Crossing Number of Graphs with Rotation Systems

Michael J. Pelsmajer[1], Marcus Schaefer[2], and Daniel Štefankovič[3]

[1] Illinois Institute of Technology, Chicago, IL 60616, USA
pelsmajer@iit.edu
[2] DePaul University, Chicago, IL 60604, USA
mschaefer@cs.depaul.edu
[3] University of Rochester, Rochester, NY 14627, USA
stefanko@cs.rochester.edu

Abstract. We show that computing the crossing number of a graph with a given rotation system is **NP**-complete. This result leads to a new and much simpler proof of Hliněný's result, that computing the crossing number of a cubic graph (without rotation system) is **NP**-complete. We also investigate the special case of multigraphs with rotation systems on a fixed number k of vertices. For $k = 1$ and $k = 2$ the crossing number can be computed in polynomial time and approximated to within a factor of 2 in linear time. For larger k we show how to approximate the crossing number to within a factor of $\binom{k+4}{4}/5$ in time $O(m^{k+2})$ on a graph with m edges.

Keywords: crossing number, computational complexity, computational geometry.

1 Introduction

Computing the crossing number is **NP**-complete, as shown by Garey and Johnson [5]. Hliněný recently showed, using a rather complicated construction, that even determining the crossing number of a cubic graph is **NP**-complete [6], a long-standing open problem [1].

We investigate a new approach to cubic graphs through graphs with rotation systems. We show that determining the crossing number of a graph with a given rotation system is **NP**-complete, and then prove that this problem is equivalent to determining the crossing number of a cubic graph. This also gives a new and easy proof that determining the minor-monotone crossing number (defined in [2]) is **NP**-complete.

Graphs with rotation systems are of interest in their own right; we have encountered them several times during recent research projects [12,14,13]. Indeed, at the core of our separation of the crossing number from the odd crossing number is a loopless multigraph on two vertices with rotation [13]. In Section 4 we will see that the crossing number can be computed efficiently for one-vertex graphs with rotation and at least approximated efficiently for loopless multigraphs on

two vertices (the problem is in polynomial time for two-vertex multigraphs but requires linear programming [14]). We also show some interesting connections to string matching problems. Finally, we give an approximation algorithm to compute the crossing number of k-vertex multigraphs with rotation to within a factor of $O(k^4)$. We do not know whether this problem can be solved exactly in polynomial time.

2 NP- Hardness

Consider a graph drawn in the plane (or any orientable surface). The *rotation* of a vertex is the clockwise order of its incident edges. A *rotation system* is the list of rotations of every vertex. We are interested in drawings of a graph in the plane with a fixed rotation system.

We also consider "flipped" rotations (previously seen in [13]). Given a rotation of a vertex v, the *flipped rotation* reverses the cyclic order of the edges incident to v.

Theorem 1. *Computing the crossing number of a graph with a given rotation system is* **NP***-complete. The problem remains* **NP***-complete if we allow the rotation at each vertex to flip independently.*

Proof. We adapt Garey and Johnson's reduction from OPTIMAL LINEAR ARRANGEMENT to CROSSING NUMBER [5]. Given a graph $G = (V, E)$, a *linear arrangement* is an injective function $f : V \to 1, \ldots, |V|$, and the *value* of the arrangement is computed as

$$\sum_{uv \in E} |f(u) - f(v)|.$$

Given G and k, deciding whether G allows a linear arrangement of value at most k is **NP**-complete [5, GT42].

Let us fix a connected graph $G = (V, E)$, with $V = v_1, \ldots, v_n$, $m = |E|$, and k. We may assume that $n \leq m$. From G we construct an edge-weighted graph H with fixed rotation system, as shown in Figure 1. The use of weighted edges simplifies the construction; later we will replace each weighted edge by a small unweighted graph, obtaining a simple graph H' with a fixed rotation system. Note that for a fixed drawing of a weighted graph, a crossing of an edge of weight k with an edge of weight l contributes kl to the crossing number.

