

Caterpillar の列挙アルゴリズム

菊地 洋右 田中 博行 中野 眞一 柴田 幸夫

群馬大学 工学部 情報工学科

376-8515 群馬県桐生市天神町 1-5-1

{kikuchi,hiroyuki,nakano,shibata}@msc.cs.gunma-u.ac.jp

要旨 キヤタピラを重複なく、かつ、もれなく生成する簡単なアルゴリズムを提案する。キヤタピラは木の部分クラスの一つである。本列挙アルゴリズムは、すべてのキヤタピラを1個あたり定数時間で生成する。さらに本列挙アルゴリズムを利用して、いくつかの木の部分クラスに対する列挙アルゴリズムも与える。
キーワード: キヤタピラ, 列挙アルゴリズム, 平面キヤタピラ, スパイダー, スコーピオン

How to obtain the complete list of caterpillars

Yosuke Kikuchi, Hiroyuki Tanaka, Shin-ichi Nakano and Yukio Shibata

Department of Computer Science, Gunma University

Kiryu, Gunma, 376-8515, Japan

{kikuchi,hiroyuki,nakano,shibata}@msc.cs.gunma-u.ac.jp

abstract We propose a simple algorithm to generate all caterpillars without repetition. Also some other generation algorithms are presented. Our algorithm generates each caterpillar in constant time per caterpillar. Also some other generation algorithms are presented.
keywords: caterpillar, generating algorithm, plane caterpillar, spider, scorpion.

1 はじめに

グラフの個数の数え上げには様々な研究がある。例えば、指定されたクラスのグラフの個数の数え上げ [2, 3, 7, 10], 指定されたグラフの指定されたクラスの部分グラフの個数の数え上げ [4, 6, 8] などである。これらの研究は、主に組み合わせ論の手法を用いてなされている。一方、それらのグラフのすべてを実際に計算機等で出力することは、グラフの個数が一般に膨大であるため困難であった。しかし、近年の計算機の性能の向上に伴い、計算機による列挙が可能になり、様々なアルゴリズムも提案されてきた [1, 5, 9]。ここで、重複なく、かつ、もれなく、指定されたクラスのグラフを生成することを、指定されたクラスのグラフの列挙と呼ぶ。本論文は、木の主要な部分クラスである caterpillar の列挙に関するものである。

これまでに、様々な、木の列挙に関する研究がある。例えば、根付き木の列挙アルゴリズム [1] や、平面植木の列挙アルゴリズム [9] などである。また、頂点の個数が $n+4$ である caterpillar の個数は、 $2^n + 2^{\lfloor n/2 \rfloor}$ であることが知られている [7]。

本論文では、まず、頂点の個数が n であり、直径

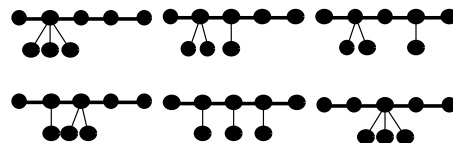


図 1: 頂点数 8, 直径 4 の caterpillar

が d である caterpillar を列挙するアルゴリズムを設計する。頂点の個数が 8 であり、直径が 4 である caterpillar の全てを、図 1 に示す。

次に、このアルゴリズムを用いて、頂点の個数が n である caterpillar を列挙するアルゴリズムを設計する。

次に、頂点の個数が n である全ての caterpillar の、全ての平面埋め込みを列挙するアルゴリズムを設計する。

さらに、caterpillar の列挙アルゴリズムと同様のアイデアを用いて、二つのクラスの木の列挙アルゴリズムを設計する。この二つのクラスとは、spider と scorpion である。

本論文の構成は次のようである。2章では、本論文で用いる定義を与える。第3章では、頂点の個数が n であり、直径が d であるような caterpillar を列挙するアルゴリズムを与える。第4章では、このアルゴリズムを高速化する。第5章では、頂点の個数が n であるような caterpillar を列挙するアルゴリズムと、頂点の個数が n である全ての caterpillar の、全ての平面埋め込みを列挙するアルゴリズムを与える。第6章では、さらに二つのアルゴリズムを設計する。頂点の個数が n である spider と、頂点数 n の scorpion とを列挙するアルゴリズムである。第7章は、まとめと今後の課題である。

2 諸定義

本論文では、有限・無向・単純なグラフのみを扱う。 $G = (V, E)$ を頂点集合 V と辺集合 E を持つグラフとする。グラフ G の、頂点 v に接続する辺の本数を $d(v)$ と書き、 v の次数と呼ぶ。グラフ G の二つの頂点 u, v に対して、その最短パスの長さを $\ell(u, v)$ で表す。ここで、パスの長さとはパスに含まれる辺の本数である。グラフ G の頂点 u に対して、 $\max_{v \in V(G)} \ell(u, v)$ を u の離心率と呼び、 $\text{ecc}(u)$ で表わす。グラフ G に対して、 $\max_{u \in V(G)} \text{ecc}(u)$ を G の直径と呼び、 $\min_{u \in V(G)} \text{ecc}(u)$ を G の半径と呼ぶ。サイクルを含まない連結グラフを木という。

