Recognizing String Graphs in NP.

Marcus Schaefer[a] Eric Sedgwick[a] Daniel Štefankovič[b]

[a]*Department of Computer Science*
DePaul University
243 South Wabash
Chicago, Illinois 60604, USA

[b]*Department of Computer Science*
University of Chicago
1100 East 58th Street
Chicago, Illinois 60637, USA

Abstract

A *string graph* is the intersection graph of a set of curves in the plane. Each curve is represented by a vertex, and an edge between two vertices means that the corresponding curves intersect. We show that string graphs can be recognized in **NP**. The recognition problem was not known to be decidable until very recently, when two independent papers established exponential upper bounds on the number of intersections needed to realize a string graph (Pach and Tóth, 2001; Schaefer and Štefankovič, 2001). These results implied that the recognition problem lies in **NEXP**. In the present paper we improve this by showing that the recognition problem for string graphs is in **NP**, and therefore **NP**-complete, since Kratochvíl showed that the recognition problem is **NP**-hard (Kratochvíl, 1991b). The result has consequences for the computational complexity of problems in graph drawing, and topological inference. We also show that the string graph problem is decidable for surfaces of arbitrary genus.

Key words: String graphs, **NP**-completeness, graph drawing, topological inference, Euler diagrams
2000 MSC: 05C62, 68Q17, 05C10

Email addresses: `mschaefer@cs.depaul.edu` (Marcus Schaefer), `esedgwick@cs.depaul.edu` (Eric Sedgwick), `stefanko@cs.uchicago.edu` (Daniel Štefankovič).

1 Strings, Drawings, and Diagrams

A *string graph* is the intersection graph of a set of curves in the plane. A *(Jordan) curve*, or *string*, is a set homeomorphic to $[0, 1]$. By this definition, curves are compact sets which do not self-intersect. Given a collection of curves $(C_i)_{i\in I}$ in the plane, the corresponding intersection graph is $(I, \{\{i, j\} : C_i \text{ and } C_j \text{ intersect}\})$. The *size* of a collection of curves is the number of intersection points (we assume that no three curves intersect in the same point). A graph isomorphic to the intersection graph of a collection of curves in the plane is called a *string graph*.

The string graph problem asks whether string graphs can be recognized. The problem made its first explicit appearance in a 1966 paper by Sinden on circuit layout (Sinden, 1966), although a similar question had been suggested earlier by Benzer on genetic structures (Benzer, 1959). The string graph problem was introduced to the combinatorial community by Ron Graham in 1976 (Graham, 1976).

From a combinatorial point of view we are interested in $c_s(G)$, the smallest number of intersections of a set of curves necessary to realize a string graph G; $c_s(G)$ is finite for string graphs G (as we will show later). For graphs G that are not string graphs, we let $c_s(G)$ be infinity. Define $c_s(n) = \max\{c_s(G) : G \text{ is a string graph on } n \text{ vertices}\}$. A computable upper bound on $c_s(n)$ implies decidability of the string graph problem. Kratochvíl and Matoušek (1991) showed, unexpectedly, that $c_s(n) \geq 2^{cn}$ for some constant c, and conjectured that $c_s(n) \leq 2^{cn^k}$ for some k. The papers by Pach and Tóth (2001), and Schaefer and Štefankovič (2001) established upper bounds of this form, implying decidability of the string graph problem in nondeterministic exponential time.

The string graph problem is closely related to a graph drawing problem, a connection we will make use of later. Given a graph $G = (V, E)$ and a set $R \subseteq \binom{E}{2} = \{\{e, f\} : e, f \in E\}$ on E, we call a drawing D of G in the plane a *weak realization* of (G, R) if only pairs of edges which are in R are allowed to intersect in D (they do not have to intersect, however). In this case we call (G, R) *weakly realizable*. We say that D is a *realization* of G if *exactly* the pairs of edges in R intersect in D.[1] Let us define $c_w(G, R)$ as the smallest number of intersections in a weak realization of (G, R), $c_w(G) = \max\{c_w(G, R) : (G, R) \text{ has a weak realization}\}$, and $c_w(m) = \max\{c_w(G) : G \text{ has } m \text{ edges}\}$.

The string graph problem reduces to the weak realizability problem in polynomial time (Matoušek et al., 1988; Kratochvíl, 1991b) as follows: Given a graph $G = (V, E)$, let $G' = (V \cup E, \{\{u, e\} : u \in e \in E\})$, and $R = \{\{\{u, e\}, \{v, f\}\} : \{u, v\} \in E\}$. Then G is a string graph if and only if (G', R) is weakly realizable.

[1] Kratochvíl (1998, 1991a,b) calls (G, R) an *abstract topological graph*, and uses the word *feasible* for weakly realizable.

In Theorem 4.4 we show that the weak realizability problem lies in **NP**. Because of the reduction of the string graph problem to the weak realizability problem, and Kratochvíl's proof of **NP**-hardness of the string graph problem (Kratochvíl, 1991b) this implies the following corollaries.

Corollary 1.1 *The string graph problem is* **NP***-complete.*

Corollary 1.2 *The weak realizability problem is* **NP***-complete.*

The corollaries imply that the weak realizability problem can be reduced to the string graph problem in polynomial time. No natural polynomial time reduction witnessing this relationship is known (although there is an **NP**-reduction).

The weak realizability problem generalizes the concept of the *crossing number* of a graph G, which is the smallest number of intersections necessary to draw G in the plane. Garey and Johnson (1983) showed that computing the crossing number is **NP**-complete. Many variants of this problem have been considered in the literature, including the *pairwise crossing number* (or *crossing pairs number*), which is the smallest number of pairs of edges that need to intersect to draw G. Pach and Tóth (2000) recently showed that computing the pairwise crossing number is **NP**-hard. Since there is an **NP**-reduction from this problem to the weak realizability problem (guessing the pairs of edges that are allowed to intersect), we have the following corollary.

