# New Lower Bounds on the Weighted Chromatic Number of a Graph

Massimiliano Caramia[1*] and Jiří Fiala[2†]

[1] Dipartimento di Informatica, Sistemi e Produzione
Università di Roma "Tor Vergata"
Via di Tor Vergata, 110 - Roma, Italy.
caramia@disp.uniroma2.it

[2] Inst. of CS & App. Math., C.-A. University, Kiel, Germany
and ITI, Charles University, Prague, Czech Republic[‡]
jfi@informatik.uni-kiel.de

**Abstract**

In this paper we present some algorithmic results for the computation of lower bounds on the chromatic number of a weighted graph. In particular, we develop four algorithms, giving a particular analysis also on triangle free graphs. Experimental results are provided.

**Keywords**: Clique, Coloring, Weight, List.

*Work partially supported by the Italian National Research Council (CNR) and by the Italian Ministry of University and Scientific Research (Project "Algorithms for Large Data Sets: Science and Engineering")

†Research supported in part by EU ARACNE project HPRN-CT-1999-00112

‡Supported by the Ministery of Education of the Czech Republic as project LN00A056

# 1 Introduction and definitions

Let $G$ be an arbitrary undirected graph, with vertex (node) set $V$ and edge set $E$. Given a subset $V' \subseteq V$, $N_{V'} = \{j \in V \setminus V' | (i,j) \in E, i \in V'\}$ is the *neighborhood* of $V'$. A *clique* is a subset $W \subseteq V$ of nodes that induces a complete subgraph in $G$. A *proper $k$-coloring* is a function $f : V \rightarrow \{1, 2, \ldots, k\}$ such that $f(i) \neq f(j)$ for each $(i,j) \in E$, i.e., no two adjacent nodes are assigned the same color. The smallest number $k$ of colors for which a proper $k$-coloring of $G$ exists is called *chromatic number* of $G$ and is denoted by $\chi(G)$.

Assume that in a graph $G$ to each node $i \in V$ is assigned a positive integer weight $p_i$. For any set $V' \subseteq V$ we define $p_{V'} = \sum_{i \in V'} p_i$. A clique $W$ of weight $p_W$ is *maximum* if a clique of $G$ of larger weight does not exist. A proper $k$-coloring in a weighted graph is an assignment to each vertex $i \in V$ of a set of colors $S_i \subseteq \{1, 2, \ldots, k\}$, such that $|S_i| = p_i$ and any adjacent vertices $i, j$ have $S_i \cap S_j = \emptyset$. The minimum number of colors for which there exists a proper $k$-coloring in a weighted graph $G$ is called *weighted chromatic number* of $G$ and is denoted as $\chi_\omega(G)$. If in $G$ all weights are equal to one then the problem reduces to the unweighted case.

The weighted chromatic number problem is a generalization of the well known vertex coloring problem which is $\mathcal{NP}$-hard [7, 8], and for which a number of exact algorithms have been presented in the literature (see, for example: [3, 8, 9, 10, 11, 12]). These algorithms are able to solve problems on randomly generated graphs up to 70 vertices for any density. More recently in [5] an exact algorithm is presented capable of solving random instances up to 100 vertices with any density.

For the minimum *weighted* coloring problem, to the best of our knowledge, only few heuristics have been successfully developed with the goal of finding tighter upper bounds for the minimum weighted clique problem (e.g., see [1, 2]).

We would like to illustrate the behavior of the weighted chromatic number on the following examples:

- We start with a negative result: Descartes [6] proved that for any $k$ there exists a graph $G_k$ of unweighted chromatic number $k$. These graphs demonstrate the fact that the lower bound $p_W$ can be arbitrarily far from the chromatic number $\chi(G_k)$.

- On the other hand observe that for a perfect graph $G$, the weighted chromatic number $\chi_w(G)$ is equal to the weight $p_W$ of a maximum clique. This example shows that there exists a large class of graphs that attain the lower bound.

In this paper we pose the question on how to improve the lower bound given by the maximum weighted clique.

