

Recognizing Map Graphs of Bounded Treewidth

Patrizio Angelini ✉ 

Department of Mathematics, Natural, and Applied Sciences, John Cabot University, Rome, Italy

Michael A. Bekos ✉ 

Department of Mathematics, University of Ioannina, Greece

Giordano Da Lozzo ✉ 


Department of Engineering, Roma Tre University, Rome, Italy

Martin Gronemann ✉ 

Algorithms and Complexity Group, Technische Universität Wien, Austria

Fabrizio Montecchiani ✉ 

Department of Engineering, University of Perugia, Italy

Alessandra Tappini ✉ 

Department of Engineering, University of Perugia, Italy

Abstract

A map graph is one admitting a representation in which vertices are nations on a spherical map and edges are shared curve segments or points between nations. We present an explicit fixed-parameter tractable algorithm for recognizing map graphs parameterized by treewidth. The algorithm has time complexity that is linear in the size of the graph and, if the input is a yes-instance, it reports a certificate in the form of a so-called witness. Furthermore, this result is developed within a more general algorithmic framework that allows to test, for any k , if the input graph admits a k -map (where at most k nations meet at a common point) or a hole-free k -map (where each point is covered by at least one nation). We point out that, although bounding the treewidth of the input graph also bounds the size of its largest clique, the latter alone does not seem to be a strong enough structural limitation to obtain an efficient time complexity. In fact, while the largest clique in a k -map graph is $\lfloor 3k/2 \rfloor$, the recognition of k -map graphs is still open for any fixed $k \geq 5$.

2012 ACM Subject Classification Theory of computation \rightarrow Fixed parameter tractability; Mathematics of computing \rightarrow Graph algorithms

Keywords and phrases Map graphs, Recognition, Parameterized complexity

Digital Object Identifier 10.4230/LIPIcs.SWAT.2022.8

Funding *Michael A. Bekos*: Partially supported by DFG grant KA812/18-1.

Giordano Da Lozzo: Partially supported by MSCA-RISE project “CONNECT”, N° 734922, and by MIUR, grant 20174LF3T8 “AHeAD: efficient Algorithms for HARnessing networked Data”.

Fabrizio Montecchiani: Partially supported by MIUR, grant 20174LF3T8 “AHeAD: efficient Algorithms for HARnessing networked Data”, and by Dipartimento di Ingegneria, University of Perugia, grants RICBA20ED and RICBA21LG.

Alessandra Tappini: Partially supported by MIUR, grant 20174LF3T8 “AHeAD: efficient Algorithms for HARnessing networked Data”, and by Dipartimento di Ingegneria, University of Perugia, grants RICBA20ED and RICBA21LG.

Acknowledgements We thank the anonymous reviewers of a previous version of this paper for pointing out that the map recognition problem admits an MSO₂ formulation.



© Patrizio Angelini, Michael A. Bekos, Giordano Da Lozzo, Martin Gronemann, Fabrizio Montecchiani, and Alessandra Tappini;

licensed under Creative Commons License CC-BY 4.0

18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2022).

Editors: Artur Czumaj and Qin Xin; Article No. 8; pp. 8:1–8:18



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

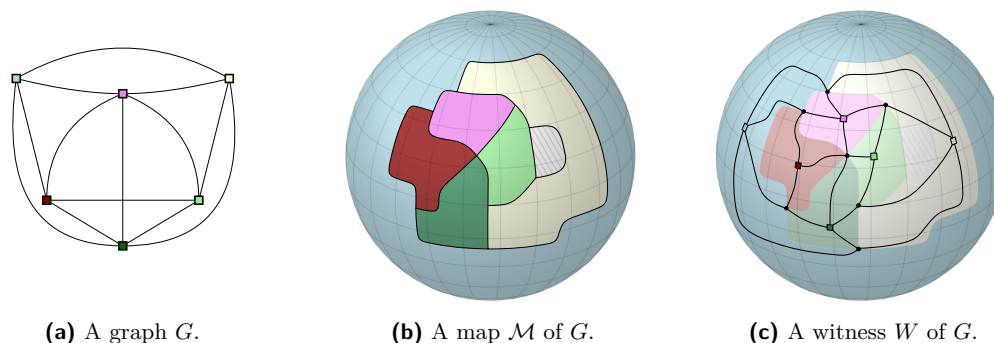
1 Introduction

Planarity is one of the most influential concepts in Graph Theory. Inspired by topological inference problems and by intersection graphs of planar curves, in 1998, Chen, Grigni and Papadimitriou [8] suggested the study of map graphs as a generalized notion of planarity. A *map* of a graph G is a function \mathcal{M} that assigns each vertex v of G to a region $\mathcal{M}(v)$ on the sphere homeomorphic to a closed disk such that no two regions share an interior point, and any two distinct vertices v and w are adjacent in G if and only if the boundaries of $\mathcal{M}(v)$ and $\mathcal{M}(w)$ share at least one point. For each vertex v of G , the region $\mathcal{M}(v)$ is called the *nation* of v . A connected open region of the sphere that is not covered by nations is a *hole*. A graph that admits a map is a *map graph*, whereas a graph that admits a map without holes is a *hole-free map graph*; Figs. 1a and 1b show a graph and a map of it, respectively. Map graphs generalize planar graphs by allowing local non-planarity at points where more than three nations meet. In fact, the planar graphs are exactly those graphs having a map in which at most three nations share a boundary point [8, 19].

Besides their theoretical interest, the study of map graphs is motivated by applications in graph drawing, circuit board design, and topological inference problems [1, 4, 5, 11]. Map graphs are also useful to design parameterized and approximation algorithms for several optimization problems that are NP-hard on general graphs [6, 16, 21, 22, 23].

A natural and central algorithmic question regards the existence of efficient algorithms for recognizing map graphs. Towards an answer to this question, Chen et al. [8, 9] first gave a purely combinatorial characterization of map graphs: A graph is a map graph if and only if it admits a witness, formally defined as follows; see Fig. 1c. A *witness* of a graph $G = (V, E)$ is a bipartite planar graph $W = (V \cup I, A)$ with $A \subseteq V \times I$ and such that $W^2[V] = G$, where the graph $W^2[V]$ is the *half-square* of W , that is, the graph on the vertex set V in which two vertices are adjacent if and only if their distance in W is 2. Here, the vertices in I are meant to represent the adjacencies among nations. Since W can always be chosen to have linear size in the number of vertices of G [9], the problem of recognizing map graphs is in NP. In 1998, Thorup [31] proposed a polynomial-time algorithm to recognize map graphs. However, the extended abstract by Thorup does not contain a complete proof of the result and, to the best of our knowledge, a full version has not appeared yet. Moreover, the proposed algorithm has two drawbacks. First, the time complexity is not specified explicitly (the exponent of the polynomial bounding the time complexity is estimated to be about 120 [10]; see also [5, 28]). Second, it does not report a certificate in the positive case; a natural one would be a witness.

Hence, the problem of finding a simple and efficient recognition algorithm for map graphs remains open. In the last years, several authors focused on graphs admitting restricted types of maps. Aside from the already defined hole-free maps, another notable example consists of the *k-maps*, in which at most k nations meet at a common point; observe that, when $k \geq n - 1$, map graphs and k -map graphs trivially coincide. For instance, Chen studied the density of k -map graphs [7], while in a recent milestone paper on linear layouts Dujmović et al. [20] proved that the queue number of k -map graphs is cubic in k . Note that the algorithm by Thorup [31] cannot be directly used to recognize k -map graphs (unless $k \geq n - 1$). Chen et al. [10] focused on hole-free 4-map graphs and gave a cubic-time recognition algorithm for this graph family. Later, Brandenburg [5] gave a cubic-time recognition algorithm for general (i.e., not necessarily hole-free) 4-map graphs, by exploiting an alternative characterization of these graphs closely related to maximal 1-planarity. Notably, a polynomial-time recognition algorithm for the family of (general or hole-free) k -map graphs with $k > 4$ is still missing. In particular, for $k > 4$, the only result we are aware of is a characterization of 5-map graphs in



■ **Figure 1** (a) A graph G , (b) a map of G - the striped region is a hole, and (c) a witness of G .

terms of forbidden crossing patterns [4]. A different approach for the original problem is the one by Mnich, Rutter, and Schmidt [28], who proposed a linear-time algorithm to recognize the map graphs with an outerplanar witness, which also reports a certificate witness, if any.

