Journal of Engineering Science and Technology Review 13 (1) (2020) 126 - 132

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

Research Article

Strong Vertex-distinguishing Total Coloring Algorithm for Complete Graphs based on Equitable Coloring

Zhao Huanping[1,2,*], Xue Dangqin[1,2] and Shi Huojie[3]

[1]*College of Computer and Information Engineering, Nanyang Institute of Technology, Nanyang 473004, China*
[2]*College of Information Engineering, Zhengzhou Institute of Technology, Zhengzhou 450044, China*
[3]*Department of Biological Systems Engineering, Washington State University, Pullman, MI 98101, United States*

Received 22 August 2019; Accepted 4 February 2020

Abstract

Graph coloring has important research significance in graph theory. Strong vertex-distinguishing total coloring is a type of multi-conditional coloring in graph coloring, but existing associated studies lack analysis on constraint conditions. In this study, a new coloring algorithm was designed to increase the coloring efficiency of the strong vertex-distinguishing total coloring of a complete graph. By combining characteristics of complete graphs and strong vertex-distinguishing total coloring, the proposed algorithm decomposed the coloring color numbers into propercolor numbers and overcolor numbers, and the algorithm determined the filling quantity of each color number based on the idea of even coloring. The proposed algorithm implemented regular stepwise iteration by searching abnormal color sets on the edge coloring matrix until the constraint condition was achieved. The accuracy of the approach was proven by theoretical analysis and experimental comparison. The multiple experiments on 14–64 orders of complete graphs indicate that the 16-, 32-, and 64-order complete graphs require total coloring combination of overcolor numbers; this process generally needs 0.6–0.7 s. By contrast, the operation times for other orders of complete graphs are generally in the range 0.3–0.4 s. The proposed algorithm can effectively calculate the strong vertex-distinguishing total chromatic number of the complete graph with a fixed vertex number, and its time complexity is lower than $O(2^{m+1})$. These findings can provide important references in studying adjacent vertex-distinguishing total coloring and vertex-distinguishing total coloring.

Keywords: Equitable coloring, Strong vertex-distinguishing total coloring, Strong vertex-distinguishing total chromatic number, Complete graph, Overcolor number

1. Introduction

Graph coloring, a classical problem in graph theory, originates from the well-known "four-color conjecture." Many problems in practical life, such as computer communication, traffic orientation, goods storage, and combined optimization, can be solved by transforming them into graph coloring [1-2]. Thus, graph coloring is one topic with important practical value and theoretical importance in graph theory. However, classical intelligent optimization algorithms, such as genetic algorithm and neural network, have shortages and limitations in the strong vertex-distinguishing total coloring of graph coloring. Therefore, acquiring strong vertex-distinguishing total coloring performance quickly and effectively is a key problem that needs to be solved.

Existing studies on graph coloring mainly cover those on theories and on algorithms. In modern times, some mathematical researchers emphasize graph coloring problems [3-6]. Zhang et al. [7] proposed the concept and conjecture of strong vertex-distinguishing total coloring for graphs based on adjacent vertex-distinguishing total coloring and strong adjacent vertex-distinguishing total coloring. Nevertheless, graph coloring is considered an NP-complete problem. Traditional intelligence algorithms [8], such as genetic algorithm, ant colony algorithm, and neural network, are generally limited to solving graph coloring problems of single constraint, and they could obtain the expected coloring results under small-scale graphs. Ran and Zhang [9] achieved four-color graph coloring effectively by the improved heuristic ant colony algorithm. Zhang et al. [10] searched the initial solution of the genetic algorithm by using the ant colony algorithm and solved the vertex coloring problem involving multiple vertices by using the improved ant colony algorithm based on the genetic algorithm. However, the ordinary intelligent algorithm presents great limitations to solve multi-constraint graph coloring problems, such as strong vertex-distinguishing total coloring.

At present, few studies have been made on the strong vertex-distinguishing total coloring algorithm. For these reasons, the present study combined the characteristics of complete graph and strong vertex-distinguishing total coloring, after which the coloring number was decomposed into overcolor numbers and propercolor numbers based on equitable coloring. Moreover, overcolor numbers were filled in accordance with the principle of coloring combined maximum filling. This process effectively shortens operation time and increases the coloring efficiency of the algorithm.

2. State of the art

Existing studies on graph coloring mainly focus on theoretical studies. In 1993, Burris [11] introduced and

*E-mail address: spring1610@163.com
ISSN: 1791-2377
doi:10.25103/jestr.131.17

studied vertex-distinguishing edge coloring (also known as strong edge coloring). Major conclusions on the vertex-distinguishing edge coloring of graphs were mainly summarized in previous studies [12-15]. In 2002, Zhang et al. [16] proposed the concept and conjecture of strong adjacent vertex-distinguishing edge coloring based on vertex-distinguishing edge coloring. Many relevant studies have been reported worldwide [17-21]. In 2007, Zhang et al. [22] added a constraint to the concept of adjacent vertex-distinguishing edge coloring and proposed the concept of strong adjacent vertex-distinguishing total coloring. Meanwhile, they gained the accurate value of strong adjacent vertex-distinguishing total coloring for special graphs and the upper boundary conjecture. For a simple graph with less than three orders, $\chi_{vst} \le n + \lceil \log_2 n \rceil + 1$ may be used.

