UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GABRIEL COPLAS BECHER

COLORAÇÃO TOTAL DE GRAFOS BIPARTIDOS

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA
2019

GABRIEL COPLAS BECHER

COLORAÇÃO TOTAL DE GRAFOS BIPARTIDOS

Trabalho de conclusão de curso apresentado como requisito parcial à obtenção do grau de Bacharel em Ciência da Computação, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná.

Orientador: Drª. Sheila Morais de Almeida

PONTA GROSSA
2019

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Ponta Grossa
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Informática
Bacharelado em Ciência da Computação

TERMO DE APROVAÇÃO

Coloração Total de Grafos bipartidos

por

Gabriel Coplas Becher

Este Trabalho de Conclusão de Curso (TCC) foi apresentado em 26 de junho de 2019 como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof(a). Dra. Sheila Morais de Almeida
Orientador(a)

_____
Prof Dr Gleifer Vaz Alves
Membro titular

_____
Prof MSc Saulo Jorge Beltrão Queiroz
Membro titular

_____
Prof MSc Denise do Rocio Maciel
Membro titular

_____
Prof. MSc Geraldo Ranthum
Responsável pelo Trabalho de Conclusão de Curso

_____
Prof. Dr. Saulo Jorge Beltrão Queiroz
Coordenador do curso

Dedico esse trabalho à minha família,
aos amigos, a minha namorada e a todos os professores
que me incentivaram a continuar
mesmo nas horas em que pensei em desistir.

AGRADECIMENTOS

Agradecimento em primeiro lugar a Deus pelas oportunidades.

Em segundo lugar à minha família pelo apoio.

À minha namorada pela paciência e por acreditar em mim.

E a minha orientadora Drª Sheila Morais de Almeida por toda a dedicação em me fornecer todo apoio necessário para eu concluir esse trabalho.

“Se não puder voar, corra.
Se não puder correr, ande.
Se não puder andar, rasteje,
mas continue em frente de qualquer jeito.”
Martin Luther King

RESUMO

C. BECHER, Gabriel. **Coloração total de grafos bipartidos**. 2019, 58 p. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2019.

Uma coloração total em um grafo G é uma atribuição de cores para os elementos de G de forma que quaisquer dois elementos adjacentes tenham cores diferentes. O Problema da Coloração Total é determinar o menor número de cores com que se pode obter uma coloração total de um dado grafo G. Esse número é denominado de número cromático total e denotado por $\chi''(G)$. Em sua versão de decisão, o Problema da Coloração Total recebe como entrada um grafo G e um número inteiro k para os quais deseja-se saber se é possível realizar uma coloração total de G com no máximo k cores. A versão de decisão do Problema da Coloração Total é um problema NP-completo mesmo quando restrita à classe dos grafos bipartidos, que são aqueles cujos vértices podem ser particionados em dois conjuntos independentes. Não se conhece algoritmo eficiente que resolva em tempo polinomial qualquer dos problemas NP-completos, mas caso algum algoritmo de complexidade polinomial exista, então o mesmo poderá ser utilizado para resolver qualquer problema da classe de problemas NP. Apresentar um algoritmo eficiente para um problema NP-completo ou provar que tal algoritmo não existe é um dos sete problemas mais desafiadores do milênio, segundo o *Clay Mathematics Institute*. Nesse trabalho determinou-se um subconjunto dos grafos bipartidos em que o Problema da Coloração Total permanece em aberto.

Palavras-chaves: Coloração total. Grafos bipartidos. Número cromático total.

ABSTRACT

C. BECHER, Gabriel. **Total coloring of bipartite graphs**. 2019, 58 f. Work of Conclusion Course (Graduation in Computer Science) - Federal University of Technology - Paraná. Ponta Grossa, 2019.

A total coloring in a graph G is a color assignment to elements to G so that any two adjacent elements have different colors. The Total Coloring Problem is to determine the smallest number of colors to obtain a total coloring for a given graph G. This number is called the total chromatic number and it is denoted by $\chi''(G)$. In its decision version, the Total Coloring Problem receives as input a graph G and an integer number, k, and it is desired to know if there is a total coloring for G with at most k colors. The decision version of Total Coloring Problem is an NP-Complete problem, even when restricted to the class of bipartite graphs, which are those whose vertices can be partitioned into two independent sets. An efficient algorithm that solves any of the NP-Complete problems in polynomial time is unknown, but if any algorithm of polynomial complexity exists, then it can be used to solve any problem of the class NP. Presenting an efficient algorithm for an NP-complete problem or proving that such an algorithm does not exist is one of the seven most challenging problems of the millennium, according to the *Clay Mathematics Institute*. In this work we present a subset of bipartite graphs for which the Total Coloring Problem remains open.

Key-words: Total coloring. Bipartite graphs. Total chromatic number

LISTA DE ILUSTRAÇÕES

LISTA DE SÍMBOLOS

$V(G)$	Conjunto de vértices do grafo G
$E(G)$	Conjunto de arestas do grafo G
$\mathbb{N}$	Conjunto dos números naturais
$\chi(G)$	Número cromático do grafo G
$\chi'(G)$	Índice cromático do grafo G
$\chi''(G)$	Número cromático total do grafo G
$\Delta(G)$	Grau máximo do grafo G
$K_{\|A\|,\|B\|}$	Grafo bipartido completo com partição dos vértices em conjuntos independentes A e B
$E(G)\backslash e$	Conjunto de arestas $E(G)$ menos a aresta e.
$E(G)\backslash E(H)$	Conjunto de arestas $E(G)$ menos as arestas do conjunto $E(H)$.
$G \cup H$	Grafo resultante da operação de união dos grafos G e H.
	$G \times H$ Produto cartesiano entre os grafos G e H.

SUMÁRIO

1 INTRODUÇÃO

Na década de 60, foi apresentado por Behzad (1965) um problema computacional que ficou conhecido como Problema da Coloração Total. Mais de 50 anos depois, ainda não se conhece um algoritmo eficiente que resolva esse problema. Durante este meio século muitos pesquisadores dedicaram-se a identificar a complexidade do problema, que é computacionalmente difícil (SÁNCHEZ-ARROYO, 1989), e a apresentar soluções para diversos casos especiais, como por exemplo nos trabalhos de Vijayaditya (1971), Rosenfeld (1971), McDiarmid e Sánchez-Arroyo (1994), Chew e Yap (1992) e Machado e Figueiredo (2011), dentre outros. Mesmo em classes de grafos para as quais diversos problemas da Teoria dos Grafos possuem algoritmos eficientes, este problema permanece em aberto. Uma dessas classes é a classe dos grafos bipartidos, que será definida a seguir. Como são conhecidas soluções computacionais eficientes para o Problema da Coloração Total em variados subconjuntos de grafos bipartidos (BEHZAD; CHARTRAND; JR, 1967; BERMOND, 1974; VIJAYADITYA, 1971; YAP; JIAN-FANG; ZHONGFU, 1989), é difícil identificar para quais grafos desta classe o problema permanece em aberto. A identificação deste subconjunto de grafos é importante para o prosseguimento de pesquisas sobre o tema. Uma solução computacional eficiente do Problema da Coloração Total para todos os grafos bipartidos implica na solução de um dos problemas matemáticos mais desafiadores deste milênio (CARLSON *et al.*, 2006). Este Trabalho de Conclusão de Curso apresenta um conjunto de grafos bipartidos para os quais o Problema da Coloração Total permanece em aberto.

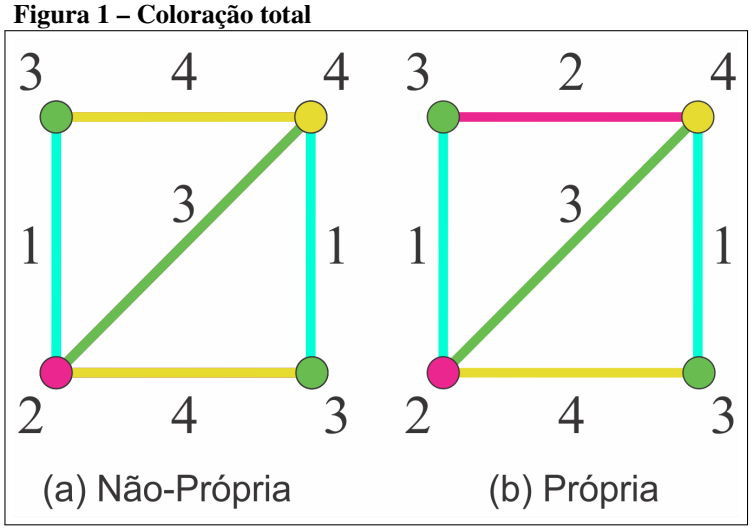
Para que esse texto seja autocontido, os conceitos fundamentais da Teoria dos Grafos e da Complexidade de Algoritmos são apresentados a seguir. Para mais detalhes, sugerimos os livros de Bondy, Murty *et al.* (1976), de Chartrand e Zhang (2008) e de Cormen *et al.* (2012).

Seja $G = (V(G), E(G))$ um grafo com um conjunto não vazio de vértices $V(G) = \{v_1, v_2, \ldots, v_n\}, n \in N$, e um conjunto de arestas $E(G) = \{e_1, e_2, \ldots, e_m\}, m \in N$, tal que $E(G)$ é um conjunto de pares não ordenados de elementos de $V(G)$. Nesse trabalho, denota-se $|V(G)| = n$, $|E(G)| = m$ e cada aresta cujos extremos são os vértices u e v por $\{u, v\}$. Toda aresta e todo vértice de um grafo G é um elemento de G. Os elementos do grafo são considerados adjacentes quando: i) são duas arestas distintas incidentes no mesmo vértice, ii) são dois vértices distintos que são extremos da mesma aresta, ou iii) são uma aresta e um de seus vértices extremos. Um conjunto de elementos que sejam dois a dois não-adjacentes é um conjunto independente. O grau de um vértice v em um grafo G é o número de arestas adjacentes a v no grafo G. O maior grau de um vértice do grafo G é chamado de grau máximo de G e denotado por $\Delta(G)$.

Uma coloração total de um grafo G é uma atribuição de cores para os elementos de G, definida por uma função $f : V(G) \cup E(G) \longrightarrow C$, onde C é um conjunto de cores. Uma coloração total é própria quando elementos adjacentes têm cores distintas. A Figura 1 apresenta dois exemplos de coloração total, onde o exemplo da esquerda é uma coloração total não própria

e o exemplo da direita é uma coloração total própria.

Figura 1 – Coloração total



Fonte: Autoria Própria

O Problema da Coloração Total é determinar qual é o menor número de cores para uma coloração total própria de um dado grafo G. Esse número de cores é chamado de número cromático total e denotado por $\chi''(G)$. Esse problema foi introduzido por Vizing (1964) e Behzad (1965) em trabalhos distintos, os quais relacionaram o número cromático total com o grau máximo do grafo, conforme mostra a seguinte conjectura.

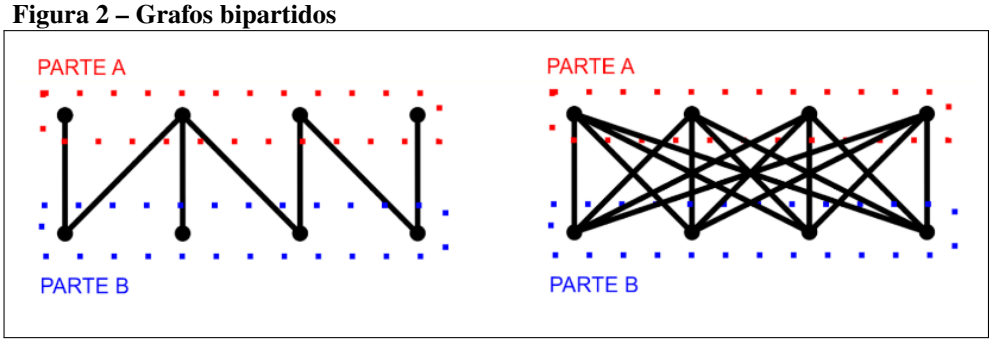
Conjectura 1. (VIZING, 1964; BEHZAD, 1965) Se G é um grafo simples, então $\chi''(G) \leq \Delta(G) + 2$.

Pela própria definição de coloração total própria, pode-se concluir que $\chi''(G)$ é pelo menos $\Delta(G) + 1$. Grafos que possuem número cromático total exatamente igual a $\Delta(G) + 1$ são chamados de tipo 1. Grafos com número cromático total igual a $\Delta(G) + 2$ são chamados de tipo 2. Se a Conjectura 1 for verdadeira, todo grafo é tipo 1 ou tipo 2.

Neste trabalho será abordado o Problema da Coloração Total em grafos bipartidos. Um grafo bipartido $G(A \cup B, E(G))$ é um grafo cujo conjunto de vértices pode ser particionado em dois conjuntos independentes, A e B. Um grafo bipartido completo é um grafo bipartido $G(A \cup B, E(G))$, em que cada vértice de A é adjacente a todos os vértices de B e é denotado por $K_{|A|,|B|}$. Os primeiros resultados sobre o Problema da Coloração Total foram obtidos por Behzad, Cooper e Chartrand (BEHZAD; CHARTRAND; JR, 1967), que determinaram o número cromático total para os grafos bipartidos completos. Exemplos de grafos bipartidos podem ser vistos na Figura 2. Observe que o grafo da Figura 2 a direita é um bipartido completo.

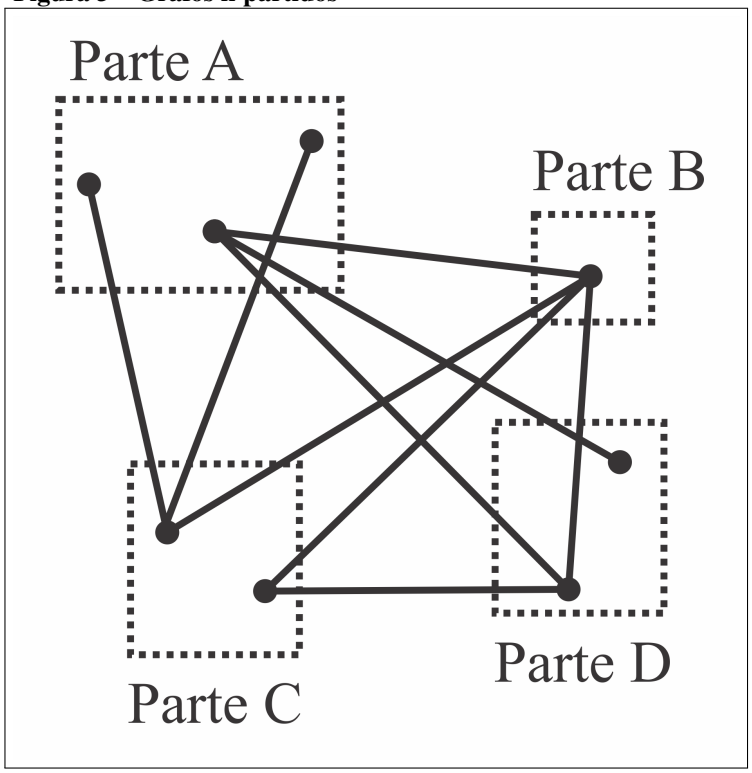
Uma generalização dos grafos bipartidos é o grafo k-partido, definido como grafo cujo conjunto de vértices pode ser particionado em k conjuntos independentes. A Figura 3 apresenta um exemplo de grafo 4-partido com uma parte de tamanho 1, duas partes de tamanho 2 e uma parte de tamanho 3.

Figura 2 – Grafos bipartidos



Fonte: Autoria Própria

Figura 3 – Grafos k-partidos



Fonte: Autoria Própria

Observe que os grafos bipartidos são um subconjunto dos k-partidos em que $k = 2$. Portanto, os resultados apresentados a seguir para grafos k-partidos também se aplicam aos grafos bipartidos.

Grafos k-partidos completos são grafos k-partidos em que todos os vértices em partes distintas são adjacentes. Grafos k-partidos balanceados são grafos k-partidos cujas partes têm o mesmo tamanho. Bermond (BERMOND, 1974) determinou o número cromático total para todos os k-partidos completos balanceados. Em trabalhos distintos, Chew e Yap (1992) e Hoffman e Rodger (1992) mostraram que todo grafo k-partido completo tendo número ímpar de vértices é tipo 1.

Mais recentemente, surgiram novos resultados para subclasses de grafos bipartidos quando Campos (2006) determinou o número cromático total de grafos grade, grades parciais, grafos quase-escada e cubos k-dimensionais.

Embora essa proposta apresente alguns resultados sobre o Problema da Coloração Total em grafos bipartidos, essa é uma classe muito abrangente, que inclui todos os grafos que não possuem ciclos ímpares como subgrafos. Então, muitos outros resultados obtidos para o Problema da Coloração Total em classes de grafos que não estão diretamente relacionadas com os grafos bipartidos podem apresentar solução em subconjuntos desta classe. Então é interessante responder à seguinte questão: qual é o subconjunto dos grafos bipartidos para o qual não se conhece um algoritmo eficiente que resolva o Problema da Coloração Total? É essa pergunta que este trabalho investigou, determinando o estado da arte da Coloração Total em grafos bipartidos.

1.1 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado da seguinte forma. O Capítulo 2 apresenta conceitos fundamentais sobre complexidade computacional. O estudo da complexidade computacional é importante para se definir quão difícil é um problema computacional e o que são algoritmos eficientes. No contexto deste Trabalho de Conclusão de Curso, a Teoria da Complexidade Computacional é importante por auxiliar o leitor a entender por que o Problema da Coloração Total permanece sem solução computacional eficiente mesmo sendo estudado desde a década de 60 e por que uma solução computacional eficiente para este problema implica na solução de um dos problemas matemáticos mais desafiadores do milênio (CARLSON *et al.*, 2006). O Capítulo 3 mostra soluções do Problema da Coloração Total em subconjuntos dos grafos bipartidos. Nesse capítulo, são apresentados os resultados em classes de grafos muito conhecidas como árvores e ciclos pares. Ainda no Capítulo 3, é apresentada uma extensão de resultados em coloração de arestas para a solução do Problema da Coloração Total em alguns grafos bipartidos. O Capítulo 4 mostra casos mais específicos e com soluções mais complexas. Dada a dificuldade de se resolver o Problema da Coloração Total em grafos cúbicos (aqueles em que todo vértice tem grau 3), são considerados grafos bipartidos em que todo vértice tem grau no máximo 3, além de grades e grades parciais, que são grafos com estrutura bastante rígida e com grau no máximo 4. As soluções apresentadas nesses dois capítulos resultam em algoritmos polinomiais para uma coloração total ótimo desses grafos. O Capítulo 5 discute os casos resolvidos e os casos em aberto do Problema da Coloração Total em grafos bipartidos.

2 COMPLEXIDADE COMPUTACIONAL

Este capítulo apresenta conceitos fundamentais sobre complexidade computacional. O estudo da complexidade computacional é importante para se definir quão difícil é um problema computacional e o que são algoritmos eficientes. Esse capítulo é baseado no livro de Thomas Cormen (CORMEN *et al.*, 2009), além de outras referências que serão citadas no decorrer do texto. O objetivo desse capítulo não será resolver problemas computacionais, mas utilizar alguns desses problemas para mostrar que existem necessidades computacionais que vão além da resolução dos mesmos.

2.1 ANÁLISE DE ALGORITMOS

A análise de algoritmos trata de dois pontos chaves para um dado problema computacional (FEOFILOFF, 2013):

- Provar que o algoritmo está correto.
- Estimar o tempo que a execução do algoritmo consome.

