Crossing-Optimal Extension of Simple Drawings

Robert Ganian
Algorithms and Complexity Group, TU Wien, Austria

Thekla Hamm
Algorithms and Complexity Group, TU Wien, Austria

Fabian Klute
Deptartment of Information and Computing Sciences, Utrecht University, The Netherlands

Irene Parada
TU Eindhoven, The Netherlands

Birgit Vogtenhuber
Graz University of Technology, Austria

Abstract

In extension problems of partial graph drawings one is given an incomplete drawing of an input graph G and is asked to complete the drawing while maintaining certain properties. A prominent area where such problems arise is that of crossing minimization. For plane drawings and various relaxations of these, there is a number of tractability as well as lower-bound results exploring the computational complexity of crossing-sensitive drawing extension problems. In contrast, comparatively few results are known on extension problems for the fundamental and broad class of simple drawings, that is, drawings in which each pair of edges intersects in at most one point. In fact, the extension problem of simple drawings has only recently been shown to be NP-hard even for inserting a single edge.

In this paper we present tractability results for the crossing-sensitive extension problem of simple drawings. In particular, we show that the problem of inserting edges into a simple drawing is fixed-parameter tractable when parameterized by the number of edges to insert and an upper bound on newly created crossings. Using the same proof techniques, we are also able to answer several closely related variants of this problem, among others the extension problem for k-plane drawings. Moreover, using a different approach, we provide a single-exponential fixed-parameter algorithm for the case in which we are only trying to insert a single edge into the drawing.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Computational geometry

Keywords and phrases Simple drawings, Extension problems, Crossing minimization, FPT-algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.72

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2012.07457` [24]

Funding *Robert Ganian*: Supported by the Austrian Science Fund (FWF) via projects Y1329 (*Parameterized Analysis in Artificial Intelligence*) and P31336 (*New Frontiers for Parameterized Complexity*).
Thekla Hamm: Supported by the Austrian Science Fund (FWF) via projects Y1329 (*Parameterized Analysis in Artificial Intelligence*), P31336 (*New Frontiers for Parameterized Complexity*) and W1255-N23.
Fabian Klute: Supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 612.001.651.
Birgit Vogtenhuber: Partially supported by Austrian Science Fund (FWF) within the collaborative DACH project *Arrangements and Drawings* as FWF project I 3340-N35.

Acknowledgements This work was started during the Austrian Computational Geometry Reunion Meeting in Strobl (Austria), August 10 to 14, 2020. We thank all the participants for the nice working atmosphere as well as fruitful discussions on this as well as other topics. The authors would also like to thank Eduard Eiben for his insightful comments.

1 Introduction

The study of the crossing number of graphs, that is, the minimum number of edge crossings necessary to draw a given graph, is a major research direction in the field of computational geometry [10, 33, 36]. More recently, there have been a number of works focusing on minimizing or restricting edge crossings when the task is not to draw a graph from scratch, but rather to extend a partial drawing that is provided on the input. Prominently, Chimani et al. [14] showed that extending a plane drawing with a star in a way that minimizes the number of crossings of the resulting drawing is polynomial-time tractable. Later, Angelini et al. [1] obtained a polynomial-time algorithm for extending plane drawings so that the crossing number remains 0 (i.e., the resulting drawing is plane).

While the two results mentioned above give rise to polynomial-time variants of crossing-minimization extension problems, a number of important cases are known to be NP-hard; a prototypical example is the Rigid Multiple Edge Insertion (RMEI) problem, which asks for a crossing-minimal insertion of k edges into a plane drawing of an n-vertex graph [15, 37]. To deal with this, in recent years the focus has broadened to also consider a weaker notion of tractability, namely, *fixed-parameter tractability* (FPT) [17, 19]. Chimani and Hliněný [15] have shown that RMEI is FPT, i.e., there is an algorithm which solves that problem in time $f(k) \cdot n^{\mathcal{O}(1)}$. Other works have considered various relaxations of crossing minimization; for instance, recently Eiben et al. [20] established the fixed-parameter tractability of extending drawings by k edges in a way which does not minimize the total number of crossings, but rather bounds the number of crossings per edge to at most 1.

For many problems in the intersection of crossing minimization and graph extension, an important goal is that the desired extension should maintain certain properties of the given partial representation. In the problems studied in [1] and [20], the input is a plane or 1-plane[1] drawing, respectively, and the desired extension must maintain the property of being (1-)plane. There have been a plethora of results exploring such extension problems, especially on plane drawings, for a range of other, often more restrictive properties [3, 9, 12, 13, 31, 32, 34].

Beyond planarity, the perhaps most prominent class of drawings with respect to crossing minimization are *simple drawings* (also called *good drawings* [8, 21], *simple topological graphs* [29], or simply *drawings* [26]). A drawing is simple if every pair of edges intersects in at most one point that is either a common endpoint or a proper crossing. Simplicity is an extremely natural restriction that is taken as a basic assumption in a range of settings, e.g., [2, 5, 11, 30], and that constitutes a necessary requirement for crossing-minimal drawings [36].

Contribution. In this work we study the extension problem for simple drawings in the context of crossing minimization. In other words, our aim is to extend a given simple drawing with k new edges while maintaining simplicity and restricting newly created crossings. Naturally, the most obvious way of restricting such crossings is by bounding their number, leading us to our first problem of interest:[2]

[1] A drawing of a graph is ℓ-plane if every edge is involved in at most ℓ crossings.

[2] *Decision* versions of problems are provided purely for complexity-theoretic reasons; every algorithm provided in this article is constructive and can also output a solution as a witness.

Simple Crossing-Minimal Edge Insertion (SCEI)	
Input:	A graph $G = (V, E)$ along with a connected simple drawing $\mathcal{G}$, an integer ℓ, and a set F of k edges of the complement of G.
Question:	Can $\mathcal{G}$ be extended to a simple drawing $\mathcal{G}'$ of the graph $G' = (V, E \cup F)$ such that the number of crossings in $\mathcal{G}'$ that involve an edge of F is at most ℓ?

Note that we require the initial drawing $\mathcal{G}$ to be connected. While this is a natural assumption that is well-justified in many situations, it would certainly also make sense to consider the more general setting in which this is not the case. A short discussion of how the connectivity of $\mathcal{G}$ is used in our proof is provided in Section 4.

SCEI was recently shown to be NP-complete already when $|F| = 1$ and $\ell \geq |E|$ (meaning that the aim is merely to obtain a simple drawing) [7]. On the other hand, dropping the simplicity requirement of the resulting drawing, the problem reduces to RMEI which is FPT.

The main contribution of this article is an FPT algorithm for SCEI parameterized by $k + \ell$. The result is obtained via a combination of techniques recently introduced in [20] and completely new machinery. A high-level overview of challenges posed by the problem and our strategies for overcoming them is provided in the next part of this introduction. Before that, let us mention other natural crossing-sensitive restrictions of simple drawing extension.

Instead of restricting the *total number* of newly created crossings, one may aim to extend $\mathcal{G}$ in a way which bounds the number of crossings involving each of the newly added edges – akin to the restrictions imposed by ℓ-planarity. We call this problem Simple Locally Crossing-Minimal Edge Insertion (SLCEI), where the role of ℓ is that it bounds the maximum number of crossings involving any one particular edge of F. Alternatively, one may simply require that *every* edge in the resulting drawing is involved in at most ℓ crossings, i.e., that the whole $\mathcal{G}'$ is ℓ-plane. This results in the Simple ℓ-Plane Edge Insertion (Sℓ-PEI) problem. Both of these problems are known to be NP-hard when either $\ell = 1$ or $k = 1$, meaning that we can drop neither of our parameters if we wish to achieve tractability.

