An Edge-Based Framework for Enumerating 3-Manifold Triangulations*

Benjamin A. Burton and William Pettersson

School of Mathematics and Physics, The University of Queensland
Brisbane QLD 4072, Australia
bab@maths.uq.edu.au, william@ewpettersson.se

Abstract

A typical census of 3-manifolds contains all manifolds (under various constraints) that can be triangulated with at most n tetrahedra. Although censuses are useful resources for mathematicians, constructing them is difficult: the best algorithms to date have not gone beyond $n = 12$. The underlying algorithms essentially (i) enumerate all relevant 4-regular multigraphs on n nodes, and then (ii) for each multigraph G they enumerate possible 3-manifold triangulations with G as their dual 1-skeleton, of which there could be exponentially many. In practice, a small number of multigraphs often dominate the running times of census algorithms: for example, in a typical census on 10 tetrahedra, almost half of the running time is spent on just 0.3% of the graphs.

Here we present a new algorithm for stage (ii), which is the computational bottleneck in this process. The key idea is to build triangulations by recursively constructing neighbourhoods of edges, in contrast to traditional algorithms which recursively glue together pairs of tetrahedron faces. We implement this algorithm, and find experimentally that whilst the overall performance is mixed, the new algorithm runs significantly faster on those "pathological" multigraphs for which existing methods are extremely slow. In this way the old and new algorithms complement one another, and together can yield significant performance improvements over either method alone.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2.2 Graph Theory, G.4 Mathematical Software

Keywords and phrases triangulations, enumeration, graph theory

Digital Object Identifier 10.4230/LIPIcs.SOCG.2015.270

1 Introduction

In many fields of mathematics, one can often learn much by studying an exhaustive "census" of certain objects, such as knot dictionaries. Our focus here is on censuses of closed 3-manifolds – essentially topological spaces that locally look like $\mathbb{R}^3$. Combinatorially, any closed 3-manifold can be represented by a *triangulation*, formed from tetrahedra with faces identified together in pairs [14]. A typical census of 3-manifolds enumerates all 3-manifolds under certain conditions that can be constructed from a fixed number of tetrahedra.

One of the earliest such results was a census of all cusped hyperbolic 3-manifolds which could be built from at most five tetrahedra, by Hildebrand and Weeks [7]; this was later extended to all such manifolds on at most nine tetrahedra [4, 6, 16]. For closed orientable 3-manifolds, Matveev gave the first census of closed orientable prime manifolds on up to six tetrahedra [11]; this has since been extended to 12 tetrahedra [9, 12].

* Partially supported by the Australian Research Council (projects DP1094516, DP110101104). A detailed version of this paper appears at http://arxiv.org/abs/1412.2169

31st International Symposium on Computational Geometry (SoCG'15).
Editors: Lars Arge and János Pach; pp. 270–284

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Most (if not all) census algorithms in the literature enumerate 3-manifolds on n tetrahedra in two main stages. The first stage is to generate a list of all 4-regular multigraphs on n nodes. The second stage takes each such graph G, and sequentially identifies faces of tetrahedra together to form a triangulation with G as its dual 1-skeleton (for a highly tuned implementation of such an algorithm, see [5]).

There are $|S_3| = 6$ possible maps to use for each such identification of faces. Thus for each graph G, the algorithm searches through an exponential (in the number of tetrahedra) search tree where each leaf in this tree represents a triangulation but not necessarily a 3-manifold triangulation. Much research has focused on pruning this search tree by identifying and removing subtrees which only contain non-3-manifold triangulations [2, 3, 10, 11].

In this paper we describe a different approach to generating a census of 3-manifolds. The first stage remains the same, but in the second stage we build up the neighbourhood of each *edge* in the triangulation recursively, instead of joining together faces one at a time. This is, in a sense, a paradigm shift in census enumeration, and as a result it generates significantly different search trees with very different opportunities for pruning. We implement the new algorithm in a specific setting (potential minimal triangulations), and compare its performance against existing algorithms. We find that this new search framework complements existing algorithms very well, and we predict that a heuristic combination that combines the benefits of this with existing algorithms can significantly speed up census enumeration.

The key idea behind this new search framework is to extend each possible dual 1-skeleton graph to a "fattened face pairing graph", and then to find particular cycle-based decompositions of these new graphs. We also show how various improvements to typical census algorithms (such as those in [1]) can be translated into this new setting.

2 Definitions and notation

In combinatorial topology versus graph theory, the terms "edge" and "vertex" have distinct meanings. Therefore in this paper, the terms *edge* and *vertex* will be used to mean an edge or vertex of a tetrahedron, triangulation or manifold; and the terms *arc* and *node* will be used to mean an edge or vertex in a graph respectively.

A 3-manifold is a topological space that locally looks like either 3-dimensional Euclidean space (i.e., $\mathbb{R}^3$) or closed 3-dimensional Euclidean half-space (i.e., $\mathbb{R}^3_{z\geq 0}$). In this paper when we mention 3-manifolds we always mean compact and connected 3-manifolds. When we refer to faces, we are explicitly talking about 2-faces (i.e., facets of a tetrahedron). We represent 3-manifolds combinatorially as triangulations [14]: a collection of tetrahedra (3-simplices) with some 2-faces pairwise identified.

▶ **Definition 1.** A *general triangulation* is a collection $\Delta_1, \Delta_2, \ldots, \Delta_n$ of n abstract tetrahedra, along with bijections $\pi_1, \pi_2, \ldots, \pi_m$ where each π_i is an affine map between two faces of tetrahedra, and where each face of each tetrahedron is in at most one such bijection.

