From matchings to independent sets

Vadim Lozin

Mathematics Institute, University of Warwick, Coventry CV4 7AL, UK.

Abstract

In 1965, Jack Edmonds proposed his celebrated polynomial-time algorithm to find a maximum matching in a graph. It is well-known that finding a maximum matching in G is equivalent to finding a maximum independent set in the line graph of G. For general graphs, the maximum independent set problem is NP-hard. What makes this problem easy in the class of line graphs and what other restrictions can lead to an efficient solution of the problem? In the present paper, we analyze these and related questions. We also review various techniques that may lead to efficient algorithms for the maximum independent set problem in restricted graph families, with a focus given to augmenting graphs and graph transformations. Both techniques have been used in the solution of Edmonds to the maximum matching problem, i.e. in the solution to the maximum independent set problem in the class of line graphs. We survey various results that exploit these techniques beyond the line graphs.

Keywords: maximum matching; maximum independent set; polynomial algorithm

1 Introduction

In a graph, a *matching* is a set of edges no two of which share a vertex. The MAXIMUM MATCHING problem asks to find in a graph a matching of maximum size. The key idea to solve the problem was proposed in 1957 by Claude Berge [9] who showed that a matching M in a graph G is maximum if and only if G contains no augmenting path with respect to M. However, the question of the complexity of finding augmenting paths remained open until 1965, when Jack Edmonds proposed a polynomial-time algorithm to solve this problem [14]. Lovász and Plummer in their book "Matching Theory" refer to the solution of Edmonds as "one of the most involved of combinatorial algorithms" [28].

The MAXIMUM MATCHING problem is a special case of a more general problem, known as MAXIMUM INDEPENDENT SET. In a graph, an *independent set* is a set of vertices no two of which are adjacent, and the MAXIMUM INDEPENDENT SET problem is the problem of finding in a graph an independent set of maximum size. A relationship between the two problems can be established through the notion of line graph. The *line graph* of a graph G, denoted $L(G)$, is the graph whose vertices represent the edges of G with two vertices being adjacent if and only if the corresponding edges of G share a vertex. Therefore, a set of vertices of $L(G)$ is independent if and only if the corresponding edges of G form a matching. This correspondence between the two problems together with the matching algorithm of Edmonds show that finding a maximum independent set in a line graph is a polynomially solvable task. On the other hand, for general graphs the MAXIMUM INDEPENDENT SET problem is known to be NP-hard.

What makes the MAXIMUM INDEPENDENT SET problem easy in the class of line graphs and what other restrictions can lead to an efficient solution for this problem? We analyze these and related questions in Section 3. Then in Section 4 we review various techniques that may lead to efficient algorithms for the MAXIMUM INDEPENDENT SET problem in restricted graph families, with a focus given to augmenting graphs and graph transformations. Both techniques have been used in the solution of Edmonds to the MAXIMUM MATCHING problem, i.e. in the solution to the MAXIMUM INDEPENDENT SET problem in the class of line graphs. We survey various results that exploit these techniques beyond the line graphs.

We start our journey to the MAXIMUM INDEPENDENT SET problem in Section 2 with some preliminary information and conclude it in Section 5 with a number of open questions.

2 Preliminaries

In this section, we introduce basic definitions and notation used in the paper and present some information about the MAXIMUM INDEPENDENT SET problem and about hereditary classes of graphs.

2.1 Definitions and notation

Given a graph G, we denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. The *neighbourhood* $N(x)$ of a vertex $x \in V(G)$ is the set of vertices adjacent to x and the *degree* of x is the size of its neighbourhood. Graphs all of whose vertices have degree 3 are known as *cubic* and graphs of vertex degree at most 3 are known as *subcubic*.

If $U \subset V(G)$, then $G[U]$ is the subgraph of G induced by U, i.e. the graph with vertex set U in which two vertices are adjacent if and only if they are adjacent in G. Also, $N(U)$ is the neighbourhood of U, i.e. the set of vertices outside of U that have at least one neighbour in U.

As usual, C_n, P_n and K_n denote the chordless cycle, the chordless path and the complete graph with n vertices, respectively. Also, $K_{n,m}$ denotes the complete bipartite graph with parts of size n and m. The graph $K_{1,3}$ is known in the literature as the *claw*. For two graphs G and H, we denote by $G + H$ their disjoint union. Also, mG is the disjoint union of m copies of G.

A *clique* in a graph is a set of pairwise adjacent vertices. In other words, a set $U \subseteq V(G)$ is a clique in G if and only if U is an independent set in the complement of G. The size of a maximum clique in G is called the *clique number* of G.

A *vertex cover* in G is a subset of vertices containing at least one endpoint of each edge of the graph. Clearly, $U \subseteq V(G)$ is a vertex cover if and only if $V(G) - U$ is an independent set. The size of a minimum vertex cover in G is denoted $\tau(G)$ and the size of a maximum independent set, known as the *independence number* of G, is denoted $\alpha(G)$. Therefore, $\tau(G) + \alpha(G) = |V(G)|$.

The subdivision of an edge is the operation of introducing a new vertex on the edge.

2.2 The maximum independent set problem

Finding a maximum independent set in a graph is one of the central problems of combinatorial optimization and theoretical computer science with numerous applications and various connections to other problems. For instance, as we mentioned already, MAXIMUM CLIQUE, MINIMUM VERTEX COVER, MAXIMUM MATCHING admit easy reductions to MAXIMUM INDEPENDENT SET. Also, finding a maximum induced matching (also known as a 1-regular graph) in G is equivalent to finding a maximum independent set in the square of the line graph of G. In addition, in [13] it was shown that the weighted version of the MAXIMUM INDEPENDENT SET problem, also known as VERTEX PACKING, is equivalent to maximizing a pseudo-Boolean function, i.e. a real-valued function with Boolean variables. Notice that pseudo-Boolean optimization is a general framework for a variety of problems of combinatorial optimization such as MAX-SAT or MAX-CUT [10].

Among various applications of the MAXIMUM INDEPENDENT SET problem we distinguish two examples. The origin of the first one is the area of computer vision and pattern recognition, where one of the central problems is the matching of relational structures. In graph theoretical terminology, this is the GRAPH ISOMORPHISM, or more generally, MAXIMUM COMMON SUBGRAPH problem. It reduces to the MAXIMUM CLIQUE problem by associating with a pair of graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ a special graph $G = (V, E)$ (known as the *association graph* [32]) with vertex set $V = V_1 \times V_2$ so that two vertices $(i, j) \in V$ and $(k, l) \in V$ are adjacent in G if and only if $i \neq k$, $j \neq l$ and $ik \in E_1 \Leftrightarrow jl \in E_2$. Then a maximum common subgraph of the graphs G_1 and G_2 corresponds to a maximum clique in G and hence to a maximum independent set in the complement of G.

