Available online at www.sciencedirect.com
ScienceDirect
Procedia Social and Behavioral Sciences 00 (2013) 1–10
Procedia Social and Behavioral Sciences
www.elsevier.com/locate/procedia

A mathematical programming approach for the maximum labeled clique problem

Francesco Carrabs[a], Raffaele Cerulli[b], Paolo Dell'Olmo[c]

[a]*Department of Mathematics, University of Salerno, Italy. fcarrabs@unisa.it*
[b]*Department of Mathematics, University of Salerno, Italy. raffaele@unisa.it*
[c]*Department of Statistic Sciences, Sapienza - University of Rome, Italy. paolo.dellolmo@uniroma1.it*

Abstract

This paper addresses a variant of the classical clique problem in which the edges of the graph are labeled. The problem consists of finding a clique as large as possible whose edge set contains at most $b \in Z^+$ different labels. Moreover, in case of more feasible cliques of the same maximum size, we look for the one with the minimum number of labels. We study the time complexity of the problem, also in special cases, and we propose a mathematical programming approach for its solution by introducing two different formulations: the basic and the enforced. We experimentally evaluate the performance of the proposed approach on a set of benchmark instances (DIMACS) suitably adapted to the problem.

Keywords: Clique; Labeled Graph; Colored Graph; Mathematical Models

1. Introduction

The maximum clique problem (MC) is one of the most important combinatorial optimization problem, with application in many real world situations [1]. In this paper we study a variant of the maximum clique problem, namely, the *Maximum Labeled Clique problem* (MLC). Given a graph G, with a label (color) assigned to each edge, we look for a clique of G as large as possible but with the the number of usable labels limited by a fixed constant (budget). Moreover in case of more feasible cliques of the same maximum size we look for the one with the minimum number of different edge labels.

The MLC problem has several applications, among others, in telecommunication and social networks. For example, let us consider a telecommunication network where the connections belong to different companies, each one identified by a different label. Our aim is to localize the maximum number of nodes connected with each other where to place mirroring servers. These servers share the same information and the direct connection with each other guarantees that when a server falls down the others remain synchronized. Since the use of connections have to be paid to the owner company and budget is limited, then the number of different labels in the solution cannot exceed the available budget. Then our problem is to find a maximum clique with the minimum number of labels without exceeding the budget available. As a consequence if the budget is small, depending on the distribution of the labels, the clique found can be small as well. A further application may be found in social network analysis. Examples of application of graph theory to social networks are given for instance in [2] where nodes represent persons and undirected edges represent a generic relationship among individuals (communication, friendship, working cooperation and so on). In

this context, cliques represent groups of individuals mutually connected by the mentioned relation. The label of edges may represent the specific topic or argument which motivated the communication. Hence, a (maximum) clique with a small number of labels corresponds to the largest group (mutually connected) with small heterogeneity on the exchanged subjects. The study of such structures on the web and organizational communities (e.g. large corporate groups) has several motivations from marketing, to security issues like crime prevention and fraud detection, to general business opportunities identification.

The MLC problem belongs to the class of labeled problems that, in the last decade, has received a growing interest probably because these problems may be used to model many real world problems arising, for example, in communication networks, multimodal transportation networks, etc. The Minimum Label Spanning Tree Problem is one of the first problem introduced in this area [3, 4] for which various metaheuristics are proposed [5, 6]. Many other works involving classic combinatorial optimization problems, defined on labeled graphs, include: the Labeled Maximum Matching problem [7, 8, 9], the Minimum Label Path problem [10], the Minimum Labeled Hamiltonian problem [11], the Maximum Labeled Flow problem [12], the Minimum Label Generalized Forest problem [3] and the Minimum Label Steiner Tree Problem [13]. To the best of our knowledge, the MLC problem has not been studied before.

In this paper we deal with the time complexity of MLC problem and we provide two formulations: the Basic and the Enforced. This last formulation is "enforced" by a set of constraints that we derived by various properties of the problem. The performance of the two models and the capability to find the optimal solution, within the time limit, are evaluated on an adapted version of the DIMACS benchmark instances for the maximum clique problem [14].

The remainder of the paper is organized as follows. Section 2 introduces the definitions and the notations that are used throughout the paper. In Section 3 the time complexity of the problem is studied and in Section 4 our models are described. Finally, the computational results are presented in Section 5 and some concluding remarks are given in Section 6.

2. Definitions and notations

Let $G = (V, E, L)$ be a labeled, connected and undirected graph, where V denotes the set of n vertices, E the set of m edges and L the set of labels associated to the edges. In Figure 1a is shown a labeled graph with 7 vertices and 4 labels. Let us define the function $l : E \to L$ that, given an edge (i, j), returns its label and the function $\ell : V \to L$ that, given in input a set of vertices $V' \subseteq V$, returns the labels associated to the edges (i, j) with $i, j \in V'$.

