Vol.2015-AL-154 No.5
2015/9/28

二部グラフ中に含まれる弦二部誘導グラフの列挙

和佐 州洋[1]　有村 博紀[1]　宇野 毅明[2]　平田 耕一[3]

概要：本稿では，弦二部誘導グラフの列挙問題を考察する．二部グラフ B が弦二部グラフであるとは，B に含まれる長さが 6 以上の任意のサイクル C に対して，C 上で隣り合わない 2 頂点の間に，少なくともひとつの辺があるときをいう．主結果として，与えられた二部グラフに含まれるすべての弦二部誘導グラフを，もれなく重複なく出力する多項式遅延列挙アルゴリズムを与える．

Enumeration Algorithm for Chordal Induced Bipartite Graphs of a Bipartite Graph

Kunihiro Wasa[1]　Hiroki Arimura[1]　Takeaki Uno[2]　Kouichi Hirata[3]

Abstract: In this paper, we study an enumeration problem for chordal induced bipartite graphs of a bipartite graph. A bipartite graph B is bipartite chordal if for any cycle C in B whose length is more than four, there exists at least one edge between two vertices that are not adjacent on C. As the main result, we develop a polynomial delay algorithm that enumerates all chordal induced bipartite graphs in a given bipartite graph.

1. はじめに

部分グラフ列挙問題とは，入力グラフ G とグラフクラス $\mathcal{C}$ が与えられた時に，$\mathcal{C}$ に含まれる，G 中のすべての部分グラフ S を漏れなく重複なく出力する問題である．列挙問題については，これまで非常に多く研究が行われてきた [2, 5–8, 15].

列挙問題における解は，一般に，その数が非常に多く，単に解をすべて出力するだけでも膨大な時間がかかる．一方で，解を出力する間隔が短いこともある．したがって，アルゴリズムの性能を図る上で，アルゴリズムの開始から終了までの時間で評価するのは適切ではない．そこで，列挙アルゴリズムの性能は，解の個数 N であり，かつ，列挙アルゴリズム $\mathcal{A}$ の時間計算量が $O(Nt)$ であるとき，$\mathcal{A}$ は，解 1 つあたりならし $O(t)$ 時間で出力する，というように評価される．また，任意の連続する 2 つの出力 S_i と S_{i+1} に対して，$\mathcal{A}$ が S_i から S_{i+1} まで出力する時間が $O(t)$ で抑えられるとき，$\mathcal{A}$ は $O(t)$ 遅延であるという．一般に，$O(t)$ 遅延ならば，ならし $O(t)$ 時間であるといえるが，その逆は成り立たない．

グラフ G が (a, b)-chordal であるとは，G に含まれる長さ a 以上の任意のサイクルに，少なくとも b 本の弦があるときをいう．ここで，弦とはサイクル上で隣接しない頂点間にある辺のことをいう．特に，$(4, 1)$-chordal を**弦グラフ**と呼び，$(6, 1)$-chordal な二部グラフを**弦二部グラフ** [3] と呼ぶ．これまで知られている弦二部グラフを認識するアルゴリズムの時間計算量の上界は，$O(\min\{n^2, m \log n\})$ 時間である [11, 13]．弦二部グラフを線形時間で認識するアルゴリズムを提案されているが [14]，Sawada と Spinrad により，誤りが指摘されている [12].

[1] 北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University, {wasa, arim}@ist.hokudai.ac.jp
[2] 国立情報学研究所
National Institute of Informatics, uno@nii.ac.jp
[3] 九州工業大学大学院 情報工学研究院
Department of Artificial Intelligence, Kyushu Institute of Technology, hirata@dumbo.ai.kyutech.ac.jp

1.1 主結果

本稿では，二部グラフに含まれるすべての弦二部誘導グラフの列挙する問題について考察する．我々は，特に，グラフの縮退数 k について着目し，アルゴリズムを構築する．グラフ G が k-縮退であるとは，G 中の任意の誘導グラフが，次数が高々 k の頂点を持つときをいう [9]．このような最小の k を G の縮退数と呼ぶ．G の縮退数を求める線形時間アルゴリズムが，Matula と Beck [10] によって示されている．また，縮退数が定数であるようなグラフとして，木や，格子グラフ，平面グラフなどが知られている．

主結果として，弦二部誘導グラフを，解 1 つあたりならし $O(nk^3)$ 時間で出力する列挙アルゴリズムを提案する．ただし，n と k を，それぞれ，入力グラフ中の頂点数と縮退数とする．分割法をベースにした提案アルゴリズムは，空集合から始まり，弦二部誘導グラフに頂点をひとつずつ加えながらサイズを大きくすることで，列挙するアルゴリズムである．我々は，Uehara [14] が示した弦二部グラフ中の各頂点の特徴付けに関する定理をベースにし，加えられる頂点集合を高速に判定するデータ構造を管理することで，アルゴリズムの高速化を図っている．

この結果は，平面グラフなどの定数縮退数を持つ入力グラフに対して，解 1 つあたりならし線形時間で列挙できることを意味する．これまで知られている認識にかかる時間の上界が $O(\min\{n^2, m\log n\})$ 時間であることを考えると，この結果は，$n^2 < m\log n$ のとき，この上界に一致するアルゴリズムである．