Algoritmo é um procedimento para executar determinada tarefa, é a ideia por trás de qualquer programa de computador. Um algoritmo deve resolver um problema genérico e bem especificado. Um problema algorítmico é especificado pela descrição de um conjunto completo de instâncias que devem ser tratadas e as respectivas saídas que devem ser geradas após a execução do algoritmo (SKIENA, 1998). A complexidade de um algoritmo é o custo que pode ser mensurado pelo tempo de execução, espaço de memória necessário para execução, utilização de disco, consumo de energia, ou qualquer recurso ou unidade relevante que o algoritmo utilize para resolver um problema (WILF, 1994). Na Seção 2.1 serão introduzidos conceitos referentes a análise de algoritmos.

O problema algorítmico citado por Skiena (1998) é denominado no livro de Cormen *et al.* (2012) como problema computacional. Um exemplo de problema computacional é o *Problema da Ordenação*, onde dada uma sequência de n números, busca-se obter como saída uma permutação da sequência de entrada, em geral em ordem não-decrescente de seus termos. Por exemplo dada a sequência $< 3, 2, 4, 1, 5 >$ o algoritmo deve retornar a mesma sequência reordenada $< 1, 2, 3, 4, 5 >$. Uma sequência dada como entrada é chamada de instância do problema. Um algoritmo é dito correto se para cada instância possível ele devolve a saída correta ou esperada. Mesmo se os computadores tivessem recursos infinitos a corretude ainda seria razão para a análise de algoritmos, uma vez que a função de um algoritmo é resolver um problema computacional corretamente.

O mesmo problema computacional pode ser resolvido por mais de um algoritmo, e esses algoritmos podem ter diferenças mais significativas que diferenças de hardware. Unidades de custo computacional, como memória, influenciam na complexidade de um algoritmo. Mas o conteúdo referente a hardware, máquina ou autômato foge do escopo deste trabalho, uma vez que a análise dos algoritmos tratada nesse trabalho se baseia em tempo de execução. A pegunta a ser respondida é: "mesmo na melhor máquina já inventada pelo homem seria possível construir um algoritmo que desse uma resposta para determinado problema em tempo hábil?".

Um exemplo interessante pode ser extraído do livro de Cormen *et al.* (2012). Suponha que um computador A execute 1 bilhão de instruções por segundo e um computador B execute 10 milhões de instruções por segundo. Assim o computador A será 100 vezes mais rápido que o computado B em capacidade de computação. Agora suponha que serão utilizados dois algoritmos para o problema da ordenação, onde o algoritmo X exija $2n^2$ instruções para ordenar n números, e o algoritmo Y exija $50n \log_2 n$ instruções para ordenar os mesmos n números. Para ordenar um milhão de números o Computador A demora:

$$\frac{2.(10^6)^2 \text{ instruções}}{10^9 \text{ intruções/segundo}} = 2000 \text{ segundos}$$

Enquanto o Computador B demora:

$$\frac{50.(10^6) \log_2 10^6 \text{ instruções}}{10^7 \text{ instruções/segundo}} = 100 \text{ segundos}$$

Mesmo o computador A sendo 100 vezes mais rápido que o computador B em capacidade de computação, o computador B foi 20 vezes mais rápido que o computador A para resolver o mesmo problema. Essa diferença fica mais evidente se a quantidade de números a serem ordenados aumentar significativamente. Por exemplo com 10 milhões de instruções, o computador A demoraria:

$$\frac{2.(10^7)^2 \text{ instruções}}{10^9 \text{ instruções/segundo}} = 200000 \text{ segundos}$$

Enquanto o Computador B demoraria:

$$\frac{50.(10^7) \log_2 10^7 \text{ instruções}}{10^7 \text{ instruções/segundo}} \cong 1163 \text{ segundos}$$

Enquanto o computador A demora mais de 2 dias para dar a saída, o computador B demora menos de 20 minutos para dar a mesma resposta. Suponha que o algoritmo X do exemplo anterior seja o famoso Algoritmo de Ordenação por Inserção. Para analisar o tempo de execução deste algoritmo será necessário analisar seu pseudocódigo e entender como funciona a contagem de instruções necessárias para um algoritmo apresentar a saída esperada. Dada uma sequência

de n números como entrada $(a_1, a_2, \ldots, a_n)$, o Algoritmo de Ordenação por Inserção retorna uma permutação $(a'_1, a'_2, \ldots, a'_n)$, tal que $a'_1 \leq a'_2 \leq \ldots a'_n$. Cada elemento da instância ou número a ser ordenado é chamado de chave. Neste trabalho todos os algoritmos serão descritos utilizando um pseudocódigo, muito semelhante a linguagens de programação em vários aspectos, a diferença entre um pseudocódigo e um código em alguma linguagem de programação é que no pseudocódigo existe uma semelhança com as linguagens naturais, sem a necessidade de uma sintaxe robusta com as sintaxes de linguagens de programação. A intenção de um pseudocódigo é ser entendida por qualquer pessoa sem a necessidade de um conhecimento prévio em programação em alguma linguagem.

O pseudocódigo que descreve o Algoritmo de Ordenação por Inserção é apresentado no Algoritmo 1 e será chamado de INSERTION-SORT. Ele toma como parâmetro uma sequência de n elementos $A[1 \ldots n]$ que será a entrada do algoritmo, onde n será o comprimento da sequência.

Algoritmo 1: INSERTION-SORT

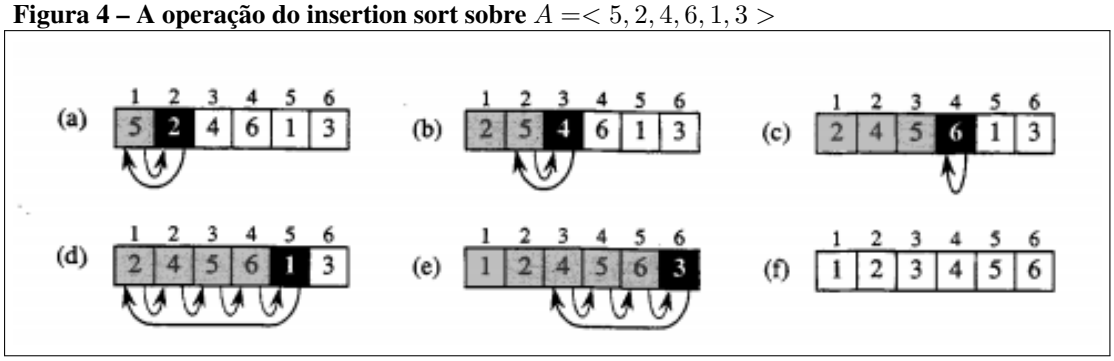
```
   Entrada: A[], n
1  início
2      para j ← 2 até n faça
3          chave ← A[j]
4          ▷Inserir A[j] na sequência ordenada A[1 . . . j − 1]
5          i ← j − 1
6          enquanto i > 0 e A[i] > chave faça
7              A[i + 1] ← A[i]
8              i ← i − 1
9          fim
10         A[i + 1] ← chave
11     fim
12 fim
13 retorna A
```

Fonte: Cormen *et al.* (2012)

Note que no Algoritmo 1 o valor de j na linha 2 inicia em 2 e vai até o comprimento da sequência, levando em conta que não é necessário executar o laço para j igual a 1, uma vez que se existe apenas um elemento na sequência, ela já está ordenada.

Suponha que existe uma instância $A =< 5, 2, 4, 6, 1, 3 >$, a Figura 4 mostra a execução do algoritmo 1 para esta instância específica, onde as sequências que vão de a até f representam cada interação do laço que vai da linha 1 até a linha 12 do Algoritmo 1. O quadrado preto representa a chave obtida na linha 3 ou o "número atual"que deve ser inserido na posição correta

Figura 4 – A operação do insertion sort sobre $A =< 5, 2, 4, 6, 1, 3 >$



Fonte: (CORMEN *et al.*, 2009)

da sequência. Os quadrados cinzas representam a sequência já ordenada e são definidos como $A[1 \ldots j-1]$, presentes no comentário da linha 4, e por fim as setas representam para onde cada elemento será realocado. A sequência f representa o conjunto A já reordenado.

Os elementos ainda não ordenados da sequência são representados por $A[j+1 \ldots n]$. As propriedades referentes ao montante ordenado são enunciadas como invariantes de laço e são utilizadas para provar que o algoritmo está correto. A invariante de laço deve ser garantida em um prova por indução, devendo-se manter verdadeira antes, durante e depois da execução do respectivo laço, como mostrado a seguir:

1. Inicialização: a invariante do laço é verdadeira antes da primeira interação do laço.
2. Manutenção: se a invariante do laço for verdadeira antes de uma interação do laço, ela permanecerá verdadeira antes da próxima interação.
3. Término: Quando o laço termina, a invariante do laço fornece uma informação útil que ajuda a mostrar que o algoritmo é correto.

Quando as duas primeiras opções são verdadeiras consequentemente a invariante do laço é verdadeira antes mesmo da última interação do laço. Para o Algoritmo 1, as etapas acima são descritas abaixo:

1. Inicialização: ocorre logo após o momento em que é atribuído o valor inicial de j na linha 2, onde $j = 2$, nesse momento o único elemento de A contido em $A[1 \ldots j-1]$ é $A[1]$, que por consequência já está ordenado uma vez que é um único elemento. Isso mostra que a invariante do laço é verdadeira antes da primeira interação.
2. Manutenção: agora é necessário que cada interação mantenha a invariante do laço. Dentro do corpo do laço externo da linha 2, o deslocamento acontece da direita pra esquerda enquanto o valor da chave for menor que o elemento que está sendo comparado no laço interno da linha 6 e o índice for diferente de zero, ou seja o elemento $A[j]$ é deslocado para $A[j-1]$, $A[j-2]$, $A[j-3]$, ... até encontrar a posição correta que pode acontecer da primeira interação do laço interno até a última. Assim que a posição correta para inserção é

encontrada, a linha 10 é executada. Sendo assim, a invariante do laço se mantém verdadeira antes da próxima interação.

3. Término: Com as duas opções acima verdadeiras, a invariante do laço é verdadeira. O laço externo termina quando j excede o valor do comprimento da sequência, ou seja quando $j = n + 1$. O sub arranjo $A[1 \ldots n]$ consiste nos elementos originalmente contidos em $A[1 \ldots n]$ mas em sequência ordenada. Assim, o subarranjo continua inteiro, sem nenhum elemento faltante. O que significa que o algoritmo é correto.

Como dito na Seção 2.1, existem várias unidades de custo e cada uma delas influencia na complexidade do algoritmo. Algumas destas unidades são dependentes do modelo computacional utilizado para executar o algoritmo, como memória, espaço em disco, etc. Os modelos computacionais determinísticos podem simular um ao outro com perda de eficiência desconsiderável (PAPADIMITRIOU, 2003). Porém mesmo os modelos computacionais determinísticos sendo equivalentes em tempo de simulação, é necessário escolher um modelo para tornar a contagem de instruções concreta, nesse trabalho será escolhida uma configuração do modelo RAM, sigla que significa *Random Access Machine* que em português quer dizer máquina de acesso aleatório, esse modelo foi apresentado pela primeira vez por Cook Cook e Reckhow (1973), na configuração do modelo RAM apresentada neste trabalho existe um único processador e as instruções são executadas uma após a outra, sem instruções executadas em paralelo. A configuração utilizada contém instruções comuns em computadores reais como instruções aritméticas, instruções de movimentos de dados e de controle, cada uma com valor constante.

Os tipos de dados neste modelo RAM são inteiros ou de ponto flutuante. Também é limitado o tamanho de cada palavra de dados. Existem instruções em computadores reais que não existem no modelo RAM utilizado nesse trabalho. Por exemplo existem computadores que possuem instruções específicas para cálculo de uma potência. O modelo aqui utilizado também não irá tratar hierarquia de memória. Mesmo com todos essas divergências entre o modelo RAM utilizado para análise de algoritmos nesse trabalho com computadores utilizados atualmente ainda é possível realizar previsões muito próximas às de computadores reais.

O *tempo de execução* de um algoritmo, dada uma determinada entrada, será dado pelo número de instruções primitivas ou "etapas"executadas até o final da execução do algoritmo. É conveniente definir a noção de etapa ou passo para que esta seja a mais independente da máquina possível. Será convencionado que cada linha i levará um tempo constante, c_i, para ser executada. De maneira geral, essa convenção atende tanto o modelo RAM quanto os modelos computacionais reais.

A partir do modelo RAM e das convenções citadas acima, é possível analisar o Algoritmo 1 e contar a quantidade de instruções necessárias para que o mesmo retorne a saída correta. A tabela 1 contém o número das linhas do algoritmo com o custo e número de vezes que cada linha é executada, onde t_j é o número de vezes que a condição do laço da linha 6 é testada, para $j = 2, 3, \ldots, n$.

Tabela 1 – Contagem de instruções do Algoritmo Insertion Sort

Número da linha	custo	número de vezes
1		
2	c_2	n
3	c_3	$n-1$
4	0	$n-1$
5	c_5	$n-1$
6	c_6	$\sum_{j=2}^{n} t_j$
7	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8	c_8	$\sum_{j=2}^{n} (t_j - 1)$
9		
10	c_{10}	$n-1$
11		
12	0	

Fonte – Adaptação de Cormen *et al.* (2012)

.

O tempo de execução para uma entrada de tamanho n, denotado por $T(n)$, é dado pela soma dos tempos de execução das instruções executadas. Para calcular $T(n)$ no Algoritmo INSERTION-SORT, soma-se o produto das colunas custo com a coluna número de vezes como mostra a Tabela 1 e a Equação 2.1.

$$T(n) = c_2 n + c_3(n-1) + c_5(n-1) + c_6 \sum_{j=2}^{n} t_j + c_7(\sum_{j=2}^{n}(t_j - 1)) + c_8(\sum_{j=2}^{n}(t_j - 1)) + c_{10}(n-1) \quad (2.1)$$

O tempo de execução sofre alterações dependendo do tamanho de entrada, ou seja, ordenar quinhentos mil números pode demorar mais que ordenar quinhentos números. Porém mesmo com entradas iguais esse valor também pode ser diferente para o mesmo algoritmo. No exemplo do Algoritmo 1, se uma entrada A possuir o mesmo tamanho que uma entrada B, mas se a entrada A possuir metade dos números já ordenados e a entrada B não possuir nenhum dos elementos ordenados, então o algoritmo possuirá melhor desempenho executando para a entrada A do que para a entrada B. Sabendo disto, alguém pode projetar um algoritmo para ser mais eficiente considerando-se diferentes cenários, a saber, o melhor caso, o caso médio e o pior caso, considerando diferentes entradas. No Algoritmo 1, o melhor caso e o pior caso acontecem, respectivamente, quando a entrada está inteiramente ordenada e quando a entrada está ordenada de maneira contrária ao melhor caso,ou seja, quando os números estiverem ordenados de maneira decrescente, o caso médio depende da distribuição para entrada, ou seja com que probabilidade cada instância pode aparecer, essa distribuição pode ser uniforme, mas nem sempre isto acontece.

No melhor caso os números estariam ordenados e o algoritmo nunca entraria no laço, uma vez que $A[i] > chave$ sempre retornaria *falso*. A contagem de instruções do Algoritmo Insertion Sort no melhor caso pode ser ilustrada como mostra a Tabela 2. Sendo assim, o tempo

de execução no melhor caso pode ser descrito como a Equação 2.2. O tempo de execução é, portanto, uma função linear que pode ser escrita como $an+b$, onde os valores de a e b dependem dos custos das instruções c_i, $1 \leq i \leq 8$.

$$T(n) = c_2 n + c_3(n-1) + c_5(n-1) + c_6(n-1) + c_1 0(n-1)$$
$$= (c_2 + c_3 + c_5 + c_6 + c_{10})n - (c_3 + c_5 + c_6 + c_{10}) \quad (2.2)$$

Tabela 2 – Contagem de instruções do Algoritmo Insertion Sort no Melhor caso

Número da linha	custo	número de vezes
1		
2	c_2	n
3	c_3	$n-1$
4	0	$n-1$
5	c_5	$n-1$
6	c_6	$n-1$
7	c_7	0
8	c_8	0
9		
10	c_{10}	$n-1$
11		
12	0	

Fonte – Adaptação de Cormen *et al.* (2012).

No pior caso do algoritmo, a condição do laço da linha 6 é verdadeira para todos os elementos do vetor testados. Cada elemento $A[j]$ é comparado com todos os elementos em $A[1..j-1]$. Então, $t_j = j$. Substituindo os valores do somatório na Equação 2.1 de contagem de instruções, se obtém a equação do pior caso, como segue.

$$T(n) = c_2 n + c_3(n-1) + c_5(n-1) + c_6 \sum_{j=2}^{n} j + c_7(\sum_{j=2}^{n}(j-1)) + c_8(\sum_{j=2}^{n}(j-1)) + c_{10}(n-1)$$

$$= c_2 n + c_3(n-1) + c_5(n-1) + (c_6 + c_7 + c_8)\frac{(n+2)(n-1)}{2} - (c_7 + c_8 - c_{10})(n-1)$$

$$= \frac{(c_6 + c_7 + c_8)}{2}n^2 + (c_2 + c_3 + c_5 + \frac{c_6}{2} - \frac{c_7}{2} - \frac{c_8}{2} + c_{10})n - (c_3 + c_5 + c_6 + c_{10}) \quad (2.3)$$

que pode ser descrita como uma função quadrática $an^2 + bn + c$, onde os valores de a,b e c dependem dos custos de instrução c_i, $1 \leq i \leq 10$.

Para simplificar a análise, é ignorado o custo real de cada instrução, usando as constantes c_i para representar estes custos, sendo que mesmo estas constantes c_i oferecem mais detalhes

do que é necessário para a análise. A Tabela 3 mostra por que as constantes podem ser ignoradas sem perda significativa. A primeira coluna da tabela representa a *ordem de crescimento* ou taxa de crescimento do algoritmo, que é a função que representa o tempo de execução ignorando as constantes, cada coluna da primeira linha a partir da segunda coluna representa o número de instruções a serem executadas que vai de 10 a 100000. Supondo que um computador processa cem mil instruções em um segundo, ou seja, cada instrução vai ser processada em um microssegundo, então os valores da Tabela 3 estão todos em microssegundos.

Tabela 3 – Quadro de comparação de tempo de execução em microssegundos.

	10	**100**	**1000**	**10000**	**100000**
$\log_2 n$	3	7	10	13	17
$\sqrt{n}$	3	10	32	100	316
n	10	100	$10 * 10^2$	$10 * 10^3$	$10 * 10^4$
$n \log_2 n$	33	664	$99,65 * 10^2$	$13,28 * 10^4$	$16,60 * 10^5$
n^2	$1 * 100$	$10 * 10^3$	$10 * 10^5$	$10 * 10^7$	$10 * 10^9$
n^3	$10 * 10^2$	$10 * 10^5$	$10 * 10^8$	$10 * 10^{11}$	$10 * 10^{14}$

Fonte – Adaptação de Prestes (2011)

Vale lembrar que $1 * 10^6 \mu s = 1s$, sendo assim a mudança significativa não ocorre quando a constante varia na função. Suponha que existam dois algoritmos com a mesma finalidade, um algoritmo A e um algoritmo B, porém o algoritmo A tem uma taxa de crescimento logarítmica representada pela primeira linha da Tabela 3 e o algoritmo B tem uma taxa de crescimento cúbica representada pela última linha da tabela. Note que para pequenos valores de entrada a diferença não chega a ser discrepante, uma vez que para 10 instruções a diferença é de $997\mu s$, porém se o número de instruções aumenta, a diferença se torna significativa, sendo que para 100000 instruções a diferença de tempo de execução dos dois algoritmos é de mais de três décadas.

