Proceedings of the 2023 International Conference on Mathematical Physics and Computational Simulation
DOI: 10.54254/2753-8818/28/20230312

Optimizing map coloring: Using linear programming to find the minimum number of colors

Yao Li

University of Washington, Seattle, WA, 98105, U.S.

JLIY0502@163.com

Abstract. Map coloring is a classic problem in graph theory and it relates to many optimization techniques in mathematics such as linear programming and simulated annealing. This paper investigates the minimum number of colors required to color a map under different constraints and situations using linear programming. Specifically, it examines three different scenarios: (1) coloring each district on the map with the constraint that adjacent districts must be colored differently, (2) adding the additional constraint that two regions bordering the same region cannot be colored the same, and (3) assigning two colors to each district with the constraint that adjacent districts must be colored differently. To proceed with the research, hypotheses are formulated regarding the impact of these additional constraints on the minimum number of colors required to color the map. The data in the paper is collected by creating sample maps and analyzing the minimum number of colors required to color them under each of the different scenarios. The findings of this research suggest that the addition of constraints, indicating a complex situation, increases the minimum number of colors needed to color the map. Thus, linear programming is found to be an effective optimization technique for solving map coloring problems under these constraints. This research makes a valuable contribution to the field of mathematics and computer science, providing insights into the application of optimization techniques to real-world problems like map coloring. The findings of the research have significant implications for practitioners working in the field of optimization and inform the development of more efficient algorithms for solving map coloring problems.

Keywords: linear programming, map coloring, optimization, constraints, modeling.

1. Introduction

The classic graph theory problem of "map coloring" entails giving different regions of a map different colors while ensuring that no two adjacent regions have the same color. This problem has many real-world applications, including scheduling, timetabling, and register allocation in computer science. However, finding the minimum number of colors needed to color a map can be a difficult and time-consuming task, especially when additional constraints are introduced. This paper explores the use of linear programming as an optimization technique to find the minimum number of colors needed to color a map under different constraints and situations. Specifically, it investigates three different situations: (1) coloring each district on the map with the constraint that adjacent districts must be colored differently, (2) adding the additional constraint that two regions bordering the same region cannot be colored the same, and (3) assigning two colors to each district with the constraint that adjacent districts must be colored differently.

2. Literature review

The map coloring problem could date to the 19th century when Francis Guthrie [1] posed the famous four-color problem: Can every map be colored with only four colors such that no two adjacent regions share the same color? This problem was later proven true by Appel and Haken in 1977 [2]. Since then, the map coloring problem has been extended to different variations, including cases with constraints such as adjacent regions having different colors or regions that share a border that cannot have the same color. The problem's objective is to find the minimum number of colors required to color a map, and this is where linear programming comes into play in the research.

As an optimization method that is applied to the Map Coloring problem, linear programming [3] is a mathematical optimization technique that can be used to solve problems with linear constraints and linear objective functions. The technique involves defining decision variables, which represent the unknowns in the problem and formulating constraints that describe the problem's limitations.

Several previous studies have used computational methods to solve map coloring problems, with the objective of finding the minimum number of colors needed to color the map under different constraints. For instance, a study by M. Kaufmann and R. Wiese [4] used a computation technique called simulated annealing [5], which is a stochastic optimization method, to solve the map coloring problem with the constraint that adjacent regions should have different colors. The idea is to start with a random coloring of the map and then gradually modify the coloring to reduce the number of adjacent regions with the same color, while allowing for occasional "mistakes" to avoid getting stuck in local optima. This process continues until a satisfactory coloring is found.

3. Application of linear programming in map coloring problems

According to the Graph Theory [6] in discrete mathematics. A graph is a type of mathematical structure made up of pairs of connected vertices (also known as nodes or points) and a set of edges (also known as links or lines). An edge is an unordered pair of vertices, so it can be represented as a set u, v where u and v are the two vertices it connects. In other words, a graph G can be defined as $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, where V is a set of vertices and E is a set of edges, and each edge $\mathrm{e} \in \mathrm{E}$ is a set of two vertices $\mathrm{e} = \mathrm{u}, \mathrm{v}$ that it connects.

