# Lower Bound for (Sum) Coloring Problem

Alexandre Gondran, Vincent Duchamp, Laurent Moalic

# Lower Bound for (Sum) Coloring Problem

Alexandre Gondran
ENAC, French Civil Aviation University, Toulouse, France
alexandre.gondran@enac.fr

Vincent Duchamp
ENAC, French Civil Aviation University, Toulouse, France
vincent.duchamp@alumni.enac.fr

Laurent Moalic
UHA, University of Upper Alsace, Mulhouse, France
laurent.moalic@uha.fr

The Minimum Sum Coloring Problem is a variant of the Graph Vertex Coloring Problem, for which each color has a weight. This paper presents a new way to find a lower bound of this problem, based on a relaxation into an integer partition problem with additional constraints. We improve the lower bound for 18 graphs of standard benchmark DIMACS, and prove the optimal value for 4 graphs by reaching their known upper bound.

## 1 Introduction

The Minimum Sum Coloring Problem $MSCP$ is a variant of the Graph Vertex Coloring Problem (GVCP), with weights associated to colors. This problem can be applied to various domains such as scheduling, resource allocation or VLSI design [1, 2].

Given an undirected graph $G = (V, E)$ with $V$ a set of $n$ vertices and $E \subset V^2$ a set of edges, graph vertex coloring involves assigning each vertex with a color so that two adjacent vertices (linked by an edge) feature different colors. An equivalent formulation is to consider a coloring as a partition of $G$ into subsets of vertices so that two adjacent vertices not belong to the same subset[3].

The GVCP consists in finding the minimum number of colors (or equivalently the minimum number of subsets), called *chromatic number* $\chi(G)$, required to color (or equivalently to partition) the graph $G$.

The MSCP is a variant of GVCP, in which each color has a cost equals to the integer that represents the color. The objective of MSCP is to minimize the sum of the cost of the coloring, called *chromatic sum* of $G$ and denoted $\Sigma(G)$. Figure 1 gives an example of MSCP from Jin and Hao [4] on a graph with $n = 9$ vertices and shows the difference between the two problems.

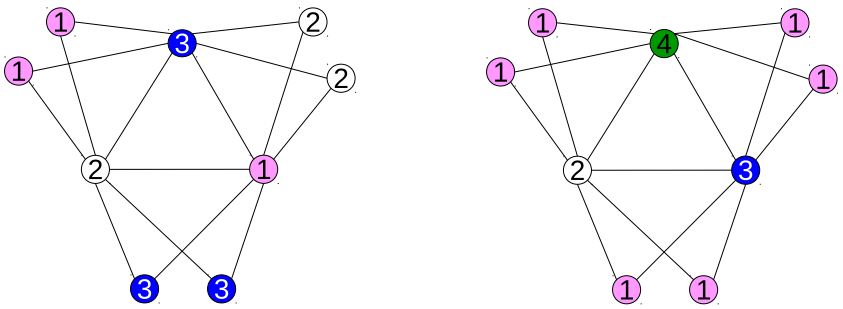


Figure 1: Example of graph for MSCP from Jin and Hao [4]. The left coloring uses 3 colors (integers 1, 2 and 3). It is an optimal solution of GVCP and the chromatic number $\chi(G) = 3$. Moreover, its sum coloring cost is equal to 18. The right coloring uses one more color (integers 1, 2, 3 and 4) but its sum coloring cost is equal to 15, which is the chromatic sum of $G$: $\Sigma(G) = 15$.

More precisely, one possible formulation of the MSCP is the following :

$$(MSCP)\begin{cases} \text{Min.} & f_\Sigma(\mathbf{V}) = \sum_{l=1}^{n} l|V_l| & & (1)\\ \text{s.c.} & \cup_{l=1..n} V_l = V & & (2)\\ & V_l \cap V_k = \emptyset, & \forall (l,k), l \neq k & (3)\\ & |V_l| \geq |V_{l+1}|, & \forall l = 1..n-1 & (4)\\ & \{i,j\} \nsubseteq V_l, & \forall (i,j) \in E,\ \forall l = 1..n & (5)\\ & V_l \subset V & \forall l = 1..n & (6) \end{cases}$$

We denoted $\mathbf{V} = (V_1, V_2..., V_n)$ the partition of $G$ (constraints (2) and (3)) into $n$ sets with $n$, the number of vertices of $G$. For all $1 \leq i \leq n$, each $V_i$ is called color class. To be as general as possible, we do not precise the number $k$ of colors used, we only know that $1 \leq k \leq n$. Then the number of colors used in a coloring is equal to the number of none empty color classes : $k = |\{V_i \in \mathbf{V} \mid |V_i| \geq 1\}|$. Constraint (5) indicates that two adjacent vertices cannot be in the same color class. That is the coloring constraint. Constraint (4) forces the color classes to be ordered from the largest to the smallest size. With this convention, the objective function can be expressed as equation (1). An optimal sum coloring of graph $G$, noted $\mathbf{V}^*$, has its objective function equals to the chromatic sum: $f_\Sigma(\mathbf{V}^*) = \Sigma(G)$.

$MSCP$ is an NP-hard problem[5] and exact methods to solve it are effective only on small instances or specific graphs. For general graphs, we use heuristics to get a sub-optimal solution[6]. It provides an upper bound of the optimal solution. We can compare it to a lower bound to estimate the quality of the solution.

This paper presents a new way to find lower bounds for this problem. We obtain it by relaxing MSCP into an Integer Partition Problem (IPP). A similar approach has recently be use by Lecat, Lucet and Li [7]. They find a lower bound, called $LBM\Sigma$, using the notion of *motif* that improve largely the best lower bound of the literature (DIMACS benchmarks [8]). This paper improves this lower bound by counting the maximal number of independent sets of maximal size in a graph. An Independent Set (IS) or stable set is a set of vertices of $G$, no two of which are adjacent. A color class is by definition an IS. Others approaches designed to find the lower bound for $MSCP$ are proposed in [3, 4, 9].

Experiments on standard benchmarks DIMACS [8] of graph instances show that we improve the lower bound for several graphs and we sometimes prove their optimal value by reaching their known upper bound.

In the following sections, we first present how we relax $MSCP$ to an $IPP$ (Section 2). In Section 3, we detail a new way of ordering integer partitions, and we use it to solve exactly our relaxed problem. Section 4 shows how to extend this lower bound to GVCP. We show and analyze our results in Section 5, and then we conclude.

## 2 Relaxation as an integer partition problem

### 2.1 Integer partitions

We relax the problem of sum coloring $MSCP$ into a problem of integer partition, denoted $IPP_{\Sigma M}$, with the same objective function but with less constraints.

An integer partition is a way of writing a positive integer $n$ as a sum of other positive integers. We say that the vector $\mathbf{a} = (a_i)_{1 \leq i \leq n} \in \mathbb{N}^n$ is a partition of $n$ if :

$$(IP)\begin{cases} \sum_{i=1}^{n} a_i = n \\ a_i \geq a_{i+1},\ \forall i = 1..n-1 \end{cases}$$

We arbitrarily choose to rank $a_i$ in decreasing order. Partitions can be graphically visualized with Young diagrams [10] or Ferrer diagrams. Figure 2 represents the partition of the integer $12 = 4+4+3+1+0+0+0+0+0+0+0+0$. The vector of this partition is: $\mathbf{a} = (4,4,3,1,0,0,0,0,0,0,0,0)$ or simply noted $\mathbf{a} = (4,4,3,1)$ because the two first lines have four squares, the third line, three squares and the last one, one square.

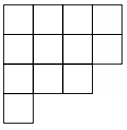

Figure 2: Example of Young diagram for the partition of integer 12 into a sum of four integers : 4, 4, 3 and 1 ($12 = 4+4+3+1$). Each line corresponds to one positive integer of the partition. Integers are ranked in descending order.

