A Linear Time Algorithm for Embedding Arbitrary Knotted Graphs into a 3-Page Book

Vitaliy Kurlin[1,2] and Christopher Smithers[2]

[1] Microsoft Research Cambridge, 21 Station Road, Cambridge CB1 2FB, UK
vitaliy.kurlin@gmail.com, http://kurlin.org
[2] Durham University, Durham DH1 3LE, United Kingdom
christopher.smithers@durham.ac.uk

Abstract. We introduce simple codes and fast visualization tools for knotted structures in complicated molecules and brain networks. Knots, links and more general knotted graphs are studied up to an ambient isotopy in Euclidean 3-space.

A knotted graph can be represented by a plane diagram or a Gauss code. First we recognize in linear time if an abstract Gauss code represents a graph embedded in 3-space. Second we design a fast algorithm for drawing any knotted graph in the 3-page book, which is a union of 3 half-planes along their common boundary.

The complexity of the algorithm is linear in the length of a Gauss code. Three-page embeddings are encoded in such a way that the isotopy classification for graphs in 3-space reduces to a word problem in finitely presented semigroups.

1 Introduction: Motivation and Problems on Knotted Structures

This is an extended version of the conference paper [13] with extra Appendices B, C, D that describe key stages of the full algorithm for drawing 3-page embeddings.

Knotted structures are common in nature. For example, microscopic lines in liquid-crystals [18] or Reeb graphs of complex shapes [2] can be knotted. Figure 1 shows large brain neurons with many branching points. These structures are usually huge and more complicated than simple closed curves studied in classical knot theory.

Pictures of knots can be attractive for humans, but robots would prefer a smaller form or codes representing the same knotted object. Such codes are needed for automatic analysis, however a final output is also important to visualise. We summarise our requirements for processing knotted structures in the following 3 problems.

• **Modeling**: find a mathematical model for all possible knotted structures in $\mathbb{R}^3$.

• **Encoding**: represent any knotted structure by a simple code in a computer memory.

• **Visualization**: design a fast algorithm to visualize knotted structures given by codes.

Our suggested model for knotted structures is a possibly disconnected graph with branching vertices and multiple edges that might be knotted in 3-space, see Definition 1. For instance, any knot in 3-space is a non-self-intersecting closed curve or a loop.

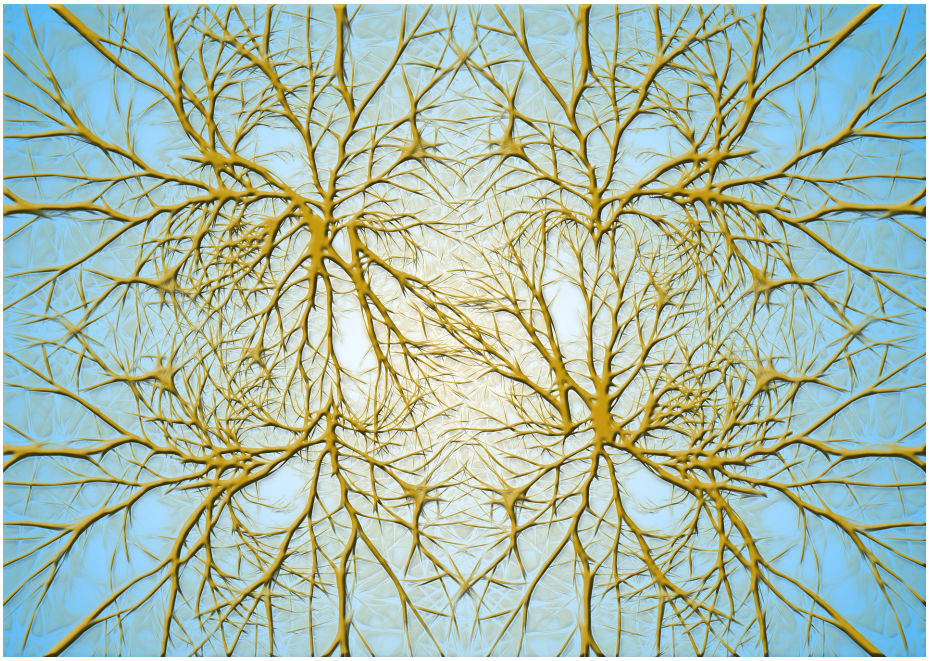

Fig. 1. Neurons in the brain form a large knotted graph with many branching points in 3-space.

Knots live in 3-space, but it is easier to draw their planar projections with double crossings. Such plane diagrams are usually represented by Gauss codes that specify the order of overcrossings and undercrossings along a knot. We will extend classical Gauss codes of knots and links to arbitrary knotted graphs in Definition 4.

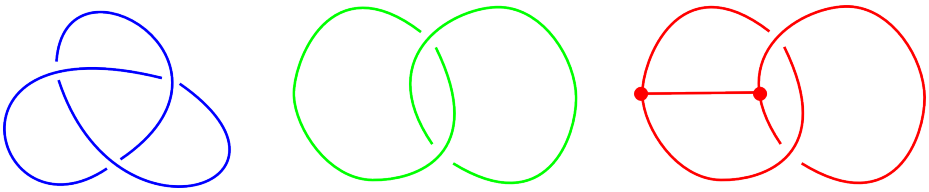

Fig. 2. Plane diagrams (projections) of the trefoil, the Hopf link and a simple knotted graph.

A random code (of a required form) may not represent a real knotted graph, because a planar drawing may need extra crossings. We solve this planarity problem for Gauss codes of knotted graphs in Theorem 9. Our algorithm checks if a Gauss code is realized by a graph in 3-space with a linear time complexity in the length of the given code.

Starting from any realizable Gauss code, we draw a corresponding graph in the *3-page book*, see Theorem 12. This book consists of 3 half-planes attached along their

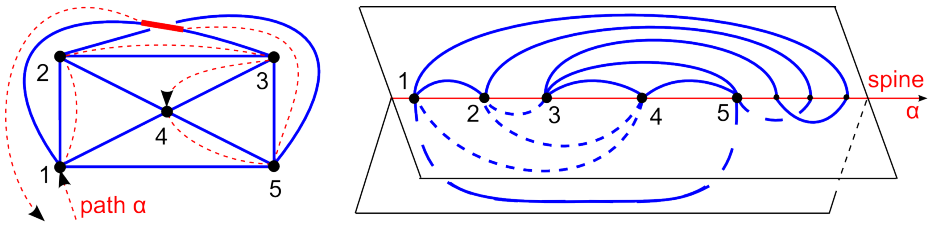

Fig. 3. Straighten a path α to build a 3-page embedding of a graph $K_5 \subset \mathbb{R}^3$ from its diagram.

common boundary α called the *spine*. It is well-known that any graph can be topologically embedded in the 3-page book [1, Theorem 5.4]. However, an embedded graph may cross many times the spine of the book. It is only known that $O(|E| \log |V|)$ spine crossings suffice for embedding a graph with $|V|$ vertices and $|E|$ edges [6].

We largely strengthen the former result by designing a linear time algorithm to continuously move any graph embedded in 3-space to a graph within 3 pages. We review other related work throughout the paper. Figure 3 is a high-level illustration of the fast algorithm for a 3-page embedding of the graph $K_5 \subset \mathbb{R}^3$. Appendix A contains more details on the following advantages of 3-page embeddings over plane diagrams.

• Theorem 13 encodes 3-page embeddings of all knotted graphs in 3-space by easy linear codes that form a finitely presented semigroup.

• Theorem 14 decomposes any topological equivalence between 3-page embeddings of knotted graphs into finitely many local relations between 3-page codes.

2 Key Concepts on Knotted Graphs and Isotopy in 3-Space

A *homeomorphism* between spaces is a bijection that is continuous in both directions. An *embedding* of one space into another is a continuous function $f : X \to Y$ that induces a homeomorphism between X and its image $f(X) \subset Y$.

We study embeddings of undirected finite graphs, possibly disconnected and with loops or multiple edges. The concept of a knotted graph extends the classical theory of knots to arbitrary graphs considered up to isotopy in 3-space $\mathbb{R}^3$.

Definition 1 *A* knotted graph $G \subset \mathbb{R}^3$ *is an embedding of a finite graph* G. *An ambient* isotopy *between knotted graphs* $G, H \subset \mathbb{R}^3$ *is a continuous family of homeomorphisms* $f_t : \mathbb{R}^3 \to \mathbb{R}^3$, $t \in [0, 1]$ *such that* $f_0 = \mathrm{id}$ *is the identity map on* $\mathbb{R}^3$ *and* $f_1(G) = H$.

An isotopy between directed graphs is similarly defined and should respect directions of edges. If the underlying graph G is a circle S^1, then a knotted graph is a *knot*. If G is a disjoint union of several circles, $G \subset \mathbb{R}^3$ is a *link*. A link isotopic to a union of disjoint circles in $\mathbb{R}^2$ is *trivial*. The simplest non-trivial knot is the *trefoil* in the 1st picture of Figure 2. The simplest non-trivial link is the *Hopf* link in the middle of Figure 2.

If an ambient isotopy keeps a small neighborhood of each vertex of a knotted graph in one moving plane, the graph is called *rigid*. Rigid knotted graphs with vertices of only degree 4 are sometimes called *singular knots*, because they consist of one or several circles intersecting each other at singular points.

