Counting and detecting small subgraphs via equations and matrix multiplication

Mirosław Kowaluk* Andrzej Lingas † Eva-Marta Lundell ‡

Abstract

We present a general technique for detecting and counting small subgraphs. It consists in forming special linear combinations of the numbers of occurrences of different induced subgraphs of fixed size in a graph. The combinations can be efficiently computed by rectangular matrix multiplication.

Our two main results utilizing the technique are as follows. Let H be a fixed graph with k vertices and an independent set of size s.

1. Detecting if an n-vertex graph contains a (non-necessarily induced) subgraph isomorphic to H can be done in time $O(n^{k-s} + n^{\omega(\lceil (k-s)/2\rceil,1,\lfloor (k-s)/2\rfloor)})$, where $\omega(p,q,r)$ is the exponent of fast arithmetic matrix multiplication of an $n^p \times n^q$ matrix by an $n^q \times n^r$ matrix.
2. When $s = 2$, counting the number of (non-necessarily induced) subgraphs isomorphic to H can be done in the same time, i.e., in time $O(n^{k-2} + n^{\omega(\lceil (k-2)/2\rceil,1,\lfloor (k-2)/2\rfloor)})$. (This improves for $s = 2$ on a counting algorithm of Vassilevska and Williams, running in time $O(n^{k-s+3})$.)

It follows in particular that we can count the number of subgraphs isomorphic to any H on four vertices that is not K_4 in time $O(n^\omega)$, where $\omega = \omega(1,1,1)$ is known to be smaller than 2.376. Similarly, we can count the number of subgraphs isomorphic to any H on five vertices that is not K_5 in time $O(n^{\omega(2,1,1)})$, where $\omega(2,1,1)$ is known to be smaller than 3.334.

1 Introduction

The problems of detecting subgraphs or induced subgraphs of a graph that are isomorphic to another given graph are classical in algorithmics. They are generally termed as *subgraph isomorphism* and *induced subgraph isomorphism* problems, respectively. Their decision, finding, counting and even enumeration versions have been extensively investigated in the literature. In particular, the decision versions include as special cases such well-known NP-hard problems as the independent set, clique, Hamiltonian cycle or path problems [10]. For arbitrary graphs, they are known to admit polynomial-time solutions solely when the other graph, often termed as a pattern graph, is of fixed size.

In this paper we study the complexity of the decision and counting versions of subgraph isomorphism and induced subgraph isomorphism under the assumption that the pattern graph is of a fixed size k, denoting the number of vertices in the input host graph by n.

1.1 Related results on subgraph isomorphism with a fixed pattern graph Already three decades ago, Itai and Rodeh [13] demonstrated that these problems in case the pattern graph is a triangle can be solved in time $O(n^\omega)$. Next, Nešetřil and Poljak [16] presented reductions of the variants of the k-clique problem to those of the triangle problem and its generalization to include not necessarily k-cliques. Subsequently, Kloks, Kratsch and Müller [14], and finally Eisenbrand and Grandoni [8] improved on the reductions to show that generally these problems for k-vertex pattern graphs can be solved in time $O(n^{\omega(\lfloor k/3\rfloor,\lceil (k-1)/3\rceil,\lceil k/3\rceil)})$. This is substantially faster than the time $O(n^k)$ required by an exhaustive enumeration. Recently, Vassilevska and Williams [22] showed that the number of occurrences of a pattern graph with an independent set of size s can be computed in time $2^s n^{k-s+3} k^{O(1)}$.

There are also known examples of pattern graphs where the decision and finding versions can be solved much faster. Namely, already at the beginning of 90s, Plehn and Voight [17] showed that if the fixed pattern graph has treewidth tw then the decision and finding versions of subgraph isomorphism admit an $O(n^{tw+1})$-time solution while those of induced subgraph isomorphism also admit an $O(n^{tw+1})$-time solution in case the maximum degree in the input graph is constant. Yuster and Zwick showed in particular in [25] that cycles of given even length can be found in quadratic time. In [3] Alon, Yuster and Zwick introduced the nowadays classical technique of color coding to detect cycles or paths of

*Institute of Informatics, Warsaw University, Warsaw, Poland. Email: kowaluk@mimuw.edu.pl. Research supported by the grant of the Polish Ministry of Science and Higher Education N20600432/0806.

†Department of Computer Science, Lund University, 22100 Lund, Sweden. Email: Andrzej.Lingas@cs.lth.se. Research supported in part by VR grant 621-2008-4649.

‡Department of Computer Science, Lund University, 22100 Lund, Sweden. Email: Eva-Marta.Lundell@cs.lth.se

constant length roughly in matrix multiplication time, i.e., in time $\widetilde{O}(n^{\omega})$, where the notation $\widetilde{O}$ suppresses polylogarithmic factors. The same authors showed in particular in [4] that for $k = 3,..,7$, the number of k-cycles can be counted in time $O(n^{\omega})$, extending on the classical result of Itai and Rodeh [13] for triangles. In [14], Kloks, Kratsch and Müller showed for the induced variant that if the occurrences of some pattern graph on 4 vertices can be counted in time $T(n)$ then the occurrences of any other pattern graph on 4 vertices can be counted in time $O(n^{\omega} + T(n))$.

More recently, Vassilevska [20] has demonstrated that an induced subgraph isomorphic to $K_k \setminus e$, i.e., K_k with a single edge removed, can be detected in time $O(m^{(k-1)/2}) = O(n^{k-1})$, where m is the number of edges in the input graph, by incorporating among other things earlier results on induced $K_4 \setminus e$ from [8, 14]. She has also presented relatively fast algorithms for the so called semi-cliques in [19]. Williams [23] has showed how to find a path of length k in $\mathcal{O}^*(2^k)$ time, while Björklund, Husfeldt, Kaski, and Koivisto [6] have given an algorithm for counting the number of k-paths running in time $\mathcal{O}^*(\binom{n}{k/2})$, where $\mathcal{O}^*$ suppresses polynomial factors. For a subgraph with treewidth tw, Fomin, Lokshtanov, Raman, Rao, and Saurabh [9] have derived algorithms for the decision and counting versions that run in time $\mathcal{O}^*(2^k n^{2tw})$ and $\binom{n}{k/2} n^{\mathcal{O}(tw \log k)}$, respectively.

