An Evolutionary Approach for Graph Multi-Coloring Problem

Keunhee Han
Department of Applied Mathematics
Kongju National University, Gongju, Korea

Chansoo Kim
Department of Applied Mathematics
Kongju National University, Gongju, Korea
(Corresponding author)

Copyright © 2015 Keunhee Han and Chansoo Kim. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Multi-coloring problem is a generalization of the well known Graph coloring problem which is known to be NP-hard. Multi-coloring problem can be solved by algorithms designed for the graph coloring problem after transformation of the graphs. However, since the graph transformations increase the size and order of the given graph, in some cases, it may be impractical to solve multi-coloring problem by the algorithms for graph coloring problem using transformations. In this paper, we suggest a genetic algorithm that solves multi-coloring problem without transformation of graphs.

Keywords: Graph Coloring Problem, multi-coloring Problem, Genetic Algorithm

1 Introduction

Let $G = (V, E)$ be a simple undirected graph where $V = \{1, 2, ..., n\}$ is the vertex set and $E$ is the edge set. A coloring of $G$ is a mapping $c: V \rightarrow \{1, 2, ..., k\}$ such that if $uv \in E$ then $c(u) \neq c(v)$. Graph coloring problem (GCP) is a problem to find a coloring for a given graph $G$ with smallest $k$ (the chromatic number $\chi (G)$ of $G$). We call such a coloring of $G$ as proper coloring of $G$. Let $w (v)$ be a positive weight...
defined for each vertex \( v \in V \). The multi-coloring problem (MCP) is a generalization of GCP such that each vertex \( v \in V \) must be assigned \( w(v) \) distinct colors from \( \{1, ..., k\} \) such that if \( ij \in E \) then \( C(i) \cup C(j) = \emptyset \), where \( C(v) \) denotes the set of \( w(v) \) distinct colors assigned to vertex \( v \). The objective of MCP is to minimize the number of colors assigned to vertices. Therefore, GCP is a special case of MCP with \( w(v) = 1 \) for all the vertices of \( G \). MCP is NP-hard since MCP generalize GCP, that is known to be NP-hard [1]. MCP has its applications in the various area of problems including scheduling and frequency assignment problems. For the detailed introduction to the application areas of MCP, please refer to [2, 3].

MCP can be solved by the algorithms developed for GCP after transform the given graphs [3]. Let \( G = (V, E, W) \) be a (vertex) weighted graph such that \( V = \{1, 2, ..., n\} \). Based on \( G \) we construct an unweighted graph \( C(G) = (V', E') \) as follows. In \( C(G) \), each vertex \( i \in V \) expands to a clique of size \( w(i) \), where a clique is a set of pairwise adjacent vertices. Specifically, let \( c(i) = \{c_{i1}, c_{i2}, ..., c_{iw(i)}\} \) (\( |c(i)| = w(i) \)) be the vertices expanded from \( i \in V \) and for \( i \in V \) and \( j \in \{1, ..., w(i)\} \), let \( p(c_{ij}) = i \), i.e., \( p(c_{ij}) \) is the vertex in \( V \) such that \( p(c_{ij}) \) generates the vertices of clique \( c(i) \) in \( C(G) \). Then \( C(G) = (V', E') \) is defined as follows:

\[
V' = \bigcup_{i \in V} c(i), \\
E' = \{xy \mid x, y \in c(i), i \in V\} \cap \{xy \mid p(x)p(y) \in E, x, y \in V'\}
\]

We denote \( C(G) \) an expanded graph of \( G \). If we let \( e(i, j) = 1 \) if \( ij \in E \) and \( e(i, j) = 0 \) if \( ij \notin E \) (\( i, j \in V \)), then the size and order of \( C(G) \) are as follows:

\[
|V'| = \sum_{i \in V} w(i) \\
|E'| = \sum_{i = 1}^{n} \frac{w(i)(w(i)-1)}{2} + \sum_{1 \leq i < j \leq n} w(i)w(j)e(i, j)
\]

