# Retracted Spun Normal Surfaces

**A MASTER'S PROJECT**
**SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL**
**OF THE UNIVERSITY OF MINNESOTA**
**BY**

**William Koller**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS**
**FOR THE DEGREE OF**
**Master of Science**

**Advisor: Dr. Neil Hoffman**

**April 14, 2025**

**April 14, 2025**

# Acknowledgements

I would like to thank my advisor, Professor Neil Hoffman, for his immense support, guidance, and patience throughout my project. His positive attitude and eagerness to help throughout my challenges was instrumental to my success. I am exceedingly grateful for having had the opportunity to work with him.

I would also like to thank all of the faculty and staff I had the opportunity to interact with throughout my time in the program. Their assistance, teachings, and warmth had positive impact on my journey, and I am grateful for the experience.

# Abstract

Normal surfaces and spun normal surfaces in ideal triangulations are described by the quantities of disk types belonging to the surface. In the case of spun normal surfaces, there may be infinitely many triangles of some types. In this paper we introduce the concept of a retracted spun normal surface. A retracted spun normal surface comes from retracting a spun normal surface, resulting in a finite surface (possibly) with boundary while maintaining many properties of the spun normal surface. The finiteness of retracted spun normal surfaces allows for the algorithms introduced in this paper to compute various properties of the surface. This paper shows that every spun normal surface has a corresponding retracted spun normal surface, and provides an implementation to construct a retracted spun normal surface from a spun normal surface. Additionally, implementations are provided for computing the connectedness, boundary slopes, number of boundary components, euler characteristic, and orientability of retracted spun normal surfaces, which together provide sufficient information to classify retracted spun normal surfaces.

# Contents

# 1 Introduction

Given a triangulation $\mathcal{T}$, the vector representation of closed surfaces can found using existing methods. A spun normal surface, which is not closed, has infinitely many triangles resulting in some coordinates being infinite. In this paper we provide methods for finding retracted spun normal surface solutions by truncating the spun normal surface with a boundary and giving a description of a finite surface and the boundary. We also provide a method for classifying the retracted spun normal surface.

**Definition 1.1.** (Normal Disk) Let $\mathcal{T}$ be a 3D triangulation and $T$ be a tetrahedron in $\mathcal{T}$. A normal disk is $T$ is a topological disk embedded in $T$ which does not contain any of the vertices in $T$ and lies in the interior of $T$ and whose boundary lies on either three of the faces of $T$ or four of the faces of $T$. A disk with a boundary on exactly three faces of $T$ is called a triangular disk and a disk with boundary on four faces of $T$ is called a quadrilateral disk. Finally, we say a triangular disk surrounds a vertex $v$ if the disk cuts $v$ off from the other three vertices in the tetrahedron.

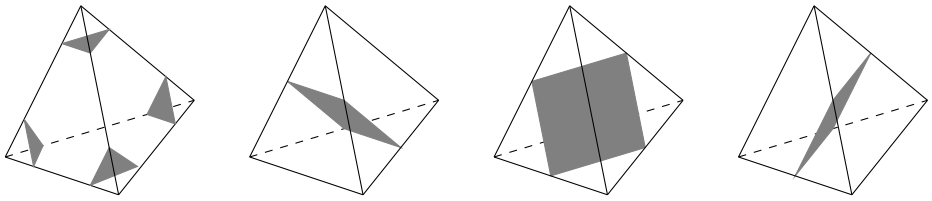

Figure 1: The normal disk types in a tetrahedron.

**Definition 1.2.** (Normal Surface and Normal Surface Solution) Let $\mathcal{T}$ be a 3D triangulation comprised of $t$ tetrahedra indexed by $0, 1, ..., t-1$. A *normal surface* is a finite collection of disjoint normal disks in $\mathcal{T}$ such that each edge bounding a normal disk is identified with another edge bounding a normal disk via piece-wise linear homeomorphism induced by the face gluings of $\mathcal{T}$. A *normal surface solution vector* is a vector in $\mathbb{R}^{7t}$ where each coordinate corresponds to a normal disk type. Let $t_{i,j}$ be the number of normal triangle disks in tetrahedron $i$ surrounding vertex $j \in \{0, 1, 2, 3\}$ of tetrahedron $i$. Let $q_{i,0j}$ be the number of normal quadrilateral disks in tetrahedron $i$ which do not intersect the edge $[0j]$, $j \in \{1, 2, 3\}$ in tetrahedron $i$. Then the normal surface solution vector is

$$\begin{aligned} v = (&t_{0,0}, t_{0,1}, t_{0,2}, t_{0,3}, q_{0,01}, q_{1,02}, q_{1,03}, \\ &t_{1,0}, t_{1,1}, t_{1,2}, t_{1,3}, q_{1,01}, q_{1,02}, q_{1,03}, \\ &\dots \qquad\qquad\qquad\qquad\qquad\qquad , q_{t-1,03}) \end{aligned}$$

**Notation.** It will prove convenient to sometimes describe a quadrilateral disk by any edge that is not incident to it and also to not worry about the orientation

of the edge. For example, $q_{i,01} = q_{i,10} = q_{i,23} = q_{i,32}$. We note that this notation convention still unambiguously identifies a quadrilateral type. We will also use $I = \{(0,01),(0,02),(0,03),...(t-1,01),(t-1,02),(t-1,03)\}$ as an indexing set of the quadrilateral disk types and denote a disk type by $q_i$ with $i \in I$.

**Definition 1.3.** (Standard Matching Equations) Let $\mathcal{T}$ be a triangulation and $F_i$,$F_j$ be a pair of tetrahedral faces belonging tetrahedra $T_i$ and $T_j$ which are glued together in $\mathcal{T}$. Let a $a \in \{0,1,2,3\}$ be a vertex in $F_i$ and $a' \in \{0,1,2,3\}$ be the vertex in $F_j$ which is identified with $a$ via the face gluing $i(abc) = j(a'b'c')$. There is exactly one triangle disk type and one quadrilateral disk type in each $T_i$ and $T_j$ with an edge on face $F_i$ and $F_j$ which cuts off vertex $a$ and vertex $a'$ in $T_i$ and $T_j$ respectively. The *standard matching equation* corresponding to the gluing between $F_i$ and $F_j$ and vertices $a$ and $a'$ is

$$t_{i,a} + q_{i,ad} = t_{j,a'} + q_{j,a'd'}$$

The set of solutions to the standard matching equations represents the normal surfaces that can exist in $\mathcal{T}$.

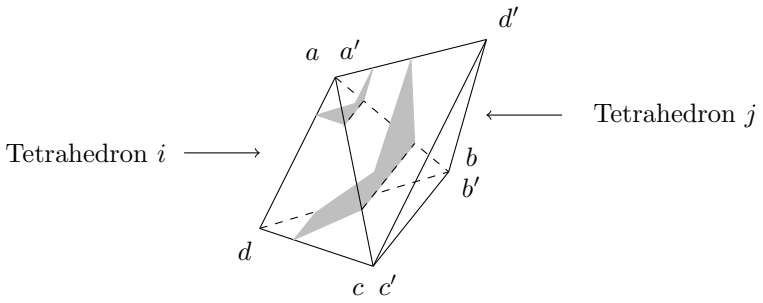


Figure 2: Standard matching equations example.
The equation is $t_{i,a} + q_{i,ad} = t_{j,a'} + q_{j,a'd'}$ where
$t_{i,a} = 1$, $q_{i,ad} = 1$, $t_{j,a'} = 2$, $q_{j,a'd'} = 0$

Quadrilateral ($Q$) matching equations (defined in section 2) can be obtained by relating quantities of quadrilateral disk types indecent to an edge of a triangulation $\mathcal{T}$. Similar to the standard matching equations, the $Q$ matching equations are homogeneous and the solution set contains vector representations of the normal surfaces that can exist in a triangulation $\mathcal{T}$. Unlike the standard matching equations where the solutions are all normal surfaces, the solution set to the $Q$-matching equations also contains solutions representing spun normal surfaces which are not normal surfaces because they contain infinitely many triangular disks. However, solutions to the $Q$-matching equations have the advantage of being $3t$ dimensional where $t$ is the number of tetrahedron, so finding solutions is more efficient than solving the standard matching equations which are of dimension $7t$.

### 1.1 Summary of results

The main results of this report are contained in Section 6.

We define a retracted spun normal surface which is a finite subset of a spun normal surface which retains much of the information about the spun normal surface. Since the retracted spun normal surface is finite, computing the properties is more manageable. For example, the culminating result of this report (Corollary 6.10) classifies the topological type of the retracted spun normal surface.

## 2 Background

**Definition 2.1.** (2D Triangulation) A *2D Triangulation* is a finite collection of triangles $t_1, \ldots, t_n$, where some or all of the $3n$ triangle edges are identified in pairs and where the resulting topological space is a compact 2-manifold.

**Definition 2.2.** (3D Triangulation) A *3D Triangulation or triangulation* is a finite collection of tetrahedra $T_1, \ldots, T_t$, where

(i) some or all of the $4t$ tetrahedron faces are identified in pairs and each face belongs to at most one pair and no face is glued to itself and

(ii) each equivalence class of edges in the triangulation can be given a local orientation, that is no edge is glued to itself.

Given a triangulation $\mathcal{T}$ the pairs of faces which are identified lead to vertices of the tetrahedra being identified.

**Notation.** A face gluing between tetrahedra $T_i$ and $T_j$ that identifies the ordered list of vertices $a, b, c$ in $T_i$ with the ordered list $a', b', c'$ in $T_j$ can be recorded as $i(abc) = j(a'b'c')$. This agrees with the notation convention of [2].