For example, let $G = (V, E)$ be a graph, where $V = 1,2,3,4,5$ is the set of vertices and $E = 1,2,1,3,2,3,2,4$ is the set of edges. The diagram of G could be (Figure 1):

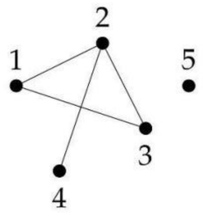


Figure 1. Diagram of G.

In this graph, the set of vertices $V = 1,2,3,4,5$ represents the five nodes in the graph, and the set of edges $E = 1,2,1,3,2,3,2,4$ represent the connections between the nodes. For example, the edge 1,2 connects vertex 1 to vertex 2, and the edge2,3 connects vertex 2 to vertex 3. Vertex 5 does not connect to any other vertex.

A useful tool to visualize the vertices and the corresponding edge in a complicated graph is the adjacency matrix [7]. For a graph $G = (V, E)$ that has n vertices, define its adjacency matrix A to be the $n \times n$ matrix with the rules:

$$A_{ij} = 1 \text{ if } (i, j) \in E \tag{1}$$

$$A_{ij} = 0 \text{if } (i, j) \notin \mathrm{E} \tag{2}$$

For the G = (V, E) that is defined before, the adjacency matrix is in Eq. (3):

$$\begin{pmatrix} 01100 \\ 10110 \\ 11000 \\ 01000 \\ 00000 \end{pmatrix} \tag{3}$$

In terms of map coloring. For the graph $G = (V, E)$, define the coloring of the graph to be a set of colors $C = \{c_1, c_2, \ldots, c_r\}$ and an assignment $f : V \Rightarrow C$ such that $f(v) \neq f(w)$ whenever $(v, w) \in E$. Thus, two vertices are assigned distinct colors if they are connected by an edge. As an example, in the case of an empty graph, the graph can be colored as shown in Figure 2:

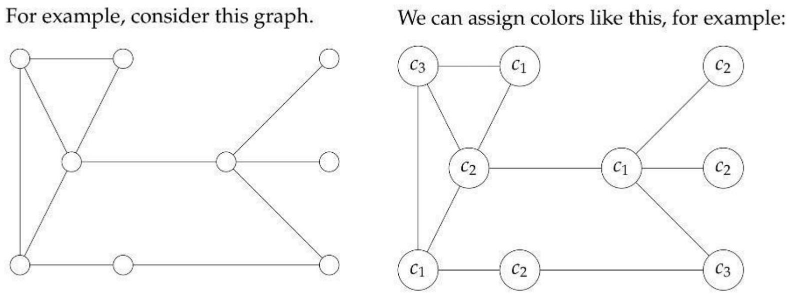

For example, consider this graph.
We can assign colors like this, for example:
c_3
c_1
c_2
c_2
c_1
c_2
c_1
c_2
c_3

Figure 2. Example of coloring.

This research paper has used the map of Zhengzhou, a city in Henan province in China, as an example. Figure 3 is the map of Henan [8]:

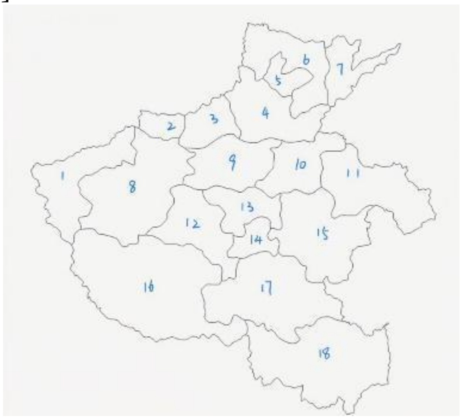


Figure 3. Map of Henan.