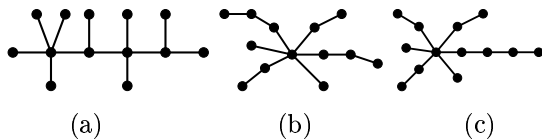


図 2: (a)caterpillar (b)spider (c)scorpion の例図

木の頂点で、次数が 1 であるものを葉と呼ぶ。パス P_n とは辺の本数が n であり、各頂点の次数が高々 2 である木である。全ての葉を除去するとパスとなる木を caterpillar という。本論文では、点の個数が 3 以上の caterpillar のみを扱う。caterpillar の例を図 1 に示す。caterpillar の葉を foot と呼び、foot と foot 以外の頂点を結ぶ辺を leg と呼ぶ。直径が d の caterpillar が与えられたとき、長さが d のパスをひとつ選び、backbone と呼ぼう。backbone を $P_d = (u_0, u_1, \dots, u_d)$ のように点列で書く。

spider とは、次数が 3 以上の頂点を高々 1 つ持つ木である。次数が 3 以上の頂点を spider の body と呼ぶ。body と葉を結ぶパスを leg と呼び、leg に含まれる辺の数を leg の長さと呼ぶ。1 つの leg を除いた他の全ての leg の長さが 2 以下のとき、spider は scorpion と呼ばれる。

3 直径 d の caterpillar の列挙

3.1 caterpillar に固有のラベル列

本章は、頂点の個数が n であり、直径が d である全ての caterpillar を、列挙するアルゴリズムを与える。このとき、foot の個数は $n - d + 1$ であることに注意しよう。

まず、準備として、各々の caterpillar に固有のラベル列を次のように定義しよう。

与えられた caterpillar C の直径は d であり、backbone P_d を任意に選んだとしよう。backbone に含まれない頂点を caterpillar の cocoon と呼ぶ。 $P_d = (u_0, u_1, \dots, u_d)$ とする。各頂点 u_i に隣接する cocoon の数を、各々、 $l(u_i)$ とする。cocoon の総数は $n - d - 1$ であるので、各 i について、 $l(u_i) \leq n - d - 1$ であることに注意されたい。また、各 $l(u_i)$ の総和は $n - d - 1$ となる。これらの $l(u_i)$ を対応する点が P_d に現れる順に並べたラベル列を $L = \{l(u_0), l(u_1), \dots, l(u_d)\}$ としよう。例えば、図 1 の左上の caterpillar において、 P_d を左から右への水平線分列に選ぶと $L = (0, 3, 0, 0, 0)$ であり、 P_d を右から左への水平線分列に選ぶと、 $L = (0, 0, 0, 3, 0)$ である。

caterpillar の構造より、 $l(u_0) = l(u_d) = 0$ である。よって、この二つのラベルを今後はラベル列から除外することにしよう。

また、どのように P_d を選んでも、ラベル列は高々 2 種類しか得られないことが容易にわかる。(なぜならば、caterpillar から全ての foot を除去するとパスになり、 P_d はこのパスを必ず含むので、本質的に、 P_d はこのパスをどちらの方向に含むかの 2 種類しかないからである。) また、2 種類のラベル列が得られたとき、それらは互いに他方のラベルを逆順に並べたものになっている。(ここで、対称性より、ラベル列が 1 種類しか得られない場合もあることに注意しよう。例えば、図 1 右下の caterpillar では $L = (0, 3, 0)$ の 1 種類しか得られない。) そこで、もし 2 種類のラベル列が得られた場合には、それぞれのラベル列を $d - 1$ 桁の $n - d$ 進数とみなしたとき、値が大きくなる方を選び、これを caterpillar C のラベル列であると定義しよう。例えば、図 1 の左上の caterpillar のラベル列は $d = 4$ 、 $n = 8$ であるので、3 桁の 4 進数とみなすことができ、 $(3, 0, 0) > (0, 0, 3)$ であるので $(3, 0, 0)$ を選ぶ。

また、caterpillar の構造より、二つのラベル列が異なれば、対応する二つの caterpillar は同型ではないことがわかる。すなわち、上で定義したラベル列は各 caterpillar について固有のものとなる。

本論文では、ラベル列 $L_1 = \{l(u_1), l(u_2), \dots, l(u_{d-1})\}$ と、そのラベルを逆順に並べたもの $L_2 = \{l(u_{d-1}), l(u_{d-2}), \dots, l(u_1)\}$ のそれぞれを $d - 1$ 桁

の $n-d$ 進数とみなしたとき, L_1 が L_2 より真に大きいならば, L_1 を正ラベル列と呼び, L_2 を逆ラベル列と呼ぶことにする. 特に, L_1 と L_2 が等しいときは, これらを対称ラベル列と呼ぶ.

もし, 各々の caterpillar に対応する固有のラベル列が列挙できれば, この列挙は caterpillar の列挙に相当する.

d と n が与えられたとき, 点の個数が n であり直径が d である各 caterpillar の, 各固有ラベル列を列挙するアルゴリズムを設計しよう. 我々のアルゴリズムは, これらのラベル列を $d-1$ 桁の $n-d$ 進数とみなしたとき, ラベル列を降順に列挙する. すなわち, $d-1$ 桁の $n-d$ 進数とみなせるラベル列で, 各桁の総和が $n-d-1$ (=cocoon の総数) であり, かつ, 正ラベル列もしくは対称ラベル列であるものすべてを, 列挙すればよい.