1.2 関連研究

これまで，弦グラフの列挙について広く研究行われてきた．Kiyomi ら [7] は，与えられたグラフを部分グラフとして持つ弦グラフの $O(n^3)$ 時間遅延列挙アルゴリズムを与えた．また，Kiyomi と Uno [8] は，与えられたグラフに含まれる弦誘導グラフを，定数遅延で列挙するアルゴリズムを示した．Kijima ら [6] は，与えられた 2 つのグラフ $\overline{G}$ と $\underline{G}$ に対して，$\overline{G}$ または $\underline{G}$ が弦グラフであるときに，$\overline{G}$ に含まれ，かつ，$\underline{G}$ を含むような弦グラフを，多項式時間遅延で列挙するアルゴリズムを与えた．

Ausiello ら [1] によって，(a,b)-chordal と超グラフの非巡回性には深い関係があることが示されている．実際に，超グラフ $\mathcal{H}$ の接続グラフが弦二部グラフならば，その時に限り，$\mathcal{H}$ は β-非巡回である．同様に，弦グラフならばその時に限り Berge-非巡回，また，$(6,1)$-chordal ならばその時に限り γ-非巡回である．α-非巡回に関しても，二部グラフの弦に着目した特徴付けができる．

超グラフの非巡回な部分超グラフ列挙として，Daigo と Hirata [2] は，入力超グラフに含まれる極大な α-非巡回な部分超グラフを，多項式遅延かつ線形領域で列挙するアルゴリズムを与えた．また，Wasa ら [15] は，Berge-非巡回な部分超グラフを多項式時間遅延かつ線形領域で列挙するアルゴリズムを与えた．

しかし，筆者らの知る限り，弦二部誘導グラフ，または，それと等価な β-非巡回な部分超グラフを，多項式時間遅延で列挙する非自明なアルゴリズムはまだ発見されていない．

1.3 本稿の構成

本稿の構成は次のとおりである．2 節と 3 節では，本稿で用いる用語と本稿で扱う問題について定義する．さらに，グラフの縮退性に関する性質や，弦二部グラフと特徴付けに関して議論する．4 節では，我々の提案アルゴリズムと，その正当性を与え，5 節では，計算量について議論する．6 節では，結論を述べる．

2. 準備

無向グラフ (undirected graph, 以下，単にグラフという) $G = (V(G), E(G))$ を，**頂点集合** $V(G)$ と**辺集合** $E(G) \subseteq V(G)^2$ の組で表す．任意の 2 頂点 $u, v \in V(G)$ に対して，u と v が G 中で**隣接**しているとは，$(u,v) \in E(G)$ であるときをいう．さらに，$N_G(u) = \{v \in V(G) \mid (u,v) \in E(G)\}$ を，u の G における隣接頂点集合という．頂点 $u \in V$ の**次数**を $d_G(u) = |N_G(u)|$ とする．$G = (U(G), V(G), E(G))$ が**二部グラフ**であるとは，次の (I)-(III) を満たす 2 つの頂点集合 $U(G)$ と $V(G)$，と辺集合 $E(G)$ からなるグラフである：(I) $U(G) \cap V(G) = \emptyset$ かつ，(II) 任意の $u, v \in U(G)$ に対して，$(u,v) \notin E(G)$，(III) 任意の $u, v \in V(G)$ に対して，$(u,v) \notin E(G)$ を満たす．以下では，グラフを $G = (V, E)$ に固定し，さらに，G は単純，つまり，多重辺と自己閉路を持たないとする．また，文脈から明らかならば，$V(G)$ と，$E(G)$，$N_G(u)$ を，それぞれ，単に，V と，E，$N(u)$ と書く．また，$n = |V|$，$m = |E|$ とする．

G 中の任意の 2 頂点を $u, v \in V$ とし，j 個の頂点の列を $\pi(u,v) = (v_1 = u, \ldots, v_j = v)$ とする．このとき，各 $i = 1, \ldots, j-1$ に対して，相異なる $j-1$ 個の辺 (v_i, v_{i+1}) が E に含まれるとき，$\pi(u,v)$ を u から v への**パス**という．路の**長さ**を $|\pi(u,v)| = j-1$ とする．ここで，路の中で，長さが 3 以上，かつ，$u = v$ であるとき $\pi(u,v)$ を**閉路**といい，G が閉路を持たないとき，G を**非巡回**であるという．また，G 中の任意の相異なる 2 頂点間にパスが存在するとき，G は**連結**であるという．

V の任意の頂点集合 S に対して，S によって誘導されるグラフを $G[S] = (S, E[S])$ とし，$G[S]$ を G の**誘導グラフ** (induced graph) という．ここで，$E[S] = \{(u,v) \in E \mid u, v \in S\}$ とする．$G[S]$ がパス，または，閉路をなすとき，$G[S]$ を，それぞれ，**誘導パス** (induced path)，または，**誘導閉路** (induced cycle) という．連結な極大誘導グラフを G の**成分** (component) と呼ぶ．また，$G - S = G[V \setminus S]$ とする．$G[S]$ は G と S から一意に決ま

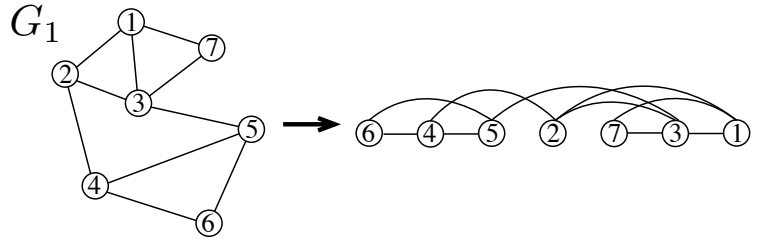

図 1 グラフ $G_1 = (V_1, E_1)$ における縮退順の例. G_1 の縮退数は 2 である．また，右のグラフは，G_1 の頂点を縮退順に左から右へ並べたものである．