We call these affine bijections *face identifications* or simply *identifications*. Note that unlike simplicial complexes, we do allow identifications between two distinct faces of the same tetrahedron. If the quotient space of such a triangulation is a 3-manifold, we will say that the triangulation represents said 3-manifold.

▶ **Notation 2.** *Given a tetrahedron with vertices a, b, c and d, we will define face a to be the face opposite vertex a. That is, face a is the face consisting of vertices b, c and d. We will sometimes also refer to this as face bcd. We will write abc ↔ efg to mean that face abc is*

identified with face efg and that in this identification we have vertex a identified with vertex e, vertex b identified with vertex f and vertex c identified with vertex g.

We will also use the notation ab to denote the edge joining vertices a and b on some tetrahedron. Note that by this notation, the edge ab on a tetrahedron with vertices labelled a, b, c and d will be the intersection of faces c and d.

As a result of the identification of various faces, some edges or vertices of various tetrahedra are identified together. The *degree* of an edge of the triangulation, denoted $\deg(e)$, is defined to be the number of edges of tetrahedra which are identified together to form the edge of the triangulation.

We also need to define the *link* of a vertex before we can discuss triangulations of 3-manifolds.

▶ **Definition 3.** Given a vertex v in some triangulation, the link of v, denoted $Link(v)$, is the (2-dimensional) frontier of a small regular neighbourhood of v.

We now detail the properties a general triangulation must have to represent a 3-manifold. Recall that we only discuss connected 3-manifolds.

▶ **Lemma 4.** *A general triangulation represents a 3-manifold if the following additional conditions hold:*

- *the triangulation is connected;*
- *the link of any vertex in the triangulation is homeomorphic to either a 2-sphere or a disc;*
- *no edge in the triangulation is identified with itself in reverse.*

We will call such a triangulation a *3-manifold triangulation*. It is straight-forward to show that these conditions are both necessary and sufficient for the underlying topological space to be a 3-manifold (possibly with boundary). However in this paper we only consider 3-manifolds without boundary. That is, every face of a tetrahedron will be identified with some other face in a 3-manifold triangulation.

▶ **Lemma 5.** *Given any connected closed triangulation T on n tetrahedra with k vertices where no edge is identified with itself in reverse, the triangulation has $n + k$ edges if and only if the link of each vertex in T is homeomorphic to a 2-sphere.*

The above result is routine to show. We also need to define the *face pairing graph* of a triangulation. The *face pairing graph* of a triangulation, also known as the dual 1-skeleton, is a graphical representation of the face identifications of the triangulation. Each tetrahedron is associated with a node in the face pairing graph, and one arc joins a pair of tetrahedra for each identification of faces between the two tetrahedra. Note that face pairing graph is not necessarily a simple graph. Indeed, it will often contain both loops (when there is an identification of two distinct faces of the same tetrahedron) and parallel arcs (when there are multiple face identifications between two tetrahedra).

Lastly, we need a few properties of manifolds and triangulations.

▶ **Definition 6.** A 3-manifold $\mathcal{M}$ is *irreducible* if every embedded 2-sphere in $\mathcal{M}$ bounds a 3-ball in $\mathcal{M}$.

▶ **Definition 7.** A 3-manifold $\mathcal{M}$ is *prime* if it cannot be written as a connected sum of two manifolds where neither is a 3-sphere.

▶ **Definition 8.** A 3-manifold is $\mathbb{P}^2$*-irreducible* if it is irreducible and also contains no embedded two-sided projective plane.

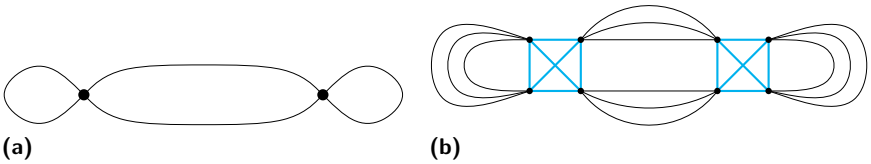

Figure 1 The face pairing graph (a) and fattened face pairing graph (b) of a triangulation. Note that the blue arcs are internal arcs, while the black arcs are external arcs.

Prime manifolds are the most fundamental manifolds to work with. We note that prime 3-manifolds are either irreducible, or are one of the orientable direct product $S^2 \times S^1$ or the non-orientable twisted product $S^2 \tilde{\times} S^1$. As these are both well known and have triangulations on two tetrahedra, for any census of minimal triangulations on three or more tetrahedra we can interchange the conditions "prime" and "irreducible". Any non-prime manifold can be constructed from a connected sum of prime manifolds, so enumerating prime manifolds is sufficient for most purposes. A similar (but more complicated) notion holds for $\mathbb{P}^2$-irreducible manifolds in the non-orientable setting. As such, minimal prime $\mathbb{P}^2$-irreducible triangulations form the basic building blocks in combinatorial topology.

▶ **Definition 9.** A 3-manifold triangulation of a manifold $\mathcal{M}$ is *minimal* if $\mathcal{M}$ cannot be triangulated with fewer tetrahedra.

Minimal triangulations are well studied, both for their relevance to computation and for their applications in zero-efficient triangulations [8]. Martelli and Petronio [10] also showed that, with the exceptions S^3, RP^3 and $L_{3,1}$, the minimal number of tetrahedra required to triangulate a closed, irreducible and $\mathbb{P}^2$-irreducible 3-manifold $\mathcal{M}$ is equal to the *Matveev complexity* [12] of $\mathcal{M}$.

