Inserting Multiple Edges into a Planar Graph

Markus Chimani[*] [1] **and Petr Hliněný**[†] [2]

1 **Theoretical Computer Science, University Osnabrück, Osnabrück, Germany**
markus.chimani@uni-osnabrueck.de
2 **Faculty of Informatics, Masaryk University Brno, Brno, Czech Republic**
hlineny@fi.muni.cz

Abstract

Let G be a connected planar (but not yet embedded) graph and F a set of additional edges not in G. The *multiple edge insertion* problem (MEI) asks for a drawing of $G+F$ with the minimum number of pairwise edge crossings, such that the subdrawing of G is plane. An optimal solution to this problem is known to approximate the crossing number of the graph $G+F$.

Finding an exact solution to MEI is NP-hard for general F, but linear time solvable for the special case of $|F|=1$ (SODA 01, Algorithmica) and polynomial time solvable when all of F are incident to a new vertex (SODA 09). The complexity for general F but with constant $k=|F|$ was open, but algorithms both with relative and absolute approximation guarantees have been presented (SODA 11, ICALP 11). We show that the problem is fixed parameter tractable (FPT) in k for biconnected G, or if the cut vertices of G have bounded degrees. We give the first exact algorithm for this problem; it requires only $\mathcal{O}(|V(G)|)$ time for any constant k.

1998 ACM Subject Classification F.2.2 [Nonnumerical Algorithms and Problems] Geometrical problems and computations

Keywords and phrases crossing number; edge insertion; parameterized complexity; path homotopy; funnel algorithm

Digital Object Identifier 10.4230/LIPIcs.SoCG.2016.30

1 Introduction

The crossing number $\mathrm{cr}(G)$ of a graph G is the minimum number of pairwise edge crossings in a drawing of G in the plane. Finding the crossing number of a graph is one of the most prominent difficult optimization problems in graph theory [18] and is NP-hard already in very restricted cases, e.g., even when considering a planar graph with one added edge [6] (such graphs are called *almost-planar* or near-planar). The problem has been vividly investigated for over 60 years, but there is still surprisingly little known about it; see e.g. [32] for an extensive reference. There exists a $c>1$ such that $\mathrm{cr}(G)$ cannot be approximated within a factor c in polynomial time [4], but we do not know whether $\mathrm{cr}(G)$ is approximable within some constant ratio for general G.

Several approximation algorithms arose for special graph classes. For general graphs with bounded degree, there is an algorithm that approximates the quantity $n+\mathrm{cr}(G)$ instead, giving an approximation ratio of $\mathcal{O}(\log^2 n)$ [2, 17]. A sublinear approximation factor of $\tilde{\mathcal{O}}(n^{0.9})$ for $\mathrm{cr}(G)$ in the bounded-degree setting was given in an involved algorithm [13]. We know constant factor approximations for bounded-degree graphs that are embeddable in

[*] M. Chimani has been supported by the German Research Foundation (DFG) project CH 897/2-1.
[†] P. Hliněný has been supported by the research center Institute for Theoretical Computer Science (CE-ITI); Czech Science foundation project No. P202/12/G061.

some surface of higher genus [19, 26, 24], or that have a small set of graph elements whose deletion leaves a planar graph – removing and re-inserting these elements can give strong approximation bounds such as [25, 5, 12, 14].

In this paper, we follow the latter idea and concentrate on the *Multiple Edge Insertion* problem MEI(G, F). Intuitively, we are given a planar graph G, and ask for the best way (in terms of total crossing number) to draw G and insert a set of new edges F into G such that the final drawing of $G + F$ (i.e., of the graph including the new edges of F) restricted to G remains planar. The formal definitions follow in Section 2.

This MEI problem is polynomial-time solvable for $|F| = 1$ [22] (unlike the crossing number problem of almost-planar graphs) or if all edges of F are incident to a common vertex [10], but NP-hard for general F [35]. An exact or at least approximate MEI solution constitutes an approximation for the crossing number of the graph $G+F$ [12]; see Section 2. Considering general constant $k := |F|$, there have been two different approximation approaches [14] and [11]; the former one directly targets the crossing number and achieves only a relative approximation guarantee for MEI; the latter one first specifically attains an approximation of MEI with only an additive error term, and then uses [12] to deduce a crossing number approximation. While the former is not directly practical, the latter algorithm [11] in fact turns out to be one of the best choices to obtain strong upper bounds in practice [9].

In this paper, we show that for every constant k and under mild connectivity assumptions, there is an exact linear time algorithm; till now, this was an open problem even for $k = 2$.

▶ **Theorem 1.** *Let G be a planar graph and F a set of $k \geq 1$ new edges (vertex pairs, in fact). Assume G is biconnected, or G is connected and all cut vertices of G have degree bounded from above by $2^{p(k)}$ where p is a polynomial. Then there is a polynomial function q such that the problem* MEI(G, F) *is solvable to optimality in time $\mathcal{O}(2^{q(k)} \cdot |V(G)|)$. In terms of parameterized complexity, our algorithm is linear-time FPT with the parameter $k = |F|$.*

Note that the bounded-degree requirement for cut vertices compares favorably to existing approximation algorithms for the crossing number, all of which require a degree bound for *all* vertices. We also remark that while the crossing number problem is in FPT w.r.t. the objective value (cr(G)) [20, 28], already a planar graph with one added edge may have unbounded crossing number and so these two modes of parameterization are incomparable.

Our high-level approach is a standard idea in this area, using dynamic programming on a decomposition (known as SP(Q)R-tree) of the planar graph; see Section 4. Similar ideas have been used, e.g., in aforementioned [22, 10, 14, 11]. However, this time it turns out that the most interesting and difficult case is the basic one of rigid components. The corresponding problem *Rigid MEI*, i.e., MEI under the restriction that a planar embedding of G is fixed beforehand, is still NP-hard [35]. An FPT algorithm for Rigid MEI is given in Section 3.

On an informal level, the algorithm for Rigid MEI simultaneously searches for shortest dual paths corresponding to the edges of F in rigid G, while keeping track of their mutual crossings. Although this task may seem similar, in the dual, to the notoriously hard problem of shortest disjoint paths in planar graphs [15, 30], there is the crucial difference that our paths may share sections as long as they do not cross. Our algorithm utilizes the concept of *path homotopy* in the plane with obstacles, and uses a special structure which we call a *trinet*, to represent and search for shortest dual paths of a given homotopy.

Organization. Due to restricted space, the proofs of some statements are left for the full version of this paper (see also `http://arxiv.org/abs/1509.07952`); these statements are marked with an asterisk *.

2 Definitions

We use the standard terminology of graph theory. By default, we use the term *graph* to refer to a multigraph, i.e., we allow parallel edges (also self-loops are allowed but those can be safely ignored in the context of crossing numbers in the plane). If there is no danger of confusion, we denote an edge with the ends u and v chiefly by uv.

