Benchmark Problems for Exhaustive Exact Maximum Clique Search Algorithms

Sándor Szabó
Institute of Mathematics and Informatics, University of Pécs
Ifjuság utja 6, H-7624 Pécs, Hungary
E-mail: sszabo7@hotmail.com

Bogdán Zaválnij
Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences
Reáltanoda u. 13–15, H-1053 Budapest, Hungary
E-mail: bogdan@renyi.hu

Keywords: clique, maximal clique, maximum clique, graph coloring, random graph, Mycielski graph

Received: February 12, 2019

There are well established widely used benchmark tests to assess the performance of practical exact clique search algorithms. In this paper a family of further benchmark problems is proposed mainly to test exhaustive clique search procedures.

Povzetek: Podanih je nekaj novih standardnih problemov za testiranje algoritmov za iskanje klik.

1 Preliminaries

Let $G = (V, E)$ be a finite simple graph. Here V is the set of vertices of G and E is the set of edges of G. The finiteness of G means that G has finitely many nodes and finitely many vertices, that is, $|V| < \infty$, $|E| < \infty$. The simplicity of G means that G does not have any loop and it does not have double edges.

Let C be a subset of V. If two distinct nodes in C are always adjacent in G, then C is called a clique in G. When C has k elements, then we talk about a k-clique. We include the cases $k = 0$ and $k = 1$ as well when $|C| = 0$ and $|C| = 1$, respectively. Though in these cases C does not have two distinct elements.

A clique is maximal in G if it is not part of any larger clique in G. In other words a clique is maximal clique of the graph G if it cannot be extended to a larger clique by adding a new node of the graph G. A k-clique is a maximum clique in G if G does not contain any $(k + 1)$-clique. All the maximum cliques of a graph G have the same number of nodes. We call this well defined number the clique number of G and we denote it by $\omega(G)$.

A number of problems is referred as clique search problems [1].

Problem 1. *Given a finite simple graph G. List all maximal cliques of G without repetition.*

Problem 2. *Given a finite simple graph G and given a positive integer k. Decide if G has a k-clique.*

Problem 3. *Given a finite simple graph G. Determine $\omega(G)$.*

Problem 4. *Given a finite simple graph G. List all maximum cliques of G without repetition.*

An algorithm to solve Problem 1 can be found in [2]. The algorithm is commonly known as Bron-Kerbosch algorithm. Obviously, the Bron-Kerbosch algorithm can be used to solve Problems 2, 3 and 4. A more efficient algorithm to solve these problems was first given in [3]. The algorithm is known under the name Caraghan-Pardalos algorithm. The Bron-Kerbosch and Carraghan-Pardalos algorithms are the classical algorithms that form the base of many further clique search procedures. These algorithms are presented in [14], [26], [24], [10], [9]. But this list is not intended to be complete.

The complexity theory of the algorithm tells us that Problem 2 is in the NP-complete complexity class. (See for instance [6].) Consequently, Problem 3 must be NP-hard.

We color the vertices of G such that the following conditions are satisfied.

(1) Each vertex of G is colored with exactly one color.

(2) Vertices of G connected by an edge receive distinct colors.

A coloring of the nodes of G that satisfies conditions (1), (2) is called a legal coloring or well coloring of the nodes of G.

Suppose that the nodes of G can be colored legally using k colors. We may use a map $f : V \to \{1, \ldots, k\}$ to describe a coloring of the nodes of G. The numbers $1, \ldots, k$ play the roles of the colors and $f(v)$ is the color of the vertex v for each $v \in V$. If for adjacent nodes u and v of G

the equation $f(u) = f(v)$ implies $u = v$, then the coloring defined by the map f is a legal coloring.

There is a number of the colors k such that the nodes of G can be colored legally using k colors and the nodes of G cannot be colored legally using $k-1$ colors. This well defined number k is called the chromatic number of the graph G and it is denoted by $\chi(G)$.

Graph coloring is a vast subject and we cannot make justice to this venerable field. In this paper we take a very narrow view. We are interested in only one particular application. Note that $\omega(G) \leq \chi(G)$ holds for each finite simple graph G and so coloring of the nodes can be used to estimate the size of a maximum clique. However, the gap between $\omega(G)$ and $\chi(G)$ can be arbitrarily large. J. Mycielski [13] exhibited a graph $M^{(k)}$ for which $\omega(M^{(k)}) = 2$ and $\chi(M^{(k)}) = k$ for each integer $k \geq 2$.

In order to find bounds for $\omega(G)$ the following node coloring was proposed in [21]. Let us choose an integer $s \geq 2$ and consider a coloring of the nodes of G that satisfies the following conditions.

(1′) Each vertex of G is colored with exactly one color.

(2′) If the vertices $v_1, \ldots, v_s$ are vertices of a clique in G, then all the vertices $v_1, \ldots, v_s$ cannot receive the same color.

A coloring of the nodes of G satisfying the conditions (1′), (2′), is called a monochrome s-clique free coloring. In short we will talk about s-clique free coloring. For $s = 2$ the monochrome s-clique free coloring of the nodes gives back the legal coloring of the nodes. There is a well defined minimum number k such that the nodes of G have an s-clique free coloring using k colors. This k is referred to as the the s-clique free chromatic number of G and it is denoted by $\chi^{(s)}(G)$. The inequality $\omega(G) \leq (s-1)\chi^{(s)}(G)$ shows that s-clique free coloring of the nodes can be used to establish upper bound for the clique number.

A number of problems is considered in connection with coloring the nodes of a graph customarily.

Problem 5. *Given a finite simple graph G and given a positive integer k. Decide whether the nodes of G admit a legal coloring using k colors.*

Problem 6. *Given a finite simple graph G. Determine $\chi(G)$.*

It is a well known result of the complexity theory of algorithms that Problem 5 belongs to the P complexity class for $k = 2$ and it belongs to the NP-complete complexity class for $k \geq 3$. (See for example [15].) It follows that for $k \geq 3$ Problem 6 is NP-hard.

Problem 7. *Given a finite simple graph G and two positive integers s, k. Decide if the nodes of G have a legal s-clique free coloring with k colors.*

It was established in [23] that for $k = 3$, $s \geq 3$ Problem 7 is NP-complete.

2 A Mycielski type result

As we have already mentioned the chromatic number can be a poor upper estimate of the clique number. By Mycielski's construction there are 3-clique free graphs with arbitrarily large chromatic number. P. Erdős [5] generalized this result. Let us call the length of a shortest cordless circle in a graph the girth of the graph. Clearly, the girth of a 3-clique free graph must be at least 4. Erdős has proved that for given positive integers k and l, there is a finite simple graph G with $\mathrm{girth}(G) \geq l$, $\chi(G) \geq k$. Erdős's proof is not constructive and so it is not at all straight forward how the resulting graphs could be used in constructing test instances.

In this section we present another extension of Mycielski's result. We replace the legal coloring of the nodes of a graph by a legal s-clique free coloring of the nodes of the graph. Consequently, the s-clique free chromatic number $\chi^{(s)}(G)$ will play the role of the chromatic number $\chi(G)$.

