Matemática Contemporânea, Vol. 44, 1–10
http://doi.org/10.21711/231766362015/rmc4418

Maximum Clique via MaxSat and Back Again

Alexandre Prusch Züge **Renato Carmo**

Abstract

Branch and bound algorithms stand out as the best performing exact solutions for the Maximum Clique problem. The main recent advances are basically refinements on the same algorithm, where heuristic coloring functions are used for bounding. We look into an attempt to get a tighter bound by applying an heuristic employed in MaxSat solvers. Such heuristic depends on a reduction between the two problems. We show how this heuristic works without resorting to MaxSat terminology, and reformulate it using only graph theoretic concepts.

1 Introduction

Among the exact algorithms proposed for the Maximum Clique problem (MC) in the literature, branch and bound schemes stand out as the best performing from an experimental point of view. Moreover, authors seem to agree in that the best choice for the bounding function in such algorithms is the use of (an upper bound on) the chromatic number of the graph.

2000 AMS Subject Classification: 05C69.
Key Words and Phrases: maximum clique, exact solution, branch and bound.
Partially funded by CAPES – Programa de Demanda Social.

Indeed, it is remarkable that the main recent advances on the subject are essentially variations on the same algorithm, changing only in the proposed way to color the graph (see [CZ12, Pro12] for recent reviews).

This is also the focus in [LQ10b], which proposes the idea of refining the bound given by coloring via a rather unusual reduction to the Partial Maximum Satisfiability problem (PMaxSat). The authors report the experimental performance of their implementation and, based on them, conclude that the approach *"is a very promising research direction"*. Also based on experimental data, they note that, even though the refined encoding allows PMaxSat solvers to perform better than is possible using an encoding previously found in the literature, such solvers are not able to compete with dedicated branch and bound algorithms for MC.

Schematically their idea may be described as follows. Given a graph G and a coloring of G, an instance of PMaxSat is computed. This instance is then given as input to a certain heuristic algorithm that is part of a PMaxSat solver. If this heuristic is able to output an upper bound on the number of satisfiable clauses of the PMaxSat instance, this bound is then translated into an upper bound for the size of the maximum clique in G, which is tighter than the number of colors in the coloring originally given. The resulting algorithm is called the MaxCLQ algorithm.

While the reduction proposed in [LQ10b] is not particularly complicated or unnatural, the fact that the algorithm is described in terms of propositional calculus obscures its graph theoretic meaning. In this work we show that, although the main novelty presented in [LQ10b] is the use of (as the authors put it) "MaxSat technology", its idea can be expressed in pure graph theoretical terms, and that such description has several advantages. It leads to an algorithm which is shorter and simpler to describe and resorts only to usual graph theoretic concepts. Moreover, the resulting algorithm is a natural one in the sense that no artificial constructions or formulations are needed when the workings of the heuristic for PMaxSat are interpreted back in graph theoretic terms. As such, the idea can be used as a starting point for further refinements and, last but not least, the

simplification of the algorithm simplifies its analysis.

The work is organized as follows. In Section 2 we outline the usual branch and bound scheme for MC and how it is improved through an encoding to PMaxSat. We present this improvement using only graph theoretical terms in Section 3 and conclude in Section 4.

1.1 Definitions and Notation

A *graph* G is a pair $(V(G), E(G))$ where $V(G)$ is a finite set of *vertices* and $E(G)$ is a set of (unordered) pairs of vertices, called *edges*. Two vertices u and v are *neighbors* in G if $\{u, v\} \in E(G)$. The *neighborhood* of v in G is the set of its neighbors in G and is denoted $\Gamma_G(v)$. The graph G is *complete* if any two vertices of G are neighbors.

If $X \subseteq V(G)$, then $G[X]$ is the induced subgraph given by $V(G[X]) = X$ and $E(G[X]) = \{\{x, y\} \in E(G) \colon x, y \in X\}$. The set X is a *clique* if the graph $G[X]$ is complete and is *independent* if $G[X]$ has no edges. The maximum size of a clique in G is denoted $\omega(G)$. The Maximum Clique problem (MC) is the problem of finding a clique of maximum size on a given graph.

Given an integer k, a *k–coloring* of a graph G is a surjection $f \colon V(G) \to \{1, \ldots, k\}$ satisfying $f(u) \neq f(v)$ for every $\{u, v\} \in E(G)$. For each $1 \leq i \leq k$, the set $f^{-1}(i) = \{v \in V(G) \mid f(v) = i\}$ is called a *color* so that the value of $f(v)$ is called the color of v and k is called the *number of colors* in the k-coloring f. A *coloring* of G is a k-coloring of G for some k. We note that $\omega(G) \leq k$ for any k-coloring of G.