In Figure 1, (a) shows a graph \( G = (V, E, W) \) such that \( V = \{1, 2, 3\} \) and \( w(1) = 3 \), \( w(2) = 2 \) and \( w(3) = 1 \). The graph in (b) shows the expanded graph \( C(G) = (V', E') \) of \( G \). Note that according to (1), \( |V'| = 6 \), and \( |E'| = 12 \) while \( |V| = 3 \) and \( |E| = 2 \).
It is not hard to see that if we apply proper coloring to $C(G)$, then the resultant coloring is a multi-coloring of $G$ and these are the approaches taken by the authors in [2, 3] for MCP. In [3], the authors solved MCP by transforming graphs into expanded graphs and applying the combination of Squeaky Wheel Optimization and Tabu Search techniques (SWO+TS). However, as noted in [2], since the transformation to expanded graph introduces extra vertices and edges if the weights of vertices grows fast then applying proper coloring to the expanded graph becomes impractical. Therefore, in this study we suggest an Genetic Algorithm for MCP without any transformation of graphs.

The rest of the paper is organized as follows. In Section 2, we suggest an genetic algorithm for MCP. In Section 3, we present the experimental results of our algorithm. The last section includes a conclusion.

2 Genetic Algorithm for MCP

Genetic Algorithm (GA) were introduced by Holland in 1975 [4]. Since then GAs have been applied successfully to variety of combinatorial optimization problems. GAs maintain a population, which consists of a some number of possible solutions (called chromosomes) and utilize three genetic operators such as selection, crossover and mutation. After generating the initial population selection operation chooses the more promising chromosomes to construct a next generation according to their fitness values. Chromosomes with more promising fitness values will most likely be selected to reproduce, whereas, those with less promising fitness values will be discarded. After the selection, the chromosomes in population are subjected to crossover and mutation operations. Crossover is the operation of generating new chromosomes (called offsprings) from the existing two chromosomes (called parents) in such a way that the offsprings inherit the good properties of both parents. Mutation is the operation of changing some properties of the chosen chromosomes in order to prevent convergence of the population into a local optimum. For the rest of the paper, we freely use the notations and terminologies defined in [5].

2.1 Initialization

Let $G = (V, E, W)$ be a vertex weighted undirected graph such that $w(i)$ denotes the weight of a vertex $i \in V$. We also let $V = \{1, 2, ..., n\}$; hence $|V| = n$. For MCP, chromosomes stores the orderings of vertices, where an ordering of $V$ is a bijection $\alpha: V \leftrightarrow \{1, 2, ..., n\}$. Therefore, for the suggested GA, a chromosome is a vector $\alpha$ containing a permutation of $[1..n]$. We denote $\alpha[i]$ ($1 \leq i \leq n$) the $i$th element in $\alpha$. 
According to the orderings of a chromosome \( \alpha \) the greedy algorithm in Figure 2 assigns \( w(v) \) colors to vertex \( v \) in greedy manner. In this algorithm, let \( F_v \) be a 0-1 vector of length \( \text{len} = \sum_{i=1}^{n} w(i) \). \( F_v \), initially containing all 0’s, denotes the forbidden colors for vertex \( v \), i.e., if \( F_v[i] = 1 \) then color \( i \) cannot be assigned to \( v \). \( C_v \) represents a vector of length \( w(v) \) which contains the colors assigned to vertex \( v \).

**Algorithm Greedy**

**Input:** \( G = (V, E, W) \) and an ordering \( \alpha \) of \( V \), \( \text{len} = \sum_{i=1}^{n} w(i) \).

**Output:** Multi-coloring of \( G \)

\[
\begin{align*}
\text{begin} & \\
\text{for } i = 1 \text{ to } n & \text{ do} \\
& v = \alpha(i); \\
& \text{cnt} = 0; \\
& \text{while } \text{cnt} < w(v) \text{ do} \\
& \quad \text{for } j = 1 \text{ to } \text{len} \\
& \quad \quad \text{if } F_v[i] = 0 \text{ then} \\
& \quad \quad \quad C_v[\text{cnt++}] = j; \\
& \quad \text{endwhile} \\
& \text{let } S = \{ u \mid uv \in E \} \\
& \text{for } j = 1 \text{ to } w(v) \\
& \quad \text{for each } u \in S \text{ do} \\
& \quad \quad F_u[C_v[j]] = 1; \\
& \quad \text{endfor} \\
& \text{endfor} \\
\text{end} \\
\end{align*}
\]

Figure 2. Greedy algorithm for multi-coloring

Let \( k \) be the maximum color used by the greedy algorithm for a graph \( G \). In order to treat the MCP as a maximization problem we define the fitness function as follows:

\[
\text{fitness} = \frac{n}{2} w(i) - k.
\]

