Permutation-Based Optimization Method for Solving Graph Coloring Problems

วิธีหาค่าเหมาะสมที่สุดเชิงการเรียงสับเปลี่ยนสำหรับการแก้ปัญหาการให้สีกราฟ

Benyapa Wadsungnoen (เบญญาภา วาดสูงเนิน)* Dr.Pikul Puphasuk (ดร.พิกุล ภูผาสุข)**

ABSTRACT

In this research, we propose a permutation-based optimization method for solving the Graph coloring problems (GCP). The proposed method uses three heuristic methods: Greedy method, Welsh-Powell method and Dsatur method to color the vertices for a given permutation of vertices. We test the performance of the proposed method with 25 graphs selected from DIMACS benchmark. The results show that the proposed method can find the minimum numbers of colors as reported in the previous works for all cases, which indicates that it is effective for GCP.

บทคัดย่อ

งานวิจัยนี้นำเสนอวิธีหาค่าเหมาะสมที่สุดเชิงการเรียงสับเปลี่ยนสำหรับการแก้ปัญหาการให้สีกราฟ วิธีที่นำเสนอใช้วิธีเชิงฮิวริสติก 3 วิธี คือ วิธี Greedy วิธี Welsh-Powell และวิธี Dsatur ในการลงสีให้กับจุดตามลำดับของการเรียงสับเปลี่ยน และทดสอบประสิทธิภาพของวิธีที่นำเสนอกับกราฟทดสอบจำนวน 25 กราฟที่เลือกมาจากกราฟมาตรฐาน DIMACS ผลการทดลองแสดงว่าวิธีที่นำเสนอสามารถหาค่าสีน้อยสุดซึ่งมีค่าเท่ากับค่าสีที่ได้นำเสนอไว้ในงานวิจัยก่อนหน้าได้ทุกกรณีและเป็นวิธีที่มีประสิทธิภาพสำหรับปัญหาการให้สีกราฟ

Keywords: Vertex graph coloring problems, Permutation-based optimization methods, Heuristic methods

คำสำคัญ: ปัญหาการให้สีกราฟ การหาค่าเหมาะสมที่สุดเชิงการเรียงสับเปลี่ยน วิธีเชิงฮิวริสติก

*Student, Master of Science Program in Applied Mathematics, Department of Mathematics, Faculty of Science, Khon Kaen University

**Assistant Professor, Department of Mathematics, Faculty of Science, Khon Kaen University

Introduction

The graph coloring problem (GCP) is finding the least number of colors needed for coloring of graph such that no two adjacent vertices use the same color. Let $G = (V, E)$ be an undirected graph with a vertex set V and an edge set E. A function $c : V \rightarrow \{1, \ldots, k\}$ such that $c(u) \neq c(v)$ whenever $(u, v) \in E$ is called k-coloring. The graph G is k-colorable if it has a k-coloring. The minimum number of colors k needed to color the graph G is called *chromatic number* denoted by $\chi(G)$. There are many applications of GCP such as scheduling (Leighton, 1979), timetabling (Werra, 1985) and communication networks (Woo et al., 2002). Since the GCP is NP-hard, it is difficult to find the $\chi(G)$ for any given G. There are some exact algorithms proposed for GCP but these algorithms are able to solve consistently only small randomly generated instances, with up to 80 vertices (Malaguti, Toth, 2010). In real world applications, graphs often contain hundreds or thousands of vertices, for which the use of heuristic and metaheuristic techniques is necessary.

The well-known heuristic methods for solving the GCP are Greedy algorithm, Welsh-Powell algorithm (Welsh, Powell, 1967), and Degree of Saturation algorithm (Bre´laz, 1979). The Greedy algorithm is very fast but the results are sensitive to the order of vertices. Welsh-Powell method can provide a good solution, but it may not always be the best solution. The Degree of Saturation method (Dsatur) is more complicated than Greedy algorithm and Welsh-Powell algorithm but generally can give less number of used colors.

