Vol.2017-AL-161 No.5
2017/1/17

グラフ中に含まれる縮退部分グラフの列挙

和佐 州洋[1]　宇野 毅明[1]

概要：本稿では，グラフに含まれ，かつ，与えられた縮退数を持つ部分グラフを列挙する問題について考察する．あるグラフ G の縮退数が k であるとは，G の任意の誘導グラフが高々 k の次数を持つ頂点を含むときをいう．主結果として，与えられたグラフ G と，正整数 k に対して，多項式時間の前処理の後，(1) G に含まれる k-縮退誘導グラフを解 1 つあたり $\mathcal{O}(\Delta(k+\log n))$ 時間で，(2) G に含まれる k-縮退部分グラフを解 1 つあたり $\mathcal{O}(\log n)$ 時間で，列挙するアルゴリズムを与えた．ただし，n を G 中の頂点数とし，Δ を G 中の最大次数とする．また，ともに使用する空間計算量は入力グラフのサイズに線形である．これらは逆探索法 [Avis and Fukuda, DAM, 1996] に基づいたアルゴリズムであり，縮退部分グラフを列挙するアルゴリズムとしては，初の多項式遅延のアルゴリズムである．

Enumeration Algorithms for Degenerate Subgraphs in a Graph

Kunihiro Wasa[1]　Takeaki Uno[1]

1. はじめに

部分グラフ列挙問題とは，与えられたグラフに含まれ，かつ，与えられたグラフクラスに属する部分グラフを，漏れなく重複なく列挙する問題である．部分グラフ列挙問題の応用としては，電力網の性能検査 [14] や，SNS からのコミュニティ発見 [8,18]，グラフ間の類似度計算 [10] などが挙げられ，重要な基礎的な問題である．このような状況のもとで，部分グラフ列挙問題について，多くの研究者が取り組み，多くのアルゴリズムが発表されてきた [19].

列挙問題は，その他の問題に比べて，解の数が指数的に大きくなりえるため，一般的な入力の大きさを用いた多項式性によるアルゴリズムの効率性の評価が難しい．しかし，実際の出力サイズは入力サイズの指数長よりも遥かに小さいこともあり，列挙アルゴリズムの計算量を入力サイズのみで評価してしまうと，実際の性能との乖離が生じる．そこで，列挙アルゴリズム $\mathcal{A}$ の計算量は，入力サイズ n に加えて，出力サイズ N に関するオーダーで評価する．$\mathcal{A}$ が**出力多項式**であるとは，$\mathcal{A}$ が入力を受け取り，停止するまでの総時間が $\mathcal{O}(poly(n,N))$ で抑えられるときをいう．ただし，$poly(\cdot)$ を多項式関数とする．$\mathcal{A}$ が**ならし多項式**であるとは，$\mathcal{A}$ が停止するまでの総時間が $\mathcal{O}(poly(n)N)$ で抑えられるときをいう．つまり，$\mathcal{A}$ がならし多項式であるならば，解 1 つあたり出力にかかる平均の時間計算量が $\mathcal{O}(poly(n))$ である．

1970 年代，Read と Tarjan は，グラフに含まれる全域木，パス，および，サイクルを，解 1 つあたり入力サイズの線形時間で出力するアルゴリズムを提案した [14]．その後，全域木に関しては，Shioura らが解 1 つあたり定数時間で出力するアルゴリズムを提案した [15]．また，パスやサイクルに関しては，Birmelé らが出力サイズに関して最適なアルゴリズムを提案した [4]．他にも，クリーク [5,12,16] や，マッチング [2,7,17]，独立点集合 [6,9] などの基礎的な

[1] 国立情報学研究所
National Institute of Informatics, {wasa, uno}@nii.ac.jp

グラフクラスに属する部分グラフを列挙するアルゴリズムが提案されている．一方で，最適なアルゴリズムが知られていない列挙問題や，さらには，出力多項式なアルゴリズムが存在するかわかっていない列挙問題があるのも現状である．

1.1 主結果

そこで本研究では，定数縮退数を持つ部分グラフの列挙問題に着目し，それらを効率よく列挙するアルゴリズムの構築を目標とする．縮退数は，グラフがどのくらい密かを示す指標の一つである．縮退数が定数なグラフとしては，木や，格子グラフ，外平面グラフ，平面グラフなどが挙げられる．また，経験的に，実世界に存在するグラフ構造の縮退数は，入力の頂点数に比べて小さいことが知られており，縮退数が小さいグラフに対する高速な列挙アルゴリズムも提案されている [5, 16, 20]．しかし，与えれた頂点数や辺数の定数縮退数を持つグラフの列挙問題に関しては結果が知られているものの [3]，定数縮退数を持つ部分グラフを列挙する問題については知らていなかった．この列挙問題に対して，我々は本稿の主結果として，次に掲げる定理 1，および，定理 2 で示す通り，初めてのならし多項式な列挙アルゴリズムを提案する．

定理 1. 任意のグラフ G と正整数 k に対して，k-縮退誘導グラフを $\mathcal{O}(n \log n + m)$ 時間の前処理の後，$\mathcal{O}(\Delta(k + \log n))$ ならし時間で列挙できる．ただし，n と m を，それぞれ，G の頂点数と辺数とし，Δ を G 中の最大次数とする．

定理 2. 任意のグラフ G と正整数 k に対して，k-縮退部分グラフを $\mathcal{O}(n \log n + m \log m)$ 時間の前処理の後，$\mathcal{O}(\log n)$ ならし時間で列挙できる．ただし，n と m を，それぞれ，G の頂点数と辺数とする．

