Algoritmos de Varredura para Listagem de Bicliques Maximais em Modelos Bipartidos de Intervalos e Arco-Circulares

Edmilson Pereira da Cruz
Universidade Federal do Paraná, Curitiba - Brazil
epcruz@inf.ufpr.br

Marina Groshaus
Universidad de Buenos Aires, Buenos Aires - Argentina
marina-groshaus@yahoo.es

André Luiz Pires Guedes
Universidade Federal do Paraná, Curitiba - Brazil
andre@inf.ufpr.br

RESUMO

Um modelo bipartido de intervalos é uma bipartição de um número finito de intervalos na reta real. Um grafo de bi-intervalos é um grafo de intersecção de um modelo bipartido de intervalos onde cada vértice corresponde a um intervalo e dois vértices compartilham uma aresta se, e somente se, ambos intervalos correspondentes possuem intersecção e não pertencem à mesma parte. Uma biclique de um grafo é um subconjunto de vértices que induz um subgrafo bipartido completo. Este artigo apresenta um algoritmo de varredura para encontrar todas bicliques maximais de um grafo de bi-intervalos a partir de seu modelo bipartido de intervalos e propõe uma modificação para grafos bi-arco-circulares.

ABSTRACT

A bipartite interval model is a bipartition of a finite number of intervals on the real line. An interval bigraph is the intersection graph of a bipartite interval model in which each vertex corresponds to an interval and two vertices share an edge if, and only if, both corresponding intervals have an intersection and do not belong to the same part. A biclique of a graph is a subset of vertices that induces a complete bipartite subgraph. This article presents a sweep line algorithm for finding all maximal bicliques of an interval bigraph from its bipartite interval model and proposes a modification for circular-arc bigraphs.

Palavras-chave

bicliques; grafos de bi-intervalos; grafos bi-arco-circulares

1. INTRODUÇÃO

Um **modelo bipartido arco-circular** é uma tripla $(A, B, \mathbb{E})$ onde A, B são conjuntos finitos disjuntos, que chamamos de partes, cujos elementos chamamos de intervalos (ou arcos) e $\mathbb{E}$ é uma ordenação total do conjunto dos extremos de cada intervalo, que definimos por $\{0,1\} \times (A \cup B)$. Seja um modelo bipartido arco-circular M, denotamos suas componentes A, B e $\mathbb{E}$ por, respectivamente, $A(M)$, $B(M)$ e $\mathbb{E}(M)$.

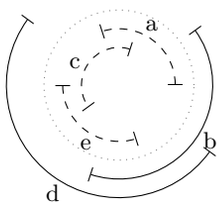

Figura 1: Modelo bipartido arco-circular

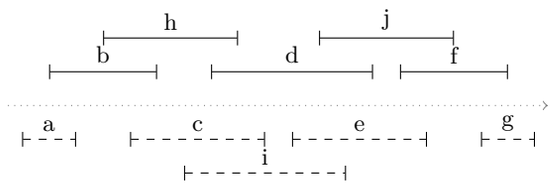

Figura 2: Modelo bipartido de intervalos

Para todo intervalo $d \in A(M) \cup B(M)$, chamamos $(0, d)$ e $(1, d)$ de, respectivamente, **evento de início de** d e **evento de fim de** d e denotamos por, respectivamente, s_d e f_d.

Dizemos que um modelo bipartido arco-circular é um **modelo bipartido de intervalos** se o evento de início de todo intervalo ocorre antes de seu respectivo evento de fim.

A Figura 1 é um exemplo de um modelo bipartido arco-circular M onde $A(M) = \{a, c, e\}$, $B(M) = \{b, d\}$ e $\mathbb{E}(M) = (s_a, f_b, s_c, f_a, s_d, s_e, f_c, s_b, f_e, f_d)$. Arcos de A estão representados em tracejado e arcos de B estão em linhas contínuas. Note que f_b ocorre antes de s_b.

