A-006

グラフ彩色問題における解構築法の効率化
Speeding-up of Constructive Algorithms for the Graph Coloring Problem

金原 一歩[1)] 片山 謙吾[1)] 富田 悦次[2)] 岡野 傑士[1)] 三宅 孝史[1)] 西原 典孝[1)]
Kazuho Kanahara Kengo Katayama Etsuji Tomita Takeshi Okano Takahumi Miyake Noritaka Nishihara

1 まえがき

実用上重要な応用を有する組合せ最適化問題の一つであるグラフ彩色問題 (Graph Coloring Problem, GCP) に対する代表的な解構築法として DSATUR[1] と RLF[2] がよく知られている．DSATUR，RLF 共に彩色する頂点の選択を行う際に，部分グラフ内の次数を更新する必要があり，与えられたグラフの辺密度が高いほど処理に時間がかかるという問題点がある．本論文では，その問題点を改善する次数更新方式を提案し，提案方式を導入した DSATUR，RLF と従来法との比較実験により，提案法の有効性を示す．

2 グラフ彩色問題

グラフ彩色問題 (GCP) とは，V を頂点の集合，E を頂点間の枝の集合とする無向グラフ $G = (V, E)$ が与えられた時，すべての隣接する頂点同士が同色にならないように彩色可能な最小の色数を求める問題である．

3 DSATUR

DSATUR は飽和次数 (degree of saturation) が最大の頂点を逐次的に頂点を彩色する解構築法である．

以下，DSATUR のアルゴリズムについて述べる．図 1 に DSATUR の擬似コードを示す．図 1 における $C(v)$ は頂点 v の彩色番号，S は彩色済みの頂点集合，U は未彩色の頂点集合である．また，$deg_{G(U)}(v)$ は未彩色部分グラフ $G(U)$ に対する頂点 $v \in U$ の次数, $dsatur_{G(S)}(v)$ は彩色済み部分グラフ $G(S)$ に対する頂点 $v \in U$ に隣接している異なる色の数であり，これを飽和次数とよぶ．

まず，与えられたグラフ G から次数が最大の頂点 v を選択し，選択した頂点 v を最小の色番号である色 1 で彩色を行い，彩色済み部分グラフ $G(S)$ 対する飽和次数及び未彩色部分グラフ $G(U)$ の次数の更新をする (Lines 2–4)．次に，彩色済み部分グラフ $G(S)$ に対する飽和次数 $dsatur_{G(S)}(v)$ が最大の頂点 v を選び，飽和次数が最大の頂点が複数あった場合は未彩色部分グラフ $G(U)$ 内の次数 $deg_{G(U)}(v)$ が最大の頂点 v をランダムに選択する (Lines 6–8)．そして，選択した頂点 v を彩色可能な最小の色 k で彩色し，彩色済み部分グラフ $G(S)$ に対する飽和次数 $dsatur_{G(S)}(v)$ 及び未彩色部分グラフ $G(U)$ の次数 $deg_{G(U)}(v)$ を更新する (Lines 9–10)．この一連の処理を未彩色頂点集合 U の頂点がなくなるまで繰り返す (Lines 5–12)．

4 Recursive Largest First(RLF)

RLF は，彩色されてない頂点集合内のすべての頂点同士が隣接していない集合（独立集合）を探索し，彩色する処理を繰り返し行う解構築法である．

図 2 に RLF の処理手順の擬似コードを示す．色 k で彩色可能な集合 U' 内の次数 $deg_{G(U')}(v)$ が最大の頂点 v を選択し彩色を行い，集合 U' から頂点 v に隣接している頂点集合 $\Gamma(v)$ を削除し，未彩色部分グラフ $G(U)$ に対する頂点 $v \in U$ 及び部分グラフ $G(U')$ に対する頂点 $v \in U'$ の次数更新を行う (Lines 4–6)．そして，未彩色部分グラフ $G(U)$ に対する次数が最大の頂点 $v \in U'$ を選び，選択した頂点 v が複数あった場合は部分グラフ $G(U')$ 内の次数 $deg_{G(U')}(v)$ が最大の頂点 v を選択し彩色する (Lines 9–11)．そして，色 k で彩色可能な頂点集合 U' の頂点がなくなった場合は，再度色 $k+1$ で彩色する (Lines 7–14)．この一連の処理を未彩色頂点集合 U の頂点がなくなるまで繰り返す (Lines 2–15)．

```
    procedure DSATUR Algorithm
    input: graph G = (V, E);
    output: vertices color C;
    begin
1     S := ∅; U := V; compute deg_G(U);
2     select one vertex v randomly with max_{v∈U} {deg_G(U)(v)};
3     C(v) := 1; S := S ∪ {v}; U := U\{v};
4     update dsatur_G(S) and deg_G(U);
5     repeat
6       find a vertex v with max_{v∈U} {dsatur_G(S)(v)};
7       if a subset U' of multiple vertices
                with the same max degree of saturation are found
8       then select one vertex v randomly with max_{v∈U'} {deg_G(U)(v)};
9       find the least possible color k that can color the selected vertex v;
10      C(v) := k; S := S ∪ {v}; U := U\{v};
11      update dsatur_G(S) and deg_G(U);
12    until U = ∅;
    end
```

