アルゴリズム 28−11
（1992. 7. 18）

正則グラフ生成アルゴリズムのランダム性について

前川 浩二　松田 洋一†　榎原 博之　中野 秀男

大阪大学工学部 通信工学科
†NTT

頂点数nと次数dが与えられたとき、任意のd次正則グラフを生成するアルゴリズムが提案されている[4]。このアルゴリズムは、与えられた頂点数と次数を持つ全ての正則グラフの中の任意のグラフを生成できることは保証しているが、生成されるグラフの発生がランダムであるかどうかは保証していない。そこで本稿では、このアルゴリズムにより生成される正則グラフとランダム正則グラフの生成法により生成される正則グラフの発生確率を比較し、このアルゴリズムがランダムに正則グラフを生成するかどうかを計算機実験を通して検証する。その結果、このアルゴリズムが生成する正則グラフの発生確率は、十分にランダムであることが確認できた。

On Randomness in the Algorithm for Generating Regular Graphs

Koji Maekawa　Yoichi Matsuda†　Hiroyuki Ebara　Hideo Nakano

Department of Communication Engineering, Faculty of Engineering, Osaka University
†NTT

Given the number of vertices n and degree d, we proposed an algorithm which generated an arbitary d-regular graphs. We established that it was possible for this algorithm to generate an arbitary graphs out of all such regular graphs. We have, however, no verification of random occurrence of the generated graphs. In this paper, we verify the randomness in the algorithm, by comparing the occurence probability of the regular graphs generated by this algorithm with that of the graphs by the algorithm which generates random regular graphs, through our computational experiment. As a result, the randomness in our algorithm is certificated.

1 はじめに

グラフの同形判定問題は、計算量理論の分野においてNP完全であるかどうか未解決であるが、多項式時間では解き難い問題である。さらにこの問題は、その解き難さから、暗号理論においては認証の一方式である零知識対話証明に応用されている。グラフの同形判定の中でも、頂点の次数が全て等しい正則グラフに対する同形判定問題は、もっとも難しい問題の1つであると考えられている。このことから、正則グラフの同形判定問題に対する解法の実験的評価、さらには暗号理論への応用のために、任意の頂点数と次数を持つ正則グラフを効率良く生成する方法が必要とされている。

確率的グラフ理論の立場からは、ランダム正則グラフを生成するアルゴリズムがいくつか提案されている [1][2][3]。しかし、これらのアルゴリズムは、生成できる正則グラフの次数に制限がある。つまり、頂点数を n としたとき、[1] では $\sqrt{n/2}$ まで、[2] では $O(\sqrt{\log n})$ まで、[3] では $O(n^{1/3})$ までの次数しか効率良く生成することができない。

それに対し、頂点数及び次数が与えられたとき、任意の正則グラフを生成するアルゴリズムが著者らによって提案されている [4]。しかし、このアルゴリズムは生成される正則グラフの発生がランダムであるかどうかについては保証していない。ここで、応用面まで考えた場合には‘ランダム性’の定義すら一意的なものではないため、ランダム性の評価を絶対的に行なうことは困難なこととなっている。また、仮にそれが定義されたとしても、ランダム性を理論的に評価することは非常に難しいことである。さらに、このランダム性を実験的に評価する場合においても、有効なアルゴリズムが存在しないため、頂点数が大きくなると計算時間が指数関数的に長くなり、有効に評価を行なうことが困難になる。

そこで本稿では、頂点数の比較的小さい場合に限って、このアルゴリズムによって生成される正則グラフの発生のランダム性を、ランダム正則グラフとの計算機実験による比較を通して検証する。

2 正則グラフの生成方法

ここでは、[4] で提案されている正則グラフの生成方法について述べる。この生成方法は、クリーク挿入法、頂点挿入法、枝交換法の 3 つより構成され、頂点数と次数が与えられたとき、頂点数 n の d 次正則グラフ全ての中から、任意のグラフを生成するものである。

