Computing the Girth of a Planar Graph *

Hristo N. Djidjev[1]

Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK

Abstract. The girth of a graph G has been defined as the length of a shortest cycle of G. We design an $O(n^{5/4} \log n)$ algorithm for finding the girth of an undirected n-vertex planar graph, giving the first $o(n^2)$ algorithm for this problem. Our approach combines several techniques such as graph separation, hammock decomposition, covering of a planar graph with graphs of small tree-width, and dynamic shortest path computation. We discuss extensions and generalizations of our result.

1 Introduction

Given an (unweighted) graph G, the *length* of a path p in G is the number of the edges in p. The *girth* of G (denoted by $girth(G)$) was defined by Harary [15] as the length of a shortest cycle in G, or infinity if G has no cycle. The girth is a basic combinatorial characteristic of graphs and its relations to other graph properties have been extensively studied. In particular, Erdös [12], Lovasz [19], Bollobás [4], Cook [5], and others studied the relationship between the girth and the chromatic number of a graph. Thomassen [22] and Mader [20] studied the relationship between the girth and the existence of certain type of minors of the graph. Other results relate the girth of the graph to the minimum or the average degrees of its vertices, its diameter, its maximum genus, and its connectivity (see [6]).

The first efficient algorithm for computing the girth of graph was given by Itai and Rodeh [16], who describe an $O(nm)$ algorithm for computing the girth of a general n-vertex m-edge graph G. They also design an $O(n^2)$ algorithm for computing the girth within an additive error of one. Finding shortest cycles of even and odd lengths have been studied by Monien [21] and Vazirani and Yannakakis [23]. There are numerous results on finding a cycle of a given length in general or special graphs; see e.g. [1] and [2] for recent results and references.

In the case of planar graphs Itai and Rodeh [16] give an $O(n)$ algorithm for finding a triangle in the graph, if one exists (and thus solves the girth problem for planar graphs in case $girth(G) \leq 3$ in $O(n)$ time). Their results were generalized by Eppstein [11], who developed an $O(n)$ algorithm for finding the girth of a planar graph G provided $girth(G) = O(1)$. (His algorithm, however, is superexponential with respect to $girth(G)$.)

* This work was partially supported by the EPA grant R82-5207-01-0, EPSRC grant GR/M60750, and RTDF grant 98/99-0140.

Note that, for an embedded planar graph G, the size of each face of G is an upper bound on the girth of G. However, the shortest cycle of G does not necessarily needs to be a face of G.

In this paper we present an algorithm that finds the girth of an arbitrary undirected n-vertex planar graph in $O(n^{5/4} \log n)$ time.

Our approach makes use of recently developed fast dynamic algorithms for computing shortest paths in planar graphs with small face-on-vertex cover [7]. If G has an appropriately large girth, then we show that the shortest cycle in G can be computed by combining separator based divide-and-conquer with the dynamic shortest path algorithm from [7]. If, on the other hand, the girth of G is very small, then we can decompose G into subgraphs of small diameter and such that any shortest cycle of G is contained also in one of the subgraphs of the decomposition. Therefore, we can search for a shortest cycle in each subgraph independently, which will be less expensive in terms of total computation time.

The rest of the paper is organized as follows. In Section 2, we give some definitions and review basic facts related to graph separators. In Section 3 and Section 4, we describe algorithms efficient for graphs of large or small girth, respectively. In Section 5, we describe the algorithm for the case of general graphs. In the last section we discuss some related results and open problems.

2 Preliminaries

By $G = (V, E)$ we denote in the rest of this paper an undirected connected graph with n vertices. Given an edge $e \in E$ and a set $S \subset V$, by $G - e$ we denote the graph $(V, E \setminus \{e\})$ and by $G - S$ we denote the graph $(V \setminus S, E \setminus (S \times S))$. By $deg(G)$ we denote the maximum degree of a vertex of G.

