Computing Crossing Numbers in Quadratic Time

Martin Grohe*

November 29, 2002

Abstract

We show that for every fixed $k \geq 0$ there is a quadratic time algorithm that decides whether a given graph has crossing number at most k and, if this is the case, computes a drawing of the graph into the plane with at most k crossings.

1. Introduction

Hopcroft and Tarjan [13] showed in 1974 that planarity of graphs can be decided in linear time. It is natural to relax planarity by admitting a small number of edge-crossings in a drawing of the graph. The *crossing number* of a graph is the minimum number of edge crossings needed in a drawing of the graph into the plane. Not surprisingly, it is NP-complete to decide, given a graph G and a k, whether the crossing number of G is at most k [12]. On the other hand, for every *fixed* k there is a simple polynomial time algorithm deciding whether a given graph G has crossing number at most k: It guesses $l \leq k$ pairs of edges that cross[1] and tests if the graph obtained from G by adding a new vertex at each of these edge crossings is planar. The running time of this algorithm is $n^{\Theta(k)}$. Downey and Fellows [7] raised the question of whether the crossing-number problem is *fixed parameter-tractable*, that is, whether there is a constant $c \geq 1$ such that for every fixed k the problem can be solved in time $O(n^c)$. We answer this question positively with $c = 2$. In other words, we show that for every fixed k there is a quadratic time algorithm deciding whether a given graph G has crossing number at most k. Moreover, we show that if this is the case a drawing of G into the plane with at most k crossings can be computed in quadratic time.

It is interesting to compare our result to similar results for computing the *genus* of a graph. (The genus of a graph G is the minimum taken over the genera of all surfaces S such that G can be embedded into S.) As for the crossing number, it is NP-complete to decide if the genus of a given graph is less than or equal to a given k [18]. For a fixed k, at first sight the genus problem looks much harder. It is by no means obvious how to solve it in polynomial time; this has been proved possible by Filotti, Miller, and Reif [10]. In 1996, Mohar [14] proved that for every k there is actually a linear time algorithm deciding whether the genus of a given graph is k. However, the fact that the genus problem is fixed-parameter tractable was known earlier as a direct consequence of a strong general theorem due to Robertson and Seymour [17] stating that all classes of graphs that are closed under taking minors are recognizable in cubic time. Recall that a minor of a graph G is a graph obtained from a subgraph of G by contracting edges. It is easy to see that the class of all graphs of genus at most k is closed under taking minors.

Unfortunately the class of all graphs of crossing number at most k is not closed under taking minors. So in general Robertson and Seymour's theorem cannot be applied to compute crossing numbers. An exception is the case of graphs of degree at most 3; Fellows and Langston [9] observed that for such graphs Robertson and Seymour's result immediately yields a cubic time algorithm for computing crossing numbers.[2] Although we cannot apply Robertson and Seymour's result directly, the overall strategy of our algorithm is inspired by their ideas: The algorithm first iteratively reduces the size of the input graph until it reaches a graph of bounded tree-width, and then solves the problem on this graph. For the reduction

*Laboratory for Foundations of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, Scotland, UK.
Email: grohe@inf.ed.ac.uk

[1] This can be implemented by exhaustive search of the space of m^{2k} k-tuples of edge pairs, where m denotes the number of edges of the input graph.

[2] This is simply because for graphs of degree at most 3 the minor relation and the topological subgraph relation coincide.

step, we use Robertson and Seymour's Excluded Grid Theorem [16] together with a nice lemma due to Thomassen [19] stating that in a graph of bounded genus (and thus in a graph of bounded crossing number) every large grid contains a subgrid that, in some precise sense, lies "flat" in the graph. Such a flat grid does not essentially contribute to the crossing number and can therefore be contracted. For the remaining problem on graphs of bounded tree-width we apply a theorem due to Courcelle [4] stating that all properties of graphs that are expressible in monadic second-order logic are decidable in linear time on graphs of bounded tree-width.

Let me remark that the hidden constant in the quadratic upper bound for the running time of our algorithm heavily depends on k. As a matter of fact, the running time is $O(f(k) \cdot n^2)$, where f is at least doubly exponential. Thus our algorithm is only of theoretical interest.

2. Preliminaries

Graphs in this paper are undirected and loop-free, but they may have multiple edges.[3] The vertex set of a graph G is denoted by V^G, the edge set by E^G. We always assume that $V^G \cap E^G = \emptyset$. For graphs G and H we let $G \cup H := (V^G \cup V^H, E^G \cup E^H)$ and $G \setminus H := \big(V^G \setminus V^H, \{e \in E^G \setminus E^H \mid \text{both endpoints of } e \text{ are contained in } V^G \setminus V^H\}\big)$. A *subgraph* of a graph G is a graph H with $V^H \subseteq V^G$ and $E^H \subseteq E^G$. We formally treat paths and cycles in a graph as subgraphs of this graph (as opposed to, say, sequences of vertices). Paths and cycles are always *simple*, that is, they have no self-intersections.

2.1. Topological Embeddings. A *topological embedding* of a graph G into a graph H is a mapping h that associates a vertex $h(v) \in V^H$ with every $v \in V^G$ and a path $h(e)$ in H with every $e \in E^G$ in such a way that:

- For distinct vertices $v, w \in V^G$, the vertices $h(v)$ and $h(w)$ are distinct.
- For distinct edges $e, f \in E^G$, the paths $h(e)$ and $h(f)$ are internally disjoint (that is, they have at most their endpoints in common).
- For every edge $e \in E^G$ with endpoints v and w, the two endpoints of the path $h(e)$ are $h(v)$ and $h(w)$, and $h(u) \notin V^{h(e)}$ for all $u \in V^G \setminus \{v, w\}$.

We let $h(G)$ be the subgraph of H consisting of the images of the vertices and edges of G under h. Formally, $h(G) := \big(h(V^G), \emptyset\big) \cup \bigcup_{e \in E^G} h(e)$.

2.2. Drawings and Crossing Numbers. A *drawing* of a graph G is a mapping Δ that associates with every vertex $v \in V^G$ a point $\Delta(v) \in \mathbb{R}^2$ and with every edge $e \in E^G$ a simple curve $\Delta(e)$ in $\mathbb{R}^2$ in such a way that:

- For distinct vertices $v, w \in V^G$, the points $\Delta(v)$ and $\Delta(w)$ are distinct.
- For distinct edges $e, f \in E^G$, the curves $\Delta(e)$ and $\Delta(f)$ have at most one interior point in common (and possibly their endpoints).
- For every edge $e \in E^G$ with endpoints v and w, the two endpoints of the curve $\Delta(e)$ are $\Delta(v)$ and $\Delta(w)$, and $\Delta(u) \notin \Delta(e)$ for all $u \in V^G \setminus \{v, w\}$.
- At most two edges intersect in one point. Formally, $|\{e \in E^G \mid x \in \Delta(e)\}| \leq 2$ for all $x \in \mathbb{R}^2 \setminus \Delta(V^G)$.

