UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

COLORAÇÕES EM GRAFOS COM RESTRIÇÕES DE DISTÂNCIA: CARACTERIZAÇÕES, POLITOPOS E ESTRATÉGIAS EM PROGRAMAÇÃO MATEMÁTICA

BRUNO RAPHAEL CARDOSO DIAS

Outubro de 2019

Manaus - AM

BRUNO RAPHAEL CARDOSO DIAS

COLORAÇÕES EM GRAFOS COM RESTRIÇÕES DE DISTÂNCIA: CARACTERIZAÇÕES, POLITOPOS E ESTRATÉGIAS EM PROGRAMAÇÃO MATEMÁTICA

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas, como parte dos requisitos necessários à obtenção do título de Doutor em Informática.

Orientadora: Rosiane de Freitas Rodrigues, Dr[a].

Co-orientador: Nelson Maculan Filho, Dr.

Outubro de 2019

Manaus - AM

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Dias, Bruno Raphael Cardoso
D541c Colorações em grafos com restrições de distância: caracterizações, politopos e estratégias em programação matemática / Bruno Raphael Cardoso Dias. 2019
196 f.: il. color; 31 cm.

Orientadora: Rosiane de Freitas Rodrigues
Coorientador: Nelson Maculan Filho
Tese (Doutorado em Informática) - Universidade Federal do Amazonas.

1. Coloração de vértices. 2. Combinatória poliédrica. 3. Geometria de distâncias. 4. Otimização combinatória. 5. Teoria dos grafos. I. Rodrigues, Rosiane de Freitas II. Universidade Federal do Amazonas III. Título

PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
INSTITUTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UFAM

FOLHA DE APROVAÇÃO

"Colorações em grafos com restrições de distância: caracterizações, politopos e estratégias em programação matemática"

BRUNO RAPHAEL CARDOSO DIAS

Tese de Doutorado defendida e aprovada pela banca examinadora constituída pelos Professores:

Profa. Rosiane de Freitas Rodrigues - PRESIDENTE

Prof. Altigran Soares da Silva - MEMBRO INTERNO

Prof. Edleno Silva de Moura - MEMBRO INTERNO

Prof. Geraldo Robson Mateus - MEMBRO EXTERNO

Prof. Manoel B. Campêlo Neto - MEMBRO EXTERNO

Prof. Javier Leonardo Marenco - MEMBRO EXTERNO

Manaus, 09 de Outubro de 2019

À minha família,
aos meus amigos verdadeiros
e a quem amo.

Agradecimentos

O caminho no doutorado não foi fácil. Muitas coisas aconteceram que me abalaram, mas ainda assim consegui me reerguer e chegar aqui, ao ponto no qual me tornei doutor. E devo isso a muitas pessoas que passaram pela minha vida nesses últimos anos.

Agradeço, inicialmente, aos meus pais, Maria da Conceição Cardoso Dias e Tomaz da Silva Dias, por tudo que me fizeram desde que vim a esse mundo. Minha mãe, que teve problemas de saúde que mudaram para sempre minha vida e a de minha família, que me fez criar um lado que eu não tinha, o de cuidador... ainda assim ela sempre me incentivou a seguir em frente e não me deixar abalar. E meu pai, que também me deu todo o apoio, me ajudando financeiramente quando precisei, permitindo que eu tivesse meus momentos próprios para evoluir na vida... nunca conseguirei agradecer a vocês o suficiente por tudo.

À minha irmã, Bianca Gabriela Cardoso Dias e meu cunhado (e quase ex-aluno) Marcos Antônio Batista de Oliveira, também devo-lhes agradecer por terem me ajudado a superar meus medos após minha mudança de vida, não deixando eu me sentir desamparado durante os anos do doutorado. E deixo registrado também um agradecimento à minha sobrinha Maria Clara Dias de Oliveira, que conseguiu amolecer um pouco mais meu coração, o que me deu ânimo pra seguir neste trabalho.

Aos meus amigos, que tanto me ajudaram (e ajudam) nos momentos de alegria e dor, e permitiram que eu pudesse aproveitar a vida de diferentes formas que eu sozinho não conseguiria. Nesses anos de doutorado, foram muitos amigos que fiz, além dos que já tinha, e gosto muito de conviver com todos, pois são muito importantes para mim. Apesar disso, preciso deixar um agradecimento ao amigo Ítalo Veloso, que está sempre ao meu lado em tudo que preciso, que ouve meus desabafos e me dá conselhos valiosos. Também deixo aqui um agradecimento muito especial ao Rafael Pablo Chaparro, que conheci há pouco mais de um ano desde o momento em que escrevo e se tornou uma pessoa muito importante em minha vida, um grande companheiro nessa aventura da vida.

À minha orientadora, Rosiane de Freitas Rodrigues, pelo trabalho dedicado a minha formação acadêmica desde que fui seu aluno de graduação e pelo apoio no caminho trilhado rumo ao meu sonho profissional. Ainda lembro quando a procurei, enquanto seu aluno de graduação de Projeto e Análise de Algoritmos, para tentar entender o teorema de Cook-Levin, e eventualmente ela me abriu as portas para mergulhar mais profundamente no mundo da teoria da computação e otimização combinatória. Meu muito obrigado pela parceria.

Ao prof. Javier Marenco, pela grande ajuda que me permitiu navegar no mundo da combinatória poliédrica.

Aos meus colegas do grupo de Otimização, Algoritmos e Complexidade Computacional do IComp/UFAM, pela companhia tanto física quando virtual, pelas discussões sobre os temas estudados e, claro, pelos momentos descontraídos.

Aos membros da banca examinadora, que forneceram contribuições importantes e sugestões interessantes relacionadas a este trabalho.

À secretaria do Instituto de Computação da UFAM, que sempre foi prestativa nas mais diversas solicitações feitas durante este doutorado.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Méliuz, pelo indispensável suporte financeiro por meio de bolsas de estudo.

Resumo da Tese apresentada ao PPGI/IComp/UFAM como parte dos requisitos necessários para a obtenção do grau de Doutor em Informática.

COLORAÇÕES EM GRAFOS COM RESTRIÇÕES DE DISTÂNCIA: CARACTERIZAÇÕES, POLITOPOS E ESTRATÉGIAS EM PROGRAMAÇÃO MATEMÁTICA

Bruno Raphael Cardoso Dias

Outubro/2019

Orientadora: Prof[a]. Dr[a]. Rosiane de Freitas Rodrigues

Co-orientador: Prof. Dr. Nelson Maculan Filho

Coloração em grafos constitui uma das classes de problemas de otimização combinatória de maior relevância teórica e prática, com variações envolvendo restrições adicionais nos vértices ou arestas. Uma das aplicações mais importantes ocorre em telecomunicações, envolvendo a alocação de canais em redes móveis sem fio, na qual canais devem ser atribuídos a um conjunto de dispositivos, de modo a se evitar interferências. O problema de coloração em largura de banda (do inglês, *Bandwidth Coloring Problem* - BCP) modela o caso geral de tal aplicação, onde vértices adjacentes recebem cores diferentes mas tais cores devem respeitar uma separação imposta por meio de pesos nas arestas, generalizando o problema clássico de coloração de vértices em grafos (do inglês, *Vertex Coloring Problem* - VCP). No entanto, esse modelo não aborda todos os casos aplicáveis à alocação de canais, possibilitando a utilização de outros ferramentais teóricos para identificar novos modelos. Sendo assim, nesta tese, são caracterizados novos problemas de coloração de vértices com restrições adicionais de distâncias, baseados em conceitos de geometria de distâncias e teoria dos grafos, com tipos de restrições de adjacência envolvendo igualdades e desigualdades, e valores arbitrários e uniformes para representar as distâncias, aplicáveis a diferentes características da alocação de canais, como comunicação uni- e bidirecional. Esses modelos teóricos definem uma grande subclasse de problemas de colorações com distâncias em grafos, para a qual foram estabelecidas algumas propriedades de factibilidade e complexidade computacional. Formulações de programação por restrições foram propostas com base nas definições dos problemas abordados, com o uso de restrições globais para tratar de vértices que requeiram mais

de uma cor, bem como formulações em programação inteira, onde uma já existente foi refinada e outras duas novas foram propostas, baseadas em orientações do grafo de entrada e nas distâncias entre cores atribuídas. Ambas as formulações propostas possuem tamanho polinomial, ao contrário dos modelos existentes que são pseudo-polinomiais. Um estudo poliedral foi realizado onde, para os novos politopos correspondentes, são definidas propriedades e algumas famílias de desigualdades válidas que induzem a facetas sob determinadas condições. Para avaliar o impacto de tais formulações matemáticas, estratégias computacionais exatas foram utilizadas, com destaque para um algoritmo *cut-and-branch* desenvolvido com base no politopo de orientação e nas desigualdades válidas propostas. Experimentos realizados utilizando instâncias do BCP e de alocação de canais da literatura (onde os vértices podem necessitar de uma cor ou de mais cores) permitiram estabelecer melhores limites superiores para determinadas instâncias, com a prova de otimalidade de soluções heurísticas apresentadas na literatura, e principalmente a obtenção de novos ótimos para outros casos. Foi realizada uma análise comparativa entre as estratégias de programação inteira e por restrições, considerando se os vértices demandam uma ou mais cores. Por fim, foi possível verificar que as novas formulações, em especial a baseada em orientação e o método *cut-and-branch*, permitiram a obtenção de soluções ótimas em menor tempo de execução para diversas instâncias utilizadas, estabelecendo assim que os novos modelos têm bom potencial para solucionar problemas de colorações com distâncias. Os resultados obtidos fornecem contribuições importantes em geometria de distâncias, combinatória poliédrica e teoria dos grafos para a classe de problemas de colorações de vértices em grafos.

Palavras-chave: coloração de vértices, combinatória poliédrica, geometria de distâncias, otimização combinatória, teoria dos grafos.

Abstract of Thesis presented to PPGI/IComp/UFAM as a partial fulfillment of the requirements for the degree of Doctor in Informatics.

GRAPH COLORING PROBLEMS WITH DISTANCE CONSTRAINTS: CHARACTERIZATIONS, POLYTOPES AND MATHEMATICAL PROGRAMMING STRATEGIES

Bruno Raphael Cardoso Dias

October/2019

Advisor: Prof. Dr. Rosiane de Freitas Rodrigues

Co-advisor: Prof. Dr. Nelson Maculan Filho

Graph coloring constitute a class of combinatorial optimization problems of great theoretical and practical relevance, with variations involving additional constraints to vertices or edges. One of its most important applications is in telecommunications, involving channel assignment in mobile wireless networks, where channels must be attributed to a set of devices while avoiding interferences. The Bandwidth Coloring Problem (BCP) addresses the general scenario of this application, where adjacent vertices receive different colors, which in turn must respect a separation that is imposed using weighted edges, generalizing the classic Vertex Coloring Problem (VCP). However, this model does not take into account all scenarios of channel assignment, creating the possibility of using other theoretical approaches to identify new models. In this thesis, we characterize new coloring problems with additional distance constraints, based on distance geometry and graph theory concepts, involving equalities and inequalities, and arbitrary and uniform values to represent distances, which can be applied to different characteristics of channel assignment, such as uni- and bidirectional communication. These theoretical models define a subclass of colorings with distances in graphs, for which we establish feasibility and computational complexity properties. We propose constraint programming formulations based on such problem definitions, using global constraints when vertices need more than one color. Also, we explore integer programming models for BCP, improving an existing one and proposing two new others, based on orientations of the input graph and distances between colors assigned to vertices. Both new models have polynomial size, in contrast to existing ones that are pseudopolynomial. A polyhedral

study is made where, for the new corresponding polytopes, we present their properties and some families of valid inequalities which define facets under certain conditions. To evaluate these new mathematical formulations, we developed exact computational strategies, including a cut-and-branch algorithm based on the orientation polytope and its valid inequalities. Experiments made using literature instances for BCP and channel assignment (where vertices may need one color or more colors) led to new better upper bounds for some instances and optimality proofs of heuristic solutions presented in the literature, and, principally, to new optimal solutions in other cases. Through these experiments, a comparative analysis between integer and constraint programming is presented, considering if vertices demand one or more colors. We observed that the new formulations, specially the orientation-based model and the cut-and-branch developed with it, were able to obtain optimal solutions in less time for many instances, establishing that the models have good potential to solve distance coloring problems. Our results provide important contributions for Graph Theory, Distance Geometry and Polyhedral Combinatorics for the graph coloring problems class.

Keywords: combinatorial optimization, distance geometry, graph theory, polyhedral combinatorics, vertex coloring.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

B&B	*Branch-and-bound*.
B&C	*Branch-and-cut*.
BCP	Problema de Coloração em Largura de Banda (*Bandwidth Coloring Problem*).
BMCP	Problema de Multicoloração em Largura de Banda (*Bandwidth Multicoloring Problem*).
BP	*Branch-and-prune*.
BPB	*Branch-prune-and-bound*.
CDGP	Problema de Coloração em Geometria de Distâncias (*Coloring Distance Geometry Problem*).
DDCF	Factibilidade Direta de Coloração com Distâncias (*Direct Distance Coloring Feasibility*).
DDF	Factibilidade Direta de Distância (*Direct Distance Feasibility*).
DDGP	Problema de Geometria de Distâncias Discretizáveis (*Discretizable Distance Geometry Problem*).
EQ-CDGP	Problema de Coloração em Geometria de Distâncias Iguais (*Equal Distance Geometry Problem*).
FEA	Alocação Exaustiva por Frequência (*Frequency Exhaustive Allocation*).
LEQ-CDGP	Problema de Coloração em Geometria de Distâncias Menores ou Iguais (*Less than or Equal Distance Geometry Problem*).

MI-CAP	Problema de Alocação de Canais de Mínima Interferência (*Minimum Interference Channel Assignment Problem*).
MinEQ-CDGP	Problema de Coloração Mínima em Geometria de Iguais (*Minimum Equal Distance Geometry Problem*).
MinGEQ-CDGP	Problema de Coloração Mínima em Geometria de Distâncias Maiores ou Iguais (*Minimum Greater than or Equal Distance Geometry Problem*).
MinLEQ-CDGP	Problema de Coloração Mínima em Geometria de Distâncias Menores ou Iguais (*Minimum Less than or Equal Distance Geometry Problem*).
MinEQ-CDGP-Unif	Problema de Coloração Mínima em Geometria de Distâncias Uniformes Iguais (*Minimum Equal Uniform Distance Geometry Problem*).
MinGEQ-CDGP-Unif	Problema de Coloração Mínima em Geometria de Distâncias Uniformes Maiores ou Iguais (*Minimum Greater than or Equal Uniform Distance Geometry Problem*).
MinLEQ-CDGP-Unif	Problema de Coloração Mínima em Geometria de Distâncias Uniformes Menores ou Iguais (*Minimum Less than or Equal Uniform Distance Geometry Problem*).
MO-CAP	Problema de Alocação de Canais de Mínima Ordem (*Minimum Order Channel Assignment Problem*).
MS-CAP	Problema de Alocação de Canais de Mínima Extensão (*Minimum Span Channel Assignment Problem*).
PI	Programação inteira.
REA	Alocação Exaustiva por Requisição (*Requirement Exhaustive Assignment*.)
VCP	Problema de Coloração de Vértices (*Vertex Coloring Problem*).

Capítulo 1

Introdução

Seja $G = (V, E)$ um grafo não direcionado, onde V é seu conjunto de vértices e E o de arestas. O problema de coloração de vértices em grafos (*vertex coloring problem* - VCP) consiste em colorir os elementos de V de modo que vértices adjacentes (ou seja, ligados por uma aresta) não compartilhem a mesma cor. Logo, deve-se atribuir, para cada vértice $i \in V$, uma cor $c(i)$ (representada por um número natural) tal que para cada aresta $(i, j) \in E$, tem-se que $c(i) \neq c(j)$ (Bondy e Murty, 1982). A Figura 1.1 fornece um pequeno exemplo de instância do VCP.

O estudo da coloração de grafos foi iniciado em 1852 por Francis Guthrie, que, ao tentar colorir o mapa da Inglaterra, percebeu que bastavam quatro cores e conjecturou que isso seria o caso para qualquer mapa (Lewis, 2016). Um mapa pode ser visto como um grafo planar, onde cada território corresponde a um vértice e, entre territórios que compartilham uma fronteira, há uma aresta. Como o grafo é planar, há pelo menos uma forma de as arestas serem esenhadas sem que se cruzem.

A conjectura de Guthrie somente foi provada no século XX por Appel e Haken (1977). Uma das características mais interessantes da prova é que mesma foi a primeira desenvolvida com auxílio de computador. Nela, os autores reduziram o conjunto de todos os

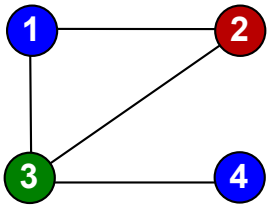


Figura 1.1: Examplo de instância do problema clássico de coloração de vértices em grafos e sua solução ótima com 3 cores.

grafos planares a um conjunto de 1936 configurações inevitáveis, que foi posteriormente reduzido a 1482 elementos (Appel et al., 1977).

Atualmente, coloração de grafos constitui um dos problemas mais estudados em teoria dos grafos e otimização combinatória, possuindo inúmeras generalizações e aplicações, tais como escalonamento de tarefas (de Freitas et al., 2010), atribuição de registradores em compiladores (Chaitin, 1982), determinação de horários escolares (*timetabling*) (Burke et al., 2010), dentre outros. Entretanto, o problema é NP-difícil (e sua versão de decisão é um dos 21 problemas NP-completos de Karp (Karp, 1972)), o que requer o uso de técnicas mais avançadas para a obtenção de soluções exatas em tempo computacional aceitável (mas ainda exponencial).

Além das aplicações supracitadas, uma das mais importantes é a alocação de canais em redes móveis sem fio, que é amplamente explorada com o avanço das tecnologias de comunicação móvel nos últimos anos. Nela, há um conjunto de dispositivos que devem receber canais, respeitando restrições de separação de modo a evitar interferências (Audhya et al., 2011; Dias et al., 2012; Dias, 2014). Diversas variações de colorações em grafos foram propostas para esta aplicação, como a T-coloração, na qual há o conceito de conjuntos proibidos (onde, para cada aresta, existe um conjunto $T_{i,j}$), de forma que uma coloração válida passa a ser um mapeamento $c : V \to \mathbb{N}$, onde $\forall (i,j) \in E,\ |c(i) - c(j)| \notin T_{i,j}$ (Hale, 1980). Um caso especial muito importante da T-coloração ocorre quando, para cada aresta $(i,j) \in E$, tem-se $T_{i,j} = \{0, 1, \ldots, d(i,j) - 1\}$, onde $\forall (i,j) \in E,\ d(i,j) \in \mathbb{N}$. O valor $d(i,j)$ representa uma distância a ser respeitada entre as cores atribuídas aos vértices i e j, ou seja, $|c(i) - c(j)| \geq d(i,j)$. Tal caso é conhecido como coloração em largura de banda (*bandwidth coloring problem* - BCP), já que surgiu em tal contexto de alocação de canais (Trick et al., 2002; Lai e Lü, 2013). A Figura 1.2 mostra um exemplo de problema de alocação de canais e sua correlação com a coloração em grafos.

Durante a pesquisa a nível de mestrado (Dias, 2014), explorou-se aspectos tecnológicos de alocação de canais, seguindo um esquema fixo de com acesso múltiplo dividido por frequência (*Frequency-division Multiple Access* - FDMA), sendo também apresentada uma proposta de uso de métodos para esse esquema em rádios cognitivos a partir de uma extensão de pré-alocação em um retrato temporal da rede. Com base nisso, foi desenvolvido um método aproximado, seguindo a meta-heurística *simulated annealing*, para a alocação de canais (Dias et al., 2013a), considerando o problema modelado usando

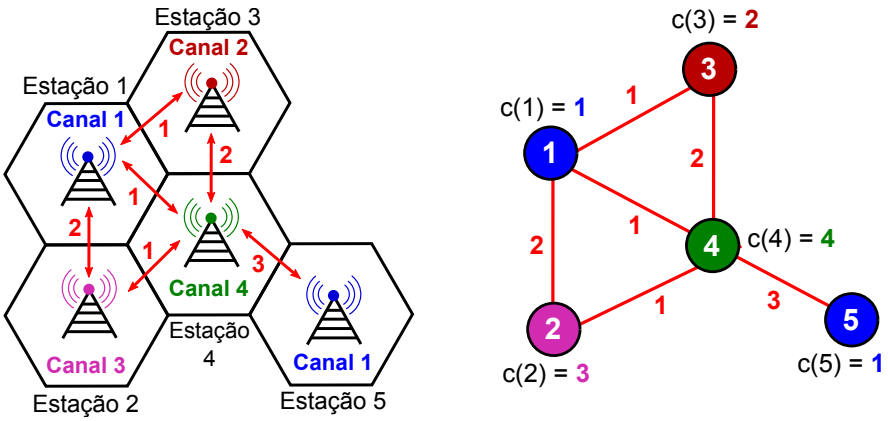


Figura 1.2: Exemplo de problema de alocação de canais e sua modelagem como coloração em largura de banda. O valor em uma aresta $(i, j) \in E$ indica o valor $d(i, j)$.

os problemas de coloração em grafos citados acima. Como continuação da pesquisa realizada no mestrado, esta tese aborda aspectos teóricos dos problemas de coloração em grafos relacionados à alocação de canais, incluindo modelos com restrições adicionais de distâncias, baseando-se no ferramental de geometria de distâncias, bem como contribuições científicas em combinatória poliédrica, teoria dos grafos e programação matemática para a determinação de soluções ótimas.

1.1 Questão de pesquisa e objetivos

Apesar da existência de variações de colorações em grafos verificou-se uma lacuna em relação a alguns tipos de restrições que podem ser exploradas, em especial as que envolvem distâncias e que podem ser aplicadas no contexto de aplicação de canais. Assim, esta tese tem como objetivo principal responder ao seguinte questionamento: é possível estabelecer modelos teóricos representativos e métodos de resolução eficazes que explorem características geométricas de problemas de coloração de vértices com distâncias? Tal questão origina-se do fato de que certas variações de coloração são pouco exploradas do ponto de vista de obtenção de métodos de solução, em especial exatos. Baseando-se em tais questionamentos, tem-se três objetivos específicos do trabalho, que são:

1. Identificação de variações interessantes do problema de coloração em grafos envolvendo pesos nas arestas, podendo ser valores inteiros positivos arbitrários ou uniformes, e diferentes tipos de restrições entre cores de vértices adjacentes, bem

como propriedades grafo-teóricas relacionadas à factibilidade e complexidade computacional das mesmas.

2. Desenvolvimento de formulações baseadas em modelagens matemáticas, em especial programação inteira e por restrições, para colorações com distâncias, com base em características geométricas do problema que permitam a exploração de conceitos de geometria de distâncias e combinatória poliédrica.

3. Elaboração de algoritmos para solução de tais colorações, em especial métodos exatos com planos de corte em enumeração implícita baseados nos modelos desenvolvidos no item anterior, que permitam a obtenção de soluções de forma mais eficiente em determinados cenários.

1.2 Contribuições da tese

Nesta tese, foram obtidos os principais conjuntos de contribuições científicas enumerados a seguir:

1. Caracterização de novos problemas de colorações com distâncias, que foram correlacionados em uma estrutura hierárquica entre os problemas existentes de colorações de vértices e geometria de distâncias, a qual forma a subclasse de problema de colorações com distâncias, sendo apresentadas propriedades de factibilidade e complexidade computacional para a mesma.

2. Estratégias baseadas em programação matemática, de modo que uma formulação de programação inteira existente para o BCP, que é relacionado às colorações com distâncias, foi melhorada e duas de programação por restrições foram propostas para o BCP e sua variante com demandas múltiplas, onde os vértices podem precisar de uma ou mais cores distintas, havendo também uma distância mínima entre cores do mesmo vértice.

3. Novos modelos de programação inteira, sendo um, para o BCP, dito baseado em orientações (uma vez que as soluções de tal modelo induzem a orientações do grafo de entrada), tendo tamanho linear em relação aos vértices e arestas, e o outro, para o VCP, baseado nas distâncias entre cores atribuídas aos vértices.

4. Métodos de resolução baseados nos modelos abordados nesta tese, como um *cut-and-branch* para a formulação baseada em orientações, além de algoritmos *branch-prune-and-bound* propostos a partir dos modelos teóricos de colorações com distâncias, cujos estudos foram aprofundados nesta tese a partir da versão preliminar feita na pesquisa de mestrado.

5. Novos limites superiores para diversas instâncias do BCP presentes na literatura, incluindo novas soluções ótimas para algumas destas, bem como provas de otimalidade para resultados heurísticos da literatura.

Além disso, algumas outras contribuições de produção também foram obtidas, tais como uma revisão em português da literatura sobre colorações em grafos, bem como diversas publicações em eventos e periódicos da área.

1.3 Organização do documento

O restante desta tese está organizado como se segue. O Capítulo 2, mostra uma revisão da literatura sobre colorações em grafos, em especial com resultados para a coloração clássica e em largura de banda, envolvendo limites, formulações de programação inteira e algoritmos exatos e aproximados. No Capítulo 3, são apresentados os resultados teóricos obtidos para modelos de colorações com restrições de distância (*distance coloring*), tais como caracterizações de factibilidade e complexidade. O Capítulo 4, mostra modelos de programação inteira e por restrições para a coloração em largura de banda, além de detalhes de suas implementações computacionais e resultados empíricos com instâncias da literatura. No Capítulo 5 são apresentados dois novos modelos de programação inteira. O primeiro é dito ser baseado em orientação para a coloração em largura de banda (já que as soluções induzem a orientações do grafo a ser colorido), com o qual também é mostrado um método *cut-and-branch*. Já o segundo é baseado em distâncias, proposto para a coloração clássica e desenvolvido com base no primeiro, com resultados teóricos preliminares. Por fim, no Capítulo 6, são feitas as considerações finais, incluindo uma lista de atividades realizadas, e indicados os próximos passos da pesquisa.

Capítulo 2

Coloração em largura de banda

Neste capítulo, será apresentada uma revisão da literatura para a coloração em largura de banda (BCP). Mais especificamente, será mostrada uma definição formal de tal problema, bem como modelos de programação inteira e algoritmos exatos e aproximados presentes em trabalhos publicados na literatura.

A revisão para esse problema foi feita pelo fato de que a coloração em largura de banda é um dos problemas que já envolve pesos nas arestas, servindo de base para a generalização de outros tipos de restrição de adjacência, que são discutidos no Capítulo 3.

2.1 Definições

Como já foi dito, no problema de coloração em largura de banda (*bandwidth coloring problem* - BCP), a restrição de cores para vértices adjacentes não exige apenas que as mesmas sejam diferentes, mas sim que respeitem uma determinada separação. Uma definição formal do BCP é fornecida a seguir.

Definição 2.1.1. ***Problema de Coloração em Largura de Banda (Bandwidth Coloring Problem - BCP):*** *dado um grafo simples não direcionado $G = (V, E)$, onde V é o conjunto de vértices e E o de arestas, e uma função $d : E \to \mathbb{N}$ de pesos para as arestas encontrar um mapeamento $c : V \to \mathbb{N}$ tal que, para cada $(i, j) \in E$, tenha-se que $|c(i) - c(j)| \geq d(i, j)$ e a maior cor atribuída a um vértice ($\max_{i \in V} c(i)$) seja minimizada.*

Pela definição do BCP, observa-se duas principais características: a introdução de pesos nas arestas, que representam limites inferiores para a diferença absoluta de cores

entre vértices vizinhos, e a mudança na função objetivo, onde, ao invés de minimizar a quantidade de cores usadas, deve-se minimizar a maior cor usada (o que equivale à extensão (*span*) da coloração). O BCP generaliza a coloração clássica de forma que toda instância desta pode ser vista como uma do BCP na qual, para toda aresta $(i, j) \in E$, tem-se que $d(i, j) = 1$. Nesse cenário, a extensão da coloração equivalerá a menor quantidade de cores necessária para uma k-coloração.

Assim como na coloração clássica, existe uma variação do BCP onde cada vértice pode requisitar mais de uma cor. Esta variação é denominada de problema de multicoloração em largura de banda (*bandwidth multicoloring problem* - BMCP), cuja definição é dada a seguir.

Definição 2.1.2. ***Problema de Multicoloração em Largura de Banda (Bandwidth Multicoloring Problem - BMCP):*** *dado um grafo não direcionado $G = (V, E)$, onde V é o conjunto de vértices e E o de arestas, uma função $d : E \to \mathbb{N}$ de pesos para as arestas e uma função $w : V \to \mathbb{N}$ de pesos para os vértices determinar, para cada $i \in V$, $w(i)$ cores distintas ($c(i,1)$, $c(i,2)$, $\ldots$, $c(i,w(i))$) tais que para toda aresta $(i, j) \in E$, tenha-se, para todo $1 \leq p \leq w(i)$ e $1 \leq q \leq w(j)$, que $|c(i, p) - c(j, q)| \geq d(i, j)$, de modo que a maior cor atribuída a um vértice ($\max_{i \in V, 1 \leq p \leq w(i)} c(i, p)$) seja minimizada.*

Ambas as colorações acima originaram-se no contexto do planejamento de redes de telecomunicações sem fio, mais especificamente na alocação de canais em tais redes, que são modeladas por meio de colorações em grafos (Hale, 1980). Nesse problema, deve-se atribuir porções discretizadas do espectro eletromagnético (os canais) para dispositivos da rede, tais como estações de rádio-base (ERB), de modo a evitar ou minimizar interferências. Uma abordagem clássica para isso envolve o Problema de Alocação de Canais de Mínima Extensão, cuja definição é dada abaixo para fins de completude.

Definição 2.1.3. ***Problema de Alocação de Canais de Mínima Extensão (Minimum Span Channel Assignment Problem - MS-CAP):*** *é dado um conjunto E de estações em uma rede móvel sem fio, onde cada uma requer r_i canais, e uma matriz de compatibilidade $M_{n \times n}$, onde o elemento m_{ij} da mesma indica a separação mínima entre canais atribuídos às estações i e j. Uma alocação de canais válida para E consiste na atribuição de r_i inteiros positivos (representando os canais) para cada estação i de modo que, para duas estações i e j, os canais atribuídos às mesmas devem respeitar a separação mínima,*

isto é, para cada $1 \leq p \leq r_i$ *e* $1 \leq q \leq r_j$*, sendo* $f_{i,p}$ *o p-ésimo canal de i, tem-se que* $|f_{i,p} - f_{j,q}| \geq m_{ij}$*. Deve-se, então, determinar a alocação de menor extensão (ou seja, onde o maior canal usado é o menor possível)*

Dada uma instância do MS-CAP, pode-se tratá-la como uma do BMCP associando-se o conjunto E de estações como o conjunto V de vértices, utilizando cada valor r_i de demanda de canais como o valor $w(i)$ de demanda de cores, além de considerar os valores m_{ij} da matriz de compatibilidade como o peso $d(i, j)$ de uma aresta entre os vértices i e j. Tal matriz serve como abstração dos diversos fatores que causam interferências entre dispositivos de comunicação sem fio, tais como a potência do sinal e a relação sinal-ruído, sendo que os valores da matriz podem ser definidos como funções destas propriedades. As porções do espectro eletromagnético corresponderão às cores no BMCP (Audhya et al., 2011; Dias et al., 2012). Seguindo isso, muitas estratégias algorítmicas e propriedades para o MS-CAP são diretamente aplicáveis às colorações com largura de banda, portanto, nas subseções seguintes, serão abordados também alguns resultados para alocação de canais de mínima extensão onde o modelo utilizado é o mesmo definido acima.

2.2 Modelos teóricos e de programação matemática

Ao contrário do VCP, não existem muitos modelos teóricos distintos que saiam da definição padrão do problema. Apesar de que vértices utilizando cores iguais continuam definindo conjuntos independentes no BCP, os valores numéricos de tais cores passam a ter importância, uma vez que a restrição de adjacência envolve a diferença absoluta entre cores dos extremos das arestas. Sendo assim, os modelos de programação matemática a seguir são derivados da definição do problema.

O primeiro modelo definido para o BCP é baseado na formulação padrão para a coloração clássica , que foi originalmente proposta para uma variante mais geral do MS-CAP que permite que a menor cor usada seja diferente de 1, o que não é necessário no BCP.

Para maior clareza, o modelo a ser mostrado a seguir foi adaptado nesta revisão diretamente para o BCP. São usados os seguintes conjuntos de variáveis:

- $x_{ik} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao vértice } i; \\ 0 & \text{caso contrário.} \end{cases}$

- $y_k = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída a algum vértice;} \\ 0 & \text{caso contrário.} \end{cases}$
- z_{max}: maior cor usada na solução.

Assim, tem-se a seguinte formulação de programação inteira:
(PI-BCP-Padrão)

$$\text{Minimizar} \quad z_{max} \tag{2.1}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{U} x_{ik} = 1 \qquad (\forall i \in V) \tag{2.2}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ |k-m| < d(i,j)) \tag{2.3}$$

$$x_{ik} \leq y_k \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{2.4}$$

$$z_{max} \geq k y_k \qquad (\forall k = 1,2,...,U) \tag{2.5}$$

$$x_{ik} \in \{0,1\} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{2.6}$$

$$y_k \in \{0,1\} \qquad (\forall k = 1,2,...,U) \tag{2.7}$$

A função objetivo (2.1) consiste no valor da maior cor usada, que será minimizada. As restrições (2.2) indicam que cada vértice deve receber uma cor (no caso do BMCP, pode-se substituir o lado direito por $w(i)$ ao invés de expandir o grafo de entrada). O conjunto de restrições (2.3) faz com que vértices adjacentes não recebam cores que violem a separação mínima exigida entre elas. As restrições (2.4) sincronizam as variáveis do conjunto x e y, de modo que para uma determinada cor k, houver uma variável x_{ik} com valor 1 (independentemente do vértice i), então a variável y_k também deverá ter valor 1. O conjunto de restrições (2.5) garantem que o valor da variável z_{max} será igual ao valor da maior cor usada. Por fim, as restrições (2.6) e (2.7) são de integralidade para os conjuntos de variáveis x e y.

No modelo acima, existem $O(U|V|)$ variáveis no conjunto x e $O(U)$ variáveis no conjunto y, além de uma variável adicional z_{max}, bem como $O(U^2|E| + U|V| + U + |V|)$ restrições. Como as variáveis são indexadas pela cor, a quantidade das mesmas depende de um limite superior U para a maior cor possível, que é derivado dos valores dos pesos nas arestas. Assim, o modelo tem dimensão pseudopolinomial.

Uma segunda formulação para o BCP é baseada no uso de outra função objetivo en-

volvendo a minimização da ordem (ou seja, de cores usadas). No contexto de alocação de canais, há uma variante similar ao MS-CAP (de onde os modelos anteriores para o BCP se originaram), denominado de Problema de Alocação de Canais de Mínima Ordem (*Minimum Order Channel Assignment Problem*, MO-CAP), onde a diferença é que, ao invés de minimizar-se o maior canal (cor) usado, deve-se minimizar a quantidade de canais (cores) usados. Na formulação padrão, como as variáveis y_k indicam se a cor k foi usada ou não, tem-se que a expressão $\sum_{k=1}^{U} y_k$ equivale à quantidade de cores usadas.

No BCP, a menor cor usada é sempre 1, logo, é possível elaborar uma formulação onde a função objetivo é a mesma do MO-CAP, bastando uma restrição adicional. Tal restrição surgiu de outro modelo originalmente proposto por Baybars (1982) para um caso especial do MO-CAP onde, para cada vértice i, apenas há restrições relacionadas a quais outros vértices não podem usar a mesma cor k de i e quais não podem usar cores adjacentes à de i (ou seja, $k-1$ e $k+1$). Ressalta-se que o BCP é mais geral, pois os pesos nas arestas implicam que dois vértices podem não usar cores que distem mais de 1 entre si (ou seja, que sejam além de apenas adjacentes).

São usados os seguintes conjuntos de variáveis (os mesmos da formulação padrão):

- $x_{ik} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao vértice } i; \\ 0 & \text{caso contrário.} \end{cases}$
- $y_k = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída a algum vértice;} \\ 0 & \text{caso contrário.} \end{cases}$

Assim, tem-se a seguinte formulação já adaptada para o BCP (Aardal et al., 2007):

(PI-BCP-MinOrdem)

$$\text{Minimizar} \quad \sum_{k=1}^{U} y_k \tag{2.8}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{U} x_{ik} = 1 \qquad (\forall i \in V) \tag{2.9}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ |k-m| < d(i,j)) \tag{2.10}$$

$$x_{ik} \leq y_k \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{2.11}$$

$$y_{k+1} \leq y_k \qquad (\forall k = 1,2,...,U-1) \tag{2.12}$$

$$x_{ik} \in \{0,1\} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{2.13}$$

$$y_k \in \{0,1\} \qquad (\forall k = 1,2,...,U) \tag{2.14}$$

As restrições (2.9), (2.10) e (2.11) são, respectivamente, iguais às restrições (2.2), (2.3) e (2.4) da formulação padrão. O conjunto de restrições 2.12 faz com que cores que não tenham sido atribuídas a nenhum vértice, mas que sejam menores que a maior cor usada, sejam contabilizadas na soma da quantidade de cores usadas. Por fim, as restrições (2.13) e (2.14) são de integralidade.

Similarmente à formulação padrão, no modelo acima, existem $O(U|V|)$ variáveis no conjunto x e $O(U)$ variáveis no conjunto y, além de $O(U^2|E| + U|V| + U + |V|)$ restrições Por conta da similaridade entre esta formulação e a padrão, a dimensão de ambas é assintoticamente igual e pseudopolinomial.

A última formulação para o BCP, originalmente desenvolvida para o MS-CAP, foi proposta por Giortzis e Turner (1997). Nesta formulação, todas as variáveis usadas são binárias, de modo que, ao invés de uma variável z_{max} contendo o valor da maior cor usada, é usado um conjunto de variáveis indexadas por cor, que indicam se dada cor é ou não a maior usada.

Novamente, para fins de clareza, a formulação a ser dada abaixo já foi adaptada para o BCP nesta revisão. São usados os seguintes conjuntos de variáveis:

- $x_{ik} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao vértice } i; \\ 0 & \text{caso contrário.} \end{cases}$
- $u_k = \begin{cases} 1 & \text{se a cor } k \text{ for a maior usada;} \\ 0 & \text{caso contrário.} \end{cases}$

A formulação é, então, dada por:

(PI-BCP-TotalBinaria)

$$\text{Minimizar} \quad \sum_{k=1}^{U} k u_k \tag{2.15}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{U} x_{ik} = 1 \qquad (\forall i \in V) \tag{2.16}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ |k-m| < d(i,j)) \tag{2.17}$$

$$\sum_{k=1}^{U} u_k = 1 \tag{2.18}$$

$$x_{ik} + u_m \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ k > m) \tag{2.19}$$

$$x_{ik} \in \{0,1\} \quad (\forall i \in V;\ k = 1,2,...,U) \tag{2.20}$$

$$u_k \in \{0,1\} \quad (\forall k = 1,2,...,U) \tag{2.21}$$

A função objetivo (2.15) consiste em minimizar a soma de cada variável u_k multiplicada pelo respectivo índice k, o que, devido às restrições impostas, equivalerá ao valor da maior cor usada. As restrições (2.16) indicam que cada vértice deve receber uma cor (novamente, para o BMCP, pode-se substituir o lado direito por $w(i)$). O conjunto de restrições (2.17) faz com que vértices adjacentes não recebam cores que violem a separação mínima exigida entre elas. As restrições (2.18) garantem que apenas uma variável u_k terá valor 1. O conjunto de restrições (2.19) faz com que nenhum vértice use uma cor maior que k caso $u_k = 1$ e vice-versa (ou seja, caso um vértice use a cor k, então para qualquer cor $m < k$, $u_m = 0$). Por fim, as restrições (2.20) e (2.21) são de integralidade para os conjuntos de variáveis x e y.

Como nas formulações anteriores, tem-se $O(U|V|)$ variáveis no conjunto x e $O(U)$ variáveis no conjunto u. O maior conjunto de restrições é (2.19), onde há $O(U^2|E|)$ restrições. No entanto, devido ao modo como as restrições estão dispostas, na prática, esta formulação tende a gerar modelos ligeiramente maiores ao ser usada para resolução de uma instância do BCP. Sendo assim, o modelo tem $O(2U^2|E|)$ restrições, além de mais $O(|V|)$ outras de 2.16, totalizando $O(2U^2|E| + |V|)$. Ressalta-se também que, do mesmo modo que ocorre na formulação padrão, esse modelo também tem tamanho pseudopolinomial.

A Tabela 2.1 fornece um sumário das formulações de programação inteira descritas para a coloração clássica.

Tabela 2.1: Formulações de programação inteira para a coloração em largura de banda (BCP).

Formulação	Variáveis	Restrições	Referências
PI-BCP-Padrão - Padrão	$O(U\|V\|+U+1)$	$O(U^2\|E\|+U\|V\|+U+\|V\|)$	-
PI-BCP-MinOrdem - Minimização da ordem	$O(U\|V\|+U)$	$O(U^2\|E\|+U\|V\|+U+\|V\|)$	Baybars (1982); Aardal et al. (2007)
PI-BCP-TotalBinaria - Inteiramente binária	$O(U\|V\|+U+1)$	$O(2U^2\|E\|+\|V\|)$	Giortzis e Turner (1997)

2.3 Algoritmos exatos e aproximados

Ao contrário do que ocorre com a coloração clássica, há poucos métodos exatos para encontrar a solução ótima de uma instância do BCP, sendo os mesmos originados do problema de alocação de canais.

Um método *branch-and-cut* foi proposto por Aardal et al. (1996), com base na formulação padrão (PI-BCP-Padrão). O algoritmo foi proposto originalmente para a minimização da ordem (quantidade de cores usadas), porém, o politopo definido pela formulação é exatamente o mesmo, logo, pode-se usar o método com uma modificação na função objetivo ou, mantendo a mesma, usando o modelo que adiciona restrições para que a minimização da ordem seja equivalente à minimização da maior cor usada (PI-BCP-MinOrdem). Tal algoritmo usa dois tipos de grafos da instância: o primeiro, de interferência, equivale ao original de entrada, enquanto o segundo, de variáveis, utiliza um vértice para cada par de vértice original e cor e arestas indicando violações das restrições. O algoritmo usa desigualdades válidas baseada em cliques de ambos os grafos e um critério de dominância para reduzir o tamanho do problema, bem como heurísticas para transformar soluções contínuas parciais em inteiras e dois critérios de ramificação.

Outro algoritmo exato foi proposto por Mannino e Sassano (2003), também desenvolvido originalmente no contexto de alocação de canais. O método é uma enumeração implícita baseada em *backtracking* com critérios de fixação e consiste em colorir cada vértice de acordo com um critério de seleção e, após esse processo, bloquear, em seus vizinhos, todas as cores que, se atribuídas aos mesmos, violem as restrições de separação entre cores. Para reduzir o tempo de execução, é usado um *backtracking* restrito, no qual apenas uma parte da árvore de enumeração é explorada, isto é, entre dois subproblemas, uma quantidade fixa de soluções parciais é considerada e o restante é ignorada. Dois critérios de fixação são usados no algoritmo. O primeiro, de fixação de cor, tenta remover, de alguns vértices, cores de seu conjunto de possibilidades que não levarão a soluções factíveis. O segundo critério é o de fixação de vértice, que tenta encontrar vértices que podem ser sempre coloridos após todos os outros.

Uma outra abordagem foi proposta por Shirazi e Amindavar (2005), utilizando uma programação dinâmica modificada e também sendo desenvolvido, originalmente, para o MS-CAP, podendo ser aplicado diretamente ao BCP e ao BMCP. O algoritmo trata a coloração em largura de banda como a determinação de um caminho de menor custo

Tabela 2.2: Algoritmos exatos para a coloração em largura de banda (BCP).

Referência	Estratégia algorítmica
Aardal et al. (1996)	*Branch-and-cut* com cortes clique
Mannino e Sassano (2003)	*Backtracking* com podas e restrições
Shirazi e Amindavar (2005)	Programação dinâmica estendida

em um grafo Trellis, onde as colunas correspondem a cada demanda de cores de cada vértice e as linhas às cores disponíveis. A determinação da solução é feita por meio dos caminhos determinados, onde a posição p de um valor no caminho indica a cor do p-ésimo vértice (em um grafo expandido, no caso do BMCP). Dentre todos os caminhos gerados, verifica-se qual tem o menor valor para a maior cor usada e retorna-se o mesmo como solução ótima.

A Tabela 2.2 fornece um resumo das referências citadas de algoritmos exatos para o BCP e BMCP.

Existem diversas heurísticas e meta-heurísticas para o BCP/BMCP na literatura, muitas delas baseadas em construções e operadores usados para a coloração clássica, devidamente modificados para tratar a separação entre cores de vértices adjacentes. As duas heurísticas construtivas mais utilizadas para a coloração clássica, DSATUR$_H$ (Brélaz, 1979) e RLF (Leighton, 1979), também podem ser usadas para a geração de soluções do BCP. No caso do DSATUR$_H$, ao selecionar-se a cor de um vértice, ao invés de somente verificar se algum vizinho usa tal cor, deve-se checar se a separação mínima é respeitada. Já para o RLF, o conceito de conjuntos independente ainda é válido, porém, ao preencher uma classe de cor, um vértice não colorido pode não ser compatível com a cor por violar a separação mínima com algum vértice já colorido (enquanto na coloração clássica, bastaria verificar quais vértices não são adjacentes ao que já foi colorido), o que exige verificar tal restrição ao tentar adicionar cada vértice em uma classe de cor.

Outras heurísticas construtivas, que podem ser usadas tanto para o BCP quanto para o BMCP, foram desenvolvidas por Sivarajan et al. (1989) com base no trabalho de Zoellner e Beall (1977), formando uma coleção de heurísticas. Todos os algoritmos propostos pelos autores partem de uma lista de demandas de cores (para o BCP, basta uma permutação dos vértices). Para a determinação de tal lista, parte-se de uma ordenação dos vértices, que pode ser obtida de duas formas, ambas baseadas nos graus dos vértices. Em seguida, tal ordenação é colocada em uma matriz, repetindo os vértices de acordo com a quantidade

de cores exigidas de cada um, e então podem ser usadas duas estratégias para obter a lista de demandas, uma enumerando cada demanda por linhas e outra por colunas da matriz. Enfim, com base nesta lista, obtém-se uma solução seguindo duas estratégias:

1. Alocação Exaustiva por Frequência (*Frequency Exhaustive Assignment* - FEA): partindo do primeiro elemento da lista de demandas, atribuir, para a demanda atual, a menor cor possível que não viole as restrições.

2. Alocação Exaustiva por Requisição (*Requirement Exhaustive Assignment* - REA): partindo da cor 1, aloca-se, para cada demanda na lista (seguindo a ordem da mesma), tal cor caso nenhuma restrição seja violada. O processo é feito novamente começando da cor 2, colorindo-se as demandas restantes e assim segue-se até que todas sejam satisfeitas.

Cada algoritmo é composto de um método de ordenação de vértices, um de listagem de demandas e uma estratégia de alocação. Sendo assim, Sivarajan et al. (1989) obtiveram oito algoritmos para o BMCP.

Um outro tipo de heurística construtiva compatível com o BMCP foi proposta por Chakraborty (2001), com a finalidade de obter um conjunto de soluções para a instância de entrada. O algoritmo representa uma solução por meio de uma matriz onde cada elemento indica de um vértice usa uma determinada cor ou, caso contrário, se pode ou não usá-la. Partindo de uma matriz inicial, é gerada uma permutação das demandas de cores dos vértices, que servirá de ordem para atribuir a menor cor possível a cada um deles, de modo que, ao atribuir uma cor a um vértice, os vizinhos não poderão usar cores que violem restrições do BMCP. Ao variar as permutações, obtém-se um conjunto de soluções para o BMCP.

Phan e Skiena (2002) desenvolveram um *software* de otimização combinatória denominado *Discropt*, contendo heurísticas *hill climbing*, *simulated annealing* e gulosa incremental, podendo ser utilizadas de forma genérica em diversos problemas discretos. Os autores utilizaram o sistema para resolver o BCP, utilizando as duas primeiras heurísticas citadas, onde, para tratar possíveis soluções infactíveis, é usada uma função objetivo composta, baseada no valor da maior cor usada e na factibilidade da mesma, de acordo com o número de conflitos.

Diversas meta-heurísticas foram desenvolvidas para BCP e BMCP, algumas diretamente para os mesmos e outras originalmente para o MS-CAP, sendo descritas a seguir.

Uma das técnicas usadas para solução do BMCP é a busca local, sendo a base do algoritmo proposto por Lau e Tsang (1998). A função objetivo consiste em minimizar a quantidade de violações das restrições em uma solução e a vizinhança usada consiste em mudar a cor de um vértice de modo a reduzir a quantidade de conflitos. Para que a busca seja mais rápida, ao invés de explorar toda a vizinhança, o algoritmo verifica somente trocas que podem levar a uma melhoria da solução. Assim, caso uma variável não tenha seu valor alterado, a mesma é marcada como inativa e não é mais levada em consideração nas próximas iterações da busca local. Os autores encapsularam esse procedimento em uma busca local guiada (*guided local search - GLS*), onde são definidas características das soluções, com custos e penalidades, e deseja-se remover as características ruins das soluções obtidas.

Hao et al. (1998) desenvolveram uma busca tabu enxuta para o BMCP. O espaço de busca do método consiste em soluções nas quais as restrições entre cores para um mesmo vértice são sempre respeitadas, porém a separação entre cores de vértices distintos pode ser violada. A vizinhança usada também envolve trocar as cores dos vértices. Quando a cor de um vértice é alterada em uma solução, um par indicando o vértice e a cor anterior é colocado na lista tabu, sendo o período tabu dinâmico.

Uma outra abordagem baseada em busca tabu foi proposta por Lai e Lü (2013), utilizando reinícios múltiplos (*multistart*). As soluções usadas permitem violações das restrições, sendo a inicial é aleatória. A vizinhança usada é a mesma dos dois algoritmos anteriores, com troca de cores nos vértices, mas com o uso de estratégias para avaliação rápida dos movimentos da busca local. O período tabu é dinâmico e proporcional ao custo da melhor solução (maior cor usada) e à frequência de atribuições da cor antiga ao vértice considerado e inversamente proporcional à maior frequência de atribuição de uma cor a um vértice. Por fim, para escapar de ótimos locais e permitir o *multistart*, é usado um critério de perturbação, no qual a cor de cada vértice é incrementada de acordo com um escopo de perturbação.

Outro método que usa busca tabu foi proposto por Lim et al. (2005), baseado também na meta-heurística *Squeaky Wheel Optimization* (SWO), onde sequências de vértices são usadas para definir prioridades e as soluções são melhoradas por meio de busca tabu em uma vizinhança de troca entre pares nas sequências de vértices.

Lim et al. (2005) desenvolveu um método envolvendo busca tabu e *Squeaky Wheel*

Optimization, onde sequências de vértices são usadas para definir prioridades e as soluções são melhoradas por meio de busca tabu em uma vizinhança de troca entre pares nas sequências de vértices. Tal método foi usado tanto com BCP e BMCP quanto com a coloração clássica.

Prestwich (2008) propôs um método híbrido combinando busca local e conceitos de programação por restrições. A vizinhança é a mesma usada na coloração clássica por Malaguti et al. (2008), onde uma solução vizinha é obtida ao colorir um vértice i com uma cor k e remover a cor de todos os vértices adjacentes a i que usem tal cor, com técnicas de verificação prévia para avaliar mais rapidamente os movimentos. Para a seleção de um vértice a ser colorido, são usados dois critérios: um baseado no DSATUR de Brélaz (1979) e outro baseado na seleção de vértices com mais possibilidades de cores. A seleção dos vértices a serem descoloridos parte de um parâmetro indicando quantos devem ter a cor removida. Por fim, para selecionar a cor de um vértice, as cores possíveis são ordenadas aleatoriamente e o algoritmo usa dois modos: tenta usar uma cor diferente da última que o vértice teve ou dá a preferência para a mesma cor, sendo que o método alterna entre os dois modos.

Chen et al. (2005) propuseram uma heurística envolvendo busca local cujo foco é na preservação da separação de cores de um mesmo vértice, possibilitando, porém, que as cores violem restrições com outros vértices. A solução inicial é obtida alocando, para cada vértice, uma cor aleatória e, para preencher suas demandas, cores adicionais que respeitem a restrição de separação para um mesmo vértice. Em seguida, cada vértice é processado, onde suas cores podem ser alteradas de modo que as restrições de separação do mesmo vértice continuem válidas, mas que a quantidade de conflitos com outros vértices seja a menor possível. A solução obtida, passa por um processo de ajuste fino, onde há a tentativa de remover os conflitos a partir uma visão global, com a identificação de cores atribuídas a vértices que encadeiam conflitos em outros vértices não adjacentes. Por fim, é usado um esquema de randomização para as próximas iterações do método, de modo que cada vértice tem seu intervalo de cores modificado aleatoriamente, mantendo as restrições de separação dos canais individuais para um mesmo vértice.

Kendall e Mohamad (2005) desenvolveram um método de busca local com critério de aceitação Monte Carlo exponencial, similar ao *simulated* annealing. O algoritmo trabalha no espaço de soluções válidas e inicia gerando uma permutação aleatória das demandas de

cores, obtendo uma solução a partir da mesma por meio do esquema FEA (Sivarajan et al., 1989). A geração de uma solução vizinha é feita removendo-se a cor da demanda onde a maior cor usada foi alocada, e uma nova cor é determinada verificando-se qual a menor cor possível que possa substituir a removida sem que nenhuma restrição seja violada. Se a nova solução for melhor, então ela é considerada a nova melhor, caso contrário, isso é feito com uma determinada probabilidade proporcional à diferença de soluções.

Chiarandini e Stützle (2007) propuseram algoritmos de busca local estocástica para o BMCP usando duas visões do problema: com o grafo original e o grafo estendido substituindo vértices por cliques de acordo com a demanda de cores. Na primeira abordagem, considera-se o conjunto de cores fixo ou variável enquanto na segunda apenas considera-se um número fixo de cores, permitindo conflitos somente entre diferentes vértices. São usadas duas vizinhanças: a básica, onde troca-se a cor de um vértice, e a de reatribuição de cores de um vértice, usada apenas com o grafo original, na qual uma solução vizinha é obtida ao trocar-se todas as cores de um vértice. Tais abordagens foram embutidas em busca tabu (baseada nos parâmetros usados por Galinier e Hao (1999) para a coloração clássica), busca local guiada (baseada na proposta por Lau e Tsang (1998)), algoritmo evolucionário híbrido (utilizando operadores do método de Galinier e Hao (1999)) e *squeaky wheel optimization* (seguindo a proposta de Lim et al. (2005)).

Lai et al. (2016) desenvolveram uma estratégia de reconexão de caminhos (*path relinking*) baseada em aprendizado para o BCP. O espaço de busca considerado é o de soluções conflitantes, de modo que a qualidade de uma solução é determinada pela quantidade de restrições violadas. O algoritmo utiliza uma população aleatória, onde a inicialização é feita por meio da geração do triplo de soluções a serem usadas na população, que serão melhoradas por busca tabu e, posteriormente, as melhores irão para a população. Uma segunda busca tabu também é usada na população, onde há duas fases: a primeira envolve a busca tabu básica com uma função de avaliação aumentada, de modo que, para cada aresta, é calculado um valor que representa quantas vezes a mesma foi conflitante em uma solução durante toda a execução do algoritmo, e a segunda fase repete mesmo procedimento, mas usando a função original sem o esquema aumentado. Por fim, partindo de duas soluções, são usados dois procedimentos de reconexão de caminhos: o primeiro muda a cor de um vértice da solução inicial, colocando a mesma cor da solução guiada, e o segundo esquema envolve mudanças nas duas soluções, sendo uma alteração na ini-

cial para a guiada e uma na guiada para a inicial, sendo que, ao tornarem-se iguais, o procedimento é concluído.

Algoritmos GRASP também foram propostos para o BCP e BMCP, principalmente no contexto de alocação de canais. Gomes et al. (2001) propuseram uma estratégia combinando GRASP reativo e reconexão de caminhos, na qual o grafo tem seus vértices estendido em cliques, e cada um dos mesmos tem uma lista proibida de cores. Na fase construtiva, os vértices são ordenados em ordem crescente de quantidade de cores disponíveis (inicialmente, todos os vértices podem usar qualquer cor) A lista restrita de candidatos consiste, então, em uma porção dos vértices iniciais da ordenação, da qual seleciona-se um e atribui-se ao mesmo a menor cor possível. As listas de cores bloqueadas dos vizinhos do vértice selecionado são atualizadas de acordo com possíveis violações das restrições. A busca local usada consiste em perturbar a solução atual, que será aceita caso seja melhor. Por sua vez, a perturbação envolve mudar as cores de todos os vértices, respeitando a separação de cores de um mesmo vértice. Por fim, a reconexão de caminhos envolve a seleção aleatória um vértice da solução inicial, o qual tem sua cor mudada para a usada na solução guiada, seguida de uma rápida busca local.

Wang e Rushforth (1996) propuseram uma busca local adaptiva para o BMCP, que foi posteriormente adaptada em um GRASP por Vieira et al. (2008) (chamado de GRASP-FEA). As soluções são codificadas como permutações das demandas de cores, ou seja, de vértices do grafo estendido em cliques, que serão transformadas em atribuições de cores pelo método FEA. A fase construtiva do algoritmo de Wang e Rushforth (1996) envolve uma função que mede a dificuldade de alocar uma cor a um vértice i A solução inicial, então, é uma ordenação dos vértices de acordo com tal função. Para a utilização deste esquema em um GRASP, Vieira et al. (2008) definiram uma LRC que consiste, então, em todo vértice cujo valor da função esteja a uma dada porcentagem do valor máximo. A cada vez que um vértice é selecionado, o mesmo é adicionado à solução (ou seja, à permutação de vértices construída) e a LRC é atualizada. A busca local consiste na troca de posição entre dois vértices na permutação que codifica a solução, a qual será decodificada pela estratégia FEA.

Marti et al. (2010) propuseram métodos combinando GRASP e busca tabu. A fase construtiva é similar à de Wang e Rushforth (1996), utilizando outra função de avaliação Os autores também introduzem memória à construção, que envolve uma medida de simi-

laridade entre cores e a distância da diferença entre as cords para o peso da aresta. São aplicadas duas buscas locais: a primeira consiste em remover a cor de todos os vértices que estejam usando a maior cor da solução e recolorir os mesmos com a menor cor possível (ou seja, que minimize conflitos), enquanto a segunda identifica vértices que forçam outros a usarem cores conflitantes, por meio de uma função de recorrência.

Diversos autores propuseram estratégias baseadas em algoritmos genéticos para o BCP/BMCP. Valenzuela et al. (1998) desenvolveram um método simples desse tipo, no qual as soluções são codificadas como permutações de vértices de modo que as alocações de cores são sempre factíveis e a função de *fitness* é a maior cor usada. A decodificação das soluções é feita de acordo com a regra FEA. A seleção de indivíduos segue o método da roleta, sendo que na primeira iteração também é usado um algoritmo com graus de saturação (semelhante ao DSATUR). A mutação é realizada trocando a posição de dois elementos em um indivíduo entre si. O cruzamento é feito pelo operador de cruzamento cíclico (*cyclic crossover* - CX), onde o primeiro gene do filho é o igual ao primeiro gene do pai, e então deve-se colocar o primeiro gene do segundo pai na posição onde o gene está no primeiro pai, seguindo tal procedimento até preencher o filho. Em seguida, partindo-se da última posição preenchida do filho, copia-se o gene de um pai para o filho, colocando-o a posição onde o mesmo está no outro pai.

Beckmann e Killat (1999) propuseram outro algoritmo genético para a coloração em largura de banda. As soluções também são codificadas como permutações de demandas de cores de vértices e geradas com a estratégia FEA, porém, há a imposição de um limite para o conjunto de cores, de modo que algumas demandas de cores podem não ser satisfeitas, cuja quantidade é usada na função de *fitness*. A seleção de soluções para os operadores de mutação é feita de forma aleatória. Para a mutação, são usados dois operadores: o primeiro seleciona uma demanda não satisfeita na permutação e a recoloca numa posição aleatória, enquanto a segunda seleciona quatro demandas e troca as mesmas de posição entre si aleatoriamente. A operação de cruzamento consiste na geração de um vetor binário de tamanho igual a um cromossomo, onde as posições com valor 1 farão com que as mesmos locus no filho gerado tenham os genes copiados do primeiro pai, enquanto os elementos restantes são preenchidos na ordem em que aparecem no segundo pai.