Studies on graph coloring algorithms can generally be divided into three types. The first type recognizes the law of graph coloring by combined construction, but this method is only applicable to unique graphs. The second type gives rough lower and upper boundaries of graph coloring based on the probability statistical method. However, such boundaries are markedly rough and, therefore, have some limitations. The third type processes graph coloring problems by using a computer, which can solve large-scale graph coloring problems by designing a reasonable high-efficiency algorithm based on the great operating capacity of computers. This approach can prove some conjectures. For example, based on the sequence approximation method, Appel and Haken [23] proved and solved the four-color graph coloring problem under computer assistance based on order approximation in 1976 (the computer operated for more than 1,200 hours). However, their method required a great deal of time in solving large-scale problems and the computation time was proportional to $O(n^2)$, where n represented the number of regions in the graph. By combining the advantages and disadvantages of the taboo search and genetic algorithms, Li and He [24] generated an initial solution by using the genetic algorithm to implement field-changing searching and update the vertex coloring by the taboo searching algorithm; their results showed an increase in the searching speed of the algorithm. Liao and Ma[25] analyzed graph coloring based on the heuristic searching ant algorithm and gained the expected results for small graphs. Yu [26] applied the simulated annealing algorithm to graph coloring, but the initial value and parameter determination of the algorithm could directly affect its performance. Any improper setting of parameters would lead to slow convergence and long implementation time. Yu et al. [27] provided the graph coloring model of uncertainty based on "DNA Origami." Li [28–29] studied the strong vertex-distinguishing total coloring and effectively calculated the vertex-distinguishing total coloring number for a graph with a fixed random number of vertices. Moreover, the time complexity of the solving algorithm was lower than $O(n^3)$.

Previous studies on graph coloring are all based on K-vertex coloring, but few studies have been conducted on strong vertex-distinguishing total coloring. Given that the constraints of strong vertex-distinguishing total coloring are significantly more complicated than K-vertex coloring, these algorithms will claim unacceptable operation time and convergence speed. In the preset study, a mathematical model of coloring algorithm was constructed based on the idea of equitable coloring, and constraints against coloring were determined. The coloring number was decomposed into overcolor numbers and propercolor numbers. Then, the coloring numbers were filled in after obtaining the coloring numbers and times, accelerating algorithm convergence.

The remainder of this study is organized as follows. Section 3 describes the concept, mathematical model, and algorithm design of strong vertex-distinguishing total coloring for complete graphs. Section 4 introduces the experimental results and analyzes the algorithms. Section 5 summarizes the relevant conclusions.

3. Methodology

3.1 Relevant definitions

For any undirected graph G(V,E), V(G) is the vertex set of G, E(G) is the edge set of G, and C(u) is the color set of vertex U and its associated edges in G. Therefore, relevant definitions for G coloring are introduced as follows.

Definition: $G(V,E)$ is the simply connected graph that has less than three orders, and k is a natural number and f is the mapping from $V(G) \cup E(G)$ to $\{1,2,...,k\}$. These parameters meet the following requirements:

(1) For any edge $uv \in E(G)$, $f(u) \ne f(v)$, $f(u) \ne f(uv)$, and $f(v) \ne f(uv)$ exist.

(2) For any two adjacent edges $uv, uw \in E(G)$ and $(v \ne w)$, $f(uv) \ne f(uw)$ exists.

(3) For any two vertices $i, j \in V(G)$, the color sets of the vertices meet $C(i) \ne C(j)$, where the color set of vertex i is $C(i) = \{f(i)\} \cup \{f(p) \mid ip \in E(G)\} \cup \{f(ip) \mid ip \in E(G)\}$.

Then, f is a strong vertex-distinguishing total coloring technique (k-VSDTC) of G.

Therefore, $\chi_{vst} = \min(k \mid k - VSDTC \ of \ G)$ is the strong vertex-distinguishing total chromic number of G.

Conjecture: If K_n expresses the n-order complete graph ($n \ge 3$), then the following occurs:

$$\chi_{vst}(K_n) = \begin{cases} n + \lceil \log_2 n \rceil + 1, & 2^{\lceil \log_2 n \rceil} - n = 2 \\ n + \lceil \log_2 n \rceil, & otherwise \end{cases} \quad (1)$$

3.2 Strong vertex-distinguishing total coloring algorithm for complete graph

3.2.1 Algorithm model

The basic idea of this algorithm is that for an n-order complete graph, k-VSDTC of K_n is decomposed into two parts, namely, propercolor and overcolor numbers. The latter only colors edges. Owing to the symmetry of a complete graph, vertices were colored by the propercolor number first, followed by the coloring of the edges. Given that the coloring combinations of the overcolor number were significantly smaller than those of the propercolor number, the maximum filling of the former was the filling principle. Finally, the objective function was used to determine whether or not the coloring results were satisfactory. Otherwise, the coloring results with conflicts were adjusted gradually based on the algorithm rules.