Drawing a graph. A *drawing* of a graph $G = (V, E)$ is a mapping of the vertices V to distinct points on a surface Σ, and of the edges E to simple (polygonal) curves on Σ, connecting their respective end points but not containing any other vertex point. Unless explicitly specified, we will always assume Σ to be the plane (or, equivalently, the sphere). A *crossing* is a common point of two distinct edge curves, other than their common end point. A drawing is *plane* if there are no crossings. *Plane embeddings* form equivalence classes over plane drawings, in that they only define the cyclic order of the edges around their incident vertices (and, if desired, the choice of the outer, infinite face). A *planar* graph is one that admits a plane embedding. A *plane* graph is an embedded graph, i.e., a planar graph together with a plane embedding.

Given a plane embedding G_0 of G, we define its geometric *dual* G_0^* as the embedded multigraph that has a (dual) vertex for each face in G_0; (dual) vertices are joined by a (dual) edge for each (primal) edge shared by their respective (primal) faces. The cyclic order of the (dual) edges around any common incident (dual) vertex v^*, is induced by the cyclic order of the (primal) edges around the (primal) face corresponding to v^*.

We refer to a path/walk in G_0^* as a *dual path/walk* in G_0, and we speak about a *dual path/walk π in G_0 between vertices u, v* if the π starts in a face incident with u and ends in a face incident with v. We shortly say a *route from u to v* (a *u–v route*) to mean a dual walk between vertices u, v (recall that a walk, unlike a path, may repeat vertices and edges).

Crossing numbers and edge insertion. Given a drawing D of G, let $\mathrm{cr}(D)$ denote the number of pairwise edge crossings in D. The *crossing number* problem asks for a drawing D° of a given graph G with the least possible number $\mathrm{cr}(D^\circ) =: \mathrm{cr}(G)$. By saying "pairwise edge crossings" we emphasize that we count a crossing point x separately for every pair of edges meeting in x (e.g., ℓ edges meeting in x give $\binom{\ell}{2}$ crossings).

It is well established that the search for optimal solutions to the crossing number problems can be restricted to so-called *good* drawings: any pair of edges crosses at most once, adjacent edges do not cross, and there is no point that is a crossing of three or more edges.

In this paper we especially consider the following variant of the crossing number problem:

▶ **Definition 2** (Multiple edge insertion, Rigid MEI and MEI).
Consider a planar, connected graph G and a set of edges (vertex pairs, in fact) F not in $E(G)$. We denote by $G + F$ the graph obtained by adding F to the edge set of G.

Let G_0 be a plane embedding of G. The *Rigid Multiple Edge Insertion* problem, denoted by r-MEI(G_0, F), is to find a drawing D of the graph $G + F$ with minimal $\mathrm{cr}(D)$ such that the restriction of D to G is the plane embedding G_0. The attained number of crossings is denoted by r-ins(G_0, F).

The *Multiple Edge Insertion* problem MEI(G, F) is to find an embedding G_1 of G (together with the subsequent drawing D as above), for which r-MEI(G_1, F) attains the minimum number of crossings. The latter is denoted by $\mathrm{ins}(G, F)$.

As mentioned above, a solution of a MEI(G, F) instance – which is a trivial upper bound on $\mathrm{cr}(G+F)$, readily gives an approximate solution of the crossing number problem of $G+F$ by the following inequality:

▶ **Theorem 3** (see [12]). *Consider a planar graph G and a set of edges F not in $E(G)$. Then* $\mathrm{ins}(G, F) \leq |F| \cdot \Delta(G) \cdot \mathrm{cr}(G+F) + \binom{|F|}{2}$, *where $\Delta(G)$ is the maximum degree in G.*

3 Rigid MEI

We first give an FPT algorithm for solving the rigid version r-MEI(G, F), parameterized by $k = |F|$. G is hence a plane graph (i.e., with a fixed embedding) throughout this section.

We first illustrate the simple cases. Solving r-MEI$(G, \{uv\})$, the fixed embedding edge insertion problem with $k = 1$, is trivial. Augment dual G^* with edges of length 0 between the terminals u, v and their respective incident faces (vertices in G^*), to suit the definition of a u–v route in G. Then, simply compute the shortest u–v route in this graph. *Realizing* a route for uv means to draw uv along it within G. If the shortest route has length ℓ, realizing it attains r-ins$(G, \{vw\}) = \ell$, the smallest number of crossings in the Rigid MEI setting.

For $k \geq 2$, the situation starts to be more interesting: not every collection of shortest routes gives rise to an optimal solution of r-MEI(G, F) since there might arise crossings between edges of F. While for $k = 2$, the only question is whether some pair of shortest routes of the two edges in F can avoid crossing each other, for larger values of k we can encounter situations in which all the optimal solutions of r-MEI(G, F) draw some edges of F quite far from their individual shortest routes (in order to avoid crossings with other edges of F), and a more clever approach is needed.

On a very high level, our approach to finding a drawing D of $G + F$ that is an optimal solution to r-MEI(G, F), can be described as follows:

1. We guess, for each pair $f, f' \in F$, whether f and f' will cross each other in D. Since $k = |F|$ is a parameter, all the possibilities can be enumerated in FPT time.
2. Let $X \subseteq \binom{F}{2}$ be a (guessed) set of pairs of edges of F. We find a collection of shortest routes for the edges of F in G under the restriction that exactly the pairs in X cross; Drawing D_X is obtained by inserting the edges of F along their computed routes. As we will see, we may restrict our attention to routes pairwise crossing at most once.
3. We select $D := D_X$ which minimizes the sum of $|X|$ and of the lengths of the routes.

3.1 Handling path homotopy of routes

Obviously, the core task of the scheme (1)–(3) is to solve the point (2) of finding a collection of shortest routes under the restriction that every route avoids crossing certain other routes (note; none of these routes are fixed in advance). The key to this is the concept of *path homotopy* in the plane with point obstacles. Due to the nature of our arguments, in this paper we choose to deal with path homotopy in a combinatorial setting of T-sequences as in Definition 5. This new setting is closely inspired by the discrete-geometry view of boundary-triangulated 2-manifolds by Hershberger and Snoeyink [23]:

First, we "triangulate" the point set $V(F)$ (our obstacles) using transversing paths in the embedding G. A *transversing path* between vertices x, y of G is a path whose ends are x, y and whose internal vertices subdivide some edges of G. Let T be the union of these transversing paths and G' denote the corresponding subdivision of G. In order to avoid a terminology clash with graph triangulations, we will call T in the pair (G', T) a *trinet of* G. This is illustrated in Figure 1 and formally given here (where $V(F) = N$):

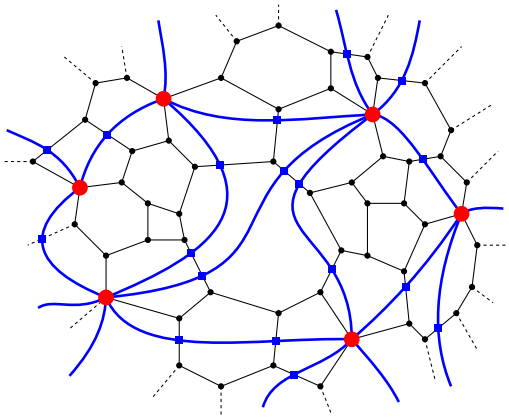

Figure 1 An example of a trinet of a plane graph G (see Definition 4): underlying G is in thin black, the trinodes in red, and the triedges in blue with square blue nodes subdividing edges of G.