3 Manifold decompositions

In this section we define a fattened face pairing graph, and show how we can represent any general triangulation as a specific decomposition of its fattened face pairing graph. This allows us to enumerate general triangulations by enumerating graph decompositions. We then demonstrate how to restrict this process to only enumerate 3-manifold triangulations.

A *fattened face pairing graph* is an extension of a face pairing graph F which we use in a dual representation of the corresponding triangulation. Instead of one node for each tetrahedron, a fattened face pairing graph contains one node for each face of each tetrahedron. Additionally, a face identification in the triangulation is represented by *three* arcs in the fattened face pairing graph; these three arcs loosely correspond to the three pairs of edges which are identified as a consequence of the face identification.

▶ **Definition 10.** Given a face pairing graph F, a fattened face pairing graph is constructed by first tripling each arc (i.e., for each arc e in F, add two more arcs parallel to e), and then replacing each node ν of F with a copy of K_4 such that each node of the K_4 is incident with exactly one set of triple arcs that meet ν.

▶ **Example 11.** Figure 1 shows a face pairing graph and the resulting fattened face pairing graph. The arcs shown in blue are what we call *internal* arcs. Each original node has been replaced with a copy of K_4 and in place of each original arc a set of three parallel arcs have been added.

We will refer to the arcs of each K_4 as *internal arcs*, and the remaining arcs (coming from the triple edges) as *external arcs*. As a visual aid we will draw internal arcs in blue. Each such K_4 represents a tetrahedron in the associated triangulation, and as such we will say that a fattened face pairing graph has n tetrahedra if it contains $4n$ nodes.

Triangulations are often labelled or indexed in some manner. Given any labelling of the tetrahedra and their vertices, we label the corresponding fattened face pairing graph as follows. For each tetrahedron i with faces a, b, c and d, we label the nodes of the corresponding K_4 in the fattened face pairing graph $v_{i,a}$, $v_{i,b}$, $v_{i,c}$ and $v_{i,d}$, such that if face a of tetrahedron i is identified with face b of tetrahedron j then there are three parallel external arcs between $v_{i,a}$ and $v_{j,b}$.

In such a labelling, the node $v_{i,a}$ represents face a of tetrahedron i. Each internal arc $\{v_{i,a}, v_{i,b}\}$ represents the unique edge common to faces a and b of tetrahedron i. Each external arc $\{v_{i,a}, v_{j,b}\}$ represents one of the three pairs of edges of tetrahedra which become identified as a result of identifying face a of tetrahedron i with face b of tetrahedron j. Note that the arc only represents the pair of edges being identified, and does not indicate the orientation of said identification.

We now define *ordered decompositions* of fattened face pairing graphs. Later, we show that there is a natural correspondence between such a decomposition and a general triangulation, and we show exactly how the 3-manifold constraints on general triangulations (see Lemma 4) can be translated to constraints on these decompositions. There is also a natural relationship between such decompositions and *spines* of 3-manifolds, as used by Matveev and others [12]; we touch on this relationship again later in this section.

▶ **Definition 12.** An *ordered decomposition* of a fattened face pairing graph $F = (E, V)$ is a set of closed walks $\{P_1, P_2, \ldots, P_n\}$ such that:

- $\{P_1, P_2, \ldots, P_n\}$ partition the arc set E;
- P_i is a closed walk of even length for each i; and
- if arc e_{j+1} immediately follows arc e_j in one of the walks then exactly one of e_j or e_{j+1} is an internal arc.

An ordered decomposition of a fattened face pairing graph exactly describes a general triangulation. We outline this idea here by showing how three parallel external arcs can represent an identification of faces. Complete technical details are given in in the full version of this paper.

Since the ordered decomposition consists of closed walks of alternating internal and external arcs, the decomposition pairs up the six arcs exiting each node so that each external arc is paired with exactly one internal arc. To help visualise this, we can draw such nodes as larger ellipses, with three external arcs and three internal arcs entering the ellipse, as in Figure 2. Each external arc meets exactly one internal arc inside this ellipse. This only represents how such arcs are paired up in a given decomposition – the node is still incident with all six arcs. We also see in Figure 2 that the fattened face pairing graph can always be drawn such that any "crossings" of arcs only occur between external arcs. Such crossings are simply artefacts of how the fattened face pairing graph is drawn in the plane, and in no way represent any sort of underlying topological twist.

Figure 3 shows a partial drawing of an ordered decomposition of a fattened face pairing graph. In this, we see a set of three parallel external arcs between nodes $v_{1,d}$ and $v_{2,h}$. This tells us that face d of tetrahedron 1 is identified with face h of tetrahedron 2. Additionally, we see that one of the external arcs connects internal arc $\{v_{1,c}, v_{1,d}\}$ with internal arc $\{v_{2,g}, v_{2,h}\}$. This tells us that edge ab of tetrahedron 1 (represented by $\{v_{1,c}, v_{1,d}\}$) is identified with edge

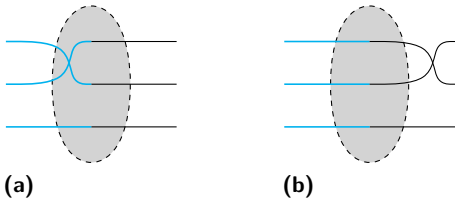

Figure 2 Two close up views of a node of a fattened face pairing graph with the same pairing of arcs. The node itself is represented by the grey ellipse, and all six arcs are incident upon this node. Note how both figures show the same pairing of edges, the only difference is where the "crossing" occurs.