Definition 2 *A* plane diagram *D of a knotted graph $G \subset \mathbb{R}^3$ is the image of G under a projection $\mathbb{R}^3 \to \mathbb{R}^2$ from 3-space $\mathbb{R}^3$ to a horizontal plane $\mathbb{R}^2$. In a general position we assume that all intersections of a plane diagram D are double* crossings *so that the crossings and the projections of all vertices of G are distinct. For each crossing of D, we specify one of two intersecting arcs that* crosses over *another arc.*

The key problem in knot theory is to efficiently classify knots and graphs up to ambient isotopy. The first natural step is to reduce the dimension from 3 to 2. Any isotopy of knotted graphs can be realized by finitely many moves on plane diagrams. The following result extends Reidemeister's theorem from knots to any knotted graphs.

Theorem 3 *[8] Two plane diagrams represent isotopic knotted graphs in 3-space $\mathbb{R}^3$ if and only if the diagrams can be obtained from each other by an isotopy in $\mathbb{R}^2$ and finitely many Reidemeister moves in Fig. 4. (The move* R5 *is only for rigid graphs, the move* R5′ *is only for non-rigid graphs.)*

The move R4 is shown in Fig. 4 only for a degree 4 vertex, moves for other degrees are similar. The move R5 turns a small neighborhood of a vertex in the plane upside down. So a cyclic order of edges at vertices is preserved in rigid graphs. The move R5′ can arbitrarily reorder all edges at a vertex. Theorem 3 formally includes all symmetric images of moves in Figure 4.

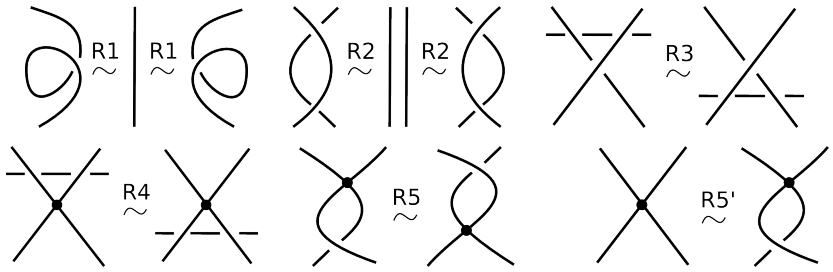

Fig. 4. These Reidemeister moves on diagrams generate any isotopy of graphs in $\mathbb{R}^3$.

The Reidemeister moves or their analogs on Gauss codes are not local as they involve distant parts of a graph or a Gauss code. This non-locality is a key obstacle for simplifying codes of knots. That is why we later consider 3-page embeddings that allow only finitely many local moves, see Theorems 12, 13, 14.

3 Gauss Codes of Knotted Graphs and Abstract Gauss Codes

A standard way to encode a plane diagram of a knot is to write down labels of crossings in a Gauss code. The Gauss code of a link has several words corresponding to all connected components of the link. We extend this classical concept to any knotted graphs $G \subset \mathbb{R}^3$. If a component of a knotted graph $G \subset \mathbb{R}^3$ is a circle without vertices, we add a *base point* (a degree 2 vertex) to this circle.

Definition 4 *Let $D \subset \mathbb{R}^2$ be a plane diagram of a knotted graph G with vertices $A, B, C, \dots$ We fix directions of all edges of G and arbitrarily label all crossings of D by $1, 2, \dots, n$. Then each crossing of D has the* sign *locally defined in Figure 5.*

The Gauss code *W of the diagram D consists of all words W_{AB}, where each word W_{AB} is associated to a directed edge from a vertex A to a vertex B as follows:*

• *W_{AB} starts with A, finishes with B and has the labels of all crossings in AB;*

• *if AB goes under another edge at a crossing i with a sign $\varepsilon \in \{\pm\}$ as in Figure 5, we add the superscript ε to i and get the symbol i^ε with the sign ε in W_{AB}.*

In the plane diagram the edges at each vertex A of the graph G are clockwisely ordered in $\mathbb{R}^2$, so the Gauss code also specifies a cyclic order of all edges at the vertex A.

If G is a knot, Definition 4 requires at least one degree 2 vertex (a base point) on the circle G. Then we may ignore degree 2 vertices and consider W as a *cyclic* word.

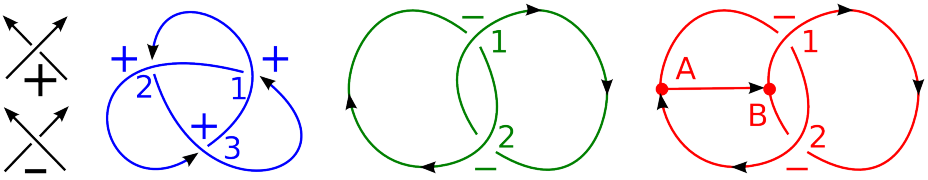

Fig. 5. Local rules for assigning signs of crossings in plane diagrams of knotted graphs.

In Figure 5 the diagram of the blue trefoil has the cyclic Gauss code $12^+31^+23^+$. The diagram of the red knotted graph has the Gauss code $W = \{AB;\ A1^-2A;\ B12^-B\}$ with the cyclic orders of edges at vertices $(AB, 2A, A1^-)$ and $(AB, B1, 2^-B)$. In this example each edge is denoted by the pair of its endpoints . In general, if there are multiple edges with the same endpoints, we use distinct labels for all different edges.

A Gauss code of any undirected graph depends on a choice of extra degree 2 vertices, directions of edges, an order of crossings. If a plane diagram of a knotted graph corresponds to a Gauss code, then this diagram is unique up to isotopy in the plane. We explicitly construct a plane diagram from a Gauss code in the proof of Theorem 9.

Here is a naive approach to drawing a plane diagram represented by a Gauss-like code W. We can plot vertices $A, B, C, \dots$ and crossings $1, 2, \dots, n$ anywhere in $\mathbb{R}^2$.

Since W specifies the cyclic order of edges at each vertex A in Definition 4, we may draw short arcs around A in a correct cyclic order. Now we should connect all vertices and crossings that have adjacent positions in the code W by continuous non-intersecting arcs in the plane.

The last step fails for the word 12^+1^+2 that does not encode any plane diagram. Indeed, if we try to draw a closed curve with 2 self-intersections as required by 12^+1^+2, we have to add a 3rd intersection (a virtual crossing) to make the curve closed. This obstacle can be resolved if we draw a diagram on a torus as in Figure 7, because we can hide a virtual crossing by adding a handle. Another approach is to embrace virtual crossings, which has led to virtual knots.

If we study properly embedded graphs, we need to recognize planarity of Gauss codes, namely we will determine if a Gauss code W represents the plane diagram of some knotted graph $G \subset \mathbb{R}^3$. So we first introduce abstract Gauss codes in Definition 5 and then recognize their planarity in the general case of knotted graphs in Theorem 9.

Definition 5 *Let the alphabet consist of m letters $A, B, C, \dots$ and $3n$ symbols i, i^+, i^- for $i = 1, \dots, n$. An* abstract Gauss code W *is a set of words such that*

• *the first and last symbols of each word in the code W are letters (of vertices),*

• *the set of symbols in all words (apart from the initial and final letters) contains, for each $i = 1, \dots, n$, the symbol i and exactly one symbol from the pair i^+, i^-.*

Each of the m letters defines a cyclic order of all symbols adjacent to this letter. The length $|W|$ *is the total length of all words minus the number of words.*

The Gauss code of any plane diagram of a knotted graph G from Definition 4 satisfies the conditions above. Indeed, the letters $A, B, C, \dots$ denote (projections of) vertices of G. Then every edge contains crossings labeled by i, i^+ or i^- for $i = 1, \dots, n$.

The clockwise order of edges around any vertex A in the plane diagram of G in $\mathbb{R}^2$ defines the cyclic order of vertices and crossings adjacent to A. If a component of G is a circle, we may remove its vertices of degree 2 and write the remaining symbols as in the cyclic code $12^+31^+23^+$ of the trefoil in Figure 5. The total number of these symbols equals the double number of crossings.

4 Planarity Criterion for Gauss Codes of Knotted Graphs

The planarity problem is to determine whether it is possible to draw a plane diagram represented by an abstract Gauss code W. To avoid potential self-intersections, we shall draw a diagram not in the plane, but in the Gauss surface $S(W)$ defined below.

First we introduce the abstract graph $G(W)$ describing the adjacency relations between symbols in a Gauss code W. Then we attach disks to $G(W)$ to get the surface $S(W)$ containing a required diagram without self-intersections. The criterion of planarity will check if the surface $S(W)$ is a topological sphere S^2.

Definition 6 *Any abstract Gauss code W with m letters $A, B, \ldots$ and $2n$ symbols from $\{i, i^+, i^- \mid i = 1, \ldots, n\}$ gives rise to the Gauss graph $G(W)$ with $m + n$ vertices labeled by $A, B, \ldots$ and $1, 2, \ldots, n$.*

We connect vertices p, q by a single edge in $G(W)$ if p, q (possibly with signs) are adjacent symbols in W. Below when we travel along an edge from p to q, we record our path by $(p, q)_+$ if q follows p in the code W (in the cyclic order), otherwise by $(p, q)_-$.

