SIAM J. DISCRETE MATH.
Vol. 27, No. 2, pp. 892–909

COUNTING AND DETECTING SMALL SUBGRAPHS VIA EQUATIONS*

MIROSŁAW KOWALUK[†], ANDRZEJ LINGAS[‡], AND EVA-MARTA LUNDELL[‡]

Abstract. We present a general technique for detecting and counting small subgraphs. It consists of forming special linear combinations of the numbers of occurrences of different induced subgraphs of fixed size in a graph. These 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 (not necessarily induced) subgraph isomorphic to H can be done in time $O(n^{\omega(\lceil (k-s)/2\rceil,1,\lfloor (k-s)/2\rfloor)})$, 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 (not necessarily induced) subgraphs isomorphic to H can be done in the same time, i.e., in time $O(n^{\omega(\lceil (k-2)/2\rceil,1,\lfloor (k-2)/2\rfloor)})$.

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 2.373. 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 3.257. Finally, we derive input-sensitive variants of our time upper bounds. They are partially expressed in terms of the number m of edges of the input graph and do not rely on fast matrix multiplication.

Key words. subgraph and induced subgraph isomorphism, counting and detection of subgraphs, linear equations, exact algorithms, rectangular matrix multiplication

AMS subject classifications. 68W01, 68W40, 68Q25, 68R10, 05C50

DOI. 10.1137/110859798

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 *subgraph isomorphism* and *induced subgraph isomorphism* problems, respectively. Their decision, finding, counting, and even enumeration versions (see the preliminaries) 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 Hamiltonian path problems [12]. For arbitrary graphs, they are known to admit polynomial-time solutions solely when the other graph, often termed 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 and the input graph has n vertices and m edges.

*Received by the editors December 21, 2011; accepted for publication (in revised form) February 12, 2013; published electronically May 2, 2013. A preliminary and shorter version of this paper appeared in *Proceedings of the* 22*nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA* 2011*)*, San Francisco, CA, 2011.
http://www.siam.org/journals/sidma/27-2/85979.html

[†]Institute of Informatics, Warsaw University, Warsaw, Poland (kowaluk@mimuw.edu.pl). This author's research was supported by grant N20600432/0806 from the Polish Ministry of Science and Higher Education.

[‡]Department of Computer Science, Lund University, 22100 Lund, Sweden (Andrzej.Lingas@cs.lth.se, Eva-Marta.Lundell@cs.lth.se). The second author's research was supported in part by VR grant 621-2008-4649.

1.1. Related results on subgraph isomorphism with a fixed pattern graph. Three decades ago, Itai and Rodeh [15] demonstrated that detection and counting in the case when the pattern graph is a triangle can be done in $O(n^{\omega})$ time, or alternatively in $O(m^{3/2})$ time. The finding variant can be solved within $O(n^{\omega})$ time by self-reducibility, e.g., partition the vertex set into four roughly equal parts and run detection on all four possible unions of three parts and then recourse on one of the unions that returns yes.

Next, Chiba and Nishizeki [7] provided an input-sensitive algorithm for listing all triangles in a graph G running in time $O(a(G)m)$, where $a(G)$ is the arboricity of G, i.e., the minimum number of edge-disjoint forests into which G can be decomposed. They also generalized their result to include listing of copies of all K_k, $k \geq 3$, in G in $O(ka(G)^{k-2}m)$ time.

Furthermore, Nešetřil and Poljak [19] presented reductions of the variants of the k-clique problem to those of the triangle problem and its generalization to include other k-subgraphs besides k-cliques. Recall that $\omega(p,q,r)$ denotes the exponent of fast arithmetic matrix multiplication of an $n^p \times n^q$ matrix by an $n^q \times n^r$ matrix and ω stands for $\omega(1,1,1)$ (see [8, 9, 14, 18, 24, 21]). Subsequently, Kloks, Kratsch, and Müller [16] and finally Eisenbrand and Grandoni [10] improved on the reductions to show that generally these problems for k-vertex pattern graphs can be solved in time $O(n^{\omega(\lfloor k/3\rfloor,\lceil (k-1)/3\rceil,\lceil k/3\rceil})$, or alternatively in time $O(m^{\omega(\lfloor k/3\rfloor,\lceil (k-1)/3\rceil,\lceil k/3\rceil)/2})$ for $k \geq 6$. This is substantially faster than the $O(n^k)$ time required by an exhaustive enumeration. Recently, Vassilevska and Williams [25] showed that the number of occurrences of a pattern graph with an independent set of size s can be computed in $2^s n^{k-s+3} k^{O(1)}$ time. Importantly, their method is combinatorial, and hence it does not rely on fast matrix multiplication.

There are also known examples of pattern graphs where the decision and finding versions can be solved much faster. Namely, at the beginning of 1990s, Plehn and Voight [20] showed that if the fixed pattern graph has treewidth t, then the decision and finding versions of subgraph isomorphism admit an $O(n^{t+1})$-time solution while those of induced subgraph isomorphism also admit an $O(n^{t+1})$-time solution in the case when the maximum degree in the input graph is constant. Yuster, and Zwick [28] showed in particular that cycles of given even length can be found in $O(n^2)$ time for any fixed even length. In [2] Alon, Yuster, and Zwick introduced the now classical technique of color coding to detect cycles or paths of 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 [3] how to find a triangle in $O(m^{2\omega/(\omega+1)})$ time and how to find a cycle of given length k in an unweighted, directed or undirected, graph in $\mathcal{O}(m^{2-2/k})$ time for even k and in $\mathcal{O}(m^{2-2/(k+1)})$ time for odd k. For even cycles in unweighted, undirected graphs, they also demonstrated that C_{4k} can be found in $\mathcal{O}(m^{2-(1/k-1/(2k+1))})$ time and C_{4k-2} in $\mathcal{O}(m^{2-(1+1/k)/2k})$ time. In particular, their time upper bounds for C_3 through C_6 are $O(m^{1.41})$, $O(m^{1.34})$, $O(m^{1.67})$, and $O(m^{1.63})$, respectively. They also showed in [3] that for $k = 3,\ldots,7$, the number of C_k can be counted in $O(n^{\omega})$ time, extending on the classical result of Itai and Rodeh [15] for triangles. In [16], Kloks, Kratsch, and Müller showed for the induced variant that the occurrences of K_4 can be counted in time $O(m^{(\omega+1)/2}) = O(n^{\omega+1})$, and if the occurrences of some pattern graph on four vertices can be counted in time $T(n)$, then the occurrences of any other pattern graph on four vertices can be counted in $O(n^{\omega} + T(n))$ time. They also showed that counting occurrences of four-vertex pattern graphs different from K_4 can be done in time $O(n^{\omega} + m^{(\omega+1)/2})$ [16].

TABLE 1.1

Selected time upper bounds on detecting, finding, and counting small subgraphs in an undirected, unweighted graph G on n vertices and m edges. $\mathcal{H}_k$ stands for the class of pattern graphs on k vertices, and additional subscripts s and t denote the size of an independent set and the treewidth, respectively.