A Figura 2 é um exemplo de um modelo bipartido de intervalos M onde $A(M) = \{a, c, e, g, i\}$, $B(M) = \{b, d, f, h, j\}$ e $\mathbb{E}(M) = (s_a, s_b, f_a, s_h, s_c, f_b, s_i, s_d, f_h, f_c, s_e, s_j, f_i, f_d, s_f, f_e, f_j, s_g, f_f, f_g)$. Intervalos de A estão representados em tracejado e intervalos de B estão em linhas contínuas.

Um **grafo bi-arco-circular** é um grafo de intersecção de um modelo bipartido arco-circular onde cada vértice corresponde a um intervalo e dois vértices são conectados por uma aresta se, e somente se, ambos os intervalos correspondentes

WPCCG '17 Outubro, 2017, Ponta Grossa, Paraná, Brasil

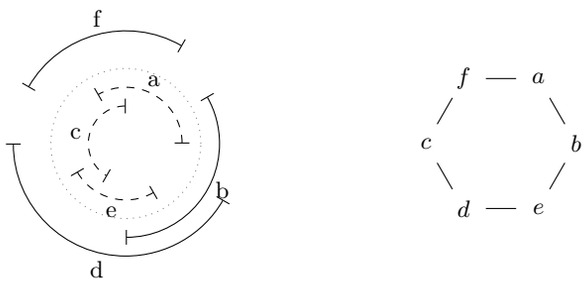

Figura 3: Grafo bi-arco-circular e seu modelo bipartido arco-circular

possuem intersecção não vazia entre si e não pertencem a mesma parte. Um grafo bi-arco-circular é **de bi-intervalos** se admite um modelo bipartido de intervalos como modelo de intersecção.

A Figura 3 é exemplo de um modelo bipartido arco-circular M, definido por $A(M) = \{a, c, e\}$, $B(M) = \{b, d, f\}$ e $\mathbb{E}(M) = (s_a, f_b, s_f, s_c, f_a, f_f, s_d, s_e, f_c, s_b, f_e, f_d)$, e seu respectivo grafo bi-arco-circular. Arcos de A estão representados em tracejado e arcos de B estão em linhas contínuas.

Uma biclique de um grafo é um subconjunto de seus vértices que induz um bipartido completo. Uma biclique S de um grafo G é dita maximal se não existir um vértice $v \in V(G) \backslash S$ tal que $S \cup \{v\}$ induz um bipartido completo.

O grafo representado na Figura 3 possui seis bicliques maximais: $\{f, a, b\}$, $\{a, b, e\}$, $\{b, e, d\}$, $\{e, d, c\}$, $\{d, c, f\}$ e $\{c, f, a\}$.

1.1 Motivação

O estudo de problemas computacionais em grafos busca melhor compreensão da relação entre as classes e a complexidade de sua solução. Dos algoritmos para listagem de bicliques maximais, podemos citar o de Alexe et al. que, embora seja exponencial, admite variantes polinomiais em relação a entrada e saída e duas classes de grafos onde o algoritmo é polinomial em relação a entrada [1]. Dias, Figueiredo e Szwarcfiter [2], Makino e Uno [4] propõem algoritmos polinomiais para certas classes de grafos. Eppstein apresenta um algoritmo de complexidade linear quando parametrizada em função da arboricidade do grafo de entrada [3].

Müller prova que C_6 é uma estrutura proibida dos grafos de bi-intervalos [5] e, a partir desse resultado, Prisner encontra um limitante superior de $(|A||B|)^2$ para o número de bicliques maximais do grafo, para as partes A e B [6].

O algoritmo apresentado neste trabalho usa uma estratégia de varredura sobre a ordem dos eventos de início e fim dos intervalos de algum modelo bipartido de intervalos para encontrar todas bicliques maximais do grafo correspondente. Acreditamos que, como a ordem dos eventos é preservada no modelo, é possível ter mais controle sobre a identificação de bicliques do que quando a entrada é um grafo.

Neste trabalho, também propomos uma modificação do algoritmo para tratar o caso de grafos bi-arco-circulares, os quais podem ser descritos como uma generalização dos grafos de bi-intervalos.

