Programação Linear Inteira aplicada a problemas de Coloração em Grafos

Ieremies Vieira da Fonseca Romero

Orientador: Rafael C. S. Schouery

Resumo

Uma coloração própria de vértices de um grafo consiste em uma alocação de cores tal que nenhum par de vértices adjacentes possuam a mesma cor. No problema de coloração de vértices, dado um grafo, procuramos uma coloração própria utilizando o menor número possível de cores. Além de ser um tradicional problema de teoria dos grafos, o problema se apresenta como NP-difícil e resolvê-lo de forma exata é extremamente relevante na atualidade. Tais aplicações vão desda alocação de recursos como registradores até o gerenciamento de eventos.

Soluções exatas para este problema utilizam técnicas como enumeração com *Branch-and-Bound* [10], programação dinâmica [9, 4] ou programação linear inteira. Nesse projeto, almejamos aplicar a última e utilizar algoritmos de *Branch-and-Price* para superar os resultados da literatura. Acreditamos que nosso trabalho se diferencia ao aplicar técnicas recentes para tais algoritmos, o que pode levar a uma melhora do estado-da-arte. Dessa forma, objetivamos resultados competitivos nas instâncias consolidadas na literatura, bem como possíveis novas formulações, cortes e estratégias para melhor resolvê-las.

1 Introdução

Dado um grafo não direcionado, uma **coloração própria** deste é uma alocação de cores tal que nenhum vértice seja adjacente a outro de mesma cor. O **Problema de Coloração de Vértices** (VCP) requer que encontremos uma coloração própria usando o menor número possível de cores. Decidir se um grafo possui uma coloração própria que usa até $k \geq 3$ cores é um problema NP-completo [12], de forma que o problema de otimização também é NP-difícil.

Muitas vezes, deseja-se alocar eventos ou recursos sem que hajam conflitos, seja na divisão de tarefas num cenário de fábricas ou escritórios, ou na organização de eventos. Assim, desejamos construir uma alocação de eventos a horários sem que, por exemplo, requeira que uma pessoa esteja em dois locais simultaneamente. Tal problema pode ser convertido a um de coloração transformando cada evento a um vértice no qual dois vértices estão ligados se há necessidade de ao menos uma pessoa estar em ambos eventos. Assim, numa solução, cada cor irá representar um horário para os eventos acontecerem.

Além dessa, como discutido por Malaguti e Toth [21], diversas são as aplicações do problema, como: agendamento [16], tabela de horários [26], alocação de registradores [6], comunicação de redes [5] e alocação de banda [11].

As aplicações acima deixam claro que encontrar uma boa (ou até ótima) solução para o problema é crucial. Cenários reais, muitas vezes, lidam com centenas de milhares de vértices, tornando necessário também resolvermos de forma rápida.

Neste projeto, utilizaremos de Programação Linear Inteira para resolver o problema de forma exata, como discutido na Seção 2. É importante mencionar que Programação Dinâmica também é comumente utilizada [9, 4], mas que vê apenas desenvolvimentos teóricos sem grande aplicabilidade prática no VCP. Já técnicas de enumeração baseado em *branch-and-bound* [3] possuem resultados práticos melhores que programação dinâmica, mas ainda são superados por técnicas de programação linear inteira. Por fim, recomendamos as seguintes revisões bibliográficas sobre o problema [21, 17, 14, 18].

Neste momento, é importante reforçar algumas notações. Seja $G = (V, E)$ um grafo simples não direcionado, denotamos como $n = |V|$ a quantidade de vértices e $m = |E|$ a quantidade de arestas. O número cromático de G, $\chi(G)$, é a menor quantidade de cores que pode constituir uma coloração do grafo G. Por fim, usaremos H como limitante superior ao número cromático. Uma cor h é dita válida para o vértice v se ela não foi atribuida a nenhum vértice da visinhança aberta de v.

Um conjunto $S \subseteq V$ é dito independente em G se, e somente se, não existem arestas em E que conectem dois elementos de S. Perceba que o conjunto de vértices coloridos com mesma cor deve ser um conjunto independente.

