Iranian Journal of Numerical Analysis and Optimization
Vol. 8, No. 2, (2018), pp 39–54
DOI:10.22067/ijnao.v8i2.61465

New variable neighborhood search method for minimum sum coloring problem on simple graphs

Kh. Erfani*, S. Rahimi and J. Fathali

Abstract

The minimum sum coloring problem (MSCP) is to find a legal vertex coloring for G using colors represented by natural numbers $(1, 2, \dots)$ such that the total sum of the colors assigned to the vertices is minimized. The aim of this paper is to present the skewed variable neighborhood search (SVNS) for this problem based on a new structure of neighborhoods. To increase the speed of the neighborhood search process, we present the new concepts of holder vertex and set. Tested on 23 commonly used benchmark instances, our algorithm shows acceptable competitive performance with respect to recently proposed heuristics.

Keywords: Minimum sum coloring; Variable neighborhood search; Skewed variable neighborhood search; Chromatic sum; Holder vertex; Holder set; Reducer set.

1 Introduction

The general graph coloring is one of the most well-known problems in combinatorial optimization. Besides its theoretical significance as a canonical NP-Hard problem [9], graph coloring arises naturally in a variety of real-world

*Corresponding author
Received 31 December 2016; revised 17 June 2017; accepted 28 February 2018
Kh. Erfani
Department of Applied Mathematics, Shahrood University of Technology, Shahrood, Iran.
e-mail: khalilerfani@gmail.com

S. Rahimi
Department of Applied Mathematics, Shahrood University of Technology, Shahrood, Iran.
e-mail: srahimi40@yahoo.co.uk

J. Fathali
Department of Applied Mathematics, Shahrood University of Technology, Shahrood, Iran.
e-mail: fathali@shahroodut.ac.ir

applications such as timetable problems [3], warehouse management [29], frequency allocation in mobile network [28], register allocation in optimizing compilers [4], scheduling problem [8], design and operation of exible manufacturing systems [10]. For example, in computer systems with multiprocessors that are in competition over resources, we might seek an allocation under which no two jobs with conflicting requirements are executed simultaneously while minimizing the average completion time of the jobs.

MSCP is closely related to the basic graph coloring problem. It was proposed by Kubicka [21] in the field of graph theory and by Supowit [30] in the field of VLSI design.

Suppose that $G = (V, E)$ is a simple undirected graph (without loop, multi, and directed edge) with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and edge set $E \subseteq V \times V$. A proper vertex k-coloring of G is an assignment of positive integers to its vertices by function $c : V \to \{1, 2, \ldots, k\}$ such that no two adjacent vertices are assigned the same number (i.e. $c(v_i) \neq c(v_j), \forall (i, j) \in E$). The color of a vertex v is denoted by $c(v)$.

A legal k-coloring can also be defined as partition of V in to k independent sets $V_1, V_2, \ldots, V_k$ such that for all $u, v \in V_i (i = 1, \ldots, k), (u, v) \notin E$. In other word, c can also be represented as a partition of V in to k mutually disjoint independent set (called color classes) $V_1, V_2, \ldots, V_k$ such that $\bigcup_{i=1}^{n} V_i = V$ and $v \in V_i$ if and only if $c(v) = i$.

The general graph coloring problem (GCP) is to determine a proper k-coloring with a minimum value of k. This value is called the chromatic number of G and denoted by $\chi(G)$. A related problem to the GCP is the minimum sum coloring problem (MSCP), which is to find a proper coloring $c = \{V_1, V_2, \ldots, V_k\}$ such that the following total sum of color labels, denoted by $\sum_c G$, to be minimized.

$$\sum_c G = \sum_{i=1}^{n} c(v_i) = \sum_{i=1}^{k} i|V_i|.$$

Now the vertex chromatic sum or minimum sum coloring of G is denoted by $\sum G$, and defined as

$$\min \left\{ \sum_c G \mid c \in C \right\},$$

where C is the collection of all proper coloring of G. The vertex strength of G denoted by $s(G)$, is the smallest number s, where there is a proper coloring c with s colors such that $\sum_c(G) = \sum G$. It is clear that $s(G)$ is lower bounded by $\chi(G)$. The distance between $s(G)$ and $\chi(G)$ can be large. Indeed there are many instances in which $s(G) >> \chi(G)$ [21].

As shown in [20, 21], the decision version of the MSCP is NP-complete in the general case. As a result, solving the MSCP is computationally challenging and any algorithm able to determine the optimal solution of the problem

is expected to require an exponential complexity. Also, it has several practical applications including VLSI design, scheduling, and distributed resource allocation (see [24] for a list of references for more application).