## 2.2 Definition of $IPP_{\Sigma M}$

Instead of considering a set partition (or equivalently vertices partition), we focus on the cardinality of each set and we study an integer partition of the number $n = |V|$, corresponding to the number of vertices. We note for each vertices set $V_i$, its cardinality $a_i = |V_i|$. The objective function (1) of $MSCP$ becomes $\sum_{i=1}^{n} ia_i$.

By this way, the two first constraints of $MSCP$ (constraints (2) and (3)) imply $\sum_{i=1}^{n} a_i = n$. Constraint (4) of $MSCP$ is still valid with the new notations: $a_i \geq a_{i+1}$. We represent a coloring with Young diagram where each square represents a vertex. The number of color used is the number of lines. Squares in the same line are in the same color class. Figure 3 represents Young diagrams of the two colorings presented in Figure 1.

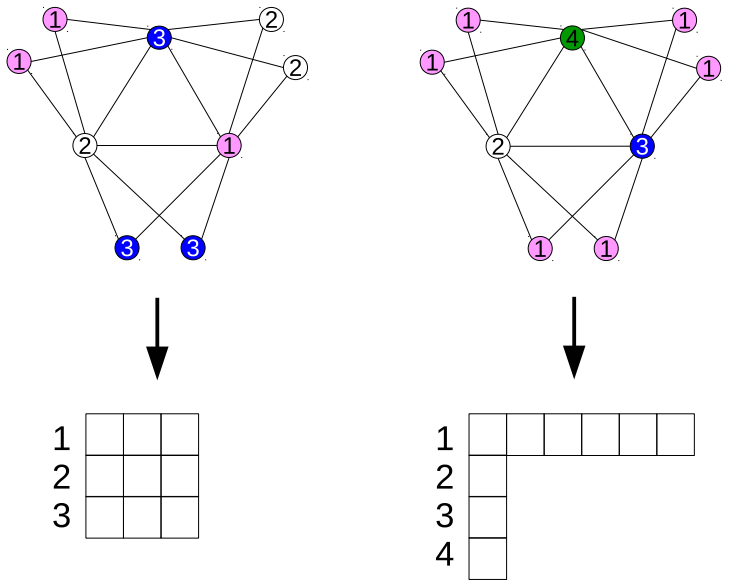


Figure 3: Young diagrams of two colorings of the same graph as in Figure 1. Each square represents a vertex. Squares on the same line share the same color. The objective function value of $MSCP$ is calculated by counting the number of squares for each line: $\sum_{i=1}^{n} ia_i$.

The *stability number* of $G$, noted $\alpha(G)$, is the size of the maximum IS. It indicates the maximum number of columns of a Young diagram that it is possible to use for a coloring, therefore $a_i \leq \alpha(G)$. Moreover, we will define $m$ as an upper bound of the maximum number of color classes of size $\alpha(G)$ which can be in a coloring. Then, we can add an extra constraint : $|\{a_i | a_i = \alpha(G)\}| \leq m$ or equivalently if $m < n$ to $a_{m+1} \leq \alpha(G) - 1$. Finding $\alpha(G)$ is a NP-hard problem [11] and counting all maximal IS is #-P complete [12]. Therefore, there are no polynomial-time algorithm able to solve these problems, unless $P = NP$. However, the graph size for which it is possible to solve them exactly in a reasonable time (less a few minutes) is around 1000 vertices for random graphs with 0.5 density by running a solver such as MoMC [13], the state-of-art exact maximum clique algorithm. We define $\overline{\alpha}$ an upper bound of $\alpha(G)$, for which the above constraints are still true: $a_i \leq \overline{\alpha}$ and $|\{a_i | a_i = \overline{\alpha}\}| \leq m$. If MoMC takes too much time to compute $\alpha(G)$, then a possible upper bound of the largest IS size, $\overline{\alpha}$, is the highest positive integer, $k$, verifying: $|\{v \in V, d_{\overline{G}}[v] \geq k\}| \geq k$ with $d_{\overline{G}}[v]$ the degree of the vertex $v$ in $\overline{G}$, the complement graph of $G$[1]. Another upper bound is the number of colors used for a vertex coloring of $\overline{G}$ not necessarily optimal.

The *chromatic strength* of a graph $G$, noted $s(G)$, is the minimum number of colors used in all the optimal colorings of $MSCP$. It indicates the minimum number of lines of a Young diagram that are needed for a coloring. We note $\underline{s}$ a lower bound of $s(G)$ and we notice that :

$$s(G) \geq \chi(G) \geq \left\lceil \frac{n}{\alpha(G)} \right\rceil \tag{7}$$

Indeed the *chromatic number* $\chi(G)$ is at least a lower bound of $s(G)$ because an optimal coloring of $MSCP$ is at least a legal coloring for GVCP. Moreover, a simple lower bound of $\chi(G)$ is $\left\lceil \frac{n}{\alpha(G)} \right\rceil$ or at least $\left\lceil \frac{n}{\overline{\alpha}} \right\rceil$. The number of coefficients of $\mathbf{a}$ not null (or equivalently the number of lines of Young diagram) is at least higher than $s(G)$ and therefore higher than $\underline{s}$, i.e. $|\{a_i | a_i \geq 1\}| \geq \underline{s}$ or equivalently $a_{\underline{s}} \geq 1$ because $(a_i)$ is a decreasing suite of integer.

[1] Notice that the maximum IS of a graph $G$ is the maximum clique of $\overline{G}$.

Given a graph $G = (V, E)$, knowing $n = |V|$ its size, $\overline{\alpha}$ an upper bound of its stability number, $\underline{s}$ a lower bound of its chromatic strength and $m$, an upper bound of the maximum number of color classes of size $\overline{\alpha}$, we define $IPP_{\Sigma M}$ problem as:

$$(IPP_{\Sigma M}) \begin{cases} \text{Min.} & f_{\Sigma M}(\mathbf{a}) = \sum_{i=1}^{n} i a_i & & (8) \\ \text{s.c.} & \sum_{i=1}^{n} a_i = n & & (9) \\ & a_i \geq a_{i+1}, & \forall i = 1..n-1 & (10) \\ & a_i \leq \overline{\alpha}, & \forall i = 1..n & (11) \\ & |\{a_i | a_i = \overline{\alpha}\}| \leq m & & (12) \\ & a_{\underline{s}} \geq 1 & & (13) \\ & \mathbf{a} = (a_i)_{1 \leq i \leq n} \in \mathbb{N}^n & & (14) \end{cases}$$

$IPP_{\Sigma M}$ depends only of four integers: $n$, $\overline{\alpha}$, $\underline{s}$ and $m$. We denote $\Sigma M$ the optimal objective function value of one of its optimal partition, $\mathbf{a}^*$: $f_{\Sigma M}(\mathbf{a}^*) = \Sigma M$. Therefore, $\Sigma M$ is function of these four parameters: $\Sigma M(n, \overline{\alpha}, \underline{s}, m)$.

**Theorem 1.** *$IPP_{\Sigma M}$ is a relaxation of $MSCP$, that is, $\Sigma M(n, \overline{\alpha}, \underline{s}, m)$ is a lower bound of $\Sigma(G)$:*

$$\Sigma M \leq \Sigma(G) \tag{15}$$

**Proof 1.** *The fourth constraint of $MSCP$ (coloring constraint (5)) imply constraints (11-13), then $IPP_{\Sigma M}$ is a relaxation of $MSCP$.*

### 2.3 Analysis

Note that if we remove the constraint (13) from $IPP_{\Sigma M}$ (or equivalently if we fix $m$ at $+\infty$), we would get a problem equivalent to the one used by Lecat et al. [7] to find the $LBM\Sigma$ lower bound: $LBM\Sigma = \Sigma M(n, \overline{\alpha}, \underline{s}, m = +\infty)$ with $\overline{\alpha} = \alpha(G)$, if it is known and :

$$\underline{s} = \begin{cases} \chi(G) & \text{if it is known} \\ \left\lceil \frac{n}{\alpha(G)} \right\rceil & \text{otherwise} \end{cases} \tag{16}$$

Therefore, $\Sigma M$ is an improvement of $LBM\Sigma$, obtained by using the integer $m$ corresponding to the maximum number of IS of size $\alpha(G)$ that is possible to use in a coloring of $G$ graph.

$$LBM\Sigma \leq \Sigma M \leq \Sigma(G) \tag{17}$$

For this reason, we need to find the smallest possible value of $m$ to have the highest possible value of $\Sigma M$. To obtain our value of $m$, we define a new graph called *the maximum independent set graph*, noted $\tilde{G} = (\tilde{V}, \tilde{E})$ as follow.

