Vol.2019-AL-172 No.2
2019/3/5

ポリアの数え上げに基づく非同型な塗分けの索引化

角野 周平[1,a)]　堀山 貴史[1,b)]

概要：自己同型群が与えられたときに、Pólya の数え上げを利用して非同型な塗分けの個数を数え上げることができる。また、2018 年に Horiyama らにより提案された同型除去の手法により、零抑制型二分決定グラフ (ZDD: Zero-suppressed Binary Decision Diagram) を利用して、自己同型群のもとで非同型な塗分けを列挙することもできる。この手法は、実行時間の観点からも、メモリ使用量の観点からも、従来手法よりも大幅に効率的であるが、結果の ZDD のサイズが大きい場合には、必然的にメモリ使用量も大きくなる。本研究では、自己同型群と整数 a が与えられたときに、非同型な塗分けに番号付けを行い、索引番号 a の非同型な塗分けを求める問題を提案する。また、数え上げと列挙の両方の手法を利用して、非明示的に (つまり明示的に列挙することなく) 番号付けを行いこの問題を解くための手法を提案する。

Indexing Nonisomorphic Graph Colorings Based on the Pólya's Theorem

Shuhei Kadono[1,a)]　Takashi Horiyama[1,b)]

Abstract: Given an automorphism on a set, we can count the number of nonisomorphic colorings by the Pólya's counting theorem. We can also enumerate the nonisomorphic colorings on the automorphism by the isomorphism elimination method with ZDDs (Zero-suppressed Binary Decision Diagrams), which is proposed by Horiyama et al. in 2018. This method is more than 100 times efficient on the computation time and memory consumption than conventional methods. Unfortunately, however, the memory consumption becomes large in case the resulting ZDD is large. In this paper, we propose the following problem: Given an automorphism on a set and an integer a, we implicitly make an index on the nonisomorphic colorings, and generate the one with index a. We also propose an algorithm that solve this problem by implicitly making an index (without enumerating the nonisomorphic colorings). We utilize both of the techniques on counting and enumeration.

1. はじめに

列挙アルゴリズムは、指定された条件を満たすものを、漏れなく重複なく、すべて求めるための技術である。また、数え上げアルゴリズムは、条件を満たすものが何個あるかを、正確に求めるための技術である。

たとえば、多面体 P の展開図が得られるための必要十分条件は、その多面体 P の 1-スケルトン G (多面体の頂点と辺からなるグラフ構造) において切り開く辺が全域木となることである ([7] など参照)。全域木の列挙アルゴリズムとして、逆探索による方法 [1] や、二分決定グラフ (BDD: Binary Decision Diagrams) [3] およびその亜種である零抑制型二分決定グラフ (ZDD: Zero-suppressed BDD) [17] を用いる方法 [19][14] が知られている。逆探索では、台グラフ G の全域木の間に親子関係を定義し、G のすべての全域木が先祖/子孫の関係を持つ家系木をたどることで、全域木の列挙ができる。BDD/ZDD は、有向非巡回グラフによる集合族の表現法である。全域木を構成する辺を集合として表し、その族を表す BDD/ZDD を構築する手法は、Sekine らによって提案され [19]、また近年、Knuth によるパス列挙手法 [16] の一般化であるフロンティア法 [14] として提案されている。

[1] 埼玉大学 理工学研究科
Graduate School of Science and Engineering, Saitama University, Japan
[a)] s.kadono.119@ms.saitama-u.ac.jp
[b)] horiyama@al.ics.saitama-u.ac.jp

また、全域木の数え上げには行列木定理 [15] が知られており、理工学の様々な分野で用いられている。たとえば、計算化学の分野では、フラーレンの仲間たちに対して数え上げがされており、サッカーボールの形をした C_{60} (切頂二十面体) における全域木の個数 (したがって展開図が得られるような辺の切り開き方) は 375,291,866,372,898,816,000 通りであると求められている [2]。

列挙や数え上げにおいて、対象の同型性に着目して、非同型なもの (すなわち本質的に異なるもの) のみの列挙や数え上げが求められることもある。たとえば、展開図の列挙においては、前述の列挙アルゴリズムにより、どの辺 (どのラベルの辺) を切り開けば展開図を得られるかが分かる。立方体に対して、この方法により 384 通りの辺の切り開き方が得られるが、辺のラベルは無視して、本質的に異なる非同型な 11 種類の展開図が必要となることも多い。Horiyama と Shoji は、多面体の回転や鏡映反転による辺の置換からなる自己同型群を利用し、BDD により非同型な展開図を列挙する手法を提案している [11]。また、近年、Horiyama らは、フロンティア法 [14] の枠組みに基づき、多面体の展開図に限らず任意の同型性を排除する BDD/ZDD の構築手法 [10] を提案し、最大 100 倍の高速化と最大 1,000 倍の省メモリ化を達成している。

非同型なものの数え上げにおいては、Pólya の数え上げ [18] や Cauchy-Frobenius の補題 [6][8] (Burnside の補題 [5]) と呼ばれる群論に基づく手法が知られている。たとえば、非同型な展開図の数え上げでは、計算機により具体的に列挙する [11] 前から、正十二面体および正二十面体が 43,380 種類の非同型展開図を持つことが知られていた [9][13]。この手法は、具体的な列挙なしで数え上げを行う手法であり、4 次元正多胞体の非同型な展開図の数え上げ [4] へと拡張され、さらに、任意の 3 次元多面体に対する非同型な展開図の数え上げ [12] へと一般化された。これにより、切頂二十面体における非同型な展開図は 3,127,432,220,939,473,920 種類であると求められている。