If any edge e of G has a non-negative cost $cost(e)$ associated with it, then the *length* of a path p of G is the sum of the costs of all edges of p. The *distance* between two vertices x and y of G is the minimum length of a path joining x and y. The *single-source shortest path problem* asks, given a source vertex s of G, to compute the distances between s and all other vertices of G. If G is planar, then the single-source shortest path problem for G can be solved in $O(n)$ time [17].

The graph G is *planar* if G can be embedded in the plane so that no two edges intersect except at a common endpoint. A planar graph of n vertices has at most $3n - 3 = O(n)$ edges. A graph already embedded in the plane is a *plane graph*.

A *separator* of G is a set of vertices whose removal leaves no connected component of more than $n/2$ vertices. If G is planar, then G has a separator of size $O(\sqrt{n})$ [18, 9] and if G has genus $g > 0$, then G has a separator of $O(\sqrt{gn})$ vertices [8, 14]. In both cases, the corresponding separators can be found in $O(n)$ time.

If a graph G has non-negative weights associated with its vertices, then a *weighted separator* of G is a set of vertices whose removal leaves no connected component of weight greater than half the weight of G.

For other standard definitions from graph theory see [6] or any other textbook on graphs.

3 Finding shortest cycles in graphs of large girth

In this section we assume that the girth of G is bounded from bellow by some constant $\gamma > 0$. We will use the dynamic shortest path algorithm from [7] that runs faster for graphs of small face-on-vertex cover.

A *hammock decomposition* of an n-vertex graph G was defined by Frederickson in [13] as a decomposition of G into certain outerplanar digraphs called *hammocks*. Hammocks satisfy the following properties:

(i) each hammock has at most four vertices, called *attachment vertices*, shared with the rest of the graph;

(ii) the hammock decomposition spans all the edges of G, i.e., each edge belongs to exactly one hammock; and

(iii) the number of hammocks produced is the minimum possible (within a constant factor) among all possible decompositions.

Frederickson showed in [13] that hammock decompositions of a small cardinality can be computed fast for planar graphs if a good approximation of the minimum cardinality of a face-on-vertex cover is known.

Theorem 1. *[13] Let G be an n-vertex planar graph and let $f(G)$ be the minimum number of faces (over all embeddings of G in the plane) that cover all vertices of G. Then G can be decomposed into $O(f(G))$ hammocks in $O(n)$ time.*

The results of Frederickson [13], Djidjev et al. [7], and others have shown that several shortest paths problems can be solved more efficiently for certain classes of graphs with hammock decompositions of a small cardinality.

Our goal is to make use of the following shortest path algorithm based on the idea of a hammock decomposition.

Theorem 2. *[7] Let G be an n-vertex planar digraph with nonnegative edge costs and assume that G has a hammock decomposition of cardinality q. There exists an algorithm for the dynamic shortest path problem on G with the following performance characteristics:*

(i) preprocessing time and space $O(n)$;

(ii) query time for computing the distance between any two vertices $O(\log n + q)$;

(iii) time for updating the data structure after any edge cost modification or edge deletion $O(\log n)$.

The following algorithm for finding the girth is based on Theorem 1, Theorem 2, and the observation that the size of each face of a plane graph is not less than the girth of that graph.

First we transform G into a planar graph G' of degree at most 3 and $O(m)$ vertices, where $m = O(n)$ is the number of edges of G, by replacing each vertex of G of degree $k > 3$ by a balanced binary tree with k leaves as illustrated on Figure 1. Assign a cost $cost(e)$ to any edge e of the tree equal either to one, if e is incident to a leaf of the tree, or zero, otherwise. This transformation has the following properties:

(i) The edges of G correspond to those edges of G' that have weight one;

(ii) G and G' have the same number of faces;

(iii) Any path of G of length (i.e. number of edges since G is unweighted) l is transformed into a path of weighted length l and at most $l \log n$ edges.

Next we apply to G' the following algorithm.