図 1 DSATUR の擬似コード

```
    procedure Recursive Largest First Algorithm
    input: graph G = (V, E);
    output: vertices color C;
    begin
1     k := 1; S := ∅; U := V; compute deg_G(U);
2     repeat
3       U' := U; deg_G(U') = deg_G(U);
4       select one vertex v randomly with max_{v∈U'} {deg_G(U')(v)};
5       C(v) := k; S := S ∪ {v}; U := U\{v}; U' := U'\{v};
6       U' := U'\Γ(v); update deg_G(U) and deg_G(U');
7       repeat
8         find a vertex v with max_{v∈U'} {deg_G(U)(v) − deg_G(U')(v)};
9         if a subset U'' of multiple vertices
                  with the same max degree difference are found
10        then select one vertex v randomly with max_{v∈U''} {deg_G(U')(v)};
11        C(v) := k; S := S ∪ {v}; U := U\{v}; U' := U'\{v};
12        U' := U'\Γ(v); update deg_G(U) and deg_G(U');
13      until U' = ∅;
14      k := k + 1;
15    until U = ∅;
    end
```

図 2 Recursive Largest First の擬似コード

5 提案部分グラフ次数更新法

5.1 標準的な部分グラフ次数更新法

DSATUR，RLF 共に頂点を彩色するたびに部分グラフ内の頂点の次数を更新する処理が必要である．ループ対象の集合が異なる 2 種類の標準的な次数更新法 A(図 3) 及び，B(図 4) では，部分集合 S 内の頂点 v_d を削除するたびに隣接する頂点 $v \in S$(図 3)，$v_\gamma \in \Gamma(v_d)$ の次数を-1 する処理を行う．

```
    procedure Standard Degree Update Method A
    input: graph G(V, E), subset S, subset degree deg_G(S), drop vertex v_d;
    begin
1     for each v ∈ S do
2       if (v_d, v) ∈ E then deg_G(S)(v) := deg_G(S)(v) − 1;
3     endfor
    end
```

図 3 基本的な次数更新法 A の擬似コード

1） 岡山理科大学
2） 電気通信大学

```
    procedure Standard Degree Update Method B
    input: graph G(V, E), subset S, subset degree deg_{G(S)}, drop vertex v_d;
    begin
1     for each v_γ ∈ Γ(v_d) do
2       if v_γ ∈ S then deg_{G(S)}(v_γ) := deg_{G(S)}(v_γ) − 1;
3     endfor
    end
```

図 4　基本的な次数更新法 B の擬似コード

5.2　改良部分グラフ次数更新法

上述の標準的な次数更新法では，与えられたグラフの辺密度 (頂点に隣接する辺の割合) が高いほど処理時間を要するという問題点がある．そこで，密なグラフに対しても高速に次数更新を行うために（以下に述べる）次数更新プロセスを適応的に切り替える方法を提案する．

標準的な次数更新法では，部分集合 S から複数の頂点が削除される場合，頂点を削除するたびに更新が行なわれるため非効率である．そこで提案法では，部分集合 S から複数の頂点 D を削除したあとに部分集合 S 内の頂点 $v \in S$ の次数 $deg_{G(S)}(v)$ を更新する．提案法の主要ループでは，3 つの処理 (Lines 2, 6, 10) を適応的に切り替え更新する．それら 3 つの処理として，更新法 A の処理 (Lines 3–5) と更新法 B の処理 (Lines 7–9) に加えて 3 つ目の処理として，頂点 v に隣接していない集合 $\Lambda(v)$ を基準に Lines 11–14 の次数更新プロセスを導入する．3 つ目の更新プロセスでは，部分集合 S 内の削除された頂点 v_d に隣接しない頂点 $v \in S \cap \Lambda(v_d)$ の次数を+1 することにより次数の更新を行う．ただし，3 つ目の更新プロセスを行った場合は集合 S 内のすべての頂点の $deg_{G(S)}$ が実際の次数より 1 大きい状態になるため，実際の次数よりどの程度大きいか保持するためにバイアス値 ($bias$) を+1 する (Line 11)．もし，ある頂点 v_i の実際の次数を求める場合は $deg_{G(S)}(v_i) - bias$ とする．なお，2 つの頂点 v_i, v_j の次数を比較する場合は，