Given a budget $b \leq |L|$, the Maximum Labeled Clique problems (MLC) consists in finding a maximum clique C of G such that $|\ell(C)| \leq b$ (budget constraint), where $b \in Z^+$. In the following, we denote by "*feasible clique*" and "*maximum feasible clique*" any clique of G and a maximum clique of G, respectively, that satisfy all the constraints of the MLC problem. If there are more maximum feasible cliques in G, as secondary goal we want to find the one containing the minimum number of labels. In other words, the *optimal clique* C^* we are looking for is a maximum feasible clique of G with the minimum number of labels.

Let $\delta(v)$ and $\mathcal{N}(v) = \{u \in V : (v, u) \in E\}$ be the *degree* and the *neighborhood* of the vertex v in G, respectively. In Figure 1a the neighborhood of the vertex 2 is $\mathcal{N}(2) = \{1, 3, 4, 5\}$. Given a set of vertices $V' \subseteq V$, we denote by $G[V']$ the subgraph of G induced by V'. Formally: $G[V'] = (V', E[V'], L[V'])$, where $E[V'] = \{(i, j) \in E : i, j \in V'\}$ and $L[V'] = \{l \in L : \exists (i, j) \in E[V'] \text{ with } l(i, j) = l\}$. In Figure 1b, 1c and 1d the subgraphs induced by $V' = \{1, 2, 3, 4, 5\}$, by $V' = \{2, 3, 4, 5\}$ and by $V' = \{4, 5, 6, 7\}$, are shown, respectively. Given a set of labels $L' \subseteq L$ we also define $G[L']$ the subgraph of G induced by the edges whose labels belong to L'. Formally, $G[L'] = (V[L'], E[L'], L')$, where $V[L'] = \{v \in V : \exists (v, j) \in E \text{ with } l(v, j) \in L'\}$ and $E[L'] = \{(i, j) \in E : l(i, j) \in L'\}$. For instance, in Figure 1a the subgraph $G[l_3]$ is the path $< 1, 4, 7, 6, 5 >$ having edges with label l_3.

3. Time complexity of the MLC problem

The MLC is NP-complete because it is a generalization of MC problem that can be seen as a MLC where $|L| = b$. In this section we will show that also on the complete graphs, where MC is trivial, the MLC is NP.

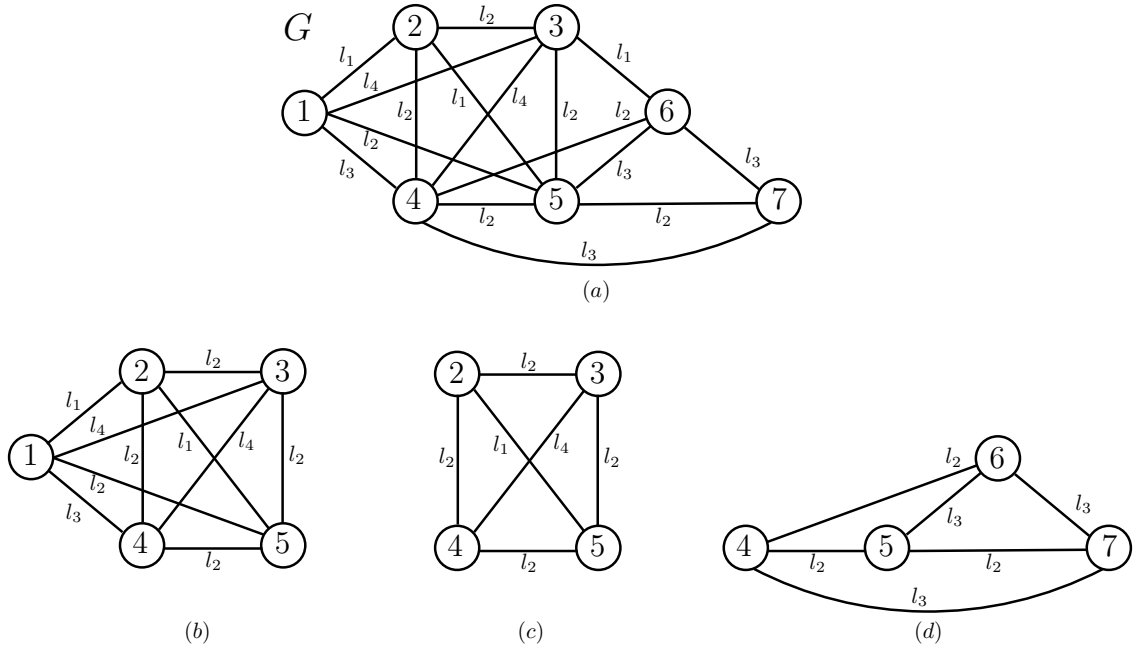

Fig. 1. *a) A labeled graph G with 7 vertices and 4 labels. If the budget b is equal to 3 we have: b) a maximum clique of G that is infeasible for the MLC problem, c) a maximum feasible clique of G, d) the optimal clique C^* of G.*