## 2 Lower bounds algorithms: $LB_1$ and $LB_2$

Through the paper $W$ is a maximum weighted clique of $G$ and $N_W$ is its neighborhood. Assign to the nodes $i \in W$ non intersecting lists $L_i$, with $|L_i| = p_i$, such that the highest label assigned is $p_W$. Assign to each node $j \notin W$ a list $L_j = \{1, \ldots, p_W\}$, and for each node $j \in N_W$ delete from $L_j$ colors already assigned to nodes $i \in W$ adjacent to $j$. Denote this algorithm $\mathcal{A}_w(N_W)$.

**Definition 1** *A list $L_j$ in $N_W$ is called minimal if it has been obtained by the algorithm $\mathcal{A}_w(N_W)$.*

**Proposition 1** *Each minimal list $L_j$ has $|L_j| \geq p_j$.*

**Proof:** When $\mathcal{A}_w(N_W)$ stops, each node $j \in N_W$ has:

$$|L_j| = p_W - \sum_{i \in W | (i,j) \in E} p_i.$$

It has to be proved that:

$$|L_j| = p_W - \sum_{i \in W | (i,j) \in E} p_i \geq p_j.$$

By contradiction, if $p_W < \sum_{i \in W:(i,j) \in E} p_i + p_j$, the nodes $\{j\} \cup \{i \in W | (i,j) \in E\}$ would form a clique of weight greater than $p_W$, making $W$ not maximum. $\square$

Let $G_N$ and $G_{NW}$ denote the subgraphs induced by $N_W$ and $W \cup N_W$, respectively, and let $E_{G_N}$ denote the edge set of $G_N$. An enumerative lower bound on $\chi_w(G)$ (different from the maximum weighted clique size) is given by the weighted chromatic number of any node induced subgraph of $G$. If the subgraph considered contains $W$, then this lower bound is certainly not worse than $p_W$. Following this reason and exploiting the chromatic number $\chi_w(G_N)$ we have:

$$\chi_w(G) \geq \chi_w(G_{NW}) \geq p_W + \max\{0, \chi_w(G_N) - | \cup_{j \in N_W} L_j|\} \geq p_W,$$

where $\chi_w(G_N) - | \cup_{j \in N_W} L_j|$ measures the difference between the minimum number of colors required for $G_N$ and the total number of remaining available colors in the lists of the nodes in $G_N$ after the coloring of $W$ has been fixed and $\mathcal{A}_w(N_W)$ has been executed. Given $W$, the computation of the lower bound $LB = p_W + \max\{0, \chi_w(G_N) - | \cup_{j \in N_W} L_j|\}$ remains $\mathcal{NP}$-complete unless $G_N$ admits a polynomial coloring. Hence, to be computationally effective we define a reduced $LB$, denoted as $LB_1$, which involves particular subgraphs of $G_N$ which can be identified and colored in polynomial time.

We start considering as subgraphs of $G_N$ its edges: these are cliques of cardinality two and their weighted chromatic number is equal to the sum of node weights. Thus:

$$LB_1 = p_W + max\{0, max_{\{(i,j) \in E | i,j \in N_W\}}\{p_i + p_j - |L_i \cup L_j|\}\}.$$

More generally, if we are able to find a significant subset of cliques in $G_N$, different from the considered edges, we can compute $LB$ on these subgraphs. Let us find these cliques through the following heuristic algorithm *Find Special Cliques* (FSC) whose complexity is bounded by $O(|V|^3)$:

**Algorithm Finding special cliques (FSC)**
**Input:** `A subgraph` $G_N$
**Output:** `A set of cliques` $K = \{C_1, \ldots, C_l\}$ `and`
`for each clique` $C_s$ `the bound` $lb_s$

***begin algorithm***

**Step 1** $K = \emptyset$;
    $s = 1$;
    $C_s = \emptyset$;
    `order nodes` $j \in N_W$ `in non decreasing values of the ratio` $\frac{p_j}{|L_j|}$
    `and renumber them accordingly;`

**Step 2** `for` $j = 1$ `to` $|N_W|$ `do`

**Step 2.1** $C_s = \{j\}$;

$p_{C_s} = p_j$;

$L_{C_s} = L_j$;

$lb_s = max\{0, p_{C_s} - |L_{C_s}|\}$;

**Step 2.2** `for` $k = j+1$ `to` $|N_W|$ `do`

`if` $C_s \cup \{k\}$ `induces a clique in` $G_N$ `then`

$C_s = C_s \cup \{k\}$;