We let $\Delta(G) := \Delta(V^G) \cup \bigcup_{e \in E^G} \Delta(e)$.

An $x \in \mathbb{R}^2 \setminus \Delta(V^G)$ with $|\{e \in E^G \mid x \in \Delta(e)\}| = 2$ is called a *crossing* of Δ. The *crossing number* of Δ is the number of crossings of Δ. A drawing of crossing number 0 is called *plane*. The *crossing number* of a graph G is the minimum taken over the crossing numbers of all drawings of G. A graph of crossing number 0 is called *planar*.

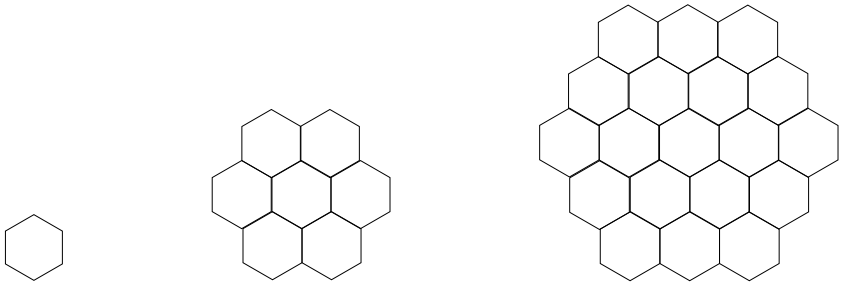

Figure 1. The hexagonal grids H_1, H_2, H_3

2.3. Hexagonal Grids. For $r \geq 1$, we let H_r be the hexagonal grid of radius r. Instead of giving a formal definition, we refer the reader to Figure 1 to see what this means. The *principal cycles* $C_1, \ldots, C_r$ of H_r are the the concentric cycles, numbered from the interior to the exterior (see Figure 2).

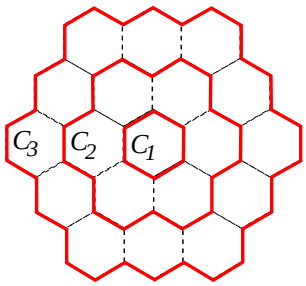

Figure 2. The principal cycles of H_3

2.4. Flat Grids in a Graph. For graphs $H \subseteq G$, an *H-component (of G)* is either a connected component C of $G \setminus H$ together with all edges connecting C with H and their endpoints in H or an edge in $E^G \setminus E^H$ whose endpoints are both in H together with its endpoints. The vertices in the intersection of an H-component with H are called the *vertices of attachment* of the component.

Let G be a graph and $h : H_r \to G$ a topological embedding. The *interior of* $h(H_r)$ is the subgraph $h(H_r \setminus C_r)$ (remember that C_r is the outermost principal cycle of H_r). A *proper* $h(H_r)$-component is an $h(H_r)$-component that has at least one vertex of attachment in the interior of $h(H_r)$. The topological embedding h is *flat* if the union of $h(H_r)$ with all its proper components is a planar graph.

We shall use the following theorem due to Thomassen [19]. Actually, Thomassen stated the result for the *genus* of a graph rather than its crossing number. However, it is easy to see that the crossing number of a graph is an upper bound for its genus.

Theorem 1 (Thomassen [19]). *For all $k, r \geq 1$ there is an $s \geq 1$ such that the following holds: If G is a graph of crossing number at most k and $h : H_s \to G$ a topological embedding, then there is a subgrid $H_r \subseteq H_s$ such that the restriction $h|_{H_r}$ of h to H_r is flat.*

2.5. Tree-Width. We assume that reader is familiar with the notion *tree-width (of a graph)*. It is no big problem if not; we never really work with tree-width, but just take it as a black box in Theorems 2–4. Robertson and Seymour's deep *Excluded Grid Theorem* [16] states that every graph of sufficiently large tree-width contains the homeomorphic image of a large grid. We use the following algorithmic version of this theorem.

[3]Note that loops are completely irrelevant for the crossing number, whereas multiple edges are not.

Theorem 2. (Robertson and Seymour [17], Bodlaender [1], Perković and Reed [15]). *Let $r \geq 1$. Then there is a $w \geq 1$ and a linear time algorithm that, given a graph G, either (correctly) recognizes that the tree-width of G is at most w or computes a topological embedding $h : H_r \to G$.*

Robertson and Seymour [17] gave a quadratic time algorithm, but they pointed out that it can be improved to linear time using Bodlaender's [1] linear time algorithm for computing tree-decompositions. However, this improvement is not entirely straightforward: Let us fix a constant $w \geq 1$. The essential part of Robertson and Seymour's algorithm for the problem stated in the theorem is a quadratic time algorithm that, given a graph G, returns a tree-decomposition of G of width at most $4w$ if the tree-width of the input graph is at most w. Furthermore, the algorithm returns a "counterexample" subgraph H of G of tree-width larger than w and at most $2w$ if the tree-width of G is greater than w. This counterexample is very important here because the subgraph H is then used to find the topological embedding of H_r into G.

Bodlaender [1] gave a linear time algorithm computing a tree-decomposition of width at most w if the tree-width of the input graph G is at most w, but his algorithm does not return a counterexample if the tree-width of G is greater than w. Perković and Reed [15] extended Bodlaender's algorithm in such a way that it still works in linear time, but does return a counterexample if the tree-width of the input graph is greater than w.

2.6. Courcelle's Theorem. Courcelle's theorem states that properties of graphs definable in *Monadic Second-Order Logic MSO* can be checked in linear time on input graphs of bounded tree-width. In this logical context we consider graphs as relational structures of vocabulary $\{E, V, I\}$, where V and E are unary relation symbols interpreted by the vertex set and edge set of a graph and I is a binary relation symbol interpreted by the incidence relation. To simplify the notation, for a graph G we let $U^G := V^G \cup E^G$ and call U^G the *universe* of G.

I assume that the reader is familiar with the definition of MSO. However, for those who are not I have included it in Appendix A.

Theorem 3 (Courcelle [4]). *Let $w \geq 1$ and let*

$$\varphi(x_1, \ldots, x_k, X_1, \ldots, X_l)$$

be an MSO-formula. Then there is a linear time algorithm that, given a graph G of tree-width at most w and $a_1, \ldots, a_k \in U^G$, $A_1, \ldots, A_l \subseteq U^G$, decides whether $G \models \varphi(a_1, \ldots, a_k, A_1, \ldots, A_l)$.