Corollary 1.3 *The pairwise crossing number problem is* **NP***-complete.*

The string graph problem is also related to Euler (or Venn) diagrams, and through these to topological inference. Given a specification of the relationships of concepts, such as "some A is B, some B is C, but no A is C", we can ask whether a diagram can be drawn in the plane which illustrates the relationship of the concepts (regions homeomorphic to the unit disk). In this particular case Figure 1 illustrates the given situation.

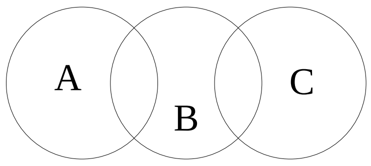

Fig. 1. Some A is B, some B is C, but no A is C.

This problem is polynomial-time equivalent to the string graph problem. Topological inference allows a more refined set of predicates to describe relationship between regions, but even in this case a reduction to the string graph problem can be established, giving us the following result.

Corollary 1.4 *The existential theory of diagrams is* **NP***-complete.*

Details of this reduction (which is an **NP**-reduction rather than a polynomial time one) and the definitions involved can be found in Schaefer and Štefankovič (2001). Several restricted versions of this problem were shown to be solvable in **P** and **NP** earlier, but the general problem was not known to be decidable (Grigni et al., 1995; Chen et al., 1998; Thorup, 1998).

For the proof of our main theorem, Theorem 4.4 in Section 4, the same general approach as in our earlier paper (Schaefer and Štefankovič, 2001) proves successful: we reinterpret the weak realizability problem as a problem over words. However, this time we use more sophisticated techniques from topology and monoids. The necessary background material on words and word equations is covered in Section 2, and the topological aspects of the proof are covered in Section 3. Section 5 shows that the string graph problem is decidable in surfaces other than the plane by using recent results on trace monoids.

2 Word Equations

Let Σ be an alphabet of symbols and Θ be a disjoint alphabet of variables. A *word equation* $u = v$ is a pair of words $(u, v) \in (\Sigma \cup \Theta)^* \times (\Sigma \cup \Theta)^*$. The *size of the equation* $u = v$ is $|u| + |v|$. A *solution of the word equation* $u = v$ is a morphism $h : (\Sigma \cup \Theta)^* \to \Sigma^*$ such that $h(a) = a$ for all $a \in \Sigma$ and $h(u) = h(v)$ (h being a morphism means that $h(wz) = h(w)h(z)$ for any $w, z \in (\Sigma \cup \Theta)^*$). A morphism h is uniquely determined by how it is defined on Θ (assuming that $h(a) = a$ for all $a \in \Sigma$). Therefore we can define the *length of the solution* h as $\sum_{x\in\Theta} |h(x)|$.

A *word equation with specified lengths* is a word equation $u = v$ and a function $f : \Theta \to \mathbb{N}$. The solution h has to respect the lengths, i.e. we require $|h(x)| = f(x)$ for all $x \in \Theta$.

Let w be a word in Σ^*. We write $w[i..j]$ for the subword of w starting at position i and ending in position j. We can represent w as $w = c_1 f_1 c_2 \ldots c_k f_k$ where the c_i are characters in Σ, and the f_i are subwords of w. More precisely, $c_1 = w[1]$ and f_i is the longest prefix of $w[|w| - |c_1 f_1 \ldots f_{i-1} c_i|..|w|]$ which occurs in $c_1 f_1 \ldots f_{i-1} c_i$. The *Lempel-Ziv (LZ) encoding* of w is $LZ(w) = c_1[a_1, b_1]c_2 \ldots c_k[a_k, b_k]$ where $f_i = w[a_i...b_i]$. The *size of the encoding* is $|LZ(w)| = k(2 \log |w| + \log |\Sigma|)$. Note that some words can be compressed exponentially.

Let $h : (\Sigma \cup \Theta)^* \to \Sigma^*$ be a solution of an equation $u = v$. The LZ-encoding of h is the sequence of LZ-encodings of $h(x)$ for all $x \in \Theta$. The size of the encoding is $|LZ(h)| = \sum_{x\in\Theta} |LZ(h(x))|$.

If we know that a word equation $u = v$ has a solution that can be compressed to polynomial size then we can use the following result to verify such a solution.

Theorem 2.1 (Gąsieniec et al. (1996)) *Let $u = v$ be a word equation. Given an LZ-encoding of a morphism h we can check whether h is a solution of the equation in time polynomial in $|LZ(h)|$.*

Translating the weak realizability problem into a word problem results in word equations with given lengths. For these we will be able to find a solution in polynomial time by using the following result.

Theorem 2.2 (Plandowski and Rytter (1998)) *Let $u = v$ be a word equation with lengths specified by a function f. Assume that $u = v$ has a solution respecting the lengths given by f. Then there is a solution h respecting the lengths such that $|LZ(h)|$ is polynomial in the size of a binary encoding of f and the size of the equation. Moreover, the lexicographically least such solution can be found in polynomial time.*

Given an equation with specified lengths there might be solutions which cannot be LZ-compressed. However Theorem 2.2 says that there is a solution which can be LZ-compressed. In particular if the equation has a unique solution then that solution can be LZ-compressed. Note that it is easy to encode several equations into one equation (Lothaire, 2002, Proposition 12.1.8), hence Theorems 2.1 and 2.2 hold for systems of equations as well as single equations.

We will need the following two results which easily follow from Gąsieniec et al. (1996). A *palindrome* is a word w that is identical to its *reverse* w^R.