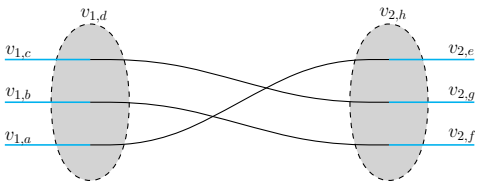

Figure 3 A partial drawing of a fattened face pairing graph.

ef of tetrahedron 2 (represented by $\{v_{2,g}, v_{2,h}\}$). Since we know that face *abc* is identified with face *efg* modulo a possible reflection and/or rotation, this tells us that vertex c is identified with vertex g in this face identification. We can repeat this process for the other paired arcs to see that vertex a is identified with vertex e and vertex b is identified with vertex f. The resulting identification is therefore $abc \leftrightarrow efg$.

Repeating this for each set of three parallel external arcs gives the required triangulation. The process is easily reversed to obtain an ordered decomposition from a general triangulation. Complete constructions for both processes are given in the full version of this paper.

Recall that $\deg(e)$ is the number of edges of tetrahedra identified together to form edge e in the triangulation. The following corollary follows immediately from the constructions.

▶ **Corollary 13.** *Given an ordered decomposition $\{P_1, \ldots, P_t\}$, each walk P_i corresponds to exactly one edge e in the corresponding general triangulation. In addition, $|P_i| = 2\deg(e)$.*

Recall that in a 3-manifold triangulation, no edge may be identified with itself in reverse. In a triangulation one may check this by considering a ring of tetrahedra around some edge. By tracking face identifications through the tetrahedra in the ring, one can determine if the central edge is identified with itself in reverse. The following definition combined with Lemma 15 achieves the same result in our new framework.

▶ **Definition 14.** Given an ordered decomposition $\mathcal{P} = \{P_1, P_2, \ldots, P_t\}$, we can *mark* a walk P_x as follows.

Pick an external arc e_s from P_x. Arbitrarily pick an external arc e_S parallel to e_s, and mark e_S as being "above" e_s. Then let $e_a = e_s$ and $e_A = e_S$ and continue as follows (see Figure 4 for a diagram of the construction):

- Let e_b be the next external arc in P_x after e_a.
- The internal arc preceding e_b joins two nodes. Call these nodes i and j, such that e_b is incident on j.
- Some external arc e_A incident on i must be marked as "above" e_a. Find the closed walk which e_A belongs to. In this closed walk there must exist some internal arc which either

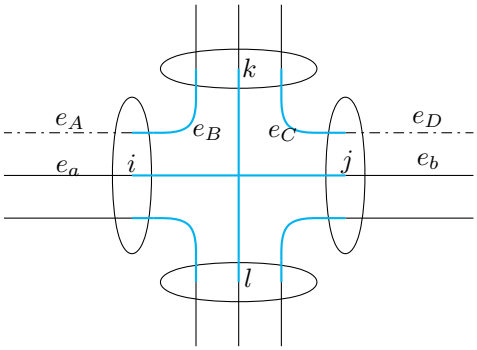

Figure 4 The process used to mark edges as per Definition 14. The dot-dashed arcs are the ones marked as "above". Recall that the ellipses are whole nodes, the insides of which denote how internal and external arcs are paired up in the decomposition.

immediately precedes or follows e_A through node i. Call this internal arc e_B. Note that the walk containing these two arcs need not be, and often is not, P_x. Arc e_B must be incident to i, and some other node which we shall call k.

- Find the internal arc e_C between nodes k and j, and find the walk P_y that it belongs to. In this walk, one of the arcs parallel to e_b must either immediately precede or follow e_C and be incident upon node j. Call this arc e_D.
- If $e_b = e_s$, and e_D is already marked as being above e_b, we terminate the marking process.
- Otherwise, mark the arc e_D as being above e_b and repeat the above steps, now using e_b in place of e_a, and using e_D in place of e_A.

Note that this process of marking specifically marks one arc as being "above" another. It does not mark arcs as being "above" in general.

To visualise this definition in terms of the decomposition, see Figure 4. The arcs e_a and e_b are part of a closed walk, and we are marking the edges "above" this walk. Arc e_A was arbitrarily chosen. Arc e_B follows e_A, and then we find e_C as the arc sharing one node with e_B and one with e_b. From e_C we can find and mark e_D.

In brief, the walks containing e_A and e_D represent edges of tetrahedra in the triangulation that share triangles with the common edge represented by P_x, and which both sit "above" this common edge (assuming some up/down orientation). Both e_B and e_C are internal arcs of the same tetrahedron and share a common node k, so we know that both these internal arcs represent edges of the same tetrahedron which share a common face k. The external arcs e_A and e_D represent identifications of e_B and e_C respectively with edges of (typically different) adjacent tetrahedra.

▶ **Lemma 15.** *Take an ordered decomposition containing a walk P_x with arcs marked according to Definition 14, and consider the corresponding triangulation. Then the edge of the triangulation represented by P_x is identified to itself in reverse if and only if there exists some external arc e in P_x that has two distinct external arcs both marked as "above" e.*

Essentially, this condition indicates that the marking procedure cycles through the entire walk twice (marking two parallel arcs as "above" each arc of the walk), as opposed to once (marking only one arc as "above" each arc of the walk). The proof of this lemma is routine, and is given in full in the complete version of this paper.

If a walk P_x in an ordered decomposition can be marked according to Definition 14 such that each external arc e in P_x has exactly one other external arc marked as "above" e, we say that this walk is *non-reversing*.