本稿では、これまでに述べた非同型なものの列挙と数え上げの 2 つをつなぐ技術として、非同型なものの索引化に取り組む。これは、非同型なものすべてに対して非明示的に番号付けを行うことで、与えられた索引番号 (整数) a に対して、a 番目のものを求める問題である。より具体的には、台集合 Γ とその上での置換からなる自己同型群 $\mathrm{Aut}\,\Gamma$、および整数 a が与えられたときに、非同型な塗分けに非明示的に番号付けを行い、a 番目の非同型な塗分けを求める。ここで、非明示的に番号付け、すなわち、すべての非同型な塗分けを列挙することなく、それらに番号付けを行うことが、本手法の主眼である。

たとえば、非同型な塗分けの索引化として、白黒に塗り分けた数珠の索引化では、数珠の m 個の珠の集合 $\{1, 2, \ldots, n\}$ を台集合とし、数珠の回転や鏡映反転を表す珠の置換からなる自己同型群 $\mathrm{Aut}\,\Gamma$ が与えられる。また、たとえば、展開図の索引化では、多面体 P の 1-スケルトン $G = (V, E)$ の辺集合である E を台集合とし、P の回転や鏡映反転を表す辺の置換からなる自己同型群 $\mathrm{Aut}\,\Gamma$ が与えられる。各展開図と対応する全域木は、ラベル付きの辺の集合として表すが、これは、台集合 E の各辺を全域木に採用するかしないかを塗分けているとみなすことができる。全域木と対応するすべての塗分けに対し、$\mathrm{Aut}\,\Gamma$ に関して同型となるものを除去し、非同型な塗分けに (非明示的に) 番号付けを行い、a 番目のものを取り出す。

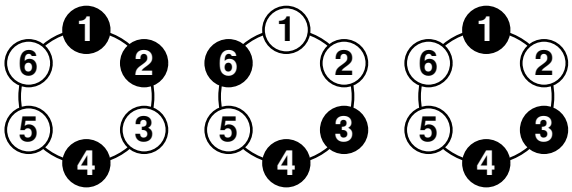

図 1 珠 6 個の数珠の同型な塗分け

Horiyama らによる BDD/ZDD を用いた非同型なものの列挙 [10] は、計算時間の観点からもメモリ使用量の観点からも大きな進展であったが、それでもなお、依然としてメモリ使用量が膨大で列挙できない場合が多く存在する。Pólya の数え上げにより、その場合でも非同型な塗分けが何個あるかを求めることはできるが、具体的に塗分けを得ることはできない。本稿で提案する索引化では、そのような場合にも、索引番号から非同型な塗分けを具体的に得ることができ、疑似的に列挙結果のカタログと持っているのと同じとみなせるようになる。これは、たとえば、すべての非同型な塗分けを列挙しなくても、その列挙結果をランダムサンプリングできるようになるなど、さまざまな応用に適用できることを示唆している。

非同型な塗分けの索引化のために、本稿では以下のアプローチをとる。自己同型群 $\mathrm{Aut}\,\Gamma$ の置換をもとに、非同型な塗分けを非明示的にグループ分けする。$\mathrm{Aut}\,\Gamma$ の各置換ごとに Pólya の数え上げを適用することで、二分探索の要領で、索引番号 a の非同型な塗分けが含まれるグループを特定し、その中から目的の塗分けを取り出す。この二分探索において、探索範囲の非同型な塗分けを保持したり、Pólya の数え上げを行うために、塗分けの集合を ZDD で表し、ZDD の強力な表現能力と演算処理系を用いる。なお、本稿では、非同型な数珠の塗分け問題を例に、索引化の手法を説明するが、この手法は、非同型な展開図の索引化など、任意の塗分けの索引化に適用することができる。

2. 準備

2.1 Pólya の数え上げ

台集合 $\Gamma = \{1, 2, \ldots, n\}$ の上での置換からなる自己同型群 $\mathrm{Aut}\,\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ が与えられたとする。台集

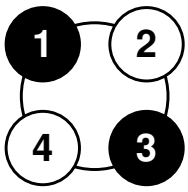

図 2 $\sigma = (1\,3)(2\,4)$ で動かない数珠の塗分け

合 Γ の各要素の c 色の塗分けは、Γ から $\{1, 2, \ldots, c\}$ への写像 f_c により表される。本稿では、主に $c = 2$ の場合を扱うが、提案アルゴリズム (3 章の Algorithm 1) は一般の c に対して動作する。塗分け f_c と置換 σ_i に対し、$\sigma_i \circ f_c(j) := f_c(\sigma_i(j))$ により定義される塗分け、すなわち、塗分け f_{c_1} の各要素に置換 σ_i を適用した後の塗分け $\sigma_i \circ f_c$ を、塗分け f_c の置換 σ_i による移動後の塗分けと呼ぶ。2 つの塗分け f_{c_1}, f_{c_2} に対し、$\exists\sigma_i (\in \mathrm{Aut}\,\Gamma)\ \sigma_i \circ f_{c_1} = f_{c_2}$ が成立するとき、すなわち、塗分け f_{c_1} の置換 σ_i による移動後の塗分けが f_{c_2} と一致するような σ_i が存在するときに、2 つの塗分けは同型 $f_{c_1} \sim f_{c_2}$ であるという。