2. 準備

無向グラフ $G = (V(G), E(G))$ を**頂点集合** $V(G)$ と**辺集合** $E(G)$ の組とし，$E(G) \subseteq V(G) \times V(G)$ とする．以下では，文脈から明らかならば，$V = V(G)$，また，$E = E(G)$ と略記する．また，$V = \{v_1, \ldots, v_n\}$ と $E = \{e_1, \ldots, e_m\}$ とし，頂点数と辺数を，それぞれ，n と m に固定する．G 中の相異なる 2 頂点 $u, v \in V$ に対して，$(u, v) = (v, u)$ とする．ここで，$(u, v) \in E$ のとき，u と v は**隣接している**といい，u に隣接する頂点集合を $N_G(u)$ とする．また，$d_G(u) = |N_G(u)|$ を u の G における**次数**とよび，$V(G, d)$ を，グラフ G 含まれる次数 d の頂点の集合とする．ここで，$d_G(u) = 0$ となる頂点 u を G 中の**孤立点**とよぶ．また，$\Delta(G) = \max_{u \in V} d_G(u)$ とし，単に Δ とかく．辺 $e = (u, v) \in E$ に対して，u と v を e の**端点**といい，e は u，あるいは，v に**接続する**という．ここで，$(u, v) \in E$ のとき，$M_G(u, v) = 1$ とし，$(u, v) \notin E$ のとき，$M_G(u, v) = 0$ とする．また，頂点 u が辺 e の端点であるとき，$u \in e$ とかく．また，グラフ G 中の次数が 1 以上，かつ，最も小さい，2 番目に小さい，および，3 番目に小さい次数を持つ頂点の次数を，それぞれ，$\delta_1(G)$，$\delta_2(G)$，および，$\delta_3(G)$ とする．ここで，たとえば，もっとも小さい次数を持つ頂点が 3 つ存在する場合，$\delta_1(G) = \delta_2(G) = \delta_3(G)$ となることに注意されたい．

グラフ G が**連結**であるとは，G 中の任意の 2 頂点 u, v に対して，$\pi = (u = u_1, u_2, \ldots, u_\ell = v)$ となる頂点列が存在し，これが任意の $1 < i \leq \ell$ に対して，$(u_{i-1}, u_i) \in E$ を満たすときをいう．さらに，π 中の頂点がすべて異なるとき，π を u から v への**路**という．グラフ G 中の辺 e を除くことで，G が非連結となるとき，この e を G の**橋**とよび，G 中の橋の集合を $B(G)$ とする．以下では，G は，自己閉路および多重辺を持たないとし，また，連結であると仮定する．

任意の頂点 v_i について，$id(v_i) = i$ を v の添字を返す関数とする．任意の 2 頂点に関して，その辞書順 $(V, \geq)$ を関数 id の大小で与える．また，辺については，2 つの異なる辺 $e_i = (a, b)$ と $e_j = (c, d)$ に対して，(1) 端点を共有する場合 ($a = c$ とする)，$b < d$ ならば，$e_i < e_j$ とし，(2) 端点を共有しない場合，$\min\{a, b\} < \min\{c, d\}$ ならば，$e_i < e_j$ とする．

V の任意の頂点部分集合 S に対して，$E[S] = \{(u, v) \in E \mid u, v \in S\}$ とする．また，E の任意の部分集合 F に対して，$V[F] = \{v \in V \mid v \in e \in F\}$ とする．ここで，$G[F] = (V[F], F)$ を，F **に関する** G **の部分グラフ**とよび，$G[S] = (S, E[S])$ を，S **に関する** G **の誘導グラフ**とよぶ．また，$G = (V, E)$ に対して，$E[S]$ は S から，$V[F]$ は F から一意に導かれることから，$G[S]$ と S，$G[F]$ と F を，それぞれ，同一視する．また，任意の頂点 $u, v \in V$ と任意の辺 $e = (u, v) \notin E$ に対して，$G - v = G[V \setminus \{v\}]$ とし，$G + e = G[E \setminus \{e\}]$ とする．

2.1 K-縮退グラフ

グラフ G が k-**縮退**であるとは，G の任意の誘導グラフ S に対して，S が次数 k 以下の頂点をもつときをいい，このような G を k-**縮退グラフ**という [11]．また，k を G の**縮退数**とよぶ．任意の k-縮退グラフについて，以下が成り立つ．

事実 1. グラフ $G = (V, E)$ が k-縮退ならば，そのときに限り，次の条件を満たす全単射関数 $f_G : V \to \{1, \ldots, n\}$ が存在する: 任意の $v \in V$ に対して，

$$\left|N_G^<(v)\right| \leq k \tag{1}$$

を満たす．ただし，$N_G^<(v) = \{u \in N_G(v) \mid f_G(v) < f_G(u)\}$ とする．

つまり，G が k-縮退ならば，ある頂点 v に着目したとき，v に隣接する頂点で，かつ，v よりも大きい頂点が高々 k 個となるような，頂点に対する順序付けが存在する．事

Algorithm 1: MB 法

```
1 手続き MB(G)
2     for i = 1, ..., n do
3         v ← G 中の最小次数の頂点;
          //複数ある場合は辞書順最小の頂点
4         G ← G − v;
5         f*_G(v) ← i;
6     return f*_G;
```

実 1 を満たす関数を**縮退関数**とよぶ．また，縮退関数を求める線形時間アルゴリズムを Matula と Beck が与えている [13]．また，定数の縮退数を持つようなグラフとして，木や，格子グラフ，直並列グラフ，外平面グラフ，平面グラフなどが知られており，縮退数は，それぞれ，1, 2, 2, 2, 5 である．

