Compressed Data Structures for Heegaard Splitting

Henrique Ennes
Université Côte d'Azur, Nice, France
INRIA d'Université Côte d'Azur, Sophia Antipolis, France

Clément Maria
INRIA d'Université Côte d'Azur, Sophia Antipolis, France

Abstract

Heegaard splittings provide a natural representation of closed 3-manifolds by gluing handlebodies along a common surface. These splittings can be equivalently given by two finite sets of meridians lying on the surface, which define a Heegaard diagram. We present a data structure to effectively represent Heegaard diagrams as normal curves with respect to triangulations of a surface of complexity measured by the space required to express the normal coordinates' vectors in binary. This structure can be significantly more compressed than triangulations of 3-manifolds, giving exponential gains for some families. Even with this succinct definition of complexity, we establish polynomial-time algorithms for comparing and manipulating diagrams, performing stabilizations, detecting trivial stabilizations and reductions, and computing topological invariants of the underlying manifolds, such as their fundamental and homology groups. We also contrast early implementations of our techniques with standard software programs for 3-manifolds, achieving faster algorithms for the average cases and exponential gains in speed for some particular presentations of the inputs.

2012 ACM Subject Classification Theory of computation → Computational geometry; Mathematics of computing → Geometric topology; Theory of computation → Data structures design and analysis

Keywords and phrases 3-manifold, Heegaard splitting, curves on surfaces, surface theory, data structure, computational topology

Digital Object Identifier 10.4230/LIPIcs.SoCG.2026.42

Related Version *Full Version*: `https://arxiv.org/abs/2507.11406`

Supplementary Material *Software (Source Code)*: `https://doi.org/10.5281/zenodo.19236220`

Funding This work has been partially supported by the ANR project ANR-20-CE48-0007 (AlgoKnot) and the project ANR-15-IDEX-0001 (UCA JEDI).
Henrique Ennes: The PhD thesis, which this work is a part of, has been supported by the French government, through the France 2030 investment plan managed by the Agence Nationale de la Recherche, as part of the "UCA DS4H" project, reference ANR-17-EURE-0004.

1 Introduction

Since the early days of computational topology, *3-manifolds* have attracted researchers' keen interest [27, 31, 40] for lying exactly midway between the well-understood surfaces and the wildly behaved 4-manifolds. For example, several 3-manifold problems were related to important questions in complexity theory: S^3 recognition was proved by Schleimer to be in NP [50], whereas approximations of some quantum invariants are firmly tied to quantum computing and are known to be #P-hard [1, 2]. Similarly, the problems of classifying [17, 39] and manipulating the geometry and topology of these spaces [8, 42] have also received significant theoretical and practical attention, with implementations of different methods publicly available in the most important software in the field, `Regina` [12] and `SnapPy` [18].

The algorithmic point of view is particularly relevant for the construction of *censuses* of homomorphism classes of 3-manifolds [7, 9, 10, 13]. Drawing inspiration from the related problem of knot classification (this time, up to ambient isotopies), where the cumulative work

© Henrique Ennes and Clément Maria;
licensed under Creative Commons License CC-BY 4.0

of researchers [11, 15, 21] has culminated in the widely accessible census of $\sim 350'000'000$ prime knot diagrams, computational methods are expected to foster the elaboration of tables of 3-manifolds, particularly valuable assets in the proposition and verification of conjectures, as well as in the discovery of counterexamples.

Abstracting modern approaches [6, 9, 11, 13] of the construction of censuses of topological spaces, one requires (1) compact data structures to represent massive amounts of (candidate) spaces and the iterative construction of all spaces up to a maximum complexity; (2) efficient operations to disqualify inputs (such as composite knots, links, non-prime 3-manifolds) and simplify the representation combinatorics (Reidemeister moves, Pachner moves, etc); and, (3) methods for discriminating spaces, such as invariants. For 3-manifolds, these steps are usually addressed through *triangulations* – homeomorphic copies described by a finite set of tetrahedra and gluing rules among their faces – of complexity often measured by the number of tetrahedra. On the other hand, theoretical bounds on the minimal number of tetrahedra to represent some spaces [19, 33, 34, 44] and the difficulty of generating particular structures, such as hyperbolicity [44], can limit the effectiveness of triangulation-based censuses.

Heegaard splittings emerge as an alternative to triangulations. They depict a closed 3-manifold M as the result of gluing two better-behaved pieces called *handlebodies* along a common boundary. Splittings can also be combinatorially described by *Heegaard diagrams*, two sets of curves in a surface whose intersection pattern conveys a lot of topological information about M. We suggest here a new data structure to represent Heegaard diagrams based on *normal coordinates*, of a size as large as *binary* of the number of intersections between the two sets of curves. The gain in measuring the complexity of 3-manifolds through this notion is not to be overlooked: we present in Section 5 families of manifolds whose diagrams' complexity grows linearly with the size of the input, but whose triangulations necessarily use exponentially many tetrahedra. However, what is more important is that due to the proliferation of efficient operations in normal coordinates [24, 41, 48, 49, 53], we can establish polynomial-time algorithms even in this compressed notion of complexity, leading to basic manipulations of Heegaard splittings and the computation of simple 3-manifold invariants such as the fundamental and homology groups. Moreover, as we point out in Remark 9, some important complexity results about 3-manifolds – such as sphere (co-)recognition – are still preserved if we assume compressed inputs, which tells us much about the inherent difficulty of these problems. We hope that these techniques are useful for the study and classification of topological spaces that are beyond the reach of efficiency when represented by other methods.

The paper begins with a background section on both low-dimensional and computational topology (Section 2), where we review some of the concepts and results that will be needed in the later parts. Our contribution starts in Section 3, where we formally present our data structure for Heegaard splittings. Section 4 establishes basic algorithms in this structure, which are further expanded the paper's full version. We compare early implementations of these algorithms with triangulation-dependent methods in Section 5.

Before we close this introduction, we draw the reader's attention to two subtleties of the current work. First, although most of our tools apply to *generalized* Heegaard splittings defined for *orientable compact* 3-manifolds, potentially with boundary [37, 51], we will focus solely on *closed* 3-manifolds. Therefore, all 3-manifolds here will be assumed orientable and closed, unless explicitly stated otherwise. Second, we assume throughout a *unit-cost RAM model* with some bit-size W [26]. This means that the four integer operations and integer comparison can be executed in constant time, provided that the numbers involved have binary representations of size at most W.

2 Background material

2.1 Background on low-dimensional topology

We assume the reader is acquainted with the basic notions of surface theory at the level of [25], referring to the paper's full version for details. We use Σ to denote a general surface and Σ_g the (unique) closed surface of genus g. Whenever possible, we reserve Greek letters for multicurves and Latin letters for curves and arcs. All (multi)curves are assumed simple. As usual, we do not differentiate (multi)curves from their isotopy classes if no confusion arises.

The *mapping class group* of the surface Σ_g, $\mathrm{Mod}(\Sigma_g)$, is the group of orientation-preserving $\Sigma_g \to \Sigma_g$ homeomorphisms up to isotopy. For each g, $\mathrm{Mod}(\Sigma_g)$ is finitely generated as a group by an alphabet $\mathcal{L}$, consisting of *Dehn twists* about $3g-1$ (isotopy classes of) canonically defined essential curves $s_1, \ldots, s_{3g-1}$ [43]. Formally, a Dehn twist $\tau_s : \Sigma_g \to \Sigma_g$ about s is given by cutting out a cylindrical regular neighborhood around the curve, applying a 2π left-twist to one of the cylinder's boundaries, and gluing the cylinder back into the surface. We call the curves $s_1, \ldots, s_{3g-1}$ the surface's *Lickorish generators*. Hopefully without confusion, we also call the Dehn twists in the alphabet $\mathcal{L} = \{\tau_{s_1}, \ldots, \tau_{s_{3g-1}}\}$ Lickorish generators.