The MLC problem presents two objectives to pursue that have different priority. The primary objective is to compute the maximum feasible clique of graph. The secondary objective is to find, among all the maximum feasible cliques, the one with the minimum number of labels. This means that a clique with k vertices and any number of labels is better than a clique with $k-1$ vertices and just one label. For instance, let us suppose to solve the MLC problem on the graph G, depicted in Figure 1a, with a budget $b = 3$. It is easy to see that the maximum clique of G is composed by five vertices {1,2,3,4,5} shown in Figure 1b. However, this is not a feasible clique because it is composed by 4 labels $\{l_1, l_2, l_3, l_4\}$ and then it violates the budget constraint. Since there are no other cliques with cardinality 5 in G, we have to look for feasible cliques with cardinality 4. This is an example in which the solution of the MLC problem does not coincide with the solution of the MC problem. In other words, the maximum feasible clique and the maximum clique of G have different cardinality. The clique depicted in Figure 1c is a maximum feasible clique because its cardinality is equal to 4 and it has 3 labels while the clique depicted in Figure 1d is the optimal one because it has only 2 labels (no monochromatic cliques, with cardinality 4, are present in G). This example gives a preliminary insight regarding the structure differences of MLC problem with respect to the well known MC problem. In particular, the exact approaches for the MC problem can generate infeasible solutions for the MLC problem because of the budget constraint. This means that these approaches cannot be directly adopted for the MLC without taking into account the labels and the budget. Furthermore, it is also interesting to notice that the polynomial cases of MC, like the complete graphs, are not necessary polynomial for the MLC because the maximum cliques may require more than b different labels and then they are not feasible cliques for the MLC problem.

In the following we formally prove that MLC is NP-complete on the complete graph, i.e. $G \equiv K_n$, if $b < |L|$. To this end, we define the Maximum Clique Size Problem (MCS), known to be NP-Complete in the strong sense [15], and the decision version (*P1*) of MLC problem as follows:

Definition 3.1 (MCS)**.** *Given a graph $G = (V, E)$ and a positive integer $k \leq |V|$, does the largest complete subgraph in G contain exactly $k > 2$ vertices?*

Definition 3.2 (P1)**.** *Given an edge labeled graph G, with L labels and a fixed budget b does there exist a clique C of G such that* $|C| = k > 2$ *and* $|\ell(C)| \leq b$*?*

Proposition 3.1. *P1 is NP-complete on* K_n *if* $b < |L|$.

Proof. We prove the statement by reducing the MCS problem to the MLC problem. To this end, given the graph $G = (V, E)$ for the MCS problem, we build the complete graph $G_{P1} = (V, E \cup E')$ where $(i, j) \in E'$ iff $(i, j) \notin E$ and $L = \{1, 2, ..., |E'| + 1\}$. Moreover, to each edge $(i, j) \in E$ we assign the label 1, i.e. $l(i, j) = 1$ $\forall (i, j) \in E$, and to each edge $(u, v) \in E'$ we assign a label such that $l(u, v) \in L \setminus \{1\}$ and $l(u, v) \neq l(p, q)$ where $(u, v) \in E'$ and $(p, q) \in E'$. Finally, let $b = 1$.
If exists a solution C of P1 on the graph G_{P1} then to each edge of C is associated the label 1 since $|C| > 2$. This means that, by construction, all the edges of C belongs to E, hence C is a solution also for the MCS problem. If, on the other hand, MCS has a solution on the graph G, then this is a solution on the graph G_{P1} also for the problem P1 where the clique found has all edges labeled with 1. □

As a consequence of the example in Figure 1 and of the proposition 3.1, any optimal algorithm for the MLC problem cannot be based on a (smart) enumeration of all maximum cliques of G and a different search method, that takes into account the constraint on the total number of labels in the solution, has to be devised and the solution has to be found in all, not only maximum, cliques of the graph. We are aware of a number of different exact algorithms to find the maximum clique of the graph that also enumerate all the maximal ones. However, if one is willing to adopt these algorithms, at least in a direct approach he should generate all the $\binom{|L|}{b}$ subgraphs inducted by all subsets of labels of cardinality b and apply the algorithm on these instances. Obviously, such an approach could be pursued only for very small value of $\binom{|L|}{b}$, and this latter observation motivated the study of a new mathematical formulation for general instances of the problem.

4. Mathematical models

In this section the formulations for the MLC problem are introduced. Consider an edge labeled graph $G = (V, E, L)$ as input and having a budget b on the total number of labels. The set of variables used are:

- the binary variable x_i, for each $i \in V$, that assumes value 1 if the vertex i is selected and 0 otherwise;
- the binary variable y_l, for each $l \in L$, that assumes value 1 if there is at least an edge whose label is l into the clique and 0 otherwise.