ここで，Matula と Beck のアルゴリズム（以下，MB 法とする）をもとに，**正規縮退関数** f^*_G を考える．MB 法は，入力グラフ G が空になるまで最小次数の頂点を繰り返し取り除くアルゴリズム（Algorithm 1）である．ここで，3 行目において，複数の頂点が候補となったとき，辞書順で最も小さい頂点を採用することで，f^*_G の一意性を担保する．

2.2 K-縮退部分グラフ列挙問題

本稿で扱う k-縮退部分グラフに対する列挙問題を次に与える．

問題 1. グラフ $G = (V, E)$ と正整数 k が与えられたとき，G に含まれるすべての k-縮退誘導グラフ S を漏れなく重複なく列挙せよ．

問題 2. グラフ $G = (V, E)$ と正整数 k が与えられたとき，G に含まれるすべての k-縮退部分グラフ S を漏れなく重複なく列挙せよ．

ここで注意されたいのは，非連結な k-縮退グラフも解に含まれる点である．以下では，問題 2，および，問題 1 に対する効率良いアルゴリズムを提案する．アルゴリズムは，ともに，Avis と Fukuda による逆探索法 [1] を元に構成する．

3. K-縮退誘導グラフ列挙アルゴリズム

本小節では，グラフ $G = (V, E)$ に含まれる k-縮退誘導グラフを列挙するアルゴリズムを与える．以下では，G 中の k-縮退誘導グラフすべてを含む集合を $\mathcal{D}_1(G, k)$ とする．ここで，文脈から明らかならば，$\mathcal{D}_1(G, k)$ を，単に，$\mathcal{D}_1$ とする．

まず，k-縮退誘導グラフに対して，その親を次のように定義する．ただし，$R = (\emptyset, \emptyset)$ とし，R を**根**と呼ぶ．

定義 1. $\mathcal{D}_1 \setminus \{R\}$ 中の任意の k-縮退誘導グラフ S に対して，その**親** $P_1(S)$ を

Algorithm 2: K-縮退誘導グラフ列挙

```
1 手続き Main(G = (V, E), k)
2     R ← (V, ∅);
3     Rec(G, R, k);
4 副手続き Rec(G, S, k)
5     S を出力;
6     foreach S' ∈ C_1(S) do
7         Rec(G, S', k);
```

$$P_1(S) = (S \setminus \{u_*(S)\}, E[S \setminus \{u_*(S)\}]) \tag{2}$$

と定義する．ただし，$u_*(S)$ を S 中の次数最小，かつ，そのような頂点の中でも辞書順最小の頂点とする．

上記の定義から，明らかに，親は唯一に決まる．ここで，$\mathcal{D}_1$ 上に，次のような有向グラフ $\mathcal{T}_1$ を定義する．

$$\mathcal{T}_1(G, k) = (\mathcal{D}_1, \mathcal{E}_1(\mathcal{D}_1)) \tag{3}$$

ただし，$\mathcal{E}_1(\mathcal{D}_1) = \{(S, P_1(S)) \mid S \in \mathcal{D}_1 \setminus \{R\}\}$ とする．また，文脈から明らかならば，$\mathcal{T}_1(G, k)$ を単に $\mathcal{T}_1$ とする．ここで，任意の $S \in \mathcal{D}_1 \setminus \{R\}$ に対して，繰り返し P を適用することで，R に明らかに到達する．さらに，$P_1(S)$ の頂点は，S の頂点数よりも真に小さい．以上の事実から，次の補題が言える．

補題 1. $\mathcal{T}_1$ は根付き有向木である．

本小節で与える列挙アルゴリズムは，$\mathcal{T}_1$ 上を根 R から深さ優先探索することで，列挙を行う．したがって，ある k-縮退誘導グラフから，その親へ辿るだけでなく，逆方向の操作も必要となる．そこで，まず，次のように k-縮退誘導グラフの子を定義する．

定義 2. $\mathcal{D}_1$ 中の任意の S と S' に対し，S' が S の**子**であるとは，$P_1(S') = S$ を満たすときをいい，S の**子集合**を

$$\mathcal{C}_1(S) = \{S' \in \mathcal{D}_1 \mid P_1(S') = S\} \tag{4}$$

とする．また，S の i 番目の子を $\mathcal{C}^i_1(S)$ とする．

上記の定義を使った列挙アルゴリズムを Algorithm 2 に与える．親子関係から，アルゴリズムの正当性は明らかである．以下では，時間計算量について考察する．Algorithm 2 のボトルネックは，6 行目の子集合の計算である．子集合を生成する最も単純なアルゴリズムは，S に対して，S に属さない頂点を加え，その親が S に一致するかを調べるアルゴリズムである．本稿では，この試行錯誤による手法よりも効率良い手法を提案する．まず，k-縮退誘導グラフ S' が k-縮退誘導グラフ S の子になる必要十分条件について考察する．

補題 2. $\mathcal{D}_1$ 中の任意の S と S' に対し，$S' = S \cup \{u\}$ とする．ここで，次の 2 つは等価である．

(1) $P_1(S') = S$.

(2) $0 \leq |N_G(u) \cap S| \leq k$ である．さらに，S 中で次数 $\delta_1(S)$ の任意の頂点 v に対し，次の (I) または (II) のいずれかを満たす：

(I) $|N_G(u) \cap S| < d_S(v) + M_G(u,v)$.
(II) $|N_G(u) \cap S| = d_S(v) + M_G(u,v)$，かつ，$u < v$.