例えば, 頂点の個数が 8 であり, 直径が 4 である各 caterpillar は, それぞれ, 3 桁の 4 進数とみなせるラベル列に対応する. 3 桁の 4 進数とみなせるラベル列を降順に列挙すると下記のようになる.

$$\begin{aligned} & (3,0,0) \rightarrow (2,1,0) \rightarrow (2,0,1) \\ \rightarrow & (1,2,0) \rightarrow (1,1,1) \rightarrow (1,0,2) \\ \rightarrow & (0,3,0) \rightarrow (0,2,1) \rightarrow (0,1,2) \\ \rightarrow & (0,0,3). \end{aligned}$$

これらの中に, 正ラベル列と対称ラベル列は合計 6 個あり, この 6 個のラベル列が図 1 の 6 個の caterpillar に対応する.

ここで, 上記のラベル列の総数は簡単に求めることができる. すなわち, $n-d-1$ 個の同一種類の cocoon の, $d-1$ 個のラベル付きの点への分割の総数と同じであるから, 重複組み合わせの個数となり, $\binom{(n-d-1)+(d-1)-1}{(d-1)-1} = \binom{3+3-1}{2} = 10$ となる.

3.2 頂点数 n , 直径 d である caterpillar の総数

正ラベル列と対称ラベル列の総数を求める. 頂点数 n , 直径 d の caterpillar について, backbone を $P_d = u_0, u_1, \dots, u_d$ とする. また頂点数 n , 直径 d の非同型な caterpillar の総数を $g(n, d)$ で表す. つまり, $g(n, d)$ は正ラベル列と対称ラベル列の総数である.

$d-1$ 桁の $n-d$ 進数で, 各桁の和が $n-d-1$ となるラベル列で, その総数は正ラベル列, 逆ラベル列, 対称ラベル列を合わせて

$$\binom{(n-d-1)+(d-1)-1}{(d-1)-1} = \binom{n-3}{d-2}$$

である.

生成されるラベル列について, 正ラベル列と逆ラベル列の個数は等しい. 対称ラベル列の個数を S で表すと,

$$g(n, d) = \frac{\binom{n-3}{d-2} + S}{2}$$

が成り立つ.

そこで対称ラベル列の総数を求める. 以下の 4 つの場合に分け, 考える.

Case 1 d が奇数かつ n が奇数,

Case 2 d が奇数かつ n が偶数,

Case 3 d が偶数かつ n が奇数,

Case 4 d が偶数かつ n が偶数.

Case 1 $d-1$ が偶数, $n-d-1$ が奇数となる. このとき $d-1$ 桁の $n-d$ 進数で, 各桁の和が $n-d-1$ となる対称ラベル列は存在しない. よって $S=0$ である.

Case 2 $d-1$ と $n-d-1$ がともに偶数となる.

このとき, 対称ラベル列に対応する caterpillar において, backbone 上の $\frac{d-1}{2}$ 個の頂点 $u_1, u_2, \dots, u_{\frac{d-1}{2}}$ に隣接する cocoon の総数は $\frac{n-d-1}{2}$ となっている.

このとき, 対称ラベルの総数は $\frac{n-d-1}{2}$ 個の同一種類の cocoon を, $\frac{d-1}{2}$ 個のラベル付頂点への分割の総数と等しく,

$$S = \binom{\frac{n-d-1}{2} + \frac{d-1}{2} - 1}{\frac{d-1}{2} - 1} = \binom{\frac{n-4}{2}}{\frac{d-3}{2}}$$

である.

Case 3 $d-1$ が奇数, $n-d-1$ が偶数となる.

このとき, caterpillar のセンターは backbone 上の頂点 $u_{\frac{d}{2}}$ のみである. $d-1$ が奇数, $n-d-1$ が偶数より, 対称ラベルに対応する caterpillar において, $u_{\frac{d}{2}}$ に隣接する頂点数は偶数である. つまり,

$$l(u_{\frac{d}{2}}) = 2i, \left(i = 0, 1, \dots, \frac{n-d-1}{2} \right)$$

である. また, backbone 上の $\frac{d-2}{2}$ 個の頂点, $u_1, u_2, \dots, u_{\frac{d-2}{2}}$

に隣接する cocoon の総数は $\frac{n-d-1-l(u_{\frac{d}{2}})}{2}$ となっている.

このとき,

対称ラベルの総数は $\frac{n-d-1-l(u_{\frac{d}{2}})}{2}$ 個の

同一種類の cocoon を, $\frac{d-2}{2}$ 個のラベル付頂点へ

の分割の総数と等しく,

$$\begin{aligned} S &= \sum_{i=0}^{\frac{n-d-1}{2}} \left(\frac{n-d-1-2i}{2} + \frac{d-2}{2} - 1 \right) \\ &= \sum_{i=0}^{\frac{n-d-1}{2}} \left(\frac{n-2i-5}{2} \right) \end{aligned}$$

である.

ここで $x = \frac{n-5}{2}$, $y = \frac{d-4}{2}$ とおくと,

$$\begin{aligned} S &= \sum_{i=0}^{x-y} \binom{x-i}{y} \\ &= \binom{x+1}{y+1} \\ &= \binom{\frac{n-3}{2}}{\frac{d-2}{2}} \end{aligned}$$

ゆえに Case 3 において対称ラベル列の個数は $\binom{\frac{n-3}{2}}{\frac{d-2}{2}}$ である.

Case 4 $d-1$ が奇数, $n-d-1$ も奇数となる.

Case 3 と同様に caterpillar のセンターは backbone 上の頂点 $u_{\frac{d}{2}}$ のみである. $d-1$ が奇数, $n-d-1$ が奇数より, 対称ラベルに対応する caterpillar において, $u_{\frac{d}{2}}$ に隣接する頂点数は奇数である. つまり $l(u_{\frac{d}{2}}) = 2i+1$, ($i=0, 1, \dots, \frac{n-d-2}{2}$) である. また, backbone 上の $\frac{d-2}{2}$ 個の頂点, $u_1, u_2, \dots, u_{\frac{d-2}{2}}$ に隣接する cocoon の総数は $\frac{n-d-1-l(u_{\frac{d}{2}})}{2}$ となっている.

