Algorithmica (1997) 17: 209–223

Algorithmica
© 1997 Springer-Verlag New York Inc.

# Finding and Counting Given Length Cycles[1]

N. Alon,[2] R. Yuster,[2] and U. Zwick[2]

**Abstract.** We present an assortment of methods for finding and counting simple cycles of a given length in directed and undirected graphs. Most of the bounds obtained depend solely on the number of edges in the graph in question, and not on the number of vertices. The bounds obtained improve upon various previously known results.

**Key Words.** Graph algorithms, Cycles.

**1. Introduction.** The problem of deciding whether a given graph $G = (V, E)$ contains a simple cycle of length $k$ is among the most natural and easily stated algorithmic graph problems. If the cycle length $k$ is part of the input, then the problem is clearly NP-complete as it includes in particular the Hamiltonian cycle problem. For every fixed $k$, however, the problem can be solved in either $O(VE)$ time [11] or $O(V^{\omega} \log V)$ [2], where $\omega < 2.376$ is the exponent of matrix multiplication.

The main contribution of this paper is a collection of new bounds on the complexity of finding simple cycles of length exactly $k$, where $k \geq 3$ is a fixed integer, in a directed or an undirected graph $G = (V, E)$. These bounds are of the form $O(E^{\alpha_k})$ or of the form $O(E^{\beta_k} \cdot d(G)^{\gamma_k})$, where $d(G)$ is the *degeneracy* of the graph (see below). The bounds improve upon previously known bounds when the graph in question is relatively sparse or relatively degenerate.

We let $C_k$ stand for a simple cycle of length $k$. When considering directed graphs, a $C_k$ is assumed to be directed. We show that a $C_k$ in a directed or undirected graph $G = (V, E)$, if one exists, can be found in $O(E^{2-2/k})$ time, if $k$ is even, and in $O(E^{2-2/(k+1)})$ time, if $k$ is odd. For finding triangles ($C_3$'s), we get the slightly better bound of $O(E^{2\omega/(\omega+1)}) = O(E^{1.41})$, where $\omega < 2.376$ is the exponent of matrix multiplication.

Even cycles in undirected graphs can be found even faster. A $C_{4k-2}$ in an undirected graph $G = (V, E)$, if one exists, can be found in $O(E^{2-(1/2k)(1+1/k)})$ time. A $C_{4k}$, if one exists, can be found in $O(E^{2-(1/k-1/(2k+1))})$ time. In particular, we can find an undirected $C_4$ in $O(E^{4/3})$ time and an undirected $C_6$ in $O(E^{13/8})$ time.

The *degeneracy* $d(G)$ of an undirected graph $G = (V, E)$ is the smallest number $d$ for which there exists an acyclic orientation of $G$ in which all the out-degrees are at most $d$. The degeneracy $d(G)$ of a graph $G$ is linearly related to the *arboricity* $a(G)$ of the graph, i.e., $a(G) = \Theta(d(G))$, where $a(G)$ is the minimal number of forests needed

[1] This work was supported in part by The Basic Research Foundation administrated by The Israel Academy of Sciences and Humanities.

[2] School of Mathematical Sciences, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel. {noga,raphy,zwick}@math.tau.ac.il.

Received October 18, 1994; revised May 5, 1995. Communicated by N. Megiddo.

**Table 1.** Finding small cycles in directed graphs—some of the new results.

| Cycle | Complexity | Cycle | Complexity |
|---|---|---|---|
| $C_3$ | $E^{1.41}$, $E \cdot d(G)$ | $C_7$ | $E^{1.75}$, $E^{3/2} \cdot d(G)$ |
| $C_4$ | $E^{1.5}$, $E \cdot d(G)$ | $C_8$ | $E^{1.75}$, $E^{3/2} \cdot d(G)$ |
| $C_5$ | $E^{1.67}$, $E \cdot d(G)^2$ | $C_9$ | $E^{1.8}$, $E^{3/2} \cdot d(G)^{3/2}$ |
| $C_6$ | $E^{1.67}$, $E^{3/2} \cdot d(G)^{1/2}$ | $C_{10}$ | $E^{1.8}$, $E^{5/3} \cdot d(G)^{2/3}$ |

to cover all the edges of $G$. The degeneracy of a directed graph $G = (V, E)$ is defined to be the degeneracy of the undirected version of $G$. The degeneracy of a graph is an important parameter of the graph that appears in many combinatorial results. It is easy to see that for any graph $G = (V, E)$ we have $d(G) \le 2E^{1/2}$. For graphs with relatively low degeneracy we can improve upon the previously stated results. A $C_{4k}$ in a directed or undirected graph $G = (V, E)$ that contains one can be found in $O(E^{2-1/k} \cdot d(G))$ time. A $C_{4k+1}$, if one exists, can be found in $O(E^{2-1/k} \cdot d(G)^{1+1/k})$ time. Similar results are obtained for finding $C_{4k-2}$'s and $C_{4k-1}$'s. In particular, $C_3$'s and $C_4$'s can be found in $O(E \cdot d(G))$ time and $C_5$'s in $O(E \cdot d(G)^2)$ time. Some of the results mentioned are summarized in Tables 1 and 2.

As any planar graph has a vertex whose degree is at most 5, the degeneracy of any planar graph is at most 5. As a consequence of the above bounds we get, in particular, that $C_3$'s, $C_4$'s, and $C_5$'s in planar graphs can be found in $O(V)$ time. This in fact holds not only for planar graphs but for any nontrivial *minor-closed* family of graphs.

Another contribution of this paper is an $O(V^{\omega})$ algorithm for *counting* the number of $C_k$'s, for $k \le 7$, in a graph $G = (V, E)$.

A preliminary version of this work appeared in [1].

**2. Comparison with Previous Works.** Monien [11] obtained, for any fixed $k \ge 3$, an $O(VE)$ algorithm for finding $C_k$'s in a directed or undirected graph $G = (V, E)$. In a previous work [2] we showed, using the *color-coding* method, that a $C_k$, for any fixed $k \ge 3$, if one exists, can also be found in $O(V^{\omega})$ expected time or in $O(V^{\omega} \log V)$ worst-case time, where $\omega < 2.376$ is the exponent of matrix multiplication.

