Dixon らによるランダムなラベルなしグラフ生成アルゴリズムの実際的な実装とその性能評価

関　建二郎† 　片山　　薫††

† 東京都立大学 工学部 電子情報工学科
†† 首都大学東京大学院 システムデザイン学科

あらまし　近年，データベースで管理される情報は，半構造データやグラフ構造をもつデータへと広がっている．グラフを対象としたシステムの性能を評価するためには，多様なグラフからなるテストデータが必要となる．そこで我々は，頂点数 n のラベルなしグラフをランダムに生成するアルゴリズムを開発した．ここでいうランダムとは，頂点数 n 互いに同型でないすべてのラベルなしグラフの集合の中から 1 つのグラフを等確率で取り出すことを意味する．Dixon らはこの目的のための効率的なアルゴリズムを示したが，彼らのアルゴリズムは計算が困難な頂点数 n の互いに同型でないラベルなしグラフの数 $\mathcal{G}_n$ を入力として与える必要がある．本研究では，完全にランダムなグラフの出力は保証できないが，$\mathcal{G}_n$ を必要としないアルゴリズムを提案する．また，そのアルゴリズムを実装し，性能評価を行った．

キーワード　ランダム，同型，ラベルなしグラフ，生成器，ジェネレータ

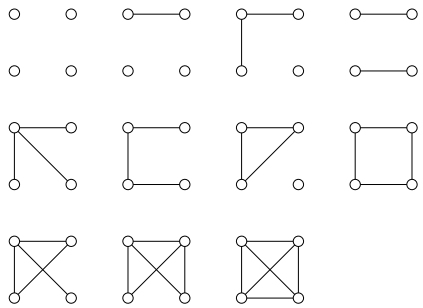

図 1　頂点数 4 の互いに同型でないラベルなしグラフの集合

表 1　頂点数 n のラベルなしグラフの数 $\mathcal{G}_n$

n	$\mathcal{G}_n$
1	1
2	2
3	4
4	11
5	34
6	156
7	1044
8	12346
9	274668
10	12005168
11	1018997864
12	165091172592
13	50502031367952

1. はじめに

近年，データベースで管理される情報は，XML などの半構造データや化学式などのグラフ構造をもつデータへと広がっている．それに応じて，このような複雑な構造をもったデータを対象としたデータマイニング (グラフマイニング) 等の研究が行われている ([11] など)．グラフを対象としたシステムの性能を評価するためには，テストデータとして，目的に応じた特徴をもつグラフ集合が必要であり，しばしば偏りのないグラフ集合が必要となる．

本研究の目的は，頂点数 n のラベルなしグラフをランダムに生成する実際的なアルゴリズムの実装である．ここでいうランダムとは，頂点数 n の互いに同型でない全てのラベルなしグラフの集合の中から 1 つのグラフを等確率で取り出すことを意味する．具体例として，図 1 に頂点数 4 の互いに同型でないラベルなしグラフの集合を示す．

例えば，頂点数 n のラベルなしグラフに対する，あるアルゴリズムの実行速度を評価することを考える．頂点数 n のラベルなしグラフの数 $\mathcal{G}_n$ は，n の増加に伴い非常に大きくなり (表 1)，n がある程度大きいと $\mathcal{G}_n$ 個のグラフの実行速度を測定することは現実的ではない．しかし，頂点数 n のラベルなしグラフをランダムに出力するジェネレータがあれば，いくつかのランダムに生成されたグラフの実行速度を測定することで，頂点数 n のラベルなしグラフに対する実行速度を推定できる．

Dixon らは，群の作用を利用した効率的なグラフ生成アルゴリズムを提案した [1]．しかし，彼らのアルゴリズムには非常に大きな計算コストを必要とする可能性があり，実際的なアルゴリズムではない．また，$\mathcal{G}_n$ を既知の定数としている．$\mathcal{G}_n$ を計算する際のコストは極めて大きく，その値は著者の知る限り 140 頂点までしか判明していない [14]．

本研究では，彼らのアルゴリズムを元に，これらの問題を改善した実際的なアルゴリズムを提案する．また，提案アルゴリズムを実装し，実行時間，ランダム性について評価を行った．

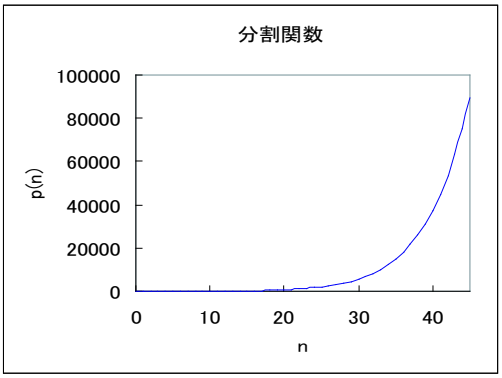

図 2　分 割 関 数

2. 関 連 研 究

本研究は，Dixon らによる群の作用を利用したランダムなラベルなしグラフ生成アルゴリズムを元にしている [1]．Wormald による [6] は本研究とは違った手法で彼らのアルゴリズムを改良している．

Kerber は群の作用の利用を，ラベルなしグラフだけではなく，ラベルのない構造に広げて，その全てを列挙する手法やランダムに選択する手法を提案した [2]．同様に Goldberg による [8] もラベルのない構造を対象にしたランダムな選択に関する研究である．Zito らはラベルのない構造をランダムに生成する並列アルゴリズムを提案している [7]．

表 1 のラベルなしグラフの数は [4] を参考にして数え上げた．

群の作用を利用したランダムなラベルなしグラフ生成には，整数の分割の生成が必要である．Zoghbi らはさまざまな整数の分割生成アルゴリズムを対象とした計算量の比較実験結果を示した [5]．