An essential multicurve γ in Σ_g is a *system* if Σ_g cut along γ, $\Sigma_g \backslash \gamma$, is a $2g$ punctured sphere. A system γ in Σ_g can be used to construct the genus g *handlebody* $\mathcal{H}_\gamma$ where we attach 2-handles along the curves γ in $\Sigma \times \{0\}$ and fill any spherical boundary component with 3-handles. Note that $\partial \mathcal{H}_\gamma = \Sigma_g$ and that $\mathcal{H}_\gamma$ is homeomorphic to the direct sum of g solid tori. Components of γ are *meridians* of $\mathcal{H}_\gamma$ in the sense that they bound disks in the handlebody's interior. Two systems γ and γ' in Σ_g will be called *equivalent* if γ' is a set of meridians in $\mathcal{H}_\gamma$ (which implies that γ is a set of meridians in $\mathcal{H}_{\gamma'}$).

Let c_1, c_2 be two components of a system γ and p an arc from a puncture corresponding to c_1 to a puncture corresponding to c_2 in $\Sigma_g \backslash \gamma$. The *band sum* of c_1, c_2 and p is the curve c in $\Sigma_g \backslash \gamma$ obtained by the surgery of Figure 1. One can show that c is a meridian in $\mathcal{H}_\gamma$ [22, 37], so $\gamma' = \gamma - \{c_1\} \cup \{c\}$ is a system equivalent to γ. We call this whole operation a *disk slide*. With finitely many applications of disk slides and isotopies, a system γ can be transformed into any equivalent system γ' [37].

A genus g *Heegaard splitting* is the closed 3-manifold resulting from the gluing of two handlebodies with boundaries homeomorphic to Σ_g through a gluing map $f : \Sigma_g \to \Sigma_g$. In fact, one might assume that f is given by a $\phi \in \mathrm{Mod}(\Sigma_g)$ composed with some canonical orientation-reversing homeomorphism. This means that a Heegaard splitting can be represented by a pair (Σ_g, ϕ) where ϕ is a word on Lickorish generators. We call this pair the *Heegaard word* for the splitting. Every closed 3-manifold has a Heegaard splitting [46, 47].

Note that if we fix a system of meridians α on one of the two handlebodies, $\mathcal{H}_\alpha$, then $\beta = \phi(\alpha)$ is a system of meridians of the *other* handlebody, $\mathcal{H}_\beta$. We call the tuple $(\Sigma_g, \alpha, \beta)$ a *Heegaard diagram*. Heegaard diagrams are algorithmically independent of words: any two systems α and β will define the same splitting, provided that the curves are meridians in their respective handlebodies. Diagrams are therefore not unique for a given splitting. For example, if β and β' are *isotopic diagrams* on Σ_g – that is, β and β' are isotopic multicurves in Σ_g – then $(\Sigma_g, \alpha, \beta)$ and $(\Sigma_g, \alpha, \beta')$ define the same manifold. Similarly, if β is equivalent to β', we say that the diagrams are *equivalent* and $(\Sigma_g, \alpha, \beta)$ and $(\Sigma_g, \alpha, \beta')$ also define the same 3-manifold. In this sense, disk slides play for Heegaard diagrams a role similar to the Reidemeister moves for knots.

A splitting $(\Sigma_g, \mathcal{H}_\alpha, \mathcal{H}_\beta)$ is *reducible* if there is a curve c in Σ_g that is a meridian of both $\mathcal{H}_\alpha$ and $\mathcal{H}_\beta$. We call a diagram $(\Sigma_g, \alpha, \beta)$ *reducible* if either (1) there is a component $a \in \alpha$ and a component $b \in \beta$ that are isotopic in Σ_g; or (2) there is a separating curve s, disjoint

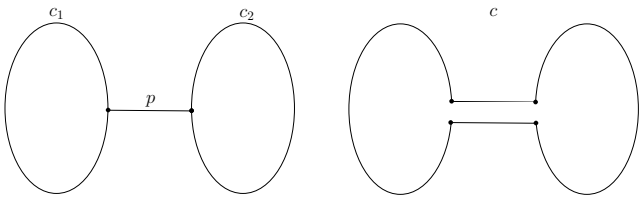

Figure 1 The result c of a band sum of c_1, c_2, and p.

from both α and β, and such that $\Sigma_g \backslash s$ defines a connected sum of two Heegaard splittings of smaller genera. The uniqueness of fundamental group decomposition [3] and [51][Theorem 6.3.5] imply that a reducible diagram represents a reducible splitting. A reducible splitting can always be written as the connected sum of sub-splittings [3, 37].

A presentation $\langle x_1, \ldots, x_g | r_1, \ldots, r_g \rangle$ of the *fundamental group* $\pi_1(M)$ can be obtained as follows [30, 37]: label the curves $a_1, \ldots, a_g \in \alpha$ and $b_1, \ldots, b_g \in \beta$ and give them an orientation; then traverse each curve b_i along the assumed orientation and, whenever it intersects some a_j, we append x_j to the relation r_i if the index of the intersection is positive, otherwise we append x_j^{-1} to it. Because the *first (integral) homology group of* M, $H_1(M)$ is the abelianization of $\pi_1(M)$ [28], we can present $H_1(M)$ as $\langle x_1, \ldots, x_g | r'_1, \ldots, r'_g, [x_i, x_j] \rangle$, where each $r'_i = x_1^{k_i^1} \ldots x_g^{k_i^g}$ can be computed by commuting the generators x_j in the relations r_i. Note that k_i^j equals the algebraic intersection number $\hat{i}(b_i, a_j)$ and we store this information in a $g \times g$ *presentation matrix* for $H_1(M)$, whose Smith normal form carries all the necessary information of the homology group.

2.2 Background on computational topology

Heegaard diagrams consist of three essential components that any candidate data structure must encode: a surface, Σ_g, and two multicurves, α and β. Starting with the surface, a graph $T = (V, E)$ embedded in Σ is a *cellular embedding* if the manifold defined by cutting Σ along T, $\Sigma \backslash T$, is homeomorphic to a collection of disks. We call each component of $\Sigma \backslash T$ a *face* of T. A cellular embedding T in the surface Σ is said to be *oriented* if we assign to each face an ordered list of its vertices, consistent with the orientation of Σ as in the right-hand rule. We assume every cellular embedding to be oriented. A cellular embedding T is a *(generalized) triangulation* of Σ if exactly three edges of T bound each face. We define the size of a triangulation, $|T|$, as the number of faces; note that $|V|, |E| = O(|T|)$ where $|V|$ is the number of vertices of the triangulation and $|E|$ is the number of edges. Surface triangulations are not to be confused with the similar concept of 3-manifold triangulations.

Every oriented surface can be described as a data structure by a triangulation. For the triangulations themselves, we assume that they are given as a set of vertices, edges, and triangles, with bidirectional pointers between vertices and edges, and between edges and the triangles, therefore requiring a total space of $O(|T|)$. In particular, determining whether two triangles are adjacent, getting the incident vertices and edges of a triangle, and finding triangles bounded by a specific edge take constant time. Other, more compact data structures for triangulation, such as CGAL's [20], can be converted into our structure in time $O(|T|)$ and at a similar cost to memory. More general cellular embeddings are treated similarly.

A curve c is *edged* with respect to T if it is disjoint from its faces. We represent an edged curve as a data structure by the list $E_T(c) = \{e \in E : |c \cap e| = e\}$, assumed to be ordered according to an implicit traversing orientation of c. Similarly, a multicurve γ can be represented by a set of $\#\gamma$ edge lists, one for each component, which we denote as $E_T(\gamma)$. We

assume that no two components can share an edge, nor can the same component be incident to a vertex more than once; nevertheless, two distinct components can share a vertex. The *complexity* of $E_T(\gamma)$ is $\|E_T(\gamma)\| = \sum_{c\in\gamma} |E_T(c)|$.