Our new $O(E^{2-2/k})$ algorithm is better than both the $O(VE)$ and the $O(V^{\omega})$ algorithms when the input graph $G = (V, E)$ is sufficiently sparse. It is interesting to note that, for $k \le 6$, Monien's $O(VE)$ bound is superseded by either the $O(V^{\omega})$ algorithm, when the graph is dense, or by the $O(E^{2-1/\lceil k/2 \rceil})$ algorithm, when the graph is sparse.

**Table 2.** Finding small cycles in undirected graphs—some of the new results.

| Cycle | Complexity | Cycle | Complexity |
|---|---|---|---|
| $C_4$ | $E^{1.34}$ | $C_8$ | $E^{1.7}$ |
| $C_6$ | $E^{1.63}$ | $C_{10}$ | $E^{1.78}$ |

For every $k \geq 7$, each one of the four bounds (including the bound that involves the degeneracy) beats the others on an appropriate family of graphs.

In a previous work [16] we have also shown that cycles of an *even* length in *undirected* graphs can be found even faster. Namely, for any even $k \geq 4$, if an undirected graph $G = (V, E)$ contains a $C_k$, then such a $C_k$ can be found in $O(V^2)$ time. Our $O(E^{2-(1/2k)(1+1/k)})$ bound for $C_{4k-2}$ and $O(E^{2-(1/k-1/(2k+1))})$ bound for $C_{4k}$ are again better when the graph is sparse enough.

Itai and Rodeh [8] showed that a *triangle* (a $C_3$) in a graph $G = (V, E)$ that contains one can be found in $O(V^\omega)$ or $O(E^{3/2})$ time. We improve their second result and show that the same can be done, in directed or undirected graphs, in $O(E^{2\omega/(\omega+1)}) = O(E^{1.41})$ time.

Chiba and Nishizeki [6] showed that triangles ($C_3$'s) and quadrilaterals ($C_4$'s) in graphs that contain them can be found in $O(E \cdot d(G))$ time. As $d(G) = O(E^{1/2})$ for any graph $G$, this extends the result of Itai and Rodeh. We extend the result of Chiba and Nishizeki and show that $C_{4k-1}$'s and $C_{4k}$'s can be found in $O(E^{2-1/k} \cdot d(G))$ time. We also show that $C_{4k+1}$'s can be found in $O(E^{2-1/k} \cdot d(G)^{1+1/k})$ time. This gives, in particular, an $O(E \cdot d(G)^2)$ algorithm for finding pentagons ($C_5$'s). Our results apply to both directed and undirected graphs.

Itai and Rodeh [8] and also Papadimitriou and Yannakakis [13] showed that $C_3$'s in planar graphs can be found in $O(V)$ time. Chiba and Nishizeki [6] showed that $C_3$'s as well as $C_4$'s in planar graphs can be found in $O(V)$ time. Richards [14] showed that $C_5$'s and $C_6$'s in planar graphs can be found in $O(V \log V)$ time. We improve upon the result of Richards and show that $C_5$'s in planar graphs can be found in $O(V)$ time. In a previous work [2] we showed, using color-coding, that, for any $k \geq 3$, a $C_k$ in a planar graph, if one exists, can be found in either $O(V)$ *expected* time or $O(V \log V)$ worst-case time.

The fact that the number of triangles in a graph can be counted in $O(V^\omega)$ time is trivial. In [2] we showed, using color-coding, that, for any $k \geq 3$, a $C_k$, if one exists, can be found in either $O(V^\omega)$ *expected* time or in $O(V^\omega \log V)$ worst-case time. Here we show that for any $k \leq 7$ the number of $C_k$'s in a graph can be counted in $O(V^\omega)$ time. The counting method used here yields, in particular, a way of finding $C_k$'s for $k \leq 7$, in $O(V^\omega)$ worst-case time.

Sundaram and Skiena [15] have recently presented some more fixed-subgraph isomorphism algorithms. The results presented here, and in [2] and [16], improve some of their results.

Eppstein [7] has recently shown that the fixed-subgraph isomorphism problem for *planar* graphs, i.e., given a fixed graph $H$ and a planar graph $G = (V, E)$, find a subgraph of $G$ isomorphic to $H$, can be solved, for every fixed $H$, in $O(V)$ time.

## 3. Finding Cycles in Sparse Graphs.

Monien [11] obtained his $O(VE)$ algorithm by the use of *representative collections*. Such collections are also used by our algorithms. In what follows, a $p$-set is a set of size $p$.

DEFINITION 3.1 [11]. Let $\mathcal{F}$ be a collection of $p$-sets. A subcollection $\hat{\mathcal{F}} \subseteq \mathcal{F}$ is $q$-representative for $\mathcal{F}$ if, for every $q$-set $B$, there exists a set $A \in \mathcal{F}$ such that $A \cap B = \emptyset$ if and only if there exists a set $A \in \hat{\mathcal{F}}$ with this property.

It follows from a combinatorial lemma of Bollobás [3] that any collection $\mathcal{F}$ of $p$-sets, no matter how large, has a $q$-representative subcollection of size at most $\binom{p+q}{p}$. Monien [11] describes an $O(pq \cdot \sum_{i=0}^{q} p^i \cdot |\mathcal{F}|)$-time algorithm for finding a $q$-representative subcollection of $\mathcal{F}$ whose size is at most $\sum_{i=0}^{q} p^i$. Relying on Monien's result we obtain the following lemma:

LEMMA 3.2. *Let $\mathcal{F}$ be a collection of $p$-sets and let $\mathcal{G}$ be a collection of $q$-sets. Consider $p$ and $q$ to be fixed. In $O(|\mathcal{F}| + |\mathcal{G}|)$ time, we can either find two sets $A \in \mathcal{F}$ and $B \in \mathcal{G}$ such that $A \cap B = \emptyset$ or decide that no two such sets exist.*

PROOF. We use Monien's algorithm to find a $q$-representative subcollection $\hat{\mathcal{F}}$ of $\mathcal{F}$ whose size is at most $\sum_{i=0}^{q} p^i$ and a $p$-representative subcollection $\hat{\mathcal{G}}$ of $\mathcal{G}$ whose size is at most $\sum_{i=0}^{p} q^i$. This takes only $O(|\mathcal{F}| + |\mathcal{G}|)$ time (as $p$ and $q$ are constants).