3. 準　　備

Dixon らのアルゴリズムは整数の分割に関する研究や群論のアイデアを利用している．この章では，その基本的な言葉や定義などの説明をしていく．[12] [13] などを参考にした．

3.1　整数の分割

整数の分割とは，1 つの正整数をいくつかの正整数の和に分ける仕方のことである．たとえば，4 の分割は

$4 = 4$
$4 = 3 + 1$
$4 = 2 + 2$
$4 = 2 + 1 + 1$
$4 = 1 + 1 + 1 + 1$

である．このとき，分けられた 1 つ 1 つの正整数を**和因子**という．上記の標準的な分割の表現に対して，n の分割を i の和因子 (以下 和因子 i) の数 k_i で $(k_1, k_2, \cdots, k_n)$ と表す重複度による表現がある．分割の標準的な表現と重複度による表現の対応を例で示す．

4		(0,0,0,1)
3+1		(1,0,1,0)
2+2	$\Longleftrightarrow$	(0,2,0,0)
2+1+1		(2,1,0,0)
1+1+1+1		(4,0,0,0)

n の分割の総数を表す関数を $p(n)$ と書いて**分割関数**という．分割関数は n の増加に従い，急速に大きくなる．図 2 は分割関数の値のグラフである．

3.2　群　　論

3.2.1　置　換　群

集合 $\{1, 2, \cdots, n\}$ から自分自身への全単射を n 文字の**置換**という．置換 σ を次のように表す．

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 2 & 5 & 4 \end{pmatrix}$$

さらにこれを，

$$\sigma = (132)(45)$$

とも表す．ここで例えば，(1 3 2) は 3 項**巡回置換**といい，$1 \mapsto 3 \mapsto 2 \mapsto 1$ のように巡回的に文字を写像する置換を表す．任意の置換は，互いに交わらない巡回置換の積に分解できる．そこに現れる 1 つ 1 つの巡回置換を**巡回置換成分**と呼ぶ．置換 σ が i 項巡回置換成分を k_i 個持つとき，$\langle k_1, k_2, \cdots, k_n \rangle$ を置換 σ の**巡回置換型**という．

集合 X から自分自身への全単射全体の集合を $S(X)$ と書くと，$S(X)$ は写像の合成を積として群を成す．特に，$S(\{1, 2, \cdots, n\})$ を n **次対称群**といい，S_n と書く．対称群の部分群を置換群という．図 3 は，n 次対称群の具体的な例である．

3.2.2　群 の 作 用

群 $(G, \circ)$ と集合 X に対して，写像 $\cdot : G \times X \to X$ が

（1）　全ての $g_1, g_2 \in G$ に対して，$g_1 \cdot (g_2 \cdot x) = (g_1 \circ g_2) \cdot x$

（2）　全ての $x \in X$ に対して，$e \cdot x = x$(e は G の単位元)

を満たすとき，X へ G が $\cdot$ によって**作用**しているという．

群 G が集合 X に作用しているとき，X の二つの元が G の元によって移り合うという関係を同値関係という．この同値関係による同値類を**軌道**という．また，X 中の G の軌道全体の集合を $G\backslash X$ と書いて X の G による**商空間**という．

群 G が集合 X に作用しているとき，G の元 g の作用によって，他の元に移らない X の元全体の集合を X_g と書いて g による X の**不動点集合**とよぶ．

群 G の二つの元 α, β に対して，$\alpha = g\beta g^{-1}$ を満たす $g \in G$ が存在するとき，α と β は**共役**であるという．また，共役による同値類を**共役類**という．

3.2.3　n 次対称群の共役類

n 次対称群の元の共役類は，その巡回置換型 $\langle k_1, k_2, \cdots, k_n \rangle$ で定まる (詳細は [13])．したがって，$1k_1 + 2k_2 + \cdots + nk_n = n$ に注意すると，n 次対称群の共役類は n の分割と 1 対 1 対応になることがわかる．巡回置換型 $\langle k_1, k_2, \cdots, k_n \rangle$ に定まる n 次対称群の共役類を $[(k_1, k_2, \cdots, k_n)]$ で表す．その位数は $n! / \sum_{i=1}^{n} (i^{k_i} k_i!)$ で求まる (詳細は [3])．

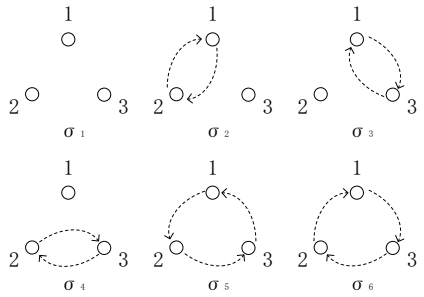

図 3　3 次対称群 S_3

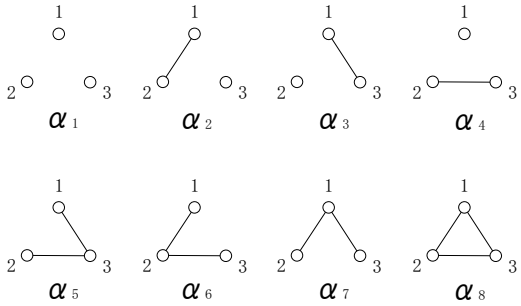

図 4　頂点数 3 のラベルつきグラフの集合 $\mathcal{M}$

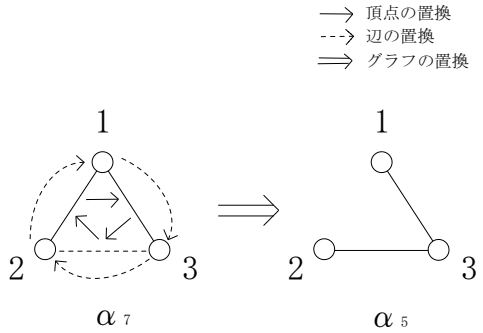

図 5　頂点の置換に誘導される辺の置換とそれに伴うグラフの置換

4.　Dixon らによるアルゴリズム

ここでは，Dixon らによるアイデアを説明していく．

4.1　軌道によるラベルなしグラフの表現

頂点数 n のラベルつきグラフの集合 $\mathcal{M}$(図 4) を考える．$\alpha \in \mathcal{M}$ の頂点を置換するとき，誘導される辺の置換 (図 5) を定義する．

