Linear Time Algorithms on Chordal Bipartite and Strongly Chordal Graphs

Ryuhei UEHARA**

Natural Science Faculty, Komazawa University
uehara@komazawa-u.ac.jp

Abstract. Chordal bipartite graphs are introduced to analyze nonsymmetric matrices, and form a large class of perfect graphs. There are several problems, which can be solved efficiently on the class using the characterization by the doubly lexical ordering of the bipartite adjacency matrix. However, the best known algorithm for computing the ordering runs in $O(\min\{m \log n, n^2\})$, which is the bottleneck of the problems. We show a linear time algorithm that computes the ordering of a given chordal bipartite graph. The result improves the upper bounds of several problems, including recognition problem, from $O(\min\{m \log n, n^2\})$ to linear time. Strongly chordal graphs are well-studied subclass of chordal graphs, and that have similar characterization. The upper bounds of several problems on a given strongly chordal graph are also improved from $O(\min\{m \log n, n^2\})$ to linear time.
Keywords: chordal bipartite graphs, design and analysis of algorithms, lexicographic breadth first search, vertex elimination ordering, strongly chordal graphs.

1 Introduction

Chordal bipartite graphs are bipartite graphs such that every cycle of length greater than 4 has a chord. The graph class is introduced by Golumbic and Goss [11], and it has applications to nonsymmetric matrices (see [10]) including Gaussian elimination [1], integer programming [13], and matrix analysis [15, 14]. The class also forms a large class of perfect graphs (see [20]).

On the class, several problems can be solved efficiently. Most of the algorithms are based on the following characterization; a bipartite graph G is chordal bipartite if and only if the doubly lexical ordering of the bipartite adjacency matrix of G is Γ-free. However, the best known algorithm for computing the doubly lexical ordering runs in $O(\min\{m \log n, n^2\})$ time, and this upper bound is the bottleneck of the efficiency of many problems. Typically, the best known upper bound for recognition of the class is $O(\min\{m \log n, n^2\})$ [13, 19, 21, 27]. All these recognition algorithms use the same underlying idea: First compute a doubly lexical ordering of the bipartite adjacency matrix of a given bipartite graph and then check whether it is Γ-free.

** Work done while visiting University of Waterloo.

Our main result is a linear time algorithm for the bottleneck. We first introduce a vertex elimination ordering, called *weak elimination ordering* (WEO), which characterizes a chordal bipartite graph. Next we construct a linear time algorithm that generates a WEO for given chordal bipartite graph. Using the WEO, we can improve the upper bounds of the following problems on a chordal bipartite graph from $O(\min\{m \log n, n^2\})$ to linear time: (1) the recognition problem [13, 19, 21, 27], and constructing Γ-free bipartite adjacency matrix, which means that we can perform Gaussian elimination efficiently [10, 1], and (2) computing a perfect edge without vertex elimination ordering and a list of all maximal complete bipartite subgraphs [17], and (3) computing a strong T-elimination ordering and finding a maximum matching [4]. We also show that a maximum vertex-weighted matching can be found in linear time.

Next we consider strongly chordal graphs, which are originally introduced by Farber as a subclass of chordal graphs for which the domination problem can be solved efficiently [8, 9]. This class has similar characterization to the class of chordal bipartite graphs, and has the same bottleneck. We also improve the upper bounds of the following problems on a strongly chordal graph from $O(\min\{m \log n, n^2\})$ to linear time: (1) the recognition problem and computing a strong elimination ordering [13, 19, 21, 27] (see also [24]), and (2) finding a maximum (vertex-weighted) matching [3, 7], an independent dominating set and a minimum weighted dominating set [9], a minimum dominating clique [18], and a domatic partition [23].

From the viewpoint of algorithmic graph theory, it is worth remarking that our algorithm for chordal bipartite graph recognition uses the lexicographic breadth first search (LexBFS). For recognition of a chordal graph, two linear time algorithms are well known; the lexicographic breadth first search (LexBFS) [25], and the maximum cardinality search (MCS) [28]. The LexBFS has been investigated deeply [26, 22], and recently, it is used for recognition of not only chordal graphs, but also the other graph classes including interval graphs [5], distance hereditary graphs, etc (see [2, Chapter 5] for references). Our results extend the list of the graph classes, which can be recognized by LexBFS.

2 Preliminaries

Given a graph $G = (V, E)$ and a subset $U \subseteq V$, the *subgraph of G induced by U* is the graph (U, F), where $F = \{\{u, v\} | \{u, v\} \in E, \text{ and } u, v \in U\}$, and denoted by $G[U]$. A vertex set I is an *independent set* if $G[I]$ contains no edges. The *(open) neighborhood* of a vertex v in G is the set $N_G(v) = \{u \in V | \{u, v\} \in E\}$, and the *closed neighborhood* is $N_G[v] = N_G(v) \cup \{v\}$. We will omit the index G if no confusion can arise. The *degree* of a vertex v is $|N(v)|$ and denoted by $d(v)$. For a vertex set U, $N(U)$ denotes the set $\cup_{u \in U} \{v | v \in N(u)\}$. A sequence of the vertices $v_1, \cdots, v_l$ is a *path*, denoted by $(v_1, \cdots, v_l)$, if $\{v_j, v_{j+1}\} \in E$ for each $1 \leq j \leq l - 1$. The *length* of a path is the number of edges in the path. A *cycle* is a path beginning and ending with the same vertex. An edge which joins two vertices of a cycle but is not itself an edge of the cycle is a *chord* of that cycle.