The result is motivated by the fact that one might try to construct clique search test instances by replacing the Mycielski graph by the graph emerging from the proof the generalized version.

Theorem 1. *Let us choose two positive integers s and k with $s \geq 3$ and $k \geq 2^{(s-1)}/(s-1)$. There is a finite simple graph $L^{(s,k)}$ such that $\omega(L^{(s,k)}) \leq 2^{(s-1)}$ and $\chi^{(s)}(L^{(s,k)}) \geq k$.*

The reader may notice that the graph $L^{(2,k)}$ is isomorphic to the Mycielski graph $M^{(k)}$.

Proof. The proof will be constructive. We start with the special case $s = 3$. We choose an integer k for which $k \geq 2^{(3-1)}/(3-1) = 2$. Let $M^{(k)}$ be the Mycielski graph with parameter k. Let $u_1, \ldots, u_n$ be the nodes of $M^{(k)}$. We substitute the node u_i of $M^{(k)}$ by an isomorphic copy $M_i^{(k)}$ of the Mycielski graph for each i, $1 \leq i \leq n$. Let $v_{i,1}, \ldots, v_{i,n}$ be the nodes of $M_i^{(k)}$. We assume that the nodes

$$v_{1,1}, \ldots, v_{1,n}, \ldots, v_{n,1}, \ldots, v_{n,n}$$

are pair-wise distinct. These nodes will be the nodes of the graph $L^{(3,k)}$.

The edges of $M_i^{(k)}$ are going to be edges of $L^{(3,k)}$ for each i, $1 \leq i \leq n$. Further, whenever the unordered pair $\{u_i, u_j\}$ is an edge of $M^{(k)}$, then we add the edge $\{v_{i,\alpha}, v_{j,\beta}\}$ to $L^{(3,k)}$ for each α, β, $1 \leq \alpha, \beta \leq n$.

The dedicated reader will not fail to notice that the construction we just presented is the so called lexicographic product of the graphs $M^{(k)}$ and $M^{(k)}$.

We claim that $\omega(L^{(3,k)}) \leq 4$.

In order to verify the claim we assume on the contrary that $\omega(L^{(3,k)}) \geq 5$. Let C be a 5-clique in $L^{(3,k)}$. Set $V_i = \{v_{i,1,}, \ldots, v_{i,n}\}$. Note that the set V_i induces $M_i^{(k)}$ in $L^{(3,k)}$ as a subgraph of $L^{(3,k)}$. From the fact that $\omega(M_i^{(k)}) \leq 2$ it follows that C may have at most 2 nodes in V_i for each i, $1 \leq i \leq n$. Therefore C has nodes in some $M_i^{(k)}$ for at least 3 distinct values of i.

Suppose that i and j are distinct numbers in the set $\{1,\ldots,n\}$. A node in $M_i^{(k)}$ and a node in $M_j^{(k)}$ can be adjacent only if the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$. This means that $M^{(k)}$ contains a 3-clique. But $M^{(k)}$ does not contain any 3-clique. This contradiction completes the verification of the claim.

We claim that $\chi^{(3)}(L^{(3,k)}) \geq k$.

In order to prove the claim let us assume on the contrary that $\chi^{(3)}(L^{(3,k)}) \leq k-1$. Set

$$\begin{aligned} V &= V_1 \cup \cdots \cup V_n \\ &= \{v_{1,1},\ldots,v_{1,n},\ldots,v_{n,1},\ldots,v_{n,n}\}. \end{aligned}$$

Suppose that the map $f : V \to \{1,\ldots,k-1\}$ defines a 3-clique free coloring of the nodes of $L^{(3,k)}$.

The restriction of f to the set V_i is a map $g_i : V_i \to \{1,\ldots,k-1\}$. Clearly, the map g_i defines a coloring of the nodes of the graph $M_i^{(k)}$. From the fact that $\chi(M_i^{(k)}) \geq k$ it follows that there are two distinct adjacent nodes of $M_i^{(k)}$ such that the two nodes receive the same color c_i. Set $U = \{u_1,\ldots,u_n\}$. Using the color c_i we define a map $h : U \to \{1,\ldots,k-1\}$. We set $h(u_i) = c_i$ for each i, $1 \leq i \leq n$.

Note that the map h defines a legal coloring of the nodes of the graph $M^{(k)}$. The only thing which needs verification is that if u_i and u_j are distinct adjacent nodes of $M^{(k)}$, then $c_i \neq c_j$.

Let us assume on the contrary that u_i and u_j are distinct adjacent nodes of $M^{(k)}$ and $c_i = c_j$. The graph $M_i^{(k)}$ has two distinct adjacent nodes $v_{i,i(1)}$ and $v_{i,i(2)}$ such that

$$f(v_{i,i(1)}) = f(v_{i,i(2)}) = c_i.$$

Similarly, the graph $M_j^{(k)}$ has two distinct adjacent nodes $v_{j,j(1)}$ and $v_{j,j(2)}$ such that

$$f(v_{j,j(1)}) = f(v_{j,j(2)}) = c_j.$$

Note that the nodes $v_{i,i(1)}$, $v_{i,i(2)}$, $v_{j,j(1)}$, $v_{j,j(2)}$ are the nodes of a 4-clique in $L^{(3,k)}$. This means that the coloring defined by the map f is not a 3-clique free coloring of the nodes of $L^{(3,k)}$. This shows that the coloring defined by the map h is a legal coloring of the nodes of $M^{(k)}$.

The coloring defined by h is using at most $k-1$ colors. This contradicts the fact that $\chi(M^{(k)}) \geq k$. Thus we may conclude that $\chi^{(3)}(L^{(3,k)}) \geq k$ as we claimed.

Let us turn to the special case $s = 4$. We choose a integer k for which $k \geq 2^{(4-1)}/(4-1)$, that is, $k \geq 3$. Let $M^{(k)}$ be the Mycielski graph with parameter k. Let $u_1,\ldots,u_n$ be the nodes of $M^{(k)}$. We substitute the node u_i of $M^{(k)}$ by an isomorphic copy $L_i^{(3,k)}$ of the graph $L^{(3,k)}$ for each i, $1 \leq i \leq n$. Let $v_{i,1},\ldots,v_{i,m}$ be the nodes of $L_i^{(3,k)}$. We assume that the nodes

$$v_{1,1},\ldots,v_{1,m},\ldots,v_{n,1},\ldots,v_{n,m}$$

are pair-wise distinct. These nodes will be the nodes of the graph $L^{(4,k)}$.

The edges of $L_i^{(3,k)}$ are going to be edges of $L^{(4,k)}$ for each i, $1 \leq i \leq n$. Further, whenever the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$, then we add the edge $\{v_{i,\alpha},v_{j,\beta}\}$ to $L^{(4,k)}$ for each α, β, $1 \leq \alpha,\beta \leq m$.

We claim that $\omega(L^{(4,k)}) \leq 8$.