たとえば、珠 6 個 $\Gamma = \{1, 2, \ldots, 6\}$ からなる数珠において、回転及び鏡映反転からなる自己同型群 $\mathrm{Aut}\,\Gamma$ は 12 種類の置換からなる。図 1 (a), (b) の塗分けは、(a) のそれぞれの珠を時計回りに 120 度回転させる置換 $(1\,3\,5)(2\,4\,6)$ により、同型であると分かる。また、図 1 (a), (c) の塗分けは、(a) のそれぞれの珠を上下に鏡映反転させる置換 $(1\,4)(2\,3)(5\,6)$ により、同型であると分かる。したがって、推移律より、図 1 (a), (b), (c) の 3 つの塗分けは同型である。

塗分け f_c と置換 σ_i に対し、$\sigma_i \circ f_c = f_c$ が成立するとき、すなわち、塗分け f_c の置換 σ_i による移動後の塗分けが f_c 自身となるとき、f_c は置換 σ_i により動かない塗分けであるという。置換 σ_i において、要素 j が σ_i により移動する $\{\sigma_i^k(j) \mid k \text{ は整数}\}$ を σ_i による j の軌道と呼び、すべての要素に関する軌道の集合 θ_{σ_i} を σ_i の軌道の集合と呼ぶ。f_c が σ_i の各軌道上の要素に同じ塗分けを与えることが、f_c が σ_i により動かない塗分けであるための必要十分条件である。たとえば、$\sigma = (1\,3)(2\,4)$ は、$\theta_\sigma = \{\{1, 3\}, \{2, 4\}\}$ を軌道の集合として持つ。図 2 の塗分けは、珠 1, 3 が黒、珠 2, 4 が白の塗分けであり、σ の各軌道上の珠は同色であるため、σ で動かない塗分けである。

塗分けの集合 S_c と自己同型群 $\mathrm{Aut}\,\Gamma$ に対して、非同型な塗分けの個数 $N(S_c, \mathrm{Aut}\,\Gamma)$ は、

$$\frac{1}{|\mathrm{Aut}\,\Gamma|} \sum_{\sigma_i \in \mathrm{Aut}\,\Gamma} |\{f_c \in S_c \mid \sigma_i \circ f_c = f_c\}| \qquad (1)$$

により求められる [18]。たとえば、珠 4 個の数珠に対し、恒等置換 σ_1、時計回りに 90 度, -90 度, 180 度回転を表す置換 σ_2 から σ_4, および 4 つの対称軸による鏡映反転 σ_5 から σ_8 からなる自己同型写像 $\mathrm{Aut}\,\Gamma$ を考える。珠 4 個の数珠の塗分けは、全部で $2^4 = 16$ 種類ある。その各塗分けごとに、置換 σ_1 から σ_8 で動くかどうかを表 1 に示す。このようにして作った表を、同型表と呼ぶ。同型表 (表 1) の各行は、それぞれ列 1〜4 に示す 1 つの塗分け f_c に対応している。以降の σ_i の列には、各塗分け f_c が $\sigma_i \circ f_c = f_c$ すなわち置換 σ_i で動かない (○) か否 (×) を表す。非同型な塗分けの個数 $N(S_c; \mathrm{Aut}\,\Gamma)$ は、この表の各列の○の個数を累算することで、$N(S_c; \mathrm{Aut}\,\Gamma) = 48/8 = 6$ と求められる。

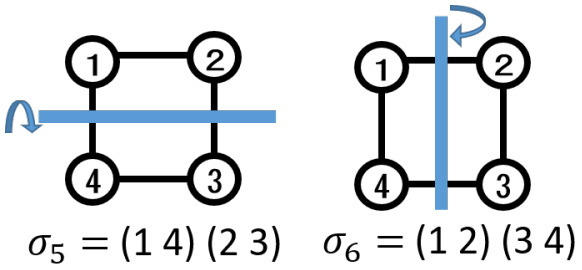

図 3 本質的に同じ操作を行う置換 σ_5, σ_6

3. 非同型な塗分けの検索アルゴリズム

本章では、台集合 Γ の上での置換からなる自己同型群 $\mathrm{Aut}\,\Gamma$ と、台集合の各要素の塗分けの集合 S_c および索引番号 (整数) a が与えられたときに、S_c の $\mathrm{Aut}\,\Gamma$ に関する非同型な塗分けに非明示的に番号付けを行い、a 番目の非同型な塗分けを求める手法を提案する。このために、まず基本的なアイデアを述べ、次に、このアイデアを利用して二分探索に基づいて a 番目の非同型な塗分けを求める手法の詳細について述べる。

3.1 基本アイデア

まず、2.1 章で述べた非同型な塗分けの個数 $N(S_c, \mathrm{Aut}\,\Gamma)$ の求め方を変更する。2.1 章では、表 1 に示すような同型表において、各列の○の個数を累算することで $N(S_c, \mathrm{Aut}\,\Gamma)$ を求めた。ここで、表の見方を変更し、同型表の各行の○の個数を累積する。すなわち、式 (1) の Pólya の数え上げの式を変形し、

$$\frac{1}{|\mathrm{Aut}\,\Gamma|} \sum_{f_c \in S_c} |\{\sigma_i \in \mathrm{Aut}\,\Gamma \mid \sigma_i \circ f_c = f_c\}| \qquad (2)$$

