

Extensões e caracterizações de buscas para hipergrafos com arestas múltiplas

Silvio Luiz Bragatto Boss ¹

Jair Donadelli ²

André Luiz Pires Guedes ³

Resumo: busca em grafos é uma das técnicas mais utilizadas para o projeto de algoritmos em grafos. Em tais buscas, a ordem em que os vértices de um grafo são visitados definem uma enumeração nos vértices do grafo e uma caracterização dessas enumerações que, de acordo com a estratégia usada na busca, tem se mostrado importante nas aplicações de algoritmos em grafos. Neste trabalho generalizamos o conceito de busca e as caracterizações, bem conhecidas no caso de grafos, para o caso de hipergrafo.

Palavras-chave: Algoritmos de buscas. Hipergrafos. Teoria dos grafos.

Abstract: *graph search, or graph traversal, is possibly the most used tool in design of graph algorithms. The order in which the vertices of a graph are visited during a search define a enumeration of the vertices and the characterization of these enumerations which, according to the strategy used in search, has been useful and important in some applications of graph algorithms. In this manuscript we extend the known results of characterizations of graph searches to the more general case of hypergraphs.*

Keywords: *Graph Theory. Hypergraph. Search algorithms.*

1 Introdução

Um grafo é, formalmente, um par ordenado de conjuntos finitos (V, E) tal que cada elemento de E é um subconjunto formado por dois elementos de V . Cada elemento de V é chamado de vértice do grafo e cada elemento de E é chamado de aresta do grafo. Se G denota o grafo que é definido pelo par (V, E) então escrevemos $G = (V, E)$.

Chamamos genericamente de *algoritmo de busca* em grafo qualquer algoritmo que, quando recebe um grafo $G = (V, E)$ como entrada, visita sistematicamente todos os vértices e todas as arestas do grafo G . Nesse processo, o algoritmo mantém um conjunto C dos elementos (vértices ou arestas) de um grafo que são candidatos a serem visitados e em cada passo um elemento de C é escolhido. Quando $C \subseteq V$ dizemos que a busca é *guiada por vértice* e, nesse caso, estamos interessados na sequência $v_1, v_2, \dots, v_{|V|}$ em que os vértices de V são visitados; quando $C \subseteq E$ dizemos que a busca é *guiada por aresta* e estamos interessados na sequência $e_1, e_2, \dots, e_{|E|}$ em que as arestas de E são visitadas. A política de gerenciamento de C define uma *estratégia* de busca. Em uma busca guiada por vértice sobre um grafo o algoritmo visita, a partir de um vértice inicial, um novo vértice por iterativas travessias em arestas incidentes a esse vértice. Os novos vértices descobertos são guardados em C , de onde um é escolhido para que o processo se repita até que todos os vértices tenham sido visitados. As buscas tradicionais guiadas por vértice, conhecidas por Busca em Largura (BFS, do inglês *Breadth-First Search*) e Busca em Profundidade (DFS, do inglês *Depth-First Search*), são executadas quando C é gerenciado como uma fila e uma pilha, respectivamente

¹Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Pato Branco, Paraná
{silvioboss@utfpr.edu.br}

²Centro de Matemática, Computação e Cognição, Universidade Federal do ABC, Santo André, São Paulo
{jair.donadelli@ufabc.edu.br}

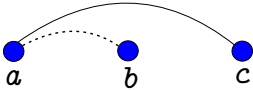
³Departamento de Informática, Universidade Federal do Paraná, Curitiba, Paraná
{andre@inf.ufpr.br}

[1]. Esses algoritmos são base para a solução de muitos algoritmos em grafos, como o algoritmo de *Prim*, para árvore geradora mínima, e o algoritmo de *Dijkstra*, para caminhos mínimos e o algoritmo de Corneil e outros para reconhecimento da classes de grafos de intervalos unitários [1][2].

Outras estratégias de busca são conhecidas na literatura. Rose, Tarjan e Lueker, introduziram, em 1976, a Busca em Largura Lexicográfica (LexBFS, *Lexicographic Breadth-first Search*) que é uma variante da busca em largura [3]. A estratégia utilizada por esse algoritmo foi usada para reconhecimento de grafos cordais. Posteriormente, essa estratégia foi utilizada para reconhecer grafos intervalares [4][5][6] e permutações bipartidas [7][6]. Mais recentemente, Corneil e Krueger [8] introduziram duas novas estratégias: Busca em Profundidade Lexicográfica (LexDFS, *Lexicographic Depth-first Search*) e Busca pela Vizinhança *Maximal* (MNS, *Maximal Neighborhood Search*), esse último usado para encontrar esquemas de eliminação perfeita e triangulações minimais em grafos arbitrários [9][10].

Tais estratégias de busca guiada por vértice são caracterizadas pela ordem em que os elementos do grafo da entrada são visitados, essa caracterização é crucial para as provas de correção dos algoritmos derivados nas aplicações dos algoritmos de busca. Essas caracterizações têm o seguinte princípio: na execução de uma busca num grafo dado, se para toda tripla de vértices a, b e c , em que a é visitado antes de b e b visitado antes de c , sendo que $\{a, c\} \in E$ e $\{a, b\} \notin E$ (veja Figura 1), como pode o vértice b ter sido enumerado antes do vértice c ? Para o caso do algoritmo LexBFS isso ocorre se, e somente se, existe um vértice d visitado antes de a tal que $\{d, b\} \in E$ e $\{d, c\} \notin E$ [11]. Corneil e Krueger estenderam essa caracterização do LexBFS para outras buscas [8].

