Integer Programming Approach to Graph Colouring Problem and Its Implementation in GAMS

MILOŠ ŠEDA
Institute of Automation and Computer Science
Brno University of Technology
Technická 2, 616 69 Brno
CZECH REPUBLIC

Abstract: - The graph colouring problem is one of the most studied combinatorial optimisation problems, one with many applications, e.g., in timetabling, resource assignment, team-building problems, network analysis, and cartography. Because of its NP-hardness, the question arises of its solvability for larger instances. Instead of the traditional approaches based on the use of approximate or stochastic heuristic methods, we focus here on the direct use of an integer programming model in the GAMS environment. This environment makes it possible to solve instances much larger than in the past. Neither does it require complex parameter settings or statistical evaluation of the results as in the case of stochastic heuristics because the computational core of software tools, nested in GAMS, is deterministic in nature.

Key-Words: - graph colouring, independent set, NP-hard problem, integer programming, GAMS

Received: August 21, 2022. Revised: April 11, 2023. Accepted: April 25, 2023. Published: May 26, 2023.

1 Introduction

The goal of the graph colouring problem is to find the minimum number of colours assigned to each vertex such that no two vertices connected by an edge have the same colour. Due to the large number of applications in various fields of science, this problem is not missing in any book dealing with graph theory, e.g., in monographs, [7][7], [15], [34].

Special cases of graph colouring under specific constraints are studied in [6], [17], [25]. The paper, [29], deals with edge colouring, [24], presents a probabilistic approach, [9], proposes an algorithm running in $O(n^3)$ time, several algorithms are compared in [3], applications are addressed in [16], [33].

Heuristic methods, [30], and approximation algorithms, [20], are used for large instances of graph colouring problems, e.g., genetic algorithm, [2], [10], [12], simulated annealing, [26], local search, [5], memetic algorithm, [8], [13], and also nature-inspired algorithms, e.g., particle swarm optimization, [1], [27], and ant colony optimization, [11].

However, the exact methods, [14], [18], [35], integer programming approach, [23], branch-and-cut algorithm, [21], cutting plane algorithm, [22], [28], are also applicable, so we will again focus on the usability of the GAMS tool, which has already been successfully used in previous papers to solve the Permutation Flow Shop Scheduling Problem, the Traveling Salesman Problem, [31], and the Steiner Tree Problem in Graphs, [32].

2 Basic Notions

Definition 1

- Let $G=(V, E)$ be a graph. A *colouring* of the vertices of G is an assignment of colours to the vertices in such a way that adjacent vertices of G have distinct colours.
- The (*vertex*) *chromatic number* of G, denoted $\chi(G)$, is the minimum number of different colours required for a proper colouring of G.
- A graph is (*vertex*) *r-colourable* if $\chi(G) = r$. □

Theorem 1 (*W. Haken* and *K. Appel*)
Every planar graph is 4-colourable.

Proof. The proof, based on modelling of 1900 basic configurations, is not generally accepted, and the theorem is considered a hypothesis. ■

If the degree of a vertex v, denoted $d(x)$, is the number of edges incident on v, then the following theorem is satisfied.

Theorem 2
In a simple graph G without self-loops
$\chi(G) \leq 1 + \max\{d(x) \mid x \in V(G)\}$.

Proof. Let us denote $d_{max} = \max \{d(x) \mid x \in V(G)\}$.
1. Order the vertices in a sequence $x_1, x_2, \ldots, x_n$.
2. Assign colour 1 to the vertex x_1.
3. Vertices x_i, $i = 2, 3, \ldots, n$ are coloured by a colour that represents the lowest number unused by the coloured neighbours of the vertex x_i.
Since the degree of each vertex, i.e., the number of its neighbours cannot be higher than d_{max}, among the $d_{max}+1$ colours, there must be at least one colour b not used by any of the neighbours. Since it is satisfied for all vertices, $d_{max}+1$ colours are sufficient for graph colouring. ■

The assertion of the theorem is very loose, e.g., in the graph in Fig. 1, the maximum vertex degree is equal to 5, and only 2 colours are needed to colour it. This result follows from Theorem 4 because the graph in Fig. 1 is a tree.