It is easy to see that if there exist $A \in \mathcal{F}$ and $B \in \mathcal{G}$ such that $A \cap B = \emptyset$, then there also exist $A' \in \hat{\mathcal{F}}$ and $B' \in \hat{\mathcal{G}}$ such that $A' \cap B' = \emptyset$. To see this note that if $A \cap B = \emptyset$, then, by the definition of $q$-representatives, there must exist a set $A' \in \hat{\mathcal{F}}$ such that $A' \cap B = \emptyset$ and then there must exist a set $B' \in \hat{\mathcal{G}}$ such that $A' \cap B' = \emptyset$ as required.

After finding the representative collections $\hat{\mathcal{F}}$ and $\hat{\mathcal{G}}$ it is therefore enough to check whether they contain two disjoint sets. This can be easily done in constant time (as $p$ and $q$ are constants). □

We also need the following lemma that follows immediately from the work of Monien [11].

LEMMA 3.3 [11]. *Let $G = (V, E)$ be a directed or undirected graph, let $v \in V$, and let $k \geq 3$. A $C_k$ that passes through $v$, if one exists, can be found in $O(E)$ time.*

We are finally able to present our improved algorithm.

THEOREM 3.4. *Deciding whether a directed or undirected graph $G = (V, E)$ contains simple cycles of length exactly $2k - 1$ and of length exactly $2k$, and finding such cycles if it does, can be done in $O(E^{2-1/k})$ time.*

PROOF. We describe an $O(E^{2-1/k})$-time algorithm for finding a $C_{2k}$ in a directed graph $G = (V, E)$. The details of all the other cases are similar. Let $\Delta = E^{1/k}$. A vertex in $G$ whose degree is at least $\Delta$ is said to be of *high degree*. The graph $G = (V, E)$ contains at most $2E/\Delta = O(E^{1-1/k})$ high-degree vertices. We check, using Monien's algorithm (Lemma 3.3), whether any of these high-degree vertices lies on a simple cycle of length $2k$. For each vertex this costs $O(E)$ operations and the total cost is $O(E^2/\Delta) = O(E^{2-1/k})$. If one of these vertices does lie on a cycle of length $2k$ we are done. Otherwise, we remove all the high-degree vertices and all the edges adjacent to them from $G$ and obtain a subgraph $G'$ that contains a $C_{2k}$ if and only if $G$ does. The maximum degree of $G'$ is at most $\Delta = E^{1/k}$ and there are therefore at most $E \cdot \Delta^{k-1} = E^{2-1/k}$ simple directed paths of length $k$ in $G'$. We can find all these simple paths in $O(E^{2-1/k})$

time. We divide these paths into groups according to their endpoints. This can be done using radix sort in $O(E^{2-1/k})$ time and space. We get a list of all the pairs of vertices connected by simple directed paths of length exactly $k$. For each such pair $u$, $v$, we get a collection $\mathcal{F}_{u,v}$ of $(k-1)$-sets. Each $(k-1)$-set in $\mathcal{F}_{u,v}$ corresponds to the $k-1$ intermediate vertices that appear on simple directed paths of length $k$ from $u$ to $v$. For each pair $u$, $v$ that appears on the list, we check whether there exist two directed paths of length $k$, one from $u$ to $v$ and the other from $v$ to $u$, that meet only at their endpoints. Such two paths exist if there exist $A \in \mathcal{F}_{u,v}$ and $B \in F_{v,u}$ such that $A \cap B = \emptyset$. This can be checked, as shown in Lemma 3.2, in $O(|\mathcal{F}_{u,v}| + |\mathcal{F}_{v,u}|)$ time. As the sum of the sizes of all these collections is $O(E^{2-1/k})$, the total complexity is again $O(E^{2-1/k})$. This completes the proof. □

In the case of triangles we can get a better result by using fast matrix multiplication.

THEOREM 3.5. *Deciding whether a directed or an undirected graph $G = (V, E)$ contains a triangle, and finding one if it does, can be done is $O(E^{2\omega/(\omega+1)}) = O(E^{1.41})$ time.*

PROOF. Let $\Delta = E^{(\omega-1)/(\omega+1)}$. A vertex is said to be of *high degree* if its degree is more than $\Delta$ and of *low degree* otherwise. Consider all directed paths of length 2 in $G$ whose intermediate vertex is of low degree. There are at most $E \cdot \Delta$ such paths and they can be found in $O(E \cdot \Delta)$ time. For each such path, check whether its endpoints are connected by an edge in the appropriate direction. If no triangle is found in this way, then any triangle in $G$ must be composed of three high-degree vertices. As there are at most $2E/\Delta$ high-degree vertices, we can check whether there exists such a triangle using matrix multiplication in $O((E/\Delta)^{\omega})$ time. The total complexity of the algorithm is therefore

$$O\left(E \cdot \Delta + \left(\frac{E}{\Delta}\right)^{\omega}\right) = O(E^{2\omega/(\omega+1)}).$$

This completes the proof. □

We have not been able to utilize matrix multiplication to improve upon the result of Theorem 3.4 for $k \geq 4$. This constitutes an interesting open problem.