See Table 1.1 which summarizes the aforementioned upper-time bounds for detection, finding, and counting small subgraphs.

For several other interesting upper time-bounds in terms of m established for the aforementioned problems, especially when the pattern graph is a triangle, or it has four vertices, or it is a fixed clique, which are superior in the sparse case, see [4, 8, 13, 14].

1.2 Our contributions We present a general technique of deriving independent linear dependencies among the numbers of occurrences of different induced subgraphs of fixed size in a host graph. The coefficients at the numbers in the dependencies are easily computable while the computation of the right-hand sides of the dependencies reduces to the so called l-neighborhood problem. We show that the latter problem can be relatively efficiently solved via rectangular matrix multiplication [7, 12].

In [14], Kloks, Kratsch and Müller described some of the dependencies in the special case of some subgraphs of size 4. Therefore, our technique can be seen as a far-reaching generalization and systematization of their idea.(On the other hand, the dependencies and matrix computations used by Alon, Yuster and Zwick in [4] to derive their results on counting k-cyclic graphs for $k = 3, ..., 7$ rely on a different idea of computing traces of matrix powers.)

Let H_k denote the family of single representatives of all isomorphism classes for undirected graphs on k vertices, and let $H_k(l)$ stand for its subfamily comprised of all graphs in H_k having an independent set of size at least $k - l$.

Assume $k = O(1)$. If for all graphs in $H_k \setminus H_k(l)$ their numbers of occurrences either as induced or non-necessarily induced subgraph of the input graph are known then we can compute the number of occurrences of any $H \in H_k$ both as induced and non-necessarily induced subgraph in time $O(n^l + n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$. The latter term in the upper bound stands for the time required to solve the aforementioned l-neighborhood problem.

In the case $l = k - 2$, the knowledge of the number of occurrences of any given graph in H_k as an induced subgraph is sufficient to compute the number of occurrences of any $H \in H_k$ both as induced and non-necessarily induced subgraph in time $O(n^{k-2} + n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$. (This generalizes the corresponding fact shown for $k = 4$ in [14]).

Our main results utilizing this technique are two new upper time-bounds on detecting and counting occurrences of $H \in H_k(l)$ as (non-necessarily induced) subgraphs in the host graph on n vertices. We show that

1. detecting if an n-vertex graph contains a (non-necessarily induced) subgraph isomorphic to H can be done in time $O(n^l + n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$, and that
2. when $l = k - 2$, counting the number of (non-necessarily induced) subgraphs isomorphic to H can be done in the same time, i.e., in time $O(n^{k-2} + n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$. (This improves, for $k - l = 2$, on the aforementioned counting algorithm of Vassilevska and Williams [22], the running time of which can be rephrased as $O(n^{l+3})$ in terms of our notation.)

It follows in particular that the counting version can be solved for any $H \in H_4 \setminus \{K_4\}$ in time $O(n^{\omega})$ and for any $H \in H_5 \setminus \{K_5\}$ in time $O(n^{\omega(2,1,1)})$, where $\omega < 2.376$ and $\omega(2, 1, 1) < 3.334$.

1.3 Organization In the next section we briefly introduce a notation corresponding to our counting versions of induced subgraph isomorphism and subgraph isomorphism and a related known fact. In Section 3, we present our aforementioned general technique. In the next section, we derive our general results on counting and detecting copies of graphs from $H_k(l)$, including our first main result on detection. Section 5 is devoted to our second main result on fast counting of small subgraphs with an independent set of size at least two. In Section 6, we present our solution to the aforementioned problem of l-neighborhood which allows us to compute the right-hand sides of our equations efficiently. In consequence, we can specify exactly upper bounds in our main theorems and derive concrete corollaries on counting copies of graphs from the sets $H_4(2)$ and $H_5(3)$, respectively. We conclude with final remarks.

subgraph	time complexity	problem	reference
K_3	n^{ω}	detection/finding	Itai-Rodeh [13]
K_3	n^{ω}	counting	Itai-Rodeh [13]
K_4	$\mathcal{O}(m^{(\omega+1)/2})$	counting	Kloks et al. [14]
H_4	$\mathcal{O}(n^{\omega} + m^{(\omega+1)/2})$	counting	Kloks et al. [14]
H_k	$O(n^{\omega(\lfloor k/3\rfloor,\lceil (k-1)/3\rceil,\lceil k/3\rceil)})$	detection	Eisenbrand-Grandoni [8]
$H_{k,s}$	$2^s n^{k-s+3} k^{O(1)}$	counting	Vassilevska-Williams [22]
$H_{k,tw}$	$\mathcal{O}(n^{tw+1})$	detection/finding	Plehn-Voight [17]
$C_k, k \leq 7$	n^{ω}	counting	Alon et al. [4]
C_k	$n^{\omega} \log n$	finding	Alon et al. [3]
$K_k \setminus e$	$\mathcal{O}(n^{k-1})$	detection	Vassilevska [20]
P_k	$\mathcal{O}^*(2^k)$	detection	Williams [23]
P_k	$\mathcal{O}^*(\binom{n}{k/2})$	counting	Björklund et al. [6]
$H_{k,tw}$	$\mathcal{O}^*(\binom{n}{k/2} n^{2p})$	counting	Fomin et al. [9]

Table 1: Upper time-bounds on detecting, finding and counting small subgraphs in an undirected, unweighted graph G on n vertices and m edges. H_k stands for the class of pattern graphs on k vertices, additional subscripts s, tw, and p denote the size of an independent set, the treewidth, and the path-width, respectively.

2 Preliminaries

Recall that for a positive integer k, H_k denotes a family of single representatives of all isomorphism classes for graphs on k vertices while for $l \in \{1, 2, ..., k-1\}$, $H_k(l)$ denotes the family of all graphs in H_k that contain an independent set on $k - l$ vertices.

DEFINITION 1. *For a graph $H \in H_k$ and a host graph G on at least k vertices, the number of sets of k vertices in G that induce a subgraph of G isomorphic to H is denoted by $NI(H, G)$. Similarly, the number of subgraphs of G that are isomorphic to H (where all automorphic transformations of a subgraph are counted as one) is denoted by $N(H, G)$. Finally, for a vertex v of G and a subgraph F of G, the neighborhood of v in F is the set of all neighbors of v in F.*

It is well known that computing $NI(H, G)$ for $H \in H_k$ is interchangeable with computing $N(H, G)$ for $H \in H_k$ (e.g., see Theorem 2.3 in [15]). We rephrase this known result in terms of our notation as follows.

