Coloração de arestas em grafos indiferença

Flávio de Freitas Stecca

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

Coloração de arestas em grafos indiferença

Flávio de Freitas Stecca[1]

Dezembro de 2003

Banca Examinadora:

- João Meidanis
  Instituto de Computação
  Universidade Estadual de Campinas (Orientador)
- Célia Picinin de Mello
  Instituto de Computação
  Universidade Estadual de Campinas
- Celina Miraglia Herrera de Figueiredo
  Instituto de Matemática
  Departamento de Ciência da Computação
  Universidade Federal do Rio de Janeiro
- Ricardo Dahab (suplente)
  Instituto de Computação
  Universidade Estadual de Campinas

[1]**Apoio Financeiro Parcial : Coordenadoria de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)**

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Stecca, Flávio de Freitas

St31c Coloração de arestas em grafos indiferença / Flávio de Freitas Stecca -- Campinas, [S.P. :s.n.], 2003.

Orientador : João Meidanis

Dissertação (mestrado) - Universidade Estadual de Campinas. Instituto de Computação.

1. Teoria da computação. 2. Teoria dos grafos. 3. Programação linear. I. Meidanis, João. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Coloração de arestas em grafos indiferença

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Flávio de Freitas Stecca e aprovada pela Banca Examinadora.

Campinas, 13 de janeiro de 2004.

João Meidanis
Instituto de Computação
Universidade Estadual de Campinas
(Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 12 de dezembro de 2003, pela Banca examinadora composta pelos Professores Doutores:

Profa. Dra. Celina Miraglia Herrera de Figueiredo
UFRJ

Profa. Dra. Célia Picinin de Mello
IC - UNICAMP

Prof. Dr. João Meidanis
IC - UNICAMP

Ao meu pai (in memoriam), com todo carinho.

Resumo

Esta dissertação aborda o problema da coloração de arestas restrito aos grafos indiferença. O teorema de *Vizing* diz que qualquer grafo pode ter suas arestas coloridas com $\Delta(G)$ ou $\Delta(G)+1$ cores. Grafos pertencentes à *Classe* 1 são os grafos cujo índice cromático (número mínimo de cores suficientes para pintar suas arestas) χ' é igual a $\Delta(G)$. Se $\chi' = \Delta(G)+1$, o grafo pertence à *Classe* 2. Um grafo é dito *overfull* se $\Delta(G)\lfloor\frac{n}{2}\rfloor < m$, onde n e m são o número de vértices e o número de arestas, respectivamente. Grafos *neighborhood overfull* são grafos que têm um vértice de grau máximo cuja vizinhança induz um subgrafo *overfull*. Grafos indiferença *overfull* ou *neighborhood overfull* pertencem à *Classe* 2.

Apresentaremos uma breve compilação de resultados de pesquisas. Dois destes resultados mostram que grafos indiferença com grau máximo ímpar e grafos indiferença reduzidos pertencem à *Classe* 1, porém o problema ainda está em aberto para um grafo indiferença qualquer.

Abordamos o problema criando um modelo de programação linear para coloração de arestas. Implementamos um gerador que nos permitiu gerar grafos indiferença de diferentes estruturas. Estes grafos tiveram suas arestas coloridas através de programação linear. Definimos um tipo especial de grafo indiferença denominado grafo indiferença semi-universal. Criamos um método que permite cobrir um grafo indiferença com grafos indiferença semi-universais. Mostramos que resolver o problema para um grafo indiferença qualquer equivale a estender certas colorações parciais para um grafo indiferença semi-universal qualquer. Reforçamos a conjectura de que todos os grafos indiferença não *neighborhood overfull* são *Classe* 1, através de testes práticos em milhares de grafos indiferença.

Abstract

This dissertation is on the subject of edge coloring restricted to indifference graphs. Vizing's theorem states that any graph can be edge-colored with Δ or $\Delta+1$ colors. Graphs are said to be *Class 1* if their chromatic index (minimum number of colors required to produce an edge-coloring) χ' equals $\Delta(G)$. If $\chi' = \Delta(G)+1$ the graph is said to be *Class 2*. A graph is *overfull* if $\Delta(G)\left\lfloor \frac{n}{2} \right\rfloor < m$, where n and m are the number of vertices and number of edges, respectively. Graphs are said to be *neighborhood overfull* if they have a maximum-degree vertex whose neighborhood induces an overfull subgraph. Overfull and neighborhood overfull indifference graphs are *Class 2*.

We will show a brief compilation of research results. Two of these results show that indifference graphs with odd maximum degree and reduced indifference graphs are *Class 1*, however the problem is open for a generic indifference graph.

The approach used for the problem was the creation of a linear programming model for edge coloring. A graph generator program that allowed creation of indifference graphs with different structures was implemented. These graphs were edge colored using linear programming. We defined a special type of graph called semi-universal indifference graph. We created a method for covering an indifference graph with semi-universal indifference graphs. We show that solving the problem for indifference graphs is equivalent to extending a partial edge coloring in a semi-universal indifference graph. We reinforce the conjecture that says that all indifference graphs not neighborhood overfull are *Class 1*, through practical tests in thousands of indifference graphs.

Agradecimentos

Fazer este agradecimento não é tão simples quanto parece, ao parar para pensar e me deparar com tantas pessoas que de uma forma ou de outra me ajudaram e foram muito importantes para mim, torna-se difícil expressar em poucas palavras todo meu agradecimento.

Gostaria de agradecer primeiramente a Deus, pelo dom da vida, pela oportunidade de estar aqui, por jamais ter me desamparado, e ter me permitido realizar este sonho.

Um agradecimento especial a minha mãe, que tanto fez por mim. Que muitas vezes abdicou de seus sonhos pelos meus, que lutou contra toda dificuldade priorizando meus estudos. Mãe, muito obrigado!

Ao professor e meu orientador João Meidanis, pela confiança, paciência, pela compreensão nos momentos difíceis, pelo incentivo, e pelos ensinamentos que tanto ajudaram na minha formação pessoal e profissional.

Aos professores da Universidade Católica de Goiás, Wellington, Thierson, Sibelius e Priscila, pelos ensinamentos, confiança e incentivo. Em especial aos professores Alexandre Ribeiro e Augusto Silva, pela especial amizade, companheirismo e confiança em mim depositada.

Aos meus grandes amigos Cristiano, Hiata e Stela, pelos momentos agradáveis que passamos juntos, pela amizade verdadeira, preocupação, incentivo e ajuda nos momentos mais difíceis, pelas críticas, sugestões, discussões, e pela revisão desta dissertação.

Em especial ao Cristiano, por ter me aguentado durante todo este tempo em nossa república, sendo um verdadeiro irmão para mim.

À todos da célula, Ricardo, Judy, Filipe, Cleison, Bete, Eduardo, Thaís, Karrie por terem tornado minha vida em Campinas mais agradável, pelas orações, e em especial por

me mostrarem o caminho da verdade e da vida, Jesus!

À todos meus amigos e amigas de Goiânia, pela amizade, preocupação, e momentos de alegria.

À Coordenadoria de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro a esta pesquisa.

À Unicamp e ao Instituto de Computação por terem me acolhido e me proporcionado uma infra-estrutura de estudo e pesquisa de primeira grandeza.

A todos vocês, o meu mais sincero obrigado.

F.F.S.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Coloração de arestas é um problema bastante pesquisado em *Teoria dos Grafos*. Este problema pode ser definido como descobrir o número mínimo de cores suficientes para colorir todas as arestas de um grafo, de forma que duas arestas adjacentes possuam cores distintas. Coloração de arestas é tipicamente utilizada em aplicações de agendamento onde se deseja otimizar a alocação de eventos, recursos ou tarefas.

O problema da coloração de arestas surgiu a partir do famoso **problema das quatro cores**, enunciado por *Francis Guthrie*, em 1852. *Guthrie* questionava se qualquer mapa poderia ter suas regiões pintadas usando no máximo quatro cores, onde regiões vizinhas tenham cores distintas. Esta questão foi respondida afirmativamente apenas 124 anos mais tarde pelos pesquisadores *K. Appel* e *W. Haken*.

O primeiro artigo que tratava de coloração de arestas foi publicado por *Tait* em 1880. Neste artigo, o autor prova que o problema das quatro cores é equivalente ao problema da coloração de arestas de um grafo plano cúbico sem aresta de corte utilizando 3 cores.

Em 1916, *D. König* exibiu um teorema que prova que todo grafo bipartido de grau máximo Δ pode ter suas arestas coloridas com Δ cores.

Em 1964 surgiu um dos mais importantes resultados sobre coloração de arestas. *V. G. Vizing* mostrou que o número mínimo de cores suficientes para colorir as arestas de um grafo G, índice cromático, $\chi'(G)$, é igual a seu grau máximo, $\Delta(G)$, ou igual a seu grau máximo acrescido de um, $\Delta(G) + 1$.

O Teorema de Vizing permitiu classificar os grafos em duas classes. Um grafo G é dito pertencer à *Classe* 1 se $\chi'(G) = \Delta(G)$, e à *Classe* 2 se $\chi'(G) = \Delta(G) + 1$. Decidir a qual classe um grafo pertence é denominado *problema da classificação.*

Uma forma de resolver o problema da classificação para grafos que suspeita-se estarem na *Classe* 1, é exibir uma coloração usando Δ cores para estes grafos. Já para grafos que suspeitamos estarem na *Classe* 2 não é suficiente exibir uma $(\Delta + 1)$-coloração, uma vez que os grafos que pertencem à *Classe* 1 também são $(\Delta + 1)$-coloríveis. Portanto, para se afirmar que um dado grafo pertence à *Classe* 2 é necessário encontrar algum argumento que demonstre que não pertence à *Classe* 1.

Decidir em qual das duas classes um grafo está, parece ser um problema trivial. Entretanto, em 1981, *I. Holyer* [23] provou que, apesar de ser aparentemente simples, determinar a classe de um grafo qualquer é um problema NP-Completo. Neste artigo, *I. Holyer* exibe uma redução polinomial do problema 3-SAT ao problema de encontrar o índice cromático de um grafo 3-regular. Por este fato, vários pesquisadores de todo o mundo dirigem suas pesquisas para o problema da coloração de arestas restrito a uma determinada classe de grafos.

Nesta dissertação estaremos estudando o problema da coloração de arestas restrito a classe dos *grafos indiferença.* Um grafo G é dito ser *indiferença* caso seja um *grafo de intervalos* e os intervalos possam ser tomados unitários. *Grafos de intervalos* são os grafos que admitem um bijeção do seu conjunto de vértices com um conjunto de intervalos da reta real, tal que exista uma aresta entre dois vértices se e somente se os intervalos correspondentes têm intersecção.

A classe dos grafos de intervalos é uma das classes de grafos mais utilizada para modelar problemas do mundo real. Existem várias aplicações que podem ser modeladas através de grafos de intervalos, como por exemplo, análise de cadeias de DNA, seriação arqueológica, temporização de semáforos, entre outras.

1.1 A dissertação e seus objetivos

O objetivo inicial desta pesquisa era encontrar um algoritmo polinomial que, dado como entrada um grafo indiferença qualquer, determine a quantidade de cores necessárias para pintar o grafo.

Abordamos o problema partindo da criação de um modelo linear que, através de programação matemática, resolve o problema para um grafo indiferença qualquer, e a partir destes resultados podemos analisar o comportamento do problema. Compreendido o problema, podemos tentar encontrar uma solução algorítmica ou uma prova de NP-completude para o problema.

Infelizmente, nesta dissertação, não chegamos nem ao algoritmo que pinta um grafo indiferença qualquer nem à prova de NP-completude. Apresentaremos como novas abordagens para o problema, um método para particionar grafos indiferença e a utilização de programação matemática, que poderão ser úteis em pesquisas futuras. Reforçaremos a conjectura que diz que todo grafo indiferença não *neighborhood overfull* pertence à *Classe* 1.

1.2 Organização do texto

A dissertação está organizada da seguinte forma:

O capítulo 2 descreve os conceitos básicos de coloração de arestas em grafos e os limites e restrições conhecidos para coloração. Apresenta os resultados conhecidos para algumas classes de grafos.

O capítulo 3 descreve a classe dos grafos indiferença, as propriedades e caracterizações destes grafos. Existem vários trabalhos sobre coloração de arestas em grafos indiferença, sendo alguns destes apresentados neste capítulo.

O capítulo 4 apresenta o modelo linear que criamos para coloração de arestas. Esta dissertação é baseada em extensivos testes de coloração de grafos e, neste capítulo, é apresentada a forma como são gerados grafos indiferença, como transformar estes grafos para um modelo linear e como resolvê-los.

O capítulo 5 descreve os métodos que utilizamos para colorir grafos indiferença, um

algoritmo para particionar este grafos e um método de como pintar estes grafos por pedaços.

O capítulo 6 apresenta algumas conclusões e sugestões para trabalhos futuros.

Capítulo 2

Coloração de arestas em grafos

Neste capítulo serão comentados os conceitos básicos sobre coloração de arestas, limites e restrições para a coloração. Serão apresentadas algumas classes de grafos que já tiveram o problema da classificação resolvido. Iniciaremos o capítulo com algumas definições básicas em teoria dos grafos, que serão utilizadas ao longo do texto.

2.1 Definições em teoria de grafos

Definição 2.1 *Um* **grafo** G *é definido por um conjunto* $V(G)$ *de pontos, chamados de* **vértices** *de* G *e por um conjunto* $E(G)$ *de pares não ordenados destes pontos, chamados de* **arestas** *de* G*, que conectam estes vértices. Dois vértices conectados por uma aresta são denominados* **adjacentes**. *Denotaremos* $|V(G)|$ *por* n *e* $|E(G)|$ *por* m*, ao longo de todo o texto.*

Definição 2.2 *Podemos definir grafos onde as arestas são pares ordenados de vértices distintos. Denominado grafo* **orientado** *ou* **direcionado**, *e denotado por* $\vec{G}$.

Definição 2.3 *Um* **subgrafo** *de* G *é um grafo* H *com* $V(H) \subseteq V(G)$ *e* $E(H) \subseteq E(G)$. *Para* $X \subseteq V(G)$, *denotamos por* $G[X]$ *o* **subgrafo induzido** *por* X, *onde* $V(G[X]) = X$ *e* $\{x, y\} \in E(G[X])$ *se e somente se* $x, y \in X$ *e* $\{x, y\} \in E(G)$. *Para* $Y \subseteq E(G)$, *o* **subgrafo induzido** *por* Y, $G[Y]$, *é o subgrafo de* G *onde* $E(G[Y]) = Y$ *e* $x \in V(G[Y])$

se e somente se $\{x, y\} \in Y$, *para algum* $y \in V(G)$. *A notação* $G \backslash Y$ *denota o sugbrafo de* G *com* $V(G \backslash Y) = V(G)$ *e* $E(G \backslash Y) = E(G) \backslash Y$.

Definição 2.4 *Um* **caminho** *em um grafo é uma seqüencia de vértices em que cada dois vértices consecutivos são ligados por uma aresta. Considere a seqüência* $(v_0, v_1, \ldots, v_{k-1}, v_k)$ *de vértices como sendo um caminho, então, para todo* i, *o par* $\{v_{i-1}, v_i\}$ *é uma aresta.*

Definição 2.5 *Um grafo* G *é dito* **conexo** *se existe um caminho entre qualquer par de vértices distintos do grafo.*

Definição 2.6 *A cardinalidade* n *de* $V(G)$, *também é chamada de* **ordem** *de* G.

Definição 2.7 *Uma* **coloração** *é uma função* $f : E(G) \rightarrow C$, *onde* C *é um conjunto finito de cores.*

Definição 2.8 *Dizemos que uma coloração com as cores de um conjunto* C *é* **válida** *para as arestas de* G *se arestas incidentes a um mesmo vértice possuem cores diferentes. Se* $|C| = p$, *então temos uma* p-***coloração*** *para* G, *neste caso, também dizemos que* G *é* p-*colorível. Se* $|C|$ *é o menor possível para colorir de forma válida as arestas de* G, $|C|$ *é o* **índice cromático** *de* G, *denotado por* $\chi'(G)$.

A figura 2.1 mostra o mesmo grafo colorido de duas formas; a segunda revela uma coloração ótima, conseqüentemente, seu índice cromático.

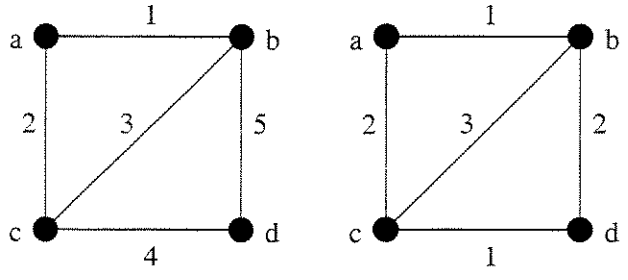

Figura 2.1: Um grafo e duas possíveis colorações, onde $C_1 = \{1, 2, 3, 4, 5\}$ e $C_2 = \{1, 2, 3\}$.

Definição 2.9 *Definimos por* **vizinhança aberta** *de $v \in V(G)$ em G, denotada por $N_G(v)$, ao conjunto de vértices em G que são adjacentes a v. A* **vizinhança fechada** *de v no grafo G, $N_G[v]$, é igual a $N_G(v) \cup \{v\}$. Com isso, temos que o* **grau** *$gr_G(v)$ do vértice v em G é igual a cardinalidade de sua vizinhança aberta em G e que $\Delta(G) = \max_{v \in V(G)} gr_G(v)$. Dois vértices u e v de um grafo G são ditos* ***gêmeos*** *se $N_G[u] = N_G[v]$.*

Definição 2.10 *Chamamos u de* **Δ-vértice** *de G se $u \in V(G)$ e $gr_G(u) = \Delta(G)$.*

Definição 2.11 *Denominamos* ***clique*** *de G um subgrafo completo de G. Uma clique é* ***maximal*** *se não está propriamente contida em nenhuma outra clique de G.*

Definição 2.12 *Dizemos que um conjunto $E \subseteq E(G)$ é um* **emparelhamento** *se as arestas de E são duas a duas não adjacentes (não incidem num mesmo vértice).*

2.2 Teorema de Vizing

Em 1964, *Vizing* provou que qualquer grafo G pode ter suas arestas pintadas com $\Delta(G)$ ou $\Delta(G) + 1$ cores. A seguir apresentaremos a prova do teorema de Vizing encontrada no artigo de *Fiorini e Wilson* [16].

Teorema 2.1 (Teorema de Vizing) *Seja G um grafo de grau máximo Δ. Então:*

$$\Delta(G) \leq \chi' \leq \Delta(G) + 1$$

Prova:

Se G é um grafo colorido com as cores $\alpha, \beta, ...$, então denotamos $H(\alpha, \beta)$ o subgrafo de G induzido pelas arestas cuja cor é α ou β. Observe que cada componente de $H(\alpha, \beta)$ deve ser um caminho ou um ciclo de comprimento par.

A primeira desigualdade é trivial. Para provar que $\chi' \leq \Delta(G) + 1$ o autor utilizou indução no número de arestas de G. A idéia é provar que se todas as arestas de G, menos uma delas, têm sido coloridas com $\Delta(G) + 1$ cores, então existe uma coloração para todas as arestas de G com $\Delta(G) + 1$ cores.

Suponha que cada aresta de G tenha sido colorida com uma das $\Delta(G)+1$ cores dadas, com a única exceção da aresta $e_1 = vw_1$. Então deve existir pelo menos uma cor não representada no vértice v e pelo menos uma cor não representada no vértice w_1. Se alguma cor não está representada nem no vértice v nem em w_1, então esta cor pode ser utilizada para colorir a aresta e_1, o que completaria a prova. Caso contrário, seja α uma das cores não representada em v e seja $\beta_1 \neq \alpha$ uma das cores não representada em w_1. A prova continua nos três passos seguintes :

PASSO 1:

Seja $e_2 = vw_2$ a aresta incidente no vértice v que tem atribuída a cor β_1. Esta aresta deve existir pois, caso contrário, a cor β_1 não estaria representada nem em v nem em w_1, o que levaria a uma contradição com a hipótese inicial. Em seguida apaga-se a cor da aresta e_2 e associa-se a cor β_1 a aresta e_1. Veja figura 2.2.

