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 Rochester
Rochester, New York, USA
stefanko@cs.rochester.edu

Abstract

Simple curves on surfaces are often represented as sequences of intersections with a triangulation. However, topologists have much more succinct ways of representing simple curves such as normal coordinates which are exponentially more succinct than intersection sequences.

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 normal coordinate representation. These are the first algorithms that solve these problems in time polynomial in the succinct representations.

As an application we show that a generalized notion of crossing number can be decided in **NP**, even though the drawings can have exponential complexity.

1 Introduction

In an earlier paper we started investigating algorithms for basic problems of computational topology [14]; we extend this work to deal with crossings of curves in surfaces which has applications to graph drawing.

One of the driving problems of computational topology, long before it acquired the name, has been the problem of recognizing the unknot. The story begins in 1930 with Kneser [9] who introduced the succinct *normal coordinate* representation for curves and surfaces. This led to the theory of normal surfaces which was used by Haken in 1961 to show that the unknot can be recognized by an algorithm. Haken's approach was pushed further by Hass, Lagarias, and Pippenger who exploited the succinctness of the representation to show that the unknot can be recognized in **NP** [8]. To this end they had to verify in polynomial time that a special type of normal surface is an essential disk, in particular, that it is connected. Agol, Hass, and Thurston [1] strengthened this result by showing 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 [14], that grew out of our work on the string graph recognition problem [15]. We showed how to compute connected components, count them, decide isotopy of curves, and compute the algebraic intersection number of two curves. The novel ingredient in our approach was that it was based on recent developments of algorithms over free monoids rather than groups.

In the current paper we continue the study of curves by showing how to efficiently perform Dehn twists, a fundamental topological operation. As a consequence we obtain an algorithm for computing the geometric intersection number of two curves. The following theorem summarizes our main results.

Theorem 1 *The tasks of performing a Dehn twist of a curve along another curve on an arbitrary surface, and computing the geometric intersection number of two curves on a surface with boundary can be solved in polynomial time in the normal coordinate representation (in a given triangulation).*

These results are very strong, since the normal coordinate representation is very succinct: the length of a curve is exponential in the size of its representation. As an application of Theorem 1 we show that a generalized notion of crossing number lies in **NP**, which unifies several non-trivial complexity results in graph drawing.

There has been previous work on calculating the effect of a Dehn twist. Penner [12] gave explicit formulas describing the action of the Lickorish generators on the Dehn-Thurston coordinates for 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 may be needed.

More recently Hamidi-Tehrani and Chen [7] gave an algorithm to compute the action of a set of generators on the space $\mathrm{CS}(M)$ given by measured π-train tracks, but its running time is exponential in the representation.

Hamidi-Tehrani's thesis [6] 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 Surfaces, Curves, and Words

2.1 Representations of Surfaces and Simple Curves

By a *surface* M we mean a connected, compact, orientable 2-manifold with boundary ∂M.[1] A *curve* in M is the image of $[0,1]$ under some continuous function f; it is *simple* if f is injective with the possible exception of $f(0) = f(1)$. Intuitively, the curve is simple if it has no self-intersections; a curve is *closed* if it has no endpoints, that is, $f(0) = f(1)$. A closed curve is *trivial* if it can be contracted to a point (null-homotopic) or a boundary component. We call a curve a *properly embedded simple arc* if it is a simple curve with both of its endpoints on ∂M.

A simple *multi-curve* α in M is the disjoint union of any number of non-trivial simple closed curves and properly embedded arcs.[2] Call a simple multi-curve *closed* if all its connected components are closed. By *isotopy* we mean isotopy rel boundary, i. e., a continuous deformation which leaves ∂M fixed and does not introduce self-intersections. Let $\mathrm{CS}(M)$ be the set of all isotopy classes of simple multi-curves. Let $\mathrm{CS}_0(M) \subseteq \mathrm{CS}(M)$ 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 (isotopy classes of) simple multi-curves $\alpha, \beta \in \mathrm{CS}(M)$ is the minimal number of intersections between any of their representatives.