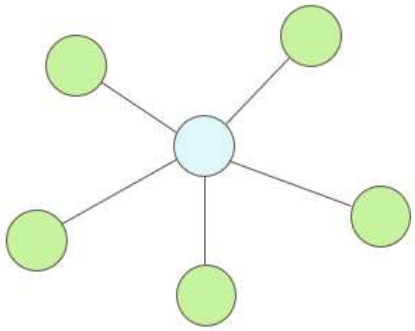

Fig. 1: Planar graph with $d_{max}=5$ and 2 colours

Definition 2

- A graph G is *bipartite* if $V(G)$ can be partitioned into two (nonempty) subsets V_1 and V_2 such that every edge of G joins a vertex of V_1 with a vertex of V_2.
- A *complete bipartite graph* is a bipartite graph such that each vertex of V_1 is adjacent to every vertex of V_2. Let $|V_1|=m$ and $|V_2|=n$. Then, this graph is denoted by $K_{m,n}$. □

Theorem 3
Let G be a graph. The following are satisfied:
1. $\chi(G)=1 \Leftrightarrow$ *G has no edge.*
2. $\chi(G)=2 \Leftrightarrow$ *G does not contain a cycle with an odd number of edges.*
3. $\chi(G)=2 \Leftrightarrow$ *G is bipartite.*

Proof.
1. This is straightforward.
2. [$L \Rightarrow R$] The chromatic number of each cycle with an odd number of edges is 3. Hence, if $\chi(G)=2$, then G cannot contain a cycle with an odd number of edges.
2. [$L \Leftarrow R$] Assume that G does not contain a cycle with an odd number of edges. We will colour it as follows: (a) We select an arbitrary vertex x, (b) vertices with an *odd* distance from x (in the sense of the number of edges on the shortest path) will be coloured by 1, (c) vertices with an *even* distance by 2.

Now we must prove that we will get a colouring, i.e., no pair of vertices with the same colour is connected.

Consider 2 vertices u, v that both have an even or an odd distance from the vertex x, i.e., they have the same colour.

Thus, the path $u - x - v$ has an even distance because both the double of an even number and the double of an odd number are even. If vertices u, v were to be connected by an edge, then this edge together with the path from u to v would create a cycle of an odd length (1+ *even* = *odd*), contradiction.

3. [$L \Rightarrow R$] Each colouring with 2 colours determines a decomposition of the set of vertices into 2 sets where $W_G(A)=W_G(B)=E(G)$ and, thus, the graph is bipartite.

[$L \Leftarrow R$] If a graph is bipartite, then there is a decomposition mentioned above and it is sufficient to colour each of these two sets by only one colour. ■

Theorem 4
Trees have $\chi(G)=2$.

Proof. By definition, trees do not contain cycles and this implies that trees do not contain cycles of odd lengths. Hence, we get by the second assertion of Theorem 3 that trees are 2-colourable. ■

3 Algorithms

The basic algorithm is based on successively assigning the lowest possible colour number to the not yet coloured vertices, so that the colour of any of the neighbouring, already coloured vertices, is not used.

Another approach consists in the sequential construction of maximal independent sets (a vertex set is *independent* if no two of its vertices are connected by an edge). The number of independent sets corresponds to the chromatic number, and vertices from the same independent set have the same colour.

We will not deal with these algorithms (or any other algorithms) or vertex selection strategies for colouring because we will use the professional optimization tool GAMS.

In terms of time complexity, in the first step of the assignment-based colouring algorithm, we select from n uncoloured vertices, in the second step from the number of vertices reduced by the already coloured vertices, and so on until all vertices are coloured. The upper time complexity of this procedure, i.e., the worst case where only one randomly selected vertex is coloured in each step (this situation would occur in a complete graph), is expressed by $n(n–1)(n–2)\ldots 2.1=O(n!)$.

4 Integer Programming Model for Vertex Graph Colouring

The classical integer programming model for a graph $G = (V, E)$, described, e. g., in [14], is based on assigning colour c to vertex $v \in V$.

Denote by x_{vc} the assignment variables for each vertex v and colour c (c = 1, …, H), where H is an upper bound of the number of colours. Here, x_{vc} = 1 if vertex v is assigned to colour c and x_{vc} = 0 otherwise. Binary variables w_c get a value of 1 if and only if colour c is used in the colouring (c = 1, … , H).

