Journal of Physics: Conference Series

Parallel algorithms for finding cliques in a graph

To cite this article: S Szabó 2011 *J. Phys.: Conf. Ser.* **268** 012030

Journal of Physics: Conference Series **268** (2011) 012030 doi:10.1088/1742-6596/268/1/012030

Parallel algorithms for finding cliques in a graph

S Szabó

Institute of Mathematics and Informatics, University of Pécs, Ifjúság u. 6. 7624 Pécs, HUNGARY

E-mail: sszabo7@hotmail.com

Abstract. A clique is a subgraph in a graph that is complete in the sense that each two of its nodes are connected by an edge. Finding cliques in a given graph is an important procedure in discrete mathematical modeling. The paper will show how concepts such as splitting partitions, quasi coloring, node and edge dominance are related to clique search problems. In particular we will discuss the connection with parallel clique search algorithms. These concepts also suggest practical guide lines to inspect a given graph before starting a large scale search.

1. Introduction

In a naive but intuitive level a graph is simply a collection of points on the plane some of which are connected by an arc while others are not. In this paper we do not need a more sophisticated graph concept. The points are called nodes or vertices invariably. The arcs are termed edges. If two nodes are connected by an edge, then we may express this fact by saying that the nodes are adjacent or neighbors of each other. The graphs we use will have finitely many points. A node is never connected to itself, that is, the graphs do not admit loops. Two distinct nodes are connected with one edge or not connected at all. In other words, the graphs do not have double edges.

Let Γ be a graph. The set of vertices of Γ is denoted by V. Suppose U is a subset of V. If each two distinct points of U are always connected by an edge of Γ, then we have a subgraph Δ of Γ that is called a clique. The set of vertices of Δ is U. If U has k elements we say that Δ is a clique of size k or simply that Δ is a k-clique in Γ. When U is the empty set, then although the graph Δ does not have any nodes or edges, by definition Δ is considered to be a clique of size zero. When U contains one single node of Γ, then by definition Δ considered to be a clique of size one. A clique Δ in Γ is called maximal if it cannot be extended to a larger clique by adding a further point of Γ. Maximal cliques do not make any appearance in this paper. We mention them just to contrast them to the concept of maximum cliques we define next. A clique Δ in Γ is a maximum clique in Γ if the size of Δ is the largest possible among the cliques in Γ.

It is an empirical fact that finding cliques in graphs can be used in mathematical modeling very much in the same fashion as say solving differential equations. In the course of modeling the graphs are man made objects designed cleverly to express pertinent aspects of the problem in the same spirit as one sets up differential equations. We resist the temptation to detail these problems for it would lead off from our main track. The interested reader may find applications in [1], [2], [3], [4], [5], [6], [7], [8]. The clique search problems of course must be labeled and categorized. Otherwise they could not be used systematically.

Problem 1 *Given a graph Γ and a positive integer k. Decide if Γ has a clique of size k.*

Problem 2 *Given a graph Γ and a positive integer k. List all k-cliques in Γ.*

Problem 1 is called the decision version of the k-clique search problem. Problem 2 is called the listing version or generation version of the k-clique search problem. There are analogous problems to list all maximum cliques in a given graph or exhibit one maximum clique or simply just find the size of the maximum cliques.

By the theory of the complexity of computations, Problem 1 is an NP complete problem. (The acronym "NP" stands for "nondeterministic polynomial".) Loosely speaking the theory predicts that in this family of clique search problems some hard instances are lurking. The theory does not predict where these instances are located.

There are algorithms and well tested programs for solving these problems. (See for example [9], [10], [11], [12], [13], [14], [15], [16], [17], [18].) Parallel programs are also available. (see for instance [19].) In this paper we propose further parallel algorithms. We divide the original problem to smaller problems that can be processed independently. To achieve a reasonable load balancing we try to choose many times more subproblems than processors are available and try to choose subproblems of similar sizes hoping that their running times do not vary too widely.

On the practical side if someone has to carry out a clique search in a hurry, then please use well tested existing programs. (The author does the same.) However, before embarking on a large scale clique search it is advisable to carry out a thorough inspection of the original graph to detect deletable nodes and edges. In the paper we present results relevant to such an inspection and we do recommend using these ideas.

Section 2 describes a parallel algorithm based on splitting partitions. Section 3 provides us with a way to construct splitting partitions using biclique search in a bipartite graph. Section 4 contains two extensions of the coloring idea the triangle free partitions and a new type of edge coloring to estimate the clique size in a graph. The second parallel algorithm is presented in Section 5 and it relies on quasi-coloring of the graph. Section 6 and Section 7 are devoted to the same theme dominance relation defined on nodes and edges to spot nodes and edges that are redundant during clique searches. Finally, Section 8 shows how normal partitions can be used for a parallel clique search.

2. Splitting partitions

Let Γ be a graph and let C_1, C_2, C_3 be a partition of the vertex set V of Γ.

Definition 1 *We say that the partition C_1, C_2, C_3 is a splitting partition of Γ if $C_1 \neq \emptyset$, $C_3 \neq \emptyset$ and if there is no edge of Γ whose one end point is in C_1 and the other end point is in C_3.*

For a splitting partition C_1, C_2, C_3 of Γ we define two subgraphs Γ_1, Γ_3 of Γ. Let Γ_1 be the subgraph of Γ spanned by the vertex set $C_2 \cup C_3$. Let Γ_3 be the subgraph of Γ spanned by the vertex set $C_1 \cup C_2$.

Proposition 1 *If Δ is a clique of Γ, then either Δ is a clique in Γ_1 or Δ is a clique in Γ_3.*

Proof. Assume on the contrary that Δ is neither a clique in Γ_1 nor is a clique in Γ_3. As Δ is not a clique in Γ_1, there is a node a of Δ such that $a \in C_1$. As Δ is not a clique in Γ_3, there is a node b of Δ such that $b \in C_3$. The facts that Δ is a clique in Γ and a, b are nodes of Δ give that $\{a, b\}$ is an edge of Δ and so it is an edge of Γ. This violates that the partition C_1, C_2, C_3 is a splitting partition of Γ. □

A word of warning. Proposition 1 does not imply that if Δ is a clique in Γ, then Δ is a clique only in one of Γ_1 or Γ_3. In fact Δ can be a clique in both Γ_1 and Γ_3. Therefore when one lists all maximum cliques (or k-cliques) in Γ_1 and Γ_3 one may meet a given clique in Γ twice.

Let a be a node of Γ. We say that a has full degree in Γ if a is adjacent to each point of Γ distinct from a.

Proposition 2 *Let C_1, C_2, C_3 be a splitting partition of Γ. If there is node in C_1 which is full degree in Γ_1 or there is a node in C_3 which is full degree in Γ_3, then a maximum clique in Γ cannot be a clique in both of Γ_1 and Γ_3.*

Proof. For the sake of definiteness assume that there is a node $a \in C_1$ which is full degree in Γ_1. Let Δ be a maximum clique in Γ and assume on the contrary that Δ is a clique in Γ_1 and Γ_3. Then Δ is a clique in the subgraph Γ' of Γ spanned by C_2. Note that a is not a node of Δ and $\Delta \cup \{a\}$ is a clique in Γ. Thus Δ cannot be a maximum clique in Γ. □

If C_1, C_2, C_3 is a partition of the set of nodes of Γ, then after rearranging rows and columns, the adjacency matrix $\mathbf{M}$ of Γ can be written in the form

$$\begin{bmatrix} \mathbf{M}_{1,1} & \mathbf{M}_{1,2} & \mathbf{M}_{1,3} \\ \mathbf{M}_{2,1} & \mathbf{M}_{2,2} & \mathbf{M}_{2,3} \\ \mathbf{M}_{3,1} & \mathbf{M}_{3,2} & \mathbf{M}_{3,3} \end{bmatrix}.$$

The block $\mathbf{M}_{i,j}$ is a $|C_i|$ by $|C_j|$ matrix. Here $|C_i|$ stands for the number of the elements of the subset C_i. If the partition C_1, C_2, C_3 is a splitting partition, then each entry of $\mathbf{M}_{1,3}$ and $\mathbf{M}_{3,1}$ is zero. The adjacency matrices of Γ_1 and Γ_3 are

$$\begin{bmatrix} \mathbf{M}_{2,2} & \mathbf{M}_{2,3} \\ \mathbf{M}_{3,2} & \mathbf{M}_{3,3} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{M}_{1,1} & \mathbf{M}_{1,2} \\ \mathbf{M}_{2,1} & \mathbf{M}_{2,2} \end{bmatrix}$$

respectively. In Proposition 1 we may choose the clique Δ to be a maximum clique and we get that finding a maximum clique in Γ can be reduced to finding a maximum clique in two smaller graphs Γ_1 and Γ_3.

Let $\rho = \min\{|C_1|, |C_2|\}$. The larger is ρ the larger is the reductive power of the splitting partition. The reason for this is that the number of vertices of the larger of the graphs Γ_1 and Γ_3 is equal to

$$\max\{|C_2| + |C_3|, |C_1| + |C_2|\} = \max\{|V| - |C_1|, |V| - |C_3|\} = |V| - \rho.$$

As a consequence we try to choose splitting partitions for which ρ is as large as possible.

Problem 3 *Given a graph Γ construct a C_1, C_2, C_3 splitting partition with possibly large value of ρ.*

There is a rather trivial way to construct splitting partitions for which $\rho = 1$. Suppose that a is a node of Γ such that a has at least one non-neighbor in Γ. The set of neighbors of a in Γ we will denote by $N(a)$. Set

$$C_1 = \{a\}, \quad C_2 = N(a), \quad C_3 = V \setminus N(a).$$

Then C_1, C_2, C_3 is a splitting partition of Γ with $\rho = 1$.