Kim et al. (2007) desenvolveram um algoritmo memético, combinando algoritmo genético e busca local. As soluções são codificadas como permutações de vértices e a deco-

dificação destas é feita por uma variação da regra FEA. A avaliação das soluções é feita pelo número de de demandas não satisfeitas. Se tal quantidade for zero, o conjunto de cores é reduzido na esperança de encontrar uma solução factível ainda melhor. A seleção de soluções é feita de modo aleatório e a mutação ocorre trocando genes de posição. O cruzamento é realizado gerando-se uma máscara de bits aleatória, de modo que são gerados dois filhos e, em cada um deles, o valor de cada bit da máscara indica de qual pai o gene correspondente será herdado. É usado um conjunto elite contendo as melhores soluções e por fim, a busca local usada consiste na troca do gene cuja cor é a maior de todas por outro aleatório, sendo que, se a nova solução for melhor, a mesma é mantida.

San José-Revuelta (2007) propôs um algoritmo micro-genético guiado por diversidade, cuja população é pequena. As soluções são codificadas de maneira similar ao método de Beckmann e Killat (1999) sendo o *fitness* igual à quantidade de conflitos na solução. A seleção de soluções para os operadores genéticos é feita por meio de uma roleta viciada, onde os indivíduos de melhor *fitness* tem maior probabilidade de serem selecionados, semelhante ao descrito no método de Beckmann e Killat (1999). A mutação, que ocorre com uma dada probabilidade, consiste em selecionar um vértice e mudar, aleatoriamente, metade das cores alocadas ao mesmo. Devido a possibilidade de mudanças muito bruscas na população, a probabilidade de mutação é mantida entre 0,01 (1%) e 0,05 (5%). O cruzamento consiste no operador PMX (*Partially Mapped Crossover*, ou Cruzamento Parcialmente Mapeado) utilizado na coloração clássica por Mabrouk et al. (2009) (visto na Seção C.3). O método usa elitismo simples, onde algumas das melhores soluções são preservadas para a geração seguinte. Já a diversidade é mantida calculando-se a entropia de Shannon da população, com o *fitness* de cada solução normalizado em relação a soma de todos os indivíduos. Quando tal valor se torna grande, há pouca diversidade, de modo que haverá menos cruzamentos e mais mutações.

Malaguti e Toth (2008) desenvolveram uma abordagem evolutiva que combina um algoritmo genético com busca tabu. As soluções usadas são colorações parciais onde deve-se colorir a maior quantidade possível de vértices sem que as restrições sejam violadas. O procedimento de busca tabu proposto pelos autores é usado durante a inicialização da população, bem como após o uso dos operadores genéticos, e é uma generalização, para o BCP, da busca tabu usada pelos mesmos autores para a coloração clássica (Malaguti et al., 2008). A inicialização da população consiste em gerar dois terços da mesma

por meio de métodos gulosos e o restante como soluções nas quais nenhum vértice está colorido, sendo que todas passarão pela busca tabu antes da inserção final na população. A seleção de soluções é aleatória e o cruzamento usado é chamado de *distance_crossover* e consiste em copiar, para o filho, as cores de pares de vértices do pai tais que a diferença absoluta entre elas seja igual à distância imposta na aresta entre eles. Em seguida, as cores do pai 2 são copiadas para preencher vértices não coloridos no filho.

Fijuljanin (2012) propôs dois algoritmos genéticos, sendo um com a adição de busca local. Ambos os algoritmos usam um método guloso para definir o conjunto de cores a serem atribuídas. Para o primeiro algoritmo genético, as soluções são codificadas do mesmo modo que Beckmann e Killat (1999). Assim, para cada locus, determina-se as cores livres e aloca-se a cor de acordo com o valor do gene. A função de *fitness* consiste na maior cor usada e a seleção de indivíduos é feita por meio da estratégia de torneio, onde alguns candidatos são escolhidos aleatoriamente e o melhor deles é selecionado. O cruzamento é simples de ponto único e A mutação consiste na mudança dos valores dos genes, sendo que os que sejam considerados congelados (ou seja, cujo valor é igual em todos os indivíduos) sofrem mutações mais fortes. O segundo algoritmo proposto é igual ao primeiro, com a mudança na função objetivo, onde será fixado um valor máximo de cor possível e o *fitness* contabilizará a quantidade de conflitos nas colorações e é adicionada uma busca local na qual, para cada vértice, tenta-se trocar uma de suas cores, sendo que a nova solução é aceita se gerar menos conflitos.

Uma variação das abordagens genéticas é a aprendizagem incremental baseada em população (*population-based incremental learning* - PBIL), onde a população explícita é substituída por um vetor de probabilidades (indicando a probabilidade de um gene ter um determinado valor), que é usado para obter amostrar a população. Esta abordagem foi usada por Chaves-González et al. (2008), na qual a solução inicial é obtida, primeiramente, atribuindo-se apenas uma cor para cada vértice, sem que as restrições sejam violadas. Em seguida, tal estrutura é duplicada, de modo que cada vértice tenha duas cores, sendo que, caso surjam conflitos, os mesmos são resolvidos incrementando-se todas as novas cores adicionadas até que não haja mais conflitos. Tal processo é repetido até que todas as demandas sejam atendidas e, por fim, como a duplicação é feita como se todos os vértices tivessem a mesma demanda, as cores desnecessárias alocadas a alguns vértices são removidas. A solução gerada é usada para criar o vetor de probabilidades

iniciais, que será usado, então, para geração de p amostras da população, que serão avaliadas de acordo com a maior cor usada. As soluções, então, são usadas para que o vetor de probabilidades tenda a gerar amostras cuja maior cor usada seja menor.

Uma outra abordagem também usada para coloração em largura de banda foi a otimização por colônia de formigas, que foi usada por Yin e Li juntamente com uma busca local. O algoritmo considera uma cor inicial e, então, cada formiga inicia em um vértice do grafo, efetuando um caminhamento pelo grafo, de acordo com as trilhas de feromônios, e atribuido a cor sempre que possível. Quando nenhum outro vértice puder usar tal cor, ela é incrementada e o processo repetido, até que todas as demandas de cores sejam satisfeitas. A busca local consiste na seleção de algumas cores de cada vértice, que serão removidas e, para cada uma delas, a menor cor possível que não viole restrições é alocada em seu lugar.

A Tabela 2.2 fornece um resumo das referências citadas nesta subseção em ordem cronológica de publicação.

2.4 Notas do capítulo

Nesse capítulo, foi apresentada uma revisão da literatura existente a respeito da coloração em largura de banda (BCP), bem como sua variante com multicoloração (BMCP). Optou-se por fazer uma revisão desse problema devido ao fato de que o BCP utiliza pesos nas arestas que envolvem distâncias, servindo de base aos modelos que serão discutidos posteriormente nesta tese.

Para o BCP, foram apresentados modelos de programação inteira e algoritmos exatos e aproximados, que servem para a construção de muitos métodos de solução. Espera-se que, além de base para os resultados obtidos na pesquisa, que a revisão realizada sirva como fonte de consulta para outros pesquisadores interessados no tema, sendo uma contribuição adicional desta tese.

Tabela 2.3: Algoritmos aproximados para a coloração em largura de banda (BCP).

Referência	Estratégia algorítmica
Sivarajan et al. (1989)	Heurísticas construtivas baseadas em ordenações dos vértices
Wang e Rushforth (1996)	Busca local adaptiva
Lau e Tsang (1998)	Busca local guiada (GLS)
Hao et al. (1998)	Busca tabu na vizinhança básica modificada
Valenzuela et al. (1998)	Algoritmo genético com soluções factíveis
Beckmann e Killat (1999)	Algoritmo genético com soluções conflitantes
Chakraborty (2001)	Heurística de geração de múltiplas soluções
Gomes et al. (2001)	GRASP reativo com reconexão de caminhos
Phan e Skiena (2002)	*Discropt - software* com meta-heurísticas gerais
Lim et al. (2005)	*Squeaky Wheel Optimization* (SWO)
Chen et al. (2005)	Busca local com preservação de restrições do mesmo vértice
Kendall e Mohamad (2005)	Busca local com aceitação Monte Carlo
Chiarandini e Stützle (2007)	Buscas locais estocásticas diversas
Kim et al. (2007)	Algoritmo memético com elitismo
San José-Revuelta (2007)	Algoritmo μ-genético guiado por diversidade com elitismo
Yin e Li	Colônia de formigas com busca local
Prestwich (2008)	Busca local com conceitos programação por restrições
Vieira et al. (2008)	GRASP-FEA: baseado na busca local de Wang e Rushforth (1996).
Malaguti e Toth (2008)	Algoritmo genético com busca tabu
Chaves-González et al. (2008)	Aprendizagem incremental baseada em população (PBIL)
Marti et al. (2010)	GRASP com memória e busca tabu
Fijuljanin (2012)	Dois algoritmos genéticos, sendo um com busca local
Lai e Lü (2013)	Busca tabu iterada com reinícios múltiplos
Lai et al. (2016)	Reconexão de caminhos baseada em aprendizado com busca tabu

Capítulo 3

Colorações em grafos com restrições de distâncias

Neste capítulo, serão apresentados os modelos teóricos dos problemas de colorações com distâncias, que são baseados em conceitos de teoria dos grafos e geometria de distâncias. Tais modelos generalizam restrições da coloração clássica e são baseados no BCP, que também é tratado sob essa visão neste capítulo.

Na dissertação de mestrado desenvolvida pelo autor desta tese (Dias, 2014), foram propostos modelos teóricos para coloração de vértices baseados em geometria de distâncias, de modo a tornar possível o uso do ferramental desse campo de estudo para modelar alguns cenários específicos de alocação de canais como coloração em grafos. Sendo assim, será apresentado o modelo em geometria de distâncias para colorações para que, em seguida, sejam mostrados os resultados obtidos para os mesmos nesta tese.

3.1 Modelo teórico em geometria de distâncias

O BCP, por definição, requer que as cores atribuídas a vértices adjacentes respeitem uma separação mínima determinada por um peso na aresta entre os mesmos, isto é, para dois vértices i e j, tem-se que $|c(i) - c(j)| \geq d(i, j)$. Esse tipo de separação é um tipo de restrição de distância, o que possibilita o uso de conceitos de geometria de distâncias para modelagem e solução dos mesmos.

O Problema de Geometria de Distâncias Discretizáveis (*Discretizable Distance Geometry Problem* - DDGP) é um caso especial do problema fundamental de geometria de

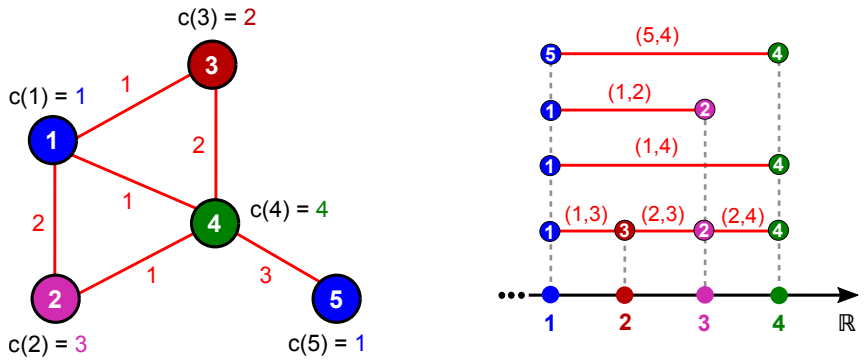


Figura 3.1: Exemplo de instância resolvida do BCP com a solução representada por meio de 0-esferas (segmentos de reta).

distâncias. No DDGP em três dimensões (ou apenas DDGP$_3$), o conjunto V de vértices do grafo de entrsda G deve ser ordenado de modo que exista um subconjunto V_0 de V tal que $|V_0| = 3$, V_0 forme uma clique e para cada $i \in V \backslash V_0$ exista um subconjunto $\{v_1, v_2, v_3\}$ de V tal que $v_1 < i,\ v_2 < i,\ v_3 < i$; $\{(v_1, i),\ (v_2, i),\ (v_3, i)\} \subseteq E$ e uma desigualdade triangular estrita seja válida entre $\{v_1, v_2, v_3\}$, ou seja, $d_{v_1,v_3} < d_{v_1,v_2} + d_{v_2,v_3}$ (Mucherino et al., 2012). Desse modo, a posição de um ponto $i \geq 4$ pode ser determinada usando as posições dos três pontos anteriores $i-1, i-2$ e $i-3$ por meio da intersecção de esferas com raios $d_{i-3,i}, d_{i-2,i}$ and $d_{i-1,i}$, obtendo dois possíveis pontos cujas factibilidades deverão ser verificadas. (Lavor et al., 2012).

Pode-se modelar problemas de coloração em grafos como instâncias do DDGP em uma dimensão (DDGP$_1$), onde a distância entre dois pontos equivale à diferença absoluta dos mesmos (Dias et al., 2013b). Desse modo, para colorações em grafos, tem-se que deve existir um subconjunto V_0 de V tal que $V_0 = 1$, V_0 forme uma clique (o que é sempre verdade, pois um único vértice pode ser considerado uma clique) e, para cada $i \in V \backslash V_0$, exista um subconjunto $\{v_1\}$ of V (ou seja, um outro vértice) tal que $v_1 < i$ e $(v_1, i) \in E$ (que será sempre verdade se o grafo for conexo). A desigualdade triangular estrita não se aplica nesse caso, pois considera-se apenas dois vértices em uma dimensão. A posição de um vértice i pode ser determinada por meio de um vizinho j já posicionado. Assim, forma-se uma 0-esfera, ou seja, um segmento de reta, cujo raio (ou seja, o comprimento do segmento de reta) é $d(i, j)$. Desse modo, colorações factíveis consistem em determinar intersecções de tais 0-esferas. A Figura 3.1 mostra um exemplo de instância do BCP com sua solução representada como intersecção de tais segmentos de reta.

Com base na discussão acima, define-se o seguinte modelo básico de coloração com distâncias.

Definição 3.1.1. ***Problema de Coloração em Geometria de Distâncias (Coloring Distance Geometry Problem - CDGP)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e uma função de distâncias* $d : E \to \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i, j) \in E,\ |x(i) - x(j)| \boxtimes d(i, j)$*, sendo* $\boxtimes$ *um operador binário de comparação.*

O CDGP é um modelo genérico, uma vez que o operador que compara a diferença entre cores de vértices adjacentes com a distância imposta na aresta entre os mesmos não é explicitamente definido. Sendo assim, partindo-se de tal problema, pode-se explorar diversas generalizações de restrições de adjacência, de acordo com diferentes operadores de comparação.

Formulações de programação inteira e por restrições para os modelos teóricos de coloração de grafos em geometria de distâncias serão apresentadas no Capítulo 4.

3.2 Generalizações das restrições de adjacência

Como citado anteriormente, o CDGP por si só é um modelo básico de coloração em geometria de distâncias, uma vez que o operador de comparação de distância não é explicitamente definido. Nesta seção, serão exploradas três possibilidades para $\boxtimes$ no CDGP: desigualdade $\geq$, igualdade ($=$) e desigualdade $\leq$. Cada um dos operadores leva a diferentes características relacionadas à existência ou não de projeções (ou seja, de soluções factíveis), de acordo com a estrutura do grafo, como será apresentado a seguir.

3.2.1 Restrições de desigualdade $\geq$

Um dos modelos de coloração com distâncias apresentado em Dias (2014) envolve restrições de desigualdades do tipo $\geq$, como descrito a seguir.

Definição 3.2.1. ***Problema de Coloração Mínima em Geometria de Distâncias Maiores ou Iguais (Minimum Greater than or Equal Distance Geometry Problem - MinGEQ-CDGP)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e uma função de distâncias* $d : E \to \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i, j) \in E,\ |x(i) - x(j)| \geq d(i, j)$ *e a extensão da projeção, definida por* $S = \max_{i \in V} x(i)$*, seja a menor possível.*

O modelo MinGEQ-CDGP é equivalente ao BCP e ao MS-CAP, utilizando o mesmo tipo de restrição e as mesmas definições. Tal problema sempre admite ao menos uma

coloração factível, pois ao contrário do EQ-CDGP, onde há apenas um valor possível para a diferença absoluta entre cores de vértices adjacentes i e j, equivalente à distância $d(i,j)$ imposta entre os mesmos, enquanto que, no MinGEQ-CDGP (e BCP), há infinitas possibilidades bastando que, para dois vértices i e j, o valor de tal diferença seja maior ou igual à distância $d(i,j)$. Uma versão do mesmo com distâncias uniformes é apresentada abaixo.

Definição 3.2.2. ***Problema de Coloração Mínima em Geometria de Distâncias Uniformes Maiores ou Iguais (Minimum Greater than or Equal Uniform Distance Geometry Problem - MinGEQ-CDGP-Unif)****: Dado um grafo simples ponderado não direcionado $G = (V,E)$ e um número $\varphi \in \mathbb{N}$, encontrar uma projeção $x : V \to \mathbb{N}$ tal que $\forall (i,j) \in E,\ |x(i) - x(j)| \geq \varphi$ e a extensão da projeção, definida por $S = \max_{i \in V} x(i)$, seja a menor possível.*

O MinGEQ-CDGP-Unif é equivalente ao problema de coloração clássica. Independentemente do valor de φ, uma solução para o problema de coloração clássica induzida pelo grafo G pode ser transformada em solução para o MinGEQ-CDGP-Unif definido por G e φ definindo-se $x(i) = (\varphi \times (c(i) - 1)) + 1$, onde $c(i)$ é a cor atribuída ao vértice i na solução do problema de coloração clássica. De modo similar, uma solução para o MinGEQ-CDGP-Unif definido por G e φ pode ser transformada em uma para a coloração clássica considerando-se $c(i) = \frac{x(i)-1}{\varphi} + 1$.

3.2.2 Restrições de igualdade

Um outro modelo de coloração com distâncias envolve restrições de igualdade nas arestas, ao invés de desigualdades do tipo $\geq$, mantendo-se os pesos nas arestas definidos como números naturais do mesmo modo do BCP. A definição do problema de decisão associado a esse cenário é dada abaixo.

Definição 3.2.3. ***Problema de Coloração em Geometria de Distâncias Iguais (Equal Distance Geometry Problem - EQ-CDGP)****: Dado um grafo simples ponderado não direcionado $G = (V,E)$ e uma função de distâncias $d : E \to \mathbb{N}$, encontrar uma projeção $x : V \to \mathbb{N}$ tal que $\forall (i,j) \in E,\ |x(i) - x(j)| = d(i,j)$.*

A Figura 3.2 mostra um exemplo do modelo e sua representação por 0-esferas. Tal problema é NP-completo, como mostrado a seguir.

Teorema 3.1. *EQ-CDGP é NP-completo.*

Prova. Para provar que EQ-CDGP é NP-completo, deve-se mostrar que o problema é tanto NP quanto NP-difícil, o que será realizado em duas etapas.

1. EQ-CDGP é NP.

Para um grafo $G = (V,E)$, dada uma projeção $x : V \to \mathbb{N}$, sua factibilidade pode ser verificada considerando cada aresta $(i, j) \in E$ e examinando se as cores de suas extremidades não violam a restrição de distância correspondente, isto é, se $|x(i) - x(j)| = d(i, j)$. Se todas as restrições forem satisfeitas, então x é um certificado para uma resposta positiva para a instância do EQ-CDGP, o que significa que um certificado para uma solução SIM pode ser verificado em tempo $O(|E|)$, que é linear. Portanto, EQ-CDGP é NP.

2. EQ-CDGP é NP-difícil.

Para provar isto, será feita uma redução do problema de particionamento de conjuntos, que é NP-completo (Garey e Johnson, 1979), para o EQ-CDGP. Considere uma instância do problema de particionamento, consistindo em um conjunto M de r inteiros, ou seja, $M = \{m_1, m_2, \ldots, m_r\}$. A partir de M, pode-se obter um grafo $G = (V,E)$, de modo que G seja um ciclo, $|V| = |E| = r$ e exista uma função $d : E \to \mathbb{N}$ de ponderamento de arestas. O grafo G e a função d são dados então pela seguinte construção:

- $V = \{i_0, i_1, \ldots, i_{r-1}\}$.
- $E = \{(i_b, i_{b+1 \bmod r}) \mid 0 \leq b \leq r\}$.
- $d(i_b, i_{b+1 \bmod r}) = m_b$ $(\forall 0 \leq b \leq r)$.

Sendo assim, deve-se demonstrar que a instância M do particionamento de conjuntos tem solução SIM se, e somente se, o grafo ciclo G construído acima e sua função de ponderação de arestas d também tiverem solução SIM, como feito a seguir.

($\rightarrow$) Seja então $x : V \to \mathbb{N}$ uma projeção dos vértices na reta numérica. Se a mesma for válida, então pode-se definir dois conjuntos:

- $S_1 = \{m_b \mid x(i_b) < x(i_{b+1 \bmod r})\}$.
- $S_2 = \{m_b \mid x(i_b) > x(i_{b+1 \bmod r})\}$.

Tem-se então que S_1 e S_2 são subconjuntos disjuntos de M (ou seja, eles formam um particionamento de M), de modo que $\sum S_1 = \sum S_2$, ou seja, se o grafo G possuir uma projeção na reta numérica (o que significa que a solução do EQ-CDGP é SIM), então M tem solução SIM para o problema de particionamento.

($\leftarrow$) Considere um grafo ciclo $G = (V,E)$, onde $|V| = |E| = r$ e cujas arestas são ponderadas por uma função $d : E \to \mathbb{N}$. Fixando-se um dos vértices como i_0 e sendo $V = \{i_0, i_1, \ldots, i_{r-1}\}$, tem-se que $E = \{(i_b, i_{b+1 \bmod r}) \mid 0 \leq b \leq r\}$ (ou seja, vértices consecutivos possuem uma aresta entre si). Sendo M o conjunto de todos os valores de d (ou seja, $M = \bigcup_{0 \leq b \leq r} d(i_b, i_{b+1 \bmod r})$), considere um particionamento de M em dois subconjuntos disjuntos S_1 e S_2. Se $\sum S_1 = \sum S_2$, então pode-se obter uma projeção $x : V \to \mathbb{N}$ considerando-se:

- Se $d(i_b, i_{b+1 \bmod r}) \in S_1$, então $x(i_b) < x(i_{b+1 \bmod r})$.
- Se $d(i_b, i_{b+1 \bmod r}) \in S_2$, então $x(i_b) > x(i_{b+1 \bmod r})$.

Sendo assim, basta atribuir um número natural para cada $x(i_b)$ que satisfaça tais condições (sendo que cada i_b deverá levar em conta apenas os valores atribuídos a dois outros vértices, i_{b-1} e $i_{b+1 \bmod r}$. Logo, se M possuir solução SIM para o problema de particionamento, então o grafo G terá uma projeção válida na reta numérica, levando a uma solução SIM para o EQ-CDGP.

Tem-se então que existe uma redução do problema de particionamento de conjuntos para o EQ-CDGP, que pode ser feita em tempo $O(r)$. Portanto, EQ-CDGP é NP-difícil.

Como o EQ-CDGP é NP e NP-difícil, conclui-se então que o mesmo é NP-completo. □

A versão de otimização do EQ-CDGP, onde deve-se encontrar uma projeção que minimize a utilização de cores, é definida abaixo.

Definição 3.2.4. ***Problema de Coloração Mínima em Geometria de Distâncias Iguais (Minimum Equal Distance Geometry Problem - MinEQ-CDGP)****: Dado um grafo simples ponderado não direcionado $G = (V,E)$ e uma função de distâncias $d : E \to \mathbb{N}$, encontrar uma projeção $x : V \to \mathbb{N}$ tal que $\forall (i, j) \in E,\ |x(i) - x(j)| = d(i, j)$ e a extensão da projeção, definida por $S = \max_{i \in V} x(i)$, seja a menor possível.*

Um caso especial do EQ-CDGP (e de sua versão de otimização) substitui a função de distâncias por um único valor, conforme definido a seguir.

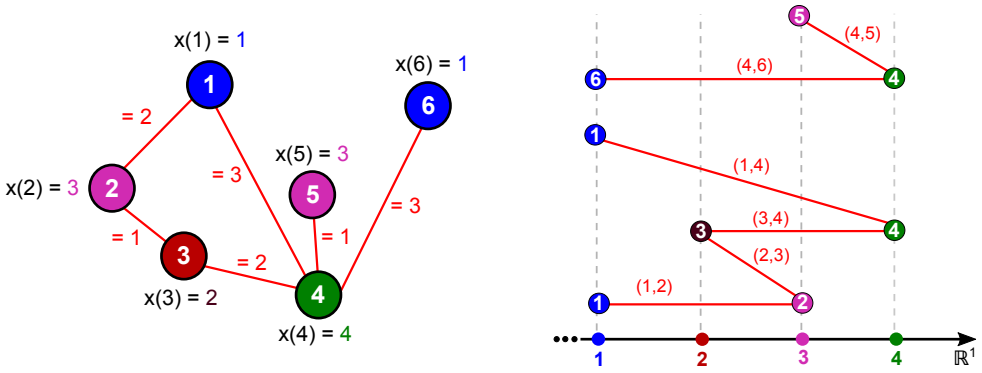


Figura 3.2: Instância do MinEQ-CDGP resolvida e sua representação com 0-esferas.

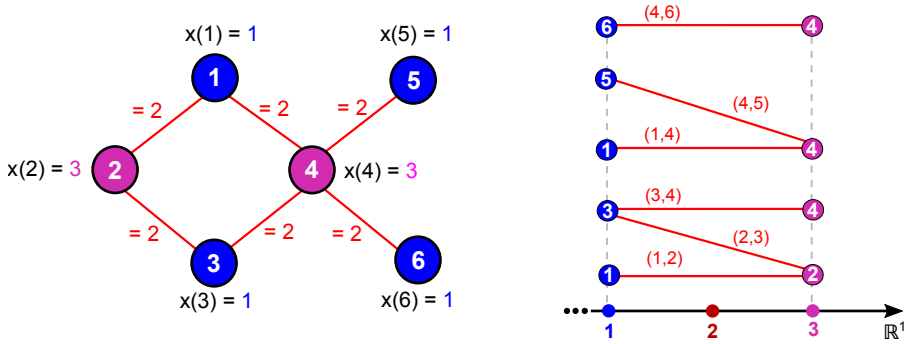


Figura 3.3: Instância do EQ-CDGP-Unif resolvida e sua representação com 0-esferas.

Definição 3.2.5. ***Problema de Coloração em Geometria de Distâncias Uniformes Iguais (Equal Uniform Distance Geometry Problem - EQ-CDGP-Unif)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e um número* $\varphi \in \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i,j) \in E,\ |x(i) - x(j)| = \varphi$.

A Figura 3.3 exemplifica esse caso especial com distâncias uniformes. A versão de otimização correspondente do EQ-CDGP-Unif é enunciada abaixo.

Definição 3.2.6. ***Problema de Coloração Mínima em Geometria de Distâncias Uniformes Iguais (Minimum Equal Uniform Distance Geometry Problem - MinEQ-CDGP-Unif)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e um número* $\varphi \in \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i,j) \in E,\ |x(i) - x(j)| = \varphi$ *e a extensão da projeção, definida por* $S = \max_{i \in V} x(i)$*, seja a menor possível.*

3.2.2.1 Propriedades de factibilidade

Nesta tese, foram identificadas propriedades grafo-teóricas dos problemas acima, em específico, classes de grafos onde a solução dos problemas de decisão (EQ-CDGP ou EQ-CDGP-Unif) é SIM (equivalentemente, onde há ao menos uma solução factível para o

MinEQ-CDGP ou MinEQ-CDGP-Unif). A seguir, serão enunciados os teoremas e suas respectivas provas.

O primeiro teorema caracteriza quais grafos que possuem solução SIM para o EQ-CDGP-Unif, como pode ser visto a seguir.

Teorema 3.2. *Seja $G = (V,E)$ um grafo não direcionado. Então, G possui solução SIM para o EQ-CDGP-Unif se, e somente se, G for bipartido.*

Prova. ($\rightarrow$) Considere que G é bipartido e, sem perda de generalidade, que também é conexo. Sejam V_1 e V_2 dois subconjuntos disjuntos de V (ou seja, $V_1 \cup V_2 = V$) tais que para todo $i, j \in V_1$ ou $i, j \in V_2$ tem-se que $(i, j) \notin E$, que podem ser obtidos pelo uso de um algoritmo de busca em grafos, como a busca em profundidade (*depth-first search* - DFS). Como o grafo é bipartido, tem-se, para cada $(i, j) \in E$, que i e j pertencem a subconjuntos diferentes. Assuma, sem perda de generalidade, que $i \in V_1$ e $j \in V_2$. Pela definição do EQ-CDGP-Unif, tem-se que $d(i, j) = \varphi$ para toda aresta $(i, j) \in E$. Logo, pode-se atribuir a cor 1 a todos os vértices de V_1 e a cor $1 + \varphi$ a todos os vértices de V_2. Desse modo, as restrições de distâncias serão satisfeitas e a solução gerada é factível. Pode-se concluir, então, que se G é bipartido, então G possui solução SIM para o EQ-CDGP-Unif.
($\leftarrow$) Seja $x : V \rightarrow \{1, 1 + \varphi\}$ uma coloração factível para o EQ-CDGP-Unif de um grafo G. Assuma, sem perda de generalidade, que G é conexo. Pela definição do EQ-CDGP, para cada par de vértices $i, j \in V$ tais que $x(i) = x(j)$, tem-se que $(i, j) \notin E$. Portanto, pode-se dividir os vértices de G em dois subconjuntos disjuntos V_1 e V_2, onde $V_1 = \{i \in V | x(i) = 1\}$ e $V_2 = \{i \in V | x(i) = 1 + \varphi\}$. Assim, conclui-se que se G tem solução SIM para o EQ-CDGP-Unif, então o grafo é bipartido.

Prova alternativa. ($\rightarrow$) Considere que G é bipartido. Assim, tem-se que o mesmo é 2-colorível (considerando o problema de coloração clássica), isto é, o grafo todo pode ser colorido usando apenas duas cores. Levando-se em conta, agora, as restrições do EQ-CDGP-Unif, todas as arestas do grafo tem a mesma distância, ou seja, para toda aresta $(i, j) \in E$, tem-se a restrição $|x(i) - x(j)| = \varphi$. Portanto, as duas cores usadas serão {1, 1+φ}, que compõem a solução da instância do EQ-CDGP-Unif. Logo, se G é bipartido, então a solução correspondente do EQ-CDGP-Unif é SIM.
($\rightarrow$) Para provar-se que, se um grafo G tem solução SIM para o EQ-CDGP-Unif, então G é bipartido, será usada uma prova por contraposição, ou seja, será mostrado que se um

grafo G não for bipartido, então G possui solução NÃO para o EQ-CDGP-Unif. Para isso, será usada indução matemática em ciclos ímpares, uma vez que um grafo não é bipartido se, e somente se, contiver pelo menos um ciclo ímpar. Seja $|V| = 2z+1$. A indução será feita em cima de z.

Caso base: $z = 1$. Tem-se, então, o ciclo C_3, com três vértices ($V = \{1,2,3\}$) e três arestas ($\{(1,2),(1,3),(2,3)\}$), com $|x(i) - x(j)| = \varphi$ para todas elas. Sem perda de generalidade, considere que $x(1) = 1$ e $x(2) = 1+\varphi$. Logo:

- Como $(1,3) \in E$ e $x(1) = 1$, então $|x(3) - 1| = \varphi$. Todas as cores devem ser números naturais, portanto $x(3) = 1+\varphi$.
- Como $(2,3) \in E$ e $x(2) = 1+\varphi$, então $|x(3) - (1+\varphi)| = \varphi \;\Rightarrow\; |x(3) - 1 - \varphi| = \varphi$. Por esta inequação, tem-se que $x(3) = 1$ ou $x(3) = 1+2\varphi$.

A partir desse resultado, tem-se que $x(3) = 1+\varphi$ e ($x(3) = 1$ ou $x(3) = 1+2\varphi$) ao mesmo tempo, o que é impossível. Assim, C_3 tem solução NÃO para o EQ-CDGP-Unif, como pode ser visualizado na Figura 3.4.

Hipótese de indução: O ciclo C_{2z+1} tem solução NÃO para o EQ-CDGP.

Passo indutivo: Pela hipótese de indução, o ciclo C_{2z+1} não possui coloração válida para o EQ-CDGP-Unif, isto é, a solução do problema de decisão é NÃO. Considerando o ciclo $C_{2(z+1)+1} = C_{2z+3}$, verifica-se que o tamanho do mesmo aumenta em dois vértices, mas continua sendo ímpar. Se for adicionado apenas um vértice, ou seja, considerando o ciclo $C_{2z+1+1} = C_{2z+2}$, tem-se então um ciclo par, que é bipartido e terá solução SIM para o EQ-CDGP-Unif como visto anteriormente. Agora, considere que um vértice é adicionado ao ciclo C_{2z+2}, transformando-o no ciclo C_{2z+3}. Sem perda de generalidade, considere que o novo vértice é adjacente aos vértices $2z+2$ e 1, ou seja, $\{(2z+2,2z+3),(2z+3,1)\} \subseteq E$, e $x(2z+2) = 1+\varphi$ e $x(1) = 1$ (tais cores podem ser vistas como atribuídas quando adicionou-se apenas um vértice, o que gerou um ciclo par). Então:

- Como $(2z+2,2z+3) \in E$ e $x(2z+2) = 1+\varphi$, então $|x(2z+3) - (1+\varphi)| = \varphi \;\Rightarrow$ $|x(2z+3) - 1 - \varphi| = \varphi$. Por esta inequação, tem-se que $x(2z+3) = 1$ ou $x(2z+3) = 1+2\varphi$.
- Como $(2z+3,1) \in E$ and $x(1) = 1$, então $|x(2z+3) - 1| = \varphi$. Todas as cores devem ser números naturais, portanto $x(2z+3) = 1+\varphi$.

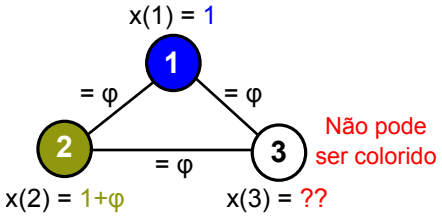


Figura 3.4: Ciclo C_3 que tem solução NÃO para o EQ-CDGP-Unif.

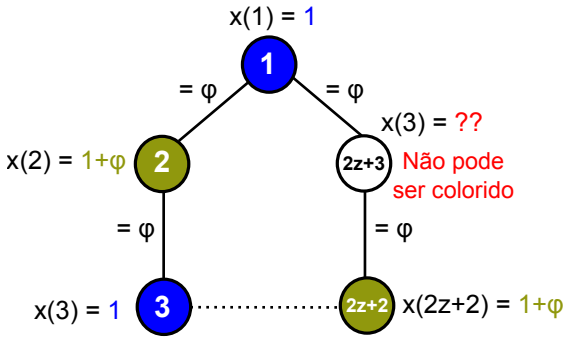
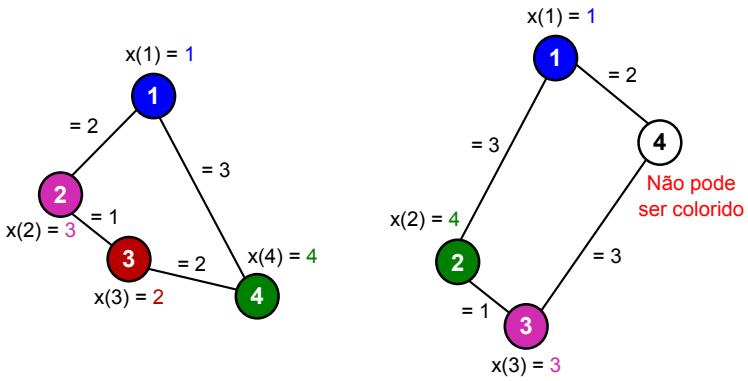


Figura 3.5: Ciclo ímpar C_{2z+3} que tem solução NÃO para o EQ-CDGP-Unif.

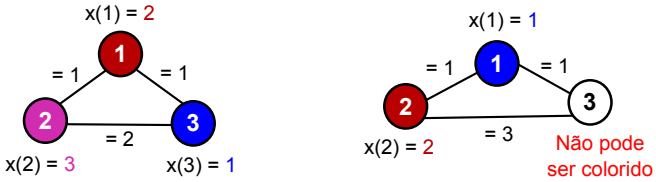
Deste resultado, tem-se que $x(2z+3) = 1+\varphi$ e ($x(2z+3) = 1$ ou $x(2z+3) = 1+2\varphi$) ao mesmo tempo, o que é impossível. Assim, C_{2z+3} tem solução NÃO para o EQ-CDGP-Unif, como pode ser visto na Figura 3.5. Portanto, se um grafo G não for bipartido, então G possui solução NÃO para o EQ-CDGP-Unif, ou seja, se um grafo G tem solução SIM para o EQ-CDGP-Unif, então G é bipartido. □

Uma consequência indireta do Teorema 3.2 é que não se pode definir modelos de multicoloração que envolvam as mesmas restrições do EQ-CDGP. Como visto no Capítulo 2, quando há, em um vértice i, demanda de $w(i)$ cores ($w(i) > 1$), há uma equivalência em colorir um grafo onde o vértice é convertido em uma clique de $w(i)$ vértices. Como uma clique é um subgrafo completo e todo grafo completo K_n com $n \geq 3$ contém ciclo ímpar, então os únicos valores possíveis para $w(i)$ seriam 1 e 2, limitando a demanda de cores. Ressalta-se que essa característica vale somente se as restrições de igualdade também existirem em tal clique, sendo que se as cores do mesmo vértice tiverem apenas de ser distintas, então não há esse problema.

O Teorema 3.2 não é aplicável em instâncias do EQ-CDGP, ou seja, com distâncias arbitrárias. Dependendo dos valores de tais distâncias, grafos bipartidos podem ter solução NÃO para o EQ-CDGP, enquanto grafos que contenham ciclos ímpares podem ter solução SIM, como exemplificado na Figura 3.6. Entretanto, o problema de decisão EQ-CDGP sempre tem solução SIM para árvores, como visto abaixo.



(a) Grafos bipartidos.



(b) Grafos com ciclo ímpar.

Figura 3.6: Exemplos de instâncias do EQ-CDGP onde a solução das mesmas (SIM ou NÃO) independe do tipo de grafo de entrada (bipartido ou não).

Teorema 3.3. *Seja $G = (V, E)$ um grafo árvore e $d : E \to \mathbb{N}$ uma função de distâncias. Então, G e d sempre têm solução SIM para o EQ-CDGP.*

Prova. Um algoritmo simples que constrói uma coloração válida para uma instância do EQ-CDGP é descrito a seguir.

Inicialmente, desmarque todos os vértices. Escolha um vértice i arbitrário, atribua uma cor $\gamma \in \mathbb{N}$ ao mesmo (ou seja, $x(i) = \gamma$) e marque i como colorido. Iterativamente, selecione um vértice j que seja adjacente a algum outro vértice i que já tenha sido marcado como colorido. Atribua a cor $x(i) + d(i, j)$ a j (ou seja, $x(j) = x(i) + d(i, j)$) e marque j como colorido. Repita o passo iterativo até que todos os vértices sejam marcados como coloridos. □

O algoritmo descrito na prova acima consiste, basicamente, em seguir os ramos da árvore, colorindo cada vértice encontrado. Como, após o primeiro vértice, só serão coloridos vértices que tenham ao menos um vizinho já colorido, o procedimento garante que o caminhamento por tais ramos será feito corretamente. Em grafos caminho, o algoritmo começará de um vértice de grau 1 (ou seja, uma das extremidades) e seguirá os outros vértices em sequência, até atingir a outra extremidade, como pode ser visto na Figura 3.7. Em árvores propriamente ditas, cada ramo é um caminho, como visto na Figura 3.8. Tal algoritmo pode ser visto como uma extensão da busca em grafos, de modo que tanto a

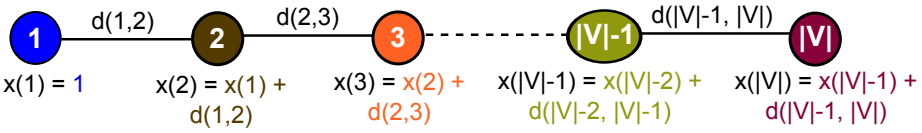


Figura 3.7: Coloração de um grafo caminho por meio do algoritmo do Teorema 3.3.

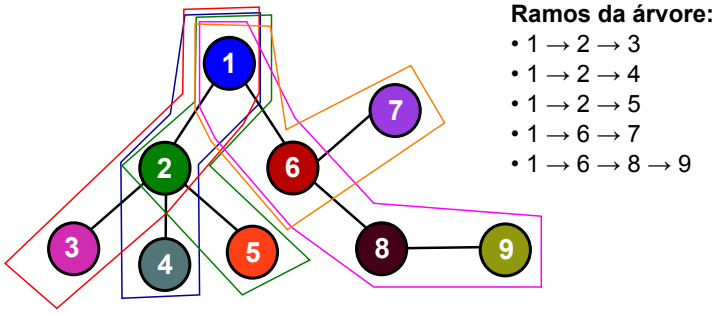


Figura 3.8: Ramos de uma árvore. Neste grafo, os rótulos indicam a ordem de visitação de uma busca em profundidade adaptada para colorir os vértices de acordo com o Teorema 3.3.

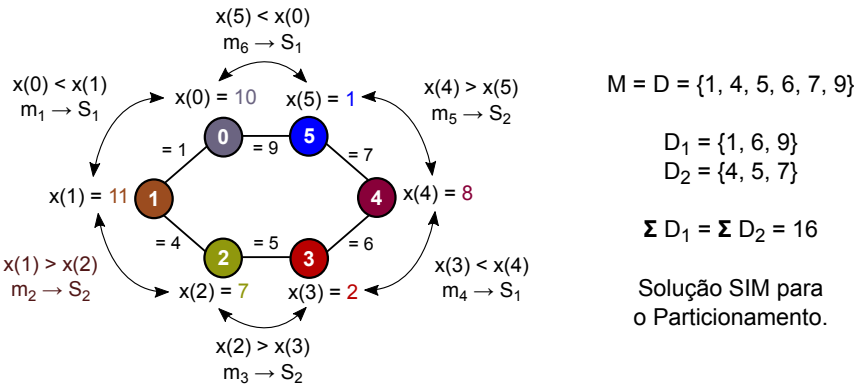
busca em profundidade quanto a busca em largura (*breadth-first search* - BFS) podem ser modificados para que sejam capazes de obter uma coloração válida para o EQ-CDGP em um árvore.

O último resultado a ser apresentado nesta seção relaciona-se a ciclos, quando são instâncias do EQ-CDGP. Tal resultado é derivado diretamente da prova da NP-dificuldade do problema, como vista no Teorema 3.1. O resultado de factibilidade correspondente é descrito a seguir.

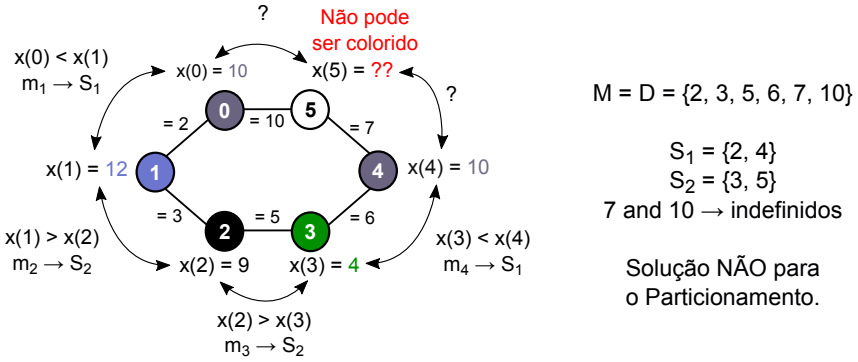
Corolário 3.1. *Seja $G = (V,E)$ um grafo ciclo, $d : E \to \mathbb{N}$ uma função de distâncias e D o conjunto de todas as distâncias presentes no grafo (ou seja, $D = \bigcup_{(i,j)\in E} d(i,j)$). Se for possível particionar D em dois subconjuntos disjuntos D_1 e D_2 tais que $\sum D_1 = \sum D_2$, então G e d sempre têm solução SIM para o EQ-CDGP.*

Prova. Diretamente pelo Teorema 3.1. □

O Corolário 3.1 caracteriza um subconjunto de instâncias do EQ-CDGP, compostas por grafos ciclo, que possuem solução SIM para o problema. Conjectura-se que, quando o grafo da instância entrada é um ciclo, apenas os que se encaixam no Corolário 3.1 possuem solução SIM. Ressalta-se que, apesar de este resultado ser específico para um tipo de grafo, os valores numéricos das distâncias são diretamente responsáveis pela factibilidade do referido ciclo, como mostrado na Figura 3.9. Infelizmente, este resultado, ao contrário dos obtidos nos Teoremas 3.2 e 3.3, não permite que o EQ-CDGP seja resolvido



(a) Instância com solução SIM.



(b) Instância com solução NÃO.

Figura 3.9: Instâncias do problema de Particionamento de Conjuntos e suas transformações para o EQ-CDGP.

em tempo polinomial para as instâncias especificadas, uma vez que o próprio problema de particionamento de conjuntos também é NP-completo Karp (1972).

3.2.3 Restrições de desigualdade $\leq$

Além dos problemas acima definidos em Dias (2014) e suas propriedades obtidas nesta tese, explorou-se outras possibilidades de colorações com distâncias. Uma delas envolve a inversão da desigualdade do BCP, de modo que, ao invés de que a diferença absoluta entre cores de dois vértices adjacentes tenha de ser maior ou igual que a distância entre eles, tal diferença terá de ser menor ou igual que a distância. O problema, em sua versão de decisão, é definido abaixo.

Definição 3.2.7. ***Problema de Coloração em Geometria de Distâncias Menores que ou Iguais (Less than or Equal Distance Geometry Problem - LEQ-CDGP)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e uma função de distâncias* $d : E \to \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i, j) \in E,\ 1 \leq |x(i) - x(j)| \leq d(i, j)$.

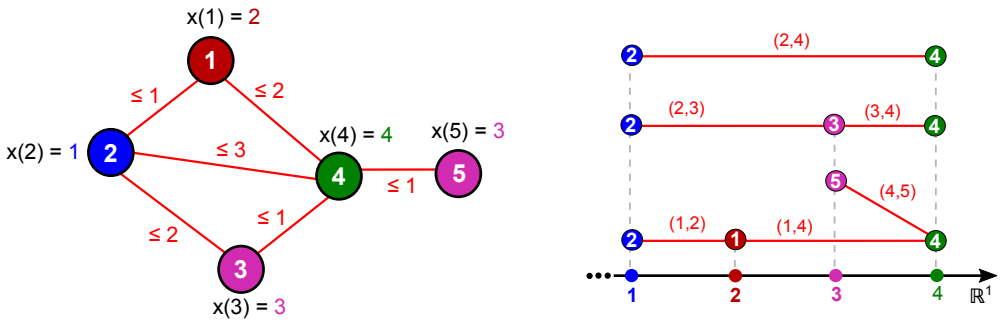


Figura 3.10: Instância do LEQ-CDGP resolvida e sua representação com 0-esferas.

A Figura 3.10 mostra um exemplo do LEQ-CDGP. Para este tipo de restrição, ressalta-se que não basta definir apenas $|x(i) - x(j)| \leq d(i,j)$ para dois vértices adjacentes i e j, uma vez que, sendo $d(i,j)$ um número natural, tal expressão permitiria que $|x(i) - x(j)| = 0$, o que equivale aos dois vértices utilizando a mesma cor. Sendo assim, deve-se definir o valor mínimo 1 para a diferença absoluta entre cores. Por conta disso, o modelo de decisão se torna importante do mesmo modo que com as restrições de igualdade do EQ-CDGP, uma vez que há uma quantidade limitada de possibilidades para a diferença absoluta entre cores de dois vértices (ao contrário do BCP), o que implica diretamente em resultados de factibilidade, como será visto mais adiante.

A versão de otimização do LEQ-CDGP, envolvendo a minimização da maior cor usada, é dada abaixo.

Definição 3.2.8. ***Problema de Coloração Mínima em Geometria de Distâncias Menores que ou Iguais (Minimum Less than or Equal Distance Geometry Problem - MinLEQ-CDGP)****: Dado um grafo simples ponderado não direcionado $G = (V,E)$ e uma função de distâncias $d : E \to \mathbb{N}$, encontrar uma projeção $x : V \to \mathbb{N}$ tal que $\forall (i,j) \in E,\ 1 \leq |x(i) - x(j)| \leq d(i,j)$ e a extensão da projeção, definida por $S = \max_{i \in V} x(i)$, seja a menor possível.*

Como já feito com os outros modelos, um caso especial deste problema envolve distâncias uniformes, isto é, todas as arestas possuem a mesma distância. Tal cenário é formalmente descrito a seguir.

Definição 3.2.9. ***Problema de Coloração em Geometria de Distâncias Uniformes Menores que ou Iguais (Less than or Equal Uniform Distance Geometry Problem - LEQ-CDGP-Unif)****: Dado um grafo simples ponderado não direcionado $G = (V,E)$ e um número $\varphi \in \mathbb{N}$, encontrar uma projeção $x : V \to \mathbb{N}$ tal que $\forall (i,j) \in E,\ 1 \leq |x(i) - x(j)| \leq \varphi$.*

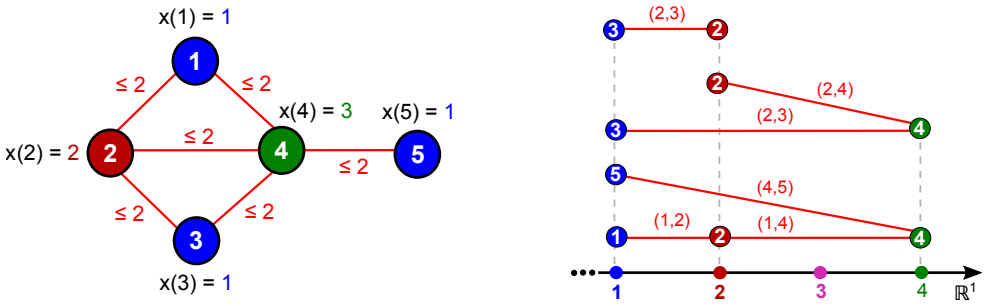


Figura 3.11: Instância do LEQ-CDGP-Unif resolvida e sua representação com 0-esferas.

A Figura 3.10 fornece um exemplo do LEQ-CDGP-Unif. O problema de otimização correspondente é dado abaixo.

Definição 3.2.10. ***Problema de Coloração Mínima em Geometria de Distâncias Uniformes Menores que ou Iguais (Minimum Less than or Equal Uniform Distance Geometry Problem - MinLEQ-CDGP-Unif)****: Dado um grafo simples ponderado não direcionado* $G = (V,E)$ *e um número* $\varphi \in \mathbb{N}$*, encontrar uma projeção* $x : V \to \mathbb{N}$ *tal que* $\forall (i,j) \in E,\ 1 \leq |x(i) - x(j)| \leq \varphi$ *e a extensão da projeção, definida por* $S = \max_{i \in V} x(i)$*, seja a menor possível.*

Conforme explicado anteriormente, todos os problemas apresentados neste capítulo envolvem a intersecção de geometria de distâncias com a classe de colorações de vértices em grafos, seguindo uma hierarquia de problemas, que pode ser vista graficamente na Figura 3.12.

3.2.3.1 Propriedades de factibilidade

Propriedades de factibilidade para o LEQ-CDGP e LEQ-CDGP-Unif também foram determinadas. De modo similar ao que ocorre com o EQ-CDGP, além do tipo de grafo de entrada, os valores das distâncias tem impacto direto na factibilidade da instância (ou seja, na solução do problema de decisão). O primeiro teorema diz respeito a instâncias do LEQ-CDGP que possuem solução NÃO para o problema.

Teorema 3.4. *Seja* $G = (V,E)$ *um grafo completo onde* $|V| \geq 3$ *(ou seja,* $G = K_n$*,* $n \geq 3$*) e* $\varphi \in \mathbb{N}$*. Então, se* $\varphi \leq |V| - 2$*, a instância correspondente do LEQ-CDGP-Unif possui solução NÃO (ou seja, não admite coloração factível).*

Prova. Seja $i \in V$ um vértice qualquer de G e seja $N(i)$ o conjunto de vértices adjacentes a i. Como o grafo é completo, tem-se que $|N(i)| = |V| - 1$. Além disso, as cores de

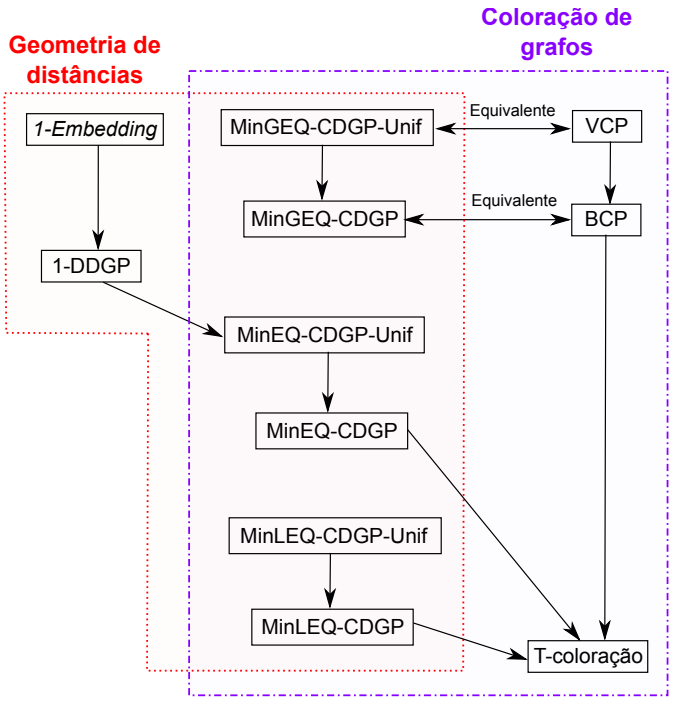


Figura 3.12: Hierarquia dos problemas de colorações com distâncias definidos neste capítulo.

todos os vértices devem ser diferentes (ou seja, cada vértice usa uma cor única). Como as restrições são no formato $1 \leq |x(i) - x(j)| \leq \varphi$, tem-se que as diferenças absolutas ($|x(i) - x(j)|$) devem ser minimizadas. O conjunto com as $|V|$ menores cores diferentes que minimiza tais diferenças absolutas é $\{1, 2, \ldots, |V|\}$.

Como, para toda aresta $(i, j) \in E$ tem-se que $1 \leq |x(i) - x(j)| \leq \varphi$ (e $x(i), x(j) \in \mathbb{N}$), então o conjunto de valores possíveis para $|x(i) - x(j)|$ é $\{1, 2, \ldots, \varphi\}$. Para um vértice i qualquer, tem-se que, como o mesmo tem a cor diferente de todos os outros, então o valor de $|x(i) - x(j)|$ para cada aresta $(i, j) \in E$ tal que $j \in N(i)$ deverá ser diferente, ou seja, $\forall j_1, j_2 \in N(i),\ |x(i) - x(j_1)| \neq |x(i) - x(j_2)|$. Assim, como há $|N(i)|$ arestas incidentes em i, cada uma com um valor diferente para a diferença absoluta entre a cor de i e o outro vértice j onde a mesma incide, tem-se que o conjunto de tais valores vai de 1 até $|V| - 1$ (ou seja, é $\{1, 2, \ldots, |V| - 1\}$). Porém, como $\varphi \leq |V| - 2$, tem-se que o conjunto possível é $\{1, 2, \ldots, |V| - 2\}$, o que pelo princípio da casa de pombo é impossível, ou seja, a aresta cuja diferença absoluta entre as cores de suas pontas é $|V| - 1$ violaria a restrição $|x(i) - x(j)| \leq \varphi$. Logo, a solução do EQ-CDGP-Unif é NÃO em tal situação. □

A Figura 3.13 mostra um exemplo da aplicação do Teorema 3.4 em uma instância do LEQ-CDGP-Unif composta pelo grafo completo K_5 e onde todas as distâncias são 3. Observe que como é necessário que o valor 4 esteja no conjunto de valores possíveis para

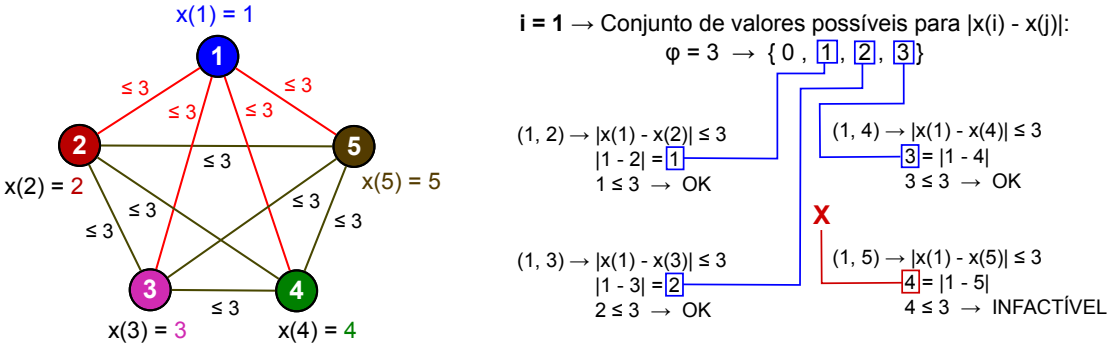


Figura 3.13: Exemplo de instância do LEQ-CDGP-Unif com solução NÃO, onde o valor da distância implica diretamente na infactibilidade.

as diferenças absolutas entre cores, caso as distâncias fossem 4 ou mais, a instância teria solução SIM.

O segundo resultado obtido foi para o LEQ-CDGP (ou seja, com distâncias arbitrárias), como pode ser visto no seguinte teorema.

Teorema 3.5. *Seja $G = (V, E)$ um grafo bipartido e $d : V \to \mathbb{N}$ uma função de distâncias. Então, a instância correspondente do LEQ-CDGP sempre tem solução SIM.*

Prova. Pela definição do problema, tem-se que $\forall (i, j) \in E,\ 1 \leq |x(i) - x(j)| \leq d(i, j)$, ou seja, o menor valor possível para a diferença absoluta entre cores de vértices adjacentes é 1. Dessa forma, para cada aresta, independentemente do valor de $d(i, j)$, se for possível colorir i e j de tal modo que a diferença absoluta entre eles seja 1, transitivamente não haverá violação da restrição. Sendo assim, pode-se proceder de duas formas distintas para a prova:

1. **Transformação da instância do LEQ-CDGP para uma do EQ-CDGP-Unif**: para cada restrição $1 \leq |x(i) - x(j)| \leq d(i, j)$, substitui-se a mesma por $|x(i) - x(j)| = 1$. Assim, passa-se a ter uma instância do EQ-CDGP-Unif. Pelo Teorema 3.2, tem-se que todo grafo bipartido tem solução SIM para este problema, de modo que a coloração ótima usará as cores 1 e 1+φ, onde φ é o peso associado a todas as arestas. Tem-se que $\forall (i, j) \in E,\ |x(i) - x(j)| = 1$, então $\varphi = 1$, assim, a coloração ótima usará as cores 1 e 2. Tal coloração satisfaz $\forall (i, j) \in E,\ 1 \leq |x(i) - x(j)| \leq d(i, j)$, portanto, é válida para o LEQ-CDGP.

2. **Transformação da instância do LEQ-CDGP para uma de coloração clássica**: para cada restrição $1 \leq |x(i) - x(j)| \leq d(i, j)$, substitui-se a mesma por $x(i) \neq x(j)$ ou, equivalentemente, por $|x(i) - x(j)| \geq 1$. Todo grafo bipartido é 2-colorível,

assim, pode-se usar as cores 1 e 2 para colorir os vértices do grafo original. Assim como na forma acima, tem-se tal coloração satisfaz $\forall (i,j) \in E,\ 1 \leq |x(i) - x(j)| \leq d(i,j)$ (uma vez que a diferença absoluta de cores entre vértices adjacentes será sempre 1), portanto, é válida para o LEQ-CDGP.

□

3.3 Métodos *branch-prune-and-bound*

Para os modelos de colorações com restrições de distâncias, foram desenvolvidas estratégias algorítmicas que combinam técnicas de otimização com propagação de restrições. Os algoritmos apresentados foram inicialmente desenvolvidos na pesquisa a nível de mestrado (Dias et al., 2013b; Dias, 2014), de modo que um estudo mais aprofundado do comportamento dos mesmos nos modelos apresentados foi realizado nesta tese, que será apresentado a seguir.