Subgraph	Time Complexity	Problem	Reference
K_3	$\mathcal{O}(n^{\omega})$	finding	Itai–Rodeh [15]
K_3	$\mathcal{O}(n^{\omega})$	counting	Itai–Rodeh [15]
K_4	$\mathcal{O}(m^{(\omega+1)/2})$	counting	Kloks et al. [16]
$\mathcal{H}_4$	$\mathcal{O}(n^{\omega} + m^{(\omega+1)/2})$	counting	Kloks et al. [16]
$\mathcal{H}_k$	$O(n^{\omega(\lfloor k/3 \rfloor, \lceil (k-1)/3 \rceil, \lceil k/3 \rceil})$	detection	Eisenbrand–Grandoni [10]
$\mathcal{H}_{k,s}$	$\mathcal{O}(n^{k-s+3})$	counting	Vassilevska–Williams [25]
$\mathcal{H}_{k,t}$	$\mathcal{O}(n^{t+1})$	finding	Plehn–Voight [20]
$C_k, k \leq 7$	$\mathcal{O}(n^{\omega})$	counting	Alon et al. [3]
C_k	$\mathcal{O}(n^{\omega} \log n)$	finding	Alon et al. [2]
$K_k \backslash e$	$\mathcal{O}(n^{k-1})$	detection	Vassilevska [23]
P_k	$\mathcal{O}^*(2^k)$	detection	Williams [26]
P_k	$\mathcal{O}^*(\binom{n}{k/2})$	counting	Björklund et al. [5]
$\mathcal{H}_{k,t}$	$\mathcal{O}^*(\binom{n}{k/2} n^{2t})$	counting	Fomin et al. [11]

More recently, Vassilevska [23] demonstrated that an induced subgraph isomorphic to $K_k \backslash e$, 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 $K_4 \backslash e$ from [10, 16]. She also presented relatively fast algorithms for the so-called semicliques in [22]. Williams [26] showed how to find a path of length k in time $\mathcal{O}^*(2^k)$, while Björklund et al. [5] obtained 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 t, Fomin et al. [11] derived algorithms for the decision and counting versions that run in time $\mathcal{O}^*(2^k n^{2t})$ and $\binom{n}{k/2} n^{\mathcal{O}(t \log k)}$, respectively.

Table 1.1 presents some of the aforementioned time upper bounds for detecting, finding, and counting small subgraphs.

1.2. Our contributions. We present a general technique for deriving independent linear dependencies among the numbers of occurrences of different induced subgraphs of fixed size in a host graph. The coefficients at the unknowns corresponding to these numbers in the dependencies are easily computable, while the computation of the right-hand sides of the dependencies reduces to the following *l-neighborhood problem*.

Determine for each (ordered) 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 ith vertex in the l-tuple iff $b(i) = 1$.

We show that the latter problem can be relatively efficiently solved via rectangular matrix multiplication [8, 14, 18].

In [16], Kloks, Kratsch, and Müller described some of the dependencies in the special case of some subgraphs of size 4. 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 [3] to derive their results on counting k-cyclic graphs for $k = 3, \ldots, 7$ rely on a different idea of computing traces of matrix powers.)

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

Assume $k = O(1)$. We show that if for all graphs in $\mathcal{H}_k \setminus \mathcal{H}_k(l)$ their numbers of occurrences either as an induced or a not necessarily induced subgraph of the input graph are known, then the number of occurrences of any $H \in \mathcal{H}_k$ both as an induced and a not necessarily induced subgraph can be computed in time $O(n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$. The upper bound stands for the time required to solve the aforementioned l-neighborhood problem.

In the case $l = k-2$, we show that the knowledge of the number of occurrences of any given graph in the whole $\mathcal{H}_k$ as an induced subgraph is sufficient to compute the number of occurrences of any $H \in \mathcal{H}_k$ both as an induced and a not necessarily induced subgraph in time $O(n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$. (This generalizes the corresponding fact shown for $k = 4$ in [16].)

Our main results utilizing this technique are two new time upper bounds on detecting and counting occurrences of $H \in \mathcal{H}_k(l)$ as (not necessarily induced) subgraphs in the host graph on n vertices. We show that

1. detecting if an n-vertex graph contains a (not necessarily induced) subgraph isomorphic to H can be done in time $O(n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)})$, and that
2. when $l = k-2$, counting the number of (not necessarily induced) subgraphs isomorphic to H can be done in the same time, i.e., in time $O(n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$. (This improves, but only for $k - l = 2$, on the aforementioned general combinatorial counting algorithm of Vassilevska and Williams [25], the running time of which can be rephrased as $O(n^{l+3})$ in terms of our notation. By straightforward calculations, our upper bound is never worse than roughly $n^{k+\omega-4}$, and if $\omega = 2$, then it's roughly n^{k-2}. By generalizing the method of Nešetřil and Poljak [19], one can also count the number of occurrences of H in time $O(n^{r+z\omega})$, where $k = 3z + r$ and $r \in \{0, 1, 2\}$. This yields better time upper bounds than ours for $k > 10$.)

It follows in particular that the counting version can be solved for any $H \in \mathcal{H}_4 \setminus \{K_4\}$ in time $O(n^{\omega})$ and for any $H \in \mathcal{H}_5 \setminus \{K_5\}$ in time $O(n^{\omega(2,1,1)})$, where $\omega < 2.373$ [21, 24] and $\omega(2, 1, 1) < 3.257$ [18].

Finally, we derive input-sensitive variants of our time upper bounds expressed also in terms of the number m of edges of the input graph. Importantly, they do not rely on fast matrix multiplication.

1.3. Organization. In the next section we briefly introduce 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 section 4, we derive our general results on counting and detecting copies of graphs from $\mathcal{H}_k(l)$, including our first main result on detection. Section 5 is devoted to our second main result on fast counting of small nonclique subgraphs. 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 obtain upper bounds on the run-times in our main theorems and derive concrete corollaries on counting copies of graphs from the sets $\mathcal{H}_4(2)$ and $\mathcal{H}_5(3)$, respectively. In section 7, we present the input-sensitive counterparts of our time upper bounds. We conclude with final remarks.

2. Preliminaries.

An *isomorphism* between two graphs F and G is a one-to-one mapping f of the vertices of F onto vertices of G such that $\{u, v\}$ is an edge of F

iff $\{f(u), f(v)\}$ is an edge of G. If $F = G$, then an isomorphism between F and G is called an *automorphism* of F. F is *isomorphic* to G if there is an isomorphism between F and G.

A *subgraph* of a graph $G = (V, E)$ is a graph $G' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$. Such a subgraph G' of G is *induced* if $E' = (V' \times V') \cap E$.

A *subgraph isomorphism* between two graphs F and G is an isomorphism between F and a subgraph of G.

The *detection version* or, equivalently, the *decision version* of the *subgraph isomorphism* problem is to decide for a host graph and a pattern graph if the host graph has a subgraph isomorphic to the pattern graph. The *finding version* of subgraph isomorphism asks for returning a subgraph of the host graph isomorphic to the pattern graph. Finally, the *counting version* of subgraph isomorphism asks for reporting the total number of subgraphs of the host graph isomorphic to the pattern graph. The corresponding versions of *induced subgraph isomorphism* are defined analogously by replacing "subgraph" with "induced subgraph".

