The complexity of problems concerning matrix powers and the numbers of paths in a graph

Daniel Meister

Report No. 369 Oktober 2005

Preprint-Reihe
Fakultät für Mathematik und Informatik
Universität Würzburg

The complexity of problems concerning matrix powers and the numbers of paths in a graph

Daniel Meister
Theoretische Informatik
Bayerische Julius-Maximilians-Universitaet Wuerzburg
97074 Wuerzburg, Germany
meister@informatik.uni-wuerzburg.de

Abstract. Let k, a and ν be natural numbers. Let M be a square matrix over the natural numbers. How difficult is it to decide whether a specified entry of the k-th power of M is at least a? Let G be a graph. How difficult is it to decide whether there are at least ν paths of length k between specified vertices of G? We consider a path as a sequence of vertices where consecutive vertices are connected by an arc or an edge. We study these and further—related—problems. We investigate in which manner the complexities of our problems depend on the number representation system, whether the given graph is directed or undirected. The notion of *isomorphic paths* is introduced as an alternative to transfer the notion of equal paths in a directed graph to undirected graphs. Our main results show NL-, $\mathrm{C_{=}L}$- and PL-completeness.

1. Introduction. One of the author's favorite small examples of combinatorial propositions with a nice small proof was taught in his second semester at university: given $r, k \geq 1$, there are exactly $\binom{r+k-1}{k-1}$ tuples $(a_1, \ldots, a_k)$ of natural numbers such that $a_1 + \cdots + a_k = r$. We extend the sum condition to arbitrary linear combinations and ask the question, given natural numbers $r, k, c_1, \ldots, c_k \geq 1$, how many tuples $(a_1, \ldots, a_k)$ of natural numbers satisfy $c_1 a_1 + \cdots + c_k a_k = r$? This question already becomes difficult for $k = 2$, but can still be solved with moderate effort. We will extend this problem to a set of linear equations and ask for a lower bound of the accumulated number of solutions.

The problem that has been sketched above does not seem to be connected to any of the title's problems. The actual problems that we will consider are of a completely different kind. At first glance! Let G be a graph. A path in G is a sequence of vertices where every pair of consecutive vertices is connected by an arc or an edge, depending on whether the graph is directed or undirected. We want to decide whether the number of paths of given length between two given vertices is at least a given number. For this problem, we consider directed as well as undirected graphs. For both kinds of graphs, we say that two paths are equal if they are of equal length and coincide in every position, i.e., they are equal if and only if the sequences are equal. However, it is an interesting question whether this notion of equality really captures the notion of equal paths in undirected paths. To give an alternative we introduce the notion of *isomorphic* paths in undirected graphs. This isomorphism notion is motivated by the observation that a cycle in an undirected graph can be traversed in two directions, and none of these directions can be preferred. We will give a characterisation of isomorphic paths in terms of $\mathcal{Q}$-trees, that are based on the concept of $\mathcal{PQ}$-trees for interval graphs [4].

Number-of-paths problems are of great interest in connection with logarithmic-space computations. One of the basic problems for graphs is the graph accessibility problem GAP. The input is a (labelled) directed graph, and the question is whether there is at least one path from a given vertex to another; the length is not specified. This problem is complete for nondeterministic logarithmic space, proved implicitly by Savitch [15]. In case of undirected graphs, a new complexity class was introduced to capture the undirectedness. Lewis and Papadimitriou speak of *symmetric* computation, and the class which the undirected version of GAP—the connectivity problem—is complete for is called SL (by its modern name) [11]. The class SL lies between L and NL, and Armoni, Ta-Shma, Wigderson, Zhou gave a deterministic $\mathcal{O}(\log^{\frac{4}{3}} n)$ space upper bound for SL [3]. Recently, Reingold showed that SL is contained in L by giving an algorithm deciding the connectivity problem that uses only logarithmic space [13]. Hence, SL = L by this result. Allender, Reinhardt, Zhou showed that it can be verified in nondeterministic logarithmic space whether an acyclic (directed) graph has ν u, v-paths, if number ν is given in unary representation [2]; this problem is certainly a generalisation of GAP.

For all these problems just mentioned the lengths of paths do not matter, and if there is a desired path, then there is a path containing every vertex at most once. Our problems, that consider the length of a path as additional input parameter, become considerably difficult when the length is greater than the number of vertices of the input graph. Of course, vertices may have multiple occurrences. To determine the influence of input parameter representations, we study the complexities of our problems with respect to the number system that is used to encode length and number of paths. We will see that the cases that have to be considered reduce to unary and binary number representation.

The second type of problems that we are interested in is a matrix problem. For a square matrix we want to decide whether a specified entry in a specified power of the matrix has a value that is at least a given number. Input numbers may be given in k-ary form for any natural $k \geq 1$. Again, we want to determine the complexities of these problems with respect to the encoding number system. It turns out that these problems are closely related to number-of-paths problems. The connection is based on the fact that powers of adjacency matrices of graphs contain the numbers of paths of fixed length between two vertices. It is known that computing a power of a matrix where the power is at most the dimension of the matrix is as hard as computing the determinant of a matrix [7]. Both these problems are complete (with respect to NC^1-reducibility) for Cook's function class DET, and DET is contained in NC^2 [7]. We want to consider arbitrary powers of matrices, especially powers that are much larger than the dimension of the matrix.

This report is organised as follows. In Section 3, we define our problems and introduce formally the notion of isomorphic paths in undirected graphs. A characterisation of isomorphic paths using $\mathcal{Q}$-trees is given there. In Section 4, we will establish first relations between the problems defined in Section 3. A main result in this section shows the isomorphism between a number-of-paths problems for directed graphs and our power-of-matrix problem. In the following sections, we will focus on number-of-paths problems. Precisely, in Section 5, we consider number-of-paths problems for *marked* graphs. Marked graphs contain vertices that are marked, and instead of asking for the number of paths of given length, we ask for the number of paths containing a given number of marked vertices. We will show that, in case of directed graphs, these problems reduce to corresponding problems for unmarked graphs. In case of undirected graphs, a similar result cannot be shown but we will observe a useful structural property. Sections 6 and 7 are devoted to solving number-of-paths problems. In Section 6, problems where the number of paths is represented in unary form are investigated. We will show that these problems are NL-complete. In Section 7, problems where the number of paths is represented in binary form are studied, and we will show PL- and $C_=L$-completeness results. However, the hardest number-of-paths problem for directed graphs remains unsolved in the sense that we cannot give tight upper or lower bounds. In Section 8, we will introduce another problem that is closely related to that hardest problem, and we expect that it can help understanding and solving the original one.

2. Preliminaries. We fix an alphabet Σ. The empty word is denoted by ε. For set $A \subseteq \Sigma^*$, by c_A, we denote the *characteristic function* of A, i.e., the total function with co-domain $\{0,1\}$, and, for every $x \in \Sigma^*$, $c_A(x) = 1 \leftrightarrow x \in A$. Let $\leq$ be a reducibility, and let $\mathcal{C}$ be a complexity class. Set A is *$\mathcal{C}$-hard* with respect to $\leq$-reducibility, if $X \leq A$ for every $X \in \mathcal{C}$. A is *$\mathcal{C}$-complete* with respect to $\leq$-reducibility, if A is $\mathcal{C}$-hard with respect to $\leq$ and $A \in \mathcal{C}$. Since it will always be clear with respect to which reducibility hardness and completeness is meant, we will also speak of $\mathcal{C}$-hardness and $\mathcal{C}$-completeness.

Logarithmic-space computations. A deterministic logarithmic-space Turing machine is a Turing machine with input and working tape that, on input x, uses at most $\lfloor c \cdot \log |x| \rfloor$ cells on the working tape for a constant $c \geq 1$ independent of x. A nondeterministic logarithmic-space Turing machine is a nondeterministic Turing machine that accepts input x if and only if there is an accepting computation path that uses at most $\lfloor c \cdot \log |x| \rfloor$ cells on the working tape for a constant $c \geq 1$ independent of x. By L and NL we denote the sets of languages that can be accepted in deterministic and nondeterministic logarithmic space, respectively. For some oracle set A, $L \in \mathrm{L}^A$ if and only if there are a constant $c \geq 1$ and an oracle Turing-machine that uses, on input x, at most $\lfloor c \cdot \log |x| \rfloor$ cells on the working tape. In particular, the number of cells used on the oracle tape does not count. However, $L \in \mathrm{NL}^A$ if and only if there is a nondeterministic oracle Turing-machine accepting as described above but generating oracle queries deterministically, i.e., while generating a query the machine is not allowed to make nondeterministic steps. This model is known as the so-called Ruzzo-Simon-Tompa relativization [14]. The function classes FL and FL^A are defined straightforwardly. The complexity class coNL contains the complements of NL-languages. In Section 7, we will encounter further logarithmic-space complexity classes. As usual, for complexity classes $\mathcal{C}, \mathcal{D}$, $\mathcal{C}^{\mathcal{D}}$ is the union of all sets $\mathcal{C}^A$ for $A \in \mathcal{D}$. The following inclusions are non-trivial results for logarithmic-space computations.

Theorem 1. [10], [16] $\mathrm{L} \subseteq \mathrm{NL} = \mathrm{coNL} = \mathrm{L}^{\mathrm{NL}} = \mathrm{NL}^{\mathrm{NL}}$.

For two sets A and B, we say that A *logarithmic-space many-one reduces* to B, $A \leq_m^{\mathrm{L}} B$, if there is a function $f \in \mathrm{FL}$ such that, for every $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) \in B$. Mostly, we say that A *reduces* to B for short and mean logarithmic-space many-one reducibility. Only in rare cases, *to reduce* signifies another reducibility, that is always precised afterwards, however.

Arithmetics. The set of natural numbers is denoted by $\mathbb{N}$; 0 is a natural number. The interval of natural numbers bounded by a and b where $a \leq b$ is denoted by $[a,b]$. Given $r \geq 1$ natural numbers $m_1, \ldots, m_r$ the *iterated addition problem* ItAdd is the problem to compute $\sum_{i=1}^{r} m_i$. The *iterated multiplication problem* ItMult is the corresponding problem for computing $\prod_{i=1}^{r} m_i$. The *division problem* Div has two numbers a and b as inputs where $b > 0$ and $\lfloor \frac{a}{b} \rfloor$ is to be computed. Input numbers and results are represented in binary form.

Theorem 2. [17], [6], [9] ItAdd, ItMult *and* Div *are in* FL.

Let $k \geq 1$. In the *k-ary number system*, natural numbers are represented by words over the digits $\{0, \ldots, k-1\}$, and a k-ary representation $m = m_r \cdots m_1$ defines number $\sum_{i=1}^{r} m_i \cdot k^{i-1}$ or r, if $k \geq 2$ or $k = 1$, respectively. Applying the obvious rules, sum and difference of two numbers in k-ary representation can be computed in logarithmic space. The function conv_k for $k \geq 3$ converts k-ary representation of natural numbers into binary representation. The inverse translation is performed by $\overline{\mathrm{conv}}_k$.

Lemma 3. *For every* $k \geq 3$*,* conv_k *and* $\overline{\mathrm{conv}}_k$ *are in* FL *.*

Proof: Let $k \geq 3$. Let $m = \sum_{i=1}^{r} a_i \cdot k^{i-1}$ for $r \geq 1$, $a_i \in [0, k-1]$, be a natural number in k-ary representation. Since k is fixed, binary representations of $0, \ldots, k$ are known, and by Theorem 2, the binary representation of each $a_i k^i$ can be written on the output tape in logarithmic space using iterated multiplication. Note that i is at most the number of digits of the k-ary representation of m. Since iterated addition can be performed in logarithmic space by Theorem 2, the binary representation of m is computed in logarithmic space.