るため，文脈から明らかならば，$G[S]$ と S を同一視する．

G が k-**縮退** (k-degenerate) であるとは，G 中の任意の誘導グラフが，次数が高々 k の頂点を持つときをいう [9]．上記を満たす最小の k を，G の**縮退数** (degeneracy) と呼ぶ．k-縮退グラフについて，次の性質が知られている．

補題 1 (Lick and White [9], Proposition 1)**.** グラフ $G = (V, E)$ が k-縮退であるとする．このとき，グラフ中の頂点間に，次を満たす全順序が存在する．

$$\forall v(v \in V \wedge |\{u \mid (u, v) \in E \wedge v < u\}| \leq k) \qquad (1)$$

つまり，自身より大きく，かつ，隣接する頂点が高々 k 個しかないような全順序が存在する．ここで，(1) を満たす全順序のことを，**縮退順**と呼ぶ（図 1）．Matula と Beck は，最小次数の頂点を順次取り除いていくことで，縮退順を $O(|V| + |E|)$ 時間で求めるアルゴリズムを与えている [10]．以下では，頂点間の大小をこの縮退順で与えるものとし，入力グラフの縮退数を k とする．また，v に隣接する v より小さい頂点集合と，大きい頂点集合を，それぞれ，$N_<(v) := \{u \mid (u, v) \in E \wedge u < v\}$ とし，$N_>(v) := \{u \mid (u, v) \in E \wedge u > v\}$ とする．

3. 弦二部グラフ

二部グラフ G が**弦二部グラフ**であるとは，G が長さ 6 以上の誘導閉路を持たないときをいう [3]．また，$\mathcal{CBS}(G)$ を G 中の連結な弦二部誘導グラフすべてからなる族とする．ここで，本稿で扱う問題を次のように定義する：

問題 1. 二部グラフ G に含まれるすべての連結な弦二部誘導グラフ $S \in \mathcal{CBS}(G)$ をもれなく重複なく出力せよ．

3.1 Semi-simplicial による特徴付け

本節では，弦二部グラフを semi-simplicial を用いて特徴づけする．グラフ中の相異なる 2 頂点 $x, y \in V(G)$ について，x と y が *two-pair* であるとは，x から y へのすべての誘導パスが，ちょうど 2 つの辺からなるときをいう [4]．頂点集合 $S \subseteq V(G)$ が，*semi-clique* であるとは，S 中の任意の 2 頂点 x と y が，$G[(V - S) \cup \{x, y\}]$ 中で two-pair であるときをいう．ここで，頂点 $u \in V(G)$ が G 中で *semi-simplicial* であるとは，$N(u)$ が semi-clique であるときをいう．Semi-simplicial による弦二部グラフの特徴付けについては，次の定理が知られている [14]．

定理 1 (Uehara [14], Theorem 2)**.** グラフ G が弦二部グラフならば，その時に限り，G 中のすべての頂点は，semi-simplicial である．

$Semi_G(S)$ を，G の連結な誘導グラフ S に加えた時，$S \cup \{u\}$ が連結，かつ，$S \cup \{u\}$ 中で semi-simplicial になような S に含まれない頂点 $u \in V(G) \setminus S$ の集合とする．$Semi_G(S)$ について，定理 1 から，次の補題が得られる．

補題 2. S を G 中の連結な誘導グラフとし，$u \in V(G) \setminus S$ を頂点とする．このとき，S が連結な弦二部グラフかつ $u \in Semi_G(S)$ ならば，その時に限り，$S \cup \{u\}$ は連結な弦二部グラフである．

4. 分割法による列挙アルゴリズム

本節では，問題 1 を効率よく解くためのアルゴリズム `EnumCBS` を提案する（Algorithm 1）．以下では，混乱がない限り，単に弦二部誘導グラフといったとき，連結なものを指すこととする．提案アルゴリズム `EnumCBS` は，補題 2 に示した必要十分条件をベースにした，分割法による列挙アルゴリズムである．アルゴリズム中において，S を弦二部誘導グラフとし，**候補集合** $CAND$ を $S \cup \{u\}$ が弦二部誘導グラフとなるような頂点 u の集合，**禁止集合** X を S に追加しない頂点の集合とする．提案アルゴリズムは，空集合からはじまり，再帰的に `EnumRec` を呼び出す．`EnumRec` は，$CAND$ が空の時は，S を出力し，手続きを終了する．一方で，空でない時，`EnumRec` は，$CAND$ から縮退順で最小の頂点 u を除き，u を含む弦二部誘導グラフの列挙問題と，u を含まない弦二部誘導グラフの列挙問題の 2 つの部分問題に分割し，再帰的に弦二部誘導グラフを列挙する．

4.1 縮約したグラフにおける semi-simplicial の特徴付け

アルゴリズムを構成する上で重要な点は，どの頂点が $CAND$ に入るか，つまり，頂点 u を S へ加えた後に，どの頂点が $S \cup \{u\}$ へ加えた時に semi-simplicial になるか，を効率よく判定する手法を与えることである．そこで，まず，u に隣接する 2 頂点が two-pair であるかを判定する方法について考察する．次の補題は，Uehara が [14] 中で言

