Um algoritmo baseado em LBFS para reconhecimento de grafos ordenáveis por vizinhança gêmea

Kaio Henrique Masse Vieira
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
kaiovieira@dcc.ufmg.br

Orientador: Vinicius Fernandes dos Santos
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
viniciussantos@dcc.ufmg.br

Resumo

Grafos ordenáveis por vizinhança gêmea são definidos como aqueles que admitem uma ordem de eliminação em que cada vértice a ser removido possui a propriedade de que todos os seus vizinhos são gêmeos verdadeiros. Essa classe de grafos foi introduzida recentemente, inspirada por semelhanças entre as ordens de eliminação de grafos cordais e dualmente cordais. Neste trabalho, abordamos o problema de reconhecimento para essa classe de grafos. Propomos um novo algoritmo que utiliza duas passadas de uma variante da Busca em Largura Lexicográfica (LBFS) para reconhecer grafos ordenáveis por vizinhança gêmea em tempo linear.

I. Introdução

Ordens de eliminação são ordenações dos vértices de um grafo que exibem certas propriedades estruturais. Essas ordenações são uma ferramenta poderosa no estudo de classes de grafos, pois permitem a construção de algoritmos eficientes para problemas que são intratáveis em grafos gerais. Ordens de eliminação também são frequentemente utilizadas como base para algoritmos de reconhecimento de classes de grafos, onde o objetivo é determinar, de maneira eficiente, se um grafo dado como entrada pertence a uma classe de interesse.

Diversas classes de grafos podem ser caracterizadas por meio de ordens de eliminação [1]. A primeira a ser estudada, e a mais conhecida, é a ordem de eliminação perfeita, que caracteriza os grafos cordais. Ordens de eliminação perfeita podem ser usadas para resolver problemas como coloração e cobertura por cliques em tempo polinomial [2]. Além disso, o reconhecimento de grafos cordais pode ser feito em tempo linear utilizando um algoritmo de busca em grafos chamado de Busca em Largura Lexicográfica (LBFS), introduzido por Rose, Tarjan e Lueker para esse fim [3].

Desde então, algoritmos de busca em grafos foram muito relevantes para o projeto de algoritmos de reconhecimento de classes de grafos. É especialmente interessante o caso de algoritmos de reconhecimento que realizam mais de uma passada de um algoritmo de busca, usando a informação obtida em uma passada para guiar as passadas subsequentes, como é o caso de algoritmos de reconhecimento de cografos [4] e de grafos de intervalo [5].

Grafos ordenáveis por vizinhança gêmea são uma classe de grafos recentemente introduzida [8], cuja definição é inspirada pelas semelhanças entre as ordens de eliminação de grafos cordais e dualmente cordais. Apesar de ter sido pouco estudada, essa classe já possui caracterizações estruturais e por subgrafos proibidos, além de um algoritmo de reconhecimento em tempo linear. Entretanto, apesar de sua simplicidade, o algoritmo de reconhecimento proposto em [8] não utiliza algoritmos de busca em grafos tradicionais, o que poderia ser interessante para entender melhor a relação entre essa classe de grafos e outras classes conhecidas.

Neste trabalho, desenvolvemos um algoritmo de reconhecimento de grafos ordenáveis por vizinhança gêmea que utiliza duas passadas de uma variante da Busca em Largura Lexicográfica (LBFS). Para apresentar nosso resultado, este artigo está organizado da seguinte forma. Na Seção 2, revisamos as definições fundamentais de teoria dos grafos que serão utilizadas. Na Seção 3, definimos formalmente os grafos ordenáveis por vizinhança gêmea e apresentamos suas caracterizações. A Seção 4 detalha o funcionamento do algoritmo de reconhecimento proposto, que utiliza duas passadas de uma variante da LBFS. Por fim, na Seção 5, discutimos as conclusões e possíveis direções para trabalhos futuros.