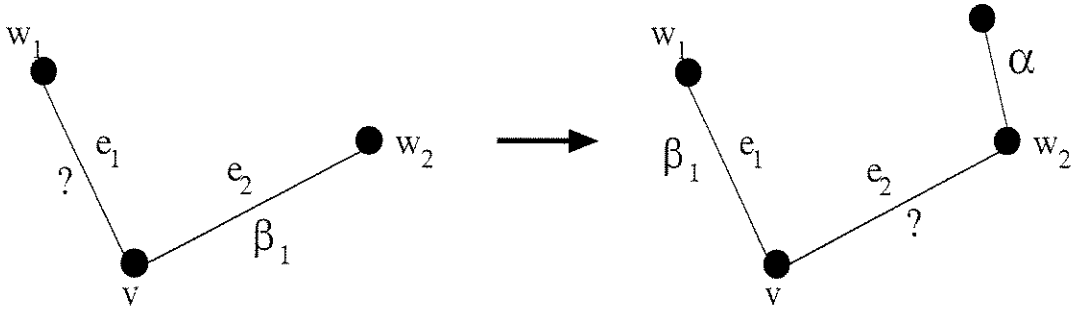

Figura 2.2: Passo 1a - Teorema de Vizing

Pode-se assumir que os vértices v, w_1 e w_2 pertencem a mesma componente do subgrafo $H(\alpha, \beta_1)$ pois, caso contrário, poderia-se trocar as cores das arestas na componente que contém w_2 sem alterar a cor da aresta e_1. Isto implica que a aresta e_2 pode ser colorida com α, completando desta forma a coloração de G. Assim, a situação é mostrada na figura 2.3.

PASSO 2:

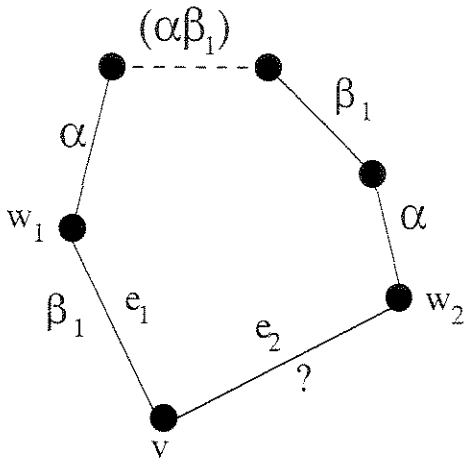

Figura 2.3: Passo 1b - Teorema de Vizing

Seja $\beta_2 \neq \beta_1$ uma das cores não representadas em w_2. Pode-se assumir que a cor β_2 está representada em v pois, caso contrário, poderia-se completar a coloração de G colorindo e_2 com a cor β_2. Portanto, seja $e_3 = vw_3$ a aresta incidente em v que possui a cor β_2. Então apaga-se a cor da aresta e_3 e associa-se a cor β_2 à aresta e_2. Pelo mesmo argumento do passo 1, podemos supor que os vértices v, w_2 e w_3 pertencem à mesma componente de $H(\alpha, \beta_2)$.

A figura 2.4 mostra a situação.

PASSO 3:

Repetindo o procedimento anterior, obteremos eventualmente um vértice w_k adjacente ao vértice v, tal que a aresta vw_k não está colorida e alguma cor β_i com $(i < k-1)$ não está representada em w_k. Como antes, podemos assumir que os vértices v, w_i e w_{i+1} pertencem a mesma componente T de $H(\alpha, \beta_i)$. Dado que, α não está representada em v, e β_i não está representada em w_{i+1}, T deve ser o caminho desde v até w_{i+1} que passa pelo vértice w_i e que é formado só de arestas coloridas alternadamente com β_i e α (figura 2.5). Este caminho não contém w_k, pois β_i não está representado em w_k.

Se $T^\star$ é a componente de $H(\alpha, \beta_i)$ que contém o vértice w_k, então T e $T^\star$ devem ser disjuntas. Portanto, é possível trocar as cores de $T^\star$ e colorir a aresta vw_k com a cor α.

□

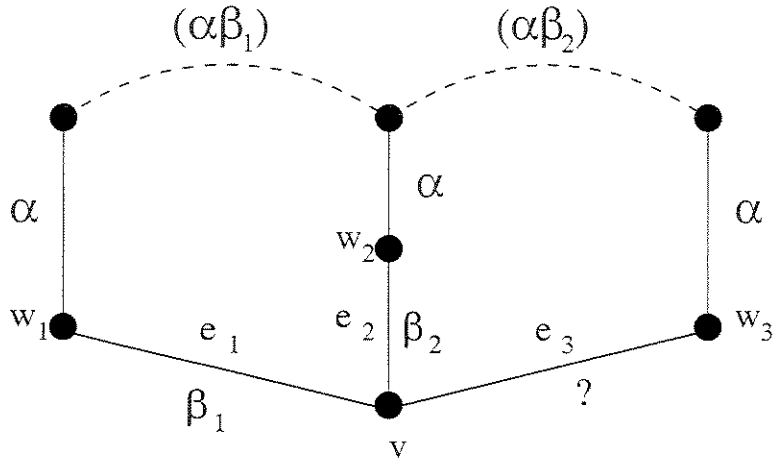

Figura 2.4: Passo 2 - Teorema de Vizing

2.3 Vizing e o problema da classificação

O Teorema de *Vizing* criou uma forma simples de classificar os grafos em duas classes.

$$\begin{cases} \textit{Classe } 1\ (C1), & \textit{onde } \chi' = \Delta \\ \textit{Classe } 2\ (C2), & \textit{onde } \chi' = \Delta + 1. \end{cases}$$

Com apenas duas opções parece ser simples decidir a classe a qual pertence determinado grafo. Entretanto, *Holyer* provou que o problema de decidir a classe a qual pertence um grafo arbitrário é NP-Completo [23], então tendemos a acreditar que não existe nenhum algoritmo polinomial que resolva este problema.

Em virtude do problema da coloração de arestas ser NP-Completo para um grafo qualquer, pesquisadores têm direcionado suas pesquisas para classes particulares de grafos. Conforme veremos na seção §2.5, para várias classes de grafos existe solução polinomial para o problema da classificação, enquanto para outras classes está provado ser NP-Completo.

Conforme observado por *Fiorini* e *Wilson* [16], aparentemente o número de grafos que pertencem a *Classe* 1 é bem superior ao número de grafos que pertencem a *Classe* 2. Examinando todos os 143 grafos conexos formados por no máximo 6 vértices, *Fiorini* e

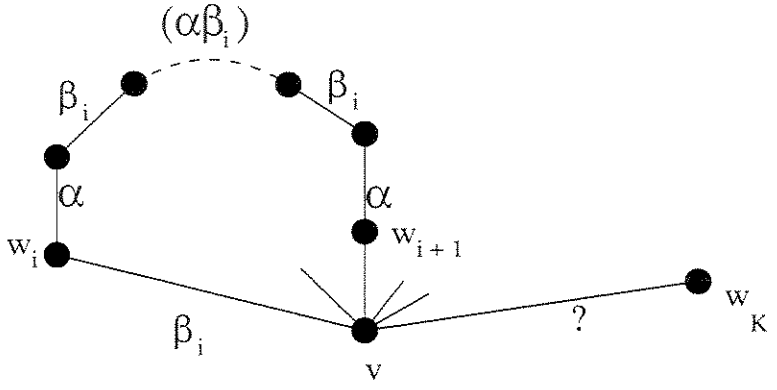

Figura 2.5: Passo 3 - Teorema de Vizing

Wilson encontraram somente 8 grafos que são *Classe* 2. Estes grafos estão ilustrados na figura 2.6.

2.4 Restrições à coloração

Intuitivamente podemos pensar que quanto mais arestas o grafo tiver maior será a probabilidade de ser *Classe* 2. Existe um limite para a quantidade de arestas que podem ser pintadas com Δ cores.

Por definição, cada cor corresponde a um emparelhamento e pode cobrir no máximo $\lfloor \frac{n}{2} \rfloor$ arestas, onde n é o número de vértices. Conseqüentemente, se o número de arestas é maior do que o produto do grau máximo por $\lfloor \frac{n}{2} \rfloor$, então o grafo necessariamente pertence à *Classe* 2. Grafos com esta característica são denominados de grafos *overfull.*

Grafos overfull são todos aqueles grafos em que o número de vértices n é ímpar e a seguinte regra é válida: $\Delta \frac{(n-1)}{2} < m$, onde Δ é o grau máximo e m é o número de arestas. Todo grafo overfull pertence à *Classe* 2. Grafos com o número de vértices par nunca são overfull, pois o produto $\frac{n}{2}\Delta$ será sempre maior ou igual ao número de arestas.

Um grafo G é dito ser *subgrafo-overfull* quando existe um subgrafo overfull H de G tal que $\Delta(H) = \Delta(G)$. Um subgrafo é sempre formado com algumas arestas e vértices do grafo original, neste caso o subgrafo não precisa ser induzido. Em grafos subgrafo-overfull também são necessárias $\Delta(G) + 1$ cores para colorir suas arestas.

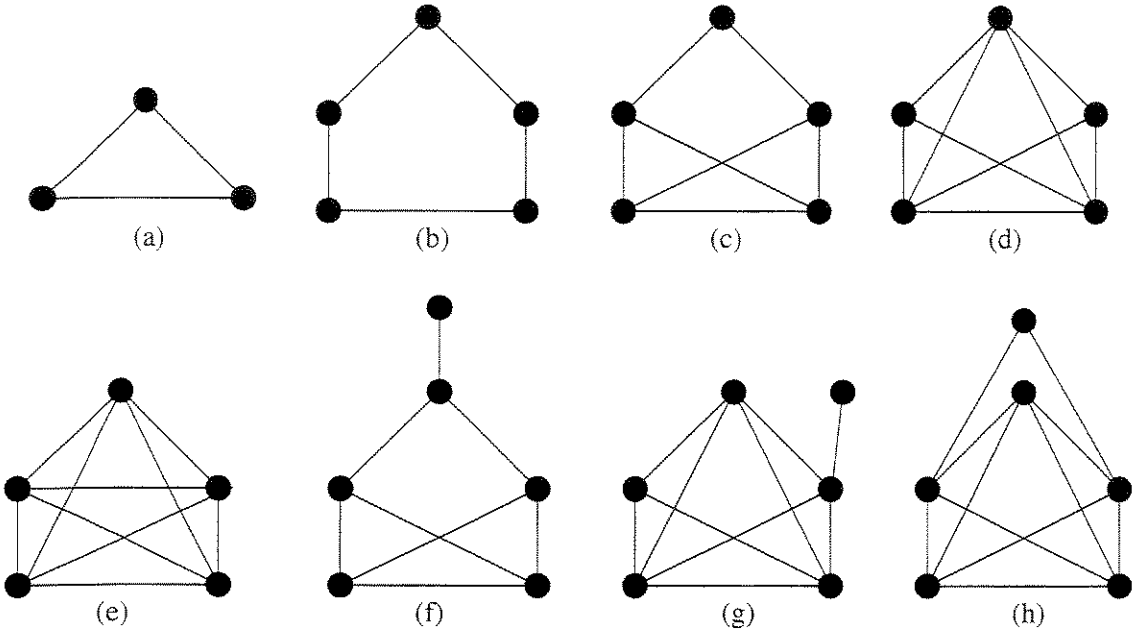

Figura 2.6: Grafos *Classe* 2 com no máximo 6 vértices.

Um grafo é *neighborhood overfull* quando possui um vértice de grau máximo, do qual a vizinhança induz um subgrafo overfull. Grafos neighborhood overfull são *Classe* 2 já que possuem um subgrafo overfull H com $\Delta(H) = \Delta(G)$.

Todo grafo neighborhood overfull (NO) é também subgrafo-overfull (SO). Assim como todo grafo overfull (O) é também subgrafo-overfull. Já grafos overfull e neighborhood overfull são incomparáveis. (O $\subset$ SO $\subset$ C2 e NO $\subset$ SO $\subset$ C2) [11].

É importante ressaltar que as restrições apresentadas nesta seção implicam que o grafo necessariamente pertence à *Classe* 2, mas a volta não é verdadeira. Um grafo pode pertencer à *Classe* 2 e não respeitar nenhuma das restrições apresentadas. A figura 2.7 ilustra esta relação. Já na figura 2.8 é apresentado o grafo de *Petersen* que não é subgrafo-overfull e mesmo assim pertence à *Classe* 2.

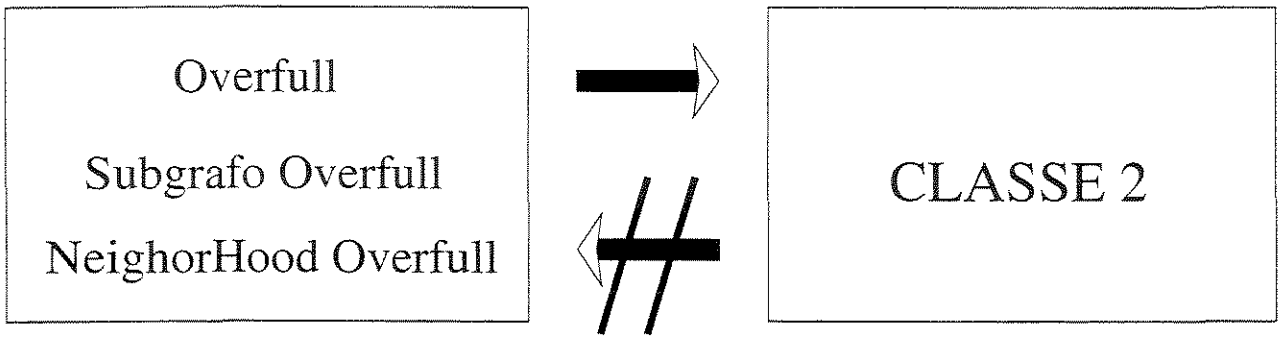

Figura 2.7: Relação entre as restrições e a *Classe* 2

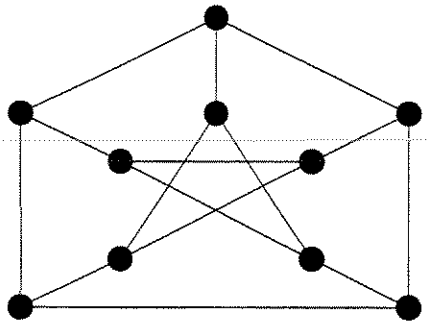

Figura 2.8: Grafo de Petersen

2.4.1 A conjectura de grafos overfull

Em 1986, *Hilton* e *Chetwynd* enunciaram uma importante conjectura a respeito da relação entre subgrafo-overfull (*SO*) e *Classe* 2. Neste trabalho os autores questionam se SO e *Classe* 2 são equivalentes quando o grafo possui Δ superior a $\frac{n}{3}$.

Conjectura 1 (Hilton e Chetwynd) *Se um grafo G com n vértices possui grau máximo maior que* $\frac{n}{3}$*, então G pertence à Classe* 1 *se e somente se G não possui nenhum subgrafo overfull.*

O grafo mostrado na figura 2.8 é um exemplo de grafo que pertence à *Classe* 2 e não é *SO*. Este grafo não pertence ao conjunto de grafos que satisfaz a Conjectura 1, pois seu Δ é igual a $\frac{n}{3}$. Isto também mostra que a restrição desta conjectura é a melhor possível.

Vários pesquisadores têm trabalhado com esta conjectura, conseguindo muitos resultados. Podemos citar o resultado mostrado por *M. Plantholt* [31] que provou ser verdadeira esta conjectura para grafos que possuem $\Delta(G) = n-1$. Os próprios autores da conjectura, *A. G. Chetwynd* e *A. J. W. Hilton* [4], provaram que a conjectura é verdadeira para grafos com $\Delta(G) \geq n-3$. Além destes, *D. G. Hoffman* e *C. A. Rodger* [22], demonstraram que os multipartidos completos também satisfazem a conjectura.

2.5 Resultados conhecidos

Conforme já mencionado, pesquisadores têm estudado o problema da coloração de arestas restrito a classes de grafos. Nesta seção estaremos apresentando os resultados conhecidos para algumas classes.

2.5.1 Casos polinomiais

Nas classes de grafos apresentadas nesta seção é possível determinar $\chi'(G)$ em tempo polinomial.

Grafos caminhos

O grafo formado por um caminho induzido por n vértices é chamado de *grafo caminho* e denotado por P_n.

Teorema 2.2 *Seja G um grafo caminho P_n. Então : $\chi' = 2$*

Prova:

O maior grau de P_n é 2, portanto $\chi' \geq 2$. Para obter uma coloração com duas cores é suficiente alternar essas duas cores nas arestas ao longo do caminho. □

Grafos ciclos

O grafo formado por um ciclo induzido por n vértices é chamado de *grafo ciclo* e denotado por C_n.

Teorema 2.3 *Seja G um grafo ciclo, C_n com n vértices. Então:*

$$\chi'(C_n) = \begin{cases} 2, & \text{se } n \text{ é par (Classe 1)} \\ 3, & \text{se } n \text{ é ímpar (Classe 2).} \end{cases}$$

Prova:

O maior grau do C_n é 2, portanto $\chi' \geq 2$. Se n é par, então duas cores são suficientes, pois basta alternar estas duas cores nas arestas ao longo do ciclo.

Caso n seja ímpar, é necessária uma terceira cor para evitar que a primeira aresta colorida e a última tenham a mesma cor. □

Bipartidos

Teorema 2.4 (Teorema de König) *Todo grafo bipartido é Classe 1.*

Prova:

Considere $e \in E$ como sendo uma aresta qualquer de grafo bipartido G, e $G - e$ o grafo obtido de G retirando a aresta e. É suficiente provar que se o grafo $G - e$ é colorido com $\Delta(G)$ cores então G também pode ser colorido com Δ cores. Suponha que a aresta e ligue os vértices w, v e que $G - e$ pode ser colorido com Δ cores. Considere C como sendo o conjunto das $\Delta(G)$ cores. Como o vértice v tem no máximo $\Delta(G) - 1$ arestas, v tem pelo menos uma cor faltante $\alpha \in C$. Da mesma forma, o vértice w possui pelo menos uma cor faltante $\beta \in C$. Se $\alpha = \beta$ então a aresta e pode ser pintada com a cor α, e a prova estaria completa. Caso contrário, considere $H(\alpha, \beta)$ como sendo o subgrafo definido pelas seguintes propriedades: ser conexo, conter w e conter todos as arestas e vértices de G que podem ser atingidas a partir de w por um caminho consistindo somente de arestas com as cores α e β. Como G é bipartido, $H(\alpha, \beta)$ não pode conter o vértice v. Logo, pode-se trocar as cores α e β em $H(\alpha, \beta)$ sem afetar v ou o resto da coloração. Agora, a aresta vw pode ser colorida com a cor α para obter a coloração desejada para G com Δ cores.

□

Grafos completos

Teorema 2.5 *Um grafo completo, K_n, pertence à Classe 1, se n é par, e pertence à Classe 2, se n é ímpar.*

Prova:

Caso n seja ímpar, temos que todo K_n com n ímpar é overfull, logo pertence a *Classe* 2. Apresentaremos uma função de coloração para o K_n com n cores [14] . Esta função fornece um algoritmo para coloração com $\Delta + 1$ cores mais eficiente do que o demonstrado por *Vizing*.

Considere a função $c : E(G) \rightarrow \{0, 1, \ldots, \Delta(K_n)\}$ tal que $c(i, j) = (i+j)\ mod\ (\Delta(K_n)+1)$ é uma coloração para as arestas de K_n com $\Delta + 1$ cores. Para provar que a função c é uma coloração válida, os autores mostram que na coloração gerada não há conflitos. Considere duas arestas distintas de K_n, ij e ik. Suponha que estas arestas gerem um conflito, ou seja, tenham a mesma cor, com $c(ij) = c(ik)$. Então, $(i + j)\ mod\ (\Delta(K_n) + 1) \equiv (i + k)\ mod\ (\Delta(K_n) + 1)$, logo $j = k$, contradição.

A figura 2.9 ilustra uma coloração dada pela função c para o K_5.