Due to its high computational complexity, no polynomial-time algorithm can solve or approximate the problem efficiently unless $P = NP$. In the past several decades, much effort has been devoted to developing various heuristic and metaheuristic algorithms. For the purpose of practical solving of the general MSCP, several heuristic algorithms have recently been proposed to find suboptimal solutions. This class of algorithms has been mainly developed since 2009 and is not proved the optimality of the solution found. Some examples of the heuristic algorithms include tabu search [2], greedy algorithms [23], genetic and memetic algorithms [5, 17–19, 26, 32], breakout local search [1], iterated local search [15] , ant colony [6] as well as heuristics based on independent set extraction [33, 34]. But in contrast, some kinds of literature have addressed the issue of theoretical bounds that formally proved, included [7, 11, 12, 16, 22, 31] for instance.

In recent years, variable neighborhood search (VNS) has been proven as a very effective and adaptable metaheuristic, used for solving a wide range of complex optimization problems. In this paper, we design and test a VNS algorithm for finding proper colorings, which correspond to upper bounds of the chromatic sum. In the literature on graph coloring several proposals appeared, combining the following characteristics: partial vs complete colorings, proper vs improper colorings, fixed vs variable k (see [15] for a review).

We opt vector X with size of $|V|$ for a solution representation consisting of colors that must be proper. We then devise a search strategy that borrows ideas from modified VNS method called skewed VNS method (SVNS) towards special kind of neighborhood. The fact that increasing the number of colors may decrease the chromatic sum was the main motivation of definition of such new neighborhood structure (see Fig. 1).

In the following sections, we first provide integer nonlinear mathematical formulation along with a definition of N_k neighborhood structure of MSCP. In the next section, we offer a new concept called *holding* to expedite the related local search method. In Section 4 we present our SVNS method with respect to the neighbors that are mentioned in Section 2. Before concluding, detailed computational results and comparisons with five algorithms are presented in Section 5.

2 Modeling and N_k Neighborhoods of MSCP

2.1 Modeling

Let $G = (V, E)$ be a simple undirected graph with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and edge set $E \subseteq V \times V$. Although the MSCP can be formulated as a binary quadratic problem, we also presented another formulation of this problem. Define variable x_i as the color of vertex i for $i = 1, 2, \ldots, n$. Therefore the integer nonlinear mathematical formulation is defined as follows:

$$\begin{aligned} Min \quad & \sum_{i=1}^{n} x_i \\ s.t. \quad & \\ & |x_i - x_j| \geq 1 \qquad : \forall (i,j) \in E \qquad (1) \\ & 1 \leq x_i \leq n \quad , integer \quad : i = 1, 2, \ldots, n \qquad (2) \end{aligned}$$

The previous model can be reduced with elimination of redundant constraints and suitable changing of variables,

$$\begin{aligned} P) \quad Min \quad & \sum_{i=1}^{n} x_i \qquad (3) \\ s.t. \quad & \\ & |x_i - x_j| \geq 1 \qquad : \forall (i,j) \in E \qquad (4) \\ & x_i \geq 0, integer \quad : i = 1, 2, \ldots, n \qquad (5) \end{aligned}$$

This mathematical formulation expresses MSCP correctly, since each vector $X = (x_1, x_2, \ldots, x_n)$ that satisfies equations (4) and (5) is a proper coloring c for G and the objective function (3) minimizes $\sum_c G$. $\sum(X)$ denotes the sum of colors of proper coloring c. It should be noted that this model does not claim about $s(G)$ and focuses on $\sum G$. By assuming that S be set of all proper coloring of G, P can be written briefly as follows:

$$\begin{aligned} P) \quad Min \quad & \overrightarrow{1}.X \qquad (6) \\ s.t. \quad & \\ & X \in S, \qquad (7) \end{aligned}$$

where $\overrightarrow{1}.X$ is the inner product of $\overrightarrow{1} = (1, 1, \ldots, 1)$ and X. Indeed $\sum(X) = \sum_{i=1}^{n} x_i = \overrightarrow{1}.X$. According to the description stated above, any feasible solution for P is a vector $X = (x_1, x_2, \ldots, x_n)$ satisfied equations (4) and (5). So the next neighborhoods should be defined around this vector.

2.2 New Neighborhoods for MSCP

We know that the neighborhood is an important element that influences the local search procedure. So, searching for feasible spaces is meaningless without knowing the definition of the neighborhood. Therefore according to the mathematical model of the previous part, we introduce neighboring structure for MSCP here. To read more about neighborhoods for this problem you can see [1, 14, 15, 17].

