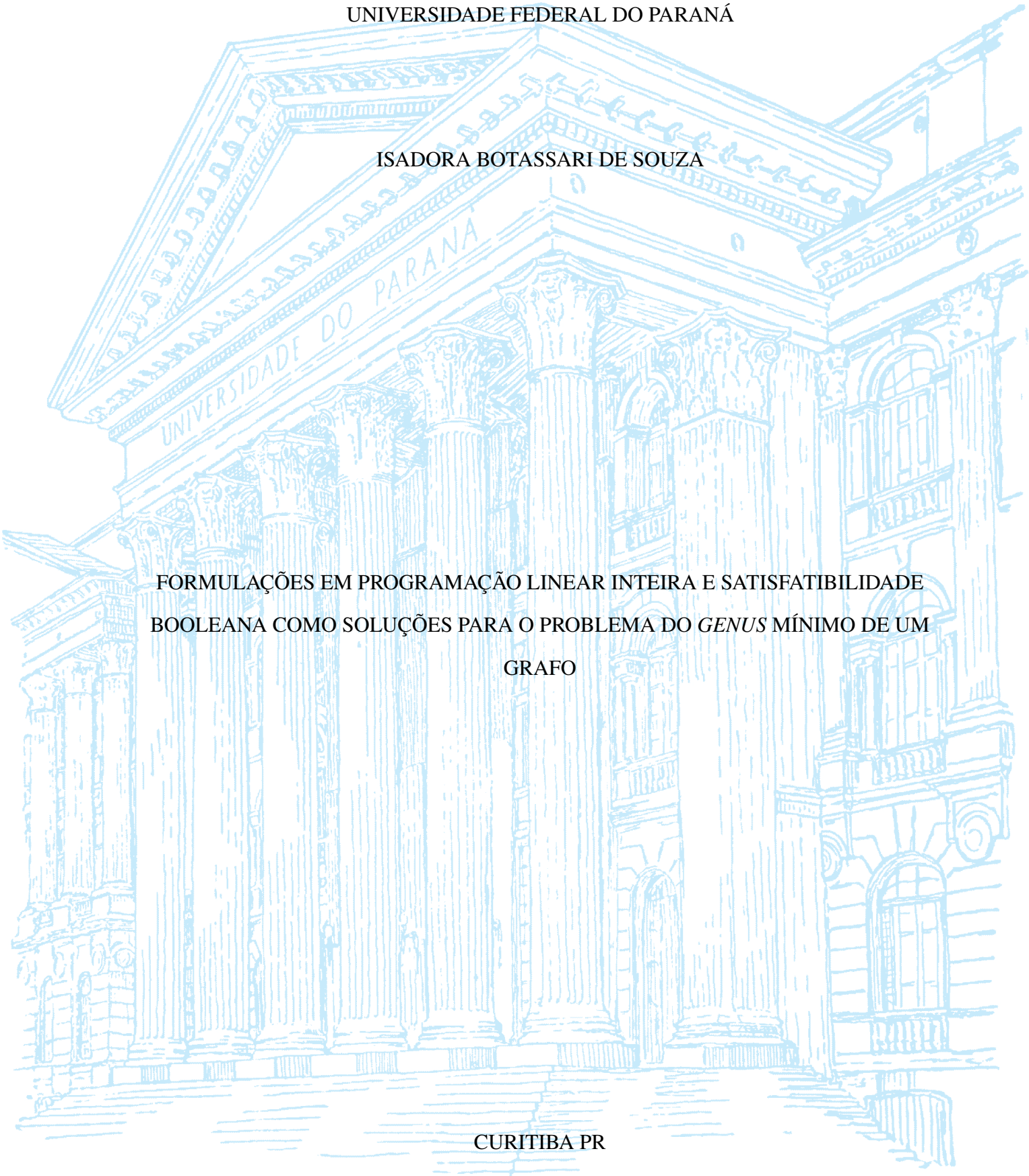
UNIVERSIDADE FEDERAL DO PARANÁ

ISADORA BOTASSARI DE SOUZA



FORMULAÇÕES EM PROGRAMAÇÃO LINEAR INTEIRA E SATISFATIBILIDADE BOOLEANA COMO SOLUÇÕES PARA O PROBLEMA DO *GENUS* MÍNIMO DE UM GRAFO

CURITIBA PR

2025

ISADORA BOTASSARI DE SOUZA

FORMULAÇÕES EM PROGRAMAÇÃO LINEAR INTEIRA E SATISFATIBILIDADE BOOLEANA COMO SOLUÇÕES PARA O PROBLEMA DO *GENUS* MÍNIMO DE UM GRAFO

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Computação*.

Orientador: André Luiz Pires Guedes.

CURITIBA PR

2025

"Este mundo é o bastante — tem de ser. É a maior e mais gentil combinação que os átomos foram capazes de formar."

AGRADECIMENTOS

Agradeço à minha mãe, minha avó e minhas tias pelo amor, pelo cuidado e pela força que sempre me deram. Vocês são as raízes de tudo que eu sou hoje.

Agradeço à minha namorada, Izabel, por ser minha companheira, meu lar e o motivo do meu sorriso. Obrigada por sempre me ouvir, sempre me apoiar e por ter unido sua vida à minha.

Agradeço a todos os amigos que me acompanharam nessa jornada pela universidade, próximos ou não. Cada momento, do mais sério ao mais bobo, foi precioso para mim. Esta etapa da minha vida tem o rosto de todos vocês.

Agradeço ao meu orientador, prof. André Guedes, pela orientação, pelas aulas, pelo apoio teórico e por ter me inspirado a me aprofundar nesta magnífica área da Computação.

Por fim, estendo minha gratidão a todos que contribuíram com minha formação nesses anos.

RESUMO

O Problema do *Genus* Mínimo de um Grafo G consiste em determinar se um grafo G pode ser imerso em uma superfície orientável de *genus* K sem que haja cruzamento entre suas arestas. O conceito de *genus* vem da Topologia e é uma importante característica topológica que indica a quantidade de "buracos" ou "alças" presentes numa superfície orientável; por exemplo, a esfera possui *genus* 0, e o toro, *genus* 1. Para grafos, no caso de $K = 0$, o problema corresponde ao teste de planaridade, que pode ser resolvido em tempo polinomial. Contudo, para o caso geral, Thomassen (1989) demonstrou que se trata de um problema $N\mathcal{P}$-completo, ou seja, computacionalmente intratável. Desde então, variados algoritmos foram propostos para resolver versões mais restritas do problema, incluindo algoritmos de parâmetro fixo e algoritmos de aproximação. Entretanto, conforme revisão realizada por Myrvold e Kocay (2011), muitos desses algoritmos apresentam alta complexidade de implementação ou estão incorretos. Diante desse cenário, Beyer et al. (2016) propuseram modelagens em Programação Linear Inteira (PLI) e Satisfatibilidade (SAT). As soluções são embasadas na característica de Euler e na maximização do número de faces de uma imersão de grafo válida. O objetivo principal do trabalho consiste em aprofundar a análise das regras e restrições envolvidas em cada modelo, examinando o papel de cada regra e restrição e como contribuem para o cálculo da imersão do grafo de entrada. Através da revisão do histórico do problema e da análise das formulações, foi possível evidenciar a alta complexidade do problema e compreender como essas modelagens, apoiadas em fundamentos geométricos, são capazes de oferecer uma solução prática (mesmo que limitada) para um problema notoriamente difícil.

Palavras-chave: Grafos. Genus. Programação Linear Inteira. Satisfatibilidade booleana.

ABSTRACT

The *Minimum Genus Problem* for a graph G consists of determining whether the graph G can be embedded in an orientable surface of *genus* K without any edge crossings. The concept of *genus* comes from Topology and is an important topological invariant that indicates the number of "holes" or "handles" in an orientable surface; for example, the sphere has *genus* 0, and the torus has *genus* 1. For graphs, when $K = 0$, the problem corresponds to planarity testing, which can be solved in polynomial time. However, in the general case, Thomassen (1989) proved that the problem is $N\mathcal{P}$-complete, meaning it is computationally intractable. Since then, several algorithms have been proposed to solve more restricted versions of the problem, including fixed-parameter algorithms and approximation algorithms. Nevertheless, as reviewed by Myrvold and Kocay (2011), many of these algorithms are either highly complex to implement or contain errors. In light of this scenario, Beyer et al. (2016) proposed formulations using Integer Linear Programming (ILP) and Satisfiability (SAT). These solutions are based on Euler's characteristic and the maximization of the number of faces in a valid graph embedding. The main objective of this work is to deepen the analysis of the rules and constraints involved in each model, examining the role of each rule and how they contribute to computing a valid embedding of the input graph. Through a review of the problem's background and an analysis of the formulations, it was possible to highlight the high complexity of the problem and to understand how these models, supported by geometric foundations, can offer a practical (although limited) solution to a notoriously difficult problem.

Keywords: Graphs. Genus. Integer Linear Programming. Boolean Satisfiability.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

$\delta^-(v)$	vizinhança de entrada do vértice v
$\delta^+(v)$	vizinhança de saída do vértice v
$\gamma(G)$	valor do *genus* de um grafo G

SUMÁRIO

1 INTRODUÇÃO

O *genus* de uma superfície orientável é um número não-negativo que representa a quantidade de "buracos" ou "alças" da superfície. Por exemplo, uma esfera tem *genus* 0, enquanto um toro tem *genus* 1. Aplicado à Teoria dos Grafos, o *genus* mínimo de um grafo G é definido como o menor *genus* de uma superfície na qual G pode ser imerso sem cruzamento de arestas. O Problema do *Genus* Mínimo consiste em responder se é possível realizar a imersão do grafo G numa superfície de *genus* K.

