Discrete Applied Mathematics 242 (2018) 118–129

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

Decomposing clique search problems into smaller instances based on node and edge colorings

Sándor Szabó, Bogdan Zavalnij [*]

Institute of Mathematics and Informatics, University of Pecs, H-7624, Pecs, Ifjusag utja 6, Hungary

A R T I C L E   I N F O

Article history:
Received 30 November 2016
Received in revised form 17 November 2017
Accepted 8 January 2018
Available online 2 February 2018

Keywords:
k-clique
Maximum clique
Clique search algorithm
Independent set
Branch and Bound
Node coloring
Edge Coloring
Greedy coloring
Combinatorial optimization

A B S T R A C T

To carry out a clique search in a given graph in a parallel fashion, one divides the problem into a very large number of smaller instances. To sort out as many resulted smaller problems as possible, one can rely on upper estimates of the clique sizes. Legal coloring of the nodes of the graphs is a commonly used tool to establish upper bound of the clique size. We will point out that coloring of the nodes can also be used to divide the clique search problem into smaller ones. We will introduce a non-conventional coloring of the edges of the given graph. We will gather theoretical and computational evidence that the proposed edge coloring provides better estimates for the clique size than the node coloring and can be used to divide the original problem into subproblems.

1. Introduction

Let $G = (V, E)$ be a finite simple graph. It means that G has finitely many nodes, and G does not have double edges or loops. Let k be a fixed positive integer. A subgraph Δ of G is called a k-clique if each two distinct nodes of Δ are adjacent, and Δ has k nodes. Sometimes we call a k-clique a clique of size k. A k-clique Δ in G is called a maximal clique if Δ is not a subgraph of any $(k + 1)$-clique in G. A k-clique Δ in G is called a maximum clique if G does not contain any $(k + 1)$-clique. The size of a maximum clique in G is called as the clique number of G, and it is denoted by $\omega(G)$.

We describe the clique search problems relevant to this paper.

Problem 1. Given a finite simple graph G and given a positive integer k. Decide if G contains a k-clique.

Problem 1 is a decision problem, and it is well-known that it belongs to the NP-complete complexity class [9].

Problem 2. Given a finite simple graph G and a positive integer k. List all k-cliques in G.

It is clear that Problem 2 is not a decision problem and that it cannot be computationally less demanding than Problem 1. In other words Problem 2 belongs to the NP-hard complexity class.

Clique search problems have many practical applications, and there is a considerable amount of research devoted to them. For details, see for example [3,8,10,11,13,16,18].

* Corresponding author.
E-mail addresses: sszabo7@hotmail.com (S. Szabó), bogdan@ttk.pte.hu (B. Zavalnij).

https://doi.org/10.1016/j.dam.2018.01.006
0166-218X/

Definition 1. We color the nodes of a given finite simple graph G with k colors such that

(1) each node receives exactly one of the colors.
(2) adjacent nodes never receive the same color.

This is the most commonly encountered coloring of the nodes of a graph. We will refer to it as a legal coloring of the nodes of G. If the nodes of G have a legal coloring with k colors, then $\omega(G) \leq k$.

We will point out that coloring of the nodes of the graph helps in dividing the problem into a large number of smaller problems that can be executed in a completely independent fashion. In other words coloring the nodes is relevant in constructing parallel clique search algorithms. This is one of the main results of the paper. As the other main result of this paper, we propose a non-conventional coloring of the edges of a graph, in contrast to edge coloring proposed by Vizing [20].

Definition 2. We color the edges of a graph G with k colors in the following way.

(1) Each edge receives exactly one color.
(2) If x, y, z are distinct nodes of a 3-clique in G, then the edges $\{x, y\}$, $\{y, z\}$, $\{x, z\}$ cannot receive the same color.
(3) If x, y, u, v are distinct nodes of a 4-clique in G, then the edges $\{x, y\}$, $\{u, v\}$ cannot receive the same color.

We call this type of coloring of the edges of G a legal edge coloring. It turns out that if the edges of G can be legally colored with t colors and k is the largest integer for which $k(k-1)/2 \leq t$ holds, then $\omega(G) \leq k$. Therefore, legal coloring the edges of a graph G can be used to establish an upper bound for $\omega(G)$.

The outline of the paper is the following. In Section 2 we will define a set of nodes, the so-called k-clique covering set, that plays a role in dividing a clique search problem into smaller problems. We will point out that legal coloring of the nodes can be used to construct such sets.

In order to use legal edge coloring to divide a clique search problem into smaller problems, we define a k-clique covering edge set in Section 3. We will introduce the concept of quasi-coloring of the nodes and show how this concept can be utilized in constructing k-clique covering edge sets.

We will introduce the concept of derived graph and show that legal edge coloring cannot give weaker clique estimates than the legal node coloring.

In order to compare the performances of greedy node and edge colorings in Section 4, we carried out a large-scale numerical experiment. The benchmark problems and the results are presented in the last part of the paper. The results confirm that the proposed edge coloring method systematically gives us better estimates than legal coloring. We also present examples of when the edge coloring can actually give better bound than the chromatic number.

2. *k*-clique covering node set and coloring of the nodes

Definition 3. Let $G = (V, E)$ be a finite simple graph, and let k be a positive integer. Let $W \subseteq V$. If each k-clique in G has at least one node in W, then we call W a k-clique covering node set of G. (See Fig. 1.)

Let W be a k-clique covering node set in G and let

$$\{v_1, v_2, \ldots, v_n\}$$

be all the nodes in W. Consider the subgraph H_i of G induced by the set of nodes $N(v_i)$ in G for each i, $1 \leq i \leq n$. Here $N(v_i)$ denotes the set of all the nodes of V that are neighbors to v_i.

Let Δ be a k-clique in G. The definition of W states that v_i is a node of Δ for some i, $1 \leq i \leq n$. Consequently, the subgraph H_i contains exactly $k-1$ nodes of the clique Δ. This observation has a clear intuitive meaning. The problem of locating a k-clique in G can be reduced to a list of smaller problems of locating a $(k-1)$-clique in the graph H_i for each i, $1 \leq i \leq n$.