［定義 1］ (**辺の置換**) $S_n^{(2)}$ を自然数 $1, 2, \cdots, n$ の対の集合上の置換群とし，置換 $\sigma \in S_n$ に誘導される置換 $\sigma^* \in S_n^{(2)}$ を $\sigma^*(\{i, j\}) = \{\sigma(i), \sigma(j)\}$ で定義する．

このとき，α にどのような $\sigma^* \in S_n^{(2)}$ が作用しても，α は α と同型なグラフにしか置換されない．また，全ての $\sigma^* \in S_n^{(2)}$ を作用させれば，α は集合 $\mathcal{M}$ 内の α と同型なグラフを全て網羅するように置換される．よって，商空間 $S_n \backslash \mathcal{M}$ は $\mathcal{M}$ を同型なグラフに分割する．

図 6 は $\mathcal{M}$ への S_n の辺の置換による作用の例である．図 4 と合わせて見ると，同型なグラフ同士の移り合いの様子がわかる．また，図 7 はその商空間を表している．

以上から，$\mathcal{M}$ へ S_n が辺の置換により作用するとき，1 つの軌道は 1 つのラベルなしグラフに対応していることがわかる．

4.2　ランダムな軌道の選択アルゴリズム

集合 X への群 G の作用に定まる軌道をランダムに選択するアルゴリズム Random Orbit を求める．次の定理 1,2 はよく知られている [1].

［定理 1］　任意の軌道 $\omega \in G\backslash X$ について次式が成り立つ．

$$|\{(g, x) \mid g \in G, x \in \omega \cap X_g\}| = |G| \qquad (1)$$

定理 1 は，全ての不動点集合 $X_g (g \in G)$ を結合したリストが与えられたとき，ある軌道に含まれる全ての元をリストの中から数え上げるとちょうど $|G|$ 個になることを示している．ここで重要なのは全ての軌道について，数え上げた値が等しいことである．つまり，このリストの中から 1 つの元をランダムに取り出すことは，商空間から 1 つの軌道をランダムに選択することに等しい．

［例 1］　頂点数 3 のラベルつきグラフの集合 $\mathcal{M}$(図 4) へ対称群 S_3(図 3) が辺の置換 (図 5) により作用するとき，不動点集合と商空間 (図 7) は次のようになる．

$\mathcal{M} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8\}$
$S_3 = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6\}$
$\sigma_1 = e$
$\sigma_2 = (23)$
$\sigma_3 = (13)$
$\sigma_4 = (12)$
$\sigma_5 = (123)$
$\sigma_6 = (132)$

不動点集合
$\mathcal{M}_{\sigma_1} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8\}$
$\mathcal{M}_{\sigma_2} = \{\alpha_1, \alpha_2, \alpha_5, \alpha_8\}$
$\mathcal{M}_{\sigma_3} = \{\alpha_1, \alpha_3, \alpha_6, \alpha_8\}$
$\mathcal{M}_{\sigma_4} = \{\alpha_1, \alpha_4, \alpha_7, \alpha_8\}$
$\mathcal{M}_{\sigma_5} = \{\alpha_1, \alpha_8\}$
$\mathcal{M}_{\sigma_6} = \{\alpha_1, \alpha_8\}$

商空間
$S_n \backslash \mathcal{M} = \{\omega_1, \omega_2, \omega_3, \omega_4\}$
$\omega_1 = \{\alpha_1\}$
$\omega_2 = \{\alpha_2, \alpha_3, \alpha_4\}$
$\omega_3 = \{\alpha_5, \alpha_6, \alpha_7\}$
$\omega_4 = \{\alpha_8\}$

このとき，ω_1 に含まれる元は全ての不動点集合に 1 つずつあり，合計は 6. ω_2 に含まれる元は $\mathcal{M}_{\sigma_1}$ に 3 つ，$\mathcal{M}_{\sigma_2}, \mathcal{M}_{\sigma_3}, \mathcal{M}_{\sigma_4}$ にそれぞれ 1 つずつあるので，合計は 6 となる．同様にして，残りの軌道についても合計は 6 となることが確認できる．つまり，$\mathcal{M}_{\sigma_1}$ から $\mathcal{M}_{\sigma_6}$ のリストから元を 1 つ取り出すことは，$S_3 \backslash \mathcal{M}$ から軌道を 1 つ取り出すことに等しい．ここまでで，ランダムに軌道を選択できる．次に，共役類を利用することでリストからの元の選択を効率化する．その際に，次の定理が有効である．

