The Pachner graph and the simplification of 3-sphere triangulations

Benjamin A. Burton

November 18, 2010

Abstract

It is important to have fast and effective methods for simplifying 3-manifold triangulations without losing any topological information. In theory this is difficult: we might need to make a triangulation super-exponentially more complex before we can make it smaller than its original size. Here we present experimental work suggesting that for 3-sphere triangulations the reality is far different: we never need to add more than two tetrahedra, and we never need more than a handful of local modifications. If true in general, these extremely surprising results would have significant implications for decision algorithms and the study of triangulations in 3-manifold topology.

The algorithms behind these experiments are interesting in their own right. Key techniques include the isomorph-free generation of all 3-manifold triangulations of a given size, polynomial-time computable signatures that identify triangulations uniquely up to isomorphism, and parallel algorithms for studying finite level sets in the infinite Pachner graph.

1 Introduction

Triangulations of 3-manifolds are ubiquitous in computational knot theory and low-dimensional topology. They are easily obtained and offer a natural setting for many important algorithms.

Computational topologists typically allow triangulations in which the constituent tetrahedra may be "bent" or "twisted", and where distinct edges or vertices of the same tetrahedron may even be joined together. Such triangulations (sometimes called *semi-simplicial* or *pseudo-triangulations*) can describe rich topological structures using remarkably few tetrahedra. For example, the 3-dimensional sphere can be built from just one tetrahedron, and more complex spaces such as non-trivial surface bundles can be built from as few as six [15].

An important class of triangulations is the *one-vertex triangulations*, in which all vertices of all tetrahedra are joined together as a single point. These are simple to obtain [11, 14], and they are often easier to deal with both theoretically and computationally [6, 10, 14].

Keeping the number of tetrahedra small is crucial in computational topology, since many important algorithms are exponential (or even super-exponential) in the number of tetrahedra [5, 6]. To this end, topologists have developed a rich suite of local simplification moves that allow us to reduce the number of tetrahedra without losing any topological information [2, 16].

The most basic of these are the four *Pachner moves* (also known as *bistellar moves*). These include the 3-2 move (which reduces the number of tetrahedra but preserves the number of vertices), the 4-1 move (which reduces both numbers), and also their inverses, the 2-3 and 1-4 moves. It is known that any two triangulations of the same closed 3-manifold are related by a sequence of Pachner moves [21]. Moreover, if both are one-vertex triangulations then we can relate them using 2-3 and 3-2 moves alone [14].

However, little is known about how *difficult* it is to relate two triangulations by a sequence of Pachner moves. In a series of papers, Mijatović develops upper bounds on the number of moves required for various classes of 3-manifolds [17, 18, 19, 20]. All of these bounds are super-exponential in the number of tetrahedra, and some even involve exponential towers of exponential functions. For relating one-vertex triangulations using only 2-3 and 3-2 moves, no explicit bounds are known at all.

In this paper we focus on one-vertex triangulations of the 3-sphere. Here simplification is tightly linked to the important problem of *3-sphere recognition*, where we are given an input triangulation $\mathcal{T}$ and asked whether $\mathcal{T}$ represents the 3-sphere. This problem plays an key role in other important topological algorithms, such as connected sum decomposition [11, 12] and unknot recognition [9], and it is now becoming important in computational *4-manifold* topology. We can use Pachner moves for 3-sphere recognition in two ways:

- They give us a *direct* 3-sphere recognition algorithm: try all possible sequences of Pachner moves on $\mathcal{T}$ up to Mijatović's upper bound, and return "true" if and only if we reach one of the well-known "canonical" 3-sphere triangulations with one or two tetrahedra.
- They also allow a *hybrid* recognition algorithm: begin with a fast and/or greedy procedure to simplify $\mathcal{T}$ as far as possible within a limited number of moves. If we reach a canonical 3-sphere triangulation then return "true"; otherwise run a more traditional 3-sphere recognition algorithm on our new (and hopefully simpler) triangulation.

The direct algorithm lies well outside the realm of feasibility: Mijatović's bound is super-exponential in the number of tetrahedra, and the running time is at least exponential in Mijatović's bound. Current implementations [3] use the hybrid method, which is extremely effective in practice. Experience suggests that when $\mathcal{T}$ *is* the 3-sphere, the greedy simplification almost always gives a canonical triangulation. If simplification fails, we revert to the traditional algorithm of Rubinstein [22]; although this runs in exponential time, recent improvements by several authors have made it extremely effective for moderate-sized problems [6, 7, 11, 23].

Our aims in this paper are:

- to measure how easy or difficult it is *in practice* to relate two triangulations of the 3-sphere using Pachner moves, or to simplify a 3-sphere triangulation to use fewer tetrahedra;
- to understand why greedy simplification techniques work so well in practice, despite the prohibitive theoretical bounds of Mijatović;
- to investigate the possibility that Pachner moves could be used as the basis for a direct 3-sphere recognition algorithm that runs in sub-exponential or even polynomial time.

Fundamentally this is an experimental paper (though the theoretical underpinnings are interesting in their own right). Based on an exhaustive study of ~ 150 million triangulations (including ~ 31 million one-vertex triangulations of the 3-sphere), the answers appear to be:

- we can relate and simplify one-vertex triangulations of the 3-sphere using remarkably few Pachner moves;
- both procedures require us to add *at most two* extra tetrahedra, which explains why greedy simplification works so well;

- the number of moves required to simplify such a triangulation could also be bounded by a constant, which means polynomial-time 3-sphere recognition may indeed be possible.

These observations are extremely surprising, especially in light of Mijatović's bounds. If they can be proven in general—yielding a polynomial-time 3-sphere recognition algorithm—this would be a significant breakthrough in computational topology.

In Section 2 we outline preliminary concepts and introduce the *Pachner graph*, an infinite graph whose vertices represent triangulations and whose edges represent Pachner moves. This graph is the framework on which we build the rest of the paper. We define *simplification paths* through the graph, as well as the key quantities of *height* and *length* that we seek to measure.

We follow in Section 3 with two key tools for studying the Pachner graph: an isomorph-free census of all closed 3-manifold triangulations with ≤ 9 tetrahedra (which gives us the vertices of the graph), and *isomorphism signatures* of triangulations that can be computed in polynomial time (which allow us to construct the edges of the graph).

Section 4 describes parallel algorithms for bounding both the height and length of simplification paths, and presents the highly unexpected experimental results outlined above. We finish in Section 5 with a discussion of the implications and consequences of these results.