Figura 1: Uma tripla característica com respeito a uma enumeração σ .



Fonte: Corneil e Krueger (2008) [8].

O objetivo deste trabalho é estender o estudo de Corneil e Krueger sobre algoritmos de busca em grafos [8]. Neste artigo apresentamos versões dos algoritmos de [8] para o caso de buscas guiadas por aresta em hipergrafos com arestas múltiplas e provarmos caracterizações das enumerações produzidas por tais algoritmos de busca. Tanto os algoritmos de busca quanto as propriedades e teoremas são adaptações de [8] para o caso de hipergrafos com arestas múltiplas.

2 Algoritmos de buscas e caracterizações em hipermultigrafos

Nesta seção serão apresentadas as notações utilizadas no restante do artigo, os algoritmos de busca para o caso de hipergrafos com busca guiada por aresta e as caracterizações dessas buscas por propriedades da enumeração das arestas que os algoritmos de buscas determinam.

Na próxima subseção introduzimos as notações e definições que serão utilizadas para as provas de correção dos algoritmos de buscas aqui apresentados. Seguimos as notações tradicionais na área [12][13].

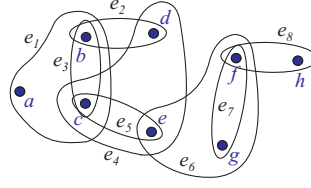
2.1 Notações e definições

Um *hipermultigrafo* H é definido por uma tripla (V, E, ι) em que V e E são conjuntos finitos e disjuntos e $\iota: E \rightarrow \wp(V)$ é a *função de incidência* de H . Os elementos do conjunto V são chamados de *vértices* e os do conjunto E de *arestas* do hipermultigrafo. A função de incidência ι associa a cada aresta e de H um subconjunto de vértices de V , são os vértices que compõem a aresta. É um exemplo de hipermultigrafo:

$$\begin{array}{l}
 V = \{a, b, c, d, e, f, g, h\}, \quad E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\} \quad \text{e} \\
 \begin{array}{c|cccccccc}
 E & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \\
 \hline
 \iota(E) & \{a, b, c\} & \{b, d\} & \{b, c\} & \{c, d, e\} & \{c, e\} & \{e, f, g\} & \{f, g\} & \{f, h\}.
 \end{array}
 \end{array}
 \tag{1}$$

Um representação gráfica desse exemplo é dada na Figura 2.

Figura 2: Representação do hipergrafo definido na equação (1).



Fonte: Elaborada pelos autores.

Notemos que a definição possibilita a existência de arestas diferentes com o mesmo conjunto de vértices, isto é, $e, f \in E$ com $\iota(e) = \iota(f)$.

Para um hipermultigrafo (V, E, ι) qualquer, para $u, v \in V$ e $e, f \in E$ usamos as abreviações

- $v \in e$ significa $v \in \iota(e)$ e $v \notin e$ significa $v \notin \iota(e)$;
- $\{u, v\} \subset e$ significa $\{u, v\} \subset \iota(e)$ e $\{u, v\} \not\subset e$ significa $\{u, v\} \not\subset \iota(e)$.
- $e \cap f$ significa $\iota(e) \cap \iota(f)$ e $e \cup f$ significa $\iota(e) \cup \iota(f)$;

ou seja, toda relação e operação de conjuntos com respeito às arestas dizem, de fato, respeito aos subconjuntos de vértices na imagem dessas arestas pela função ι . Com tais suposições, dizemos que dois vértices u e v são adjacentes em $H = (V, E, \iota)$ se $\{u, v\} \subset e$ para algum $e \in E$, duas arestas g e f são adjacentes em H se $g \cap f \neq \emptyset$. Dizemos que a aresta e é incidente aos vértices em $\iota(e) \subset V$.

Um hipermultigrafo H é dito *conexo* se para todo par de vértices $u, v \in V$ existe uma sequência (e_1, \dots, e_k) de arestas tais que: (i) $u \in e_1$; (ii) $v \in e_k$; (iii) $e_i \cap e_{i+1} \neq \emptyset$ para todo $1 \leq i < k$.

O número de vértices num hipermultigrafo H é denotado, genericamente, por n (de modo que $n = n(H)$) e o número de arestas por m (de modo que $m = m(H)$).

Uma *enumeração* das arestas de um hipermultigrafo $H = (V, E, \iota)$ é uma sequência $\sigma = (e_1, \dots, e_m)$ em que $e_i \in E$ para todo i e $m = |E|$. Se uma aresta $e \in E$ aparece na i -ésima posição da enumeração σ , então denotamos tal fato por $\sigma(e) = i$, de modo que $e = e_{\sigma(e)}$. Ademais, definimos a ordem \prec_σ sobre E pondo $e \prec_\sigma f$ se, e somente se, $\sigma(e) < \sigma(f)$, isto é, se e ocorre antes de f na enumeração σ . Nesse caso, também dizemos que a aresta e foi *enumerada antes* da aresta f , ou que e *ocorre antes* de f ou ainda que e está à *esquerda* de f em σ .

Para quaisquer três arestas a, b, c de um hipermultigrafo H com uma enumeração σ de suas arestas, dizemos que (a, b, c) é uma tripla *característica* de H com respeito a σ se:

- $a \prec_\sigma b \prec_\sigma c$,
- a e c são adjacentes, isto é, $a \cap c \neq \emptyset$ e
- a e b não são adjacentes, isto é, $a \cap b = \emptyset$.