For the initial population we generate the chromosomes as follows. Let \( \alpha \) be a chromosome. \( \alpha[1] \) is selected randomly. For \( 2 \leq i \leq n \), among all the unselected vertices, \( \alpha[i] \) is the vertex not adjacent to \( \alpha[i-1] \). If there is no such vertex exist then \( \alpha[i] \) is selected randomly.
2.2 Genetic operators
For the selection operations, we use the roulette wheel selection mechanism with slots sized proportionally according to the fitness of each chromosome. Our GA also use elitism which is a mechanism to ensure that the most highly fit chromosome of the population is passed on to the next generation without any modifications.

For the crossover operations, we use Partially Mapped Crossover (PMX) originally developed for Travelling Saleman Problem by Goldberg [4]. We also developed a new crossover operator for MCP and call it a R-crossover. The R-crossover works as follows. Let $p_1$, $p_2$ be two parent chromosomes and $o_1$, $o_2$ be two offsprings generated from $p_1$ and $p_2$. Also let $pos$ be a random number such that $1 \leq pos \leq n$. Then the elements of $p_1[0] \sim p_1[pos-1]$ and $p_2[pos] \sim p_2[n]$ are inherited to $o_1[0] \sim o_1[pos-1]$ and $o_2[pos] \sim o_2[n]$, respectively. The elements of $p_1[pos] \sim p_1[n]$ are inherited to $o_1[pos] \sim o_1[n]$ according to the relative orderings of $p_2$. Likewise, the elements of $p_2[0] \sim p_2[pos-1]$ are inherited to $o_2[0] \sim o_2[pos-1]$ according to the relative ordering of $p_2$. For example, let $n = 7$ and $p_1$ and $p_2$ as follows:

$$p_1 = (2 \ 4 \ 3 \ 1 \ 7 \ 5 \ 6)$$
$$p_2 = (1 \ 3 \ 2 \ 7 \ 6 \ 5 \ 4)$$

Assume that $pos = 4$ is selected randomly. Then the first step of R-crossover produces $o_1$ and $o_2$ as follows:

$$o_1 = (2 \ 4 \ 3 \ * \ * \ * \ *)$$
$$o_2 = (* \ * \ * \ 7 \ 6 \ 5 \ 4)$$

For $p_1$ the remaining elements are $\{1, 7, 5, 6\}$ and they appear in the order of $\{1, 7, 6, 5\}$ in $p_2$. Therefore, $o_1[pos] \sim o_1[n]$ are filled with $\{1, 7, 6, 5\}$ in this order. Similarly, the remaining elements of $p_2$ are $\{1, 3, 2\}$ and they appear in the order of $\{2, 3, 1\}$ in $p_1$. Therefore, $o_1[0] \sim o_1[pos-1]$ are filled with $\{2, 3, 1\}$ in this order. Hence, the second step of R-crossover produces $o_1$, $o_2$ as follows:

$$o_1 = (2 \ 4 \ 3 \ 1 \ 7 \ 6 \ 5)$$
$$o_2 = (2 \ 3 \ 1 \ 7 \ 6 \ 5 \ 4).$$

In order to introduces extra diversity among population, our GA algorithm applies the two mentioned crossover operators with 50% chance. In our extensive experiments, we found that using two different crossover operators always produce better solutions than using single crossover operator for MCP.
For the mutation operations, we use the exchange mutation. This operation first choose two random integers \( \text{pos}_1 \) and \( \text{pos}_2 \) such that \( \text{pos}_1 \neq \text{pos}_2 \) and \( 1 \leq \text{pos}_1, \text{pos}_2 \leq n \). Then simply swaps \( p[\text{pos}_1] \) with \( p[\text{pos}_2] \).

### 3. Experiments

In this section, we describe the computational experiments of our suggested GA algorithm. For tests, we have employed the set of GEOM graphs which consist of 33 geometric graphs. GEOM graphs are well known benchmark graphs for MCP and freely available in [7]. We have implemented our GA algorithm with C# and the experiments were conducted on a desktop computer at 2.69 GHz with 4GB of RAM.