A graph $G = (V, E)$ is *bipartite* if V can be divided into two sets X and Y with $X \cup Y = V$ and $X \cap Y = \emptyset$ such that every edge joins a vertex in X and another one in Y. A graph G is bipartite if and only if G contains no cycle of odd length [16]. Throughout the paper, we denote the number of vertices and edges of the input graph $G = (V, E)$ by n and m, respectively.

An *ordering* α of $G = (V, E)$ is a bijection $\alpha : V \to \{1, \cdots, n\}$. Often it will be convenient to denote an ordering by using it to index the vertex set, so that $\alpha(v_i) = i$ for $1 \leq i \leq n$ where i will be referred to as the *index* of v_i. Let $v_1, \cdots, v_n$ be an ordering of V. For $1 \leq i \leq n$, we define $\mathcal{L}_i$ to be the set of vertices with indices greater than $i - 1$: $\mathcal{L}_i = \{v_i, \cdots, v_n\}$. For given graph and ordering, we denote by G_i the graph $G[\mathcal{L}_i]$. We also denote by G_v, for vertex v, the graph $G[\mathcal{L}_{\alpha(v)}]$. For given vertices u, v we denote by $u < v$ if $\alpha(u) < \alpha(v)$, and $N_{>v}(u)$ denotes the set of neighbors of u greater than v.

Given bipartite graph $G = (V_1, V_2, E)$ with $V_1 = \{v_1, \cdots, v_{n_1}\}$ and $V_2 = \{u_1, \cdots, u_{n_2}\}$, we define *bipartite adjacency matrix* by $n_1 \times n_2$ $(0,1)$-matrix $A = (a_{ij})$ with $a_{ij} = 1$ if and only if $\{v_i, u_j\} \in E$. Given graph $G = (V, E)$ with $V = \{v_1, \cdots, v_n\}$, its *adjacency matrix* is defined by $n \times n$ $(0,1)$-matrix $A = (a_{ij})$ with $a_{ij} = 1$ if and only if either $i = j$ or $\{v_i, v_j\} \in E$. An ordered $(0,1)$-matrix is *Γ-free* if it has no ordered submatrix which is a $\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

Lemma 1 (see [13, 19] for the details). *(1) A bipartite graph is chordal bipartite if and only if there is an ordering of vertices such that its bipartite adjacency matrix is Γ-free. (2) A graph is strongly chordal if and only if there is an ordering of vertices such that its adjacency matrix is Γ-free.*

3 Characterizations of chordal bipartite graph

A *chordal bipartite graph* is the bipartite graph such that each cycle of length greater than 4 has a chord. In this section, we show two characterizations of a chordal bipartite graph. The first characterization is based on the following folklore; a bipartite graph $G = (V, E)$ is chordal bipartite if and only if G has an ordering $v_1, v_2, \cdots, v_n$ such that for each $\{x_1, y_2\}, \{x_2, y_1\}, \{x_1, x_2\} \in E$ with $x_1 < y_1$ and $x_2 < y_2$, we have $\{y_1, y_2\} \in E$. (Although we omit the proof of the folklore since it appears in literature, e.g. [6], it can be proved using Lemma 1.) A vertex v is *weak-simplicial* if (1) $N(v)$ is an independent set, and (2) for each $x, y \in N(v)$, either $N(x) \subseteq N(y)$ or $N(y) \subseteq N(x)$. A vertex ordering $v_1, v_2, \cdots, v_n$ is a *weak elimination ordering* (WEO) if each vertex v_i is weak-simplicial on G_i, and, for each $x, y \in N_{>v_i}(v_i)$, $x < y$ implies $N_{>v_i}(x) \subseteq N_{>v_i}(y)$.

Theorem 1. *A graph is chordal bipartite if and only if it admits a WEO.*

Proof. ($\Rightarrow$) Let G be a chordal bipartite graph. Then we have an ordering $v_1, v_2, \cdots, v_n$ such that for each $\{x_1, y_2\}, \{x_2, y_1\}, \{x_1, x_2\} \in E$ with $x_1 < y_1$ and $x_2 < y_2$, we have $\{y_1, y_2\} \in E$. We show the ordering is a WEO. Since G is bipartite, $N(v)$ is an independent set for each $v \in V$. Let x, y be vertices that $x, y \in N(v)$ and $v < x < y$. Let x' be any vertex in $N_{>v}(x)$. Then