Moreover, let $a^l_{ij} = 1$ if $l(i, j) = l$. Then, the (IP) formulation of MLC is the following:

$$(IP) \quad \max z_{IP} = |L| \sum_{v \in V} x_v - \sum_{l \in L} y_l \tag{C1}$$

$$\sum_{l \in L} y_l \leq b \tag{C2}$$

$$x_i + x_j \leq 1 \qquad \forall (i, j) \in \bar{E} \tag{C3}$$

$$a^l_{ij}(x_i + x_j) \leq y_l + 1 \qquad \forall (i, j) \in E, l \in L \tag{C4}$$

$$x_i \in \{0, 1\} \qquad \forall i \in V \tag{C5}$$

$$y_l \in \{0, 1\} \qquad \forall l \in L \tag{C6}$$

The objective function (C1) maximize the difference between the number of selected nodes and the number of labels of the clique inducted by these nodes. The first sum is multiplied by $|L|$ so that the primary objective is to maximize the number of selected nodes. In this way, a clique with cardinality k is always preferred to a clique with a smaller cardinality whatever is the number of its labels. The budget constraint (C2) guarantees that the number of labels inside the clique, induced by selected nodes, does not exceed the

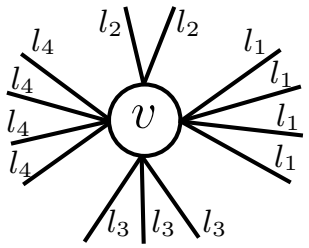

Fig. 2. *The edges connected to the vertex v and their labels.*

budget b. Constraints (C3) ensure that two nodes i and j can be selected only if exists the edge (i, j) in G. Here $\bar{E} = \{(i, j) : (i, j) \notin E\}$. Constraints (C4) ensure that y_l is equal to 1 if there is an edge (i, j), with $l(i, j) = l$, into the clique. Notice that, when a label l is not in the clique, y_l will be equal to zero because, in the objective function, we minimize $\sum_{l \in L} y_l$. Finally, constraints (C5) and (C6) require x_i and y_l to be binary variables.

4.1. Enhanced Model

There are various information, derivable from the problem structure, that can be used to enforce the IP model by adding new constraints. Essentially, this information is derived by the labels incidence to each vertex and by the budget constraint. Before describing the new constraints added, let us introduce the following two functions:

- $\delta(v, k)$: the maximum number of edges connected to v by using k labels.
  This function returns an upper bound to the cardinality of any clique, with at most k labels, that contains the vertex v. For instance, let us consider the vertex shown in Figure 2. If $k = 2$ then $\delta(v, 2) = 8$ because, by using the labels l_1 and l_4, it is possible to select 8 edges connected to v. This means that, by using at most two labels, the vertex v can belong to cliques with at most 9 vertices. Consequently, the value of $\delta(v, b)$ represents an upper bound to the cardinality of any feasible clique of G that contains v.

  Proposition 4.1. *For a given k, the total cost to compute $\delta(v, k)$ is $O(m \log m)$, $\forall v \in V$.*

  Proof. Given a vertex v, in $O(|\delta(v)|)$ we compute the occurrences of each label connected to v. Successively, in $O(|\delta(v)| \log |\delta(v)|)$ we sort, in decreasing order, these occurrences. Finally, we sum the first k occurrences obtaining the value of $\delta(v, k)$. Since the sum of $\delta(v)$ $\forall v \in V$ is $O(m)$, the total cost required is $O(m \log m)$. □

- $\ell(v, k)$: the minimum number of labels necessary to select k edges connected to v.
  This function represents the minimum cost that we have to pay, in terms of labels, to insert the vertex v inside any clique with size $k+1$. For instance, with reference to Figure 2, if $k = 10$ then $\ell(v, 10) = 3$.

  Proposition 4.2. *The total cost to compute $\ell(v, k)$, $\forall v \in V$, is equal to $O(m \log m)$.*

  Proof. Similar to the proof of the proposition 4.1. □

Now we are ready to introduce the additional (enforcing) constraints.

*E*1. Let us compute the value $\delta(v, b)$, for each vertex $v \in V$, and let $ub_e = \max_{v \in V}\{\delta(v, b)\}$. Obviously, $ub_e + 1$ is a weak upper bound on the size of any maximum feasible clique of G. However we can make tighter this bound by noting that, in order to have a clique with cardinality $ub_e + 1$ in G, the following two conditions have to hold:

- there must be at least $ub_e + 1$ vertices in G with $\delta(v, b) \geq ub_e$;
- for each vertex v, with $\delta(v, b) \geq ub_e$, there must be a set of vertices $P \subseteq \mathcal{N}(v)$ such that $|P| \geq ub_e$ and $\delta(w, b) \geq ub_e \ \ \forall w \in P$;

If the previous two conditions hold then it is not possible to improve the upper bound $ub_e + 1$. Otherwise, we decrease the value of ub_e until both the conditions are satisfied. Once the value of ub_e is fixed, we add to the model the new constraint: $\sum_{v \in V} x_v \leq ub_e + 1$. Notice that $\sum_{v \in V} x_v$ is in the objective function and then the solver tends to maximize the number of vertices to select. With the introduction of this constraint we reduce the possible combinations of selected vertices speeding up the computational time of the solver on the model.

Let C_i be a feasible clique of G computed by any metaheuristic. By using the cardinality and the number of labels in C_i we derive the following enforcing constraints.

*E*2. Since C_i is a feasible clique, the value of the objective function on this solution $|L| * |C_i| - |\ell(C_i)|$ represents a lower bound to the value of the optimal solution. Therefore, we introduce the constraint $z_{en} \geq |L| * |C_i| - |\ell(C_i)|$, to the enforced model, where z_{en} is the objective function value of the enforced model. This constraint speeds up the model resolution because it allows to the solver to reduce the size of search tree by applying the pruning operations.