Um algoritmo *branch-and-prune* foi proposto por Lavor et al. (2012) para o DDGP em $\mathbb{R}^3$ (denotado por DDGP$_3$), baseado em uma versão anterior para o problema contínuo por Liberti et al. (2008). Tal método enumera as possíveis posições para os vértices, os quais devem ser localizados no espaço tridimensional de acordo com as distâncias impostas. A posição de um vértice i, onde $i \in [4, |V|]$, é determinada de acordo com os três vértices anteriores já posicionados, seguindo o esquema de ordenamento e intersecção de esferas descrito na Seção 3.1. No entanto, uma vez que uma distância entre o último vértice posicionado e um anterior aos últimos três pode ser violada, deve-se usar testes de factibilidade para garantir que a solução parcial seja válida. Lavor et al. (2012) usaram o teste de Factibilidade Direta de Distância (*Direct Distance Feasibility* - DDF) em seu *branch-and-prune*, onde $\forall (i,j) \in E \ \ | \|x(i) - x(j)\| - d_{ij} | < \varepsilon$, e ε é uma dada tolerância.

Para os modelos de colorações com distâncias, foram adaptados tais conceitos para a utilização do método nos problemas considerados. Uma das primeira observações que deve ser feita é que, para as colorações com distâncias, não existem suposições iniciais a serem respeitadas, de modo que não há ordenações explícitas de vértices a considerar, de modo que pode-se construir uma por um processo de enumeração implícita. Desse modo, foram combinados conceitos do *branch-and-prune* para o DDGP com procedimentos *branch-and-bound* de modo a evitar soluções parciais (ou seja, sequências de vértices

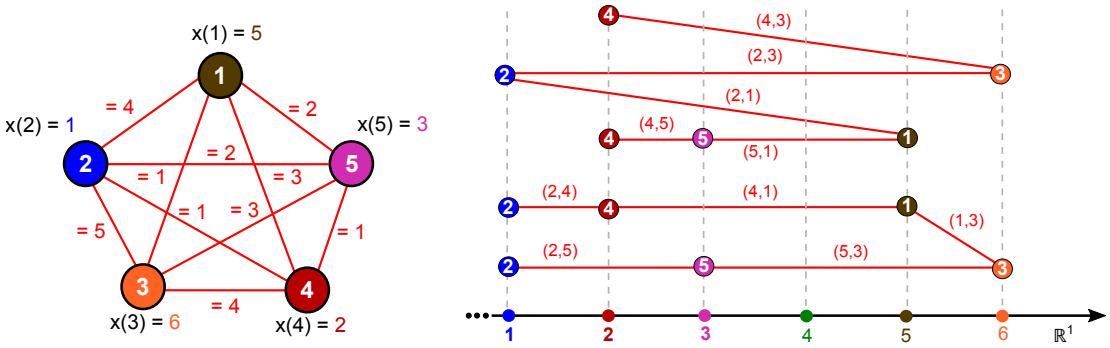


Figura 3.14: Exemplo de instância resolvida do MinEQ-CDGP para demonstração da execução dos métodos BPBs nas Figuras 3.15 e 3.16.

já coloridos) que não sejam capazes de obter um limite superior mais forte que a melhor solução corrente.

Sendo assim, o funcionamento básico do método *branch-prune-and-bound* (BPB) é descrito a seguir. Primeiramente, um vértice i não colorido é selecionado como ponto de partida. Tal vértice recebe a cor 1, a menor possível (uma vez que as cores são inteiros positivos). Em seguida, um vizinho j não colorido de i é selecionado. Um algoritmo de seleção de cor é usado para definir a cor de j e o processo é aplicado recursivamente para os vizinhos não coloridos de j. Quando um vizinho não colorido do vértice atual não puder ser encontrado, um vértice não colorido qualquer do grafo é usado. O pseudocódigo deste procedimento é dado no Algoritmo 3.1.

3.3.1 Seleção de cor

No *branch-prune-and-bound* proposto, existem duas possibilidades para determinar as cores que um vértice pode usar, que são usadas pela chamada à função SELECAOCORES(), a qual retorna o conjunto de cores possíveis.

O primeiro critério, denotado como **BPB-Prev**, é baseado diretamente no *branch-and-prune* original de (Lavor et al., 2012). Nele, quando um vértice i tiver de ser colorido, o vértice j que foi colorido imediatamente antes é usado para calcular a cor. Desse modo, se não houver nenhum vizinho colorido de i, a única cor possível é 1. Caso contrário, a função retornará um conjunto de cardinalidade no máximo 2, cujos elementos são:

1. $x(j) + d_{i,j}$.
2. $x(j) - d_{i,j}$ (retornado apenas se $x(j) > d_{i,j}$).

Algoritmo 3.1 Algoritmo *branch-prune-and-bound* geral.

Entrada: grafo $G = (V,E)$, distâncias $d : E \to \mathbb{Z}_{\geq 0}$, vértice anterior i, vértice atual j a ser colorido, solução (coloração) parcial x, melhor solução encontrada x_{melhor}, limite superior u, vetor *predec* de predecessores de cada vértice (inicializado com -1 em todas as posições) e profundidade da enumeração prf.

Função BRANCH-PRUNE-AND-BOUND($G = (V,E), d, i, j, x, x_{best}, u, pred, dpt$)
  para cada vizinho k de j **faça**
    se $predec[k] = -1$ **então**
      $predec[k] \leftarrow i$ ▷ Vértice atual é o predecessor dos vizinhos não visitados
  se $i = -1$ **então**
    $i \leftarrow predec[j]$ ▷ Se não veio de um vizinho, usar informação de predecesso
  $coresDisponiveis \leftarrow$ SELECAOCORES(G, d, i, j, x, u)
  enquanto $coresDisponiveis \neq \emptyset$ **faça**
    $cor \leftarrow$ elemento de $coresDisponiveis$
    $coresDisponiveis \leftarrow coresDisponiveis \setminus \{cor\}$
    $x(j) \leftarrow cor$
    se $\max_{v \in V \mid v \text{ está colorido}} x(v) \geq u$ **então**
      Remover cor de i
      continue ▷ Descartar esta solução parcial por *bounding*
    se TESTEFACTIBILIDADE(G, d, f, x, i) = **falso então**
      Remover cor de i
      retorne ▷ Violação de distância, descartar solução por *pruning*
    se $prf = |V|$ **então** ▷ Se verdadeiro, todos os vértices estão coloridos
      se $\max_{v \in V} x(v) < \max_{v \in V} x_{melhor}(v)$ **então**
        $x_{melhor} \leftarrow x$
        $u \leftarrow \max_{v \in V} x(v)$
    senão
      $temVizinho \leftarrow$ **falso**
      para cada vizinho k of j **faça**
        se k não está colorido **então**
          $temVizinho \leftarrow$ **verdadeiro**
          BRANCH-PRUNE-AND-BOUND($G, d, f, j, k, x, x_{melhor}, u, prf + 1$)
      se $temVizinho =$ **falso então**
        para cada vértice k de G tal que $predec[k] \neq -1$ **faça** ▷ Apenas vértices com predecessores
          se k não está colorido **então**
            BRANCH-PRUNE-AND-BOUND($G, d, f, -1, k, x, x_{melhor}, u, prf + 1$)
      Remover cor de i
  retorne x_{melhor}

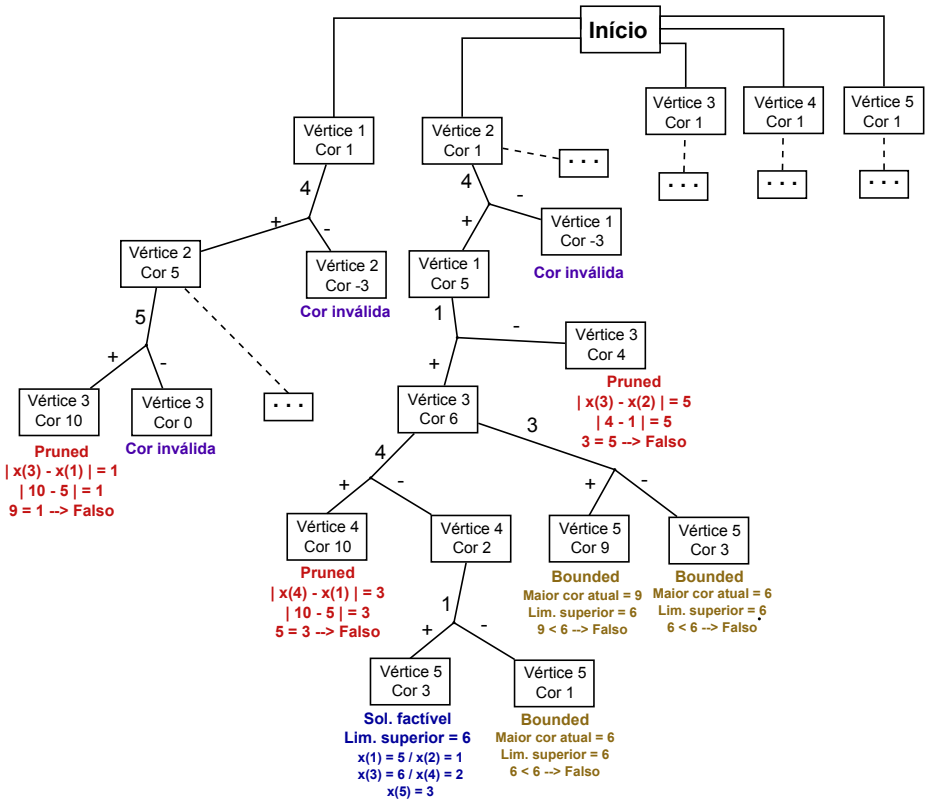


Figura 3.15: Enumeração parcial de soluções, iniciando pelo vértice 2, para a instância do MinEQ-CDGP definida pela Figura 3.14, usando o critério BPB-Prev, com seleção de cor baseada no vértice anterior.

Para melhorar o algoritmo, é utilizada informação de predecessores, de maneira que ao colorir um vértice, seus vizinhos que não têm informações de predecessor até a chamada corrente do *branch-prune-and-bound* são configurados a usar o vértice recém-colorido como predecessor. Esta informação é usada no caso de a busca reiniciar de um de seus vizinhos, o que ajuda a guiar o algoritmo. Este critério de seleção de cor, BPB-Prev, é realizado em tempo $O(1)$, já que são necessárias apenas duas operações aritméticas para determinar as cores. Para o exemplo de instância do MinEQ-CDGP dado na Figura 3.14, tem-se uma porção da enumeração realizada pela execução do BPB com este critério de seleção de cor apresentada na Figura 3.15.

O segundo critério de seleção, denotado como **BPB-Select**, leva em conta todos os vizinhos coloridos para determinar a cor para o vértice atual. Isso é feito resolvendo um sistema de expressões de valor absoluto (inequações ou equações) derivado das restrições de distâncias impostas pelas arestas. Sendo i o vértice a ser colorido, sua cor $x(i)$ será o resultado do sistema onde, para cada vizinho já colorido j de i, há uma expressão:

$$|x(j) - x(i)| \ \boxtimes \ d_{i,j}$$

Onde $\boxtimes$ é "=" (para MinEQ-CDGP, ou seja, restrições de igualdade), "$\geq$" (para MinGEQ-

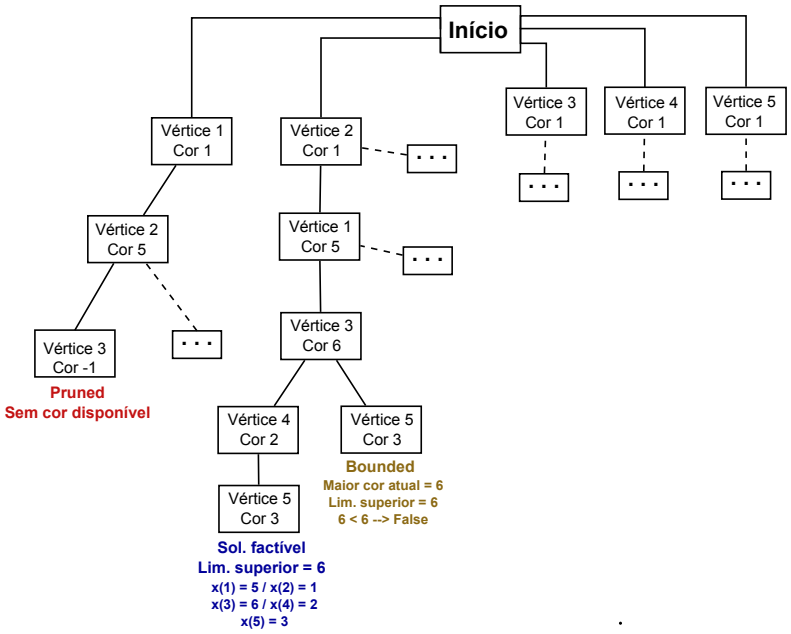


Figura 3.16: Enumeração parcial de soluções, iniciando pelo vértice 2, para a instância do MinEQ-CDGP definida pela Figura 3.14, usando o critério BPB-Select, com cor determinada pela solução de um sistema de expressões de valor absoluto.

CDGP, ou seja, restrições de desigualdade maior ou igual) ou "$\leq$" (para MinLEQ-CDGP, ou seja, restrições de desigualdade menor ou igual). A cor que será atribuída a j é o menor inteiro positivo a satisfazer todas as expressões do sistema. Ressalta-se que este procedimento sempre retorna um conjunto de cardinalidade 1, ou seja, apenas uma cor (já que somente a menor solução é retornada), bem como o mesmo já trata de garantir factibilidade, uma vez que a solução do sistema deverá satisfazer as restrições entre o vértice i e seus vizinhos. No entanto, a determinação da cor exige um trabalho maior, de modo que este critério de seleção é realizado em tempo $O(u)$, onde u é um limite superior para a maior cor usada, já que, para resolver o sistema, deve-se marcar cada possível solução do intervalo $[1..u]$ e selecionar o menor índice. A Figura 3.16 apresenta um exemplo da execução do BPB usando este critério de seleção de cor na instância da Figura 3.14.

3.3.2 Teste de factibilidade

Durante a construção de soluções, ao selecionar uma cor para um vértice, deve-se verificar se a mesma é factível quando apenas algumas distâncias são usadas na determinação da cor, especialmente no BPB-Prev. Desse modo, usou-se um teste baseado na Factibilidade Direta de Distância (*Direct Distance Feasibility* - DDF) usada no algoritmo *branch-and-*

Algoritmo 3.2 Teste de Factibilidade Direta de Coloração com Distâncias (*Direct Distance Coloring Feasibility* - DDCF)

Entrada: grafo $G = (V,E)$, tipo de restrição $\boxtimes$ do problema ("=", "$\geq$" ou "$\leq$"), distâncias $d : E \to \mathbb{Z}_{\geq 0}$, solução (coloração) parcial x e vértice atual i.

```
1: Função TESTE-DDCF(G, d, f, x, i)
2:     para cada vizinho k de i faça
3:         se k está colorido então
4:             se não |x(k) − x(i)| ⊠ d_{i,j} então
5:                 retorne falso
6:     retorne verdadeiro
```

prune para o DDGP (Mucherino et al., 2012).

Seja i o vértice que acabou de ser colorido. Então, deve-se verificar, para cada vizinho j de i que já foi colorido, se a condição $|x(i) - x(j)| \boxtimes d_{i,j}$ é válida (onde, novamente, $\boxtimes$ é "=", "$\geq$" ou "$\leq$", dependendo do modelo de coloração com distâncias sendo resolvido). Este teste pode ser visto como uma variação do DDF onde $\varepsilon = 0$ e com diferentes tipos de expressões utilizáveis (ao invés de apenas equações). Tal teste será chamado como Factibilidade Direta de Coloração com Distâncias (*Direct Distance Coloring Feasibility* - DDCF), cujo pseudocódigo é dado no Algoritmo 3.2.

Uma vez que o teste DDCF para um vértice verifica somente a factibilidade em relação aos seus vizinhos, o mesmo é executado em tempo $O(|V|)$ apenas para o vértice considerado. Nota-se também que, ao usar o critério BPB-Select, o teste pode ser dispensado já que a cor retornada sempre é factível.

A tabela 3.1 apresenta um sumário dos métodos *branch-prune-and-bound*, de acordo com critérios de seleção e de seus usos do teste de factibilidade.

Tabela 3.1: Sumário dos métodos *branch-prune-and-bound*.

Algoritmo	Conj. cores	Seleção de cor		Teste de factibilidade	
		Estratégia	Complex.	Quando	Complex.
BPB-Prev - Vizinho anterior	2	$x(i) = x(j) + d_{i,j}$ ou $x(i) = x(j) + d_{i,j}$ (se $x(i) > x(j) + d_{i,j}$)	$O(1)$	A cada vértice colorido	$O(\|V\|)$ para cada vértice
BPB-Select - Todos os vizinhos	1	$x(i) = \min\{k \in [1,u] : \forall (i,j) \in E,\ \|x(j) - k\| \boxtimes d_{i,j}\}$, onde $\boxtimes$ é "=", "$\geq$" ou "$\leq$"	$O(u)$	Não é necessário	-

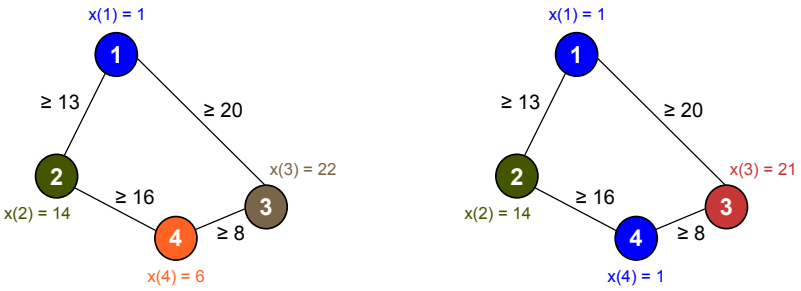
3.3.3 Aspectos de otimalidade

O método *branch-prune-and-bound* é um esquema de enumeração implícita, pelo qual as soluções são construídas colorindo-se cada vértice sendo que, a cada passo, uma solução

parcial pode ser descartada se for infactível (*pruned*) ou se não puder ser melhor que a melhor solução factível já encontrada (*bounded*). Esta abordagem objetiva encontrar a solução ótima, entretanto, verificou-se que isso depende da combinação de critério de seleção de cor e problema resolvido, como discutido a seguir.

O primeiro critério de seleção de cor, BPB-Prev, segue a mesma ideia usada pelo *branch-and-prune* para o DDGP. Tal critério efetua um caminhamento no grafo de entrada, de modo que quando um vértice é selecionado para ser colorido, isso é feito de modo a usar um que seja vizinho ao último que foi colorido e, se não houver vértice vizinho sem cor atribuída, então um outro qualquer que não tenha sido colorido é selecionado. Isso significa que nem todas as ordens de vértices são usadas, já que algumas permitiriam saltar de um vértice para outro que não é adjacente a ele, ainda que existam vizinhos não coloridos, sendo essas ordens não usadas no BPB-Prev.

A cor de um vértice é determinada usando informação dos vértices previamente coloridos, por meio de adição e subtração da distância imposta na aresta entre eles, o que significa que há no máximo duas cores possíveis para cada vértice durante a enumeração. Seguindo esta estratégia, o BPB-Prev determina intersecções de 0-esferas, onde cada par de vértices adjacentes entre si é posicionado na reta numérica, com a distância entre eles sendo exatamente igual a imposta na aresta correspondente. Isso é ótimo para o MinEQ-CDGP, já que o problema requer que todas as distâncias entre cores dos vértices seja exatamente igual ao peso da aresta. No entanto, para o MinGEQ-CDGP, a distância entre vértices adjacentes pode ser igual ou maior que o valor imposto na aresta correspondente. Uma vez que apenas ordens de vértices que fazem com que vizinhos sejam coloridos em sequência são usadas no BPB-Prev, e esse esquema faz com que a separação entre cores seja igual à distância exigida pela aresta, o mesmo pode não encontrar a solução ótima quando, para um par de vértices adjacentes i e j, tiver-se que $|x(i)-x(j)| > d_{i,j}$ na mesma. Um exemplo é apresentado na Figura 3.17.



Maior cor usada = 22
(a) Solução pelo BPB-Prev.

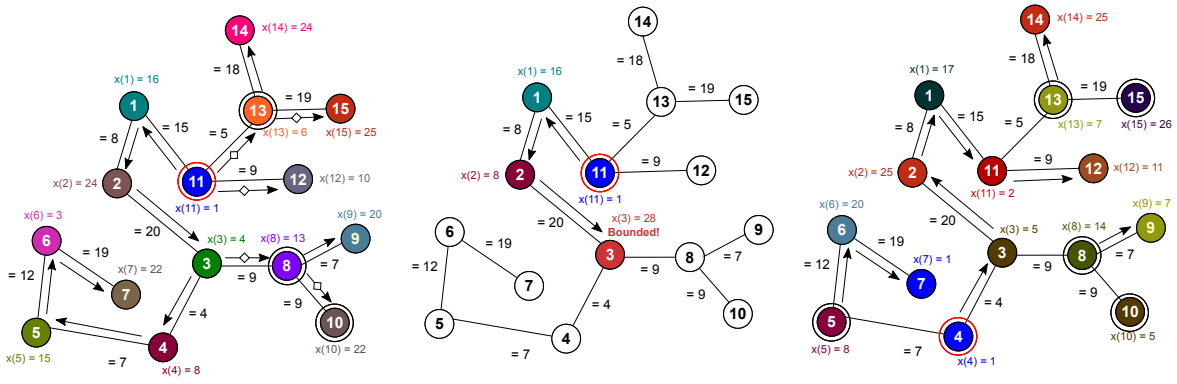
Maior cor usada = 21
(b) Solução ótima.

Figura 3.17: Exemplo de instância do MinGEQ-CDGP onde o BPB-Prev não consegue encontrar a solução ótima. Observe que, no BPB-Prev, para que os vértices 1 e 4 usem a mesma cor, um deles teria de ser o primeiro vértice i_1 da ordem de vértices e o outro deveria ser o segundo i_2, de modo que não haveria vizinho colorido de i_2 e a enumeração faria com que ele recebesse a cor 1. Entretanto, há sempre um vizinho não colorido de i_1 que não é i_2, então tal ordem não é usada pelo BPB-Prev.

Uma situação inversa ocorre com o BPB-Select. O algoritmo tenta determinar uma cor que satisfaça todas as restrições de distância de todos os vizinhos ao mesmo tempo, retornando a menor cor que não viole nenhuma delas. Para instâncias do MinGEQ-CDGP, esta estratégia permite encontrar soluções nas quais, para um par de vértices i e j, tenha-se que $|x(i) - x(j)| > d_{i,j}$, que poderiam não ser encontradas pelo BPB-Prev. Entretanto, em algumas instâncias do MinEQ-CDGP, na solução ótima, a igualdade $x(i) - x(j) = d_{i,j}$ pode ocorrer, onde $x(i) \geq x(j)$ e$x(j) \geq d_{i,j}$. Sob essas duas condições, tem-se que $x(i) = x(j) - d_{i,j}$ ou $x(i) = x(j) + d_{i,j}$. Se, na solução ótima, a segunda possibilidade estiver presente, então o BPB-Select não conseguirá atingir o ótimo, já que, quando estiver colorindo i, apenas a menor cor possível será usada, que é $x(j) - d_{i,j}$ e não $x(j) + d_{i,j}$ como na solução ótima. Para uma dada instância do MinEQ-CDGP, a mesma ordem de vértices que leva a esta situação no BPB-Select levaria a uma solução factível no BPB-Prev, dada que uma exista para a dada ordem. A Figura 3.18 mostra um exemplo deste cenário. Dadas tais considerações, verifica-se que as ordens de vértices são importantes ao determinar soluções, o que também ocorre com outros problemas de geometria de distâncias (Gonçalves e Mucherino, 2016).

3.4 Experimentos computacionais

Para avaliar o comportamento dos modelos de colorações com distância e dos algoritmos *branch-prune-and-bound*, foram realizados dois conjuntos de experimentos: o primeiro



(a) Solução ótima pelo BPB-Prev.

(b) Infactibilidade pelo BPB-Select usando a mesma ordem de vértices da solução ótima pelo BPB-Prev.

(c) Melhor solução pelo BPB-Select.

Figura 3.18: Exemplo de instância do MinEQ-CDGP onde o BPB-Select não consegue encontrar a solução ótima, com limite superior igual a 26. As setas seguem a ordem de vértices dada pelas ordens de vértices e, no caso do BPP-Prev, um círculo no meio da seta indica qual vizinho foi considerado para um vértice quando a enumeração foi reiniciada. Um círculo ao redor de um vértice indica que a enumeração reiniciou dele, e um círculo duplo indica o primeiro vértice da ordem. O verdadeiro ótimo foi encontrado pelo BPB-Prev usando a dada ordem de vértices, mas o BPB-Select não consegue encontrar uma solução factível com a mesma ordem de vértices e limite superior (sem ele, a solução encontrada pelo BPB-Select com a mesma ordem teria 28 como maior cor usada). A melhor solução possível pelo BPB-Select usa uma ordem diferente de vértices, que não leva à solução ótima..

envolveu a geração de vários grafos aleatórios com diferentes números de vértices, de acordo com algumas configurações, para determinação de quantos possuem ciclos ímpares e pares ou somente pares (sendo árvores o restante dos grafos), já que algumas propriedades das colorações com distâncias são relacionadas a estes tipos de grafos. Já o segundo envolveu a avaliação do comportamento dos algoritmos *branch-prune-and-bound* nos diferentes tipos de instâncias e problemas.

Todas as implementações foram realizadas em linguagem C e compiladas com o `gcc` 4.8.4 usando as opções `-Ofast -march=native -mtune=native` e executadas em um computador equipado com processador Intel Core i7-3770 (3,4GHz), 8GB de memória e sistema operacional Linux Mint 17. Tais experimentos são descritos a seguir.

3.4.1 Contagem de tipos de grafos em instâncias aleatórias

Por meio dos Teoremas 3.2 e 3.3, tem-se informações sobre quais tipos de grafos sempre tem soluções factíveis para o EQ-CDGP e EQ-CDGP-Unif. Baseando-se nisso, gerou-se uma grande quantidade de grafos aleatórios com diferentes números de vértices e contou-

se quantos possuem ciclos (e dentre esses, quais possuem ciclos pares e ímpares ou somente pares) e quantos são árvores.

Cada grafo aleatório inicia como uma árvore geradora aleatória, ou seja, um grafo conexo não direcionado $G = (V, E)$, onde $|E| = |V| - 1$. Para gerar esta árvore inicial, usou-se um algoritmo de passeio aleatório proposto por Aldous (1990). O procedimento funciona usando um conjunto V^* de vértices fora da árvore e um conjunto W de arestas da árvore geradora. Então, toda vez que o passeio aleatório chega a um vértice j fora da árvore, a aresta (i, j) é adicionada a E, e j é removido de V^*. Isso continua até que $V^* = \varnothing$. Nota-se que este procedimento equivale a realizar um passeio aleatório em um grafo completo de $|V|$ vértices, sendo que o mesmo gera árvores de maneira uniforme, ou seja, cada árvore geradora tem a mesma probabilidade de ser gerada.

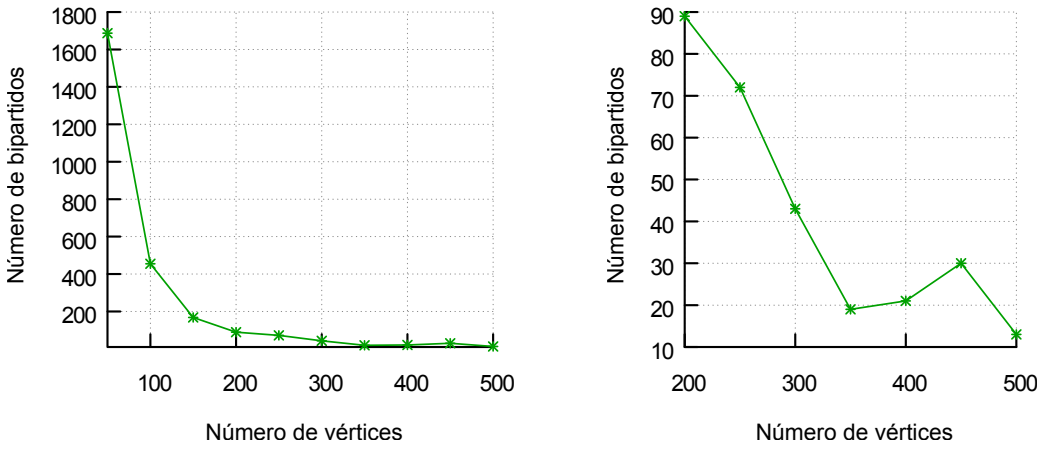
Depois que a árvore inicial é gerada, são adicionadas novas arestas aleatórias até que o grafo tenha o número desejado de arestas. Este parâmetro também é gerado aleatoriamente e amostrado do intervalo $\left[|V| - 1, \frac{|V|(|V|-1)}{2}\right]$. Tal intervalo garante que o grafo gerado sempre seja conexo e seja ao menos uma árvore e no máximo um grafo completo.

Na Tabela 3.2, são mostradas estatísticas obtidas ao usar o procedimento descrito acima para gerar 1000000 (um milhão) de grafos aleatórios para cada $|V| \in \{50, 100, 150, 200, 250, 300, 350, 400, 450, 500\}$. Pode-se observar que a maior parte dos grafos gerados (mais de 99%) tem ciclos pares e ímpares, o que significa que o conjunto de instâncias do EQ-CDGP-Unif que possuem soluções factíveis é muito pequeno. Pode-se deduzir, então, que esta configuração gerou poucos grafos bipartidos. Ressalta-se que como foi gerada a mesma quantidade de grafos para cada $|V|$ usado, este comportamento é esperado, entretanto, se o número de grafos gerados for aumentado, espera-se que mais bipartidos sejam gerados. Para o EQ-CDGP (com distâncias arbitrárias), o espaço de instâncias que tenham garantidamente solução factível é ainda menor, já que apenas árvores se encaixam nisso. Entretanto, como mostrado na Subseção 3.2.2 grafos com ciclos pares e/ou ímpares podem ter soluções dependendo de como as arestas são ponderadas.

Na Figura 3.20, pode-se observar que o crescimento do número médio de arestas entre todos os grafos gerados para cada número de vértices. Como a quantidade de arestas é proporcional ao quadrado do número de vértices (já que $\frac{|V|(|V|-1)}{2} \in O(|V|^2)$), a curva segue um padrão similar, sendo uma meia parábola.

Tabela 3.2: Quantidade de grafos aleatórios com ciclos pares e ímpares, apenas ciclos pares e nenhum ciclo (árvores) e bipartidos para cada número de vértices. Para cada tamanho, foram gerados 1000000 grafos.

|V|	Média |E|	# Grafos com ciclos pares e ímpares	# Grafos apenas com ciclos pares	# Grafos acíclicos (Árvores)	# Grafos bipartidos
50	637,56	998309	854	837	1691
100	2523,52	999553	238	209	447
150	5656,64	999832	74	94	168
200	10059,56	999910	45	45	90
250	15675,94	999926	41	33	74
300	22586,52	999958	18	24	42
350	30688,21	999975	13	12	25
400	40120,76	999975	14	11	25
450	50678,60	999971	15	14	29
500	62628,32	999988	6	6	12



(a) $|V| \in [50, 500]$ **(b)** $|V| \in [200, 500]$ (para escala)

Figura 3.19: Número total de grafos bipartidos gerados de 1000000 de grafos aleatórios para cada número de vértices.

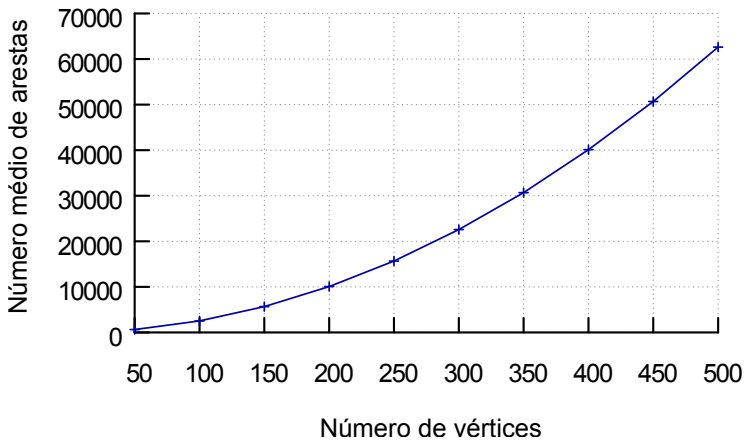


Figura 3.20: Crescimento da quantidade média de arestas geradas quando o número de vértices aumenta.

3.4.2 Resultados dos algoritmos *branch-prune-and-bound*

Para usar alguns dos grafos aleatórios nos experimentos com os BPBs, selecionou-se três grafos de cada tipo considerado: com ciclos pares e ímpares (denotados por *OddCycle*), apenas com ciclos ímpares (denotados por *EvenCycle*) e árvores (denotados por *Tree*). Em seguida, as arestas foram ponderadas aleatoriamente seguindo uma distribuição uniforme no intervalo $[1..30]$. Foram realizados dois subconjuntos de experimentos: o primeiro envolveu somente os problemas EQ-CDGP e EQ-CDGP-Unif, de modo a obter uma solução factível para cada uma de suas instâncias (ou seja, os algoritmos usam apenas procedimentos de *pruning*, terminando a busca assim que a primeira solução válida é encontrada), e o segundo envolveu os modelos de otimização dos problemas de coloração com restrições de $=$ e $\geq$ (MinEQ-CDGP, MinEQ-CDGP-Unif, MinGEQ-CDGP e MinGEQ-CDGP-Unif). Como os tempos de execução de todos os grafos com 4 a 12 vértices foram muito pequenos (no máximo 0,3 segundos), optou-se por omitir tais resultados.

A Tabela 3.3 apresenta os resultados obtidos para um número de vértices entre 14 e 26, com incrementos de 2 a 2, considerando cada algoritmo BPB aplicado às versões de decisão. Observa-se que, apesar de o BPB-Prev efetuar menos trabalho que o BPB-Select para determinar a cor do vértice atual (ou seja, tem complexidade de tempo menor, como visto na Tabela 3.1), este último consegue chegar a uma solução factível antes do primeiro (ou seja, resolve o problema de decisão em menos tempo) e com um menor valor de função objetivo (maior cor usada). Isso ocorre porque a cada nó da árvore de enumeração do BPB-Prev existem duas possibilidades de cor, o que aumenta a ramificação da mesma. Tal fato pode ser observado também nas colunas #Prunes e #Nós da tabela, de modo que, em toda instância, BPB-Select efetua menos *prunes* que o BPB-Prev (ou seja, podas por infactibilidade ou falta de cor válida), já que existem menos nós. Quando a instância é infactível, ambos os métodos geram grandes árvores de enumeração, com muitas ramificações, de modo que quando as opções possíveis são exauridas e não foi encontrada solução válida, então a instância é infactível. No entanto, é importante ressaltar que isto só é garantido se o algoritmo não passar do tempo limite imposto. Por exemplo, nas instâncias *Tree* com 24 e 26 vértices, mesmo que o Teorema 3.3 garanta que exista ao menos uma solução factível para as mesmas, uma ou ambas as variantes do BPB não conseguiram encontrar uma solução factível no tempo limite imposto.

Da mesma maneira, a Tabela 3.4 apresenta resultados para os algoritmos BPB consi-

Tabela 3.3: Resultados dos algoritmos BP (decisão/busca) aplicados a instâncias do EQ-CDGP - 14 a 26 vértices.

			BPB-Prev				BPB-Select			
|V|	**Tipo**	**Inst**	**Maior cor**	**#Prunes**	**#Nós**	**Tempo CPU (s)**	**Maior cor**	**#Prunes**	**#Nós**	**Tempo CPU (s)**
14	OddCycle	1	Infactível	3930625	4151102	7.649	Infactível	119828	218219	0.327
	OddCycle	2	Infactível	35681	25394	0.051	Infactível	5217	6949	0.012
	OddCycle	3	Infactível	25749	16751	0.034	Infactível	4890	6169	0.011
	EvenCycle	1	Infactível	22438	20427	0.039	Infactível	286	955	0.001
	EvenCycle	2	Infactível	14590592	17014208	30.952	Infactível	240185	480564	0.695
	EvenCycle	3	Infactível	9318840	10006150	18.597	Infactível	146774	329350	0.453
	Tree	1	41	3	20	0.000	29	54	138	0.000
	Tree	2	63	593	940	0.002	37	164	385	1.000
	Tree	3	67	139	209	0.000	31	60	166	0.000
16	OddCycle	1	Infactível	16720253	18533711	33.922	Infactível	229364	464467	0.655
	OddCycle	2	Infactível	40827	26130	0.049	Infactível	10334	12037	0.021
	OddCycle	3	Infactível	64340	40739	0.077	Infactível	13417	15677	0.027
	EvenCycle	1	79	1608	1961	0.004	35	356	799	1.000
	EvenCycle	2	Infactível	340726831	428448739	736.649	Infactível	10410162	19341115	27.659
	EvenCycle	3	72	6098	9620	0.017	29	1390	3171	4.000
	Tree	1	65	1324	1820	0.003	30	222	502	1.000
	Tree	2	86	41	61	0.000	30	15	50	0.000
	Tree	3	74	2539	4707	0.009	36	329	756	1.000
18	OddCycle	1	Infactível	852848	743950	1.369	Infactível	18527	33062	0.050
	OddCycle	2	Infactível	1282955	933589	1.724	Infactível	29654	46174	0.073
	OddCycle	3	Infactível	165004	138284	0.256	Infactível	1411	3456	0.005
	EvenCycle	1	Infactível	18409076	17857282	32.227	Infactível	92766	208614	0.295
	EvenCycle	2	Infactível	264088256	314471197	557.056	Infactível	996127	2340311	3.174
	EvenCycle	3	Infactível	695095555	673477736	1209.959	Infactível	2312073	4373462	6.270
	Tree	1	88	115791	105413	0.190	31	6170	12737	18.000
	Tree	2	55	7556	7961	0.015	25	1171	2731	4.000
	Tree	3	91	540	782	0.001	28	57	157	0.000
20	OddCycle	1	Infactível	397251801	343891224	619.704	Infactível	1786324	3216901	4.772
	OddCycle	2	Infactível	2446590	1682426	3.161	Infactível	63034	89385	0.143
	OddCycle	3	Infactível	1152062	849161	1.555	Infactível	31642	48916	0.076
	EvenCycle	1	Nenhuma/TLE	5236973472	5891034620	10800.000	Infactível	20719998	47209517	65.792
	EvenCycle	2	76	914490	869714	1.582	36	13158	25609	36.000
	EvenCycle	3	Infactível	31560482	28428856	50.572	Infactível	167148	351029	0.490
	Tree	1	64	2446961	3920630	7.028	24	107586	201000	285.000
	Tree	2	103	5736	9114	0.017	29	560	1443	2.000
	Tree	3	96	77841	88010	0.157	36	2908	6536	9.000
22	OddCycle	1	Infactível	26254768	23426389	54.311	Infactível	104901	215651	0.427
	OddCycle	2	Infactível	36293718	30515680	69.453	Infactível	109170	215741	0.418
	OddCycle	3	Infactível	1642123	973686	2.043	Infactível	128492	149152	0.344
	EvenCycle	1	Infactível	224340499	251232540	567.965	Infactível	571017	1351418	2.510
	EvenCycle	2	Nenhuma/TLE	3865244040	4660919459	10800.000	Infactível	19329265	52300771	93.771
	EvenCycle	3	Nenhuma/TLE	5092081153	6081809449	10800.000	Infactível	128729661	291841844	540.137
	Tree	1	110	2786508998	4305296369	10800.000	38	30077382	61986076	118083.000
	Tree	2	150	66370008	90610477	209.395	52	926771	1912598	3639.000
	Tree	3	173	240927	402687	0.809	46	3811	8508	16.000
24	OddCycle	1	Infactível	641048895	507509535	924.188	Infactível	2317601	3780090	7.909
	OddCycle	2	Infactível	123005	76833	0.147	Infactível	48071	50843	0.123
	OddCycle	3	Infactível	65782	42183	0.086	Infactível	32667	34377	0.084
	EvenCycle	1	127	165262470	208210517	357.978	53	3189	6383	12.000
	EvenCycle	2	Infactível	5346209932	5873842065	10800.000	Infactível	72906817	164254755	306.506
	EvenCycle	3	Nenhuma/TLE	5881951679	5954683241	10800.000	Nenhuma/TLE	2667932569	5772168624	10800.000
	Tree	1	86	3138775	4790089	8.727	49	44922956	94109754	183801.000
	Tree	2	Nenhuma/TLE	5271476912	5873172244	10800.000	48	15927595	33378945	64015.000
	Tree	3	65	2407147598	3732640459	6421.631	50	37578502	74620519	143960.000
26	OddCycle	1	Infactível	7580340832	5958839459	10800.000	Infactível	26753348	42141613	64.120
	OddCycle	2	Infactível	3643726	2126536	3.926	Infactível	328702	365478	0.636
	OddCycle	3	Infactível	1370179	809597	1.493	Infactível	222908	242362	0.453
	EvenCycle	1	Nenhuma/TLE	6206760415	5967575526	10800.000	Infactível	35785765	76334277	105.767
	EvenCycle	2	Nenhuma/TLE	7674629273	5970591346	10800.000	Nenhuma/TLE	4212458957	7441431184	10800.000
	EvenCycle	3	Nenhuma/TLE	5545756309	6074016769	10800.000	Infactível	2442228012	6367668435	8458.570
	Tree	1	Nenhuma/TLE	3732652924	5975408570	10800.000	46	22911283	46945890	66454.000
	Tree	2	152	3021403092	4951393771	8545.174	56	12752749	25106292	35637.000
	Tree	3	Nenhuma/TLE	4289259339	6288070478	10800.000	Nenhuma/TLE	4254335955	7402431191	10800.000

derando suas versões de otimização e aplicados apenas a instâncias factíveis do MinEQ-CDGP (que possuíram ao menos uma solução por qualquer método usado). Novamente, o BPB-Select teve um tempo de execução menor que o BPB-Prev em tais instâncias. Quando ambos os algoritmos atingiram o tempo limite sem encontrar a solução ótima (e sem provar que a encontrada o seja), o BPB-Select obteve uma melhor solução que o BPB-Prev.

Os últimos experimentos foram realizados aplicando-se os algoritmos BPB a todas as instâncias, mas agora considerando o problema MinGEQ-CDGP (ou seja, mudando o $=$ nas restrições por $\geq$). Os resultados são apresentados na Tabela 3.5. Como as instâncias do MinGEQ-CDGP sempre são factíveis, deve-se considerar apenas suas versões de otimização, como já feito na Subseção 3.2.1. Observa-se que há um comportamento diferente para o MinGEQ-CDGP, de modo que não há um método que seja o melhor absoluto dentre BPB-Prev e BPB-Seect. Em algumas instâncias, como as *OddCycle* 2 e 3 com 14, 16, 18 e 22 vértices, *EvenCycle* 2 com 18 e 20 vértices, *EvenCycle* 1 com 26 vértices e algumas instâncias *Tree* com 20 e 22 vértices (mas não limitando a estas), BPB-Prev encontrou a melhor solução e provou sua otimalidade mais rapidamente, enquanto em outras, BPB-Select foi mais rápido. Tal situação ocorre pois ambos os algoritmos encontram muito mais soluções factíveis, já que toda solução do MinEQ-CDGP é válida para o MinGEQ-CDGP, mas o inverso nem sempre é verdade, de movo que existem muito mais soluções possíveis para o MinGEQ-CDGP que cada algoritmo pode encontrar.

Uma outra importante observação sobre os resultados para o MinGEQ-CDGP é que a melhor solução retornada pelo BPB-Prev pode, em algumas instâncias, não ser a ótima, como discutido na Subseção 3.3.3. Este é o caso dos resultados obtidos para as instâncias *OddCycle* 3 e *EvenCycle 3*, ambas com 20 vértices. Ressalta-se também que, para instâncias maiores, quando os algoritmos não retornam a mesma solução e ambos atingem o tempo limite imposto, nenhuma solução pode ser dita garantidamente ótima.

Por fim, os tamanho das instâncias que cada algoritmo atinge o tempo limite para cada problema de otimização são dados na Tabela 3.6. Pode-se observar que, nos experimentos realizados, quando há 22 ou mais vértices, há uma grande probabilidade de que o algoritmo não atingirá a solução ótima no tempo dado. Entretanto, é importante que ressaltar que apesar de as instâncias terem sido organizadas apenas pelo número de vértices, a densidade do grafo de entrada pode impactar diretamente no tempo de execução.

3.5 Notas do capítulo

Os modelos apresentados neste capítulo envolveram a aplicação de conceitos de geometria de distâncias aos problemas de coloração em grafos, de modo a obter variantes com características interessantes, tanto do ponto de vista teórico quanto prático. Para cada um deles, foram identificadas propriedades envolvendo factibilidade (ou seja, a existência ou não de solução) e complexidade computacional, sendo estas as principais contribuições deste capítulo.

Os algoritmos *branch-prune-and-bound*, inicialmente propostos no mestrado, foram estudados de forma mais aprofundada nesta tese, de modo a verificar seu comportamento de acordo com o modelo de coloração e tipo de instância. Observou-se que nem sempre os métodos são exatos, de maneira similar ao *branch-and-prune* para o DDGP. Apesar dos métodos não serem competitivos, o estudo de tal comportamento pode ser útil para determinar rotinas que melhorem o desempenho de outros tipos de algoritmos.

Convém ressaltar que outras variações de colorações ainda podem ser exploradas com base nas identificadas nesta tese e em trabalhos de geometria de distâncias, envolvendo tanto outros tipos de restrições de adjacência, bem como outras características, como listas permitidas de distâncias (não contíguas).

Tabela 3.4: Resultados dos algoritmos BPB (otimização) aplicados a instâncias factíveis do MinEQ-CDGP - 14 a 26 vértices.

			BPB-Prev (otimização)							BPB-Select (otimização)						
|V|	**Tipo**	**Inst**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**
14	Tree	1	23	73169	131887	5	330867	0.000	0.582	23	1128	37719	3	88155	0.000	0.087
		2	23	24629	40448	8	109606	0.002	0.200	23	2510	26224	2	61087	0.001	0.059
		3	31	15948	25736	6	67059	0.000	0.129	31	981	19645	1	45496	0.000	0.047
16	EvenCycle	1	26	734907	1444932	20	3430433	0.004	6.442	26	13238	429023	3	965581	0.001	0.937
		3	26	382461	478520	6	1341496	0.017	2.357	26	3968	105082	2	245263	0.004	0.239
	Tree	1	30	1181358	1587427	10	4086926	0.003	7.336	30	35892	620039	1	1338678	0.001	1.310
		2	27	669737	1210214	22	2903465	0.000	5.112	27	2611	216168	2	502088	0.000	0.472
		3	24	789653	1566492	15	3302294	0.009	5.845	24	2437	495712	3	1061101	0.001	1.006
18	Tree	1	28	36615569	40788308	25	105873095	0.190	186.338	28	340240	6686934	3	13736111	0.018	12.973
		2	25	22063979	49610912	6	97340621	0.015	171.816	25	1124920	11173394	1	24302076	0.004	22.866
		3	23	2533460	4177180	24	10505001	0.001	18.544	23	2843	1460443	3	3054495	0.000	2.855
20	EvenCycle	2	24	151689781	345164555	16	576184304	1.582	1008.727	24	2750542	55696609	5	105240633	0.036	101.383
	Tree	1	24	267889809	423860003	10	905816365	7.028	1639.773	24	14799751	221687235	1	436478894	0.285	426.623
		2	27	5168120	11062144	17	20785962	0.017	36.394	27	74695	2918725	2	6562696	0.002	6.235
		3	24	43956039	60091614	25	144775201	0.157	250.804	24	151157	9245694	3	18762973	0.009	18.051
22	Tree	1	32	135610562	2977763180	14	4821423002	9644.021	10800.000	31	2380566	335092187	3	676132261	118.083	907.120
		2	42	1680426488	2560100224	21	5467261228	209.395	10800.000	42	9794640	881203038	3	1679857563	3.639	2167.077
		3	40	951641886	3519285927	31	5995079706	0.809	10800.000	40	11077774	1474744293	5	2953821790	0.016	3858.260
24	EvenCycle	1	45	1303800435	3547526485	11	5958438816	357.978	10800.000	42	39660438	1453410172	4	3114013223	0.012	4066.296
	Tree	1	39	1434035663	2791529314	14	6173893175	8.727	10800.000	39	16950544	842284814	3	1695927132	183.801	2314.789
		2	Nenhuma	0	5271476912	0	5873172244	–	10800.000	46	3952610	3667193238	2	8018634142	64.015	10800.000
		3	50	433541618	3573077273	3	6250834667	6421.631	10800.000	39	261747766	3728282741	2	8162494055	143.960	10800.000
26	Tree	1	Nenhuma	0	3732652924	0	5975408570	–	10800.000	46	171310935	5519429069	1	10855894570	66.454	10800.000
		2	109	11050015	3844032304	6	6262294917	8545.174	10800.000	56	8987188	4614079720	1	10914203771	35.637	10800.000

Tabela 3.5: Resultados dos algoritmos BPB (otimização) aplicados a instâncias do MinGEQ-CDGP - 14 a 26 vértices.

			BPB-Prev (otimização)							BPB-Select (otimização)						
|V|	**Tipo**	**Inst**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**
14	OddCycle	1	21	645077	450680	20	1566639	0.000	2.844	21	211465	550635	3	1669427	0.000	1.623
		2	37	341525	1207988	15	1226427	0.000	2.411	35	58843	894007	4	2035550	0.000	2.146
		3	38	924821	2933266	21	2789762	0.000	5.464	38	812206	4861676	6	11829374	0.000	12.994
	EvenCycle	1	20	1457	2389	10	6072	0.006	0.011	20	218	535	5	2197	0.000	0.002
		2	17	294751	220488	14	748444	0.000	1.393	17	217216	218973	4	936865	0.000	0.898
		3	21	94191	67037	9	238018	0.000	0.458	21	59565	131454	2	440280	0.000	0.422
	Tree	1	18	33888	10263	7	73883	0.000	0.125	18	8925	7703	3	39260	0.000	0.038
		2	21	21266	5669	4	49842	0.000	0.093	21	35518	9076	1	108651	0.000	0.105
		3	21	1795	555	7	3950	0.000	0.008	21	1806	558	2	5532	0.000	0.006
16	OddCycle	1	18	126328	161820	10	371541	0.000	0.682	18	54419	53715	4	228464	0.000	0.222
		2	53	42249077	129768220	31	104325789	0.000	194.604	53	38366062	972153720	14	1746554116	0.000	1980.567
		3	48	27501531	79483153	31	64892955	0.000	121.646	46	17189261	401975939	14	696082548	0.000	803.593
	EvenCycle	1	20	657205	348380	19	1677773	0.000	3.106	20	258893	570336	4	2015700	0.000	1.930
		2	20	2872809	1646080	12	6797918	0.000	11.987	20	1034882	3519549	5	9530309	0.000	9.234
		3	21	135359	16277	6	270124	0.000	0.478	21	27916	29465	3	127857	0.000	0.123
	Tree	1	21	394331	135792	13	871108	0.000	1.626	21	149139	336081	2	1213969	0.000	1.176
		2	21	1135197	308686	17	2710177	0.000	4.730	21	273524	592748	3	2264385	0.000	2.162
		3	21	964567	631390	10	2454902	0.000	4.340	21	860528	1271113	1	4889001	0.000	4.642
18	OddCycle	1	23	403646	689781	29	1096305	0.000	1.995	23	89251	2157123	6	4215186	0.000	4.113
		2	28	270417	501032	24	711228	0.000	1.293	28	156374	725207	10	1786184	0.000	1.788
		3	27	68883	72174	28	166526	0.000	0.302	27	28930	127942	4	383073	0.000	0.373
	EvenCycle	1	21	226428	172924	33	544713	0.000	0.972	21	153559	261663	6	929927	0.000	0.895
		2	21	574529	272687	17	1333025	0.000	2.320	21	441889	1561547	4	5096657	0.000	4.877
		3	21	1959536	3201061	19	5804091	0.000	10.301	21	2328257	2049088	3	8815279	0.000	8.435
	Tree	1	21	23651729	5659135	8	42814302	0.000	75.126	21	3480825	11725139	5	31510553	0.000	29.640
		2	21	12707141	6951990	8	27625579	0.000	48.768	21	3385503	5562843	3	16351954	0.000	15.559
		3	19	2971604	1596362	22	8321143	0.000	14.544	19	1168701	3161985	2	10148322	0.000	9.588

Continua na próxima página.

Tabela 3.5 – *Continuação da página anterior.*

			BPB-Prev (otimização)							BPB-Select (otimização)						
|V|	**Tipo**	**Inst**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**
20	OddCycle	1	25	80860020	130297491	22	250587621	0.000	447.298	25	30005823	182202237	2	481689865	0.000	478.757
		2	30	22109746	70876798	24	77086730	0.000	139.693	30	9486329	105756952	5	248377444	0.000	255.918
		3	30	5854520	15090450	34	18833088	0.000	33.663	29	3280215	14139380	4	35525456	0.000	35.784
	EvenCycle	1	21	306492466	290976595	21	748440185	0.000	1335.263	21	244902785	91113414	5	724708042	0.000	709.116
		2	21	76491974	70989106	7	196478938	0.000	338.722	21	265125525	35429733	2	548890756	0.000	520.729
		3	22	941508	1090455	18	2418408	0.000	4.274	21	4529934	1546737	5	12937093	0.000	12.531
	Tree	1	21	501421339	233786884	9	1231692406	0.000	2159.958	21	497365528	932560094	5	3285799702	0.000	3186.089
		2	21	8215852	10377774	14	25479374	0.000	45.027	21	27313206	17555281	4	98225542	0.000	93.566
		3	21	15766711	13251908	14	41314410	0.000	72.232	21	11863544	18549550	4	65387940	0.000	62.723
22	OddCycle	1	35	42938429	40380216	27	95066073	0.000	218.883	35	97405696	156750327	2	587060884	0.000	603.005
		2	37	11311525	19097577	23	32949477	0.000	75.509	37	3848239	44411978	4	107463430	0.000	108.827
		3	73	2000325327	7280778464	38	5792046990	0.000	10800.000	68	224507385	5182891368	18	9279818233	0.000	10800.000
	EvenCycle	1	30	20043221	14089082	12	52206915	0.000	117.054	30	13013868	45340553	2	130340949	0.000	133.761
		2	31	31012652	60034933	14	148811258	0.000	337.540	31	35432407	48769777	4	186354740	0.000	185.295
		3	31	23411492	17381110	11	62086727	0.000	110.881	31	23741231	29109093	7	104954830	0.000	102.303
	Tree	1	27	270203508	76667564	8	553985858	0.000	1220.984	27	172372918	360684580	2	1046326590	0.000	991.931
		2	30	1943743038	920983002	17	4542439718	0.000	8993.205	30	1436150566	3201778158	4	10833815876	0.000	10800.000
		3	30	1485642659	713665659	22	3198738831	0.000	5776.637	30	1670595723	3233015335	4	10875030416	0.000	10800.000
24	OddCycle	1	47	732050577	3948302960	20	5846159217	0.020	10800.000	47	1753474763	3033132004	2	10702330028	0.000	10800.000
		2	187	3326239091	7393153183	48	5778965820	0.000	10800.000	190	86957928	3972381158	19	6714361416	0.000	10800.000
		3	207	3276549688	7126762369	67	5586407952	0.000	10800.000	190	144070174	4264496756	17	6712630855	0.000	10800.000
	EvenCycle	1	29	2728732397	853862019	20	5508977994	0.000	9895.354	29	288966325	112927315	3	777625631	0.000	752.544
		2	31	1124175678	995413847	35	2986509166	0.000	5260.684	31	2111407527	2771811106	5	10977102372	0.000	10800.000
		3	77	3504643842	3280262618	10	5852208979	0.093	10800.000	40	2577213292	2934594601	2	11334482922	0.000	10800.000
	Tree	1	34	2240431219	2066840924	6	6162097374	0.000	10800.000	29	1020853189	2143005122	5	6497004349	0.000	6207.489
		2	29	3170945967	932984259	11	5994115145	0.000	10800.000	29	5027356324	2614110038	5	12545263004	0.000	10800.000
		3	30	3078162435	504276478	8	6395072659	2.978	10800.000	30	492291164	4799922157	3	11278320407	0.000	10800.000

Continua na próxima página.

Tabela 3.5 – *Continuação da página anterior.*

			BPB-Prev (otimização)							BPB-Select (otimização)						
|V|	**Tipo**	**Inst**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**	**Maior cor**	**#Bounds**	**#Prunes**	**#Sol.**	**#Nós**	**Tempo p/ 1^a (s)**	**Tempo Total (s)**
26	OddCycle	1	44	1176646851	4096641234	8	6101302200	0.000	10800.000	53	4564897330	316839007	3	10603481104	0.000	10800.000
		2	94	1895611489	7785217781	44	5743375749	0.000	10800.000	99	179155625	4415411739	12	8217012017	0.000	10800.000
		3	117	1900039455	7977540922	53	5700048800	0.000	10800.000	114	336266831	4293974272	9	7961956055	0.000	10800.000
	EvenCycle	1	30	91326165	90597668	19	235367863	0.390	419.074	30	276059731	269419373	5	1114770418	0.000	1131.649
		2	57	1645730092	2629876932	5	6256562220	0.000	10800.000	57	4328305486	721308565	1	10894682505	0.000	10800.000
		3	31	2795326285	1585605446	20	6088222903	0.063	10800.000	31	3337154071	2509493385	4	10823011250	0.000	10800.000
	Tree	1	46	2686041997	884280629	12	6174625648	0.000	10800.000	46	4652634796	159177582	1	10815054531	0.000	10800.000
		2	43	1841427421	1920867573	18	6427915575	0.000	10800.000	43	4315155472	631308463	1	10711895707	0.000	10800.000
		3	48	3143086399	978959526	9	6055121395	0.000	10800.000	48	4837852468	26880865	2	10683687124	0.000	10800.000

Tabela 3.6: Menor tamanho de instância onde cada algoritmo e problema de otimização atingem o tempo limite imposto de 10800 segundos (3 horas).

Algoritmo	**MinEQ-CDGP (restrições $=$)**			**MinGEQ-CDGP (restrições $\geq$)**		
	OddCycle	**EvenCycle**	**Tree**	**OddCycle**	**EvenCycle**	**Tree**
BPB-Prev	Nenhuma	22	22	22	24	24
BPB-Select	Nenhuma	24	24	22	24	22

Capítulo 4

Elaboração e implementação de modelos de programação matemática

Apesar da existência de alguns métodos exatos que podem ser utilizados no BCP, como visto na Seção 2.3 do Capítulo 2, todos são aplicados a problemas de alocação de canais com alguma variante ao invés do BCP. Assim, durante a pesquisa, verificou-se que há uma lacuna na literatura referente a resultados de otimalidade para instâncias do BCP amplamente usadas em trabalhos específicos do problema, em sua maior parte contendo algoritmos heurísticos.

Para a obtenção de soluções ótimas para algumas instâncias do BCP, foram usados modelos de programação inteira e por restrições, que serão descritos a seguir. Ressalta-se que, no Capítulo 5, serão apresentados novas formulações de programação inteira, seguindo as características dos modelos de colorações com distâncias.

Do mesmo modo que no Capítulo 3, será assumido que $\boxtimes$ é um operador relacional, de tal modo que, para dois vértices i e j, tem-se a restrição $|x(i) - x(j)| \boxtimes d_{i,j}$, onde $\odot$ é $=$ para o MinEQ-CDGP e $\geq$ para o MinGEQ-CDGP.

4.1 Programação inteira

Os modelos de programação inteira descritos e implementados foram baseados na formulação padrão para o BCP (descrita na Seção 2.1 do Capítulo 2) de Koster (1999), modificando-o reduzir a quantidade de variáveis e restrições.