A análise de algoritmos compreende então a verificação de sua corretude quando executada para qualquer entrada válida e a estimativa de uso de recursos, como tempo de execução, por exemplo, para diferentes entradas. Com essa análise, é possível estimar o uso de recursos e decidir qual o melhor algoritmo quando se tem mais de uma opção de solução computacional.

2.2 NOTAÇÃO ASSINTÓTICA

Embora seja possível determinar um valor em unidades de tempo (microssegundos, por exemplo) para o tempo de execução $T(n)$ de um algoritmo considerando uma entrada específica, como foi feito na seção anterior, não é viável utilizar esses valores para determinar a eficiência do algoritmo, uma vez que diferentes entradas podem demandar tempos diferentes. Entradas de dimensões menores normalmente demandam poucos recursos computacionais para que um algoritmo apresente a solução. Mas para entradas suficientemente grandes, os recursos compu-

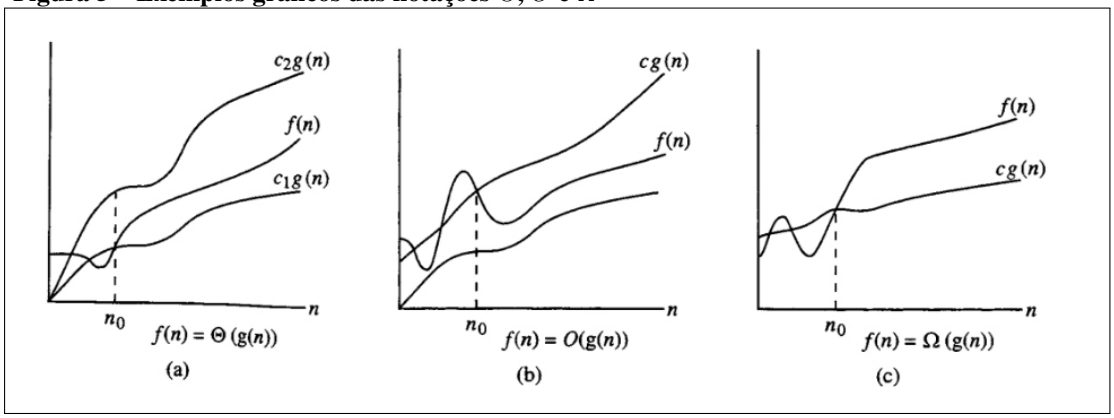
tacionais necessários podem inviabilizar o uso de determinados algoritmos, como pode-se ver na Tabela 3.

O comportamento de uma função qualquer $T(n)$, onde n é real e suficientemente grande, é chamado de comportamento assintótico da função $T(n)$. As notações assintóticas são usadas para descrever conjuntos de funções cujos comportamentos assintóticos apresentam alguma característica em comum, ou seja, funções $T(n)$ cujas curvas têm limitantes inferiores ou superiores ou ambos parecidos quando se considera grandes valores de n. Dentre as notações assintóticas, as mais comuns são O, Ω e Θ, as quais são definidas a seguir.

A notação $O(g(n))$ indica o conjunto das funções $f(n) \leq cg(n)$, para alguma constante $c > 0$ e para todo n a partir de algum valor constante $n_0 \geq 0$. Por definição, $f(n)$ é limitada superiormente, em todos os pontos a partir de n_0, por $cg(n)$. Por exemplo, as funções $n^3 + 2n^2$ e $n - 5$ pertencem ao conjunto $O(n^3)$, já que $n^3 + 2n^2 \leq 5n^3$ para todo $n \geq 1$ e $n - 5 \leq n^3$ para todo $n \geq 1$. Se $f(n)$ descreve o número de instruções executadas por um algoritmo (o tempo), então pode-se dizer que $f(n)$ consome um tempo que é no máximo $g(n)$, a menos de um fator constante. Se um Algoritmo A consome tempo $f(n)$, um Algoritmo B consome um tempo $g(n)$ e $f(n) \in O(g(n))$, então o Algoritmo A é melhor ou equivalente ao Algoritmo B em consumo de tempo, a menos de um fator constante.

A notação $\Omega(g(n))$ indica o conjunto das funções $f(n) \geq cg(n)$, para alguma constante $c > 0$ e para todo n a partir de algum valor constante $n_0 \geq 0$. Por definição, $f(n)$ é limitada inferiormente, em todos os pontos a partir de n_0, por $cg(n)$. Por exemplo, as funções $n^3 + 2n^2$ e $n - 5$ pertencem ao conjunto $\Omega(n)$, já que $n^3 + 2n^2 \geq n$ para todo $n \geq 1$ e $n - 5 \geq \frac{1}{2}n$ para todo $n \geq 10$. Se $f(n)$ descreve o tempo de execução de um algoritmo, então pode-se dizer que $f(n)$ consome um tempo que é pelo menos $g(n)$, a menos de um fator constante. Se um Algoritmo A consome tempo $f(n)$, um Algoritmo B consome um tempo $g(n)$ e $f(n) \in \Omega(g(n))$, então o Algoritmo A no máximo é tão bom quanto o Algoritmo B em consumo de tempo, a menos de um fator constante.

Figura 5 – Exemplos gráficos das notações Θ, O e Ω



Fonte: (CORMEN *et al.*, 2009)

A notação $\Theta(g(n)$ representa o conjunto de funções com ordem de crescimento similar a $g(n)$. Se uma função $f(n)$ pertence a $\Theta(g(n))$, significa que existem duas constantes c_1 e c_2

positivas e um valor inicial n_0 onde $n \geq n_0$, tais que a partir de um valor constante n_0 a função $f(n)$ tem os seguintes comportamentos:

- $f(n) \geq c_1 g(n)$
- $f(n) \leq c_2 g(n)$

Na Figura 5 o gráfico (a) ilustra o comportamento de uma função $f(n) = \Theta(g(n))$. Formalmente, $\Theta(g(n)) = \{f(n) :$ existem constantes $c_1 > 0, c_2 > 0$ e $n_0 \geq 0$ tais que todo $n \geq n_0$ satisfaz $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$. Diz-se que $g(n)$ é um limitante assintoticamente restrito para $f(n)$.

Suponha que $f(n) = \frac{1}{2}n^2 - 3n$. Usando definição formal é possível demonstrar que $f(n) \in \Theta(n^2)$ (também denota-se $f(n) = \Theta(n^2)$). É suficiente encontrar constantes c_1 e c_2 tais que

$$c_1 n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2 n^2. \tag{2.4}$$

É possível resolver a Inequação 2.4 dividindo a mesma em duas partes:

1. $c_1 n^2 \leq \frac{1}{2}n^2 - 3n$
2. $c_2 n^2 \geq \frac{1}{2}n^2 - 3n$

Primeiramente prova-se que a inequação é valida, dividindo a primeira parte por n^2, restando:

$$c_1 \leq \frac{1}{2} - \frac{3}{n}$$

Para $n < 7$ o valor de c_1 não assume valores positivos e como citado anteriormente, os valores de c_1 e c_2 são inteiros positivos. Ou seja, para que a inequação da primeira parte seja válida, tem-se $n \geq 7$. Sendo assim:

$$c_1 \leq \frac{1}{2} - \frac{3}{7} = \frac{1}{14}$$

Para a segunda inequação, observe que:

$$\frac{1}{2}n^2 - 3n \leq \frac{1}{2}n^2$$

para todo $n \geq 0$. Então, para $c_2 = \frac{1}{2}$, a segunda inequação é válida. Assim, tem-se $c_1 = \frac{1}{14}$, $c_2 = \frac{1}{2}$ e $n_0 = \max 7, 0 = 7$.

Pela definição da notação Θ, é possível mostrar que se $f(n) = 6n^3$, então $f(n) \notin \Theta(n^2)$. Suponha por contradição que exitam constantes positivas c_1 e c_2, e uma constante não negativa n_0 que atendam a Equação 2.4. É possível dividir o problema em duas partes:

1. $c_1 n^2 \leq 6n^3$
2. $c_2 n^2 \geq 6n^3$

Como feito anteriormente, na primeira parte da inequação é possível dividir por n^2:

$$c_1 \leq 6n$$

Essa inequação é verdadeira uma vez que, para um n consideravelmente grande, $6n$ sempre vai ser maior que uma constante. Neste caso, para valores de n maiores que $\frac{c_1}{6}$.

A segunda parte da inequação pode ser divida por n^2 também, obtendo-se a inequação abaixo:

$$c_2 \geq 6n$$

Note que a inequação acima não se sustenta uma vez que para um n muito alto a função terá valor maior que c_2. Basta considerar valores de n maiores que $\frac{c_2}{6}$.

Para casos como o mostrado acima em que $f(n) \geq cg(n)$ para alguma constante c positiva e $f(n) \nleq c'g(n)$ para qualquer que seja a constante positiva c', existe apenas um *limite assintótico inferior* como mostra o item (c) a Figura 5. Nesse caso $f(n) \in \Omega g(n)$. Caso contrário, se $f(n) \ngeq c'g(n)$ para qualquer constante positiva c' mas existe constante c tal que $f(n) \leq cg(n)$, existe apenas *um limite assintótico superior* como mostra o item (b) da Figura 5. Nesse caso, $f(n) \in O(g(n))$.

2.3 REDUÇÃO ENTRE PROBLEMAS

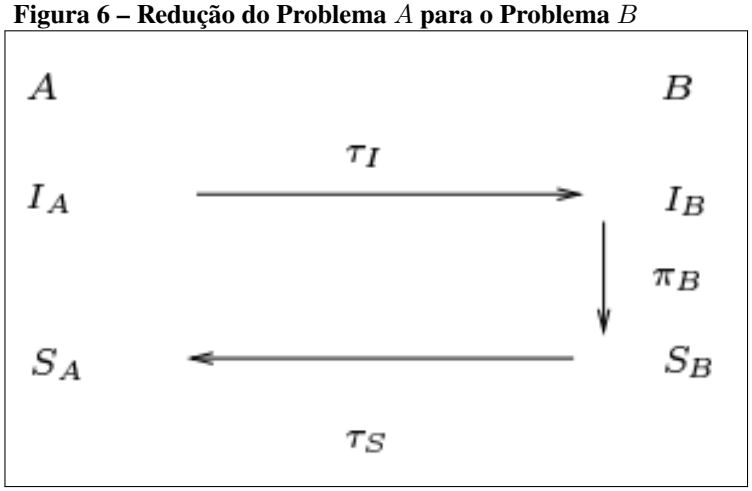
A redução entre problemas pode ser descrita como um procedimento para transformar um problema A em um problema B. Esse procedimento se torna útil quando se conhece um algoritmo que resolva um dos problemas. Suponha que o problema A é um problema para o qual não se conhece algoritmo polinomial e o problema B tem um algoritmo polinomial conhecido. Se for possível reduzir o problema A ao problema B utilizando um procedimento que consome tempo polinomial, então existe um algoritmo que resolve A em tempo polinomial.

Para uma definição formal considere que o problema A tem conjunto de instâncias I_A e conjunto de saídas S_A e que o problema B tem um conjunto de instâncias I_B e um conjunto de saídas S_B. Considere um par de algoritmos $\mathcal{T}_I$ e $\mathcal{T}_S$, tais que:

- $\mathcal{T}_I$ transforma cada instância do conjunto I_A em uma instância do conjunto I_B.
- $\mathcal{T}_S$ transforma cada elemento do conjunto S_B em um elemento do conjunto S_A.

Um problema A é redutível a um problema B, denotado por $A \propto_{f(n)} B$, se existe um algoritmo π_B que, dada uma entrada I_B, devolve a saída esperada S_B, e é possível realizar as transformações $\mathcal{T}_I$ e $\mathcal{T}_S$, como mostra a Figura 6.

Figura 6 – Redução do Problema A para o Problema B

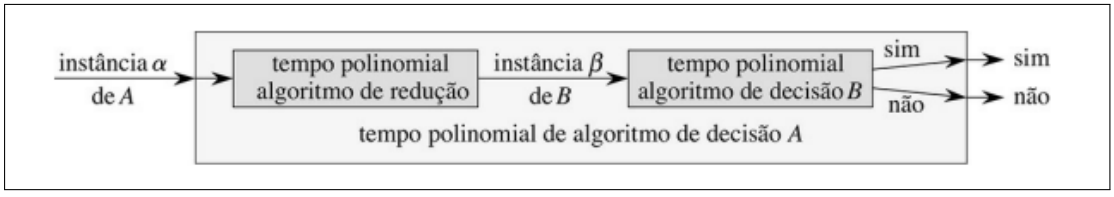


Fonte: Autoria Própria

A notação $A \propto_{f(n)} B$ também indica que a soma das complexidades de $\mathcal{T}_I$, $\mathcal{T}_S$ e de π_A é $O(f(n))$.

Grande parte dos problemas de interesse computacional são problemas de otimização, ou seja, tendo uma entrada qualquer, a saída esperada é um valor que expressa a melhor solução possível com o melhor valor (máximo ou mínimo, dependendo do problema). Um exemplo de problema de otimização é o Problema de Menor Caminho em um grafo, que consiste em, dado um grafo $G(V(G), E(G))$ e dois vértices $s \in V(G)$ e $v \in V(G)$ como entrada, determinar um caminho com o menor número de arestas que conecta s a v. Outro conjunto de problemas são os problemas de decisão, nos quais o conjunto de saídas possíveis se limita a "sim"ou "não"(ou qualquer outra forma de representação para os termos como 0 e 1 ou *true* ou *false*). Um exemplo é o Problema do k-caminho em que, dado um grafo G e um par de vértices s e v, deve-se responder se existe um caminho de tamanho k, ou seja, com k arestas entre os vértices s e v. Nesse caso, o conjunto de saídas possíveis se limita a "sim"e "não". A Figura 7 mostra como seria resolvido um problema de decisão A, utilizando as transformações citadas acima, ou seja reduzindo o problema A ao problema B.

Figura 7 – Processo de Redução de um problema de decisão A utilizando um problema de decisão B



Fonte: (CORMEN *et al.*, 2012, p.1050)

Um exemplo para entender a redução em tempo polinomial é a redução do problema da equação do primeiro grau no problema da equação do segundo grau. Na lista abaixo encontra-se

a definição de cada problema:

- O problema da equação do primeiro grau: dado um número arbitrário inteiro x, determinar se existe um valor para x tal que $ax+b = 0$ e $a \neq 0$, onde a e b são coeficientes da equação.
- O problema da equação do segundo grau: dado um número arbitrário inteiro x, determinar se existe um valor para x tal que $ax^2 + bx + c = 0$ e $a \neq 0$, onde a, b e c são coeficientes da equação.

Agora suponha que A seja o problema da equação do primeiro grau, logo o Algoritmo 2 transforma as instâncias de A em B. Note que este algoritmo executa em tempo $O(c)$ no pior caso, onde c é uma constante, levando em conta que tanto a atribuição quanto retornar o resultado consomem tempo do processador.

Algoritmo 2: ALGORITMO $\mathcal{T}_I$

```
Entrada: b,c,x
1 início
2 |   a ← 0
3 |   retorna a,b,c,x
4 fim
5 retorna A
```

Após a transformação das instâncias da entrada, para que a redução seja possível, é necessário um algoritmo π_B que retorne a solução do problema B recebendo as instâncias de A logo após a transformação, nesse caso o Algoritmo 3.

Algoritmo 3: ALGORITMO $\mathcal{T}_I$

Entrada: a,b,c,x

```
1  início
2      delta ← b² − 4 ∗ a ∗ c
3      se delta < 0 então
4          escreva("Não existem raizes reais.")
5          retorna falso
6      senão
7          se delta > 0 então
8              escreva("Duas raizes diferentes:")
9              x1 ← (−b + raizQ(delta))/2 ∗ a
10             x2 ← (−b − raizQ(delta))/2 ∗ a
11             escreva("Primeira raiz X'", x1)
12             escreval("Segunda raiz X"", x2)
13             retorna verdadeiro
14         senão
15             escreva("Duas raizes iguais:")
16             x1 ← (−b + raizQ(delta))/2 ∗ a
17             escreval("Raiz X'", x1)
18             retorna verdadeiro
19         fim
20     fim
21 fim
```

Após a execução desses dois algoritmos, é necessário que haja uma transformação $\mathcal{T}_S$, para se obter a resposta do Problema A. Note que I_A foi transformada em I_B e A e B são problemas de decisão. Como há uma função bijetora que mapeia os elementos do conjunto de resultados do problema B nos elementos do conjunto de resultados do problema A, a redução pode ser feita em tempo polinomial no pior caso, já que $\mathcal{T}_I$, $\mathcal{T}_S$ e o algoritmo π_b podem ser executados em tempo $O(n^k)$.

2.4 CLASSES DE COMPLEXIDADE

Essa seção é baseada no capítulo 34 do livro Algoritmos - Teoria e Prática (CORMEN *et al.*, 2012) e no livro Computers and intractability (GAREY; JOHNSON, 2002), as outras referências serão citadas no decorrer da seção. Como citado no começo da Seção 2.3, existem problemas considerados difíceis ou intratáveis e problemas considerados fáceis ou tratáveis. Este capítulo tem o objetivo de classificar esses problemas de maneira mais especifica e delimitar até

onde um problema pode ser fácil, difícil ou até mesmo aonde ficam os problemas com status indefinido.

A classe P é a que abrange todos os problemas que tem algoritmos com o pior caso em tempo polinomial, como por exemplo o Problema da Equação de Primeiro Grau e o Problema da Equação do Segundo Grau, para os quais foram apresentados algoritmos $O(n^k)$ na seção anterior.

A classe NP é a classe dos problemas verificáveis em tempo polinomial. Ou seja, se existir uma possível saída para um problema NP, existe um algoritmo polinomial no pior caso, capaz de verificar se essa saída está correta ou errada. Para essa verificação são necessários um algoritmo verificador e um certificado, onde um certificado é uma prova de que a resposta para o problema é "sim", se for o caso, e o algoritmo verificador é o algoritmo através do qual valida-se o certificado. Se o algoritmo que resolve o problema de decisão responder "sim", significa que o certificado apresentado é válido.

Observe que os problemas da classe P também são verificáveis em tempo polinomial, ou seja $p \subseteq NP$. Por exemplo, o Problema da Ordenação, tratado na Seção 2.1, pode ser resolvido por um algoritmo polinomial e é fácil imaginar que se existir um vetor é possível criar um algoritmo que verifica se esse vetor está ordenado ou não, percorrendo o vetor e comparando valores consecutivos, verificando se o valor mais a direita é sempre maior ou igual que o valor mais a esquerda.

Em 1971, Cook (1971) apresentou o seguinte problema: provar que as classes de problemas P e NP são equivalentes, ou seja, que contém exatamente os mesmos problemas. Note que isso é o mesmo que dizer que todo problema para os quais as saídas podem ser verificadas em tempo polinomial podem ser resolvidos também em tempo polinomial, no pior caso. Se convencionou a chamar esse problema de P vs NP. Por mais que os problemas P sejam verificáveis em tempo polinomial, isto não basta para afirmar que P = NP. É preciso provar que $NP \subseteq P$ ou que $NP \nsubseteq P$ e essa é a parte difícil do problema. O Instituto Clay de Matemática apresentou sete problemas matemáticos que considera os mais desafiadores deste milênio e os chamou de Problemas do Milênio. Um deles é o Problema P vs NP. A resolução de qualquer um deles resulta em um prêmio de 1 milhão de dólares (CARLSON *et al.*, 2006).