Algorithm 1: Enuemraiton algorithm for chordal bipartite induced graphs.

```
Procedure EnumCBS(G = (U(G), V(G), E(G)))
    input  : G = (U(G), V(G), E(G)): 入力二部グラフ
    output : All chordal induced bipartites of G
    Output ∅;
    X ← ∅ ; TP ← ∅;
    while G ≠ ∅ do
        u ← Pick the minimum vertex of U(G) ∪ V(G);
        X ← X ∪ {u};
        TP ← UpdateTP(TP, u, ∅, G);
        EnumRec(G, {u}, N(u), X, TP);

Subprocedure EnumRec(G, S, CAND, X, TP)
    if CAND = ∅ then output S; return;
    u ← Pick the minimum vertex of CAND;
    EnumRec(G, S, CAND \ {u}, X ∪ {u}, TP);
    TP ← UpdateTP(TP, u, S, G);
    EnumRec(G, S ∪ {u}, ComputeCAND(TP, S ∪ {u}, G, X), X ∪ {u}, TP);
```

及している，two-pair に関する性質である．

補題 3. x と y を二部グラフ G 中の任意の 2 頂点とする．ここで，次の (I) と (II) は等価である．

(I) x と y が two-pair である．

(II) 任意の $x' \in N(x) \setminus N(y)$ と $y' \in N(x) \setminus N(y)$ に対して，x' から y' へのすべての誘導パスは，$N(x) \cap N(y)$ 中の頂点を少なくともひとつ含む．

証明. 以下では，G が二部グラフであることから，$x, y \in U(G)$ としても一般性を失わない．(I) $\rightarrow$ (II): $x' \in N(x) \setminus N(y)$ から $y' \in N(y) \setminus N(x)$ へ，$N(x) \cap N(y)$ を通らない誘導パス $P \subseteq V(G)$ が存在すると仮定する．ここで，$P \cap (N(x) \setminus N(y)) = \{x'\}$ と $P \cap (N(y) \setminus N(x)) = \{y'\}$ と仮定しても，一般性を失わない．また，x' と y' は隣接していないので，P の長さは 2 以上である．さらに，P 中で x, y と隣接するのは，x', y' のみであるので，$P \cup \{x, y\}$ は，長さ 4 以上の誘導パスである．したがって，x と y は two-pair ではない．よって，対偶から示された．(II) $\rightarrow$ (I): x と y が two-pair でないとする．このとき，x から y へ，長さ 4 以上の誘導パス P が存在する．つまり，$N(x) \setminus N(y)$ から $N(x) \setminus N(y)$ への誘導パス $P' = P \setminus \{x, y\}$ は，$N(x) \cap N(y)$ 中の頂点を含むと，P は長さ 4 以上の誘導パスではなくなる．したがって，P' は，$N(x) \cap N(y)$ 中の頂点を含まない．よって，補題は成り立つ． □

補題 3 より，頂点 u が semi-simplicial であるかを判定するには，two-pair による semi-simplicial の特徴づけから，u に隣接する任意の 2 頂点 x と y について，$N(x) \setminus N(y)$ から $N(y) \setminus N(x)$ への誘導パスすべてを調べれば良いことがわかる．しかし，誘導パスは指数的に存在する．そこで，

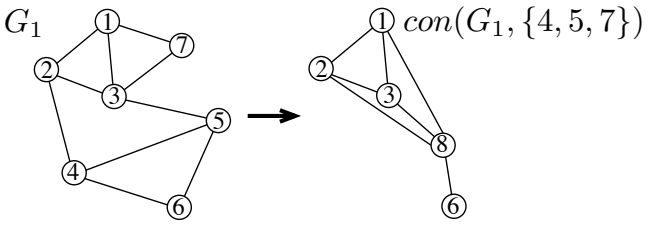

図 2 グラフ G_1 を頂点集合 $S_1 = \{4, 5, 7\}$ について縮約し，新たな頂点 8 と，辺集合 $\{(1,8), (2,8), (3,8), (6,8)\}$ を加えた例．グラフが単純であることに注意されたい．

次に与えるグラフの縮約のアイディアと補題 4 から，グラフ連結性を用いることで，指数的なチェックを省き，頂点が semi-simplicial かを判定する手法を与える．

$G = (V, E)$ 中の頂点集合 $S \subseteq V$ に対して，$con(G, S)$ を，**頂点集合 S に関して縮約したグラフ**といい，$con(G, S) := ((V \setminus S) \cup \{v_S\}, (E \setminus \{(u, v) \in E \mid u \in S \vee v \in S\}) \cup \{(u', v_S) \mid \exists v (u' \in V \setminus S \wedge v \in S \wedge (u', v) \in E)\})$ とする．つまり，頂点集合 S を一つの頂点 v_S にまとめ，S 中の頂点に接続する辺を v_S に接続し直す操作である．

例 1. 図 2 の右側のグラフは，グラフ G_1 を頂点集合 $S_1 = \{4, 5, 7\}$ について縮約したグラフ $con(G_1, S_1)$ である．

補題 4. $G = (V, E)$ を任意のグラフとし，x, y を G 中の任意の頂点，$S \subseteq V$ を V の部分集合とする．ここで，次の 2 つは等価である．