このとき, 対称ラベルの総数は $\frac{n-d-1-l(u_{\frac{d}{2}})}{2}$ 個の同一種類の cocoon を, $\frac{d-2}{2}$ 個のラベル付頂点への分割の総数と等しく,

$$\begin{aligned} S &= \sum_{i=0}^{\frac{n-d-2}{2}} \left(\frac{n-d-1-(2i+1)}{2} + \frac{d-2}{2} - 1 \right) \\ &= \sum_{i=0}^{\frac{n-d-2}{2}} \left(\frac{n-2i-6}{2} \right) \end{aligned}$$

である.

ここで $x = \frac{n-6}{2}$, $y = \frac{d-4}{2}$ とおくと,

$$\begin{aligned} S &= \sum_{i=0}^{x-y} \binom{x-i}{y} \\ &= \binom{x+1}{y+1} \\ &= \binom{\frac{n-4}{2}}{\frac{d-2}{2}}. \end{aligned}$$

ゆえに Case 4 において対称ラベル列の個数は $\binom{\frac{n-4}{2}}{\frac{d-2}{2}}$ である.

以上より Case 1 において, $S=0$, Case 2 から 4 において, $S = \binom{\left\lfloor \frac{n-4}{2} \right\rfloor}{\left\lfloor \frac{d-2}{2} \right\rfloor}$ がいえる.

ゆえに頂点数 n , 直径 d の caterpillar の総数 $g(n, d)$ は

$$g(n, d) = \begin{cases} \frac{\binom{n-3}{d-2}}{2} & d \text{ と } n \text{ が共に奇数のとき,} \\ \frac{\binom{n-3}{d-2} + \binom{\left\lfloor \frac{n-4}{2} \right\rfloor}{\left\lfloor \frac{d-2}{2} \right\rfloor}}{2} & \text{それ以外のとき} \end{cases}$$

となる.

3.3 単純な caterpillar の列挙アルゴリズム

3.2 で求めたように頂点数 n , 直径 d の caterpillar は膨大である. そこで, 頂点数 n , 直径 d の caterpillar に対応するラベル列の全てを生成し, 列挙するには, どうしたらよいのであろうか?

一般に, a 桁の b 進数と見なせるラベル列で, 各桁の総和が定数であるもの全てを, 降順に列挙したとき, (c_1, c_2, \dots, c_a) の直後のラベルは, 下記のように定義できる.

場合 A: $1 \leq i \leq a-1$ なる i のうち, $c_i \neq 0$ を見たすものが存在するとき.

そのような i のうち, 最大の i を選び, $s=i$ としよう. (c_1, c_2, \dots, c_a) の直後のラベル列は $(c_1, c_2, \dots, c_s-1, 1+c_s, 0, 0, \dots, 0)$ である. (場合 A 終)

場合 B: $1 \leq i \leq a-1$ なる i のすべてにおいて, $c_i = 0$ のとき.

(c_1, c_2, \dots, c_a) の直後のラベル列はない. すなわち, このラベル列は, 列挙すべき最後のラベル列である.

(場合 B 終)

すなわち, あるラベル列が与えられたとき, このラベル列の直後のラベル列は次のようにして求めることができる.

最下位桁以外の桁のうち値が 0 以外である最も下位の桁を選ぶ. これを第 s 桁とする.

もし $s = d-2$, すなわち, 選んだ桁のすぐ下位の桁が最下位の桁である場合は, 第 s 桁 (=第 $d-2$ 桁) の値を 1 だけ減らし, 最下位の桁 (=第 $s+1$ 桁) の値を 1 だけ増やす.

もし $s < d-2$, すなわち, 選んだ桁のすぐ下位の桁が最下位の桁でない場合は, 第 s 桁の値を 1 だけ減らし, 第 $s+1$ 桁の値を最下位の桁の値より 1 だけ増やしたものとし, さらに, 最下位の桁の値を 0 にする. 桁の選び方より, 第 $s+2$ 桁と, 最下位の桁の間の各桁の値はすべて 0 であることに注意されたい.

例えば, $(1, 2, 3, 4, 5)$ の直後のラベルは $(1, 2, 3, 3, 6)$ であり, $(1, 2, 3, 0, 2)$ の直後のラベルは $(1, 2, 2, 3, 0)$ である.

このようにして, 生成されたラベル列から, 正ラベル列と対称ラベル列のみを選んで出力すれば, 点の個数が n であり直径が d である caterpillar を列挙することができる. アルゴリズムは次の通りである.

