Unfolding and Reconstructing Polyhedra

by

Brendan Lucier

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2006

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Brendan Lucier

Abstract

This thesis covers work on two topics: unfolding polyhedra into the plane and reconstructing polyhedra from partial information. For each topic, we describe previous work in the area and present an array of new research and results.

Our work on unfolding is motivated by the problem of characterizing precisely when overlaps will occur when a polyhedron is cut along edges and unfolded. By contrast to previous work, we begin by classifying overlaps according to a notion of locality. This classification enables us to focus upon particular types of overlaps, and use the results to construct examples of polyhedra with interesting unfolding properties.

The research on unfolding is split into convex and non-convex cases. In the non-convex case, we construct a polyhedron for which every edge unfolding has an overlap, with fewer faces than all previously known examples. We also construct a non-convex polyhedron for which every edge unfolding has a particularly trivial type of overlap. In the convex case, we construct a series of example polyhedra for which every unfolding of various types has an overlap. These examples disprove some existing conjectures regarding algorithms to unfold convex polyhedra without overlaps.

The work on reconstruction is centered around analyzing the computational complexity of a number of reconstruction questions. We consider two classes of reconstruction problems. The first problem is as follows: given a collection of edges in space, determine whether they can be rearranged *by translation only* to form a polygon or polyhedron. We consider variants of this problem by introducing restrictions like convexity, orthogonality, and non-degeneracy. All of these problems are NP-complete, though some are proved to be only weakly NP-complete. We then consider a second, more classical problem: given a collection of edges in space, determine whether they can be rearranged *by translation and/or rotation* to form a polygon or polyhedron. This problem is NP-complete for orthogonal polygons, but polynomial algorithms exist for non-orthogonal polygons. For polyhedra, it is shown that if degeneracies are allowed then the problem is NP-hard, but the complexity is still unknown for non-degenerate polyhedra.

Acknowledgments

First and foremost, I would like to thank my supervisors Ian Munro and Anna Lubiw. They consistently directed me toward interesting problems, and this thesis would be in a poor state indeed without their invaluable feedback on my innumerable first chapters. I would also like to thank Anna Lubiw again in her capacity as instructor for the course Algorithms for Polyhedra that began my journey toward this thesis topic.

Contents

List of Tables

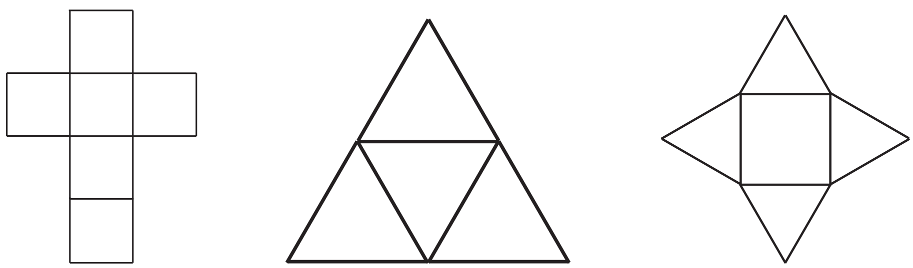
List of Figures

Chapter 1

Introduction

The idea of folding and unfolding polyhedra is as intuitive as it is long-studied. If one were charged with the task of creating a model of a simple three-dimensional object, a natural solution is to cut some pattern from a piece of paper and fold it into the desired shape. For example, to create a cube one might use the well-known cross pattern shown in Figure 1.1(a). In a similar way, one might construct a tetrahedron or a square pyramid from paper cut-outs given in Figures 1.1(b) and 1.1(c).

Paper cut-outs of this form are called nets of polyhedra. These constructions have been studied for hundreds of years, at least as far back as Dürer in 1525 [9]. Since then, many questions have been raised about nets. For example, does every convex polyhedron have a net? In Dürer's time it was assumed that the answer is yes. However, a proof that every convex polyhedron has a net has eluded computational geometers since it was first formally posed by Shephard in 1975 [32]. Since then, researchers have been trying to find algorithms to generate nets for convex polyhedra,



(a) Cube (b) Tetrahedron (c) Square Pyramid

Figure 1.1: Forming polyhedra from paper cut-outs

but thus far none have been successful.

To see what makes this question difficult, we can think of the act of unfolding a polyhedron. Since a net can be folded into a polyhedron, we can consider unfolding a polyhedron to get back the net. We call this operation edge-unfolding, since we cut a polyhedron along its edges to allow it to unfold. Every polyhedron has many edge-unfoldings: simply cut edges until the surface can unfold into the plane. The resulting object is called an edge-unfolding, but not every unfolding is a net. The issue is that two or more faces might overlap when the surface is unfolded. In such a situation, multiple faces occupy the same location, so the planar figure cannot be cut out of a piece of paper and folded back into the original polyhedron.

The problem of finding nets for polyhedra is therefore the same as the problem of avoiding overlaps in unfoldings of polyhedra. Unfortunately, it has proven quite difficult to analyze overlaps in polyhedron unfoldings. It was not even known whether every polyhedron (not necessarily convex) with convex faces had a net until this past decade (they do not; counterexamples were constructed in [4] and [34]).

In the first half of this thesis, we present some results on unfoldings and overlaps. We first consider non-convex polyhedra, and construct some examples of polyhedra with interesting unfoldings. We first present a polyhedron for which every unfolding has an overlap of a particularly trivial form. This polyhedron is then modified to form a polyhedron with no net that has only 9 faces, improving upon the previously best known bound of 13. We then turn to convex polyhedra, and consider certain conjectures of the form "every convex polyhedron has an unfolding of the form x that is a net." We disprove these conjectures by constructing counterexamples that generate overlaps of a special form.

Let us now turn away from unfoldings. Suppose that we are asked once again to construct a model of a polyhedron. Instead of modeling the surface of the polyhedron with paper, perhaps we create our model by constructing a skeleton of its edges. A natural thing to do, then, is to take a set of rigid bars or sticks of the appropriate lengths and attach them at their endpoints to form a three-dimensional shape. Every polyhedron, convex or not, can certainly be recreated in this way. The cube of Figure 1.1(a) could be constructed out of 12 equal-length sticks. The tetrahedron of Figure 1.1(b) and the square pyramid of Figure 1.1(c) can be constructed from 6 and 8 equal-length sticks, respectively.

Suppose now that one were presented with a pile of sticks and asked: "can *some* polyhedron be constructed with these sticks as edges?" This problem has been fully solved in two dimensions: a polygon can be reconstructed precisely when no stick is longer than all the others put together. In three dimensions, however, the answer is much less clear.

In the second half of this thesis, we analyze the computational complexity of this reconstruction problem, along with a number of variants. We also examine the complexity of a related problem, which is to reconstruct a polygon or polyhedron when both the edge lengths and their orientations are given. That is, one is presented with a set of rigid sticks floating in space. One must then form a polyhedron simply by translating the given sticks only, not rotating them. It will turn out that most variants of these problems are hard, but the complexity depends on such factors as the type of polyhedron to construct (e.g. convexity) and types of degeneracies that are allowed (e.g. collinearity of incident edges).

Chapter 2

Preliminaries

Polyhedra have great intuitive appeal. One can visualize a cube, consider the process of folding and unfolding that cube, and imagine constructing the cube from its component edges without too much difficulty. Certainly one does not need any more than informal notions of "polyhedron" and "unfolding" and "edge" to think and reason about such properties of cubes.

Unfortunately, as in any other discipline, our intuitive ideas about unfolding and reconstruction must be grounded by formal definitions. Though tedious, this process is absolutely vital: otherwise we would certainly become muddled in the details of what precisely constitutes a polyhedron or some similar thing. To this end, we shall use this chapter to rigorously define notation for describing and discussing polyhedra.

Our definitions are loosely grouped by category: general definitions regarding polyhedra, the formal definitions involved in unfolding polyhedra, and analytic definitions used for reconstructing polyhedra. It should go without saying that no results in this chapter should be regarded as new. In fact, many of the "results" herein are so well-established in the study of polyhedra that we shall simply state them as properties of the objects in question. If more detail on the derivation of properties of polyhedra is desired, we recommend any one of a number of surveys on the subject [9, 10, 27].

2.1 Polygons and Polyhedra

A *polygon* is a union of straight line segments in the plane that define a planar figure topologically equivalent to a disc. Every line segment intersects another at each of its two endpoints, and two line segments can intersect only at their endpoints. See Figure 2.1. The area bounded by these line segments is the *interior* of the polygon, and the remaining region of the plane is the *exterior*.

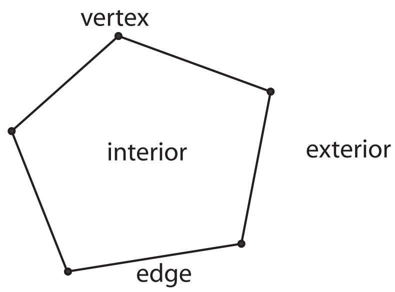


Figure 2.1: A polygon

The line segments themselves are called the *edges* of the polygon. The *boundary* of a polygon is the union of its edges. Any point at which two edges meet is a *vertex* of the polygon. In a slight abuse of notation, we shall sometimes use the term *polygon* to refer to the boundary plus interior of a polygon.

We take $V(P)$ and $E(P)$ to mean the sets of vertices and edges of polygon P, respectively. We say that vertex v and edge e are *incident* if $v \in e$, and two edges are *incident* if they have an incident vertex in common. Two vertices are *adjacent* precisely when they are incident with a common edge.