により $N(S_c; \mathrm{Aut}\,\Gamma)$ を求めることにする。こうすることにより、同型表での○/×が同じ行 (塗分け) は、同型な塗分けを持つことが分かる。さらに、自己同型群 $\mathrm{Aut}\,\Gamma$ と同型表を観察することで、以下のことが分かる。

[観察 1] 置換 $\sigma_2 = (1\,2\,3\,4)$ と $\sigma_3 = (1\,4\,3\,2)$ は、$\theta_{\sigma_2} = \theta_{\sigma_3} = \{\{1, 2, 3, 4\}\}$ と軌道の集合が同じであるので、同型表の σ_2 と σ_3 の列は○/×が一致する。

[観察 2] 置換 $\sigma_5 = (1\,4)(2\,3)$ と $\sigma_6 = (1\,2)(3\,4)$ は、図 3 に示すように、共に、珠 2 個と珠 2 個を分かつ対称軸によ

表 1 珠 4 個の数珠の塗分けと、各置換で動かない塗分け

1	2	3	4	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8
●	●	●	●	○	○	○	○	○	○	○	○
○	○	○	○	○	○	○	○	○	○	○	○
●	○	●	○	○	×	×	○	×	×	○	○
○	●	○	●	○	×	×	○	×	×	○	○
●	●	○	○	○	×	×	×	×	○	×	×
○	○	●	●	○	×	×	×	×	○	×	×
○	●	●	○	○	×	×	×	○	×	×	×
●	○	○	●	○	×	×	×	○	×	×	×
●	○	○	○	○	×	×	×	×	×	○	×
○	○	●	○	○	×	×	×	×	×	○	×
○	●	●	●	○	×	×	×	×	×	○	×
●	●	○	●	○	×	×	×	×	×	○	×
○	●	○	○	○	×	×	×	×	×	×	○
○	○	○	●	○	×	×	×	×	×	×	○
●	○	●	●	○	×	×	×	×	×	×	○
●	●	●	○	○	×	×	×	×	×	×	○

る鏡映反転を表す置換である。これらは、対象となる珠は異なるものの、本質的に同じ操作を表している。また、置換 $\sigma_7 = (1)(3)(2\,4)$ と $\sigma_8 = (2)(4)(1\,3)$ も共に、対蹠となる珠 2 個を結ぶ対称軸による鏡映反転を表す置換であり、やはり本質的に同じ操作を表している。

これらの観察より、σ_2 と σ_3 を同一視し、σ_5 と σ_6、σ_7 と σ_8 をそれぞれ同じ型の置換とみなすことで、表 1 の同型表は、表 2 のように塗分けをグループ分けすることができる。ここで、σ_2 と σ_3 を同一視しているため、表 2 において、σ_2 の列の○に重み 2 を掛けて○の個数を求めることで、$N(S_c; \mathrm{Aut}\,\Gamma)$ は正しく求められる。また、(σ_5, σ_6) が $(○, ×)$ もしくは $(×, ○)$ となる状態を同一視し、(σ_7, σ_8) が $(○, ×)$ もしくは $(×, ○)$ となる状態も同一視することで、同型な塗分けが同じグループになるように、塗分けの集合 S_c を分割することができる。

観察 1 について、塗分け f_c が置換 σ_{i_1} で動かないのは、定義より、$\sigma_{i_1} \circ f_c = f_c$、すなわちすべての $j \in \{1, 2, \ldots, n\}$ に対して $f_c(\sigma_{i_1}(j)) = f_c(j)$ が成り立つときである。したがって、2 つの置換 $\sigma_{i_1}, \sigma_{i_2}$ の軌道の集合が等しい (すなわち $\theta_{\sigma_{i_1}} = \theta_{\sigma_{i_2}}$) ならば、以下の補題が成り立つ。

補題 1 $\mathrm{Aut}\,\Gamma$ の 2 つの置換 $\sigma_{i_1}, \sigma_{i_2}$ に対し、$\theta_{\sigma_{i_1}} = \theta_{\sigma_{i_2}}$ が成り立つならば、任意の塗分け f_c に対して $(\sigma_{i_1} \circ f_c = f_c) \iff (\sigma_{i_2} \circ f_c = f_c)$ が成り立つ。

このとき、軌道の集合の等価性 $\theta_{\sigma_{i_1}} = \theta_{\sigma_{i_2}}$ による二項関係を $\sigma_{i_1} \approx_\theta \sigma_{i_2}$ と書くことにすると、$\mathrm{Aut}\,\Gamma$ を $\approx_\theta$ に関する同値類 $\mathcal{Q} = \{Q_i\ (\subseteq \mathrm{Aut}\,\Gamma)\}$ に分割することができる。ここで、$\mathcal{Q}$ は $\mathrm{Aut}\,\Gamma$ の分割であり、(i) $\bigcup_{Q_i \in \mathcal{Q}} Q_i = \mathrm{Aut}\,\Gamma$, (ii) $\forall \sigma_{i_1}, \sigma_{i_2} \in Q_i\ \sigma_{i_1} \approx_\theta \sigma_{i_2}$, (iii) $\forall i \neq j', \sigma_{i_1} \in Q_i, \sigma_{i'_1} \in Q_{i'}\ \sigma_{i_1} \not\approx_\theta \sigma_{i'_1}$ を満たす。さらに、分割 $\mathcal{Q}$ の各 Q_i に対して、代表元 $\sigma_{i_1}\ (\in Q_i)$ と頻度 $q_i = |Q_i|$ のペア (σ_{i_1}, q_i) を定義し、その集合を Σ とする。

