element indicating the 1-factor in which each pair of vertices is adjacent. For each vertex $u \in V$, we consider all the pairs (v, w) of vertices in $V \setminus u$, with $v \neq v'$. Let F_i, F_j, and F_k be the 1-factors of the edges uv, vw, and wu, respectively. We just have to account this triangle towards the value $N(F_i, F_j, F_k)$. This can be done in $O(n^2)$ for each vertex, implying $O(n^3)$ for all the vertices. Notice that the value of each $N(F_i, F_j, F_k)$ must be divided by three since each triangle will be found three times, once from each one of its vertices. Given the values of $N(F_i, F_j, F_k)$ for every triplet $\{F_i, F_j, F_k\}$, we now compute the invariant using counting in $O(n^3)$. ∎

2.4.3 Divisions

Given a 1-factorization $\mathcal{F}$, a *d-division* is a set of d 1-factors whose union is disconnected. A *d-division* is considered *maximal* if it is not contained in a $(d+1)$*-division*, i.e., any 1-factor added to this *d-division* will make their union connected. As an example, for $d = 2$, taking F_i and F_j as two 1-factors of a 1-factorization, their union will have at least one cycle. If the subgraph induced by the union of these two 1-factors has more than one cycle, we have a 2*-division*. Let the value $\alpha_d(\mathcal{F})$ be the number of maximal *d-divisions* in the 1-factorization $\mathcal{F}$. The *divisions* invariant (Mendelsohn; Rosa, 1985; Wallis, 1997), which we denote by $I^d_f(\mathcal{F})$, is given by the vector formed by the values α_d for all the possible d. Observe that the *divisions* value for P1Fs is always zero as the union of any two 1-factors is a *Hamiltonian Cycle* (HC).

Table 2.4 shows the divisions invariant values for the six non-isomorphic 1-factorizations of K_8. The table also provides the maximal divisions corresponding to those values. Notice that the 1-factorization $\mathcal{F}_2$ of K_8 has a unique maximal 3-division $= \{\{F_1, F_2, F_3\}\}$ and four 2-divisions $= \{\{F_3, F_4\}, \{F_3, F_6\}\ , \{F_4, F_6\}, \{F_5, F_7\}\}$. Except for $F_1 \cup F_2$, $F_1 \cup F_3$, $F_2 \cup F_3$, and each one of the four combinations that form a maximal 2-division, the remaining 14 combinations of two 1-factors of $\mathcal{F}_2$ from K_8 form a HC. Notice that divisions is a complete invariant for the 1-factorizations of K_8.

Proposition 2.3. *The divisions invariant for a given 1-factorization can be computed in $O(n^d \times nd)$.*

Proof. Observe that there are $\binom{2n}{d} = O(n^d)$ combinations of d 1-factors. For each one of these combinations, connectivity can be evaluated in $O(nd)$. Thus, all the calculations can be performed in $O(n^d \times nd)$. This leads to $O(n^{d+1})$ whenever d is a constant. ∎

2.4.4 Trains

Given a 1-factorization $\mathcal{F}$, its associated *train graph* $T(\mathcal{F})$ is a directed graph with $n(2n-1)^2$ vertices in which each vertex is a triple $\{u, v, F\}$, where $\{u, v\}$ is an unordered pair of vertices and F is a 1-factor of $\mathcal{F}$. In a $T(\mathcal{F})$, exactly one arc leaves from each vertex, and the direction of each arc is determined as follows:

(i) a loop at the vertex $\{u, v, F_i\}$, if $uv \in F_i$;

(ii) an arc from the vertex $\{u, v, F_i\}$ to $\{w, z, F_j\}$, if $uw \in F_i$, $vz \in F_i$ and $uv \in F_j$.

Figure 2.3 illustrates the train graph associated with the 1-factorization of K_4 depicted in Figure 1.1.

Table 2.4: Divisions invariant for the six non-isomorphic 1-factorizations of K_8 (Adapted from (Wallis, 2007)).

	α_3	α_2	*3-division maximal*	*2-division maximal*
$I_f^d(\mathcal{F}_0)$	7	0	$\{F_1, F_2, F_3\}$, $\{F_1, F_4, F_5\}$, $\{F_1, F_6, F_7\}$, $\{F_2, F_4, F_6\}$, $\{F_2, F_5, F_7\}$, $\{F_3, F_4, F_7\}$, $\{F_3, F_5, F_6\}$	
$I_f^d(\mathcal{F}_1)$	3	4	$\{F_1, F_2, F_3\}$, $\{F_1, F_4, F_5\}$, $\{F_1, F_6, F_7\}$	$\{F_2, F_4\}$, $\{F_2, F_5\}$, $\{F_3, F_4\}$, $\{F_3, F_5\}$
$I_f^d(\mathcal{F}_2)$	1	4	$\{F_1, F_2, F_3\}$	$\{F_3, F_4\}$, $\{F_3, F_6\}$, $\{F_4, F_6\}$, $\{F_5, F_7\}$
$I_f^d(\mathcal{F}_3)$	1	6	$\{F_1, F_2, F_3\}$	$\{F_4, F_5\}$, $\{F_4, F_6\}$, $\{F_4, F_7\}$, $\{F_5, F_6\}$, $\{F_5, F_7\}$, $\{F_6, F_7\}$
$I_f^d(\mathcal{F}_4)$	0	3		$\{F_1, F_2\}$, $\{F_3, F_6\}$, $\{F_5, F_7\}$
$I_f^d(\mathcal{F}_5)$	0	0		

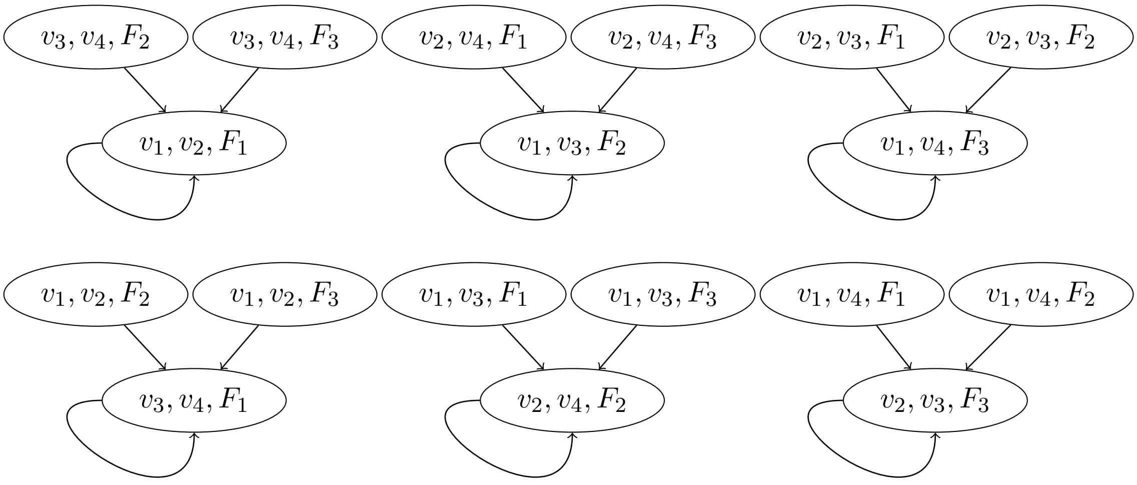


Figure 2.3: A *train* graph obtained from the K_4 in Figure 1.1.

Two isomorphic 1-factorizations have isomorphic associated train graphs. However, whether verifying the isomorphism of train graphs can be solved in polynomial time is unknown. The simplified trains invariant (Dinitz; Wallis, 1991) is based on the sequence of indegrees of the vertices in the train graph. Thus, for a given 1-factorization $\mathcal{F}$, let $(t_0, t_1, \ldots, t_{\ell_{max}})$ be a sequence, where t_ℓ is equal to the number of vertices in $T(\mathcal{F})$ that have ℓ input arcs, with ℓ_{max} defining the index of the last nonzero element. The trains invariant, which we denote by $I^t_f(\mathcal{F})$, is defined as the vector corresponding to the sequence $(t_0, t_1, \ldots, t_{\ell_{max}})$. Dinitz and Wallis (1991) showed that $\ell_{max} \leq 2n-1$ and, thus, the invariant has size $\Theta(n)$. The trains invariant associated with the 1-factorization of K_4 depicted in Figure 1.1, whose train graph is provided in Figure 2.3, is given by $(12, 0, 0, 6)$. This means that twelve vertices have an indegree equal to zero, and six vertices have an indegree equal to three.

Table 2.5 shows the simplified trains invariant values for the six non-isomorphic 1-factorizations of K_8. It shows that trains is complete for K_8.

Table 2.5: Trains invariant values for the six non-isomorphic 1-factorizations of K_8.

	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
$I^t_f(\mathcal{F}_0)$	168	0	0	0	0	0	0	28
$I^t_f(\mathcal{F}_1)$	144	0	16	8	8	16	0	4
$I^t_f(\mathcal{F}_2)$	112	16	36	24	4	4	0	0
$I^t_f(\mathcal{F}_3)$	108	48	0	12	28	0	0	0
$I^t_f(\mathcal{F}_4)$	72	64	48	12	0	0	0	0
$I^t_f(\mathcal{F}_5)$	42	112	42	0	0	0	0	0

Proposition 2.4. *The simplified trains invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. Assume that each 1-factor is represented by adjacency lists. In addition, assume an adjacency matrix is available with each element indicating the 1-factor in which each pair of vertices is adjacent. For each combination of pair of vertices and 1-factor $\{u, v, F\}$ forming a vertex of the trains graph, the outgoing arc must be computed. The destination of this arc can be computed in constant time by using the adjacency matrix to determine the 1-factor where the edge $[u, v]$ is and the adjacency lists to consult the vertices that are adjacent to them in F. After computing the indegree of each vertex in $O(n^3)$, the invariant can be computed by using counting sort to determine the number of vertices with each possible indegree in $O(n^3)$. ∎

2.4.5 Trains-path

For each vertex $\{u, v, F_i\}$ of the train graph (see Section 2.4.4), define $p(\{u, v, F_i\})$ as the length of the shortest directed path from $\{u, v, F_i\}$ to any vertex $\{w, z, F_j\}$ belonging to a directed cycle. Notice that if $\{u, v, F_i\}$ itself is in a cycle then $p(\{u, v, F_i\}) = 0$. Given a 1-factorization $\mathcal{F}$, the *trains-path* invariant (Gill; Wanless, 2020), which we denote by $I^{tp}_f(\mathcal{F})$, is defined as the vector $(p_0, p_1, \ldots, p_{\ell_{max}})$, where p_ℓ is the number of vertices in $T(\mathcal{F})$ that have $p(u, v, F) = \ell$. Notice that trains-path has size $O(n^3)$. For the train graph in Figure 2.3, the value of the invariant is obtained from $(p_0, p_1) = (6, 12)$. This is because the six vertices that are in a cycle (they have a loop) have $p(u, v, F) = 0$, while the other twelve vertices have $p(u, v, F) = 1$. Table 2.6 presents the value of the trains-path invariant for each one of the six non-isomorphic 1-factorizations of K_8. It shows that, like the previous invariants, trains-path is complete for K_8.

Table 2.6: The trains-path invariant values for the six non-isomorphic 1-factorizations of K_8.

	p_0	p_1	p_2	p_3	p_4	p_5
$I_f^{tp}(\mathcal{F}_0)$	28	168	0	0	0	0
$I_f^{tp}(\mathcal{F}_1)$	28	104	64	0	0	0
$I_f^{tp}(\mathcal{F}_2)$	28	56	80	32	0	0
$I_f^{tp}(\mathcal{F}_3)$	28	72	96	0	0	0
$I_f^{tp}(\mathcal{F}_4)$	28	24	48	48	24	24
$I_f^{tp}(\mathcal{F}_5)$	154	42	0	0	0	0

Proposition 2.5. *The trains-path invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. Recall that the trains graph has $O(n^3)$ vertices and arcs. Besides, every vertex has an outdegree one. The value $p(\{u, v, F\})$ for all the vertices of the train can be computed with a DFS traversal of the trains graph as follows. Starting from an unvisited vertex, the search recursively visits its successor. Whenever a directed cycle is found, the search starts traversing the path backward, labeling the vertices in the cycle and setting their $p(\{u, v, F\})$ values to 0. Besides, for the remaining vertices in the current DFS-tree (that is, in fact, a path) that are not in the cycle, the $p(\{u, v, F\})$ values are determined by adding one unit to that of its DFS-tree descendant stopping in the original unvisited vertex. Then the search continues from another unvisited vertex if there is any. Whenever the traversal reaches a vertex that was already visited before, the search starts to traverse the path backward, setting the value $p(\{u, v, F\})$ as the value of its successor plus one. Recall that the construction of the trains graph takes $O(n^3)$. Thus, as the DFS traversal on the trains graph can be performed in linear time on the size of the graph, that is $O(n^3)$, the whole procedure takes $O(n^3)$. ∎

We remark that Proposition 2.5 answers a question of the authors in Gill and Wanless (2020), that claimed that trains-path could not be obviously computed in cubic time.

2.4.6 Row-cycles and row-cycles-per-row

A *Latin square* of order m is an $m \times m$ array containing m different symbols such that each symbol occurs exactly once in each line and column of the array. A Latin Square L is said to be *symmetric* if $L(i, j) = L(j, i)$ for all $1 \leq i, j \leq m$ and is said to be *unipotent* if $L(i, i) = a$ for all $1 \leq i \leq m$ and some value a. A symmetric and unipotent Latin square provides a natural way to describe a 1-factorization $\mathcal{F}$, and it is denoted by $\mathcal{U}(F)$. The element $L(u, v)$ of a symmetric and unipotent Latin square determines the 1-factor containing the edge uv.

A *Latin rectangle* of L is a matrix in which each symbol occurs exactly once in each row and at most once in each column. A *Latin subrectangle* is a submatrix that is a Latin rectangle. If R is a $2 \times l$-Latin subrectangle of L, and R is minimal in that it does not contain any $2 \times l'$-Latin subrectangle for $2 \leq l' < l$, then, we say that R is a *row cycle* of length l in L (Wanless, 2004; Gill; Wanless, 2020). Table 2.7 shows two Latin subrectangles from $\mathcal{U}(\mathcal{F}_2)$ of K_8. The cells that are double underlined form a 2×2 Latin subrectangle, i.e., a row cycle of length two. On the other hand, the cells that are underlined form a 2×6 Latin subrectangle, i.e., a row cycle of length six.

The last two invariants from Gill and Wanless (2020) are based on the row cycles of a $\mathcal{U}(\mathcal{F})$, which were used to modify 1-factorizations in Kaski et al. (2014). We will refer to these two invariants as row-cycles and row-cycles-per-row.

Table 2.7: Symmetric unipotent Latin square corresponding to 1-factorization $\mathcal{F}_2$ of K_8.

·	$\underline{\underline{1}}$	2	$\underline{\underline{3}}$	4	5	6	7
1	·	3	2	6	7	4	5
2	$\underline{\underline{3}}$	·	$\underline{\underline{1}}$	7	4	5	6
$\underline{3}$	$\underline{2}$	$\underline{1}$	·	$\underline{5}$	$\underline{6}$	$\underline{7}$	4
4	6	7	5	·	1	2	3
5	7	4	6	1	·	3	2
6	4	5	7	2	3	·	1
$\underline{7}$	$\underline{5}$	$\underline{6}$	4	$\underline{3}$	$\underline{2}$	$\underline{1}$	·

2.4.6.1 Row-cycles

Given a 1-factorization $\mathcal{F}$, for each pair of rows of $\mathcal{U}(\mathcal{F})$, determine the Latin subrectangles that can be formed. The *row-cycles* invariant, which we denote by $I_f^r(\mathcal{F})$, is defined as the vector formed by the number of row cycles of each size that can be found in $\mathcal{U}(\mathcal{F})$. Notice that row-cycles has size $\Theta(n)$.

In Table 2.7, the subrectangle in rows 1 and 3 (double underlined) forms a row-cycle of size two, and the subrectangle in lines 4 and 8 (underlined) forms a row-cycle of size six. Table 2.8 displays the value of the row-cycles invariant for each one of the six non-isomorphic 1-factorizations of K_8.