$$deg_{G(S)}(v_i) - bias < deg_{G(S)}(v_j) - bias \qquad (1)$$

となる．つまり，$deg_{G(S)}(v_i) < deg_{G(S)}(v_j)$ となるため，バイアス値を考慮する必要はない．

6　実験結果

提案した改良部分グラフ次数更新法を評価するために，提案方式を導入した DSATUR，RLF と従来法との比較実験を行った．対象とするグラフは，GCP の標準的なグラフ例題を集めた DIMACS ベンチマークグラフ (119 例題) と，ランダムグラフの頂点数 n を 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000, 3000, 4000, 5000, 10000, 15000, 20000 の 17 種，辺密度 ρ を 0.05 から 0.95

```
    procedure Fast Degree Update Method
    input: graph G(V, E), subset S, subset degree deg_{G(S)},
           drop vertices D, bias value bias;
    begin
1     for each v_d ∈ D do
2       if |S| ≤ min(|Γ(v_d)|, |Λ(v_d)|) then
3         for each v ∈ S do
4           if (v_d, v) ∈ E then deg_{G(S)}(v) := deg_{G(S)}(v) − 1
5         endfor
6       else if |Γ(v_d)| < min(|S|, |Λ(v_d)|) then
7         for each v_γ ∈ Γ(v_d) do
8           if v_γ ∈ S then deg_{G(S)}(v_γ) := deg_{G(S)}(v_γ) − 1;
9         endfor
10      else  //|Λ(v_d)| < min(|S|, |Γ(v_d)|)
11        bias := bias + 1;
12        for each v_λ ∈ Λ(v_d) do
13          if v_λ ∈ S then deg_{G(S)}(v_λ) := deg_{G(S)}(v_λ) + 1;
14        endfor
15      endif
16    endfor
    end
```

図 5　改良次数更新法の擬似コード

表 1　提案 RLF と従来 RLF の実験結果

instance			DSATUR		RLF	
			提案法	従来法	提案法	従来法
Name	n	ρ	Time(s)	Time(s)	Time(s)	Time(s)
latin_square_10	900	0.76	**0.005**	0.019	**0.019**	0.022
DSJC1000.5	1000	0.50	**0.007**	0.019	0.053	**0.021**
DSJC1000.9	1000	0.90	**0.010**	0.030	**0.019**	0.062
4-FullIns_5	4146	0.01	**0.026**	0.040	**0.018**	0.024
qg.order100	10000	0.02	**0.155**	0.246	0.406	**0.233**

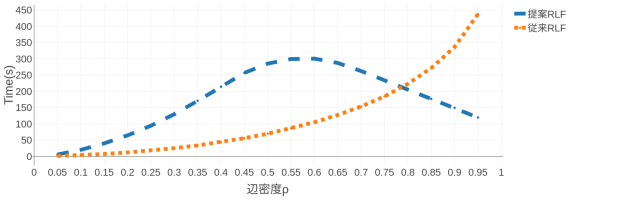

図 6　提案 *RLF* と従来 *RLF* の実行時間の比較

までを 0.05 区切りとしたの 19 種の組合せ 323 種を 10 種類の乱数シードのもとでそれぞれ作成した 3230 例題の合計 3349 例題とした．試行回数は各問題例に対して 10 回とし，各アルゴリズムは，C++によってコード化し，コンパイラは最適化オプション-O3 を付与した g++（Ver. 8.3.0）である．全ての計算は，計算機（CPU: Intel Core i7 3.6GHz, RAM: 22.4GiB）上で実行した．

表 1 に DIMACS ベンチマークグラフから比較的大規模な 5 例題のみ，提案法と従来法の実験結果を示す．表 1 の左の欄から，問題例名 Name，総頂点数 n，辺密度 ρ，DSATUR，RLF の提案法，従来法それぞれでの 1 試行あたりの平均実行時間 Time(s) を示す．なお，表中の太字は各解法との比較においてより良好な結果であることを示す．図 6 は，生成したランダムグラフの中の最大規模の頂点数 20000 のグラフに対する提案 RLF と従来 RLF の実行時間の比較グラフである．比較的大規模で辺密度 0.9 である DSJC1000.9 のグラフにおいて提案 RLF は，従来 RLF と比べて約 3.26 倍の高速化に成功している．頂点数 20000 のランダムグラフに対しても辺密度 ρ が 0.8 以上の高密度なグラフに対して従来法より高速に解を算出している．これらの結果より，提案法は，従来法に比べ高密度なグラフに対して有効であることを示した．なお，実験で使用したグラフ例題においても提案法は全 3349 例題中 DSATUR では 3337(99%) 例題，RLF では 1358(41%) 例題において従来法と比べ同等以上の結果であることを観測している．このように多くのグラフ例題において良好な結果が得られていることから，提案法は従来法に比べ高密度なグラフに対して高速な次数更新法であると考えられる．

7　むすび

本論文では，グラフ彩色問題に対する代表的な解構築法である DSATUR，RLF に改良部分グラフ次数更新法を導入することにより，高速化を行った．実験結果より提案法は，従来法と比べ高密度なグラフに対して有効であることを示した．今後の課題として，最大クリーク問題をはじめとするグラフ問題に対して提案法の利用可能性の検討が考えられる．

謝辞

本研究の一部は JSPS 科研費（基盤研究（C）19K12166）の助成を受けたものである．

参考文献

[1] D. Brélaz. New methods to color vertices of a graph. *Commun. ACM*, Vol. 22, pp. 251–256, 1979.

[2] F.T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, Vol. 84, , 11 1979.