Definition 1. Let X and S be a feasible solution and feasible space of P, respectively. For $k = 0, 1, \ldots, n$, a neighborhoods of X are defined by

$$N_k(X) = \{Y \in S \mid Y \ \textit{is greater than } X \textit{ in at most } k \textit{ element.}\} \quad (8)$$

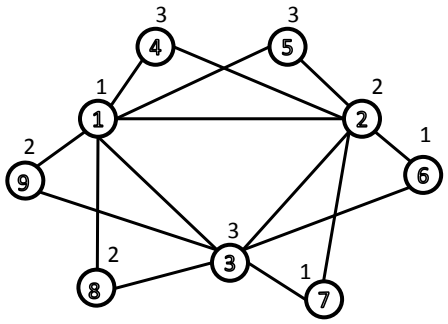
In fact, $N_k(X)$ contains all of the feasible solutions (proper coloring) that are greater than X in the k element at most. The size of the $N_k(.)$ becomes bigger if k increases. Indeed $|N_k(.)| \in O(\binom{n}{k})$ and we have

$$N_0(X) \subseteq N_1(X) \subseteq \cdots \subseteq N_k(X)$$

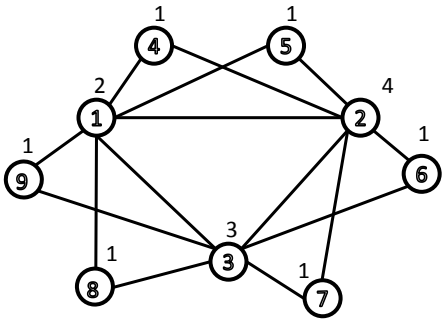
As you can see, these neighborhoods will be nested; that is, each one contains the previous. So if X is locally optimal with respect to N_k, then it will be locally optimal with respect to N_i for $i = 0, 1, \ldots, k-1$. Also, if it is not to be locally optimal w. r. t. N_i, then it will not be locally optimal w. r. t. N_i for $i = k+1, \ldots, n$. Clearly $S = N_n(X)$. But it's important to know the minimum value of k such that $N_k = S$. In other words, for which value of k the neighborhood N_k is exact? For a given neighborhood N if any local optimal is also global optimal, then N is exact [27].

To clarify the issue, Figure 1 shows an illustrative example for $N_k(.)$. The numbers that are in and out the vertices are vertex number and vertex color, respectively. With the given 3-coloring X (left figure), we achieve the suboptimal sum of 18 while Y leads to chromatic sum of 15 (right figure). By Definition 1, $N_0(X) = \{X\}$ and $Y \notin N_0(X), N_1(X)$ but $Y \in N_k(X) for all k \geq 2$. Therefore by starting with X, we achieve the locally optimal Y w. r. t. N_2, that is also globally optimal.

The next important point that should be noted is about the size of N_k. As stated previously by increasing k the size of neighborhood becomes large and this causes the process of searching happens slowly. Therefore we will offer a new concept called *holding* and try to expedite to find the local optimum with this new concept.



(a) $X = (1, 2, 3, 3, 3, 1, 1, 2, 2)$



(b) $Y = (2, 4, 3, 1, 1, 1, 1, 1, 1)$

Figure 1: An illustrative example for $N_k(.)$

3 Holder and Reducer Set

Definition 2. ***k-holder set:*** Given a proper coloring $X = (x_1, \ldots, x_n)$ and k-element subset $H = \{t_1, t_2, \ldots, t_k\}$ from V. H is a k-holder set, when for some $v \in N(H)$[1] and index $0 \leq r \leq k$ the following two conditions hold:

1) $x_{t_r} < x_v$;

2) For any vertex $w \in N(v) \setminus H$, we have $x_w \neq x_{t_r}$.

In such case, we say that H holds vertex t_r.

Especially in above definition when $k = 1$, we obtain 1-holder set called *holder vertex*. Indeed, for two adjacent vertices a and b, we say a holds b and denoted by $a \nearrow b$, if Definition 1 is stated for $k = 1$. Consider again the graph of Figure 1a with 3-coloring X. Both two sets $H_1 = \{1\}$ and $H_2 = \{2\}$ are holder vertices according to Definition 2. H_1 is a *good holder vertex* because H_1 holds $4, 5, 9, 8$ and when its color is increased the objective function

[1] Denote the set of adjacent vertices of H

is reduced three units. In contrast H_2 is not a good one because no change does happen when its color starts to increase. Thus, the holding feature not sufficient to improve the objective function and should have *reduction* property.

Definition 3. ***k-reducer set:*** Given a proper coloring X. The k-holder set H is a k-reducer set, when the objective function is improved (reduced) by increasing its color of vertices.

The above definitions lead us to the following obvious propositions, which more specifies the relationship between N_k and these two new concepts.

Proposition 1. *The proper coloring X is locally optimal w.r.t. N_k if and only if G has no k-reducer set.*

Proposition 2. *The proper coloring X is a global optimal if and only if there exists $k_0 \in \mathbb{Z}^+$ such that G has no k-reducer set w.r.t. X for any $k \geq k_0$.*

For a given solution X, if G has no k-holder set for any $k \in \mathbb{Z}^+$, then it has no k-reducer set for any $k \in \mathbb{Z}^+$ and so according to Proposition 2 X is globally optimal.