We remark that the size of the largest clique in a k -map graph is $\lfloor 3k/2 \rfloor$ (see, e.g., [9]), thus bounding the size of the largest clique does not seem to be a strong enough structural limitation of the input to obtain an efficient time complexity. Despite the notable amount of work, no prior research focuses on further structural parameters of the input graph to design efficient recognition algorithms. In this paper, we address precisely this challenge.

Our contribution. Our main result is a novel algorithmic framework that can be used to recognize map graphs, as well as variants thereof; in particular, hole-free k -map graphs and k -map graphs. Recall that, by setting $k = n - 1$, our algorithm also recognizes (hole-free) map graphs. In fact, we can also compute the minimum value of k within the same asymptotic running time. The proposed algorithm is parameterized by the treewidth [18, 29] of the n -vertex input graph G and its time complexity has a linear dependency in n , while it does not depend on the natural parameter k . Notably, for graphs of bounded treewidth, our algorithm improves over the existing literature [5, 10, 31] in three ways: it solves the problem for *any* fixed k , it can deal with *both* scenarios where holes are or are not allowed in the sought map, and it exhibits an asymptotically *optimal* running time in the input size. The following theorem summarizes our main contribution.

► **Theorem 1.** *Given an n -vertex graph G and a tree-decomposition of G of width t , there is a $O(t^{O(t)} \cdot n)$ -time algorithm that computes the minimum k , if any, such that G admits a (hole-free) k -map. In the positive case, the algorithm returns a certificate in the form of a witness of G within the same time complexity.*

We remark that the problem of recognizing map graphs can be expressed by using MSO_2 logic. Thus the main positive result behind Theorem 1 can be alternatively achieved by Courcelle's theorem [13]. However, with this approach, the dependency of the time complexity on the treewidth is notoriously very high. As a matter of fact, Courcelle's theorem is generally used as a classification tool, while the design of an explicit ad-hoc algorithm remains a challenging and valuable task [15].

To prove Theorem 1, we first solve the decision version of the problem. For a fixed k , we use a dynamic-programming approach, which can deal with different constraints on the desired witness. While we exploit such flexibility to check whether at most k nations intersect at any point and whether holes can be avoided, other constraints could be plugged into the framework such as, for example, the outerplanarity of the witness (as in [28]). In view of this versatility, future applications of our tools may be expected.

Proof strategy. We exploit the characterization in [9] and test for the existence of a suitable witness of the input graph. The crux of our technique is in the computation of suitable records that represent equivalent witnesses and contain only vertices of a tree-decomposition bag. Each such record must carry enough information, in terms of embedding, so to allow testing whether it can be extended with a new vertex or merged with another witness. Moreover, we need to check whether any such witness yields a k -map and, if required, a hole-free one. To deal with the latter property, we provide a strengthening of the characterization in [9], which we believe to be of independent interest, that translates into maintaining suitable counters on the edges of our records. Additional checks on the desired witness can be plugged in the presented algorithmic framework, provided that the records store enough information. One of the main difficulties is hence “sketching” irrelevant parts of the embedded graph without sacrificing too much information. (A similar challenge is faced in the context of different planarity and beyond-planarity problems [17, 25, 27].) Also, when creating such sketches, multiple copies (potentially linearly many) of the same edge may appear, which we need to simplify to keep our records small. The formalization of such records then allows us to exploit a dynamic-programming approach on a tree-decomposition.

Paper structure. Section 2 contains preliminary definitions. Section 3 illustrates basic properties of map graphs that will be used throughout the paper. Section 4 introduces the concept of “sketching” an embedding of a witness, the key ingredient of the algorithmic framework, which we present in Section 5. Section 6 contains open problems raised by our work. The proofs of the statements marked as \star have been omitted.

2 Preliminaries

We only consider finite, undirected, and simple graphs, although some procedures may produce non-simple graphs. In such a case the presence of self-loops or multiple edges will be clearly indicated. Let $G = (V, E)$ be a graph; for a vertex $v \in V$, we denote by $N(v)$ the set of neighbors of v in G , and by $\deg(v)$ the *degree* of v , i.e., the cardinality of $N(v)$.

Embeddings. A *topological embedding* of a graph G on the sphere Σ is a representation of G on Σ in which each vertex of G is associated with a point and each edge of G with a simple arc between its two endpoints in such a way that any two arcs intersect only at common endpoints. A topological embedding of G subdivides the sphere into topologically connected regions, called *faces*. If G is connected, the *boundary* of a face f is a closed walk, that is, a circular list of alternating vertices and edges; otherwise, the boundary of f is a *set* of closed walks. Note that a cut-vertex of G may appear multiple times in any such walk. A topological embedding of G uniquely defines a *rotation system*, that is, a cyclic order of the edges around each vertex. If G is connected, the boundary defining each face can be reconstructed from a rotation system; otherwise, to reconstruct the boundary of every face f , we also need to know which connected components are incident to f . We call the incidence relationship between closed walks of different components and faces the *position system* of G . A *combinatorial embedding* of G is an equivalence class of topological embeddings that define the same rotation and position systems. An *embedded graph* G is a graph along with a combinatorial embedding. A pair of parallel edges e and e' of G with end-vertices v and w is *homotopic* if there is a face of G whose boundary consists of a single closed walk $\langle v, e, w, e' \rangle$.

Tree-decompositions. Let (\mathcal{X}, T) be a pair such that $\mathcal{X} = \{X_1, X_2, \dots, X_\ell\}$ is a collection of subsets of vertices of a graph G , called *bags*, and T is a tree whose nodes are in one-to-one correspondence with the elements of \mathcal{X} . When this creates no ambiguity, X_i will denote both a bag of \mathcal{X} and the node of T whose corresponding bag is X_i . The pair (\mathcal{X}, T) is a *tree-decomposition* of G if: (i) for every edge (u, v) of G , there exists a bag X_i that contains both u and v , and (ii) for every vertex v of G , the set of nodes of T whose bags contain v induces a non-empty (connected) subtree of T .

The *width* of (\mathcal{X}, T) is $\max_{i=1}^{\ell} |X_i| - 1$, while the *treewidth* of G is the minimum width over all tree-decompositions of G . For an n -vertex graph of treewidth t , a tree-decomposition of width t can be found in FPT time [2].

► **Definition 2.** A tree-decomposition (\mathcal{X}, T) of a graph G is called *nice* if T is a rooted tree with the following properties [3].

- (P.1) Every node of T has at most two children.
- (P.2) If a node X_i of T has two children whose bags are X_j and $X_{j'}$, then $X_i = X_j = X_{j'}$. In this case, X_i is a *join bag*.
- (P.3) If a node X_i of T has only one child X_j , then $X_i \neq X_j$ and there exists a vertex $v \in G$ such that either $X_i = X_j \cup \{v\}$ or $X_i \cup \{v\} = X_j$. In the former case X_i is an *introduce bag*, while in the latter case X_i is a *forget bag*.
- (P.4) If a node X_i is a leaf of T , then X_i contains exactly one vertex, and X_i is a *leaf bag*.

Note that, given a tree-decomposition of width t , a nice tree-decomposition can be computed in $O(t \cdot n)$ time (see, e.g., [26]).

3 Basic Properties of Map Graphs and Their Witnesses

The following statements (in a weaker or different form) have already been discussed in the work by Chen et al. [9] and their proofs are omitted here.

Let $G = (V, E)$ be a map graph and let $W = (V \cup I, A)$ be a witness of G , i.e., W is a planar bipartite graph such that $W^2[V] = G$. A vertex $u \in I$ is an *intersection vertex* of W , while a vertex $v \in V$ is a *real vertex* of W . Also, we let $n_V = |V|$, $n_I = |I|$, and $n = n_V + n_I$.

► **Property 3.** A graph is a k -map graph if and only if it admits a witness such that the maximum degree of every intersection vertex is k .

► **Property 4.** A graph G admits a map if and only if each of its biconnected components admits a map. Also, if G admits a hole-free map, then G is biconnected.