For solving the GCP by metaheuristic methods, tabu search procedure, called TABUCOL algorithm (Hertz, Werra, 1987) was successfully improved and embedded as a procedure in more complex algorithms. A new genetic local search algorithm for the GCP was proposed (Dorne, Hao, 1998). This algorithm combines the crossover based on the notion of Union of Independent Sets (UISX) and tabu search. The Hybrid Coloring Algorithm (HCA) for the GCP was presented (Galinier, Hao, 1999). It is the combining of an improved version of TABUCOL with a Greedy Partitioning Crossover operator (GPX).

In this study, we present the simple permutation-based optimization method for solving the GCP. The heuristic methods used in this work are Greedy method, Welsh-Powell method and Dsatur method. Since these heuristic methods are sensitive to the order of vertices, we propose a new crossover operator to construct the suitable permutations of vertices.

Objectives of the study

The aim of this work is to propose a permutation-based optimization method for solving the GCP. The proposed method uses three heuristic methods: Greedy method, Welsh-Powell method and Dsatur method to color the vertices for a given permutation of vertices. We test the performance of the proposed method with 25 graphs selected from DIMACS benchmark.

Methodology

1. Heuristic methods for the GCP

Let $G = (V, E)$, where $V = \{1,2, \dots, n\}$ is the vertex set and $E \subseteq V \times V$ is the edge set. The heuristic methods for coloring the vertices of a given order consist of Greedy method, Welsh-Powell method and Dsatur method.

1.1 Greedy method

The vertices are labelled $v_1, \dots, v_n$. Vertex v_1 is assigned to the first color class, and thereafter vertex v_i, $i = 2, \dots, n$ is assigned to the lowest indexed color class that contains no vertices adjacent to v_i. The algorithm can be described as follows:

Step 1. Assign the first color $c_1 = 1$ to the first vertex v_1.

Step 2. The vertex v_2 is assigned color c_1 if it is not adjacent to v_1; otherwise it gets assigned color $c_2 = 2$.

Step $i = 3,4, \dots, n$, the vertex v_i is assigned the least possible color not already used by all the neighbors of v_i.

1.2 Welsh-Powell method

For Welsh-Powell method, the vertices of a high degree are colored in first priority. The algorithm is described as follows:

Step 1. Arrange the vertices by the decreasing order of degrees.

Step 2. Color the first vertex in the list (the vertex with the maximum degree) with color 1.

Step 3. Go down the list and color every vertex not connected to the colored vertices above the same color. Then cross out all colored vertices in the list.

Step 4. Repeat the process on the uncolored vertices with a new color.

1.3 Dsatur method

The Dsatur method is based on the idea of reordering the vertices at each stage. The degree of saturation of a vertex is the number of different colors used for its neighbors in the current solution. The Dsatur method is a sequential algorithm in which vertices are chosen based on the degree of saturation. A vertex with maximum saturation degree is given the priority to be placed in the first legal color class. The algorithm can be described as follows:

Step 1. Arrange the vertices by the decreasing order of degrees.

Step 2. Color a vertex of maximum degree with color 1.

Step 3. Choose a vertex with a maximal saturation degree. If there is an equality, choose any vertex of maximal degree in the uncolored subgraph.

Step 4. Color the chosen vertex with the least possible color.

Step 5. Return to step 3 until all vertices are colored.

2. Proposed permutation-based optimization method

Since the heuristic methods are sensitive to the order of vertices, we propose a new crossover operator to construct the suitable permutations of vertices. The proposed method is described as follows.

Initialization

1. Random N permutations of n vertices and color vertices based on these permutations by the three heuristic methods.

2. Select NP permutations from (1) which give the minimum numbers of colors to be initial permutations.

Crossover

For permutation $P_i^{(t)} = (P_{i1}^{(t)}, P_{i2}^{(t)}, \dots, P_{in}^{(t)})$ where $i = 1,2, \dots, NP$,

1. Random permutation $P_j^{(t)}$ where $j \in \{1,2, \dots, NP\}$ and $j \neq i$.

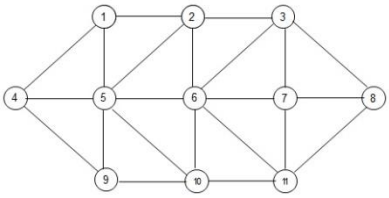
2. Find a position s of $P_i^{(t)}$ such that $c(P_{is}^{(t)}) = \max_{1 \le r \le n}\{c(P_{ir}^{(t)})\}$ and set $k = P_{is}^{(t)}$.