Fact 1. *For $H \in H_k$, the equalities $N(H, G) = \sum_{H'' \in H_k} N(H, H'')NI(H'', G)$ hold. The $|H_k| \times |H_k|$ matrix $M = [N(H, H'')]_{H,H'' \in H_k}$ is non-singular and M^{-1} has integer entries.*

3 Forming equations in terms of $NI(H'', G)$

Let H be a graph on k vertices and let H' be an induced subgraph of H on l vertices such that the $k - l$ vertices of H outside H' form an independent set. Consider a family of supergraphs H'' of H (including H) such that H'' has the same vertex set as H and the set of edges between H' and $H'' \setminus H'$ is the same as that between H' and $H \setminus H'$. We denote this family by $H_k(H, H')$. For an illustration see Fig 1(a).

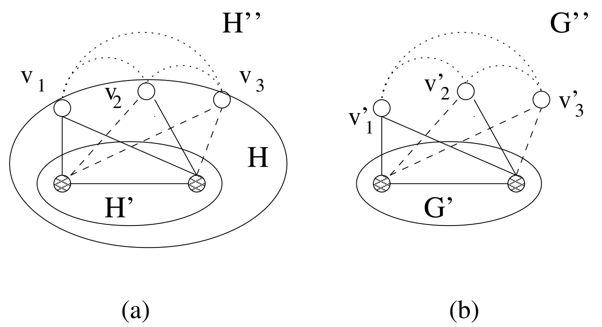

Figure 1: (a) An example of a graph H composed of the induced subgraph H' and the vertex set $\{v_1, v_2, v_3\}$ which forms an independent set in H, and a supergraph H'' of H. Edges of H' and/or H are denoted by continuous segments, absent edges between H' and $H \setminus H'$ are denoted by broken segments while edges of H'' outside H are denoted by dotted segments. (b) An example of an induced subgraph G''.

The main idea of our method relies on the fact that a linear combination of the numbers of induced copies of $H'' \in H_k(H, H')$ in the host graph G , i.e., $NI(H'', G)$ for $H'' \in H_k(H, H')$, can be computed relatively efficiently. The coefficient at $NI(H'', G)$ in the linear combination is equal to the number of automorphisms of H'' divided by the number of automorphisms of H'' that are identity on H'. We denote it by $A(H', H'')$. To form an equation, we shall place the linear combination $\sum_{H'' \in H_k(H,H')} A(H', H'')NI(H'', G)$ on the left side and its computed value on the right side of the equality, treating $NI(H'', G)$ as unknowns.

To show that our linear combination can be computed efficiently, we proceed as follows.

Consider an l-tuple α of vertices of G such that the mapping assigning the j-th vertex in the tuple to the j-th vertex in H' is an isomorphism between the subgraph G' of G induced by the tuple and H'. We shall call such an l-tuple α *relevant*.

For all relevant l-tuples, we shall count the number of equivalence classes of $(k-l)$-tuples of vertices $v'_1, ..., v'_{k-l}$ in $G \setminus G'$ where the neighborhood of v'_i in G' corresponds to that of the i-th vertex of $H \setminus H'$ in H' under the isomorphism between G' and H'. Two such $(k-l)$-tuples belong to the same equivalence class with respect to α iff one of them can be obtained from the other by permutations of vertices v'_i having the same neighborhood in G'.

Proposition 1. *The total number of the equivalence classes of $(k-l)$-tuples summed over all relevant l-tuples α is equal to $\sum_{H'' \in H_k(H,H')} A(H', H'')NI(H'', G)$.*

Proof. Consider an induced subgraph G'' of G isomorphic to $H'' \in H_k(H, H')$, see Fig. 1(b). Let G' be the induced subgraph of G'' corresponding to H' under this isomorphism. Renumber the vertices of G'' so first come the vertices of G' and then those in $G'' \setminus G'$. Next, consider the l-tuple α of vertices of G as well as the $(k-l)$-tuple β of vertices of G which concatenated yield G'' (i.e., the j-th vertex in the combined k-tuple is the j-th vertex in G''). Consider any other $(k-l)$-tuple γ which combined with α yields an induced subgraph automorphic to G''. It follows that γ can be obtained from β by applying a collection of permutations π_t to the groups of vertices in the first $(k-l)$-tuple that have the same neighborhood in G', respectively. Hence, all the $(k-l)$-tuples γ complementing α to a k-tuple yielding a subgraph automorphic to G'' fall in the same equivalence class with respect to α and are counted as one. Their number is equal to the number of automorphisms of H'' which are the identity on H'.

Note also that no other $(k-l)$-tuple γ that together with the l-tuple α yields an induced subgraph isomorphic to another graph in $H_k(H, H')$ can fall in the same equivalence class with respect to α as β. Simply, the permutations of vertices in our $(k-l)$-tuple β with the same neighborhood in G' that yield γ have to define an automorphism on the subgraph induced by the vertices of β. Since this automorphism does not change the neighborhoods in G', it can be can easily extended to an automorphism of the whole subgraph G'' by using identity mapping on the vertices of G'.

Furthermore, any $(k-l)$-tuple satisfying the requirements of neighborhood in G' when combined with α has to yield an induced subgraph isomorphic to some graph in $H_k(H, H')$.

We conclude that for each l-tuple α of vertices in G such a single equivalence class is in one-to-one correspondence with the set of all isomorphisms between a graph $H'' \in H(H, H')$ and an induced subgraph G'' of G that map the i-th vertex of H' on the i-th vertex of α. Note that for the aforementioned H'' and G'', there are exactly $A(H', H'')$ (i.e., the number of automorphisms of H'' divided by the number of automorphisms of H'' that are identity on H') different l-tuples α that define such a set of isomorphisms between H'' and G''. The proposition follows. □

We shall show that computing the total number of the equivalence classes easily reduces to the following l-neighborhood problem.

DEFINITION 2. *The l-neighborhood problem is to determine for each l-tuple of vertices of G and each binary vector b with l coordinates the number of vertices v in G outside the l-tuple such that v is a neighbor of the i-th vertex in the l-tuple iff $b(i) = 1$.*

We shall denote the time required to solve the l-neighborhood problem by $T_l(n)$.