証明. (1)→(2) を示す．親の定義から，$u = u_*(S')$ の次数は $0 \leq \delta_1(S') \leq k$ である．また，$\delta_1(S') = d_{S'}(u) = |N_G(u) \cap S|$ である．ここで，次の (A) と (B) に分けて考察する．

(A) $N_{S'}(u) \cap V(S', \delta_2(S')) = \emptyset$ とする．このとき，$\delta_1(S) = \delta_2(S')$ である．ここで，v を $V(S', \delta_2(S'))$ 中の任意の頂点とすると，v は $d_S(v) = \delta_1(S) = d_{S'}(v)$ である．また，u の定義より，$d_{S'}(u) < d_{S'}(v)$，または，$d_{S'}(u) = d_{S'}(v)$ ならば $u < v$ を満たす．したがって，(2) を満たす．

(B) $N_{S'}(u) \cap V(S', \delta_2(S')) \neq \emptyset$ とする．ここで，v を $N_{S'}(u) \cap V(S', \delta_2(S'))$ 中の任意の頂点とすると，$d_S(v) = \delta_1(S) = d_{S'}(v) - 1$ である．また，u の定義より，$d_{S'}(u) < d_{S'}(v)$，または，$d_{S'}(u) = d_{S'}(v)$ ならば $u < v$ を満たす．よって，(2) を満たす．

(2)→(1) を示す．u が $0 \leq |N_G(u) \cap S| \leq k$ を満たしているとする．さらに，$d_{S'}(v) = d_S(v) + M_G(u,v)$ であり，かつ，(I) あるいは (II) を満たしていることから，$u = u_*(S')$ である．したがって，$P_1(S') = S$ である．以上の議論から，題意は示された． □

上記の議論から，S の子集合を生成するための頂点に関する必要十分条件がわかった．次に，上記の条件を満たす頂点 u を，効率よく選ぶためのデータ構造について考察する．ここで，任意の整数 $i \in [0, n-1]$ に対して，$V(S,i)$ をヒープで保存する．ヒープは，関数 id をキーとする．このとき，$V(S,i)$ への頂点の追加削除は $\mathcal{O}(\log n)$ 時間，および，辞書順最小の頂点の検索は $\mathcal{O}(1)$ 時間で可能である．さらに，各頂点 v に対して，$\alpha_i(v)$ を v と隣接する，S における次数が i の頂点の数とし，$\alpha'_i(v)$ を v に隣接し，v より添字が小さい S における次数 i の頂点の数とする．また，任意の頂点 $u \in G$ に対して，$d_G(u) \leq \Delta$ であることから，次の補題が得られる．

補題 3. $\mathcal{D}_1^*(G,k)$ 中の任意の k-縮退誘導グラフ S と，任意の $i \in [0, n-1]$，任意の頂点 $v \in S$ に対して，S の頂点に関するデータ構造 $V(S,i)$，$\alpha_i(v)$，$\alpha'_i(v)$ から，その子 $S' \in \mathcal{C}_1(S)$ の頂点に関するデータ構造の計算に必要な必要な時間計算量は，$\mathcal{O}(\Delta(k + \log n))$ である．

上記の議論から，次の定理が導ける．

定理 1. 任意のグラフ G と正整数 k に対して，k-縮退誘導グラフを $\mathcal{O}(n \log n + m)$ 時間の前処理の後，$\mathcal{O}(\Delta(k + \log n))$ ならし時間で列挙できる．ただし，n と m を，それぞれ，G の頂点数と辺数とし，Δ を G 中の最大次数とする．

証明. まず，Algorithm 2 は，グラフ G と正整数 k が与えられたとき，明らかにすべての k-縮退誘導グラフを漏れなく重複なく出力する．ここで，ある k-縮退誘導グラフ S に対して，データ構造の更新時間は，補題 3 より，S から子 $S' = S \cup \{u\}$ を生成したとき，各子に対して，$\mathcal{O}(\Delta(k + \log n))$ 時間かかる．

S' に対する子の集合の計算には，補題 2 の (2) を満たす頂点 u を探せば良い．(I) を満たす頂点を探すには，$\delta_1(S)$ を次数とする任意の頂点 v と任意の $j < \delta_1(S)$ に対して，$\alpha_j(v)$ 中の頂点を探せばよく，また，(II) を満たす頂点を探すには，$\delta_1(S)$ を次数とし，かつ，辞書順最小の頂点 v に対して，$\alpha'_{\delta_1(S)}(v)$ 中の頂点を探せばよい．これにかかる時間計算量は，生成される子の数 $|\mathcal{C}_1(S)|$ に比例する．

以上の議論から，解一つを計算した後，そのデータ構造の更新，および，解の子集合の計算にかかる時間は，$\mathcal{O}(|\mathcal{C}_1(S)|\,\Delta(k + \log n))$ 時間である．したがって，アルゴリズム全体でかかる時間は，$\mathcal{O}(|\mathcal{D}_1^*(G,k)|\,\Delta(k + \log n))$ である．また，前処理には，頂点の大小関係の計算に $\mathcal{O}(n \log n + m)$ 時間必要である．以上で，定理は示された． □

4. K-縮退部分グラフ列挙アルゴリズム

本小節では，グラフ $G = (V, E)$ に含まれる k-縮退部分グラフを列挙するアルゴリズムを与える．以下では，G 中の k-縮退部分グラフすべてを含む集合を $\mathcal{D}_2(G,k)$ とし，G 中の n 頂点からなる k-縮退部分グラフの集合を $\mathcal{D}_2^*(G,k)$ とする．ここで，文脈から明らかならば，$\mathcal{D}_2(G,k)$ と $\mathcal{D}_2^*(G,k)$ を，それぞれ，単に $\mathcal{D}_2$ と $\mathcal{D}_2^*$ とする．本研究で与える k-縮退部分グラフの列挙アルゴリズムは，まず，$\mathcal{D}_2^*$ 中の k-縮退部分グラフ S を列挙し，次に，各 S に対して，適宜，S 中の孤立点を削除することで，$\mathcal{D}_2$ 中の k-縮退部分グラフを列挙する．