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

DEFINITION 2.1. *For a graph $H \in \mathcal{H}_k$ 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 not necessarily induced 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 $N(H, G)$ for $H \in \mathcal{H}_k$ can be reduced to computing $NI(H, G)$ for $H \in \mathcal{H}_k$ and vice versa (e.g., see Theorem 2.3 in [17]). We rephrase this known result in terms of our notation as follows.

FACT 2.1. *For $H \in \mathcal{H}_k$, the equalities $N(H, G) = \sum_{H' \in \mathcal{H}_k} N(H, H') NI(H', G)$ hold. The $|\mathcal{H}_k| \times |\mathcal{H}_k|$ matrix $M = [N(H, H')]_{H, H' \in \mathcal{H}_k}$ is nonsingular and M^{-1} has integer entries.*

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

In this section, we formulate equations with variables corresponding to the number of occurrences of particular induced subgraphs and give a reduction of the problem of computing the right-hand sides of these equations to the l-neighborhood problem (Propositions 3.2 and 3.3, Lemma 3.5). We also simplify the definition of the coefficients in the equations for noncliques (Lemma 3.6) and prove that appropriate sets of such equations are linearly independent (Lemma 3.7).

Let H be a graph on k vertices and let H_{sub} be an induced subgraph of H on l vertices such that the $k - l$ vertices in $H \backslash H_{sub}$ form an independent set. Consider the family of all supergraphs H' of H (including H) in $\mathcal{H}_k$ such that H' has the same vertex set as H, H_{sub} is also an induced subgraph of H', and the set of edges between H_{sub} and $H' \backslash H_{sub}$ is the same as that between H_{sub} and $H \backslash H_{sub}$. We denote this family by $\mathcal{H}_k(H_{sub}, H)$ and its subfamily of single representatives of all isomorphism classes in $\mathcal{H}_k(H_{sub}, H)$, i.e., its intersection with $\mathcal{H}_k$, by $\mathcal{SH}_k(H_{sub}, H)$. For an illustration see Figure 3.1(a), (b).

The main idea of our method relies on the fact that a linear combination of the numbers $NI(H', G)$ of induced copies of $H' \in \mathcal{SH}_k(H_{sub}, H)$ in the host graph G can be computed relatively efficiently without the explicit knowledge of these numbers.

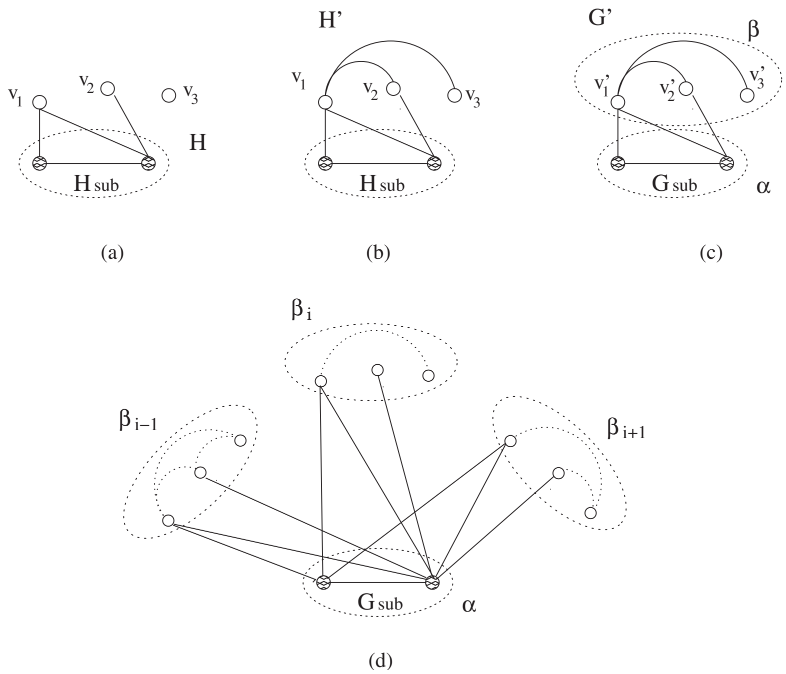

FIG. 3.1. (a) *An example of a graph H composed of the induced subgraph H_{sub} and the vertex set $\{v_1, v_2, v_3\}$ that forms an independent set in H.* (b) *An example of a supergraph H' of H_{sub} in $\mathcal{H}_k(H_{sub}, H)$.* (c) *An example of a subgraph G' of G induced by an l-tuple α of vertices in G corresponding to H_{sub} jointly with a $(k-l)$-tuple β of vertices in G corresponding to the independent set $\{v_1, v_2, v_3\}$ in H.* (d) *An example of a set of $(k-l)$-tuples of vertices in G which are connected with the l-tuple α by edges corresponding to those between $H \backslash H_{sub}$ and H_{sub}.*

DEFINITION 3.1. *For $H' \in \mathcal{SH}_k(H_{sub}, H)$, let $B(H_{sub}, H')$ denote the number of isomorphisms between H_{sub} and an induced subgraph of H', say, H^f_{sub}, that can be extended to an isomorphism between H and the subgraph of H' consisting of H^f_{sub}, all edges of H' incident to H^f_{sub}, and all the remaining vertices of H'.*

$B(H_{sub}, H')$ is the coefficient of the corresponding term $NI(H', G)$ in the aforementioned linear combination. This coefficient can be easily calculated in $O(1)$-time by enumerating subgraph isomorphisms between H and H' (see proof of Theorem 4.1).

To form an equation, we shall place the linear combination

$$\sum_{H' \in \mathcal{SH}_k(H_{sub}, H)} B(H_{sub}, H') NI(H', G)$$

on the left-hand side, treating $NI(H', G)$ as unknowns, and its value for explicit values of $NI(H', G)$ on the right-hand side of the equality. (In a latter formal definition of our equations, i.e., in Definition 3.4, we replace $NI(H', G)$ with corresponding variables on the left-hand side.)

Example 3.1. Let H be a graph on three vertices with exactly two edges, and let H_{sub} be just K_1, i.e., a single vertex graph. Then $\mathcal{SH}_3(H_{sub}, H)$ consists of H and K_3. Let T_2 and T_3 be unknowns that represent $NI(H, G)$ and $NI(K_3, G)$, respectively. Since $B(K_1, H) = 1$ and $B(K_1, K_3) = 3$, we obtain the following linear combination on the left-hand side of the corresponding equation: $T_2 + 3T_3$.

We prove that the right-hand side of such an equation can be computed efficiently in three stages. First, in Proposition 3.2, we prove that the right-hand side can be expressed as the number of equivalence classes of $(k-l)$-tuples of vertices in G. Then, in Proposition 3.3, we show that the latter number can be efficiently reduced to the l-neighborhood problem defined in the introduction. Later, in section 6, we show that the l-neighborhood problem can be solved in time $O(2^l n^{\omega(\lceil l/2\rceil,1,\lfloor l/2\rfloor)})$ for $l \geq 2$.