2 Triangulations and the Pachner graph

A *3-manifold triangulation of size* n is a collection of n tetrahedra whose $4n$ faces are affinely identified (or "glued together") in $2n$ pairs so that the resulting topological space is a closed 3-manifold.[1] We are not interested in the shapes or sizes of tetrahedra (since these do not affect the topology), but merely the combinatorics of how the faces are glued together. Throughout this paper, all triangulations and 3-manifolds are assumed to be connected.

We do allow two faces of the same tetrahedron to be identified, and we also note that distinct edges or vertices of the same tetrahedron might become identified as a by-product of the face gluings. A set of tetrahedron vertices that are identified together is collectively referred to as a *vertex of the triangulation*; we define an *edge* or *face of the triangulation* in a similar fashion.

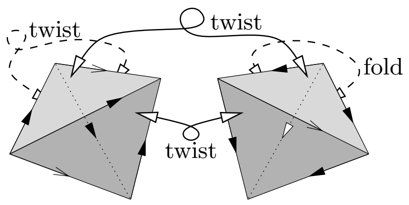


Figure 1: A 3-manifold triangulation of size $n = 2$

Figure 1 illustrates a 3-manifold triangulation of size $n = 2$. Here the back two faces of the first tetrahedron are identified with a twist, the front faces of the first tetrahedron are identified with the front faces of the second using more twists, and the back faces of the second tetrahedron are identified together by directly "folding" one onto the other. This is a *one-vertex*

[1] It is sometimes useful to consider *bounded* triangulations where some faces are left unidentified, or *ideal* triangulations where the overall space only becomes a 3-manifold when we delete the vertices of each tetrahedron. Such triangulations do not concern us here.

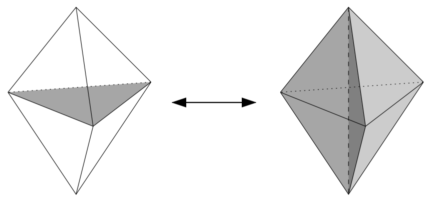
triangulation since all eight tetrahedron vertices become identified together. The triangulation has three distinct edges, indicated in the diagram by three distinct arrowheads.

Mijatović [17] describes a *canonical triangulation* of the 3-sphere of size $n = 2$, formed by a direct identification of the boundaries of two tetrahedra. In other words, given two tetrahedra $ABCD$ and $A'B'C'D'$, we directly identify face ABC with $A'B'C'$, ABD with $A'B'D'$, and so on. The resulting triangulation has four faces, six edges, and four vertices.

The four *Pachner moves* describe local modifications to a triangulation. These include:

- the *2-3 move*, where we replace two distinct tetrahedra joined along a common face with three distinct tetrahedra joined along a common edge;
- the *1-4 moves*, where we replace a single tetrahedron with four distinct tetrahedra meeting at a common internal vertex;
- the *3-2* and *4-1 moves*, which are inverse to the 2-3 and 1-4 moves.

These four moves are illustrated in Figure 2. Essentially, the 1-4 and 4-1 moves retriangulate the interior of a pyramid, and the 2-3 and 3-2 moves retriangulate the interior of a bipyramid. It is clear that Pachner moves do not change the topology of the triangulation (i.e., the underlying 3-manifold remains the same). Another important observation is that the 2-3 and 3-2 moves do not change the number of vertices in the triangulation.


(a) The 2-3 and 3-2 moves

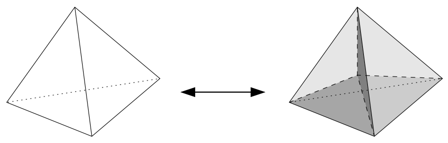

(b) The 1-4 and 4-1 moves

Figure 2: The four Pachner moves for a 3-manifold triangulation

Two triangulations are *isomorphic* if they are identical up to a relabelling of tetrahedra and a reordering of the four vertices of each tetrahedron (that is, isomorphic in the usual combinatorial sense). Up to isomorphism, there are finitely many distinct triangulations of any given size.

Pachner originally showed that any two triangulations of the same closed 3-manifold can be made isomorphic by performing a sequence of Pachner moves [21].[2] Matveev later strengthened this result to show that any two *one-vertex* triangulations of the same closed 3-manifold with at least two tetrahedra can be made isomorphic through a sequence of 2-3 and/or 3-2 moves [14]. The two-tetrahedron condition is required because it is impossible to perform a 2-3 or 3-2 move upon a one-tetrahedron triangulation (each move requires two or three distinct tetrahedra).

In this paper we introduce the *Pachner graph*, which describes *how* distinct triangulations of a closed 3-manifold can be related via Pachner moves.

Definition (Pachner graph)**.** Let M be any closed 3-manifold. The *Pachner graph* of M, denoted $\mathscr{P}(M)$, is an infinite graph constructed as follows. The vertices of $\mathscr{P}(M)$ correspond

[2]As Mijatović notes [17], Pachner's original result was proven only for true simplicial complexes, but it is easily extended to the more flexible definition of a triangulation that we use here.

to isomorphism classes of triangulations of M. Two vertices of $\mathscr{P}(M)$ are joined by an edge if and only if there is some Pachner move that converts one class of triangulations into the other.

The *restricted Pachner graph* of M, denoted $\mathscr{P}_1(M)$, is the subgraph of $\mathscr{P}(M)$ defined by only those vertices corresponding to one-vertex triangulations. The vertices of $\mathscr{P}(M)$ and $\mathscr{P}_1(M)$ are partitioned into finite *levels* $1, 2, 3, \ldots$, where each level n contains the vertices corresponding to n-tetrahedron triangulations.