2.2 Busca genérica

Os algoritmos descritos abaixo possuem como entrada dois parâmetros: um hipermultigrafo conexo e uma aresta pelo qual se inicia a busca. A saída é uma enumeração σ das arestas da entrada, a qual representa a ordem em que as arestas foram enumeradas.

A busca genérica é um algoritmo simples que se utiliza de uma estrutura de dados \mathcal{C} para armazenar as arestas candidatas a serem escolhidas. Tal algoritmo tem esse nome porque tem a liberdade de escolher qualquer aresta que está em \mathcal{C} , ou seja, inserção, consulta e remoção é feita sem uma política pré-definida. Assim, se e está em \mathcal{C} então será uma candidata a ser enumerada. O Algoritmo 1 descreve a busca genérica por aresta.

Algoritmo 1: Busca genérica para hipermultigrafo

Entrada: um hipermultigrafo conexo $H = (V, E, \iota)$ e uma aresta inicial $e_0 \in E$

Saída: uma enumeração σ de E

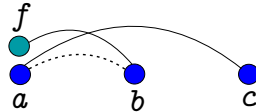
```
1  $U \leftarrow \{e_0\}$ 
2  $i \leftarrow 1$ 
3 para cada  $e$  adjacente a  $e_0$  faça
4   adicione a aresta  $e$  em  $\mathcal{C}$ 
5 enquanto  $\mathcal{C} \neq \emptyset$  faça
6   remova uma aresta  $e$  de  $\mathcal{C}$ 
7    $\sigma(e) \leftarrow i$ 
8   se  $U \neq U \cup e$  então
9     para cada aresta não enumerada  $f$  adjacente a  $e$  faça
10      se  $f$  não está em  $\mathcal{C}$  então adicione  $f$  em  $\mathcal{C}$ 
11       $U \leftarrow U \cup e$ 
12    $i \leftarrow i + 1$ 
```

Uma possível resposta do Algoritmo 1 com o hipergrafo apresentado na equação (1), representado pela Figura 2, e com aresta inicial e_1 como entradas é a enumeração $\sigma = (e_1, e_5, e_6, e_4, e_7, e_8, e_3, e_2)$. Essa enumeração não caracteriza nem BFS (dada na seção 2.3) nem DFS (dada na seção 2.4) sobre o hipergrafo, o que é permitido pelo critério de escolha na linha 6 do Algoritmo 1. Neste contexto pode-se verificar que a escolha para uma aresta é feita sem critério pré-definido pelo algoritmo. Isso ocorre pois neste trabalho não estamos considerando qualquer função que atribua um peso as arestas, o que levaria a ter um critério previamente definido. Observe que esta função poderia ser facilmente implementada nos algoritmos apresentados a seguir neste trabalho.

A Propriedade 1 e o Teorema 2, a seguir, caracterizam a busca genérica em hipermultigrafo.

Propriedade 1. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 1 se para qualquer tripla característica (a, b, c) de H com respeito a σ , existe uma aresta f tal que $f \prec_\sigma b$ e $f \cap b \neq \emptyset$.*

Figura 3: Exemplo para uma ordem de busca para a busca genérica.



Fonte: Corneil e Krueger (2008) [8].

Teorema 2. *Dado um hipermultigrafo arbitrário H , uma enumeração σ das arestas de H é uma saída de busca genérica sobre H se, e somente se, σ tem Propriedade 1.*

Demonstração. Seja σ uma enumeração dada por uma busca genérica, de acordo com o Algoritmo 1. Se (a, b, c) é uma tripla característica de arestas com respeito a σ então no momento em que a aresta b foi *enumerada* (linha 6 do Algoritmo 1) ambas as arestas b e c deviam estar em \mathcal{C} e alguma aresta adjacente a b já deve ter sido previamente escolhida, aquela que causou a inserção de b em \mathcal{C} . Denotemos essa aresta por f . Assim, f é uma aresta adjacente a b enumerada antes de b , ou seja, tal que $f \prec_\sigma b$ e $f \cap b \neq \emptyset$. Portanto, uma enumeração dada por uma busca genérica tem a Propriedade 1.

A recíproca é provada por contradição. Suponhamos que $\sigma = (e_1, \dots, e_m)$ é uma enumeração das arestas de H com a Propriedade 1. Dentre todas as enumerações das arestas de H que podem ser obtidas como resposta de uma busca genérica com entradas H e e_0 , seja $\alpha = (a_1, \dots, a_m)$ a que tem o maior número de coincidências num segmento inicial da enumeração, isto é, para a qual existe um número $i \in \{2, \dots, m+1\}$ que é máximo com respeito a propriedade:

$$a_j = e_j \text{ para todo } 1 \leq j \leq i - 1. \quad (2)$$

Notemos $i > 1$ pois podemos considerar que qualquer busca em H com aresta inicial e_0 pode começar visitando e_1 .

Deve-se supor, para a contradição, que σ é uma enumeração que não possa ser obtida como resposta de uma busca genérica no hipermultigrafo H . Então, pela escolha de α , $i - 1 < m$ e

$$\text{no momento da enumeração de } a_i \text{ pelo algoritmo vale que } e_i \notin \mathcal{C} \quad (3)$$

caso contrário, e_i poderia ser escolhida pelo algoritmo contrariando a maximalidade de i . Seja a_k , para $k < i$, a aresta adjacente a a_i que causou sua inserção em \mathcal{C} durante a execução do algoritmo com resposta α .

Se a_k é adjacente a e_i , então quando a_k é enumerada e_i é adicionada a \mathcal{C} , contrariando (3).