Since each triangle disk corresponds to exactly one vertex in the triangulation it will serve useful to have some nomenclature to talk about the triangle disks surrounding a vertex in a triangulation. Many of the algorithm in this report will use information about this collection of disks.

**Definition 2.3.** (Vertex Link) Let $\mathcal{T}$ be a triangulation and $v$ be some vertex of $\mathcal{T}$. The *vertex link* of $v$ is the normal surface containing each normal triangular disk surrounding $v$ and no other disks.

**Definition 2.4.** (Ideal Triangulation) A *ideal triangulation* is a 3D triangulation where each vertex link is a torus.

In this report we will be concerning ourselves only with 3D triangulations which are ideal triangulations as the core methods rely on the vertex link being a torus.

**Definition 2.5.** (1-Skeleton) A *1-skeleton* of a triangulation $\mathcal{T}$ is the union of edges and vertices in $\mathcal{T}$.

**Definition 2.6.** (2-Skeleton) A *2-skeleton* of a triangulation $\mathcal{T}$ is the union of triangles, edges, and vertices in $\mathcal{T}$.

**Definition 2.7.** (locally a-oriented, locally b-oriented, locally c-oriented quadrilaterals of an oriented tetrahedral edge) Let $\mathcal{T}$ be a 3D triangulation. Let $e = [a, b]$ be an oriented edge in a tetrahedron $T_i$ of $\mathcal{T}$. The quadrilateral disk type $q_{i,ab}$ in $T_i$ is not incident to the edge $e$ and we say $q_{i,ab}$ is a *c-oriented* quadrilateral disk with respect to $e$. Generically denote one of the other two quadrilateral disk types in $T_i$ by $q_i'$ (so $q_i' \in \{q_{i,ac}, q_{i,ad}\}$), which each have a corner with two edges incident with $e$. If rotating counter-clockwise about $e$ through $T$ connects face $f_0$ of $T$ to the face $f_1$ of $T$, we say a quadrilateral $q_i'$ is *locally a-oriented relative to e* if the normal arc in $f_0$ cuts off vertex $a$ (and consequently the normal arc of $q_i'$ in face $f_1$ cuts off vertex $b$). A quadrilateral $q_i'$ is *locally b-oriented relative to e* if the normal arc in $f_0$ cuts off vertex $b$ (and consequently the normal arc of $q_i'$ in face $f_1$ cuts off vertex $a$).

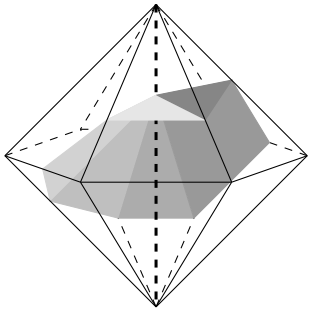

Figure 3: Example of surface through an edge in a triangulation

**Definition 2.8.** (Q-Matching Equations and spun normal surface solution vector) Let $\mathcal{T}$ be a triangulation with $t$ tetrahedra and $e_k$ be an edge class in $\mathcal{T}$, that is a circularly ordered list $e_{k,j}$ of tetrahedral edges identified by the face-pairings of $\mathcal{T}$. Arbitrarily assign a orientation to $e_k$. Start with $V_k = (\epsilon_{k,i})$ as the $3t$ dimensional zero vector where each entry corresponds to a quadrilateral disk type (with $i \in I = \{(0, 01), (0, 02), ..., (t-1, 01), (t-1, 02), (t-1, 03)\}$). Rotate about $e_k$ counter-clockwise through the tetrahedra incident to $e_k$. For each tetrahedral edge $e_{k,j}$, denote the tetrahedron contained $e_{k,j}$ by $T$. For each quadrilateral disk type $q_i$ in $T$ if $q_i$ is locally a-oriented relative to $e_{k,j}$, add $+1$ to the coordinate $i$ in $V_k$. If $q_i$ is locally $b$-oriented, add $-1$ to the corresponding coordinate in $V_k$. Finally, if $q_i$ is locally $c$-oriented (does not touch $e_k$), add 0 or do nothing. For each $e_k$, we have the following equation

$$\sum_{i \in I} \epsilon_{k,i} q_i = 0$$

where the $q_i$ is a count of the number of quadrilateral disks of type $i$. The *Q-Matching equations* are the system of linear equations constructed by defining

one equation as above for each edge $e_k$ in $\mathcal{T}$. If $S \in \mathbb{R}^{3t}$ is a solution to this system we say $S$ is a solution vector to the Q-Matching equations or a *spun normal surface solution vector*.

It is shown by Tollefson [5] that an admissible solution to the $Q$-matching equations uniquely determines a normal surface without trivial components. We will discuss converting between a solution vector to these equations and a solution vector to the standard matching equations later.

**Definition 2.9.** (Spun Normal Surface) Let $\mathcal{T}$ be an ideal triangulation comprised of $t$ tetrahedra. A *spun normal surface* is a collection of disjoint normal disks in $\mathcal{T}$ such that each edge bounding a normal disk is identified with another edge bounding a normal disk via piece-wise linear homeomorphism induced by the face gluings of $\mathcal{T}$ and the quadrilateral coordinates corresponding to the quantities of quadrilateral disk types satisfies the $Q$-matching equations.

A key difference between spun normal surfaces and normal surfaces is that there exist spun normal surfaces that do not satisfy the standard matching equations. Such surfaces will contain infinitely many copies of one or more vertex links in the corresponding triangulation. The core of this report is about how to cut such surfaces off in the vertex links resulting in a surface with finitely many disks and information about how the surface was cut. To find a satisfactory place to cut a spun normal surface within a vertex link we will find the meridian and longitude of the vertex link (see following definitions).

**Definition 2.10.** (Meridian) Let $\mathcal{T}$ be a 2D triangulation of a torus and $v$ be a vertex in $\mathcal{T}$. A *meridian* $M$ at $v$ in $\mathcal{T}$ is a shortest cycle in $\mathcal{T}$ containing $v$, which when removed from $\mathcal{T}$, results in a 2D triangulation $\mathcal{T}'$ which is still connected.

**Definition 2.11.** (Longitude) Let $\mathcal{T}$ be a 2D triangulation of a torus, $M$ be a meridian in $\mathcal{T}$, and $u$ be a vertex in $M$. A *longitude* $\Lambda$ of $M$ and $u$ in $\mathcal{T}$ is a shortest cycle in $\mathcal{T}$ containing $u$ such that $\Lambda \neq M$ and when $M$ and $\Lambda$ are both removed from $\mathcal{T}$, the resulting $2D$-triangulation is still connected.

Note that these definitions of meridian and longitude are specific definitions which serve useful to this report. More general definitions of meridian and longitude exist.

**Definition 2.12.** Given an triangulation $\mathcal{T}$, a *peripheral system of curves at an ideal vertex* is a meridian and longitude for the link of that vertex given in terms of normal arcs in $\mathcal{T}$. A *complete peripheral system* is comprised of a meridian and longitude for each ideal vertex of $\mathcal{T}$.

Given a triangulation $\mathcal{T}$ with $c$ ideal vertices the complete peripheral system $\Gamma$ will contain $c$ meridians and $c$ longitudes for a total of $2c$ curves.

**Definition 2.13.** (Retracted Spun Normal Surface and Retracted Spun Normal Surface Solution) Given a triangulation $\mathcal{T}$ with $c$ ideal vertices, $S$ a solution

to the Q-matching equations for $\mathcal{T}$, and a complete peripheral system of curves $\Gamma$ for $\mathcal{T}$, a *retracted spun normal surface solution* is a vector in $\mathbb{R}^{7t+2c}$ with $7t$ coordinates corresponding to the quantities of the $7t$ disk types and $2c$ coordinates corresponding to the quantities of boundary components corresponding to the $2c$ curves in $\Gamma$.

A *retracted spun normal surface* comes from a spun normal surface in which a vertex has been retracted creating a boundary on the surface. A *retracted spun normal surface solution* contains the quadrilateral and triangular disk coordinates $S$ of the retracted spun normal surface, the meridian and longitude coordinates $\mu$,$\lambda$, and the meridian and longitude cycles corresponding to $\mu$ and $\lambda$.

Note that when each of the $2c$ peripheral system coordinates are zero, then the $7t$ coordinates corresponding to normal disk types satisfy the standard matching equations. So a retracted spun normal surface with zeros in each peripheral system coordinate is a normal surface.