Existe ainda um subconjunto especial dos problemas em NP, chamado de Classe NP-Completo ou NPC. Um problema X é NP-Completo se pertence a NP e satisfaz a propriedade especial de que todo problema da classe NP pode ser reduzido a X em tempo polinomial. Dado um problema NP-Completo X, então por definição todo problema NP pode ser reduzido a X em tempo polinomial. Então, se X puder ser reduzido em tempo polinomial a Y, um outro problema NP, pode-se concluir que Y também é NP-Completo. Então, para provar que um problema A é NP-Completo é suficiente mostrar que A pertence a NP e que algum problema NP-Completo pode ser reduzido polinomialmente a A. Quando todo problema NP pode ser reduzido a um problema X e X não pertence a NP, dizemos que X é NP-difícil.

Não se sabe da existência ou não de algoritmos com complexidade de tempo polinomial

que resolvam algum dos problemas NP-completos. Mas, por definição, se algum problema NP-completo puder ser resolvido em tempo polinomial, então todos os problemas em NP tem um algoritmo de tempo polinomial, o que implicaria que $P = NP$.

2.5 COMPLEXIDADE DO PROBLEMA DA COLORAÇÃO TOTAL

Para os grafos em geral, McDiarmid e Sánchez-Arroyo (1994) mostraram que decidir se o grafo é tipo 1 é um problema NP-Completo, neste mesmo trabalho mostraram que o problema da coloração total é NP-Completo para os bipartidos cúbicos, que são grafos bipartidos onde todos os vértices tem grau 3. Por esse resultado, pode-se concluir que a existência de um algoritmo polinomial que resolva o Problema da Coloração Total ainda que apenas para os grafos bipartidos cúbicos, é suficiente para provar que P = NP, resolvendo o Problema P vs NP proposto em Carlson *et al.* (2006).

McDiarmid e Sánchez-Arroyo (1994) também mostraram que determinar o número cromático total é NP-Difícil mesmo para grafos bipartidos em que todos os vértices têm grau k, onde k é um valor fixo maior ou igual a 3. Uma corda é uma aresta entre vértices não consecutivos de um ciclo no grafo. Em 2011, Machado e Figueiredo (2011) provaram que o Problema da Coloração Total para um bipartido livre de corda única é NP-Completo. Então, neste caso também vale a observação de que a existência de um algoritmo polinomial que resolva o Problema da Coloração Total para os grafos bipartidos livres de corda única implicaria em solução para o Problema P vs NP.Um grafo livre de corda única é um grafo que não possui um ciclo com um única corda, corda é uma aresta que liga dois vértices não consecutivas de um ciclo.

Embora existam algumas evidências de que o Problema da Coloração Total é difícil, também existem trabalhos que apresentam algoritmos polinomiais para solução desse problema em alguns subconjuntos de grafos bipartidos, como será visto nos próximos capítulos.

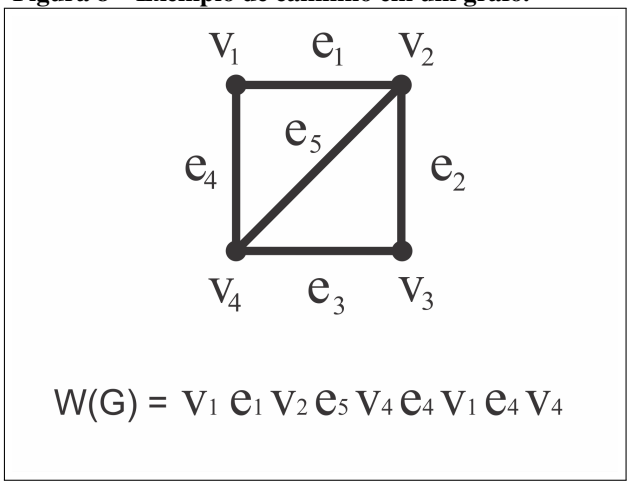
3 RESULTADOS SEMINAIS DE COLORAÇÃO TOTAL EM GRAFOS BIPARTIDOS

Neste capítulo serão abordados os casos em que o Problema da Coloração Total é resolvido em tempo polinomial com técnicas bem difundidas e de maneira intuitiva. Esse capítulo é baseado no livro Graph theory with applications de Bondy, Murty *et al.* (1976). Outras referências serão citadas no decorrer do texto.

A seguir, sãs apresentados alguns conceitos fundamentais da Teoria dos Grafos que são importantes para a compreensão deste capítulo.

Um passeio $W := v_0e_1v_1 \dots v_{\ell-1}e_\ell v_\ell$ em um grafo G é uma sequência de vértices e arestas tal que os vértices v_{i-1} e v_i são extremos de e_i, $1 \leq i \leq \ell \in \mathbb{N}$, as arestas e vértices aparecem de forma alternada e podem aparecer repetidamente na sequência. A Figura 8 mostra um exemplo de passeio em um grafo. Note que os vértices v_1 e v_4 assim como a aresta e_4 aparecem repetidamente na sequência. Um passeio é fechado se começa e termina no mesmo vértice.

Figura 8 – Exemplo de caminho em um grafo.



Fonte: autoria própria

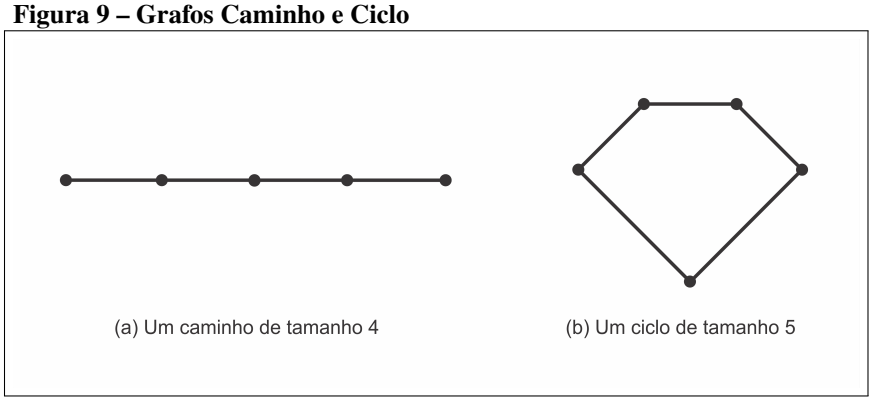
Uma trilha é um passeio em que não se repetem arestas na sequência, ou seja, pode-se repetir vértices contanto que não se repitam arestas. Na Figura 8, uma trilha é v_4 e_4 v_1 e_1 v_2 e_5 v_4 e_3 v_3. Note que o vértice v_4 aparece mais de uma vez, porem nenhuma aresta se repete. Uma trilha fechada, é uma trilha que começa e termina no mesmo vértice. Trilhas fechadas são também chamadas de circuito.

Um caminho é um grafo cujos vértices podem ser arranjados em uma sequência linear de maneira que dois vértices são adjacentes se eles são consecutivos na sequência. Um caminho é um passeio em que não se repetem arestas nem vértices. Um ciclo em um grafo é um caminho fechado, ou seja, o último vértice da sequência é igual ao primeiro vértice da sequência.

Não se deve confundir um caminho em um grafo com o grafo caminho. O grafo caminho P_n é um grafo com n, $v_1, v_2, \dots, v_n$ tal que existe aresta entre dois vértices v_i e v_j se e só se $i = j-1, 2 \leq j \leq n$. Da mesma forma, não se deve confundir um ciclo em um grafo com o grafo ciclo. O grafo ciclo com n vértices, denotado por C_n, é um grafo cujo conjunto de vértices tem

tamanho n e tal que todo vértice do grafo tem grau igual a 2. O grafo ciclo C_1 corresponde a um laço, e o grafo C_2 corresponde a um grafo com arestas múltiplas, onde existem exatamente duas arestas e elas têm os mesmos dois vértices como extremos. Na Figura 9 tem-se a representação dos grafos P_4 e C_5. Um caminho ou ciclo de tamanho k é chamado de k-caminho ou k-ciclo respectivamente. Um dado ciclo é par ou impar dependendo da paridade do k, ou seja se k for par, o ciclo é par, e se k for ímpar, o ciclo é ímpar.

Figura 9 – Grafos Caminho e Ciclo



Fonte: autoria própria

Para a escolha das classes de grafos que são apresentadas neste capítulo, é importante conhecer a seguinte caracterização dos grafos bipartidos.

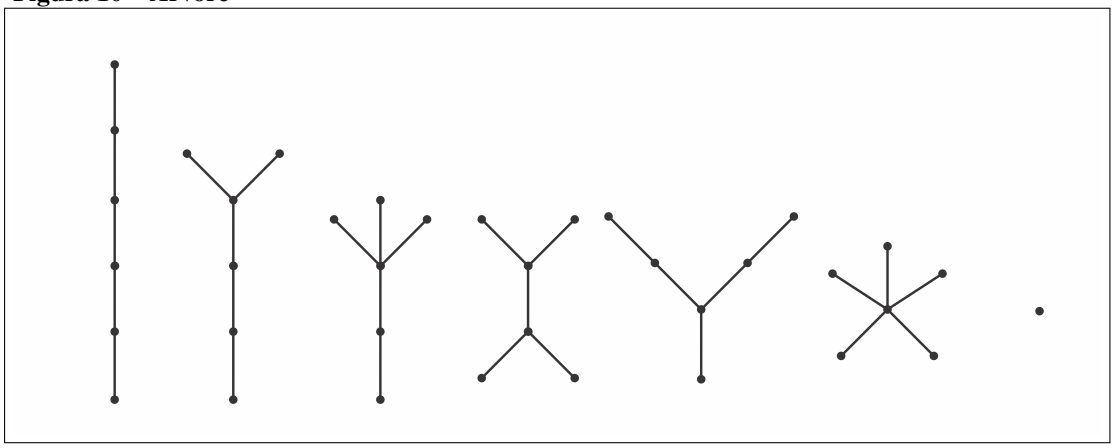
Teorema 1. (BONDY; MURTY *et al.*, 1976) Um grafo é bipartido se, e somente se, não contém ciclos ímpares.

Pela caracterização dos grafos bipartidos apresentada no Teorema 1, os grafos que são considerados neste trabalho apresentam uma das seguintes propriedades: ou não contém ciclos como subgrafos ou, caso tenham ciclos, estes devem ter um número par de vértices. Os grafos que não contém ciclos como subgrafos são chamados de árvores. Pela caraterização apresentada no Teorema 1, toda árvore é um grafo bipartido. A coloração total em árvores é apresentada na Seção 3.1. Todo grafo bipartido que não é uma árvore deve, portanto, conter um ciclo e, pela definição de grafos bipartidos, este ciclo contém um número par de vértices. A Seção 3.2 mostra que o Problema da Coloração Total está resolvido para ciclos pares. Entretanto, existem grafos bipartidos que contém ciclos pares (não são árvores) mas não são eles próprios grafos ciclos. Para estes, nem sempre se conhece solução do Problema da Coloração Total. O final desse capítulo mostra alguns casos em que é possível determinar o número cromático total desses grafos a partir de resultados conhecidos para um outro problema de coloração, o Problema da Coloração de Arestas, que será definido na Seção 3.3.

3.1 ÁRVORES

Uma árvore é um grafo conexo e acíclico, ou seja, existe um caminho entre qualquer par de vértices $\{u, v\}$ e o grafo não possui ciclos. Na Figura 10 são dados exemplos de árvores. Note que um vértice isolado também é uma árvore.

Figura 10 – Árvore



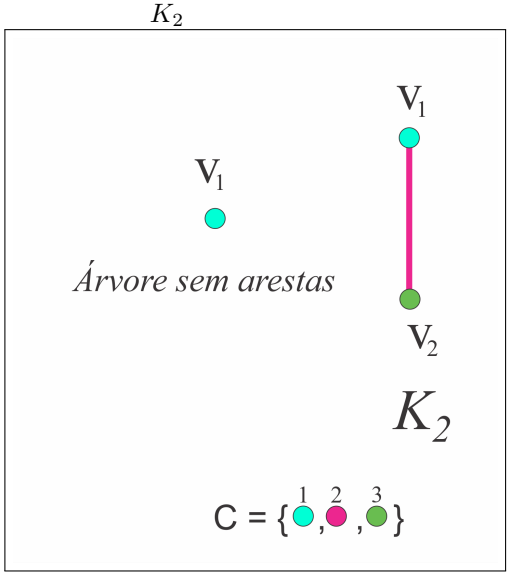
Fonte: autoria própria

O Problema da Coloração Total está resolvido para árvores. Várias demonstrações desse resultado estão disponíveis na literatura. A prova a seguir foi apresentada por Campos (2006, p.64-65).

Teorema 2. Toda árvore é Tipo 1, exceto o grafo K_2, que é Tipo 2.

Demonstração. Seja T uma árvore. Se T não tem arestas, então T é tipo 1. Se T é o grafo K_2, então T é tipo 2, como mostra a Figura 11.

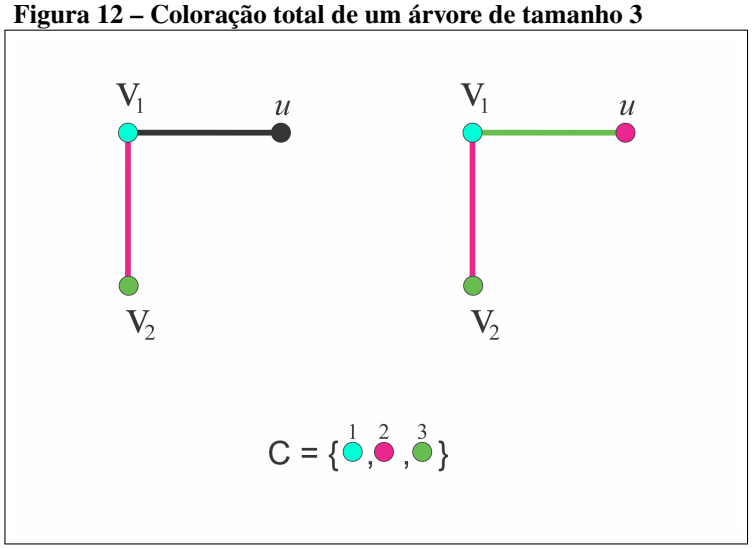
Figura 11 – Árvore sem arestas e Árvore K_2



Fonte: autoria própria

Suponha agora que $\Delta(T) \geq 2$ e seja $u \in V(T)$, um vértice de grau 1. Seja $T' := T - u$. Se T' é o grafo K_2, então T' é tipo 2 e pode-se facilmente estender qualquer coloração total ótima do grafo K_2 para uma 3-coloração total de T sem adicionar novas cores, como mostra a Figura 12.

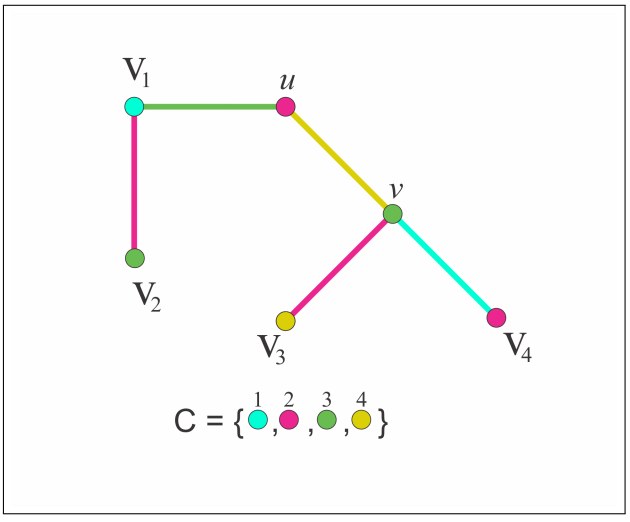
Figura 12 – Coloração total de um árvore de tamanho 3



Fonte: autoria própria

Resta considerar o caso em que T' não é isomorfo a um K_2. Por hipótese de indução, existe uma $(\Delta(T') + 1)$-coloração total para T'. Se a adição da aresta incidente em u aumentar $\Delta(T')$, pode-se colorir a aresta adicionada com uma nova cor e ainda tem-se um grafo tipo 1. Então, considere o caso em que $\Delta(T') = \Delta(T) - 1$ e seja v o vértice de T adjacente a u. Então v é uma vértice de grau máximo de T'. Portanto existe uma cor usada da coloração total de T' e que está faltando em v. Atribui-se a cor faltante à aresta uv, como mostra a Figura 13. Finalmente, em ambos os casos, atribuímos ao vértice u uma cor diferente das cores de uv e v.

Figura 13 – Coloração total de um árvore de tamanho 6



Fonte: autoria própria

□

Note que a prova apresentada para a coloração total de árvores possibilita a construção de um algoritmo recursivo para uma coloração total ótima de árvores com complexidade $O(m)$, onde m é o número de arestas da árvore. Para a versão de decisão do Problema da Coloração Total em árvores, a solução pode ser obtida em tempo constante, já que é suficiente verificar se o grafo é um K_2 para responder sim ou não para a possibilidade de se obter uma coloração total com qualquer número de cores.

3.2 CICLOS PARES

Pelo Teorema 1, um grafo ciclo C_n é bipartido se, e somente se, n é par. Então, esta seção concentra-se em apresentar uma coloração total ótima para grafos ciclos pares.

Em 1965 em sua tese Behzad afirma sem apresentar a prova, que ciclos com tamanho (número de vértices) múltiplo de 3 têm número cromático total igual a 3 (BEHZAD, 1965) .

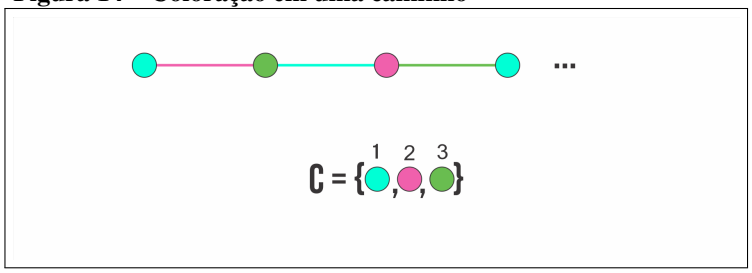
Lema 1. (BEHZAD, 1965) Se C_n tem $n \equiv 0 \pmod 3$, então C_n é tipo 1.

Demonstração. Considere o ciclo C_n, $n \equiv 0 \pmod 3$, com conjunto de vértices $V(C_n) = \{v_1, v_2, \ldots, v_n\}$ e conjunto de arestas $E(C_n) = \{v_i v_{i+1} : 1 \leq i < n\} \cup \{v_n v_1\}$. É suficiente apresentar uma 3-coloração total para o grafo C_n. Pinte o vértice v_i, $1 \leq i \leq n$, com cor 1 se $i \equiv 1 \pmod 3$, com cor 2 se $i \equiv 0 \pmod 3$, e com cor 3 se $i \equiv 2 \pmod 3$. Pinte a aresta $v_i v_{i+1}$, $1 \leq i < n$, com cor 2 se $i \equiv 1 \pmod 3$, com cor 1 se $i \equiv 2 \pmod 3$, e com cor 3 se $i \equiv 0 \pmod 3$. Pinte a aresta $v_n v_1$ com cor 3.

Como esta é uma coloração total válida, $\chi''(C_n) = \Delta(C_n) + 1 = 3$. Portanto, se $n \equiv 0 \pmod 3$, C_n é tipo 1. □

A Figura 14 mostra como a sequência das cores se alterna na coloração total apresentada no Lema 1.