たとえば、珠が 4 個の数珠での $\mathrm{Aut}\,\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_8\}$ では、$\sigma_2 \approx_\theta \sigma_3$ かつ、それ以外の i_1, i_2 に対しては $\sigma_{i_1} \not\approx_\theta \sigma_{i_2}$ が成り立つので、$\Sigma = \{(\sigma_2, 2)\} \cup \{(\sigma_i, 1) \mid i = 1, 4, 5, 6, 7, 8\}$ を得る。

観察 2 について、図 3 の σ_5, σ_6 のように、2 つの置換 $\sigma_{i_1}, \sigma_{i_2}$ は、ラベルの入れ換えにより同じ置換となるときに、本質的に同じ操作であると呼ぶ。ここで、ラベルの置換は、任意のものが許されるのではなく、$\sigma_k \in \mathrm{Aut}\,\Gamma$ のみが許される。そこで、置換 σ_k による置換 $\sigma_{i_1} : j \mapsto \sigma_{i_1}(j)$ のラベルの入れ換え操作 $\sigma_k \bigstar \sigma_{i_1}$ を、$\sigma_k(j) \mapsto \sigma_k(\sigma_{i_1}(j))$ として定義する。一般に、$\sigma_k \bigstar \sigma_{i_1} = \sigma_{i_2}$ が成立するならば、同時に $\sigma_k^{-1} \bigstar \sigma_{i_2} = \sigma_{i_1}$ も成立する。

2 つの置換 $\sigma_{i_1}, \sigma_{i_2}$ に対して

$$\exists \sigma_k (\in \mathrm{Aut}\,\Gamma)\ \sigma_k \bigstar \sigma_{i_1} = \sigma_{i_2}$$

となるとき、すなわち σ_{i_1} と σ_{i_2} が本質的に同じ操作であるときに、二項関係 $\sigma_{i_1} \approx_\bigstar \sigma_{i_2}$ が成り立つと書く。二項関係 $\approx_\bigstar$ に関して、以下の補題が成り立つ。

表 2 珠 4 個の数珠の塗分けのグループ分け

	1	2	3	4	σ_1	σ_2	σ_4	σ_5	σ_6	σ_7	σ_8
グループ1	●	●	●	●	○	○	○	○	○	○	○
	○	○	○	○	○	○	○	○	○	○	○
グループ2	●	○	●	○	○	×	○	×	×	○	○
	○	●	○	●	○	×	○	×	×	○	○
グループ3	●	●	○	○	○	×	×	×	○	×	×
	○	○	●	●	○	×	×	×	○	×	×
	○	●	●	○	○	×	×	○	×	×	×
	●	○	○	●	○	×	×	○	×	×	×
グループ4	●	○	○	○	○	×	×	×	×	○	×
	○	○	●	○	○	×	×	×	×	○	×
	○	●	●	●	○	×	×	×	×	○	×
	●	●	○	●	○	×	×	×	×	○	×
	○	●	○	○	○	×	×	×	×	×	○
	○	○	○	●	○	×	×	×	×	×	○
	●	○	●	●	○	×	×	×	×	×	○
	●	●	●	○	○	×	×	×	×	×	○

補題 2 Σ の 2 つのペア (σ_{i_1}, q_i), $(\sigma_{i'_1}, q_{i'})$ に対し、$(\sigma_{i_1} \approx_\bigstar \sigma_{i'_1}) \Rightarrow (q_i = q_{i'})$ が成り立つ。

補題 3 2 つの置換 $\sigma_{i_1}, \sigma_{i_2} \in E_i$ と、$\mathrm{Aut}\,\Gamma$ に関する同型な塗分けをすべて含んだ塗分けの集合 S_c に対し、

$$|\{f_c \in S_c \mid \sigma_{i_1} \circ f_c = f_c\}| = |\{f_c \in S_c \mid \sigma_{i_2} \circ f_c = f_c\}|$$

が成り立つ。

ここで、Σ のすべての置換を、$\approx_\bigstar$ による同値類 $\{E_i\}$ に分割する。ここで、E_i は Σ 置換を要素に持つ集合であり、補題 2 より、$E_i = \{\sigma_{i_1}, \sigma_{i_2}, \ldots\}$ とすると、$|\sigma_{i_1}| = |\sigma_{i_1}| = \cdots = q_{i_1}$ が成立する。以降では、E_i と頻度のペアを $\mathcal{E} = \{(E_i, q_{i_i}) \mid q_{i_1} \in E_i\}$ と表す。