We shall also use the following strengthening of Courcelle's theorem, a proof of which can be found in [11]:

Theorem 4. *Let $w \geq 1$ and let*

$$\varphi(x_1, \ldots, x_k, X_1, \ldots, X_l, y_1, \ldots, y_m, Y_1, \ldots, Y_n)$$

be an MSO-formula. Then there is a linear time algorithm that, given a graph G of tree-width at most w and $b_1, \ldots, b_m \in U^G$, $B_1, \ldots, B_n \subseteq U^G$, decides if there exist $a_1, \ldots, a_k \in U^G$, $A_1, \ldots, A_l \subseteq U^G$ such that

$$G \models \varphi(a_1, \ldots, a_k, A_1, \ldots, A_l, b_1, \ldots, b_m, B_1, \ldots, B_n),$$

and, if this is the case, computes such elements $a_1, \ldots, a_k$ and sets $A_1, \ldots, A_l$.

3. The Algorithm

For an $l \geq 1$, a graph G, and a subset $F \subseteq E^G$ of *forbidden edges*, an *l-good drawing of G with respect to F* is a drawing Δ of G of crossing number at most l such that no forbidden edges are involved in any crossings, that is, for every crossing $x \in \Delta(e) \cap \Delta(f)$ of Δ we have $e, f \in E^G \setminus F$.

We fix a $k \geq 1$ for the whole section. We shall describe an algorithm that solves the following *generalized k-crossing number problem* in quadratic time:

Input:	Graph G and subset $F \subseteq E^G$.
Problem:	Decide if G has a k-good drawing with respect to F.

Later, we shall extend our algorithm in such a way that it actually computes a k-good drawing if there exists one.

Our algorithm works in two phases. In the first, it iteratively reduces the size of the input graph until it obtains a graph whose tree-width is bounded by a constant only depending on k. Then, in the second phase, it solves the problem on this graph of bounded tree-width.

3.1. Phase I. We let $r := 4k + 3$ and choose s sufficiently large such that for every graph G of crossing number at most k and every topological embedding $h : H_s \to G$ there is a subgrid $H_r \subseteq H_s$ such that the restriction $h|_{H_r}$ of h to H_r is flat. Such an s exists by Theorem 1. Then we choose w with respect to s according to Theorem 2 such that we have a linear time algorithm that, given a graph of tree-width greater than w, finds a topological embedding $h : H_s \to G$. We keep r, s, w fixed for the rest of the section.

Lemma 5. *There is a linear time algorithm that, given a graph G, either recognizes that the crossing number of G is greater than k, or recognizes that the tree-width of G is at most w, or computes a flat topological embedding $h : H_r \to G$.*

Proof: We first apply the algorithm of Theorem 2. If it recognizes that the tree-width of the input graph G is at most w, we are done. Otherwise, it computes a topological embedding $h : H_s \to G$. By our choice of s, we know that either the crossing number of G is greater than k or there is a subgrid $H_r \subseteq H_s$ such that the restriction of h to H_r is flat.

For each $H_r \subseteq H_s$ we can decide whether $h|_{H_r}$ is flat by a planarity test, which is possible in linear time [13]. Our algorithm tests whether $h|_{H_r}$ is flat for all $H_r \subseteq H_s$. Either it finds a flat $h|_{H_r}$, or the crossing number of G is greater than k.[4]

Since s is a fixed constant, the overall running time is linear. □

Let G be a graph and $h : H_r \to G$ a topological embedding. For $1 \leq i \leq r-1$, we let H^i be the subgrid of H_r bounded by the ith principal cycle C_i. We let K_i be the subgraph of G consisting of $h(H^i)$ and all $h(H_r)$-components all of whose vertices of attachment are in $h(H^i)$. Moreover, we let B_i be the subgraph of G consisting of $h(C_i)$ and all $h(H_r)$-components all of whose vertices of attachment are in $h(C_i)$. In particular, we call the subgraph $K_2 \subseteq G$ the *kernel* of h, B_2 the *boundary of the kernel*, and $K_2 \setminus B_2$ the *interior of the kernel* (see Figure 3).

Lemma 6. *Let G be a graph, $F \subseteq E^G$, and let Δ be a k-good drawing of G with respect to F of minimum crossing number. Let $h : H_r \to G$ be a flat topological embedding. Then none of the edges of the kernel of h is involved in any crossing of Δ.*

To understand the significance of this lemma, note that the flatness of the topological embedding h guarantees that the graph $h(K_i)$ is planar for all $i \leq r-1$. However, this does not necessarily mean that the restriction of the specific drawing Δ to K_i is plane. The lemma implies that at least the restriction of Δ to the kernel K_2 is plane (the actual statement of the lemma is slightly stronger).

Proof: For $1 \leq i \leq r-1$, the ith *ring* of h is the subgraph R_i of G consisting of $h(C_i)$ and $h(C_{i+1})$ (the images of the ith and $(i+1)$th principal cycle) and the images of all edges in H_r connecting these two cycles (see Figure 4). Then for i, j with $1 \leq i < j-1 \leq r-2$, the graphs $R_i \cup B_i$ and $R_j \cup B_j$ are disjoint. Recall that $r = 4k+3$. Since at most two edges are involved in any crossing, by the pigeonhole-principle there is an $i_0, 2 \leq i_0 \leq r-1$ such that none of the edges in $B_{i_0} \cup R_{i_0}$ is involved in any crossing of Δ. Let $R := R_{i_0}$, $B := B_{i_0}$, $C := h(C_{i_0})$, $K := K_{i_0}$ and $I := K \setminus B$. Then K and I are both connected planar graphs.

Let $\tilde{I}$ be the $G \setminus I$-component that contains I. Thus $\tilde{I}$ consists of I, all edges connecting I to $G \setminus I$, and the endpoints of these edges. (Recall the definition of an H-component of a graph G from the beginning of Subsection 2.4.) Note that the vertices of attachment of $\tilde{I}$ are all on C.

[4] The proof of Thomassens's theorem reveals that actually we do not have to test all $H_r \subseteq H_s$ for flatness, but only a number that is linear in k.