Algorithm LARGE_GIRTH

{Finds the girth of a biconnected plane graph G with n vertices, f faces, maximum vertex degree 3, and nonnegative costs $cost(\cdot)$ on its edges}

1. Find a hammock decomposition of cardinality $O(f)$ of G using the algorithm from Theorem 1.
2. Preprocess G for shortest path queries using the preprocessing algorithm from Theorem 2.
3. Construct a separator S of G of size $O(\sqrt{n})$ that divides G into components of at most $n/2$ vertices each.
4. For each resulting component K, find the length of a shortest cycle in K by applying this algorithm recursively.
5. For each edge e incident to a vertex of S compute the length $c(e)$ of the shortest cycle in G containing e as follows.
   (a) Change the cost of e to $+\infty$ (which has the effect of deleting e from G) by using the update algorithm from Theorem 2.
   (b) Compute the distance $c'(e)$ between the endpoints of e in the modified graph using the query algorithm from Theorem 2.
   (c) Assign $c(e) := c'(e) + cost(e)$, where $cost(e)$ denotes the original cost of e.
   (d) Assign to e its original cost by using the update algorithm from Theorem 2.
6. Return the minimum length of a cycle in G by combining the results from Steps 4 and 5.

Now we analyze the correctness of the algorithm and its running time for $\gamma = O(n^{1/2-\varepsilon})$ for some $\varepsilon > 0$, since the value of the parameter γ that will be chosen in Section 5 will satisfy that condition.

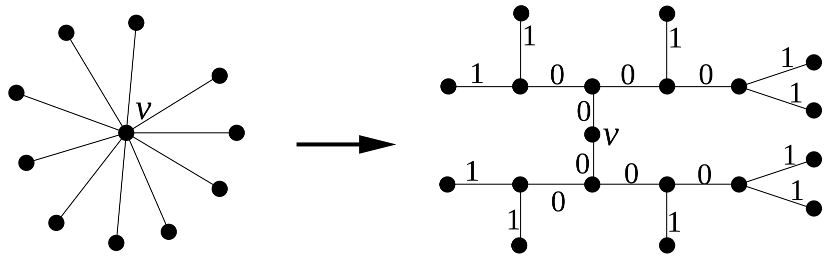

Fig. 1. Reducing the maximum vertex degree.

Lemma 1. *If G is a biconnected planar graph G of degree three and $girth(G) \geq \gamma$, then Algorithm* LARGE_GIRTH *computes the girth of G in $O(n^{3/2}/\gamma)$ time, assuming $\gamma = O(n^{1/2-\varepsilon})$ for some $\varepsilon > 0$.*

Proof: *Correctness:* Assume that c is a shortest cycle of G. If c does not contain a vertex of S, then c will belong to some component K considered in Step 4. K can not contain a cycle shorter than c since otherwise c will not be a shortest cycle of G. Hence the length of c will be correctly computed in Step 4.

If c contains a vertex from S, then c will contain an edge e incident to a vertex from S that is considered in Step 5. Since c is a shortest cycle of G, then $c - e$ will be a shortest path between the endpoints of e in $G - e$ (Figure 2). Hence in that case the length of c will be correctly computed in Step 5.

Time complexity: The time for Steps 1, 2, 3, and 6 is $O(n)$. The number of iterations of Step 5 is $O(\sqrt{n})$, since each vertex of G (and thus each vertex of S) has degree $O(1)$. The time for each iteration of Step 5 is dominated by the time of Step 5 (b), which time, according to Theorem 2 and Theorem 1, is $O(\log n + f)$. Hence the time $T(n, f)$ for Algorithm LARGE_GIRTH satisfies the recurrence

$$\begin{aligned} &T(1, f) = O(1), \\ &T(n, f) \leq \max\{T(n/2, f_1) + T(n/2, f_2) \mid f_1 + f_2 \leq f\} \\ &\qquad\qquad + O(n) \;+\; O(\sqrt{n}\,(f + \log n)), \quad \text{for } n > 1. \end{aligned}$$