A *triangulation* T of a surface M is a set of points V in M and an 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 and each boundary component contains exactly one vertex of V.

Let M be given by a triangulation. A simple multi-curve γ is *normal* w.r.t. T if all 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 e. Any simple multi-curve can be made normal by simple redrawing moves, so for any simple multi-curve there is a normal multi-curve isotopic to it. Normal curves are very well-behaved with respect to the triangulation; in particular, given a triangle a, b, c the curve segments within the triangle fall into three types: segments crossing from ab to ac, from ab to bc and from ac to bc.

Moreover, the number of segments of each type is determined by the number of intersections of the multi-curve with ab, bc and ac: for example there are $(|\gamma \cap ab| + |\gamma \cap ac| - |\gamma \cap bc|)/2$ segments crossing from ab to ac. This allows us to describe a normal curve γ by its *normal coordinates*, which is the vector $(|\gamma \cap e|)_{e \in T}$. The *complexity* of the normal coordinates is the number of bits needed to encode the vector $(|\gamma \cap e|)_{e \in T}$, by writing each coordinate in binary. Any two simple multi-curves with the same normal coordinates are isotopic if they agree on the boundary. If the triangulation is minimal then the converse is true—any isotopic curves have the same normal coordinates. (Note that only surfaces with non-empty boundary have minimal triangulations.)

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 by 360 degrees and gluing the two copies back together. More precisely, if an annular neighborhood around α is parameterized by $\{(x, \varphi), 1 \le x \le 2, 0 \le \varphi \le 2\pi\}$ then D_α is $(x, \varphi) \mapsto (x, \varphi + 2\pi(x-1) \bmod 2\pi)$ on the annulus and the identity elsewhere. If α is a simple closed multi-curve, we define D_α to be the composition of $D_{\alpha'}$ for all connected components α' of α (note that all α' are simple closed curves).

2.2 Quadratic Word Equations and Compressed Representation of Words

Let Σ be an *alphabet*. A word in Σ^* can be represented by a *straight-line program* (SLP), which is a sequence of assignments $X_i := \text{EXPR}$, $i = 1, \ldots, n$, where EXPR is a either a symbol from Σ or $X_j X_k, 1 \le j, k < i$. The *length* n of an SLP representing a word w can be exponentially smaller than $|w|$, the length of w.

Example 2 *Over the alphabet $\{a, b\}$ the SLP $X_1 = a$, $X_2 = b$, $X_3 = X_1X_2$, $X_4 = X_3X_3$, $\cdots$, $X_n = X_{n-1}X_{n-1}$ of length n represents the word $(ab)^{2^{n-3}}$.*

Equality of two words given by SLPs of lengths m and n can be tested in deterministic time $O(m^2n^2)$ [11] and in randomized time $O(m+n)$ [5].

A *word equation with specified lengths* is an equation over a free monoid in which every variable has to be replaced with a word of a specified length.

Example 3 *The word equation $XabY = YbaX$ in variables X and Y has two shortest solutions: $X = \epsilon, Y = a$ and $X = b, Y = \epsilon$, where ϵ is the empty word. If we require $|X| = 4$ and $|Y| = 2$, the equation has no solution. However, if we require $|X| = 3$ and $|Y| = 1$, then $X = aba$ and $Y = a$ is a solution.*

[1] In other words, each point in the manifold has a neighborhood homeomorphic to a disk or a half-disk (if it is on the boundary).

[2] Formally, it is a proper 1-dimensional submanifold of M such that no component of α is null-homotopic or homotopic to the boundary.

Finding a solution of a word equation in general is **NP**-hard. However, the word equations arising in our context all have specified lengths and are *quadratic*: every variable occurs at most twice in the equation. For quadratic word equations a faster and simple algorithm is known which also guarantees that the solution can be expressed as an SLP.