The smaller is the n, the fewer are the subproblems we end up with.

Problem 3. Given a finite simple graph G and a positive integer k, find a minimum size k-clique covering set in G.

Problem 3 cannot be computationally easier than Problem 1, and so Problem 3 belongs to the NP-hard complexity class.

The message is that determining the optimal size of the k-clique covering node sets is a computationally demanding problem. For this reason, instead of working with optimal size k-clique covering node sets, we will work with not necessarily optimal size k-clique covering node sets. There are many widely-used methods in the literature for colorings of the nodes, as for example in [4,6]. One should color the nodes of the graph legally, and then choose the $(k-1)$ biggest color classes $C_1, C_2, \ldots, C_{k-1}$. Let the set of nodes U be the union of these color classes. Clearly, the maximum clique in the subgraph induced by U at most $(k-1)$ as the coloring puts an upper bound on the clique size. From this, it follows that for any k-clique, there should be at least one node outside this set U, so the nodes outside these color classes, namely the nodes in the set $W = V \setminus U$, are forming a k-clique covering node set. This implies that the above described subproblems of searching $(k-1)$-cliques in the H_i induced subgraphs will form a branching in a Branch-and-Bound algorithm. In fact this method was described with some minor modifications in considerable details in [1]. We included it as an example to illustrate our more general approach.

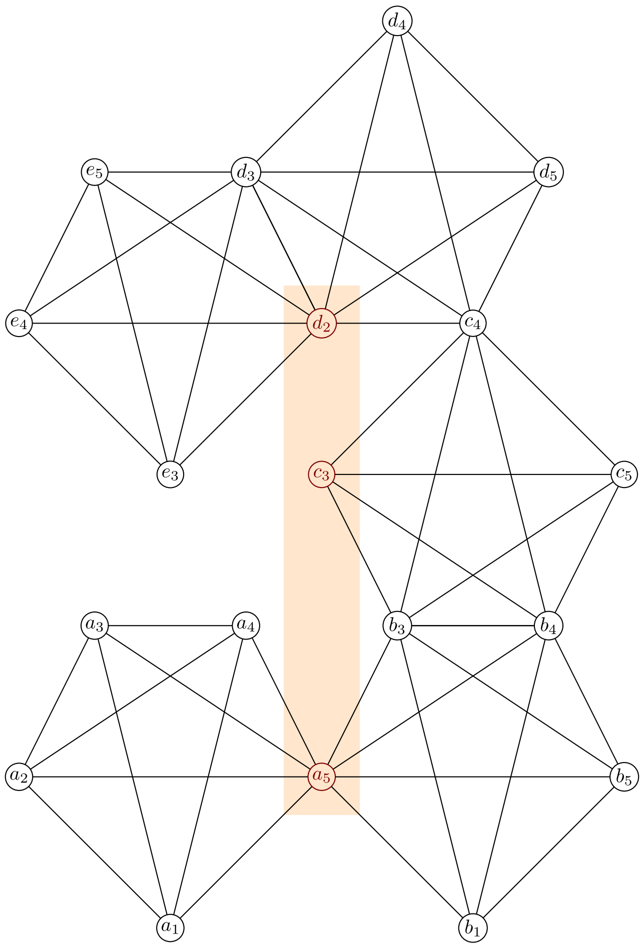

Fig. 1. 5-clique covering node set.

3. *k*-clique covering edge set and quasi-coloring

Definition 4. Let $G = (V, E)$ be a finite simple graph, and let k be a positive integer. Let $F \subseteq E$. If each k-clique in G has at least one edge in F, then we call F a k-clique covering edge set of G. (See Fig. 2.)

Let F be a k-clique covering edge set in G and let

$$e_1 = \{u_1, v_1\}, \dots, e_n = \{u_n, v_n\}$$

be all the edges in F. Consider the subgraph H_i of G induced by the set of nodes $N(u_i) \cap N(v_i)$ in G for each i, $1 \le i \le n$.

Let Δ be a k-clique in G. The definition of F gives that $e_i = \{u_i, v_i\}$ is an edge of Δ for some i, $1 \le i \le n$. Consequently, the subgraph H_i contains exactly $k - 2$ nodes of the clique Δ. This observation has a clear intuitive meaning. The problem of locating a k-clique in G can be reduced to a list of smaller problems of locating a $(k - 2)$-clique in the graph H_i for each i, $1 \le i \le n$.

The smaller the n, the fewer the subproblems we end up with. Thus starting with a k-clique covering edge set with a minimum number of edges may lead to saving computational resources.

Problem 4. Given a finite simple graph G and a positive integer k. What is the minimum size of a k-clique covering edge set in G?

A standard argument gives that Problem 4 is in the NP-hard complexity class [7,13,14].

The message of this observation is that determining the optimal size of the k-clique covering edge sets is a computationally demanding problem. For this reason, instead of working with optimal size k-clique covering edge sets, we will work with not necessarily optimal size k-clique covering edge sets. We will describe polynomial running time greedy algorithms to locate