On the other hand, a multicurve γ is called *standard* in T if it intersects the graph only at the edges and transversely. If γ is standard, we represent it as a collection of words in a finite alphabet: arbitrarily orient the edges E and take $E^{\pm} = E \cup E^{-1}$ for the letters. Then for each component c of γ, the word $I_T(c)$ in $(E^{\pm})^*$ is given by traversing c along some arbitrary direction and, whenever we intersect an edge $e \in E$, we append e^i to $I_T(c)$ where i is the index of the intersection. An *intersection sequence* representation for γ, $I_T(\gamma)$, is the set $\{I_T(c)\}_{c\in\gamma}$. While $E_T(\gamma)$ is well defined at the multicurve level, intersection sequences are defined only up to isotopies inside the cellular embedding's faces. Moreover, cyclic permutations and taking inverses of the words $I_T(c)$ in $I_T(\gamma)$ define the same isotopy classes.

Instead of working with explicit intersection sequences, we will often assume *straight line programs* (*SLPs*) to express $I_T(c)$. Formally, an SLP on the alphabet $E^{\pm}$ is a finite sequence of equations $[x_1 = \mathrm{EXPR}_1, \ldots, x_\ell = \mathrm{EXPR}_\ell]$, where the left-hand side, x_i, are symbols not in $E^{\pm}$ called *variables*, and the expressions on the right-hand side, $\mathrm{EXPR}_i \in (E^{\pm} \cup \{x_1, \ldots, x_{i-1}\})^*$, are called *assignments*. Each assignment EXPR_i is assumed to be a symbol from $E^{\pm}$, in which case the assignment is called *simple*, or of the form $\mathrm{EXPR}_i = x_{j_1}^{\pm 1} \cdot x_{j_2}^{\pm 1} \cdot \cdots \cdot x_{j_k}^{\pm 1}$, where $1 \le j_1, \ldots, j_k < i$, in which case we say the assignment is *proper*. In particular, the SLP $[x_1, \ldots, x_\ell]$ represents a word w in $(E^{\pm})^*$ if recursively substituting the appropriate x_j in each proper assignment, gives x_ℓ equal to w. We call ℓ the *length* of the SLP and define its *complexity* $m = \sum_{i=1}^{\ell} |\mathrm{EXPR}_i|$, where $|\mathrm{EXPR}_i|$ is the number of variables and symbols in the assignment $|\mathrm{EXPR}_i|$. Given any word $e_1 \ldots e_n$ in $(E^{\pm})^*$, one can trivially build an SLP of complexity $2n$, but there are instances in which SLPs can be used to give exponential gains over uncompressed representation of words [49]. Nonetheless, even in extremely compressed cases, several useful operations can still be efficiently performed, as stated in the following lemma, proved in the paper's full version. Here and throughout, we assume that $\lg k$ is the number of bits necessary to represent the potentially signed integer k; in particular, $\lg k = \log k + 2$ if k is signed and $\log k + 1$ if unsigned.

▶ **Lemma 1.** *Let $\langle x_1, \ldots, x_\ell \rangle$ be an SLP of complexity m representing a word $w \in (E^{\pm})^*$. In a unit-cost RAM model of bit-size $m + \lg k$, for a fixed $k \in \mathbb{Z}^+$, one can compute*

(a) *an SLP of w^{-1} in time $O(1)$;*

(b) *an SLP of w^k in time $O(\lg k)$;*

(c) *the number of occurrences of a symbol $e \in E^{\pm}$ in w, $e(w)$, in time $O(m)$.*

We will assume that every intersection sequence $I_T(c)$ is given by an SLP of complexity $\|I_T(c)\|$ and let $\|I_T(\gamma)\| = \sum_{c\in\gamma} \|I_T(c)\|$.

Now, suppose that $T = (V, E)$ is a triangulation. A standard (multi)curve γ is *normal* if it intersects each face of T in *elementary arcs* that connect distinct sides of the triangle in which they are contained. Equivalently, γ is normal if each (uncompressed) intersection sequence in $I_T(\gamma)$ is *cyclic reduced*, that is, if there are no substrings of type ee^{-1} or $e^{-1}e$ in any of the cyclic permutations of the components' sequences. When the curve c is normal for a triangulation, its isotopy class is fixed by the number of times it intersects each labeled edge [48], implying that we can describe it through a vector in $N_T(c)$ in $\mathbb{N}^{|E|}$. We call $N_T(c)$ the curve's *normal coordinates*, whose complexity, $\|N_T(c)\|$, is the number of bits necessary to store this vector, i.e., $\sum_{e\in E} \lg e(c) = \sum_{e\in E} (\log e(c) + 1)$. Naturally, we define $N_T(\gamma)$ as the set $\{N_T(c)\}_{c\in\gamma}$.

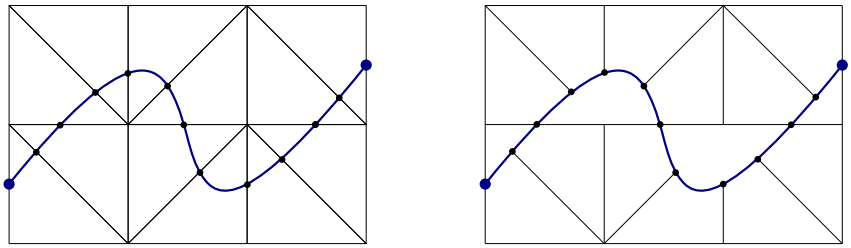

Figure 2 Left: the overlay graph of a normal arc (blue) with respect to a triangulation (black). Right: its street complex.

▶ Remark 2. We could represent a normal multicurve γ with a *single* vector $\sum_{c\in\gamma} N_T(\gamma)$. Similarly, it is possible (and common) to extend the notion of SLPs to encode more than one uncompressed word, which, in our case, implies giving all components of γ as a single SLP. Although these approaches are more parsimonious, they are less useful for our needs. Moreover, Lemma 5.1 of [24] gives an algorithm to transform, in time $O(|T|\|N_T(\gamma)\|)$, the normal coordinates of a multicurve given by a sum of its components' coordinates to our expected form.

Any normal (multi)curve γ partitions the edges of T into segments called *ports*. The *overlay graph* of γ is defined by having as vertices the union of V and the intersections $E \cap \gamma$, and as edges the union of the ports and the elementary arcs of γ (see Figure 2). The overlay graph splits each triangle t of the original triangulation into two types of regions: *junctions*, which are adjacent to three edges of T, and *blocks*, which are adjacent to only two. A port of the overlay graph is called *redundant* if it separates blocks. Following Erickson and Nayyeri [24], we define the *street complex* of γ, $S(T,\gamma)$, by deleting redundant ports and their endpoints from the overlay graph. In practice, the street complex merges blocks into longer faces called *streets*; naturally, $S(T,\gamma)$ is a cellular embedding in the surface triangulated by T, see Figure 2. We define the *crossing sequence* of a street in $S(T,\gamma)$ as the intersection sequence of a curve that crosses the street from one end to the other with respect to the original triangulation T. For simplicity, we assume that any port that separates two junctions is a degenerate street with a crossing sequence of length 1. The street complexes of all multicurves in this paper will have complexity $O(|T|)$ [24].

3 Data structure for Heegaard diagrams

Normal coordinates suggest a very natural encoding of Heegaard diagrams and, consequently, of closed 3-manifolds. Explicitly, we propose representing $(\Sigma_g, \alpha, \beta)$ by a tuple $(T, E_T(\alpha), N_T(\beta))$, where $T = (V, E)$ is a triangulation, $E_T(\alpha)$ is an edge representation of α, and $N_T(\beta)$ is a normal coordinate representation of β. We will often refer to the pair $(T, E_T(\alpha))$ as a *marked triangulation* of Σ_g, in which context $N_T(\beta)$ will be called the diagram itself. The *complexity* of the diagram is

$$\|(T, E_T(\alpha), N_T(\beta))\| = |T| + \|E_T(\alpha)\| + \|N_T(\beta)\|. \tag{1}$$

We note that $\|E_T(\alpha)\| = O(|T|) = O(\|N_T(\beta)\|/|\beta \cap E|)$, that is, for complexity purposes, only the last term of (1) matters. However, for the sake of clarity, we will still refer to the terms of equation (1) arising from the marked triangulation in our analyses. All complexities will be reported under the assumption that $|\beta \cap T| = \Omega(|T|^2)$, as otherwise there are some faster trivial algorithms.