4.1.1 Demandas unitárias

Considerando-se o seguinte conjunto de variáveis:

$$x_{ik} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao vértice } i; \\ 0 & \text{caso contrário.} \end{cases}$$

E uma variável livre adicional z_{max}, que corresponderá ao valor da maior cor usada, tem-se o seguinte modelo de programação inteira:

$$\text{Minimizar} \quad z_{max} \tag{4.1}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{U} x_{ik} = 1 \qquad (\forall i \in V) \tag{4.2}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ \neg(|k-m| \boxtimes d(i,j))) \tag{4.3}$$

$$z_{max} \geq k x_{ik} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{4.4}$$

$$x_{ik} \in \{0,1\} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{4.5}$$

$$\tag{4.6}$$

A função objetivo é a minimização da maior cor usada. As restrições (4.2) fazem com que cada vértice receba uma cor O conjunto de restrições (4.3) correspondem à separação entre cores de vértices adjacentes de acordo com o tipo de restrição do problema considerado. As restrições (4.4) garante que a variável z_{max} terá o valor da maior cor usada e, por fim, as restrições (4.5) são de integralidade.

O modelo modificado possui $O(U|V|)$ variáveis e $O(U^2|E| + |V|)$ restrições, sendo menor que a formulação padrão original, que tem $O(U|V| + U)$ variáveis e $O(U^2|E| + U|V| + U + |V|)$

4.1.2 Multicoloração

Para que o modelo acima possa ser usado no MinGEQ-Multi-CDGP/BMCP, deve-se modificá-lo de modo a acomodar as demandas de multicoloração. Assim, duas alterações são necessárias para que se obtenha um outro modelo para o MinGEQ-Multi-CDGP. A primeira consiste em alterar o lado direito das restrições (4.2) de 1 para $w(i)$, o que fará

com que, para cada vértice i, exatamente $w(i)$ variáveis x_{ik} tenham valor 1, o que significa que o vértice terá $w(i)$ cores distintas. A segunda modificação consiste em expandir o conjunto de restrições (C.9) de modo a adicionar novos elementos que garantirão que cada vértice i não use cores distintas que violem a distância $d_{i,i}$, ou seja, além de uma restrição para cada $(i,j) \in E;\ 1 \leq k,m \leq U$ tal que $\neg(|k-m| \geq d_{i,j})$ e também uma para cada $i \in V;\ 1 \leq k,m \leq U$ tal que $\neg(|k-m| \geq d_{i,i})$. Esta modificação equivale a adicionar uma aresta (i,i) no grafo de entrada, cujo peso associado será igual a $d(i,i)$, de tal modo que aplicar a formulação original de programação inteira ao grafo resultante desta modificação automaticamente adicionaria as novas restrições. As duas modificações citadas não tem impacto no tamanho assintótico da formulação. O modelo resultante usa as mesmas variáveis e é, então, definido por:

$$\text{Minimizar} \quad z_{max} \tag{4.7}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{U} x_{ik} = 1 \qquad (\forall i \in V) \tag{4.8}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall (i,j) \in E;\ k,m = 1,2,...,U\ :\ |k-m| < d(i,j)) \tag{4.9}$$

$$z_{max} \geq k x_{ik} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{4.10}$$

$$x_{ik} \in \{0,1\} \qquad (\forall i \in V;\ k = 1,2,...,U) \tag{4.11}$$

Ressalta-se que, como o problema considerado nesta formulação é especificamente com restrição de adjacência do tipo $\geq$ (ou seja, $\boxtimes = \geq$), substituiu-se a expressão $\not(|k-m| \boxtimes d(i,j))$ pela equivalente $|k-m| < d(i,j)$ na quantificação das restrições 4.9, da mesma forma como a formulação de Koster (1999) descrita na Seção 2.2.

4.2 Programação por restrições

Os modelos de geometria de distâncias para problemas de coloração em grafos podem ser mapeados diretamente para uma formulação de programação por restrições (*constraint programming* - CP). Isso é possível já que tais modelos envolvem a projeção do grafo em uma dimensão, ou seja, uma rotulação dos vértices do grafo usando números naturais que indicam sua posição na linha. Desse modo, as distâncias dos segmentos definidos pelos pesos das arestas devem ser respeitadas. Nesse aspecto, a técnica de programação por restrições se mostra capaz de tratar diretamente estas características.

4.2.1 Demandas unitárias

A primeira formulação utiliza em programação por restrições e é baseada diretamente nas definições mostradas no Capítulo 3. Seja x_i uma variável inteira consistindo na cor atribuída ao vértice i. Então, o modelo, denotado por CDGP-CP, é definido por:

$$\text{Minimizar} \quad \max_{i \in V} x(i) \tag{4.12}$$

$$\text{Sujeito a} \quad |x(i) - x(j)| \boxtimes d(i,j) \quad (\forall (i,j) \in E) \tag{4.13}$$

$$x(i) \in \mathbb{Z}^* \quad (\forall i \in V) \tag{4.14}$$

A função objetivo (4.12) consiste na minimização da maior cor usada dentre todos os vértices. O conjunto de restrições (4.13) envolve as distâncias entre vértices. Paraa cada variável x_i, o domínio inicial $D(x_i)$ consiste em todos os inteiros entre 1 e um dado limite superior U, ou seja, $D(x_i) = [1 \,..\, U]$. Ressalta-se que todos os domínios iniciais de todas as variáveis são iguais. Este modelo possui $O(|V|)$ variáveis (uma por vértice) e $O(|E|)$ restrições (uma por aresta).

Para o MinGEQ-CDGP-Unif (ou seja, onde todas as distâncias são iguais e as restrições são do tipo $\geq$) em grafos completos, pode-se usar uma restrição global específica. Tal modelo, denotado por MinGEQ-CDGP-Unif-CP, é definido por:

$$\text{Minimizar} \quad \max_{i \in V} x(i) \tag{4.15}$$

$$\text{Sujeito a} \quad \text{allMinDistance}(\{x(i) : i \in V\},\ \varphi) \tag{4.16}$$

$$x(i) \in \mathbb{Z}^* \quad (\forall i \in V) \tag{4.17}$$

A restrição global *allMinDistance* usa, como parâmetros, um conjunto de variáveis e um valor w e faz com que, para todos os pares de variáveis y e z no conjunto, a condição $|y - z| \geq w$ seja respeitada, o que é exigido pelo MinGEQ-CDGP. Esta formulação tem $O(|V|)$ variáveis e $O(1)$ restrição. Apesar de bastante compacta, a mesma funciona apenas em grafos completos, uma vez que a restrição global *allMinDistance* trata das distâncias entre todos os pares de variáveis dadas como parâmetros.

Ressalta-se que a *allMinDistance* generaliza a restrição global clássica *allDifferent*, de modo que se o valor w (ou seja, a distância a ser respeitada) for 1, o resultado obtido pela propagação de ambas as restrições com o mesmo conjunto de variáveis seria o mesmo.

Entretanto, não há garantia de que os algoritmos próprios para propagação de *allDifferent* seriam usados nesse cenário.

4.2.2 Multicoloração

Para o MinGEQ-Multi-CDGP, que usa multicoloração, uma formulação de programação por restrições pode ser construída com base em características de ambos os modelos anteriores. Como abordado no Capítulo 3, um problema com multicoloração pode ser convertido em uma versão com demandas unitárias transformando-se cada vértice i em uma clique de w_i vértices, cada um adjacente aos outros que eram originalmente adjacentes a i. Com base nisto, tem-se que cada vértice consiste em uma pequena sub-instância do MinGEQ-CDGP-Unif em um grafo completo, onde o grafo maior (ou seja, considerando as restrições impostas nas arestas originais entre vértices diferentes) é uma instância do MinGEQ-CDGP se as demandas de multicoloração forem ignoradas. Assim, tem-se o seguinte modelo de programação por restrições, denotado por MinGEQ-Multi-CDGP-CP:

$$\text{Minimizar} \quad \max_{\substack{i \in V \\ 1 \leq k \leq w(i)}} x(i,k) \tag{4.18}$$

$$\text{Sujeito a} \quad |x(i,k) - x(j,m)| \geq d(i,j) \qquad (\forall (i,j) \in E,\ 1 \leq k \leq w(i),\ 1 \leq m \leq w(j)) \tag{4.19}$$

$$\text{allMinDistance}(\{x(i,k) : 1 \leq k \leq w_i\},\ d(i,i)) \quad (\forall i \in V) \tag{4.20}$$

$$x(i,k) \in \mathbb{Z}^* \qquad (\forall i \in V,\ 1 \leq k \leq w(i)) \tag{4.21}$$

Na formulação acima, restrições (4.19) garantem que as cores atribuídas a vértices diferentes respeitem as distâncias das arestas. As restrições (4.19) fazem com que as cores distintas atribuídas a um mesmo vértice i respeitem a distância mínima $d_{i,i}$ entre elas (por meio da restrição global *allMinDistance*, já que a clique do vértice define uma pequena sub-instância do MinGEQ-CDGP-Unif). Esta formulação possui $O(|V| \times w_{max})$ variáveis e $O(|E| \times w_{max})$ restrições, onde $w_{max} = \max_{i \in V} w(i)$.

4.3 Experimentos computacionais

As formulações de programação inteira e por restrições foram implementadas por meio da ferramenta IBM/ILOG CPLEX 12.6.3 e seu componente CP Optimizer, com a biblioteca *Concert Technology*. A codificação foi feita em linguagem C++ e os programas resultantes foram executados em um computador com processador Intel Core i7-3770 (3.4GHz), 8GB de RAM e sistema operacional Ubuntu Linux 16.04.2 LTS. Em todos os testes, foram utilizados os parâmetros originais do CPLEX, mas limitando-se os processos a apenas uma *thread*, e cada instância foi limitada a 48 horas (172800 segundos) de tempo de CPU.

Foram usados dois conjuntos de instâncias de *benchmark* da literatura. O primeiro conjunto é conhecido como GEOM e foi gerado por Trick et al. (2002) tanto para o BCP quanto para o BMCP, e consiste em 33 instâncias de grafos geométricos de três tipos: GEOMn, que são grafos esparsos, enquanto GEOMna e GEOMnb são grafos mais densos (onde n é o número de vértices da instância tal que $n \in \{20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120\}$).

O segundo conjunto consiste em três subconjuntos de instâncias para o MS-CAP, sendo os mesmos compostos pelas instâncias clássicas Philadelphia (21 vértices) e Helsinki (25 vértices), baseadas em redes celulares das cidades de mesmo nome, e um problema artificial de 55 vértices correspondendo a uma extensão da rede Philadelphia (Chakraborty, 2001; Kendall e Mohamad, 2005). Tais instâncias são usadas como exemplos do BMCP, porém, ressalta-se que a matriz de compatibilidade de cada uma delas, que define as separações mínimas de cores entre vértices adjacentes, não é dada diretamente. Ao invés disso, é dado um outro grafo, consistindo no *layout* da rede, de onde é determinada a distância entre cada par de vértices (que pode ser obtida por algoritmos como Floyd-Warshall) e a partir destas distâncias determina-se as separações de cores para o BMCP (Audhya et al., 2011).

Ao construir os modelos de cada instância a serem resolvidos, é necessário obter um limite superior U para limitar o conjunto de cores. Para isso, aplicou-se um algoritmo guloso baseado na regra de alocação exaustiva por frequência (FEA) de Sivarajan et al. (1989). De modo a ajudar o processamento dos resolvedores, a solução obtida foi repassada inteiramente aos métodos (consistindo em um ponto inicial do CP Optimizer e um *MIP start* para o CPLEX), ao invés de somente usá-la para o limite superior. Tal estratégia

é importante, em especial, para os modelos de programação inteira com CPLEX, uma vez que isso permite que o mesmo calcule um *gap* de otimalidade logo no início do processo de enumeração.

Os primeiros resultados apresentados são para o BCP (ou seja, com demandas unitárias de cores). A Tabela 4.1 mostra os resultados para as instâncias GEOM. Para verificação dos valores obtidos e comparação com soluções heurísticas da literatura, comparou-se os resultados com o *framework* heurístico Discropt de Phan e Skiena (2002) e a busca tabu iterada com reinícios múltiplos de Lai e Lü (2013). Para todas as instâncias esparsas (ou seja, as que não contém os sufixos "a" ou "b"), a implementação de programação por restrições foi capaz de obter todas as soluções ótimas. Porém, verificou-se que, em algumas instâncias, nenhum método (incluindo as estratégias exatas abordadas aqui) foi capaz de obter as soluções de Phan e Skiena (2002), o que também foi observado anteriormente por Lai e Lü (2013) na comparação com outras heurísticas. Isso indica um possível equívoco nos resultados apresentados por Phan e Skiena (2002), que foram destacados na Tabela 4.1

A Tabela 4.2 fornece os resultados obtidos para as instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) sem as demandas de multicoloração (ou seja, considerando que cada vértice usa somente uma cor). Nota-se que, para cada matriz de compatibilidade C^i_{21} tal que i é ímpar ($i \in \{1,3,5,7\}$), ao desconsiderar-se as demandas de múltiplas cores, as instâncias se tornam iguais, já que as únicas diferenças entre elas é a separação entre cores do meesmo vértice. O mesmo ocorre para os valores pares de i ($i \in \{2,4,6,8\}$). Assim, na Tabela 4.2, os resultados foram agrupados para valores pares e ímpares de i. Novamente, a formulação de programação por restrições obteve resultados ótimos mais rapidamente, apesar de que os tempos de execução são muito pequenos devido ao tamanho de tais instâncias relaxadas.

Os próximos resultados apresentados são para o BMCP. Para instâncias deste problema, um limite inferior trivial L pode ser calculado como $L = \max[(d_{i,i} \times (w_i - 1)) + 1]$ (Chakraborty, 2001). Este valor foi inserido nos modelos adicionando-se uma restrição na qual exige-se que o valor da variável que representa a solução objetivo seja maior ou igual a L, o que permite que a enumeração seja finalizada mais rapidamente quando a solução ótima equivale ao limite inferior, em especial na programação por restrições. Isso também permite que a programação inteira forneça um *gap* de otimalidade mesmo quando o nó

Tabela 4.1: Resultados obtidos pelas formulações de programação inteira e por restrições aplicadas às instâncias GEOM para o BCP. Os valores sublinhados nas colunas "Melhor Obtido"indicam soluções ótimas.

Instância	|V|	|E|	Phan e Skiena (2002)	Lai e Lü (2013)	Prog. por Restr. (CP Optimizer)		Programação Inteira (CPLEX - B&C)			
			Melhor Reportado	Melhor Reportado	Melhor Obtido	Tempo (s)	Melhor Obtido	Melhor LI	Gap (%)	Tempo (s)
GEOM20	20	40	20*	21	21	0.00	21	21.000	0.00	0.33
GEOM20a		57	20	20	20	0.02	20	20.000	0.00	0.95
GEOM20b		52	13	13	13	0.01	13	13.000	0.00	0.09
GEOM30	30	80	27*	28	28	0.05	28	28.000	0.00	0.88
GEOM30a		111	27	27	27	0.05	27	27.000	0.00	8.06
GEOM30b		111	26	26	26	0.03	26	26.000	0.00	2.27
GEOM40	40	118	27*	28	28	0.05	28	28.000	0.00	1.97
GEOM40a		186	38	37	37	1.39	37	37.000	0.00	278.66
GEOM40b		197	36	33	33	2.06	33	33.000	0.00	252.39
GEOM50	50	177	29	28	28	0.26	28	28.000	0.00	21.44
GEOM50a		288	54	50	50	374.42	50	50.000	0.00	3457.25
GEOM50b		299	40	35	35	144.69	35	35.000	0.00	8494.52
GEOM60	60	245	34	33	33	1.12	33	33.000	0.00	45.73
GEOM60a		339	54	50	50	684.59	50	50.000	0.00	16755.65
GEOM60b		426	47	41	41	22915.94	41	41.000	0.00	134996.77
GEOM70	70	337	40	38	38	2.39	38	38.000	0.00	533.53
GEOM70a		529	64	61	61	24798.03	≤ 62	38.000	38.71	172815.55
GEOM70b		558	54	47	47	534.65	≤ 49	44.0000	10.20	172834.40
GEOM80	80	429	44	41	41	8.18	41	41.000	0.00	3019.18
GEOM80a		692	69	63	63	87770.77	≤ 65	39.0000	40.00	172803.55
GEOM80b		743	70	60	60	54320.89	≤ 66	21.0000	68.18	172800.25
GEOM90	90	531	48	46	46	55.18	46	46.000	0.00	7768.62
GEOM90a		879	74	63	63	130050.12	≤ 72	7.000	90.28	173100.57
GEOM90b		950	83	69	≤ 69	172800.00	≤ 85	2.1127	97.51	172802.83
GEOM100	100	647	55	50	50	545.79	50	50.0000	0.00	78836.94
GEOM100a		1092	84	67	≤ 70	172800.01	≤ 85	3.0863	96.37	172824.54
GEOM100b		1150	87	72	**≤ 71**	172800.02	≤ 101	2.2271	97.75	172840.38
GEOM110	110	748	59	50	50	2982.24	50	50.0000	0.00	170043.88
GEOM110a		1317	88	72	≤ 73	172800.01	≤ 97	2.1963	97.70	172811.66
GEOM110b		1366	87	78	≤ 79	172800.01	≤ 99	2.2208	97.76	172821.35
GEOM120	120	893	67	59	59	10778.18	≤ 60	24.0000	60.00	173296.11
GEOM120a		1554	101	82	≤ 84	172800.01	$\leq 110^{\dagger}$	2.1039	98.09	173181.91
GEOM120b		1611	103	84	≤ 85	172800.01	$\leq 113^{\dagger}$	2.1245	98.12	173187.16

*Resultados inferiores ao ótimo obtido - possível equívoco no trabalho correspondente.

†Nenhuma solução encontrada pelo *branch-and-cut*, mas o CPLEX retornou a solução gulosa inicial.

Tabela 4.2: Resultados obtidos pelas formulações de programação inteira e por restrições aplicadas às instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) sem demandas de multicoloração. Como a programação inteira obteve todas as soluções ótimas, a coluna de Melhor LI foi omitida.

Mat. de Compat.	**|V|**	**|E|**	**Prog. por Restr. (CP Optimizer)**		**Prog. Inteira (CPLEX - B&C)**	
			Melhor Obtido	**Tempo (s)**	**Melhor Obtido**	**Tempo (s)**
C^1_{21}, C^3_{21}, C^5_{21} and C^7_{21}	21	102	7	0.40	7	0.87
C^2_{21}, C^4_{21}, C^6_{21} and C^8_{21}			9	0.06	9	2.66
C^1_{25}	25	134	8	4.71	8	1.90
C^1_{55}	55	362	7	0.79	7	30.63

raiz do *branch-and-cut* usa muito tempo de CPU até mesmo na relaxação linear.

A Tabela 4.3 fornece os resultados obtidos para as instâncias GEOM, agora considerando demandas de multicoloração. Os resultados obtidos para tais instâncias foram verificados e comparados com a busca tabu iterada com reinícios múltiplos de Lai e Lü (2013), na qual o algoritmo para o BCP é aplicado no BMCP expandindo-se os vértices do grafo em cliques, como já descrito no Capítulo 2. Já a Tabela 4.4 apresenta os resultados obtidos para as instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) em suas formas originais (ou seja, incluindo multicoloração), que foram comparados com a heurística construtiva de Chakraborty (2001), o algoritmo de busca local de Kendall e Mohamad (2005) e o algoritmo memético de Kim et al. (2007).

No BMCP, a relação de eficiência entre programação inteira e por restrições é praticamente invertida em relação ao BCP: a maior parte das instâncias é resolvida a otimalidade mais rapidamente utilizando programação inteira que por restrições. Com isso, foi possível obter soluções ótimas para todas as instâncias do MS-CAP usando o modelo de programação inteira. Isso ocorre porque, como discutido anteriormente, tal modelo é capaz de considerar demandas de multicoloração sem aumentar a quantidade de vértices, sendo necessário apenas alterar as restrições que indicam quantas cores um vértice deve receber e adicionar um outro conjunto de restrições que trata da separação de cores atribuídas ao mesmo vértice, sendo que tal conjunto tem tamanho linear em relação ao número de vértices. Em contrapartida, o modelo de programação por restrições acaba crescendo mais rapidamente, uma vez que a quantidade de variáveis e de restrições aumenta a cada cor adicional necessária, tornando o modelo pseudopolinomial. A Figura 4.1 mostra graficamente esta diferença de eficiência entre os métodos tendo como base as instâncias GEOM.

Tabela 4.3: Resultados obtidos pelas formulações de programação inteira e por restrições aplicadas às instâncias GEOM para o BMCP. Os valores sublinhados nas colunas "Melhor Obtido"indicam soluções ótimas.

Instância	|V|	|E|	LI Trivial	Lai e Lü (2013)	Prog. por Restr. (CP Optimizer)		Programação Inteira (CPLEX - B&C)			
				Melhor Reportado	Melhor Obtido	Tempo (s)	Melhor Obtido	Melhor LI	Gap (%)	Tempo (s)
GEOM20	20	40	91	149	≤ 149	172800.01	<u>149</u>	149.0000	0.00	15.17
GEOM20a		57	91	169	≤ 169	172800.01	<u>169</u>	169.0000	0.00	18.49
GEOM20b		52	21	44	<u>44</u>	476.92	<u>44</u>	44.0000	0.00	1.58
GEOM30	30	80	91	160	≤ 160	172800.01	≤ 160	159.0000	0.62	172830.72
GEOM30a		111	91	209	≤ 215	172800.01	≤ 211	189.0000	10.43	172813.47
GEOM30b		111	21	77	≤ 77	172800.00	<u>77</u>	77.0000	0.00	41.87
GEOM40	40	118	91	167	≤ 168	172800.01	<u>167</u>	167.0000	0.00	1192.28
GEOM40a		186	91	213	≤ 225	172800.01	<u>213</u>	213.0000	0.00	111262.08
GEOM40b		197	21	74	≤ 74	172800.00	<u>74</u>	74.0000	0.00	17027.77
GEOM50	50	177	91	224	≤ 226	172800.02	<u>224</u>	224.0000	0.00	52450.85
GEOM50a		288	91	314	≤ 332	172800.03	≤ 361	95.5218	73.54	172820.13
GEOM50b		299	21	83	≤ 85	172800.00	≤ 87	52.0000	40.23	172819.47
GEOM60	60	245	91	258	≤ 259	172800.02	<u>258</u>	258.0000	0.00	156987.80
GEOM60a		339	91	356	≤ 380	172800.03	≤ 448	93.5801	78.93	172813.01
GEOM60b		426	21	113	≤ 117	172800.01	≤ 125	34.0000	72.80	172897.07
GEOM70	70	337	91	270	≤ 284	172800.03	≤ 305	94.2092	69.11	172804.56
GEOM70a		529	91	467	≤ 483	172800.05	≤ 578	91.0000	84.26	172839.51
GEOM70b		558	21	116	≤ 123	172800.01	≤ 134	22.7359	83.03	172805.88
GEOM80	80	429	91	381	≤ 395	172800.04	≤ 511	95.2644	80.19	172809.70
GEOM80a		692	91	361	≤ 382	172800.05	≤ 479	91.0000	81.00	172885.02
GEOM80b		743	21	139	≤ 145	172800.01	≤ 170	23.0547	86.44	172820.56
GEOM90	90	531	91	330	≤ 342	172800.05	≤ 423	93.2736	77.73	172811.82
GEOM90a		879	91	375	≤ 392	172800.07	≤ 452	91.0000	79.87	172830.60
GEOM90b		950	21	144	≤ 156	172800.01	≤ 212	22.2574	89.50	172844.07
GEOM100	100	647	91	404	≤ 426	172800.07	≤ 493	94.2797	80.88	173190.69
GEOM100a		1092	91	442	≤ 465	172800.08	≤ 596	91.0000	84.73	172871.84
GEOM100b		1150	21	156	≤ 169	172800.02	≤ 220	21.0000	90.45	172810.33
GEOM110	110	748	91	381	≤ 399	172800.07	≤ 500	91.0000	81.80	172840.98
GEOM110a		1317	91	488	≤ 527	172800.10	≤ 610	91.0000	85.08	173095.42
GEOM110b		1366	21	204	≤ 207	172800.01	≤ 250	22.0001	91.20	172835.82
GEOM120	120	893	91	396	≤ 427	172800.06	≤ 505	93.9762	81.39	172923.18
GEOM120a		1554	91	554	≤ 585	172800.16	≤ 641	91.0000	85.80	173312.14
GEOM120b		1611	21	189	≤ 202	172800.01	≤ 247	21.8723	91.14	172852.82

Apesar de a programação inteira tender a ser mais eficiente para o BMCP, há uma vantagem do modelo de programação por restrições: quando não é possível obter a solução ótima no tempo limite por nenhum dos dois métodos, a solução retornada pela programação por restrições tem qualidade melhor que a achada pela programação inteira (ou seja, a maior cor usada é menor). Isso ocorre porque a programação por restrições tem informações explícitas sobre quais cores cada vértice pode assumir, ao invés de calcular as mesmas como ocorre na outra abordagem de programação inteira.

Tabela 4.4: Resultados obtidos pelas formulações de programação inteira e por restrições aplicadas às instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) com multicoloração. Como a programação inteira obteve todas as soluções ótimas, a coluna de Melhor LI foi omitida.

Mat. de Compat.	Vet. Dem.	|V|	|E|	LI	Chakraborty (2001)	Kendall e Mohamad (2005)	Kim et al. (2007)	Prog. por Restr. (CP Optimizer)		Prog. Inteira (CPLEX - B&C)	
					Melhor Reportado	Melhor Reportado	Melhor Reportado	Melhor Obtido	Tempo (s)	Melhor Obtido	Tempo (s)
C_{21}^1	D_{21}^1	21	102	533	533	533	533	533	4.20	533	0.50
C_{21}^1	D_{21}^2			309	309	309	309	309	1.34	309	1.22
C_{21}^2	D_{21}^1			533	533	533	533	533	10.53	533	308.04
C_{21}^2	D_{21}^2			309	309	309	309	309	625.93	309	165.54
C_{21}^3	D_{21}^1			457	457	457	–	457	3.96	457	0.39
C_{21}^3	D_{21}^2			265	265	265	–	265	3.54	265	1.52
C_{21}^4	D_{21}^1			457	457	457	–	457	41.24	457	202.01
C_{21}^4	D_{21}^2			265	265	265	–	≤ 266	172800.06	265	214.01
C_{21}^5	D_{21}^1			381	381	381	381	381	3.23	381	0.29
C_{21}^5	D_{21}^2			221	221	221	221	221	100.81	221	5.09
C_{21}^6	D_{21}^1			381	463	435	427	≤ 449	172800.08	427	6827.49
C_{21}^6	D_{21}^2			221	273	268	253	≤ 266	172800.04	253	2026.67
C_{21}^7	D_{21}^1			305	305	305	–	305	12.85	305	1.10
C_{21}^7	D_{21}^2			177	197	185	–	≤ 180	172800.06	180	24.54
C_{21}^8	D_{21}^1			305	465	444	–	≤ 435	172800.07	427	1185.27
C_{21}^8	D_{21}^2			177	278	271	–	≤ 267	172800.06	253	1116.45
C_{25}^1	D_{25}^3	25	134	21	73	73	–	≤ 73	172800.00	73	1.10
C_{25}^1	D_{25}^4			89	121*	200	–	≤ 200	172800.07	200	2.18
C_{55}^1	D_{55}^5	55	362	309	309	309	–	309	11078.95	309	460.12
C_{55}^1	D_{55}^6			71	79	72	–	71	6.33	71	28.56

*Resultados inferiores ao ótimo obtido - possível equívoco no trabalho correspondente.

De maneira similar ao que ocorreu nos testes para o BCP, também detectou-se um possível equívoco nos resultados reportados por Chakraborty (2001), onde o autor fornece um resultado heurístico para a instância composta pela matriz de compatibilidade C_{25}^1 e vetor de demandas de cores D_{25}^4 melhor (com valor de função objetivo 121) que as soluções exatas obtidas (com valor de fumção objetivo 200). Nenhum outro resultado da literatura foi capaz de obter soluções com função objetivo melhor que 200.

Com a finalidade de tentar identificar padrões no funcionamento dos algoritmos de resolução de acordo com as instâncias usadas, para cada uma destas, foi determinada a quantidade de ramificações (*branches*) e falhas (caminhos que não atingem solução factível) na enumeração da programação por restrições, bem como o número de cortes gerados de cada tipo pelo *branch-and-cut* usado no modelo de programação inteira. Tais valores são fornecidos nas Tabelas 4.5 e 4.6 para o BCP, e nas Tabelas 4.7 e 4.8 para o BMCP. Não foi possível encontrar uma correlação clara entre cada tipo e tamanho de instância e estes dados, o que indica que tais partes dos algoritmos são mais sensíveis aos valores das separações mínimas necessárias entre cores e às demandas exigidas nas

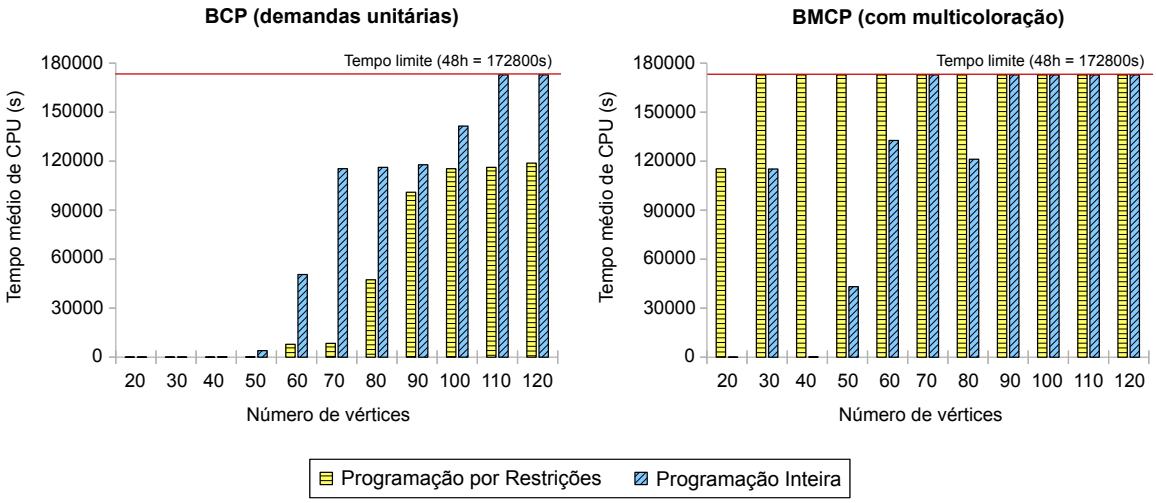


Figura 4.1: Número de vértices $\times$ tempo de CPU necessário para encontrar a solução ótima (até o tempo limite) para cada método nas instâncias GEOM.

instâncias. Conjectura-se que o formato do grafo de entrada também tem impacto na enumeração, uma vez que certas desigualdades válidas, como cortes clique (Méndez-Díaz e Zabala, 2006), são derivadas de acordo com o grafo a ser colorido. Tal característica pôde ser observada no modelo baseado em orientação proposto nesta tese, como será visto mais adiante.

4.4 Notas do capítulo

Com a finalidade de obter soluções exatas para instâncias do BCP e do BMCP, foram implementadas formulações de programação inteira e por restrições no resolvedor CPLEX. As soluções obtidas foram comparadas com resultados da literatura, de modo a verificar a otimalidade das mesmas. A principal motivação para isso foi a falta de trabalhos na literatura envolvendo resultados obtidos a partir de abordagens exatas para o BCP e BMCP.

Os resultados obtidos permitiram provar a otimalidade de diversas soluções, além de encontrar um novo limite superior para uma das instâncias usadas. Também foi possível identificar algumas inconsistências em resultados reportados na literatura, onde valores heurísticos forneceram limites superiores abaixo da solução ótima, o que é uma contradição.

Uma comparação entre as modelagens de programação inteira e por restrições utilizadas mostrou que não há um método absolutamente melhor para todos os casos. Devido

Tabela 4.5: Número de ramificações e falhas (para programação por restrições) e de desigualdades válidas geradas de cada tipo (para programação inteira) para o BCP com as instâncias GEOM (demandas unitárias).

Instância	|V|	Prog. por Restr. (CP Optimizer)		Prog. Inteira (CPLEX)				
		# Ramificações	# Falhas	# Cortes Clique	# Cortes *Implied Bound*	# Cortes *MIP Rounding*	# Cortes *Zero-Half*	# Cortes Gomory
GEOM20		972	462	9	15	0	8	10
GEOM20a	20	4667	2298	3	27	0	2	9
GEOM20b		2224	1096	14	7	0	5	0
GEOM30		10102	4782	13	62	0	4	31
GEOM30a	30	9024	4353	25	473	0	20	60
GEOM30b		6651	3210	9	194	0	11	49
GEOM40		9965	4699	13	130	0	3	56
GEOM40a	40	206960	98635	43	4	117	3	1
GEOM40b		275519	131721	14	543	0	30	15
GEOM50		57869	27159	27	0	0	9	0
GEOM50a	50	40966958	19591854	227	13	138	12	0
GEOM50b		14438962	6973218	564	3	6	18	0
GEOM60		178478	82471	63	0	0	0	2
GEOM60a	60	59350249	28292260	241	19	182	28	24
GEOM60b		1700740733	807817043	1691	25	176	23	0
GEOM70		320560	148877	128	0	12	3	1
GEOM70a	70	1662200599	781294815	881	6	192	20	0
GEOM70b		301138496	143985463	1171	21	226	46	5
GEOM80		2173324	1008986	372	5	131	19	0
GEOM80a	80	8859155916	4149659761	761	30	389	85	5
GEOM80b		3687195162	1739030200	480	29	303	138	0
GEOM90		3841482	1748958	471	2	223	31	9
GEOM90a	90	8424930433	3953124503	143	3	411	349	0
GEOM90b		6454145085	3036820947	317	11	417	93	0
GEOM100		33141115	15198269	702	11	221	54	27
GEOM100a	100	6622094014	3084753118	330	2	1367	2162	245
GEOM100b		5409742123	2511274478	130	301	0	567	136
GEOM110		12496119255	5760860721	426	16	320	54	2
GEOM110a	110	5930484572	2724545532	151	501	0	619	79
GEOM110b		4177753606	1922426902	118	754	0	460	0
GEOM120		637959908	289378147	1015	19	357	128	8
GEOM120a	120	5560296354	2542244856	172	782	0	678	0
GEOM120b		4003420813	1841115383	273	1258	0	959	0

a características dos problemas, a programação por restrições é melhor para o BCP, enquanto a programação inteira tem vantagem para o BMCP.

No Capítulo a seguir, serão mostradas novas formulações de programação inteira propostas nesta tese.

Tabela 4.6: Número de ramificações e falhas (para programação por restrições) e de desigualdades válidas geradas de cada tipo (para programação inteira) para o BCP com as instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) e demandas unitárias.

Instância	|V|	Prog. por Restr. (CP Optimizer)		Prog. Inteira (CPLEX)				
		# Ramificações	# Falhas	# Cortes Clique	# Cortes *Implied Bound*	# Cortes *MIP Rounding*	# Cortes *Zero-Half*	# Cortes Gomory
C_{21}^1, C_{21}^3, C_{21}^5 and C_{21}^7	21	49177	24602	3	5	0	9	0
C_{21}^2, C_{21}^4, C_{21}^6 and C_{21}^8		6033	2970	5	5	0	58	3
C_{25}^1	25	176565	87496	0	0	0	0	0
C_{55}^1	55	49096	24214	0	0	0	0	0

Tabela 4.7: Número de ramificações e falhas (para programação por restrições) e de desigualdades válidas geradas de cada tipo (para programação inteira) para o BMCP com as instâncias GEOM (multicoloração).

Instância	|V|	Prog. por Restr. (CP Optimizer)		Prog. Inteira (CPLEX)				
		# Ramificações	# Falhas	# Cortes Clique	# Cortes *Implied Bound*	# Cortes *MIP Rounding*	# Cortes *Zero-Half*	# Cortes Gomory
GEOM20		1847656377	823996186	26	33	0	22	0
GEOM20a	20	2060224899	926128135	35	378	0	92	91
GEOM20b		23433594	11236673	19	28	0	47	49
GEOM30		1594445634	702092983	345	1	7	29	0
GEOM30a	30	778845728	343378790	1234	6	189	3	0
GEOM30b		4336696005	2043720645	23	96	0	17	53
GEOM40		1101215203	477955956	186	287	0	158	0
GEOM40a	40	853750393	376905743	131	7	322	4	0
GEOM40b		2915214186	1378502269	361	1	4	2	0
GEOM50		858223844	366277155	575	0	79	31	0
GEOM50a	50	373860395	159472193	183	446	0	235	4
GEOM50b		2205618883	1031493835	1498	2	295	7	0
GEOM60		884613825	374341633	468	1	34	38	0
GEOM60a	60	327100218	135221648	1109	335	0	1213	0
GEOM60b		1625337918	741993923	143	3	456	7	0
GEOM70		480579106	200225715	497	1363	0	275	0
GEOM70a	70	178618063	71625162	0	0	0	0	0
GEOM70b		1153010252	520892432	133	2	645	165	29
GEOM80		491579072	131225337	786	1842	0	1343	85
GEOM80a	80	206713326	84260206	0	0	0	0	0
GEOM80b		972753631	434139515	328	1065	0	690	97
GEOM90		289804704	110251804	1321	0	0	1894	0
GEOM90a	90	203904382	82252836	0	0	0	0	0
GEOM90b		677312582	300211365	344	481	0	877	0
GEOM100		219953270	80390988	0	0	0	0	0
GEOM100a	100	134018202	52245146	0	0	0	0	0
GEOM100b		511570190	224573165	0	0	0	0	0
GEOM110		170668054	64299043	0	0	0	0	0
GEOM110a	110	113908674	41331333	0	0	0	0	0
GEOM110b		409339582	177985395	0	0	0	0	0
GEOM120		207980564	81060477	0	0	0	0	0
GEOM120a	120	82058712	30432596	0	0	0	0	0
GEOM120b		386743755	164997551	0	0	0	0	0

Tabela 4.8: Número de ramificações e falhas (para programação por restrições) e de desigualdades válidas geradas de cada tipo (para programação inteira) para o BMCP com as instâncias do MS-CAP (Philadelphia, Helsinki e artificiais) e multicoloração.

Matr. de Compat.	Vet. Dem.	|V|	Const. Prog. (CP Optimizer)		Integer Progr. (CPLEX)				
			# Ramificações	# Falhas	# Cortes Clique	# Cortes *Implied Bound*	# Cortes *MIP Rounding*	# Cortes *Zero-Half*	# Cortes Gomory
C_{21}^{1}	D_{21}^{1}	21	2555	504	0	0	0	0	0
C_{21}^{1}	D_{21}^{2}		469	4	0	0	0	0	0
C_{21}^{2}	D_{21}^{1}		6610	1505	0	0	0	0	0
C_{21}^{2}	D_{21}^{2}		822377	309790	0	0	0	0	0
C_{21}^{3}	D_{21}^{1}		2451	504	0	0	0	0	0
C_{21}^{3}	D_{21}^{2}		2874	505	0	0	0	0	0
C_{21}^{4}	D_{21}^{1}		20982	5811	0	0	0	0	0
C_{21}^{4}	D_{21}^{2}		44845827	17108454	0	0	0	0	0
C_{21}^{5}	D_{21}^{1}		2305	504	0	0	0	0	0
C_{21}^{5}	D_{21}^{2}		158368	50239	0	0	0	0	0
C_{21}^{6}	D_{21}^{1}		180812651	65917318	160	4	210	116	0
C_{21}^{6}	D_{21}^{2}		347622756	121153772	170	1	37	159	0
C_{21}^{7}	D_{21}^{1}		4106	505	0	0	0	0	0
C_{21}^{7}	D_{21}^{2}		352244757	132070514	4	0	0	8	35
C_{21}^{8}	D_{21}^{1}		199948906	75399919	163	1	379	224	0
C_{21}^{8}	D_{21}^{2}		451729822	146753868	225	204	0	190	0
C_{25}^{1}	D_{25}^{3}	25	2435656356	1064961796	2	134	0	81	143
C_{25}^{1}	D_{25}^{4}		374343491	115700707	0	0	0	0	0
C_{55}^{1}	D_{55}^{5}	55	7809104	2371461	0	0	0	0	0
C_{55}^{1}	D_{55}^{6}		27897	9552	0	0	0	0	0

Capítulo 5

Formulações de PI e contribuições em combinatória poliédrica

Neste capítulo, serão apresentados dois modelos de programação inteira desenvolvidos nesta pesquisa, sendo um, baseado em orientação, para o BCP e outro, baseado em distâncias, específico para a coloração clássica, mas derivado do anterior. Tais formulações são derivadas de características observadas nos modelos de colorações com distâncias. Ressalta-se que o modelo baseado em orientação é o primeiro para o BCP com tamanho polinomial, como será discorrido no capítulo.

Para cada formulação, são apresentadas contribuições em combinatória poliédrica, de modo que são mostradas condições para que a dimensão do poliedro correspondente seja plena, bem como duas famílias de desigualdades válidas, com condições para que as mesmas sejam indutoras de facetas. Além disso, um método *cut-and-branch* foi desenvolvido com o modelo baseado em orientação, para o qual são apresentados resultados de experimentos computacionais, bem como uma comparação empírica do método e do modelo com o padrão da literatura.

5.1 Modelo baseado em orientação

Como visto anteriormente no Capítulo 2, os modelos de programação inteira disponíveis na literatura para o BCP consistem em variações da formulação padrão para a coloração clássica, onde as restrições são alteradas de modo a acomodar as separações mínimas entre cores de vértices adjacentes. Como tais modelos têm dimensão pseudopolinomial,

torna-se interessante a busca por formas de escrever o problema com programação inteira que levem a formulações com melhores dimensões.

Borndörfer et al. (1998) propuseram um modelo baseado em orientação para o problema de alocação de canais de mínima interferência (*Minimum Interference Channel Assignment Problem* - MI-CAP), que é conceitualmente similar ao MS-CAP, porém, o espectro eletromagnético (ou seja, o conjunto de cores) é fixo e fornecido na instância, e as soluções podem violar as separações mínimas, levando a interferências (ao contrário do BCP e MS-CAP, onde o conjunto de cores que leve à soluções sem violações pode ser determinado por meio de heurísticas). Assim, o objetivo do MI-CAP é encontrar a alocação de canais com menor quantidade de interferências. Uma variação desta formulação foi proposta por Delle-Donne (2016), onde o objetivo é minimizar a interferência geradas entre canais (ou seja, cores) adjacentes, isto é, quando dois vértices i e j usam cores k e m tais que $|k-m|=1$.

Com base nas formulações de Borndörfer et al. (1998) e de Delle-Donne (2016), desenvolveu-se um novo modelo de programação inteira para o BCP, podendo também ser usado em coloração clássica. São usados dois conjuntos de variáveis: o primeiro define uma variável x_i para todo vértice $i \in V$, que indica a cor utilizada pelo mesmo. O segundo conjunto consiste nas variáveis binárias:

- $y_{ij} = \begin{cases} 1 & \text{se } x_i < x_j; \\ 0 & \text{caso contrário.} \end{cases}$
- z_{max}: maior cor usada na solução.

Desse modo, tem-se o seguinte modelo baseado em orientação para o BCP:

$$\text{Minimizar} \quad z_{max} \tag{5.1}$$

$$\text{Sujeito a} \quad x_i + d(i,j) \leq x_j + s(1-y_{ij}) \quad \forall (i,j) \in E,\ i<j \tag{5.2}$$

$$x_j + d(i,j) \leq x_i + s y_{ij} \quad \forall (i,j) \in E,\ i<j \tag{5.3}$$

$$z_{max} \geq x_i \quad \forall i \in V \tag{5.4}$$

$$x_i \in \mathbb{Z} \quad \forall i \in V \tag{5.5}$$

$$x_i \geq 1 \quad \forall i \in V \tag{5.6}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in E,\ i<j \tag{5.7}$$

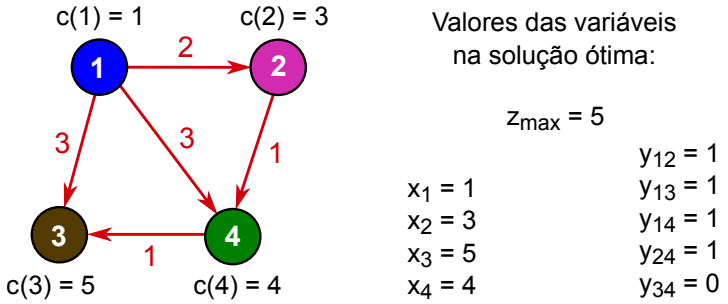


Figura 5.1: Exemplo de instância do BCP cuja solução é codificada de acordo com o modelo baseado em orientação. No grafo, também é mostrada a orientação induzida.

A Figura 5.1 mostra um exemplo de instância resolvida pelo modelo e a orientação gerada no grafo de entrada. Na formulação, s é uma constante suficientemente grande. A função objetivo consiste na minimização da variável contendo o valor da maior cor usada. As restrições (5.2) e (5.3) correspondem à separação mínima de cores entre vértices adjacentes e induzem à orientação do grafo de entrada. Assim, o conjunto (5.2) garante que $y_{ij} = 1$ se $x_i < x_j$ para toda aresta $(i, j) \in E$. Por outro lado, se $x_j < x_i$, então as restrições (5.3) fazem com que $y_{ij} = 0$. Ambas as restrições também garantem que $|x_i - x_j| \geq d(i, j)$. O conjunto de restrições (5.4) impõe que z_{max} tenha um valor maior ou igual a todas as cores usadas na solução. Por fim, as restrições (5.5) a (5.7) definem tipos e limites das variáveis.

O modelo baseado em orientação tem $O(|V| + |E|)$ variáveis e $O(|V| + |E|)$ restrições, tendo tamanho linear em relação ao grafo de entrada, independentemente dos valores das distâncias impostas nas arestas. Como já dito, a constante s, que é baseada no método de *big-M* para modelagem em programação linear, deve ter um valor suficientemente grande, de modo que s seja maior ou igual ao valor (ainda desconhecido) de $\chi(G, d)$ (que é o valor da maior cor usada na solução ótima de uma instância dada pelo grafo G e função de distâncias d). Desse modo, s deve ser um limite superior, que pode ser obtido por heurísticas primais simples. Apesar da possibilidade de um valor muito alto para s criar instabilidades numéricas, é possível definir limites inferiores para a constante a partir dos quais diversas características podem ser observadas, como visto a seguir.

Seja $PO(G, d, s)$ o fecho convexo de soluções válidas do modelo baseado em orientação, ou seja, dos pontos $(x, y, z_{max}) \in \mathbb{R}^{|V|+|E|+1}$ que satisfazem as restrições (5.2) a (5.7) (Marenco, 2005). Além disso, seja $d_{max} = \max_{(i,j)\in E} d(i, j)$. Desse modo, tem-se o seguinte resultado:

Teorema 5.1. *Se $s \geq \chi(G,d) + 2d_{max}$, então $PO(G,d,s)$ tem dimensão plena.*

Prova. Para demonstrar tal propriedade, deve-se provar que, se $\lambda_x^T x + \lambda_y^T y = \lambda_0$ para todo $(x,y) \in PO(G,d,s)$ (onde $\lambda_x \in \mathbb{R}^{|V|}$, $\lambda_y \in \mathbb{R}^{|E|}$ e $\lambda_0 \in \mathbb{R}$), ou seja, $\sum_{i \in V} \lambda_{x_i} x_i + \sum_{(i,j) \in E} \lambda_{y_{ij}} y_{ij} = \lambda_0$, então $\lambda_x = 0$ e $\lambda_y = 0$. O restante da prova será dividido em duas partes.

Para a primeira parte, seja $(x,y) \in PO(G,d,s) \cap \mathbb{R}^{|V|+|E|}$ uma solução na qual $x_i \in [1, s-1]$ para todo $i \in V$. A partir dela, construa um grafo direcionado acíclico $G^* = (V^*, E^*)$, onde $V^* = V$ e $(i,j) \in E^*$ se, e somente se, $(i,j) \in E$ e $x_i > x_j$ (tal grafo é igual a G, mas orientado de forma inversa a induzida pelas variáveis y_{ij}). Seja $A = (a_1, a_2, \ldots, a_{|V|})$ uma ordenação topológica de G^*. Com base em A, construa $|V|$ novas soluções $(x,y)^1, (x,y)^2, \ldots, (x,y)^{|V|}$, onde a k-ésima solução consiste em fixar $x^k_{a_\ell} = x_{a_\ell} + 1$ para todo $\ell = 1, \ldots, k$. Todas essas soluções são factíveis, já que quando um vértice a_ℓ tem sua cor incrementada, seus vizinhos que usam cores maiores que x_{a_ℓ} já tiveram suas cores incrementadas, pois os mesmos vêm antes de a_ℓ na ordenação topológica. Isso também garante que a orientação será mantida, de modo que todas as variáveis y mantêm seus valores originais.

Considere um par de soluções $(x,y)^k$ e $(x,y)^{k+1}$ dentre as geradas. Seguindo o procedimento de obtenção das mesmas, a única diferença entre elas é o valor da variável x_{a_k}, de modo que pode-se derivar o seguinte:

$$\begin{aligned} \sum_{i \in V \setminus a_k} \lambda_{x_i} x_i + \lambda_{x_{a_k}} x^k_{a_k} + \sum_{(i,j) \in E} \lambda_{y_{ij}} y_{ij} &= \lambda_0 \\ - \sum_{i \in V \setminus a_k} \lambda_{x_i} x_i + \lambda_{x_{a_k}} x^{k+1}_{a_k} + \sum_{(i,j) \in E} \lambda_{y_{ij}} y_{ij} &= \lambda_0 \\ \hline \lambda_{x_{a_k}} (x^k_{a_k} - x^{k+1}_{a_k}) &= 0 \end{aligned}$$

Como $x^k_{a_k} \in [1, s-1]$ e $x^{k+1}_{a_k} = x^k_{a_k} + 1$, pode-se concluir que $\lambda_{x_{a_k}} = 0$. Além disso, como a_k é um vértice arbitrário, tem-se que $\lambda_{x_i} = 0$ para todo $i \in V$, ou seja, $\lambda_x = 0$.

Para a segunda parte da prova, seja $(x,y) \in PO(G,d,s) \cap \mathbb{R}^{|V|+|E|}$ uma solução tal que $\max_{i \in V} x(i) = \chi(G,d)$. Selecione uma aresta arbitrária $(i,j) \in E$ e construa uma nova solução $(x,y)^1$ da seguinte forma:

- $x_k^1 = \begin{cases} \chi(G,d) + d_{\max} & \text{se } k = i; \\ \chi(G,d) + 2d_{\max} & \text{se } k = j; \\ x_k & \text{caso contrário.} \end{cases}$

- $y_{uv}^1 = \begin{cases} 1 & \text{se } v \in \{i,j\} \text{ ou } (u,v) = (i,j); \\ y_{uv} & \text{caso contrário.} \end{cases}$

Com base em $(x,y)^1$, construa outra solução $(x,y)^2$ trocando as cores de i e j entre si, ou seja, $x_i^2 = x_j^1$ e $x_j^2 = x_i^1$. Isso também inverte a orientação imposta na aresta (i,j), de modo que como $y_{ij}^1 = 1$, então $y_{ij}^2 = 0$. Entre $(x,y)^1$ e $(x,y)^2$, as únicas diferenças são os valores das variáveis x_i, x_j e y_{ij}. Assim, levando-se em conta o resultado da primeira parte da prova (ou seja, que $\lambda_{x_i} = 0$ para todo $i \in V$), pode-se derivar o seguinte:

$$\begin{array}{rl} & \sum_{u \in V \setminus \{i,j\}} \cancelto{0}{\lambda_{x_u} x_u} + \cancelto{0}{\lambda_{x_i} x_i^1} + \cancelto{0}{\lambda_{x_j} x_j^1} + \sum_{(u,v) \in E \setminus (i,j)} \lambda_{y_{ij}} y_{ij}^1 + \lambda_{y_{ij}} y_{ij}^1 = \lambda_0 \\ - & \sum_{u \in V \setminus \{i,j\}} \cancelto{0}{\lambda_{x_u} x_u} + \cancelto{0}{\lambda_{x_i} x_i^2} + \cancelto{0}{\lambda_{x_j} x_j^2} + \sum_{(u,v) \in E \setminus (i,j)} \lambda_{y_{ij}} y_{ij}^2 + \lambda_{y_{ij}} y_{ij}^2 = \lambda_0 \\ \hline & \lambda_{y_{ij}} (y_{ij}^1 - y_{ij}^2) = 0 \end{array}$$

Como $y_{ij}^1 = 1$ e $y_{ij}^2 = 0$, pode-se concluir que $\lambda_{y_{ij}} = 0$. Além disso, como $(i,j) \in E$ é uma aresta arbitrária, então tem-se que $\lambda_{y_{ij}} = 0$ para todo $(i,j) \in E$, ou seja, $\lambda_y = 0$.

Por fim, tem-se que $\lambda_x = 0$ e $\lambda_y = 0$, o que, como descrito no início da prova, indica que o poliedro tem dimensão plena. Logo, pode-se concluir que se $s \geq \chi(G,d) + 2d_{\max}$, então $PO(G,d,s)$ tem dimensão plena. □

5.1.1 Desigualdades válidas

Para o modelo baseado em orientação, pode-se derivar desigualdades válidas que reforçam as restrições de distâncias do modelo, bem como os limites das variáveis x_i. Com base nas desigualdades que definem facetas apresentadas por Delle-Donne (2016) para o modelo de orientação desenvolvido para a alocação de canais com minimização de interferências entre canais adjacentes (ou problema de coloração de vértices de mínima adjacência - *minimum-adjacency vertex coloring problem*), obteve-se algumas desigualdades

adaptadas que definem facetas para $PO(G,d,s)$. Para que tais desigualdades válidas possam ser definidas, deve-se adicionar as seguintes variáveis que complementam o conjunto y do modelo baseado em orientação:

- $y_{ji} = \begin{cases} 1 & \text{se } x_i < x_j; \\ 0 & \text{caso contrário.} \end{cases}$

Além disso, deve-se introduzir no modelo o seguinte conjunto de restrições:

$$y_{ij} + y_{ji} = 1 \quad \forall (i,j) \in E,\ i < j \tag{5.8}$$

A adição de tais variáveis e restrições pode ser substituída pelo uso da expressão $1 - y_{ij}$ onde houver y_{ji} tal que $i > j$.

5.1.1.1 Desigualdade clique

O primeiro conjunto de desigualdades válidas para $PO(G,d,s)$ é derivado a partir de cliques do grafo de entrada (Marenco e Wagler, 2009), como definido a seguir.

Definição 5.1.1. *Seja $i \in V$ um vértice do grafo e $K \subseteq N(i)$ uma clique. Para todo $j \in K$, seja $\delta^i_K(j) = \min_{k \in K \cup \{i\} \setminus \{j\}} d(j,k)$. Assim, a expressão:*

$$x_i \ \geq \ \sum_{j \in K} \delta^i_K(j)\, y_{ji} + 1 \tag{5.9}$$

define a desigualdade clique associada ao vértice i e à clique K.

A Figura 5.2 mostra um exemplo de grafo onde é selecionado um vértice i e uma clique $K \subseteq N(i)$ no mesmo e é dada a desigualdade clique correspondente. A expressão fornece um limite inferior mais forte para as variáveis x_i, de modo que vértices de K usem cores menores que x_i. A expressão generaliza a desigualdade clique usada na coloração clássica, definida como $x_i \ \geq \ \sum_{j \in K} y_{ji}$. Os coeficientes das variáveis y_{ji} são modificados de forma a tornar a restrição mais forte. A validade da expressão é mantida, como pode ser visto a seguir.

Teorema 5.2. *A desigualdade clique (5.9) é válida para $PO(G,d,s)$.*

Prova. Seja $(\bar{x},\bar{y}) \in PO(G,d,s)$ uma solução inteira e considere $L = \{j \in K : \bar{y}_{ji} = 1\}$. Assuma que $L = \{j_1,\ldots,j_t\}$, onde $x_{j_\ell} < x_{j_{\ell+1}}$ para cada $\ell = 1,\ldots,t-1$, e defina $j_{t+1} = i$. Como $x_{j_1} \geq 1$ e $x_{j_\ell} + d_{j_\ell j_{\ell+1}} \leq x_{j_{\ell+1}}$ para cada $\ell = 1,\ldots,t$ (já que cada par de vértices consecutivos em L são adjacentes), então

$$x_i \;\geq\; 1 + \sum_{\ell=1}^{t} d_{j_\ell j_{\ell+1}} \;\geq\; 1 + \sum_{\ell=1}^{t} \delta_K^i(j_\ell) \;=\; 1 + \sum_{j\in K} \delta_K^i(j)\, y_{ji}.$$

Como $(\bar{x},\bar{y})$ é uma solução factível arbitrária, então a desigualdade clique é válida para $PO(G,d,s)$. □

Sob determinadas condições, a desigualdade clique induz a uma faceta de $PO(G,d,s)$, como demonstrado a seguir.

Teorema 5.3. *A desigualdade clique (5.9) induz a uma faceta de $PO(G,d,s)$ se:*

(a) para todo $\ell \in N(i)\backslash K$ existe um $j \in K$ tal que $(j,\ell) \notin E$ e $d_{i,\ell} \leq d_{i,j}$,

(b) $d_{i,j} = \delta_K^i(j)$ para todo $j \in K$, e

(c) $s \geq \chi(G,d) + 3d_{\max}$.

Prova. Para provar que a desigualdade clique é indutora de faceta sob as condições estabelecidades, será mostrado que a face F correspondente é maximal. Suponha que $\lambda_x^T + \lambda_y^T = \lambda_0$ para todo $(x,y) \in PO(G,d,s)$ tal que (5.9) é satisfeita na igualdade. Será mostrado, a seguir, que o vetor $(\lambda_x,\lambda_y,\lambda_0)$ é múltiplo do vetor de coeficientes de (5.9), o que implica que a desigualdade induz a uma faceta de acordo com as condições enunciadas.

A prova é composta de várias partes, como visto a seguir.

1. $\lambda_{x_u} = 0$ para todo $u \in V$ tal que $u \neq i$. Considere duas soluções factíveis $(x,y)^1$ e $(x,y)^2$ tais que $x_i^1 = x_i^2 = 1$ (de modo que ambas as soluções pertencem a F), $x_u^1 = \chi(G,d) + d_{\max}$ e $x_u^2 = \chi(G,d) + d_{\max} + 1$. Tais soluções existem pela hipótese

(c). A única diferença entre elas é em x_u, logo, tem-se que:

$$\begin{array}{r} \displaystyle\sum_{p\in V\setminus\{u\}} \lambda_{x_p}x_p + \lambda_{x_u}x_u^1 + \sum_{(u,v)\in E} \lambda_{y_{uv}}y_{uv} = \lambda_0 \\ - \displaystyle\sum_{p\in V\setminus\{u\}} \lambda_{x_p}x_p + \lambda_{x_u}x_u^2 + \sum_{(u,v)\in E} \lambda_{y_{uv}}y_{uv} = \lambda_0 \\ \hline \lambda_{x_u}(x_u^1 - x_u^2) = 0 \end{array}$$

Como $x_u^1 \neq x_u^2$, pode-se concluir que $\lambda_{x_u} = 0$.

2. $\lambda_{y_{uv}} = 0$ para toda $(u,v) \in E$, $u,v \neq i$. Considere uma solução factível (x,y) de $PO(G,d,s)$, onde $x_i = 1$ (so $(x,y) \in F$) e $\max_{i\in V} x_i = \chi(G,d) + d_{\max}$. Tal solução existe pela hipótese (c). Agora, construa uma solução $(x,y)^1$ do seguinte modo:

- $x_p^1 = \begin{cases} \chi(G,d) + 2d_{\max} & \text{se } p = u; \\ \chi(G,d) + 3d_{\max} & \text{se } p = v; \\ x_p & \text{caso contrário.} \end{cases}$
- $y_{pq}^1 = \begin{cases} 1 & \text{se } q \in \{u,v\} \text{ ou } (p,q) = (u,v); \\ y_{uv} & \text{caso contrário.} \end{cases}$

Com base em $(x,y)^1$, construa outra solução $(x,y)^2$ trocando entre si as cores de u e v, ou seja, $x_u^2 = x_v^1$ e $x_v^2 = x_u^1$, o que também inverterá o valor de y_{uv}. Entre $(x,y)^1$ e $(x,y)^2$, as únicas diferenças são em x_u, x_v e y_{uv}. Seguindo esta observação e que, como mostrado anteriormente, $\lambda_{x_u} = 0$ para todo $u \in V$ com $u \neq i$, tem-se que:

$$\begin{array}{r} \displaystyle\sum_{p\in V\setminus\{u,v\}} \cancelto{0}{\lambda_{x_p}}x_p + \cancelto{0}{\lambda_{x_u}}x_u^1 + \cancelto{0}{\lambda_{x_v}}x_v^1 + \sum_{(p,q)\in E\setminus(u,v)} \lambda_{y_{pq}}y_{pq}^1 + \lambda_{y_{uv}}y_{uv}^1 = \lambda_0 \\ - \displaystyle\sum_{p\in V\setminus\{u,v\}} \cancelto{0}{\lambda_{x_p}}x_p + \cancelto{0}{\lambda_{x_u}}x_u^2 + \cancelto{0}{\lambda_{x_v}}x_v^2 + \sum_{(p,q)\in E\setminus(u,v)} \lambda_{y_{pq}}y_{pq}^2 + \lambda_{y_{uv}}y_{uv}^2 = \lambda_0 \\ \hline \lambda_{y_{uv}}(y_{uv}^1 - y_{uv}^2) = 0 \end{array}$$

Como $y_{uv}^1 = 1$ e $y_{uv}^2 = 0$, pode-se concluir que $\lambda_{y_{uv}} = 0$.