## 2.1 Background algorithms

Here we introduce an example triangulation which will be used for each of the examples in this section.

| Tetrahedron | Face 012 | Face 013 | Face 023 | Face 123 |
|---|---|---|---|---|
| 0 | 3(012) | 1(021) | 2(023) | 1(123) |
| 1 | 0(031) | 3(321) | 2(013) | 0(123) |
| 2 | 3(032) | 1(023) | 0(023) | 3(310) |
| 3 | 0(012) | 2(321) | 2(021) | 1(310) |

The Q-matching equations for the above triangulation are:

$$2q_{0,02} - 2q_{0,03} + q_{1,01} - q_{1,02} + q_{2,01} - 2q_{2,02} + q_{2,03} - q_{3,01} + q_{3,03} = 0$$

$$-q_{0,01} + q_{0,03} + q_{2,01} - q_{2,03} + q_{3,01} - q_{3,03} = 0$$

$$q_{0,01} - q_{0,02} - q_{1,01} + q_{1,03} - 2q_{2,01} + 2q_{2,02} + q_{3,01} - q_{3,02} = 0$$

$$-q_{0,02} + q_{0,03} + q_{1,02} - q_{1,03} - q_{3,01} + q_{3,02} = 0$$

The standard matching equations are:

$$t_{0,0} + q_{0,03} - t_{3,0} - q_{3,03} = 0$$

$$t_{0,0} + q_{0,01} - t_{2,0} - q_{2,01} = 0$$

$$t_{0,0} + q_{0,02} - t_{1,0} - q_{1,03} = 0$$

$$t_{0,1} + q_{0,02} - t_{3,1} - q_{3,02} = 0$$

$$t_{0,1} + q_{0,01} - t_{1,1} - q_{1,01} = 0$$

$$t_{0,1} + q_{0,03} - t_{1,2} - q_{1,01} = 0$$

$$t_{0,2} + q_{0,01} - t_{3,2} - q_{3,01} = 0$$

$$t_{0,2} + q_{0,02} - t_{1,2} - q_{1,02} = 0$$
$$t_{0,2} + q_{0,03} - t_{2,2} - q_{2,03} = 0$$
$$t_{0,3} + q_{0,01} - t_{1,1} - q_{1,02} = 0$$
$$t_{0,3} + q_{0,03} - t_{1,3} - q_{1,03} = 0$$
$$t_{0,3} + q_{0,02} - t_{2,3} - q_{2,02} = 0$$
$$t_{1,0} + q_{1,01} - t_{2,0} - q_{2,02} = 0$$
$$t_{1,0} + q_{1,02} - t_{3,3} - q_{3,03} = 0$$
$$t_{1,1} + q_{1,03} - t_{3,2} - q_{3,02} = 0$$
$$t_{1,2} + q_{1,03} - t_{2,1} - q_{2,03} = 0$$
$$t_{1,3} + q_{1,01} - t_{3,1} - q_{3,01} = 0$$
$$t_{1,3} + q_{1,02} - t_{2,3} - q_{2,01} = 0$$
$$t_{2,0} + q_{2,03} - t_{3,0} - q_{3,01} = 0$$
$$t_{2,1} + q_{2,02} - t_{3,3} - q_{3,02} = 0$$
$$t_{2,1} + q_{2,01} - t_{3,3} - q_{3,01} = 0$$
$$t_{2,2} + q_{2,01} - t_{3,2} - q_{3,03} = 0$$
$$t_{2,2} + q_{2,02} - t_{3,1} - q_{3,03} = 0$$
$$t_{2,3} + q_{2,03} - t_{3,0} - q_{3,02} = 0$$

A central question is under what circumstances does a solution to the $Q$-matching equations also extend to a solution of the standard equations. A general method due to Burton will be described later in Algorithm 1, but this can be done concretely as in the next example.

**Example 2.14.** An example that is a solution to the Q-matching equations which can be promoted to a solution to the standard matching is given by:

$$S_q = (0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0)$$

To promote this to a standard solution (in $\mathbb{R}^{4(7)} = \mathbb{R}^{28}$), we will apply a generalization of Burton's method (Algorithm 1) to see if the solution corresponds to a normal surface, and if so, find the solution to the standard matching equations. The main idea of the algorithm is that for each equation, if one triangle coordinate in the equation is set, then the other triangle coordinate is determined by the set triangle coordinate and the quadrilateral coordinates in $S_q$. Additionally, since each equation has two triangle coordinates with opposite sign, if we find a solution with some coordinates having negative values we can shift each triangle coordinate up by the smallest negative value and we will have a solution with all non-negative coordinates. Since vertex links are connected, we can start at any arbitrary triangle coordinate in a given vertex link, set the coordinate to zero, and perform a depth first search through the vertex link

assigning the coordinates as we go. Note that this example triangulation has only one vertex link, so we can pick any triangle coordinate to start. We will start with $t_{0,0} = 0$. Then performing the depth first search we have the following

$$t_{0,0} + q_{0,03} - t_{3,0} - q_{3,03} = 0 \Rightarrow t_{3,0} = 0$$
$$t_{2,0} + q_{2,03} - t_{3,0} - q_{3,01} = 0 \Rightarrow t_{2,0} = 0$$
$$t_{1,0} + q_{1,01} - t_{2,0} - q_{2,02} = 0 \Rightarrow t_{1,0} = 1$$
$$t_{1,0} + q_{1,02} - t_{3,3} - q_{3,03} = 0 \Rightarrow t_{3,3} = 1$$
$$t_{2,1} + q_{2,02} - t_{3,3} - q_{3,02} = 0 \Rightarrow t_{2,1} = 1$$
$$t_{1,2} + q_{1,03} - t_{2,1} - q_{2,03} = 0 \Rightarrow t_{1,2} = 1$$
$$t_{0,1} + q_{0,03} - t_{1,2} - q_{1,01} = 0 \Rightarrow t_{0,1} = 1$$
$$t_{0,1} + q_{0,02} - t_{3,1} - q_{3,02} = 0 \Rightarrow t_{3,1} = 1$$
$$t_{1,3} + q_{1,01} - t_{3,1} - q_{3,01} = 0 \Rightarrow t_{1,3} = 1$$
$$t_{0,3} + q_{0,03} - t_{1,3} - q_{1,03} = 0 \Rightarrow t_{0,3} = 1$$
$$t_{0,3} + q_{0,01} - t_{1,1} - q_{1,02} = 0 \Rightarrow t_{1,1} = 1$$
$$t_{1,1} + q_{1,03} - t_{3,2} - q_{3,02} = 0 \Rightarrow t_{3,2} = 0$$
$$t_{0,2} + q_{0,01} - t_{3,2} - q_{3,01} = 0 \Rightarrow t_{0,2} = 0$$
$$t_{0,2} + q_{0,03} - t_{2,2} - q_{2,03} = 0 \Rightarrow t_{2,2} = 0$$
$$t_{1,3} + q_{1,02} - t_{2,3} - q_{2,01} = 0 \Rightarrow t_{2,3} = 1$$

Now each triangle coordinate has been set. We now check that these coordinates are consistent with each of the other standard matching equations. We find they are all consistent, so we have found the coordinates for a normal surface solution.

$$S = (0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0)$$

However, the method outlined above does not always succeed in finding a consistent solution as recorded by the following example.

**Example 2.15.** An example of a solution to the Q-matching equations, which can *not* be promoted to a solution to the standard matching equations is given by:

$$S_q = (0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0)$$

To see this will not work, we can try to proceed as above. Performing the generalization of Burton's method (Algorithm 1) we find the following coordinates

$$S = (2, 1, 1, 1, 2, 1, 1, 0, 2, 2, 1, 1, 2, 0, 1, 2)$$

When checking against all standard matching equations we find $S$ is inconsistent with each of the following equations.

$$t_{0,0} + q_{0,02} - t_{1,0} - q_{1,03} = 0$$

$$t_{1,3} + q_{1,02} - t_{2,3} - q_{2,01} = 0$$

$$t_{2,1} + q_{2,02} - t_{3,3} - q_{3,02} = 0$$

$$t_{2,2} + q_{2,02} - t_{3,1} - q_{3,03} = 0$$

$$t_{2,3} + q_{2,03} - t_{3,0} - q_{3,02} = 0$$

In particular, if we look the the pair of standard matching equations

$$t_{2,1} + q_{2,02} - t_{3,3} - q_{3,02} = 0$$

$$t_{2,1} + q_{2,01} - t_{3,3} - q_{3,01} = 0$$

and substitute the quadrilateral coordinates

$$t_{2,1} + 0 - t_{3,3} - 1 = 0$$

$$t_{2,1} + 0 - t_{3,3} - 0 = 0$$

We can see there does not exist a solution with finitely many triangles. The convention is to set all the triangle coordinates in this vertex link (which for this example is all of the triangle coordinates, since we have only one vertex link) to infinity. With this convention, we do have a solution which is consistent with repeatably gluing in new triangles to unpaired normal arcs. Let $v$ be the one vertex class in the triangulation. The surface is said to spin into $v$, hence the name spun normal surface. In future examples we will look at how the spun normal surface corresponding to $S_q$ can be cut off resulting in a retracted spun normal surface, which has finitely many disks and a boundary.

The implementation of Burton's Method (Algorithm 1, see also [4]) differs from the original implementation in [1, Algorithm 3.12] since it is used to check if a $Q$-matching solution vector corresponds to a normal surface or not. The original implementation takes only $Q$-matching coordinates which correspond to a normal surface, then returns the normal surface solution vector. The above implementation takes any $Q$-matching solution vector, runs the algorithm as usual, then checks if the coordinates are consistent with the standard matching equations. If they are, it returns `True`, otherwise it returns `False`. When returning `False`, we know the $Q$-matching solution vector corresponds to a spun normal surface.

In the case that Algorithm 1 returns `False`, we still can build a solution vector with finite coordinates, but this vector does not correspond to a a standard normal surface. Instead, it corresponds to a retracted spun normal surface. The remainder of this report will be dedicated to detailing the process of finding a solution in this case.

**Algorithm 1** A generalization of Burton's method (compare to [1, Algorithm 3.12]) for going from $Q$-matching coordinates to standard coordinates

**Input:** A triangulation $\mathcal{T}$ and a solution $S$ to the $Q$-matching equations.

Create an array to represent if a triangular disk type has been visited

**for** Each vertex $v \in \mathcal{T}$ **do**

Pick a triangular disk type $t$ surrounding $v$

Set the coordinate corresponding to $t$ to 0.

Perform a depth-first-search starting at $t$ through the adjacent triangles.

During the depth-first-search, when a triangle $t_i$ is visited coming from a coordinate $t_j$, there is a standard matching equation for $\mathcal{T}$ of the form $t_i + q_k = t_j + q_m$. Use the known values of $t_j, q_k, q_m$ to determine the value of $t_i$. Set the coordinate corresponding to $t_i$ to the determined value.