$\{v,y\},\{x,x'\},\{v,x\} \in E$, and $x < y$. Thus, by the folklore, $v < x'$ implies $\{x',y\} \in E$. Therefore, any vertex x' in $N_{>v}(x)$ is adjacent to y, consequently, $N_{>v}(x) \subseteq N_{>v}(y)$.
($\Leftarrow$) Assume that an ordering $v_1, v_2, \cdots, v_n$ is a WEO of G. We first show that each cycle of length greater than 4 has a chord. We suppose that (u_1, u_2, u_3, $\cdots$, u_4, u_5, u_1) be a chordless cycle of length greater than 4. Without loss of generality, we assume that u_1 has the smallest index on the cycle. That is, G_{u_1} contains the cycle. Then, since u_1 is weak-simplicial on G_{u_1}, either $N(u_2) \subseteq N(u_5)$ or $N(u_5) \subseteq N(u_2)$. Thus we have $\{u_3,u_5\} \in E$ or $\{u_2,u_4\} \in E$, which contradicts that the cycle is chordless. Thus every cycle of length greater than 4 has a chord. We next show that G is bipartite. We assume that G has a cycle of odd length. Since every cycle of G of length greater than 4 has a chord, G contains at least one cycle of length 3. Let w,x,y be the vertices on the cycle of length 3 with $w < x < y$. Then, $N(w)$ is not independent set on G_w, which is a contradiction. Thus G is bipartite. □

We next show the second characterization, which is independent of the ordering of vertices. In a graph, a pair of two vertices x and y is said to be *two-pair* if each chordless path between x and y has exactly two edges. (The notion two-pair is originally introduced by Hayward, Hoàng, and Maffray to characterize weakly chordal graphs [12].) We here introduce two notions of *semi-clique* and *semi-simplicial*. We say a vertex set S is *semi-clique* if every pair of two vertices x and y in S is a two-pair in $G[(V-S) \cup \{x,y\}]$. Intuitively, there is no path joining $N(x)-N(y)$ and $N(y)-N(x)$ not through $N(x) \cap N(y)$. We say a vertex v is *semi-simplicial* if $N(v)$ is a semi-clique.

Theorem 2. *A graph is chordal bipartite if and only if every vertex is semi-simplicial.*

Proof. ($\Rightarrow$) Suppose that G is chordal bipartite and G contains a vertex s that is not semi-simplicial. By definition, $N(s)$ contains x and y such that they are not two-pair on $G[(V-N(s)) \cup \{x,y\}]$. Then there is a chordless path P joining x and y of length not equal to 2. If the length of P is 1, (x,s,y) and P induce a cycle of length 3, which contradicts that G is bipartite. On the other hand, if the length of P is greater than 2, (x,s,y) and P induce a chordless cycle of length greater than 4, which contradicts that G is chordal bipartite. (Note: There is no chord joining s and a vertex on P since x and y are the only neighbors of s in $G[(V-N(s)) \cup \{x,y\}]$.)
($\Leftarrow$) We first show that G has no chordless cycle of length greater than 4. Suppose that every vertex is semi-simplicial and G contains a chordless cycle of length greater than 4. We let the chordless cycle be $C = (v_1, v_2, v_3, \cdots, v_4, v_5, v_1)$. Then $\{v_2,v_5\} \notin E$, $\{v_2,v_4\} \notin E$, and $\{v_3,v_5\} \notin E$. Hence, since v_1 is semi-simplicial, v_2 and v_5 have a common neighbor on the path joining v_3 and v_4. However, this contradicts that C is chordless. Thus G has no chordless cycle of length greater than 4. Next we show that G is bipartite. We assume that G contains a cycle of odd length. Then, since G has no chordless cycle of length greater than 4, G has at least one cycle of length 3. Let x,y,s be the vertices on the cycle. However,

this contradicts that s is semi-simplicial since $\{x, y\} \in E$. Thus G is chordal bipartite. □

4 Linear time algorithm for generating WEO

In this section we first construct an algorithm GEN that generates a WEO for a given chordal bipartite graph. We next simplify the algorithm for implementation. For lack of space, linear time implementation is omitted (see [29] for details).

The LexBFS contains two phases [25]: (1) generate an ordering α of G, and (2) compute a fill-in F and check if $F = \emptyset$. GEN uses the first phase of the LexBFS. It always generates an ordering α, and α is a WEO if given graph is chordal bipartite.

Algorithm description

To describe GEN, we introduce some notions. Let α be an ordering, and $A = \{a_1, \cdots, a_k\}$ and $B = \{b_1, \cdots, b_h\}$ be two sets of vertices with $a_1 > \cdots > a_k$ and $b_1 > \cdots > b_h$. Then we say A is *lexicographically larger* (lex-larger) than B if there is an index i such that $a_j = b_j$ for all $j < i$, and either $a_i > b_i$ or $h = i - 1$. (For example, we have $\{5,4,3\} > \{5,4,2,1\} > \{5,4,2\} > \{5,4\} > \{4,3\}$.) We denote by $N^+(u)$ ($N^-(u)$) the sets of numbered (unnumbered, resp.) neighbors of u. $N^*(u)$ denotes the set of vertices defined by $N^*(u) = \cup_{u' \in N^-(u)}\{w | w \in N^+(u')\}$. GEN computes the ordering from n down to 1 as follows:

A0. the first vertex v_n is arbitrary chosen;
A1. let v be the largest numbered vertex that has unnumbered neighbors; let L be the set of unnumbered vertices in $N(v)$;
A2. number all vertices in L as follows:
  A2.1. pick up the vertex u such that u is not numbered, and $N^+(u)$ is lex-larger than any other $N^+(u')$ with unnumbered u' in L;
  A2.2. number u; set $U := \{u\}$;
  A2.3. set $W := \{w \mid w \in L$ is not numbered, and $N^-(w) \cap N^-(u') \neq \emptyset$ for some $u' \in U\}$;
  A2.4. if $W = \emptyset$ and L contains the vertices that are not numbered, go to step A2.1;
  A2.5. while W contains unnumbered vertex, pick up the vertex w such that $N^*(w)$ is lex-larger than any other $N^*(w')$ with unnumbered w' in W, ties are broken by $|N^-(w)|$ (that is, two vertices w and w' with $N^*(w) = N^*(w')$ are $w > w'$ if $|N^-(w)| > |N^-(w')|$);
  A2.6. set $U := W$ and go to step A2.3;
A3. if unnumbered vertices remain then go to step A1.

We note that GEN is the LexBFS; the ordinary LexBFS in [25] decides that the vertices in L are tie, which can be broken arbitrary, and GEN checks and sorts them in step A2 and its substeps.

In GEN, for each vertex u in $V - \{v_n\}$, there exists a unique vertex v such that v is picked up in step either A1, and u is numbered as a neighbor of v in

step A2. We say v is the *parent* of u, and denote by $p(u) = v$. Here we observe that $p(v_1) < p(v_2)$ implies $v_1 < v_2$ by step A1.

For a parent v, we have a set L of unnumbered neighbors. The main loop of A2 divides the vertices in L into some *groups*. The first group is constructed as follows; GEN first picks up a vertex u in step A2.1, constructs W in step A2.3, numbers all vertices in W in step A2.5, updates U by W in step A2.6, updates W in step A2.3 for new U, and the process is repeated until $W = \emptyset$. Then, if L contains some vertices not numbered, GEN picks up the next u in step A2.1 again, constructs the second group, and so on. We note that each group has unique vertex u picked up in step A2.1.

Correctness of GEN

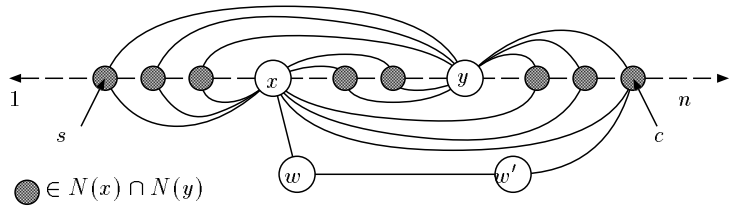

Fig. 1. The vertices not linearly ordered

Lemma 2. *For a given chordal bipartite graph,* GEN *generates a WEO.*

Proof. We assume that the ordering generated by GEN is not a WEO, and derive contradictions. By assumption, there are vertices s, x, y, and w such that $s < x < y$, $s < w$, $s \in N(x) \cap N(y)$, $\{x, w\} \in E$, and $\{y, w\} \notin E$ (Figure 1). We assume that x is greater than any other x' that has the same properties.

Since $\{s, y\} \in E$, we have $s' = p(s) \geq y$. On the other hand, $w > s$ and $w \notin N(y)$. Thus $p(w) > s' \geq y$. Let $w' = p(w)$. Then, since $w' > y$, $p(w') > w' > y$. We here show that $p(w') \in N(x) \cap N(y)$. Suppose that $p(w') \notin N(x) \cap N(y)$. Then, since $p(w')$ is in G_y, there is a (shortest) path P between y and $p(w')$ on G_y. By Theorem 2, s is semi-simplicial. Thus there is a vertex c on P in $N(x) \cap N(y)$. Since $c \neq p(w')$, the path $(c, x, w, w', p(w'))$ and the part of P joining $p(w')$ and c induce a chordless cycle of length greater than 4. Thus we have $p(w') = c \in N(x) \cap N(y)$. That is, we have $w' = p(w) > s' \geq y$, and $p(w') = c \in N(x) \cap N(y)$ (see Figure 1 again).

Since $c \in N(x) \cap N(y)$, we have $c \leq p(x)$ and $c \leq p(y)$. (We remind that if x is unnumbered and c is chosen in step A1, then x will be numbered in step A2 as a neighbor of c, consequently, c will be the parent of x.) If $p(y) > c = p(w')$, we have $y > w'$, which is a contradiction. Thus $c = p(y)$. On the other hand, $c < p(x)$ implies that $p(x) > c = p(y)$, which contradicts that $x < y$. Hence we have $c = p(x) = p(y)$. Since $y \leq s' < w'$, we also have $c = p(s')$. That is, we have $c = p(s') = p(x) = p(y) = p(w')$.

We now suppose that c is chosen in step A1, and x, y, w', and s' are in L, and GEN performs step A2. We first assume that y has another numbered neighbor c'.