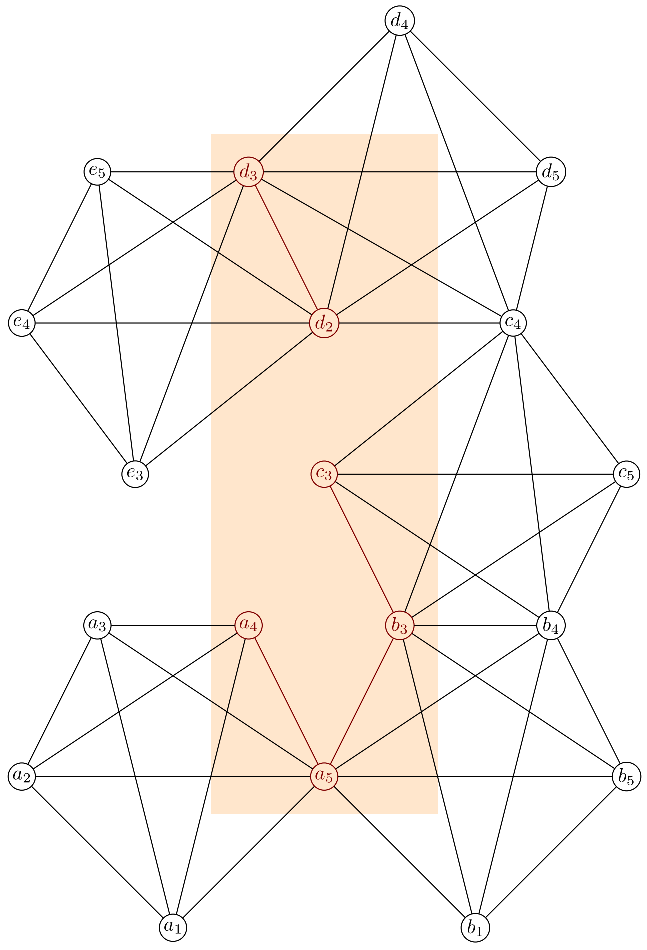

Fig. 2. 5-clique covering edge set.

suboptimal k-clique covering edge sets. These greedy algorithms are based on coloring the nodes or edges of the given graph. We try to assess the performance of these greedy algorithms by carrying out large-scale numerical experiments.

Let $G = (V, E)$ be a finite simple graph. We partition V into the subsets $V_1, \ldots, V_k$. Let us consider the particular case when V_i is an independent set in G. In other words, for each distinct $u, v \in V_i$, the unordered pair $\{u, v\}$ is not an edge of G. If V_i is an independent set in G for each i, $1 \leq i \leq k$, then we can use the sets $V_1, \ldots, V_k$ to define a legal coloring of the nodes of G. Namely, we identify the sets $V_1, \ldots, V_k$ with the color classes of the coloring. In plain English, we color the node v of G with the color i whenever $v \in V_i$.

Let L_i be the subgraph of G induced by the node set V_i in G for each i, $1 \leq i \leq k$. Let t_i be the number of edges of the graph L_i. In the special case $t_i = 0$, the set V_i is an independent set in G. When $t_i \neq 0$, then the set V_i is not an independent set in G. The smaller the value of t_i, the closer V_i is to being an independent set in G.

Definition 5. We refer to the edges of the graph L_i as disturbing edges of the graph G. Set $t = t_1 + \cdots + t_k$. We call the partition $V_1, \ldots, V_k$ of the node set V of G a (k, t)-quasi-coloring of the nodes of G.

The closer the value of t to zero, the closer the quasi-coloring is to being a legal coloring of the nodes of G.

Suppose that the partition $V_1, \ldots, V_{k-1}$ of the node set V of G is a $(k-1, t)$-quasi-coloring of the nodes of G. Let

$$e_1 = \{u_1, v_1\}, \ldots, e_t = \{u_t, v_t\}$$

be all the disturbing edges of the graph G.

Lemma 1. *The edge set $F = \{e_1, \ldots, e_t\}$ is a k-clique covering edge set in the graph G.*

Table 1
The four cases.

Case 1	$x = y$	$u = v$
Case 2	$x = y$	$u \neq v$
Case 3	$x \neq y$	$u = v$
Case 4	$y \neq y$	$u \neq v$

Proof. Let us delete the disturbing edges $e_1, \ldots, e_t$ from the graph G and let G' be the resulting graph. (When we delete the edges, we never delete any of the end nodes of the edges.) Note that the nodes of G' have a legal coloring using $k-1$ colors. It follows that $\omega(G') \leq k-1$.

Let us pick a k-clique Δ in G. If e_i is not an edge of Δ for each i, $1 \leq i \leq t$, then Δ is a k-clique in the graph G'. However, this is not possible as $\omega(G') \leq k-1$. Therefore, e_i is an edge of Δ for some i, $1 \leq i \leq t$. In other words, F is a k-clique covering edge set in G. □

The main difference between the node coloring based k-clique covering node set and the above described k-clique covering edge set is that the second divides the problem into a larger collection of independent subproblems. Numerical experiments show that when in the first case one gets around 10–100 subproblems in connection with a given graph, in the second case one gets over 1000 subproblems [22]. This illustrates that the k-clique covering edge set is more suitable for large-scale parallelization up to thousand processors. In fact, we used this approach with some modifications in our previously published work with great success [22].

3.1. A greedy algorithm for quasi-coloring the nodes

We know from Section 3 that constructing a $(k-1, t)$-quasi-coloring is sufficient to locate a k-clique covering edge set. We would prefer quasi-coloring with a minimum value of t since this is the number of the subproblems we are facing. This motivates the following problem.

Problem 5. Given a finite simple graph G and given a positive integer k, construct a $(k-1, t)$-quasi-coloring of G for which the value of t is the smallest possible.

A routine consideration provides that Problem 5 is NP-hard. The intuitive meaning of this fact is that determining $(k-1, t)$-quasi-colorings with a minimum value of t is a computationally expensive task. Thus in practical computations we have to accept $(k-1, t)$-quasi-colorings with suboptimal values of t.

There are simple and practical greedy algorithms to color the nodes of a graph in a legal manner. A legal coloring of the nodes such as [4,6] can be used conveniently to construct a quasi coloring of the nodes of this graph. Suppose we would like to construct a $(k-1, t)$-quasi-coloring of the nodes of the graph G, where k is a given specified number. Using a greedy coloring algorithm, we color the nodes of the graph G legally involving r colors, and $C_1, \ldots, C_r$ are the color classes.

The colors classes $C_1, \ldots, C_r$ may play the roles of the quasi-color classes $V_1, \ldots, V_r$, and we end up with a $(r, 0)$-quasi-coloring of the nodes of G. If $r \leq k-1$, then there is nothing left to do.

In the $r > k-1$ case, we still set $V_1 = C_1, \ldots, V_r = C_r$. Note that by uniting two quasi-color classes V_i and V_j in a (r, p)-quasi-coloring, we get a $(r-1, p+q)$-quasi-coloring. Here q is the number of such edges of G whose one end node is in V_i, and the other end node is in V_j. In a greedy manner, we will choose V_i and V_j in such a way which keeps the value of q as small as possible. Repeating this procedure, finally we get a $(k-1, t)$-quasi-coloring of the node of G.