For the inverse function, let $m' = \sum_{i=1}^{r'} b_i \cdot 2^{i-1}$, $r' \geq 1$, $b_i \in \{0, 1\}$, be a natural number in binary representation. Note that $m' < k^{r'}$. First, the products k^i for $i \in [1, r]$ can be written on the output tape using iterated multiplication in logarithmic space. Second, the quotients $\lfloor \frac{m'}{k^i} \rfloor$ for $i \in [1, r]$ can be written on the output tape in logarithmic space. Third, for each quotient the residue class modulo k can be determined, each of which gives a digit in the k-ary representation of m'. Since this algorithm uses only logarithmic space due to Theorem 2, we conclude the lemma. ∎

Graphs. We consider only simple and finite graphs. An undirected graph $G = (V, E)$ is a pair composed of the vertex set V and the edge set $E \subseteq V^{(2)}$, where $V^{(2)}$ is the set of two-elementary subsets of V. A directed graph $G = (V, A)$ is a pair composed of the vertex set V and the arc set $A \subseteq V^2$. We always assume that $V = [1, n]$ where $n = |V|$. We further assume that graphs are given as adjacency lists or lists of arcs. So, adjacency of two vertices and the existence of arcs can be tested in logarithmic space. For a directed graph $G = (V, A)$, a *u, v-path* of G is a sequence $(x_0, \ldots, x_r)$ for $r \geq 0$ such that $x_0 = u$, $x_r = v$ and $(x_{i-1}, x_i) \in A$ for every $i \in [1, r]$. Similarly, for $G' = (V, E)$ an undirected graph, a u, v-path of G' is a sequence $(z_0, \ldots, z_s)$ where $z_0 = u$, $z_s = v$ and $\{z_{i-1}, z_i\} \in E$ for every $i \in [1, s]$. Note that two occurrences of the same vertex in a path in an undirected graph are not consecutive. The *length* of a path is the number of its vertices minus 1. Differently to the case of directed graphs, we may consider u, v-paths also to be v, u-paths since in undirected graphs paths do not begin and end at some vertices but are *between* two vertices. This notion will lead to the definition of *isomorphic* paths in an undirected graph in the next section. It may be that isomorphism of paths in undirected graphs captures the notion of equality better.

By GAP and GIP we denote the *graph accessibility problem* and the *graph inaccessibility problem*, respectively. GAP is the set of all triples (G, u, v) where G is a simple finite directed graph, u and v are vertices of G, and there is a u, v-path in G. GIP is similarly defined with the modification that there is **no** u, v-path in G. Both problems are NL- and coNL-complete due to Savitch [15] and Immerman and Szelepcsényi [10], [16].

3. Problem definitions. Our main interest in this paper is the study of the complexity of the problem to determine whether a given graph has a given number of paths between two vertices. We will see in which way the complexity depends on the representations of the input numbers and whether the graph is directed or undirected. We will also identify a relationship to similar problems for powers of matrices. First, we introduce our graph problems and show easy relationships between these problems.

Let $G = (V, A)$ be a directed graph, and let $(x_0, \ldots, x_r)$ and $(x'_0, \ldots, x'_{r'})$ be two paths of G. We say that these paths are equal if and only if $r = r'$ and $x_i = x'_i$ for every $i \in [0, r] \cap [0, r']$. Hence, it is easy to determine whether two given paths are equal.

Problem definitions.

NDP(α, β) for $\alpha, \beta \geq 1$ (number of paths in a directed graph).
Instance. (G, k, ν, u, v) where $G = (V, A)$ is a simple finite directed graph, $u, v \in V$ and $k, \nu \in \mathbb{N}$, and k is represented in α-ary form and ν is represented in β-ary form.
Question. Are there ν u, v-paths of length k in G?

NMDP(α, β) for $\alpha, \beta \geq 1$ (number of paths in a marked directed graph).
Instance. (G, M, k, ν, u, v) where $G = (V, A)$ is a simple finite directed graph, $M \subseteq V$, $u, v \in V$ and $k, \nu \in \mathbb{N}$, and k is represented in α-ary form and ν is represented in β-ary form.
Question. Let $r \geq 0$. If a path of G contains exactly r marked vertices, i.e., vertices from M, we say that it is *r-marked*. Are there ν k-marked u, v-paths in G?

Instead of a lower bound on the number of paths we also ask for the exact number of paths. These problems are denoted by adding a prefix 'x', and we obtain problems like xNDP(1, 1) or xNMDP(2, 1). Furthermore, we may require that start and end vertex of the paths are different or they are the same. These problems are denoted by additional prefixes 's' and 's̄', e.g., sNDP(1, 2), sxNDP(2, 2) or s̄NDP(2, 2). Note that, if start and end vertex are equal, we ask for the number of cycles containg the specified start vertex.

For undirected graph, the definition of *equal paths* is similar to the one for directed graphs. Let $G = (V, E)$ be an undirected graph, and let $P_1 = (z_0, \ldots, z_s)$ and $P_2 = (z'_0, \ldots, z'_{s'})$ be two paths in G. Then, P_1 and P_2 are equal if and only if $s = s'$ and $z_i = z'_i$ for every $i \in [0, s] \cap [0, s']$. We define problems analogue to the ones for directed graphs but write 'U' instead of 'D'. The problem classes then are denoted by NUP, NMUP, sNUP, sNMUP, s̄NUP, s̄NMUP, xNUP, xNMUP, sxNUP, sxNMUP, s̄xNUP, s̄xNMUP.

The case of undirected graphs, however, is more complicated than it seems at first glance. Let $C = (u_1, \ldots, u_q)$ be a cycle in G, i.e., C is a path in G where first and last vertex are equal. Since the edges of G have no orientation, C is not oriented either. The question arises whether $C = (u_1, \ldots, u_q)$ and $C' =_{\text{def}} (u_q, \ldots, u_1)$ should be considered equal or not. By our definition above, equality of undirected paths is exactly defined. Therefore, we will say that C and C' are *isomorphic*. Starting from this idea, we shall precise what isomorphic paths should be. Let $P_1 = (z_0, \ldots, z_s)$ and $P_2 = (z'_0, \ldots, z'_{s'})$ be paths of G. Let $\mathfrak{I} =_{\text{def}} [0, s] \cap [0, s']$. If there are no $k, k' \in [0, s]$, $k + 2 < k' < s + k$, such that $z_k = z_{k'}$, then P_1 and P_2 are isomorphic if and only if $s = s'$ and (a) $z_i = z'_i$ for all $i \in \mathfrak{I}$ or (b) $z_i = z'_{s'-i}$ for all $i \in \mathfrak{I}$. Let $\mathfrak{u}$ be a vertex that is not contained in G. Let G' emerge from G by adding $\mathfrak{u}$ to G and making it adjacent with every vertex of G. Let $k, k' \in [0, s]$. We define $P_1^{k \rightharpoonup k'}$ as the path $(z_0, \ldots, z_k, \mathfrak{u}, z_{k'}, \ldots, z_s)$; an analogue definition for P_2. If there are $k, k' \in [0, s]$, $k + 2 < k' < s + k$, such that $z_k = z_{k'}$, then P_1 and P_2 are isomorphic if and only if $s = s'$ and there are $k, k' \in [0, s]$, $k + 2 < k' < s + k$, such that $z_k = z_{k'}$ and (a) $P_1^{k \rightharpoonup k'}$ and $P_2^{k \rightharpoonup k'}$ are isomorphic paths in G' and $(z_k, \ldots, z_{k'})$ and

$(z'_k, \ldots, z'_{k'})$ are isomorphic paths in G or (b) $P_1^{k \rightharpoonup k'}$ and $P_2^{s'-k' \rightharpoonup s'-k}$ are isomorphic paths in G' and $(z_k, \ldots, z_{k'})$ and $(z'_{s'-k'}, \ldots, z'_{s'-k})$ are isomorphic paths in G.

The definition of isomorphic paths looks rather involved. For a better understanding, we will discuss a characterisation of isomorphic paths that is based on a special tree representation. This representation is related to $\mathcal{PQ}$-trees of interval graphs. $\mathcal{PQ}$-trees were introduced by Booth and Lueker to recognise interval graphs in linear time [4]. We will use $\mathcal{Q}$-trees, that are $\mathcal{PQ}$-trees without $\mathcal{P}$-vertices. A *$\mathcal{Q}$-tree* is a rooted and ordered tree whose vertices are labelled. By *ordered tree* we mean that child vertices of a vertex are ordered, i.e., there is a first child vertex, a second, a third. A $\mathcal{Q}$-tree representation of a path in G is obtained inductively. Let $P = (z_0, \ldots, z_s)$ be a path in G. A *cycle* in P is a subpath $(z_i, \ldots, z_j)$, $0 \leq i < j \leq s$ of P such that $z_i = z_j$. A cycle in P is a *proper subcycle* of P if the cycle does not contain both the first and last vertex of P. A proper subcycle of P is *maximal* in P if there are no i', j' such that $0 \leq i' \leq i < j \leq j' \leq s$ and $j - i < j' - i' < s$ and $z_{i'} = z_{j'}$. Two cycles $(z_k, \ldots, z_{k'})$ and $(z_\ell, \ldots, z_{\ell'})$ in P, $k < k'$ and $\ell < \ell'$, are *disjoint* if $k' \leq \ell$ or $\ell' \leq k$. Now, let $\mathcal{C}$ be a maximal set of pairwise disjoint maximal proper subcycles of P. Construct a $\mathcal{Q}$-tree representation for every cycle in $\mathcal{C}$. Let P' be obtained from P by deleting the vertices of each cycle from $\mathcal{C}$. Observe that P' may not be a path in G. Let P' have n' vertices. A $\mathcal{Q}$-tree representation of P' has $n' + 1$ vertices where n' vertices are the child vertices of the root vertex $\mathfrak{r}$. The root vertex has no label, and the labels of the leaf vertices are the vertices of P' in the canonical order. We obtain a $\mathcal{Q}$-tree representation of P by making the root vertex of the $\mathcal{Q}$-tree representation of every cycle in $\mathcal{C}$ a child vertex of $\mathfrak{r}$ that is situated where the vertices of the cycle have been deleted. We see that P is the leaf word of its $\mathcal{Q}$-tree where consecutive, multiple occurrences of the same vertex are ignored. Note that a path may have more than one $\mathcal{Q}$-tree representation, and they depend on the choices of the proper subcycles. Two $\mathcal{Q}$-trees are *isomorphic* if and only if the first $\mathcal{Q}$-tree can be transformed into the second by reversing the order of child vertices in subtrees, i.e., performing a $\mathcal{Q}$-operation. *$\mathcal{Q}$-vertices* in a $\mathcal{Q}$-tree are those vertices that have no label, i.e., the root vertex of the $\mathcal{Q}$-tree and all other vertices that are root vertices of a subtree with more than two vertices.

Lemma 4. *Let $G = (V, E)$ be an undirected graph, and let P_1 and P_2 be two paths in G. P_1 and P_2 are isomorphic if and only if they have isomorphic $\mathcal{Q}$-tree representations.*

Proof: Let $P_1 = (z_0, \ldots, z_s)$ and $P_2 = (z'_0, \ldots, z'_{s'})$. By definition of isomorphic paths as well as isomorphic $\mathcal{Q}$-tree representations, we can assume that s equals s'. We will show the claim by induction over s. We distinguish two cases: P_1 and P_2 have proper subcycles or not. First, let P_1 and P_2 have no proper subcycles. By definition, P_1 and P_2 are isomorphic if and only if $s = s'$ and $z_i = z'_i$ for all $i \in [0, s] \cap [0, s']$ or $z_i = z'_{s'-i}$ for all $i \in [0, s] \cap [0, s']$. But this is the case if and only if P_1 and P_2 have isomorphic $\mathcal{Q}$-tree representations. Note that the $\mathcal{Q}$-tree representations of P_1 and P_2 are unique and each have a root and every other vertex is child of the root.

Let P_1 and P_2 have proper subcycles. Let $C = (z_q, \ldots, z_{q'})$ be a proper subcycle of P_1 where $q < q' - 2$. If such a cycle does not exist, then for every proper subcycle C' of P_1 there is $q < s - 1$ such that $C' = (z_q, z_{q+1}, z_{q+2})$, and the claim holds. As we have defined for isomorphic paths, let $\mathfrak{u}$ be a vertex that is not contained in G, and let G' emerge from G by adding $\mathfrak{u}$ and making it adjacent with every vertex of G. Assume that P_1 and P_2 are isomorphic. This is the case if and only if there are $k, k' \in [0, s]$, $k + 2 < k' < s + k$, such that (here, we assume the first case; if the second holds, the proof is similar) $P_1^{k \rightharpoonup k'}$ and

$P_2^{k \rightharpoonup k'}$ are isomorphic in G' and $(z_k, \ldots, z_{k'})$ and $(z'_k, \ldots, z'_{k'})$ are isomorphic in G. Since the four paths have lengths less than s, by induction hypothesis, there are appropriate $\mathcal{Q}$-tree representations proving isomorphism. Observe that the $\mathcal{Q}$-tree for $P_1^{k \rightharpoonup k'}$ contains a $\mathcal{Q}$-vertex $\mathfrak{r}$ that has exactly three child vertices whose labels are the vertices of the cycle $(z_k, \mathfrak{u}, z_{k'})$. This is due to the fact that for constructing $\mathcal{Q}$-tree representations only *maximal* proper subcycles are chosen. Replacing the subtree rooted at $\mathfrak{r}$ by the $\mathcal{Q}$-tree representation for $(z_k, \ldots, z_{k'})$ we obtain a $\mathcal{Q}$-tree representation for P_1. Analogously, we obtain a $\mathcal{Q}$-tree representation for P_2 from the $\mathcal{Q}$-tree representations for $P_2^{k \rightharpoonup k'}$ and $(z'_k, \ldots, z'_{k'})$, and the $\mathcal{Q}$-tree representations for P_1 and P_2 are isomorphic. ■

As for directed and undirected paths, we define number-of-paths problems for isomorphic paths, that we denote as NIP, NMIP and so on, also using the introduced prefixes. An additional and immediate problem arises: What is the complexity of deciding whether two paths are isomorphic?

