Computing Dehn Twists and Geometric Intersection Numbers in Polynomial Time

Marcus Schaefer
Department of Computer Science
DePaul University
243 South Wabash
Chicago, Illinois 60604, USA
mschaefer@cs.depaul.edu

Eric Sedgwick
Department of Computer Science
DePaul University
243 South Wabash
Chicago, Illinois 60604, USA
esedgwick@cs.depaul.edu

Daniel Štefankovič
Department of Computer Science
University of Chicago
1100 East 58th Street
Chicago, Illinois 60637, USA
stefanko@cs.uchicago.edu

November 4, 2004

Abstract

Simple curves on surfaces are often represented as sequences of intersections with a triangulation. However, there are much more succinct ways of representing simple curves used by topologists, including normal coordinates and Dehn-Thurston coordinates. In these representations, the length of a curve might be exponential in the size of its representation.

Nevertheless, we show that the following two basic tasks of computational topology, namely

- performing a Dehn-twist of a curve along another curve, and
- computing the geometric intersection number of two curves,

can be solved in polynomial time even in the succinct Dehn-Thurston, or normal coordinate representations. These are the first algorithms for these two problems that solve these problems for arbitrary simple curves, and, moreover, solve them in polynomial time in the succinct representations.

1 Introduction

For many problems in topology algorithms have been known for some time but the systematic study of the complexity of topological problems is more recent and belongs to an area known as computational topology.

One fundamental problem has been recognizing the unknot. The story begins with Kneser [Kne30] who, in 1930, introduced a succinct representation for curves and surfaces in which these objects are described by their *normal coordinates*. This led to the theory of normal surfaces which was used by Haken in 1961 to show that the unknot could be recognized by an algorithm (which, much later, was shown to run in exponential time). Haken's approach was pushed further by Hass, Lagarias, and Pippenger who exploited the succinctness of the representation to show that the unknot could be recognized in NP [HLP99]. To this end they had to verify in polynomial time that a special type of normal surface was an essential disk, in particular, that is was connected. This result was strengthened by Agol, Hass, and Thurston [AHT02] who showed that the number of connected components of a normal surface can be computed in polynomial time. This immediately implies polynomial time algorithms for checking whether a normal surface is connected, and whether it is orientable.

Independently, we developed a set of tools for algorithms on normal curves in [SSŠ03], work that grew out of the string graph recognition problem [SŠ04]. Among others, we showed how to compute connected components, count them, decide isotopy of curves, and compute the algebraic intersection number of two curves. The essential new feature of our approach was that it was based on recent developments of algorithms over free monoids (see [Loh03] for a survey) rather than free groups.

In the current paper we continue the study of curves by studying the complexity of performing Dehn twists, a fundamental topological operation, and applying the result to show how to determine the geometric intersection number of two curves. Our main results are that

- performing a Dehn twist of a curve along another curve, and
- computing the geometric intersection number of two curves

can be solved in polynomial time in both the normal coordinate representation (in a given triangulation), and for curves given by Dehn-Thurston coordinates (for a given pair-of-pants decomposition).

The statements will be made more precise in the following sections. For the time being, let us point out, however, that these results are very strong, since both the representations we use, are very succinct: the length of the curve is exponential in the size of its representation.

There have been previous attempts at calculating the effect of a Dehn twist. Penner [Pen84] gave explicit formulas describing the action of the Lickorish generators on the Dehn-Thurston coordinates for $\mathcal{C}S(M_g)$, the set of all isotopy classes of simple curves. This solves the problem for a very restricted case only, the Lickorish generators. Dehn twists along other curves can be obtained by applying Dehn twists along Lickorish generators, but exponentially many Dehn twists can be needed.

On the other hand, more recently Hamidi-Tehrani and Chen [HTC96] gave an algorithm to compute the action of a set of generators on the space $\mathcal{C}S(M)$ given by measured π-train tracks, but its running time is exponential in the representation.

Hamidi-Tehrani's thesis [HT97] gave an algorithm for the geometric intersection number running in time polynomial in the length of the curves. Again, this translates only to an exponential time algorithm in the succinct representations.

2 Preliminaries

2.1 Representations of Surfaces and Simple Curves

By a *surface* M we mean a connected, compact, orientable topological surface with boundary ∂M. Let g be the genus of M and let h be the number of boundary components. A simple *multi-curve* α in M is a proper 1-dimensional submanifold of M such that no component of α is null-homotopic or homotopic to the boundary. A connected components of a simple multi-curve is either a *properly* *embedded simple arc* or a *simple closed curve*. By *isotopy* we mean isotopy rel boundary, i.e., a continuous deformation which leaves ∂M fixed. Let $\mathrm{C}S(M)$ be the set of all isotopy classes of simple multi-curves. Let $\mathrm{C}S_0(M) \subseteq \mathrm{C}S(M)$ be the set of isotopy classes of simple multi-curves whose components are simple closed curves.