In order to verify the claim we assume on the contrary that $\omega(L^{(4,k)}) \geq 9$. Let C be a 9-clique in $L^{(4,k)}$. Note that the set $V_i = \{v_{i,1,},\ldots,v_{i,m}\}$ induces $L_i^{(3,k)}$ in $L^{(4,k)}$ as a subgraph of $L^{(4,k)}$. From the fact that $\omega(L_i^{(3,k)}) \leq 4$ it follows that C may have at most 4 nodes in V_i for each i, $1 \leq i \leq n$. Therefore C has nodes in some $L_i^{(3,k)}$ for at least 3 distinct values of i.

Suppose that i and j are distinct numbers in the set $\{1,\ldots,n\}$. A node in $L_i^{(3,k)}$ and a node in $L_j^{(3,k)}$ can be adjacent only if the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$. This means that $M^{(k)}$ contains a 3-clique. But $M^{(k)}$ does not contain any 3-clique. This contradiction completes the proof of the claim.

We claim that $\chi^{(4)}(L^{(4,k)}) \geq k$.

In order to prove the claim let us assume on the contrary that $\chi^{(4)}(L^{(4,k)}) \leq k-1$. Set

$$\begin{aligned} V &= V_1 \cup \cdots \cup V_n \\ &= \{v_{1,1},\ldots,v_{1,n},\ldots,v_{n,1},\ldots,v_{n,n}\}. \end{aligned}$$

Suppose that the map $f : V \to \{1,\ldots,k-1\}$ defines a 4-clique free coloring of the nodes of $L^{(4,k)}$.

The restriction of f to the set V_i is a map $g_i : V_i \to \{1,\ldots,k-1\}$. Clearly, the map g_i defines a coloring of the nodes of the graph $L_i^{(3,k)}$. From the fact that $\chi^{(3)}(L_i^{(3,k)}) \geq k$ it follows that there is a 3-clique in $L_i^{(3,k)}$ such that the 3 nodes of the clique receive the same color c_i. Set $U = \{u_1,\ldots,u_n\}$. Using the color c_i we define a map $h : U \to \{1,\ldots,k-1\}$. We set $h(u_i) = c_i$ for each i, $1 \leq i \leq n$.

Note that the map h defines a legal coloring of the nodes of the graph $M^{(k)}$. The only thing which needs verification is that if u_i and u_j are distinct adjacent nodes of $M^{(k)}$, then $c_i \neq c_j$.

Let us assume on the contrary that $c_i = c_j$. The graph $L_i^{(3,k)}$ has 3 distinct pair-wise adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$, $v_{i,i(3)}$ such that

$$f(v_{i,i(1)}) = f(v_{i,i(2)}) = f(v_{i,i(2)}) = c_i.$$

Similarly, the graph $L_i^{(3,k)}$ has 3 distinct pair-wise adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$, $v_{j,j(2)}$ such that

$$f(v_{j,j(1)}) = f(v_{j,j(2)}) = f(v_{j,j(3)}) = c_j.$$

Note that the nodes

$$v_{i,i(1)}, v_{i,i(2)}, v_{i,i(3)}, v_{j,j(1)}, v_{j,j(2)}, v_{j,j(3)}$$

are the nodes of a 6-clique in $L^{(4,k)}$. This means that the coloring defined by the map f is not a 4-clique free coloring of the nodes of $L^{(4,k)}$. This shows that the coloring defined by the map h is a legal coloring of the nodes of $M^{(k)}$.

In the coloring defined by h most $k-1$ colors occur. This contradicts the fact that $\chi(M^{(k)}) \geq k$. Thus we may draw the conclusion that $\chi^{(4)}(L^{(4,k)}) \geq k$ as we claimed.

Continuing in this way we can complete the proof of the theorem. □

3 Test problems

Problems 2 and 3 are commonly referred as k-clique and maximum clique problems, respectively. As we have pointed out it is a well known result of the complexity theory of algorithms that the maximum clique problem is NP-hard. Loosely speaking it can be interpreted such that the maximum clique problem is computationally demanding.

As at this moment there are no readily available mathematical tools to evaluate the performance of practical clique search algorithms, the standard procedure is to carry out numerical experiments on a battery of well selected benchmark tests.

The most widely used test instances are the Erdős–Rényi random graphs, graphs from the the second DIMACS challenge[1] [8], combinatorial problems of monotonic matrices [25, 22], and hard coding problems of Deletion-Correcting Codes[2] [20].

Evaluating the performances of various clique search algorithms is a delicate matter. On one hand one would like to reach some practically relevant conclusion about the competing algorithms. On the other hand this conclusion is based on a finite list of instances.

One has to be ever cautious not to draw overly sweeping conclusions from these inherently limited nature experiments. (We intended to contrast this approach to the asymptotic techniques which are intimately tied to infinity.) The situation is of course not completely pessimistic. After all, these benchmarks were successful at shedding light on the practicality of many of the latest clique search procedures. However, we should strive for enhancing the test procedures. The main purpose of this paper is to propose new benchmark instances.

There are occasions when we are trying to locate a large clique in a given graph such that the clique is not necessarily optimal. This approach is referred as non-exact method to contrast it to the exhaustive search. For instance constructing a large time table in this way can be practically important and useful even without a certificate of optimality.

The benchmark tests are of course relevant in connection with non-exact procedures too. In order to avoid any unnecessary confusion we would like emphasize that in this paper we are focusing solely on the exact clique search methods.

Let n be a positive integer and let p be a real number such that $0 \leq p \leq 1$. An Erdős-Rényi random graph with parameters n, p is a graph G with vertices $1, 2, \ldots, n$. The probability that the unordered pair $\{x, y\}$ is an edge of G is equal to p for each x, y, $1 \leq x < y \leq n$. The events that the distinct pairs

$$\{x_{i(1)}, y_{i(1)}\}, \ldots, \{x_{i(s)}, y_{i(s)}\}$$

are edges of G are independent of each other for each subset $\{i(1), \ldots, i(s)\}$ of $\{1, 2, \ldots, n\}$, where $s \geq 2$.

In a more formal way the Erdős-Rényi random graph of parameters n, p is a random variable whose values are all the simple graphs with n vertices. The probability distribution over these graphs is specified in the manner we have described above. In this paper we can work safely in a more intuitive level. We start with a complete graph on n vertices and we decide the fate of each edge by flipping a biased coin.

In the case $p = 0$ we end up with a graph consisting of n isolated nodes. In the case $p = 1$ we end up with a complete graph on n nodes. (Paper [4] is the basic reference on Erdős-Rényi random graph.)

Let l, m be positive integers. Let $H_i = (V_i, E_i)$ be a graph consisting of l isolated nodes. This means that $|V_i| = l$ and $E_i = \emptyset$ for each i, $1 \leq i \leq m$. Let $V_i = \{v_{i,1}, \ldots, v_{i,l}\}$. We construct a new graph $G = (V, E)$. We set $V = V_1 \cup \cdots \cup V_m$. The nodes $v_{i,r}$, $v_{j,s}$ are connected by an edge in G whenever $i \neq j$. We may say that the graph G is isomorphic to the lexicographic product of the graphs H and K_m, where H consists of l isolated nodes and K_m is the complete graph on m nodes. (For further details of graph products see [7].)