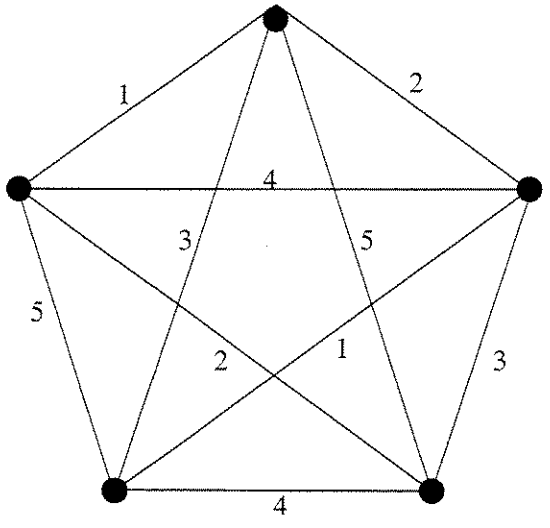

Figura 2.9: Coloração do K_5

Caso n seja par, escolhe-se um vértice qualquer, v de K_n, e o subgrafo induzido $K_n - v$, que é um grafo completo com $n-1$ vértices é colorido da forma descrita acima. Em cada vértice há exatamente uma cor faltando e as cores faltantes são todas distintas. As arestas de K_n incidentes em v são coloridas utilizando estas cores faltantes.

A figura 2.10 apresenta a coloração para o K_6.

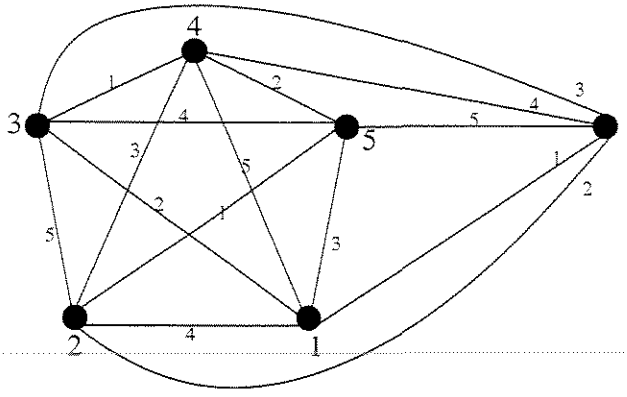

Figura 2.10: Coloração do K_6

□

2.5.2 Casos NP-Completos

Grafos regulares

Em 1981, *Holyer* provou que o problema da classificação para um grafo 3-regular é NP-Completo. Este resultado implica que o problema é NP-Completo para um grafo qualquer.

Para provar a NP-completude do problema, o autor utilizou a técnica de redução polinomial do problema 3-SAT, definido a seguir, ao problema de decidir o índice cromático de um grafo 3-regular.

Considere um conjunto de variáveis booleanas $u_1, ..., u_m$ e os seus complementos $\overline{u_1}, ..., \overline{u_m}$. Isto é, cada variável u_i toma um valor verdadeiro ou falso e u_i é verdadeiro se e somente se $\overline{u_i}$ for falso. Considere, também, uma conjunto de cláusulas $C = C_1, C_2, ..., C_n$ nas

variáveis $u_1, ..., u_m$, cada cláusula C_i consistindo de três literais l_{i1}, l_{i2} e l_{i3} onde cada literal l_{ij} é uma variável u_k ou o seu complemento $\overline{u_k}$.

Uma cláusula é dita satisfatível se existe uma atribuição de valores verdadeiro ou falso para as variáveis de tal modo que pelo menos um dos literais seja verdadeiro. O problema de 3-SAT consiste em determinar se existe uma atribuição de valores verdadeiro às variáveis tal que toda cláusula em C seja satisfatível.

Teorema 2.6 (Holyer) *O problema de determinar se o índice cromático de um grafo 3-regular é 3 ou 4 é NP-Completo.*

A idéia da prova é a seguinte. Dada uma instância C do problema 3-SAT, constrói-se um grafo 3-regular G que é 3-colorível se e somente se C é satisfatível.

O grafo G é construído através da união de "partes" ou componentes, cada uma delas realizando uma tarefa específica. Pares de arestas levam informação entre as diversas componentes. Em uma 3-coloração de G, estes pares de arestas representam o valor verdadeiro se as arestas possuem a mesma cor, e representam o valor falso se as arestas possuem cores distintas.

O autor define três tipos de componentes: uma associada a cada variável u_i, uma para cada cláusula C_j e outra para os literais que são $\overline{u_i}$. Ele mostra como construir um grafo 3-regular a partir delas em tempo polinomial.

Neste mesmo trabalho, *Holyer* conjecturou que o problema também é NP-completo para grafos regulares de grau máximo Δ. Em 1983, *Leven e Galil* provaram esta conjectura, demonstrando o seguinte teorema.

Teorema 2.7 (Leven e Galil [25]) *Dado um inteiro $k \geq 3$, o problema da classificação para um grafo k-regular é NP-completo.*

Grafos perfeitos

Cai e Ellis [2] mostraram a NP-Completude do problema da classificação para grafos regulares de comparabilidade e, portanto, para os *grafos perfeitos.*

Analogamente ao estudo da coloração de arestas, também existem estudos sobre coloração de vértices. Dizemos que uma coloração dos vértices de G é uma coloração válida

se vértices adjacentes possuem cores distintas. O **número cromático**, $\chi(G)$, de um grafo G é o menor número de cores com o qual é possível gerar uma coloração de vértices válida de G.

Um grafo é perfeito quando para todo subgrafo induzido o número cromático é igual ao tamanho da maior clique.

Um grafo $\overrightarrow{G}$ é dito ser de *comparabilidade* se para cada par de arestas (u, v) e (v, w) de G, orientadas de u para v e de v para w, existir a aresta (u, v) orientada de u para w.

Com este resultado, os autores mostraram que o problema da classificação continua NP-completo mesmo quando restrito a classes de grafos. Eles demonstraram o seguinte teorema.

Teorema 2.8 (Cai e Ellis) *Dado $k \geq 3$ inteiro, determinar o índice cromático de um grafo k-regular de comparabilidade é NP-Completo.*

Corolário 2.9 *Dado inteiro $k \geq 3$, determinar o índice cromático de um grafo k-regular perfeito é NP-Completo.*

Capítulo 3

Grafos indiferença

Neste capítulo estaremos comentando a respeito da classe dos grafos indiferença, apresentando suas definições e caracterizações. Apresentaremos alguns detalhes inerentes desta classe de grafos como ordem indiferença e aresta maximal. O problema da coloração de arestas em grafos indiferença já vem sendo estudado a vários anos, conseguiu-se resolver o problema para grafos indiferença com determinadas características. Apresentaremos os resultados conhecidos para os grafos indiferença de grau máximo ímpar e para os grafos indiferença reduzidos.

3.1 A classe dos grafos indiferença

Um grafo é dito de *intervalos* caso seja o grafo de intersecção de uma família de intervalos da reta real. Isto acontece caso o seu conjunto de vértices admita uma bijeção com um conjunto de intervalos da reta real, tal que exista uma aresta entre dois vértices do grafo se e somente se os intervalos correspondentes têm intersecção. Grafos indiferença são uma subclasse dos grafos de intervalos. Um grafo de intervalos é dito *indiferença* caso os intervalos possam ser tomados unitários. A restrição sobre o tamanho dos intervalos gera uma subclasse própria dos grafos de intervalos. Na verdade, esta subclasse é exatamente a dos grafos de intervalos que não admitem o grafo bipartido completo $K_{1,3}$ como subgrafo induzido, conforme foi caracterizado por *Gilmore e Hoffman* em 1964 [17].

Roberts em 1969, demonstrou o seguinte teorema a respeito dos grafos indiferença.

Para o teorema faz-se necessário as seguintes definições:

Definição 3.1 *Considerando $\delta > 0$, uma função real $u : V \to \mathbb{R}$ é chamada semiordem para uma relação binária (V, P), se a seguinte condição é válida.*

$$xy \in P \Leftrightarrow u(x) \geq u(y) + \delta \qquad (x, y \in V).$$

É provado que existe um semiordem para a relação binária (V, P) se e somente se as seguintes condições são válidas.

Para todo $x, y, z, w \in V$,

- *P é irreflexiva (ou seja, $(x, x) \notin P$ para todo $x \in V$).*
- *$xy \in P$ e $zw \in P$ implica $xw \in P$ ou $zy \in P$.*
- *$xy \in P$ e $yz \in P$ implica $xw \in P$ ou $wz \in P$.*

Definição 3.2 *Um grafo G é dito grafo de intervalos próprios se G é um grafo de intervalos e nenhum de seus intervalos está totalmente contido em outro.*

Definição 3.3 *Denotamos por P^{-1} a relação binária onde $(y, x) \in P^{-1}$ se e somente se $(x, y) \in P$.*

Denotamos por $\overline{E}$ o conjunto de pares não ordenados que é o complemento de E em relação ao conjunto de todos os pares não ordenados de elementos de V.

Teorema 3.1 (Roberts (1969) [32]) *Considere $G = (V, E)$ sendo um grafo não direcionado. As seguintes condições são equivalentes.*

(i) Existe uma função real u: $V \to \mathbb{R}$ que satisfaça, para todos os vértices $x, y \in V$,

$$xy \in E \longleftrightarrow |\, u(x) - u(y) \,| < 1.$$

(ii) Existe uma semiordem (V, P) tal que $\overline{E} = P + P^{-1}$.

(iii) $\overline{G} = (V, \overline{E})$ é um grafo de comparabilidade e toda orientação transitiva de $\overline{G}$ é uma semiordem.

(iv) G é um grafo de intervalos que não possui nenhum $K_{1,3}$ como subgrafo induzido.

(v) G é um grafo de intervalos próprios.

(vi) G é um grafo de intervalos unitários (indiferença).

3.1.1 Ordem indiferença

Outra caracterização para grafos indiferença é em função de uma ordem linear sobre seus vértices. Em 1964, *Gilmore e Hoffman* [17] demonstraram que grafos de intervalos são os que admitem uma ordem linear sobre as suas cliques maximais, na qual as que contém um dado vértice são consecutivas. *Maehara* [26] demonstrou que um grafo é indiferença se e somente se admite uma ordem linear sobre os seus vértices, na qual os vértices contidos em uma mesma clique maximal são consecutivos, esta ordem é denominada *ordem indiferença.*

Roberts [33] também caracterizou a classe dos grafos indiferença através de uma ordem linear. Uma ordem linear sobre os vértices de um grafo é dita compatível com a relação de adjacência E quando $x < y < z$ e $\{x, z\} \in E$ implica que $\{x, y\}, \{y, z\} \in E$. Conforme observado em [13] as ordens lineares definidas por Maehara e Roberts são equivalentes, de fato, dada uma ordem compatível e uma clique maximal em G, considerando o primeiro e o último vértice desta clique temos que os vértices entre estes dois são exatamente os vértices da clique maximal. Por outro lado, dada uma ordem indiferença, $x < y < z$ e $\{x, z\} \in E$, temos que y, assim como todos os vértices pertencentes a clique maximal que contém x e z, é adjacente tanto a x como a z.

Jayme Szwarcfiter, Celina M. H. Figueiredo, Sulamita Klein e Célia P. Mello em [13], demonstram duas caracterizações para ordens indiferença. Pode-se reconhecer grafos indiferença através destas caracterizações. A seguir apresentaremos os dois lemas demonstrados pelos autores.

A primeira caracterização permite reconhecer grafos indiferença através de uma ordem linear. É observado que uma ordem indiferença é um esquema de eliminação perfeita especial para os vértices do grafo. Para caracterizar os vértices simpliciais que definem esta ordem é necessária a seguinte definição.

Definição 3.4 *Um vértice simplicial v de G é extremo se G for uma clique, ou caso contrário, para todo par x,y de vértices adjacentes a v, tal que x,y e v não são gêmeos, existe pelo menos um vértice z em G adjacente a x e a y, mas não a v.*

O lema a seguir identifica quais vértices de um grafo indiferença poderão se constituir no primeiro ou no último elemento de uma ordem indiferença.

Lema 3.2 *Os vértices gêmeos ao primeiro e os vértices gêmeos ao último vértice de uma ordem indiferença são os únicos vértices extremos de um grafo indiferença.*

A segunda caracterização da ordem indiferença dada em [13] é em função de uma condição de conectividade sobre as suas cliques maximais. O lema a seguir caracteriza e fornece um método alternativo para reconhecimento de grafos indiferença.

Lema 3.3 *Dado um grafo indiferença que não é uma clique, cada um dos seus subgrafos induzidos conexos tem exatamente duas cliques maximais cuja remoção não desconecta o grafo.*

3.1.2 Reconhecimento de grafos indiferença

Reconhecer grafos indiferença é de complexidade de tempo linear. Vários algoritmos que fazem este reconhecimento em tempo linear estão disponíveis na literatura.

Em [7] é apresentado um algoritmo que reconhece grafos indiferença em tempo linear. É um algoritmo de dois passos, no primeiro passo é feita uma busca em largura lexicográfica no grafo enquanto o segundo passo utiliza a saída da busca para particionar o grafo em classes de vértices gêmeos. Desta partição é dada a ordem indiferença do grafo.

Em [5] é apresentado um algoritmo que também reconhece grafos indiferença em tempo linear. Este algoritmo é baseado em uma caracterização de grafos indiferença em função de uma ordem de seus vértices. *Roberts* provou que um grafo é indiferença se e somente se existe uma ordem $<$ sobre seus vértices de modo que para todo vértice v do grafo, a vizinhança de v é um conjunto de vértices consecutivos (com relação a $<$). O algoritmo utiliza busca em largura no grafo para definir uma ordem linear e depois verifica se esta ordem é uma ordem indiferença.

3.1.3 Arestas maximais

Fixe uma certa ordem indiferença num grafo indiferença G. Para cada clique maximal A de G, definimos como *aresta maximal* a aresta cujos vértices são o primeiro e o último vértices de A com relação a ordem indiferença de G.

A figura 3.1 mostra o grafo completo K_5 e ao lado uma de suas ordens indiferença. Nesta figura as arestas tracejadas indicam as arestas maximais. O K_5 tem apenas uma clique maximal que é o próprio grafo, então qualquer ordem linear de seus vértices é uma ordem indiferença.

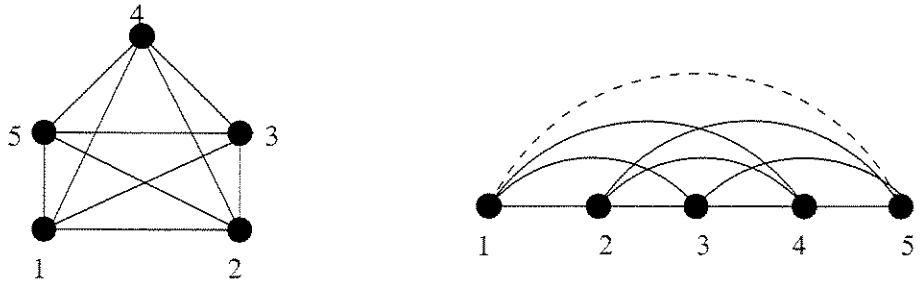

Figura 3.1: K_5 e uma ordem indiferença

A figura 3.2 apresenta um outro grafo e uma ordem indiferença. Nesta figura as arestas tracejadas indicam as arestas maximais. Observe que o grafo possui três cliques maximais e para cada clique há uma aresta maximal.

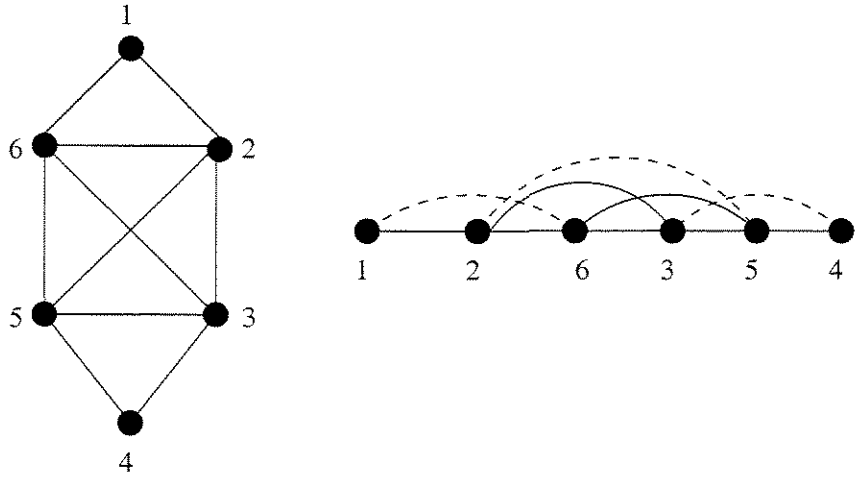

Figura 3.2: Um grafo com três cliques maximais e uma ordem indiferença

É interessante notar que visualmente falando uma aresta maximal é aquela que não está "encoberta" por nenhuma outra aresta. O subgrafo formado pelo intervalo de vértices,

na ordem indiferença. delimitado por cada um dos vértices de uma aresta maximal é uma clique.

3.2 Coloração em grafos indiferença

Nesta seção apresentaremos dois resultados conhecidos para coloração arestas em grafos indiferença. Este problema já vem sido estudado a bastante tempo e conseguiu-se resultados parciais para a classe dos grafos indiferença. Mostraremos como foi resolvido o problema da classificação para os grafos indiferença de máximo ímpar e grafos indiferença reduzidos.

3.2.1 Grafos indiferença de grau máximo ímpar

Neste trabalho os autores mostram que todo grafo indiferença de grau máximo ímpar pode ter suas arestas coloridas com Δ cores.

Teorema 3.4 (Figueiredo, Meidanis e Mello - 1997[9]) *Todos os grafos indiferença com grau máximo ímpar pertencem a Classe* 1.

Prova:

Considere G sendo um grafo indiferença onde $\Delta = \Delta(G)$ é ímpar.

Organiza-se os vértices em uma ordem indiferença e numera-se eles com inteiros consecutivos começando do 0. Também rotula-se cada vértice com um elemento do conjunto $\{0,1,2,...,\Delta - 1,D\}$, onde D é um símbolo especial. A rotulação é a seguinte:

$$\lambda(x) = \begin{cases} D, & \text{se } x \equiv \Delta \bmod(\Delta + 1) \\ x \bmod (\Delta + 1), & \text{caso contrário.} \end{cases}$$

A cor $\kappa(xy)$ de uma aresta xy será a função simétrica dos rótulos de x e y. As cores utilizadas serão os elementos do conjunto $\{0,1,2,...,\Delta - 1\}$, associadas da seguinte forma:

$$\kappa(xy) = \begin{cases} \lambda(x) + \lambda(y) \bmod \Delta, & \text{se tanto } \lambda(x) \text{ e } \lambda(y) \text{ são numéricos} \\ 2\lambda(x) \bmod \Delta, & \text{se } \lambda(x) \text{ é numérico e } \lambda(y) = D \\ 2\lambda(y) \bmod \Delta, & \text{se } \lambda(x) = D \text{ e } \lambda(y) \text{ é numérico} \end{cases}$$

Nenhuma regra é necessária para colorir arestas conectando dois vértices rotulados de D, porque tal aresta não existe. Na verdade se um vértice x rotulado com D fosse adjacente a outro vértice y também rotulado com D, supondo que existe a aresta xy, pela propriedade da ordem indiferença este dois vértices seriam adjacentes a todos os vértices entre eles na ordem indiferença. Pelo processo de rotulação entre o vértice x e y existem Δ vértices, portanto teríamos $gr_G(x) \geq \Delta + 1$, contrariando o fato de que Δ é o grau máximo. Este mesmo argumento pode ser usado para mostrar que:

1. Não existe aresta entre dois vértices com o mesmo rótulo.

2. Nenhum vértice possui dois vizinhos com o mesmo rótulo.

Para mostrar que κ é uma coloração. Suponha que existem duas arestas distintas xy e xz, pelo argumento anterior, sabemos que os rótulos de x, y e z são distintos. Assumindo que $\kappa(xy) \equiv \kappa(xz)$, então existem dois casos a considerar.

Caso 1: $\lambda(x) = D$

Neste caso, os rótulos de y e z são necessariamente numéricos e

$$2\lambda(y) \equiv 2\lambda(z) \pmod{\Delta}.$$

Como Δ é ímpar, pode-se cancelar o fator 2 de ambos os lados para obter

$$\lambda(y) \equiv \lambda(z) \pmod{\Delta}.$$

Isto implica que $\lambda(y) = \lambda(z)$, o que é uma contradição.

Caso 2: $\lambda(x)$ é numérico.