**Definition 1.** *Let a graph $G = (V, E)$, we call the **maximum independent set graph** of $G$, the graph $\tilde{G} = (\tilde{V}, \tilde{E})$ build as follow :*

- *each vertex of $\tilde{V}$ is a maximum independent set[2] of $G$;*
- *it exists an edge $e = (u, v) \in \tilde{E}$ between two independent sets $u \in \tilde{V}$ and $v \in \tilde{V}$, if and only if $u$ and $v$ have at least one vertex $w \in V$ of $G$ in common (i.e. $u \cap v \neq \emptyset$); we said that $u$ and $v$ are* incompatible *because both can not be part of the same coloring of $G$.*

**Theorem 2.** *An optimal sum coloring $\mathbf{V}^*$, of a graph $G$ (i.e. an optimal solution of $MSCP$) can not have more than $\alpha(\tilde{G})$ color classes of size $\alpha(G)$ :*

$$|\{V_i \in \mathbf{V}^*, |V_i| = \alpha(G)\}| \leq \alpha(\tilde{G})$$

**Proof 2.** *Finding the maximum IS of $\tilde{G}$ corresponds to find the maximal number of maximal ISs compatibles in $G$. In others words, $\alpha(\tilde{G})$ is the maximum number of IS of size $\alpha(G)$ that are possible to include in a same coloring of $G$. This is true not only for sum coloring problem but also for all coloring problems.*

Therefore, we use in experimental tests, when it is possible to compute it:

$$m = \alpha(\tilde{G}) \tag{18}$$

[2] independent set of size $\alpha(G)$

## 3 Solving $IPP$

### 3.1 Ordering integer partitions

We defined a relaxation problem, $IPP_{\Sigma M}$, of $MSCP$. Our aim is to solve exactly this problem in order to provide a lower bound to $MSCP$. We define an order between integer partitions corresponding to the objective function $f(\mathbf{a}) = \sum_{i=1}^{n} ia_i$. Given a partition $\mathbf{a}$, we define a set of successor partitions if we move down only one square in the corresponding Young diagram of $\mathbf{a}$ by the following rule :

**Data:** $n$
**Fonction** *successor*(**a**)**:**
  $succ \leftarrow \varnothing$
  **foreach** $i = 1..n-1$ **do**
    **if** $a_i \neq 1 \ \wedge \ a_{i+1} < a_i$ **then**
      $j \leftarrow i+1$
      **while** $a_j \geq a_i - 1$ **do**
        $j \leftarrow j+1$
      $\mathbf{b} \leftarrow change(\mathbf{a}, i, j)$ *// we note :* $\mathbf{b} \leftarrow \mathbf{a} \oplus (i,j)$
      $succ \leftarrow succ \cup \{\mathbf{b}\}$
  **return** *succ*

**Algorithm 1:** Function that return all the successors of the partition $\mathbf{a}$.

The *change* function, noticed $\oplus$ operator, is defined as follows :

**Data:** $n$
**Fonction** *change*(**a**,*i*,*j*)**:**
  **foreach** $k = 1..n$ **do**
    $b_k \leftarrow a_k$
  $b_i \leftarrow a_i - 1$
  $b_j \leftarrow a_j + 1$
  **return b**

**Algorithm 2:** Function that return a neighbor partition of $\mathbf{a}$, just two values of $\mathbf{b}$ differ from $\mathbf{a}$; we note $\mathbf{b} \leftarrow \mathbf{a} \oplus (i,j)$

Figure 4 illustrates the successor operator with Young diagram. Algorithm 1 shows that it is possible to move down the last square of each line if :

- the following line has not the same number of squares; it is why the red square of first line of Figure 4 can not move.
- the line has not an unique square; it is why the orange square of last line of Figure 4 can not move.

When it is possible to move down a square (case of blue and green squares of Figure 4), the square takes last place of the first possible line. Blue square of Figure 4 can take the place in the just following line. But green square of Figure 4 must go two lines down.

**Remark 1.** *By construction, if* $\mathbf{a}$ *is an admissible solution of* $IPP_{\Sigma M}$*, then all of* $\mathbf{a}$*'s succesor interger partitions are also an admissible solution of* $IPP_{\Sigma M}$*. In other words, by noting* $\Omega(IPP_{\Sigma M})$ *the set of all admissible solutions of* $IPP_{\Sigma M}$*: if* $\mathbf{a} \in \Omega(IPP_{\Sigma M})$*, then succesor*$(\mathbf{a}) \subset \Omega(IPP_{\Sigma M})$.

**Remark 2.** *If* $\mathbf{a}$ *is an admissible solution of* $IPP_{\Sigma M}$ *without succesor i.e. successor*$(\mathbf{a}) = \varnothing$*, then it means that* $\mathbf{a}$ *is the column partition* $\mathbf{a} = (\underbrace{1, ..., 1}_{n})$.

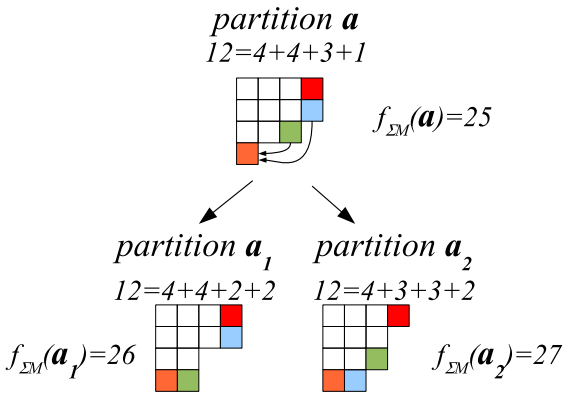


Figure 4: Illustration for the way we find the two successors of a partition $\mathbf{a}$: $\mathbf{a_1}$ and $\mathbf{a_2}$. From $\mathbf{a}$, we move a square from a line $i(=2$ or $3)$ to the closest line $j$ where $a_i > a_j$; $j = 4$.

We define symmetrically the *predecessor* function :

**Data:** $n$, $\overline{\alpha}$, $\underline{s}$, $m$
**Fonction** *predecessor*(**a**)**:**
  $pred \leftarrow \varnothing$
  **foreach** $i = 2..n-1$ **do**
    **if** $[(i \leq \underline{s} \ \wedge \ a_i > 1) \ \vee \ (i > \underline{s} \ \wedge \ a_i > 0)]$
      $\wedge \ [(i \leq m \ \wedge \ a_{i-1} < \overline{\alpha}) \ \vee \ (i > m \ \wedge \ a_{i-1} < \overline{\alpha} - 1)]$
      $\wedge \ a_{i+1} < a_i$ **then**
      $j \leftarrow i - 1$
      **while** $j \neq 1 \ \wedge \ a_j \geq a_{j-1}$ **do**
        $j \leftarrow j - 1$
      $\mathbf{b} \leftarrow change(\mathbf{a}, i, j)$
      $pred \leftarrow pred \cup \{\mathbf{b}\}$
  **return** *pred*

**Algorithm 3:** Function that returns all the predecessors of the partition $\mathbf{a}$.

Figure 5 illustrates the predecessor operator with Young diagram. Algorithm 3 shows that it is possible to move up the last square of each line $i > 1$ if :

- the line just below $(i + 1)$ has not the same number of squares; it is why the red squares of the two first lines of partition $\mathbf{a}$ of Figure 5-up-right can not move.
- the line just above $(i - 1)$ has strictly less than $\overline{\alpha}$ squares if $i - 1 \leq m$; it is why the red square of the third line of partition $\mathbf{a}$ of Figure 5-up-right can not move.
- the line just above $(i-1)$ has strictly less than $\overline{\alpha}-1$ squares if $i-1 > m$; this constraint and the previous one define the hashed area (forbidden area) of Figure 5-up-right.
- the square of line $\underline{s}$ is alone on its line; it is why the red square of last line of partition $\mathbf{a}$ of Figure 5-up-right can not move. This mandatory square is noted in bold on Figure 5-up-right.