In the form of vertices and edges, the map looks like Figure 4:

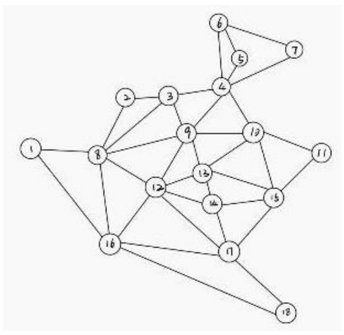
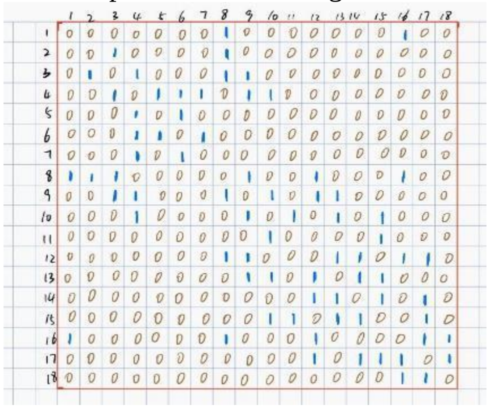


Figure 4. Henan in the form of vertices and edges.

And the adjacency matrix for the map of Henan is in Figure 5:



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
2	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
3	0	1	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0
4	0	0	1	0	1	1	1	0	1	1	0	0	0	0	0	0	0	0
5	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
8	1	1	1	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0
9	0	0	1	1	0	0	0	1	0	1	0	1	1	0	0	0	0	0
10	0	0	0	1	0	0	0	0	1	0	1	0	1	0	1	0	0	0
11	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
12	0	0	0	0	0	0	0	1	1	0	0	0	1	1	0	1	1	0
13	0	0	0	0	0	0	0	0	1	1	0	1	0	1	1	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0
15	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	0
16	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1
17	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Figure 5. Adjacency matrix of Henan in situation 1.

The following step is to examine the implementation of color constraints on the map by constructing linear programming models for each situation.

4. Investigation in the specific scenario

4.1. *Situation 1*

Coloring each district on the map with the constraint that adjacent districts must be colored differently.

The map of Henan is a graph $G = (V, E)$ with 18 vertices, which $V = 1,2, \dots ,18$ and E represents the edges between the two vertices. Then, define the graph's adjacency matrix A to be a $n \times m$ matrix with $A_{ij} = 1$ if $(i, j) \in E$, and $A_{ij} = 0$ if $(i, j) \notin E$.

The Four-Color Theorem states [9] that "a map can be colored while ensuring that two adjacent districts are not assigned the same color, and at most four colors would be required". Hence, it is feasible to set the upper limit of the number of colors to four while formulating the linear programming model. This technique can help to decrease the computational complexity of the process and accelerate the computation process.

Following the previous procedure, define $y_i \in 0,1$, where $i = 1, 2, 3, 4$, and $y_i = 1$ if and only if the map has already used the color "i". Also define $x_{ij} \in 0,1$, where $i = 1, 2, 3, 4$ and $j \in [1,18]$, with $x_{ij} = 1$ if and only if vertex j will have the color "i". Generally, y_i and x_{ij} is mainly used to ensure the two adjacent districts have different colors. The subscript "i" represents which color to use and the subscript "j" represents which vertex to paint.

The objective and constraint functions of the linear program were designed as follows.

The goal of the map coloring problem is to use the fewest colors possible to draw the map. This is achieved by minimizing the sum of y_i, where $y_i \in 0,1$ (Note that the subscript "i" represents which color to use in the vertex. Then, $y_i \in 0,1$ represents whether the color has been used, and thus the sum of y_i

would represent the total number of colors). The objective function of the problem is therefore defined as:

$$\min: \sum_{i=1}^{4} y_i \tag{4}$$

In addition to the objective function, several constraints need to be applied to meet the requirements of the map coloring problem.