We start with a cycle $(u_1, \ldots, u_{4n})$, and a single vertex u_0 connected to each vertex on the cycle. We choose the edge-weights of this part of the graph so high that it has to be embedded without any intersections.

For every $1 \leq i \leq 2n$ we connect u_i to u_{4n+1-i} by a path P_i of length 2 and edges of weight w. Furthermore, we connect the midpoints of P_i and P_{2n+1-i} by a path Q_i of length 3 with edges of weight w', whose middle edge $a_i b_i$ has been replaced by two edges of weight $w'/2$ $(1 \leq i \leq n)$.

Finally, we encode G as follows: for each edge $v_i v_j \in E$ we add an edge from a_i to b_j (with $i < j$, an arbitrary choice). The rotation of H is as shown in

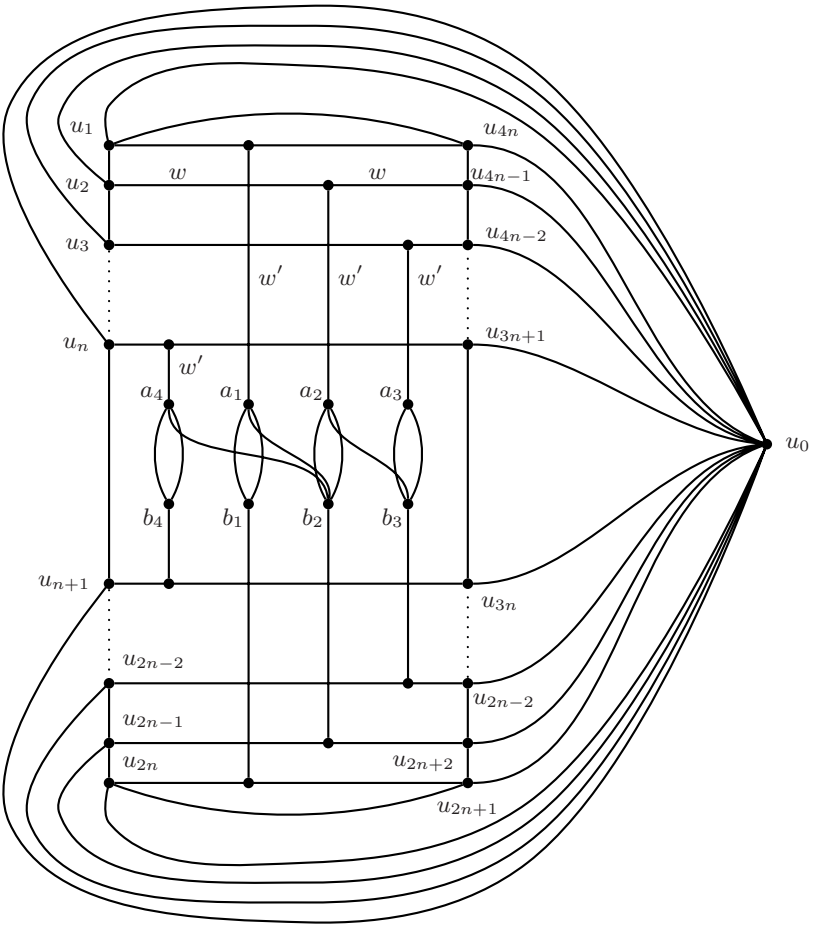

Fig. 1. The graph H

Figure 1. At a_i, each edge from E is inserted into the rotation at a_i between the two a_i, b_i-edges of weight $w'/2$; we do likewise at every b_i. The edges of E at a_i can be ordered arbitrarily (same at b_i).

This concludes the description of H. We let $k' = n(n-1)ww' + kw' + m^2$, where $w = 5m^4$ and $w' = 2m^2$. We claim that G allows a linear arrangement of value at most k if and only if H (with the rotation system shown in the drawing) has crossing number at most k'.