Proposition 2.6. *The row-cycles invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. We consider an auxiliary data structure that stores, for each row and symbol, the column in which that symbol is in the row. For each pair of rows (r_1, r_2), the subrectangles formed from these rows can be obtained in linear time. Starting from the first unvisited column c, look for the symbol in $s = \mathcal{U}(\mathcal{F})(r_1, c)$ and search in constant time, using the auxiliary data structure, the column having symbol s in row r_2. This becomes the new current column. Continue in this way until the symbol stored in the current column of the line r_1 is the starting symbol s. At this point, a subrectangle is found, and it is accounted for based on its size. The procedure continues finding each subrectangle involving r_1 and r_2. ∎

Table 2.8: The row-cycles invariant values for the six non-isomorphic 1-factorizations of K_8.

	size 2	size 3	size 4	size 6
$I_f^r(\mathcal{F}_0)$	84	0	0	0
$I_f^r(\mathcal{F}_1)$	52	0	16	0
$I_f^r(\mathcal{F}_2)$	28	0	16	8
$I_f^r(\mathcal{F}_3)$	36	0	0	16
$I_f^r(\mathcal{F}_4)$	12	8	12	12
$I_f^r(\mathcal{F}_5)$	0	14	0	21

2.4.6.2 Row-cycles-per-row

Given a 1-factorization $\mathcal{F}$, the *row-cycles-per-row* invariant, which we denote by $I_f^{rr}(\mathcal{F})$, is defined as the vector formed by the number of Latin subrectangles of each size that can be found in each line $\mathcal{U}(\mathcal{F})$. The values for each row should be sorted in lexicographical order. Row-cycles-per-row has size $\Theta(n^2)$.

In Table 2.7, the subrectangle in rows 1 and 3 (double underlined) forms a row-cycle of size two accounted for rows 1 and 3, and the subrectangle in rows 4 and 8 (simple underlined) forms a row-cycle of size six accounted for rows 4 and 8. Table 2.9 shows the value of the row-cycles-per-row invariant for each one of the six non-isomorphic 1-factorizations of K_8.

Table 2.9: The row-cycles-per-row invariant values for the six non-isomorphic 1-factorizations of K_8.

	row 0				row 1				row 2				row 3				row 4				row 5				row 6				row 7			
$I_f^{rr}(\mathcal{F}_0)$	21	0	0	0	21	0	0	0	21	0	0	0	21	0	0	0	21	0	0	0	21	0	0	0	21	0	0	0	21	0	0	0
$I_f^{rr}(\mathcal{F}_1)$	13	0	4	0	13	0	4	0	13	0	4	0	13	0	4	0	13	0	4	0	13	0	4	0	13	0	4	0	13	0	4	0
$I_f^{rr}(\mathcal{F}_2)$	7	0	4	2	7	0	4	2	7	0	4	2	7	0	4	2	7	0	4	2	7	0	4	2	7	0	4	2	7	0	4	2
$I_f^{rr}(\mathcal{F}_3)$	9	0	0	4	9	0	0	4	9	0	0	4	9	0	0	4	9	0	0	4	9	0	0	4	9	0	0	4	9	0	0	4
$I_f^{rr}(\mathcal{F}_4)$	3	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3
$I_f^{rr}(\mathcal{F}_5)$	0	2	0	0	0	2	0	6	0	2	0	6	0	2	0	6	0	2	0	6	0	2	0	6	0	2	0	6	0	14	0	6

Proposition 2.7. *The row-cycles-per-row invariant for a given 1-factorization can be computed in $O(n^3)$.*

Proof. This can be accomplished with a slight variation of the procedure described for row-cycles. The only difference is that we have to consider the size of the subrectangle obtained when accounting for the two rows under consideration. ∎

2.5 NEW INVARIANTS FOR 1-FACTORIZATIONS

In this section, we propose two new invariants. Subsection 2.5.1 presents the lantern profiles invariant. Subsection 2.5.2 introduces the even-size bichromatic chains invariant. Subsection 2.5.3 summarizes the sizes and the computational complexities for the invariants described in this tutorial.

2.5.1 Lantern profiles

Given a vertex v and $W \subset N(v)$, define $B(v,W) = \{vw \mid w \in W\}$. Consider two vertices u and v, with $u \neq v$, and $W \subset N(u) \cap N(v) \setminus \{u,v\}$. Let $C(E')$ be the set of colors occurring in $E' \subseteq E$ in a coloring of K_{2n}. Consider the graph formed by $B(u,W) \cup B(v,W)$ and assume a coloring $C(E)$. If $C(B(u,W)) = C(B(v,W))$, with $W \neq \varnothing$ and inclusion-wise minimal for equality to hold, the subgraph wih vertices $\{u,v\} \cup W$ and edges $B(u,W) \cup B(v,W)$ is called a *colorful chordless lantern* $L(u,v,W)$ (Urrutia; de Werra; Januario, 2021). Colorful chordless lanterns are illustrated in Figure 2.4.

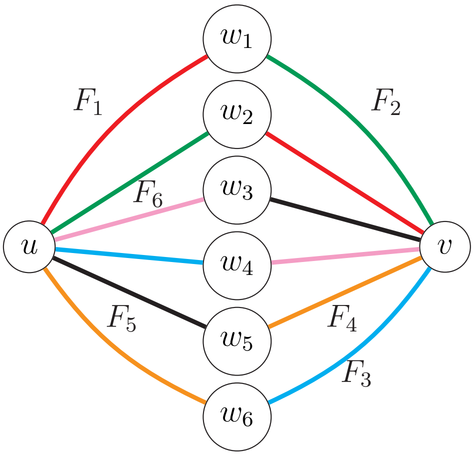


Figure 2.4: Two colorful chordless lanterns $L(u,v,W_1)$ and $L(u,v,W_2)$ associated with a 1-factorization of K_8. The first one has $W_1 = \{w_1, w_2\}$ while the second takes $W_2 = \{w_3, w_4, w_5, w_6\}$.

Let the *degree of a lantern* $l = \Delta(L(u,v,W))$ be the degree of vertex u in the lantern. Note that for each pair of vertices (u,v), the graph formed by $B(u, V \setminus \{u,v\}) \cup B(v, V \setminus \{u,v\})$ is divided by the 1-factorization in a number of lanterns, each of them with some degree $2 \leq l \leq 2n-2$. The sum of the degrees of such lanterns is $2n-2$.

Let $f(w,u,v)$, with $w \neq u$, $w \neq v$, and $u \neq v$, be the degree of the lantern $L(u,v,W)$ with $w \in W$. Let $f_k(w) = |\{(u,v) : f(w,u,v) = k, w \neq u, w \neq v, u \neq v\}|$ be the total number of lanterns of degree k containing the vertex w considering any pair of distinct vertices u and v. Given a 1-factorization $\mathcal{F}$ and an integer k, $2 \leq k \leq 2n-2$, the *lantern profiles* invariant, which we denote by $I_f^{lp}(\mathcal{F})$, is defined as the vector corresponding to the concatenation of the sorted sequences $(f_k(v_{\pi_k(1)}), f_k(v_{\pi_k(2)}), \ldots, f_k(v_{\pi_k(2n)}))$, where $v_{\pi_k(\ell)}$ is the vertex participating in the ℓ^{th} largest number of lanterns of degree k, for all possible values of k. Consequently, if two 1-factorizations have distinct profiles, they are not isomorphic. Table 2.10 shows the value of the lantern profiles invariant for each one of the six non-isomorphic 1-factorizations of K_8. Notice that in the graph K_8, the degree of any lantern will be either two, three, four, or six. Thus, the choice of the value of the parameter k is restricted to $k \in \{2, 3, 4, 6\}$. The values show that lantern profiles is a complete invariant for the 1-factorizations of K_8.

Table 2.10: Lantern profiles invariant values for the six non-isomorphic 1-factorizations of K_8.

	Lanterns with degree 2								Lanterns with degree 3								Lanterns with degree 4								Lanterns with degree 6							
$I_f^{lp}(\mathcal{F}_0)$	21	21	21	21	21	21	21	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$I_f^{lp}(\mathcal{F}_1)$	13	13	13	13	13	13	13	13	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	0	0	0	0	0	0	0	0
$I_f^{lp}(\mathcal{F}_2)$	7	7	7	7	7	7	7	7	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	6	6	6	6	6	6	6	6
$I_f^{lp}(\mathcal{F}_3)$	9	9	9	9	9	9	9	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	12	12	12	12	12	12	12
$I_f^{lp}(\mathcal{F}_4)$	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	6	6	6	6	6	6	6	6	9	9	9	9	9	9	9	9
$I_f^{lp}(\mathcal{F}_5)$	0	0	0	0	0	0	0	0	0	6	6	6	6	6	6	6	0	0	0	0	0	0	0	0	15	15	15	15	15	15	15	21

Proposition 2.8. *The lantern profiles invariant for a given 1-factorization can be computed in* $O(n^3)$.

Proof. For each pair of vertices $\{u, v\}$, one can determine all the lanterns $L(u, v, W)$ from the graph formed by $B(u, V \setminus \{u, v\}) \cup B(v, V \setminus \{u, v\})$ in linear time. Starting from u, traverse the edge colored with a previously unselected color c reaching vertex w_i. Let c_i be the color of the edge $w_i v$. The procedure iteratively traverses the edge uw_j with color c_i and continues computing color c_j as the color of $w_j v$. Whenever $c_j = c$, a complete lantern $L(u, v, W)$ was obtained, and its degree k is accounted for in $f_k(w)$ for every $w \in W$. The procedure thus resumes by starting from an edge with a previously unselected color. Thus, the complete procedure can be implemented to run in $O(n^3)$. ∎

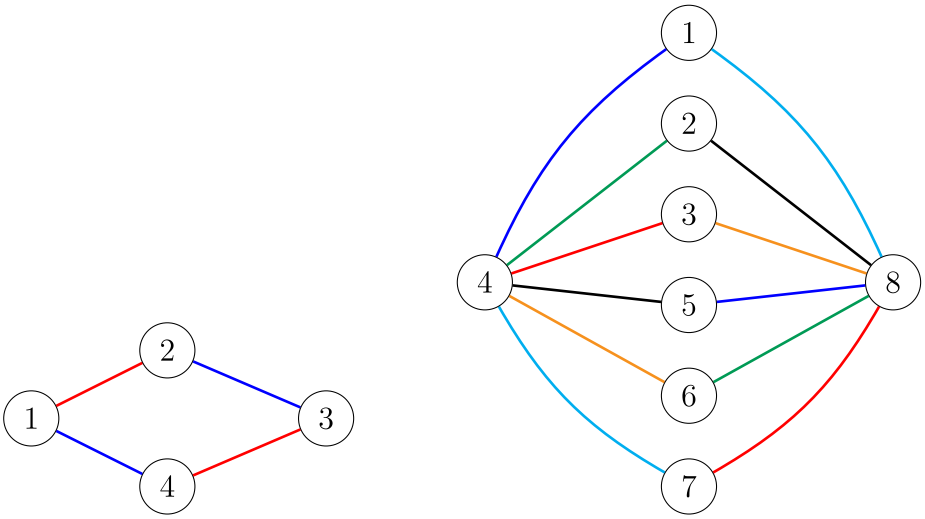
Notice that row-cycles are alternative ways to see lanterns. For instance, the row cycles of length two and six, shown in Table 2.7, correspond to the two lanterns described in Figure 2.5 (a) and Figure 2.5 (b), respectively. The first lantern has degree two, while the second has degree six. Thus, the lantern profiles invariant is strongly related to the row-cycles and row-cycles-per-row invariant. Besides that, the distinguishing capacities of row-cycles-per-row and lantern profiles are very similar, as we will see in the next section.

2.5.2 Even-size bichromatic chains

Let $\mathcal{F}$ be a 1-factorization, and u and v be a pair of distinct vertices. Denote by $q(u, v)$ the number of even-size bichromatic chains connecting u and v. Figure 2.6 illustrates the four even-size bichromatic chains connecting the vertices u and v, implying $q(u, v) = 4$.

Let $q_k = |\{\{u, v\} : q(u, v) = k, \quad u, v \in V\}|$ for $2n - 2 \leq k \leq 2\binom{2n-2}{2}$. The lower bound for k comes from the fact that any of the other $2n - 2$ vertices $w \in V \setminus \{u, v\}$ form the even-size bichromatic chain (u, w, v). The upper bound is defined as twice the number of possible pairs of colors, not taking into account the color of the edge $[u, v]$.

The *even-size bichromatic chains* invariant, which we denote by $I_f^{ec}(\mathcal{F})$, is defined as the vector formed by the sequence of q_k for the possible values of k.



(a) A lantern with degree 2

(b) A lantern with degree 6

Figure 2.5: Two colorful chordless lanterns. The first one (on the left) and the second (on the right) correspond, respectively, to the row cycles of length two and six, illustrated in Table 2.7.

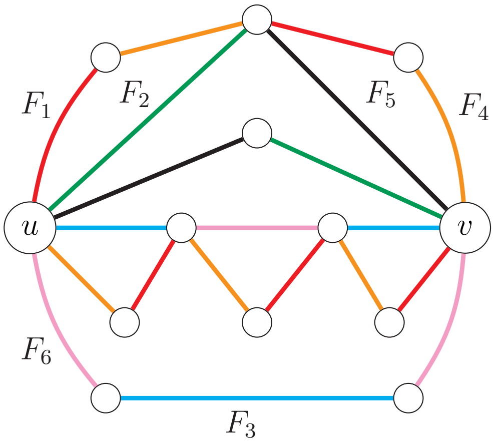


Figure 2.6: Vertices u and v are linked by four even-size bichromatic chains corresponding to the pairs of 1-factors (F_1, F_4) and (F_2, F_5).

Table 2.11 shows the value of the even-size bichromatic chains invariant for each one of the six non-isomorphic 1-factorizations of K_8. It can be noticed that the invariant is complete for the 1-factorizations of K_8.

Table 2.11: Even-size bichromatic chains invariant values for the six non-isomorphic 1-factorizations of K_8.

	6	10	12	14	16	18	22
$I_f^{es}(\mathcal{F}_0)$	28	0	0	0	0	0	0
$I_f^{es}(\mathcal{F}_1)$	8	16	0	0	0	0	4
$I_f^{es}(\mathcal{F}_2)$	0	8	8	4	0	4	4
$I_f^{es}(\mathcal{F}_3)$	0	0	16	12	0	0	0
$I_f^{es}(\mathcal{F}_4)$	0	0	4	0	12	12	0
$I_f^{es}(\mathcal{F}_5)$	0	0	0	0	0	28	0

Proposition 2.9. *The even-size bichromatic chains invariant for a given 1-factorization can be computed in $O(n^4)$.*

Proof. For each pair of colors, consider the resulting graph formed by the edges having these colors. Notice that such a graph is composed of even-size cycles. Thus, for each vertex v, traverse the cycle to which it belongs, and for every other vertex u that is at an even distance from v, take such chain into account for $q(u, v)$. This can be done in $O(n^2)$ for each pair of colors, implying $O(n^4)$. The values of q_k can thus be determined by going through the values of $q(u, v)$ for every pair $\{u, v\}$ in $O(n^2)$ and using a counting technique. Thus, the overall procedure takes $O(n^4)$. ∎

2.5.3 Summary of the sizes, calculation times and classification of the considered invariants

Table 2.12 summarizes the invariants detailed in this tutorial. It provides their sizes, complexities for calculation, and classifications (i.e., what they are based on).

Table 2.12: Invariant sizes, running times for their calculation, and classifications.

Invariant	Size	Calculation time	Classification (based on)
cycle profiles	$\Theta(n^2)$	$O(n^3)$	union of two 1-factors
tricolor vectors	$\Theta(n)$	$O(n^3)$	union of three 1-factors
divisions	$\Theta(d)$	$O(n^d \times nd)$	union of d 1-factors
trains	$\Theta(n)$	$O(n^3)$	trains graph
trains-path	$\Theta(n^3)$	$O(n^3)$	trains graph
row-cycles	$\Theta(n)$	$O(n^3)$	row-cycles / lanterns
row-cycles-per-row	$\Theta(n^2)$	$O(n^3)$	row-cycles / lanterns
lantern profiles	$\Theta(n^2)$	$O(n^3)$	row-cycles / lanterns
even-size bichromatic chains	$\Theta(n^2)$	$O(n^4)$	union of two 1-factors

2.6 EXPERIMENTAL RESULTS

In this section, we summarize the results of the experiments carried out to evaluate the strength of the invariants on different sets of non-isomorphic 1-factorizations found in the literature. The benchmark set consists of

(a) the set of 396 non-isomorphic 1-factorizations of K_{10};

(b) the sets of 5 non-isomorphic P1Fs of K_{12}, 23 non-isomorphic of K_{14}, and 3,155 non-isomorphic of K_{16};

(c) subsets of the 526,915,620 non-isomorphic 1-factorizations of K_{12};

(d) sets of 25,000,000 randomly generated 1-factorizations of K_{16} and K_{20}.