The *geometric intersection number* $i(\alpha, \beta)$ of two curve systems $\alpha, \beta \in \mathrm{C}S(M)$ is the minimal number of intersections between their representatives.

Common combinatorial representations of surfaces are triangulation, polygonal schema and pair of pants decomposition. A *triangulation* T of a surface M is a set of points V in M and a embedded collection of arcs E such that each component of $M - E$ is an open disc bounded by three curves from E. A triangulation is *minimal* if the vertices V are on the boundary ∂M. A *pair of pants* is a disk with two holes. A *pants decomposition* of a surface M with negative Euler characteristic is a simple multi-curve C such that all the connected components of $M - C$ are pair of pants. A simple Euler characteristic computation shows that C has $3g - 3 + h$ simple curves and $M - C$ consists of $2g - 2 + h$ pairs of pants. A *polygonal schema* is a polygon with even number of edges together with information on how to identify pairs of edges to obtain the surface.

Let M be given by a triangulation. A simple multi-curve γ is *normal* w.r.t. T if all the intersections of γ with T are transversal and if γ enters a triangle t via an edge e then it leaves t via an edge different from t. The vector $(|\gamma \cap e|)_{e \in T}$ is called *normal coordinates* of γ. Any two simple multi-curves with the same normal coordinate are isotopic. If the triangulation is minimal then any isotopic curves have the same normal coordinates.

Dehn-Thurston coordinates parameterize the set of curves on M_g^s by a subset of $\mathbb{Z}^m$ where $m = 6g - 6 + 2s$. We have a pair of pants decomposition C of M and on each curve of C we have an interval called a window. Given a simple multi-curve α, the first $3g - 3 + h$ coordinates are the geometric intersection numbers of α with the components of C. The remaining $3g - 3 + h$ describe how α "twists" within a small neighborhood of each of the component of C (after the curve is isotoped to a certain normal form in each pair of pants).

Let M be given by a triangulation T. Fix an orientation $\vec{e}$ of each edge $e \in T$. Given a simple curve γ let w be a word obtained by following γ and appending e to w if $\vec{e}$ is crossed from left to right and appending e^{-1} is $\vec{e}$ is crossed from right to left. Then w is called *intersection sequence with the triangulation*. If γ is simple closed curve the intersection sequence is not unique. The length of an intersection sequence can exponentially longer than the number of bits needed to write down normal coordinates or Dehn-Thurston coordinates. Fortunately, for simple curves the sequence can always be compressed, as we show in Section 3.

Let α be a simple closed curve in M. A *Dehn twist* $D_\alpha : M \to M$ along α is a homeomorphism of M obtained by cutting M along α, rotating one of the copies of α, 360 degrees and gluing them back. More precisely, if an annulus around α is parameterized by $\{(x, \varphi), 1 \leq x \leq 2, 0 \leq \varphi \leq 2\pi\}$ then D_α is $(x, \varphi) \mapsto (x, \varphi + 2\pi(x-1) \pmod{2\pi})$ on the annulus and identity elsewhere. Dehn twists are interesting because they generate *mapping class group* $\mathcal{M}_g^s$ (the group of orientation preserving homeomorphisms of the surface M_g^s to itself modulo isotopy). Dehn showed that $\mathcal{M}_g$ is generated by finitely many Dehn twists. Lickorish [Lic64, Lic66] gave a set of $3g - 1$ generators of $\mathcal{M}_g$.

2.2 Compressed representation of words and quadratic word equations

Let Σ be an *alphabet*. A word in Σ^* can be represented by a *straight line program* (SLP), a context free grammar in Chomsky normal form generating one string. A straight line program P of length

n is a sequence of assignments $x_i := \text{expr},\ i = 1, \ldots, n$ where expr is a either a symbol from Σ or $x_i x_j, 1 \leq j, k < i$. The length of an SLP representing a word w can be exponentially smaller than $|w|$, the length of w.

Given a word w it is NP-hard to find the shortest SLP generating w. In fact the length of the shortest SLP is NP-hard to approximate within a factor of 1.0001 ([LS02]) but an approximation algorithm with factor $O(\log |w|)$ is known ([CLL$^+$02, Ryt02]).