3. $\lambda_{y_{ij}} = \lambda_{x_i} d_{i,j}$, para todo $j \in K$. Considere uma solução (x,y) com $x_j = 1$, $x_i = d_{i,j} + 1$, e $x_u \geq x_i + d_{\max}$ para todo $u \in V\setminus\{i,j\}$ (tal solução existe pela hipótese (c)). Observe que (x,y) satisfaz (5.9) na igualdade, já que $\delta_K^i(j) = d_{i,j}$ pela hipótese. Tal solução tem $y_{ji} = 1$ e $y_{ui} = 0$ para todo $u \in K\setminus\{j\}$. Seja $(x,y)^1 = (x,y)$ e considere

outra solução $(x,y)^2$ onde $x_i^2 = 1$ e $x_j^2 = d_{i,j} + 1$, o que também implica que $y_{ji}^2 = 0$ e $y_{ij}^2 = 1$. As diferenças entre $(x,y)^1$ e $(x,y)^2$ ocorrem em x_i, x_j e y_{ij}, logo, levando-se em conta que $\lambda_{x_u} = 0$ para todo $u \in V$ tal que $u \neq i$ e que $\lambda_{y_{uv}} = 0$ para todo $(u,v) \in E$ tal que $u, v \neq i$, tem-se que:

$$\begin{array}{rl} & \sum_{u \in V \setminus \{i,j\}} \cancelto{0}{\lambda_{x_u} x_u} + \lambda_{x_i} x_i^1 + \cancelto{0}{\lambda_{x_j} x_j^1} + \sum_{(u,v) \in E \setminus \{(i,j)\}} \lambda_{y_{uv}} y_{uv}^1 + \lambda_{y_{ij}} y_{ij}^1 = \lambda_0 \\ - & \sum_{u \in V \setminus \{i,j\}} \cancelto{0}{\lambda_{x_u} x_u} + \lambda_{x_i} x_i^2 + \cancelto{0}{\lambda_{x_j} x_j^2} + \sum_{(u,v) \in E \setminus \{(i,j)\}} \lambda_{y_{uv}} y_{uv}^2 + \lambda_{y_{ij}} y_{ij}^2 = \lambda_0 \\ \hline & \lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{y_{ij}}(y_{ij}^1 - y_{ij}^2) = 0 \end{array}$$

Como dito anteriormente, $x_i^2 = x_i^1 - d_{i,j}$, $y_{ij}^1 = 0$ e $y_{ij}^2 = 1 - y_{ij}^1 = 1$, logo, pode-se concluir que $\lambda_{y_{ij}} = \lambda_{x_i} d_{i,j}$.

4. $\lambda_{y_{i\ell}} = 0$, para todo $\ell \in N(i) \setminus K$. A hipótese (a) implica que há pelo menos um vértice $j \in K$ tal que $(\ell, j) \notin E$ e $d_{i,\ell} \leq d_{i,j}$ (isso também implica que K é uma clique maximal em $N(i)$). Considere uma solução $(x,y)^1$, onde $x_i^1 = d_{i,j} + 1$ e $x_\ell^1 = x_j^1 = 1$ (que é factível, já que $d_{i,\ell} \leq d_{i,j}$), o que também faz com que $y_{ij}^1 = 0$, $y_{ji}^1 = 1$, $y_{i\ell}^1 = 0$ e $y_{\ell i}^1 = 1$. Agora, construa uma solução $(x,y)^2$, onde a cor de i é trocada com a cor de j e ℓ, ou seja, $x_i^2 = 1$ e $x_\ell^2 = x_j^2 = d_{i,j} + 1$, o que também implica que $y_{ij}^2 = 1$, $y_{ji}^2 = 0$, $y_{i\ell}^2 = 1$ e $y_{\ell i}^2 = 0$ Entre $(x,y)^1$ and $(x,y)^2$, as mudanças ocorrem em x_i, x_j, x_ℓ, y_{ij} e $y_{i\ell}$, logo, tem-se que:

$$\begin{array}{rl} & \sum_{u \in V \setminus \{i,j,\ell\}} \cancelto{0}{\lambda_{x_u} x_u} + \lambda_{x_i} x_i^1 + \cancelto{0}{\lambda_{x_j} x_j^1} + \cancelto{0}{\lambda_{x_\ell} x_\ell^1} + \\ & \sum_{(p,q) \in E \setminus \{(i,j),(i,\ell)\}} \lambda_{y_{pq}} y_{pq} + \lambda_{y_{ij}} y_{ij}^1 + \lambda_{y_{i\ell}} y_{i\ell}^1 = \lambda_0 \\ - & \sum_{u \in V \setminus \{i,j,\ell\}} \cancelto{0}{\lambda_{x_u} x_u} + \lambda_{x_i} x_i^2 + \cancelto{0}{\lambda_{x_j} x_j^2} + \cancelto{0}{\lambda_{x_\ell} x_\ell^2} + \\ & \sum_{(p,q) \in E \setminus \{(i,j),(i,\ell)\}} \lambda_{y_{pq}} y_{pq} + \lambda_{y_{ij}} y_{ij}^2 + \lambda_{y_{i\ell}} y_{i\ell}^2 = \lambda_0 \\ \hline & \lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{y_{ij}}(y_{ij}^1 - y_{ij}^2) + \lambda_{y_{i\ell}}(y_{i\ell}^1 - y_{i\ell}^2) = 0 \end{array}$$

Como $x_i^2 = x_i^1 - d_{i,j}$, $y_{ij}^1 = 0$, $y_{i\ell}^1 = 0$, $y_{ij}^2 = 1$, $y_{i\ell}^2 = 1$ e, como mostrado anterior-

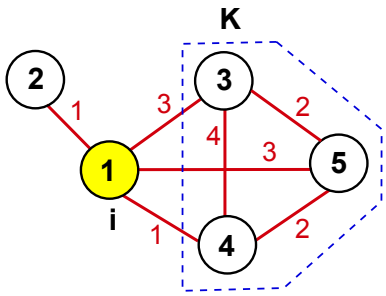


Figura 5.2: Exemplo de desigualdade clique para um vértice i e uma clique $K \subseteq N(i)$ de um grafo.

mente, $\lambda_{y_{ij}} = \lambda_{x_i} d_{i,j}$, tem-se que:

$$\lambda_{x_i}(x_i^1 - (x_i^1 - d_{i,j})) - \lambda_{x_i} d_{i,j} - \lambda_{y_{i\ell}} = 0 \;\Rightarrow\; \lambda_{x_i} d_{i,j} - \lambda_{x_i} d_{i,j} - \lambda_{y_{i\ell}} = 0$$

Pode-se concluir, então, que $\lambda_{y_{i\ell}} = 0$.

Para finalizar a prova, observe que a desigualdade clique pode ser escrita como:

$$x_i \geq \sum_{j\in K} \delta_K^i(j)\, y_{ji} + 1 \;\Rightarrow\; x_i - \sum_{j\in K} d_{i,j}\delta_K^i(j)\, y_{ji} \geq 1 \;\Rightarrow\; x_i - \sum_{j\in K} \delta_K^i(j)\,(1 - y_{ij}) \geq 1$$

Como $\delta_K^i(j) = d_{i,j}$ para todo $j \in K$, tem-se que:

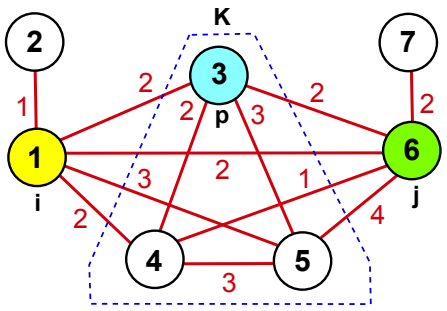
$$x_i - \sum_{j\in K} \delta_K^i(j)\,(1 - y_{ij}) \geq 1 \;\Rightarrow\; x_i + \sum_{j\in K} d_{i,j}\, y_{ij} \geq 1 + \sum_{j\in K} d_{i,j}$$

Desta observação e das afirmações anteriores, tem-se que:

$$\lambda_x x + \lambda_y y = \sum_{u\in V\setminus\{i\}} \cancelto{0}{\lambda_{x_u}} x_u + \lambda_{x_i} x_i + \sum_{(i,u)\in E: u\notin K} \cancelto{0}{\lambda_{y_{iu}}} y_{iu} + \sum_{j\in K} \lambda_{y_{ij}} y_{ij} = \lambda_0$$

$$\Rightarrow\; \lambda_{x_i} x_i + \sum_{j\in K} \lambda_{x_i} d_{i,j} y_{ij} = \lambda_0 \;\Rightarrow\; \lambda_{x_i}\left(x_i + \sum_{j\in K} d_{i,j} y_{ij}\right) = \lambda_0$$

Como $\lambda_x^T + \lambda_y^T = \lambda_0$ é não trivial, considerando-se $\lambda_{x_i} \neq 0$ e $\frac{\lambda_0}{\lambda_{x_i}} = 1 + \sum_{j\in K} d_{i,j}$ tem-se que $\lambda_x^T + \lambda_y^T = \lambda_0$ é λ_{x_i} vezes a desigualdade clique, o que implica que a mesma induz a uma faceta de $PO(G,d,s)$ sob as condições especificadas. □



$$x_1 + 2 + 2(y_{13} - y_{63}) + 0(y_{14} - y_{64}) + y_{15} - y_{65} \leq x_6 + (s-1)y_{61}$$

Figura 5.3: Exemplo de desigualdade clique dupla para uma aresta $(i,j) \in E$, uma clique $K \subseteq N(i) \cap N(j)$ e um vértice $p \in K$ de um grafo.

5.1.1.2 Desigualdade clique dupla

O segundo conjunto de desigualdades válidas é obtido por meio de uma clique que esteja presente na vizinhança de dois vértices ao mesmo tempo, como definido a seguir.

Definição 5.1.2. *Seja $(i,j) \in E$ uma aresta de um grafo G e considere uma clique $K \subseteq N(i) \cap N(j)$. Para todo $k \in K$, seja $\delta_K^{ij}(k) = \min_{\ell \in K \cup \{i,j\} \setminus \{k\}} d_{k,\ell}$. Fixando-se um vértice $p \in K$, pode-se definir a expressão:*

$$x_i + d(i,j) + \sum_{k \in K} \gamma_k (y_{ik} - y_{jk}) \leq x_j + (s + d(i,j) - \gamma(K) - 1) y_{ji} \tag{5.10}$$

como a desigualdade clique dupla (double clique) associada à aresta (i,j), à clique K e ao vértice p, na qual $\gamma_p = \max\{0, 2\delta_K^{ij}(p) - d(i,j)\}$ e $\gamma_k = \max\{0, \delta_K^{ij}(k) - d_{i,j}\}$ para todo $k \in K \setminus \{p\}$.

A Figura 5.3 mostra um exemplo de grafo onde é selecionada uma aresta $(i,j) \in E$, uma clique $K \subseteq N(i) \cap N(j)$ e um vértice $p \in K$ no mesmo e é dada a desigualdade clique dupla correspondente. A expressão fortalece as restrições de distância do modelo, eliminando possibilidades de cores para os vértices que satisfariam as restrições originais, mas não levariam diretamente a uma solução inteira ao resolver-se a relaxação linear. Do mesmo modo que ocorre com a desigualdade clique, algumas condições fazem com que a desigualdade clique dupla defina facetas de $PO(G,d,s)$, como visto a seguir.

Teorema 5.4. *A desigualdade clique dupla (5.10) é válida para $PO(G,d,s)$.*

Prova. Considere uma solução arbitrária factível $(x,y) \in PO(G,d,s)$. Serão examinados dois casos:

1. $y_{ji} = 0$ (e $y_{ij} = 1$). Isso implica que $x_i < x_j$. Seja $M \subseteq K$ o conjunto de vértices cujas cores estejam entre x_i e x_j, ou seja, $M = \{k \in K : y_{ik} = 1 \text{ and } y_{jk} = 0\}$. Seja $M^{\prec}$ uma ordenação de $M \cup \{i, j\}$ de acordo com as cores dos vértices (variáveis x), isto é, $M^{\preceq} = (k_1, k_2, \ldots, k_t)$, onde $k_1 = i$, $k_t = j$ e $x_{k_\ell} < x_{k_\ell+1}$ para todo $\ell = 1, 2, \ldots, t-1$. Pela definição do BCP e da ordenação $M^{\prec}$, tem-se que $x_{k_\ell+1} - x_{k_\ell} \geq d_{k_\ell,k_\ell+1} \geq \delta_K^{ij}(k_\ell + 1)$. Se $p \notin M$ então:

$$\begin{aligned} x_j - x_i &= \sum_{\ell=1}^{t-1} (x_{k_\ell+1} - x_{k_\ell}) \\ &\geq \max\{d_{i,j}, \sum_{\ell=1}^{t-1} \delta_K^{ij}(k_\ell)\} \\ &\geq d_{i,j} + \sum_{k \in M} \gamma_k \\ &= d_{i,j} + \sum_{k \in K} \gamma_k (y_{ik} - y_{jk}). \end{aligned}$$

   Logo, a desigualdade é satisfeita. Por outro lado, se $p \in M$, então seja $r \geq 2$ o índice de p em $M^{\prec}$, ou seja, tal que $k_r = p$. Tem-se então:

$$\begin{aligned} x_j - x_i &= \sum_{\ell=1}^{t-1} (x_{k_\ell+1} - x_{k_\ell}) \\ &= \sum_{\ell=1}^{r-2} (x_{k_\ell+1} - x_{k_\ell}) + (x_{k_r+1} - x_{k_r-1}) + \sum_{\ell=r+1}^{t-1} (x_{k_\ell+1} - x_{k_\ell}) \\ &\geq \sum_{\ell=1}^{r-2} \delta_K^{ij}(k_{\ell+1}) + 2\delta_K^{ij}(p) + \sum_{\ell=r+1}^{t-1} \delta_K^{ij}(k_\ell) \\ &\geq d_{i,j} + \sum_{k \in M} \gamma_k \\ &= d_{i,j} + \sum_{k \in K} \gamma_k (y_{ik} - y_{jk}). \end{aligned}$$

   Novamente, a desigualdade é satisfeita.

2. $y_{ji} = 1$ (e $y_{ij} = 0$). Isso implica que $x_j < x_i$. Particione K nos seguintes subconjuntos disjuntos:

   - $L = \{k \in K : y_{ik} = y_{jk} = 0\}$.
   - $M = \{k \in K : y_{ik} = 0 \text{ e } y_{jk} = 1\}$.
   - $R = \{k \in K : y_{ik} = y_{jk} = 1\}$.

   Sejam $L^{\prec}$, $M^{\prec}$ e $R^{\prec}$, respectivamente, ordenações de $L \cup \{j\}$, $M \cup \{i, j\}$ e $R \cup \{i\}$

de acordo com as cores dos vértices (variáveis x), isto é:

- $L^{\prec} = (\ell_1, \ell_2, \ldots, \ell_t)$, onde $\ell_t = j$ e $x_{\ell_a} < x_{\ell_a+1}$ para todo $a = 1,2,\ldots,t-1$.
- $M^{\prec} = (m_1, m_2, \ldots, m_h)$, onde $m_1 = j$, $m_h = i$ e $x_{m_a} < x_{m_a+1}$ para todo $a = 1,2,\ldots,h-1$.
- $R^{\prec} = (r_1, r_2, \ldots, r_w)$, onde $r_1 = i$ e $x_{r_a} < x_{r_a+1}$ para todo $a = 1,2,\ldots,h-1$.

Observe que $x_j \geq 1 + \sum_{a=1}^{t-1} (x_{\ell_{a+1}} - x_{\ell_a})$. Também note que $x_i \leq s - \sum_{a=1}^{w-1} (x_{r_{a+1}} - x_{r_a})$. Por fim, observe que:

$$\begin{aligned}\sum_{k\in K} \gamma_k (y_{ik} - y_{jk}) &= \sum_{\ell\in L} \gamma_\ell \underbrace{(y_{i\ell} - y_{j\ell})}_{\to 0} + \sum_{m\in M} \gamma_m (y_{im} - y_{jm}) + \sum_{r\in R} \gamma_r \underbrace{(y_{ir} - y_{jr})}_{\to 0} \\ &= -\sum_{m\in M} \gamma_m = -\gamma(M).\end{aligned}$$

Então, tem-se que:

$$\begin{aligned} & x_i + d_{i,j} - x_j + \sum_{k\in K} \gamma_k (y_{ik} - y_{jk}) \\ \leq\ & s - \sum_{a=1}^{w-1} (x_{r_{a+1}} - x_{r_a}) + (d_{i,j} - 1) - \sum_{a=1}^{t-1} (x_{\ell_{a+1}} - x_{\ell_a}) - \gamma(M) \\ \leq\ & s - \sum_{a=1}^{w-1} \delta_K^{ij}(r_a) + (d_{i,j} - 1) - \sum_{a=1}^{t-1} \delta_K^{ij}(\ell_a) - \gamma(M) \\ \leq\ & s - \gamma(R) - \gamma(L) - \gamma(M) + d_{i,j} - 1 \\ =\ & s - \gamma(K) + d_{i,j} - 1. \end{aligned}$$

Logo, a desigualdade é satisfeita. □

Teorema 5.5. *A desigualdade clique dupla (5.10) induz a uma faceta de $PO(G,d,s)$ se:*

a) $s \geq \chi(G,d) + 4d_{max}$,

b) $d(i,k) = d(j,k) = \delta_K^{ij}(k)$ *para todo* $k \in K$,

c) $d(p,k) = d(p,j)$ *para todo* $k \in K\backslash\{p\}$,

d) $d(i,j) \leq \delta_K^{ij}(k)$ *para todo* $k \in K\backslash\{p\}$, *e*

e) $(t,p) \notin E$ *e* $d(i,t) + d(t,j) \leq d(i,j)$ *para todo* $t \in [N(i) \cap N(j)]\backslash K$

Prova. Para provar que a desigualdade clique dupla é indutora de faceta sob as condições dadas, será mostrado que a face F correspondente é maximal. Suponha que $\lambda_x^T + \lambda_y^T = \lambda_0$ para todo $(x,y) \in F$. Para isso, será demonstrado que o vetor $(\lambda_x, \lambda_y, \lambda_0)$ é múltiplo do vetor de coeficientes de (5.10).

A prova é composta de diversas partes, como visto a seguir:

1. $\lambda_{x_u} = 0$ para $u \neq i, j$. Considere uma solução $(x,y)^1 \in F$, onde $x_i^1 = 1$, $x_j^1 = d_{i,j} + 1$, $x_k^1 \in [2d_{\max}, \chi(G,d) + 2d_{\max}]$ para $k \neq i, j, u$, e $x_u = \chi(G,d) + 3d_{\max}$ (tal solução existe pela hipótese (a)). Agora, construa uma solução $(x,y)^2 \in F$, na qual $x_u^2 = x_u^1 + 1$, que também é factível e satisfaz and satisfies (5.10) na igualdade. A única diferença entre $(x,y)^1$ e $(x,y)^2$ é x_u, então tem-se que:

$$\begin{array}{rl} & \sum_{t \in V \setminus \{u\}} \lambda_{x_t} x_t + \lambda_{x_u} x_u^1 + \sum_{(t,q) \in E} \lambda_{y_{tq}} y_{tq} = \lambda_0 \\ - & \sum_{t \in V \setminus \{u\}} \lambda_{x_t} x_t + \lambda_{x_u} x_u^2 + \sum_{(t,q) \in E} \lambda_{y_{tq}} y_{tq} = \lambda_0 \\ \hline & \lambda_{x_u}(x_u^1 - x_u^2) = 0 \end{array}$$

   Como $x_u^1 \neq x_u^2$, pode-se concluir que $\lambda_{x_u} = 0$.

2. $\lambda_{y_{uv}} = 0$ se ambos u, v diferem de i, j. Seja $(x,y)^1 \in F$ uma solução factível satisfazendo (5.10) na igualdade, tal que $x_u^1 = 1$, $x_v^1 = 1 + d_{u,v}$ (o que implica que $y_{uv}^1 = 1$), e $x_t^1 \geq x_v^1 + d_{\max}$ para todo $t \in V \setminus \{u, v\}$ (a existência de tal solução é assegurada pela hipótese (a)). Construa uma nova solução $(x,y)^2$ trocando entre si as cores atribuídas a u e v, ou seja, $x_u^2 = x_v^1$ e $x_v^2 = x_u^1$ (o que implica que $y_{uv}^2 = 1$). A nova solução também pertence a F e as diferenças entre ela e a solução original ocorrem em x_u, x_v e y_{uv}, de modo que tem-se que:

$$\begin{array}{rl} & \sum_{t \in V \setminus \{u,v\}} \cancelto{0}{\lambda_{x_t} x_t} + \cancelto{0}{\lambda_{x_u} x_u^1} + \cancelto{0}{\lambda_{x_v} x_v^1} + \sum_{(t,q) \in E \setminus \{(u,v)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{uv}} y_{uv}^1 = \lambda_0 \\ - & \sum_{t \in V \setminus \{u,v\}} \cancelto{0}{\lambda_{x_t} x_t} + \cancelto{0}{\lambda_{x_u} x_u^2} + \cancelto{0}{\lambda_{x_v} x_v^2} + \sum_{(t,q) \in E \setminus \{(u,v)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{uv}} y_{uv}^2 = \lambda_0 \\ \hline & \lambda_{y_{uv}}(y_{uv}^1 - y_{uv}^2) = 0 \end{array}$$

   Como $y_{uv}^1 = 1$ e $y_{uv}^2 = 0$, pode-se concluir que $\lambda_{y_{uv}} = 0$.

3. $\lambda_{x_i} = -\lambda_{x_j}$. Considere uma solução factível $(x,y)^1 \in F$ com $x_i^1 = 1$, $x_j^1 = 1 + d_{i,j}$

e tal que $x_t^1 \in [2d_{\max}+1, s-1]$ para $t \neq i, j$. Construa uma outra solução $(x,y)^2$ onde $x_t^2 = x_t^1 + 1$ para todo $t \in V$, logo $(x,y)^2 \in F$. Tais soluções diferem entre si nas variáveis x, então, seguindo o resultado anterior, tem-se que:

$$\begin{array}{rc} & \sum_{t \in V \setminus \{i,j\}} \lambda_{x_t} \overbrace{x_t}^{0} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \sum_{(t,q) \in E} \lambda_{y_{tq}} y_{tq} = \lambda_0 \\ - & \sum_{t \in V \setminus \{i,j\}} \lambda_{x_p} \overbrace{x_p}^{0} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \sum_{(t,q) \in E} \lambda_{y_{tq}} y_{tq} = \lambda_0 \\ \hline & \lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{x_j}(x_j^1 - x_j^2) = 0 \end{array}$$

Como $x_i^2 = x_i^1 + 1$ e $x_j^2 = x_j^1 + 1$, pode-se concluir que $\lambda_{x_i} = -\lambda_{x_j}$.

4. $\lambda_{y_{ik}} = -\gamma_k \lambda_{x_j}$ para todo $k \in K$. Seja $(x,y)^1$ uma solução em F definida como se segue:

   - Se $k = p$, então $x_p^1 = 1$, $x_i^1 = d_{\max}$, $x_j^1 = x_i^1 + d_{i,j}$, e $x_t^1 \geq x_j^1 + d_{\max}$ para todo $t \in V \setminus \{i, j, p\}$.
   - Se $k \neq p$, então $x_k^1 = 1$, $x_i^1 = d_{\max}$, $x_p^1 = x_i^1 + d_{i,p}$, $x_j^1 = x_p^1 + d_{p,j}$, e $x_t^1 \geq x_j^1 + d_{\max}$ para todo $t \in V \setminus \{i, j, k, p\}$.

   A hipótese (a) garante que haverá cores suficientes para tais construções, enquanto as hipóteses (b)-(d) garantem que a solução $(x,y)^1$ é factível e satisfaz 5.10 na igualdade. Construa uma solução $(x,y)^2$ com base em $(x,y)^1$, trocando a ordem dos vértices k e i entre si, como definido a seguir:

   - Se $k = p$, então $x_i^2 = 1$, $x_p^2 = 1 + d_{i,p}$, $x_j^2 = x_p^2 + d_{p,j}$, e mantenha as variáveis x restantes sem modificações.
   - Se $k \neq p$, então $x_1^2 = 1$, $x_k^2 = 1 + d_{i,k}$, $x_p^2 = x_k^2 + d_{k,p}$, $x_j^2 = x_p^1 + d_{p,j}$, e mantenha as variáveis x restantes sem modificações.

   Entre as duas soluções, as mudanças ocorrem nas variáveis x_t para $t \in \{i, j, k, p\}$ e

y_{ik}, então tem-se que:

$$\begin{array}{r} \sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \sum_{(t,q)\in E\setminus\{(i,k)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{ik}} y_{ik}^1 = \lambda_0 \\ - \sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \sum_{(t,q)\in E\setminus\{(i,k)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{ik}} y_{ik}^2 = \lambda_0 \\ \hline \lambda_{x_j}(x_j^1 - x_j^2) + \lambda_{y_{ik}}(y_{ik}^1 - y_{ik}^2) = 0 \end{array}$$

Os valores das variáveis que diferem em ambas as soluções permitem concluir que $\lambda_{y_{ik}} = (d_{i,p} + d_{p,j} - d_{i,j})\lambda_{x_j} = \gamma_k \lambda_{x_j}$.

5. $\lambda_{y_{jk}} = \gamma_k \lambda_{x_j}$ para todo $k \in K$. Uma argumentação similar a do item anterior pode ser usada também neste caso. Construa uma solução $(x,y)^1$ in F como a seguir:

   - Se $k = p$, então $x_i^1 = 1$, $x_j^1 = 1 + d_{i,j}$, $x_p^1 = x_j^1 + d_{j,p}$, e $x_t^1 \geq x_p^1 + d_{\max}$ para todo $t \in V\setminus\{i,j,p\}$.
   - Se $k \neq p$, então $x_i^1 = 1$, $x_p^1 = 1 + d_{i,p}$, $x_j^1 = x_p^1 + d_{p,j}$, $x_k^1 = x_j^1 + d_{j,k}$, para todo $x_t^1 \geq x_k^1 + d_{\max}$ for all $t \in V\setminus\{i,j,k,p\}$.

   Novamente, a hipótese (a) garante que haverá cores suficientes para tais construções, enquanto as hipóteses (b)-(d) garantem que a solução $(x,y)^1$ é factível e satisfaz 5.10 na igualdade. Construa agora uma solução $(x,y)^2$ com base em $(x,y)^1$, como definido a seguir:

   - Se $k = p$, então $x_i^1 = 1$, $x_p^1 = 1 + d_{i,p}$, $x_j^1 = x_p^1 + d_{p,j}$, e mantenha as variáveis x restantes sem modificações.
   - Se $k \neq p$, então $x_i^1 = 1$, $x_p^1 = 1 + d_{i,p}$, $x_k^1 = x_p^1 + d_{p,k}$, $x_j^1 = x_k^1 + d_{j,k}$, e mantenha as variáveis x restantes sem modificações.

   Entre as duas soluções, as mudanças ocorrem nas variáveis x_t para $t \in \{i,j,k,p\}$ e y_{jk}, então tem-se que:

$$\begin{array}{r} \sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \sum_{(t,q)\in E\setminus\{(j,k)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{jk}} y_{jk}^1 = \lambda_0 \\ - \sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \sum_{(t,q)\in E\setminus\{(i,k)\}} \lambda_{y_{tq}} y_{tq} + \lambda_{y_{jk}} y_{jk}^2 = \lambda_0 \\ \hline \lambda_{x_j}(x_j^1 - x_j^2) + \lambda_{y_{jk}}(y_{jk}^1 - y_{jk}^2) = 0 \end{array}$$

Mais uma vez, os valores das variáveis que diferem em ambas as soluções permitem concluir que $\lambda_{y_{jk}} = \gamma_k \lambda_{x_j}$.

6. $\lambda_{y_{it}} = 0$ para todo $t \in [N(i) \cap N(j)] \backslash K$. Considere uma solução $(x,y)^1$ tal que $x_j^1 = x_i^1 + d_{i,j}$ e $x_t^1 < x_i^1$ (que implica em $y_{it}^1 = 0$ e $y_{jt}^1 = 0$). Agora, construa uma solução $(x,y)^2$ onde $x_i^2 = x_i^1$, $x_j^2 = x_j^1$ e $x_t^2 = x_i^2 + d_{i,t}$ (o que implica que $y_{it}^2 = 1$), que é factível de acordo com a hipótese (e). As diferenças entre as soluções ocorrem em x_t e y_{it}, então tem-se que:

$$\begin{array}{rl} & \sum_{r \in V \backslash \{t\}} \lambda_{x_r} x_r + \cancelto{0}{\lambda_{x_t} x_t^1} + \sum_{(r,q) \in E \backslash \{(i,t)\}} \lambda_{y_{rq}} y_{rq}^1 + \lambda_{y_{it}} y_{it}^1 = \lambda_0 \\ - & \sum_{r \in V \backslash \{t\}} \lambda_{x_r} x_r + \cancelto{0}{\lambda_{x_t} x_t^2} + \sum_{(r,q) \in (E \backslash (i,t))} \lambda_{y_{rq}} y_{rq}^2 + \lambda_{y_{it}} y_{it}^2 = \lambda_0 \\ \hline & \lambda_{y_{it}} (y_{it}^1 - y_{it}^2) = 0 \end{array}$$

Como $y_{it}^2 \neq y_{it}^1$, pode-se concluir que $\lambda_{y_{it}} = 0$.

7. $\lambda_{y_{jt}} = 0$ para todo $t \in [N(i) \cap N(j)] \backslash K$. Considere uma solução $(x,y)^1$ tal que $x_j^1 = x_i^1 + d_{i,j}$ e $x_t^1 > x_j^1$ (o que implica em $y_{jt}^1 = 1$ e $y_{it}^1 = 1$). Agora, construa uma solução $(x,y)^2$ na qual $x_i^2 = x_i^1$, $x_j^2 = x_j^1$ e $x_t^2 = x_j^2 - d_{j,t}$ (o que implica em $y_{jt}^2 = 0$) que também é factível pois $d_{i,t} + d_{t,j} \leq d_{i,j}$. As diferenças entre $(x,y)^1$ e $(x,y)^2$ ocorrem em x_t e y_{jt}, então tem-se que:

$$\begin{array}{rl} & \sum_{r \in V \backslash \{t\}} \lambda_{x_r} x_r + \cancelto{0}{\lambda_{x_t} x_t^1} + \sum_{(r,q) \in E \backslash \{(j,t)\}} \lambda_{y_{rq}} y_{rq}^1 + \lambda_{y_{jt}} y_{jt}^1 = \lambda_0 \\ - & \sum_{r \in V \backslash \{t\}} \lambda_{x_r} x_r + \cancelto{0}{\lambda_{x_t} x_t^2} + \sum_{(r,q) \in E \backslash \{(j,t)\}} \lambda_{y_{rq}} y_{rq}^2 + \lambda_{y_{jt}} y_{jt}^2 = \lambda_0 \\ \hline & \lambda_{y_{jt}} (y_{jt}^1 - y_{jt}^2) = 0 \end{array}$$

Como $y_{jt}^2 \neq y_{jt}^1$, pode-se concluir que $\lambda_{y_{zt}} = 0$.

8. $\lambda_{y_{ik}} = 0$ para todo $k \in N(i) \backslash N(j)$. Seja $(x,y)^1 \in F$ uma solução onde $x_i^1 = 1$, $x_j^1 = d_{i,j} + 1$, $x_k^1 = x_j^1 + d_{i,k}$ (o que implica em $y_{ik}^1 = 1$) e $x_t^1 > d_{i,j} + d_{i,k} + 1$ para todo $t \notin \{i,j,k\}$. Esta construção é factível porque $(k,j) \notin E$ e pela hipótese (a). Construa uma outra solução $(x,y)^2 \in F$ na qual $x_k^2 = 1$, $x_i^2 = 1 + d_{i,k}$ e $x_j^2 = x_i^2 + d_{i,j}$ (o que implica que $y_{ik}^2 = 0$). As mudanças entre as duas soluções são em x_i, x_j, x_k e y_{ik},

então tem-se que:

$$\begin{array}{r} \sum_{t\in V\setminus\{i,j,k\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \cancelto{0}{\lambda_{x_k} x_k^1} + \sum_{(t,q)\in E\setminus\{(i,k)\}} \lambda_{y_{tq}} y_{tq}^1 + \lambda_{y_{ik}} y_{ik}^1 = \lambda_0 \\ - \sum_{t\in V\setminus\{i,j,k\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \cancelto{0}{\lambda_{x_k} x_k^2} + \sum_{(t,q)\in E\setminus\{(i,k)\}} \lambda_{y_{tq}} y_{tq}^2 + \lambda_{y_{ik}} y_{ik}^2 = \lambda_0 \\ \hline \lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{x_j}(x_j^1 - x_j^2) + \lambda_{y_{ik}}(y_{ik}^1 - y_{ik}^2) = 0 \end{array}$$

Assim, tem-se que $\lambda_{y_{ik}} = 0$.

9. $\lambda_{y_{jk}} = 0$ para todo $k \in N(i)\backslash N(j)$. Seja $(x,y)^1 \in F$ uma solução onde $x_i^1 = 1$, $x_j^1 = d_{i,j} + 1$, $x_k^1 = x_j^1 + d_{j,k}$ (o que implica que $y_{jk}^1 = 1$) e $x_t^1 > d_{i,j} + d_{j,k} + 1$ para todo $t \notin \{i,j,k\}$. Tal solução é factível pela hipótese (a). Agora, construa outra solução $(x,y)^2 \in F$ atribuindo $x_k^2 = 1$, $x_i^2 = 1 + d_{j,k}$ e $x_j^2 = x_i^2 + d_{i,j}$ (o que implica em $y_{jk}^2 = 0$). Esta solução também é factível pois $(k,i) \notin E$. As mudanças entre as duas soluções são em x_i, x_j, x_k e y_{jk}, então tem-se que:

$$\begin{array}{r} \sum_{t\in V\setminus\{i,j,k\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \cancelto{0}{\lambda_{x_k} x_k^1} + \sum_{(t,q)\in E\setminus\{(j,k)\}} \lambda_{y_{tq}} y_{tq}^1 + \lambda_{y_{jk}} y_{jk}^1 = \lambda_0 \\ - \sum_{t\in V\setminus\{i,j,k\}} \cancelto{0}{\lambda_{x_p} x_p} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \cancelto{0}{\lambda_{x_k} x_k^2} + \sum_{(t,q)\in E\setminus\{(j,k)\}} \lambda_{y_{tq}} y_{tq}^2 + \lambda_{y_{jk}} y_{jk}^2 = \lambda_0 \\ \hline \lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{x_j}(x_j^1 - x_j^2) + \lambda_{y_{jk}}(y_{jk}^1 - y_{jk}^2) = 0 \end{array}$$

Assim, conclui-se que $\lambda_{y_{jk}} = 0$.

10. $\lambda_{y_{ij}} = -(s + d_{i,j} - \gamma(K) - 1)\lambda_{x_j}$. Seja $(x,y)^1 \in F$ uma solução onde $x_i^1 = 1$, $x_j^1 = 1 + d_{i,j}$, e $x_t^1 \geq x_j^1 + d_{\max}$ para todo $t \in V\backslash\{i,j\}$ (o que implica que $y_{ij}^1 = 1$ e $y_{it}^1 = y_{jt}^1 = 1$ para $t \neq i,j$). Com isso, tem-se que $y_{ik}^1 - y_{jk}^1 = 0$. Agora, construa outra solução $(x,y)^2$ com $x_i^2 = s$ e $x_j^2 = 1$ (o que implica em $y_{ij}^2 = 0$, $y_{it}^2 = 0$ e $y_{jt}^2 = 1$ para $t \neq i,j$).

Pode-se derivar então o seguinte:

$$\sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^1 + \lambda_{x_j} x_j^1 + \sum_{t\in N(i)\setminus(K\cup\{j\})} \cancelto{0}{\lambda_{y_{it}} y_{it}^1} + \sum_{q\in N(j)\setminus(K\cup\{i\})} \cancelto{0}{\lambda_{y_{jq}} y_{jq}^1} +$$
$$\sum_{k\in K} (\lambda_{y_{ik}} y_{ik}^1 + \lambda_{y_{jk}} y_{jk}^1) + \lambda_{y_{ij}} y_{ij}^1 = \lambda_0$$
$$- \sum_{t\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_t} x_t} + \lambda_{x_i} x_i^2 + \lambda_{x_j} x_j^2 + \sum_{t\in N(i)\setminus(K\cup\{j\})} \cancelto{0}{\lambda_{y_{it}} y_{it}^2} + \sum_{q\in N(j)\setminus(K\cup\{i\})} \cancelto{0}{\lambda_{y_{jq}} y_{jq}^2} +$$
$$\sum_{k\in K} (\lambda_{y_{ik}} y_{ik}^2 + \lambda_{y_{jk}} y_{jk}^2) + \lambda_{y_{ij}} y_{ij}^2 = \lambda_0$$

$$\lambda_{x_i}(x_i^1 - x_i^2) + \lambda_{x_j}(x_j^1 - x_j^2) + \sum_{k\in K} (\lambda_{y_{ik}} y_{ik}^1 + \lambda_{y_{jk}} y_{jk}^1) - \sum_{k\in K} (\lambda_{y_{ik}} y_{ik}^2 + \lambda_{y_{jk}} y_{jk}^2) +$$
$$\lambda_{y_{ij}}(y_{ij}^1 - y_{ij}^2) = 0$$

O que implica em:

$$\lambda_{x_i}(x_i^1 - x_i^2 - x_j^1 + x_j^2) + \sum_{k\in K} (d_{j,k}\lambda_{x_j} - d_{j,k}\lambda_{x_j}) - \sum_{k\in K} d_{j,k}\lambda_{x_j} + \lambda_{y_{ij}} = 0 \quad \Rightarrow$$

$$-\lambda_{x_j}(x_i^1 - s - x_i^1 - d_{i,j} + 1) - \sum_{k\in K} d_{j,k}\lambda_{x_j} + \lambda_{y_{ij}} = 0 \;\Rightarrow\; \lambda_{y_{ij}} = \lambda_{x_j}(-s - d_{i,j} + 1) + \sum_{k\in K} d_{j,k}\lambda_{x_j}$$

Tem-se então que $\lambda_{y_{ij}} = \lambda_{x_j}(-s - d_{i,j} + 1 + \sum_{k\in K} d_{j,k})$. Como $d_{i,k} = d_{j,k} = \delta_K^{ij}(k)$ para todo $k \in K$, conclui-se que $\lambda_{y_{ij}} = -(s + d_{i,j} - \gamma(K) - 1)\lambda_{x_j}$

Para finalizar a prova, note que a desigualdade clique dupla pode ser escrita como:

$$x_i + d_{i,j} + \sum_{k\in K} \gamma_k(y_{ik} - y_{jk}) \;\leq\; x_j + (s + d_{i,j} - \gamma(K) - 1)y_{ji} \quad \Rightarrow$$

$$x_i + d_{i,j} + \sum_{k\in K} \gamma_k(y_{ik} - y_{jk}) \;\leq\; x_j + (s + d_{i,j} - \gamma(K) - 1)(1 - y_{ij}) \quad \Rightarrow$$
$$x_i - x_j + \sum_{k\in K} \gamma_k(y_{ik} - y_{jk}) + (s + d_{i,j} - \gamma(K) - 1)y_{ij} \leq s - \gamma(K) - 1$$

Seguindo as afirmações anteriores, $\lambda_x x + \lambda_y y = \lambda_0$ pode ser escrito como:

$$\lambda_x x + \lambda_y y = \lambda_0 \quad \Rightarrow$$

$$\lambda_{x_i} x_i + \lambda_{x_j} x_j + \sum_{p\in V\setminus\{i,j\}} \cancelto{0}{\lambda_{x_p} x_p} + \sum_{\substack{(p,q)\in E \\ (p,q)\neq(i,j)}} \cancelto{0}{\lambda_{y_{pq}} y_{pq}} + \sum_{k\in K} \lambda_{y_{ik}} y_{ik} + \sum_{k\in K} \lambda_{y_{jk}} y_{jk} + \sum_{t\in[N(i)\cap N(j)]\setminus K} \cancelto{0}{\lambda_{y_{it}} y_{it}} +$$

$$\sum_{t\in[N(i)\cap N(j)]\setminus K} \cancelto{0}{\lambda y_{jt}} y_{jt} + \sum_{u\in N(i)\setminus N(j)} \cancelto{0}{\lambda y_{iu}} y_{iu} + \sum_{v\in N(j)\setminus N(j)} \cancelto{0}{\lambda y_{jv}} y_{jv} + \lambda y_{ij} y_{ij} = \lambda_0 \quad \Rightarrow$$

$$-\lambda_{x_j} x_i + \lambda_{x_j} x_i - \sum_{k\in K} d_{j,k} y_{ik} + \sum_{k\in K} d_{j,k} y_{jk} - (s + d_{i,j} - \gamma(K) - 1)\lambda_{x_j} y_{ij} = \lambda_0 \quad \Rightarrow$$

$$-\lambda_{x_j} \left(x_i - x_j + \sum_{k\in K} d_{j,k}(y_{ik} - y_{jk}) + (s + d_{i,j} - \gamma(K) - 1) y_{ij} \right) = \lambda_0$$

Como $d_{i,k} = d_{j,k} = \delta_K^{ij}(k)$ para todo $k \in K$, $d_{p,k} = d_{p,j} = \delta_K^{ij}(k)$ para todo $k \in K$ e $d_{i,j} \leq \delta_K^{ij}(k)$ para todo $k \in K\setminus\{p\}$:

$$-\lambda_{x_j} \left(x_i - x_j + \sum_{k\in K} \delta_K^{ij}(k)(y_{ik} - y_{jk}) + (s + d_{i,j} - \gamma(K) - 1) y_{ij} \right) = \lambda_0 \quad \Rightarrow$$

$$-\lambda_{x_j} \left(x_i - x_j + \sum_{k\in K} \gamma_k(y_{ik} - y_{jk}) + (s + d_{i,j} - \gamma(K) - 1) y_{ij} \right) = \lambda_0$$

Como $\lambda_x^T + \lambda_y^T = \lambda_0$ é não trivial, considerando-se $\lambda_{x_j} \neq 0$ e $\frac{\lambda_0}{\lambda_{x_j}} = -s + \gamma(K) + 1$ tem-se que $\lambda_x^T + \lambda_y^T = \lambda_0$ é $-\lambda_{x_j}$ vezes a desigualdade clique dupla, o que permite concluir que a mesma induz a uma faceta de $PO(G,d,s)$ sob as condições especificadas. □

Além das mostradas, há a possibilidade de adaptação de outras famílias de desigualdades válidas para o modelo de orientação usado no BCP, sendo objeto de estudo de trabalhos futuros.

5.1.2 Método *cut-and-branch*

Para a utilização do modelo baseado em orientação para resolução do BCP, utilizou-se uma estratégia algorítmica do tipo *cut-and-branch* (C&B), envolvendo a geração das desigualdades específicas do problema no nó raiz da árvore e, em seguida, modificando o esquema de solução para um *branch-and-cut* genérico.

No *cut-and-branch* desenvolvido, inicialmente resolve-se a relaxação linear do modelo baseado em orientação (ou seja, considerando as variáveis como contínuas). Se a solução obtida não for inteira, gera-se as desigualdades do tipo clique e clique dupla seguindo um procedimento iterativo, com base nas cliques maximais do grafo de entrada. Quando todas as desigualdades são adicionadas e a solução ainda não é inteira, as variáveis passam a serem consideradas como inteiras (exceto z_{max}) e o modelo resultante

é resolvido por um *branch-and-cut* genérico de modo a obter a solução ótima (ou pelo menos uma inteira, dependendo do que se deseja). O Algoritmo 5.3 mostra o esquema básico deste procedimento.

O modelo baseado em orientação e a estratégia *cut-and-branch* foram implementados usando o resolvedor IBM/ILOG CPLEX 12.6.3 e sua biblioteca *Concert Technology* em C++. Sendo assim, foram utilizados dois critérios de aplicação das desigualdades geradas, como descritos a seguir.

No primeiro critério, denotado por **C&B-BCP-Orient-Completo**, a solução da relaxação linear é verificada de modo a detectar quais desigualdades geradas (tanto clique quanto clique dupla) são violadas pela solução. Cada desigualdade violada é adicionada ao modelo, enquanto as outras são descartadas. A formulação resultante terá, então, as restrições originais e as desigualdades válidas adicionais, além de suas variáveis mudadas para inteiras (exceto z_{max}). O modelo então é resolvido novamente, de modo a obter a solução inteira ótima.

Já no segundo critério, denotado por **C&B-BCP-Orient-Incremental**, ao invés de adicionar as desigualdades direto ao modelo, todas as desigualdades são mantidas em um conjunto, e o modelo é resolvido com as variáveis mudadas para inteiras (exceto z_{max}). Porém, durante a enumeração do *branch-and-cut*, a solução obtida em cada subproblema é verificada, de modo que as desigualdades do conjunto que violarem a solução são inseridas no modelo atual caso ainda não estejam no mesmo. Esta estratégia é realizada por meio do método `IloCplex::addUserCut` do CPLEX.

Como discutido anteriormente, as desigualdades devem ser geradas previamente para que sejam usadas por ambos os critérios. Para obter todas as cliques maximais de um dado grafo, utilizou-se o pacote *QuickCliques* (Eppstein et al., 2010), que contém algoritmos para a enumeração de tais cliques. Optou-se pelo algoritmo proposto por Tomita et al. (2006), no entanto, todos os algoritmos do pacote têm bom desempenho. Destaca-se que, apesar de haver $O(3^{|V|/3})$ cliques maximais em um grafo (Tomita et al., 2006), na prática há muito menos na maioria dos grafos, logo, foi possível obter todas as cliques maximais em menos de um segundo para cada grafo usado.

Algoritmo 5.3 Estrutura básica do *cut-and-branch* para o BCP usando o modelo baseado em orientação.

Entrada: grafo $G = (V, E)$, distâncias $d : E \to \mathbb{Z}_{\geq 0}$.

Função CUTANDBRANCH-BCP-ORIENTACAO($G = (V, E), d$)
  $pl \leftarrow$ MONTARMODELOPL-ORIENTACAO-RELAXLINEAR(G)
  $(x, y, z_{\max}) \leftarrow$ RESOLVEDORPL(lp)
  se solução $(x, y, z_{\max})$ não for inteira **então**
    $H \leftarrow$ GERARCLIQUESMAXIMAIS(G)
    $conjDesigClique \leftarrow \varnothing$
    $conjDesigCliqueDupla \leftarrow \varnothing$
    para cada clique $K \in V$ **faça**
      $desigClique \leftarrow$ GERAR-DESIGCLIQUE(pl, $(x, y, z_{\max})$, i, K)
      $cjDesigClique \leftarrow cjDesigClique \cup desigClique$
      $desCliqDupla \leftarrow$ GERAR-DESIGCLIQUEDUPLA(pl, $(x, y, z_{\max})$, i, j, K)
      $cjDesigCliqueDupla \leftarrow cjDesigCliqueDupla \cup desCliqDupla$
      APLICAR-DESIGUALDADES(pl, $cjDesigClique$, $cjDesigCliqueDupla$)
    $pim \leftarrow$ MUDARVARIAVEISPARAINTEIRAS(pl)
    $(x, y, z_{\max}) \leftarrow$ RESOLVEDORPIM-B&C(pim)
  retorne $(x, y, z_{\max})$

5.1.2.1 Experimentos computacionais

Com a finalidade de verificar se as desigualdades válidas aplicadas no *cut-and-branch* descrito na seção anterior são capazes de melhorar o desempenho em alguma instância do BCP em relação ao modelo baseado em orientação puro, foram realizados experimentos utilizando instâncias da literatura e implementações dos métodos descritos neste capítulo.

O modelo baseado em orientação e o *cut-and-branch* foram implementados em linguagem C++ e utilizando o resolvedor IBM/ILOG CPLEX 12.6.3, com sua biblioteca *Concert Technology*, conforme citado na seção anterior. O compilador utilizado foi o `g++` 5.4. Os experimentos foram executados em um computador com processador Intel Core i7-3770 (3.4GHz), 8GB de RAM e sistema operacional Ubuntu Linux 16.04.2 LTS.

Cada experimento consistiu em aplicar tanto a implementação do modelo baseado em orientação puro (sem desigualdades adicionais) quanto o *cut-and-branch* (combinando o modelo com as desigualdades clique e clique dupla) usando ambos os critérios para cada instância do BCP do conjunto GEOM. Para fins de comparação, utilizou-se também uma implementação do modelo padrão de PI para o BCP (conforme descrito na Subseção 2.2 do Capítulo 2). O tempo limite imposto para cada método aplicado a cada instância foi de 3 horas (10800 segundos).

Para a realização dos experimentos, é necessário também determinar um limite su-

perior U suficientemente forte para $\chi(G,d)$, que será usado no lugar deste valor quando necessário. Para isso, aplicou-se a heurística gulosa baseada na alocação exaustiva por frequências (Sivarajan et al., 1989), também usada nos experimentos descritos na Seção 4.3.

O primeiro conjunto de experimentos envolve a comparação de desempenho entre a formulação padrão do BCP e o modelo baseado em orientação. Ambos foram executados sob as mesmas condições do CPLEX, destacando-se o uso de apenas uma *thread* e a não utilização de métodos para passagem de solução inicial completa ao resolvedor (ao contrário dos experimentos da Seção 4.3), de modo a investigar melhor o comportamento da enumeração em relação a obtenção de limites superiores. Os resultados são dados na Tabela 5.1, na qual pode-se ver que o modelo baseado em orientação é claramente superior à formulação padrão, o que é esperado, uma vez que esta última tem tamanho pseudopolinomial, ao contrário do modelo proposto neste capítulo, que é linear. Entretanto, como discutido na prova do Teorema 5.1, o parâmetro s tem impacto no tamanho do conjunto de soluções factíveis do politopo, de modo que a utilização de um valor muito grande pode degradar o desempenho e levar à instabilidade numérica. Nos resultados obtidos, definiu-se $s = U + 4d_{\max}$.

O segundo conjunto de experimentos consiste em uma verificação empírica do impacto da adição das desigualdades válidas ao modelo baseado em orientação, de acordo com as duas estratégias descritas anteriormente para o *cut-and-branch*. Os resultados obtidos são dados na Tabela 5.2. Das 33 instâncias GEOM, as abordagens *cut-and-branch* obtiveram soluções ótimas mais rapidamente que o modelo de orientação puro em 14 (42,43%) delas. Mais especificamente, C&B-BCP-Orient-Completo foi o melhor em 9 (27,3%) instâncias, enquanto C&B-BCP-Orient-Incremental foi o primeiro a encontrar a solução ótima em 5 (15,15%) instâncias. Tal resultado mostra que as desigualdades válidas são capazes de melhorar o processo de resolução do modelo baseado em orientação. Porém, a formulação pura conseguiu encontrar a solução ótima mais rapidamente ou melhores limites superiores (quando o tempo limite foi excedido para todas as abordagens) em 19 (57,6%) instâncias. Isso é explicado, principalmente, por três fatores. Primeiramente, na abordagem C&B-BCP-Orient-Completo, as desigualdades foram adicionadas diretamente no modelo gerado, o que aumentou seu tamanho de forma bastante significativa, levando a um número maior de variáveis duais e aumentando o tempo de solução

Tabela 5.1: Resultados obtidos pelas implementações dos modelos padrão e baseado em orientação para as instâncias do BCP do conjunto GEOM.

Instância	|V|	Modelo padrão				Modelo baseado em orientação			
		Melhor Sol.	Melhor LI	Gap (%)	Tempo (s)	Melhor Sol.	Melhor LI	Gap (%)	Tempo (s)
GEOM20	20	21	21	0.00	0.47	**21**	**21**	**0.00**	**0.02**
GEOM20a		20	20	0.00	2.03	**20**	**20**	**0.00**	**0.05**
GEOM20b		13	13	0.00	0.08	**13**	**13**	**0.00**	**0.02**
GEOM30	30	28	28	0.00	0.78	**28**	**28**	**0.00**	**0.15**
GEOM30a		27	27	0.00	3.58	**27**	**27**	**0.00**	**0.51**
GEOM30b		26	26	0.00	1.20	**26**	**26**	**0.00**	**0.49**
GEOM40	40	28	28	0.00	1.10	**28**	**28**	**0.00**	**0.35**
GEOM40a		37	37	0.00	505.66	**37**	**37**	**0.00**	**5.90**
GEOM40b		33	33	0.00	179.09	**33**	**33**	**0.00**	**51.90**
GEOM50	50	**28**	**28**	**0.00**	**3.85**	28	28	0.00	11.59
GEOM50a		50	50	0.00	8931.83	**50**	**50**	**0.00**	**34.12**
GEOM50b		≤ 35	34	3.00	10800.00	**35**	**35**	**0.00**	**159.32**
GEOM60	60	33	33	0.00	155.56	**33**	**33**	**0.00**	**14.35**
GEOM60a		≤ 56	32.13	43.00	10800.00	**50**	**50**	**0.00**	**680.90**
GEOM60b		≤ 51	14	73.00	10800.00	**41**	**41**	**0.00**	**1191.01**
GEOM70	70	**38**	**38**	**0.00**	**1337.17**	38	38	0.00	7212.36
GEOM70a		≤ 67	5	93.00	10800.00	**61**	**61**	**0.00**	**701.51**
GEOM70b		None	7	∞	10800.00	**47**	**47**	**0.00**	**117.80**
GEOM80	80	41	41	0.00	5383.91	**41**	**41**	**0.00**	**1023.54**
GEOM80a		≤ 73	2.176	97.00	10800.00	**63**	**63**	**0.00**	**7079.91**
GEOM80b		None	2.0272	∞	10800.00	**60**	**60**	**0.00**	**918.16**
GEOM90	90	≤ 48	8.367	83.00	10800.00	**46**	**46**	**0.00**	**39.20**
GEOM90a		≤ 81	2.093	97.00	10800.00	**≤ 64**	**38**	**41.00**	**10800.00**
GEOM90b		None	2	∞	10800.00	**≤ 71**	**50**	**30.00**	**10800.00**
GEOM100	100	≤ 55	5	91.00	10800.00	**50**	**50**	**0.00**	**63.00**
GEOM100a		≤ 97	1.404	99.00	10800.00	**≤ 70**	**47**	**33.00**	**10800.00**
GEOM100b		None	2	∞	10800.00	**≤ 73**	**40**	**45.00**	**10800.00**
GEOM110	110	≤ 58	2.287	96.00	10800.00	**≤ 50**	**49**	**2.00**	**10800.00**
GEOM110a		≤ 98	2	98.00	10800.00	**≤ 74**	**33**	**55.00**	**10800.00**
GEOM110b		None	2	∞	10800.00	**≤ 79**	**49**	**38.00**	**10800.00**
GEOM120	120	≤ 64	2.276	96.00	10800.00	**59**	**59**	**0.00**	**570.45**
GEOM120a		≤ 108	1.46	99.00	10800.00	**≤ 85**	**49**	**42.00**	**10800.00**
GEOM120b		None	1.4162	∞	10800.00	**≤ 90**	**32**	**64.00**	**10800.00**

para cada relaxação linear. O segundo fato é que todas as desigualdades foram geradas do conjunto de cliques maximais, de modo que não há algoritmos específicos de separação, o que aumenta o tempo de geração das desigualdades, além de impactar nas de tipo clique dupla, pois tais cliques maximais não levam necessariamente em conta situações onde alguns vértices são adjacentes apenas a um dos vértices da aresta selecionada. Por fim, o terceiro fato é que, devido às condições muito específicas para que as desigualdades induzam a facetas, esta característica nem sempre é válida para a maioria das instâncias do BCP, como as GEOM.

Apesar disso, o potencial das desigualdades para melhorar algoritmos baseados em planos de corte permanece válido. Dentre todos os resultados da Tabela 5.2 destaca-

Tabela 5.2: Resultados obtidos pelas implementações do modelo baseado em orientação puro e os algoritmos *cut-and-branch* para as instâncias do BCP do conjunto GEOM.

Instância	|V|	Modelo baseado em orientação puro				C&B-BCP-Orient-Completo						C&B-BCP-Orient-Incremental					
		Melhor Sol.	Melhor LI	Gap (%)	Tempo (s)	Melhor Sol.	Melhor LI	Gap (%)	Tempo Desig. (s)	Tempo PI (s)	Tempo Total (s)	Melhor Sol.	Melhor LI	Gap (%)	Tempo Desig. (s)	Tempo PI (s)	Tempo Total (s)
GEOM20	20	**21**	**21**	**0.00**	**0.02**	21	21	0.00	0.00	0.03	0.04	21	21	0.00	0.01	0.05	0.06
GEOM20a		**20**	**20**	**0.00**	**0.05**	20	20	0.00	0.01	0.18	0.19	20	20	0.00	0.03	0.13	0.16
GEOM20b		**13**	**13**	**0.00**	**0.02**	13	13	0.00	0.01	0.05	0.06	13	13	0.00	0.02	0.07	0.09
GEOM30	30	**28**	**28**	**0.00**	**0.15**	28	28	0.00	0.01	0.21	0.23	28	28	0.00	0.04	0.15	0.20
GEOM30a		27	27	0.00	0.51	**27**	**27**	**0.00**	**0.03**	**0.29**	**0.32**	27	27	0.00	0.09	0.27	0.36
GEOM30b		26	26	0.00	0.49	**26**	**26**	**0.00**	**0.02**	**0.13**	**0.15**	25	26	0.00	0.09	0.19	0.28
GEOM40	40	28	28	0.00	0.35	**28**	**28**	**0.00**	**0.02**	**0.21**	**0.23**	28	28	0.00	0.08	0.20	0.28
GEOM40a		37	37	0.00	5.90	**37**	**37**	**0.00**	**0.06**	**1.29**	**1.35**	$\leq$ 37	35	5.00	0.50	10800.00	10800.50
GEOM40b		33	33	0.00	51.90	**33**	**33**	**0.00**	**0.08**	**0.89**	**0.97**	33	33	0.00	0.43	1.07	1.50
GEOM50	50	28	28	0.00	11.59	**28**	**28**	**0.00**	**0.05**	**0.48**	**0.54**	28	28	0.00	0.24	0.62	0.86
GEOM50a		50	50	0.00	34.12	50	50	0.00	0.18	18.05	18.24	**50**	**50**	**0.00**	**0.98**	**15.61**	**16.59**
GEOM50b		35	35	0.00	159.32	35	35	0.00	0.19	174.05	174.24	**35**	**35**	**0.00**	**1.39**	**83.78**	**85.17**
GEOM60	60	**33**	**33**	**0.00**	**14.35**	33	33	0.00	0.08	715.62	715.70	33	33	0.00	0.60	255.09	255.69
GEOM60a		50	50	0.00	680.90	50	50	0.00	0.31	106.43	106.74	**50**	**50**	**0.00**	**2.68**	**62.75**	**65.44**
GEOM60b		**41**	**41**	**0.00**	**1191.01**	$\leq$ 42	39	7.00	0.47	10800.00	10800.47	41	41	0.00	4.33	6391.15	6395.48
GEOM70	70	38	38	0.00	7212.36	**38**	**38**	**0.00**	**0.20**	**2.32**	**2.52**	38	38	0.00	1.68	2.19	3.87
GEOM70a		**61**	**61**	**0.00**	**701.51**	61	61	0.00	0.59	1355.45	1356.04	61	61	0.00	6.33	715.02	721.35
GEOM70b		**47**	**47**	**0.00**	**117.80**	$\leq$ 48	40	17.00	0.64	10800.00	10800.64	$\leq$ 49	38	22.00	9.48	10800.00	10809.49
GEOM80	80	41	41	0.00	1023.54	41	41	0.00	0.38	1575.27	1575.66	**41**	**41**	**0.00**	**3.33**	**1003.32**	**1006.65**
GEOM80a		63	63	0.00	7079.91	63	63	0.00	1.20	5575.55	5576.75	**63**	**63**	**0.00**	**15.05**	**4465.78**	**4480.84**
GEOM80b		**60**	**60**	**0.00**	**918.16**	$\leq$ 62	42	32.00	1.49	10800.00	10801.49	$\leq$ 62	36.749	41.00	20.69	10800.00	10820.69
GEOM90	90	46	46	0.00	39.20	**46**	**46**	**0.00**	**0.50**	**14.04**	**14.54**	46	46	0.00	6.58	15.74	22.32
GEOM90a		**$\leq$ 64**	**38**	**41.00**	**10800.00**	$\leq$ 67	42.932	36.00	1.91	10800.00	10801.91	$\leq$ 67	47.558	29.00	31.58	10800.00	10831.58
GEOM90b		**$\leq$ 71**	**50**	**30.00**	**10800.00**	$\leq$ 76	44	42.00	2.58	10800.00	10802.58	$\leq$ 75	46.164	38.00	40.35	10800.00	10840.35
GEOM100	100	**50**	**50**	**0.00**	**63.00**	50	50	0.00	0.75	115.08	115.83	50	50	0.00	11.88	307.88	319.76
GEOM100a		**$\leq$ 70**	**47**	**33.00**	**10800.00**	$\leq$ 77	38.321	50.00	3.61	10800.00	10803.61	$\leq$ 76	42.088	45.00	60.80	10800.00	10860.80
GEOM100b		**$\leq$ 73**	**40**	**45.00**	**10800.00**	$\leq$ 80	37.028	54.00	6.36	10800.00	10806.36	$\leq$ 80	39.217	51.00	82.78	10800.00	10882.78
GEOM110	110	$\leq$ 50	49	0.02	10800.00	**50**	**50**	**0.00**	**1.19**	**565.37**	**566.58**	$\leq$ 51	50	2.00	17.80	10800.00	10817.80
GEOM110a		**$\leq$ 74**	**33**	**55.00**	**10800.00**	$\leq$ 79	39.033	51.00	6.59	10800.00	10806.59	$\leq$ 81	40.361	50.00	95.03	10800.00	10895.03
GEOM110b		**$\leq$ 79**	**49**	**38.00**	**10800.00**	$\leq$ 84	39.592	53.00	8.40	10800.00	10808.40	$\leq$ 86	40.917	52.00	134.31	10800.00	109234.31
GEOM120	120	**59**	**59**	**0.00**	**570.45**	59	59	0.00	1.94	3570.58	3572.53	59	59	0.00	32.10	3074.84	3106.95
GEOM120a		**$\leq$ 85**	**49**	**42.00**	**10800.00**	$\leq$ 90	40.614	55.00	8.91	10800.00	10808.91	$\leq$ 96	37.053	61.00	199.13	10800.00	10999.13
GEOM120b		**$\leq$ 90**	**32**	**64.00**	**10800.00**	$\leq$ 94	37.31	60.00	27.35	10800.00	10827.35	$\leq$ 95	39.358	59.00	240.54	10800.00	11040.54

Tabela 5.3: Cortes genéricos gerados pelo CPLEX em cada experimento.