If G has a linear arrangement of value at most k, we can draw H using the order of the v_i in that linear arrangement to obtain a drawing of crossing number at most k' (the m^2 term compensates for the potential pairwise crossings of the edges in H that represent edges in E).

For the reverse implication, consider a drawing of H with crossing number at most $k' = n(n-1)ww' + kw' + m^2$. Then $k' < n^2ww' + m^2w' + m^2$, and

by choice of w and w' this is at most $10m^8 + 2m^4 + m^2 < w^2$. Hence, in our drawing, no two edges of weight w intersect each other, and, therefore, the paths P_i $(1 \leq i \leq 2n)$ are drawn as shown in Figure 1.

Next, consider the modified paths Q_i. Q_i must intersect each of the paths P_{i+1} through P_{2n-i}, contributing $(2n-2i)ww'$ to the crossing number. Summing these values for $i = 1, \ldots, n$, we observe a contribution of at least $n(n-1)ww'$ by intersections between the Q_i and the P_i to the crossing number. This leaves $k' - n(n-1)ww' = kw' + m^2 < m^2w' + m^2 = (w'/2)w' + w'/2 < w'w' < w'w$ crossings, implying that there cannot be any further intersections between a Q_i and a P_i (since it would contribute $w'w$ to the crossing number, more than is left). By the same reasoning, we also do not have intersections between any two Q_i.

Finally, we want to argue that all the a_i and b_i lie between P_n and P_{n+1}. Since Q_n lies entirely between P_n and P_{n+1} (as we argued earlier), so do a_n and b_n. Consider any a_i or b_i. As G is connected by assumption, there is a path from a_n to a_i using edges encoding G and edges of weight $w'/2$. If this path intersects P_n or P_{n+1}, it contributes w or more to the crossing number. However, since $k' - n(n-1)ww' = kw' + m^2 < m^2w' + m^2 = 2m^4 + m^2 < 5m^4 = w$, this is not possible. Therefore, a_i and b_i are also located between P_n and P_{n+1}.

In summary, the drawing of H looks as shown in Figure 1. This drawing clearly indicates a linear arrangement f of G. An edge $e = uv$ contributes at least $|f(u) - f(v)|w'$ to the crossing number of H, so $\sum_{uv \in E} |f(u) - f(v)| \leq kw' + m^2$. Since $m^2 < w$, the value of the linear arrangement is at most k.

The last step is to replace each edge e of weight x by x parallel edges, and then subdivide each of those edges: the effect is that e is replaced by a copy of $K_{2,x}$ with the endpoints of e identified with the partite set of size 2. The new edges are inserted in the rotation at where e was, and the new edges are ordered as indicated in Figure 2. Thus we obtain an unweighted graph H' from H. Since we can draw any of the parallel edges alongside whichever one is involved in the smallest number of crossings, we may assume that an optimal drawing of H' has all parallel edges routed in parallel; also, subdivisions do not affect the crossing number. Therefore, $\mathrm{cr}(H') = \mathrm{cr}(H)$, and H' is an unweighted graph with fixed rotation system for which is it is **NP**-hard to determine the crossing number.

Note that the argument showing that the drawing of H looks as shown in Figure 1 did not make any assumptions about the rotation at a vertex. Therefore, even if we allow flipped rotations, we can still conclude that the drawing of H yields a linear arrangement of value at most k. Consequently, computing the crossing number of graphs with rotation systems remains **NP**-complete if we allow rotations to flip.

Remark 1. The construction in the proof of Theorem 1 can be modified to work for other crossing number variants, such as odd-crossing number, pair-crossing number, and rectilinear crossing number (for which all edges of the graph have to be realized as line segments).