Se a_k não é adjacente a e_i , então (a_k, e_i, a_i) é uma tripla característica de arestas com respeito a σ donde deduzimos, pela Propriedade 1, que existe uma aresta f tal que $f \prec_\sigma e_i$ e $f \cap e_i \neq \emptyset$, ou seja, $f = a_{k'}$ para algum $k' < i$ e f é adjacente a e_i , então quando $a_{k'}$ é enumerada e_i é adicionada a \mathcal{C} , contrariando (3).

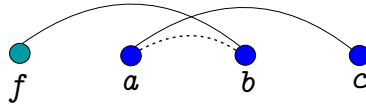
Portanto, σ é uma enumeração que pode ser obtida como resposta da execução de uma busca sobre o hipermultigrafo H . \square

2.3 Busca em largura

A busca em largura é uma restrição da busca genérica, portanto tem a mesma descrição do Algoritmo 1 exceto pelo fato de que nas linhas 4 e 10 as arestas são inseridas no fim da fila \mathcal{C} e na linha 6 a aresta é removida do começo da fila \mathcal{C} .

Propriedade 3. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 3 se para qualquer tripla característica (a, b, c) de H com respeito a σ , existe uma aresta f tal que $f \prec_\sigma a$ e $f \cap b \neq \emptyset$.*

Figura 4: Exemplo para uma ordem de busca em largura.



Fonte: Corneil e Krueger (2008) [8].

Teorema 4. *Dado um hipermultigrafo arbitrário H , uma enumeração σ das arestas de H é uma saída de busca em largura se, e somente se, σ tem a Propriedade 3.*

Demonstração. Seja σ uma enumeração dada por uma busca em largura, de acordo com o Algoritmo 1 considerando que \mathcal{C} é gerenciado como uma fila. Seja (a, b, c) uma tripla característica de arestas com respeito a σ . Disso temos que a é adjacente a c e não a b ; como a aresta b foi removida da fila antes da aresta c , deve existir uma aresta f adjacente a b que foi inserida na fila antes que a , isto é $f \prec_\sigma a$ e $f \cap b \neq \emptyset$, portanto, uma enumeração dada por uma busca em largura tem a Propriedade 3.

A recíproca é provada por contradição. Seja $\sigma = (e_1, \dots, e_m)$ uma enumeração das arestas de H com a Propriedade 3 e dentre todas as enumerações das arestas de H que podem ser obtidas como resposta de uma busca em largura sobre H , seja $\alpha = (a_1, \dots, a_m)$ a que tem o maior número de coincidências num segmento inicial: $a_j = e_j$ para todo $1 \leq j \leq i - 1$. Para a contradição, suponhamos que $i - 1 < m$. Assim, e_i é a primeira aresta da enumeração σ que uma execução de busca em largura não pôde escolher no momento certo.

Seja a_k , para $k < i$, a aresta adjacente a a_i que causou sua inserção na fila \mathcal{C} durante a execução do algoritmo com resposta α , de modo que a_k é a aresta mais a esquerda em α que é adjacente a a_i . Temos $a_k \prec_\sigma e_i$ e, também, $a_k \cap e_i = \emptyset$ senão contrariaríamos a maximalidade de α . Claramente, $e_i \prec_\sigma a_i$ logo (a_k, e_i, a_i) é uma tripla característica de arestas com respeito a σ , a qual satisfaz a Propriedade 3. Portanto, existe uma aresta f tal que $f \prec_\sigma a_k$ e $f \cap e_i \neq \emptyset$, logo $f \prec_\alpha a_k$ e $f \cap e_i \neq \emptyset$, porém, tal f implicaria na ocorrência de e_i antes que a_i

na fila \mathcal{C} , uma contradição. Portanto, σ é uma enumeração que pode ser obtida como resposta da execução de uma busca sobre o hipermultigrafo H . □

2.4 Busca em profundidade (DFS)

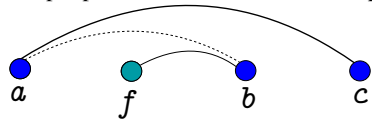
A busca em profundidade também é uma restrição da busca genérica, logo tem essencialmente a mesma descrição do Algoritmo 1, exceto pelo fato de que as arestas são inseridas e removidas no topo da pilha \mathcal{C} . Um ponto importante para termos o comportamento desejado é que uma aresta redescoberta é re-empilhada nas linhas 10 e 11 do Algoritmo 2.

Algoritmo 2: Busca em profundidade em hipermultigrafo	
Entrada: um hipermultigrafo conexo $H = (V, E, \iota)$ e uma aresta inicial $e_0 \in E$	
Saída: uma enumeração σ de E	
1	$U \leftarrow \{e_0\}$
2	$i \leftarrow 1$
3	para cada e <i>adjacente a</i> e_0 faça
4	empilha a aresta e em \mathcal{C}
5	enquanto $\mathcal{C} \neq \emptyset$ faça
6	desempilha a aresta e no topo de \mathcal{C}
7	$\sigma(e) \leftarrow i$
8	se $U \neq U \cup e$ então
9	para cada aresta não enumerada f <i>adjacente a</i> e faça
10	se f <i>já está na pilha</i> então desempilha f
11	empilha f
12	$U \leftarrow U \cup e$
13	$i \leftarrow i + 1$

A Propriedade 5 e o Teorema 6 caracterizam a busca em profundidade em hipermultigrafos.

Propriedade 5. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 5 se para qualquer tripla característica (a, b, c) de H com respeito a σ , existe uma aresta f tal que $a \prec_\sigma f \prec_\sigma b$ e $f \cap b \neq \emptyset$.*

Figura 5: Exemplo para uma ordem de busca em profundidade.