■ **Figure 2** (a) Inessential intersection vertices, and (b) a twin-pair.

Let $W = (V \cup I, A)$ be an embedded witness (i.e., with a prescribed combinatorial embedding). An intersection vertex $u \in I$ is *inessential* if $\deg(u) = 2$ and there exists $u' \in I$ such that $N(u) \subset N(u')$; see Fig. 2a. Furthermore, a pair of intersection vertices $u_1, u_2 \in I$ is a *twin-pair* if $N(u_1) = N(u_2) = \{v, w\}$, for some $v, w \in V$, and W contains a face whose

boundary consists of a single closed walk with exactly four edges with end-vertices v, u_1, w, u_2 ; see Fig. 2b. Note that removing an inessential vertex or one vertex of a twin-pair from W does not modify $W^2[V]$.

► **Definition 5.** *An embedded witness of a map graph is compact if it contains neither inessential intersection vertices nor twin-pairs.*

We remark that a compact witness is not necessarily minimal, i.e., it may contain intersection vertices of degree greater than 2 whose removal does not modify its half-square; see also [9]. However, in our setting, removing further information from a witness would have an impact on the proof of Theorem 7 and on the recognition algorithm (Section 5).

The next lemma shows that focusing on compact witnesses is not restrictive.

► **Lemma 6** (\star). *A graph $G = (V, E)$ is a map graph if and only if it admits a compact witness. Also, G is a k -map graph if and only if it admits a compact witness whose intersection vertices have degree at most k .*

In [9], it is observed (without a formal argument) that a map graph is hole-free if and only if it admits a witness whose faces have 4 or 6 edges each. The next characterization improves over this observation and hence can be of independent interest. A connected embedded graph is a *quadrangulation* if each face boundary consists of a single closed walk with 4 edges.

► **Theorem 7** (\star). *A graph is a hole-free map graph if and only if it admits a compact witness that is a biconnected quadrangulation.*

► **Lemma 8** (\star). *A (hole-free) map graph G admits a compact witness with $n \leq 6n_V - 10$ (respectively, $n \leq 3n_V - 4$) vertices.*

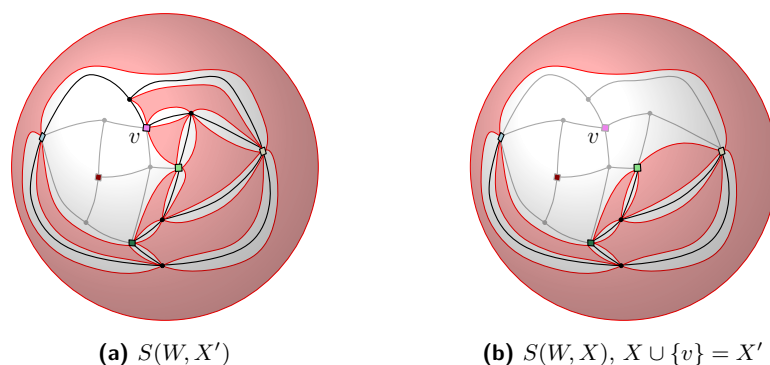
Based on Lemma 8, we can make the following remark.

► **Remark 9.** Without loss of generality, we assume in the following that any compact witness W of G has $n \leq 3n_V - 4$ vertices if G is hole-free, or $n \leq 6n_V - 10$ vertices otherwise.

4 Embedding Sketches

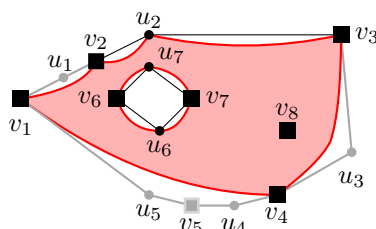
Let G be an input graph. Property 4 allows us to assume that G is biconnected, and thus every witness of G , if any, is connected. Also, by Lemma 6, it suffices to consider compact witnesses.

Let (\mathcal{X}, T) be a nice tree-decomposition of G of width $t = \omega - 1$, i.e., each bag contains at most ω vertices. Given a bag $X \in \mathcal{X}$, we denote by T_X the subtree of T rooted at X , and by $G_X = (V_X, E_X)$ the subgraph of G induced by all the vertices in all bags of T_X . Let $W_X = (V_X \cup I_X, A_X)$ be a compact witness of G_X (in particular, $W_X^2[V_X] = G_X$). Note that, although G is connected, G_X may have multiple connected components. However, since G is connected, each connected component of G_X must contain at least one vertex of X . Moreover, for each connected component C of G_X , there is a connected component C' of W_X such that C' is a witness of C . A vertex of W_X is an *anchor vertex* if it is either a real vertex of X or an intersection vertex whose neighbors in W_X all belong to X . Observe that if an intersection vertex u has a neighbor v in $V_X \setminus X$, then no real vertex in $V \setminus V_X$ is adjacent to v , and therefore there is no way to add further edges to u without creating a false adjacency involving v .



■ **Figure 3** (a) A sketch $S(W, X')$ computed from the witness W of Fig. 1 with respect to a bag X' ($V_{X'} = V$). The anchor vertices of X' are opaque, while the non-anchor vertices are faded. The active boundaries are red and the background of the active faces is light red. (b) A sketch $S(W, X)$, where $X \cup \{v\} = X'$ computed from $S(W, X')$ by applying the deletion operation (Section 5).

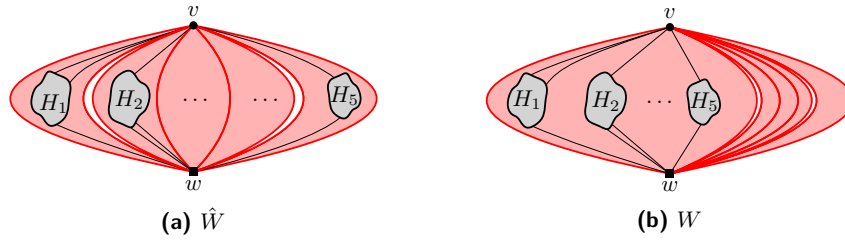
We will exploit anchor vertices to reduce the size of W_X from $O(|V_X|)$ to $O(\omega)$, by “sketching” parts of the embedding that are not relevant¹. The idea of sketching an embedded graph is inspired by a previous work about orthogonal planarity [17]; applying such idea to our problem requires the development of several new tools and concepts, described in the remainder of this section (and partly in Section 2). A face f of W_X is *active* either if its boundary contains only one vertex v (which implies $W_X = (\{v\}, \emptyset)$ and v is an anchor vertex, or if its boundary contains more vertices among which there are at least two anchor vertices; refer to Fig. 3a. The *active boundary* of f (red in Fig. 3a) is obtained by shortcutting all non-anchor vertices of f , where the *shortcut operation* is defined as follows. For a closed walk π and a vertex v in π , shortcutting v consists of removing each occurrence of v (if more than one), together with the edge (u, v) that precedes it in π , and the edge (v, u') that follows it in π , and of adding the edge (u, u') between u and u' in π . Fig. 4 illustrates a single face f



■ **Figure 4** An active boundary (red) made of three closed walks (edges are omitted): $\langle v_1, v_2, u_2, v_3, u_3, v_4, v_1 \rangle$, $\langle u_6, v_6, u_7, v_7 \rangle$, $\langle v_8 \rangle$; vertices u_1, u_3, u_4, v_5, u_5 have been shortcut.

and the corresponding active boundary. The *embedding sketch* (for short the *sketch*) of W_X with respect to X is the embedded graph $S(W_X, X)$ formed by all the vertices and edges that belong to the active boundaries of W_X . For each active boundary B_f of an active face f of W_X , $S(W_X, X)$ has an *active face* f^* (light red in Figs. 3a and 4). Note that $S(W_X, X)$ also has faces that are not active (white in Figs. 3a and 4). Also, the position system of

¹ In the database and data engineering fields, sketching algorithms form a powerful toolkit to compress data in a way that supports answering various queries [12]. Our idea of sketching has some similarities with this concept but serves a different purpose.



■ **Figure 5** Illustrations for the proof of Lemma 10. Modifying the rotation system of \hat{W} such that each H_i lies in B_1 and all other non-extensible active boundaries become empty.