The second type of problems concerns entries in matrices. More precisely, we want to decide the value of a specified entry in a power of a given matrix.

Problem definition.

POM(α, β) for $\alpha, \beta \geq 1$ (element of a power of a matrix).
Instance. (M, k, a, i, j) where M is an $n \times n$ matrix for some $n \geq 1$ over $\mathbb{N}$, $k, a \geq 0$, $i, j \in [1, n]$. All entries of M and i and j are represented in binary form, and k is represented in α-ary form and a is represented in β-ary form.
Question. $(M^k)_{ij} \geq a$?

We define xPOM(α, β) to be the variant of POM(α, β) where we ask for the exact value of the specified entry.

4. First relations. In this section, we want to establish fundamental relations between the problems defined in the last section. We will see that the power-of-matrix problems are only reformulations of number-of-paths problems for directed paths. We begin by showing that for containment results we only have to consider number systems with bases 1 and 2. Note that the prefixes s and $\bar{\text{s}}$ are not applicable to the matrix problems; the statement of the following lemma should be understood accordingly.

Lemma 5. *For all $\alpha, \beta \geq 1$ and all $\mathcal{P} \in$ {NDP, NMDP, NUP, NMUP, NIP, NMIP, POM} and $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}, \mathrm{x}, \mathrm{sx}, \bar{\mathrm{s}}\mathrm{x}\}$,*

(1) $\delta\mathcal{P}(1, \beta) \leq_m^{\mathrm{L}} \delta\mathcal{P}(\alpha, \beta)$,

(2) $\delta\mathcal{P}(\alpha, 1) \leq_m^{\mathrm{L}} \delta\mathcal{P}(\alpha, \beta)$.

(3) $\delta\mathcal{P}(\alpha + 1, \beta) \equiv_m^{\mathrm{L}} \delta\mathcal{P}(2, \beta)$,

(4) $\delta\mathcal{P}(\alpha, \beta + 1) \equiv_m^{\mathrm{L}} \delta\mathcal{P}(\alpha, 2)$.

Proof: It obviously suffices to transform numbers from one representation system into another. For the cases (1–2), this is easily done by incrementation, which needs only logarithmic space for every k-ary representation system. Note that, if $k = 1$, the numbers only have to be copied. For cases (3–4), if $\alpha \geq 2$ or $\beta \geq 2$, we establish the reductions between both problems by applying functions conv and $\overline{\text{conv}}$. The lemma follows by Lemma 3. ■

It follows from Lemma 5 that it suffices to consider only unary or binary number

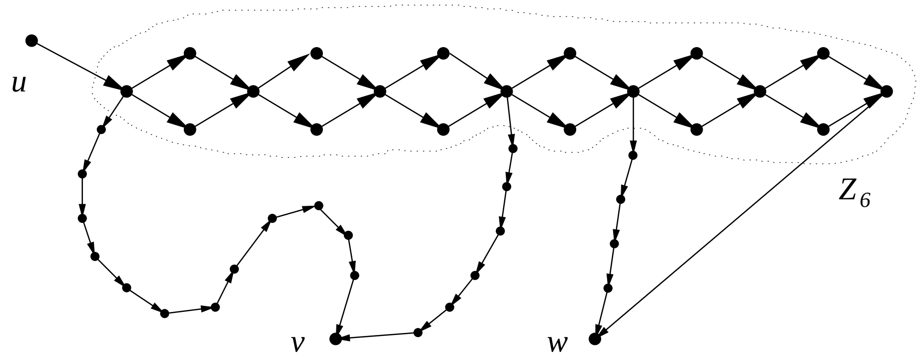

Figure 1. A graph with nine u, v-paths and 80 u, w-paths.

representations.

Now, we relate number-of-paths problems for directed graphs to power-of-matrix problems. We need a well-known result from graph theory. Let $G = (V, A)$ be a directed graph, and let $n =_{\mathrm{def}} |V|$. By $\mathcal{A}dj(G)$ we denote the $n \times n$ matrix M over $\{0, 1\}$ for which $M_{ij} = 1$ if and only if $(i, j) \in A$, $i, j \in [1, n]$.

Lemma 6. *Let $G = (V, A)$ be a directed graph. Let $n =_{\mathrm{def}} |V|$, and let $k \geq 1$, $i, j \in [1, n]$. Let $M =_{\mathrm{def}} \mathcal{A}dj(G)$. Then, $(M^k)_{ij}$ is the number of i, j-paths of length k in G.*

Lemma 7. *Let $\alpha, \beta \in \{1, 2\}$ and $\delta \in \{\varepsilon, \mathrm{X}\}$. Then, $\delta\mathrm{NDP}(\alpha, \beta) \equiv_m^{\mathrm{L}} \delta\mathrm{POM}(\alpha, \beta)$.*

Proof: Note that $\delta\mathrm{NDP}(\alpha, \beta)$ reduces to $\delta\mathrm{POM}(\alpha, \beta)$ by computing matrix $\mathcal{A}dj(G)$ due to Lemma 6, which is possible in logarithmic space. For $\delta\mathrm{POM}(\alpha, \beta) \leq_m^{\mathrm{L}} \delta\mathrm{NDP}(\alpha, \beta)$, let (M, k, a, i, j) be an instance of $\mathrm{POM}(\alpha, \beta)$, and let ω be the size of this instance. Consider the acyclic graph depicted in Figure 1. All u, v- and u, w-paths have the same length and there are exactly $1 + 8 = 9$ u, v-paths and $16 + 64 = 80$ u, w-paths. Graphs of the form of Z_6 as it is denoted in Figure 1 play the central part in our reduction. By Z_0 we mean a single vertex, and Z_s is extended to Z_{s+1} by identifying the sink of Z_s with the source vertex 1 of the graph $(\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (2, 4), (3, 4)\})$. Let $n =_{\mathrm{def}} \dim M$, and let r denote the length of a largest entry in M, i.e., the number of bits representing a largest entry of M. We construct graph G as follows. For every $x \in [1, n]$, let G have a vertex with name x and a copy of the graph Z_{r-1}. Add an arc from x to the source vertex of its Z_{r-1} copy. For every entry M_{ij} of M add, as it is shown in Figure 1, paths to G such that the number of i, j-paths of length $2r$ is exactly the number M_{ij}. Note that G can have at most $n \cdot 3r + n^2 \cdot 2r$ vertices, and every vertex has a name of length logarithmic in ω. It is clear that G can be computed in logarithmic space and that further vertices that are not adjacent to any other vertex can be added to G such that the final graph contains exactly $3 + 3 \log \omega \geq 5rn^2$ vertices (in order to make the names of the vertices a complete interval of $\mathbb{N}$); let G' be the final graph. Then, the number of i, j-paths in G' of length $2kr$ is equal to $(M^k)_{ij}$. Due to Theorem 2, the product $2kr$ can be computed in logarithmic space. Then, $(M, k, a, i, j) \in \delta\mathrm{POM}(\alpha, \beta)$ if and only if $(G', 2kr, a, i, j) \in \delta\mathrm{NDP}(\alpha, \beta)$. Note that our argumentations are for the case $\alpha = 2$. In case of $\alpha = 1$, $2kr$ can be computed straightforwardly. ■

Lemma 7 justifies that—with a single exception—we will consider only graph problems in the rest of this work.

Lemma 8. *Let $\alpha, \beta \geq 1$, and let $\delta \in \{\varepsilon, \mathrm{S}, \bar{\mathrm{S}}, \mathrm{X}, \mathrm{SX}, \bar{\mathrm{S}}\mathrm{X}\}$.*

(1) $\delta\mathrm{NDP}(\alpha, \beta) \leq_m^{\mathrm{L}} \delta\mathrm{NMDP}(\alpha, \beta)$,

(2) $\delta\mathrm{NUP}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathrm{NMUP}(\alpha,\beta)$ *,*

(3) $\delta\mathrm{NIP}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathrm{NMIP}(\alpha,\beta)$ *.*

Proof: We only prove statement (1) for $\delta = \varepsilon$. All other cases follow with the same argumentations, since our reduction does not manipulate the numbers of paths. Let (G,k,ν,u,v) be an instance of $\mathrm{NDP}(\alpha,\beta)$ where $G = (V,A)$. Then, $(G,k,\nu,u,v) \in \mathrm{NDP}(\alpha,\beta)$ if and only if $(G,V,k+1,\nu,u,v) \in \mathrm{NMDP}(\alpha,\beta)$. Remember that the length of a path is one less the number of its vertices. ■

Lemma 9. *Let* $\alpha,\beta \in \{1,2\}$*, and let* $\delta \in \{\varepsilon, \mathrm{X}\}$ *.*

(1) $\mathrm{S}\delta\mathcal{P}(\alpha,\beta) \equiv_m^{\mathrm{L}} \delta\mathcal{P}(\alpha,\beta)$ *for* $\mathcal{P} \in \{\mathrm{NDP},\ \mathrm{NMDP}\}$ *,*

(2) $\mathrm{S}\delta\mathcal{P}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathcal{P}(\alpha,\beta)$ *for* $\mathcal{P} \in \{\mathrm{NUP},\ \mathrm{NIP},\ \mathrm{NMUP},\ \mathrm{NMIP}\}$ *,*

(3) $\bar{\mathrm{S}}\delta\mathcal{P}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathcal{P}(\alpha,\beta)$ *for* $\mathcal{P} \in \{\mathrm{NDP},\ \mathrm{NUP},\ \mathrm{NIP},\ \mathrm{NMDP},\ \mathrm{NMUP},\ \mathrm{NMIP}\}$ *.*

Proof: The reductions of statements (2–3) and the reductions from left hand side to right hand side of statement (1) are clear: for a given instance, it only has to be verified that the specified start and end vertices are different or equal, depending on the problem. So, it remains to show the reductions from right hand side to left hand side of statement (1). We first show that $\mathrm{NDP}(\alpha,\beta)$ reduces to $\mathrm{SNDP}(\alpha,\beta)$. Let (G,k,ν,u,v) be an instance of $\mathrm{NDP}(\alpha,\beta)$ where $G = (V,A)$. If u and v are different, then $(G,k,\nu,u,v) \in \mathrm{NDP}(\alpha,\beta)$ if and only if $(G,k,\nu,u,v) \in \mathrm{SNDP}(\alpha,\beta)$. If u and v are equal, we construct the following graph G'. Let u' be a vertex that is not contained in G. Let $G' =_{\mathrm{def}} (V',A')$ where $V' =_{\mathrm{def}} V \cup \{u'\}$ and $A' =_{\mathrm{def}} A \cup \{(x,u') : (x,u) \in A\}$, i.e., we add a copy of u that has the same incoming arcs as u but no outgoing arc. There is a 1-1 correspondence between the u,u-paths of G and the u,u'-paths of G' by replacing end vertex u by u' and vice versa. Hence, $(G,k,\nu,u,u) \in \mathrm{NDP}(\alpha,\beta)$ if and only if $(G',k,\nu,u,u') \in \mathrm{SNDP}(\alpha,\beta)$. A similar construction also works for the other problems of statement (1). Note that the additional vertex u' must be marked if and only if u is marked. ■

The construction of the proof above for showing $\mathrm{NDP}(\alpha,\beta) \leq_m^{\mathrm{L}} \mathrm{SNDP}(\alpha,\beta)$ cannot be used for the undirected problems. Consider the connected graph on two vertices; let its vertices be u_1 and u_2. We ask for the number of u_1,u_1-paths. Certainly, there are not more than one for given length; however, the transformed graph, containing a “copy” u_1' of u_1, contains two u_1,u_1'-paths of length 4, four u_1,u_1'-paths of length 6. We cannot avoid that u_1' is visited more than once; in the directed case, this copy vertex has no outgoing arcs. We only want to mention at this point that $\leq_m^{\mathrm{L}}$ can be replaced by $\equiv_m^{\mathrm{L}}$ in Lemma 9 in most cases, which is later shown by completeness results.