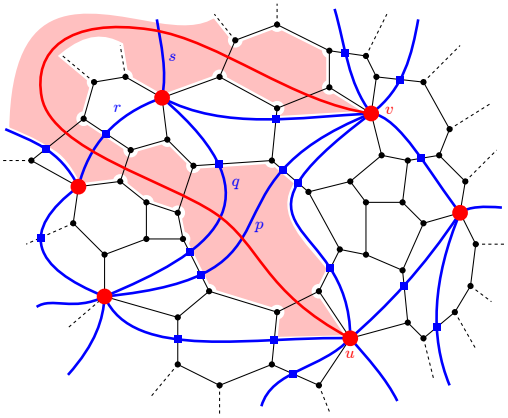

Figure 2 An example (see Definition 5): the T-sequence of the u–v route depicted in red is (p, q, r, s) from u to v. It is a proper T-sequence from u to v (Definition 7); its corresponding alley is shaded in light red.

▶ **Definition 4** (Trinet)**.** Let G be a connected plane graph and $N \subseteq V(G)$, $|N| \geq 4$. A plane graph T such that $V(T) \cap V(G) = N$ is called a *trinet* of G if the following holds:

(a) T is a subdivision of a 3-connected plane triangulation on the vertex set N (in particular, every face of T is incident with precisely three vertices of N), and

(b) there exists a subdivision G' of G such that $V(G') \setminus V(G) = V(T) \setminus N$, $E(G') \cap E(T) = \emptyset$ and the union $G' \cup T$ is a plane embedding.

Pair (G', T) is a *full trinet* of G. The vertices in $N(T) := N$ are called *trinodes* of T, the maximal paths in T internally disjoint from N are *triedges* and their set is denoted by $I(T)$, and the faces of T are *tricells*. Note that the triedges of T are transversing paths of G.

We need to introduce terms related to (and describing) a path homotopy in a full trinet (G', T) of a plane graph G. See also Figure 2 for an illustration of this definition.

▶ **Definition 5** (Alley and T-sequence)**.** Let (G', T) be a full trinet of a plane graph G. Consider a route π between $u, v \in V(G)$ in the graph $G' \cup T$. Then $V(\pi) = \{\phi_0, \phi_1, \ldots, \phi_m\}$ where each dual vertex ϕ_i of π is an open face of $G' \cup T$. Let these faces $(\phi_0, \phi_1, \ldots, \phi_m)$ be ordered along π such that ϕ_0 is incident to u and ϕ_m incident to v. Let $(e_1, e_2, \ldots, e_m) \subseteq E(G' \cup T)$ be the sequence of the primal edges of the dual edges of π, ordered from ϕ_0 to ϕ_m. As a point set, each edge e_i is considered without the endpoints.

(a) The union $\{u, v\} \cup \bigcup_{i=0}^{m} \phi_i \cup \bigcup_{i=1}^{m} e_i$ is called the *alley of* π (or, an *alley between* u, v).

(b) Let $(e'_1, \ldots, e'_\ell) \subseteq (e_1, e_2, \ldots, e_m)$ be the restriction to $E(T)$, and let $(p_1, p_2, \ldots, p_\ell) \subseteq I(T)$ be the sequence of triedges such that p_i contains the edge e'_i for $i = 1, \ldots, \ell$. Then $(p_1, p_2, \ldots, p_\ell)$ is called the *T-sequence of* π from u to v (or, of the corresponding alley from u to v).

The purpose of introducing an alley is to describe a topological corridor for all u–v arcs of a similar kind (and same number of crossings) in the embedding $G' \cup T$. The correspondence is clear: a route π *crosses* a triedge p if the alley of π contains one of the G'-edges forming p. The T-sequence of π hence describes the unique order (with repetition) in which π crosses the triedges of T. Usually, we shall consider only the case of $u, v \in N(T)$.

A route may, in general, cross the same triedge many times (even in one place), but we aim to prove that for "reasonable" routes there is an explicit upper bound; see Lemma 9.

3.2 Refined approach to Rigid MEI

We slightly generalize the shortest route setting to allow for a connected plane graph G with *integer edge weights* $w\colon E(G) \to \mathbb{N}_+ \cup \{\infty\}$. For a full trinet (G', T) of G, we define the edge weights of $G' \cup T$ as follows: $w'(p) := 0$ for all $p \in E(T)$ and $w'(e') := w(e)$ where $e' \in E(G')$ is obtained by subdividing $e \in E(G)$. This w' is the *weight induced by* w in the trinet (G', T). We give the same weights w' also to the edges of the geometric dual of $G' \cup T$. If α is the alley of a route π between vertices x, y in $G' \cup T$, then the *length of* α equals the length of π, i.e., the sum of the w'-weights of the dual edges of π.

For any weighted graph G' with $w\colon E(G') \to \mathbb{N}_+ \cup \{\infty\}$, as above, we correspondingly generalize the notion of a crossing number to *weighted crossing number* as follows: a crossing between two edges $e_1, e_2 \in E(G')$ accounts for the amount of $w(e_1) \cdot w(e_2)$ in $\mathrm{cr}(G')$. For MEI(G, F) problem variants with weighted G, we shall always assume that the weight of each edge $f \in F$ is $w(f) = 1$, and so the (sum of) weighted crossings of f in a drawing of $G + f$ are naturally determined by the weights of the edges crossed by f.

With the help of the framework developed in the previous section, we can now give an (again informal) high-level refinement of our solution steps of r-MEI(G, F) as follows:

4. Consider a trinet T of G on the trinodes $V(F)$. If we fix a (realizable) T-sequence S, then we can use established tools, namely an adaptation of the idea of the funnel algorithm [7, 31], to efficiently compute a shortest alley among those having the same T-sequence S. For $uv \in F$, if we compute an alley α between u, v of length ℓ, then we can easily draw the new edge uv as an arc in α with ℓ weighted crossings in G.
5. Suppose that, for $i = 1, 2$, α_i is a shortest alley between x_i and y_i having the T-sequence S_i. Then, as detailed later in Lemma 14, we can decide from *only* S_1, S_2 whether there exist arcs from x_1 to y_1 in α_1 and from x_2 to y_2 in α_2, which do not cross (note that $\alpha_1 \cap \alpha_2$ may be nonempty and yet there may exist such a pair of non-crossing arcs). Moreover, if the two arcs cross then it should be only once.
6. Consequently, it will be enough to loop through all "suitable" T-sequences for every edge of F and independently perform the steps (4), (5) for each combination of them, in order to get an optimal solution of r-MEI(G, F) as in (3). The point is to bound the number of considered T-sequences in terms of only the parameter $k = |F|$.

3.3 T-sequences of potential shortest routes

Considering the outline (4)–(6), we first resolve the last point which is a purely mathematical question. In order to achieve the goal, we shall build a special trinet of G along shortest dual paths between the trinodes in G (Definition 6), and then we will be able to restrict our attention to special T-sequences (Definition 7) of bounded length. The latter is formulated in Lemma 9 whose proof presents the main piece of technical work in this paper.