Figura 14 – Coloração em uma caminho



Fonte: autoria própria

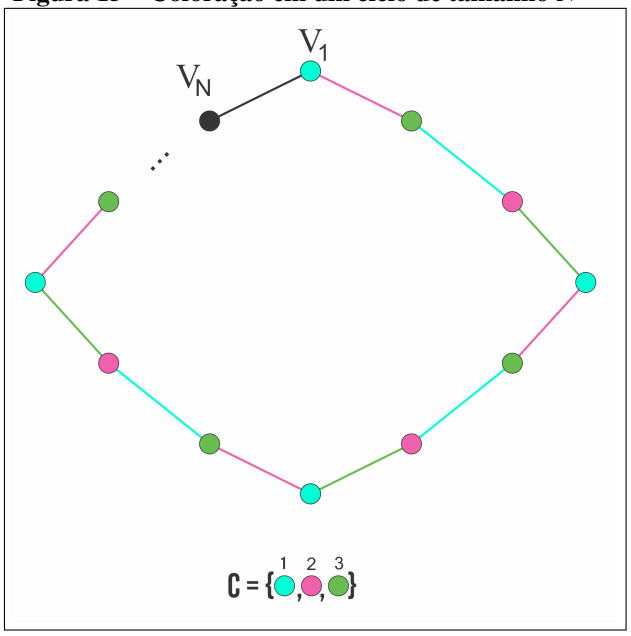
Agora, vamos provar que quando o ciclo C_n tem $n \not\equiv 0 \pmod 3$, não existe uma 3-coloração total para o grafo.

Lema 2. Se C_n tem $n \not\equiv 0 \pmod 3$, então $\chi''(C_n) > 3$.

Demonstração. Considere o ciclo C_n, $n \not\equiv 0 \pmod 3$, com conjunto de vértices $V(C_n) = \{v_1, v_2, \ldots, v_n\}$ e conjunto de arestas $E(C_n) = \{v_i v_{i+1} : 1 \leq i < n\} \cup \{v_n v_1\}$. É suficiente mostrar que é impossível apresentar uma 3-coloração total para o grafo C_n.

Por contradição, suponha que C_n tem uma coloração total com 3 cores. Sem perda de generalidade, assuma que v_1 está colorido com cor 1, v_1v_2 está colorida com cor 2 e v_2 está colorido com cor 3 (como esses elementos são dois a dois adjacentes, em qualquer coloração total, seriam coloridos com três cores distintas). Assim, a aresta $v2v3$ não está colorida com cor 2 pois é adjacente à aresta v_1v_2, nem com cor 3 pois é adjacente a v_3. Como é uma 3-coloração total, $v2v_3$ está colorida com cor 1. Adotando o mesmo raciocínio para analisar os elementos de C_n em sequência: agora v_3, depois v_3v_4, depois v_4 e assim sucessivamente, é possível identificar de forma única qual a cor usada para colorir cada um desses elementos. A coloração é como a apresentada na Figura 15.

Figura 15 – Coloração em um ciclo de tamanho N



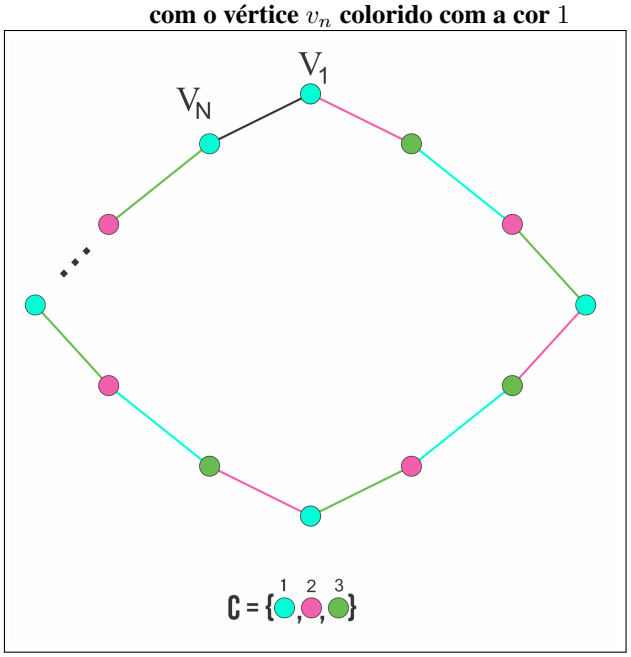
Fonte: autoria própria

A cor do vértice v_n depende do valor de n. Como $n \not\equiv 0 \pmod 3$, há dois casos: ou $n \equiv 1 \pmod 3$ ou $n \equiv 2 \pmod 3$.

Caso 1: $n \equiv 1 \pmod 3$. Nesse caso, v_n está colorida com cor 1. Como v_n é adjacente a v_1 e v_1 também está colorido com cor 1, esta não é uma coloração total própria do C_n, um absurdo. A Figura 16 mostra esta coloração do ciclo C_n.

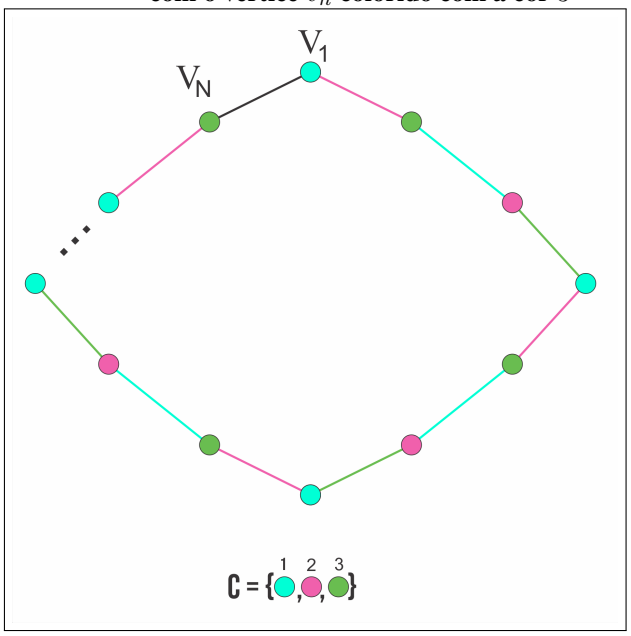
Caso 2: $n \equiv 2 \pmod 3$. Nesse caso, v_n está colorida com cor 3 e v_nv_1 está colorida com cor 1. Como v_nv_1 é adjacente a v_1 e v_1 também está colorido com cor 1, esta não é uma coloração total própria do C_n, um absurdo. A Figura 17 mostra esta coloração total do ciclo C_n.

Figura 16 – Coloração em um ciclo de tamanho n com o vértice v_n colorido com a cor 1



Fonte: autoria própria

Figura 17 – Coloração em um ciclo de tamanho n com o vértice v_n colorido com a cor 3



Fonte: autoria própria

Em ambos os casos, não existe uma 3-coloração total do ciclo C_n. Portanto, $\chi''(C_n) > 3$. □

Observe que os ciclos são grafos em que todos os vértices tem grau 2. Um grafo G é k-regular (ou regular de grau k) quando todos os seus vértices têm grau igual a k. Em 1971 Vijayaditya provou que o número cromático total de todos os grafos k-regulares é no máximo 4 (VIJAYADITYA, 1971).

Teorema 3. (VIJAYADITYA, 1971) Se G é k-regular, então $\chi''(G) \leq 4$.

A seguir, vamos apresentar a prova do Teorema 3 para o caso dos grafos 2-regulares. Observe que um grafo é 2-regular se, e somente se, é um ciclo.

Lema 3. Se G é grafo 2-regular, então $\chi''(G) \leq 4$

Demonstração. Considere um grafo 2-regular, ou seja, um ciclo C_n. A atribuição de uma cor a cada vértice e aresta será dada por uma função sobrejetora $f : V(C_n) \cup E(C_n) \rightarrow \{1, 2, 3, 4\}$.

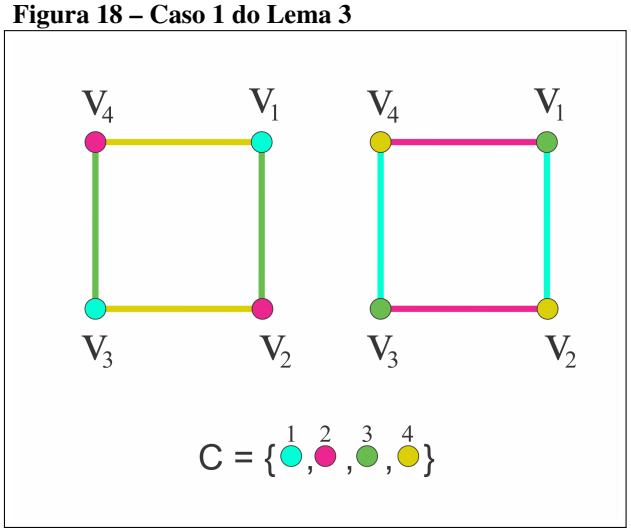
Caso 1: n é par. A coloração total é definida da seguinte forma.

$f(v_i) = 1$, se i é ímpar e $f(v_i) = 2$, se i é par.

$f(e = \{v_i, v_{i+1}\}) = 3$ se i é ímpar ou $f(e = \{v_i, v_{i+1}\}) = 4$, se i é par.

Pode-se verificar que esta é uma coloração total para C_n, mesmo se trocarmos as cores de um vértice v_i, $i \in \{1, 2, \ldots, n$, pela cor da sua aresta incidente $v_i v_{i+1}$, como mostra a Figura 18.

Figura 18 – Caso 1 do Lema 3



Fonte: autoria própria

Caso 2: n é impar. A coloração total é definida da seguinte forma.

$f(v_n) = 3$ e $f(v_i) = 1$ ou $f(v_i) = 2$, de acordo com a paridade de i, para $1 \leq i \leq n-1$.

$f(e = \{v_n, v_1\}) = 2$ e $f(e = \{v_i, v_{i+1}\}) = 3$ ou $f(e = \{v_i, v_{i+1}\}) = 4$, de acordo com a paridade de i, para $1 \leq i \leq n - 1$.

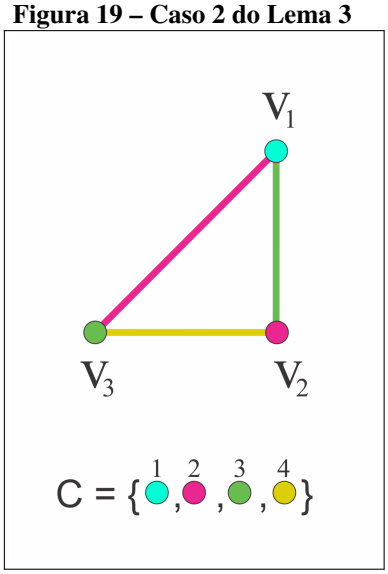
No *caso 2*, com o n ímpar, pode-se verificar que é possível colorir um ciclo ímpar com $n \leq 4$ cores como mostra a Figura 19, ou seja, os ciclos são 4-coloríveis. □

A partir dos resultados anteriores, é possível determinar o número cromático total dos ciclos.

Teorema 4. Seja C_n um ciclo, $n \geq 3$. Se n é múltiplo de 3, então $\chi''(C_n)$ = 3. Caso contrário, $\chi''(C_n) = 4$.

Demonstração. Considere um ciclo C_n. Se n é múltiplo de 3, então $\chi''(C_n) = 3$, pelo Lema 1. Se $n \not\equiv 0 \pmod 3$, então qualquer coloração total do ciclo C_n necessita de pelo menos 4 cores,

Figura 19 – Caso 2 do Lema 3



Fonte: autoria própria

como mostra o Lema 2. Então, nesse caso, $\chi''(C_n) \geq 4$. O Lema 3 prova que quando $n \not\equiv 0 \pmod 3$, existe uma coloração total para o grafo C_n com 4 cores, ou seja, $\chi''(C_n) \leq 4$. Portanto, quando $n \not\equiv 0 \pmod 3$, tem-se $\chi''(C_n) = 4$. □

3.3 BIPATIDOS G = (A.B,E(G)) COM NÚCLEO EM A

Nesse seção, o Problema da Coloração Total será resolvido para alguns grafos bipartidos a partir de uma extensão da solução de um outro problema de coloração, o Problema da Coloração de Arestas. O Problema da Coloração de Arestas tem solução computacional em tempo polinomial para os grafos bipartidos. Nessa seção, veremos que as arestas de um grafo bipartido podem ser coloridas com $\Delta(G)$ cores. Em alguns casos, é possível estender essa coloração para uma coloração total (atribuindo-se cores para os vértices de G) utilizando exatamente mais uma cor.

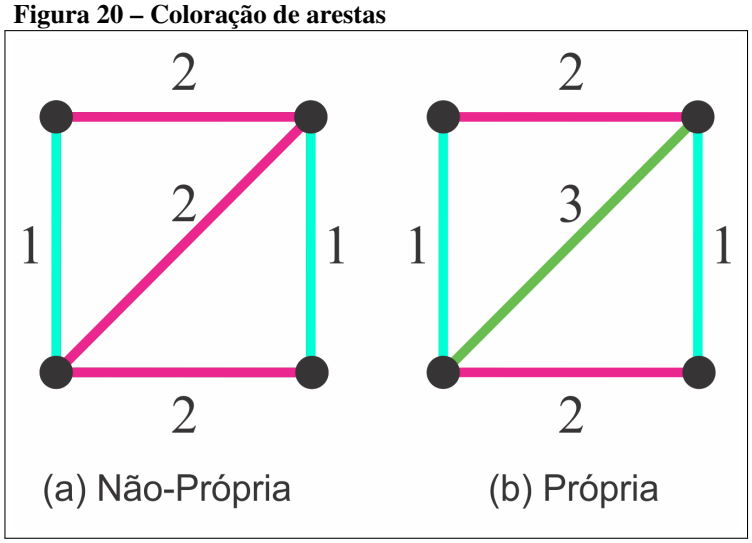
Uma coloração de arestas de um grafo G é uma função $f : E(G) \longrightarrow C$, onde C é um conjunto de cores geralmente representadas por números naturais. Uma coloração de arestas é própria quando $f(e_i) \neq f(e_j)$ para todo par de arestas adjacentes e_i e e_j. A Figura 20 apresenta dois exemplos de coloração de arestas. O exemplo da esquerda é uma coloração não própria e o exemplo da direita é uma coloração de arestas própria.

O Problema de Coloração de Arestas é determinar qual é o menor número de cores para uma coloração de arestas própria de um dado grafo G. Esse número de cores é chamado de índice cromático e denotado por $\chi'(G)$.

É fácil verificar que o índice cromático de qualquer grafo simples[1] G é pelo menos $\Delta(G)$, já que as cores das arestas incidentes em um vértice com grau máximo ($\Delta(G)$) devem

[1] Um grafo simples é um grafo sem laços, nem arestas múltiplas.

Figura 20 – Coloração de arestas

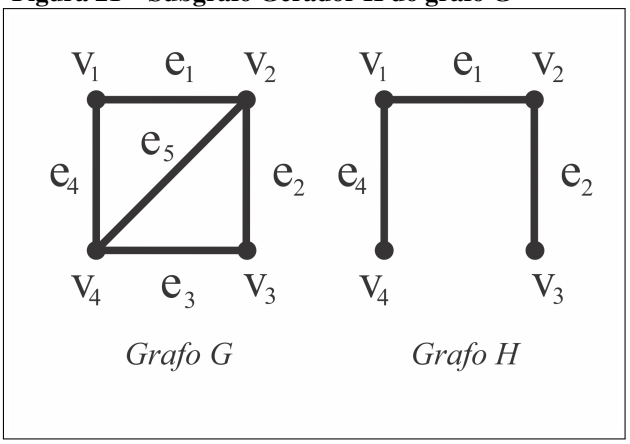


Fonte: Autoria Própria

ser todas diferentes. Em 1964, Vizing (1964) provou que todo grafo siples tem uma coloração de arestas com $\Delta(G) + 1$ cores. Portanto, todo grafo simples tem $\chi'(G) \in \{\Delta(G), \Delta(G) + 1\}$. Apesar de existirem apenas dois valores possíveis para o índice cromático de um grafo G, decidir se um grafo G tem um coloração de arestas com $\Delta(G)$ cores é NP-Completo (CAI; ELLIS, 1991). Apesar da dificuldade do Problema da Coloração de Arestas, existem algumas classes de grafos para as quais há algoritmo polinomial que exibe uma coloração de arestas ótima. Uma dessas classes é a dos grafos bipartidos (KONIG, 1916). Konig (1916) provou que todo grafo bipartido tem $\chi'(G) = \Delta(G)$. Esse resultado será apresentado mais adiante. Para sua compreensão, alguns conceitos básicos são necessários.

Um subgrafo gerador H de um grafo G é um subgrafo que é obtido a partir da remoção de arestas G mantendo o mesmo conjunto de vértices (BONDY; MURTY *et al.*, 1976, p.46-47). A Figura 21 mostra um exemplo de grafo G e um de seus subgrafos geradores, H.

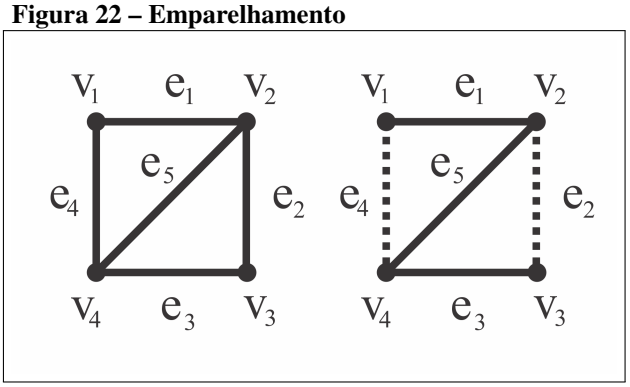
Figura 21 – Subgrafo Gerador H do grafo G



Fonte: autoria própria

Um emparelhamento M é um conjunto independente de arestas, ou seja, é conjunto de arestas duas a duas não adjacentes. A Figura 22 mostra um exemplo de emparelhamento em um grafo, representado pelas arestas pontilhadas.

Figura 22 – Emparelhamento



Fonte: autoria própria

Se um vértice incide em uma aresta de M, ele é coberto por M. Um emparelhamento é perfeito se cobre todos os vértices do grafo. Um grafo é emparelhável se ele possui um emparelhamento perfeito. Pela definição de emparelhamento perfeito, um grafo com um número ímpar de vértices não é emparelhável. Um emparelhamento máximo é um emparelhamento com o maior número possível de arestas do grafo. O número de arestas de um emparelhamento máximo de G é chamado de número de emparelhamento denotado por $\alpha'(G)$. Um emparelhamento maximal é em emparelhamento que não está propriamente contido em nenhum outro emparelhamento do grafo.

Dois grafos são disjuntos se não possuem vértices em comum, dois grafos são aresta-disjuntas se não possuem arestas em comum. A união de um grafo G com um grafo H, denotada por $G \cup H$, é o grafo com conjunto de vértices $V(G) \cup V(H)$ e conjunto de arestas $E(G) \cup E(H)$. Se G e H são disjuntos, então $G \cup H$ é uma união disjunta e é denotada por $G + H$. Um grafo G é desconexo se, e somente se, G é a união disjunta de dois grafos não vazios. Cada subgrafo conexo maximal de um grafo G é chamado de componente conexo de G ou simplesmente componentes (BONDY; MURTY *et al.*, 1976, p.29).

Observe que, dado um grafo G com uma coloração de arestas própria, o conjunto de arestas coloridas com uma mesma cor é um emparelhamento em G. Uma k-coloração-de-arestas é, portanto, uma partição do conjunto de arestas $E(G)$ em partes $M_1, M_2, \ldots, M_k$, onde cada M_i é o conjunto de arestas coloridas com a mesma cor i, $1 \leq i \leq k$.