4 VNS Method with respect to N_k for MSCP

Variable neighborhood search (VNS) algorithm was originally described by Mladenovic and Hansen ([13,25]). The basic strategy of the VNS is to focus the investigation of the solutions, which belong to some neighborhood of the current best one. In order to avoid being trapped in local suboptimal solutions, VNS changes the neighborhoods, directing the search in the promising and unexplored areas. By this systematic change of neighborhoods, VNS iteratively examines a sequence of neighbors of the current best solution. But, it may happen that some instances have several separated and possibly far apart local optimum containing near-optimal solutions. If one considers larger and larger neighborhoods, the information related to the currently best local optimum dissolves. It is therefore of interest to modify VNS schemes in order to explore more fully local optima, which are far away from the current solution. This will be done by accepting to recenter the search when a solution close to the best one known, but not necessarily as good, is found, provided that it is far from this last solution. So, we study on the modified and special ingredient of VNS called skewed variable neighborhood search (SVNS) that is presented in Algorithm 1. The relaxed rule for recentering uses an evaluation function linear in the distance from the current solution; that is, $\sum(X'')$ is replaced by $\sum(X'') - \alpha\rho(X, X'')$ (line 13 of Algorithm 1). where $\rho(X, X'')$ is the distance from X to X'' and α a parameter that is determined experimentally. A metric for distance between solutions is usually

Algorithm 1 Skewed variable neighborhood search for MSCP

Adjacency matrix of graph G , Select the set of neighborhood structures $N_k , k = 1, \ldots, k_{max}$. Find an initial proper coloring X and its value $\sum(X)$. Set $X_{opt} \longleftarrow X$. Choose a stopping condition and a parameter value α. A solution X_{opt} and its value $\sum(X_{opt})$. stopping condition not reached Set: $k \longleftarrow 1 \;\; k \leq k_{max}$ Generate a solution X' at random from the k^{th} neighborhood of X or perturb X, randomly./**Shaking or Perturbing* */ Apply some local search method with X' as initial solution. Denote with X'' the so obtained local optimum. /* *Local search**/ $\sum(X'') < \sum(X_{opt})$ $Xopt \longleftarrow X'' \sum(X_{opt}) \longleftarrow \sum(X'')$ /**Improvement or not**/ $\sum(X'') - \alpha\rho(X, X'') < \sum(X)$ $X \longleftarrow X'' \; k \longleftarrow 1 \; k \leftarrow k + 1$ /**Move or not.**/

easy to find, for example, the hamming distance when solutions are described by boolean vectors or the Euclidean distance in the continuous case. Here we use p-norm distance for p=1,2. P-norm is a class of vector norms, denoted by $||.||_p$, is defined as

$$||X||_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}} \quad p \geq 1, X \in \mathbb{R}^n$$

and the distance between to two n-vectors X and Y based on this norm, is defined as

$$||(X, Y)||_p = (|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p)^{\frac{1}{p}}$$

Algorithm1 presents the general SVNS algorithm for the MSCP, whose ingredients are detailed in the following. It has four main phases, *shaking or perturbing, local search, improve or not* and *move or not.* Usually, the second phase will have maximum computing time. SVNS starts with an initial random coloring X, which must be proper. In the first phase (line 4) we use shaking or perturbing at random. Indeed, we decide shaking to shake against perturbing strategy with probability p in each iteration of local search procedure. k_{max}, the maximum value for the size of neighborhoods, must be identified at first. In local search phase (line 6), we apply variable neighborhood descent (VND) as an ingredient of VNS method. This method is illustrated in Algorithm 2. Then SVNS check to ensure that any improvement has happened or not (lines 8–11)? In lines 13–18 it decides on moving to the new solution or increasing k. SVNS iterates these steps while the maximum number of iterations since the last improvement is reached.

As noted previously, the maximum computing time is spent on the local search phase. To improve the solution quality and acceleration of local search, we will use the holding concept described in Section 2. With determining the suitable *order of falling*, the exploration velocity of neighbors is increased. A

perturbation mechanism will be used to improve solution quality and jumping from local optimums (attractors).

Algorithm 2 Variable neighborhood descent

Select the set of neighborhood structures $N_l\,, l = 1, \ldots, l_{max}$. Find an initial proper coloring X and its value $\sum(X)$. A solution X and its value $\sum(X)$. there is improvement Set $l \longleftarrow 1 \;\; l \leq l_{max}$ Find the best neighbor X' of X $(X' \in N_l(X))$./*Exploration of Neighborhood*/ $\sum(X') < \sum(X) \;\; X \leftarrow X' \; l \longleftarrow 1 \; l \leftarrow l + 1$ /*Move or not*/

4.1 Order of Falling

According to Section 2, the existence of k-holder sets are a *necessary condition* for the existence of k-reducer sets. Therefore, we will use this necessary condition for recognition of reducer sets. Then we increase all color of vertices in reducer set for *falling* the colors of remaining vertices. It is important that the remaining vertices meet *in what order* to take maximum falling of color vertices happens.