The set of 396 non-isomorphic 1-factorizations of K_{10} can be found in Colbourn and Dinitz (2006) (Table 5.31). The complete set of non-isomorphic 1-factorizations of K_{12} (Dinitz; Garnick; McKay, 1994) was provided by Petteri Kaski. Regarding the set of 23 non-isomorphic P1Fs of K_{14}, twenty were obtained in Seah and Stinson (1987) and the other three in Seah and Stinson (1988) and Dinitz and Garnick (1996). Finally, the set of 3,155 non-isomorphic P1Fs of K_{16} can be downloaded from Wanless (2023).

All the experiments were performed on a machine running under Ubuntu 22.04.1 LTS with an Intel Core i5-9300H 2.40 GHz processor and 8 GB of RAM. The codes were written in C++ and compiled with g++ version 11.3.0, using the options '-O3' and '-std=c++20'.

Given our hardware limitations, we did not run computational experiments to calculate the strength of the invariants on the full set of non-isomorphic 1-factorizations of K_{12}. Therefore, we did not experimentally measure the effort to compute each invariant, nor pairwise compare their values for the considered non-isomorphic 1-factorizations of K_{12}.

Each one of the following subsections considers one of the enumerated benchmark sets. More specifically, Subsection 2.6.1 presents the results of the invariants for the non-isomorphic 1-factorizations of K_{10}. Subsection 2.6.2 shows the results for the non-isomorphic P1Fs of K_{12}, K_{14}, and K_{16}. Subsection 2.6.3 displays the results for the non-isomorphic 1-factorizations of K_{12} and also evaluates the combination of different invariants in an attempt to improve the distinguishing ability. Subsection 2.6.4 presents the results for some randomly generated 1-factorizations of K_{16} and K_{20}.

2.6.1 Non-isomorphic 1-factorizations of K_{10}

Table 2.13 shows the strength of the different invariants to distinguish between the 396 non-isomorphic 1-factorizations of K_{10}. The first column presents the invariant, and the second column the amount (absolute value and percentage) of isomorphism classes distinguished, that is, the strength of this invariant. For instance, the tricolor vectors can classify the 396 non-isomorphic 1-factorizations of K_{10} into 323 isomorphism classes. More specifically, this invariant can distinguish only 262 non-isomorphic 1-factorizations and there are 51 pairs, 8 triples, and 2 sets of four pairwise non-isomorphic 1-factorizations with the same values.

Table 2.13: Strength of the invariants to distinguish non-isomorphic 1-factorizations of K_{10}.

Invariant	Isomorphism classes
cycle profiles	346 (87.4%)
tricolor vectors	323 (81.6%)
divisions	46 (11.6%)
trains	394 (99.5%)
trains-path	**396** (100.0%)
row-cycles	374 (94.4%)
row-cycles-per-row	395 (99.7%)
lantern profiles	395 (99.7%)
even-size bichromatic chains	**396** (100.0%)

The table shows that trains-path and even-size bichromatic chains are complete for the non-isomorphic 1-factorizations of K_{10}. Besides, row-cycles-per-row and lantern profiles can distinguish all but a single pair of non-isomorphic 1-factorizations. Moreover, divisions has very poor performance for this benchmark set, as it can only distinguish 46 isomorphism classes (11.6%). Given the low performance of divisions for K_{10}, it will not be considered in the remaining tests.

2.6.2 Non-isomorphic perfect 1-factorizations of K_{12}, K_{14}, and K_{16}

Table 2.14 shows the strength of the different invariants to distinguish between the five non-isomorphic P1Fs of K_{12}, 23 non-isomorphic P1Fs of K_{14}, and 3,155 non-isomorphic P1Fs of K_{16}. The table does not present values for cycle profiles as it cannot distinguish between P1Fs (see Section 2.4.1).

Table 2.14: Strength of the invariants to distinguish non-isomorphic perfect 1-factorizations of K_{12}, K_{14}, and K_{16}.

Invariant	K_{12}	K_{14}	K_{16}
tricolor vectors	**5**	**23**	2,320
trains	**5**	**23**	3,104
trains-path	**5**	**23**	**3,155**
row-cycles	4	22	**3,155**
row-cycles-per-row	4	22	**3,155**
lantern profiles	4	22	**3,155**
even size bichromatic chains	4	**23**	**3,155**

The table shows that trains-path is complete for the non-isomorphic P1Fs of K_{12}, K_{14}, and K_{16}. Tricolor vectors and trains are complete for the non-isomorphic P1Fs of

K_{12} and K_{14}. Even size bichromatic chains is complete for the non-isomorphic P1Fs of K_{14} and K_{16}. Row-cycles, row-cycles-per-row, and lantern profiles are only complete for the non-isomorphic P1Fs of K_{16}.

2.6.3 Non-isomorphic 1-factorizations of K_{12}

This subsection reports the results for the non-isomorphic 1-factorizations of K_{12}. In the first moment, we consider a preliminary subset of 5,000,000 1-factorizations on which we test all the invariants. Based on the results of this first experiment, we select the strongest invariants to be part of the experiments carried out in the second part of this section.

2.6.3.1 Preliminary subset

Table 2.15 shows the strength of the different invariants to distinguish between 5,000,000 non-isomorphic 1-factorizations of K_{12}. This is a proper subset of the set with 526,915,620 non-isomorphic 1-factorizations of K_{12}. The first column presents the invariant, and the second column the amount (absolute value and percentage) of isomorphism classes distinguished.

Table 2.15: Strength of the invariants to distinguish 5,000,000 non-isomorphic 1-factorizations of K_{12}.

Invariant	Isomorphism classes
cycle profiles	2,984,500 (59.690%)
tricolor vectors	283,044 (05.661%)
trains	1,698,355 (33.967%)
trains-path	4,999,812 (99.996%)
row-cycles	3,371,571 (67.431%)
row-cycles-per-row	4,999,564 (99.991%)
lantern profiles	4,999,624 (99.992%)
even-size bichromatic chains	4,999,375 (99.987%)

The table shows that trains-path, row-cycles-per-row, lantern profiles, and even-size bichromatic chains are much stronger than the others. All of them distinguish more than 99.9% of the isomorphism classes, while the others distinguish less than 68% of them. For that reason, in the following experiments, we only show the results for these four strongest invariants and their combinations.

2.6.3.2 Larger sets of non-isomorphic 1-factorizations

Table 2.16 shows the average strength of the selected invariants to distinguish between 5,000,000 non-isomorphic 1-factorizations for 105 disjoint subsets of non-isomorphic 1-factorizations of K_{12}. Notice that, in this way, 525,000,000 (almost all of the 526,915,620) non-isomorphic 1-factorizations of K_{12} are considered in this experiment.

Table 2.16: Average strength of the invariants to distinguish 5,000,000 non-isomorphic 1-factorizations of K_{12}, considering 105 sets of 5,000,000.

Invariant	Isomorphism classes (avg)
trains-path	4,999,992.428
row-cycles-per-row	4,999,979.371
lantern profiles	4,999,983.695
even-size bichromatic chains	4,998,103.457

The table shows that trains-path is slightly stronger than the other invariants. Among the invariants with guaranteed quadratic size, lantern profiles was the strongest, followed by row-cycles-per-row.

The next experiment considers 21 subsets of 25,000,000 non-isomorphic 1-factorizations of K_{12}. Table 2.17 shows that the strength order of the invariants is preserved from the previous experiment. The four invariants identify, on average, more than 99.8% of the isomorphism classes.

Table 2.17: Average strength of the invariants to distinguish 25,000,000 non-isomorphic 1-factorizations of K_{12}, considering 21 sets of 25,000,000.

Invariant	Isomorphism classes (avg)
trains-path	24,999,964.75
row-cycles-per-row	24,999,877.15
lantern profiles	24,999,902.55
even-size bichromatic chains	24,955,330.70

To measure the significance difference between the strength of the four invariants, a pairwise statistical comparison was performed using the Wilcoxon test (Demsar, 2006) at a significance level of 1%. The Wilcoxon test is a non-parametric statistical test that ranks the differences in performances of two classifiers for each data set and compares the ranks for the differences. Here, the classifiers are the invariants, and the data set are the strength values obtained by each invariant considering 21 sets of 25,000,000 non-isomorphic 1-factorizations. The Wilcoxon test indicates if there is any statistical difference between each analyzed set. It was found that there is a statistical difference between even-size bichromatic chains and the other invariants, and there is a statistical difference between the row-cycles-per-row and lantern profiles invariants, because p-value < 0.01. There is no statistical difference for the remaining pairwise comparisons, because p-value > 0.01.

2.6.3.3 How can the combination of invariants improve the distinguishing strength?

We now analyze the strength of different invariant combinations. We consider all the possible combinations of the four selected invariants. Given a 1-factorization $\mathcal{F}$ and two invariants I_f^a and I_f^b, the combination of these invariants consists of the joint set of their values, denoted by $I_f^a(\mathcal{F}) \oplus I_f^b(\mathcal{F})$.

Table 2.18 shows the average strength of the different combinations of the invariants to distinguish between 5,000,000 non-isomorphic 1-factorizations for 105 disjoint subsets of non-isomorphic 1-factorizations of K_{12}.

Table 2.18: Average strength of the invariant combinations to distinguish 5,000,000 non-isomorphic 1-factorizations of K_{12}, considering 105 sets of 5,000,000.

Invariant combination	Isomorphism classes (avg)
trains path ⊕ row-cycles-per-row	4,999,999.723
trains path ⊕ lantern profiles	4,999,999.809
trains path ⊕ even-size bichromatic chains	4,999,999.885
row-cycles-per-row ⊕ lantern profiles	4,999,983.742
row-cycles-per-row ⊕ even-size bichromatic chains	4,999,999.685
lantern profiles ⊕ even-size bichromatic chains	4,999,999.657
trains path ⊕ row-cycles-per-row ⊕ lantern profiles	4,999,999.809
trains path ⊕ row-cycles-per-row ⊕ even-size bichromatic chains	4,999,999.942
trains path ⊕ lantern profiles ⊕ even-size bichromatic chains	4,999,999.942
row-cycles per row ⊕ lantern profiles ⊕ even-size bichromatic chains	4,999,999.685
trains path ⊕ row-cycles-per-row ⊕ lantern profiles ⊕ even-size bichromatic chains	4,999,999.942

The table shows that although the even-size bichromatic chains invariant was shown to be the weakest of the invariants individually, when combined with a strong complementary invariant such as trains-path, it is able to obtain promising results. Notice that the combination of row-cycles-per-row with lantern profiles is slightly weaker than the combination of any other two invariants. This shows that contrary to what happens with trains-path and even-size bichromatic chains, the invariants are strongly related, and their distinguishing capacities seem very similar. The combination of more than two invariants is not significantly stronger than the best combinations of two invariants.

To measure the significance difference between the strength of all combinations of invariants (described in Table 2.18), we performed a pairwise statistical comparison using the Wilcoxon test at a significance level of 1%. The results indicated that there is a statistical difference only between the combination *row-cycles-per-row ⊕ lantern profiles* and the other combinations. For the remaining pairwise comparisons, there is no statistical difference.

Given our hardware limitations, we do not analyze the combinations of invariants for the 21 subsets of 25,000,000 non-isomorphic 1-factorizations of K_{12}.

2.6.4 Randomly generated 1-factorizations of K_{16} and K_{20}

In this experiment, we used a randomized variant of Vizing's algorithm (Vizing, 1964; Misra; Gries, 1992) to randomly generate sets of 25,000,000 1-factorizations of K_{16} and K_{20}. The results in Table 2.19 show that the four selected invariants are able to distinguish between all the 25,000,000 1-factorization for both sets.

Table 2.19: Strength of the invariants to distinguish 25,000,000 randomly generated 1-factorizations of K_{16} and K_{20}.

Invariant	K_{16}	K_{20}
trains-path	25,000,000	25,000,000
row-cycles-per-row	25,000,000	25,000,000
lantern profiles	25,000,000	25,000,000
even size bichromatic chains	25,000,000	25,000,000

2.7 CONCLUDING REMARKS

We analyzed invariants for 1-factorizations of K_{2n}. We described seven of the main invariants available in the literature (cycle profiles, tricolor vectors, divisions, trains, trains-path, row-cycles, and row-cycles-per-row). Furthermore, we proposed two new invariants, denoted lantern profiles and even-size bichromatic chains. For all the nine invariants presented, we analyzed their size and computational complexity.

Furthermore, we performed experiments to evaluate the strength of the invariants. We used a benchmark set composed of the non-isomorphic 1-factorizations of K_{10}, the non-isomorphic P1Fs of K_{12}, K_{14}, and K_{16}, and subsets of the non-isomorphic 1-factorizations of K_{12}. The results show that trains-path and even-size bichromatic chains are complete for the non-isomorphic 1-factorizations of K_{10}. Besides, only trains-path is complete for all the three sets of non-isomorphic P1Fs considered. Moreover, four of the invariants (trains-path, row-cycles-per-row, lantern profiles, and even-size bichromatic chains) have shown to be much stronger than the others when considering the sets of non-isomorphic 1-factorizations of K_{12} tested. Last but not least, the strengths of the combinations of invariants were also evaluated, showing a complementarity in their distinguishing abilities.

To measure the statistical significance between the strength of the four strongest invariants it was performed a pairwise statistical comparison. The statistical analyses showed that there is a statistical difference between even-size bichromatic chains and the other three invariants, and there is a statistical difference between the row-cycles-per-row and lantern profiles invariants. For all combinations of invariants, it was also performed a pairwise statistical comparison. The results showed a statistical difference only between the combination *row-cycles-per-row* $\oplus$ *lantern profiles* and the other combinations.

Chapter 3

THE GENERALIZED PARTIAL TEAM SWAP NEIGHBORHOOD: ALGORITHMIC AND COMPUTATIONAL ASPECTS

The concept of 1-factorization is of great interest due to its applications in modeling sports tournaments. Different neighborhood structures have been used in local search procedures for tournament scheduling problems. These neighborhood structures can be associated with partial modifications of a given 1-factorization. The resulting 1-factorization represents a neighbor of the current 1-factorization in the neighborhood structure under consideration.

This chapter introduces algorithms to explore the *Generalized Partial Team Swap* (GPTS) neighborhood, a neighborhood structure recently proposed in the literature. In this regard, this chapter presents a study on algorithmic and computational aspects of this neighborhood. Furthermore, we provide a discussion on how this neighborhood structure increases the connectivity of the search space defined by non-isomorphic 1-factorizations of K_{2n} (for $8 \leq 2n \leq 12$) when compared to other neighborhood structures. Finally, we present preliminary computational experiments that were conducted to evaluate the performance of the GPTS, having the *Weighted Carry-Over Effects Value* (WCOEV) Minimization Problem as a case study.

3.1 INTRODUCTION

Several sports tournaments involving $2n$ teams are organized as *Single Round-Robin* (SRR) tournaments in which teams face each other once. In such a tournament, there is a sequence of $2n - 1$ rounds, with each team playing once in each round. In a basic sports scheduling problem, one has to determine in which round each pair of teams will face each other. It is natural to model such a problem as a 1-factorization of a graph K_{2n}, with each vertex representing a team and each edge representing the game between the teams corresponding to its endpoints. In this way, a schedule for such a tournament can be determined by computing a 1-factorization of the graph such that each one of its

1-factors represents a round of the schedule. Figure 3.1 illustrates a SRR schedule with six teams (vertices) and five rounds (1-factors), represented by a 1-factorization of K_6.