Theorem 1: We let K_n be the n-order complete graph ($n \ge 3$) and the total overcolor number is M.

$$M = \begin{cases} \lceil \log_2 n \rceil \times 2^{\lceil \log_2 n \rceil - 1} - 2 \times \lfloor \log_2 (2^{\lceil \log_2 n \rceil} - n) \rfloor & 5 \le n \le 15, n \ne 8 \\ \lceil \log_2 n \rceil \times 2^{\lceil \log_2 n \rceil - 1} - 2 \times \lceil \log_2 (2^{\lceil \log_2 n \rceil} - n + 1) \rceil & n = 3, n = 2^k, k = 2,3,4; \\ (\lceil \log_2 n \rceil - 4) \times 2^{\lceil \log_2 n \rceil - 1} + 2n + 8, & 2^{k-1} + 1 \le n \le 2^k - 10, k = 5,6,7 \cdots \\ (\lceil \log_2 n \rceil - 4) \times 2^{\lceil \log_2 n \rceil - 1} + 2n + 6 & 2^k - 9 \le n \le 2^k - 7, k = 5,6,7 \cdots \\ (\lceil \log_2 n \rceil - 4) \times 2^{\lceil \log_2 n \rceil - 1} + 2n + 4 & 2^k - 6 \le n \le 2^k - 4, k = 5,6,7 \cdots \\ (\lceil \log_2 n \rceil - 4) \times 2^{\lceil \log_2 n \rceil - 1} + 2n + 2 & n = 2^k - 3, n = 2^k - 1, k = 5,6,7 \cdots \\ (\lceil \log_2 n \rceil - 4) \times 2^{\lceil \log_2 n \rceil - 1} + 2n & n = 2^k, k = 5,6,7 \cdots \end{cases} \quad (2)$$

Theorem 2: We let K_n be the n-order complete graph ($n \ge 3$), and x and y represent the a and b times of overcolor number, respectively. Here, equitable coloring was applied. In other words, the difference between a and b is no higher than 2. Therefore,

$$\begin{cases} x = (1 - \frac{a}{2})n + \frac{M}{2}; \\ y = \frac{an - M}{2}; \\ a = \begin{cases} c, & c \equiv 0 (\mod 2); \\ c + 1, & c \equiv 1 (\mod 2). \end{cases} \\ c = \lceil M / \lceil \log_2 n \rceil \rceil; \end{cases} \quad (3)$$

where $c = \lceil M / \lceil \log_2 n \rceil \rceil$ means the minimum integer, which is no smaller than $c = \lceil M / \lceil \log_2 n \rceil \rceil$.

Proof: Theorem 1 indicates that it at least needs the $\lceil \log_2 n \rceil$ overcolor number to accomplish strong vertex-distinguishing total coloring for K_n. Theorem 2 indicates that the total overcolor number (M) can be calculated. Given that the equitable coloring of the overcolor number (*a=b+2*) is applied and the mean of $\lceil \log_2 n \rceil$ overcolor number is $M / \lceil \log_2 n \rceil$, with consideration toward the integer of color number, the finite equation of overcolor number is expressed as follows:

$$\begin{cases} x = (1 - \frac{a}{2})n + \frac{M}{2}; \\ y = \frac{an - M}{2}; \\ a = \begin{cases} c, & c \equiv 0 (\mod 2); \\ c + 1, & c \equiv 1 (\mod 2). \end{cases} \\ c = \lceil M / \lceil \log_2 n \rceil \rceil; \end{cases} \quad (4)$$

By solving the equation, we can obtain

$$\begin{cases} x = (1 - \frac{a}{2})n + \frac{M}{2} \\ y = \frac{an - M}{2} \end{cases} . \quad (5)$$

End of proof.

Theorem 3: We let K_n be the n-order complete graph ($n \ge 3$), and x and y are a and b times of propercolor number. Here, equitable coloring was applied. In other words, the difference between a and b is no higher than 2 (we only considered edge coloring here and vertex coloring was accomplished). Therefore, we have the expressions below.

$$\begin{cases} x = \frac{1}{2}(n^2 - (a - 1)n - m) \\ y = \frac{1}{2}(n^2 - (a + 1)n - m) \\ a = \begin{cases} c, & c \equiv 0 (\mod 2) \\ c + 1, & c \equiv 1 (\mod 2) \end{cases} \\ c = \lceil (n^2 - n - M) / n \rceil \end{cases} \quad (6)$$

In the equations above, $c = \lceil (n^2 - n - M) / n \rceil$ means no lower than $(n^2 - n - M) / n$.

Proof: The total propercolor number is $n^2 - n - m$. The proof is similar to that of Theorem 2 and the limited equation set of the propercolor number is easily gained.

$$\begin{cases} x + y = n \\ ax + by = n^2 - n - M \\ b = a - 2 \\ c = \lceil (n^2 - n - M) / n \rceil \\ a = \begin{cases} c, & c \equiv 0 (\mod 2) \\ c + 1, & c \equiv 1 (\mod 2) \end{cases} \end{cases} \quad (7)$$

The limited equation of the propercolor number can be gained by solving the following equation set:

$$\begin{cases} x = \frac{1}{2}(n^2 - (a - 1)n - M) \\ y = \frac{1}{2}(n^2 - (a + 1)n - M) \end{cases} \quad (8)$$

End of proof.

Based on coloring times for the overcolor number and the propercolor number and the limited equation, an algorithm could be designed to accomplish specific coloring.