Proposition 2. *The total number of the equivalence classes of $(k-l)$-tuples summed over all relevant l-tuples α can be computed in time $O(n^l(k-l) + T_l(n))$.*

Proof. There are at most $k-l$ different neighborhoods of $v'_i \in G \setminus G'$ in the subgraph G' induced by a relevant l-tuple α, corresponding to those of $v_i \in H \setminus H'$ for $i = 1, ..., k-l$ in the subgraph H' under the isomorphism between G' and H'. Each of these neighborhoods can be identified with a binary vector of length l termed as the type of the neighborhood.

To compute the number of equivalence classes with respect to α it is sufficient to compute for each type t of neighborhood of $v'_i \in G \setminus G'$ in G' corresponding to those of $v_i \in H \setminus H'$ in H', the number n_t of vertices in $G \setminus G'$ having the neighborhood of type t in G'. Note that the number of occurrences of a given neighborhood type t in any of the $(k-l)$-tuples corresponding to $H \setminus H'$ is fixed, say o_t. Therefore, the aforementioned number of equivalence classes for the $(k-l)$-tuples complementing the l-tuple α is simply $\Pi_t \binom{n_t}{o_t}$.

For an arbitrary l-tuple α, let $N(\alpha)$ stand for the set of all (at most $k-l$) neighbor types of vertices in $G \setminus G'$ in G'. Then, the number of all equivalence classes over all relevant l-tuples α is given by the sum $\sum_\alpha \Pi_{t \in N(\alpha)} \binom{n_t}{o_t}$. If the numbers n_t are given then this sum can be easily computed in time $O(n^l(k-l))$. It is sufficient to observe that these numbers can be determined by solving the l-neighborhood problem. □

The easily computable values of $A(H', H'')$ (recall $k = O(1)$) can be treated as coefficients at the unknowns which correspond to $NI(H'', G)$ for $H'' \in H_k(H, H')$ respectively, in order to form the left-hand side of an equation whose right-hand side is the computed value of our linear combination.

We let $Eq(H, l)$, where $l \in \{1, ..., k-1\}$, denote the set of such equations, each one with $|H_k(H, H')|$ unknowns corresponding to $NI(H'', G)$ for $H'' \in H_k(H, H')$, respectively.

Summarizing, for $H \in H_k(l)$, the set $Eq(H, l)$ consists of equations in one-to-one correspondence with induced subgraphs H' of H on l vertices whose left sides have the form $\sum_{H'' \in H_k(H,H')} A(H', H'')x_{H'',G}$, where the variables $x_{H'',G}$ correspond to $NI(H'', G)$, respectively.

By Propositions 1,2, we obtain the following lemma.

LEMMA 3.1. *For $H \in H_k(l)$, the right-hand side of an equation in $Eq(H, l)$ can be evaluated in time $O(n^l(k-l) + T_l(n))$*

See Example 1 and Example 2 for examples of systems of equations in $Eq(H, l)$ where $H \in H_k(l)$. The equations in Example 2 can be regarded as an extension of those for connected $H \in H_4$ given in [14].

Example 1: The following is an example of equations in $Eq(H, 1)$ where $H \in H_3(1)$ (corresponding to those in [11]).
Let $G = (V, E)$ be a graph on n vertices, and for $v \in V$, let $deg(v)$ stand for the degree of v in G. Next, for $i = 1, ..., 3$, let t_i denote a graph on three vertices that contains exactly i edges. Thus in particular t_0 consists of three isolated vertices while t_3 is a triangle, i.e., K_3. For $i = 0, 1, 2$, we obtain the three following equations in $Eq(t_i, 1)$ respectively:

a) $A(K_1, t_0)NI(t_0, G) + A(K_1, t_1)NI(t_1, G) = \#\{< v, \{u, w\} > \mid \{v, u, w\} \subset V \;\&\; \{(v, u), (v, w)\} \cap E = \emptyset\}$,

b) $A(K_1, t_1)NI(t_1, G) + A(K_1, t_2)NI(t_2, G) = \#\{< v, < u, w >> \mid \{v, u, w\} \subset V \;\&\; (v, u) \in E \;\&\; (v, w) \notin E\}$,

c) $A(K_1, t_2)NI(t_2, G) + A(K_1, t_3)NI(t_3, G) = \#\{< v, \{u, w\} > \mid \{(v, u), (v, w)\} \subset E\}$.

By computing the coefficients $A(K_1, t_i)$, setting $T_i = NI(K_1, t_i)$ for $i = 0, 1, ..., 3$, and evaluating the right hand sides, we obtain the following system of linearly independent equations:

(i) $3T_0 + T_1 = \sum_{v \in V} \binom{n-deg(v)-1}{2}$,

(ii) $2T_1 + 2T_2 = \sum_{v \in V} deg(v)(n - deg(v) - 1)$, and

(iii) $T_2 + 3T_3 = \sum_{v \in V} \binom{deg(v)}{2}$.

Example 2: Assume the notation from Example 1. Next, let

- Q_0 denote the number of quadruples of vertices in G which form independent sets, i.e., equivalently, the number of K_4 in the the complement graph;
- $Q_|$ denote the number of quadruples of vertices in G which induce exactly only one edge;
- $Q_{||}$ denote the number of quadruples of vertices in G which induce exactly two non-incident edges;
- $Q_\bigwedge$ denote the number of quadruples of vertices in G which induce exactly a path on two edges and an isolated vertex;
- $Q_\bigsqcup$ denote the number of quadruples in G that induce a path on three edges;
- $Q_\Yup$ denote the number of quadruples in G that induce exactly a star composed of three incident edges (claw);
- $Q_{\triangleright.}$ denote the number of quadruples in G that induce exactly a triangle and an isolated vertex;
- $Q_{\triangleright-}$ denote the number of quadruples in G that induce exactly a triangle and an edge incident to it (paw);
- $Q_\square$ denote the number of quadruples of vertices in G that induce exactly C_4;
- $Q_\boxslash$ denote the number of quadruples of vertices in G that induce exactly five edges of G, (diamond);
- Q_6 denote the number of quadruples of vertices in G that induce six edges of G, i.e., K_4.

