**Int. J. Contemp. Math. Sciences, Vol. 6, 2011, no. 10, 453 - 463**

# New Algorithm for the Sum Coloring Problem

**Sidi Mohamed Douiri**

Research laboratory Mathematics, Computing and Applications
Faculty of Sciences Rabat, Morocco
douirisidimohamed@hotmail.fr

**Souad Elbernoussi**

Research laboratory Mathematics, Computing and Applications
Faculty of Sciences Rabat, Morocco
bernous@fsr.ac.ma

**Abstract**

In this paper we are interested in the elaboration of an approached solution to the sum coloring problem (MSCP), which is an NP-hard problem derived from the graphs coloring (GCP). The problem (MSCP) consists in minimizing the sum of colors in a graph.
Our resolution approach is based on an hybridization of a genetic algorithm and a local heuristic based on an improvement of the maximal independent set algorithm given by F.Glover [4].

**Keywords:** The sum coloring problem, graph coloring problem, maximal independent set, genetic algorithm

# 1 Introduction

The graph coloring problem is a central problem of combinatorial optimization. It has several practical applications, such as: the timetable problems,warehouse management, frequency allocation in mobile network, register allocation in optimizing compilers, scheduling problem, design and operation of flexible manufacturing systems [10]. There exists two categories of methods in the literature to resolve the graph coloring problem: the exact and the heuristic method [2], [6], [1]. In this study we give an improvement of F.Glover algorithm (DBG) which construct an approximation of a maximum independent set and we show the contribution of our algorithm in many examples.

We present afterward the sum coloring problem (MSCP), which is an under graphs coloring one (GCP), (MSCP) consist in finding a graph coloring, by affecting integers in each vertex, so that the sum of colors is minimal. (MSCP)

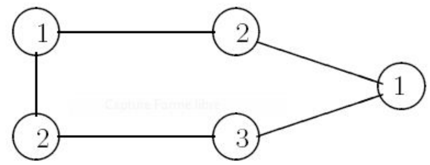


Figure 1: G coloring in 3 colors.

is a problem NP-hard [3]. The theoretical results which exist in the literature for (MSCP) are demonstrated for particular graphs [7], [9], but numerically, the only results are given in [11] and [8]. In 2007, bounds were obtained by a parallel genetic algorithm in [11], and in 2009 upper and lower bounds are determined with greedy algorithm (MRLF) and (MDSAT) [8].

In this paper we propose an hybridization of an adapted genetic algorithm and an improvement of F. Glover algorithm (DBG), and we test it on various instances.

# 2 The graphs coloring problem

The graph coloring problem is to associate a color to each vertex so that two connected vertices do not have the same color. If G contains the edge (a,b), then a and b will have different colors.
A valid k-coloring of vertices in a graph $G = (V, E)$ is an application $c : V \longrightarrow \{1, ..., k\}$ such as $c(x) \neq c(y)$, $\forall (x, y) \in E$, the value $c(x)$ associated with vertex $x$ is called color of $x$. If $(x, y) \in E$ and $c(x) = c(y)$ we say that $x$ and $y$ are in conflict (Figure 1).

**Definition 2.1** - *We call degree of a vertex, the number of relationships associated with this one. We denote by $d_i$ the degree of vertex i.*
*- Two vertices are disjoint if they are not bound by any edge.*
*- The chromatic number of a graph G is the smallest number of colors used for coloring all vertices of G, we denote it $\chi(G)$.*
*- It is well known that the k-coloring problem is NP-completeness and the $\chi(G)$ coloring is NP-hard [5], thus heuristic approaches are inevitable in practice.*

# 3 The sum coloring problem (MSCP)

(MSCP) is an NP-hard problem, it consist in finding a valid coloring so that $\sum_{v \in V} c(v)$ is minimal, this sum is denoted $\sum(G) = min_c \sum_{v \in V} c(v)$.
The smallest number of colors used to color G in the (MSCP) problem is called the strength of G and denoted s(G) (Figure 2).
If we decompose the graph G as independent subsets $X_1, X_2, ..., X_k$ , we obtain

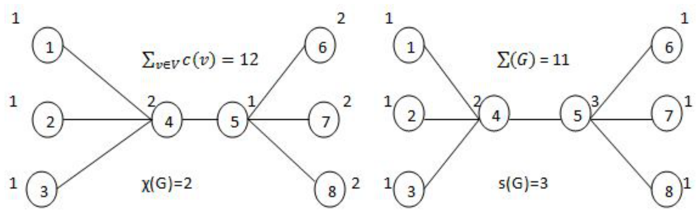


Figure 2: An optimal solution for (MSCP) requires 3 colors.

a valid k-coloring by giving to each subset $X_i$ the color i, $1 \leq i \leq k$ and $\sum(G) = \sum_{i \in (1,\dots,k)} i.card(X_i)$, with $card(X_1) \geq card(X_2) \geq .... \geq card(X_k)$.