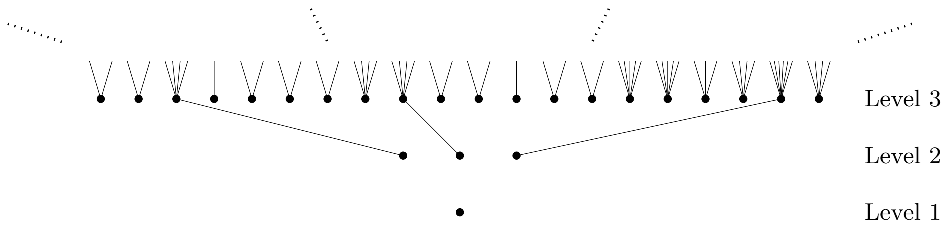


Figure 3: Levels 1–3 of the restricted Pachner graph of the 3-sphere

It is clear that the edges are well-defined (since Pachner moves are preserved under isomorphism), and that edges do not need to be directed (since each 2-3 or 1-4 move has a corresponding inverse 3-2 or 4-1 move). In the full Pachner graph $\mathscr{P}(M)$, each edge runs from some level i to a nearby level $i \pm 1$ or $i \pm 3$. In the restricted Pachner graph $\mathscr{P}_1(M)$, each edge must describe a 2-3 or 3-2 move, and must run from some level i to an adjacent level $i \pm 1$. Figure 3 shows the first few levels of the restricted Pachner graph of the 3-sphere.

We can now reformulate the results of Pachner and Matveev as follows:

Theorem 1 (Pachner, Matveev)**.** *The Pachner graph of any closed 3-manifold is connected. If we delete level 1, the restricted Pachner graph of any closed 3-manifold is also connected.*

To simplify a triangulation we essentially follow a path through $\mathscr{P}(M)$ or $\mathscr{P}_1(M)$ from a higher level to a lower level, which motivates the following definition.

Definition (Simplification path)**.** A *simplification path* is a directed path through either $\mathscr{P}(M)$ or $\mathscr{P}_1(M)$ from a vertex at some level i to a vertex at some lower level $< i$. The *length* of a simplification path is the number of edges it contains. The *height* of a simplification path is the smallest $h \geq 0$ for which the entire path stays in or below level $i + h$.

Both the length and height measure how difficult it is to simplify a triangulation: the length measures the number of Pachner moves, and the height measures the number of extra tetrahedra required. For the 3-sphere, the only known bounds on these quantities are the following:

Theorem 2 (Mijatović [17])**.** *Any triangulation of the 3-sphere can be converted into the canonical triangulation using less than* $6 \cdot 10^6 n^2 2^{2 \cdot 10^4 n^2}$ *Pachner moves.*

Corollary 3. *In the Pachner graph of the 3-sphere, from any vertex at level* $n > 2$ *there is a simplification path of length less than* $6 \cdot 10^6 n^2 2^{2 \cdot 10^4 n^2}$ *and height less than* $3 \cdot 10^6 n^2 2^{2 \cdot 10^4 n^2}$.

In the *restricted* Pachner graph, no explicit bounds on these quantities are known at all.

3 Key tools

Experimental studies of the Pachner graph are difficult: the graph itself is infinite, and even the finite level sets grow super-exponentially in size. By working with isomorphism classes of triangulations, we keep the level sets considerably smaller than if we had used labelled triangulations instead. However, the trade-off is that both the vertices and the edges of the graph are more difficult to construct.

In this section we outline two key algorithmic tools for studying the Pachner graph: a *census of triangulations* (which enumerates the vertices at each level), and polynomial-time computable *isomorphism signatures* (which allow us to construct the edges).

3.1 A census of triangulations

To enumerate the vertices of Pachner graphs, we build a census of all 3-manifold triangulations of size $n \leq 9$, with each triangulation included precisely once up to isomorphism. Because we are particularly interested in one-vertex triangulations as well as triangulations of the 3-sphere, we extract such triangulations into separate censuses with the help of the highly optimised 3-sphere recognition algorithm described in [7]. The final counts are summarised in Table 1.

Number of	All closed 3-manifolds		3-spheres only	
tetrahedra	No constraints	1-vertex only	No constraints	1-vertex only
1	4	3	2	1
2	17	12	6	3
3	81	63	32	20
4	577	433	198	128
5	5 184	3 961	1 903	1 297
6	57 753	43 584	19 935	13 660
7	722 765	538 409	247 644	169 077
8	9 787 509	7 148 483	3 185 275	2 142 197
9	139 103 032	99 450 500	43 461 431	28 691 150
Total	149 676 922	107 185 448	46 916 426	31 017 533

Table 1: Counts for 3-manifold triangulations of various types in the census

The algorithms behind this census are sophisticated; see [4] for some of the techniques involved. The constraint that the triangulation must represent a 3-manifold is critical: if we just enumerate all pairwise identifications of faces up to isomorphism, there are at least

$$\frac{[(4n-1) \times (4n-3) \times \cdots \times 3 \times 1] \cdot 6^{2n}}{n! \cdot 24^n} \quad \simeq \quad 2.35 \times 10^{16}$$

possibilities for $n = 9$. To enforce the 3-manifold constraint we use a modified union-find algorithm that tracks partially-constructed edge links and vertex links; see [4] for details.

Even with this constraint, we can prove that the census grows at a super-exponential rate:

Theorem 4. *The number of distinct isomorphism classes of 3-manifold triangulations of size n grows at an asymptotic rate of* $\exp(\Theta(n \log n))$.

The proof is detailed, and is given in the appendix.

3.2 Isomorphism signatures

To construct edges of the Pachner graph, we begin at a vertex—that is, a 3-manifold triangulation $\mathcal{T}$—and perform Pachner moves. Each Pachner move results in a new triangulation $\mathcal{T}'$, and our main difficulty is in deciding which vertex of the Pachner graph represents $\mathcal{T}'$.

A naïve approach might be to search through vertices at the appropriate level of the Pachner graph and test each corresponding triangulation for isomorphism with $\mathcal{T}'$. However, this approach is infeasible: although isomorphism testing is fast (as we prove below), the sheer number of vertices at level n of the graph is too large (see Theorem 4).

What we need is a property of the triangulation $\mathcal{T}'$ that is easy to compute, and that uniquely defines the isomorphism class of $\mathcal{T}'$. This property could be used as the key in a data structure with fast insertion and fast lookup (such as a hash table or a red-black tree), and by computing this property we could quickly jump to the relevant vertex of the Pachner graph.

Here we define such a property, which we call the *isomorphism signature* of a triangulation. In Theorem 6 we show that isomorphism signatures do indeed uniquely define isomorphism classes, and in Theorem 7 we show that they are small to store and fast to compute.

A *labelling* of a triangulation of size n involves: (i) numbering its tetrahedra from 1 to n inclusive, and (ii) numbering the four vertices of each tetrahedron from 1 to 4 inclusive. We also label the four faces of each tetrahedron from 1 to 4 inclusive so that face i is opposite vertex i. A key ingredient of isomorphism signatures is *canonical labellings*, which we define as follows.

