# A Lower Bound on the Chromatic Number of Mycielski Graphs

Massimiliano Caramia *  Paolo Dell'Olmo †

**Abstract**

In this work we give a new lower bound on the chromatic number of a Mycielski graph $M_i$. The result exploits a mapping between the coloring problem and a multiprocessor task scheduling problem. The tightness of the bound is proved for $i = 1, \ldots, 8$.

**keywords:** Benchmarks, Chromatic number, Lower bound, Mycielski graphs, Multiprocessor task scheduling.

## 1 Introduction and notation

Let $G$ be an undirected graph, with vertex (node) set $V$ and edge set $E$. A *proper coloring* is a function $f : V \to \{1, 2, \ldots, |V|\}$ such that $f(i) \neq f(j)$ for each $(i, j) \in E$, i.e. no two adjacent nodes have assigned the same color. Given a graph $G$, the smallest number of colors for which there exists a proper coloring of $G$ is called *chromatic number* and is denoted by $\chi(G)$. The chromatic number problem is, in the general case, $\mathcal{NP}$-complete [5]. Moreover let $\omega(G)$ and $\alpha(G)$ be respectively the size of the maximum clique and the size of the maximum stable set of a graph $G$, i.e. the maximum

*Dipartimento di Informatica, Sistemi e Produzione, University of Rome "Tor Vergata", Via di Tor Vergata, 110 - 00133 Rome, Italy. `e-mail: caramia@disp.uniroma2.it`

†Dipartimento di Statistica, Probabilitá e Statistiche Applicate, University of Rome "La Sapienza", Piazzale Aldo Moro, 5 - 00185 Rome, Italy. `e-mail: dellolmo@sunshine.iasi.rm.cnr.it`

number of nodes in $V$ mutually adjacent and the maximum number of nodes in $V$ mutually non adjacent.

When facing with exact algorithms for $\mathcal{NP}$-complete problems it is important, in order to verify their effectiveness, to use, along with randomly generated instances, benchmark instances such as particular structured graphs. Referring to the chromatic number problem there are instance classes that are widely used for testing algorithms: *Register graphs* [3], *Geometric graphs*, *Book graphs* [7], *Game graphs* [7], *Miles graphs* [7], *Queen graphs* [4], *Mycielski graphs* [10]. In particular, from the experimental results presented in literature (see for example [2, 9, 13]), Mycielski graphs seem to be, with respect to the others, hard-to-color instances. This can be explained by examining the structure of this graphs.

Given a graph $G$ with vertex set $\{v_1, v_2, \ldots, v_n\}$, we can get the *Mycielski transformation* $\mu(G)$ of $G$ by creating a new graph with vertex set:
$\{x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n, z\}$ and edges $(x_h, x_k)$ iff $(v_h, v_k) \in E$, $(x_h, y_k)$ iff $(v_h, v_k) \in E$, and $(y_h, z)$ for all $h$. If we let $M_1$ be the Mycielski graph with two nodes and one single edge, $M_i$ is recursively defined as $M_{i+1} = \mu(M_i)$. For the sake of simplicity in the sequel referring to a graph $M_i$ we define: $X_i = \{x_1, x_2, \ldots, x_n\}$, $Y_i = \{y_1, y_2, \ldots, y_n\}$, and $z_i = \{z\}$.

Some structural properties of these graphs can be underlined:

*i*) each $M_i$ is triangle free, i.e. $\omega(M_i) = 2$,

*ii*) has increasing chromatic number equal to $i + 1$,

*iii*) the subgraph induced by the nodes $\{x_1, x_2, \ldots, x_n\}$ of $M_{i+1}$ is the same subgraph induced by the nodes $\{v_1, v_2, \ldots, v_n\}$ of $M_i$.

Accordingly to the above definitions and property *iii*), we define: $X_i$ the set of nodes in $M_i$ which induces a $M_{i-1}$, $X_{i-1} \subset X_i$ the set of nodes in $M_i$ which induces a $M_{i-2}$, and $X_{i-2} \subset X_{i-1} \subset X_i$ the set of nodes in $M_i$ which induces a $M_{i-3}$

As such, these graphs seem difficult to color using an implicit enumeration algorithm like branch and bound: in fact exact algorithms in the literature solve instances up to $M_4$ [9] and $M_5$ [2, 13].