W_X yields a position system for $S(W_X, X)$, since if two closed walks of distinct components of W_X were incident to the same active face f , then the two corresponding closed walks of $S(W_X, X)$ are also incident to the same active face f^* . However, $S(W_X, X)$ may not be bipartite any longer (as in Fig. 4) and it may contain multiple edges (but no self-loops). It is worth noting that the embedding sketch of W_X can be defined with respect to any bag X' as long as $V_{X'} = V_X$ (see Fig. 3a).

We now further refine $S(W_X, X)$ to avoid active boundaries that are not useful for our purposes. Namely, an active boundary is *non-extensible* if it consists of two homotopic parallel edges. Given a witness W of G , the *restriction* of W to G_X is the compact witness $W[G_X]$ of G_X obtained from W by removing all the real vertices not in G_X , all the intersection vertices that are isolated (due to the removal of some real vertices) or inessential, as well as a vertex for each twin-pair until the graph contains none of them. The next lemmas allow us to bound the size of a sketch.

► **Lemma 10.** *If G is a map graph, then it admits a compact witness W with the following property. If $S(W[G_X], X)$ contains $h > 1$ non-extensible active boundaries that share the same pair of end-vertices, then the vertices of W lie in at most one of these h active boundaries.*

Proof. Refer to Fig. 5. Let \hat{W} be a compact witness of G , and suppose $S(\hat{W}[G_X], X)$ contains $h > 1$ non-extensible active boundaries B_1, B_2, \dots, B_h with common end-vertices v, w . Let H_i be the subgraph of \hat{W} that lies inside B_i (if any), for $1 \leq i \leq h$. Since each B_i consists of two parallel edges, v and w separate H_i and $S(\hat{W}[G_X], X) \setminus H_i$. We obtain a new compact witness W of G by modifying the rotation system of \hat{W} so that each H_i lies inside B_1 . ◀

► **Remark 11.** By Lemma 10, we assume in the following that for any compact witness W of G such that, for some $X \in \mathcal{X}$, the sketch $S(W[G_X], X)$ contains $h > 1$ non-extensible active boundaries, the vertices of W lie in at most one of such active boundaries. Therefore, in $S(W[G_X], X)$, we keep only one of the corresponding h pairs of homotopic parallel edges.

► **Lemma 12.** *A sketch $S(W_X, X)$ contains $O(\omega)$ vertices and edges.*

Proof. With a similar argument as in the proof of Lemma 8 we can show that, in W_X , each real vertex in X is adjacent to $O(\omega)$ intersection vertices that are anchor vertices. Therefore, $S(W_X, X)$ contains $O(\omega)$ vertices in total. Concerning the number of edges, since $S(W_X, X)$ is embedded on the sphere, it contains $O(\omega)$ edges such that each pair of edges is either non-parallel or non-homotopic parallel. In addition, since each of these edges participates in at most one homotopic pair by Remark 11, it follows that $S(W_X, X)$ contains $O(\omega)$ edges. ◀

We now exploit the concept of sketch to define an equivalence relation among witnesses.

► **Definition 13.** *Two compact witnesses W_X and W'_X of G_X are X -equivalent if they have the same sketch with respect to X , i.e., $S(W_X, X) = S(W'_X, X)$.*

The next lemma deals with the size of the quotient of such a relation.

► **Lemma 14.** *The X -equivalence relation yields $\omega^{O(\omega)}$ classes for the compact witnesses of G_X .*

Proof. Let n_1 be the number of possible (abstract) graphs that can be obtained from the real vertices of X and all possible sets of intersection vertices. For each such graph, let n_2 be the maximum number of possible rotation and position systems that it can have. It follows that the number of X -equivalent classes is upper bounded by the product of n_1 and n_2 .

Given the set X of real vertices and a compact witness W_X of G_X , any sketch $S(W_X, X)$ contains $O(\omega)$ intersection vertices, as otherwise W_X would contain inessential intersection vertices or twin-pairs. Since each intersection vertex is adjacent to a set of at most ω real vertices, we can bound the number n_{int} of possible sets of intersection vertices by $a \cdot \sum_{i=2}^{\omega} \binom{\omega}{i} < a \cdot 2^{\omega}$, where a is the maximum number of intersection vertices in any sketch that have the same set of neighbors. Since $a \in O(\omega)$, we have that $n_{\text{int}} \in 2^{O(\omega)}$. Let I_X be one of the n_{int} possible sets of intersection vertices. The number n_{abs} of distinct abstract graphs with vertex set $X \cup I_X$ can be upper bounded by the number of possible neighborhoods of a real vertex combined for all real vertices, that is

$$n_{\text{abs}} \leq \prod_{v \in X} \omega^{\deg(v)} = \omega^{\sum_{v \in X} \deg(v)} \leq \omega^{O(\omega)}$$

holds, which yields $n_1 \leq n_{\text{int}} \cdot n_{\text{abs}} \in \omega^{O(\omega)}$.

For a fixed graph S , the number of possible rotation systems n_{rot} is upper bounded by the number of possible permutations of edges around each vertex. Thus we have

$$n_{\text{rot}} \leq \prod_{v \in S} \deg(v)! < \prod_{v \in S} \deg(v)^{\deg(v)} \leq \omega^{O(\sum_{v \in S} \deg(v))} \leq \omega^{O(\omega)}.$$

Each rotation system of S fixes the closed walk of each face of each connected component of S . Since S contains, over all its connected components, at most ω closed walks (at most one for each real vertex in X) and hence at most ω faces, for the number n_{pos} of possible position systems it holds $n_{\text{pos}} \leq \omega^{\omega}$. Therefore we have $n_2 \leq n_{\text{rot}} \cdot n_{\text{pos}} \in \omega^{O(\omega)}$, which yields $n_1 \cdot n_2 \in \omega^{O(\omega)}$, as desired. ◀

5 Algorithmic Framework

Let $G = (V, E)$ be an input graph, let k be an integer, and let (\mathcal{X}, T) be a nice tree-decomposition of G of width $t = \omega - 1$. We present an algorithmic framework to test whether G is a k -map graph or a hole-free k -map graph. Namely, we traverse T bottom-up and equip each bag $X \in \mathcal{X}$ with a suitably defined set of sketches, called *record* R_X . The framework can be tailored by imposing different properties for the records. The next three properties are rather general; the first two are useful to prove the correctness of our approach, as shown in Theorem 22, whereas the third property comes into play when dealing with the efficiency of the approach, and in particular in Lemma 16.

► **Definition 15.** *The record R_X is valid if the following properties hold:*

F1 *For every compact witness W_X of G_X , R_X contains its sketch $S(W_X, X)$.*

F2 *For every entry $r \in R_X$, there is a compact witness W_X of G_X such that $r = S(W_X, X)$.*

F3 *R_X contains no duplicates.*

8:10 Recognizing Map Graphs of Bounded Treewidth

► **Lemma 16.** *For every $X \in \mathcal{X}$, if R_X is valid, it contains $\omega^{O(\omega)}$ entries, each of size $O(\omega)$.*

Proof. By F1–F3, the entries of R_X are all and only the possible sketches of W_X and are all distinct. Hence, $|R_X| \in \omega^{O(\omega)}$ by Lemma 14. Each sketch has size $O(\omega)$ by Lemma 12. ◀

We now describe the additional properties that we incorporate in the framework. In order to verify that G admits a k -map we exploit Property 3, which translates into verifying that, for each sketch, the degree of any intersection vertex is at most k .

► **Definition 17.** *A record R_X is k -map valid if it is valid and it contains a non-empty subset $R_X^* \subseteq R_X$, called subrecord, for which the following additional property holds:*

F4 *For every entry $r \in R_X$, it holds $r \in R_X^*$ if and only if r contains no intersection vertex u with $\deg(u) > k$.*

It is worth observing that, since an intersection vertex of degree k implies the existence of a clique of size k in the input graph G , property F4 is trivially verified when $k \geq \omega$. On the other hand, the size of the largest clique of a k -map graph is $\lfloor 3k/2 \rfloor$ (see, e.g., [9]).