Lemma 2.3 *Given an LZ-encoding $LZ(w)$ of w we can check whether w is a palindrome in time polynomial in $|LZ(w)|$.*

Lemma 2.4 *Given an LZ-encoding $LZ(w)$ of w and a letter $a \in \Sigma$, we can compute the number of occurrences of a in w in time polynomial in $|LZ(w)|$.*

3 Computational Topology

In the following let M be a compact orientable surface with boundary. A simple arc γ whose two endpoints $\gamma(0), \gamma(1)$ are on the boundary ∂M and whose internal points $\gamma(x), 0 < x < 1$ are in the interior $\dot{M}$ is called a *properly embedded arc*. Two properly embedded arcs γ_1, γ_2 are *isotopic rel. boundary* ($\gamma_1 \sim \gamma_2$) if there is a continuous deformation of γ_1 to γ_2 which does not move the endpoints. The *isotopy class* of γ is the set of properly embedded arcs isotopic to γ. Given two properly embedded arcs γ_1, γ_2 the *intersection number* of γ_1 and γ_2 is

$$i(\gamma_1, \gamma_2) = \min_{c_i \sim \gamma_i} |c_1 \cap c_2|.$$

We say that two properly embedded arcs γ_1, γ_2 are *isotopically disjoint* if $i(\gamma_1, \gamma_2) = 0$, that is, if they can be made disjoint by continuous transformations without moving the endpoints.

Let γ_1, γ_2 be two properly embedded arcs. A *bigon* B bounded by γ_1, γ_2 is a disc (that is, a region homeomorphic to a disc) which has exactly two intersections (of γ_1 and γ_2) on the boundary ∂B.

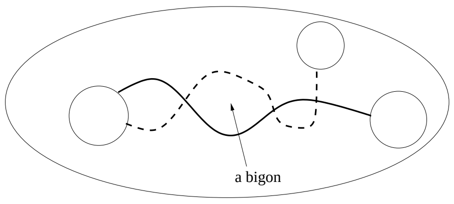

Fig. 2. Example of a bigon.

We will need the following standard result (see Farb and Thurston (1991), for example).

Lemma 3.1 *If two properly embedded arcs γ_1, γ_2 intersect more than $i(\gamma_1, \gamma_2)$ times, then they bound a bigon.*

Lemma 3.2 *Let $\gamma_1, \dots, \gamma_n$ be properly embedded arcs. Then there are $c_1, \dots, c_n$ in general position such that $c_i \sim \gamma_i,\ 1 \le i \le n$ and $|c_i \cap c_j| = i(\gamma_i, \gamma_j),\ 1 \le i, j \le n$.*

Proof Let $c_1, \dots, c_n$ be in general position such that $\sum |c_i \cap c_j|$ is minimized. If there are two properly embedded arcs c_i, c_j which intersect more than $i(\gamma_i, \gamma_j)$ times then they bound a bigon B. Let e, f be the sides of the bigon. Assume that B was selected such that it is the smallest bigon with respect to containment. Then any properly embedded arc c which crosses e also crosses f, otherwise B would not be the smallest bigon. However then we can isotope (rel. boundary) the properly embedded arc c_i to decrease the total number of intersections, a contradiction. See Figure 3. $\square$

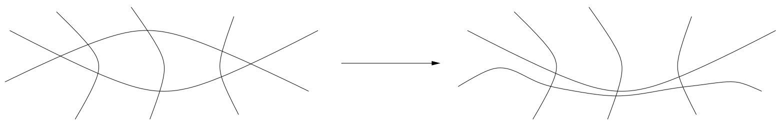

Fig. 3. Decreasing the number of intersections.

Let T be a triangulation of the compact surface M. Let γ be a properly embedded arc. We say that γ is *normal w.r.t. T* if all the intersections with T are transversal and if γ enters a triangle $t \in T$ via edge e then it leaves t via an edge different from e.

Lemma 3.3 *Let T be triangulation of a surface M. Let γ be a non-trivial properly embedded arc. Then there is $c \sim \gamma$ which is normal w.r.t. T.*

Proof Let $c \sim \gamma$ which minimizes the number of intersections with T. If c enters and leaves $t \in T$ through the same edge e then the number of intersections of c with T can be reduced (since γ is assumed to be non-trivial), a contradiction. □

Given a properly embedded arc γ which is normal w.r.t. T we can label each edge of the triangulation with the number of intersections of γ with that edge. In each triangle the labels of its edges determine the behavior of the curve inside the triangle up to isotopy (here we make use of the fact that γ is normal w.r.t. T and that γ is not self-intersecting). Hence the numbering on T determines the isotopy class of γ. We let the size of the representation be the total bitlength of the labels. An isotopy class may have many different representations.

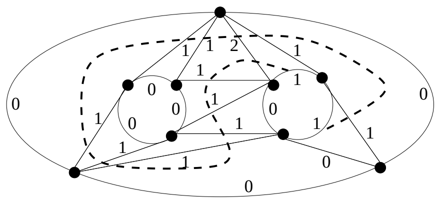

Fig. 4. Example of a numbering.

Call a numbering $\ell : E_T \to \mathbb{N}$ of T *valid* if there is a properly embedded arc γ in general position w.r.t. T which intersects each $e \in T$, $\ell(e)$ times. We say that γ *realizes* the numbering ℓ.

Let ℓ be a valid numbering. The sum of the labels of edges from $E_T \cap \partial M$ is 2. For each triangle $t \in T$ the labels a, b, c of edges of t satisfy $a + b \geq c, a + c \geq b, b + c \geq a$ and $a + b + c$ is even. These conditions are necessary for validity, but not sufficient. Call a labeling satisfying these conditions *semi-valid*. Any semi-valid labeling defines a properly-embedded arc and a (possibly empty) set of closed curves.

We associate the following system of word equations with the triangulation T. The system will emulate the behavior of a set of labeled curves on M. We assume that the curves do not intersect. For each oriented edge $(u, v) \in T$ there is a variable $x_{u,v}$ encoding the order in which the curves intersect on (u, v). Let $t \in T$ be a triangle with vertices u, v, w. We add six variables $y_{t,u}$, $y_{t,v}$, $y_{t,w}$, $y_{u,t}$, $y_{v,t}$, $y_{w,t}$ as shown in Figure 5. For example, the variable $y_{t,u}$ encodes the segments of curves between the oriented edges (w, u) and (v, u). We have the following equations:

$$x_{u,v} = y_{u,t}y_{t,v}, x_{v,u} = y_{v,t}y_{t,u},$$

$$x_{v,w} = y_{v,t}y_{t,w}, x_{w,v} = y_{w,t}y_{t,v},$$

$$x_{u,w} = y_{u,t}y_{y,w}, x_{w,u} = y_{w,t}y_{t,u}.$$

Note that if we know the lengths of the x variables, then we can calculate the lengths of the other variables, for example $|y_{u,t}| = (|x_{u,v}| + |x_{u,w}| - |x_{v,w}|)/2$.