Clearly, V_i is an independent set in G for each i, $1 \leq i \leq m$. The subgraph induced by $V_i \cup V_j$ in G is a complete bipartite graph for each i, j, $1 \leq i < j \leq m$. Obviously, $\chi(G) = m$ and $\omega(G) = m$ hold. In fact G contains l^m distinct m-cliques.

At this stage we choose a real number p such that $0 \leq p \leq 1$. At each edge of G we flip a biased coin. The edge stays with probability p. We call this step randomizing G. The resulting random graph G' belongs to the parameters l, m, p. The $l = 1$ particular case corresponds to the Erdős-Rényi random graph of parameters m, p.

It is clear that $\chi(G') \leq m$ and $\omega(G') \leq m$. In order to guarantee that $\omega(G') = n$ holds we will plant an m-clique into G'. One can achieve this by picking $x_i \in V_i$ for each i, $1 \leq i \leq m$ and connect each distinct pairs among $x_1 \ldots, x_m$ by an edge in G'.

Benchmark tests based on these random graphs are collected in the BHOSLIB library.[3] (The acronym BHOSLIB stands for Benchmarks with Hidden Optimum Solutions Library.)

After all these preparations we are ready to describe the graphs we would like to propose for testing clique search algorithms. Let k, m be positive integers. Let $M_i^{(k)} = (V_i, E_i)$ be the Mycielski graph of parameter k. Let $V_i =$

[1] ftp://dimacs.rutgers.edu/pub/challenge/
[2] http://neilsloane.com/doc/graphs.html
[3] http://www.nlsde.buaa.edu.cn/~kexu/benchmarks/graph-benchmarks.htm

$\{v_{i,1}, \ldots, v_{i,n}\}$ for each i, $1 \leq i \leq m$. We construct a new graph $G = (V, E)$. We set $V = V_1 \cup \cdots \cup V_m$. Let $v_{i,r}, v_{i,s} \in V_i$. If the unordered pair $\{v_{i,r}, v_{i,s}\}$ is an edge of $M_i^{(k)}$, then we add this pair as an edge to G. These edges will be the blue edges of G. In other words the subgraph induced by V_i in G is isomorphic to $M_i^{(k)}$ for each i, $1 \leq i \leq m$.

Pick $v_{i,r} \in V_i$, $v_{j,s} \in V_j$. We connect the nodes $v_{i,r}$, $v_{j,s}$ by an edge in G whenever $i \neq j$. These edges will be the red edges of G.

Note that the graph G is isomorphic to the lexicographic product of the graphs $M^{(k)}$ and K_m, where $M^{(k)}$ is the Mycielski graph of parameter k and K_m is the complete graph on m nodes. One can verify that $\chi(G) = (k)(m)$ and $\omega(G) = (2)(m)$.

We choose a real number p such that $0 \leq p \leq 1$. We randomize the red edges of G. We flip a biased coin and keep each red edge with probability p. The resulted random graph is denoted by G'. It is obvious that $\chi(G') \leq (k)(m)$ and $\omega(G') \leq (2)(m)$. By planting a $(2m)$-clique into G' we can guarantee that $\omega(G') = (2)(m)$. We pick $x_i, y_i \in V_i$ such that the unordered pair $\{x_i, y_i\}$ is an edge in G' for each i, $1 \leq i \leq m$. Finally, we construct a $(2m)$-clique whose nodes are $x_1, y_1, \ldots, x_m, y_m$.

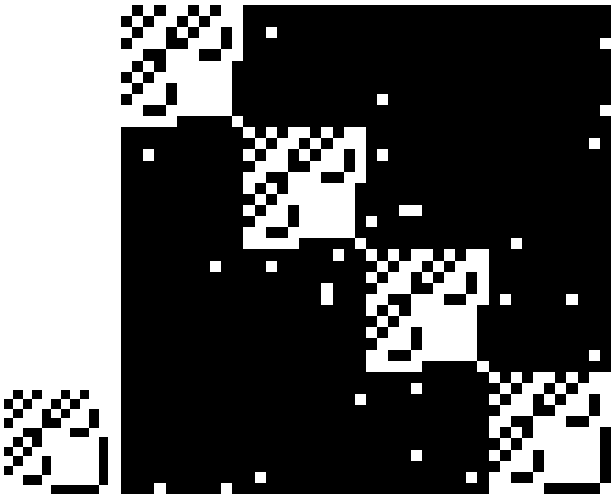

Figure 1: The adjacency matrices of the Mycielski graph $M^{(4)}$ and the random graph G'.

4 Numerical experiments

The proposed new collection of test graphs can be found on the site `clique.ttk.pte.hu/evil`. The source code of the program that generates the adjacency matrices of these graphs are also available on this site.

As an illustration Figure 1 exhibits the adjacency matrix of the Mycielski graph of parameter 4. Further it depicts the adjacency matrix of the randomized version of the lexicographic product of $M^{(4)}$ and K_4, where $M^{(4)}$ is the Mycielsky graph of parameter 4 and K_4 is the complete graph on 4 vertices. As the number of vertices of $M^{(4)}$ and K_4 are equal to 11 and 4, respectively, the product graph has 44 nodes. The probability we used for randomizing is $p = 0.98$.

In the constructions we systematically replaced the Mycielski graph $M^{(k)}$ by the graph $L^{(s,k)}$ that appeared in the proof of Theorem 1 for small values of the parameters s and k. The graph $M^{(4)}$ has 11 nodes. Its chromatic and clique numbers are 4 and 2, respectively. There is a graph with the same chromatic and clique numbers the so-called Chvatal graph. The Chvatal graph has 12 vertices but more symmetric than $M^{(4)}$. We replaced $M^{(4)}$ by the Chvatal graph systematically when we constructed test instances. Note that other graphs also can be used instead of these two examples. Presumably the kind of graphs where the clique number is far from the chromatic number. As the last step we randomly permuted the nodes of the graphs.

We carried out a large scale numerical experiment to check the proposed EVIL benchmark problems. We used 55 test graphs. We took 35 BHOSLIB graphs and 20 EVIL graphs. The experiment involved 7 programs implementing 12 different algorithms and so we are able to compare the running times of 660 clique searches. The processor of the computer we used was a 2.3 GHz, Xeon E5-2670v3.

The 12 clique search algorithms we used are the following.

(1) Östergård[4] [14] (cliquer),

(2) Li[5] [11], [12] (M-cql 10, M-cql 13-1 and M-cql 13-2).

(3) Konc[6] [9] (mcqd and mcqd-dyn)

(4) Prosser[7] (who implemented Tomita's algorithm [24]) (MCR)

(5) San Segundo[8] [16], [17], [18], [19] (BBMC, BBMC-R, BBMC-L and BBMC-X).

There are three ways to use the 2013 version of C.-M. Li program. A switch can be set to either "1" or "2" to select between two built in orderings of the nodes of the graph. In case no value of the switch is specified the program chooses between the "1" and "2" possibilities. During our test we explicitly used the switch "1" and "2" (M-cql 13-1 and M-cql 13-2).