Next we describe a greedy algorithm to try to construct splitting partition with $\rho \geq 2$. We introduce three lists L_1, L_2, L_3.

(1) Initially set $L_1 = \emptyset$, $L_2 = \emptyset$, $L_3 = V$.
(2) Pick an $a \in L_2 \cup L_3$. Let L_a be the set of all the neighbors of a in L_3. ($L_a = L_3 \cap N(a)$.) We choose such an a that minimizes $|L_a|$.

(3) If $|L_3|-|L_a|-1 \geq |L_1|+1$, then set $L_1 = L_1 \cup \{a\}$, $L_2 = (L_2 \cup L_a) \setminus \{a\}$, $L_3 = L_3 \setminus [L_a \cup \{a\}]$ and continue at step (2).

(4) If $|L_3| - |L_a| - 1 < |L_1| + 1$, then stop.

After the procedure stops set

$$C_1 = L_1, \quad C_2 = L_2, \quad C_3 = L_3.$$

If $C_1 \neq \emptyset$ and $C_3 \neq \emptyset$, then the partition C_1, C_2, C_3 is a splitting partition of Γ. It may or may not happen that $\rho \geq 2$. If for each vertex a in Γ there is exactly one node b such that $\{a, b\}$ is not an edge of Γ, then the algorithm ends up with a splitting partition with $\rho = 1$. If each node of Γ has full degree in Γ, that is if Γ itself is a clique, then the output of this algorithm is identical to the initial values and the algorithm does not produce a splitting partition in this case.

A possible parallel maximum clique (or k-clique) algorithm is now the following. Let us start with the graph Γ. The greedy algorithm provides us with a splitting partition. If the reducing factor $\rho \geq 2$, then we replace Γ by the smaller graphs Γ_1 and Γ_3. We repeat this procedure with Γ_1 and Γ_3. If the reducing factor ρ is no longer larger than one, then we do not try to split the graph into smaller ones. We end up with a long list of graphs $\Gamma'_1, \ldots, \Gamma'_s$ instead of Γ. The new graphs are smaller than the original graph Γ. These graphs then can be processed independently of each other. The independent processes can use any serial clique search algorithm.

3. Bipartite graphs and bicliques

We now turn to a more systematic non-greedy way to construct splitting partitions. For this we need bipartite graphs and bicliques in bipartite graphs. A graph Λ is called a bipartite graph if the vertex set of Λ can be partitioned into two subsets V_1, V_2 such that Λ has no edge whose end points are in V_1 and Λ has no edge whose end points are in V_2. We will talk about bicliques in bipartite graphs. Let Λ be a bipartite graph whose vertex set is partitioned into V_1, V_2. Let $U_1 \subseteq V_1$, $U_2 \subseteq V_2$. If each vertex of U_1 is adjacent to each vertex of U_2, then the subgraph Ω spanned by $U_1 \cup U_2$ in Λ is called a biclique of Λ. The biclique Ω is called a (k_1, k_2)-biclique if $|U_1| = k_1$, $|U_2| = k_2$. The next biclique search problem is relevant for us.

Problem 4 *Given a bipartite graph Λ and given two integers k_1, k_2. Decide if Λ has a (k_1, k_2)-biclique. If Λ does contain a (k_1, k_2)-biclique, then exhibit one.*

There are algorithms to solve this problem in the literature. See for instance [20].

Let Γ be a graph. We define a bipartite graph Λ whose bicliques can be used to construct splitting partitions of Γ. Let V be the vertex set of Γ and let $\mathbf{M}$ be the adjacency matrix of Γ. Each entry in the main diagonal of $\mathbf{M}$ is zero. Replace each of these zeros by one to get a new matrix $\mathbf{M}'$. The rows and columns of $\mathbf{M}$ and $\mathbf{M}'$ are labeled by the elements of V. The vertex set of Λ is partitioned into V_1, V_2. We define V_1, V_2 by

$$V_1 = \{(v, 1) : \ v \in V\}, \quad V_2 = \{(v, 2) : \ v \in V\}.$$

We connect the nodes $(v, 1)$, $(v', 2)$ of Λ by an edge if the entry of $\mathbf{M}'$ in the (v)-th row and (v')-th column is equal to zero.

Let ρ be a positive integer and suppose that the biclique search algorithm has found a (ρ, ρ)-biclique Ω in Λ. The vertex set of Ω is partitioned into U_1, U_2. There are subsets C_1, C_3 of V such that

$$U_1 = \{(u, 1) : \ u \in C_1\}, \quad U_2 = \{(u, 2) : \ u \in C_3\}.$$

Proposition 3 *If $\rho \geq 1$, then the sets C_1, $C_2 = V \setminus (C_1 \cup C_2)$, C_3 form a splitting partition of the graph Γ.*

Proof. Plainly $\rho \geq 1$ implies that $C_1 \neq \emptyset$, $C_3 \neq \emptyset$. We need to prove that $C_1 \cap C_3 = \emptyset$. This will imply that C_1, C_2, C_3 is a partition of V. Then we should prove that Γ has no edge $\{a, b\}$ with $a \in C_1$, $b \in C_3$.

In order to prove $C_1 \cap C_3 = \emptyset$ assume on the contrary that there is an $a \in C_1$, $a \in C_3$. This means that $(a, 1) \in U_1$, $(a, 2) \in U_2$. Since Ω is a biclique of Λ, the nodes $(a, 1)$, $(a, 2)$ of Λ are adjacent. This gives that the entry in the (a)-th row and (a)-th column of $\mathbf{M}'$ is zero. But the main diagonal of $\mathbf{M}'$ is filled with one. This contradiction verifies the claim that $C_1 \cap C_3 = \emptyset$.

Next suppose that Γ has an edge $\{a, b\}$ for which $a \in C_1$, $b \in C_3$. This means that $(a, 1) \in U_1$, $(b, 1) \in U_2$. Since Ω is a biclique of Λ, the nodes $(a, 1)$, $(b, 1)$ of Λ are adjacent. So the entry of $\mathbf{M}'$ in the (a)-th row and (b)-th column is zero. On the other hand note that $a \neq b$ as Γ has no loops. Since $\{a, b\}$ is an edge of Γ, the entry in the (a)-th row and (b)-th column in $\mathbf{M}$ is one. Using that $a \neq b$, it follows that this entry is the same in $\mathbf{M}$ and $\mathbf{M}'$. ($\mathbf{M}$ and $\mathbf{M}'$ differ only at the main diagonal.) This contradiction completes the proof. □

To illustrate the argument above we work out a small example in details. Let Γ be the graph depicted in figure 1. The adjacency matrix of this Γ graph is given in table 1. Using the adjacency matrix of Γ we construct the bipartite graph Λ. This Λ graph is depicted in figure 2. An inspection shows that the subgraph Ω spanned by the subsets

$$U_1 = \{(4, 1), (5, 1)\}, \quad U_2 = \{(2, 2), (6, 2)\}$$

form a $(2, 2)$-biclique of Λ. From the subsets U_1, U_2 one can read off that the sets

$$C_1 = \{4, 5\}, \quad C_2 = \{1, 3\}, \quad C_3 = \{2, 6\}$$

form a splitting partition of Γ. One can see that the 2 by 2 block at the upper right corner in the rearranged adjacency matrix of Γ is blank. As it supposed to be. The graph Γ can be replaced by the smaller Γ_1 and Γ_3 graphs. We included figure 3 to exhibit these graphs.

4. Coloring

Coloring is a well studied subject in graph theory. (See for example [21].) First we quickly glance over the basic concepts pertinent to clique search. Let Γ be a graph. We color the nodes of Γ with t distinct colors. A coloring of the nodes is called a legal or well coloring if each node is colored with one of the t colors and if adjacent nodes do not receive the same color.

Let C_i be the set of the nodes of Γ that are colored with the (i)-th color. The set C_i is termed the (i)-th color class. The sets $C_1, \ldots, C_t$ form a partition of the vertex set V of Γ. This means that $C_1 \cup \cdots \cup C_t = V$ and $C_i \cap C_j = \emptyset$ for each i, j, $1 \leq i < j \leq t$. Clearly, two distinct nodes in C_i cannot be adjacent in Γ. We call a set C_i edge free if it does not contain edges from Γ. A partition $C_1, \ldots, C_t$ is called an edge free partition if each C_i is edge free.

In a legal coloring of Γ with t colors the color classes $C_1, \ldots, C_t$ form an edge free partition of V. Conversely, if $C_1, \ldots, C_t$ is an edge free partition of V, then coloring the elements of C_i with the (i)-th color we end up with a legal coloring of Γ.

The essential link between the coloring and the clique search is the following observation. Let Δ be a clique of size k in Γ and let $C_1, \ldots, C_t$ be the color classes of a well coloring of Γ. Each color class can contain at most one node of Δ. From this it follows that $k \leq t$. In other words, if Γ admits a legal coloring with $k - 1$ colors, then Γ cannot contain a clique of size k.

Determining the least number of colors a graph can be well colored with is computationally hard. (It is an NP complete problem.) However, there are various greedy algorithms to exhibit if not optimal but still useful well colorings to provide upper bounds for the clique size can appear in a given graph. We propose now two variants of the coloring idea to obtain further upper bounds for the possible clique size.