まず、正則グラフを生成する手順の概略を以下に示す。

(1) k個 (kは与えられるものとする) のクリーク K_{d+1} を初期状態のグラフとする。ここで、K_iは頂点数 i の完全グラフである。

(2) 頂点数 n に対して

$$n = k(d+1) + l(d-1) + m \qquad (1)$$

が成り立つようにクリーク挿入法の実行回数 l、頂点挿入法によって増加させる頂点数 m を決める。このときクリーク挿入法の実行回数が多くなるように m を

$$m = \big(n - k(d+1)\big) \mod (d-1) \qquad (2)$$

と定める。

(3) (2) で定めた実行回数でクリーク挿入法と頂点挿入法を実行し、頂点数 n の d 次正則グラフを生成する。

(4) (3) で生成した正則グラフを枝交換法を $e/2$ 回 (e は正則グラフ内の枝の数) 実行することにより、同じ頂点数と次数を持つ任意の正則グラフに変換する。

次に、クリーク挿入法、頂点挿入法および枝交換法の具体的な振舞いを示す。

- クリーク挿入法

  ランダムに $(d-1)$ 本の枝を選び、これらを $e_1 = (x_1, y_1)$, $e_2 = (x_2, y_2)$, $\cdots$, $e_{d-1} = (x_{d-1}, y_{d-1})$ とする。そして、各々の枝の中間に図 1 のように新たに頂点を作り、それらの頂点の間でクリーク K_{d-1} を構成するように枝を接続する。

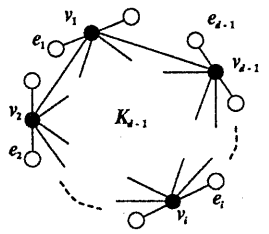

図 1: クリーク挿入法

- 頂点挿入法

クリーク挿入法では、頂点数が $(d-1)$ ずつ増加するため、これだけでは任意の頂点数を得ることができるとは限らない。また、頂点の増加数の最小値は d が偶数ならば 1、奇数ならば 2 である。そこで、これらの最小増加数を実現する生成方法を提案する。

〈 d が偶数のとき 〉

ランダムに $d/2$ 本の互いに隣接しない枝を選び、これらを $e_1 = (x_1, y_1)$, $e_2 = (x_2, y_2)$, $\cdots$, $e_{d/2} = (x_{d/2}, y_{d/2})$ とする。次に、これらの枝の端点 x_i, y_i $(1 \leq i \leq d/2)$ を挿入すべき頂点 v と接続する。最後に、e_1, e_2, $\cdots$, $e_{d/2}$ を除去する。(図 2)

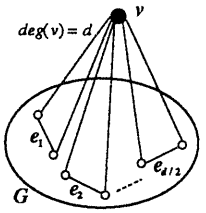

図 2: 頂点挿入法 (d : 偶数)

〈 d が奇数のとき 〉

ランダムに長さ d の点素な道を選び、これを $path = \{e_1 = (x_0, x_1),\ e_2 = (x_1, x_2),\ \cdots,\ e_d = (x_{d-1}, x_d)\}$ とする。次に、挿入すべき頂点 v_1, v_2 について、v_1 は x_0, x_1, $\cdots$, x_{d-1} と、v_2 は x_1, x_2, $\cdots$, x_d と図 3 のように接続する。最後に、$path$ を除去する。

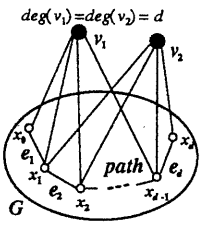

図 3: 頂点挿入法 (d : 奇数)

- 枝交換法

枝交換法の説明をする前に、ここで使用する用語について定義しておく。

〈 交互道 〉

対象としているグラフに含まれる枝 α と含まれない枝 β を、重複を許さずに連続性を保って、

$$(\alpha_1, \beta_1)(\alpha_2, \beta_2)\ldots(\alpha_n, \beta_n)$$
$$(\text{ただし、} \alpha_i \neq \alpha_j, \beta_i \neq \beta_j)$$

のように選んだもの。

長さ 2 の交互道から始め、再帰的にそれを延長していく。そして、図 4 のように交互道が閉じたとき (これを交互閉路とする)、この交互閉路において枝 α と枝 β とを交換する。