## 4. Finding Cycles in Graphs with Low Degeneracy.

An undirected graph $G = (V, E)$ is *$d$-degenerate* (see p. 222 of [4]) if there exists an acyclic orientation of it in which $d_{\text{out}}(v) \leq d$ for every $v \in V$. The smallest $d$ for which $G$ is $d$-degenerate is called the *degeneracy* or the *max-min degree* of $G$ and is denoted by $d(G)$. It can be easily seen (see again [4]) that $d(G)$ is the maximum of the minimum degrees taken over all the subgraphs of $G$. The degeneracy $d(G)$ of a graph $G$ is linearly related to the *arboricity* $a(G)$ of the graph, i.e., $a(G) = \Theta(d(G))$, where $a(G)$ is the minimal number of forests needed to cover all the edges of $G$. The degeneracy of a directed graph $G = (V, E)$ is defined to be the degeneracy of the undirected version of $G$. It is easy to see that the degeneracy of any planar graph is at most 5. Clearly, if $G$ is $d$-degenerate, then

$|E| \leq d{\cdot}|V|$. The following simple lemma, whose proof is omitted, is part of the folklore (see, e.g., [10]).

LEMMA 4.1. *Let $G = (V, E)$ be a connected undirected graph $G = (V, E)$. An acyclic orientation of $G$ such that for every $v \in V$ we have $d_{\mathrm{out}}(v) \leq d(G)$ can be found in $O(E)$ time.*

The main result of this section is the following theorem:

THEOREM 4.2. *Let $G = (V, E)$ be a directed or an undirected graph.*

(i) *Deciding whether $G$ contains a simple cycle of length exactly $4k - 2$, and finding such a cycle if it does, can be done in $O(E^{2-1/k}{\cdot}d(G)^{1-1/k})$ time.*
(ii) *Deciding whether $G$ contains simple cycles of length exactly $4k - 1$ and of length exactly $4k$, and finding such cycles if it does, can be done in $O(E^{2-1/k}{\cdot}d(G))$ time.*
(iii) *Deciding whether $G$ contains a simple cycle of length exactly $4k + 1$, and finding such a cycle if it does, can be done in $O(E^{2-1/k}{\cdot}d(G)^{1+1/k})$ time.*

PROOF. We show how to find a $C_{4k+1}$ in a directed graph $G = (V, E)$, if one exists, in $O(E^{2-1/k} \cdot d(G)^{1+1/k})$ time. The proofs of the other claims are easier. If $d(G) \geq E^{1/(2k+1)}$, we can use the algorithm of Theorem 3.4 whose complexity is $O(E^{2-1/(2k+1)}) \leq O(E^{2-1/k}{\cdot}d(G)^{1+1/k})$. Assume therefore that $d(G) \leq E^{1/(2k+1)}$.

Let $\Delta = E^{1/k}/d(G)^{1+1/k}$. As $d(G) \leq E^{1/(2k+1)}$, we have that $d(G) \leq \Delta$. A vertex is said to be of *high degree* if its degree is more than $\Delta$ and of *low degree* otherwise. As in the proof of Theorem 3.4, we can check in $O(E^2/\Delta)$ time whether any of the high-degree vertices lies on a $C_{4k+1}$. If none of them lies on a $C_{4k+1}$, we can remove all the high-degree vertices along with the edges adjacent to them from $G$ and obtain a graph whose maximal degree is at most $\Delta$. The degeneracy of a graph can only decrease when vertices and edges are deleted and $d(G)$ is therefore an upper bound on the degeneracy of the graph obtained.

Suppose therefore that $G$ is a graph with maximal degree $\Delta$ and degeneracy $d(G)$. To find a $C_{4k+1}$ in $G$, it is enough to find all directed simple paths of length $2k$ and $2k + 1$ in $G$ and then check, using the algorithm described in the proof of Lemma 3.2, whether there exist a path of length $2k$ and a path of length $2k + 1$ that meet only at their endpoints.

In $O(E)$ time we can get an acyclically oriented version $G'$ of $G$ in which the out-degree of each vertex is at most $d(G)$. The orientations of the edges in $G$ and $G'$ may be completely different.

The number of paths, not necessarily directed, of length $2k + 1$ in $G$, is at most

$$2{\cdot}2E{\cdot}\sum_{i=0}^{k}\binom{2k}{i}\Delta^{i}d(G)^{2k-i} = O(E\Delta^{k}d(G)^{k}).$$

To see this, consider the orientations, in $G'$, of the edges on a $(2k + 1)$-path in $G$. In at least one direction, at most $k$ of the edges are counterdirected. The number of paths of length $2k + 1$ in which exactly $i$ among the last $2k$ edges are counterdirected is at most $2E\cdot\binom{2k}{i}\Delta^i d(G)^{2k-i}$. The binomial coefficient $\binom{2k}{i}$ stands for the possible choices for the position of the counterdirected edges in the path. Similarly, the number of paths of length $2k$ in $G$ is $O(E\Delta^k d(G)^{k-1})$.

We can lower the number of paths of length $2k+1$ and $2k$ we have to consider by utilizing the fact that a $C_{4k+1}$ can be broken into two paths of length $2k+1$ and $2k$ in many different ways. In particular, let $C$ be a directed $C_{4k+1}$ in $G$ and consider the orientations of its edges in $G'$. As $4k+1$ is odd and as $G'$ is acyclic, $C$ must contain three consecutive edges $e_{2k}$, $e_{2k+1}$, and $e_{2k+2}$, the first two of which have the same orientation while the third one has an opposite orientation. It is therefore enough to consider all $(2k+1)$-paths that start with at least two backward oriented edges and all $2k$-paths that start with at least one backward oriented edge. The orientations referred to here are in $G'$.

The number of paths of length $2k+1$ in $G$ whose first two edges are backward oriented in $G'$ is $O(E\Delta^{k-1}d(G)^{k+1})$. To see this, note that any such path is composed of a directed path $\{e_{2k}, e_{2k+1}\}$ of length 2, attached to an arbitrarily oriented path $\{e_1,\dots,e_{2k-1}\}$ of length $2k-1$. The number of paths of length $2k-1$ is, as shown earlier, at most $O(E\Delta^{k-1}d(G)^{k-1})$ and the number of directed path of length 2 with a specified starting point is at most $d(G)^2$. Similarly, there are at most $O(E\Delta^{k-1}d(G)^k)$ $2k$-paths that start with a backward oriented edge.