Equality of two words given by SLPs can be tested in time polynomial in the length of the SLPs [HJM96, Pla94]. More precisely the following is known:

Theorem 2.1 *Let P and P' be straight line programs representing words u and v. Let the lengths of P and P' be m and n.*

- *Testing whether $u = v$ can be done in time $O(m^2n^2)$ by a deterministic algorithm ([MST97]).*
- *Testing whether $u = v$ can be done in time $O(m + n)$ by a randomized algorithm with 1-sided error. If $u = v$ then the algorithm always gives a correct answer, if $u \neq v$ it errs with probability $\leq \delta$ ([GKPR96]). The model used is that arithmetic operations on numbers of size $O(\log 1/\delta)$ have unit cost.*

Let Σ be an alphabet. Let Θ be an alphabet of variables disjoint from Σ. A *word equation* $u = v$ is a pair of words $u, v \in (\Sigma \cup \Theta)^*$. The *size* of the equation 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 all $w, z \in (\Sigma \cup \Theta)^*$.) A *word equation with specified lengths* is a word equation $u = v$ and a function $\ell : \Theta \to \mathbb{N}$. The solution h has to respect the lengths, i.e., we require $|h(x)| = \ell(x)$ for all $x \in \Theta$. An SLP for solution h is an SLP for $h(u)$. The *complexity* of a word equation $u = v$ with specified lengths ℓ is $|u| + |v| + \sum_{x \in \Omega} \log \ell(x)$.

Finding a solution of a word equation is NP-hard. The situation changes when the lengths are given. Plandowski and Rytter [PR98] found an algorithm which finds an SLP of a solution of a word equation with specified lengths in time polynomial in the complexity of the equation.

A word equation is *quadratic* if every variable occurs at most twice in the equation. For quadratic word equations with specified lengths a better algorithm is known:

Theorem 2.2 (Diekert,Robson[RD99]) *Let $u = v$ be a quadratic word equation with specified lengths. In linear time it can be decided whether $u = v$ has a solution. If it has a solution then an SLP of the solution can be found. The length of the SLP is linear.*

3 Moving between the representations

In this section we show that the Dehn-Thurston coordinates, normal coordinates and intersection sequences compressed by straight line programs are polynomially related. Moving from a compressed representation to normal coordinates is simple - we just need to count the number of each type of symbol in a compressed word.

Observation 1 *Let M be a surface given by a triangulation T. Let $\alpha \in \mathrm{CS}(M)$ be a simple curve in M given by intersection sequence with T, given by SLP of size n. The normal coordinates of α can be computed in time $O(n \cdot |T|)$.*

Lemma 3.1 *Let M be a surface given by a triangulation T. Let $\alpha \in \mathrm{CS}(M)$ be a simple curve in M given by normal coordinates on T of complexity n. An SLP of size $O(n)$ for an intersection sequence of α with T can be obtained in time $O(n)$.*

Proof :
We will create a system of quadratic word equations with given lengths such that the solution will give all the intersection sequences. Let L be the sum of all coordinates of α. The length of any intersection sequence of α is L.

For the proof we add the following curves to the surface. Let t be a triangle with vertices a, b, c and edges ab, bc, ca. Fix a vertex v_t inside t and connect v_t to $u, u = a, b, c$ by a simple curve γ_u. For $e = ab, bc, ca$ add a curve η_e in t parallel to e. We can add the curves so that any two are disjoint and they are disjoint from T (except at ∂M).

Now we isotope α so that it intersects the added curves minimally. The curve η_e is intersected $|e \cap \alpha|$ times by α. The number of intersections of α with the γ curves can be easily computed, e. g., $|\gamma_a \cap \alpha| = (|ab \cap \alpha| + |ca \cap \alpha| - |bc \cap \alpha|)/2$.

For each of the added curve we have 2 variables in the system of quadratic word equations, one for each orientation of the curve. The length of the variables for curve β is $L \cdot |\alpha \cap \beta|$.

Fix an orientation $\vec{\alpha}$ on α. For each intersection point p of α and $\vec{\beta}$ take the intersection sequence of α with T starting at p leaving from p to the left of $\vec{\beta}$ (i. e., the direction of travel depends on the sign of the intersection of $\vec{\alpha}$ and $\vec{\beta}$ in p). We set the equations so that in the solution the variable $X(\vec{\beta})$ for $\vec{\beta}$ will contain concatenation of the intersection sequences for points p in the order they occur on $\vec{\beta}$.