Neste caso, $\lambda(y)$ e $\lambda(z)$ não podem ser, simultaneamente, D. Sendo apenas um deles D, supõe-se que, $\lambda(y) = D$, então:

$$2\lambda(x) \equiv \lambda(x) + \lambda(z) \pmod{\Delta},$$

O que implica que $\lambda(x) = \lambda(z)$, uma contradição.

Pode ser que os rótulos dos três vértices sejam numéricos, então

$$\lambda(x) + \lambda(y) \equiv \lambda(x) + \lambda(z) \pmod{\Delta},$$

O que implica que $\lambda(y) = \lambda(z)$, o que também é uma contradição.

Conclui-se que as cores de xy e xz não podem ser iguais e que esta associação (conhecida como Método do Dobrador) gera uma Δ-coloração de arestas. □

A prova do teorema anterior induz um algoritmo, que gera uma Δ-coloração para qualquer grafo indiferença de grau máximo ímpar. Mostraremos um exemplo destacando a construção da coloração.

A figura 3.3 apresenta um grafo indiferença de grau máximo ímpar e a uma ordem indiferença para o grafo.

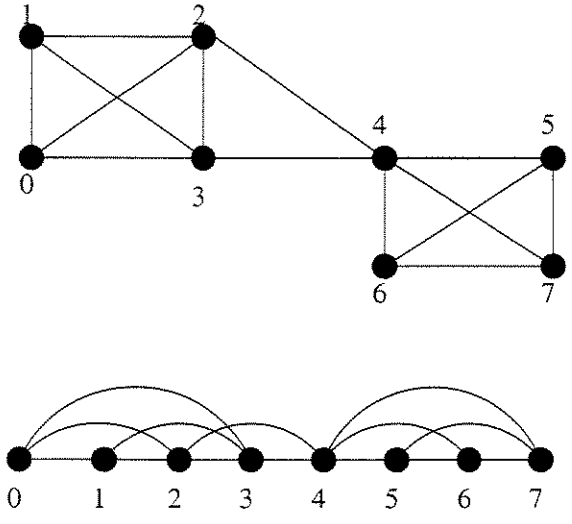

Figura 3.3: Grafo indiferença de grau máximo ímpar

Já na figura 3.4, pode-se observar como os vértices na ordem indiferença são rotulados.

E por fim, na figura 3.5 é apresentada uma Δ-coloração para o grafo baseada na função de coloração demonstrada na prova do teorema.

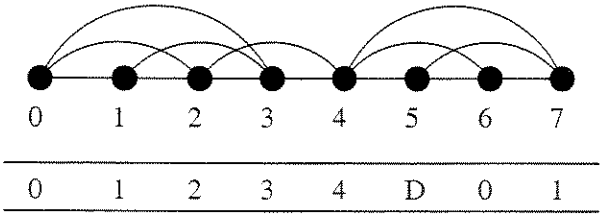

Figura 3.4: Rotulação dos vértices baseada na ordem indiferença.

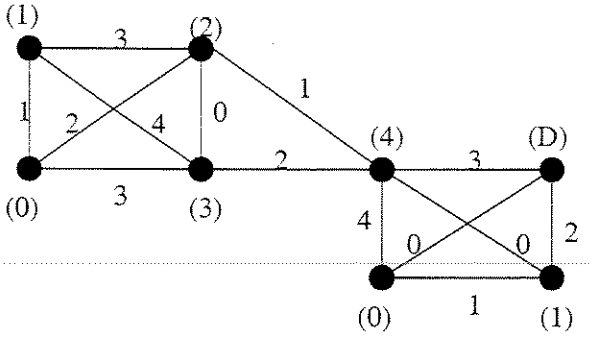

Figura 3.5: Δ-coloração para o grafo.

3.2.2 Grafos indiferença reduzidos

Neste trabalho os autores mostram que qualquer grafo indiferença reduzido de grau máximo par pode ter suas arestas pintadas com Δ cores. Na verdade a prova se estende a todos os grafos indiferença de grau máximo par que não possuem dois Δ-vértices gêmeos. Os autores também mostram que um grafo indiferença sem 2 Δ-vértices gêmeos nunca é neigborhood overfull.

Um grafo é dito ser indiferença reduzido se for um grafo indiferença e não possuir vértices gêmeos.

Neste trabalho é exibida uma Δ-coloração de arestas para um grafo indiferença reduzido de grau máximo par. Como já é sabido que grafos indiferença de grau máximo ímpar são *Classe* 1, este resultado implica que todos grafos indiferença sem dois Δ-vértices gêmeos, e em particular todos os grafos indiferença reduzidos, são *Classe* 1.

Para construir uma Δ-coloração, é aplicada a técnica da decomposição sobre um grafo

indiferença de grau máximo par e sem dois Δ-vértice gêmeos. As arestas do grafo serão particionadas em dois conjuntos E_1 e E_2, tais que, $G_1 = G[E_1]$ é um grafo indiferença de grau máximo ímpar onde $\Delta(G_1) = \Delta(G) - 1$ e $G_2 = G[E_2]$ é um emparelhamento.

Em um grafo indiferença G, cada vértice é incidente a zero, uma ou duas arestas maximais. Dada um grafo indiferença com uma ordem indiferença e uma aresta que é maximal nesta ordem, a remoção desta aresta gera um grafo indiferença menor. A ordem indiferença original é uma ordem indiferença para o grafo indiferença gerado.

O Lema a seguir permite a formulação de um algoritmo para escolher um emparelhamento de um grafo indiferença sem dois Δ-vértices gêmeos que cobre todos Δ-vértices de G.

Lema 3.5 *Considere G como um grafo não trivial. Se G é um grafo indiferença sem Δ-vértices gêmeos, então todo Δ-vértice de G é incidente a precisamente duas arestas maximais.*

Prova:

Considere G sendo um grafo indiferença não trivial sem Δ-vértices gêmeos e seja v um Δ-vértice de G. Considere $v_1, v_2, ..., v_n$ sendo uma ordem indiferença de G. Sendo v um Δ-vértice de G e G não sendo uma clique, temos que $v = v_j$ com $j \neq 1, n$. Considere v_i e v_k sendo os vértices mais a esquerda e mais a direita respectivamente adjacentes a v_j na ordem indiferença. Suponha que v_iv_j não é uma aresta maximal. Então $v_{i-1}v_j$ ou v_iv_{j+1} é uma aresta em G. A existência de $v_{i-1}v_j$ contradiz que v_i é o vizinho mais a esquerda de v_j . Como v_j e v_{j+1} não são gêmeos, a existência de v_iv_{j+1} implica que $gr_G(v_{j+1}) > \Delta(G)$, uma contradição. Analogamente, temos que v_iv_k também é uma aresta maximal. □

O seguinte algoritmo escolhe um conjunto de arestas maximais que cobre todos os Δ-vértices de G.

Algoritmo:

Entrada: Um grafo indiferença G sem dois Δ-vértices gêmeos com uma ordem indiferença $v_1, ..., v_n$ de G

Saída: Um conjunto de arestas M que cobre todos os Δ-vértices de G.

1 - Para cada Δ-vértice de G, denominado v_j, na ordem indiferença, coloque a aresta $v_i v_j$ no conjunto ε, onde v_i é o vizinho mais a esquerda de v_j na ordem indiferença. Cada componente do grafo $G[\varepsilon]$ é um caminho.

2 - Para cada componente caminho P de $G[\varepsilon]$, numere cada aresta com inteiros consecutivos a partir de 1. Se um componente caminho P_i contém um número ímpar de arestas, então forme um emparelhamento M_i de $G[\varepsilon]$ escolhendo as arestas numeradas por números ímpares. Se um componente caminho P_j contém um número par de arestas, então forme um emparelhamento M_j escolhendo as arestas numeradas por números pares.

3 - O emparelhamento de arestas M desejado é a união $\bigcup_k M_k$.

O algoritmo acima garante que o emparelhamento M cobre todos os Δ-vértices de G. Se um componente caminho de $G[\varepsilon]$ contém um número ímpar de aresta, então M cobre todos os seus vértices. Se um componente caminho de $G[\varepsilon]$ contém um número par de arestas, então o único vértice não coberto por M é o primeiro vértice do componente caminho e pela definição de $G[\varepsilon]$ este vértice não é um Δ-vértice.

Teorema 3.6 (Figueiredo, Meidanis, Mello e Ortiz - 2003[12]) *Se G é um grafo indiferença sem Δ-vértices gêmeos, então G é Classe* 1.

Prova:

Considere G sendo um grafo indiferença sem Δ-vértices gêmeos, se G tem grau máximo ímpar então G é *Classe* 1 conforme demonstrado na Seção [3.2.1]. Suponha que G possui grau máximo par. Considere $v_1, ...v_n$ sendo uma ordem indiferença de G. Use o algoritmo descrito acima para encontrar um emparelhamento M para G que cobre todos os Δ-vértices de G. O grafo $G \backslash M$ é um grafo indiferença de grau máximo ímpar porque os vértices de G e $G \backslash M$ são os mesmos e a ordem indiferença de G é também uma ordem indiferença para $G \backslash M$. Mais ainda, se M é um emparelhamento que cobre todos Δ-vértices, temos então que $\Delta(G \backslash M) = \Delta(G) - 1$ é ímpar. Como as arestas de $G \backslash M$ podem ser coloridas com $\Delta(G) - 1$ cores e somente uma cor adicional é necessária para colorir as arestas do emparelhamento M. Isto implica então que $\chi' = \Delta - 1 + 1 = \Delta$, ou seja, G pertence a *Classe* 1. □

Este teorema é automaticamente válido para todos os grafos indiferença reduzidos, já que grafos indiferença reduzidos não permitem quaisquer dois vértices gêmeos e por conseqüência não permitem Δ-vértices gêmeos.

Será apresentado um exemplo de como a prova do teorema gera uma Δ-coloração para os grafos reduzidos.

Considere o grafo G representado na figura 3.6, sendo G sem Δ-vértices gêmeos e com grau máximo par.

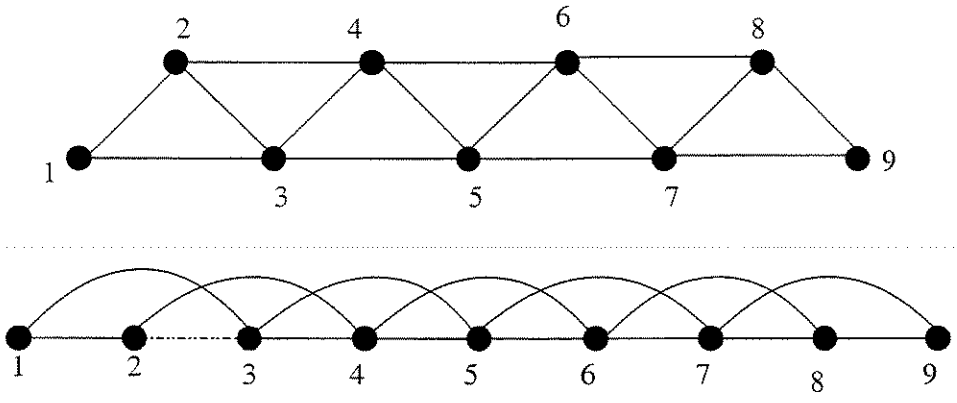

Figura 3.6: Grafo indiferença reduzido com grau máximo par

Observe na figura 3.7 as arestas maximais que devem fazer parte do conjunto ε.

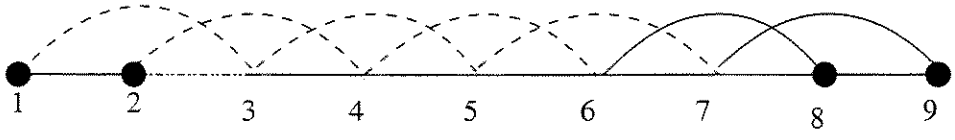

Figura 3.7: Coloração de grafos indiferença reduzidos: obtendo as arestas maximais de E

A figura 3.8 apresenta $G[\varepsilon]$, grafo G induzido pelas arestas do conjunto ε, com dois componentes conexos, que são *caminhos*. Conforme diz o algoritmo apresentado as arestas do caminho devem ser rotuladas com números consecutivos começando do 1. Caso o número de arestas da componente seja par, deve-se criar um emparelhamento M_i com as arestas da componente rotuladas por números ímpares. Caso contrário, se o número de

arestas da componente for ímpar, deve-se colocar no emparelhamento M_i as arestas rotuladas por números pares. Então teremos $M_1 = \{\{4,6\}\}$ e $M_2 = \{\{1,3\},\{5,7\}\}$. O emparelhamento M deve ser formado pela união de todos M_i, então $M = \{\{1,3\},\{4,6\},\{5,7\}\}$. Observe que M cobre todos os Δ-vértices.

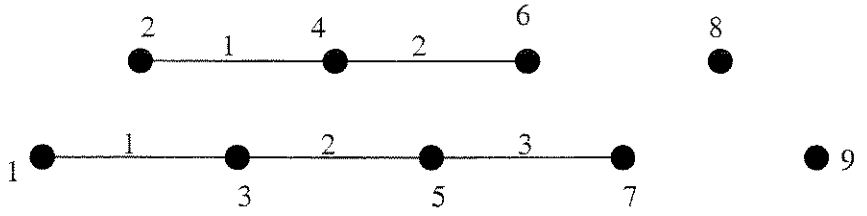

Figura 3.8: Coloração de grafos indiferença reduzidos: componentes conexos de $G[E]$.

Finalmente na figura 3.9 podemos observar o grafo $G[G \backslash M]$ onde este grafo é um grafo indiferença cujo grau máximo é ímpar e é igual a $\Delta(G) - 1$. Como já visto, este grafo tem uma $(\Delta(G) - 1)$-coloração. As arestas que faltam em $G[G \backslash M]$ com relação a G são justamente as arestas do emparelhamento M, todas as arestas de M podem ser coloridas com única cor. A cor não utilizada na coloração de $G[G \backslash M]$ pode ser utilizada para colorir as arestas de $G[M]$. Portanto, o grafo G possui uma Δ-coloração.

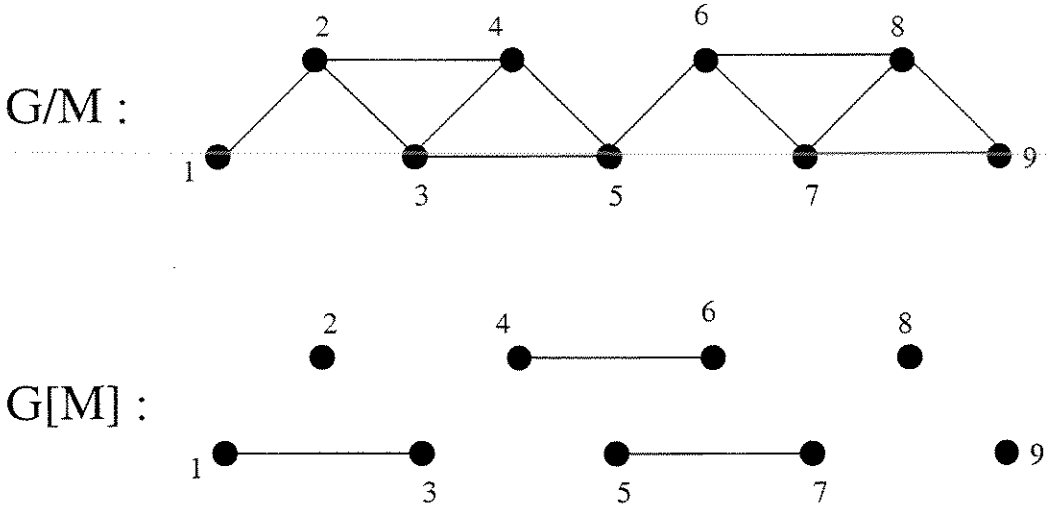

Figura 3.9: Coloração de grafos indiferença reduzidos: decomposição final.

Capítulo 4

Geração e coloração de grafos indiferença

4.1 Modelos de programação linear

Os problemas de otimização são problemas de maximização ou minimização de funções, designadas por objetivo, que dependem de um número finito de variáveis. Estas variáveis podem ser independentes uma das outras, ou podem estar relacionadas através de uma ou mais restrições.

A classe de problemas onde o objetivo e as restrições são dadas como funções matemáticas e relações funcionais é denominada *problemas de programação matemática.*

Problemas de programação linear são uma classe de problemas de programação matemática, onde a função objetivo e suas restrições podem ser representadas por funções lineares.

Os problemas de programação linear podem ser formulados de acordo com um modelo matemático geral, que consiste na determinação de valores não negativos para as variáveis $x_1, x_2, \ldots, x_j, \ldots, x_n$ satisfazendo um sistema de m equações (inequações) lineares que maximizem ou minimizem o valor de uma função (real) linear dessas variáveis.

Um modelo de programação linear é composto por:

- Um conjunto de n variáveis $(x_1, x_2, x_3, x_4, \ldots, x_n)$;

- Um conjunto de m condições lineares relacionando as variáveis por meio de desigualdades do tipo $a_1x_1 + a_2x_2 + a_3x_3 + \ldots + a_nx_n \leq b_1$, onde a_i e b_i são coeficientes reais;

- Uma função linear, também conhecida como função objetivo, a qual determina a minimização ou a maximização das variáveis.

Em adição, problemas de programação inteira contém um conjunto de restrições de domínio das variáveis.

Em 1947, o matemático americano *George Dantzig* estabeleceu uma forma sistemática de resolução do modelo de programação linear baseada na teoria das equações lineares, no conceito de independência de vetores e na álgebra matricial. Este método ficou conhecido como *método do simplex*.

4.2 CPLEX

Para resolução de nossos modelos lineares, utilizamos um software de programação matemática e otimização, CPLEX Linear Optimizer 7.0.0 ®. Este software está disponível nos laboratórios do IC-Unicamp.

CPLEX é um sistema otimizador linear interativo que inclui diversos algoritmos para resolução de problemas modelados linearmente com uma interface de software interativo. A entrada do problema linear pode ser feita de duas formas: através de uma entrada interativa dentro do próprio sistema ou através de um arquivo que deve obedecer a alguns formatos pré-estabelecidos.

Outro recurso bastante interessante do CPLEX é a disponibilização de uma biblioteca de funções que podem ser chamadas de um programa qualquer. A API desta biblioteca fornece toda a funcionalidade do CPLEX em mais de 200 rotinas que podem ser linkadas por programas escritos em C, Fortran e outras linguagens.

Esta poderosa biblioteca permite a definição, resolução, análise, consultas e criação de relatórios para problemas de programação linear. A biblioteca foi criada especificamente para integrar o alto desempenho dos algoritmos CPLEX à aplicações especializadas.

O sistema CPLEX possui também um "*Mixed Integer Solver*" que possui a capacidade para resolver problemas muito rapidamente tanto com variáveis inteiras gerais, como variáveis binárias.

4.3 Modelo linear proposto para coloração de arestas

O modelo linear tem por objetivo modelar o problema de coloração de arestas para grafos indiferença para um modelo matemático que pode ser resolvido pelo CPLEX.

Considere um conjunto E de arestas, sendo que estas arestas são: $\{1, 2, 3, ..., m\}$. Considere também um conjunto de cores $\{1, 2, ..., \Delta(G) + 1\}$

As variáveis do modelo são :

$$x_c = \begin{cases} 1 & \text{se cor } c \text{ está sendo usada} \\ 0 & \text{caso contrário.} \end{cases}$$

$$x_{ec} = \begin{cases} 1 & \text{se aresta } e \text{ tem cor } c \\ 0 & \text{caso contrário.} \end{cases}$$

Restrições :

(Pintar) - Restringe que toda aresta tenha exatamente uma cor.

$$\sum_{c \in \text{ Cores}} x_{ec} = 1 \ , \forall \ e \in \text{Arestas}$$

(Conflito) - Restringe que arestas adjacentes tenham cores diferentes.

$$\sum_{e \in \text{ Adj}(v)} x_{ec} \leq 1 \ , \forall \ v \in \text{Vértices} \ , \ \forall \ c \in \text{Cores}$$

(Uso) - Registra a utilização das cores nas variáveis x_c, que são as únicas que aparecerão na função objetivo.

$$x_{ec} \leq x_c \ , \forall \ e \in \text{Arestas} \ , \ \forall \ c \in \text{Cores}$$

Função Objetivo :

$$f = \sum_{c \in \text{Cores}} x_c$$

Queremos minimizar f. Se $\min f = \Delta$ então o grafo é *Classe* 1, caso contrário, se $\min f = \Delta + 1$ então o grafo é *Classe* 2.