たとえば、図 3 の $\sigma_5 = (1\,4)(2\,3)$ に対して、$\sigma_2 = (1\,2\,3\,4)$ によるラベルの入れ換えを行うことで $\sigma_2 \bigstar \sigma_5 = (2\,1)(3\,4) = (1\,2)(3\,4) = \sigma_6$ が得られる。また、σ_2 による $\sigma_7 = (1)(3)(2\,4)$ のラベルの入れ換えは、$\sigma_2 \bigstar \sigma_7 = (2)(4)(1\,3) = \sigma_8$ となる。したがって、珠が 4 個の数珠での $\mathrm{Aut}\,\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_8\}$ では、$\mathcal{E} = \{(\{\sigma_1\}, 1), (\{\sigma_2\}, 2), (\{\sigma_4\}, 1), (\{\sigma_5, \sigma_6\}, 1), (\{\sigma_7, \sigma_8\}, 1)\}$ が得られる。ここで、$\sigma_2 \approx_\theta \sigma_3$ は $(\{\sigma_2\}, 2)$ に反映され、$\sigma_5 \approx_\bigstar \sigma_6$ や $\sigma_7 \approx_\bigstar \sigma_8$ は $(\{\sigma_5, \sigma_6\}, 1)$ や $(\{\sigma_7, \sigma_8\}, 1)$ に反映されていることに注意が必要である。表 2 において、$\mathcal{E}$ の 4 つのペアに各置換での ○/× の違いが、塗分けの 4 つのグループに対応している。また、各グループに属す塗分けから非同型な塗分けを得ることで、索引番号 a の非同型な塗分けがどのグループに属すかが分かり、そのグループ内の非同型な塗分けから a 番目のものを得ることができる。

3.2 アルゴリズムの詳細

前節では、与えられた $\mathrm{Aut}\,\Gamma$ の置換を分割し、$\mathcal{E}$ が得られること、その置換での ○/× により塗分けの集合 $\S_c$ がグループ分けできることを示した。これにより、各グループの塗分けを明示的に求めることで、索引番号 a の非同型な塗分けを得られることが分かった。本節では、以下の 2 つについて、順に述べる。(i) Pólya の数え上げを繰り返し適用することで、索引番号 a の非同型な塗分けが属すグループを見つける方法。(ii) ZDD による集合演算を利用して、このアルゴリズムを実現する方法。

まず、(i) の鍵になるのは、Pólya の数え上げを繰り返し適用し、二分探索の要領で、索引番号 a の非同型な塗分けが属すグループを見つけることである。ここで、塗分けの集合 S_c に対して、σ_i で動かない塗分けの集合を $S_i = \{f_c\ (\subseteq S_c) \mid \sigma_i \circ f_c = f_c\}$ とする。式 (2) による $N(S_c; \mathrm{Aut}\,\Gamma)$ の数え上げをさらに変形することで、

$$\frac{1}{|\mathrm{Aut}\,\Gamma|} \sum_{f_c \in S_c \cap S_j} |\{\sigma_i \in \mathrm{Aut}\,\Gamma \mid \sigma_i \circ f_c = f_c\}| + \frac{1}{|\mathrm{Aut}\,\Gamma|} \sum_{f_c \in S_c \cap \overline{S_j}} |\{\sigma_i \in \mathrm{Aut}\,\Gamma \mid \sigma_i \circ f_c = f_c\}| \tag{3}$$

が得られる。この式の初項と第 2 項は、独立した探索空間

Algorithm 1: 索引番号 a の非同型な塗分けを求めるアルゴリズム

Input : 塗分けの集合 S_c, 台集合 $\{1,2,\ldots,n\}$ の上での自己同型群 $\mathrm{Aut}\,\Gamma$, 索引番号 a
Output: $\mathrm{Aut}\,\Gamma$ に関する a 番目の非同型な塗分け

1 S_i $(i=1,2,\ldots,|\mathrm{Aut}\,\Gamma|)$, $\mathcal{E}$ を求める
2 $\mathrm{total} := \frac{1}{|\mathrm{Aut}\,\Gamma|}\sum_{E_i\in\mathcal{E}}\sum_{\sigma_{i_j}\in E_i} q_i\,|S_c\cap S_{i_j}|$
3 **if** a **が** *1* **から** total **までの整数でない then**
4 　a 番目の塗分けが無いことを出力し、停止する
5 $T := S_c$
6 **for each** $E_i\in\mathcal{E}$ **do**
7 　$T' := T\cap\left(\bigcap_{\sigma_{i_j}\in E_i}\overline{S_{i_j}}\right)$ // E_i が×側の塗分け
8 　$\mathrm{subtotal} := \frac{1}{|\mathrm{Aut}\,\Gamma|}\sum_{E_{i'}\in\mathcal{E}}\sum_{\sigma_{i'_j}\in E_{i'}} q_{i'}\,|T'\cap S_{i'_j}|$ // $N(T',\mathrm{Aut}\,\Gamma)$ を求める
9 　**if** $a\le\mathrm{subtotal}$ **then**
10 　　$T := T'$ // a 番目の非同型な塗分けは、E_i が×側にある
11 　**else**
12 　　$T := T\setminus T'$ // a 番目の非同型な塗分けは、E_i が×側にない
13 　　$\mathrm{total} := \mathrm{total}-\mathrm{subtotal}$
14 　　$a := a-\mathrm{subtotal}$
15 T に非同型な塗分けの除去アルゴリズムを適用し、a 番目のものを出力する