The first constraint is to ensure each vertex should only have one color. Note that the subscript "i" represents which color to use and the subscript "j" represents which vertex to paint. In this way, x_{ij} represents whether the specific color "i" has been used in vertex "j". To ensure that each vertex "i" has only one color "j", a constraint can be constructed as follows:

$$\sum_{j=1}^{18} x_{ij} = 1, 1 \le i \le 4 \tag{5}$$

The second constraint is to ensure each region only should be colored once. That is, the region's color cannot be covered again. This procedure is mainly to improve the accuracy of the calculations. To fulfill this constraint, the constraint should make sure $x_{ij} \in 0,1 \le\ y_i \in 0,1$, which means the number of colors in the vertex cannot exceed one. Therefore, a new constraint equation can be obtained as follows:

$$x_{ij} \le y_j, 1 \le j \le 18 \tag{6}$$

The third constraint is to ensure the districts that share the same border should always be colored differently. Define "k" to be some vertices that have edges between the old vertex "j". Thus $x_{ik} \in 0,1$ represents if the adjacent vertex of "j" has a color "i". The constraint should make sure $x_{ij} + x_{ik} \le\ 1$. That is, the two adjacent vertices "j" and "k" should use the same specific color "i" together. A new constraint can be formulated as follows:

$$x_{ij} + x_{ik} \le 1, \mathrm{forall}(v_j, v_k) \in \mathrm{Eand}1 \le i \tag{7}$$

The fourth constraint is to make sure that the color with a small subscript should be used first. That is, in solving the linear program, color 2 will not be considered if color 1 has not been used, and color 3 will not be used if color 2 has not been used. With this algorithm, the computation will be much. In linear program notation, it can be written as:

$$y_i \le y_{i-1}, 2 \le i \le 4 \tag{8}$$

In the end, there is a constraint to ensure all the variables are binary in the linear program:

$$x_{i_j}, y_i \in 0,1; 1 \le i \le 4, 1 \le j \le 18 \tag{9}$$

Overall, the linear program could be:

$$\min: \textstyle\sum_{i=1}^{4} y_i \tag{10}$$

$$\mathrm{s.t.} \textstyle\sum_{j=1}^{18} x_{ij} = 1, 1 \le i \le 4 \tag{11}$$

$$x_{i_j} \le y_j, 1 \le j \le 18 \tag{12}$$

$$x_{i_j} + x_{i_k} \le 1, \mathrm{forall}(v_j, v_k) \in \mathrm{Eand}1 \le i \le 4 \tag{13}$$

$$y_i \le y_{i-1}, 2 \le i \le 4 \tag{14}$$

$$x_{i_j},\ y_i \in 0,1; 1 \le i \le 4, 1 \le j \le 18 \tag{15}$$

After solving the linear program, the resulting graph can be obtained as shown in Figure 6:

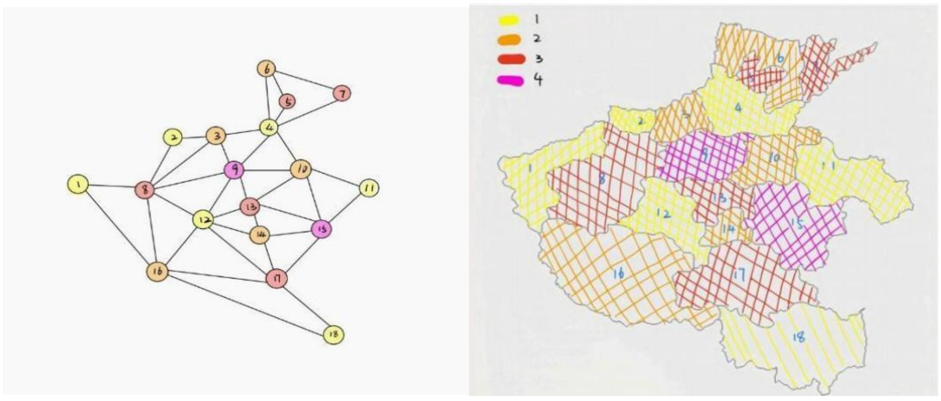


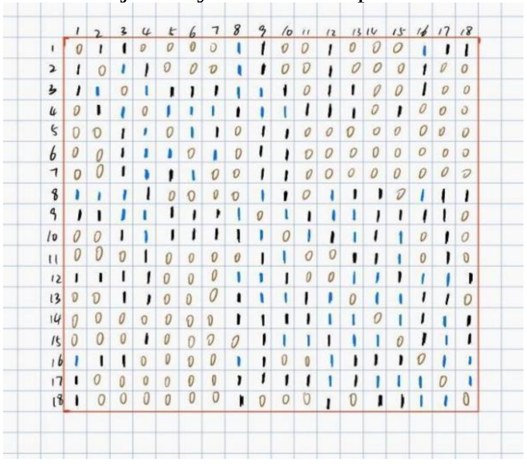
Figure 6. Colored map of Henan in situation 1.

Figure 6 shows the minimum number of colors in the map of Henan in this situation is four.

4.2. Situation 2

In addition to situation 1 adding the additional constraint that two regions bordering the same region cannot be colored the same.

In order to comply with the supplementary requirement presented in scenario 2, it is necessary to amend the adjacency matrix by introducing fresh edges connecting vertices that correspond to districts sharing a common boundary. This guarantees that two regions that share a common boundary cannot possess identical colors. The revised adjacency matrix is depicted below in Figure 7:



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	1	1	0	0	0	0	1	1	0	0	1	0	0	0	1	1	1
2	1	0	1	1	0	0	0	1	1	0	0	1	0	0	0	1	0	0
3	1	1	0	1	1	1	1	1	1	1	0	1	1	0	0	1	0	0
4	0	1	1	0	1	1	1	1	1	1	1	1	1	0	1	0	0	0
5	0	0	1	1	0	1	1	0	1	1	0	0	0	0	0	0	0	0
6	0	0	1	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0
7	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0
8	1	1	1	1	0	0	0	0	1	1	0	1	1	1	0	1	1	1
9	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0
10	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	0
11	0	0	0	1	0	0	0	0	1	1	0	0	1	1	1	0	1	0
12	1	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	1
13	0	0	1	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0
14	0	0	0	0	0	0	0	1	1	1	1	1	1	0	1	1	1	1
15	0	0	0	1	0	0	0	0	1	1	1	1	1	1	0	1	1	1
16	1	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	1	1
17	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1
18	1	0	0	0	0	0	0	1	0	0	0	1	0	1	1	1	1	0

Figure 7. Adjacency matrix of Henan in situation 2.

Due to the additional constraint in this scenario, the Four-Color Theorem cannot be utilized to simplify the linear program. Therefore, an upper limit of 18 colors is set for the computation, which corresponds to the number of vertices in the model. Thus, the objective function and constraint are updated as follows:

$$min\colon \sum_{i=1}^{18} y_i \tag{16}$$

$$s.t. \sum_{j=1}^{18} x_{ij} = 1, 1 \le i \le 18 \tag{17}$$

$$x_{i_j} \le y_j, 1 \le j \le 18 \tag{18}$$

$$x_{i_j} + x_{i_k} \le 1, for\ all\left(v_j, v_k\right) \in E\ and\ 1 \le i \le 18 \tag{19}$$

$$y_i \le y_{i-1}, 2 \le i \le 18 \tag{20}$$

$$x_{i_j},\ y_i \in 0,1; 1 \le i \le 18, 1 \le j \le 18 \tag{21}$$

Note that the linear program in this situation is only slightly different to the previous one. Besides the change of the upper bound of the number of colors in computation, the only difference is the definition of $(v_j, v_k) \in E$. Presently, it is necessary to ensure that vertices linked to the same vertex are assigned distinct colors, which necessitates the inclusion of additional edges in E. As a result, the set of edges in this scenario encompasses more elements than it did in the first scenario.