The above programs are high quality state of art programs. It seemed reasonable to enter an unsophisticated program to the competion. The program can be found on the same site as the EVIL instances and goes under the name antiB. The brief description of the program is the following.

[4] `http://users.aalto.fi/~pat/cliquer.html`

[5] `http://home.mis.u-picardie.fr/~cli/EnglishPage.html`

[6] `http://www.sicmm.org/konc/maxclique/`

[7] `http://www.dcs.gla.ac.uk/~pat/maxClique/distribution/`

[8] `https://www.biicode.com/pablodev/examples_clique`

1) Using the simplest sequential greedy algorithm color legally the nodes of the graph and save the colors of the nodes.
2) Set k to be the number of colors of the legal coloring we have located.
3) Carry out a k-clique search.
4) If a k-clique is found, then it is a maximum clique of the graph. Otherwise reduce the value of k and go to step 3.

The k-clique search is based on the Carraghan-Pardalos algorithm, where we utilized original coloring of the nodes. The ordering of the nodes was done by the size of the color classes and the node degrees.

The results of the experiment are summarized in Table 1.

The running times on the BHOSLIB instances are shown in the first part of the table. The evaluation of the results shows that the 2013 version of Li's program with the "2" switch is on is performing unexpectedly well. The running times of he cliquer and the antiB programs are outliers as well.

Our possible explanation is that although the BHOSLIB tests are excellent for heuristic big clique search programs however they are not so good for evaluating exact maximum clique search algorithms. One problem with these instances is that the nodes of these graphs are not randomly permuted. This means that a simple sequential greedy coloring will find the chromatic number at once, as the color classes are laid out consecutively. This can be remedied easily. An other possible problem is that most maximum clique search programs use coloring as an auxiliary algorithm. For these graphs the chromatic number is equal to the clique size leaving a zero optimality or duality gap. So specially designed programs, as the presented antiB, are able to take advantage of this.

The running times for the EVIL benchmark problems are shown in the second part of the table. Here the running times are more evenly distributed.

One particular result was that there is a test graph with 220 nodes – 20 copies of the $M^{(4)}$ graph, $p = 98\%$ edge probability – whose clique number could be determined by only one program in slightly less than 12 hours. We suppose that this problem is the hardest one of such small size.

For certain graph the running times of the cliquer program are again outliers. These short running times could be explained by the fact that the cliquer does not rely on legal coloring of the nodes as do the other programs. After all, we constructed the tests to widen the gap between the chromatic number and the clique number. On the same graphs but with non-permuted nodes the cliquer runs even faster. This again points out the importance of randomly permuting the nodes of the test graphs. The antiB program finishes at the last place as one would expect.

We would like to close this section with a few remarks on why the reader should appreciate the proposed benchmark problems. Although it seems that there is a large number of benchmark problems for maximum clique search algorithms the plain fact is that there are not enough of them. Many of these test problems are too easy for the modern solvers as the sizes of these problems are small. On the other hand there are test instances that are overly hard for the contemporary clique solvers. The proposed EVIL test graphs are forming parameterized families. The parameters can be tuned to produce benchmark problems in various degrees of difficulty.

5 A historical thread

The graphs that are used for benchmarking clique search algorithms are coming from various walks of discrete mathematics and its applications. In this section we will follow a particular thread in order to shed some light on the forces and demands that shaped the evolution of certain benchmark problems.

By the lack of other possibilities the performance of clique search algorithms are commonly evaluated by carrying out large scale numerical experiments. Historically the first clique search procedures were tested almost exclusively on random graphs. The Erdős-Rényi random graphs are readily available with any specified number of nodes and with any specified edge densities. These random graph are popular and useful test instances.

Since the clique size and the chromatic number of a random graph is unknown at the moment of generating them, these test graphs are not the ideal choices to test k-clique search algorithms or for algorithms to list all maximum cliques.

Let $G = (V, E)$ be a complete k-partite graph with n nodes. For the sake of definiteness we assume that V is partitioned into the sets $V_1, \ldots, V_k$ such that $|V_1| = \cdots = |V_k| = s$. Consequently $n = ks$. Suppose $V_i = \{v_{i,1}, \ldots, v_{i,s}\}$. If $i \neq j$, then we connect each node $v_{i,\alpha}$ in V_i with each node $v_{j,\beta}$ in V_j. Thus G has $(1/2)k(k-1)s^2$ edges, that is, $|E| = (1/2)k(k-1)s^2$.

The nodes of G can be legally colored using k colors. The sets $V_1, \ldots, V_k$ can play the roles of the color classes. It is clear that $\omega(G) = k$. Picking exactly one node from each V_i we get mutually adjacent nodes that form the nodes of a k-clique in G. The number of the maximum cliques in G is equal to s^k. These benchmark problems are good candidates to test clique search algorithms that list all maximum cliques. These benchmark problems painfully lack the unpredictability of random graphs.

Therefore when we connect the nodes of the sets V_i and V_j we should do it in a randomized manner. Each of the s^2 edges of the bipartite graph induced by the set $V_i \cup V_j$ in G is chosen with a fixed probability $p_{i,j}$. The clique size of the resulting random graph may decrease and the chromatic number of the resulting random graph may increase. By planting a randomly chosen k-clique into the graph we guarantee that the clique and the chromatic number are equal to k.

These graphs are the BHOSLIB instances. The development of these graphs greatly enhanced the utility of random graphs in testing clique search algorithms.

A graph whose chromatic number is equal to its clique number is by no means is a typical graph. This is why we propose a family of test graphs that have all the desired properties of the BHOSLIB graph and in the same time the gap between the clique and chromatic numbers can be set in a more flexible manner.

Let $G = (V, E)$ be the graph we intend to construct. Let us start with a graph $M = (W, F)$ such that $\omega(M) = r$, $\chi(M) = s$ are known. Suppose $W = \{w_1, \ldots, w_m\}$. We consider k isomorphic copies $M_1, \ldots, M_k$ of M. Let $M_i = (W_i, F_i)$, where $W_i = \{w_{i,1}, \ldots, w_{i,m}\}$.

In order to construct the graph G we set $V = W_1 \cup \cdots \cup W_k$. Here we assume that the sets $W_1, \ldots, W_k$ are pair-wise disjoint. Consequently G has km nodes. Let us pick two distinct W_i and W_j. We connect $w_{i\alpha}$ and $w_{j,\beta}$ for each α, β, $1 \leq \alpha, \beta \leq m$. The resulting graph G has km nodes and $k|F| + (1/2)k(k-1)m^2$ edges. Clearly, $\omega(G) = k\omega(M) = kr$ and $\chi(G) = k\chi(M) = ks$.

As a next step we randomize G. Namely, we pick each of the m^2 edges running between W_i and W_j with a fixed probability $p_{i,j}$. (In many cases we choose all $p_{i,j}$ to be equal.) Because we drop edges from the graph G, the quantities $\omega(G)$, $\chi(G)$ may change. By planting a random (kr)-clique we may restore the original $\omega(G)$. The chromatic number of the new random graph may decrease. If the probability $p_{i,j}$ is close to one then most likely the decrease in the chromatic number is small. When we choose a graph M for which $\omega(M)$ is much smaller than $\chi(M)$, then for the resulted random graph the gap between the clique and chromatic numbers most likely does not disappear.

