Crossing Minimization for Symmetries [⋆]

Christoph Buchheim[1] and Seok-Hee Hong[2]

[1] Institut für Informatik, Universität zu Köln, Germany
buchheim@informatik.uni-koeln.de
[2] School of Information Technologies, University of Sydney, Australia
shhong@it.usyd.edu.au

Abstract. We consider the problem of drawing a graph with a given symmetry such that the number of edge crossings is minimal. We show that this problem is NP-hard, even if the order of orbits around the rotation center or along the reflection axis is fixed. Nevertheless, there is a linear time algorithm to test planarity and to construct a planar embedding if possible. Finally, we devise an $O(m \log m)$ algorithm for computing a crossing minimal drawing if inter-orbit edges may not cross orbits, showing in particular that intra-orbit edges do not contribute to the NP-hardness of the crossing minimization problem for symmetries. From this result, we can derive an $O(m \log m)$ crossing minimization algorithm for symmetries with an orbit graph that is a path.

1 Introduction

In Automatic Graph Drawing, the display of symmetries is one of the most desirable aims [21]. Each symmetry displayed in the drawing of a graph reduces the complexity of the drawing for the human viewer; see Fig. 1. Furthermore, symmetric drawings are regarded as aesthetically pleasing in general. Finally, the existence of a symmetry can point to an important structural property of the data being displayed.

The usual approach for drawing graphs symmetrically consists of two steps: first, try to find an abstract symmetry, i.e., an automorphism of the graph that can be represented by a symmetric drawing in two or three dimensions; second, compute such a drawing, while trying to keep an eye on other criteria characterizing a nice and readable graph layout. The symmetry detection problem to be solved in the first step is NP-hard in general [18]. Nevertheless, exact algorithms for general graphs have been devised in [3] and [1]. In planar graphs, planar symmetries can be detected in linear time [12–15].

In this paper, we focus on the second step and on two-dimensional drawings. More specifically, we consider the problem of minimizing the number of edge crossings over all two-dimensional drawings displaying a given symmetry π. Obviously, the aims of displaying π and minimizing the number of edge crossings interfere with each other; see Fig. 1 again. We always assume that the symmetry π has to be displayed in any case. Finally observe that we do not consider the case of two symmetries that can be displayed simultaneously.

[⋆] The second author is supported by a grant from the Australian Research Council. This paper was partially written when the first author was visiting the University of Sydney.

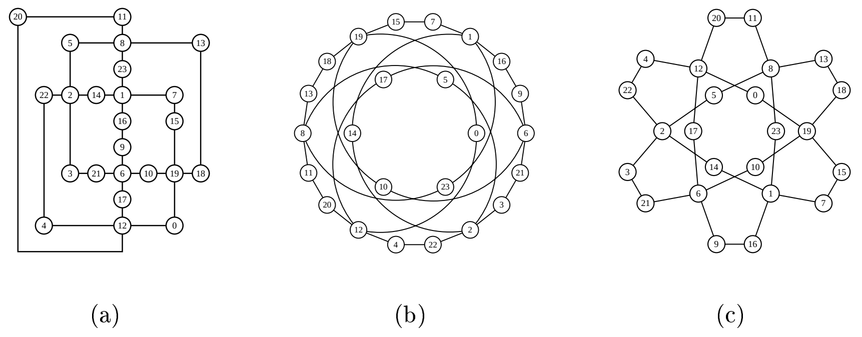


Fig. 1. Three drawings of the same abstract graph. The drawing in (a) is planar but does not display any symmetry. In (b) and (c), the same symmetries are displayed; the number of edge crossings in (b) is 18, while the number of edges crossings in (c) is 6, which is minimal for the given symmetries

As far as we know, the problem of crossing minimization for symmetries has not been examined before, so we first introduce some new notation in Section 2. In Section 3, we show that crossing minimization for symmetries is NP-hard. Next, we show that planarity can be tested in linear time; see Section 4. In Section 5, we investigate the impact of intra-orbit edges on complexity. In Section 6, we give some ideas of how the results presented in this paper can be used in order to develop heuristic crossing minimization algorithms for symmetries. Section 7 concludes.

2 Preliminaries

In the following, a *graph* is always a simple graph with n nodes and m edges. For runtime evaluations, we assume $m \geq n$. A *drawing* of a graph is a representation of its nodes by different points of the plane and of its edges by arbitrary curves in the plane that connect the corresponding points but do not traverse any other node point. An *edge crossing* in a drawing of a graph is a crossing of two curves.

A *symmetry* or *geometric automorphism* of a graph G is a permutation of the nodes of G that is induced by a symmetric drawing of G, i.e., by a drawing of G that is fixed by some non-trivial isometry of the plane. By [7], there are two different types of symmetries: a *reflection* or *axial symmetry* is a symmetry induced by a reflection of the plane at an axis, the *reflection axis*; a *rotation* is a symmetry induced by a rotation of the plane around a center point, the *rotation center*. The *order* of a symmetry π is the smallest positive integer k such that π^k equals the identity. In particular, all reflections have order one or two. For a node v, the *orbit* of v under π consists of the nodes $\pi^k(v)$ for all integers k; the orbit of an edge is defined analogously. An edge is called *intra-orbit edge* if the nodes connected by this edge belong to the same orbit, otherwise it is called *inter-orbit edge.* A *diagonal* of a rotation π of even order k is an edge $(v, \pi^{k/2}(v))$. For technical reasons, we consider diagonals as inter-orbit edges in this paper. A node v of G is a *fixed node* of π if $\pi(v) = v$.

The *orbit graph* G/π of π is the graph resulting from G when identifying each node $v \in V$ with its image $\pi(v)$ and deleting multiple edges and loops afterwards. Hence every node of G/π represents a π-orbit of G and every edge of G/π corresponds to a set of orbits of inter-orbit edges of G; see Fig. 2.

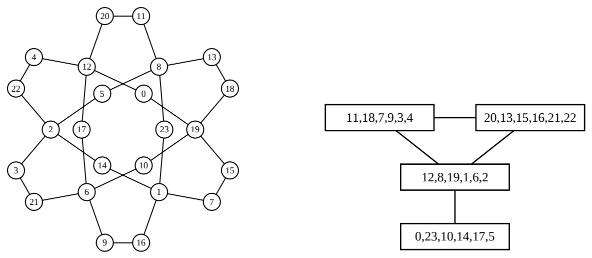


Fig. 2. A rotation of order six and its orbit graph

A *drawing* of a symmetry π of G is a drawing of G such that π is induced by an isometry of the plane fixing this drawing. In any drawing of a rotation π, every curve representing a diagonal has to cross the rotation center. By adding a fixed node to π, if none exists, and by splitting up every diagonal at the fixed node, we may assume throughout this paper that the considered rotations never have diagonals. For every drawing of a rotation of order k, there is an integer $e \in \{1, \ldots, k\}$ with $\gcd(e, k) = 1$ such that rotating the drawing by $360/k$ degrees in clockwise order around the rotation center maps each node v to $\pi^e(v)$. We call e the *exponent* of the drawing; see Fig. 3. The exponent of a reflectional drawing is always one. Different exponents correspond to what is called different *representations* in [1].