3.2.2 Description of algorithms

Input:

the number of vertices (*n*) in a complete graph

Output:

the colored adjacent matrix *Col*(*n*-1,*n*-1) that is expressed by a two-dimensional (2D) array

Step 1) Initialization

Step 1.1) Definition of variables and array:

n this integer is the number of vertices in the complete graph;

Col(*n*-1,*n*-1) this integer is the output array of the coloring result;

OverColorNum this integer is a variable that defines the overcolor number;

ProperColorNum this integer is a variable that defines the propercolor number;

OverColorArr() this integer is the array that defines the overcolor number array;

ProperColorArr() this integer is the array that defines the propercolor number array;

OverColorCount this integer is a limited variable of the coloring times of the overcolor number;

ProperColorCount this integer is a limited variable of the coloring times of the propercolor number;

Step 1.2) For a 2D array *Col*(*i*,*j*), *C*(*i*,*j*) expresses a vertex when *i≠j*; otherwise, it expresses an edge;

Step 1.3) For *OverColorArr*(*i*)= 2^{i+1} ,*i*=0,1,...,*m*-1, the overcolor number was expressed by the power set of 2 for the convenience judgment on repetition of overcolor number;

Step 1.4) For *ProperColorArr*(*i*)=*i*+100+1,*i*=0,1,...,*n*-1, the propercolor number applies a continuous integer that is higher than the overcolor number;

Step 1.5) The propercolor coloring was implemented to all vertices individually: $f(v_i v_i)$=*ProperColorArr*(i). All edges were initialized at 0, $f(v_i v_j)=0, i \neq j$;

Step 2) Overcolor Coloring

Step 2.1) The overcolor number $m=\lceil \log_2 n \rceil$ was determined based on Theorem 1, and n is the number of vertices in a complete graph;
Step 2.2) The specific overcolor number is determined according to Theorem 2;
Step 2.3) In the following text, the overcolor coloring process was described by taking K_{28} as an example. For $m=\lceil \log_2 28 \rceil=5$ (overcolor number = 2, 4, 8, 16, and 32), Theorem 1 indicates that $M=2^{\lceil \log_2 n \rceil -1}+2n+4=76$;
Step 2.4) The overcolor number set *del* that has to be deleted is determined. Any combination number of five overcolor numbers in Step (3) is 32, but there are only 28 vertices; thus, 4 combinations must be eliminated. Given that the using times of the overcolor number in a complete graph are even, the null sets Ø, {2}, {4} and {2, 4} are eliminated. In other words,*del*={ Ø ,{2},{4},{2,4}};
Step 2.5) The overcolor numbers 2 and 4 are colored first, and then the overcolor number after coloring 2 and 4 is determined. The set S1 formed by any combination of 8, 16, and 32 is solved, that is, S_1 ={Ø,{8},{16},{32},{8,16},{8,32},{16,32},{8,16,32}}. The combination of the colored overcolor number is set as a: a={Ø,{2},{4},{2,4}} where the overcolor numbers are 2 and 4 and the set to store overcolor number is P(4,32). Next, P(0,0) = Ø× S_1 -*del*=7 (numbers without 2 and 4), P(0,1) ={2}×S_1-*del*=7 (numbers only with 2), P(0,2) ={4}×S_1-*del*=7 (numbers only with 4), and P(0,3) ={2,4}× S_1 -*del*=7 (numbers with 2 and 4) are calculated first;
Step 2.6) The overcolor coloring is achieved based on the overcolor number that is calculated in Step 2.2). Next, the adjacent matrix that corresponds to the complete graph is retrieved. The number of rows in which the overcolor number set lies is calculated and compared with the value in set P. If two values are the same, we turn to Step 2.7); otherwise, we repeat Step 2.6).
Step 2.7) The coloring of the overcolor number 8 and then the quantity of overcolor number combinations is determined. At this point, the colored overcolor number combination is a={Ø,{2},{4},{8},{2,4}{2,8},{4,8},{2,4,8}}. Meanwhile, set S_2 , which is composed of any combination of 16 and 32, is calculated as S_2 ={Ø,{16},{32},{16,32}}. The following operation is implemented by each element of a and S_2: P(1,0) =a× S_2-*del*=3,P(1,1)=a× S_2 -*del*=3,P(1,2)=a× S_2 -*del*=3,P(1,3)=a× S_2 -*del*=3,P(1,4) =a× S_2-*del*=4,P(1,5)=a× S_2-*del*=4,P(1,6) =a× S_2-*del*=4,P(1,7) =a× S_2-*del*=4;
Step 2.8) We repeat Steps 2.6) and 2.7) until the coloring of overcolor number 8 is finished.
Step 2.9) Overcolor number 16 is colored, after which the number of overcolor number combinations is determined. At this point, the overcolor number combination for coloring is a={Ø,{2},{4},{8},{16},{2,4}{2,8},{2,16},{4,8},{4,16},{8,16},{2,4,8},{2,4,16},{2,8,16}{4,8,16},{2,4,8,16}}. The set S_3 formed by any combination of 32 is calculated, which is S_3= {Ø,{32}}. The following operation is implemented by each element of a and S_3: P(2,0)=a× S_3 -*del*=1,P(2,1)=a× S_3 -*del*=1,P(2,2)=a× S_3 -*del*=1,P(2,3)=a× S_3 -*del*=1,P(2,4)=a× S_3 -*del*=2, P(2,5)=a× S_3 -*del*=2, P(2,6)=a× S_3 -*del*=2, P(2,7) =a× S_3 -*del*=2, P(2,8)=a× S_3 -*del*=2, P(2,9)=a× S_3 -*del*=2, P(2,10)=a× S_3 -*del*=2, P(2,11)=a× S_3 -*del*=2, P(2,12)=a× S_3 -*del*=2, P(2,13)=a× S_3 -*del*=2, P(2,14)=a× S_3 -*del*=2, and P(2,15)=a× S_3 -*del*=2;
Step 2.10) We repeat Steps 2.8) and 2.9) until the coloring of overcolor number 16 is finished;
Step 2.11) Overcolor number 32 is colored, after which the number of overcolor number combinations is determined. At this point, the overcolor number combination for coloring is as follows: a={Ø,{2},{4},{8},{16},{32},{2,4}{2,8},{2,16},{2,32},{4,8},{4,16},{4,32},{8,16},{8,32},{16,32},{2,4,8},{2,4,16},{2,4,32},{2,8,16},{2,8,32},{2,16,32},{4,8,16},{4,8,32},{4,16,32},{8,16,32},{2,4,8,16},{2,4,8,32},{2,4,16,32},{2,8,16,32},{4,8,16,32},{2,4,8,16,32}}. Set S4 is calculated as S_4= Ø. The following operation is implemented by each element of a and S_4: P(3,0) =a× S_4 -*del*=0, P(3,1) = a× S_4- *del* =0, P(3,2) =a× S_4 -*del*=0, P(3,3)=a× S_4 -*del*=0, P(3,4)= a× S_4 -*del*=1, P(3,5) = a× S_4- *del* =1, P(3,6) = a× S_4- *del* =1, P(3,7) = a× S_4- *del* =1, P(3,8) = a× S_4- *del* =1, P(3,9) = a× S_4- *del* =1, P(3,10) = a× S_4- *del* =1, P(3,11) = a× S_4- *del* =1, P(3,12) = a× S_4- *del* =1, P(3,13) = a× S_4- *del* =1, P(3,14) = a× S_4- *del* =1, P(3,15) = a× S_4- *del* =1, P(3,16) = a× S_4- *del* =1, P(3,17) = a× S_4- *del* =1, P(3,18) = a× S_4- *del* =1, P(3,19) = a× S_4- *del* =1, P(3,20) = a× S_4- *del* =1, P(3,21) = a× S_4- *del*=1, P(3,22) = a× S_4- *del* =1, P(3,23) = a× S_4- *del* =1, P(3,24) = a× S_4- *del* =1, P(3,25) = a× S_4- *del* =1, P(3,26) = a× S_4- *del* =1, P(3,27) = a× S_4- *del* =1, P(3,28) = a× S_4- *del* =1, P(3,29) = a× S_4- *del* =1, P(3,30) = a× S_4- *del* =1, and P(3,31) = a× S_4- *del* =1;
Step 2.12) We repeat Step 2.11) until we complete the coloring of overcolor number 32;