We shall call relevant an (ordered) l-tuple α of vertices of G such that the mapping assigning the jth vertex in the tuple to the jth vertex in H_{sub} is an isomorphism between H_{sub} and the subgraph G_{sub} of G induced by the tuple.

For all relevant l-tuples α, we shall count the number of equivalence classes of (ordered) $(k-l)$-tuples β of vertices $v'_1, \dots, v'_{k-l}$ in $G \backslash G_{sub}$, where the neighborhood of v'_i in G_{sub} corresponds to that of the ith vertex of $H \backslash H_{sub}$ in H_{sub} under the isomorphism between G_{sub} and H_{sub}. (Equivalently, we shall count the number of equivalence classes of (ordered) $(k-l)$-tuples β of vertices $G \backslash G_{sub}$ such that the mapping assigning the jth vertex of the k-tuple resulting from the concatenation of α with β to the jth vertex of H, where the vertices of H_{sub} have numbers 1 through l, is a subgraph isomorphism between H and an induced subgraph of G isomorphic to a graph in $\mathcal{SH}_k(H_{sub}, H)$.)

Two $(k-l)$-tuples β_1 and β_2 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_{sub}.

PROPOSITION 3.2. *The total number of the equivalence classes of $(k-l)$-tuples summed over all relevant l-tuples α is equal to* $\sum_{H' \in \mathcal{SH}_k(H_{sub},H)} B(H_{sub}, H') \times NI(H', G)$.

Proof. Consider an equivalence class C for a relevant l-tuple α. It follows from the definition of the equivalence classes that for any $\beta \in C$, the vertices in α and β induce the same subgraph G' of G. Next, consider the mapping assigning the jth vertex of the combined k-tuple $\alpha\beta$ to the jth vertex of H, where vertices of H_{sub} have numbers 1 through l. By the definition of the equivalence classes, this mapping is a subgraph isomorphism between H and G' extending the isomorphism between H_{sub} and the subgraph of G induced by α. Hence, G' is isomorphic to a graph H' in $\mathcal{SH}_k(H_{sub}, H)$. It follows also that the mapping assigning the ith vertex of H_{sub} to the ith vertex of the l-tuple which is the image of α under the isomorphism between G' and H' is one of the isomorphisms accounted into the value of $B(H_{sub}, H')$.

On the contrary, consider a set of k vertices in G which induces a subgraph G' of G isomorphic to a graph H' in $\mathcal{SH}_k(H_{sub}, H)$.

By Definition 3.1, there are $B(H_{sub}, H')$ l-tuples α such that the mapping assigning the ith vertex of H_{sub} to the ith vertex of α is an isomorphism between H_{sub} and the subgraph of G' induced by α that can be extended to a subgraph isomorphism between H and G'. For each such subgraph isomorphism f, let β_f be the $(k-l)$-tuple whose qth element is the image of the $(l+q)$th vertex in H. All such $(k-l)$-tuples β_f can be obtained one from another by a collection of permutations applied to the groups of vertices that have the same neighborhood in G_{sub}. Thus, they fall in the same equivalence class C with respect to α.

Furthermore, any $(k-l)$-tuple γ which is a permutation of $\beta \in C$, where the neighborhood of ith vertex of γ in the subgraph G_{sub} induced by α corresponds to that of the ith vertex of $H \backslash H_{sub}$ in H_{sub} under the isomorphism between H_{sub} and G_{sub}, jointly with α defines one of the subgraph isomorphisms f and falls in the class C with respect to α.

Finally, no other $(k-l)$-tuple δ that together with the l-tuple α yields a subgraph isomorphism between H and a k-vertex induced subgraph of G different from G' can fall in the class C with respect to α. Simply, such a δ had to consist of a different set of $k-l$ vertices.

We conclude that each k-vertex set inducing a subgraph isomorphic to $H' \in \mathcal{SH}_k(H_{sub}, H)$ contributes $B(H_{sub}, H')$ distinct equivalence classes. ∎

We shall show that computing the total number of the equivalence classes easily reduces to the l-neighborhood problem defined in the introduction. We shall denote the time required to solve the l-neighborhood problem by $T_l(n)$.

PROPOSITION 3.3. *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 $v_i' \in G \backslash G_{sub}$ in the subgraph G_{sub} induced by a relevant l-tuple α, corresponding to those $v_i \in H \backslash H_{sub}$ for $i = 1, \dots, k-l$ in the subgraph H_{sub} under the isomorphism between G_{sub} and H_{sub} (see Figure 3.1). Each of these neighborhoods can be identified with a binary vector of length l, which we call the *type* of the neighborhood.

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

For an l-tuple α, let $n_t(\alpha)$ be the number of vertices in $G \backslash G_{sub}$ having the neighborhood type t in G_{sub}. Then, the number of all equivalence classes over all relevant l-tuples α is given by the sum $\sum_\alpha \prod_t \binom{n_t(\alpha)}{o_t}$. If the numbers $n_t(\alpha)$ are given, then this sum can be easily computed in $O(n^l(k-l))$ time. It is sufficient to observe by the definition of the l-neighborhood problem that these numbers can be determined by solving the latter problem. ∎

The easily computable values of $B(H_{sub}, H')$ (recall $k = O(1)$) can be treated as coefficients at the unknowns which correspond to $NI(H', G)$ for $H' \in \mathcal{SH}_k(H_{sub}, H)$, 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 $l \in \{1, \dots, k-1\}$, denote the set of such equations, each one with $|\mathcal{SH}_k(H_{sub}, H)|$ unknowns corresponding to $NI(H', G)$ for $H' \in \mathcal{SH}_k(H_{sub}, H)$, respectively.

DEFINITION 3.4. *For $H \in \mathcal{H}_k(l)$, the set $Eq(H, l)$ consists of the following equations in one-to-one correspondence with induced subgraphs H_{sub} of H on l vertices:*

$$\sum_{H' \in \mathcal{SH}_k(H_{sub}, H)} B(H_{sub}, H') x_{H',G} = \sum_{H' \in \mathcal{SH}_k(H_{sub}, H)} B(H_{sub}, H') NI(H', G),$$

where $H \backslash H_{sub}$ is an independent set in H.

Note that in these equations, the variables $x_{H',G}$ correspond to $NI(H', G)$, respectively.

By Propositions 3.2 and 3.3, we obtain the following lemma.

LEMMA 3.5. *For $H \in \mathcal{H}_k(l)$, the right-hand side of an equation in $Eq(H, l)$ can be evaluated in time $O(n^l(k-l)+T_l(n))$.*

For $H' \in \mathcal{SH}_k(H_{sub}, H)$, let $A(H_{sub}, H')$ be the number of automorphisms of H' divided by the number of automorphisms of H' *that are identity on* H_{sub}. The following lemma will be useful in evaluation of the coefficients of equations in $Eq(H, k-2)$, where $H \in \mathcal{H}_k(k-2)$.