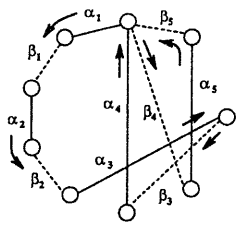

図 4: 枝交換法

3 ランダム正則グラフの生成方法

ここでは、与えられたアルゴリズムがランダムに正則グラフを生成するかどうかの検証を行なう際に、比較のために使用するランダム正則グラフの生成方法を示す。

そのためにまず、ランダムグラフについて説明する。

頂点数 n が与えられたときに、全ての頂点に対して接続可能な枝の総数 e_0 は、

$$e_0 = \frac{n(n-1)}{2} \qquad (\text{本})$$

である。これらの枝が全て等しい確率

$$\frac{1}{e_0} = \frac{1}{\frac{n(n-1)}{2}}$$

で発生するときに、異なる任意の e_1 $(0 \leq e_1 \leq e_0)$ 本の枝によって生成されるグラフをランダムグラ

フという。また、このときの各々の枝の発生確率は

$$\frac{e_1}{e_0} = \frac{e_1}{\frac{n(n-1)}{2}}$$

である。

このことから、指定された頂点数 n と次数 d をもつランダム正則グラフを生成するには、枝の数 e が全部で、

$$e = \frac{dn}{2} \qquad (本)$$

であることから、各々の枝の発生確率を

$$\frac{e}{e_0} = \frac{e}{\frac{n(n-1)}{2}} = \frac{d}{n-1}$$

としてグラフを生成し、生成されたグラフのうち正則グラフになるものだけを取り出すようにすればよいことになる。

このようにして生成されたグラフは、その生成方法により、他のグラフに比べてランダム性に優れており、発生確率は個々のグラフに関して同様に確からしいと思われる。しかし一方では、このようなランダムグラフが正則グラフになる確率は、頂点数 n が大きくなると急激に小さくなるので、n が大きくなると非常に計算時間が長くなるという欠点がある。実際の実験では、$n = 20$, $d = 4$ のときに 6000 分を越えても正則なランダムグラフは得られなかった。

4 計算機実験

4.1 同形判定の方法

前節までに示した方法によって生成された正則グラフが同形であるかどうかを判定するために、以下に示すような同形写像の全数探索法を用いる。

同形判定のプログラムでは、判定を行なう正則グラフの情報を隣接行列に収めている。そこで同形判定のプログラムでは、生成された正則グラフの情報を読みとるとそれを隣接行列へと変換する。また、判定に用いる既知のグラフの情報も隣接行列へと変換して確保する。これらの変換が終了したら、まず入力された二つの正則グラフの頂点数と次数をそれぞれ比較し、それらが共に一致していれば次のステップへと進み、次に、入力された正則グラフの頂点数 n にしたがって、1 から n までの数字の順列を生成して、その順列にしたがって判定を行なうグラフの隣接行列の行と列を入れ換えて同形写像変換を行なう。この隣接行列と既知の正則グラフの隣接行列の各要素について排他的論理和をとって、二つのグラフが同形であるか否かを判定する。判定の結果が非同形のときには、もし可能な限りの全ての順列を生成してしまっているのであれば“非同形”の文字を出力してプログラムを終了し、そうでなければ次の新しい順列を生成し同形判定を行なうということを繰り返す。