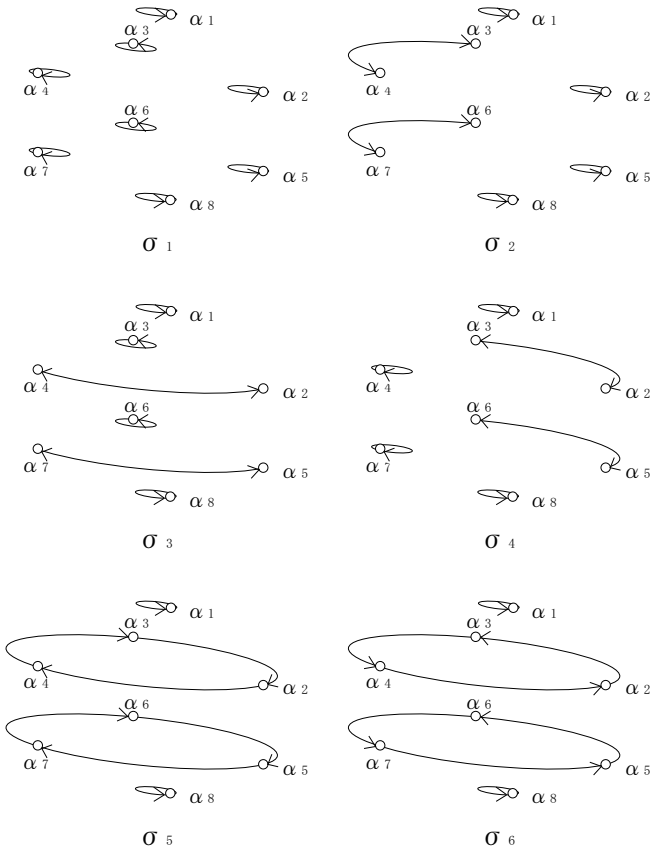

図 6　集合 $\mathcal{M}$ への対称群 S_3 の辺の置換による作用

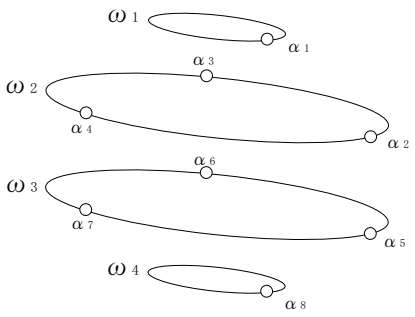

図 7　商空間 $\mathcal{M}\backslash S_3$

［定理 2］　2 つの元 $g, h \in G$ が共役であるとき，任意の軌道 $\omega \in G\backslash X$ について次式が成り立つ．

$$|X_g \cap \omega| = |X_h \cap \omega| \tag{2}$$

また，式 (2) を全ての軌道について足し上げると次式が得られる．

$$|X_g| = |X_h| \tag{3}$$

定理 2 についても，例 1 で示した不動点集合，商空間と照らし合わせればわかりやすい．

［定理 3］　不動点集合の結合リストから，元を 1 つランダムに選択する際，選択された元が共役類 C の元による不動点集合に含まれている確率で C を選択し，その後，C の任意の元の不動点集合の中から，元を 1 つランダムに選択しても，ある元が選択される確率は変わらない (証明は [1])．

［例 2］　例 1 の場合で見る．S_3 の共役類は巡回置換型で定まる．

$$C_1 = \{\sigma_1\} = \{(1)(2)(3)\} = [(3,0,0)]$$
$$C_2 = \{\sigma_2, \sigma_3, \sigma_4\} = \{(1)(23), (2)(13), (3)(12)\} = [(1,1,0)]$$
$$C_3 = \{\sigma_5, \sigma_6\} = \{(132), (123)\} = [(0,0,1)]$$

このとき，まず，C_1 を $\frac{1*8}{24}$，C_2 を $\frac{3*4}{24}$，C_3 を $\frac{2*2}{24}$ で選択する．その後，C の元を任意に選択し (それぞれ $\frac{1}{1}, \frac{1}{3}, \frac{1}{2}$)，その不動点集合の中から，ランダムに $\frac{1}{8}, \frac{1}{4}, \frac{1}{2}$ で元を選択しても，最終的な確率は $\frac{1}{24}$ である．

ここで，C の元を任意に選択しているのは，式 (2) より，C の全ての元の不動点集合について，任意の軌道に属する数が等しいからである．例えば，

$$\mathcal{M}_{\sigma_2} = \{\alpha_1, \alpha_2, \alpha_5, \alpha_8\}$$
$$\mathcal{M}_{\sigma_3} = \{\alpha_1, \alpha_3, \alpha_6, \alpha_8\}$$
$$\mathcal{M}_{\sigma_4} = \{\alpha_1, \alpha_4, \alpha_7, \alpha_8\}$$

のリストの中から元を 1 つ選択する場合と，

$$\mathcal{M}_{\sigma_2} = \{\alpha_1, \alpha_2, \alpha_5, \alpha_8\}$$

の中から元を 1 つ選択する場合とでは，ある軌道が選択される確率が等しい．

これを一般化する．群 G の共役類を $C_1, C_2, \cdots, C_m$ とするとき，C_i の**重み**を $w(C_i) = |C_i|\,|X_{g_i}|\,(g_i \in C_i)$ と定義し（このとき，式 (3) により $w(C_i)$ は g_i の選択に依らない.），C_i を選択する確率を

$$Prob(C_i) = \frac{w(C_i)}{\sum_{k=1}^{m} w(C_k)} \tag{4}$$

とする．

以上より，集合 X への群 G の作用に対して定まる軌道 $\omega \in G\backslash X$ をランダムに選択するアルゴリズム Random Orbit を導出する．

Random Orbit(X, G)

1: 確率 $Prob(C)$ で群 G の共役類 C を選択する
2: $X_g(g \in C)$ の中から元 x を 1 つランダムに選択する
3: **return**　x を元に持つ軌道 ω