(I) x から y へのすべてのパス P に対して，P は S 中の頂点を少なくともひとつ含む．

(II) x から y へのすべての誘導パス P' に対して，P' は S 中の頂点を少なくともひとつ含む．

証明. $(I) \rightarrow (II)$: グラフ中の任意の誘導パスはパスであ

る．従って明らか．$(II) \to (I)$: x から y へのあるパス P' が S 中の頂点を含まないとする．このとき，P' が弦を含まなければ，P' は誘導パスである．一方で，P' が弦 (u, v) を含むならば，P' 上の u から v へのパスを (u, v) に置き換える．これを，P' から弦がなくなるまで繰り返し行うことで，誘導パス Q を得る．$Q \subseteq P' \subset P$ であることから，Q は S 中の頂点を含まない．したがって，x から y へのあるパス Q は，S 中の頂点を含まないので，対偶から補題は示された． □

以下では，G 中の頂点 x, y に対して，$c(G, x, y) = con(G, N(x) \setminus N(y), N(y) \setminus N(x)) - (N(x) \cap N(y))$ とする．ただし，$con(G, S_1, S_2) := con(con(G, S_1), S_2)$ とする．また，以下では，$v_{x,y} := v_{N(x) \setminus N(y)}$ とし，$v_{y,x} := v_{N(y) \setminus N(x)}$ とする．つまり，$c(G, x, y)$ は，G 中で x に隣接し y に隣接しない頂点の集合と，y に隣接し x に隣接しない頂点の集合を，それぞれ，G 中で縮約し，さらに，その縮約したグラフから $N(x) \cap N(y)$ を除いたグラフである．

補題 5. 二部グラフ G 中で，頂点 u が semi-simplicial であるならば，その時に限り，任意の相異なる 2 頂点 $x, y \in N(u)$ に対して，$c(G, x, y)$ 中で，$v_{x,y}$ と $v_{y,x}$ は，異なる成分に含まれる．

証明. 補題 4 から，$c(G, x, y)$ 中で，$v_{x,y}$ と $v_{y,x}$ が同じ成分に含まれないことと，$v_{x,y}$ から $v_{y,x}$ への任意の誘導パスは，$N(x) \cap N(y)$ 中のある頂点を通ることは同値である．したがって，補題 3 と定義から，補題は成り立つ． □

4.2 アルゴリズムの正当性

補題 5 から，縮約したグラフの連結性を確認することで，頂点が semi-simplicial か判定できることがわかる．提案アルゴリズムは，連結性を効率よく判定するために，次の 3 次元配列 $TP = TP(S, G)$ である TP 表を用いる．以下では，文脈から明らかならば，TP の添字を省略する．TP の各要素 $TP[v][x][y]$ は，任意の $v \in U(G) \cup V(G)$ と，$x, y \in N(v)$ に対して，S を $N(x) \setminus N(y)$ と $N(y) \setminus N(x)$ に関して縮約し，$N(x) \cap N(y)$ を除いたグラフ $S' = c(S, x, y)$ を格納している．さらに，S' 中の各頂点には，$v_{x,y}$，または，$v_{y,x}$ と同じ成分に含まれるかどうかを示すラベル $\ell_{v_{x,y}}$ と $\ell_{v_{y,x}}$ を与える．$v_{x,y}$ と $v_{y,x}$ が S' 中で連結か判定するには，ともに $\ell_{v_{x,y}}$ と $\ell_{v_{y,x}}$ の両ラベルを持たないことを確認すればよい．以下では，文脈から明らかなとき，ラベルと頂点を同一視する．さらに，u に隣接する各頂点に対して，S に含まれるかを示すラベルを与える．以上の議論を用いて，TP 表の逐次的な更新アルゴリズム `UpdateTP` を Algorithm 2 に，$CAND$ の計算アルゴリズム `ComputeCAND` を Algorithm 3 に与える．

補題 6. `UpdateTP`(*TP, u, S, G*) が，S の TP 表 TP から，$S \cup \{u\}$ の TP 表を正しく計算する．

Algorithm 2: Incremental updating the TP table

```
1  Subprocedure UpdateTP(TP, u, S, G) //Add u to S
2      for each v ∈ U(G) ∪ V(G) do
3          for each pair (x, y) ∈ N(v) × N(v) do
4              ℓ(u) ← ∅;
5              if u ∈ N(x) \ N(y) then
6                  ℓ(u) ← ℓ(u) ∪ {v_{N(x)\N(y)}};
7              if u ∈ N(y) \ N(x) then
8                  ℓ(u) ← ℓ(u) ∪ {v_{N(y)\N(x)}};
9              for each w ∈ S ∩ N(u) do
10                 if u < w then
11                     Label u in N(w) ∩ S;
12                 if the label of w is v_{N(x)\N(y)} then
13                     ℓ(u) ← ℓ(u) ∪ {v_{N(x)\N(y)}};
14                 if the label of w is v_{N(y)\N(x)} then
15                     ℓ(u) ← ℓ(u) ∪ {v_{N(y)\N(x)}};
16             if |ℓ(u)| > 0 then
17                 for each component C including u in
                   c(S, x, y) do
18                     Label each vertex in C as ℓ(u);
19     return TP;
```

Algorithm 3: Computing the candidate set

```
1  Subprocedure ComputeCAND(TP, S, G, X)
2      CAND ← ∅;
3      for each v ∈ (U(G) ∪ V(G)) \ X do
4          if for all pair u, w ∈ N(v) ∩ S the labels of v_{x,y} and
             v_{y,x} are different then
5              CAND ← CAND ∪ {v};
6      return CAND;
```