以上の流れを図5に示す。

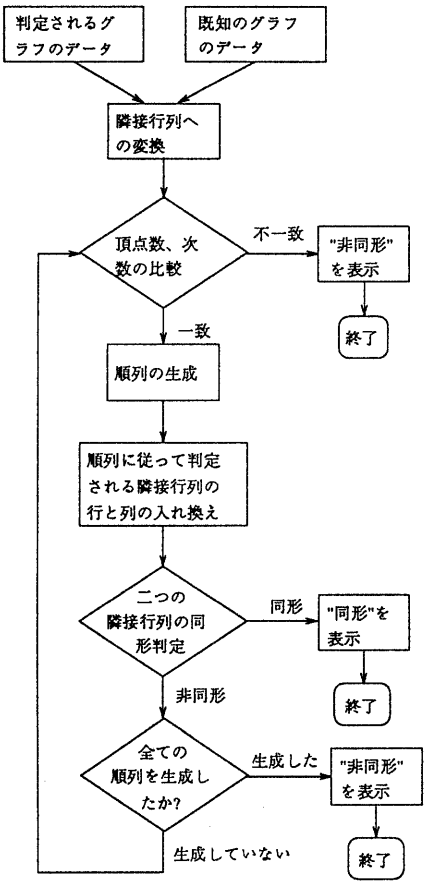

図 5: 全数探索法を用いた同形判定の流れ図

4.2 実験結果

4.1 に示した同形判定の方法を用いて、3 までに述べた生成方法によって得られた正則グラフとあらかじめ用意しておいた既知のグラフとを同形判定した結果を以下に示す。ここで、試行回数は 1500 回であり、2 で述べた生成方法を‘ 生成方法 1 ’とし、3 で述べたランダム正則グラフの生成方法を‘ 生成方法 2 ’とする。また、既知のグラフとしては、あらかじめ生成方法 1 によってサンプリングして得られたも

のを用いており、それを $\mathrm{graph}(n-d)-x$ のように表すことにする。ただし、言語は C 言語を、計算機としては SONY NEWS : NWS-3870 を使用する。

4.2.1 頂点数= 6 , 次数= 3 の場合

頂点数が 6 で次数が 3 の正則グラフは、上記の 2 種類の生成方法によって図 6 に示すような 2 種類のグラフが生成された。各々の生成方法における発生比率を表 1に示す。

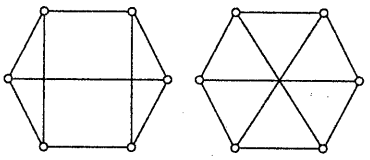

図 6: 頂点数= 6 , 次数= 3 のグラフ

表 1: 正則グラフ $(n=6, d=3)$ の発生比率

	発生比率 (%)	
	(a)	(b)
生成方法 1	85.6	14.4
生成方法 2	84.9	15.1

(ただし、(a),(b) は図 6 に示したもの)

4.2.2 頂点数= 7 , 次数= 4 の場合

頂点数が 7 で次数が 4 の正則グラフは、2 種類の生成方法によって図7に示すような 2 種類のグラフが生成された。各々の生成方法における発生確率を表2に示す。

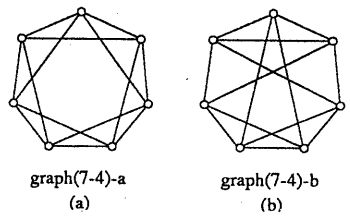

図 7: 頂点数= 7 , 次数= 4 のグラフ

表 2: 正則グラフ $(n=7, d=4)$ の発生比率

	発生比率 (%)	
	(a)	(b)
生成方法 1	81.8	18.2
生成方法 2	78.6	21.4

(ただし、(a),(b) は図 7 に示したもの)

4.2.3 頂点数= 8 , 次数= 3 の場合

頂点数が 8 で次数が 3 の正則グラフは、2 種類の生成方法によって図 8 に示すような 6 種類のグラフが生成された。各々の生成方法における発生確率を表3に示す。ただし、生成方法 1 の場合には用意するクリーク K_4の数 k が、$k=1$ と $k=2$ の 2 通りの場合があるので、それぞれを ' 生成方法 $1_{(1)}$ ', ' 生成方法 $1_{(2)}$ ' と表している。

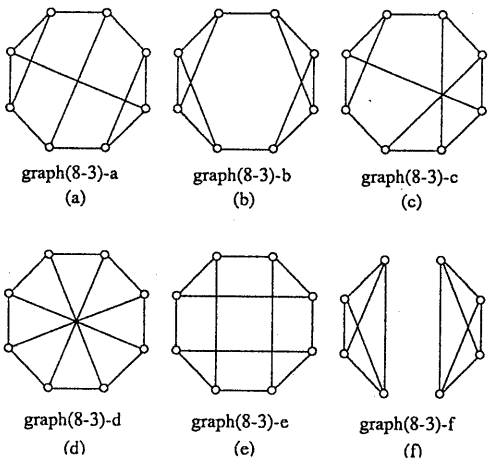

図 8: 頂点数= 8 , 次数= 3 のグラフ

表 3: 正則グラフ $(n=8, d=3)$ の発生確率

	発生比率 (%)					
	(a)	(b)	(c)	(d)	(e)	(f)
生成方法 $1_{(1)}$	50.3	11.2	18.8	15.7	3.9	0.1
生成方法 $1_{(2)}$	51.8	11.7	17.9	14.9	3.7	0.1
生成方法 2	51.6	12.4	18.5	13.3	3.9	0.3