For $n > 1$, one can represent $T(n, f)$ as

$$T(n, f) = T_1(n) + T_2(n, f),$$

where

$$\begin{aligned} &T_1(n) \leq 2T_1(n/2) + O(n) \quad \text{and} \\ &T_2(n, f) \leq \max\{T_2(n/2, f_1) + T_2(n/2, f_2) \mid f_1 + f_2 \leq f\} \\ &\qquad\qquad + O(\sqrt{n} \cdot f). \end{aligned}$$

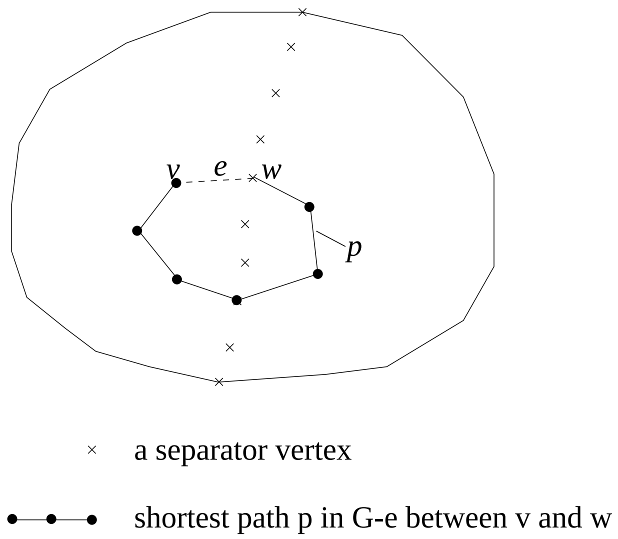

Fig. 2. An Illustration to the proof of Lemma 1. The cycle $p + e$ is a shortest cycle containing e.

By unfolding, we have for the case $n > 1$

$$\begin{aligned} T_2(n, f) &\leq c\,(\sqrt{n} \cdot f + (\sqrt{n/2} \cdot f_1 + \sqrt{n/2} \cdot f_2) + \cdots) \\ &= c\,f\,(\sqrt{n} + \sqrt{n/2} + \cdots) = cfO(\sqrt{n}) \;=\; O(f\sqrt{n}). \end{aligned}$$

Clearly $T_1(n) = O(n \log n)$ and hence $T(n, f) = O(f\sqrt{n} + n \log n)$.

Finally, since G is biconnected each face of G is a cycle, which implies that the size of each face is not less than γ. Moreover, each edge of G belongs to exactly two faces of the embedding and hence $f \leq 2 \cdot |E(G)| \,/\, \gamma = O(n/\gamma)$.

Thus $T(n, f) = O(f\sqrt{n} + n \log n) = O(n^{3/2}/\gamma)$ for $\gamma = O(n^{1/2-\varepsilon})$. □

4 Finding shortest cycles in graphs of small girth

In this section we assume that the girth of the input graph is smaller than certain parameter γ whose value will be determined in Section 5. For the proof of the next lemma we use a technique developed by Baker [3] and Eppstein [11].

Lemma 2. *Let G be an n-vertex planar graph and let d be any integer. Then we can find in $O(n)$ time a set of subgraphs G_i of G with the following properties:*

(i) The sum of the sizes of all subgraphs G_i is $O(n)$;

(ii) Every subgraph of G_i has a separator of size $O(d)$;

(iii) Any cycle of length at most $4d$ is contained in some of the subgraphs G_i.

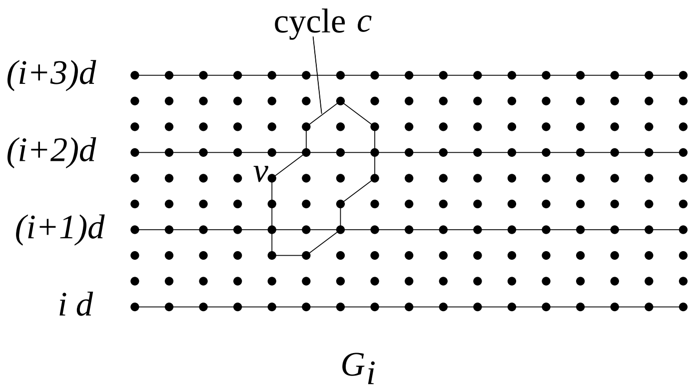

Fig. 3. An Illustration to the proof of Lemma 2.

