Computers & Operations Research 44 (2014) 185–192

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Relaxed approximate coloring in exact maximum clique search

Pablo San Segundo*, Cristobal Tapia

Centre of Automation and Robotics (UPM-CSIC), C/Jose Gutiérrez Abascal, 2, 28006 Madrid, Spain

A R T I C L E   I N F O

Available online 13 November 2013

Keywords:
Combinatorial optimization
Approximate coloring
Branch-and-bound
Global search

A B S T R A C T

This paper presents selective coloring as a new paradigm for branch-and-bound exact maximum clique search. Approximate coloring has, in recent, years been at the heart of leading solvers in the field. Selective coloring proposes to relax coloring up to a certain threshold. The result is a less informed but lighter decision heuristic.

Different operators for the remaining uncolored vertices give rise to algorithmic variants integrated in a new BBMCL framework. BBMCL allows for an interesting comparison between approximate coloring and degree-based decision heuristics.

The paper also reports extensive empirical tests. Selective coloring algorithms are fastest for a large subset of the graphs considered.

1. Introduction

A complete graph, or clique, is a graph such that all its vertices are pairwise adjacent. For a given graph, determining whether a hidden clique of a fixed size k exists is a well known and deeply studied NP-complete problem referred to as *k-clique* [1]. The corresponding optimization problem is the *maximum clique problem* (MCP) which looks for the largest possible clique. Besides its theoretical relevance as an NP-hard problem, MCP is known to have many real applications in a wide scope of fields as in bioinformatics and computational biology [2], computer vision [3], robotics [4,5] etc.

1.1. Definitions and notation

A simple undirected graph $G=(V,E)$ consists of a finite set of vertices $V=\{v_1,v_2,\cdots,v_n\}$ and edges $E\subseteq VxV$ which pair distinct vertices. Two vertices are said to be adjacent (alias neighbors) if they are connected by an edge. For any vertex $v\in V$, $N_G(v)$ (or simply $N(v)$ when the graph is clear from the context) refers to the neighbor set of v in G. Any subset of vertices $U\subseteq V$ *induces* (more precisely *vertex-induces*) a new subgraph $G'=G[U]$ with vertex set U and edge-set $E'\subseteq E$ such that both endpoints of any edge in E' are in U.

The paper employs standard notation for valencies in undirected graphs: $\deg(v)$ for vertex degree (the cardinality of $N(v)$) and Δ for graph degree (the maximum degree of any of its vertices). Also standard notation $\omega(G)$ refers to the order of a maximum clique in G.

Vertex coloring is another well known NP-complete problem very much related to maximum clique. In its existential formulation, its goal is to find whether a proper label (color) assignment $c(v):V\to\mathbb{N}$ of range k exists (denoted as *k-coloring*), such that adjacent vertices all have different labels (i.e. $v_2\in N(v_1)\Rightarrow c(v_2)\neq c(v_1)$). In the paper, $C(G)=\{C_1,C_2,\cdots,C_k\}$ refers to any (proper) k-coloring in G. $C(G)$ partitions V in k disjoint independent *color sets* C_i, one for each color label (i.e. $v\in C_i\Leftrightarrow c(v)=i$). $|C(G)|$ denotes the *size* of the coloring, the number of different colors used.

Trivial upper bounds for $\omega(G)$ based on structure are $|V(G)|$ or $\Delta+1$; but they are not tight. A well known tighter bound is the size of any proper coloring:

$$|C(G)|\geq\omega(G) \tag{1}$$

since only one vertex from each color class can make part of a clique. This important property is commonly used in branch-and-bound MCP algorithms since efficient approximate coloring heuristics are known.

We represent vertex colorings in graph drawings using colors. In particular, green is used to refer to a maximum clique or, vertices belonging to C_1 depending on the context; dark blue and yellow refer to the second and third color labels respectively. There is no particular criterion for higher color assignments. Additionally, vertices in graph drawings are all numbered according to their initial order in the graph.

1.2. Reference MCP algorithm

Listing 1 describes the outline of REFCLIQUE, an efficient approximate color branch-and-bound MCP solver which corresponds roughly to Tomita and Seki's MCQ [6]. REFCLIQUE enumerates cliques in S constructively, starting from a single vertex and enlarging the

* Corresponding author. Tel.: +34 91 7454660, +34 91 3363061; fax: +34 91 3363010.
E-mail address: pablo.sansegundo@upm.es (P. San Segundo).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.10.018

current clique by one at each step of the search. A typical brute force enumeration picks vertices one at a time from the induced subgraph $G[U]$ at the current step. This leads to a binomial search tree where, at level k, all possible different cliques of size k are considered (steps 1–3, 5–6, 13–15). Once a leaf node is reached, the current maximal clique in S is stored in S_{max} if it improves the best solution found so far (step 8); the algorithm then backtracks to the previous node and continues enumeration.