Suppose graph G with proper coloring X. Construct weighted directed graph $D = (V(D), E(D), W)$ from G as follows:

1) $V(D) = V(G)$.

2) $E(D) = \{(a, b) |\, a, b \in V(D)\, \&\, a \nearrow b\}$.

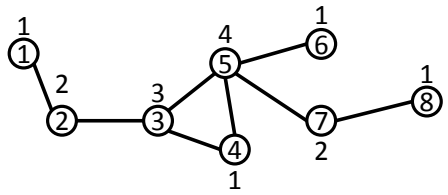
3) $\forall\, e = (a, b) \in E(D)$, $w_e = x_b - x_a$.

D is a weighted directed graph with no directed cycle, multiple edge, and loop. Actually, D is a directed acyclic graph (DAG). The longest directed path in D can help us to find the best order of falling.

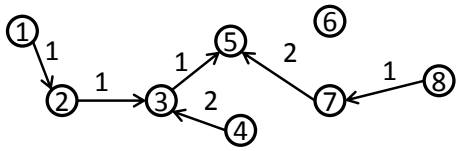
Although the longest path problem is NP-hard for a general graph, it has a linear time solution for directed acyclic graphs. The idea is based on topological sorting. Topological sorting for a DAG is a linear ordering of vertices such that for every directed edge (u, v), vertex u comes before v in the ordering. Topological sorting for a graph is not possible if the graph is not a DAG.

Figure 2 shows an illustrative example of the order of falling. Figure 2a shows G with proper 4-coloring X with sum 15 and corresponding weighted DAG D is depicted in Figure 2b. D has three directed longest paths $p_1 = 1, 2, 3, 5$, $p_2 = 4, 3, 5$ and $p_3 = 8, 7, 5$ of length 3. Therefore they are examples of order of falling. For p_1 it means that if the first vertex of p_1 (vertex 1) is omitted (or increase its color), then the vertex colors of $2, 3, 5$ begin to decrease, sequentially. Finally, the color of vertex 1 re-determined and

another proper 3-coloring $Y = (2, 1, 2, 1, 3, 1, 2, 1)$ with sum 13 is obtained from X. In fact, $Y \in N_1(X)$. Now we can apply the above process for new solution Y.
With this idea, the examination of all elements of N_k to find a local optimum of N_k is not necessary and therefore the exploration velocity of neighbors in local search phase of Algorithm 1 increases.



(a) G with $X = (1, 2, 3, 1, 4, 1, 2, 1)$



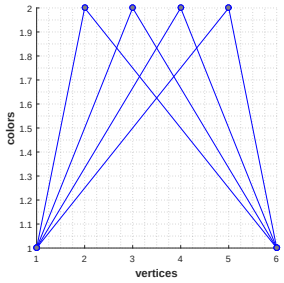
(b) The weighted DAG D

Figure 2: An illustrative example for the order of falling

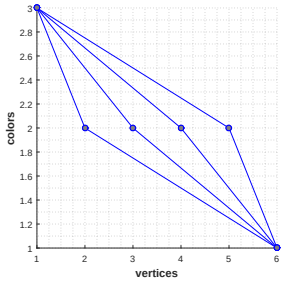
4.2 Perturbation Mechanism

Consider the sub optimal solution X, which has not any suitable falling order. Indeed the weighted DAG D is an empty or low-density graph. Increasing the value of k usually is not useful and lead to increase the computation time. In these cases, local search procedure may be induced to cycle between two or more locally optimal solutions and leading to search stagnation. So, to jump from this situations we change in X to create some good falling orders. We will choose vertex (or vertices) that has more interruptions for holding of the other vertices and increase their colors to produce holder vertices and consequently generate orders of falling to decrease objective function.

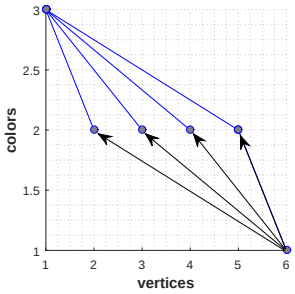
We explain this mechanism on the example shown in Figure 3. We have plotted this example on two-dimensional axis vertex-color. The horizontal and vertical axises are the index and color of vertices respectively. Suppose $k_{max} = 1$, and we stop in proper coloring $X = (1, 2, 2, 2, 2, 1)$ (Fig.3a). There is no holder set of size one and consequently, there is no one-reducer set according to X. Therefore Proposition 1 implies that X is locally optimal (but not globally) with respect to N_1. To jump from this situation we should gen-