We obtain the following system of ten linearly independent left-hand sides of simplified equations respectively in $Eq(Q_s, 2)$, where $s \neq 6$, whose right-hand sides can be computed in time $O(n^\omega)$. In part, they coincide with the equations for connected Q_s presented in [14]. It is indicated in the parentheses whether K_2 or an independent set on two vertices, denoted by I_2, is respectively used as H'.

1) $6Q_0 + Q_| \; (I_2)$

2) $Q_| + 2Q_{||} \; (K_2)$

3-4) $Q_\bigwedge + 3Q_\Yup \; (I_2)$, $4Q_{||} + Q_\bigsqcup \; (I_2)$

5-7) $3Q_{\triangleright.} + Q_{\triangleright-} \; (K_2)$, $2Q_\bigsqcup + 2Q_{\triangleright-} \; (I_2)$,
$3Q_\Yup + Q_{\triangleright-} \; (K_2)$

8) $2Q_\square + Q_\boxslash \; (I_2)$

9) $2Q_{\triangleright-} + 4Q_\boxslash \; (K_2)$

10) $Q_\boxslash + 6Q_6 \; (K_2)$

Note that in particular the obvious equation $Q_0 + Q_{|} + Q_{\bigwedge} + Q_{||} + Q_{\triangleright.} + Q_{\bigsqcup} + Q_{\curlywedge} + Q_{\triangleright-} + Q_{\square} + Q_{\boxslash} + Q_6 = \binom{n}{4}$ can be easily derived from these equations.

LEMMA 3.2. *For each H in $H_k(l)$, pick an arbitrary equation from $Eq(H,l)$. The resulting system of $|H_k(l)|$ equations is linearly independent.*

Proof. Order the graphs in H_k so the number of edges is non-decreasing. Let A be the $|H_k(l)| \times |H_k|$ matrix corresponding to the left-hand side of the equations in $Eq(H,l)$ for $H \in H_k(l)$. Note that for each $H,\ H'' \in H_k$, where H'' has the same number of edges as H and $H \neq H''$, if H' is an induced subgraph of H such that $H \backslash H'$ is an independent set on $k-l$ vertices then H'' cannot be a member of $H_k(H,H')$ and consequently the coefficient at $NI(H'',G)$ in the equation from $Eq(H,l)$ is 0. It follows that the leftmost maximal square submatrix of M of size $|H_k(l)| \times |H_k(l)|$ has non-zero elements along the diagonal starting from the top-left corner and only zeros below the diagonal. □

4 Counting and detection of induced subgraphs of equal size

In this section, we shall use the equations derived in the previous section to count and detect different induced subgraphs of equal fixed size.

THEOREM 4.1. *If for all $H \in H_k \setminus H_k(l)$ the values $NI(H,G)$ are known then for all $H'' \in H_k$, the numbers $NI(H'',G)$ and $N(H'',G)$ can be determined in time $O(|H_k(l)|(n^l(k-l) + |H_k|k^2k! + |H_k(l)|^2) + T_l(n))$, in particular in time $O(n^l + T_l(n))$ for $k = O(1)$.*

Proof. We can enumerate all automorphisms of a graph on k vertices in time $O(k^2k!)$. Hence, computing all the possible coefficients $A(H,H')$ on the left-sides of the equations from Lemma 3.2 takes time $O(|H_k(l)||H_k|k^2k!)$. It follows by Lemma 3.1 that forming the aforementioned equations takes time $O(|H_k(l)||H_k|k^2k! + |H_k(l)|n^l(k-l) + T_l(n))$. If for all $H \in H_k \setminus H_k(l)$, the values $NI(H,G)$ are known then we can substitute these values for the corresponding variables in the aforementioned equations. By arguing analogously as in the proof of Lemma 3.2, we infer that the resulting $|H_k(l)|$ equations with $|H_k(l)|$ unknowns are also linearly independent. Hence, we can solve the resulting equations completely in time $O(|H_k(l)|^3)$. It remains to apply Fact 1, to obtain all the values $N(H'',G)$ as well. □

Clearly, if we are interested in the number of bijections $b : V(H) \to V(G)$ such that $(b(u), b(v)) \in E(G)$ iff $(u,v) \in E(H)$ then one should multiply $NI(H'',G)$ with the number of automorphisms of H''. The latter can be computed by checking all permutations of vertices in time $O(k!k^2)$.

Marginally, Theorem 4.1 can be extended to the following form, symmetric with respect to $NI(H,G)$ and $N(H,G)$, by Fact 1.

THEOREM 4.2. *If for all $H \in H_k \setminus H_k(l)$ either the values $N(H,G)$ or the values $NI(H,G)$ are known then for all $H'' \in H_k$, the numbers $N(H'',G)$ and $NI(H'',G)$ can be determined in time $O(n^l + T_l(n))$ for $k = O(1)$.*

Proof. By Theorem 4.1, we may assume w.l.o.g that $N(H,G)$ are known for all $H \in H_k \setminus H_k(l)$. Form the initial $|H_k(l)|$ linearly independent equations with $|H_k|$ unknowns corresponding to $NI(H'',G)$ where $H'' \in H_k$, as in the proof of Theorem 4.1. Let A be the $|H_k(l)| \times |H_k|$ matrix of coefficients of the left-hand sides of the aforementioned equations. By Fact 1, these equations can be transformed into another set of $|H_k(l)|$ equations with $|H_k|$ unknowns corresponding to $N(H'',G)$, where $H'' \in H_k$. The matrix of coefficients of the left-hand sides of the new set of equations is the matrix product of A with the inverse of the matrix M given in Fact 1. Since A has rank $|H_k(l)|$ and M is non-singular, the product matrix has also rank $|H_k(l)|$. Thus, the new set of $|H_k(l)|$ equations is also linearly independent. Note also that each of the new equations corresponding to an original equation in $Eq(H',H)$ will have a non-zero coefficient solely at $N(H,G)$ and $N(H'',G)$, where H'' is a supergraph of H in H_k, by the analogous property of the original equations and Fact 1.