We define unoriented cycles in the graph $G(W)$ by going along edges and turning at vertices according to the following rules illustrated in Figure 6:

• *if we came to one of the vertices $A, B, C, \ldots$ from its neighbor, then we turn to the next neighbor in the clockwise order specified in the Gauss code W;*

• *at each vertex labeled by $i \in \{1, \ldots, n\}$ we turn to the next edge by one of the rules below for a unique possible choice of $\delta \in \{+, -\}$ and both $\varepsilon \in \{+, -\}$*

$$(p, i)_+ \to (i^\delta, q)_\delta, \quad (p, i)_- \to (i^\delta, q)_{-\delta}, \quad (p, i^+)_\varepsilon \to (i, q)_{-\varepsilon}, \quad (p, i^-)_\varepsilon \to (i, q)_\varepsilon.$$

We stop traversing cycles when every edge was passed once in each direction. The Gauss surface *$S(W)$ is obtained from $G(W)$ by gluing a disk to each cycle.*

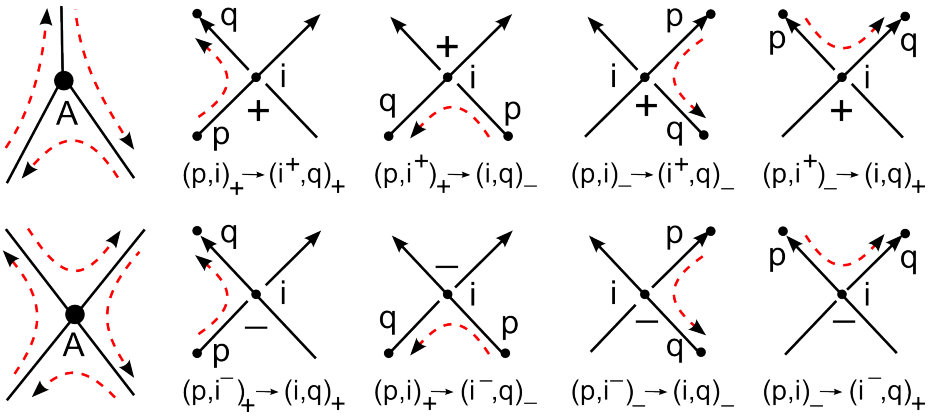

Fig. 6. Interpretation of the 'turning-left' rules for traversing cycles in the Gauss graph $G(W)$.

The number of edges in the graph $G(W)$ equals the length $|W|$ of the code W. The rules for traversing cycles in Definition 6 geometrically mean that at each vertex or crossing we turn left to a unique edge and can pass every edge exactly once in each direction. Hence the Gauss surface of any abstract Gauss code is a compact orientable surface without boundary. From now on we assume that all diagrams, Gauss graphs and surfaces are connected. Otherwise each connected component is considered separately.

Lemma 7 *For the Gauss code W of any connected plane diagram of a knotted graph $G \subset \mathbb{R}^3$, the Gauss surface $S(W)$ is homeomorphic to a topological sphere S^2.*

Proof. We assume that the given diagram D is contained in a sphere S^2 instead of a plane $\mathbb{R}^2$. Then the Gauss graph $G(W)$ can be identified with the diagram D, though $G(W)$ was introduced as an abstract graph not embedded into any space. When we traverse the cycles in $D = G(W)$ from Definition 6, we pass over the boundaries of all connected components of $S^2 - D$. Indeed, each time we turn left in the diagram $D \subset S^2$ according to the geometric rules in Figure 6. Hence the Gauss surface $S(W)$ can be identified with the sphere S^2 containing the diagram $D = G(W)$. □

Example 8 *We construct the Gauss surface of the abstract Gauss code* $W = 12^+1^+2$*, whose diagram with one virtual crossing is in Figure 7. For simplicity, we removed the degree 2 vertex from the circle and consider the word* 12^+1^+2 *in the cyclic order.*

Then 4 pairs 12^+*,* 2^+1^+*,* 1^+2*,* 21 *of adjacent symbols in the code* W *lead to the Gauss graph* $G(W)$ *whose 2 vertices with labels* $1, 2$ *are connected by 4 edges with labels* $(1, 2^+)$*,* $(2^+, 1^+)$*,* $(1^+, 2)$*,* $(2, 1)$*, see Fig. 7. Recall that the edges labeled by* $(2, 1)$ *and* $(2^+, 1^+)$ *meet at a non-avoidable virtual crossing in the plane, but the abstract Gauss graph* $G(W)$ *has only 2 vertices.*

If we start traveling from the edge $(1, 2^+)_+$ *in the same direction as in* W*, the next edge should be* $(2, 1^+)_-$ *by the rule* $(p, i^+)_\varepsilon \to (i, q)_{-\varepsilon}$*, where* $p = 1$*,* $i = 2$*,* $\varepsilon = +$ *uniquely determine the next symbol* $q = 1^+$ *from the code* W *(going from* 2 *in the opposite direction). After passing the second edge* $(2, 1^+)_-$*, we return to the first edge* $(1, 2^+)_+$ *by the same rule* $(p, i^+)_\varepsilon \to (i, q)_{-\varepsilon}$ *for* $p = 2$*,* $i = 1$*,* $\varepsilon = -$*,* $q = 2^+$*.*

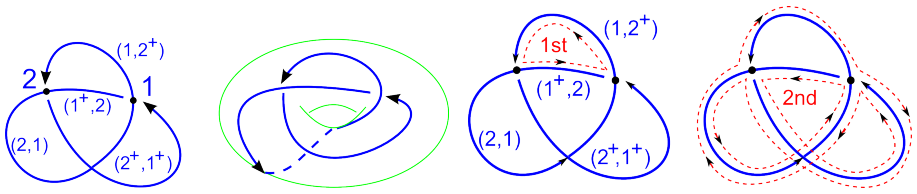

Fig. 7. The code $W = 12^+1^+2$ is realizable on a torus, $G(W)$ has two red dashed cycles.

So the 1st cycle consists of 2 edges $(12^+)_+$ *and* $(2, 1^+)_-$*. The 2nd cycle consists of 6 edges* $(1^+, 2)_+ \to (2^+, 1^+)_+ \to (1, 2)_- \to (2^+, 1)_- \to (1^+, 2^+)_- \to (2, 1)_+$*. Both cycles of* $G(W)$ *are shown by red dashed closed curves in Figure 7. The resulting Gauss surface* $S(W)$ *with 2 vertices, 4 edges, 2 faces has the Euler characteristic* $\chi = 2 - 4 + 2 = 0$ *and should be a torus as expected from the 2nd picture in Figure 7.*

The *Euler characteristic* of a surface subdivided by a graph with $|V|$ vertices and $|E|$ edges into $|F|$ faces (topological disks) is defined as $\chi = |V| - |E| + |F|$ and is invariant up to a *homeomorphism* (a bijection continuous in both directions).

Any orientable connected compact surface of a genus g (the number of handles) and b boundary components (circles) has $\chi = 2 - 2g - b \leq 2$. Hence a sphere S^2 with $\chi = 2$ is detectable by the Euler characteristic among connected compact surfaces.

Theorem 9 extends [12, Algorithm 1.4] from links to arbitrary knotted graphs.

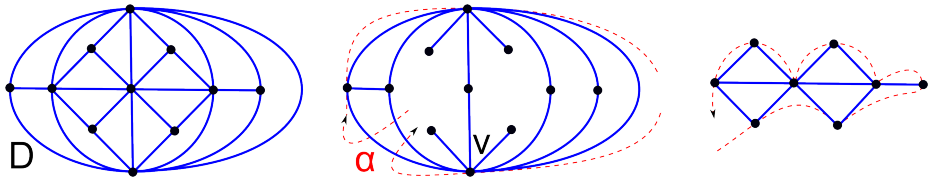

Fig. 8. A path α through all vertices of any planar graph meets every edge at most once

Theorem 9 *Given an abstract Gauss code W of a length $|W|$, an algorithm of time complexity $O(|W|)$ can determine if the given Gauss code W represents a plane diagram of a knotted graph $G \subset \mathbb{R}^3$.*

Proof. The Gauss surface $S(W)$ of any abstract Gauss code W contains the diagram D encoded by W due to the geometric interpretation of the rules in Figure 6. We assume that $S(W)$ is connected, otherwise we separately consider each connected component below. This surface has the maximum Euler characteristic χ among all orientable connected compact surfaces S that contain the diagram D and have no boundary.

Indeed, after cutting the underlying graph of the diagram $D \subset S$, the surface S splits into several components. The Euler characteristic of S is maximal when all these components are disks as in the Gauss surface. The disk has $\chi = 1$, which is maximal among all compact surfaces whose boundary is a circle.

To decide the planarity of the Gauss code W, it remains to determine if the Gauss surface $S(W)$ is a sphere S^2, which is detectable by the Euler characteristic $\chi = 2$ in the class of all orientable connected compact surfaces S without boundary. For computing the Euler characteristic χ, we use the Gauss graph $G(W)$, which splits the Gauss surface $S(W)$ into topological disks by Definition 6.