Podemos fazer a análise de custo do modelo, considerando o tamanho de cada restrição como sendo o número de variáveis envolvidas, da seguinte forma:

$x_{ec} =$	$O(m\Delta)$	Variáveis
$x_c =$	$O(\Delta)$	Variáveis
Pintar =	$O(m\Delta)$	Tamanho das restrições
Conflito =	$O(m\Delta)$	Tamanho das restrições
Uso =	$O(m\Delta)$	Tamanho das restrições
$f =$	$O(\Delta)$	Variáveis

O tamanho das restrições de conflito é $O(m\Delta)$, pois cada variável x_{ec} aparecerá apenas duas vezes no total, uma para cada extremo da aresta e.

O custo total do modelo é $O(m\Delta)$, incluindo variáveis e tamanho total das restrições.

4.4 Gerando grafos indiferença

Como pretendíamos analisar o comportamento do problema em relação ao tamanho da entrada, decidimos criar e resolver modelos lineares para o problema da coloração de arestas em grafos indiferença através de um *solver* (CPLEX). Para conseguirmos este objetivo precisávamos implementar um software para gerar estes grafos, transformar o grafo gerado em um modelo linear e interagir com o CPLEX para resolução do modelo.

A geração destes grafos deveria ser bastante flexível, pois gostaríamos de gerar grafos indiferença de diferentes estruturas e com diferentes propriedades.

Para gerarmos os grafos, utilizamos o seguinte método:

- Organizamos os vértices em uma ordem linear, que será a ordem indiferença do grafo.
- A partir da ordem linear, geramos somente arestas maximais entre os vértices.

Gerando os grafos desta forma temos a certeza que todo grafo gerado será um grafo indiferença, pois um grafo é indiferença se e somente se possui uma ordem indiferença sobre seus vértices. O que fazemos é gerar uma ordem indiferença para o grafo.

A figura 4.1 apresenta um exemplo da forma como o grafo é gerado.

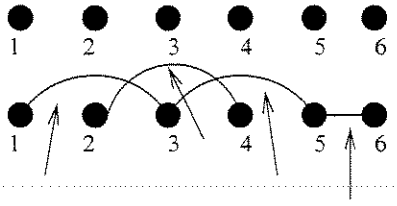

Figura 4.1: Geração de um grafo indiferença de 6 vértices

Na parte superior da figura, os vértices são colocados em uma ordem linear que será a ordem indiferença do grafo. Já na parte inferior da figura, existem algumas arestas maximais destacadas por setas. Só são geradas as arestas maximais, já que para determinamos as demais arestas do grafo sabemos que os vértices "encobertos" por cada aresta maximal formam uma clique.

Determinamos que gostaríamos de gerar grafos em função de três parâmetros que regem a estrutura do grafo gerado. São eles:

- Quantidade de vértices, n.
- Quantidade de arestas, m.
- Grau máximo desejado, Δ.

Antes de gerar o grafo é necessário definir o valor dos parâmetros para a estrutura desejada.

4.4.1 O algoritmo para geração randômica de grafos

Vamos descrever a idéia básica do algoritmo que utilizamos para geração dos grafos indiferença. Os vértices são numerados de 1 a n, onde n é o número de vértices desejados. Começamos com o grafo sem nenhuma aresta. Existe um laço principal que passa por todos os vértices. Em cada instância desse laço pode ser criada ou não uma aresta maximal para o vértice. Só são geradas arestas maximais. Todos os grafos gerados devem ser conexos, pois neste trabalho só estamos olhando para grafos indiferença conexos.

A cada iteração, são feitas duas perguntas:

- Quantas arestas este vértice DEVE gerar considerando o número máximo de arestas, Δ e que o grafo é conexo?
- Quantas arestas este vértice PODE gerar considerando o número máximo de arestas e Δ?

Então, a cada iteração são calculados dois valores, o número máximo de arestas que pode ser gerado e o número mínimo de arestas que deve ser gerado, para que se mantenham os valores requeridos de arestas e Δ. O número de arestas a ser geradas é dado por um número randômico entre a quantidade máxima e mínima de arestas calculada. Esta quantidade de arestas é transformada em distância de vértices, ou seja, por quantos vértices esta aresta maximal deverá passar para respeitar os quesitos de conectividade, arestas e Δ desejados. Lembrando que os vértices já estão numerados e que esta ordem deve ser uma ordem indiferença.

A forma geral do algoritmo é a seguinte :

Algoritmo de geração de grafos indiferença

```
for (vertice ← 1;vertice ≤ VERTICES;vertice++)
    min ← minimo();
    max ← maximo();
    casas ← numero randômico entre min e max;
    criar aresta maximal (vértice,vértice + casas);
    se VerticeAtual + Casas = VERTICEFINAL
```

então
sair do laço;
senão
atualizar o numero de arestas geradas;
atualizar o Δ;

4.4.2 Armazenando os grafos gerados

O primeiro problema encontrado nesta abordagem foi como armazenar os grafos gerados, de forma que os arquivos gerados não ocupassem espaço demasiado, já que trabalharíamos com um número muito grande de grafos.

Já vimos que os grafos gerados possuem somente arestas maximais. Sabemos através da propriedade da ordem indiferença que um grafo indiferença possui várias arestas maximais e que todos os vértices "encobertos" por cada uma destas arestas maximais formam uma clique. Então, se soubermos apenas quais são as arestas maximais de um grafo podemos deduzir todas as demais arestas.

Visando minimizar espaço em memória e disco, representamos os grafos somente através das arestas maximais, já que quando gerarmos os modelos lineares e precisarmos das demais arestas sabemos como determiná-las. Desta forma, sempre trabalhamos somente com as cliques maximais do grafo, armazenando o vértice inicial e o vértice final de cada clique maximal do grafo com relação à ordem indiferença.

A figura 4.2 ilustra o arquivo de um grafo gerado, mostrando como este grafo é armazenado.

A primeira linha é um delimitador que marca o inicio do grafo. A segunda linha diz a quantidade de vértices do grafo. A terceira linha diz a quantidade de arestas do grafo. A quarta linha diz o Δ do grafo. A quinta linha traz separados por ponto e vírgula os vértices do grafo que possuem grau igual a Δ. As linhas subseqüentes determinam as arestas maximais do grafo (vértice inicial e o vértice final para cada aresta). As duas últimas linhas marcam o fim do grafo.

```
$
30
101
10
12;22;
1–3
2–4
3–6
4–10
5–12
12–15
17–21
18–22
22–28
24–29
25–30
0–0
$
```

Figura 4.2: Arquivo gerado de um grafo

4.4.3 A estrutura do gerador

Criamos o gerador de forma que outros módulos e funções pudessem ser acopladas facilmente sem influenciar nas funções já existentes.

O gerador foi criado através de vários módulos independentes que são chamados através de um módulo principal. O módulo principal passa aos demais módulos os parâmetros exigidos. O módulo principal faz a interface da aplicação. Cada função do programa possui um comando associado, todo comando é formado por um caracter. O modulo principal faz a leitura dos comandos e chama a função associada.

Além da interatividade com o usuário, que pode escolher diferentes funções e comandos em um mesmo programa, o gerador também permite a leitura de vários comandos através de um arquivo de entrada passado pelo interpretador de comandos (*shell*) do sistema operacional. Desta forma, é possível organizar um processo para gerar e trabalhar com milhares de grafos sem a interferência direta do usuário, sendo controlado apenas através deste arquivo que é colocado na entrada padrão de onde o gerador lê seus comandos. Esse

arquivo, em formato texto, pode facilmente ser gerado através de qualquer editor, scripts, linguagem interpretada ou mesmo qualquer linguagem de programação.

Os comandos aceitos pelo gerador estão relacionados abaixo. Alguns deles não existiam nas primeiras versões do gerador, tendo sido motivados pelas pesquisas relatadas no capítulo 5.

- h : ajuda.
- s : termina o programa.
- m <valor> : seta o parâmetro m, número de arestas.
- n <valor> : seta o parâmetro n, número de vértices.
- d <valor> : seta o parâmetro Δ.
- p : imprime parâmetros n, m e Δ na tela.
- g <arqsaida> : gera um grafo indiferença randômico, delimitado pelos parâmetros setados. O grafo gerado é gravado em um arquivo cujo nome é passado como parâmetro em <arqsaida>.
- G : gera todos os grafos indiferença com n vértices. Cada grafo é gravado em um arquivo cujo nome é prefixado por "g_" seguido de um número seqüencial, ex: g_1, g_2, g_3, g_4, ...
- u : gera todos os grafos indiferença com n vértices que tenham pelo menos 1 vértice universal. Cada grafo é gravado em um arquivo cujo nome é prefixado por "gu_" seguido de um número seqüencial, ex: gu_1, gu_2, gu_3, gu_4, ...
- w <arqentrada> : verifica a condição do grafo passado como parâmetro. As condições são Δ par ou ímpar, overfull, neighborhood overfull. As condições são impressas na tela e também gravadas em um arquivo chamado "verif.log" que mantém o resultado de todos os grafos verificados.

- v <arqentrada> : verifica a coloração gerada pelo CPLEX para o grafo passado como parâmetro. Quando o CPLEX resolve um modelo linear, são gerados dois arquivos. Estes arquivos possuem o mesmo nome do arquivo com o modelo linear (".lp") passado ao CPLEX, alterando apenas a extensão para ".log" e ".mst". O arquivo ".log" contém como informações mais relevantes, o tempo gasto para resolução e o valor da função objetivo. Já o arquivo ".mst" contém os valores das variáveis do modelo. Esta função do nosso gerador verifica se o arquivo passado como parâmetro mais a extensão ".mst" existe, caso exista verifica se os valores das variáveis do modelo, que estão neste arquivo (".mst"), formam uma coloração válida para o grafo. Como resultado, é impresso na tela "Coloração Válida" ou "Coloração Inválida" e também são gravados em um arquivo chamado "cores.log", o nome do arquivo e o status da coloração.

- t <arqentrada> : cria o modelo linear para o grafo representado no arquivo passado como parâmetro. O modelo é gravado em um arquivo com o mesmo nome passado como parâmetro alterando apenas a extensão para ".lp".

- a <arqentrada> : cria o modelo linear para o grafo representado no arquivo passado como parâmetro. Adiciona restrições para que a coloração gerada seja balanceada. O modelo é gravado em um arquivo com o mesmo nome passado como parâmetro alterando apenas a extensão para ".lp".

- r <arqentrada> : "Resolve" um grafo passado como parâmetro. Gera o modelo linear, interage com o CPLEX para resolução do modelo, verifica a coloração.

Os principais módulos do Gerador são:

Geraran: chamado através do comando "g" <arqsaida>. Gera grafos randomicamente obedecendo aos parâmetros: quantidade de vértices, quantidade de arestas e delta. Este módulo gera o grafo e armazena no arquivo cujo caminho é passado como parâmetro para o módulo.

Gerador: chamado através do comando "G". Gera todos os grafos possíveis obedecendo somente ao parâmetro quantidade de vértices. Cada grafo gerado é armazenado

em um arquivo cujo nome é determinado pela seqüência do grafo. Também é gerada uma lista com todos os arquivos gerados.

Transfor: chamado através do comando "t" <arqentrada>. Transforma um grafo lido através do parâmetro <arqentrada> em um modelo linear que é armazenado em um arquivo com o mesmo caminho do <arqentrada> alterando-se somente a extensão para ".lp".

Resolve: chamado através do comando "r" <arqentrada>. Conforme veremos no capítulo seguinte, utilizamos a técnica de particionar o grafo para resolvê-lo. Este módulo particiona o grafo, gera o modelo linear para cada pedaço em separado, interage com o CPLEX para resolver o modelo linear, e verifica a coloração gerada pelo CPLEX para determinar se é uma coloração válida.

4.5 Criando o modelo linear

Para criarmos o modelo linear, devemos gerar um arquivo contendo as variáveis, restrições e função objetivo determinadas pelo modelo linear apresentado na seção 4.3.

Este arquivo exige um formato específico pois será lido e resolvido pelo CPLEX. O formato do arquivo está apresentado na figura 4.3.

```
MINIMIZE ou MAXIMIZE
    Função a ser minimizada ou maximizada
SUBJECT TO
    Restrições
BINARIES
    Declaração de todas as variáveis binárias
END
```

Figura 4.3: Formato do arquivo - modelo linear

Conforme pode ser observado na figura, existem três seções principais que devem ser preenchidas. A primeira seção MINIMIZE deve ser preenchida com a função objetivo que deve ser minimizada. Esta seção também poderia ser MAXIMIZE, neste caso a função objetivo seria maximizada. A segunda seção SUBJECT TO, deve ser preenchida com todas as restrições que existem no modelo. E por fim, na terceira seção, BINARIES, devem ser declaradas todas as variáveis, que no caso do nosso modelo linear, são binárias.

Ilustraremos como é o processo de geração de grafos e da criação do modelo linear.

Considere o grafo da figura 4.4.

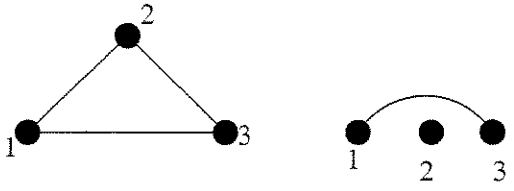

Figura 4.4: Grafo ilustrativo

A figura ilustra um grafo indiferença de 3 vértices e ao lado os vértices estão colocados em uma ordem indiferença e só estão representadas as arestas maximais.

O arquivo gerado que representa este grafo está ilustrado na figura 4.5.

```
$ // delimitador
3 // número de vértices
3 // número de arestas
2 // grau máximo
1;2;3 // vértices que possuem grau máximo
1-3 // aresta maximal
0-0 // delimitador
$ // delimitador
```

Figura 4.5: Representação do K_3

O arquivo com o modelo linear gerado está representado na figura 4.6. Neste modelo, as variáveis x_i representam "a cor i está sendo usada?", para $1 \leq i \leq \Delta + 1$, e as

variáveis $x_i v_j c_k$ representam "a aresta ij está pintada com a cor k?", para $1 \leq i, j \leq n$ e $1 \leq k \leq \Delta + 1$.

Observe pela figura que para um grafo pequeno como este, o modelo linear gerado é bem extenso. Para grafos grandes os arquivos de modelos lineares gerados tornaram-se gigantescos. Isto é decorrente da complexidade de nosso modelo linear.

Pode-se observar na figura que existem todas as restrições necessárias para que a coloração gerada seja uma coloração válida e ótima.

```
Minimize
x1 +x2 +x3
Subject To
 PINTAR
x2v3c1 + x2v3c2 + x2v3c3 = 1
x1v2c1 + x1v2c2 + x1v2c3 = 1
x1v3c1 + x1v3c2 + x1v3c3 = 1
\COMPLITO
x1v3c1 + x2v3c1 <= 1
x1v3c2 + x2v3c2 <= 1
x1v3c3 + x2v3c3 <= 1
x1v2c1 + x2v3c1 <= 1
x1v2c2 + x2v3c2 <= 1
x1v2c3 + x2v3c3 <= 1
x1v2c1 + x1v3c1 <= 1
x1v2c2 + x1v3c2 <= 1
x1v2c3 + x1v3c3 <= 1
\USO
x2v3c1 - x1 <= 0
x2v3c2 - x2 <= 0
x2v3c3 - x3 <= 0
x1v2c1 - x1 <= 0
x1v2c2 - x2 <= 0
x1v2c3 - x3 <= 0
x1v3c1 - x1 <= 0
x1v3c2 - x2 <= 0
x1v3c3 - x3 <= 0
BINARIES
x1       x2
x3       x2v3c1
x2v3c2   x2v3c3
x1v2c1   x1v2c2
x1v2c3   x1v3c1
x1v3c2   x1v3c3
END
```

Figura 4.6: Modelo linear gerado para o K_3

Capítulo 5

Os modelos lineares e seus resultados

Neste capítulo estaremos apresentando os testes realizados. Através da geração de grafos e do modelo linear criado, buscamos realizar vários testes que nos permitem extrair informações para tentarmos encontrar uma solução para o problema da coloração de arestas para os grafos indiferença.

5.1 A primeira seqüencia de testes

O objetivo deste teste era obter uma impressão geral sobre o assunto.

5.1.1 Contexto do teste

Foram gerados, randomicamente, 96 grafos com o número de vértices fixo de 5 a 100 e o número de arestas fixo em três vezes o número de vértices, exceto para $n = 5, 6$ quando $3n > \frac{n(n-1)}{2}$. Os grafos foram referenciados pelo seu número de vértices.

Gerados os grafos, foram gerados os modelos lineares para cada grafo. Após gerados, os modelos foram passados ao CPLEX para a sua resolução. Foi dado como tempo limite 500 segundos por modelo linear. O CPLEX ultrapassou o tempo limite para alguns grafos e nestes casos não obtivemos a função objetivo. Esta limitação de tempo foi necessária porque em alguns grafos o CPLEX gastava tempo demais para resolver, inviabilizando a coleta dos tempos de todos os grafos.

Após a resolução dos modelos, verificamos a coloração gerada pelo CPLEX para todos os grafos que não ultrapassaram o tempo limite e, portanto, tiveram uma resolução.

Foram realizadas quatro instâncias deste teste, sempre seguindo os mesmos passos: gerar os grafos randomicamente; gerar os modelos lineares; passar ao CPLEX e verificar a coloração. É valido ressaltar que os grafos com o mesmo número de vértices e arestas em diferentes instâncias não são necessariamente iguais, já que os mesmos foram gerados randomicamente. As instâncias foram denominadas Teste1, Teste2, Teste3 e Teste4.

Também é válido ressaltar que em nenhuma dessas instâncias foi aplicado algum tipo de filtro aos grafos gerados. Foram gerados grafos ignorando propriedades como Δ, grafos overfull e neighborhood overfull que como sabemos podem determinar o valor de χ'.

Os dados resultantes destes testes estão expostos nas tabelas 5.1 e 5.2 nas páginas 50 e 51. As colunas das tabelas são, nesta ordem, N(vértices), M(arestas), para cada uma das quatro instâncias do teste Δ, χ' e Tempo (segundos). Alguns valores de Δ na tabela estão representados por "-", isso ocorre porque nestes casos o CPLEX ultrapassou o tempo máximo permitido (500s) e não encontrou a função objetivo, que é o nosso χ'.

A figura 5.1 na página 52 apresenta um gráfico relacionando a quantidade de arestas do grafo em função do tempo gasto para sua resolução. O gráfico representa os dados da instância 1 do teste. Para uma melhor visualização do grafo, o eixo Y foi limitado ao valor máximo de 200, os tempos que ultrapassaram este valor não são visíveis no gráfico.

5.1.2 Resultados e Conclusões

Todos os grafos, que não extrapolaram o limite de tempo, eram *Classe* 1. Verificamos que a grande maioria dos grafos que extrapolou o limite de tempo eram grafos overfull ou neighborhood overfull e, portanto, pertencem à *Classe* 2.

O tempo que o CPLEX leva para resolver cada modelo não depende diretamente de Δ, quantidade de vértices ou arestas. Verificamos que existem muitos grafos que saem do padrão de tempo e gastam um tempo exageradamente superior a outros grafos com o mesmo número de vértices e aresta.

Concluímos que para obter resultados mais significativos é preciso restringir os grafos analisados.