2 Metodologia

Programação Linear é uma técnica de otimização a partir da modelagem de **programas lineares**. Nestes, definimos uma função objetivo, a qual queremos maximizar ou minimizar com suas variáveis sujeitas a um conjunto de restrições lineares (equações ou inequações lineares) [7] . Um programa linear pode ser escrito da seguinte forma:

$$\begin{aligned} \text{minimize} \quad & cx \\ \text{sujeito a} \quad & Ax \geq b \\ & x \in \mathbb{R}_+ \end{aligned}$$

Para encontrar soluções viáveis com valores ótimos, conhecemos o algoritmo **simplex** que, apesar de ter complexidade exponencial, no caso médio executa em tempo polinomial [2]. Além deste, é importante mencionar os métodos de ponto interno [15] e elipsóide [1] como alternativas ao simplex.

Para alguns problemas, como o de coloração de grafos, não faz sentido falar em soluções fracionárias, afinal, não conseguimos designar "meia cor" a um vértice. Para isso, restringimos as variáveis aos inteiros, fazendo assim um **Programa Linear Inteiro**. Caso apenas um subconjunto das variáveis possuam a restrição de integralidade, chamamos esse programa de linear misto.

O que a princípio pode parecer uma pequena alteração, torna o problema computacionalmente muito mais complexo. Para encontramos boas soluções viáveis para esse tipo de programa, algoritmos como o simplex não são o suficiente. Para isso, utilizamos técnicas como *branch-and-bound*, que consiste em dividir o problema em subproblemas menores e, durante o processo, encontrar limitantes que permitam diminuir o espaço de busca.

Por fim, é interessante nesse momento traçar o paralelo com a **Combinatória Poliédrica**. Definimos um **poliedro** como um subconjunto $P \subseteq \mathbb{R}^n$ tal que ele possa ser descrito por meio de inequações lineares em tal espaço, similar aos programas lineares apresentados acima. Caso esse poliedro seja limitado, o chamamos de **politopo**. Dizemos que um subconjunto do poliedro definido por uma inequação é uma **face**. Se esta não for igual ao poliedro inteiro, dizemos ser própria e, caso também não seja vazia, dizemos ser não-trivial.

A seguir, apresentamos as metodologias e técnicas que nos são relevantes para resolver o problema de forma exata. Em geral, as instâncias usadas para os resultados de experimentos computacionais são as apresentadas em [8], que foram atualizadas com o passar dos anos.

Proposto por Mehrotra e Trick [22], uma forma de entender o problema é imaginá-lo como um **problema de cobertura de conjuntos** (SC) onde os conjuntos disponíveis são os conjuntos independentes dos vértices.

Assim, seja S a família de conjuntos impendentes do grafo G, a variável binária x_s representa se o conjunto $s \in S$ está sendo usado ou não na solução. Nossa formulação então se dá por:

$$\begin{aligned} \text{(SC)} \quad \text{minimize} \quad & \sum_{s \in S} x_s & & \\ \text{sujeito a} \quad & \sum_{s \in S : i \in s} x_s \geq 1 & i \in V & \qquad (1) \\ & x_s \in \{0,1\} & s \in S & \end{aligned}$$

O primeiro conjunto de restrições garante que todos os vértices de V estão contidos em algum conjunto independente escolhido. Apesar de poucas restrições, essa formulação sofre de ter um número exponencial de variáveis, o que a torna impraticável de implementá-la em resolvedores convencionais como *Gurobi*.

Por tal motivo, Mehrotra e Trick [22] propuseram um algoritmo de *branch-and-price* baseado nesta modelagem. Essa abordagem permite lidar com um número exponencial de variáveis, já que iniciamos com um subconjunto destas e adicionamos com o passar do desenvolvimento da árvore de *branch-and-bound*.

Neste algoritmo, a decisão de qual variável será adicionada é feita pelo subproblema de geração de coluna carac-

teriza um **Problema de Conjunto Independente de Peso Máximo**:

$$\begin{aligned} &\text{maximize} \sum_{i \in V} \pi_i z_i \\ &\text{sujeito a} \quad z_i + z_j \leq 1 \qquad (i,j) \in E \\ &\qquad\qquad z_i \in \{0,1\} \quad i \in V \end{aligned}$$

onde z_i é uma variável binária que indica se o vértice i está incluso no conjunto independente e π_i é o valor ótimo da variável dual associado à restrição (1). Tal problema pode ser resolvido de forma heurística para encontrar a coluna de custo reduzido com valor negativo. Em caso de soluções fracionárias, os autores sugerem uma estratégia que garante que os subproblemas continuam a ser de coloração de vértices e apenas requer que o grafo original seja alterado.