# 4 Maximal independent set

An independent set is also called stable, is a set of vertices of which no pair is adjacent. The independent set problem of maximum cardinality and the graph coloring problem are correlated. So, it is normal in coloring problems to look for means to restructure the independent sets, which correspond to various given colors. The purpose is to increase the size of the independent set to reduce the number of sets and subsequently the number of colors. We are inspired in our work by the F. Glover algorithm [4], to build an approximation of a maximal independent set.

## 4.1 Surrogate constraint

The surrogate constraint is to combine all the constraints in one as followed :

$$(IP) \begin{cases} max\ c^t.x \\ A.x \leq b \\ x \in \{0,1\}^n \end{cases}$$

Where $x = (x_1, ...., x_n)^t$, $c = (c_1, ...., c_n)^t$, $b = (b_1, ...., b_m)^t$, $A = a_{i,j}$ for $1 \leq i \leq m$ and $1 \leq j \leq n$.
The surrogate relaxation is given by :

$$(SC) \begin{cases} max\ c^t.x \\ w^t.A.x \leq w^t b \\ x \in \{0,1\}^n \end{cases}$$

With $w = (w_1, ...., w_m)^t \geq 0$

## 4.2 Formulation of the maximum independent set problem

We consider a graph $G = (V, E)$, where $V = \{1, ..., n\}$ denotes the set of vertices in the graph and E the set of edges for each vertex $i \in V$. We have:

$$Nodestar(i) = \{j : \{i, j\} \in E\}$$

$$d_i = card(Nodestar(i))$$

$$d_0 = |E| = \text{number of edges.}$$

The usual formulation in mathematical programming of the maximal independent set associates a binary variable $x_i$ at every vertex $i \in V$, where $x_i = 1$ if and only if the vertex i is chosen as element of the independent set. Thus the problem can be expressed as an integer programming one as follows:

$$(IP)\begin{cases} max\ x_0 = \sum(x_i : i \in V) \\ x_i + x_j \leq 1, \{i, j\} \in E \\ x_i\ binaire, i \in V \end{cases}$$

We use the surrogate constraint heuristics, obtained by replacing all constraints by a single linear combination one to facilitate the initial problem and so to produce an approached solution quicker. For the (IP) problem, the surrogate constraint is determined by a simple sum of all constraints as follows:

$$\sum(d_i x_i : i \in V) \leq d_0$$

The surrogate constraint problem associated to (IP) is :

$$(SC)\begin{cases} max\ x_0 = \sum(x_i : i \in V) \\ \sum(d_i x_i : i \in V) \leq d_0 \\ x_i\ binaire, i \in V \end{cases}$$

## 4.3 F. Glover's algorithm

We consider the following non-oriented graph $G = (V, E)$, where $V = \{1, ..., n\}$ denotes the set of vertices and E denotes the set of edges.

**Algorithm :**

1. The $d_i$ coefficients are in ascending order.
2. Let $i = 1$ and $V' = V$.

3. Identify the first (smallest) $j > i$ such that $\{i, j\} \in E$ and $j \in V'$.

   (a)If no such vertex j exists, proceed directly to step 4.

   (b) If vertex j exists, remove vertex j from $V'$.

4. Let $i' = \min\{q : q > i \; et \; q \in V'\}$. If $i'$ does not exist, or is the last (largest index) vertex in $V'$, stop : $V'$ now has its final form.
   Otherwise, let $i := i'$ and return to step 2.

## 4.4 Improvement by a method based on the multiplier $w$ of the surrogate constraint

For the case of a non-oriented graph $G = (V, E)$, we consider $w = (w_1, ...., w_m)^t$, with $m = d_0$ (the number of edges). We express first the following result:

### Result

Let (SC) the surrogate relaxation of the (IP) problem, we choose a variable $x_r = 1$. If there exists $j \in Nodestar(r)$ and $1 \leq k \leq m$, so that $a_{k,j} = 1$ then $w_k = 0$, for all $1 \leq j \leq n$ and all $1 \leq k \leq m$.