Another example comes from information theory. The graph theoretical model arising here can be roughly described as follows. An information source sends messages in the alphabet $X = \{x_1, x_2, \ldots, x_n\}$. Along the transmission some symbols of X can be changed to others because of random noise. Let G be a graph with $V(G) = X$ and $x_i x_j \in E(G)$ if and only if x_i and x_j can be interchanged during the transmission. Then a noise-resistant code should consist of the symbols of X that constitute an independent set in G. Therefore, a largest noise-resistant code corresponds to a largest independent set in G.

As we mentioned earlier, computationally the MAXIMUM INDEPENDENT SET problem is difficult, i.e. it is NP-hard. This motivates looking at restricted versions of the problem, such as finding an approximate solution or solving the problem for graphs in particular classes. In the present paper, we focus on the second issue. All classes studied here are hereditary. A brief overview of this notion is given in the next section.

2.3 Hereditary classes of graphs

A class X of graphs is *hereditary* if it is closed under taking induced subgraphs. In other words, X is hereditary if $G \in X$ implies $G - v \in X$ for every vertex $v \in V(G)$. The family of hereditary classes includes many classes of theoretical or practical importance, such as planar graphs, perfect graphs, line graphs, etc. Many important classes that are not hereditary have natural hereditary extensions. For instance, the minimal hereditary extension of the set of trees consists of all forests, i.e. graphs without cycles, and the minimal hereditary extension of the cubic graphs consists of all subcubic graphs.

An important feature of hereditary (and only hereditary) classes is that they admit so called induced subgraph characterizations. To make things more precise, let us denote by

$Free(M)$ the class of all graphs containing no induced subgraphs from the set M.

It is not difficult to see that a class X is hereditary if and only if $X = Free(M)$ for a set M, in which case graphs in M are called *forbidden induced subgraphs* for the class X. In general, all graphs that are not in X are forbidden induced subgraphs for X. However, to describe X it is sufficient to indicate only *minimal* forbidden induced subgraphs. It is known that for every hereditary class the set of minimal forbidden induced subgraphs is unique. For instance, for the class of forests this set consists of all chordless cycles.

Of particular interest in this paper are classes defined by finitely many forbidden induced subgraphs. We call such classes *finitely defined*. If M consists of a single graph, then the class $Free(M)$ is called *monogenic*. Examples of finitely defined classes include subcubic graphs and line graphs. For the class of subcubic graphs, the set of minimal forbidden induced subgraphs consists of 11 graphs and for the class of line graphs the set of minimal forbidden induced subgraphs consists of 9 graphs [8]. It is an easy exercise to verify that one of the minimal non-line graphs is the claw.

The induced subgraph characterization of hereditary classes provides a uniform tool to describe them, and hence, a systematic way to study various problems on hereditary classes. In the next section, we study the MAXIMUM INDEPENDENT SET problem.

3 The maximum independent set problem on hereditary classes of graphs

Being NP-hard in general, the MAXIMUM INDEPENDENT SET problem remains difficult under substantial restrictions, for instance for triangle-free, subcubic or planar graphs. On the other hand, for graphs in some particular classes the problem admits polynomial-time solutions. One of the most remarkable results of this type is the matching algorithm that solves the MAXIMUM INDEPENDENT SET problem in the class of line graphs. This solution raises the following natural question.

3.1 What makes the maximum independent set problem easy in the class of line graphs?

We claim that the answer to this question is the claw-freeness of line graphs. Indeed, if we remove the claw from the set of minimal non-line graphs, we obtain a class containing all triangle-free graphs (since each of the remaining minimal non-line graphs contains a triangle), where the problem is known to be NP-hard (see e.g. [31]). Therefore, the claw-freeness is a necessary condition for polynomial-time solvability of the problem in the case of line graphs. On the other hand, this condition is also sufficient, because by forbidding the claw alone we obtain a class where the problem can be solved in polynomial time, which was proved by Minty [29] and Sbihi [33] independently of each other. This discussion explains *what* makes the

MAXIMUM INDEPENDENT SET problem easy in the class of line graphs, but it does not explain *why*.

3.2 Why the claw and what other restrictions can make the problem easy?

To answer these questions, we need to recall a couple of facts about the complexity of the MAXIMUM INDEPENDENT SET problem in particular classes of graphs.

Fact 1. *The* MAXIMUM INDEPENDENT SET *problem is NP-hard for graphs of vertex degree at most 3.*

Fact 2. *A double subdivision of an edge increases the independence number of the graph by exactly 1.*

Both facts are well-known. However, what is less known is that both of them can be derived from the same argument proposed by Alekseev in [2]. To explain this argument, let us introduce the following operation.

Definition 1. For a graph G and a vertex $x \in V(G)$, *vertex splitting* is a transformation shown in Figure 1, where $Y \cup Z$ is an arbitrary partition of the neighbourhood of x into two subsets, and y and z are new vertices.

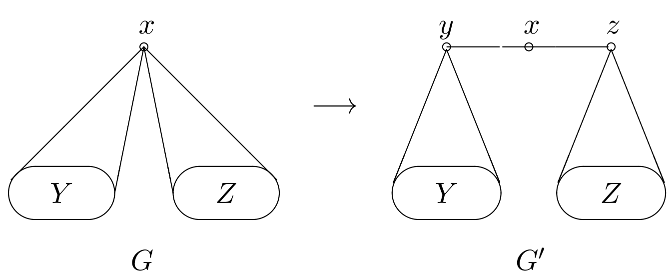


Figure 1: Vertex splitting

The importance of this operation is due to the following claim, which is easy to see.

Claim 1. *If G' is a graph obtained from a graph G by vertex splitting, then $\alpha(G) = \alpha(G') - 1$.*

If x is a vertex of degree more than 3 in a graph G, then the application of vertex splitting with $|Y| = 2$ and $Z = N(x) - Y$ replaces x with three new vertices each of which has degree less than the degree of x. Repeated applications of this operation allow us to transform G into a graph of degree at most 3. Clearly, this transformation can be implemented in polynomial time, which shows Fact 1. Also, it is not difficult to see that the application of vertex splitting with $|Y| = 1$ is equivalent to a double subdivision of an edge incident to x, which shows Fact 2. From these two facts we derive the following natural conclusion.

Proposition 1. *For every fixed k, the* MAXIMUM INDEPENDENT SET *problem restricted to the class of $(C_3, C_4, \ldots, C_k)$-free graphs of vertex degree at most 3 is NP-hard.*

Indeed, if a graph G of vertex degree at most 3 contains short cycles (i.e. cycles of length at most k), then by subdividing the edges of these cycles we can reduce the problem (in polynomial time) to a graph without short cycles. This simple fact was observed by many researchers including, for instance, Murphy [31]. What was not observed by Murphy and by many other researchers is that with the very same transformation we can get rid of small induced copies of the graph H_k represented in Figure 2. This observation allows us to strengthen Proposition 1 as follows.