LEMMA 3.6. *Let $H \in H_k(k-2)$, let H_{sub} be an induced subgraph of H on l vertices such that the two vertices in $H \backslash H_{sub}$ form an independent set, and let $H' \in \mathcal{SH}_k(H_{sub}, H)$. The equality $A(H, H') = B(H, H')$ holds.*

Proof. Let $H' \in \mathcal{SH}_k(H_{sub}, H)$, and let $\mathcal{F}$ be the set of all isomorphisms f between H_{sub} and an induced subgraph of H' satisfying the requirements from Definition 3.1.

Consider an extension f' of $f \in \mathcal{F}$ to an isomorphism between H and the subgraph of H' composed of H^f_{sub}, all edges of H' incident to H^f_{sub}, and all other vertices of H'. If $H' = H$, then f' is an automorphism of H'. Otherwise, H' is the other member of $\mathcal{H}_k(H_{sub}, H)$ obtained by adding the edge between the two independent vertices of H outside H_{sub}. Then, f' is also an automorphism of H' since the only edge in H' not incident to H^f_{sub} has to connect the f' images of the aforementioned two independent vertices in H.

It follows that each $f \in \mathcal{F}$ can be identified with the class of all automorphisms of H' that are equal each other on H_{sub}. Conversely, each such a class yields a distinct member in $\mathcal{F}$.

We conclude that $B(H_{sub}, H')$ is equal to the number of automorphisms of H' divided by the number of automorphisms of H' that are identity on H_{sub}. $\square$

See Examples 3.2 and 3.3 for examples of systems of equations in $Eq(H, l)$, where $H \in \mathcal{H}_k(l)$. The equations in Example 3.3 can be regarded as an extension of those for connected $H \in \mathcal{H}_4$ given in [16].

Example 3.2. The following is an example of equations in $Eq(H, 1)$, where $H \in \mathcal{H}_3(1)$ (corresponding to those in [13]).

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 $i = 0, 1, 2, 3$, let t_i denote a graph on three vertices that contains exactly i edges. Thus in particular t_0 consists of three K_1, i.e., 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) $A(K_1, t_0)x_{t_0,G} + A(K_1, t_1)x_{t_1,G} =$
$\#\{(v, \{u, w\}) \mid \{v, u, w\} \subset V \ \wedge \{\{v, u\}, \{v, w\}\} \cap E = \emptyset\}$,

(b) $A(K_1, t_1)x_{t_1,G} + A(K_1, t_2)x_{t_2,G} =$
$\#\{(v, (u, w)) \mid \{v, u, w\} \subset V \wedge \{v, u\} \in E \ \wedge \ \{v, w\} \notin E\}$,

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

By computing the coefficients $A(K_1, t_i)$, letting the unknows T_i represent $NI(t_i, G)$ instead of $x_{t_i,G}$ for $i = 0, 1, 2, 3$, 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 3.3. Assume the notation from Example 3.2. Let quadruples stand for unordered four-element sets in this example. 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 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 nonincident 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;
- $Q_{\curlywedge}$ denote the number of quadruples in G that induce exactly a star composed of three incident edges (claw);
- $Q_{\triangleright\cdot}$ 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);
- $Q_{\boxtimes}$ 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 10 linearly independent left-hand sides of simplified equations respectively in $Eq(H_s, 2)$, where H_s is a subgraph of K_4 counted in Q_s, and for simplicity Q_s stand also for the variable corresponding to Q_s. The right-hand sides of these equations can be computed in $O(n^\omega)$ time. In part, these equations coincide with the equations for connected Q_s presented in [16]. It is indicated in parentheses whether K_2 or an independent set on two vertices, denoted by I_2, is respectively used as H_{sub}.

1. $12Q_0 + 2Q_{|}$ (I_2),
2. $2Q_{|} + 4Q_{||}$ (K_2),

3–4. $2Q_{\bigwedge} + 6Q_{\curlywedge}$ (I_2), $8Q_{||} + 2Q_{\bigsqcup}$ (I_2),

5–7. $6Q_{\triangleright\cdot} + 2Q_{\triangleright-}$ (K_2), $4Q_{\bigsqcup} + 4Q_{\triangleright-}$ (I_2),
$6Q_{\curlywedge} + 2Q_{\triangleright-}$ (K_2),

8. $4Q_{\square} + 2Q_{\boxslash}$ (I_2),
9. $4Q_{\triangleright-} + 8Q_{\boxslash}$ (K_2),
10. $2Q_{\boxslash} + 12Q_{\boxtimes}$ (K_2).

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

LEMMA 3.7. *For each H in $\mathcal{H}_k(l)$, pick an arbitrary equation from $Eq(H, l)$. The resulting system of $|\mathcal{H}_k(l)|$ equations is linearly independent.*

Proof. Sort the graphs in $\mathcal{H}_k$ so that the number of edges is nondecreasing and the graphs in $\mathcal{H}_k(l)$ form a prefix of the sorted sequence. Let B be the $|\mathcal{H}_k(l)| \times |\mathcal{H}_k|$ matrix corresponding to the left-hand side of the equations in $Eq(H, l)$ for $H \in \mathcal{H}_k(l)$ with the rows of B corresponding to $H \in \mathcal{H}_k(l)$ and the columns of B corresponding to $H' \in \mathcal{H}_k$ sorted in the aforementioned way. It follows from the definition of the equations that the leftmost maximal square submatrix M of B of size $|\mathcal{H}_k(l)| \times |\mathcal{H}_k(l)|$ has nonzero elements along the diagonal starting from the top-left corner. Furthermore, below the diagonal there are only zeros, since each supergraph H' of H on the same vertex set, that is identical on H_{sub} and the edges between H_{sub} and $H \backslash H_{sub}$, cannot have fewer or equally many edges as H unless $H' = H$. ∎

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

Proof. We can enumerate all subgraph isomorphisms between two graphs on k vertices in $O(k^2 k!)$ time. Hence, computing all the possible coefficients $B(H_{sub}, H)$ on the left sides of the equations from Lemma 3.7 takes time $O(|\mathcal{H}_k(l)||\mathcal{H}_k|k^2k!)$. It follows by Lemma 3.5 that forming the aforementioned equations takes time $O(|\mathcal{H}_k(l)||\mathcal{H}_k|k^2k! + |\mathcal{H}_k(l)|n^l(k-l)+T_l(n))$. If for all $H \in \mathcal{H}_k \backslash \mathcal{H}_k(l)$, the values $NI(H,G)$ are known, then we can substitute these values for the corresponding variables in the aforementioned equations.

Assume the definition of the matrix B from the proof of Lemma 3.7. Note that the aforementioned substitutions do not affect the leftmost maximal square submatrix M of the matrix B. Since M has zeros below the diagonal starting from the top-left corner, we infer that the resulting $|\mathcal{H}_k(l)|$ equations with $|\mathcal{H}_k(l)|$ unknowns are also linearly independent. Hence, we can solve the resulting equations completely in $O(|\mathcal{H}_k(l)|^3)$ time. It remains to apply Fact 2.1 to obtain all the values $N(H', G)$ as well. ∎

Let $H = (V_H, E_H)$ and $G = (V_G, E_G)$. 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 we should multiply $NI(H', G)$ with the number of automorphisms of H'. The latter can be computed by checking all permutations of vertices in $O(k!k^2)$ time.