Note that with a little extra work, we might be able to further reduce the value of t. If moving a vertex to another quasi-color class reduces the number of disturbing edges, then move it, and repeat this until such vertex exists.

3.2. The derived graph

Using a finite simple graph $G = (V, E)$ we construct a new graph $\Gamma = (W, F)$. The nodes of Γ are the edges of G. Let $\{x, u\}$, $\{y, v\}$ be distinct edges of G. Let us consider the subgraph H of G induced by the set $\{x, y, u, v\}$. If H is a clique in G, then the nodes $\{x, u\}$, $\{y, v\}$ of Γ are connected by an edge in Γ.

Let us distinguish four cases depicted in Table 1.

In case 1, the edges $\{x, u\}$, $\{y, v\}$ of G are identical, contrary to our assumption that the edges are distinct. In other words, this case cannot occur. In case 2, we connect the nodes $\{x, u\}$, $\{y, v\}$ of Γ if $\{u, v\}$ is an edge of G, that is, if $\{u, v\} \in E$. Similarly, in case 3, we connect the nodes $\{x, u\}$, $\{y, v\}$ of Γ if $\{x, y\} \in E$. Finally, in case 4, we connect the nodes $\{x, u\}$, $\{y, v\}$ in Γ if each of the unordered pairs $\{x, y\}$, $\{u, v\}$, $\{x, v\}$, $\{y, u\}$ is an edge of G.

Definition 6. The graph Γ we just constructed from G is called the derived graph of G.

Lemma 2. *If there is an m-clique in G, then there is an $[m(m-1)/2]$-clique in Γ.*

Proof. Let Δ be an m-clique in G. The clique Δ has $m(m-1)/2$ edges in G. These edges of G form $m(m-1)/2$ nodes in Γ, and clearly any two distinct nodes among these nodes are connected in Γ. □

Lemma 3. *If there is a maximum r-clique in Γ, then there is an m-clique in G such that $r = m(m-1)/2$.*

Proof. Let Ω be a maximum r-clique in Γ, such that

$$W = \{\{x_1, u_1\}, \dots, \{x_r, u_r\}\}$$

is the set of nodes of Ω in Γ. Set

$$U = \{x_1, u_1, \dots, x_r, u_r\}.$$

Of course there may be repetition among the elements $x_1, u_1, \dots, x_r, u_r$. We claim that if x, y are distinct elements of U, then the unordered pair $\{x, y\}$ is an edge of G. Indeed, since $x \in U$, there is an $u \in U$, such that $\{x, u\} \in W$. Similarly, since $y \in U$, there is a $v \in U$ for which $\{y, v\} \in W$. In case 1, the edges $\{x, u\}, \{y, v\}$ of G are identical. Case 2 is not possible since $x \neq y$. In case 3, the end points of the edges $\{x, u\}, \{x, v\}, \{x, y\}$ of G are nodes of the clique Ω in Γ. In case 4, the end points of the edges

$$\{x, u\}, \{x, v\}, \{x, y\}, \{u, v\}, \{x, v\}, \{y, u\}$$

of G are nodes of the clique Ω n Γ. In each possible case, the unordered pair $\{x, y\}$ is an edge in G. From the fact that Ω is a maximum clique in Γ, it follows that

$$W = \{\{x, y\} : x, y \in U, x \neq y\}$$

and so $r = |W| = m(m-1)/2$, where $|U| = m$. The subgraph spanned by U in G is a clique Δ in G. Since $|U| = m$, it follows that Δ is an m-clique in G. □

A legal coloring of the nodes of the graph Γ leads a legal coloring of the edges of G. The graph Γ is typically much larger than the graph G. For instance when G has 1000 nodes and 400 000 edges, then the derived graph Γ has 400 000 nodes. A greedy sequential coloring of the edges of G can be carried out using the adjacency matrix of G. That is, for a greedy sequential coloring of the edges of G, we do not need to construct and store the adjacency matrix of Γ. If G has n nodes, then the colors of the edges of G can be stored in an n by n matrix such that the entry at the intersection of row p and column q contains the colors of the edge $\{p, q\}$. In fact, the coloring of the edges of G can be stored in the entries that are above the main diagonal of the matrix. Therefore, if needed, one can store two simultaneous colorings of the edges of G in one n by n matrix.

The edge coloring of G can be utilized relatively easily in the known clique search algorithms like the Caraghan–Pardalos algorithm [5]. Before starting the clique search algorithm, we color the edges of the graph G and store the edge coloring in a matrix M. The clique search algorithms typically unfold by constructing a search tree. The nodes of the search tree represent subgraphs of G. Suppose that the subgraph represented by a node of the search tree is spanned by the subset U of the nodes of G. The rows and columns of the matrix M are labeled by the nodes of G. The subset U identifies a submatrix N of M. From the matrix N, one can read off the number of colors used to color the edges of the subgraph of G spanned by U. The number of colors provides an upper bound for the clique size of the subgraph. This upper estimate may lead to the elimination of the subgraph from the search. In other words, an edge coloring of G can be used to prune the search tree. Since we may store $2, 4, 6, \dots$ edge colorings conveniently, it looks reasonable to try to use an even number of edge colorings.

Observation 1. *The size t of a maximal clique in the derived graph of a finite simple graph must be in the form $t = s(s-1)/2$ for some integer $s \geq 2$.*

Proof. Let G be a finite and let Γ be the derived graph of G. Suppose that Γ contains a maximal t-clique Δ, where $t \geq 1$. Let $\{x_1, y_1\}, \dots, \{x_t, y_t\}$ be the nodes of Δ. Let $u_1, \dots, u_s$ be all the distinct nodes among the nodes $x_1, y_1, \dots, x_t, y_t$ of G and set $U = \{u_1, \dots, u_s\}$. Note that U is the set of nodes of a maximal s-clique in G and so $\{u_i, u_j\}$, $1 \leq i < j \leq s$ are nodes of a maximal clique of size $s(s-1)/2$ in Γ. □

Let $G = (V, E)$ be a finite simple graph.