$p_{C_s} = p_{C_s} + p_k$;

$L_{C_s} = L_{C_s} \cup L_k$;

$lb_s = max\{lb_s, p_{C_s} - |L_{C_s}|\}$;

**Step 2.3** $K = K \cup \{C_s\}$;

$s = s + 1$;

$C_s = \emptyset$;

***end algorithm***

Roughly speaking, FSC tries to find cliques with a weight greater than the total number of colors present in the lists assigned to its nodes. In doing so, it orders them as described in Step 1 and starts to consider those with the highest ratio between $p_j$ and $|L_j|$. Thus, another lower bound can be also defined as:

$$LB_2 = p_W + max_{\{C_s \in K\}}\{lb_{C_s}\}.$$

We have integrated the computation of the bound $lb_{C_s}$ into the algorithm, because the values $\sum_{j \in C_s} p_j - |\cup_{j \in C_s} L_j|$ do not necessarily form a monotone sequence as $C_s$ gains new vertices, e.g., when $|L_k| > p_k$ and $L_k \cap L_{C_s} = \emptyset$.

Observe that the computation of both bounds $LB_1$ and $LB_2$ can be done by a common algorithm running in time $O(|V|^2|E|)$. The new algorithm differs only by the beginning of the Step 2, which is modified as follows:

**Step 2** `for all` $(j', j) \in E(N_W), j' < j$ `do`

**Step 2.1** $C_s = \{j', j\}$;

$p_{C_s} = p_{j'} + p_j$;

$L_{C_s} = L_{j'} \cup L_j$;

$lb_s = max\{0, p_{C_s} - |L_{C_s}|\}$;

`etc.`

### 2.1 A result on triangle free graphs

**Theorem 1** *For a triangle free graph* $\chi_w(G_{NW}) = p_W$.

**Proof:** A maximum weighted clique in a triangle free graph is formed by one edge, say $(i,j)$, whose weight sum is maximum over all the edges. $p_W$ is such a weight sum. Consider now $G_{NW}$ where $W = \{i,j\}$. The easiest way to prove the thesis is to show that $G_{NW}$ is bipartite, i.e., it does not contain odd cycles. It is simple to see that for a given pair of adjacent nodes in $G_N$, say $(h,k)$, $h$ and $k$ must be not adjacent to the same node of $W$ to avoid triangle. Thus, each $(h,k) \in E(G_N)$ will necessarily form an even cycle of length four with $(i,j)$. Moreover, we cannot have odd cycles in $G_N$ for the following reason. Suppose an odd cycle of length $2c+1$ (with $c > 1$) occurs in $G_N$, and let us denote as $\ell_1, \ell_2, \ldots, \ell_{2c+1}$ its nodes, where $(\ell_i, \ell_{i+1}) \in E(G_N)$ for each $i = 1, \ldots, 2c$. This means that $\ell_1$ must be adjacent to $i$, $\ell_2$ to $j$, $\ell_3$ to $i$, and so on until $\ell_{2c}$ which must be adjacent to $j$ (even numbered nodes are adjacent to $j$ and odd numbered ones to $i$). But in doing so, $\ell_{2c+1}$ must be adjacent to $i$ exactly as $\ell_1$ does, and this is not allowable as $(\ell_{2c+1}, \ell_1) \in E_{G_N}$ and thus will form a triangle $\{i, \ell_1, \ell_{2c+1}\}$, which denies the hypothesis that $G$ is triangle free. The reader can prove the same if $\ell_1$ is made adjacent to $j$ instead of $i$. □

By the above theorem the following corollary is straightforward.

**Corollary 1** *If a triangle free graphs is formed by a unique neighborhood of an edge (whatever is that edge) then the graph is bipartite.*

## 3 Lower bounds algorithms: $LB_3$ and $LB_4$

In order to improve the bounds presented and to give a change to augment $p_W$ in triangle free graphs we take a further step by means of the following observation. After the execution of $\mathcal{A}_w(N_W)$, if there exist lists $L_j$ such that $|L_j| = p_j$, the corresponding nodes $j$ cannot assume any other color except those currently in their lists. Thus, these colors can be temporarily fixed and can be deleted from the lists of the nodes $k$ such that $(j,k) \in E$. This can be done repeatedly until there is no node $k \in V \setminus W$ with $p_k = |L_k|$ in the graph.