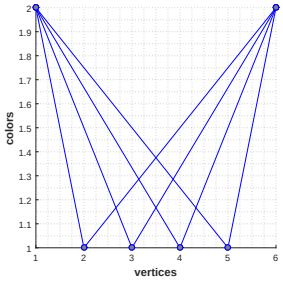
(a) Local optimal that isn't global



(b) Increasing the color of vertex 3



(c) Vertex 6 becomes a holder vertex



(d) Global optimal

Figure 3: An illustrative example for perturbation mechanism

erate holder set by increasing the color of some vertices. But which vertices? Increasing color of one or even all 2,3,4, and 5 causes nothing in order to improve the solution. But increasing color of vertex 1 (or 6) is very useful because it causes to generate holder and reducer set $H = \{6\}$(Figs.3c,3b). If now we apply local search w.r.t N_1 on new solution $Y = (3, 2, 2, 2, 2, 1)$, then we get the new better proper coloring $(2, 1, 1, 1, 1, 2)$, which is globally optimal (Fig.3d).

5 Experimental Results

Our SVNS algorithm is programmed in *MATLAB(R2015a)* and compiled on a PC with 2.7 GHz CPU and 4 Gb RAM. The 23 of well-known benchmark graphs from *COLOR*[2] $2002 - 2004$ in the literature, which is commonly used to test sum coloring algorithms is considered in Table 1 and Table 2. The

[2] http://mat.gsia.cmu.edu/COLOR02/

reported values are based on 10 independent runs. Table 1 gives the detailed characteristics of these benchmark graphs. Columns 1–6 indicate the number of vertices ($|V|$), the number of edges ($|E|$), the density $d = \frac{2mn}{n-1}$, the chromatic sum ($\sum$) and the smallest number of required colors (k) of graphs. Columns 7–9 present detailed computational results of our SVNS algorithm: Best result obtained ($\sum_*$) with the number of required colors (k_*), average coloring sum (*Avg.*) and standard deviation (*Std.*). The results, for instances, of COLOR02 benchmark indicate that SVNS attained the best-known result for 19 instances, and was unable to reach the current best result for four instances *(homer, queen8.8, miles250 and miles500)*. Table 2 reports the

Table 1: Computational results of SVNS on 23 COLOR02 instances

	Characteristics of graphs					SVNS		
Instances	$\|V\|$	$\|E\|$	d	$\sum$	k	$\sum_*(k_*)$	Avg.	Std.
myciel3	11	20	0.36	21	4	21 (4)	21.0	0.0
myciel4	23	71	0.28	45	5	45 (5)	45.0	0.0
myciel5	47	236	0.22	93	6	93 (6)	93.0	0.0
myciel6	95	755	0.17	189	7	189 (7)	189.0	0.0
myciel7	191	2360	0.13	381	8	381 (8)	381.4	0.52
anna	138	986	0.05	276	11	276 (11)	276.3	0.48
david	87	812	0.11	237	11	237 (11)	238.6	0.84
huck	74	602	0.11	243	11	243 (11)	243.0	0.0
jean	80	508	0.08	217	10	217 (10)	217.1	0.31
homer	561	1629	0.01	-	10	1163 (13)	1168.5	3.5
queen5.5	25	160	0.53	75	5	75 (5)	75.0	0.0
queen6.6	36	290	0.46	138	7	138 (8)	138.2	0.42
queen7.7	49	476	0.4	196	7	196 (7)	197.4	4.4
queen8.8	64	728	0.36	291	9	294 (9)	301	5.33
games120	120	638	0.09	443	9	443 (9)	445.3	1.56
miles250	128	387	0.05	325	8	329 (8)	333.2	2.1
miles500	128	1170	0.11	$\leq$709	20	719(20)	728.2	7.03
mug88-1	88	146	0.04	178	4	178 (4)	178.0	0.0
mug88-25	88	146	0.04	178	4	178 (4)	178.0	0.0
mug100-1	100	166	0.03	202	4	202 (4)	202.0	0.0
mug100-25	100	166	0.03	202	4	202 (4)	202.0	0.0
2-insertion-3	37	72	0.11	62	4	62 (4)	62.0	0.0
3-insertion-3	56	110	0.07	92	4	92 (4)	92.0	0.0

comparative results with the following approaches from the literature on the tested COLOR02 instances: a heuristic EXSCOL [33], a greedy algorithm (MRLF) based on the popular RLF graph coloring heuristic [23], MDS5 [15], a parallel genetic algorithm (PGA) [19], a hybrid local search (HLS) [5]. The comparisons are based on the criterion of quality; that is, the smallest sum of colors reached by a given algorithm. Notice that information like computing

Table 2: Comparative results between our SVNS algorithm and five reference approaches on the set of COLOR02 instances