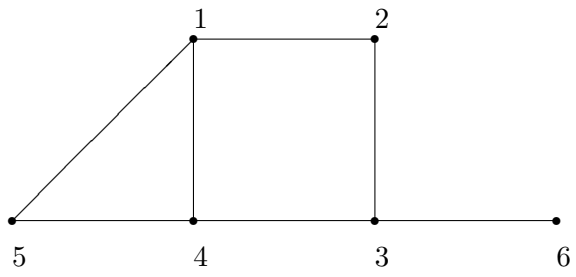

Figure 1. The Γ graph in the first example.

Table 1. The adjacency matrix of Γ in the first example and the rearranged adjacency matrix of Γ.

	1	2	3	4	5	6
1	×	•		•	•	
2	•	×	•			
3		•	×	•		•
4	•		•	×	•	
5	•			•	×	
6			•			×

	4	5	1	3	2	6
4	×	•	•	•		
5	•	×	•			
1	•	•	×		•	
3	•			×	•	•
2			•	•	×	
6				•		×

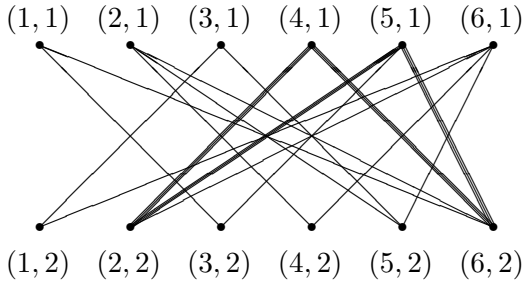

Figure 2. The bipartite graph Λ in the first example. A $(2, 2)$-biclique is highlighted.

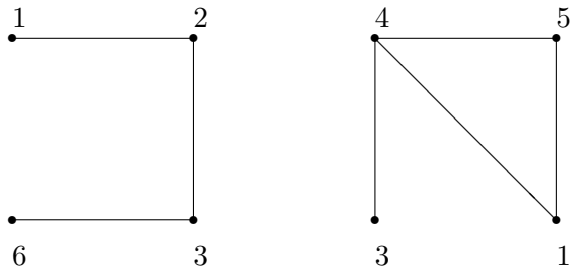

Figure 3. The graph Γ can be replaced by the smaller Γ_1 and Γ_3 graphs.

Definition 2 *A subset C_i of the vertex set V of the graph Γ is called triangle free if C_i does not contain a triangle of Γ. A partition $C_1, \ldots, C_t$ of V is called a triangle free partition if each C_i is triangle free.*

Proposition 4 *Suppose the graph Γ admits a triangle free partition $C_1, \ldots, C_t$. Let Δ be a clique of size k in Γ. Then $k \leq 2t$.*

Proof. As $C_1 \cup \cdots \cup C_t = V$, each node of Δ falls into one of the classes $C_1, \ldots, C_t$. Each C_i can contain at most two nodes of Δ, since C_i does not contain any triangle of Γ. This implies $k \leq 2t$, as required. □

One can describe the triangle free partitions in terms of coloring the nodes of a hyper graph. Recall that a 3-uniform hyper graph Ξ on the set of vertices V is a collection E of hyper edges, where each hyper edge contains three distinct vertices from V. We color the nodes of Ξ using t colors. We say that a coloring is a well coloring or a legal coloring if each node receives exactly one color and if the three nodes of a hyper edge never receive the same color. Let C_i be the set of nodes of Ξ colored with the (i)-th color. We call C_i the (i)-th color class of the coloring.

For a given graph Γ we define a 3-uniform hyper graph Ξ. The nodes of Ξ are the nodes of Γ. Three distinct nodes a, b, c of V form a 3-hyper edge of Ξ if a, b, c are mutually adjacent in Γ. Now the color classes $C_1, \ldots, C_t$ of Ξ form a triangle free partition of Γ.

Problem 5 *Given a graph Γ. Construct a triangle free partition of Γ with possibly few members of the partition.*

Here is a greedy algorithm for solving this problem. Suppose the vertex set V of Γ has n nodes. We start with $n+1$ lists $L, L_1, \ldots, L_n$. Set $L = V$, $L_1 = \cdots = L_n = \emptyset$. While $L \neq \emptyset$ repeat the following. Choose and $a \in L$. If for each edge $\{b, c\}$ in L_i, either $\{a, b\}$ or $\{a, c\}$ is not an edge of Γ, then place a to L_i. If a cannot be placed to L_i, then try to place a to L_{i+1}.

Numerical experiments indicate that greedily constructed triangle free partitions usually provide better upper bounds than greedily constructed edge free partitions. However, this improvement comes for a higher price we should pay in computational resources.

Instead of the nodes of Γ one may try to color the edges of Γ using t distinct colors. For such coloring we single out certain edge colored subgraph of Γ and call these prohibited configurations. Suppose that the edges $\{x, y\}$, $\{u, v\}$ of Γ receive the same color. Since Γ does not have loops, $x \neq y$ and $u \neq v$. If the nodes x, y, u, v are all distinct and they are pair-wise adjacent, then this configuration is prohibited. If two of the nodes x, y, u, v are identical, say $y = u$, and the other two nodes x, v are adjacent, then this configuration is prohibited. The prohibited configurations are depicted in figure 4.

Definition 3 *We call a coloring of the edges of a graph Γ a legal edge coloring of Γ if each edge is colored with one of the t colors and no prohibited configuration occurs.*

Proposition 5 *Suppose that the edges of Γ are legally colored using t colors. Let Δ be a clique of size k in Γ. Then $k(k-1) \leq 2t$.*

Proof. Let C_i be the set of all the edges of Γ that are colored with the (i)-th color. We call C_i the (i)-th color class of Γ. The sets $C_1, \ldots, C_t$ form a partition of the edge set E of Γ since each edge receives exactly one color. As $C_1 \cup \cdots \cup C_t = E$, each edge of Δ falls into one of the color classes. If two distinct edges of Δ falls into the same color class, then this color class will contain a prohibited configuration. By the definition of the legal coloring, Γ does not contain any prohibited configuration. Consequently each color class contains at most one edge of Δ. Since Δ has $(1/2)k(k-1)$ edges, it follows that $(1/2)k(k-1) \leq t$, as required. □

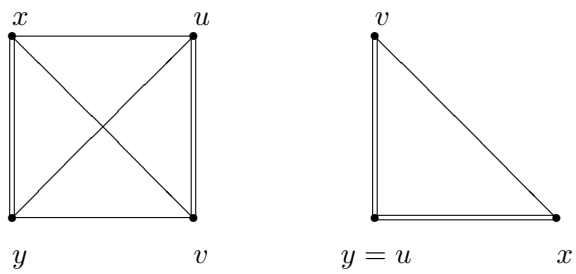

Figure 4. Prohibited configurations. The double line edges are colored with the same color.

The edge coloring of Γ can be expressed in terms of coloring the nodes of a derived graph Θ. The nodes of Θ are the edges of Γ. Temporarily we extend Γ to Γ' by connecting each node of Γ to itself, that is, introducing loops into Γ. Two distinct edges $\{x, y\}$, $\{u, v\}$ of Γ will be adjacent nodes in Θ if the nodes x, y, u, v are mutually adjacent in Γ'. Now a legal coloring of the nodes of Θ results a legal coloring of the edges of Γ.

Problem 6 *Given a graph Γ. Construct a legal coloring of edges of Γ with possibly few colors.*

A straight forward greedy algorithm starts with two lists $L = E$ and $L_1 = \emptyset$, where E is the edge set of Γ. In a typical situation there are $s + 1$ lists $L, L_1, \ldots, L_s$. While $L \neq \emptyset$ repeat the following. Choose an edge e of L_1 and test if e can be placed to L_i by checking if for each edge $f \in L_i$ the edges e, f colored with the same color form a prohibited configuration. If e cannot be placed to any of the lists $L_1, \ldots, L_s$, then open a new list L_{s+1} for e. In either cases set $L = L \setminus \{e\}$.

Alternatively one may start constructing a legal edge coloring of Γ with a legal node coloring of Γ. Suppose $C_1, \ldots, C_s$ are the color classes of a legal vertex coloring of Γ. We arrange the situation such that $|C_1| \leq \cdots \leq |C_s|$. Choose an $a \in C_s$. If $\{a, b\}$ is an edge of Γ such that $b \in C_i$, then color the edge $\{a, b\}$ with color i, that is, place $\{a, b\}$ to the list L_i. After repeating this for each $a \in C_s$ we get $s - 1$ lists $L_1, \ldots, L_{s-1}$. Set $L = E \setminus (L_1 \cup \cdots \cup L_{s-1})$ and start the earlier algorithm with the initial lists $L, L_1, \ldots, L_{s-1}$.

Numerical experiments show that greedy edge colorings can provide tighter bounds than greedy node colorings. But this improvement comes for higher computational price.