O problema foi descrito por Garey e Johnson (1979) como, então, um dos principais problemas em aberto na computação. Na época, já era conhecido o fato de que, para casos de $K = 0$, trata-se do problema de imersão de grafos planares e, portanto, possui solução em tempo polinomial. Thomassen (1989) provou que, para o caso geral, esse problema é $N\mathcal{P}$-completo. Ainda, em 1996, o mesmo autor provou que determinar o *genus* de um grafo cúbico (3-regular) também é $N\mathcal{P}$-completo.

Desde a prova da $N\mathcal{P}$-completude do problema, foram propostos diversos algoritmos de tempo polinomial que respondem instâncias mais reduzidas do problema; por exemplo, os algoritmos de parâmetro fixo de Filotti et al. (1979) e Djidjev e Reif (1991), ou então, os algoritmos de aproximação propostos por Chekuri e Sidiropoulos (2016). Entretanto, uma revisão feita por Myrvold e Kocay (2011) demonstrou que os principais algoritmos conhecidos e referenciados para o problema apresentavam falhas.

Diante da falta de soluções práticas para o problema do *genus* mínimo, Beyer et al. (2016) propuseram modelagens do problema em Programação Linear Inteira e Satisfatibilidade. As formulações são baseadas em encontrar uma imersão do grafo G que obedeça à característica de Euler, buscando maximizar o número de faces da imersão (no caso da formulação PLI), ou então, encontrar uma imersão válida com f faces (no caso da formulação SAT). O objetivo deste trabalho é apresentar as formulações propostas por Beyer et al. e aprofundar a explicação das restrições e regras que as compõem.

No Capítulo 2, são apresentados os fundamentos necessários para a compreensão do problema: o *genus* de uma superfície, a imersão de um grafo e o *genus* de um grafo. No Capítulo 3, o problema é descrito formalmente e é exposto um histórico dos algoritmos propostos para sua resolução. O Capítulo 4 está organizado da seguinte maneira: na Seção 4.1, são expostos conceitos importantes para a compreensão das formulações; na Seção 4.2, é apresentada a formulação PLI; na Seção 4.3, é apresentada a formulação SAT. As Seções 4.4 e 4.5 focam em possíveis otimizações e em métodos que podem ser empregados para obter um número polinomial de restrições/regras (em vez de exponencial).

2 FUNDAMENTOS

Neste capítulo, são apresentados os fundamentos de Topologia e Teoria dos Grafos necessários para a leitura e compreensão deste trabalho.

2.1 GENUS DE UMA SUPERFÍCIE

São apresentadas as seguintes definições a respeito do conceito de *genus* de uma superfície. De acordo com Popescu-Pampu (2016):

> O *genus* de uma superfície compacta, conexa e orientável sem fronteira é o número máximo de círculos disjuntos dois a dois que podem ser desenhados sobre a superfície, de modo que o complemento permaneça conexo.

De acordo com Gross e Tucker (1987):

> Se S é uma superfície orientável, o *genus* pode ser definido pelo número de alças que devem ser adicionadas a uma esfera para a obtenção de seu tipo homeomorfo (topologicamente equivalente). Assim, o *genus* de uma superfície orientável S_g é g.

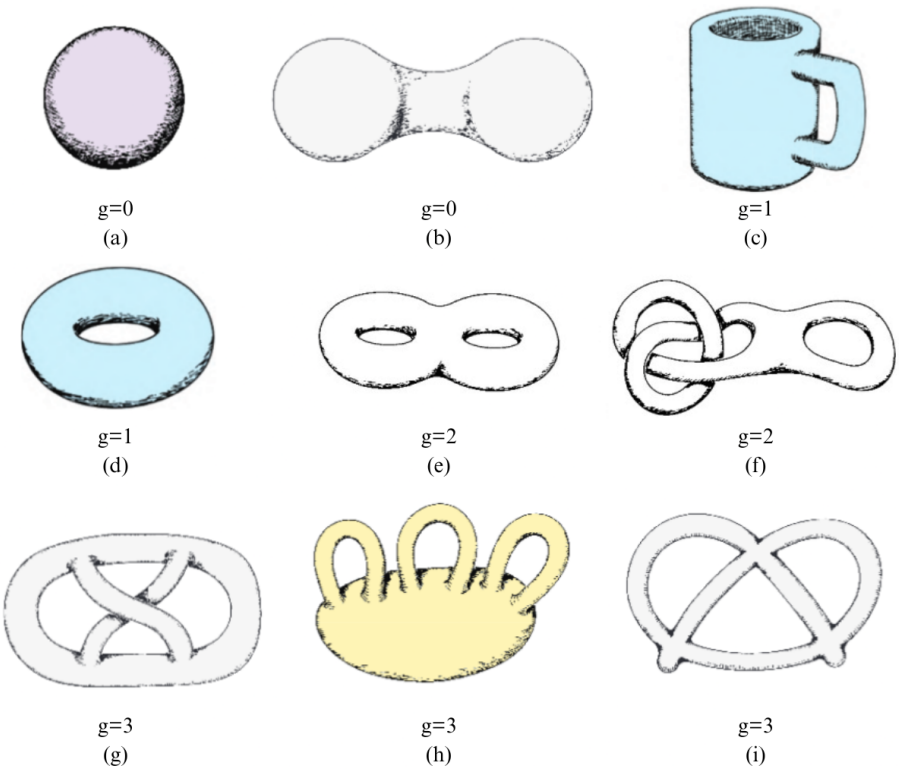


Figura 2.1: Superfícies com diferentes *genus* (indicado por g). Fonte: Weeks (2002)

De forma didática, o valor do *genus* se refere ao número de "buracos" numa superfície orientável. A Figura 2.1 demonstra essa intuição com ilustrações de diversas superfícies; dentre

os itens, estão presentes uma esfera (a), toro (d), um toro duplo (e, f) e um toro triplo (h, i). As superfícies ilustradas se assemelham a objetos esburacados.

Os itens (c), (g) e (h) demonstram uma intuição alternativa: a de alças conectadas a uma esfera. Cada alça adicionada a uma esfera aumenta o valor do *genus* em 1, criando uma nova "ponte" na superfície.

O valor do *genus* é um invariante topológico, ou seja, permanece inalterado sob deformações contínuas da superfície. As figuras (c) e (d), apesar da aparência distinta, possuem o mesmo *genus* e, portanto, são topologicamente equivalentes. O mesmo se aplica para as figuras (a) e (b).

2.2 IMERSÃO DE UM GRAFO

A ideia de imersão de um grafo consiste no desenho de um grafo G numa superfície S de forma que as arestas de G não se cruzem. De acordo com Gross e Tucker (1987):

> A imersão de um grafo é definida por uma função contínua e injetora de uma representação topológica do grafo na superfície.

Para os propósitos deste trabalho, a terminologia será estendida de forma que a imagem da representação topológica também seja referenciada pelo termo "grafo".

2.3 GENUS DE UM GRAFO

A respeito da definição do *genus* de um grafo, é adotada a descrição dada por Gross e Tucker (1987):

> O *genus* de um grafo G, indicado por $\gamma(G)$, é definido pelo menor número g de forma que o grafo G possa ser imerso na superfície orientável S_g.

A característica do *genus* de um grafo pode ser usada como critério de definição de classes de grafos. Um dos conceitos que auxilia a caracterizar essas classes é o de grafo proibido. Neste caso, trata-se do conjunto de grafos que não podem ser imersos numa superfície com um certo *genus* qualquer. Grafos com essa característica também podem ser chamados de obstruções.

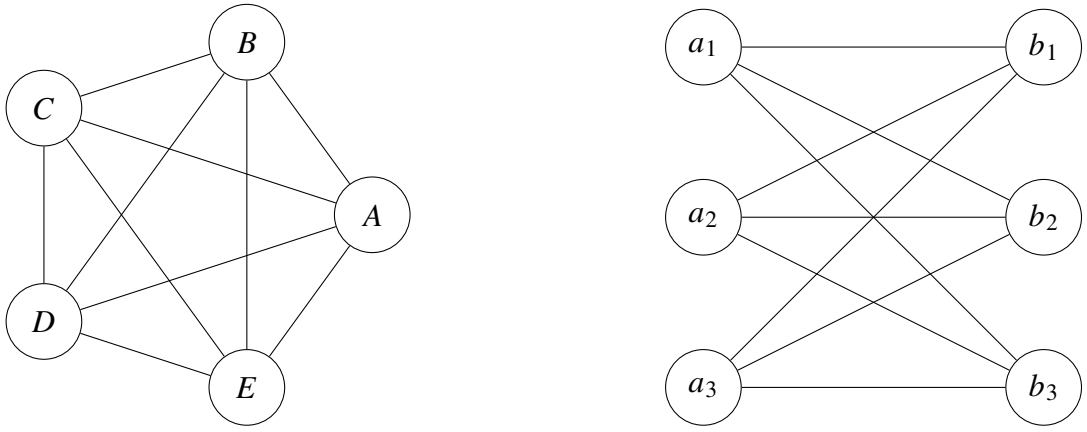
No caso dos grafos planares — grafos que podem ser imersos numa superfície de *genus* 0 —, há dois tipos de subgrafos proibidos: o K_5 e o $K_{3,3}$ (Figura 2.2); subdivisões desses grafos também são proibidas. Essas famílias são chamadas de grafos de Kuratowski, em razão do autor que provou a impossibilidade da presença desses subgrafos num grafo planar. Desta forma, caso estes subgrafos possam ser induzidos de alguma forma no grafo G, é impossível que G seja imerso numa superfície de *genus* 0.

