F-034

グラフ彩色インスタンス生成のための GA に基づく極小非可解構造の導出

Acquiring Minimal Unsolvable Structures Based on GA for Generating Graph Colorability Instances

水野 一徳 †　早川 大貴 †　佐々木 整 †　西原 清一 ‡
Kazunori Mizuno　Daiki Hayakawa　Hitoshi Sasaki　Seiichi Nishihara

1　はじめに

グラフ彩色問題（COL）は，制約充足問題の代表的な例題であり，解探索アルゴリズムの開発や計算量評価などの観点から研究されている．頂点に塗る色の数を 3 とした COL（3COL）は NP 完全であり，ヒューリスティクスの研究題材としても興味深く多くの研究報告がある．特に，その可解性を判別するのに非常に手間のかかる具体的問題（インスタンス）の存在する状況（相転移）や原因およびそれらインスタンスの生成法については，多くの有効なパラメータが提案されている．しかし，これらの研究におけるインスタンス生成は，そのほとんどが生成検査に基づくものであり，そのような困難なインスタンスを安定的に得ることは難しい．

これに対して，本研究では，3COL について，そのような難しいインスタンスを組織的に生成する方法を開発している [1]．本方法は，極小非可解構造を互いに繰り返し埋め込むことにより，任意に大きいかつ難しいインスタンスを安定的に生成するものである．本報告では，本方法にとって必須である極小非可解構造を遺伝的アルゴリズム（GA）を用いて導出する方法を提案する．また，そのインスタンスを解く計算量が問題サイズ（頂点数）の指数オーダになることを実験的に示す．

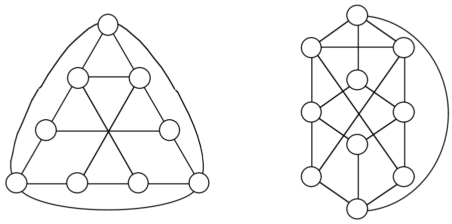
2　研究分野の概要

2.1　グラフ彩色問題と相転移

3COL とは，無向グラフの隣接する頂点が同じ色にならないように，すべての頂点に 3 色のうちを 1 色を塗り分ける問題である．近年，3COL を含む多くの組合せ探索問題において，解くのに非常に手間のかかる難しいインスタンスが問題空間の一部の領域に局所的に存在するということが明らかになりつつある（相転移現象）．本研究は，そのような難しいインスタンスを意図的かつ組織的に生成することを通して，それらのインスタンスの持つ構造条件を明らかにする一試みである．

2.2　極小非可解構造に基づく組織的生成

文献 [1] では，解くのが難しいインスタンスを組織的に生成する方法が提案されている．この方法では，図 1 のようなグラフを図 2 に示す手続きにより繰り返し埋め込むことにより，任意に大きいインスタンスを組織的に生成するものである．ここで，図 1 の構造は，極小非可解グラフ（MUG）と呼び，下記のような性質がある．



(a) MUG9 (n = 9, m = 16)　(b) MUG10 (n = 10, m = 18)

図 1　極小非可解グラフ（MUG）構造の例 [1]

procedure graph-generator(k)
begin
　input an initial graph G_{init};　(1)
　$G := G_{init}$;
　for $w := 1$ **to** k **do**
　　choose randomly an edge(i, j) ∈ $E(G)$ where deg(i)≤ 3;
　　choose randomly MUG$_{n,t}$　(2)
　　embed_MUG$_{nt}$(i, j);
　end for;
end.
procedure embed_MUG$_{nt}$(i, j)
begin
　choose randomly an edge(x, y) ∈ $E(MUG_{n,t})$ where deg(x)≤ 3;
　remove the edge (i, j)
　remove the edge (x, y);
　add an edge (j, y);
　merge x with i;
end.

図 2　MUG の埋め込みアルゴリズム [1]

性質 1　極小非可解構造（非可解であるが，任意の辺を 1 本削除すると可解になる構造）である．
性質 2　準正則（頂点の次数が 3 か 4 のみで，その分散が小さい）である．
性質 3　n4c 構造（4 頂点クリークから辺を 1 本削除した構造）を部分グラフとして含まない．

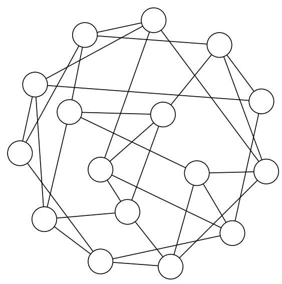
また，図 2 のアルゴリズムは，上記の 3 つの性質を保ったままグラフサイズを大きくすることができる．

3　GA を用いた極小非可解構造の導出

3.1　概要

文献 [1] の方法でインスタンスを生成するには，図 1 のような構造が重要であり，それらが多様なほどより複雑なインスタンスが生成できる，文献 [1] では，試行錯誤によってこのような構造を導出しており，より大きな

† 拓殖大学 工学部 情報工学科, Department of Computer Science, Takushoku University
‡ 筑波大学大学院コンピュータサイエンス専攻，Department of Computer Science, University of Tsukuba