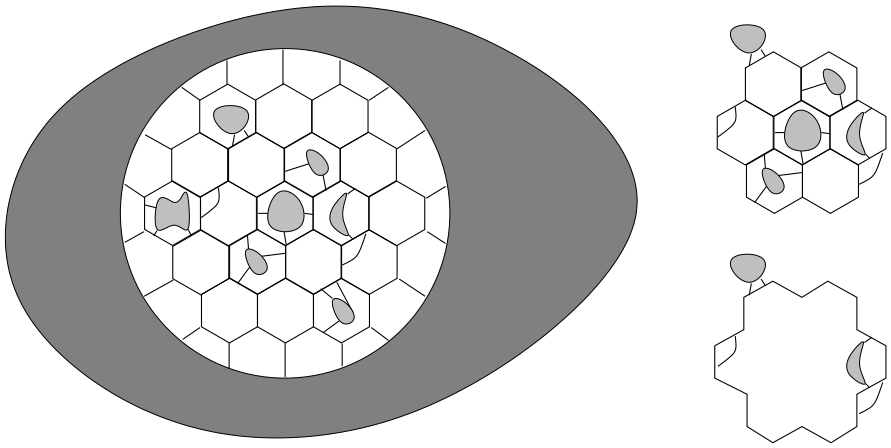

Figure 3. A flat grid in a graph, its kernel, and the boundary of the kernel

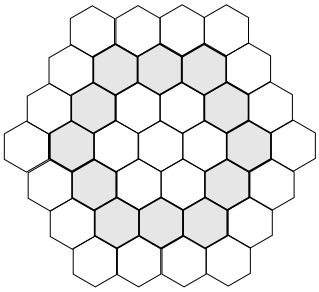

Figure 4. The ring R_2 in a grid

Without loss of generality we can assume that G is connected. Then the graph $G \setminus I$ consists of the boundary B and one connected component *EXT* that contains the exterior part of the grid (in particular the cycle C_r — recall that $i_0 < r$) and the rest of G. B consists of the cycle C and possibly additional C-components $D_1, \ldots, D_\ell$. Let us first consider the restriction of Δ to $G \setminus I$.

Claim 1: There exists a connected component Z of $\mathbb{R}^2 \setminus \Delta(G \setminus I)$ such that the Δ-images of all vertices of attachment of $\tilde{I}$ are on the boundary of Z.

Proof: Since C is a cycle and none of the edges of C is involved in any crossing of Δ, $\Delta(C)$ is a simple closed curve in the plane $\mathbb{R}^2$. Furthermore, since *EXT* is a connected graph, $\Delta(EXT)$ must be entirely contained in one connected component of $\mathbb{R}^2 \setminus \Delta(C)$, say, in the exterior.

Let $Y \subseteq \mathbb{R}^2$ be the interior of $\Delta(C)$. Then $\Delta(\tilde{I})$ must be contained in Y (except for its vertices of attachment, which are on the boundary of Y). To see this, suppose for contradiction that $\Delta(\tilde{I})$ is not contained in Y. Since $\tilde{I}$ is a connected graph and none of the edges of the ring R is involved in any crossing of Δ, either $\Delta(\tilde{I})$ is contained in the Δ-image one of the hexagons of the ring R, or it is contained in the exterior of the ring. But this can both not happen, because the vertices of attachment of $\tilde{I}$ are on $h(C)$, and they are not contained in the boundary of a single hexagon. So $\Delta(\tilde{I})$ is contained in Y.

Now if $B = C$ (that is, if there are no components D_i), or if Δ maps all components D_i to the exterior of $\Delta(C)$, then Y is a connected component of $\mathbb{R}^2 \setminus \Delta(G \setminus I)$ such that the Δ-images of all vertices of attachment of $\tilde{I}$ are on the boundary of Y. But in general, Δ may map some of the components D_i to Y. Suppose for contradiction that there are two connected component Z, Z' of $Y \setminus \Delta(B)$ such that the boundary of both contains the image of a vertex of attachment of $\tilde{I}$. Since I is connected, there is a path $P \subseteq \tilde{I}$ connecting these two vertices, and $\Delta(P)$ must intersect $\Delta(B)$. This contradicts the fact that none

of the edges of B is involved in any crossing of Δ.

Claim 2: The restriction of Δ to K is plane.

Proof: Suppose for contradiction that this is not the case. Then two edges of $\tilde{I}$ must cross under Δ.

Let Π be a plane drawing of the planar graph K. Let Z' be the the connected component of $\Pi(K \setminus I)$ that contains I. Without loss of generality we can assume that Z' is not the exterior component, that is, Z' is homeomorphic to an open disc. Similarly we can assume that the set Z of Claim 1 is homeomorphic to an open disc. Let $v_1, \ldots, v_m$ be the vertices of attachment of $\tilde{I}$, and let f be a homeomorphism from $Z' \cup \{\Pi(v_1), \ldots, \Pi(v_m)\}$ to $Z \cup \{\Delta(v_1), \ldots, \Delta(v_m)\}$. We define a new drawing Δ' of G by letting $\Delta' = \Delta$ on $G \setminus I$ and $\Delta' = f \circ \Pi$ on $\tilde{I}$.

Then no edges of $\tilde{I}$ are involved in any crossing of Δ', thus the crossing number of Δ' is smaller than that of Δ. This contradicts the minimality of the crossing number of Δ' and proves Claim 2.

Claim 3: None of the edges of K is involved in any crossing of Δ.

Proof: Since the restriction of Δ to K is plane, and since none of the edges of $B = K \setminus I$ is involved in any crossing, the only possible crossing involving an edge of K would be between an edge of *EXT* and an edge of $\tilde{I}$. But since $\tilde{I}$ and *EXT* are embedded into different components of $\mathbb{R}^2 \setminus h(C)$ (we showed this in the proof of Claim 1), each such crossing would induce a crossing with an edge of $C \subseteq B$. So there cannot be any such crossings, and Claim 3 is proved.

To complete the proof of the lemma, we just recall that the kernel K_2 is a subgraph of $K = K_{i_0}$. □

Now we are ready to describe the main reduction step our algorithm performs.

Lemma 7. *There is a linear time algorithm that, given a graph G and an edge set $F \subseteq E^G$, either recognizes that the crossing number of G is greater than k, or recognizes that the tree-width of G is at most w, or computes a graph G' and an edge set $F' \subseteq E^{G'}$ such that $|V^{G'}| < |V^G|$, and G has a k-good drawing with respect to F if, and only if, G' has a k-good drawing with respect to F'.*

Proof: We first apply the algorithm of Lemma 5. If it tells us that the crossing number of G is greater than k or that the tree-width of G is at most w, there is nothing else we need to do. So suppose the algorithm returns a flat topological embedding $h : H_r \to G$.

Let K be the kernel of h, I its interior, and B its boundary. Let G' the graph obtained from G by contracting the connected subgraph I to a single vertex v_I (see Figure 5).[5]

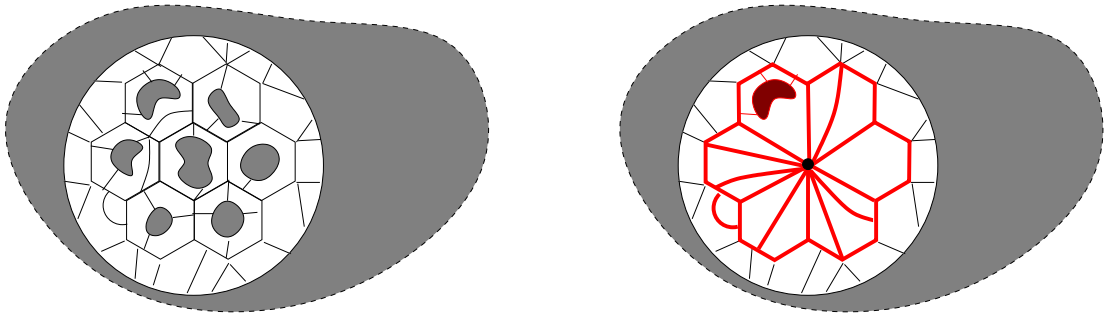

Figure 5. The transformation from a graph G to G'

Let F' be the union of F with the set of all edges of B and all edges incident with the new vertex v_I. I claim that G has a k-good drawing with respect to F if, and only if, G' has a k-good drawing with respect to F'. The forward direction of this claim follows from Lemma 6. For the backward direction we observe

[5] In other words, G' is obtained from G by deleting all vertices of I, deleting all edges with both endpoints in I, adding a new vertex v_I, and replacing, for all edges with one endpoint in I, this endpoint by v_I.

that every k-good drawing Δ' of G' with respect to F' can be turned into a k-good drawing of G with respect to F by embedding the planar graph I into a small neighborhood of $\Delta'(v_I)$.