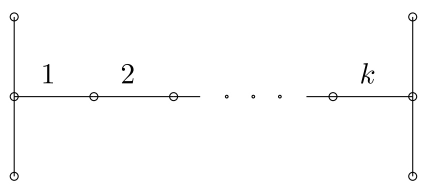


Figure 2: Graph H_k

Proposition 2. *For every fixed k, the* MAXIMUM INDEPENDENT SET *problem restricted to the class of $(C_3, C_4, \ldots, C_k, H_1, H_2, \ldots, H_k)$-free graphs of vertex degree at most 3 is NP-hard.*

Let us denote by

$\mathcal{S}_k$ the class of $(C_3, C_4, \ldots, C_k, H_1, H_2, \ldots, H_k)$-free graphs of vertex degree at most 3.

This defines an infinite sequence of graph classes $\mathcal{S}_3 \supset \mathcal{S}_4 \supset \ldots \supset \mathcal{S}_k \supset \ldots$, and by Proposition 2, the MAXIMUM INDEPENDENT SET problem is NP-hard in each class of this sequence. Throughout the paper, we denote by

$\mathcal{S}$ the intersection of all classes of the sequence $\mathcal{S}_3 \supset \mathcal{S}_4 \supset \ldots \supset \mathcal{S}_k \supset \ldots$.

We will say that the sequence $\mathcal{S}_3 \supset \mathcal{S}_4 \supset \ldots \supset \mathcal{S}_k \supset \ldots$ *converges* to $\mathcal{S}$ and that $\mathcal{S}$ is the *limit class* of the sequence.

It is not difficult to see that any graph G either belongs to *all* classes of the sequence converging to $\mathcal{S}$, in which case it belongs to $\mathcal{S}$, or to at most finitely many (possibly to no) classes of this sequence. Therefore:

Proposition 3. *If $X = Free(M)$ is a finitely defined class of graphs containing $\mathcal{S}$, then the* MAXIMUM INDEPENDENT SET *problem is NP-hard in X.*

Indeed, if $X = Free(M)$ contains $\mathcal{S}$, then none of the graphs in M belongs to $\mathcal{S}$. Therefore, if M is finite, then graphs in M belong to at most finitely many classes of the sequence converging to $\mathcal{S}$ and hence there must exist a class $\mathcal{S}_k$ contained in X, in which case the problem is NP-hard in X by Proposition 2.

The same is true for any other limit class, i.e. for the limit of any other sequence of graph classes in each of which the problem is NP-hard. Therefore, to solve the problem efficiently in a finitely defined class, one needs to destroy (to forbid at least one graph from) each limit class. In other words, limit classes play the role of "forbidden elements" for the family of finitely defined classes with polynomial-time solvable independent set problem. Similarly to the induced subgraph characterization of hereditary classes, only minimal limit classes are of interest, which justifies the following definition introduced in [3].

Definition 2. A minimal limit class is called a *boundary* class.

The importance of this notion for the MAXIMUM INDEPENDENT SET problem is due to the following theorem.

Theorem 1. *Unless $P = NP$, the* MAXIMUM INDEPENDENT SET *problem can be solved in polynomial time in the class defined by a finite set M of forbidden induced subgraphs if and only if $Free(M)$ contains none of the boundary classes (or equivalently, M contains at least one graph from each of the boundary classes).*

This theorem was proved by Alekseev in [3]. Moreover, in the same paper Alekseev proved that the class $\mathcal{S}$ is a minimal limit class, i.e. a boundary class for the problem.

From Theorem 1 it follows, in particular, that the problem can be solved in polynomial time in a finitely defined class $Free(M)$ *only if* M contains at least one graph from the class $\mathcal{S}$. This motivates us to look at the structure of graphs in the class $\mathcal{S}$.

Let us repeat that $\mathcal{S}$ is the limit of the sequence $\mathcal{S}_3 \supset \mathcal{S}_4 \supset \ldots \supset \mathcal{S}_k \supset \ldots$. Since $\mathcal{S}_k$ does not contain cycles of length up to k, the limit contains no cycles at all. Therefore, $\mathcal{S}$ is a class of forests. Since each class in the sequence converging to $\mathcal{S}$ contains graphs of vertex degree at most 3, $\mathcal{S}$ is a class of forests of vertex degree at most 3. Finally, the limit contains no graphs of the form H_k, and hence every connected graph in $\mathcal{S}$ has at most one vertex of degree 3. Therefore, $\mathcal{S}$ is the class of graphs every connected component of which has the form $S_{i,j,k}$ represented in Figure 3.

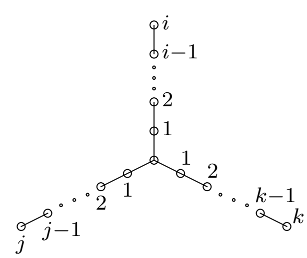


Figure 3: The graph $S_{i,j,k}$

One of the smallest non-trivial graphs of this form is $S_{1,1,1}$, i.e. the claw. This explains what is so special about the claw as a forbidden subgraph. Namely, it belongs to the boundary class $\mathcal{S}$. In the case of finitely many forbidden induced subgraphs, excluding a graph from $\mathcal{S}$ is a necessary condition for polynomial-time solvability of the problem. Whether this necessary condition is sufficient is a big open question, and an answer to this question depends on whether the class $\mathcal{S}$ is the *only* boundary class for the problem. We conjecture that

Conjecture 1. The class $\mathcal{S}$ is the unique boundary class for the MAXIMUM INDEPENDENT SET problem.

To prove this conjecture, one has to show that by forbidding *any* graph from $\mathcal{S}$ we obtain a class where the problem is polynomial-time solvable. Up to date, this was verified only for a few graphs in $\mathcal{S}$, the claw being one of them. But the class of claw-free graphs is not a maximal monogenic class, where the problem admits a polynomial-time solution. The result for claw-free graphs was generalized in a number of ways and below we present the most recent list of *all maximal monogenic* classes, where the MAXIMUM INDEPENDENT SET problem can be solved in polynomial-time.

3.3 Maximal monogenic classes with polynomial-time solvable independent set problem

Before we present the list, let us make the following observation. It is not difficult to see that if the MAXIMUM INDEPENDENT SET problem can be solved in polynomial time in a monogenic class $Free(G)$, then it also admits a polynomial-time solution in the class $Free(G+K_1)$. Therefore, any monogenic class $Free(G)$ can be trivially extended to a larger class by adding isolated vertices to G. Taking into account this observation, in what follows, we avoid monogenic classes where the only forbidden induced subgraph contains isolated vertices.