Definition (Canonical labelling)**.** Given a labelling of a triangulation of size n, let $A_{t,f}$ denote the tetrahedron which is glued to face f of tetrahedron t (so that $A_{t,f} \in \{1, \ldots, n\}$ for all $t = 1, \ldots, n$ and $f = 1, \ldots, 4$). The labelling is *canonical* if, when we write out the sequence $A_{1,1}, A_{1,2}, A_{1,3}, A_{1,4}, A_{2,1}, \ldots, A_{n,4}$, the following properties hold:

(i) For each $2 \leq i < j$, tetrahedron i first appears before tetrahedron j first appears.

(ii) For each $i \geq 2$, suppose tetrahedron i first appears as the entry $A_{t,f}$. Then the corresponding gluing uses the *identity map*: face f of tetrahedron t is glued to face f of tetrahedron i so that vertex v of tetrahedron t maps to vertex v of tetrahedron i for each $v \neq f$.

As an example, consider the triangulation of size $n = 3$ described by Table 2. This table lists the precise gluings of tetrahedron faces. For instance, the second cell in the bottom row indicates that face 2 of tetrahedron 3 is glued to tetrahedron 2, in such a way that vertices $1, 3, 4$ of tetrahedron 3 map to vertices $4, 2, 3$ of tetrahedron 2 respectively. This same gluing can be seen from the other direction by examining the first cell in the middle row.

	Face 1	Face 2	Face 3	Face 4
	Vertices 234	Vertices 134	Vertices 124	Vertices 123
Tet. 1	Tet. 1: 231	Tet. 2: 134	Tet. 3: 124	Tet. 1: 423
Tet. 2	$\boxed{\text{Tet. 3: 341}}$	Tet. 1: 134	Tet. 2: 123	Tet. 2: 124
Tet. 3	Tet. 3: 123	$\boxed{\text{Tet. 2: 423}}$	Tet. 1: 124	Tet. 3: 234

Table 2: The tetrahedron face gluings for an example 3-tetrahedron triangulation

It is simple to see that the labelling for this triangulation is canonical. The sequence $A_{1,1}, \ldots, A_{n,4}$ is $1, 2, 3, 1, 3, 1, 2, 2, 3, 2, 1, 3$ (reading tetrahedron numbers from left to right and then top to bottom in the table), and tetrahedron 2 first appears before tetrahedron 3 as required. Looking closer, the first appearance of tetrahedron 2 is in the second cell of the top row where vertices $1, 3, 4$ map to $1, 3, 4$, and the first appearance of tetrahedron 3 is in the following cell where vertices $1, 2, 4$ map to $1, 2, 4$. In both cases the gluings use the identity map.

Lemma 5. *For any triangulation $\mathcal{T}$ of size n, there are precisely $24n$ canonical labellings of $\mathcal{T}$, and these can be enumerated in $O(n^2 \log n)$ time.*

Proof. In summary, we can choose any of the n tetrahedra to label as tetrahedron 1, and we can choose any of the $4! = 24$ labellings of its four vertices. From here the remaining labels are forced, and can be deduced in $O(n \log n)$ time. The full proof is given in the appendix. □

Definition (Isomorphism signature)**.** For any triangulation $\mathcal{T}$ of size n, enumerate all $24n$ canonical labellings of $\mathcal{T}$, and for each canonical labelling encode the full set of face gluings as a sequence of bits. We define the *isomorphism signature* to be the lexicographically smallest of these $24n$ bit sequences, and we we denote this by $\sigma(\mathcal{T})$.

To encode the full set of face gluings for a canonical labelling, we could simply convert a table of gluing data (such as Table 2) into a sequence of bits. For practical implementations we use a more compact representation, which will be described in the full version of this paper.

Theorem 6. *Given two 3-manifold triangulations $\mathcal{T}$ and $\mathcal{T}'$, we have $\sigma(\mathcal{T}) = \sigma(\mathcal{T}')$ if and only if $\mathcal{T}$ and $\mathcal{T}'$ are isomorphic.*

Proof. It is clear that $\sigma(\mathcal{T}) = \sigma(\mathcal{T}')$ implies that $\mathcal{T}$ and $\mathcal{T}'$ are isomorphic, since both signatures encode the same gluing data. Conversely, if $\mathcal{T}$ and $\mathcal{T}'$ are isomorphic then their $24n$ canonical labellings are the same (though they might be enumerated in a different order). In particular, the lexicographically smallest canonical labellings will be identical; that is, $\sigma(\mathcal{T}) = \sigma(\mathcal{T}')$. □

Theorem 7. *Given a 3-manifold triangulation $\mathcal{T}$ of size n, the isomorphism signature $\sigma(\mathcal{T})$ has $O(n \log n)$ size and can be generated in $O(n^2 \log n)$ time.*

Proof. To encode a full set of face gluings, at worst we require a table of gluing data such as Table 2, with $4n$ cells each containing four integers. Because some of these integers require $O(\log n)$ bits (the tetrahedron labels), it follows that the total size of $\sigma(\mathcal{T})$ is $O(n \log n)$.

The algorithm to generate $\sigma(\mathcal{T})$ is spelled out explicitly in its definition. The $24n$ canonical labellings of $\mathcal{T}$ can be enumerated in $O(n^2 \log n)$ time (Lemma 5). Because a full set of face gluings has size $O(n \log n)$, we can encode the $24n$ bit sequences and select the lexicographically smallest in $O(n^2 \log n)$ time, giving a time complexity of $O(n^2 \log n)$ overall. □

This space complexity of $O(n \log n)$ is the best we can hope for, since Theorem 4 shows that the number of distinct isomorphism signatures for size n triangulations grows like $\exp(\Theta(n \log n))$.

It follows from Theorems 6 and 7 that isomorphism signatures are ideal tools for constructing edges in the Pachner graph, as explained at the beginning of this section. Moreover, the relevant definitions and results are easily extended to bounded and ideal triangulations (which are beyond the scope of this paper). We finish with a simple but important consequence of our results:

Corollary 8. *Given two 3-manifold triangulations $\mathcal{T}$ and $\mathcal{T}'$ each of size n, we can test whether $\mathcal{T}$ and $\mathcal{T}'$ are isomorphic in $O(n^2 \log n)$ time.*

4 Analysing the Pachner graph

As discussed in the introduction, our focus is on one-vertex triangulations of the 3-sphere. We therefore direct our attention to $\mathscr{P}_1(S^3)$, the restricted Pachner graph of the 3-sphere.