*E*3. For each vertex $v \in V$ we compute $\ell(v, |C_i| - 1)$, i.e. the minimum number of colors that have to be used by v to be inside any clique with $|C_i|$ vertices. Since $|C^*| \geq |C_i|$, if $\ell(v, |C_i| - 1) > b$ then the vertex v cannot belong to C^*, without violating the budget constraint, and consequently x_v should be fixed to zero.

*E*4. The values $\ell(v, |C_i| - 1)$, computed in the constraint *E*3 for any vertex $v \in V$, allow to find a lower bound lb_l of $|\ell(C^*)|$. The idea is to sort, in increasing order, the values $\ell(v, |C_i| - 1)$ and to set lb_l to the $|C_i|^{th}$ value of the list. The new constraint is: $\sum_{l \in L} y_l \geq lb_l$.

*E*5. Since C_i is a feasible clique than $|C^*| \geq |C_i|$. This means that $\sum_{v \in V} x_v \geq |C_i|$. Our computational tests reveal that this constraint speed up the resolution of model (on average there is an increment of 5% on largest instances with lowest budget) but it delays the individuation of a feasible clique because all the feasible cliques with cardinality lower than $|C_i|$ are rejected. Since we prefer to find a feasible solution as many times as possible, within the time limit, we do not use this constraint.

Summarizing, the enforced model is:

$$(EN) \qquad \max z_{en} = |L| \sum_{v \in V} x_v - \sum_{l \in L} y_l \tag{C1}$$

$$\sum_{l \in L} y_l \leq b \tag{C2}$$

$$x_i + x_j \leq 1 \qquad \forall (i, j) \in \bar{E} \tag{C3}$$

$$a^l_{ij}(x_i + x_j) \leq y_l + 1 \qquad \forall (i, j) \in E, l \in L \tag{C4}$$

$$\sum_{i \in V} x_v \leq ub_e + 1 \tag{E1}$$

$$z_{en} \geq |L| * |C_i| - |\ell(C_i)| \tag{E2}$$

$$\sum_{l \in L} y_l \geq lb_l \tag{E4}$$

$$x_i \in \{0, 1\} \qquad \forall i \in V \tag{C5}$$

$$y_l \in \{0, 1\} \qquad \forall l \in L \tag{C6}$$

id	Instance	n	m	|L|	b	Basic			Enhanced			Gap
					75%	$\|C\|$	$\|\ell(C)\|$	$Time$	$\|C\|$	$\|\ell(C)\|$	$Time$	
1	johnson8-2-4_c4	28	210	4	3	4	2	0.03	4	2	0.01	0.00%
2	johnson8-2-4_c8	28	210	8	6	4	2.66	0.03	4	2.66	0.02	0.00%
3	johnson8-2-4_c12	28	210	12	9	4	2.66	0.02	4	2.66	0.02	0.00%
4	MANN_a9_c11	45	918	11	9	13	9	8.79	13	9	6.47	**-26.39%**
5	MANN_a9_c21	45	918	21	16	13.33	15.66	19.04	13.33	15.66	15.55	**-18.33%**
6	MANN_a9_c32	45	918	32	24	14	23	19.02	14	23	13.18	**-30.70%**
7	hamming6-4_c6	64	704	6	5	4	2	0.49	4	2	0.35	0.00%
8	hamming6-4_c11	64	704	11	9	4	2.33	0.55	4	2.33	0.37	0.00%
9	hamming6-4_c17	64	704	17	13	4	3	0.59	4	3	0.44	0.00%
10	hamming6-2_c15	64	1824	15	12	15.33	12	750.66	15.33	12	711.2	**-5.26%**
11	hamming6-2_c29	64	1824	29	22	16	22	2255.13	16	22	2079.5	**-7.79%**
12	hamming6-2_c44	64	1824	44	33	17.33	32.66	2375.7	17.33	32.66	2170.84	**-8.62%**
13	johnson8-4-4_c14	70	1855	14	11	11	10.33	16.46	11	10.33	16.36	0.00%
14	johnson8-4-4_c27	70	1855	27	21	12	20	14.79	12	20	11.8	**-20.22%**
15	johnson8-4-4_c40	70	1855	40	30	13	28.66	3.2	13	28.66	2.39	0.00%
16	johnson16-2-4_c23	120	5460	23	18	8	9	2195.87	8	9	2066.26	**-5.90%**
17	johnson16-2-4_c46	120	5460	46	35	8	12.33	3029.92	8	12.33	3290.52	7.92%
18	johnson16-2-4_c69	120	5460	69	52	8	14.33	4031.29	8	14.33	3909.15	**-3.03%**
19	keller4_c28	171	9435	28	21	11	19.66	3762.87	11	19.66	3304.3	**-12.19%**
20	keller4_c55	171	9435	55	42	11	27	3920.18	11	27	4173.34	6.07%
21	keller4_c83	171	9435	83	63	11	32.33	4681.83	11	32.33	5028.16	6.89%
Avg.												**-9.04%**
Impr.												**10/13**

Table 1. Test results with the budget b equal to the 75% of $|L|$.