Namely, the surface $S(W)$ has $m + n$ vertices, $|W|$ edges and the number of faces equal to the number of cycles. We count all cycles in the graph $G(W)$ in time $O(|W|)$ by a double traversal of W according to the rules in Figure 6. Hence in time $O(|W|)$ we compute $\chi = m + n - |W| + \#(\text{cycles})$ and determine if the Gauss surface $S(W)$ is homeomorphic to a topological sphere S^2. □

5 Embedding any Knotted Graph into a 3-Page Book

Our algorithm will draw a 3-page embedding of a knotted graph G, which is usually represented by a plane diagram or by a Gauss code. Even for knots, an abstract Gauss code may not represent a closed curve in 3-space. That is why we first solve the planarity problem for Gauss codes of knotted graphs in Theorem 9.

If we know that a given Gauss code represents a plane diagram D of a knotted graph G, the next step in Theorem 11 is to draw the diagram D in a 2-page book as defined below. After that we upgrade this topological 2-page embedding of D to a 3-page embedding of G in linear time, see Theorem 12.

Definition 10 *The k-page* book *consists of k half-planes with a common boundary line α called the* spine *of the book. An embedding of an undirected graph G into the k-page book is* topological *if the intersection of G with the spine α is finite and includes all vertices of G. A* spine point *of the embedded graph G is any non-vertex point in the spine α. If G has no spine points, so every edge of G is contained in a single page, then the k-page book embedding of G is called* combinatorial.

A graph D is *planar* if D can be embedded in $\mathbb{R}^2$. Any undirected planar graph has a combinatorial 4-page embedding [20]. Figure 8 shows a non-hamiltonian maximal planar graph that can not be combinatorially embedded into 2 pages [1, section 5]. Any topological 2-page embedding of this graph will have spine points. The linear time algorithm below guarantees at most two spine points per edge.

Theorem 11 *[5, Theorem 1] Given a planar undirected graph $D \subset \mathbb{R}^2$ with $|V|$ vertices, an algorithm of linear time complexity $O(|V|)$ can draw a topological embedding of the graph D in the 2-page book with at most two spine points per edge.*

Two more pictures in Figure 8 illustrate the key idea how we can construct a non-self-intersecting path α that passes through each vertex once and intersects each edge at most once. By an isotopic deformation of $\mathbb{R}^2$, the path α can be converted into a straight spine, which splits the plane into 2 pages. Since all vertices and crossings of D are in the spine α, we get a required topological 2-page embedding of D.

We are not going to minimize the number of bends of edges in a 2-page embedding of a plane diagram D, because we shall construct 3-page embeddings of original knotted graphs with a linear number $O(|W|)$ of total bends in the length of a Gauss code W.

Theorem 12 *Given an abstract Gauss code W, an algorithm of time complexity $O(|W|)$ determines if W represents a plane diagram of a knotted graph $G \subset \mathbb{R}^3$ and then draws a topological 3-page embedding of a graph H isotopic to G. Moreover, the graph H has at most $12|W|$ intersections with the spine of the book.*

Proof. We first apply the linear time algorithm from Theorem 9 to determine if the code W represents a plane diagram D of a knotted graph G. If yes, we draw a 2-page embedding of the diagram $D \subset \mathbb{R}^2$ in linear time using the algorithm of Theorem 11.

At every crossing in the diagram D, we mark a short red arc that crosses over another arc in D. The centers of all these marked arcs are all crossings of D, which are already in the straight spine α of the 2-page book. We may slightly deform the embedding of D by pushing the marked red arcs into the spine α. The full list of local upgrades of crossings has only 10 types in Table 1 justified by Lemma 20 in Appendix D.

Now we push all marked red arcs into the extra 3rd page attached along α above the diagram D. So we have upgraded the 2-page embedding of D to a 3-page embedding of a knotted graph H isotopic to the original graph G, see Fig. 9 and 10.

We need a constant time per crossing, so $O(|W|)$ in total, for a 3-page embedding of H. Since the diagram D has $|W|$ edges, the 2-page embedding of D with at most 2 spine points per edge has at most $3|W|$ points in the spine α. Each crossing of D is

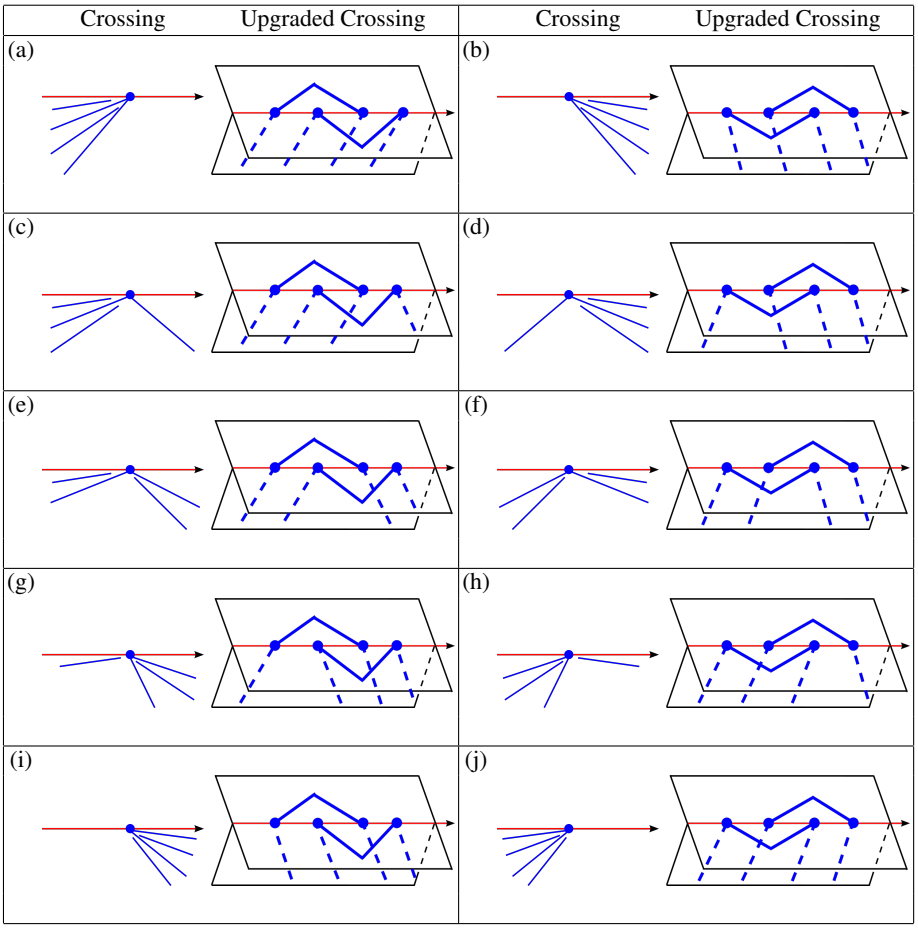

Table 1. Necessary local upgrades of crossings from 2 to 3 pages, see Lemma 20 in Appendix D.

replaced by at most 4 intersections with the spine α in a 3-page embedding of H. The total number of points in the intersection of H and the spine α is at most $12|W|$. □

The codes of 3-page embeddings in Fig. 9 and 10 are explained in Appendix A.

6 Discussion and 10 Open Problems on Knotted Graphs

We now discuss our results in the light of a huge gap between real-life experiments and pure mathematics. Experimental data are usually given in the form of unstructured and noisy clouds of points. If we have only 2D images as in Figure 1, then we also need to extract a knotted structure in a suitable form.

Pure mathematicians have developed deep theories how to classify complicated geometric objects including knots. However, all mathematical algorithms start from ideal

models, say a closed curve given by continuous functions or a polygonal curve given by a sequence of points connected by straight edges.

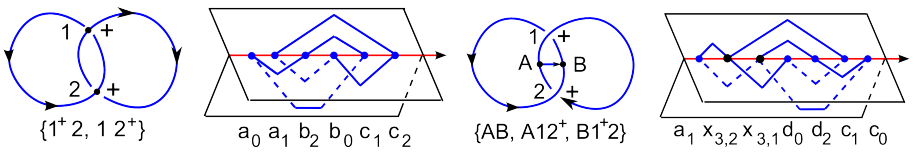

Fig. 9. Hopf link and Hopf graph with Gauss codes and 3-page embeddings

The key challenge is to convert any unstructured experimental data into an ideal theoretical model that can be rigorously analyzed by existing mathematical methods. The first advance in this direction is computing the fundamental group of a knot complement from a point cloud in [4]. We state open problems relating practice and theory for knotted graphs. We are open to collaboration on these and any related projects.

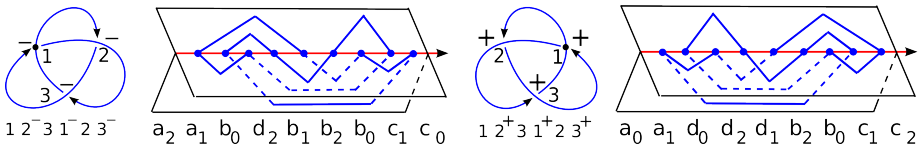

Fig. 10. Left-handed and right-handed trefoils with Gauss codes and 3-page embeddings

1. State and prove a criterion of planarity of Gauss codes of knotted graphs using combinatorial invariants like sums of signs similarly to [12, Theorem 3.6].