In this section we develop algorithms to bound the shortest length and smallest height of any simplification path from a given vertex at level n of $\mathscr{P}_1(S^3)$. By running these algorithms over the full census of 31 017 533 one-vertex triangulations of the 3-sphere (as described in Section 3.1), we obtain a computer proof of the following results:

Theorem 9. *From any vertex at level n of the graph $\mathscr{P}_1(S^3)$ where $3 \leq n \leq 9$, there is a simplification path of length ≤ 13, and there is a simplification path of height ≤ 2.*

The bound $3 \leq n$ is required because there are no simplification paths in $\mathscr{P}_1(S^3)$ starting at level 2 or below (see Figure 3). For $n > 9$ a computer proof becomes computationally infeasible.

The results of Theorem 9 are astonishing, especially in light of Mijatović's super-exponential bounds. Furthermore, whilst it can be shown that the height bound of ≤ 2 is tight, the length estimate of ≤ 13 is extremely rough: the precise figures could be much smaller still. These results have important implications, which we discuss later in Section 5.

In this section we describe the algorithms behind Theorem 9, and we present the experimental results in more detail. Our algorithms are constrained by the following factors:

- Their time and space complexities must be close to linear in the number of vertices that they examine, due to the sheer size of the census.
- They cannot loop through all vertices in $\mathscr{P}_1(S^3)$, since the graph is infinite. They cannot even loop through all vertices at level $n \geq 10$, since there are too many to enumerate.
- They cannot follow arbitrary breadth-first or depth-first searches through $\mathscr{P}_1(S^3)$, since the graph is infinite and can branch heavily in the upward direction.[3]

Because of these limiting factors, we cannot run through the census and directly measure the shortest length or smallest height of any simplification path from each vertex. Instead we develop fast, localised algorithms that allow us to bound these quantities from above. To our delight, these bounds turn out to be extremely effective in practice. The details are as follows.

4.1 Bounding path heights

In this section we compute bounds H_n so that, from every vertex at level n of the graph $\mathscr{P}_1(S^3)$, there is some simplification path of height $\leq H_n$. As in Theorem 9, we compute these bounds for each n in the range $3 \leq n \leq 9$.

Algorithm 10 (Algorithm for computing H_n)**.** *This algorithm runs by progressively building a subgraph $G \subset \mathscr{P}_1(S^3)$. At all times we keep track of the number of distinct components of G (which we denote by c) and the maximum level of any vertex in G (which we denote by ℓ).*

1. *Initialise G to all of level n of $\mathscr{P}_1(S^3)$. This means that G has no edges, the number of components c is just the number of vertices at level n, and the maximum level is $\ell = n$.*
2. *While $c > 1$, expand the graph as follows:*

[3]In general, a vertex at level n can have up to $2n$ distinct neighbours at level $(n + 1)$.

(a) *Construct all edges from vertices in G at level ℓ to (possibly new) vertices in $\mathscr{P}_1(S^3)$ at level $\ell + 1$. Insert these edges and their endpoints into G.*

(b) *Update the number of components c, and increment ℓ by one.*

3. *Once we have $c = 1$, output the final bound $H_n = \ell - n$ and terminate.*

In step 2(a) we construct edges by performing 2-3 moves. We only construct edges from vertices *already* in G, which means we only work with a small portion of level ℓ for each $\ell > n$. In step 2(b) we use union-find to update the number of components in small time complexity.

It is clear that Algorithm 10 is correct for any $n \geq 3$: once we have $c = 1$ the subgraph G is connected, which means there is a path from any vertex at level n to any other vertex at level n. By Theorem 1 at least one such vertex allows a 3-2 move, and so any vertex at level n has a simplification path of height $\leq \ell$.

However, it is not clear that Algorithm 10 terminates: it might be that *every* simplification path from some vertex at level n passes through vertices that we never construct at higher levels $\ell > n$. Happily it does terminate for all $3 \leq n \leq 9$, giving an output of $H_2 = 2$ each time. Table 3 shows how the number of components c changes throughout the algorithm in each case.

Input level n	3	4	5	6	7	8	9
Value of c when $\ell = n$	20	128	1 297	13 660	169 077	2 142 197	28 691 150
Value of c when $\ell = n + 1$	8	50	196	1 074	7 784	64 528	557 428
Value of c when $\ell = n + 2$	1	1	1	1	1	1	1
Final bound H_n	**2**	**2**	**2**	**2**	**2**	**2**	**2**

Table 3: Results obtained when running Algorithm 10 for $3 \leq n \leq 9$

It is straightforward to show that the space and time complexities of Algorithm 10 are linear and log-linear respectively in the number of vertices in G (other small polynomial factors in n and ℓ also appear). Nevertheless, the memory requirements for $n = 8$ were found to be extremely large in practice (∼29 GB), and for $n = 9$ they were too large for the algorithm to run (estimated at 400–500 GB). In the case of $n = 9$ a *two-phase* approach was necessary:

1. Use Algorithm 10 for the transition from level n to level $n + 1$, and terminate if $H_n = 1$.

2. From each vertex v at level $n + 1$, try all possible *combinations* of a 2-3 move followed by a 3-2 move. Let w be the endpoint of such a combination (so w is also a vertex at level $n + 1$). If $w \in G$ then merge the components and decrement c if necessary. Otherwise do nothing (since w would never have been constructed in the original algorithm).

3. If $c = 1$ after this procedure then output $H_n = 2$; otherwise terminate with no result.

It is important to note that, if this two-phase approach *does* output a result, it will always be the same result as Algorithm 10. Essentially Step 2 simulates the transition from level $n + 1$ to $n+2$ in the original algorithm, with the advantage of a much smaller memory footprint (since it does not store any vertices at level $n + 2$), but with the disadvantage that it cannot move on to level $n + 3$ if required (and so it cannot output any result if $H_n > 2$).

Of course by the time we reach $n = 9$ there are reasons to suspect that $H_n = 2$ (following the pattern for $3 \leq n \leq 8$), and so this two-phase method seems a reasonable (and ultimately successful) approach. For $n = 9$ the memory consumption was ∼50 GB, which was (just) within the capabilities of the host machine.

4.2 Bounding path lengths

Our next task is to compute bounds L_n so that, from every vertex at level n of $\mathscr{P}_1(S^3)$, there is some simplification path of length $\leq L_n$. Once again we compute L_n for $3 \leq n \leq 9$.