Lemma 10. *For all* $\alpha,\beta \in \{1,2\}$ *and all* $\delta \in \{\varepsilon, \mathrm{S}, \bar{\mathrm{S}}, \mathrm{X}, \mathrm{SX}, \bar{\mathrm{S}}\mathrm{X}\}$ *,*

(1) $\delta\mathrm{NUP}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathrm{NDP}(\alpha,\beta)$ *,*

(2) $\delta\mathrm{NMUP}(\alpha,\beta) \leq_m^{\mathrm{L}} \delta\mathrm{NMDP}(\alpha,\beta)$ *,*

(3) $\delta\mathrm{NIP}(\alpha,\beta) \subseteq \delta\mathrm{NUP}(\alpha,\beta)$ *,*

(4) $\delta\mathrm{NMIP}(\alpha,\beta) \subseteq \delta\mathrm{NMUP}(\alpha,\beta)$ *.*

Proof: We first prove statements (1–2). Let $G = (V,E)$ be an undirected graph. Consider the following definition. Let $G' =_{\mathrm{def}} (V,A)$ be a directed graph where $A =_{\mathrm{def}} \{(u,v),(v,u) : \{u,v\} \in E\}$. For every pair x,z of vertices of G, it holds that P is an x,z-path in G if and only if P is an x,z-path in G'. Futhermore, by definitions of equal paths in directed and undirected graphs, two paths P_1 and P_2 of G are equal if and only if P_1 and P_2 are equal paths in G'. Hence, for every pair x, z of vertices of G and every number $k \geq 0$, the number of x, z-paths of length k in G is equal to the number of x, z-paths of length k in G'. Given G, G' can be computed in logarithmic space.

For statements (3–4), let $G = (V, E)$ be an undirected graph. Let u and v be vertices of G, and let $k \geq 0$. Let $\mathfrak{M}$ be the set of u, v-paths of length k in G. Isomorphism is an equivalence relation on $\mathfrak{M}$, so that the maximal number of pairwise non-isomorphic u, v-paths of length k in G is at most the cardinality of $\mathfrak{M}$, i.e., the number of u, v-paths of length k in G is an upper bound for the maximal number of pairwise non-isomorphic u, v-paths of length k in G. Since this relation also holds if we restrict to subsets of $\mathfrak{M}$, the same relation holds for marked graphs. ∎

In this paper, we mostly use variants of many-one reducibility. The last result in this section gives a relation between lower-bound number-of-paths problems and their corresponding exact-number problems. Here, we need a variant of truth-table reducibility. Let h be a k-ary Boolean function. We say that A is *logarithmic-space truth-table reducible* to B *via* h, $A \leq^{\mathrm{L}}_{h\mathrm{tt}} B$, if there is an FL-function that, on input x, outputs k words $y_1, \ldots, y_k$ such that $x \in A$ if and only if $h(c_B(y_1), \ldots, c_B(y_k)) = 1$. Observe that, for every Boolean function, NL, PL and P are closed under fixed-function logarithmic-space truth-table reducibility (Theorem 1).

Lemma 11. *Let* $a(x_1, x_2) =_{\mathrm{def}} (x_1 \wedge \neg x_2)$. *For all* $\alpha, \beta \in \{1, 2\}$, *all* $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}\}$ *and all* $\mathcal{P} \in \{\mathrm{NDP}, \mathrm{NMDP}, \mathrm{NUP}, \mathrm{NMUP}, \mathrm{NIP}, \mathrm{NMIP}\}$, $\delta\mathrm{x}\mathcal{P}(\alpha, \beta) \leq^{\mathrm{L}}_{a\mathrm{tt}} \delta\mathcal{P}(\alpha, \beta)$.

Proof: Let (G, k, ν, u, v) be an instance of $\mathrm{xNDP}(\alpha, \beta)$. $(G, k, \nu, u, v) \in \mathrm{xNDP}(\alpha, \beta)$ if and only if $(G, k, \nu, u, v) \in \mathrm{NDP}(\alpha, \beta)$ and $(G, k, \nu{+}1, u, v) \notin \mathrm{NDP}(\alpha, \beta)$. Adding a constant to a number can be done in logarithmic space for every α-ary number system. The other proofs are analogous. ∎

5. Number-of-paths problems for marked graphs. We will show that the number-of-paths problems for directed marked graphs reduce to the corresponding problems for unmarked graphs, if we use a weaker reducibility than our standard logarithmic-space many-one reducibility. For a marked graph, if there is a path containing a given number of marked vertices and that passes through a cycle without marked vertices, this cycle can be passed through arbitrarily often and we always obtain a new path with the same given number of marked vertices. If there is such a path, our number-of-paths problems are easy. If there is no such path, the given graph has some interesting structural properties. This brief sketch is the basic idea of our reduction. We will see that it can be determined using only nondeterministic logarithmic space which of the two cases holds for an input graph.

We say that a function f is an FNL-function if there are a logarithmic-space Turing transducer M and set $A \in \mathrm{NL}$ such that f can be computed by M^A. Since $\mathrm{L}^{\mathrm{NL}} = \mathrm{NL}$ the definition of FNL is well-justified. Set A NL-*reduces* to set B, $A \leq^{\mathrm{NL}}_{m} B$, if there is a function $f \in \mathrm{FNL}$ such that, for all $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) \in B$.

Lemma 12. *The complexity classes* NL, $\mathrm{C_{=}L}$ *and* PL *are closed under* NL*-reducibility.*

As an auxiliary problem we have to decide whether a graph contains a certain path of given length. Let $G = (V, A)$ be a directed graph. For two vertices $u, v \in V$ and $\ell \in \mathbb{N}$, let P_G be the function such that $P_G(\overrightarrow{uv}, \ell) \in \{0, 1\}$ and $P_G(\overrightarrow{uv}, \ell) = 1$ if and only if there is a u, v-path of length ℓ in G. With P_G, we associate the decision problem PATH that is the set of tuples (G, u, v, ℓ) where $G = (V, A)$ is a directed graph, $u, v \in V$, $\ell \in \mathbb{N}$ is represented in binary form and $P_G(\overrightarrow{uv},\ell)=1$. For an acyclic single-source graph G we define the *depth* of a vertex u of G as the length of a longest path from the source vertex to u in G. Note that acyclic single-source graphs are connected. By DEPTH we denote the set of triples (G,u,ℓ) where G is an acyclic single-source graph, and vertex u has depth ℓ in G.

Lemma 13. PATH *and* DEPTH *are in* NL.

Proof: Let (G,u,v,ℓ) be an instance of PATH, and let n be the number of vertices of G. Let $P=(u_0,\ldots,u_\ell)$ be a u,v-path in G. If $\ell\le(n+1)^2$ we can find a path straightforwardly by guessing ℓ successor vertices. Now, let $\ell>(n+1)^2$. Let $r\in[0,n-1]$ such that there is $d\in[1,n-r]$ and $u_r=u_{r+d}$. Consider the vertices $(u_r,u_{r+d},u_{r+2d},\ldots,u_{r+ed})$ where $\ell-d<r+ed\le\ell$. The first two vertices of this sequence are equal. If there are other positions containing the same vertices, say $u_{r+id}=u_{r+jd}$, $i<j$, $2\le j\le e$, we can delete the enclosed subsequence, precisely $(u_{r+id},\ldots,u_{r+(j-1)d})$, and add further copies of u_r from the left. We obtain a sequence of vertices containing at most $n-1$ vertices different from u_r, and all these are at the right end of the sequence. This shows that there is a u,v-path $P'=(v_0,\ldots,v_\ell)$ where $v_i=u_i$ for all $i\in[0,r+d]$ and $v_{r+jd}=u_r$ for all $j\in[0,e-n+1]$. This structural property gives an NL-algorithm for solving the problem. First, guess vertex w of G and $d\le n$ and verify the existence of a path $(w_0,\ldots,w_d)$ where $w_0=w_d=w$. Then, verify the existence of a u,v-path $(x_0,\ldots,x_m)$ where $m<n+(n-1)\cdot d+d=n\cdot(d+1)$, $m\equiv\ell \pmod d$, and that there is $i<n$ such that $x_i=w$.

Let (G,u,k) be an instance of DEPTH. In nondeterministic logarithmic space, it can be determined by standard procedures whether G is a connected acyclic single-source graph. Let s be the unique source of G. Determine the largest number ℓ such that $P_G(\overrightarrow{su},\ell)=1$, which can be done in logarithmic space with oracle access to PATH. If the largest number is equal to k accept, otherwise reject. ■

Theorem 14. *For all* $\alpha,\beta\in\{1,2\}$ *and* $\delta\in\{\varepsilon,\mathrm{X}\}$, $\delta\mathrm{NMDP}(\alpha,\beta)\le_m^{\mathrm{NL}}\delta\mathrm{NDP}(\alpha,\beta)$.

Proof: We first prove the theorem for $\delta=\varepsilon$; the case $\delta=\mathrm{X}$ is discussed at the end of the proof and is similar in large parts. Let (G,M,k,ν,u,v) be an instance of $\mathrm{NMDP}(\alpha,\beta)$. Let $G=(V,A)$ and $n=_{\mathrm{def}}|V|$. If there is a vertex $x\in V$ such that $(x,x)\in A$, replace this loop by the path (x,w,x) for some new vertex w. If $u\notin M$ then add a new vertex u' to G that is predecessor of every successor of u, add u' to M and increase k by 1; u' becomes the new start vertex. If $v\notin M$ then add a new vertex v' to G that is successor of every predecessor of v, add v' to M and increase k by 1; v' becomes the new end vertex. So, without loss of generality we can assume that G is loop-free and $u,v\in M$. For every $x\in M$, let $G_x=_{\mathrm{def}}(V\setminus(M\setminus\{x\}),E_x)$ where $E_x=_{\mathrm{def}}E\cap(V\setminus(M\setminus\{x\}))^2$. For every pair $x,z\in M$ of vertices, let K'_{xz} denote the set of vertices $w\in V$ for which there exist an x,w-path in G_x and a w,z-path in G_z. Let $K_{xz}=_{\mathrm{def}}K'_{xz}\cup\{x,z\}$ if $K'_{xz}\ne\emptyset$ or $(x,z)\in A$; otherwise let $K_{xz}=_{\mathrm{def}}K'_{xz}$. With oracle set PATH it can be tested whether K_{xz} is empty. Suppose there are $x,z\in M$ such that the subgraph of G induced by K'_{xz}, $G[K'_{xz}]$, has a cycle. Then, there are infinitely many x,z-paths in G containing only two vertices from M. Such vertices x and z can be found in logarithmic space with oracle access to PATH. If there are a u,x-path containing exactly k_1 vertices from M and an x,v-path containing exactly k_2 vertices from M such that $k_1+k_2=k$ then the input tuple must be accepted. To verify the existence of these paths we construct graph G^* which contains a copy of every vertex in M and arc (p,q) for two vertices $p,q\in M$ if and only if K_{pq} is not empty. Let G'_{pq} be the disjoint union of two copies of G^* where the vertices of the first copy have index 1 whereas the vertices of the second copy have index 2 and containing the arc (p_1,q_2). Then,

G contains a u,v-path containing k vertices from M and containing p and q as consecutive vertices from M if and only if G'_{pq} contains a u_1,v_2-path of length k. For every pair of vertices $x,z \in M$ where $G[K'_{xz}]$ contains a cycle verify whether G'_{xz} has a u_1,v_2-path of length k. If such a path has been found output some element of $\mathrm{NDP}(\alpha,\beta)$.