Now, if we substitute the known values $N(H,G)$ for the corresponding variables in these new equations, we obtain $|H_k(l)|$ equations with $|H_k(l)|$ unknowns. The resulting equations are also linearly independent by the arguments analogous to that in the proof of Lemma 3.2. Hence, we can solve them completely to obtain all values $N(H'',G)$ for $H'' \in H_k$. By symmetrically applying Fact 1, we obtain then also all values $NI(H'',G)$ for $H'' \in H_k$. □

For the problem of deciding whether or not the input graph G has a subgraph isomorphic to a given $H \in H_k \setminus H_k(l)$, we obtain the following stronger result (our first main result).

THEOREM 4.3. *For $k = O(1)$ and any $H \in H_k(l)$, one can decide whether or not $N(H,G) = 0$ in time $O(n^l + T_l(n))$.*

Proof. Let $H \in H_k(l)$. $N(H,G) > 0$ iff there is a supergraph H_1 of H in H_k such that $NI(H_1,G) > 0$. Therefore, for each supergraph H_1 of H (including H), we proceed as follows.

If $H_1 \in H_k(l)$, we consider the equation in $Eq(H,l)$ in the set of equations from the proof of Lemma 3.2 and Theorem 4.1. Its left side is a linear combination of variables $x_{H''}$ in one-to-one correspondence to $NI(H'',G)$ where $H'' = H_1$ or H'' is some supergraph of H_1 in H_k, and all coefficients are positive. Hence, by computing the right-hand side of the equation in time $O(n^l + T_l(n))$ according to Lemma 3.1, we can decide whether or not there is a supergraph H'' of H in a set of supergraphs of H_1 including H_1 such that $NI(H'', G) > 0$. If the right-hand side is positive we know that $N(H, G) > 0$.

If $H_1 \notin H_k(l)$, we consider the supergraph H_2 of H which results from H_1 by deleting all edges between the $k-l$ independent vertices of H. Clearly, H_2 is also a subgraph of H_1 and it belongs to $H_k(l)$. Importantly, in the equation in $Eq(H_2, l)$ there must be a variable x_{H_1} corresponding to $NI(H_1, G)$. Hence, similarly as in the previous case, by computing the right-hand side of the equation in time $O(n^l + T_l(n))$, we can decide whether or not there is a supergraph H'' of H in a set of supergraphs of H including H_1 such that $NI(H'', G) > 0$.

If we obtain negative answers for all supergraphs H_1 of H then we know that $N(H, G) = 0$.

Since for $k = O(1)$ the total number of supergraphs $H_1 \in H_k$ is $O(1)$, the total time complexity remains $O(n^l + T_l(n))$. □

Note that we can also estimate $N(H, G)$ for $H \in H_k(l)$ within a constant multiplicative factor in time $O(n^l + T_l(n))$. It is sufficient to compute the sum of the right-hand sides of the equations used in the proof of Theorem 4.3. Since for $k = O(1)$ the total number of the equations is $O(1)$ and the coefficients at $NI(H_1, G)$, where H_1 is a supergraph of H in H_k, are also $O(1)$, each copy of such supergraph H_1 will be counted only $O(1)$ times in the sum.

5 Fast counting of small subgraphs with independent set of size 2

For $l = k - 2$, we can derive our most interesting results on computing $N(H, G)$. We begin with the following useful transformation of our equations.

LEMMA 5.1. *The set of equations in $Eq(H, k-2)$ for $H \in H_k(k-2)$ from the proof of Theorem 4.1 and Lemma 3.2 can be transformed to an equivalent set of equations whose left sides are of the form $x_H + (-1)^{\binom{k}{2} - m_H + 1} N(H, K_k) x_{K_k}$, where x_H and x_{K_k} are respectively in one-to-one correspondence with $NI(H, G)$ and $NI(K_k, G)$, m_H stands for the number of edges of H, and whose right sides are computable in time $O(n^{k-2} + T_{k-2}(n))$.*

Proof. Consider the set S of linearly independent equations from $Eq(H, k-2)$, $H \in H_k(k-2)$, from the proof of Theorem 4.1 and Lemma 3.2. By the structure of these equations, they can be easily transformed into the set of equations with the left side of the form $x_H + c_H x_{K_k}$, where x_H is the variable corresponding to $NI(H, G)$, x_{K_k} is the variable corresponding to $NI(K_k, G)$, c_H is a constant, and the right side is computable in time $O(n^{k-2} + T_{k-2}(n))$.

To show that $c_H = (-1)^{\binom{k}{2} - m_H + 1} N(H, K_k)$, we need to introduce the following notation.

For $F \in H_k$, let $aut(F)$ be the number of automorphisms of F and let $autid(H', F)$ be the number of automorphisms of F that are identity on H'.

Note that for $F \in H_k$, $N(F, K_k) = k!/aut(F) = aut(K_k)/aut(F)$ holds.

We shall prove by induction on the number of edges missing to K_k, i.e., $\binom{k}{2} - m_F$, that for $F \in H_k(k-2)$, the equality $c_F = (-1)^{\binom{k}{2} - m_F + 1} aut(K_k)/aut(F)$ holds.

Consider an original equation whose left side is of the form $A(H', H)x_H + A(H', H'')x_{H''}$, where H' is a subgraph of H including all vertices and edges of H but two vertices not connected by an edge and edges incident to them, and H'' denotes H augmented by the edge connecting these two vertices.

By the definition, we have $A(H', F) = aut(F)/autid(H', F)$ for $F \in \{H, H''\}$. Note also that if there is an automorphism of $F \in \{H, H''\}$ in $autid(H', F)$ that is not identity on F then the two vertices of F outside H' have to have the same neighborhood in H'. It follows that $autid(H', H) = autid(H', H'')$.

Suppose $H = K_k \setminus e$. By the equalities $A(H', H) = aut(K_k \setminus e)/autid(H', K_k \setminus e)$, $A(H', K_k) = aut(K_k)/\ autid(H', K_k)$, and $autid(H', K_k \setminus e) = autid(H', K_k)$, it is sufficient to multiply the equation by $autid(H', K_k)/aut(K_k \setminus e)$ to transform its left side to the form $x_{K_k \setminus e} + \frac{aut(K_k)}{aut(H)} x_{K_k}$. Thus, the induction hypothesis holds for $F = K_k \setminus e$.

We may assume further that H is a strict subgraph of $K_k \setminus e$, and that the induction hypothesis holds for $F = H''$.