II. Referencial Teórico

Um *grafo* G é um par ordenado (V, E), com $n = |V|$ *vértices* e $m = |E|$ *arestas*, onde as arestas são conjuntos de pares distintos de vértices, isto é, $E \subseteq \{\{u, v\} \mid u, v \in V \wedge u \neq v\}$. A aresta $\{u, v\}$ também pode ser denotada por uv. A *vizinhança aberta* $N_G(v)$ de um vértice v é o conjunto de vértices adjacentes a v, isto é, $N_G(v) = \{u \in V \mid uv \in E\}$. A *vizinhança fechada* de v é $N_G[v] = N_G(v) \cup \{v\}$. A *k-ésima vizinhança* de v, denotada por $N_G^k[v]$ é o conjunto $\{u \mid d(u, v) \leq k\}$, onde $d(u, v)$ é o número de arestas no caminho mínimo entre u e v. Nos casos em que é evidente qual o grafo em questão, os

subscritos serão omitidos. O *grau* de um vértice v é o número de vértices adjacentes a v, isto é, $d(v) = |N(v)|$. Dizemos que v é *isolado* se $d(v) = 0$, *pendente* se $d(v) = 1$ e universal se $d(v) = |V(G)| - 1$.

Um caminho entre os vértices p_1 e p_k é uma sequência de vértices $p_1p_2 \dots p_k$ tal que $p_ip_{i+1} \in E$ para todo $i \in \{1, 2, \dots, k-1\}$ e $p_i \neq p_j$ para todo $i \neq j$. Um *ciclo* é uma sequência de vértices $p_1p_2 \dots p_kp_1$ tal que $p_1p_2 \dots p_k$ é um caminho e $p_1p_k \in E(G)$. Uma *clique* de k vértices é um subgrafo completo, isto é, com $|E(G)| = \binom{|V(G)|}{2}$ arestas. Denotamos por P_k o caminho com k vértices, por C_k o ciclo com k vértices, e por K_k o grafo completo com k vértices. Um grafo G é *conexo* se, para todo par de vértices $u, v \in V(G)$, existe um caminho entre u e v. Uma componente conexa de um grafo é um subgrafo induzido conexo maximal. Uma *árvore* é um grafo conexo e acíclico.

Um *subgrafo* de um grafo G é um grafo H tal que $V(H) \subseteq V(G)$ e $E(H) \subseteq E(G)$. H é um *subgrafo induzido* de G se é subgrafo de G e $E(H) = \{\{u, v\} \in E(G) \mid u, v \in V(H)\}$. Se H é um subgrafo induzido de G com vértices $V(H)$, é comum escrever $G[V(H)]$ para se referir ao grafo H.

Uma relação binária α sobre um conjunto X é uma *ordem parcial* se for reflexiva ($\forall v \in X, (v, v) \in \alpha$), antissimétrica ($\forall u, v \in X, (u, v) \in \alpha \wedge (v, u) \in \alpha \implies u = v$) e transitiva ($\forall u, v, w \in X, (u, v) \in \alpha \wedge (v, w) \in \alpha \implies (u, w) \in \alpha$). Se α é uma ordem parcial, define-se $u \leq_\alpha v$ quando $(u, v) \in \alpha$, e também $u <_\alpha v$ quando $u \leq_\alpha v$ e $u \neq v$. Uma ordem parcial α é uma *ordem total* se, para todo par de vértices distintos $u, v \in X$, $u <_\alpha v$ ou $v <_\alpha u$. Se α é uma ordem total, os elementos de X podem ser ordenados unicamente em uma sequência $x_1, x_2, \dots, x_{|X|}$, onde $\forall i < j, x_i <_\alpha x_j$. Nesse contexto, definimos $\alpha^{-1}(x_i) = i$. O reverso de uma ordem total α é a ordem total γ tal que $\forall u, v \in X, u <_\gamma v \Leftrightarrow v <_\alpha u$.

Uma *partição* $\mathcal{P}$ de um conjunto X é uma família $\{P_1, P_2, \dots, P_k\}$ de subconjuntos disjuntos de X tal que $\bigcup_{P_i \in \mathcal{P}} P_i = X$. Uma partição $\mathcal{P}$ de X é *refinada* por uma partição $\mathcal{Q}$ de X se, para todo $Q \in \mathcal{Q}$, existe $P \in \mathcal{P}$ tal que $Q \subseteq P$.

Um grafo G é isomorfo a H se existe uma bijeção $f : V(G) \to V(H)$ tal que $f(u)f(v) \in E(H) \Leftrightarrow uv \in E(G)$. Dizemos que um grafo G é livre de H se G não contém um subgrafo induzido isomorfo a H. Um grafo G é *cordal* se G é livre de C_k para todo $k \geq 4$.