2. ALGORITMO

O algoritmo para listar todas as bicliques maximais que propomos usa uma estratégia de varredura sobre os eventos de início e de fim do modelo bipartido de intervalos pela ordem dada por $\mathbb{E}$. Uma lista de potenciais bicliques é mantida com partes que podem receber intervalos a cada evento de início e cada parte pode ser impedida de receber intervalos a medida que eventos de fim são percorridos.

Porém, o algoritmo também encontra bicliques não-maximais e subconjuntos de vértices que pertencem a uma mesma parte. Ao fim da varredura, potenciais bicliques com uma das partes vazias ou que não são maximais são removidas, restando apenas as bicliques maximais do grafo correspondente ao modelo.

2.1 Lista de Potenciais Bicliques

A lista começa com uma potencial biclique vazia com ambas as partes abertas para receber intervalos e percorre todos os eventos pela ordem definida por $\mathbb{E}$. Para cada evento de fim encontrado, é feita uma cópia de cada potencial biclique que possui o intervalo, removendo-o, e a potencial biclique original tem a parte contrária a do intervalo fechada.

Definimos as sub-rotinas como:

- INSERE(L, x): Insere elemento x ao fim da lista L;
- REMOVE(L, x): Remove elemento x da lista L;
- BICLIQUE_VAZIA(): Gera uma estrutura de potencial biclique com ambas as partes vazias e abertas;
- INTERVALO_EVENTO(e): Retorna um elemento de $A \cup B$ correspondente ao evento e;
- PARTE_ABERTA(b, p): Retorna verdadeiro se a parte p da biclique b está aberta e falso caso o contrário;
- FECHA_PARTE(b, p): Marca a parte p da biclique b como "fechada"; e
- COPIA(b): Retorna uma cópia independente da biclique b.

Algoritmo 1 Geração da lista de potenciais bicliques

Entrada: Modelo bipartido de intervalos $(A, B, \mathbb{E})$
Saída: Lista de potenciais bicliques

1: **Função** BIVSWEEPLINE$(A, B, \mathbb{E})$
2: Lista L
3: INSERE$(L$, BICLIQUE_VAZIA()$)$
4: **Para todo** $e \in \mathbb{E}$ **:**
5: Intervalo $d \leftarrow$ INTERVALO_EVENTO(e)
6: **Se** $d \in A$ **então:**
7: $p \leftarrow 1$
8: **Senão:** ▷ $d \in B$
9: $p \leftarrow -1$
10: **Se** $e \in \{0\} \times (A \cup B)$ **então:**
11: **Para todo** $l \in L$ **:**
12: **Se** PARTE_ABERTA(l, p) **então:**
13: INSERE$(l[p], d)$
14: **Senão:** ▷ $e \in \{1\} \times (A \cup B)$
15: **Para todo** $l \in L$ **:**
16: **Se** $d \in l[p]$ **então:**
17: $l' \leftarrow$ COPIA(l)
18: FECHA_PARTE$(l, -p)$
19: REMOVE$(l'[p], d)$
20: INSERE(L, l')
21: **Retorne** L

Como descrito pelo algoritmo 1, a lista das potenciais bicliques começa com uma potencial biclique vazia com ambas as partes abertas para receber novos intervalos. Eventos são percorridos pela ordem dada em $\mathbb{E}$ e cada parte é indexada na potencial biclique por 1 e -1. Se o evento encontrado for de início, seu intervalo correspondente é inserido em todas potenciais bicliques com a parte correspondente ao intervalo aberta. Se o evento for de fim, todas as potenciais bicliques que contêm o intervalo correspondente ao evento são copiadas sem o intervalo e as potenciais bicliques que foram copiadas têm suas partes contrárias correspondentes ao intervalo fechadas.

2.2 Limpeza da Lista

A limpeza da lista é uma simples comparação duas-a-duas procurando por potenciais bicliques que estejam contidas em outras. Um teste para determinar potenciais bicliques com uma das partes vazias também é feito durante as comparações.