One key strength of the framework we develop for solving SCEI is its universality. Notably, we obtain the fixed-parameter tractability of SLCEI as an immediate corollary of the proof of our main theorem, while the fixed-parameter tractability of Sℓ-PEI follows by a minor adjustment of the final part of our proof. Moreover, it is trivial to use the framework to solve the considered problems when one drops the requirement that the final drawing is simple – allowing us to, e.g., generalize the previously established fixed-parameter tractability of 1-Planar Edge Insertion [20] to ℓ-Planar Edge Insertion (ℓ-PEI).

Finally, we note that a core ingredient in our approach is the use of Courcelle's theorem [16], and hence the algorithms underlying our tractability results will have an impractical dependency on k. However, for the special case of $|F| = 1$ (i.e., when inserting a single edge), we use so-called *representative sets* to provide a single-exponential fixed-parameter algorithm which is tight under the Exponential Time Hypothesis [27].

Proof Overview. On a high level, our approach follows the general strategy co-developed by a subset of the authors in [20] for solving the problem of inserting k edges into a drawing while maintaining 1-planarity. This general strategy can be summarized as follows:

1. We preprocess G and the planarization of $\mathcal{G}$ to remove parts of $\mathcal{G}$ which are too far away to interact with our solution. This drawing is then translated into a graph representation of bounded *treewidth* [35].
2. We identify a combinatorial characterization that captures how the solution curves will be embedded into $\mathcal{G}$. Crucially, the characterization has size bounded by our parameters.
3. We perform brute-force branching over all characterizations to pre-determine the behavior of a solution in $\mathcal{G}$, and for each such characterization we employ *Courcelle's theorem* [16] to determine whether there exists a solution with such a characterization.

The specific implementation of this strategy differs substantially from the previous work [20] – for instance, the combinatorial characterization of solutions in Step 2 and the use of Courcelle's theorem in Step 3 are both different. But the by far greatest challenge in implementing this strategy occurs in Step 1. Notably, removing the parts of $\mathcal{G}$ required to obtain a bounded-treewidth graph representation creates *holes* in the drawing, and these could disconnect edges intersecting these holes. The graph representation can then lose track of "which edge parts belong to each other", which means we can no longer use it to determine whether the extended drawing is simple. We remark that specifically for Sℓ-PEI and ℓ-PEI, it would be possible to directly adapt Step 1 to ensure that no edge is disconnected in this manner, thus circumventing this difficulty. To handle this problem, we employ an in-depth geometric analysis combined with a careful use of the sunflower lemma and subroutines which invoke Courcelle's theorem to construct a representation which (a) still has bounded treewidth, and (b) contains partial information about which edge parts belong to the same edge in $\mathcal{G}$. A detailed overview of how this is achieved is presented at the beginning of Section 3.

Related Work. There have been two distinct lines of work that recently considered simple drawings in the context of drawing extension problems. The first studied a closely related notion of *saturated* simple drawings [25, 28], while the second studied the computational complexity of the extension problem for simple drawings [6, 7].

Statements where proofs or more details are provided in the full version are marked with ($\star$).

2 Preliminaries

We use standard terminology for undirected and simple graphs [18]. The *length* of a walk or a path is the number of edges it visits. For $r \in \mathbb{N}$, we write $[r]$ as shorthand for the set $\{1, \ldots, r\}$.

A *simple drawing* of a graph G is a drawing $\mathcal{G}$ of G in the plane such that every pair of edges shares at most one point that is either a unique crossing point or a common endpoint. In particular, no tangencies between edges are allowed, edges must not contain any vertices in their relative interior, and no three edges intersect in the same point. Given a simple drawing $\mathcal{G}$ of a graph G and a set of edges F of the complement of G we say that the edges in F can be *inserted* into $\mathcal{G}$ if there exists a simple drawing $\mathcal{G}^+$ of $G^+ = (V(G), E(G) \cup F)$ that contains $\mathcal{G}$ as a subdrawing. The *planarization* of a simple drawing $\mathcal{G}$ of G is the plane graph $\mathcal{G}^\times$ obtained from $\mathcal{G}$ by subdividing the edges of G at the crossing points of $\mathcal{G}$. We call each part of the subdivision of $e \in E(G)$ in $\mathcal{G}^\times$ an *edge segment* (of e). Furthermore, we consider the faces of $\mathcal{G}^\times$ as the *cells* of $\mathcal{G}$ and call $\mathcal{G}$ *connected* if $\mathcal{G}^\times$ is a connected graph.

Sunflower Lemma. One tool we use to obtain our results is the classical sunflower lemma of Erdős and Rado. A *sunflower* in a set family $\mathcal{F}$ is a subset $\mathcal{F}' \subseteq \mathcal{F}$ such that all pairs of elements in $\mathcal{F}'$ have the same intersection.

▶ **Lemma 1** ([22, 23]). *Let $\mathcal{F}$ be a family of subsets of a universe U, each of cardinality at most b, and let $a \in \mathbb{N}$. If $|\mathcal{F}| \geq b!(a-1)^b$, then $\mathcal{F}$ contains a sunflower $\mathcal{F}'$ of cardinality at least a. Moreover, $\mathcal{F}'$ can be computed in time polynomial in $|\mathcal{F}|$.*

Parameterized Complexity. In parameterized complexity [17, 19, 23], the complexity of a problem is studied not only with respect to the input size, but also with respect to some problem parameter(s). The core idea behind parameterized complexity is that the

combinatorial explosion resulting from the NP-hardness of a problem can sometimes be confined to certain structural parameters that are small in practical settings. We now proceed to the formal definitions.

A *parameterized problem* Q is a subset of $\Omega^* \times \mathbb{N}$, where Ω is a fixed alphabet. Each instance of Q is a pair (I, κ), where $\kappa \in \mathbb{N}$ is called the *parameter*. A parameterized problem Q is *fixed-parameter tractable* (FPT) [23, 19, 17], if there is an algorithm, called an FPT-*algorithm*, that decides whether an input (I, κ) is a member of Q in time $f(\kappa) \cdot |I|^{\mathcal{O}(1)}$, where f is a computable function and $|I|$ is the input instance size. The class FPT denotes the class of all fixed-parameter tractable parameterized problems. A parameterized problem Q is FPT-*reducible* to a parameterized problem Q' if there is an algorithm, called an FPT-*reduction*, that transforms each instance (I, κ) of Q into an instance (I', κ') of Q' in time $f(\kappa) \cdot |I|^{\mathcal{O}(1)}$, such that $\kappa' \leq g(\kappa)$ and $(I, \kappa) \in Q$ if and only if $(I', \kappa') \in Q'$, where f and g are computable functions.

Monadic Second Order Logic. We consider *Monadic Second Order* (MSO) logic on (edge-)labeled directed graphs in terms of their incidence structure, whose universe contains vertices and edges; the incidence between vertices and edges is represented by a binary relation. We assume an infinite supply of *individual variables* $x, x_1, x_2, \ldots$ and of *set variables* $X, X_1, X_2, \ldots$. The *atomic formulas* are Vx ("x is a vertex"), Ey ("y is an edge"), Ixy ("vertex x is incident with edge y"), $x = y$ (equality), $P_a x$ ("vertex or edge x has label a"), and Xx ("vertex or edge x is an element of set X"). *MSO formulas* are built up from atomic formulas using the usual Boolean connectives $(\neg, \wedge, \vee, \rightarrow, \leftrightarrow)$, quantification over individual variables ($\forall x$, $\exists x$), and quantification over set variables ($\forall X$, $\exists X$).

Free and bound variables of a formula are defined in the usual way. To indicate that the set of free individual variables of formula Φ is $\{x_1, \ldots, x_\ell\}$ and the set of free set variables of formula Φ is $\{X_1, \ldots, X_q\}$ we write $\Phi(x_1, \ldots, x_\ell, X_1, \ldots, X_q)$. If G is a graph, $v_1, \ldots, v_\ell \in V(G) \cup E(G)$ and $S_1, \ldots, S_q \subseteq V(G) \cup E(G)$ we write $G \models \Phi(v_1, \ldots, v_\ell, S_1, \ldots, S_q)$ to denote that Φ holds in G if the variables x_i are interpreted by the vertices or edges v_i, for $i \in [\ell]$, and the variables X_i are interpreted by the sets S_i, for $i \in [q]$.