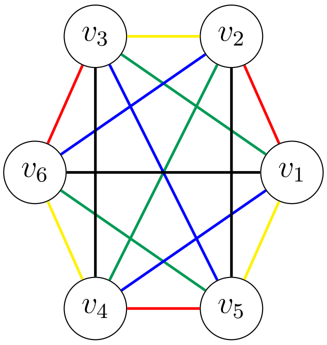


Figure 3.1: A 1-factorization $\mathcal{F} = \{F_1, F_2, F_3, F_4, F_5\}$ of K_6, with $F_1 = \{v_1v_2, v_3v_6, v_4v_5\}$, $F_2 = \{v_1v_3, v_2v_4, v_5v_6\}$, $F_3 = \{v_1v_4, v_2v_6, v_3v_5\}$, $F_4 = \{v_1v_5, v_2v_3, v_4v_6\}$, and $F_5 = \{v_1v_6, v_2v_5, v_3v_4\}$.

The GPTS neighborhood was introduced by Januario et al. (2016). As pointed out by Ribeiro, Urrutia and de Werra (2023), due to the size and complex structure of GPTS neighborhood, it might be hard to design algorithms that systematically explore this neighborhood. By taking this challenge into consideration, this chapter introduces algorithms to explore the GPTS neighborhood and presents a study on algorithmic and computational aspects of this neighborhood.

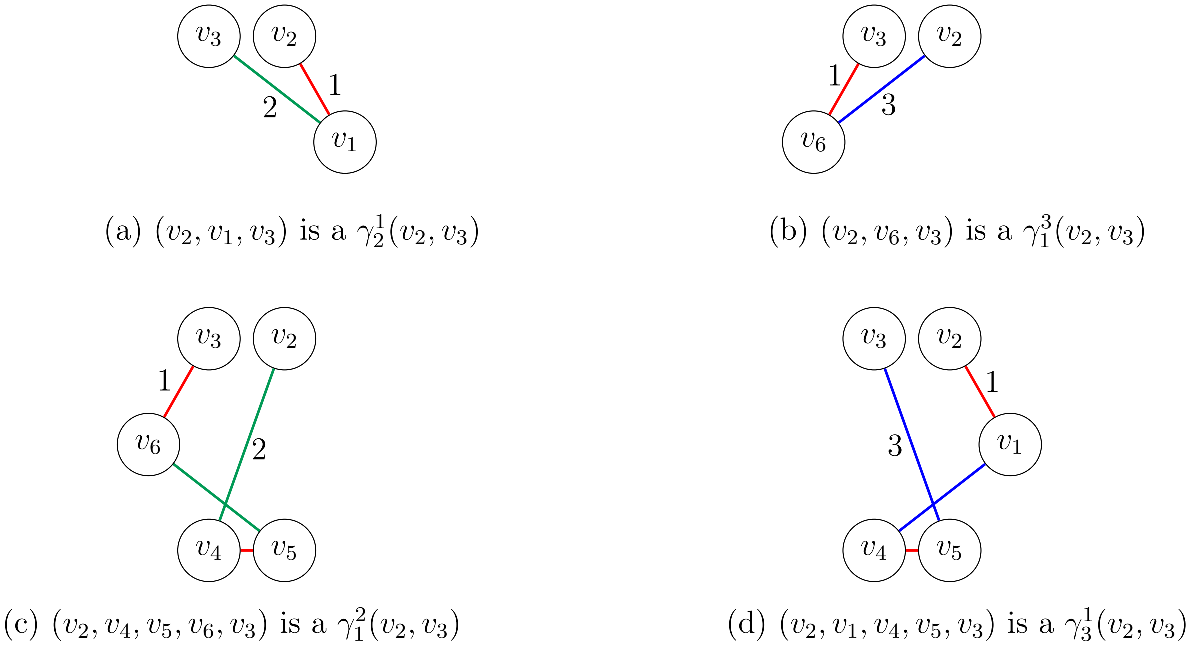
3.1.1 Basic definitions

Compatible chains, compatible set, and balanced set

Two bichromatic chains are said to be *compatible* if they are edge-disjoint; otherwise, they are said to be *incompatible*. A set of bichromatic chains is said to be a *compatible set* if all its chains are pairwise compatible; otherwise, it is said to be an *incompatible set*. A *balanced set* is a set in which each of the $2n-1$ colors of the graph K_{2n} is present on the edges of exactly two or zero bichromatic chains. A set of bichromatic chains that is both balanced and compatible is said to be a *compatible balanced set*. A α/β-chain (or γ_β^α) is a bichromatic chain in which its edges are alternately colored with two different colors α and β, starting from α. A $\gamma_\beta^\alpha(u, v)$ is a γ_β^α that links two vertices u and v, starting from u. These concepts are illustrated in Example 1.

Example 1. Consider the colored graph K_6 in Figure 3.1. The vertices v_2 and v_3 are linked by the following eight even-size bichromatic chains: $\gamma_2^1(v_2, v_3), \gamma_1^2(v_2, v_3), \gamma_3^1(v_2, v_3), \gamma_1^3(v_2, v_3), \gamma_5^2(v_2, v_3), \gamma_2^5(v_2, v_3), \gamma_5^3(v_2, v_3)$, and $\gamma_3^5(v_2, v_3)$. Figure 3.2 illustrates four of these chains. Among these, the chains $\gamma_2^1(v_2, v_3)$ (Figure 3.2 (a)) and $\gamma_1^3(v_2, v_3)$ (Figure 3.2 (b)) are compatible. On the other hand, the chains $\gamma_1^2(v_2, v_3)$ (Figure 3.2 (c)) and $\gamma_3^1(v_2, v_3)$ (Figure 3.2 (d)) are incompatible, because they have a common edge $e = v_4, v_5$.

Note that, from these eight even-size bichromatic chains, it is possible to construct five compatible balanced sets, namely: $\{\gamma^1_2(v_2,v_3), \gamma^2_1(v_2,v_3)\}$, $\{\gamma^1_2(v_2,v_3), \gamma^2_5(v_2,v_3), \gamma^5_3(v_2,v_3), \gamma^3_1(v_2,v_3)\}$, $\{\gamma^1_3(v_2,v_3), \gamma^3_1(v_2,v_3)\}$, $\{\gamma^2_5(v_2,v_3), \gamma^5_2(v_2,v_3)\}$, and $\{\gamma^3_5(v_2,v_3), \gamma^5_3(v_2,v_3)\}$. △



(a) (v_2, v_1, v_3) is a $\gamma^1_2(v_2, v_3)$

(b) (v_2, v_6, v_3) is a $\gamma^3_1(v_2, v_3)$

(c) $(v_2, v_4, v_5, v_6, v_3)$ is a $\gamma^2_1(v_2, v_3)$

(d) $(v_2, v_1, v_4, v_5, v_3)$ is a $\gamma^1_3(v_2, v_3)$

Figure 3.2: Four even-size bichromatic chains linking the vertices v_2 and v_3 of the edge colored graph K_6 illustrated in Figure 3.1.

In the remainder of the chapter, we denote bichromatic chains by chains, unless stated otherwise.

Search space, neighborhood, and local search

The *search space* of an optimization problem can be defined by a directed graph $U = (S, A)$, where the set of vertices S corresponds to the solutions of the problem and the set of arcs A corresponds to the *move operators* used to generate new neighborhood solutions. There is an arc between solutions $s_a, s_b \in S$, if the solution s_b can be generated from the solution s_a using a move operator. A search space is connected if, for any pair of solutions s_a and s_b, it is possible to generate s_b from s_a, using a finite number of move operators.

The *non-isomorphic 1-factorization search space* can be modeled through a graph $\mathcal{G}^M_{2n}$ in which each vertex of this graph represents a 1-factorization belonging to an isomorphism class of 1-factorizations of K_{2n}. Each edge that connects the vertices $\mathcal{F}_a$ and $\mathcal{F}_b$ corresponds to a move operator M, which from a 1-factorization isomorphic to $\mathcal{F}_a$ generates a 1-factorization isomorphic to $\mathcal{F}_b$.

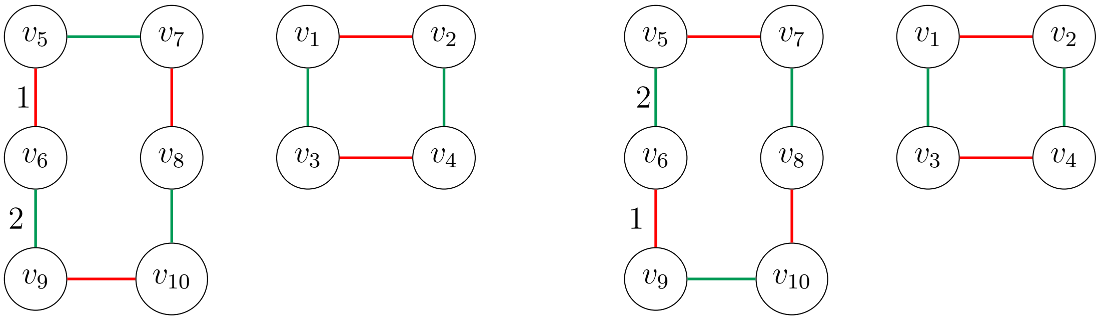
A *neighborhood structure* $\mathcal{V}$ is a mapping that assigns to each solution $s \in S$ a set of neighboring solutions $\mathcal{V}(s)$ that are neighbors of s. Each neighboring solution s' is generated from a move in the neighborhood $\mathcal{V}$ that performs a modification in the structure of the solution s. A neighborhood is connected if it is possible to move from any solution to any other with a finite number of moves using the neighborhood.

Local search is a simple heuristic method that uses the concept of neighborhoods to move from one solution s to a neighbor solution $s' \in \mathcal{V}(s)$ (Talbi, 2009). The local search starts from a given initial solution s associated with a cost function denoted by $f(s)$. Then, at each iteration, the current solution is replaced by a neighbor solution that improves the cost function. When all the neighbors are worse than the current solution, it means that a local optimum has been found. One of the most important aspects of local search, widely used in optimization problems, it is the definition of its neighborhood structure.

PRS, PTS, and GPTS neighborhood structures

In the following, all neighborhood structures are modeled as operators over 1-factorizations of K_{2n}. Each neighborhood structure can be associated with partial modifications of a given 1-factorization. The resulting 1-factorization represents a neighbor of the current 1-factorization in the neighborhood structure under consideration.

To perform a move in the *Partial Round Swap* (PRS) neighborhood structure, select two distinct colors and consider a cycle in the subgraph induced by the edges colored with these two colors. Then, exchange the color assignment of edges in the cycle, leading to a neighbor coloring. Figure 3.3 illustrates a move in the PRS neighborhood, in which the color of the edges in the cycle formed by vertices v_5, v_6, v_7, v_8, v_9, and v_{10} have been exchanged (Januario et al., 2016). Note that, if this cycle is a *Hamiltonian Cycle* (HC), then the neighbor coloring is isomorphic to the original one, since the recoloring is equivalent to exchange two 1-factors.



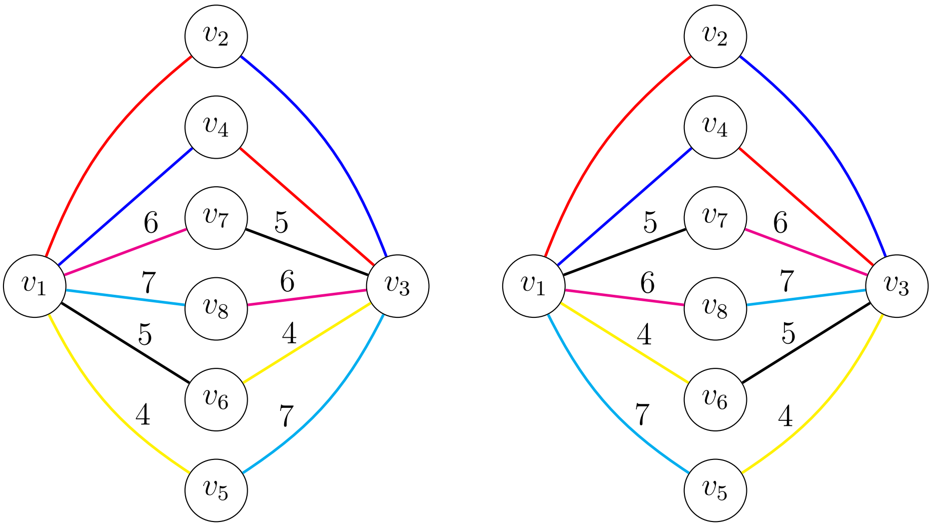
(a) Subgraph before a PRS move

(b) Subgraph after a PRS move

Figure 3.3: Illustration of a PRS move: (a) a subgraph induced by edges with colors 1 and 2, obtained from a 1-factorization of K_{10}; (b) the same subgraph after a PRS move, by exchanging the color assignment of edges in the 6-cycle.

In order to obtain a neighbor in a *Partial Team Swap* (PTS) neighborhood, we select a set $\mathcal{A}$ of edge-disjoint chains of length two between two vertices u and v of K_{2n}. In addition, the set Υ of colors assigned to the edges of the chains in $\mathcal{A}$ adjacent to u is the same set of colors assigned to the edges of the chains in $\mathcal{A}$ adjacent to v. We choose Υ to be minimal, i.e., no proper subset of $\mathcal{A}$ satisfies this same condition. Next, exchanging the color assignment of edges in each chain, leading to a neighbor coloring (Ribeiro; Urrutia; de Werra, 2023).

Figure 3.4 illustrates a move in the PTS neighborhood with the set of chains $\{\gamma_5^6(v_1, v_3)$, $\gamma_6^7(v_1, v_3)$, $\gamma_4^5(v_1, v_3)$, $\gamma_7^4(v_1, v_3)\}$. Note that, if the number of chains is equal to $2n-2$, then the neighbor coloring is isomorphic to the original one, since the recoloring is equivalent to exchange the labels of vertices u and v.

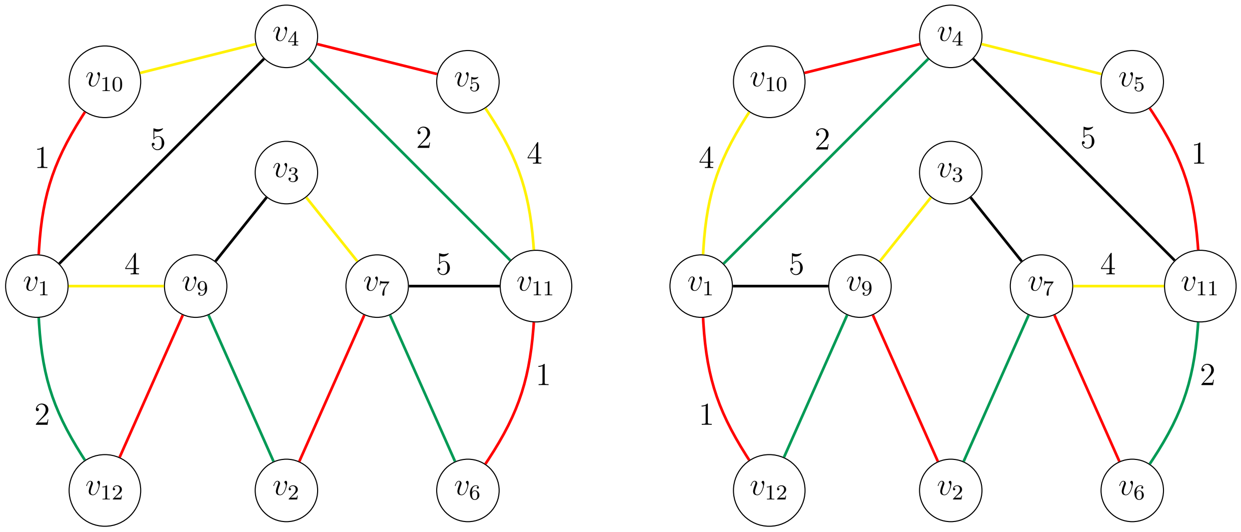


(a) Subgraph before a PTS move (b) Subgraph after a PTS move

Figure 3.4: Illustration of a PTS move: (a) a subgraph $K_{2,6}$ obtained from a 1-factorization of K_8; (b) the same subgraph after a PTS move using the set of chains $\{\gamma_5^6(v_1, v_3)$, $\gamma_6^7(v_1, v_3)$, $\gamma_4^5(v_1, v_3)$, $\gamma_7^4(v_1, v_3)\}$.