Fonte: Corneil e Krueger (2008) [8].

Teorema 6. *Dado um hipermultigrafo arbitrário H , uma enumeração σ das arestas de H é uma saída de busca em profundidade se, e somente se, σ tem a Propriedade 5.*

Demonstração. Seja σ uma enumeração dada por uma busca em profundidade em H , de acordo com o Algoritmo 2. Suponhamos que a tripla de arestas (a, b, c) é característica com relação a σ . Quando a aresta b foi escolhida ela estava no topo da pilha. Como b não foi a primeira aresta escolhida pela algoritmo, alguma aresta adjacente a b já foi escolhida num momento anterior. Dentre todas as arestas adjacentes a b e a esquerda de b em σ denotemos por f a aresta mais a direita em σ . Assim, $f \prec_\sigma b$ e $f \cap b \neq \emptyset$. Agora, suponha que $f \prec_\sigma a$. Se a foi escolhida depois de f , então c deve aparecer na pilha acima de b e como não existe nenhuma aresta adjacente a b antes dela que poderia ter sido escolhida, temos que c foi escolhida antes de b , pois está acima de b na pilha, uma contradição.

A recíproca é provada por contradição. Suponhamos que $\sigma = (e_1, \dots, e_m)$ é uma enumeração das arestas de H com a Propriedade 5, mas não é resultado de busca em profundidade em H . Dentre todas as enumerações das arestas de H que podem ser obtidas como resposta de uma busca em profundidade sobre H , seja $\alpha = (a_1, \dots, a_m)$ a que tem o maior número de coincidências num segmento inicial: $a_j = e_j$ para todo $1 \leq j \leq i - 1$. Para a contradição, suponhamos que $i - 1 < m$. Assim, e_i é a primeira aresta da enumeração σ que uma execução de busca em profundidade não pôde escolher no momento certo.

Seja e_j a aresta mais a direita em σ com $j < i$ e adjacente a a_i . Claramente, $e_j \cap e_i = \emptyset$ e mais especificamente, $e_k \cap e_i = \emptyset$, para todo $k, j \leq k < i$. Aplicando a Propriedade 5 para a tripla característica de arestas (e_j, e_i, a_i) com respeito a σ , sabemos que existe uma aresta f entre as arestas e_j e e_i tal que, $f \cap e_i \neq \emptyset$, tendo assim, uma contradição. Portanto, σ é uma enumeração que pode ser obtida como resposta da execução de uma busca sobre o hipermultigrafo H . □

2.5 Busca pela vizinhança maximal

O princípio para a estratégia de busca pela vizinhança maximal é que a cada passo no algoritmo escolhemos uma aresta f na qual o conjunto de todas as arestas que incidem em f (a vizinhança) e que já estão enumeradas é maximal com respeito a ordem parcial \subseteq de inclusão de conjuntos.

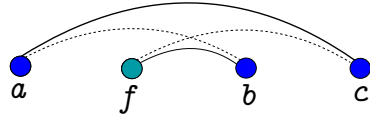
O Algoritmo 3 descreve a busca pela vizinhança maximal guiada por aresta.

Algoritmo 3: Busca pela vizinhança maximal para hipermultigrafo	
Entrada: um hipermultigrafo conexo $H = (V, E, \iota)$ e uma aresta inicial $e_0 \in E$	
Saída: uma enumeração σ de E	
1	Atribua o rótulo \emptyset para todas as arestas de H
2	escolha uma aresta e adjacente a e_0
3	$\text{rótulo}(e) \leftarrow \text{rótulo}(e) \cup \{m\}$
4	$U \leftarrow \{e_0\}$
5	$i \leftarrow 1$
6	para cada e adjacente a e_0 faça
7	$\text{rótulo}(e) \leftarrow \text{rótulo}(e) \cup \{i\}$
8	para i de 1 até m faça
9	escolha aresta não enumerada f com rótulo maximal
10	$\sigma(e) \leftarrow i$
11	se $U \neq U \cup e$ então
12	para cada aresta não enumerada f adjacente a e faça
13	$\text{rótulo}(f) \leftarrow \text{rótulo}(f) \cup \{i\}$
14	$U \leftarrow U \cup e$

A Propriedade 7 e o Teorema 8 caracterizam a busca pela vizinhança maximal em hipermultigrafos.

Propriedade 7. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 7 se para qualquer tripla característica (a, b, c) de arestas de H , com respeito a σ , existe uma aresta $f \prec_\sigma b$ tal que $f \cap b \neq \emptyset$ e $f \cap c = \emptyset$.*

Figura 6: Exemplo para uma ordem de busca pela vizinhança maximal.



Fonte: Corneil e Krueger (2008) [8].

Teorema 8. *Dado um hipermultigrafo arbitrário H , uma enumeração σ das arestas de H é uma saída de busca pela vizinhança maximal se, e somente se, σ tem a Propriedade 7.*

Abaixo denotamos por $\text{rótulo}_a(b)$ o conjunto rótulo(b) na rodada do algoritmo em que a aresta a foi enumerada.

Demonstração. Seja σ uma enumeração das arestas do hipermultigrafo H dada por uma busca pela vizinhança maximal, de acordo com o Algoritmo 3, e (a, b, c) uma tripla característica de arestas de H . Suponhamos, para contradição, que a tripla viola a Propriedade 7, isto é, suponhamos que não exista $f \prec_\sigma b$ tal que $f \cap b \neq \emptyset$ e $f \cap c = \emptyset$. Se esse é o caso então $\text{rótulo}_b(b) \subsetneq \text{rótulo}_b(c)$, assim b não poderia ter sido escolhida pela busca porque não seria maximal, o que é uma contradição.