To check whether G has a hole-free k -map, we exploit Theorem 7. Namely, consider a sketch $S(W_X, X)$ and an active boundary B_f of $S(W_X, X)$. Let f be the active face of W_X corresponding to B_f . Note that any edge e that is part of B_f represents a subsequence π_e of a closed walk π in the boundary of f . Therefore, to control the number of edges on the boundary of each face of W_X , for every edge e that is part of an active boundary of $S(W_X, X)$ we also store a counter $c(e) \geq 1$, which represents the number of edges in π_e . If there is an edge e such that $c(e) > 4$, then G does not admit a compact witness W that is a quadrangulation and such that $W_X = W[G_X]$; hence we can avoid storing counters greater than 4. Moreover, for any face f of a compact witness W of G , we know there exist two bags \hat{X}' and \hat{X} in T such that \hat{X}' is the child of \hat{X} , \hat{X} is a forget bag, the active boundary representing f in \hat{X}' has more than one anchor vertex, while the one in \hat{X} has only one anchor vertex (and hence is not part of $S(W_{\hat{X}}, \hat{X})$). We call such an active boundary *complete* in \hat{X}' , as it will not be modified anymore by the algorithm. As such, for each complete active boundary, the sum of the counters of its edges in $S(W_{\hat{X}'}, \hat{X}')$ must be exactly 4, otherwise G does not admit a compact witness W that is a biconnected quadrangulation such that $W_{\hat{X}} = W[G_{\hat{X}}]$.

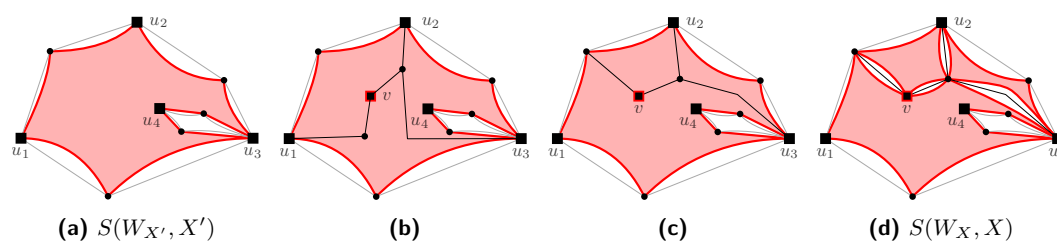
► **Definition 18.** *A record R_X is hole-free valid if it is valid and it contains a non-empty subset $R_X^\circ \subseteq R_X$, called subrecord, for which the following additional property holds:*

F5 *For every entry $r \in R_X$, it holds $r \in R_X^\circ$ if and only if r contains no intersection vertex u with $\deg(u) > k$ and each complete active boundary of r (if any) is such that its edge counters sum up to 4.*

Each leaf bag contains only one vertex v , thus its record consists of one sketch with only one active face whose active boundary is $\langle v \rangle$. Such a record can be computed in $O(1)$ time and it is trivially valid. Also, it is hole-free (and hence k -map) valid, as its unique active boundary is not complete. The next three operations are performed on a non-leaf bag X of T , based on the type of X , to compute a k -map or hole-free valid record R_X , if any.

Deletion operation. Let X be a forget bag whose child X' in T has a k -map (hole-free) valid record $R_{X'}$. Let v be the vertex forgotten by X . We generate R_X from $R_{X'}$ as follows.

For a fixed sketch $S(W_{X'}, X')$ of $R_{X'}$, let $N_I(v) \subseteq N(v)$ be the set of intersection vertices adjacent to v in $S(W_{X'}, X')$. Since v is forgotten by X , all its neighbors have already been processed, thus no vertex in $N_I(v)$ can connect vertices that will be introduced by bags visited after X . Therefore, for every vertex $y \in N_I(v) \cup \{v\}$ and for every sketch $S(W_{X'}, X')$



■ **Figure 6** Illustration for the addition of vertex v . (a) Details of a face of $S(W_{X'}, X')$ that contains all the neighbors of v . (b–c) Two distinct embedded graphs computed from $S(W_{X'}, X')$ by introducing vertex v in different ways (as described in the proof of Lemma 20). (d) The sketch $S(W_X, X)$ obtained by replacing the active boundary of the red face with the new active boundaries corresponding to the three newly created active faces in (c).

of $R_{X'}$, we apply a *deletion operation*, which consists of updating each active boundary B_f of $S(W_{X'}, X')$ containing y ; see Fig. 3b. Namely, let B_f be one of these active boundaries, we distinguish two cases based on whether B_f contains only y or it contains further vertices. Let π_y be the closed walk of B_f that contains all occurrences of y (there might be more than one). If B_f contains only y , we remove π_y (and hence the whole active boundary B_f) from $S(W_{X'}, X')$. If B_f contains further vertices, we shortcut every occurrence of y in π_y . Also for each edge e introduced to shortcut y such that e replaces edges e_1 and e_2 of π_y , we set $c(e) = c(e_1) + c(e_2)$. Observe that, if y has only one neighbor u in π_y , this procedure creates a self-loop at u , which we remove. If this procedure generates more than one pair of homotopic parallel edges with the same pair of end-vertices, then we keep only one such pair. Once all active boundaries have been updated, the resulting embedded graph is stored in R_X . After each sketch of $R_{X'}$ has been processed, we might have produced the same embedded graph for R_X from two distinct sketches of $R_{X'}$; in this case we keep only one copy.

Addition operation. Let X be an introduce bag whose child X' in T has a k -map (hole-free) valid record $R_{X'}$. Let v be the vertex introduced by X and $N_X(v) \subseteq N(v)$ be the set of vertices that are neighbors of v and belong to X . We generate R_X from $R_{X'}$ with the following *addition operation*. For each sketch $S(W_{X'}, X')$ of $R_{X'}$, the high-level idea is to exhaustively generate all possible embedded graphs that can be obtained by introducing v in $S(W_{X'}, X')$. We distinguish two cases.

Case 1. $N_X(v) = \emptyset$. For each active boundary B_f of $S(W_{X'}, X')$, we generate a new embedded graph by adding the closed walk $\langle v \rangle$ to B_f .

Case 2. $N_X(v) \neq \emptyset$. We look for a face f^* of $S(W_{X'}, X')$ that contains all the vertices of $N_X(v)$ on its active boundary B_f (which may consist of multiple closed walks). If such a face does not exist, we discard $S(W_{X'}, X')$. Else, for each such face, we generate a set of entries E_{f^*} as follows. Intuitively, we will insert v inside f^* and generate one entry of E_{f^*} for each possible way in which v can be connected to its neighbors. Namely, we can connect v to its neighbors by means of different intersection vertices and by realizing different permutations of the edges around v and around those neighbors that appear multiple times along some closed walk of B_f ; refer to Fig. 6 for an illustration. Concerning the intersection vertices, we can use those that already belong to B_f and are adjacent only to vertices in $N_X(v)$, as well as we can create new ones. We note that since v has at most $\omega - 1$ neighbors in $N_X(v)$, there are $\sum_{i=1}^{\omega-1} \binom{\omega-1}{i} = 2^{\omega-1}$ possible combinations of intersection vertices (see also the proof of

8:12 Recognizing Map Graphs of Bounded Treewidth

Lemma 14). This is done avoiding inessential intersection vertices and twin-pairs. For each choice of intersection vertices, since the degree of a vertex is $O(\omega)$, there are $\omega^{O(\omega)}$ distinct rotation systems to consider. Additionally, if B_f consists of multiple closed walks, we shall consider all possible permutations of the edges around v that do not cause edge crossings (i.e., any edge permutation in which there are no four edges e_1, e_2, e_3, e_4 in this order around v , such that e_1, e_3 connect v to the vertices of a closed walk π and e_2, e_4 connect v to the vertices of a closed walk π' with $\pi \neq \pi'$), and we consider each of them independently as a new embedded graph. Based on the fixed intersection vertices and rotation system, if the insertion of v does not split f^* into multiple faces, we can suitably update B_f , otherwise we can generate the new active boundaries that appear in place of B_f ; see in particular Fig. 6d. Also, for each newly introduced edge e in a closed walk, we set $c(e) = 1$.