Algoritmo 2 Limpeza da lista de bicliques maximais

Entrada: Lista de potenciais bicliques L e número de intervalos $n = |A \cup B|$
Saída: Lista de bicliques maximais

1: **Função** LIMPEZA(L, n)
2: $k \leftarrow |L|$
3: Vetor $v_1[n] \leftarrow (0, \ldots, 0)$
4: Vetor $v_2[k] \leftarrow (0, \ldots, 0)$
5: **Para todo** $l_1 \in [1..k]$ e $v_2[l_1] = 0$ **:**
6: **Se** $l_1[1] = \varnothing$ ou $l_1[-1] = \varnothing$ **então:**
7: $v_2[l_1] \leftarrow 1$
8: **Senão:**
9: **Para todo** $i \in L[l_1]$ **:**
10: $v_1[i] \leftarrow 1$
11: **Para todo** $l_2 \in [1..k]$ e $l_1 \neq l_2$ e $v_2[l_2] = 0$ **:**
12: contida $\leftarrow 1$
13: **Para todo** $i \in L[l_2]$ **:**
14: **Se** $v_1[i] = 0$ **então:**
15: contida $\leftarrow 0$
16: **Se** contida $= 1$ **então:**
17: $v_2[l_2] \leftarrow 1$
18: **Para todo** $i \in L[l_1]$ **:**
19: $v_1[i] \leftarrow 0$
20: Lista L'
21: **Para todo** $i \in k$ **:**
22: **Se** $v_2[i] = 0$ **então:**
23: INSERE($L', L[i]$)
24: **Retorne** L'

Como descrito pelo algoritmo 2, as comparações da limpeza, que são feitas duas-a-duas, marcam intervalos presentes na primeira potencial biclique e testam se todos os intervalos da segunda potencial biclique foram marcados. Havendo um intervalo não marcado, a segunda potencial biclique difere da primeira por algum intervalo não presente. Se todos os intervalos da segunda potencial biclique estão marcados, então ela está contida na primeira e seu índice na lista é marcado como "para remoção". Potenciais bicliques marcadas para remoção não são comparadas com as demais.

Por fim, a limpeza retorna a lista de todas potenciais bicliques que não foram marcadas para remoção.

2.3 Corretude

Como um dos resultados recentes do projeto, temos uma prova de que todas bicliques maximais do grafo correspondente ao modelo de entrada são encontradas pelo algoritmo. A prova de corretude se baseia na invariante de laço:

TEOREMA 1. *Seja e_i o evento contido na variável e do laço da linha 4 de* BIVSWEEPLINE *durante sua i-ésima iteração. Após a execução da i-ésima iteração, o algoritmo encontra todas bicliques maximais contidas no conjunto de todos intervalos que começam antes do último evento de fim de $(e_1, \ldots, e_i)$.*

PROVA. Seja $\mathcal{B}_i$ o conjunto de todas as bicliques maximais que possuem apenas intervalos que começam antes do último evento de fim de $(e_1, \ldots, e_i)$, para todo $i \in [1..|\mathbb{E}(M)|]$ e qualquer modelo bipartido de intervalos M. Seja também $\mathcal{B}_0 = \varnothing$.

Assumindo que o algoritmo encontra todas as bicliques maximais em $\mathcal{B}_{i-1}$, provamos que ele encontra todas as bicliques maximais em $\mathcal{B}_i$. Para isso, consideramos dois casos: e_i é evento de início e e_i é evento de fim.

Para o caso de e_i ser evento de início, provamos que o intervalo correspondente a e_i não é inserido em alguma potencial biclique de modo que alguma biclique maximal seja perdida.

Provamos isso pela seguinte redução ao absurdo: assumimos que o intervalo d correspondente a e_i é inserido na potencial biclique correspondente a biclique maximal $b \in \mathcal{B}_{i-1}$. Logo, temos que a parte de b correspondente a d estava aberta. Como b é biclique, temos que a parte contrária a de d não é vazia. Como a parte correspondente a d estava aberta, nenhum intervalo da parte contrária terminou, o que faz que d intersecte todos os intervalos da parte contrária. Logo, $b \cup \{d\}$ é biclique e, portanto, temos que b não é maximal.