Since c is the largest numbered vertex that has unnumbered neighbors, $c' < c$. Since $y < w'$, w' also has numbered neighbor c'' with $c' \leq c'' < c$. We have a path P joining c' and c'' on $G_{c'}$. Thus, $(c', y, s, x, w, w', c'')$ and P induce a cycle of length at least 6. Here, $\{y, w\} \notin E$ and $\{s, w'\} \notin E$ since $s' = p(s) < w'$. Thus $\{c', x\}, \{c'', x\} \in E$. If $c' \neq c''$, x is lex-larger than y and we have $y < x$, which is a contradiction. Thus we have $c' = c''$. Thus, x, y, and w' are lexicographically tie. The ordering will satisfy $x < y < w'$. Thus w' will be chosen before x and y, and it will be numbered in step A2.1 or A2.5. When w' is numbered, it will be put into U, and x will be put into W, and numbered (or x may be numbered in the same W of w'). Thus, since $x < y < w'$, they are numbered in the same group. Let u be the first vertex in the group. The vertices in the group will be numbered step by step. During ith step, we let the set U_i defines the set W_i; that is, $U_0 = \{u\}$, W_0 is the set of unnumbered vertices that share unnumbered neighbor with u, $U_1 = W_0$, W_1 is the set of unnumbered vertices that share unnumbered neighbor with u' for some $u' \in U_1$, $U_2 = W_1$, and so on. We assume that w' in U_i for some $i \geq 0$. Then x is either in W_i or U_i. We have two cases.

(1) x is in U_i. Since $x < y < w'$, we have $y \in U_i$. We consider the set W_{i-1}, which is the set of unnumbered vertices, including x, y, w'. Since y is in W_{i-1}, we have at least one unnumbered vertex y' in $N^-(y) \cap N^(U_{i-1})$. y' has a neighbor in U_{i-1}, and so does w'. Thus, there is a path P between y' and w' not through any vertices in $\{x, y, s, w\}$. Hence (y', y, s, x, w, w') and P induce a cycle of length at least 6. Here, $\{w, y\} \notin E$, and $\{s, w'\} \notin E$. Thus $\{x, y'\}$ is in E. That is $N^-(y) \cap N^(U_{i-1}) \subseteq N^-(x) \cap N^(U_{i-1})$, and $N^*(y) \subseteq N^*(x)$ even if any other vertices in W are numbered. Thus, we have $N^*(y) = N^*(x)$ since $x < y$. Thus, x and y are tie in step A2.5, and when w' is numbered, the tie is broken by $|N^-(x)| > |N^-(y)|$ since $w \in N^-(x) - N^-(y)$. We finally have $x > y$, which is a contradiction.

(2) x is in W_i. Using the same argument in (1), any unnumbered neighbor y' of y is also incident to x. Thus, by the definiton of W in step A2.3, y is also in W_i (not in U_i). Thus, using the same argument in (1), $N^*(y) \subseteq N^*(x)$. Hence, when w' is numbered, the ties are broken by the vertex $w \in N^-(x) - N^-(y)$, which contradicts that $x < y$.

Thus GEN generates a WEO. □

Simplified GEN

For any given bipartite graph $G = (X, Y, E)$, we can obtain another bipartite graph $G' = (X \cup \{x\}, Y \cup \{y\}, E')$ with $E' = E \cup \{\{x, y'\} | y' \in Y\} \cup \{\{x', y\} | x' \in X\} \cup \{x, y\}$. The vertices x and y are called *universal vertices*. Then G is chordal bipartite iff G' is chordal bipartite. For the graph G', GEN can generate a WEO such that the largest two vertices are the universal vertices. Then we can obtain a WEO of G by removing the last x and y. The WEO for G satisfies that all vertices in X are less than each vertex in Y. That is, any given chordal bipartite graph $G = (X, Y, E)$, there is a WEO such that $\max\{x \in X\} < \min\{y \in Y\}$. (In fact, the assumption has no effect on the proof of Theorem 1.) This fact simplifies the algorithm GEN. The modified algorithm GEN' runs on G and generates the

WEO of G, which is obtained from the WEO of G' generated by GEN running on G' by removing the last universal vertices:

A'1. number all vertices in Y as follows:
  A'1.1. pick up any vertex y_n;
  A'1.2. number y; set $U := \{y\}$;
  A'1.3. set $W := \{w \mid w \in Y$ is not numbered, and $N(w) \cap N(U) \neq \emptyset\}$;
  A'1.4. while W contains unnumbered vertex, pick up the vertex w such that $N^*(w)$ is lex-larger than any other $N^*(w')$ with unnumbered w' in W, ties are broken by $|N(w)|$;
  A'1.5. set $U := W$ and go to step A'1.3;
A'2. number all vertices in X such that $x > x'$ if and only if $N(x)$ is lex-larger than $N(x')$;

Linear time implementation of GEN' is omitted for lack of space, and can be found in [29].

5 Applications on chordal bipartite graph

Recognition of chordal bipartite graph
We assume that given graph is bipartite, which can be determined in $O(n+m)$ time by the depth first search. Let $G = (V_1, V_2, E)$ be a given bipartite graph, and suppose that $V = V_1 \cup V_2$ is ordered by GEN.