5. Quasi coloring

Let Γ be a graph with vertex set V and let $C_1, \ldots, C_t$ be a partition of V. After rearranging rows and columns the adjacency matrix $\mathbf{M}$ of Γ can be written in the form

$$\begin{bmatrix} \mathbf{M}_{1,1} & \mathbf{M}_{1,2} & \ldots & \mathbf{M}_{1,t} \\ \mathbf{M}_{2,1} & \mathbf{M}_{2,2} & \ldots & \mathbf{M}_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{M}_{t,1} & \mathbf{M}_{t,2} & \ldots & \mathbf{M}_{t,t} \end{bmatrix},$$

where the block $\mathbf{M}_{i,j}$ is a $|C_i|$ times $|C_j|$ matrix. The rows of $\mathbf{M}_{i,j}$ are labeled with the elements of C_i and the columns of $\mathbf{M}_{i,j}$ are labeled by the elements C_j. The set C_i is called independent in Γ if any two distinct vertices in C_i are not adjacent. If C_i is an independent set in Γ, then each entry of the matrix $\mathbf{M}_{i,i}$ is zero, that is, $\mathbf{M}_{i,i}$ is a zero matrix. If each $\mathbf{M}_{i,i}$ is a zero matrix, then each C_i is an independent set in Γ and so the nodes of Γ can be well colored with t colors. The sets $C_1, \ldots, C_t$ can be chosen to be color classes of the coloring. In a typical situation the

$\mathbf{M}_{i,i}$ matrices are not necessarily zero matrices. Let 2σ be the overall number of the non-zero entries in the $\mathbf{M}_{i,i}$ matrices. The closer is σ to zero, the closer is the partition $C_1, \ldots, C_t$ to be the color classes of a well coloring of Γ.

Definition 4 *We will refer to a partition $C_1, \ldots, C_t$ of the vertex set of Γ as a (σ, t)-quasi coloring of Γ.*

We propose a parallel k-clique search algorithm based on quasi colorings. Given a graph Γ and a positive integer k. We would like to decide if Γ has a clique of size k. Suppose we have a $(\sigma, k-1)$-quasi coloring of Γ at our disposal. Let $\{a_1, b_1\}, \ldots, \{a_\sigma, b_\sigma\}$ be the edges of Γ that correspond to the 2σ non-zero entries in the matrices $\mathbf{M}_{1,1}, \ldots, \mathbf{M}_{k-1,k-1}$. Let Γ_i be the subgraph of Γ spanned by the set $N(a_i) \cap N(b_i)$.

Proposition 6 *If Γ has a clique of size k, then at least one of the graphs $\Gamma_1, \ldots, \Gamma_\sigma$ has a clique of size $k-2$.*

Proof. Assume on the contrary that Γ_i does not contain a clique of size $k-2$ for each i, $1 \le i \le \sigma$.

There is a clique of size k in Γ. Let Δ be such a clique. If $\{a_1, b_1\}$ is an edge of Δ, then there is a clique Δ_1 of size $k-2$ in $N(a_1) \cap N(b_1)$. Hence Δ_1 is a clique of size $k-2$ in Γ_1. This contradicts the indirect assumption. Thus $\{a_1, b_1\}$ is not an edge of any clique of size k in Γ. This means that we may delete the edge $\{a_1, b_1\}$ from Γ without disturbing the number of cliques of size k in Γ. Let $\Gamma^{(1)}$ be the graph we get from Γ after deleting the node $\{a_1, b_1\}$. (We do not delete any of the nodes a_1 or b_1.)

There must be a clique of size k in $\Gamma^{(1)}$. Let Δ be such a clique. If $\{a_2, b_2\}$ is an edge of Δ, then there is a clique Δ_2 of size $k-2$ in $N(a_2) \cap N(b_2)$. Hence Δ_2 is a clique of size $k-2$ in Γ_2. This contradicts the indirect assumption. Thus $\{a_2, b_2\}$ is not an edge of any clique of size k in $\Gamma^{(1)}$. This means that we may delete the edge $\{a_2, b_2\}$ from $\Gamma^{(1)}$ to get the graph $\Gamma^{(2)}$.

Continuing in this way after σ steps we get that there must be a clique of size k in $\Gamma^{(\sigma)}$. On the other hand note that in the adjacency matrix $\mathbf{M}^{(\sigma)}$ of $\Gamma^{(\sigma)}$ the $\mathbf{M}_{i,i}^{(\sigma)}$ blocks are zero matrices as we have deleted the edges $\{a_1, b_1\}, \ldots, \{a_\sigma, b_\sigma\}$. In other words the nodes of $\Gamma^{(\sigma)}$ can be well colored using $k-1$ colors. Thus $\Gamma^{(\sigma)}$ cannot have any clique of size k. This contradiction completes the proof. □

Problem 7 *Let Γ be a graph and let t be a positive integer. Construct a (σ, t)-quasi coloring of Γ keeping the value of σ possibly small.*

Here is a straight forward greedy algorithm in which we maintain $t+1$ lists $L, L_1, \ldots, L_t$. At the beginning of the computation $L = V$, $L_1 = \cdots = L_t = \emptyset$. While $L \neq \emptyset$ let us iterate the following procedure. Pick an $a \in L$. Let d_i be the number of neighbors of a in L_i. Let d_j be the least among $d_1, \ldots, d_t$. Then set $L = L \setminus \{a\}$, $L_j = L_j \cup \{a\}$. When $L = \emptyset$ set $C_1 = L_1, \ldots, C_t = L_t$. Clearly, the sets $C_1, \ldots, C_t$ form a partition of V. A moment of contemplation will convince the reader that the number of times this procedure reads an entry of the adjacency matrix is at most $(1/2)n(n-1)$, where n is the number of the nodes of Γ.

Alternatively one can construct a (σ, t)-quasi coloring starting with a well coloring of Γ. Let $C_1, \ldots, C_s$ be the color classes of a well coloring of Γ. Suppose that the situation is arranged such that $|C_1| \ge \cdots \ge |C_s|$. Initialize the lists $L, L_1, \ldots, L_t$ by setting

$$L = C_{t+1} \cup \cdots \cup C_s, \qquad L_1 = C_1, \ldots, L_t = C_t.$$

Then distribute the elements of L among the lists $L_1, \ldots, L_t$ using the algorithm above.

A parallel k-clique search algorithm can be summarized in the following way. Let us start with the graph Γ. Construct a $(\sigma, k-1)$-quasi coloring for Γ. Locate the edges $\{a_1, b_1\}, \ldots, \{a_\sigma, b_\sigma\}$. Based on these edges construct the graphs $\Gamma_1, \ldots, \Gamma_\sigma$. By Proposition 6, one of $\Gamma_1, \ldots, \Gamma_\sigma$ has a clique of size $k-2$ if and only if Γ has a clique of size k. In short we reduced the problem into a large number of smaller problems that can be processed independently of each other. The independent processes are free to use any serial $(k-2)$-clique search algorithm.

The algorithm above can also be used to list all k-cliques in Γ. Namely, list all $(k-2)$-cliques in Γ_i then augment them with the nodes a_i, b_i to get the k-cliques in Γ. We should be aware of that in this way we may meet a given k-clique in Γ more than once. If this is undesirable, then one should use the graphs $\Gamma_1^*, \ldots, \Gamma_\sigma^*$ in place of the graphs $\Gamma_1, \ldots, \Gamma_\sigma$. Here Γ_i^* is constructed from Γ_i by deleting the edges $\{a_1, b_1\}, \ldots, \{a_{i-1}, b_{i-1}\}$. (But not deleting the end points of the edges.) The next proposition justifies this claim.

Suppose that Δ_i is a clique of size $k-2$ in Γ_i^*. Obviously Δ_i is a clique in Γ_i too. Set $\Delta^{(i)} = \Delta_i \cup \{a_i\} \cup \{b_i\}$. By the construction of Γ_i, $\Delta^{(i)}$ is a clique of size k in Γ.

Proposition 7 *$\Delta^{(i)} = \Delta^{(j)}$ implies $i = j$.*

Proof. Assume on the contrary $i \neq j$ and $\Delta^{(i)} = \Delta^{(j)}$. We may assume that $i < j$ since this is only a matter of swapping the roles of i and j. From $\Delta^{(i)} = \Delta^{(j)}$ it follows that $\{a_i, b_i\}$ is an edge of $\Delta^{(j)}$. Hence $\{a_i, b_i\}$ is an edge of Γ_j^*. But by the construction of Γ_j^*, $\{a_i, b_i\}$ is not an edge of Γ_j^*. □

6. Dominating nodes

We introduce a binary relation between the nodes of a graph.

Definition 5 *Let Γ be a graph and let a, b be distinct nodes of Γ. We say that b dominates a if a and b are not connected by an edge of Γ and $N(a) \subseteq N(b)$.*

An example of dominated nodes can be seen in figure 5. One can observe that a dominated node can be dropped from the graph during the search for a maximum clique. This is the very reason why we introduced the concept of dominance We spell out this observation formally as a proposition.

Proposition 8 *If a is dominated by b, then a can be canceled from Γ when we are deciding if Γ contains a clique of size k for any given fixed k.*

Proof. Let Δ be a clique of size k of Γ. Let Γ' be the graph we get from Γ after deleting the node a. Only one of the following four possibilities may hold in connection with a, b, Δ

(i) $a \in \Delta$, $b \in \Delta$,
(ii) $a \in \Delta$, $b \notin \Delta$,
(iii) $a \notin \Delta$, $b \in \Delta$,
(iv) $a \notin \Delta$, $b \notin \Delta$.