Let $NW'$ be the set of nodes which have been updated by the described process, but do not belong to $N_W$. Consider in $G$ sets of three nodes where two nodes belong to $N_W$ and the other one to $NW'$. Moreover, consider these two occurrences in the sets selected:

1. The three nodes are mutually adjacent, i.e., they form a clique.
2. The three nodes form a path.

In the first case we can define the following lower bound computed for all the cliques $\bar{C} = \{i, j, k : (i, j) \in E, (i, k) \in E, (j, k) \in E, \{i, k\} \in N_W, j \in NW'\}$:

$$LB_3 = p_W + max\{0, max_{\{i,j,k \in \bar{C}\}} p_i + p_j + p_k - |L_i \cup L_j \cup L_k|\}.$$

In the second case, instead, we define another lower bound computed for all the paths $\bar{P} = \{i, j, k : (i, j) \in E, (j, k) \in E, (i, k) \notin E, \{i, k\} \in N_W, j \in NW'\}$:

$$LB_4 = p_W + max\{0, max_{\{i,j,k \in \bar{P}\}} \lceil \frac{p_i + p_j + p_k - |L_i \cup L_j \cup L_k|}{2} \rceil\}.$$

The motivation for $LB_4$ to have $max_{\{i,j,k \in \bar{P}\}} \lceil \frac{p_i + p_j + p_k - |L_i \cup L_j \cup L_k|}{2} \rceil$ divided by 2 is as follows. Suppose that the term $(p_i + p_j + p_k - |L_i \cup L_j \cup L_k|) = s$ is positive for a path $\bar{P}$ formed by $\{i, j, k\}$. In the best case we can assign the same $\lceil \frac{s}{2} \rceil$ additional colors to the endpoints $i$ and $k$ of $\bar{P}$. Observe that the effective number of additional colors that are necessary to color $\bar{P}$ may vary between $\lceil \frac{s}{2} \rceil$ and $s$.

An example for the calculation of $LB_4$ is given below.

**Example.** Consider the weighted graph in Figure 1 and its maximum weighted clique $W = \{3, 4, 5, 7\}$. After the execution of $\mathcal{A}_w(N_W)$ (see the right-hand side graph for details) node 2 has $p_2 = |L_2|$, thus we have temporarily fixed the colors on its lists and eliminated the latter from $L_1$. The set $NW' = \{1\}$ and we compute $LB_4$. As $N_W = \{6, 2\}$, this involves the exam of only one path, i.e., the one formed by the nodes 6, 1 and 2, which returns that $p_6 + p_1 + p_2 = 7$ while $|L_6 \cup L_1 \cup L_2| = 5$. Thus, while the difference is 2, the clique bound $p_W = 6$ can be raised *only* to 7. Indeed, a feasible coloring with 7-colors is obtained by assigning to the endpoints of the path, i.e., 6 and 2, the same color 7 and then completing the assignment for node 1. For example, suppose that node 6 receives colors $\{3, 7\}$ and node 2 colors $\{1, 7\}$. As can be inferred, the introduction of the new color 7, and