No entanto, G poderia ser imerso numa superfície de *genus* 1 (classe dos grafos toroidais), desde que G também não contivesse obstruções de grafos de *genus* 1 (Figura 2.3). Este é um exemplo de como, conforme o *genus* aumenta, o conjunto de subgrafos proibidos se torna menor, assim como mais específico, pois existem menos configurações possíveis de obstruções. Essa relação foi provada através do Teorema de Robertson-Seymour, também chamado de teorema dos menores de um grafo.

O conceito de menor de um grafo G considera três operações (Lovász, 2005):

a) remoção de uma aresta;

b) contração de uma aresta;



(a) Grafo completo K_5 (b) Grafo bipartido completo $K_{3,3}$

Figura 2.2: Grafos de Kuratowski.

c) remoção de um vértice isolado.

Um grafo G' que pode ser obtido a partir de uma sequência de aplicações dessas operações em G é considerado um menor de G. Um grafo que é isomorfo em relação a um menor de G também é um menor de G.

Portanto, o Teorema de Robertson-Seymour afirma que grafos não direcionados, parcialmente ordenados pela relação dos menores de grafo, formam uma bem-quasi-ordem. Como consequência, é possível afirmar que a obstrução representada pelos grafos de Kuratowski (que consistem em "menores de grafos proibidos") se generaliza para superfícies de *genus* superior, ou seja, cada superfície possui um conjunto finito de grafos proibidos de acordo com seu *genus*, e esses conjuntos apresentam certa ordem entre si.

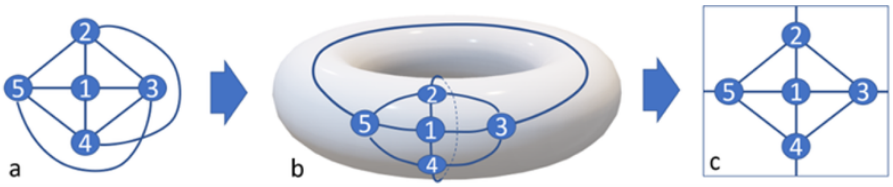


Figura 2.3: O grafo K_5 imerso numa superfície de *genus* 1. Fonte: Miller et al. (2024)

Formalmente, para um número inteiro fixo $g \geq 0$, há uma lista de grafos $L(g)$ com a seguinte propriedade: um grafo G pode ser imerso em uma superfície de *genus* g se, e somente se, ele não contiver, como menor, nenhum dos grafos da lista L.

Para a classe dos grafos toroidais, o conjunto de subgrafos proibidos ainda não foi completamente definido. Naturalmente, isso também é válido para classes com *genus* mais altos. Gagarin et al. (2009) descreveram os conjuntos de menores de grafos e obstruções para grafos toroidais sem a presença do $K_{3,3}$.

Historicamente, um problema que é próximo do problema do *genus* de um grafo é o de coloração de mapas, mais especificamente, a versão generalizada para superfícies de *genus* superior a 0 (um mapa é uma superfície de *genus* 0). Uma forma de visualizar essa conexão é considerar um grafo completo como representação do "pior caso" de um problema de coloração de elementos numa superfície de *genus* k: uma instância em que todos os elementos são adjacentes.

Uma importante contribuição para essa questão foi a Conjectura de Heawood (ou então, Conjectura de Coloração de Mapas), proposta por Heawood (1890), a qual fornece um limite

inferior para o número de cores necessário para colorir um mapa numa superfície de *genus* qualquer. Como parte da prova da conjectura, Ringel e Youngs (1968) estabeleceram que o *genus* de um grafo completo K_n pode ser dado por:

$$\gamma(K_n) = \left\lceil \frac{(n-3)(n-4)}{12} \right\rceil, \quad n \geq 3.$$

Dessa forma, uma outra classe de grafos que possui *genus* conhecido é a dos grafos completos (K_n). Por exemplo, para $\gamma(K_5)$, $\gamma(K_6)$ e $\gamma(K_7)$, o valor é igual: 1. A partir de $n = 8$, $\gamma(K_8) = 2$, e o mesmo vale para $\gamma(K_9)$. Portanto, sabe-se que o K_8 e o K_9 exigem superfícies com *genus* maior que 1 para serem imersos.

3 O PROBLEMA DO GENUS MÍNIMO DE UM GRAFO

Neste capítulo, é apresentada a descrição formal do problema do *genus* mínimo de um grafo, sua complexidade computacional e o histórico de algoritmos e abordagens existentes até então.

3.1 DESCRIÇÃO DO PROBLEMA

O problema do *genus* mínimo de um grafo foi descrito por Garey e Johnson (1979) como um dos vários problemas da computação em aberto na data de publicação.

Instância: um grafo $G = (V, E)$ e um número inteiro não-negativo K.

Problema: É possível realizar a imersão do grafo G numa superfície de *genus* K de forma que nenhuma das arestas se cruzem?

Para $K = 0$, o problema possui solução polinomial, uma vez que se trata do problema de imersão de grafos planares.

Thomassen (1989) provou que o caso geral do problema é $N\mathcal{P}$-completo. Posteriormente, o mesmo autor examinou também a complexidade do problema para grafos cúbicos. No artigo, Thomassen menciona que a complexidade de um problema combinatório, quando restrito ao caso cúbico, pode se diferenciar do caso geral. Considerando a importância do grau dos vértices na prova de 1989, essa restrição apresentaria a possibilidade de ser uma versão mais simples e tratável do problema. Entretanto, foi provado que, mesmo para grafos cúbicos, o problema do *genus* mínimo permanece $N\mathcal{P}$-completo (Thomassen, 1997).

3.2 ABORDAGENS

Diante da complexidade do caso geral, ao longo das décadas, foram levantadas diversas abordagens para tratar do problema — em particular, propondo soluções para versões mais específicas dele.

Considerando o teorema de Robertson-Seymour (mencionado em seções anteriores), é sabido que o problema do *genus* mínimo é tratável por parâmetro fixo, uma vez que os conjuntos de grafos menores proibidos são finitos e, consequentemente, seria possível testar a presença desses grafos para determinar o *genus* do grafo. O principal obstáculo dessa abordagem é o fato de que, exceto pelos grafos planares, as listas de obstruções da classe toroidal e das classes acima não são completamente conhecidas.

Um dos primeiros e mais conhecidos algoritmos descritos para esse problema foi proposto por Filotti et al. (1979). A base do algoritmo consiste em fixar o parâmetro k e responder se, para uma superfície de *genus* k (informado na entrada), é possível realizar a imersão do grafo G. Um algoritmo similar foi descrito por Djidjev e Reif (1991) o qual possui a mesma entrada e responde a mesma questão; além disso, também fornece um subgrafo presente em G que seja homeomorfo a um grafo proibido em superfícies de *genus* $g - 1$. Os dois algoritmos descritos possuem complexidade polinomial.

Entretanto, Myrvold e Kocay (2011) demonstraram que esses dois algoritmos possuem falhas fundamentais em suas premissas. A conclusão do artigo aponta que, à época de sua publicação, não existiam algoritmos corretos capazes de lidar com o caso toroidal — o caso mais simples, à parte do caso dos planares — do problema, em especial, para os grafos que contêm o $K_{3,3}$.

Mohar et al. (1995) propuseram um algoritmo linear capaz de encontrar uma imersão de G em S, ou então, encontrar um subgrafo em G que seja homeomorfo a um grafo menor proibido em S. A corretude do algoritmo não foi contestada, no entanto, Myrvold e Kocay (2011) e Beyer et al. (2016) apontam que a implementação dessa solução é muito complexa, ao ponto de que esse algoritmo, até o presente momento, não foi implementado de forma correta.

Por fim, houve também tentativas de encontrar algoritmos de aproximação para esse problema, assim como de definir o fator de aproximação. De acordo com Chekuri e Sidiropoulos (2016), o fator de aproximação do problema conhecido até então era de $O(\sqrt{n})$, por mais que a possibilidade de existência de um fator de $O(1)$ não tenha sido totalmente descartada. Os autores apresentam um algoritmo que, dentre outras consequências, implica num fator de aproximação de $O(n^{\frac{1}{2}-\alpha})$ para alguma constante $\alpha > 0$.

Por exemplo, suponha um grafo G com número de vértices $n = 100$ e, *genus* mínimo OPT = 5 e uma constante α. Um algoritmo com fator de aproximação $O(\sqrt{n})$ garante que a resposta dada seja, no máximo, $\sqrt{n}$ OPT, ou seja, neste exemplo, 50. Um caso mais simples, com OPT = 1, teria como resposta um valor, no máximo, até 10. Desta forma, embora a contribuição de Chekuri & Sidiropoulos seja relevante — afinal, a função $n^{\frac{1}{2}-\alpha}$ cresce mais lentamente do que a função $n^{\frac{1}{2}}$ —, uma vez que a variável n segue presente no fator de aproximação, a resposta continua assumindo valores cada vez mais distantes do ótimo conforme o tamanho do grafo (n) aumenta.