Merge operation. Let X be a join bag whose children X_1 and X_2 in T have k -map (hole-free) valid records R_{X_1} and R_{X_2} , respectively. We generate R_X from R_{X_1} and R_{X_2} . Since X is a join bag, X , X_1 , and X_2 contain the same vertices, whereas G_{X_1} and G_{X_2} only share the vertices in X . Consider any pair of sketches $S(W_{X_1}, X)$ of R_{X_1} and $S(W_{X_2}, X)$ of R_{X_2} . Such sketches share the same set of real vertices, whereas they may have different sets of intersection vertices and different combinatorial embeddings. At high-level, we aim at combining $S(W_{X_1}, X)$ and $S(W_{X_2}, X)$ in all possible ways, provided that the original rotation and position systems of each sketch are preserved and that we never insert a subgraph of one sketch into a non-active face of the other. In practice, we apply the *merge operation*, consisting of the next steps.

- (S.1) We compute all possible unions of the two abstract graphs underlying the two sketches. Namely, let I_{X_1} and I_{X_2} be the sets of intersection vertices of $S(W_{X_1}, X)$ and $S(W_{X_2}, X)$, respectively. We identify each pair of real vertices the two sketches share, and we consider all possible abstract graphs whose set of intersection vertices I_X is such that: (a) $I_X \subseteq I_{X_1} \cup I_{X_2}$; (b) for each intersection vertex of I_{X_1} there is an intersection vertex in I_X with the same set of neighbors, and the same holds for I_{X_2} .
- (S.2) For each generated graph S^* , we compute all combinatorial embeddings, i.e., all possible rotation and position systems yielding a topological embedding on the sphere of S^* . If no such combinatorial embeddings exist, we discard S^* , else we go to the next step.
- (S.3) We generate all possible one-to-one mappings ϕ_1 between intersection vertices of S^* and of $S(W_{X_1}, X)$, and all possible one-to-one mappings ϕ_2 between intersection vertices of S^* and of $S(W_{X_2}, X)$.
- (S.4) We check, for each pair ϕ_1, ϕ_2 , that the restriction of the resulting embedded graph on the real vertices, intersection vertices (up to the mapping defined by ϕ_1 and ϕ_2) and edges of each of the two sketches preserves the corresponding rotation and position systems. If so, we go to the next step; otherwise, we discard the candidate solution.
- (S.5) Since the previous step guaranteed that the active boundaries of each sketch are preserved when looking at the corresponding restriction, we can verify that there is no subgraph of one sketch inside a non-active face of the other.
- (S.6) We suitably update the active boundaries of the resulting embedded graph and we add it to R_X . More precisely, the boundary of a face is active if it does not correspond to a non-active boundary in any of the two sketches and it contains either exactly one anchor vertex or at least two anchor vertices.
- (S.7) We remove inessential intersection vertices and iteratively one intersection vertex for each twin-pair, until there are no twin-pairs.
- (S.8) Once all pairs of sketches have been processed, we remove possible duplicates.

This concludes the description of the main algorithmic steps for proving Theorem 1. Next, we provide lemmas to establish the correctness and the time complexity of these steps.

► **Lemma 19.** *Let X be a forget bag whose child X' in T has a k -map (resp. hole-free) valid record $R_{X'}$. The algorithm either rejects the instance or computes a k -map (resp. hole-free) valid record R_X of X in $\omega^{O(\omega)}$ time.*

Proof. Let v be the vertex forgotten by X . We prove that the record R_X generated by applying the deletion operation is valid, given that $R_{X'}$ is valid. In particular, since we removed possible duplicates, F3 holds and it remains to argue about F1 and F2. To this aim, since X is a forget bag, note that $G_X = G_{X'}$. Hence any compact witness $W_{X'}$ of $G_{X'}$ is also a compact witness of G_X . Moreover, since $R_{X'}$ is valid, it follows by F1 that $R_{X'}$ contains a sketch $S(W_{X'}, X')$ for every compact witness $W_{X'}$. Now since $X' = X \cup \{v\}$, the sketch of $W_{X'}$ with respect to X , namely $S(W_{X'}, X)$, coincides with the one obtained by applying the deletion operation to $S(W_{X'}, X')$. Thus F1 holds for X . Similarly, since $R_{X'}$ is valid, it follows by F2 that every entry of $R_{X'}$ is the sketch $S(W_{X'}, X')$ of a compact witness $W_{X'}$ of $G_{X'}$. Again since $X' = X \cup \{v\}$, the entry of R_X obtained by applying the deletion operation to $S(W_{X'}, X')$ corresponds to the sketch $S(W_{X'}, X)$. Thus F2 holds for X and consequently R_X is valid, as claimed. Suppose now that $R_{X'}$ is k -map valid, i.e., $R_{X'}^* \neq \emptyset$. We show how to check whether a sketch of R_X belongs to R_X^* . Since the deletion operation does not modify the degree of any intersection vertex, the subrecord R_X^* contains all sketches of R_X generated from sketches in $R_{X'}^*$. Based on this observation, we can check whether $R_X^* = \emptyset$ or not. In the former case the algorithm rejects the instance, in the latter case R_X is k -map valid. Suppose that $R_{X'}$ is hole-free valid, i.e., $R_{X'}^\circ \neq \emptyset$. Again the subrecord R_X° contains all sketches of R_X that have been generated from sketches in $R_{X'}^\circ$, and that contain no active boundary whose edge counters sum up to 4. To decide whether an active boundary is complete, it suffices to check whether the parent of X is a forget bag such that the shortcuttings due to the removal of the forgotten vertex make that active boundary a self-loop. If any complete active boundary does not meet this condition, the corresponding sketch does not belong to R_X° . As before if $R_{X'}^\circ = \emptyset$ the algorithm rejects the instance, otherwise R_X is hole-free valid.

By Lemma 16, $R_{X'}$ contains $\omega^{O(\omega)}$ entries, each of size $O(\omega)$. Updating each of them takes $O(\omega)$ time. Also, R_X contains at most as many entries as $R_{X'}$. It follows that removing duplicates can be naively done in $(\omega^{O(\omega)})^2 \in \omega^{O(\omega)}$ time. For the sake of efficiency, if we interpret each rotation and position system together as a number with $\tilde{O}(\omega^2)$ bits, then removing duplicates can be done in $\tilde{O}(\omega^2) \cdot \omega^{O(\omega)} \in \omega^{O(\omega)}$ time by using radix sort (we omit the details as the asymptotic running time would be the same). We have seen that condition F4 is always verified. Checking condition F5 requires scanning each active boundary in R_X and decide whether it is complete or not, and if so to verify whether it will become a self-loop when visiting the parent of X . This can be done in $O(\omega)$ time for each of the $O(\omega)$ active boundaries of each of the $\omega^{O(\omega)}$ sketches, and thus in $\omega^{O(\omega)}$ time overall. Thus R_X and its subrecords can be computed in $\omega^{O(\omega)}$ time, as desired. ◀

► **Lemma 20.** *Let X be an introduce bag whose child X' in T has a k -map (resp. hole-free) valid record $R_{X'}$. The algorithm either rejects the instance or computes a k -map (resp. hole-free) valid record R_X of X in $\omega^{O(\omega)}$ time.*

Proof. Let v be the vertex introduced by X . We prove that the record R_X generated by applying the addition operation is valid, given that $R_{X'}$ is valid. Regarding F1, let $W_{X'}$ and W_X be a witness of $G_{X'}$ and G_X , respectively, such that $W_X[G_{X'}] = W_{X'}$. Since F1

8:14 Recognizing Map Graphs of Bounded Treewidth

holds for $R_{X'}$, we know that $S(W_{X'}, X') \in R_{X'}$. Observe that the only difference between W_X and $W_{X'}$ lies in the presence of vertex v and of a (possibly empty) set I_v of intersection vertices adjacent to v .

If $N_X(v) = \emptyset$, then v forms a trivial closed walk that might be added in any face of $W_{X'}$ that either consists of exactly one anchor vertex or contains at least two anchor vertices (among possibly other non-anchor vertices). We recall that an active face satisfying the mentioned properties corresponds to an active boundary of the witness' sketch. Also, adding the closed walk to a face that contains more than one vertex, but at most one anchor vertex, on its boundary would imply that the resulting witness cannot be augmented to a witness of G , since G is biconnected. Since Case 1 places v in all possible active boundaries of $S(W_{X'}, X')$, we can conclude that $S(W_X, X)$ belongs to R_X .