Algorithm generate-all-caterpillars(d, n)
 $\{ d-1$ 桁の $n-d$ 進数とみなせるラベル列で, 各桁の値の総和が $n-d-1$ であるものを列挙する. }

- 1 $(c_1, c_2, \dots, c_{d-1}) = (n-d-1, 0, 0, \dots, 0)$ とし, このラベル列を出力する;
- 2 **while** $1 \leq i \leq d-2$ なる i のうち, $c_i \neq 0$ を満たすものが存在する;
- 3 **do** { まだ次のラベル列がある }
begin
- 4 そのような i のうち, 最大の i を選び, $s = i$ とする;
- 5 **if** $s = d-2$
- 6 **then** c_{d-2} の値を 1 減らし, c_{d-1} の値を 1 増やす;
else $\{s < d-2\}$
- 7 **then**
- 8 c_s の値を 1 減らし, c_{s+1} の値を $c_{d-1} + 1$ とし, c_{d-1} の値を 0 とする;
- 9 **if** 得られたラベル列が正もしくは対称ラベル

列である

10 **then** 得られたラベル列を出力する;
end

補題 3.1 このアルゴリズムは頂点数が n であり, 直径が d であるすべての caterpillar を列挙する. このアルゴリズムの作業用計算領域は $O(d)$ であり, 計算時間は $O(d \cdot g(n, d))$ である. ここで, $g(n, d)$ は頂点の個数が n であり, 直径が d であるような caterpillar の総数である.

Proof. このアルゴリズムは $d-1$ 桁のラベル列を, 次々に更新していくものである. $d-1$ 桁のラベル列の他に必要なメモリは高々定数個であるので, 作業用計算領域が $O(d)$ であることは自明である.

アルゴリズムの 2,4 行目は, 次の工夫により, 高速に実行できる. $1 \leq i \leq d-2$ なる i のうち, $c_i \neq 0$ なる最大のものを s とする. このとき, c_1, c_2, \dots, c_s のうち, 値が 0 以外であるもののみを, 順に双方向リストで結ぶ. このような双方向リストの更新は, ラベル列ひとつにつき $O(1)$ 時間で行える. この双方向リストの最後の部分をチェックすることにより, アルゴリズムの 2,4 行目は 1 回あたり $O(1)$ 時間しかかからない.

6,8 行目は $O(1)$ 時間かかる. ここで, 更新されるラベルは高々三つであることに注意しよう.

9,10 行目は 1 回あたり $O(d)$ 時間かかる.

このアルゴリズムは生成したラベル列の少なくとも半数以上を出力する. なぜなら, 正ラベル列と逆ラベル列は同じ個数だけ存在し, アルゴリズムは逆ラベル列のみ生成後に出力することなく捨てるが, 正ラベル列と対称ラベル列は全てを出力するからである.

以上より, このアルゴリズムの計算時間は $O(d \cdot g(n, d))$ である. Q.E.D.

4 高速化

前章のアルゴリズム generate-all-caterpillars は, $d-1$ 桁の $n-d$ 進数とみなせるラベル列で, 各桁の値の和が $n-d-1$ であるものすべてを次々に生成し, それぞれが正ラベル列・対称ラベル列・逆ラベル列のいずれであるのかをチェックし, 正ラベル列と対称ラベル列のみを出力する. このチェックは, 簡易な方法を用いるとラベル列 1 つあたり $O(d)$ 時間かかる.

しかし, 次に説明するデータ構造を用いると, このチェックをラベル列 1 つあたり $O(1)$ 時間で行なうことができる.

$1 \leq i \leq d-1$ なる各 i について, ラベル c_i と c_{d-i} が同じであるものを記憶しよう. 図 3 を参照されたい. 図では, そのようなデータに対応する桁を灰色で示している. さらに, それらのうち連続して表わ

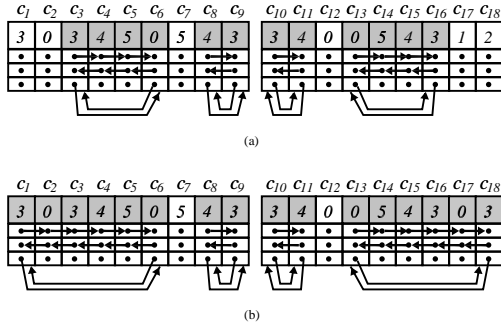


図 3: 高速化のためのデータ構造

れるものを、双方向リストで結ぶことにする。さらに、そのような連続の極大の部分列の最初と最後の桁が、互いに $O(1)$ 時間でアクセスできるように双方向ポインタで結ぼう。

ラベル列 $L = (c_1, c_2, \dots, c_{d-1})$ が与えられたとしよう。このラベル列と、このラベル列の次に生成されるラベル列の違いは、高々 3 個所のラベルのみである。(そのうち二つは常に連続しており、それは、アルゴリズム 4 行目で選ばれる c_s と c_{s+1} である。 c_s は 1 だけ値が減り、 c_{s+1} は $c_{d-1} + 1$ となる。特に、 $s \neq d - 2$ のときは、 c_{d-1} は 0 になる。) よって、上記のデータ構造はラベル列 1 個あたり定数時間で更新できる。

このデータ構造により、ラベル列が与えられたとき、灰色でない部分で最も下位にある桁に $O(1)$ 時間でアクセスすることができる。すなわち、 $O(1)$ 時間で、正ラベル列・逆ラベル列・対称ラベル列のいずれであるかの判定ができる。上記の工夫により次がいえる。

補題 4.1 アルゴリズム **generate-all-caterpillars** の作業用計算領域は $O(d)$ 、計算時間は $O(g(n, d))$ である。ここで、 $g(n, d)$ は頂点の個数が n 、直径が d であるような *caterpillar* の個数である。すなわち、アルゴリズムは *caterpillar* を 1 個あたり平均 $O(1)$ 時間で生成する。

上記の方法を、さらに、正ラベル列と対称ラベル列のみを生成し、逆ラベル列を全て生成することのないようにしたい。そのための準備として、アルゴリズム **generate-all-caterpillars** が生成するラベル列のリストについて考えよう。

いま、正ラベル列または対称ラベル列 $L = (c_1, c_2, \dots, c_{d-1})$ を出力したとしよう。この後、連続して逆ラベル列が生成される限り、これらは出力されずに捨てられる。次に、出力すべきラベル列 L_{next} は何であろうか?