Names	$\sum$	EXCOL [33]	MRLF [23]	MDS5 [15]	PGA [19]	HLS [5]	SVNS
myciel3	21	21	21	21	21	21	**21**
myciel4	45	45	45	45	45	45	**45**
myciel5	93	93	93	93	93	93	**93**
myciel6	189	189	189	189	189	189	**189**
myciel7	381	381	381	381	382	381	**381**
anna	276	283	277	276	281	-	**276**
david	237	237	241	237	243	-	**237**
huck	243	243	244	243	243	243	**243**
jean	217	217	217	217	217	-	**217**
queen5.5	75	75	75	75	75	-	**75**
queen6.6	150	138	138	138	138	138	**138**
queen7.7	196	196	196	196	196	-	**196**
queen8.8	291	291	303	291	302	-	294
games120	443	443	446	443	460	446	**443**
miles250	325	328	334	325	347	343	329
miles500	$\leq$709	709	715	712	762	755	719
mug88-1	178	-	-	178	-	-	**178**
mug88-25	178	-	-	178	-	-	**178**
mug100-1	202	-	-	202	-	-	**202**
mug100-25	202	-	-	202	-	-	**202**
2-insertion-3	62	-	-	62	-	-	**62**
3-insertion-3	92	-	-	92	-	-	**92**

time are not available for all the reference algorithms. Also the symbol "-" means that the information is not available.

6 Conclusion

In this paper, we have presented the skewed variable neighborhood search algorithm towards new kind of neighborhood for solving the minimum sum coloring problem. To acceleration, we introduce a new holding concept in graph coloring problems too. The computational evaluation of the proposed algorithm on 23 of COLOR02 benchmark instances has revealed that SVNS attain the best-known results for 19 instances while failing to reach the best ones for four instances. Also, these results have acceptable competitive compared with five algorithms for MSCP in final.

Acknowledgements

Authors are grateful to there anonymous referees and editor for their constructive comments.

References

1. Benlic, U. and Hao, J. K. *A study of breakout local search for the minimum sum coloring problem*, pp. 128–137, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

2. Bouziri, H. and Jouini, M. *A tabu search approach for the sum coloring problem*, Electronic Notes in Discrete Math. 36 (2010) 915–922.

3. Burke, E. K., McCollum, B., Meisels, A., Petrovic, S. and Qu, R. *A graph-based hyper-heuristic for educational timetabling problems*, Eur. J. Oper. Res. 176 (2007), no. 1, 177 – 192.

4. de Werra, D., Eisenbeis, Ch.,Lelait, S. and Marmol, B.*On a graph- theoretical model for cyclic register allocation*, Discrete Appl. Math. 93 (1999), no. 23, 191 – 203.

5. Douiri S.M. and Elbernoussi, S. *New algorithm for the sum coloring problem*, Int. J. Contemporary Math. Sci. 6 (2011) no. 9-12, 181–192.

6. Douiri S. M. and Elbernoussi, S. *A new ant colony optimization algorithm for the lower bound of sum coloring problem*, J. Math. Model. Algorithms 11 (2012), no. 2, 181–192.

7. Erdos, P., Kubicka, E. and Schwenk, A. J. *Graphs that require many colors to achieve their chromatic sum*, In Proceedings of the Twentieth South-eastern Conference on Combinatorics Graph Theory and Computing (Boca Raton, FL), vol. 71, 1990, pp. 17–28.

8. Gamache, M., Hertz, A. and Ouellet, J. O. *A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding*, Comput. Oper. Res. 34 (2007), no. 8, 2384–2395.

9. Garey, M. R. and Johnson, D. S. *Computers and intractability; a guide to the theory of np-completeness*, W. H. Freeman & Co., New York, NY, USA, 1990.

10. Glass, C. A. *Bag rationalisation for a food manufacturer*, J. Oper. Res. Soc. 53 (2002), no. 5, 544–551.

11. Hajiabolhassan, H.,Mehrabadi, M.L. and Tusserkani, R. *Minimal coloring and strength of graphs*, Discrete Math. 215 (2000), no. 1, 265 – 270.

12. Hajiabolhassan, H.,Mehrabadi, M. L. and Tusserkani, R. *Tabular graphs and chromatic sum*, Discrete Math. 304 (2005), no. 1-3, 11–22.

13. Hansen, P. and Mladenovic, N. *Variable neighborhood search: Principles and applications*, Eur. J. Oper. Res. 130 (2001) no. 3, 449–467.

14. Hao J. -K. and Wu, Q. *Improving the extraction and expansion method for large graph coloring*, Discrete Appl. Math. 160 (2012), no. 1617, 2397 – 2407.

15. Helmar, A. and Chiarandini, M. *A local search heuristic for chromatic sum*, Proceedings of the 9th Metaheuristics International Conference, MIC 2011 (L. D. Gaspero, A. Schaerf, and T. Stutzle, eds.), 2011, pp. 161–170.

16. Jiang, T. and West, D. B. *Coloring of trees with minimum sum of colors*, J. Graph Theory 32 (1999), no. 4, 354–358.

17. Jin, Y.,Hao, J. K. and Hamiez, J. P. *A memetic algorithm for the minimum sum coloring problem*, Comput. Oper. Res. 43 (2014), 318 – 327.