We add six equations per triangle, relating the variables for the γ curves and η curves. For example we add $X(ab) = X(av_t)X(v_t b)$. Let η_e and η'_e be a pair of η curves arising from the same edge $\vec{e}$ and having the same orientation. Suppose that η_e is to the left of η'_e. We add equation $X(\eta'_e)s = sX(\eta_e)$ where $s = e$ or $s = e^{-1}$ depending on which symbol represents crossing of e from left to right.

There is a unique solution of the described system of equation, this can be show by starting at the leftmost intersection of α with β on $\vec{\beta}$ and following the curve α. ∎

Let M be a surface and let C be a pair of pants decomposition. For each curve of C add a parallel copy to C. Now $M - C$ consists of pairs of pants and annuli. Add two vertices to each curve and triangulate the annuli and pairs of pants. This will be the triangulation T for the pair of pants decomposition C.

The normal form of a curve within each pair of pants describes how the curve "looks" w.r.t. T and hence its normal coordinates can be computed. On the other hand a curve in T can be locally put in the normal form within each pair of pants. More precisely we have:

Lemma 3.2 *Let C be a pair of pants decomposition of M and let T be the triangulation for C. Let γ be given by Dehn-Thurston coordinates in C of complexity n. Normal coordinates of γ in T can be found in time $O(n)$. Let α be given by normal coordinates in T of complexity n. Then Dehn-Thurston coordinates of α in C can be found in time $O(n)$.*

Proof :
We omit the proof. ∎

4 Computing Dehn twists and geometric intersection numbers

In this section we show how a Dehn twist can be computed in polynomial time. We then use it to compute geometric intersection number.

Theorem 4.1 *Let M be a surface of genus g given by a triangulation T. Let be $\alpha \in \mathrm{CS}(M)$ and $\beta \in \mathrm{CS}_0(M)$ be simple multi-curves in M given by normal coordinates of complexity $\leq n$. The normal coordinates of the Dehn twist $D_\beta(\alpha)$ can be computed*

- *in time $O(g \cdot n^3)$ by a randomized algorithm with small probability of error; and*
- *in time $O(g \cdot n^{10})$ by a deterministic algorithm.*

Proof :
Assume that β is connected. The isotopy class of a Dehn twist $D_\beta(\alpha)$ is independent of the choice of the representatives for α and β. For each edge of $e \in T$ define three segments on e, left, middle and right window. Isotope α, β so that β always crosses the triangulation in middle windows, α so that it always crosses the triangulation in left and right windows and that all the crossings of α and β happen between the η edges defined in the proof of Lemma 3.1.

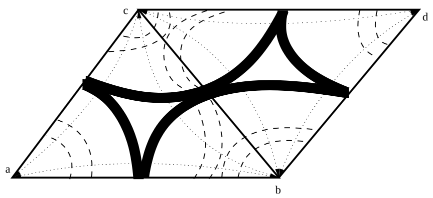

Figure 1: The choice of representatives of α, β for computing $D_\beta(\alpha)$

Let a, b, c, d be vertices of triangles abc and dbc sharing the edge bc. The only segments in α which intersect β correspond to sequences $ab\,bc\,cd$, $ac\,bc\,bd$ (and their inverses). Dehn twist along α transforms the sequence $ab\,bc\,cd$ to $ab\,X(bc)\,bc\,cd$ where X is the variable from the proof of Lemma 3.1, containing the sequence of intersection sequences of α with T. All the transformations can performed simultaneously on the SLP representation in linear time. The resulting SLP has size $\leq 3n$. The resulting sequence is a representative of $D_\beta(\alpha)$ but it is not in normal position w.r.t. T. To obtain a representative in normal position we need to reduce the word which can be done in polynomial time by Lemma 4.5. Once in normal position, we can compute the normal coordinates.

If β is not connected then it has $O(g)$ isotopy classes of components. These can be identified, and the number of copies of each class counted $O(n^2 g)$ time. Now we can perform Dehn twist for each isotopy class separately. If there are c copies of a simple closed curve γ in β then twisting along all of them simultaneously is similar to twisting along a single curve (the only difference is that we have $X(\eta'_e)s^c = s^c X(\eta_e)$ in the proof of Lemma 3.1). ■

Once we can compute Dehn twists in polynomial time we can compute geometric intersection numbers in polynomial time. The reason is that by observing the length of Dehn twists we can compute the geometric number. Looking at representatives of α, β which intersect minimally it follows that $|D^n_\beta(\alpha) \cap T| \leq |\alpha \cap T| + n \cdot i(\alpha, \beta)|\beta \cap T|$, i.e., $D^n_\beta(\alpha)$ "grows" by rate at most $i(\alpha, \beta)|\beta \cap T|$. For sufficiently large n is grows by that rate.