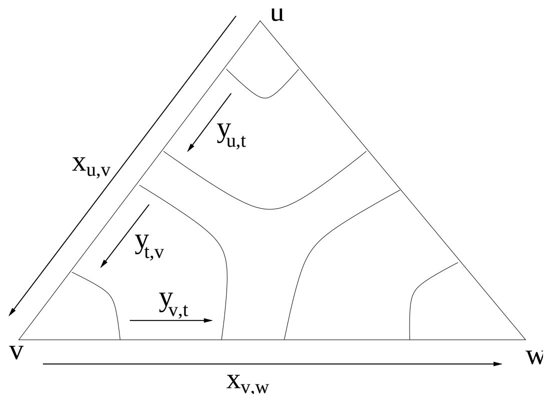

Fig. 5. The variables for a triangle t.

Lemma 3.4 *Given a numbering $\ell : E_T \to \mathbb{N}$ we can test whether ℓ is valid in polynomial time.*

Proof We first verify that ℓ is semi-valid, and reject ℓ if it is not.

Let $\Sigma = \{a, b\}$. Take the set of equations associated with T over Σ. For each $e = (u, v) \in E_T$ specify $|x_{u,v}| = \ell(e)$. For each edge $e = (u, v) \in E_T \cap \partial M$ we specify $x_{u,v} = b^{\ell(e)}$.

We claim that if ℓ is valid then the system of equations has a unique solution. Take the properly embedded arc γ which realizes ℓ. Number the intersections of γ with T in the order in which they occur on γ. Each intersection corresponds to a position in some variable. By induction on the number of intersections it follows that each position in every variable is forced to be b.

On the other hand let us assume that ℓ is not valid. Because it is semi-valid, there is a solution to the set of word equations. However, a lexicographically smallest solution will now contain the letter a (corresponding to the closed curves, disconnected from the endpoints labeled b). Because of Theorem 2.2 we can compute the lexicographically least solution in polynomial time, and—using Lemma 2.4—check that it does not contain any occurrences of a.

Thus by solving the system, we can check whether ℓ is valid. □

Lemma 3.5 *Let γ_1, γ_2 be properly embedded arcs given by valid numberings ℓ_1, ℓ_2. If γ_1, γ_2 do not intersect then we can verify that $i(\gamma_1, \gamma_2) = 0$ in polynomial time. Furthermore if the verification concludes that $i(\gamma_1, \gamma_2) = 0$, then γ_1 and γ_2 are isotopically disjoint.*

Proof Let $\Sigma = \{a, b\}$. Take the set of equations associated with T over Σ. For each $e = (u, v) \in E_T$ specify $|x_{u,v}| = \ell_1(e) + \ell_2(e)$. For each edge $e = (u, v) \in E_T \cap \partial M$ we specify $x_{u,v} = w$ where w represents the order in which γ_1 and γ_2 occur on e.

Let h be a solution of the system. Assume that the number of occurrences of a in $h(x_{u,v})$ is $\ell_1(u,v)$ and $x_{u,v} = x^R_{v,u}$ for all $(u,v) \in E_T$. Then γ_1 and γ_2 are isotopically disjoint.

If γ_1 and γ_2 are disjoint then the system has a unique solution h. The proof is analogous to the argument in the proof of Lemma 3.4.

Since the solution h is unique we can find $LZ(h)$ in polynomial time by Theorem 2.2. For each $e = (u,v) \in E$ we verify that the number of occurrences of a in $h(x_{u,v})$ is $\ell_1(e)$, and $x_{u,v} = x^R_{v,u}$ using Lemma 2.3, and Lemma 2.4. □

4 Weak Realizability

In this section we show that the weak realizability problem can be decided in **NP**. We first translate the weak realizability problem into a more topological version (Proposition 4.2), so we can apply the topological results.

At this point we need the following bound on the number of intersections in a drawing of a weak realization with the smallest number of intersections.

Theorem 4.1 (Schaefer and Štefankovič (2001)) *Let G be a graph with m edges, $R \subseteq \binom{E}{2}$ such that (G,R) is weakly realizable, and let D be a weak realization of (G,R) with the minimal number of intersections. Then for any edge $e \in G$ there are less than 2^m intersections on the curve realizing e in D.*

With this bound we can derive an upper bound on intersections in the topological variant of weak realizability (Lemma 4.3). We can then apply the results on numberings from Section 3 to finish the proof (Theorem 4.4).

To apply our topological results, we define a topological variant of the weak realizability problem. Let $G = (V,E)$ be a graph. Let M be the surface obtained from the plane by drilling $|V|$ holes, each hole labeled by a vertex of G. Let $R \subseteq \binom{E}{2}$. A set S of properly embedded arcs on M is called a *weak realization with holes* of (G,R) if for each $e = \{u,v\} \in E$ there is a properly embedded arc in S connecting hole u to hole v and if $(e,f) \notin R$ then the properly embedded arcs e, f are isotopically disjoint.

Given a weak realization D we can drill small holes in place of the vertices to obtain a weak realization with holes. Given a weak realization with holes, by Lemma 3.2 there is a weak realization with holes in which for $(e,f) \notin R$ the properly embedded arcs e, f are disjoint, rather than just isotopically disjoint. Contracting the holes we obtain a weak realization of (G,R). This proves the following proposition.