Step 3) Propercolor Coloring

Step 3.1) We let A be a set variable and numbers with 0 medians in the *col*(n-1,n-1) array are added into A, where $a(i,j)$∈A and $i \neq j$. Moreover, a random array is generated by using the random function based on length in A. Elements in A are rearranged according to the random array.
Step 3.2) The limitation table of the propercolor number is calculated from the propercolor limited equation according to the initialization function. This process is performed to determine the times for the appropriate occurrence of each propercolor number: *limitcounter*(k) (k=0,1,…,n-1);
Step 3.2.1) One element $a(i,j)$ is collected from A according to an order and the nowcolor number is selected from the limited table *nowcolor* = *ProperColorArr*(k), where k= 0,1,2,...,n-1. If $f(col(i,p)$ ≠ *nowcolor* $f(col(p,j)$ ≠*nowcolor* , p=0,1,...,n-1, there's $f(col(i,j)$=*nowcolor*. Then, $a(i,j)$ is deleted from A and the value of *limitcounter*(k) is added by 1;
Step 3.2.2) If $f(col(i,p)$=*nowcolor* or $f(col(p,j)$= *nowcolor* , one element $a(i,j)$ is selected again from A according to the order and then repeat 1);
Step 3.2.3) If the value of *limitcounter*(k) reaches the value in the limit value, k=k+1. Then, we repeat Step 3.2.1).

4. Results analysis and discussion

4.1 Experimental results

In this study, complete graphs with 8–64 orders were chosen in the experiment. The expected experimental results were gained in a short period. The experimental results were outputted into one text document in the system catalog. The achievement results in the document are introduced below.

(1) The coloring results of K_{16} (all overcolor coloring combinations need to appear) are listed in Table.1.