Lemma 4.2 *For $n \geq |\alpha \cap T|$,*

$$|D_\beta^{n+1}(\alpha) \cap T| - |D_\beta^n(\alpha) \cap T| = i(\alpha, \beta)|\beta \cap T|.$$

Proof :
For a simple curve α, $|\alpha \cap T|$ is the length of an intersection sequence of α with T. An intersection sequence of $D_\beta^n(\alpha)$ with T is a reduction of a word of the form $a_1(b_1)^n a_2(b_2)^n a_3 \ldots a_k(b_k)^n a_{k+1}$ where length of each $|b_i|$ is $|\beta \cap T|$ and the length of $a_1 a_2 \ldots a_{k+1}$ is $|\alpha \cap T|$. It is a consequence of Fine-Wilf theorem that for words of type $(b_{i-1})^n a_i (b_i)^n$ reductions either "gets stuck" or eats through. The reduction word then has a form $a'_1 {b'_1}^{n-n_1} a'_2 \cdots a'_{k'} {b'_{k'}}^{n-n_{k'}} a'_{k'+1}$, $n_1, \ldots, n_{k'} \leq |a_1 \ldots a_{k+1}|$, see appendix for details. Thus for $n \geq |a_1 \ldots a_{k+1}|$, $|D_\beta^n(\alpha) \cap T|$ increases by a constant rate which has to be $i(\alpha, \beta)|\beta \cap T|$. ■

Corollary 4.3 *Geometric intersection number of two curves given by normal coordinates can be computed in polynomial time.*

A result analogous to Lemma 4.2 is true for lengths measure w.r.t. a pair of pants decomposition allowing us to compute geometric intersection numbers for curves given by Dehn-Thurston coordinates. This together with Lemma 3.2 gives the following result.

Corollary 4.4 *Geometric intersection number of two curves given by Dehn-Thurston coordinates can be computed in polynomial time.*

One component of the Dehn twist algorithm is computing a reduction of a compressed word in a free group. Let Γ be a set of generators of a free group and let $\Sigma = \Gamma \cup \Gamma^{-1}$ be an alphabet. A word over Σ is *reduced* if no $a, a^{-1}, a \in \Gamma$ are neighbors. The *reduction* of w is the unique word obtained by removing neighboring a, a^{-1} pairs until no such pairs remain.

Lemma 4.5 *Let w be a word over $\Sigma = \Gamma \cup \Gamma^{-1}$ given by an SLP P of length n. An SLP P' of the reduction w' of w can be computed by*

- *in time $O(n^3)$ by a randomized algorithm, P' has length $O(n^2)$; and*
- *in time $O(n^{10})$ by a deterministic algorithm.*

Note that the SLP equality testing problem can be reduced to the problem of computing a reduction of a word over $\Sigma = \Gamma \cup \Gamma^{-1}$ in linear time. Hence, unless there is an improvement in SLP equality testing, a randomized algorithm for the reduction problem will be faster. It might be that the reduction problem is easier for the intersection sequences since their equality can be tested in $O(n \cdot t)$ time (by transforming to normal coordinates).

Note that we can compute the cyclic reduction of a word w by computing reductions w', u' of w and $u = ww$ and then taking $w'[1+r...|w'|-r]$ where $r = |w'| - |u'|/2$. Hence we have the following observation.

Corollary 4.6 *Let w be a word over $\Sigma = \Gamma \cup \Gamma^{-1}$ given by an SLP P of length n. An SLP P' of the cyclic reduction w' of w can be computed by a randomized algorithm in time $O(n^3)$. The length of P' is $O(n^2)$.*

Let $\mathrm{SL}_2(K)$ be the group of unimodular 2×2 matrices over a commutative ring K. The modular group $\mathrm{PSL}_2(\mathbb{Z})$ is the factor group $\mathrm{SL}_2(\mathbb{Z})/(\pm I)$. There is a homeomorphism from $\mathrm{SL}_2(\mathbb{Z})/(\pm I)$ to $\mathrm{SL}_2(\mathbb{Z}/p\mathbb{Z})/(\pm I)$ obtained by reducing each entry modulo p (where p is a prime).