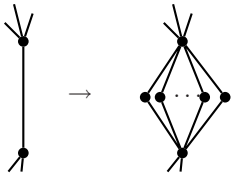

Fig. 2. Replacing an edge by parallel paths

3 Cubic Graphs

We can use Theorem 1 to prove that computing the crossing number of a cubic graph is **NP**-complete. This was a long-standing open question that was solved only recently by Petr Hliněný, using a rather complicated construction. The idea of the proof is to replace each vertex of a graph with rotation system with a hexagonal grid, simultaneously making the graph cubic and mimicking the rotation system. (Hexagonal grids are used in Hliněný's original proof as well.)

Theorem 2 (Hliněný [6]). *Computing the crossing number of a* 3*-connected, cubic graph is* **NP***-complete.*

Remark 2. The argument of Theorem 2 also works for straight-line drawings. Combining this observation with Remark 1 shows that it is **NP**-hard to compute the rectilinear crossing number of a cubic graph.

As Hliněný observes, Theorem 2 also implies that computing the minor-monotone crossing number is **NP**-complete [6]. Another result, which follows immediately (as observed in [3]) is that it is **NP**-hard to find a drawing of a directed graph in which all incoming (and therefore all outgoing) edges at a vertex are consecutive and which minimizes the crossing number.

Our Theorem 1 is in turn derivable from Hliněný's result, as we will show in the full version of the paper.

4 Parameterization

One way to parameterize the crossing number problem is by the number of vertices of the graph. The question becomes interesting if we allow multiple edges and loops. Without rotation, the problem is equivalent to computing the crossing number of a weighted graph without multiple edges and loops, with the cost of an intersection being the product of the weights of the edges involved: Given a graph $G = (V, E)$ with multiple edges and loops, note that in a crossing-number optimal drawing any two edges with the same endpoints can be routed in parallel. If we let G' be the complete graph on V with edge weights $w(uv)$ equal to the number of edges in E between u and v, then the weighted crossing number of G' equals $\mathrm{cr}(G)$. Moreover, that weighted crossing number of G' can be easily computed by exhaustively trying all possible drawings in time $O(2^{|V|^2}(\log |E| + |V|^2))$.

The problem becomes nontrivial if the graph G is given with a rotation system of its edges. In the following sections we discuss the cases of one and two vertices connecting them with well-known problems such as determining the number of inversions in a permutation and finding the edit distance of two cyclic words. We also include a weak approximation result for the general case. We start by investigating the case of two vertices.

4.1 Two Vertices

In this section we consider graphs on two vertices, allowing multiple edges, but no loops. The crossing number of a two-vertex graph can be expressed as the solution of an integer linear program whose relaxation can be used to compute the optimal integer solution in polynomial time as we showed earlier [13].

Here we want to give a fast and simple 2-approximation algorithm for the two-vertex case. To do so, we look at the crossing number problem as an *edit-distance* problem on words. The edit distance between two words is the smallest number of operations transforming one word into the other. There are numerous variants of this problem depending on which operations are allowed and what the associated costs are [15,9]. There are also several papers studying objects other than words, such as trees and cyclic words (also known as necklaces) [10,11,7], but it seems the particular variant we find needful here—allowing only swaps (at unit cost) on cyclic words—has not so far been considered at all. A *swap* is the transposition of two adjacent letters in a word. A *cyclic word* is the equivalence class of a word under cyclic shifts (we will use the letter ρ to denote the cyclic shift of a word by one position to the right). The last and first letter of a cyclic word are considered adjacent. Let $d_s(u, v)$ be the smallest number of swaps transforming u into v, where u and v are normal words. Similarly, let $d_s^\rho(u, v)$ be the smallest number of swaps transforming u into v allowing cyclic shifts at no cost. Then $d_s^\rho(u, v)$ is the swapping distance of the two *cyclic* words represented by the words u and v. E.g. $d_s^\rho(abcd, cdba) = 1$, while $d_s(abcd, cdba) = 5$.

Computing d_s is easy (see [15]). Our goal is the computation of $d_s^\rho(u, v)$.

Swapping distance of Cyclic Words
Instance: Two words u, v, integer k.
Question: Is $d_s^\rho(u, v) \leq k$?