Generalized Partial Team Swap (GPTS) neighborhood involves a set of edge-disjoint chains of even length between two vertices u and v of K_{2n}. As in PTS, the set of colors on edges of these chains incident to vertex u is the same set of colors on edges incident to v. Finally, exchanging the color assignment of edges in each chain gives a neighbor coloring (Januario et al., 2016).

Figure 3.5 illustrates a move in the GPTS neighborhood with the set of chains $\{\gamma_4^1(v_1, v_{11})$, $\gamma_2^5(v_1, v_{11})$, $\gamma_5^4(v_1, v_{11})$, $\gamma_1^2(v_1, v_{11})\}$. In the subgraph on the left, $\gamma_4^1(v_1, v_{11})$ is a 4-chain, $\gamma_2^5(v_1, v_{11})$ is a 2-chain, $\gamma_5^4(v_1, v_{11})$ is a 4-chain, and $\gamma_1^2(v_1, v_{11})$ is a 6-chain.



(a) Subgraph before a GPTS move

(b) Subgraph after a GPTS move

Figure 3.5: Illustration of a GPTS move: (a) a subgraph obtained from a 1-factorization of K_{12}; (b) the resulting subgraph after a GPTS move, using the set of chains $\{\gamma_4^1(v_1, v_{11})$, $\gamma_2^5(v_1, v_{11})$, $\gamma_5^4(v_1, v_{11})$, $\gamma_1^2(v_1, v_{11})\}$.

3.1.2 Contributions and organization

The main contribution of this chapter is a study of algorithmic and computational aspects of the GPTS neighborhood. First, we present algorithms to systematically explore this neighborhood. After that, we provide a discussion on how this neighborhood structure increases the connectivity of the non-isomorphic 1-factorization search space of K_{2n} (for $8 \leq 2n \leq 12$) when compared to the PRS and PTS neighborhood structures. Finally, we conduct preliminary computational experiments to evaluate the performance of PRS, PTS, and GPTS, having the WCOEV Minimization Problem as a case study.

The remainder of this chapter is organized as follows. Section 3.2 reviews the related literature. Section 3.3 discusses the algorithmic aspects of exploring the GPTS neighborhood. Section 3.4 presents a discussion analyzing the structure of the neighborhoods PRS, PTS, and GPTS for non-isomorphic 1-factorizations. Section 3.5 presents preliminary computational results.

3.2 LITERATURE REVIEW

This section reviews some of the literature on Sport Scheduling Problems and search methods on 1-factorizations.

3.2.1 Applications in sports scheduling

The study of the relationship between single round-robin scheduling problems and 1-factorizations of complete graphs began in the 1970s with the work of Gelling (1973). Several types of 1-factorizations can be constructed based on specific requirements of sports leagues (de Werra, 1980; de Werra, 1982; de Werra, 1985; Geinoz; Ekim; de Werra, 2008). In the sports scheduling literature, some of the common solution approaches are: integer programming (Briskorn; Drexl, 2009; Durán et al., 2019), constraint programming (Trick, 2001; Russell; Urban, 2006), and local search metaheuristics such as simulated annealing (Anagnostopoulos et al., 2006; Lim; Rodrigues; Zhang, 2006), tabu search (Hamiez; Hao, 2000; di Gaspero; Schaerf, 2007), and iterated local search (Costa; Urrutia; Ribeiro, 2012).

The traveling tournament problem (TTP) is one of the most studied problems in sports scheduling (Easton; Nemhauser; Trick, 2001). Its goal consists in minimizing the total distance traveled by the teams throughout a double round-robin tournament with some additional constraints. Single round-robin variants of the TTP are studied in the literature, such as the mirrored TTP (Ribeiro; Urrutia, 2007) and the TTP with predefined venues (Melo; Urrutia; Ribeiro, 2009; Costa; Urrutia; Ribeiro, 2012).

Several authors presented literature reviews about sports scheduling problems, their different solution approaches, and applications (Rasmussen; Trick, 2008; Kendall et al., 2010; Ribeiro, 2012; Durán, 2021). A recent book on the subject was written by Ribeiro, Urrutia and de Werra (2023). In general, sports scheduling problems tend to be NP-hard, and they deal with a large solution space related to different 1-factorizations of K_{2n}. Consequently, heuristics are often used in practice.

3.2.2 Search methods on 1-factorizations

Different neighborhood structures have been used in local search procedures for sports scheduling problems (Anagnostopoulos et al., 2006; di Gaspero; Schaerf, 2007; Ribeiro; Urrutia, 2007; Costa; Urrutia; Ribeiro, 2012; Januario; Urrutia, 2016). Januario et al. (2016) describe, from an edge coloring perspective, four neighborhoods commonly used in local search heuristics for sports scheduling problems: *Round Swap*, *Team Swap*, PRS, and PTS. Kaski et al. (2014) defined two types of *switching operations* between 1-factorizations of complete graphs, called *factor-switching* and *vertex-switching*. These operations allow converting a 1-factorization $\mathcal{F}$ of K_{2n} into a different 1-factorization $\mathcal{F}'$ of the same graph. A factor-switching operation is equivalent to a PRS move and a vertex-switching operation is equivalent to a PTS move.

It is known that these neighborhoods do not fully connect the solution space, as the solutions obtained with the neighborhoods belong to a proper subset of the set of non-isomorphic 1-factorizations of K_{2n}. Costa, Urrutia and Ribeiro (2012) investigated the connectivity of the solution space of single round-robin tournaments. They established that the solution space is not connected by the Team swap, Round swap, PTS, and PRS neighborhood structures for tournaments with 20 teams, since these neighborhoods are not able to escape from 1-factorizations that are isomorphic to the 1-factorization generated by the circle method (Kirkman, 1847). Later, Januario and Urrutia (2015) extended the research and showed that these neighborhoods do not connect the solution space for several values of $2n \leq 100$. Their study showed that when $2n = p+1$ (with p being a prime number) PRS neighborhood is not connected and if the initial 1-factorization is built with the circle method, it is impossible to move to other non-isomorphic 1-factorization with a PRS move. Januario, Urrutia and de Werra (2016) proved the conjecture introduced by Januario and Urrutia (2015), characterizing the cases in which one can escape from 1-factorizations generated by the circle method using PTS neighborhood. Kaski et al. (2014) studied the connectivity of the non-isomorphic 1-factorization search space under a so-called switching operation. The study refers to the connectivity of non-isomorphic 1-factorizations of K_8, K_{10}, and K_{12}.

3.3 ALGORITHMIC ASPECTS FOR EXPLORING THE GPTS NEIGHBORHOOD

In this section, we discuss the algorithmic aspects for exploring the GPTS neighborhood. The exploration of the GPTS neighborhood structure consists of three phases: *selection*, *construction*, and *change*.

Consider a complete graph K_{2n} together with a proper edge coloring c and two distinct vertices $u, v \in K_{2n}$. First, the selection phase constructs a list $\mathcal{L}^{comp}$ with all the pairs of compatible even-size chains linking u and v. Given $\mathcal{L}^{comp}$ as input, the construction phase builds a list $\mathcal{L}^{bal}$ of compatible balanced sets. The change phase generates neighbor colorings by modifying the color assignments in the subgraphs corresponding to each of the compatible balanced sets in $\mathcal{L}^{bal}$. Finally, the change phase returns the list $\mathcal{S}$ of neighbor colorings. In what follows, we detail the selection phase in Subsection 3.3.1. In Subsection 3.3.2, we describe the construction phase and propose two possible strategies. Last, in Subsection 3.3.3, we explain the change phase.

3.3.1 The selection phase

Given a complete graph K_{2n}, a proper edge coloring c, and two distinct vertices $u, v \in K_{2n}$, the selection phase builds a list with all the pairs of compatible even-size chains linking u and v. Algorithm 1 details the procedure. Initially, lists $\mathcal{L}^{even}$ and $\mathcal{L}^{comp}$ are initialized as empty in line 1. After that, for each pair of distinct colors (α, β), it is checked whether there is an even size chain $\gamma^{\alpha}_{\beta}(u, v)$. If so, such a chain is inserted into $\mathcal{L}^{even}$ (line 4). Then, the pairs of distinct chains belonging to $\mathcal{L}^{even}$ that are compatible are inserted into $\mathcal{L}^{comp}$ (lines 5-7). Finally, the algorithm ends by returning the list $\mathcal{L}^{comp}$ containing all the pairs of compatible chains that link u and v (line 8). Example 2 illustrates this phase.

Example 2. Consider as input for the selection phase the graph K_6 of Figure 3.1 with the coloring corresponding to the depicted 1-factorization and the vertices v_2 and v_3. This implies that this phase will return an $\mathcal{L}^{comp}$ list containing the following pairs of compatible chains: $\{\gamma^1_2(v_2, v_3), \gamma^2_1(v_2, v_3)\}$, $\{\gamma^1_2(v_2, v_3), \gamma^3_1(v_2, v_3)\}$, $\{\gamma^1_2(v_2, v_3), \gamma^2_5(v_2, v_3)\}$, $\{\gamma^1_2(v_2, v_3), \gamma^3_5(v_2, v_3)\}$, $\{\gamma^1_2(v_2, v_3), \gamma^5_3(v_2, v_3)\}$, $\{\gamma^2_1(v_2, v_3), \gamma^3_5(v_2, v_3)\}$, $\{\gamma^2_1(v_2, v_3), \gamma^5_3(v_2, v_3)$ $\}$, $\{\gamma^1_3(v_2, v_3), \gamma^3_1(v_2, v_3)\}$, $\{\gamma^1_3(v_2, v_3), \gamma^2_5(v_2, v_3)\}$, $\{\gamma^1_3(v_2, v_3), \gamma^5_2(v_2, v_3)\}$, $\{\gamma^3_1(v_2, v_3), \gamma^2_5(v_2, v_3)\}$, $\{\gamma^3_1(v_2, v_3), \gamma^5_2(v_2, v_3)\}$, $\{\gamma^3_1(v_2, v_3), \gamma^5_3(v_2, v_3)\}$, $\{\gamma^2_5(v_2, v_3), \gamma^5_2(v_2, v_3)\}$, $\{\gamma^2_5(v_2, v_3), \gamma^5_3$ $(v_2, v_3)\}$, $\{\gamma^3_5(v_2, v_3), \gamma^5_3(v_2, v_3)\}$. △

```
Algorithm 1: SELECTION-PHASE
  Input  : graph $K_{2n}$, coloring $c$, and two distinct vertices $u, v \in K_{2n}$
  Output: list $\mathcal{L}^{comp}$ with all the pairs of compatible even-size chains linking $u$
          and $v$
  $\mathcal{L}^{even} \leftarrow \mathcal{L}^{comp} \leftarrow \varnothing$;
  foreach $(\alpha, \beta) \in \mathcal{C}$ do
      if there is an $\gamma^{\alpha}_{\beta}(u, v)$ then
          Add $\gamma^{\alpha}_{\beta}(u, v)$ to $\mathcal{L}^{even}$;
  foreach $(\gamma^{\alpha_1}_{\beta_1}(u, v), \gamma^{\alpha_2}_{\beta_2}(u, v)) \in \mathcal{L}^{even}$ do
      if $\gamma^{\alpha_1}_{\beta_1}(u, v)$ and $\gamma^{\alpha_2}_{\beta_2}(u, v)$ are compatible then
          Add $(\gamma^{\alpha_1}_{\beta_1}(u, v), \gamma^{\alpha_2}_{\beta_2}(u, v))$ to $\mathcal{L}^{comp}$;
  return $\mathcal{L}^{comp}$;
```

3.3.2 The construction phase

The construction phase builds different compatible balanced sets based on an auxiliary graph, given as input a list $\mathcal{L}^{comp}$ of pairs of compatible chains obtained in the selection phase. We describe the auxiliary graph in Subsection 3.3.2.1 and present two possible strategies to be used in this phase in Subsection 3.3.2.2.

3.3.2.1 Graph H

The auxiliary graph H is obtained as follows. Its vertex set $V(H)$ is composed of a vertex for each color $\alpha \in \mathcal{C}$ such that there is a chain in $\mathcal{L}^{comp}$ with color α. Its edge set $E(H)$ is built as follows: there is an edge $\alpha\beta$ if and only if there is a pair of chains $(\gamma^{\alpha}_{\beta}(u, v), \gamma^{\beta}_{\alpha}(u, v)) \in \mathcal{L}^{comp}$. Example 3 illustrates the construction of H.

Example 3. To construct the graph H illustrated in Figure 3.6, consider the list $\mathcal{L}^{comp}$ containing all the pairs of compatible chains linking the vertices v_2 and v_3 of the edge colored graph K_6 in Figure 3.1. First, for each of the colors found in these chains, we have a vertex labeled by this color in the graph H, i.e., the colors in $\{1, 2, 3, 5\}$. Next, for each pair of chains that have the same two colors, we insert an edge between the vertices labeled by these two colors. For example, the edge $e = 13 \in H$ was inserted as a result of the chains $\gamma^{1}_{3}(v_2, v_3)$ and $\gamma^{3}_{1}(v_2, v_3)$. △

Proposition 3.1 establishes properties of the graph H.

Proposition 3.1. *The following properties hold for H:*

1. *H has $2n - 2$ vertices.*
2. *H has at least $(2n - 2)/2$ edges.*
3. *The number of edges in H is equal to half the number of chains used to construct H.*

4. *Each edge of H represents a compatible balanced set of two chains.*

5. *Each k-cycle in H represents a balanced set of k chains.*

Proof. The result follows from Lemmas 3.2, 3.6, 3.7, and 3.8 that are demonstrated in the following. ∎

Lemma 3.1. *There are $2n-2$ chains of length two linking vertices u and v.*

Proof. Urrutia, de Werra and Januario (2021) (in Proposition 1) showed that for each pair of vertices $u, v \in K_{2n}$, there is one or more compatible balanced set consisting of chains of length two linking u and v. For each set, the subgraph formed by the chains linking u and v is called a *colorful chordless lantern*, each of them with some degree $2 \leq l \leq 2n-2$, such that the sum of the degrees is $2n-2$. This implies that there are $2n-2$ chains of length two linking u and v. ∎

Lemma 3.2. *H has $2n-2$ vertices.*

Proof. From Lemma 3.1, at least $2n-2$ colors are represented by the pairs of compatible chains in $\mathcal{L}^{comp}$. Furthermore, notice that a chain with a given color either starts or ends with that color. Thus, the color of edge uv is not present in any chain linking u and v. This implies that exactly $2n-2$ colors are represented in $\mathcal{L}^{comp}$. Thus, H has $2n-2$ vertices.

∎

Lemma 3.3. *H has at least $(2n-2)/2$ edges.*

Proof. From Lemma 3.1, there are $2n-2$ chains of length two linking vertices u and v. Note that if u and v are linked by exactly $2n-2$ chains of even length and all colorful chordless lantern have degree equal to two, then H has $(2n-2)/2$ edges. ∎

Lemma 3.4. *Two even-size chains $\gamma^{\alpha}_{\beta}(u,v)$ and $\gamma^{\beta}_{\alpha}(u,v)$ are always compatible.*

Proof. Observe that the subgraph of K_{2n} formed by the union of the edges colored with α and β always form one or more cycles. Thus, the chains $\gamma^{\alpha}_{\beta}(u,v)$ and $\gamma^{\beta}_{\alpha}(u,v)$ have no edges in common. ∎

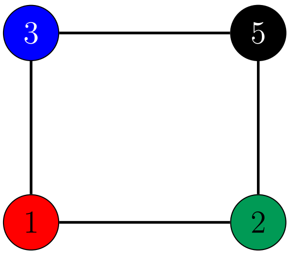


Figure 3.6: Graph H constructed using the even-size chains linking the vertices v_2 and v_3 of the edge colored graph K_6 in Figure 3.1, whose vertices $\{1, 2, 3, 5\}$ represent the four colors found in these chains.