まず、次がいえる。もし $c_1 = 0$ ならば、その後に

生成するすべての正ラベル列において $c'_{d-1} = 0$ である。よって、この問題は、最上位と最下位の二つの桁を取り除き、 $(d - 1) - 2$ 桁の $n - d$ 進数と見なせるラベル列で、各桁の値の和が $n - d - 1$ であるものをすべて降順に列挙する問題に帰着する。

以降も同様に、最上位の桁の値が 0 になれば、その後に生成するすべての正ラベル列において最下位の桁の値は 0 である。よって、桁数が 2 だけ小さい列挙問題に帰着する。アルゴリズムにこの工夫を加えることにより、例えば、 $n = 12, d = 8$ のときは次のようなラベル列のリストが得られる。このリスト中、下線があるものは逆ラベル列を表している。

- | | | | |
|-------------------------|---|-------------------------|---|
| $(3, 0, 0, 0, 0, 0, 0)$ | \rightarrow | $(2, 1, 0, 0, 0, 0, 0)$ | |
| \rightarrow | $(2, 0, 1, 0, 0, 0, 0)$ | \rightarrow | $(2, 0, 0, 1, 0, 0, 0)$ |
| \rightarrow | $(2, 0, 0, 0, 1, 0, 0)$ | \rightarrow | $(2, 0, 0, 0, 0, 1, 0)$ |
| \rightarrow | $(2, 0, 0, 0, 0, 0, 1)$ | \rightarrow | $(1, 2, 0, 0, 0, 0, 0)$ |
| \rightarrow | $(1, 1, 1, 0, 0, 0, 0)$ | \rightarrow | $(1, 1, 0, 1, 0, 0, 0)$ |
| \rightarrow | $(1, 1, 0, 0, 1, 0, 0)$ | \rightarrow | $(1, 1, 0, 0, 0, 1, 0)$ |
| \rightarrow | $(1, 1, 0, 0, 0, 0, 1)$ | \rightarrow | $(1, 0, 2, 0, 0, 0, 0)$ |
| \rightarrow | $(1, 0, 1, 1, 0, 0, 0)$ | \rightarrow | $(1, 0, 1, 0, 1, 0, 0)$ |
| \rightarrow | $(1, 0, 1, 0, 0, 1, 0)$ | \rightarrow | $(1, 0, 1, 0, 0, 0, 1)$ |
| \rightarrow | $(1, 0, 0, 2, 0, 0, 0)$ | \rightarrow | $(1, 0, 0, 1, 1, 0, 0)$ |
| \rightarrow | $(1, 0, 0, 1, 0, 1, 0)$ | \rightarrow | $(1, 0, 0, 1, 0, 0, 1)$ |
| \rightarrow | $(1, 0, 0, 0, 2, 0, 0)$ | \rightarrow | $(1, 0, 0, 0, 1, 1, 0)$ |
| \rightarrow | <u>$(1, 0, 0, 0, 1, 0, 1)$</u> | \rightarrow | <u>$(1, 0, 0, 0, 0, 1, 1)$</u> |
| \rightarrow | <u>$(1, 0, 0, 0, 0, 0, 2)$</u> | \rightarrow | $(-, 3, 0, 0, 0, 0, -)$ |
| \rightarrow | $(-, 2, 1, 0, 0, 0, -)$ | \rightarrow | $(-, 2, 0, 1, 0, 0, -)$ |
| \rightarrow | $(-, 2, 0, 0, 1, 0, -)$ | \rightarrow | $(-, 2, 0, 0, 0, 1, -)$ |
| \rightarrow | $(-, 1, 2, 0, 0, 0, -)$ | \rightarrow | $(-, 1, 1, 1, 0, 0, -)$ |
| \rightarrow | $(-, 1, 1, 0, 1, 0, -)$ | \rightarrow | $(-, 1, 1, 0, 0, 1, -)$ |
| \rightarrow | $(-, 1, 0, 2, 0, 0, -)$ | \rightarrow | $(-, 1, 0, 1, 1, 0, -)$ |
| \rightarrow | $(-, 1, 0, 1, 0, 1, -)$ | \rightarrow | <u>$(-, 1, 0, 0, 1, 1, -)$</u> |
| \rightarrow | <u>$(-, 1, 0, 0, 0, 2, -)$</u> | \rightarrow | $(-, -, 3, 0, 0, -, -)$ |
| \rightarrow | $(-, -, 2, 1, 0, -, -)$ | \rightarrow | $(-, -, 2, 0, 1, -, -)$ |
| \rightarrow | $(-, -, 1, 2, 0, -, -)$ | \rightarrow | $(-, -, 1, 1, 1, -, -)$ |
| \rightarrow | <u>$(-, -, 1, 0, 2, -, -)$</u> | \rightarrow | $(-, -, -, 3, -, -, -)$ |

今後は、前章のアルゴリズムに上の工夫を行なったことにしよう。

仮定より、正ラベル列または対称ラベル列 $L = (c_1, c_2, \dots, c_{d-1})$ を出力したとしよう。上のアルゴリズムに基づき L の次に生成するラベルを $L' = (c'_1, c'_2, \dots, c'_{d-1})$ とする。もし L' が逆ラベル列である場合は連続する逆ラベル列をすべてスキップし、次に出力すべきラベル列を求めなくてはならない。 L が与えられたとき、次に出力すべきラベル列 L_{next} は、次のようにして決定できる。次の三つの