We do not know how hard this problem is in general; however, with the restriction that the words contain each letter exactly once, we can solve the problem. Indeed, in that case it is equivalent to computing the crossing number of a graph G with rotation system on two vertices (details will appear in the journal version).

We rephrase the restricted swapping-distance problem as follows: we can assume that $u = 123 \cdots m$ and $v = \sigma(1)\sigma(2) \cdots \sigma(m)$ for some permutation σ of Z_m (the cyclic group of m elements). Letting G be the 2-vertex multigraph defined by the clockwise rotations u and v^R, we define $\mathrm{cr}(\sigma) := \mathrm{cr}(G)$. We call two permutations σ, τ *circular-equivalent* if there exists a k such that $\sigma(i) = \tau(i + k)$ for all $i \in Z_n$. Each equivalence class is a *circular permutation* (this corresponds

exactly to the cyclic words). We will use σ to represent a permutation as well as the corresponding circular permutation. If σ and τ are circular equivalent, then $\mathrm{cr}(\sigma) = \mathrm{cr}(\tau)$.

We next define a function $\widetilde{\mathrm{cr}}$ on circular permutations σ which will be seen to be related to the crossing number of the corresponding 2-vertex multigraph G. Consider a fixed permutation τ. We wish to consider "forward" and "backward distance" from i to $\tau(i)$ in Z_m, as if the the elements in the list τ were placed clockwise along a circle with the same distance between each consecutive pair (including $\tau(m)$ and $\tau(1)$). We define $d^+(i)$ to be $\tau(i) - i \mod m$; note that $0 \leq d^+(i) < m$. Also let $d^-(i) = i - \tau(i) \mod m$ and let $d(i) = \min(d^+(i), d^-(i))$. Note that if the aforementioned circle has circumference m, then $d^+(i)$ measures the clockwise distance along the circle from i to $\tau(i)$, and $d^-(i)$ measures the counterclockwise distance from i to $\tau(i)$. Finally, we define $d(\tau)$ to be the sum of $d(i)$ over $1 \leq i \leq m$.

For a circular permutation σ, let $\widetilde{\mathrm{cr}}(\sigma)$ be the minimum of $d(\tau)$ over all $\tau \equiv \sigma$. Equivalently, $\widetilde{\mathrm{cr}}(\sigma) = \min_{1 \leq i \leq m} d(\sigma \circ \rho^i)$, where $\rho^i(j) = i + j$ for all $1 \leq i \leq m$.

We claim that $\widetilde{\mathrm{cr}}$ approximates the cyclic swapping distance of two words to within a factor of 2. We leave the proof to the journal version.

Theorem 3. *For a 2-vertex loopless multigraph G represented by a circular permutation σ,*

$$\mathrm{cr}(G) \leq \widetilde{\mathrm{cr}}(\sigma) \leq 2\,\mathrm{cr}(G).$$

Remark 3. The bounds of Theorem 3 are asymptotically optimal: for $\sigma_n := (1\ 2)(3\ 4)\cdots(2n-1\ 2n)$ we have $\widetilde{\mathrm{cr}}(\sigma) = 2n$ and $\mathrm{cr}(G) = n$; for the lower bound consider $\tau_n := (1\ n)$ (as a permutation of numbers $1, \ldots, 2n$), then $\widetilde{\mathrm{cr}}(\tau_n) = 2n - 2$ and $\mathrm{cr}(G) = 2n - 3$.

Remark 4. We have seen that the crossing number of a two-vertex graph equals the swapping distance of two cyclic words. If instead of cyclic words we consider normal words, the swapping distance still equals the crossing number of a two-vertex graph where both vertices lie on the boundary of a disk (and all the edges are within the disk). In that context, the analogue of Theorem 3 is known as Spearman's Footrule and was first proved by Diaconis and Graham [4].