In a drawing of a rotation, all nodes of a common orbit have the same distance to the rotation center; the corresponding circle is called the *orbit circle.* For a drawing of a reflection, the nodes of a common orbit lie on the same line orthogonal to the reflection axis; this line is called the *orbit line.*

A graph is *planar* if it has a *plane drawing*, i.e., a two-dimensional drawing without edge crossings. A symmetry π of a graph G is *planar* if there is a drawing of π that is planar as a drawing of G. We call a drawing of a symmetry *loopless* if all orbit circles or lines are distinct and if no edge crosses more orbit circles or lines than necessary. More precisely, in a loopless drawing of a rotation, every curve connecting two orbits may only cross the circles of the orbits in-between, and each only once; see Fig. 4. Analogously, in a loopless drawing of a reflection, every curve connecting two orbits may only cross the lines of orbits in between,

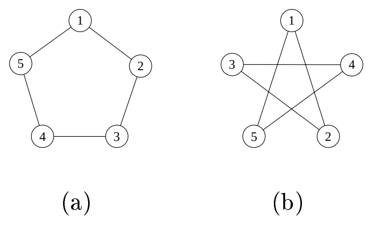


Fig. 3. Two drawings displaying the same symmetry $\pi = (12345)$. The exponent of the planar drawing (a) is one; the exponent of the non-planar drawing (b) is three

and each only once. In a loopless drawing, we add a *dummy* at each crossing of an edge with an orbit circle or line; see Fig. 4(a) again.

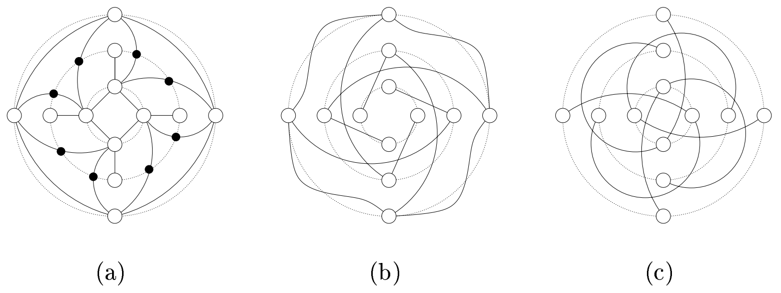


Fig. 4. The drawing (a) is loopless, the dummies are represented by filled circles. In both drawings (b) and (c), all edges are loops

In this paper, we will mainly consider the following problems:

Problem (SCM): Given a symmetry π, compute a drawing of π with a minimal number of edge crossings.

Problem (SCM+): Given a symmetry π, compute a loopless drawing of π with a minimal number of edge crossings.

Problem (SCM+1): Given a symmetry π and a fixed order of its orbits, compute a loopless drawing of π respecting this order such that the number of edge crossings in this drawing is minimal.

Problem (SCM+2): Given a symmetry π, a fixed order of its orbits, and a fixed order of nodes and dummies on each orbit circle or orbit line, compute a loopless drawing of π respecting these orders such that the number of edge crossings in this drawing is minimal.

3 Crossing Minimization for Symmetries is NP-hard

In the following, we prove that the problems (SCM), (SCM+), and (SCM+1) defined in the last section are NP-hard. In fact, the proofs show that the crossing minimization problem for symmetries can be considered as a generalization of the crossing minimization problem for graphs. To prepare these results, first observe

Lemma 1. *Every planar symmetry admits a planar loopless drawing.*

Proof. We can remove each loop in a planar drawing by pulling it out of the forbidden area, dragging along the nodes inside the loop; see Fig. 5 for an illustration. Of course, this has to be done in a symmetrical way. □

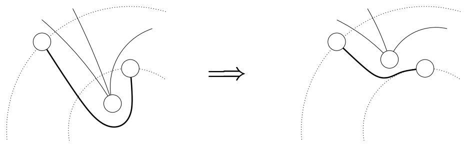


Fig. 5. Removing a loop

Lemma 2. *The problem (SCM) can be reduced to (SCM+) in $O(m^2)$ time, also if both problems are restricted to reflections or rotations of fixed order.*

Proof. Let π be a symmetry of G. Add m new nodes on each edge of G and let G' be the resulting graph. Let π' be the symmetry of G' induced by π in the obvious way. Then we claim that

(*) for every drawing of π, there is a loopless drawing of π' with the same number of edge crossings.

Hence a crossing minimal loopless drawing of π' gives rise to a crossing minimal general drawing of π by removing the new nodes. In other words, allowing m changes of direction for each edge of G suffices to allow drawing π with the minimum number of edge crossings. Thus, to solve (SCM) for π, it suffices to construct π' and to solve (SCM+) for π'.

To prove (*), consider any rotational drawing D of π; in the reflectional case, one can argue analogously. We may assume that each pair of edges has at most one crossing. In order to construct a loopless drawing of π' with the same number of edge crossings as D, we first add a virtual node on each edge crossing of D, thus creating a planar symmetry π^*; see Fig. 6(b). By Lemma 1, there is a planar loopless drawing D^* of π^*; see Fig. 6(c). We remove the virtual nodes from D^*;

for each non-fixed virtual node, we add a new node on each of the two edges involved by the crossing represented by this virtual node, both placed close to the former position of the virtual node. By this, we get a loopless drawing D'; see Fig. 6(d). Since each original edge of G contains at most m new nodes, we can easily transform D' into a loopless drawing of π' with the same number of edge crossings as in D. □

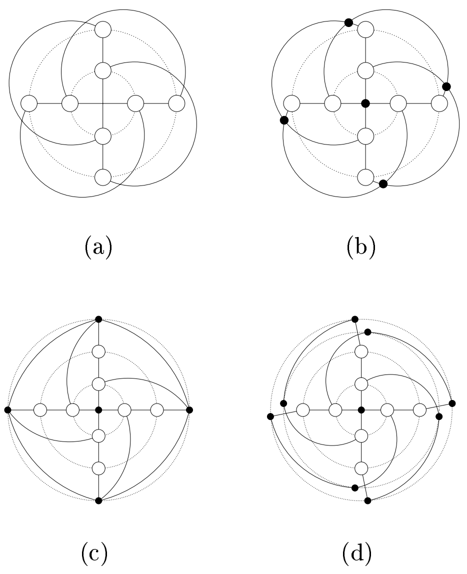


Fig. 6. Transforming a drawing of π into a loopless drawing of π'

Observe that these transformations do not have to be performed by any algorithm presented in this paper. They just show the existence of a loopless drawing of π' with the required number of edge crossings and hence the correctness of the reduction algorithm from (SCM) to (SCM+).