This is explainable because neither $\omega(M_i)$ nor the linear relaxation of the independent set formulation of the problem can provide tight lower bounds. The results present is the literature show that the dual integer solution of the linear relaxation of the program [9] yield to weak bounds, especially if the we deal with large $i$.

The some reasons apply to the slight improvement on $\omega(M_i)$ obtained for low indexed instances by the lower bound expression $\lceil\frac{|V_i|}{\alpha(M_i)}\rceil$ which is always less or equal to three and becomes closer to $\omega(M_i)$ as $i$ increases.

Our idea is to transform the coloring problem of $M_i$ in a scheduling problem and derive a new lower bound from this latter that is tight for $i = 1, \ldots, 8$ and in particular is tight for Mycielski benchmark graphs currently used in exact algorithms.

# 2 Transforming the coloring problem in a multiprocessor task scheduling problem

Starting from a Mycielski graph $M_i = (V_i, E_i)$ let us define one instance of a multiprocessor task scheduling problem with a prespecified processor allocation [1].

There are $|E_i|$ processors and $|V_i|$ non preemptive tasks of unit execution time. Each task $j$ in order to be executed requires the simultaneous allocation of a set $P_j$ of processors. Processors are allocated to tasks so that the intersection graph of the tasks processor set is the given Mycielski graph. This can be done by the following algorithm:

**Algorithm Assign Processors**
**Input**: A Mycielski graph $M_i$;
**Output**: An allocation of processors to tasks;
*begin algorithm*

**Step 1** $k = 1$;
$P_j = \emptyset \; \forall j \in V_i$;
for $j := 1$ to $|V_i|$ do
for each $j' > j$ adjacent to $j$ do

**Step 1.1** if the processor $k \notin P_j$ then $P_j = P_j \cup k$;

**Step 1.2** if the processor $k \notin P_{j'}$ then $P_{j'} = P_{j'} \cup k$;

**Step 1.3** $k = k + 1$;

*end algorithm.*

**Remark 1** *The algorithm Assign Processors uses a number $k$ of processors equal to $|E_i|$.*

**Remark 2** *The number of processors assigned to each task is equal to the degree of the corresponding node.*

We recall that a mapping $\phi : V \rightarrow 2^N$, where $N$ is an arbitrary finite set, is an intersection representation of $G$ if it has the following property:

$$(j, j') \in E \Longleftrightarrow \phi(j) \cap \phi(j') \neq \emptyset.$$

Let $\theta(G)$ be the minimum cardinality $|N|$ of a set $N$ for which an intersection representation $\phi : V \rightarrow 2^N$ of $G$ exists. The quantity $\theta(G)$ is commonly called the *intersection number* of a graph $G$ (Harary [6], Roberts [12] and West [14]). It was proved by Marczewski [8] that $\theta(G) < \infty$ for any $G$, i.e. every graph has an intersection representation. In particular it is known ([12]) that given a graph $G$ its intersection number is equal to its edge clique covering number. As $M_i$ is triangle free its edge clique covering number is $|E_i|$ and then $\theta(M_i) = |E_i|$. Thus:

**Claim 1** *The minimum number of processors such that the intersection graph of the processor task requirements is the given Mycielski graph $M_i$ is exactly $|E_i|$.*

Now notice that tasks $j$ and $j'$ for which $P_j \cap P_{j'} \neq \emptyset$ can not be executed simultaneously (i.e. in parallel). Let us denote a set of tasks which can be executed simultaneously "feasible set" of tasks. A schedule $s$ is defined by assigning a starting time to each task such that processor constraints are satisfied. Let $C_s$ be the length of a schedule $s$. A schedule is optimal if it is of minimum length. We denote the optimal schedule length by $C_{max}$.

Moreover, we denote with $S^i$ the set of all schedules of the instance corresponding to $M_i$, such that $C^i_{max} = min_{s \in S^i} C_s$.

One time instant in the schedule can be considered as a color assigned to the tasks (nodes) scheduled in that instant, thus the following claim can be formalized:

**Claim 2** *The minimum completion time $C^i_{max}$ of the multiprocessor task scheduling problem obtained from $M_i$ is equal to $\chi(M_i)$.*

**Definition 1** *Let us define with $a_{s,t}$ the idle area at the instant $t$ in a schedule $s \in S^i$, i.e. the number of idle processors at that instant in that schedule.*

Let $a^1_i$ be the difference between the total number of processors $|E_i|$ and the maximum number of different processors that can work simultaneously in the same instant, i.e.

$$a^1_i = min_{s \in S^i, t} a_{s,t} \tag{1}$$

is the minimum number of processors idle at the same time.