(2) The coloring results of K_{26} are listed in Tables.2(a) and 2(b):

Table 1. The coloring results of a complete graph with 16 orders

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9	V_{10}	V_{11}	V_{12}	V_{13}	V_{14}	V_{15}	V_{16}
V_1	101	108	111	106	114	103	112	107	104	113	102	105	18	109	116	110
V_2	108	102	110	114	109	104	113	101	103	112	106	115	111	12	14	107
V_3	111	110	103	101	107	112	104	109	106	105	116	13	115	108	102	14
V_4	106	114	101	104	111	113	102	105	12	109	107	116	110	115	112	108
V_5	114	109	107	111	105	14	115	106	110	104	113	102	108	101	103	112
V_6	103	104	112	113	14	106	105	110	101	107	108	114	12	102	18	109
V_7	112	113	104	102	115	105	107	18	13	14	101	109	106	103	108	12
V_8	107	101	109	105	106	110	18	108	114	103	112	111	104	13	113	102
V_9	104	103	106	12	110	101	13	114	109	108	115	107	102	113	105	111
V_{10}	113	112	105	109	104	107	14	103	108	110	114	101	116	18	106	13
V_{11}	102	106	116	107	113	108	101	112	115	114	111	104	105	110	109	103
V_{12}	105	115	13	116	102	114	109	111	107	101	104	112	103	106	110	113
V_{13}	18	111	115	110	108	12	106	104	102	116	105	103	113	107	101	114
V_{14}	109	12	108	115	101	102	103	13	113	18	110	106	107	114	111	105
V_{15}	116	14	102	112	103	18	108	113	105	106	109	110	101	111	115	104
V_{16}	110	107	14	108	112	109	12	102	111	13	103	113	114	105	104	116

Table 2(a). The coloring results of a complete graph with 26 orders

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9	V_{10}	V_{11}	V_{12}	V_{13}	V_{14}	V_{15}	V_{16}
V_1	101	119	115	112	8	110	121	102	116	106	122	104	103	16	120	105
V_2	119	102	106	105	109	118	124	103	117	114	123	122	126	32	2	120
V_3	115	106	103	116	114	108	118	124	2	107	119	102	109	101	112	8
V_4	112	105	116	104	2	119	106	114	108	122	101	110	8	4	113	118
V_5	8	109	114	2	105	120	111	116	32	123	124	113	118	117	106	112
V_6	110	118	108	119	120	106	112	2	125	4	109	117	111	113	103	107
V_7	121	124	118	106	111	112	107	125	122	120	32	114	116	109	101	113
V_8	102	103	124	114	116	2	125	108	105	121	117	111	120	122	119	104
V_9	116	117	2	108	32	125	122	105	109	118	115	107	104	121	102	110
V_{10}	106	114	107	122	123	4	120	121	118	110	8	109	117	108	126	101
V_{11}	122	123	119	101	124	109	32	117	115	8	111	105	112	104	118	108
V_{12}	104	122	102	110	113	117	114	111	107	109	105	112	108	119	123	16
V_{13}	103	126	109	8	118	111	116	120	104	117	112	108	113	106	107	124
V_{14}	16	32	101	4	117	113	109	122	121	108	104	119	106	114	105	102
V_{15}	120	2	112	113	106	103	101	119	102	126	118	123	107	105	115	122
V_{16}	105	120	8	118	112	107	113	104	110	101	108	16	124	102	122	116

Table 2(b). The coloring results of a complete graph with 26 orders

	V_{17}	V_{18}	V_{19}	V_{20}	V_{21}	V_{22}	V_{23}	V_{24}	V_{25}	V_{26}
V_{17}	117	104	121	110	16	120	126	107	32	101
V_{18}	104	118	103	112	117	109	113	122	108	102
V_{19}	121	103	119	114	113	105	118	2	106	108
V_{20}	110	112	114	120	118	116	108	119	102	125
V_{21}	16	117	113	118	121	106	105	101	124	116
V_{22}	120	109	105	116	106	122	114	118	8	107
V_{23}	126	113	118	108	105	114	123	104	109	112
V_{24}	107	122	2	119	101	118	104	124	117	106
V_{25}	32	108	106	102	124	8	109	117	125	122
V_{26}	101	102	108	125	116	107	112	106	122	126

4.2 Algorithm analysis

4.2.1 Validity of the proposed algorithm

(1) According to the definition of strong vertex-distinguishing total coloring, the colored adjacent matrix must meet the following conditions:

the color numbers of the adjacent vertices must be different;

the color numbers of each row must be different;

the sets formed by all color numbers of each row must be different.

(2) The overcolor coloring is the principal part of the proposed algorithm. Therefore, the algorithm accomplishes overcolor coloring first. Given that the overcolor number in the final colored matrix exists as one form in the combination of 2, $C_5^0 + C_5^1 + C_5^2 + C_5^3 + C_5^4 + C_5^5$ = 1+5+10 +10+5+1 exist for five types of overcolor numbers. The combinations include the following: Ø, {2}, {4}, {8}, {16},