We now turn to the definition of a polyhedron, which is not as straightforward as that of a polygon. The following definition is taken from Page 209 of Cromwell [9].

Definition 2.1.1 (Polyhedron)**.** A *polyhedron* is the union of a finite set of polygons such that

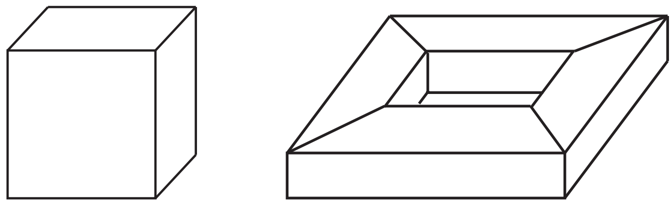
(i) Any pair of polygons meet only at their edges or vertices.

(ii) Each edge of each polygon meets exactly one other polygon along that edge.

(iii) It is possible to travel from the interior of any polygon to the interior of any other, where crossing from one polygon to another occurs via a common edge.

(iv) Let V be any vertex and let $F_1, F_2, \ldots, F_n$ be the n polygons which meet at V. It is possible to travel over the polygons F_i from one to any other without passing through V.

Each polygon making up the polyhedron is a *face* of the polyhedron. When two faces intersect along their edges, the intersection is an *edge* of the polyhedron. Similarly, when multiple faces intersect at their vertices, that intersection point is a *vertex* of the polyhedron. The *interior* of a polyhedron is the open region bounded by its faces. We shall sometimes refer to a polyhedron as



(a) Genus 0 (b) Genus 1

Figure 2.2: Genus of a polyhedron

the *surface* of a polyhedron to further distinguish it from its interior. The *exterior* of a polyhedron is set of points not in the interior or surface of the polyhedron.

We take $V(P)$, $E(P)$ and $F(P)$ to be the sets of vertices, edges, and faces of a polyhedron P. A vertex v and edge or face e are *incident* if and only if $v \in e$. Similarly, an edge e is *incident* with face f if and only if $e \in f$. Two faces or vertices are said to be *adjacent* if they are incident with a common edge, and two edges are said to be *incident* if they are incident with a common vertex.

We now briefly mention the notion of genus. Unlike a polygon, which is always topologically equivalent to a circle, a polyhedron may not be topologically equivalent to the surface of a sphere. The *genus* of a polyhedron can be defined informally as the number of so-called tunnels it has. A cube, for example, has genus zero, while a toroidal polyhedron has genus one. See Figure 2.2. A slightly more formal definition is as follows: a sphere has genus zero, a torus with n holes has genus n, and any polyhedron that can be continuously deformed into a surface of genus g has genus g. In particular, a polyhedron has genus zero precisely when it can be continuously deformed into a sphere.

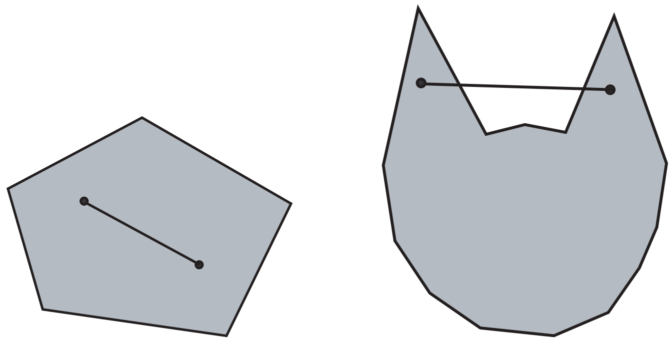
For a more formal definition of genus please see a survey on topology [21, 29]. For the remainder of this paper, we shall only consider polyhedra of genus zero. We shall therefore take "polyhedron" to mean "polyhedron of genus zero."

A key result on polyhedra is Euler's formula relating the numbers of vertices, edges, and faces.

Theorem 2.1.2 (Euler's Formula)**.** *For any polyhedron P of genus zero, we have*

$$|V(P)| - |E(P)| + |F(P)| = 2.$$

We now turn to the notion of convexity. A set of points is *convex* if the line segment between any two points in the set is completely contained in the set. More formally, given set $S \in \mathbf{R}^d$, we say that S is convex if and only if, given any $p, q \in S$, we have that $\{tp + (1-t)q \mid 0 \leq t \leq 1\} \subset S$. See Figure 2.3.



(a) Convex (b) Nonconvex

Figure 2.3: Convex and non-convex sets

We say that a polygon is *convex* if and only if its interior is a convex set. Similarly, a *convex polyhedron* is one whose interior is a convex set. Equivalently, a polygon or polyhedron P is convex if the line segment joining two points upon the surface of P does not intersect the exterior of P.

Given any set of points $S \subset \mathbf{R}^d$, we define the *convex hull* of S to be

$$CH(S) = \{tp + (1-t)q \mid p, q \in S, 0 \leq t \leq 1\}.$$

Then the convex hull of S is the smallest convex set containing S.

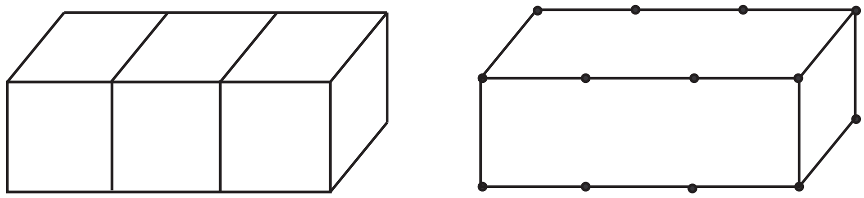
A set of points is *starlike* if it contains a fixed point p such that the line segment between p and any other point in the set is completely contained in the set. More formally, set $S \in \mathbf{R}^d$ is starlike if and only if there is a $p \in S$ such that, given any $q \in S$, we have that $\{tp+(1-t)q \mid 0 \leq t \leq 1\} \in S$. As with convexity, a polygon (polyhedron) is starlike if and only if its interior is a starlike set.

Let us now turn to particular types of polyhedra. A polyhedron is said to be *convex-faced* if its faces are all convex as polygons. Further, a polyhedron is *simplicial* if its faces are all triangles.

A polygon is *orthogonal* if its edges are all axis-aligned. A polyhedron is *orthogonal* if each of its faces is perpendicular to the x-, y-, or z-axis. Equivalently, an orthogonal polyhedron is one in which each face is parallel to one of the xy-, yz-, or xz-plane.

We say that a polygon is *degenerate* if it contains a pair of adjacent edges that are collinear. A polyhedron is *degenerate* if it contains a degenerate face or if it has a pair of adjacent faces that are coplanar. See Figure 2.4.

We now define special angles of polygons and polyhedra. If v is a vertex of a polygon, we say that the *interior angle* at v is the angle between its incident edges that faces toward the



(a) Coplanar Faces (b) Collinear Edges

Figure 2.4: Examples of degenerate polyhedra

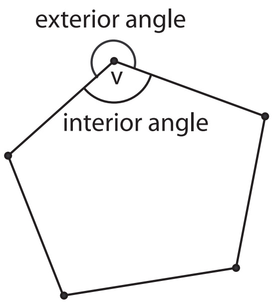
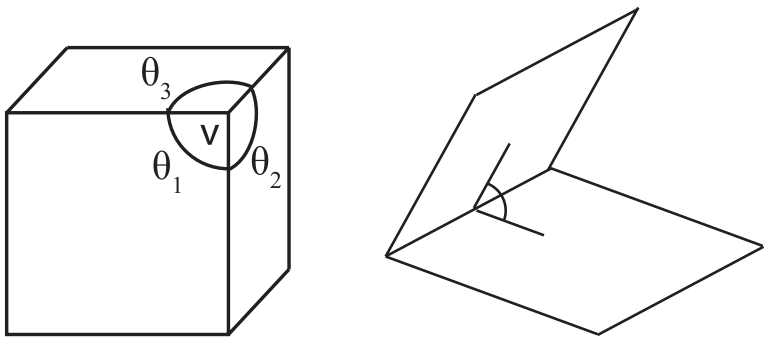


Figure 2.5: Polygon angles



(a) Face Angles (b) Dihedral Angle

Figure 2.6: Angles in a polyhedron

interior of the polygon. Similarly, the *exterior angle* at v is the angle facing toward the exterior of the polygon. Note that the interior and exterior angles at a vertex v always sum to 2π. See Figure 2.5.

Now suppose v is a vertex of a polyhedron. The interior angle at v in an incident face is called a *face angle*. The *total face angle* at v is the sum of all face angles at v. The *curvature* at v is 2π minus the total face angle at v. Finally, a *dihedral angle* is the interior angle between two adjacent faces. See Figure 2.6.

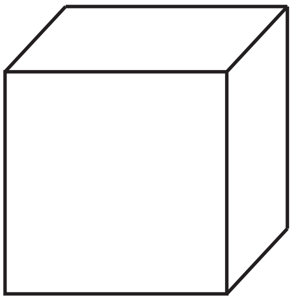
Another key result is Descartes' characterization of the total curvature of a polyhedron of genus 0. Namely, the sum of the curvatures at all vertices of a polyhedron is $2\pi(2-2G)$, where G is the genus of the polyhedron. Thus the sum of curvatures for a polyhedron of genus zero is 4π.