場合にわけて考えよう。

場合 1: $c'_1 = c'_{d-1}$ のとき。

さらに三つの場合にわけて考えよう。

場合 1(a): L' は対称ラベル列であるとき。

もし L' が高々二つのラベルのみからなるならば、 L' を出力し、アルゴリズムを終了する。

そうでないときは $L_{next} = L'$ となる。 (場合 1(a) 終)

以下、場合 1(a) でなかったとしよう。 $c'_j \neq c'_{d-j}$ なる最小の j を選ぶ。

場合 1(b): $c'_j < c'_{d-j}$ のとき。 (すなわち L' は逆ラベル列である。)

もし $j = 2$ ならば、 $1 \leq s \leq d-3$ なる s のうち、 $c'_s > 0$ なる最大の s を選ぶ。

c'_s の値を 1 減らし、 c'_{s+1} の値を $c'_{d-2} + c'_{d-1} + 1$ とし、 c'_{d-2} の値を 0 とし、 c'_{d-1} の値を 0 とする。 (仮定より最上位桁の値は 0 ではないが、最下位の値は必ず 0 になることに注意されたい。すなわち正ラベル列である。)

もし $j > 2$ かつ $c'_{d-j} > 0$ ならば、 c'_s の値を 1 減らし、 c'_{s+1} の値を $c'_{s+1} + c'_{s+2} + \dots + c'_{d-1} + 1$ とし、 $c'_{s+2} = c'_{s+3} = \dots = c'_{d-1} = 0$ とする。 (仮定より最上位桁の値は 0 ではないが、最下位の値は必ず 0 になることに注意されたい。)

もし $j > 2$ かつ $c'_{d-j} = 0$ ならば、 $c'_j < c'_{d-j}$ に矛盾する。すなわち、このような場合はない。

(場合 1(b) 終)

場合 1(c): $c'_j > c'_{d-j}$ のとき。 (すなわち L' は正ラベル列である。)

$L_{next} = L'$ となる。 (場合 1(c) 終)

(場合 1 終)

場合 2: $c'_1 > c'_{d-1}$ のとき。

$L_{next} = L'$ となる。 (場合 2 終)

場合 3: $c'_1 < c'_{d-1}$ のとき。

L が正ラベル列または、対称ラベル列より、 $c_1 = c_{d-1} = c'_{d-1} - 1$ 、かつ、 $0 < c_{d-2} = c'_{d-2} + 1$ である。 $c_j \neq c_{d-j}$ なる最小の j が存在するならば、その j を選ぶ。このとき L は正ラベル列となり、 $c_j > c_{d-j}$ である。この場合は L から L_{next} を次のように構成する。

もし $j \geq 3$ ならば、 $1 \leq k \leq d-j$ なる k のうち、 $c_k \neq 0$ を満たす最大の k を選ぶ。

c_k の値を 1 減らし、 c_{k+1} の値を $c_{k+1} + c_{k+2} + \dots + c_{d-1} + 1$ とし、 $c_{k+2} = c_{k+3} = \dots = c_{d-1} = 0$ とする。 (仮定より最上位桁の値は 0 ではないが、最下位の値は必ず 0 になることに注意されたい。)

もし $j = 2$ ならば $1 \leq k \leq d-3$ なる k のうち、 $c_k \neq 0$ を満たす最大の k を選ぶ。

c_k の値を 1 減らし、 c_{k+1} の値を $c_{k+1} + c_{k+2} + \dots + c_{d-1} + 1$ とし、 $c_{k+2} = c_{k+3} = \dots = c_{d-1} = 0$ と

する。

$c_j \neq c_{d-j}$ なる最小の j が存在しないならば、 L は対称ラベル列となる。このとき、 $1 \leq k \leq d-3$ なる k のうち、 $c_k \neq 0$ を満たす最大の k を選ぶ。

c_k の値を 1 減らし、 c_{k+1} の値を $c_{d-2} + c_{d-1} + 1$ とし、 $c_{k+2} = c_{k+3} = \dots = c_{d-1} = 0$ とする。

(場合 3 終)

補題 4.2 上記の工夫により、 $O(d)$ 時間の前処理の後、 $O(1)$ 時間ごとに 1 つの caterpillar を生成するアルゴリズムが得られる。作業用計算領域は、 $O(d)$ である。

5 様々なクラスの木の列挙

前章のアルゴリズムを各々の直径の値 $k = 2, 3, \dots, n-1$ に対して実行することにより、頂点の個数が n である caterpillar を 1 個あたり $O(1)$ 時間で列挙するアルゴリズムが得られる。

平面埋め込みを固定したグラフを平面グラフという。同一のグラフも、幾つかの平面グラフを持つことがある。例えば、図 4 に、同じ caterpillar に対応する様々な平面 caterpillar を示す。

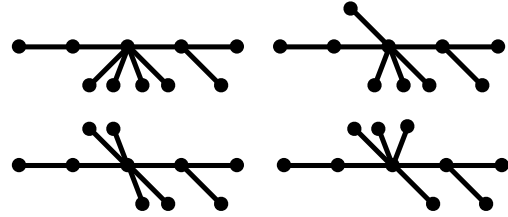


図 4: 様々な平面 caterpillar

平面 caterpillar の列挙も次のようにして可能である。

点の個数が n であり、直径が d であるような平面 caterpillar が与えられたとしよう。任意の長さ d のパスを $P_d = (u_0, u_1, \dots, u_d)$ とする。