4.3　ランダムなラベルなしグラフ生成アルゴリズム

頂点数 n のラベルつきグラフの集合 $\mathcal{M}$ への S_n の辺の置換による作用に定まる軌道をアルゴリズム Random Orbit で選択するアルゴリズム Random Unlabelled Graph を考える．ここで複雑なのは，共役類の重みの計算である．これは共役類に対応する n の分割の値から求まる．n の分割を $\pi = (k_1, k_2, \cdots, k_n)$ とおくと，重みの定義から次式が得られる．

$$w([\pi]) = |[\pi]|\,|\mathcal{M}_\sigma|\,, (\sigma \in [\pi]) \tag{5}$$

［定理 4］　置換 σ を巡回置換の積に分解したときの巡回置換成分の数を $q(\sigma)$ とおくと

$$|\mathcal{M}_\sigma| = 2^{q(\sigma^*)} \tag{6}$$

[証明]　$\mathcal{M}_\sigma$ は，σ^* の作用により自身に置換されるグラフ $\alpha \in \mathcal{M}$ を集めた集合である．グラフ α がこの条件を満たすためには，σ^* において，同じ巡回置換成分に属する辺が全て存在するか，全て存在しないかでなければならない．したがって，$\mathcal{M}_\sigma$ の位数は σ^* の全ての巡回置換成分から集合 $\{0,1\}$ への写像の数に等しい．

[定理 5]　置換 $\sigma \in [\pi](\pi = (k_1, k_2, \cdots, k_n))$ が誘導する辺の置換 σ^* の巡回置換成分の数 $q(\sigma^*)$ は次式で求まる (詳細は [3] など)．

$$q(\sigma^*) = \sum_i i({}_{k_i}C_2 + k_{2i} + k_{2i+1}) + \sum_{r<s} gcd(r,s)k_r k_s \quad (7)$$

ここで，$gcd(r,s)$ は r と s の最大公約数を表す．

式 (5)〜(7) より，重みが次式で求まる．

$$w([\pi]) = \frac{n!2^{q(\sigma^*)}}{\sum_{i=1}^{n}(i^{k_i}k_i!)}(\pi = (k_1, k_2, \cdots, k_n)) \quad (8)$$

頂点数 n のラベルなしグラフをランダムに選択するアルゴリズム Random Unlabelled Graph を導出する．

Random Unlabelled Graph($\mathcal{M}, n$)

Ensure: $\mathcal{M}$ = 頂点数 n のラベルつきグラフの集合
1: n の分割 π を確率 $Prob([\pi])$ で選択する
2: $\mathcal{M}_\sigma(\sigma \in [\pi])$ の中からグラフ α を 1 つランダムに選択する
3: **return** α

5. Dixon らのアルゴリズムの実装と提案手法

Random Unlabelled Graph アルゴリズムは工夫なしで実装すると，$p(n)$ 回のループ処理が必要であり，計算コストが極めて大きくなる (Simple Algorithm)．そのため，アルゴリズムを実際に使用するためには何らかの工夫が必要である．Dixon らは，非常に大きな確率で $p(n)$ 回のループ処理を $O(1)$ 回とするアルゴリズムを提案した (Dixon/Wilf Algorithm)．しかし，Dixon/Wilf Algorithm には，入力として $\mathcal{G}_n$ を必要とするという問題と非常に小さな確率だが，$p(n)$ 回のループ処理を必要とするという問題があるため，実際的ではない．我々は，彼らのアルゴリズムを改変した実際的なアルゴリズムを提案する (Practical Algorithm)．

5.1 Simple Algorithm

Random Unlabelled Graph アルゴリズムの単純な実装は以下のようになる．

Simple Algorithm($n, p(n)$)

1: n の分割 $\pi_1, \pi_2, \cdots, \pi_{p(n)}$ を全て生成する
2: **for** $i = 1$ to $p(n)$ **do**
3:　重み $w([\pi_i])$ を計算する
4:　重みの累積 $a([\pi_i]) = w([\pi_i]) + a([\pi_{i-1}])$ を計算する (初期値 $a([\pi_1]) = w([\pi_1])$)
5: **end for**
6: 1 から $a([\pi_{p(n)}])$ までの自然数 r をランダムに生成し，$a([\pi_{s-1}]) < r \leqq a([\pi_s])$ を満たす s を求める
7: 任意の $\sigma \in [\pi_s]$ について，σ^* を計算し，巡回置換の積に分解する
8: **for** σ^* の全ての巡回置換成分 **do**
9:　0 か 1 をランダムに割り当てる
10: **end for**
11: **return** 1 が割り当てられた巡回置換成分に含まれる辺から成るグラフ x

Simple Algorithm は，ラベルなしグラフを完全にランダムに生成する．また，N 個のグラフを生成するときは，6〜11 ステップを N 回繰り返せばよい．しかし，Simple Algorithm では，分割の生成と重みの計算を $p(n)$ 回行う必要がある (1〜5 ステップ)．$p(n)$ は n の増加とともに急速に大きくなり (図 2)，実際的な計算量ではない．そこで Dixon らは，$p(n)$ 回のループ処理を回避するため，分割を選択するための 1〜5 ステップの代わりに別の手法を用いた．

5.2 Dixon/Wilf Algorithm

Dixon らは，少数の和因子 1 の数が多い分割によって，重みの合計の大部分が占められていることを利用し，$p(n)$ の計算量を回避する方法を提案した．次の定理を用いると，重みの合計を頂点数 n のラベルなしグラフの数 $\mathcal{G}_n$ から求めることができ，重みの合計を求めるために $p(n)$ 個の分割の重みを計算する必要がない．ここで $\mathcal{G}_n$ は既知の定数として，事前に与えられていることを前提としている．

[定理 6]　(バーンサイドの定理) 集合 X 上に群 G が作用するとき，その軌道の数は次式で与えられる．

$$|G\backslash X| = \frac{1}{|G|}\sum_{g\in G}|X_g| \quad (9)$$

バーンサイドの定理から重みの合計が求まる．

$$\begin{aligned}\sum_{k=1}^{m} w(C_i) &= \sum_{k=1}^{m} |C_i|\,|\mathcal{M}_{\sigma_i}|\,(\sigma_i \in C_i)\\ &= \sum_{\sigma\in S_n}|\mathcal{M}_\sigma|\\ &= n!\mathcal{G}_n\end{aligned} \quad (10)$$