Lemma 3. *Given bipartite graph is chordal bipartite iff the bipartite adjacency matrix A ordered by* GEN *is Γ-free.*

Proof. By Lemma 1(1), "only if" part is trivial. Let v_i, v_j, u_k, and u_l be the vertices with $v_i < v_j$, $u_k < u_l$, and $a_{ik} = a_{jk} = a_{il} = 1$. We show that $a_{jl} = 1$. Without loss of generality, we assume that $v_i < u_k$. We have two cases; $v_i < u_k < v_j < u_l$ and $v_i < v_j < u_k < u_l$. In both cases, since $\{v_i, u_k\}, \{v_i, u_l\} \in E$ and $u_k < u_l$, we have $N_{>v_i}(u_k) \subseteq N_{>v_i}(u_l)$ by the definition of WEO. Thus $v_j \in N_{>v_i}(u_k) \subseteq N_{>v_i}(u_l)$, or $a_{jl} = 1$. □

Corollary 1. *Given graph $G = (V, E)$, the ordering $v_1, \cdots, v_n$ is WEO iff $i < j$, $k < l$, $v_k, v_l \in N(v_i)$, and $v_k \in N(v_j)$ imply $v_l \in N(v_j)$ for each i, j, k and l.*

Theorem 3. *The recognition problem for chordal bipartite graph can be solved in $O(n+m)$ time.*

Proof. (Sketch) Using Lubiw's algorithm shown in [19, Section 5], we can check whether the ordered matrix by GEN is Γ-free in $O(n+m)$ time. □

Corollary 2. *Any given $n_1 \times n_2$ $(0,1)$-matrix A with m 1's, we can compute the doubly lexical ordering of A in $O(n_1 + n_2 + m)$ time if A is Γ-free.*

Other characterizations

Let $G = (V_1, V_2, E)$ be a bipartite graph. Then $\{x, y\} \in E$ is called a *bisimplicial edge* if $G[N(x) \cup N(y)]$ is a complete bipartite graph. Let $e_1, e_2, \cdots, e_m$ be an ordering of the edges of G. For $i = 0, 1, \cdots, m$ define the subgraph $G^i = (V_1, V_2, E_i)$ by $E_0 = E$, and $E_j = E_{j-1} - \{e_j\}$ for $j \geq 1$. The ordering $e_1, e_2, \cdots, e_m$ is a *perfect edge without vertex elimination ordering* (pewveo) for G if each edge e_i is bisimplicial in G^{i-1}. A bipartite graph is chordal bipartite iff it has a pewveo (see [2, Remark 5.9.1]).

Given bipartite graph $G = (S, T, E)$ with $S = \{s_1, s_2, \cdots, s_p\}$ and $T = \{t_1, t_2, \cdots, t_q\}$, a pair of two orderings $s_1 < s_2 < \cdots < s_p$ and $t_1 < t_2 < \cdots < t_q$ is a *strong T-elimination ordering* if, for each $1 \leq i \leq q$ and $1 \leq j < k \leq p$ where $s_j, s_k \in N(t_i)$, we have that $N(s_j) \subseteq N(s_k)$ on the graph $G[S \cup \{t_h \in T | h \geq i\}]$. Chang shows that a bipartite graph is chordal bipartite iff it has a strong T-elimination ordering [4].

Theorem 4. *Given chordal bipartite graph, we can compute a pewveo and a strong T-elimination ordering in $O(n + m)$ time.*

Proof. (Sketch) Let $v_1, v_2, \cdots, v_n$ be the WEO of the chordal bipartite graph G. Then the following algorithm produces the pewveo:

For each $v = v_1, v_2, \cdots, v_n$, let $N(v) = \{u_1, u_2, \cdots, u_{d(v_i)}\}$ with $u_1 < u_2 < \cdots < u_{d(v)}$; for each $u = u_1, u_2, \cdots, u_{d(v)}$, output $\{v, u\}$, and delete the edge from G.

The correctness of the algorithm follows from the results by Kloks and Kratsch [17] with Lemma 3. The algorithm runs in $O(n + m)$ time. The WEO directly gives a strong T-elimination ordering. □

Other problems

A *matching* M is a subset of E such that no two edges in M are incident to a common vertex. A *maximum matching* is a matching of maximum cardinality.

Theorem 5. *The following problems on a chordal bipartite graph can be solved in linear time: (1) Computing the list of all maximal complete bipartite subgraphs, and (2) finding a maximum matching.*

Proof. (Sketch) Using Theorem 4, the results by Kloks and Kratsch in [17], and the results by Chang in [4], we have the theorem. □

Any given matching M, if an edge $\{u, v\}$ belongs to M, we say that M *covers* the vertices u and v. Given a real weight for each vertex of G, the *vertex-weight of a matching M* is defined to be the sum of the weights of the vertices covered by M. A *maximum vertex-weighted matching* is one which has the maximum vertex-weight among all possible matchings of G.