2. Let a link of n components be given as an unordered union of $m \geq 2n$ open arcs (or sequences of points). How can we 'correctly' join corresponding endpoints of the arcs to form n closed curves in $\mathbb{R}^3$?

3. When drawing pictures on a tablet, a few intersecting curves can be represented by several sequences of 2D points sampled along the curves. Under what conditions on the curves and sample, can we quickly reconstruct the curves using only the sample?

4. Design a fast algorithm to convert an unstructured 3D point cloud sampled around an unknown knotted structure into a Gauss code W of a knotted graph.

5. Design an algorithm to convert a 2D image of a knotted graph into a Gauss code W.

Our current work on visualizing Gauss codes is an important step in the hard problems above. First, we may try to recognize small patches of vertices and crossings in a 2D image of a knotted graph, but after that we should combine them in a Gauss code whose planarity can be quickly checked by Theorem 9.

Second, if we need to visualize any noisy cloud sampled from an unknown knot $K \subset \mathbb{R}^3$, we may draw a knot isotopic to K using its Gauss code and Theorem 12. Even more importantly we often wish to get a simplified (minimal) version of a knot.

The state-of-the-art simplification algorithm for recognizing trivial knots available at http://www.javaview.de/services/knots is based on 3-page embeddings. We remind how to extend this approach to graphs in Appendix A and state more problems below.

6. Design an algorithm to untangle diagrams of graphs isotopic to planar graphs.

7. Extend the algorithm for drawing knotted graphs in 3 pages to drawing 2-dimensional surfaces in a universal 3-dimensional polyhedron (the *hexabasic book*) from [9].

8. Decide if the problem to find a 3-page embedding of a knotted graph $G \subset \mathbb{R}^3$ having the minimum number of intersections with the spine α is NP-hard.

9. Use the computed invariants to build a database of isotopy classes of knotted graphs similarly to the Knot Atlas at http://katlas.math.toronto.edu.

10. Define a kernel [16] on point clouds representing knotted graphs so that one can use tools of machine learning for automatic recognition of real-life knotted structures.

The earlier version [13] of this paper had other Problems 1 and 8 about knotted graphs $G \subset \mathbb{R}^3$ given as sequences of points, say positions of atoms in a protein backbone. These problems were solved in [14] and replaced above by harder questions.

Algorithms from Theorems 9, 11 and 12 are described in Appendices B, C, D, respectively. A C++ code will be on the webpage http://kurlin.org of the first author, who thanks EPSRC for funding his secondment at Microsoft Research Cambridge. More examples are included in the forthcoming MSc thesis [17] of the second author.

References

1. Bernhart, F., Kainen, P. (1979) The book thickness of a graph. J. Combinatorial Theory B, v. 27, p. 320-331.
2. Biasotti, S., Giorgi, D., Spagnuolo, Falcidieno, M. (2008) Reeb graphs for shape analysis and applications. Theoretical Computer Science, v. 392, p. 5-22.
3. Boost C++ Libraries (version 1.59.0), http://www.boost.org.
4. Brendel, P., Dlotko, P., Ellis, G., Juda, M., Mrozek, M. (2015) Computing fundamental groups from point clouds. Appl. Algebra in Engineering, Communication and Comp., v. 26, p. 27-48.
5. Di Giacomo, E., Didimo, W., Liotta, G., Wismath, S. (2005) Curve-constrained drawings of planar graphs. Computational Geometry, v. 30, p. 1-23.
6. Enomoto, H., Miyauchi, M. (1999) Lower bounds for the number of edge-crossings over the spine in a topological book embedding of a graph. SIAM J. Discrete Math., v. 12, p. 337-341.
7. De Fraysseix, H. and Pach, J. and Pollack, R. (1990) How to draw a planar graph on a grid. Combinatorica, v. 10, no. 1, p. 41-51.
8. Kauffman, L. (1989) Invariants of graphs in three-space. Trans. AMS, v. 311, p. 697-710.
9. Kearton, C., Kurlin, V. (2008) All 2-dimensional links live inside a universal 3-dimensional polyhedron. Algebraic and Geometric Topology, v. 8 (2008), no. 3, p. 1223-1247.
10. Kurlin, V. (2001) Dynnikov three-page diagrams of spatial 3-valent graphs. Functional Analysis and Its Applications, v. 35 (2001), no. 3, p. 230-233.

11. Kurlin, V. (2007) Three-page encoding and complexity theory for spatial graphs. J. Knot Theory Ramifications, v. 16, no. 1, p. 59-102.
12. Kurlin, V. (2008) Gauss paragraphs of classical links and a characterization of virtual link groups. Mathematical Proceedings of Cambridge Phil. Society, v. 145 , no. 1, p. 129–140.
13. Kurlin, V. (2015) A linear time algorithm for visualizing knotted structures in 3 pages. Proceedings of IVAPP 2015: Information Visualization Theory and Applications, p. 5–16.
14. Kurlin, V. (2015) Computing invariants of knotted graphs given by sequences of points in 3-dimensional space. Proceedings of TopoInVis 2015: Topology-Based Methods in Visualization.
15. Kurlin, V., Vershinin, V. (2004) Three-page embeddings of singular knots. Functional Analysis and Its Applications, v. 38, no. 1, p. 14-27.
16. Schölkopf, B. and Smola, A. (2002) Learning with kernels. MIT Press, Cambridge, MA.
17. C. Smithers. (2015) A Linear Time Algorithm for Embedding Arbitrary Knotted Graphs into a 3-Page Book. MSc thesis, Durham University, UK.
18. Tkalec, U., Ravnik, M., Copar, S., Zumer, S., Musevic, I. (2011) Reconfigurable Knots and Links in Chiral Nematic Colloids. Science, v. 333, no. 6038, p. 62–65,
19. Whitney, H. (1932) Congruent graphs and the connectivity of graphs. American Journal of Mathematics, v. 54, no. 1, p. 150-168.
20. Yannakakis, M. (1989) Embedding planar graphs in four pages. J. Comp. System Sciences, v. 38, p. 36–67.

Appendix A: semigroups for classifying graphs up to isotopy

We remind how to encode 3-page embeddings of all knotted graphs by words in a simple alphabet. Since edges with vertices of degree 1 can be easily unknotted by isotopy in 3-space, for simplicity we consider below only graphs without degree 1 vertices.

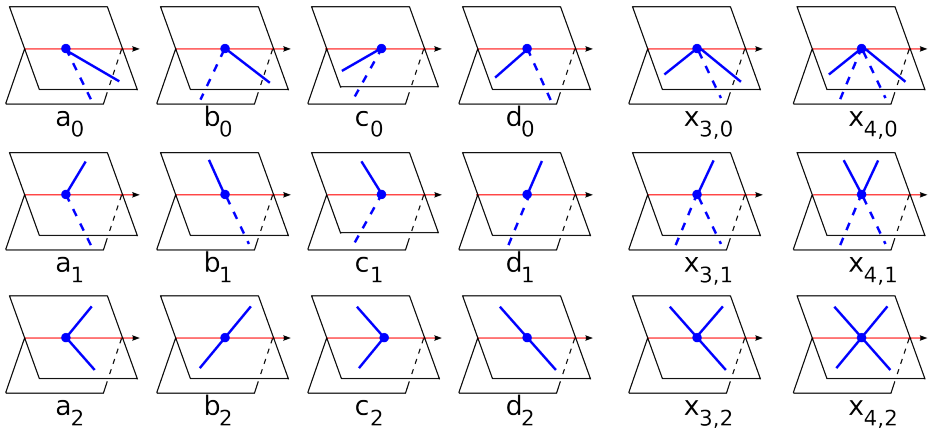

Fig. 11. Local 3-page embeddings for the generators of the semigroups from Theorem 14.

To explain the 3-page encoding of knotted graphs, let us deform any 3-page embedding so that all arcs are monotonically projected to the spine α. Then the 3-page embedding can be uniquely reconstructed from its thin neighborhood around α.

Namely, if we know only directions of arcs going from all spine intersections, we can uniquely join these arcs in each of 3 pages. Hence we can encode any 3-page embedding by the ordered list of local embeddings at all intersections in the spine α.

Theorem 13 *[11, Theorem 1.6a] Any 3-page embedding of a knotted graph G with vertices up to degree n can be encoded by a word in the alphabet consisting of the letters a_i, b_i, c_i, d_i and $x_{k,i}$ for each degree $k = 3, \dots, n$, where $i = 0, 1, 2$, see Fig. 11.*

Fig. 11 shows 12 local embeddings a_i, b_i, c_i, d_i, $i \in \mathbb{Z}_3 = \{0, 1, 2\}$, which are sufficient for encoding 3-page embeddings of knots and links. The notation a_i emphasizes that all a_i can be obtained from each other by a rotation around the spine α.

For encoding graphs in Theorem 13, we can make sure that at each vertex the spine separates one or two arcs from others.Then only 3 local embeddings are enough for each degree, see 3 neighborhoods $x_{3,0}$, $x_{3,1}$, $x_{3,2}$ of a degree 3 vertex in Fig. 11.

The 3-page embedding $K_5 \subset \mathbb{R}^3$ in Fig. 3 can be represented by the 3-page code $w = a_1 d_1\ (a_1 b_1 x_{4,1})^2 (a_1 d_1 x_{4,1}) d_1 (x_{4,1} d_1 c_1)(x_{4,1} b_1 c_1) d_2 c_1 c_2$.