The modular group is a free product of finite groups generated by

$$S = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \text{and} \quad R = \begin{pmatrix} 0 & 1 \\ -1 & 1 \end{pmatrix}.$$

Lemma 4.7 ([Imr84]) *Let B_m be a set of elements*

$$S^{a_1}(R^{a_2}S)\dots(R^{a_m}S)RSR(SR^{-a_m})\dots(SR^{-a_2})S^{a_1},$$

where $a_1 \in \{0,1\}$ and $a_2,\dots,a_m \in \{-1,1\}$. Elements of B_m are free generators of a free subgroup of $\mathrm{PSL}_2(\mathbb{Z})$. Moreover any $X \in B_m$ has $||X||_2 \leq \phi^{2m}$ where $\phi = 1.618...$ is the golden ratio.

Proof (sketch) of Lemma 4.5
Pick a random prime $p, 2 \leq p \leq N$ where N will be specified later. To each symbol from Γ assign a matrix from the set B_m in Lemma 4.7. This defines a map $\psi : \Sigma^* \to \mathrm{PSL}_2(\mathbb{Z})$, a word w maps to 1 if and only if its reduction is the empty word.

Let $x_1,\dots,x_n$ be the variables in the straight line program P. In time $O(n)$ we can compute $\psi(x_i) \pmod p$ for $i = 1,\dots,n$.

Claim: Let $y_1,\dots,y_k$ be the reductions of $x_1,\dots,x_k$. Suppose that we know the lengths of the y_i. Let $a, 1 \leq a \leq |y_k|$. In time $O(k)$ we can compute $\psi(y_k[1\dots a]) \pmod p$ and a' such that $y_k[1\dots a]$ is the reduction of $x_k[1\dots a']$.

Proof: Let $y_k = y_i y_j$. If $a \leq |y_i|$ then we just need to compute $\psi(y_i[1\dots a]) \pmod p$ and a' such that $y_i[1\dots a]$ is the reduction of $x_i[1\dots a']$. If $a > |y_i|$ we compute $\psi(x_i)\psi(y_j[1\dots(a-|y_i|+c)]) \pmod p$, where $c = (|y_i|+|y_j|-|y_k|)/2$ is the length of "cancellation" between y_i and y_j. We also find a'' such that $y_j[1\dots(a-|y_i|+c)]$ is the reduction of $x_j[1\dots a'']$ and let $a' = |x_i| + a''$.

In both cases we spent only constant time and the index of the variable for which we need to recursively solve the subword problem decreased by at least 1. □

Suppose that we computed the lengths of the reductions $y_1,\dots,y_{k-1}$ of $x_1,\dots,x_{k-1}$. Let $x_k = x_i x_j$ be an assignment. Using binary search and the previous claim we can in time $O(n^2)$ find the length ℓ of the longest common prefix of y_j and y_i^{-1}. Now we can compute $|y_k| = |y_i| + |y_j| - 2\ell$.

After we computed lengths of the y_i, $i = 1,\dots,n$ we will generate the program for y_n. For k from n to 1 we do the following. Let $x_k = x_i x_j$ be an assignment. Let a' be such that y_i is the reduction of $x_i[1\dots a']$ and let b' be such that y_j is the reduction of $x_j[b'\dots|x_j|]$. In time $O(n)$ we can compute $a_1,\dots,a_m,b_1,\dots,b_{m'}$ such that $x_i[1\dots a'] = x_{a_1}\dots x_{a_m}$ and $x_j[b'\dots|x_j|] = x_{b_1}\dots x_{b_{m'}}$ where $m, m' \leq n$. We replace $x_k = x_i x_j$ by $x_n = x_{a_1}\dots x_{a_m} x_{b_1}\dots x_{b_{m'}}$. The resulting SLP has size $O(n^2)$ and it represents the reduction w' of w.

The algorithm executes $O(n^2)$ randomized identity tests. It will fail if for x that was tested $\psi(x) \neq 1$ but $\psi(x) \pmod p = 1$. Note that $||\psi(x)||_2 \leq |\Sigma|^{2\cdot 2^n}$ and hence for a random prime $\leq N$ for $N = O(n)$ the algorithm fails with probability $O(2^{-n})$.

Now we sketch a deterministic algorithm for the problem. The algorithm considers the assignments in the order they occur in the program and computes SLP for the reductions. Suppose that we have an SLP for $x_1,\dots,x_{k-1}$ of size m. Let $x_k = x_i x_j$ be an assignment. We can compute the length of the reduction between x_i and x_j in time $O(n \cdot m^4)$ using binary search and the deterministic equality testing algorithm. In time $O(n^2)$ we can compute $a_1,\dots,a_t,b_1,\dots b_{t'}$ such that the reduction of $x_i x_j$ is $x_{a_1} \dots x_{a_t} x_{b_1} \dots x_{b_{t'}}$ and $t, t' \le n$. We increased the size of the program by $O(n)$. Hence $m = O(n^2)$ and the total running time is $O(n^{10})$. ■

References

[AHT02] Ian Agol, Joel Hass, and William Thurston. 3-manifold knot genus is NP-complete. In *Proceedings of the 33th Annual ACM Symposium on Theory of Computing (STOC-2002)*, 2002.