The following result (the well-known Courcelle's theorem [16]) shows that if G has bounded treewidth [35] then we can find an assignment φ to the set of free variables $\mathcal{F}$ with $G \models \Phi(\varphi(\mathcal{F}))$ (if one exists) in linear time.

▶ **Theorem 2** (Courcelle's theorem [4, 16]). *Let $\Phi(x_1, \ldots, x_\ell, X_1, \ldots, X_q)$ be a fixed MSO formula with free individual variables $x_1, \ldots, x_\ell$ and free set variables $X_1, \ldots, X_\ell$, and let w a constant. Then there is a linear-time algorithm that, given a labeled directed graph G of treewidth at most w, either outputs $v_1, \ldots, v_\ell \in V(G) \cup E(G)$ and $S_1, \ldots, S_q \subseteq V(G) \cup E(G)$ such that $G \models \Phi(v_1, \ldots, v_\ell, S_1, \ldots, S_q)$ or correctly identifies that no such vertices $v_1, \ldots, v_\ell$ and sets $S_1, \ldots, S_q$ exist.*

We remark that since an understanding of the definition of *treewidth* is not required for our presentation, we merely refer to the literature for a discussion of the notion [17, 19, 35]. We denote the treewidth of a graph G as $\mathrm{tw}(G)$.

Problem Definition and Terminology. We formulate the following generalization of SLCEI in which we allow the numbers of crossings allowed for each newly added edge to differ. Note that this formulation also fixes a parameterization.

Simple Crossing-Restricted Edge Insertion (SCREI) Parameter: $k + \max_{i\in[k]} \ell_i$	
Input:	A graph $G = (V, E)$ along with a connected simple drawing $\mathcal{G}$, a set $F = \{e_1, \ldots, e_k\}$ of k edges of the complement of G, and $\ell_1, \ldots, \ell_k \in \mathbb{N}$.
Question:	Can $\mathcal{G}$ be extended to a simple drawing $\mathcal{G}'$ of the graph $G' = (V, E \cup F)$ such that the drawing of each edge $e_i \in F$ has at most ℓ_i crossings in $\mathcal{G}'$?

For an instance of SCREI we refer to elements in F as *added edges* and denote the endpoints of e_i as s_i and t_i (where $s_1, t_1 \ldots, s_k, t_k$ are not necessarily distinct). For brevity we denote $\ell = \max_{i\in[k]} \ell_i$. Although SCREI is stated as a decision problem, we will want to speak about hypothetical *solutions* of SCREI, which will naturally correspond to the drawings of added edges in $\mathcal{G}'$ (if one exists) as the rest of $\mathcal{G}'$ is predetermined by $\mathcal{G}$. This means that a solution is a set of drawings of added edges in $\mathcal{G}'$ where $\mathcal{G}'$ witnesses the fact that the given instance is a `yes`-instance. If no such $\mathcal{G}'$ exists, then we say that the SCREI-instance *has no solution*.

The reason we focus our presentation on SCREI is that the fixed-parameter tractability of SCREI immediately implies the fixed-parameter tractability of both SLCEI parameterized by the number of added edges and crossings per added edge, and SCEI parameterized by the number of added edges and crossings of all added edges. The former is just a subcase of SCREI. The latter admits a straightforward FPT-reduction to SCREI by branching over the number ℓ_i of crossings each edge $e_i \in F$ is at most involved in. Hence obtaining a fixed-parameter algorithm for SCREI provides a unified reason for the fixed-parameter tractability of both SCEI and SLCEI. Furthermore, we will later show that the result for SCREI can be straightforwardly adapted to solve the other problems mentioned in the introduction.

3 Stitches

Let $\big(G, \mathcal{G}, F, (\ell_i)_{i\in[|F|]}\big)$ be an instance of SCREI. Recalling the Proof Overview provided in Section 1, we want to identify parts of $\mathcal{G}$ that may be considered "unimportant" because they can never be intersected by the drawing of any of the edges $s_it_i \in F$ with at most ℓ_i crossings. Formally, consider the dual G^* of the planarization $\mathcal{G}^\times$ of $\mathcal{G}$, and for each vertex $v \in V(G)$ let $U_v \subseteq G^*$ be the set of vertices that correspond to cells $\mathfrak{c}$ of $\mathcal{G}$ such that v lies on the boundary of $\mathfrak{c}$. We say a cell $\mathfrak{c}$ of $\mathcal{G}$ is *s_it_i-far* if it corresponds to a vertex $v_\mathfrak{c} \in V(G^*)$ at distance more than ℓ_i from U_{s_i} or U_{t_i}, and $\mathfrak{c}$ is *far* if it is s_it_i-far for all $i \in [k]$. Observe that in any solution of SCREI for $\big(G, \mathcal{G}, F, (\ell_i)_{i\in[|F|]}\big)$ no drawing of an edge in F can intersect far cells of $\mathcal{G}$. We refer to maximal unions of far cells in $\mathcal{G}$ which form subsets of $\mathbb{R}^2$ whose interior is connected as *holes*. The interiors of holes are a natural choice for information that is not immediately relevant for the insertion of drawings for F, in the sense that no intersections with these drawings can occur in far cells. However, as mentioned in the Proof Overview, omitting the interior of holes destroys the information about which parts of edges belong to the same edge whenever an edge is disconnected by the removal of a hole.

To transfer this information between different parts of one edge – parts which could be crossed by a hypothetical solution but which are disconnected by holes – we introduce *stitches* into the respective holes. More formally, for a hole H in $\mathcal{G}$ we call an edge e in $E(G)$ *H-torn* if e is split into at least two curves by the removal of the interior of H from $\mathcal{G}$. We call maximal subcurves of an H-torn edge after removing H *(edge) parts* of e and refer to the endpoints of these subcurves as *endpoints* of the corresponding edge part. Stitches will correspond to paths between the endpoints of edge parts of H-torn edges. To construct these paths we introduce so-called *threads* which are edges that we insert into a hole H to connect parts of H-torn edges and derive the stitches from them by considering their planarization.

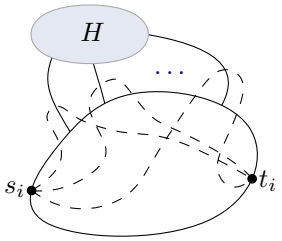

Figure 1 Assuming $\ell_i = 3$, then the potential drawings of s_it_i, depicted as dashed curves, can cross an arbitrary number of H-torn edges.

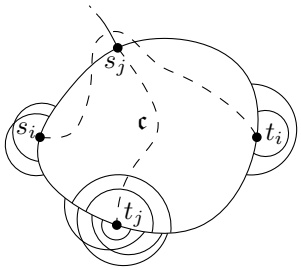

Figure 2 Assuming $\ell_i = \ell_j = 3$, then the drawing of s_jt_j has to go through $\mathfrak{c}$ and the drawing of s_it_i has to revisit cell $\mathfrak{c}$.

To ensure that the obtained combinatorialization of $\mathcal{G}$ has bounded treewidth, the main goal of this section will be to bound the number of stitches for each edge $s_it_i \in F$ and hole H by some function of $k + \ell_i$. We do this by considering which and how many edge parts of H-torn edges any simple s_it_i-curve in a hypothetical solution can cross. Here we face an apparent difficulty: it *is* possible that there is an unbounded number of edge parts which are crossed by drawings of an added edge s_it_i in hypothetical solutions and each edge part belongs to a different H-torn edge (see Figure 1). However, such situations can be safely avoided by restricting our attention to "reasonable" solutions, as we will see in Subsection 3.1. In particular, to specify "reasonable" solutions, we turn our attention to the behavior of drawings of added edges in a hypothetical solution when they *revisit* a cell of $\mathcal{G}$. Then, bounding the number of stitches we introduce for an added edge s_it_i and hole H is equivalent to showing that we can identify all but a bounded number of edges in $E(G)$ which are H-torn and cannot be crossed by a drawing of s_it_i in a "reasonable" hypothetical solution. This is what we focus on in Subsection 3.2.