We do not know whether (*) still holds if π' arises from π by adding a constant number of new nodes per edge. If so, the problem (SCM) could be reduced to (SCM+) in linear time.

Theorem 1. *The problems (SCM) and (SCM+) are NP-hard, even if restricted to reflections or rotations of fixed order.*

Proof. We can easily reduce the NP-hard problem of crossing minimization for graphs [10] to the crossing minimization problem for reflections or rotations of fixed order k. For this, let G be any graph. Construct a new graph G' as the disjoint sum of k copies of G. Define π by mapping the copies cyclically to each other. Obviously, drawing π with a minimal number of edge crossings is equivalent to drawing G' without intersections between the copies of G and drawing the copies of G with a minimal number of edge crossings. Hence an optimal drawing of π according to (SCM) gives rise to a drawing of G with a minimal number of edge crossings. Finally, the NP-hardness of (SCM+) follows from Lemma 2. $\square$

Theorem 2. *The problem (SCM+1) is NP-hard, even if restricted to reflections or rotations of fixed order.*

Proof. We reduce the NP-hard fixed linear crossing minimization problem [20] to (SCM+1). Consider an instance of this problem, given by an ordered sequence of nodes $(v_1, \dots, v_n)$ and a set of edges E. Define an instance for (SCM+1) as follows: start with $G = (\{v_1, \dots, v_n\}, E)$ and add nodes $v_{i,j}$ for $i = 1, \dots, n-1$ and $j = 1, \dots, m^2$. Define a node order by $v_i < v_{i,j} < v_{i+1}$ and $v_{i,j} < v_{i,j+1}$. Add new edges $(v_i, v_{i,j})$ and $(v_{i,j}, v_{i+1})$ for each $i = 1, \dots, n-1$ and $j = 1, \dots, m^2$. Finally, define G' and π as in the proof of Theorem 1.

Now consider an optimal drawing of π according to (SCM+1) with the given order of orbits. We claim that the center of the rotation lies outside of each component of G' then. Indeed, the new edges would induce at least m^2 edge crossings otherwise, but this is not possible since a drawing with less than m^2 edge crossings obviously exists. By the same reason, no edge crosses the new edges of its own component. In summary, the solution of (SCM+1) gives rise to a solution of the fixed linear crossing minimization problem. $\square$

We do not know whether problem (SCM+2) is NP-hard in general. If no dummies exist, (SCM+2) can be solved in $O(m \log m)$ time by Theorem 4 below.

4 Testing Planarity of Symmetries in Linear Time

We now consider a more promising candidate:

Problem (SPL): Given a symmetry π, decide whether there is a drawing of π without edge crossings. If there is such a drawing, compute one.

Lemma 3. *Every planar symmetry admits a planar straight-line drawing.*

Proof. Let π be a planar symmetry of G. If G is triconnected, the result follows from Mani's Theorem as in the proof of Lemma 3.1 in [15]. Otherwise, it is easy to see that there is a triconnected graph G' with a planar symmetry π' such that G is a subgraph of G' and π agrees with π' on G. Since π' admits a planar straight-line drawing, the same is true for π. $\square$

By Lemma 3, planarity of symmetries does not depend on whether we require straight lines or not. This is not true for the crossing number of a symmetry, which follows from the corresponding result for graphs [2] but is obvious even for symmetries with a single orbit; see Fig. 7.

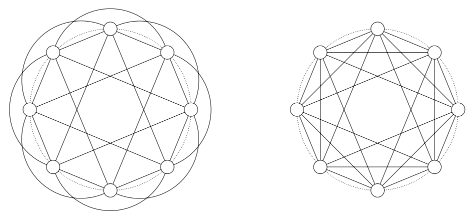


Fig. 7. Intra-orbit edges may be drawn inside or outside their orbit circle. In general, requiring all edges to be drawn on the same side, for example in straight-line drawings, increases the number of necessary edge crossings

Theorem 3. *The problem (SPL) can be solved in $O(n)$ time.*

Proof. The outline of the following algorithm to check planarity and to create planar drawings of symmetries is similar to the algorithm presented in [12–15]. In particular, we also consider the triconnected, biconnected, one-connected, and disconnected cases one after another. Observe that only straight-line drawings are allowed in [12–15], but Lemma 3 shows that this is equivalent to the definition used here.

Fix a non-trivial symmetry π of an arbitrary graph G. First we check G for planarity in linear time [16]. In case of a negative result, we know that π is not planar, so in the following we may assume that G is planar and hence $m \in O(n)$.

Triconnected case. If G is triconnected, the result follows from Lemma 3.1 in [15], stating that in this case π is planar if and only if a face of the unique combinatorial embedding of G is fixed. In the affirmative case, a linear time algorithm to draw π is given by Theorem 4.1 in [15].

Biconnected case. If G is biconnected, we distinguish two cases, the order of π being two or at least three. In the latter case, the symmetry π is a rotation of G, and we consider the SPQR-tree T of G, see [6]. It is easy to see that π induces an automorphism $\overline{\pi}$ of T. We claim that π is planar if and only if exactly one node v of T is fixed by $\overline{\pi}$ and the restriction of π to the skeleton of v is planar. These conditions are sufficient, since they allow to construct a planar drawing of π as follows: first we compute a planar drawing of the restriction of π to the skeleton of v. If v is an R-node, we use the triconnected case above, otherwise this is trivial. Then we replace all virtual edges in the skeleton of v by planar drawings of the corresponding subgraphs in a symmetric way; see Fig. 8. This is possible since G is planar and no edge of the skeleton of v is fixed as the order of π is at least three. On the other hand, both conditions are necessary: it is easy to see that if there are either two nodes of T fixed by $\overline{\pi}$ or none, then there is a separation pair of G the nodes of which are either fixed or exchanged by π, hence the order of π is two. Furthermore, the restriction of the planar rotation π to any fixed skeleton is obviously planar.

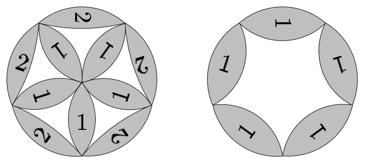


Fig. 8. Reduction to triconnected graphs, R-node and S-node

Now let the order of π be two. In this case, the situation is more complicated because of a possible enclosing composition, see [12]. To avoid dealing with this complication, we make use of Theorem 1 in [12]: there is a linear time algorithm that constructs maximally symmetric planar drawings of biconnected planar graphs. This remains true if given node colors have to be respected. We reduce Theorem 3 to this result in the following way: we assign a different color to each node orbit of π. Then we claim that π is the only possible non-trivial planar symmetry of the colored graph G. Hence, if π is planar, the computed group of symmetries will be generated by π; otherwise, it will be trivial.