3.3.1 $S_{1,1,2}$-free graphs

The first non-trivial extension of the solution for claw-free graphs was proposed by Alekseev in [4]. It deals with the class of $S_{1,1,2}$-free (also known as $fork$-free or $chair$-free) graphs. What is important is that the solution of Alekseev generalizes not only the class but also the technique. Similarly to the claw-free and line graphs, Alekseev exploits the idea of augmenting graphs, but he goes beyond augmenting paths. He identifies other augmenting graphs in the class $Free(S_{1,1,2})$ and shows how to find them in polynomial time.

In addition to claw-free graphs, the class $Free(S_{1,1,2})$ extends another important monogenic class, where the MAXIMUM INDEPENDENT SET was known to be solvable in polynomial time, namely, P_4-free graphs. Also, the solution of Alekseev generalizes polynomial-time algorithms for some non-monogenic subclasses of $S_{1,1,2}$-free graph, such as $(chair, bull)$-free graphs [11].

An entirely different approach to the problem in the class of $S_{1,1,2}$-free graphs was proposed in [25]. It solves the problem in the more general setting where the vertices of the input graph are equipped with weights and the problem asks to find an independent set of maximum total weight.

3.3.2 $S_{0,2,2}$-free graphs

It is not difficult to see that $S_{0,2,2}$ is a P_5. Therefore, the $S_{0,2,2}$-free graphs are precisely the P_5-free graphs. The class of P_5-free graphs does not generalize the claw-free graphs, but for several decades the complexity of the MAXIMUM INDEPENDENT SET problem in this class was an open problem. Moreover, P_5 was the *only* minimal connected graph G such that the complexity of the problem in the class $Free(G)$ was open. It was solved for P_5-free graphs in [22]. This solution was preceded by numerous partial results dealing with subclasses of P_5-free graphs. One of them is the class $2K_2$-free graphs, where the problem was solved by Farber [16]. Later Farber's solution was extended to a more general class, which is the subject of the next section.

3.3.3 $mS_{1,0,0}$-free graphs

In more convenient notation, this is the class of mK_2-free graphs, where m is a fixed constant. A solution to the MAXIMUM INDEPENDENT SET problem for mK_2-free graphs is based on two results. First, graphs in this class have only polynomially many maximal (with respect to set inclusion) independent sets [7], and second, all of them can be enumerated in polynomial time [34].

3.3.4 $S_{1,1,1} + S_{1,0,0}$-free graphs

The graph $S_{1,1,1}+S_{1,0,0}$ is another extension of the claw and hence the class of $S_{1,1,1}+S_{1,0,0}$-free graphs is one more extension of the class of claw-free graph. In this class, the problem was solved in [26].

3.3.5 $2S_{1,1,0}$-free graphs

For $2S_{1,1,0}$-free (or $2P_3$-free) graphs a solution to the MAXIMUM INDEPENDENT SET problem was proposed in [27]. With a simple induction (not described in [27]) this solution can be extended to mP_3-free graphs for any value of m, generalizing both $2P_3$-free and mK_2-free graphs.

3.3.6 What is next?

There are several minimal monogenic classes of graphs for which the complexity of the problem is open. These are $S_{1,2,2}$-free graphs, $S_{1,1,3}$-free graphs, $P_4 + P_2$-free graphs and P_6-free graphs. The first two classes are of special interest, because they extend simultaneously two monogenic classes with polynomial-time solvable independent set problem, namely, the $fork$-free and P_5-free graphs.

4 Algorithmic techniques for the maximum independent set problem

The celebrated solution of the MAXIMUM MATCHING problem includes two main ingredients: augmenting paths and cycle shrinking. The first of them is a special case of the general approach to solve the MAXIMUM INDEPENDENT SET problem known as *augmenting graphs*. We review basic results related to this notion in Section 4.1.

Cycle shrinking is a graph transformation that applies to certain cycles of odd length $2k+1$ and reduces the size of a maximum matching by exactly k. Various other helpful reductions for the MAXIMUM INDEPENDENT SET problem can be found in the literature. We review many them in Section 4.2.

4.1 Augmenting graphs

Let G be a graph, S an independent set in G and $R = V(G) - S$. We say that the vertices in S are *white* and the vertices in R are *black*. Consider two subsets $W \subseteq S$ and $B \subseteq R$. Note that W is an independent set. If B also is an independent set such that $|B| > |W|$ and $N(B) \cap S \subseteq W$, then we say that the bipartite graph $H = G[W \cup B]$ is *augmenting for the set* S.

Clearly, if G contains an augmenting graph $H = G[W \cup B]$ for S, then S is not maximum, because $T := (S - W) \cup B$ is an independent set larger than S, in which case we say that T is obtained from S by *H-augmentation*. On the other hand, if S is not maximum and T is a larger independent set, then the bipartite subgraph of G induced by $T - S$ and $S - T$ is augmenting for S. This discussion leads to the following well-known result.

Theorem 2 (Augmenting Graph Theorem)**.** *An independent set S in a graph G is maximum if and only if there are no augmenting graphs for S.*

This theorem suggests the following general approach to find a maximum independent set in a graph G: begin with any independent set S in G and as long as S admits an augmenting graph H, apply H-augmentations to S. Unfortunately, the problem of finding augmenting graphs is generally NP-hard, as the MAXIMUM INDEPENDENT SET problem is NP-hard. However, for graphs in some special classes this approach can lead to polynomial-time algorithms.

> *Example.* Consider, for instance, the class of claw-free graphs. By definition, every augmenting graph is bipartite. Clearly, bipartite claw-free graphs have vertex degree at most 2. Therefore, every connected claw-free bipartite graph is either a path or an even cycle. Even cycles can never be augmenting, since they have equally many black and white vertices. For the same reason, paths with an even number of vertices cannot be augmenting. Therefore, every connected claw-free augmenting graph is a path P_k with odd k. Since the line graph of P_k is P_{k-1}, we conclude that the Augmenting Graph Theorem restricted to the class of line graphs coincides with Berge's Lemma on augmenting paths.

The above example shows that the structure of augmenting graphs in a particular class can be rather simple. However, the problem of finding (detecting) augmenting graphs is generally far from being trivial even for graphs of simple structure. As we mentioned earlier, in the case of augmenting paths, this problem was first solved in 1965 by Edmonds, but only within the class of line graphs. Later, in 1980, this solution was extended to claw-free graphs by Minty and Sbihi. Much later, in 2006, a polynomial-time algorithm for detecting augmenting paths was developed for $S_{1,2,3}$-free graphs [19]. This class provides a vast generalization of claw-free graphs, but the algorithm for detecting augmenting paths does not solve the MAXIMUM INDEPENDENT SET problem for $S_{1,2,3}$-free graphs, because this class contains other types of augmenting graphs.