(ただし、(a),(b),(c),(d),(e),(f) は図 8 に示したもの)

4.2.4 頂点数= 8 , 次数= 4 の場合

頂点数が 8 で次数が 4 の正則グラフは、2 種類の生成方法によって図 9 に示すような 6 種類のグラフが得られた。各々の生成方法における発生確率を表 4 に示す。

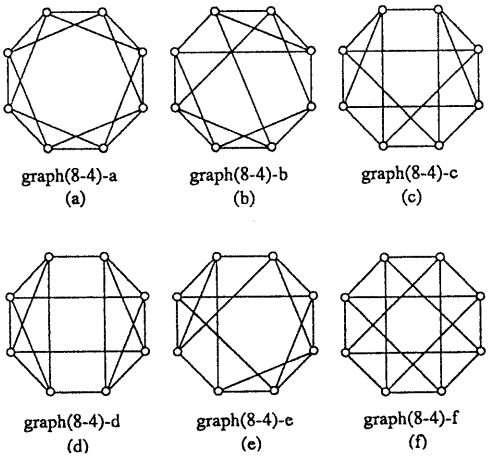

図 9: 頂点数= 8 , 次数= 4 のグラフ

表 4: 正則グラフ $(n = 8, d = 4)$ の発生確率

	発生比率 (%)					
	(a)	(b)	(c)	(d)	(e)	(f)
生成方法 1	13.7	52.0	11.8	5.0	17.5	0.1
生成方法 2	12.1	54.2	11.9	4.1	17.5	0.3

(ただし、(a),(b),(c),(d),(e),(f) は図 9 に示したもの)

4.2.5 頂点数= 8 , 次数= 5 の場合

頂点数が 8 で次数が 5 の正則グラフは、2 種類の生成方法によって図 10に示すような 3 種類のグラフが得られた。各々の生成方法における発生確率を表 5 に示す。

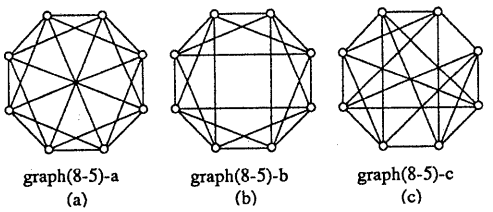

図 10: 頂点数= 8 , 次数= 5 のグラフ

表 5: 正則グラフ $(n = 8, d = 5)$ の発生確率

	発生比率 (%)		
	(a)	(b)	(c)
生成方法 1	74.5	7.7	17.8
生成方法 2	71.3	9.9	18.8

(ただし、(a),(b),(c) は図 10 に示したもの)

4.3 考察

4.3.1 生成方法のランダム性

生成方法 1 によって正則グラフを生成すれば、任意の正則グラフが生成することができるということは、既に [4] において示されているのだが、生成されるグラフの発生確率がランダムであるということは保証していない。そこで、生成方法 1 によって生成される正則グラフを中心に、そのランダム性について実験結果をもとに考察する。

‘ ランダムである ’ という定義に対して、

- どの任意のグラフも、同様な確からしさで発生する。
- どの同形グループのグラフも、同様な確からしさで発生する。

という 2 通りの解釈ができる。したがって、ランダム性を評価するに当たっては、上記のどちらの定義に対して‘ ランダムである ’とするのかを決めておく必要がある。本考察においては、ランダムグラフの定義に合わせて前者の場合をもって‘ ランダムである ’とする。ランダム性をこのように定義した場合、各々の同形グループの正則グラフが発生する確率は、各同形グループに属する同形写像の数に比例すると考えられる。またこの場合には、本実験に用いた生成方法 2 は非常にランダム性に優れているといえる。そこで、生成方法 2 によって得られた正則グラフの発生確率を、今後、ランダムな発生確率の基準と考える。