Marginally, Theorem 4.1 can be extended to the following form, symmetric with respect to $NI(H, G)$ and $N(H, G)$, by Fact 2.1.

THEOREM 4.2. *If for all $H \in \mathcal{H}_k \backslash \mathcal{H}_k(l)$ either the values $N(H,G)$ or the values $NI(H,G)$ are known, then for all $H' \in \mathcal{H}_k$, 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 $H \in \mathcal{H}_k \backslash \mathcal{H}_k(l)$. Form the initial $|\mathcal{H}_k(l)|$ linearly independent equations with $|\mathcal{H}_k|$ unknowns corresponding to $NI(H', G)$, where $H' \in \mathcal{H}_k$, as in the proof of Theorem 4.1. Let B be the $|\mathcal{H}_k(l)| \times |\mathcal{H}_k|$ matrix of coefficients of the left-hand sides of the aforementioned equations. By Fact 2.1, these equations can be transformed into another set of $|\mathcal{H}_k(l)|$ equations with $|\mathcal{H}_k|$ unknowns corresponding to $N(H', G)$, where $H' \in \mathcal{H}_k$. The matrix of coefficients of the left-hand sides of the new set of equations is the matrix product of B with the inverse of the matrix M given in Fact 2.1. Since B has rank $|\mathcal{H}_k(l)|$ and M is nonsingular, the product matrix has also rank $|\mathcal{H}_k(l)|$. Thus, the new set of $|\mathcal{H}_k(l)|$ equations is also linearly independent. Note also that each of the new equations corresponding to an original equation in $Eq(H_{sub}, H)$ will have a nonzero coefficient solely at $N(H,G)$ and $N(H', G)$, where H' is a supergraph of H in $\mathcal{H}_k$, by the analogous property of the original equations and Fact 2.1.

Now, if we substitute the known values $N(H,G)$ for the corresponding variables in these new equations, we obtain $|\mathcal{H}_k(l)|$ equations with $|\mathcal{H}_k(l)|$ unknowns. The resulting equations are also linearly independent by the arguments analogous to that in the proof of Lemma 3.7. Hence, we can solve them completely to obtain all values $N(H', G)$ for $H' \in \mathcal{H}_k$. By symmetrically applying Fact 2.1, we also obtain all values $NI(H', G)$ for $H' \in \mathcal{H}_k$. ∎

For the problem of deciding whether the input graph G has a subgraph isomorphic to a given $H \in \mathcal{H}_k \backslash \mathcal{H}_k(l)$, we obtain the following stronger result (our first main result).

THEOREM 4.3. *For $k = O(1)$ and any $H \in \mathcal{H}_k(l)$, one can decide whether $N(H, G) = 0$ in time $O(n^l + T_l(n))$.*

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

If $H_1 \in \mathcal{H}_k(l)$, we consider the equation in $Eq(H,l)$ in the set of equations from the proof of Lemma 3.7 and Theorem 4.1. Its left-hand 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 $\mathcal{H}_k$, 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 3.5, we can decide whether 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 $H_1 \notin \mathcal{H}_k(l)$, 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 $\mathcal{H}_k(l)$. 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 to the previous case, by computing the right-hand side of the equation in time $O(n^l + T_l(n))$, we can decide whether 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 $H_1 \in \mathcal{H}_k$ is $O(1)$, the total time complexity remains $O(n^l + T_l(n))$. $\square$

Note that we can also estimate $N(H,G)$ for $H \in \mathcal{H}_k(l)$ 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 $\mathcal{H}_k$, 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 an 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 $H \in \mathcal{H}_k(k-2)$ from the proofs of Theorem 4.1 and Lemma 3.7 can be transformed to an equivalent set of equations whose left-hand 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 $NI(K_k,G)$, where m_H stands for the number of edges of H, and whose right-hand sides are computable in time $O(n^{k-2} + T_{k-2}(n))$.*

Proof. Consider the set S of linearly independent equations from $Eq(H,k-2)$, $H \in \mathcal{H}_k(k-2)$ from the proofs of Theorem 4.1 and Lemma 3.7. By the structure of these equations, they can be easily transformed into the set of equations with the left-hand 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-hand 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 $F \in \mathcal{H}_k$, let $aut(F)$ be the number of automorphisms of F and let $autid(H_{sub},F)$ be the number of automorphisms of F that are identity on H_{sub}.

Note that for $F \in \mathcal{H}_k$, $N(F,K_k) = k!/aut(F) = aut(K_k)/aut(F)$ holds.

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

Recall Lemma 3.6. Consider an original equation whose left-hand side is of the form $A(H_{sub},H)x_H + A(H_{sub},H')x_{H'}$, where H_{sub} 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 $A(H_{sub}, F) = aut(F)/autid(H_{sub}, F)$ for $F \in \{H, H'\}$. Note also that if there is an automorphism of $F \in \{H, H'\}$ in $autid(H_{sub}, F)$ that is not identity on F, then the two vertices of F outside H_{sub} have to have the same neighborhood in H_{sub}. It follows that $autid(H_{sub}, H) = autid(H_{sub}, H')$.

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

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

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

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

By $autid(H_{sub}, H') = autid(H_{sub}, H)$ and straightforward simplifications, we obtain the induction hypothesis for $F = H$. $\square$

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

THEOREM 5.2. *For any $H \in \mathcal{H}_k$, if the value of $NI(H, G)$ is known, then for all $H' \in \mathcal{H}_k$, 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. $\square$

Fact 2.1 combined with Lemma 5.1 yields our main result in this section.

THEOREM 5.3. *For any $H \in \mathcal{H}_k(k-2)$, 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 $H' \neq K_k$, let $C_{H'}$ be the right-hand side of the normalized equation in $Eq(H', k-2)$ with variables $x_{H'}$ and x_{K_k} in Lemma 5.1. Note that $C_{H'}$ can be computed in time $O(n^{k-2} + T_{k-2}(n))$ by Lemma 5.1. For convention, we set $C_{K_k} = 0$. Let $k' = \binom{k}{2}$. Since $x_{H'}$ corresponds to $NI(H', G)$, we obtain the following equality:

$$NI(H', G) = C_{H'} + (-1)^{k' - m_{H'}} N(H', K_k) NI(K_k, G).$$

For $H' \in \mathcal{H}_k$, we shall denote the set of edges of H' by $E_{H'}$ and its cardinality by $m_{H'}$. Let $H \in \mathcal{H}_k(k-2)$. By combining the expression of $N(H, G)$ in terms of $NI(H', G)$, where H' ranges over supergraphs of H in $\mathcal{H}_k$, given in Fact 2.1 with the aforementioned equalities for $NI(H', G)$, we obtain

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

where $C = \sum_{H' \in \mathcal{H}_k \& E_H \subseteq E_{H'}} N(H, H') C_{H'}$ 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

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

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 (see the introduction) 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 nonneighbors 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 kth 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 $A[t_1,k]$ 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 and 5.3, and observing that $\omega(\lceil l/2\rceil, 1, \lfloor l/2\rfloor) \ge l$, we obtain the following more explicit formulation of our main results.