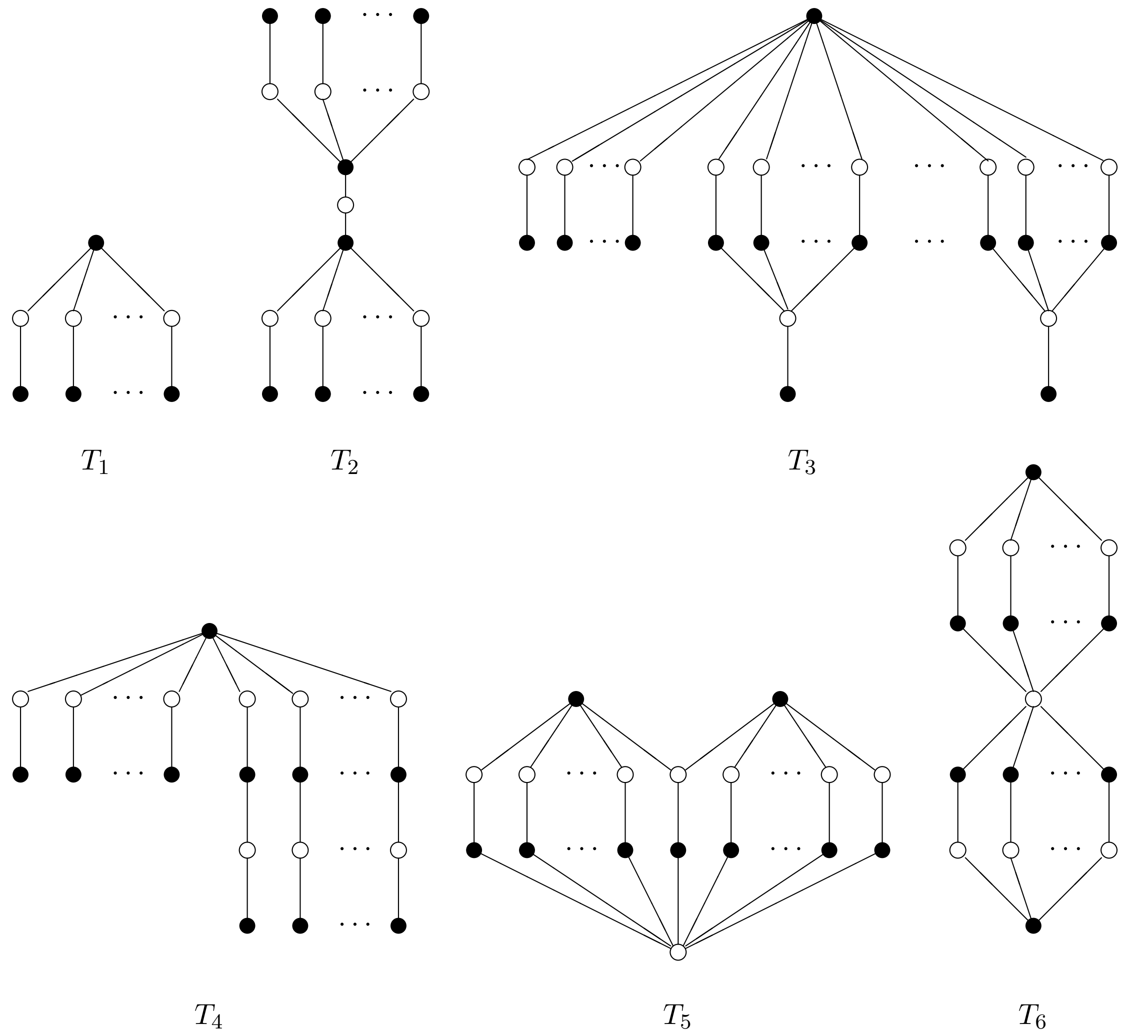


Figure 4: Some augmenting graphs

It is important to emphasize that augmenting paths constitute just one particular family of augmenting graphs. However, for several decades, since the solution of Edmonds, the idea of augmenting graphs did not see any development beyond augmenting paths. A breakthrough result was obtained in 1999 by Alekseev, who applied the idea of augmenting graphs to solve the problem for $S_{1,1,2}$-free graphs [4]. This result generalizes the solution of Minty and Sbihi for claw-free graphs (and hence the matching algorithm of Edmonds). But most importantly, it generalizes the technique. Alekseev showed that the class of $S_{1,1,2}$-free graphs contains two types of augmenting graphs (augmenting paths and so called complexes) and proposed polynomial-time algorithms to detect both of them.

One more type of augmenting graphs was discovered by Mosca [30], also in 1999. These are the graphs obtained from a star $K_{1,s}$ by subdividing each edge exactly once (the graph T_1 in Figure 4). Mosca called them *simple augmenting trees* and showed that this is the only family of augmenting graphs containing neither P_6 nor C_4. He also proposed a polynomial-time algorithm for detecting simple augmenting trees within (P_6, C_4)-free graphs, thus solving the MAXIMUM INDEPENDENT SET problem for (P_6, C_4)-free graphs.

Since 1999, many more types of augmenting graphs have been discovered in the literature (some of them are represented in Figure 4) and the augmenting graph technique has been repeatedly applied to solve the MAXIMUM INDEPENDENT SET problem in particular classes of graphs (see e.g. [5, 24]). Still, the area of augmenting graphs remains largely unexplored and its potential is far from being exhausted. We conclude this section with a couple of open questions in this area.

Question 1. Characterize the structure of $S_{1,2,2}$-free and $S_{1,1,3}$-free augmenting graphs.

We repeat that the complexity of the MAXIMUM INDEPENDENT SET problem in the classes of $S_{1,2,2}$-free and $S_{1,1,3}$-free graphs is an open question and characterizing the structure of augmenting graphs is the first step in resolving this question by means of augmenting graphs. The second step is developing algorithms for detecting all types of augmenting graphs in these classes. Clearly, both of them contain augmenting paths and, fortunately, in both of them augmenting paths can be detected in polynomial time, since augmenting paths can be found efficiently in $S_{1,2,3}$-free graphs [19]. Determining the complexity of finding augmenting paths in $S_{i,j,k}$-free graphs for larger values of i, j, k is another open questions that would be interesting to investigate.

Question 2. Determine the complexity of detecting augmenting paths in $S_{i,j,k}$-free graphs containing $S_{1,2,3}$.