証明. S へ u へ追加するため，ラベルの変化は，TP の各要素に対して，(1) u がどのようなラベルを持つか，u に隣接する S 中の頂点をすべてチェックし，(2) u を含む成分に含まれる頂点すべてに u と同じラベルを与えれば良い．ここで，`UpdateTP` の 5 行目から 15 行目は，(1) に対応する．また，16 行目から 18 行目は，(2) に対応する．したがって，`UpdateTP`(*TP, u, S, G*) は TP から $S \cup \{u\}$ の TP 表を正しく計算する． □

補題 7. `ComputeCAND`(*TP, S, G*) の返す頂点集合 $CAND$ は，$Semi_G(S) \setminus X$ である．

証明. $(U(G) \cup V(G)) \setminus X$ 中の各頂点 v に対して，補題 5 より，4 行目で v が semi-simplicial かどうか判定できる．よって，`ComputeCAND`(*TP, S, G*) は，$Semi_G(S) \setminus X$ を返す． □

次に，EnumCBS の正当性に関する定理を与える．

定理 2. EnumCBS(G) は，入力二部グラフ G に含まれるすべての弦二部誘導グラフをもれなく重複なく出力する．

証明. アルゴリズム全体の健全性を示す．まず，アルゴリズムは，補題 2 と，補題 6，補題 7，さらに，サイズ 1 のグラフがすべて連結な弦二部グラフであることから，出力される誘導グラフはすべて連結な弦二部グラフである．次に，完全性をサイズ s に関する帰納法を用いて示す．まず，サイズ $s \leq 1$ の弦二部誘導グラフは明らかにすべて出力される．次に，サイズ $s \leq i$ の弦二部誘導グラフがすべて出力されると仮定する．ここで，サイズ $i+1$ のある弦二部誘導グラフを S とする．このとき，任意の $u \in S$ と，$S' = S \cup \{u\}$ に対して，u を含むような $Semi_G(S')$ が存在しないと仮定する．ここで，u が $Semi_G(S')$ に入らないのは，u が X に属している時である．一方で，u が X に含まれているのは，12 行目と 14 行目から，ある $S'' \subset S$ の $Semi_G(S'')$ に u が属している時である．さらに，$Semi_G(S'')$ に属しているならば，14 行目で S'' に追加される．従って，仮定は矛盾し，u を含むような $Semi_G(S')$ が存在するので，S は出力される．最後に，出力に重複がないか考察する．EnumCBS の 7 行目の再帰呼び出しで，頂点 u を最小とする弦二部誘導グラフを列挙する部分問題を解いている．また，EnumRec の 12 行目と 14 行目の再帰呼び出しで，頂点 u を含む弦二部誘導グラフを列挙する部分問題と，頂点 u を含まない弦二部誘導グラフを列挙する部分問題を解いている．したがって，重複して弦二部誘導グラフを出力することはない．以上の議論から，定理は示された． □

5. 計算量

本節では，提案アルゴリズムの時間計算量と空間計算量について考察する．前節で示したアルゴリズムにおける TP 表のサイズを単純に見積もると，$O(nd^2)$ であるので，UpdateTP の時間計算量が $O(nd^3)$ で抑えられることから，解 1 つあたり，$O(nd^3)$ 時間で列挙できることがわかる．しかし，この時間計算量は，たとえば，木のような，大きい最大次数を取りうるが，平均次数が低いグラフに対しては，直感的ではない．そこで，我々は縮退順に着目することで，弦二部誘導グラフを，解 1 つあたりならし $O(nk^3)$ 時間で列挙できることを示す．ここで，k を入力グラフの縮退数とする．

EnumCBS から頂点 u を追加することで呼び出した EnumRec を u に関する**根**と呼び，$CAND = \emptyset$ であるような EnumRec の再帰を**葉**と呼ぶ．また，葉ではない EnumRec R で呼び出される 2 つの再帰 R_1, R_2 に対し，R を R_1, R_2 の**親**と呼び，R_1, R_2 を R の**子**と呼ぶ．この親子関係で定義される探索空間を，**探索木**と呼ぶ．

提案アルゴリズムの具体的な実装を次に述べる：グラフは隣接リストで実装し，隣接する頂点を自身との大小で分け，さらに，各ラベルを持つかどうかで分ける．つまり，$2 \times 6 = 12$ 通りに隣接頂点を分類する．さらに，$CAND$ は双方向連結リストで保存する．また，TP 表は大域変数として保存する．ここで，ある再帰呼び出し EnumRec R で TP 表に変更を加えた際に，その操作を保存し，R の子から R に戻ってきた際には，保存しておいた操作を逆に行うことで，TP 表を復元する．このようにすることで，アルゴリズム中で子を再帰呼び出しする際の不要なコピーを防止できる．

5.1 UpdateTP と ComputeCAND の時間計算量

本小節では，EnumRec の副手続き UpdateTP と ComputeCAND の時間計算量を考察する．まず，TP の要素数に関して，次の補題を与える．ここで，グラフにおける辺数 m と，頂点数 n，縮退数 k としたとき，$m = \sum_{v \in U(G) \cup V(G)} |N_>(v)| \leq \sum_{v \in U(G) \cup V(G)} k = nk$ が成り立つ．