Let π' be any non-trivial planar symmetry of the colored graph G. We show that π' equals π. By the choice of colors, it suffices to show that each node fixed by π' is also fixed by π. This is trivial if π' admits a planar rotational drawing, since otherwise π' had at least two fixed nodes. If π' admits a planar reflectional drawing, consider any non-fixed node v of π. Since G is biconnected, there is a cycle of G containing both v and $\pi(v)$. If v was fixed by π', the same would follow for all nodes of this cycle, but a cycle cannot be fixed by a reflection.

One-connected case. Next, let G be a one-connected graph. By the planarity of G, the symmetry π is planar if and only if its restriction to each fixed block of the connected graph G is planar and there is at most one fixed block if π is a rotation; compare Theorem 10 in [13]. The combined drawing is constructed as follows: if π is a rotation, we first draw the fixed block, if any, as explained for the biconnected case above; then we add planar drawings of the remaining blocks symmetrically; see Fig. 9(a). If π is a reflection, we first draw all fixed blocks as in the biconnected case, placing all fixed cut-nodes on the axis, then we add the remaining blocks symmetrically again; see Fig. 9(b).

Disconnected case. Now let G be disconnected. It is easy to see that the symmetry π is planar if and only if its restriction to each fixed connected component is planar with the same exponent e, and a drawing of π can easily be constructed by drawing the fixed components around each other or along the axis; see Fig. 10(a) for rotations and Fig. 10(b) for reflections.

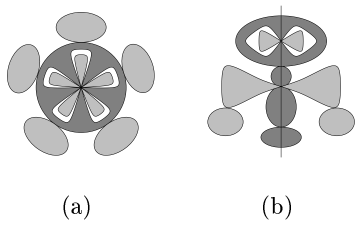


Fig. 9. Reduction to biconnected graphs; the fixed blocks are shaded

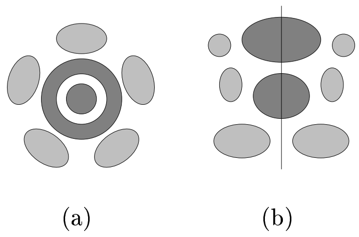


Fig. 10. Reduction to connected graphs; the fixed components are shaded

The computation of the exponents e is crucial; see Fig. 11. Let G' be any component and let π' be the restriction of π to G'. We claim that either there exists an integer e such that any planar drawing of π' has exponent e or $-e$, or there is a planar drawing for any integer exponent; furthermore, all possible exponents can be determined and a planar drawing for an eventual common exponent can be constructed in linear time.

If π' is a reflection, the exponent is one. If π' is a rotation and G' is unconnected after removing an eventual fixed node, the exponent is arbitrary, and a drawing with any given exponent can be produced by permuting the components in the drawing computed above according to this exponent. Finally, if G' is still connected after removing fixed nodes, let v be any non-fixed node of G'. Consider a path from v to $\pi'(v)$ without fixed nodes and let ${\pi'}^{e}(v)$ be the first node after v on this path that belongs to the orbit of v. It is easy to see that e and $-e$ are the only possible exponents of a planar drawing of π' then. □

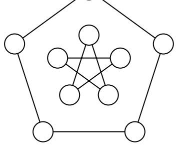


Fig. 11. A non-planar rotation with planar components

5 Crossing Minimization and Intra-Orbit Edges

In Section 3, we showed that the symmetric crossing minimization problems (SCM), (SCM+), and (SCM+1) are NP-hard. In the reductions of the proofs, we did not make use of intra-orbit edges at all. These problems thus remain NP-hard even for symmetries containing only inter-orbit edges. In this section, we show that the corresponding statement for intra-orbit edges is not true. More generally, we show that (SCM+2) can be solved in $O(m \log m)$ time if no dummies exist. Since no information about intra-orbit edges is given in (SCM+2), we derive that intra-orbit edges do not make crossing minimization significantly harder. Before showing these statements, we list two auxiliary results, the proofs of which are given in Appendix A and B.

Lemma 4. *Let k be a positive integer and $(a_1, \ldots, a_r)$ a sequence of integers. Let $I = \{1, \ldots, r\}$. Then an integer sequence $(c_1, \ldots, c_r)$ minimizing the sum*

$$\sum_{\substack{i,j \in I \\ i<j}} |a_j + c_j k - a_i - c_i k|$$

as well as the minimal objective value can be computed in $O(r \log r)$ time.

Lemma 5. *Let I be a set of r non-negative integers and $t_1, t_2 \geq 0$. Then a partition $I = I_1 \oplus I_2$ minimizing*

$$\sum_{\substack{i,j \in I_1 \\ i<j}} i + \sum_{\substack{i,j \in I_2 \\ i<j}} i + t_1 \sum_{i \in I_1} i + t_2 \sum_{i \in I_2} i$$

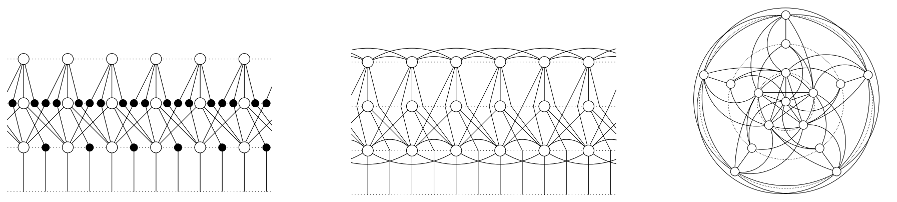
as well as the minimal objective value can be computed in $O(r \log r)$ time.

Theorem 4. *The problem (SCM+2) can be solved in $O(m \log m)$ time for symmetries without dummies. The corresponding number of edge crossings can be computed in $O(m/k \cdot \log m)$ time, where k is the order of the symmetry.*

Proof. First note that this statement is trivial for reflections. Hence for the rest of this proof we consider a rotation π of order k. Consider the plane polar coordinate function φ mapping a point (α, d) in the plane to $(d \sin \alpha, d \cos \alpha)$. For simplicity, we will not construct the required drawing D directly but a drawing D' such that $\varphi(D') = D$; see Fig. 12.