4.2 Graph transformations

We have seen already a graph transformation helpful for the MAXIMUM INDEPENDENT SET problem, namely, the vertex splitting in Section 3.2. Now, let us look at this transformation in the reverse order, in which case it consists in deleting a vertex x of degree 2 with two non-adjacent neighbours and "folding" the neighbours of x. Implemented in this direction, the transformation is known as *vertex folding*. In spite of its simplicity, vertex folding is of great practical and theoretical importance. In particular, it was used in [17], along with two other reductions, to develop a simple and fast exact algorithm for the MAXIMUM INDEPENDENT SET problem. Even more importantly, the idea of vertex folding leads to a general approach to solve the problem known as *struction*.

4.2.1 Struction

Let $G = (V, E)$ be a graph, v_0 an arbitrary vertex of G and $\{v_1, \ldots, v_p\}$ the set of neighbours of v_0. The struction centered at v_0 is the graph transformation consisting of the following steps:

- remove the vertices $v_0, v_1, \ldots, v_p$ from G and denote the rest of the graph by R,
- add to R a set of new vertices $W = \{v_{i,j} : 1 \le i < j \le p \text{ and } v_i v_j \notin E\}$,
- join two new vertices $v_{i,j}$ and $v_{k,l}$ by an edge whenever $i \neq k$ or $v_j v_l \in E$,
- join every new vertex $v_{i,j} \in W$ to a vertex $u \in R$ by an edge whenever u is adjacent to v_i or to v_j in G.

We observe that the result of struction depends on the choice of vertex v_0 and the order of its neighbours. However, regardless of these choices, struction reduces the independence number by exactly 1.

Theorem 3. [13] *Let G^s be a graph obtained from G by struction, then $\alpha(G^s) = \alpha(G) - 1$.*

Clearly, if v_0 is a vertex of degree 2 with two non-adjacent neighbours, then struction coincides with vertex folding. Therefore, struction generalizes vertex folding. The crucial importance of this generalization is that it is applicable to *any* graph. Therefore, applying this transformation repeatedly to an n-vertex graph, in at most n iterations we can compute the independence number of the graph. Moreover, it is not difficult to see that every single application of struction can be implemented in time bounded by a polynomial in the number of vertices of the graph it applies to. This, however, does not lead to a polynomial-time algorithm, because with each application of struction the number of vertices can increase leading, in the worst case, to an overall exponential growth. This resembles the idea of resolution developed for the SATISFIABILITY problem. In the next section, we show that there is more in common between struction and resolution than a resemblance.

4.2.1.1 Struction versus resolution

Given a CNF formula, the *resolution rule* with respect to a variable x applies to two clauses C^1 and C^2, one containing x positively and the other containing x negatively. The *resolvent* of C^1 and C^2 is a new clause containing all literals of C^1 and C^2, except for x and $\overline{x}$.

The idea of resolution was proposed by Davis and Putnam in [12] and its importance can be seen through the following proposition.

Proposition 4. *Let F be a CNF formula and x a variable in F. If F^r is a CNF obtained from F by*

- *adding a resolvent for each pair of clauses one of which contains x positively and the other negatively,*
- *removing all clauses containing x (positively or negatively),*

then F^r is satisfiable if and only if F is.

Now let us translate Proposition 4 to the language of graph theory by means of the notion of incidence graph, which is defined as follows. Given a CNF formula F, we denote by G_F the graph containing a vertex for each clause and for each variable of F, and the edges connecting variables to clauses containing them (positively or negatively). We call G_F the *incidence graph* of F. In terms of incidence graphs, the transformation of a CNF formula presented in Proposition 4 can be described as follows.

Let G_F be the incidence graph of a CNF formula F, x a variable vertex of G_F and $\{C^1, \ldots, C^p\}$ the neighbourhood of X (i.e. the set of all clauses containing x, positively or negatively). Denote by G_{F^r} the graph obtained from G_F as follows:

– remove the vertices $x, C^1, \ldots, C^p$ from G_F and denote the rest of the graph by R,

– add to R a set of new vertices $W = \{C^{i,j} : C^i$ contains x and C^j contains $\overline{x}\}$,

– join every new vertex $C^{i,j} \in W$ to a vertex $u \in R$ by an edge whenever u is adjacent to C^i or to C^j in G_F.

It is not difficult to see that F transforms into F^r if and only if G_F transforms into G_{F^r}, i.e. the transformation $G_F \to G_{F^r}$ is a graph-theoretic description of the transformation $F \to F^r$.

This graph-theoretic interpretation of Proposition 4 reveals an amazing similarity (but not identity) between struction and resolution. Moreover, if we apply resolution on a variable which appears in the formula exactly twice, once positively and once negatively, then in terms of graphs resolution *coincides* with struction, in which case it becomes vertex folding.

4.2.1.2 Total struction

In [6], the idea of struction was generalized under the name *total struction* as follows.

Assume the vertices of a graph G are labelled by integers $1, 2, \ldots, n$, where $n = |V(G)|$. The vertex with maximum label in a subset A is denoted $m(A)$, and A^- is defined to be $A - \{m(A)\}$.

Given a graph $G = (V, E)$, an induced subgraph H of G, and a positive integer $p \leq \alpha(H)$, we define $R := V - N[V(H)]$ and associate with the triple (G, H, p) a graph $S(G, H, p)$ as follows:

- the vertex set of $S(G, H, p)$ is $R \cup W$, where W is the family of all independent sets of cardinality $p + 1$ in the subgraph of G induced by the vertices of $N[V(H)]$;
- the edge set of $S(G, H, p)$ consists of
  - the edges of the subgraph $G[R]$,
  - the edges linking vertices $A \in W$ and $B \in W$ whenever $A^- \neq B^-$ or $(m(A), m(B)) \in E(G)$,
  - the edges linking a vertex $A \in W$ to a vertex $v \in R$ whenever v has a neighbor in the subset A in the graph G.

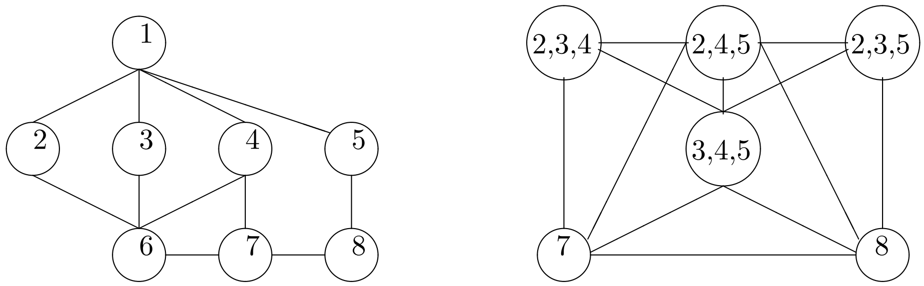
The transformation of G into $S(G, H, p)$ is called the *total struction*. An example of the total struction is given in Figure 5. The importance of this notion is due to the following theorem proved in [6].