Seja H um subgrafo gerador do grafo G e seja $C = \{M_1, M_2, \ldots, M_k\}$ uma k-coloração-de-arestas de H. Uma cor i incide no vértice v quando i é a cor de alguma aresta e que incide em v. Caso contrário, ela está disponível em v. Uma cor está disponível para uma aresta, se estiver disponível para os dois extremos da aresta. Portanto, se uma aresta $e \in E(G) \setminus E(H)$ não está colorida, qualquer cor disponível para e pode ser atribuída, estendendo C para uma k-coloração-de-arestas de $H + e$. Sejam i e j duas cores distintas quaisquer e H_{ij} o subgrafo de G com conjunto de arestas $E(H_{i,j}) = M_i \cup M_j$ e conjunto de vértices $V(H_{i,j})$ dado pelo conjunto de vértices de G cobertos por $E(H_{ij})$. Observe que, como M_i e M_j são emparelhamentos disjuntos, cada componente de H_{ij} é um ciclo par ou um caminho. Os componentes conexos de H_{ij} que são caminhos são chamados de componentes-caminhos de H_{ij} ou ij-caminhos.

Seja G um grafo simples e M um emparelhamento em G. Um caminho alternante em G é um caminho em G cujas arestas se alternam pertencendo a M e fora de M. Um M-caminho-de-aumento é um caminho alternante em G no qual nenhum de seus vértices extremos estão cobertos pelo emparelhamento M. Observe que uma outra forma de definir emparelhamento máximo é por caminhos de aumento: um emparelhamento M em um grafo G é um emparelhamento máximo se, e somente se, G não contém um M-caminho-de-aumento. O teorema a seguir prova que todo grafo bipartido tem uma coloração de arestas com $\Delta(G)$ cores. Este resultado foi dado por Konig (1916) e a demonstração apresentada aqui está no livro de Bondy, Murty *et al.* (1976).

Teorema 5. (BONDY; MURTY *et al.*, 1976, p.453) Se G é um grafo bipartido simples, então $\chi'(G) = \Delta(G)$.

Demonstração. Por indução, seja G um grafo bipartido e $e = u, v$ uma aresta de G. Vamos assumir que $H = G \backslash e$ tem uma $\Delta(G)$-coloração-de-arestas $\{M_1, M_2, \ldots, M_{\Delta(G)}\}$. Se alguma cor está disponível para e, esta cor pode ser atribuída para e e obtém-se uma $\Delta(G)$-coloração-de-arestas de G. Então, pode-se assumir que cada uma das $\Delta(G)$ cores incide em u ou v. Como o grau de u em $G \backslash e$ é no máximo $\Delta - 1$, pelo menos uma cor i está disponível para u e representa v. Igualmente, pelo menos uma cor j está disponível para v e representa u. Como u tem grau um no subgrafo H_{ij}, o componente que contém u em H_{ij} contém um ij-caminho com arestas de cor i e j, que será chamado de P. Esse caminho não termina em v, pois se terminasse, P seria de tamanho par, começando com uma aresta de cor i e terminando com uma aresta de cor j, e $P + e$ seria um ciclo de tamanho impar em G, contradizendo a hipótese de que G é um grafo bipartido. Então pode-se inverter as cores do caminho P, comece trocando a cor da aresta incidente em v de i para j e a cor da aresta seguinte do caminho deve ser trocada de j para i e assim sucessivamente. As cores serão alternadas até que se atinja o outro extremo desse caminho, que não é u, ou seja, P é um caminho alternante. Agora, a cor i está disponível tanto para u quanto para v. Assim, pode-se atribuir a cor i para e, obtendo-se uma $\Delta(G)$-coloração-de-arestas para G. □

O resultado apresentado no Teorema 5 sucita a sguinte questão: e se realizarmos uma coloração de arestas em um grafo bipartido e depois colorirmos os vértices um cores que estão disponíveis ou com cores novas? O primeiro resultado relacionado com essa questão é a prova de todo grafo bipartido G tem uma coloração total com $\Delta(G) + 2$ cores, ou seja, $chi''(G) \leq \Delta(G) + 2$.

Teorema 6. Se G é um grafo bipartido, então $\chi''(G) \leq \Delta(G) + 2$.

Demonstração. Seja G um grafo bipartido com partição de $V(G)$ em conjuntos independentes $[A, B]$. Primeiro, faça uma coloração de arestas própria em G com $\Delta(G)$ cores do conjunto $\{1, 2, 3, \ldots, \Delta(G)\}$. Tal coloração de arestas existe, pelo Teorema 5. Para que não haja conflito de cores, pinte os vértices de A com a cor $\Delta(G) + 1$ e pinte os vértices da parte B com a cor $\Delta(G) + 2$. Como A e B são conjuntos independentes, não há vértices adjacentes com a mesma cor. Como as cores $\Delta(G) + 1$ e $\Delta(G) + 2$ não foram atribuídas às arestas, não há aresta e vértice

que sejam adjacentes e tenham a mesma cor. Por fim, não há duas arestas adjacentes com a mesma cor, já que a coloração de arestas de G é própria. Portanto, G tem uma coloração total com $\Delta(G) + 2$ cores, ou seja, $\chi''(G) \leq \Delta(G) + 2$. □

Pelo Teorema 6, todo grafo bipartido G tem uma coloração total com $\Delta(G) + 2$ cores. Resta saber quais desses grafos tem $\chi''(G) = \Delta(G)+1$. Lembre-se que essa resposta implica na solução do Problema da Coloração Total para grafos bipartidos e implica na solução do Problema P vs NP. O restante desta seção destina-se a apresentar um caso em que a coloração de arestas ótima pode ser estendida para uma $(\Delta(G)+1)$-coloração total. Tudo depende de qual das partes do grafo bipartido contém os vértices com grau $\Delta(G)$.

O núcleo de um grafo G é o subconjunto de $V(G)$ formado apenas pelos $\Delta(G)$-vértices do grafo, os vértices de grau máximo. Dado um grafo bipartido G com partição $[A, B]$, existem três possibilidades:

1. O núcleo do grafo está contido apenas em A.
2. O núcleo do grafo está contido apenas em B.
3. O núcleo do grafo tem vértices tanto em A quanto em B.

Os dois primeiros casos são equivalentes. Então pode-se assumir, sem perda de generalidade, que nesses casos o núcleo está contido em A (se estiver contido em B, basta trocar o nome das partes). Para esse caso, pode-se provar que o grafo bipartido é tipo 1, como mostra o seguinte teorema.

Teorema 7. Se G é um grafo bipartido com partição $[A, B]$ com núcleo contido em A, então G é tipo 1.

Demonstração. Seja G um grafo bipartido com partição do conjunto de vértices em dois conjuntos independentes $[A, B]$ de forma que o núcleo de G está contido em A. Conforme o Teorema 5, existe uma coloração de arestas própria para G com $\Delta(G)$ cores. Seja $C = \{M_1, M_2, \ldots, M_{\Delta(G)}\}$ tal coloração de arestas, onde cada cor é representada por um emparelhamento M_i, $1 \leq i \leq \Delta(G)$. Resta colorir os vértices de G. Como nenhum vértice em B é $\Delta(G)$-vértice, todos os vértices em B tem grau no máximo $\Delta(G) - 1$ e, portanto, tem pelo menos uma cor disponível do conjunto $\{1, 2, \ldots, \Delta(G)\}$. Pinte cada vértice de B com uma cor que esteja disponível para este vértice. Portanto, não há conflito entre as cores de vértices em B, já que trata-se de um conjunto independente, nem entre cores dos vértices de B e de arestas adjacentes a estes, pois foram usadas cores disponíveis. Resta colorir os vértices de A. Pinte os vértices da parte A com uma cor nova. Uma vez que esta cor ainda não foi utilizada em nenhum elemento da parte B e em nenhuma das arestas, e como A é um conjunto independente, esta é uma coloração total própria do grafo G. Portanto, $\chi''(G) = \Delta(G) + 1$. □

Seja G um grafo bipartido com partição do conjuntos de vértices $V(G)$ em dois conjuntos independentes A e B. Quando o núcleo está contido em uma das partes da partição, o grafo é tipo 1 pelo Teorema 7. Esse resultado implica que o Problema da Coloração Total em grafos bipartidos está em aberto apenas para os casos em que há vértices com grau máximo tanto em A quanto em B. Vamos chamar a classe dos grafos bipartidos cujo núcleo tem vértices em A e em B de binúcleo-partido. No conjunto dos grafos binúcleo-partido, são conhecidos grafos tipo 1 e tipo 2, como é o caso dos ciclos e dos grafos bipartidos completos balanceados, que serão apresentados na próxima seção.

3.4 COLORAÇÃO TOTAL DE GRAFOS BIPARTIDOS COMPLETOS

Essa seção apresenta a solução do Problema da Coloração Total em grafos bipartidos completos. Lembre-se que um grafo bipartido completo $K_{m,n}$ é um grafo bipartido com partição do conjunto de vértices em dois conjuntos independentes $[A, B]$ tal que $|A| = m$, $|B| = n$ e cada vértice em A é adjacente a todo vértice em B. Note que cada vértice em A tem grau igual a $|B| = n$ e cada vértice em B tem grau igual a $|A| = m$. Então, quando $m \neq n$, o núcleo do grafo está todo em uma das partes (ou todo em A ou todo em B). Neste caso, pelo Teorema 7, o grafo $K_{m,n}$ é tipo 1, como enuncia o Corolário 1.

Corolário 1. Se o grafo bipartido completo $K_{m,n}$ tem $m \neq n$, então $\chi''(K_{m,n}) = \Delta(K_{m,n}) + 1$.

Seja $K_{m,n}$ um grafo bipartido completo com partição do conjunto de vértices em dois conjuntos independentes $[A, B]$. Se $|A| = m = |B| = n$, o grafo é chamado de equipartido ou balanceado. Considere um grafo bipartido completo balanceado $K_{n,n}$. Note que o núcleo do grafo $K_{n,n}$ está tanto em A quanto em B, ou seja, o grafo $K_{n,n}$ é binúcleo-partido. É fácil verificar que o grafo $K_{n,n}$ tem $|V(K_{n,n})| = 2n$ e $|E(K_{n,n})| = n^2$, uma vez que as partes têm o mesmo tamanho e cada vértice tem n arestas entre si e os vértices da parte vizinha.

Dado um grafo G, um conjunto independente total em G é um subconjunto de $V(G) \cup E(G)$ em que quaisquer dois elementos não são adjacentes. Um conjunto independente total máximo é o conjunto independente total em G com o maior número de elementos. O tamanho do conjunto independente total máximo em G é denotado por $\alpha_T(G)$. Em qualquer coloração total própria de um grafo G, cada cor pode ser usada para colorir no máximo o número de elementos de um conjunto independente total do grafo. O grafo $K_{n,n}$ tem conjunto independente total máximo igual a n, como mostra o Lema 4.

Lema 4. Se G é grafo um grafo bipartido completo balanceado, então $\alpha_T(G) = n$.

Demonstração. Seja G um grafo bipartido completo balanceado com partição do conjunto de vértices em dois conjuntos independentes $[A, B]$ tal que $|A| = |B| = n$. Então, G pode ser

denotado por $K_{n,n}$. Observe que como cada vértice de A é adjacente a todo vértice de B, não há conjunto independente total que possua vértices tanto da parte A quanto da parte B. Então, qualquer conjunto independente total tem no máximo n vértices, todos da mesma parte. Se todos os vértices de uma parte fazem parte do conjunto independente total, então este conjunto não terá arestas, já que cada aresta incide em um desses vértices. Nesse caso, o tamanho do conjunto independente total é n. Sem perda de generalidade, seja A a parte escolhida como conjunto independente total. Para cada vértice $v \in A$ que for retirado do conjunto independente total, pode-se inserir neste mesmo conjunto exatamente uma das arestas incidentes em v. Logo, a cardinalidade do conjunto independente total não se altera e é no máximo n. □

O próximo teorema mostra que o grafo $K_{n,n}$ não pode ser colorido com $\Delta(K_{n.n}) = n$ cores e que, portanto, os grafo bipartidos completos balanceados são tipo 2.

Teorema 8. Se G é um grafo bipartido completo balanceado, então $\chi''(G) = \Delta(G) + 2$.

Demonstração. Seja G um grafo bipartido completo balanceado com partição do conjunto de vértices em dois conjuntos independentes $[A, B]$ tal que $|A| = |B| = n$. Suponha, por contradição, que existe uma coloração total do grafo $K_{n,n}$ com $\Delta(K_n, n) = n + 1$ cores. Como cada cor pinta no máximo um número de elementos igual ao tamanho do conjunto independente total máximo, pode-se concluir que $n + 1$ cores pintam no máximo $(n + 1)n = n^2 + n$ elementos. Como o número de elementos de um grafo $K_{n,n}$ é $n^2 + 2n$, faltam cores para pintar o grafo inteiro, um absurdo.

Resta provar que existe uma coloração total do $K_{n,n}$ com $n + 2$ cores. Pelo Teorema 5, é possível colorir as arestas do $K_{n,n}$ com n cores. Utilize uma nova cor para colorir os vértices da parte A e uma outra nova cor para colorir os vértices da parte B. Então existe uma coloração total para o $K_{n,n}$ com $n+2$ cores. Como $\Delta(K_{n,n}) = n$, tem-se $\chi''(K_{n,n}) = n+2$ e $K_{n,n}$ é Tipo 2. □

O resultado do Teorema 8 é uma evidência de que existem grafos binúcleo-partidos que são tipo 2. O próximo capítulo apresenta casos de grafos binúcleo-partidos que que são tipo 1.

4 GRADES, GRADES PARCIAIS E CUBOS K-DIMENSIONAIS

Nesse capítulo são apresentadas técnicas mais elaboradas de solução em tempo polinomial para o Problema da Coloração Total em subconjuntos dos grafos bipartidos. Esse capítulo é baseado nos resultados do trabalho de Campos (2006) e considera as classes de grafos grades, grades parciais, quase-escadas e cubos k-dimensionais. Todas as classes abordadas nesse capítulo, com exceção das grades parciais, são um subconjunto dos grafos binúcleo-partidos, para os quais o Problema da Coloração Total permanece em aberto. Mesmo para as grades parciais o Problema da Coloração Total permanece em aberto, como será visto nesse capítulo.

4.1 GRADES E GRADES PARCIAIS

Nessa seção serão apresentadas as provas do número cromático total para os grafos grades e algumas grades parciais.

Para definir os grafos grades, é necessário o conceito de produto cartesiano. O produto cartesiano entre dois conjuntos $A = \{a_1, a_2, \ldots, a_n\}$ e $B = \{b_1, b_2, \ldots, b_m\}$ é o conjunto $\{(a_i, b_j) : 1 \leq i \leq n \text{ e } 1 \leq j \leq m\}$. Considere dois grafos caminhos, P_m e P_n, onde m e n são números naturais, $V(P_n) = \{v_1, , v_2, v_3, \ldots, v_n\}$ e $V(P_m) = \{u_1, u_2, \ldots, u_m\}$. O grafo resultante do produto cartesiano $P_m \times P_n$ é o grafo $G_{m\times n}$ com conjunto de vértices $V(G) = V(P_m) \times V(P_n)$ e conjunto de arestas $E(G) = \{(v_i, u_j)(v_k, u_l) : (v_i, u_j)(v_k, u_l) \in V(G) \text{ e } |i - k| + |j - l| = 1\}$. A Figura 23 mostra o grafo grade $G_{4\times 3}$.

Se o grafo grade G tem $\Delta(G) = 1$, então G é o grafo K_2, que como já dito tem $\chi''(K_2) = 3$, ou seja, é tipo 2. Se $\Delta(G) = 2$, o grafo grade é um ciclo com 4 vértices, ou seja, é o grafo C_4 e, pelo Teorema 4, $\chi''(C_4) = 4$, ou seja, é tipo 2. Se $\Delta(G) \in \{3, 4\}$, o grafo grade tem uma $(\Delta(G) + 1)$-coloração total, como mostra o Teorema 9.

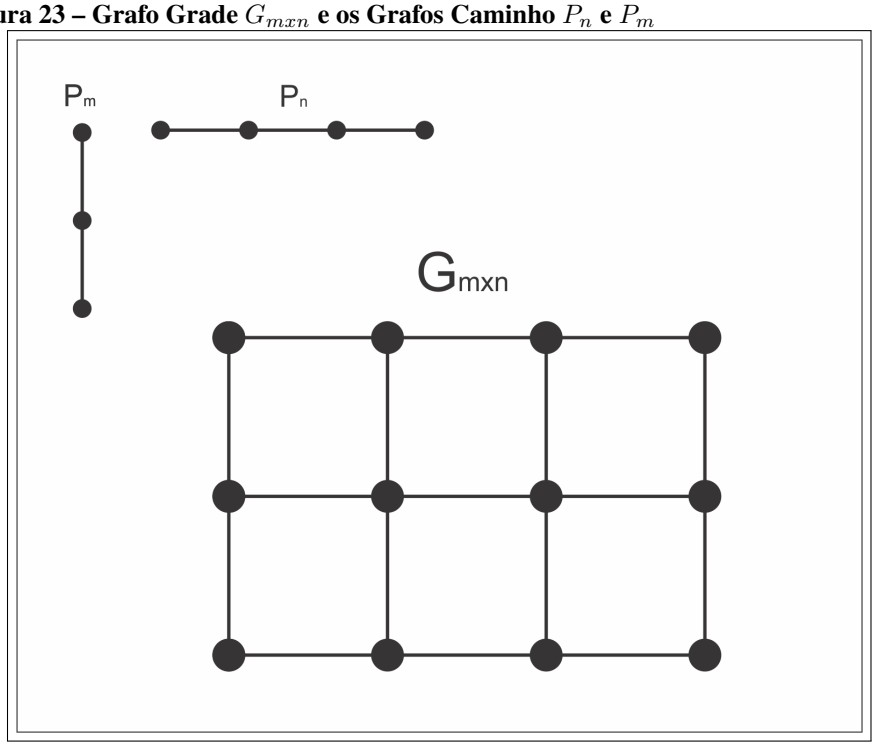
Teorema 9. (CAMPOS, 2006) O grafo grade $G = G_{m\times n}$ com $m \geq 1$ e $n \geq 2$ e diferente do C_4 é tipo 1.

Demonstração. Seja $G_{m\times n}$ um grafo grade com $m \geq 1$ e $n \geq 2$ tal que $G_{m\times n}$ não é o ciclo C_4. Para provar que $G_{m\times n}$ é tipo 1, é suficiente apresentar uma coloração total $\pi : V(G_{m\times n}) \cup E(G_{m\times n}) \to \{1, 2, \ldots, \Delta(G_{m\times n}) + 1\}$. A função π pode ser definida como segue.

$$\begin{cases} \pi((v_i, u_j)) := (2j + i - 2) \bmod 3; \\ \pi((v_i, u_j)(v_i, u_{j+1})) := (2j + i - 1) \bmod 3; \\ \pi((v_i, u_j)(v_{i+1}, u_j)) := 4-(i \bmod 2). \end{cases}$$

Para todo $j \in \{1, 2, \ldots, n\}$, seja P_m^j o subgrafo de $G_{m\times n}$ induzido pelo conjunto de vértices $V(P_m^j) = \{(v_i, u_j) : 1 \leq i \leq n\}$. Similarmente, para todo $i \in \{1, 2, \ldots, m\}$, seja P_n^i o

Figura 23 – Grafo Grade G_{mxn} e os Grafos Caminho P_n e P_m



Fonte: autoria própria

subgrafo de $G_{m\times n}$ induzido pelo conjunto de vértices $V(P_n^i) = \{(v_i, u_j) : 1 \leq j \leq m\}$.