Proposition 4.2 *(G,R) is weakly realizable iff there is a weak realization with holes.*

Lemma 4.3 *Let G be a graph with m edges and n vertices. Assume that (G, R) has a weak realization with holes. Let M be the surface obtained from the plane by drilling $|V|$ holes. Let T be a minimal triangulation of M. Then there is a weak realization with holes of (G, R) in M such that there are at most 2^{12n+m} intersections on each edge of T.*

Proof We can construct a triangulation T with $3n$ vertices, using 3 vertices for each boundary component (hole) and no vertices in the interior of the surface. By a simple application of Euler's formula, T has $9n - 6$ edges.

Consider the following weak realization problem. Graph H has vertices $V_H = V_T \cup V_G$. We also include all edges from G and T in H, so that $E_T \cup E_G \subseteq E_H$. Moreover, there are edges connecting $v \in V_G$ to all vertices of T which lie on the boundary of the hole labeled v. The pairs P of edges which are allowed to intersect are the following:

- All pairs in R are allowed to intersect.
- For every $v \in G$ fix an edge $e_v \in T$ on the boundary of the hole labeled v. Then e_v is allowed to intersect edges in E_G incident to v.
- Any edge in T which is not on the boundary ∂M can intersect any edge in E_G.

By construction, (G, R) is weakly realizable iff (H, P) is weakly realizable.

Consider the realization of H with the smallest number of intersections. By Theorem 4.1 there is a realization of H such that along each edge there are at most $2^{|E_H|} \leq 2^{12n+m}$ intersections. □

Theorem 4.4 *The weak realizability problem is in* **NP**.

Proof Let (G, R) be an instance of the weak realizability problem with $n = |V_G|$, and $m = |E_G|$. We will show that deciding whether (G, R) has a weak realization with holes lies in **NP**. Since Proposition 4.2 established the equivalence of weak realizability and weak realizability with holes, this proves the result.

Suppose (G, R) has a weak realization with holes. Let T be a minimal triangulation of the surface M. Lemma 4.3 implies that there is a weak realization with holes in which every edge of T is intersected at most 2^{12n+m} times. Hence every edge e of G can be represented by an arc γ with a numbering $\ell_\gamma : E_T \to \{0, \ldots, 2^{12n+m}\}$. Furthermore, by Lemma 3.2, we can assume that two arcs γ_1 and γ_2 representing two edges $(e, f) \notin R$ are disjoint. To verify weak realizability with holes of (G, R) it is therefore sufficient to guess, for each edge e of G, a numbering $\ell_e : E_T \to \{0, \ldots, 2^{12n+m}\}$ of T (note that the numbering has size polynomial in G). We then check that all guessed numberings are valid, and verify that for every $(e, f) \notin R$ the curves representing e and f are isotopically disjoint. Both tasks can be performed in polynomial time (Lemma 3.4 and Lemma 3.5). Since the verification succeeds if and only if (G, R) has a weak realization by holes, this implies that weak realizability with holes can be verified in **NP**. □

5 String Graphs on Surfaces and Trace Monoids

In proving Theorem 4.4 on string graphs in the plane we only once used the fact that the surface was a plane, namely when we applied Theorem 4.1 which has only been shown for the plane. The rest of the proof—Lemma 3.2 and the assumption that we can triangulate the surface in Theorem 4.4—hold true for any compact surface. However, we do not currently have any good bounds on the number of intersections necessary to realize a string graph in a surface other than the plane, since the methods in the proof of Theorem 4.4 do not seem to apply to surfaces of higher genus than the plane. We make the following conjecture extending the original conjecture of Kratochvíl and Matoušek (1991).

Conjecture 5.1 *A graph that has a weak realization on any compact surface can be realized on that surface with at most $O(2^{n^k})$ intersections for some fixed $k > 0$.*

If the conjecture were true, then the proof of Theorem 4.4 would generalize to compact surfaces of any genus, and we would obtain that the weak realization problem, and therefore the string graph problem, are in **NP** for arbitrary orientable surfaces. Meanwhile we can only state the following result, for which we let $c_s^S(G)$ be the generalization of $c_s(G)$ to a surface S.

Theorem 5.2 *String graphs can be recognized in* $\mathbf{NTIME}(\log c_s^S(n))$ *on a compact surface S, where n is the number of vertices of the string graph.*

In the absence of any upper bound on c_s^S for any surface other than the plane this means that we face the question of decidability anew. This time we use a different approach already suggested by the proof of Theorem 4.1. We will use the connection between weak realizability and equations on words. Words over an alphabet form a monoid, with concatenation as the operation on the monoid.

Our successful use of monoids in the proof of Theorem 4.4 relied on three facts: the exponential upper bound on c_w which allowed us to guess the lengths of the variables involved in the word equations; the fact that we only had to verify arcs that do not intersect making it possible to phrase the problem as an equation between words; and that we could assume the solution to be unique, resolving the question of how to deal with the operator $\cdot^R$.

Since none of these conditions are guaranteed in the general case, we need stronger results on word equations:

- we cannot make assumptions on the lengths of variables,
- arcs can intersect, that is some pairs of letters must be allowed to commute, and
- we have to allow equations using $\cdot^R$.

The main problem here is the second item: allowing selected pairs of letters to commute. Let F be the free monoid over the alphabet Γ, and I be an irreflexive, symmetric relation on Γ. I defines an equivalence relation $\equiv_I$ on F by taking the transitive and reflexive closure of $\equiv_I^1$ which is defined as $u \equiv_I^1 v$ if and only if $u = xaby$, and $v = xbay$ for some $(a, b) \in I$. That is, $\equiv_I$ identifies words in F that can be obtained from each other by permuting pairs of letters in I. The monoid $M(\Gamma, I) = F/\equiv_I$ is called a *trace monoid* over alphabet Γ with *independence relation* I. Intuitively, we equate words over Γ modulo the allowed commutations in I.

Trace monoids have been under investigation for a while, and Matiyasevich showed in 1996 that the solvability of equations in trace monoids is decidable (Diekert et al., 1997). For our purpose we need a stronger result that allows equations with the operator $\cdot^R$. This result was only recently obtained by Diekert and Muscholl (2001), building on work by Diekert et al. (2001). For our purposes, a simplified version of this result is sufficient. Note that our translation from weak realizability to word equations resulted in a quadratic system; that is, every variable occurred at most twice. For these systems, Diekert showed the following result, whose proof can be found in the appendix.