The following result completely reduces the topological classification of knotted graphs up to isotopy in $\mathbb{R}^3$ to a word problem in finitely presented semigroups. The cases of 3-regular graphs and 4-regular graphs (singular knots) appeared in [10], [15].

Theorem 14 *[11, Theorems 1.6 and 1.7] There is a finitely presented semigroup whose all central elements are in a 1-1 correspondence with all isotopy classes of knotted graphs with vertices of degree up to n. An algorithm of a linear complexity $O(|w|)$ decides if an element w of the semigroup is central, i.e. commutes with all other elements.*

So two knotted graphs $G, H \subset \mathbb{R}^3$ are isotopic in 3-space if and only if their corresponding central elements w_G, w_H are equal in the semigroup. A stronger result in [9] says that all isotopies between 3-page embeddings of arbitrary knotted graphs are realizable in the *hexabasic book* $U \times [0, 1]$, where U is the union of the 3-page book $\mathbb{P}_0 \cup \mathbb{P}_1 \cup \mathbb{P}_2$ (with the common boundary line α) and a plane $\mathbb{P}_3$ orthogonal to α. Theorem 14 has been extended to the isotopy classification of surfaces in $\mathbb{R}^4$ [9].

There are two semigroups: RSG_n for rigid knotted graphs with vertices up to degree n and NSG_n for non-rigid graphs. Both semigroups have 12 generators a_i, b_i, c_i, d_i, $i \in \{0, 1, 2\}$, and $3(n-2)$ generators for vertices up to degree n, so 3 generators for each degree from 3 to n, see Fig. 11. The operation in the semigroups is the concatenation of words. The unit is the empty word $\emptyset$. The generators $a_i, c_i, x_{k,i}$ are not invertible, while b_i, d_i are inverses of each other. In the case of links for $n = 2$, the semigroup has 48 relations (1)–(4), where the index $i \in \mathbb{Z}_3 = \{0, 1, 2\}$ is considered modulo 3.

(1) $d_0 d_1 d_2 = 1$ and $b_i d_i = 1 = d_i b_i$;

(2) $a_i = a_{i+1} d_{i-1}$, $\quad b_i = a_{i-1} c_{i+1}$, $c_i = b_{i-1} c_{i+1}$, $\quad d_i = a_{i+1} c_{i-1}$;

(3) $w(d_i c_i) = (d_i c_i) w$ for $w \in \{\ c_{i+1},\ b_i d_{i+1} d_i\ \}$;

(4) $uv = vu$, where $u \in \{\ a_i b_i,\ b_{i-1} d_i d_{i-1} b_i\ \}$, $v \in \{\ a_{i+1},\ b_{i+1},\ c_{i+1},\ b_i d_{i+1} d_i\ \}$.

One of the 7 relations in (1) is superfluous as it follows from the remaining 6. The generators a_i, b_i, c_i, d_2 can be expressed only in terms of d_0, d_1, but the resulting relations between d_0, d_1 will be longer. All defining relations of the semigroups represent elementary isotopies between 3-page embeddings, see [13, Appendix].

For knotted graphs with vertices of only degree 3, any non-rigid isotopy can be made rigid, because we can keep 3 short arcs at any vertex in a moving plane. Hence both semigroups for rigid and non-rigid isotopies from Theorem 14 are the same for $n = 3$. In this case the extra relations in addition to (1)–(4) are (5)–(9), see [10], [11]:

(5) $x_{3,i-1} = d_{i-1}x_{3,i}d_{i+1}$;

(6) $x_{3,i}b_i(d_i^2d_{i+1}^2d_{i-1}^2) = (d_id_{i+1}d_{i-1})x_{3,i}b_i$;

(7) $x_{3,i}d_i = a_i(x_{3,i}d_i)c_i, \quad b_ix_{3,i}b_i = a_i(b_ix_{3,i}b_i)c_i$;

(8) $ux_{3,i+1} = x_{3,i+1}u$ for any word u from $\{\ a_ib_i,\ d_ic_i,\ x_{3,i}b_i,\ b_{i-1}d_id_{i-1}b_i\ \}$;

(9) $(x_{3,i}b_i)v = v(x_{3,i}b_i)v$ for any word v from $\{\ a_{i+1},\ b_{i+1},\ c_{i+1},\ b_id_{i+1}d_i\ \}$.

Knotted graphs that have only vertices of degree 4 and are considered up to rigid isotopy are often called *singular knots*. Each singular point remains a transversal intersection of two arcs during a rigid isotopy, so the cyclic order of all arcs at any degree 4 vertex is invariant. The semigroup of Theorem 14 for singular knots has 15 generators $a_i, b_i, c_i, d_i, x_{4,i}$, relations (1)–(4) above and relations (10)–(14) below, see [11, 15]:

(10) $x_{4,i-1} = b_{i+1}x_{4,i}d_{i+1}$;

(11) $(d_ix_{4,i}b_i)(d_i^2d_{i+1}^2d_{i-1}^2) = (d_i^2d_{i+1}^2d_{i-1}^2)(d_ix_{4,i}b_i)$;

(12) $d_ix_{4,i}d_i = a_i(d_ix_{4,i}d_i)c_i,\ b_ix_{4,i}b_i = a_i(b_ix_{4,i}b_i)c_i$;

(13) $wx_{4,i+1} = x_{4,i+1}w$ for any word w from $\{\ a_ib_i,\ d_ic_i,\ d_ix_{4,i}b_i,\ b_{i-1}d_id_{i-1}b_i\ \}$;

(14) $(d_ix_{4,i}b_i)v = v(d_ix_{4,i}b_i)v$ for any word v from $\{\ a_{i+1},\ b_{i+1},\ c_{i+1},\ b_id_{i+1}d_i\ \}$.

The hard part of Theorem 14 says that any isotopy between graphs decomposes into finitely many elementary isotopies involving *a small part* of a 3-page code. This is the main advantage of the 3-page encoding over plane diagrams and Gauss codes. Indeed, Reidemeister moves in Fig. 4 and their analogues on Gauss codes are not local.

The linear time algorithm for detecting a central element w checks if the arcs corresponding to all letters of w properly meet each other in every page to form an embedding of a graph without hanging edges. For example, the letter a_2 doesn't encode any knotted graph, but a_2c_2 does, because the arcs of a_2, c_2 meet and form a closed curve.

The 3-page code of a knotted graph commutes with any other element w in the semigroups from Theorem 14. For instance, a trivial knot has the code a_2c_2 and can be isotopically moved in $\mathbb{R}^3$ to another side of the 3-page embedding represented by w.

Appendix B: Algorithm 1 for checking planarity of any Gauss code

The input is an abstract Gauss code W from Definition 5. The output is a plane diagram (if it exists) having the same Gauss code W. The plane diagram will be obtained as the Gauss graph $G(W)$ with topological disks attached to certain cycles of $G(W)$.

Stage 1.1: simplifying a Gauss code W by Reidemeister moves I, see Fig. 4.

We go along a Gauss code W and check all pairs of two successive symbols. If the pair is one of $(k, k^+), (k, k^-), (k^+, k), (k^-, k)$ corresponding to the same k-th crossing, we remove this pair from W and continue from the symbol before the removed pair. If W is cyclic (for a circle), at the end we compare the 1st and last symbols of W.

Stage 1.2: building the abstract Gauss graph $G(W)$ of a Gauss code W.

The Gauss graph $G(W)$ was introduced in Definition 6. The nodes of $G(W)$ are all different vertices and crossings extract from the Gauss code W after forgetting all signs. We store the graph $G(W)$ in memory by using three arrays NodeTypes, EdgeList, WedgeList. NodeTypes[r] is 0 if r is a vertex, 1 if r is a positive crossing, -1 if r is a negative crossing. The EdgeList consists of ordered pairs (i, j), where i, j are indices of nodes in NodeTypes. The WedgeList for each node n contains indices of edges attached to n and ordered according to the cyclic order from W (starting from any edge).

Stage 1.3: subdividing the Gauss graph $G(W)$ to remove multiple edges.

It will be convenient to avoid multiple edges of $G(W)$ for Algorithm 2 building a 2-page embedding. For each node n, we check all attached edges. If we find two edges with the same endpoint $k \neq n$, we add a midpoint to one of these edges.

Stage 1.4: splitting the Gauss graph into different connected components.

We split the subdivided Gauss graph into connected components by using the Boost algorithm [3] based on a Breadth First Search. From now on we assume that $G(W)$ is a connected graph without loops and double edges.

Stage 1.5: finding cycles in the Gauss graph $G(W)$ and checking planarity.

We initialise the boolean $PassList$ whose 2 halves can be viewed as two (forward and backward) lists indexed by edges of $G(W)$. Every bit in each of the halves indicates whether we have passed the corresponding edge in the forward or backward direction whilst building cycles. Each entry in $PassList$ is initially false. We shall keep track of the index $least$ in this list so that all edges with indices less than $least$ are passed.

Step 1. Starting at the index $least$, check each entry of the edge list until a false entry is found. If we reach the end of the list, we have found all cycles.

Step 2. Change $least$ to the found index i of the next edge that wasn't passed yet. Set $PassList[i] = true$ and start a new cycle from this i-th oriented edge e. Repeat the following substeps until the next passed edge is once again e.