Our choices of data structures for α and β allow us to represent curves of exponentially many intersection points with respect to the complexity of the diagram. Referring to the discussion at the end of Section 2.1, more intersections may lead to more complicated fundamental groups and, consequently [44][Proposition 2.6.6], more complicated 3-manifolds. In this sense, our structure is significantly more compressed than representations of Heegaard diagrams with *both* α and β edged, which can have, at most, $E_T(\alpha) \times E_T(\beta) = O(|T|^2)$ intersections.

Complexity gains are also noticeable if we compare our structure with triangulations of 3-manifolds. The usual algorithm to triangulate Heegaard diagrams [29] takes polynomial-time in $|\alpha \cap \beta|$. Moreover, as we empirically verify in Section 5, converting a Heegaard word to a diagram can be exponentially faster than converting the word to a triangulation for some choices of input and at least as fast for the general case. On the downside, it is unclear how to obtain and manipulate geometric information (such as *hyperbolicity*) or to compute certain quantum invariants directly from Heegaard diagrams, making triangulations still preferable for these kinds of problems.Refer to Section 6 of the paper's full version.

As stated in the introduction, our goal is to establish basic efficient algorithms even in this very compressed setting. For such a purpose, we will make extensive use of the next two results. The first of them is (almost) fully established by Erickson and Nayyeri [23].

▶ **Theorem 3.** *Suppose that $(T, E_T(\alpha), N_T(\beta))$ is a genus g Heegaard diagram of complexity m. In a unit-cost RAM model of bit-size $\log|\beta \cap E| = O(m/|T|)$, one may compute*

(a) *the street complex $S(T, \beta)$ in time $O(m|T|)$;*

(b) *SLPs of total complexity $O(m|T|)$ of the crossing sequence of* all *streets of $S(T, \gamma)$ in time $O(m|T|)$;*

(c) *SLPs $I_T(\beta)$ of complexity $O(m)$, in time $O(m)$.*

For the next theorem, we might assume that a Heegaard word ϕ is given in a *power-notation form*, that is, $\phi = \tau_{s_n}^{k_n} \circ \cdots \circ \tau_{s_1}^{k_1}$, $s_i \neq s_{i+1}$ for $1 \leq i < n$, and with each $k_i \in \mathbb{Z}$ in signed binary. A *fully uncompressed* Heegaard word $\phi' = \tau_{s'_{|\phi'|}}^{\pm 1} \circ \ldots \tau_{s'_1}^{\pm 1}$, with some s'_i potentially equal to s'_{i+1}, can be put in power-notation form in time $O(|\phi'|)$ with a unit-cost RAM model of bit-size $\lg|\phi'|$. For convenience, we shall also assume that the Lickorish generators are given as edged curves with respect to T. Nevertheless, even if we suppose that $\mathcal{L}$ is given by normal coordinates or intersection sequences, the theorem still holds with similar complexity bounds by multiple applications of Theorem 4.9 of the full version.

▶ **Theorem 4.** *Suppose $\phi = \tau_{s_n}^{k_n} \circ \cdots \circ \tau_{s_1}^{k_1}$, $s_i \neq s_{i+1}$ for $1 \leq i < n$, is a Heegaard word in $\mathrm{Mod}(\Sigma_g)$ and let T be a triangulation of Σ_g with edged α and Lickorish generators. There is an algorithm to compute intersection sequences $I_T(\phi(\alpha))$ in time $\Theta(g|T| + n|T|\lg k)$ in a unit-cost RAM model of bit-size $\lg k$ where $\lg k = \sum_{i=1}^{n} \lg k_i$. Moreover, the Heegaard diagram $(T, E_T(\alpha), N_T(\phi(\alpha)))$ of complexity $O(|T|^3 \lg^2 k)$ can be computed in time $O((|T| \lg k)^{10})$.*

Proof. For each $s \in \mathcal{L}$, denote its edge list by $E_T(s) = \{e_{s,1}, \ldots, e_{s,\|E_T(s)\|}\}$, where $e_{s,j} \in E$ for $1 \leq j \leq \|E_T(s)\|$, and define $I_T(s, j)$ as (an SLP of) the intersection sequence of a normal curve isotopic to s starting at a face adjacent to $e_{s,j}$. We can compute $I_T(s, j)$ for each $1 \leq j \leq \|E_T(s)\|$ in time $O(g|T|)$ (Lemma 3.4 in the full version). We also compute $I_T(a)$ for each $a \in \alpha$; in total, this initial step takes time $O(g|T|)$.

Let $I_T(\beta_0) = I_T(\alpha)$. Assuming $k_i > 0$, we recursively construct $I_T(\beta_i)$, for $1 \leq i \leq n$, by replacing any occurrence of an edge $e_{s_i,j} \in E_T(s_i)$ in the SLPs $I_T(\beta_{i-1}) = \{[x_1 = \mathrm{EXPR}_1, \ldots, x_p = \mathrm{EXPR}_p]\}_{b \in \beta}$ with an SLP $[y_1, \ldots, y_q]$ of $I_T(s_i)^{k_i} \cdot s_{i,j}$, that is,

$$I_T(\beta_i) = \{[y_1, \ldots, y_q, x'_{q+1} = \mathrm{EXPR}'_1, \ldots, x'_{q+p} = \mathrm{EXPR}'_q]\}_{b \in \beta}$$

where EXPR'_r is the word EXPR_r with each occurrence of x_t substituted by x'_{p+t} and each occurrence of $s_{i,j}$ substituted by y_q. The case of $k_i < 0$ is treated similarly. By Lemma 1, we note that this can be done in time $O(\|I_T(\beta_{i-1})\| + \lg k_i \|I_T(s_i)\|)$ in the assumed model of computation. In particular, $I_T(\beta_i)$ is an intersection sequence representation of $\beta_i = \tau_{s_i}^{k_i} \circ \cdots \circ \tau_{s_1}^{k_1}(\alpha)$ and has complexity

$$\begin{aligned} O(\|I_T(\beta_i)\|) &= O(\|I_T(\beta_{i-1})\| + \lg k_i \|I_T(s_i)\|) \\ &= O\left(\|I_T(\alpha)\| + \sum_{i'=1}^{i} \lg k_{i'} \|I_T(s_{i'})\|\right) \\ &\leq O\left(|T| + |T| \sum_{i'=1}^{i} \lg k_{i'}\right) = O\left(|T| \sum_{i'=1}^{i} \lg k_{i'}\right), \end{aligned}$$

where in the second line we recursively applied the equation of the first. The total time to compute $I_T(\beta) = I_T(\beta_n)$ is bounded by $n \times \|I_T(\beta)\| = O(n|T| \lg k)$, and, in the case $k_i = 1$ for all $1 \leq i \leq n$, this becomes $\Theta(n^2|T|) = \Theta(n|T| \lg k)$.