Clearly, given G, F and h, the graph G' and the edge-set F' can be computed in linear time. Moreover $|V^{G'}| < |V^G|$. This yields the desired algorithm □

Iterating the algorithm of the lemma, we obtain:

Corollary 8. *There is a quadratic time algorithm that, given a graph G, either recognizes that the crossing number of G is greater than k or computes a graph G' and an edge set $F' \subseteq E^{G'}$ such that the tree-width of G' is at most w and G has a k-good drawing with respect to F if, and only if, G' has a k-good drawing with respect to F'.*

3.2. Phase II. If the algorithm has not found out that the graph has crossing number greater than k in Phase I, it has produced a graph G' of tree-width at most w and a set $F' \subseteq E^{G'}$ such that G has a k-good drawing with respect to F if, and only if, G' has a k-good drawing with respect to F'. In Phase II, the algorithm has to decide whether G' has a k-good drawing with respect to F'. Using Courcelle's Theorem 3, we prove that this can be done in linear time.

To this end, we shall find an MSO-formula $\varphi(X)$ such that for every graph G and every set $F \subseteq E^G$ we have $G \models \varphi(F)$ if, and only if, G has a k-good drawing with respect to F. We rely on the well-known fact that there is an MSO-formula φ_{planar} saying that a graph is planar. (Actually, this is quite easy to see: φ_{planar} just says that G neither contains K_5 nor $K_{3,3}$ as a topological subgraph. Also see [6].)

For a graph G and edges $e_1, e_2 \in E^G$ that do not have an endpoint in common we let $G^{e_1 \times e_2}$ be the graph obtained from G by deleting the edges e_1 and e_2 and adding a new vertex x and four edges connecting x with the endpoints of the edges of e_1 and e_2 in G (see Figure 6). Observe that for every $l \geq 1$

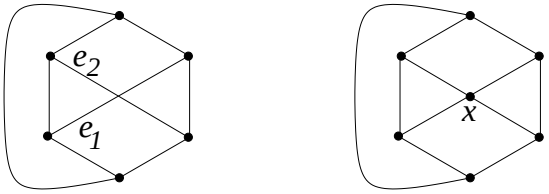

Figure 6. A graph G with selected edges e_1, e_2 and the resulting $G^{e_1 \times e_2}$

a graph G has an l-good drawing with respect to an edge set $F \subseteq E^G$ if, and only if, either G has an $(l-1)$-good drawing with respect to F or there are edges $e_1, e_2 \in E^G \setminus F$ that do not have an endpoint in common such that $G^{e_1 \times e_2}$ has an $(l-1)$-good drawing with respect to F.

Lemma 9. *For every MSO-formula $\varphi(Y)$ there exists an MSO-formula $\varphi^*(y_1, y_2, Y)$ such that for all graphs G, edge sets $F \subseteq E^G$ and edges $e_1, e_2 \in E^G \setminus F$ that do not have an endpoint in common we have:*

$$G \models \varphi^*(e_1, e_2, F) \iff G^{e_1 \times e_2} \models \varphi(F).$$

This lemma can easily be proved by a standard technique from logic, the method of *syntactic interpretations*.[6] For readers not familiar with syntactic interpretations, a direct proof of the lemma can be found in Appendix B.

Using this lemma, we inductively define, for every $l \geq 1$, formulas $\varphi_l(Y)$ and $\psi_l(y_1, y_2, Y)$ such that for every graph G and edge set $F \subseteq E^G$ we have

$$G \models \varphi_l(F) \iff \begin{array}{l} G \text{ has an } l\text{-good drawing} \\ \text{with respect to } F, \end{array}$$

[6] For an introduction to the technique I refer the reader to [8], for the particular situation of MSO on graphs to [3, 5].

and for all G, $F \subseteq E^G$, and edges $e_1, e_2 \in E^G \setminus F$ that do not have an endpoint in common we have

$$G \models \psi_l(e_1, e_2, F) \iff G^{e_1 \times e_2} \text{ has an } (l-1)\text{-good drawing with respect to } F.$$

We let

$$\psi_1(y_1, y_2, Y) := \varphi^*_{\text{planar}}(y_1, y_2)$$

and, for $l \geq 1$,

$$\begin{aligned}\varphi_l(Y) :=& \varphi_{l-1}(Y) \vee \exists y_1 \exists y_2 \exists x_{11} \exists x_{12} \exists x_{21} \exists x_{22} \big(Ey_1 \wedge Ey_2 \wedge \neg Y y_1 \wedge \neg Y y_2 \\ & \wedge \bigwedge_{\substack{1 \leq i,i',j,j' \leq 2 \\ (i,j) \neq (i',j')}} x_{ij} \neq x_{i'j'} \wedge Ix_{11}y_1 \wedge Ix_{12}y_1 \wedge Ix_{21}y_2 \wedge Ix_{22}y_2 \\ & \wedge \psi_l(y_1, y_2, Y)\big), \\ \psi_{l+1}(y_1, y_2, Y) :=& \varphi^*_l(y_1, y_2, Y).\end{aligned}$$

This completes our proof that there is a quadratic time algorithm deciding if a graph G has a good drawing with respect to a set $F \subseteq E^G$.

3.3. Computing a Good Drawing. So far we have only proved that there is a quadratic time algorithm deciding if a graph has a good drawing. It is not hard to modify this algorithm so that it actually computes a drawing: For Phase I, we observe that if we have a good drawing of G' with respect to F' then we can easily construct a good drawing of G with respect to F. So we only have to worry about Phase II.

By induction on l, for every $l \geq 0$ we define a linear-time procedure DRAW$_l$ that, given a graph G of tree-width at most w and a subset $F \subseteq E^G$, computes an l-good drawing of G with respect to F (if there exists one). DRAW$_0$ just has to compute a plane drawing of G.