3. Find the position l of $P_j^{(t)}$ such that $P_{jl}^{(t)} = k$.

4. Construct $P_i^{*\,(t)}$ from $P_i^{(t)}$ by switching $P_{is}^{(t)}$ and $P_{il}^{(t)}$.

5. Color the vertices by using $P_i^{*\,(t)}$ for the three heuristic methods.

The example of crossover operator using a Greedy method is presented as follows. For a given graph G,



1. For a permutation P_1, random a permutation P_2 and color vertices of graph G based on both permutations.

P_1 : 3 2 4 5 6 1 7 8 9 10 11 $\qquad$ P_2 : 4 5 1 2 6 3 7 9 11 10 8

P_1	3	2	4	5	6	1	7	8	9	10	11
C	1	2	1	3	4	4	2	3	2	1	5

P_2	4	5	1	2	6	3	7	9	11	10	8
C	1	2	3	1	3	2	1	3	2	1	3

2. Find a position s of P_1 which gives a maximum color. In this case $s = 11$.

P_1 : 3 2 4 5 6 1 7 8 9 10 11 $\qquad$ P_2 : 4 5 1 2 6 3 7 9 11 10 8

P_1	3	2	4	5	6	1	7	8	9	10	11
C	1	2	1	3	4	4	2	3	2	1	5

max

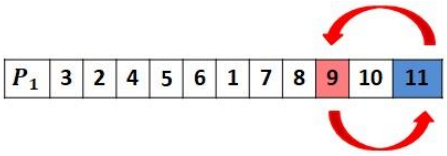
P_2	4	5	1	2	6	3	7	9	11	10	8
C	1	2	3	1	3	2	1	3	2	1	3

3. Find the position l of P_2 such that $P_{2l} = 11$. In this case $l = 9$.

P_1	3	2	4	5	6	1	7	8	9	10	11

P_2	4	5	1	2	6	3	7	9	11	10	8

4. Construct P_1^* from P_1 by switching $P_{1,11}$ and $P_{1,9}$.



P_1	3	2	4	5	6	1	7	8	9	10	11

5. Color the vertices of graph G by using P_1^*. In this case P_1^* gives four colors whereas P_1 gives five colors. This shows the efficiency of algorithm.

P_1^*	3	2	4	5	6	1	7	8	11	10	9
C	1	2	1	3	4	4	2	3	1	2	4

Selection

Compare the numbers of colors obtained from $P_i^{*\,(t)}$ and $P_i^{\,(t)}$ for selecting $P_i^{(t+1)}$ in the next generation by

$$P_i^{(t+1)} = \begin{cases} P_i^{*\,(t)} & if \quad \min\limits_{1\le r\le n}\{c(P_{ir}^{*\,(t)})\} \le \min\limits_{1\le r\le n}\left\{c\left(P_{ir}^{(t)}\right)\right\} \\ P_i^{\,(t)} & if \quad Otherwise. \end{cases}$$

3. Classification of the benchmark problems

We classify the 25 tested graphs selected from DIMACS benchmark into two groups by using the total successful percentages of three heuristic methods on 100 random permutations. The graph that the total successful percentages greater or equal than 70% is contained in the first group, otherwise it is contained in the second group.

4. Parameter setting

We set parameters for the proposed algorithm as follows.