Let $V = \{x_1, x_2, \ldots, x_n\}$ be a finite set of *variables*. A *literal* on V is a variable x_i or its negation $\overline{x_i}$, for some $x_i \in V$. An *assignment* for V is a set A of literals on V, such that either $x_i \in A$ or $\overline{x_i} \in A$ for each $x_i \in V$. A *clause* on V is a set of literals on V. A clause is *satisfied* by an assignment if (at least) one of its literals is in the assignment. A *formula* on V is a set of clauses on V. The Partial Maximum Satisfiability problem (PMaxSat) is the problem of, given a triple (V, S, H) where V is a set of variables and S and H are formulas on V, finding an assignment for V that satisfies all

clauses in H and the maximum possible number of clauses in S.

2 Maximum Clique via MaxSat ...

The usual branch and bound scheme for solving MC can be stated as follows (for a detailed explanation, see [CZ12], which also includes several specific algorithms).

MaxClique(G, Q, K, C)

Input : a graph G, cliques Q and C in G and a set K of vertices in G.

Output: the clique C or a maximum clique in $G[Q \cup K]$ containing Q, whichever is larger.

if $K = \emptyset$ **then**
  if $|Q| > |C|$ **then**
    $C \leftarrow Q$
else
  if $|Q| +$ upper-bound$(G, K) > |C|$ **then**
    $v \leftarrow$ a vertex from K
    MaxClique$(G, Q \cup \{v\}, K \cap \Gamma_G(v), C)$
    MaxClique$(G, Q, K - \{v\}, C)$
return C

In Algorithm MaxClique(G, Q, K, C), the function upper-bound(G, K) returns an integer b such that $\omega(G[K]) \leq b$. In this way, the execution of MaxClique$(G, \emptyset, V(G), \emptyset)$ returns a maximum clique in G.

Often, the function upper-bound(G, K) computes a coloring of $G[K]$ and returns the number of colors in this coloring. This bound, however, may not be tight for two reasons. First, the coloring may not be optimal, and second, even if it were, the gap between the size of the maximum clique and the number of colors needed to color a graph may be arbitrarily large [GST12].

The coloring bound is refined in [LQ10b] through the resort to a heuristic found in a PMaxSat solver, which is able to detect a set of clauses in S with a particular property. The idea can be summarized as follows.

1. compute a coloring f for $G[K]$;
2. set the upper bound b to the number of colors in this coloring;
3. from G, K and the coloring f, compute an instance (V, S, H) for PMaxSat;
4. use a certain heuristic common in PMaxSat solvers to find a nonempty set $I \subseteq S$ such that not all clauses in I can be simultaneously satisfied while all clauses in H are satisfied;
5. if no such set I is found, return b;
6. otherwise,
   (a) decrease the value of b by 1;
   (b) remove the clauses in this set I from S;
   (c) go back to step (4).

In the next section, we explain all this in graph theoretical terms.

3 . . . and Back Again

Let G be a graph and let f be a coloring of G. We call a set of colors of f *loose* if there is no clique in G with one vertex from each color from the set. As an example, any set of three colors in a coloring of a circuit on 5 vertices is a loose set. We define a *loose family* as a family of mutually disjoint loose sets of colors of f. The idea for the upper bound in Algorithm MaxClique is straightforward from the following.

Theorem 1. Let G be a graph and let f be a k–coloring of G.

1. If L is a loose set of colors of f, then $\omega(G) \leq k - 1$, and
2. if S is a loose family, then $\omega(G) \leq k - |\mathsf{S}|$.

upper-bound(G, K)

Input : a graph G and a set K of vertices in G.
Output: an upper bound on $\omega(G[K])$.
$f \leftarrow$ a coloring of $G[K]$
$\mathcal{C} \leftarrow$ the set of colors in f
$k \leftarrow |\mathcal{C}|$
return $k -$ size-of-loose-family(G[K],$\mathcal{C}$)

The Algorithm size-of-loose-family$(G, \mathcal{C})$ finds and removes from the set $\mathcal{C}$ one loose set of colors at a time in a greedy fashion. Note that, as every time a loose set of colors is found it is removed from $\mathcal{C}$, it follows that all loose sets found in this way are mutually disjoint. This is an important point because otherwise the bound could be decreased erroneously. For example, this would happen if a loose set is found and this set is actually a superset of another loose set previously found (any superset of a loose set is also a loose set).