If $a \notin \Delta$, then $\Delta \subseteq \Gamma'$ and so Γ' contains a clique of size k. For the remaining part of the proof we may assume that $a \in \Delta$. Note that $b \in \Delta$ cannot hold since a and b are not connected by edge in Γ. Thus $b \notin \Delta$. Since $a \in \Delta$, there is a clique Δ_1 of size $k-1$ in $N(a)$. Now $\Delta_1 \subseteq N(b)$ as $N(a) \subseteq N(b)$. Clearly, $\Delta_1 \cup \{b\}$ is a clique of size k. Thus Γ' contains a clique of size k. □

We would like to emphasize that listing all maximum cliques (or k-cliques) in Γ' does not necessarily provide a complete list of the maximal cliques (or k-cliques) of Γ. In other words the reduction offered by Proposition 8 is not applicable when one needs complete lists of maximum

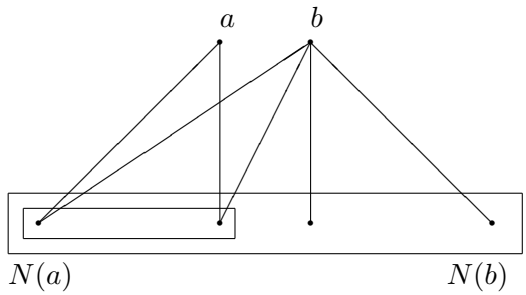

Figure 5. Node a is dominated by node b.

or k-cliques. On the other hand getting rid of certain maximal cliques (or k-cliques) promises that an algorithm visits a smaller portion of the search space.

The dominance relation is obviously not symmetric. Further the dominance relation is not anti symmetric either as it may happen that a and b are distinct nodes of Γ such that $N(a) = N(b)$, a and b are not connected by an edge in Γ. In this case a and b are mutually dominate each other in the same time a and b are not equal.

Proposition 9 *The dominance relation is transitive.*

Proof. Let a, b, c be vertices of the graph Γ. Suppose that a and b are not connected by edge in Γ and $N(a) \subseteq N(b)$. Further suppose that b and c are not connected and $N(b) \subseteq N(c)$. If a and c are not connected by edge in Γ, then c dominates a. For the remaining part of the proof we may assume that there is an edge between a and c. This means that $c \in N(a)$. Using $N(a) \subseteq N(b)$ it follows that $c \in N(b)$. Therefore b and c are connected by edge in Γ. This is an outright contradiction. □

Let a, b, c distinct nodes of Γ such that b dominates a. Let Γ' be the graph we get from Γ after deleting c.

Proposition 10 *Node b dominates node a in Γ' too.*

Proof. Assume that b dominates a in Γ. This means that a and b are not connected distinct nodes such that $N(a) \subseteq N(b)$. For a node u of Γ' the set of neighbors of u in Γ' is denoted by $N'(u)$. It is plain that a and b are not connected in Γ'. We should verify that $N'(a) \subseteq N'(b)$. We distinguish the following four cases

(i) $c \notin N(a)$, $c \notin N(b)$,
(ii) $c \notin N(a)$, $c \in N(b)$,
(iii) $c \in N(a)$, $c \notin N(b)$,
(iv) $c \in N(a)$, $c \in N(b)$.

If $c \notin N(a)$ and $c \notin N(b)$, then $N'(a) = N(a)$ and $N'(b) = N(b)$. From $N(a) \subseteq N(b)$ it follows that $N'(a) \subseteq N'(b)$.

If $c \notin N(a)$ and $c \in N(b)$, then $N'(a) = N(a)$ and $N'(b) = N(b) \setminus \{c\}$. Subtracting $\{c\}$ from both sides of $N(a) \subseteq N(b)$ implies that $N'(a) \subseteq N'(b)$.

If $c \in N(a)$ and $c \notin N(b)$, then $N(a) \subseteq N(b)$ gives the contradiction that $c \in N(b)$. Therefore this case is not possible.

If $c \in N(a)$ and $c \in N(b)$, then $N'(a) = N(a) \setminus \{c\}$ and $N'(b) = N(b) \setminus \{c\}$. Subtracting $\{c\}$ from both sides of $N(a) \subseteq N(b)$ we get that $N'(a) \subseteq N'(b)$. □

In order to illustrate the results in this section let us consider an example. Let Γ be the graph depicted in figure 6. The adjacency matrix of Γ is given by table 2. An inspection reveals that

node 2 dominates nodes 1, 5, 9,
node 3 dominates node 4,
node 5 dominates nodes 1, 2, 9,
node 6 dominates node 4,
node 7 dominates nodes 3, 4, 6.

By Proposition 8, we may delete nodes 1, 2, 3, 4, 6, 9 from the graph when we are looking for a maximum clique. The remaining nodes are 5, 7, 8. They form a clique of size 3. Therefore the size of the maximum clique in the original graph is 3.

The graph in the example above is exceedingly small. However it illustrates the point well. Namely, before embarking on determining the size of a maximum clique in a given graph we should inspect the graph to spot dominating nodes. When found the dominated node can be deleted and we should start to look for dominating nodes in the reduced graph.

Problem 8 *Given a graph Γ and a node a of Γ. Decide if there is node b in Γ that dominates node a.*

A proposed algorithm for solving Problem 8 works with two lists L_1 and L_2. We fill in L_1 with the neighbors of a. The order of the elements in L_1 is irrelevant. We fill in L_2 with the non-neighbors of a. Again, the ordering of the elements in L_2 is irrelevant. (We do not place a to L_2.) Pick a $b \in L_2$. Note that if for each $c \in L_1$ the edge $\{c, b\}$ is edge of Γ, then b dominates a. In order to verify the claim note that the nodes a, b are not adjacent as $b \notin N(a)$. Further $N(a) \subseteq N(b)$ holds since $c \in N(a)$ implies $c \in N(b)$.

The algorithm above reads an entry of the adjacency matrix of Γ to check if two nodes are adjacent or not. Let s be the number of times the algorithm reads a single entry of the adjacency matrix. Let n be the number of nodes of Γ.

Proposition 11 $s \leq n + (1/4)n^2$.

Proof. Let n_1 be the length of L_1 and let n_2 be the length of L_2. Clearly $n_1 + n_2 = n - 1$. Filling in L_1 and L_2 needs n readings of the adjacency matrix. For an element $b \in L_2$ there are n_2 choices. For an element $c \in L_1$ there are n_1 choices. After at most $n_1 n_2$ readings of the adjacency matrix we will reach a definite conclusion if there is a $b \in L_2$ such that $\{b, c\}$ is an edge of Γ for each $c \in L_1$.

The solution of the optimization problem, maximize $n_1 n_2$ subject to $n_1 + n_2 = n - 1$, is $n_1 = n_2 = (1/2)(n - 1)$. Therefore $s \leq n + n_1 n_2 \leq n + (1/4)(n - 1)^2 \leq n + (1/4)n^2$. □

Problem 8 is formulated solely for the purpose to incorporate the algorithm that solves it into the Carraghan-Pardalos clique search algorithm. (The description of the algorithm with a complete program can be found in [11].) The author has experimented with this modified clique search algorithm. The result is that there are graphs for which the extra work of testing dominance pays off and there are graph for which the modified algorithm slows down very slightly.

Another way to utilize this algorithm is simply to inspect the initial adjacency matrix of Γ to test if there are nodes to delete before one starts the proper clique search. By Proposition 11, the full inspection can be accomplished in at most $n^2 + (1/4)n^3$ looks up of a single entry of the adjacency matrix. When a dominated node is found we can delete the node at once. Since

Table 2. The adjacency matrix of Γ in the second example

	1	2	3	4	5	6	7	8	9
1	×		•	•		•	•		
2		×	•	•		•	•	•	
3	•	•	×		•			•	
4	•	•		×	•				
5			•	•	×	•	•	•	
6	•	•			•	×			•
7	•	•			•		×	•	•
8		•	•		•		•	×	•
9						•	•	•	×

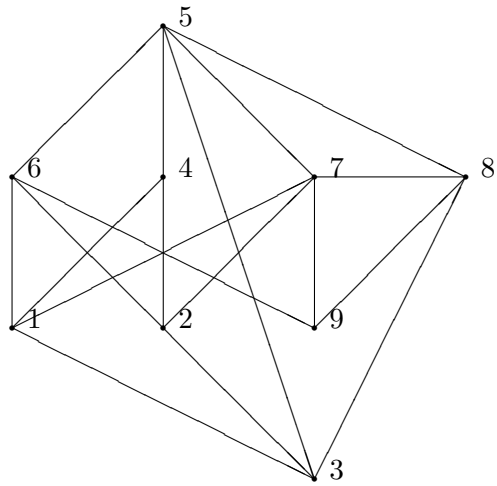

Figure 6. The graph Γ in the second example.

by Proposition 10, the dominated nodes in the original graph remain dominated in the reduced graph.

The edge dominance problem can be put in a different form. Given a node b of Γ. Find all nodes a of Γ that is dominated by b. This problem can be solved in a similar way. Initially set the list L to be the non-neighbors of b. ($L = V \setminus [N(b) \cup \{b\}]$.) Note that if $a \in L$ and for each $c \in L$, $\{c, a\}$ is not edge of Γ, then b dominates a.

Let $m = |L|$, $n = |V|$. The list L can be initialized with n looks up of the adjacency matrix of Γ. The node $c \in L$ can be chosen in m ways. Similarly, the node $a \in L$ can be chosen in m ways. Therefore after at most $n + m^2$ looks up we have a list of the nodes dominated by b.

In challenging clique search instances the nodes usually have more neighbors than non-neighbors. In the $m < n/2$ particular case the worst case estimate for this algorithm is better than the worst case estimate for the previous algorithm. The author has been experimenting with the dominance spotting in the original graph for a considerable time. There are cases when no dominated nodes are spotted at all and the initial graph cannot be reduced in this manner However, there are cases when the clique search problem with the original graph is out of the feasibility range while it is feasible for the reduced graph.