Let $P_j$ be the set of processors assigned to the task $j \in V_i$, and let $\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}$ be the minimum total idle area obtainable among all schedules in $S^i$ corresponding to $M_i$. It is known that [11]:

$$C^i_{max} = \lceil \frac{\sum_{j=1}^{|V_i|} |P_j| + \min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}}{|E_i|} \rceil.$$

Recalling Remark 2 the number of processors assigned to a task is equal to the number of its adjacent tasks (nodes) in $M_i$, and then $\sum_{j=1}^{|V_i|} |P_j| = 2 * |E_i|$. The previous expression becomes:

$$C^i_{max} = 2 + \lceil \frac{\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}}{|E_i|} \rceil.$$

We have reduced the initial problem of finding the chromatic number of $M_i$ to finding $\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}$. The latter is a combinatorial optimisation problem where one has to find a complete schedule $s^* \in S^i$ with the minimum total idle area. Note that in the above inequality the term 2 is the cardinality of the maximum clique; thus the term $\lceil \frac{\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}}{|E_i|} \rceil$ represents an additive quantity over the clique bound. In the next section we compute a lower bound on $\lceil \frac{\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}}{|E_i|} \rceil$ by analysis of cases. In particular we need only two cases to prove the tightness of the lower bound on $\chi(M_i)$ for $i = 1, \ldots, 8$.

# 3 Lower bound calculation

A lower bound on $\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}$ can be computed considering that for each instant of any schedule $s \in S^i$ there are at least $a_i^1$ processors which are idle (non processing any task). Next we will also prove $a_i^1 > 0$.

Because there are no precedence relations among tasks, feasible sets in the optimal schedule $s^*$ can be rearranged such that $a_{s*,1} \leq a_{s*,2}, \leq \ldots, \leq a_{s*,C_{max}^i}$. In general, we can consider the whole set of feasible sets and order them $f_1, f_2, \ldots$ in non decreasing values of idle area, obtaining:

$$\sum_{t=1}^{C_{max}^i} a_{s^*,t} \geq a_i^1 + a_i^2 + \ldots + a_i^{C_{max}^i} \tag{2}$$

where $a_i^1, a_i^2, \ldots, a_i^{C_{max}^i}$ are the idle area associated respectively to feasible sets $f_1, f_2, \ldots, f_{C_{max}^i}$.

Let $LB^i$ be a lower bound on $C_{max}^i$, then recalling (1) the following expression holds:

$$\sum_{t=1}^{C_{max}^i} a_{s^*,t} \geq a_i^1 + \sum_{t=2}^{C_{max}^i} a_{s^*,t}.$$

**Case 1:** $a_{s*,1} = a_i^1$

If the above hypothesis is verified we have:

$$\sum_{t=1}^{C_{max}^i} a_{s^*,t} = a_i^1 + \sum_{t=2}^{C_{max}^i} a_{s^*,t}.$$

The number of edges of $M_i$ can be recursively defined as:

$$|E_i| = |E_{i-1}| + 2 * |E_{i-1}| + |Y_i| = 3 * |E_{i-1}| + |Y_i| \tag{3}$$

$$|E_1| = 1 \tag{4}$$

where by construction of $M_i$:

- $|E_{i-1}|$: number of edges of $M_{i-1}$ that is the subgraph induced by $X_i$ (see property *iii*) in section 1);

- $2 * |E_{i-1}|$: number of edges linking nodes of $Y_i$ with nodes of $X_i$;

- $|Y_i|$: number of edges linking nodes of $Y_i$ to $z$, i.e. degree of $z$;

By hypothesis of Case 1 we have:

$$C_{max}^i = 2 + \lceil \frac{\min_{s \in S^i} \sum_{t=1}^{C_s} a_{s,t}}{|E_i|} \rceil = 2 + \lceil \frac{a_i^1 + \sum_{t=2}^{C_{max}^i} a_{s*,t}}{|E_i|} \rceil.$$