まず，k-縮退部分グラフに対して，その親を次のように定義する．ただし，$R = (\emptyset, \emptyset)$ とし，R を**根**と呼ぶ．

定義 3. $\mathcal{D}_2^* \setminus \{R\}$ 中の任意の k-縮退部分グラフ S に対して，その**親** $P_2(S)$ を

$$P_2(S) = (V, E(S) - e_*(S)) \tag{5}$$

と定義する．ただし，$e_*(S) = (u_*(S), v_*(S))$ とし，u_* を S 中の次数が 1 以上で最小，かつ，そのような頂点の中でも辞書順最小の頂点，さらに，v_* を u_* の S 中の隣接頂点の中でも，辞書順最小の頂点とする．

上記の定義から，明らかに，親は唯一に決まる．さらに，$P_2(S)$ は孤立点を持たない．ここで，$\mathcal{D}_2^*$ 上に，次のような有向グラフ $\mathcal{T}_2$ を定義する．

$$\mathcal{T}_2(G,k) = (\mathcal{D}_2^*, \mathcal{E}_2(\mathcal{D}_2^*)) \tag{6}$$

Algorithm 3: K-縮退部分グラフ列挙

```
1 手続き Main(G = (V, E), k)
2     R ← (V, ∅);
3     Rec(G, R, k);
4 副手続き Rec(G, S, k)
5     S を出力;
6     foreach S' ∈ C_2(S) do
7         Rec(G, S', k);
```

ただし，$\mathcal{E}_2(\mathcal{D}_2^*) = \{(S, P_2(S)) \mid S \in \mathcal{D}_2^* \setminus \{R\}\}$ とする．また，文脈から明らかならば，$\mathcal{T}_2(G,k)$ を単に $\mathcal{T}_2$ とする．ここで，任意の $S \in \mathcal{D}_2^* \setminus \{R\}$ に対して，繰り返し P を適用することで，R に到達する．さらに，任意の S に対して，$P_2(S)$ の辺数は，S の辺数よりも真に小さい．以上の事実から，次の補題が言える．

補題 4. $\mathcal{T}_2$ は根付き有向木である．

本小節で与える列挙アルゴリズムは，$\mathcal{T}_2$ 上を根 R から深さ優先探索することで，列挙を行う．したがって，ある k-縮退部分グラフから，その親へ辿るだけでなく，逆方向の操作も必要となる．そこで，まず，次のように k-縮退部分グラフの子を定義する．

定義 4. $\mathcal{D}_2^*$ 中の任意の S と S' に対し，S' が S の**子**であるとは，$P_2(S') = S$ を満たすときをいい，S の**子集合**を

$$\mathcal{C}_2(S) = \{S' \in \mathcal{D}_2^* \mid P_2(S') = S\} \qquad (7)$$

とする．また，S の i 番目の子を $\mathcal{C}_2^i(S)$ とする．

上記の定義を使った列挙アルゴリズムを Algorithm 3 に与える．親子関係から，アルゴリズムの正当性は明らかである．以下では，時間計算量について考察する．Algorithm 3 のボトルネックは，6 行目の子集合の計算である．子集合を生成する最も単純なアルゴリズムは，S に対して，S に属さない頂点を加え，その親が S に一致するかを調べるアルゴリズムである．本稿では，この試行錯誤による手法よりも効率良い手法を提案する．まず，k-縮退部分グラフ S' が k-縮退部分グラフ S の子になる必要十分条件について考察する．

補題 5. $\mathcal{D}_2$ 中の任意の相異なる 2 つの k-縮退部分グラフを S と S' とし，ある辺 $(u,v) \in E[S'] \setminus E[S]$ に対して，$S' = S + (u,v)$ が成り立つとする．ここで，次の 2 つは等価である．

(1) $P_2(S') = S$

(2) 次の (I)-(IV) のいずれかを満たす辺 (u,v) が存在する．ただし，$d_S(u) < k$ とする．

- (I) $d_S(u) = 0$，かつ，S に含まれる次数が 1 の任意の頂点 w に対し，$u < w$.
- (II) $0 < d_S(u) < \delta_2(S) - 1$，かつ，$v < \min N_S(u)$.
- (III) $0 < d_S(u) = \delta_2(S) - 1$，かつ，次の 2 つを満たす．
  - (III.a) S 中の任意の頂点 w に対し，$d_S(w) = \delta_2(S)$，かつ，$u < w$.
  - (III.b) $v < \min N_S(u)$.
- (IV) $0 < d_S(u) = d_S(v) = \delta_2(S)$，かつ，次のいずれかを満たす．
  - (IV.a) $d_S(u) < \delta_3(S) - 1$.
  - (IV.b) $d_S(u) = \delta_3(S) - 1$，かつ，次の 2 つを満たす．
    - (IV.b.1) S 中の任意の頂点 w に対し，$d_S(w) = \delta_3(S)$，かつ，$u < w$.
    - (IV.b.2) $v < \min N_S(u)$.

証明. (1)→(2) を示す．$P_2(S') = S$ とし，$u = u_*(S')$，$v = v_*(S')$ とする．ここで，$d_{S'}(u) = \delta_1(S')$ であり，また，$\delta_1(S) = d_S(u) = d_{S'}(u) - 1$ である．まず，S' は k-縮退であることから，$d_{S'}(u) \le k$ である．したがって，$d_S(u) < k$ である．つぎに，u の次数に関して，(A) から (D) の 4 つに場合分けして考える．

(A) $d_{S'}(u) = 1$ のとき，親の定義から，$d_S(u) = 0$ である．さらに，$u_*(S')$ の定義から，S' 中で次数が 1 の任意の頂点 w に対して，$u < w$.