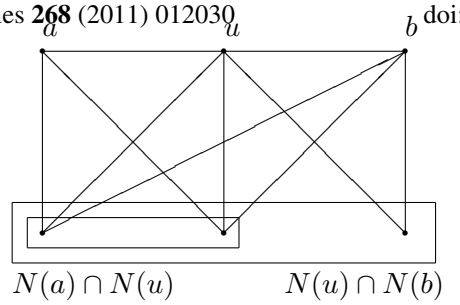

Figure 7. Edge $\{a, u\}$ is dominated by edge $\{u, b\}$. The dominance is short.

7. Dominating edges

We introduce dominance relations between edges of a graph.

Definition 6 *Let Γ be a graph and let a, u, b be distinct nodes of Γ such that $\{a, u\}$, $\{u, b\}$ are edges of Γ. We say that edge $\{u, b\}$ dominates edge $\{a, u\}$ if $b \notin N(a) \cap N(u)$ and $N(a) \cap N(u) \subseteq N(u) \cap N(b)$.*

The dominance in Definition 6 is illustrated by figure 7. We will introduce another edge dominance later so let us call the above defined edge dominance short edge dominance. Let Γ' be the graph we get from Γ after deleting the edge $\{a, u\}$. (But not deleting the nodes a or u.)

Proposition 12 *If Γ contains a clique of size k, then so does Γ'.*

Proof. Let Δ be a clique of size k in Γ. If $\{a, u\}$ is not an edge of Δ, then Δ is a clique of size k in Γ' too. For the remaining part of the proof we may assume that $\{a, u\}$ is an edge of Δ. Now there is a clique Δ_1 of size $k-2$ in $N(a) \cap N(u)$. Simply we can choose Δ_1 to be $\Delta \setminus (\{a\} \cup \{u\})$. Using $N(a) \cap N(u) \subseteq N(u) \cap N(b)$ we get that Δ_1 is a clique of size $k-2$ in $N(u) \cap N(b)$. It follows that $\Delta_2 = \Delta_1 \cup \{u\} \cup \{b\}$ is a clique in Γ. Note that $a \notin \Delta_1$, $u \notin \Delta_1$. Consequently Δ_2 is a clique of size k in Γ'. □

Proposition 13 *If edge $\{a, u\}$ is dominated by edge $\{u, b\}$ and edge $\{b, u\}$ is dominated by edge $\{u, c\}$, then edge $\{a, u\}$ is dominated by edge $\{u, c\}$.*

Proof. We have to verify that $c \notin N(a) \cap N(u)$ and $N(a) \cap N(u) \subseteq N(u) \cap N(c)$. In order to show that $c \notin N(a) \cap N(u)$ assume on the contrary that $c \in N(a) \cap N(u)$. The fact that edge $\{a, u\}$ is dominated by edge $\{u, b\}$ means that $N(a) \cap N(u) \subseteq N(u) \cap N(b)$. Thus $c \notin N(u) \cap N(b)$. But as edge $\{b, u\}$ is dominated by edge $\{u, c\}$, $c \notin N(b) \cap N(u)$. This contradiction shows that $c \notin N(a) \cap N(u)$, as required.

In order to show that $N(a) \cap N(u) \subseteq N(u) \cap N(c)$ note that $N(a) \cap N(u) \subseteq N(u) \cap N(b)$ since node $\{a, u\}$ is dominated by edge $\{u, b\}$. Further $N(b) \cap N(u) \subseteq N(u) \cap N(c)$ since edge $\{b, u\}$ is dominated by edge $\{u, c\}$. Thus $N(a) \cap N(u) \subseteq N(u) \cap N(c)$, as required. □

Let a, b, c, u, v be nodes of Γ such that $\{a, u\}$ is dominated by $\{u, b\}$ and $\{a, v\}$ is dominated by $\{v, c\}$. Let Γ' be the graph we get from Γ after deleting the edge $\{a, u\}$. (But not deleting the nodes a or u.)

Proposition 14 *Node $\{a, v\}$ is dominated by $\{v, b\}$ in Γ'.*

Proof. From the assumptions of the proposition we know that

$$\{a, u\}, \{u, b\} \text{ are edges of } \Gamma, \tag{1}$$

$$a, b \text{ are not adjacent in } \Gamma, \tag{2}$$

$$N(a) \cap N(u) \subseteq N(u) \cap N(b). \tag{3}$$

Further

$$\{a, v\}, \{v, c\} \text{ are edges of } \Gamma, \tag{4}$$

$$a, c \text{ are not adjacent in } \Gamma, \tag{5}$$

$$N(a) \cap N(v) \subseteq N(v) \cap N(c). \tag{6}$$

Using these we would like to verify that

$$\{a, v\}, \{v, c\} \text{ are edges of } \Gamma', \tag{7}$$

$$a, c \text{ are not adjacent in } \Gamma', \tag{8}$$

$$N'(a) \cap N'(v) \subseteq N'(v) \cap N'(c). \tag{9}$$

Here $N'(x)$ stands for the set of neighbors of x in Γ'.

By (5), a and c are not adjacent in Γ. Deleting the edge $\{a, u\}$ is not going to make a and u adjacent. This verifies (8).

By (4), $\{a, v\}$, $\{v, c\}$ are edges of Γ. Deleting the edge $\{a, u\}$ cannot alter this fact. This verifies (7).

In order to verify (9) note that only the following cases may occur in connection with u, $N(a) \cap N(v)$, $N(v) \cap N(c)$.

(i) $u \notin N(a) \cap N(v)$, $\quad u \notin N(v) \cap N(c)$,
(ii) $u \notin N(a) \cap N(v)$, $\quad u \in N(v) \cap N(c)$,
(iii) $u \in N(a) \cap N(v)$, $\quad u \notin N(v) \cap N(c)$,
(iv) $u \in N(a) \cap N(v)$, $\quad u \in N(v) \cap N(c)$.

By (6), the third case is not possible. In the first case

$$N'(a) \cap N'(v) = N(a) \cap N(v), \quad N'(v) \cap N'(c) = N(v) \cap N(c)$$

and so by (6), (9) holds. In the fourth case

$$N'(a) \cap N'(v) = [N(a) \cap N(v)] \setminus \{u\}, \quad N'(v) \cap N'(c) = [N(v) \cap N(c)] \setminus \{u\}$$

and so by (6), (9) follows. In the second case

$$N'(a) \cap N'(v) = [N(a) \cap N(v)] \setminus \{u\}, \quad N'(v) \cap N'(c) = N(v) \cap N(c)$$

and (9) follows. □

Definition 7 *Let Γ be a graph and let x, y, u, v be distinct point of Γ such that $\{x, y\}$, $\{u, v\}$ are edges of Γ. We say that edge $\{u, v\}$ dominates edge $\{x, y\}$ if each of the following holds*

(i) *$\{u, x\}$ or $\{u, y\}$ is not edge of Γ,*
(ii) *$\{v, x\}$ or $\{v, y\}$ is not edge of Γ,*
(iii) *$N(x) \cap N(y) \subseteq N(u) \cap N(v)$.*

The edge dominance in Definition 7 can be seen in figure 8. The edge dominance defined here will be referred as long edge dominance. Let Θ be the subgraph of Γ spanned by the set of vertices $\{x, y, u, v\}$. In the situation described in the definition $\{x, y\}$, $\{u, v\}$ are edges of Θ. Then any of the remaining pairs $\{x, u\}$, $\{x, v\}$, $\{y, u\}$, $\{y, v\}$ is either an edge of Θ or not. Out of the overall 16 cases conditions (i) and (ii) hold in 7 cases and do not hold in 9 cases.

Let us use the notation Γ' for the graph we get from Γ after deleting the edge $\{x, y\}$. (We do not delete the nodes x or y.)

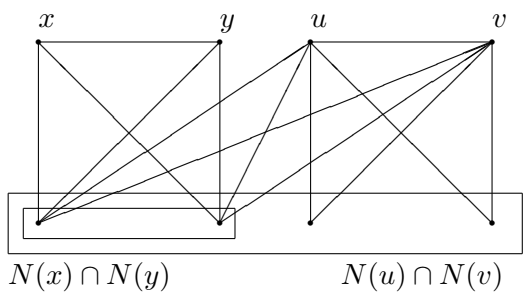

Figure 8. Edge $\{x, y\}$ is dominated by edge $\{u, v\}$. The dominance is long.

Proposition 15 *If Γ contains a clique of size k, then so does Γ'.*

Proof. Let Δ be a clique of size k in Γ. If x is not a node of Δ, then edge $\{x, y\}$ is not an edge of Δ. Consequently, Δ is a clique of size k in Γ' too. Similarly, if y is not a node of Δ, then Δ is a clique of size k in Γ' again. In both cases Γ' contains a clique of size k, as required. For the rest of the proof we may assume that x, y are nodes of Δ. The nodes of Δ that are distinct from x and y are coming from $N(x) \cap N(y)$ and so $\Delta_1 = \Delta \setminus (\{x\} \cup \{y\})$ is a clique of size $k-2$ in $N(x) \cap N(y)$. Using $N(x) \cap N(y) \subseteq N(u) \cap N(v)$ we can see that Δ_1 is a clique of size $k-2$ in $N(u) \cap N(v)$. Each node of Δ_1 is connected to u and v. This means that $\Delta_2 = \Delta_1 \cup \{u\} \cup \{v\}$ is a clique in Γ'. We will show that $u \notin \Delta_1$, $v \notin \Delta_1$ This will provide that Δ_2 is a clique of size k in Γ'.