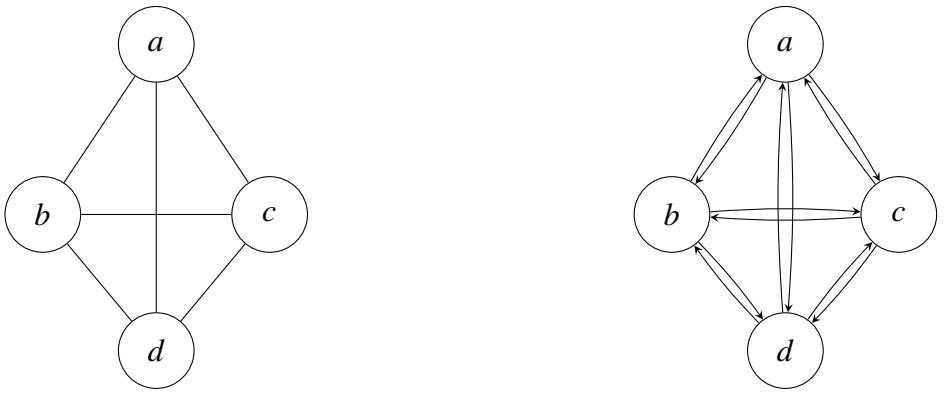
4 FORMULAÇÕES EM PLI E SAT

Diante da ausência de soluções práticas e corretas para o problema do *genus* mínimo, Beyer et al. (2016) propuseram formulações no formato de Programação Linear Inteira (PLI) e de Problema de Satisfatibilidade Booleana (SAT). Uma vez que a modelagem do problema em ambas as abordagens não é imediata, é necessário indicar as bases teóricas por trás das formulações.

4.1 CONCEITOS IMPORTANTES

4.1.1 Como são os grafos, e generalizações de arcos

O trabalho considera grafos finitos, não direcionados, simples, conexos e com grau mínimo 3. Apesar disso, a terminologia a respeito de arcos (arestas direcionadas) é útil para visualizar as estratégias presentes no trabalho. Assim, dado um grafo $G = (V, E)$, $A = \{uv, vu \mid \{u, v\} \in E\}$ denota o conjunto de arcos que são gerados por E ao substituir cada uma das arestas não direcionadas pelos dois arcos possíveis correspondentes (vu, uv). A Figura 4.1 demonstra essa equivalência.



(a) Grafo G com um conjunto E de arestas não-direcionadas.

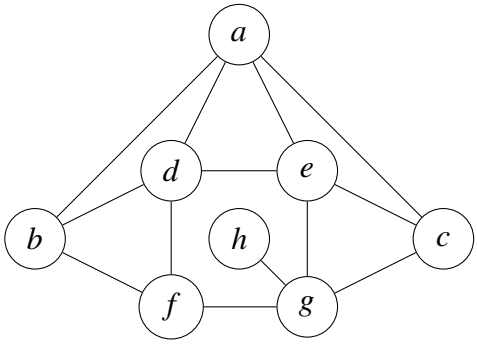
(b) Grafo G com um conjunto A de arestas direcionadas.

Figura 4.1: Relação entre os conjuntos de arestas E e A.

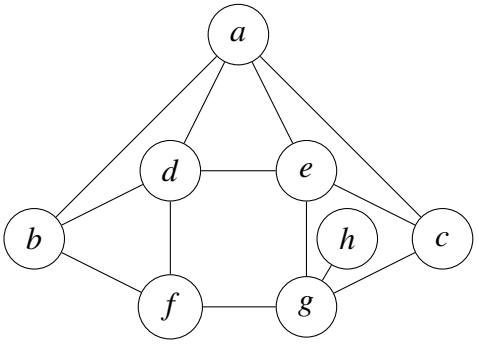
Considerar apenas vértices com grau mínimo 3 é uma medida capaz de potencialmente diminuir a complexidade da entrada, o que, dado o caráter exponencial das soluções que serão apresentadas, é desejável. Vértices de grau 0 ou 1 podem ser removidos sem prejuízo à imersão: no primeiro caso, por não participarem de nenhuma aresta; no segundo caso, por estarem conectados a somente um vértice, o que permite que sejam imediatamente posicionados na mesma face que seu vizinho. Na Figura 4.2, observa-se que o vértice h, o qual possui grau 1, pode ser posicionado em qualquer face sem causar prejuízo à imersão. Já os vértices de grau 2 podem ser substituídos por uma aresta entre seus dois vizinhos sem detrimento da estrutura ou da conectividade do grafo, como pode ser visualizado na Figura 4.3.

4.1.2 Rotações e sistemas de rotação

Para cada vértice v, a ordem cíclica (anti-horária) dos vizinhos de v é denominada rotação. Para o grafo G, um conjunto de rotações é chamado de sistema de rotações, no qual cada v possui uma rotação própria.

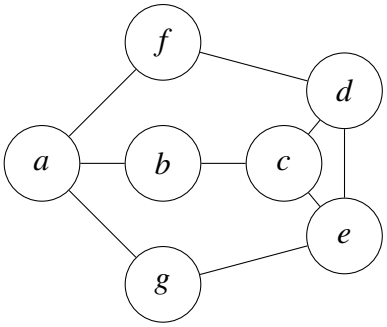


(a) Grafo com o vértice h na face $dfge$.

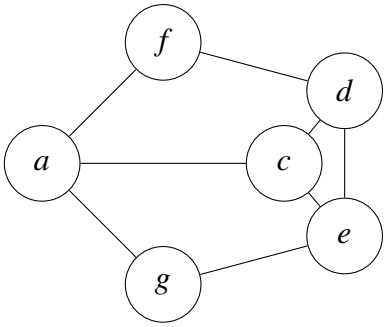


(b) Grafo com o vértice h na face egc.

Figura 4.2: O vértice h pode ser posicionado em qualquer lugar, sem prejuízo à imersão.



(a) Grafo original.



(b) Grafo com supressão do vértice b.

Figura 4.3: Substituição do vértice b por uma aresta entre a e c.

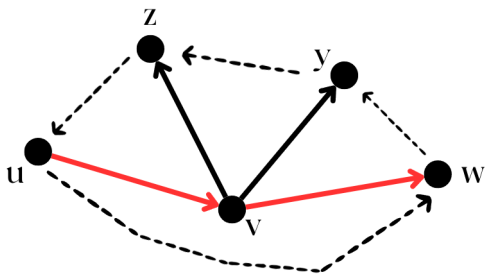


Figura 4.4: Visualização de uma rotação do vértice v.

Dado um sistema de rotação de G, é possível obter a imersão equivalente através do seguinte processo de *face tracing* descrito pelos autores: a partir de um arco uv, percorre-se esse arco de u até v, e continua-se com o arco vw, onde w é o vértice que sucede u na rotação de v. O fim do processo se dá quando o arco inicial é encontrado novamente, e o resultado é o cálculo de uma das faces da imersão. Nesse percurso, os arcos são escolhidos de forma que a face sendo calculada permaneça à direita do trajeto, ou seja, a fronteira da face é percorrida no sentido horário. Repetindo o procedimento para todos os arcos ainda não processados, consegue-se todas as faces da imersão. Na Figura **??**, é possível enxergar uma possível rotação em v; a ordem da rotação está indicada pelos arcos pontilhados. Chama-se a atenção para os arcos uv, vw e a ordem de u e w.

4.1.3 Característica de Euler

A característica de Euler provê um limite definido para o valor do *genus* de um grafo, assumindo que sua imersão possui um caráter celular. A característica enuncia que:

$$|V| - |E| + f = 2 - 2g \tag{4.1}$$

onde f é o número de faces da imersão e g é o *genus* da superfície. Portanto, ao determinar f, é possível determinar também o valor de g. Na prática, é necessário encontrar o sistema de rotação que fornece o número máximo de faces de uma dada imersão.

4.1.4 Limite superior do número de faces

Visto que o número de faces é um elemento central tanto na formulação PLI quanto na formulação SAT, é necessário definir um limite superior para esse valor. No caso da formulação SAT, a primeira condição estabelecida é considerar somente instâncias que obedecem a $f \equiv |E| - |V| \mod 2$. É simples ver a razão: rearranjando a fórmula, temos que:

$$f = |E| - |V| + 2 - 2g \tag{4.2}$$

O próximo passo é definir uma função de paridade $P(x)$ de maneira que $P(x) = 0$ se x é par e $P(x) = 1$ se x é ímpar. Aplicando a função $P(x)$ nos termos da equação 4.2:

$$P(f) = P(E - V) + P(2 - 2g) \tag{4.3}$$

É trivial ver que $P(2 - 2g) = 0$, independente do valor de g. Por conseguinte, $P(f)$ e $P(E - V)$ precisam ser iguais para que a igualdade seja obedecida. Seguindo essa propriedade, é possível cortar pela metade a lista de candidatos a valor de f a serem testados na formulação SAT. Dessa forma, são tentados valores crescentes de f, até que seja encontrada a primeira instância que não satisfaça a formulação. O maior valor de f que satisfaz as regras é o que pode fornecer o menor valor de g possível, de acordo com a característica de Euler. É possível observar em 4.2 que, conforme f aumenta, g é forçado a diminuir.