Instância	|V|	Modelo baseado em orientação puro							C&B-BCP-Orient-Completo							C&B-BCP-Orient-Incremental						
		Cover	*Flow*	*Lift&Proj.*	*Impl. Bnd.*	*MI-Round*	*0-Half*	Gomory	Cover	*Flow*	*Lift&Proj.*	*Impl. Bnd.*	*MI-Round*	*0-Half*	Gomory	Cover	*Flow*	*Lift&Proj.*	*Impl. Bnd.*	*MI-Round*	*0-Half*	Gomory
GEOM20		0	0	0	23	4	0	1	0	0	2	5	1	2	9	0	0	0	10	5	5	5
GEOM20a	20	0	0	0	47	7	4	4	0	0	1	11	1	5	11	0	0	0	4	10	6	11
GEOM20b		0	0	0	4	0	0	4	0	0	3	10	1	1	5	0	0	0	4	14	8	10
GEOM30		0	0	0	71	23	8	2	0	0	5	6	0	8	20	0	0	0	9	13	12	17
GEOM30a	30	0	0	0	137	35	6	4	0	0	0	10	4	14	22	0	0	0	5	23	25	24
GEOM30b		0	7	0	24	24	5	11	0	0	9	6	4	5	24	0	0	0	4	13	6	7
GEOM40		0	9	0	47	49	9	4	0	0	6	10	0	7	20	0	0	0	14	24	18	18
GEOM40a	40	1	20	10	149	171	22	23	0	0	7	29	5	21	39	0	0	0	45	84	90	56
GEOM40b		0	24	6	93	119	15	10	0	0	2	58	1	3	26	0	0	0	18	25	19	15
GEOM50		0	11	4	131	98	9	13	0	0	6	13	1	14	40	0	0	0	11	19	28	18
GEOM50a	50	1	23	14	239	119	24	25	0	0	4	113	2	11	62	0	0	5	46	39	59	70
GEOM50b		0	10	2	287	119	13	19	0	0	0	281	14	25	81	0	0	0	81	62	65	27
GEOM60		0	14	5	19	79	11	17	0	0	12	245	16	32	76	0	0	0	60	47	47	32
GEOM60a	60	4	84	17	599	509	82	40	0	0	0	129	12	70	62	0	0	12	71	45	93	34
GEOM60b		1	34	8	731	374	63	49	0	0	8	287	6	39	95	2	0	8	181	85	101	50
GEOM70		2	14	6	537	346	30	38	0	0	0	45	10	8	17	0	0	0	15	20	61	45
GEOM70a	70	4	53	10	815	430	73	51	0	0	3	267	4	49	61	1	0	5	129	60	87	69
GEOM70b		0	75	12	257	540	81	63	0	0	0	349	3	36	84	1	0	15	198	56	119	37
GEOM80		1	32	11	523	351	64	30	0	0	2	294	25	40	32	0	0	7	82	58	149	37
GEOM80a	80	52	263	40	1600	1674	175	86	1	0	2	579	31	107	80	6	0	9	131	129	190	56
GEOM80b		16	117	17	617	856	150	52	0	0	2	567	40	89	90	0	0	4	247	159	116	50
GEOM90		5	31	7	187	300	52	38	0	0	0	130	9	31	38	0	0	9	57	15	57	29
GEOM90a	90	55	384	49	778	2429	345	56	5	0	0	358	21	56	70	10	0	5	198	131	83	63
GEOM90b		46	203	16	849	1670	265	66	5	0	2	607	17	67	87	8	0	0	117	142	100	77
GEOM100		9	87	9	951	821	109	45	0	0	3	229	10	48	55	1	0	9	145	102	235	46
GEOM100a	100	35	279	14	1076	2014	220	54	8	0	3	633	30	82	82	37	0	0	136	117	100	77
GEOM100b		103	556	9	2745	3656	359	79	15	0	0	632	38	116	75	16	0	0	130	150	157	77
GEOM110		17	189	10	1509	1261	162	62	0	0	4	242	18	117	88	0	0	11	126	80	116	31
GEOM110a	110	131	722	13	3014	4378	453	103	15	0	2	282	31	66	71	44	0	0	158	137	64	68
GEOM110b		63	335	7	1616	2199	261	82	17	0	1	384	31	92	78	35	0	0	175	163	79	70
GEOM120		11	113	25	1351	877	178	42	3	0	6	223	13	93	71	7	0	11	172	63	127	49
GEOM120a	120	47	275	12	1772	2121	312	76	76	0	0	257	101	82	87	169	0	0	168	369	231	87
GEOM120b		83	645	4	1848	3272	377	63	41	0	0	419	38	77	69	102	0	0	94	252	97	70

se que, para a instância GEOM40b, o modelo de orientação puro (sem as desigualdades adicionais) levou 51,9s para encontrar a solução ótima, enquanto C&B-BCP-Orient-Completo usou 0,97s (5350,5% mais rápido) e C&B-BCP-Orient-Incremental precisou de 1,5s (3460% mais rápido). Além disso, para a instância GEOM60a, o modelo de orientação puro utilizou 680,9s para achar a solução ótima, mais que C&B-BCP-Orient-Completo, que levou 106,74s (638% mais rápido) e C&B-BCP-Orient-Incremental, que usou 65,44s (1040,5% mais rápido). As maiores melhorias obtidas no *cut-and-branch* ocorreram na instância GEOM70, para a qual o modelo puro usou 7212,36s para encontrar a solução ótima, enquanto C&B-BCP-Orient-Completo levou 2,12s (340205,7% mais rápido) e C&B-BCP-Orient-Incremental usou 3,87s (104983,4% mais rápido).

Por fim, mensurou-se também a quantidade de cortes genéricos gerados pelo CPLEX quando as variáveis são mudadas para inteiras, levando em conta que todos os parrâmetros do resolvedor são os mesmos para todos os experimentos. A Tabela 5.3 apresenta tais estatísticas. Pode-se observar que o modelo puro usa mais cortes do tipo *flow* (fluxo) e *mixed-integer rounding* (arredondamento inteiro misto) em comparação com as abordagens *cut-and-branch*. Para instâncias menores, ambos C&B-BCP-Orient-Completo e C&B-BCP-Orient-Incremental não geram cortes *cover* (cobertura), mas a medida que o tamanho da instância aumenta, alguns cortes deste tipo são gerados. Um comportamento similar é observado para cortes *flow* no modelo puro, entretanto, nenhum deste tipo é gerado no *cut-and-branch*. Os outros tipos de cortes seguem um padrão similar de geração em ambas as abordagens *cut-and-branch* e o modelo baseado em orientação puro, onde C&B-BCP-Orient-Completo usa mais cortes *implied-bound* (limite implícito), enquanto C&B-BCP-Orient-Incremental usa mais cortes *mixed-integer rounding* e *zero-half*.

5.2 Modelo baseado em distâncias

Como citado anteriormente, o modelo baseado em orientação foi originalmente proposto para o problema de alocação de canais com minimização de interferência (MI-CAP), sendo também usado no cenário onde deve-se minimizar a interferência de canais (cores) adjacentes. Para este segundo caso, Delle-Donne (2016) propôs uma variação do modelo baseado em orientação, agora dita ser baseada em distâncias, onde as variáveis indicam a diferença entre cores alocadas a cada par de vértices, ao invés das cores dos vértices em

si.

Seguindo essa formulação de Delle-Donne (2016), bem como o modelo baseado em orientação para o BCP descrito no capítulo anterior, desenvolveu-se uma adaptação preliminar d a formulação baseada em distâncias para o problema de coloração clássica (VCP), onde estudou-se o poliedro definido pela formulação com as soluções do VCP, porém sem a determinação de uma função objetivo. Ressalta-se que tal formulação também pode ser aplicada ao Problema de Coloração Mínima em Geometria de Distâncias Uniformes Iguais (MinGEQ-CDGP-Unif), descrito na Seção 3.2.1 do Capítulo 3, devido a sua equivalência com o VCP.

A formulação baseada em distâncias usa os seguintes conjuntos de variáveis:

- q_{ij} = diferença entre cores atribuídas a i e j (ou seja, $q_{ij} = x(i) - x(j)$).
- $y_{ij} = \begin{cases} 1 & \text{se } q_{ij} < 0; \\ 0 & \text{caso contrário.} \end{cases}$

Com tais variáveis, sendo $|C|$ um limite superior para o número cromático (ou seja, tem-se um conjunto de cores $C = \{1, 2, \ldots, |C|\}$), define-se o seguinte politopo:

$$q_{ij} = q_{ik} + q_{kj} \qquad \forall i, j, k \in V,\ i < k < j \tag{5.11}$$

$$q_{ij} \geq 1 - |C| y_{ij} \qquad \forall (i, j) \in E,\ i < j \tag{5.12}$$

$$q_{ij} \leq -1 + |C|(1 - y_{ij}) \qquad \forall (i, j) \in E,\ i < j \tag{5.13}$$

$$q_{ij} \in \{-|C| + 1, \ldots, |C| - 1\} \quad \forall i, j \in V,\ i < j \tag{5.14}$$

$$y_{ij} \in \{0, 1\} \qquad \forall (i, j) \in E,\ i < j \tag{5.15}$$

As restrições (5.11) refletem a separação de cores entre os vértices, inclusive entre os não adjacentes entre si, de acordo com a interpretação das variáveis x_{ij}. Os conjuntos de restrições (5.12) e (5.13) induzem à orientação do grafo de acordo com as cores atribuídas. Por fim, (5.14) e (5.15) referem-se a integralidade e limites das variáveis.

A formulação acima segue o conceito de 0-esferas utilizadas nos modelos de colorações com distâncias descritos no Capítulo 3. Desse modo, cada variável corresponde a uma 0-esfera, de modo que o módulo do valor da mesma indica o comprimento do segmento de reta correspondente. Um exemplo de instância cuja solução é codificada pela

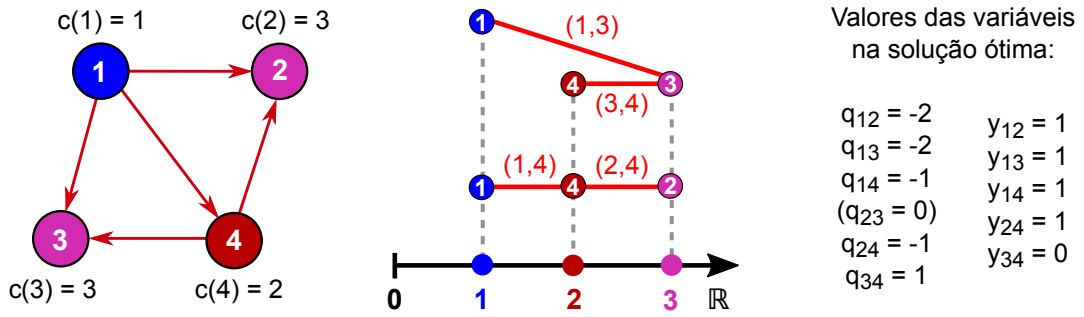


Figura 5.4: Exemplo de instância de coloração clássica (VCP) cuja solução é codificada de acordo com o modelo baseado em distâncias, sendo também mostrada pela representação em 0-esferas. No grafo, também é mostrada a orientação induzida.

formulação acima, bem como sua representação por 0-esferas, pode ser visto na Figura 5.4.

O conjunto de restrições 5.11 é composto de $O(|V|^3)$ equações, porém, o mesmo pode ser substituído por um conjunto de $O(|V|^2)$ equações, como visto a seguir.

Teorema 5.6. *Se $V = \{1,\ldots,|V|\}$, então as restrições (5.11) equivalem a:*

$$q_{i,i+1} + q_{i+1,i+2} = q_{i,i+2} \qquad \forall i \in V, i \leq |V|-2 \tag{5.16}$$

$$q_{ij} + q_{i+1,j-1} = q_{i,j-1} + q_{i+1,j} \qquad \forall i,j \in V, i \leq |V|-3, i+3 \leq j \tag{5.17}$$

Prova. A demonstração a seguir é baseada na prova do Teorema 2.3.1 de Delle-Donne (2016). O enunciado do teorema equivale a dizer que, para $V = \{1,\ldots,n\}$, tem-se que as restrições (5.11) são válidas se, e somente se, as restrições (5.16) e (5.17) também forem. Assim, deve-se proceder com uma prova em duas partes, como feito a seguir.

($\rightarrow$) Considere um vetor $q \in \mathbb{R}^{\frac{n(n-1)}{2}}$ para o qual as restrições (5.11) são respeitadas. É possível observar que (5.11) e (5.16) são iguais se $k = i+1$ e $j = i+2$, como visto a seguir:

(5.11): $q_{ij} = q_{ik} + q_{kj} \;\Rightarrow\; q_{ik} + q_{kj} = q_{ij} \xrightarrow{k=i+1,\ j=i+2} q_{i,i+1} + q_{i+1,i+2} = q_{i,i+2}$

(5.16): $q_{i,i+1} + q_{i+1,i+2} = q_{i,i+2}$

Assim, q satisfaz (5.16). Considere agora dois vértices a e b tais que $1 \leq a \leq |V|-3$ e $a+3 \leq b \leq |V|$. Desse modo, tem-se as duas seguintes expressões, ambas derivadas de (5.11):

$$q_{ij} = q_{ik} + q_{kj} \xrightarrow{i=a,\ k=b-1,\ j=b} q_{ab} = q_{a,b-1} + q_{b-1,b}$$
$$q_{ij} = q_{ik} + q_{kj} \xrightarrow{i=a+1,\ j=b-1,\ k=b} q_{a+1,b} = q_{a+1,b-1} + q_{b-1,b}$$

Subtraindo-se as duas expressões e reorganizando o resultado, obtém-se:

$$\begin{aligned} & q_{ab} = q_{a,b-1} + q_{b-1,b} \\ - \quad & q_{a+1,b} = q_{a+1,b-1} + q_{b-1,b} \\ \hline & q_{ab} - q_{a+1,b} = q_{a,b+1} - q_{a+1,b-1} \\ \Rightarrow \quad & q_{ab} + q_{a+1,b-1} = q_{a+1,b} + q_{a,b+1} \end{aligned}$$

O resultado é equivalente à expressão 5.17, bastando efetuar as substituições de variáveis $a = i$ e $b = j$. Assim, q também satisfaz tal conjunto de restrições.

($\leftarrow$) Considere agora um vetor $q \in \mathbb{R}^{\frac{n(n-1)}{2}}$ para o qual as restrições (5.16) e (5.17) são respeitadas. Será mostrado a seguir que q também satisfaz (5.11). Sejam i, j, k vértices do grafo, de modo que $1 \leq i,k,j \leq |V|$ e $i < k < j$. Alguns casos serão considerados a seguir.

- Caso 1: $k = i+1$ e $j = k+1$ (ou seja, $j = i+2$). Neste caso, tem-se que (5.16) e (5.11) são equivalentes, como já mostrado anteriormente e replicado aqui:

  (5.16): $q_{i,i+1} + q_{i+1,i+2} = q_{i,i+2} \xrightarrow{i+1=k,\ i+2=j} q_{ij} = q_{ik} + q_{kj}$
  (5.11): $q_{ij} = q_{ik} + q_{kj}$

  Como (5.16) é válido, então (5.11) também será neste caso.

- Caso 2: $k = i+1$ e $j > k+1$. Iniciando de (5.16), soma-se a expressão com (5.17) várias vezes, de modo que, na primeira vez, substitui-se j por $k+2$, e nas próximas somas incrementa-se tal expressão em uma unidade, seguindo então $k+3, k+4, \ldots,$

até o valor j selecionado, como feito a seguir.

$$
\begin{aligned}
& q_{k-1,k+1} = q_{k-1,k} + q_{k,k+1} \\
+ \quad & q_{k-1,k+2} + q_{k,k+1} = q_{k-1,k+1} + q_{k,k+2} \\
+ \quad & q_{k-1,k+3} + q_{k,k+2} = q_{k-1,k+2} + q_{k,k+3} \\
+ \quad & q_{k-1,k+4} + q_{k,k+3} = q_{k-1,k+3} + q_{k,k+4} \\
+ \quad & \dots \\
+ \quad & q_{k-1,j} + q_{k,j-1} = q_{k-1,j-1} + q_{kj} \\
\hline
& q_{k-1,j} = q_{k-1,k} + q_{kj} \\
\xrightarrow{k=i+1} \quad & q_{ij} = q_{ik} + q_{kj}
\end{aligned}
$$

O resultado obtido é exatamente igual a 5.11, o que leva a sua validade.

- Caso 3: $k > i+1$. Para este caso, partindo de (5.17) com j trocado por $k+1$, deve-se efetuar sucessivas somas com a mesma expressão, mas trocando j por $k+2$, $k+3$, $\dots$, até o valor j original, como feito a seguir:

$$
\begin{aligned}
& q_{i,k+1} + q_{i+1,k} = q_{ik} + q_{i+1,k+1} \\
+ \quad & q_{i,k+2} + q_{i+1,k+1} = q_{i,k+1} + q_{i+1,k+2} \\
+ \quad & q_{i,k+3} + q_{i+1,k+2} = q_{i,k+2} + q_{i+1,k+3} \\
+ \quad & \dots \\
+ \quad & q_{ij} + q_{i+1,j-1} = q_{i,j-1} + q_{i+1,j} \\
\hline
& q_{ij} + q_{i+1,k} = q_{ik} + q_{i+1,j}
\end{aligned}
$$

Em seguida, deve-se efetuar, com o resultado obtido, novas sucessivas somas de (5.17), agora trocando i por $i+1$ (em todas as somas) e j por $k+1$, $k+2$, $k+3$, $\dots$,

até o valor j original, como feito a seguir:

$$
\begin{aligned}
& q_{ij} + \cancel{q_{i+1,k}} = q_{ik} + \cancel{q_{i+1,j}} \\
+ \quad & \cancel{q_{i+1,k+1}} + q_{i+2,k} = \cancel{q_{i+1,k}} + \cancel{q_{i+2,k+1}} \\
+ \quad & \cancel{q_{i+1,k+2}} + \cancel{q_{i+2,k+1}} = \cancel{q_{i+1,k+1}} + \cancel{q_{i+2,k+2}} \\
+ \quad & \cancel{q_{i+1,k+3}} + \cancel{q_{i+2,k+2}} = \cancel{q_{i+1,k+2}} + \cancel{q_{i+2,k+3}} \\
+ \quad & \dots \\
+ \quad & \cancel{q_{i+1,j}} + \cancel{q_{i+2,j-1}} = \cancel{q_{i+1,j-1}} + q_{i+2,j} \\
\hline
& q_{ij} + q_{i+2,k} = q_{ik} + q_{i+2,j}
\end{aligned}
$$

O processo deve ser repetido novamente, sempre somando o resultado do último somatório com novas somas de (5.17), incrementando i em uma unidade em relação ao último valor considerado até $k-2$ e, para cada variação em i, mudando j por $k+1$, $k+2$, $k+3$, ..., até o valor j original. Seguindo o padrão acima, observa-se que, para $i+\ell$ (onde ℓ é um número natural), o somatório resultante é

$$q_{ij} + q_{i+\ell+1,k} = q_{ik} + q_{i+\ell+1,j}$$

Assim, para a troca de i por $k-2$ (ou seja, $\ell = k-i-2$), tem-se:

$$q_{ij} + q_{k-1,k} = q_{ik} + q_{k-1,j}$$

Somando-se esta expressão com a restrição 5.16, mas substituindo a variável i da mesma por $k-1$ (observando que o i desta **não é**, necessariamente, o mesmo das somas realizadas até o momento, já que a restrição será válida para qualquer $i < k < j$), obtém-se:

$$
\begin{aligned}
& q_{ij} + \cancel{q_{k-1,k}} = q_{ik} + \cancel{q_{k-1,j}} \\
+ \quad & \cancel{q_{k-1,j}} = \cancel{q_{k-1,k}} + q_{kj} \\
\hline
& q_{ij} = q_{ik} + q_{kj}
\end{aligned}
$$

Que é exatamente a expressão (5.11).

A partir das demonstrações acima, tem-se provada a equivalência entre as restrições (5.11) e as expressões (5.16) e (5.17).

□

Seguindo o Teorema 5.6, então, a formulação baseada em distâncias para o VCP pode ser reescrita da seguinte forma:

$$q_{i,i+1} + q_{i+1,i+2} = q_{i,i+2} \qquad \forall i \in V, i \leq |V| - 2 \tag{5.18}$$

$$q_{ij} + q_{i+1,j-1} = q_{i,j-1} + q_{i+1,j} \quad \forall i,j \in V, i \leq |V| - 3, i+3 \leq j \tag{5.19}$$

$$q_{ij} \geq 1 - |C| y_{ij} \qquad \forall (i,j) \in E,\ i < j \tag{5.20}$$

$$q_{ij} \leq -1 + |C|(1 - y_{ij}) \qquad \forall (i,j) \in E,\ i < j \tag{5.21}$$

$$q_{ij} \in \{-|C|+1, \ldots, |C|-1\} \qquad \forall i,j \in V,\ i < j \tag{5.22}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in E,\ i < j \tag{5.23}$$

Para o modelo definido pelas expressões (5.18)–(5.23), tem-se que existem $\frac{|V|(|V|-1)}{2}$ variáveis q e $|E|$ variáveis y, de modo que a formulação tem, então $O(|V|^2)$ variáveis. Além disso, o maior conjunto de restrições é (5.19), de modo que a formulação tem $O(|V|^2)$ restrições.

Seja $PD(G,C)$ o fecho convexo de soluções válidas do modelo baseado em distâncias, ou seja, dos pontos $(q,y) \in \mathbb{R}^{\frac{|V|(|V|-1)}{2}+|E|}$ que satisfazem as restrições (5.18)–(5.23). A dimensão de tal politopo é dada pelo teorema a seguir.

Teorema 5.7. *Se* $|C| > \chi(G) + 1$*, então* $\dim(PD(G,C)) = |V| + |E| - 1$.

Prova. A dimensão de um politopo P (ou seja, $dim(P)$) equivale à dimensão de seu fecho afim ($aff(P)$), ou seja o maior número possível de pontos afim-independentes menos 1. Se $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$, pode-se particionar $Ax \leq b$ em $A^< x < b^<$ e $A^= x = b^=$. Logo:

$$dim(P) = n - posto(A^=) \tag{5.24}$$

onde $posto(A^=)$ é o número de linhas linearmente independentes de $rank(A^=)$ (Ferreira e Wakabayashi, 1996; Pendavingh, 2015).

Seguindo tais conceitos, seja $PD^=$ o conjunto de igualdades de $PD(G,C)$ e $PD^<$ o de desigualdades. Para mostrar que $dim(PD(G,C)) = |V| + |E| - 1$ se $|C| > \chi(G) + 1$, será

mostrado que para tal limite de $|C|$, existirá ao menos um ponto $(q,y)^1$ tal que (5.20) será uma desigualdade estrita, ou seja:

$$q_{ij} > 1 - |C| y_{ij} \quad \forall (i,j) \in E,\ i < j \tag{5.25}$$

e ao menos um ponto $(q,y)^2$ tal que 5.21 será uma desigualdade estrita, ou seja:

$$q_{ij} < -1 + |C|(1 - y_{ij}) \quad \forall (i,j) \in E,\ i < j \tag{5.26}$$

(observando que $(q,y)^1$ não precisa ser necessariamente diferente de $(q,y)^2$). Já que, para $|C| > \chi(G) + 1$, o menor valor inteiro possível para $|C|$ é $\chi(G) + 2$, pode-se efetuar as seguintes substituições em (5.25) e (5.26):

$$q_{ij} > 1 - (\chi(G) + 2) y_{ij} \qquad \forall (i,j) \in E,\ i < j \tag{5.27}$$

$$q_{ij} < -1 + (\chi(G) + 2)(1 - y_{ij}) \quad \forall (i,j) \in E,\ i < j \tag{5.28}$$

Ao encontrar, para cada uma das restrições modificadas, ao menos uma solução que a satisfaça, garante-se que que as únicas em $P^{=}$ são (5.18)–(5.19). Desse modo, será possível usar a fórmula em (5.24) para calcular $dim(PD(G,C))$.

Primeiramente, observe que para qualquer grafo não direcionado simples G, tem-se, para toda aresta $(i,j) \in E$, que $1 \leq |x(i) - x(j)| \leq \chi(G) - 1$, já que $\chi(G) - 1$, que é a diferença entre a maior ($\chi(G)$) e a menor (1) cor usada e a diferença de cor entre dois vértices usando cores consecutivas é 1.

Agora, considere uma aresta $(i,j) \in E$ arbitrária. Quando $y_{ij} = 1$ (ou seja, $q_{ij} < 0$), tem-se que:

$$q_{ij} > 1 - (\chi(G) + 2) y_{ij} \tag{5.29}$$

$$q_{ij} < -1 + (\chi(G) + 2)(1 - y_{ij}) \tag{5.30}$$

Que resulta em:

$$q_{ij} > -1 - \chi(G) \tag{5.31}$$

$$q_{ij} < -1 \tag{5.32}$$

De modo similar, quando $y_{ij} = 0$ (ou seja, $q_{ij} > 0$), tem-se que:

$$q_{ij} > 1 - (\chi(G) + 2)y_{ij} \tag{5.33}$$

$$q_{ij} < -1 + (\chi(G) + 2)(1 - y_{ij}) \tag{5.34}$$

Que resulta em:

$$q_{ij} > 1 \tag{5.35}$$

$$q_{ij} < \chi(G) + 1 \tag{5.36}$$

Tem-se então que (5.31)–(5.32) e (5.35)–(5.36) implicam que:

$$1 < |q_{ij}| < \chi(G) + 1$$

Desse modo, para que uma solução satisfaça as restrições modificadas para uma aresta (i, j), a coloração correspondente deve garantir que $2 \leq |x(i) - x(j)| \leq \chi(G)$.

Considere uma coloração $x^{OPT} : V \to [1..\chi(G)]$ (ou seja, a ótima). Seja:

- $v_1, v_2, \ldots, v_{|V|}$ uma ordenação dos vértices tal que $x(v_a) \leq x(v_{a+1})$ para $a = 1, 2, \ldots, |V|$.
- $e_1, e_2, \ldots, e_{|E|}$ uma ordenação das arestas tal que, sendo $e_1 = (i_1, j_1)$ e $e_2 = (i_2, j_2)$, tem-se que $e_1 = e_2$ se, e somente se, $i_1 = i_2$ e $j_1 = j_2$; $e_1 < e_2$ se, e somente se, $i_1 < i_2$ ou $i_i = i_2$ e $j_1 < j_2$; e $e_1 > e_2$ se, e somente se, $i_1 > i_2$ ou $i_i = i_2$ e $j_1 > j_2$ (ou seja, arestas ordenadas de acordo com o primeiro vértice e, em caso de serem o mesmo, de acordo com o segundo vértice).

Sem perda de generalidade, considere que $v_1 < v_2 < \ldots, v_|V|$ (já que os vértices podem ser rotulados de acordo com esta ordem e a das cores)) e que um ponto correspondente a uma solução de $PD(G,C)$ é representado como (q,y), onde $q = (q_{v_1v_2}, q_{v_1v_3}, \ldots, q_{v_1v_{|V|}}, q_{v_2v_1}, \ldots, q_{v_{|V|-1}v_{|V|}})$ e $y = (y_{e_1}, y_{e_2}, \ldots, y_{e_{|E|}}$. Denote como $(q,y)^{OPT}$ o ponto correspondente à coloração x^{OPT}.

A partir de x^{OPT}, construa as colorações $x^k : V \to [1..\chi(G)+1]$ para todo $k = |V|, |V| - 1, \ldots, 1$, onde $x^k(v_a) = x(v_a)$ para $a = 1, 2, \ldots, |V| - k$ e $x^k(v_b) = c(v_b) + 1$ para $b = |V| - k + 1, \ldots, |V|$. Tais soluções existem pois $|C| > \chi(G)$. Para cada x^k, o ponto correspondente em $PD(G,C)$ será $(q,y)^k$. Observe que:

- $q^k_{v_a v_b} = q^{|V|}_{v_a v_b}$ para todo $a, b < k$, uma vez que as cores de v_a e v_b para tais valores de a e b são as mesmas da solução inicial (o que mantém o valor da distância entre as cores sem alteração).

- $q^k_{v_a v_k} = q^{|V|}_{v_a v_b} + 1$ para todo $a < k$, pois como a cor de v_k foi incrementada, a distância da mesma para as cores dos vértices $v_1, v_2, \ldots, v_{k-1}$ (que são menores, pela ordenação dos vértices) é aumentada de acordo com o incremento da cor. Ressalta-se que como consequência disso, existirá, para cada aresta (i, v_k), ao menos um ponto onde as restrições modificadas (5.27) e (5.28) são satisfeitas.

- $q^k_{v_k v_b} = q^{|V|}_{v_k v_b}$ para todo $b > k$, já que as cores dos vértices $v_{k+1}, v_{k+2}, \ldots, v_{|V|}$ também já foram incrementadas em relação à coloração original $c^|V|$, de modo que o incremento na cor de v_k cancela a distância adicional para as cores dos vértices posteriores.

Pela existência dos pontos $(q,y)^k$, tem-se então que para cada $(i,j) \in E$ há ao menos um ponto em $PD(G,C)$ onde as restrições (5.27) e (5.28) são satisfeitas, isto é, as restrições originais (5.20) e (5.21) são satisfeitas como desigualdades estritas. Logo, tem-se que apenas (5.18) e (5.19) serão sempre igualdades em todos os pontos de $PD(G,C)$, de modo que $PD^{=} = \{(5.18), (5.19)\}$.

Observando-se o modelo definido por (5.18)–(5.23), tem-se que existem $\frac{|V|(|V|-1)}{2}$ variáveis q e $|E|$ variáveis y, totalizando $\frac{|V|(|V|-1)}{2} + |E|$ variáveis, logo, $PD(G,C) \in \mathbb{R}^{\frac{|V|(|V|-1)}{2}+|E|}$. Em relação às igualdades, existem $|V|-2$ restrições em (5.18) e $\frac{|V|^2-5|V|+6}{2}$ restrições em (5.19). Portanto, tem-se que:

$$posto(PD^{=}) = |V| - 2 + \frac{|V|^2 - 5|V| + 6}{2}$$

Sendo assim, pode-se calcular a dimensão do politopo pela fórmula (5.24):

$$dim(PD(G,C)) = \frac{|V|(|V|-1)}{2} + |E| - \left(|V| - 2 + \frac{|V|^2 - 5|V| + 6}{2}\right) \quad \Rightarrow$$

$$dim(PD(G,C)) = \frac{|V|^2 - |V| + 2|E| - (|V|^2 - 5|V| + 6 + 2|V| - 4)}{2} \quad \Rightarrow$$

$$dim(PD(G,C)) = |V| + |E| - 1$$

□

Como consequência direta do Teorema 5.7, tem-se o seguinte resultado adicional.

Corolário 5.1. *Se $|C| > \chi(G)+1$, então $PD(G,C)$ não tem dimensão plena.*

5.2.1 Desigualdades válidas

Uma vez que o modelo baseado em distâncias, definido pelas expressões (5.18)–(5.23), é derivado da formulação baseada em orientação, é possível utilizar as desigualdades válidas definidas para esta última na primeira, seguindo algumas modificações. Desse modo, é possível dizer que as desigualdades de um modelo são obtidas a partir das do outro e vice-versa, como será mostrado a seguir.

5.2.1.1 Equivalências com modelo de orientação

Nos teoremas a seguir, considere que que d_1 denota uma função de distância unitária, ou seja, $d_1 : E \rightarrow \{1\}$. O primeiro deles mostra que as desigualdades válidas para o modelo baseado em orientação podem ser usadas na formulação de distâncias.

Teorema 5.8. *Seja $\alpha x_i + \pi y \leq \alpha x_j + \pi_0$ uma desigualdade válida (respectivamente, indutora de faceta) para $PO(G,d_1,|C|)$, onde $(i,j) \in E$. Então, $\alpha q_{ij} + \pi y \leq \pi_0$ é válida (respectivamente, indutora de faceta se $|C| \geq \chi(G)+2$) para $PD(G,C)$.*

Prova. A validade da desigualdade para $PD(G,C)$ segue diretamente da definição da variável q_{ij}, uma vez que:

$$\alpha x_i + \pi y \leq \alpha x_j + \pi_0 \quad \Rightarrow \quad \alpha x_i - \alpha x_j \pi y \leq \alpha x_j + \pi_0 \quad \Rightarrow \quad \alpha(x_i - x_j)\pi y \leq \alpha x_j + \pi_0$$

$$\Rightarrow \quad \alpha q_{ij} + \pi y \leq \pi_0$$

Para verificar a indução de faceta, considere $|V|+|E|$ pontos afim independentes $(x^k, y^k) \in PO(G,d_1,|C|)$ (com $k = 1,2,\ldots,|V|+|E|$), os quais satisfazem $\alpha x_i + \pi y = \alpha x_j + \pi_0$ (ou seja, a desigualdade válida para $PO(G,d_1,|C|)$ na igualdade). Para $k = 1,2,\ldots,|V|+|E|$, mapeie o vetor x^k em $\hat{x}^k = (x^k_1, x^k_2 - x^k_1, x^k_3 - x^k_2, x^k_4 - x^k_3, \ldots, x^k_{|V|} - x^k_{|V|-1})$. Os vetores $\hat{x}^k$ (com $k = 1,2,\ldots,|V|+|E|$) são afim independentes, uma vez que o mapeamento de x^k em $\hat{x}^k$ é uma transformação linear com uma matriz não singular (ou seja, invertível). Para $k = 1,2,\ldots,|V|+|E|$, remova a primeira componente de $\hat{x}^k$, de modo a obter um vetor $\hat{x}^{k*} = (x^k_{21}, x^k_{32}, x^k_{43}, \ldots, x^k_{n,n-1})$. Como se tem pontos $(\hat{x}^{k*}, y^k)$ para todo

$k = 1,2,\ldots,|V|+|E|$, há $|V|+|E|$ vetores e, por consequência, existem $|V|+|E|-1$ pontos afim independentes. Sendo assim, defina, para $k = 1,2,\ldots,|V|+|E|-1$, o vetor q^k, no qual se tem $q^k_{t,t-1} = \hat{x}^{k*}_{t,t-1}$ para $t = 2,\ldots,|V|$ e o restante das variáveis com valores determinados pelas restrições de igualdade (5.18) e (5.19). Tem-se então pontos (q^k, y^k) para todo $k = 1,2,\ldots,|V|+|E|-1$, ou seja, tem-se $|V|+|E|-1$ pontos afim independentes que satisfazem $\alpha q_{ij} + \pi y = \pi_0$ (ou seja, a desigualdade válida para $PD(G,C)$ na igualdade). □

O próximo teorema envolve a afirmação reversa, isto é, que as desigualdades válidas para o modelo baseado em distâncias podem ser usadas na formulação de orientação.

Teorema 5.9. *Considere que* $C = \{1,\ldots,|V|\}$. *Seja* $\gamma q + \pi y \leq \pi_0$ *uma desigualdade válida (respectivamente, indutora de faceta) para* $PD(G,C)$. *Então,* $\sum_{i\neq j} \gamma_{ij}(x_i - x_j) + \pi y \leq \pi_0$ *é válida (respectivamente, indutora de faceta se* $|C| \geq \chi(G)+2$*) para* $PO(G,1,C\cup\{|C|+1\})$.

Prova. A validade da desigualdade para $PD(G,C)$ segue diretamente da definição da variável q_{ij}, uma vez que:

$$\gamma q + \pi y \leq \pi_0 \quad \Rightarrow \quad \sum_{i\neq j} \gamma_{ij} q_{ij} + \pi y \leq \pi_0 \quad \Rightarrow \quad \sum_{i\neq j} \gamma_{ij}(x_i - x_j) + \pi y \leq \pi_0$$

Para verificar a indução de faceta, considere $|V|+|E|-1$ pontos afim independentes $(q^k, y^k) \in PD(G,C)$ (com $k = 1,2,\ldots,|V|+|E|$), os quais satisfazem $\gamma x + \pi y = \pi_0$. Para $k = 1,2,\ldots,|V|+|E|-1$, seja $c^k : V \to C$ uma atribuição de cores compatível com (q^k, y^k) (que pode não ser única se $\max_{i\neq j} |q^k_{ij}| < |C|-1$, uma vez que as variáveis q armazenam apenas as diferenças entre cores, mas não as cores em si). Construa o ponto $(x^k, y^k) \in PO(G, d_1, |C|)$ correspondente a c^k (ou seja, $x^k_i = c^k(i)$). Tal ponto também pertence a $PO(G, d_1, |C|+1)$, já que $PO(G, d_1, |C|) \subseteq PO(G, d_1, |D|)$ if $C \subseteq D$. O mapeamento $(x^k, y^k) \to (q^k, y^k)$ é uma transformação linear $L(x,y) = A \times (x,y)$ para uma matriz A não quadrada. Considere os pontos (x^1, y^1) e $(\bar{x}^1, y^1)$, onde $\bar{x}^1_i = x^1_i + 1$ para todo $i \in V$. Ambos os pontos pertencem a $PO(G, C\cup\{|C|+1\})$ e são distintos, logo, são afim independentes. Para $k = 2,\ldots,|V|+|E|-1$, o ponto (x^k, y^k) é afim independente em relação aos pontos $\{(\bar{x}^1, y^1)\} \cup \{(x^1, y^1),\ldots,(x^{k-1}, y^{k-1})\}$, já que, caso contrário, o ponto $L(x^k, y^k)$ seria afim independente em relação a $\{L(\bar{x}^1, y^1)\} \cup \{L(x^1, y^1),\ldots,L(x^{k-1}, y^{k-1})\}$, o que é

uma contradição. Desse modo, tem-se $|V|+|E|$ pontos $\{(\bar{x}^1, y^1)\} \cup \{(x^1, y^1), (x^2, y^2), \ldots, (x^{|V|+|E|-1}, y^{|V|+|E|-1})\}$ que satisfazem $\sum_{i \neq j} \gamma_{ij}(x_i - x_j) + \pi y = \pi_0$ (ou seja, a desigualdade válida para $PO(G, 1, C \cup \{|C|+1\})$ na igualdade). □

Partindo-se dos resultados acima, pode-se reescrever as desigualdades utilizadas na Seção 5.2.1 com o modelo baseado em orientação, de modo a torná-las compatíveis com a formulação de distâncias, como feito a seguir.

5.2.1.2 Desigualdade clique

Considere um vértice $i \in V$ e uma clique $K \subseteq N(i)$. Com base na Definição 5.1.1 da desigualdade clique para o modelo baseado em orientação para o BCP, tem-se que, para o VCP, a expressão da desigualdade passa a ser:

$$x_i \geq \sum_{j \in K} y_{ji} + 1$$

Para utilizar a desigualdade no modelo baseado em distâncias, deve-se adicionar um termo x_j (onde j é algum outro vértice diferente de i) para o lado direito da desigualdade, de modo a obter-se uma expressão $x_i - x_j$ a ser substituída por q_{ij}. Porém, deve-se adicionar um termo suficientemente grande também ao lado esquerdo para compensar tal adição. Desse modo, adiciona-se um termo $\Delta(G)+1$ do lado esquerdo, já que tal valor é um limite superior válido para a maior cor usada no VCP (Welsh e Powell, 1967). Tem-se então, com isso, a definição a seguir.

Definição 5.2.1. *Seja $i \in V$ um vértice do grafo e $K \subseteq N(i)$ uma clique do mesmo. Desse modo, a expressão:*

$$q_{ij} + \Delta(G) \geq \sum_{j \in K} y_{ji} \tag{5.37}$$

define a desigualdade clique associada ao vértice i e à clique K para $PD(G, C)$.

5.2.1.3 Desigualdade clique dupla

Seja $(i, j) \in E$ e considere uma clique $K \subseteq N(i) \cap N(j)$. Baseando-se na Definição 5.1.2, tem-se a seguinte expressão para a desigualdade clique dupla para modelo baseado em

orientação para o VCP:

$$x_i + 1 + \sum_{k \in K} (y_{ik} - y_{jk}) \leq x_j + (|C| - |K|) y_{ji}$$

A definição da desigualdade adaptada para o modelo baseado em distâncias segue diretamente pela substituição de variáveis apresentada nos Teoremas 5.8 e 5.9, como mostrado a seguir.

Definição 5.2.2. *Seja $(i, j) \in E$ uma aresta de um grafo G e considere uma clique $K \subseteq N(i) \cap N(j)$. Define-se a expressão*

$$x_i + 1 + \sum_{k \in K} (y_{ik} - y_{jk}) \ \leq \ x_j + (|C| - |K|) y_{ji} \tag{5.38}$$

como a desigualdade clique dupla (double clique) associada à aresta (i, j) e à clique K e ao vértice p para $PD(G,C)$.

5.3 Notas do capítulo

Neste capítulo, foram apresentadas duas novas formulações de programação inteira para problemas de coloração em grafos. Na primeira, que foi proposta para o BCP, as apresentada uma nova formulação de programação inteira para o BCP, na qual as soluções induzem a orientações do grafo. A formulação tem tamanho linear, sendo que a quantidade de variáveis e restrições não depende de limites superiores ou dos valores das distâncias do grafo de entrada. Para a nova formulação, também foram apresentadas desigualdades válidas, que induzem a facetas sob determinadas condições. Baseando-se nesses resultados, foi implementado um algoritmo *cut-and-branch* para o BCP, onde são geradas desigualdades para o modelo e, em seguida, o mesmo é resolvido por um método tradicional de programação inteira Os experimentos realizados mostram que tais desigualdades são promissoras, permitindo que o *cut-and-branch* obtenha soluções mais rapidamente em algumas instâncias utilizadas.

O segundo modelo, baseado em distâncias, para a coloração clássica em grafos (VCP), é derivado do anterior e usa variáveis que correspondem às arestas do grafo de entrada, seguindo o conceito de 0-esferas utilizado na modelagem das colorações com distâncias (como visto no Capítulo 3). Mostrou-se que as desigualdades válidas determinadas para

a formulação de orientação também podem ser usadas no novo modelo baseado em distâncias, e que existe uma equivalência entre facetas de ambos os modelos. Sendo assim, foram apresentadas as desigualdades para a formulação de orientação de modo adaptado para o novo modelo.

Melhorias futuras podem ser realizadas no modelos e no *cut-and-branch* usado. Novas desigualdades válidas indutoras de facetas podem ser determinadas seguindo características usadas em outros problemas de coloração (como feito por Méndez-Díaz e Zabala (2006) e Marenco e Wagler (2009)), além do uso de algoritmos de separação de cortes específicos e heurísticas para determinação de melhores limites. Para o modelo baseado em distâncias, outra vertente importante para é a determinação de uma função objetivo para guiar a busca no poliedro, bem como adaptações para que o modelo possa também ser usado no BCP.

Capítulo 6

Considerações finais

Esta tese abordou problemas de colorações em grafos com restrições adicionais de distâncias impostas nas arestas, envolvendo conceitos de teoria dos grafos e geometria de distâncias, cuja principal aplicação ocorre na alocação de canais em redes sem fio.

Para a realização do trabalho, foi feita uma revisão da literatura sobre colorações em grafos, em especial para o problema de coloração clássica (VCP) e em largura de banda (BCP), uma vez que o primeiro problema é o principal da classe de colorações em grafos e o segundo o mais usado no contexto de alocação de canais. Para cada um dos problemas, foram apresentados limites inferiores e superiores, bem como modelos de programação inteira e métodos exatos e aproximados. Tal revisão permitiu a identificação de diversas características que podem ser exploradas nos problemas de colorações com distâncias.

Seguindo tal estudo, novas variações de colorações com distâncias foram caracterizadas, com diferentes tipos de restrições de adjacência envolvendo desigualdades e igualdades. Para estas colorações, foram identificadas propriedades que indicam alguns tipos de grafos para os quais não há solução factível, de modo que, se uma instância contiver um subgrafo induzido de um dos tipos identificados, pode-se dizer que tal instância não terá solução, sem necessidade de executar métodos exatos.

Além disso, um modelo de programação inteira existente foi melhorado e um de programação por restrições foi proposto para a coloração em largura de banda, sendo ambos implementados computacionalmente, com a finalidade de identificar soluções ótimas. Foi possível obter um limite superior melhor para uma das instâncias utilizadas, bem como identificar inconsistências em resultados da literatura. Adicionalmente, foi realizada uma comparação entre o comportamento das formulações de programação inteira e por restri-

ções de acordo com a demanda de cores da instância, de modo que a programação por restrições se mostrou melhor com demandas unitárias e a programação inteira foi melhor com multicoloração.

Outra contribuição foi uma nova formulação de programação inteira para o BCP, onde uma solução define uma orientação. A nova formulação tem tamanho linear, ao contrário das já existentes que são pseudopolinomiais, e tem dimensão plena de acordo com um parâmetro de modelagem. Para o novo modelo, foram apresentadas duas famílias de desigualdades válidas, baseadas em cliques e cliques duplas, as quais são indutoras de facetas sob determinadas condições. Um método *cut-and-branch* foi implementado de modo a aplicar as desigualdades no modelo e permitir testes empíricos. A nova formulação apresentou bom desempenho devido ao seu menor tamanho, e as desigualdades válidas permitiram, para algumas instâncias, encontrar soluções mais rapidamente que o modelo puro, mostrando que há potencial para utilização delas.

Por fim, uma contribuição adicional da pesquisa foi o modelo de programação inteira baseado em distâncias para o VCP, derivado da formulação de orientação. As variáveis do mesmo, ao invés de indicarem cores, passam a indicar diferenças entre cores de pares de vértices, seguindo a modelagem em 0-esferas das colorações com distâncias. Para esta formulação, foi verificado que há equivalência entre as desigualdades válidas da mesma e as do modelo baseado em orientação, inclusive com respeito à definição de facetas, sendo fornecidas as desigualdades clique e clique dupla de modo adaptado para a formulação baseada em distâncias.

Para ambas as hipóteses que baseiam esta pesquisa, foi possível obter respostas positivas para as mesmas, de modo que variações representativas de coloração foram geradas com diferentes tipos de restrições e pesos nas arestas, bem como modelos teóricos representativos foram caracterizados, embasando métodos de resolução eficientes.

Apesar do contexto teórico deste trabalho, espera-se também que as contribuições fornecidas na mesma tenham impacto em aplicações de colorações em grafos, em especial no contexto de alocação de canais, que foi o principal problema motivador das abordagens utilizadas. Com os resultados apresentados, torna-se possível modelar diferentes cenários de comunicações, o que pode permitir um planejamento mais prático das redes móveis sem fio, além de melhorar estratégias de alocação já existentes e utilizadas em cenários reais. Diversas publicações em periódicos e apresentações em eventos nacionais e inter-

nacionais foram realizadas com os resultados obtidos, que podem ser vistas no Apêndice A.

A pesquisa envolvendo os problemas de colorações com distâncias continua. Trabalhos já iniciados incluem a identificação de famílias de desigualdades válidas para os novos modelos de programação inteira (baseados em orientação e distâncias) e melhorias para a formulação baseada em distâncias, incluindo a determinação de uma função objetivo que permita o uso da mesma em algoritmos de otimização e a generalização do modelo para o BCP.

Diversas vertentes desta pesquisa podem ser exploradas em trabalhos futuros além do que já está em desenvolvimento. Para as variações de colorações com distâncias, características adicionais podem ser aplicadas, como variações nas listas de distâncias permitidas, bem como diferentes tipos de restrição de adjacência, incluindo intervalos. Apesar de a modelagem de programação por restrições se mostrar eficaz, pode ser possível aplicar outros tipos de restrições globais ao modelo, de modo a efetuar podas mais fortes dos domínios das variáveis. Para o método *cut-and-branch* baseado na formulação de orientação, algoritmos específicos de separação de cortes que não enumerem todas as desigualdades podem ser aplicados de acordo com as estruturas do grafo, o que permite determinar cortes dominantes durante o processo de obtenção de novas soluções, reduzindo o tempo de solução das relaxações e transformando-o num método *branch-and-cut*.

Referências

Aardal, K., van Hoesel, S., Koster, A., Mannino, C., e Sassano, A. Models and solution techniques for frequency assignment problems. *Annals of Operations Research*, 153: 79–129, 2007.

Aardal, K. I., Hipolito, A., van Hoesel, C. P. M., e Jansen, B. A branch-and-cut algorithm for the frequency assignment problem. *Research Memorandum 96/011*, 1996.

Abbasian, R. e Mouhoub, M. A hierarchical parallel genetic approach for the graph coloring problem. *Applied Intelligence*, 39(3):510–528, 2013. ISSN 0924669X. doi: 10.1007/s10489-013-0429-5.

Aldous, D. The random walk construction of uniform spanning trees and uniform labelled trees. *SIAM Journal on Discrete Mathematics*, 3(4):450–465, 1990.

Appel, K. e Haken, W. Every planar map is four colorable. Part I: Discharging. *Illinois Journal of Mathematics*, 21(3):429–490, 09 1977.

Appel, K., Haken, W., e Koch, J. Every planar map is four colorable. Part II: Reducibility. *Illinois Journal of Mathematics*, 21(3):491–567, 09 1977.

Audhya, G., Sinha, K., Ghosh, S., e Sinha, B. A survey on the channel assignment problem in wireless networks. *Wireless Communications and Mobile Computing*, 11: 583–609, 2011.

Avanthay, C., Hertz, A., e Zufferey, N. A variable neighborhood search for graph coloring. In *European Journal of Operational Research*, volume 151, pages 379–388, 2003. doi: 10.1016/S0377-2217(02)00832-9.

Baybars, I. Optimal assignment of broadcasting frequencies. *European Journal of Operational Research*, 9:257–263, 1982.

Beckmann, D. e Killat, U. A new strategy for the application of genetic algorithms to the channel-assignment problem. *Vehicular Technology, IEEE Transactions on*, 48(4): 1261–1269, 1999. ISSN 0018-9545. doi: 10.1109/25.775374.

Beeker, N., Gaubert, S., Glusa, C., , e Liberti, L. Is the Distance Geometry Problem in NP? In Mucherino, A., Lavor, C., Liberti, L., e Maculan, N., editors, *Distance Geometry*, pages 85–93. Springer, 2013.

Beldiceanu, N. e Beldiceanu, N. Global Constraints Catalog. `http://www.emn.fr/z-info/sdemasse/gccat/`, 2014.

Blöchliger, I. e Zufferey, N. A graph coloring heuristic using partial solutions and a reactive tabu scheme. *Computers and Operations Research*, 35(3):960–975, 2008. ISSN 03050548. doi: 10.1016/j.cor.2006.05.014.

Bockmayr, A. Integer Programming, Constraint Programming, and their Combination. `http://www.win.tue.nl/diamant/ipday060127/bockmayr.pdf`, 2006.

Bondy, J. e Murty, U. *Graph Theory with Applications*. Elsevier Science Publishing, 1982.

Borndörfer, R., Eisenblätter, A., Grötschel, M., e Martin, A. The orientation model for frequency assignment problems. Technical report, ZIB Berlin, 1998.

Boyd, S. e Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

Bradley, S. P., Hax, A. C., e Magnanti, T. L. *Applied Mathematical Programming*. Addison-Wesley, 1977. Reeditado digitalmente e disponivel em `http://web.mit.edu/15.053/www/`.

Brown, J. R. Chromatic scheduling and chromatic number problems. *Management Science*, 19(4):456–463, 1972.

Brélaz, D. New Methods to Color the Vertices of a Graph. *Communications of the ACM*, 22(4):251–256, 1979.

Burke, E., Mareček, J., Parkes, A., e Rudová, H. A supernodal formulation of vertex colouring with applications in course timetabling. *Annals of Operations Research*, 179:105–130, 2010.

Campêlo, M., Campos, V., e Corrêa, R. On the asymmetric representatives formulation for the vertex coloring problem. *Discrete Applied Mathematics*, 156:1097–1111, 2008.

Campêlo, M., Campos, V., e Corrêa, R. Um algoritmo de planos-de-corte para o número cromático fracionário de um grafo. *Pesquisa Operacional*, 29(1):179–193, 2009.

Caramia, M. e Dell'Olmo, P. A Fast and Simple Local Search for Graph Coloring. *Proceedings of the 3rd International Workshop on Algorithm Engineering*, pages 316–329, 1999. URL `http://dl.acm.org/citation.cfm?id=647256.720628`.

Caramia, M., Dell'Olmo, P., e Italiano, G. F. CHECKCOL: Improved local search for graph coloring. *Journal of Discrete Algorithms*, 4(2):277–298, 2006. ISSN 15708667. doi: 10.1016/j.jda.2005.03.006.

Chaitin, G. Register Allocation & Spilling via Graph Coloring. In *Proceedings of the 1982 SIGPLAN Symposium on Compiler Construction*, SIGPLAN '82, pages 98–105. ACM, 1982.

Chakraborty, G. An Efficient Heuristic Algorithm for Channel Assignment Problem in Cellular Radio Networks. *IEEE Transactions on Vehicular Technology*, 50(6):1528–1539, 2001.

Chalupa, D. Population-based and learning-based metaheuristic algorithms for the graph coloring problem. *Proceedings of the 13th annual conference on Genetic and evolutionary computation - GECCO '11*, page 465, 2011. doi: 10.1145/2001576.2001641.

Chams, M., Hertz, A., e de Werra, D. Some experiments with simulated annealing for coloring graphs. *European Journal of Operational Research*, 32(2):260–266, 1987. ISSN 03772217. doi: 10.1016/S0377-2217(87)80148-0.

Chartrand, G. e Zhang, P. *Chromatic Graph Theory*. Chapman & Hall/CRC, 2009.

Chaves-González, J. M., Vega-Rodríguez, M. A., Domínguez-González, D., e Juan A. Gómez-Pulido, J. M. S.-P. Population-Based Incremental Learning to Solve the FAP Problem. In *Proceedings of the 2th International Conference on Advanced Engineering Computing and Applications in Sciences*, 2008.

Chen, R.-H., Yu, C.-W., e Wu, T.-K. A Novel Approach to the Fixed Channel Assignment Problem. *Journal of Information Science and Engineering*, 58:39–58, 2005.

Chiarandini, M. e Stützle, T. Stochastic local search algorithms for graph set T-colouring and frequency assignment. *Constraints*, 12(3):371–403, 2007. ISSN 13837133. doi: 10.1007/s10601-007-9023-y.

Costa, D. e Hertz, A. Ants can colour graphs. *Journal of the Operational Research Society*, 48:295–305, 1997.

Datorro, J. *Convex Optimization and Euclidean Distance Geometry*. Meboo Publishing, 2013. Disponível em `https://ccrma.stanford.edu/~dattorro/mybook.html`.

de Freitas, R., Dourado, M., e Szwarcfiter, J. Graph coloring and scheduling problems. 4th Latin American Workshop on Cliques in Graphs, 2010.

de Freitas, R., Dias, B., Maculan, N., e Szwarcfiter, J. On feasibility conditions in graph coloring problems with distance constraints. In *Many Faces of Distances (Workshop)*, 2014.

de Freitas, R., Dias, B., Maculan, N., e Szwarcfiter, J. Distance geometry approach for special graph coloring problems. *ArXiv e-prints*, June 2016. URL `https://arxiv.org/abs/1606.04978`.

de Freitas, R., Dias, B., Maculan, N., e Szwarcfiter, J. On distance graph coloring problems. *International Transactions in Operational Research*, 2019. doi: 10.1111/itor.12626.

Delle-Donne, D. *Estudios poliedrales de problemas de coloreo de grafos*. PhD thesis, Universidad de Buenos Aires, Argentina, 2016. In spanish.

Dias, B., de Freitas, R., e Maculan, N. Alocação de canais em redes celulares sem fio: algoritmos e modelos teóricos em grafos e escalonamento. In *Proceedings of the XVI Latin-Ibero-American Conference on Operations Research / XLIV Brazilian Symposium of Operations Research*. Brazilian Society of Operations Research (SOBRAPO), 2012. In portuguese.

Dias, B., de Freitas, R., e Maculan, N. *Simulated annealing* para a alocação de canais em redes móveis celulares. In *Anais do XLV Simpósio Brasileiro de Pesquisa Operacional*. Sociedade Brasileira de Pesquisa Operacional (SOBRAPO), 2013a.

Dias, B., de Freitas, R., e Szwarcfiter, J. On graph coloring problems with distance constraints. In *Proceedings of I Workshop on Distance Geometry and Applications (DGA 2013)*, 2013b.

Dias, B., de Freitas, R., Maculan, N., e Szwarcfiter, J. Some Challenges in n-dimensional Euclidean Distance Geometry Problems. In *INFORMS Annual Meeting 2014*, 2014a.

Dias, B., de Freitas, R., Maculan, N., e Szwarcfiter, J. Distance coloring problems, spatial properties and feasibility conditions. In *VI Latin American Workshop on Cliques in Graphs (LAWCG 2014)*, 2014b.

Dias, B., de Freitas, R., Santos, C., Lavor, C., Maculan, N., e Szwarcfiter, J. Some notes on Euclidean distance geometry and graph theory involving telecom, computer networks and molecular biology applications. In *20th Conference of the International Federation of Operational Research Societies (IFORS 2014)*, 2014c.

Dias, B., de Freitas, R., Maculan, N., Szwarcfiter, J., e Michelon, P. Channel assignment as a distance geometry graph coloring problem. In *27th European Conference on Operational Research (EURO 2015)*, 2015.