In the scheduling problem associated to a Mycielski graph $M_i$ the minimum idle area obtainable is given by:

$$a_i^1 = |E_i| - \sum_{j \in Y_i} |P_j| = |E_i| - 2 * |E_{i-1}| - |Y_i| \tag{5}$$

where $\sum_{j \in Y_i} |P_j|$ is the maximum number of processors which can work simultaneously in any schedule of the problem associated to $M_i$. Substituting (3) into (5) we have:

$$a_i^1 = |E_{i-1}| > 0.$$

Once we scheduled at instant 1 the tasks in $Y_i$, we can state that we need at least $C_{max}^{i-1}$ instant of time to schedule the remaining tasks, as we have to schedule among the remaining tasks a $M_{i-1}$, i.e. the subgraph induced by $X_i$. This leads to have an idle area at least equal to $|E_i| * C_{max}^{i-1} - 2 * |E_i| + 2 * |E_{i-1}| + |Y_i|$ obtained by subtracting the processors of all the remaining tasks to be scheduled $(2 * |E_i| - 2 * |E_{i-1}| - |Y_i|)$ from the lower bound of the area $|E_i| * C_{max}^{i-1}$ of the schedule from 2 to $C_{max}^i$ where the remaining tasks can be executed, obtaining:

$$C_{max}^i \geq 2 + \lceil \frac{a_i^1 + |E_i| * C_{max}^{i-1} - 2 * |E_i| + 2 * |E_{i-1}| + |Y_i|}{|E_i|} \rceil$$

Thus, we have:

$$C_{max}^i \geq 2 + \lceil \frac{|E_{i-1}| + |E_i| * C_{max}^{i-1} - 2 * |E_i| + 2 * |E_{i-1}| + |Y_i|}{|E_i|} \rceil$$

$$C_{max}^i \geq \lceil \frac{2 * |E_i| + |E_{i-1}| + |E_i| * C_{max}^{i-1} - 2 * |E_i| + 2 * |E_{i-1}| + |Y_i|}{|E_i|} \rceil$$

$$C^i_{max} \geq \lceil \frac{|E_i| * C^{i-1}_{max} + 3 * |E_{i-1}| + |Y_i|}{|E_i|} \rceil$$

Substituting (3) into the previous inequality we have:

$$C^i_{max} \geq \lceil \frac{|E_i| * C^{i-1}_{max} + |E_i|}{|E_i|} \rceil$$

$$C^i_{max} \geq C^{i-1}_{max} + 1 \tag{6}$$

**Claim 3** $C^i_{max} = i + 1$ *for each* $i$.

**Proof:** In order to prove Claim 3 we find an admissible schedule $s$ with completion time $C^i_s$, and hence an upper bound on $C^i_{max}$. An easy way to do this is described by the following recursive expression that takes into account the property of $M_i$ to contain a $M_{i-1}$ (the subgraph induced by the node set $X_i$) as a proper induced subgraph:

$$C^i_{max} \leq C^{i-1}_{max} + 1 \tag{7}$$

In fact, starting from a schedule of $M_{i-1}$, i.e. the subgraph induced by $X_i$, having completion time $C^{i-1}_{max}$, all tasks in $Y_i$ can be executed in the same instant $C^{i-1}_{max} + 1$ as they form a feasible set. This means that we can assign to the node set $Y_i$ the same color $C^{i-1}_{max} + 1$. Moreover $z_i$ can be processed in any arbitrary instant in the set $\{1, \ldots, C^{i-1}_{max}\}$.

From (6) and (7) we have:

$$C^i_{max} = C^{i-1}_{max} + 1 \tag{8}$$

Equation (8) can be solved with $C_1 = 2$, obtaining $C^i_{max} = i + 1$, which proves the thesis. □

**Case 2:** $a_{s*,1} > a^1_i$

If in the optimal schedule there is not $a^1_i$, then we can neglect those schedules having at instant 1 a feasible set formed only by a subset of tasks in $Y_i$, otherwise from the optimal schedule we would have moved the remaining tasks in $Y_i$ to that instant obtaining back the result of Case 1.

Secondly, it is trivial to consider at instant 1 those feasible sets, different from $a_i^1$, which leave a $M_{i-1}$ as a proper induced subgraph otherwise we fall in the same computation of Case 1, i.e. we have $LB^i = i+1$.