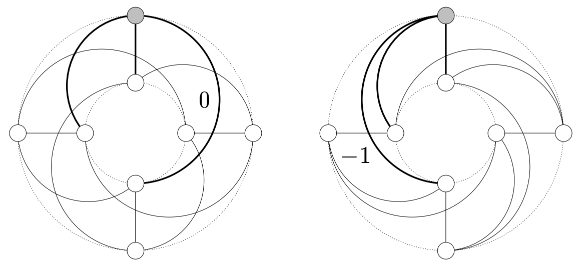
First observe that the placement of nodes is essentially determined by the data given in (SCM+2): the α-th node on the d-th orbit is placed to the point $(2\pi\alpha/k, d)$, where we count orbits from inside to outside, the number of the first non-trivial orbit being one and the number of an eventual fixed orbit being zero.

For drawing edges, we first consider the inter-orbit edges independently. If an edge e connects two nodes placed to (α_1, d_1) and (α_2, d_2), the optimal way to draw e is by a straight line. However, e does not have to connect (α_1, d_1) to (α_2, d_2) but may connect it to $(\alpha_2 + 2\pi c, d_2)$ for any integer c. This corresponds to



(a) Inter-orbit edges (b) Intra-orbit edges (c) Result

Fig. 12. Constructing an optimal drawing



(a) (b)

Fig. 13. Different wrapping coefficients c induce different numbers of edge crossings. The number of crossings in (a) is 12, the number of crossings in (b) is 4

the possibility of wrapping e around the inner orbit circle. The resulting number of edge crossings is influenced by the choice of c in general; see Fig. 13. Below we explain how to choose the wrapping coefficients c optimally.

After drawing the inter-orbit edges, we consider intra-orbit edges. For an intra-orbit edge e connecting (α_1, d_1) and (α_2, d_2), the optimal coefficient c is obviously the one minimizing $|\alpha_2 - \alpha_1 + 2\pi c|$. But now we have to decide for each orbit of intra-orbit edges whether its edges are drawn inside or outside the orbit circle; see Fig. 7. The algorithm to decide this is explained below. For a given partition into inside and outside edges, we proceed as follows: the edge e is drawn as an arc. The angle of the arc when leaving a node is the same for all arcs; it is equal to the minimum angle of a line representing an inter-orbit edge incident to the same node, but small enough to ensure that the distance of the longest arc from the line of the orbit is at most $1/3$. If e is chosen to be an inside edge, the corresponding arc is drawn below the orbit line, otherwise above.

In summary, for fixed wrapping coefficients and a fixed partition into inside and outside edges, all crossings involving intra-orbit edges are unavoidable by the data given in (SCM+2). In particular, the optimal partition into inside and outside edges depends only on the given orders but not on the drawing of the inter-orbit edges computed before. Since the number of crossings between inter-orbit edges in the drawing created by this algorithm is minimal, this justifies our strategy to consider inter-orbit edges and intra-orbit edges separately. So to prove the first statement of Theorem 4, it remains to compute optimal wrapping coefficients and an optimal partition of intra-orbit edges into inside and outside edges in $O(m \log m)$ time.

For the first part, we may assume that π has exactly two orbits, and that both orbits have k nodes, since if the inner orbit contains only one fixed node, the problem of determining optimal wrapping coefficients is trivial. First we explain how to count the number of inter-orbit edge crossings depending on the choice of wrapping coefficients. Let v be any node on the outer orbit and number the nodes on the inner orbit $0, \ldots, k-1$ according to the order given in (SCM+2), starting at any node. Let $e_1, \ldots, e_r$ be the inter-orbit edges incident to v, sorted by ascending x-coordinate of the corresponding node on the inner orbit. Let $I = \{1, \ldots, r\}$. Note that each orbit of inter-orbit edges has exactly one representing edge in $\{e_1, \ldots, e_r\}$; hence $r \leq m/k$. For $i \in I$, let a_i be the number of the node adjacent to e_i on the inner orbit, and let c_i be the unique integer such that the x-coordinate of a_i minus the x-coordinate of the node with number 0 is contained in $[2\pi c_i, 2\pi(c_i+1))$. Then for $i < j$ the number of edge crossings between the orbits of e_i and e_j is

$$k(|a_j + c_j k - a_i - c_i k| - 1) \ .$$

Since inter-orbit edges of the same orbit do not cross each other, the total number of crossings between inter-orbit edges is

$$k \sum_{\substack{i,j \in I \\ i<j}} |a_j + c_j k - a_i - c_i k| - k \cdot \#\{i, j \in I \mid i < j\} \ .$$

By Lemma 4, the optimal coefficients c_i can be computed in $O(m/k \cdot \log m)$ time.

Next we determine the optimal partition of intra-orbit edges into inside and outside edges. For this, consider a fixed orbit p. Assume that the nodes on p are numbered $0, 1, \ldots, k-1$ according to the order given in (SCM+2). Depending on a given partition, we can compute the number of edge crossings involving intra-orbit edges of orbit p as follows: consider the set

$$I = \{i \in \{1, 2, \ldots, \lfloor k/2 \rfloor\} \mid (0, i) \in E\} \ .$$

Let $I = I_1 \oplus I_2$ be the partition of I representing the chosen partition of edges into inside and outside edges; i.e., let the set $\{(0, i) \mid i \in I_1\}$ contain exactly one representing edge for each orbit of inside edges. Observe that the total sum of elements of I for all orbits is at most m/k, since all edge orbits contain exactly k edges (recall that we excluded diagonals). Furthermore, the number of crossings between inside edges is given by the sum of the numbers of crossings between the orbits of $(0, i)$ and $(0, j)$ for $i, j \in I_1$ and $i \leq j$ (the case $i = j$ corresponds to crossings between edges of the same inside edge orbit). In an optimal drawing as described above, this crossing number for i and j is $2k(i-1)$. So the total number of crossings between inside edges is

$$\sum_{\substack{i,j\in I_1\\ i\leq j}} 2k(i-1)\ .$$

The situation for outside edges is symmetric. Additionally, we have

$$\sum_{i\in I_1} kt_1(i-1) + \sum_{i\in I_2} kt_2(i-1)$$

crossings between an intra-orbit edge of p and inter-orbit edges incident to p, where t_1 is the number of edges connecting node 0 with an orbit inside of p and t_2 is the number of edges connecting node 0 with an orbit outside of p. So the total number of crossings involving at least one intra-orbit edge is

$$2k\left(\sum_{\substack{i,j\in I_1\\ i<j}}(i-1) + \sum_{\substack{i,j\in I_2\\ i<j}}(i-1) + \frac{t_1}{2}\sum_{i\in I_1}(i-1) + \frac{t_2}{2}\sum_{i\in I_2}(i-1)\right) + 2k\sum_{i\in I}(i-1)$$

and can be minimized by Lemma 5 after decreasing each element of I by one. The total runtime for all orbits is $O(m/k\cdot\log m)$.