O complemento de um grafo G é o grafo $\overline{G}$ tal que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{\{u, v\} \mid u, v \in V(G) \wedge u \neq v \wedge uv \notin E(G)\}$. Um grafo G é *split* se ambos G e $\overline{G}$ são cordais. Se G é split, então $V(G)$ pode ser particionado em (K, I), onde K é uma clique e I é um conjunto independente, isto é, não existe aresta entre vértices de I. Tal partição é chamada de *split partition* de G.

Um grafo G com n vértices possui uma ordem de eliminação de seus vértices $v_1, v_2, \dots, v_n$ satisfazendo a propriedade P se, para todo $i \in \{1, \dots, n\}$, o vértice v_i satisfaz a propriedade P em $G_i = G[\{v_i, \dots, v_n\}]$. Nesse contexto é comum se referir a $N_{G_i}(u)$ como a *vizinhança à direita* de u (com respeito à ordem $v_1, v_2, \dots, v_n$). Alternativamente, define-se que um vértice v pode dar início a uma P-ordem de eliminação em G se ele satisfaz a propriedade P.

Um vértice v é simplicial se existe aresta entre todos os seus vizinhos, isto é, $N(v)$ é uma clique. Um vértice v pode começar uma ordem de eliminação perfeita (OEP) se v é simplicial. G é cordal se, e somente se, G possui uma ordem de eliminação perfeita.

III. Grafos ordenáveis por vizinhança gêmea

Os grafos ordenáveis por vizinhança gêmea são definidos como aqueles que admitem uma ordem de eliminação satisfazendo a seguinte propriedade: um vértice v satisfaz a propriedade Y em G se $\forall u \in N_G(v) : N^2_G[v] \subseteq N_G[u]$. Por esse motivo, a classe de grafos ordenáveis por vizinhança gêmea também é chamada de classe $\mathcal{Y}$. Uma forma equivalente, talvez mais intuitiva, de entender a propriedade Y é apresentada na proposição a seguir, em termos de *gêmeos verdadeiros*, ou seja, vértices com a mesma vizinhança fechada.

Proposição 1. Seja G um grafo. Um vértice $v \in V(G)$ satisfaz a propriedade Y se, e somente se, os vértices em $N(v)$ são gêmeos verdadeiros.

Frequentemente, descrever de forma sucinta as estruturas que não podem estar presentes em uma família de objetos que estamos estudando é uma maneira de obter resultados interessantes. Nesse contexto, encontrar uma família de subgrafos proibidos para uma classe de grafos é um problema importante. Similarmente, encontrar uma configuração proibida dentre as ordens satisfazendo alguma propriedade pode ser útil no problema de reconhecimento de uma classe de grafos, dentre outras aplicações.

Os grafos cordais são um bom exemplo de uma classe de grafos em que análises dessa natureza são bastante úteis. A sua família de subgrafos proibidos (os ciclos C_k para $k \geq 4$) pode ser usada em conjunto com os padrões proibidos em 4 vértices de ordens LBFS e MCS para mostrar a corretude dos algoritmos de reconhecimento de grafos cordais em tempo linear. Essa análise foi discutida em detalhes na primeira parte deste trabalho [6]. Além disso, as ordens de eliminação perfeitas também podem ser descritas de forma simples em termos de um padrão proibido, como é mostrado na Proposição 2 (veja [7] para uma discussão mais detalhada sobre o uso de padrões proibidos para outras classes de grafos).

Proposição 2. Uma ordem $v_1, v_2, \dots, v_n$ dos vértices de um grafo G é uma ordem de eliminação perfeita se, e somente se, não existem índices $i < j < k$ tais que $v_iv_j \in E(G)$ e $v_iv_k \in E(G)$, mas $v_jv_k \notin E(G)$.

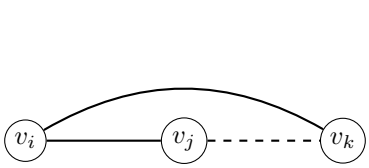

Figura 1: Padrão proibido para ordens de eliminação perfeita. As arestas sólidas representam arestas presentes no grafo, enquanto a aresta tracejada representa uma aresta ausente.

O Lema 3 apresenta uma família de padrões proibidos para ordens de vizinhança gêmea, inspirada pela proposição anterior, sendo mais uma caracterização relacionada a grafos ordenáveis por vizinhança gêmea.