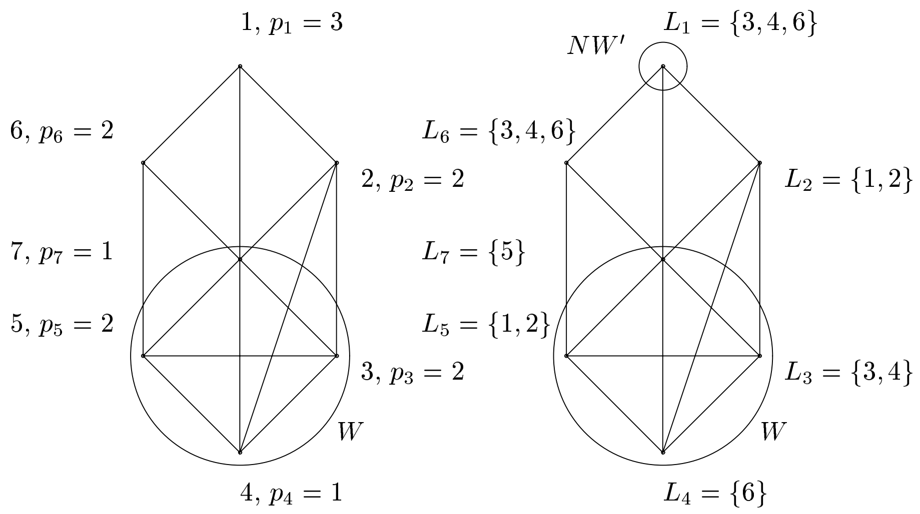


Figure 1: Example of a graph in which $LB_4 > p_W$

its assignment to nodes 2 and 6, gives rise to a new availability for node 1 of two colors that were previously forbidden, e.g., colors 2 and 4. Hence, a possible assignment for node 1 is $\{2, 4, 6\}$. □

From the above expressions, $NLB = \max\{LB_1, LB_2, LB_3, LB_4\}$ is a lower bound on the weighted chromatic number.

### 3.1 Generalization of the algorithm $LB_4$ for triangle free graphs

Let $N_W^2$ be the set of nodes which are adjacent to $N_W$, but do not belong to $W$. $LB_4$ can be generalized for triangle free graphs considering all the paths $\bar{P}' = \{i, j, k : (i, j) \in E, (j, k) \in E, (i, k) \notin E, \{i, k\} \in N_W, j \in N_W^2\}$:

$$LB_4^{tf} = p_W + max\{0, max_{\{i,j,k \in \bar{P}'\}} \lceil \frac{p_i + p_j + p_k - |L_i \cup L_j \cup L_k|}{2} \rceil\}.$$

## 4 Experimental results

In order to deeply analyze our algorithm, we have used two kinds of test problems in our data set: the first one is given by generic random graphs, the second one is given by triangle free random graphs to evaluate the efficacy of $LB_4^{tf}$. A weight function assigns to each node in a graph an integer chosen at random uniformly from the set $\{1, \ldots, mw\}$, where $mw$ is the maximum weight allowable for a node. All algorithms have been coded in C language and run on a Pentium III PC with a 600 MhZ processor and 256 MB RAM. In the columns are listed the following parameters:

| Type | $pw$ $mw = 5$ | $NLB$ $mw = 5$ | $pw$ $mw = 10$ | $NLB$ $mw = 10$ | $pw$ $mw = 20$ | $NLB$ $mw = 20$ |
|---|---|---|---|---|---|---|
| 30.01 | 10.2 | 10.8 | 19.8 | 20.0 | 40.4 | 41.2 |
| 30.03 | 15.4 | 16.0 | 31.0 | 32.2 | 62.8 | 65.6 |
| 30.05 | 24.6 | 25.6 | 45.0 | 47.0 | 90.8 | 95.2 |
| 30.07 | 30.2 | 31.2 | 58.0 | 60.4 | 116.8 | 122.2 |
| 30.09 | 38.6 | 40.2 | 74.0 | 77.6 | 148.8 | 156.4 |
| 50.01 | 11.2 | 11.4 | 20.8 | 21.0 | 42.4 | 43.2 |
| 50.03 | 18.4 | 19.4 | 35.4 | 37.2 | 71.6 | 75.6 |
| 50.05 | 25.8 | 28.2 | 48.2 | 51.2 | 97.2 | 103.6 |
| 50.07 | 34.8 | 37.2 | 64.8 | 67.4 | 130.4 | 136.4 |
| 50.09 | 44.6 | 46.8 | 82.6 | 85.6 | 166.0 | 172.4 |
| 70.01 | 12.4 | 12.8 | 22.8 | 23.2 | 46.4 | 47.6 |
| 70.03 | 21.2 | 22.8 | 40.6 | 42.2 | 82.0 | 85.6 |
| 70.05 | 29.6 | 31.6 | 54.8 | 57.6 | 110.4 | 116.4 |
| 70.07 | 38.8 | 40.0 | 75.4 | 78.2 | 151.6 | 157.6 |
| 70.09 | 47.0 | 49.2 | 92.4 | 95.2 | 185.6 | 191.6 |
| 80.01 | 13.2 | 13.8 | 24.4 | 25.2 | 49.6 | 51.6 |
| 80.03 | 23.4 | 26.4 | 44.0 | 47.4 | 88.8 | 96.0 |
| 80.05 | 35.6 | 37.8 | 67.2 | 70.2 | 135.2 | 141.6 |
| 80.07 | 46.8 | 50.2 | 89.2 | 93.6 | 179.2 | 188.4 |
| 80.09 | 52.0 | 56.2 | 102.8 | 107.0 | 206.4 | 215.2 |
| 90.01 | 14.6 | 15.4 | 26.2 | 27.2 | 53.2 | 55.6 |
| 90.03 | 24.8 | 27.0 | 45.2 | 48.2 | 91.2 | 97.6 |
| 90.05 | 36.0 | 39.4 | 70.8 | 74.4 | 142.4 | 150.0 |
| 90.07 | 50.2 | 53.8 | 96.4 | 100.4 | 193.6 | 202.0 |
| 90.09 | 59.2 | 63.4 | 115.8 | 120.0 | 232.4 | 241.2 |