Let it be the case that no such path exists (and has been found). For every pair of vertices $x,z \in M$, let G_{xz} emerge from $G[K_{xz}]$ where, in case of $x \neq z$, arcs of the forms (w,x) and (z,w) are deleted. By our assumption G_{xz} contains (at least) three vertices. If $x \neq z$ and G_{xz} is not acyclic then let G_{xz} be empty; if $G_{xx} - (V \times \{x\})$ is not acyclic then let G_{xx} be empty. For every pair of vertices $x,z \in M$, $x \neq z$, determine the depth of z in G_{xz}. Let the maximum value be denoted by a'. For every $x \in M$, determine the depth of every predecessor of x in G_{xx} in $G_{xx} - (V \times \{x\})$. Let the maximum be denoted by a''. Let $a =_{\mathrm{def}} \max\{a', a''+1\}$. With oracle access to Depth, a can be determined in logarithmic space. Observe the following: Let $P = (w_0,\dots,w_\ell)$ be a u,v-path in G containing exactly k vertices from M, let $I \subseteq [0,\ell]$ such that, for all $i \in [0,\ell]$, $w_i \in M$ if and only if $i \in I$. Then, for consecutive w_i, w_j, $i,j \in I$, $(w_i, w_{i+1}, \dots, w_j)$ is a path in $G_{w_i w_j}$ (remember that the first case, considered in the paragraph above, has not been applicable). We add new vertices to make paths between vertices from M of equal length. Let $x,z \in M$. Let H_{xz} emerge from G_{xz} as follows. H_{xz} contains a copy of every vertex in G_{xz}. (We add further vertices and arcs.) For every pair of vertices p and q where (p,q) is an arc in G_{xz}, if $x \neq z$ determine the depths of p and q in G_{xz}, if $x = z$ and $q \neq x$ determine the depths of p and q in $G_{xx} - (V \times \{x\})$, if $x = q = z$ determine the depth of p in $G_{xx} - (V \times \{x\})$. Let the depths be denoted by d_p and d_q, respectively. If $q \neq z$ let $d =_{\mathrm{def}} d_q - d_p - 1$, if $q = z$ let $d =_{\mathrm{def}} a - d_p - 1$. Add the path $(p, w_1, \dots, w_d, q)$ to H_{xz}. Observe that there is a 1-1 correspondence between the x,z-paths in G_{xz} and H_{xz}. Let H be the union of all H_{xz} for $x,z \in M$ where all copies of the same vertex from M are identified. Note that H is a graph similar to G^* but that still contains information about the number of paths. It holds that the u,v-paths containing exactly k vertices from M in G correspond to the u,v-paths of length $a \cdot (k-1)$ in H and vice versa. H can be computed in logarithmic space using oracle sets Path and Depth. Hence, $(G,M,k,\nu,u,v) \in \mathrm{NMDP}(\alpha,\beta)$ if and only if $(H, a\cdot(k-1), \nu, u, v) \in \mathrm{NDP}(\alpha,\beta)$. We conclude the theorem by Lemma 13.

In case of $\delta = \mathrm{X}$, we can basically use the same construction with the following slight modification: if we find a cycle, i.e., if the test of the first paragraph is successful and the graph contains infinitely many requested paths, output a non-element of $\mathrm{XNDP}(\alpha,\beta)$. The construction of the second paragraph can be used without changes, since it preserves the number of paths. ■

As a consequence of Theorem 14, it suffices to study containment, i.e., upper complexity bounds, of the problems for directed paths only for unmarked graphs.

In case of undirected graphs, we cannot show a similar reduction from problems for marked graphs to problems for unmarked graphs. Yet, we can reduce the set of instances that have to be considered. We introduce a new class of problems—subsets of already known ones—where we restrict the sets of marked vertices. Let $\mathrm{RNMUP}(\alpha,\beta)$ denote the set of instances (G,M,k,ν,u,v) of $\mathrm{NMUP}(\alpha,\beta)$, $\alpha,\beta \geq 1$, where $N_G(x) \subseteq M$ for every vertex $x \in V(G) \setminus M$. In other words, unmarked vertices of graphs of instances in $\mathrm{RNMUP}(\alpha,\beta)$ are adjacent only to marked vertices. The other types of problems are defined in the usual way. Similar definitions are also made for problems for isomorphic paths.

Theorem 15. *For all $\alpha, \beta \in \{1,2\}$ and $\delta \in \{\varepsilon, \text{S}, \text{X}, \text{SX}\}$,*

(1) $\delta\text{RNMUP}(\alpha,\beta) \leq_m^{\text{L}} \delta\text{NMUP}(\alpha,\beta)$,

(2) $\delta\text{RNMIP}(\alpha,\beta) \leq_m^{\text{L}} \delta\text{NMIP}(\alpha,\beta)$,

(3) $\delta\text{NMUP}(\alpha,\beta) \leq_m^{\text{NL}} \delta\text{RNMUP}(\alpha,\beta)$,

(4) $\delta\text{NMIP}(\alpha,\beta) \leq_m^{\text{NL}} \delta\text{RNMIP}(\alpha,\beta)$.

Proof: For statements (1) and (2), it suffices to verify the correctness of the structural restriction on the sets of unmarked vertices. Given an instance of such a problem, in logarithmic space it can be determined whether every unmarked vertex has only marked neighbours.

For the proof of statements (3) and (4), let $G = (V, E)$ be an undirected graph, and let $M \subseteq V$ be the set of marked vertices of G. Let u, v be vertices of G, and let $k \geq 0$. Suppose there are a k-marked u, v-path $P = (z_0, \ldots, z_r)$ in G and $i \in [0, r-1]$ such that neither z_i nor z_{i+1} belongs to M, then there are infinitly many pairwise isomorphic k-marked u, v-paths in G: pass the cycle (z_i, z_{i+1}, z_i) arbitrarily often. The proof of Theorem 14 gives an algorithm for deciding this case in nondeterministic logarithmic space, if we apply the algorithm to the directed graph $G_1 =_{\text{def}} (V, \{(x,z),(z,x) : xz \in E\})$. Note that z_i and z_{i+1} define a cycle of only unmarked vertices in G_1. The reducing function outputs a correct instance of the problem to which we reduce. Now, let G contain no such path, i.e., no k-marked u, v-path in G contains two consecutive unmarked vertices. Let G_2 emerge from G by deleting every edge between unmarked vertices. G_2 can be computed in logarithmic space. By assumption and construction, the sets of k-marked u, v-paths in G and G_2 are equal. This establishes the desired reductions. ∎

6. Determining small numbers of paths. In this section, we will determine the complexity of number-of-paths problems for which the queried number of paths is given in unary form. The main result will be that all these problems are solvable using nondeterministic logarithmic space. A central idea is to use an interesting result by Fredman, Komlós, Szemerédi, which is stated in Lemma 16.

Lemma 16. **[8]** *Let $S \subseteq \mathbb{N}$ be finite. Let $n =_{\text{def}} |S|$, and let $m =_{\text{def}} \max S$. Then, there is a prime number $p < n^2 \log m$ such that no two elements of S lie in the same residue class modulo p.*

Allender, Reinhardt, Zhou considered a restricted variant of problem NDP$(1,1)$. They were interested in the number of paths in acyclic graphs. They used Lemma 16 to obtain an NL-algorithm [2]. In our case, we have to solve the problem how to find sufficiently many paths of a certain length if this length is given in binary representation. The following result shows that we only have to find enough paths of a certain *short* length.

Lemma 17. *Let $G = (V, A)$ be a directed graph, and let $n =_{\text{def}} |V|$. Let $k, r \in \mathbb{N}$, $2n^2 \leq r < k - n$, and let $P_1 = (u_0, \ldots, u_k)$ and $P_2 = (v_0, \ldots, v_k)$ be two paths in G such that $u_i = v_i$ for all $i \in [0, r] \cup \{k\}$ and $u_{r+1} \neq v_{r+1}$. Then, there are $r' \in [r-2n^2, r-1]$ and path $P = (w_0, \ldots, w_k)$ such that $u_i = w_i$ for all $i \in [0, r'] \cup \{k\}$ and $u_{r'+1} \neq w_{r'+1}$.*

Proof: Let $s_1, s_2 \in [r-2n^2, r-2n^2+n]$, $s_1 < s_2$, such that $u_{s_1} = u_{s_2}$; the existence of s_1 and s_2 is guaranteed by the pigeon hole principle. Let $d =_{\text{def}} s_2 - s_1$. If there is a u_{s_1}, u_{s_2}-path of length d in G that is different from $P_u =_{\text{def}} (u_{s_1}, u_{s_1+1}, \ldots, u_{s_2})$, replace this sequence in P_1 by the other path and obtain the desired path P. Otherwise, P_u is the unique u_{s_1}, u_{s_2}-path of length d in G. Consider vertices $u_{s_2}, u_{s_2+d}, u_{s_2+2d}, \dots, u_{s_2+ed}$ where $r - d < s_2 + ed \le r$. If there are $a_1, a_2 \in [0, e]$, $a_1 < a_2$, such that $u_{s_2+a_1d} = u_{s_2+a_2d} \neq u_{s_2}$ or if there are $a_1, a, a_2 \in [0, e]$, $a_1 < a < a_2$, such that $u_{s_2+a_1d} = u_{s_2+a_2d} = u_{s_2}$ and $u_{s_2+ad} \neq u_{s_2}$, then, extract $(u_{s_2+a_1d}, u_{s_2+a_1d+1}, \dots, u_{s_2+a_2d})$ from P_1, insert appropriate many copies of P_u after u_{s_2} and obtain P. Otherwise, $u_{s_1} = u_{s_2} = u_{s_2+d} = \dots = u_{s_2+(e-n+1)d}$. Let $c_1, c_2 \in [k-n, k]$, $c_1 < c_2$, such that $u_{c_1} = u_{c_2}$, and let $d' =_{\text{def}} c_2 - c_1$. Obtain P' from P_1 by deleting the subpath $(u_{s_1+1}, u_{s_1+2}, \dots, u_{s_1+d'd})$ and adding d copies of $(u_{c_1+1}, u_{c_1+2}, \dots, u_{c_2})$ at position c_1 in P_1. Certainly, P' is a u_0, u_k-path of length k. If P' is different from P_1, we found path P. Otherwise, $u_{s_1} = u_{s_2} = u_{s_2+d} = u_{s_2+2d} = \dots = u_{s_2+(e+1)d}$ by induction. Now, P_1 and P_2 differ in u_{r+1} and v_{r+1}. So, we repeat the construction of the last case with P_2 instead of P_1 and obtain the desired path P, since P_u is unique. ■

For a directed graph $G = (V, A)$ and two vertices u, v of G, let $\#_G(\overrightarrow{uv}, k)$ denote the number of u, v-paths of length $k \ge 0$ in G, i.e., the function that we wish to compute.

Corollary 18. *Let $G = (V, A)$ be a directed graph, and let $n =_{\text{def}} |V|$. Let $u, v \in V$ be two vertices of G, and let ν and k such that $0 \le 2\nu n^2 \le k$. If $\nu \le \#_G(\overrightarrow{uv}, k)$, then*

$$\nu \le \sum_{w \in V} \#_G(\overrightarrow{uw}, 2\nu n^2) \cdot P_G(\overrightarrow{wv}, k - 2\nu n^2)\,.$$

Proof: Let $P_1 = (u_0, \dots, u_k)$ and $P_2 = (v_0, \dots, v_k)$ be two u, v-paths in G such that there is $r \in [2\nu n^2, k-n-1]$ such that $u_i = v_i$ for all $i \in [0, r]$ and $u_{r+1} \neq v_{r+1}$. By iterated application of Lemma 17 we obtain the claim. ■

To solve problem NDP$(2,1)$ we will apply the method of Allender, Reinhardt, Zhou [2]. Let G be a directed graph. Our chosen representation of G establishes an ordering on the vertices of G by their names. So, every path in G can uniquely be interpreted as a natural number in $(n+1)$-ary representation, where n is the number of vertices of G. Hence, we can determine the residue class modulo some number for every path. Let ResidueOfPath be the set of all tuples (G, k, q, r, u, v) where $G = (V, A)$ is a directed graph, $k, q, r \in \mathbb{N}$, k and q are represented in unary form and $u, v \in V$ such that there is a u, v-path of length k in G whose $(n+1)$-ary representation, where $n =_{\text{def}} |V|$, lies in the residue class of r modulo q.

Lemma 19. ResidueOfPath *is in* NL.

Proof: Let (G, k, q, r, u, v) be an instance of ResidueOfPath. If $r \ge q$ compute $r' \in [0, q-1]$ such that $r' \equiv r \pmod{q}$ and replace r by r'. Guess a u, v-path of length k iteratively and compute its residue class mod q. Since k and q are given in unary representation, this algorithm needs logarithmic space. ■

Theorem 20. NDP$(2,1)$ *is in* NL.