Lema 3. *Uma ordem $v_1, v_2, \ldots, v_n$ dos vértices de um grafo G é uma ordem de vizinhança gêmea se, e somente se, a ordem não contém nenhum dos seguintes padrões proibidos:*

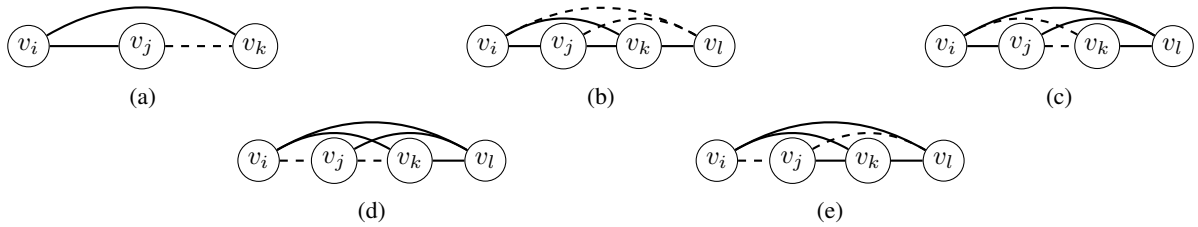

Figura 2: Padrões proibidos para ordens de vizinhança gêmea. As arestas sólidas representam arestas presentes no grafo, enquanto as arestas tracejadas representam arestas ausentes. Os vértices estão ordenados da esquerda para a direita pelos índices $i < j < k < l$.

Demonstração. A ida é imediata, pois cada padrão proibido é uma testemunha de que a vizinhança à direita de v_i não é composta apenas de gêmeos verdadeiros. A demonstração da volta é também uma simples análise de casos. Basta assumir que a vizinhança à direita de v_i não é composta apenas de gêmeos verdadeiros e analisar os casos possíveis. Caso $N_{G_i}(v_i)$ não seja uma clique, então é possível encontrar o padrão (a). Caso contrário, é necessário que exista um vértice $w \notin N_{G_i}(v_i)$ que seja vizinho de algum vizinho à direita de v_i, mas não de todos. Nesse caso, basta considerar todas as ordenações entre 4 vértices: v_i, w, um vizinho de ambos v_i e w, e um vizinho de w que não é vizinho de v_i. A ordenação desses 4 vértices irá ser igual a um dos padrões proibidos (b), (c), (d) ou (e), ou conter o padrão (a). □

Por sua vez, grafos ordenáveis por vizinhança gêmea também possuem uma família de subgrafos proibidos, que é apresentada no Teorema 4. Esse teorema é importante para uma caracterização estrutural dos grafos $\mathcal{Y}$ que será discutida posteriormente, embora os detalhes dessas caracterizações não sejam o foco deste trabalho.

Teorema 4 (Vieira e dos Santos 2025 [8])**.** *Um grafo G pertence à classe $\mathcal{Y}$ se, e somente se, G é:*

- *livre de C_k para todo $k \geq 4$,*
- *livre de DEP$_k$ para todo $k \geq 0$, e*
- *livre de Gema.*

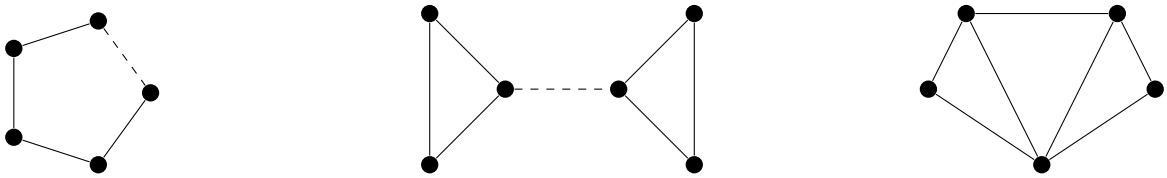

Figura 3: Ciclo C_{4+k} (esquerda), *double-ended paw* DEP$_k$ (centro) e gema (direita). Uma linha tracejada representa um caminho com $k \geq 0$ arestas.

A caracterização estrutural dos grafos $\mathcal{Y}$ depende da definição a seguir. Um grafo G é *nested split* se existe uma partição split (K, I) de G tal que para todo par de vértices $u, v \in I$, vale que $N_G[u] \subseteq N_G[v]$, $N_G[v] \subseteq N_G[u]$ ou $N_G[u] \cap N_G[v] = \emptyset$. Ou seja, as vizinhanças de vértices em I são aninhadas ou disjuntas. O Teorema 5 apresenta a caracterização estrutural, que é fundamental para o nosso algoritmo de reconhecimento.