5. Computational results

The models were coded in C++ and solved by CPLEX 12 on a 2.33 GHz Intel Core2 processor. In order to have significative test results, we adapted the well knows benchmark instances for the MC problem [14] to the MLC problem by introducing labels randomly. In particular, each instance is characterized by the number of vertices, the number of edges and the number of labels $|L|$. The budget b for our instances is fixed to the 25%, 50% and 75% of $|L|$. To assure that the value of b affects the optimal solution of problem it is necessary to avoid values of $|L|$ too large. For this reason $|L|$ is computed by using the following equation $|L| = n \times d_G \times p$, with $p \in \{0.25, 0.50, 0.75\}$, that takes into account the number of vertices n and the density of graph d_G. In this way, a variation of budget (in the ranges defined above) often implies a variation of cardinality of C^*. The variation of all these parameters (structure of graph, density, number of labels and budget) allow us to verify how they affect the performance of models.

In Table 1 the results of two models, using a budget equal to 75% of $|L|$, are shown. The first six columns list the id (id), the instance name (Instance), number of vertices (n), number of edges (m), number of labels ($|L|$) and the budget (b), respectively. The columns Basic and Enhanced are divided in three subcolumns ($|C|$, $|\ell(C)|$ and $Time$) reporting the cardinality and the number of labels of maximum clique computed by Basic and Enhanced models, respectively, and the CPU time (in seconds) spent. Finally, the last column (Gap) shows the percentage gap between the CPU time of two models. The negative gaps (marked in bold) represent the instances where the Enhanced model is faster than the Basic one. The results reported in each line are the average values over three instances with the same characteristics but a different labels assignment. The second last line of the table reports the average gap computed by using only the values different to 0%. The very last line shows how many times the Enhanced model is faster than the Basic one, on the instances with a gap value different from 0%. From now on, we will refer to these latter instances as the *"significative instances"*. A maximum CPU time of 3 hours was imposed for the solution of each instance. When a model fails to find the optimal solution within this threshold, the marker "*" is reported on the column $|C|$. Moreover, if the model fails to find also a feasible solution within the time limit, the value -1 is reported in the column $|C|$ and $|\ell(C)|$.

The results of Table 1 show that the Enhanced model is faster than Basic one on 10 out of 13 significative

id	Instance	n	m	|L|	b	Basic			Enhanced			Gap
					50%	$\|C\|$	$\|\ell(C)\|$	*Time*	$\|C\|$	$\|\ell(C)\|$	*Time*	
1	johnson8-2-4_c4	28	210	4	2	4	2	0.03	4	2	0.02	0.00%
2	johnson8-2-4_c8	28	210	8	4	4	2.66	0.03	4	2.66	0.01	0.00%
3	johnson8-2-4_c12	28	210	12	6	4	2.66	0.03	4	2.66	0.02	0.00%
4	MANN_a9_c11	45	918	11	6	9	6	17.58	9	6	13.98	**-20.48%**
5	MANN_a9_c21	45	918	21	11	9.66	10.66	67.34	9.66	10.66	46.52	**-30.92%**
6	MANN_a9_c32	45	918	32	16	10	14.33	161.45	10	14.33	108.13	**-33.03%**
7	hamming6-4_c6	64	704	6	3	4	2	0.47	4	2	0.33	0.00%
8	hamming6-4_c11	64	704	11	6	4	2.33	0.53	4	2.33	0.42	0.00%
9	hamming6-4_c17	64	704	17	9	4	3	0.57	4	3	0.43	0.00%
10	hamming6-2_c15	64	1824	15	8	10	8	712.32	10	8	382.73	**-46.27%**
11	hamming6-2_c29	64	1824	29	15	11	14.66	2701.77	11	14.66	2368.78	**-12.32%**
12	hamming6-2_c44	64	1824	44	22	12	21.66	5589.16	12	21.66	4955.96	**-11.33%**
13	johnson8-4-4_c14	70	1855	14	7	8	7	86.76	8	7	85.73	**-1.19%**
14	johnson8-4-4_c27	70	1855	27	14	9	13	112.42	9	13	110.53	**-1.68%**
15	johnson8-4-4_c40	70	1855	40	20	10	19	61.36	10	19	60.79	0.00%
16	johnson16-2-4_c23	120	5460	23	12	8	9	1815.51	8	9	1943.33	6.58%
17	johnson16-2-4_c46	120	5460	46	23	8	12.33	2911.49	8	12.33	3044.75	4.38%
18	johnson16-2-4_c69	120	5460	69	35	8	14.33	3883.6	8	14.33	4227.76	8.14%
19	keller4_c28	171	9435	28	14	9*	13	10813.14	-1	-1	10812.12	0.00%
20	keller4_c55	171	9435	55	28	11	27	4965.46	11	27	4081.97	**-17.79%**
21	keller4_c83	171	9435	83	42	11	32.33	4713.18	11	32.33	4827.99	2.38%
Avg.												**-11.81%**
Impr.												**9/13**

Table 2. Test results with the budget b equal to the 50% of $|L|$.