Dias, B., de Freitas, R., e Maculan, N. On Special Vertex Coloring Models in Graphs Using Distance Geometry. In *XI Brazilian Workshop on Continuous Optimization (BRAZOPT 2016)*, 2016a.

Dias, B., de Freitas, R., Maculan, N., e Michelon, P. Constraint and integer programming models for bandwidth coloring and multicoloring in graphs. In *XLVIII Simpósio Brasileiro de Pesquisa Operacional (SBPO 2016)*, 2016b.

Dias, B., de Freitas, R., Marenco, J., e Maculan, N. Politopo baseado em distâncias para o problema clássico de coloração de vértices em grafos. In *III Encontro de Teoria da Computação (ETC 2018)*, 2016c.

Dias, B., de Freitas, R., Maculan, N., Szwarcfiter, J., e Michelon, P. Sobre modelos e algoritmos para coloração em grafos com restrições de distância. In *XLIX Simpósio Brasileiro de Pesquisa Operacional (SBPO 2017)*, 2017a.

Dias, B., de Freitas, R., Marenco, J., e Maculan, N. Facet-inducing inequalities and a cut-and-branch algorithm for the bandwidth coloring polytope based on orientation model. In *IX Latin and American Algorithms, Graphs and Optimization Symposium (LAGOS 2017)*, 2017b.

Dias, B., de Freitas, R., Marenco, J., Maculan, N., Szwarcfiter, J., e Michelon, P. Advances in solving graph coloring problems. In *21th Conference of the International Federation of Operational Research Societies (IFORS 2017)*, 2017c.

Dias, B., de Freitas, R., Maculan, N., Szwarcfiter, J., e Michelon, P. Notes on advances in theoretical modeling and resolution of graph coloring with distance constraints. In *Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization (ALIO/EURO 2018)*, 2018a.

Dias, B., de Freitas, R., Marenco, J., e Maculan, N. The distance polytope for vertex coloring. In *V International Symposium on Combinatorial Optimization (ISCO 2018)*, 2018b.

Dias, B., de Freitas, R., Marenco, J., Maculan, N., Szwarcfiter, J., e Michelon, P. Sobre o dgp, realizações de grafos em $\mathbb{R}^n$ e colorações. In *XXXVIII Congresso Nacional de Matemática Aplicada e Computacional (CNMAC 2018)*, 2018c.

Dias, B., de Freitas, R., Marenco, J., Maculan, N., Szwarcfiter, J., e Michelon, P. On distance colorings, graph embedding and ip/cp models. In *VIII Latin American Workshop on Cliques in Graphs (LAWCG 2018)*, 2018d.

Dias, B., de Freitas, R., e Maculan, N. Notes on the Bandwidth Coloring Problem and IP / CP Models. In *2019 INFORMS International Conference*, 2019a.

Dias, B. Modelos teóricos e algoritmos para a otimização da alocação de canais em redes móveis sem fio. Master's thesis, Instituto de Computação - Universidade Federal do Amazonas, Brasil, 2014.

Dias, B., de Freitas, R., Maculan, N., e Michelon, P. Solving the bandwidth coloring problem applying constraint and integer programming techniques. *Optimization Online (e-print)*, 2016d. URL http://www.optimization-online.org/DB_HTML/2016/06/5514.html.

Dias, B., de Freitas, R., Maculan, N., e Marenco, J. Facet-inducing inequalities and a cut-and-branch for the bandwidth coloring polytope based on the orientation model. *Electronic Notes in Discrete Mathematics*, 62:141 – 146, 2017d. ISSN 1571-0653. doi: https://doi.org/10.1016/j.endm.2017.10.025. URL https://www.sciencedirect.com/science/article/pii/S1571065317302640. LAGOS'17 – {IX} Latin and American Algorithms, Graphs and Optimization.

Dias, B., de Freitas, R., Maculan, N., e Marenco, J. The Distance Polytope for the Vertex Coloring Problem. In Lee, J., Rinaldi, G., e Mahjoub, A. R., editors, *Combinatorial Optimization*, pages 144–156. Springer International Publishing, 2018e.

Dias, B., de Freitas, R., Marenco, J., Maculan, N., e Szwarcfiter, J. Notes on models for distance coloring problems. *Matemática Contemporânea*, 2019b. A ser publicado.

Dong, Q. e Wu, Z. A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *Journal of Global Optimization*, 22(1–4): 365–375, 2002.

Dorne, R. e Hao, J. Tabu search for graph coloring, t-coloring and set t-colorings. In Voss, S., Martello, S., Osman, I., e Roucairol, C., editors, *Metaheuristics: advances and trends in local search paradigms for optimization*, pages 77–92. Kluwer Academic Publishers, 1998a.

Dorne, R. e Hao, J.-k. A new genetic local search algorithm for graph coloring. In *Parallel Problem Solving from Nature - PPSN V*, pages 745–754, 1998b.

Dowsland, K. A. e Thompson, J. M. An improved ant colony optimisation heuristic for graph colouring. *Discrete Applied Mathematics*, 156(3):313–324, 2008. ISSN 0166218X. doi: 10.1016/j.dam.2007.03.025.

Eppstein, D., Löffler, M., e Strash, D. Listing All Maximal Cliques in Sparse Graphs in

Near-Optimal Time. In *Algorithms and Computation: 21st International Symposium, ISAAC 2010*, pages 403–414, 2010.

Ferreira, C. E. e Wakabayashi, Y. *Combinatória Poliédrica e Planos-de-Corte Faciais*. Departamento de Ciência da Computação - IME/USP, 1996.

Fijuljanin, J. Two genetic algorithms for the bandwidth multicoloring problem. *Yugoslav Journal of Operations Research*, 22(2):225–246, 2012. ISSN 03540243. doi: 10.2298/YJOR100927020F.

Fister, I., Brest, J., e Ieee. Using Differential Evolution for the Graph Coloring. *2011 IEEE Symposium on Differential Evolution*, pages 143–149, 2011.

Fleurent, C. e Ferland, J. A. Genetic and hybrid algorithms for graph coloring. *Annals of Operations Research*, 63:437–461, 1996. ISSN 0254-5330. doi: 10.1007/BF02125407.

Furini, F., Gabrel, V., e Ternier, I.-C. Lower Bounding Techniques for DSATUR-based Branch and Bound. *Electronic Notes in Discrete Mathematics*, 52:149–156, 2016.

Galinier, P. e Hao, J.-k. Hybrid Evolutionary Algorithms for Graph Coloring. *Journal of Combinatorial Optimization*, 3:379–397, 1999. ISSN 13826905. doi: 10.1023/A:1009823419804.

Garey, M. e Johnson, D. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

Giortzis, A. I. e Turner, L. F. Application of mathematical programming to the fixed channel assignment problem in mobile radio networks. *IEE Proceedings - Communications*, 144(4):257–264, 1997.

Goldbarg, M. C. e Luna, H. P. L. *Otimização Combinatória e Programação Linear*. Elsevier, 2^a edition, 2005.

Gomes, F., Pardalos, P., Oliveira, C., e Resende, M. Reactive GRASP with path relinking for channel assignment in mobile phone networks. *Proceedings of the 5th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 60–67, 2001. doi: 10.1145/381448.381456. URL `http://portal.acm.org/citation.cfm?id=381448.381456`.

Gonçalves, D. S. e Mucherino, A. Optimal partial discretization orders for discretizable distance geometry. *International Transactions in Operational Research*, 23:947–967, 2016.

Hale, W. Frequency assignment: Theory and applications. *Proceedings of the IEEE*, 25: 1497–1514, 1980.

Hamiez, J.-p. e Hao, J.-k. Scatter search for graph coloring. In *Artificial Evolution*, pages 168–179, 2002.

Hao, J.-k., Dorne, R., e Galinier, P. Tabu Search for Frequency Assignment in Mobile Radio Networks. *journal of Heuristics*, 4:47–62, 1998. ISSN 13811231. doi: 10.1023/A:1009690321348.

Hertz, A. e Werra, D. Using tabu search techniques for graph coloring. *Computing*, 39(4):345–351, 1987. ISSN 0010-485X. doi: 10.1007/BF02239976. URL `http://www.springerlink.com/index/Y766J232466L1674.pdf`.

Hertz, A., Plumettaz, M., e Zufferey, N. Variable space search for graph coloring. *Discrete Applied Mathematics*, 156(13):2551–2560, 2008. ISSN 0166218X. doi: 10.1016/j.dam.2008.03.022.

Hsu, L.-y., Horng, S.-j., Fan, P., Khan, M. K., Wang, Y.-r., Run, R.-s., Lai, J.-l., e Chen, R.-j. MTPSO algorithm for solving planar graph coloring problem. *Expert Systems With Applications*, 38(5):5525–5531, 2011. ISSN 0957-4174. doi: 10.1016/j.eswa.2010.10.084. URL `http://dx.doi.org/10.1016/j.eswa.2010.10.084`.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., e Schevon, C. Optimization by Simulated Annealing: An Experimental Evaluation; Part II, Graph Coloring and Number Partitioning. *Operations Research*, 39(3):378–406, 1991. ISSN 0030-364X. doi: 10.1287/opre.39.3.378.

Karp, R. Reducibility Among Combinatorial Problems. In Miller, R. E. e Thatcher, J. W., editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

Kendall, G. e Mohamad, M. Solving the Fixed Channel Assignment Problem in Cellular Communications Using an Adaptive Local Search. In Burke, E. e Trick, M., editors,

5th International Conference for the Practice and Theory of Automated Timetabling (PATAT 2004), Lecture Notes in Computer Science, vol. 3616. Springer, Heidelberg, 2005.

Kim, S. S., Smith, A. E., e Lee, J. H. A memetic algorithm for channel assignment in wireless FDMA systems. *Computers and Operations Research*, 34(6 SPEC. ISS.): 1842–1856, 2007. ISSN 03050548. doi: 10.1016/j.cor.2005.05.038.

Koster, A. *Frequency assignment: models and algorithms*. PhD thesis, Universiteit Maastricht, the Netherlands, 1999.

Laguna, M. e Martí, R. A GRASP for coloring sparse graphs. *Computational Optimization and Applications*, 19(2):165–178, 2001. ISSN 09266003. doi: 10.1023/A:1011237503342.

Lai, X. e Lü, Z. Multistart Iterated Tabu Search for Bandwidth Coloring Problem. *Computers & Operations Research*, 40:1401–1409, 2013.

Lai, X., Hao, J. K., Lü, Z., e Glover, F. A learning-based path relinking algorithm for the bandwidth coloring problem. *Engineering Applications of Artificial Intelligence*, 52:81–91, 2016. ISSN 09521976. doi: 10.1016/j.engappai.2016.02.008. URL `http://dx.doi.org/10.1016/j.engappai.2016.02.008`.

Lau, T. L. e Tsang, E. P. K. Solving the Radio Link Frequency Assignment Problem using Guided Local Search. In *NATO Symposium on Radio Length Frequency Assignment*, 1998.

Lavor, C., Liberti, L., Maculan, N., e Mucherino, A. The discretizable molecular distance geometry problem. *Computational Optimization and Applications*, 52(1):115–146, 2012.

Lawler, E. *Combinatorial Optimization: networks and matroids*. Dover Publications, 1976.

Leighton, F. T. A Graph Coloring Algorithm for Large Scheduling Problems. *Journal of Research of the National Bureau of Standards*, 84(6):489–506, 1979.

Lewis, R. M. R. *A Guide to Graph Colouring: Algorithms and Applications*. Springer, 2016.

Liberti, L., Lavor, C., e Maculan, N. A branch-and-prune algorithm for the molecular distance geometry problem. *International Transactions in Operational Research*, 15 (1):1–17, 2008.

Liberti, L., Lavor, C., Maculan, N., e Mucherino, A. Euclidean Distance Geometry and Applications. *SIAM Review*, 56(1):3–69, 2014.

Lim, A., Zhu, Y., Lou, Q., Rodrigues, B., Algorithms, F. N., e Se, P. Heuristic Methods for Graph Coloring Problems. In *2005 ACM Symposium in Applied Computing*, pages 933–939, 2005. ISBN 1581139640. doi: 10.1145/1066677.1066892.

Lü, Z. e Hao, J.-k. A memetic algorithm for graph coloring. *European Journal of Operational Research*, 203(1):241–250, 2010. ISSN 0377-2217. doi: 10.1016/j.ejor.2009.07.016. URL `http://dx.doi.org/10.1016/j.ejor.2009.07.016`.

Lucet, C., Mendes, F., e Moukrim, A. An exact method for graph coloring. *Computers & Operations Research*, 33:2189–2207, 2006.

Luke, S. *Essentials of Metaheuristics*. Lulu, 2009. Disponível em `http://cs.gmu.edu/~sean/book/metaheuristics/`.

Mabrouk, B. B., Hasni, H., e Mahjoub, Z. On a parallel genetic-tabu search based algorithm for solving the graph colouring problem. *European Journal of Operational Research*, 197(3):1192–1201, 2009. ISSN 0377-2217. doi: 10.1016/j.ejor.2008.03.050. URL `http://dx.doi.org/10.1016/j.ejor.2008.03.050`.

Maculan, N. e Fampa, M. *Otimização Linear*. Editora da Universidade de Brasília (UnB), 2006.

Malaguti, E. e Toth, P. A survey on vertex coloring problems. *International Transactions in Operational Research*, 17(1):1–34, 2010.

Malaguti, E. e Toth, P. An evolutionary approach for bandwidth multicoloring problems. *European Journal of Operational Research*, 189(3):638–651, 2008. ISSN 03772217. doi: 10.1016/j.ejor.2006.09.095.

Malaguti, E., Monaci, M., e Toth, P. A Metaheuristic Approach for the Vertex Coloring Problem. *INFORMS Journal on Computing*, 20(2):302–316, 2008. ISSN 1091-9856. doi: 10.1287/ijoc.1070.0245. URL http://joc.journal.informs.org/cgi/doi/10.1287/ijoc.1070.0245.

Mannino, C. e Sassano, A. An enumerative algorithm for the frequency assignment problem. *Discrete Applied Mathematics*, 129:155–169, 2003.

Marenco, J. *Chromatic Scheduling Polytopes Coming from the Bandwidth Allocation Problem in Point-to-Multipoint Radio Access Systems*. PhD thesis, Universidad de Buenos Aires, Argentina, 2005. Em espanhol.

Marenco, J. e Wagler, A. Facet-inducing inequalities for chromatic scheduling polytopes based on covering cliques. *Discrete Optimization*, 6(1):64 – 78, 2009. ISSN 1572-5286. doi: https://doi.org/10.1016/j.disopt.2008.09.002. URL http://www.sciencedirect.com/science/article/pii/S1572528608000649.

Marti, R., Gortazar, F., e Duarte, A. Heuristics for the bandwidth colouring problem. *International Journal of Metaheuristics*, 1:11, 2010. ISSN 1755-2176. doi: 10.1504/IJMHEUR.2010.033121.

Mehrotra, A. e Trick, M. A. A column generation approach for graph coloring. *INFORMS Journal on Computing*, 8(4):344–354, 1996.

Mehrotra, A. e Trick, M. A. A Branch-And-Price Approach for Graph Multi-Coloring. In Baker, E., Joseph, A., Mehrotra, A., e Trick, M. A., editors, *Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies*, volume 37 of *Operations Research/Computer Science Interfaces Series*, pages 15–29. Springer, 2007.

Moon, J. e Moser, L. On cliques in graphs. *Israel Journal of Mathematics*, 3:23–28, 1965.

Mucherino, A., Lavor, C., e Liberti, L. The discretizable distance geometry problem. *Optimization Letters*, 6(8):1671–1686, 2012.

Mucherino, A., Lavor, C., Liberti, L., e Maculan, N. *Distance Geometry: Theory, Methods and Applications*. Springer, 2013.

Méndez-Díaz, I. e Zabala, P. A branch-and-cut algorithm for graph coloring. *Discrete Applied Mathematics*, 154(5):826–847, 2006.

Pahlavani, A. e Eshghi, K. A hybrid algorithm of simulated annealing and tabu search for graph colouring problem. *Int. J. Oper. Res.*, 11(2):136–159, 2011. ISSN 1745-7645; 1745-7653/e. doi: 10.1504/IJOR.2011.040694.

Papadimitriou, C. e Steiglitz, K. *Combinatorial Optimization: algorithms and complexity*. Dover Publications, 1998.

Paquete, L. e Stutzle, T. An experimental investigation of Iterated Local Search for coloring graphs. *Applications of Evolutionary Computing, Proceedings*, 2279:122–131, 2002. ISSN 03029743. URL http://gateway.isiknowledge.com/gateway/Gateway.cgi?GWVersion=2{&}SrcAuth=Alerting{&}SrcApp=Alerting{&}DestApp=WOS{&}DestLinkType=FullRecord;KeyUT=000181139600013.

Pendavingh, R. *Optimization in* $\mathbb{R}^n$. Technische Universiteit Eindhoven, 2015. Disponível em http://www.win.tue.nl/~rudi/linear_optimization_course.html.

Phan, V. e Skiena, S. Coloring graphs with a general heuristic search engine. In *Computational Symposium on Graph Coloring and Its Generalizations*, pages 92–99, 2002.

Plumettaz, M., Schindl, D., e Zufferey, N. Ant Local Search and its efficient adaptation to graph colouring. *Journal of the Operational Research Society*, 61(5):819–826, 2009. ISSN 0160-5682. doi: 10.1057/jors.2009.27. URL http://dx.doi.org/10.1057/jors.2009.27.

Prestwich, S. Generalised graph colouring by a hybrid of local search and constraint programming. *Discrete Applied Mathematics*, 156(2):148–158, 2008.

Salari, E. e Eshghi, K. An ACO Algorithm for Graph Coloring Problem. *2005 ICSC Congress on Computational Intelligence Methods and Applications*, vol., no., (6):1–5, 2005. ISSN 0095-2338. doi: 10.1109/CIMA.2005.1662331. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1662331.

Sallaume, S., Martins, S. L., Ochi, L. S., Silva, W., Lavor, C., e Liberti, L. A discrete search algorithm for finding the structure of protein backbones and side chains. *International Journal of Bioinformatics Research and Applications*, 9(3):261–270, 2013.

San José-Revuelta, L. A new adaptive genetic algorithm for fixed channel assignment. *Information Sciences*, 177(13):2655–2678, 2007. ISSN 00200255. doi: 10.1016/j.ins.2007.01.003. URL http://www.sciencedirect.com/science/article/pii/S0020025507000205.

Saxe, J. B. Embeddability of weighted graphs in *k*-space is strongly NP-hard. In *Proceedings of 17th Allerton Conference in Communications, Control and Computing*, pages 480–489, 1979.

Schriver, A. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

Segundo, P. S. A new DSATUR-based algorithm for exact vertex coloring. *Computers & Operations Research*, 39:1724–1733, 2012.

Sewell, E. An improved algorithm for exact graph coloring. In Trick, M. e Johnson, D., editors, *Cliques, coloring, and satisfiability. Proceedings of the 2nd DIMACS implementation challenge, vol. 26*, pages 359–373. American Mathematical Society, 1996.

Shirazi, S. e Amindavar, H. Fixed channel assignment using new dynamic programming approach in cellular radio networks. *Computers and Electrical Engineering*, 31(4-5): 303–333, 2005.

Sivarajan, K. N., McEliece, R. . J., e Ketchum, J. W. Channel assignment in cellular radio. In *IEEE 39th Vehicular Technology Conference*, volume 2, pages 846–850, 1989.

Szwarcfiter, J. L. *Grafos e Algoritmos Computacionais*. Elsevier, 1986.

Talbi, E.-G. *Metaheuristics: From Design to Implementation*. John Wiley & Sons, 2009.

Titiloye, O. e Crispin, A. Quantum annealing of the graph coloring problem. *Discrete Optimization*, 8(2):376–384, 2011. ISSN 1572-5286. doi: 10.1016/j.disopt.2010.12.001. URL http://dx.doi.org/10.1016/j.disopt.2010.12.001.

Tomita, E., Tanaka, A., e Takahashi, H. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363: 28–42, 2006.

Trick, M., Mehrotra, A., e Johnson, D. COLOR02/03/04: Graph Coloring and its Generalizations. http://mat.gsia.cmu.edu/COLOR02/, 2002.

Trick, M. Constraint Programming: A Tutorial. http://mat.gsia.cmu.edu/trick/cp.ppt.

Trick, M. A. e Yildiz, H. A Large Neighborhood Search Heuristic for Graph Coloring. In Van Hentenryck, P. e Wolsey, L., editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: 4th International Conference, CPAIOR 2007*, pages 346–360. Springer Berlin Heidelberg, 2007. doi: 10.1007/978-3-540-72397-4_25. URL http://link.springer.com/10.1007/978-3-540-72397-4{_}25.

Valenzuela, C., Hurley, S., e Smith, D. A permutation based Genetic Algorithm for minimum span frequency assignment. In *5th International Conference on Parallel Problem Solving from Nature*, pages 907–916, 1998. doi: 10.1007/BFb0056932. URL http://dx.doi.org/10.1007/BFb0056932.

van Omme, N., Perron, L., e Furnon, V. or-tools user's manual. Technical report, Google, 2014.

Vieira, C. E. C., Gondim, P. R. L., Rodrigues, C. A., e Bordim, J. L. A new technique to the channel assignment problem in mobile communication networks. In *2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*, number 3, pages 1–5. IEEE, sep 2008. ISBN 978-1-4244-2643-0. doi: 10.1109/PIMRC.2008.4699742. URL http://ieeexplore.ieee.org/document/4699742/.

Wang, W. e Rushforth, C. K. An Adaptive Local-Search Algorithm for the Channel-Assignment Problem (CAP). *IEEE Transactions on Vehicular Technology*, 45(3), 1996.

Welsh, D. J. A. e Powell, M. B. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, 1967.

Wu, Q. e Hao, J.-K. Coloring large graphs based on independent set extraction. *Computers & Operations Research*, 39:283–290, 2012.

Yesil, C., Yilmaz, B., e Korkmaz, E. Hybrid Local Search Algorithms on Graph Coloring Problem. In *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on*, pages 468 – 473, 2011.

Yin, P.-y. e Li, S.-c. Hybrid Ant Colony Optimization for the Channel Assignment Problem in Wireless Communication. In *Swarm Intelligence, Focus on Ant and Particle Swarm Optimization*, number December. I-Tech Education and Publishing, dec . ISBN 9783902613097. doi: 10.5772/5116.

Zoellner, J. A. e Beall, C. L. A breakthrough in spectrum conserving frequency assignment technology. *IEEE Transactions on Electromagnetic Compatibility*, EMC-19(3): 313–319, 1977.

Apêndice A

Lista de eventos e publicações

A.1 Artigos em periódicos

1. "On distance graph coloring problems", publicado no periódico *International Transactions in Operational Research* - Qualis A2 (de Freitas et al., 2019).

    - Versão preliminar (anterior) disponibilizada no *arXiv*, com o título "Distance geometry approach for special graph coloring problems" (de Freitas et al., 2016).

2. "Notes on models for distance coloring problems", aceito para publicação no periódico Matemática Contemporânea (Dias et al., 2019b).

3. "The Distance Polytope for the Vertex Coloring Problem", publicado na coleção *Combinatorial Optimization* da revista *Lecture Notes in Computer Science*, edição ISCO 2018 (Dias et al., 2018e).

4. "Integer and constraint programming approaches for providing optimality to the bandwidth coloring problem", submetido (em revisão) para o periódico *RAIRO - Operations Research* - Qualis B1.

    - Versão preliminar disponibilizada no *Optimization Online*, com o título "Solving the bandwidth coloring problem applying constraint and integer programming techniques" (Dias et al., 2016d).

5. "Facet-inducing inequalities and a cut-and-branch algorithm for the bandwidth coloring polytope based on orientation model", publicado na revista *Electronic Notes*

in Discrete Mathematics, edição LAGOS 2017 Dias et al. (2017d).

- Versão estendida submetida (em revisão) para o periódico *Discrete Applied Mathematics* - Qualis A1.

A.2 Conferências internacionais

1. "Notes on the Bandwidth Coloring Problem and IP / CP Models", resumo aceito e trabalho apresentado no *2019 INFORMS International Conference* (9 a 12 de junho de 2019, Cancún - México) (Dias et al., 2019a).

2. "On distance colorings, graph embedding and IP/CP models", resumo aceito e trabalho apresentado no *VIII Latin American Workshop on Cliques in Graphs* - LAWCG 2018 (9 a 11 de agosto de 2018, Rio de Janeiro, RJ - Brasil) (Dias et al., 2018d).

3. "Notes on advances in theoretical modeling and resolution of graph coloring with distance constraints", resumo aceito e trabalho apresentado na *Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization* - ALIO/EURO 2018 (25 a 26 de junho de 2018, Bolonha - Itália) (Dias et al., 2018a).

4. "The distance polytope for vertex coloring", resumo expandido aceito e trabalho apresentado no *V International Symposium on Combinatorial Optimization* - ISCO 2018 (11 a 13 de abril de 2018, Marraquexe - Marrocos) (Dias et al., 2018b).

5. "Facet-inducing inequalities and a cut-and-branch algorithm for the bandwidth coloring polytope based on orientation model", artigo aceito e trabalho apresentado no *IX Latin and American Algorithms, Graphs and Optimization Symposium* - LAGOS 2017 (11 a 15 de setembro de 2017, Marselha - França) (Dias et al., 2017b).

6. "Advances in solving graph coloring problems", apresentado na *21st Conference of the International Federation of Operational Research Societies* - IFORS 2017 (17 a 21 de julho de 2017, Quebéc - Canadá) (Dias et al., 2017c).

7. "Channel assignment as a distance geometry graph coloring problem", apresentado na *27th European Conference on Operational Research* - EURO 2015 (12 a 15 de julho de 2015, Glasgow - Escócia/Reino Unido) (Dias et al., 2015).

8. “Distance coloring problems, spatial properties and feasibility conditions”, resumo aceito e trabalho apresentado no *VI Latin American Workshop on Cliques in Graphs* - LAWCG 2014 (9 a 12 de novembro de 2014, Pirenópolis, RS - Brasil) (Dias et al., 2014b).

9. “Some Challenges in n-dimensional Euclidean Distance Geometry Problems”, apresentado no *INFORMS Annual Meeting 2014* (9 a 12 de novembro de 2014, São Francisco - Estados Unidos) (Dias et al., 2014a).

10. “On feasibility conditions in graph coloring problems with distance constraints”, apresentado no workshop *Many Faces of Distances* (22 a 24 de outubro de 2014, Campinas, SP - Brasil) (de Freitas et al., 2014).

11. “Some notes of Euclidean distance geometry and graph theory involving telecom, computer networks and molecular biology applications”, apresentado na *20th Conference of the International Federation of Operational Research Societies* - IFORS 2014 (13 a 18 de julho de 2014, Barcelona - Espanha) (Dias et al., 2014c).

A.3 Conferências nacionais

1. “Sobre o DGP, realizações de grafos em $\mathbb{R}^n$ e colorações”, trabalho apresentado no XXXVIII Congresso Nacional de Matemática Aplicada e Computacional - CNMAC 2018 (17 a 21 de setembro de 2018, Campinas, SP - Brasil) (Dias et al., 2018c).

2. “Politopo baseado em distâncias para o problema clássico de coloração de vértices em grafos”, resumo aceito e trabalho apresentado no III Encontro de Teoria da Computação, dentro do XXXVIII Congresso da Sociedade Brasileira de Computação - ETC/CSBC 2018 (22 a 26 de julho, Natal, RN - Brasil) (Dias et al., 2016c).

3. “Sobre modelos e algoritmos para coloração em grafos com restrições de distância”, artigo aceito e trabalho apresentado no XLIX Simpósio Brasileiro de Pesquisa Operacional - SBPO 2017 (27 a 30 de agosto de 2017, Blumenau, SC - Brasil) (Dias et al., 2017a).

4. “Constraint and integer programming models for bandwidth coloring and multicoloring in graphs”, artigo aceito e trabalho apresentado no XLVIII Simpósio Brasileiro de Pesquisa Operacional - SBPO 2016 (27 a 30 de setembro de 2016, Vitória, ES - Brasil) (Dias et al., 2016b).

5. “On Special Vertex Coloring Models in Graphs Using Distance Geometry”, apresentado no *XI Brazilian Workshop on Continuous Optimization* - BRAZOPT 2016 (22 a 27 de maio de 2016, Manaus, AM - Brasil) (Dias et al., 2016a).

6. “Some results in graph coloring as Euclidean distance geometry problem”, apresentado no II Workshop Franco-brasileiro de Grafos e Otimização Combinatória - GCO 2016 (28 de março a 1 de abril de 2016, Praia da Redonda, CE - Brasil) (Dias et al., 2017c).

Apêndice B

Conceitos básicos

Neste apêndice, serão abordados alguns conceitos necessários para compreensão do trabalho elaborado, envolvendo otimização discreta, combinatória poliédrica, teoria dos grafos e geometria de distâncias. A escolha por tratar destes temas neste capítulo decorre de sua aplicação para obtenção dos resultados desta tese.

Ressalta-se que a parte envolvendo combinatória poliédrica foi colocada em uma seção própria pois esta tese é a primeira do PPGI/IComp/UFAM a abordar tal técnica, logo, espera-se que tal material possa auxiliar a introduzir estes conceitos a novos pesquisadores do programa.

B.1 Otimização linear e discreta

Um problema de otimização consiste em encontrar a melhor solução dentre um conjunto de candidatos. A qualidade de uma solução é medida por meio de uma **função objetivo**, sendo que em um problema de minimização, a melhor solução será aquela de menor valor, e em um problema de maximização, a solução será a de maior valor (Papadimitriou e Steiglitz, 1998). O formato básico de um problema de otimização é:

$$\text{Minimizar / maximizar} \quad f(x) \tag{B.1}$$

$$\text{Sujeito a} \quad x \in P \tag{B.2}$$

Onde x é uma solução, f é a função objetivo e P é o conjunto de soluções factíveis (viáveis). Este conjunto é definido pelas características e restrições cada problema. O

tamanho deste conjunto pode ser contável ou não, sendo que no primeiro caso, o problema é dito ser de otimização combinatória. Sem perda de generalidade, serão considerados problemas de minimização, uma vez que $\max f(x) = -\min -f(x)$ (Goldbarg e Luna, 2005).

Uma ferramenta bastante utilizada para modelagem de problemas de otimização é a programação matemática, na qual o problema a ser resolvido é enunciado por um conjunto de expressões matemáticas, sendo uma a função objetivo, consistindo no critério a ser otimizado, e as demais as restrições do problema. Uma das principais classes de modelos programação matemática é a programação linear. Nela, a função objetivo é linear (ou seja, um polinômio de grau 1), e o conjunto de soluções é definido por restrições também lineares. Sendo n e m dois inteiros positivos, onde n é o número de variáveis e m o de restrições; $A = [a_{ij}]$ uma matriz de dimensão $n \times m$ (os coeficientes das restrições); um vetor $c = [c_i] \in \mathbb{R}^n$ (os custos das variáveis em um problema de minimização) e um vetor $b = [b_i] \in \mathbb{R}^m$ (as demandas), a seguinte formulação é um problema de programação linear:

$$\text{Minimizar} \quad \sum_{i=1}^{n} c_i x_i \tag{B.3}$$

$$\text{Sujeito a} \quad \sum_{i=1}^{n} a_{ij} x_i = b_j \quad (\forall j = 1, ..., m) \tag{B.4}$$

$$x_i \geq 0 \qquad (\forall i = 1, ..., n) \tag{B.5}$$

Neste caso, as variáveis podem assumir valores reais, o que torna o conjunto de soluções que atendem às restrições (chamadas de soluções factíveis) incontável. Porém, existe um método, denominado de simplex, que na prática é bastante eficaz na resolução de problemas de programação linear com todas as variáveis contínuas. Apesar disso, a complexidade de pior caso do mesmo é exponencial, mas na grande maioria dos casos, o algoritmo é o mais rápido empiricamente. Outros métodos incluem o das elipsóides e o de pontos interiores (Papadimitriou e Steiglitz, 1998; Maculan e Fampa, 2006).

Nos problemas de otimização combinatória, entretanto, as variáveis devem assumir valores inteiros, uma vez que as soluções possuem elementos discretos. A programação inteira é uma variação da programação linear com essa restrição adicional. Quando apenas algumas variáveis devem ser inteiras, diz-se que o problema é de programação inteira

mista. Sendo p o número de variáveis inteiras e q o de variáveis não inteiras (onde $p+q=n$), com as variáveis inteiras indexadas por l e as contínuas por k, a formulação a seguir é de programação inteira mista:

$$\text{Minimizar} \quad \sum_{l=1}^{p} c_l x_i + \sum_{k=1}^{q} c_k x_k \tag{B.6}$$

$$\text{Sujeito a} \quad \sum_{l=1}^{p} a_{lj} x_l + \sum_{k=1}^{q} a_{kj} x_i = b_j \quad (\forall j = 1, ..., m) \tag{B.7}$$

$$x_l \geq 0 \qquad (\forall i = 1, ..., n) \tag{B.8}$$

$$x_k \geq 0 \qquad (\forall i = 1, ..., q) \tag{B.9}$$

Outra técnica de modelagem utilizada é a programação por restrições, cujo foco é encontrar uma solução factível para um conjunto de restrições, e não a solução ótima. Nela, ao invés de uma função objetivo, são considerados os domínios de valores para as variáveis, além das restrições (van Omme et al., 2014).

Uma de suas principais vantagens é a facilidade de modelagem. As restrições podem assumir várias formas, como aritméticas (sem a necessidade de serem lineares) e lógicas (booleanas), além de globais e simbólicas para subestruturas naturais dos problemas (Bockmayr, 2006; Trick). Restrições deste último tipo costumam já ser implementadas em ferramentas computacionais de programação por restrições, utilizando algoritmos eficientes especializados na satisfação das mesmas. O Catálogo de Restrições Globais (*Global Constraint Catalog*) (Beldiceanu e Beldiceanu, 2014) conta com 354 restrições desse tipo já conhecidas e estabelecidas na literatura e em ferramentas.

Uma das restrições mais utilizadas é conhecida como *AllDifferent*(variáveis), que exige que todos os valores das variáveis consideradas sejam diferentes uns dos outros. Tal restrição exige uma certa expansão do modelo quando traduzida para a programação matemática. Para duas variáveis x_i e x_j, a restrição *AllDifferent*(x_i, x_j) (ou seja, $x_i \neq x_j$) fica do seguinte modo:

$$x_i - x_j + 1 \leq m y_i$$
$$x_j - x_i + 1 \leq m y_j$$
$$y_i + y_j = 1$$
$$y_i, y_j \in \{0, 1\}$$
$$0 \leq x_i, x_j \leq m - 1$$

Apesar da equivalência, além de ser mais simples de ser enunciada, a restrição *All-Different*(variáveis) é bastante estudada na literatura e, por isso, possui muitos algoritmos eficientes disponíveis para seu tratamento específico.

Uma grande parte dos problemas de otimização combinatória pertence à classe **NP-difícil**, o que significa que provavelmente não exista algoritmo eficiente (ou seja, de complexidade polinomial) para resolvê-los. Além disso, o conjunto de soluções é exponencial, o que torna uma busca exaustiva inviável na maioria dos casos. Para resolvê-los, existem duas vertentes de pesquisa: os métodos exatos, que fornecem soluções garantidamente ótimas em um tempo computacional que pode ser grande, e os métodos aproximados, onde as soluções podem não ser ótimas, mas são de boa qualidade e obtidas em um tempo mais satisfatório.

Apesar de o tempo computacional dos métodos exatos ser inviável em alguns casos, há grande interesse em algoritmos desse para problemas NP-difíceis. O uso dos mesmos pode ser útil para verificação de limites inferiores ou mesmo valores de soluções ótimas em instâncias de *benchmark* e também para aplicação em cenários onde a solução desejada deve, obrigatoriamente, ser a melhor possível.

Alguns métodos exatos são:

- ***Branch-and-bound*:** envolve a utilização de algoritmos para programação linear contínua (como o simplex) no problema com variáveis inteiras, removendo temporariamente as restrições de integralidade (processo chamado de relaxação). Se a solução obtida para o problema relaxado satisfizer as restrições de integralidade já neste momento, a solução é ótima também para o problema original. Caso contrário, sendo x_i uma variável do problema relaxado que deveria ser inteira no problema original e cujo valor é contínuo (denotado por x_i^R), são criados dois subproblemas (passo de ramificação), onde cada um equivale ao problema relaxado original, e cada um recebe uma restrição adicional:

  - Problema 1: $x_i \leq \lfloor x_i^R \rfloor$.
  - Problema 2: $x_i \leq \lceil x_i^R \rceil$.

  Cada subproblema é resolvido de forma recursiva, até que a melhor solução inteira seja encontrada. Para evitar a enumeração completa dos subproblemas, são usados limites inferior (LI) e superior (LS) em cada subproblema. Se o problema for de

$$
\begin{aligned}
\text{Minimizar} \quad & z = -5x_1 - 8x_2 \\
\text{Sujeito a} \quad & x_1 + x_2 \leq 6 \qquad \text{(B.10)} \\
& x_1, x_2 \geq 0 \qquad \text{(B.11)} \\
& x_1, x_2 \in \mathbb{Z} \qquad \text{(B.12)}
\end{aligned}
$$

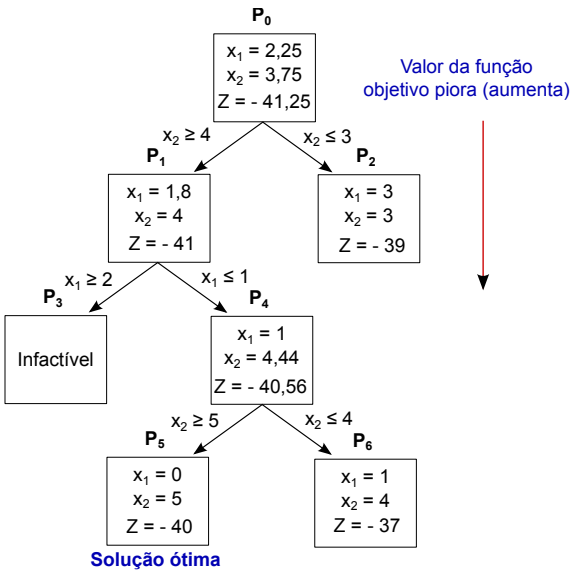


Figura B.1: Exemplo de aplicação do *branch-and-bound*.

minimização, a solução relaxada de um problema será um LI e a melhor solução inteira será um LS (caso o problema seja de maximização, será o contrário). Como a adição de restrições tende a piorar a qualidade das soluções, o valor do LI tenderá a aumentar e o do LS a diminuir. Quando LI $\geq$ LS, não há necessidade de ramificar o subproblema, uma vez que não haverá como obter uma solução melhor. A Figura B.1 exemplifica o método.

- **Planos de corte:** a ideia consiste em adicionar novas restrições ao problema relaxado, de forma que a solução ótima contínua seja removida do conjunto de soluções factíveis, sem remover pontos inteiros do mesmo. Aplicando sucessivos cortes, a solução ótima inteira será encontrada diretamente por métodos de programação linear contínua.

  Existem duas categorias de cortes: os genéricos e os específicos de um problema. Os genéricos são aplicáveis em qualquer problema de programação inteira, como os cortes de Gomory e *zero-half*, no entanto, podem ter convergência lenta por não haver exploração inteligente da região factível do problema. Já os específicos são baseados em características do problema a ser resolvido, por exemplo, garantindo que os vértices de uma clique em um grafo utilizem cores diferentes.

- ***Branch-and-cut*:** combina características do *branch-and-bound* com planos de corte. A cada subproblema gerado no *branch-and-bound*, se a solução relaxada não for inteira, são encontrados cortes que eliminem tal solução, na esperança de que

$$
\begin{aligned}
\text{Minimizar} \quad & z = -4x_1 - 8x_2 \\
\text{Sujeito a} \quad & 2x_1 + 2x_2 \leq 19 && \text{(B.13)} \\
& -2x_1 + 2x_2 \leq 3 && \text{(B.14)} \\
& x_1, x_2 \geq 0 && \text{(B.15)} \\
& x_1, x_2 \in \mathbb{Z} && \text{(B.16)} \\
& && \text{(B.17)}
\end{aligned}
$$

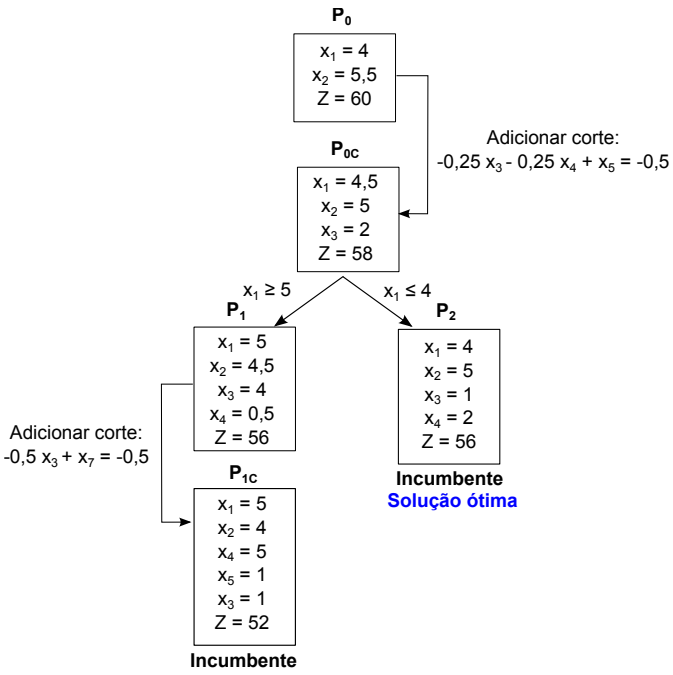


Figura B.2: Exemplo de aplicação do *branch-and-cut*, onde a cada subproblema com solução contínua, é aplicado um corte de Gomory.

seja encontrada uma solução inteira. A geração de cortes pode seguir diferentes critérios, por exemplo, inserindo uma quantidade máxima de restrições genéricas ou específicas ou efetuando a separação de todos os cortes que podem ser obtidos a partir de uma mesma solução relaxada, ainda que a solução obtida após tal processo ainda não seja inteira, situação na qual será realizada uma ramificação da mesma forma que ocorre no *branch-and-bound* tradicional.

- ***Branch-and-prune*:** este método é utilizado em modelos de programação por restrições. Consiste em atribuir um valor a uma variável de acordo com seu domínio. Em seguida, realiza-se a propagação das restrições (processo também conhecido como *pruning*), onde valores dos domínios das outras variáveis que violem restrições em relação ao valor atribuído à variável atual são removidos. Se um domínio se tornar vazio, a solução parcial é infactível, assim, é feito um *backtrack* a um passo anterior e um outro valor é testado. Caso o domínio tenha apenas um elemento, ele será o valor da variável. Quando todos os domínios têm um único elemento, tem-se uma solução factível completa. Para que o método seja aplicado à otimização, sendo z_0 o custo da solução obtida, deve-se adicionar uma restrição $z < z_0$ ao problema (van Omme et al., 2014).

Em muitos casos práticos, mesmo bons métodos exatos podem consumir muito tempo

para obter soluções ótimas para os problemas de otimização combinatória. Sendo assim, muitos pesquisadores investem nas abordagens aproximadas, que fornecem boas soluções (mas não necessariamente as melhores possíveis) em um tempo computacional razoável. Uma abordagem aproximada bastante utilizada consiste em algoritmos heurísticos, que exploram o espaço de busca utilizando características desejadas nas melhores soluções para o problema, sem garantias de quão perto da melhor solução o algoritmo chegará.

As meta-heurísticas são abordagens aproximadas mais genéricas, consistindo em *frameworks* algorítmicos que podem ser adaptados à maioria dos problemas de otimização combinatória (Luke, 2009). Em geral, as meta-heurísticas possuem um passo estocástico, que permite efetuar uma exploração mais eficaz do espaço de soluções. Algumas das principais meta-heurísticas bastante utilizadas incluem busca local iterada (*iterated local search* - ILS), *simulated annealing*, algoritmos genéticos e meméticos, busca tabu, nuvem de partículas, reconexão de caminhos, Procedimentos de Busca Adaptativa Gulosa Aleatória (*Greedy Randomized Adaptive Search Procedures* - GRASP) e *scatter search* (Luke, 2009; Talbi, 2009; Dorne e Hao, 1998a)

B.2 Combinatória poliédrica

As formulações de programação linear, técnica introduzida na seção anterior, são amplamente usadas para a modelagem de diversos tipos de problemas. Isso leva ao surgimento de propriedades matemáticas interessantes, que podem ter ligação direta com características combinatórias do problema a ser resolvido. Por meio da combinatória poliédrica, estuda-se tais características de modo a obter melhoras na resolução dos problemas desejados. Alguns dos principais conceitos da área serão introduzidos a seguir.

Seja n um número natural. Considere, então, um conjunto de m pontos $x^1, x^2, \ldots, x^m \in \mathbb{R}^n$. Um ponto $z \in \mathbb{R}^n$ é dito ser uma **combinação afim** de $x^1, x^2, \ldots, x^m \in \mathbb{R}^n$ se existirem $\lambda_1, \lambda_2, \ldots, \lambda_n \in \mathbb{R}$ tais que $z = \sum_{i=1}^{n} \lambda_i x^i$ e $\sum_{i=1}^{n} \lambda_i = 1$. Se, além disso, para todo $1 \leq i \leq m$, tiver-se que $\lambda_i \geq 0$, então z é dito também ser **combinação convexa** dos pontos $x^1, x^2, \ldots, x^m \in \mathbb{R}^n$ (Boyd e Vandenberghe, 2004). A Figura B.3 mostra exemplos destas combinações.

Considere um conjunto $X \subseteq \mathbb{R}^n$ de pontos. Diz-se que X é um **conjunto convexo** se o mesmo for um subconjunto do espaço afim que seja fechado sob combinações convexas

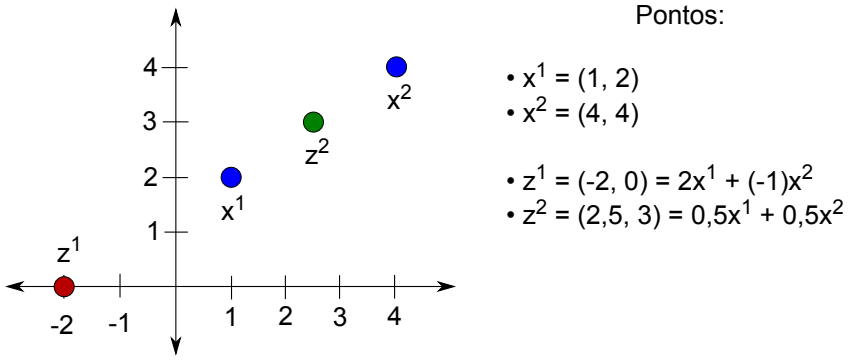


Figura B.3: Exemplo gráfico de combinações afim e convexa. O ponto z^1 é combinação afim dos pontos x^1 e x^2, pois para o mesmo, tem-se que $\lambda_1 = 2$ e $\lambda_2 = -1$, de modo que $\lambda_1 + \lambda_2 = 1$, porém, não é combinação convexa já que $\lambda_2 < 0$. Já o ponto z^2, além de combinação afim, também é combinação convexa de x^1 e x^2, já que, para o mesmo, $\lambda_1 = \lambda_2 = 0,5$ (ou seja, $\lambda_1, \lambda_2 \geq 0$) e, por consequência e $\lambda_1 + \lambda_2 = 1$.

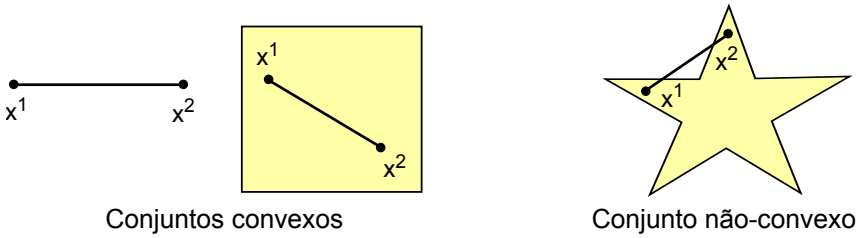


Figura B.4: Exemplos gráficos de conjuntos convexos e não-convexos em espaços Euclidianos.

(Ferreira e Wakabayashi, 1996). Em espaços Euclidianos, como consequência desta definição, tem-se que, para qualquer par de pontos (x^1, x^2) no conjunto convexo, quaisquer outros pontos no segmento de reta que liga x^1 a x^2 também farão parte do conjunto, como pode ser visto na Figura B.4.

Seja novamente $X \subseteq \mathbb{R}^n$ um conjunto de pontos. O **fecho convexo** (também chamado de casco convexo) de X, denotado por $\text{conv}(X)$, consiste no conjunto de todas as combinações possíveis dos pontos em X, ou seja, é o menor conjunto convexo que contém X. Desse modo, qualquer ponto de X pode ser reescrito como uma combinação convexa dos pontos extremos do fecho convexo (Boyd e Vandenberghe, 2004). Um exemplo gráfico deste conceito é dado na Figura B.5.

O conceito fundamental da combinatória poliédrica é o de **poliedro**, denotado por P e que consiste em um subconjunto de $\mathbb{R}^n$, descrito por um conjunto finito de restrições lineares, ou seja, $P = \{x \in \mathbb{R} \mid Ax \leq b\}$. A **dimensão** de um poliedro P, denotada por $dim(P)$, equivale ao número máximo de pontos afim independentes em P menos 1. Um caso particular de poliedro é o **politopo**, que também é um subconjunto de $\mathbb{R}^n$, de modo que exista um conjunto finito $X \subseteq \mathbb{R}^n$ cujo fecho convexo seja P (Schriver, 1986). Como consequência, tem-se que P é um politopo se, e somente se, P for um poliedro limitado,

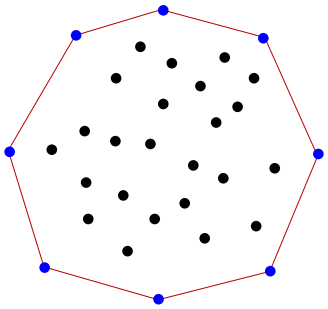


Figura B.5: Exemplo gráfico de fecho convexo de um conjunto de pontos. Os pontos azuis são os extremos (também pertencentes ao conjunto de pontos), que servem de vértices para o polígono definido por tal fecho.

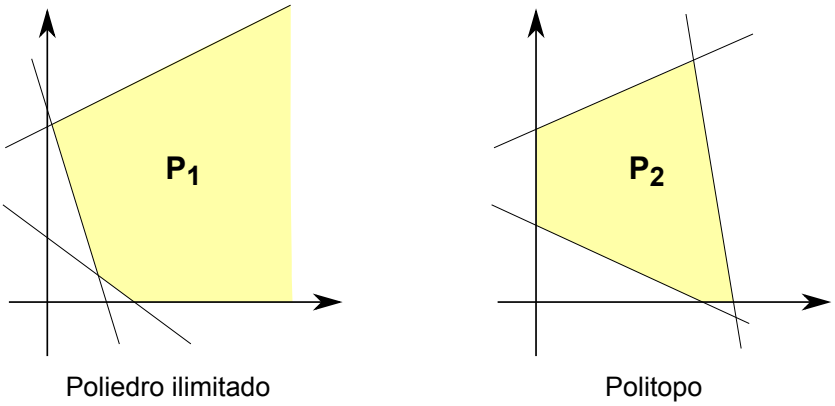


Figura B.6: Exemplos de poliedro ilimitado e politopo.

como exemplificado na Figura B.6.

Pelos conceitos apresentados anteriormente, é possível observar que modelos de programação linear definem poliedros em $\mathbb{R}^n$, sendo n o número de variáveis do modelo. Para problemas de programação inteira mista, nos quais algumas variáveis assumir apenas valores inteiros, o conjunto de pontos que definem soluções viáveis passa a ser $X \subseteq \mathbb{Z}^{n_i} \times \mathbb{R}^{n_c}$, onde n_i é o número de variáveis inteiras e n_c o de contínuas (de modo que $n = n_i + n_c$). Diz-se então que um poliedro P é uma **formulação** para X se, e somente se, $X = P \cap (\mathbb{Z}^{n_i} \times \mathbb{R}^{n_c})$.

Para problemas de programação linear, a solução ótima é sempre um dos vértices do politopo definido pelo modelo correspondente. Entretanto, quando o problema é de programação inteira mista, tais vértices podem não ser solução, já que as variáveis inteiras podem ter valores contínuos nos mesmos. Desse modo, a solução ótima para um problema de programação inteira pode não estar num dos vértices do politopo. Para que isso sempre ocorra, deve-se obter uma descrição exata do fecho convexo dos pontos que são solução para o modelo (ou seja, nos quais as variáveis inteiras não tenham valores contínuos), o que só é possível se $P = NP$, uma vez que programação inteira é NP-completo, mas com

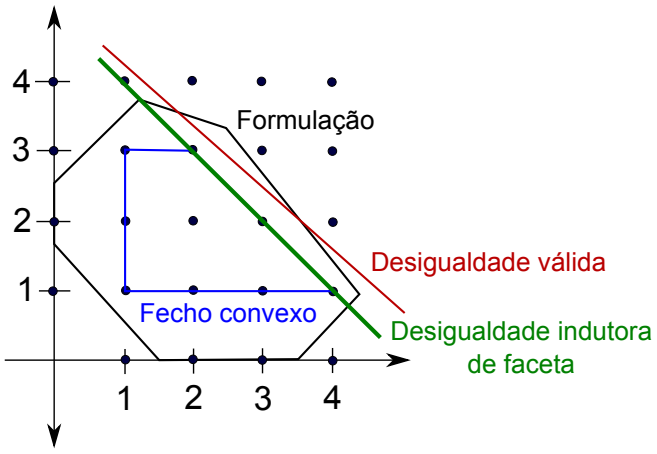


Figura B.7: Exemplos de formulação, desigualdade válida e faceta do fecho convexo de um politopo.

a descrição exata do fecho convexo das soluções, o problema passa a ser de programação linear, que possui algoritmo polinomial (Schriver, 1986).

Os algoritmos de *branch-and-bound*, planos de core e *branch-and-cut* citados na seção anterior são capazes de lidar com esse problema. Para estes dois últimos, são adicionadas novas restrições ao modelo, de modo a remover soluções contínuas, o que pode fazer com que a solução ótima inteira se torne um vértice do politopo. Assim, pode-se definir uma expressão $\alpha^T x \leq \beta$ como uma desigualdade válida para um conjunto de pontos $X \subseteq \mathbb{Z}^{n_i} \times \mathbb{R}^{n_c}$ se $\alpha^T x \leq \beta$ para todo $x \in X$. As melhores desigualdades válidas são capazes de definir **facetas** de um politopo P, o que ocorre se $dim(\{x \in P \mid \alpha^T x \leq \beta\}) = dim(P) - 1$, ou seja, se existirem $dim(P) - 1$ pontos $p_1, p_2, \ldots, p_{dim(P)-1}$ afim independentes tais que $\alpha^T x \leq \beta$ para $i = 1, 2, \ldots, dim(P) - 1$ (Ferreira e Wakabayashi, 1996; Pendavingh, 2015). A dimensão de um politopo corresponde também à diferença entre a dimensão do espaço e o posto das restrições de igualdade do politopo (isto é, a quantidade de igualdades linearmente independentes), de modo que $dim(P) = n - posto(P^=)$. A Figura B.7 mostra um exemplo gráfico destes conceitos.

B.3 Teoria dos grafos

Um grafo é um par ordenado $G = (V, E)$, onde V é um conjunto de vértices e E é um conjunto de arestas, onde $n = |V|$ e $m = |E|$. Cada elemento e no conjunto E é um par (i, j) que indica que o vértice i é ligado ao vértice j (ou seja, são adjacentes, e a aresta e incide em i e j). O grafo é dito não-direcionado quando os pares que representam as arestas não são ordenados, isto é, $(i, j) = (j, i)$. Além disso, um grafo é dito ser conexo

se existe pelo menos um caminho entre cada par de vértices (ou fortemente conexo se tal propriedade existir em um grafo direcionado).

A representação gráfica de um grafo consiste em pontos distintos do plano associados a cada vértice e, para cada aresta (i, j), uma linha conectando os pontos correspondentes aos vértices i e j. Se for possível efetuar uma representação gráfica do um grafo G sem que as arestas se cruzem, diz-se que G é **planar** (Szwarcfiter, 1986). Um exemplo é dado na Figura B.8. Ressalta-se, no entanto, que grafos em geral apresentam propriedades relacionais e topológicas, uma vez que existem diferentes representações gráficas para um mesmo grafo.

O **grau** de um vértice i do grafo equivale à quantidade de arestas que incidem em i. O grau máximo do grafo, denotado por $\Delta(G)$, é o valor do maior grau dentre todos os vértices de G. De maneira similar, o grau mínimo, denotado por $\delta(G)$, é definido como o valor do menor grau de G. Um grafo é dito ser **k-regular** se todos os vértices possuem grau igual a um inteiro positivo k (ou seja, se G é k-regular, então $\Delta(G) = \delta(G) = k$). Um **grafo completo**, denotado por K_n é aquele no qual cada vértice é vizinho de todos os demais, ou seja, o grau de todos os vértices é exatamente igual a $n-1$ e, por consequência, o grafo é $(n-1)$-regular. Já um grafo vazio ou nulo é aquele no qual não há arestas, ou seja, todos os vértices têm grau 0, o que faz o grafo ser 0-regular. O grafo vazio é, então, o complemento de um grafo completo (Bondy e Murty, 1982). Um exemplo de grafo completo e seu complemento é dado na Figura B.9.

Uma **clique** H de um grafo G é um subgrafo completo de G, ou seja, cada vértice de H está ligado a todos os outros do subgrafo. Uma clique é dita maximal se a mesma não puder ser estendida pela inclusão de algum vértice adjacente. A clique máxima de um grafo é a maior dentre todas, sendo necessariamente também maximal (ressalta-se, porém, que nem toda clique maximal é máxima). O tamanho da clique máxima de G é denominado de número clique e denotado por $\omega(G)$ (Chartrand e Zhang, 2009). Um

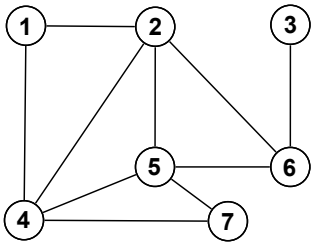


Figura B.8: Representação gráfica de um grafo não-direcionado planar com 7 vértices e 10 arestas.

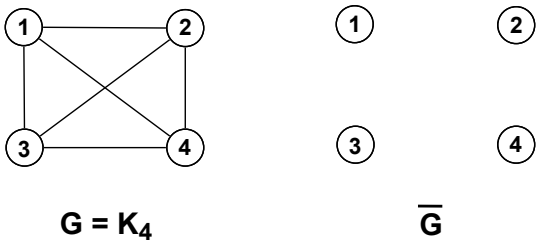


Figura B.9: Exemplo de grafo completo com 4 vértices (K_4) e seu complemento.

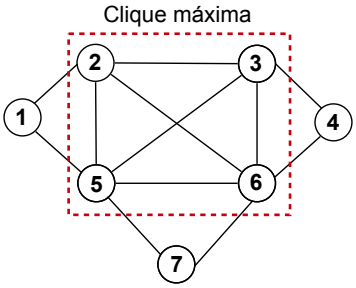


Figura B.10: Exemplo de clique máxima de um grafo, onde tem-se que o número clique do mesmo é $\omega(G) = 4$.

exemplo de clique máxima de um grafo pode ser visto na Figura B.10.

Um **conjunto independente** S de G é um subgrafo vazio de G, ou seja, não há nenhuma aresta entre os vértices de S. Similarmente ao que ocorre com a clique, um conjunto independente é dito maximal se o mesmo não puder ser estendido pela inclusão de algum vértice adicional. O conjunto independente máximo é o maior dentre todos, sendo também maximal, e seu tamanho é denominado de número de independência ou de estabilidade, denotado por $\alpha(G)$. Nota-se que este conceito é complementar ao de clique, de modo que uma clique de um grafo G é um conjunto independente do complemento $\overline{G}$ do grafo e vice-versa (Chartrand e Zhang, 2009). Um exemplo destes conceitos é dado na Figura B.11.