In order to prove that $u \notin \Delta_1$ assume on the contrary that $u \in \Delta_1$. Plainly $u \in \Delta$. Using the fact that $\{x, y\}$ is an edge of Δ we get that $\{u, x\}$, $\{u, y\}$ are edges of Δ. Then $\{u, x\}$, $\{u, y\}$ are edges of Γ. But this is not the case. Thus $u \notin \Delta_1$. A similar argument gives that $v \notin \Delta_1$. $\square$

Proposition 16 *If node $\{r, s\}$ is dominated by edge $\{x, y\}$ and node $\{x, y\}$ is dominated by edge $\{u, v\}$, then node $\{r, s\}$ is dominated by edge $\{u, v\}$.*

Proof. It is enough to verify that

$$\{u, r\} \text{ or } \{u, s\} \text{ is not an edge of } \Gamma \tag{10}$$

$$\{v, r\} \text{ or } \{v, s\} \text{ is not an edge of } \Gamma \tag{11}$$

$$N(r) \cap N(s) \subseteq N(u) \cap N(v) \tag{12}$$

In order to prove (10) assume on the contrary that $\{u, r\}$, $\{u, s\}$ are edges of Γ. This means that $u \in N(r) \cap N(s)$. As edge $\{r, s\}$ is dominated by edge $\{x, y\}$, it holds $N(r) \cap N(s) \subseteq N(x) \cap N(y)$ and so $u \in N(x) \cap N(y)$. As edge $\{x, y\}$ is dominated by edge $\{u, v\}$, it holds that $\{u, x\}$ or $\{u, y\}$ is not an edge of Γ and so $u \notin N(x) \cap N(y)$. This contradiction implies (10), as required. The statement (11) can be verified in a similar way.

To prove (12) note that $N(r) \cap N(s) \subseteq N(x) \cap N(y)$ since edge $\{r, s\}$ is dominated by edge $\{x, y\}$. Further, $N(x) \cap N(y) \subseteq N(u) \cap N(v)$ since edge $\{x, y\}$ is dominated by edge $\{u, v\}$. $\square$

We would like to point out that the dominance concepts we introduced are not independent of each other. The dominance of nodes and the (short) dominance of edges are related in the following way. Let us suppose that edge $\{a, u\}$ is (short) dominated by edge $\{u, b\}$ in Γ. Working in $N(u)$ one can see that node a is (node) dominated by b.

The short and long dominance of edges are related in the next way. Let us suppose that edge $\{x, y\}$ is (long) dominated by edge $\{u, v\}$ in Γ. Assume that the nodes y and u are identical but the nodes x, y, v are distinct. By the definition of long dominance

$$\{x, y\}, \quad \{u, v\} \text{ are edges of } \Gamma, \tag{13}$$

$$\{x, u\} \text{ or } \{y, u\} \text{ is not an edge of } \Gamma, \tag{14}$$

$$\{x, v\} \text{ or } \{y, v\} \text{ is not an edge of } \Gamma, \tag{15}$$

$$N(x) \cap N(y) \subseteq N(u) \cap N(v). \tag{16}$$

For the sake of clarity let us denote the identical y and u by z and set $x = a$, $v = b$. Now conditions (13), (14), (15), (16) reduce to

$$\{a, z\}, \quad \{z, b\} \text{ are edges of } \Gamma, \tag{17}$$

$$\{a, z\} \text{ or } \{z, z\} \text{ is not an edge of } \Gamma, \tag{18}$$

$$\{a, b\} \text{ or } \{z, b\} \text{ is not an edge of } \Gamma, \tag{19}$$

$$N(a) \cap N(z) \subseteq N(z) \cap N(b). \tag{20}$$

As Γ does not have any loop, $\{z, z\}$ cannot be an edge of Γ and so condition (18) drops out. By (17), $\{z, b\}$ is an edge of Γ and so by (19), $\{a, b\}$ is not an edge of Γ. We are left with the following conditions

$$\{a, z\}, \quad \{z, b\} \text{ are edges of } \Gamma, \tag{21}$$

$$\{a, b\} \text{ is not an edge of } \Gamma, \tag{22}$$

$$N(a) \cap N(z) \subseteq N(z) \cap N(b). \tag{23}$$

Therefore edge $\{a, z\}$ is (short) dominated by edge $\{z, b\}$.

We consider the following problem.

Problem 9 *Given a node a of Γ. Find all nodes u of Γ for which there is a node b of Γ such that the edge $\{a, u\}$ is (short) dominated by the edge $\{u, b\}$.*

We describe an algorithm that solves this problem. The algorithm maintains four lists L_1, L_2, L_3, L_4. Initially, L_1 contains the neighbors of a and L_2 contains the non-neighbors of a. (We do not place a to L_2.) Pick a $u \in L_1$. Then fill in L_3 with the elements of $L_1 \cap N(u)$ and fill in L_4 with the elements of $N(u) \cap L_2$.

Proposition 17 *If there is a $b \in L_4$ such that $\{b, c\}$ is an edge of Γ for each $c \in L_3$, then edge $\{a, u\}$ is (short) dominated by edge $\{u, b\}$.*

Proof. We will verify that

$$\{a, u\}, \quad \{u, b\} \text{ are edges of } \Gamma, \tag{24}$$

$$\{a, b\} \text{ is not an edge of } \Gamma, \tag{25}$$

$$N(a) \cap N(u) \subseteq N(u) \cap N(b). \tag{26}$$

Using $u \in L_1$ and $L_1 = N(a)$ we get that $\{a, u\}$ is an edge of Γ. Using $b \in L_4$ and the definition of L_4 we get that $\{u, b\}$ is an edge of Γ. These prove (24).

From $b \in L_4$ it follows that $b \in L_2$. The elements of L_2 are not adjacent to a and so $\{a, b\}$ is not an edge of Γ. This settles (25).

To prove (26) choose a $c \in N(a) \cap N(u)$. In particular it holds that $c \in N(a)$. Using $N(a) \cap N(u) = L_1 \cap N(u) = L_3$ we get that $c \in L_3$. By the definition of b, $\{c, b\}$ is an edge of Γ and so $c \in N(b)$. Combining $c \in N(a)$ and $c \in N(b)$ implies $c \in N(u) \cap N(b)$, as required. □

Let s be the number of times the above algorithm reads single entries of the adjacency matrix of Γ during execution and let n be the number of nodes of Γ.

Proposition 18 $s \leq n + n^2 + (2/9)n^3$

Proof. Let n_1 be the length of L_1 and let n_2 be the length of L_2. Note that $n_1 + n_2 = n - 1$. The lists L_1 and L_2 can be filled in using n looks up of the adjacency matrix. For a fixed u the lists L_3 and L_4 can be filled in using at most n look ups. Clearly the length of L_3 is at most n_1 and the length of L_4 is at most n_2 in any time during the execution. For $u \in L_1$ there are at most n_1 choices. For $b \in L_4$ there are at most n_2 choices. For $c \in L_3$ there are at most n_1 choices. (We included figure 9 to assist in following the argument.)

An upper estimate for s is

$$n + n_1(n + n_1 n_2) = n + n_1 n + n_1^2 n_2 \leq n + n^2 + n_1^2 n_2.$$

The solution of the optimization problem, maximize $n_1^2 n_2$ subject to $n_1 + n_2 = n - 1$, is $n_1 = (2/3)(n-1)$, $n_2 = (1/3)(n-1)$. Therefore $s \leq n + n^2 + (2/9)(n-1)^3 \leq n + n^2 + (2/9)n^3$. □

We formulated Problem 9 with the definite purpose to incorporate the algorithm that solves it into the Carraghan-Pardalos clique search algorithm. But in the same time the algorithm above can be used to inspect the initial adjacency matrix to spot edges to delete. When we spot a deletable edge we delete it immediately. This might look as a rather simple minded approach. The deleted edge may dominate a number of further edges and so these edges will not be deleted later. We included Proposition 14 to reassure the reader that in fact this is not going to happen. By Proposition 18, the complete inspection looks up entries of the adjacency matrix at most $n^2 + n^3 + (2/9)n^4$ times.

8. Normal partitions

Let Γ be a graph with vertex set V.

Definition 8 *A partition $C_1, \ldots, C_t$ of V is called a normal partition of Γ if for each i, $1 \leq i \leq t$ there is an element $c_i \in C_i$ such that the elements of $C_i \setminus \{c_i\}$ are not adjacent to c_i and the elements of $C_{i+1} \cup \cdots \cup C_t$ are adjacent to c_i.*

The element c_i is called the leading element of the class C_i.

Consider the graph Γ in the first example. We remind the reader that figure 1 depicts a geometrical representation of Γ and table 1 contains two forms of the adjacency matrix of Γ. Let

$$\begin{array}{lll} C_1 = \{1, 3, 6\}, & C_2 = \{4, 2\}, & C_3 = \{5\}, \\ c_1 = 1, & c_2 = 4, & c_3 = 5. \end{array}$$

It is a routine exercise to check that C_1, C_2, C_3 is a normal partition of Γ with leading elements c_1, c_2, c_3. The list

$$(1)\ \ 3\ \ 6\ \ ||\ \ (4)\ \ 2\ \ ||\ \ (5)$$

could represent this partition in a more concise way.