2.2 Unfoldings

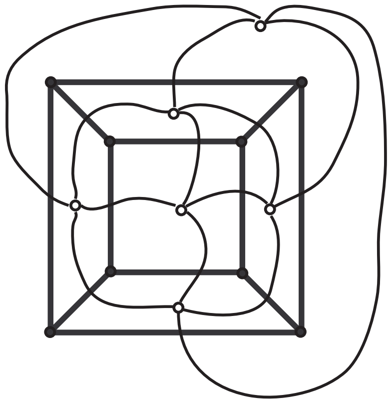
2.2.1 Formal Definition

Informally, one unfolds a polyhedron by first cutting its surface and then flattening it into the plane. In this section we define this concept more rigorously, making use of the underlying graph structure of a polyhedron.

Given a polyhedron P, the *graph of* P is the graph $(V(P), E(P))$. That is, it is the graph of vertices and edges of P. A graph contains a *cycle* if it contains a sequence of distinct edges $e_0, e_1, \ldots, e_{n-1}$ such that e_i is incident with $e_{(i+1 \mod n)}$ for all i. A graph is *connected* if for any vertices p and q there is a sequence of edge $e_1, e_2, \ldots, e_n$ such that p is incident with e_1, q is incident with e_n, and e_i is incident with e_{i+1} for all i. A graph that is connected and contains no



(a) A cube



(b) The graph and adjacency graph of the cube

Figure 2.7: Graphs of a cube

cycles is a *tree*. A *spanning tree* of a graph G is a subgraph of G that contains all vertices of G and is a tree.

Given any edge $e \in E(P)$, we define the *dual* of e to be an edge e^* that connects the two faces incident with e. The *dual graph* of P is the graph $(F(P), \{e^* \mid e \in E(P)\})$. The vertices of this graph are the faces of P, and two vertices are connected by an edge if and only if the corresponding faces are adjacent. See Figure 2.7.

A *cut tree* T of polyhedron P is a spanning tree of the graph of P. The *adjacency tree* of P corresponding to T is the complement of T in the dual graph of P. In other words, if A is the adjacency tree corresponding to T, then two faces are adjacent in A if and only if their common edge is *not* in T. We also say that T is the cut tree corresponding to A. See Figure 2.8 for an example.

Definition 2.2.1 (Unfolding)**.** Let P be a polyhedron with adjacency tree A. For each $f \in F(P)$, define an isometric function $\phi_f : f \to \mathbf{R}^2$. That is, ϕ_f is a function mapping each face into the plane that preserves distances. We further require that for every edge $e \in E(p)$ such that $e^* = (f_1, f_2)$ is in A, $\phi_{f_1}(e) = \phi_{f_2}(e)$. Then the *unfolding* of P with respect to A is the mapping $\phi : P \to \mathbf{R}^2$ defined by $\phi(p) = \cup_{f : p \in f} \phi_f(p)$. In other words, $\phi(P)$ is the union of all the images $\phi_f(f)$. Note that edges and vertices can have multiple images under ϕ.

We will also sometimes refer to the image $\phi(P)$ as an unfolding of P, for convenience. These unfoldings are sometimes called *edge-unfoldings*, to distinguish them from the more general *general unfoldings*. See Section 2.2.5 for more information on this distinction.

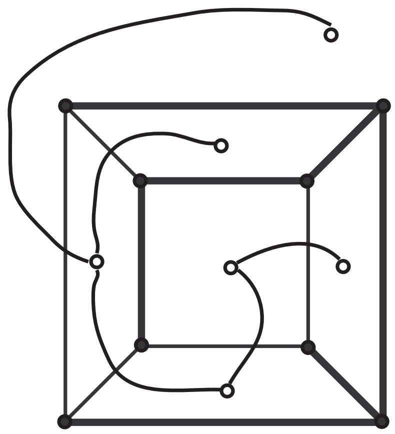
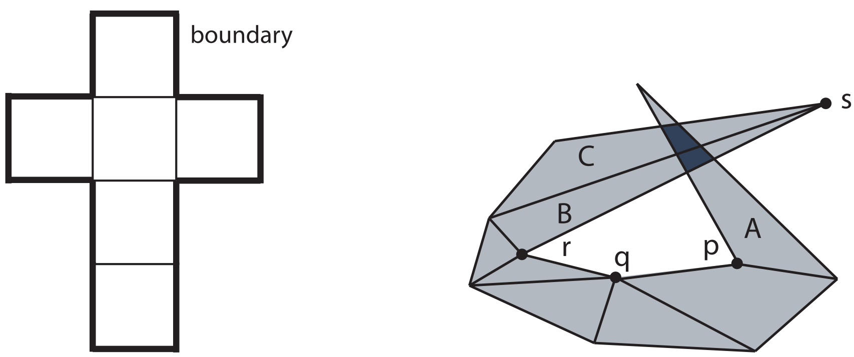


Figure 2.8: A cut tree (edges in bold) and corresponding adjacency tree for a cube



(a) A simple unfolding of the cube

(b) Overlaps in an unfolding

Figure 2.9: Simple and overlapping unfoldings

Now suppose ϕ is an unfolding of P. If there are two faces f_1, f_2 of P such that $\phi(f_1) \cap \phi(f_2)$ contains interior points then we say that f_1 and f_2 *overlap* in the unfolding. Note that we only consider interior points, so faces can intersect at edges or vertices without creating overlaps. An unfolding that contains no overlaps is a *simple unfolding*. A simple unfolding is also called a *net*. A polyhedron that has no simple unfolding is called *ununfoldable*, since every unfolding contains an overlap.

The mapping ϕ maps each face of P to a polygon in the plane. Consider the behaviour of ϕ upon the edges and vertices of P. Each edge e whose dual e^* is in A will be mapped to a single edge by ϕ, since both faces incident with e will be connected in $\phi(P)$. On the other hand, an edge e whose dual is not in A will have two images, corresponding to the images of its two incident faces. These edges with two images are precisely the edges in C, the cut tree corresponding to A. The union of all such images of edges is called the *boundary of the unfolding*. See Figure 2.9(a).

Every image of every vertex of P will lie upon the boundary of the unfolding, since the cut tree spans all vertices. Further, the number of images of vertex v is precisely the degree of v in the cut tree C. This is because every image of v is the intersection of the images of two cut edges incident with v, and each cut edge has two images.

2.2.2 Zero-curvature vertices

We now comment briefly on vertices of curvature 0. Suppose a polyhedron contains vertex v of curvature 0. Technically speaking, if we want to unfold the polyhedron into the plane, we do not need to cut an edge incident with v. However, if we take such an unfolding and simply add cuts so that we obtain a spanning tree of the vertices (and, in particular, have a cut incident with v), we obtain a new unfolding. In this new unfolding all faces are in the same locations as before, but certain edges between faces are now thought of as two edges that happen to occur in the same position, and the corresponding faces are no longer adjacent. In effect, we have the same unfolding whether or not we make a cut into vertex v; we are simply interpreting the unfolding slightly differently either way. We can therefore assume without loss of generality that all unfoldings are generated by spanning trees of cuts, even if vertices of zero curvature are present.

2.2.3 Unfolding Angles

Let T be a cut tree and consider the faces incident with a vertex v. We wish to partition these faces into groups, each group corresponding to the faces whose images are incident with a given image of v. More precisely, let the images of v in the unfolding be $v_1, \ldots, v_k$. Then all of the faces incident with a given v_i in the unfolding form an *unfolding group* or *component* of v. Note that no face can belong to more than one unfolding group, since such a face would have to be

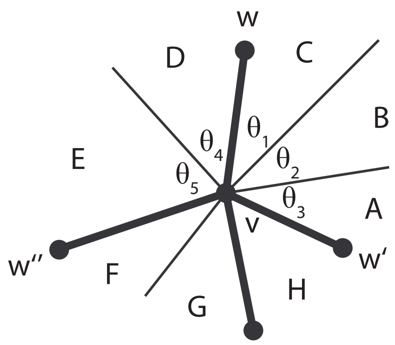


Figure 2.10: Unfolding angles at vertex v. The unfolding groups at v are ABC, DE, FG, and H. The unfolding angle bounded by (v, w) and (v, w') is $\theta_1 + \theta_2 + \theta_3$. The unfolding angle bounded by (v, w) and (v, w'') is $\theta_4 + \theta_5$. Both are considered to be unfolding angles bounded by (v, w).

incident with v at two places along its boundary, an absurdity.

There is a relationship between unfolding groups and the cut tree T. Two faces f_1 and f_2 are in the same unfolding group precisely when one can traverse the faces incident with v from f_1 to f_2 (either clockwise or counterclockwise) without crossing an edge in T. Informally speaking, the edges in T split the faces incident with v into the unfolding groups. This implies that the number of unfolding groups at v is precisely the degree of v in T. See Figure 2.10.

The sum of the face angles at v over all faces in an unfolding group is called an *unfolding angle* at v. The important thing to note is that the unfolding angles at v are precisely the interior angles of $v_1, \ldots, v_k$ in the unfolding. This is because the interior angle at some v_i is simply the sum of all face angles at v_i, which is the same as the face angles at v for all faces in the unfolding group corresponding to v_i.

Finally, if e is a cut edge incident with v, we say that an unfolding group is *bounded by* e if a face in the group is incident with e. The unfolding angle of such a group is referred to as an *unfolding angle bounded by* e. See Figure 2.10.

2.2.4 Defining Unfoldings Algorithmically

We have presented a mathematical definition of polyhedron unfoldings. It is also possible to define unfoldings algorithmically, so that it becomes clear that there is a unique unfolding (up to translation/rotation) for a given polyhedron and cut tree. To this end, we describe a process by which an unfolding is created.