		Teste1			Teste2			Teste3			Teste4		
N	M	Δ	χ'	T	Δ	χ'	T	Δ	χ'	T	Δ	χ'	T
5	10	4	5	0.14	4	5	0.15	4	5	0.12	4	5	0.12
6	15	5	5	0.03	5	5	0.03	5	5	0.03	5	5	0.04
7	21	6	7	1.02	6	7	1.02	6	7	1.04	6	7	1.04
8	24	7	7	0.12	7	7	0.13	7	7	0.15	7	7	0.11
9	28	8	8	0.17	7	7	8.41	8	8	0.23	8	8	0.15
10	30	9	9	0.22	9	9	0.16	9	9	0.22	8	8	0.16
11	34	10	10	0.19	10	10	0.22	9	9	0.2	8	8	0.34
12	36	9	9	0.16	8	8	0.21	10	10	0.2	10	10	0.12
13	39	11	11	0.2	9	9	0.29	10	10	0.2	9	9	0.44
14	42	9	9	0.31	8	8	2.85	9	9	0.25	10	10	0.27
15	45	9	9	0.27	8	8	0.24	9	9	0.28	11	11	0.39
16	48	8	8	0.24	8	8	0.27	8	8	0.35	10	10	0.29
17	51	8	8	7.88	9	9	0.45	8	8	0.27	10	10	0.43
18	54	8	-	500.06	9	9	0.51	8	8	0.4	9	9	0.45
19	57	9	9	0.59	9	9	3.14	8	8	0.42	9	9	0.56
20	60	8	-	500.07	9	9	0.44	8	8	0.41	9	9	0.46
21	63	10	10	0.5	9	9	0.52	8	8	0.73	10	10	0.42
22	66	10	10	0.64	9	9	0.45	8	8	0.62	10	10	0.38
23	69	10	10	0.63	9	9	0.55	8	8	8.06	8	8	1.22
24	72	10	10	0.51	9	9	5.33	8	-	500.04	8	8	0.73
25	75	11	11	0.55	10	10	14.3	8	-	500.08	9	9	0.91
26	78	11	11	0.76	10	10	17.96	9	9	7.22	9	9	0.98
27	81	11	11	0.82	10	-	500.18	10	10	0.9	9	9	1.33
28	84	11	11	1.27	10	-	500.08	9	9	1.87	10	10	0.98
29	87	11	11	0.66	11	11	0.81	10	10	0.79	10	10	1.02
30	90	11	11	1.08	11	11	0.96	10	10	1.94	11	11	0.88
31	93	11	11	0.71	11	11	1.47	10	-	500.17	10	10	0.88
32	96	10	10	0.99	11	11	2.04	10	-	500.18	10	10	1.66
33	99	10	-	500.16	11	11	1.3	11	11	32.52	10	10	1.82
34	102	11	11	23.13	11	11	2.03	11	11	62.55	10	10	27.47
35	105	11	11	30.75	11	11	1.18	11	11	6.78	11	11	44.44
36	108	11	11	3.79	12	12	1.8	12	12	5.94	11	11	42.06
37	111	11	11	2.21	12	12	1.25	11	11	1.08	12	12	1.43
38	114	11	11	34.95	12	12	2.08	11	11	1.06	11	11	1.56
39	117	11	11	38.06	10	10	2.41	11	11	2.11	11	11	47.35
40	120	11	11	2.47	12	12	1.51	11	11	18.5	11	11	33.32
41	123	11	11	2.08	12	12	1.91	11	11	1.98	11	11	1.48
42	126	12	12	3.13	13	13	2.34	12	12	39.45	13	13	1.51
43	129	12	12	2.29	13	13	1.71	11	11	3.18	12	12	1.84
44	132	12	12	2.08	13	13	2.43	11	11	7.79	11	11	57.28
45	135	12	12	1.66	11	11	74.88	11	11	51.78	11	11	33.26
46	138	12	12	3.73	11	11	41.63	11	11	50.72	11	11	1.93
47	141	14	14	1.48	11	11	109.43	11	11	66.07	11	11	101.06
48	144	14	14	1.91	12	12	3.79	12	-	500.24	12	12	8.76
49	147	14	14	3.02	12	12	2.18	11	11	6.68	12	12	83.84

Tabela 5.1: Resultado dos primeiros testes - parte 1

		Teste1			Teste2			Teste3			Teste4		
N	M	Δ	χ'	T	Δ	χ'	T	Δ	χ'	T	Δ	χ'	T
50	150	14	14	1.91	12	12	3.89	13	13	56.22	11	11	6.92
51	153	11	11	95.1	12	12	2.99	11	11	80.07	13	13	3.17
52	156	14	14	5.68	13	13	2.77	13	13	16.72	12	12	5.7
53	159	12	12	6.55	13	13	8.45	13	13	3.52	12	12	7.44
54	162	13	13	192.86	13	13	6.74	14	14	2.17	13	13	182.83
55	165	12	-	500.39	13	13	87.03	14	14	3.9	14	14	5.47
56	168	13	13	135.97	13	13	15.7	14	14	3.88	12	12	8.4
57	171	14	14	6.28	13	13	183.36	13	13	4.08	13	13	7.7
58	174	12	12	125.19	14	14	88.97	13	13	111.33	12	-	500.38
59	177	12	-	500.34	13	13	15.42	13	13	85.47	13	13	12.16
60	180	13	13	201.71	12	12	10.59	13	13	5.63	14	14	7.93
61	183	13	13	11.46	12	12	133.68	12	12	172.82	14	14	4.82
62	186	13	13	269.06	12	-	500.42	13	13	9.11	15	15	3.54
63	189	13	13	149.28	12	-	500.54	13	13	204.46	13	13	5.59
64	192	13	13	15.19	12	-	500.37	13	13	78.89	13	13	6.31
65	195	13	13	16.11	12	-	500.44	14	14	118.11	14	14	7.41
66	198	13	13	110.56	12	-	500.47	14	14	85.3	14	14	5.44
67	201	13	13	102.62	13	13	6	14	14	8.44	14	14	12.39
68	204	15	15	7.55	14	14	10.21	14	14	17.22	13	13	175.94
69	207	15	15	249.95	15	15	11.26	13	13	260.6	14	14	8.01
70	210	15	15	429.11	15	15	78.06	14	14	111.8	14	14	109.06
71	213	15	15	475.9	14	-	500.51	14	14	11.66	15	15	4.44
72	216	15	15	206.84	14	14	28.65	14	14	12.92	15	15	17.14
73	219	15	15	10.97	14	-	500.47	14	14	7.44	13	13	16.76
74	222	15	15	7.9	14	14	127.42	14	14	109.77	14	-	500.48
75	225	15	15	185.83	14	14	17.45	14	14	6.19	14	14	194.62
76	228	15	15	18.97	14	14	139.92	14	14	17.25	15	15	23
77	231	15	15	82.4	14	14	13.55	14	14	234.62	14	14	14.22
78	234	15	15	8.73	15	15	7.5	14	14	182.56	16	16	7.34
79	237	17	17	18.58	15	15	6.72	15	15	131.13	15	15	20.47
80	240	17	17	14.95	15	15	19.26	16	16	5.48	14	14	20.1
81	243	17	17	108.5	14	14	33.75	16	16	17.39	15	15	27.33
82	246	14	14	239.73	14	-	500.6	16	16	8.8	15	15	10.58
83	249	15	15	466.3	15	15	418.54	15	15	18.43	14	14	392.16
84	252	15	15	451.64	14	-	500.71	15	15	14.29	15	-	500.9
85	255	14	14	334.86	15	15	454.44	17	17	18.97	17	17	23.45
86	258	15	15	182.16	15	15	223.46	17	17	7.83	17	17	229.63
87	261	15	15	24.25	17	17	32.45	17	17	7.67	16	16	16.77
88	264	16	16	10.32	17	17	22.7	17	17	178.69	15	15	40.2
89	267	16	16	15.16	16	16	21.29	16	16	11.05	15	-	500.64
90	270	16	16	39.25	16	16	21.65	16	16	43.97	16	16	155.37
91	273	15	15	28.14	16	16	30.01	16	16	9.21	16	16	39.37
92	276	15	15	331.69	15	15	394.67	16	16	22.03	15	15	381.51
93	279	15	15	305.72	18	18	260.53	16	16	42.88	16	16	28.33
94	282	15	15	45.18	14	-	500.89	17	17	201.84	16	16	30.65
95	285	15	15	38.75	16	16	406.51	17	17	19.87	17	-	500.78
96	288	16	16	51.19	16	-	500.95	17	17	17.87	16	16	23.82
97	291	16	16	34.3	18	18	31.75	18	18	15.44	16	16	69.79
98	294	16	16	42.83	18	18	208.68	18	18	21.13	18	18	9.64
99	297	16	16	53.73	18	18	20.46	16	16	11.01	18	18	30
100	300	15	15	70.89	18	18	8.07	16	16	280.75	18	18	26.32

Tabela 5.2: Resultado dos primeiros testes - parte 2

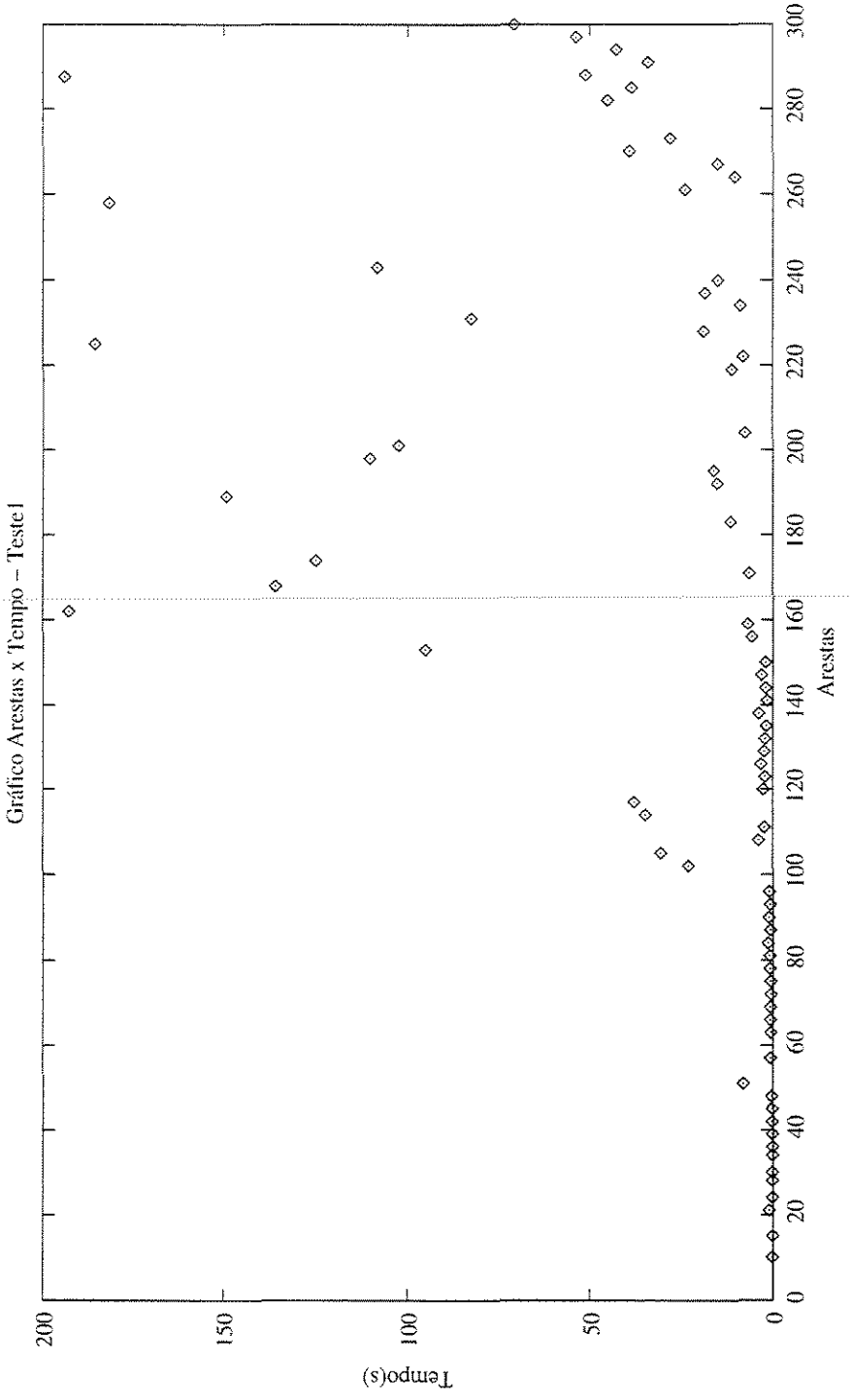

Figura 5.1: Gráfico: Arestas x Tempo - Primeiro Teste

5.2 A segunda seqüencia de testes

O objetivo principal destes testes era encontrar grafos indiferença que fossem *Classe* 2 e que não fossem nem overfull nem neighborhood overfull.

5.2.1 Contexto do teste

Foram gerados todos os grafos indiferença com 10 vértices independente do número de arestas. Como nosso objetivo era encontrar grafos *Classe* 2, filtramos os grafos eliminando todos aqueles que tivessem grau máximo ímpar, que fossem overfull ou mesmo neighborhood, já que é sabido que grafos de grau máximo ímpar pertencem à *Classe* 1, e grafos com as outras duas propriedades não nos interessam, pois são *Classe* 2.

O processo foi o seguinte: gerar todos os grafos com 10 vértices eliminando grafos de grau máximo ímpar, grafos overfull e grafos neighborhood overfull; gerar o modelo linear para estes grafos; passar estes modelos para o CPLEX e por fim verificar a coloração.

No total foram gerados 4862 grafos indiferença com 10 vértices, sendo que destes 2431 possuíam grau máximo ímpar e 93 eram neighborhood overfull, nenhum é overfull já que o número de vértices é par. Então foram efetivamente gerados os modelos lineares e resolvidos para 2338 grafos.

É valido ressaltar que o gerador de grafos não faz distinção entre grafos isomorfos devido a complexidade do algoritmo necessário para essa distinção. Com certeza o número de grafos indiferença com 10 vértices não isomorfos é bem inferior ao número dado no parágrafo anterior.

O gráfico resultante deste teste em Arestas x Tempo está representado no figura 5.2. Para gerar este gráfico foi feita um média entre todos os tempos gastos de todos os grafos que têm a mesma quantidade de arestas. Como foram gerados todos os grafos com 10 vértices, gerou-se vários grafos com 9, 11, 12, 13, ... 36 arestas. Foram gerados pontos onde X é quantidade de arestas e Y é a média de tempo gasto pelos grafos com a mesma quantidade de arestas.

5.2.2 Resultados e Conclusões

O objetivo maior deste teste era encontrar grafos *Classe* 2 com Δ par e que não fossem neighborhood overfull, porém não foi encontrado nenhum grafo nestas condições. A grande maioria dos grafos foi pintada em menos de meio segundo. Os grafos que gastaram mais tempo não passaram de 3 segundos.

Concluímos que a abordagem de modelar linearmente o problema é bastante útil na sua análise. Podemos analisar o comportamento do problema e tentar encontrar contra-exemplos ou idéias para uma prova.

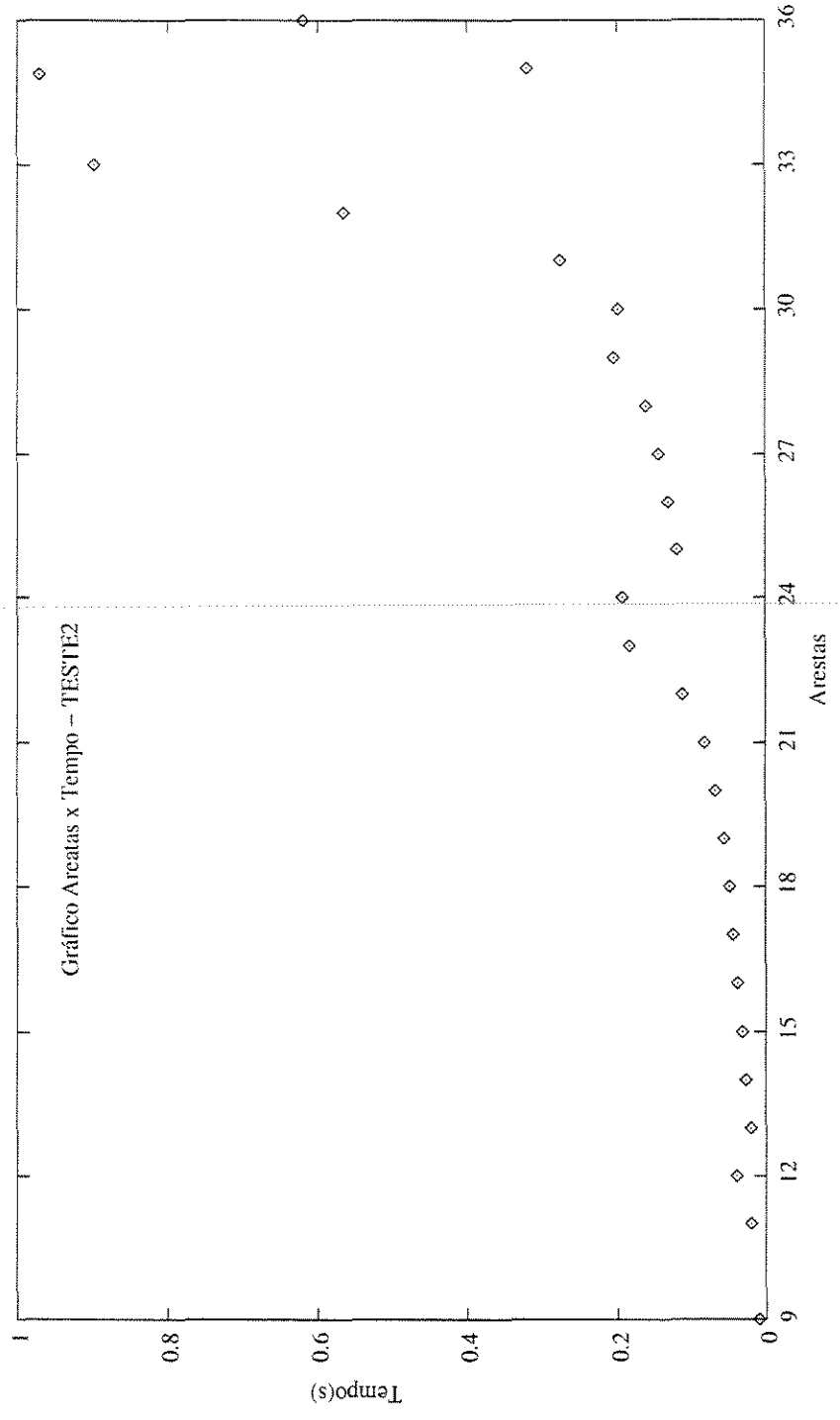

Figura 5.2: Gráfico: Arestas x Tempo - Segundo Teste

5.3 Decompondo grafos indiferença

A idéia inicial do projeto era criar novos modelos lineares e realizar testes com estes novos modelos, comparando-os e tentando definir o comportamento da complexidade do problema. Porém, no decorrer do projeto esbarramos na dificuldade de criar um novo modelo linear com custo computacional inferior ao modelo já proposto. Fizemos também uma extensa pesquisa bibliográfica a procura de modelos lineares para coloração de arestas e não conseguimos encontrar nenhum que tivesse um custo inferior ao proposto.

Além desta dificuldade, há o fato de que o número de grafos gerados cresce exponencialmente à medida que se aumenta o número de vértices, principalmente porque o nosso gerador de grafos não diferencia grafos isomorfos, criando então uma dificuldade extra de tempo e espaço para realização dos testes.

Nossa primeira bateria de testes nos mostrou que estávamos fazendo o computador trabalhar de forma errada, gastando grande parte do seu tempo em tarefas não importantes.

Observe um grafo indiferença típico com 30 vértices apresentado na figura 5.3. Este grafo está organizado em função da sua ordem indiferença e só são apresentadas as arestas maximais.

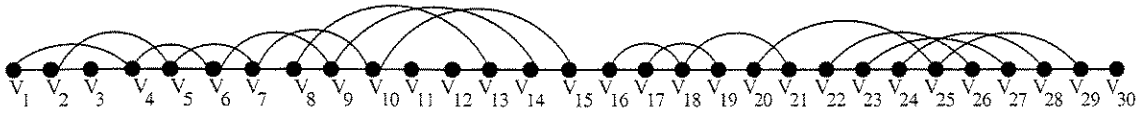

Figura 5.3: Grafo indiferença típico com 30 vértices

É muito simples pintar as regiões "finas" com Δ cores. O verdadeiro desafio está em pintar as regiões "grossas" e, é claro, se formos pensar em pintar por regiões, devemos ter uma maneira de harmonizar as pinturas em regiões que se tocam.

Conhecíamos os trabalhos de *Planthold* e colegas [31], que nos ensinam a pintar uma vizinhança de vértices de grau Δ, com Δ cores.

Partindo deste principio, resolvemos então decompor os grafos indiferença em vizinhanças e passar a estudar as conexões entre estes pedaços. De forma que, estaríamos

resolvendo o problema se conseguíssemos resolvê-lo para dois pedaços, já que todo o grafo indiferença estaria decompondo em pedaços e, conforme veremos seguir, estes pedaços só se intersectam com um único outro pedaço de cada lado.