[CLL+02] Moses Charikar, Eric Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, April Rasala, Amit Sahai, and Abhi Shelat. Approximating the smallest grammar: Kolmogorov complexity in natural models, 2002. In Proc. 29th Ann. Sympo. on Theory of Computing, 792-801.

[GKPR96] Leszek Gąsieniec, Marek Karpinski, Wojciech Plandowski, and Wojciech Rytter. Randomized efficient algorithms for compressed strings: The finger-print approach (extended abstract). In *Proceedings of the 7th Annual Symposium on Combinatorial*, pages 39–49, 1996.

[HJM96] Yoram Hirshfeld, Mark Jerrum, and Faron Moller. A polynomial algorithm for deciding bisimilarity of normed context-free processes. *Theoretical Computer Science*, 158(1-2):143–159, 1996.

[HLP99] Joel Hass, Jeffrey Lagarias, and Nicholas Pippenger. The computational complexity of knot and link problems. *Journal of ACM*, 46(2):185–211, 1999.

[HT97] Hessam Hamidi-Tehrani. *Algorithms in the Mapping Class Groups.* PhD thesis, Columbia, 1997.

[HTC96] Hessam Hamidi-Tehrani and Zong-He Chen. Surface diffeomorphisms via train-tracks. *Topology and its Applications*, 73:141–167, 1996.

[Imr84] Wilfried Imrich. Explicit construction of regular graphs without small cycles. *Combinatorica*, 4(1):53–59, 1984.

[Kne30] H. Kneser. Geschlossene Flächen in dreidimensionalen Mannigfaltigkeiten. *Jahresbericht der Deutschen Mathematikver-Vereinigung*, pages 248–260, 1930.

[Lic64] W. B. R. Lickorish. A finite set of generators for the homeotopy group of a 2-manifold. *Proc. Cambridge Philos. Soc.*, 60:769–778, 1964.

[Lic66] W. B. R. Lickorish. Corrigendum: On the homeotopy group of a 2-manifold. *Proc. Cambridge Philos. Soc.*, 62:679–681, 1966.

[Loh03] Markus Lohrey. *Computational and logical aspects of infinite monoids.* PhD thesis, Universitat Stuttgart, 2003. Habilitation.

[LS02] Eric Lehman and Abhi Shelat. Approximations algorithms for grammar-based compression, 2002. In Thirteenth Annual Symposium on Discrete Algorithms (SODA'02).

[MST97] M. Miyazaki, A. Shinohara, and M. Takeda. An improved pattern matching algorithm for strings in terms of straight-line programs. In A. Apostolico and J. Hein, editors, *Proceedings of the 8th Annual Symposium on Combinatorial Pattern Matching*, number 1264, pages 1–11, Aarhus, Denmark, 1997. Springer-Verlag, Berlin.

[Pen84] Robert C. Penner. The action of the mapping class group on curves in surfaces. *L'Enseignement Mathématique*, 30:39–55, 1984.

[Pla94] Wojciech Plandowski. Testing equivalence of morphisms on context-free languages. In *Algorithms—ESA '94 (Utrecht)*, volume 855 of *Lecture Notes in Computer Science*, pages 460–470. 1994.

[PR98] Wojciech Plandowski and Wojciech Rytter. Application of Lempel-Ziv encodings to the solution of words equations. In *Automata, Languages and Programming*, pages 731–742, 1998.

[RD99] John M. Robson and Volker Diekert. On quadratic word equations. *Lecture Notes in Computer Science*, 1563:217–226, 1999.

[Ryt02] Wojciech Rytter. Application of Lempel-Ziv factorization to the approximation of grammar-based compression, 2002. In Proceedings 13th Annnual Symposium Combinatorial Pattern Matching, 20-31, 2002.

[SŠ04] Marcus Schaefer and Daniel Štefankovič. Decidability of string graphs. *J. Comput. System Sci.*, 68(2):319–334, 2004.

[SSŠ03] Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Recognizing string graphs in NP. *J. Comput. System Sci.*, 67(2):365–380, 2003. Special issue on STOC2002 (Montreal, QC).