Além disso, Malaguti, Monaci e Toth [20] propõem meta-heurísticas para inicialização e geração de colunas bem como novos esquemas de branching.

Já Morrison et al. [24] apresentam outra forma de realizar a ramificação do processo de *branch-and-bound*. Tradicionalmente, esta é feita dividindo apenas em dois subproblemas, mas os autores propõem uma abrangente quantidade de subproblemas a cada ramificação com intuito de impedir que cheguemos a subgrafos onde o problema de precificação é mais difícil. Mais uma vez, resultados computacionais provam que a ideia é competitiva com as anteriores.

Recentemente, Morrison, Sewell e Jacobson [23] propuseram uma estratégia utilizando diagramas de decisão para agilizar a resolução do problema de precificação que, segundo seus experimentos, possui bons resultados práticos bem como fornece limites inferiores durante o processo.

Por fim, Hoeve [13] também técnicas de diagrama de decisão. Nesta, representamos cada atribuição por um arco que liga o "estado da solução" antes e depois de tomá-la. Mais formalmente, para um problema P definido por um conjunto de variáveis ordenadas $X = \{x_1, x_2, \ldots, x_n\}$, construímos um diagrama de decisão, um grafo simples acíclico de $n+1$ níveis. O primeiro destes, possui apenas um vértice, r, chamado raiz, assim como o último com o vértice t. O nível i é um conjunto de nós associados à variável x_i onde cada um destes possuem arcos para vértices do nível $i+1$ e tais arcos possuem etiquetas, 0 ou 1, correspondendo da variável associada.

Hoeve [13] utiliza a formulação de conjuntos independentes para aplicar tal técnica, onde cada variável indica o uso, ou não, de um vértice em um conjunto. Se conseguíssemos achar um diagrama de decisão que correspondesse exatamente ao problema de coloração, poderíamos resolver o problema por meio de uma formulação de fluxo, na qual cada $\{r,t\}-$caminho seria um conjunto independente e, como demostrado pelos autores, a função objetivo seria nosso número cromático. Infelizmente, tal diagrama pode conter um número exponencial de nós, o que requer técnicas mais sofisticadas. Além disso, quando a solução não é exata, a qualidade da solução depende da ordenação das variáveis.

Os autores relatam ter obtido resultados competitivos com outros estados-da-arte utilizando estratégias específicas da literatura de diagrama de decisões para resolver as dificuldades apresentadas acima.

3 Objetivos

Recentemente, houve avanços significativos na abordagem de branch-and-price para o problema de coloração, resultando em vários trabalhos publicados para outros problemas que utilizam e melhoram essa técnica. Um exemplo notável é o trabalho de Lima, Iori e Miyazawa [19], que apresenta técnicas fortes que melhoram o estado da arte para problemas como o Problema de Empacotamento.

Os autores propõem um modelo baseado em fluxo de arcos para auxiliar em algoritmos de geração de colunas. Eles também comentam que qualquer problema de cobertura de conjunto consegue ser transformado em um problema de fluxo, o que indica que suas técnicas podem ser relevantes para o nosso problema. Além disso, eles utilizam **fixação de variáveis**, técnica na qual é possível provar que algumas variáveis nunca poderão entrar no modelo de *branch-and-pricing* e tornar a solução melhor. A dificuldade reside em encontrar uma solução dual viável que possibilite computar o custo reduzido, necessário para provar esta afirmação. Um importante desenvolvimento proposto pelos autores é justamente um modelo linear capaz de encontrar eficientemente uma solução, mesmo que não seja ótima (o que, como argumentado por eles, é ainda mais eficiênte).

Outro bom indicador do que pretendemos fazer é a semelhança de bons resultados recentes como [13] que utilizam ideias muito similares.