Alguns tipos de grafos são muito comuns e possuem propriedades que os destacam, sendo alguns deles enunciados a seguir. Um **caminho** de n vértices, denotado por P_n, é um grafo no qual dois vértices têm grau 1 (chamados de extremidades) e onde os $n-2$ vértices restantes têm grau 2. Um **ciclo**, denotado por C_n, é similar a um caminho com a adição de uma aresta ligando as extremidades, ou seja, é um grafo onde todos os vértices têm grau 2 e o número de arestas é o mesmo de vértices. Uma **árvore** é um grafo conexo acíclico no qual existe exatamente um caminho entre cada par de vértices (ou seja, um caminho é um caso particular de árvore). Um **grafo bipartido** é um grafo cujo conjunto de vértices pode ser dividido em dois subconjuntos disjuntos, de modo que não exista aresta entre vértices

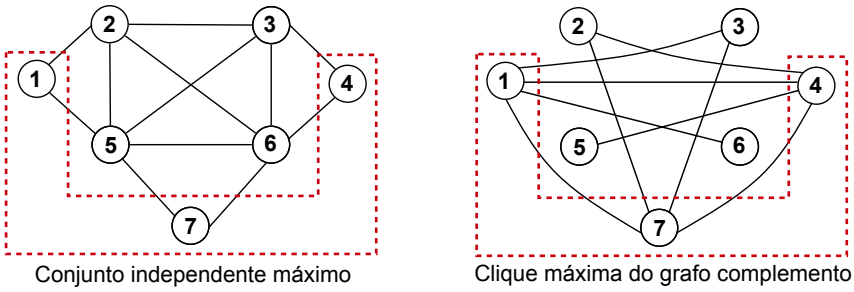


Figura B.11: Exemplo de conjunto independente máximo de um grafo, onde tem-se que o número de independência do mesmo é $\alpha(G) = 3$, e sua equivalência à clique máxima do complemento do grafo.

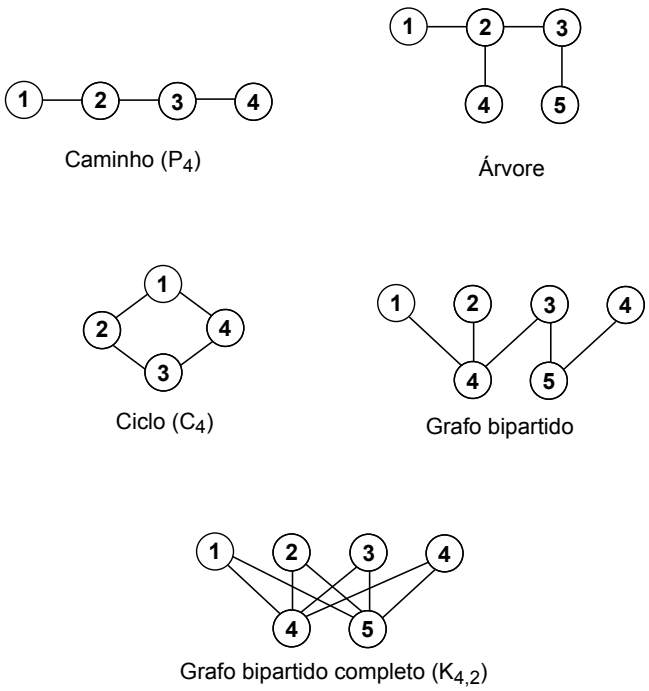


Figura B.12: Exemplos de grafos de algumas das classes mais conhecidas.

de um mesmo subconjunto, mas somente entre vértices de subconjuntos diferentes. Diz-se, então, que um grafo é **bipartido completo** se houver aresta entre cada par de vértices onde cada um dos mesmos esteja em um subconjunto diferente, sendo tal grafo denotado por $K_{n,m}$, onde n é o número de vértices de um subconjunto e m o do outro (Szwarcfiter, 1986). A Figura B.12 fornece exemplos destes grafos.

B.4 Geometria de distâncias

A geometria de distâncias (GD) consiste em determinar localizações de pontos no espaço respeitando distâncias entre pares conhecidas. O problema fundamental de GD pode ser formalizado como se segue (Liberti et al., 2014).

Definição B.4.1. ***Problema de Geometria de Distâncias (Distance Geometry Problem - DGP):*** *dado um inteiro positivo K e um grafo não-direcionado simples $G = (V,E)$, cujas arestas são ponderadas por uma função $d : E \to \mathbb{R}^+$, determinar se existe ou não uma função $x : V \to \mathbb{R}^n$ tal que $\forall (i,j) \in E, ||x(i) - x(j)|| = d(i,j)$ (Saxe, 1979).*

Tal problema pode ser visto, então, como a projeção de um grafo no espaço euclideano n-dimensional. O mesmo é NP-completo para $n = 1$, como provado por Saxe (1979) e fortemente NP-difícil para $n > 1$. Uma tentativa de prova de que o problema é NP para $n = 2$ (o que, por consequência, provaria sua NP-completude para tal caso) foi realizada por Beeker et al. (2013), mas sem sucesso.

Um caso especial do DGP com estruturas discretas no espaço tridimensional ($\mathbb{R}^3$), aplicado a problemas de obtenção de estruturas de proteínas, é definido a seguir.

Definição B.4.2. ***Problema de Geometria de Distâncias Moleculares Discretizáveis (Discretizable Molecular Distance Geometry Problem - DMDGP):*** *dado um grafo não direcionado simples $G = (V,E)$, cujas arestas são ponderadas por uma função $d : E \to \mathbb{R}^+$ e em cujo conjunto V de vértices exista uma ordem $v_1, v_2, \ldots, v_{|V|}$ tal que:*

1. *todas as quádruplas de vértices consecutivos formam cliques ($\forall i \in \{4, \ldots, n\}\ \forall j,k \in \{i-3, \ldots, i\}\ (\{j,k\} \in E)$);*

2. *a desigualdade triangular estrita a seguir é válida:*
   $\forall i \in \{2, \ldots, n-1\}\ \ d(i-1,i+1) < d(i-1,i)\ +\ d(i,i+1)$

determinar se existe ou não uma função $x : V \to \mathbb{R}^3$ tal que $\forall (i,j) \in E, ||x(i) - x(j)|| = d(i,j)$ (Lavor et al., 2012).

Tal definição permite que a determinação da projeção do grafo em $\mathbb{R}^3$ se torne um problema de busca discreta, o que fornece ganhos de tempo de execução e precisão em comparação a métodos contínuos. O DMDGP pode ser generalizado para um problema com definições menos rígidas, definido a seguir para o espaço tridimensional.

Definição B.4.3. ***Problema de Geometria de Distâncias Discretizáveis (Discretizable Distance Geometry Problem - DDGP):*** *dado um grafo não direcionado simples $G = (V,E)$, cujas arestas são ponderadas por uma função $d : E \to \mathbb{R}^+$ e em cujo conjunto V de vértices exista uma ordem parcial $v_1, v_2, \ldots, v_{|V|}$ tal que:*

1. *exista um subconjunto V_0 de V no qual:*

   - $|V_0| = 3$*;*
   - *a relação de ordem em V_0 seja total;*
   - V_0 *induz a uma clique de G;*
   - *para todo $u_0 \in V_0$ e para todo $u \in (V \backslash V_0)$, tem-se que $u_0 < u$.*

2. *para cada $i \in V \backslash V_0$, exista um subconjunto $\{v_1, v_2, v_3\}$ de V tal que:*

   - $v_1 < i,\ v_2 < i,\ v_3 < i$*;*
   - $\{(v_1, i),\ (v_2, i),\ (v_3, i)\} \subseteq E$*;*
   - *a desigualdade triangular estrita a seguir é válida:*
     $d(v_1, v_3) < d(v_1, v_2) + d(v_2, v_3)$.

determinar se existe ou não uma função $x : V \to \mathbb{R}^3$ tal que $\forall (i, j) \in E, ||x(i) - x(j)|| = d(i, j)$ (Mucherino et al., 2012).

Os principais métodos para problemas de geometria de distâncias, em especial em sua aplicação na determinação da estrutura de proteínas (Sallaume et al., 2013), incluem um algoritmo de tempo linear, proposto por Dong e Wu (2002), que pode ser usado quando as distâncias entre todos os pares de vértices são conhecidas (ou seja, o grafo de entrada é completo), e o algoritmo *branch-and-prune* de Lavor et al. (2012) para o DMDGP, que foi posteriormente usado também para o DDGP por Mucherino et al. (2012), usado quando nem todas as distâncias são conhecidas. Em tal algoritmo, devido às características geométricas da definição dos problemas, a posição de um vértice $i \geq 4$ pode ser determinada pela posição dos vértices anteriores $i-1, i-2$ e $i-3$, por meio da intersecção de três esferas com raios $d(i-3, i), d(i-2, i)$ and $d(i-1, i)$, de modo que são obtidas duas posições possíveis cujas factibilidades devem ser verificadas, como exemplificado na Figura B.13.

B.5 Informações adicionais

Este capítulo apresentou brevemente alguns conceitos teóricos explorados nos resultados apresentados nesta tese. No entanto, existem muitos outros conceitos, por exemplo, outras classes de problemas de otimização, técnicas para resolução dos mesmos e avaliação

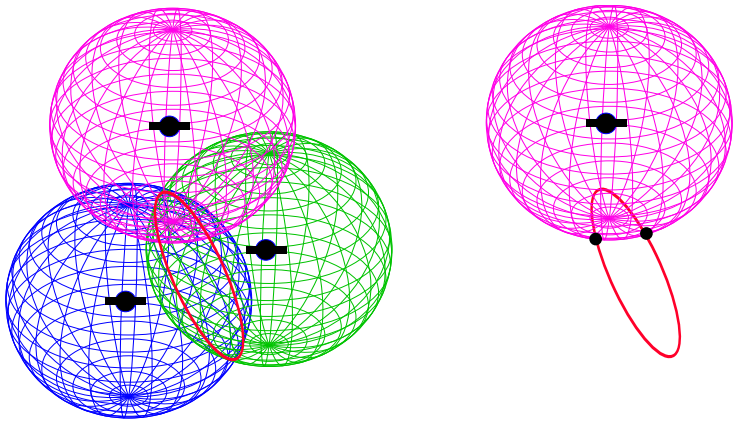


Figura B.13: Exemplos de intersecção de 3 esferas e os pontos correspondentes obtidos.

da eficiência dos algoritmos. Mais sobre este assunto pode ser visto nos livros de Papadimitriou e Steiglitz (1998) e Lawler (1976). Uma obra sobre programação matemática por Bradley et al. (1977) e livros específicos sobre meta-heurísticas por Luke (2009) e Talbi (2009) também estão disponíveis na literatura. Para programação por restrições, uma boa fonte introdutória é o manual das ferramentas *or-tools* disponibilizadas pelo Google (van Omme et al., 2014).

Sobre combinatória poliédrica, devido a sua forte ligação com o estudo de programação inteira e otimização convexa, diversas obras abordam estes assuntos de forma conjunta, como os livros de Schriver (1986) e Boyd e Vandenberghe (2004). A apostila de Ferreira e Wakabayashi (1996) é um bom material introdutório em português. Um outro texto recomendado para saber mais foi desenvolvido por Pendavingh (2015) para um curso de otimização linear.

Em teoria dos grafos, as obras consideradas referências do assunto são os livros de Szwarcfiter (1986) e Bondy e Murty (1982), que contém diversas características e propriedades de grafos. Uma obra mais recente é o livro de Chartrand e Zhang (2009), com ênfase em problemas de coloração em grafos (chamada pelo autor de teoria cromática de grafos).

Por fim, existem muitas outras aplicações de geometria de distâncias. Um conjunto de trabalhos da área pode ser visto no trabalho de Mucherino et al. (2013). Um *survey* da teoria e aplicações de geometria de distâncias foi elaborado por Liberti et al. (2014). Há também uma obra que explora relações entre otimização convexa e geometria de distâncias euclidianas, desenvolvida por Datorro (2013) e disponível livremente.

Apêndice C

Coloração clássica de vértices

Neste apêndice, será apresentado um complemento da revisão apresentada no Capítulo 2. Desta vez, a coloração clássica (VCP) será abordada, sendo que, para a mesma, também será mostrada uma definição formal de tal problema, bem como modelos de programação inteira e algoritmos exatos e aproximados da literatura.

Muitos dos resultados disponíveis para o VCP podem servir como base para novos métodos de resolução para o BCP e colorações com distâncias, e como a coloração clássica é a mais estudada na literatura, torna-se interessante conhecer seus resultados.

C.1 Definições

Para um grafo não direcionado $G = (V, E)$, uma coloração própria de G consiste em uma atribuição de cores (números naturais) para cada elemento de V de forma que vértices adjacentes não compartilhem a mesma cor. Uma k-coloração de G é uma coloração própria que usa no máximo k cores, de forma que para todo $i \in V$, $c(i) \leq k$.

Usando estas definições, pode-se expressar a versão de otimização do problema de coloração clássica de vértices em grafos, como mostrado abaixo.

Definição C.1.1. ***Problema de Coloração de Vértices em Grafos (Vertex Coloring Problem - VCP):*** *dado um grafo simples não direcionado $G = (V, E)$, onde V é o conjunto de vértices e E o de arestas, encontrar a k-coloração de G onde k seja o menor valor possível.*

A função objetivo do VCP, ou seja, o menor número de cores necessárias para colorir todos os vértices de G, é conhecida como número cromático de G, denotado por

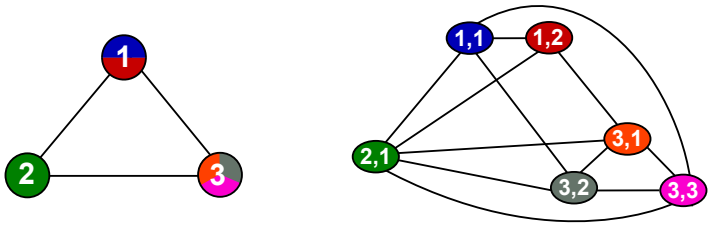


Figura C.1: Exemplo de instância de multicoloração e sua conversão para o VCP.

$\chi(G)$. Toda k-coloração de G onde $k = \chi(G)$ é ótima. Destaca-se que há uma variação do VCP, denominada de multicoloração, onde cada vértice pode ter uma demanda de mais de uma cor (ou seja, o mesmo pode necessitar ser colorido com uma ou mais cores distintas). Neste caso, a definição de k-coloração não se aplica diretamente, sendo a versão de otimização desta variação definida como se segue.

Definição C.1.2. ***Problema de Multicoloração de Vértices em Grafos:*** *dado um grafo não direcionado $G = (V, E)$, onde V é o conjunto de vértices e E o de arestas, e, para cada vértice $i \in V$, um peso $w_i \in \mathbb{N}$, determinar, para cada $i \in V$, w_i cores distintas $(c(i,1),\ c(i,2),\ \ldots,\ c(i,w_i))$ tais que para toda aresta $(i,j) \in E$, tenha-se, para todo $1 \leq p \leq w_i$ e $1 \leq q \leq w_j$, que $c(i,p) \neq c(j,q)$, e a quantidade de valores distintos para cada $c(i,p)$ (ou seja, de cores distintas usadas) seja a menor possível.*

Apesar da definição diferente, o problema é equivalente ao VCP. Uma instância de multicoloração pode ser convertida em uma do VCP transformando-se cada vértice i em uma clique de w_i vértices. Além disso, para cada aresta $(i,j) \in E$, tem-se que cada um dos w_i vértices da clique associada a i será adjacente a cada um dos w_j vértices da clique associada a j. Por meio desta transformação, todo método de resolução do VCP pode ser usado para multicoloração. A Figura C.1 exemplifica esta técnica.

Sendo o primeiro problema de coloração em grafos abordado na literatura, o mesmo tem sido extensivamente estudado de várias formas, incluindo a complexidade computacional do problema para algumas classes específicas de grafos, além de limites superiores e inferiores para o número cromático. A seguir, são mostradas algumas das características mais importantes relativas ao tema.

C.2 Modelos teóricos e de programação matemática

A forma mais comum de representar o VCP é simplesmente seguindo a definição do problema tal como definida na subseção anterior, isto é, determinando-se diretamente cores para os vértices que respeitem a restrição de que extremos de uma mesma aresta não usem cores iguais. Seguindo tal modelo, uma formulação padrão de programação inteira para o VCP pode ser definida utilizando dois conjuntos de variáveis:

- $x_{ik} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao vértice } i; \\ 0 & \text{caso contrário.} \end{cases}$
- $y_k = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída a algum vértice;} \\ 0 & \text{caso contrário.} \end{cases}$

O conjunto de cores possíveis deve ser pré-determinado, uma vez que as variáveis são indexadas tanto pelos vértices quanto pelas cores (no caso das variáveis x). Ao invés de considerar o conjunto de cores possíveis como $\mathbb{N}$, pode-se limitá-lo ao intervalo $[1..LS]$, onde LS é um limite superior válido dentre os mostrados na subseção anterior. Será assumido $LS = \Delta(G)+1$ (Welsh e Powell, 1967), por ser o limite mais conhecido, o que torna a função de coloração definida como $c : V \rightarrow [1, \Delta(G)+1]$.

A formulação de PI é, então:

(PI-Col-Padrão)

$$\text{Minimizar} \quad \sum_{k=1}^{\Delta(G)+1} y_k \tag{C.1}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{\Delta(G)+1} x_{ik} = 1 \quad (\forall i \in V) \tag{C.2}$$

$$x_{ik} + x_{jm} \leq 1 \quad (\forall i, j \in V;\ k, m = 1, 2, ..., \Delta(G)+1 : (i, j) \in E) \tag{C.3}$$

$$x_{ik} \leq y_k \quad (\forall i \in V;\ k = 1, 2, ..., \Delta(G)+1) \tag{C.4}$$

$$x_{ik} \in \{0, 1\} \quad (\forall i \in V;\ k = 1, 2, ..., \Delta(G)+1) \tag{C.5}$$

$$y_k \in \{0, 1\} \quad (\forall k = 1, 2, ..., \Delta(G)+1) \tag{C.6}$$

A função objetivo (C.1) é a minimização do número de variáveis do conjunto y que

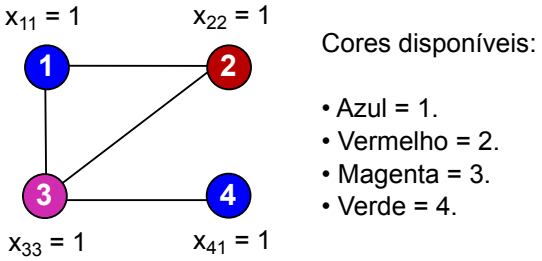


Figura C.2: Examplo de instância do problema clássico de coloração de vértices em grafos e sua solução ótima com 3 cores codificada no modelo de programação inteira clássico.

recebam o valor 1, o que corresponde à minimização do número de cores usadas. As restrições (C.2) requerem que cada vértice receba exatamente uma cor (sendo que, para multicoloração, pode-se substituir o lado direito por w_i ao invés de expandir o grafo de entrada). O conjunto de restrições (C.3) exigem que vértices adjacentes não utilizem a mesma cor. Restrições (C.4) garantem que se, para uma determinada cor k, houver uma variável x_{ik} com valor 1 (independentemente do vértice i), então a variável y_k também deverá ter valor 1. Por fim, restrições (C.5) e (C.6) são de integralidade. Na Figura C.2, tem-se um exemplo de instância cuja solução é codificada como variáveis desta formulação.

A formulação acima tem dimensão polinomial. Existem $O(|V|^2)$ variáveis no conjunto x e $O(|V|)$ variáveis no conjunto y. Além disso, a formulação possui $O(|V|^2|E|+|V||E|+|V|)$ restrições. Tal modelo pode ser adaptado para outras variações de colorações de vértices, como será visto nas próximas seções. Porém, há problemas de simetria, uma vez que, dada uma solução, outra equivalente pode ser obtida pela mera troca dos vértices que usam uma determinada cor pelos vértices que usam outra e vice-versa. O limite inferior obtido pela sua relaxação linear também é considerado fraco (Malaguti e Toth, 2010). Apesar disso, o modelo tem sido utilizado com sucesso em algoritmos exatos (Méndez-Díaz e Zabala, 2006; Aardal et al., 1996).

Uma segunda forma de representar o VCP é baseada em conjuntos independentes. Como definido na Seção B.3, um conjunto independente de um grafo é um subconjunto de vértices não adjacentes entre si. Sendo assim, é possível colorir todos os membros de um conjunto independente com a mesma cor, sem violar restrições do VCP. Diz-se então que cada um desses conjuntos define uma classe de cor do grafo.

É importante ressaltar, para este modelo teórico, que o número de independência do grafo $\alpha(G)$ não corresponde ao número cromático do mesmo, nem tampouco o conjunto

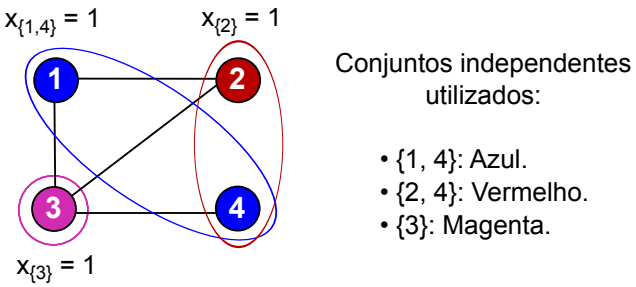


Figura C.3: Examplo de instância do problema clássico de coloração de vértices em grafos e sua solução ótima com 3 cores codificada como conjuntos independentes.

independente máximo corresponderá a uma classe de cor presente na solução ótima do VCP (já que o número cromático seria equivalente, na verdade, ao número de conjuntos independentes usados). Porém, existe uma relação entre ambos, dada pela expressão $\chi(G) \geq \left\lceil \frac{|V|}{\alpha(G)} \right\rceil$ (Chartrand e Zhang, 2009; Lewis, 2016).

Com base nesta modelagem, uma formulação de programação inteira foi desenvolvida por Mehrotra e Trick (1996). Seja I o conjunto de todos os conjuntos independentes maximais de G. Tem-se o seguinte conjunto de variáveis:

- $x_S = \begin{cases} 1 & \text{se o conjunto independente } S \in I \text{ receberá uma única cor;} \\ 0 & \text{caso contrário.} \end{cases}$

O modelo é definido como:

(PI-Col-ConjIndep)

$$\text{Minimizar} \quad \sum_{S \in I} x_S \tag{C.7}$$

$$\text{Sujeito a} \quad \sum_{S \in I \,:\, i \in S} x_S \geq 1 \quad (\forall i \in V) \tag{C.8}$$

$$x_S \in \{0,1\} \qquad (\forall S \in I) \tag{C.9}$$

A função objetivo (C.7) é a minimização do número de conjuntos independentes maximais que receberão uma única cor, o que equivale à minimização da quantidade de cores usadas. Restrições (C.8) exigem que cada vértice do grafo deve estar presente em pelo menos um dos conjuntos independentes que serão utilizados como classe de cor. Por fim, o conjunto (C.9) consiste de restrições de integralidade. A Figura C.3 mostra um exemplo de instância cuja solução é codificada como variáveis desta formulação.

Apesar de esta formulação ter $O(|V|)$ restrições, a mesma tem um número exponencial de variáveis. Como há uma para cada conjunto independente maximal do grafo, há $O(3^{|V|/3})$ variáveis no modelo completo (seguindo o mesmo limite da quantidade de conjuntos independentes de um grafo (Moon e Moser, 1965)). No entanto, este modelo foi usado com sucesso no desenvolvimento de algoritmos *branch-and-price*, como será mostrado mais adiante. Além disso, a formulação não sofre do mesmo problema de simetria da anterior, pois as cores não são diretamente atribuídas, já que os vértices de cada conjunto independente podem usar qualquer cor, desde que, caso pertençam ao mesmo conjunto, usem as mesmas cores e, em caso contrário, usem cores diferentes (Mehrotra e Trick, 1996; Malaguti e Toth, 2010).

Outra forma de modelar o VCP envolve a definição de representantes assimétricos, proposta por Campêlo et al. (2008) e que também é baseada baseado no conceito de classes de cores (ou seja, conjuntos independentes).

Seja W_k a classe de cor contendo os vértices que usam a cor k. Para cada classe W_k, um vértice $i \in W_k$ é escolhido para representar a mesma. Para evitar simetria entre vértices na mesma classe de cor, uma ordem $\prec$ é definida tal que, se $i \prec j$, então j não pode representar a classe da cor de i, e a variável x_{ij} é fixada com valor 0. Considerando o complemento $\overline{G} = (\overline{V}, \overline{E})$ do grafo $G = (V, E)$ (onde $\overline{V} = V$ e $\overline{E} = \{(i, j) \ : \ i, j \in V \text{ and } (i, j) \notin E\}$, verifica-se que $\prec$ induz a uma orientação de $\overline{G}$, uma vez que vértices adjacentes no complemento podem utilizar a mesma cor.

Seja $N(i)$ o conjunto de vértices adjacentes a i e $\overline{N}(i)$ seu complemento (ou seja, $\overline{N}(i) = V \setminus (N(i) \cup \{i\})$). Além disso, define-se $\overline{N}^+(i) = \{j \in \overline{N}(i) \ : \ i \prec j\}$, $\overline{N}^-(i) = \{j \in N(i) \ : \ j \prec i\}$, $G^+(v)$ o subgrafo induzido contendo apenas os vértices de $\overline{N}^+(i)$ e $G^-(v)$ o subgrafo induzido contendo apenas os vértices de $\overline{N}^-(i)$. A orientação induzida por $\prec$ em $\overline{G}$ é acíclica, logo, há dois conjuntos especiais de vértices: $S = \{s \ : \ \overline{N}^-(s) = \emptyset\}$ e $T = \{t \ : \ \overline{N}^+(t) = \emptyset\}$, cujos subgrafos induzidos são cliques.

Para utilizar o modelo teórico com uma formulação de programação inteira, considera-se variáveis:

- $x_{ij} = \begin{cases} 1 & \text{se o vértice } i \in V \text{ representa a classe da cor de } j \in \overline{N}(i); \\ 0 & \text{caso contrário.} \end{cases}$

- $y_i = \begin{cases} 1 & \text{se o vértice } i \in V \text{ também está em S;} \\ x_{ii} & \text{caso contrário.} \end{cases}$

A formulação é definida como:

(PI-Col-ReprAssim)

$$\text{Minimizar} \quad \sum_{i \in V} x_{ii} \tag{C.10}$$

$$\text{Sujeito a} \quad \sum_{j \in \overline{N}^-(i)} x_{ji} \geq 1 \quad (\forall i \in V \setminus S) \tag{C.11}$$

$$\sum_{j \in L} x_{ij} \leq y_i \qquad (\forall i \in V \setminus T;\ L \subseteq \overline{N}^+(i)\ :\ \text{clique induzida tem tamanho 1 ou 2}) \tag{C.12}$$

$$x_{ij} \in \{0,1\} \qquad (\forall i \in V;\ j \in \overline{N}(i)) \tag{C.13}$$

$$y_i \in \{0,1\} \qquad (\forall i \in V) \tag{C.14}$$

A função objetivo (C.10) consiste na minimização do número de vértices cujo representante de sua cor na coloração é ele próprio, o que equivale a minimizar a quantidade de cores usadas, já que vértices representados por outros compartilharão as mesmas cores. Tal função também pode ser reescrita como $\sum_{i \in V \setminus S} x_{ii} + |S|$, pois vértices no conjunto S (chamados de origens) não podem ser representados por outros. Restrições (C.11) garantem que cada vértice deve ser representado por si mesmo ou por um outro vértice não adjacente. O conjunto (C.12) exige que vértices adjacentes entre si, mas que não são a um determinado vértice adicional, tenham representantes diferentes. Por fim, as restrições (C.13) e (C.14) são de integralidade. A Figura C.4 mostra um exemplo de solução codificada por este modelo.

Esta formulação possui $O(|V|^2 - |E|)$ variáveis, já que há uma para cada aresta do complemento do grafo de entrada. O maior conjunto de restrições é (C.12), onde há $O(|V|(|V|^2 - |E|)) = O(|V|^3 - |V||E|)$ elementos. Assim, o modelo tem tamanho polinomial, além de não ter problemas de simetria pela definição da ordenação $\prec$. Campêlo et al. (2008) também apresentaram um estudo poliédrico da formulação e restrições adicionais que definem facetas, de forma a utilizá-la em algoritmos *branch-and-cut*, como feito para o problema de coloração fracionária (Campêlo et al., 2009).

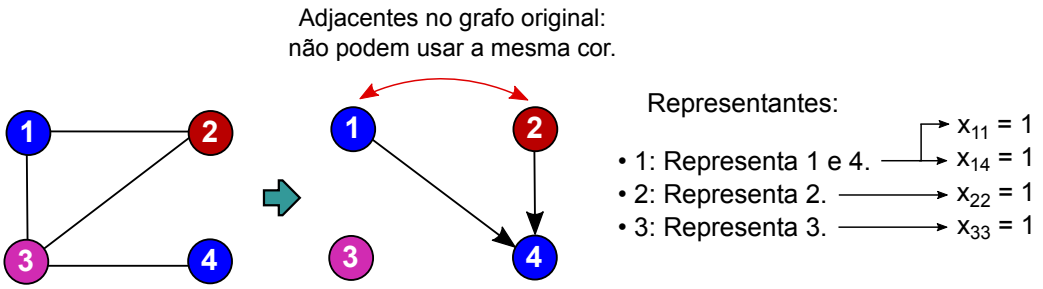


Figura C.4: Examplo de instância do problema clássico de coloração de vértices em grafos e sua solução ótima com 3 cores codificada utilizando representantes assimétricos e grafo complementar.

O último modelo teórico para o VCP foi proposto por Burke et al. (2010) e é derivada de um particionamento do grafo em cliques, baseando-se também em conjuntos independentes. Dessa maneira, dado um grafo $G = (V,E)$, tem-se que cada clique do mesmo corresponde a um conjunto independente do complemento $\overline{G}$ do grafo, e cada vértice da clique deverá usar uma cor diferente dos demais (enquanto que, em uma clique de $\overline{G}$, todos os vértices da mesma usarão a mesma cor). Particionando-se os vértices do grafo complementar $\overline{G}$ em k subconjuntos $(H_1, H_2, \ldots, H_k)$ tais que os subgrafos induzidos pelos mesmos formam cliques, tem-se que o menor k para o qual existe tal particionamento é igual ao número cromático do grafo original G.

Seja $G' = (V',E')$ um grafo induzido por uma partição $H = \{H_1, H_2, \ldots, H_k\}$ em cliques de G, onde $V' = H$ e $E' = \{(D,F) : (i,j) \in E; D,F \in H; D \neq F; i \in D; j \in F\}$. Tal particionamento é dito reversível se, para cada $D \in H$, todos $i \in D$ são indistinguíveis, ou seja, são adjacentes uns aos outros e têm a mesma vizinhança. Cada $D \in H$ é um supernó, onde o mesmo representa mais de um vértice do grafo original. Considerando G' como entrada de um problema de multicoloração, onde cada vértice recebe mais de uma cor, a solução desta multicoloração equivale a da coloração clássica no grafo G original.

Para este modelo teórico, uma formulação de programação inteira do mesmo usa variáveis:

$$x_{Dk} = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída ao supernó } D \in H; \\ 0 & \text{caso contrário.} \end{cases}$$

$$y_k = \begin{cases} 1 & \text{se a cor } k \text{ for atribuída a algum supernó;} \\ 0 & \text{caso contrário.} \end{cases}$$

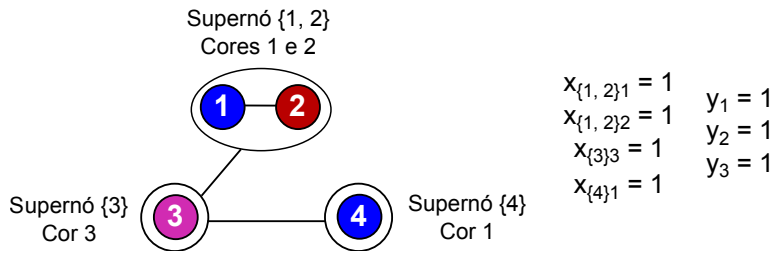


Figura C.5: Examplo de instância do problema clássico de coloração de vértices em grafos e sua solução ótima com 3 cores codificada utilizando particionamento reversível em cliques.

A formulação é definida então como:

(PI-Col-PartClique)

$$\text{Minimizar} \quad \sum_{k=1}^{\Delta(G)+1} y_k \tag{C.15}$$

$$\text{Sujeito a} \quad \sum_{k=1}^{\Delta(G)+1} x_{Dk} = |D| \quad (\forall D \in H) \tag{C.16}$$

$$x_{Dk} + x_{Fm} \leq 1 \quad (\forall D, F \in H;\ k, m = 1, 2, ..., \Delta(G)+1 : (D, F) \in E') \tag{C.17}$$

$$x_{Dk} \leq y_k \quad (\forall D \in H;\ k = 1, 2, ..., \Delta(G)+1) \tag{C.18}$$

$$x_{Dk} \in \{0, 1\} \quad (\forall D \in H;\ k = 1, 2, ..., \Delta(G)+1) \tag{C.19}$$

$$y_k \in \{0, 1\} \quad (\forall k = 1, 2, ..., \Delta(G)+1) \tag{C.20}$$

Este modelo é praticamente igual ao clássico aplicado no grafo G', mas considerando, como conjunto de cores, o intervalo $[1..\Delta(G)+1]$ ao invés de $[1..\Delta(G')+1]$ (que equivale a $[1..|H|]$). Porém, como discutido anteriormente, o particionamento reversível em cliques deve ser multicolorido ao invés de receber apenas uma coloração clássica, logo, o lado direito do conjunto de restrições (C.16) é modificado de forma que cada supernó D receba $|D|$ cores, o que assegura que cada vértice do grafo original receba uma cor. A Figura C.5 mostra um exemplo de solução codificada por este modelo.

Existem $O(|V(G)||H|)$ variáveis no conjunto x e $O(|H|)$ variáveis no conjunto y, de forma similar à formulação clássica. Há também $O(|H| + |V(G)||E'|)$ restrições. Este modelo evita problemas de simetria, uma vez que as cores atribuídas a cada supernó podem ser distribuídas de qualquer modo aos vértices internos. A formulação foi utilizada com sucesso em problemas de agendamento de horários (*timetabling*), no entanto, foi

apenas implementada diretamente com um resolvedor geral de programação inteira e não em um algoritmo específico.

A Tabela C.1 fornece um sumário das formulações de programação inteira descritas para a coloração clássica.

Tabela C.1: Formulações de programação inteira para a coloração clássica de vértices.

Formulação	Variáveis	Restrições	Referências
PI-Col-Padrão - Padrão	$O(\|V\|^2)$	$O(\|V\|^2\|E\| + \|V\|\|E\| + \|V\|)$	-
PI-Col-ConjIndep - Conjuntos independentes	$O(3^{\|V\|/3})$	$O(\|V\|)$	Mehrotra e Trick (1996, 2007)
PI-Col-ReprAssim - Representantes assimétricos	$O(\|V\|^2 - \|E\|)$	$O(\|V\|^3 - \|V\|\|E\|)$	Campêlo et al. (2008, 2009)
PI-Col-PartClique - Particionamento em cliques	$O(\|V\|\|H\|)$	$O(\|H\| + \|V\|\|E'\|)$	Burke et al. (2010)

C.3 Algoritmos exatos e aproximados

A seguir, serão brevemente apresentados alguns dos principais algoritmos propostos na literatura para a coloração clássica de vértices, com o objetivo de encontrar o número cromático do grafo.

Os primeiros algoritmos apresentados serão exatos. Um dos primeiros da literatura foi proposto por Brown (1972), baseado na técnica de *backtracking*, onde cada vértice tem uma lista de possíveis cores e a coloração é construída vértice a vértice até atingir uma solução completa. Tal algoritmo foi posteriormente melhorado por Brélaz (1979), que introduziu o conceito de grau de saturação de um vértice, equivalente à quantidade de vértices vizinhos que já estão coloridos, sendo usado em um algoritmo *branch-and-bound*, denominado de DSATUR$_{B\&B}$ que colore o grafo de acordo com uma ordem de vértices seguindo os graus de saturação. Esse, por sua vez, também foi posteriormente melhorado com regras alternativas de desempate de vértices com mesmo grau de saturação por Sewell (1996) e Segundo (2012) e com técnicas de atualização de limite inferior por Furini et al. (2016) para que o *gap* de otimalidade seja reduzido a zero mais rapidamente.

Outras abordagens exatas para o VCP foram desenvolvidas com base em programação inteira. Um algoritmo *branch-and-price* foi desenvolvido por Mehrotra e Trick (1996) a

Tabela C.2: Algoritmos exatos para a coloração clássica de vértices.

Referência	Estratégia algorítmica
Brown (1972)	*Backtracking*
Brélaz (1979)	DSATUR$_{B\&B}$ - *Branch-and-bound*
Sewell (1996)	Regra de desempate para DSATUR$_{B\&B}$
Mehrotra e Trick (1996)	*Branch-and-price* (formulação por conjuntos independentes)
Méndez-Díaz e Zabala (2006)	*Branch-and-cut* (formulação clássica)
Lucet et al. (2006)	Decomposição linear
Segundo (2012)	Regra de desempate para DSATUR$_{B\&B}$
Furini et al. (2016)	Limites inferiores para DSATUR$_{B\&B}$

partir da formulação por conjuntos independentes (PI-Col-FormIndep), tomando proveito de seu número exponencial de variáveis, as quais são selecionadas por meio de geração de colunas cujo subproblema envolve a determinação de um conjunto independente máximo ponderado. Tal algoritmo também foi posteriormente explorado em multicoloração (Mehrotra e Trick, 2007).

Um método exato seguindo a técnica de *branch-and-cut* foi proposto por Méndez-Díaz e Zabala (2006), usando a formulação padrão (PI-Col-Padrão), onde são usadas desigualdades válidas adicionais baseadas em conjuntos independentes, cliques, buracos, caminhos e bloqueios no grafo de entrada.

Uma última abordagem exata distinta das anteriores foi proposta por Lucet et al. (2006), baseada em decomposição linear, onde o grafo de entrada é decomposto em subgrafos menores de acordo com um conjunto fronteira, que contém vértices comuns a ambos os subgrafos, os quais sempre usarão as mesmas cores em ambos os subgrafos. Uma vez que o número de configurações possíveis é exponencial, Lucet et al. propõem aplicar a decomposição linear apenas com k ou menos blocos.

A Tabela C.2 fornece um resumo das referências citadas de algoritmos exatos para o VCP.

Em contraste com a quantidade de algoritmos exatos presentes na literatura, muitas heurísticas e meta-heurísticas foram propostas ao longo dos anos para a coloração clássica, com variados graus de sucesso. Devido a similaridades entre algumas delas, o foco será em mostrar, para diferentes tipos conhecidos de heurísticas e meta-heurísticas, os principais trabalhos da literatura.

Um algoritmo heurístico construtivo muito conhecido na literatura é o DSATUR$_H$,

proposto por Brélaz (1979) (citado na subseção anterior como base para um método exato), que segue o paradigma guloso e onde o potencial de um vértice a ser selecionado para ser colorido segue os graus de saturação da solução parcial. Outra heurística construtiva bastante usada é o o método RLF (*Recursive Largest First*), proposto por Leighton (1979), onde uma cor é selecionada por vez e o algoritmo identifica conjuntos independentes no grafo cujos membros receberão tal cor, repetindo o processo ao incrementar a cor.

A primeira meta-heurística aplicada à coloração de vértices foi proposta por Chams et al. (1987), sendo um *simulated annealing* (SA) aplicado ao espaço de soluções conflitantes (ou seja, onde há soluções infactíveis). A cada iteração em uma dada temperatura, novas soluções são geradas aleatoriamente da vizinhança básica (onde um vértice tem sua cor alterada).

Outros algoritmos usando *simulated annealing* foram propostos por Johnson et al. (1991), que forneceram três implementações do método, variando o espaço de soluções (válidas ou conflitantes) e vizinhanças, todas com resfriamento exponencial e com a adição de dois parâmetros: porcentagem mínima de movimentos aceitos para cada temperatura e quantidade de vezes que tal porcentagem mínima deve ser atingida para que o processo chegue ao congelamento.

Uma variação interessante do *simulated annealing* é o *quantum annealing* (QA) que insere flutuações quânticas artificiais (conceitos vindos da mecânica quântica) ao SA, sendo que um algoritmo desse tipo para a coloração de grafos foi proposto por Titiloye e Crispin (2011), onde são usadas soluções conflitantes inicialmente aleatórias.

Hertz e Werra (1987) propuseram o algoritmo TABUCOL, que é uma busca tabu para a coloração de vértices. O método também usa soluções conflitantes e procura minimizar a quantidade de conflitos até atingir uma coloração válida. Apesar da aparente simplicidade, muitos outros algoritmos usam o TABUCOL como busca local em suas fases de melhoria de solução.

Outro método que usa busca tabu foi proposto por Lim et al. (2005), baseado também na meta-heurística *Squeaky Wheel Optimization* (SWO), onde sequências de vértices são usadas para definir prioridades e as soluções são melhoradas por meio de busca tabu em uma vizinhança de troca entre pares nas sequências de vértices.

Blöchliger e Zufferey (2008) desenvolveram o algoritmo PARTIALCOL, que é uma

busca tabu que trabalha no espaço de busca de soluções parciais, ou seja, que contém vértices não coloridos, e usa vizinhanças que tentam atribuir cor a vértices não coloridos, removendo violações. O método usa dois esquemas para a lista tabu, de acordo com a solução atual e movimento na vizinhança.

Uma busca tabu pseudo-reativa foi proposta por Chalupa (2011), que usa um esquema tabu dinâmico similar ao de Blöchliger e Zufferey (2008), porém, altera os parâmetros utilizados para o cálculo do período tabu ao invés do mesmo diretamente. Assim, se a busca estiver presa em uma região do espaço de soluções, os parâmetros são alterados para aumento do período tabu.

Métodos híbridos combinando *simulated annealing* e busca tabu foram propostos mais recentemente. Yesil et al. (2011) propuseram um algoritmo SA com *backtracking* que também usa uma lista tabu. O procedimento utiliza soluções parciais, as quais são codificadas como sequências de vértices e separadores. Vértices delimitados entre dois separadores usam a mesma cor, e o objetivo é aumentar o tamanho das soluções parciais, de forma que todos os vértices estejam na sequência. A construção de novas soluções é feito por *backtracking* de acordo com os vértices não coloridos, e as soluções parciais geradas são colocadas na lista tabu.

Outro algoritmo combinando SA e busca tabu foi desenvolvido por Pahlavani e Eshghi (2011), onde são usadas soluções conflitantes e a solução inicial é obtida por uma estratégia gulosa. O SA usado é clássico, usando a vizinhança básica e o critério de aceitação de Metropolis. A cada iteração, é usada a busca tabu com vizinhança básica de troca de cores, com período tabu dinâmico, proporcional ao número de conflitos da solução atual.

Diversas meta-heurísticas baseadas em busca local sem busca tabu como procedimento principal também foram propostas na literatura. Caramia e Dell'Olmo (1999) propuseram o algoritmo HCD, que também procura evitar movimentos repetidos durante a exploração da vizinhança, mas usa prioridades de vértices ao invés de lista tabu para isso. O método explora o espaço de k-colorações válidas, mas aceita soluções piores que a melhor encontrada de forma a visitar mais porções do espaço de busca. O método HCD foi posteriormente melhorado, dando origem ao algoritmo CHECKCOL, proposto por Caramia et al. (2006). Nele, é introduzido um esquema de *checkpointing*, onde, depois de algumas iterações, o algoritmo é reiniciado (como um método *multistart*).

Laguna e Martí (2001) propuseram um GRASP (*Greedy Randomized Adaptive Search*

Procedure, ou Procedimento de Busca Gulosa Aleatória Adaptativa) para colorir grafos esparsos, utilizando soluções conflitantes. A fase construtiva trabalha construindo uma classe de cor por vez e usa uma lista restrita de candidatos baseada no grau dos vértices. A fase de melhoria tenta fundir classes pequenas de cores e, posteriormente, aplica busca local na solução gerada.

Uma busca local iterada (em inglês, *iterated local search* - ILS) foi desenvolvida por Paquete e Stutzle (2002), a qual explora soluções conflitantes, sendo a inicial definida aleatoriamente. O método utiliza três vizinhanças que reduzem conflitos e alteram cores dos vértices, além de quatro operadores de perturbação para escapar de ótimos locais, incluindo um baseado em busca tabu (diversificação direcionada).

Avanthay et al. (2003) propuseram um algoritmo de busca em vizinhanças variadas (em inglês, *variable neighborhood search* - VNS), onde seis tipos de movimentos são usados de forma a escapar de ótimos locais. As soluções usadas são do espaço das conflitantes, onde a inicial é definida aleatoriamente. Cada iteração do VNS usa uma das vizinhanças. Três são baseadas em vértices, onde um vértice conflitante recebe a melhor cor possível, e as outras três são baseadas em classes de cores. Como busca local, o método aplica execuções do TABUCOL (Hertz e Werra, 1987) em cima da solução encontrada após explorar a vizinhança. Para fugir dos ótimos locais, o algoritmo efetua uma perturbação que consiste em procurar uma solução aleatória na vizinhança atualmente considerada.

Trick e Yildiz (2007) propuseram um algoritmo de busca em vizinhança grande (em inglês, *large neighborhood search* - LNS), que também utiliza soluções conflitantes. A solução inicial é aleatória e as vizinhanças são derivadas de trabalhos em visão computacional de tamanho exponencial. Para determinar o melhor movimento da vizinhança, o algoritmo determina cortes máximos em subgrafos definidos de acordo com a coloração atual e a vizinhança.

Uma busca em espaço variável foi proposta por Hertz et al. (2008), além das vizinhanças, a função objetivo também é alterada. Três espaços de busca são usados, baseados nos algoritmos TABUCOL, PARTIALCOL e em orientações acíclicas do grafo. Destaca-se que esse método não utiliza critério de perturbação, uma vez que o uso de espaços de busca distintos ajuda na exploração de regiões distintas de soluções.

Um algoritmo genético com busca local foi proposto por Fleurent e Ferland (1996),

onde o espaço de busca do algoritmo é o de soluções conflitantes com conjunto fixo de cores) A população é ordenada de acordo com a função de *fitness* (número de conflitos) e a seleção de um indivíduo é aleatória. No método, são usados dois ipos de codificações de solução, cada uma com operadores genéticos próprios. Um operador adicional de cruzamento, utilizável em ambas as codificações, faz com que um filho use a mesma coloração do pai 1, onde vértices conflitantes no mesmo passarão a usar a cor do pai 2 caso a mesma remova o conflito. O algoritmo de melhoria (busca local) é o TABUCOL.

Outro método hibridizando algoritmo genético e busca local foi proposto Dorne e Hao (1998b), onde também são usadas soluções conflitantes com conjunto fixo de cores. Os indivíduos seguem a codificação por *string* de Fleurent e Ferland (1996) e a população inicial é obtida por uma variante aleatorizada do algoritmo RLF. A seleção de indivíduos para o cruzamento é feita aleatoriamente, com uma probabilidade dada como parâmetro, e o cruzamento em si usa o operador UIS (*Unified Independent Sets*, ou Conjuntos Independentes Unificados), o qual une classes de cores com mais vértices em cada pai. O processo de mutação envolve um processo de busca tabu com caminho aleatório.

Galinier e Hao (1999) propuseram um algoritmo evolucionário híbrido, utilizando soluções conflitantes. A população inicial é obtida por uma adaptação do DSATUR$_H$, com conjunto de cores é fixo, levando alguns vértices de cada solução a não terem cor factível, recebendo então uma cor aleatória que será conflitante e, ao final, é aplicada uma busca local. A seleção de pais na população é feita de forma aleatória e o cruzamento é realizado pelo operador GPX (*Greedy Partition Crossover*, ou Cruzamento de Particionamento Guloso), que consiste em selecionar, a cada passo, um dos pais (pai 1 se o passo for ímpar e pai 2 se for par) e encontrar a maior classe de cores no mesmo. Tal classe vai para o filho e os vértices da mesma são removidos de ambos os pais. Ao final, vértices não coloridos no filho recebem cores aleatórias. A busca local utilizada para melhoria de soluções é o algoritmo TABUCOL adaptado para esse método híbrido, que também substitui o operador de mutação.

Um método de duas fases, chamado de MMT, que envolve conceitos evolutivos e programação inteira, foi proposto por Malaguti et al. (2008), onde são usadas soluções parciais. A primeira fase é a evolutiva, na qual são gerados os limites iniciais. A população inicial é determinada por uma adaptação do DSATUR$_H$ e um algoritmo guloso clássico com ordens aleatórias de vértices. Uma busca local é aplicada em cada solução

inicial gerada, consistindo em uma busca tabu com vizinhança que muda a cor de um vértice e remove as cores de vértices vizinhos que forem iguais à atribuída. O operador de cruzamento é uma modificação do GPX de Galinier e Hao (1999) A segunda fase envolve otimização por colunas, com base no modelo de programação inteira por conjuntos independentes (PI-Col-ConjIndep) de Mehrotra e Trick (1996), onde as colunas usadas são determinadas pelas soluções da fase evolutiva.

Mabrouk et al. (2009) propuseram um algoritmo genético com busca tabu e processamento paralelo, onde o conjunto inicial de cores e a população são determinados por uma aplicações do algoritmo (Leighton, 1979). A seleção de indivíduos é baseada em torneio e o cruzamento é feito pelo operador PMX (*Partially Mapped Crossover*, ou Cruzamento Parcialmente Mapeado) onde duas colorações, representadas como permutações, são cortadas em dois pontos e os filhos são gerados com base na mistura de partes de cada coloração, nos quais é aplicado procedimento de conserto para remover inconsistências. A mutação baseia-se em busca tabu O paralelismo do algoritmo é implementado com o modelo de mestre e escravos, no qual um processador é denotado como mestre e distribui as tarefas aos outros. A população é dividida entre os escravos, que farão cruzamentos e mutações, e repassarão os resultados ao mestre para avaliação e reorganização.

Outro método que combina algoritmo genético com busca tabu é a heurística MACOL, proposta por Lü e Hao (2010), que usa colorações com conflitos. A população inicial é gerada por uma heurística baseada em chances de cada vértice não poder ser colorido no futuro se não receber uma cor na iteração atual. O cruzamento é feito pelo operador AMPaX (*Adaptive Multi-Parent Crossover*, ou Cruzamento Adaptativo com Múltiplos Pais), que é baseado no GPX, porém usando mais de dois pais para determinar um filho. Uma busca tabu é aplicada em seguida, baseada na desenvolvida por Galinier e Hao (1999) e no TABUCOL.

Fister et al. (2011) propuseram um algoritmo de evolução diferencial específico para determinar se um grafo possui ou não uma 3-coloração, chamado de HSA-DE (*Hybrid Self-Adaptive Differential Evolution*, ou Evolução Diferencial Híbrida Auto-Adaptativa). Cada solução consiste em um vetor de $|V|$ números reais, onde o i-ésimo valor do vetor indica a dificuldade para colorir o vértice i (ou seja, define a ordem dos vértices). A decodificação das soluções é baseada no DSATUR$_H$ com base nos vetores gerados. A mutação envolve três soluções aleatórias, onde uma nova é gerada com a soma do vetor

de uma solução à diferença ponderada dos outros dois vetores selecionados. Para o cruzamento, um vetor solução é gerado baseado em dois pais (um da população e outro gerado da mutação), onde, para cada posição do mesmo, com uma dada probabilidade, o valor da mutação é copiado (caso contrário, o valor da solução da população é copiado). Para melhorar as soluções, é utilizada uma busca local em uma vizinhança de troca de posições de vértices nas permutações geradas.

Um algoritmo evolutivo multiagente foi proposto por Chalupa (2011), que se destaca por usar apenas mutação e não cruzamento. Neste método, cada agente é uma solução com tempo de vida limitado, e a população inicial consiste em soluções aleatórias. Cada agente passa por uma busca local, que consiste no TABUCOL e também determina o tempo de vida da solução de acordo com a melhoria. Uma lista elite é usada para armazenar as melhores soluções, para a qual cada agente só pode passar uma solução.

Um método baseado na extração de conjuntos independentes do grafo por meio de um algoritmo genético com busca tabu foi proposto por Wu e Hao (2012), sendo chamado de EXTRACOL. O método consiste em duas fases, pré-processamento e coloração, o que torna necessário o uso de colorações parciais na primeira fase, que serão estendidas para soluções completas na segunda. O pré-processamento consiste na aplicação de uma busca tabu adaptativa para identificar o melhor conjunto independente maximal possível. A função objetivo desta busca tabu é a quantidade de arestas entre os vértices selecionados como conjunto independente (caso seja 0, então os vértices formam um conjunto independente válido). A vizinhança usada envolve trocas entre pares de vértices dentro e fora do conjunto independente.

Abbasian e Mouhoub (2013) desenvolveram um algoritmo paralelo genético hierárquico, onde a busca é realizada no espaço de soluções conflitantes e são usados vários processos mestre (denotando ilhas), cada um com seus escravos. Os mestres se reportam a um processo coordenador, que recebe soluções válidas dos mesmos. Os indíviduos consistem em um vetor de $|V|$ inteiros, onde o i-ésimo elemento é a cor de i. O cruzamento utilizado é o PSC (*Parental Success Crossover*, ou Cruzmento de Sucesso Parental), onde, para cada um dos dois pais, determina-se, para cada vértice, o número de conflitos do mesmo e, então, cada vértice do filho receberá a cor usada no pai que tenha menos conflitos para o vértice. São usados dois tipos de mutação: o primeiro seleciona um vértice i aleatório e troca as cores de seus vizinhos que estejam usando a mesma cor, enquanto o

segundo apenas troca a cor de um vértice por outra aleatória.

Costa e Hertz (1997) propuseram um dos primeiros algoritmos de colônia de formigas para a coloração clássica, denominado de ANTCOL. As soluções usadas correspondem a k-colorações válidas, de tal modo que cada formiga gera uma solução por meio de adaptações dos algoritmos construtivos DSATUR$_H$ e RLF bem como por outros procedimentos gulosos e aleatórios. Os métodos construtivos baseiam-se em uma tabela que corresponde à experiência adquirida pela colônia a cada construção (trilha de feromônios), Quanto melhor a solução, mais as próximas soluções tenderão a usar essa característica memorizada.

Uma melhoria do ANTCOL foi proposta por Salari e Eshghi (2005), cujo algoritmo resultante é chamado de MMGC. Nele, é utilizada uma outra função objetivo, a maximização da soma dos quadrados da quantidade de vértices em cada classe de cor, o que leva o algoritmo a encontrar soluções que usam menos cores. O método também usa heurística na seleção do primeiro vértice a ser colorido no métodos construtivos. Por fim, o MMGC adiciona uma busca local ao algoritmo utilizando uma vizinhança definida por cadeias de Kempe. Tanto a função objetivo quanto a vizinhança são baseadas no *simulated annealing* de Johnson et al. (1991).

Outra melhoria do ANTCOL foi desenvolvida por Dowsland e Thompson (2008), onde são adicionados diversificação, melhorias no algoritmo construtivo e busca local. A diversificação envolve a utilização de duas funções de avaliação de solução e multiplicadores de trilhas de feromônios que criarão uma tendência de vértices não coloridos a serem selecionados. A construção de soluções é baseada no GRASP de Laguna e Martí (2001), de modo a usar construção gulosa aleatória para obter conjuntos independentes maximais que minimizem o número de arestas induzidas para os vértices não coloridos. A busca local é baseada em embaralhamentos de vértices, árvores de decisão e no algoritmo TABUCOL.

Um outro algoritmo de colônia de formigas foi desenvolvido por Plumettaz et al. (2009) com base na busca tabu PARTIALCOL de Blöchliger e Zufferey (2008). O algoritmo também usa colorações parciais e, nele, cada formiga efetua uma busca local além da construção, que consiste na aplicação do PARTIALCOL à solução obtida. Após esse procedimento, é repassada, para iterações futuras, a informação de vértices que usam a mesma cor (o que define a trilha de feromônios).

Outra meta-heurística populacional é a busca dispersa (*scatter search*), que difere dos

algoritmos genéticos por selecionar um subconjunto da população, chamado de conjunto referência, que é dividido em conjunto de melhores e de diversificadas para aplicar operadores de combinação entre soluções e também por substituir indivíduos da população pelo *fitness* e pela diversidade. Para o VCP, Hamiez e Hao (2002) propôs um método desse tipo, que trabalha no espaço de soluções conflitantes. A população inicial é obtida gerando-se colorações com k cores com modificações aleatórias. selecionando, para cada cor, um vértice aleatório e impedindo que seus vizinhos possam usar a mesma cor. Cada solução é melhorada aplicando-se iterações do algoritmo TABUCOL. Para atualização do conjunto referência, se uma nova solução for melhor que a pior do conjunto de melhores, então a nova entra no lugar da pior. Caso contrário, se a sua diversidade for maior que a da menos diversa conjunto de diversificadas, esta última é removida e a nova solução entra em seu lugar. A combinação de soluções é realizada por uma generalização do GPX a um número arbitrário de pais.

Um método que também usa população de soluções é a otimização por nuvem de partículas (*particle swarm optimization* - PSO). Hsu et al. (2011) aplicaram tal metaheurística à coloração de grafos, onde há várias partículas movendo-se em um espaço $|V|$-dimensional. Cada partícula tem um vetor de $|V|$ velocidades e de $|V|$ posições e a sua posição é usada para determinar a cor do vértice. As velocidades são atualizadas de acordo com as melhores soluções que cada partícula encontrou. Como a PSO é um método tradicionalmente contínuo, uma tradução quaternária é usada para mapeamento ao espaço discreto de cores, na qual a cor é determinada pela relação de um valor aleatório com uma constante real entre 0 e 1. O algoritmo aplica a estratégia chamada de *walking one*, que estende a tradução quaternária de modo que vértices com muitos conflitos criem tendência a mudar de cor. Para evitar convergência prematura, é aplicada uma turbulência que aumenta a velocidade de vértices caso que estejam com a mesma baixa em uma partícula.

A Tabela C.3 fornece um sumário dos trabalhos citados nesta subseção em ordem cronológica de publicação.

Tabela C.3: Algoritmos aproximados para a coloração clássica de vértices.

Referência	Estratégia algorítmica
Brélaz (1979)	DSATUR$_H$: Algoritmo guloso de graus de saturação
Leighton (1979)	RLF: Algoritmo guloso recursivo
Chams et al. (1987)	*Simulated annealing* clássico
Hertz e Werra (1987)	TABUCOL: Busca tabu
Johnson et al. (1991)	*Simulated annealing* em três espaços de busca
Fleurent e Ferland (1996)	Algoritmo genético + busca tabu
Costa e Hertz (1997)	ANTCOL: Colônia de formigas
Dorne e Hao (1998a)	Algoritmo genético + busca local
Caramia e Dell'Olmo (1999)	HCD - Busca local com prioridades de vértices
Galinier e Hao (1999)	Algoritmo genético + busca tabu
Laguna e Martí (2001)	Procedimento de busca gulosa aleatória adaptativa (GRASP)
Hamiez e Hao (2002)	Busca dispersa (*scatter search*)
Paquete e Stutzle (2002)	Busca local iterada (ILS)
Avanthay et al. (2003)	Busca em vizinhanças variadas (VNS)
Lim et al. (2005)	*Squeaky Wheel Optimization* (SWO)
Salari e Eshghi (2005)	MMGC: Colônia de formigas + busca local
Caramia et al. (2006)	CHECKCOL: HCD com *checkpointing*
Trick e Yildiz (2007)	Busca em vizinhança grande (LNS)
Blöchliger e Zufferey (2008)	PARTIALCOL: Busca tabu
Dowsland e Thompson (2008)	Colônia de formigas + diversificação + busca tabu
Hertz et al. (2008)	Busca em espaços variados (VSS)
Malaguti et al. (2008)	MMT: Algoritmo genético + programação inteira
Mabrouk et al. (2009)	Algoritmo genético paralelo + busca tabu
Plumettaz et al. (2009)	Colônia de formigas + busca tabu
Lü e Hao (2010)	MACOL: Algoritmo genético + busca tabu
Chalupa (2011)	Algoritmo evolutivo multiagente / busca tabu pseudo-reativa
Fister et al. (2011)	HSA-DE: Algoritmo genético + busca local para 3-coloração
Hsu et al. (2011)	Otimização por nuvem de partículas (PSO)
Pahlavani e Eshghi (2011)	*Simulated annealing* + busca tabu
Titiloye e Crispin (2011)	*Quantum annealing*
Yesil et al. (2011)	*Simulated annealing* + *backtracking* + busca tabu
Abbasian e Mouhoub (2013)	Algoritmo genético paralelo hierárquico