表 1 ～ 表 5 において、生成方法 1 と生成方法 2 の結果を比較してみると、発生確率の高いグラフと発生確率の低いグラフの発生状況は、おおむね一致している。したがって、生成方法 1 はランダムな生成を行なっているといえる。

5 結論

本稿では、任意の正則グラフを生成できるとして提案されている、正則グラフの生成アルゴリズムを実現して、そのランダム性について考察を行なった。

まず、提案されているアルゴリズム (生成方法 1) を C 言語で実現し、その他に、このアルゴリズムと比較させるために、与えられた頂点数と次数を持ち、枝の発生がランダムな正則グラフを生成するアルゴリズム (生成方法 2) を提案し、C 言語で実現した。その後、各々の生成方法によって生成される正則グラフの同形判定を行なうために、全数探索法による同形判定法を C 言語で実現した。

以上のものを用いて、正則グラフを生成し、その同形判定を行なうことによって各同形グループの正則グラフの発生確率を調べ、それをもとに、生成方法 1 のランダム性について考察を行なった。

生成方法 1 にしたがって生成される正則グラフの発生確率は、全体的に見ると、生成方法 2 にしたがって生成されるランダムな正則グラフの発生確率と、ほぼ同じような傾向を示した。この点から考えると、生成方法 1 のランダム性はかなり優れていると考える。

また、正則グラフを図 6〜図 10 のように平面上に表したとき、グラフ内の対称点と対称線の総和 (これを対称数と呼ぶことにする) の大きいものは発生確率が低くなるという傾向が見られた。顕著な例としては、頂点数=8, 次数=4 のとき、対称数が 4 である graph(8-4)-f の発生確率と対称数が 9 である graph(8-4)-a の発生確率がそれぞれ 0.3%, 12.1% と低いのに対し、対称数が 0 である graph(8-4)-b の発生確率が 54.2% と高くなっているのがあげられる (図 11参照)。しかしながら、この傾向を裏付ける確かな論証は得られていない。

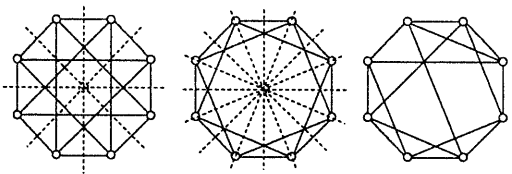

図 11: 頂点数= 8 , 次数= 4 のときの比較

本稿では、頂点数が 6 , 7 , 8 の正則グラフについて 1500 個のサンプルを生成して検証を行ない、提案されているアルゴリズムがランダム性に優れているという結果を得た。これは、頂点数がより大きい場合にも成り立つであろうと推測される。今後の課題としては、より大きな頂点数についての実験を行なうことが必要である。このためには、生成方法 2 によるランダム正則グラフの生成アルゴリズムと、全数探索法による同形判定のアルゴリズムを、もっと効率の良いものに改良する必要がある。

参考文献

[1] M.Jerrum, A.Sinclair : "Fast Uniform Generation of Regular Graphs", *Theoretical Computer Science*, Vol.73, No.1, pp.91-100 (1990)

[2] N.C.Wormald : "Generating Random Regular Graphs", *Journal of Algorithms*, Vol.5, No.2, pp.247-280 (1984)

[3] B.D.McKay, N.C.Wormald : "Uniform Generation of Random Regular Graphs of Moderate Degree", *Journal of Algorithms*, Vol.11, No.1, pp.52-67 (1990)

[4] 松田, 榎原, 中野, 堀内 : "正則グラフを生成するアルゴリズム", 情処学アルゴリズム研報, 92-AL-25-3 (1992)

[5] M.Behzad, G.Chartrand, L.Lesniak-Foster / 秋山 仁, 西関隆夫 訳 : "グラフとダイグラフの理論", 共立出版 (1981)