▶ **Definition 16.** A *manifold decomposition* is an ordered decomposition of a fattened face pairing graph satisfying all of the following conditions.

- The ordered decomposition contains $n + 1$ closed walks.
- The fattened face pairing graph contains $4n$ nodes.
- Each walk is non-reversing.
- The associated manifold triangulation contains exactly 1 vertex.

▶ **Theorem 17.** *Up to relabelling, there is a one-to-one correspondence between manifold decompositions of connected fattened face pairing graphs and 1-vertex 3-manifold triangulations.*

Proof. Earlier in this section we described the correspondence between general triangulations and ordered decompositions. All that remains is to show that the extra properties of a manifold decomposition force the corresponding triangulation to be a 3-manifold triangulation. Since the decomposition contains $n + 1$ walks, Corollary 13 tells us the triangulation has $n + 1$ edges. Additionally, each tetrahedron corresponds to four nodes in the fattened face pairing graph, so the triangulation has n tetrahedra and thus by Lemma 5 we see that the link of each vertex is homeomorphic to a 2-sphere. Each walk is non-reversing so Lemma 15 says that no edge in the corresponding triangulation is identified with itself in reverse, and we have the required result. ◀

We now define the notation used to express specific ordered decompositions. The notation is defined such that it can also be interpreted as a *spine code* (as used by Matveev's Manifold Recognizer [13]), and that the spine generated from such a spine code is a dual representation of the same combinatorial object represented by the manifold decomposition. For more detail on spine codes, see [12].

▶ **Notation 18.** *Take an ordered decomposition of a fattened face pairing graph with $4n$ nodes, and label each set of three parallel external arcs with a distinct value taken from the set $\{1, \ldots, 2n\}$ (so two external arcs receive the same label if and only if they are part of the same triple of parallel arcs). Assign an arbitrary orientation to each set of three parallel external arcs. For each walk in the ordered decomposition:*

1. *Create an empty ordered list.*
2. *Follow the external arcs in the walk.*
   a. *If an external arc is traversed in a direction consistent with its orientation, add $+i$ to the end of the corresponding ordered list.*
   b. *If instead the arc in the walk is traversed in the reverse direction, add $-i$ to the end of the list.*
   c. *Continue until the first external arc in the walk is reached.*

Note that this notation only records the external arcs, and does not record any internal arcs in walks.

We can also reconstruct the face pairing graph (and therefore the fattened face pairing graph) from this notation (in particular, we can reconstruct the internal arcs). The method essentially uses the fact that each external arc represents some identification of two faces (and three parallel external arcs will represent the same identification of two faces), and so we can use the orientation of each arc to distinguish between the two faces in each identification and thereby build up the face pairing graph.

An implementation note: it is trivial, given a fattened face pairing graph and a "partial" ordered decomposition in which all the internal arcs are missing, to reconstruct the complete ordered decomposition. For the theoretical discussions in this paper we work with the

full ordered decompositions, but in the implementation we only store the sequential list of external arcs as in Notation 18.

4 Algorithm and improvements

In this section we give various improvements that may be used when enumerating manifold decompositions (i.e., 3-manifold triangulations). These are based on known theoretical results in 3-manifold topology, combined with suitable data structures.

Enumeration algorithms [1, 2, 5, 9, 11, 12] in 3-manifold topology often focus on closed, minimal, irreducible and $\mathbb{P}^2$-irreducible 3-manifold triangulations. These properties were all defined in Section 2. For brevity, we say that a triangulation (or manifold decomposition) has such a property if and only if the underlying manifold has the property. In both this section and Section 5, we will restrict our algorithm to this same setting. This highlights the usefulness of our algorithm, and allows us to demonstrate how existing results can be translated into our new framework.

Many existing algorithm implementations in the literature [5, 12] build triangulations by identifying faces pairwise (or taking combinatorially equivalent steps, such as annotating edges of special spines [12]). The algorithm we give here essentially constructs the neighbourhood of each *edge* of the triangulation one at a time. Therefore the search tree traversed by our new algorithm is significantly different than that traversed by other algorithms. This is highlighted experimentally by the results given in Section 5.

4.1 Algorithm

The basis of our implementation is a simple backtracking approach to enumerate manifold decompositions. Walks are built up one arc at a time, and recursion ensures that every possible manifold decomposition is found. However, this approach is not tractable for any interesting values of n, and so we introduce the following improvements.

4.2 Limiting the size of walks

The following results are taken from [1], though in the orientable case similar results were known earlier by other authors [9, 12].

▶ **Lemma 19.** (2.1 in [1]) *No closed minimal triangulation has an edge of degree three that belongs to three distinct tetrahedra.*

▶ **Lemma 20.** (2.3 and 2.4 in [1]) *No closed minimal $\mathbb{P}^2$-irreducible triangulation with ≥ 3 tetrahedra contains an edge of degree ≤ 2.*

Given that the degree of an edge e of a triangulation is the number of tetrahedron edges which are identified to form e, these results translate to manifold decompositions as follows.