頂点数 (n)=16
辺数 (m)=31
n_3=2, n_4=14
適応度 ($f(G)$)=4.0

図 3　導出された MUG 構造の例

サイズの MUG を導出することは困難となってしまう．そこで，本研究では，GA を用いることによりより多様な MUG 構造の導出を試みる．

3.2　コード化と適応度

ここでは，グラフ G を GA における 1 個体と考え，G の隣接行列（上三角成分）を 1 列に並べたものを染色体とする [2]．グラフ G を評価するための適応度 $F(G)$ は，前節の MUG 構造の性質を満たすように，それぞれを次式のように表現する．

$$F(G) = \sum_{i=1}^{4} \omega_i \times f_i(G) \tag{1}$$

$$f_1(G) = \begin{cases} 1 & G \text{ が性質 1 を満たす} \\ 0 & Otherwise \end{cases} \tag{2}$$

$$f_2(G) = \frac{n_3 + n_4}{n} \tag{3}$$

$$f_3(G) = \begin{cases} 1 & G \text{ が性質 3 を満たす} \\ 0 & Otherwise \end{cases} \tag{4}$$

$$f_4(G) = \begin{cases} 1 & var(G) \le 1 \\ 1/var(G) & var(G) > 1 \end{cases} \tag{5}$$

ここで，ω_i は関数 f_i に対する重み，n_j は次数 j の頂点数，$var(G)$ は G の頂点の次数の分散を表わしている．

3.3　導出結果

前節のコード化および適応度を用いて GA を実装し，文献 [1] では扱っていない頂点数 n=16 の MUG 構造の導出を試みた．ここでは，GA の世代交代モデルとして SGA，交叉方法として二点交叉を用いて，集団サイズ 20，世代数の上限は 2×10^{12} とした．図 3 は，その結果導出された n=16 の MUG 構造を表わしている．

4　評価実験

本方法によって導出された MUG 構造を用いて 3COL インスタンスを生成し実験を試みた．ここでは，図 3 の MUG を図 2 の (1),(2) に与えることによって，インスタンスを生成した．埋め込み回数（図 2 の k）を k=1~8 の 8 ケース（頂点数は n=31~136 となる）に対して，ケースごとに 100 個のインスタンスを生成した．また，生成したインスタンスを解くアルゴリズムは Smallk Coloring Program を用いた．

図 4，5 に，実験結果（それぞれ探索コストおよび探索時間）を示す．図 4，5 より，本方法で導出した MUG 構造を用いて生成したインスタンスを解く計算量が，頂点数のほぼ指数オーダとなっていることが分かる．

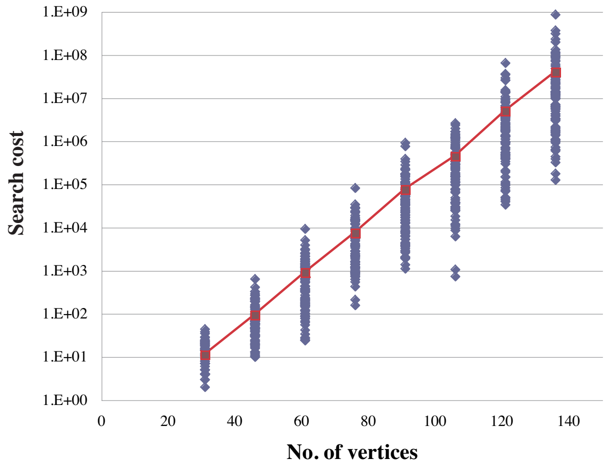


図 4　実験結果（探索コスト）

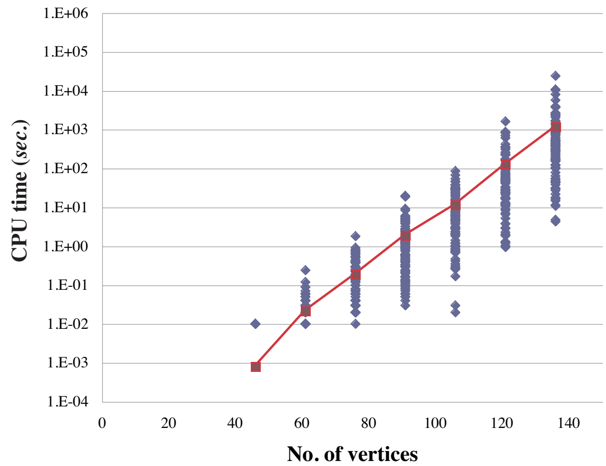


図 5　実験結果（CPU time）

5　おわりに

本報告では，解くのに手間のかかる難しい 3COL インスタンスを生成するために，重要な構造である MUG 構造を GA によって導出する方法を提案した．また，新たな MUG 構造を導出して，それを用いて生成されたインスタンスを解くことにより，その難しさが問題サイズのほぼ指数オーダとなることを実験で確認した．今後の課題としては，より大きなサイズの MUG 構造導出のための，適応度関数の改良，および GA による処理の高速化などがあげられる．

参考文献

[1] Mizuno, K. and Nishihara, S.: Constructive generation of very hard 3-Colorability Instances, *Discrete Applied Mathematics*, No. 156 (2), pp. 218–229 (2008).

[2] 花田良子, 佐藤史隆, 廣安知之, 三木光範, 鈴木泰博: 遺伝的アルゴリズムによるネットワーク特性量に着目したネットワーク設計法, コンピュータソフトウェア, Vol. 24, No.1, pp. 91–100 (2007).