No caso da formulação PLI, $\bar{f}$ consiste no limite superior do número de faces da imersão. O limite é dado por

$$\bar{f} = \min\left\{\left\lfloor \frac{2|E|}{3} \right\rfloor,\ |E| - |V|\right\},$$

ou seja, é selecionado o valor mínimo entre dois termos: o primeiro oferece um limite considerando que cada aresta está presente na fronteira de, no máximo, duas faces, e que uma face é composta por, no mínimo, três vértices. O segundo termo consiste no valor que f assume de acordo com a característica de Euler quando $g = 1$.

4.2 FORMULAÇÃO EM PLI

A formulação em PLI busca encontrar uma imersão com o número máximo de faces, sendo, desta forma, um problema de maximização.

As seguintes variáveis foram criadas:

- x_i: variável binária que indica se a face i existe ou não.
- c_a^i: variável binária que indica se o arco a é atravessado ou não pela face i, ou seja, se o arco faz parte da fronteira da face i.

- $p^v_{u,w}$: variável binária que indica se w é o sucessor de u na rotação do vértice v.

As restrições foram escritas da seguinte maneira:

$$\max \quad \sum_{i=1}^{\bar{f}} x_i \tag{1a}$$

$$\text{s.t.} \quad x_i \leq \frac{1}{3} \sum_{a \in A} c^i_a \qquad \forall i \in [\bar{f}] \tag{1b}$$

$$\sum_{i=1}^{\bar{f}} c^i_a = 1 \qquad \forall a \in A \tag{1c}$$

$$\sum_{a \in \delta^-(v)} c^i_a = \sum_{a \in \delta^+(v)} c^i_a \, \forall i \in [\bar{f}],\ v \in V \tag{1d}$$

$$c^i_{vw} \geq c^i_{uv} + p^v_{u,w} - 1 \ \ \forall i \in [\bar{f}],\ v \in V,\ u \neq w \in N(v) \tag{1e}$$

$$c^i_{uv} \geq c^i_{vw} + p^v_{u,w} - 1 \ \ \forall i \in [\bar{f}],\ v \in V,\ u \neq w \in N(v) \tag{1f}$$

$$\sum_{w \in N(v),\, u \neq w} p^v_{u,w} = 1 \quad \forall v \in V,\ u \in N(v) \tag{1g}$$

$$\sum_{u \in N(v),\, w \neq u} p^v_{u,w} = 1 \quad \forall v \in V,\ w \in N(v) \tag{1h}$$

$$\sum_{u \in U} \sum_{w \in N(v) \setminus U} p^v_{u,w} \geq 1 \forall v \in V,\ \emptyset \neq U \subsetneq N(v) \tag{1i}$$

$$x_i \in \{0, 1\} \qquad \forall i \in [\bar{f}] \tag{1j}$$

$$c^i_a \in \{0, 1\} \qquad \forall i \in [\bar{f}],\ a \in A \tag{1k}$$

$$p^v_{u,w} \in \{0, 1\} \qquad \forall v \in V,\ u \neq w \in N(v) \tag{1l}$$

Restrição (1a):

Apresenta a função objetivo a ser maximizada nesta solução: o somatório de x_i.

Restrição (1b):

Garante que, se x_i for 1 (ou seja, a face i existe na imersão), a soma dos valores de c^i_a é igual ou superior a 3. Na prática, essa condição exige que uma face existente seja atravessada por 3 ou mais arcos (o valor mínimo de arcos necessário para definir uma face).

Restrição (1c):

Para todo $a \in A$, o somatório das variáveis binárias c^i_a (iterando por todas as i faces da imersões) deve ser exatamente igual a 1. Essa condição garante que cada arco seja atravessado por exatamente 1 face; caso o arco seja atravessado por mais faces, o resultado do somatório será maior que 1.

Restrição (1d):

$\delta^-(v)$: conjunto de arestas de entrada no vértice v.

$\delta^+(v)$: conjunto de arestas de saída no vértice v.

Para cada vértice v, para cada face i, o somatório das variáveis c^i_a, com $a \in \delta^-(v)$ e o somatório das variáveis c^i_a, com $a \in \delta^+(v)$, devem resultar no mesmo valor. Na prática, a restrição exige que, dada uma face i, é obrigatório que todos os vértices de sua fronteira possuam a mesma quantidade de arcos de entrada e de saída. Esse valor, se não for 0, deve ser 1, uma vez que, se um arco existe na fronteira da face i, não pode compor nenhuma outra fronteira. Por fim,

para que o fluxo da fronteira da face seja respeitado, é necessário que cada vértice possua um arco de entrada e um de saída (e não conte, por exemplo, com dois arcos de entrada e nenhum de saída). A Figura 4.5 ilustra um conjunto de faces que obedecem à restrição; cada face é composta por arcos na mesma cor. Os arcos em roxo pertencem à face exterior.

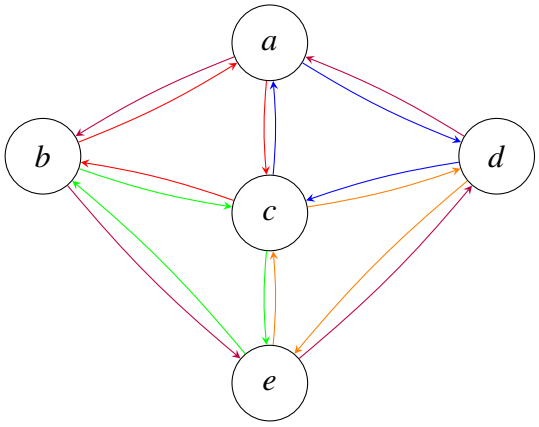


Figura 4.5: Uma composição correta de vértices e arcos em faces.

Restrições (1e) e **(1f)**:

Essas restrições trabalham em conjunto para garantir que os arcos uv e vw estejam numa mesma face i, caso w seja o sucessor de u na rotação em v. Ambas envolvem as variáveis c^i_{vw}, c^i_{uv} e $p^v_{u,w}$ e são escritas de forma similar, diferindo apenas nas posições das variáveis c^i_{vw} e c^i_{uv}. No caso de (1e), caso a variável c^i_{vw} seja 1, ou seja, a face i é atravessada pelo arco vw, os três cenários são admitidos:

- A face i é atravessada pelo arco uv e o vértice w é o sucessor de u na rotação em v;
- A face i é atravessada pelo arco uv, mas o vértice w não é o sucessor de u na rotação em v;
- A face i não é atravessada pelo arco uv e o vértice w não é o sucessor de u na rotação em v.

Entretanto, caso a variável c^i_{vw} seja 0, há um cenário particular que não é permitido pela restrição: a face i é atravessada pelo arco uv e o vértice w é o sucessor de u na rotação em v. Nesse cenário, os valores assumidos pelas variáveis c^i_{vw}, c^i_{uv} e $p^v_{u,w}$ seriam 0, 1, 1 , e as inequações assumiriam os seguintes valores:

(1e):

$$0 >= 1 + 1 - 1 \tag{4.4}$$

(1f):

$$1 >= 0 + 1 - 1 \tag{4.5}$$

É evidente que, assumindo tal conjunto de valores, a restrição (1f) não é cumprida. Dessa forma, a imersão final não pode admitir que, sendo w o vértice que sucede u na rotação em v, os arcos uv e vw estejam em faces diferentes (Figura 4.6).

Restrições (1g) e **(1h)**:

Essas restrições, em conjunto, garantem que a rotação em v (P^v) represente uma permutação válida. As duas regras exigem que, considerando a vizinhança de v, $N(v)$, para cada $w \in N(v)$, $u \neq w$, o somatório de $p^v_{u,w}$ deve ser igual a 1. Na prática, isso significa que cada

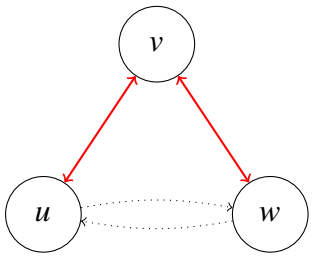


Figura 4.6: No caso desse subgrafo, é obrigatório que os arcos uv e vw estejam na mesma face.

vértice pertencente à vizinhança deve ter exatamente um sucessor. A restrição (1h) simplesmente troca $w \in N(v), u \neq w$ por $u \in N(v), w \neq u$. Assim, cada vértice presente na rotação do vértice v precisa ter, além de um único sucessor, um único antecessor. A Figura 4.7 demonstra uma rotação (indicada por arcos pontilhados) inválida dos vizinhos de v.

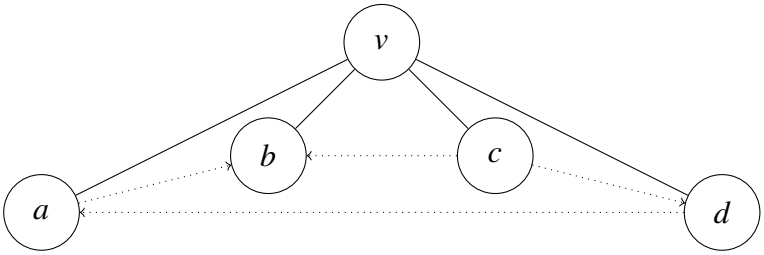


Figura 4.7: Uma rotação que viola a restrição (1h), pois o vértice y é sucessor de dois vértices distintos.