$S_c\cap S_j$ および $S_c\cap\overline{S_j}$ について、非同型な塗分けの個数を求めている。したがって、初項により $S_c\cap S_j$ に属す非同型な塗分けの個数 $N(S_c\cap S_j)$ が求められれば、索引番号 a の塗分けが $S_c\cap S_j$ 側と $S_c\cap\overline{S_j}$ 側のどちらに属すのかが判定できる。これを、二分探索の要領で、各置換に対し繰り返し適用する。

また、$\sigma_{i_1}\approx_\theta\sigma_{i_2}$ が成り立つ 2 つの置換 σ_{i_1}, σ_{i_2} については補題 1 が、$\sigma_{i_1}\approx_\bigstar\sigma_{i_2}$ が成り立つ 2 つの置換 σ_{i_1}, σ_{i_2} については補題 3 が成り立つ。したがって、式 (2) を変形することで、

$$\frac{1}{|\mathrm{Aut}\,\Gamma|}\sum_{f_c\in S_c}\sum_{E_i\in\mathcal{E}} q_i\left|\{\sigma_{i_j}\in E_i\mid\sigma_{i_j}\circ f_c=f_c\}\right|$$

となる。さらに、S_i の定義より、

$$\sum_{f_c\in S_c}\left|\{\sigma_{i_j}\in E_i\mid\sigma_{i_j}\circ f_c=f_c\}\right| = \sum_{\sigma_{i_j}\in E_i}|S_c\cap S_{i_j}|$$

が成り立つので、式 (2) は

$$\frac{1}{|\mathrm{Aut}\,\Gamma|}\sum_{E_i\in\mathcal{E}}\sum_{\sigma_{i_j}\in E_i} q_i\,|S_c\cap S_{i_j}| \tag{4}$$

と変形することができる。

以上の議論をもとに、a 番目の非同型な塗分けを求めるアルゴリズムを、Algorithm 1 に示す。アルゴリズムの 2 行目では、式 (4) に基づいて、非同型な塗分けの総数を求める。5 行目で T に塗分けすべての集合 S_c を設定し、以降で二分探索を行う。7, 8 行目では、T の塗分けの内で E_i が×側になるもの T' に対し、非同型な塗分けの個数を subtotal として求めている。$a\le$ subtotal が成り立つ場合には、a 番目の非同型な塗分けは、E_i が×側すなわち T' にあることになる。以降の探索を続けるために、T の範囲を T' に絞る。そうでない場合には、a 番目の非同型な塗分けは、E_i が×側にはないことになる。この場合には、以降の探索を続けるために、T の範囲を $T\setminus T'$ に絞る。また、T' に含まれる非同型な塗分けは subtotal 個であるため、以降の探索では、$T\setminus T'$ から $a-$subtotal 番目の非同型な塗分けを見つける。14 行目までの二分探索が終わると、a 番目の非同型な塗分けが属すグループの塗分けが T に得られる。15 行目では、非同型な塗分けの除去アルゴリズム [10] を適用することで、T から非同型な塗分けを列挙し、a 番目のものを出力する。

以降では、Algorithm 1 を集合族の演算により実現する方法について述べる。ZDD は集合族の効率的な表現方法であり、集合族の演算を高速に実現することができるため、Algorithm 1 を効率的に実行できることになる。

塗分けの色数 c を、$c=2$ と仮定すると、$f_c:\Gamma\to\{1,2\}$ であるため、f_c は集合 $\{j\in\{1,2,\ldots,n\}\mid f_c(j)=1\}$ で表すことができ、塗分けの集合は $\{1,2,\ldots,n\}$ の部分集合の族として表すことができる。塗分け f_c と集合 $S\subseteq\{1,2,\ldots,n\}$ を同一視すると、置換 σ_i で動かない塗分けの集合 S_i は、$\{f_c\subseteq\{1,2,\ldots,n\}\mid\forall S\in\theta_{\sigma_i}\,\forall j_k\in S\ f_c(j_k)=f_c(j_1)\}$ で求めることができる。Algorithm 1 の 2, 7, 8, 12 行目の演算では、S_c, S_i, T, T' がすべて塗分けの集合であり、$\{1,2,\ldots,n\}$ の部分集合の族で表されているため、すべて集合族演算により実現できる。

4. 計算機実験

3 章で提案した Algorithm 1 を、ZDD を用いた集合演算により実現した。Intel(R) Core(TM) i7-3770K (3.50

表 3 数珠の塗分けの個数 (探索空間) と非同型な塗分けの個数

珠の個数 n	10	15	20	25	30	35	40
塗分け	1.0×10^3	3.3×10^4	1.0×10^6	3.4×10^7	1.1×10^9	3.4×10^{10}	1.1×10^{12}
非同型な塗分け	7.8×10^1	1.2×10^3	2.7×10^4	6.8×10^5	1.8×10^7	4.9×10^8	1.4×10^{10}

表 4 計算時間の比較 (秒)

n	10	15	20	25	30	35	40
非同型な塗分けの列挙 [10]	0.01	0.03	0.09	0.37	2.78	46.95	281.10
提案手法	0.03	0.05	0.17	0.40	4.80	18.64	220.15

表 5 メモリ使用量の比較 (MB)

n	10	15	20	25	30	35	40
非同型な塗分けの列挙 [10]	6.00	6.00	8.00	26.00	168.00	1,482.00	12,392.00
提案手法	32.06	33.00	34.00	49.00	212.85	1,466.00	6,902.93