▶ **Definition 6** (Shortest-spanning trinet)**.** Let (G', T) be a full trinet of a plane graph G, and let the weights w' in (G', T) be induced by weights w in G. For a triedge $q \in I(T)$, every internal vertex t of q is incident with two edges e, e' of G' of weight $w'(e) = w'(e')$ which we call the weight of t. The *transversing weight of* q equals the sum of the weights of the internal vertices of q.

A triedge $q \in I(T)$ between trinodes x, y is *locally-shortest* if the transversing weight of q is equal to the length of a shortest dual path π in $G' \cup T$ between x, y, such that π is contained in(!) the union of the two tricells incident with q (including the points of q itself). Similarly, q is *globally-shortest* if the transversing weight of q is equal to the dual distance between x, y in $G' \cup T$.

We say that T has the *shortest-spanning property* if every triedge in $I(T)$ is locally-shortest, and there exists a subset of triedges $J \subseteq I(T)$ forming a connected subgraph of T spanning all the trinodes such that every triedge in J is globally-shortest.

▶ **Definition 7** (Proper T-sequence)**.** Consider a trinet T and trinodes $u \neq v \in N(T)$. A *nonempty* sequence $S = (p_1, p_2, \ldots, p_m) \subseteq I(T)$ of triedges of T (repetition allowed) is a *proper T-sequence from u to v* if the following holds: u is disjoint from p_1 but there exists a tricell θ_0 incident with both u and p_1, v is disjoint from p_m but there exists a tricell θ_m incident with both v and p_m, and each two consecutive triedges p_i, p_{i+1} are distinct and incident to a common tricell θ_i for $1 \leq i < m$. Finally, the *empty* sequence $S = \emptyset$ is considered a proper T-sequence from u to v if u, v are incident to a common tricell θ_0.

Since T is a subdivision of a graph triangulation and u, v are nodes of this triangulation, we immediately obtain the following fact complementing Definition 7:

▶ **Claim 8.** *For every proper nonempty T-sequence S, the sequence of tricells $(\theta_0, \theta_1, \ldots, \theta_m)$ as in Definition 7 is uniquely determined by T and S. If $S = \emptyset$, then uv is a triedge of T and there are two choices of θ_0 incident with uv; we simply make an arbitrary deterministic choice of θ_0 among those two in each case.*

Now the main technical finding, necessary for our algorithm, comes in:

▶ **Lemma 9 (*).** *Consider an instance* r-MEI(G, F) *where G is a connected plane graph. Let (G', T) be a full trinet of G having the shortest-spanning property. There exists a set $\{\pi_f : f \in F\}$ where π_f for $f = uv$ is a route in $G' \cup T$ between the trinodes u, v, such that the following hold:*

- **(a)** *There exists an optimal drawing D of $G + F$ with* r-ins(G, F) *crossings such that each edge $f \in F$ is drawn in the alley of π_f, and no two edges of F cross each other more than once.*
- **(b)** *The T-sequence S_f of each π_f is a proper T-sequence, and no triedge occurs in S_f more than $8k^4$ times where $k = |F|$.*

The full proof of Lemma 9 is rather long and cannot fit into the restricted short paper, but due to great importance of the statement we at least provide here a brief informal sketch:

- We show by local modifications of the routes for F that there exists an optimal solution which fulfills (a) and determines only proper T-sequences in the considered trinet.
- If some triedge repeats too many times in the T-sequence of some edge $f \in F$, then also one of the globally-shortest triedges of T, say p, repeats many times in this sequence. Hence, by the globally-shortest property, re-routing f partly "along" p does not increase crossings of f with $E(G)$. Though, there may be no strict improvement either. Even worse, re-routing of f may incur new crossings with F which is difficult to handle locally.
- A careful analysis of the situations in which the T-sequence of f crosses many times the same globally-shortest triedge of T then gives the desired conclusion; that either the local situation contradicts optimality of the whole solution, or a strict improvement (in terms of the lengths of T-sequences) can be achieved by a local move in the drawing.

3.4 Shortest routes in a sleeve

Next, we consider point (4) of the above outline. To recapitulate, for trinodes u, v of a trinet T of G and a given proper T-sequence S from u to v, the task is to find a shortest route from u to v among those having the same T-sequence S (and independently of other

searched routes). Since we cannot, in general, avoid repeating triedges in S and tricells in the sequence $(\theta_0, \theta_1, \ldots, \theta_m)$ in Definition 7, we use a similar workaround as in [23]; "lifting" the respective sequence of tricells into a universal cover as follows.

▶ **Definition 10** (Sleeve of a T-sequence)**.** Let (G', T) be a full trinet of a plane graph G, and consider a proper T-sequence $S = (p_1, p_2, \ldots, p_m)$ from u to v determining the sequence of tricells $(\theta_0, \theta_1, \ldots, \theta_m)$ by Claim 8. For $i = 0, 1, \ldots, m$, let L_i be a disjoint copy of the embedded subgraph of $G' \cup T$ induced by θ_i. Construct a plane graph L from the union $L_0 \cup \cdots \cup L_m$ by identifying, for $j = 1, \ldots, m$, the copy of the triedge p_j in L_{j-1} with the copy of p_j in L_j. We call L the *sleeve of* S in the trinet (G', T), and we identify u and v with their copies in L_0 and L_m, respectively. We make the unique face of L that is not covered by a copy of any tricell of T the *outer face* of L.

Observe that every route from u to v in $G' \cup T$ having its T-sequence equal to S can be easily lifted into a corresponding u–v route in the sleeve L of S. Conversely, any u–v route in L avoiding the outer face and crossing the copies of triedges in L at most once each, can be obviously projected down to $G' \cup T$ to make a route with the T-sequence equal to S. In fact, we can routinely prove that some shortest u–v route in L must be of the latter kind, under the shortest-spanning property (cf. Definition 6).

▶ **Lemma 11 (*).** *Let (G', T) be a shortest-spanning full trinet of an edge-weighted plane graph G, S a proper T-sequence between trinodes u, v of T, and let L be the sleeve of S. Let ℓ be the length of a shortest route from u to v among those having the T-sequence S. Then, ℓ is equal to the dual distance from u to v in L without the outer face, and at least one of the u–v routes of length ℓ in L crosses the copy of each triedge from S in L exactly once.*

Hence we can straightforwardly compute desired shortest routes using established linear time shortest path algorithms, such as the algorithm of Klein et al. [29] since L is planar, or Thorup's algorithms [33] since we have integral weights. In fact, since we are fine with edge weights of G given in unary, a simple adaptation of BFS can do the job for us, too.

▶ **Corollary 12 (*).** *Let (G', T) be a shortest-spanning full trinet of a plane graph G with integer edge weights, and S a proper T-sequence between trinodes u, v of T. A shortest u–v route among those having the T-sequence S can be found in $\mathcal{O}(|S| \cdot |N(T)| \cdot |V(G)|)$ time.*

Observe that in our case, by Lemma 9, we have $|S| \cdot |N(T)| \leq (6k \cdot 8k^4) \cdot 2k = 96k^6$.