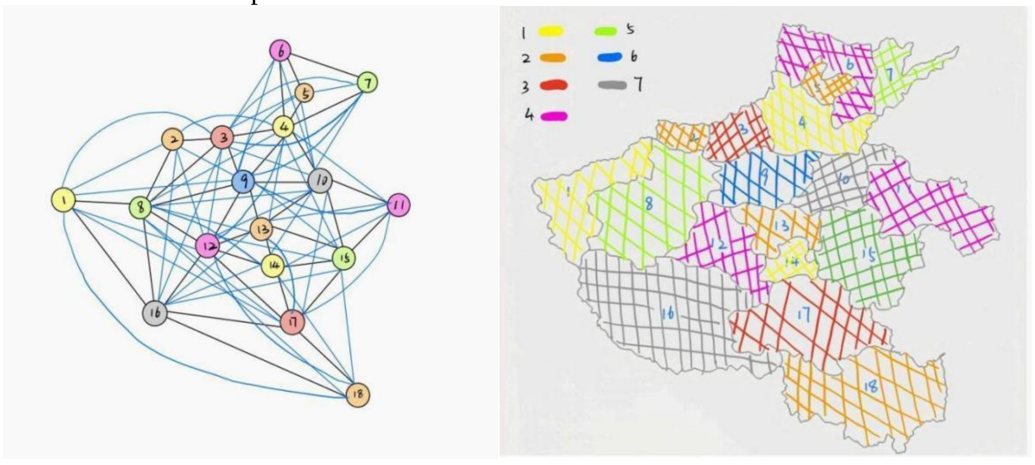


Figure 8. Colored map of Henan in situation 2.

In this situation, the minimum number of colors is seven, as shown in Figure 8.

4.3. Situation 3

Assigning two colors to each district with the constraint that adjacent districts must be colored differently.

In the third scenario, the goal is to assign two distinct colors to each region in a manner that ensures neighboring regions do not share the same color. In order to meet this requirement, it is necessary to modify the first constraint of the linear program to allow each vertex to accommodate two colors. Therefore, the revised first constraint is as follows:

$$\sum_{j=1}^{18} x_{i_j} = 2 \tag{22}$$

Given that there are 18 vertices in the map, and each vertex must be assigned two distinct colors, a total of 36 colors may be required to draw the map while ensuring neighboring regions have different colors. Therefore, set the upper bound for the number of colors used in the map to be 36, which is double the number of vertices. As a result, the updated linear program in this scenario is as follows:

$$min: \sum_{i=1}^{36} y_i \tag{23}$$

$$s.t. \sum_{j=1}^{18} x_{ij} = 2, 1 \le i \le 36 \tag{24}$$

$$x_{i_j} \le y_j, 1 \le j \le 18 \tag{25}$$

$$x_{i_j} + x_{i_k} \le 1, forall(v_j, v_k) \in Eand1 \le i \le 36 \tag{26}$$

$$y_i \le y_{i-1}, 2 \le i \le 36 \tag{27}$$

$$x_{i_j},\ y_i \in 0,1; 1 \le i \le 36, 1 \le j \le 18 \tag{28}$$

Upon solving the linear program, this study obtains the following outcome:

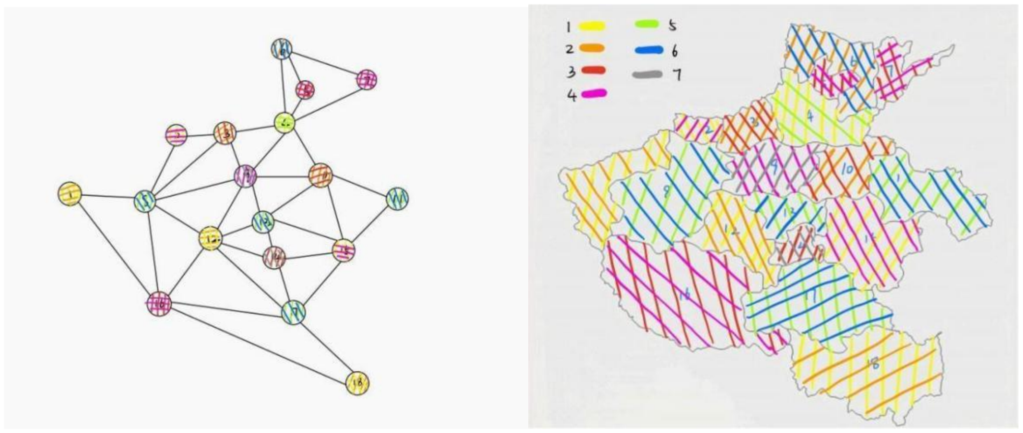


Figure 9. Colored map of Henan in situation 3.