Take an adjacency tree A of polyhedron P and root it at some arbitrary face f_0. Place face f_0 in the plane, such that the side facing the interior of P faces upward. Now any child f_c of f_0 is connected to f_0 via an edge $e^* \in A$. Then e^* is the dual of an edge $e \in E(P)$ that is incident with both f_0 and f_c. We now place f_c in the plane (again, side facing the interior of P facing upward) by attaching it to f_0 at the common edge e.

We repeat this process for all children of f_0. We then continue by repeating for all children of the faces that have been placed. We are effectively performing a traversal of A, at the end of which all faces are placed in the plane. The resulting complex of polygons is the unfolding of P corresponding to A.

2.2.5 General Unfoldings

The terms "Unfolding" and "Cut tree" are motivated by the process by which one would convert the surface of a polyhedron into an unfolding. Suppose we have a polyhedron P and a cut tree T. Let C be the union of the edges of T. We now remove C from P; effectively "cutting" along the cut tree. The resulting surface $P - C$ can now be unfolded (i.e. isometrically mapped) into the plane, and this process is equivalent to the unfolding ϕ corresponding to cut tree T.

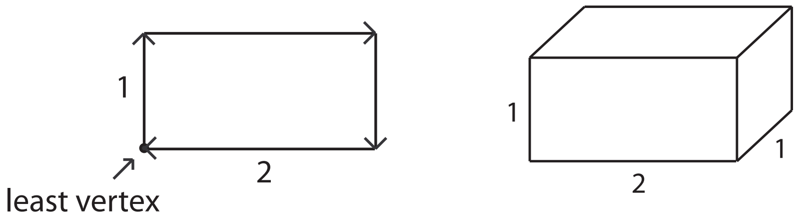
Motivated by the description of an unfolding as a polyhedron with a set of edges removed, we now generalize our notion of an unfolding. Let C be the union of some connected set of curves on the surface of polyhedron P, where any two such curves intersect only at their endpoints. Note here that a curve in C may contain points on the interior of faces. Then as long as C contains every vertex of P, the surface $P - C$ can be isometrically mapped into the plane. The resulting planar surface is called a *general unfolding* of P. Less formally, a general unfolding is one in which cuts are allowed to occur across faces. We will sometimes call unfoldings *edge-unfoldings* to distinguish them from general unfoldings.

2.3 Reconstruction

We now give some rather technical definitions to assist in analytic discussions of polygons and polyhedra.

2.3.1 Vectors and Lengths

Given a vector v, we denote by $|v|$ the length of v. The coordinates of v are denoted $v.x$, $v.y$, and (if applicable) $v.z$. Similarly, the coordinates of a point p are denoted $p.x$, $p.y$, and (if applicable) $p.z$. We say that vector v is *positive facing* (or just *positive*) if its first non-zero coordinate is



(a) Rectangle (b) Rectangular Prism

Figure 2.11: Examples of edge vectors

positive. A vector that is neither positive nor the zero vector is *negative*. The *slope* of a two-dimensional vector v is given by $\frac{v.y}{v.x}$ if $v.x \neq 0$. When $v.x = 0$, we will take the slope of v to be $+\infty$ if $v.y > 0$, $-\infty$ if $v.y < 0$, or undefined if $v.y = 0$. In particular, when positive vectors are sorted in decreasing order by slope, those with $v.x = 0$ will always occur first.

Suppose edge e of a polygon or polyhedron has endpoints p_1 and p_2. Then we denote by $v(e)$ the vector $p_2 - p_1$ or $p_1 - p_2$, whichever is positive. Then, given a polygon or polyhedron P, we can consider the multiset of edge vectors $Vec(P) = \{v(e) \mid e \in E(P)\}$. We can also consider the multiset of edge lengths $Len(P) = \{|v(e)| \mid e \in E(P)\}$. Consider the examples in Figure 2.11. If P_{2d} is the rectangle in Figure 2.11(a), then we have $Vec(P_{2d}) = \{(0,1),(0,1),(2,0),(2,0)\}$ and $Len(P_{2d}) = \{1,1,2,2\}$. Now take P_{3d} to be the rectangular prism in Figure 2.11(b). Then

$$\begin{aligned} Vec(P_{3d}) = \{&(0,1,0),(0,1,0),(0,1,0),(0,1,0),(0,0,1),(0,0,1),\\ &(0,0,1),(0,0,1),(2,0,0),(2,0,0),(2,0,0),(2,0,0)\} \end{aligned}$$

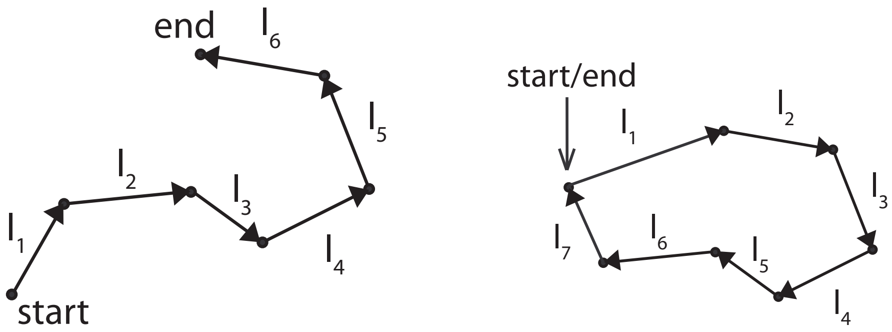
and

$$Len(P_{3d}) = \{1,1,1,1,1,1,1,1,2,2,2,2\}$$

.

2.3.2 Chains and Polygons

Suppose we have a sequence of line segments $l_1, l_2, \ldots, l_n$ such that each l_i has one endpoint in common with l_{i-1} and its other endpoint in common with l_{i+1} for $1 < i < n$. Then this sequence of line segments forms a *chain*. We say that the endpoint that l_{i-1} shares with l_i is the *end* of l_{i-1} and *start* of l_i. In the case of l_1, the endpoint that is not the end is taken to be the start, and vice-versa for l_n. The *start of the chain* is the start of l_1, and the *end of the chain* is the



(a) Open Chain (b) Closed Chain

Figure 2.12: Open and closed chains

end of l_n. If the start and end of the chain coincide, we say that the chain is *closed*, otherwise it is *open*. For each l_i, the *direction* of l_i is the vector resulting from subtracting the start of l_i from the end of l_i. The *orientation* of l_i is either the direction of l_i or its negative, whichever is positive.

Note the relationship between closed chains and polygons. A closed chain corresponds to a polygon precisely when the only intersections between line segments occur between l_i and l_{i+1} at their common endpoint for some i, or between l_1 and l_n at the start/end point of the chain.

Given a polygon P in the plane, the *least vertex* of P is the vertex p with minimum $p.x$, then minimum $p.y$. The *standard traversal* of the edges and vertices of P is the traversal beginning at p and proceeding in clockwise order. Note that this traversal implies a closed chain with start and end point p. This chain imposes an order and direction upon the edges of P; we call these the *standard order* and *standard direction* of (the edges of) P. For example, the rectangle P_{2d} in Figure 2.11(a) has arrows to show the standard direction of its edges. Recall that the edge vectors of P_{2d} are $Vec(P_{2d}) = \{(0,1),(0,1),(2,0),(2,0)\}$, but the *sequence of edge vectors in standard order and direction* is $((0,1),(2,0),(0,-1),(-2,0))$.

The following technical lemma will be of great use.

Lemma 2.3.1. *Suppose a closed chain of line segments corresponds to a polygon, where the start point of the chain is the least vertex of the polygon. Then this chain corresponds to a* convex *polygon if and only if its direction vectors are ordered as*

1. *positive vectors in decreasing order by slope, then*

2. *negative vectors in decreasing order by slope.*

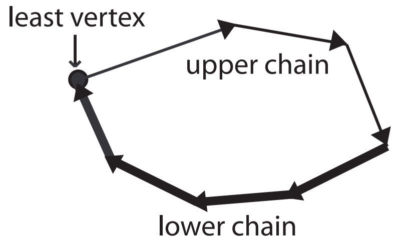


Figure 2.13: A convex polygon as a pair of chains

Proof. This result follows from the fact that a polygon is convex if and only if it has no interior angle greater than π. The details are omitted. □

In light of this lemma we shall consider a convex polygon as having two halves. The partial chain containing all positive vectors (under standard direction) is the *upper chain*, and the remaining partial chain containing all of the negative vectors in standard direction is the *lower chain*. See Figure 2.13.

2.3.3 Problem Complexity

We now give a brief review of some terminology regarding complexity classes of problems. We direct the reader to Chapter 34 of [8] for a more complete introduction to complexity. A *decision problem* is a question, asked over some set of possible input values, to which the answer is "yes" or "no" for each possible input. For example, the question "is this polygon convex?" is a decision problem where the set of possible input values is precisely the set of polygons. We say that a machine *solves* a decision problem if it accepts precisely those input strings which encode instances of the problem for which the correct answer is "yes." Note that the manner in which input should be encoded is specified as part of a decision problem. An *oracle* for a decision problem is a machine which instantly gives the correct answer to any given instance of that problem.

We say that an algorithm is *polynomial* or *polytime* if it runs in a number of steps that is polynomial with respect to its input size. The set of decision problems that can be solved in polynomial time by a non-deterministic Turing machine is denoted NP. Given two problems P_1 and P_2, a *polytime reduction* from P_1 to P_2 is a polynomial (deterministic) algorithm that solves P_1, making use of an oracle for P_2. Polytime reductions are also called *Cook reductions* or *Turing reductions*. A polytime reduction in which the oracle is used only once at the end of the algorithm is called a *Karp reduction*.