Técnicas interessantes também foram propostas por Pessoa, Sadykov e Uchoa [25] que apresentam um modelo genérico para resolver problema de roteamento. Quando adiciona-se um corte no *branch-cut-and-price*, isso corresponde a variáveis no dual o que dificulta o subproblema de geração de colunas. Os autores utilizam cortes de rank-1 com memória limitada para melhorar tal processo. Além disso, eles usam **propagação de etiquetas**, técnica comum na resolução de problemas de precificação e que pode ser interessante para o nosso problema.

Nosso principal objetivo é aplicarmos tais novas tecnologias ao problema de coloração. Além disso, estudaremos a possibilidade de novos cortes e limitantes para as formulações. Por fim, como apresentado no começo desse projeto, diversos são os problemas similares ou generalizações bem como as aplicações de tais. Assim, podemos também voltar nossas técnicas e implementações a tais variantes.

4 Cronograma

O projeto está dividido em etapas distintas, cada uma com um objetivo específico. A primeira fase é de revisão bibliográfica, onde serão dedicados 2 trimestres para estudar a literatura existente sobre o assunto. Reforçamos que, apesar deste ser o foco da primeira etapa, a revisão da bibliografia será permanente. Em seguida, serão dedicados 2 trimestres para implementar o estado da arte, onde aplicaremos os conhecimentos adquiridos na fase anterior. A próxima etapa é o desenvolvimento de novos algoritmos, bem como a implementação e experimentação de tais soluções, onde serão dedicados os trimestres seguintes. Por último, nos dedicaremos a escrever o relatório final, que apresentará as conclusões e resultados do projeto.

Por fim, o aluno irá, como requerido pela instituição, cursar as três matérias de pós-graduação restantes no primeiro ano, defender seu exame de qualificação de mestrado (EQM) na metade de 2023 e realizar o Programa de Estágio em Docência (PED) no segundo semestre de 2023.

Refletimos o cronograma planejado na Tabela 1.

Tabela 1: Cronograma trimestral para este projeto de mestrado, iniciando em Março de 2023.

Atividade	Jun	Set	Dez	Mar	Jun	Set	Dez	Mar
Disciplinas	•	•						
EQM		•						
PED			•					
Revi. Biblio.	•	•	•					
Implem. estado-da-arte			•	•				
Desenvolvimento			•	•	•	•	•	•
Escrita							•	•

Referências

[1] Robert G. Bland, Donald Goldfarb e Michael J. Todd. "Feature Article—The Ellipsoid Method: A Survey". Em: *Operations Research* 29.6 (1981), pp. 1039–1091. DOI: 10.1287/opre.29.6.1039.

[2] Karl Heinz Borgwardt. *The Simplex Method - Algorithms and Combinatorics*. Springer, 1986. ISBN: 9783540170969.

[3] Daniel Brélaz. "New methods to color the vertices of a graph". Em: *Communications of the ACM* 22.4 (1979), pp. 251–256. DOI: 10.1145/359094.359101.

[4] Jesper Makholm Byskov. "Chromatic Number in Time $O(2.4023^n)$ Using Maximal Independent Sets". Em: *BRICS Report Series* 9.45 (2002). DOI: 10.7146/brics.v9i45.21760.

[5] Alberto Caprara et al. "Passenger Railway Optimization". Em: *Discrete optimization* (2007). DOI: 10.1016/S0927-0507(06)14003-7.

[6] Fred C. Chow e John L. Hennessy. "The priority-based coloring approach to register allocation". Em: *ACM Transactions on Programming Languages and Systems* 12.4 (1990), pp. 501–536. DOI: 10.1145/88616.88621.

[7] Vašek Chvátal. *Linear Programming*. W. H. Freeman, 1983, p. 478. 500 pp. ISBN: 978-0-7167-1587-0.

[8] DIMACS. *Graph Coloring Instances*. URL: https://mat.tepper.cmu.edu/COLOR/instances.html (acedido em 14/12/2022).

[9] David Eppstein. "Small Maximal Independent Sets and Faster Exact Graph Coloring". Em: *Graph Algorithms and Applications 4*. World Scientific, 2006, pp. 131–140. DOI: 10.1142/9789812773296_0006.