Theorem 4 (Robson, Diekert [13]) *The solvability of a quadratic word equation with specified lengths can be decided in time linear in the complexity of the equation. If there exists a solution, a linear-size SLP for any subword of the solution can be found in linear time.*

2.3 Representations of Simple Curves

We have seen that simple curves can be succinctly represented using normal coordinates, assuming they have been normalized with respect to some triangulation T. Another natural way of representing such a simple curve would be by listing the order in which it crosses the edges of the triangulation: Arbitrarily fix an orientation $\vec{e}$ of each edge $e \in T$. Given an oriented simple curve γ let w be a word obtained by traversing 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 an *intersection sequence of γ with the triangulation*. (Note that for closed curves the intersection sequence is only determined up to cyclic permutation.)

An intersection sequence can be exponentially long compared to the normal coordinates of a simple curve, but, as it turns out, intersection sequences are highly compressible since they can be obtained as solutions of quadratic word equations and can therefore be represented by SLPs. Moreover, moving from an SLP representation of an intersection sequence to normal coordinates is simple: we just need to count the number of occurrences of each symbol in the compressed word, a task that can be performed in time $O(n)$ for each symbol, see [4]. In short, normal coordinate representation and SLP representation of intersection sequences are polynomially equivalent.

Theorem 5 *Let M be a surface given by a triangulation T. Let $\alpha \in \mathrm{CS}(M)$ be a simple curve in M. If α is given by an intersection sequence with T encoded by an SLP of length n, then the normal coordinates of α can be computed in time $O(n \cdot |T|)$. If α is given by normal coordinates in T with complexity n, then an SLP of size $O(n)$ for an intersection sequence of α with T can be obtained in time $O(n)$.*

We omit the proof of this result, the interesting direction (from normal coordinates to an SLP for an intersection sequence), follows techniques suggested in [14]. Note that while normal coordinates can encode multi-curves (which can have many connected components), intersection sequences are restricted to simple curves (which have a single component).

3 Dehn Twists and Applications

3.1 Computing Dehn Twists

The Dehn twist of a curve given by normal coordinates can be computed in time polynomial in the size of the representation. Our algorithm works for all surfaces (that is, with or without boundary).

Theorem 6 *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 n. The normal coordinates of a representative 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 $O(g \cdot n^9)$ by a deterministic algorithm.*

We have to omit the proof, but we can give a rough outline: we first prove the result for a simple closed curve β. Performing a Dehn twist along a simple closed curve α can be captured by a quadratic word equation and therefore $D_\beta(\alpha)$ can be represented by a compressed SLP. There is some subtlety here, since the word obtained as a solution of the quadratic word equation may not correspond to a normalized simple curve; however, normalization can be performed on the word (it corresponds to cancellation over a free monoid with inverses); this step is computationally expensive, since it requires iterated equality testing. The result can then be generalized to simple closed multi-curves α; this might require performing an exponential number of Dehn twists, but this can be done "in parallel" using word equations.

3.2 Computing Geometric Intersection Numbers

For surfaces with a boundary we can compute geometric intersection numbers in polynomial time once we can compute Dehn twists in polynomial time. The reason is that the complexity of Dehn twists is closely related to the geometric intersection number: Considering representatives of β and γ which intersect minimally shows that for any simple curve α

$$|D_\gamma^n(\beta) \cap \alpha| \leq |\beta \cap \alpha| + n \cdot i(\beta,\gamma)|\gamma \cap \alpha|,$$

where D_γ^n is the n-fold application of D_γ. Hence, $D_\gamma^n(\beta)$ grows by a rate of at most $i(\beta,\gamma)|\gamma \cap \alpha|$ in n. For sufficiently large n, it grows exactly at that rate:

Lemma 7 *Let M be a surface and α, β, γ simple curves on M with γ closed. Let $n = 2i(\alpha,\beta)$. Then*

$$i(\gamma,\beta) = \frac{i(\alpha, D_\gamma^{n+1}(\beta)) - i(\alpha, D_\gamma^n(\beta))}{i(\alpha,\gamma)}.$$

We omit the proof which is based on results by Fathi, Laudenbach, Poénaru [3] and Luo [10]. Lemma 7 is the core observation needed to establish the main result:

Theorem 8 *The geometric intersection number of two curves given by normal coordinates on a surface with boundary can be computed in polynomial time.*

3.3 Generalized Crossing Number

We can apply our algorithm for calculating the geometric intersection number of two curves to a notoriously intractable graph drawing problem: the crossing number. The *crossing number*, $\mathrm{cr}(G)$, of a graph $G = (V, E)$ is the smallest number of intersections in a drawing of G in the plane (making certain standard assumptions about the drawing). Given a weight function $w : E^2 \to \mathbb{N}$, we can define a generalization, $\mathrm{cr}_w(G)$ of the crossing number as the minimum value of

$$\sum_{e,f \in E} i_D(e, f) \cdot w(e, f)$$

over all drawings D of G in the plane. For $w(e, f) = 1$ we obtain the usual crossing number.

Theorem 9 *Deciding whether* $\mathrm{cr}_w(G) \leq k$ *lies in* **NP** *for any polynomial-time computable weight function* w.

The generalized crossing number is a powerful modeling tool. For example, the weak realizability problem is a special case, and, therefore, lies in **NP**. This, in turn implies that the string graph problem, topological inference, and several other problems also lie in **NP**, see [15]). As another application, we consider the recently introduced simultaneous graph drawing model SCM^+ introduced by Chimani, Jünger, and Schulz [2]: given two planar graphs on the same vertex set, a *simultaneous drawing with fixed edges* is a drawing in which each graph by itself is planar and shared edges are drawn identically. Now, if we relax the condition that the two graphs be planar, we get the *simultaneous crossing number*, which is the smallest number of crossings in a drawing of the union of the two graphs that occur between edges of the same graph. If, in addition, we require that the total number of crossings be minimized, we get the SCM^+ model. It is easy to see that this problem is a special case of our generalized crossing number problem.

References

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

[2] M. Chimani, M. Jünger, and M. Schulz. Crossing minimization meets simultaneous drawing. Technical report, Zentrum für Angewandte Informatik Köln, Lehrstuhl Jünger, May 2007.

[3] A. Fathi, F. Laudenbach, and V. Poénaru. Travaux de Thurston sur les surfaces. *Astérisque*, 66–67, 1979.

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

[5] L. Gąsieniec, M. Karpinski, W. Plandowski, and W. Rytter. Randomized efficient algorithms for compressed strings: The finger-print approach. In *LNCS 1075*, pages 39–49, 1996.

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

[7] H. Hamidi-Tehrani and Z.-H. Chen. Surface diffeomorphisms via train-tracks. *Topology and its Applications*, 73:141–167, 1996.

[8] J. Hass, J. Lagarias, and N. Pippenger. The computational complexity of knot and link problems. *Journal of ACM*, 46(2):185–211, 1999.

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

[10] F. Luo. Some applications of a multiplicative structure on simple loops in surfaces. In *Knots, braids, and mapping class groups*, volume 24, pages 123–129. Amer. Math. Soc., Providence, RI, 2001.

[11] M. Miyazaki, A. Shinohara, and M. Takeda. An improved pattern matching algorithm for strings in terms of straight-line programs. In *LNCS 1264*.

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

[13] J. M. Robson and V. Diekert. On quadratic word equations. *LNCS*, 1563:217–226, 1999.

[14] M. Schaefer, E. Sedgwick, and D. Stefankovic. Algorithms for normal curves and surfaces. In *COCOON '02*, pages 370–380, London, UK, 2002. Springer-Verlag.

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