Theorem 6. *A maximum vertex-weighted matching on a chordal bipartite graph can be found in linear time.*

Proof. Campêlo and Klein presented a linear time algorithm for a maximum vertex-weighted matching in given strongly chordal graph with its strong elimination ordering [3]. Their algorithm uses the property "for each i, j, k and l, if $i < j, k < l$, $v_k, v_l \in N[v_i]$, and $v_k \in N[v_j]$, then $v_l \in N[v_j]$". However, it is sufficient for their algorithm to satisfy the property "for each i, j, k and l, if $i < j, k < l$, $v_k, v_l \in N(v_i)$, and $v_k \in N(v_j)$, then $v_l \in N(v_j)$". Thus, by Corollary 1, the algorithm also finds a maximum vertex-weighted matching of a chordal bipartite graph with the WEO in linear time. □

It is worth mentioning that Chang's algorithm for a maximum matching is very simple, while the algorithm for a maximum vertex-weighted matching due to Campêlo and Klein uses an integer linear programming.

6 Applications on strongly chordal graph

We now turn to strongly chordal graphs, which are introduced by Farber as follows[8]: a *strong elimination ordering* of a graph $G = (V, E)$ is an ordering $v_1, v_2, \cdots, v_n$ of V such that $i < j$, $k < l$, $v_k, v_l \in N[v_i]$, and $v_k \in N[v_j]$ imply $v_l \in N[v_j]$ for each i, j, k and l. A graph is *strongly chordal* if it admits a strong elimination ordering.

Recognition of strongly chordal graph and generating strong elimination ordering

Theorem 7. *The recognition problem for strongly chordal graph can be solved in $O(n+m)$ time. Moreover, we can compute the strong elimination ordering in linear time.*

Proof. (Sketch) Given graph $G = (V, E)$ with $V = \{v_1, v_2, \cdots, v_n\}$ and $|E| = m$, we define $G' = (V \cup U, E_1 \cup E_2)$ as follows: $U = \{u_1, u_2, \cdots, u_n\}$, E_1 contains $\{v_i, u_j\}$ and $\{v_j, u_i\}$ for each $\{v_i, v_j\} \in E$, and E_2 contains $\{u_i, v_i\}$ for each $1 \leq i \leq n$. Since $|V \cup U| = 2n$ and $|E_1 \cup E_2| = 2m + n$, the construction of G' can be done in linear time. We show that G is strongly chordal iff G' is chordal bipartite. By Lemma 1, if G is strongly chordal, G' is chordal bipartite. Thus we assume that G' is chordal bipartite, and show that G is strongly chordal. Since G' is chordal bipartite, G' has a WEO over $V \cup U$. We show the WEO over V is the strong elimination ordering of G. Since the ordering is the WEO of G', for each $i < j$ and $k < l$, $\{v_i, u_k\} \in E_1$, $\{v_i, u_l\} \in E_1$ and $\{v_j, u_k\} \in E_1$ imply $\{u_l, v_j\} \in E_1$. Then, it is sufficient to show that, on G, for each $i < j$ and $k < l$, $v_k, v_l \in N[v_i]$ and $v_k \in N[v_j]$ imply $v_l \in N[v_j]$. Since $\{v_i, v_j\} \in E$ on G iff $\{v_i, u_j\}, \{v_j, u_i\} \in E_1$ on G', we have nothing to do when i, j, k and l are all different. Thus we have three cases to consider: $k = i$, $l = i$, and $k = j$. For lack of space, we only consider the first case here. Assume $k = i$. On G', we have $\{v_i, u_l\} \in E_1$ and $\{v_j, u_i\} \in E_1$. Since $i < j$, $i < l$, $\{v_i, u_j\} \in E_1$, and the ordering is WEO, $N(u_j) \subseteq N(u_l)$ or $N(u_l) \subseteq N(u_j)$. If $N(u_j) \subseteq N(u_l)$, $v_j \in N(u_j) \subseteq N(u_l)$. Thus $\{v_j, u_l\}, \{v_l, u_j\} \in E_1$, thus $\{v_j, v_l\} \in E$. Hence

$v_l \in N[v_j]$ on G. The case $N(u_l) \subseteq N(u_j)$ is symmetrical. The second and third cases are similar, and omitted here.

Therefore to determine whether G is strongly chordal, we check whether G' is chordal bipartite, which can be done in linear time by Theorem 3. The strong elimination ordering of G can be obtained from the WEO of G'. □

Other problems on strongly chordal graph

For a given graph $G = (V,E)$, $U \subseteq V$ is a *dominating set* if $N[U] = V$. The *domatic number* of G, denoted by $dn(G)$, is the maximum number of pairwise disjoint dominating sets in G. The *domatic partition* is a partition V into $dn(G)$ disjoint dominating sets.

If a strongly chordal graph is given with its strong elimination ordering, several problems can be solved in linear time. Thus we immediately have the following by Theorem 7.

Theorem 8. *Given strongly chordal graph, we can find a maximum (vertex-weighted) matching, an independent dominating set, a minimum weighted dominating set, a minimum dominating clique, and a domatic partition in linear time.*

Proof. A maximum matching is due to Dahlhaus and Karpinski [7], a weighted maximum matching is due to Campêlo and Klein [3], an independent dominating set and a minimum weighted dominating set are due to Farber [9], a minimum dominating clique is due to Kratsch [18], and a domatic partition is due to Peng and Chang [23]. □

Acknowledgments

The author is grateful to Jeremy Spinrad and Feodor Dragan for many useful discussions. They pointed out the errors of GEN in an early version. The idea of universal vertices, which made the algorithm and its proof much simpler, is due to Jeremy Spinrad. The author is also grateful to Haiko Müller for helpful comments. He introduces a notion of two-pair, which simplify the definitions of semi-clique and semi-simplicial.