For tests, we have used the following parameters for our suggested GA. Population Size = 40, Crossover Rate = 0.8, Mutation Rate = 0.04 and Maximum Generation = 800. Table 1 contains an empirical comparison of our suggested GA algorithm and the SWO+TS algorithm in [3] for the 33 GEOM graphs. The contents of Table 1 are as follows:

- \( G \) is the name of the instance graphs.
- \( |V| \) and \( |E| \) denote the number of vertices and edges of \( G \), respectively.
- \( |V'| \) and \( |E'| \) denote the number of vertices and edges of expanded graph \( C(G) \), respectively.
- \( k^* \) is the best known value (maximum color used) of the graph \( G \). (these are also the results of SWO+TS)
- \( k \) is the maximum color used by our suggested GA algorithm.
- \( t_1 \) is the time (in seconds) taken by SWO+TS[3].
- \( t_2 \) is the time (in seconds) taken by our suggested GA algorithm.

From the table, one can see that the suggested GA algorithm is able to find the best known value of the all 33 benchmark graphs with much less time compare to SWO+TS algorithm. The table also shows that the size and order of \( C(G) \) is much greater than those of the original graph \( G \). Compare to other benchmark graphs such as the DIMACS graphs, the sizes of GEOM graphs are relatively small. Even if the sizes of the graphs are small, if the weights of vertices grow fast, then the size of the transformed graphs grow fast accordingly. Therefore, under these cases the suggested GA algorithm is useful.

### 4. Conclusions

In this work, we showed that relatively simple genetic algorithm can solve the multi-coloring problem without transformations of the original graphs. Bandwidth
Coloring and Bandwith multi-coloring Problem are further generalizations of GCP. Therefore, we will try to apply the suggested GA to these problems without transformations of graphs.

References


Table 1. Results of the Experiments

| $G$     | $|V|$ | $|E|$ | $|V'|$ | $|E'|$ | $k^*$ | $k$ | $t_1$ | $t_2$ |
|---------|------|------|------|------|------|-----|------|------|
| GEOM20  | 20   | 20   | 118  | 1,048| 28   | 28  | 4    | 0    |
| GEOM20a | 20   | 37   | 100  | 1,327| 30   | 30  | 3    | 0    |
| GEOM20b | 20   | 32   | 40   | 132  | 8    | 8   | 7    | 0    |
| GEOM30  | 30   | 50   | 143  | 1,419| 26   | 26  | 61   | 0    |
| GEOM30a | 30   | 81   | 171  | 3,288| 40   | 40  | 99   | 0    |
| GEOM30b | 30   | 81   | 69   | 447  | 11   | 11  | 20   | 0    |
Table 1. (Continued): Results of the Experiments