{32}, {2,4}, {2,8}, {2,16}, {2,32}, {4,8}, {4,16}, {4,32}, {8,16}, {8,32}, {16,32}, {2,4,8}, {2,4,16}, {2,4,32}, {2,8,16}, {2,8,32}, {2,16,32}, {4,8,16}, {4,8,32}, {4,16,32}, {8,16,32}, {2,4,8, 16}, {2,4,8,32}, {2,4,16,32}, {2,8,16,32}, {4,8,16,32}, and {2,4,8,16,32}. For a complete graph with 28-vertex coloring, Ø, Ø, Ø, {2}, {2}, {2}, {4}, {4}, {4}, Ø, Ø, Ø, {2,4}, {2,4}, {2,4}, {2}, {2}, {2}, {4}, {4}, {4}, Ø, {2,4}, {2,4}, {2,4}, {2}, {4}, and {2,4} must occur after the coloring of overcolor number 4 in order to complete the follow-up coloring of overcolor number. Similarly, {8}, Ø, Ø, {2,8}, {2}, {2}, {4,8}, {4}, {4}, {8}, {8}, Ø, {2,4,8}, {2,4}, {2,4}, {2,8}, {2,8}, {2}, {4,8}, {4,8}, {4}, {8}, {2,4,8}, {2,4,8}, {2,4}, {2,8}, {4,8}, and {2,4,8} are needed to finish the follow-up coloring after the coloring of overcolor number 8. Thus, the reasonable number of combinations that is needed after filling of overcolor number 16 and 32 can be calculated. On this basis, the coloring steps 2.1)–2.12) of overcolor coloring in the proposed algorithm were designed and $a \times S$ represented all sets that would occur. By eliminating *del* which shall not occur, the reasonable combinations which shall occur can be obtained. In this way, the overcolor number combinations for coloring can be concluded by using *del* cleverly. Therefore, completing the overcolor coloring is highly easy, This is also a difficult part of the whole algorithm. Of course, conditions 1) and 2) need to be met during the filling of overcolor number.

(3) After completing the overcolor number, the filling of the propercolor number is relatively easier. All elements with the value 0 are added into *A*. One is selected randomly and put into the set B_i, which has a tag of color number and determined capacity. In this case, capacity of B_i is calculated from the limited equation of the propercolor number, and it shall be put based on the following rules:

1) If *i*=1,2,…,16 and the element *a*(*u*,*v*) is selected from *A*, *a* can be placed in B_i with a tag unequal to *u* or *v*.

2) Elements are inputted into B_i according to a sequence. Once B_i is full, the remaining elements shall be placed into the next B_i, which is not yet full.

3) Step 3 of the algorithm is designed and it is very effective for completing the propercolor coloring based on the ideal of average filling of the propercolor number.

4.2.2 Time complexity of the proposed algorithm

Two types of factors that influence the time complexity of the algorithm, namely, propercolor and overcolor coloring.

(1) Time complexity of overcolor number: the proposed algorithm colors each overcolor number independently. The overcolor number *i* needs to be judged and compared by its combination with 2^i. Therefore, the time complexity of overcolor coloring is $O(2^{m+1})$ for a complete graph with *m* overcolor numbers.

(2) Time complexity of propercolor number: for the *n* propercolor numbers, the total propercolor number for filling is $n^2 - n - M$, where M is the total overcolor number. Each filling of one propercolor number will be compared by whether its rows and columns have the same color. Thus, the time complexity for filling propercolor number is $O((n^2 - n - M) \times n)$.

To sum up, the operation time of the proposed algorithm is determined by overcolor number (m) for coloring. Thus, the time complexity in the worst situation is $O(2^{m+1})$.

5. Conclusions

Achieving strong vertex-distinguishing total coloring is a complicated problem in graph coloring. To solve strong vertex-distinguishing total coloring, the proposed algorithm decomposes the color number into overcolor and propercolor numbers after obtaining the strong vertex-distinguishing total color number of a complete graph. On the basis of the idea of average coloring, edges, and vertices are colored by overcolor coloring. This process not only reduces the operation time but also increases the convergence speed of the proposed algorithm. The following conclusions could be drawn.

(1) We verify $2^{\lceil \log_2 n \rceil} - n = 2$ by analyzing the constraints against the strong vertex-distinguishing total coloring for a complete graph. At this point, the strong vertex-distinguishing total chromatic number for a n-order complete graph is $n + \lceil \log_2 n \rceil + 1$; otherwise, its value is $n + \lceil \log_2 n \rceil$.

(2) The overcolor number is far smaller than the propercolor number. On the basis of properties of arrangement combination, the overcolor number combinations are significantly fewer than propercolor number combinations. The maximum overcolor number can shorten the operation time of the proposed algorithm.

(3) The overcolor number can be expressed in a power set of 2. The overcolor number combination (*del*), which needs to be deleted, can be determined in advance by the mathematical model of the algorithm, thereby obtaining the reasonable and accurate color number combinations and finishing overcolor coloring quickly.

The proposed algorithm classifies coloring based on the constraints and characteristics of strong vertex-distinguishing total coloring for complete graphs. This algorithm focuses on filling the overcolor number, supplemented by filling the propercolor number. It completed the coloring of the n-order complete graph in a short period of time. This algorithm can be modified slightly, by which other coloring results of complete graphs can be gained, such as adjacent vertex-distinguishing total coloring and vertex-distinguishing total coloring. However, the overcolor number combinations in the proposed algorithm are completed manually. Therefore, an appropriate method for determining overcolor number combinations will be further designed to shorten the operation time of the algorithm.

Acknowledgements

The study was supported by the National Natural Science Foundation of China (Grant No. 10771091) and the Science and Technology Research Project of Henan Province (Grant No. 142102210554).