Because it is infeasible to perform arbitrary breadth-first searches through $\mathscr{P}_1(S^3)$, we only consider paths that can be expressed as a series of *jumps*, where each jump involves a pair of 2-3 moves followed by a pair of 3-2 moves. This keeps the search space and memory usage small: we always stay within levels n, $n+1$ and $n+2$, and we never need to explicitly store any vertices above level n. On the other hand, it means that our bounds L_n are very rough—there could be much shorter simplification paths that we do not detect.

Algorithm 11 (Algorithm for computing L_n)**.** *First identify the set I of all vertices at level n of $\mathscr{P}_1(S^3)$ that have an edge running down to level $n-1$. Then conduct a breadth-first search across level n, beginning with the vertices in I and using jumps as the steps in this breadth-first search. If j is the maximum number of jumps required to reach any vertex in level n from the initial set I, then output the final bound $L_n = 4j+1$.*

To identify the initial set I we simply attempt to perform 3-2 moves. When we process each vertex v, we must enumerate all jumps out from v; that is, all combinations of two 2-3 moves followed by two 3-2 moves. The number of such combinations is $O(n^4)$ in general.

This time we can guarantee both correctness and termination if $3 \leq n \leq 9$. Because $n \geq 3$ the initial set I is non-empty (Theorem 1), and from our height experiments in Section 4.1 we know that our search will eventually reach all of level n. It follows that every vertex at level n of $\mathscr{P}_1(S^3)$ has a path of length $\leq 4j$ to some $v \in I$, and therefore a simplification path of length $\leq 4j+1$. Table 4 shows how the search progresses for each n.

Input level n	3	4	5	6	7	8	9
Size of I	3	46	504	6 975	91 283	1 300 709	18 361 866
Vertices remaining	17	82	793	6 685	77 794	841 488	10 329 284
Vertices remaining after 1 jump	3	1	19	75	496	4 222	31 250
Vertices remaining after 2 jumps	0	0	1	1	0	6	12
Vertices remaining after 3 jumps	0	0	0	0	0	0	0
Final bound L_n	**9**	**9**	**13**	**13**	**9**	**13**	**13**

Table 4: Results obtained when running Algorithm 11 for $3 \leq n \leq 9$

This time the space and time complexities are linear and log-linear respectively in the number of vertices at level n (again with further polynomial factors in n). This is considerably smaller than the number of vertices processed in Algorithm 10, and so for Algorithm 11 memory is not a problem: the case $n = 9$ runs in under 4 GB.

4.3 Parallelisation and performance

For $n = 9$, both Algorithms 10 and 11 have lengthy running times: Algorithm 10 requires a very large number of vertices to be processed at levels 9, 10 and 11 of the Pachner graph, and Algorithm 11 spends significant time enumerating the $O(n^4)$ available jumps from each vertex.

We can parallelise both algorithms by processing vertices simultaneously (in step 2 of Algorithm 10, and during each stage of the breadth-first search in Algorithm 11). We must be careful however to serialise any updates to the graph.

The experiments described here used an 8-core 2.93 GHz Intel Xeon X5570 CPU with 72 GB of RAM (using all cores in parallel). With the serialisation bottlenecks, Algorithms 10 and 11 achieved roughly 90.5% and 98.5% CPU utilisation for the largest case $n = 9$, and ran for approximately 6 and 15 days respectively. All code was written using the topological software package *Regina* [1, 3].

5 Discussion

As we have already noted, the bounds obtained in Section 4 are astonishingly small. Although we only consider $n \leq 9$, this is not a small sample: the census includes $\sim$ 150 million triangulations including $\sim$ 31 million one-vertex 3-spheres; moreover, nine tetrahedra are enough to build complex and interesting topological structures [4, 13]. Our results lead us to the following conjectures:

Conjecture 1. *From any vertex at any level $n \geq 3$ of the graph $\mathscr{P}_1(S^3)$ there is a simplification path of height ≤ 2.*

If true, this result (combined with Theorem 4) would reduce Mijatović's bound in Theorem 2 from $\exp(O(n^2))$ to $\exp(O(n \log n))$ for one-vertex triangulations of the 3-sphere. Furthermore, it would help explain why 3-sphere triangulations are so easy to simplify in practice.

There are reasons to believe that a proof might be possible. As a starting point, a simple Euler characteristic argument shows that every closed 3-manifold triangulation has an edge of degree ≤ 5; using *at most two* "nearby" 2-3 moves, this edge can be made degree three (the setting for a possible 3-2 simplification). The details will appear in the full version of this paper.

Conjecture 2. *From any vertex at any level $n \geq 3$ of the graph $\mathscr{P}_1(S^3)$ there is a simplification path of length ≤ 13.*

This is a bolder conjecture, since the length experiments are less consistent in their results. However, the fact remains that every 3-sphere triangulation of size $n \leq 9$ can be simplified after just three jumps, and this number does not rise between $n = 5$ and $n = 9$.

If true, this second conjecture would yield an immediate polynomial-time 3-sphere recognition algorithm: for any triangulation of size $n \geq 3$ we can enumerate all $O(n^{4\times 3})$ combinations of three jumps, and test each resulting triangulation for a 3-2 move down to $n - 1$ tetrahedra. By repeating this process $n - 2$ times, we will achieve either a recognisable 2-tetrahedron triangulation of the 3-sphere, or else a proof that our input is not a 3-sphere triangulation.

Even if Conjecture 2 is false and the length bounds do grow with n, this growth rate appears to be extremely slow. A growth rate of $L_n \in O(\log n)$ or even $O(\sqrt{n})$ would still yield the first known sub-exponential 3-sphere recognition algorithm (using the same procedure as above), which would be a significant theoretical breakthrough in algorithmic 3-manifold topology.

The author is grateful to the Australian Research Council for their support under the Discovery Projects funding scheme (project DP1094516). Computational resources used in this work were provided by the Queensland Cyber Infrastructure Foundation.

References

[1] Benjamin A. Burton, *Regina: Normal surface and 3-manifold topology software*, `http://regina.sourceforge.net/`, 1999–2009.

[2] ———, *Face pairing graphs and 3-manifold enumeration*, J. Knot Theory Ramifications **13** (2004), no. 8, 1057–1101.

[3] ———, *Introducing Regina, the 3-manifold topology software*, Experiment. Math. **13** (2004), no. 3, 267–272.