3.5 Crossing of routes

Finally, it remains to address point (5). Consider a 4-tuple of distinct trinodes u, v, u', v'. Let π be a u–v route and π' be a u'–v' route. We say that an *arc b follows the route π* if b is contained in the alley of π and b intersects the faces forming the alley exactly in the order given by π (recall that a route is technically a dual walk and hence, possibly, some face might repeat in π). We say that the pair of routes π, π' is *non-crossing*, if there exist a u–v arc b following π and a u'–v' arc b' following π' such that $b \cap b' = \emptyset$. In order to characterize possible non-crossing pairs of routes in terms of their T-sequences, we bring the following:

▶ **Definition 13** (Crossing certificate)**.** Let (G', T) be a full trinet of a plane graph G, and let π be a route from u to v and π' be a route from u' to v' in $G' \cup T$, where u, v, u', v' are distinct trinodes of T. Assume the T-sequences $S = (p_1, \ldots, p_n)$ of π and $S' = (p'_1, \ldots, p'_\ell)$ of π' are proper and let $(\theta_0, \ldots, \theta_n)$ and $(\theta'_0, \ldots, \theta'_\ell)$ be their tricell sequences by Claim 8. For technical reasons, let $p_0 := u$, $p_{n+1} := v$ and $p'_0 := u'$, $p'_{\ell+1} := v'$.

A *crossing certificate for* S, S' is a triple of indices (c, d, m) where $c, d, m \geq 0$, $c + m \leq n$, $d + m \leq \ell$, such that the following holds:

(a) $\theta_{c+j} = \theta'_{d+j}$ for $0 \leq j \leq m$, but $p_c \neq p'_d$ and $p_{c+m+1} \neq p'_{d+m+1}$,

(b) the triple p_c, p'_d, p_{c+1} occurs around the tricell $\theta_c = \theta'_d$ in the same cyclic orientation as the triple $p_{c+m+1}, p'_{d+m+1}, p_{c+m}$ occurs around $\theta_{c+m} = \theta'_{d+m}$.

Furthermore, a crossing certificate for the same sequence S and the reversal of S' from v' to u' is also called a crossing certificate for S, S'.

Definition 13 deserves a closer explanation. Assume that a crossing certificate satisfies $0 < c < n$ and $0 < d < \ell$. Then all four elements $p_c, p'_d, p_{c+1}, p'_{d+1}$ are triedges of the same tricell $\theta_c = \theta'_d$, and since $p_{c+1} \neq p_c \neq p'_d \neq p'_{d+1}$, we get $p_{c+1} = p'_{d+1}$. Hence $m > 0$ and the situation is such that S and S' "merge" at θ_c where (up to symmetry) S comes on the left of S', and they again "split" at θ_{c+m} where S leaves on the right of S', thereby "crossing it". The full definition, though, covers also the boundary cases of crossing certificates for which $c \in \{0, n\}$ or $d \in \{0, \ell\}$ (or both), and when S and S' may have no triedge in common; those can be easily examined case by case.

▶ **Lemma 14 (*).** *Let (G', T) be a full trinet of an edge-weighted plane graph G, and u_i, v_i, $i = 1, 2$, be four distinct trinodes. Assume that S_i from u_i to v_i are proper T-sequences. In $G' \cup T$, for $i = 1, 2$, there exist routes π_i from u_i to v_i having the the T-sequence S_i, such that π_1, π_2* are non-crossing*, if and only if there exists no crossing certificate for S_1, S_2.*

3.6 Summary of the r-MEI algorithm

We can now summarize the overall algorithm to solve Rigid MEI, see Algorithm 1. Based thereon, together with Lemmas 9, 11, Corollary 12, and Lemma 14 we obtain:

▶ **Theorem 15 (*).** *Let G be a connected plane graph with edge weights $w\colon E(G) \to \mathbb{N}_+ \cup \{\infty\}$, and F a set of $k \geq 1$ new edges (vertex pairs, in fact) such that $w(f) = 1$ for all $f \in F$. Algorithm 1 finds an optimal solution to the w-weighted* r-MEI(G, F) *problem, if a finite solution exists, in time $\mathcal{O}\big(2^{p(k)} \cdot |V(G)|\big)$, where $p(k)$ is a polynomial function in k.*

In fact, one can see that $p(k) = \mathcal{O}(k^6 \log k)$ in Theorem 15.

4 General MEI

Now, we turn our attention to the general MEI(G, F) problem, in which the embedding of a planar graph G is not pre-specified. Recall that triconnected planar graphs have a unique embedding (up to mirroring), but already biconnected graphs can have an exponential number of embeddings in general. As it is commonly done in insertion problems since [22], we will use the *SPR-tree* datastructure (sometimes also known as SPQR-tree) to encode and work with all these possible embeddings. The structure was first defined in a slightly different form in [16], based on prior work of [3, 34]. It can be constructed in linear time [27, 21] and only requires linear space.

▶ **Definition 16** (SPR-tree, cf. [8])**.** Let G be a biconnected graph with at least three vertices. The *SPR-tree* $\mathcal{T}$ of G is the unique smallest tree satisfying the following properties:

(a) Each node ν in $\mathcal{T}$ holds a specific (small) graph $S_\nu = (V_\nu, E_\nu)$, with $V_\nu \subseteq V(G)$, called a *skeleton*. Each edge e of E_ν is either a *real* edge $e \in E(G)$, or a *virtual* edge $e = xy \notin E(G)$ (while still, $x, y \in V(G)$).

In: a plane graph G, edge weights $w\colon E(G) \to \mathbb{N}_+ \cup \{\infty\}$, new edge set F s.t. $w(f) = 1$ for $f \in F$.
Out: an optimal solution to (w-weighted) r-MEI(G, F).

1. Compute a full trinet (G', T), $N(T) := V(F)$, with the shortest-spanning property of T.
   - **a.** Pick any trinode $x \in N(T)$ and greedily compute globally-shortest triedges (Def. 6) from x to all other trinodes, using a simple shortest path computation.
   - **b.** The remaining triedges can be greedily computed as locally-shortest, one after another.
2. For each $f = uv \in F$:
   - **a.** Compute $\mathcal{S}_f$ as the set of all its possible and relevant proper T-sequences from u to v. The size of $\mathcal{S}_f$ is bounded due to Lemma 9(b) by $2^{s(|F|)}$ where $s(k) = \mathcal{O}(k^5 \log k)$.
   - **b.** For each $S \in \mathcal{S}_f$, compute a shortest u–v route π_S in $G' \cup T$ among those having the T-sequence S (where the length function is induced by w), using Corollary 12.
3. For each possible system of representatives $\mathcal{P} = \{S_f\}_{f \in F}$ with $S_f \in \mathcal{S}_f$:
   - **a.** Check, for each pair $f, f' \in F$, whether there exists a crossing certificate for $S_f, S_{f'}$ (e.g., using brute force by Def. 13).
     Let $X_\mathcal{P}$ be the set of pairs $\{f, f'\}$ for which such a certificate has been found.
   - **b.** If any pair $\{f, f'\} \in X_\mathcal{P}$ requires more than a single crossing (which can be found by checking again for two "independent" (✱) crossing certificates of $S_f, S_{f'}$), let $\mathrm{cr}_\mathcal{P} := \infty$.
   - **c.** Otherwise, let $\mathrm{cr}_\mathcal{P} := |X_\mathcal{P}| + \sum_{f \in F} len_w(\pi_{S_f})$, where π_{S_f} is the shortest route for f and S_f computed in step (2) and $len_w(\pi_{S_f})$ is the length.
4. Among all $\mathcal{P}$ considered in (3), pick one with smallest $\mathrm{cr}_\mathcal{P} < \infty$. Let this be $\mathcal{P}^\circ = \{S_f^\circ\}_{f \in F}$.
5. In the plane graph G, realize each edge $f \in F$ following its respective route $\pi_{S_f^\circ}$, such that the overall resulting weighted number of crossings is $\mathrm{cr}_{\mathcal{P}^\circ}$:
   - **a.** By the minimality setup in (4), no $\pi_{S_f^\circ}$ is self-intersecting.
   - **b.** A standard postprocessing argument – removing consecutive crossings between any pair f, f' (within a section of $S_f^\circ \cap S_{f'}^\circ$) by re-routing f' partially along f – prevents multiple crossings in the pairs from $X_\mathcal{P}$ and makes the pairs of F not in $X_\mathcal{P}$ crossing-free.