References

1. Han, G. L., Du, W., “The application research of parking space allocation based on graph coloring model”. *Telecom Express*, (7), 2019, pp.27-32.
2. Liu, W., Liao, R. B., “PCI automatic allocation method based on improved graph colorling algorithm”. *Computer engineering and design*, 39(2), 2018, pp.352-357.
3. Burris, A. C., Schelp, R. H., “Vertex-distinguishing Proper Edge-Colorings”. *Journal of Graph Theory*, 26(2), 1997, pp.73-82.
4. Bo, Y. H., Wang, L. X., “2-distance Coloring of Sparse Planar Graphs”, *Advances in Mathematics*, 48(2), 2019, pp.145-155.
5. Dong, X. Y., Ma, D. J., “Strong edge coloring of Goldberg snark”. *Journal of Northeast Normal University (Natural Science Edition)*, 50(4), 2018, pp.16-19.
6. Li, T., Chen, X. E., Wang, Z. W., “General Vertex-Distinguishing Total coloring of Corona graph of 2K2VK1”, *Journal of Jilin University(Science Edition)*, 57(3), 2019, pp.544-552.
7. Zhang, Z. F., Li, J. w., C, X. E., “On adjacent vertex distinguishing total coloring of graphs”. *Science in China (Ser A)*, 48(3), 2005, pp. 289-299.
8. Wei, J., “Application of Graph Coloring Algorithm Based on Ant Colony Optimization in Intelligent Community”. *Journal of Jishou University (Natural Science Edition)*, 40(2), 2019, pp. 27-30, 33.
9. Ran, C. S., Zhang, T., “Improved coloring algorithm based on ant colony for application in intelligence community”. *Computer Engineering and Design*, 37(1), 2016, pp.152-153.
10. Zhang, X. P., Zhang, Y. Q., “An Ant Colony Genetic Algorithm for Graph Colouring Problem”. *Computer Application and Software*, 31(11), 2014, pp. 207-209.
11. Burris, A. C., “*Vertex-Distinguishing Edge-Colorings*”. Doctoral Dissertation of Memphis State University, United States, 1993, pp. 118 -129.
12. Dong, X. Y., “Strong Edge Coloring of Flower Snark”. *Journal of Changchun Normal University*, 38(2), 2019, pp.4-8.
13. Whitehead, C., Zageglia, N., Obser, S., “Observability of the extended Fibonacci cubes”. *Discrete Mathematics*, 266(1-3), 2003, pp.431-440.
14. Balister, P. N., Riordan, M. O., Schelp, R. H., “Vertex-distinguishing edge coloring of graphs”. *Journal of Graph Theory*, 42(2), 2003, pp.95-109.
15. Cui, J. F., “Discussing the Upper Bounds on Vertex-Distinguishing Edge coloring of a graph by probality method”. *Journal of Capital Normal University (Natural Sciences Edition)*, 40(1), 2019, pp.12-14.
16. Zhang, Z. F., Lin, L. Z., Wang, J. F., “Adjacent strong edgecolorings of graphs”. *Applied Mathematics Letters*, 15(3), 2002, pp. 623-626.
17. Tian, S. L., Dong, X. F., Liu, R. L., “Adjacent vertex-distinguishing coloring of the semistrong product of graphs”. *Operations Research Transactions*, 21(3), 2017, pp. 119-125.
18. Balister, P. N., Guori, E., Lehel, J., “Adjacent vertex distinguishing edge colorings”, *SIAM Journal on Discrete Mathematics*, 21(1), 2007, pp. 237-250.
19. Baril, J. L., Kheddouci, H., Togni, O., “Adjacent vertex distinguishing edge coloring of meshes and hypercubes”, *Australasian Journal of Combinatory*, 35(1), 2006, pp. 237-250.
20. Hocquard, H., Montassier, M., “Adjacent vertex-distinguishing edge coloring of graphs with maximum degree at least five”, *Electronic Notes in Discrete Mathematics*, 38(1), 2011, pp.457-462.
21. Hocquard, H., Montassier, M., “Adjacent vertex distinguishing edge coloring of graphs with maximum degree Δ ”. *Journal of Combinatorial Optimization*, 26(1), 2013, pp.152-160.
22. Zhang, Z. F., Chen, H., Yao, B., et al. “On the adjacent-vertex-strongly-distinguishing total coloring of graphs”. *Science in China A Mathematics*, 51(3), 2008, pp.427-436.
23. Appel, K., Haken, W., “The solution of the four-color-map problem”. *Scientific American*, 237(4), 1977, pp.108-121.
24. Li. S. Z., He, S. F., “Hybrid o timization algorithm based on graph coloring problem”. *Application Research of Computers*, (1), 2016, pp.90-100.
25. Liao, F. X., Ma, L., “Heuristic Search-based Ant Algorithm of Solving Graph Coloring Problem”, *Computer Engineering*, 33(16), 2007, pp. 191-195.
26. Yu, P. J., “*The Study of Graph Coloring Based on Simulated Annealing Genetic Algorithm*”, Master thesis of GuiZhou University, China, 2009, pp2-4.
27. Yu, Y., Su, S., Zhao, J., “A “DNA origami”-based approach to the solution of graph coloring problem”. *Journal of Nanjing University (Natural Science)*, 52(4), 2016, pp. 656-660.
28. Li, J. W., Li, X. H., Dong, W., “Algorithm for vertex distinguishing total coloring of random graphs”. *Application Research of Computers*, 32(6), 2015, pp. 1707-1710.
29. Cao, D. T., Li, J. W., Jiang, D. D., “Adjacent vertex-distinguishing equitable V-total coloring algorithm of graph based on multi-objective optimization”. *Journal of Computer Application*, 37(2), 2017, pp. 457-462.