size-of-loose-family$(G, \mathcal{C})$

Input : a graph G and the set $\mathcal{C}$ of colors of a coloring of G.
Output: the size of a loose family.
mark each color in $\mathcal{C}$ as non–tested
$s \leftarrow 0$
while $\mathcal{C}$ *contains a non–tested color* **do**
  $X \leftarrow$ a non–tested color in $\mathcal{C}$ of minimum size
  mark X as tested
  $L \leftarrow$ loose-set$(G, \mathcal{C}, X)$
  if $L \neq \emptyset$ **then**
    $\mathcal{C} \leftarrow \mathcal{C} - L$
    $s \leftarrow s + 1$
return s

Algorithm loose-set$(G, \mathcal{C}, X)$ returns a loose set in $\mathcal{C}$ containing the color X. Note that deciding if the set of all colors in a k-coloring of a graph G is loose is the same as deciding if there is a clique of size k in G and, therefore, is an NP-complete problem.

The following heuristic is used: starting from some vertex v, greedily look for a clique using one vertex from each color without choosing vertices from colors of size two or more. The actual clique is not stored, it is enough to determine if such clique can be found.

Now suppose such heuristic was not able to produce a clique. This fact alone is not sufficient to find a loose set of colors, mainly because it fixes a certain vertex v. So, the process is repeated for all vertices v in the color X. Then, if no clique was found for any v in X, the color X along with the colors of all vertices that were greedily chosen and the colors where no vertex could be chosen form a loose set of colors. This process is described in the Algorithm loose-set$(G, \mathcal{C}, X)$.

---

loose-set$(G, \mathcal{C}, X)$

Input : a graph G, a set of colors $\mathcal{C}$ and a color $X \in \mathcal{C}$.
Output: a loose set of colors in $\mathcal{C}$ containing X, or $\emptyset$ if none was found.

$L \leftarrow \{X\}$
for each $v \in X$ **do**
  $T \leftarrow \mathcal{C}$
  remove X from T
  remove every $u \notin \Gamma_G(v)$ from each color in T
  while *there is no empty color and*
      there is an unitary color $Y = \{w\}$ *in* T **do**
    remove Y from T
    remove every $u \notin \Gamma_G(w)$ from each color in T
  if $T = \emptyset$ **then**
    return $\emptyset$
  else
    add to L the colors in $\mathcal{C}$ that became unitary or empty in T
return L

---

4 Conclusion and Future Work

Branch and bound algorithms stand out as the best performing algorithms in the literature for MC from an experimental point of view. Several algorithms rely on some heuristic coloring as the bounding function. The MaxCLQ algorithm is no exception, however its main novelty is applying a heuristic taken from a PMaxSat solver to get a tighter bound. In this work we converted this novelty back to pure graph theoretic terms, resulting in a natural heuristic that can be used directly in any branch and bound algorithm for MC.

As future work, this heuristic can be inserted in other algorithms, quite directly in those that already use coloring for bounding. Also, there are other works applying MaxSat techniques for MC [LQ10a, SSTL13] that may have natural interpretations on graph theory, so similar studies like the present one may be conducted.

References

[CZ12] Renato Carmo and Alexandre Züge, *Branch and bound algorithms for the maximum clique problem under a unified framework*, Journal of the Brazilian Computer Society **18** (2012), no. 2, 137–151.

[GST12] András Gyárfás, András Sebö, and Nicolas Trotignon, *The chromatic gap and its extremes*, Journal of Combinatorial Theory, Series B **102** (2012), no. 5, 1155–1178.

[LQ10a] Chu-Min Li and Zhe Quan, *Combining graph structure exploitation and propositional reasoning for the maximum clique problem*, 2010 22nd IEEE International Conference on Tools with Artificial Intelligence, 1:344–351, IEEE, 2010.

[LQ10b] _____, *An efficient branch-and-bound algorithm based on maxsat for the maximum clique problem*, Twenty-Fourth AAAI Conference on Artificial Intelligence, 2010.

[Pro12] Patrick Prosser, *Exact algorithms for maximum clique: A computational study*, Algorithms **5** (2012), no. 4, 545–587.

[SSTL13] Pablo San Segundo, Cristobal Tapia, and Alvaro Lopez, *Watching subgraphs to improve efficiency in maximum clique search*, Contemporary Challenges and Solutions in Applied Artificial Intelligence (Moonis Ali, Tibor Bosse, Koen V. Hindriks, Mark Hoogendoorn, Catholijn M. Jonker, and Jan Treur, eds.), Studies in Computational Intelligence, vol. 489, Springer, 2013, pp. 115–122.

Alexandre Prusch Züge
Departamento de Informática
Universidade Federal do Paraná
Brazil
alexandrezuge@gmail.com

Renato Carmo
Departamento de Informática
Universidade Federal do Paraná
Brazil
renato.carmo.rc@gmail.com