重みの合計が最初に求まると，乱数 r を生成できるので，重みの累積が r の値に達したところで分割の生成を止めることができる．さらに Dixon らは，和因子 1 の数が多い順に分割を生成すると，どんなに n が大きくても平均すると 3 回より少ない分割の生成で r の値に達すると示した．彼らはこのことを利用

し，平均すると 3 回より少ない分割の生成で，分割を選択するアルゴリズムを提案した．

図 8 は，分割を和因子 1 の数が多い順に生成した場合の，その重みの具体的な例である．横軸に全ての 30 の分割を和因子 1 が多い順に並べ，縦軸に分割の重みの常用対数をとった．最初の数個の分割によって，重みの合計の大部分が占められている様子がわかる．また，最初の数個までで分割の計算が終わると，大量の分割の計算が省略できることがわかる．

Dixon/Wilf Algorithm$(n, \mathcal{G}_n)$

1: 1 から $n!\mathcal{G}_n$ までの自然数 r をランダムに生成する
2: **for** $k = 1$ to n **do**
3: **for** 和因子 1 を $n-k$ 個もつ全ての n の分割 π **do**
4: 対応する共役類の重み $w([\pi])$ を計算する
5: 重みの累積 $a([\pi])$ を計算する
6: **if** $r \leqq a([\pi])$ **then**
7: $\pi_s \leftarrow \pi$
8: 12 へジャンプする
9: **end if**
10: **end for**
11: **end for**
12: 任意の $\sigma \in [\pi_s]$ について，σ^* を計算し，巡回置換の積に分解する
13: **for** σ^* の全ての巡回置換成分 **do**
14: 0 か 1 をランダムに割り当てる
15: **end for**
16: **return** 1 が割り当てられた巡回置換成分に含まれる辺から成るグラフ x

5.3 提案手法：Practical Algorithm

Dixon/Wilf Algorithm では，$\mathcal{G}_n$ は既知の定数として事前に与えられていることを前提としている．与えられていない場合は，$\mathcal{G}_n$ を計算する必要があるが，$\mathcal{G}_n$ を計算する際のコストは大きい．また，非常に小さな確率だが，r が重みの合計に近い値をとった場合，$p(n)$ に近い数の分割を生成しなければならない．複数のグラフを生成するとき，生成数が大きいほど多くの分割を生成する可能性が増える．そこで，選ばれる確率が非常に小さい分割を最初から省いてしまうのが，次のアルゴリズムである．

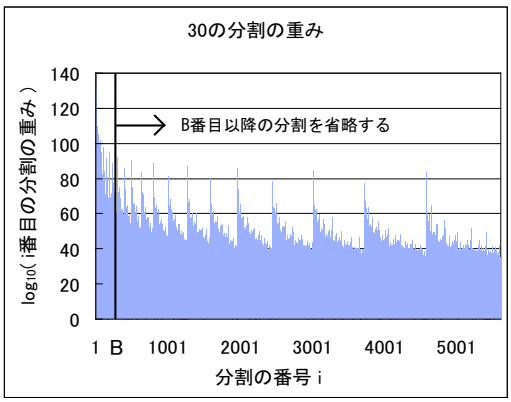

図 8　和因子 1 の数が多い順に生成した場合の分割の重みと B による分割の省略

我々は，和因子 1 が多い順に分割を生成する Dixon らの手法に加えて，生成する分割の個数を適当な定数 B 個に制限する手法を提案する．Practical Algorithm は，まず分割を和因子 1 の数が多い順に適当な定数 B 個まで生成する．$B+1$ 個目以降の分割は省略し，生成された分割の中から，重みの比にしたがって 1 つ分割を選択する．

これにより，Dixon/Wilf Algorithm は非常に小さい確率だが $p(n)$ 回のループ処理を必要とする可能性があるのに対し，Practical Algorithm は確実にループ処理を $O(1)$ 回まで減少させる．また，Practical Algorithm は Simple Algorithm と同様，重みの合計を重みを累積して求めるため，$\mathcal{G}_n$ を必要としない．

Dixon/Wilf Algorithm と同様に図 8 を見れば，$B+1$ 個目以降の分割を省略することの有効性がわかる．また，最初の数個の分割によって，重みの合計の大部分が占められていることから，省略の妥当性がみてとれる．

Practical Algorithm(n, B)

1: **for** $k = 0$ to n **do**
2: 和因子 1 を $n-k$ 個もつ全ての n の分割 π を総合計 B 個になるまで生成する
3: **end for**
4: **for** $i = 1$ to B **do**
5: 重み $w([\pi_i])$ を計算する
6: 重みの累積 $a([\pi_i]) = w([\pi_i]) + a([\pi_{i-1}])$ を計算する (初期値 $a([\pi_1]) = w([\pi_1])$)
7: **end for**
8: 1 から $a([\pi_B])$ までの自然数 r ランダムに生成し，$a([\pi_{s-1}]) < r \leqq a([\pi_s])$ を満たす s を求める
9: 任意の $\sigma \in [\pi_s]$ について，σ^* を計算し，巡回置換の積に分解する
10: **for** σ^* の全ての巡回置換成分 **do**
11: 0 か 1 をランダムに割り当てる
12: **end for**
13: **return** 1 が割り当てられた巡回置換成分に含まれる辺から成るグラフ x

Practical Algorithm は，他のアルゴリズムと違いラベルなしグラフを厳密にはランダムに生成しない．$B+1$ 個目以降の分割を省略するため，省略された分割による重みの比が大きいほど，生成されるグラフに偏りが出てくる．しかし，適切な B を選べばその誤差は無視できるほど小さくなる．