A decision problem to which there is a polytime reduction from any other problem in NP is called *NP-hard*. A decision problem in NP that is NP-hard is called *NP-complete*. To show that a problem is NP-hard, one typically performs a polytime reduction from another problem known to be NP-hard. For more information on NP-hardness and reductions, see [17].

When dealing with problems that involve numerical value, the choice of an encoding for the input is very important. The complexity of a problem can often change depending on whether numbers are encoded in binary or unary, since the size of a unary encoding may be exponential in the size of a corresponding binary encoding. We say that a problem is *strongly NP-hard* if it is NP-hard even when all numerical values are represented in unary. If a problem is NP-hard but not strongly NP-hard, we say that it is *weakly NP-hard*. To show that an NP-hard problem is weakly NP-hard, one gives an algorithm to solve the problem that is polynomial with respect to the numerical values in the problem (not the input size). Such an algorithm is called a *pseudo-polynomial* or *pseudo-polytime* algorithm.

Chapter 3

Background

Before we move on to our main results, it is worthwhile to discuss the work that has come before. There has been a recent surge of interest in the study of polyhedra unfoldings. This thesis is largely motivated by results discovered and questions posed in only the last few years.

In staying with the common theme of this thesis, this chapter will be split into two parts. These correspond to the two fundamental problems that we address: the unfolding of a polyhedron into the plane and the complexity of determining whether a polyhedron can be reconstructed from certain partial information. In general these areas are largely disjoint but, as we shall see, work on folding polyhedra has provided a link between the two.

3.1 Unfolding Polyhedra

3.1.1 Polyhedra and Art

The first person known to study unfoldings was Albrecht Dürer in the early 16$^{\text{th}}$ century. He published a sequence of four books entitled "Instruction in the Art of Measurement with Compasses and Rule of Lines, Planes and Solid Bodies" (translated from German). The purpose of this work was to instruct artists in perspective and the theoretical aspects of illustration. In the fourth book, Dürer analyzes solid geometry; he considers various well-known polyhedra and how they should be correctly illustrated. It is here that Dürer introduced the notion of using a paper cut-out of a polyhedron to convey information about it. Though he did not use the term at the time, Dürer had created the concept of a net.

Dürer implicitly assumed that every convex polyhedron had a net. Of course, we now know that this assumption was unfounded: it is still open whether or not this is true, even after 500 years.

Edge-Unfoldings of Convex Polyhedra

In 1975 the question of whether or not every convex polyhedron has a net was posed formally by Shephard [32].

Conjecture 3.1.1 (Shephard's Conjecture)**.** *Every convex polyhedron can be cut along some of its edges and unfolded into the plane without overlap.*

There have been many attempts to resolve Shephard's Conjecture since it was posed in 1975. Much of this work has manifested in numerous proposed algorithms meant to create a simple unfolding for any given convex polyhedron. Unfortunately, a counterexample has been found for every algorithm yet proposed.

Fukuda put forth a number of conjectures on simple unfoldings of convex polyhedra [18]. He first proposed that cutting along the minimal edge-length spanning tree for a convex polyhedron would always yield a net, but a counterexample was found by Günter Rote [18]. Fukuda also suggested that the shortest-path tree would form a cut tree for a convex polyhedron, but this was disproved in an experiment by Schlickenrieder [30]. Finally, Schlickenrieder conjectured that the Steepest-Edge Cut algorithm would always be successful for finding a net. However, a main result of this thesis is the construction of a counterexample to Schlickenrieder's conjecture. For a more detailed discussion of the Steepest-Edge Cut algorithm and our counterexample, please see Section 5.3.2.

In related work, Fukuda and Namiki performed experiments on random unfoldings of large convex polyhedra. These experiments were performed using the UnfoldPolytope mathematica package, which performs unfolding operations using a number of heuristics to form cut trees [24, 25]. Based on the results of these experiments, it was conjectured that the probability that a random unfolding of a random n-vertex convex polyhedron has an overlap approaches 1 as $n \to \infty$. In other words, almost all unfoldings of large convex polyhedra contain overlaps. This does not discount the possibility of at least one non-overlapping unfolding of every convex polyhedron. However, the high density of overlaps in large polyhedra implies that if Shephard's Conjecture is true then simple unfoldings for convex polyhedra are exceptions to the rule that most unfoldings contain overlaps.

3.1.2 Other Types of Simple Unfoldings

While there has been little progress in resolving Shephard's Conjecture, there has been some work in considering other types of unfoldings and showing that they can be formed without overlap.

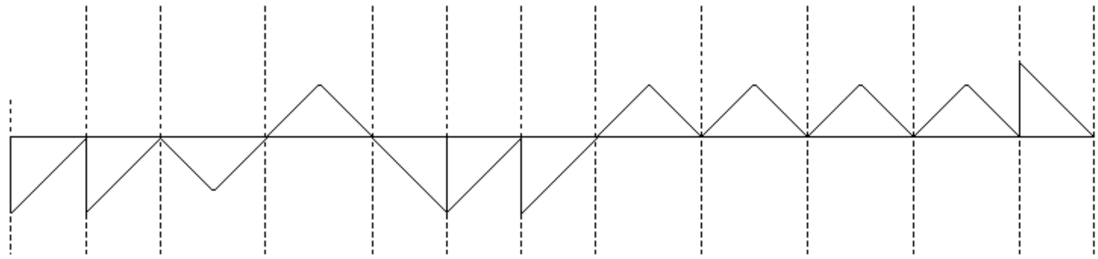


Figure 3.1: A vertex-unfolding of a simplicial cube. Image due to [14].

General Unfoldings of Convex Polyhedra

The first significant progress was the demonstration that every convex polyhedron has a general unfolding that does not overlap. Recall that a general unfolding is an unfolding that allows cuts across faces, not just along edges. In fact, two types of non-overlapping general unfoldings are known: the source unfolding and the star unfolding.

In the star unfolding, a point x is chosen on the surface of the polyhedron such that there is a unique shortest path from x to each vertex. The polyhedron is then cut along each of those shortest paths. It was shown in [3] that the star unfolding does not overlap. The star unfolding also has some applications to efficient computation of shortest-paths on the surface of a polyhedron [1]. The source unfolding is similar, but instead of cutting along the path from x to each vertex, one cuts along the locus of all points for which there are two or more distinct shortest paths from x. The source unfolding also does not overlap; the proof of this is also much simpler than that for the star unfolding [31].

Vertex-Unfoldings

Another recent approach has been the exploration of *vertex-unfoldings* [14]. In a vertex-unfolding, cuts are made along edges (just as in edge-unfoldings). However, faces are not required to meet at common edges; two faces are allowed to meet at a common vertex. The resulting unfolded figure is connected, but the interior of the figure may not be. It has been shown that every simplicial manifold (including polyhedra, surfaces with boundary, and surfaces in higher dimensions) has a non-overlapping vertex-unfolding. The idea is to form a sequence of cuts that splits the surface into individual faces connected at vertices, then “string” these faces in an approximate line. See Figure 3.1 for a vertex-unfolding of a triangulated cube. It is still open whether all manifolds with non-triangular faces, in particular non-simplicial convex polyhedra, have non-overlapping vertex-unfoldings.

Multi-piece Unfoldings

In an edge-unfolding, it is required that the unfolded planar figure be connected. However, we can imagine allowing our unfolding to split the faces of a polyhedron into a number of disconnected pieces. Another avenue of attack upon Shephard's Conjecture is determining bounds upon the number p of pieces necessary to obtain a non-overlapping unfolding. Taking $p = 1$ corresponds to an edge-unfolding. On the other extreme, taking $p = n$ (where n is the number of faces in the polyhedron) corresponds to the trivial case where each face lies in a separate component.

If we consider p to be a function of n, we obtain a manner of discussing lower bounds on the number of pieces necessary to form a simple unfolding with respect to n. Michael Spriggs has obtained a bound of $p \leq \frac{1}{2}n$ [33]. Proving a sublinear bound would represent significant progress in this area.

3.1.3 Unfolding Classes of Orthogonal Polyhedra

Another avenue of research is to consider particular classes of polyhedra. Biedl et al. demonstrated that certain classes of orthogonal polyhedra have simple general unfoldings [6]. In particular, they showed that orthostacks and orthotubes can be cut across faces and unfolded without overlap.

Another type of unfolding, grid unfoldings, applies only to orthogonal polyhedra. In this unfolding type one is allowed to cut across faces, but only along axis-aligned grid lines. The maximum number of grid lines given per face is the degree of the grid unfolding. It was shown in [12] that another class of orthogonal polyhedra, dubbed Manhattan Towers, are unfoldable with grid unfoldings of degree 5. It was also shown in [16] that orthostacks (a class of orthogonal polyhedra) can be vertex-unfolded using only grid cuts.

Further, in [11], it was shown that well-separated orthotrees can be edge-unfolded without overlap. It was suggested that a similar method might be applied to demonstrate that all orthotrees can be edge-unfolded, but this problem is still open.

3.1.4 Ununfoldable Polyhedra

The question of whether every convex polyhedra has a net has proved quite vexing, but what of non-convex polyhedra? There are very simple examples of general polyhedra with no nets. In the star-shaped prism of Figure 3.2, for example, no face adjacent to the top face can be attached without causing an overlap.