Theorem 5.3 (Diekert (2002)) *The solvability of quadratic systems of word equations in trace monoids with involution can be decided in* **PSPACE**. *Furthermore, if the system is solvable, there is a solution of at most double-exponential length.*

We can apply this result to the string graph problem on an arbitrary compact surface S. Our goal is to decide whether (G, R) is weakly realizable in S. We add $|G|$ holes to the surface S, and fix a triangulation T of the resulting surface. As we did in the proof of Lemma 3.4 we introduce a variable $x_{u,v}$ for each edge (u, v) in T, and variables $y_{t,u}$, $y_{t,v}$, $y_{t,w}$, $y_{u,t}$, $y_{v,t}$, $y_{w,t}$ for each triangle t in T. (See Figure 5.) We require that $x_{u,v} = y_{u,t}y_{t,v}$, $x_{v,u} = y_{v,t}y_{t,u}$, $x_{v,w} = y_{v,t}y_{t,w}$, $x_{w,v} = y_{w,t}y_{t,v}$, and $x_{u,w} = y_{u,t}y_{y,w}$, $x_{w,u} = y_{w,t}y_{t,u}$. Furthermore, we ask that $y_{t,u} = y_{u,t}^R$ for all vertices u, and triangles t of T. We associate each vertex h of G arbitrarily with an edge (u, v) of T on the boundary of S such that no two vertices are associated with edges that share a vertex. Let $\Gamma = \{e : e \text{ is an edge of } G\}$. For each vertex h of G, let $E(h)$ be the set of edges that contain h. For each h fix an arbitrary permutation p_h of these edges (as letters in Γ), and add the equation $x_{u,v} = e_h$ to the system, where (u, v) is the edge associated with h. Consider the trace monoid over Γ with independence relation R. We claim that the set of equations we built has a solution in that trace monoid, if and only if (G, R) is weakly realizable in S. This, however, is immediate, since the allowable crossings in the trace monoid correspond to the allowed intersections in R. We also observe that every variable occurred at most twice, making the system quadratic.

Corollary 5.4 *The string graph problem, and the weak realizability problem are decidable in* **PSPACE** *on any compact surface. Furthermore,* $c_s^S(G) = O(2^{2^{poly(|G|)}})$.

The upper bound on c_s^S is one exponential jump beyond Conjecture 5.1, one that will probably be hard to gap, since it would have consequences for word and trace equations (see the appendix).

6 Conclusion

We now have three essentially different proofs of the decidability of string graphs. Pach and Tóth (2001) solved the problem using purely graph theoretical and topological methods. At the same time Schaefer and Štefankovič (2001) settled the problem based on the idea of looking at intersections along edges as words, and looking for avoidable patterns in words. In Section 5 we modified this idea and showed that we can easily rephrase the realizability of string graphs—even on arbitrary surfaces—as a solvability problem over trace monoids with involutions, a problem shown decidable by Diekert and Muscholl (2001). While this has led us to a better understanding of the connections between word equations and graph drawing, we feel that we still have not been able to lay our hands on the combinatorial and topological heart of the string graph problem. It is still as mysterious as it used to be.

Acknowledgment. We would like to thank Volker Diekert and Géza Tóth for helpful discussions, and Volker Diekert for custom-tailoring Theorem 5.3 for our purposes, and allowing us to include it in the paper.

References

Benzer, S., 1959. On the topology of the genetic fine structure. Proceedings of the National Academy of Science 45, 1607–1620.

Chen, Z.-Z., Grigni, M., Papadimitriou, C., 1998. Planar map graphs. In: Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98). ACM Press, New York, pp. 514–523.

Diekert, V., 2002. Personal communication.

Diekert, V., Gutiérrez, C., Hagenah, C., 2001. Existential theory of equations with rational constraints in free groups is PSPACE-complete. In: Proceedings of the 18th International Symposium on Theoretical Aspects of Computer Science, STACS 2001. pp. 170–182.

Diekert, V., Matiyasevich, Y., Muscholl, A., 1997. Solving trace equations using lexicographical normal forms. In: Proceedings of the24th International Colloquium on Automata, Languages and Programming (ICALP'97), Bologna (Italy) 1997. Springer, Berlin-Heidelberg-New York, pp. 336–346.
URL `citeseer.nj.nec.com/diekert97solving.html`

Diekert, V., Métivier, Y., 1997. Partial commutation and traces. In: Rozenberg, G., Salomaa, A. (Eds.), Handbook on Formal Languages. Vol. III. Springer, Berlin-Heidelberg-New York.

Diekert, V., Muscholl, A., 2001. Solvability of equations in free partially commutative groups is decidable. In: ICALP 2001. pp. 543–554.

Farb, B., Thurston, B., 1991. Homeomorphisms and simple closed curves. unpublished manuscript .

Garey, M. R., Johnson, D. S., 1983. Crossing number is NP-complete. SIAM Journal on Algebraic and Discrete Methods 4 (3), 312–316.

Gąsieniec, L., Karpinski, M., Plandowski, W., Rytter, W., 1996. Efficient algorithms for Lempel-Ziv encoding. in Proceedings of SWAT'96, LNCS 1097 , 392–403.

Graham, R. L., 1976. Problem 1. In: Open Problems at 5th Hungarian Colloquium on Combinatorics.

Grigni, M., Papadias, D., Papadimitriou, C., 1995. Topological inference. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence. pp. 901–906.

Kratochvíl, J., 1991a. String graphs. I. the number of critical nonstring graphs is infinite. Journal of Combinatorial Theory, Series B 52.

Kratochvíl, J., 1991b. String graphs. II. recognizing string graphs is NP-hard. Journal of Combinatorial Theory, Series B 52.

Kratochvíl, J., 1998. Crossing number of abstract topological graphs. Lecture Notes in Computer Science 1547, 238–245.