Entretanto, esse conjunto de restrições ainda não garante que os vértices vizinhos de v formem um único ciclo conexo. Com a adição das próximas restrições, essa condição será garantida.

Restrição (1i):

Para esta restrição, é necessário considerar todos os subconjuntos U que são subconjuntos próprios de $N(v)$ e que não são vazios. Ainda, considere-se $W = N(v) - U$, ou seja, os vértices vizinhos de v que não pertencem a U. Para cada $v \in V$ e para cada U existente, o somatório de $p^v_{u,w}$, onde $u \in U$ e $w \in W$, deve ser, no mínimo, 1.

Essa restrição garante que a rotação em v não tenha ciclos disjuntos, ao forçar que sempre exista um arco que saia do subconjunto U e entre no subconjunto W, ou seja, proíbe subconjuntos de vizinhos que se conectem apenas entre si. Impedir a existência de ciclos disjuntos nas rotações de um vértice v é essencial para garantir que a rotação possa corresponder ao contorno de uma única face da imersão. Se houver ciclos disjuntos, a formação de uma face geométrica coerente na imersão é inviabilizada — de forma didática, é como se a face ficasse "rasgada". A Figura 4.8 mostra uma rotação que obedece às restrições (1g) e (1h), mas não à restrição (1i); portanto, é inválida.

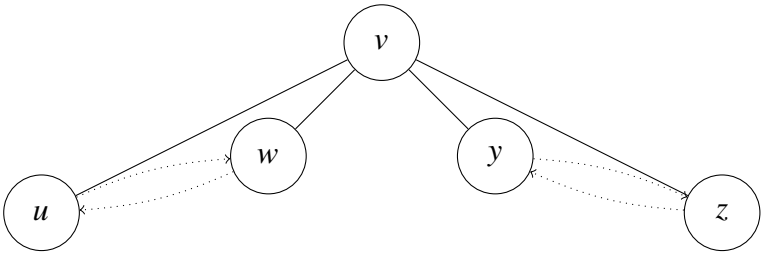


Figura 4.8: Um grafo que obedece às restrições (1g) e (1h), mas viola a restrição (1i).

Restrições (1j), (1k), (1l):

O conjunto de restrições assegura que os valores de x_i, c_a^i e $p_{u,w}^v$ se limitem a 0 ou 1, sem valores intermediários, de acordo com o princípio de uma formulação em programação linear inteira.

Em particular, (1j) estabelece que, para todo $i \in [\bar{f}]$, os valores de x_i sejam ou 0, ou 1.

(1k) estabelece que, para todo $i \in [\bar{f}]$ e para todo $a \in A$, os valores de c_a^i sejam ou 0, ou 1.

(1l) estabelece que, para todo $v \in V$, com $u \neq w \in N(v)$, os valores de $p_{u,w}^v$ sejam ou 0, ou 1.

A Figura 4.9 ilustra um grafo que atende a todas as restrições expostas nesta seção.

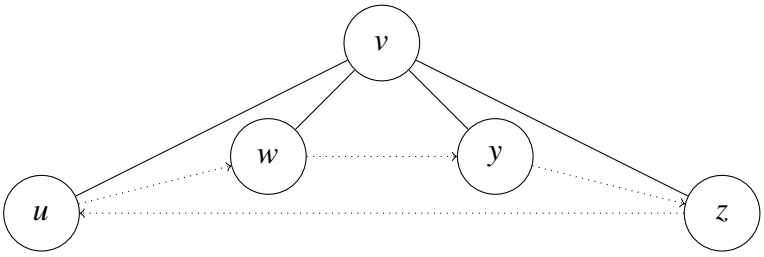


Figura 4.9: Este subgrafo satisfaz todas as restrições descritas.

4.3 FORMULAÇÃO EM SAT

A formulação em SAT busca responder se a imersão com no mínimo f faces é possível.

$$\neg(c_a^i \wedge c_a^j) \qquad \forall a \in A,\ i \neq j \in [f] \tag{2a}$$

$$\bigvee_{a \in A} c_a^i \qquad \forall i \in [f] \tag{2b}$$

$$p_{u,w}^v \rightarrow (c_{uv}^i \leftrightarrow c_{vw}^i) \qquad \forall v \in V,\ u \neq w \in N(v),\ i \in [f] \tag{2c}$$

$$\bigvee_{u \in N(v),\, u \neq w} p_{u,w}^v \qquad \forall v \in V,\ w \in N(v) \tag{2d}$$

$$\neg(p_{u,w}^v \wedge p_{u',w}^v) \qquad \forall v \in V,\ w \in N(v),\ u \neq u' \in N(v) \setminus \{w\} \tag{2e}$$

$$\bigvee_{w \in N(v),\, w \neq u} p_{u,w}^v \qquad \forall v \in V,\ u \in N(v) \tag{2f}$$

$$\neg(p_{u,w}^v \wedge p_{u,w'}^v) \qquad \forall v \in V,\ u \in N(v),\ w \neq w' \in N(v) \setminus \{u\} \tag{2g}$$

$$\bigvee_{u \in U,\, w \in N(v) \setminus U} p_{u,w}^v \qquad \forall v \in V,\ \emptyset \neq U \subseteq N(v) \tag{2h}$$

Regra (2a):

(2a) apresenta uma negação de uma conjunção de dois termos, c_a^i e c_a^j. Caso ambos sejam verdadeiros simultaneamente, significa que o mesmo arco a atravessa tanto a face i quanto a face j, o que invalida a imersão, pois cada arco a deve estar presente em apenas uma face.

Regra (2b):

(2b) assegura que, para cada face i da imersão, existe pelo menos um arco $a \in A$ tal que c_a^i seja verdadeiro. Uma vez que as variáveis c_a^i estão conectadas por uma disjunção, essa regra garante que toda face tenha, no mínimo, um arco a atribuído a si.

Regra (2c):

Nesta regra, é apresentada uma implicação lógica no formato A→B. O termo A é composto pela variável $p^v_{u,w}$ para todo $v \in V$, $u \neq w \in N(v)$. O termo B consiste numa bicondicional composta pelas variáveis c^i_{vw} e c^i_{uv}, ou seja, c^i_{vw} será verdadeira se e somente se c^i_{uv} também for. Portanto, obriga-se que os arcos uv e vw estejam ou simultaneamente presentes na face i, ou então, simultaneamente ausentes.

A condição (2c), dessa forma, será falsa apenas se o termo A for verdadeiro e o termo B falso, o que se trata de um cenário em que w é o sucessor de u na rotação em v, mas os arcos uv e uw não estão na mesma face i.

Regra (2d):

(2d) garante que, para todo vértice $v \in V$, $w \in N(v)$, há pelo menos um vértice u (dentre os vizinhos de v), $u \neq w$, que seja sucedido por w, ou seja, $p^v_{u,w}$ é verdadeiro em pelo menos um dos casos. Essa condição é cumprida desde que um dos termos na disjunção seja verdadeiro.

Regra (2e):

(2e) garante que, para todo vértice $v \in V$, todo vizinho $w \in N(v)$, e quaisquer $u \neq u' \in N(v) \setminus \{w\}$, não é permitido que dois vértices distintos u e u' tenham w como sucessor de forma simultânea na rotação de v. Unida à regra (2d), essa regra desempenha o mesmo papel da restrição (1g) na formulação PLI apresentada anteriormente.

Regra (2f):

(2f) garante que, para todo vértice $v \in V$, $u \in N(v)$, há pelo menos um vértice w (dentre os vizinhos de v), $w \neq u$, que seja precedido por u, ou seja, $p^v_{u,w}$ é verdadeiro em pelo menos um dos casos. Essa condição é cumprida desde que um dos termos na disjunção seja verdadeiro.

Regra (2g):

(2g) garante que, para todo vértice $v \in V$, todo vizinho $u \in N(v)$, e quaisquer $w \neq w' \in N(v) \setminus \{u\}$, não é permitido que dois vértices distintos w e w' tenham u como antecessor de forma simultânea na rotação de v. Unida à regra (2f), essa regra desempenha o mesmo papel da restrição (1h) na formulação PLI apresentada anteriormente.

Regra (2h):

A presente regra se encarrega de impedir que existam ciclos disjuntos nas rotações de cada vértice v. Assim, é necessário enumerar todos os subconjuntos U que são subconjuntos próprios não vazios de $N(v)$. Para todo vértice $v \in V$ e $U \subsetneq N(v)$, deve haver, no mínimo, um termo $p^v_{u,w}$ que seja verdadeiro, uma vez que os termos estão conectados por uma disjunção para todo $u \in U$ e $w \in N(v) - U$. Assim, essa regra força que uma rotação só seja válida caso todos os subconjuntos U tenham, no mínimo, um arco que sai de U e entre em $N(v) - U$.

4.4 COMPARAÇÃO ENTRE AS DUAS FORMULAÇÕES

Após explorar as formulações e analisar as restrições e regras, é possível compará-las e apontar similaridades e diferenças. A princípio, observa-se que as duas soluções buscam assegurar as seguintes condições: exclusividade de arcos por face, exclusividade de antecessor e sucessor na rotação, ausência de ciclos disjuntos, obrigatoriedade de arcos uv e vw estarem na mesma face e garantia de que toda face possui, pelo menos, n arcos (o valor de n difere entre as formulações). A Tabela 4.1 lista essas condições e indica quais restrições/regras são responsáveis por estabelecê-las.