▶ **Corollary 21.** *No closed minimal $\mathbb{P}^2$-irreducible manifold decomposition with ≥ 3 tetrahedra contains a walk which itself contains less than three external arcs.*

▶ **Corollary 22.** *No closed minimal manifold decomposition contains a walk which itself contains exactly three internal arcs representing edges on distinct tetrahedra (i.e., belonging to three distinct K_4 subgraphs).*

The above results are direct corollaries, as it is simple to translate the terms involved and the results are simple enough to implement in an algorithm. In the backtracking algorithm,

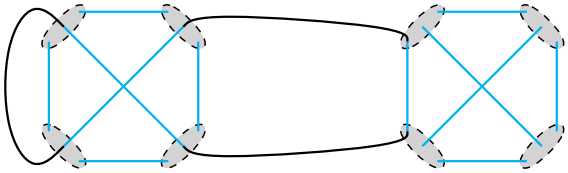

Figure 5 The only possible walk containing 3 internal arcs not all from distinct tetrahedra in a fattened face pairing graph on more than 1 tetrahedron. Only the external arcs used in the walk are shown, other external arcs are not shown.

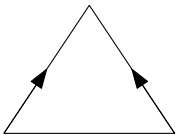

Figure 6 A one-face cone formed by identifying the two marked edges.

this means we can implement a check on the number of arcs in a walk before adding the walk to the decomposition. This is implementable as a constant time check if the length of the current partial walk is stored.

Additionally, for a census of 1-vertex triangulations on n tetrahedra, a manifold decomposition must contain exactly $n+1$ walks. If the algorithm has completed k walks, then there are $n+1-k$ walks left to complete. We use this idea in the following improvement.

By Corollary 22 a closed walk in a manifold decomposition which contains three internal arcs must contain two internal arcs belonging to the same K_4, as in Figure 5. We modify our algorithm to enumerate all such closed walks first. Each such walk is either present or absent in any manifold decomposition. For each possible combination of such walks, we fix said walks and then run the search on the remaining arcs. All other walks must now contain at least four external arcs, so during the census on n tetrahedra if the algorithm has completed k walks and there are less than $4(n+1-k)$ unused external arcs we know that the partial decomposition cannot be completed to a 1-vertex manifold decomposition.

▶ **Improvement 23.** *For each K_4 in the given graph, determine if two of its internal arcs can be used together in a walk containing exactly three internal arcs. If this is possible, add said walk to the set S. Then, for each subset $s \subseteq S$, use s as a starting set of walks and attempt to complete the ordered decomposition. If during the enumeration process k walks have been completed and there are less than $4(n+1-k)$ unused external arcs, prune the search tree at this point.*

4.3 Avoiding cone faces

For some properties of minimal triangulations, it is not clear that the corresponding tests can be implemented cheaply. Here, we identify further results from the literature that enable fast implementations in our setting. The following was shown in [1].

▶ **Lemma 24.** (2.8 in [1]) *Let T be a closed minimal $\mathbb{P}^2$-irreducible triangulation containing ≥ 3 tetrahedra. Then no single face of T has two of its edges identified to form a cone as illustrated in Figure 6.*

For manifold decompositions, our translation of this result also requires the underlying manifold to be orientable in order to give a fast algorithmic test.

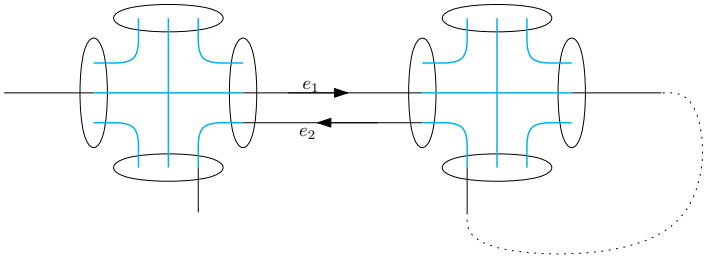

Figure 7 The depicted walk cannot occur in a closed minimal $\mathbb{P}^2$-irreducible orientable manifold decomposition as external arcs e_1 and e_2 are used in opposite directions. The dotted lines indicates the walk continues through undrawn parts of the fattened face pairing graph.

▶ **Lemma 25.** *Let D be a closed minimal $\mathbb{P}^2$-irreducible manifold decomposition of an orientable manifold containing ≥ 3 tetrahedra. Then no walk of D can use two parallel external arcs in opposite directions (as seen in Figure 7).*

A complete proof appears in the full version of this paper. The proof assigns orientations to corresponding tetrahedra, and then tracks orientations of the edges of tetrahedra to show that if a one-face cone is present then an edge must be identified with itself in reverse. This result leads to the following.

▶ **Improvement 26.** *When enumerating* orientable *manifold decompositions, if an external arc e is to be added to some walk W, and e is parallel to another external arc e' which itself is in W, check whether e and e' will be used in opposite directions. If so, do not use e at this point; instead backtrack and prune the search tree.*

4.4 One vertex tests

Definition 16 requires that the associated manifold only have one vertex. We test this by tracking properties of the vertex links as the manifold decomposition (i.e., triangulation) is built up. Specifically, while the manifold decomposition is still being constructed, no vertex link may be a closed surface.

▶ **Improvement 27.** *When building up a manifold decomposition, track how many "frontier edges" remain around each vertex link. If any vertex links are closed off before the manifold decomposition is completed, backtrack and prune the current subtree of the search space.*

The number of frontier edges of each vertex link, as well as which vertex links are identified together, are tracked via a union-find data structure. The data structure is slightly tweaked to allow back tracking (see [2] for details), storing the number of frontier edges at each node. For more details on the union-find algorithm in general, see [15].

4.5 Canonicity and Automorphisms

When running a search, many equivalent manifold decompositions will be found. These decompositions may differ in the order of the walks found, or two walks might have different starting arcs or directions. For example, the two walks (a, b, c) and $(-b, -a, -c)$ are equivalent. The second starts on a different arc, and traverses the walk backwards, but neither of these change the manifold decomposition. Additionally, the underlying face pairing graph often has non-trivial automorphism group.