### Proof

For $x_r = 1$ and $j \in$ nodstar(r) we put $\sum_{k=1}^{m} a_{k,j} = d_j$ (the number of the edges connected to vertex j), we have $w^t.A.x = \sum_{l=1}^{m} w_l.a_{l,1}.x_1 + ... + \sum_{l=1}^{m} w_l.a_{l,n}.x_n (*)$. The choice $x_r = 1$ implies $x_j = 0$. Thus all the edges associated to j will be eliminated. In other words for each $k \in \{1, ..., m\}$ if $a_{k,j} = 1$ we can replace it by zero in the (IP) problem, or by using the (*) relation we can put $w_k = 0$.

### Remark

The update of (SC) by using the previous result and by leaving A unchanged is:

$$d_i = \sum_{w_k \neq 0} (a_{k,i})$$

and

$$d_0 = \sum_{w_k \neq 0} w_k$$

From this result, we can give the following algorithm for constructing an approached maximal independent set.

### 4.5 Algorithm (DBG)

1. Let $w = (1, ...., 1)^t$ and $V' = \emptyset$.

2. Calculate the surrogate constraint $wA = \sum_{w_k \neq 0}(a_{k,.})$.

3. Give $i = index(\min(wA))$, let $x_i = 1$ and $V' = V' \cup \{i\}$.

4. For all $j \in$ Nodestar(i) if $a_{k,j} = 1$, then $w_k = 0$, if $\sum_{k=1}^{m} w_k = 0$ stop. Otherwise return to step 2.

### 4.6 The obtained results

We compared the two algorithms that give an approximation of the maximum independent set by implementing them on a PC windows xp pro 2002, 1.6 GH (2CPU) and 1G RAM. We tested different instances arising from the computational Symposium COLOR02 and library DIMACS, for a description of the instances treated [12]. The results obtained by both algorithms are shown in Table 1. For each instance, we indicate the number of vertices n, the number of edges $d_0$, the values $d_{max}$ of the maximum degree, the optimal solutions of the maximal independent set $X_G$ and $X_{DBG}$ given respectively by F.Glover algorithm and by (DBG) algorithm.

| Graph | n | $d_0$ | $d_{max}$ | $X_{DBG}$ | $X_G$ |
|---|---|---|---|---|---|
| 1-FullIns5 | 282 | 3247 | 95 | 138 | 129 |
| 3-FullIns4 | 405 | 3524 | 84 | 193 | 152 |
| 2-Inser4 | 149 | 541 | 37 | 74 | 68 |
| 3-Inser3 | 56 | 110 | 11 | 27 | 23 |
| 4-Inser3 | 79 | 26100 | 13 | 39 | 37 |
| david | 87 | 812 | 82 | 36 | 34 |
| fpsol2.i.1 | 496 | 11654 | 252 | 307 | 307 |
| games120 | 120 | 638 | 13 | 22 | 17 |
| inithx.i.3 | 621 | 13969 | 542 | 360 | 360 |
| mulsol.i.1 | 197 | 3925 | 121 | 100 | 100 |
| mulsol.i.5 | 186 | 3973 | 159 | 88 | 87 |
| mug88-1 | 88 | 146 | 4 | 29 | 26 |
| myciel3 | 11 | 20 | 5 | 5 | 5 |
| myciel5 | 47 | 236 | 23 | 23 | 21 |
| queen13-13 | 169 | 6656 | 48 | 12 | 5 |
| zeroin.i.1 | 211 | 4100 | 111 | 120 | 120 |
| zeroin.i.3 | 206 | 3540 | 140 | 123 | 123 |

Table 1 - Results obtained

# 5 Genetic algorithms

The genetic algorithms (GA) are stochastic optimization algorithms based on the mechanisms of Natural Selection and Genetics. Their operation is extremely simple. We start with an initial population of potential solutions (chromosomes) chosen arbitrarily. We evaluate their relative performance (fitness). Based on this performance, we create a new population of potential solutions using simple evolutionary operators: selection, crossover and mutation. We re-do this cycle until it finds a satisfactory solution.

## 5.1 Description of the approach to the resolution of (MSCP)

Our algorithm is applied to find both the sum of minimum graph coloring $\sum(G)$ and the strength s(G). More exactly, the proposed algorithm begins with the construction of an initial population of individuals by applying the (DBG) algorithm and by fixing a Number k of colors. Then we execute a series of cycles called generations. For each generation we follow the steps as in (Figure 3):

(a)Evaluate each individual by calculating its fitness.

(b) Make a selection by the method of roulette.

(c)Apply a crossover operator applied to two points in both parents

(d) Apply the mutation with a very low probability.

(e) Evaluate the individuals.

(f) Repeat(b) - (e).