Practical Algorithm では B を大きくすれば，誤差は少なくなるが計算コストが増える．逆に B を小さくすれば，計算コストは減るが誤差が大きくなる．また，生成数が大きいほど誤差による影響が大きくなるため，許容できる誤差の範囲と生成数，および計算コストとのバランスが重要である．

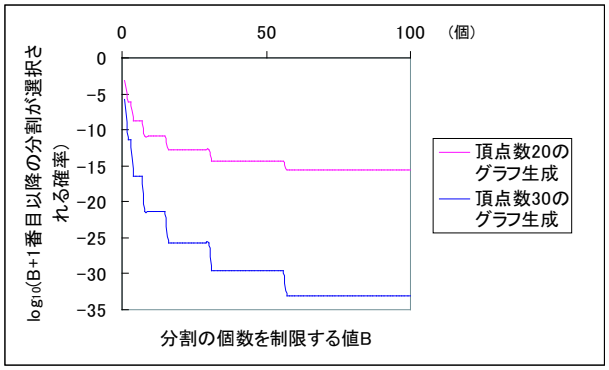

図 9　B による省略を行った際のグラフ生成と完全にランダムなグラフ生成との誤差

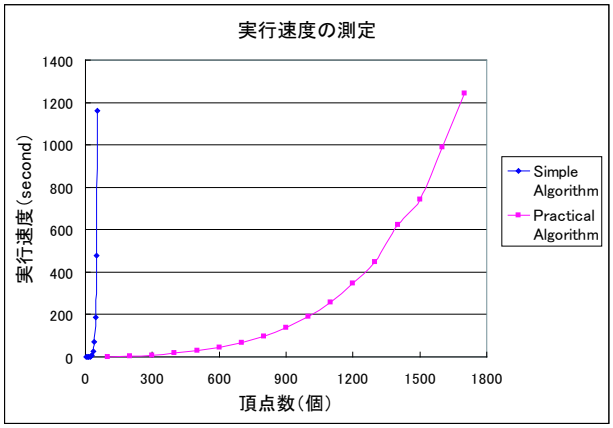

図 10　実行速度の測定

図 9 は，B による省略を行った際のグラフ生成 (Practical Algorithm) と完全にランダムなグラフ生成 (Simple Algorithm, Dixon/Wilf Algorithm) の誤差を示したグラフである．横軸に B を，縦軸に全ての分割を生成した際に，$B+1$ 番目以降の分割が選択される確率の常用対数をとっている．例えば頂点数 20 では，2 番目以降の分割が選択される確率は約 $10^{-3.1}$，101 番目以降の分割が選択される確率は約 $10^{-15.5}$ である．これに対して頂点数 30 では，2 番目以降の分割が選択される確率は約 $10^{-5.7}$，101 番目以降の分割が選択される確率は約 $10^{-33.1}$ である．

固定した B に対して，頂点数 n が大きいほど誤差の影響が少なくなる傾向が多く見られた．そこで，小さい頂点数で許容できる誤差の B を定めておき，その B を大きな頂点数においても適用すれば，誤差がそれ以上大きくなることはない．

6. 性 能 評 価

6.1 実 験 環 境

CPU:Intel(R) Pentium(R) 4 2.20GHz，メモリ:1.00GB，OS:Windous XP を搭載した PC を使用し，Mathematica4.2 を使用して実装を行った．

擬似乱数の生成には Mathematica の Random 関数を，ランダム性の評価には (部分) グラフ同型判定アルゴリズム VF2 [10] を利用した．

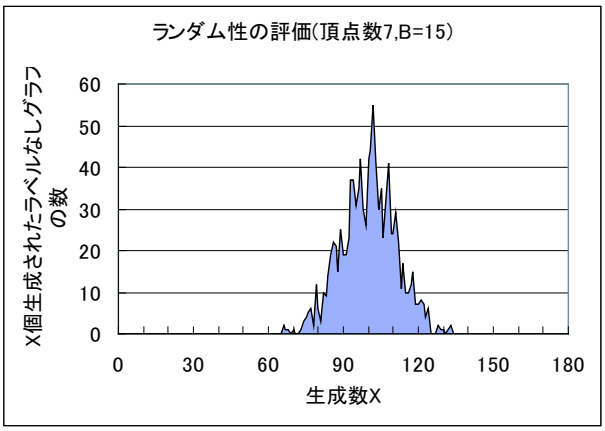

図 11　ランダム性の評価 (頂点数 $7, B = 15$)

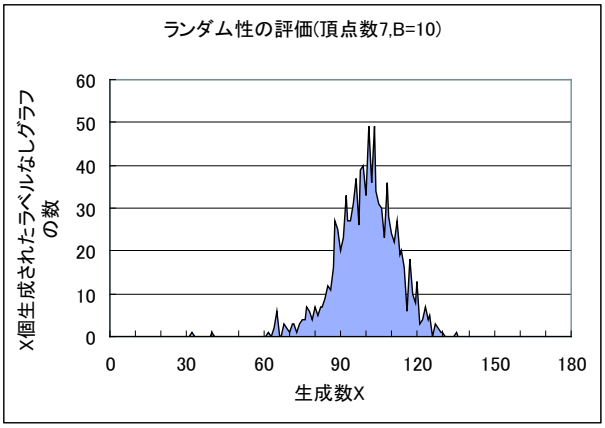

図 12　ランダム性の評価 (頂点数 $7, B = 10$)