補題 8. TP の要素数は，$O(nk^2)$ である．ただし，n と k を，それぞれ，入力二部グラフ G の頂点数と縮退数とする．

証明. G 中の任意の頂点を v とする．v の隣接頂点の組は，$O(d(v)^2)$ 通り存在し，$TP[v]$ は，$O(d(v)^2)$ 個の要素をもつ．これは，v に接続する各辺が，それぞれ，$TP[v]$ 中の $O(d(v))$ 個の要素に対応するといえる．すなわち，表の要素数は，各辺の端点の次数を足し合わせることで得られる．よって，

$$\begin{aligned}
|TP| &= \sum_{(u,v) \in E} O(d(v) + d(u)) \\
&= \sum_{(u,v) \in E} O(d_>(v) + d_<(v) + d_>(u) + d_<(u)) \\
&= 2 \sum_{(u,v) \in E} O(d_<(v) + d_<(u)) \\
&= \sum_{(u,v) \in E} O(k) = O(mk) = O(nk^2).
\end{aligned}$$

□

補題 8 より，TP の要素数は，単純に見積もると $O(nd^2)$ であるが，縮退数 k を用いると $O(nk^2)$ がいえ，定数縮退数を持つようなグラフに対し，頂点数に対して線形であることがわかる．以下では，$|TP| = O(nk^2)$ を TP 表 TP に含まれるの要素数とする．

補題 9. UpdateTP のならし時間計算量は，$O(|TP|k)$ 時間である．

証明. 以下では，葉を固定し，その葉における弦二部誘導グラフを S_* とする．また，その頂点数と変数を，それぞれ，n_{S_*} と m_{S_*} とする．まず，探索木における葉から根ま

でのパス上で，実行された UpdateTP の時間計算量の和について考察する．ある UpdateTP(TP, u, S, G) の呼び出しにおける TP 中の $c(S,x,y)$ を考える．まず，9 行目から 15 行目までのループに必要な時間計算量を考える．ループ内の各行は $O(1)$ 時間で実行可能である．一方で，9 行目でチェックする w について考える．頂点 w が G' 中で頂点 u に隣接し，かつ，S に含まれているのは，$w < u$ の時，u に隣接する頂点で，S に含まれるラベルを持つものに限られる．また，$u < w$ の時は，u に隣接する k 個の w に対して，S に含まれるかどうかをチェックする．したがって，葉から根までのパスにおいてチェックされる w の総数は，$O(m_{S_*} + kn_{S_*}) = O(kn_{S_*} + kn_{S_*}) = O(kn_{S_*})$ である．ラベルの種類は 3 種類であることから，S 中の各頂点は，パス上で高々 3 回しかラベルが付与されない．したがって，一度ラベルを与えた頂点にもう一度おなじラベルを与えるのを回避するように UpdateTP を適用すると，葉から根までの手続きで必要なラベルの更新にかかる時間計算量の和は，$O(|TP|(kn_{S_*} + 3n_{S_*})) = O(|TP|kn_{S_*})$ である．ここで，根から葉までの EnumRec の呼び出し回数は，根から葉までに加えた頂点の数で抑えられるので，$O(n_{S_*})$ 回である．さらに，EnumRec 中で UpdateTP は高々一度しか呼び出されない．したがって，UpdateTP のならし時間計算量は，$O(|TP|k)$ 時間である． □

補題 10. ComputeCAND の時間計算量は，$O(|TP|)$ 時間である．

証明. 4 行目の判定は，v に隣接する任意の 2 頂点の組 x, y に対して，補題 5 より，縮約した 2 頂点が $c(S,x,y)$ 中で，$v_{x,y}$ と $v_{y,x}$ の両方のラベルを持つかチェックすれば良い．したがって，ComputeCAND の時間計算量は，$O(|TP|)$ 時間である． □

5.2 EnumRec の計算量

最後に，本稿の主定理を示す．

定理 3. 二部グラフ G に含まれるすべての連結な弦二部誘導グラフを，$O(n+m)$ 前処理時間かけたのち，1 つあたりならし $O(nk^3)$ 時間で列挙できる．また，空間計算量は，$O(n(n+m)k^2)$ である．ただし，n と，m，k を，それぞれ，G の頂点数と，辺数，縮退数とする．

証明. 補題 8 より，$|TP| = O(nk^2)$ である．また，補題 9 より, UpdateTP のならし時間計算量が $O(|TP|k) = O(nk^3)$ であること，補題 10 より，ComputeCAND の時間計算量が $O(|TP|) = O(nk^2)$ であることから，EnumRec のならし時間計算量は，$O(nk^3)$ 時間である．ここで，葉を除くすべての呼び出し操作は，ちょうど 2 つの再帰呼び出しを行うため，$N = |\mathcal{CBS}(G)|$ とした時，EnumRec が実行される回数はたかだか $2N$ 回である．さらに，各 EnumRec における復元に係るコストは，高々 $O(nk^3)$ 時間である．以上から，解を 1 つあたり，$O(nk^3)$ 時間である．また，縮退順を求めるのに，前処理として $O(n+m)$ 時間が必要である．

次に，空間計算量について考察する．アルゴリズムが必要とする空間は，グラフを保存するのに必要な領域に加え，TP 表の各要素中で用いるラベルを保存する必要がある．ここで，TP 表の各要素については，各頂点と各辺に定数個のラベルを与えれば良いので，必要な空間計算量は $O(n+m)$ である．従って，EnumCBS が必要とする空間計算量は，$O(n(n+m)k^2)$ 領域である．以上の議論から，定理は示された． □