Find the minimum value $\lambda$ of the assigned triangle coordinates

Add $-\lambda$ to each of the assigned triangle coordinates.

**end for**

Return a boolean that records if these values are a solution to the standard matching equations and if so, return coordinates.

# 3 Meridian and Longitude

Notation: Vertices in the vertex link will be denoted $v_{i,j,k}$ where $t_{i,j}$ is the disk type and $[jk]$ is the edge of tetrahedron $i$ corresponding to the vertex in the link. For example, in Figure 4, the triangle $t_{0,0}$ has vertices $t_{0,0,1}$,$t_{0,0,2}$, and $t_{0,0,3}$, which are the intersections of $t_{0,0}$ and the edges of tetrahedron 0 labeled by [01],[02] and [03], respectively. Edges in the vertex link will be denoted $e_{i,j,k,l}$ where $e_{i,j,k,l}$ is the arc between vertices $v_{i,j,k}$ and $v_{i,j,l}$.

**Example 3.1.** Finding a meridian and longitude of the vertex link for the example triangulation
We will go through (Algorithm 2) to find the meridian of the vertex link for the example triangulation. The vertex link can be seen in Figure 4. The meridian finding algorithm implemented in this report is a breath first search through the 1-skeleton of the vertex link from some starting vertex $v$ which finds a shortest cycle $C$ such that after ungluing along the edges of $C$ the resulting 2D-triangulation is still connected. While going through the algorithm vertices will be marked as visited. When doing so, each identified vertex will also be considered visited. For example, if $v_{0,0,1}$ is marked as visited then each of $v_{0,3,2}, v_{1,0,2}, v_{2,3,2}, v_{3,0,1}$ are also marked visited. The algorithm allows us to choose any vertex in the vertex link to start at. We will start at the lexicographically smallest vertex, namely $v_{0,0,1}$.

Create an array to represent if a vertex has been visited.
Create an empty list of paths *paths*.
Mark $v_{0,0,1}$ as visited.
Add the path $[v_{0,0,1}]$ to the list of paths.
Create an empty list of paths *newpaths*.
Retrieve the next path $p$ from the paths (path $[v_{0,0,1}]$).
Retrieve the last vertex in $p$ (vertex $v_{0,0,1}$).
Create a new path $[v_{0,0,1}, e_{0,0,1,2}, v_{0,0,2}]$.
Check if $[v_{0,0,1}, e_{0,0,1,2}, v_{0,0,2}]$ is a cycle.
Since it is not a cycle, add it to *newpaths*.
Mark vertex $v_{0,0,2}$ as visited.
Create a new path $[v_{0,0,1}, e_{0,0,1,3}, v_{0,0,3}]$.
Check if $[v_{0,0,1}, e_{0,0,1,3}, v_{0,0,3}]$ is a cycle.
Since it is not a cycle, add it to *newpaths*.
Mark vertex $v_{0,0,3}$ as visited.
Create a new path $[v_{0,0,1}, e_{0,3,2,1}, v_{0,3,1}]$.
Check if $[v_{0,0,1}, e_{0,3,2,1}, v_{0,3,1}]$ is a cycle.
Since it is not a cycle, add it to *newpaths*.
Mark vertex $v_{0,3,1}$ as visited.
Create a new path $[v_{0,0,1}, e_{0,3,2,0}, v_{0,3,0}]$.
Check if $[v_{0,0,1}, e_{0,3,2,0}, v_{0,3,0}]$ is a cycle.
Since it is not a cycle, add it to *newpaths*.
Mark vertex $v_{0,3,0}$ as visited.

Create a new path $[v_{0,0,1}, e_{2,3,1,2}, v_{2,3,2}]$.
Check if $[v_{0,0,1}, e_{2,3,1,2}, v_{2,3,2}]$ is a cycle.
Since it is a cycle, unglue the edge in the path and check if the vertex link is still connected.
Since it is still connected, we have found a meridian.
Return $[v_{0,0,1}, e_{2,3,1,2}, v_{2,3,2}]$.

Comparing this result against Figure 4, we can see that the edge from 1 to 2 in $t_{2,3}$ is edge in the vertex link picked out by starting at vertex 1 of $t_{0,0}$. We point out that the result is sensitive to the starting vertex. For example, if we start at vertex $v_{1,2,0}$ then the edge from 3 to 0 in $t_{1,2}$ is picked out instead.

The longitude finding algorithm (Algorithm 3) follows the same process as the meridian algorithm with an additional check when a cycle $C$ is found that $C$ differs from the meridian by at least one edge and the check for connectedness checks that the vertex link is connected after ungluing both $C$ and the meridian. Since the algorithm follows the same process as above, and for brevity, we will not go through each step by hand. The longitude found for the example triangulation is $[v_{0,0,1}, e_{0,3,2,1}, v_{0,3,1}, e_{0,1,2,0}, v_{0,1,0}, e_{2,1,0,2}, v_{2,1,2}, e_{0,0,3,1}, v_{0,0,1}]$.

Note that when checking if the vertex link is still connected after ungluing along a cycle in the meridian algorithm we are check for two possible cases. Either the unglued vertex link is a cylinder without ends and is still connected, or it is torus with a disk removed and is not connected. In the longitude algorithm, ungluing a cycle and the meridian results in the unglued vertex link being a either a disk and is still connected, or a cylinder with a hole cut out of it and is not connected.

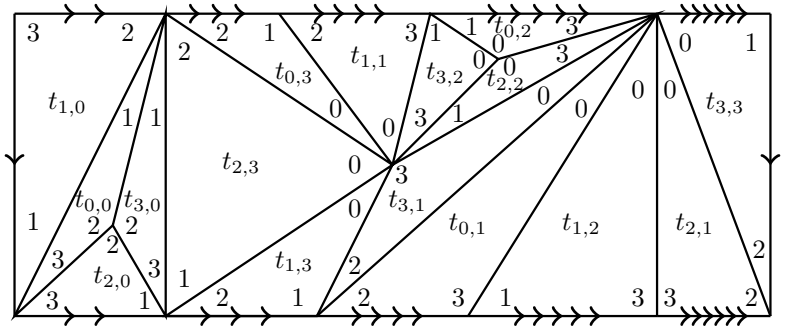


Figure 4: The vertex link for the example triangulation

The meridian algorithm finds the shortest meridian from some starting vertex, but does not find the shortest meridian in the entire vertex link. If the shortest meridian in the vertex link is desired, one can perform the meridian algorithm for each vertex in the vertex link and keeping the shortest one found.

We will see later in the report that the meridian and longitude of the vertex

links in a triangulation provides a set of curves (a complete peripheral system) for which boundaries can be created by cutting off the infinite copies of the vertex links in a spun normal surface, allowing for us to construct a retracted spun normal surface solution with finitely many disks.

**Proposition 1.** *For each longitude $L_v$ computed by Algorithm 3, there is at least one edge in $L_v$ which is not in the meridian of that vertex.*

*Proof.* Proof by contradiction. Suppose Algorithm 3 returns a longitude $L$ in which all edges in $L$ are also in the meridian. Since Algorithm 3 only returns cycles, $L$ is a cycle. Since each cycle found is check to not be the meridian, the set of edges in $L$ is a proper subset of the meridian. Since the meridian is a cycle, any proper subset is not a cycle, so $L$ is not a cycle and we have found a contradiction. □

**Algorithm 2** Meridian based at a point $v$

**Input:** A vertex link and starting vertex $v$ in the 1-skeleton vertex link.

```
Create an array to represent if a vertex has been visited.
Create an empty array of paths paths
Add a path containing v and only v to paths
Mark the v as visited
Initialize meridian to None
while length of paths is greater than zero do
    Create an empty array of paths new paths
    for each path p in paths do
        Retrieve the last vertex u in p
        for each edge e containing u do
            if e is the last edge in path p then
                continue
            end if
            Retrieve the other vertex w in e.
            if w has been visited then
                Call the shortest path from v to w, p'. Trim the overlap between
                    p and p' to create a cycle based at w.
                Unglue the vertex link along C
                if the vertex link is still connected then
                    We have found a meridian
                    if meridian is None or length of meridian > length of C
                        then
                        Assign C to meridian
                    end if
                end if
                Re-glue the vertex link along C.
            else
                Mark w as visited
                Add the path p + e + w to new paths
            end if
        end for
    end for
    if meridian is not None then
        return meridian
    end if
    Assign new paths to paths
end while
```

**Algorithm 3** Longitude based at a point after meridian is computed.

**Input:** A vertex link, a meridian, and a starting vertex $v$ in the meridian.

```
Create an array to represent if a vertex has been visited.
Create an empty array of paths paths
Add a path containing v and only v to paths
Mark the v as visited
Initialize longitude to None
while length of paths is greater than zero do
    Create an empty array of paths new paths
    for each path p in paths do
        Retrieve the last vertex u in p
        for each edge e containing u do
            if e is the last edge in path p then
                continue
            end if
            Retrieve the other vertex w in e.
            if w has been visited then
                We have found a cycle.
                Use the path from v to w and p to create the cycle C.
                if C is not the meridian then
                    Unglue the vertex link along C and the meridian
                    if the vertex link is still connected then
                        We have found a longitude
                        if longitude is None or length of longitude is greater than
                            length of C then
                            Assign C to longitude
                        end if
                    end if
                    Re-glue the vertex link along C and the meridian
                end if
            else
                Mark w as visited
                Add the path p + e + w to new paths
            end if
        end for
    end for
    if longitude is not None then
        return longitude
    end if
    Assign new paths to paths
end while
```

## 3.1 Adjusted standard matching equations

**Definition 3.2.** (Positive and Negative Rotations of a Normal Triangle) Let $\mathcal{T}$ be a triangulation and $t_{i,j}$ be a normal triangle disk of $\mathcal{T}$. Let the four vertices of tetrahedron $i$ be $j, a, b, c$ where, up to relabelling, $a < b < c$. A rotation in $t_{i,j}$ in the direction of the perimeter from $a$ to $b$ to $c$ to $a$ is a *positive rotation*. A rotation in $t_{i,j}$ in the direction of the perimeter from $c$ to $b$ to $a$ to $c$ is a *negative rotation*. Positive rotations will be indicated with a 1, and negative rotation with a $-1$.