2a. For each edge, use $EdgeList$, $WedgeList$ to find the node being moved towards.

2b. To make the "left-turn" for traversing a cycle from Definition 6 and Fig. 6, we take the next edge from the list of cyclically ordered edges at the node from the substep 2a.

2c. Change the boolean entry in $PassList$ for the new passed edge to true.

Step 3. Store the cyclic order of the edges in the found cycle, and move back to Step 1.

Step 4. Compute the Euler characteristic of the Gauss surface $S(W)$ by the formula $\chi = V - E + F$. Here V, E are the numbers of nodes and edges, respectively, in $G(W)$. The number of cycles (or disks attached to the graph) is denoted by F.

Step 5. If $\chi \neq 2$, the given Gauss code W doesn't correspond to any knotted graph as shown in the proof of Theorem 9. Otherwise, we output an embedding $G(W) \subset S^2$ as the graph $G(W)$ with its $NodeTypes$, $EdgeList$, $PassList$ and all found cycles.

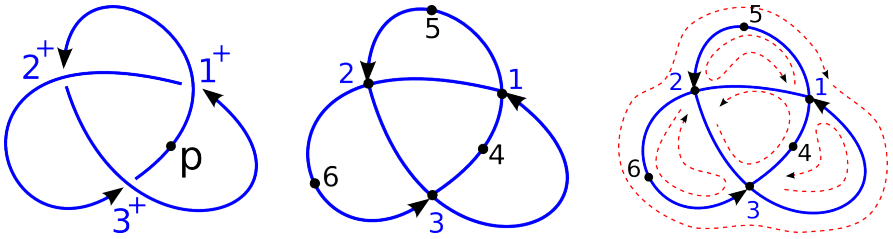

Fig. 12. Left: trefoil with signs of crossings and a base point p for $W_{Tref} = 12^+31^+23^+$. **Middle**: subdivided Gauss graph without double edges from Stage 1.3 of Algorithm 1. **Right**: 5 cycles found in the subdivided graph at Stage 1.5 of Algorithm 1, so $\chi = 6 - 9 + 5 = 2$.

Appendix C: Algorithm 2 for drawing a 2-page embedding

The input is an embedding $G \subset S^2$ of an abstract graph G with the boundary cycles of all faces (connected components of $S^2 - G$). The output is a topological 2-page embedding $G \subset \mathbb{P}_0 \cup \mathbb{P}_1$ (of a graph isotopic to the given one in S^2). Here the half-planes $\mathbb{P}_0$ and $\mathbb{P}_1$ have the common boundary (spine) α containing all nodes of G.

Stage 2.1: extending a given graph G to a maximal planar graph $\bar{G}$.

We triangulate each face whose boundary has more than 3 nodes as follows.

Step 1. Pick a node v and use $WedgeList$, $EdgeList$ to find its neighbors u, w.

Step 2. If u, w are connected by an edge (outside the current face), v cannot be connected to any other node $x \neq u, v, w$ of the face, so we add all such edges (v, x). Otherwise, we add the edge (u, w) cutting the triangle (u, v, w) from the current face.

Step 3. Start again with a new face if it still has more than 3 nodes.

For the embedded Gauss graph of the trefoil code W_{Tref}, the only non-triangular face in Fig. 12 is $\{5, 1, 4, 2, 6, 3\}$. To triangulate this face, we begin with node 1, and check if its neighbors in the face are adjacent in the graph. Node 1 has neighbours 4 and 5 in the face, which a quick check revels is not an edge in our list and so we add edge (4,5), and create new cycles $\{5, 1, 4\}$ and $\{4, 2, 6, 3, 5\}$. Similarly edges (4,6) and (6,5) are added, in order to get a triangulation from the graph $\bar{G}(W_{Tref})$.

Stage 2.2: building a canonical ordering on the nodes of the maximal planar graph $\bar{G}$

A graph G is k-*connected* if G has at least $k+1$ nodes and the removal of any $k-1$ or fewer nodes with all their incident edges keeps the graph connected.

Definition 15 *Given a maximal planar graph $G \in \mathbb{R}^2$ on $n \geq 3$ nodes, an ordering of the nodes $v_1, v_2, \ldots, v_n$ of G is called* canonical *if for each $3 \leq k < n$ the subgraph G_k induced by the nodes $v_1, \ldots, v_k$ has the following properties.*

(a) The subgraph G_k is 2-connected and its external face F_k has the edge v_1v_2.

(b) The neighbors of v_{k+1} in G_k form a path on the boundary of the face F_k.

A canonical ordering exists by [7, Proposition 3] and is implemented in [3].

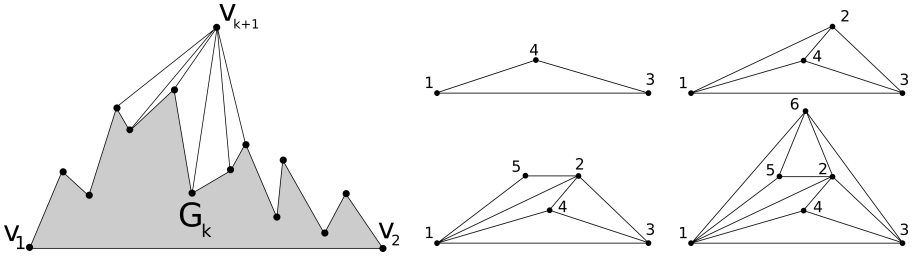

Fig. 13. Left: illustration of Definition 15. **Right**: drawing $\bar{G}(W_{Tref})$ using a canonical ordering.

Stage 2.3: drawing a 2-page embedding of the maximal planar graph $\bar{G}$

We implemented the algorithm from [5] drawing a topological 2-page embedding $G \subset \mathbb{P}_0 \cup \mathbb{P}_1$ of any maximal planar graph whose nodes $v_1, \ldots, v_n$ are added to the spine according to their canonical ordering. We put the edge between the first 2 nodes v_1, v_2 into the lower page $\mathbb{P}_1$. For each next node v_k, we follow the steps below.

Step 1. Find the embedded neighbors $w_1, \ldots, w_l$ of v_k, where w_1 is the leftmost in α.

Step 2. Embed v_k into α just to the right of w_1, and embed the edge (v_k, w_1) in $\mathbb{P}_1$.

Step 3. We connect v_k with its neighbour w_2 according to the substeps below.

3a. If v_k and w_2 are consecutive in the spine, we embed the edge (v_k, w_2) in $\mathbb{P}_1$.

3b. If v_k, w_2 are not consecutive, we embed (v_k, w_2) with 2 extra spine intersections p, q, namely (v_k, w_2) splits into 3 subedges $(v_k, p) \subset \mathbb{P}_1$, $(p, q) \subset \mathbb{P}_0$, $(q, w_2) \subset \mathbb{P}_1$.

Step 4. For each neighbor w_i, $3 \leq i \leq l$, we embed the edge (v_k, w_i) as in substep *3b*.

Step 5. Update $EdgeList$ when we subdivide an edge into 3 subedges in *Steps 3b, 4*.

We explain the steps above by drawing the trefoil graph $\bar{G}(W_{Tref})$ from the last picture in Fig. 14. This graph has a canonical ordering of nodes $\{1, 3, 4, 2, 5, 6\}$. We embed the node $v_3 = 4$ between the first 2 nodes $v_1 = 1$ and $v_2 = 3$ by Steps 2 and 3a in the 2nd picture of Figure 14. We embed the node $v_4 = 2$ with 3 already embedded neighbors $1, 4, 3$ by Steps 2, 3, 4 above in the 3rd picture of Figure 14.

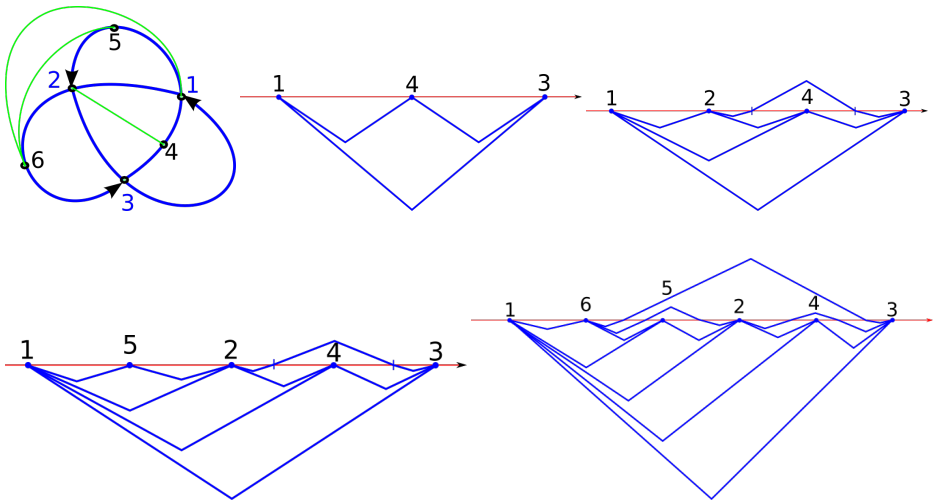

Fig. 14. A maximal planar graph $\bar{G}(W_{Tref})$ and drawing its 2-page embedding in Stage 2.3.