Consider the graph Γ in the second example. For the reader convenience figure 6 presents Γ in a geometric form with points and arcs while table 2 presents the adjacency matrix of Γ. Set

$$\begin{array}{lll} C_1 = \{2, 1, 5, 9\}, & C_2 = \{8, 4, 6\}, & C_3 = \{3, 7\}, \\ c_1 = 2, & c_2 = 8, & c_3 = 3. \end{array}$$

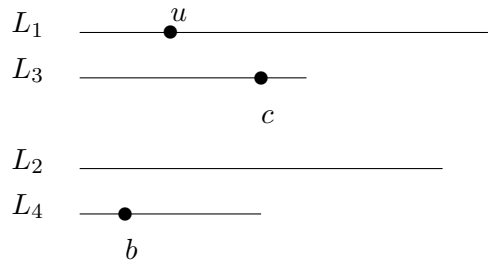

Figure 9. The lists L_1, L_2, L_3, L_4 in the proof of Proposition 18.

One can verify that C_1, C_2, C_3 is a normal partition of Γ with leading elements c_1, c_2, c_3. The list

$$(2)\ \ 1\ \ 5\ \ 9\ \ ||\ \ (8)\ \ 4\ \ 6\ \ ||\ \ (3)\ \ 7$$

could also be used to record this partition.

The normal partition $C_1, \ldots, C_t$ of Γ can be viewed as a (σ, t)-quasi coloring of Γ. The adjacency matrix $\mathbf{M}$ of Γ is cut into blocks $\mathbf{M}_{i,j}$ as we have seen in Section 5. Here 2σ is the overall number of ones in the $\mathbf{M}_{i,i}$ blocks and $\{a_1, b_1\}, \ldots, \{a_\sigma, b_\sigma\}$ are the edges of Γ corresponding to these ones. (Two 1's in symmetrical positions with respect to the main diagonal represent the same edge.) The subgraph of Γ spanned by $N(a_i) \cap N(b_i)$ is denoted by Γ_i. Let Γ' be the graph we get from Γ after deleting each of the edges $\{a_1, b_1\}, \ldots, \{a_\sigma, b_\sigma\}$. (But not deleting any of the end points of the edges.)

Proposition 19 *If Γ has a clique of size k, then at least one of $\Gamma_1, \ldots, \Gamma_\sigma$ has a clique of size $k-2$ or Γ' has a clique of size k.*

Proof. Assume that none of $\Gamma_1, \ldots, \Gamma_\sigma$ has a clique of size $k-2$. Let Δ be a clique of size k of Γ. If for some i, $1 \le i \le \sigma$, $\{a_i, b_i\}$ is an edge of Δ, then Γ_i contains a clique of size $k-2$. This is not the case so $\{a_i, b_i\}$ is not an edge of Δ for each i, $1 \le i \le \sigma$. After deleting these edges from Γ the subgraph Δ is still a clique of size k in the remaining graph, that is, Δ is a clique in Γ'. □

Proposition 20 *The size of the maximum clique in Γ' is t.*

Proof. As the elements of $C_2 \cup \cdots \cup C_t$ are adjacent to c_1 and $c_2, \ldots, c_t \in C_2 \cup \cdots \cup C_t$, it follows that c_1 is adjacent to each of $c_2, \ldots, c_t$. Similarly, as the elements of $C_3 \cup \cdots \cup C_t$ are adjacent to c_2 and $c_3, \ldots, c_t \in C_3 \cup \cdots \cup C_t$, it follows that c_2 is adjacent to each of $c_3, \ldots, c_t$. Continuing in this way finally we have that any two distinct nodes among $c_1, \ldots, c_t$ are adjacent. Thus there is a clique of size t in Γ'.

Let $\mathbf{M}'$ be the adjacency matrix of Γ'. Note that the blocks $\mathbf{M}'_{i,i}$ of $\mathbf{M}'$ are zero matrices and so Γ' can be well colored using t colors. Consequently Γ' cannot have a clique of size $t+1$. □

Problem 10 *Given a graph Γ. Construct a normal partition $C_1, \ldots, C_t$ for Γ.*

One can construct a normal partition of a given graph using the following greedy procedure. Let V be the vertex set of the given graph Γ. Set $i = 0$, $L = V$. While $L \neq \emptyset$ choose an $a \in L$ for which $|N(a)|$ is the largest possible. Increment i by one and set $c_i = a$, $C_i = L \setminus N(a)$, $L = N(a)$.

By Proposition 19, when we are looking for a k-clique in Γ we should check if the graphs $\Gamma_1, \ldots, \Gamma_\sigma$ has a clique of size $k-2$ or the graph Γ' has a clique of size k. If $k = t$, then by Proposition 20, Γ' has a clique of size k. If $k > t$, then again by Proposition 20, we do not need to check Γ' at all. Thus we need to work only with the graphs $\Gamma_1, \ldots, \Gamma_\sigma$.

The normal partition $C_1, \ldots, C_t$ helps us to detect dominated nodes. To see how let $C_1 = \{a_1, \ldots, a_s\}$, where $a_1 = c_1$. Consider the block $\mathbf{M}_{1,1}$ of the adjacency matrix $\mathbf{M}$ of Γ. The first row and the first column of $\mathbf{M}_{1,1}$ is filled with zeros as a_1 is not adjacent to $a_2, \ldots, a_s$. If the second row and second column is filled with zeros, then node a_1 dominates a_2. Indeed, a_1 and a_2 are not adjacent. Further $N(a_2) \subseteq N(a_1)$ holds as $c_1 = a_1$ is adjacent to the elements of $C_2 \cup \cdots \cup C_t$. In short if the second row of $\mathbf{M}_{1,1}$ contains only zeros, then the node a_2 can be canceled from the graph Γ when we are looking for a k-clique in Γ.

Suppose there are $2r$ non-zero elements in the second row and the second column of the matrix $\mathbf{M}_{1,1}$ and $\Gamma_1, \ldots, \Gamma_r$ are the graphs among $\Gamma_1, \ldots, \Gamma_\sigma$ that correspond to ones in the second row and second column of $\mathbf{M}_{1,1}$. After completing the clique search for the graphs $\Gamma_1, \ldots, \Gamma_r$ we can delete the ones from the second row and second column of $\mathbf{M}_{1,1}$ for the remaining part of the work. Therefore the node a_2 can be deleted from the graphs $\Gamma_{r+1}, \ldots, \Gamma_\sigma$ to get smaller graphs to work with. We can repeat this reduction at any time when the non-zero elements in a given row and in the corresponding column of the matrix are completely processed. In fact these reductions can be carried out at the beginning of the computation and we end up with a large collection of graphs that can be processed independently of each other.

References

[1] I M Bomze M Budinich P M Pardalos M Pelillo *The Maximum Clique Problem* Handbook of Combinatorial Optimization Vol 4 Kluwer Academic Publishers 1999
[2] S Butenko W E Wilhelm Clique-detection models in computational biochemistry and genomics *European Journal of Operational Research* **173** (2006) 1–17
[3] D S Johnson M A Trick *Cliques Colorings and Satisfiability* 2nd DIMACS Implementation Challenge American Mathematical Society 1996
[4] J Hasselberg P M Pardalos V Vairaktarakis Test case generators and computational results for the maximum clique problem *J Global Optim* **3** (1993) 463–482
[5] P Kaski P R J Östergård *Classification Algorithms for Codes and Designs* Springer 2006
[6] D E Knuth Dancing links in *Millennial Perspectives in Computer Science* J Davies B Roscoe J Woodcock Eds Palgrave Macmillan Basingstoke 2000 pp 187–214
[7] P M Pardalos J Xue The maximum clique problem *J. Global Optim* **4** (1994) 301–328
[8] `ftp://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/cliques/` (DIMACS clique benchmark instances)
[9] L Babel Finding maximum cliques in arbitrary and in special graphs *Computing* **46** (1991) 321–341
[10] E Balas J Xue Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring *Algorithmica* **15** (1996) 397–412
[11] R Carragham P M Pardalos An exact algorithm for the maximum clique problem *Oper Res Lett* **9** (1990) 375–382
[12] J Konc D Janežič An improved branch and bound algorithm for the maximum clique problem *MATCH Communications in Mathematical and Computer Chemistry* **58** (2007) 569–590
[13] D Kumlander A new exact algorithm for the maximum-weight clique problem based on a heuristic vertex-coloring and back-track search *Proceeding of The Fourth International Conference on Engineering Computational Technology* Civil-Comp Press 2004
[14] P R J Östergård A fast algorithm for the maximum clique problem *Discrete Applied Mathematics* **120** (2002) 197–207
[15] E C Sewell A branch and bound algorithm for the stability number of a sparse graph *INFORMS J Comput* **10** (1998) 438–447
[16] E Tomita T Seki An efficient branch-and-bound algorithm for finding a maximum clique *Lecture Notes in Computer Science* 2631 (2003) 278–289
[17] D R Wood An algorithm for finding a maximum clique in a graph *Operations Research Letters* **21** (1997) 211–217
[18] `ftp://dimacs.rutgers.edu/pub/challenge/graph/solvers/` (The `dfmax` clique solver)

[19] P M Pardalos J Rappe M G C Resende An exact parallel algorithm for the maximum clique problem in *High Performance Algorithms and Software in Nonlinear Optimization* R De Leone A Murli P M Pardalos G Toraldo (eds) Kluwer 279–300 Dordrecht 1998

[20] Y Zhang E J Chester M A Langston On finding bicliques in bipartite graphs: A novel approach with applications to the integration of diverse biological data types *Proceedings of the 41st Hawaii International Conference on System Sciences* (2008)

[21] M Molloy B Reed *Graph Coloring and The Probabilistic Method* Springer 2002