**Definition 3.3.** (Oriented Normal Triangle) Let $\mathcal{T}$ be a triangulation and $t_{i,j}$ be a normal triangle disk of $\mathcal{T}$ with a label indicating either a positive or negative rotation. Then $t_{i,j}$ is an *oriented normal triangle*.

**Definition 3.4.** (Consistent Oriented Normal Triangles) Let $\mathcal{T}$ be a triangulation and $\mathcal{L}$ be a vertex link of $\mathcal{T}$. Let $t_{i,j}$ and $t_{k,m}$ be oriented normal triangles in $\mathcal{L}$ which share an edge $e$ in the 1-skeleton of $\mathcal{L}$. If the direction along $e$ induced by the rotation of $t_{i,j}$ opposes the direction along $e$ induced by the rotation of $t_{k,m}$ then $t_{i,j}$ and $t_{k,m}$ are said to be *consistent* with each other.

**Definition 3.5.** (Oriented Vertex Link) Let $\mathcal{T}$ be a triangulation and $\mathcal{L}$ be a vertex link of $\mathcal{T}$ in which each normal triangle in $\mathcal{L}$ is oriented and consistent with each normal triangle it shares an edge with. Then $\mathcal{L}$ is an *oriented vertex link*.

**Definition 3.6.** (Left side of an edge in a peripheral curve) Let $\mathcal{T}$ be a triangulation and $\mathcal{L}$ be an oriented vertex link in $\mathcal{T}$. Let $P$ be a directed peripheral curve of $\mathcal{L}$. Let $e$ be an edge in $P$. Let $t_1$ and $t_2$ be the two normal triangles glued together at $e$. If the direction of $e$ in $P$ opposes the direction induced by the rotation of $t_1$, then $t_1$ is on the *left side* of $e$, otherwise $t_2$ in on the *left side* of $e$.

Note that the above convention for left side is consistent with a positive rotation corresponding to a clockwise rotation. If it were desired to have a positive rotation correspond to counter clockwise rotation then the left side would be the side with the triangle whose rotation follows the direction of the edge.

**Example 3.7.** Orienting the vertex link for the example triangulation. When orienting vertex links we will need to compute orientations such that each pair of triangles sharing an edge are consistent. We will accomplish this by setting one triangle orientation, then performing a breath first search computing consistent orientations for the triangles as we go (See Algorithm 4). We will start by orienting $t_{0,0}$ with a positive rotation. Next we find a consistent orientation for $t_{3,0}$. Since the vertex cut off by both triangles in their tetrahedron is 0 the orientation of $t_{3,0}$ is the orientation of $t_{0,0}$ times the sign of the tetrahedral permutation of the face gluing times $-1$. The permutation between the tetrahedron from the face gluing is $[0, 1, 2, 3] \to [0, 1, 2, 3]$. The permutation is even and the sign of the permutation is 1. The orientation of $t_{3,0}$ is

then $(1)(1)(-1)$. In general, if $sign(t_{i,j})$ denotes the orientation of $t_{i,j}$ and $sign(j) = 1$ if $j \in \{0, 2\}$ and $-1$ otherwise, and $sign(g_{a\to b})$ denotes the sign of the permutation of tetrahedral vertices from a face gluing corresponding to some triangle $t_{k,m}$ sharing and edge with $t_{i,j}$, then the consistent orientation of $t_{k,m}$ is $(-1)sign(g_{a\to b})sign(t_{i,j})sign(j)sign(m)$. The portion of the formula $(-1)sign(g_{a\to b})$ corresponds to the change in orientation between the two tetrahedra. If the triangles change from cutting of 0 or 2 to 1 or 3, or vise versa, then the there is a change of orientation of the triangles within their tetrahedron, requiring a multiplication of $-1$ to keep the orientations consistent, which is included in the formula as $sign(j)sign(m)$. By inspection of Figure 4 we can see a negative rotation of $t_{3,0}$ gives a clockwise rotation consistent with the clockwise rotation of $t_{0,0}$ as desired. Continuing the breath first search we find the following triangle orientations

$$
\begin{aligned}
t_{2,0} &\to -1 \\
t_{1,0} &\to -1 \\
t_{2,3} &\to +1 \\
t_{3,3} &\to +1 \\
t_{1,3} &\to +1 \\
t_{0,3} &\to -1 \\
t_{2,1} &\to +1 \\
t_{3,1} &\to +1 \\
t_{1,1} &\to +1 \\
t_{1,2} &\to -1 \\
t_{0,1} &\to -1 \\
t_{2,2} &\to -1 \\
t_{3,2} &\to -1 \\
t_{0,2} &\to +1
\end{aligned}
$$

**Definition 3.8.** (Adjusted standard matching equation) Let $\mathcal{T}$ be a triangulation with oriented vertex links and let $\Gamma$ be a complete set of peripheral curves. Let $E$ be a standard matching equation of $\mathcal{T}$. Let $\mathcal{L}$ be the vertex link corresponding to $E$ and $e$ be the edge in $\mathcal{L}$ corresponding to $E$. Let $v$ be the vertex in $\mathcal{T}$ corresponding to $\mathcal{L}$. Let $M$ and $L$ be directed meridian and longitude of $\mathcal{L}$ in $\Gamma$ respectively. Let $t_i$ and $q_i$ be the normal triangle and normal quadrilateral on the left side of $e$, and $t_j$ and $q_j$ be the normal triangle and normal quadrilateral on the other side of $e$.

*Case 1.* If $e$ is in both $M$ and $L$, then the *adjusted standard matching equation* corresponding to $E$ is $t_i + q_i + \mu_v + \lambda_v = t_j + q_j$.

*Case 2.* If $e$ is in $M$ and not in $L$, then the *adjusted standard matching equation* corresponding to $E$ is $t_i + q_i + \mu_v = t_j + q_j$.

**Algorithm 4** Orienting a Vertex Link

**Input:** A triangulation $\mathcal{T}$, an orientable vertex link $\mathcal{L}$ of $\mathcal{T}$, a starting triangle $t_{i,j}$ in $\mathcal{L}$, and an orientation for $t_{i,j}$.

```
Assign the input orientation to t_{i,j}
Create an empty first in first out queue q.
Create an empty set visited of visited triangles.
Add t_{i,j} to q
Add t_{i,j} to visited
while visited is not empty do
    Retrieve the next triangle t_{k,m} from q
    for Each triangle t_{n,l} sharing an edge with t_{k,m} in L do
        if t_{n,l} is not in visited then
            Retrieve the permutation p between between tetrahedron k and n
                for the face gluing corresponding to the shared edge from T.
            if (m ∈ {0,2} and l ∈ {0,2}) or (m ∈ {1,3} and l ∈ {1,3}) then
                Assign sign(p)(orientation of t_{k,m})(−1) as the orientation of t_{n,l}
            else
                Assign sign(p)(orientation of t_{k,m}) as the orientation of t_{n,l}
            end if
            Add t_{n,l} to visited
            Add t_{n,l} to q
        end if
    end for
end while
```

*Case 3.* If $e$ is not in $M$ and is in $L$, then the *adjusted standard matching equation* corresponding to $E$ is $t_i + q_i + \lambda_v = t_j + q_j$.

*Case 4.* If $e$ is not in $M$ or $L$, then the *adjusted standard matching equation* corresponding to $E$ is $t_i + q_i = t_j + q_j$.

**Example 3.9.** Finding the adjusted standard matching equations for the example triangulation. We start by orienting the vertex link. With reference to Figure 4 we will orient each triangle to have clockwise rotation. Recall the meridian and longitude of the vertex link are $[v_{0,0,1}, e_{2,3,1,2}, v_{2,3,2}]$ and $[v_{0,0,1}, e_{0,3,2,1}, v_{0,3,1}, e_{0,1,2,0}, v_{0,1,0}, e_{2,1,0,2}, v_{2,1,2}, e_{0,0,3,1}, v_{0,0,1}]$ respectively. We will define the direction of the meridian and longitude to be in the forward direction of the lists. The edge $e_{2,3,1,2}$ in the meridian is not in the longitude. The standard matching equation corresponding to $e_{2,3,1,2}$ is $t_{2,3} + q_{2,03} = t_{3,0} + q_{3,02}$. Looking at Figure 4 we see $t_{3,0}$ is on the left side, so the adjusted normal arc equation is $t_{3,0} + q_{3,02} + \mu = t_{2,3} + q_{2,03}$. The standard matching equations for each of the longitude edges are

$$t_{0,3} + q_{0,03} = t_{1,3} + q_{1,03}$$

$$t_{0,1} + q_{0,02} = t_{3,1} + q_{3,02}$$

$$t_{2,1} + q_{2,01} = t_{3,3} + q_{3,01}$$

$$t_{0,0} + q_{0,02} = t_{1,0} + q_{1,03}$$

respectively. Note that each edge in the longitude belongs to the longitude and not the meridian. By inspection of Figure 4 we determine the left sides of the edges and get the corresponding adjusted standard matching equations as

$$t_{1,3} + q_{1,03} + \lambda = t_{0,3} + q_{0,03}$$

$$t_{3,1} + q_{3,02} + \lambda = t_{0,1} + q_{0,02}$$

$$t_{3,3} + q_{3,01} + \lambda = t_{2,1} + q_{2,01}$$

$$t_{1,0} + q_{1,03} + \lambda = t_{0,0} + q_{0,02}$$

Each standard matching equation not corresponding to a meridian or longitude edge remains unchanged in the adjusted standard matching equations.