To eliminate such duplication, we only search for *canonical* manifold decompositions. We use the obvious definition for a canonical walk (lowest-index arc is written first and is used in the positive direction).

There are two points in the algorithm where we might test for canonical decompositions.

▶ **Improvement 28.** *Every time an external arc is added to a walk, check if the current decomposition is canonical. If not, disregard this choice of arc and prune the search tree.*

▶ **Improvement 29.** *Every time a walk is completed, check if the current decomposition is canonical. If not, disregard this choice of arc and prune the search tree.*

Unfortunately, checking if a (possibly partial) decomposition is canonical is not computationally cheap. Experimental results showed that using Improvement 29 was significantly faster than using Improvement 28 as fewer checks for canonicity were made.

5 Results and Timing

In this section we detail the results from testing the algorithm. We test the manifold decomposition algorithm and its improvements from Section 4 against the existing implementation in *Regina*. Our algorithm (and indeed all known enumeration algorithms) are exponential in the number of nodes on a given graph. As a result, testing is limited to graphs of at most 10 nodes. Recall also that we are testing the enumeration of closed, minimal, irreducible and $\mathbb{P}^2$-irreducible 3-manifold triangulations.

Regina is a suite of topological software and includes state of the art algorithms for census enumerations in various settings, including non-orientable and hyperbolic manifolds [3, 4]. *Regina* and its source code are freely available, which facilitates comparable implementations and fair testing. *Regina* also filters out invalid triangulations as a final stage, which allows us to test the efficiency of our various improvements by enabling or disabling them independently. Like other census algorithms in the literature, *Regina* builds triangulations using the traditional framework by identifying faces two at a time.

In testing, we measure time to begin when either algorithm is given some textual representation of a face pairing graph, and ending when all triangulations are found. That is, testing times include the calculation of automorphisms (for both *Regina* and our new algorithms), as well as the construction of the fattened face pairing graph.

We find that while *Regina* outperforms our new algorithms overall, there are non-trivial subcases for which our algorithm runs an order of magnitude faster. Importantly, in a typical census on 10 tetrahedra, *Regina* spends almost half of its running time on precisely these subcases. This shows that our new algorithm has an important role to play: it complements the existing framework by providing a means to remove some of its most severe bottlenecks. Section 5.2 discusses these cases in more detail.

These observations are, however, in retrospect: what we do not have is a clear indicator in advance for which algorithm will perform best for any given subcase.

Recall that a full census enumeration involves generating all 4-regular multigraphs, and then for each such graph G, enumerating triangulations with face pairing graph G. In earlier sections we only dealt with individual graphs, but for the tests here we ran each algorithm on all 4-regular multigraphs of a given order n.

In the following results, we use the term MD to denote our basic algorithm, using improvements 23, 27 and 29. For enumerating orientable manifolds only, we also use Improvement 26 and denote the corresponding algorithm as MD-o. Experimentation indicated that Improvement 27 was computationally expensive, and so we also tested algorithm MD*

Table 1 Running time of *Regina* and the manifold decomposition (MD) algorithms when searching for manifold decompositions on n tetrahedra.

(a) Running times (in seconds) for the general setting.

n	*Regina*	MD
7	29	80
8	491	2453
9	11 288	79 685
10	323 530	3 406 211

(b) Running times (in seconds) for the orientable setting

n	*Regina*	MD-o
7	< 1	25
8	147	535
9	3 499	13 161
10	90 969	430 162

(using only Improvements 23 and 29) and algorithm MD*-o (using Improvements 23 26 and 29). Note that these last two algorithms may find ordered decompositions which are not necessarily manifold decompositions, but we can easily filter these out once the enumeration is complete.

The algorithms were tested on a cluster of Intel Xeon L5520s running at 2.27GHz. Times given are total CPU time; that is, a measure of how long the test would take single-threaded on one core. The algorithms themselves, when run on all 4-regular multigraphs on n nodes, are trivially parallelisable which allows each census to complete much faster by taking advantage of available hardware.

We note that, as expected, the census results are consistent across both algorithms.

5.1 Aggregate tests

In the general setting, where we allow orientable and non-orientable triangulations alike, Table 1a highlights that *Regina* outperforms MD when summed over all face pairing graphs. The difference seems to grow slightly as n increases, pointing to the possibility that more optimisations in this setting are possible.

We suspect that tracking the orientability of vertex links is giving *Regina* an advantage here (see [2], Section 5). Tracking orientability is more difficult with ordered decompositions, as the walks are built up one at a time – each external arc represents an identification of edges, but does not specify the orientation of this identification. Thus orientability cannot be tested until at least two of any three parallel external arcs are used in walks.

We also compare MD-o to *Regina*, where we ask both algorithms to only search for orientable triangulations. Both algorithms run significantly faster than in the general setting (demonstrating that Improvement 26 is a significant improvement). Table 1b shows that *Regina* outperforms MD-o roughly by a factor of four. This appears to be constant, and here we expect MD to be comparable to *Regina* after more careful optimisation (such as *Regina*'s own algorithm has enjoyed over the past 13 years [1, 2]).

To test Improvement 27 (the one-vertex test), we compare MD* and MD*-o against MD and MD-o respectively. The timing data in Tables 2b and 2a shows that MD* and MD*-o outperformed MD and MD-o, demonstrating that Improvement 27 actually slows down the algorithm. We verified that Improvement 27 is indeed discarding unwanted triangulations – the problem is that tracking the vertex links is too computationally expensive. Algorithms MD* and MD*-o instead enumerate these unwanted triangulations and test for one vertex after the fact, discarding multiple vertex triangulations after they have been explicitly constructed. The cost of this is included in the timing results, which confirms that such an "after the fact" verification process is indeed faster than the losses incurred by Improvement 27.