It should be clear from the above discussion that all the required $(2k+1)$- and $2k$-paths, whose total number is $O(E\Delta^{k-1}d(G)^{k+1})$, can be found in $O(E\Delta^{k-1}d(G)^{k+1})$ time. Paths which are not properly directed, in $G$, are thrown away. Properly directed paths are sorted, using radix sort, according to their endpoints. Using Lemma 3.2 we then check whether there exist a directed $(2k+1)$-path and a directed $2k$-path that close a directed simple cycle. All these operations can again be performed in $O(E\Delta^{k-1}d(G)^{k+1})$ time.

Recalling that $\Delta = E^{1/k}/d(G)^{1+1/k}$, we get that the overall complexity of the algorithm is

$$O\left(\frac{E^2}{\Delta}+E\cdot\Delta^{k-1}d(G)^{k+1}\right)=O(E^{2-1/k}d(G)^{1+1/k}).$$

This completes the proof of the theorem. □

As an immediate corollary we get:

COROLLARY 4.3. *If a directed or undirected planar graph $G=(V,E)$ contains a pentagon (a $C_5$), then such a pentagon can be found in $O(V)$ worst-case time.*

By combining the ideas of this section, the $O(E^{2\omega/(\omega+1)})$ algorithm of Theorem 3.5, and the color-coding method [2] we can also obtain the following result.

THEOREM 4.4. *Let $G=(V,E)$ be a directed or undirected graph. A $C_6$ in $G$, if one exists, can be found in either $O((E\cdot d(G))^{2\omega/(\omega+1)})=O((E\cdot d(G))^{1.41})$ expected time or $O((E\cdot d(G))^{2\omega/(\omega+1)}\cdot\log V)=O((E\cdot d(G))^{1.41}\log V)$ worst-case time.*

PROOF. We show how to find a $C_6$ in an undirected graph $G=(V,E)$, if one exists,

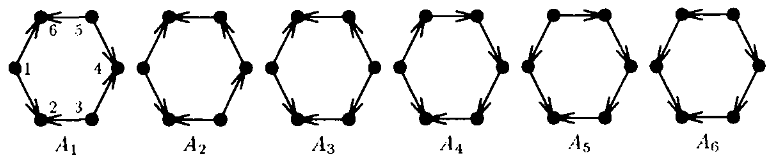

**Fig. 1.** The possible orientations of $C_6$ in $G'$.

in $O((E \cdot d(G))^{2\omega/(\omega+1)})$ expected time, or $O((E \cdot d(G))^{2\omega/(\omega+1)} \log V)$ worst-case time. The proof of the directed case is similar.

In $O(E)$ time we can get an acyclically oriented version $G' = (V, E')$ of $G$ in which the out-degree of each vertex is at most $d(G)$. Suppose that $G$ contains a $C_6$. The six possible nonisomorphic orientations of this $C_6$ in $G'$ are shown in Figure 1. We refer to these orientations as $A_1, \ldots, A_6$. Our algorithm checks, for each $1 \le i \le 6$, whether $G'$ contains an $A_i$ and if so finds one.

We show how to find an $A_1$ in $G'$, if one exists. The other cases are similar, and in fact easier. As in [2], we color the vertices of $G'$ randomly using six colors (i.e., every vertex receives a number between 1 and 6, all numbers equally likely). Let $c(v)$ denote the color of vertex $v$. Let $A$ be a specific copy of an $A_1$ in $G'$. We say that $A$ is *well-colored* if its vertices are consecutively colored by 1 through 6, and the color 1 is assigned to one of the three vertices having only outgoing edges in $A$. (By "consecutively colored" we mean that each $v \in A$ with $c(v) < 6$ has a neighbor $u \in A$ with $c(u) = c(v) + 1$). The probability that $A$ is well-colored is $6/6^6$. We now show how to detect a well-colored copy of an $A_1$ deterministically, if one exists.

Create a new undirected graph $G^* = (V^*, E^*)$ defined as follows:

$$V^* = \{v \in V \colon c(v) \in \{2, 4, 6\}\}$$

$$\begin{aligned} E^* &= \{(u, v) \colon c(u) = 6,\ c(v) = 2,\ (\exists\, w \in V)\ (c(w) = 1,\ (w, u), (w, v) \in E')\} \\ &\cup \{(u, v) \colon c(u) = 2,\ c(v) = 4,\ (\exists\, w \in V)\ (c(w) = 3,\ (w, u), (w, v) \in E')\} \\ &\cup \{(u, v) \colon c(u) = 4,\ c(v) = 6,\ (\exists\, w \in V)\ (c(w) = 5,\ (w, u), (w, v) \in E')\}. \end{aligned}$$

Clearly, $V^* \le V$. To create $G^*$, we examine each edge $(w, u) \in E'$ with $c(w)$ odd. Suppose $c(w) = 1$ and $c(u) = 6$. We create edges in $G^*$ between $u$ and all vertices $v$ such that $(w, v) \in E'$ and $c(v) = 2$. There are at most $d(G) - 1$ such vertices. We therefore have $E^* < E \cdot d(G)$ and $G^*$ can be constructed in $O(E \cdot d(G))$ time from $G'$. Clearly, there exists an undirected triangle in $G^*$ iff there exists a well-colored $A_1$ in $G'$. We can detect such a triangle in $G^*$ in $O((E^*)^{2\omega/(\omega+1)}) = O((Ed(G))^{2\omega/(\omega+1)})$ time using the algorithm of Theorem 3.5. If such a triangle is not found, we repeat the whole process using a new random coloring. If $G'$ contains an $A_1$, then such an $A_1$ will be found after an expected number of $6^5 = 7776$ attempts.

We have thus shown how to detect an $A_1$ in $G'$, if one exists, in $O((Ed(G))^{2\omega/(\omega+1)}) = O((E \cdot d(G))^{2\omega/(\omega+1)})$ expected time. As shown in [2], such a coloring scheme can be derandomized at the price of an $O(\log V)$ factor. $\square$

**5. Finding Cycles in Sparse Undirected Graphs.** To obtain the results of this section we rely on the following combinatorial lemma of Bondy and Simonovits [5].