# 4 The main algorithm

**Example 4.1.** Computing a retracted spun normal surface solution for the example triangulation. We will compute the retracted spun normal surface solution for the $Q$-matching solution from Example 2.15. Recall

$$S_q = (0,0,0,0,0,1,0,0,0,0,1,0)$$

First we apply the modified Burton's method (Algorithm 1) and find that $S_q$ corresponds to a spun normal surface which is not a normal surface. Next we find a meridian and longitude of the vertex link. Recall we found $[v_{0,0,1}, e_{2,3,1,2}, v_{2,3,2}]$ and $[v_{0,0,1}, e_{0,3,2,1}, v_{0,3,1}, e_{0,1,2,0}, v_{0,1,0}, e_{2,1,0,2}, v_{2,1,2}, e_{0,0,3,1}, v_{0,0,1}]$ as a meridian and longitude respectively. Next we want to determine the triangle coordinates. Unglue the vertex link along the meridian and longitude and perform the modified Burton's method to determine the triangle coordinates. Note that since the vertex link is still connected after ungluing along the meridian and longitude, each triangle coordinate is determined. The complete disk coordinate vector is then

$$S_d = (1,0,0,0,0,0,0,1,0,0,0,0,0,1,1,1,0,0,0,0,0,1,0,0,1,0,1,0)$$

We now can determine $\mu$ and $\lambda$ from adjusted standard matching equations. For $\mu$ we have

$$t_{3,0} + q_{3,02} + \mu = t_{2,3} + q_{2,03}$$
$$1 + 1 + \mu = 0 + 0$$
$$\mu = -2$$

For $\lambda$ we have

$$t_{1,3} + q_{1,03} + \lambda = t_{0,3} + q_{0,03}$$
$$0 + 1 + \lambda = 0 + 0$$
$$\lambda = -1$$

We now have the retracted spun normal surface solution $[S_d, \mu, \lambda, M, L]$.

**Algorithm 5** The main algorithm

**Input:** A triangulation $\mathcal{T}$ with $t$ tetrahedra, a solution $q \in \mathbb{R}^{3t}$ to the $Q$-matching equations for $\mathcal{T}$, a complete peripheral system $\Gamma$ determined by applying Algorithms 2 and 3 to each vertex link. (The starting vertex for each vertex link will be the lexicographically smallest vertex in that link.)

Using Burton's method (Algorithm 1) Attempt to find a solution $S$ to the standard matching equations with quadrilateral coordinates equal those of $q$.

**if** We have found a solution to the standard matching equations **then**
**return** $S$
**end if**

At this point we know $q$ corresponds to a spun normal surface and not a normal surface.

Use a modified version Algorithm 1 which removes the standard matching equations corresponding to the normal arcs in $M_v$ and $L_v$ to compute the disk coordinates, $S_d$.

Let $S_\mu$ and $S_\lambda$ be empty arrays.

**for** each vertex $v$ in $\mathcal{T}$ **do**
Let $\mathcal{L}_v$ be the vertex link of $v$.
Let $M_v$ and $L_v$ be the meridian and longitude of $\mathcal{L}_v$ in $\Gamma$.
Use a modified standard matching equation corresponding to an edge in $L_v$ which in not in $M_v$ (see Proposition 1) to compute $\lambda_v$.
Use a modified standard matching equation corresponding to an edge in $M_v$ to compute $\mu_v$.
Append $S_\mu$ and $S_\lambda$ by $\mu_v$ and $\lambda_v$ respectively.
**end for**

**return** $[S_d, S_\mu, S_\lambda, \Gamma]$

# 5 The algorithms for Euler characteristic, boundary slopes, connectedness, and orientability

Given we have computed a retracted spun normal surface, it is of interest to know some additional properties of the surface and of its boundary components. Here we provide algorithms for computing the Euler characteristic of the surface and boundary slopes of the boundaries.

# 6 The Main Results

**Definition 6.1.** (Connected Component) Let $\mathcal{T}$ be a triangulation and $S$ be a spun normal surface in $\mathcal{T}$. Let $C$ be a collection of disks in $S$ such that for each pair of disk $d_1, d_2$ in $C$ there exists a path of finite length through adjacent disks (disks sharing and edge) in $S$ from $d_1$ to $d_2$ and there does not exist a finite path from any disk $d$ in $C$ to a disk $d'$ in $S$ not in $C$. Then we say $C$ is a *connected component* of $S$.

**Theorem 6.2.** *Given a triangulation $\mathcal{T}$ with a complete peripheral system $\Gamma$ and a solution $S$ to the Q-matching equations (for $\mathcal{T}$), there exists a description of a retracted spun normal surface solution with the same quadrilateral coordinates as $S$.*

*Proof of Theorem 6.2.* A solution $S$ to the $Q$ matching equations corresponds to a normal surface or a spun normal surface. Consider the following cases.

**Case 1.** $S$ carries the quadrilateral coordinates of a (closed) normal surface. Burton's Method [1] (see also Algorithm 1) gives a complete and unique description of normal surface solution, which is also a retracted spun normal surface solution with meridian and longitude coordinates set to zero.

**Case 2.** $S$ does not give not the quadrilateral coordinates of a (closed) normal surface. Then $S$ carries the quadrilateral coordinates of a spun normal surface $P$. We will show the existence of a retracted spun normal surface solution by showing the existence of the coordinates for each connected component of $P$, and adding them together. Let $n$ be the number of vertex classes in $\mathcal{T}$. Denote the meridian and longitude of vertex class $i$ in $\Gamma$ as $M_i$ and $\Lambda_i$ respectively. Let $C$ be a connected component $P$.

**Subcase (a)** The component $C$ is finite. Then $C$ is a (closed) normal surface by definition. The triangle coordinates of $C$ can be determined counting.

**Subcase (b)** The component $C$ is not finite. Then a finite connected subset of $C$ containing all quadrilateral disks can be created by cutting along each $M_i$ and $\Lambda_i$ some number of times (possibly zero). To show this, consider performing a breath first search through the disks making up $C$ starting at a quadrilateral disk in $C$. We know a quadrilateral disk exists since if not, $C$ would be a

**Algorithm 6** Euler Characteristic

**Input:** A triangulation $\mathcal{T}$, and a retracted spun normal surface solution $S'$. Let $\Gamma$ be the complete peripheral system of curves used to compute $S'$.
Sum the triangular and quadrilateral coordinates of $S'$, call the sum $F$.
$F$ now represents the number of faces in $S'$
Initialize $E$ to zero.
Let $sd$ be the disk coordinate vector in $S'$
**for** Each each coordinate $i$ of $sd$ **do**
  **if** $i$ corresponds to a triangle disk type **then**
    Add $3sd[i]$ to $E$
  **else if** $i$ corresponds to a quadrilateral disk type **then**
    Add $4sd[i]$ to $E$
  **end if**
**end for**
**for** Each vertex link $\mathcal{L}$ in $\mathcal{T}$ **do**
  Let $M$ be the meridian for $\mathcal{L}$ in $\Gamma$
  Let $\Lambda$ be the longitude for $\mathcal{L}$ in $\Gamma$
  Let $\mu$ be the coordinate in $S'$ corresponding to $M$
  Let $\lambda$ be the coordinate in $S'$ corresponding to $\Lambda$
  Let $e_M$ be the number of edges in $M$
  Let $e_\Lambda$ be the number of edges in $\Lambda$
  Add $\mu e_M$ to $E$
  Add $\lambda e_\Lambda$ to $E$
**end for**
Divide $E$ by two.
$E$ is now the number of edges in $S'$.
For each edge class $e$ in $\mathcal{T}$ let $d(e)$ denote the number of tetrahedra in $\mathcal{T}$ with an edge in the edge class $e$.
Initialize $V$ to zero
**for** Each coordinate $i$ of $sd$ **do**
  Let $D$ be the disk type corresponding to $i$.
  **for** Each vertex $v$ in $D$ **do**
    Let $e$ be the edge class in $\mathcal{T}$ corresponding to $v$
    Add $\frac{1}{d(e)}$ to $V$
  **end for**
**end for**
**for** Each vertex link $\mathcal{L}$ in $\mathcal{T}$ **do**
  Let $M$ be the meridian for $\mathcal{L}$ in $\Gamma$
  Let $\Lambda$ be the longitude for $\mathcal{L}$ in $\Gamma$
  Let $\mu$ be the coordinate in $S'$ corresponding to $M$
  Let $\lambda$ be the coordinate in $S'$ corresponding to $\Lambda$
  **for** Each vertex $v$ in $M$ **do**
    Let $d'(v)$ be the number of disk on the opposite of $M$ as the boundary which contain $v$
    Add $\frac{d'(v)}{d(v)}$ to $V$
  **end for**
  **for** Each vertex $v$ in $\Lambda$ **do**
    Let $d'(v)$ be the number of disk on the opposite of $\Lambda$ as the boundary which contain $v$
    Add $\frac{d'(v)}{d(v)}$ to $V$
  **end for**
**end for**
Now $V$ is the number of vertices in $S'$
**return** $V - E + F$