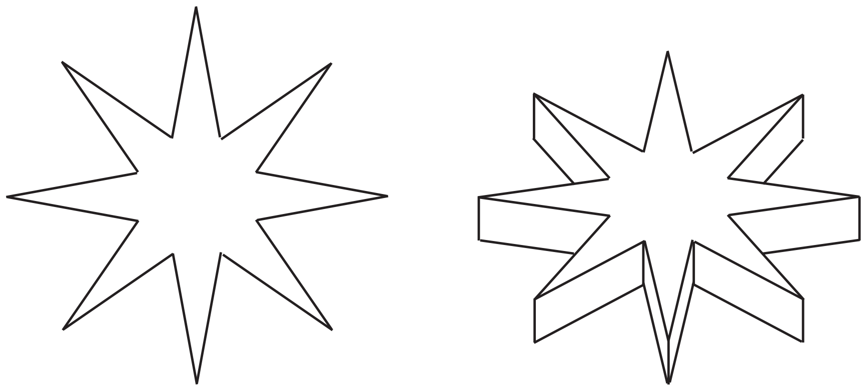


Figure 3.2: An ununfoldable polyhedron with non-convex faces

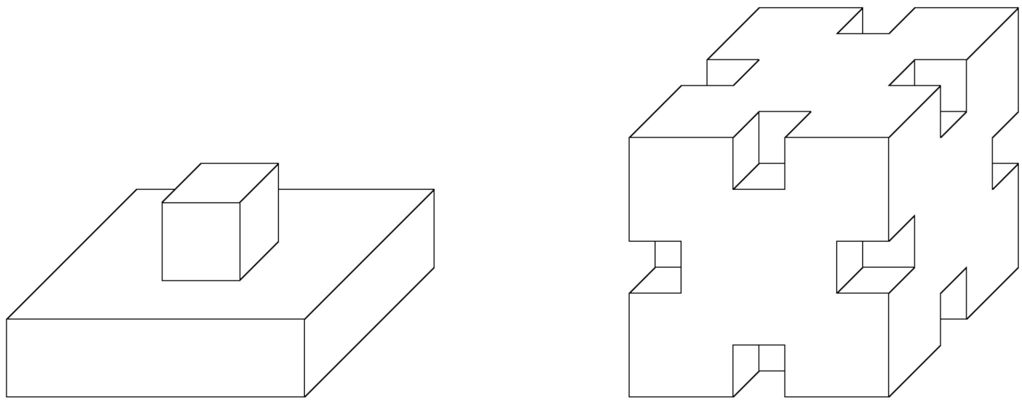


Figure 3.3: Ununfoldable orthogonal polyhedra due to [6].

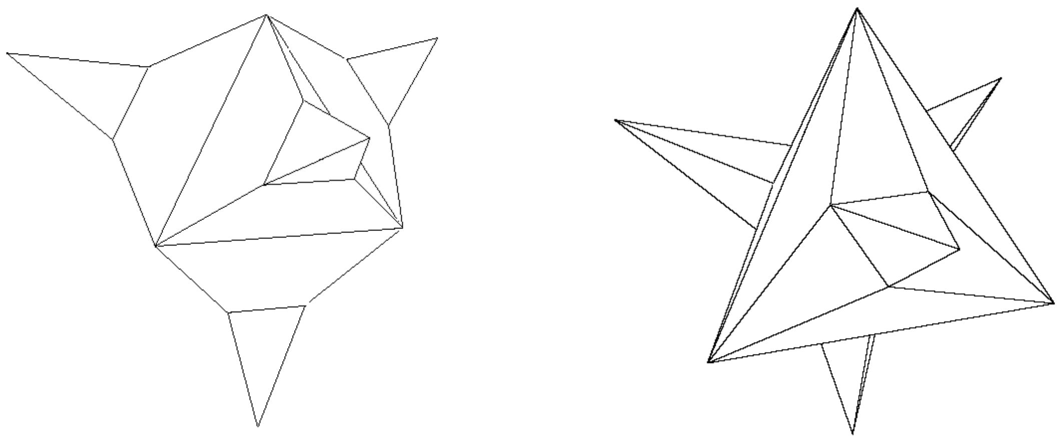
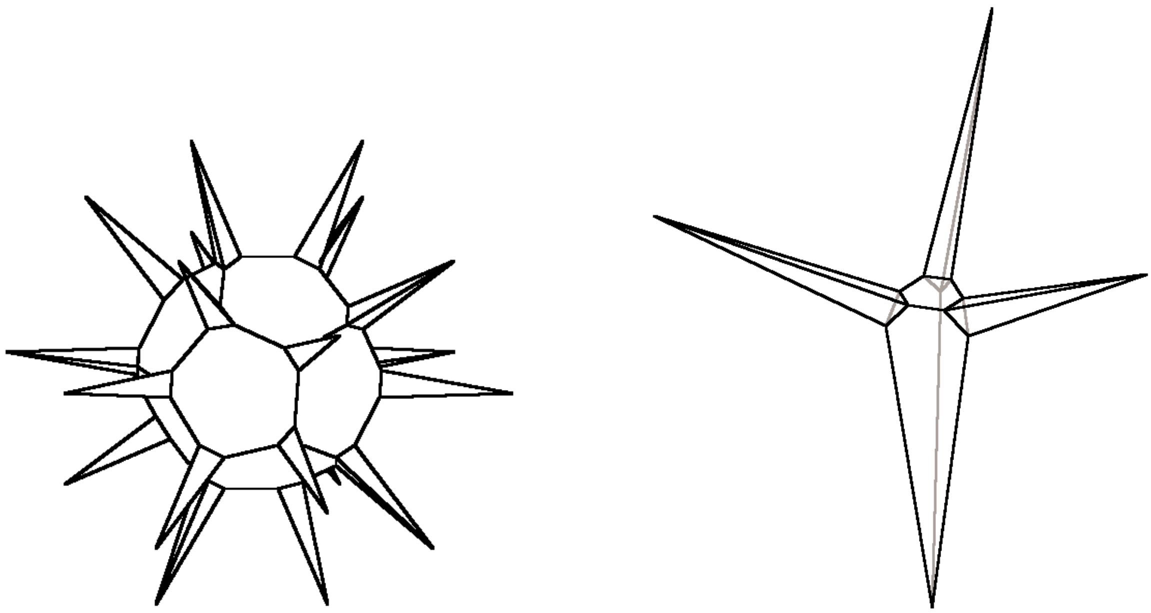


Figure 3.4: Unfoldable polyhedra formed by applying Witch's Hats to the sides of a tetrahedron, from [5].

The work in this area is therefore to characterize precisely which polyhedra are unfoldable. Motivated by Figure 3.2, it is tempting to ask whether every orthogonal polyhedron is unfoldable without overlap. Once again, the answer is negative. The two examples shown in Figure 3.3 have no nets: the first because the smaller cube does not have enough space to unfold within the hole of the larger face, and the second because when two large faces are connected (either directly or via smaller faces), there is not enough space for the smaller faces to unfold [6]. These two examples are similar in that they are not *topologically convex*. A polyhedron is topologically convex if it has the same graph as a convex polyhedron. This is not true for the two polyhedra in Figure 3.3 since the first has a face not topologically equivalent to a disc and the second has instances of two faces being connected at two different edges.

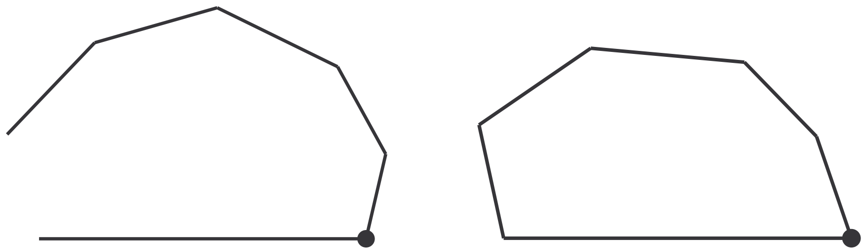
The next logical question is whether all topologically convex polyhedra are unfoldable without overlap. This turns out to be false as well; examples by Tarosov [34] (cited from [19]) and Demaine et al. [4] were found independently. In fact, these polyhedra are even convex faced and starlike. Demaine et al. constructed their example using a structure they call the Witch's Hat: replacing a triangular face of a polyhedron with a terrain that includes a large spike. Making this modification to the faces of a tetrahedron forms an unfoldable polyhedron. See Figure 3.4(a). Tarasov's construction was similar, but he added spikes at the vertices of a polyhedron rather than at the interior of its faces. See Figure 3.5(a).

The Witch's Hat construction was later modified to consist only of triangles [5]. See Figure 3.4(b). This resolved the stronger open question of whether every simplicial polyhedron is



(a) (b)

Figure 3.5: Ununfoldable polyhedra constructed by Grünbaum. (a) An ununfoldable polyhedron similar to the one constructed by Tarasov. Image due to [19]. (b) An ununfoldable polyhedron with 13 faces, from [20].



(a) Long edge and arm (b) Matching endpoints

Figure 3.6: Reconstructing a polygon from a set of sticks.

simply unfoldable: they are not.

A few years later, Grünbaum considered lower bounds on the number of faces of an ununfoldable polyhedron [19]. He constructed an ununfoldable starlike polyhedron with 13 faces, whereas the Witch's Hat construction uses 24 faces and the simplicial version has 36 faces. See Figure 3.5(b). Grünbaum conjectured that 13 was optimal:

Conjecture 3.1.2 (Grünbaum)**.** *Every convex-faced starlike polyhedron with at most* 12 *faces has a net.*

This thesis shall resolve Conjecture 3.1.2 negatively: one of the main results of this thesis is the construction of an ununfoldable polyhedron with 9 faces. We also improve on the number of vertices: Grünbaum's ununfoldable polyhedron has 13 vertices, whereas our ununfoldable polyhedron has only 8. See Section 4.4.