For $l \geq 1$, we apply Theorem 4 to the MSO-formula

$$\begin{aligned}\chi_l(y_1, y_2, Y) :=& \quad Ey_1 \wedge Ey_2 \wedge \neg Y y_1 \wedge \neg Y y_2 \\ & \wedge \exists x_{11} \exists x_{12} \exists x_{21} \exists x_{22} \big(\bigwedge_{\substack{1 \leq i,i',j,j' \leq 2 \\ (i,j) \neq (i',j')}} x_{ij} \neq x_{i'j'} \wedge Ix_{11}y_1 \wedge Ix_{12}y_1 \wedge Ix_{21}y_2 \wedge Ix_{22}y_2 \big) \\ & \wedge \psi_l(y_1, y_2, Y).\end{aligned}$$

It yields a linear time algorithm that, given a graph G and an $F \subseteq E^G$, computes two edges $e_1, e_2 \in E^G \setminus F$ such that $G \models \chi_l(e_1, e_2, F)$ (if such edges exist). It follows immediately from the definition of ψ_l that $G \models \chi_l(e_1, e_2, F)$ if, and only if, e_1, e_2 are in $E^G \setminus F$ that do not have an endpoint in common and $G^{e_1 \times e_2}$ has an l-good drawing with respect to F.

Given G and F, the procedure DRAW$_l$ applies this linear-time algorithm to compute edges e_1, e_2 such that

$$G \models \chi_l(e_1, e_2, F).$$

Then it applies DRAW$_{l-1}$ to the graph $G^{e_1 \times e_2}$ to compute an $(l-1)$-good drawing of a graph $G^{e_1 \times e_2}$ with respect to F. It modifies this drawing in a straightforward way to obtain an l-good drawing of G with respect to F.

Remark 10. In the conference version of this paper I claimed that the use of Courcelle's theorem in our proof can be avoided in favor of a direct algorithm that employs "the usual dynamic programming techniques". Although this is true — after all, Courcelle's algorithm also employs these dynamic programming techniques, so we can simply take the formula constructed above and extract an algorithm from Courcelle's proof — it is not as simple as I thought then.

The formula we construct essentially says: "There exists edges $e_1, \ldots, e_{2k}$ such that if we cross e_i with e_{i+1} (for $1 \leq i \leq k$) the graph does not contain a K_5-minor or a $K_{3,3}$-minor." This statement involves a non-trivial quantifier alternation, which is what makes the translation to an algorithm difficult.

At this point, I think that the route through logic and Courcelle's theorem is essential for our proof.

3.4. Uniformity. Inspection of our proofs and the proofs of the results we use shows that actually there is *one* algorithm that, given a graph G with n vertices and a non-negative integer k, decides whether the crossing number of G is at most k in time $O(f(k) \cdot n^2)$ for a suitable function f.

Unfortunately, f grows extremely fast, at least doubly exponentially. To see this, note that the tree-width $w = w(k)$ we derive a from the excluded grid theorem is exponential in k. Testing whether a graph has tree-width w requires time exponential in w, that is, doubly exponential in k.

To give an upper bound on the growth of f, we mainly have to analyse the running time of the algorithm we get out of Courcelle's theorem. Its dependence on the formula φ is q-fold exponential in the tree-width and the formula length, where q is the number of quantifier alternations of the formula. The straightforward bound on the number of quantifier alternations of our formula saying that the crossing number is at most k is 3 (independent of k).

There may be slight improvements reducing the running time by one or two exponentials, but as we saw our approach will not give us a running time whose dependence on k is less than doubly exponential.

4. Conclusions

We have proved that for every $k \geq 0$ there is a quadratic time algorithm deciding whether a given graph has crossing number at most k. The running time of our algorithm in terms of k is enormous, which makes the algorithm useless for practical purposes. This is partly due to the fact that the algorithm heavily relies on graph minor theory.

However, knowing the crossing number problem to be fixed-parameter tractable may help to find better algorithms that are practically applicable for small values of k. This has happened in a similar situation for the vertex cover problem. The first proof [9] showing the fixed-parameter tractability of vertex cover used Robertson and Seymour's theorem that classes of graphs closed under taking minors are recognizable in cubic time. Starting from there, much better algorithms have been developed; by now, vertex cover can be (practically) solved for a quite reasonable problem size (see [2] for a state-of-the-art algorithm).

Although I do not expect there to be such a simple algorithm for deciding whether the crossing number of a graph is at most k as there is for deciding whether there is a vertex cover of size at most k, I conjecture that there is a more elementary algorithm for the crossing number problem whose running time is $2^{O(k)} \cdot n$.

Appendix A: Monadic Second Order Logic

We first explain the syntax of MSO: We have an infinite supply of *individual variables*, denoted by x, y, z, x_1 et cetera, and also an infinite supply of *set variables*, denoted by X, Y, et cetera. *Atomic MSO-formulas (over graphs)* are formulas of the form $x = y$, Vx, Ex, Ixy, and Xx, where x, y are individual variables and X is a set variable. The class of MSO-formulas is defined by the following rules:

- Atomic MSO-formulas are MSO-formulas.
- If φ is an MSO-formula, then so is $\neg\varphi$.
- If φ and ψ are MSO-formulas, then so are $\varphi \wedge \psi$, $\varphi \vee \psi$, and $\varphi \to \psi$.
- If φ is an MSO-formula and v is a variable (either an individual variable or a set variable), then $\exists v\varphi$ and $\forall v\varphi$ are MSO-formulas.

Let G be a graph. Recall that $U^G = V^G \cup E^G$. A *G-assignment* is a mapping α that associates an element of U^G with every individual variable and a subset of U^G with every set variable. We inductively define what it means that a graph G together with a G-assignment α *satisfies* an MSO-formula φ (we write $(G, \alpha) \models \varphi$):

- $(G, \alpha) \models x = y \iff \alpha(x) = \alpha(y)$,
  $(G, \alpha) \models Vx \iff \alpha(x) \in V^G$,
  $(G, \alpha) \models Ex \iff \alpha(x) \in E^G$,
  $(G, \alpha) \models Ixy \iff \big(\alpha(x) \in V^G, \alpha(y) \in E^G, \text{ and } \alpha(x) \text{ endpoint of } \alpha(y)\big)$,
  $(G, \alpha) \models Xx \iff \alpha(x) \in \alpha(X)$,

– $(G,\alpha) \models \neg\varphi \iff (G,\alpha) \not\models \varphi$,

– $(G,\alpha) \models \varphi \wedge \psi \iff \big((G,\alpha) \models \varphi \text{ and } (G,\alpha) \models \psi\big)$,
and similarly for $\vee$, meaning "or", and $\rightarrow$, meaning "implies".

– $(G,\alpha) \models \exists x\varphi \iff$ there exists an $a \in U^G$
such that $(G, \alpha\frac{a}{x}) \models \varphi$,
where $\alpha\frac{a}{x}$ denotes the assignment with $\alpha\frac{a}{x}(x) = a$ and $\alpha\frac{a}{x}(v) = \alpha(v)$ for all $v \neq x$.