A seguir, apresentaremos o algoritmo utilizado para decomposição.

Algoritmo da Decomposição

Entrada: Um grafo indiferença G com vértices 1,2,3,...n em ordem indiferença.

Saída: Decomposição de G em pedaços $P_i = N[U_i]$

Ache o vértice mais à direita que seja vizinho do primeiro vértice da ordem indiferença e denomine de U_1.

Considere a vizinhança de U_1 como sendo o primeiro pedaço, P_1.

$i \leftarrow 2$

Repita, até terminar o grafo

Ache o primeiro vértice não adjacente a U_{i-1} e denomine U_i.

Considere a vizinhança de U_i como sendo um novo pedaço, P_i.

$i \leftarrow i + 1$

Fim { Repita }

Apresentaremos agora um exemplo de decomposição aplicado pelo algoritmo acima, utilizando o grafo já apresentado na figura 5.3.

O primeiro passo do algoritmo é definir o vértice denominado U_1, que é definido como o vizinho mais à direita do primeiro vértice na ordem indiferença. Os demais vértices U são definidos como o primeiro vértice não adjacente ao vértice U anterior.

A figura 5.4 apresenta o grafo e todos os vértices U determinados.

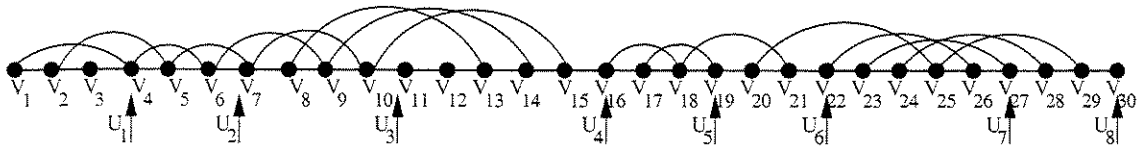

Figura 5.4: Definição dos vértices U_i. Os vértices U_i estão destacados por setas.

Cada pedaço é formado pela vizinhança dos vértices U. A figura 5.5 apresenta os pedaços decompostos em separado.

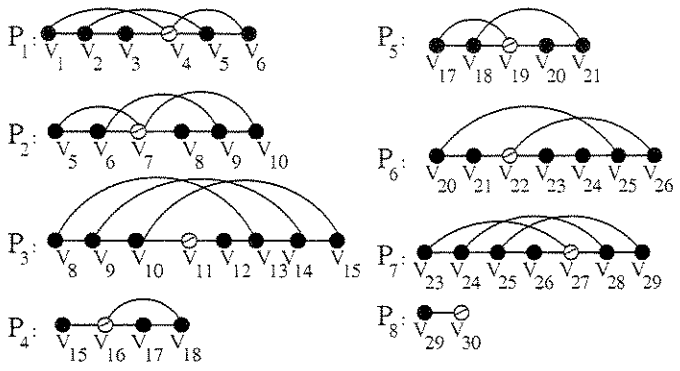

Figura 5.5: Grafo decomposto. Em cada pedaço P_i, o vértice U_i está mais claro.

Utilizando o algoritmo descrito acima, podemos enunciar o seguinte lema.

Lema 5.1 *Cada vértice pertence a no máximo dois pedaços, que devem ser necessariamente consecutivos.*

Prova:

Em primeiro lugar observamos que todos os U_i formam um conjunto independente, ou seja, não são adjacentes entre si.

Segue-se daí que cada U_i está em apenas um pedaço, o pedaço P_i.

Se um vértice v está em três pedaços, P_i, P_j e P_k, com $i < j < k$, então v é adjacente a U_i, U_j e U_k simultaneamente, formando um $K_{1,3}$ induzido. Contradição, pois o grafo é indiferença.

Se um vértice v está entre U_i e U_{i+1} então ele não pode ser adjacente a U_k para $k < i$ (caso contrário U_k e U_i seriam adjacentes) nem pode ser adjacente a U_l com $l > i + 1$ (caso contrário U_l e U_{i+1} seriam adjacentes). Conclui-se que v só pode estar nos pedaços P_i e P_{i+1}.

Raciocínios análogos valem para vértices antes de U_1 e depois de U_p, onde p é o número de pedaços. □

A partir do momento que resolvemos pintar pedaços de um grafo, precisamos de um método que permitisse que a união das colorações de cada pedaço formasse uma coloração válida para o grafo.

Percebemos que não poderíamos colorir um pedaço isoladamente, porque existem restrições de cores em determinados vértices que são determinadas pelo pedaço anterior. A figura 5.6 ilustra bem esta situação, nesta figura só estão representadas as arestas maximais.

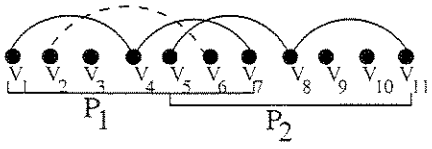

Figura 5.6: Restrições de cores em determinados vértices

Considere o vértice V_6 da figura. Este vértice pertence a P_1 e a P_2. Ao se colorir P_1, todas arestas (V_i, V_j) com $i, j \leq 7$ são coloridas. Na coloração de P_2 já haverá uma coloração parcial determinada pela coloração de P_1. Suponha que queiramos atribuir uma cor à aresta (V_6, V_8). Para determinar uma cor válida é necessário conhecer as cores incidentes no vértice V_6. Estas cores incidentes foram determinadas pela coloração de P_1, por arestas contidas unicamente em P_1 como, por exemplo, a aresta em destaque na figura por uma linha pontilhada.

Ao pintar um pedaço do grafo precisamos de informações da coloração do pedaço anterior. Poderíamos gerar e passar como parâmetro uma lista de cores já utilizadas para cada vértice em comum dos dois pedaços, porém preferimos sempre estar colorindo dois pedaços ou duas vizinhanças, sendo que um pedaço já vem pintado (coloração parcial) e deve-se simplesmente estender esta coloração para o restante do grafo.

A seguir apresentaremos uma conjectura criada para reger a forma como os pedaços devem ser coloridos. Para que a propriedade descrita acima seja respeitada, no final da coloração dos pedaços a união de todas as colorações deve ser uma coloração ótima para o grafo.

Definição 5.1 ***Grafo indiferença semi-universal.***

Um grafo indiferença H é dito semi-universal se existem dois vértices U_1 e U_2 satisfazendo:

1. *U_1 e U_2 não são adjacentes.*
2. *$N[U_1] \cup N[U_2] = V(H)$.*
3. *U_1 é adjacente ao vértice imediatamente anterior a U_2, numa ordem indiferença.*

Um vértice U_i com as propriedades acima é chamado de ***semi-universal.***

A figura 5.7 ilustra um grafo indiferença semi-universal. Neste grafo os vértices U_1 e U_2 são vértices semi-universais.

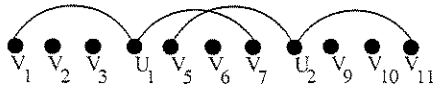

Figura 5.7: Exemplo de um grafo semi-universal.

Definição 5.2 *Uma "d-**Coloração Parcial**" de um grafo indiferença semi-universal é uma coloração válida de $N[U_1]$ com d cores, onde U_1 tem as propriedades mencionadas na definição 5.1.*

Conjectura 2 *Dada uma d-coloração parcial de um grafo indiferença semi-universal não neighborhood overfull H, com $d \leq \Delta(H)$, é possível estendê-la para uma $\Delta(H)$-coloração das arestas de H.*

O seguinte algoritmo pintaria grafos indiferença G não neighborhood overfull, com Δ cores:

Decomposição (Algoritmo apresentado acima)

Colorir $N[U_1] = P_1$ de acordo com Planthold.

Repita, para i de 2 até p (onde p é quantidade de pedaços).

Estenda a coloração parcial de $G[N[U_{i-1}]]$ para $G[N[U_{i-1}] \cup N[U_i]]$.

Fim { Repita }

Apresentaremos o funcionamento do algoritmo com um exemplo, considerando o grafo já apresentado na figura 5.3. A figura 5.8 apresenta este mesmo grafo com as seus pedaços destacados. Observe que a união de dois pedaços consecutivos forma um grafo indiferença semi-universal.

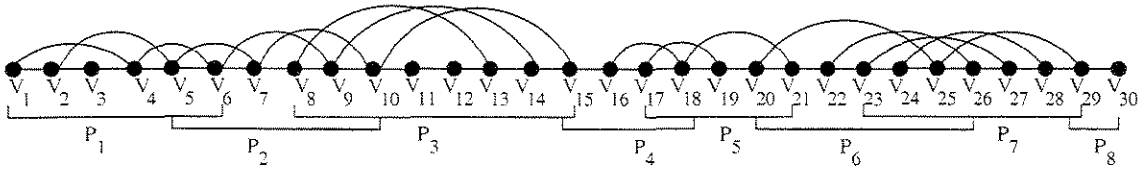

Figura 5.8: Grafo Indiferença - Pedaços destacados

O primeiro passo do algoritmo acima é decompor o grafo em pedaços utilizando o algoritmo de decomposição apresentado na seção anterior. Então, para o nosso exemplo, são válidas os mesmos pedaços já apresentados na figura 5.5.

Após decompor o grafo, deve-se colorir o primeiro pedaço utilizando o método proposto por *Planthold* [31] onde todo grafo com um vértice universal é colorível com Δ cores.

A figura 5.9 apresenta o grafo já com o primeiro pedaço pintado. Na figura, as arestas já pintadas estão destacadas com uma linha pontilhada. Lembrando que nesta figura só estão sendo representadas as arestas maximais.

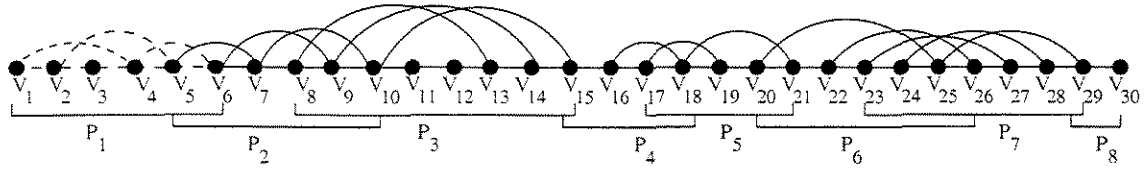

Figura 5.9: Grafo Indiferença - Pintando o primeiro pedaço

A coloração será estendida consecutivamente a cada pedaço até que todo o grafo esteja pintado. Pode-se observar que ao se colorir um pedaço, automaticamente algumas arestas

do pedaço posterior também são coloridas. Estas arestas são justamente as arestas que pertencem a intersecção de dois pedaços.

A figura 5.10 apresenta o grafo já com o segundo pedaço pintado. Observe na figura que as arestas que formam a coloração parcial para o P_3 estão destacadas por setas. O que deve ser feito para o pedaço P_3 é estender a coloração respeitando o limite de Δ cores.

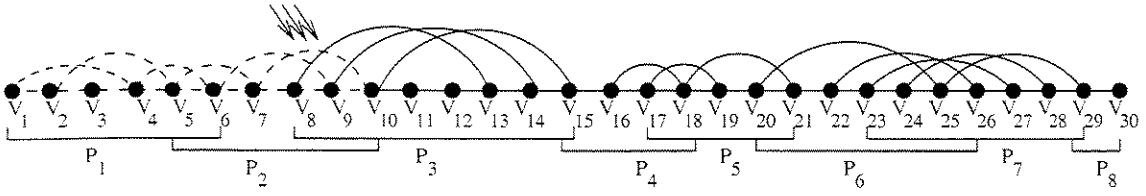

Figura 5.10: Grafo Indiferença - Pintando os demais pedaços

Este processo se repete até que o último pedaço seja pintado.

Com esta conjectura e um algoritmo para decompor e pintar os grafos, resolvemos realizar alguns testes utilizando o CPLEX para verificarmos se realmente a coloração gerada era válida e ótima.

Os testes foram realizados da seguinte forma:

- Foi necessário adicionar uma nova opção que permitisse a geração de todos os grafos indiferença com n vértices e que possuíam um vértice universal. Neste teste consideramos que o primeiro pedaço sempre seria uma clique, ou seja, U_1 sempre seria o primeiro vértice do grafo.

- Geramos todos os grafos indiferença com 15 vértices que possuem um vértice universal.

- Alteramos o nosso programa gerador de grafos, adicionando uma nova opção que decomponha o grafo gerando um modelo linear para o grafo $G[N[U_1]]$. Geramos este modelo linear parcial para todos os grafos gerados.

- Passamos os modelos lineares dos grafos parciais para o CPLEX resolver, realizando assim uma "coloração parcial".

- Também foi necessário adicionar uma nova opção que permitisse a geração de um modelo linear para um grafo inserindo neste modelo valores para algumas variáveis que eram extraídos do arquivo de saída do CPLEX, gerado na resolução dos modelos parciais. Portanto geramos modelos lineares, inserindo neste modelos as cores que o CPLEX já tinha usado para pintar o pedaço anterior. Ou seja, estaríamos agora pintando o grafo completo que já possui uma parte pintada, feita pela "coloração parcial".

- Passamos os modelos lineares dos grafos gerados para o CPLEX resolver.

Porém logo nos primeiros testes, verificamos que a conjectura 2 é falsa. Podemos provar utilizando o seguinte contra-exemplo.

Considere o grafo indiferença semi-universal H com nove vértices mostrado na figura 5.11. Este grafo não é neighborhood overfull, possui $\Delta = 6$ e dois vértices semi-universais (U_1 e U_2). Os Δ-vértices do grafo são os vértices 2, 5, 6 e 7. Pela conjectura 2 este grafo pode ser pintado com 6 cores, porém vamos apresentar uma coloração parcial para o grafo que impossibilita a extensão da coloração para o grafo utilizando 6 cores.

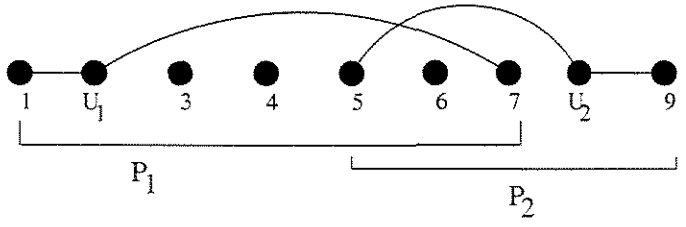

Figura 5.11: Grafo contra-exemplo para a conjectura 2

O grafo possui dois pedaços $N[U_1]$ e $N[U_2]$, destacados na figura. Suponha que o algoritmo de coloração parcial pinte o primeiro pedaço com uma coloração com $d = 6$ cores. Nesta coloração suponha que sejam utilizadas as cores 1, 2, 3, 4 e 5 para pintar a clique $H[U_1..7]$ e a cor 6 para pintar a aresta $\{1, U_1\}$. A figura 5.12 apresenta o primeiro pedaço do grafo com a clique $H[U_1..7]$ destacada.

Com esta coloração parcial todos os vértices da clique $H[U_1..7]$, exceto U_1, possuem apenas a cor 6 faltante. Os vértices 5, 6 e 7 pertencem à vizinhança de U_1 e de U_2, portanto

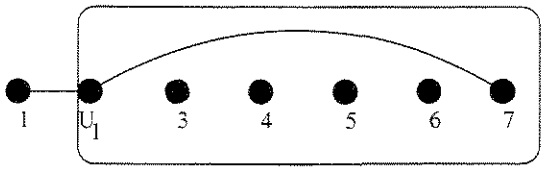

Figura 5.12: Grafo contra-exemplo - Coloração de $N[U_1]$.

ao se estender a coloração para todo o grafo, as cores representadas nestes vértices irão influenciar a coloração de $N[U_2]$. Observe que as cores 1, 2, 3, 4 e 5 estão representadas nestes três vértices.

Ao tentarmos estender a coloração para todo o grafo, teremos a situação ilustrada na figura 5.13. Apenas mais quatro arestas devem ser pintadas.

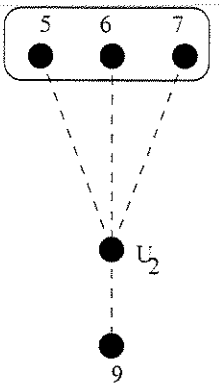

Figura 5.13: Grafo contra-exemplo - Extensão da coloração.

Para realizarmos uma coloração válida para este grafo, não temos outra opção a não ser atribuir a cada aresta saindo do vértice U_2 uma cor diferente. Porém, os vértices 5, 6 e 7 já possuem as cores 1, 2, 3, 4 e 5 representadas neles, portanto precisamos de mais três cores (uma cor diferente para cada aresta). Com esta coloração parcial de $N[U_1]$ este grafo será pintado com 8 cores, contrariando a conjectura 2.

Chegamos a conclusão de que deve haver algumas restrições na "coloração parcial", de forma que a coloração seja feita com todas as cores disponíveis deixando algumas cores faltantes nos vértices. Com estas observações elaboramos uma nova conjectura que será

apresentada na próxima seção.

5.4 Ajustando a conjectura

Utilizando uma coloração qualquer para cada pedaço não conseguimos utilizar Δ cores para colorir um grafo porque pode ocorrer que um vértice passe a não ter muitas cores faltantes, sendo que aquele vértice ainda possui arestas não pintadas em outro pedaço.

Um grafo indiferença *Classe* 1 com $\Delta = x$ possui x cores para serem distribuídas por suas arestas. Pensando em um pedaço P_i de um grafo qualquer, $\Delta(P_i)$ pode ser bem inferior a x. Se colorirmos um pedaço P_i de G utilizando somente $\Delta(P_i)$ cores poderão ocorrer situações onde não haverá cores faltantes suficientes para determinados vértices.

Precisamos restringir de alguma forma a coloração realizada de modo que possibilite a coloração das arestas do próximo pedaço.

Detectado este problema, resolvemos colorir cada pedaço utilizando $\Delta(G)$ cores independente de quanto é $\Delta(P_i)$. Desta forma estamos distribuindo as cores mais uniformemente pelo grafo. E é claro, se conseguirmos colorir todos os pedaços do grafo com $\Delta(G)$ estaremos pintando o grafo com $\Delta(G)$ cores. Resolvemos utilizar coloração balanceada para garantir que todas as $\Delta(G)$ seriam utilizadas uniformemente.

Definimos uma nova conjectura que restringe que a coloração parcial e a extensão da coloração sejam feitas de forma balanceada.

Definição 5.3 *Uma coloração é dita* ***balanceada*** *se a diferença entre o número de arestas pintadas por duas cores i e j é no máximo uma unidade.*

Conjectura 3 *Dada uma d-coloração parcial balanceada de $N[U_1]$ de um grafo indiferença semi-universal não neighborhood overfull H, com $d \geq \Delta(H)$, é possível estendê-la para uma d-coloração das arestas de H, balanceada em $N[U_2]$.*

Desta forma, todo pedaço será colorido utilizando no máximo $\Delta(G)$ cores, já que restringimos a coloração parcial e a extensão da coloração a sempre utilizarem as $\Delta(G)$ cores disponíveis de forma balanceada.

Com esta nova conjectura elaboramos novos testes que pudessem validar o nosso método. Tivemos que criar um novo modelo linear que além das restrições necessárias para a coloração também restringisse que a coloração fosse feita de forma balanceada.

Adicionamos ao modelo linear apresentado no capítulo 4, seção 4.3 a seguinte restrição que garante que a coloração seja feita de forma balanceada.

(Balancear) - Restringe que a coloração seja balanceada.

$$\left| \sum_{e \in E} x_{ec} - \sum_{e \in E} x_{ek} \right| \leq 1 \ , \forall \ c, k \in \text{Cores}$$

onde E é o conjunto de arestas que se deseja balancear.

Na próxima seção apresentaremos os resultados dos testes realizados com esta nova conjectura e este novo modelo linear.

5.4.1 Seqüencia de testes

Para realização destes testes foi utilizada a seguinte metodologia:

- Pegar o primeiro pedaço do grafo;
- Gerar o modelo linear para o primeiro pedaço, colocando as restrições de balanceamento no modelo;
- A partir do segundo pedaço, sempre gerar o modelo linear para 2 pedaços consecutivos, inserindo no modelo as restrições de arestas já coloridas no primeiro anterior e as restrições de balanceamento no segundo pedaço;
- Ao término de todos os pedaços, gerar e verificar a coloração do grafo.