(B) $1 < d_{S'}(u) < \delta_2(S')$ とする．まず，$v_*(S)$ の定義から，$v < \min N_S(u)$ である．(B.a) ここで，$\delta_2(S') < d_{S'}(v)$ ならば，$\delta_2(S) = \delta_2(S')$ であり，$d_S(u) < \delta_2(S) - 1$ である．(B.b) 一方，$\delta_2(S') = d_{S'}(v)$ ならば，$\delta_2(S) = \delta_2(S') - 1$ であり，$d_S(v) = \delta_2(S)$ である．さらに，$d_S(u) < \delta_2(S)$ である．したがって，$1 < d_{S'}(u) < \delta_2(S')$ ならば，$d_S(u) < \delta_2(S) - 1$ である．

(C) $1 < d_{S'}(u) = \delta_2(S') < d_{S'}(v)$ とする．ここで，$u_*(S')$ の定義から，$\delta_1(S') = d_{S'}(u)$ である．よって，$\delta_2(S) = \delta_2(S')$ であり，$d_S(u) = \delta_1(S') - 1$ であるので，$d_S(u) = \delta_2(S) - 1$ である．さらに，S' 中の次数が $\delta_1(S') = \delta_2(S')$ の頂点 w に対して，$u < w$ であり，また，$v_*(S')$ の定義から，$v < \min N_S(u)$ である．

(D) $1 < d_{S'}(u) = d_{S'}(v) = \delta_2(S')$ とする．ここで，u と v 以外の頂点の次数は S と S' で変わらないため，$\delta_3(S') = \delta_3(S)$ である．ここで，$u_*(S')$ の定義から，(D.a) $\delta_1(S') < \delta_3(S')$ を満たす，あるいは，(D.b) $\delta_1(S') = \delta_3(S)$，かつ，(D.b.i) S' 中の $\delta_1(S') = \delta_3(S')$ を次数とする任意の頂点 $w \neq u$ に対し $u < w$，かつ，(D.b.ii) $v_*(S')$ の定義から，$v < \min N_S(u)$ を満たす．以上の (A) から (D) の議論より，(1)→(2) は示された．

次に，(2)→(1) を示す．これを示すには，$P_2(S') = S$ を示すには，$e_*(S') = (u,v)$ であることを示せば良い．まず，

$d_S(u) < k$ であることから，このような e は存在する．以下では，(I) から (IV) のそれぞれについて考察する．

(I) を満たすとする．このとき，$d_{S'}(u) = 1$ である．さらに，S' 中で次数が 1 となる任意の頂点 w に対して $u < w$ であるので，S' 中で u に隣接する頂点を v とすると，$e_*(S') = (u, v)$ である．

(II) を満たすとする．このとき，$d_{S'}(u) + 1 < \delta_2(S) \leq \delta_2(S')$ である．したがって，u は S' 中で最小次数の唯一の頂点である．さらに，$v = \min N_{S'}(u)$ であるため，$e_*(S') = (u, v)$ である．

(III) を満たすとすると，$\delta_1(S) = d_S(u)$ である．ここで，$\delta_1(S')$ を次数とする u あるいは v でない頂点を w とする．このとき，$\delta_1(S) = \delta_1(S') = d_{S'}(w) = d_S(w)$ となるが，$d_S(u) = \delta_1(S)$ に矛盾する．一方，$d_{S'}(v) = d_S(v) = \delta_1(S')$ とすると，$\delta_1(S) \leq \delta_1(S') \leq \delta_1(S)+1$ から，$\delta_1(S)-1 \leq d_S(v) \leq \delta_1(S)$ である．これは，$\delta_1(S) < d_S(v)$ に矛盾する．したがって，$\delta_1(S') = d_{S'}(u)$ である．さらに，$d_{S'}(u) = d_S(u) + 1 = \delta_2(S)$ であり，かつ，(III.a) を満たすことから，$u = u_*(S')$ である．加えて，(III.b) を満たすので，$v = v_*(S')$ である．

(IV) を満たすとする．ここで, (IV.a) を満たすならば, u と v は S' 中で最小次数の頂点であることから, $e_*(S') = (u, v)$ である．一方, (IV.b) を満たすとする．このとき, $w \in \{u, v\}$ に対して，$d_{S'}(w) = d_S(w) + 1 = \delta_3(S) \leq \delta_3(S')$ となる．ここで，u と v は，それぞれ，(IV.b.1) と (IV.b.2) を満たす．したがって，$(u, v) = e_*(S')$ である．よって，上記の議論から題意は示された． □

補題 5 より，ある k-縮退部分グラフの S に対して，その子となる S' を求めるための必要十分条件がわかった．ここで，次の補題は自明に得られる．

補題 6. 補題 5 の (II) または，(III) を満たす辺が存在するとき，$|V(S, \delta_1(S))| = 1$，かつ，そのような頂点の次数は k 以下である．さらに，(IV) を満たす辺が存在するとき，$|V(S, \delta_1(S))| = |V(S, \delta_2(S))| = 1$，かつ，そのような頂点の次数は k 以下である．

任意の $d \in \{1, \ldots, n-1\}$ に対して，$V(S, d)$ 中の頂点をヒープで保存すると，次数 d の頂点集合に対して，頂点の追加削除が $\mathcal{O}(\log n)$，辞書順最小の頂点の検索が $\mathcal{O}(1)$ 時間で可能である．ただし，ヒープは関数 id をキーとする．したがって，次の補題が直ちに得られる．