**Algorithm 7** Boundary Slope

**Input:** A retracted spun normal surface solution $R'$.
  Let $b$ be an empty list.
  **for** each pair of meridian and longitude coordinates $\mu, \lambda$ in $R'$ **do**
    Append $\frac{\mu}{\lambda}$ to $b$
  **end for**
  **return** $b$

vertex linking surface which is finite, a contradiction. Since there are finitely many quadrilateral disks, and $C$ is a connected component, we will reach each quadrilateral disk in finitely many steps. Call the surface composed of the disks searched $C'$. Note that every disk in $C$ not in $C'$ is a triangular disk and belongs a vertex link. Continue the breath first search outward from $C'$. If a disk edge with an arc type in $\Gamma$ is found do not add the adjacent disk to the head of the breath first search. Since each vertex link is finite in size and there are finitely many vertex links, this search will terminate in finitely many steps. Call the surface created through this extended breath first search $C''$. Since each boundary edge of $C''$ belongs to a peripheral curve in $\Gamma$, each boundary curve in $C''$ is a boundary curve in $\Gamma$. Each boundary curve in $\Gamma$ is a meridian, a longitude, or the union of a meridian and longitude of $\Gamma$. So $C''$ is a connected subset of $C$ containing all quadrilateral disks and can be created by cutting along each $M_i$ and $\Lambda_i$ in $\Gamma$ some number of times. The number of times a meridian $M$ or longitude $\Lambda$ in $\Gamma$ is cut along is the value of the corresponding coordinate for the connected component $C$.

Since the disk and peripheral curve coordinates for each connected component of $P$ exist, the sum of those coordinates also exists. The sum of the coordinates is a retracted spun normal surface solution of $P$, and a retracted spun normal surface solution $P'$, which has the same quadrilateral coordinates as $S$ (and $P$), as desired. □

**Lemma 6.3.** *Let $\mathcal{T}$ be a triangulation $\Gamma$ be a complete system of peripheral curves and $R'$ be a retracted spun normal surface solution, then the coordinates in $R'$ satisfy the adjusted standard matching equations.*

*Proof of Lemma 6.3.* Let $\mathcal{L}$ be a vertex link of $\mathcal{T}$ and let $M$ and $L$ be the meridian and longitude of $\mathcal{L}$ in $\Gamma$. Let $\mu$ and $\lambda$ be the coordinates in $R'$ corresponding to $M$ and $L$ respectively. Let $e$ be an edge in $M$ or $L$. Let $t_i$ and $q_i$ be the disks on the left side of $e$ and $t_j$ and $q_j$ be the disks on the right side of $e$ (as set by Definition 3.6). Let $t_i^{nb}$, $q_i^{nb}$, $t_j^{nb}$ and $q_j^{nb}$ denote the number of disks of type $t_i$, $q_i$, $t_j$, and $q_j$ in $R'$ which do not have a boundary on $e$ respectively. Then $t_i^{nb} + q_i^{nb} = t_j^{nb} + q_j^{nb}$. Let $t_i^b$, $q_i^b$, $t_j^b$ and $q_j^b$ denote the number of disks of type $t_i$, $q_i$, $t_j$, and $q_j$ in $R$ which do have a boundary on $e$ respectively. By definition, $\mu$ and $\lambda$ are the number of boundary components along the left side minus the number on the right side of $M$ and $L$ respectively. Consider the following cases.

**Case 1** Edge $e$ belongs to $M$ and not $L$, then the number of boundary components on the left side minus the number on the right side of $e$ is $\mu$ and $t_i^b + q_i^b + \mu = t_j^b + q_j^b$

**Case 2** Edge $e$ belongs to $L$ and not $M$, then the number of boundary components on the left side minus the number on the right side of $e$ is $\lambda$ and $t_i^b + q_i^b + \lambda = t_j^b + q_j^b$

**Case 3** Edge $e$ belongs to $M$ and $L$, then the number of boundary components on the left side minus the number on the right side of $e$ is $\mu + \lambda$ and $t_i^b + q_i^b + \mu + \lambda = t_j^b + q_j^b$ □

**Theorem 6.4.** *Given a triangulation $\mathcal{T}$ with a complete peripheral system $\Gamma$ and a solution $S$ to the Q-matching equations (for $\mathcal{T}$), Algorithm 5 promotes $S$ to a description of a retracted spun normal surface solution. Moreover, this description is unique.*

*Proof of Theorem 6.4.* A solution $S$ to the $Q$ matching equations corresponds to a normal surface or a spun normal surface. Consider the following cases.

**Case 1.** $S$ carries the quadrilateral coordinates of a (closed) normal surface. This is Case 1 of the previous theorem which was shown to be unique.

**Case 2.** $S$ does not give not the quadrilateral coordinates of a (closed) normal surface. Then $S$ carries the quadrilateral coordinates of a spun normal surface $P$. By Theorem 6.2, we know there exists a retracted spun normal surface solution $P'$ that has the quadrilateral coordinates as $P$. Call one such solution $R'$. Let $v$ be an ideal vertex in $\mathcal{T}$ and $\mathcal{L}$ be the vertex link of $v$. Let $M$ and $L$ be the meridian and longitude of $\mathcal{L}$. Unglue $\mathcal{L}$ along $M$ and $L$ and denote the result $\mathcal{L}'$. Run the modified Burton's method on vertex $v$, call the output $B$. Since $R'$ exists, we know $B$ is `True` with triangle coordinates consistent with the matching equations of $\mathcal{L}'$. Additionally, since the triangle coordinates are unique, the disk coordinates in $B$ must be equal to those in $R'$. There exists an edge $e_L$ in $L$ not in $M$. Let $t_i$ and $t_j$ be the normal triangles sharing edge $e_L$. Up to relabelling, let $t_i$ be on the left side of $e_L$. Then the adjusted standard matching equation corresponding to $e_L$ is $t_i + q_i + \lambda = t_j + q_j$. Since each disk coordinate has been computed, $\lambda$ is determined. Let $e_M$ be an edge in $M$. The adjusted standard matching equation corresponding to $e_M$ is either of the form $t_k + q_k + \mu + \lambda = t_m + q_m$ or $t_k + q_k + \mu = t_m + q_m$. In both cases $\mu$ is determined. By Lemma 6.3, a retracted spun normal surface solution must satisfy the adjusted standard matching equations, and $\mu$ and $\lambda$ are uniquely determined by the disk coordinates. Since the disk coordinates are the same as in $R'$, $\mu$ and $\lambda$ must be the meridian and longitude coordinates corresponding to $\mathcal{L}$ in $R'$.

Since Algorithm 5 computes the triangle coordinates and the peripheral curve

coordinates of $R'$ for any vertex link of $\mathcal{T}$, it will do so for each vertex link of $\mathcal{T}$. So we can compute the retracted spun normal surface solution $R'$. Additionally, since $R'$ was any retracted spun normal surface solution of $\mathcal{T}$ with quadrilateral coordinates $S$ and peripheral curves $\Gamma$, the retracted spun normal surface solution $R'$ is unique with respect to $\mathcal{T}$, $S$, and $\Gamma$. □

**Corollary 6.5.** *Given a triangulation $\mathcal{T}$ with a complete peripheral system $\Gamma$ and a solution $S$ to the Q-matching equations, there exists an algorithm to compute the Euler characteristic of $S$ that is independent of the peripheral system used and can be computed as in Algorithm 6.*

*Proof.* First compute the retracted spun normal surface solution $S'$ from $S$ using Algorithm 5.

**Faces** Since each face in $S'$ is a normal disk, we can compute the number faces by summing the triangle and quadrilateral coordinates of $S'$.

**Edges** Each edge in the surface not on a boundary is an edge of exactly two faces. Each edge in the surface on a boundary is an edge of exactly one face. Let $e = (3t + 4q)/2$ where $t$ is the number of triangle disks and $q$ is the number of quadrilateral disks in $S'$. At this stage, each non boundary edge will have been counted exactly once in $e$ since it belongs to two faces and we have divided by two. Each boundary edge is contributing $\frac{1}{2}$ since it belongs two only one face and we have divided by two. Let $\mathcal{L}$ be a vertex link of $\mathcal{T}$ and $\mu$ and $\lambda$ be the meridian and longitude coordinates for $\mathcal{L}$. Then the number of edges in the boundary component of $\mathcal{L}$ is $\mu e_\mu + \lambda e_\lambda$ where $e_\mu$ and $e_\lambda$ are the number of edges in the meridian and longitude of $\mathcal{L}$ in $\Gamma$ respectively. Then the total number of edges in $S'$ is

$$E = (3t + 4q)/2 + \sum_{\mathcal{L} \in \mathcal{T}} (\mu e_\mu + \lambda e_\lambda)/2.$$

**Vertices** Similar to counting the edges, we will compute the number of vertices if there was no boundary, then adjust for the boundary. For each edge $e$ in $\mathcal{T}$ compute the number of tetrahedra containing $e$, call this $d(e)$. Each vertex $v$ in the surface $S'$ corresponds to an edge $e$ in $\mathcal{T}$, so we will denote $d(v) = d(e)$. Then if $v$ is not a boundary vertex, $d(v)$ will equal the number disks indecent to $v$. Initialize $V$ to zero. For each disk in $S'$ and for each vertex $v$ in the disk add $\frac{1}{d(v)}$ to $V$. Now each non-boundary vertex in $S'$ is contributing exactly one to $V$. Let $v$ be a vertex on the boundary of $S'$. The count $v$ was under counted due to the disks missing on the opposite side of the boundary. Since the boundary is on a vertex link, we can count the number of triangle disks in the vertex link on the opposite side of the boundary, call it $d'(v)$. For each boundary vertex $v$ add $\frac{d'(v)}{d(v)}$ to $V$. Now each boundary vertex in contributing exactly one to $V$, so $V$ is the number of vertices in $S'$.