It remains to explain how to compute the total number of edge crossings in $O(m/k\cdot\log m)$ time. This number is given as the sum of the two formulas derived above for the number of crossings between inter-orbit edges and the number of crossings involving intra-orbit edges. Hence it can be computed in $O(m/k\cdot\log m)$ time by Lemma 4 and 5. □

Corollary 1. *The problem (SCM+) can be solved in $O(m\log m)$ time if the orbit graph of π is a path.*

Proof. Obviously, the optimal order of orbits is one of the two orders given by the path; for rotations with a fixed node, this node has to be the innermost orbit. Hence it suffices to show that (SCM+1) can be solved in $O(m\log m)$ time if all inter-orbit edges connect neighboring orbits. Let k be the order of the symmetry to be drawn. Observe that there are at most k possible orders of nodes and dummies on the orbits, since we have no dummies and the order of nodes on the orbits is determined by the exponent of the drawing. By Theorem 4, we can compute the minimal number of edge crossings for each possible exponent in $O(m/k\cdot\log m)$, since after fixing the exponent we have an instance of (SCM+2) without dummies. Hence we can determine the best exponent in $O(m\log m)$ time. Finally, we can compute an optimal drawing for this exponent in $O(m\log m)$ time by Theorem 4 again. □

Corollary 2. *The problem (SCM) can be solved in $O(m\log m)$ time for symmetries without inter-orbit edges.*

Observe that Corollary 1 and Lemma 2 imply that the problem (SCM) can be solved in $O(m^2\log m)$ time if the orbit graph of π is a path.

6 Heuristic Crossing Minimization for Symmetries

From the results given in the last section, we can derive some ideas for a first approach to heuristic crossing minimization for symmetries. Corollary 1 suggests the following algorithmic framework for problem (SCM+); by Lemma 2, this also yields a heuristic for the general problem (SCM):

1. Test planarity of π by Theorem 3. If π is planar, draw it without crossings by Theorem 3 and stop.
2. Compute the orbit graph G/π.
3. Compute a layering of G/π with only one node per layer, such that the resulting number of dummy nodes is small.
4. For every edge in G/π connecting non-neighboring layers, delete the corresponding edge orbits in G. Let π' be the resulting symmetry.
5. The orbit graph of π' is a path; draw it optimally using Corollary 1.
6. Reinsert the deleted edge orbits one after another, each one optimally.

In step 3, we have an optimal linear arrangement problem, since the number of dummies equals the total length of all edges minus m. This problem is NP-hard [9], but many heuristic and exact algorithms have been devised; see [17] for an overview. In step 6, we can add a single edge optimally in linear time by searching for a shortest path in the dual graph. After that, we can insert all other edges of the same orbit symmetrically.

This algorithmic framework is similar to the one used for the planarization method for graphs. In both methods, edges of the graph are removed until the resulting graph or symmetry is tractable. After solving the reduced instance, the deleted edges are reinserted. However, note that we do not have to delete edges until the symmetry is planar but only until Corollary 1 is applicable. Finally note that the number of edges in the orbit graph G/π is at most m/k, where k is the order of π. Loosely speaking, the more symmetry we have, the smaller G/π is, and the fewer edges we have to delete in step 4.

7 Conclusion

In this paper, we started the discussion of crossing minimization for graphs with a given symmetry. This is a generalization of the usual crossing minimization problem for graphs and hence a hard problem both theoretically and practically. The presented results allow to restrict the attention to symmetries without intra-orbit edges, since these do not increase the complexity significantly. In the case of a single orbit we have a crossing minimization problem that is interesting on its own and that can be solved in $O(m \log m)$ time.

This is a first approach to symmetric crossing minimization, so there is much left for future research; heuristic crossing minimization algorithms, for example in the framework of Section 6, have to be developed and evaluated. Furthermore, all results should be generalized to the case where not only one single symmetry can be considered but also two symmetries that can be displayed simultaneously (the dihedral case). Finally, it is open how much exactly has to be fixed for the drawing of a symmetry to make the crossing minimization problem polynomial time solvable.

References

1. D. Abelson, S. Hong, and D. Taylor. A group-theoretic method for drawing graphs symmetrically. Technical Report IT-IVG-2002-01, University of Sydney, 2002.
2. D. Bienstock and N. Dean. Bounds for rectilinear crossing numbers. *Journal of Graph Theory*, 17:333–348, 1993.
3. C. Buchheim and M. Jünger. Detecting symmetries by branch & cut. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Graph Drawing 2001*, volume 2265 of *Lecture Notes in Computer Science*, pages 178–188. Springer-Verlag, 2001.
4. H. Carr and W. Kocay. An algorithm for drawing a graph symmetrically. *Bulletin of the ICA*, 27:19–25, 1999.
5. G. Di Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing - Algorithms for the Visualization of Graphs.* Prentice Hall, 1999.
6. G. Di Battista and R. Tamassia. On-line maintenance of triconnected components with SPQR-trees. *Algorithmica*, 15:302–318, 1996.
7. P. Eades and X. Lin. Spring algorithms and symmetry. *Theoretical Computer Science*, 240(2):379–405, 2000.
8. H. de Fraysseix. An heuristic for graph symmetry detection. In J. Kratochvíl, editor, *Graph Drawing 1999*, volume 1731 of *Lecture Notes in Computer Science*, pages 276–285. Springer-Verlag, 1999.
9. M. Garey and D. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.
10. M. Garey and D. Johnson. Crossing number is NP-complete. *SIAM Journal on Algebraic and Discrete Methods*, 4:312–316, 1983.
11. S. Hong. Drawing graphs symmetrically in three dimensions. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Graph Drawing 2001*, volume 2265 of *Lecture Notes in Computer Science*, pages 189–204. Springer-Verlag, 2001.
12. S. Hong and P. Eades. Drawing planar graphs symmetrically II: Biconnected graphs. Technical Report CS-IVG-2001-01, University of Sydney, 2001.
13. S. Hong and P. Eades. Drawing planar graphs symmetrically III: Oneconnected graphs. Technical Report CS-IVG-2001-02, University of Sydney, 2001.
14. S. Hong and P. Eades. Drawing planar graphs symmetrically IV: Disconnected graphs. Technical Report CS-IVG-2001-03, University of Sydney, 2001.
15. S. Hong, B. McKay, and P. Eades. Symmetric drawings of triconnected planar graphs. In *SODA 2002*, pages 356–365, 2002.
16. J. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the Association for Computing Machinery*, 21:549–568, 1974.
17. S. Horton. *The optimal linear arrangement problem: Algorithms and approximation.* PhD thesis, Georgia Institute of Technology, 1997.
18. J. Manning. *Geometric Symmetry in Graphs.* PhD thesis, Purdue University, 1990.
19. S. Masuda, T. Kashiwabara, K. Nakajima, and T. Fujisawa. An NP-hard crossing minimization problem for computer network layout. Technical Report SRC TR 86-80, Electrical Engineering Department and Systems Research Center, University of Maryland, 1986.
20. S. Masuda, K. Nakajima, T. Kashiwabara, and T. Fujisawa. Crossing minimization in linear embeddings of graphs. *IEEE Transactions on Computers*, 39(1):124–127, 1990.
21. H. Purchase. Which aesthetic has the greatest effect on human understanding? In Giuseppe Di Battista, editor, *Graph Drawing '97*, volume 1353 of *Lecture Notes in Computer Science*, pages 248–261. Springer-Verlag, 1997.
22. I. Vrt'o. Crossing numbers of graphs: A bibliography. `www.ifi.savba.sk/~imrich`.