THEOREM 6.2. *For $k = O(1)$ and any $H \in \mathcal{H}_k(l)$, one can decide whether $N(H,G) = 0$ in time $O(n^{\omega(\lceil l/2\rceil,1,\lfloor l/2\rfloor)})$.*

By [8, 14], when $1 \le 0.294\ l/2 = 0.147l$ and so if $l \ge 7$, then the time upper bound in Theorem 6.2 does not exceed $O(n^l)$.

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

COROLLARY 6.4. *For all $H \in \mathcal{H}_4 \backslash \{K_4\}$, the numbers $N(H, G)$ can be computed in $O(n^{\omega})$ time.*

COROLLARY 6.5. *For all $H \in H_5 \backslash \{K_5\}$, the numbers $N(H, G)$ can be computed in $O(n^{\omega(2,1,1)})$ time.*

Recently, Le Gall has shown that $\omega(2, 1, 1) < 3.257$ [18].

In the particular case of a few graphs termed 4-cyclic by Alon, Yuster, and Zwick in [3], Corollary 6.4 coincides with their result stating that for $k = 3, \ldots, 7$ and any k-cyclic graph H, $N(H, G)$ can be computed in $O(n^{\omega})$ time [3]. The k-cyclic graphs form a narrow family of sparse graphs in $\mathcal{H}_k$ that are homomorphic images of C_k.

To estimate $O(n^{\omega(\lceil (k-2)/2 \rceil, 1, \lfloor (k-2)/2 \rfloor)})$ the following facts proved by Coppersmith [8] and Huang and Pan [14] are useful.

FACT 6.1 (see [8, 14]). *Let $\alpha = sup\{0 \le t \le 1 : \omega(1, t, 1) = 2 + o(1)\} < 0.294$. Then $\omega(1, t, 1) \le 2 + o(1)$ for $t \in [0, \alpha]$ and $\omega(1, t, 1) = 2 + \frac{\omega - 2}{1 - \alpha}(t - \alpha) + o(1)$ for $t \in [\alpha, 1]$.*

With more work, Huang and Pan [14] derived the following generalization of Fact 6.1.

FACT 6.2 (see [14]). *Let α be defined as in Fact 6.1. Suppose $0 \ge t \le 1 \le r$. Then $\omega(t, 1, r) = r + 1 + o(1)$ for $t \in [0, \alpha]$ and $\omega(t, 1, r) = r + 1 + \frac{\omega - 2}{1 - \alpha}(r - \alpha) + o(1)$ for $t \in [\alpha, 1]$.*

By combining the inequality $\omega(p, q, r) \le a\omega(p/a, q/a, r/a)$ for $a \ge 1$ with Fact 6.1 for even l and with Fact 6.2 for odd l, we obtain the following estimation of the run-times in Theorems 6.2 and 6.3.

Remark 6.1. *For even l, $n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)} \le n^{0.922l + 0.533}$ holds, while for odd l, $n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)} \le n^{0.922l + 1.533}$ holds.*

7. Detecting and counting fixed subgraphs in sparse graphs. Recall the proof of Proposition 3.3. Given a list L of the relevant l-tuples, one can also solve the l-neighborhood problem in time proportional to $|L|n$. Hence, we obtain the following counterpart of Proposition 3.3.

PROPOSITION 7.1. *Let $k = O(1)$, $H \in \mathcal{H}_k(l)$, and let H_{sub} be a subgraph of H such that $H \backslash H_{sub}$ forms an independent set of size $k - l$. Suppose that all induced subgraphs of G isomorphic to H_{sub} can be listed in time $T_{H_{sub}}$. Then the right-hand side of the equation in $Eq(H, l)$ with H_{sub} (see section 3) can be computed in time $O(T_{H_{sub}} + NI(H_{sub}, G)n)$.*

In particular, if $H_{sub} = K_l$, then we have $T_{H_{sub}} = O(a(G)^{l-2}m)$ by [7]. (Recall that $a(G)$ stands for the arboricity of G and m for the number of edges in G.) Hence, using also Theorem 6.1, we obtain the following lemma.

LEMMA 7.2. *Let $k = O(1)$ and $H \in \mathcal{H}_k(l)$. The right-hand side of the equation in $Eq(H, l)$ with $H_{sub} = K_l$ can be computed in time $O(\min\{a(G)^{l-2}mn, n^{\omega(\lceil l/2 \rceil, 1, \lfloor l/2 \rfloor)}\} + a(G)^{l-2}m)$.*

We can also list induced copies of H_{sub} having relatively large maximum matching substantially faster than in $O(n^l)$ time when the input graph G is sparse. Suppose that H_{sub} has a matching of size q. It follows that the relevant l-tuples of vertices inducing a subgraph isomorphic to H_{sub} can be generated in time at most proportional to the number of pairs composed of a q-tuple of edges and $(l - 2q)$-tuple of vertices jointly inducing a subgraph isomorphic to H_{sub}. Hence, assuming $l = O(1)$, we obtain $T_{H_{sub}}(n) = O(m^q n^{l-2q})$. Clearly, we have also $NI(H_{sub}, G) \le O(m^q n^{l-2q})$ in this case. By plugging the aforementioned upper bounds into Proposition 7.1 and using Theorem 6.1, we obtain the following lemma.

LEMMA 7.3. *Let $k = O(1)$ and $H \in \mathcal{H}_k(l)$. The right-hand side of the equation in $Eq(H, l)$ with H_{sub} having a matching of size q can be computed in time $O(m^q n^{l-2q} + \min\{m^q n^{l-2q+1},\ n^{\omega(\lceil l/2\rceil,1,\lfloor l/2\rfloor)}\})$.*

Observe that our main results on detection (Theorems 4.3 and 6.2) and counting (Theorems 5.3 and 6.3) hold if we restrict the set of equations used in their proofs to the following single representatives of $Eq(\tilde{H}, l)$:

1. $\tilde{H} \in \mathcal{H}_k(l)$ can be decomposed into a supergraph H^* of our particular H_{sub} having exactly l vertices so that $\tilde{H}\backslash H^*$ forms an independent set on $k - l$ vertices.
2. The aforementioned supergraph H^* plays the role of H_{sub} in the single equation from $Eq(\tilde{H}, l)$.

Simply, the aforementioned proofs use only equations for supergraphs of H in $\mathcal{H}_k(l)$, which in turn are included in the set of $\tilde{H}$'s.

Since H^* as a supergraph of H_{sub} has also a matching of size at least q, the time upper bound of Lemma 7.3 holds also for the equation in $Eq(\tilde{H}, l)$.

Summarizing, we obtain the following sparse extensions of Theorems 6.2 and 6.3 by Lemma 7.2 and Lemma 7.3, respectively.