– $(G,\alpha) \models \forall x\varphi \iff$ for all $a \in U^G$, $(G, \alpha\frac{a}{x}) \models \varphi$.

– $(G,\alpha) \models \exists X\varphi \iff$ there exists an $A \subseteq U^G$
such that $(G, \alpha\frac{A}{X}) \models \varphi$,
and similarly for $\forall X$ meaning "for all $A \subseteq U^G$".

It is easy to see that the relation $(G,\alpha) \models \varphi$ only depends on the values of α at the *free variables* of φ, that is, those variables v not occurring in the scope of a quantifier $\exists v$ or $\forall v$. We write

$$\varphi(x_1, \ldots, x_k, X_1, \ldots, X_l)$$

to indicate that the free individual variables of φ are among $x_1, \ldots, x_k$ and the free set variables are among $X_1, \ldots, X_l$. Then for a graph G and $a_1, \ldots, a_k \in U^G$, $A_1, \ldots, A_l \subseteq U^G$ we write

$$G \models \varphi(a_1, \ldots, a_k, A_1, \ldots, A_l)$$

if for every assignment α with $\alpha(x_i) = a_i$ and $\alpha(X_j) = A_j$ we have $(G,\alpha) \models \varphi$. A *sentence* is a formula without free variables.

For example, for the sentence

$$\begin{aligned}\varphi := \exists X \exists Y \Big(& \forall x \big(Vx \rightarrow (Xx \vee Yx)\big) \\ & \wedge \forall x \forall y \Big(\big(x \neq y \wedge \exists z (Ixz \wedge Iyz)\big) \rightarrow \neg\big((Xx \wedge Xy) \vee (Yx \wedge Yy)\big)\Big)\Big)\end{aligned}$$

we have $G \models \varphi$ if, and only if, G is 2-colorable.

Appendix B: Proof of Lemma 9

For the reader's convenience, we repeat the statement of the Lemma:

> *For every MSO-formula $\varphi(Y)$ there exists an MSO-formula $\varphi^*(y_1, y_2, Y)$ such that for all graphs G, edge sets $F \subseteq E^G$ and edges $e_1, e_2 \in E^G \setminus F$ that do not have an endpoint in common we have:*
>
> $$G \models \varphi^*(e_1, e_2, F) \iff G^{e_1 \times e_2} \models \varphi(F). \tag{1}$$

Proof: Let G be a graph, $F \subseteq E^G$, and let $e_1, e_2 \in E^G \setminus F$ be edges that do not have an endpoint in common. Let v_{11}, v_{12} be the endpoints of e_1 and v_{21}, v_{22} the endpoints of e_2. We define a graph G^* as follows:

– The vertex set of G^* is the set

$$V^{G^*} := \{(v,v) \mid v \in V^G\} \cup \{(e_1, e_2)\}.$$

– The edge set of G^* is the set

$$E^{G^*} := \{(e,e) \mid e \in E^G\} \cup \{(e_1, v_{11}), (e_1, v_{12}), (e_1, v_{21}), (e_1, v_{22})\}.$$

– A vertex $(u,w) \in V^{G^*}$ is an endpoint of an edge $(x,y) \in E^{G^*}$ if

– $(u,w) = (v,v)$ for a vertex $v \in V^G$ and $(x,y) = (e,e)$ for an edge $e \in E^G$, and v is an endpoint of e, or
– $(u,v) = (e_1,e_2)$ and $(x,y) = (e_1, v_{ij})$ for some $i,j, 1 \leq i,j \leq 2$, or
– $(u,v) = (v_{ij}, v_{ij})$ and $(x,y) = (e_1, v_{ij})$ for some $i,j, 1 \leq i,j \leq 2$.

Then it is easy to see that G^* is isomorphic to the graph $G^{e_1 \times e_2}$. (Note that in order for this to be true we need our general assumption that $V^G \cap E^G = \emptyset$.) The point of getting an isomorphic copy of $G^{e_1 \times e_2}$ this way is that G^* is *definable within* G.

Now let $\varphi(Y)$ be an MSO-formula. We want to translate it to a formula $\varphi^*(y_1, y_2, Y)$ that G satisfies $\varphi^*(e_1, e_2, F)$ if, and only if, G^* satisfies $\varphi(F)$.

Without loss of generality, we can assume that the variables $y_1, y_2, x_{11}, x_{12}, x_{21}, x_{22}$ do not appear in φ and that the set variable Y only appears as a free variable (that is, there are no subformulas $\exists Y \psi$ or $\forall Y \psi$). Since elements of the universe of G^* are pairs of elements of the universe of G, for every variable v of φ we will have two variables v', v'' in φ^*. For an element $a = (b,c)$ of the universe of G^* we let $a' = b$, $a'' = c$. For a subset A of the universe of G^* we let $A' = \{b \mid (b,b) \in A\}$ and A'' the subset of $\{e_1, v_{11}, v_{12}, v_{21}, v_{22}\}$ defined by

$$\begin{cases} e_1 \in A'' & \text{if } (e_1, e_2) \in A \\ v_{ij} \in A'' & \text{if } (e_1, v_{ij}) \in A \text{ (for } 1 \leq i,j \leq 2). \end{cases}$$

We shall inductively define for every subformula $\psi(z_1, \ldots, z_k, Z_1, \ldots, Z_\ell, Y)$ of φ a formula

$$\psi'(y_1, y_2, x_{11}, x_{12}, x_{21}, x_{22}, z_1', z_1'', \ldots, z_k', z_k'', Z_1', Z_1'', \ldots, Z_\ell', Z_\ell'', Y)$$

such that for all $a_1, \ldots, a_k \in U^{G^*}$ and $A_1, \ldots, A_\ell \subseteq U^{G^*}$ we have

$$\begin{aligned} & G^* \models \psi(a_1, \ldots, a_k, A_1, \ldots, A_\ell, F) \\ \iff \quad & G \models \psi'(e_1, e_2, v_{11}, v_{12}, v_{21}, v_{22}, a_1', a_1'', \ldots, a_k', a_k'', A_1', A_1'', \ldots, A_\ell', A_\ell'', F). \end{aligned} \tag{2}$$

After we have done this, we let

$$\begin{aligned} \varphi^*(y_1, y_2, Y) = \exists x_{11} \exists x_{12} \exists x_{21} \exists x_{22} \Big(& \bigwedge_{\substack{1 \leq i,i',j,j' \leq 2 \\ (i,j) \neq (i',j')}} x_{ij} \neq x_{i'j'} \wedge I x_{11} y_1 \wedge I x_{12} y_1 \wedge I x_{21} y_2 \wedge I x_{22} y_2 \\ & \wedge \varphi'(y_1, y_2, x_{11}, x_{12}, x_{21}, x_{22}, Y) \Big). \end{aligned}$$

Clearly, (2) implies that $\varphi^*(y_1, y_2, Y)$ satisfies (1).