補題 7. $\mathcal{D}_2^*(G, k)$ 中の任意の k-縮退部分グラフ S に対して，S の頂点に関するヒープから，その子 $S' \in \mathcal{C}_2(S)$ の頂点に関するヒープを計算するのに必要な時間計算量は，$\mathcal{O}(\log n)$ である．

証明． S と S' との間で，次数が変わる頂点の数は 2 つである．したがって，これらの頂点に対して，ヒープから頂点の追加削除をちょうど 2 回ずつ行うと，S' の頂点に関するヒープが得られる． □

以上の議論から，次の定理が導ける．

定理 2. 任意のグラフ G と正整数 k に対して, k-縮退部分グラフを $\mathcal{O}(n \log n + m \log m)$ 時間の前処理の後，$\mathcal{O}(\log n)$ ならし時間で列挙できる．ただし，n と m を，それぞれ，G の頂点数と辺数とする．

証明． $\mathcal{D}_2$ に含まれるすべての要素を列挙するには，$\mathcal{D}_2^*$ に含まれるすべての要素を列挙しながら，適宜孤立点を削除すれば良い．したがって，$\mathcal{D}_2^*$ に含まれるすべての要素を列挙するアルゴリズムについて考察する．以下では，S を $\mathcal{D}_2^*$ 中の k-縮退部分グラフとし，$A_{\mathrm{I}}(S)$ を S に含まれる，補題 5 の (I) を満たす辺の集合とする．$A_{\mathrm{II}}(S), A_{\mathrm{III}}(S), A_{\mathrm{IV}}(S)$ も同様に定義する．

Algorithm 3 は，親子関係の定義から，自明にすべての k-縮退部分グラフを列挙する．まず，$\mathcal{C}_2(S)$ が空集合かを判定するには，補題 6 の条件の判定，および，$A_{\mathrm{I}}(S)$ が空集合か判定すればよく，これらにかかる時間計算量は $\mathcal{O}(1)$ である．次に，補題 7 から，頂点に関するヒープの更新にかかる時間計算量は $\mathcal{O}(1)$ である．また，$A_{\mathrm{I}}(S)$ 中の辺 $e_1, \ldots, e_\ell$ とし，任意の $1 \leq i < j \leq \ell$ に対して，e_j を S に加えたとき，e_i は S に加えられないとする．つまり，$A_{\mathrm{I}}(S)$ 中の辺を加えてできる S の子の数は ℓ である．ここで，$A_{\mathrm{I}}(R)$ の計算をするのに $\mathcal{O}(m \log m)$ 時間必要である．一方で，S の子で，$A_{\mathrm{I}}(S)$ 中の辺を加えてできる j 番目の子 S' に対して，$A_{\mathrm{I}}(S')$ を計算するには，$j-1$ 個の辺を $A_{\mathrm{I}}(S)$ から除かなくてはいけないが，e_1 から順に S に追加していくことで，$A_{\mathrm{I}}(S')$ の計算にかかる時間は $\mathcal{O}(1)$ 時間であることがわかる．

次に，$\mathcal{C}_2(S)$ を計算する方法について考察する．前処理として，入力グラフの任意の頂点に対して，その隣接頂点集合は，辞書順で格納されているとする．また，補題 5 の (2) の u は明らかに S 中で次数最小，かつ，その中でも辞書順最小の頂点であり，これは $O(1)$ 時間で求めることができる．まず，補題 5 の (I) を満たす辺の集合は，上記の議論から $\mathcal{O}(1)$ 時間で求めることができる．次に，補題 5 の (II) を満たす辺の集合 $A_{\mathrm{II}}(S)$ は，u の隣接頂点は辞書順でソートしてあるため，$\mathcal{O}(|A_{\mathrm{II}}(S)|)$ 時間で求めることができる．次に，補題 5 の (III) を満たす辺の集合 $A_{\mathrm{III}}(S)$ とする．このとき，(III.a) の判定に $\mathcal{O}(1)$ 時間かかる．さらに，u の隣接頂点は辞書順でソートしてあるため，$A_{\mathrm{III}}(S)$ は，$\mathcal{O}(|A_{\mathrm{III}}(S)|)$ 時間で求めることができる．最後に，補題 5 の (IV) を満たす辺の集合 $A_{\mathrm{IV}}(S)$ とする．まず，(IV.a) の判定に $\mathcal{O}(1)$ 時間かかる．したがって，これを満たす辺の集合 $A'_{IV}(S)$ は，$\mathcal{O}(|A'_{IV}(S)|)$ 時間で計算できる．一方，(IV.b.1) の判定に $\mathcal{O}(1)$ 時間かかり，さらに，u の隣接頂点は辞書順でソートしてあるため，(IV.b) を満たす辺の集合 $A''_{IV}(S)$ の計算はには，$\mathcal{O}(|A''_{IV}(S)|)$ 時

間かかる．したがって，$A_{\mathrm{IV}}(S)$ をもとめるのかかる時間計算量は，$|O(A_{\mathrm{IV}}(S))|$ 時間である．

以上の議論から，$\mathcal{C}_2(S)$ の計算にかかる時間は，$\mathcal{O}(|\mathcal{C}_2(S)|)$ 時間であり，S の各子 S' のデータ構造の計算にかかる時間は，$\mathcal{O}(\log n)$ 時間であることから，解 1 つあたり，$\mathcal{O}(\log n)$ ならし時間で計算できる．また，入力グラフに対して頂点，ならびに，辺の辞書順でソートしておく必要があるため，前処理には，$\mathcal{O}(n \log n + m \log m)$ 時間必要である． □

5. おわりに

本稿では，k-縮退誘導グラフ列挙問題，および，k-縮退部分グラフ列挙問題について考察した．主結果として，これらをならし多項式時間で解く列挙アルゴリズムを提案した．今後の課題として，より高速な列挙アルゴリズムの開発，および，極大な k-縮退誘導グラフおよび部分グラフ列挙問題を解く列挙アルゴリズムの開発が挙げられる．

参考文献

[1] Avis, D. and Fukuda, K.: Reverse search for enumeration, *Discrete Applied Mathematics*, Vol. 65, No. 1-3, pp. 21–46 (1996).