Although $I_T(\beta)$ is, by construction, standard in T, it might not be normal. We can use the main algorithm of [45] to deterministically compute, in time $O(\|I_T(\beta)\|^{10})$, cyclic reduced SLPs for β with complexity $O(\|I_T(\beta)\|^2)$. Alternatively, this can be done with a randomized algorithm in time $O(\|I_T(\beta)\|^3)$ with probability of error at most δ and in a unit-cost RAM model of bit-size $O(\|I_T(\beta)\| + \lceil 1/\delta \rceil)$ [53]. Part (c) of Lemma 1 finishes the proof. ◀

4 Algorithms on Heegaard diagrams

We now describe some polynomial-time algorithms and operations on a genus g Heegaard diagram $(T, E_T(\alpha), N_T(\beta))$ with complexity m and representing a closed 3-manifold M. We choose here to highlight algorithms that refer back to the discussion on censuses from the introduction, giving examples on how to simplify the underlying combinatorics of splittings (Theorem 5), disqualify inputs (Theorem 6), and compute invariants (Theorems 7 and 8). For the complexities reported, we assume a unit-cost RAM model of bit-size $\log |\beta \cap E| = O(\|N_T(\beta)\|/|T|)$.

▶ **Theorem 5** (`DO A DISK SLIDE`). *A diagram $(T, E_T(\alpha), N_T(\beta'))$, where β' is the result of applying a disk slide on any two components of β, can be computed in time $O(m|T|)$.*

Proof. Compute the street complex $S(T, \beta)$; recall that it has complexity of $O(|T|)$. Using breadth-first in the dual graph of $S(T, \beta)$, find, in time $O(|T|^2)$, the shortest arc p that connects faces adjacent to b_1 and b_2 and that does not cross any component of β, which can be avoided by adding infinite weights to the edges of $S(T, \beta)$ not in T. Note that such an arc must exist because no component of β can be separating in the surface. Moreover, because p is shortest and therefore intersects at most $O(|T|)$ streets in $S(T, \beta)$, we can compute using Part (b) of Theorem 3, an SLP of complexity $O(m|T| + |T|) = O(m|T|)$ of an intersection sequence of p with respect to T, call it $I_T(p)$.

Suppose that p is not trivial, that is, it is not just a single point. By Theorem 6.2 of [24], find, in time $O(m|T|)$, SLPs of the intersection sequences of b_1 and b_2 with respect to T starting at edges adjacent to the endpoint faces of p, call them $I_T(b_1)$ and $I_T(b_2)$. Finally, let $I_T(b) = I_T(b_2)^{-1} \circ I_T(p)^{-1} \circ I_T(b_1) \circ I_T(p)$. Note that $I_T(b)$ already is an intersection sequence of a normal curve in T. For, since p is a shortest path, it is normal in $S(T, \beta)$, which implies that it is also normal in T. Moreover, p has junctions as endpoints; if it had streets as endpoints, there would be a shorter arc connecting b_1 to b_2 starting at one of

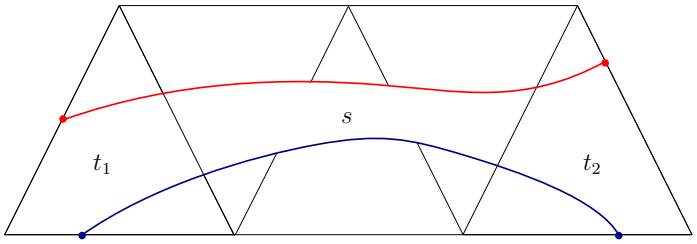

Figure 3 Diagram for the proof of Theorem 5. The red and blue curves represent b_1 and b_2, respectively. The black lines represent the other edges of $S(T, \beta)$. The edges e_1 and e_2 are adjacent to s and t_1 and t_2, respectively.

the ends of the street. Therefore, any other reduction in $I_T(b)$ would necessarily arise by p entering a junction of $S(T, \beta)$ through a non-redundant port e that is also intersected by either b_1 or b_2. But then, the other triangle in T adjacent to e is adjacent to either b_1 or b_2, implying that p is not shortest. By repeated applications of Part (c) of Lemma 1, one can compute, in time $O(|T| \times m)$, the number of (unsigned) occurrences of each edge of E in the SLP of $I_T(b)$, resulting in $N_T(b)$.

If p is trivial, then b_1 and b_2 bound the same face in $S(T, \beta)$. Nonetheless, because the curves are not isotopic, there exists at least one face bounded by b_1 and not bounded by b_2. We call such a face *good*. Choose, in time $O(|T|)$, a non-good face t_1 that is adjacent to a good face through an edge e_1, see Figure 3. Note that e_1 is the endpoint of a street s of sides in b_1 and b_2. Look for the other end of this street, e_2 – again, because b_1 and b_2 are not isotopic, the street has another end. Let t_2 be the face (in fact, a junction) not equal to s, but also adjacent to e_2. Compute, again through [24][Theorem 6.2], the sequence of faces adjacent to b_1 starting at t_1 and going all the way up to t_2 without crossing s, call this sequence $I_T(\widetilde{b_1})$. Compute the related sequence for b_2, this time starting in t_2, call it $I_T(\widetilde{b_2})$. Let $I_T(b) = I_T(\widetilde{b_2}) \circ I_T(\widetilde{b_1})$, note that it is also normal in T as b_1 and b_2 are normal and it enters t_1 and t_2 through distinct edges. Part (c) of Lemma 1 finishes this case. ◀

▶ **Theorem 6** (`DETECT REDUCTION`). *There is an algorithm that runs in time $O(\text{poly}(m, |T|))$ and detects whether a diagram is reducible. Moreover, if the diagram is reducible, it returns either an isotopic pair of $a \in \alpha$ and $b \in \beta$, or a separating curve s.*

Proof. We divide the algorithm into two parts, corresponding to the two hypotheses of reducibility of diagrams, namely the existence of an isotopic pair of meridians $a \in \alpha$ and $b \in \beta$, or of an essential separating curve that splits Σ_g into subdiagrams. For the first case, apply [41][Theorem 6.3] to find, in polynomial-time on the input, an isotopic diagram (α, β') in efficient position. This procedure increases the complexity of β to $m' = m(1 + 2|T|)$ We note that although this subroutine is polynomial-time on the input, the author of [41] does not provide an estimation of the polynomial degree, and any attempt on our part to give one would be out of the scope of this paper. Because we made the diagram efficient, there can only be a component $a \in \alpha$ isotopic to some $b' \in \beta'$ if $e(b') = 0$ for all $e \in E_T(\alpha)$. If that is the case, we trace $S(T, b')$. Since $b' \cap a = \emptyset$, the curve a is still fully in the edges of the street complex. Nonetheless, a is isotopic to b' if and only if they cobound a cylinder [25], a case that can be determined in time $O(|S(T, b')|)$. Checking this for every pair of curves a and b' that do not intersect – there can be at most g^2 of these pairs – gives a test for the first hypothesis.

If the above test returns no clear reduction pair, we look for a separating curve, s, disjoint from α and β, which exists if and only if s is non-contractable in $\Sigma'' = \Sigma_g \backslash (\alpha \cup \beta)$. Because $\Sigma' = \Sigma_g \backslash \beta$ is a $2g$ puncture sphere, Σ'' is a union of (potentially one) punctured spheres, and

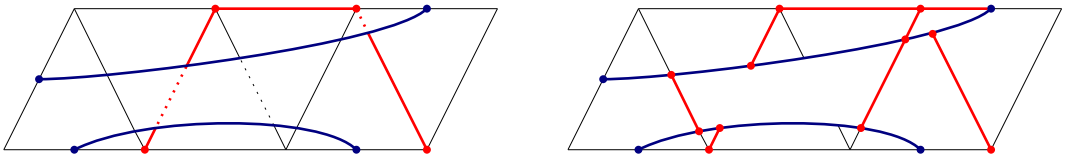

Figure 4 Diagram illustrating the proof of Theorem 6. Left: part of the overlay graph defined by the β curves (blue) with respect to a triangulation (black) and the corresponding embeddings of a segment of the α curves (red). Redundant ports are shown in dashed. A cellular embedding of the surface Σ'' can be obtained by puncturing the diagram along both blue and red curves. Right: another cellular embedding for Σ'' in which we enlarge the holes punctured when cutting α by moving redundant ports in $E_T(\alpha)$ to both ends of the streets.