Proof: Construct a breadth-first spanning tree T of G and divide the vertices of G into levels according to their distance to the root of T. Define the graph G_i to be the graph induced by the vertices on levels from id to $(i+3)d-1$.

Any vertex on a level between id and $(i+1)d-1$ will belong only to the subgraphs G_{i-2}, G_{i-1} and G_i (if they exist). Thus any vertex of G will belong to at most 3 of the subgraphs G_i. Property (i) follows.

In order to prove (ii), for each graph G_i define a new vertex v_i and connect it to all vertices on level id. The resulting graph G'_i has a breadth first spanning tree of radius $r = 3d$. According to a lemma from [9], each weighted planar graph with a breadth-first spanning tree of radius r has a weighted separator of no more than $3r+1$ vertices one of which is the root of the tree. Let H be any subgraph of G_i. Assign a weight 1 to each vertex of H and a weight 0 to each vertex of G'_i that is not in H. By applying the lemma from [9] on the weighted version of G'_i, we find a separator of H of size at most $9d = O(d)$.

Finally, for proving that condition (iii) holds, we note that the endpoints of any edge of G belong either to the same level, or to two consecutive levels of G. Let c be a cycle of length not exceeding $4d$ (Figure 3). Then the set of the levels of the vertices on c form an interval $[a,b]$ of integers and let v be a vertex on level $l = \lceil (a+b)/2 \rceil$. Then each vertex on c is at distance at most d from v. Hence, if the level l is between levels $(i+1)d$ and $(i+2)d-1$, then all vertices from c will belong to G_i. Thus requirement (iii) is also satisfied. □

Next we show how to compute the length of a shortest cycle efficiently for weighted graphs with small separators using divide-and-conquer.

Algorithm SMALL_SEPARATOR

{Finds the girth of a graph G with nonnegative costs $cost(\cdot)$ on its edges if G has $O(\tau$ separator and $degree(G) = O(1)$}

1. Construct a separator S of G of size $O(\tau)$ dividing the graph into components of no more than $n/2$ vertices each, where n denotes the number of vertices of G.
2. For each component K of the partition of G find the length of the shortest cycle in K by this algorithm recursively.
3. For each edge e incident to a vertex from S find the length $c(e)$ of the shortest cycle in G containing e as follows.
   (a) Compute the distance $c'(e)$ between the endpoints of e in $G - e$ by running the linear time single-source shortest paths algorithm from [17] on $G - e$ with source any of the endpoints of e.
   (b) Assign $c(e) := c'(e) + cost(e)$.
4. Return the minimum length of a cycle found in Step 2 or Step 3.

Lemma 3. *Algorithm* SMALL_SEPARATOR *computes the girth of* G *in* $O(\tau n \log n)$ *time, assuming the tree-width of* G *is no more than* τ*.*

Proof: *Correctness.* Let c be a shortest cycle in G. If c contains a vertex of S, then its length will be computed in some iteration of Step 3. Otherwise c will be contained in some component of $G - S$ and its length will be computed in Step 2.
Time complexity. The time needed for Step 1 is $O(n)$, where n is the number of vertices of G [18]. Since each vertex of S has degree $O(1)$, then the number of iterations of Step 3 is $O(|S|) = O(\tau)$. The time for each iteration is dominated by the time for running the shortest paths algorithm in Step 3(a), which is $O(n)$. This implies $O(\tau n)$ time bound for Step 3. Thus the total time $T(n)$ for Algorithm SMALL_SEPARATOR satisfies the recurrence

$$\begin{aligned} &T(1) = O(1), \\ &T(n) \leq 2T(n/2) + O(\tau n), \quad \text{for } n > 1, \end{aligned}$$

whose solution is $T(n) = O(\tau n \log n)$. □

In a preprocessing step, we transform G into a graph G' of maximum vertex degree three, where any edge e of G' has a cost $cost(e)$ equal to 1 or 0 depending on whether e corresponds to an edge of the original graph or not (see Figure 1). Then we apply to G' the following algorithm which is based on Algorithm SMALL_SEPARATOR and Lemma 2.