Teorema 5 (Vieira e dos Santos 2025 [8])**.** *Um grafo G pertence a $\mathcal{Y}$ se, e somente se:*

1) *G é um nested split; ou*

2) *Existe um vértice pendente u tal que $G \setminus \{u\} \in \mathcal{Y}$; ou*
3) *Existem $G_1, G_2 \in \mathcal{Y}$ tais que $G = G_1 \cup G_2$.*

Uma consequência direta do Teorema 5 é que o conjunto de vértices de um grafo conexo $G \in \mathcal{Y}$ pode ser particionado em (K, I, T), onde (K, I) é uma nested partition de $G[K \cup I]$ e os vértices de T podem ser arranjados em uma ordem $(t_1, t_2, \ldots, t_{|T|})$ tal que t_i é um vértice pendente em $G[K \cup I \cup \{t_1, t_2, \ldots, t_i\}]$ para todo i $(1 \leq i \leq |T|)$. Uma partição satisfazendo essas propriedades é chamada de uma partição admissível para $\mathcal{Y}$.

Intuitivamente, os vértices pendentes de G podem ser removidos iterativamente até que reste um grafo G' que seja nested split. Note que, para alguns grafos, a escolha de G' nesse processo não é única, pois vértices de grau 1 podem ser atribuidos a I ou a T. Isso nos mostra que a partição (K, I, T) não é única, o que também é evidenciado pelo fato de grafos nested split possivelmente admitirem múltiplas partições (K, I). Apesar disso, para o algoritmo de reconhecimento, será útil considerar uma partição tal que o conjunto K seja uma clique máxima de G. O Lema 6 garante que essa partição existe.

Lema 6. *Todo grafo conexo $G \in \mathcal{Y}$ admite uma partição (K, I, T) tal que $|K| = \omega(G)$.*

Demonstração. Se $\omega(G) \leq 2$, é fácil ver que $(\{u, v\}, \emptyset, V(G) \setminus \{u, v\})$ é uma partição admissível para $\mathcal{Y}$ para qualquer aresta $uv \in E(G)$.

Seja (K', I', T') uma partição que comprova o pertencimento de G em $\mathcal{Y}$. Vamos mostrar que $|K'| \geq \omega(G) - 1$.

Primeiramente, note que é impossível que $|K'| \leq \omega(G) - 3$, pois existiria uma clique de tamanho 3 em $G[I \cup T]$, o que é impossível.

Mostraremos agora que $|K'| = \omega(G) - 2$ também é impossível. Seja K a clique máxima de G. Considere os vértices u, v tais que $K' \cup \{u, v\} = K$. Como $\omega(G) \geq 3$, existe $w \in K'$ tal que $\{u, v, w\}$ é um triângulo. É impossível que $u \in I' \wedge v \in I'$, pois não existe aresta entre vértices de I. Também é impossível que $u \in T' \vee v \in T'$, uma vez que os vértices do triângulo demonstram que é impossível encontrar alguma ordem tal que u (ou v) sejam um vértice pendente no subgrafo correspondente.

Considere então uma partição (K', I', T') tal que $K' = K \setminus \{u\}$. É fácil ver que $u \in I'$ e que $(K' \cup \{u\}, I' \cup \{u\}, T')$ também é uma partição admissível para $\mathcal{Y}$, pois mover u para a clique não viola a restrição de vizinhanças aninhadas no conjunto independente. □

IV. Reconhecimento

O reconhecimento de grafos em $\mathcal{Y}$ pode ser feito da maneira canônica para reconhecer uma classe de grafos baseada em ordens de eliminação. Considere um grafo G e uma propriedade P que vértices podem satisfazer. Deseja-se encontrar uma P-ordem de eliminação de G ou determinar que não é possível. Para isso, o algoritmo funcionará da seguinte maneira:

1) A primeira etapa consiste em computar uma ordem dos vértices do grafo de entrada G através de um algoritmo que retorna uma P-ordem de eliminação se G possui uma.
2) A segunda etapa verifica se a ordem obtida no passo (1) de fato é uma P-ordem de eliminação.