[2] Basavaraju, M., Heggernes, P., van t Hof, P., Saei, R. and Villanger, Y.: Maximal Induced Matchings in Triangle-Free Graphs, *WG 2014: the 40th International Workshop on Graph-Theoretic Concepts in Computer Science*, Lecture Notes in Computer Science, Vol. 8747, Springer International Publishing, pp. 93–104 (online), DOI: 10.1007/978-3-319-12340-0_8 (2014).

[3] Bauer, R., Krug, M. and Wagner, D.: Enumerating and Generating Labeled k-degenerate Graphs, *ANALCO 2010: the 7th Workshop on Analytic Algorithmics and Combinatorics*, Society for Industrial and Applied Mathematics, pp. 90–98 (online), DOI: 10.1137/1.9781611973006.12 (2010).

[4] Birmelé, E., Ferreira, R., Grossi, R., Marino, A., Pisanti, N., Rizzi, R. and Sacomoto, G.: Optimal Listing of Cycles and st-Paths in Undirected Graphs, *In proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012)*, New Orleans, LA, USA, pp. 1884–1896 (online), DOI: 10.1137/1.9781611973105.134 (2012).

[5] Conte, A., Grossi, R., Marino, A. and Versari, L.: Sublinear-Space Bounded-Delay Enumeration for Massive Network Analytics: Maximal Cliques, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), Vol. 55, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 148:1–148:15 (online), DOI: 10.4230/LIPIcs.ICALP.2016.148 (2016).

[6] Eppstein, D.: Small Maximal Independent Sets and Faster Exact Graph Coloring, *Journal of Graph Algorithms and Applications*, Vol. 7, No. 2, pp. 131–140 (online), DOI: 10.7155/jgaa.00064 (2003).

[7] Fukuda, K. and Matsui, T.: Finding All the Perfect Matchings in Bipartite Graphs, *Applied Mathematics Letters*, Vol. 7, No. 1, pp. 15–18 (online), DOI: 10.1016/0893-9659(94)90045-0 (1994).

[8] Girvan, M. and Newman, M. E. J.: Community structure in social and biological networks, *Proceedings of the National Academy of Sciences*, Vol. 99, No. 12, pp. 7821–7826 (online), DOI: 10.1073/pnas.122653799 (2002).

[9] Johnson, D. S., Yannakakis, M. and Papadimitriou, C. H.: On generating all maximal independent sets, *Information Processing Letters*, Vol. 27, No. 3, pp. 119–123 (online), DOI: 10.1016/0020-0190(88)90065-8 (1988).

[10] Kimura, D., Kuboyama, T., Shibuya, T. and Kashima, H.: A Subpath Kernel for Rooted Unordered Trees, *In Proceedings of the 15th Pacific-Asia Conference of Advances in Knowledge Discovery and Data Mining (PAKDD2011)*, Vol. 2, pp. 62–74 (online), DOI: 10.1007/978-3-642-20841-6_6 (2011).

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

[12] Makino, K. and Uno, T.: New algorithms for enumerating all maximal cliques, *SWAT 2004: the 9th Scandinavian Workshop on Algorithm Algorithm Theory*, Lecture Notes in Computer Science, Vol. 3111, Springer Berlin Heidelberg, pp. 260–272 (online), DOI: 10.1007/978-3-540-27810-8_23 (2004).

[13] 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).

[14] Read, R. C. and Tarjan, R. E.: Bounds on backtrack algorithms for listing cycles, paths, and spanning trees, *Networks*, Vol. 5, No. 3, pp. 237–252 (1975).

[15] Shioura, A., Tamura, A. and Uno, T.: An Optimal Algorithm for Scanning All Spanning Trees of Undirected Graphs, *SIAM Journal on Computing*, Vol. 26, No. 3, pp. 678–692 (online), DOI: 10.1137/S0097539794270881 (1997).

[16] Tomita, E., Tanaka, A. and Takahashi, H.: The Worst-Case Time Complexity for Generating All Maximal Cliques and Computational Experiments, *Theoretical Computer Science*, Vol. 363, No. 1, pp. 28–42 (online), DOI: 10.1016/j.tcs.2006.06.015 (2006).

[17] Uno, T.: A Fast Algorithm for Enumeration of Maximal Matchings in General Graphs, *Journal of National Institute of Informatics*, Vol. 3, No. -, pp. 89–97 (2001).

[18] Uno, T., Maegawa, H., Nakahara, T., Hamuro, Y., Yoshinaka, R. and Tatsuta, M.: Micro-Clustering: Finding Small Clusters in Large Diversity, *CoRR*, Vol. abs/1507.03067 (online), available from ⟨http://arxiv.org/abs/1507.03067⟩ (2015).

[19] Wasa, K.: Enumeration of Enumeration Algorithms, *CoRR*, Vol. abs/1605.05102 (online), available from ⟨http://arxiv.org/abs/1605.05102⟩ (2016).

[20] Wasa, K., Arimura, H. and Uno, T.: Efficient Enumeration of Induced Subtrees in a K-Degenerate Graph, *ISAAC 2014: the 25th International Symposium on Algorithms and Computation*, Lecture Notes in Computer Science, Vol. 6506, Springer Berlin Heidelberg, pp. 94–102 (2014).