When it is possible to move a square up (case of blue and green squares of partition $\mathbf{b}$ of Figure 5-center), the square takes last place of the first possible line (line 3 and 5 respectively).

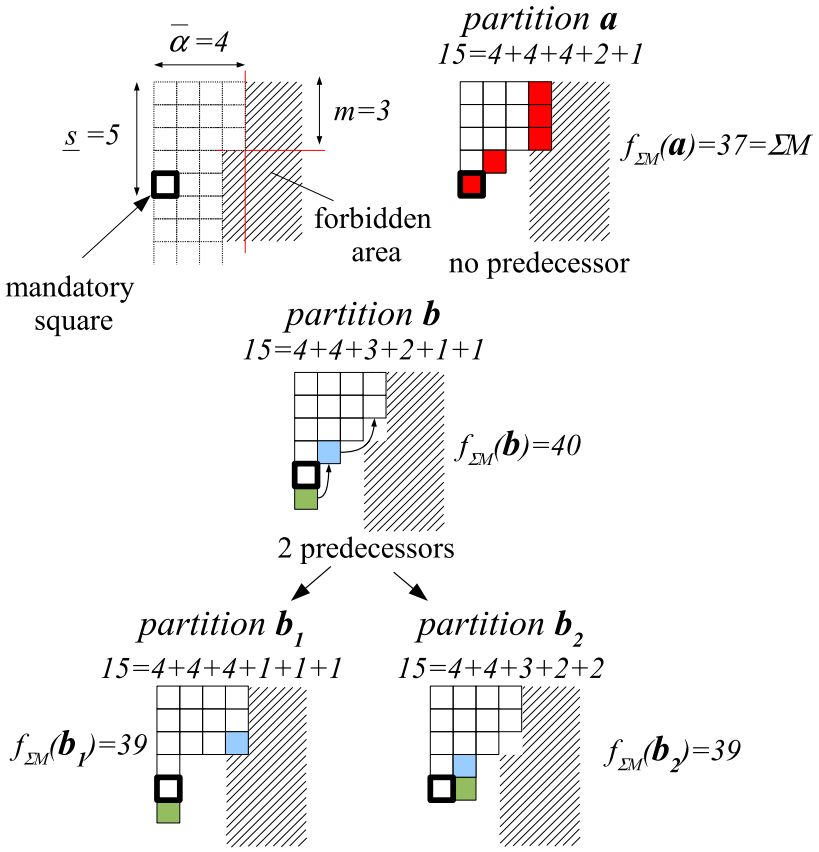


Figure 5: Illustration of the way we find a predecessor of a partition. The partition $\mathbf{a}$ (up-right figure) has no predecessor because the hashed area is the forbidden area and the bold square (last line) is mandatory (a square must be place on it). The partition $\mathbf{b}$ has two possible predecessors, $\mathbf{b_1}$ and $\mathbf{b_2}$, by moving blue square or respectively green square to line 3 or respectively line 5.

**Remark 3.** *By construction, if* $\mathbf{b} \in \Omega(IPP_{\Sigma M})$*, then predecessor*$(\mathbf{b}) \subset \Omega(IPP_{\Sigma M})$.

**Theorem 3.** *If a partition* $\mathbf{b}$ *is a successor of the partition* $\mathbf{a}$ *(respectively* $\mathbf{b}$ *is a predecessor of* $\mathbf{a}$*), so that* $\mathbf{b} \leftarrow \mathbf{a} \oplus (i,j)$*, therefore* $j > i$ *(resp.* $i < j$*) and*

$$f(\mathbf{b}) = f(\mathbf{a}) + j - i > f(\mathbf{a}) \text{ (resp. } < f(\mathbf{a})) \tag{19}$$

*Proof.*

$$\begin{aligned} f(\mathbf{b}) &= \sum_{k=1}^{n} k b_k = \sum_{k\neq i;\, k\neq j} k b_k + i b_i + j b_j \\ &= \sum_{k\neq i;\, k\neq j} k a_k + i(a_i - 1) + j(a_j + 1) = \sum_{k=1}^{n} k a_k - i + j \\ &= f(\mathbf{a}) - i + j \end{aligned}$$

□

If we list the partitions of an integer, we can compare their costs (of sum coloring) using this theorem. We get a comparison relation between partitions which is similar to the comparison relation between motifs used in [7] (definition 4).

As an example, the Figure 6 presents a graph with $n = 9$ vertices for which it exists an unique maximum IS of size 6 ($\overline{\alpha} = \alpha(G) = 6$ and $m = 1$). Moreover we take $\underline{s}$ equals to the chromatic number $\chi(G) = 3$. Figure 7 details all the integer partitions of $n = 9$ in the form of Young diagram as well as their order. The optimal solution of $IPP_{\Sigma M}$ is the integer partition without valid predecessor.

**Lemme 1.** *For all* $\mathbf{a}$ *and* $\mathbf{b} \in \Omega(IPP_{\Sigma M})$*, it exists a set of* $k > 1$ *integer partitions* $\mathbf{c_i} \in \Omega(IPP_{\Sigma M})$*, with* $i = 1...k$ *so that* $\mathbf{c_1} = \mathbf{a}$*,* $\mathbf{c_k} = \mathbf{b}$ *and* $\mathbf{c_{i+1}} \in$ *predecessor*$(\mathbf{c_i})$ *or* $\mathbf{c_{i+1}} \in$ *successor*$(\mathbf{c_i})$.

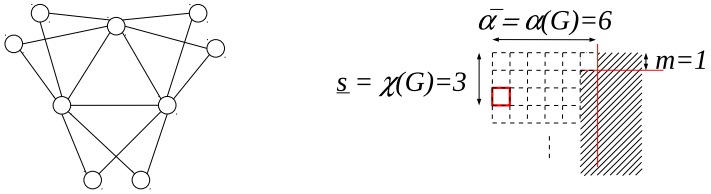


Figure 6: Graph (left figure) with $n = 9$ vertices with $\overline{\alpha} = \alpha(G) = 6$ and $m = 1$ (it exists an unique maximum IS of size 6) and we take $\underline{s} = \chi(G) = 3$. Then, the constraints on the Young diagram (right figure) imply that only one line can have 6 squares, the others has at most 5 square (i.e. no squares in the hatched area) and it must have at least one square on the third line (i.e. red square is mandatory).

*Proof.* It means that the graph of integer partitions (the vertices are the integer partitions and an edge links two integer partitions $\mathbf{a}$ and $\mathbf{b}$ if and only if $\mathbf{a} \in predecessor(\mathbf{b})$ or $\mathbf{a} \in successor(\mathbf{b})$) is connexe. It is evident because it exists always a path between an integer partition and the column integer partition: $(\underbrace{1, ..., 1}_{n})$ where $n$ is the integer to partition. □

**Theorem 4.** *The optimal integer partition of $IPP_{\Sigma M}$ is the integer partition without predecessor.*

*Proof.* If $\mathbf{a}$ is an optimal integer partition of $IPP_{\Sigma M}$ and has at least one predecessor $\mathbf{b} \in predessor(\mathbf{a})$, therefore $f_{\Sigma M}(\mathbf{b}) < f_{\Sigma M}(\mathbf{a})$ by theorem 3 with $\mathbf{b} \in \Omega(IPP_{\Sigma M})$. It refutes the optimality of $\mathbf{a}$. □

## 3.2 Resolution of the relaxed problem $IPP_{\Sigma M}$

The optimal integer partition of $IPP_{\Sigma M}$ is the integer partition without predecessor.