1. First group : $N = 10, NP = 5, NG = 1$.

2. Second group : $N = 50, NP = 25, NG = 5$ where NG is the maximum number of generations.

3. Each tested graph is performed 20 runs.

Results and Discussion

The 25 tested graphs are classified into two groups. The first group and the second group of graphs are shown in Table 1 and Table 2, respectively. Those tables show maximum (Max) and minimum (Min) numbers of colors and the number of successful runs (NS). From Table 1, there are 15 graphs with the vertex range and edge range of 37-561 and 72-5198, respectively. The total successful percentages are between 70% - 98%. The Greedy method gives lowest NS for all cases. From Table 2, there are 10 graphs with the vertex range and edge range of 25-1085 and 160-12201, respectively. The total successful percentages are between 21.67% - 68.33%.

Table 1 The graphs of first group

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Successful Percentages
1	homer	561	1629	13	Greedy	16	13	31	77.00
					Welsh-Powell	13	13	100	
					Dsatur	13	13	100	
2	miles1500	128	5198	73	Greedy	77	73	26	75.33
					Welsh-Powell	73	73	100	
					Dsatur	73	73	100	
3	myciel5	47	236	6	Greedy	7	6	84	85.00
					Welsh-Powell	6	6	100	
					Dsatur	7	6	71	
4	myciel7	191	2360	8	Greedy	10	8	55	82.00
					Welsh-Powell	8	8	100	
					Dsatur	9	8	91	
5	1-Insertions_4	67	232	5	Greedy	6	5	82	93.67
					Welsh-Powell	5	5	100	
					Dsatur	6	5	99	
6	1-Insertions_5	202	1227	6	Greedy	8	6	30	75.67
					Welsh-Powell	6	6	100	
					Dsatur	7	6	97	
7	2-Insertions_3	37	72	4	Greedy	5	4	95	98.00
					Welsh-Powell	5	4	99	
					Dsatur	4	4	100	
8	2-Insertions_4	149	541	5	Greedy	7	5	64	80.00
					Welsh-Powell	6	5	91	
					Dsatur	6	5	85	
9	3-Insertions_3	56	110	4	Greedy	5	4	93	97.33
					Welsh-Powell	5	4	99	
					Dsatur	4	4	100	
10	3-Insertions_4	281	1046	5	Greedy	6	5	39	70.33
					Welsh-Powell	6	5	81	
					Dsatur	6	5	91	

Table 1 The graphs of first group (Cont.)

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Successful Percentages
11	4-Insertions_3	79	156	4	Greedy	5	4	93	96.00
					Welsh-Powell	5	4	95	
					Dsatur	4	4	100	
12	4-Insertions_4	475	1795	5	Greedy	7	5	39	70.00
					Welsh-Powell	6	5	83	
					Dsatur	6	5	88	
13	1-FullIns_4	93	593	5	Greedy	7	5	20	70.67
					Welsh-Powell	5	5	100	
					Dsatur	6	5	92	
14	2-FullIns_4	212	1621	6	Greedy	10	6	26	71.00
					Welsh-Powell	7	6	99	
					Dsatur	7	6	88	
15	3-FullIns_3	80	346	6	Greedy	8	6	52	83.67
					Welsh-Powell	7	6	99	
					Dsatur	6	6	100	

Table 2 The graphs of second group

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Successful Percentages
1	miles250	128	387	8	Greedy	11	8	3	59.67
					Welsh-Powell	9	8	76	
					Dsatur	8	8	100	
2	miles500	128	1170	20	Greedy	24	20	5	68.33
					Welsh-Powell	20	20	100	
					Dsatur	20	20	100	
3	queen5_5	25	160	5	Greedy	9	5	3	21.67
					Welsh-Powell	8	5	55	
					Dsatur	8	5	7	
4	1-Insertions_6	607	6337	7	Greedy	10	7	2	64.33
					Welsh-Powell	8	7	92	
					Dsatur	8	7	99	

Table 2 The graphs of second group (Cont.)