Em [8], é apresentado um algoritmo de reconhecimento linear de grafos em $\mathcal{Y}$ que segue essa abordagem. Em particular, a segunda etapa desse algoritmo é capaz de determinar se uma ordem qualquer é uma ordem de vinhanhança gêmea. Porém, a primeira etapa não utiliza nenhum dos métodos de buscas tradicionais, o que motivou o desenvolvimento do algoritmo que será apresentado nesta seção. Note que focaremos apenas na primeira etapa, pois a segunda etapa do algoritmo apresentado em [8] pode ser usada sem alterações.

Rose, Tarjan e Lueker foram os primeiros a apresentar um algoritmo de reconhecimento linear de grafos cordais, usando essa abordagem e definindo a busca LBFS [3] para realizar a etapa (1). Posteriormente, Tarjan e Yannakakis apresentaram um outro algoritmo de busca mais simples, chamado MCS, que também é suficiente para a etapa (1) [10].

Desde então, variações desses algoritmos de busca têm sido usadas para reconhecer outras classes de grafos. Na primeira parte deste trabalho [6], apresentamos esses algoritmos de diferentes maneiras, partindo dos algoritmos clássicos de busca em largura (BFS) e profundidade (DFS), com o intuito de analisá-los de uma forma unificada, como apresentado em [11], [12].

É útil convencionar que um algoritmo de busca $\mathcal{A}(G)$ é um procedimento que visita os vértices de um grafo G em uma ordem específica e retorna essa ordem. Quando apenas uma passada é suficiente para alguma aplicação, é comum que hajam empates na escolha do próximo vértice a ser visitado, e esses empates sejam resolvidos de maneira arbitrária. No entanto, alguns algoritmos de reconhecimento precisam realizar mais de uma passada de um algoritmo de busca, utilizando as ordens obtidas em passadas anteriores para resolver os empates das próximas passadas. Dessa forma, definimos então como $\mathcal{A}^+(G, \alpha)$ um algoritmo de busca $\mathcal{A}$ que recebe um grafo G e uma ordem total α e retorna uma ordem de visitação dos vértices de G, resolvendo os empates de acordo com α.

Para reconhecer grafos em $\mathcal{Y}$, usaremos duas passadas de uma busca em largura lexicográfica. Para isso, será necessário obter, além da ordem de visitação dos vértices, um vértice x de maior rótulo lexicográfico na ordem retornada. Para isso, é necessário fazer uma mudança simples na busca em largura lexicográfica (LBFS). O algoritmo 1 define essa variação.

Algoritmo 1: Busca em largura lexicográfica com desempate extra — LBFS$^+(G, \alpha)$

Dados: Um grafo G e uma ordem total α.
Resultado: Uma ordem σ de visitação dos vértices de G e um vértice x de maior |rótulo(x)| na ordem σ.

para $v \in V(G)$ **faça**
  rótulo(v) $\leftarrow \emptyset$
fim
$x \leftarrow \infty$
para i *de* 1 *até* $|V(G)|$ **faça**
  dentre os vértices não-visitados com rótulo lexicograficamente maximal, seja u aquele com menor $\alpha^{-1}(u)$
  marque u como visitado
  se $x = \infty$ *ou* |*rótulo*(u)| > |*rótulo*(x)| **então**
    $x \leftarrow u$
  fim
  $\sigma^{-1}(u) = i$
  para cada $v \in N_G(u)$ *não-visitado* **faça**
    rótulo(v) $\leftarrow$ rótulo(v) $\cup \{|V(G)| - i\}$
  fim
fim
retorna (σ, x)

Considere então um grafo G, em que se deseja reconhecer se $G \in \mathcal{Y}$. Seja ι uma ordem total arbitrária de seus vértices. Para obter uma ordem candidata a ser uma ordem de vizinhança gêmea, rodamos a LBFS$^+$ duas vezes, da seguinte maneira:

- Seja σ a ordenação obtida a partir de LBFS$^+(G, \iota)$, x o vértice de maior rótulo e i o índice $\sigma^{-1}(x)$ de x na ordenação σ.
- Definimos ζ tal que $\zeta^{-1}(x) = 1$ e $\zeta^{-1}(u) = \sigma^{-1}(u) + 1$ para todo vértice $u \neq x$.
- Por fim, seja γ o reverso da ordenação obtida a partir de LBFS$^+(G, \zeta)$.