The resulting random graphs are the test instances proposed in this paper.

The Szemerédi regularity lemma teaches us that a large dense graph can be transformed into a form which is very much similar to our proposed graph. So the proposed graph in a sense is not overly artificial.

6 Reducing the number of nodes

We use the graphs $L^{(s,k)}$ defined in the proof of Theorem 1 to construct benchmark instances. It is imperative to construct hard benchmark instances with as few nodes as possible. In this section we will show that with a little extra care we can reduce the number of the nodes and still get a graph which has the required properties of the graphs $L^{(s,k)}$. For the sake of simplicity we will restrict our attention to the special case $s = 3$.

Let $G = (V, E)$ be a finite simple graph. A subset C of V is called an edge covering set of G if each edge of G has at least one end point in C.

Let $\mu(k)$ be the number of the nodes of the Mycielski graph $M^{(k)}$ of parameter k. The Mycielski graph $M^{(3)}$ is a circle consisting of 5 nodes and 5 edges. Thus $\mu(3) = 5$. The Mycielski graph $M^{(k+1)}$ is constructed from the Mycielski graph $M^{(k)}$ in the following way.

Suppose $U = \{u_1, \ldots, u_n\}$ is the set of nodes of $M^{(k)}$. Here of course $n = \mu(k)$. In order to construct $M^{(k+1)}$ from $M^{(k)}$ we add new nodes $v_1, \ldots, v_n$ to the existing nodes $u_1, \ldots, u_n$. We set $V = \{v_1, \ldots, v_n\}$. We assume that the nodes $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ are pair-wise distinct. In other words we assume that $U \cap V = \emptyset$. We draw an edge between the node v_i and each neighbor of the node u_i for each i, $1 \leq i \leq n$. Finally we add a new node w to the set of nodes $U \cup V$ and draw an edge between the node w and v_i for each i, $1 \leq i \leq n$.

Note that the equation $\mu(k+1) = 2\mu(k) + 1$ holds and using $\mu(3) = 5$ one can compute the number of nodes of the Mycielski graph $M^{(k)}$.

Lemma 1. *The Mycielski graph $M^{(k)}$ has an edge covering set of size $\mu(k-1)+1$ for each $k \geq 3$.*

Proof. For $k = 3$ the Mycielski graph is a circle consisting of 5 vetices and 5 edges. Two adjacent nodes x, y and a node z that is not adjacent to any of x, y form an edge covering set in $M^{(3)}$. This proves the lemma in the special case $k = 3$.

For the remaining part of the proof we may assume that $k \geq 4$. We proceed by an induction on k. The Mycielski graph $M^{(k)}$ has some nodes $u_1, \ldots, u_n$ such that $n = \mu(k-1)$ and the subset $U = \{u_1, \ldots, u_n\}$ induces a subgraph that is isomorphic to $M^{(k-1)}$.

The Mycielski graph $M^{(k)}$ has some nodes $v_1, \ldots, v_n$ such that v_i is adjacent to each neighbor of u_i for each i, $1 \leq i \leq n$. Finally, $M^{(k)}$ has a node w which is adjacent to v_i for each i, $1 \leq i \leq n$.

The set $C = U \cup \{w\}$ is an edge covering set in $M^{(k)}$. This completes the proof of the lemma. □

We do not claim that this edge covering set has the smallest possible number of nodes.

Lemma 2. *For each integer $k \geq 3$ there is a graph $N^{(k)}$ such that it has $[\mu(k-1)+1]\mu(k) + \mu(k-1)$ nodes $\omega(N^{(k)}) \leq 4$ and $\chi^{(3)}(N^{(k)}) \geq k$.*

Proof. We start the construction of $N^{(k)}$ with the Mycielski graph $M^{(k)}$. We assume that $U = \{u_1, \ldots, u_n\}$ is the set of nodes of $M^{(k)}$. By Lemma 1, the graph $M^{(k)}$ has an edge covering set C, where $n = \mu(k)$ and $|C| = \mu(k-1)+1$.

We will assign a subgraph L_i to the node u_i of the graph $M^{(k)}$ for each i, $1 \leq i \leq n$. The graph L_i is either a graph consisting of one single node $v_{i,1}$ or L_i is an isomorphic copy of the Mycielski graph $M^{(k)}$. In this second case the set of nodes of L_i is equal to $V_i = \{v_{i,1}, \ldots, v_{i,r}\}$, where $r = \mu(k)$.

If $u_i \notin C$, then to the node u_i we assign the graph L_i which consists of a single node. If $u_i \in C$, then to the node u_i we assign the graph L_i which has $\mu(k)$ nodes.

Let us suppose that the unordered pair $\{u_i, u_j\}$ is an edge of the graph $M^{(k)}$. We draw edges between each node

of L_i and each node of L_j. The result of repeating this procedure for each edges of the graph $M^{(k)}$ will be the graph $N^{(k)}$. Clearly, $N^{(k)}$ has

$$[\mu(k-1)+1]\mu(k)+\mu(k)-\mu(k-1)-1=$$
$$[\mu(k-1)+1]\mu(k)+2\mu(k-1)+1-\mu(k-1)-1=$$
$$[\mu(k-1)+1]\mu(k)+\mu(k-1)$$

nodes. It remains to show that $\omega(N^{(k)}) \leq 4$ and $\chi^{(3)}(N^{(k)}) \geq k$.

In order to prove that $\omega(N^{(k)}) \leq 4$ we assume on the contrary that $\omega(N^{(k)}) \geq 5$. Let Δ be a 5-clique in $N^{(k)}$. Note that

$$\omega(L_i) = \begin{cases} 1, & \text{if } |V_i| = 1, \\ 2, & \text{if } |V_i| = \mu(k). \end{cases}$$

The clique Δ may contain at most one node from L_i for which $|V_i| = 1$. The clique Δ may contain at most two nodes from L_i when $|V_i| = \mu(k)$. It follows that there must be at least three distinct values of i for which the graph L_i contains a node from the 5-clique Δ. This gives at least three distinct values of i for which the nodes u_i pair-wise adjacent. But $M^{(k)}$ does not have any 3-clique as $\omega(M^{(k)}) \leq 2$.

In order to prove that $\chi^{(3)}(N^{(k)}) \geq k$ we assume on the contrary that $\chi^{(3)}(N^{(k)}) \leq k-1$. Let $V = V_1 \cup \cdots \cup V_n$ be the set of nodes of $N^{(k)}$ and let $f : V \to \{1, \ldots, k-1\}$ be a map that describes the monochrome 3-clique free coloring of the nodes of $N^{(k)}$.