Observation 2. *If there is a legal coloring of the edges of G with t colors, then $\omega(G) \leq (1 + \sqrt{1+8t})/2$.*

Proof. Set $\omega(G) = k$. It follows that G contains a k-clique Δ. Then the edges of Δ need at least $k(k-1)/2$ colors in a legal edge coloring. This means $k(k-1)/2 \leq t$ must hold. In other words, if k is a positive integer for which $t < k(k-1)/2$, then G cannot contain a k-clique. Shortly, $t < k(k-1)/2$ implies $\omega(G) < k$, or equivalently, $t \leq k(k-1)/2$ implies $\omega(G) \leq k$.

The inequality $t \leq k(k-1)/2$ holds for $k \leq (1 - \sqrt{1+8t})/2$ or for $(1 + \sqrt{1+8t})/2 \leq k$. As $\omega(G), k, t$ are non-negative integers, $\omega(G) \leq (1 + \sqrt{1+8t})/2$. □

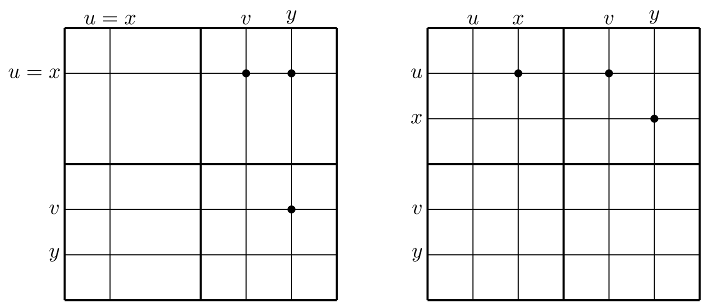

Fig. 3. The adjacency matrix in the proof of Lemma 4.

Replacing t by $\chi(\Gamma)$ in Observation 2, we get

$$\omega(G) \leq (1 + \sqrt{1 + 8\chi(\Gamma)})/2. \tag{1}$$

It is known that

$$\omega(G) \leq \chi(G). \tag{2}$$

Between the two upper bounds (1) and (2), the first one is the better. This claim is equivalent to

$$(1 + \sqrt{1 + 8\chi(\Gamma)})/2 \leq \chi(G),$$

which in turn is equivalent to

$$\chi(\Gamma) \leq \{\chi(G)[\chi(G) - 1]/2\}.$$

Lemma 4. *Let G be a finite simple graph and let Γ be the derived graph of G. Then $\chi(\Gamma) \leq \{\chi(G)[\chi(G) - 1]/2\}$.*

Proof. Let $s = \chi(G)$. There is a legal coloring of the nodes of G with s colors. We will show that there is a legal coloring of the edges of G with $s(s - 1)/2$ colors.

Let $C_1, \ldots, C_s$ be the color classes of the legal coloring of the nodes of G. Let $t = s(s - 1)/2$. We list the ordered pairs (i, j), $1 \leq i < j \leq s$ in a fixed order and number them by the numbers $1, 2, \ldots, t$. In other words, we define a bijection f between the sets $\{(i, j) : \ 1 \leq i < j \leq s\}$ and $\{1, 2, \ldots, t\}$. If the unordered pair $\{x, y\}$ is an edge of G such that $x \in C_i$ and $y \in C_j$, then we color the edge $\{x, y\}$ with the color $f((i, j))$.

We claim that this is a legal coloring of the edges of G. In order to prove the claim, let $\{u, v\}$ and $\{x, y\}$ be distinct edges of G. As $\{u, v\}$ is an edge of G and as G does not contain any loop, $u \neq v$ must hold. Similarly, $x \neq y$ must hold. Since the edges $\{u, v\}$ and $\{x, y\}$ are distinct, $|\{u, v, x, y\}|$ is either 3 or 4.

In the case $|\{u, v, x, y\}| = 3$, we may assume that $u = x$ since this is only a matter of changing the labeling of the nodes. As $\{u, v\}$ is an edge of G, the nodes u and v receive different colors in a legal coloring of the nodes of G. For the sake of definiteness, we assume that u receives color 1 and v receives color 2. Thus $u \in C_1$ and $v \in C_2$. It means that the edge $\{u, v\}$ is colored with color $f((1, 2))$.

The legal coloring of the edges of G is violated only if $\{y, v\}$ is an edge of G and $\{x, y\}$ is colored with color $f((1, 2))$. In this situation, $x \in C_1$ and $y \in C_2$. Now $v \in C_2$ and $y \in C_2$. In particular, the nodes v and y of G receive the same color in a legal coloring of the nodes of G. This is clearly not possible as y and v are adjacent nodes in G. The first diagram of Fig. 3 illustrates the argument in the adjacency matrix of G.

Let us turn to the case $|\{u, v, x, y\}| = 4$. Now the nodes u, v, x, y are the nodes of a 4-clique in G. As $\{u, v\}$ is an edge of G, the nodes u and v receive different colors in a legal coloring of the nodes of G. We assume that $u \in C_1$ and $v \in C_2$. The legal coloring of the edges of G is violated only if

$$\{u, x\}, \ \{u, y\}, \ \{v, x\}, \ \{v, y\}$$

are edges of G and the edge $\{x, y\}$ receives color $f((1, 2))$. By symmetry, we may assume that $x \in C_1$ and $y \in C_2$. Note that $u \in C_1$ and $x \in C_1$. Next note that $\{u, x\}$ is an edge of G. The fact that adjacent nodes in a legal coloring of the nodes receive the same color is an outright contradiction. The second diagram of Fig. 3 illustrates the argument in the adjacency matrix of G. □

4. Numerical results

Using a greedy algorithm, one can color the nodes or the edges of the given graph G. The number of colors provide upper estimates for $\omega(G)$. In order to get reliable results, we selected test graphs to cover a broad range and carried out a large-scale numerical experiment to compare the upper estimates for the clique size $\omega(G)$.

Table 2
The symbols used for labeling the columns.