Teorema 7. *Se $G \in \mathcal{Y}$, então a ordenação γ é uma ordem de vizinhança gêmea de G.*

Demonstração. Seja (K, I, T) uma partição de $V(G)$ admissível para $\mathcal{Y}$, garantida pelo Lema 6. Basta demonstrar que a condição de vizinhança gêmea é satisfeita para os vértices de cada uma das partes. É importante notar que, caso $|K| \leq 2$, então qualquer OEP também é uma ordem de vizinhança gêmea. Portanto, assumimos que $|K| \geq 3$. Além disso, é possível assumir que $d_{G[K \cup I]}(u) \geq 2$ para todo vértice $u \in I$, pois caso contrário, u seria um vértice pendente em $G[K \cup I]$ e poderia ser movido para T.

Primeiramente, destacamos que o reverso de toda ordem LBFS é uma OEP se G é cordal [3]. Portanto, o reverso de σ é uma OEP. Consequentemente, γ também é uma OEP.

Com isso, é fácil mostrar que a vizinhança à esquerda do vértice x em σ é uma clique máxima de G. Considere então a construção de ζ. Ao definir que x é o primeiro vértice da ordenação ζ garantimos que x será o último vértice de γ. Além disso, temos também que os vizinhos de x que também estão em K ocorrerão antes dos outros vizinhos de x na ordenação ζ, por construção. Portanto, os $|K|$ últimos vértices de γ são exatamente os vértices de K, e eles satisfazem a propriedade Y por serem simpliciais em $G[K]$.

Agora considere um vértice $u \in T$. Uma consequência da caracterização estrutural é que $N_G(u)$ é um conjunto independente, pois caso contrário não seria possível encontrar uma ordem válida para remoção de vértices pendentes em T. Portanto, como γ é uma OEP, segue que a vizinhança à direita de u em γ tem tamanho 1, e portanto, essa vizinhança é gêmea.

Resta mostrar que os vértices do conjunto independente também satisfazem a propriedade Y. Seja $u \in I$ e i o índice tal que $\gamma^{-1}(u) = i$. Primeiramente, note que $N_{G_i}(u) \subseteq K$, pois γ é uma OEP. Além disso, também temos que $N^2_{G_i}[u] \cap T = \emptyset$, pois potenciais vizinhos de um vértice $v \in K \cap N_{G_i}(u)$ que também estejam em T teriam apenas esse vértice como vizinho, enquanto u teria pelo menos 2 vizinhos em K, o que contradiria a escolha de um vértice com rótulo lexicograficamente maximal em uma posição da ordenação obtida pela LBFS$^+$.

Por fim, falta apenas mostrar que os vértices de I em $N^2_{G_i}(u)$ não são testemunha de que a vizinhança à direita de u não é gêmea. Para isso, considere um vértice $v \in I \cap N^2_{G_i}(u)$ tal que $\gamma^{-1}(v) > i$. Pela definição de nested split, sabemos que $N_{G[K \cup I]}(u) \subseteq N_{G[K \cup I]}(v)$ ou $N_{G[K \cup I]}(v) \subseteq N_{G[K \cup I]}(u)$. Entretanto, para que v fosse uma testemunha de que a vizinhança à direita de u não é gêmea, seria necessário que $N_{G[K \cup I]}(u) \subsetneq N_{G[K \cup I]}(v)$, o que contradiria a escolha de um vértice com rótulo lexicograficamente maximal em $\gamma^{-1}(v)$, pois o rótulo de u seria maior que o de v. □

Com isso, demonstramos a corretude do algoritmo de reconhecimento de grafos em $\mathcal{Y}$ baseado na busca LBFS. O leitor atento pode ter notado que a descrição do algoritmo em 1 não exibe facilmente uma implementação em tempo linear, pois requer a manutenção e comparação de conjuntos de inteiros para cada vértice. De fato, uma implementação em tempo linear

de LBFS tipicamente não mantém os conjuntos de rótulos, mas sim uma partição de vértices em classes de equivalência, onde cada classe contém os vértices com o mesmo rótulo. Essa partição é atualizada (ou refinada) a cada iteração, e é facilmente adaptável para manter um critério de desempate extra. Para encontrar o vértice de maior rótulo, basta manter separadamente o grau à esquerda de cada vértice. Em [13], os autores apresentam uma implementação de LBFS usando essa técnica, dentre outras aplicações interessantes onde o refinamento de partições é útil. Com isso, como o algoritmo de reconhecimento de grafos em $\mathcal{Y}$ requer apenas duas passadas de LBFS, é possível implementá-lo em tempo linear.