Proof: Let (G, k, ν, u, v) be an instance of NDP$(2,1)$. Let $n =_{\text{def}} |V|$. By Lemma 13, Path is in NL. To decide whether $\#_G(\overrightarrow{uv}, k) \ge \nu$ it suffices, due to Corollary 18, to guess numbers $\nu_w \le \nu$ for all $w \in V$ such that $\nu \le \sum_{w \in V} \nu_w \cdot P_G(\overrightarrow{wv}, k - 2\nu n^2)$. Remember that ν is given in unary representation, so that the guessed numbers can be represented in binary form in logarithmic space. To verify the sufficiency of each guessed ν_w where $P_G(\overrightarrow{wv}, k - 2\nu n^2) = 1$ we apply the following logarithmic-space algorithm with oracle ResidueOfPath. Note that all paths of length $2\nu n^2$ have a numerical representation with value not exceeding $(n+1)^{2\nu n^2} = 2^{2\nu n^2 \cdot \log(n+1)} \le 2^{2\nu n^3}$. By Lemma 16, we have to find a prime number $p < (\nu_w)^2 \cdot 2\nu n^3$ such that there are ν_w residue classes modulo q each containing the numerical representation of a u, w-path of length $2\nu n^2$. This can be done in logarithmic space. Since $\mathrm{L}^{\mathrm{NL}} = \mathrm{NL}$, $\mathrm{NDP}(2,1)$ is in NL. ■

Theorem 21. *For $\alpha \in \{1,2\}$ and $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}\}$, $\delta\mathrm{NDP}(\alpha,1)$, $\delta\mathrm{NMDP}(\alpha,1)$, $\delta\mathrm{NUP}(\alpha,1)$, $\delta\mathrm{NMUP}(\alpha,1)$ and $\delta\mathrm{POM}(\alpha,1)$ are NL-complete.*

Proof: For containment, we apply Lemmata 5, 8, 9, 10 and 7 and Theorem 14 to conclude that all mentioned problems NL-reduce to $\mathrm{NDP}(2,1)$, which is contained in NL due to Theorem 20. Since NL is closed under $\leq_m^{\mathrm{NL}}$-reducibility, all these problems are contained in NL.

For proving hardness, we apply the same idea that was used to show hardness for GAP. Let $A \in \mathrm{NL}$. We will reduce A to $\mathrm{sNUP}(1,1)$ and $\bar{\mathrm{s}}\mathrm{NUP}(1,1)$. Applying the results cited above we can show that $\mathrm{sNUP}(1,1)$ or $\bar{\mathrm{s}}\mathrm{NUP}(1,1)$ can be reduced to every of our problems, which shows completeness. Let $c \geq 1$ such that there is a nondeterministic logarithmic-space Turing machine M accepting A such that M on input x, $|x| \geq 2$, halts on every computation path after exactly $|x|^c$ steps. Furthermore, M accepts with a unique accepting end configuration. Let $\mathcal{C}$ be the set of possible configurations of M on input x. Construct graph G with vertex set $\mathcal{C} \times \{0, \ldots, |x|^c\}$ and the following edge set: two vertices (c_1, t_1) and (c_2, t_2) of G are adjacent if and only if $t_2 = t_1 + 1$ and c_1 yields c_2 in one step of M. Let $\mathfrak{s}$ and $\mathfrak{a}$ be the unique vertices containing the start and accepting end configuration of $M(x)$, respectively. Note that $\mathfrak{s}$ and $\mathfrak{a}$ are different. Then, M on input x accepts if and only if there is an $\mathfrak{s}, \mathfrak{a}$-path of length $|x|^c$ in G. Hence, $x \in A$ if and only if $(G, |x|^c, 1, \mathfrak{s}, \mathfrak{a}) \in \mathrm{sNUP}(1,1)$.

For the reduction from A to $\bar{\mathrm{s}}\mathrm{NUP}(1,1)$, we take graph G and subdivide every edge by a new vertex; we obtain G'. Then, every $\mathfrak{s}, \mathfrak{a}$-path in G' has even length. We add edge $\mathfrak{s}\mathfrak{a}$ to G', obtain G'' and observe that there is an $\mathfrak{s}, \mathfrak{a}$-path of length $|x|^c$ in G if and only if there is an $\mathfrak{s}, \mathfrak{s}$-path of length $2|x|^c + 1$ in G''. Hence, $x \in A$ if and only if $(G'', 2|x|^c + 1, 1, \mathfrak{s}, \mathfrak{s}) \in \bar{\mathrm{s}}\mathrm{NUP}(1,1)$. ■

Corollary 22. *For all $\alpha \in \{1,2\}$ and $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}\}$, $\delta\mathrm{xNDP}(\alpha,1)$, $\delta\mathrm{xNMDP}(\alpha,1)$, $\delta\mathrm{xNUP}(\alpha,1)$, $\delta\mathrm{xNMUP}(\alpha,1)$ and $\delta\mathrm{xPOM}(\alpha,1)$ are NL-complete.*

Proof: Since NL is closed under $\leq_{\mathrm{att}}^{\mathrm{L}}$-reducibility for $a(x_1, x_2) = (x_1 \wedge \neg x_2)$, all problems are contained in NL by Lemma 11 and Theorem 21. For hardness, reconsider the proof of Theorem 21. It holds that $x \notin A$ if and only if there is no $\mathfrak{s}, \mathfrak{a}$-path of length $|x|^c$. Note that $\mathfrak{s}$ and $\mathfrak{a}$ may lie in the same connected component of G, but any connecting path must have length at least $|x|^c + 2$. Let $u_1, \ldots, u_{|x|^c - 1}$ be new vertices. We add the path $(\mathfrak{s}, u_1, \ldots, u_{|x|^c-1}, \mathfrak{a})$ to G; we obtain G_1. Then, $x \notin A$ if and only if $(G_1, |x|^c, 1, \mathfrak{s}, \mathfrak{a}) \in \mathrm{sxNUP}(1,1)$. Since $A \in \mathrm{NL}$ if and only if $\overline{A} \in \mathrm{NL}$ due to Theorem 1, $\mathrm{sxNUP}(1,1)$ is NL-hard.

For NL-hardness of $\bar{\mathrm{s}}\mathrm{xNUP}(1,1)$, we take G'' of the proof of Theorem 21 and add an additional $\mathfrak{s}, \mathfrak{a}$-path of length $2|x|^c$ to G''; we obtain G_2. We observe that $x \notin A$ if and only if G_2 has exactly two $\mathfrak{s}, \mathfrak{s}$-paths of length $2|x|^c + 1$: one path containing $\mathfrak{a}$ as last but first, another path containing $\mathfrak{a}$ as second vertex. This shows NL-hardness of $\bar{\mathrm{s}}\mathrm{xNUP}(1,1)$. Hardness of all other problems holds due to the relevent results. ■

7. Determining large numbers of paths. As we have seen in the last section, number-of-paths problems for directed and undirected graphs can be decided in nondeterministic logarithmic space, if the queried number of paths is small, i.e., is given in unary form. We have also seen that the representations of the lengths do not have an impact on the complexity. In case of large numbers, i.e., if the queried numbers of paths are given in binary form, a homogeneous behaviour of our problems similar to the case of small numbers will not be observed. More precisely: will not be observed so entirely.

In this section, we will show completeness results for the complexity classes PL and $\mathrm{C_{=}L}$. The class PL is the set of languages A for which there is a probabilistic Turing machine using at most $\lfloor c \cdot \log |x| \rfloor$ cells on its working tape for some constant $c \geq 1$ and each input $x \in \Sigma^*$ such that $x \in A$ if and only if the probability of reaching an accepting end configuration is at least $\frac{1}{2}$. The class PL^B is defined similar to NL^B, i.e., the involved oracle machines are required to make only deterministic moves while generating oracle queries [1]. The function class #L is the set of all functions acc_M for M a nondeterministic logarithmic-space Turing machine, and $\#\mathrm{L}^B$ is defined with the given restrictions on the generation of queries. The complexity class $\mathrm{C_{=}L}$ is the set of languages A for which there are functions $f, g \in \#\mathrm{L}$ such that, for every $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) - g(x) = 0$. The following characterisations of PL and $\mathrm{C_{=}L}$ are non-trivial results; especially in case of PL: PL may compute exponentially many steps whereas #L-functions are computed by polynomial-time machines.

Theorem 23. **[1]**

(1) Language A is in PL *if and only if there are functions $f \in \#\mathrm{L}$ and $g \in \mathrm{FL}$ such that, for all $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) \geq g(x)$.*

(2) Language A is in $\mathrm{C_{=}L}$ *if and only if there are functions $f \in \#\mathrm{L}$ and $g \in \mathrm{FL}$ such that, for all $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) = g(x)$.*

Theorem 24. **[1], [12], [5]** $\mathrm{NL} \subseteq \mathrm{C_{=}L} \subseteq \mathrm{PL} = \mathrm{PL}^{\mathrm{PL}} \subseteq \mathrm{P} = \mathrm{P}^{\mathrm{P}}$.

First, we determine a complexity upper bound for the easiest number-of-paths problems with binary representation of the queried number of paths for directed graphs.

Theorem 25. $\mathrm{NDP}(1,2)$ *is in* PL *.*

Proof: Due to Theorem 23, we will define appropriate functions $f \in \#\mathrm{L}$ and $g \in \mathrm{FL}$ to show containment. Let (G, k, ν, u, v) be an instance of $\mathrm{NDP}(1,2)$. Since k is given in unary form there is a nondeterministic logarithmic-space Turing machine having, on input (G, k, ν, u, v), exactly the number of u,v-paths of length k in G many accepting computation paths: starting with u guess a sequence of $k+1$ vertices each pair of consecutive vertices of which is joined by an arc pointing from the former to the latter vertex and accept if and only if the last vertex is v. If the described machine is M, let $f =_{\mathrm{def}} \mathrm{acc}_M$. Function g on input (G, k, ν, u, v) simply outputs ν, which clearly is an FL-function. Then, it holds that $(G, k, \nu, u, v) \in \mathrm{NDP}(1,2)$ if and only if $f(G, k, \nu, u, v) \geq g(G, k, \nu, u, v)$. ■

Corollary 26. $\mathrm{xNDP}(1,2)$ *is in* $\mathrm{C_{=}L}$ *.*

Proof: We take functions f and g from the proof of Theorem 25 and observe that $(G, k, \nu, u, v) \in \mathrm{xNDP}(1,2)$ if and only if $f(G, k, \nu, u, v) = g(G, k, \nu, u, v)$. We conclude the statement by Theorem 23. ■

Second, we will see that the problems above are even at least as hard as the hardest number-of-paths problems for undirected graphs. In Sections 4 and 5, we have already established a number of relations between different problems. To show the desired reduction, it suffices to show two remaining reductions, explicitly, we will show that $\mathrm{rNMUP}(2,2)$ and $\mathrm{rxNMUP}(2,2)$ NL-reduce to $\mathrm{NMUP}(1,2)$ and $\mathrm{xNMUP}(1,2)$, respectively. This seems surprising at first glance. The main reason is that, if there is one desired path, there are many such paths. As an auxiliary result, we need a combinatorial representation of Fibonacci numbers. Remember that the Fibonacci numbers are the elements of the sequence $1, 1, 2, 3, 5, 8, 13, \ldots$, described by the recurrence

$$F(m+2) =_{\text{def}} F(m+1) + F(m) \ , \quad m \geq 0$$

and the initial values $F(0) =_{\text{def}} F(1) =_{\text{def}} 1$. A closed formula for the Fibonacci numbers is also known, but we are only interested in the function's growth. By a simple induction, it is verified that $F(m) \geq 1.6^{m-1}$ for every $m \geq 0$.

Lemma 27. *For all $m \geq 1$:*

$$\begin{aligned} F(2m-2) &= \sum_{i=1}^{m} \binom{m+i-2}{2i-2} \\ F(2m-1) &= \sum_{i=1}^{m} \binom{m+i-1}{2i-1} . \end{aligned}$$

Proof: We show the claim by induction over m. It is immediate that the equalities hold for $m = 1$ and $m = 2$. Let $m + 1 \geq 3$. Then:

$$\begin{aligned} \sum_{i=1}^{m+1} \binom{m+1+i-2}{2i-2} &= \binom{2m}{2m} + \binom{m}{0} + \sum_{i=2}^{m} \binom{m+i-1}{2i-2} \\ &= \binom{m}{0} + \sum_{i=2}^{m} \binom{m+i-2}{2i-2} + \binom{2m}{2m} + \sum_{i=2}^{m} \binom{m+i-2}{2i-3} \\ &= F(2m-2) + \binom{2m-1}{2m-1} + \sum_{i=1}^{m-1} \binom{m+i-1}{2i-1} \\ &= F(2m-2) + F(2m-1) \\ &= F(2m) \end{aligned}$$

and

$$\begin{aligned} \sum_{i=1}^{m+1} \binom{m+1+i-1}{2i-1} &= \binom{2m+1}{2m+1} + \sum_{i=1}^{m} \binom{m+i}{2i-1} \\ &= \sum_{i=1}^{m} \binom{m+i-1}{2i-1} + \binom{2m}{2m} + \sum_{i=1}^{m} \binom{m+i-1}{2i-2} \\ &= F(2m-1) + \sum_{i=1}^{m+1} \binom{m+i-1}{2i-2} \\ &= F(2m-1) + F(2m) \\ &= F(2m+1) \end{aligned}$$

■

The following theorem is proved by using some combinatorial arguments. Remember the known result about the number of combinations with repetition. Let A be a set of n elements. The number of k-element combinations over A with repetition is $\binom{n+k-1}{k}$, which is the number of natural solutions of the equation that we considered in the Introduction.