Para o caso de e_i ser evento de fim, temos um novo último evento de fim percorrido em relação à $(i-1)$-ésima iteração e, portanto, mais bicliques maximais podem ter sido encontradas dependendo dos seguintes subcasos: e_i é o primeiro evento de fim percorrido, e_{i-1} também é evento de fim e e_{i-1} é evento de início.

Se e_i for o primeiro evento de fim percorrido, temos apenas uma potencial biclique e ambas as partes estão abertas até a $(i-1)$-ésima iteração. Temos que, se houver uma biclique em $\mathcal{B}_i$, ela possui apenas os intervalos da potencial biclique encontrada.

Se e_{i-1} for evento de fim, temos que $\mathcal{B}_i = \mathcal{B}_{i-1}$ e, como o algoritmo apenas fecha a parte contrária a do intervalo que termina e faz cópias removendo tal intervalo em todas as potenciais bicliques que o contêm, todas bicliques maximais encontradas até a $(i-1)$-ésima iteração não são perdidas na i-ésima iteração.

Se e_{i-1} for evento de início, mostramos que todos os intervalos que começam entre o penúltimo e o último evento de fim são inseridos nas potenciais bicliques de tal forma que todas as bicliques maximais em $\mathcal{B}_i \setminus \mathcal{B}_{i-1}$ são encontradas.

Portanto, mostramos que as potenciais bicliques que geram as bicliques maximais a partir dessas inserções já foram produzidas pelo algoritmo até a iteração do penúltimo evento de fim encontrado. Entretanto, a prova encontrada da existência de tais potenciais bicliques é muito longa para ser apresentada neste trabalho. □

Com o término do laço e como consequência da invariante

de laço, o algoritmo encontra todas bicliques maximais por ter percorrido todos eventos de início que, por lidar com modelos bipartidos de intervalos, ocorrem antes do último evento de fim.

2.4 Adaptação para o Caso Bi-Arco-Circular

Para tratar o caso bi-arco-circular, percorremos os eventos de início e fim uma segunda vez. Como as bicliques maximais podem ser "cortadas" pela entrada com eventos de fim ocorrendo antes de seu respectivo eventos de início, percorremos os eventos uma segunda vez para garantir que todo evento de fim possua um respectivo evento de início o precedendo.

Algoritmo 3 Geração da lista de potenciais bicliques adaptada para o caso bi-arco-circular

Entrada: Modelo bipartido arco-circular $(A, B, \mathbb{E})$
Saída: Lista de potenciais bicliques
1: **Função** BACSWEEPLINE$(A, B, \mathbb{E})$
2: **Retorne** BIVSWEEPLINE$(A, B, \mathbb{EE})$

A notação $\mathbb{EE}$, usada pelo algoritmo 3, representa a concatenação de $\mathbb{E}$ com $\mathbb{E}$. Como descrito pelo algoritmo, eventos de início e fim são processados da mesma forma que o caso bi-intervalo. Eventos de fim que não possuem seu respectivo evento de início o precedendo são ignorados, pois nenhuma potencial biclique possui o intervalo correspondente no momento. Intervalos que começam uma segunda vez são tratados da mesma forma que na primeira. Se eles puderem ser inseridos em potenciais bicliques que já os contêm, a segunda inserção pode ser omitida.

No caso bi-arco-circular, não apenas bicliques não-maximais são encontradas, mas também duplicatas de bicliques maximais. O algoritmo da limpeza sugerido também trata da remoção de duplicatas, pois nenhuma potencial biclique é marcada para remoção se não houver outra que a contém e que não esteja marcada.

Portanto, ainda não temos uma prova de corretude para o caso bi-arco-circular. Ainda é necessário analisar casos como bicliques cujos intervalos cobrem todo o círculo.