V. Conclusão

Neste trabalho, apresentamos um algoritmo eficiente, baseado em duas passadas de uma busca em largura lexicográfica (LBFS), para o reconhecimento da classe dos grafos ordenáveis por vizinhança gêmea. Exploramos as principais propriedades estruturais dessa classe, destacando o papel das partições nested split e dos padrões proibidos em ordens de eliminação, e mostramos como essas propriedades da classe e da LBFS permitem estabelecer a corretude do método proposto.

Apesar de já possuir um algoritmo de reconhecimento linear para grafos em $\mathcal{Y}$, o uso de uma busca em largura lexicográfica para essa tarefa pode levar a um melhor entendimento da estrutura desses grafos e como eles se relacionam com outras classes de grafos que possuem uma estrutura similar.

Como exemplo de um trabalho futuro, destaca-se investigar a possibilidade de encontrar uma ordem de vizinhança gêmea em que os vértices de T apareçam antes dos vértices de I, e estes antes dos vértices de K. Observamos que uma ordem satisfazendo essa propriedade sempre existe, e esta pode ser encontrada através de uma MCS$^+$, mas a existência de uma MCS$^+$ em tempo linear é um problema em aberto. Portanto, seria interessante investigar se é possível encontrar uma ordem de vizinhança gêmea com essa propriedade sem depender de uma MCS$^+$. Outro ponto interessante seria investigar se é possível usar apenas as ordens retornadas pelas passadas do algoritmo de busca, sem alterar a ordem em passos intermediários, para encontrar uma ordem de vizinhança gêmea em tempo linear.

Referências

[1] A. Brandstädt, V. B. Le, and J. P. Spinrad, *5. Vertex and Edge Orderings*. SIAM Monographs on Discrete Mathematics and Applications, 1999, pp. 67–89. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9780898719796.ch5

[2] F. Gavril, "Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 180–187, 1972.

[3] D. J. Rose, R. E. Tarjan, and G. S. Lueker, "Algorithmic aspects of vertex elimination on graphs," *SIAM Journal on Computing*, vol. 5, no. 2, pp. 266–283, 1976. [Online]. Available: https://doi.org/10.1137/0205021

[4] A. Bretscher, D. Corneil, M. Habib, and C. Paul, "A simple linear time lexbfs cograph recognition algorithm," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 4, pp. 1277–1296, 2008. [Online]. Available: https://doi.org/10.1137/060664690

[5] D. G. Corneil, S. Olariu, and L. Stewart, "The lbfs structure and recognition of interval graphs," *SIAM Journal on Discrete Mathematics*, vol. 23, no. 4, pp. 1905–1953, 2010. [Online]. Available: https://doi.org/10.1137/S0895480100373455

[6] K. Vieira, "Algoritmos de busca em grafos baseados em ordenações parciais e seus usos no reconhecimento eficiente de classes de grafos," *Projeto Orientado em Computação*, 2024.

[7] L. Feuilloley and M. Habib, "Graph classes and forbidden patterns on three vertices," *SIAM Journal on Discrete Mathematics*, vol. 35, no. 1, pp. 55–90, 2021.

[8] K. Vieira and V. F. dos Santos, "Twin-neighborhood orderable graphs: characterizations and efficient recognition of a new subclass of chordal and dually chordal graphs," *Submitted*, 2025.

[9] N. Pardal, G. A. Durán, L. N. Grippo, and M. D. Safe, "On nested and 2-nested graphs: two subclasses of graphs between threshold and split graphs," *Matemática Contemporânea*, vol. 46, 2021. [Online]. Available: http://doi.org/10.21711/231766362020/rmc4612

[10] R. E. Tarjan and M. Yannakakis, "Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs," *SIAM Journal on computing*, vol. 13, no. 3, pp. 566–579, 1984.

[11] D. G. Corneil and R. M. Krueger, "A unified view of graph searching," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 4, pp. 1259–1276, 2008. [Online]. Available: https://doi.org/10.1137/050623498

[12] R. Scheffler, "Ready to order? : on vertex and edge orderings of graphs," doctoralthesis, BTU Cottbus - Senftenberg, 2023.

[13] M. Habib, C. Paul, and L. Viennot, "Partition refinement techniques: An interesting algorithmic tool kit," *International Journal of Foundations of Computer Science*, vol. 10, no. 02, pp. 147–170, 1999. [Online]. Available: https://doi.org/10.1142/S0129054199000125