[4] ———, *Enumeration of non-orientable 3-manifolds using face-pairing graphs and union-find*, Discrete Comput. Geom. **38** (2007), no. 3, 527–571.

[5] ———, *The complexity of the normal surface solution space*, SCG '10: Proceedings of the Twenty-Sixth Annual Symposium on Computational Geometry, ACM, 2010, pp. 201–209.

[6] ———, *Optimizing the double description method for normal surface enumeration*, Math. Comp. **79** (2010), no. 269, 453–484.

[7] ———, *Quadrilateral-octagon coordinates for almost normal surfaces*, Experiment. Math. **19** (2010), no. 3, 285–315.

[8] Nathan M. Dunfield and William P. Thurston, *Finite covers of random 3-manifolds*, Invent. Math. **166** (2006), no. 3, 457–521.

[9] Masao Hara, Seiichi Tani, and Makoto Yamamoto, UNKNOTTING *is in* $\mathbf{AM} \cap \text{co-}\mathbf{AM}$, SODA '05: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2005, pp. 359–364.

[10] William Jaco, David Letscher, and J. Hyam Rubinstein, *Algorithms for essential surfaces in 3-manifolds*, Topology and Geometry: Commemorating SISTAG, Contemporary Mathematics, no. 314, Amer. Math. Soc., Providence, RI, 2002, pp. 107–124.

[11] William Jaco and J. Hyam Rubinstein, *0-efficient triangulations of 3-manifolds*, J. Differential Geom. **65** (2003), no. 1, 61–168.

[12] William Jaco and Jeffrey L. Tollefson, *Algorithms for the complete decomposition of a closed* 3-*manifold*, Illinois J. Math. **39** (1995), no. 3, 358–406.

[13] Bruno Martelli and Carlo Petronio, *Three-manifolds having complexity at most 9*, Experiment. Math. **10** (2001), no. 2, 207–236.

[14] Sergei Matveev, *Algorithmic topology and classification of 3-manifolds*, Algorithms and Computation in Mathematics, no. 9, Springer, Berlin, 2003.

[15] Sergei V. Matveev, *Complexity theory of three-dimensional manifolds*, Acta Appl. Math. **19** (1990), no. 2, 101–130.

[16] ———, *Computer recognition of three-manifolds*, Experiment. Math. **7** (1998), no. 2, 153–161.

[17] Aleksandar Mijatović, *Simplifying triangulations of S^3*, Pacific J. Math. **208** (2003), no. 2, 291–324.

[18] ———, *Triangulations of Seifert fibred manifolds*, Math. Ann. **330** (2004), no. 2, 235–273.

[19] ———, *Simplical structures of knot complements*, Math. Res. Lett. **12** (2005), no. 5-6, 843–856.

[20] ———, *Triangulations of fibre-free Haken 3-manifolds*, Pacific J. Math. **219** (2005), no. 1, 139–186.

[21] Udo Pachner, *P.L. homeomorphic manifolds are equivalent by elementary shellings*, European J. Combin. **12** (1991), no. 2, 129–145.

[22] J. Hyam Rubinstein, *An algorithm to recognize the* 3-*sphere*, Proceedings of the International Congress of Mathematicians (Zürich, 1994), vol. 1, Birkhäuser, 1995, pp. 601–611.

[23] Abigail Thompson, *Thin position and the recognition problem for S^3*, Math. Res. Lett. **1** (1994), no. 5, 613–630.

Benjamin A. Burton
School of Mathematics and Physics, The University of Queensland
Brisbane QLD 4072, Australia
(bab@maths.uq.edu.au)

Appendix: Additional proofs

In this appendix we offer full proofs of Theorem 4 and Lemma 5. These proofs were omitted from the main text due to space considerations.

Theorem 4. *The number of distinct isomorphism classes of 3-manifold triangulations of size n grows at an asymptotic rate of* $\exp(\Theta(n \log n))$.

Proof. An upper bound of $\exp(O(n \log n))$ is easy to obtain. If we count all possible gluings of tetrahedron faces, without regard for isomorphism classes or other constraints (such as the need for the triangulation to represent a closed 3-manifold), we obtain an upper bound of

$$[(4n-1) \times (4n-3) \times \cdots \times 3 \times 1] \cdot 6^{2n} < (4n)^{2n} \cdot 6^{2n} \in \exp(O(n \log n)).$$

Proving a lower bound of $\exp(\Omega(n \log n))$ is more difficult—the main complication is that most pairwise identifications of tetrahedron faces do not yield a 3-manifold at all [8]. We work around this by first counting *2-manifold* triangulations (since closed 2-manifolds are much easier to obtain), and then giving a construction that "fattens" these into 3-manifold triangulations without introducing any unwanted isomorphisms.

To create a 2-manifold triangulation of size $2m$ (the size must always be even), we identify the $6m$ edges of $2m$ distinct triangles in pairs. Any such identification will always yield a closed 2-manifold (that is, nothing can "go wrong", in contrast to the three-dimensional case). Each of the $3m$ pairs of edges can be identified using one of two orientations, and so the number of *labelled* 2-manifold triangulations is precisely

$$[(6m-1) \times (6m-3) \times \cdots \times 3 \times 1)] \cdot 2^{3m}.$$

Each isomorphism class can contain at most $(2m)! \cdot 6^{2m}$ labelled triangulations, and so the number of distinct *isomorphism classes* of 2-manifold triangulations is bounded below by