it embeds a non-contractable curve if and only if one of its components is not a disk. This means that we only need to compute a cellular embedding of Σ'' and check its topological type. We start computing a triangulation for Σ' as in the proof of Theorem 5 by tracing $S(T,\beta)$ and disconnecting from the gluing rules the edges of $E_{S(T,\beta)}(\beta)$. We then want to further cut Σ' along α. But, because α is edged with respect to T, each segment of $E_T(\alpha)$ will be a port in the overlay graph of β (with respect to T). Each such a port will either be non-redundant – in which case it will still be an edge of $S(T,\beta)$) – or it will be redundant – in which case it will be in the crossing sequence of some non-degenerate street and will be deleted to form $S(T,\beta)$ (left side of Figure 4). Therefore, for each street in $E_{S(T,\beta)}(\beta)$ (degenerate or not), we can use Part (c) of Lemma 1 and Part (b) of Theorem 3 to determine, in time $O(\|E_T(\alpha)\| \times m|T|)$, if there is an occurrence of an edge in $E_T(\alpha)$ in a street's crossing sequence and delete from the complex $S(T,\beta)$ (the interior of) any such a street. Because the streets are disks in Σ', bounded by one or two segments of β, we note that, topologically, deleting the full interior of the streets that are transverse to some edge in $E_T(\alpha)$ is equivalent to puncturing a hole corresponding to a redundant port of the overlay graph by simply "enlarging" the hole (right side of Figure 4). This means that we have a cellular embedding of Σ'' which can be queried, in polynomial-time, for its topological type. Assuming that Σ'' has a component that is not a disk: we can use the techniques of [23] (refer to Section 2.3 of [14] for details) to find, in time $O(|T'|)$ a non-contractable curve s in Σ'' expressed as an SLP of complexity $O(m|T| + |T|) = O(m|T|)$. We finally compute $S(T,s)$, in time $O(m|T|^2)$, to check whether a diagram is defined on each component of $\Sigma_g \backslash s$. For such, we need only to find 1. the number of α and β components on each side of s; and, 2. whether s is not isotopic to an α or β curve by again looking for bounded cylinders. ◀

▶ **Theorem 7** (GET $H_1(M)$)**.** *The Smith normal form of the presentation matrix of $H_1(M)$ can be found in time $\widetilde{O}(m^2g^3 + m|T|)$, where $\widetilde{O}$ indicates some hidden polylogarithmic factors.*

Proof. We first compute the presentation matrix K of the homology group whose coefficients are algebraic intersection numbers between the diagram's curves. Because we assume that each a_j is a list of edges $E_T(a_j)$, given an SLP $I_T(b_i)$,

$$\hat{i}(b_i, a_j) = \sum_{e \in E_T(a_j)} e(I_T(b_i)) - e^{-1}(I_T(b_i)),$$

which, by Lemma 1, can be computed in time $O(\|I_T(b_i)\| \times \|E_T(a_j)\|)$. Therefore, referring to Theorem 3 to compute $I_T(\beta)$, the total computational time of this procedure is $O\Big(m|T| + \sum_{i=1}^{g}\sum_{j=1}^{g} \|I_T(b_i)\|\|E_T(a_j)\|\Big) = O(m|T| + m|T|)$. Finally, we reduce the matrix K to its Smith normal form in time $\widetilde{O}(g^3m^2)$ by the Kannan-Bachem algorithm [52]. ◀

▶ **Theorem 8** (`GET` $\pi_1(M)$)**.** *A presentation of* $\pi_1(M)$ *can be computed in time* $O(m|T|)$*.*

Proof. Given $I_T(\beta) = \{I_T(b)\}_{b\in\beta}$ delete, in linear time on the complexity of the SLPs, all references to edges $E - E_T(\alpha)$ and substitute each edge of $E_T(\alpha)$ by the symbol that represents its associated component $a \in \alpha$, here seen as a generator $\pi_1(M)$. This gives g distinct SLPs $r_i = [x_1^i = \text{EXPR}_1^i, \dots, x_{\ell_i}^i = \text{EXPR}_{\ell_i}^i]$ whose complexities add up to m and describe the relations of the presentation of $\pi_1(M)$. A presentation of $\pi_1(M)$ can then be defined with generators $a_1, \dots, a_g, x_1^1, \dots, x_{\ell_i-1}^1, \dots, x_{\ell_g-1}^g$ and relators $(x_j^i)^{-1} \cdot \text{EXPR}_j^i$ (this corresponds to repeated applications of a Tietze move [36] to the presentation of $\pi_1(M)$ computed using the method described at the end of Section 2.1). The output is a balanced presentation of $\pi_1(M)$ with $\|I_T(\beta)\|$-many generators and relators. The total time is dictated by Theorem 3. ◀

▶ Remark 9. Because the outputted presentation of $\pi_1(M)$ in Theorem 8 has linear size on the input, most complexity results about *uncompressed presentations* of 3-manifolds (such as triangulations) that appeal to the fundamental groups still hold in our *compressed* data structure. This is the case, for example, of Zentner's proof that sphere recognition lies in co-`NP` [55] (assuming the generalized Riemann hypothesis). Similarly, the usual bounds on the volume [16] and diameter [54] of hyperbolic 3-manifolds can be naturally translated to our notion of complexity.

5 Examples and experiments

Given a Heegaard word presentation (Σ_g, ϕ) of a closed 3-manifold M, there exist algorithms to compute, in linear time on the length of ϕ, a triangulation of M with as many tetrahedra [1, 2, 5, 38]. This implies, for example, that the first homology group of M, $H_1(M)$, can be computed in polynomial-time on $|\phi|$. These algorithms should be contrasted with Theorem 4, which says that SLPs of the intersection sequence $I_T(\beta = \phi(\alpha))$ can be computed in linear time on the complexity of ϕ *measured in power-notation form*. Because the fundamental group of M depends only on $I_T(\beta)$, algebraic information on splittings with significantly complicated gluing maps can be retrieved by Theorems 7 and 8.

Here, we explore the gains from the compressed data structure by comparing an early implementation of our methods with `SnapPy`'s [18] `Twister` [4] module, commonly used to compute triangulations from Heegaard words. Since experiments are limited to Heegaard words as input, there is no need to use proper normal coordinates or even triangulations of Σ_g; instead, we may compute the SLPs of the induced diagram with respect to general cellular embeddings, therefore avoiding the high cost of cyclic reductions. This guarantees that, for the problems analyzed, our theoretical complexity is at least equal to `Twister`'s. Even without these simplifying assumptions, we are still exponentially faster in the worst cases, as we show in the following section about lens spaces.

5.1 Genus 1

Heegaard splittings over the genus 1 surface – i.e., the torus – form the well-known family of *lens spaces*. Every lens space is uniquely given by a pair of integers $L(p, q)$, which, for a Heegaard splitting (Σ_g, a, b), represents the algebraic intersection numbers of b with the usual longitude and meridian of the torus (there is some ambiguity in the values of p, q, see [47]). A Dehn twist about the meridian m of the torus maps (p, q) to $(p + q, q)$, while about the longitude ℓ maps to $(p, q - p)$. This gives a representation of $\text{Mod}(\Sigma_1)$ in $\mathbb{Z}^2$ from which the homology of the Heegaard splitting (Σ_g, ϕ) can be exactly computed by $|\phi|$ matrix