$2 \leq i \leq d-2$ なる各点 u_i で、接続する点を時計回りに並べたものを $(u_{i-1}, v_1, v_2, \dots, v_{a(i)}, u_{i+1}, w_1, w_2, \dots, w_{b(i)})$ とする。ここで、 $v_1, v_2, \dots, v_{a(i)}$ や $w_1, w_2, \dots, w_{b(i)}$ は、点 u_i に隣接する各 cocoon を示している。また、 u_1 に接続する foot の個数を $c(1)$ とし、 u_{d-1} に接続する foot の個数を $c(d-1)$ としよう。

以上より、各平面 caterpillar に対し、 P_d を選ぶと、ラベル列 $(c(1), a(2), a(3), \dots, a(d-2), c(d-1), b(2), b(3), \dots, b(d-2))$ が定義できる。 P_d の選び方は本質的に 2 種類だけであり、かつ、これらは互いに点列を逆順にしたものに対応する。よって、各平面 caterpillar に対して高々 2 種類のラベル列しか得られない。これらは、 $2 + (d-3)2$ 桁の $n-d$ 進

数で各桁の値の和が $n - d - 1$ であるものに相当する。よって、これまでに述べたアルゴリズムと同様の手法で、平面 caterpillar を 1 個あたり $O(1)$ 時間で列挙することができる。

6 spider, scorpion の列挙アルゴリズム

caterpillar の列挙アルゴリズムと同様の手法により、頂点の個数が n である spider や、頂点の個数が n である scorpion を列挙するアルゴリズムも次のようにして得られる。以下に、概要のみ説明する。

まず、各 spider に対し、ラベル列を定義する。spider の body を v とし、その次数を $d(v)$ とする。spider の持つ各 leg (パス) を、 $L_1, L_2, \dots, L_{d(v)}$ としよう。各 L_i に含まれる辺の本数を $l(L_i)$ で表す。このとき、 $(l(L_1) - 1, l(L_2) - 1, \dots, l(L_{d(v)}) - 1)$ を spider に対応するラベル列としよう。例えば、図 2 の spider に対しては、六つのラベル $0, 0, 1, 1, 2, 2$ を任意の順に並べたラベル列である $(1, 0, 2, 1, 2, 0)$, $(2, 2, 1, 1, 0, 0)$, $(2, 1, 2, 1, 0, 0)$ 等となる。これらは、 $d(v)$ 桁の $n - d(v)$ 進数とみなせるラベル列で、各桁の値の和が $n - d(v) - 1$ であるものに対応する。

各 spider にユニークなラベル列を定義するために、spider から得られるラベル列のうち、 $d(v)$ 桁の $n - d(v)$ 進数とみたときに最大のものを選ぶことにする。例えば、図 2 の spider に対しては、 $(2, 2, 1, 1, 0, 0)$ となる。

これらのラベル列を降順に生成することにより、頂点の個数が n であり、body の次数が $d(v)$ であるような spider を列挙するアルゴリズムが得られる。

特に、ちょうど 1 つの leg を除き、全ての leg (パス) の長さが 2 以下であるような spider は、scorpion と呼ばれる。spider と同様の方法により、scorpion を列挙するアルゴリズムも得られる。

7 まとめと今後の課題

本論文では、頂点の個数が n であり、直径が d であるすべての caterpillar を、重複なく、かつ、もれなく高速に列挙するアルゴリズムを与えた。本論文のアルゴリズムは、そのような caterpillar を 1 個あたり、定数時間で生成する。また、様々なクラスの木を列挙するアルゴリズムを与えた。

直径が k の木を高速に列挙するアルゴリズムの開発が、今後の課題である。

参考文献

[1] Beyer, T. and Hedetniemi, S. M., Constant Time Generation of Rooted Trees, *SIAM J. Comput.*, 9 (1980) 706–712.

[2] Biggs, N. L., Lloyd, E. K., Willson, R. J., *Graph Theory 1736-1936*, Clarendon Press, Oxford, (1976)

[3] Chauve, C., Dulucq, S. and Rechnitzer, A., Enumerating Alternating Trees, *Journal of Combinatorial Theory, Series A*, 94 (2001) 142–151.

[4] Chung, K.-L. and Yan, W.-M., On the Number of Spanning Trees of a Multi-Complete/Star Related Graph, *Information Processing Letters*, 76 (2000) 113–119.

[5] Li, Z. and Nakano, S., Listing All Connected Plane Triangulations, *Proc. of Canadian Conference on Computational Geometry* (2001) 121–124.

[6] Lonc, Z., Parol, K. and Wojciechowski, J. M., On the Number of Spanning Trees in Directed Circulant Graphs, *Networks*, 37 (2001) 129–133.

[7] Harary, F. and Schwenk, A. J., The Number of Caterpillars, *Discrete Mathematics*, 6 (1973) 359–365.

[8] Hasunuma, T. and Shibata, Y., Counting Small Cycles in Generalized de Bruijn Digraphs, *Networks*, 29 (1997) 39–47.

[9] Nakano, S., Efficient Generation of Plane Trees, *Information Processing Letters*, 84 (2002) 167–172.

[10] Simone, C. D., Lucertini, M., Pallottino, S. and Simeone, B., Fair Dissections of Spiders, Worms, and Caterpillars, *Networks*, 20 (1990) 323–344.