### 3.2.1 Without constraint $a_{\underline{s}} \geq 1$

We define $(IPP_0)$ an intermediate problem corresponding to $IPP_{\Sigma M}$ but without the constraint (13): $a_{\underline{s}} \geq 1$. Let $\mathbf{a}_0^*$ the optimal partition of $(IPP_0)$; this partition is a function of $n$, $\overline{\alpha}$ and $m$.

By definition, $\mathbf{a}_0^*(n, \overline{\alpha}, m)$ satisfies the constraints of $(IPP_0)$ and has no valid predecessor among these constraints. To have the best cost (which implies no predecessor), the partition contains as many lines (i.e. color classes) of maximal size (equal to $\overline{\alpha}$) as possible. $m$ corresponds to the maximum number of lines of size $\overline{\alpha}$, then we take :

$$m = \min\left(\left\lfloor \frac{n}{\alpha(G)} \right\rfloor,\ \#IS(\alpha(G)),\ \alpha(\tilde{G})\right)$$

where $\#IS(k)$ is the number of ISs of $G$ with size equals to $k$ a positive integer. $\alpha(G)$, $\#IS(\alpha(G))$ and $\alpha(\tilde{G})$ can be calculated with the open source code MoMC[3] [13]. If it is too time-consuming, we take: $m = \lfloor \frac{n}{\overline{\alpha}} \rfloor$.

The remaining integer $n - m \times \overline{\alpha}$ uses then as many independent sets of size $\overline{\alpha} - 1$ as possible.

**Theorem 5.** *Let be the euclidean division of $n - m\overline{\alpha}$ by $(\overline{\alpha} - 1)$ :*

$$n - m\overline{\alpha} = q \times (\overline{\alpha} - 1) + r \tag{20}$$

*with $q$ and $r < \overline{\alpha} - 1$ two positive integers, therefore the optimal partition of $(IPP_0)$ is :*

$$\mathbf{a}_0^* = (\overbrace{\overline{\alpha},\ \overline{\alpha},\ ...,\ \overline{\alpha}}^{m}, \overbrace{\overline{\alpha} - 1, ...,\ \overline{\alpha} - 1}^{q}, r) \tag{21}$$

*and the optimal objective function is equal to :*

$$\begin{aligned} \Sigma M_0(n, \overline{\alpha}, m) &= f_{\Sigma M}(\mathbf{a}_0^*) \\ &= \frac{m(m+1)}{2}\overline{\alpha} + \frac{q(2m+q+1)}{2}(\overline{\alpha} - 1) + (m + q + 1)r \end{aligned} \tag{22}$$

[3] code available on: https://home.mis.u-picardie.fr/∼cli/EnglishPage.html

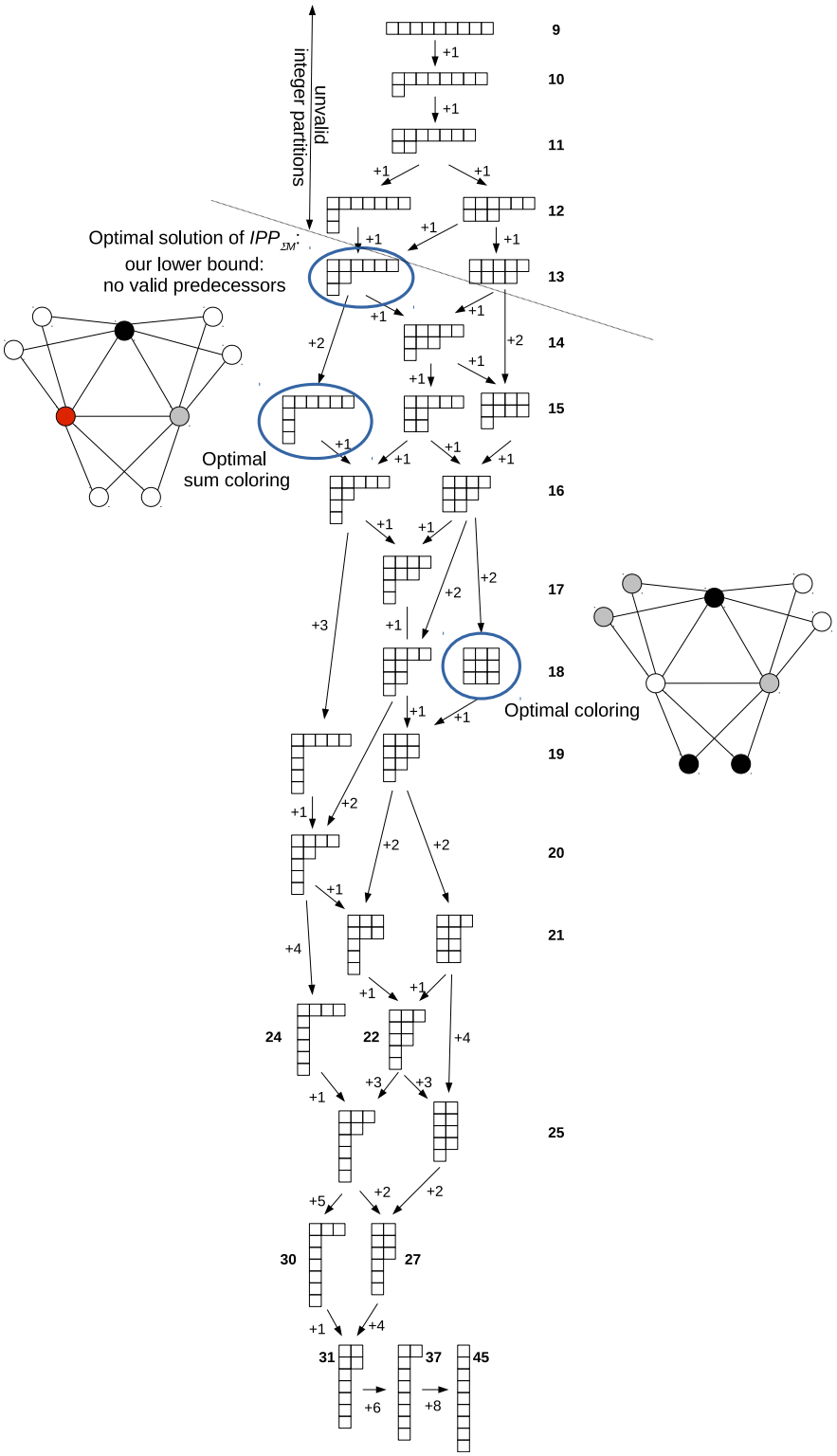


Figure 7: **Young Diagrams of all possible integer partitions succeeding a graph $G$.** Each arrow indicates a successor of a partition and the number indicates the additional cost function (of sum coloring) to pass from a partition to the other.

An young diagram of the optimal partition $\mathbf{a}_0^*$ is given is Figure 8.

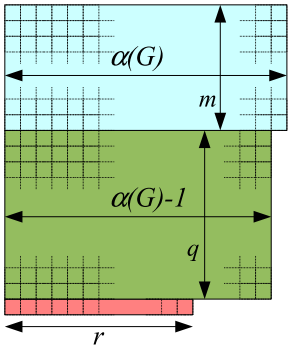


Figure 8: Young diagram of the optimal partition $\mathbf{a}_0^*$ of $(IPP_0)$.

Figure 9 gives an example of optimal partition with $n = 16$, $\overline{\alpha} = \alpha(G) = 4$ and $m = 2$. Notice that in the worst case, when $m$ has its maximal value equals to $\left\lfloor \frac{n}{\alpha(G)} \right\rfloor$, then $q = 0$ so $\mathbf{a}_0^* = (\overbrace{\lfloor n/\alpha(G) \rfloor ..., \lfloor n/\alpha(G) \rfloor}^{\alpha(G)}, r)$.