Para provarmos a recíproca, suponhamos que $\sigma = (e_1, \dots, e_m)$ é uma enumeração das arestas de H que tem a Propriedade 7, mas não é uma enumeração que possa ser obtida como resposta da execução de uma busca pela vizinhança maximal sobre o hipermultigrafo H . Como argumentamos antes, nas provas dos teoremas anteriores, consideremos i tal que (e_1, \dots, e_{i-1}) uma subsequência máxima que pode ser obtida durante uma execução do Algoritmo 3. Se e_i não pode ser escolhida então devemos ter

$$\text{rótulo}_h(e_i) \subsetneq \text{rótulo}_h(h) \tag{4}$$

e para qualquer aresta x tal que $\sigma(x) \in \text{rótulo}_h(h) \setminus \text{rótulo}_h(e_i)$ temos x adjacente a h mas não a e_i , logo a tripla de arestas (x, e_i, h) é característica com respeito a σ . Pela Propriedade 7, deduzimos uma aresta $f \prec_\sigma e_i$ adjacente a e_i e não adjacente a h contrariando a inclusão (4). Portanto e_i pode ser escolhida, mas isso contraria o fato de (e_1, \dots, e_{i-1}) ser uma subsequência máxima que pode ser obtida durante uma execução do Algoritmo 3 com entrada H . Portanto, σ é uma enumeração que pode ser obtida como resposta da execução de uma busca pela vizinhança maximal sobre o hipermultigrafo H . □

Como casos particulares dessa busca temos as buscas lexicográficas descritas nas próximas seções. Elas são obtidas quando especificamos como comparar duas vizinhanças que são incomparáveis com respeito a ordem de inclusão de conjuntos. Na busca em profundidade damos preferência para as arestas adjacentes a mais arestas antigas na enumeração e na busca em largura, a que tem mais adjacências com as arestas mais recentes.

2.6 Busca em largura lexicográfica

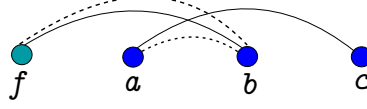
O Algoritmo 4 descreve a busca em largura lexicográfica guiada por aresta. Durante a execução, cada aresta está rotulada com uma sequência numérica e a escolha de arestas é decidida comparando as sequências de acordo com a ordem lexicográfica.

Algoritmo 4: Busca em largura lexicográfica para hipermultigrafo	
Entrada: um hipermultigrafo conexo $H = (V, E, \iota)$ e uma aresta inicial $e_0 \in E$	
Saída: uma enumeração σ de E	
1	atribua o rótulo 0 para todas as arestas de H
2	$U \leftarrow \{e_0\}$
3	$i \leftarrow 1$
4	para cada e adjacente a e_0 faça
5	acrescente o sufixo m no rótulo da aresta e
6	para i de 1 até m faça
7	pegue uma aresta não enumerada e com maior rótulo lexicográfico
8	$\sigma(e) \leftarrow i$
9	se $U \neq U \cup e$ então
10	para cada aresta não enumerada f adjacente a e faça
11	acrescente o sufixo $(m - i)$ no rótulo da aresta f
12	$U \leftarrow U \cup e$

A Propriedade 9 e o Teorema 10 caracterizam a busca em largura lexicográfica em hipermultigrafo.

Propriedade 9. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 9 se para qualquer tripla característica (a, b, c) de H com respeito a σ , existe uma aresta $f \prec_{\sigma} a$ tal que $f \cap b \neq \emptyset$ e $f \cap c = \emptyset$.*

Figura 7: Exemplo para uma ordem de busca em largura lexicográfica.



Fonte: Corneil e Krueger (2008) [8].

Teorema 10. *Dado um hipermultigrafo arbitrário H , uma enumeração das arestas de H é uma saída de busca em largura lexicográfica se, e somente se, vale a Propriedade 9.*

Demonstração. Seja σ uma enumeração das arestas de H dada por uma busca em largura lexicográfica, de acordo com o Algoritmo 4. Seja (a, b, c) uma tripla característica de arestas com respeito a σ . Quando a aresta a foi enumerada acrescentou um rótulo a c e não a b e para que b seja enumerada antes de c deve haver uma aresta f enumerada antes de a , i.e. $f \prec_{\sigma} a$, adjacente a b e não a c , para que b tenha rótulo lexicograficamente maior que c .

Para a recíproca, suponhamos por contradição, que $\sigma = (e_1, \dots, e_m)$ é uma enumeração das arestas de H que tem a Propriedade 9, mas que não pode ser obtida como resposta de uma busca em largura lexicográfica sobre H . Como antes, supomos que, para algum $2 \leq i \leq m$, (e_1, \dots, e_{i-1}) é a maior subsequência comum com σ que pode ser obtida por uma execução do Algoritmo 4 com entrada H .

Se $h \neq e_i$ é a próxima aresta escolhida na execução, então deve existir uma aresta a esquerda de e_i em σ e adjacente a h mas não a e_i . Tomemos g como tal aresta que está mais à direita em (e_1, \dots, e_{i-1}) e, desse modo temos que (g, e_i, h) é uma tripla característica com respeito a σ . Aplicando a Propriedade 9 para a tripla concluímos que deve existir uma aresta f à esquerda de g e adjacente a e_i mas não a h .