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Successful Percentages
5	2-Insertions_5	597	3936	6	Greedy	8	6	3	50.33
					Welsh-Powell	7	6	71	
					Dsatur	7	6	77	
6	1-FullIns_5	282	3247	6	Greedy	10	6	2	57.00
					Welsh-Powell	6	6	100	
					Dsatur	8	6	69	
7	2-FullIns_5	852	12201	7	Greedy	12	8	0	44.67
					Welsh-Powell	8	7	96	
					Dsatur	9	7	38	
8	3-FullIns_4	405	3524	7	Greedy	10	7	4	65.00
					Welsh-Powell	8	7	96	
					Dsatur	8	7	95	
9	4-FullIns_4	690	6650	8	Greedy	11	8	2	66.33
					Welsh-Powell	9	8	97	
					Dsatur	8	8	100	
10	5-FullIns_4	1085	11395	9	Greedy	13	10	0	65.00
					Welsh-Powell	10	9	95	
					Dsatur	9	9	100	
					Dsatur	9	9	100	

The performances of the proposed method using three heuristic methods for first and second groups are shown in Table 3 and Table 4, respectively. The tables show Max, Min, NS and CPU times. The results show that the proposed method can find the minimum numbers of colors as reported in the previous works for all cases. From Table 3, the Welsh-Powell method gives the minimum CPU times for 11 cases whereas Greedy method gives the minimum CPU times for 4 cases. For graphs in second group, Welsh-Powell and Greedy methods give the minimum CPU times for 6 cases and 4 cases, respectively as shown in Table 4.

Table 3 The performances of the proposed method of first group

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Times (s)
1	homer	561	1629	13	Greedy	13	13	20	37.2913
					Welsh-Powell	13	13	20	32.7894
					Dsatur	13	13	20	102.3325
2	miles1500	128	5198	73	Greedy	73	73	20	3.2152
					Welsh-Powell	73	73	20	8.9826
					Dsatur	73	73	20	9.3432
3	myciel5	47	236	6	Greedy	6	6	20	0.3921
					Welsh-Powell	6	6	20	0.4164
					Dsatur	6	6	20	0.9754
4	myciel7	191	2360	8	Greedy	8	8	20	4.9123
					Welsh-Powell	8	8	20	4.2783
					Dsatur	8	8	20	11.7605
5	1-Insertions_4	67	232	5	Greedy	5	5	20	0.7938
					Welsh-Powell	5	5	20	0.7446
					Dsatur	5	5	20	1.6583
6	1-Insertions_5	202	1227	6	Greedy	6	6	20	5.2368
					Welsh-Powell	6	6	20	4.8671
					Dsatur	6	6	20	13.5397
7	2-Insertions_3	37	72	4	Greedy	4	4	20	0.2360
					Welsh-Powell	4	4	20	0.2282
					Dsatur	4	4	20	0.7209
8	2-Insertions_4	149	541	5	Greedy	5	5	20	2.9688
					Welsh-Powell	5	5	20	2.8296
					Dsatur	5	5	20	7.2125
9	3-Insertions_3	56	110	4	Greedy	4	4	20	0.4731
					Welsh-Powell	4	4	20	0.5882
					Dsatur	4	4	20	1.2460
10	3-Insertions_4	281	1046	5	Greedy	5	5	20	10.8396
					Welsh-Powell	5	5	20	9.0805
					Dsatur	5	5	20	24.2780
11	4-Insertions_3	79	156	4	Greedy	4	4	20	1.0079
					Welsh-Powell	4	4	20	0.8992
					Dsatur	4	4	20	2.2470

Table 3 The performances of the proposed method of first group (Cont.)

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Times (s)
12	4-Insertions_4	475	1795	5	Greedy	5	5	20	31.7937
					Welsh-Powell	5	5	20	25.5394
					Dsatur	5	5	20	72.7382
13	1-FullIns_4	93	593	5	Greedy	5	5	20	1.2320
					Welsh-Powell	5	5	20	1.2799
					Dsatur	5	5	20	3.1820
14	2-FullIns_4	212	1621	6	Greedy	6	6	20	5.8928
					Welsh-Powell	6	6	20	5.0096
					Dsatur	6	6	20	14.5669
15	3-FullIns_3	80	346	6	Greedy	6	6	20	0.9896
					Welsh-Powell	6	6	20	0.8861
					Dsatur	6	6	20	2.4399