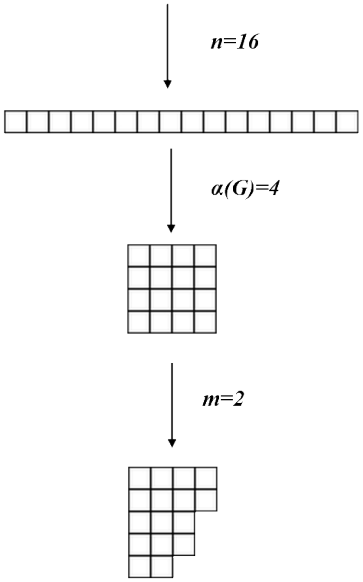


Figure 9: Illustration of how to find the best partition given $n$, $\overline{\alpha} = \alpha(G)$ and $m$. The length of a line can not be over $\alpha(G)$. The number of lines with maximum lengthy can not be over $m$.

### 3.2.2 With constraint $a_{\underline{s}} \geq 1$

If we know that we have to use at least $\underline{s}$ lines (i.e. colors), we have two possibilities :

- the previous solution already uses $\underline{s}$ lines, i.e. $m + q + 1 \geq \underline{s}$. Then the optimal solution of $(IPP_0)$ is also the optimal solution of $IPP_{\Sigma M}$, so :

$$\Sigma M = \Sigma M_0(n, \overline{\alpha}, m) = f_{\Sigma M}(\mathbf{a}_0^*)$$

- the previous solution uses less than $\underline{s}$ lines. Then we start with one vertex in $\underline{s}$ different sets and we solve $(IPP_0)$ with the remaining vertices $(n - \underline{s})$ to find the solution. An illustration is given in Figure 10. Theorem 6 gives the cost of the optimal partition.

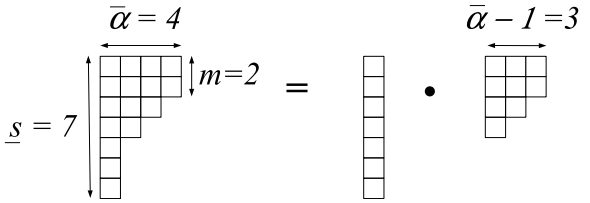


Figure 10: Illustration of how to find the best partition with $n = 16$, $\overline{\alpha} = 4$, $m = 2$ and $\underline{s} = 7$. We add two partitions: the first one is a column of $\underline{s}$ squares; the second one is the optimal partition of one ($IPP_0$) problem with $n' = n - \underline{s}$ squares, $\overline{\alpha}' = \overline{\alpha} - 1$ and $m' = m$.

**Theorem 6.** *Let $\bullet$ be the line by line addition of two partitions. Let four integer $n, \overline{\alpha}, \underline{s}, m$ defining $IPP_{\Sigma M}$. The optimal partition of $IPP_{\Sigma M}$ noted $\mathbf{a}^*$ is equal to:*

$$\mathbf{a}^*(n, \overline{\alpha}, \underline{s}, m) = \mathbf{a}_{1\times\underline{s}} \bullet \mathbf{a}_0^*(n - \underline{s}, \overline{\alpha} - 1, m) \tag{23}$$

*with $\mathbf{a}_{1\times\underline{s}} = (\overbrace{1,\ 1, ...,\ 1}^{\underline{s}})$ the partition of the number $\underline{s}$ into ones and with $\mathbf{a}_0^*(n-\underline{s}, \overline{\alpha}-1, m)$ the optimal partition of ($IPP_0$) with $n - \underline{s}, \overline{\alpha} - 1$ and $m$ parameters, therefore:*

$$\Sigma M = \frac{\underline{s}(\underline{s}+1)}{2} + \Sigma M_0(n - \underline{s}, \overline{\alpha} - 1, m) \tag{24}$$

## 4 Lower bound for graph coloring

The relaxation used for MSCP can also be applied for GVCP. GVCP is a special case of MSCP for which all colors have the same cost, then let be a possible formulation of GVCP :

$$(GVCP) \begin{cases} \text{Min.} & f_\chi(\mathbf{V}) = |\{V_l \mid |V_l| > 0,\ \forall l = 1..n\}| & \\ \text{s.c.} & Eq.\ (2-5) & \\ & V_l \subset V & \forall l = 1..n \end{cases} \tag{25}$$

This formulation differs from MSCP just by the objective function, that counts the number of non-empty partitions. Therefore, the relaxation into an integer partition problem becomes :

$$(IPP_\chi) \begin{cases} \text{Min.} & f_\chi(\mathbf{a}) = |\{a_i | a_i > 0,\ \forall i = 1..n\}| \\ \text{s.c.} & Eq.\ (9-12) \\ & \mathbf{a} = (a_i)_{1\le i\le n} \in \mathbb{N}^n \end{cases} \tag{26}$$

Again $IPP_\chi$ differs from $IPP_{\Sigma M}$ just by the objective function, that corresponds to the number of lines in Young diagram and because Eq. (13) is useless. In this case, the optimal objective function value is equal to :

$$LB_\chi = f_\chi(\mathbf{a}^*) = m + q + [r \neq 0] \tag{27}$$

with $m$, $q$ and $r$ given by the Thm.5 and $\mathbf{a}^*$ the optimal solution of $IPP_\chi$ and [ ] is Iverson bracket : $[r \neq 0] = 1$ if $r \neq 0$ is true and equals 0 otherwise. $LB_\chi$ is a lower bound of $\chi(G)$ and its value depends only of three integers: $n$, $\overline{\alpha}$ and $m$.

## 5 Results

### 5.1 Procedure

To compute our lower bound $\Sigma M$, we use $\overline{G}$ the complementary graph of $G$. We run MoMC solver to find $\alpha(G)$, the size of the maximal clique of $\overline{G}$. We also save the list of all the maximal IS of $G$ to compute $\#is$ and to

build $\tilde{G}$. Algorithm 4 recall the global procedure to compute $\Sigma M$.

**Data:** $G$, $n$

$\overline{\alpha} \leftarrow \begin{cases} \alpha(G) & \text{if it is not too time-consuming with MoMC solver,} \\ \overline{\alpha}(G) & \text{otherwise.} \end{cases}$

$\#is \leftarrow \begin{cases} \#is(\overline{\alpha}) & \text{if it is not too time-consuming with MoMC solver,} \\ \lfloor \frac{n}{\overline{\alpha}} \rfloor & \text{otherwise.} \end{cases}$

Build $\tilde{G}$, the graph of the maximum IS.

$\tilde{\alpha} \leftarrow \begin{cases} \alpha(\tilde{G}) & \text{if it is not too time-consuming with MoMC solver,} \\ \#is & \text{otherwise.} \end{cases}$

$m \leftarrow \min\left(\lfloor \frac{n}{\overline{\alpha}} \rfloor,\ \#is,\ \tilde{\alpha}\right)$

$q \leftarrow \left\lfloor \frac{n - m\overline{\alpha}}{\overline{\alpha} - 1} \right\rfloor$

$r \leftarrow n - m\overline{\alpha} - q \times (\overline{\alpha} - 1)$

$\underline{s} \leftarrow \begin{cases} \underline{\chi}(G) & \text{the best known lower bound of } \chi(G) \text{ if it is known,} \\ \lceil \frac{n}{\overline{\alpha}} \rceil & \text{otherwise.} \end{cases}$

$\Sigma M \leftarrow \begin{cases} \Sigma M_0(n, \overline{\alpha}, m) & \text{if } m + q + 1 \geq \underline{s} \\ \frac{\underline{s}(\underline{s}+1)}{2} + \Sigma M_0(n - \underline{s}, \overline{\alpha} - 1, m) & \text{otherwise} \end{cases}$

**Algorithm 4:** Procedure to compute $\Sigma M$.

## 5.2 Empirical Results and Analysis

We tested our procedure on some graph instances of DIMACS and COLOR benchmarks, which are frequently used for performance evaluation of $MSCP$ algorithms [4]. For several instances, the list of maximum independent set takes too much computational time to be found. There are two possible reasons for this :

- $\overline{G}$ is too dense and MoMC needs too much time to find a maximum clique.
- the number of maximum cliques is too high and we can't compute their graph.