<table>
<thead>
<tr>
<th>GEOM</th>
<th>40</th>
<th>78</th>
<th>220</th>
<th>3,074</th>
<th>31</th>
<th>31</th>
<th>131</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEOM40a</td>
<td>40</td>
<td>146</td>
<td>203</td>
<td>4,473</td>
<td>46</td>
<td>46</td>
<td>138</td>
<td>1</td>
</tr>
<tr>
<td>GEOM40b</td>
<td>40</td>
<td>157</td>
<td>84</td>
<td>743</td>
<td>14</td>
<td>14</td>
<td>29</td>
<td>0</td>
</tr>
<tr>
<td>GEOM50</td>
<td>50</td>
<td>127</td>
<td>285</td>
<td>4,935</td>
<td>35</td>
<td>35</td>
<td>227</td>
<td>1</td>
</tr>
<tr>
<td>GEOM50a</td>
<td>50</td>
<td>238</td>
<td>302</td>
<td>9,649</td>
<td>61</td>
<td>61</td>
<td>312</td>
<td>2</td>
</tr>
<tr>
<td>GEOM50b</td>
<td>50</td>
<td>249</td>
<td>104</td>
<td>1,140</td>
<td>17</td>
<td>17</td>
<td>46</td>
<td>1</td>
</tr>
<tr>
<td>GEOM60</td>
<td>60</td>
<td>185</td>
<td>315</td>
<td>6,174</td>
<td>36</td>
<td>36</td>
<td>275</td>
<td>1</td>
</tr>
<tr>
<td>GEOM60a</td>
<td>60</td>
<td>339</td>
<td>362</td>
<td>13,105</td>
<td>65</td>
<td>65</td>
<td>391</td>
<td>2</td>
</tr>
<tr>
<td>GEOM60b</td>
<td>60</td>
<td>366</td>
<td>127</td>
<td>1,785</td>
<td>22</td>
<td>22</td>
<td>64</td>
<td>1</td>
</tr>
<tr>
<td>GEOM70</td>
<td>70</td>
<td>267</td>
<td>384</td>
<td>8,534</td>
<td>44</td>
<td>44</td>
<td>374</td>
<td>2</td>
</tr>
<tr>
<td>GEOM70a</td>
<td>70</td>
<td>459</td>
<td>379</td>
<td>14,821</td>
<td>71</td>
<td>71</td>
<td>439</td>
<td>3</td>
</tr>
<tr>
<td>GEOM70b</td>
<td>70</td>
<td>488</td>
<td>148</td>
<td>2,212</td>
<td>22</td>
<td>22</td>
<td>84</td>
<td>1</td>
</tr>
<tr>
<td>GEOM80</td>
<td>80</td>
<td>349</td>
<td>465</td>
<td>12,927</td>
<td>63</td>
<td>63</td>
<td>551</td>
<td>3</td>
</tr>
<tr>
<td>GEOM80a</td>
<td>80</td>
<td>612</td>
<td>389</td>
<td>15,545</td>
<td>68</td>
<td>68</td>
<td>476</td>
<td>3.8</td>
</tr>
<tr>
<td>GEOM80b</td>
<td>80</td>
<td>663</td>
<td>169</td>
<td>3,028</td>
<td>25</td>
<td>25</td>
<td>111</td>
<td>2</td>
</tr>
<tr>
<td>GEOM90</td>
<td>90</td>
<td>441</td>
<td>530</td>
<td>16,180</td>
<td>51</td>
<td>51</td>
<td>697</td>
<td>3</td>
</tr>
<tr>
<td>GEOM90a</td>
<td>90</td>
<td>789</td>
<td>454</td>
<td>20,455</td>
<td>65</td>
<td>65</td>
<td>625</td>
<td>5</td>
</tr>
<tr>
<td>GEOM90b</td>
<td>90</td>
<td>860</td>
<td>184</td>
<td>3,602</td>
<td>28</td>
<td>28</td>
<td>135</td>
<td>3</td>
</tr>
<tr>
<td>GEOM100</td>
<td>100</td>
<td>547</td>
<td>581</td>
<td>19,829</td>
<td>60</td>
<td>60</td>
<td>845</td>
<td>4</td>
</tr>
<tr>
<td>GEOM100a</td>
<td>100</td>
<td>992</td>
<td>528</td>
<td>28,496</td>
<td>81</td>
<td>81</td>
<td>854</td>
<td>6</td>
</tr>
<tr>
<td>GEOM100b</td>
<td>100</td>
<td>1,050</td>
<td>200</td>
<td>4,429</td>
<td>30</td>
<td>30</td>
<td>156</td>
<td>3</td>
</tr>
<tr>
<td>GEOM110</td>
<td>110</td>
<td>638</td>
<td>643</td>
<td>24,799</td>
<td>62</td>
<td>62</td>
<td>1,022</td>
<td>5</td>
</tr>
<tr>
<td>GEOM110a</td>
<td>110</td>
<td>1,207</td>
<td>602</td>
<td>38,783</td>
<td>91</td>
<td>91</td>
<td>1,172</td>
<td>8</td>
</tr>
<tr>
<td>GEOM110b</td>
<td>110</td>
<td>1,256</td>
<td>220</td>
<td>5,163</td>
<td>37</td>
<td>37</td>
<td>211</td>
<td>4</td>
</tr>
<tr>
<td>GEOM120</td>
<td>120</td>
<td>773</td>
<td>680</td>
<td>27,759</td>
<td>64</td>
<td>64</td>
<td>1,268</td>
<td>6</td>
</tr>
<tr>
<td>GEOM120a</td>
<td>120</td>
<td>1,434</td>
<td>664</td>
<td>46,429</td>
<td>93</td>
<td>93</td>
<td>2,349</td>
<td>9</td>
</tr>
<tr>
<td>GEOM120b</td>
<td>120</td>
<td>1,491</td>
<td>235</td>
<td>5,779</td>
<td>34</td>
<td>34</td>
<td>489</td>
<td>5</td>
</tr>
</tbody>
</table>

Received: February 2, 2015; Published: March 5, 2015