Uma diferença interessante entre as formulações se dá na maneira de garantir (ou não) que todos os arcos estejam associados a uma face: na formulação PLI, há uma restrição que assegura essa característica (1c), porém, na formulação SAT, não há nenhuma regra com papel análogo. Uma regra com esse propósito não é necessária, pois não é uma condição almejada pela

	PLI	SAT
Um arco só pertence a uma face	(1c)	(2a)
Cada vértice na rotação só possui um sucessor, um antecessor	(1g), (1h)	(2d), (2e), (2f), (2g)
Ausência de ciclos disjuntos	(1i)	(2h)
Todos os arcos precisam ser usados	(1c)	não há
Garantia de que toda face possui, pelo menos, n arcos	(1b), n = 3	(2b), n = 1
Arcos uv e vw na mesma face	(1e) e (1f)	(2c)

Tabela 4.1: Comparação entre quais restrições/regras garantem as condições desejadas para a imersão de G.

formulação SAT. A razão pode ser encontrada ao reexaminar o objetivo posto: decidir se o grafo de entrada admite uma imersão com f faces.

Ainda, uma vez que o método para encontrar a resposta desejada consiste em, de forma iterada, aumentar o valor de f até encontrar a primeira instância incorreta, é possível visualizar como as imersões "intermediárias", ou seja, encontradas antes da resposta definitiva, podem não conter todos os arcos disponíveis para a construção da imersão, pois ainda há faces "faltando". Assim, a formulação precisa permitir que nem todos os arcos estejam na composição de uma face. Na formulação PLI, por outro lado, tendo em vista que a função objetivo busca maximizar o número de faces da imersão, não há imersões "intermediárias", portanto, a imersão calculada deve empregar todos os arcos existentes no grafo G.

4.5 DESEMPENHO E OTIMIZAÇÕES

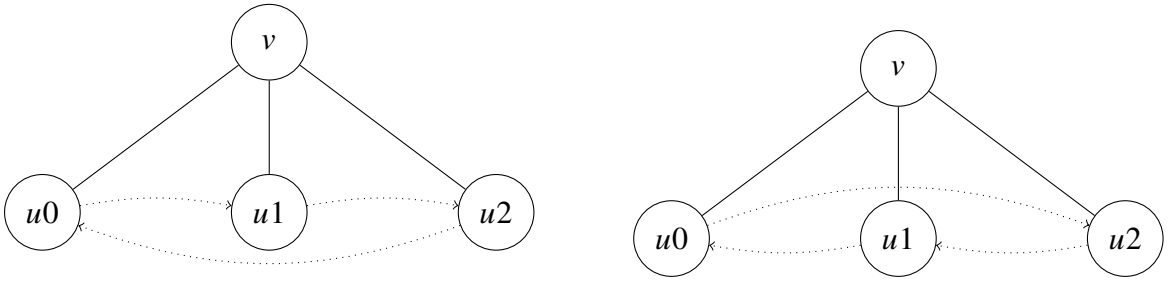
No artigo original, os autores apresentam resultados de experimentos práticos com as formulações que foram discutidas. As soluções demonstraram bom desempenho para grafos com $|V| \leq 30$, mas a eficiência decai progressivamente até $|V| = 70$. Para grafos com mais de 70 vértices, o tempo de processamento torna-se excessivamente elevado. Detalhes adicionais sobre os experimentos podem ser encontrados em Beyer et al. (2016).

Em adição às formulações, os autores também proveram otimizações interessantes que podem melhorar o desempenho das implementações de cada solução; uma delas será trazida como exemplo.

4.5.1 Vértices com grau 3

Essa otimização pode ser aplicada a todos os vértices do grafo G que possuem grau 3, definidos pelo conjunto $V_3 = \{v \in V \mid deg(v) = 3\}$. Considere um vértice $v \in V_3$ e sua vizinhança, $N(v) = \{u_0, u_1, u_2\}$. As rotações possíveis em v são apenas duas: $u0 \rightarrow u1 \rightarrow u2$ ou $u2 \rightarrow u1 \rightarrow u0$, ilustradas na Figura 4.10.

Portanto, é possível representar as duas possíveis permutações com as mesmas variáveis binárias usadas até então: 0 ou 1 (formulação PLI), verdadeiro ou falso (formulação SAT). Ao determinar a rotação em v dessa forma, é possível substituir as combinações representadas pelas variáveis $p^v_{u,w}$ por definições estáticas indicadas por uma variável única p^v. Ainda, essa determinação é capaz de garantir dois dos requisitos cobertos pelas restrições (1e) - (1i): arcos de vértices adjacentes na rotação em v devem estar na mesma face, vértices na rotação devem possuir apenas um antecessor e sucessor, e ausência de círculos disjuntos. Assim, as restrições (1e) - (1i) podem ser substituídas pelas seguintes restrições alternativas:



(a) Rotação $u0 \to u1 \to u2$.

(b) Rotação $u0 \to u2 \to u1$.

Figura 4.10: Rotações possíveis para V_3.

$$c^i_{vu_{k+1}} \geq c^i_{u_kv} + p^v - 1 \quad \forall i \in [f],\ v \in V_3,\ k \in [3] \tag{3a}$$
$$c^i_{u_kv} \geq c^i_{vu_{k+1}} + p^v - 1 \quad \forall i \in [f],\ v \in V_3,\ k \in [3] \tag{3b}$$
$$c^i_{vu_k} \geq c^i_{u_{k+1}v} - p^v \quad \forall i \in [f],\ v \in V_3,\ k \in [3] \tag{3c}$$
$$c^i_{u_{k+1}v} \geq c^i_{vu_k} - p^v \quad \forall i \in [f],\ v \in V_3,\ k \in [3] \tag{3d}$$

O conjunto de restrições acima deve ser aplicado para cada face, para cada $v \in V_3$, para cada $k \in [3]$, sendo k o identificador de cada vértice $u \in N(v)$.

Três vértices devem ser considerados para a compreensão dessas restrições: v, u_k e u_{k+1}. Caso $p^v = 0$, (3c) e (3d) proíbem que $c^i_{vu_k}$ e $c^i_{u_{k+1}v}$ tenham valores distintos; ou ambos devem ser 0, ou ambos devem ser 1. Esse par de restrições obriga que os arcos $u_{k+1}v$ e vu_k estejam na mesma face i, uma vez que, caso o arco $u_{k+1}v$ exista na face i, o arco vu_k também deve existir nessa mesma face. A integridade da face é assegurada, dessa forma, pela existência desses dois arcos e pela ordem da rotação em v quando $p^v = 0$.

Caso $p^v = 1$, (3a) e (3b) proíbem que $c^i_{u_kv}$ e $c^i_{vu_{k+1}}$ tenham valores distintos, seguindo o mesmo propósito explicado no parágrafo anterior. A principal diferença entre esse par de restrições e (3c) e (3d) é o sentido dos arcos e da rotação. Portanto, obriga-se que exista um arco que sai u_k e entra v, e que sai de v e entra em u_{k+1}. Novamente, a integridade da face é assegurada pela existência desses dois arcos e pela ordem da rotação em v quando $p^v = 1$. É possível visualizar essa configuração na Figura 4.11.

Outras combinações de valores não invalidam as inequações.

(a) Rotação $u0 \to u1 \to u2$.

(b) Rotação $u0 \to u2 \to u1$.

Figura 4.11: Arcos que são garantidos pelas restrições de acordo com o valor de p^v.

Aplicada à formulação SAT, essa modificação pode ser expressa com as seguintes regras:

$$p^v \rightarrow \left(c^i_{u_k v} \leftrightarrow c^i_{v u_{k+1}}\right) \qquad \forall v \in V_3,\ k \in [3],\ i \in [f] \tag{4a}$$

$$\neg p^v \rightarrow \left(c^i_{u_{k+1} v} \leftrightarrow c^i_{v u_k}\right) \qquad \forall v \in V_3,\ k \in [3],\ i \in [f]. \tag{4b}$$

Neste formato, a rotação $u0 \rightarrow u1 \rightarrow u2$ é representada pelo termo $\neg p^v = true$, enquanto a rotação $u2 \rightarrow u1 \rightarrow u0$ é representada pelo termo $p^v = true$.

O formato de cláusulas deixa o objetivo da modificação mais explícito: obrigar que determinados arcos estejam presentes na mesma face i. (4a) e (4b) consistem em condicionais $A \rightarrow B$, onde A é a rotação escolhida e B é uma bicondicional composta por variáveis c^i.

A condicional (4a) será inválida apenas no seguinte caso: p^v é verdadeiro, mas ($c^i_{u_k v} \leftrightarrow c^i_{v u_{k+1}}$) é falso. Visto que a proposição é uma bicondicional, só será falsa caso $c^i_{u_k v}$ e $c^i_{v u_{k+1}}$ possuam valores diferentes, ou seja, os arcos $u_k v$ e $v u_{k+1}$ devem existir na mesma face.

(4b) segue a mesma ideia, substituindo o antecedente por $\neg p^v$ e o consequente por $(c^i_{u_{k+1} v} \leftrightarrow c^i_{v u_k})$.

A efetividade dessa otimização depende, claramente, da quantidade de vértices com grau 3 presentes no grafo G.