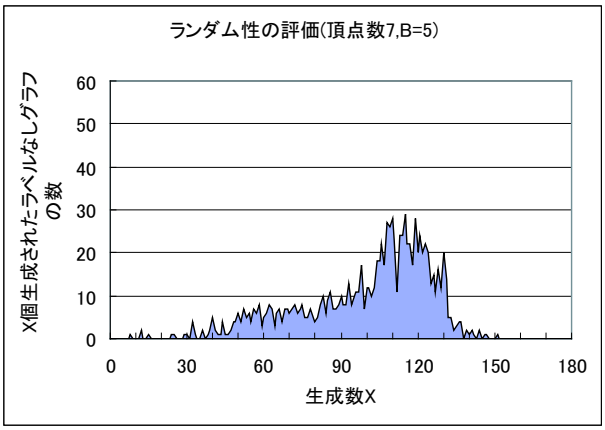

図 13　ランダム性の評価 (頂点数 $7, B = 5$)

6.2 実行速度の測定

Simple Algorithm，Practical Algorithm(B=100) を用いて実行速度を測定した．図 10 はその測定結果である．横軸に頂点数を，縦軸にグラフを 1 つ生成する際にかかる時間をとっている．Practical Algorithm のほうが，より少ない時間でグラフを生成できることがわかる．

Simple Algorithm は $p(n)$ 個の分割の重みの計算に $O(np(n))$ の計算量がかかる．それに対し，Practical Algorithm は分割の重みの計算にかかる計算量が $O(n)$ であるため，全体の計算

量を $O(n^2)$ まで下げている．

頂点数 n のラベルなしグラフの数 $\mathcal{G}_n$ が与えられている場合，Dixon/Wilf Algorithm も非常に大きな確率で Practical Algorithm と同程度の実行速度が期待できるが，逆に非常に小さな確率だが Simple Algorithm と同程度の実行速度になる可能性もある．

6.3 ランダム性の評価

Practical Algorithm を用い，頂点数 7 のラベルなしグラフを $B = 5, 10, 15$ の 3 ケースで，104400 個 ($\mathcal{G}_7 = 1044$) 生成した．1 つのグラフが 100 個ずつ生成されているのが理想である．図 11〜13 はその実験結果である．横軸が生成数 X を，縦軸は 1044 種類のラベルなしグラフのうち X 個生成されたグラフの数を表している．

図 11 は $B = p(7)$ であり，グラフを完全にランダムに生成するケースである．それに対して，図 12，13 は分割を省略しており，完全にランダムなグラフ生成とは誤差が生じるケースである．図 12 は図 11 に比べて生成数の分布がばらついてはいるが大きな違いは見られない．それに対して，図 13 は分布が大きくばらついている．頂点数 7，$B = 5$ では省略による誤差が大きく，実際的でないことがわかる．

Practical Algorithm の実際性を評価するためには，より大きな頂点数でのランダム性の評価を行う必要があったが，テストにかかる計算コストの大きさから頂点数 7 が限界であった．

7. ま と め

本研究では，Dixon らの和因子 1 の数が多い順に分割を生成するという手法に加えて，生成する分割の個数を制限することで，$\mathcal{G}_n$ を必要としないランダムなラベルなしグラフ生成アルゴリズムを提案した．ここでいうランダムとは，頂点数 n の全てのラベルなしグラフの集合の中から 1 つを等確率で取り出すという意味である．しかし，頂点数 n の増加に伴い，$\mathcal{G}_n$ が急速に増加するため，大きな n についての提案手法のランダム性の評価ができなかった．今後の課題は，大きな n についてのランダム性の評価とラベルつきグラフへの提案手法の適用である．

謝辞 本研究の一部は，(独) 日本学術振興会科学研究費補助金基盤研究（B）(2)（課題番号:16300030）による．

文 献

[1] J.D.Dixon and H.S.Wilf, *The Random Selection of Unlabelled Graphs*, Journal of Algorithms 4 (1983) 205-213.

[2] A.Kerber, *Constructing Finite Unlabelled Structures Using Group Actions*, Annals of Combinatorics 5 (2001) 363-380.

[3] F.Harary, *the number of linear, directed, rooted, and connected graphs*, Trans.Amer.Math.Soc. 78 (1955) 445-463.

[4] F.Harary and E.M.Palmer, *Graphical Enumeration*, Academic Press, New York, 1973.

[5] A.Zoghbi and I.Stojmenovic, *Fast Algorithms for Generating Integer Partitions*, Intern.J.Computer.Math. 70 (1998) 319-332.

[6] N.C.Wormald, *Generating Random Unlabelled Graphs*, SIAM.J.Comput 16 (1987) 717-727.

[7] M.Zito, Ida Pu, M.Amos and A.Gibbons, *RNC Algorithms for the Uniform Generation of Combinatorial Structures*, SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms) (1996).

[8] L.A.Goldberg, *Randomly Sampling Unlabelled Structures*, Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom.

[9] D.L.Kreher and D.R.Stinson, *Combinatorical Algorithms :Generation,Enumeration,and Search*, CRC Press, Boca Raton, London, New York, Washington,D.C, 1999.

[10] L.P. Cordella, P. Foggia, C. Sansone, M. Vento, *An Improved Algorithm for Matching Large Graphs*, Proc. of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition, Ischia, May 23-25 (2001) 149-159.

[11] W.Wang, C.Wang, Y.Zhu, B.Shi, J.Pei, X.Yan and J.Han, *GraphMiner: a structural pattern-mining system for large disk-based graph databases and its applications*, Proceedings of the 2005 ACM SIGMOD international conference on Management of data 2005, Baltimore, Maryland, June 14-16 (2005) 879-881

[12] ジョージ・アンドリュース，キムモ・エリクソン著 佐藤 文広訳『整数の分割』 数学書房，2006 年.

[13] 寺田 至，原田 耕一郎 『岩波講座 現代数学の基礎 群論』 岩波書店，1997 年.

[14] combinatorial graph theory，http://keithbriggs.info/cgt.html, Keith Briggs，(2007/2/16 アクセス)