Lemma 3.5. *If there is an even-size chain $\gamma_\beta^\alpha(u,v)$, then there is an even-size chain $\gamma_\alpha^\beta(u,v)$.*

Proof. Since $\gamma_\beta^\alpha(u,v)$ has even size, then its first and last edges are colored, respectively, with the colors α and β. It is known from Lemma 3.4 that $\gamma_\beta^\alpha(u,v)$ is in a bichromatic cycle. Thus there must be an even-size chain $\gamma_\alpha^\beta(u,v)$ in such a cycle. ∎

Lemma 3.6. *The number of edges in H is equal to half the number of chains used to construct H.*

Proof. From Lemma 3.5, for each pair of chains with the same colors connecting u and v, there will be an edge in the graph H. Thus, the result follows. ∎

Lemma 3.7. *Each edge of H represents a compatible balanced set of two chains.*

Proof. From Lemmas 3.4 and 3.5, an edge corresponds to two even-size compatible chains $\gamma_\beta^\alpha(u,v)$ and $\gamma_\alpha^\beta(u,v)$. Notice that for the set $\{\gamma_\beta^\alpha(u,v), \gamma_\alpha^\beta(u,v)\}$, the colors α and β belong to two chains and the other colors are not present. Thus, $\{\gamma_\beta^\alpha(u,v), \gamma_\alpha^\beta(u,v)\}$ is a compatible balanced set. ∎

Lemma 3.8. *Each k-cycle in H represents a balanced set of k chains.*

Proof. In a k-cycle $(e_1, e_2, \ldots, e_k)$ in the graph H, the chain represented by edge e_i has its second color equal to the first color of the chain represented by edge e_{i+1}, for $1 \le i < k$. In addition, the chain represented by edge e_k has its second color equal to the first color of the chain represented by edge e_1. This implies that the k colors represented in this k-cycle are present at the edges of exactly two chains, thus this k-cycle represents a balanced set of k chains. ∎

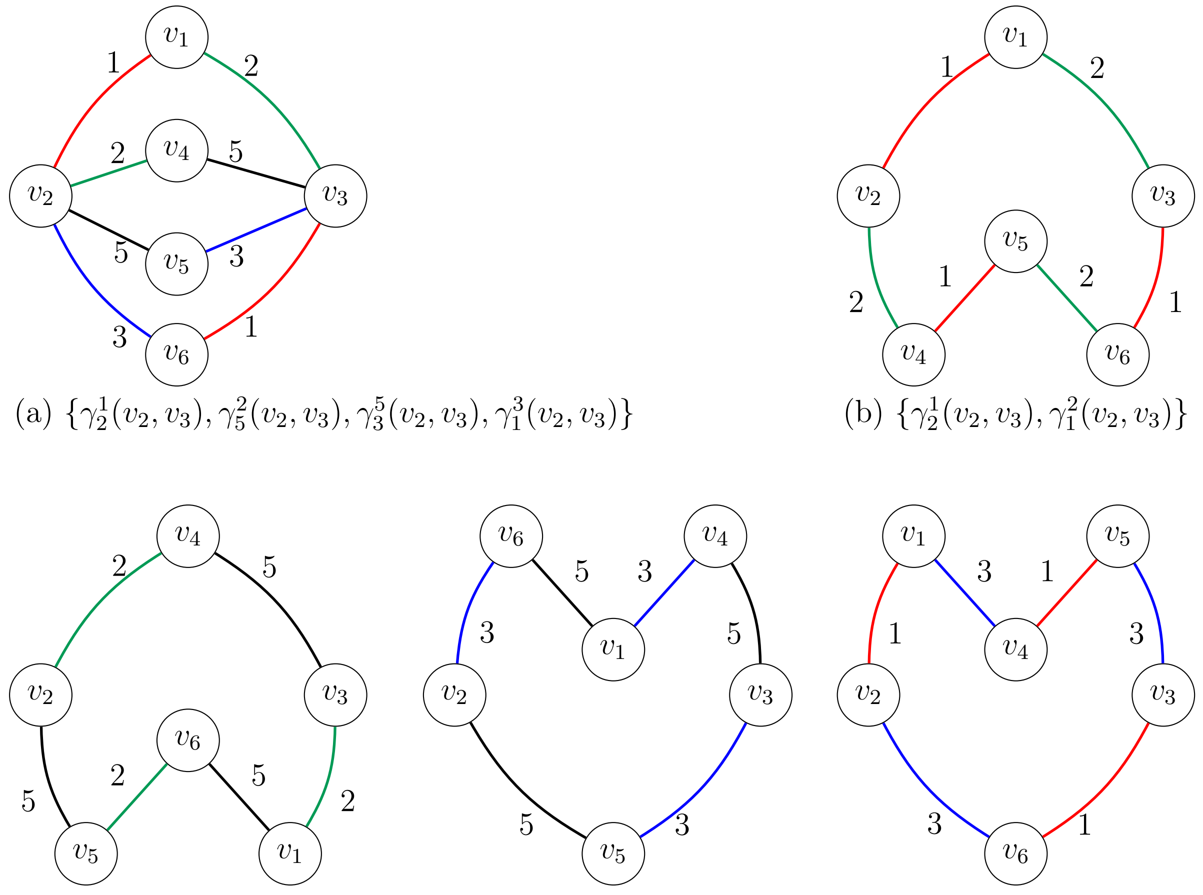
Notice that the balanced sets defined in Lemma 3.8 are not necessarily compatible.

3.3.2.2 Two strategies to obtain compatible balanced sets

In what follows, we present two strategies to be used in this phase: the complete systematic and the incomplete systematic. Both strategies obtain compatible balanced sets by identifying cycles in the auxiliary graph H. The complete systematic strategy obtains such cycles using backtracking-based enumeration while the incomplete systematic traverses H using graph-search algorithms such as depth-first search (DFS).

Given the graph H as input, for each k-cycle $(x_1, \ldots, x_k, x_1)$ obtained, we extract the chains represented by the edges of this cycle, $e = x_i x_{i+1}$ for all $1 \le i < k$ and $e = x_k x_1$, such that the chain $\gamma_{x_{i+1}}^{x_i}(u,v)$ is represented by the edge $e = x_i x_{i+1}$. Thus, each k-cycle represents a balanced set of k chains. Note that the chains extracted from the cycle depend on the vertex order, so cycles $(x_1, \ldots, x_k, x_1)$ and $(x_1, x_k, \ldots, x_1)$ represent distinct balanced sets, since $\gamma_{x_k}^{x_1}(u,v) \neq \gamma_{x_1}^{x_k}(u,v)$. Finally, for each edge of H we obtain a compatible balanced set consisting by the two chains that have the two colors associated with the labels of the vertices connected by this edge. Example 4 presents all the compatible balanced sets extracted from the cycles or edges of the graph H depicted in Figure 3.6.

Example 4. In the graph of Figure 3.6, the cycles $(1, 2, 5, 3, 1)$ and cycle $(1, 3, 5, 2, 1)$ represent different balanced sets with four chains. However, only the first cycle represents a compatible balanced set, namely, $\{\gamma_2^1(v_2, v_3), \gamma_5^2(v_2, v_3), \gamma_3^5(v_2, v_3), \gamma_1^3(v_2, v_3)\}$. The second cycle, on the other hand, represents the balanced set $\{\gamma_3^1(v_2, v_3), \gamma_5^3(v_2, v_3), \gamma_2^5(v_2, v_3), \gamma_1^2(v_2, v_3)\}$, in which four pairs of chains are incompatible. At last, from edges $e = 12$, $e = 25$, $e = 53$, and $e = 31$, we obtain, respectively, the following compatible balanced sets: $\{\gamma_2^1(v_2, v_3), \gamma_1^2(v_2, v_3)\}$, $\{\gamma_5^2(v_2, v_3), \gamma_2^5(v_2, v_3)\}$, $\{\gamma_5^3(v_2, v_3), \gamma_3^5(v_2, v_3)\}$, and $\{\gamma_3^1(v_2, v_3), \gamma_1^3(v_2, v_3)\}$. In such a way, a total of five compatible balanced sets are built. Figure 3.7 illustrates these five compatible balanced sets. △



(a) $\{\gamma_2^1(v_2, v_3), \gamma_5^2(v_2, v_3), \gamma_3^5(v_2, v_3), \gamma_1^3(v_2, v_3)\}$

(b) $\{\gamma_2^1(v_2, v_3), \gamma_1^2(v_2, v_3)\}$

(c) $\{\gamma_5^2(v_2, v_3), \gamma_2^5(v_2, v_3)\}$

(d) $\{\gamma_5^3(v_2, v_3), \gamma_3^5(v_2, v_3)\}$

(e) $\{\gamma_3^1(v_2, v_3), \gamma_1^3(v_2, v_3)\}$

Figure 3.7: Five compatible balanced sets extracted from the cycles or edges of the graph H depicted in Figure 3.6.

3.3.2.2.1 Complete systematic strategy

The complete systematic strategy traverses the graph H using a backtracking-based enumeration algorithm that can find all the cycles in H that represent compatible balanced sets. The procedure is described in Algorithms 2 and 3.

Algorithm 2 receives as inputs the graph H and the list $\mathcal{L}^{comp}$. Initially, the list $\mathcal{L}^{cycles}$ is initialized as empty (line 1). Lines 2-3 label all the vertices as unvisited. The foreach-loop in lines 4-7 explores all vertices of H. For each vertex $w \in H$, P is initialized as a single-vertex path (w) (line 5), x is set as w (line 6), and a call to BACKTRACKING (Algorithm 3) is performed in line 7. Finally, the algorithm ends by returning the list $\mathcal{L}^{cycles}$ containing all the cycles in H that represent compatible balanced sets (line 8).

Algorithm 2: ENUMERATE

```
  Input  : graph $H$ and list $\mathcal{L}^{comp}$
  Output: list $\mathcal{L}^{cycles}$ of cycles
1 $\mathcal{L}^{cycles} \leftarrow \varnothing$;
2 foreach $w \in H$ do
3     Mark $w$ as unvisited;
4 foreach $w \in H$ do
5     $P \leftarrow (w)$;
6     $x \leftarrow w$;
7     BACKTRACKING $(H, w, x, P, \mathcal{L}^{comp}, \mathcal{L}^{cycles})$;
8 return $\mathcal{L}^{cycles}$;
```

Algorithm 3 is a recursive algorithm with backtracking used to explore cycles in graph H. Algorithm 3 takes as input graph H, the root vertex w, a vertex x adjacent to w, the path P, and two lists $\mathcal{L}^{comp}$, $\mathcal{L}^{cycles}$. First, if vertex x is marked as visited, then the algorithm returns (lines 1-2). In line 3, x is marked as visited. The foreach-loop in lines 4-15 visits every unvisited vertex a adjacent to x. The variable *compatible* is set to **True** in line 5. For each vertex a and for each edge yz belonging to path P (line 6), it is verified whether $\gamma_a^x(u,v)$ is incompatible with $\gamma_z^y(u,v)$ in line 7. If they are incompatible, it means that the addition of vertex a will not constitute a promising path (that is, a compatible set). In this case, the variable *compatible* is set to **False** (line 8), the inner loop stops (line 9), and the algorithm resumes. If the variable *compatible* is **True** (line 10), then, there are two possibilities. Line 11 checks if $a = w$ and in case this is true, it means that the addition of vertex a forms a cycle, and thus the path $P \cdot (a)$ is inserted into $\mathcal{L}^{cycles}$ (line 12). Otherwise (i.e., $a \neq w$), the vertex a is appended to the end of the path P (line 14) and line 15 makes a recursive call to Algorithm 3. Finally, at the end of the algorithm, we mark x as *unvisited* (line 16) and remove it from the path P (line 17).

Observation. During the search, after visiting all successor vertices of a vertex w, we have marked w as not visited. Thus making it possible to find all the cycles in the graph H that represent compatible balanced sets. Note that, by selecting an unvisited vertex, we have been selecting an edge that corresponds to a chain and this one may be incompatible with the other chains, represented by the edges previously selected in this path. Then we can perform a forward checking of the addition of this edge, ensuring that every cycle found corresponds to a compatible set. In this way, no cycles representing incompatible sets are accounted in graph H.

Algorithm 3: BACKTRACKING

```
   Input: graph $H$, vertices $w, x \in H$, path $P$, and lists $\mathcal{L}^{comp}$, $\mathcal{L}^{cycles}$
1  if $x$ is visited then
2      return;
3  Mark $x$ as visited;
4  foreach unvisited vertex $a$ adjacent to $x$ do
5      compatible ← True;
6      foreach egde $(y, z) \in P$ do
7          if chains $\gamma_a^x(u, v)$ and $\gamma_z^y(u, v)$ are incompatible then
8              compatible ← False;
9              break;
10     if compatible = True then
11         if $a = w$ then
12             Add $P \cdot (a)$ to $\mathcal{L}^{cycles}$;
13         else
14             Add vertex $a$ into $P$;
15             BACKTRACKING $(H, w, a, P, \mathcal{L}^{comp}, \mathcal{L}^{cycles})$;
16 Mark $x$ as unvisited;
17 Remove vertex $x$ from $P$;
```

3.3.2.2.2 Incomplete systematic strategy

The incomplete systematic strategy, which we describe in this section, traverses the graph H using DFS. Starting from an unvisited vertex, the search recursively visits an unvisited neighbor. Whenever the traversal reaches a vertex that was already visited before, the search continues from another unvisited vertex if there is any. Besides, whenever a cycle is found, we extract the chains represented by the edges of this cycle, which represents a balanced set of chains, and the search continues from another unvisited vertex in case there is any. The procedure is quite similar to that described in Algorithms 2 and 3. The only difference is that in Algorithm 3 there is no line 16, since during the search no vertex is marked as *unvisited.*

Note that the goal of both strategies is to find only the cycles in the graph H that represent compatible balanced sets. In preliminary studies, we used Tarjan's algorithm (Tarjan, 1973), however, this algorithm proved to be unfeasible, as it exploits all the cycles of the graph. Therefore, to enable the use of the GPTS neighborhood in larger graphs, the algorithms described here were implemented.

3.3.3 The change phase

The change phase generates different neighbor colorings given as input a list $\mathcal{L}^{bal}$ of compatible balanced sets obtained in the construction phase, the graph K_{2n}, a coloring

c, and two distinct vertices $u, v \in K_{2n}$. Algorithm 4 describes the change phase. The foreach loop in lines 1-5 is executed for each of the compatible balanced sets in $\mathcal{L}^{bal}$. First, the new coloring c' is initialized as the input coloring c (line 2). After that, for each chain $\gamma_\beta^\alpha(u, v)$ belonging to *set*, we exchange the color assignment of its edges in order to change the coloring c' (lines 3-4). In the following, the new neighbor coloring c' is inserted into $\mathcal{S}$ (line 5). Finally, the list $\mathcal{S}$ of neighbor colorings is returned in line 6.

Algorithm 4: CHANGE-PHASE

```
  Input  : list $\mathcal{L}^{bal}$, graph $K_{2n}$, coloring $c$, and two distinct vertices $u, v \in K_{2n}$
  Output: list $\mathcal{S}$ of neighbor colorings
1 foreach $set \in \mathcal{L}^{bal}$ do
2     $c' \leftarrow c$;
3     foreach $\gamma_\beta^\alpha(u, v) \in set$ do
4         Exchange the color assignment of edges in $\gamma_\beta^\alpha(u, v) \in c'$;
5     Insert $c'$ into $\mathcal{S}$;
6 return $\mathcal{S}$;
```

Observation. Note that any chain or set of chains connecting a pair of vertices of K_{2n} is a subgraph of K_{2n}. However, such a set of chains is valid for a GPTS move if it is a compatible balanced set. In addition, it is not possible to perform a GPTS move using a single chain, as it would imply an invalid resulting coloring. Observe that in each subgraph constructed, one must obtain the same set of colors on edges that incident on the two vertices used as parameters in the selection phase. Note that this will always occur since the subgraph is constructed from a compatible balanced set. This implies that each color is present in two or zero chains, thus ensuring the same set of colors at the edges incident on these two vertices. Also, because it is a balanced set, the cardinality of this set is equal to the number of colors present in the chains of this set.