By construction, the first feasible set, in terms of area idle, that does not overlap with Case 1, is formed by $z_i$, $z_{i-1}$ and the greatest (i.e. with minimum idle area) feasible set from $X_{i-2}$. This feasible set returns the following area idle $a_i^k$:

$$a_i^k = |E_i| - (6 * |E_{i-3}| + 2 * |Y_{i-2}| + 2 * |Y_{i-1}| + |Y_i|)$$

Hence:

$$C_{max}^i \geq 2 + \frac{a_i^k * C_{max}^i}{|E_i|} \tag{9}$$

and the lower bound is:

$$LB^i = 2 + \lceil \frac{a_i^k * LB^i}{|E_i|} \rceil \tag{10}$$

Thus:

$$LB^i = \lceil \frac{2 * |E_i|}{|E_i| - a_i^k} \rceil \tag{11}$$

In order to compute the above expression, let us define $V_i$ and $Y_i$ in terms of $i$:

$$|V_i| = 3 * 2^{i-1} - 1$$

$$|Y_i| = \frac{(3 * 2^{i-1} - 1) - 1}{2} = 3 * 2^{i-2} - 1$$

and let us solve the recursive expression (3), (4) as follows:

$$|E_i| = 3^{i-1} + \sum_{k=0}^{i-2} 3^k * |Y_{i-k}| = 3^{i-1} + \sum_{k=0}^{i-2} 3^k * (3 * 2^{i-k-2} - 1) =$$

$$= 3^{i-1} + \sum_{k=0}^{i-2} 3^{k+1} * 2^{i-k-2} - \sum_{k=0}^{i-2} 3^k = 3^{i-1} + \frac{3 * 2^i}{4} * \sum_{k=0}^{i-2} (\frac{3}{2})^k - \sum_{k=0}^{i-2} 3^k =$$

$$= \frac{7}{2} * 3^{i-1} - \frac{3}{2} * 2^i + \frac{1}{2}. \tag{12}$$

Now substituting the above value in (11) we obtain:

$$LB^i = \lceil \frac{\frac{7}{3} * 3^i + 3 * 2^i + 1}{\frac{7}{27} * 3^i + \frac{3}{4} * 2^i - 2} \rceil$$

Note that as $i$ grows $LB_i$ tends to 9.

The following claim can be easily verified:

**Claim 4** $LB^i = i + 1$ *for* $i = 1, \ldots, 8$.

□

As we are dealing with a lower bound we have to consider as $LB^i$ the expression found in Case 2.

## 4 Conclusions

A lower bound on the chromatic number of $M_i$ has been defined exploiting analogies between a coloring problem and a multiprocessor task scheduling problem. The proposed bound is tight for $i = 1, \ldots, 8$.

## References


[1] L., Bianco, P. Dell'Olmo and M. G., Speranza: Nonpreemptive scheduling of independent tasks with prespecified processors allocation. Naval Research Logistic 41 (1994), 959-971.

[2] M. Caramia and P. Dell'Olmo: Iterative Coloring Extension of a Maximum Clique. Tech. Rep. Centro Vito Volterra, University of Rome Tor Vergata, submitted (1998).

[3] A. Condon and G. Lewandowski: Experiments with parallel graphs coloring heuristics and applications of graph coloring, Second DIMACS Challenge: Cliques, Coloring, and Satisfiability, in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, D.S. Johnson and M.A. Trick (eds) (1996).

[4] M. Gardner: The Unexpected Hanging and Other Mathematical Diversions. Simon and Schuster, New York (1969).

[5] M. R. Garey and D. S. Johnson: Computers and Intractability. W. H. Freeman and Co., San Francisco (1979).

[6] F. Harary: Graph theory. Addison-Wesley, Reading, MA, (1969).

[7] D. E. Knuth: The Standford GraphBase. ACM Press, Addison-Wesley, New York (1993).

[8] E. Marczewski: Sur deux propriétés des classes d'ensembles. Fund. Math. 33 (1945), 303–307.

[9] A. Mehrotra and M. A. Trick: A Column Generation Approach for Graph Coloring. INFORMS J. on Computing 8 (1996), 344–354.

[10] J. Mycielski: Sur le Coloriage des Graphes. Colloquim Mathematiques 3 (1955), 161–162.

[11] M. Pinedo: Scheduling: Theory, Algorithms and Systems. Prenctice Hall, Englewood, NJ (1995).

[12] F. S. Roberts: Applications of edge coverings by cliques. Disc. Appl. Math. 10 (1985), 93–109.

[13] E. C. Sewell: An Improved Algorithm for Exact Graph Coloring. Second DIMACS Challenge (COLORING Papers), in DIMACS Series in Computer Mathematics and Theoretical Computer Science, AMS (1996).

[14] D. B. West: Parameters of partial orders and graphs: packing, covering and represetation. Graph and Orders, Reidel (1985), I.Rival, editor.