After adding stitches, we are finally able to define an appropriate combinatorialization of $\mathcal{G}$ in Section 4 which we can use for the final application of Courcelle's theorem in Section 5.

3.1 Detours and Reasonable Solutions

Fix an added edge s_it_i, a hole H, and a cell $\mathfrak{c}$ of the original drawing of G. Note that a drawing of s_it_i in a hypothetical solution might revisit the cell $\mathfrak{c}$ to avoid crossing the drawing of a different added edge s_jt_j. Figure 2 exemplifies such a situation. Understanding how and why a solution might need to revisit a cell is a major component in establishing an upper bound on the number of stitches per hole. In fact, as we will see in this section, avoiding such crossings is the only reason why a cell might have to be revisited.

Let γ_i be a drawing of s_it_i in a hypothetical solution which revisits $\mathfrak{c}$. A $\mathfrak{c}$-*detour* (of γ_i) is a maximal subcurve of γ_i whose interior is disjoint from $\text{int}(\mathfrak{c})$ and has neither s_i nor t_i as an endpoint. Note that a $\mathfrak{c}$-detour might also consist of a singular point. This case occurs when γ_i crosses an edge segment on the boundary of $\mathfrak{c}$ which does not lie on the boundary of another cell. See Figure 3 for an illustration.

▶ **Definition 3.** *Let δ be a $\mathfrak{c}$-detour, and let the embedding $\mathcal{E}$ consist only of δ and the restriction of $\mathcal{G}$ to the boundary of $\mathfrak{c}$. Then δ partitions the boundary of $\mathfrak{c}$ into two connected parts: the part incident to the unbounded (i.e. outer) cell in $\mathcal{E}$, and the* δ-avoided part *which is not incident to the outer cell in $\mathcal{E}$.*

Additionally, we call the subset of $\mathbb{R}^2$ which is enclosed by δ and the δ-avoided part of the boundary of $\mathfrak{c}$ together with the δ-avoided part of the boundary of $\mathfrak{c}$ itself the δ-avoided region.

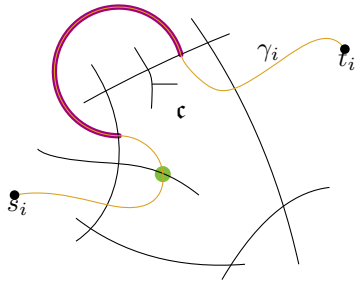

Figure 3 Drawing γ_i of s_it_i in a hypothetical solution with two $\mathfrak{c}$-detours: one is a curve (highlighted in purple) and the other is a point (highlighted in green).

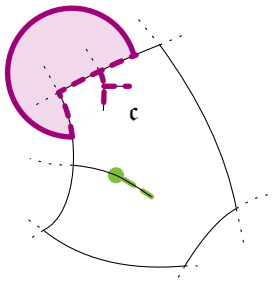

Figure 4 For the single point detour (green), the avoided part of the boundary of $\mathfrak{c}$ and the plane coincide and are dashed green. For the curve detour (purple), the avoided part of the boundary of $\mathfrak{c}$ is dashed purple and the avoided region is shaded purple.

See Figure 4 for an illustration. A $\mathfrak{c}$-detour δ is unremovable *if there exists an added edge s_jt_j with $j \neq i$ such that exactly one of s_j and t_j lies in the δ-avoided region of $\mathcal{G}$. In that case we say that the endpoint (s_j or t_j) in the δ-avoided region is* avoided *by δ, or that δ is* around *the endpoint. We call a $\mathfrak{c}$-detour* removable *if it is not unremovable.*

▶ **Lemma 4 (⋆).** *If there is a solution, then there exists a solution in which no drawing of any added edge contains a removable $\mathfrak{c}'$-detour for any cell $\mathfrak{c}'$ of $\mathcal{G}$.*

Lemma 4 allows us to restrict our attention to solutions which do not contain any removable detours (these are the solutions we intuitively referred to as "reasonable").

3.2 Defining and Finding Stitches

Let $s_it_i \in F$ and H be a hole. Our goal is to compute a, by our parameters, bounded number of edge parts in $E(G)$ which could be crossed by a drawing of s_it_i in some reasonable hypothetical solution. As we obviously do not know any hypothetical solution we cannot compute this set directly. Consequently, we identify and compute a slightly larger set: the set of all edge parts that can be crossed by some so-called *solution curve* for s_i and t_i that is superficially like an s_it_i-curve in a "reasonable" hypothetical solution (but which might induce double-crossings).

▶ **Definition 5.** *A* solution curve *for s_it_i is a simple curve γ_i that (i) starts in s_i and ends in t_i; (ii) produces at most ℓ_i crossings with $\mathcal{G}$; and (iii) whenever γ_i intersects a cell $\mathfrak{c}'$ in more than one maximal connected subcurve there is an added edge s_jt_j with $j \neq i$ such that exactly one of s_j and t_j lies in the ζ-avoided part of $\mathcal{G}$, where ζ is a maximal connected subcurve of γ_i outside of $\mathfrak{c}'$ between two intersections of γ_i with $\mathfrak{c}'$. A part of an H-torn edge $e \in E(G)$ is* crossable *for s_it_i if it is crossed by a solution curve for s_it_i.*

▶ **Lemma 6 (⋆).** *For every hole H and every added edge $s_it_i \in F$ there are less than $\ell_i(2\ell_i+1)! \cdot \left(4k(\ell_i+2)(\ell_i+1)^{\ell_i+1}\right)^{2\ell_i+1}$ parts of H-torn edges that are crossable for s_it_i.*

Proof Sketch. We show that there is a set K of less than $(2\ell_i+1)!\left(4k(\ell_i+2)(\ell_i+1)^{\ell_i+1}\right)^{2\ell_i+1}$ solution curves for s_it_i such that each crossable edge part for s_it_i is crossed by at least one of the curves in K. Then the claim follows as each solution curve crosses at most ℓ_i edges.

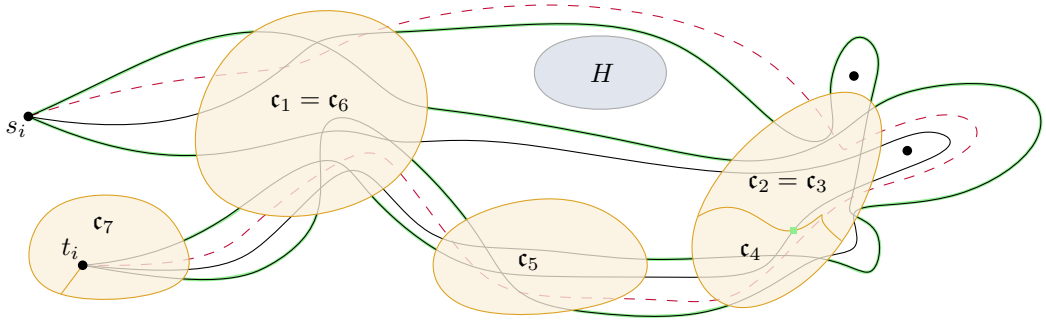

Figure 5 The cells $\mathfrak{c}_1, \ldots, \mathfrak{c}_7$ are in the core of the sunflower. The red dashed s_it_i curve cannot be part of the minimal set of curves K. The extremal subcurves are highlighted in green.

Assume for contradiction that the minimum set K that witnesses crossability of parts of H-torn crossable edges for s_it_i consists of at least $(2\ell_i+1)!\left(4k(\ell_i+2)(\ell_i+1)^{\ell_i+1}\right)^{2\ell_i+1}$ solution curves for s_it_i. Consider the restricted drawing $\mathcal{G}_H$ which is given by $\mathcal{G}$ restricted to the boundary of H, all H-torn edges in $E(G)$, as well as s_i and t_i.