instances. Moreover, on average, the gap is 9%. Analyzing the gaps in details Enhanced, in the best case, is 30% faster than the Basic (instance n°6) while, in the worst case, it is 8% slower than the Basic (instance n°17). In general, Enhanced is slower than Basic when the effectiveness of added constraints is lower than the overhead payed to solve the problem with a greater and more dense constraints matrix. On the first 15 instances this situation never occurs. In particular, on the instances 1-3 and 7-9, the results are the same because the density of the graph is lower than 0.6 and $|C^*|$ is very small. Instead, on the instance 4-6 and 10-12 the density increases over than 0.9 and Enhanced is always faster than Basic. The defeats of Enhanced occur on the last six instances where the density is at most 0.76 and the ratio $\frac{n}{|C^*|}$ is at least 15. These conditions reduces the effectiveness of added constraints which are based on the cardinality of cliques and the colors connected to the vertices. A more accurate analysis of previous results reveals that the performance of Enhanced is affected by value $\Delta = b - |\ell(C^*)|$. As Δ increases the effectiveness of new constraints decreases reducing, in this way, the performance of Enhanced. For instance, on the three instances (n°17, 20, 21) where Enhanced is slower than Basic, Δ is equal to 23, 15 and 31, respectively.

In Table 2 the results of models with a budget up to 50% are shown. With a budget lower than the one of previous tests, we expect an improvement of performance for the Enhanced model because the new constraints should be more effective. The results reported in the column Gap prove that such a conjecture holds. Indeed, for the instances 1-3 and 7-9 the situation is the same occurred in Table 1 while for the instances 4-6 and 10-12 the gap between the two models grows, on average, from 15% to the 25%. In particular, in the best case, Enhanced is 46% faster than the Basic (instance n°10) while, in the worst case, it is 8% slower than the Basic (instance n°18). On all instances the average gap grows from 9% to 12% while the number of defeats increases by one. The results in Table 2 confirm that the Enhanced model performs unfavorably on the instances with highest values of Δ. A special attention deserves the instance n°19, where both the models did not find the optimal solution within the time limit. On this instance, only the Basic model is able to find a feasible clique. This is because of constraint $E5$ that forces the model to reject all the feasible solutions worst than C_{in}.

Finally, in Table 3 the results with budget equal to 25% are reported. Due to the low value of b and, consequently, to the very low values of Δ, the performance gap between the two models significantly increases. From the last two lines of the table we can see that the average gap grows up to 26% and, on the 11

id	Instance	n	m	|L|	b	Basic			Enhanced			Gap
					25%	$\|C\|$	$\|\ell(C)\|$	*Time*	$\|C\|$	$\|\ell(C)\|$	*Time*	
1	johnson8-2-4_c4	28	210	4	1	3	1	0.04	3	1	0.03	0.00%
2	johnson8-2-4_c8	28	210	8	2	3.33	1.33	0.06	3.33	1.33	0.03	0.00%
3	johnson8-2-4_c12	28	210	12	3	4	2.66	0.03	4	2.66	0.01	0.00%
4	MANN_a9_c11	45	918	11	3	5.66	3	4.74	5.66	3	3.18	**-32.91%**
5	MANN_a9_c21	45	918	21	6	6.33	5	19.76	6.33	5	12.43	**-37.10%**
6	MANN_a9_c32	45	918	32	8	7	8	23.39	7	8	17.61	**-24.71%**
7	hamming6-4_c6	64	704	6	2	4	2	0.46	4	2	0.32	0.00%
8	hamming6-4_c11	64	704	11	3	4	2.33	0.57	4	2.33	0.43	0.00%
9	hamming6-4_c17	64	704	17	5	4	3	0.57	4	3	0.41	0.00%
10	hamming6-2_c15	64	1824	15	4	6	4	67.97	6	4	41.32	**-39.21%**
11	hamming6-2_c29	64	1824	29	8	7.66	7.66	290.14	7.66	7.66	221.77	**-23.56%**
12	hamming6-2_c44	64	1824	44	11	8	10.33	571.22	8	10.33	429.21	**-24.86%**
13	johnson8-4-4_c14	70	1855	14	4	6	4	49.67	6	4	49.82	0.00%
14	johnson8-4-4_c27	70	1855	27	7	6.66	6.33	140.34	6.66	6.33	102.6	**-26.89%**
15	johnson8-4-4_c40	70	1855	40	10	7	9	193.48	7	9	177.13	**-8.45%**
16	johnson16-2-4_c23	120	5460	23	6	6.33	5	5299.68	6.33	5	4248.29	**-19.84%**
17	johnson16-2-4_c46	120	5460	46	12	7,33*	10	10766.68	7,66*	11	5660.29	**-47.43%**
18	johnson16-2-4_c69	120	5460	69	18	8	14.33	4009.22	8	14.33	3699.71	**-7.72%**
19	keller4_c28	171	9435	28	7	6,33*	6	10813.15	-1	-1	10812.01	0.00%
20	keller4_c55	171	9435	55	14	8*	13.66	10813.19	-1	-1	10812.43	0.00%
21	keller4_c83	171	9435	83	21	8,66*	19	10812.78	-1	-1	10812.43	0.00%
Avg.												**-26.61%**
Impr.												**11/11**

Table 3. Test results with the budget b equal to the 25% of $|L|$.

significative instances, the Enhanced model results always faster than the Basic one. On the set of instances 4-6 and 10-12 there is a further improvement of performance from 25% to 30% with respect to the same instances with budget up to 50%. Moreover, a relevant gap is reported, on the instances 13-18, with a peak, equal to 47%, registered on the instance n°17. Even more interesting are the solution values of the two models, on the instances n°17, where both of them do not find the optimal solution. However, the solution value 7,66 produced by Enhanced model is better that 7,33 found by the Basic one and it is computed in half of the time. Regarding the last three instances (n°19-21), the constraint $E5$ prevented the Enhanced model to compute feasible solutions within the time limit.