Quando f foi enumerada, acrescentou $m - \sigma(f)$ como sufixo de e_i e não o fez com h . Quando g foi enumerada, depois de f , acrescentou $m - \sigma(g)$ como sufixo de h e não o fez com e_i . Ademais, g é a última antes de h com essa característica, isto é, a partir de g até a visita a h , cada aresta enumerada sufixa e_i mas não h , ou sufixa ambas, ou não sufixa nenhuma. Também, notemos que $m - \sigma(g) < m - \sigma(f)$. Desse modo, e_i deve ter um rótulo lexicograficamente maior que h . Portanto $h = e_i$, uma contradição. Mas e_i poder ser escolhida contradiz o fato (e_1, \dots, e_{i-1}) ser a maior subsequência comum com σ que pode ser obtida por uma execução do Algoritmo. Portanto, σ é uma enumeração que pode ser obtida como resposta da execução de uma busca em largura lexicográfica sobre o hipermultigrafo H . \square

2.7 Busca em profundidade lexicográfica

O Algoritmo 5 descreve a busca em profundidade lexicográfica guiada por aresta.

A Propriedade 11 e o Teorema 12 caracterizam a busca em profundidade lexicográfica em hipermultigrafos.

Propriedade 11. *Seja σ uma enumeração das arestas de um hipermultigrafo H . Dizemos que σ tem a propriedade 11 se para qualquer tripla característica (a, b, c) de arestas H com respeito σ , existe uma aresta f com $a \prec_{\sigma} f \prec_{\sigma} b$ tal que $f \cap b \neq \emptyset$ e $f \cap c = \emptyset$.*

Teorema 12. *Dado um hipermultigrafo arbitrário H , uma enumeração das arestas de H é uma saída de busca em profundidade lexicográfica se, e somente se, vale a Propriedade 11.*

Na demonstração a seguir, usar-se-á $\text{rótulo}_b(a)$ para denotar a sequência numérica que rotula a aresta a imediatamente antes da enumeração da aresta b durante uma execução da busca. Também, escreve-se \preceq_L para denotar a ordem lexicográfica.

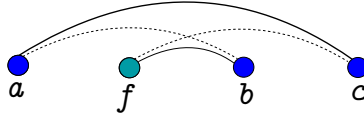
Algoritmo 5: Busca em profundidade lexicográfica para hipermultigrafo

Entrada: um hipermultigrafo conexo $H = (V, E, \iota)$ e uma aresta inicial $e_0 \in E$

Saída: uma enumeração σ de E

```
1 atribua o rótulo 0 para todas as arestas de  $H$ 
2  $U \leftarrow \{e_0\}$ 
3  $i \leftarrow 1$ 
4 para cada  $e$  adjacente a  $e_0$  faça
5   acrescente  $i$  com prefixo ao rótulo da aresta  $e$ 
6 para  $i$  de 1 até  $m$  faça
7   escolha uma aresta não enumerada  $e$  com maior rótulo lexicográfico
8    $\sigma(e) \leftarrow i$ 
9   se  $U \neq U \cup e$  então
10    para cada aresta não enumerada  $f$  adjacente a  $e$  faça
11      acrescente  $i$  com prefixo ao rótulo da aresta  $f$ 
12     $U \leftarrow U \cup e$ 
```

Figura 8: Exemplo para uma ordem de busca em profundidade lexicográfica.



Fonte: Corneil e Krueger (2008) [8].

Demonstração. Seja σ uma enumeração das arestas de H dada por uma busca em profundidade lexicográfica, de acordo com o Algoritmo 5. Se (a, b, c) é uma tripla característica de arestas com respeito a σ então para que a aresta b tenha sido escolhida antes de c deve valer $\text{rótulo}_b(c) \preceq_L \text{rótulo}_b(b)$.

Na rodada do algoritmo em que a foi enumerada, o rótulo de c ganhou o prefixo $\sigma(a)$, o que não ocorre com o rótulo de b pois essa aresta não é adjacente a a , então o rótulo de b deve conter algum outro prefixo i maior que $\sigma(a)$ e que não ocorre no rótulo de c . Seja f a aresta que atribuirá o valor i ao rótulo da aresta b , $\text{rótulo}_i(b)$. Claramente, $f \cap b \neq \emptyset$ e $f \cap c = \emptyset$. Ademais, como a aresta f atribuirá um prefixo para b maior que aquele que aresta a atribuiu para c , temos $a \prec_\sigma f$. Assim verificamos que a Propriedade 11 é satisfeita.

Por contradição, suponha que $\sigma = (e_1, \dots, e_m)$ é uma enumeração das arestas de H que tem Propriedade 11, mas não é uma ordem de busca em profundidade lexicográfica de H . Então, por argumento idêntico aos anteriormente dados, podemos assumir que existe algum $2 \leq i \leq m$, tal na execução do Algoritmo 5 com entrada H a subsequência (e_1, \dots, e_{i-1}) ocorre mas (e_1, \dots, e_i) não pode. Se h é a próxima aresta que a busca em profundidade lexicográfica escolhe, então $\text{rótulo}_h(e_i) \preceq_L \text{rótulo}_h(h)$. Seja j o maior valor no $\text{rótulo}_h(h)$ e que não ocorre em $\text{rótulo}(e_i)$. Isso é, e_j é a aresta mais à direita em (e_1, \dots, e_{i-1}) com $e_j \cap h \neq \emptyset$ e $e_j \cap e_i = \emptyset$. Com isso, (e_j, e_i, h) é uma tripla característica com respeito a σ e pela Propriedade 11 sabemos que existe uma aresta f entre e_j e e_i tal que $f \cap e_i \neq \emptyset$ e $f \cap h = \emptyset$. Desde que $e_j \prec_\sigma f$, $\text{rótulo}_h(h) \preceq_L \text{rótulo}_h(e_i)$, então e_i deveria ter sido escolhida antes de h , uma contradição. \square