5 Appendix

Lemma 5.1 *Given a sequence*

$$w_n = a_1 b_1^n a_2 \cdots a_k b_k^n a_{k+1}$$

with words a_i, b_i over Γ, the corresponding reduced sequence has, essentially, the same form:

$$w'_n = a'_1 {b'_1}^{n-n_1} a'_2 \cdots a'_{k'} {b'_{k'}}^{n-n_{k'}} a'_{k'+1}.$$

Proof :
Let the weight of a sequence $(w_n)_{n\in\omega}$ be $|w_1|$. We prove the result by induction over the weight of the sequence. If the weight is 0, the sequence consists of the empty word which is reduced. Assume then, we are given a sequence $(w_n)_{n\in\omega}$. If a_{k+1} is not the empty word, then we remove the last letter x from each w_n to get a sequence v_n of smaller weight. By the inductive assumption, this sequence has a corresponding reduced sequence of the form

$$v'_n = a'_1 {b'_1}^{n-n_1} a'_2 \cdots a'_{k'} {b'_{k'}}^{n-n_{k'}} a'_{k'+1}.$$

But then w_n is represented by

$$v'_n x = a'_1 {b'_1}^{n-n_1} a'_2 \cdots a'_{k'} {b'_{k'}}^{n-n_{k'}} a'_{k'+1} x.$$

We can assume that $a'_{k'+1}$ is not empty, if needs be by increasing $n_{k'}$ by one, and letting $a'_{k'+1} = b'_{k'}$. If a'_{k+1} ends in a letter other than x^{-1}, then $v'_n x$ is reduced. If, on the other hand, a'_{k+1} ends in x^{-1}, we simply remove both the x^{-1} from $a'_{k'+1}$ and the x, and, again, obtain a reduced word.

Before we deal with the case that a_{k+1} is the empty word, let us consider a special case: words of the form $b_1^n a b_2^n$ (where a, b_1, b_2 are words over Γ). By what we have seen already, we can assume that $b_1^n a$ is reduced for all n, and so is b_2^n (by itself). Hence, the only cancellations in $b_1^n a b_2^n$ can occur at the border of a and b_2^n. If we choose n large enough such that $n|b_2| > |a|$, we can tell whether b_2^n cancels all of a. If not, cancellation stops at some point within a, and the reduced form is $b_1^n a' b_2^{n-m}$ for some $m \geq 0$. If b_2^n cancels all of a, we are left with a term of the form

$$b_1^n b b_2^n,$$

where b is a suffix of b_2. Cancellations can now occur at the border of b_1^n and b only. Consider $n = |b_1| + |b_2|$. There are two cases: the shorter term of b_1^n and bb_2^n is entirely cancelled by the other, or cancellation stops before all of the shorter term is cancelled. In the second case, we can perform all the cancellations, and obtain a reduced term of the form ${b'_1}^{n-n_1} b' {b'_2}^{n-n_2}$ representing $b_1^n a b_2^n$.

In the first case, conclude that the shorter of b_1^n and bb_2^n is a prefix of the involution of the other term. In other words, if we let u be the common prefix of b_1^n and $(b_2^{-1})^n b^{-1}$, then the length of u is the length of the shorter word. Now u obviously has periods $|b_1|$ and $|b_2|$, therefore, by a theorem of Fine and Wilf [**?**, Theorem 8.1.4], u has period $\gcd(|b_1|, |b_2|)$. We conclude that there is a word c such that $b_1 = c^{m_1}$, $b = (c^{-1})^{m_2}$, and $b_2 = (c^{-1})^{m_3}$. Using this, we obtain

$$b_1^n b b_2^n = c^{n(m_1-m_3)-m_2} = \begin{cases} (c^{m_1-m_3})^{n-m_2} & \text{if } m_1 \geq m_3 \\ (c^{m_3-m_1})^{n+m_2} & \text{else} \end{cases}.$$

We return to the case that a_{k+1} empty, namely that

$$w_n = a_1 b_1^n a_2 \cdots a_k b_k^n.$$

We consider $v_n = a_1 b_1^n a_2 \cdots a_k$, which is of smaller weight, and therefore is represented by a reduced sequence

$$v_n' = a_1' {b_1'}^{n-n_1} a_2' \cdots a_{k'}' {b_{k'}'}^{n-n_{k'}} a_{k'+1}'.$$

Using our considerations for the case of terms of the form $b_1^n b b_2^n$, we study ${b_{k'}'}^{n-n_{k'}} a_{k'+1}' {b_k}^n$. Cancellations either stop, in which case we are done, or we can rewrite this terms in the form c^{n-m}, at which point we recursively append this term to $a_1' {b_1'}^{n-n_1} a_2' \cdots a_{k'}'$. ■