In our tests, we reuse open source code (MoMC and Cliquer) written in C and we write a script coded in Python [4]. The results were obtained with an Intel Xeon E5 2.50GHz processor - 8 cores and 16GB of RAM.

In the Table 1, we show a set of results, focused on graphs where our lower bound could be computed. We compare our lower bound, $\Sigma M$, to $\Sigma_{old}$ (the best lower bound known according to [6], [7], [3]) and $LBM\Sigma$ (lower bound described in [7], that we calculate or update for several graphs). The three first columns indicate the graph instance name, its number of vertices and its density. Columns 4 and 5 give the maximum independent set of the graph, $\alpha(G)$ and the time (in second) required to compute it with the MoMC solver. The two next columns show the number of maximum independent set of the graph, $\#is$ and the time (in second) required to compute it with the MoMC solver or the Cliquer solver. Columns 8 and 9 indicate, $m$, the maximum number of *compatibles* independent sets of the graph and the time (in second) required to compute it with the MoMC solver. If $\#is$ is too high ($> 5000$), then the computation of $m$ is not done because it would take too long. Column 10 gives the chromatic number of the graph $\chi(G)$, if it is known and its interval of belonging otherwise. Note that this information comes from other algorithms after a computation-time that may be very long. Next column provides the result of our lower bound, $LB_\chi$, computed as in section 4. Last five columns display, $\Sigma(G)$, the chromatic sum when it is known; $\Sigma_{old}$, the best lower bound of $\Sigma(G)$ known in the literature; $LBM\Sigma$, the lower bound presented in [7]; $\Sigma M_0$, our lower bound computed without $\underline{s} = \underline{\chi}(G)$; $\Sigma M$, our lower bound computed with $\underline{s} = \underline{\chi}(G)$.

[4] code available on: github.com/gondran/LowBoundSumColoring

Table 1: Results of *weak optimality* tests on some graphs of DIMACS and COLOR benchmarks.

| Instances | $n$ | $d$ | $\alpha(G)$ | time | $\#is$ | time | $m$ | time | $\chi(G)$ | $LB_\chi$ | $\Sigma(G)$ | $\Sigma_{old}$ | $LBM\Sigma$ | $\Sigma M_0$ | $\Sigma M$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| myciel3 | 11 | 0.36 | 5 | $< 0.1$ | 2 | $< 0.1$ | 1 | 0 | 4 | 3 | ? | **20** | **20** | 19 | **20** |
| myciel4 | 23 | 0.28 | 11 | $< 0.1$ | 2 | $< 0.1$ | 1 | 0 | 5 | 3 | ? | **41** | **41** | 37 | **41** |
| myciel5 | 47 | 0.22 | 23 | $< 0.1$ | 1 | $< 0.1$ | 1 | 0 | 6 | 3 | ? | 80 | **81** | 73 | **81** |
| myciel6 | 95 | 0.17 | 47 | $< 0.1$ | 1 | $< 0.1$ | 1 | 0 | 7 | 3 | ? | **158** | **158** | 145 | **158** |
| myciel7 | 191 | 0.13 | 95 | $< 0.1$ | 1 | 0.1 | 1 | 0 | 8 | 3 | ? | **308** | **308** | 289 | **308** |
| queen5_5 | 25 | 0.53 | 5 | $< 0.1$ | 10 | $< 0.1$ | 5 | $< 0.1$ | 5 | **5** | **75** | **75** | **75** | **75** | **75** |
| queen6_6 | 36 | 0.46 | 6 | $< 0.1$ | 4 | $< 0.1$ | 4 | $< 0.1$ | 7 | **7** | ? | 126 | 127 | **129** | **129** |
| queen7_7 | 49 | 0.40 | 7 | $< 0.1$ | 40 | $< 0.1$ | 7 | $< 0.1$ | 7 | **7** | **196** | **196** | **196** | **196** | **196** |
| queen8_8 | 64 | 0.36 | 8 | $< 0.1$ | 92 | $< 0.1$ | 6 | $< 0.1$ | 9 | **9** | **291*** | 288 | 289 | **291** | **291** |
| queen8_12 | 96 | 0.30 | 8 | $< 0.1$ | 195 271 | 0.3 | $\#is$ | $\times$ | 12 | **12** | ? | **624** | **624** | **624** | **624** |
| queen9_9 | 81 | 0.33 | 9 | $< 0.1$ | 352 | $< 0.1$ | 7 | $< 0.1$ | 10 | **10** | ? | 405 | 406 | **408** | **408** |
| queen10_10 | 100 | 0.30 | 10 | $< 0.1$ | 724 | 0.2 | 8 | 0.8 | 11 | **11** | **553*** | 550 | 551 | **553** | **553** |
| queen11_11 | 121 | 0.27 | 11 | $< 0.1$ | 2 680 | 0.9 | 11 | 207 | 11 | **11** | **726** | **726** | **726** | **726** | **726** |
| queen12.12 | 144 | 0.25 | 12 | $< 0.1$ | 14 200 | 6** | $\#is$ | $\times$ | 12 | **12** | **936** | **936** | **936** | **936** | **936** |
| queen13.13 | 169 | 0.23 | 13 | $< 0.1$ | 73 712 | 0.5** | $\#is$ | $\times$ | 13 | **13** | **1 183** | **1 183** | **1 183** | **1 183** | **1 183** |
| queen14.14 | 196 | 0.22 | 14 | $< 0.1$ | 365 596 | 3** | $\#is$ | $\times$ | 14 | **14** | **1 470** | **1 470** | **1 470** | **1 470** | **1 470** |
| queen15.15 | 225 | 0.21 | 15 | $< 0.1$ | 2 279 184 | 17** | $\#is$ | $\times$ | 15 | **15** | **1 800** | **1 800** | **1 800** | **1 800** | **1 800** |
| queen16.16 | 256 | 0.19 | 16 | $< 0.1$ | 14 772 512 | 113** | $\#is$ | $\times$ | 16 | **16** | ? | **2 176** | **2 176** | **2 176** | **2 176** |
| 2-Insertions_3 | 37 | 0.11 | 18 | $< 0.1$ | 1 | $< 0.1$ | 1 | 0 | 4 | 3 | ? | 55 | **59** | 58 | **59** |
| 3-Insertions_3 | 56 | 0.07 | 27 | $< 0.1$ | 11 | $< 0.1$ | 1 | 0 | 4 | 3 | ? | 84 | 88 | 88 | **89** |
| DSJC125.1 | 125 | 0.09 | 34 | $< 0.1$ | 747 | $< 0.1$ | 1 | 0 | 5 | 4 | ? | 297 | 297 | 299 | **300** |
| DSJC125.5 | 125 | 0.50 | 10 | $< 0.1$ | 2 | $< 0.1$ | 1 | 0 | 17 | 14 | ? | 851 | 855 | 918 | **924** |
| DSJC125.9 | 125 | 0.90 | 4 | $< 0.1$ | 9 | $< 0.1$ | 5 | $< 0.1$ | 44 | 40 | ? | 2 108 | 2 124 | 2 475 | **2 487** |
| DSJC250.5 | 250 | 0.50 | 12 | $< 0.1$ | 2 | $< 0.1$ | 2 | $< 0.1$ | $[\![26,\ 28]\!]$ | 23 | ? | 2 745 | 2 745 | 2 924 | **2 930** |
| DSJC250.9 | 250 | 0.90 | 5 | $< 0.1$ | 3 | $< 0.1$ | 2 | $< 0.1$ | 72 | 62 | ? | 6 651 | 6 678 | 7 815 | **7 882** |
| DSJC500.5 | 500 | 0.50 | 13 | 2 | 51 | 3 | 9 | $< 0.1$ | $[\![43,\ 47]\!]$ | 41 | ? | 9 867 | 9 877 | 10 336 | **10 339** |
| DSJC500.9 | 500 | 0.90 | 5 | $< 0.1$ | 23 | $< 0.1$ | 15 | $< 0.1$ | $[\![123,\ 126]\!]$ | 122 | ? | 25 581 | 25 581 | 29 766 | **29 768** |
| DSJC1000.5 | 1000 | 0.50 | 15 | 159 | 12 | 290 | 6 | $< 0.1$ | $[\![73,\ 82]\!]$ | 71 | ? | 33 835 | 33 856 | 35 805 | **35 808** |
| DSJC1000.9 | 1000 | 0.90 | 6 | 0.1 | 3 | 0.1 | 3 | $< 0.1$ | $[\![216,\ 222]\!]$ | 200 | ? | 85 235 | 85 294 | 99 906 | **100 078** |
| DSJR500.1c | 500 | 0.97 | 13 | $< 0.1$ | 4 | $< 0.1$ | 2 | $< 0.1$ | 85 | 42 | ? | **15 398** | 11 040 | 10 587 | 11 619 |
| DSJR500.5 | 500 | 0.47 | 7 | 0.3 | 18 | 0.8 | 2 | $< 0.1$ | 122 | 83 | ? | **23 609** | 19 599 | 20 919 | 21 832 |
| flat300_20_0 | 300 | 0.48 | 15 | $< 0.1$ | 20 | $< 0.1$ | 20 | $< 0.1$ | 20 | **20** | **3 150** | **3 150** | **3 150** | **3 150** | **3 150** |
| flat300_26_0 | 300 | 0.48 | 12 | $< 0.1$ | 31 | 1 | 14 | $< 0.1$ | 26 | **26** | **3 966*** | 3 901 | 3 901 | **3 966** | **3 966** |
| flat300_28_0 | 300 | 0.48 | 12 | $< 0.1$ | 45 | 1 | 6 | $< 0.1$ | 28 | 27 | ? | 3 906 | 3 906 | 4 098 | **4 099** |
| flat1000_50_0 | 1000 | 0.49 | 20 | 36 | 50 | 78 | 50 | $< 0.1$ | 50 | **50** | **25 500** | **25 500** | **25 500** | **25 500** | **25 500** |
| flat1000_60_0 | 1000 | 0.49 | 17 | 89 | 42 | 199 | 40 | $< 0.1$ | 60 | **60** | **30 100*** | 29 914 | 29 914 | **30 100** | **30 100** |
| flat1000_76_0 | 1000 | 0.49 | 15 | 184 | 21 | 394 | 8 | $< 0.1$ | 76 | 71 | ? | 33 880 | 33 880 | 35678 | **35 693** |