The model is given as follows:

Minimise

$$\sum_{1\le c\le H} w_c \tag{1}$$

subject to

$$\sum_{c=1}^{H} x_{v_1 c} = 1, \quad \forall v_1 \in V \tag{2}$$

$$x_{v_1 c} + x_{v_2 c} \le w_c, \ \forall (v_1, v_2) \in E, \ c \in 1, \ldots, H \tag{3}$$

$$w_c \le \sum_{v_1 \in V} x_{v_1 c}, \ c \in 1, \ldots, H \tag{4}$$

$$w_c \le w_{c-1}, \ c \in 2, \ldots, H \tag{5}$$

$$x_{v_1 c} \in \{0, 1\}, \ \forall v_1 \in V, \ c \in 1, \ldots, H \tag{6}$$

$$w_c \in \{0, 1\}, \ c \in 1, \ldots, H \tag{7}$$

However, this model cannot be used directly in GAMS as it does not give correct results without a modification described in the following section.

5 Implementation in GAMS

Equation (3) in the integer programming model says that only the edges of the graph are to be calculated, therefore, in GAMS, the multiplication by the term $E(v_1, v_2)$ is added to the corresponding equation, where only the edges, i.e., those values in the matrix that are equal to 1, are applied. This then gives the correct result of 4 colours for the graph in Figure 2. Even this modified model is not fully functional for a complete graph. Obviously, a complete graph with n vertices will need n colours for vertex colouring. Adding the condition $v_1 \neq v_2$ to Equation (3) will provide the correct function.

Thus, the model described by Equations (1)-(7) changes as follows:

Minimise

$$\sum_{1\le c\le H} w_c \tag{8}$$

subject to

$$\sum_{c=1}^{H} x_{v_1 c} = 1, \quad \forall v_1 \in V \tag{9}$$

$$\left(x_{v_1 c} + x_{v_2 c}\right) e_{v_1 v_2} \le w_c, \ \forall v_1, v_2 \in V, v_1 \neq v_2, \ c \in 1, \ldots, H \tag{10}$$

$$w_c \le \sum_{v_1 \in V} x_{v_1 c}, \ c \in 1, \ldots, H \tag{11}$$

$$w_c \le w_{c-1}, \ c \in 2, \ldots, H \tag{12}$$

$$x_{v_1 c} \in \{0, 1\}, \ \forall v_1 \in V, \ c \in 1, \ldots, H \tag{13}$$

$$w_c \in \{0, 1\}, \ c \in 1, \ldots, H \tag{14}$$

The code in GAMS is as follows:

```
$TITLE Vertex colouring
$OFFSYMXREF
$OFFUELLIST
$OFFUELXREF

* section defining indexes
SETS
  V1 vertices  /1*21/;
  ALIAS(V1,V2);
  ALIAS(V1,C);

* input data section
PARAMETER
  H;
  H=CARD(V1);
* H = an upper bound of the number of the colours, at most |V|
* E - adjacency matrix


TABLE E(V1,V2)
      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
  1   0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  2   1  0  1  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0  0
  3   1  1  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  4   1  0  1  0  1  1  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0
  5   0  0  1  1  0  1  1  0  0  0  0  1  1  0  0  0  0  0  0  0  0
  6   0  0  0  1  1  0  1  1  0  0  0  0  1  1  0  0  0  0  0  0  0
  7   0  0  0  0  1  1  0  1  0  0  1  1  0  0  0  0  0  0  0  0  0
  8   0  0  0  1  0  1  1  0  1  1  1  0  0  0  0  0  0  0  0  0  0
  9   0  0  0  1  0  0  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0
 10   0  0  0  0  0  0  0  0  1  1  0  1  0  0  0  0  0  0  1  0  0
 11   0  0  0  0  0  0  0  1  1  0  1  0  1  0  0  1  0  0  1  0  1
 12   0  0  0  0  1  0  1  0  0  0  1  0  1  0  1  1  0  0  0  0  0
 13   0  1  0  0  1  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0
 14   0  1  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0  0  0
 15   0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  1  1  1  1  0  0
 16   0  0  0  0  0  0  0  0  0  0  0  1  1  1  1  0  1  0  0  0  0
 17   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  1  0  0
 18   0  0  0  0  0  0  0  0  0  0  1  0  0  0  1  0  0  0  1  1  1
 19   0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  1  0  1  0
 20   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  1  0  1
 21   0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0  0  1  0  1  0;
```

```
* variables section (decision variables and objective function)
VARIABLES
  X(V1,C)
* X(V1,C) = 1 if vertex V1 is assigned to color C and
* X(V1,C) = 0 otherwise
  W(C)
* W(C) = 1 if color C is used in the colouring (i = 1, . . . , H).
  Cmin objective function (minimal number of colours);
  BINARY VARIABLE X;
  BINARY VARIABLE W;

*section describing the system of (in)equalities
EQUATIONS
  EQ2(V1)
  EQ3(V1,V2,C)
  EQ4(C)
  EQ5(C)
  OBJFCE number of colours;

  EQ2(V1)         .. SUM(C,X(V1,C)) =E= 1;
  EQ3(V1,V2,C)$(ORD(V1) NE ORD(V2))
                  .. (X(V1,C)+ X(V2,C)-W(C))*E(V1,V2) =L= 0;
  EQ4(C)          .. W(C)-SUM(V1,X(V1,C)) =L= 0;
  EQ5(C)$(ORD(C) GE 2)   .. W(C)-W(C-1) =L= 0;
  OBJFCE          .. Cmin =E= SUM(C,W(C));

*description of the model, running the solver and displaying
 *the results
MODEL COLOURING /ALL/;
SOLVE COLOURING USING MIP MINIMIZING Cmin;
DISPLAY X.L, W.L, Cmin.L;
```