GHz), 主記憶 64 GB の環境で実行する。実験では、n 個の珠を持つ数珠について、a 番目の非同型な塗分けを求めた。表 3 に、珠の個数 n に対する塗分けの個数 (すなわち探索空間の大きさ) と、非同型な塗分けの個数を示す。たとえば $n = 40$ の時には、1.1×10^{12} 通りの塗分けがある。これらの塗分けに対し、80 個の置換からなる自己同型群を考えるため、8.8×10^{13} 通りの可能性を考えて、1.4×10^{10} 種類の非同型な塗分けを得て、それらに非明示的に番号付けを行い、a 番目のものを取り出す問題であるとみなすことができる。

実験結果を表 4, 5 に示す。これらの表は、以下の 2 つの手法の比較である。(1) 提案手法。(2) 非同型な塗分けを除去して列挙するアルゴリズム [10] を利用して、非同型な塗分けすべてを ZDD として得た上で、a 番目のものを取り出す手法。もちろん、アルゴリズム [10] は、列挙アルゴリズムであり、本稿の問題に適用することは考慮されていないが、効率的な列挙アルゴリズムである。また、表 4 は計算時間、表 5 はメモリ使用量に関する比較である。各表の n は数珠の珠の個数であり、それぞれに対応する非同型な塗分けの個数を $N(2^{\{1,2,\ldots,n\}}, \mathrm{Aut}\,\Gamma)$ とすると、1 から $N(2^{\{1,2,\ldots,n\}}, \mathrm{Aut}\,\Gamma)$ までのランダムな数字を索引番号 a として与えることを 100 回繰り返した結果の平均である。アルゴリズム [10] を実行する上で問題になるのがメモリ使用量であり、その増加傾向は本手法の方が抑えられているのが見てとれる。

参考文献

[1] D. Avis and K. Fukuda, Reverse search for enumeration, *Discrete Applied Mathematics*, vol. 65, Issues 1-3, 1996, pp. 21–46.

[2] T. J. N. Brown, R. B. Mallion, P. Pollak, B. R. M. de Castro, and J. A. N. F. Gomes, The number of spanning trees in buckminsterfullerene, *Journal of Computational Chemistry*, vol. 12, pp. 1118–1124 (1991).

[3] R. E. Bryant, Graph-based algorithms for Boolean function manipulation, *IEEE Trans. Com.*, C-35: 677–691, 1986.

[4] F. Buekenhout, M. Parker, The number of nets of the regular convex polytopes in dimension ≤ 4, *Disc. Math.*, vol. 186, pp. 69–94, 1998.

[5] A. Burnside, *Theory of Groups of Finite Order*, Cambridge University Press, 1897.

[6] A. Cauchy, Mémoire sur diverses propriétés remarquables des substitutions régulières ou irrégulières, et des systémes de substitutiones conjugées, *Comptes Rendus Acad. Sci. Paris*, 21, 835, 1845.

[7] E. D. Demaine, J. O ' Rourk, Geometric Folding Algorithms: Linkages, Origami, Polyhedra, Cambridge University Press (2007).

[8] F. G. Frobenius, Uber die Congruenz nach einem aus zwei endlichen Gruppen gebildeten Doppelmodul, *J. reine angew. Math.*, 101, pp. 273–299, 1887

[9] C. Hippenmeyer, Die Anzahl der inkongruenten ebenen Netze eines regulären Ikosaeders, *Elem. Math.*, vol. 34, pp. 61–63, 1979.

[10] T. Horiyama, M. Miyasaka, and R. Sasaki, Isomorphism elimination by zero-suppressed binary decision diagrams, In *Proc. CCCG*, pp. 360–366, 2018.

[11] T. Horiyama and W. Shoji, Edge unfoldings of Platonic solids never overlap, In *Proc. CCCG*, pp. 65–70, 2011.

[12] T. Horiyama and W. Shoji, The number of different unfoldings of polyhedra, In *Proc. ISAAC*, LNCS 8283, pp. 623–633, 2013.

[13] M. Jeger, Über die Anzahl der inkongruenten ebenen Netze des Würfels und des regulären Oktaeders, *Elem. Math.*, 30, pp. 73–83, 1975.

[14] J. Kawahara, T. Inoue, H. Iwashita, and S. Minato, Frontier-based search for enumerating all constrained subgraphs with compressed representation, *IEICE Trans. on Fundamentals*, vol. E100-A, no. 9, pp. 1773–1784, 2017.

[15] G. Kirchhoff, Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme gefuhrt wird, *Annalen der Physik und Chemie*, 72, pp. 497–508, 1847.

[16] D. E. Knuth, The art of computer programming, vol. 4, fascicle 1, Bitwise tricks & techniques, binary decision diagrams, Addison-Wesley (2009).

[17] S. Minato. Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems. In Proc. DAC, pp. 272–277, 1993.

[18] G. Pólya, Kombinatorische Anzahlbestimmungen fur Gruppen, Graphen, und chemische Verbindungen, *Acta Math.*, 68, 145–254, 1937.

[19] K. Sekine, H. Imai, S. Tani, Computing the Tutte polynomial of a graph of moderate size, In *Proc. ISAAC*, LNCS 1004, pp. 224–233, 1995.