## 5.2 Remark

If we find a valid k-coloring, we decrement the number of colors $k \leftarrow k-1$ and we look for a new color. We repeat this procedure until we have a good k-coloring (without conflicts).

### *The objective function*

We note $M(i,j)$ the matrix of conflict and $c(i)$ the color of the vertex i.

$$M(i,j) = \begin{cases} 1 \;\; if \;\; c(i) = c(j) \;\; and \;\; \{i,j\} \in E \\ 0 \;\; otherwise \end{cases}$$

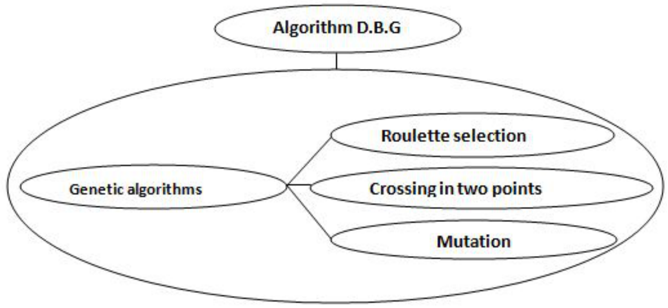


Figure 3: Algorithm for solving MSCP.

The individual $p$ =(c(1),c(2),.....,c(N)) corresponds to an assignment of k colors to all the vertices of the graph G. Then for each individual $p$, the fitness function $f(p)$ corresponds to:

$$f = \sum_{(i,j)\in E} M(i,j).$$

The goal is to reduce f to reach $f = 0$.

## The population initialization

The initialization of our population is made by applying the algorithm (DBG) on the graph G to obtain an estimate of the maximal independent set that we note $X_1$. And we associate it to the value 1, then $c(X_1) = 1$. The size of every individual is equal to the number of G vertices. We place the integer 1 in the vertices corresponding to $X_1$ and a random integer lower or equal to k and different from 1 in the remaining vertices.

## Roulette wheel selection

Every individual will be duplicated in a new population proportionally to its value of adaptation, we make as many editions with discount as there are of elements in the population. Probability with which it will be reintroduced in the new population is:

$$\frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)}$$

## Crossing

We apply a two-point crossover with a probability $p_c = 0.8$, without changing the elements of our maximal independent set generated at first by the algorithm

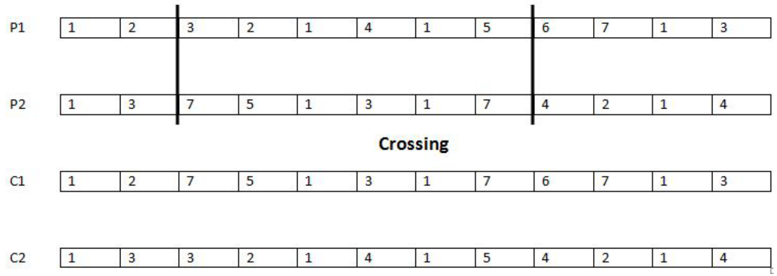


Figure 4: Crossing two points.

(DBG). We pull randomly two positions inter-genes a and b in each parent $P_1$ and $P_2$ with $1 \leq a \leq b \leq card(V)$ where V is the set of vertices of graph G. The crossing is made in the following way (Figure 4):

$$C_1(i) = \begin{cases} P_1(i) & if \quad i \in [1, a[ \\ P_2(i) & if \quad i \in [a, b[ \\ P_1(i) & if \quad i \in [b, card(V)] \end{cases}$$

$$C_2(i) = \begin{cases} P_2(i) & if \quad i \in [1, a[ \\ P_1(i) & if \quad i \in [a, b[ \\ P_2(i) & if \quad i \in [b, card(V)] \end{cases}$$

### *Mutation*

The mutation is made with a probability $p_m = 0.2$, and allows to achieve the property of ergodicity. We choose a gene in a random way different from the set $X_1$ and we change its value by an integer less than or equal to the current value k and different from 1.

## 5.3 The obtained results

Our algorithm for the sum coloring of graphs has been implemented on a PC windows xp pro 2002, 1.6 GH (2CPU) and 1G RAM. We tested different instances arising from the computational Symposium COLOR02 and library DIMACS (the description of the instances treated exist in [12]). The numerical results obtained by our algorithm are shown in Table 2. For each instance, we indicate the number of nodes n, the number of edges v, the chromatic number $\chi(G)$, the lower and upper bounds $LB_{th}$ and $UB_{kok}$ for the minimum sum coloring, the strength $s(G)_{kok}$ given in [11], $\sum(G)$ the approximate solution found by our algorithm (MSCP) and s(G) the strength of G. Most of these instances have been easily resolved.