Appendix

A Proof of Lemma 4

In order to prove Lemma 4, we first show that we can restrict our attention to r possibly optimal solutions:

Lemma 6. *If $(c_1, \ldots, c_r)$ is an optimal solution according to Lemma 4, then we have $a_j + c_j k - a_i - c_i k < k$ for all $i, j \in I$.*

Proof. Let $(c_i, \ldots, c_r)$ be any solution with $a_j + c_j k - a_i - c_i k \geq k$ for some $i, j \in I$. Denote $a_i + c_i k$ by i' in the following. Let m' be minimal within all i', $i \in I$, and let M' be maximal. Let $M' - m' \in \{ck, \ldots, (c+1)k - 1\}$, where $c \geq 1$. We will show that either by increasing all c_i with $i' = m'$ by c or by decreasing all c_i with $i' = M'$ by c we get a strictly better solution than $(c_i, \ldots, c_r)$. The change of the objective value for the first operation is

$$-\sum_{i \in I}(i' - m') + \sum_{\substack{i \in I \\ i' > m' + ck}} (i' - m' - ck)$$

$$+ \sum_{\substack{i \in I \\ i' < m' + ck}} (m' + ck - i') - ck\#\{i \in I \mid i' = m'\} ,$$

this equals

$$\begin{aligned}
&-\sum_{i \in I}(i' - m') - \sum_{\substack{i \in I \\ i' < m' + ck}} (i' - m') + \sum_{\substack{i \in I \\ i' > m' + ck}} (i' - m') \\
&+ck\#\{i \in I \mid i' < m' + ck\} - ck\#\{i \in I \mid i' > m' + ck\} \\
&-ck\#\{i \in I \mid i' = m'\} \\
= &-2 \sum_{\substack{i \in I \\ i' < m' + ck}} (i' - m') - \sum_{\substack{i \in I \\ i' = m' + ck}} (i' - m') \\
&+ck\#\{i \in I \mid i' < m' + ck\} - ck\#\{i \in I \mid i' > m' + ck\} \\
&-ck\#\{i \in I \mid i' = m'\} \\
= &-2 \sum_{\substack{i \in I \\ i' < m' + ck}} (i' - m') \\
&+ck\#\{i \in I \mid i' < m' + ck\} - ck\#\{i \in I \mid i' \geq m' + ck\} \\
&-ck\#\{i \in I \mid i' = m'\} .
\end{aligned}$$

For the second operation, the change is

$$\begin{aligned}
&-2 \sum_{\substack{i\in I\\ i'>M'-ck}} (M'-i') \\
&+ck\#\{i\in I \mid i'>M'-ck\} - ck\#\{i\in I \mid i'\le M'-ck\} \\
&-ck\#\{i\in I \mid i'=M'\}\ .
\end{aligned}$$

We show that the sum of these changes is negative, so that at least one of the operations improves the solution. The sum is

$$\begin{aligned}
&-2 \sum_{\substack{i\in I\\ i'<m'+ck}} (i'-m') - 2 \sum_{\substack{i\in I\\ i'>M'-ck}} (M'-i') && (1)\\
&+ck\#\{i\in I \mid i'<m'+ck\} + ck\#\{i\in I \mid i'>M'-ck\} && (2)\\
&-ck\#\{i\in I \mid i'\ge m'+ck\} - ck\#\{i\in I \mid i'\le M'-ck\} && (3)\\
&-ck\#\{i\in I \mid i'=m'\} - ck\#\{i\in I \mid i'=M'\}\ . && (4)
\end{aligned}$$

Since $M'-ck \le m'+ck$ and $M'-m' \ge ck$, line (1) is at most

$$-2 \sum_{\substack{i\in I\\ M'-ck<i'<m'+ck}} (M'-m') \le -2ck\#\{i\in I \mid M'-ck<i'<m'+ck\}\ ,$$

while line (2) adds up to

$$\begin{aligned}
&2ck\#\{i\in I \mid M'-ck<i'<m'+ck\} \\
&+ck\#\{i\in I \mid i'\ge m'+ck\} + ck\#\{i\in I \mid i'\le M'-ck\}\ ,
\end{aligned}$$

the second part of which neutralizes line (3). Finally, line (4) is at most $-2ck<0$; thus the total sum is negative. □

Now we can prove Lemma 4. By adding appropriate multiples of k to all a_i, we may assume $a_j - a_i < k$ for $i,j\in I$. By sorting the sequence $(a_1,\dots,a_r)$, we furthermore may assume $a_1 \le a_2 \le \cdots \le a_r < a_1 + k$. By Lemma 6, we only have r candidates for the optimal solution: for every $t\in I$, we have to consider the solution given by

$$c_i^t = \begin{cases} 0 & \text{if } i\ge t\\ 1 & \text{otherwise.}\end{cases}$$

All other solutions are either not optimal by Lemma 6 or equivalent to one of the solutions $(c_1^t,\dots,c_r^t)$, i.e., given by adding a common integer to all c_i^t.

We claim that all objective values for these r solutions can be computed in a total runtime of $O(r)$. The objective value for $(c_1^t,\dots,c_r^t)$ is

$$d_t = \sum_{\substack{i,j\in I\\ t\le i<j}} (a_j-a_i) + \sum_{\substack{i,j\in I\\ i<t\le j}} (a_i+k-a_j) + \sum_{\substack{i,j\in I\\ i<j<t}} (a_j-a_i)\ .$$

We have

$$d_1 = \sum_{\substack{i,j \in I \\ i<j}} (a_j - a_i) \quad \text{and} \quad d_2 - d_1 = -2\sum_{j \in I}(a_j - a_1) + k(r-1) \ .$$