3.4 SOME EXPERIMENTAL RESULTS FOR 1-FACTORIZATIONS OF SMALL COMPLETE GRAPHS

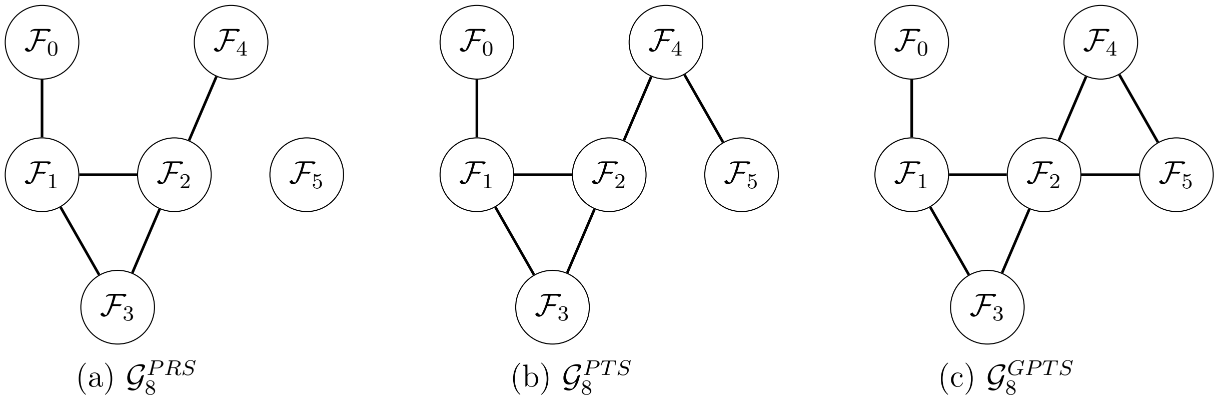
In this section, we present a brief discussion on how the GPTS neighborhood increases the connectivity of the non-isomorphic 1-factorization search space of K_{2n} (for $8 \leq 2n \leq 12$) when compared to PRS and PTS neighborhood structures. To accomplish that we use the graphs $\mathcal{G}_{2n}^{PRS}$, $\mathcal{G}_{2n}^{PTS}$, and $\mathcal{G}_{2n}^{GPTS}$, for $2n \in \{8, 10, 12\}$. Subsection 3.4.1 investigates the connectivity of these graphs and Subsection 3.4.2 reports some measures corresponding to each of these graphs.

3.4.1 Connectivity of the non-isomorphic 1-factorization search space

In this subsection, we investigate the connectivity of the graphs $\mathcal{G}_{2n}^{PRS}$, $\mathcal{G}_{2n}^{PTS}$, and $\mathcal{G}_{2n}^{GPTS}$, for $2n \in \{8, 10, 12\}$. To accomplish that we use the neighborhood structures PRS, PTS, and GPTS and the sets of non-isomorphic 1-factorizations of K_8 and K_{10}, as well as the set of *Perfect 1-factorizations* (P1Fs) of K_{12}. We seek to answer whether, given any 1-factorization of K_{2n}, it is possible to transform it into any other non-isomorphic 1-

factorization, through a sequence of moves. Remember that if a neighborhood structure is connected, it is possible, from any isomorphism class, to generate any other with a finite number of moves using the neighborhood.

First, we report the results on the connectivity of the graphs $\mathcal{G}_8^{PRS}$, $\mathcal{G}_8^{PTS}$, and $\mathcal{G}_8^{GPTS}$. Figure 3.8 presents these three graphs. From left to right, the edge set of graphs $\mathcal{G}_8^{PRS}$, $\mathcal{G}_8^{PTS}$, and $\mathcal{G}_8^{GPTS}$ corresponds, respectively, to a set of moves in the neighborhood structures PRS, PTS, and GPTS. Each edge that connects the vertices $\mathcal{F}_a$ and $\mathcal{F}_b$ indicates that the corresponding move is capable of generating a 1-factorization isomorphic to $\mathcal{F}_a$ (resp. $\mathcal{F}_b$) from a 1-factorization isomorphic to $\mathcal{F}_b$ (resp. $\mathcal{F}_a$). Note that, the graph $\mathcal{G}_8^{PRS}$ (Figure 3.8 (a)) is disconnected, whereas the graphs $\mathcal{G}_8^{PTS}$ (Figure 3.8 (b)) and $\mathcal{G}_8^{GPTS}$ (Figure 3.8 (c)) are connected. In addition, the GPTS move operator is the only one capable of directly connecting the 1-factorizations $\mathcal{F}_2$ and $\mathcal{F}_5$. The number of edges in each graph is respectively 5, 6, and 7 edges.



(a) $\mathcal{G}_8^{PRS}$ (b) $\mathcal{G}_8^{PTS}$ (c) $\mathcal{G}_8^{GPTS}$

Figure 3.8: Three graphs $\mathcal{G}_8^{M}$. From left to right, the graphs $\mathcal{G}_8^{PRS}$, $\mathcal{G}_8^{PTS}$, and $\mathcal{G}_8^{GPTS}$.

Experimental results on the connectivity of the graphs $\mathcal{G}_{10}^{PRS}$, $\mathcal{G}_{10}^{PTS}$, and $\mathcal{G}_{10}^{GPTS}$, show that the graph $\mathcal{G}_{10}^{PRS}$ is disconnected, whereas the graphs $\mathcal{G}_{10}^{PTS}$ and $\mathcal{G}_{10}^{GPTS}$ are connected. The number of edges corresponding to the $\mathcal{G}_{10}^{PRS}$, $\mathcal{G}_{10}^{PTS}$, and $\mathcal{G}_{10}^{GPTS}$, is equal to 1,667, 5,212, and 40,127, respectively.

According to Kaski et al. (2014), the graphs $\mathcal{G}_{12}^{PRS}$ and $\mathcal{G}_{12}^{PTS}$ are disconnected. Additionally, there are five isolated vertices in $\mathcal{G}_{12}^{PRS}$, each one representing an isomorphism class of perfect 1-factorization. In $\mathcal{G}_{12}^{PTS}$ there are eight isolated vertices, including the two that are isolated vertices in $\mathcal{G}_{12}^{PRS}$. In other words, the graph formed by the union of the edges sets for $\mathcal{G}_{12}^{PRS}$ and $\mathcal{G}_{12}^{PTS}$, has only two isolated vertices, each one representing a isomorphism class of perfect 1-factorization. Thus, such a graph has three connected components: two involving an isolated vertex and one involving the remaining vertices.

Our experimental results using the set of non-isomorphic P1Fs of K_{12} and the GPTS move operator, show that it is possible to transform any perfect 1-factorization into other 1-factorizations, which are not P1Fs. In other words, for both isolated vertices, the GPTS move operator can connect them to other vertices. It was established in Ribeiro, Urrutia and de Werra (2023) that the GPTS generalizes both PRS and PTS. Thus, we can conclude that the graph $\mathcal{G}_{12}^{GPTS}$ is connected.

3.4.2 Other graph measures

Let $\mathcal{F}_u$ and $\mathcal{F}_v$ be two vertices of a connected $\mathcal{G}_{2n}^M$. The *distance* $d(\mathcal{F}_u, \mathcal{F}_v)$ is the length of the smallest chain (not a bichromatic chain) that links these two vertices. The *eccentricity* of $\mathcal{F}_u$ is the maximum distance from $\mathcal{F}_u$ to any other vertex. The *diameter* and *radius* of $\mathcal{G}_{2n}^M$ are the largest and smallest eccentricity of all vertices of $\mathcal{G}_{2n}^M$, respectively. If a $\mathcal{G}_{2n}^M$ has a radius equal to r, there exists some vertex that connects to all others through at most r moves. A diameter equal to d implies that each pair of vertices can be connected by at most d moves. Note that a disconnected graph has infinite radius and infinite diameter. The *density* of a $\mathcal{G}_{2n}^M$ is the ratio between its number of edges and the number of possible edges, that is, the density of a graph $G = (V, E)$ is $\frac{2|E|}{|V|(|V|-1)}$.

Table 3.1 presents the density (den), radius (rad), and diameter(dia) values for the graphs $\mathcal{G}_8^M$ and $\mathcal{G}_{10}^M$, for $M \in \{PRS, PTS, GPTS\}$. We can use the density as a metric to indicate which move operator is able to directly connect more pairs of vertices. The table shows that the GPTS move operator is able to directly connect more pairs of non-isomorphic 1-factorizations of both K_8 and K_{10}. Note that for K_8, the density values are relatively similar. The $\mathcal{G}_8^{PRS}$ is approximately 39.39% smaller than $\mathcal{G}_8^{GPTS}$, and $\mathcal{G}_8^{PTS}$ is 15% smaller than $\mathcal{G}_8^{GPTS}$. However, for the K_{10}, the density values differ significantly, since the $\mathcal{G}_{10}^{PRS}$ and $\mathcal{G}_{10}^{PTS}$ are approximately 2342.86% and 677.27% smaller than $\mathcal{G}_{10}^{GPTS}$, respectively. For both graphs $\mathcal{G}_8^{PTS}$ and $\mathcal{G}_8^{GPTS}$, the radius is equal to 2 and that means there is at least one vertex that requires 2 moves to connect to the other vertices. The diameter equal to 4 and equal to 3, respectively, for the graphs $\mathcal{G}_8^{PTS}$ and $\mathcal{G}_8^{GPTS}$, indicates that every pair of vertices requires, at most, 4 and 3 moves to connect to each other. For the graph $\mathcal{G}_{10}^{PTS}$ the radius is equal to 3 and the diameter equal to 6. For the graph $\mathcal{G}_{10}^{GPTS}$ the radius is equal to 2 and the diameter equal to 4.

Table 3.1: Density, radius, and diameter values for $\mathcal{G}_8^M$ and $\mathcal{G}_{10}^M$, for $M \in \{PRS, PTS, GPTS\}$.

M	$\mathcal{G}_8^M$			$\mathcal{G}_{10}^M$		
	den	rad	dia	den	rad	dia
PRS	0.33	∞	∞	0.021	∞	∞
PTS	0.40	2	4	0.066	3	6
GPTS	0.46	2	3	0.513	2	4

Given our hardware limitations, we do not report values for the graph $\mathcal{G}_{12}^M$. According to Kaski et al. (2014), the density of $\mathcal{G}_{12}^{PTS}$ is 0.0000001295 (and that of $\mathcal{G}_{12}^{PRS}$ is 0.0000000423).

The results showed that the GPTS neighborhood structure increases the connectivity of the non-isomorphic 1-factorization search space of K_{2n} (for $8 \leq 2n \leq 12$) when compared to PRS and PTS neighborhood structures. Although GPTS is more efficient at directly connecting more pairs of non-isomorphic 1-factorizations of K_{2n}, exploring the GPTS neighborhood has a higher computational cost than exploring the PRS and PTS neighborhoods. This is because, in addition to GPTS generalizing PRS and PTS, in this neighborhood, there are more neighbor colorings than those found in the PRS and PTS neighborhoods. In Subsection 3.5.2, we illustrate a comparison of the number of neighbor colorings obtained using each of these neighborhood structures.

3.5 PRELIMINARY COMPUTATIONAL RESULTS

In this section, we summarize the results of the preliminary experiments carried out to evaluate the performance of the neighborhood structures PRS, PTS, and GPTS using the WCOEV Minimization Problem (Guedes; Ribeiro, 2011) as a case study. We also illustrate a comparison of the number of neighbor colorings obtained using each of these neighborhood structures. Subsection 3.5.1 presents the results on the performance of neighborhood structures and Subsection 3.5.2 illustrates the comparison as to the number of neighbor colorings.

3.5.1 Experimental results

In the following, we describe the WCOEV Minimization Problem. Let a, b, and c be three teams from an SRR tournament with $2n$ teams. If team c plays against teams a and b in two consecutive rounds, team b is said to receive a carry-over effect from team a. Let C_{ab} denote the number of times team b receives a carry-over effect from team a during the tournament. In the WCOEV Minimization Problem, we assign a weight w_{ab} to every ordered pair (a, b) of teams and minimize the total weighted carry-over effects value, given by WCOEV $= \sum_{a=1}^{2n} \sum_{b=1}^{2n} w_{ab} \times C_{ab}^2$.

In order to conduct the experiments, instances for the WCOEV Minimization Problem were used. In each experiment, we evaluated the performance of neighborhood structures through a local search based on one iteration of the best improvement strategy. Algorithm 5 details the procedure. First, the algorithm starts by constructing an initial solution s with the circle method (line 1). After that, the solution $s^\star$ is set as s (line 2). Then, a local search procedure is applied (line 3) to the initial solution s, using the GPTS (complete or incomplete systematic), PRS, or PTS neighborhood structure. After that, for each solution s' (neighbor of s), it is verified whether the cost function of s' is smaller than the cost function of $s^\star$ (line 4). In this case, $s^\star$ is replaced by s' in line 5. Finally, the algorithm ends by returning the local optimum solution $s^\star$ in line 6.

Algorithm 5: SINGLE-ITERATION-BEST-IMPROVEMENT

```
1 Build a solution $s$ with the circle method;
2 $s^\star \leftarrow s$;
3 foreach solution $s' \in \mathcal{V}(s)$ do
4     if $f(s') < f(s^\star)$ then
5         $s^\star \leftarrow s'$;
6 return $s^\star$;
```

All the experiments were performed on a machine running under Ubuntu 22.04.1 LTS with an Intel Core i5-9300H 2.40 GHz processor and 8 GB of RAM. The codes were written in C++ and compiled with g++ version 11.3.0, using the options '-O3' and '-std=c++20'.

In the remainder of this section, the GPTS complete systematic may be referenced as GPTS-C. Likewise, the GPTS incomplete systematic may be referenced as GPTS-I.

3.5.1.1 Results

In this subsubsection, we present the experiments and discuss the results found. Each experiment was carried out using 10 instances generated by Guedes and Ribeiro (2011). Each instance has a size corresponding to the number of teams involved in the tournament. On all instances (except for instances of size equal to 24) and for each of the neighborhood structures, Algorithm 5 was run for 100 distinct (and isomorphic) 1-factorizations. Whereas, for the two instances of size equal to 24, this Algorithm 5 was run for 10 distinct (and isomorphic) 1-factorizations, for each the neighborhood structures. We reduced to 10 distinct 1-factorizations because, when using GPTS-C for instances of size 24, the execution took about 91 hours. The results were compared over the cost function value.

Table 3.2 presents the results on the performance of neighborhood structures. Observing the table, each row represents an instance. The first column shows the instance sizes and the second column gives the instance names. The third, fourth, fifth, and sixth columns presents the average percentage improvement of the cost function value obtained using the PRS, PTS, GPTS-C, and GPTS-I, respectively, considering all initial 1-factorizations. For each initial 1-factorization and each neighborhood, the value was computed as $\frac{f(s)-f(s^{\star})}{f(s)} \times 100\%$, where $f(s)$ stands for the initial cost function value and $f(s^{\star})$ represents the value obtained with the best improvement strategy using the corresponding neighborhood structure. Experimental results showed that for all instances, GPTS-C and GPTS-I outperform PRS and PTS neighborhood structures. Finally, as expected, it is observed that GPTS-C outperforms GPTS-I, regardless of the instance.

Table 3.2: Numerical results for the WCOEV Minimization Problem, considering a set of distinct (and isomorphic) 1-factorizations of K_{2n} for each instance.

Size	Instance	PRS	PTS	GPTS-C	GPTS-I
8	Perturbed,8,C	31.5%	28.0%	41.9%	41.2%
10	Perturbed,10,C	44.6%	35.3%	54.7%	52.1%
12	Perturbed,12,C	47.7%	24.0%	66.1%	62.6%
14	Perturbed,14,C	48.3%	17.7%	71.3%	67.7%
16	Perturbed,16,C	36.4%	26.9%	60.0%	55.2%
18	Perturbed,18,C	39.9%	17.4%	72.4%	67.7%
20	Perturbed,20,C	40.9%	27.5%	83.4%	79.4%
22	Brazil,22,2005	36.3%	32.8%	78.8%	74.0%
24	Brazil,24,2003	33.6%	27.3%	82.9%	78.0%
24	Brazil,24,2004	32.6%	17.6%	81.0%	75.3%

Lastly, we present a comparison of the duration of the experiments by considering a single 1-factorization per instance. In all experiments, when using the PRS or PTS, the execution lasted less than 1 second, regardless of the instance. In the experiments for instances of size smaller than or equal to 20, when using GPTS-I the execution lasted less than 1 second. On the other hand, for instances of size equal to 22 and 24, the execution took about 1 and 3 seconds, respectively. In the experiments for instances of size smaller than or equal to 18, when using GPTS-C the execution lasted less than 1 second. On the other hand, for instances of size equal to 20, 22, and 24, the execution took about 29 minutes, 07 hours, and 91 hours, respectively. According to these durations and the obtained performances, the GPTS-I is a viable alternative to be used for instances greater than or equal to 20, rather than GPTS-C.