Realizamos diferentes baterias de testes, entre elas:

- Todos os grafos indiferença com 10 vértices;
- Todos os grafos indiferença com 15 vértices;
- Todos os grafos indiferença com 20 vértices;

Como resultado de todos estes testes, conforme será demonstrado a seguir, encontramos alguns grafos que não são neighborhood overfull e não puderam ser coloridos com Δ cores. Esta condição de extensão da coloração balanceada no pedaço posterior não permite a coloração com Δ cores em alguns grafos que são Δ coloríveis.

Quanto ao tempo de execução e complexidade do problema, analisamos os tempos tomados por cada pedaço de grafo. Observamos que existem três fatores que influenciam no tempo de execução. São eles:

- Número de arestas a serem pintadas no pedaço;
- Número de arestas onde um dos extremos recebe arestas de outro pedaço, existindo restrições nas cores que podem ser utilizadas;
- Folga no número de cores disponíveis no pedaço (diferença entre $\Delta(G)$ e $\Delta(P)$), onde P é o pedaço.

Apresentaremos um contra-exemplo que mostra que a conjectura 3 também é falsa. O problema está nos vértices que pertencem à intersecção de pedaços. Estes vértices necessariamente devem ter cores faltantes suficientes para permitir a coloração de suas arestas no próximo pedaço com Δ cores. O fato da coloração do pedaço ser balanceada não implica que a coloração das arestas incidentes nestes vértices será balanceada, pois é possível que a coloração seja balanceada para o todo o pedaço mas não seja em partes do pedaço.

Apresentaremos um contra-exemplo que prova que a conjectura 3 também é falsa.

Considere o grafo indiferença semi-universal com sete vértices mostrado na figura 5.14. Este grafo não é neighborhood overfull, possui $\Delta = 4$ e dois vértices semi-universais (U_1 e U_2). Pela conjectura 3 este grafo pode ser pintado com 4 cores.

Suponha que a coloração parcial de $N[U_1]$ seja a coloração apresentada na figura 5.15.

Observe que esta coloração respeita as restrições de balanceamento já que as cores 1, 2 e 3 foram usadas por 2 arestas enquanto a cor 4 foi usada por uma aresta. Com esta coloração parcial os vértices 4 e 5 possuem apenas a cor 4 faltante.

Ao tentarmos estender a coloração para todo o grafo, teremos a situação ilustrada na figura 5.16. Apenas mais três arestas devem ser pintadas.

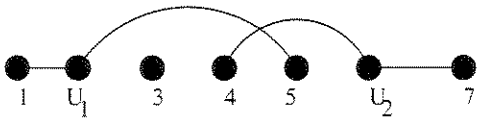

Figura 5.14: Grafo contra-exemplo para a conjectura 3

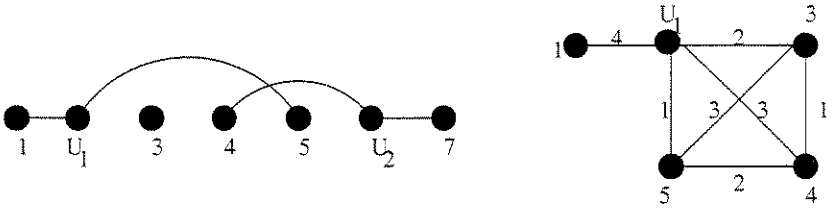

Figura 5.15: Grafo contra-exemplo - Coloração de $N[U_1]$ - Conjectura 3.

Necessariamente todas as arestas saindo do vértice U_2 devem receber uma cor diferente. Porém, os vértices 4 e 5 já possuem as cores 1, 2 e 3 representadas neles e, portanto, precisamos de mais duas cores (uma cor diferente para cada aresta). Com esta coloração parcial de $N[U_1]$ este grafo será pintado com 5 cores, contrariando a conjectura 3.

Diante deste problema, decidimos tentar uma nova abordagem para coloração parcial. Faz-se necessário uma nova definição.

Definição 5.4 *Um vértice é dito* ***bi-pedaço*** *se pertence a mais de um pedaço. Vértices bi-pedaços são os vértices que estão na intersecção de dois pedaços.*

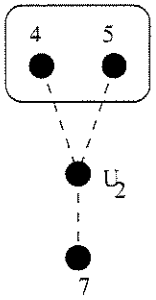

Figura 5.16: Grafo contra-exemplo - Extensão da coloração - Conjectura 3.

Um determinado pedaço pode possuir vértices bi-pedaços que intersectam com o pedaço anterior, denominados ***vértices bi-pedaços anteriores****, ou que intersectam com o pedaço posterior, denominados* ***vértices bi-pedaços posteriores****. Definimos como* ***arestas perigosas*** *as arestas incidentes em vértices bi-pedaços posteriores de um pedaço qualquer.*

Percebemos que não é suficiente balancear todo o pedaço, pois conforme já mencionado pode haver conjuntos de arestas que não são balanceados dentro do pedaço. As arestas que geram problemas são aquelas que tocam os vértices que também pertencem ao pedaço posterior (vértices bi-pedaços posteriores), pois serão as cores destas arestas que irão determinar as cores das arestas do pedaço posterior incidentes nestes vértices. O grande problema então está nos vértices bi-pedaços, já que somente as arestas incidentes nestes vértices irão influenciar no pedaço posterior. As demais arestas do pedaço não têm influência na coloração do pedaço posterior.

Observe na figura 5.17, um grafo com dois pedaços representado em uma ordem indiferença. Os vértices bi-pedaços estão destacados. Na coloração do primeiro pedaço deste grafo somente as arestas destacadas (arestas perigosas) terão influência na coloração do segundo pedaço. Estas arestas determinam as cores faltantes nos vértices bi-pedaços posteriores do pedaço P_1.

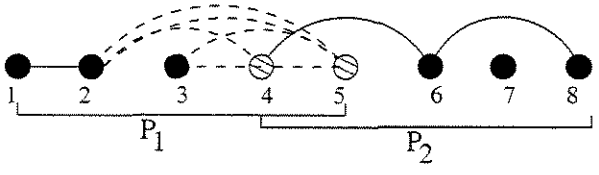

Figura 5.17: Ilustração das arestas que influenciam na coloração do pedaço posterior.

Com esta observação tentamos adotar uma nova abordagem para a coloração parcial. Ao invés de exigirmos o balanceamento de todo o pedaço passamos a exigir em cada pedaço do grafo o balanceamento apenas das arestas que tocam vértices bi-pedaços posteriores (arestas perigosas). Como estamos interessados em dar condições para coloração do pedaço posterior, em cada pedaço só nos interessam os vértices bi-pedaços poste-

riores. Desta forma estaríamos "relaxando" as cores faltantes dos vértices bi-pedaços. Acreditávamos que seria possível estender a coloração de um pedaço, respeitando esta condição de balanceamento somente das arestas perigosas.

Porém ao fazermos um teste prático desta nova abordagem descobrimos alguns grafos onde não é possível respeitar esta nova condição de balanceamento. Apresentaremos um grafo exemplificando esta situação.

Considere o grafo apresentado na figura 5.18. Este grafo possui 10 vértices, Δ=4, todos os vértices bi-pedaços estão destacados na figura, assim como as arestas perigosas.

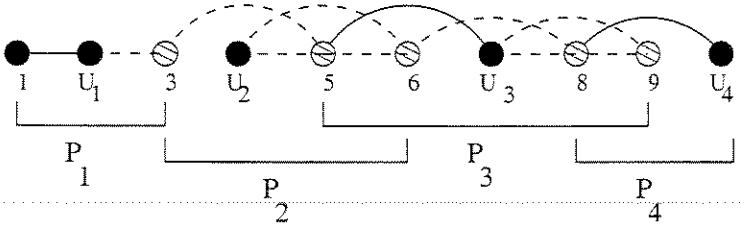

Figura 5.18: Grafo exemplo, vértices bi-pedaços destacados

Observe que no pedaço P_2 existem quatro arestas perigosas, assim como no pedaço P_3. Sendo $\Delta(G) = 4$, para respeitar a nova condição de balanceamento cada uma destas arestas deve receber uma cor diferente, caso contrário uma cor não seria utilizada enquanto outras seriam utilizadas duas ou mais vezes, desrespeitando o balanceamento que diz que a diferença de utilização das cores é no máximo 1.

O grafo da figura 5.19 ilustra melhor cada pedaço. Nesta figura estão representadas todas as arestas do grafo.

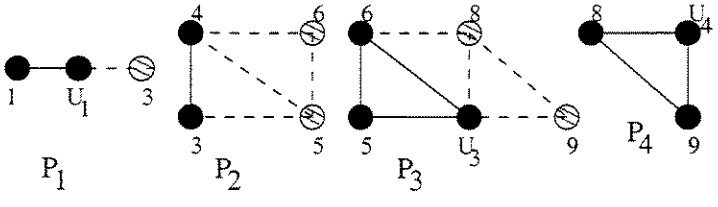

Figura 5.19: Grafo com todas as suas arestas

Vamos mostrar que no pedaço P_3 não é possível atribuir a cada aresta perigosa uma cor diferente. Considere que os pedaços P_1 e P_2 já foram pintados e que agora deve-se pintar o pedaço P_3. Como já mencionado, neste caso de 4 cores e 4 arestas perigosas cada aresta perigosa deve receber uma cor diferente. Considere que as cores são a, b, c e d. Como no pedaço P_2 cada uma das arestas perigosas recebeu uma cor diferente, podemos dizer que as cores representadas no vértice 5 são a, b e c e no vértice 6 são b e d. A única cor possível para a aresta $\{5, U_3\}$ é a cor d. Observe que no vértice 6 as cores faltantes são a e c. O vértice U_3 possui quatro arestas, obviamente cada uma destas arestas deverá receber uma cor diferente, sendo que a aresta $\{5, U_3\}$ obrigatoriamente já recebeu a cor d. A aresta $\{6, U_3\}$ usará uma das duas cores faltantes em 6, restando apenas uma cor faltante no vértice 6 que chamaremos de x ($x = a$ ou $x = c$). Esta cor x obrigatoriamente será utilizada na aresta $\{6, 8\}$ (já que é a única cor faltante no vértice 6). Com isso, temos em U_3 apenas duas cores faltantes, sendo x uma delas. Como x já foi usado em uma aresta perigosa ($\{6,8\}$), e as duas arestas não pintadas incidentes em U_3 são arestas perigosas, a cor x será utilizada 2 vezes, não respeitando o balanceamento das arestas perigosas.

5.4.2 Colorindo grafos indiferença

Também realizamos testes com uma abordagem um pouco diferente da apresentada na conjectura 3. Na conjectura 3, ao pintar um grafo indiferença semi-universal, é recebido como coloração parcial, a coloração balanceada de $N[U_1]$ que deve ser estendida para uma coloração balanceada de $N[U_2]$. Nesta nova abordagem, ao invés de restringirmos que a coloração de $N[U_2]$ fosse balanceada, restringimos que a coloração do grafo indiferença semi-universal, $N[U_1] \cup N[U_2]$, fosse balanceada.

Foram realizados testes com os seguintes grafos:

- Todos os grafos indiferença com 10, 15 e 20 vértices;
- Grafos indiferença randômicos com 10, 20, 30, 40 e 50 vértices;
- Grafos indiferença randômicos com Δ variando entre 5 e 200.

Apesar do mesmo contra-exemplo apresentado para a conjectura 3 ser válido para esta abordagem, na prática o CPLEX conseguiu realizar uma Δ-coloração para todos os grafos não neighborhood overfull, respeitando a condição de que a coloração da união de 2 pedaços consecutivos seja balanceada.

Aparentemente esta abordagem permite um pouco mais de liberdade na escolha das cores das arestas incidentes nos vértices bi-pedaços. Observamos que o CPLEX, mesmo não havendo esta restrição, distribuiu as cores nas arestas perigosas, de forma que torna possível a coloração do pedaço posterior.

5.4.3 Contribuição

Implementamos, deixando como contribuição, uma ferramenta que recebe como entrada um arquivo com grafos indiferença e gera como saída uma coloração ótima para estes grafos.

Esta ferramenta recebe os grafos indiferença, verifica se são grafos válidos, gera um modelo linear para o grafo (modelo proposto na seção 4.3), interage com o CPLEX para resolução do modelo e retorna a coloração gerada pelo CPLEX.

Deve ser passado à ferramenta um arquivo, no formato texto, contendo os grafos indiferença que serão coloridos. Este arquivo é denominado *Graph File*. O formato deste arquivo está ilustrado na figura 5.20. Pode haver mais de um grafo representado em um mesmo *Graph File*.

A ferramenta pode ser utilizada através da seguinte URL:
www.ic.unicamp.br/~ra006989/indifference.

$	delimitador, obrigatório
nome	identificador para o grafo
n	número de vértices
m	número de arestas
v1–v2	primeira aresta maximal
v2–v3	segunda aresta maximal
.	
.	
.	
v(2i –1) – v(2i)	última aresta maximal
0–0	delimitador, fim das arestas maximais, obrigatório
$	delimitador, obrigatório

Figura 5.20: Formato do arquivo de entrada

Capítulo 6

Conclusão

Nesta dissertação estudamos o problema da coloração de arestas para a família dos grafos indiferença. Os resultados mais expressivos conhecidos para esta classe de grafos são:

- Todo grafo indiferença de grau máximo ímpar pertence à *Classe* 1.
- Todo grafo indiferença reduzido pertence à *Classe* 1.

Estudamos e utilizamos uma abordagem não muito usual para analisarmos o comportamento do problema da coloração. Utilizamos um modelo de programação linear e um software de programação matemática para resolver várias instâncias de nosso problema, e a partir destes resultados analisarmos o comportamento do problema tentando encontrar uma solução. Esta abordagem mostrou ser bastante eficiente para gerar idéias que possam levar a uma prova, podendo ser utilizada em diversos outros trabalhos de pesquisa.

Definimos um novo método para coloração de grafos indiferença, decompondo o grafo em pedaços, de forma que resolver o problema para um grafo indiferença qualquer equivale a resolver o problema para um grafo indiferença semi-universal. Mais especificamente o problema se resume a harmonizar a coloração de dois pedaços que se intersectam, e que individualmente podem ser pintados com Δ cores.

Deixamos como contribuição uma ferramenta que recebe como entrada um grafo indiferença qualquer e dá como saída uma coloração ótima para o grafo. Esta ferramenta utiliza modelos de programação linear para resolver o problema da coloração do grafo, e

pode ser acessada através de uma interface web disponível em
http://www.ic.unicamp.br/~ra006989/indifference.

Fundamentado em extensivos testes práticos, realizamos coloração em milhares de grafos indiferença de diferentes tamanhos e estrutura, e reforçamos ainda mais a seguinte conjectura enunciada em [9]:

Conjectura 4 (Figueiredo, Meidanis, Mello 1997) *As seguintes sentenças são equivalentes para grafos indiferença:*

- *O grafo é neighborhood overfull.*
- *O grafo é subgrafo-overfull.*
- *O grafo é Classe* 2.

Em 2000, os mesmos autores provaram a equivalência entre subgrafo-overfull e neighborhood overfull para os grafos indiferença.

Deixamos como direções para trabalhos futuros a abordagem de se desenhar os grafos pintados pela nossa ferramenta tentando encontrar padrões nas colorações geradas, e a busca de novas alternativas para harmonizar a coloração entre pedaços de um grafo indiferença semi-universal.

Referências Bibliográficas

[1] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. Elsevier North Holland, 1976.

[2] L. Cai and J. A. Ellis. NP-completeness of edge-colouring some restricted graphs. *Discrete Applied Mathematics*, 30:15–27, 1991.

[3] B. Chen, H. Fu, and M. T. Ko. Total chromatic number and chromatic index of split graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 17:137–146, 1995.

[4] A. G. Chetwynd and A. J. W. Hilton. The edge-chromatic class of graphs with maximum degree at least $|v| - 3$. *Annals of Discrete Mathematics*, 41:91–110, 1989.

[5] D. G. Corneil, H. Kim, S. Natarajan, S. Olariu, and A. P. Sprague. Simple linear time recognition of unit interval graphs. *Information Processing Letters*, 55(2):99–104, 1995.

[6] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. A greedy method for edge-coloring odd maximum degree doubly chordal graphs. *Congressus Numerantium*, 111:170–176, 1995.

[7] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. A linear-time algorithm for proper interval graph recognition. *Information Processing Letters*, 56(3):179–184, 1995.

[8] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. On the edge-coloring of split graphs. *Technical Report Unicamp IC-96-04*, 1996.

[9] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. On edge-colouring indifference graphs. *Theoretical Computer Science*, 181:91–106, 1997.

[10] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. Total-chromatic number and chromatic index of dually chordal graphs. *Information Processing Letters*, 70(3):147–152, 1999.

[11] C. M. H. de Figueiredo, J. Meidanis, and C. P. de Mello. Local conditions for edge coloring. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 32:79–91, 2000.

[12] C. M. H. de Figueiredo, J. Meidanis, C. P. de Mello, and C. Ortiz. Decompositions for the edge colouring of reduced indifference graphs. *Theoretical Computer Science*, 297(1-3):145–155, 2003.

[13] C. M. H. de Figueiredo, C. P. Mello, J. Szwarcfiter, and S. Klein. Ordens Indiferença. *Pesquisa Operacional*, 11:43–47, 1991.

[14] C. M. H. Figueiredo, J. Meidanis, and C. P. Mello. Coloração em grafos. In *XVI Jornada de Atualização em Informática*, pages 39–83. Sociedade Brasileira de Computação, 1997.

[15] S. Fiorini. A bibliographic survey of edge-colorings. *Journal of Graph Theory*, 2:93–106, 1978.

[16] S. Fiorini and R. J. Wilson. Edge colorings of graphs. In Lowell W. Beineke and Robin J. Wilson, editors, *Selected Topics in Graph Theory*, chapter 5, pages 103–125. Academic Press, 1978.

[17] P. C. Gilmore and A. J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canada J. Math*, 16:539–548, 1964.

[18] M. C. Golumbic. *Algorithmic graph theory and perfect graphs*. Boston : Academic Press, 1980.

[19] A. J. W. Hilton. Recent progress on edge-colouring graphs. *Discrete Mathematics*, 64:303–307, 1987.

[20] A. J. W. Hilton. Two conjectures on edge-colouring. *Discrete Mathematics*, 74:61–64, 1989.

[21] A. J. W. Hilton and Z. Cheng. The chromatic index of a graph whose core has maximum degree two. *Discrete Mathematics*, 101:135–147, 1992.

[22] D. G. Hoffman and C. A. Rodger. The chromatic index of complete multipartite graphs. *Journal of Graph Theory*, 16(2):159–163, 1992.

[23] Ian Holyer. The NP-Completeness of Edge-Coloring. *SIAM Journal on Computing*, 10:718–720, 1981.

[24] T. R. Jensen and B. Toft. *Graph Coloring Problems*. Wiley-Interscience, 1995.

[25] D. Leven and Z. Galil. NP-completness of finding the chromatic index of regular graphs. *J. Algorithms*, 4:35–44, 1983.

[26] H. Maehara. On time graphs. *Discrete Mathematics*, 32:281–289, 1980.

[27] J. Misra and D. Gries. A constructive proof of Vizing's theorem. *Information Processing Letters*, 41:131–133, 1992.

[28] U. S. R. Murty. Fractional edge colorings. *Manuscript*, 1996.

[29] S. Nakano, X. Zhou, and T. Nishizeki. Edge-coloring algorithms. *Computer Science Today,Lect. Notes in Computer Science, Springer-Verlag, Special issue*, 1000:172–183, 1995.

[30] C. Ortiz. *Sobre Coloração de Arestas*. PhD thesis, COPPE-UFRJ, 1992.

[31] M. J. Plantholt. The chromatic index of graphs with a spanning star. *Journal of Graph Theory*, 5:45–53, 1981.

[32] F. S. Roberts. Indifference graphs. *in Proof Techniques in Graph Theory, F. Harary, ed., Academic Press, New York*, pages 139–146, 1969.

[33] F. S. Roberts. On the compatibility between a graph and a simple order. *Journal of Combinatorial Theory*, 11:28–38, 1971.

[34] X. Zhou and T. Nishizeki. Edge-coloring and f-coloring for various classes of graphs. *Journal of Graph Algorithms and Applications*, 3:1–18, 1999.