The trend of results shown by three tables highlights how the value of budget affects the performance of the two models. In particular, as the value of b decreases the complexity of instances increases for both models. However, thanks to the new constraints that are more effective under this condition, the Enhanced model is much faster than the Basic one. Actually, the effectiveness of the new constraints, used into the Enhanced model, is penalized by the kind of instances used because, in these instances, the cardinality of the maximum feasible cliques is often very small with respect to the size of graph. For instance, with a budget equal to 75%, the ratio $\frac{n}{|C^*|}$ is, on average, ~ 9. This situation depends on the structure of original instances in which the maximum clique has again small cardinality, often close to the cardinality of maximum feasible cliques. For this reason, we think that, on random instances with a ratio $\frac{n}{|C^*|}$ lower than 5, it should be more evident the effectiveness of our new constraints that, essentially, are based on the cardinality of feasible cliques.

6. Conclusion

We analyzed an interesting variant of the well know maximum clique problem, namely the Maximum Labeled Clique problem, for which at the best of our knowledge, existing maximum clique algorithms cannot be applied. It is a new problem whose application is related to telecommunication and social networks. We assessed its time complexity and provided two mathematical formulations: a basic and an enhanced one. The basic formulation is derived from the classical model for the standard maximum clique problem. The enhanced formulation was obtained from basic one by using bounding inequalities derived from our

analysis about some property of labeled version of the problem. In order to evaluate the performance of two models, we tested them on the benchmark instances for the classical maximum clique problem adapted to the MLC problem. Our preliminary experimentation shows that the enhanced formulation outperforms the basic formulation in most of cases. However, further tests could be performed on random graphs with greater cliques where the effectiveness of new constraints should be more evident.

References

[1] I. Bomze, M. Budinich, P. Pardalos, M. Pelillo, The maximum clique problem, in: Handbook of Combinatorial Optimization, Kluwer Academic Publishers, 1999, pp. 1–74.
[2] S. Wasserman, K. Faust, Social network analysis: methods and applications, Cambridge University Press, 1999.
[3] H. Broersma, X. Li, Spanning trees with many or few colors in edge-colored graphs, Discussiones Mathematicae Graph Theory 17 (2) (1997) 259–269.
[4] R. Chang, S. Leu, The minimum labeling spanning trees, Information Processing Letters 63 (5) (1997) 277–282.
[5] R. Cerulli, A. Fink, M. Gentili, S. Voß, Metaheuristics comparison for the minimum labelling spanning tree problem, Operations Research / Computer Science Interfaces Series, Springer, 2005, pp. 93–106.
[6] Y. Xiong, B. Golden, E. Wasil, Worst-case behavior of the mvca heuristic for the minimum labeling spanning tree problem, Operations Research Letters 33 (1) (2005) 77 – 80. doi:10.1016/j.orl.2004.03.004.
[7] F. Carrabs, R. Cerulli, M. Gentili, The labeled maximum matching problem, Computers & Operations Research 36 (6) (2009) 1859–1871.
[8] J. Monnot, The labeled perfect matching in bipartite graphs, Information Processing Letter 96 (2005) 81–88.
[9] J. Monnot, On complexity and approximability of the labeled maximum/perfect matching problems, in: ISAAC, 2005, pp. 934–943.
[10] R. Carr, S. Doddi, G. Konjedov, M. Marathe, On the red-blue set cover problem, in: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms, SODA '00, Society for Industrial and Applied Mathematics, 2000, pp. 345–353.
[11] R. Cerulli, P. Dell'Olmo, M. Gentili, A. Raiconi, Heuristic approaches for the minimum labelling hamiltonian cycle problem, Electronic Notes in Discrete Mathematics 25 (2006) 131–138.
[12] R. Cerulli, D. Granata, A. Raiconi, M. Scutellà, Maximum Flow Problems and an NP-complete variant on Edge Labeled Graphs, Handbook of Combinatorial Optimization, Springer, 2013.
[13] R. Cerulli, A. Fink, M. Gentili, S. Voß, Extensions of the minimum labelling spanning tree problem, Journal of Telecommunication and Information Technology 4 (2006) 39–45.
[14] M. Trick, D. Johnson (Eds.), Cliques, Coloring, and Satisfiability: Second Dimacs Implementation Challenge, Vol. 26 of Dimacs Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Boston, MA, USA, 1996.
[15] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, New York, 1979.