We associate each s_it_i curve in K with the set of cells of $\mathcal{G}_H$ which it intersects and the set of edge segments in $\mathcal{G}_H^\times$ which it crosses. In this way, each curve in K is associated to a set of size at most $2\ell_i+1$. By the minimality of K, no two curves in K are associated to the same set of cells and edge segments. Using the sunflower lemma [22, 23] for the set system given by the sets of cells and edge segments associated to the s_it_i curves in K we obtain a set of at least $4k(\ell_i+2)(\ell_i+1)^{\ell_i+1}$ solution curves $K^\sunflower \subseteq K$ which all intersect pairwise different cells of $\mathcal{G}_H$ and edge segments of $\mathcal{G}_H^\times$, apart from the cells and edge segments in the core of a sunflower, which they all intersect. Moreover, as curves in K intersect at most ℓ_i edges we find at most ℓ_i+1 cells in the core.

By the pigeonhole principle there is a set of at least $4k(\ell_i+2)$ curves in $K^\sunflower$ which all intersect the cells in the core of the sunflower in the same order (taking into account repetitions of cells). Let $K^\sunflower_\sigma \subseteq K^\sunflower$ be such a set of curves and let $\sigma = \mathfrak{c}_1, \ldots, \mathfrak{c}_l$ with $l \leq \ell_i+1$ be the order in which these curves traverse the cells in the core of the sunflower.

As each $\mathfrak{c}_j$ with $j \in [l]$ is a cell in a restriction of $\mathcal{G}$ containing all H-torn edges, no part of an H-torn edge intersects the interior of $\mathfrak{c}_j$. In particular parts of H-torn edges are not crossed by any curve in $K^\sunflower_\sigma$ within $\mathrm{int}(\mathfrak{c}_\mathrm{j})$.

When considering subcurves of curves *between* each $\mathfrak{c}_j$ and $\mathfrak{c}_{j+1}$, we can find at most $4(k-1)$ "extremal" such subcurves which separate all other subcurves from H together with $\mathfrak{c}_j$ and $\mathfrak{c}_{j+1}$. These extremal subcurves together cross any crossable edge part of an H-torn edge intersected by any other considered subcurve. See Figure 5 for an illustration.

In this way we find at least one curve after the removal of which from K the same crossable edge parts of H-torn edges are intersected, contradicting our minimality assumption. ◀

While the fact that the number of crossable edge parts we want to introduce stitches for is bounded by a function in our parameters is reassuring, we need to be able to actually introduce these stitches before being able to give our final MSO encoding of hypothetical solutions. For this we invoke Courcelle's theorem in Lemma 7 independently of its final application. This then allows us to insert the corresponding stitches.

▶ **Lemma 7 (⋆).** *There is a fixed-parameter algorithm parameterized by $k+\ell$ which identifies, for an added edge s_it_i and a hole H, all parts of H-torn edges which are crossable for s_it_i.*

▶ **Definition 8.** *For a hole H in $\mathcal{G}$ and an added edge $s_it_i \in F$, a* thread *is a pair of two endpoints of two distinct edge parts of the same H-torn edge in $e \in E(G)$ satisfying the following properties: (i) both edge parts are crossable for s_it_i, (ii) there is no other crossable edge part between these edge parts along a traversal of e, and (iii) there is no other endpoint of one of the two edge parts along a traversal of e. We denote the set of all threads for H and s_it_i as T_{H,s_it_i}, and define the set of all threads for H as $T_H = \bigcup_{i\in[k]} T_{H,s_it_i}$.*

An embedding of T_H is a set of curves, contained completely in H, which connect each pair of two endpoints of edge parts in T_H.

▶ **Lemma 9 (⋆).** *There is a fixed-parameter algorithm parameterized by $k + \ell$ that computes, for a hole H in $\mathcal{G}$, a simple embedding of T_H.*

For the simple embedding of T_H into H computed in Lemma 9, define the set of *stitches* S_H of H as the planarization of the threads in this embedding.

4 The Patchwork Graph

After identifying a bounded number of stitches in each hole, we are finally able to define the *patchwork graph* and prove desirable properties which we will use in our final application of Courcelle's theorem. An illustration of the patchwork graph is provided in Figure 6. The following definition also doubles as a description of how to construct the patchwork graph from a given drawing. We remark that, unlike $\mathcal{G}$, the patchwork graph might be disconnected.

▶ **Definition 10.** *The* patchwork graph *P and its embedding $\mathcal{P}$ are given by the labeled graph derived from $\mathcal{G}$ in the following steps:*

1. *Planarize $\mathcal{G}$ and label the vertices which are newly introduced by this as* crossing vertices*. Label vertices which correspond to vertices of G as* real vertices*. Additionally label each s_i and t_i with label $i \in [k]$.*
2. *Subdivide each edge e in the planarization $\mathcal{G}^\times$ of $\mathcal{G}$ by k vertices[3] $v_1^e, \ldots, v_k^e$ which are labeled as* segment vertices *– each segment vertex of e will represent a possible crossing point of the drawing of one of the k edges in F and e.*
3. *Inside each face f of $\mathcal{G}^\times$, introduce a new vertex v_f and label it as* cell vertex*.*
4. *Inside each face f of $\mathcal{G}^\times$, trace the boundary of f creating a curve at ε-distance and create a vertex labeled as* shadow vertex *on this curve every time an endpoint of an edge in F or a segment vertex is encountered. Insert two edges for each shadow vertex; one connecting the shadow vertex to the corresponding endpoint of an edge in F or segment vertex; and one connecting the shadow vertex to v_f. Note that multiple shadow vertices can be introduced for the same vertex in G (e.g. the orange vertex in Figure 6). Shadow vertices allow to distinguish different ways, more formally positions in the rotation around an endpoint, of accessing that endpoint via the inserted drawing of an edge in F (see Figure 7); this is where the connectivity of $\mathcal{G}$ is used. In this way each shadow vertex of an endpoint corresponds to an* access direction*.*
5. *Delete every vertex that is in the interior of a hole H.*
6. *For each hole H insert all stitches S_H for H into the interior of H and label the inserted vertices as* crossing vertices*.*[4]

[3] If $k = 1$ we subdivide by 2 vertices for reasons that will become clear when we introduce *tracking labels*.
[4] This means they receive the same label as vertices introduced by planarizing $\mathcal{G}$.

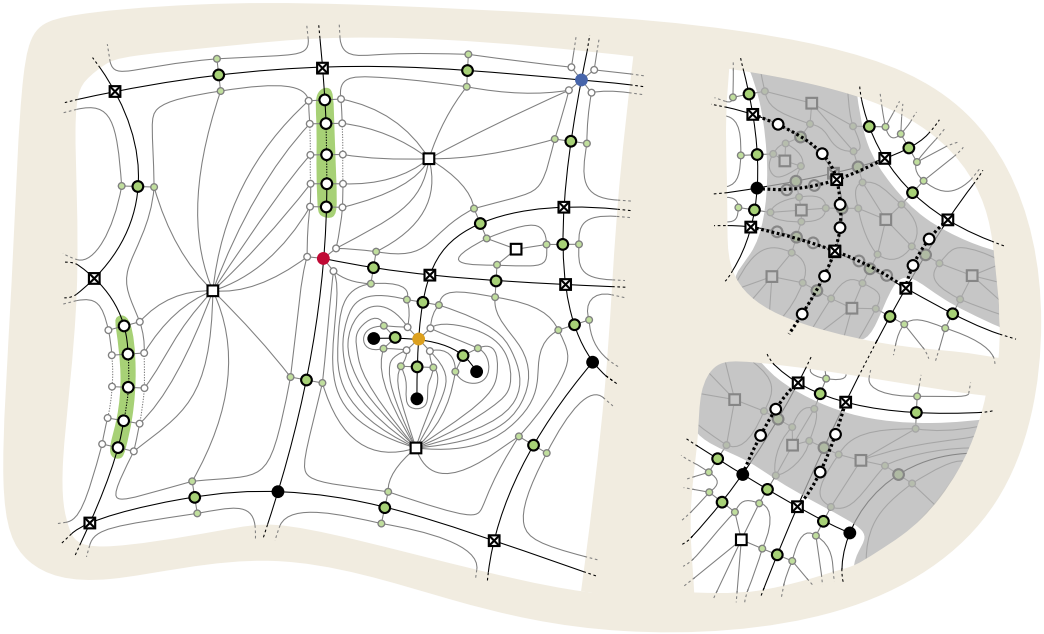

Figure 6 Illustration of a patchwork graph P. The remainder of P is hinted in beige. Black disks are original vertices. Colored disks are endpoints of edges in F. Crossing vertices are crosses. Green and white disks represent the edge segment/shadow vertices. Cell vertices are white squares. Holes are shaded in gray and stitches drawn with thick, dashed curves.