3.5.2 Number of neighbor colorings

This subsection illustrates a comparison of the number of neighbor colorings obtained by using the PRS, PTS, and GPTS neighborhood structures. To accomplish that we use nine 1-factorizations of K_{2n} (for $8 \leq 2n \leq 24$), generated by the circle method. For each 1-factorization and neighborhood structure, we counted the number of neighbor colorings obtained by using this neighborhood. The number of neighbors is determined according to the definitions described in Subsection 3.1.1. Table 3.3 shows the number of neighbor colorings obtained by using the PRS, PTS, and GPTS (complete and incomplete systematic) neighborhood structures. Note that even using the incomplete systematic (for $2n \in \{8, 10\}$), the GPTS neighborhood obtains at least three times more neighbor colorings than the sum of neighbor colorings obtained when using PRS and PTS. Additionally, the incomplete systematic obtains at least twelve times more neighbor colorings than the sum of neighbor colorings obtained when using PRS and PTS, for $2n \geq 12$.

Table 3.3: Number of neighbor colorings obtained by using the PRS, PTS, and GPTS (complete and incomplete systematic) neighborhood structures.

K_{2n}	PRS	PTS	GPTS-C	GPTS-I
K_8	21	35	224	168
K_{10}	45	72	990	394
K_{12}	55	66	16,005	1,452
K_{14}	78	91	71,032	3,177
K_{16}	165	285	308,940	4,089
K_{18}	136	170	8,306,421	11,485
K_{20}	171	190	77,178,513	18,731
K_{22}	336	567	1,287,482,301	21,707
K_{24}	253	299	16,311,678,499	45,635

Chapter 4

CONCLUDING REMARKS

This dissertation proposed two new invariants (denoted *lantern profiles* and *even-size bichromatic chains*) for 1-factorizations of K_{2n} and presented algorithms to systematically explore the *Generalized Partial Team Swap* (GPTS) neighborhood structure, proposed in Januario et al. (2016). In this dissertation, we described seven invariants available in the literature. For all the nine invariants presented, we analyzed their size and computational complexity. Furthermore, we performed experiments to evaluate the strength of the invariants. Last, the strengths of the combinations of invariants were also evaluated, showing a complementarity in their distinguishing abilities. A preliminary version of the results in Chapter 2 was presented at the LV Simpósio Brasileiro de Pesquisa Operacional (Matos et al., 2023).

An important contribution of this work was to present a study on algorithmic and computational aspects of the GPTS neighborhood structure. Furthermore, two strategies to obtain compatible balanced sets were presented. This dissertation provided a brief discussion on how the GPTS neighborhood structure increases the connectivity of the non-isomorphic 1-factorization search space of K_{2n} (for $8 \leq 2n \leq 12$) when compared to *Partial Round Swap* (PRS) and *Partial Team Swap* (PTS) neighborhood structures. We showed that the GPTS move operator is able to directly connect more pairs of non-isomorphic 1- factorizations of both K_8 and K_{10}. Additionally, the GPTS move operator is the only one capable of connecting the non-isomorphic 1-factorization search space of K_{12}. Finally, preliminary computational experiments were conducted to evaluate the performance of the GPTS, PRS, and PTS neighborhood structures, using the *Weighted Carry-Over Effects Value* (WCOEV) Minimization Problem as a case study.

IDEAS FOR FUTURE RESEARCH

Considering that all the invariants presented in this work are not complete, it is relevant to propose new invariants for 1-factorizations of K_{2n}. Additionally, to obtain more robust results, it will be necessary to computational experiments to calculate the strength of the invariants on the full set of non-isomorphic 1-factorizations of K_{12}.

Regarding the study of aspects of the GPTS neighborhood, another research direction would be to analyze the structure of the subgraphs obtained in the GPTS neighborhood to identify patterns that facilitate the exploration of the solution space graph. Furthermore, it is important to study other strategies to obtain compatible balanced sets in addition to the two strategies presented in Chapter 3.

BIBLIOGRAPHY

Anagnostopoulos, A.; Michel, L.; Van Hentenryck, P.; Vergados, Y. A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, v. 9, p. 177–193, 01 2006.

Babai, L. Graph isomorphism in quasipolynomial time. preprint, 2015. Available in: <http://arxiv.org/abs/1512.03547>.

Babai, L. Graph isomorphism in quasipolynomial time [extended abstract]. In: *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: Association for Computing Machinery, 2016. (STOC '16), p. 684–697. ISBN 9781450341325.

Brigham, R. C.; Dutton, R. D. A compilation of relations between graph invariants. *Networks*, v. 15, n. 1, p. 73–107, 10 1985.

Briskorn, D.; Drexl, A. IP models for round robin tournaments. *Computers & Operations Research*, v. 36, n. 3, p. 837 – 852, 03 2009. ISSN 0305-0548.

Colbourn, C. J.; Dinitz, J. H. *Handbook of Combinatorial Designs, Second Edition*. New York: Taylor & Francis, 2006. (Discrete Mathematics and Its Applications). ISBN 1584885068.

Costa, F. N.; Urrutia, S.; Ribeiro, C. C. An ILS heuristic for the traveling tournament problem with predefined venues. *Annals of Operations Research*, Springer US, v. 194, n. 1, p. 137–150, 2012.

de Werra, D. Geography, games and graphs. *Discrete Applied Mathematics*, Elsevier, v. 2, n. 4, p. 327–337, 1980.

de Werra, D. Minimizing irregularities in sports schedules using graph theory. *Discrete Applied Mathematics*, Elsevier, v. 4, n. 3, p. 217–226, 1982.

de Werra, D. On the multiplication of divisions: The use of graphs for sports scheduling. *Networks*, v. 15, n. 1, p. 125–136, 1985.

Demsar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, Cambridge, MA, USA, v. 7, p. 1–30, 2006.

di Gaspero, L.; Schaerf, A. A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, v. 13, p. 189 – 207, 2007.

Dickson, L. E.; Safford, F. H. Solution to problem 8 (group theory). *The American Mathematical Monthly*, Mathematical Association of America, v. 13, n. 6/7, p. 150–151, 1906. ISSN 00029890, 19300972. Available in: <http://www.jstor.org/stable/2970423>.

Dinitz, J. H.; Garnick, D. K. There are 23 nonisomorphic perfect one-factorizations of K_{14}. *Journal of Combinatorial Designs*, v. 4, n. 1, p. 1–4, 1996.

Dinitz, J. H.; Garnick, D. K.; McKay, B. D. There are 526,915,620 nonisomorphic one-factorizations of K_{12}. *J. COMBIN. DES*, v. 2, p. 273–285, 1994.

Dinitz, J. H.; Wallis, W. D. Trains: An invariant for one-factorizations. *Ars Combinatoria*, Canada, v. 32, p. 161–180, 1991.

Durán, G. Sports scheduling and other topics in sports analytics: a survey with special reference to Latin America. *TOP*, Springer, v. 29, n. 1, p. 125–155, 2021.

Durán, G.; Durán, S.; Marenco, J.; Mascialino, F.; Rey, P. Scheduling argentina's professional basketball leagues: A variation on the relaxed travelling tournament problem. *European Journal of Operational Research*, v. 275, n. 3, p. 1126–1138, 12 2019. ISSN 0377-2217.

Easton, K.; Nemhauser, G.; Trick, M. The traveling tournament problem description and benchmarks. *Lecture Notes in Computer Science*, Berlin, Heidelberg, v. 2239, p. 580–584, 2001.

Garey, M. R.; Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman & Co., 1979.

Geinoz, A.; Ekim, T.; de Werra, D. Construction of balanced sports schedules using partitions into subleagues. *Operations Research Letters*, Elsevier, v. 36, n. 3, p. 279–282, 2008. ISSN 0167-6377.

Gelling, E. N. On 1-factorizations of the complete graph and the relationship to round robin schedules. In: *MA Thesis*. Canada: University of Victoria, 1973.

Gelling, E. N.; Odeh, R. On 1-factorizations of the complete graph and the relationship to round-robin schedules. *Congressus Numerantium*, v. 9, p. 213–221, 1974.

Gill, M. J.; Wanless, I. M. Perfect 1-factorisations of K_{16}. *Bulletin of the Australian Mathematical Society*, Cambridge University Press, v. 101, n. 2, p. 177–185, abr. 2020.

Griggs, T. S.; Rosa, A. An invariant for one-factorizations of the complete graph. *Ars Combinatoria*, v. 42, p. 77–88, 1996.

Grohe, M.; Schweitzer, P. The graph isomorphism problem. *Communications of the ACM*, Association for Computing Machinery, New York, NY, USA, v. 63, n. 11, p. 128–134, oct 2020. ISSN 0001-0782.

Guedes, A. C.; Ribeiro, C. C. A heuristic for minimizing weighted carry-over effects in round robin tournaments. *Journal of Scheduling*, Springer US, v. 14, n. 6, p. 655–667, 2011. ISSN 1094-6136.

Hamiez, J.-P.; Hao, J.-K. Solving the sports league scheduling problem with tabu search. *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, Springer, Berlin, v. 2148, p. 24–36, 08 2000.

Helfgott, H. A.; Bajpai, J.; Dona, D. Graph isomorphisms in quasi-polynomial time. preprint, 2017. Available in: <https://arxiv.org/abs/1710.04574>.

Januario, T.; Urrutia, S. An analytical study in connectivity of neighborhoods for single round robin tournaments. *In Operations Research and Computing: Algorithms and Software for Analytics*, VI: Informs, 2015.

Januario, T.; Urrutia, S. A new neighborhood structure for round robin scheduling problems. *Computers & Operations Research*, v. 70, p. 127 – 139, 2016. ISSN 0305-0548.

Januario, T.; Urrutia, S.; de Werra, D. Sports scheduling search space connectivity: A riffle shuffle driven approach. *Discrete Applied Mathematics*, v. 211, p. 113–120, 2016. ISSN 0166-218X.

Januario, T.; Urrutia, S.; Ribeiro, C. C.; de Werra, D. Edge coloring: A natural model for sports scheduling. *European Journal of Operational Research*, v. 254, n. 1, p. 1 – 8, 2016.

Kaski, P.; Medeiros, A. de S.; Östergård, P. R. J.; Wanless, I. M. Switching in one-factorisations of complete graphs. *The Electronic Journal of Combinatorics*, Clemson University Digital Press, v. 21, n. 2, p. 1–24, 2014.

Kaski, P.; Östergård, P. R. J. There are 1,132,835,421,602,062,347 nonisomorphic one-factorizations of K_{14}. *Journal of Combinatorial Designs*, v. 17, n. 2, p. 147–159, 2009.

Kendall, G.; Knust, S.; Ribeiro, C. C.; Urrutia, S. Scheduling in sports: An annotated bibliography. *Computers & Operations Research*, v. 37, n. 1, p. 1 – 19, 2010. ISSN 0305-0548.

Kirkman, T. P. On a problem in combinations. *The Cambridge and Dublin Mathematical Journal*, v. 2, p. 191–204, 1847. Available in: <https://gdz.sub.uni-goettingen.de/id/PPN600493962_0002>.

Lim, A.; Rodrigues, B.; Zhang, X. A simulated annealing and hill-climbing algorithm for the traveling tournament problem. *European Journal of Operational Research*, v. 174, n. 3, p. 1459–1478, 11 2006. ISSN 0377-2217.

Matos, S.; Melo, R.; Januario, T.; Urrutia, S. Novas invariantes para 1-fatorações do grafo K_{2n}. In: *Proceedings of the LV Simpósio Brasileiro de Pesquisa Operacional.* São José dos Campos, SP, BRA: [s.n.], 2023. v. 55. ISSN 2965-1476.

McKay, B. D.; Piperno, A. Practical graph isomorphism, II. *Journal of Symbolic Computation*, v. 60, p. 94–112, 2014. ISSN 0747-7171.

Melo, R. A.; Urrutia, S.; Ribeiro, C. C. The traveling tournament problem with predefined venues. *Journal of Scheduling*, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 6, p. 607 – 622, dez. 2009. ISSN 1094-6136.

Mendelsohn, E.; Rosa, A. One-factorizations of the complete graph - A survey. *Journal of Graph Theory*, v. 9, n. 1, p. 43–65, 1985.

Meszka, M. There are 3155 nonisomorphic perfect one-factorizations of K_{16}. *Journal of Combinatorial Designs*, v. 28, n. 1, p. 85–94, 2020.

Misra, J.; Gries, D. A constructive proof of Vizing's theorem. *Information Processing Letters*, v. 41, n. 3, p. 131 – 133, 1992. ISSN 0020-0190.

Petrenyuk, A. P.; Petrenyuk, A. Y. Intersection of perfect 1-factorizations of complete graphs. *Cybernetics and Systems Analysis*, v. 16, p. 6–9, 1980.

Rasmussen, R. V.; Trick, M. A. Round robin scheduling – a survey. *European Journal of Operational Research*, v. 188, n. 3, p. 617–636, 2008. ISSN 0377-2217.

Ribeiro, C. C. Sports scheduling: Problems and applications. *International Transactions in Operational Research*, v. 19, n. 1-2, p. 201–226, 2012.

Ribeiro, C. C.; Urrutia, S. Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, v. 179, n. 3, p. 775–787, 2007. ISSN 0377-2217.

Ribeiro, C. C.; Urrutia, S. A.; de Werra, D. *Combinatorial models for scheduling sports tournaments*. [S.l.]: Springer, 2023. (EURO Advanced Tutorials on Operational Research).

Rosa, A. Perfect 1-factorizations. *Mathematica Slovaca*, De Gruyter, v. 69, n. 3, p. 479–496, 2019.

Russell, R.; Urban, T. A constraint programming approach to the multiple-venue, sport-scheduling problem. *Computers & Operations Research*, v. 33, p. 1895–1906, 07 2006. ISSN 0305-0548. Special Issue: Operations Research in Sport.

Seah, E.; Stinson, D. R. Some perfect one-factorizations of K_{14}. In: COLBOURN, C. J.; MATHON, R. (Ed.). *Combinatorial Design Theory*. Netherlands: Elsevier, 1987, (North-Holland Mathematics Studies, v. 149). p. 419–436.

Seah, E.; Stinson, D. R. On the enumeration of one-factorizations of complete graphs containing prescribed automorphism groups. *Mathematics of Computation*, v. 50, n. 182, p. 607–618, 1988. ISSN 00255718, 10886842.

Talbi, E.-G. *Metaheuristics: From Design to Implementation.* New Jersey: Wiley, 2009. ISBN 9780470278581.

Tarjan, R. Enumeration of the elementary circuits of a directed graph. *SIAM Journal on Computing*, v. 2, n. 3, p. 211–216, 1973.

Trick, M. A Schedule-Then-Break Approach to Sports Timetabling. In: BURKE, E.; ERBEN, W. (Ed.). *Practice and Theory of Automated Timetabling III.* Berlin: Springer, 2001. (PATAT 2000. Lecture Notes in Computer Science, v. 2079), p. 242–253. ISBN 978-3-540-44629-3.

Urrutia, S.; de Werra, D.; Januario, T. Recoloring subgraphs of K_{2n} for sports scheduling. *Theoretical Computer Science*, v. 877, p. 36–45, 2021. ISSN 0304-3975.

Vizing, V. G. On an estimate of the chromatic class of a p-graph. *Discrete Analysis*, v. 3, p. 25 − 30, 1964. In Russian.

Wallis, W. D. On one-factorizations of complete graphs. *Journal of the Australian Mathematical Society*, Cambridge University Press, v. 16, n. 2, p. 167–171, 1973.

Wallis, W. D. *One-Factorizations.* Kluwer, Dordrecht, Netherlands: Springer, 1997. (Mathematics and Its Applications).

Wallis, W. D. *Introduction to Combinatorial Designs, Second Edition (Discrete Mathematics and Applications).* New York: Chapman & Hall/CRC, 2007. ISBN 1584888385.

Wanless, I. M. Cycle switches in latin squares. *Graphs and Combinatorics*, v. 20, p. 545–570, 11 2004.

Wanless, I. M. Author's homepage. 2023. Available in: <https://users.monash.edu.au/~iwanless/data/P1F/newP1F.html>.