In this situation, the minimum number of colors is also seven, as shown in Figure 9.

5. Conclusion

This research paper explores the utilization of linear programming to solve the map coloring problem with diverse sets of constraints. A methodology is presented that models the problem as a linear program and uses lpsolve [10] in Java to solve it, thereby obtaining the minimum number of colors required to color the map.

The analysis of the results demonstrates that the minimum number of colors required to color a map varies depending on the applied constraints. In the first scenario where adjacent regions must have distinct colors, the minimum number of colors is four, as anticipated. In the second scenario where additional constraints are introduced, the minimum number of colors rises to seven. In the third scenario where each region is allocated two colors, the minimum number of colors is also seven.

The findings obtained from the linear programming solutions underscore the efficacy of this approach in determining the minimum number of colors needed to color a map while satisfying different sets of constraints. The methodology can be employed in other similar optimization problems, offering an efficient and scalable solution for a broad range of practical applications.

To conclude, this research has contributed to the field of optimization by showcasing the application of linear programming in solving the map coloring problem with diverse sets of constraints. Future research could investigate more complex and practical scenarios, such as non-planar graphs, various types of constraints, and larger maps. In general, this work has practical implications for domains like scheduling, resource allocation, and routing, where similar optimization problems emerge.

References

[1] Gardner, M. (1975). Mathematical Games: The Four-Color Problem. Scientific American, 232(4), 112-117.

[2] Appel, K. and Haken, W. (1977) Every Planar Map Is Four Colorable. Part I discharging. Illinoi -s Journal of Mathematics, 21, 429-490. - references - scientific research publishing. (n.d.). R -etrieved March 6, 2023, from Scirp.org website: https://www.scirp.org/(S(lz5mqp453edsnp5 -5rrgjct55))/reference/referencespapers.aspx?referenceid=1929762.

[3] Gerard Sierksma; Yori Zwols (2015). Linear and Integer Optimization: Theory and Practice (3rd ed.). CRC Press. p. 1. ISBN 978-1498710169.

[4] Kaufmann, M., & Wiese, R. (1997). Simulated annealing for coloring random graphs. Journal of Statistical Physics, 87(5–6), 1207–1225. doi:10.1007/BF02181453.

[5] Pincus, Martin (Nov–Dec 1970). "A Monte-Carlo Method for the Approximate Solution of Certain Types of Constrained Optimization Problems". Journal of the Operations Research Society of America. 18 (6): 967–1235. doi:10.1287/opre.18.6.1225.

[6] Trudeau, Richard J. (1993). Introduction to Graph Theory (Corrected, enlarged

republication. ed.). New York: Dover Pub. p. 19. ISBN 978-0-486-67870-2. Archived from the original on 5 May 2019. Retrieved 8 August 2012. A graph is an object consisting of two sets called its vertex set and its edge set.

[7] Wikipedia contributors. (2023, February 11). Adjacency matrix. Retrieved from Wikipedia, The Free Encyclopedia website: https://en.wikipedia.org/w/index.php?title=Adjacency_matrix&oldid=1138782609.

[8] Wikipedia contributors. (2023b, February 18). Henan. Retrieved from Wikipedia, The Free Encyclopedia website: https://en.wikipedia.org/w/index.php?title=Henan&oldid=1140159930.

[9] Barnette, D. (1983). Map Coloring, Polyhedra, and the Four-Color Problem. Providence, RI: Mathematical Association of America.

[10] Wikipedia contributors. (2023d, March 2). List of optimization software. Retrieved from Wikipe -dia, The Free Encyclopedia website: https://en.wikipedia.org/w/index.php?title=List_of_opt -imization_software&oldid=1142468626.