$\lvert V\rvert$	Number of nodes
$\lvert E\rvert$	Number of edges
ω	The actual clique number
χ_N	The actual node chromatic number
$\overline{\chi}_N$	Estimate of χ_N by the algorithm
$\overline{\omega}$	The estimate for ω using $\overline{\chi}_N$
$\overline{\chi}_E$	Estimate of the χ_E by the algorithm
$\hat{\omega}$	The estimate for ω using $\overline{\chi}_E$

Table 3
Monotonic matrix, sequential greedy coloring.

n	$\lvert V\rvert$	$\lvert E\rvert$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
3	27	189	6	6	10	5
4	64	1 296	12	12	37	9
5	125	5 500	20	20	113	15
6	216	17 550	30	30	273	23
7	343	46 305	42	42	565	34
8	512	106 624	56	56	1 063	46
9	729	221 616	72	72	1 807	60
10	1000	425 250	90	90	2 922	76
11	1331	765 325	110	110	4 477	95
12	1728	1 306 800	132	132	6 602	115
13	2197	2 135 484	156	156	9 390	137
14	2744	3 362 086	182	182	12 998	161
15	3375	5 126 625	210	210	17 600	188

We used the simplest greedy sequential coloring algorithm to construct a legal coloring of the nodes of G and its derived graph Γ. As a more sophisticated approach, we employed Brelaz's algorithm [4] to color the nodes of G and its derived graph Γ.

These coloring algorithms are fairly efficient. The running times for the node coloring were under a second, as the graphs mostly had a few thousand nodes.

Constructing a legal edge coloring was accomplished by legally coloring the nodes of the derived graphs. The derived graphs have as many nodes as the number of the edges of G. These problems were more time consuming. Some graphs had 8 million edges. Therefore, the corresponding derived graphs had 8 million nodes. The running times were in hours for the bigger instances. When we used Brelaz's algorithm for coloring the edges of the largest instances, we relied on a supercomputer with a hundred CPU-s. The program ran for a couple of hours. We have not listed the exact running times. The reason is the following: programs run on different computers, so the running times are not directly comparable. Furthermore, these colorings can be separated from the main body of the clique search algorithms. Legal coloring of the nodes or edges serves simply as a preconditioning step before an actual clique search. The running times of the coloring phase are way below the running times of an actual clique search algorithm. Note that the problems that needed a supercomputer to color them are far beyond the potentials of any clique search program, and one may consider that such a search would run for thousands of years. Finally, our primary aim in this paper was to demonstrate that these colorings can be carried out in reasonable time and provide useful bounds.

In our paper, we focused on the theoretical background, so we do not aim to perform tests on all possible graphs known in the maximum clique search literature. These test graphs are commonly used to test the performance of various clique search algorithms, and our aim is not such comparison. We selected three families of graphs as benchmark tests. The members of these families are covering a large range of graphs. Some of these problems are not demanding for the standard clique search algorithms while other graphs are hard instances and some of them are just not solvable with the present solvers.

The selected test graphs are related to the constructions of monotonic matrices, single deletion error detecting codes and Johnson's codes (with number of 1s is equal to 4 and the Hamming distance is equal to 3). For more background information on these graphs, the reader should consult the references [8,15,17,21].

The results are summarized conveniently in form of tables (see Tables 3–8). The symbols used for labeling the columns are summarized in Table 2.

The comparison of the results is depicted in graphs (see Figs. 4 and 5). The results of the numerical experiments show that the legal edge coloring provides better estimates for the clique number than the legal node colorings. It is also clear that as the problems get harder this difference becomes more and more pronounced. Also, the node coloring was never better than the edge coloring. Given these results, we were thinking that because the edge coloring able to outperform the node colorings, this will also give an estimate under the chromatic number, which is obviously a limit that any node coloring cannot supersede.

Table 4
Monotonic matrix, Brelaz's coloring.

n	$\|V\|$	$\|E\|$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
3	27	189	5	5	10	5
4	64	1 296	10	10	31	8
5	125	5 500	16	16	82	13
6	216	17 550	24	24	192	20
7	343	46 305	35	35	400	28
8	512	106 624	45	45	747	39
9	729	221 616	60	60	1 289	51
10	1000	425 250	75	75	2 073	64
11	1331	765 325	91	91	3135	79
12	1728	1 306 800	109	109	4 582	96
13	2197	2 135 484	128	128	6 509	114
14	2744	3 362 086	156	156	8 948	134
15	3375	5 126 625	178	178	11 981	155

Table 5
Deletion error correcting code, sequential greedy coloring.

n	$\|V\|$	$\|E\|$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
3	8	9	2	2	1	2
4	16	57	4	4	6	4
5	32	305	8	8	17	6
6	64	1 473	14	14	60	11
7	128	6 657	26	26	221	21
8	256	28 801	50	50	875	42
9	512	121 089	101	101	3 406	83
10	1024	499 713	199	199	13 081	162
11	2048	2 037 761	395	395	49 268	314
12	4096	8 247 297	782	782	186 246	610

Table 6
Deletion error correcting code, Brelaz's coloring.

n	$\|V\|$	$\|E\|$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
3	8	9	2	2	1	2
4	16	57	4	4	6	4
5	32	305	6	6	15	6
6	64	1 473	12	12	50	10
7	128	6 657	22	22	189	19
8	256	28 801	42	42	762	39
9	512	121 089	81	81	2 908	76
10	1024	499 713	157	157	10 568	145
11	2048	2 037 761	306	306	37 481	274

Table 7
Johnson code, sequential greedy coloring.

n	$\|V\|$	$\|E\|$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
6	15	45	4	4	3	3
7	35	385	10	10	23	7
8	70	1 855	20	20	107	15
9	126	6 615	35	35	391	28
10	210	19 425	56	56	1 131	48
11	330	49 665	84	84	2 754	74
12	495	114 345	120	120	5 918	109
13	715	242 385	165	165	11 610	152
14	1001	480 480	220	220	21 172	206
15	1365	900 900	286	286	36 514	270
16	1820	1 611 610	364	364	60 054	347
17	2380	2 769 130	455	455	95 038	436
18	3060	4 594 590	560	560	145 441	539