Theorem 28. *For $\delta \in \{\varepsilon, \mathrm{X}\}$, $\mathrm{R}\delta\mathrm{NMUP}(2,2) \leq_m^{\mathrm{NL}} \delta\mathrm{NMUP}(1,2)$.*

Proof: We will concentrate on the proof for $\delta = \varepsilon$. The case $\delta = \mathrm{X}$ can be shown by a proof similar to the following; differences are discussed at the end of the proof.

In order to show $\mathrm{RNMUP}(2,2) \leq_m^{\mathrm{NL}} \mathrm{NMUP}(1,2)$, we will describe a reducing function that has access to an NL-set oracle. Let (G, M, k, ν, u, v) be an instance of $\mathrm{RNMUP}(2,2)$. Let n be the number of vertices of G. If $k \leq 3n+5$, k can be written in unary representation using only logarithmic space, and our $\mathrm{RNMUP}(2,2)$ instance can be transformed into an $\mathrm{NMUP}(1,2)$ instance. Let $k > 3n+5$, and let $s =_{\mathrm{def}} \lfloor \frac{k-n}{2} \rfloor$. If $\nu \geq 2^s$, i.e., the binary representation of ν takes at least $s+1$ digits, we can also transform our instance into an instance of $\mathrm{NMUP}(1,2)$ using only logarithmic space. For the remaining cases, i.e., if $3n+5 < k$ and $\nu < 2^s$, we will show that it suffices check some structural properties of G to know whether to accept or reject. Acception and rejection mean that the reducing function maps the input instance to an instance of $\mathrm{NMUP}(1,2)$ that has to be accepted or rejected.

We first give the algorithm, i.e., describe the reducing function, and second, we prove correctness. If $\nu = 0$, then accept. So, let $\nu > 0$. If u and v are not contained in the same connected component of G, i.e., there is no u,v-path in G, the input must be rejected. This can be checked using GAP as oracle. Otherwise, u and v are contained in the same connected component of G; let it be C. Let n_C denote the number of vertices of C. If $n_C \leq 2$, which can be verified easily and without oracle access, there is at most one k-marked u,v-path in G, and it can be decided in logarithmic space whether the input has to be accepted or rejected. So, let $n_C \geq 3$. Note that every vertex of C is contained in a path of length 2 containing three different vertices. If C contains an unmarked vertex, accept. Otherwise, i.e., C contains only marked vertices, we check whether there is $k' \leq 2n$ such that there is a k'-marked u,v-path in G and $k' \equiv k \pmod 2$. Note that there is a k-marked u,v-path in G if and only if there is such a k'. For the one direction, take a k-marked u,v-path and label every second vertex beginning with the first. If $k > 2n$, we labelled $n+1$ vertices, and there must be two labelled occurrences of the same vertex. We can cut the subpath between these two occurrences, merge the cut ends and obtain a shorter u,v-path. Repeated application yields the desired path. For the other direction, if there is an appropriate k'-marked u,v-path in G, we add a sequence $(v', v, v', \ldots, v)$ of appropriate length to the k'-marked path to obtain a k-marked u,v-path; here, v' is a neighbour of v, that exists. If there is such a k', accept, else reject.

We prove the correctness of our algorithm above. We see that rejection is always correct. So, we have to consider the cases which lead to acception. We will show that there are always many k-marked u,v-paths in G. We distinguish five cases.

(1) C does not contain any unmarked vertex. Let $k' \leq 2n$ be such that there is a k'-marked u,v-path $P = (z_0, \ldots, z_r)$ in G and $k' \equiv k \pmod 2$. Note that $r = k' - 1$. Let w_1, w_2, w_3 be pairwise different vertices of C such that (w_1, w_2, w_3) is a path in G and contains v. Let $\kappa =_{\mathrm{def}} k - k' - 2$, which is an even number. Let $\kappa' =_{\mathrm{def}} \frac{1}{2}\kappa$. For every $i \in \{1, 2, \ldots, \kappa'\}$, let $a_i \in \{1, 3\}$. Assume $w_1 = v$. Then,

$$(z_0, \ldots, z_r = v = w_1, w_2, w_{a_1}, w_2, w_{a_2}, \ldots, w_2, w_{a_{\kappa'}}, w_2, w_1 = v)$$

is a k-marked u,v-path in G. If $w_3 = v$, we use a similar construction. If $w_2 = v$, we consider paths of the from

$$(z_0, \ldots, z_r = v = w_2, w_{a_1}, w_2, w_{a_2}, \ldots, w_2, w_{a_{\kappa'}}, w_2, w_1, w_2 = v)\,.$$

In all three cases, we can construct $2^{\kappa'}$ different k-marked u, v-paths, since there are $2^{\kappa'}$ choices for $(a_i)_{i\in[1,\kappa']}$.

(2) There are pairwise different vertices w_1, w_2, w_3 in C such that (w_1, w_2, w_3) is a path in G and $w_1, w_3 \notin M$ and $w_2 \in M$. Let P_1 be a shortest u, v-path in G, and let k_1' be the number of marked vertices in P_1. Let P_2 be a shorted u, w_1-path in G, and let P_3 be a shortest w_3, u-path in G. Let k_2' and k_3' denote the numbers of marked vertices in P_2 and P_3, respectively. Then, $P =_{\text{def}} P_2 \circ (w_1, w_2, w_3) \circ P_3 \circ P_1$ is a $(k_2' + 1 + k_3' + k_1')$-marked u, v-path in G. Note that $k_1' + k_2' + k_3' \le 3n$. Let $k' =_{\text{def}} k_2' + 1 + k_3' + k_1'$, and let $\kappa =_{\text{def}} k - k'$. For every $i \in \{1, \ldots, \kappa\}$, let $a_i \in \{1, 3\}$. Then,

$$P_2 \circ (w_1, w_2, w_{a_1}, w_2, \ldots, w_{a_\kappa}, w_2, w_3) \circ P_3 \circ P_1$$

is a k-marked u, v-path in G. Using similar argumentation as in case (1), there are at least 2^κ k-marked u, v-paths in G.

(3) This case is inverse to case (2). There are pairwise different vertices w_1, w_2, w_3 in C such that (w_1, w_2, w_3) is a path in G and $w_1, w_3 \in M$ and $w_2 \notin M$. Let P_1 be a shortest u, v-path in G, and let P_2 and P_3 be a shorted u, w_1- and w_3, u-paths in G, respectively. Let k_1', k_2' and k_3' denote the numbers of marked vertices in P_1, P_2 and P_3, respectively. Then, $P =_{\text{def}} P_2 \circ (w_1, w_2, w_3) \circ P_3 \circ P_1$ is a $(k_2' + k_3' + k_1')$-marked u, v-path in G. Let $k' =_{\text{def}} k_2' + k_3' + k_1'$, and let $\kappa =_{\text{def}} k - k'$. Similar to case (2), we can construct at least 2^κ k-marked u, v-paths in G.

(4) There are vertices w_1, w_2, w_3 in C such that (w_1, w_2, w_3) is a path in G and $w_1 \notin M$ and $w_2, w_3 \in M$. Let P_1, P_2 and P_3 be shortest u, v-, u, w_1- and w_3, u-paths in G, respectively, and let k_1', k_2' and k_3' denote the numbers of marked vertices in P_1, P_2 and P_3, respectively. Then, $P =_{\text{def}} P_2 \circ (w_1, w_2, w_3) \circ P_3 \circ P_1$ is a $(k_2' + 1 + k_3' + k_1')$-marked u, v-path in G. Let $k' =_{\text{def}} k_2' + 1 + k_3' + k_1'$. We want to show in a way similar to the previous cases that there are many k-marked u, v-paths. But note that we cannot choose w_1 and w_3 arbitrarily as before, since w_3 is marked and w_1 is unmarked, i.e., the number of marks in the final path depends on the number of choices of w_1. Let $\kappa =_{\text{def}} k - k'$. If κ is odd, w_1 has to be chosen odd-times often, whereas, if κ is even, w_1 has to be chosen even-times often. In both cases, it holds $\kappa = a + 2b$ where a is the number of w_1-choices and b is the number of w_3-choices. We determine the number of possible and correct choices for $(a_i)_{i\in[1,\kappa]}$. For the case κ *is odd*, let $\kappa' =_{\text{def}} \frac{1}{2}(\kappa + 1)$, and we obtain the following (in-)equalities:

$$\sum_{i=1}^{\kappa'} \binom{\kappa' - i + 1 + (2i-1) - 1}{2i - 1} = \sum_{i=1}^{\kappa'} \binom{\kappa' + i - 1}{2i - 1} = F(2\kappa' - 1) = F(\kappa) \ge 1.6^{\kappa - 1},$$

which is due to Lemma 27. Similarly, the case κ *is even*. Let $\kappa' =_{\text{def}} \frac{1}{2}\kappa$, and we obtain:

$$\sum_{i=0}^{\kappa'} \binom{\kappa' - i + 1 + 2i - 1}{2i} = \sum_{i=1}^{\kappa'+1} \binom{\kappa' + i - 1}{2i - 2} = F(2\kappa') = F(\kappa) \ge 1.6^{\kappa - 1},$$

i.e., there are at least $1.6^{\kappa-1}$ k-marked u, v-paths in G.

(5) This case is the analogue of case (4) where $w_1, w_2 \in M$ and $w_3 \notin M$.

In any of the five cases, it holds that $\kappa > 3$, since $k > 3n + 5$ and $\kappa \ge k - 3n - 1$. So, we obtain the following relations:

$$\nu < 2^s \le \sqrt{2^\kappa} < 1.6^{\kappa - 1} < 2^\kappa,$$

that can easily be verified. Hence, there are always more k-marked u, v-paths than requested, and (G, M, k, ν, u, v) must be accepted.

For the exact case, i.e., $\delta = \mathrm{x}$, we have to distinguish some cases. If $\nu = 0$, we can ask the oracle set xNMUP$(2,1)$, which is in NL due to Corollary 22. If $k \leq 3n+5$ or $\nu \geq 2^s$, we can reduce to an instance of xNMUP$(1,2)$, as discussed above. In all other cases, we proved that the existence of one k-marked u, v-path in G entails many such paths, more than ν, so that the input must be rejected. ■

The result of Theorem 28 enables us to settle most of the remaining cases.

Corollary 29. *For $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}\}$, δNDP$(1,2)$, δNMDP$(1,2)$, δNUP$(1,2)$, δNMUP$(1,2)$, POM$(1,2)$ as well as δNUP$(2,2)$, δNMUP$(2,2)$ are PL-complete.*

Proof: This proof is most similar to the proof of Theorem 21. Containment in PL follows from Theorem 25, Theorem 14, Theorem 28, Theorem 15 and the results of Section 4.

For hardness, it suffices to show PL-hardness for sNUP$(1,2)$ and $\bar{\mathrm{s}}$NUP$(1,2)$. However, this is easy, since we can use the constructions from Theorem 21. Let $A \in$ PL. Due to Theorem 23, there are functions $f \in \#$L and $g \in$ FL such that $x \in A$ if and only if $f(x) \geq g(x)$. Let f be computed by nondeterministic logarithmic-space Turing machine M. Similar to the proof of Theorem 21, there is $c \geq 1$ and graph G such that $f(x) \geq g(x)$ if and only if $(G, |x|^c, g(x), \mathfrak{s}, \mathfrak{a}) \in$ sNUP$(1,2)$. Analogously, PL-hardness of $\bar{\mathrm{s}}$NUP$(1,2)$ is proved. ■

Corollary 30. *For $\delta \in \{\varepsilon, \mathrm{s}, \bar{\mathrm{s}}\}$, δxNDP$(1,2)$, δxNMDP$(1,2)$, δxNUP$(1,2)$, δxNMUP$(1,2)$, xPOM$(1,2)$ as well as δxNUP$(2,2)$, δxNMUP$(2,2)$ are PL-complete.*

Proof: The proof is basically the same as the proof of Corollary 29. The upper complexity bound is determined by xNDP$(1,2)$ in Corollary 26. $\mathrm{C}_=$L-hardness of sxNUP$(1,2)$ and $\bar{\mathrm{s}}$xNUP$(1,2)$ are established due to Theorem 23 using the reduction of Corollary 29. ■

So, we have determined the complexity of all defined number-of-paths problems for undirected graphs. For problems for directed graphs, most problems could be classified; only the hardest problems remain. Unfortunately, we cannot give lower and upper complexity bounds that meet each other. Here, we consider our matrix problem for determining an upper complexity bound.