LEMMA 5.1 [5]. *Let $G = (V, E)$ be an undirected graph. If $|E| \geq 100k \cdot |V|^{1+1/k}$, then $G$ contains a $C_{2\ell}$ for every integer $\ell \in [k, n^{1/k}]$.*

By combining the algorithm described in the proof of Theorem 4.2 with an algorithm given in [16] we obtain the following theorem.

THEOREM 5.2. *Let $G = (V, E)$ be an undirected graph.*

(i) *A $C_{4k-2}$ in $G$, if one exists, can be found in $O(E^{2-(1/2k)(1+1/k)})$ time.*
(ii) *A $C_{4k}$ in $G$, if one exists, can be found in $O(E^{2-(1/k-1/(2k+1))})$ time.*

PROOF. We prove the second claim. The proof of the first claim is similar. Let $d = 200k \cdot E^{1/(2k+1)}$. If $d(G) \geq d$, then, by the definition of degeneracy, there is a subgraph $G' = (V', E')$ of $G$ in which the minimal degree is at least $d$. Such a subgraph can be easily found in $O(E)$ time (see, e.g., [10]). Clearly, $E' \geq dV'/2 \geq 100k \cdot V' \cdot E'^{1/(2k+1)}$ and therefore $E' \geq (100k \cdot V')^{1+1/2k} \geq 100k \cdot (V')^{1+1/2k}$. By Lemma 5.1 we get that $G'$ contains a $C_{4k}$ and such a $C_{4k}$ can be found in $O(V'^2) = O(E^{2-2/(2k+1)})$ time using the algorithm given in [16]. If, on the other hand, $d(G) \leq d$, then a $C_{4k}$ in $G$, if one exists, can be found in $O(E^{2-1/k} \cdot d) = O(E^{2-(1/k-1/(2k+1))})$ time using the algorithm of Theorem 4.2. It is easy to check that $E^{2-2/(2k+1)} \leq E^{2-(1/k-1/(2k+1))}$ with equality holding only if $k = 1$. In both cases the complexity is therefore $O(E^{2-(1/k-1/(2k+1))})$ as required. □

COROLLARY 5.3. *Let $G = (V, E)$ be an undirected graph.*

(i) *A quadrilateral ($C_4$) in $G$, if one exists, can be found in $O(E^{4/3})$ time.*
(ii) *A hexagon ($C_6$) in $G$, if one exists, can be found in $O(E^{13/8})$ time.*

**6. Counting Small Cycles.** Let $G = (V, E)$ be a simple undirected graph and let $A = A_G$ be the adjacency matrix of $G$. Assume for simplicity that $V = \{1, \ldots, n\}$. Denote by $a_{ij}^{(k)} = (A^k)_{ij}$ the elements of the $k$th power of $A$. The trace $\mathrm{tr}(A^k)$ of $A^k$, which is the sum of the entries along the diagonal of $A^k$, gives us the number of closed walks of length $k$ in $G$. If we could also compute the number of *nonsimple* closed walks of length $k$ in $G$ we would easily obtain the number of *simple* closed paths of length $k$ in $G$. This number is just $2k$ times the number of $C_k$'s in $G$.

Before describing a way of counting the number of nonsimple closed walks of length $k$, where $k \leq 7$, in a graph $G$ in $O(V^\omega)$ time, we need the following definitions:

DEFINITION 6.1. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two simple graphs. A mapping $f\colon V_1 \cup E_1 \to V_2 \cup E_2$ is a *homomorphism* if for every $v \in V_1$ we have $f(v) \in V_2$ and for every $e = (u, v) \in E_1$ we have $f(e) = (f(u), f(v)) \in E_2$. If $f$ is onto $V_2 \cup E_2$, we say that $G_2$ is a *homomorphic image* of $G_1$.

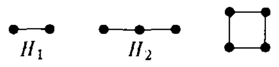

**Fig. 2.** The 4-cyclic graphs.

DEFINITION 6.2. A graph $H = (V_H, E_H)$ is said to be *k-cyclic*, for $k \geq 3$, if it is a homomorphic image of the cycle $C_k$. The number of different homomorphisms from $C_k$ to $H$ is denoted by $c_k(H)$. Clearly, $H$ is $k$-cyclic if and only if $c_k(H) > 0$.

It is easy to check, for example, that $C_3$ is $k$-cyclic for every $k \geq 3$ except $k = 4$. It is also not difficult to check that $c_3(C_3) = 6$ (and more generally $c_k(C_k) = 2k$ for every $k \geq 3$) and that $c_5(C_3) = 30$. The only 3-cyclic graph is $C_3$ itself. The $k$-cyclic graphs, for $4 \leq k \leq 7$, are given in Figures 2–5.

Let $n_G(H)$ denote the number of subgraphs of $G$ isomorphic to $H$. Clearly, the total number of closed walks of length $k$ in $G$ is

$$\mathrm{tr}(A^k) = \sum_H c_k(H) n_G(H).$$

If $c_k(H) > 0$, then $H$ is connected and has at most $k$ edges. Also, $H$ cannot be a tree on $k + 1$ vertices as each edge leading to a leaf must be the image of at least two edges in $C_k$. Hence, $|V_H| \leq k$ and in fact, $|V_H| < k$ unless $H = C_k$. We therefore obtain, for an undirected graph $G = (V, E)$:

$$(1) \qquad n_G(C_k) = \frac{1}{2k} \cdot \left[ \mathrm{tr}(A^k) - \sum_{|V_H|<k} c_k(H) n_G(H) \right].$$

A very similar formula can be obtained for directed graphs. We show how to compute $n_G(H)$, for all $k$-cyclic graphs $H$ with $3 \leq k \leq 7$, in $O(V^\omega)$ time. Hence, we obtain the following theorem.