Table 4 The performances of the proposed method of second group

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Times (s)
1	miles250	128	387	8	Greedy	8	8	20	30.5319
					Welsh-Powell	8	8	20	31.4164
					Dsatur	8	8	20	76.0324
2	miles500	128	1170	20	Greedy	20	20	20	32.3580
					Welsh-Powell	20	20	20	47.5114
					Dsatur	20	20	20	85.0471
3	queen5_5	25	160	5	Greedy	5	5	20	1.9248
					Welsh-Powell	5	5	20	2.2992
					Dsatur	5	5	20	4.8808
4	1-Insertions_6	607	6337	7	Greedy	7	7	20	302.7108
					Welsh-Powell	7	7	20	265.0647
					Dsatur	7	7	20	932.4235
5	2-Insertions_5	597	3936	6	Greedy	6	6	20	1176.4913
					Welsh-Powell	6	6	20	976.1421
					Dsatur	6	6	20	1616.6928

Table 4 The performances of the proposed method of second group (Cont.)

No.	Benchmark	Vertices	Edges	$\chi(G)$	Heuristics	Max	Min	NS	Times (s)
6	1-FullIns_5	282	3247	6	Greedy	6	6	20	93.0449
					Welsh-Powell	6	6	20	73.2367
					Dsatur	6	6	20	307.8869
7	2-FullIns_5	852	12201	7	Greedy	7	7	20	1062.4438
					Welsh-Powell	7	7	20	852.4433
					Dsatur	7	7	20	3372.6978
8	3-FullIns_4	405	3524	7	Greedy	7	7	20	183.6137
					Welsh-Powell	7	7	20	149.1324
					Dsatur	7	7	20	644.2001
9	4-FullIns_4	690	6650	8	Greedy	8	8	20	547.6110
					Welsh-Powell	8	8	20	766.1774
					Dsatur	8	8	20	2174.0355
10	5-FullIns_4	1085	11395	9	Greedy	9	9	20	2200.7630
					Welsh-Powell	9	9	20	1805.3532
					Dsatur	9	9	20	5398.7114

Conclusions

We have presented the permutation-based optimization method for solving the GCP. We used Greedy method, Welsh-Powell method and Dsatur method to color the vertices for a given permutation of vertices. We classify the tested graphs selected from DIMACS benchmark into two groups. The results show that the proposed method can find the minimum numbers of colors as reported in the previous works for all cases. It indicates that the proposed method is effective for GCP.

Acknowledgements

The authors thank Department of Mathematics, Faculty of Science, Khon Kaen University.

References

Bre´laz D. New methods to color the vertices of a graph. Communications of the ACM 1979; 22(4): 251–256.

Dorne Ra, Hao JK. A New Genetic Local Search Algorithm for Graph Coloring. PPSN V Proceedings of the 5th International Conference on Parallel Problem Solving from Nature 1998; 745-754.

Galinier P, Hao JK. Hybrid evolutionary algorithms for graph coloring. Journal of Combinatorial Optimization 1999; 3(4): 379–397.

Hertz A, Werra D. Using tabu search techniques for graph coloring. Computing 1987; 39: 345–351.

Leighton FT. A Graph Coloring Algorithm for Large Scheduling Problems. Journal of Research of the Notional Bureau of Standards 1979; 84(6): 489-506.

Malaguti E, Toth P. A survey on vertex coloring problems. International Transactions in Operational Research, 2010; 17: 1–34.

Welsh DJA, Powell MB. An upper bound for the chromatic number of a graph and its application to timetabling problems. The Computer Journal 1967; 10(1): 85–86.

Werra D. An introduction to timetabling. European Journal of Operational Research 1985; 19(2): 151-162.

Woo TK, Su SYW, Wolfe RN. Resource allocation in a dynamically partitionable bus network using a graph coloring algorithm. IEEE Trans 2002; 39(12): 794–1801.