Consider a subgraph L_i of $N^{(k)}$. Let g_i be the restriction of f to $V_i = \{v_{i,1}, \ldots, v_{i,r}\}$. Plainly, the map $g_i : V_i \to \{1, \ldots, k-1\}$ describes a coloring of the nodes of the graph L_i. Using the facts that L_i is isomorphic to $M^{(k)}$ and $\chi(M^{(k)}) \geq k$ we can draw the conclusion that there must be two distinct adjacent nodes of $M^{(k)}$ that receive the same color.

Remember that $U = \{u_1, \ldots, u_n\}$ is the set of nodes of the Mycielski graph $M^{(k)}$ we used in the construction of $N^{(k)}$. We define a map $h : U \to \{1, \ldots, k-1\}$.

If the subgraph L_i of $N^{(k)}$ has only one node, then we set $h(u_i)$ to be $f(v_{i,1})$. In plain English we assign the color of the only node of the graph L_i to the node u_i of the graph $M^{(k)}$.

If the subgraph L_i of $N^{(k)}$ has $\mu(k)$ nodes, then there are two distinct adjacent nodes of L_i, say $v_{i,i(1)}$, $v_{i,i(2)}$ such that $f(v_{i,i(1)}) = f(v_{i,i(2)})$. In this case we set $h(u_i)$ to be $f(v_{i,i(1)})$. In simple English assign the common color of the adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$ of $N^{(k)}$ to the node u_i of $M^{(k)}$.

We claim that the map $h : U \to \{1, \ldots, k-1\}$ describes a legal coloring of the nodes of $M^{(k)}$. The only thing we should check is that if u_i, u_j are distinct adjacent nodes of $M^{(k)}$, then $h(u_i) \neq h(u_j)$ must hold.

Let us start with the case when $u_i \notin C$, $u_j \notin C$. The assumption that u_i, u_j are distinct adjacent nodes of $M^{(k)}$ contradicts the fact that C is an edge covering set in $M^{(k)}$. Therefore this case cannot occur.

If $u_i \in C$, $u_j \in C$, then the subgraphs L_i, L_j of $N^{(k)}$ both have $\mu(k)$ nodes. There are distinct adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$ in L_i such that $f(v_{i,i(1)}) = f(v_{i,i(2)}) = h(u_i)$. Similarly, there are distinct adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$ in L_j such that $f(v_{j,j(1)}) = f(v_{j,j(2)}) = h(u_j)$. Here $h(u_i) \neq h(u_j)$ must hold since otherwise we have a 4-clique in $N^{(k)}$ whose nodes receive the same color. This cannot happen as the map $f : V \to \{1, \ldots, k-1\}$ describes a monochrome 3-clique free coloring of the nodes of $N^{(k)}$.

If $u_i \notin C$, $u_j \in C$, then the subgraph L_i of $N^{(k)}$ has only one node and the subgraph L_j of $N^{(k)}$ has $\mu(k)$ nodes. Now $f(v_{i,1}) = h(u_i)$. There are distinct adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$ in L_j such that $f(v_{j,j(1)}) = f(v_{j,j(2)}) = h(u_j)$. This time $h(u_i) \neq h(u_j)$ must hold since otherwise we have a 3-clique in $N^{(k)}$ whose nodes receive the same color. This cannot happen as the map $f : V \to \{1, \ldots, k-1\}$ describes a monochrome 3-clique free coloring of the nodes of $N^{(k)}$. □

Acknowledgments

This research was supported by National Research, Development and Innovation Office – NKFIH Fund No. SNN-117879.

References

[1] I. M. Bomze, M. Budinich, P. M. Pardalos, M. Pelillo (1999) The Maximum Clique Problem, *Handbook of Combinatorial Optimization* Vol. 4, Kluwer Academic Publisher. https://doi.org/10.1007/978-1-4757-3023-4_1

[2] C. Bron and J. Kerbosch (1973) Finding all cliques of an undirected graph, *Communications of ACM* **16**, 575–577. https://doi.org/10.1145/362342.362367

[3] R. Carraghan, P. M. Pardalos (1990) An exact algorithm for the maximum clique problem, *Operation Research Letters* **9**, 375–382. https://doi.org/10.1016/0167-6377(90)90057-C

[4] P. Erdős, A. Rényi (1960) On the evolution of random graphs, *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **5**, 17–61.

[5] P. Erdős (1959), Graph theory and probability, *Canad. J. Math.* **11**, 34–38.

[6] M. R. Garey and D. S. Johnson (2003), *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, New York.

[7] R. Hammack, W. Imrich, S. Klavžar (2011) *Handbook of Product Graphs*, CRC Press, Boca Raton, FL.

[8] J. Hasselberg, P. M. Pardalos, and G. Vairaktarakis (1993) Test case generators and computational results for the maximum clique problem, *Journal of Global Optimization* **3** , 463–482. http://www.springerlink.com/content/p2m65n57u657605n https://doi.org/10.1007/bf01096415

[9] J. Konc and D. Janežič (2007) An improved branch and bound algorithm for the maximum clique problem, *MATCH Communications in Mathematical and Computer Chemistry* **58**, 569–590.

[10] D. Kumlander (2005) *Some Practical Algorithms to Solve the Maximum Clique problem* PhD. Thesis, Tallin University of Technology.

[11] C.-M. Li, Z. Quan (2010) An efficient branch-and-bound algorithm based on MaxSAT for the maximum clique problem, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence.* (AAAI-10), pp. 128–133.

[12] C.-M. Li, Z. Fang, K. Xu (2013) Combining MaxSAT reasoning and incremental upper bound for the maximum clique problem, *Proceedings of the 2013 IEEE 25th International Conference on Tools with Artificial Intelligence.* (ICTAI2013), pp. 939–946. https://doi.org/10.1109/ictai.2013.143

[13] J. Mycielski (1955) Sur le coloriage des graphes, *Colloq. Math.* **3**, 161–162.

[14] P. R. J. Östergård (2002) A fast algorithm for the maximum clique problem, *Discrete Applied Mathematics* **120**, 197–207. https://doi.org/10.1016/S0166-218X(01)00290-6

[15] C. H. Papadimitriou (1994) *Computational Complexity*, Addison-Wesley Publishing Company, Inc.

[16] P. San Segundo, D. Rodriguez-Losada, A. Jimenez (2011) An exact bit-parallel algorithm for the maximum clique problem, *Computers & Operations Research.* **38**, 571–581. https://doi.org/10.1016/j.cor.2010.07.019

[17] P. San Segundo, F. Matia, D. Rodriguez-Losada, M. Hernando (2013) An improved bit parallel exact maximum clique algorithm, *Optimization Letters.* **7**, 467–479. https://doi.org/10.1007/s11590-011-0431-y

[18] P. San Segundo, C. Tapia (2014) Relaxed approximate coloring in exact maximum clique search, *Computers & Operations Research.* **44**, 185–192. https://doi.org/10.1016/j.cor.2013.10.018

[19] P. San Segundo, A. Nikolaev, M. Batsyn (2015) Infra-chromatic bound for exact maximum clique search, *Computers & Operations Research.* **64**, 293–303. https://doi.org/10.1016/j.cor.2015.06.009

[20] N. J. A. Sloane, *Challenge Problems: Independent Sets in Graphs.* http://neilsloane.com/doc/graphs.html

[21] S. Szabó (2011) Parallel algorithms for finding cliques in a graph, *Journal of Physics: Conference Series* **268** 012030 https://doi.org/10.1088/1742-6596/268/1/012030

[22] S. Szabó (2013) Monotonic matrices and clique search in graphs, *Annales Univ. Sci. Budapest., Sect. Computatorica* **41**, 307–322.