Fig. 2 shows a graph for which Fig. 3 shows the solution of the vertex colouring obtained by the GAMS calculation.

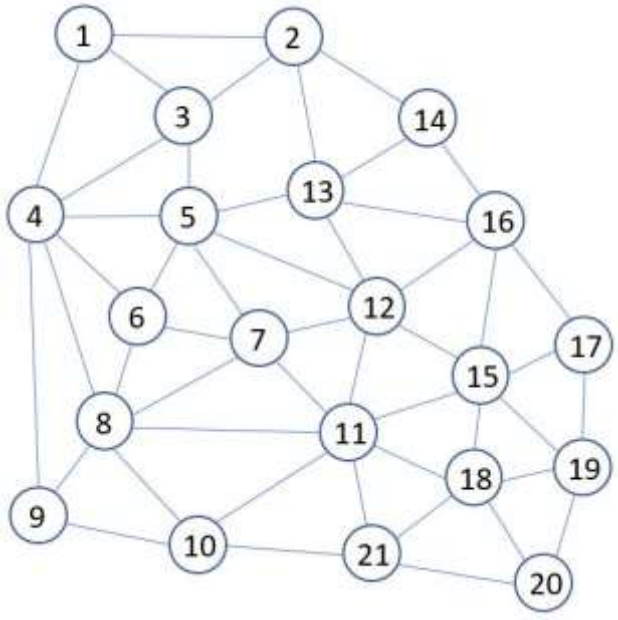

Fig. 2: Planar graph with 21 vertices

The complete graph could be specified in GAMS by explicitly specifying all elements of the matrix E with values 1, but it is easier to fill this matrix in a double loop as follows:

```
TABLE E(V1,V2);
LOOP(V1,
        LOOP(V2,
               E(V1,V2)=1;
            );
    );
```

For a complete graph with 100 vertices, a solution of 100 colours is obtained in just 0.52 sec.

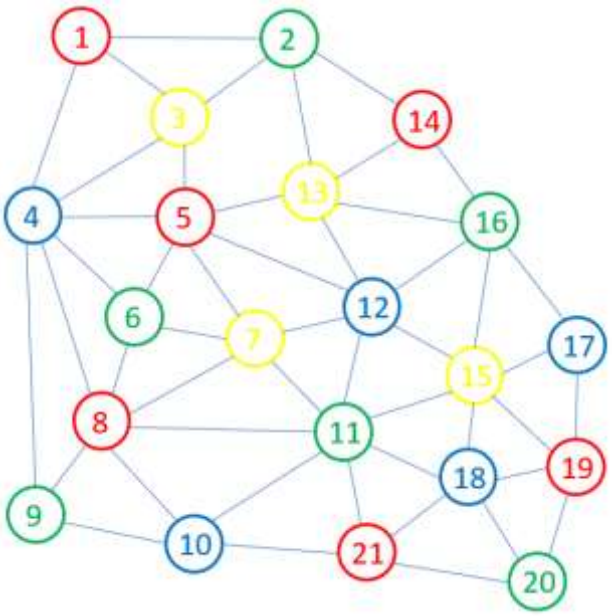

Fig. 3: Vertex colouring (for the graph from Fig. 2: Planar graph with 21 verticesFig. 2), given by the computation in GAMS

6 Computational Results

The ability to compute optimal solutions was checked using standard benchmarks from OR-Library (OR = Operations Research) accessible at Brunel University London, [4],