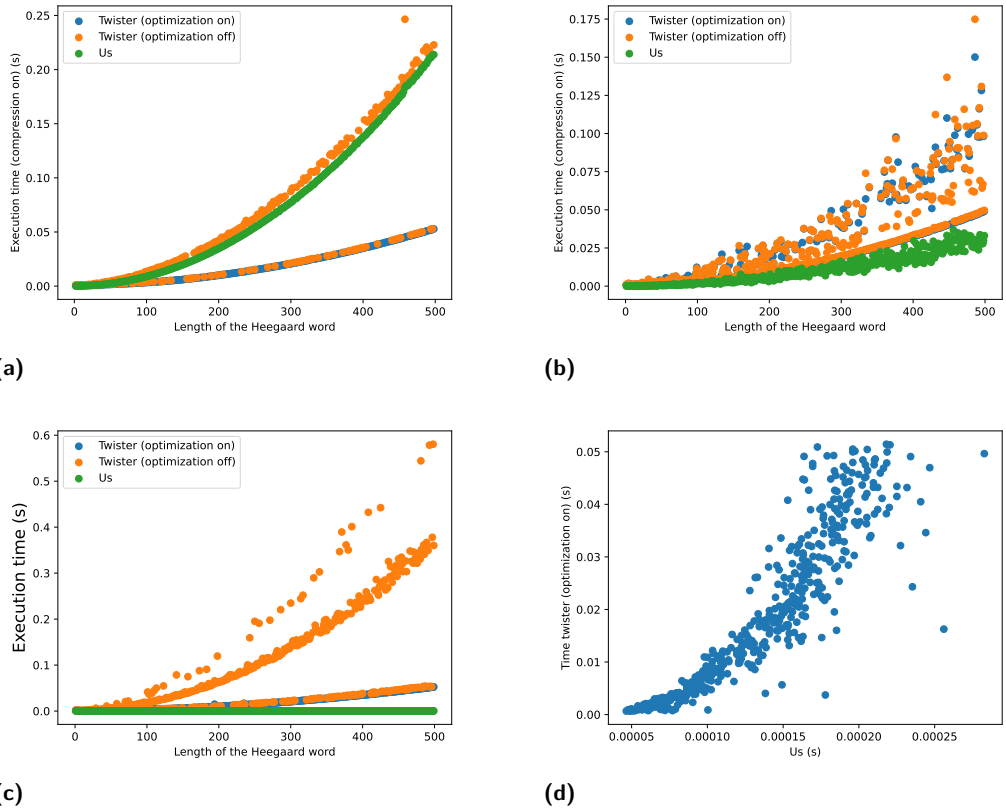

Figure 5 Experiments with computing the homology group H_1 of Heegaard splittings over the torus. Averages were computed for 30 repetitions.

multiplications and one query on the final value of q. We used this approach to benchmark the accuracy of our method, which returned the expected H_1 for input words ϕ of length up to 1000, the maximum size we tested.

Figure 5 compares the average computational time needed to find $H_1(\Sigma_1, \phi)$ taken by our method and by `Twister` for four different families of ϕ indexed by an integer n: (a) words of form $(\tau_m \tau_\ell^{-1})^n$; (b) uniformly sampled words of length n; (c) words of form τ_ℓ^n; and (d) words of form τ_ℓ^n, but this time, given in power-notation form. Averages were taken over 30 repetitions. In all cases, we compared our times with `Twister`'s with and without optimization (where, by optimization, the authors mean the greedy folding of tetrahedra; refer to their documentation for details[1]). Although optimization drastically reduces computational time, it seems to increase numerical error. In every instance, except for (d), we first transformed the words into a power-notation form before using them to compute $I_T(b)$; in total, this can be done in linear time on $|\phi|$.

▶ Remark 10. A direct application of the representation of $\mathrm{Mod}(\Sigma_1)$ implies that, for the family of case (a), $H_1^n = \mathbb{Z}/f_{2n}$ where f_n is the n-th Fibonacci number (starting at 0). We used this relation to benchmark the accuracy of the methods for different values of n. While our technique always agrees with the expected value for up to $n = 500$, `Twister` (with optimization both on and off) started deviating for values of $n \geq 25$. Interestingly, we did not observe divergences for higher genii splittings.

[1] Available at `https://snappy.computop.org`.

Case (a) is where our advantages compared to `Twister` are the worst. This is because the words $(\tau_m \tau_\ell^{-1})^n$ do not admit a non-trivial power-notation form, which is mainly responsible for our gains in the other cases. Nevertheless, it is worth highlighting that, while `Twister` is partially implemented in `C++`, our code is fully `python`. Case (b) represents the average case of computing the homology of lens spaces given a gluing map ϕ. We notice that we are now consistently faster for all values of n tested. This happens exactly as we explore redundancies by first putting the word in power-notation form, with significant gains every time the same Dehn twist occurs in sequence (since $\mathrm{Mod}(\Sigma_g)$ has two generators, this will be the case quite often). The gains due to repetitions are further highlighted in (c): while `Twister` is bound to take $O(n^c)$, for some $c > 1$, to compute the homologies, our approach takes time $O(n + \lg n)$, where the linear dependence comes solely from compressing the inputs to power-notation. In fact, if we assume the words already in power-notation, we are exponentially faster than `Twister`, as shown in case (d).

We can actually explore our structure's usage of power-notation to compute invariants of families of 3-manifolds that would necessarily require exponentially many tetrahedra to be represented. The minimum number of tetrahedra in a *layered triangulation* of a lens space $L(p, q)$ is $\sum_i (a_i) - 1$, where $(a_0, \ldots, a_k)$ is the continued fraction expansion of q/p [34]. Moreover, [32, 35] discovered families of lens spaces – including the spaces defined by $p = 2n, q = 1$ and $p = 4n, q = 2n - 1, n \geq 2$ – for which layered triangulations are minimal among *all* triangulations. So, for example, while the family $L(2^n, 1)$ necessarily requires exponentially many tetrahedra to be represented, it can be encoded in linear space in our data structure. It seems natural to expect that there are many other families of manifolds, with diagrams exponentially smaller than the minimum number of tetrahedra, for which our techniques can provide real computational gains compared to any existing method.

References

1. Gorjan Alagic and Edgar A. Bering. Quantum algorithms for invariants of triangulated manifolds. *Quant. Inf. Comput.*, 12(9-10):0843–0863, 2012. doi:10.26421/QIC12.9-10-8.
2. Gorjan Alagic and Catharine Lo. Quantum invariants of 3-manifolds and NP vs #P. *Quantum Info. Comput.*, 17(1–2):125–146, 2017.
3. M. Aschenbrenner, S. Friedl, and H. Wilton. *3-manifold Groups*. EMS series of lectures in mathematics. European Mathematical Society, 2015. URL: https://books.google.ch/books?id=syEEkui2AkoC.
4. Mark Bell, Tracy Hall, and Saul Schleimer. Twister (computer software). https://bitbucket.org/Mark_Bell/twister/, 2008–2014. Version 2.4.1.
5. Peter Brinkmann and Saul Schleimer. Computing triangulations of mapping tori of surface homeomorphisms. *Experimental Mathematics*, 10(4):571–581, 2001. doi:10.1080/10586458.2001.10504677.
6. Rhuaidi Antonio Burke, Benjamin A. Burton, and Jonathan Spreer. Small Triangulations of 4-Manifolds and the 4-Manifold Census. In Oswin Aichholzer and Haitao Wang, editors, *41st International Symposium on Computational Geometry (SoCG 2025)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:16, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2025.28.
7. Benjamin A. Burton. Efficient enumeration of 3-manifold triangulations. *Australian Mathematical Society Gazette*, 31(2):108–114, 2004.
8. Benjamin A. Burton. A new approach to crushing 3-manifold triangulations. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, pages 415–424, 2013. doi:10.1145/2462356.2462409.
9. Benjamin A. Burton. The cusped hyperbolic census is complete. *arXiv preprint*, 2014. arXiv:1405.2695.

10 Benjamin A. Burton. Tabulation of knots & 3-manifolds. Research Data Australia, University of Queensland, 2020. Dataset, https://researchdata.edu.au/tabulation-knots-3-manifolds/3370083.

11 Benjamin A. Burton. The Next 350 Million Knots. In Sergio Cabello and Danny Z. Chen, editors, *36th International Symposium on Computational Geometry (SoCG 2020)*, volume 164 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2020.25.