We have $c_H = -\frac{c_{H''} A(H', H'')}{A(H', H)}$. By $A(H', F) = aut(F)/autid(H', F)$ and the inductive hypothesis, the latter equality yields c_H equal to

$$-\frac{(-1)^{\binom{k}{2} - m_{H''} + 1} aut(K_k)}{aut(H'')} \frac{aut(H'')}{autid(H', H'')} \frac{autid(H', H)}{aut(H)}$$

By $autid(H', H'') = autid(H', H)$ and straightforward simplifications, we obtain the induction hypothesis for $F = H$. □

The following theorem is an immediate consequence of Lemma 5.1 and Theorem 4.2.

THEOREM 5.1. *For any $H \in H_k$, if the value of $NI(H, G)$ is known then for all $H'' \in H_k$, the numbers $NI(H'', G)$ and $N(H'', G)$ can be determined in time $O(n^{k-2} + T_{k-2}(n))$ for $k = O(1)$.*

Proof. If the value of $NI(H, G)$ is known then by Lemma 5.1 that of $NI(K_k, G)$ can be computed in time $O(n^{k-2} + T_{k-2}(n))$. Now the thesis follows from Theorem 4.2. □

Fact 1 combined with Lemma 5.1 yields our main result in this section.

THEOREM 5.2. *For any $H \in H_k(k-2)$, i.e., any graph H on k vertices different from K_k, $N(H,G)$ can be computed in time $O(n^{k-2} + T_{k-2}(n))$.*

Proof. For $F \in H_k$, we shall denote the set of edges of F by E_F and its cardinality by m_F.

Let $H \in H_k$ and $k' = \binom{k}{2}$. By Fact 1 and Lemma 5.1, we have

$$N(H,G) = \sum_{H'' \in H_k \& E_H \subseteq E_{H''}} C + (-1)^{k'-m_{H''}} N(H,H'')N(H'',K_k)NI(K_k,G)$$

where C can be computed in time $O(n^{k-2} + T_{k-2}(n))$. On the other hand, for any $m_H \le i \le k'$, we have

$$\sum_{H'' \in H_k, E_H \subseteq E_{H''}, m_{H''}=j} N(H,H'')N(H'',K_k)NI(K_k,G) = N(H,K_k)NI(K_k,G)\binom{k'-m_H}{j-m_H}$$

It follows that

$$N(H,G) = C + N(H,K_k)NI(K_k,G)\left(\sum_{j=m_H}^{k'} (-1)^{k'-j}\binom{k'-m_H}{j-m_H}\right)$$

On the other hand, we have

$$\sum_{j=m_H}^{k'} (-1)^{k'-j}\binom{k'-m_H}{j-m_H} = \sum_{m=0}^{k'-m_H} (-1)^{(k'-m_H)-m}\binom{k'-m_H}{m} = 0$$

We conclude that $N(H,G) = C$, i.e., $N(H,G)$ can be computed in time $O(n^{k-2} + T_{k-2}(n))$. □

6 Solving the l-neighborhood problem and finalizing the main results

We can solve the l-neighborhood problem for a graph G as follows.

If the length l of the binary vectors b is 1 then for each vertex v of G it is sufficient to report the number of neighbors if $b(1) = 1$ or non-neighbors if $b(1) = 0$.

Suppose that $l > 1$. For each binary vector b of length l, we proceed as follows. We form two arithmetic matrices A and B. The rows of the matrix A correspond to $\lceil l/2 \rceil$-tuples of vertices of G. The columns of A correspond to vertices of G. Each entry $A[t_1, k]$ is set to 1 iff the k-th vertex has the neighborhood in the subgraph induced by the $\lceil l/2 \rceil$-tuple t_1 of vertices described by the first $\lceil l/2 \rceil$ bits of the vector b, otherwise $A[t,k]$ is set to 0. We define the matrix B analogously by substituting $\lfloor l/2 \rfloor$-tuples for $\lceil l/2 \rceil$-tuples and exchanging rows with columns. Thus, in particular if l is even then the transpose of B is equal to A.

Note that the matrices A and B can be constructed in time $O(n^{\lceil l/2 \rceil + 1} l)$.

Consider now the arithmetic product C of A and B. Let t be any tuple of l vertices in G. Decompose t into the prefix t_1 of length $\lceil l/2 \rceil$ and the suffix t_2 of length $\lfloor l/2 \rfloor$. Observe that $C[t_1, t_2]$ is equal to the number of vertices in G that have neighborhood specified by the binary vector b.

It follows that it is sufficient to compute the product C. Note that there are 2^l different vectors b. Recall that $\omega(p,q,r)$ denotes the exponent of fast matrix multiplication for rectangular matrices of size $n^p \times n^q$ and $n^q \times n^r$, respectively. We obtain the following theorem.

THEOREM 6.1. *The l-neighborhood problem for a graph on n vertices can be solved in time $O(n)$ for $l = 1$ and in time $O(2^l n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$ for $l \ge 2$.*

By combining Theorem 6.1 with Theorems 4.3, 5.2, we obtain the following more explicit formulation of our main results.

THEOREM 6.2. *For $k = O(1)$ and any $H \in H_k(l)$, one can decide whether or not $N(H,G) = 0$ in time $O(n^l + n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$.*

By [7, 12], when $1 \le 0.294l/2 \le 0.147l$ and so if $l \ge 7$, then the second term in the upper time-bound of Theorem 6.2 is not greater than the first one and consequently the upper bound reduces to $O(n^l)$.

THEOREM 6.3. *Let $k = O(1)$. For all $H \in H_k(k-2)$, i.e., all $H \in H_k \setminus \{K_k\}$, the numbers $N(H,G)$ can be computed in time $O(n^{k-2} + n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$.*

COROLLARY 6.1. *For all $H \in H_4 \setminus \{K_4\}$, the numbers $N(H,G)$ can be computed in time $O(n^\omega)$.*

COROLLARY 6.2. *For all $H \in H_5 \setminus \{K_5\}$, the numbers $N(H,G)$ can be computed in time $O(n^{\omega(2,1,1)})$.*

Huang and Pan showed that $\omega(2,1,1) < 3.334$ in [12].

In the particular case of a few graphs termed 4-cyclic by Alon, Yuster and Zwick in [4], Corollary 6.1 coincides with their result stating that for $k = 3,..,7$ and any k-cyclic graph H, $N(H,G)$ can be computed in time $O(n^\omega)$ [4]. The k-cyclic graphs form a narrow family of sparse graphs in H_k that are homomorphic images of C_k.