縮退数が定数のグラフとして，平面グラフや，外平面グラフ，格子グラフなどが知られている．二部グラフもグラフの一種であるので，縮退数を定義できる．したがって，これらに対して，次の系が直ちに得られる．

系 1. 定数縮退数をもつ二部グラフ中の弦二部誘導グラフを，1 つあたりならし $O(n)$ 時間で列挙できる．

これまでに知られている，弦二部グラフのもっとも効率のよい認識アルゴリズムは，$T(m,n) = O(\min(n^2, m\log n))$ 時間 [11,13] である．この認識アルゴリムを用いて単純に列挙すると，ひとつあたり $O(nT(m,n))$ 時間かかる．したがって，提案のアルゴリズムは，$k^3 < T(m,n)$ ならば，効率よいアルゴリズムである．

6. おわりに

本稿では，グラフ中の弦二部誘導グラフの列挙問題について考察した．主定理として，弦二部誘導グラフを，解 1 つあたり，ならし $O(nk^3)$ 時間で出力する列挙アルゴリズムを提案した．ただし，n と k を，それぞれ，入力グラフ中の頂点数と縮退数とする．この結果は，平面グラフなどの定数縮退数を持つ入力グラフに対しては，解 1 つあたりならし線形時間で列挙できることを意味する．今後の展望としては，二部グラフでない一般のグラフに対して同じことが言えるか考察する，また，誘導グラフでない，弦二部部分グラフの列挙問題の考察があげられる．

謝辞 本研究は，JSPS 特別研究員奨励費 25・1149 の助成を受けたものです．

参考文献

[1] Ausiello, G., D'Atri, A. and Moscarini, M.: Chordality properties on graphs and minimal conceptual connections in semantic data models, *Journal of Computer and System Sciences*, Vol. 33, No. 2, pp. 179–202 (1986).
[2] Daigo, T. and Hirata, K.: On Generating All Maximal Acyclic Subhypergraphs with Polynomial Delay, *SOFSEM 2009: the 35th Conference on Current Trends in Theory and Practice of Computer Science*, Lecture Notes in Computer Science, Vol. 5404, Springer Berlin Heidelberg, pp. 181–192 (2009).
[3] Golumbic, M. C. and Goss, C. F.: Perfect Elimination

and Chordal Bipartite Graphs, *Journal of Graph Theory*, Vol. 2, No. 2, pp. 155–163 (1978).

[4] Hayward, R., Hoàng, C. and Maffray, F.: Optimizing weakly triangulated graphs, *Graphs and Combinatorics*, Vol. 5, No. 1, pp. 339–349 (1989).

[5] Hirata, K., Kuwabara, M. and Harao, M.: On Finding Acyclic Subhypergraphs, *FCT 2005: the 15th International Symposium on Fundamentals of Computation Theory*, Lecture Notes in Computer Science, Vol. 3623, Springer Berlin Heidelberg, pp. 491–503 (2005).

[6] Kijima, S., Kiyomi, M., Okamoto, Y. and Uno, T.: On Listing, Sampling, and Counting the Chordal Graphs With Edge Constraints, *Theoretical Computer Science*, Vol. 411, No. 26-28, pp. 2591–2601 (2010).

[7] Kiyomi, M., Kijima, S. and Uno, T.: Listing chordal graphs and interval graphs, *WG 2006: the 32nd International Workshop on Graph-Theoretic Concepts in Computer Science*, Lecture Notes in Computer Science, Vol. 4271, Springer Berlin Heidelberg, pp. 68–77 (2006).

[8] Kiyomi, M. and Takeaki, U.: Generating Chordal Graphs Included in Given Graphs, *IEICE Transactions on Information and Systems*, Vol. E89-D, No. 2, pp. 763–770 (2006).

[9] Lick, D. R. and White, A. T.: k-degenerate graphs, *Canadian Journal of Mathematics*, Vol. 22, No. 5, pp. 1082–1096 (1970).

[10] Matula, D. W. and Beck, L. L.: Smallest-last ordering and clustering and graph coloring algorithms, *Journal of the ACM*, Vol. 30, No. 3, pp. 417–427 (1983).

[11] Paige, R. and Tarjan, R. E.: Three Partition Refinement Algorithms, *SIAM Journal on Computing*, Vol. 16, No. 6, pp. 973–989 (1987).

[12] Sawada, J. and Spinrad, J. P.: From a simple elimination ordering to a strong elimination ordering in linear time, *Information Processing Letters*, Vol. 86, No. 6, pp. 299–302 (2003).

[13] Spinrad, J. P.: Doubly lexical ordering of dense 0-1 matrices, *Information Processing Letters*, Vol. 45, No. 5, pp. 229–235 (1993).

[14] Uehara, R.: Linear Time Algorithms on Chordal Bipartite and Strongly Chordal Graphs, *In the proceedings of ICALP 2002: the 29th International Colloquium on Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 2380, pp. 993–1004 (2002).

[15] Wasa, K., Uno, T., Hirata, K. and Arimura, H.: Polynomial delay and space discovery of connected and acyclic sub-hypergraphs in a hypergraph, *DS 2013: the 16th International Conference on Discovery Science*, Lecture Notes in Computer Science, Vol. 8140, Springer Berlin Heidelberg, pp. 308–323 (2013).