4.6 TAMANHO EXPONENCIAL DAS FORMULAÇÕES

Em ambas as formulações, a complexidade exponencial do problema aparece através do número de restrições/regras geradas, especificamente, através da restrição (1i) e da regra (2h), visto que são as responsáveis pela enumeração de todos os subconjuntos possíveis a partir do conjunto $N(v)$. Beyer et al. (2016) discutem alguns métodos que são capazes de aproximar a quantidade de restrições/regras para um tamanho polinomial. Os autores ressaltam que esses métodos obtêm mais sucesso no caso da formulação SAT, criando, desta forma, uma vantagem em relação à outra formulação.

Reformulação com ordenação: esse método visa substituir as variáveis relativas a permutações (p^v) por variáveis que indiquem posições fixas dos vértices numa determinada rotação. Para esse objetivo, seria introduzida uma nova variável $q^v_{j,u}$, a qual possui valor verdadeiro caso o vértice u seja o j-ésimo vértice na rotação em v, para cada $v \in V$, $u \in N(v)$. Com essa introdução, é possível descartar a variável p^v da solução e substituir as restrições (2d)-(2h) por restrições que garantam que q^v represente uma bijeção, assim como alterar a regra (2c) para:

$$\bigvee_{j \in [\deg(v)]} \left(q^v_{j,u} \wedge q^v_{j+1,w}\right) \rightarrow \left(c^i_{uv} \leftrightarrow c^i_{vw}\right) \quad \forall\, v \in V,\ u \neq w \in N(v),\ i \in [f].$$

Reformulação com intermediação: esse método prevê a inclusão de variáveis $r^v_{x,y,x}$, para cada $x, y, z \in N(v)$. Caso $r^v_{x,y,x} = 1$ ou $r^v_{x,y,x} = true$, significa que y está posicionado em algum lugar entre x e z na rotação em v. Uma vez que a rotação consiste numa ordenação cíclica, é possível estabelecer as seguintes relações de simetria: $r^v_{x,y,x} \equiv r^v_{y,z,x} \equiv r^v_{z,x,y} \equiv \neg r^v_{x,z,y} \equiv \neg r^v_{z,y,x} \equiv \neg r^v_{y,z,x}$.

Dessa forma, para garantir que p^v seja uma permutação válida, é necessário conectar a variável $p^v_{u,w}$ às novas variáveis r, o que pode ser feito por meio da regra $p^v_{u,w} \leftrightarrow \bigwedge_{y \in N(v) \setminus \{u,w\}} r^v_{u,w,y}$. Essa regra constata que w é sucessor de u na rotação em v ($p^v_{u,w} = true$) se e somente se, para todo $y \in N(v) \setminus \{u, w\}$, $r^v_{u,w,y} = true$, ou seja, se for possível afirmar que w sempre está entre u e y (para cada y).

Ainda, para assegurar que a ordem indicada pelas variáveis r^v seja coerente, faz-se necessária a adição da regra $r^v_{u,w,x} \wedge r^v_{u,x,y} \rightarrow r^v_{u,w,y} \wedge r^v_{w,x,y}$ for all $\{u, w, x, y\} \subseteq N(v)$. Dado um conjunto $\{u, w, x, y\}$, caso $(r^v_{u,w,x} \wedge r^v_{u,x,y}) = true$, significa que a rotação é $u \rightarrow w \rightarrow x \rightarrow y$. Assim, a segunda parte da condicional exige que w esteja entre u e y (pois já é sabido que w está entre u e x, e x está entre u e y), assim como também exige que x esteja entre w e y, respeitando, de forma equivalente, as relações de "intermediação" entre os vértices na rotação. É importante ressaltar que, por mais que essa regra também esteja embasada em geração de subconjuntos, não cai no problema da exponencialidade do número de regras, uma vez que todos os subconjuntos possuem tamanho fixo (4).

5 CONCLUSÃO

O trabalho teve como objetivo explorar o histórico do Problema do *Genus* Mínimo na Teoria dos Grafos e dissecar a estrutura de soluções em formato PLI e SAT. Para tal, foram definidos fundamentos importantes para contextualização do problema, e depois, a descrição formal do problema. A complexidade do problema foi evidenciada através da menção às provas de $N\mathcal{P}$-completude escritas por Thomassen e exposição dos algoritmos propostos como solução. Ainda, foi explicado que esses algoritmos não consistem em boas soluções práticas, seja por estarem incorretos, seja por possuírem implementação muito complexa.

A exploração das formulações PLI e SAT foi iniciada a partir dos fundamentos que as sustentam. Dentre esses conceitos, destacam-se o de rotação de um vértice v — a ordenação de vértices vizinhos que define a imersão — e o limite superior de faces $[\bar{f}]$ — essencial para a função objetivo a ser maximizada. Foi explicado como o *genus* de um grafo G pode ser determinado a partir da Característica de Euler, demonstrando o uso de uma abordagem geométrica para a solução de um problema topológico. Essa possibilidade é um exemplo claro de como o problema está posicionado entre essas duas áreas, o que o torna particular e interessante dentro da Teoria dos Grafos.

Ao examinar as formulações PLI e SAT, foi possível delinear as condições que são buscadas por ambas as formulações: exclusividade de arcos por face, exclusividade de antecessor e sucessor na rotação, ausência de ciclos disjuntos, obrigatoriedade de arcos uv e vw estarem na mesma face e garantia de que toda face possui, pelo menos, n arcos. Essas condições são cruciais para a formação de conjuntos de faces que sejam consistentes, de forma que os arcos sejam organizados em orientações coerentes. No entanto, foi observado que as formulações também contam com uma diferença relevante: uma procura assegurar que todos os arcos sejam utilizados na imersão (PLI); a outra, não. As restrições e regras de cada formulação foram analisadas e esclarecidas, destacando o papel de cada variável e item, como estão estruturadas as equações/inequações e cláusulas e como elas contribuem para garantir as condições mencionadas anteriormente.

Após a exposição das formulações, foi trazido um exemplo de possível otimização, o qual tira proveito das poucas rotações que podem ser formadas por uma vizinhança de três vértices — a qual é a vizinhança "mínima" que um grafo G a ser processado pode ter. O sucesso da otimização é, claramente, dependente do grafo a ser processado.

Por fim, foram discutidos métodos de mitigar o problema do número exponencial de restrições/regras, os quais envolvem criações de novas variáveis e substituição de partes das formulações. Esses métodos são mais apropriados e efetivos para a formulação SAT, o que cria uma possível vantagem em relação à outra solução.

Dessa forma, foi possível explicitar a alta complexidade do Problema do *Genus* Mínimo e entender os desafios pertinentes à sua resolução. As modelagens analisadas se mostram eficazes ao traduzir um problema topológico em termos algébricos e computacionais, apoiando-se em fundamentos geométricos, por exemplo, a característica de Euler. Por mais que sua aplicação apresente limitações, considerando a ausência notável de soluções práticas para o problema, as formulações consistem em uma contribuição importante para lidar com determinadas instâncias de um problema difícil.

REFERÊNCIAS

Beyer, S., Chimani1, M., Hedtke1, I. e Kotrbcik, M. (2016). A practical method for the minimum genus of a graph: Models and experiments. *Proceedings of the 15th International Symposium on Experimental Algorithms*, 9685:75–88.

Chekuri, C. e Sidiropoulos, A. (2016). Approximation algorithms for euler genus and related problems. *Proceedings of the 15th International Symposium on Experimental Algorithms*, 9685:75–88.

Djidjev, H. e Reif, J. (1991). An efficient algorithm for the genus problem with explicit construction of forbidden subgraphs. Em *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '91, páginas 337–347.

Filotti, I. S., Miller, G. L. e Reif, J. (1979). On determining the genus of a graph in o(v o(g)) steps(preliminary report). Em *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, página 27–37. Association for Computing Machinery.

Gagarin, A., Myrvold, W. e Chambers, J. (2009). The obstructions for toroidal graphs with no k3,3's. *Discrete Mathematics*, 309(11):3625–3631. 7th International Colloquium on Graph Theory.

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

Gross, J. L. e Tucker, T. W. (1987). *Topological Graph Theory*, páginas 26–29. Wiley.

Heawood, P. J. (1890). Map colour theorem. *Quart. J. Math*, 24:332–338.

Lovász, L. (2005). Graph minor theory. *Bulletin (New Series) of the American Mathematical Society*, 43:75–86.

Miller, J., Bhatia, D. e Kobourov, S. (2024). State of the art of graph visualization in non-euclidean spaces. *Computer Graphics Forum*, 43.

Mohar, B., Juvan, M. e Marinček, J. (1995). Embedding a graph in the torus in linear time. *Computer Graphics Forum*, 920:360–363.

Myrvold, W. e Kocay, W. (2011). Errors in graph embedding algorithms. *Journal of Computer and System Sciences*, 77:430–438.

Popescu-Pampu, P. (2016). *What is the Genus?* Springer.

Ringel, G. e Youngs, J. W. (1968). Solution of the heawood map-coloring problem. *Proceedings of the National Academy of Sciences USA*, 60:438–445.

Thomassen, C. (1989). The graph genus problem is np-complete. *Journal of Algorithms*, 10:568–576.

Thomassen, C. (1997). The genus problem for cubic graphs. *Journal of Combinatorial Theory*, Series B, 69:52–58.

Weeks, J. R. (2002). *The Shape of Space*. Marcel Dekker.