Finally, an unknown referee observed that by generalizing the method of Nešetřil and Poljak [16], one can also count the number of occurrences of H in G under the assumptions of Theorem 6.3 in time $O(n^{r+z\omega})$, where $k = 3z + r$ and $r \in \{0,1,2\}$. This observation yields better upper time-bounds than those of Theorem 6.3 for $k > 9$.

7 Final remarks

Our results confirm the following scenario for the problems of counting or detecting copies of a graph H on k vertices with an independent set of size s. In the induced subgraph isomorphism case, these problems seem to be equally hard for all such H, independently of their density and the size of s (see Theorem 5.1 and for its special four-vertex cases also [14]). On the contrary, in the subgraph isomorphism case, it seems that the larger s, the better upper bounds we can obtain (recall our two main results and [22]).

The extreme case when the pattern graph is just a set of k isolated vertices fully agrees with the scenario. In the induced subgraph isomorphism case, the problems of counting and detecting are equally hard as those for the k-clique while in the subgraph isomorphism case they become trivial.

Incidentally, our $O(n^{\omega})$-bound for $H \in H_4 \setminus \{K_4\}$ coincides with the best known running time for detecting or counting copies of K_3 while our $O(n^{\omega(2,1,1)})$-bound for $H \in H_5 \setminus \{K_5\}$ coincides with the best known running time for detecting or counting copies of K_4.

Of course, the ultimate goal is to improve the upper time bounds for complete graphs, even improvements for K_4 or K_5 could lead to such a global improvement.

However, there is a large spectrum of applications where detecting or counting non-necessarily complete small pattern graphs occurs. Very recent examples of applications include identification of computational patterns in automatic design of processor systems [24], motif counting and discovery in biomolecular networks [1], structure discovery in protein networks [5].

8 Acknowledgments

The authors are very grateful to unknown referees whose comments helped to improve the paper.

References

[1] N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. C. Sahinalp , *Biomolecular network motif counting and discovery by color coding*, Bioinformatics (ISMB 2008), 24(13) (2008), pp. 241–249 .

[2] N. Alon and M. Naor , *Derandomization, Witnesses for Boolean Matrix Multiplication and Construction of Perfect Hash Functions*, Algorithmica, 16(4/5) (1996), pp. 434–449.

[3] N. Alon, R. Yuster, and U. Zwick , *Color-coding*, Journal of the ACM, 42(4) (1995), pp. 844–856.

[4] N. Alon, R. Yuster, and U. Zwick , *Finding and counting given length cycles*, Algorithmica, 17(3) (1997), pp. 209–223.

[5] P. Bachman and Y. Liu , *Structure discovery in PPI networks using pattern-based network decomposition*, Bioinformatics, 25(14) 2009, pp. 1814–1821.

[6] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto , *Counting Paths and Packings in Halves*, In: Fiat, A., Sanders, P. (eds.) ESA 2009, LNCS, vol. 5757 (2009), pp. 578–586.

[7] D. Coppersmith , *Rectangular matrix multiplication revisited*, Journal of Complexity, 13 (1997), pp. 42–49.

[8] F. Eisenbrand and F. Grandoni , *On the complexity of fixed parameter clique and dominating set*, Theoretical Computer Science, 326 (2004), pp. 57–67.

[9] F. V. Fomin, D. Lokshtanov, V. Raman, B. V. R. Rao, and S. Saurabh , *Faster Algorithms for Finding and Counting Subgraphs*, arXiv:0912.237v1 [cs.DS] 11 Dec 2009.

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

[11] A. W. Goodman , *On Sets of Acquaintances and Strangers at any Party*, The American Mathematical Monthly, 66(9) (1959), pp. 778–783.

[12] X. Huang and V. Y. Pan , *Fast rectangular matrix multiplications and applications*, Journal of Complexity, 14(2) (1998), pp. 257–299.

[13] A. Itai and M. Rodeh , *Finding a minimum circuit in a graph*, SIAM Journal on Computing, 7(4) (1978), 413–423.

[14] T. Kloks, D. Kratsch, and H. Müller , *Finding and counting small induced subgraphs efficiently*, Information Processing Letters, 74(3-4) (2000), 115–121.

[15] W. L. Kocay , *Some new methods in reconstruction theory*, Combinatorial mathematics IX, Lecture Notes in Mathematics 952 (1982), pp. 89–114.

[16] J. Nešetřil and S. Poljak , *On the complexity of the subgraph problem*, Commentationes Mathematicae Universitatis Carolinae, 26(2) (1985), pp. 415–419.

[17] J. Plehn and B. Voigt , *Finding Minimally Weighted Subgraphs*, In: Rolf H. Möhring, (ed.) WG 1990, LNCS, vol. 484 (1991), pp. 18–29.

[18] L. G. Valiant and V. V. Vazirani , *NP is as easy as detecting unique solutions*, Theoretical Computer Science, 47(3) (1986), pp. 85–93.

[19] V. Vassilevska , *Efficient algorithms for clique problems*, Information Processing Letters, 109(4) (2009), pp. 254–257.

[20] V. Vassilevska , *Efficient Algorithms for Path Problems in Weighted Graphs*, PhD thesis, CMU, CMU-CS-08-147, 2008.

[21] V. Vassilevska Williams and R. Williams , *Subcubic Equivalences Between Path, Matrix, and Triangle Problems*, To appear in proc. FOCS 2010.

[22] V. Vassilevska and R. Williams , *Finding, Minimizing, and Counting Weighted Subgraphs*, In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009), ACM (2009), pp. 455–463.

[23] R. Williams , *Finding paths of length k in $O^*(2^k)$ time*, Information Processing Letters, 109(6) (2009), pp. 315–318.

[24] C. Wolinski, K. Kuchcinski, and E. Raffin , *Automatic Design of Application-Specific Reconfigurable Processor Extensions with UPaK Synthesis Kernel*, ACM Transactions on Design Automation of Electronic Systems, 15(1) (2009).

[25] R. Yuster and U. Zwick , *Finding Even Cycles Even Faster*, SIAM J.Discrete Mathematics, 10(2) (1997), pp. 209–222.