Algorithm 1 Algorithm to solve the (weighted) Rigid MEI problem.

(b) $\mathcal{T}$ has three different node types with the following skeleton structures: **(S)** S_ν is a simple cycle; **(P)** S_ν consists of two vertices and at least three multiple edges between them; (**R**) S_ν is a simple triconnected graph on at least four vertices.

(c) For every edge $\nu\mu$ in $\mathcal{T}$ we have $|V_\nu \cap V_\mu| = 2$. These two common vertices, say x, y, form a vertex 2-cut in G. The skeleton S_ν contains a specific virtual edge $e_\mu \in E(S_\nu)$ that represents the node μ and, symmetrically, some specific $e_\nu \in E(S_\mu)$ represents ν. Both e_ν, e_μ have the ends x, y; the two virtual edges may refer to one another as *twins*.

(d) For an edge $\nu\mu \in E(\mathcal{T})$, let $e_\mu \in E_\nu$, $e_\nu \in E_\mu$ be the pair of virtual edges as in (c) connecting the same x, y. The operation of *merging* at $\nu\mu$ creates a graph $(S_\nu \cup S_\mu) - \{e_\mu, e_\nu\}$ obtained by gluing the two skeletons S_ν, S_μ at x, y and removing e_μ, e_ν.
Consider the tree $\mathcal{T}$ rooted at any node. For $\nu \in V(\mathcal{T})$ we define the *pertinent graph* P_ν of ν by recursively merging the skeletons at every tree-edge of the subtree rooted at ν, and removing the parent virtual edge of ν (if not the root itself). Then the pertinent graph of the root of $\mathcal{T}$ is G.

We again start with an illustration of the "simple" case of $|F| = 1$. The central theorem of [22] states that an optimal solution of MEI($G, \{uv\}$) for biconnected G can be obtained by looking only at the shortest path in the SPR-tree $\mathcal{T}$ of G between a node whose skeleton contains u and a node whose skeleton contains v. For each skeleton S_ν along this path, one simply finds an optimal embedding and a shortest partial route in this embedding between the virtual edge representing u (or u itself) and the virtual edge representing v (or v itself).

In the case of S- and P-nodes this route requires no crossings. For an R-node, one is looking for a shortest route in the rigid setting. Thereby, each virtual edge e_μ in S_ν with ends x, y is assigned its *pertinent weight* $w(e_\mu)$: the size of a minimum x–y edge-cut in the pertinent graph P_μ. See [22] for more details.

We return to the general MEI(G, F) problem for biconnected planar graphs G. Considering an arbitrarily rooted SPR-tree $\mathcal{T}$ of G, we devise a dynamic programming procedure to solve MEI bottom-up over $\mathcal{T}$. The core is to describe which subproblems we are going to solve at each node ν of $\mathcal{T}$, assuming we know the solutions of the corresponding subproblems at the child nodes of ν. For better understanding, this task is illustrated and described in Figure 3. We say a virtual edge e_μ in S_ν representing a child node μ of ν, is *dirty* if its pertinent graph P_μ contains an end vertex of $f \in F$ other than one of the ends of e_μ.

On a high level, we describe our procedure as follows:

▶ **Algorithm 17.** A dynamic programming scheme to solve MEI bottom-up over an SPR-tree.

1. We compute the pertinent weights $w(e_\mu)$ for all non-dirty virtual edges that appear in skeletons used in the following step (2).
2. For each dirty child virtual edge e_μ in the skeleton S_ν, we assume to know the optimal number of crossings for every "reasonable" scheme of routing new edges of F to, from, or across e_μ (we stress that we speak about *all* edges of F, including those not having an end in P_μ). We exhaustively process, over all possible embeddings of S_ν in case of a P-node, all the subproblems stated by every admissible combination of routing schemes for the dirty child virtual edges in S_ν and for the parent virtual edge of S_ν. For each "reasonable" routing scheme of the parent virtual edge, we then select and store the best obtained solution in terms of minimal overall crossing number.
3. Each particular subproblem of the exhaustive processing in (2) can be formulated as a weighted r-MEI(H, F'') instance. The plane graph H is constructed from S_ν by inserting special gadgets at the dirty virtual edges (see the colored digons in Figure 3), and the new edges F'' are suitable segments of the edges of F (their ends are either an end of the original F-edge or a vertex on such a gadget). All non-virtual edges of S_ν and all of F'' have weight 1. Each non-dirty virtual edge e_μ gets weight $w(e_\mu)$, computed in (1), and all dirty virtual edges get weight ∞. See Figure 3 for closer details.
   This r-MEI(H, F'') instance can then be solved using Theorem 15.
4. If ν is the root node of $\mathcal{T}$, then there is no parent virtual edge and the exhaustive processing in (2) selects a single optimal solution. Realizing this solution, together with the corresponding subsolutions at the descendants, gives an optimal solution to the original MEI(G, F) instance. ◀

We first give a more precise definition regarding step (2). For a dirty virtual edge $e_\mu = xy$ in the skeleton S_ν with its pertinent graph P_μ, let M be the set of those ends of the edges of F that belong to $V(P_\mu) \setminus \{x, y\}$. A *routing scheme* of e_μ is an arbitrary pair of disjoint sequences Q_1, Q_2, where $(Q_1 \cup Q_2) \cap V(G) = \emptyset$, together with a perfect matching on the set $M \cup Q_1 \cup Q_2$ (the cardinality of which must be even). The matching edges are meant to represent segments of edges of F drawn across the pertinent graph P_μ, but on this level we do not need to distinguish which segment belongs to which of the F-edges.

If φ is the parent node of ν then a routing scheme of e_φ in S_ν, as dealt with in step (2), is formally not the same as the corresponding routing scheme of the twin virtual edge e_ν in S_φ; these two are of course actually "complementary". We neglect this technical difference in a high-level description of our dynamic programming scheme.