* new optimal solution that we proved

** computation was done with the Cliquer[5] [14] solver because for listing all maximum cliques Cliquer is faster than MoMC.

On the 37 graphs of this test set, we improve the result for 18 graphs (light blue in Table 1). For 17 graphs we find the same lower bound as in the literature. For 2 graphs, our lower bound is worse than the best lower bound of the literature. The reason is that for these graphs, the approach used by Moukrim et al. [15, 16] is totally different of ours, based on the decomposition of the graph into partition of cliques. For 9 graphs, the optimal lower bound was already found with the much simpler $LBM\Sigma$ lower bound. We have proven the exact value of $\Sigma(G)$ for 4 graphs (with * in Table 1), and have computed an example of optimal coloring. Our approach is similar to that of Lecat et al. [7] but we outperformed their results by introducing an additional constraint with the integer $m$.

Our lower bound of $\chi(G)$, $LB_\chi$, is equal to the chromatic number for all *queen* graph instances (13 graphs) and for 4 *flat* graph instances. Note that for five of those instances (*queen5_5*, *queen7_7*, *queen11_11*, *flat300_20_0* and *flat1000_50_0*), we also found the optimal coloring. Indeed, in those cases, the decomposition of the number of vertices is : $n = \alpha(G) \times m + (\alpha(G) - 1) \times q + r$ with $q = r = 0$, i.e. $\chi(G) = m$. Therefore we know the $m$ compatible independent sets that composed the optimal solution.

Moreover, our $LB_\chi$ is computed in 78 $s$ and 199 $s$ for respectively *flat1000_50_0* and *flat1000_60_0* graph instances while it takes respectively 3 331$s$ and 29 996$s$ with the most powerful method [17] to find a lower bound of $\chi(G)$.

## 6 Conclusion

We presented a new way to find lower bounds for the $MSCP$ and the $GVCP$. In order to do this, we explained how to relax $MSCP$ into an integer partition problem that can be exactly solved. We can select the constraints we want to keep in this integer partition problem, and we proposed a set of constraints used to define the $\Sigma M$ lower bound. We carried out experiments and improved the best known lower bound for 18 graphs of standard benchmark DIMACS. We also proved the optimality of 4 of them.

It is also possible to add even more constraints in the integer partition problem. Further researches could use this approach to keep improving the lower bound.

## References


[1] M. Malafiejski, Sum coloring of graphs, in: Kubale [2], pp. 55–66.

[2] M. Kubale (Ed.), Graph Colorings, Vol. 352 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, Rhode Island, USA, 2004.

[3] Q. Wu, Q. Zhou, Y. Jin, J. Hao, Minimum sum coloring for large graphs with extraction and backward expansion search, Appl. Soft Comput. 62 (2018) 1056–1065. doi:https://doi.org/10.1016/j.asoc.2017.09.043.

[4] Y. Jin, J.-K. Hao, Hybrid evolutionary search for the minimum sum coloring problem of graphs, Information Sciences (2016) 15–34.

[5] E. Kubicka, G. Kubicki, D. Kountanis, Approximation algorithms for the chromatic sum, in: Computing in the 90's, Springer, 1991, pp. 15–21.

[6] Y. Jin, J.-P. Hamiez, J.-K. Hao, Algorithms for the minimum sum coloring problem: a review, Artificial Intelligence Review (2017) 367–394.

[7] C. Lecat, C. Lucet, C.-M. Li, New Lower Bound for the Minimum Sum Coloring Problem., in: AAAI, 2017, pp. 853–859.

[8] D. S. Johnson, M. Trick (Eds.), Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, 1993, Vol. 26 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, RI, USA, 1996.

[9] Q. Wu, J.-K. Hao, Improved lower bounds for sum coloring via clique decomposition, CoRR abs/1303.6761. URL http://dblp.uni-trier.de/db/journals/corr/corr1303.html#abs-1303-6761

[10] A. Young, On quantitative substitutional analysis, Proceedings of the London Mathematical Society 33 (1900) 97–145.

[11] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness, Freeman, San Francisco, CA, USA, 1979.

[12] L. G. Valiant, The complexity of enumeration and reliability problems, SIAM J. Comput. 8 (3) (1979) 410–421. doi:10.1137/0208032.

[13] C.-M. Li, H. Jiang, F. Manyá, On minimization of the number of branches in branch-and-bound algorithms for the maximum clique problem, Computers & Operations Research 84 (2017) 1–15. doi:https://doi.org/10.1016/j.cor.2017.02.017.

[14] P. R. Östergård, A new algorithm for the maximum-weight clique problem, Nordic Journal of Computing (2001) 424–436.

[15] A. Moukrim, K. Sghiouer, C. Lucet, Y. Li, Lower Bounds for the Minimal Sum Coloring Problem, Electronic Notes in Discrete Mathematics 36 (2010) 663–670, iSCO 2010 - International Symposium on Combinatorial Optimization.

[16] A. Moukrim, K. Sghiouer, C. Lucet, Y. Li, Upper and Lower Bounds for the Minimum Sum Coloring Problem, Tech. rep., Université de Technologie de Compiègne and Université de Picardie Jules Verne (2013).
URL https://www.hds.utc.fr/~moukrim/dokuwiki/_media/en/mscp_cor13septembre2013.pdf

[17] S. Held, W. Cook, E. Sewell, Maximum-weight stable sets and safe lower bounds for graph coloring, Mathematical Programming Computation 4 (4) (2012) 363–381. doi:10.1007/s12532-012-0042-3.