Algorithm SMALL_GIRTH

{Finds the girth of an n-vertex planar graph G assuming $girth(G) \leq \gamma$}

1. Apply Lemma 2 on G with $l = \lceil \gamma \log n \rceil$. Let $G_1, \ldots, G_s$ be the resulting subgraph cover of G.
2. For each G_i, $i = 1, \ldots, s$, find the length c_i of the shortest cycle in G_i ($c_i = \infty$ if G_i is acyclic) by applying Algorithm SMALL_SEPARATOR.
3. Return $girth(G) := \min\{c_i \mid i = 1, \ldots, s\}$.

Lemma 4. *Algorithm* SMALL_GIRTH *finds the girth of a planar graph G in $O(\gamma n \log^2 n)$ time, assuming $girth(G) \leq \gamma$.*

Proof: The correctness follows from Lemma 2 and the correctness of Algorithm SMALL_SEPARATOR. The time for Step 1, and Step 3 is $O(n)$. By Lemma 3, the time for computing c_i in Step 2 is $O(\gamma \log n \, n_i \log n_i)$, where n_i is the number of the vertices of G_i, since by Lemma 2 each graph G_i has a separator of size $O(l)$, where $l = \gamma \log n$. Thus the total time for Step 2 is

$$\sum_{i=1}^{s} O(\gamma \log n \, n_i \log n_i) \leq O(\gamma \log^2 n) \sum_{i=1}^{s} n_i$$

$$= O(\gamma \log^2 n) O(n) = O(\gamma n \log^2 n).$$

□

5 The general case

In the case of an arbitrary planar input graph G we find the girth of G simply by applying one of the algorithms from the previous two sections depending on the maximum face size of a given embedding of G.

Algorithm FIND_GIRTH

{Finds the girth of an arbitrary n-vertex planar graph G}

1. If G is not biconnected, then compute its biconnected components and apply the remaining steps of this algorithm on any of the biconnected components (since each cycle in a graph belongs to exactly one of its biconnected components).
2. Transform G into a weighted planar graph G' with $n' = O(n)$ vertices and of a maximum vertex degree three (as discussed in Section 3 and illustrated on Figure 1).
3. Embed G' in the plane and find the maximum face size h of the embedding.

4. If $h = O(n^{1/4}/\log n)$, then compute the girth of G by applying on G' Algorithm Small_Girth.
5. Else (if $h = \Omega(n^{1/4}/\log n)$) compute the girth by applying on G' Algorithm Large_Girth.

Theorem 3. *Algorithm* Find_Girth *finds the girth of an n-vertex planar graph in $O(n^{5/4}\log n)$ time.*

Proof: The correctness of the algorithm follows from Lemma 4 and Lemma 1.

If $h = O(n^{1/4}/\log n)$, then the time of the algorithm will be determined by the time for running Algorithm Small_Girth. By Lemma 4, that time is $O(\gamma n \log^2 n)$, where $\gamma = girth(G)$. Since $\gamma \leq h = O(n^{1/4}\log n)$, then the time of the algorithm in this case is $O((n^{1/4}/\log n)\, n \log^2 n) = O(n^{5/4}\log n)$.

If $h = \Omega(n^{1/4}/\log n)$, then by Lemma 1 the time for running Algorithm Find_Girth will be $O(n^{3/2}/\gamma) = O(n^{3/2}/\,(n^{1/4}/\log n)\,) = O(n^{5/4}\log n)$. □

6 Related problems

The technique discussed here can be used to solve other problems related to shortest cycles in graphs. Algorithm Find_Girth can be used to find a shortest non-facial cycle of a planar graph in $O(n^{5/4}\log n)$ time. Furthermore, using graph separation and divide-and-conquer, we can compute the girth of a directed graph in $O(n^{3/2})$ time. Finally, making use of the $O(\sqrt{gn})$ separator theorem for graphs of genus $g > 0$ [8, 14, 10], one can construct efficient algorithms for the above problems for graphs of genus $g = o(n)$. We will describe algorithms for solving the above problems in the full version of the paper.