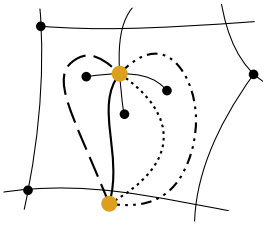

Figure 7 Illustration for different access directions. Each hypothetical drawing (indicated as thick dashes, normal, dotted, and dash-dotted lines) of the added edge between the orange vertices crosses the same edge segment of $\mathcal{G}^\times$ but separates the black vertices differently. In connected initial drawings, ways of separating vertices of the same cell by the drawing of an added edge are completely determined by potential crossing points of that drawing and its positions in the rotations around each of its endpoints. This is not the case for disconnected initial drawings.

7. *For technical reasons which will become apparent later (when we introduce* tracking labels*), we replace each edge in S_H by a path consisting of two vertices and three edges and label the inserted vertices as* segment vertices.[5]

We introduce additional crossability labels for segment vertices in the following way. For every segment vertex v corresponding to an edge segment σ of edge $e \in E(G)$, we label v as *crossable* for some edge $s_it_i \in F$ if one of the following two conditions holds:

- e is not H-torn for any hole H, or
- for *each* hole H in $\mathcal{G}$ for which e is H-torn, σ lies on a part (when considering parts arising from the removal of the interior of H) of e that is crossable for s_it_i.

[5] This means they receive the same label as vertices introduced by subdividing edge segments of the planarization of $\mathcal{G}$.

▶ **Lemma 11 (⋆).** *If there exists a solution for the considered SCREI instance, then there is a solution such that all segment vertices which correspond to edge segments of an edge that is crossed by the drawing of $s_it_i \in F$ in the solution are labeled as crossable for s_it_i.*

Note that Lemmas 7 and 9 and Definition 10 allow us to compute the patchwork graph in FPT time. Two important properties of the patchwork graph are encapsuled in Lemmas 12 and 13. The proof of Lemma 12 relies on obtaining a bound on the diameter of each connected component of the patchwork graph – a task which is intuitively clear, but requires to overcome technical challenges due to the addition of stitches. Lemma 13 later allows us to encode whether two edge segments in P belong to the same edge in $\mathcal{G}$ via an MSO formulation.

▶ **Lemma 12 (⋆).** *The patchwork graph P has treewidth bounded by $3(2 + 4(k-1))(4\ell + 8(kf(k,\ell)-1))$, where $f(k,\ell)$ is the bound on the number of crossable edge parts for a single added edge and hole obtained in Lemma 6.*

▶ **Lemma 13 (⋆).** *Segment vertices which correspond to edge segments of the same edge in $e \in E(G)$ and are labeled as crossable for s_it_i are connected via paths in P consisting only of segment and crossing vertices which correspond to segments and crossings of e and segments and crossings for threads that connect parts of e.*

Ideally, we would like Lemma 13 to lead to an MSO subformula that can check whether two segment vertices in P belong to the same edge – an important component of our algorithm for SCREI. The lemma provides us with a characterization that seems suitable for this task since it is easy to define a path in MSO, but there is an issue if we use P as it is currently defined: a crossing vertex is adjacent to 4 segment vertices, and P (viewed as a graph without an embedding) does not specify which of these segment vertices belong to the same edge. We resolve this by introducing *tracking labels*: for each crossing vertex v in P created by a crossing between edges e and e' in $\mathcal{G}$, we assign the label 1 to the two unique neighbors of v corresponding to e and the label 2 to the remaining two neighbors of v.

▶ **Corollary 14.** *Segment vertices which correspond to edge segments of the same edge in $e \in E(G)$ and are labeled as crossable for s_it_i are connected via paths in P consisting only of segment and crossing vertices with the following property: the two neighbors of each crossing vertex on the path are segment vertices with the same tracking label.*

5 Using the Patchwork Graph

Now that we have constructed the patchwork graph P and established that it has the properties we need, we can proceed to the final stage of our proof. Here, our aim will be to identify a combinatorial characterization which projects the behavior of a solution from $\mathcal{G}$ to P, establish a procedure that allows us to identify (and construct) solutions based on a characterization in P, and finally show how to find such characterizations. To streamline our presentation, at this stage we perform a brute-force branching procedure which will determine, for each $s_it_i \in F$, the number ℓ'_i of crossings between the curve connecting s_i to t_i and edges of $\mathcal{G}$ in the sought-after solution.

Consider a hypothetical solution S, and let f be a curve in S connecting vertex a to b. The *trace* r_f of f is a walk in P starting at a such that:

1. From a, r_f proceeds to the shadow vertex that corresponds to the access direction through which f connects to a, and then to the cell vertex of the first cell $\mathfrak{c}_1$ in $\mathcal{G}$ intersecting f.
2. For each intersection along f with an edge segment q between cells $\mathfrak{c}_i$ and $\mathfrak{c}_{i+1}$, r_f proceeds to the shadow vertex of a segment vertex v in $\mathfrak{c}_i$ on q, then to v, then to its shadow vertex in $\mathfrak{c}_{i+1}$, and then to the cell vertex of $\mathfrak{c}_{i+1}$, where v has the property that the number of

segment vertices of q on either side of v is at least as large as the number of drawings of added edges in F which intersect q on the respective side of its intersection with f. Such a segment vertex v exists, since there are $k = |F|$ segment vertices on q.

3. Finally, r_f continues to the shadow vertex that *corresponds* to the direction through which f enters b, and finally ends in b.

Observe that r_f visits precisely $4\ell'_i + 5$ vertices. Moreover, for two curves f, f' in S, their traces $r_f, r_{f'}$ may only intersect in cell vertices, the real vertices that form the endpoints of the curves, and the associated shadow vertices.

Now, let the *solution trace* (r_S,η_S) of S be a pair where $r_S = \{r_f | f \in S\}$ and η_S describes cyclic orders which will intuitively capture how edges cross into and out of each cell vertex in the solution. Let $R_S = \{v \mid \exists f \in S : v \in r_f\}$ be the set of all vertices occurring in the traces of S. η_S then is a mapping from each cell vertex $c \in R_S$ to a cyclic order $\prec_c$ over the shadow vertices in R_S that are incident to c. Specifically, $\prec_c$ is defined as the cyclic order given by the cycle on the neighborhood of c in P restricted to R_S.

Solution traces describe the way in which a solution can be related to a set of walks and cyclic orders in P. Of course we can abstract away from the explicit reference to a solution and define the more general notion of *preimages* whose combinatorial structure is the same as that of a solution trace but which does not arise and in particular does not even need to correspond to a solution. (Preimages and solution traces relate in a similar way as solution curves and solutions in Section 3.2.)

Formally, a *preimage* (α', β') is a tuple with the following properties. α' is a set of k walks in H which are labeled $\alpha'_1, \ldots, \alpha'_k$, where each α'_i has length $4(\ell_i + 1)$ and visits vertices with the same orders of labels as traces. Similarly, β' is a mapping from each cell vertex c visited by the walks in α' to the cyclic order over its neighbors that occur in α', along the cycle on $N_P(c)$ in P.