Table 1. Computational results

benchmark	$\|V\|$	$\|E\|$	time [sec]	$\chi(G)$
huck.col	74	602	0.41	11
jean.col	80	508	0.66	10
david.col	87	812	0.98	11
games120.col	120	1276	19.28	9
anna.col	138	986	5.30	11
fpsol2.i.3.col	425	8688	*	*

Table 1 summarizes the results of computing graph colourings for test problems with given counts of vertices and edges, computational time,

and chromatic number. It can be seen that, even for such large instances, GAMS can get an accurate solution in a time of a few seconds at most on a laptop with Intel(R) Core (TM) 389 i5-10210U CPU @ 1.60 GHz 2.11 GHz processor, 8 GB operational memory and 64-bit 390 operating system.

Only for an extremely large instance of fpsol2.i.3.col, GAMS stopped the calculation at 1000.22 sec by exceeding the time limit without finding a solution. As for the time complexity of the computation, it cannot be formally expressed exactly in Big *O* notation because the implementation of the GAMS computational kernel is not freely available.

7 Conclusion

This paper studies the graph colouring problem by an integer programming approach, which, after some modifications, can be transferred to the GAMS environment.

It has also shown that the optimal solution can be determined in the available time of a few minutes for instances with more than a hundred vertices and hundreds of edges. Previously, these boundaries were inaccessible with integer programming solvers, but with the new version of GAMS, they have been significantly extended. This of course means that, first, an appropriate model must be used and then search individually for the appropriate bounds.

In future research, we expect to focus on another NP-hard problem, namely, finding the maximum clique in a graph.

References:

[1] J. Agrawal and S. Agrawal, Acceleration Based Particle Swarm Optimization for Graph Coloring Problem, *Procedia Computer Science*, Vol. 60, 2015, pp. 714-721.

[2] J, Aguilar-Canepa1, R. Menchaca-Mendez, R. Menchaca-Mendez and J. García, A Structure-Driven Genetic Algorithm for Graph Coloring, *Computación y Sistemas*, Vol. 25, No. 3, 2021, pp. 465–481.

[3] M. Aslan and N. A. Baykan, A Performance Comparison of Graph Coloring Algorithms, *International Journal of Intelligent Systems and Applications in Engineering*, Vol. 4, 2016, pp. 1-7.

[4] J. E. Beasley. OR-Library. Report, Brunel University London. 2018. Available online: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/colourinfo.html.

[5] M. Caramia, P. Dell'Olmo and G. F. Italiano - CHECKCOL: Improved Local Search for Graph Coloring, *Journal of Discrete Algorithms*, Vol. 4, 2006, pp. 277-298.

[6] C. Charpentier, H. Hocquard, E. Sopena and X. Zhu, A Connected Version of the Graph Coloring Game, *Discrete Applied Mathematics*, Vol. 283, No. 9, 2020, 9 pp.

[7] R. Diestel, *Graph Theory*. Springer-Verlag, Berlin, 2005.

[8] T. Dokeroglu and E. Sevinc, Memetic Teaching-Learning-Based Optimization Algorithms for Large Graph Coloring Problems, *Engineering Applications of Artificial Intelligence*, Vol. 102, 2021, 104282.

[9] V. Donderia and P. K. Jana, A Novel Scheme for Graph Coloring, *Procedia Technology*, Vol. 4, 2012, pp. 261-266.

[10] R. Dorne and J.-K. Hao, A New Genetic Local Search Algorithm for Graph Coloring, *Lecture Notes in Computer Science*, Vol. 1498, 2006, pp. 745–754.

[11] K. A. Dowsland, and J. M. Thompson, An Improved Ant Colony Optimisation Heuristic for Graph Colouring, *Discrete Applied Mathematics*, Vol. 156, No. 3, 2008, pp. 313-324

[12] S. M. Douiri and S. Elbernoussi, Solving the Graph Coloring Problem via Hybrid Genetic Algorithms, *Journal of King Saud University - Engineering Sciences*, Vol. 27, 2015, pp. 114-118

[13] O. Goudet, C. Grelier and J.-K. Hao, Deep Learning Guided Memetic Framework for Graph Coloring Problems, *Knowledge-Based Systems*, Vol. 258, 2022, 109986.

[14] A. Jabrayilov and P. Mutzel, New Integer Linear Programming Models for the Vertex Coloring Problem, In *Proceedings of the 42nd Conference on Very Important Topics* (*CVIT 2016*), 23 pp.