Kratochvíl, J., Matoušek, J., 1991. String graphs requiring exponential representations. Journal of Combinatorial Theory, Series B 53, 1–4.

Lothaire, M., 2002. Algebraic Combinatorics on Words. Vol. 90 of Encyclopedia of Mathematics and its Applications. Cambridge University Press.

Matoušek, J., Nešetřil, J., Thomas, R., 1988. On polynomial time decidability of induced-minor-closed classes. Comment. Math. Univ. Carolin. 29 (4), 703–710.

Mazurkiewicz, A., 1995. Introduction to trace theory. In: Diekert, V., Rozenberg, G. (Eds.), The Book of Traces. World Scientific, Singapore, Ch. 1, pp. 3–41.

Pach, J., Tóth, G., 2000. Which crossing number is it anyway? Journal of Combinatorial Theory, Series B 80, 225–246.

Pach, J., Tóth, G., 2001. Recognizing string graphs is decidable. In: Mutzel, P. (Ed.), Graph Drawing 2001. Lecture Notes in Computer Science. Springer, pp. 247–260.

Plandowski, W., Rytter, W., 1998. Application of Lempel-Ziv encodings to the solution of words equations. In: Automata, Languages and Programming. pp. 731–742.

Robson, J. M., Diekert, V., 1999. On quadratic word equations. In: Meinel, C., et al. (Eds.), 16th STACS, Trier, 1999. Vol. 1563 of LNCS. Springer-Verlag, pp. 217–226.

Schaefer, M., Štefankovič, D., 2001. Decidability of string graphs. In: Proceedings of the 33th Annual ACM Symposium on Theory of Computing (STOC-2001). pp. 241–246, invited to the JCSS special issue on STOC'01.

Sinden, F. W., 1966. Topology of thin film circuits. Bell System Technical Journal 45, 1639–1662.

Thorup, M., 1998. Map graphs in polynomial time. In: IEEE Symposium on Foundations of Computer Science. pp. 396–405.

A Quadratic Equations

A word or trace equation is called *quadratic*, if every variable occurs at most twice. Quadratic equations are significantly easier to handle than the general case, and often better complexity bounds are available. Compare, for example, Theorem 2.2 to the following result.

Proposition A.1 (Robson and Diekert (1999)) *Let $u = v$ be a quadratic word equation with lengths specified by a function f. In linear time we can decide whether the equation has a solution, and produce a most general solution to the equation (in the form of a straight line program).*

Diekert (2002) has pointed out that the proof of this result can easily be extended to include equations that contain the involution operator $\cdot^R$ (which reverses a word). Using this extended result we could slightly simplify the proof of Lemma 3.5, and reduce the complexity of the verification step of the **NP**-algorithm in Theorem 4.4 from polynomial time to linear time.

More importantly for our purposes, a similar result can be obtained for quadratic trace equations. The original version of this paper made use of a solution to general trace equations with involution due to Diekert and Muscholl, building on work by Diekert et al. (2001), and Diekert et al. (1997).

Proposition A.2 (Diekert and Muscholl (2001)) *The solvability of systems of equations in trace monoids with involution is decidable.*

A close look at the decision procedure showed that it could probably be made to run in **EXPSPACE**. However, it turns out that the special case of quadratic trace equations, even with involution, is much simpler, and solvability can be decided in **PSPACE**. This unpublished result is due to Volker Diekert who generously allowed us to include a proof of it in this paper. We repeat the statement of Theorem 5.3 from Section 5.

Theorem 5.3 (Diekert (2002)) *The solvability of quadratic systems of word equations in trace monoids with involution can be decided in* **PSPACE**. *Furthermore, if the system is solvable, there is a solution of at most double-exponential length.* □

Theorem 5.3 is not believed to be optimal. The problem is known to be **NP**-hard, and Robson and Diekert (1999) conjectured the variant for words to be in **NP**.

For the proof we need the Levi-lemma for traces, a standard tool in the theory of traces due to Cori and Perrin (Diekert and Métivier, 1997; Mazurkiewicz, 1995). We do not include its proof. For a word w over an alphabet Γ, let $alph(w) \subseteq \Gamma$ denote the set of letters occurring in w.

Lemma A.3 *Let $M = M(\Gamma, I)$ be a trace monoid over alphabet Γ with independence relation I. If x, u, y, w are traces in M, then $xu = yw$ if and only if there are traces p, r, s, and q such that $x = pr$, $y = ps$, $u = sq$, $w = rq$ and $alph(r) \times alph(s) \subseteq I$.*

For two traces r and s we write $(r, s) \in I$ if $alph(r) \times alph(s) \subseteq I$; that is, every letter in r commutes with every letter in s. In this case $alph(r)$ and $alph(s)$ have to be disjoint, since I is irreflexive.

Proof Theorem 5.3 Fix the trace monoid $M = M(\Gamma, I)$, and the set Ω of variables. We are given a quadratic system of trace equations. A solution to the system is a mapping $\sigma : \Omega \cup \Gamma \to \Gamma^*$ which fulfills all equations, and such that σ is the identity on Γ.

For each variable x in Ω the algorithm guesses a set $\alpha(x) \subseteq \Gamma$ for the letters in $\sigma(x)$ of a solution σ, if there is one. Since **PSPACE** is closed under nondeterminism, and $\alpha(x)$ has size at most Γ, this can be done in **PSPACE**. In the following we assume we are looking at a fixed $\alpha : \Omega \to 2^\Gamma$.

The *weight* $\mu(u)$ of a word $u = u_1 \cdots u_k \in (\Gamma \cup \Omega)^*$ is defined as $\sum_{i=1}^{k} |\alpha(u_i)|$, where we let $\alpha(a) = \{a\}$ for all $a \in \Gamma$. We extend the notion of weight to equations, and system of equations by adding the weights of all parts. The weight of a system is a measure of how much space the algorithm needs to store the all information it needs about the system of equations. Note that initially the weight of a system can be at most quadratic in the size of the input. In each step of the algorithm we guarantee that the weight of the system will not increase. Hence the algorithm requires only polynomial space.