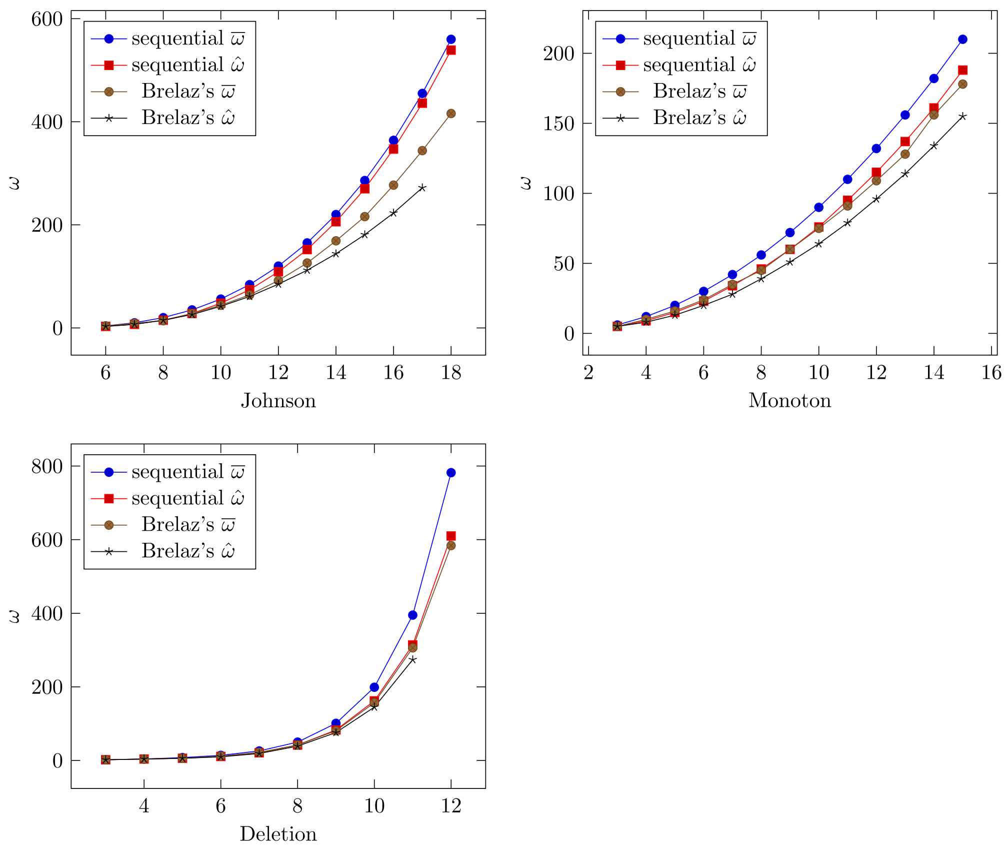

Fig. 4. Summary of the numerical data.

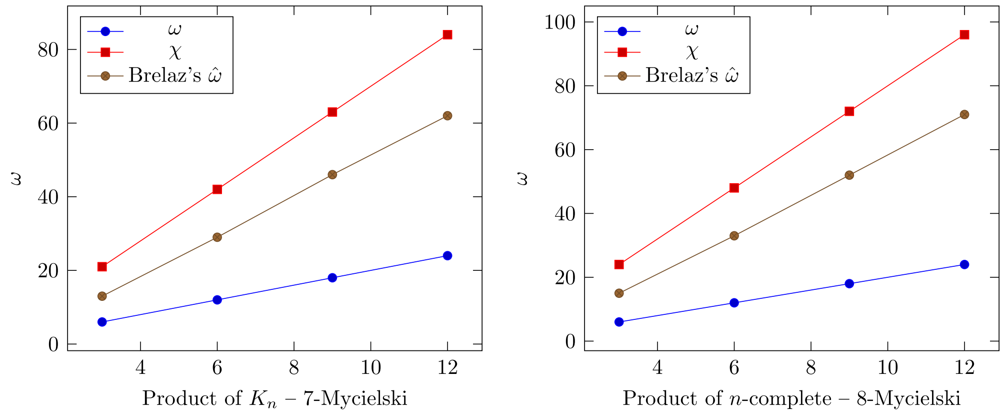

Fig. 5. Summary of the numerical data.

However, the chromatic numbers of our test graphs are not all known. Therefore, we do not know how the results of the greedy colorings compare with the actual chromatic numbers from the first set of tests.

We then selected families of graphs with known chromatic and clique numbers as benchmark problems. These are the so-called Kneser and Mycielski graphs. The definitions of these graphs can be found in [2,12]. We also used the product of the Mycielski and a complete graph. In a complete graph, one replaces each node by a Mycielski graph. This construction is called the product of a Mycielski graph and a complete graph. For the last construction, see [19]. The results are very promising, as there are several cases where the edge coloring bound was under the chromatic number of the given graph. See columns χ_N and $\hat{\omega}$. We highlighted these occurrences by boldface type letters (see Tables 9–12).

Table 8
Johnson code, Brelaz's coloring.

n	$\lvert V\rvert$	$\lvert E\rvert$	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
6	15	45	4	4	3	3
7	35	385	9	9	24	7
8	70	1 855	14	14	109	15
9	126	6 615	28	28	332	26
10	210	19 425	44	44	861	42
11	330	49 665	64	64	1 880	61
12	495	114 345	92	92	3 612	85
13	715	242 385	126	126	6 289	112
14	1001	480 480	169	169	10 411	144
15	1365	900 900	216	216	16 633	181
16	1820	1 611 610	277	277	24 877	223
17	2380	2 769 130	344	344	37 123	272

Table 9
Kneser graph, sequential greedy coloring.

n	k	$\lvert V\rvert$	$\lvert E\rvert$	ω	χ_N	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
15	1	15	105	15	15	15	15	105	15
15	2	105	4 095	7	**13**	13	13	63	**12**
15	3	445	50 050	5	**11**	11	11	32	**8**
15	4	1365	225 225	3	**9**	9	9	10	**5**
15	5	3003	378 378	3	**7**	7	7	3	**3**
16	1	16	120	16	16	16	16	120	16
16	2	120	5 460	8	**14**	14	14	75	**12**
16	3	560	80 080	5	**12**	12	12	40	**9**
16	4	1820	450 450	4	**10**	10	10	16	**6**