This paper leaves several open problems, including the following:

1. Construct an $o(n^{3/2})$ time algorithm for computing the girth of a directed planar graph.
2. Develop efficient algorithms for finding shortest cycles in graphs with arbitrary nonnegative costs (lengths) on edges.
3. Construct $o(nm)$ algorithms for finding the girth of general graphs.

It will be also interesting to implement the algorithm from this paper and test its practicality.

References

1. Alon, Yuster and Zwick. Color-coding. *Journal of the ACM*, 42:844-856, 1995.
2. Alon, Yuster and Zwick. Finding and counting given length cycles. *Algorithmica*, 17:209-223, 1997.
3. Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, January 1994.

4. Béla Bollobás. Chromatic number, girth and maximal degree. *Discrete Mathematics*, 24:311–314, 1978.
5. R. J. Cook. Chromatic number and girth. *Periodica Mathematica Hungarica*, 6(1):103–107, 1975.
6. Reinhard Diestel. *Graph Theory.* Springer-Verlag, Berlin, Heidelberg, New York, Tokio, 1997.
7. H. Djidjev, G. Pantziou, and C. Zaroliagis. Improved algorithms for dynamic shortest paths. *Algorithmica*, 1999. To appear.
8. Hristo N. Djidjev. A separator theorem. *Compt. rend. Acad. bulg. Sci.*, 34:643–645, 1981.
9. Hristo N. Djidjev. On the problem of partitioning planar graphs. *SIAM Journal on Algebraic and Discrete Methods*, 3:229–240, 1982.
10. Hristo N. Djidjev. A linear algorithm for partitioning graphs of fixed genus. *Serdica*, 11:329–341, 1985.
11. David Eppstein. Subgraph isomorphism for planar graphs and related problems. In *Proc. 6th Symp. Discrete Algorithms*, pages 632–640. Assoc. Comput. Mach. and Soc. Industrial & Applied Math., 1995.
12. Paul Erdös. Graph theory and probability. *Canad. J. Math.*, 11:34–38, 1959.
13. G.N. Frederickson. Planar graph decomposition and all pairs shortest paths. *Journal of the ACM*, 38:162–204, 1991.
14. John R. Gilbert, Joan P. Hutchinson, and Robert E. Tarjan. A separator theorem for graphs of bounded genus. *J. Algorithms*, 5:391–407, 1984.
15. F. Harary. *Graph Theory.* Addison-Wesley, Reading, Massachusetts, 1969.
16. A. Itai and M. Rodeh. Finding a minimum circuit in a graph. *SIAM J. Computing*, 7:413–423, 1978.
17. P. Klein, S. Rao, M. Rauch, and S. Subramanian. Faster shortest-path algorithms for planar graphs. In *26th ACM Symp. Theory of Computing*, pages 27–37, 1994.
18. Richard J. Lipton and Robert E. Tarjan. A separator theorem for planar graphs. *SIAM J. Appl. Math*, 36:177–189, 1979.
19. L. Lovasz. On chromatic number of finite set systems. *Acta Math. Acad. Sci. Hun.*, 19:59–67, 1968.
20. Wolfgang Mader. Topological subgraphs in graphs of large girth. *Combinatorica*, 18:405, 1998.
21. B. Monien. The complexity of determining a shortest cycle of even length. *Computing*, 31:355–369, 1983.
22. Carsten Thomassen. Paths, circuits and subdivisions. In *Selected Topics in Graph Theory, ed. Lowell W. Beineke and Robin J. Wilson, Academic Press*, volume 3. 1988.
23. Vazirani and Yannakakis. Pfaffian orientations, 0-1 permanents, and even cycles in directed graphs. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 25, 1989.