Theorem 4. $\alpha(S(G, H, p)) = \alpha(G) - p$.

It is not difficult to see that if H consists of a single vertex and $p = 1$, then total struction coincides with ordinary struction. Therefore, the total struction is a generalization of the ordinary struction.

Moreover, the total struction also generalizes the *crown reduction* proposed to develop fixed-parameter tractable algorithms for the MINIMUM VERTEX COVER problem. A *crown* C in G consists of an independent set I, its neighbourhood S (i.e. the set of vertices adjacent to at least one vertex of I) and a matching between I and S covering all vertices of S. It is known (see e.g. [1]) that if G has a crown as above, then

$$\tau(G) = \tau(G - C) + |S|.$$



Graph G Graph $S(G, H, p)$

Figure 5: Total struction of G with $H = G[1, 2, 3]$ and $p = 2$

Let us show that in this case the crown reduction coincides with total struction centered at I, i.e. $G - C = S(G, I, |I|)$. Indeed, if $C = (I, S)$ is a crown in G, then I is a maximum independent set in $G[I \cup S]$, because if B is the bipartite graph obtained from $G[I \cup S]$ by deleting all edges from S, then $|I| + |S| = |S| + \alpha(B)$ (since for any bipartite graph, the size of a minimum vertex cover coincides with the size of a maximum matching), and hence $|I| = \alpha(B) \geq \alpha(G[I \cup S])$. Therefore, C has no independent sets of size $|I| + 1$ and thus the graph $S(G, I, |I|)$ coincides with $G - C$. By Theorem 4,

$$\alpha(G - C) = \alpha(G) - |I|,$$

which, together with $\tau(G) + \alpha(G) = |V(G)|$, immediately implies $\tau(G) = \tau(G - C) + |S|$.

In addition to crown reduction, the total struction has connections to some other important transformations, which we discuss in the next section. To conclude the present one, we raise the following open question:

Question 3. Does resolution admit a generalization similar to total struction?

4.2.2 Beyond struction and related transformations

The literature contains various other useful transformations of graphs for the MAXIMUM INDEPENDENT SET and related problems. In the present section, we review many of these transformations and reveal some connections between them. We start with transformations for problems closely related to MAXIMUM INDEPENDENT SET.

4.2.2.1 Transformations for related problems

Some transformations that can be helpful for the MAXIMUM INDEPENDENT SET problem have been developed for related problems. We mentioned already in Section 4.2.1.2 crown reduction for the MINIMUM VERTEX COVER problem. Now let us consider other examples.

A transformation which preserves the clique number was proposed by Gerber and Hertz in [18]. This transformation can be described as follows: let x be a vertex in a graph G and

$H_1, \ldots, H_k$ the connected components of the subgraph of G induced by $N(x)$. If we replace x by k new non-adjacent vertices $x_1, \ldots, x_k$, each x_i being linked to all vertices in H_i $(1 \leq i \leq k)$, then the clique number of G does not change. Observe that the graph changes only if $k > 1$.

Even pair contraction is another graph transformation that preserves the clique number. A pair of non-adjacent vertices is called an even pair if every induced path between them has an even number of edges. If a graph G has an even pair, then by contracting it into a single vertex, we do not change the clique number of G. Moreover, under this transformation the chromatic number of G does not change as well (see e.g. [15]).

4.2.2.2 Local transformations for the maximum independent set problem

Let us repeat that struction and total struction are universal in the sense that they do not require any particular structure of the graph. The transformations described below are applicable only to graphs that satisfy some specific conditions.

Neighbourhood reduction. This transformation applies to a pair of adjacent vertices x and y such that every neighbour of x (different from y) also is a neighbour of y. Under this condition, every independent set I containing y contains neither x nor any neighbour of x, and hence y can be replaced in I by x. Therefore, the removal of y from the graph does not change its independence number.

Magnet. Let x and y be two adjacent vertices in a graph. Denote by X the set of private neighbours of x (i.e. the neighbours of x that are non-adjacent to y) and by Y the set of private neighbours of y. If every vertex of X is adjacent to every vertex of Y, then the deletion of y along with the edges connecting x to its private neighbours does not change the independence number of the graph. This transformation is known as *magnet* or *magnetic procedure* [21]. Clearly if X is empty, then the magnetic procedure coincides with the neighbourhood reduction. Therefore, the magnetic procedure generalizes the neighbourhood reduction. Moreover, an attentive reader can also observe that the magnetic procedure generalizes the even pair contraction applied to the complement of the graph.

BAT-reduction is one more transformation that does not change the independence number of the graph. It was proposed in [20] and then generalized in [23]. The importance of this example is that it was originally derived from some pseudo-Boolean arguments. In this sense, it was motivated by the idea of struction, as struction was also derived from pseudo-Boolean arguments.

Clique reduction was proposed by Lovász and Plummer in [28]. It consists in deleting a maximal clique K from the graph and connecting any two non-adjacent vertices x and y in the neighbourhood of K whenever $K \subseteq N(x) \cup N(y)$. Lovász and Plummer showed that if G is a claw-free graph and the independence number in the neighbourhood of K is at most 2, then the clique reduction applied to K decreases the independence number of G by exactly 1. This reduction together with one more transformation described in [28] was used by Lovász and Plummer in order to transform any claw-free graph into a line graph.

4.2.2.3 Beyond struction

In [6], in addition to generalizing struction, the authors introduce one more transformation as follows.

An induced subgraph H of G is α-maximal if $\alpha(G[V(H) \cup \{x\}]) = \alpha(H) + 1$ for every vertex $x \notin H$. The H-reduction consists in deleting H and connecting any two non-adjacent vertices x and y in the neighbourhood of H such that $\alpha(G[V(H) \cup \{x, y\}]) = \alpha(H) + 1$. The H-reduction is said to be α-perfect if it decreases the independence number of G by $\alpha(H)$.

Clearly, if H is a clique, then the H-reduction coincides with the clique reduction of Lovász and Plummer. Moreover, similarly to the clique reduction, the H-reduction becomes α-perfect whenever the independence number in the neighbourhood of H is at most 2. Furthermore, in this case the H-reduction followed by a sequence of magnet transformations coincides with the total struction. Even more importantly, it was shown in [6] that the cycle shrinking of Edmonds is nothing but an α-perfect H-reduction when translated to the language of independent sets. This returns us to the MAXIMUM MATCHING problem and completes our journey.