References

1. M. Bakonyi and A. Bono. Several Results on Chordal Bipartite Graphs. *Czechoslovak Math. J.*, 47:577–583, 1997.
2. A. Brandstädt, V.B. Le, and J.P. Spinrad. *Graph Classes: A Survey.* SIAM, 1999.
3. M.B. Campêlo and S. Klein. Maximum Vertex-Weighted Matching in Strongly Chordal Graphs. *Disc. Appl. Math.*, 84:71–77, 1998.
4. M.-S. Chang. Algorithms for Maximum Matching and Minimum Fill-in on Chordal Bipartite Graphs. In *7th ISAAC*, pp. 146–155. LNCS Vol. 1178, Springer-Verlag, 1996.
5. D.G. Corneil, S. Olariu, and L. Stewart. The Ultimate Interval Graph Recognition Algorithm? In *9th SODA*, pp. 175–180. ACM, 1998.

6. E. Dahlhaus. Generalized Strongly Chordal Graphs. Technical Report TR93-458, University of Sydney, 1993.
7. E. Dahlhaus and M. Karpinski. Matching and Multidimensional Matching in Chordal and Strongly Chordal Graphs. *Disc. Appl. Math.*, 84:79–91, 1998.
8. M. Farber. Characterization of Strongly Chordal Graphs. *Disc. Math.*, 43:173–189, 1983.
9. M. Farber. Domination, Independent Domination, and Duality in Strongly Chordal Graphs. *Disc. Appl. Math.*, 7:115–130, 1984.
10. M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, 1980.
11. M.C. Golumbic and C.F. Goss. Perfect Elimination and Chordal Bipartite Graphs. *J. of Graph Theory*, 2:155–163, 1978.
12. R. Hayward, C. Hoàng, and F. Maffray. Optimizing Weakly Triangulated Graphs. *Graphs and Combinatorics*, 5(4):339–349, 1989. [Erratum, *ibid.*, 6(1):33–35, 1990.].
13. A.J. Hoffman, A.W.J. Kolen, and M. Sakarovitch. Totally-Balanced and Greedy Matrices. *SIAM J. Alg. Disc. Meth.*, 6(4):721–730, 1985.
14. C.R. Johnson and J. Miller. Rank Decomposition Under Combinatorial Constraints. *Linear Algebra and Its Applications*, 251:97–104, 1997.
15. C.R. Johnson and G.T. Whitney. Minimum Rank Completions. *Linear and Multilinear Algebra*, 28:271–273, 1991.
16. D. Kőnig. *Theorie der endlichen und unendlichen Graphen (in German).* Akademische Verlagsgesellschaft, 1936.
17. T. Kloks and D. Kratsch. Computing a Perfect Edge Without Vertex Elimination Ordering of a Chordal Bipartite Graph. *IPL*, 55:11–16, 1995.
18. D. Kratsch. Finding Dominating Cliques Efficiently, in Strongly Chordal Graphs and Undirected Path Graphs. *Disc. Math.*, 86:225–238, 1990.
19. A. Lubiw. Doubly Lexical Orderings of Matrices. *SIAM J. on Computing*, 16(5):854–879, 1987.
20. T.A. McKee and F.R. McMorris. *Topics in Intersection Graph Theory.* SIAM, 1999.
21. R. Paige and R.E. Tarjan. Three Partition Refinement Algorithms. *SIAM J. on Computing*, 16(6):973–989, 1987.
22. B.S. Panda. New Linear Time Algorithms for Generating Perfect Elimination Ordering of Chordal Graphs. *IPL*, 58:111–115, 1996.
23. S.-L. Peng and M.-S. Chang. A Simple Linear Time Algorithm for the Domatic Partition Problem on Strongly Chordal Graphs. *IPL*, 43:297–300, 1992.
24. N.K.R. Prasad and P.S. Kumar. On Generating Strong Elimination Orderings of Strongly Chordal Graphs. In *FST&TCS*, pp. 221–232. LNCS Vol. 1530, Springer-Verlag, 1998.
25. D.J. Rose, R.E. Tarjan, and G.S. Lueker. Algorithmic Aspects of Vertex Elimination on Graphs. *SIAM J. on Computing*, 5(2):266–283, 1976.
26. K. Simon. A Note on Lexicographic Breadth First Search for Chordal Graphs. *IPL*, 54:249–251, 1995.
27. J.P. Spinrad. Doubly Lexical Ordering of Dense 0-1 matrices. *IPL*, 45:229–235, 1993.
28. 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 J. on Computing*, 13(3):566–579, 1984.
29. R. Uehara. Linear Time Algorithms on Chordal Bipartite and Strongly Chordal Graphs. `http://www.komazawa-u.ac.jp/~uehara/ps/chordal-linear-full.ps.gz`, 2002.