Table 10
Kneser graph, Brelaz's coloring.

n	k	$\lvert V\rvert$	$\lvert E\rvert$	ω	χ_N	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
15	1	15	105	15	15	15	15	105	15
15	2	105	4 095	7	**13**	13	13	55	**11**
15	3	445	50 050	5	**11**	11	11	46	**10**
15	4	1365	225 225	3	**9**	9	9	16	**5**
15	5	3003	378 378	3	**7**	8	8	3	**3**
16	1	16	120	16	16	16	16	120	16
16	2	120	5 460	8	**14**	14	14	62	**11**
16	3	560	80 080	5	**12**	12	12	57	**11**
16	4	1820	450 450	4	**10**	10	10	32	**8**

Table 11
Mycielski graphs, Brelaz's coloring.

k	$\lvert V\rvert$	$\lvert E\rvert$	ω	χ_N	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
7	95	755	2	**7**	7	7	1	**2**
8	191	2 360	2	**8**	8	8	1	**2**
9	383	7 271	2	**9**	9	9	1	**2**
10	767	22 196	2	**10**	10	10	1	**2**
11	1 535	67 355	2	**11**	11	11	1	**2**
12	3 071	203 600	2	**12**	12	12	1	**2**
13	6 143	613 871	2	**13**	13	13	1	**2**
14	12 287	1 847 756	2	**14**	14	14	1	**2**

Table 12
Product of an K_n and a k-Mycielski graph, Brelaz's coloring.

n	k	$\lvert V\rvert$	$\lvert E\rvert$	ω	χ_N	$\overline{\chi}_N$	$\overline{\omega}$	$\overline{\chi}_E$	$\hat{\omega}$
3	7	285	29 340	6	**21**	21	21	81	**13**
6	7	570	139 905	12	**42**	42	42	434	**29**
9	7	855	331 695	18	**63**	63	63	1053	**46**
12	7	1140	604 710	24	**84**	84	84	1931	**62**
3	8	573	116 523	6	**24**	24	24	107	**15**
6	8	1146	561 375	12	**48**	48	48	558	**33**
9	8	1719	1 334 556	18	**72**	72	72	1362	**52**
12	8	2292	2 436 066	24	**96**	96	96	2540	**71**

Acknowledgment

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

References

[1] E. Balas, Ch. S. Yu, Finding a maximum clique in an arbitrary graph, SIAM J. Comput. 15 (4) (1986).
[2] M. Barile, E.W. Weisstein, Kneser Graph, MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/KneserGraph.html.
[3] I.M. Bomze, M. Budinich, P.M. Pardalos, M. Pelillo, The Maximum Clique Problem, in: Handbook of Combinatorial Optimization, vol. 4, Kluwer Academic Publisher, 1999.
[4] D. Brelaz, New methods to color the vertices of a graph, Commun. ACM 22 (1979) 251–256.
[5] R. Carraghan, P.M. Pardalos, An exact algorithm for the maximum clique problem, Oper. Res. Lett. 9 (1990) 375–382.
[6] J.C. Culberson, Iterated Greedy Graph Coloring and the Difficulty Landscape, Technical Report, University of Alberta, 1992.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide To the Theory of NP-Completeness, Freeman, New York, 2003.
[8] J. Hasselberg, P.M. Pardalos, G. Vairaktarakis, Test case generators and computational results for the maximum clique problem, J. Global Optim. 3 (1993) 463–482. http://www.springerlink.com/content/p2m65n57u657605n.
[9] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum, New York, 1972, pp. 85–103.
[10] D. Kumlander, Some Practical Algorithms To Solve the Maximal Clique Problem (Ph.D. thesis), Tallin University of Technology, 2005.
[11] C. Morgan, A Combinatorial Search with Dancing Links (Ph.D. Thesis), Univ. of Warwick, 2000.
[12] J. Mycielski, Sur le coloriage des graphes, Colloq. Math. 3 (1955) 161–162.
[13] P.R.J. Östergård, A fast algorithm for the maximum clique problem, Discrete Appl. Math. 120 (2002) 197–207.
[14] C.H. Papadimitriou, Computational Complexity, Addison-Wesley Publishing Company, Inc., 1994.
[15] N.J.A. Sloane, Challenge Problems: Independent Sets in Graphs. https://oeis.org/A265032/a265032.html.
[16] S. Szabó, Parallel algorithms for finding cliques in a graph, J. Phys.: Conf. Ser. 268 (1) (2011) J. Phys.: Conf. Ser. 268, 012030.
[17] S. Szabó, Monotonic matrices and clique search in graphs, Ann. Univ. Sci. Budapest. Sect. Comput. 41 (2013) 307–322.
[18] S. Szabó, B. Zaválnij, Greedy algorithms for triangle free coloring, AKCE Int. J. Graphs Combin. 9 (2) (2012) 169–186.
[19] S. Szabó, B. Zaválnij, Benchmark problems for exhaustive exact maximum clique search algorithms, in: A. Brodnik (Ed.), Middle-European Conference on Applied Theoretical Computer Science, MATCOS 2016: Proceedings of the 19th International Multiconference Information Society–IS 2016, pp. 65–67.
[20] V.G. Vizing, Ob ocenke hromaticheskogo klassa *p*-grafa, Diskret. Anal. 3 (1964) 25–30.
[21] E.W. Weisstein, Monotonic Matrix, MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/MonotonicMatrix.html.
[22] B. Zaválnij, Speeding up parallel combinatorial optimization algorithms with Las Vegas method, in: I. Lirkov, S.D. Margenov, Waśniewski (Eds.), 10th International Conference, LSSC 2015, Sozopol, Bulgaria, June 8–12, 2015. Revised Selected Papers, in: Lecture Notes in Computer Science, vol. 9374, Springer, 2015, pp. 258–266.