Listing 1. Reference maximum clique algorithm

```
Input: A simple graph $G=(V,E)$ sorted using *smallest-degree-last*
Output: A maximum clique in $S_{max}$
REFCLIQUE $(U, C, S, S_{max})$
Initial step: $U \leftarrow V,\ c(v_i) \leftarrow \min\{i, \Delta G\},\ S \leftarrow \phi,\ S_{max} \leftarrow \phi$

1.  **repeat until** $U=\phi$
2.      select a vertex $v \in U$ with maximum color label in $C$
3.      $U \leftarrow U \setminus \{v\}$
4.      **if** $(|S|+c(v) \le |S_{max}|)$ **return**        //pruning step
5.      $S \leftarrow S \cup \{v\}$
6.      $U_v \leftarrow U \cap N_U(v)$
7.      **if** $(U_v=\phi)$ **then**
8.      **if**$(|S|>|S_{max}|)$ **then** $S_{max} \leftarrow S$
                //stores current champion at leaf node
9.          **return**
10.     **endif**
11.     REFCOLOR$(G[U_v], C_v)$        //colors child subgraph
12.     REFCLIQUE $(U_v, C_v, S, S_{max})$
13.     $S \leftarrow S \setminus \{v\}$
14. **endrepeat**
```

A more efficient strategy is to prune the search when the current growing clique cannot possibly unseat the current champion. A lower threshold for improvement at each step can trivially be established as the difference between the sizes of the best solution found so far $|S_{max}|$ and of the current growing clique $|S|$. If an upper bound on $\omega(G[U])$ is not greater than this threshold (denoted as k_{min} by Konc and Janečič in [7]) there is no point in continuing the search along this branch any further and the search space is pruned (step 4). Modern efficient MCP solvers use sequential approximate vertex coloring (SEQ) to compute a reasonably tight bound for $\omega(G[U])$ based on property (1). REFCOLOR (step 11) refers to a typical implementation similar to the one described in [6]. We will come back to this procedure throughout the paper.

At the beginning of the search, vertices are sorted by degree and picked by increasing degree, a *most-constrained* heuristic known to produce smaller search trees on average. In practice, *smallest-degree-last* sorting is used (*smallest-last* was the actual name given in Matula and Beck [8] in the context of vertex coloring; a Java implementation can be found in Listing 7 of [9]) so that candidate vertices are actually selected in reverse order at the root node. An important implementation detail is to sort child node vertices U_v by increasing color label on output of REFCOLOR; with this improvement, step 2 of RECLIQUE is reduced to selecting vertices in reverse order from U_v (maximum color first).

1.3. Sequential vertex coloring heuristics

At the heart of REFCOLOR is SEQ, a well known and deeply studied coloring heuristic; it computes a proper coloring for any given graph by picking vertices one at a time in strict order and assigning them the lowest possible color (considering previous labels). SEQ runs in worst time $O(|V|^2)$.

BBMC algorithm [10,11], initially named BB-MaxClique, combines a bit encoding with *class coloring*, an alternative implementation of SEQ. In class coloring, each color class is computed in full before starting with the next label, as opposed to coloring each vertex sequentially in traditional SEQ. The benefit is that vertices may be sorted by increasing color on the fly as required by the main MCP procedure. The disadvantage is that it operates with the full set of unlabeled vertices at the start of a new (color class) iteration.

This overhead is shown to be reduced by a careful encoding of vertex sets as bit strings and critical neighbor set computations as bitmasks; results for BBMC reported best times in literature for a number of structured and random graphs. A similar conclusion was reached by Prosser in a recent survey of existing MCP algorithms [9].

1.4. Recent improvements

One important breakthrough described in BBMC and MCS [13] independently, is to keep the initial sorting of vertices at the root node *fixed throughout the search*. It has been shown that SEQ averages tighter colorings in deeper levels of the tree when input set U_v preserves initial order, as opposed to full color sorting in MCQ or Konc & Janečič's partial color sorting improvement [7].

Another important recent idea is *recoloring*, implemented in MCS. Intuitively, recoloring is a second round coloring which attempts to reduce the number of colors obtained by SEQ. Thus, if some conditions are met, a vertex which would be assigned a label l_1 by SEQ is now reassigned a label $l_2 < l_1$. The downside is the overhead involved in certifying these conditions. In [11] recoloring was shown to be efficient only in the case of dense graphs (i.e. edge probability $p \ge 0.8$).

Finally it is worth pointing out the use of a MaxSAT encoding to compute tight upper bounds, as proposed by Li and Quan in [12]. The relevance of the result is that given a concrete subproblem graph, the bound obtained may be tighter than its chromatic number. Tests carried out over uniform random graphs, showed the proposed new algorithm MaxCLQ performing better than leading BBMC for very dense graphs (i.e. $p \ge 0.9$), but underperforming badly in the large, sparse algorithms.

1.5. Motivation

A well known reason for the success of approximate coloring in MCP is that while REFCOLOR is called just once at each node, every color assignment $c(v_k \in U)$ may be used in that same node as an upper bound on $\omega(G[W])$, $W \subseteq U = \{v_1, v_2, \ldots, v_{k-1}\}$, as long as $c(v \in W) \le c(v_k)$. Note that by choosing candidate vertices with maximum color label in step 2, REFCLIQUE enforces this property for every candidate vertex.

Moreover, since parameter k_{min} (lower threshold color label for a possible clique improvement in the child node) can be computed prior to coloring, a similar pruning effect may be obtained by removing from the output set C_v of REFCOLOR all C_i subsets such that $i < k_{min}$; vertices assigned to these sets will therefore not be eligible at step 2 in the child node and thus implicitly pruned.

A number of questions come naturally from the previous analysis. As only vertices with color assignments below k_{min} will be pruned, is it really necessary to compute the full SEQ coloring? Intuitively, expanding vertices sorted by color captures structure better, but is the overhead worthwhile?

It is important to note that in classical SEQ it is not possible to determine *a priori* if an uncolored vertex v will be assigned a color $c(v) \ge k_{min}$. However *class coloring* encodes this knowledge implicitly: these will be all remaining unlabeled vertices after $C_{k_{min}-1}$ has been obtained.