5 Conclusion: back to matchings

In 1965, Jack Edmonds proposed his celebrated polynomial-time solution to the MAXIMUM MATCHING problem, which is equivalent to the MAXIMUM INDEPENDENT SET problem restricted to the class of line graphs. In the present paper, we looked at possible extensions of this result beyond the line graphs. We also discussed various extensions of algorithmic techniques used by Edmonds in his solution. In particular, we made a short tour through the world of graph transformations. This world includes the cycle shrinking of Edmonds, as well as more advanced tools (struction, total struction) that allow to solve the problem for arbitrary graphs. Clearly, for general graphs these tools do not provide efficient solutions. However, for graphs with particular properties they may become efficient. Determining the area of efficiency of struction, total struction and related transformations is an interesting open question. We do not even know whether this area includes the line graphs.

Question 4. Is it possible to solve the MAXIMUM MATCHING problem in polynomial time by means of graph transformations?

A positive answer to this question could suggest a uniform approach to tackle the MAXIMUM INDEPENDENT SET for graphs with various properties. This approach could be based on a universal tool, total struction, combined with some specific reductions. The advantage of total struction is that it possesses a high level of flexibility and adaptability to various properties of the input graph, both local and global. But the problem of learning and exploiting these abilities is a very challenging task.

Acknowledgements

This work was supported by EPSRC, grant EP/L020408/1.

References

[1] F. N. Abu-Khzam, M.R. Fellows, M.A. Langston, W.H. Suters, Crown structures for vertex cover kernelization, *Theory Comput. Syst.* 41 (2007) 411-430.

[2] V.E. Alekseev, The effect of local constraints on the complexity of determination of the graph independence number, *Combinatorial-algebraic methods in applied mathematics*, Gorkiy University Press, (1982) 3–13 (in Russian).

[3] V.E. Alekseev, On easy and hard hereditary classes of graphs with respect to the independent set problem, *Discrete Appl. Math.* 132 (2003) 17-26.

[4] V.E. Alekseev, Polynomial algorithm for finding the largest independent sets in graphs without forks, *Discrete Appl. Math.* 135 (2004) 3-16.

[5] V.E. Alekseev, V.V. Lozin, Augmenting graphs for independent sets, *Discrete Appl. Math.* 145 (2004) 3-10.

[6] G. Alexe, P.L. Hammer, V.V. Lozin, D. de Werra, Struction revisited, *Discrete Appl. Math.* 132 (2003) 27-46.

[7] E. Balas, C.S. Yu, On graphs with polynomially solvable maximum-weight clique problem, *Networks* 19 (1989) 247-253.

[8] L.W. Beineke, Characterizations of derived graphs, *J. Combinatorial Theory* 9 (1970) 129-135.

[9] C. Berge, Two theorems in graph theory, *Proc. Nat. Acad. Sci. USA* 43 (1957) 842–844.

[10] E. Boros and P.L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123 (2002) 155–225.

[11] C. De Simone, A. Sassano, Stability number of bull- and chair-free graphs. *Discrete Appl. Math.* 41 (1993) 121–129.

[12] M. Davis, H. Putnam, A Computing Procedure for Quantification Theory, *Journal of the ACM* 7 (1960) 201-215.

[13] Ch. Ebenegger, P.L. Hammer, D. de Werra, Pseudo-Boolean functions and stability of graphs, *Annals of Discrete Math.* 19 (1984) 83–97.

[14] J. Edmonds, Paths, trees and flowers, *Canad. J. of Mathematics* 17 (1965) 449-467.

[15] H. Everett, C.M.H. de Figueiredo, C. Linhares-Sales, F. Maffray, O. Porto, B.A. Reed, Path parity and perfection, *Discrete Math.* 165/166 (1997) 233-252.

[16] M. Farber, On diameters and radii of bridged graphs, *Discrete Math.* 73 (1989) 249-260.

[17] F.V. Fomin, F. Grandoni, D. Kratsch, A measure and conquer approach for the analysis of exact algorithms, *J. ACM* 56 (2009) Art. 25, 32 pp.

[18] M.U. Gerber, A. Hertz, A transformation which preserves the clique number, *J. Combinatorial Theory B* 83 (2001) 320–330.

[19] M.U. Gerber, A. Hertz, V.V. Lozin, Augmenting chains in graphs without a skew star, *J. Combin. Theory B* 96 (2006) 352-366.

[20] A. Hertz, On the use of Boolean methods for the computation of the stability number, *Discrete Appl. Math.* 76 (1997) 183–203.

[21] A. Hertz, D. de Werra, A magnetic procedure for the stability number, *Graphs Combin.* 25 (2009) 707–716.

[22] D. Lokshantov, M. Vatshelle, Y. Villanger, Independent Set in P_5-Free Graphs in Polynomial Time, Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2014), 570–581.

[23] V.V. Lozin, Stability preserving transformations of graphs, *Annals of Operations Research*, 188 (2011) 331–341.

[24] V.V. Lozin, M. Milanič, On finding augmenting graphs, *Discrete Appl. Math.* 156 (2008) 2517–2529.

[25] V.V. Lozin, M. Milanič, A polynomial algorithm to find an independent set of maximum weight in a fork-free graph. *J. Discrete Algorithms*, 6 (2008) 595–604.

[26] V.V. Lozin, R. Mosca, Independent sets in extensions of $2K_2$-free graphs, *Discrete Appl. Math.* 146 (2005) 74–80.

[27] V.V. Lozin, R. Mosca, Maximum regular induced subgraphs in $2P_3$-free graphs, *Theoret. Comput. Sci.* 460 (2012) 26–33.

[28] L. Lovász, M.D. Plummer, Matching theory. *Annals of Discrete Mathematics*, 29. North-Holland Publishing Co., Amsterdam; Akadémiai Kiadó (Publishing House of the Hungarian Academy of Sciences), Budapest, 1986. xxvii+544 pp.

[29] G.J. Minty, On maximal independent sets of vertices in claw-free graphs, *J. Combinatorial Theory B*, 28 (1980) 284–304.

[30] R. Mosca, Stable sets in certain P_6-free graphs, *Discrete Appl. Math.* 92 (1999) 177–191.

[31] O.J. Murphy, Computing independent sets in graphs with large girth, *Discrete Appl. Math.* 35 (1992) 167–170.

[32] M. Pelillo, K. Siddiqi, S.W. Zucker, Matching hierarchical structures using association graphs, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21 (1999) 1105–1120.

[33] N. Sbihi, Algorithme de recherche d'un stable de cardinalité maximum dans un graphe sans étoile, *Discrete Math.* 29 (1980) 53–76.

[34] S. Tsukiyama, M. Ide, H. Ariyoshi, I. Shirakawa, A new algorithm for generating all the maximal independent sets, *SIAM J. Comput.* 6 (1977) 505–517.