The weight upper bound will not guarantee the algorithm's termination. For this we need a different measure. Suppose a system of equations has a solution $\sigma : \Gamma \cup \Omega \to \Gamma^*$. We can define the *length* of an equation $u_1 \cdots u_n = v_1 \cdots v_m$ with regard to σ as $\sum_{i=1}^{n} |\sigma(u_i)| + \sum_{i=1}^{m} |\sigma(v_i)|$, and the length of a system of equations with regard to σ as the sum of the lengths of all its equations (with regard to σ). The length of a system of equations then is the minimum of all lengths over all possible solutions σ. If there is no solution, we let the length be infinite.

We will show that the algorithm decreases the length of the system by at least 1 in each step, and therefore it has to terminate. Since we guarantee that the weight of the system does not increase, and that weight initially is polynomial, we will be able to conclude that the algorithm runs in polynomial space.

Consider an equation

$$xu = y_1 \cdots y_k \tag{A.1}$$

with $x, y_1, \ldots, y_k \in \Gamma \cup \Omega$ and $u \in (\Gamma \cup \Omega)^*$. Choose i maximal such that $(x, y_1 \cdots y_{i-1}) \in I$, and let $y = y_i$. (Note that we can assume that $i - 1 < k$: if $(x, y_1 \cdots y_k) \in I$, then $alph(x) \cap alph(y_1 \cdots y_k) = \emptyset$ in which case we can conclude that there is no solution to the system.) Equation (A.1) then turns into

$$xu = vyw \tag{A.2}$$

where $x, y \in \Gamma \cup \Omega$, $v, w \in (\Gamma \cup \Omega)^*$, and $(x, v) \in I$. Since $alph(x) \cap alph(v)$ has to be empty (because $(x, v) \in I$), x has to be a prefix of yw. By Levy's Lemma, we know that there are traces p, r, and s such that $x = pr$, and $y = ps$, where $(r, s) \in I$. This allows us to rewrite Equation (A.2) as

$$pru = vpsw \tag{A.3}$$

which is the same as $pru = pvsw$, since (p, v) has to be in I, as (x, v) is, and therefore

$$ru = vsw \tag{A.4}$$

because we can cancel p. At this point we add new variables r and s to Γ, and guess $\alpha(p)$, $\alpha(r)$, and $\alpha(s)$ such that

$$\begin{aligned} \alpha(x) &= \alpha(p) \cup \alpha(r) \\ \alpha(y) &= \alpha(p) \cup \alpha(s) \\ \alpha(r) \times \alpha(s) &\subseteq I \end{aligned}$$

Since we assumed that i is maximal, we know that p cannot be empty. We distinguish five cases depending on x and y.

Case 1: $x = a \in \Gamma$. In this case $p = a$ and $r = \epsilon$. If $y \in \Gamma$, then y has to be a (otherwise there is no solution), and substituting Equation (A.1) by Equation (A.4) reduced the overall weight by 2. Otherwise $y \in \Omega$, and we know that $y = as$. Hence we can substitute the second occurrence of y with as. This potentially increases the weight of the system by 1, but since we also replace Equation (A.1) by Equation (A.4) (with $r = \epsilon$) reducing the weight by 1, we have not increased the overall weight.

Case 2: $y = a \in \Gamma$. Similar to Case 1.

Hence, we know that $x, y \in \Omega$.

Case 3: $x = y$. Again, substituting Equation (A.1) by Equation (A.4) reduces the weight by 2 (remember that we removed variables u with $\alpha(u) = \emptyset$).

Case 4: $x = y^R$. In this case $ps = y = x^R = (pr)^R = r^R p^R$. Since s and r do not have any letters in common (remember that $(r, s) \in I$), this is impossible, unless both $s = r = \epsilon$, and therefore $x = y = p = p^R$. Since there is always a solution to $y = y^R$, whatever $alph(y)$, this allows us to replace Equation (A.1) by $u = vw$, reducing the weight of the system. (There cannot be any other occurrences of y, since it already appears as y and y^R.)

Case 5: $x, y \in \Omega$, and $x \neq y, y^R$. Substitute Equation (A.1) by Equation (A.4), and replace the second occurrence of x (if any) by pr (or $r^R p^R$, if it is x^R), and the second occurrence of y (if any) by ps (or $s^R p^R$, if it is y^R). We changed the overall weight of the system by

$$-2(|\alpha(x)| + |\alpha(y)|) + 2|\alpha(p)| + |\alpha(r)| + |\alpha(s)| \leq 0$$

The inequality is true, since $\alpha(x) \cap \alpha(y) = \alpha(p)$ (r and s being independent), and therefore $|\alpha(x)| + |\alpha(y)| = |\alpha(x) \cup \alpha(y)| + |\alpha(x) \cap \alpha(y)| = |\alpha(p) \cup \alpha(r) \cup \alpha(s)| + |\alpha(p)| \geq |\alpha(r) \dot{\cup} \alpha(s)| + |\alpha(p)| = |\alpha(p)| + |\alpha(r)| + |\alpha(s)|$. Therefore the weight of the system did not increase.

The five cases are exhaustive. Note that in each case the weight of the system did not increase. Furthermore, in all cases the length of the system was reduced by at least $2|p|$, since $xu = vyw$ (which is $pru = vpsw$) is replaced by $ru = vsw$. Since p cannot be empty, this means the length of the system is reduced in every step. Hence, the algorithm has to terminate, either by running into a contradiction, or with the empty system, in which case there is a solution (since the solvability of the system remained unchanged). In every step of the algorithm we remove variables x and y from the system and replace them by pr and ps. Hence $|x| + |y| \leq |p| + |r| + |p| + |s|$ which means that in every step the length of the system at most halves. Since the algorithm takes an exponential number of steps, this gives us an upper bound of $O(2^{2^{poly(n)}})$ on the length of the original system of equations, showing that there is a solution of double exponential length.

$\square$