Foram exibidas seis estratégias de busca em hipergrafos, denotadas LexDFS, LexBFS, MNS, DFS, BFS e BG, e para cada uma delas mostrou-se uma caracterização da ordem em que as arestas são visitadas. As caracterizações têm a seguinte formulação geral: a sequência de arestas $S = (e_1, \dots, e_n)$ é a sequência com que o algoritmo X visita as arestas de um hipermultigrafo se, e somente se, a sequência S tem a propriedade P . Para determinar P , responde-se à seguinte pergunta em cada caso: se na busca executada por X a aresta a é visitada antes de b , e que é visitada antes de c , e a e c são adjacentes mas a e b não o são, então como pode o algoritmo X ter enumerado b antes de c ? Com essa questão, encontramos as propriedades que caracterizam a estratégia de X .

As propriedades que encontramos são equivalentes àquelas que caracterizam as buscas clássicas (guiadas

por vértices) em grafos como em [8]. De fato, nossa apresentação segue de perto a apresentação de [8].

3 Conclusão

As caracterizações das enumerações produzidas por estratégias de busca em hipergrafos podem ser uma ferramenta poderosa e útil para o estudo e melhor compreensão dos algoritmos, além de nos dar um melhor entendimento de como uma determinada busca revela a estrutura de um hipergrafo. Também, quando da aplicação desses algoritmos de busca em problemas computacionais, as caracterizações servem como ponto de partida nas provas de correteza de tais aplicações.

No caso de busca em grafos são conhecidas várias aplicações que usam essas buscas e que se beneficiam das respectivas caracterizações. Já no caso de hipergrafos, que acabamos de apresentar, até o momento, não identificamos aplicações. Uma possível direção nesse sentido de trabalho futuro é investigar estratégias para busca em hipergrafos *direcionados* e suas implicações em bancos de dados baseados grafos, como é o caso do *Hyper-GraphDB* [14].

Agradecimentos

Este trabalho contou com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES.

Referências

- [1] CORMEN, T. H. et al. *algoritmos: teoria e prática*. [S.l.]: Campus, 2002.
- [2] CORNEIL, D. G. et al. Simple linear time recognition of unit interval graphs. *Inform. Process. Lett.*, v. 55, n. 2, p. 99–104, 1995. ISSN 0020-0190.
- [3] ROSE, D. J.; TARJAN, R. E.; LUEKER, G. S. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.*, v. 5, n. 2, p. 266–283, 1976. ISSN 0097-5397.
- [4] HABIB, M. et al. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theoretical Computer Science*, v. 234, n. 1-2, p. 59–84, 2000. ISSN 0304-3975.
- [5] CORNEIL, D. G. A simple 3-sweep LBFS algorithm for the recognition of unit interval graphs. *Discrete Applied Mathematics*, v. 138, n. 3, p. 371–379, 2004. ISSN 0166-218X.
- [6] HELL, P.; HUANG, J. Certifying LexBFS recognition algorithms for proper interval graphs and proper interval bigraphs. *SIAM Journal on Discrete Mathematics*, v. 18, n. 3, p. 554–570 (electronic), 2004/05. ISSN 0895-4801.
- [7] CHANG, J.-M.; HO, C.-W.; KO, M.-T. LexBFS-ordering in asteroidal triple-free graphs. In: *International Symposium on Algorithms and Computation, 10. Proceedings...* London, UK: Springer-Verlag, 1999. p. 163–172. ISBN 3-540-66916-7.
- [8] CORNEIL, D. G.; KRUEGER, R. M. A unified view of graph searching. *SIAM Journal on Discrete Mathematics*, SIAM, v. 22, n. 4, p. 1.259–1.276, 2008.
- [9] BERRY, A.; KRUEGER, R.; SIMONET, G. Ultimate generalizations of LexBFS and LEX M. In: *Graph-Theoretic Concepts in Computer Science: 31st International Workshop, WG 2005, Metz, France, June 23-25, 2005*. [S.l.]: Springer Berlin Heidelberg, 2005. p. 199–213.
- [10] CORNEIL, D. G.; KRUEGER, R. M. Simple vertex ordering characterizations for graph search: (expanded abstract). In: *7th International Colloquium on Graph Theory. Proceedings...* [S.l.]: Electronic Notes in Discrete Mathematics, 2005. v. 22, p. 445–449.

- [11] BRANDSTÄDT, A.; DRAGAN, F. F.; NICOLAI, F. LexBFS-orderings and powers of chordal graphs. *Discrete Math.*, v. 171, n. 1-3, p. 27–42, 1997. ISSN 0012-365X.
- [12] BONDY, U. M. A. *Graph theory*. 3rd corrected printing.. ed. [S.l.]: Springer, 2008. (Graduate texts in mathematics 244). ISBN 1846289696,9781846289699.
- [13] BERGE, C. *Graphs and Hypergraphs*. Oxford, UK, UK: Elsevier Science Ltd., 1973. (North-Holland mathematical library, 6). ISBN 9780444103994,0444103996.
- [14] IORDANOV, B. Hypergraphdb: A generalized graph database. In: *Proceedings of the 2010 International Conference on Web-age Information Management*. Berlin, Heidelberg: Springer-Verlag, 2010. (WAIM'10), p. 25–36. ISBN 3-642-16719-5, 978-3-642-16719-5.