Obviously every solution trace is a preimage. Conversely, one can derive a drawing of all edges of F into $\mathcal{G}$ from a preimage (α', β') by the *assembly procedure* **A** introduced below. For each $\alpha'_i \in \alpha'$, **A** will draw a curve u_i that starts and ends at the two vertices labeled i (i.e., the endpoints of $s_i t_i \in F$) as described in the following steps.

1. u_i exits its starting vertex via the access direction given by the first shadow vertex in α'_i.
2. For each cell vertex c such that (e_1, v_1, c, v_2, e_2) forms a subsequence of visited vertices in α'_i, expand u_i by drawing a curve ι in c connecting the edge segment (or the real vertex) e_1 to the edge segment (or the real vertex) e_2 in the following way.
   - Consider an arbitrary other curve drawn in c by **A** up to now, say ζ, that was obtained from some subsequence $(e_1^\zeta, v_1^\zeta, c, v_2^\zeta, e_2^\zeta)$. ι will intersect ζ if and only if the shadow vertices of ι interleave with the shadow vertices of ζ in $\beta(c)$ (i.e., for instance, if $v_1 \prec_c v_1^\zeta \prec_c v_2 \prec_c v_2^\zeta \prec_c v_1$).
   - Such a drawing can be achieved by, e.g., having the curve ι follow the inside boundary of c in a clockwise manner while avoiding all curves it is not supposed to cross (as these will be either completely enveloped by or completely enveloping ι).
   - We remark that v_1 and v_2 may either be shadows of segment vertices or the actual endpoints s_i or t_i.
3. u_i ends by entering the final real vertex in α'_i from the direction specified by the last shadow vertex in α'_i.

The intuition here is that **A** interprets a preimage of a template trace as a specification of precisely which parts of $\mathcal{G}$ should be crossed by the drawings of each added edge (this information is provided in α'), while controlling when and how individual curves in the newly constructed solutions should cross each other (this information is provided in β'). Note that the output of **A** for an arbitrary preimage will in general not be a solution for our edge insertion problem, but – crucially – one can check whether it is in polynomial time.

Observe that, although preimages imply curves in $\mathcal{G}$ for all added edges in F, and we can check for each of them if they are a solution, we cannot iterate over them in FPT time as the number of preimages in P is generally not FPT. We will however be able to distill the structure of preimages, independently of their exact specification in P. For this we define *template traces*. A template trace is a tuple $\tau = (T, \alpha, \beta)$ where:

- T is a graph whose vertices are equipped with a labeling that matches the vertex-labeling used in P (i.e., some may be labeled as segment vertices, some as cell vertices, etc., and in addition some of them may be labeled as the endpoints of added edges in F);
- $\alpha = \{\alpha_1, \ldots, \alpha_k\}$ is a set of walks in T, where each walk α_i has length $4(\ell'_i + 1)$ and the types of vertices visited by α_i match the types of vertices visited by a trace (i.e., α_i starts with a real vertex labeled i, then proceeds with a shadow vertex, a cell vertex, followed by a sequence of ℓ'_i-many subsequences of shadow-, segment-, shadow-, cell vertices, and ends with a shadow vertex followed by a different real vertex labeled i); and
- β is a mapping from each cell vertex in T to a cyclic order over its adjacent shadow vertices.
- For simplicity, we require that each vertex and edge in T occurs in at least one walk in α.

▶ **Proposition 15 (⋆).** *There are at most $(k\ell)^{\mathcal{O}(k\ell)}$ distinct template traces. Moreover, the set of all template traces can be enumerated in time $(k\ell)^{\mathcal{O}(k\ell)}$.*

We say that a template trace (T, α, β) *matches* a preimage (α', β') if there is a label-preserving bijective mapping γ (called the *preimaging*) from the vertices on walks in α' to $V(T)$ such that (1) for each $\alpha'_i \in \alpha'$, $\gamma(\alpha'_i) = \alpha_i$ and (2) $\gamma(\beta')$ maps each c to $\beta(\gamma(c))$. For a template trace τ that matches a preimage (α', β'), we say that (α', β') is a *preimage* of τ. Intuitively, a preimage of a template trace is its firmly embedded counterpart in P. As every solution trace is a preimage, these definitions carry over to solution traces.

The following lemma shows that a template trace τ matching the solution trace of a hypothetical solution contains a sufficient amount of information to *almost* reconstruct a solution using $\mathbf{A}$ on a preimage of τ.

▶ **Lemma 16 (⋆).** *Let S be a solution which matches a template trace $\tau = (T, \alpha, \beta)$, and let (α', β') be a preimage of τ. Let S' be the output of $\mathbf{A}$ applied to (α', β'). Then S' is either a solution, or there exists an edge e of G that intersects some curve in S' more than once.*

Next, we show that the problem of finding a preimage of a template trace (or determining that there is none) can be encoded in Monadic Second Order (MSO) logic. Which is the last ingredient needed to prove our main result.

▶ **Lemma 17 (⋆).** *Let $\tau = (T, \alpha, \beta)$ be a template trace. There exists an MSO formula $\phi_\tau(V(T))$ of size independent of G and $\mathcal{G}$ which is satisfiable in P if and only if there exists a preimage for τ in P. Moreover, if the formula is true, then the interpretation of $V(T)$ defines a* preimaging *between a preimage of τ and τ.*

▶ **Theorem 18 (⋆).** *SCREI is fixed-parameter tractable.*

Theorem 18 implies the fixed-parameter tractability of SCEI and SLCEI parameterized by $k + \ell$. Moreover, the approach can also be used to obtain fixed-parameter tractability of the other problems defined in the introduction, with only minor adaptations required.

▶ **Theorem 19 (⋆).** *Sℓ-PEI, ℓ-PEI and Locally Crossing-Minimal Edge Insertion are fixed-parameter tractable when parameterized by $\ell + k$.*

6 Inserting a Single Edge

In this section we present a single-exponential fixed-parameter algorithm for SCEI parameterized by ℓ in the case where $|F| = 1$; we hereinafter denote this problem SC1EI. We remark that the parameter dependency of this algorithm is tight under the Exponential Time Hypothesis [27], since Arroyo et al. [7] gave a reduction from 3-SAT to the simple drawing extension problem with one extra edge, and the number of edges in the obtained graphs is linear in the size of the 3-SAT instance. We note that in the same work [7], the authors also presented a single-exponential parameterized algorithm for SCEI when $|F| = 1$, however the parameter used there is the total number of crossings in the original drawing.

As a first step we transform SC1EI to the problem of finding a colorful st-path (i.e., a path where no color is repeated) of length at most κ in a vertex-colored graph with coloring χ obtained from $\mathcal{G}^\times$. Using so-called *representative sets*, see e.g. [17, Chapter 12], we can show how to efficiently find a colorful path. Theorem 22 is then an immediate consequence.

▶ **Proposition 20 (⋆).** *There is a linear-time reduction that converts an instance $(G, \mathcal{G}, \{st\}, \ell)$ of SC1EI to an equivalent instance $(G^*, \chi, s, t, 2\ell + 3)$ of* Colorful Short Path.

▶ **Theorem 21 (⋆).** *Colorful Short Path can be solved in time $\mathcal{O}(2^{\mathcal{O}(\kappa)} \cdot |E(G)| \log |V(G)|)$.*

▶ **Theorem 22.** *SC1EI can be solved in time $\mathcal{O}(2^{\mathcal{O}(\ell)} \cdot |\mathcal{G}| \log |E(G)|)$.*

7 Conclusion

In this paper we established the fixed-parameter tractability of inserting a given set of edges into a given drawing while maintaining simplicity and adhering to various restrictions on the number of crossings in the solution. While the presented results make the reasonable assumption that the initial drawing is connected, the problem is of course also interesting in the general case. We believe that our framework and methodology can also be used to handle the extension problem for disconnected drawings, albeit only after overcoming a few additional technical challenges; moreover, the algorithm presented in Section 6 does not require connectivity at all. Other than connectivity, the most glaring question left open concerns the complexity of SCEI parameterized by ℓ alone. Recall that, in contrast to this open question, SLCEI is known to be NP-hard already for $\ell = 1$. Last but not least, while here we focused on the edge insertion problem, it would also be interesting to extend the scope to also allow for the addition of vertices into the drawing.