All statistics are averages of 5 test problems.

Table 1: Results on random graphs

- $p_W$: weight of the maximum clique;
- $NLB$: lower bound values.

We have also generated some triangle free graphs modifying the previous random graphs by means of a procedure which takes in input a graph with edge density equal to 0.5 and eliminates those edges inducing a triangle in the graph.

| Type | $pw$ $mw = 5$ | $LB_4^{tf}$ $mw = 5$ | $pw$ $mw = 10$ | $LB_4^{tf}$ $mw = 10$ | $pw$ $mw = 20$ | $LB_4^{tf}$ $mw = 20$ |
|---|---|---|---|---|---|---|
| tr30 | 10.0 | 11.6 | 20.0 | 23.6 | 40.0 | 46.6 |
| tr50 | 10.0 | 11.8 | 20.0 | 24.0 | 40.0 | 48.0 |
| tr70 | 10.0 | 12.0 | 20.0 | 24.2 | 40.0 | 48.4 |
| tr80 | 10.0 | 12.2 | 20.0 | 25.0 | 40.0 | 50.0 |
| tr90 | 10.0 | 12.6 | 20.0 | 25.4 | 40.0 | 50.8 |

All statistics are averages of 5 test problems.

Table 2: Results on triangle free random graphs

In all our experimentation the time needed to compute the proposed lower bounds, once a maximum weighted clique is obtained, is negligible (at most we observed 2 seconds of CPU time).

# 5 Conclusion

In this paper we have presented new algorithms to compute improved lower bounds on the chromatic number of weighted graphs. The effectiveness of such algorithms have been evaluated by experiments on random instances.

# References

[1] L., Babel, Finding Maximum Cliques in Arbitrary and in Special Graphs, Computing 46 (1991) 321-341.

[2] E., Balas and J., Xue, Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring, Algoritmica (1995) 1-16.

[3] D., Brelaz, New methods to color the vertices of a graph, Communications of the ACM 22 (1979) 251–256.

[4] M., Cangalovic and J., Schreuder, Exact coloring algorithm for weighted graphs applied to timetabling problems with lectures of different lengths, EJOR 51 (1991) 248–258.

[5] Caramia, M., and Dell'Olmo, P., 'Iterative Coloring Extension of a Maximum Clique", to appear in Naval Research Logistics.

[6] B. Descartes, Solution to advanced problem, No 4526, Amer. Math. Monthly 61 (1954) 532.

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

[8] M., Kubale, Introduction to Computational Complexity and Algorithmic Graph Coloring, Wydawnictwo GTN, Gdansk (1998).

[9] M., Kubale and B., Jackowski, A generalized implicit enumeration algorithm for graph coloring, Communications of the ACM 28 (1985) 412–418.

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

[11] T. J., Sager and S., Lin, A Pruning Procedure for Exact Graph Coloring, ORSA J. on Computing 3 (1993) 226–230.

[12] E. C., Sewell, An Improved Algorithm for Exact Graph Coloring, in D. S. Johnson and M. A. Trick, eds., DIMACS Series in Computer Mathematics and Theoretical Computer Science 26 (1996) 359-373.