| Graph | n | v | $\chi(G)$ | $LB_{th}$ | $UB_{kok}$ | $s(G)_{kok}$ | $\sum(G)$ | s(G) |
|---|---|---|---|---|---|---|---|---|
| huck | 74 | 602 | 11 | 129 | 243 | 11 | 243 | 11 |
| queen6.6 | 36 | 580 | 7 | 57 | 138 | 8 | 138 | 8 |
| miles250 | 128 | 387 | 8 | 156 | 347 | 8 | 343 | 10 |
| miles500 | 128 | 1170 | 20 | 318 | 762 | 20 | 755 | 22 |
| games120 | 120 | 1276 | 9 | 156 | 460 | 9 | 446 | 9 |
| myciel3 | 11 | 20 | 4 | 17 | 21 | 4 | 21 | 4 |
| myciel4 | 23 | 71 | 5 | 33 | 45 | 5 | 45 | 5 |
| myciel5 | 47 | 236 | 6 | 62 | 93 | 6 | 93 | 6 |
| myciel6 | 95 | 755 | 7 | 116 | 189 | 7 | 189 | 7 |
| myciel7 | 191 | 2360 | 8 | 219 | 382 | 8 | 381 | 8 |

Table 2 - The obtained results and the comparison with the literature[KOK 07].

We added to table 3 other tests on graphs given by the same library.

| Graph | n | v | $\chi(G)$ | $\sum(G)$ | s(G) |
|---|---|---|---|---|---|
| fpsol2.i.1 | 496 | 11654 | 65 | 3405 | 65 |
| inithx.i.1 | 864 | 18707 | 54 | 3679 | 54 |
| mug88-1 | 88 | 146 | 4 | 190 | 4 |
| mug88-25 | 88 | 146 | 4 | 187 | 4 |
| mug100-1 | 100 | 166 | 4 | 211 | 4 |
| mug88-25 | 100 | 166 | 4 | 214 | 4 |
| 2-Inser 3 | 37 | 72 | 4 | 62 | 4 |
| 3-Inser 3 | 56 | 110 | 4 | 92 | 4 |
| zeroin.i.2 | 211 | 3541 | 30 | 1013 | 30 |
| zeroin.i.3 | 206 | 3540 | 30 | 1007 | 30 |

Table 3

# 6 Conclusions and perspectives

In this paper we have presented the problem of the sum coloring in graphs, we solved it by combining a genetic algorithm with a surrogate constraint heuristic (DBG), and we compared our numerical results with existing results. The effectiveness of this approach at the instance of the literature is satisfactory. We intend afterward to combine our algorithm with other heuristics to improve its performances.

# References


[1] C. Fleurent, J. Ferland, Genetic and Hybrid Algorithms for Graph Coloring. Annals of Operations Research, vol. 63, pp. 437-464, 1996.

[2] C. Lucet, F. Mendes, A. Moukrim , "An Exact Method for Graph Coloring", Computers and Operations Research, 2006, volume 33, number 8, pp. 2189-2207.

[3] E. Kubicka and A. J. Schwenk, An introduction to chromatic sums, Proceedings of the ACM Computer Science Conference, 39-45, 1989.

[4] F.Glover, Tutorial on surrogate constraint approaches for optimization in graphs. Journal of Heuristics, 175-228, 2003.

[5] Garey, Michael.R, Johnson.D.S. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York, 1979.

[6] I. Blochliger, N. Zufferey. A graph coloring heuristic using partial solutions and a reactive tabu scheme. Computers and Operations Research, vol. 35, no. 3, pp. 960-975, 2008.

[7] Leo G. Kroon, Arunabha Sen, Haiyong Deng, and Asim Roy. The optimal cost chromatic partition problem for trees and interval graphs. Graph-Theoretical Concepts in Computer Science, LNCS, pp. 279-292, 1996.

[8] Li, Y., C. Lucet, A. Moukrim and K. Sghiouer, Greedy Algorithms for the Minimum Sum Coloring Problem, International Workshop: Logistics and transport, 2009.

[9] Salavatipour, M. R., On sum coloring of graphs, Discrete Appl. Math, 127(3), pp 477-488, 2003.

[10] Stecke, K. Design planning, scheduling and control problems of flexible manufacturing, Annals of Operations Research 3: 3-12, 1985.

[11] Zbigniew Kokosiski and Krzysztof Kwarciany. On sum coloring of graphs with parallel genetic algorithms. In ICANNGA '07, part I, LNCS 4431, pp. 211-219, 2007.

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


**Received: September, 2010**