To show that this result is independent of the peripheral curves in $\Gamma$, consider two different choices for a peripheral curve in a vertex link $\mathcal{L}$ in $\mathcal{T}$. These two peripheral curves bound an annulus. Since the Euler characteristic of an annulus is 0, this does not change the Euler characteristic of the surface. Since swapping out a choice for any one of the peripheral curves does not effect the Euler characteristic of the surface, the Euler characteristic of the surface is independent of the choice of peripheral curves. Algorithm 6 performs the above calculations, and so it computes the Euler characteristic for a solution to the $Q$-matching equations. □

**Definition 6.6.** (Boundary Slope in a Vertex Link) Let $\mathcal{T}$ be a triangulation with a complete peripheral system $\Gamma$ and let $R'$ be a retracted spun normal surface for $\mathcal{T}$. Let $\mathcal{L}$ be a vertex link of $\mathcal{T}$. Let $\mu$ and $\lambda$ be the number of meridians and longitudes of in $R'$ for $\mathcal{L}$. Then the *boundary slope* of the boundary in $\mathcal{L}$ (if it exists) is the pair $(\frac{\mu}{gcd(\mu,\lambda)}, \frac{\lambda}{gcd(\mu,\lambda)})$. The multiplitiy of $(\mu, \lambda)$ in $\mathcal{L}$ is $gcd(\mu, \lambda)$. If $R'$ does not have boundary components in $\mathcal{L}$ we say $\mu = \lambda = 0$ and the multiplicity in $\mathcal{L}$ is 0.

**Corollary 6.7.** *Given a triangulation $\mathcal{T}$ with a complete peripheral system $\Gamma$ and a solution $S$ to the $Q$-matching equations which corresponds to an embedded spun normal surface, there exists an algorithm to compute the boundary slopes of $S'$ a retracted spun normal surface corresponding to $S$. Moreover, for each vertex link $\mathcal{L}$, the number of boundary components in $\mathcal{L}$ is the multiplicity and the boundary slope is an embedded simple closed curve that is part of the boundary of $S'$ implying Algorithm 7 computes this data.*

*Proof.* By Theorem 6.4, we can promote $S$ to a retracted spun normal surface solution $R'$. Let $\mathcal{L}$ be a vertex link of $\mathcal{T}$ and let $M$ and $\Lambda$ be the meridian and longitude of $\mathcal{L}$ in $\Gamma$. Let $\mu$ and $\lambda$ be the coordinates in $R'$ for $M$ and $\Lambda$ respectively. We will assume one of $\mu$ or $\lambda$ is non-zero, as the statement trivially holds otherwise.

First note, if the boundary intersects itself, then the spun normal surface must intersect itself, which is a contradiction since the spun normal surface $S$ is embedded. Furthermore, the boundary is finite in length, and therefore the boundary must be a closed curve. Thus, each boundary component must be a simple closed curve which embeds in the torus by construction. Thus, each boundary component is isotopic to $p$ meridians and $q$ longitudes where $p$ and $q$ are relatively prime.

The boundary slope of the boundary in $\mathcal{L}$ is $(\frac{\mu}{gcd(\mu,\lambda)}, \frac{\lambda}{gcd(\mu,\lambda)})$ and the number of boundary components in $\mathcal{L}$ is $gcd(\mu, \lambda)$. Since the boundary slope and number of boundary components can be computed for the meridian and longitude of any vertex link, they can be computed for each vertex link. Since the boundary components of $R'$ only exist at the meridians and longitudes of $\Gamma$, the boundary slopes for each boundary component in $R'$ and the total number of boundary components can be computed from the data in Algorithm 7. □

**Corollary 6.8.** *Let $\mathcal{T}$ be a triangulation and $R'$ be a retracted spun normal surface solution for $\mathcal{T}$. Algorithm 8 determines the orientability of $R'$.*

*Proof.* Algorithm 8 constructs the retracted spun normal surface, then performs a depth first search through the disks in the surface assigning orientations to the arcs of the disks. If an inconsistency is found, then the algorithm returns false. This is repeated for each connected component of the surface. If all arc orientations are assigned without any inconsistencies then the algorithm returns true. □

**Corollary 6.9.** *Let $\mathcal{T}$ be a triangulation and $R'$ be a retracted spun normal surface solution for $\mathcal{T}$. Algorithm 9 if $R'$ is connected.*

*Proof.* Algorithm 9 constructs the retracted spun normal surface, then performs a depth first search through the disks in the surface marking arcs as visited. If all arc are visited after the first search the algorithm returns true, otherwise it returns false. □

**Corollary 6.10.** *Let $\mathcal{T}$ be a triangulation with a complete peripheral system $\Gamma$ and $R'$ be a retracted spun normal surface solution for $\mathcal{T}$. There exists an algorithm to completely classify the surface.*

*Proof.* The number of Euler characteristic, number of boundary components, orientability, and connectness of $R'$ are determined by Corollaries 6.5, 6.7, 6.8, and 6.9, respectively. Together this data completely determines the surface [3]. □

# References


[1] Benjamin A. Burton. Converting between quadrilateral and standard solution sets in normal surface theory. *Algebr. Geom. Topol.*, 9(4):2121–2174, 2009.

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

[3] George K Francis and Jeffrey R Weeks. Conway's zip proof. *The American mathematical monthly*, 106(5):393–399, 1999.

[4] William Koller. Additional software for investigating spun-normal surfaces. available upon request. Updated: 7/29/2024.

[5] Jeffrey L. Tollefson. Normal surface $Q$-theory. *Pacific J. Math.*, 183(2):359–374, 1998.

**Algorithm 8** Determining orientability of a retracted spun normal surface

**Input:** A triangulation $\mathcal{T}$ and retracted spun normal surface solution $R'$ for $\mathcal{T}$.

```
Let S be the normal disk solution vector in R'
Let diskStacks be an empty list
for each normal disk type d do
    Let m be the value in S corresponding to d.
    Create a stack of m disks and append it to diskStacks
end for
for each normal arc type in T do
    Let b1, tStack1, and qStack1 be the number of boundary edges, triangle
        disks, and quadrilateral disks on one side of the arc type.
    Let b2, tStack2, and qStack2 be the number of boundary edges, triangle
        disks, and quadrilateral disks on the other side of the arc type.
    if neither the vertex cut off by the arc or the vertex missing from the face
        in the gluing in tetrahedron 1 is 0 then
        Reverse the order of qStack1
    end if
    if neither the vertex cut off by the arc or the vertex missing from the face
        in the gluing in tetrahedron 2 is 0 then
        Reverse the order of qStack2
    end if
    Create stack1 as the concatenation of b1, tStack1, and qStack1
    Create stack2 as the concatenation of b2, tStack2, and qStack2
    Note: Since the adjusted standard matching equations are satisfied by R',
        len(stack1) = len(stack2)
    for i in range 0 to len(stack1) do
        Glue the element at position i of stack1 to the element at position i of
            stack2.
        Initialize the orientation of the edge shared by the two elements with
            orientation of 0 (indicating it is not yet oriented).
    end for
end for
for Each stack s in diskStacks do
    for Each disk d in s do
        if d has an arc that is not oriented then
            Set the arc orientation to 1
            Perform a depth first search through the glued disks starting at d
                setting arc orientations along the way and checking for consis-
                tency with arcs already assigned.

            During the depth first search:
            When setting triangle arc orientations, set each arc in the triangle
                with the same orientation.
            When setting quadrilateral arc orientations, set the orientations of
                each pair of arcs sharing a corner in the quadrilateral as the
                negation of each other.

            After the depth first search completes:
            if an inconsistency is found then
                return False
            end if
        end if
    end for
end for
return True
```

**Algorithm 9** Determining connectedness of a retracted spun normal surface

**Input:** A triangulation $\mathcal{T}$ and retracted spun normal surface solution $R'$ for $\mathcal{T}$.

```
Let S be the normal disk solution vector in R'
Let diskStacks be an empty list
for each normal disk type d do
    Let m be the value in S corresponding to d.
    Create a stack of m disks and append it to diskStacks
end for
for each normal arc type in T do
    Let b1, tStack1, and qStack1 be the number of boundary edges, triangle
        disks, and quadrilateral disks on one side of the arc type.
    Let b2, tStack2, and qStack2 be the number of boundary edges, triangle
        disks, and quadrilateral disks on the other side of the arc type.
    if neither the vertex cut off by the arc or the vertex missing from the face
        in the gluing in tetrahedron 1 is 0 then
        Reverse the order of qStack1
    end if
    if neither the vertex cut off by the arc or the vertex missing from the face
        in the gluing in tetrahedron 2 is 0 then
        Reverse the order of qStack2
    end if
    Create stack1 as the concatenation of b1, tStack1, and qStack1
    Create stack2 as the concatenation of b2, tStack2, and qStack2
    Note: Since the adjusted standard matching equations are satisfied by R',
        len(stack1) = len(stack2)
    for i in range 0 to len(stack1) do
        Glue the element at position i of stack1 to the element at position i of
            stack2.
        Initialize the edge shared by the two elements as not visited.
    end for
end for
Create a boolean variable oneDFSComplete and set it to False.
for Each stack s in diskStacks do
    for Each disk d in s do
        if d has an arc that has not been visited then
            if oneDFSComplete then
                return False
            end if
            Mark the arc as visited
            Perform a depth first search through the glued disks starting at d
                marking arcs as visited along the way
            Set oneDFSComplete to True
        end if
    end for
end for
return True
```