Theorem 3 gives us a fast and easy way to approximate $\mathrm{cr}(G)$ for a 2-vertex multigraph with rotation system. Computing $\widetilde{\mathrm{cr}}(\sigma)$ from the definition can be done in quadratic time; however, this can easily be improved by first sorting the $d(i)$ (which can be done in linear time) and then trying all rotational shifts ρ^j of σ. We keep the optimal shifts sorted by value and distinguish between two different types of optimal shift: forward and backward. Updating the optimal shift and its direction might not be constant time for adding a single shift, but an amortized analysis shows that the whole algorithm can be made to run in linear time.

Corollary 1. *The crossing number of a 2-vertex loopless multigraph with rotation system can be approximated to within a factor of 2 in linear time.*

4.2 One Vertex

Given a graph with a rotation system on a single vertex (with loops), it is quite straightforward to compute its crossing number in quadratic time.

In contrast, a linear time algorithm for the one-vertex case would come as a surprise, since the problem contains as a special case a well-studied problem: computing the number of inversions of a permutation. Given a permutation π over $\{1, \ldots, n\}$, an *inversion* of π is a pair (i, j) such that $i < j$ and $\pi(i) > \pi(j)$. It is well-known that the number of inversions of a permutation π equals $d_s(123 \ldots n, \pi(1)\pi(2) \ldots \pi(n))$ (see, for example [8, Section 5.1.1]). The best-known algorithms for either problem run in $\Theta(n \log n)$.[1]

The inversion problem is easily encoded as a crossing number problem on a single vertex: simply let the rotation at the vertex be $12 \ldots n\pi(n)\pi(n-1) \ldots \pi(2)\pi(1)$.

However, the one-vertex case can also be considered a special case of the two-vertex case (split the vertex into two and connect the two vertices with a large number of neighboring parallel edges). Hence we can approximate the crossing number of a one-vertex graph and therefore the number of inversions of a permutation in linear time to within a factor of 2 using our approximation algorithm. As we mentioned in Remark 4, this result is known as Spearman's Footrule.

We can compute the crossing number of a one-vertex graph exactly in time $\Theta(n \log n)$, which extends the algorithm for computing the number of inversions of a permutation. The proof will appear in the journal version.

Theorem 4. *The crossing number of a one-vertex graph with rotation system can be computed in time $O(n \log n)$.*

4.3 Several Vertices

There is little we can say at this point about how hard it is to compute the crossing number of a graph with a rotation system on a fixed number k of vertices when $k \geq 3$. Using results from a previous paper [12], however, we can give at least an approximation result. In this section we allow both loops and multiple edges.

Theorem 5. *We can approximate the crossing number of a multigraph $G = (V, E)$ with rotation system on k vertices to within a factor of $\binom{k+4}{4}/5$ in time $O(m^{k+2})$ where $k = |V|$ and $m = |E|$.*

In [12] we showed that $\mathrm{cr}(G) \leq \mathrm{ocr}(G)\binom{k+4}{4}/5$, where $\mathrm{ocr}(G)$ is the *odd-crossing number* of G, that is, the smallest number of pairs of edges that cross an odd number of times in any drawing of G. In fact, the proof applies to a multigraph G with rotation system π, yielding $\mathrm{cr}(G, \pi) \leq \mathrm{ocr}(G, \pi)\binom{k+4}{4}/5$. The proof works by choosing a sequence of k edges $e_1, \ldots, e_k$ and contracting G along those edges

[1] See [8, Exercises 5.1.1-6 and 5.2.4.-21]. Wagner's linear time algorithm [15] for computing the swapping distance of words is wrong.