THEOREM 7.4. *Let $k = O(1)$ and $H \in \mathcal{H}_k(l)$. If H is decomposable into a clique on l vertices and an independent set on $k - l$ vertices and possibly some edges between these two subgraphs, then one can decide whether $N(H, G) = 0$ in time $O(a(G)^{l-2}m) + \min\{a(G)^{l-2}mn,\ n^{\omega(\lceil l/2\rceil,1,\lfloor l/2\rfloor)}\})$. Furthermore, if $k - l = 2$, then one can also compute $N(H, G)$ in time $O(a(G)^{k-4}m + \min\{a(G)^{k-4}mn,\ n^{\omega(\lceil (k-2)/2\rceil,1,\lfloor (k-2)/2\rfloor)}\})$.*

Note that if H satisfies the requirements of Theorem 7.4, then it is in particular a split graph [6].

THEOREM 7.5. *Let $k = O(1)$ and $H \in \mathcal{H}_k(l)$. If H is decomposable into a subgraph having a matching of size q and a subgraph forming an independent set on $k - l$ vertices and possibly some edges between these two subgraphs, then one can decide whether $N(H, G) = 0$ in time $O(m^q n^{l-2q} + \min\{m^q n^{l-2q+1},\ n^{\omega(\lceil l/2\rceil,1,\lfloor l/2\rfloor)}\})$. Furthermore, if $k - l = 2$, then one can also compute $N(H, G)$ in time $O(m^q n^{k-2q-2} + \min\{m^q n^{k-2q-1},\ n^{\omega(\lceil (k-2)/2\rceil,1,\lfloor (k-2)/2\rfloor)}\})$.*

Note that for any graph the complement to the set of vertices covered by a maximal matching is an independent set. Hence, we obtain the following corollary from Theorem 7.5.

COROLLARY 7.6. *Let $k = O(1)$ and $H \in \mathcal{H}_k$. If H has a maximal matching of size q, then one can decide whether $N(H, G) = 0$ in time $O(m^q + \min\{m^q n,\ n^{\omega(q,1,q)}\})$. Furthermore, if $k - 2q = 2$, then the numbers $N(H, G)$ can be computed in time $O(m^{(k-2)/2} + \min\{m^{(k-2)/2}n,\ n^{\omega((k-2)/2,1,(k-2)/2)}\})$.*

8. 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, the counting versions of these problems seem to be hard for all such H, independently of their density and the size of s. (See Theorem 5.2, and for its special four-vertex cases see also [16].) On the contrary, in the subgraph isomorphism case, it seems that the larger s, the better the upper bounds we can obtain (recall our two main results and [25]).

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

Incidentally, our $O(n^\omega)$-bound for $H \in \mathcal{H}_4 \setminus \{K_4\}$ 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 $H \in \mathcal{H}_5 \setminus \{K_5\}$ coincides with the best known running time for detecting or counting copies of K_4.

Of course, the ultimate goal is to improve the time upper bounds for complete graphs, and 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 not necessarily complete small pattern graphs occurs. Very recent examples of applications include identification of computational patterns in automatic design of processor systems [27], motif counting and discovery in biomolecular networks [1], and structure discovery in protein networks [4].

Acknowledgments. The authors are very grateful to unknown referees whose comments helped to improve a preliminary version of the paper. Special thanks go to the referee who simplified the time upper bounds in our main results and provided many other valuable comments.

REFERENCES

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

[2] N. Alon, R. Yuster, and U. Zwick, *Color-coding*, J. ACM, 42 (1995), pp. 844–856.

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

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

[5] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, *Counting Paths and Packings in Halves*, in Lecture Notes in Comput. Sci. 5757 (A. Fiat and P. Sanders, eds.), Springer, New York, 2009, pp. 578–586.

[6] A. Brandstädt, V. B. Le, and J. Spinrad, *Graph Classes: A Survey,* SIAM Monogr. Discrete Math. Appl., SIAM, Philadelphia, 1999.

[7] N. Chiba and T. Nishizeki, *Arboricity and subgraph listing algorithms*, SIAM J. Comput., 14 (1985), pp. 210–223.

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

[9] D. Coppersmith and S. Winograd, *Matrix multiplication via arithmetic progressions*, J. Symbolic Comput., 9 (1990), pp. 251–280.

[10] F. Eisenbrand and F. Grandoni, *On the complexity of fixed parameter clique and dominating set*, Theoret. Comput. Sci., 326 (2004), pp. 57–67.

[11] F. V. Fomin, D. Lokshtanov, V. Raman, B. V. R. Rao, and S. Saurabh, *Faster Algorithms for Finding and Counting Subgraphs*, J. Comput. Systems Sci., 78 (2012), pp. 698–706.

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

[13] A. W. Goodman, *On sets of acquaintances and strangers at any party*, Amer. Math. Monthly, 66 (1959), pp. 778–783.

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

[15] A. Itai and M. Rodeh, *Finding a minimum circuit in a graph*, SIAM J. Comput., 7 (1978), pp. 413–423.

[16] T. Kloks, D. Kratsch, and H. Müller, *Finding and counting small induced subgraphs efficiently*, Inform. Process. Lett., 74 (2000), pp. 115–121.

[17] W. L. Kocay, *Some new methods in reconstruction theory*, in Combinatorial Mathematics IX, Lecture Notes in Math. 952, Springer, New York, 1982, pp. 89–114.

[18] F. Le Gall, *Faster algorithms for rectangular matrix multiplication*, in Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, 2012, pp. 514–523.

[19] J. Nešetřil and S. Poljak, *On the complexity of the subgraph problem*, Comment. Math. Univ. Carolin., 26 (1985), pp. 415–419.

[20] J. Plehn and B. Voigt, *Finding Minimally Weighted Subgraphs*, Lecture Notes in Comput. Sci. 484, Springer, Berlin, 1991, pp. 18–29.

[21] A. Stothers, *On the Complexity of Matrix Multiplication*, Ph.D. thesis, University of Edinburgh, 2010.

[22] V. Vassilevska, *Efficient algorithms for clique problems*, Inform. Process. Lett., 109 (2009), pp. 254–257.

[23] V. Vassilevska, *Efficient Algorithms for Path Problems in Weighted Graphs*, Ph.D. thesis, Carnegie Mellon University, 2008.

[24] V. V. Williams, *Multiplying matrices faster than Coppersmith-Winograd*, in Proceedings of the 44th ACM Symposium on Theory of Computing Conference, 2012, pp. 887–898.

[25] V. Vassilevska and R. Williams, *Finding, minimizing, and counting weighted subgraphs*, in Proceedings of the 41st Annual ACM Symposium on Theory of Computing, 2009, pp. 455–464.

[26] R. Williams, *Finding paths of length k in* $O^*(2^k)$ *time*, Inform. Process. Lett., 109 (2009), pp. 315–318.

[27] C. Wolinski, K. Kuchcinski, and E. Raffin, *Automatic design of application-specific reconfigurable processor extensions with UPaK synthesis kernel*, ACM Trans. Design Automat. Electron. Syst., 15 (2009), pp. 1–36.

[28] R. Yuster and U. Zwick, *Finding even cycles even faster*, SIAM J. Discrete Math., 10 (1997), pp. 209–222.