3.2 Reconstructing Polyhedra

The underlying idea of a reconstruction problem is simple: one is given (supposed) partial information about an object and must either recreate the original object or determine whether such an object exists. In some cases the object is uniquely specified by the partial information, and in others there can be many satisfying objects and we need only find one. In our case, the objects to be reconstructed are polygons or polyhedra, possibly of a particular class.

3.2.1 Reconstruction from Edge Lengths

An old and well-known result in this area concerns the reconstruction of polygons from edge lengths. It turns out that a polygon can be constructed from a set of edge lengths (or, equivalently, a set of sticks) if and only if the largest stick has length less than the sum of all other lengths (Lemma 3.1 of [22]). This polygon is intuitively quite easy to construct: simply place the longest edge in the plane, then form a flexible arm from the remaining sticks. Attach the arm to one end of the longest edge and bend it so that its other endpoint meets the remaining endpoint of the longest edge. See Figure 3.6.

Unfortunately, there does not seem to be such a simple characterization of the cases in which a *polyhedron* can be built from a set of sticks [15]. The chapter of this thesis on polygon and polyhedron reconstruction is largely motivated by this problem.

3.2.2 Reconstruction from Face Directions

A very different problem is that of reconstructing a polyhedron from its face directions. That is, if one is given a sequence of vectors, can one reconstruct a polyhedron with precisely those vectors as face normals? This problem was studied by Minkowski for convex polyhedra [15]. He showed that for any sequence of vectors that sum to 0, there is a unique convex polyhedron so that each vector is the normal to a face, and the length of each vector is the area of the corresponding face. The problem of reconstructing this polyhedron from the given vectors remains open, however.

Alexandrov later studied reconstruction problems for convex polyhedra in his book *Convex Polyhedra*, recently translated from Russian [2]. Alexandrov considered reconstruction from face normals, like Minkowski, but extended the work to include unbounded polyhedra (i.e. polyhedra with infinite interior). Alexandrov also studied other reconstruction problems, including the reconstruction of convex polyhedra from unfolding information.

3.2.3 Reconstruction from Nets

A net is formed by unfolding a polyhedron into the plane. One might therefore consider the opposite act: folding a net into a polyhedron. This gives rise to a number of related problems. What is the complexity of determining whether a net can be folded into a polyhedron? What is the complexity of finding a polyhedron that can be folded from a given net? These are major open problems in this area, and have been the focus of much study.

Convex Polyhedra

There are a pair of famous results in this area that apply to convex polyhedra: Cauchy's Rigidity Theorem and Alexandrov's Theorem.

Cauchy's Rigidity Theorem states that there is only one convex polyhedron (up to rigid transformation) with a given net. In other words, a convex polyhedron is rigid: the faces of

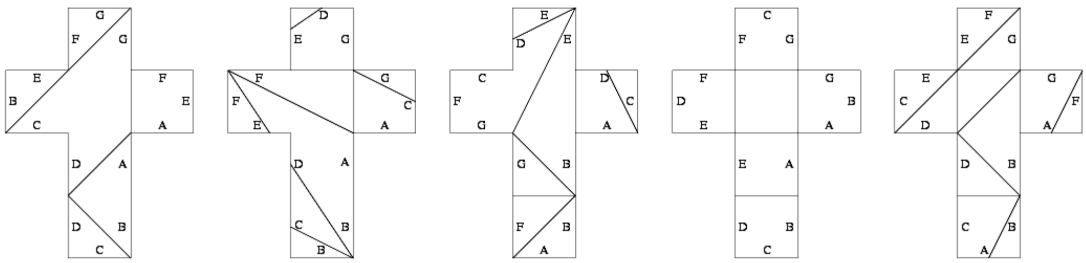


Figure 3.7: Five foldings of the latin cross, taken from [10].

a convex polyhedron cannot be flexed to form a new polyhedron with the same combinatorial structure.

Alexandrov's Theorem provides a stronger result that guarantees existence as well as uniqueness. However, Alexandrov's Theorem does not apply to nets. We must therefore define (albeit informally) Alexandrov's notion of a *development*. A development is a closed chain in the plane, much like a polygon, but with the possibility of overlap. A development also includes gluing information between edges. Each edge of the chain is glued to exactly one other edge, and the orientation of this gluing is given (so that it is known which endpoint is glued to which). The resulting planar object is very similar to an unfolding, except that no internal creases (i.e. divisions between faces) are given.

Theorem 3.2.1 (Alexandrov). *Each development homeomorphic to the sphere with the sum of angles at most 2π at each vertex defines a unique closed convex polyhedron by gluing.*

The uniqueness stated in Alexandrov's Theorem comes from the fact (also proven by Alexandrov) that two closed convex polyhedra with the same development must be the same up to rigid transformations. Thus each development that satisfies the conditions of Alexandrov's Theorem defines exactly one convex polyhedron by gluing.

Alexandrov's Theorem is very powerful, in that it completely characterizes the ways in which a net can be folded into a convex polyhedron by performing edge-to-edge gluings. Using this theorem, Lubiw and O'Rourke created a polynomial-time algorithm to determine all edge-to-edge gluings for a polygon that will yield a convex polyhedron [23]. An interesting result of this research is that the Latin Cross unfolding of the cube can be glued in different ways to obtain five different convex polyhedra. See Figure 3.7.

The complexity of actually constructing a polyhedron from a development remains unknown.

One can determine a superset of the creases to be used in the folding process in linear time by taking shortest paths between all pairs of vertices [23, 27]. However, even if one has the creases (that is, one has a net), the complexity of discovering the polyhedron to which it folds is unknown. Cauchy's Theorem implies uniqueness of a convex polyhedron folded from a net, but does not give a way to construct it.

General Polyhedra

When we consider non-convex polyhedra, Cauchy's Rigidity Theorem and Alexandrov's Theorem no longer apply. The complexity of determining whether a polyhedron even exists with a given net is not yet known. Biedl et al. proved that this problem is weakly NP-Complete when both the net and the constructed polyhedron are orthogonal [7]. The more general version of this problem is still open.

Chapter 4

Ununfoldable Polyhedra

In this chapter we explore polyhedra for which every edge-unfolding has an overlap. We introduce the notion of a k-local overlap in an unfolding, which is an overlap between faces that are connected by a path of at most k vertices in the unfolding. We present an example of a polyhedron with 16 triangular faces for which 1-local overlaps cannot be avoided, and hence no k-local overlaps can be avoided. We then modify this example to form an ununfoldable polyhedron with 9 faces. Finally, we show that if cuts through faces are allowed then certain types of local overlaps can be avoided.

4.1 Introduction

In the study of polyhedron unfolding, we are primarily concerned with overlaps. In particular, we wish to study cases where every unfolding of a polyhedron contains overlapping faces. It has been known for some time that such polyhedra, the ununfoldable polyhedra, exist [4, 5, 19, 34]. The focus of study in this area has since turned to characterizing the ununfoldable polyhedra. For example, some work has been done on showing that certain classes of orthogonal polyhedra can always be unfolded without overlap [6]. There has also been work on finding the smallest, simplest examples of ununfoldable polyhedra [20].

One problem in the analysis of unfoldings is that an overlap may occur between two faces in vastly different parts of a polyhedron. Such an overlap seems to occur for no theoretically satisfying reason; it is simply an artifact of the unfolding in question. By contrast, the analysis of overlaps between faces that are close together seems intuitively simpler.

Motivated by these informal ideas, we introduce a measure of the locality of an overlap. We shall make formal definitions later, but for now we say informally that an overlap is *k-local* if

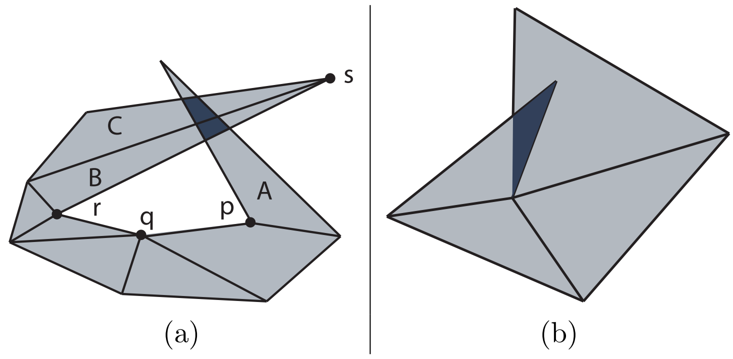


Figure 4.1: Examples of k-local overlaps for (a) $k = 3, 4$ and (b) $k = 1$

the overlapping faces are connected *in the unfolding* by a path containing at most k vertices. Note that we are concerned with the relationship between the faces in the unfolding, not in the polyhedron.

The known ununfoldable polyhedra have no simple unfoldings, but the overlaps in their unfoldings have varying degrees of locality. One of the motivating questions for this chapter is whether or not k-local overlaps can be avoided for all convex-faced polyhedra, for some sufficiently small k. It will turn out that the answer is "no": there is a polyhedron for which every unfolding contains a 1-local overlap, and hence a k-local overlap for all k. We then modify our example of a polyhedron that cannot avoid 1-overlaps to find a convex-faced ununfoldable polyhedron with 9 faces, where the previously known smallest example has 13 [20].

4.2 Local Overlaps

We now define precisely what we mean by a k-local overlap, and discuss some basic properties thereof.