Table 2 Running times of MD, MD*, MD-o, MD*-o when searching for manifold decompositions on n tetrahedra.

(a) Running times (in seconds) for the general setting.

n	MD	MD*
7	80	71
8	2 453	1 875
9	79 685	58 743
10	3 406 211	1 624 025

(b) Running times (in seconds) for the orientable setting.

n	MD-o	MD*-o
7	25	16
8	535	446
9	13 161	10 753
10	430 162	291 544

Table 3 Running time in seconds of MD* and *Regina* on particular graphs on 10 nodes.

Graph ID	*Regina*	MD*
48 308	2476	142
48 083	2487	192
48 288	2164	118
47 332	2141	229
47 333	2003	134
47 520	2083	221
46 914	2108	302

5.2 Individual graph tests

It is on individual (and often pathological) face pairing graphs that the new algorithm shines. Recall that the census enumeration problem requires running an enumeration algorithm on all connected 4-regular multigraphs of a given order. Table 3 shows the running time of both *Regina* and MD* on a cherry-picked sample of such graphs on 10 tetrahedra.

From these we can see that on some particular graphs, MD* outperforms *Regina* by an order of magnitude. While these graphs were cherry-picked, they do display the shortfalls of *Regina*. There are 48432 4-regular multigraphs on 10 nodes, and it takes *Regina* 89.9 CPU-hours to complete this census. Of these 48432 graphs, 48242 are processed in under 300 seconds each. In contrast, it takes *Regina* 43.6 CPU-hours to process the remaining 190 graphs. This accounts for 48.5% of the running time of *Regina*'s census on 10 tetrahedra triangulations.

Running these "pathological" graphs through MD takes 12.1 CPU-hours, for a saving of 31.5 CPU-hours. This would reduce the running time of the complete census from 89 hours to 58 hours, a 35% improvement. With a priori knowledge of running times, passing each graph to the faster of either *Regina* or MD* can give a 44% improvement in the running time of the complete census.

6 Discussion

Although its performance is inconsistent, it is significant that our new framework performs an order of magnitude faster than existing algorithms on those subcases where existing algorithms struggle. If we had an effective heuristic that could determine which algorithm (*Regina* or MD*) to use for any given a 4-valent graph, then this could speed up the census

enumeration process significantly. Identifying such a heuristic is the subject of ongoing work.

Beyond the enumeration problem, this framework may also help us find families of "forbidden subgraphs": subgraphs which, if present in some 4-valent graph G, indicate that there is *no* minimal 3-manifold triangulation with G as its dual 1-skeleton [1]. Such graphs G can be omitted entirely from the census enumeration, and so identifying (with proof) forbidden subgraphs can lead to further significant improvements in census running times.

References

1 Benjamin A. Burton. Face pairing graphs and 3-manifold enumeration. *J. Knot Theory Ramifications*, 13(8):1057–1101, 2004.

2 Benjamin A. Burton. Enumeration of non-orientable 3-manifolds using face-pairing graphs and union-find. *Discrete & Computational Geometry*, 38(3):527–571, 2007.

3 Benjamin A. Burton. Detecting genus in vertex links for the fast enumeration of 3-manifold triangulations. In *ISSAC 2011: Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, pages 59–66. ACM, 2011.

4 Benjamin A. Burton. The cusped hyperbolic census is complete. *arXiv:1405.2695*, 2014.

5 Benjamin A. Burton, Ryan Budney, and William Pettersson. Regina: Software for 3-manifold topology and normal surface theory. Licensed under GPLv2, 1999-2013.

6 Patrick J. Callahan, Martin V. Hildebrand, and Jeffrey R. Weeks. A census of cusped hyperbolic 3-manifolds. *Math. Comp.*, 68(225):321–332, 1999. With microfiche supplement.

7 Martin Hildebrand and Jeffrey Weeks. A computer generated census of cusped hyperbolic 3-manifolds. In *Computers and mathematics*, pages 53–59. Springer, New York, 1989.

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

9 Bruno. Martelli and Carlo. Petronio. Three-manifolds having complexity at most 9. *Experimental Mathematics*, 10(2):207–236, 2001.

10 Bruno Martelli and Carlo Petronio. A new decomposition theorem for 3-manifolds. *Illinois J. Math.*, 46:755–780, 2002.

11 Sergei V. Matveev. Computer recognition of three-manifolds. *Experiment. Math.*, 7(2):153–161, 1998.

12 Sergei V. Matveev. *Algorithmic topology and classification of 3-manifolds*, volume 9 of *Algorithms and Computation in Mathematics*. Springer, Berlin, second edition, 2007.

13 Sergei V. Matveev et al. Manifold recognizer, 2014. `http://www.matlas.math.csu.ru/?page=recognizer`.

14 Edwin E. Moise. Affine structures in 3-manifolds. V. The triangulation theorem and Hauptvermutung. *Ann. of Math. (2)*, 56:96–114, 1952.

15 Robert Sedgewick. *Algorithms in C++*. Addison-Wesley, Reading, MA, 1992.

16 Morwen Thistlethwaite. Cusped hyperbolic manifolds with 8 tetrahedra. `http://www.math.utk.edu/~morwen/8tet/`, October 2010.