Second, we informally outline three claims which make Algorithm 17 run in FPT time:

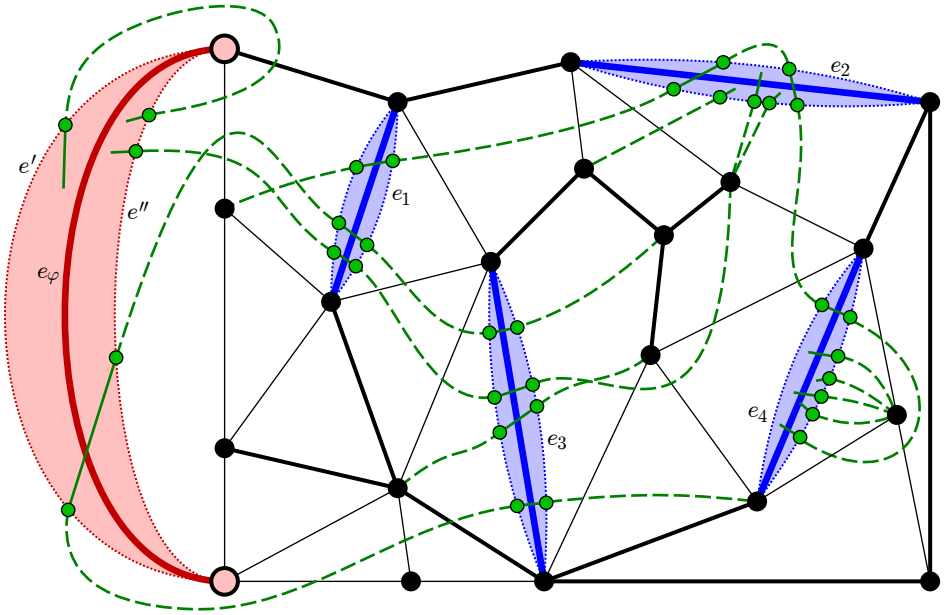

Figure 3 One of the possible r-MEI instances considered at a rigid R-node ν in Algorithm 17. The mostly black graph shows the skeleton S_ν, whereby the red virtual edge e_φ corresponds to ν's parent, the blue virtual edges $e_1, \ldots, e_4$ to ν's dirty children, and the thick black edges to the non-dirty children with their implicit pertinent weights. The new edges of F are depicted in green. For each dirty virtual edge e_i, including e_φ, we consider a fixed (by the processing in step (2)) *routing scheme* that externally encodes at which points segments of edges of F enter and leave the pertinent subgraph of e_i. This is depicted with a blue or red digon of e_i and solid green segments of the F-edges within it. Consequently, we do not have to care for internal details of the digonal parts (they are, in fact, not part of our r-MEI instance and only visualized for the reader's context), as they are either already resolved in the subproblems of the children or are going to be resolved in the ancestor nodes. Likewise, for an r-MEI instance at ν, the only relevant information for any non-dirty virtual edge is its pertinent weight and no subembedding details are necessary.
The *gadget* (see step (3)) used to enforce the aforementioned routing schemes at every dirty virtual edge e_i, is composed of the edge e_i itself (solid blue or red), and two new edges e_i', e_i'' parallel to e_i (dotted) which are subdivided by the prescribed entry/exit *terminals* (green dots) of the segments of F-edges. The gadget edges are weighted ∞ and their embedding is rigid. Possible end vertices of F-edges in $V(S_\nu)$ are also counted as the terminals. By a combination of the routing schemes we mean an arbitrary perfect matching (the green dashed segments) on these terminals. It is an *admissible combination* (cf. step (2)) if the union of the prescribed solid green and the matching dashed green segments forms an actual "realization" of the new edges of F.
An admissible combination then gives rise to the depicted r-MEI instance, as described in step (3), which can be solved using Theorem 15.

- A skeleton S_ν may contain arbitrarily more than k virtual edges, but only at most $2k$ of them may be dirty (containing an end vertex of F). The computation of the pertinent weights of non-dirty virtual edges can be done in linear overall time.
- A P-node skeleton S_ν may have an unbounded number of inequivalent embeddings (precisely $(q-1)!$ where q is the number of parallel edges in S_ν). However, if two *non-dirty* edges e_1, e_2 appear non-consecutively within the embedding of S_ν, there exists an alternative solution with the same number of crossings where e_2 is rerouted alongside e_1 (by optimality, both edges encounter the same number of crossings either way). Thus we

can restrict our attention to only those at most $(2k)!$ embeddings in which all non-dirty virtual edges are consecutive (in any internal order) within the embedding of S_ν.

- In contrast to the single edge insertion, an edge of F may easily be forced to cross the same virtual edge e_μ multiple times (a number depending only on k). However, the complexity of any "reasonable" routing of the edges of F across e_μ is bounded by a function of $k = |F|$, as shown in Lemma 9.

As a corollary we see that the number of subproblems generated in step (2), as well as the amount of information stored at any SPR-tree node, are bounded by a function of k.

The solution of each one subproblem in step (3) can be obtained using the algorithm for r-MEI in Theorem 15 in linear time for constant k, and there are at most $\mathcal{O}(|V(G)|)$ nodes in the tree $\mathcal{T}$. Instead of the naïve quadratic runtime bound, we even achieve a linear overall runtime bound by observing that the union of all skeletons is still only of $\mathcal{O}(|V(G)|)$ size. We obtain, as given in the introduction:

▶ **Theorem 18** (The biconnected case of Theorem 1 – *). *Let G be a planar biconnected graph and F a set of $k \geq 1$ new edges (vertex pairs, in fact). An optimal solution of the problem* MEI(G, F) *can be found in $\mathcal{O}(2^{q(k)} \cdot |V(G)|)$ time, where $q(k)$ is a polynomial function in k.*

For essentially all known insertion algorithms (in particular the single edge insertion [22], the vertex insertion [10], and the MEI approximation [11]), one typically first resolves the case of biconnected graphs (using SPR-trees as above). Then, it is relatively straightforward to lift the algorithms to connected graphs, by considering BC-trees (see below). Interestingly, this step seems much more complicated in the case of exact MEI.

Consider the well-known *block-cut tree* (BC-tree) decomposing any connected graph into its blocks (biconnected components). Using analogous techniques as in [11], we extend our dynamic programming approach by amalgamating the BC-tree of G with respective SPR-trees of the blocks, to obtain a linear-sized *con-tree* – with an additional node type **C**, for the cut vertices. The outline in Algorithm 17 is completed with the following:

(2⁺). If ν is a C-node representing a cut vertex c of G, then we exhaustively process all possibilities to combine the dirty blocks of G incident to c (while non-dirty blocks can be safely ignored).

Unfortunately, although we can again bound the number of dirty blocks by $2k$, there is now no easy "external description of routing" with respect to a cut vertex available (analogous to a routing scheme of a virtual edge). Consequently, the number of possibilities to consider in (2⁺) depends on k and the degree of the cut vertex c. We conclude:

▶ **Theorem 19** (The connected case of Theorem 1 – *). *Let G be a planar connected graph and F a set of $k \geq 1$ new edges (vertex pairs, in fact). Let Δ_c be the maximum degree over cut vertices of G. Then an optimal solution of the problem* MEI(G, F) *can be found in $\mathcal{O}\big(2^{q'(k)} \cdot \Delta_c^k \cdot |V(G)|\big)$ time, where $q'(k)$ is a polynomial function in k.*

5 Conclusion

In this paper, we have affirmatively answered the long standing open question, floating around ever since [1, 22], whether there is a polynomial-time algorithm to insert a constant-sized set of edges into a planar graph in a crossing minimal way. Previously, this has only been known for single edges; the problem with multiple edges could only be approximated. Our result also induces a new approximation algorithm for the general crossing number of graphs with

bounded number of edges beyond planarity. While the original problem was defined over unweighted graphs, we considered weighted graphs in our subproblems but only to limited extent.