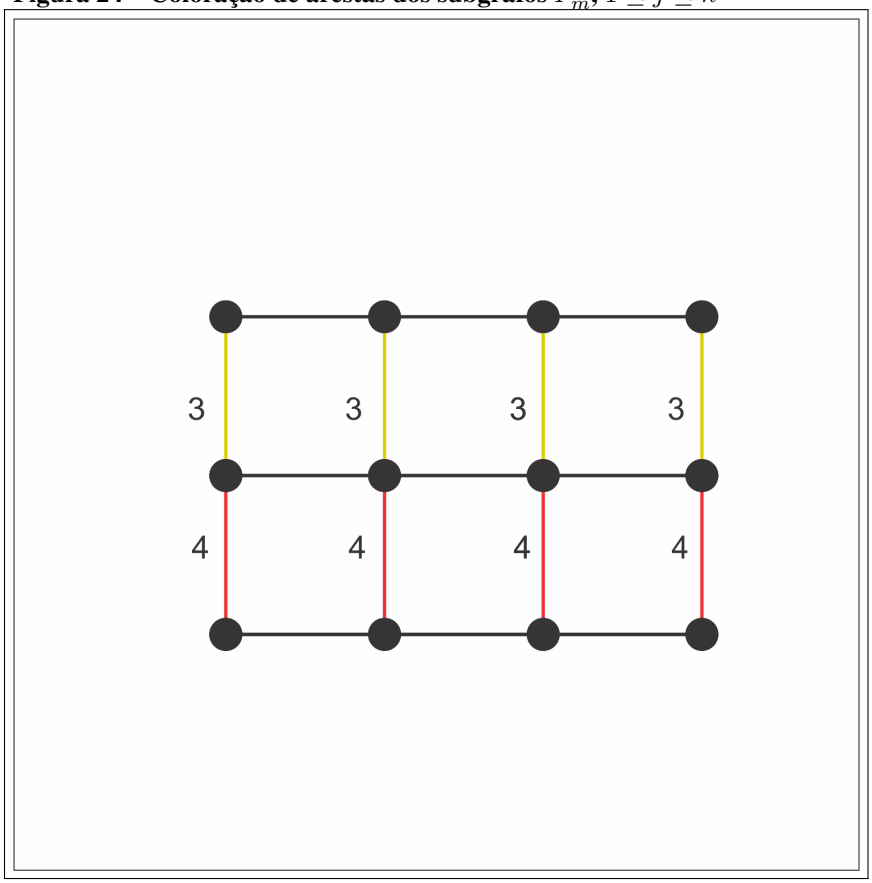
Por construção, as arestas dos subgrafos P_m^j, $1 \leq j \leq n$, vão possuir cores 3 ou 4, que não são atribuídas a nenhum vértice de $G_{m\times n}$ a nenhum aresta dos subgrafos P_n^i, $1 \leq i \leq m$. Além disso, arestas adjacentes nos caminhos Pm^j, $1 \leq j \leq n$ têm cores com diferentes paridades. Sendo assim os subgrafos P_m^j têm uma coloração de arestas própria, como mostra a Figura 24.

Ainda considerando os subgrafos P_m^j, quaisquer dois vértices adjacentes tem cores distintas. De fato, considere um vértice (v_i, u_j) que está colorido com $(2j + i - 2) \bmod 3$. Seus dois vizinhos, quando existem, são os vértices (v_{i-1}, u_j) e (v_{i+1}, j). Note que os quais (v_{i-1}, u_j) e (v_{i+1}, j) estão coloridos respectivamente com as cores $(2j+i) \mod 3$ e $(2j+i-1) \mod 3$, respectivamente. Portanto, as cores dos vizinhos diferem da cor do vértice (v_i, u_j) em uma unidade. Logo, a coloração do subgrafo P_m^j é uma coloração total própria.

Como o conjunto das cores dos vértices e arestas dos subgrafos P_n^i, $1 \leq i \leq m$, é disjunto do conjunto das cores das arestas dos grafos P_m^j, $1 \leq j \leq n$, resta provar que π é uma coloração total para os subgrafos P_n^i. Primeiro, observe que cada vértice (v_i, u_j) está colorido com a cor $(2j+i-2) \bmod 3$ e é adjacente aos vértices (v_i, u_{j-1}) e (v_i, u_{j+1}), os quais tem cores $(2j + i - 1) \bmod 3$ e $(2j + i) \bmod 3$. Então, as cores de vértices adjacentes diferem em pelo menos uma e no máximo duas unidades.

As arestas incidentes no vértice (v_i, u_j) no subgrafo P_n^i, $1 \leq i \leq m$, quando existem, são $(v_i, u_{j-1})(v_i, u_j)$ e $(v_i, u_j)(v_i, u_{j+1})$ e estão coloridas respectivamente com as cores $(2j + i) \bmod 3$ e $(2j+i-1) \bmod 3$. As cores dessas arestas diferem da cor de (v_i, u_j) em pelo menos uma e no máximo duas unidades. Por fim, duas arestas de um subgrafo P_n^i adjacentes têm cores

Figura 24 – Coloração de arestas dos subgrafos P_m^j, $1 \leq j \leq n$



Fonte: autoria própria

diferentes. De fato, considere uma aresta $(v_i, u_j)(v_i, u_{j+1})$. Por construção, essa aresta tem cor $(2j + i - 1) \bmod 3$ e é adjacente a duas arestas, $(v_i, u_{j-1})(v_i, j)$ e $(v_i, u_{j+1})(v_i, u_{j+2})$ cuja as cores são respectivamente $(2j + i) \bmod 3$, e $(2j + i + 1) \bmod 3$, respectivamente. Então, as cores de arestas adjacentes diferem em pelo menos uma e no máximo duas unidades da cor de $(v_i, u_j)(v_i, u_{j+1})$. A Figura 25 mostra a coloração total do grafo grade $G_{3\times4}$.

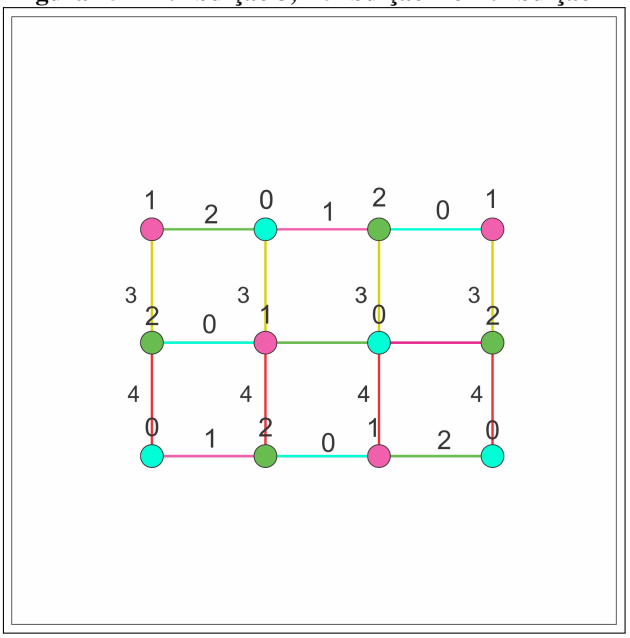
Observe que mesmo no caso em que $m = 2$ e $n > 2$ a coloração total é própria. Esses grafos tem grau máximo igual a 3 e vale a coloração total π já descrita. Como cada subgrafos P_m^j tem apenas uma aresta, a cor 4 não será utilizada. Então os grafos $G_{2,n}$ são tipo 1 e o resultado segue.

Portanto, quando $m \geq 2$ e $n > 2$, $G_{m,n}$ é tipo 1. □

Qualquer subgrafo de um grafo grade é uma grade parcial. Como as grades são grafos bipartidos e ser bipartido é propriedade hereditária, as grades parciais são um subconjunto dos grafos bipartidos. Além disso, existe interseção entre o conjunto das grades parciais e dos grafos binúcleo-partidos. Portanto, para as grades parciais, o Problema da Coloração Total permanece em aberto. Mas há subconjuntos para os quais se conhece o número cromático total. Vamos considerar grades parciais conexas, já que o número cromático total de uma grade parcial desconexa é o maior dos números cromáticos totais de suas componentes conexas.

Se a grade parcial G tem $\Delta(G) = 0$, então é o grafo trivial, tem $\chi''(G) = 1$ (uma cor

Figura 25 – Atribuição 3, Atribuição 1 e Atribuição 2



Fonte: autoria própria

para pintar o vértice) e é tipo 1. Se a grade parcial G tem $\Delta(G) = 2$, então G é um caminho ou ciclo par (já que é um grafo bipartido). Neste caso, o número cromático total de G é determinado pelos teoremas 2 e 4. Se a grade parcial G tem $\Delta(G) = 4$, então é suficiente adicionar a G as arestas faltantes para que seja um grafo grade, aplicar a coloração total π como descrita no Teorema 9 e em seguida remover as arestas que foram previamente adicionadas. Como $\Delta(G) = 4$ e π é uma 5-coloração total, G é tipo 1. Esse resultado segue, portanto, como corolário.

Corolário 2. (CAMPOS, 2006) Se G é uma grade parcial com $\Delta(G) = 4$, então G é tipo 1.

Resta considerar as grades parciais com $\Delta(G) = 3$. Este caso permanece em aberto. Campos (2006) apresentou dois subconjuntos desses grafos que são tipo 1.

Teorema 10. (CAMPOS, 2006) Seja G uma grade parcial conexa com grau máximo igual a 3. Se o tamanho do maior ciclo induzido de G é 4, então G é tipo 1

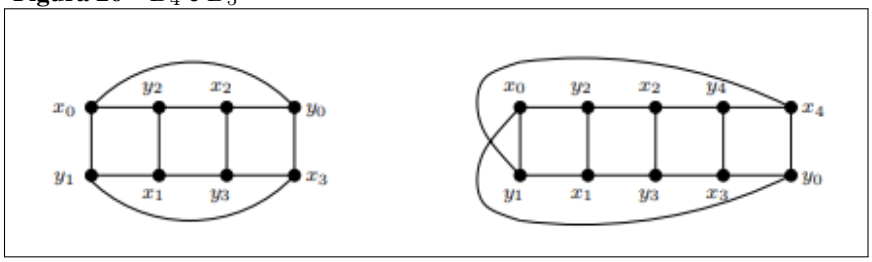
Teorema 11. (CAMPOS, 2006) Seja G uma grade parcial conexa com $\Delta(G) = 3$. Se G tem no máximo três vértices de grau 3, então G é tipo 1.

É interessante observar que grades parciais com $\Delta(G) = 3$ são grafos subcúbicos. Decidir se um grafo bipartido é tipo 1 é um problema NP-Completo mesmo quando restrito aos grafos cúbicos. O Teorema 10 garante que as grades parciais com $\Delta(G) = 3$ em que todo ciclo induzido tem tamanho no máximo 4 são tipo 1. Uma outra classe de grafos subcúbicos em que não se conhece grafos tipo 2 é a dos grafos cúbicos em que todo ciclo induzido tem tamanho maior que 4 (BRINKMANN; PREISSMANN; SASAKI, 2015).

4.2 GRAFOS QUASE ESCADAS

Quando o grafo $G_{m\times n}$ tem $m = 2$ é chamado de grafo escada. Campos (2006) define os grafos quase escadas como grafos bipartidos conexos cúbicos, com uma bipartição $[X_k, Y_k]$, tal que $X_k = \{x_0, \ldots, x_{k-1}\}, Y_k = \{y_0, \ldots, y_{k-1}\}$ e cada $x_i \in X_k$ tem $N(x_i) = \{y_i, y_{(i+1) \bmod k}, y_{(i+2) \bmod k}\}$, como mostra a Figura 26.

Figura 26 – B_4 **e** B_5



Fonte: (CAMPOS, 2006)

A definição de grafos quase escadas apresentada por Campos (2006) é importante para a apresentação da prova de que esses grafos são tipo 2 se e somente se k é ímpar. Considere a rotulação dos vértices de B_k como definida por Campos (2006) e apresentada na Figura 26. Os pares $x_i, y_{i+1}x_{i+1}$ e y_{i+1}, x_iy_{i+2} são chamados de pares equivalentes. Observe que pares equivalentes são conjuntos independentes totais. As arestas de um par equivalente são chamadas de par paralelo.

Teorema 12. Se B_k tem k par, então é tipo 1.

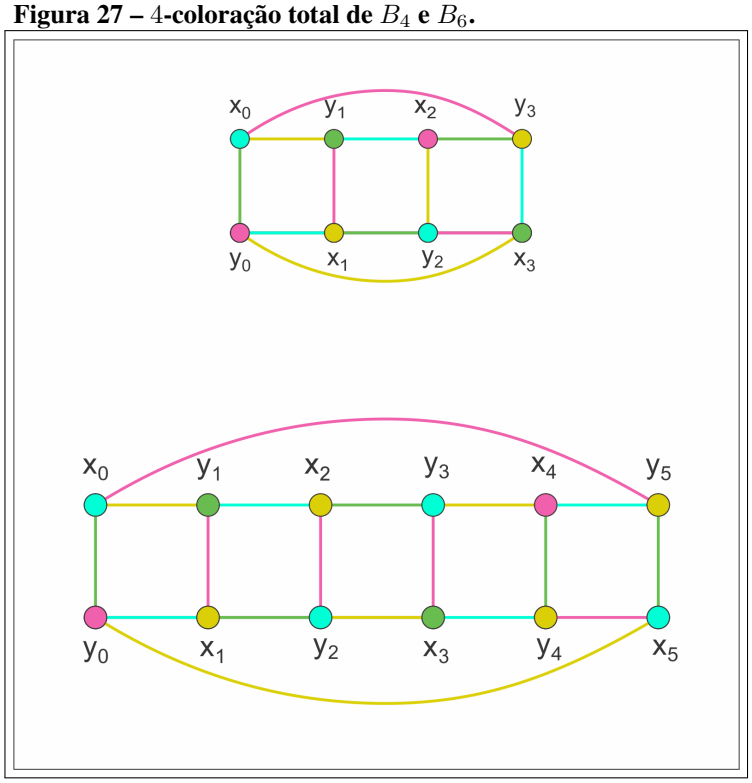
Demonstração. A prova é por indução. Para o caso base é necessário construir 4-colorações totais π_4 e π_6 para B_4 e B_6, respectivamente, como mostra a Figura 27.

Por hipótese de indução, existe um 4-coloração total para $B_{k-4}, k \geq 8$. Ajuste π_{k-4} para que x_0 seja uma âncora dos pares equivalentes x_0,y_1x_1 e y_1, x_0y_2, e para que $\pi_{k-4}(x_0) = \pi_4(x_0)$, $\pi_{k-4}(x_0y_0) = \pi_4(x_0y_0)$, $\pi_{k-4}(x_0y_1) = \pi_4(x_0y_1)$ e $\pi_{k-4}(x_0y_2) = \pi_4(x_0y_2)$. Note que, pelo Lema 5, esse ajustes implicam que $\pi_{k-4}(y_1) = \pi_4(y_1)$ e $\pi_{k-4}(y_1x_{k-5}) = \pi_4(y_1x_3)$.

O grafo B_k, k par e $k \geq 8$, pode ser obtido a partir de colagem de B_4 e B_{k-4} da seguinte forma: considere um grafo B_4 e um grafo B_{k-4}, remova as arestas que conectam x_0 e y_0 e y_1 com x_{k-5} em ambos os grafos. Adicione arestas entre x_0 de B_4 com y_0 de B_{k-4}, entre y_1 de B_4 e x_{k-5} de B_{k-4}, entre y_0 de B_4 e x_0 de B_{k-4} e entre x_3 e y_1 de B_{k-4}. Re-rotule os vértices do novo grafo adicionando 4 ao valor de cada índice dos vértices que pertenciam ao grafo $B_{k-4} - \{y_0\}$. Uma 4-coloração total, com uma atribuição de cor π para B_k pode ser construída a partir π_4 e π_{k-4} como descrito a seguir. Seja E_{out} o conjunto das arestas removidas e E_in o conjunto das arestas adicionadas.

1. Se e é um elemento de B_k correspondente a um elemento de $B_i \backslash E_{out}$, $i \in \{4, k-4\}$, então $\pi(e) := \pi_i(e)$;

Figura 27 – 4-coloração total de B_4 e B_6.



Fonte: Adaptação de (CAMPOS, 2006)

2. Pinte as arestas de E_{in} com as seguintes cores.

   - $\pi(x_0y_0) = \pi_4(x_0y_0)$;
   - $\pi(y_4x_4) = \pi_4(x_0y_0)$;
   - $\pi(x_3y_5) = \pi_4(y_1x_3)$;
   - e $\pi(y_1x_{k-1}) = \pi_4(y_1x_3)$.

Por construção de π, cada elemento de B_k recebe uma cor diferente. De fato, as colorações de $B_4 \setminus E_out$ e $B_{k-4} \setminus E_out$ são colorações totais parciais de B_k. As cores de seus elementos vieram de π_4 e π_{k-4}. Como π_4 é uma coloração total, $\pi(x_0) \neq \pi(y_4)$. Lembre-se que $y_4 \in V(B_k)$ corresponde ao vértice $y_0 \in V(B_4)$ e $\pi(y_1) \neq \pi(x_3)$. Analogamente, como π_{k-4} é uma coloração total, $\pi(x_4) \neq \pi(y_0)$ e $\pi(y_5) \neq \pi(x_{k-1})$. Por causa dos ajustes de cores feitos em π_{k-4}, $\pi(x_0) = \pi(x_4)$ e $\pi(y_1) = \pi(y_5)$. Conclui-se que $\pi(x_0) \neq \pi(y_0)$, $\pi(y_4) \neq \pi(x_4)$, $\pi(y_1) \neq \pi(x_{k-1})$ e $\pi(x_3) \neq \pi(y_5)$.

Para concluir a prova, é necessário analisar as aresta de E_{in}. Seja uv um aresta de E_{in}. Sem perda de generalidade, para a operação de colagem, u é uma vértice de B_4, v está em B_{k-4} e existem exatamente duas arestas em E_{out}, vw_1 e w_2v correspondendo a arestas de B_4 e B_{k-4} que não existem em B_k. Pelos ajustes feitos em π_{k-4} conclui-se que essas três arestas têm a mesma cor e o resultado segue. □

Os lemas 5 e 6 apesentam propriedades importantes de qualquer 4-coloração total de um grafo quase escada. Essas propriedades serão usadas para mostrar que um grafo quase escada

B_k é tipo 2 quando k é ímpar.

Lema 5. Seja $G := B_k$ e seja π uma 4-coloração total para um subgrafo $G_{2\times k}$ obtida através da remoção de exatamente um par paralelo de G. Então, para cada par equivalente não removido $x_i, y_{i+1}x_{i+1}$ e y_{i+1}, x_iy_{i+2} tem-se: (i) as arestas do par paralelo x_iy_{i+2} e $y_{i+1}x_{i+1}$ têm cores diferentes; (ii) ou $\pi(x_i) = \pi(y_{i+1}x_{i+1})$ ou $\pi(y_{i+1}) = \pi(x_iy_{i+2})$, mas não as duas ao mesmo tempo.

Demonstração. Para provar *(i)*, suponha que $\pi(x_i, y_{i+2}) = \pi(y_{i+1}x + i + 1)$. Note que os elementos y_{i+2}, x_{i+1} e $x_{i+1}y_{i+2}$ têm cores distintas e diferentes de $\pi(x_iy_{i+2})$. Além disto, $\pi(x_{i+1}) = \pi(y_{i+2}x_{i+2})$ e $\pi(y_{i+2}) = \pi(x_{i+1}y_{i+3})$. Portanto, os elementos x_{i+2}, y_{i+3} e $x_{i+2}y_{i+3}$ têm apenas duas cores atribuídas: $\pi(x_iy_{i+2})$ e $\pi(x_{i+1}y_{i+2})$, uma contradição. Portanto, $\pi(x_iy_{i+2}) \neq \pi(y_{i+1}x_{i+1})$.