18. Jin, Y. and Hao, J. -K. *Hybrid evolutionary search for the minimum sum coloring problem of graphs*, Inf. Sci. 352 (2016), no. C, 15–34.

19. Kokosiński Z. and Kwarciany, K. *On sum coloring of graphs with parallel genetic algorithms*, pp. 211–219, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

20. Kroon, L. G., Sen, A. , Deng, H. and Roy, A. *The optimal cost chromatic partition problem for trees and interval graphs*, pp. 279–292, Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.

21. Kubicka, E. *The chromatic sum of a graph*, Ph.D. thesis, Western Michigan University, Michigan, 1989.

22. Kubicka, E. and Schwenk, A. J. *An introduction to chromatic sums*, Proceedings of the 17th Conference on ACM Annual Computer Science Conference (New York, NY, USA), CSC '89, ACM, 1989, pp. 39–45.

23. Li, Y. ,Lucet, C., Moukrim, A. andSghiouer, K. *Greedy Algorithms for the Minimum Sum Coloring Problem*, Logistique et transports (Sousse, Tunisia), March 2009, pp. LT–027.

24. Malafiejski, M. *Sum coloring of graphs*, pp. 55–65, AMS, Poland, 2004.

25. Mladenovic, N. and Hansen, P. *Variable neighborhood search*, Comput. Oper. Res. 24 (1997) no.11, 1097–1100.

26. Moukrim, A., Sghiouer, K., Lucet, C. and Li, Y., *Lower bounds for the minimal sum coloring problem*, Electronic Notes in Discrete Math. 36 (2010), 663 – 670.

27. Papadimitriou, C. H. and Steiglitz, K. *Combinatorial optimization: Algorithms and complexity*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

28. Smith, D.H., Hurley, S. and Thiel, S.U. *Improving heuristics for the frequency assignment problem*, Eur. J. Oper. Res. 107 (1998), no. 1, 76 – 86.

29. Stecke, K. E. *Design, planning, scheduling, and control problems of exible manufacturing systems*, Ann. Oper. Res. 3 (1985), no. 1, 1–12.

30. Supowit, K. J. *Finding a maximum planar subset of a set of nets in a channel*, Trans. Comp.-Aided Des. Integ. Cir. Sys. 6 (2006), no. 1, 93–94.

31. Thomassen, C., Erdos, P. , Alavi, Y., Malde, P. J. and Schwenk, A. J. *Tight bounds on the chromatic sum of a connected graph*, J. Graph Theory 13 (1989) no. 3, 353–357.

32. Wang, Y., Hao, J. K.,Glover, F. and Lu, Zh. *Solving the minimum sum coloring problem via binary quadratic programming*, CoRR abs/1304.5876 (2013).

33. Wu Q. and Hao, J. -K. *An effective heuristic algorithm for sum coloring of graphs*, Comput. Oper. Res. 39 (2012), no. 7, 1593 – 1600.

34. Wu Q. and Hao, J. -K. *Improved lower bounds for sum coloring via clique decomposition*, CoRR abs/1303.6761 (2013).

روش جستجوی همسایگی متغیر برای مسئله مینیمم رنگ آمیزی مجموع روی گراف‌های ساده

خلیل عرفانی حیدرنیا، صادق رحیمی شعرباف و جعفر فتحعلی

دانشگاه صنعتی شاهرود، گروه ریاضی کاربردی

دریافت مقاله ۱۱ دی ۱۳۹۵، دریافت مقاله اصلاح شده ۲۷ خرداد ۱۳۹۶، پذیرش مقاله ۹ اسفند ۱۳۹۶

چکیده : مسئله رنگ آمیزی مجموع کمینه (MSCP) عبارتست از یافتن رنگ آمیزی راسی مجاز برای گراف G با استفاده از اعداد طبیعی (۱و۲و...) بطوری که مجموع اعدادی که بعنوان رنگ به رئوس نسبت داده شد اند، کمینه شود. هدف اصلی در این مقاله ارایه یک روش جستجوی همسایگی متغیر مورب برای این مسئله، مبتنی بر ساختار های همسایگی جدید است. برای افزایش سرعت جستجوی همسایگی از مفهوم جدیدی بنام نگهدارندگی راسی و مجموعه‌ای استفاده شده‌است. در نهایت عملکرد قابل قبول روش ارایه شده، روی ۲۳ نمونه مسئله متداول، با چند روش ابتکاری در دهه اخیر مورد مقایسه قرار گرفته است.

کلمات کلیدی : مجموع رنگ آمیزی کمینه؛ جستجوی همسایگی متغیر؛ مجموع رنگی؛ راس نگهدارنده؛ مجموعه نگهدارنده؛ مجموعه کاهنده.