▶ **Open Problem 20.** Is MEI fixed-parameter tractable if both G and F are weighted?

In fact, we can straight-forwardly answer this question affirmatively for weighted G but unweighted F, as should be clear from the above proofs. We may also assume weighted F, if the weights are bounded by k, by simply adding multiple copies of an edge to F. For generally weighted F, we observe that the dynamic programming part of solving MEI would be capable of achieving this feat. However, the required r-MEI subproblems are not, due to a technical rerouting argument in the proof of Lemma 9. It is not easy to circumvent this argument and the problem seems surprisingly related to that of decidability of string graphs – a connection that deserves future investigation.

As our final open question we include:

▶ **Open Problem 21.** Is there a polynomial-time algorithm to solve MEI for simply-connected graphs independent of Δ_c?

Such a dependency on Δ_c does not show up when inserting a single edge or a star into planar G [22, 10]. On the other hand, even more restrictive degree dependencies are very common and seemingly unavoidable in the known general crossing number approximations.

Acknowledgments. We thank Sergio Cabello and Carsten Gutwenger for helpful discussions.

References

1 C. Batini, M. Talamo, and R. Tamassia. Computer aided layout of entity relationship diagrams. *Journal of Systems and Software*, 4:163–173, 1984.

2 S. N. Bhatt and F. T. Leighton. A framework for solving VLSI graph layout problems. *J. Comput. Syst. Sci.*, 28(2):300–343, 1984.

3 D. Bienstock and C. L. Monma. On the complexity of embedding planar graphs to minimize certain distance measures. *Algorithmica*, 5(1):93–109, 1990.

4 S. Cabello. Hardness of approximation for crossing number. *Discrete & Computational Geometry*, 49(2):348–358, 2013.

5 S. Cabello and B. Mohar. Crossing number and weighted crossing number of near-planar graphs. *Algorithmica*, 60(3):484–504, 2011.

6 S. Cabello and B. Mohar. Adding one edge to planar graphs makes crossing number and 1-planarity hard. *SIAM J. Comput.*, 42(5):1803–1829, 2013.

7 B. Chazelle. A theorem on polygon cutting with applications. In *Proc. FOCS'82*, pages 339–349. IEEE Computer Society, 1982.

8 M. Chimani. *Computing Crossing Numbers*. PhD thesis, TU Dortmund, Germany, 2008. URL: `http://hdl.handle.net/2003/25955`.

9 M. Chimani and C. Gutwenger. Advances in the planarization method: effective multiple edge insertions. *J. Graph Algorithms Appl.*, 16(3):729–757, 2012.

10 M. Chimani, C. Gutwenger, P. Mutzel, and C. Wolf. Inserting a vertex into a planar graph. In *Proc. SODA'09*, pages 375–383, 2009.

11 M. Chimani and P. Hliněný. A tighter insertion-based approximation of the crossing number. In *Proc. ICALP'11*, volume 6755 of *LNCS*, pages 122–134. Springer, 2011.

12 M. Chimani, P. Hliněný, and P. Mutzel. Vertex insertion approximates the crossing number for apex graphs. *European Journal of Combinatorics*, 33:326–335, 2012.

13 J. Chuzhoy. An algorithm for the graph crossing number problem. In *Proc. STOC'11*, pages 303–312. ACM, 2011.

14 J. Chuzhoy, Y. Makarychev, and A. Sidiropoulos. On graph crossing number and edge planarization. In *Proc. SODA'11*, pages 1050–1069. ACM Press, 2011.

15 É. Colin de Verdière and A. Schrijver. Shortest vertex-disjoint two-face paths in planar graphs. *ACM Transactions on Algorithms*, 7(2):19, 2011.

16 G. Di Battista and R. Tamassia. On-line planarity testing. *SIAM Journal on Computing*, 25:956–997, 1996.

17 G. Even, S. Guha, and B. Schieber. Improved approximations of crossings in graph drawings and VLSI layout areas. *SIAM J. Comput.*, 32(1):231–252, 2002.

18 M. R. Garey and D. S. Johnson. Crossing number is NP-complete. *SIAM J. Alg. Discr. Meth.*, 4:312–316, 1983.

19 I. Gitler, P. Hliněný, J. Leanos, and G. Salazar. The crossing number of a projective graph is quadratic in the face-width. *Electronic Journal of Combinatorics*, 15(1):#R46, 2008.

20 M. Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004.

21 C. Gutwenger and P. Mutzel. A linear time implementation of SPQR trees. In *Proc. GD'00*, volume 1984 of *LNCS*, pages 77–90. Springer, 2001.

22 C. Gutwenger, P. Mutzel, and R. Weiskircher. Inserting an edge into a planar graph. *Algorithmica*, 41(4):289–308, 2005.

23 J. Hershberger and J. Snoeyink. Computing minimum length paths of a given homotopy class. *Comput. Geom.*, 4:63–97, 1994.

24 P. Hliněný and M. Chimani. Approximating the crossing number of graphs embeddable in any orientable surface. In *Proc. SODA'10*, pages 918–927, 2010.

25 P. Hliněný and G. Salazar. On the crossing number of almost planar graphs. In *Proc. GD'05*, volume 4372 of *LNCS*, pages 162–173. Springer, 2006.

26 P. Hliněný and G. Salazar. Approximating the crossing number of toroidal graphs. In *Proc. ISAAC'07*, volume 4835 of *LNCS*, pages 148–159. Springer, 2007.

27 J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM Journal on Computing*, 2(3):135–158, 1973.

28 K-I. Kawarabayashi and B. Reed. Computing crossing number in linear time. In *Proc. STOC'07*, pages 382–390, 2007.

29 P. Klein, S. Rao, M. Rauch, and S. Subramanian. Faster shortest-path algorithms for planar graphs. In *Proc. STOC'94*, pages 27–37, 1994.

30 Y. Kobayashi and C. Sommer. On shortest disjoint paths in planar graphs. *Discrete Optimization*, 7(4):234–245, 2010.

31 D.-T. Lee and F. P. Preparata. Euclidean shortest paths in the presence of rectilinear barriers. *Networks*, 14(3):393–410, 1984.

32 M. Schaefer. The graph crossing number and its variants: A survey. *Electronic Journal of Combinatorics*, #DS21, May 15, 2014.

33 M. Thorup. Undirected single source shortest paths with positive integer weights in linear time. *Journal of the ACM*, 46:362–394, 1999.

34 W. T. Tutte. *Connectivity in graphs*, volume 15 of *Mathematical Expositions*. University of Toronto Press, 1966.

35 T. Ziegler. *Crossing Minimization in Automatic Graph Drawing*. PhD thesis, Saarland University, Germany, 2001.