2.5 Complexidade

Para o cálculo de complexidade, definimos $n = |A \cup B| = \frac{|\mathbb{E}|}{2}$, para a entrada $(A, B, \mathbb{E})$, e k como o tamanho da lista de potenciais bicliques encontradas pelo algoritmo. Como é difícil prever quantas potenciais bicliques são encontradas, descrever como um valor separado do número de intervalos será conveniente para o cálculo de complexidade.

A geração da lista é feita em $O(kn^2)$, por percorrer a sequência de eventos $\mathbb{E}$ e, para cada evento, percorrer no máximo k potenciais bicliques com uma busca em até n intervalos para cada potencial biclique. Cada cópia é limitada aos n intervalos do modelo para cada potencial biclique.

Cada comparação entre bicliques é feita de forma linear limitada aos n intervalos. As potenciais bicliques são comparadas duas-a-duas, o que nos dá $O(k^2n)$ para a limpeza.

Temos a geração da lista de potenciais bicliques em $O(k^2n + kn^2) = O(kn(k + n))$.

Enquanto ainda não temos um limitante para k em função de n, suspeitamos que k seja exponencial em n. Porém, embora tenhamos tido o intuito de manter o algoritmo simples, acreditamos que é possível prever quais potenciais bicliques não serão bicliques maximais durante a varredura, evitando cópias desnecessárias e potencialmente reduzindo o limitante de k em função de n.

3. CONCLUSÕES E TRABALHOS FUTUROS

Várias estratégias foram sugeridas para a listagem de bicliques maximais em diferentes classes de grafos [3, 1, 4, 2]. Embora a estratégia de varredura sobre os eventos de início e fim do modelo bipartido de intervalos sugerida neste trabalho para grafos de bi-intervalos pareça ineficiente, ela possui uma simples adaptação que acreditamos resolver o caso bi-arco-circular. Também acreditamos que tal ineficiência não o torna irrelevante para futuras investigações das bicliques maximais em grafos de bi-intervalos.

A prova de corretude encontrada também pode contribuir para melhor entendimento da localização de bicliques maximais em modelos bipartidos de intervalos por tornar mais explícito como as bicliques maximais são descritas em termos de inserções, cópias com remoções e fechamento de partes de potenciais bicliques, levando a encontrar algoritmos de varredura mais eficientes. Porém, ainda não podemos usar os argumentos da prova do caso de bi-intervalos para o caso bi-arco-circular.

Um possível caminho para futuras investigações é determinar se a adaptação para o caso bi-arco-circular é correta, o que pode trazer algumas respostas quanto a natureza das bicliques de tais modelos. Caso seja correto, a distribuição de bicliques maximais pelos modelos bipartidos arco-circulares não é muito diferente da distribuição do caso de bi-intervalos, o que pode ser um indício de que ambas as classes não são muito distintas em relação ao problema da listagem de bicliques maximais.

Referências

[1] G. Alexe, S. Alexe, Y. Crama, S. Foldes, P. L. Hammer, and B. Simeone. Consensus algorithms for the generation of all maximal bicliques. *Discrete Applied Mathematics*, 145(1):11 – 21, 2004. Graph Optimization IV.

[2] V. M. F. Dias, C. M. H. de Figueiredo, and J. L. Szwarcfiter. On the generation of bicliques of a graph. *Discrete Applied Mathematics*, 155(14):1826–1832, 2007.

[3] D. Eppstein. Arboricity and bipartite subgraph listing algorithms. *Information processing letters*, 51(4):207–211, 1994.

[4] K. Makino and T. Uno. New algorithms for enumerating all maximal cliques. In *Scandinavian Workshop on Algorithm Theory*, pages 260–272. Springer, 2004.

[5] H. Müller. Recognizing interval digraphs and interval bigraphs in polynomial time. *Discrete Applied Mathematics*, 78(1):189 – 205, 1997.

[6] E. Prisner. Bicliques in graphs I: Bounds on their number. *Combinatorica*, 20(1):109–117, 2000.