[23] S. Szabó and B. Zaválnij (2012) Greedy algorithms for triangle free coloring, *AKCE International Journal of Graphs and Combinatorics* **9** No. 2, 169–186.

[24] E. Tomita and T. Seki (2003) An efficient branch-and-bound algorithm for finding a maximum clique, *Lecture Notes in Computer Science* **2731**, 278–289.

[25] E. W. Weisstein, Monotonic Matrix, In: *MathWorld–A Wolfram Web Resource.* http://mathworld.wolfram.com/MonotonicMatrix.html

[26] D. R. Wood (1997) An algorithm for finding a maximum clique in a graph, *Oper. Res. Lett.* **21**, 211–217. https://doi.org/10.1016/S0167-6377(97)00054-0

name	$\|V\|$	%	$\omega(G)$	antiB	BBMC	BBMC -R	BBMC -L	BBMC -X	M-clq 10	M-clq 13-1	M-clq 13-2	mcqd	mcqd -dyn	MCR	cliquer
frb30-15-1.clq	450	82	30	0	1611	1645	1694	1613	575	629	0	2735	2541	3673	21
frb30-15-2.clq	450	82	30	0	1010	1094	1191	1047	921	990	0	3329	4155	905	43
frb30-15-3.clq	450	82	30	0	602	581	623	556	432	429	0	1305	2767	300	194
frb30-15-4.clq	450	82	30	0	1855	1768	1901	1676	1154	617	0	5763	4996	2698	2
frb30-15-5.clq	450	82	30	0	1273	1213	1437	1154	726	1110	0	1997	4536	1355	0
frb35-17-1.clq	595	84	35	1	–	–	–	–	–	–	1	–	–	34983	18
frb35-17-2.clq	595	84	35	1	–	–	–	–	–	–	1	–	–	–	104
frb35-17-3.clq	595	84	35	2	–	–	–	–	–	–	0	–	–	22607	14493
frb35-17-4.clq	595	84	35	1	–	–	–	–	27231	42219	0	–	–	5249	7923
frb35-17-5.clq	595	84	35	5	–	–	–	–	–	–	0	–	–	–	181
frb40-19-1.clq	760	86	40	0	–	–	–	–	–	–	1	–	–	11589	–
frb40-19-2.clq	760	86	40	1	–	–	–	–	–	–	0	–	–	–	–
frb40-19-3.clq	760	86	40	8	–	–	–	–	–	–	0	–	–	–	353
frb40-19-4.clq	760	86	40	38	–	–	–	–	–	–	7	–	–	–	2296
frb40-19-5.clq	760	86	40	10	–	–	–	–	–	–	5	–	–	–	78
frb45-21-1.clq	945	87	45	0	–	–	–	–	–	–	119	–	–	–	–
frb45-21-2.clq	945	87	45	118	–	–	–	–	–	–	72	–	–	–	–
frb45-21-3.clq	945	87	45	122	–	–	–	–	–	–	44	–	–	–	–
frb45-21-4.clq	945	87	45	218	–	–	–	–	–	–	36	–	–	–	–
frb45-21-5.clq	945	87	45	475	–	–	–	–	–	–	203	–	–	–	–
frb50-23-1.clq	1150	88	50	16385	–	–	–	–	–	–	764	–	–	–	–
frb50-23-2.clq	1150	88	50	10145	–	–	–	–	–	–	363	–	–	–	–
frb50-23-3.clq	1150	88	50	12585	–	–	–	–	–	–	7938	–	–	–	–
frb50-23-4.clq	1150	88	50	501	–	–	–	–	–	–	17	–	–	–	2754
frb50-23-5.clq	1150	88	50	18256	–	–	–	–	–	–	221	–	–	–	–
frb53-24-1.clq	1272	88	53	73	–	–	–	–	–	–	4771	–	–	–	–
frb53-24-2.clq	1272	88	53	–	–	–	–	–	–	–	190	–	–	–	–
frb53-24-3.clq	1272	88	53	2910	–	–	–	–	–	–	2091	–	–	–	–
frb53-24-4.clq	1272	88	53	29815	–	–	–	–	–	–	4022	–	–	–	–
frb53-24-5.clq	1272	88	53	27671	–	–	–	–	–	–	1071	–	–	–	–
frb59-26-1.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-2.clq	1534	89	59	42408	–	–	–	–	–	–	–	–	–	–	–
frb59-26-3.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-4.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-5.clq	1534	89	59	–	–	–	–	–	–	–	11661	–	–	–	–
chv12x10.clq	120	92	20	–	4	5	4	1	1	8	0	4759	48	700	0
myc5x24.clq	120	97	48	14536	0	0	0	0	0	0	0	2	1	4	112
myc11x11.clq	121	93	22	–	8	12	7	2	1	7	0	1097	85	1081	239
s3m25x5.clq	125	89	20	5440	6	8	6	5	1	1	0	5	6	9	0
myc23x6.clq	138	87	12	1681	2	3	2	2	2	57	0	4	7	56	699
myc5x30.clq	150	97	60	–	1	1	1	0	0	0	0	10	2	47	42042
s3m25x6.clq	150	90	24	–	192	278	195	186	4	12	8	64	92	128	0
myc11x14.clq	154	94	28	–	486	566	422	66	33	235	23	–	11563	–	–
chv12x15.clq	180	94	30	–	26019	34161	26045	7796	1235	26798	184	–	–	–	0
myc5x36.clq	180	97	72	–	3	2	3	2	0	0	0	17	6	118	–
myc23x8.clq	184	90	16	–	115	165	112	88	215	23434	90	1138	1390	–	–
myc11x17.clq	187	95	34	–	40109	–	33957	5056	2378	43935	2375	–	–	–	3159
s3m25x8.clq	200	92	32	–	46253	–	44843	38987	181	1206	478	22778	18148	40089	0
myc5x42.clq	210	98	84	–	26	15	25	4	0	0	0	443	36	1414	–
myc11x20.clq	220	95	40	–	–	–	–	–	–	–	38519	–	–	–	–
myc23x10.clq	230	91	20	–	–	–	–	38104	26210	–	7545	–	–	–	–
chv12x20.clq	240	95	40	–	–	–	–	–	–	–	–	–	–	–	–
myc5x48.clq	240	97	96	–	23	22	25	13	0	0	0	319	39	3316	–
s3m25x10.clq	250	93	40	–	–	–	–	–	6980	44122	–	–	–	–	18
myc11x23.clq	253	95	46	–	–	–	–	–	–	–	–	–	–	–	–

Table 1: Running time results in seconds for the BHOSLIB and EVIL instances. The "–" sign indicates that the running times are exceeding the 12 hour limit.