On the other hand, if $N_X(v) \neq \emptyset$, then all v 's neighbors belong to a common boundary of some face f of $W_{X'}$, as otherwise the rotation system of W_X would not be compatible with a topological embedding (in particular, some edges would cross each other). Hence all v 's neighbors are part of the same active boundary B_f of $S(W_{X'}, X')$. Since Case 2 exhaustively considers all ways in which v can be inserted into B_f , avoiding inessential intersection vertices and twin-pairs (which cannot belong to W_X since it is compact), we can again conclude that $S(W_X, X)$ belongs to R_X . Consequently F1 holds for R_X .

About F2, it suffices to prove that each entry generated by the addition operation is indeed a sketch of some compact witness of G_X with respect to X . Since F2 holds for $R_{X'}$, the addition operation starts from a sketch $S(W_{X'}, X')$ and it generates new entries in which there are neither inessential intersection vertices nor twin-pairs; therefore, such entries are indeed sketches of compact witnesses, as desired.

Concerning F3, if R_X contained two entries r_1, r_2 that are the same (up to a homeomorphism of the sphere), then r_1 and r_2 would have been originated by the same sketch r of $R_{X'}$, as otherwise either r_1 and r_2 would not be the same or F3 would not hold for $R_{X'}$. On the other hand, since the addition operation inserts v in different ways but without repetitions, it cannot generate two entries that are the same starting from a single entry of $R_{X'}$. Thus F3 holds for R_X .

If $R_{X'}$ is k -map valid, we know that R_X^* contains those sketches of $R_{X'}$ for which the addition operation did not introduce intersection vertices of degree larger than k . Based on this observation, we can check whether $R_X^* = \emptyset$ or not. In the former case the algorithm rejects the instance, in the latter case R_X is k -map valid. The case when $R_{X'}$ is hole-free valid can be proved analogously as in the proof of Lemma 19.

Finally, each single entry constructed by the addition operation can be computed in $O(\omega)$ time and R_X contains $\omega^{O(\omega)}$ entries by Lemma 16. Also, condition F4 can be easily verified in $O(\omega)$ time, for each of the $\omega^{O(\omega)}$ sketches of R_X . Checking condition F5 requires scanning each active boundary in R_X and decide whether it is complete or not. This can be done in $O(\omega)$ time, for each of the $O(\omega)$ active boundaries of each of the $\omega^{O(\omega)}$ sketches, and thus in $\omega^{O(\omega)}$ time overall. Thus R_X and its subrecords can be computed in $\omega^{O(\omega)}$ time. ◀

The proof of the next lemma exploits the merge operation.

► **Lemma 21.** *Let X be a join bag whose children X_1 and X_2 in T both have k -map (resp. hole-free) valid records R_{X_1} and R_{X_2} . The algorithm either rejects the instance or computes a k -map (resp. hole-free) valid record R_X of X in $\omega^{O(\omega)}$ time.*

Proof. We prove that the record R_X generated by applying the merge operation is valid, given that R_{X_1} and R_{X_2} are valid. Consider any compact witness W_X of G_X and its restrictions $W_X[G_{X_1}]$ and $W_X[G_{X_2}]$ to G_{X_1} and G_{X_2} , respectively. By definition of restriction, there must

exist a mapping of the intersection vertices of W_X to the intersection vertices of $W_X[G_{X_1}]$ such that when looking at the restriction of W_X to the real and intersection vertices of $W_X[G_{X_1}]$ (up to the above mentioned mapping), the rotation and position systems of $W_X[G_{X_1}]$ are preserved. The same property must hold for $W_X[G_{X_2}]$. These properties clearly carry over to the corresponding sketches $S(W_X, X)$, $S(W_X[G_{X_1}], X)$, and $S(W_X[G_{X_2}], X)$. Since R_{X_1} and R_{X_2} are valid, they contain $S(W_X[G_{X_1}], X)$ and $S(W_X[G_{X_2}], X)$, respectively. Hence, Steps S.1–S.4 guarantee that the aforementioned mapping is considered and that all the above properties hold on the candidate solutions given by the combination of $S(W_X[G_{X_1}], X)$ and $S(W_X[G_{X_2}], X)$. Moreover, any subgraph of W_X that belongs to $W_X[G_{X_1}]$ but not to $W_X[G_{X_2}]$, except for the shared vertices of X , must lie in an active face of $W_X[G_{X_2}]$ (and vice-versa); if this is not the case, then W_X would not be augmentable to a witness of G , since G is biconnected. This property translates into verifying that any subgraph of $S(W_X[G_{X_1}], X)$ lies in an active face of $S(W_X[G_{X_2}], X)$ (and vice-versa). This is achieved in Step S.5. Step S.6 suitably updates the active boundaries so that a boundary is active only if it represents a face of W_X that either consists of exactly one anchor vertex or contains at least two anchor vertices, as by definition of active boundary. Step S.7 removes inessential intersection vertices and twin-pairs, which is a safe operation because W_X is compact. Therefore we can conclude that $S(W_X, X)$ belongs to R_X , and thus F1 holds for R_X . Concerning F2, any entry S in R_X generated by the merge operation, starting from entries $S(W_{X_1}, X) \in R_{X_1}$ and $S(W_{X_2}, X) \in R_{X_2}$, defines a way to combine the combinatorial embeddings of $S(W_{X_1}, X)$ and $S(W_{X_2}, X)$ at common real vertices and at possibly common (based on some mappings ϕ_1 and ϕ_2) intersection vertices. Such information can be used to combine in the same way the corresponding witnesses W_{X_1} and W_{X_2} , which exist because F2 holds for R_{X_1} and R_{X_2} , respectively. On the other hand, such combination yields a compact witness W_X of G_X with respect to X , whose sketch is S , as desired. Thus F2 holds for R_X . In Step S.8 we remove possible duplicates, hence F3 holds by construction for R_X . Therefore R_X is valid. Since the merge operation does not increase the degree of intersection vertices, and since R_{X_1} and R_{X_2} are k -map valid, the subrecord R_X^* contains all sketches of R_X generated from sketches in $R_{X_1}^*$ and $R_{X_2}^*$. If $R_X^* = \emptyset$, the algorithm rejects the instance, otherwise R_X is k -map valid. If R_{X_1} and R_{X_2} are hole-free valid, R_X° contains all sketches of R_X that are generated from sketches in $R_{X_1}^\circ$ and $R_{X_2}^\circ$ and whose complete active boundaries are such that the edge counters sum up to 4. If $R_X^\circ = \emptyset$, the algorithm rejects the instance, otherwise R_X is hole-free valid.

Concerning the time complexity, we process each pair of sketches, one in R_{X_1} and one in R_{X_2} , and since both R_{X_1} and R_{X_2} are valid, we have $\omega^{O(\omega)}$ such pairs. Each of Steps S.1, S.2, and S.3 generates $\omega^{O(\omega)}$ new entries, and each entry is computed in $O(\omega)$ time. The remaining steps all run in $O(\omega)$ time for each processed entry. Condition F4 can be easily verified in $O(\omega)$ time, for each of the $\omega^{O(\omega)}$ sketches of R_X . Furthermore, verifying condition F5 requires scanning the active boundaries of each entry in R_X and deciding whether it is complete or not. This can also be done in $O(\omega)$ time for each of the $O(\omega)$ active boundaries of each of the $\omega^{O(\omega)}$ sketches, and thus in $\omega^{O(\omega)}$ time overall. Consequently, R_X and its subrecords can be computed in $\omega^{O(\omega)}$ time. ◀

Lemmas 19–21 imply the next theorem, which summarizes the correctness of the approach.

► **Theorem 22.** *Let G be a graph in input to the algorithm, along with a nice tree-decomposition (T, \mathcal{X}) of G and an integer $k > 0$. Graph G is a k -map graph, respectively a hole-free k -map graph, if and only if the algorithm reaches the root ρ of T and the record R_ρ is k -map valid, respectively hole-free valid.*

We are finally ready to prove Theorem 1. We recall that if $k \geq n - 1$, recognizing n -vertex (resp. hole-free) k -map graphs coincides with recognizing general n -vertex (resp. hole-free) map graphs.