Lemma 31. POM$(2,2)$ *is in* P.

Proof: Let (M, k, a, i, j) be an instance of POM$(2,2)$. Let n be the dimension of M. We have to answer the question whether $(M^k)_{ij} \geq a$. Let M' emerge from M by replacing every entry that is larger than a by $a+1$. We denote this operation by $[M]_{\leq a}$, i.e., $M' = [M]_{\leq a}$. Then, $(M^k)_{ij} \geq a$ if and only if $((M')^k)_{ij} \geq a$. It holds that

$$[M^r]_{\leq a} = [[M^i]_{\leq a} \cdot [M^{r-i}]_{\leq a}]_{\leq a}$$

for every $r \geq 0$ and $i \in \{0, \ldots, r\}$. Observe that the left hand side matrix can be represented using only $b \cdot n^2$ bits where b is the number of bits for representing $a+1$ in binary form. Furthermore, for $r \geq 0$, $M^{2r} = M^r \cdot M^r$ and $M^{2r+1} = M^r \cdot M^r \cdot M$. Multiplying two matrices takes polynomial time, computing $[\cdot]_{\leq a}$ takes polynomial time. Since $[M^k]_{\leq a}$ can be computed using at most $2c$ matrix multiplications where c is the number of (significant) digits of the binary representation of k, $[M^k]_{\leq a}$ can be computed in polynomial time. ■

It is clear that M^k for M a quadratic matrix cannot be computed in polynomial time for all k, since entries can become very large.

Theorem 32. NDP$(2,2)$, NMDP$(2,2)$, POM$(2,2)$ *are in* P *and* PL*-hard.*

Proof: Containment is due to Lemma 7, Theorem 14 and Lemma 31. Hardness follows from Corollary 29, Lemmata 5, 7 and 8. ■

Corollary 33. xNDP$(2,2)$, xNMDP$(2,2)$, xPOM$(2,2)$ *are in* P *and* $\mathrm{C_=L}$*-hard.*

Proof: Containment of the problems is due to Lemma 11, Theorem 32 and the closure of P under fixed-function logarithmic-space truth-table reducibility. Hardness follows from Corollary 30 and Lemmata 5, 7 and 8. ■

8. Better understanding the hardest problem. NDP$(2,2)$ can be considered the hardest of all our considered problems (of course, besides all questions about problems for isomorphic paths). We could not determine its proper complexity. In order to identify useful properties of this problem, we define a new graph problem that is closely related to NDP$(2,2)$ but may provide better insight into NDP$(2,2)$.

Problem definition.

NWDP (number of weighted paths in a directed graph).
Instance. $(G,\varrho,\omega,\nu,u,v)$ where G is a directed graph, ϱ is a function that assigns natural non-zero weights to the arcs of G, ω and ν are natural numbers and u and v are vertices of G. The weights that are assigned by ϱ are given in unary from, and ω and ν are given in binary form.
Question. The *weight* of a path is the sum of the weights of the arcs appearing on the path. Are there ν u,v-paths of weight ω in G?

The problem NWDP seems familiar; the difference between NWDP and NDP$(2,2)$ is the selection of paths by weight and not by length. Therefore, the following result is not surprising.

Lemma 34. NDP$(2,2) \equiv^{\mathrm{L}}_{m}$ NWDP.

Proof: The reduction from NDP$(2,2)$ to NWDP is easily done. Let (G,k,ν,u,v) be an instance of NDP$(2,2)$. Define weight function ϱ that assigns weight 1 to every arc of G. Then, the weight of a path is equal to its length, and $(G,k,\nu,u,v) \in$ NDP$(2,2)$ if and only if $(G,\varrho,k,\nu,u,v) \in$ NWDP.

Now, let $(G,\varrho,\omega,\nu,u,v)$ be an instance of NWDP. Let (x,z) be an arc of G, and let r be its weight. We replace (x,z) by the path $(x,y_1,\ldots,y_{r-1},z)$ of length r where $y_1,\ldots,y_{r-1}$ are new vertices. We apply this procedure to every arc of G and obtain graph G'. It is not difficult to see that there is a 1-1 correspondence between the u,v-paths of weight ω in G and the u,v-paths of length ω in G'. Hence, $(G,\varrho,\omega,\nu,u,v) \in$ NWDP if and only if $(G',\omega,\nu,u,v) \in$ NDP$(2,2)$. ■

At first glance, the advantages of NWDP with respect to NDP$(2,2)$ are not obvious. However, since we want to study subsets of NDP$(2,2)$ to identify hardest instances, NWDP provides an easier description.

Example. Our introductory example dealt with the number of natural solutions of equations like $x_1 + x_2 + \cdots + x_r = k$. We can transform this question into an instance of NWDP. Construct a path on r vertices, and every vertex has a loop. All arcs have weight 1. The resulting graph is depicted in Figure 2. Then, the number of solutions of the given equation

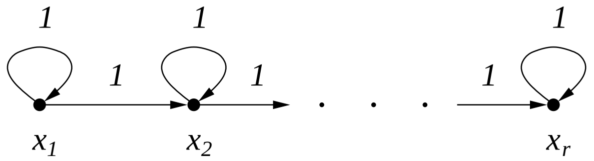

Figure 2. The weighted graph that corresponds to equation $x_1 + \cdots + x_r = k$.

is equal to the number of x_1, x_r-paths of weight $k + r - 1$ in the graph. To accept an input an algorithm must decide whether the queried number of paths of weight $k + r - 1$ is not greater than $\binom{r+k-1}{k-1}$. The question becomes more complicated, if we replace loop weights by numbers greater than 1. This corresponds to the question for the number of solutions of an equation like $a_1x_1 + \cdots + a_rx_r = k$ where $a_1, \ldots, a_r \geq 1$.

The discussed example indicates a possible restriction on the structure of input graphs. We say that the *skeleton* of a directed graph is the graph obtained by deleting all loops. So, we can restrict on graphs whose skeleton has a tree-like structure. More general, we can restrict on graphs whose skeleton is acyclic. Even for the case that the skeleton of the input graph is a path we do not have an algorithm improving the upper bound P. And it seems unlikely that this most restricted variant is already P-complete.

9. Conclusions. In this work, we introduced a number of problems all of which ask for the number of paths of given length between two specified vertices. We distinguished between directed and undirected graphs and determined the complexities of such problems also with respect to the representation system of the given natural numbers. We showed NL-completeness results as well as completeness results for $C_{=}L$ and PL. The exact complexity of our hardest problem, however, remained open. We could only show an interesting upper complexity bound. It does not seem impossible that the complexity of NDP$(2, 2)$ is captured by a complexity class that is properly between PL and P under usual assumptions.

We raised the question whether our notion for equality of paths in an undirected graph is really captured by our definition and proposed the notion of isomorphic paths. We proved a characterisation of isomorphic graphs, but did not give any further result for the corresponding number-of-paths problems. It is, however, the case that all hardness results also hold for the case of isomorphic paths, but we strongly believe that these lower bounds are not tight.

Acknowledgement. I thank Klaus W. Wagner and Elmar Böhler for interesting and helpful discussions.

Bibliography

[1] E. Allender, M. Ogihara, *Relationships among PL, #L, and the Determinant*, Informatique théorique et Applications 30, No. 1, pp. 1–21, 1996.

[2] E. Allender, K. Reinhardt, Sh. Zhou, *Isolation, Matching, and Counting Uniform and Nonuniform Upper Bounds*, Journal of Computer and System Sciences 59, pp. 164–181, 1999.

[3] R. Armoni, A. Ta-Shma, A. Wigderson, Sh. Zhou, *An $\mathcal{O}(log(n)^{4/3})$ Space Algorithm for (s, t) Connectivity in Undirected Graphs*, Journal of the ACM 47, No. 2, pp. 294–311, 2000.

[4] K.S. Booth, G.S. Lueker, *Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using PQ-Tree Algorithms*, Journal of Computer and System Sciences 13, pp. 335–379, 1976.

[5] A. Borodin, S. Cook, N. Pippenger, *Parallel Computation for Well-Endowed Rings and Space-Bounded Probabilistic Machines*, Information and Control 58, pp. 113–136,l 1983.

[6] A. Chiu, G. Davida, B. Litow, *Division in logspace-uniform NC^1*, Theoretical Informatics and Applications 35, pp. 259–275, 2001.

[7] St.A. Cook, *A Taxonomy of Problems with Fast Parallel Algorithms*, Information and Control 64, pp. 2–22, 1985.

[8] M.L. Fredman, J. Komlós, E. Szemerédi, *Storing a Sparse Table with $\mathcal{O}(1)$ Worst Case Access Time*, Journal of the Association for Computing Machinery 31, No. 3, pp. 438–544, 1984.

[9] W. Hesse, E. Allender, D.A. Mix Barrington, *Uniform constant-depth threshold circuits for division and iterated multiplication*, Journal of Computer and System Science 65, pp. 695–716, 2002.

[10] N. Immerman, *Nondeterministic space is closed under complementation*, SIAM Journal on Computing 17, pp. 935–938, 1988.

[11] H.R. Lewis, Ch.H. Papadimitriou, *Symmetric Space-Bounded Computation*, Theoretical Computer Science 19, pp. 161–187, 1982.

[12] M. Ogihara, *The PL Hierarchy Collapses*, SIAM Journal on Computing 27, No. 5, pp. 1430–1437, 1998.

[13] O. Reingold, *Undirected ST-Connectivity in Log-Space*, Electronic Colloquium on Computational Complexity, ECCC, report no. 94, 2004.

[14] W.L. Ruzzo, J. Simon, M. Tompa, *Space-Bounded Hierarchies and Probabilistic Computations*, Journal of Computer and System Sciences 28, pp. 216–230, 1984.

[15] W.J. Savitch, *Relationships Between Nondeterministic and Deterministic Tape Complexities*, Journal of Computer and System Sciences 4, pp. 177–192, 1970.

[16] R. Szelepcsényi, *The method of forced enumeration for nondeterministic automata*, Acta Informatica 26, pp. 279–284, 1988.

[17] H. Vollmer, *Introduction to Circuit Complexity*, Springer, 1999.

Corrigendum

(February 16th, 2006)

Corollary 18 is not correct as it is stated. It must be in the following form.

Corollary 35. *Let $G = (V, A)$ be a directed graph, and let $n =_{\mathrm{def}} |V|$. Let $u, v \in V$ be two vertices of G, and let ν and k such that $0 \leq 2\nu n^2 + n \leq k$. If $\nu \leq \#_G(\overrightarrow{uv}, k)$, then*

$$\nu \leq \sum_{x,z \in V} \#_G(\overrightarrow{ux}, 2\nu n^2) \cdot P_G(\overrightarrow{xz}, k - 2\nu n^2 - n) \cdot \#_G(\overrightarrow{zv}, n)\,.$$

Proof: We distinguish two cases.

(1) There are vertices x, z of G such that the following holds:

$$\begin{aligned} &\#_G(\overrightarrow{ux}, 2\nu n^2) \geq 1 \\ &\#_G(\overrightarrow{xz}, k - 2\nu n^2 - n) \geq 2 \\ &\#_G(\overrightarrow{zv}, n) \geq 1\,. \end{aligned}$$

Then, the prerequisites of Lemma 17 are fulfilled, and iterated application of the construction of Lemma 17 provides the claim. In this case, the inequality is strict.

(2) For every pair of vertices x, z of G the following holds: if $\#_G(\overrightarrow{ux}, 2\nu n^2) \geq 1$ and $\#_G(\overrightarrow{zv}, n) \geq 1$ then

$$\#_G(\overrightarrow{xz}, k - 2\nu n^2 - n) \leq 1\,.$$

Then, the claim immediately holds. In this case, equality even holds.

■

The proof of Theorem 20 must be adjusted in that way that, instead of guessing only one cardinality ν_w for every vertex w of G, two cardinalities, ν_w and ν'_w, have to be guessed. This is still possible, and the algorithm of Theorem 20 only has to take into account this slight modification.

Acknowledgement. I am deeply grateful to an anonymous referee who pointed out the insufficiency of Corollary 18.