[15] T. R. Jensen and B. Toft, *Graph Coloring Problems*, John Wiley & Sons, New York, 2011.

[16] R. M. R. Lewis, *A Guide to Graph Colouring. Algorithms and Applications*. Springer-Verlag, Berlin, 2016.

[17] C. Konrad and V. Zamaraev. Distributed Minimum Vertex Coloring and Maximum Independent Set in Chordal Graphs, *Theoretical Computer Science*, Vol. 922, 2022, pp. 486-502.

[18] C. Lucet, F. Mendes and A. Moukrim, An Exact Method for Graph Coloring, *Computers & Operations Research*, Vol. 33, 2006, pp. 2189-2207.

[19] E. Malaguti, M. Monaci and P. Toth, An Exact Approach for the Vertex Coloring Problem, *Discrete Optimization*, Vol. 8, No. 2, 2011, pp. 174-190.

[20] R. Marappan, G. Sethumadhavan and R. K. Srihari, New Approximation Algorithms for Solving Graph Coloring Problem. An Experimental Approach, *Perspectives in Science*, Vol. 8, 2016, pp. 384-387.

[21] I. Méndez-Díaz and P. Zabala, A Branch-and-Cut Algorithm for Graph Coloring, *Discrete Applied Mathematics*, Vol. 154, No. 5, 2006, pp. 826–847.
[22] I. Méndez-Díaz and P. Zabala, A Cutting Plane Algorithm for Graph Coloring, *Discrete Applied Mathematics*, Vol. 156, No. 2, 2008, pp. 159-179.
[23] I. Méndez-Díaz and P. Zabala, Solving a Multicoloring Problem with Overlaps Using Integer Programming, *Discrete Applied Mathematics*, Vol. 158, No. 4, 2010, pp. 349-354
[24] M. Molloy and B. Reed, *Graph Colouring and the Probabilistic Method*, Springer-Verlag, Berlin, 2002.
[25] K. Nishikawa, T. Nishizeki and X. Zhou, Bandwidth Consecutive Multicolorings of Graphs, *Theoretical Computer Science*, Vol. 532, 2014, pp. 64-72.
[26] A. J. Pal, B. Ray, N. Zakaria and S. S. Sarma, Comparative Performance of Modified Simulated Annealing with Simple Simulated Annealing for Graph Coloring Problem, *Procedia Computer Science*, Vol. 9, 2012, pp. 321-327.
[27] J. Qin, Hybrid Discrete Particle Swarm Algorithm for Graph Coloring Problem, *Journal of Computers*, Vol. 6, No. 6, 2011, pp. 1175-1182.
[28] O. Şeker, T. Ekim, and Z. C. Taşkin, An Exact Cutting Plane Algorithm to Solve the Selective Graph Coloring Problem in Perfect Graphs, *European Journal of Operational Research*, Vol. 291, No. 1, 2021, pp. 67-83.
[29] M. Stiebitz, D. Scheide, B. Toft and L. M. Favrholdt, *Graph Edge Coloring. Vizing's Theorem and Goldberg's Conjecture*, John Wiley & Sons, New York, 2012.
[30] W. Sun, *Heuristic Algorithms for Graph Coloring Problems*, PhD Thesis, l'Universite d'Angers, Comue Universite Bretagne Loire, 2018.
[31] M. Šeda, The Assignment Problem and Its Relation to Logistics Problems, *Algorithms*, Vol. 15, No. 10, 2022, (article number 377), 27 pp.
[32] M. Šeda, Steiner Tree Problem in Graphs and Mixed Integer Linear Programming-Based Approach in GAMS, *WSEAS Transactions on Computers*, Vol. 21, 2022, pp. 257-262.
[33] S. Thadani, S. Bagora and A. Sharma, Applications of Graph Coloring in Various Fields, *Materials Today: Proceedings*, Vol. 66, 2022, pp. 3498-3501.
[34] D. de Werra and A. Hertz, *Graph Colouring and Variations*, North-Holland, Amsterdam, 1989.
[35] Z. Zhou, C.-M. Li, C. Huang and R. Xu, An Exact Algorithm with Learning for the Graph Coloring Problem, *Computers & Operations Research*, Vol. 51, 2014, pp. 282-301.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

Miloš Šeda modified the classical integer programming model for vertex graph colouring, implemented it in GAMS and proved its applicability to large instances of the problem.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Conflict of Interest

The author declares that there is no conflict of interest.