THEOREM 6.3. *The number of $C_k$'s, for $3 \leq k \leq 7$ in an undirected (or directed) graph $G = (V, E)$, can be found in $O(V^\omega)$ time.*

PROOF. We consider the undirected case. The directed case is similar and, in fact, slightly simpler (as there are less $k$-cyclic graphs). Clearly, the traces $\mathrm{tr}(A^k)$, for $3 \leq k \leq 7$, can be computed in $O(V^\omega)$ time using fast matrix multiplication. It remains to show how to find $n_G(H)$ for all $k$-cyclic graphs $H$, where $3 \leq k \leq 7$, excluding the cycles $C_3, \ldots, C_7$ themselves, in $O(V^\omega)$ time.

The $k$-cyclic graphs shown in Figures 2–5, which are not simple cycles, are denoted by $H_1, \ldots, H_{15}$ (they are ordered according to the number of edges they contain).

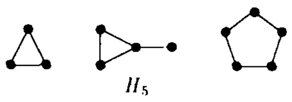

**Fig. 3.** The 5-cyclic graphs.

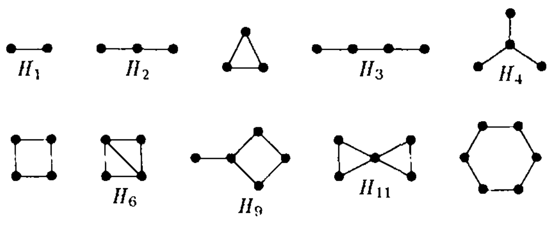

**Fig. 4.** The 6-cyclic graphs.

The following list shows how to obtain $n_G(H_i)$, for $1 \le 1 \le 15$, and $n_G(C_k)$, for $3 \le k \le 7$. In all cases the formulae reference at most $O(V^2)$ values of $a_{ij}^{(p)}$ for some $1 \le p \le k$ and can hence be computed in $O(V^\omega)$ time. We let $d_i = a_{ii}^{(2)}$ denote the degree of vertex $i$.

1.
$$n_G(C_3) = \tfrac{1}{6}\cdot \mathrm{tr}(A^3).$$

2.
$$n_G(H_1) = |E| = \sum_{1\le i<j\le n} a_{ij}^{(1)}.$$

3.
$$n_G(H_2) = \sum_{i=1}^{n} \binom{d_i}{2}.$$

4.
$$n_G(C_4) = \tfrac{1}{8}\cdot[\mathrm{tr}(A^4) - 4n_G(H_2) - 2n_G(H_1)].$$

5.
$$n_G(H_3) = \sum_{(i,j)\in E} (d_i - 1)(d_j - 1) - 3n_G(C_3).$$

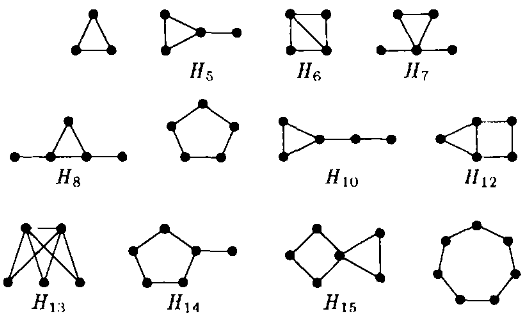

**Fig. 5.** The 7-cyclic graphs.

6.

$$n_G(H_4) = \sum_{i=1}^{n} \binom{d_i}{3}.$$

7.

$$n_G(H_5) = \frac{1}{2} \cdot \left[ \sum_{i=1}^{n} a_{ii}^{(3)} (d_i - 2) \right].$$

Note that $a_{ii}^{(3)}$ is twice the number of triangles that pass through vertex $i$.

8.

$$n_G(C_5) = \frac{1}{10} \cdot [\mathrm{tr}(A^5) - 10 n_G(H_5) - 30 n_G(C_3)].$$

9.

$$n_G(H_6) = \sum_{(i,j)\in E} \binom{a_{ij}^{(2)}}{2}.$$

Note that $a_{ij}^{(2)}$ is the number of common neighbors of $i$ and $j$, which is also the number of paths of length 2 between $i$ and $j$.

10.

$$n_G(H_7) = \frac{1}{2} \cdot \left[ \sum_{i=1}^{n} a_{ii}^{(3)} \binom{d_i - 2}{2} \right].$$

11.

$$n_G(H_8) = \sum_{(i,j)\in E} a_{ij}^{(2)} (d_i - 2)(d_j - 2) - 2 n_G(H_6).$$

Note that we must subtract $2n_G(H_6)$ to avoid the case in which the two degree-one vertices of $H_8$ are, actually, the same vertex.

12.

$$n_G(H_9) = \sum_{i=1}^{n} (d_i - 2) \sum_{j\neq i} \binom{a_{ij}^{(2)}}{2}.$$

Note that $\sum_{j\neq i} \binom{a_{ij}^{(2)}}{2}$ is exactly the number of quadrilaterals in which $i$ participates.

13.

$$n_G(H_{10}) = \sum_{i=1}^{n} \left(\tfrac{1}{2} a_{ii}^{(3)}\right) \left( \sum_{j\neq i} a_{ij}^{(2)} \right) - 6 n_G(C_3) - 2 n_G(H_5) - 4 n_G(H_6).$$

Note that $(\frac{1}{2} a_{ii}^{(3)})(\sum_{j\neq i} a_{ij}^{(2)})$ is simply the number of triangles through $i$ times the number of paths of length 2 that begin with $i$. However, we must only count such a triangle and such a path if they are disjoint, so we must subtract appropriate occurrences of $C_3$, $H_5$, and $H_6$.

14.

$$n_G(H_{11}) = \sum_{i=1}^{n} \binom{\frac{1}{2}a_{ii}^{(3)}}{2} - 2n_G(H_6).$$

15. Since we have already shown how to compute $n_G(H)$ for all the 6-cyclic graphs, excluding $C_6$, we can use (1) to compute $n_G(C_6)$.