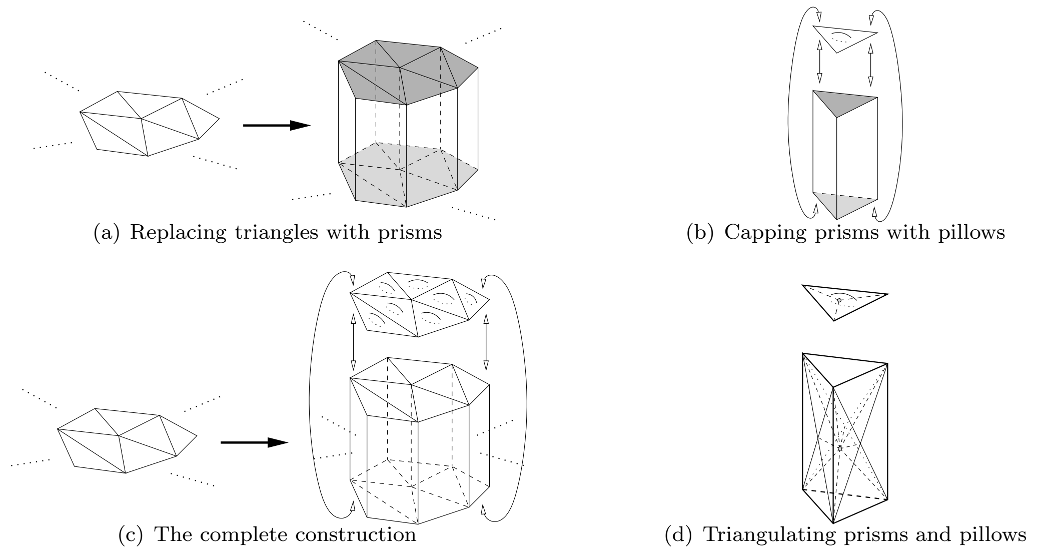
$$\begin{aligned}\frac{[(6m-1) \times (6m-3) \times \cdots \times 3 \times 1)] \cdot 2^{3m}}{(2m)! \cdot 6^{2m}} &> [(6m-1) \times (6m-3) \times \cdots \times (4m+1)] \cdot \left(\tfrac{8}{36}\right)^m \\ &> (4m)^m \cdot \left(\tfrac{8}{36}\right)^m \\ &\in \exp(\Omega(m \log m)).\end{aligned}$$

We fatten a 2-manifold triangulation into a 3-manifold triangulation as follows. Let F denote the closed 2-manifold described by the original triangulation.

1. Replace each triangle with a prism and glue the vertical faces of adjacent prisms together, as illustrated in Figure 4(a). This represents a *bounded* 3-manifold, which is the product space $F \times I$.

2. Cap each prism at both ends with a triangular pillow, as illustrated in Figure 4(b). The two faces of each pillow are glued to the top and bottom of the corresponding prism, effectively converting each prism into a solid torus. This produces the *closed* 3-manifold $F \times S^1$, and the complete construction is illustrated in Figure 4(c).

3. Triangulate each pillow using two tetrahedra, which are joined along three internal faces surrounding an internal vertex. Triangulate each prism using 14 tetrahedra, which again all meet at an internal vertex. Both triangulations are illustrated in Figure 4(d).



(a) Replacing triangles with prisms

(b) Capping prisms with pillows

(c) The complete construction

(d) Triangulating prisms and pillows

Figure 4: Fattening a 2-manifold triangulation into a 3-manifold triangulation

If the original 2-manifold triangulation uses $2m$ triangles, the resulting 3-manifold triangulation uses $n = 32m$ tetrahedra. Moreover, if two 3-manifold triangulations obtained using this construction are isomorphic, the original 2-manifold triangulations must also be isomorphic. The reason for this is as follows:

- Any isomorphism between two such 3-manifold triangulations must map triangular pillows to triangular pillows. This is because the internal vertex of each triangular pillow meets only two tetrahedra, and no other vertices under our construction have this property.
- By "flattening" the triangular pillows into 2-dimensional triangles, we thereby obtain an isomorphism between the underlying 2-manifold triangulations.

It follows that, for $n = 32m$, we obtain a family of $\exp(\Omega(m \log m)) = \exp(\Omega(n \log n))$ pairwise non-isomorphic 3-manifold triangulations.

This result is easily extended to $n \not\equiv 0 \bmod 32$. Let V_n denote the number of distinct isomorphism classes of 3-manifold triangulations of size n.

- Each triangulation of size n has at least $n-1$ distinct 2-3 moves available (since any face joining two distinct tetrahedra defines a 2-3 move, and there are at least $n-1$ such faces).
- On the other hand, each triangulation of size $n+1$ has at most $6(n+1)$ distinct 3-2 moves available (since each 3-2 move is defined by an edge that meets three distinct tetrahedra, and the triangulation has at most $6(n+1)$ edges in total).

It follows that $V_{n+1} \geq V_n \cdot \frac{n-1}{6(n+1)} \geq V_n/18$ for any $n > 1$. This gives $V_{32m+k} \geq V_{32m}/18^{31}$ for sufficiently large m and all $0 \leq k < 32$, and so we obtain $V_n \in \exp(\Omega(n \log n))$ with no restrictions on n. $\square$

Remark. Of course, we expect that $V_{n+1} \gg V_n$ (and indeed we see this in the census). The bounds that we use to show $V_{n+1} \geq V_n/18$ in the proof above are very loose, but they are sufficient for the asymptotic result that we seek.

Lemma 5. *For any triangulation $\mathcal{T}$ of size n, there are precisely $24n$ canonical labellings of $\mathcal{T}$, and these can be enumerated in $O(n^2 \log n)$ time.*

Proof. For $n = 1$ the result is trivial, since all $24 = 4!$ possible labellings are canonical. For $n > 1$ we observe that, if we choose (i) any one of the n tetrahedra to label as tetrahedron 1, and (ii) any one of the 24 possible labellings of its four vertices, then there is one and only one way to extend these choices to a canonical labelling of $\mathcal{T}$.

To see this, we can walk through the list of faces $F_{1,1}, F_{1,2}, F_{1,3}, F_{1,4}, F_{2,1}, \ldots, F_{n,4}$, where $F_{t,i}$ represents face i of tetrahedron t. The first face amongst $F_{1,1}, \ldots, F_{1,4}$ that is joined to an unlabelled tetrahedron must in fact be joined to tetrahedron 2 using the identity map; this allows us to deduce tetrahedron 2 as well as the labels of its four vertices.

We inductively extend the labelling in this manner: once we have labelled tetrahedra $1, \ldots, k$ and their corresponding vertices, the first face amongst $F_{1,1}, \ldots, F_{k,4}$ that is joined to an unlabelled tetrahedron must give us tetrahedron $k+1$ and the labels for its four vertices (again using the identity map). The resulting labelling is canonical, and all of the labels can be deduced in $O(n \log n)$ time using a single pass through the list $F_{1,1}, \ldots, F_{n,4}$. The $\log n$ factor is required for manipulating tetrahedron labels, each of which requires $O(\log n)$ bits.

It follows that there are precisely $24n$ canonical labellings of $\mathcal{T}$, and that these can be enumerated in $O(n^2 \log n)$ time using $24n$ iterations of the procedure described above. $\square$