We introduce the dual graph below in order to show in Proposition 19 that the above algorithm produces an embedding isotopic to an original maximal planar graph.

Definition 16 *Given a planar graph $G \subset \mathbb{R}^2$, the* dual graph $G^* \subset \mathbb{R}^2$ *has a node for each face of G and a continuous arc connecting nodes from any adjacent faces of G.*

Theorem 17 *(Whitney [19, Theorem 11]) A 3-connected planar graph G with a fixed external face has a unique dual graph $G^* \subset \mathbb{R}^2$ up to isotopy in the plane.*

Corollary 18 *Any 3-connected planar graph without loops and multiple edges has a unique planar embedding (up to isotopy) with a choice of external face of G*

Proof. Let $G_1, G_2 \subset \mathbb{R}^2$ be planar embeddings of the same abstract graph with the same external face. By Theorem 17, the duals ${G_1}^*$ and ${G_2}^*$ are isotopic. By Definition 16 the double dual ${G_1}^{**}$ is isotopic to G_1 and ${G_2}^{**}$ is isotopic to G_2. Since ${G_1}^* \sim {G_2}^*$, we conclude that ${G_1}^{**} \sim {G_2}^{**}$, hence $G_1 \sim {G_1}^{**} \sim {G_2}^{**} \sim G_2$.

Proposition 19 *For any maximal planar graph $\bar{G} \subset \mathbb{R}^2$ with a fixed external face, Stage 2.3 outputs a 2-page embedding isotopic to the original embedding $\bar{G} \subset \mathbb{R}^2$.*

Proof. We may assume that a given maximal planar graph G has at least 4 nodes, then G is 3-connected. Since G has a fixed external face, no loops or double edges, G has a unique planar embedding by Corollary 18. Since the 2-page embedding from Stage 2.3 has the same face, the output is isotopic to the original embedding of G. □

Stage 2.4: restricting a 2-page embedding of a maximal planar graph $\bar{G}$ to G

At Stage 2.1 we extended a planar graph G to a maximal planar graph $\bar{G}$ in order to use a canonical ordering for drawing a 2-page embedding. Erase the edges added at Stage 2.1 to get a 2-page embedding of G, see the 1st picture in Fig. 15.

Appendix D: Algorithm 3 for drawing a 3-page embedding

Algorithm 1 in Appendix B starts from a Gauss code W and outputs an embedded (possibly, subdivided) Gauss graph $G(W) \subset S^2$ (if it exists). Then we fix an external face to get an embedding $G(W) \subset \mathbb{R}^2$. Algorithm 2 in Appendix C outputs an isotopic 2-page embedding $G(W) \subset \mathbb{P}_0 \cup \mathbb{P}_1$. In Algorithm 3 the input is the 2-page embedding with signs of crossings coming from the Gauss code W. The output will be a 3-page embedding $K(W) \subset \mathbb{P}_0 \cup \mathbb{P}_1 \cup \mathbb{P}_2$ of a knotted graph isotopic to a graph given by W.

Stage 3.1: upgrading crossings in a 2-page embedding to get a 3-page embedding.

Lemma 20 *There are exactly 10 types of crossings in $\mathbb{P}_0 \cup \mathbb{P}_1$ obtained by Algorithm 2.*

Proof. According to the steps of Stage 2.3, all 4 (sub)edges incident to each each crossing are in the lower page $\mathbb{P}_1$. Each of these 4 edges goes either to the left or to the right of the crossing. There are 5 ways to split 4 edges into 2 groups of left and right edges. For each of 5 ways, there are 2 types of crossings, see all 10 types are in Table 1. □

We upgrade each crossing in a 2-page embedding $G(W) \subset \mathbb{P}_0 \cup \mathbb{P}_1$ to a local 3-page embedding according to Table 1. The resulting global embedding defines a knotted graph $K(W) \subset \mathbb{P}_0 \cup \mathbb{P}_1 \cup \mathbb{P}_2$, because in every page all arcs join each other.

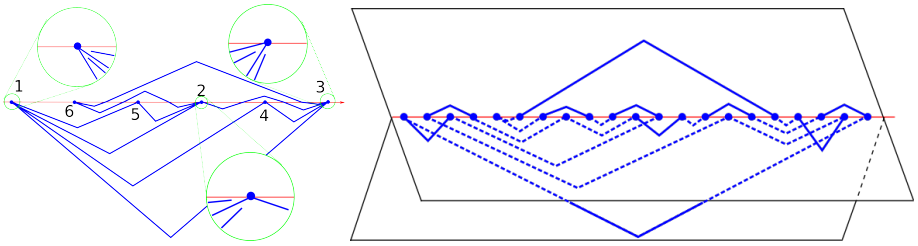

Fig. 15. Left: 2-page embedding of the Gauss graph $G(W_{Tref})$ from Stage 2.4 with resolved crossings for Stage 3.1. **Right**: 3-page embedding of the trefoil $G(W_{Tref})$ obtained at Stage 3.1.

Stage 3.2: computing the element of a knotted graph $K(W)$ in a 3-page semigroup.

The 3-page embedding obtained at Stage 3.1 may contain spine intersections with both arcs in the same page. The 3-page alphabet in Fig. 11 has only local embedding where arcs around every point in the spine occupy exactly 2 pages. Examples in the left hand side picture of Fig. 18 show how to get a 3-page embedding encoded by an element in a semigroup from Appendix A. After these upgrades the trefoil $K(W_{Tref})$ from Fig. 15 has the 3-page code $a_0a_1d_0b_1a_1b_1d_1d_1d_1b_1b_1d_1b_0c_1d_0d_1d_1b_1b_1b_1b_0d_1c_0c_1$.

Stage 3.3: local simplifications for shortening elements in a 3-page semigroup.

Relations (1)–(2) in the semigroups from Appendix A are illustrated in Fig. 17 and allow us to locally simplify the element obtained at Stage 3.2. The trefoil from

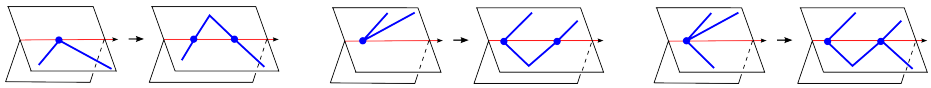

Fig. 16. Normalizations of 3-page embeddings around points with arcs in the same page.

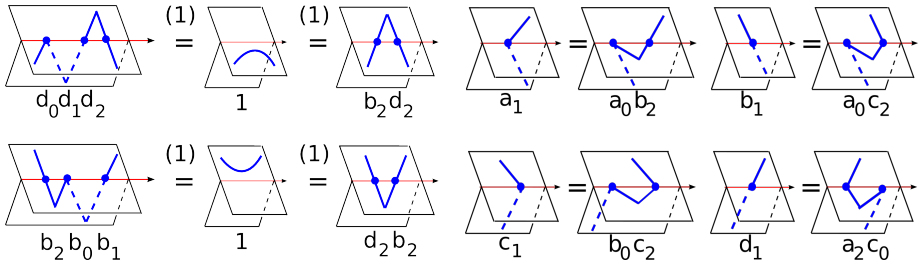

Fig. 17. A geometric interpretations of relations (1) and (2) in the semigroups from Appendix A.

Fig. 15 will have the shorter code $a_0a_1d_0b_1a_1d_1b_0c_1d_0d_2d_1c_0c_1$ in the 1st picture of Fig. 18. The 2nd picture shows a minimal 3-page embedding with a shortest code $a_2a_1d_0d_2d_1d_0d_2c_1c_0$ by using more relations in the semigroups from Appendix A.

Stage 3.4: computing 3D coordinates for a straight-line 3-page embedding of $K(W)$.

The spine α is identified with the x-axis. The upper page $\mathbb{P}_0$ is in the (x,z)-plane. The pages $\mathbb{P}_1, \mathbb{P}_2$ are obtained from $\mathbb{P}_0$ by the rotations through the angles $\pm\frac{2\pi}{3}$. If a 3-page code of a knotted graph $K(W) \subset \mathbb{P}_0 \cup \mathbb{P}_1 \cup \mathbb{P}_2$ has n letters, we embed corresponding points in the spine at $(j,0,0)$, $1 \le j \le n$. After finding which points are connected in each page $\mathbb{P}_i$, we embed all edges as broken lines with 2 straight segments.

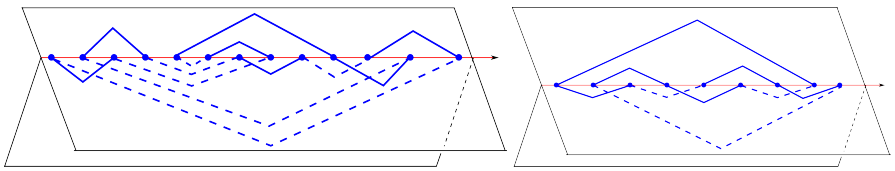

Fig. 18. Left: the 3-page embedding of a trefoil after local simplifications at Stage 3.3. **Right**: the minimal 3-page embedding of a trefoil after global simplifications by relations from Appendix A.

The MSc thesis [17] contains more details why all stages in Appendices B, C, D require a linear time and memory in the length of a Gauss code.