obtaining a graph G' with rotation system π' on a single vertex. For graphs on a single vertex crossing number and odd crossing number are the same, hence, $\mathrm{cr}(G', \pi') = \mathrm{ocr}(G', \pi')$. Furthermore, the sequence of edges is chosen such that $\mathrm{cr}(G', \pi') \leq \mathrm{ocr}(G, \pi)\binom{k+4}{4}/5$. In other words, $\mathrm{ocr}(G, \pi) \geq \mathrm{cr}(G', \pi')/(\binom{k+4}{4}/5)$. The redrawing procedure of the proof establishes that $\mathrm{ocr}(G, \pi) \leq \mathrm{ocr}(G', \pi')$. Introducing $c := \mathrm{ocr}(G', \pi')$ allows us to summarize the discussion as

$$c/(\binom{k+4}{4}/5) \leq \mathrm{ocr}(G, \pi) \leq c.$$

Since $\mathrm{ocr}(G, \pi) \leq \mathrm{cr}(G, \pi) \leq \mathrm{ocr}(G, \pi)\binom{k+4}{4}/5$, we conclude that

$$c/(\binom{k+4}{4}/5) \leq \mathrm{cr}(G, \pi) \leq c\binom{k+4}{4}/5.$$

Now c can be computed in time $O(m^2)$ using the trivial algorithm for one-vertex graphs if we know G' and π'. The only remaining problem is that we do not know the sequence of edges that determines G' and π'. Hence we have to try all possible sequences, giving a running time of $O(m^{k+2})$.

References

1. Archdeacon, D.: Problems in topological graph theory (accessed Septmeber 15, 2006), `http://www.emba.uvm.edu/~archdeac/problems/npcubic.htm`
2. Bokal, D., Fijavž, G., Mohar, B.: Minor-monotone crossing number. In: Felsner, S. (ed.) EuroComb 2005. Discrete Mathematics and Theoretical Computer Science, vol. AE, pp. 123–128 (2005)
3. Buchheim, C., Jünger, M., Menze, A., Percan, M.: Directed crossing minimization. Technical report, Zentrum für Angewandte Informatik Köln, Lehrstuhl Jünger (August 2005)
4. Diaconis, P., Graham, R.L.: Spearman's footrule as a measure of disarray. J. Roy. Statist. Soc. Ser. B 39(2), 262–268 (1977)
5. Garey, M., Johnson, D.: Crossing number is NP-complete. SIAM Journal on Algebraic and Discrete Methods 4, 312–316 (1983)
6. Hliněný, P.: Crossing number is hard for cubic graphs. J. Combin. Theory Ser. B 96(4), 455–471 (2006)
7. Kedem, Z.M., Fuchs, H.: On finding several shortest paths in certain graphs. In: 18th Allerton Conference, pp. 677–686 (1980)
8. Knuth, D.E.: The art of computer programming. In: Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing, vol. 3, Addison-Wesley Publishing Co., Reading (1973)
9. Lowrance, R., Wagner, R.A.: An extension of the string-to-string correction problem. J. Assoc. Comput. Mach. 22, 177–183 (1975)
10. Maes, M.: On a cyclic string-to-string correction problem. Inform. Process. Lett. 35(2), 73–78 (1990)
11. Marzal, A., Barrachina, S.: Speeding up the computation of the edit distance for cyclic strings. In: International Conference on Pattern Recognition, pp. 891–894 (2000)

12. Štefankovič, D., Pelsmajer, M.J., Schaefer, M.: Removing even crossings. In: Felsner, S. (ed.) EuroComb 2005, DMTCS Proceedings. Discrete Mathematics and Theoretical Computer Science, vol. AE, pp. 105–110 (2005)
13. Pelsmajer, M.J., Schaefer, M., Štefankovič, D.: Odd crossing number is not crossing number. In: Healy, P., Nikolov, N.S. (eds.) GD 2005. LNCS, vol. 3843, pp. 386–396. Springer, Heidelberg (2006)
14. Pelsmajer, M.J., Schaefer, M., Štefankovič, D.: Removing even crossings. J. Combin. Theory Ser. B (to appear)
15. Wagner, R.A.: On the complexity of the extended string-to-string correction problem. In: Robert, A. (ed.) Seventh Annual ACM Symposium on Theory of Computing, Albuquerque, N.M., Assoc. Comput. Mach., New York, pp. 218–223 (1975)