References

1 Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Vít Jelínek, Jan Kratochvíl, Maurizio Patrignani, and Ignaz Rutter. Testing planarity of partially embedded graphs. *ACM Transactions on Algorithms*, 11(4):32:1–32:42, 2015. doi:10.1145/2629341.

2 Patrizio Angelini, Michael A. Bekos, Franz J. Brandenburg, Giordano Da Lozzo, Giuseppe Di Battista, Walter Didimo, Michael Hoffmann, Giuseppe Liotta, Fabrizio Montecchiani, Ignaz Rutter, and Csaba D. Tóth. Simple k-planar graphs are simple $(k+1)$-quasiplanar. *Journal of Combinatorial Theory, Series B*, 142:1–35, 2020. doi:10.1016/j.jctb.2019.08.006.

3 Patrizio Angelini, Ignaz Rutter, and Sandhya T. P. Extending Partial Orthogonal Drawings. In *Proceedings of the 28th International Symposium on Graph Drawing and Network Visualization (GD'20)*, volume 12590 of *LNCS*, pages 265–278. Springer, 2020. doi:10.1007/978-3-030-68766-3_21.

4 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, 1991. doi:10.1016/0196-6774(91)90006-K.

5 Alan Arroyo, Julien Bensmail, and R. Bruce Richter. Extending drawings of graphs to arrangements of pseudolines. In *Proceedings of the 36th International Symposium on Computational Geometry (SoCG'20)*, volume 164 of *LIPIcs*, pages 9:1–9:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.SoCG.2020.9.

6 Alan Arroyo, Martin Derka, and Irene Parada. Extending simple drawings. In *Proceedings of the 27th International Symposium on Graph Drawing and Network Visualization (GD'19)*, volume 11904 of *LNCS*, pages 230–243. Springer, 2019. doi:10.1007/978-3-030-35802-0_18.

7 Alan Arroyo, Fabian Klute, Irene Parada, Raimund Seidel, Birgit Vogtenhuber, and Tilo Wiedera. Inserting one edge into a simple drawing is hard. In *Proceedings of the 46th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'20)*, volume 12301 of *LNCS*, pages 325–338. Springer, 2020. doi:10.1007/978-3-030-60440-0_26.

8 Alan Arroyo, Dan McQuillan, R. Bruce Richter, and Gelasio Salazar. Levi's lemma, pseudolinear drawings of K_n, and empty triangles. *Journal of Graph Theory*, 87(4):443–459, 2018. doi:10.1002/jgt.22167.

9 Guido Brückner and Ignaz Rutter. Partial and constrained level planarity. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'17)*, pages 2000–2011. SIAM, 2017. doi:10.1137/1.9781611974782.130.

10 Christoph Buchheim, Markus Chimani, Carsten Gutwenger, Michael Jünger, and Petra Mutzel. Crossings and planarization. In Roberto Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 43–85. Chapman and Hall/CRC, 2013.

11 Jean Cardinal and Stefan Felsner. Topological drawings of complete bipartite graphs. *Journal of Computational Geometry*, 9(1):213–246, 2018. doi:10.20382/jocg.v9i1a7.

12 Erin W. Chambers, David Eppstein, Michael T. Goodrich, and Maarten Löffler. Drawing graphs in the plane with a prescribed outer face and polynomial area. *Journal of Graph Algorithms and Applications*, 16(2):243–259, 2012. doi:10.7155/jgaa.00257.

13 Timothy M. Chan, Fabrizio Frati, Carsten Gutwenger, Anna Lubiw, Petra Mutzel, and Marcus Schaefer. Drawing partially embedded and simultaneously planar graphs. *Journal of Graph Algorithms and Applications*, 19(2):681–706, 2015. doi:10.7155/jgaa.00375.

14 Markus Chimani, Carsten Gutwenger, Petra Mutzel, and Christian Wolf. Inserting a vertex into a planar graph. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA'09)*, pages 375–383. SIAM, 2009.

15 Markus Chimani and Petr Hlinený. Inserting multiple edges into a planar graph. In *Proceedings of the 32nd International Symposium on Computational Geometry (SoCG'16)*, volume 51 of *LIPIcs*, pages 30:1–30:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.SoCG.2016.30.

16 Bruno Courcelle. The monadic second-order logic of graphs I: recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

17 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

18 Reinhard Diestel. *Graph Theory, 5th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

19 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1145/2744447.2744454.

20 Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, and Martin Nöllenburg. Extending partial 1-planar drawings. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP'20)*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.43.

21 Paul Erdős and Richard K. Guy. Crossing number problems. *The American Mathematical Monthly*, 80(1):52–58, 1973. doi:10.1080/00029890.1973.11993230.

22 Paul Erdős and Richard Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 1(1):85–90, 1960.

23 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*, volume XIV of *Texts in Theoretical Computer Science. An EATCS Series*. Springer, Berlin, 2006. doi:10.1007/3-540-29953-X.

24 Robert Ganian, Thekla Hamm, Fabian Klute, Irene Parada, and Birgit Vogtenhuber. Crossing-optimal extension of simple drawings. *CoRR*, abs/2012.07457, 2020. arXiv:2012.07457.

25 Péter Hajnal, Alexander Igamberdiev, Günter Rote, and André Schulz. Saturated simple and 2-simple topological graphs with few edges. *Journal of Graph Algorithms and Applications*, 22(1):117–138, 2018. doi:10.7155/jgaa.00460.

26 Heiko Harborth. Empty triangles in drawings of the complete graph. *Discrete Mathematics*, 191(1-3):109–111, 1998. doi:10.1016/S0012-365X(98)00098-3.

27 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

28 Jan Kynčl, János Pach, Radoš Radoičić, and Géza Tóth. Saturated simple and k-simple topological graphs. *Computational Geometry: Theory and Application*, 48(4):295–310, 2015. doi:10.1016/j.comgeo.2014.10.008.

29 Jan Kynčl. Enumeration of simple complete topological graphs. *European Journal of Combinatorics*, 30(7):1676–1685, 2009. doi:10.1016/j.ejc.2009.03.005.

30 Jan Kynčl. Simple realizability of complete abstract topological graphs simplified. *Discrete and Computational Geometry*, 64(1):1–27, 2020. doi:10.1007/s00454-020-00204-0.

31 Giordano Da Lozzo, Giuseppe Di Battista, and Fabrizio Frati. Extending upward planar graph drawings. *Computational Geometry: Theory and Applications*, 91:101668, 2020. doi:10.1016/j.comgeo.2020.101668.

32 Tamara Mchedlidze, Martin Nöllenburg, and Ignaz Rutter. Extending convex partial drawings of graphs. *Algorithmica*, 76(1):47–67, 2016. doi:10.1007/s00453-015-0018-6.

33 János Pach. Geometric graph theory. In Csaba D. Tóth, Joseph O'Rourke, and Jacob E. Goodman, editors, *Handbook of Discrete and Computational Geometry, Third Edition*, pages 257–279. CRC press, 2017. doi:10.1201/9781315119601.

34 Maurizio Patrignani. On extending a partial straight-line drawing. *International Journal of Foundations of Computer Science*, 17(5):1061–1070, 2006. doi:10.1142/S0129054106004261.

35 Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984. doi:10.1016/0095-8956(84)90013-3.

36 Marcus Schaefer. *Crossing numbers of graphs*. CRC Press, 2018. doi:10.1201/9781315152394.

37 Thomas Ziegler. *Crossing minimization in automatic graph drawing*. PhD thesis, Saarland University, Saarbrücken, Germany, 2001.