Resta provar a propriedade *(ii)*. Note que $\pi(x_i) \neq \pi(y_{i+1})$, $\pi(x_i) \neq \pi(x_iy_{i+2})$ e $\pi(x_i) \neq \pi(y_{i+1}x_{i+1})$ pois são elementos adjacentes. Além disto, $\pi(x_iy_{i+2}) \neq \pi(y_{i+1}x_{i+1})$, como provado. Suponha, por contradição, que $\pi(x_i) \neq \pi(y_{i+1}x_{i+1})$ e $\pi(y_{i+1}) \neq \pi(x_iy_{i+2})$. Então, $\pi(x_i)$, $\pi(y_{i+1})$, $\pi(y_{i+1}x_{i+1})$ e $\pi(x_iy_{i+2})$ são cores duas a duas distintas. A aresta x_iy_{i+2} é incidente ou adjacente a todos esses quatro elementos e portanto, deve estar colorida com uma quinta cor, um absurdo. Portanto, ou $\pi(x_i) = \pi(y_{i+1}x_{i+1})$ ou $\pi(y_{i+1}) = \pi(x_iy_{i+2})$. Suponha que ambas as igualdades sejam válidas, ou seja, suponha que $\pi(x_i) = \pi(y_{i+1}x_{i+1})$ e que $\pi(y_{i+1}) = \pi(x_iy_{i+2})$. Então $\pi(x_{i+1})$, $\pi(y_{i+2})$ e $\pi(x_{i+1}y_{i+2})$ são cores diferentes de $\pi(x_i)$ e de $\pi(y_{i+1})$, um absurdo, já que foram usadas apenas quatro cores e x_{i+1}, y_{i+2} e $x_{i+1}y_{i+2}$ são adjacentes entre si. □

Suponha que um subgrafo de uma quase escada B_k esteja colorido por uma 4-coloração total. Considere os pares equivalentes x_i, $y_{i+1}x_{i+1}$ e y_{i+1}, x_iy_{i+2}. Se $\pi(x_i) = \pi(y_{i+1}x_{i+1})$, então para esses pares equivalentes x_i é chamado de âncora. Caso contrário, y_{i+1} é chamado de âncora.

Lema 6. Sejam $G := B_k$ e π uma 4-coloração total para um subgrafo G_{2xk} obtido a partir da remoção de exatamente um par paralelo de G. Se x_i é uma âncora, então y_i e y_{i+2} são âncoras para seus respectivos pares. Por outro lado, se y_{i+1} é âncora, então x_{i-1} e x_{i+1} são as âncoras de seus respectivos pares equivalentes.

Demonstração. Suponha que x_i é uma âncora; então $\pi(x_i) = \pi(y_{i+1}x_{i+1})$. Primeiro é preciso provar que y_{i+2} é uma âncora. Pelo Lema 5, ou $\pi(y_{y_{i+2}}) = \pi(x_{i+1}y_{i+3})$ ou $\pi(x_{i+1}) = \pi(y_{i+2}x_{i+2})$. Suponha que $\pi(x_{i+1}) = \pi(y_{y_{i+2}x_{i+2}})$. Como x_{i+1} é incidente a $y_{i+1}x_{i+1}$, tem-se $\pi(x_{i+1}) \neq \pi(y_{i+1}x_{i+1})$. Sendo assim, $\pi(x_iy_{i+2})$, $\pi(y_{i+2})$ e $\pi(x_{i+2}y_{i+2})$ são diferentes de $\pi(x_i)$ e de $\pi(x_{i+1})$. Logo, $\pi(x_iy_{i+2})$, $\pi(y_{i+2}$ e $\pi(x_{i+1}y_{i+2}))$ são apenas duas cores distintas, uma contradição pois esses elementos são todos adjacentes entre si. Suponha que y_i não é uma âncora, então x_{i-1} é uma âncora, pelo Lema 5. Então, y_{i+1} é uma âncora pelo argumento anterior, mas isso contradiz o Lema 5 já que x_i é uma âncora. Portanto, y_i é uma âncora. O caso em que y_{i+1} é uma âncora é válido pela τ-simetria. □

Considerando as propriedades dos lemas 5 e 6, o Teorema 13 prova que quando k é ímpar, B_k é tipo 2.

Teorema 13. Se o grafo B_k tem k ímpar, então $\chi''(B_k) = \Delta(B_k) + 2$.

Demonstração. Primeiro suponha por contradição, que B_k é tipo 1. Então, seja π uma 4-colaração total para B_k. Sem perda de generalidade, suponha que que $\pi(x_0) = \pi(y_1x_1)$. Aplicando-se a propriedade do Lema 6 sucessivamente, tem-se que todos os vértices x_i e y_i com i par são âncoras. Como k é ímpar, x_{k-1} e y_0 são âncoras. Logo, $\pi(x_{k-1}) = \pi(x_{k-1}y_1)$, contradizendo o Lema 5. Portanto, não há uma 4-coloração total para B_k, com k ímpar. Além disso Rosenfeld (1971) e Vijayaditya (1971) provaram que $\chi''(G) \leq 5$ para os grafos cúbicos. Sendo assim, $\chi''(B_k) = 5$. □

Note que quando k é ímpar, o grafo B_k tem um ciclo induzido pelos vértices $x_0y_2x_2y_4x_4 \ldots y_{k-1}x_{k-1}$ Então, quando k é ímpar e $k \geq 5$, B_k tem um ciclo induzido de tamanho maior que 4 e, pelo Teorema 13, B_k é tipo 2.

Os resultados dos teoremas e 12 e 13 implicam no seguinte corolário.

Corolário 3. O grafo B_k é tipo 1 se, e somente se, k é par.

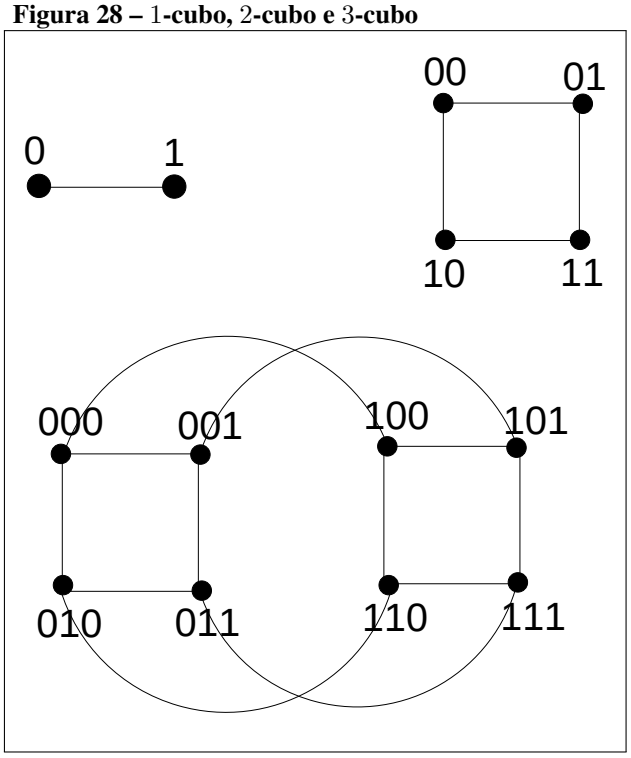
A próxima seção apresenta uma família de grafos bipartidos cúbicos que são tipo 1.

4.3 CUBOS K-DIMENSIONAIS

Um k-cubo, ou cubo k-dimensional é um grafo que pode ser definido recursivamente da seguinte forma: o grafo completo k_2 é um 1-cubo, rotule os vértices do 1-cubo com 0 e 1; para construir um k-cubo, $k \geq 2$, faça duas cópias de um $k-1$-cubo e rotule os vértices em cada cópia da mesma forma que no $k-1$-cubo; adicione uma aresta entre os vértices que têm o mesmo rótulo; em uma das cópias inclua um 0 mais a esquerda no rótulo de cada vértice e na outra cópia inclua um 1 mais a esquerda no rótulo de cada vértice. O grafo k-cubo é denotado por Q_k. Por construção, os vértices do grafo Q_k são representados por k-tuplas binárias ordenadas. O número de 1s nos rótulos de cada vértice é a paridade do vértice. Observe que vértices que têm a mesma paridade não são adjacentes entre si. Portanto, os k-cubos são bipartidos. A Figura 28 mostra exemplos de k-cubos rotulados como na construção recursiva descrita.

Os vértices que têm o mesmo rótulo nas cópias do Q_{k-1}, durante a construção do Q_k, são chamados de pares correspondentes. As arestas adicionadas entre esses vértices constituem um emparelhamento perfeito em Q_k. Os vértices que têm o mesmo rótulo nas cópias do Q_{k-1}, durante a construção do Q_k, são chamados de pares correspondentes. As arestas adicionadas entre esses vértices constituem um emparelhamento perfeito em Q_k.

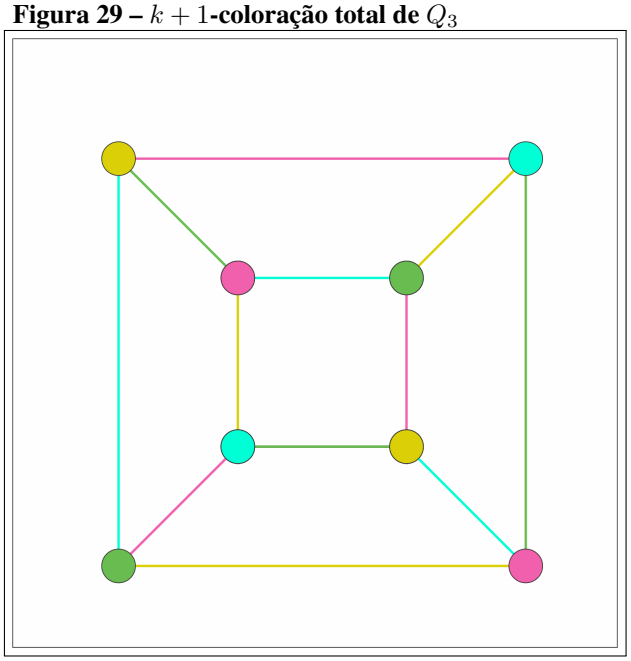
Figura 28 – 1-cubo, 2-cubo e 3-cubo



Fonte: autoria própria

Note que o grafo Q_1 é isomorfo ao K_2 e grafo Q_2 é isomorfo ao C_4. Estes grafos são tipo 2, como mostram os teoremas 2 e 4, respectivamente.

Figura 29 – $k+1$-coloração total de Q_3



Fonte: Adaptação de (CAMPOS, 2006)

Teorema 14. Para Q_k, $k \geq 3$ existe uma $(k+1)$-coloração total de Q_k tal que apenas quatro cores ocorrem no seu conjunto de vértices.

Demonstração. A Prova desta deste teorema será por indução. O caso base será uma 4-coloração total para o 3-cubo, como mostra a Figura 29.

Sejam Q_{k_1} e Q_{k_2} duas cópias do Q_{k-1} usadas para construir o grafo Q_k. Por hipótese de indução tanto Q_{k_1} quanto Q_{k_2} possuem uma coloração total com $\Delta(Q_{k_i}) + 1 = k$ cores, sendo $1 \leq i \leq 2$. Suponha uma k-coloração total dos grafos Q_{k_1} e Q_{k_2} de forma que os pares correspondentes não recebam a mesma cor. Tal coloração é possível, já que é suficiente uma rerotulação das cores de uma das cópias. Seja M o emparelhamento formado pelas arestas entre pares correspondentes no Q_k. Pinte as arestas de M com a cor $\Delta(Q_k) + 1 = k + 1$. □

5 CONCLUSÕES E TRABALHOS FUTUROS

O conjunto dos grafos bipartidos é um conjunto de grafos amplo. Como visto nos capítulos anteriores, o conjunto dos bipartidos compreende todos os grafos que não possuem ciclo ímpar, ou seja existe uma infinidade de conjuntos de grafos que estão inteiramente contidos no conjunto dos bipartidos (como por exemplo os caminhos) ou parcialmente contidos no conjunto dos bipartidos (como o conjunto dos ciclos).

O Problema da Coloração Total para um grafo bipartido regular G é um problema *NP*-Difícil, como demostrado por McDiarmid, Sánchez-Arroyo e Colin McDiarmid e Sánchez-Arroyo (1994), e permanece em aberto, para todo grafo k-regular com $k \geq 3$. O conjunto dos grafos bipartidos não-regulares contém todos os grafos bipartidos que possuem núcleo em apenas uma das partes e esses grafos são tipo 1, como demonstrado na Sessão 3.3. Para os Grafos com núcleo em ambas as partes, o Problema da Coloração Total permanece em aberto, em bora esteja resolvido em algumas subclasses, como caminhos, ciclos, algumas árvores, bipartidos completos, algumas grades parciais, grafos quase escadas e cubos k-dimensionais.

No caso dos grafos bipartidos completos, o Problema da Coloração Total está resolvido. Os grafos bipartidos completos são tipo 1 se, e somente se, não são balanceados.

O Problema da Coloração Total permanece em aberto para os grafos bipartidos regulares que possuum núcleo em ambas as partes, sendo eles balanceados ou não balanceados e para os grafos bipartidos não regulares com núcleo em ambas as partes, sendo eles balanceados ou não balanceados.

5.1 TRABALHOS FUTUROS

Nesse trabalho as subclasses de grafos bipartidos em que o Problema da Coloração Total está em aberto podem ter várias subclassificações. Limitar estruturalmente as classes para resolver o Problema da Coloração Total é uma estratégia de divisão e conquista que pode ser usada para provar que $P = NP$ (se for o caso). Para os bipartidos, algumas dessas subclasses já foram conquistadas, como é o caso das grades e grades parciais com $\Delta(G) \neq 3$, das grades, dos bipartidos completos, dos ciclos pares, das árvores. Uma das classes a ser conquista é a dos grafos bicúbicos ou bipartidos cúbicos. Como o grafo bipartido cúbico é regular, tem núcleo em ambas as partes. Nesse conjunto estão os grafos quase escada, uma subclasse já conquistada. O restante da classe pode ser dividida subclasses como, por exemplo, os grafos balanceados e não balanceados. Se for possível determinar quais grafos bicúbicos são tipo 1 e quais grafos são tipo 2, pode-se tentar estender a coloração total desta classe para supergrafos da mesma, como os bipartidos 4-regulares. Dessa maneira, pode-se tentar uma estratégia de prova por indução para todos os bipartidos regulares. Observe, por exemplo, que caso seja possível a coloração total dos

grafos bicúbicos com 4 cores, então todo grafo bicúbico ao qual se adicione um emparelhamento é passível de uma 5-coloração total, uma vez que o grau do grafo aumenta em 1 e uma cor adicional pode ser atribuída a esses elementos adicionados.

REFERÊNCIAS

BEHZAD, Mehdi. Graphs and their chromatic numbers, proquest llc. **Ann Arbor, MI**, 1965.

BEHZAD, M; CHARTRAND, G; JR, JK Cooper. The colour numbers of complete graphs. **Journal of the London Mathematical Society**, Wiley Online Library, v. 1, n. 1, p. 226–228, 1967.

BERMOND, Jean-Claude. Nombre chromatique total du graphe r-parti complet. **Journal of the London Mathematical Society**, Wiley Online Library, v. 2, n. 2, p. 279–285, 1974.

BONDY, John Adrian; MURTY, Uppaluri Siva Ramachandra *et al.* **Graph theory with applications**. [S.l.]: Citeseer, 1976. v. 290.

BRINKMANN, Gunnar; PREISSMANN, Myriam; SASAKI, Diana. Snarks with total chromatic number 5. **Discrete Mathematics and Theoretical Computer Science**, v. 17, n. 1, p. 369–382, 2015.

CAI, Leizhen; ELLIS, John A. Np-completeness of edge-colouring some restricted graphs. **Discrete Applied Mathematics**, Elsevier, v. 30, n. 1, p. 15–27, 1991.

CAMPOS, Christiane Neme. **O problema da coloração total em classes de grafos**. Tese (Doutorado) — Instituto de Computação - Unicamp, Campinas, Brasil, 4 2006.

CARLSON, James *et al.* **The millennium prize problems**. [S.l.]: American Mathematical Soc., 2006.

CHARTRAND, Gary; ZHANG, Ping. **Chromatic graph theory**. [S.l.]: CRC press, 2008.

CHEW, KH; YAP, HP. Total chromatic number of complete r-partite graphs. **Journal of graph theory**, Wiley Online Library, v. 16, n. 6, p. 629–634, 1992.

COOK, Stephen A. The complexity of theorem-proving procedures. In: ACM. **Proceedings of the third annual ACM symposium on Theory of computing**. [S.l.], 1971. p. 151–158.

COOK, Stephen A; RECKHOW, Robert A. Time bounded random access machines. **Journal of Computer and System Sciences**, Elsevier, v. 7, n. 4, p. 354–375, 1973.

CORMEN, Thomas H *et al.* **Introduction to algorithms**. [S.l.]: MIT press, 2009.

____. **Algoritmos: teoria e prática**. 3rd. ed. [S.l.]: Elsevier, 2012.

FEOFILOFF, Paulo. Departamento de ciência da computação instituto de matemática e estatística universidade de são paulo. 2013.

GAREY, Michael R; JOHNSON, David S. **Computers and intractability**. [S.l.]: wh freeman New York, 2002. v. 29.

HOFFMAN, Dean G; RODGER, Christopher A. The chromatic index of complete multipartite graphs. **Journal of Graph Theory**, Wiley Online Library, v. 16, n. 2, p. 159–163, 1992.

KONIG, Dénes. Graphok és alkalmazásuk a determinánsok és a halmazok elméletére. **Mathematikai és Természettudományi Ertesito**, v. 34, p. 104–119, 1916.

MACHADO, Raphael; FIGUEIREDO, Celina de. Complexity separating classes for edge-colouring and total-colouring. **Journal of the Brazilian Computer Society**, Springer, v. 17, n. 4, p. 281–285, 2011.

MCDIARMID, Colin J. H.; SÁNCHEZ-ARROYO, Abdón. Total colouring regular bipartite graphs is np-hard. **Discrete Mathematics**, Elsevier, v. 124, n. 1-3, p. 155–162, 1994.

PAPADIMITRIOU, Christos H. **Computational complexity**. [S.l.]: John Wiley and Sons Ltd., 2003.

PRESTES, Edson. **Complexidade de Algoritmos**. 2011. Disponível em: <http://www.inf.ufrgs.br/~prestes/Courses/Complexity/aula1.pdf>. Acesso em: 2018-12-16.

ROSENFELD, Moshe. On the total coloring of certain graphs. **Israel Journal of Mathematics**, Springer, v. 9, n. 3, p. 396–402, 1971.

SÁNCHEZ-ARROYO, Abdón. Determining the total colouring number is np-hard. **Discrete Mathematics**, North-Holland, v. 78, n. 3, p. 315–319, 1989.

SKIENA, Steven S. **The algorithm design manual: Text**. [S.l.]: Springer Science & Business Media, 1998. v. 1.

VIJAYADITYA, N. On total chromatic number of a graph. **Journal of the London Mathematical Society**, Wiley Online Library, v. 2, n. 3, p. 405–408, 1971.

VIZING, Vadim G. On an estimate of the chromatic class of a p-graph. **Diskret analiz**, v. 3, p. 25–30, 1964.

WILF, Herbert S. Algorithms and complexity. In: SUMMER. [S.l.], 1994.

YAP, Hian-Poh; JIAN-FANG, Wang; ZHONGFU, Zhang. Total chromatic number of graphs of high degree. **Journal of the Australian Mathematical Society**, Cambridge University Press, v. 47, n. 3, p. 445–452, 1989.