[10] Fabio Furini, Virginie Gabrel e Ian-Christopher Ternier. "An Improved DSATUR-Based Branch-and-Bound Algorithm for the Vertex Coloring Problem". Em: *Networks* 69.1 (2017), pp. 124–141. ISSN: 0028-3045. DOI: 10.1002/net.21716. (Acedido em 13/12/2022).

[11] A. Gamst. "Some lower bounds for a class of frequency assignment problems". Em: *IEEE Transactions on Vehicular Technology* (1986). DOI: 10.1109/T-VT.1986.24063.

[12] Michael Randolph Garey e David S. Johnson. "Computers and Intractability: A Guide to the Theory of NP-Completeness". Em: (1979).

[13] Willem-Jan van Hoeve. "Graph coloring with decision diagrams". Em: *Mathematical Programming* 192.1-2 (2021), pp. 631–674. DOI: 10.1007/s10107-021-01662-x.

[14] Thore Husfeldt. *Graph colouring algorithms*. 2015. DOI: 10.48550/ARXIV.1505.05825.

[15] N. Karmarkar. "A new polynomial-time algorithm for linear programming". Em: *Proceedings of the sixteenth annual ACM symposium on Theory of computing - STOC '84*. ACM Press, 1984. DOI: 10.1145/800057.808695.

[16] F. Leighton. "A Graph Coloring Algorithm for Large Scheduling Problems." Em: *Journal of research of the National Bureau of Standards* (1979). DOI: 10.6028/JRES.084.024.

[17] R. M. R. Lewis. *Guide to Graph Colouring Algorithms and Applications. Algorithms and Applications*. Springer, 2015, p. 253. ISBN: 9783319257303.

[18] Alane Marie De Lima e Renato Carmo. "Exact Algorithms for the Graph Coloring Problem". Em: *Revista de Informática Teórica e Aplicada* 25.4 (2018), p. 57. DOI: 10.22456/2175-2745.80721.

[19] Vinícius Loti de Lima, Manuel Iori e Flávio Keidi Miyazawa. "Exact solution of network flow models with strong relaxations". Em: *Mathematical Programming* (2022). DOI: 10.1007/s10107-022-01785-9.

[20] Enrico Malaguti, Michele Monaci e Paolo Toth. "An Exact Approach for the Vertex Coloring Problem". Em: *Discrete Optimization* 8.2 (2011), pp. 174–190. ISSN: 1572-5286. DOI: 10.1016/j.disopt.2010.07.005.

[21] Enrico Malaguti e Paolo Toth. "A Survey on Vertex Coloring Problems". Em: *International Transactions in Operational Research* 17.1 (2010), pp. 1–34. ISSN: 0969-6016. DOI: 10.1111/j.1475-3995.2009.00696.x. (Acedido em 13/12/2022).

[22] Anuj Mehrotra e Michael A. Trick. "A Column Generation Approach for Graph Coloring". Em: *INFORMS Journal on Computing* 8.4 (1996), pp. 344–354. ISSN: 1091-9856. DOI: 10.1287/ijoc.8.4.344. (Acedido em 13/12/2022).

[23] David R. Morrison, Edward C. Sewell e Sheldon H. Jacobson. "Solving the Pricing Problem in a Branch-and-Price Algorithm for Graph Coloring Using Zero-Suppressed Binary Decision Diagrams". Em: *INFORMS Journal on Computing* 28.1 (2016), pp. 67–82. ISSN: 1091-9856. DOI: 10.1287/ijoc.2015.0667. (Acedido em 13/12/2022).

[24] David R. Morrison et al. "A Wide Branching Strategy for the Graph Coloring Problem". Em: *Informs Journal on Computing* 26.4 (2014), pp. 704–717. DOI: 10.1287/IJOC.2014.0593.

[25] Artur Pessoa, Ruslan Sadykov e Eduardo Uchoa. "Solving Bin Packing Problems Using VRPSolver Models". Em: *Operations Research Forum* 2.2 (2021). DOI: 10.1007/s43069-020-00047-8.

[26] D. de Werra. "An introduction to timetabling". Em: *European Journal of Operational Research* (1985). DOI: 10.1016/0377-2217(85)90167-5.