16.

$$n_G(H_{12}) = \sum_{(i,j)\in E} a_{ij}^{(2)} \cdot a_{ij}^{(3)} - 9n_G(C_3) - 2n_G(H_5) - 4n_G(H_6).$$

Here we count the number of triangles through $(i, j)$ and multiply each triangle by the number of walks of length 3 between $i$ and $j$. Since these walks need not be simple, or may intersect the triangle, we may actually be counting $C_3$'s, $H_5$'s, or $H_6$'s. Therefore, we subtract the appropriate values.

17.

$$n_G(H_{13}) = \sum_{(i,j)\in E} \binom{a_{ij}^{(2)}}{3}.$$

18.

$$n_G(H_{14}) = \sum_{i=1}^{n}(d_i - 2) \cdot B_i - 2n_G(H_{12}),$$

where $B_i$ is the number of $C_5$'s passing through $i$. The expression for $B_i$ is

$$B_i = \frac{1}{2}\Bigg[a_{ii}^{(5)} - 10a_{ii}^{(3)} - 4a_{ii}^{(3)}(d_i - 2) - 2 \cdot \sum_{(i,j)\in E} a_{ij}^{(2)}(d_j - 2) - 2 \cdot \sum_{(i,j)\in E} (\tfrac{1}{2}a_{jj}^{(3)} - a_{ij}^{(2)})\Bigg].$$

19.

$$n_G(H_{15}) = \sum_{i=1}^{n}(\tfrac{1}{2}a_{ii}^{(3)}) \left(\sum_{j\neq i} \binom{a_{ij}^{(2)}}{2}\right) - 6n_G(H_6) - 2n_G(H_{12}) - 6n_G(H_{13}). \quad \square$$

Using slightly more effort, it can be shown that, in $O(V^{\omega})$ time, we can also count the number of $C_k$'s, for $3 \leq k \leq 7$, that pass through each vertex of $G$. We have, in fact, done this in the preceding proof for $k = 3, 4, 5$. If the graph $G$ contains a $C_k$, for some $3 \leq k \leq 7$, we can therefore find, in $O(V^{\omega})$ time, a vertex through which such a $C_k$ passes. We can then locate a $C_k$ in the graph in additional $O(E)$ time using Monien's method (Lemma 3.3).

Similar formulae can be obtained, of course, for the number of octagons ($C_8$'s) and even larger cycles. To compute the number of octagons, however, we have to compute first the number of $K_4$'s in the graph, since a $K_4$ is 8-cyclic. We do not know how to do this is $O(V^{\omega})$ time.

It is easy to count the number of $K_4$'s in a graph in $O(V^{\omega+1})$ time: for each vertex, count the number of triangles among its neighbors, sum these numbers, and divide by 4. Counting the number of $K_4$'s in a graph, or, in fact, deciding whether a graph contains a $K_4$, in $o(V^{\omega+1})$ time, is an interesting open problem.

For counting the number of larger cycles using our method, we would have to count the number of larger cliques in the graph. Nešetřil and Poljak [12] give an $O(V^{\omega\lceil \ell/3\rceil})$-time algorithm for deciding whether a graph $G = (V, E)$ contains a $K_l$. It is easy to check that their method can also be used to count the number of such cliques contained in the graph. By combining the method of Nešetřil and Poljak [12] with the ideas used in Section 4, we get the following result.

THEOREM 6.4. *The number of $K_l$'s in an undirected graph $G = (V, E)$ can be counted in either $O(V \cdot (d(G))^{\omega\lceil(\ell-1)/3\rceil})$ or $O(E \cdot (d(G))^{\omega\lceil(\ell-2)/3\rceil})$ time.*

Using an idea similar to the one used in Theorem 3.5, Kloks *et al.* [9] have recently obtained an $O(E^{(\omega+1)/2}) = O(E^{1.69})$ time algorithm for counting the number of $K_4$'s contained in a graph $G = (V, E)$. They also obtain improved results for finding larger cliques and other induced subgraphs.

## References

[1] N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Proceedings of the 2nd European Symposium on Algorithms*, Utrecht, Lecture Notes in Computer Science, Vol. 855, pages 354–364. Springer-Verlag, 1994.

[2] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42:844–856, 1995.

[3] B. Bollobás. On generalized graphs. *Acta Mathematica Academiae Scientarium Hungaricae*, 16:447–452, 1965.

[4] B. Bollobás. *Extremal Graph Theory*. Academic Press, New York, 1978.

[5] J. A. Bondy and M. Simonovits. Cycles of even length in graphs. *Journal of Combinatorial Theory, Series B*, 16:97–105, 1974.

[6] N. Chiba and L. Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14:210–223, 1985.

[7] D. Eppstein. Subgraph isomorphism in planar graphs and related problems. *Proceedings of the 6th Annual ACM–SIAM Symposium on Discrete Algorithms*, San Francisco, CA, pages 632–640, 1995.

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

[9] T. Kloks, D. Kratsch, and H. Müller. Finding and counting small induced subgraphs efficiently. *Proceedings of the 21st International Workshop on Graph-Theoretic Concepts in Computer Science*, Aachen, Lecture Notes in Computer Science, Vol. 1017, pages 14–23. Springer-Verlag, 1995.

[10] D. W. Matula and L. L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, 30:417–427, 1983.

[11] B. Monien. How to find long paths efficiently. *Annals of Discrete Mathematics*, 25:239–254, 1985.

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

[13] C. H. Papadimitriou and M. Yannakakis. The clique problem for planar graphs. *Information Processing Letters*, 13:131–133, 1981.

[14] D. Richards. Finding short cycles in a planar graph using separators. *Journal of Algorithms*, 7:382–394, 1986.

[15] G. Sundaram and S. S. Skiena. Recognizing small subgraphs. *Networks*, 25:183–191, 1995.

[16] R. Yuster and U. Zwick. Finding even cycles even faster. *Proceedings of the 21st International Colloquium on Automata, Languages and Programming*, Jerusalem, Lecture Notes in Computer Science, Vol. 820, pages 532–543. Springer-Verlag, Berlin, 1994. Journal version to appear in *SIAM Journal on Discrete Mathematics*.