Proof of Theorem 1. We first discuss the decision version of the problem for a fixed $k > 0$. Namely, the algorithm described below is used in a binary search to find the optimal value of k . Recall that t is the width of the tree decomposition (i.e., $\omega = t + 1$). Note that, if G is a positive instance, then k varies in the range $[1, t + 1]$, since the size of the largest clique of G is at most $t + 1$. Thus the algorithm is executed $O(\log t)$ times, which however does not affect the asymptotic running time.

If G is not biconnected, by Property 4, it is not hole-free, and it is k -map if and only if all its biconnected components are k -map. Hence we run our algorithm on each biconnected component independently. Theorem 22 implies the correctness of the algorithm (which assumes the input graph to be biconnected).

For the time complexity, suppose that G has $h \geq 1$ biconnected components and let n_i be the size of the i -th component C_i , for each $i \leq h$. Decomposing G into its biconnected components takes $O(n + m)$ time [30], where m is the number of edges of G and, since G has treewidth t , it holds $m \in O(n \cdot t^2)$. Given a tree-decomposition of G with $O(n)$ nodes and width t , we can easily derive a tree-decomposition (T_i, \mathcal{X}_i) for each C_i in overall $O(n)$ time, such that each T_i has $O(n_i)$ nodes and width at most t . Then we can apply the algorithm in [3] to obtain, in $O(n_i)$ -time, a nice tree-decomposition of C_i with $O(n_i)$ nodes without increasing the original width. Since each bag is processed in $t^{O(t)}$ time by Lemmas 19–21, the algorithm runs in $t^{O(t)} \cdot n_i$ time for each C_i . Since $\sum_{i=1}^h n_i \in O(n)$, decomposing the graph and applying the algorithm to all its biconnected components takes $t^{O(t)} \cdot n$ time.

To reconstruct a witness of a yes-instance, we store additional pointers for each record (a common practice in dynamic programming). Namely, for each sketch S of a record R_X of a bag X , we store a pointer to the sketch of the child bag X' that generated S , if X is an introduce or forget bag, and we store two pointers to the two sketches of the children bags X_1 and X_2 that generated S , if X is a join bag. With these pointers at hand, we can apply a top-down traversal of T , starting at any sketch of the non-empty subrecord of ρ , and reconstruct the corresponding witness W by incrementally combining the retrieved sketches, except at forget bags (the only points in which we lose information). Suppose first that G is a k -map graph but not hole-free. If G is not biconnected, a witness W^* of G is obtained by merging the witnesses of its biconnected components. Note that distinct witnesses corresponding to distinct biconnected components of G can only share real vertices. Thus, each intersection vertex of W^* has degree at most k and W^* is a certificate by Property 3. Suppose now that G is a hole-free k -map graph. Then G is biconnected and the resulting witness is a biconnected quadrangulation whose intersection vertices have degree at most k , a certificate by Theorem 7. \blacktriangleleft

6 Open Problems

Recognizing general map graphs efficiently remains a major algorithmic challenge. To restrict the complexity of the input, further parameters of interest might be the cluster vertex deletion number [24] and the clique-width [14] of the input graph, as well as the treewidth of the putative witness [28]. Another interesting line of research would be generalizing our framework to recognize (g, k) -map graphs, i.e., those graphs that admit a k -map on a surface of genus g (see, e.g., [19]).

References

- 1 Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Ignaz Rutter. Intersection-link representations of graphs. *J. Graph Algorithms Appl.*, 21(4):731–755, 2017.
- 2 Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.
- 3 Hans L. Bodlaender and Ton Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996. doi:10.1006/jagm.1996.0049.
- 4 Franz J. Brandenburg. Characterizing 5-map graphs by 2-fan-crossing graphs. *Discret. Appl. Math.*, 268:10–20, 2019.
- 5 Franz J. Brandenburg. Characterizing and recognizing 4-map graphs. *Algorithmica*, 81(5):1818–1843, 2019.
- 6 Zhi-Zhong Chen. Approximation algorithms for independent sets in map graphs. *J. Algorithms*, 41(1):20–40, 2001.
- 7 Zhi-Zhong Chen. New bounds on the edge number of a k -map graph. *J. Graph Theory*, 55(4):267–290, 2007. doi:10.1002/jgt.20237.
- 8 Zhi-Zhong Chen, Michelangelo Grigni, and Christos H. Papadimitriou. Planar map graphs. In *STOC*, pages 514–523. ACM, 1998.
- 9 Zhi-Zhong Chen, Michelangelo Grigni, and Christos H. Papadimitriou. Map graphs. *J. ACM*, 49(2):127–138, 2002.
- 10 Zhi-Zhong Chen, Michelangelo Grigni, and Christos H. Papadimitriou. Recognizing hole-free 4-map graphs in cubic time. *Algorithmica*, 45(2):227–262, 2006.
- 11 Zhi-Zhong Chen, Xin He, and Ming-Yang Kao. Nonplanar topological inference and political-map graphs. In *SODA*, pages 195–204. ACM/SIAM, 1999.
- 12 Graham Cormode. Data sketching. *ACM Queue*, 15(2):60, 2017.
- 13 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.
- 14 Bruno Courcelle, Joost Engelfriet, and Grzegorz Rozenberg. Handle-rewriting hypergraph grammars. *J. Comput. Syst. Sci.*, 46(2):218–270, 1993.
- 15 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
- 16 Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Fixed-parameter algorithms for (k, r) -center in planar graphs and map graphs. *ACM Trans. Algorithms*, 1(1):33–47, 2005.
- 17 Emilio Di Giacomo, Giuseppe Liotta, and Fabrizio Montecchiani. Orthogonal planarity testing of bounded treewidth graphs. *J. Comput. Syst. Sci.*, 125:129–148, 2022.
- 18 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.
- 19 Vida Dujmovic, David Eppstein, and David R. Wood. Structure of graphs with locally restricted crossings. *SIAM J. Discret. Math.*, 31(2):805–824, 2017. doi:10.1137/16M1062879.
- 20 Vida Dujmović, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, and David R. Wood. Planar graphs have bounded queue-number. *J. ACM*, 67(4):22:1–22:38, 2020. URL: <https://dl.acm.org/doi/10.1145/3385731>, doi:10.1145/3385731.
- 21 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar f -deletion: Approximation, kernelization and optimal FPT algorithms. In *FOCS*, pages 470–479. IEEE, 2012.
- 22 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. Decomposition of map graphs with applications. In *ICALP*, volume 132 of *LIPICs*, pages 60:1–60:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
- 23 Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Bidimensionality and geometric graphs. In *SODA*, pages 1563–1575. SIAM, 2012.

8:18 Recognizing Map Graphs of Bounded Treewidth

- 24 Falk Hüffner, Christian Komusiewicz, Hannes Moser, and Rolf Niedermeier. Fixed-parameter algorithms for cluster vertex deletion. *Theory Comput. Syst.*, 47(1):196–217, 2010.
- 25 Bart M. P. Jansen, Daniel Lokshantov, and Saket Saurabh. A near-optimal planarization algorithm. In *SODA*, pages 1802–1811. SIAM, 2014.
- 26 Ton Kloks. *Treewidth, Computations and Approximations*, volume 842 of *LNCS*. Springer, 1994.
- 27 Tomasz Kociumaka and Marcin Pilipczuk. Deleting vertices to graphs of bounded genus. *Algorithmica*, 81(9):3655–3691, 2019.
- 28 Matthias Mnich, Ignaz Rutter, and Jens M. Schmidt. Linear-time recognition of map graphs with outerplanar witness. *Discret. Optim.*, 28:63–77, 2018.
- 29 Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.
- 30 Robert Endre Tarjan and Uzi Vishkin. Finding biconnected components and computing tree functions in logarithmic parallel time (extended summary). In *FOCS*, pages 12–20. IEEE, 1984. doi:10.1109/SFCS.1984.715896.
- 31 Mikkel Thorup. Map graphs in polynomial time. In *FOCS*, pages 396–405. IEEE, 1998.