Furthermore, for $t = 1, \ldots, r-2$, we have

$$(d_{t+2} - d_{t+1}) - (d_{t+1} - d_t) = 2r(a_{t+1} - a_t) - 2k \ .$$

Hence Algorithm 1 computes $d_1, \ldots, d_r$ in linear time. $\square$

Algorithm 1: Computing the best wrapping numbers

```
Input: A set $I = \{1, \ldots, r\}$, a positive integer $k$,
        integers $(a_1, \ldots, a_r)$ with $a_1 \le a_2 \le \cdots \le a_r < a_1 + k$.
Output: Integers $(c_1, \ldots, c_r)$ minimizing $\sum_{i,j \in I, i<j} |a_j + c_j k - a_i - c_i k|$
        and the corresponding objective value.
Let change = 0;
Let sum = 0;
Let opt = $\infty$;
foreach $i \in I \setminus \{r\}$ do
    Let change += $i(a_{i+1} - a_i)$;
    Let sum += change;
% Now sum contains $d_1$
Let change = $k(r-1)$;
foreach $i \in I$ do
    Let change −= $2(a_i - a_1)$;
% Now change contains $d_2 - d_1$
foreach $t \in I$ do
    % Now sum contains $d_t$ and change contains $d_{t+1} - d_t$
    if sum < opt then
        % The solution $(c_1^t, \ldots, c_r^t)$ is the best so far
        Let opt = sum;
        Let arg = $t$;
    Let sum += change;
    Let change += $2r(a_{t+1} - a_t) - 2k$;
% Compute the solution $(c_1^t, \ldots, c_r^t)$ for $t$ = arg
foreach $i \in I$ do
    if $i \ge$ arg then
        Let $c_i = 0$;
    else
        Let $c_i = 1$;
return $(c_1, \ldots, c_r)$ and opt;
```

B Proof of Lemma 5

Let $t_1 \le t_2$. We claim that an optimal partition is found by traversing the set I in descending order, adding the first $\lfloor t_2 - t_1 \rfloor$ elements to I_1 and the following elements to I_1 and I_2 in turn. We will show that we can get this partition by a finite number of transformations starting from any partition, such that none of the transformation steps increases the objective function value. The first steps yield the required numbers of elements in I_1 and I_2, the next steps yield the required partition by traversing the elements of I in descending order and exchanging every element that does not belong to the required part with a smaller one. Applying these transformations to an optimal solution, we get the desired result.

So let $I = I_1 \oplus I_2$ be any partition. The required number of elements in I_2 is

$$\lfloor (\#I - \lfloor t_2 - t_1 \rfloor)/2 \rfloor ,$$

which is equivalent to

$$\#I_1 - \lfloor t_2 - t_1 \rfloor - 2 < \#I_2 \le \#I_1 - \lfloor t_2 - t_1 \rfloor .$$

So first assume that $\#I_2 \le \#I_1 - \lfloor t_2 - t_1 \rfloor - 2 < \#I_1 - (t_2 - t_1) - 1$. Let j be the smallest element in I_1. We claim that moving j to I_2 does not increase the objective function value. Indeed, the change of its value is

$$\begin{aligned} &\sum_{\substack{i \in I_2 \\ i<j}} i + \#\{i \in I_2 \mid i > j\} j + t_2 j - (\#I_1 - 1) j - t_1 j \\ \le\ & \#I_2 j + t_2 j - (\#I_1 - 1) j - t_1 j \\ =\ & j\,(\#I_2 - \#I_1 + 1 + t_2 - t_1) \le 0 . \end{aligned}$$

Analogously, if $\#I_2 > \#I_1 - \lfloor t_2 - t_1 \rfloor$, then $\#I_2 \ge \#I_1 + 1 - (t_2 - t_1)$. Moving the smallest element j of I_2 to I_1 increases the objective function value by

$$\begin{aligned} &\sum_{\substack{i \in I_1 \\ i<j}} i + \#\{i \in I_1 \mid i > j\} j + t_1 j - (\#I_2 - 1) j - t_2 j \\ \le\ & \#I_1 j + t_1 j - (\#I_2 - 1) j - t_2 j \\ =\ & j\,(\#I_1 - \#I_2 + 1 - (t_2 - t_1)) \le 0 . \end{aligned}$$

Now let j be the largest element of I not belonging to the required part. First assume that j belongs to I_2 but is required to belong to I_1. Since the numbers of elements of I_1 and I_2 are as required and all elements larger than j belong to the required part, there is an element $j' < j$ such that

$$\#\{i \in I_1 \mid j' \le i \le j\} = \#\{i \in I_2 \mid j' \le i \le j\} .$$

We choose j' maximally; in particular, $j' \in I_1$. Then

$$\sum_{\substack{i \in I_1 \\ j' \le i \le j}} i \le \sum_{\substack{i \in I_2 \\ j' \le i \le j}} i .$$

Let r be the number of elements of I larger than j, but at most $\lfloor t_2 - t_1 \rfloor$. Finally, exchange j and j'; i.e., move j from I_2 to I_1 and j' from I_1 to I_2. By this transformation, the objective function value increases by

$$\begin{aligned} & \sum_{\substack{i \in I_1 \\ j' \le i \le j}} i - j + rj - \sum_{\substack{i \in I_2 \\ j' \le i \le j}} i + j' - rj' + t_1(j - j') + t_2(j' - j) \\ \le \; & (j - j')(r - 1 + t_1 - t_2) \le 0 \end{aligned}$$

by definition of r, so the objective function value does not increase.

Finally, assume that j belongs to I_1 but is required to belong to I_2. In particular, j does not belong to the largest $\lfloor t_2 - t_1 \rfloor$ elements of I. Again, choose j' maximally such that

$$\#\{i \in I_1 \mid j' \le i \le j\} = \#\{i \in I_2 \mid j' \le i \le j\} \; .$$

Now $j' \in I_2$ and

$$\sum_{\substack{i \in I_1 \\ j' \le i \le j}} i - \sum_{\substack{i \in I_2 \\ j' \le i \le j}} i \ge j - j' \; .$$

Let $r = \lfloor t_2 - t_1 \rfloor$. Exchanging j and j' increases the objective function value by

$$\begin{aligned} & - \sum_{\substack{i \in I_1 \\ j' \le i \le j}} i - rj + \sum_{\substack{i \in I_2 \\ j' \le i \le j}} i + rj' + t_1(j' - j) + t_2(j - j') \\ \le \; & (j - j')(-1 - r - t_1 + t_2) \le 0 \; . \end{aligned}$$

It remains to explain how to compute the objective value corresponding to a given partition $I = I_1 \oplus I_2$ in $O(r \log r)$ time. This is done by Algorithm 2, applied to I_1 and I_2. □

Algorithm 2: Computing the number of intra-orbit edge crossings

```
Input: A set $I$ of integers and $t \ge 0$.
Output: $\sum_{\substack{i,j \in I \\ i<j}} i + t \sum_{i \in I} i$.
Let change = 0;
Let sum = 0;
Sort $I$ ascendingly;
foreach $i \in I$ do
    Let sum += change + $t$;
    Let change += $i$;
return sum;
```