12 Benjamin A. Burton, Ryan Budney, William Pettersson, et al. Regina: Software for low-dimensional topology. http://regina-normal.github.io/, 1999–2023.

13 Benjamin A. Burton and William Pettersson. An edge-based framework for enumerating 3-manifold triangulations. In *31st International Symposium on Computational Geometry (SoCG 2015)*, pages 270–284. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.SOCG.2015.270.

14 Erin W Chambers, Éric Colin De Verdière, Jeff Erickson, Francis Lazarus, and Kim Whittlesey. Splitting (complicated) surfaces is hard. In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 421–429, 2006. doi:10.1145/1137856.1137918.

15 John H Conway. An enumeration of knots and links, and some of their algebraic properties. In *Computational problems in abstract algebra*, pages 329–358. Elsevier, 1970.

16 Daryl Cooper. The volume of a closed hyperbolic 3-manifold is bounded by times the length of any presentation of its fundamental group. *Proceedings of the American Mathematical Society*, 127(3):941, 1999.

17 Francesco Costantino, Yang-Hui He, Elli Heyes, and Edward Hirst. Learning 3-manifold triangulations. *Journal of Physics A: Mathematical and Theoretical*, 2024.

18 Marc Culler, Nathan M. Dunfield, Matthias Goerner, and Jeffrey R. Weeks. SnapPy, a computer program for studying the geometry and topology of 3-manifolds. Available at http://snappy.computop.org (19/05/2025).

19 Basudeb Datta. Minimal triangulations of manifolds. *arXiv preprint math/0701735*, 2007.

20 Olivier Devillers, Jean-Daniel Boissonnat, Mariette Yvinec, and Monique Teillaud. Triangulations in CGAL. In *Proceedings of the 16th Annual Symposium on Computational Geometry*, pages 11–18. ACM, 2000.

21 Clifford H Dowker and Morwen B Thistlethwaite. Classification of knot projections. *Topology and its Applications*, 16(1):19–31, 1983.

22 Henrique Ennes and Clément Maria. Hardness of computation of quantum invariants on 3-manifolds with restricted topology, 2025. doi:10.48550/arXiv.2503.02814.

23 Jeff Erickson and Sariel Har-Peled. Optimally cutting a surface into a disk. In *Proceedings of the eighteenth annual symposium on Computational geometry*, pages 244–253, 2002. doi:10.1145/513400.513430.

24 Jeff Erickson and Amir Nayyeri. Tracing compressed curves in triangulated surfaces. *Discrete Computational Geometry*, 49(4):823–863, June 2013. doi:10.1007/s00454-013-9515-z.

25 Benson Farb and Dan Margalit. *A primer on mapping class groups*, volume 41. Princeton University Press, 2011.

26 Torben Hagerup. Sorting and searching on the word RAM. In *STACS 98: 15th Annual Symposium on Theoretical Aspects of Computer Science Paris, France, February 25–27, 1998 Proceedings 15*, pages 366–398. Springer, 1998. doi:10.1007/BFB0028575.

27 Wolfgang Haken. Über das homöomorphieproblem der 3-mannigfaltigkeiten. i. *Mathematische Zeitschrift*, 80(1):89–120, 1962.

28 Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

29 Alexander He, James Morgan, and Em K Thompson. An algorithm to construct one-vertex triangulations of Heegaard splittings. *arXiv preprint*, 2023. arXiv:2312.17556.

30 John Hempel. *3-Manifolds*, volume 349. American Mathematical Society, 2022.

31 William Jaco and Ulrich Oertel. An algorithm to decide if a 3-manifold is a Haken manifold. *Topology*, 23(2):195–209, 1984.

32 William Jaco, Hyam Rubinstein, and Stephan Tillmann. Minimal triangulations for an infinite family of lens spaces. *Journal of Topology*, 2(1):157–180, 2009.

33 William Jaco and J Hyam Rubinstein. 0-efficient triangulations of 3-manifolds. *Journal of Differential Geometry*, 65(1):61–168, 2003.

34 William Jaco and J Hyam Rubinstein. Layered-triangulations of 3-manifolds. *arXiv preprint math/0603601*, 2006.

35 William Jaco, J Hyam Rubinstein, and Stephan Tillmann. Coverings and minimal triangulations of 3–manifolds. *Algebraic & Geometric Topology*, 11(3):1257–1265, 2011.

36 David Lawrence Johnson. *Presentations of groups*. Cambridge university press, 1997.

37 Jesse Johnson. Notes on Heegaard splittings. *preprint*, 2006.

38 Robert Koenig, Greg Kuperberg, and Ben W Reichardt. Quantum computation with Turaev–Viro codes. *Annals of Physics*, 325(12):2707–2749, 2010.

39 Greg Kuperberg. Identifying lens spaces in polynomial time. *Algebraic & Geometric Topology*, 18(2):767–778, 2018.

40 Marc Lackenby. Algorithms in 3-manifold theory, 2020. `arXiv:2002.02179`.

41 Marc Lackenby. Some fast algorithms for curves in surfaces. *arXiv preprint*, 2024. `doi:10.48550/arXiv.2401.16056`.

42 Marc Lackenby and Jessica S Purcell. The triangulation complexity of elliptic and Sol 3-manifolds. *Mathematische Annalen*, 390(2):1623–1667, 2024.

43 William BR Lickorish. A finite set of generators for the homeotopy group of a 2-manifold. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 60, pages 769–778. Cambridge University Press, 1964.

44 Sergei Vladimirovich Matveev and SV Matveev. *Algorithmic topology and classification of 3-manifolds*, volume 9. Springer, 2007.

45 Masamichi Miyazaki, Ayumi Shinohara, and Masayuki Takeda. An improved pattern matching algorithm for strings in terms of straight-line programs. In *Combinatorial Pattern Matching: 8th Annual Symposium, CPM 97 Aarhus, Denmark, June 30–July 2, 1997 Proceedings 8*, pages 1–11. Springer, 1997. `doi:10.1007/3-540-63220-4_45`.

46 Edwin E Moise. Affine structures in 3-manifolds: V. the triangulation theorem and hauptvermutung. *Annals of mathematics*, 56(1):96–114, 1952.

47 Nikolai Saveliev. *Lectures on the topology of 3-manifolds: an introduction to the Casson invariant*. Walter de Gruyter, 2011.

48 Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Algorithms for normal curves and surfaces. In *Computing and Combinatorics: 8th Annual International Conference, COCOON 2002 Singapore, August 15–17, 2002 Proceedings 8*, pages 370–380. Springer, 2002. `doi:10.1007/3-540-45655-4_40`.

49 Marcus Schaefer, Eric Sedgwick, and Daniel Stefankovic. Computing Dehn twists and geometric intersection numbers in polynomial time. In *CCCG*, volume 20, pages 111–114, 2008.

50 Saul Schleimer. Sphere recognition lies in np. *Low-dimensional and symplectic topology*, 82:183–213, 2011.

51 Jennifer Schultens. *Introduction to 3-manifolds*, volume 151. American Mathematical Soc., 2014.

52 Francis Sergeraert. About the Kannan-Bachem algorithm, 2024. `doi:10.48550/arXiv.2411.02422`.

53 Daniel Štefankovič. *Algorithms for simple curves on surfaces, string graphs, and crossing numbers*. Phd thesis, University of Chicago, 2005. Available at `http://people.cs.uchicago.edu/~laci/students/stefankovic-phd.pdf`.

54 Matthew E White. A diameter bound for closed, hyperbolic 3-manifolds. *arXiv preprint*, 2001. `arXiv:math/0104192`.

55 Raphael Zentner. Integer homology 3-spheres admit irreducible representations in sl(2,c). *Duke Mathematical Journal*, 167(9), June 2018. `doi:10.1215/00127094-2018-0004`.