So let us define ψ':

– If $\psi(z_1, z_2) = (z_1 = z_2)$, then $\psi'(z_1', z_1'', z_2', z_2'') = (z_1' = z_2' \wedge z_1'' = z_2'')$.
– If $\psi(z) = Vz$, then $\psi'(z', z'') = \big(z' = z'' \wedge Vz'\big) \vee \big(z' = y_1 \wedge z'' = y_2\big)$.
– If $\psi(z) = Ez$, then $\psi'(z', z'') = \big(z' = z'' \wedge Ez'\big) \vee \bigvee_{1 \leq i,j \leq 2} \big(z' = e_1 \wedge z'' = x_{ij}\big)$.
– If $\psi(z_1, z_2) = Iz_1z_2$, then

$$\begin{aligned} \psi'(z_1', z_1'', z_2', z_2'') = \quad & \big(z_1' = z_1'' \wedge z_2' = z_2'' \wedge I z_1' z_2'\big) \\ & \vee \big(z_1' = y_1 \wedge z_1'' = y_2 \wedge z_2' = y_1 \wedge \textstyle\bigvee_{1 \leq i,j \leq 2} z_2'' = x_{ij}\big) \\ & \vee \textstyle\bigvee_{1 \leq i,j \leq 2} \big(z_1' = x_{ij} \wedge z_1'' = x_{ij} \wedge z_2' = y_1 \wedge z_2'' = x_{ij}\big). \end{aligned}$$

– If $\psi(z, Y) = Yz$, then $\psi'(z', z'', Y) = \big(z' = z'' \wedge Yz'\big)$.
– If $\psi(z, Z) = Zz$, then

$$\begin{aligned} \psi'(z', z'', Z', Z'') = \quad & \big(z' = z'' \wedge Z'z'\big) \\ & \vee \big(z' = y_1 \wedge z'' = y_2 \wedge Z''y_1\big) \\ & \vee \textstyle\bigvee_{1 \leq i,j \leq 2} \big(z' = y_1 \wedge z'' = x_{ij} \wedge Z''x_{ij}\big). \end{aligned}$$

– If $\psi = \neg\chi$, then $\psi' = \neg\chi'$.

– If $\psi = \chi \wedge \xi$, then $\psi' = \chi' \wedge \xi'$. Disjunctions and implications are handled analogously.

– If $\psi = \exists z\chi$, then

$$\psi' = \exists z' \exists z'' \Big(\big(z' = z'' \vee (z' = y_1 \wedge z'' = y_2) \vee (z' = y_1 \wedge \bigvee_{1 \leq i,j \leq 2} z'' = x_{ij})\big) \wedge \chi' \Big).$$

Universal quantification over individual variables is handled analogously.

– If $\psi = \forall Z\chi$, then

$$\psi' = \forall Z' \forall Z'' \Big(\forall z \big(Z''z \to (z = y_1 \vee \bigvee_{1 \leq i,j \leq 2} z = x_{ij})\big) \longrightarrow \chi' \Big).$$

Existential quantification over set variables is handled analogously.

It is easy to verify (2) for every clause of this inductive definition. □

Remark 11. Note that no clauses of the definition of ψ' in the proof above, except the one for quantification over set variables, did introduce any new quantifiers or negations in front of quantifiers.

It is worthwhile to observe that we can also modify the clause for quantification over set variables in such a way that no new quantifiers are introduced; (2) remains true if for $\psi = \forall Z\chi$ we simply let $\psi' = \forall Z' \forall Z''\ \chi'$ and for $\psi = \exists Z\chi$ we let $\psi' = \exists Z' \exists Z''\ \chi'$.

After this modification, ψ' and ψ have the same quantifier structure. We need to introduce additional quantifiers when going from φ' to φ^*, but we can replace the existential quantifiers we introduced in our proof by universal ones. This way, we can achieve that φ and φ^* have the same number of quantifier alternations.

References

[1] H.L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25:1305–1317, 1996.

[2] J. Chen, I.A. Kanj, and W. Jia. Vertex cover: Further observations and further improvements. In *Proceedings of the 25th International Workshop on Graph-Theoretical Concepts in Computer Science*, volume 1665 of *Lecture Notes in Computer Science*, pages 313–324. Springer-Verlag, 1999.

[3] S.S. Cosmadakis. Logical reducibility and monadic NP. In *Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science*, pages 52–61, 1993.

[4] B. Courcelle. Graph rewriting: An algebraic and logic approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume 2, pages 194–242. Elsevier Science Publishers, 1990.

[5] B. Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In G. Rozenberg, editor, *Handbook of graph grammars and computing by graph transformations, Vol. 1 : Foundations*, chapter 5, pages 313–400. World Scientific (New-Jersey, London), 1997.

[6] B. Courcelle. The monadic second-order logic of graphs XII: Planar graphs and planar maps. *Theoretical Computer Science*, 237:1–32, 2000.

[7] R.G. Downey and M.R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[8] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical Logic*. Springer-Verlag, 2nd edition, 1994.

[9] M.R. Fellows and M.A. Langston. Nonconstructive tools for proving polynomial-time decidability. *Journal of the ACM*, 35, 1988.

[10] L.S. Filotti, G.L. Miller, and J. Reif. On determining the genus of a graph in $O(v^{O(g)})$ steps. In *Proceedings of the 11th ACM Symposium on Theory of Computing*, pages 27–37, 1979.

[11] J. Flum, M. Frick, and M. Grohe. Query evaluation via tree-decompositions. To appear in *Journal of the ACM*. A preliminary version of the paper appeared in *Proceedings of the 8th International Conference on Database Theory*, LNCS 1973, Springer-Verlag, 2001.

[12] M.R. Garey and D.S. Johnson. The NP-completeness column: An ongoing guide. *Journal of Algorithms*, 3:89–99, 1982.

[13] J. E. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21:549–568, 1974.

[14] B. Mohar. Embedding graphs in an arbitrary surface in linear time. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 392–397, 1996.

[15] L. Perković and B. Reed. An improved algorithm for finding tree decompositions of small width. *International Journal of Foundations of Computer Science*, 11(3):365–371, 2000.

[16] N. Robertson and P.D. Seymour. Graph minors V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41:92–114, 1986.

[17] N. Robertson and P.D. Seymour. Graph minors XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63:65–110, 1995.

[18] C. Thomassen. The graph genus problem is NP-complete. *Journal of Algorithms*, 10:458–576, 1988.

[19] C. Thomassen. A simpler proof of the excluded minor theorem for higher surfaces. *Journal of Combinatorial Theory, Series B*, 70:306–311, 1997.