Suppose P is a polyhedron with an unfolding P'. Suppose further that there is an overlap between faces f_1 and f_2. Then if there are at most k vertices in the shortest path along edges of P' starting with a vertex incident to f_1 and ending with a vertex incident with f_2. See Figure 4.1 for an example. In particular, an overlap is 1-*local* if f_1 and f_2 are both incident with a common vertex. If the vertex separating f_1 and f_2 is an image of vertex v of P, we say that the 1-local overlap *occurs at* v.

Lemma 4.2.1. *Suppose we unfold a polyhedron P. Then a* 1*-local overlap will occur at vertex v if and only if v has an unfolding angle greater than 2π.*

Proof. Suppose an image of v, say v', has interior angle greater than 2π in the unfolding. Then the faces incident with v' in P' cannot be placed in the plane without overlap, as v' has only an angle of 2π around it in the plane. But in such an overlap, all faces involved are incident with v', and hence the overlap is 1-local.

On the other hand, suppose the total face angle at v' is no more than 2π for each image v' of v. Since the faces are convex, each will lie in a sector of the plane centered at v' corresponding to its face angle with v'. These sectors will not overlap, since the total face angle around v' is no more than 2π. Thus no two faces incident with a single image of v will intersect in the unfolding and hence no 1-local overlaps occur at v. □

Corollary 4.2.2. *A* 1*-local overlap cannot occur at a vertex with non-negative curvature.*

Proof. A 1-local overlap can occur at vertex v only if v has an unfolding angle greater than 2π. But the unfolding angles at v are no greater than the total face angle at v. Hence a 1-local overlap can occur at v only if v has total face angle greater than 2π, which is equivalent to v having negative curvature. □

Corollary 4.2.3. *No unfolding of a convex polyhedron contains a* 1*-local overlap.*

Proof. Follows from the fact that convex polyhedra contain no vertices with negative curvature. □

Corollary 4.2.4. *Any edge cut tree for P that avoids* 1*-local overlaps must have degree at least* 2 *at all vertices of P with negative curvature.*

Proof. Let T be an edge cut tree for P, and suppose v is a vertex of P with negative curvature such that the degree of v in T is 1. Then v has only one unfolding component, and hence it has an unfolding angle equal to its total face angle. But the total face angle at v is greater than 2π, and hence by Lemma 4.2.1 the unfolding implied by T has a 1-local overlap at v. □

4.3 Unavoidability of 1-Local Overlaps

In this section we show that there are polyhedra for which every unfolding contains a 1-local overlap. Even more surprisingly, there is such a polyhedron that is star-shaped, simplicial, and contains only 16 faces. It should be noted that the previously smallest known example of an ununfoldable simplicial polyhedron has 36 faces [5].

4.3.1 Intuition

The construction of unfoldable polyhedra is related to the notion of curvature. Vertices with negative curvature seem (informally) to present the largest barrier to simple unfolding. A strategy for finding an ununfoldable polyhedron is to have as many vertices with negative curvature as possible.

A theorem of Descartes states that the total sum of all curvatures on a polyhedron of genus zero is 4π. It is therefore not possible to create a polyhedron such that all vertices have negative curvature. Indeed, since the curvature at a vertex must be strictly less than 2π, a polyhedron requires at least three vertices with positive curvature. One strategy for forming an ununfoldable polyhedron is therefore to create a few vertices with as large a curvature as possible; for example, the apex of a sufficiently sharp spike. The remaining vertices may then have negative curvature, impeding the creation of a simple unfolding. All examples of ununfoldable polyhedron constructed at the time of this writing were created via this strategy: the witch's hats of [4, 5] and the spiky polyhedra of [34, 19].

We applied a similar method to find our polyhedron that cannot avoid 1-local overlaps. We use 4 spikes, roughly aligned as axes in a plane. The reasoning behind the use of four spikes, as opposed to the lower bound of three, will become more clear when we discuss the three-pointed version in Section 4.4.

4.3.2 The Polyhedron

In this section we describe our example of a polyhedron for which every unfolding contains a 1-local overlap. See Figure 4.2 for an illustration.

Let α and β be positive values. We think of α as being large and β as being small. Our polyhedron then consists of four spikes, having endpoints $A_1 = (\alpha, 0, 0)$, $A_2 = (0, -\alpha, 0)$, $A_3 = (-\alpha, 0, 0)$, and $A_4 = (0, \alpha, 0)$. These spikes intersect pairwise at vertices $B_1 = (1, 1, 0)$, $B_2 = (1, -1, 0)$, $B_3 = (-1, -1, 0)$, and $B_4 = (-1, 1, 0)$. In other words, there is an edge from A_i to B_j if and only if $i = j$ or $i + 1 \equiv j \mod 4$. Finally, there are central points $C_1 = (0, 0, \beta)$ and $C_2 = (0, 0, -\beta)$ that are connected to all other vertices, but not to each other.

The result is a symmetrical four-pointed star. We shall denote this polyhedron by $P_4(\alpha, \beta)$. See Figure 4.2.

We now wish to analyze the face angles in this construction. To do this, we consider the class of polyhedra $P_4(\alpha, \beta)$ as $\alpha \to \infty$ and $\beta \to 0$. In the limit, the vertices A_i occur at infinity, the vertices C_1 and C_2 both occur at the origin, and our polyhedron is a doubly-covered, infinite portion of the plane. See Figure 4.3. We consider this highly degenerate construction because its angles are easy to analyze. The face angles at each A_i vertex are 0, the face angles at each B_i

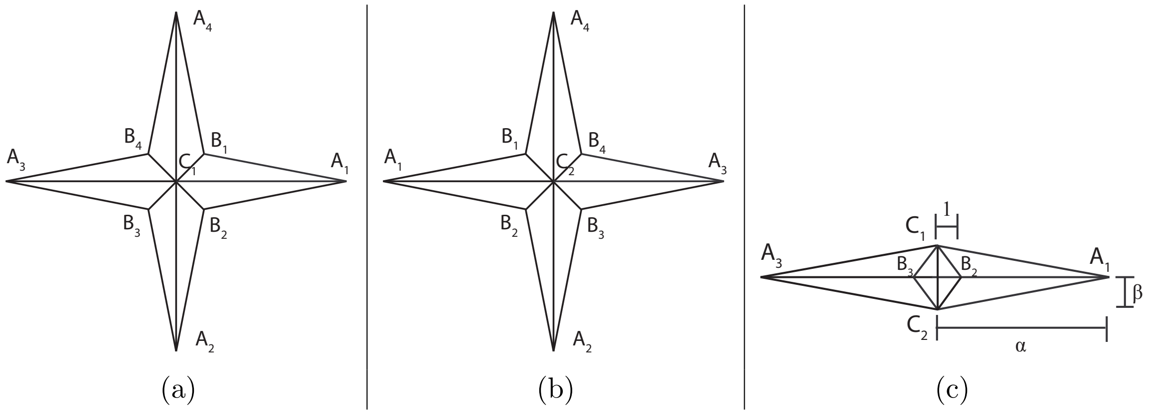


Figure 4.2: Different views of polyhedron $P_4(\alpha, \beta)$: (a) the top view, (b) the bottom view, and (c) a view from the side, illustrating parameters α and β.

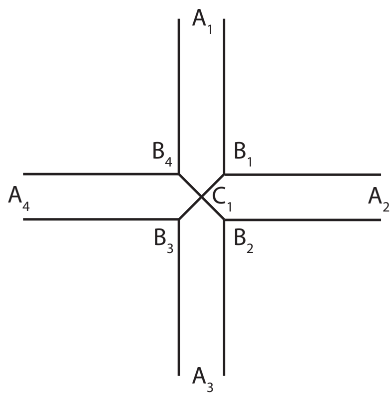


Figure 4.3: The polyhedron $P_4(\alpha, \beta)$ as $\alpha \to \infty$ and $\beta \to 0$.

vertex are $\frac{3\pi}{4}$, and the face angles at each C_i vertex are $\frac{\pi}{4}$.

Now let us return to the non-limit case. As α becomes large and β becomes small, our face angles will approach the limit values given above. In particular, the face angles at each B_i will approach $\frac{3\pi}{4}$ from below and the angles at each C_i will approach $\frac{\pi}{4}$ from above. To see the latter, note that each C_i vertex is in fact a saddle point, and will therefore have negative curvature.

To show that $P_4(\alpha, \beta)$ is ununfoldable, we shall require that the face angles at each B_i vertex are all greater than $\frac{2\pi}{3}$. We therefore take α large enough and β small enough that this condition is satisfied. Values $\alpha \geq 10$ and $\beta \leq 1$ are sufficient for this purpose. Thus, for the remainder of this section, we shall consider the polyhedron $P_4(10, 1)$.

4.3.3 1-Local Overlaps

We now prove the main result of this section, which is that all unfoldings of the polyhedron $P_4(10, 1)$ have a 1-local overlap. We proceed by a sequence of lemmas regarding the nature of cut trees for $P_4(10, 1)$. For convenience of notation, we shall henceforth use P_4 to denote $P_4(10, 1)$.

Lemma 4.3.1. *Any cut tree for P_4 that avoids 1-local overlaps must cut at least two opposing edges incident to each of B_i. In other words, there must be cuts from each B_i to the two adjacent A_j vertices or to the vertices C_1 and C_2.*

Proof. As discussed when we constructed P_4, the face angles at each B_i are larger than $\frac{2\pi}{3}$.

Suppose that there is a cut tree that does not cut two opposing edges of some B_i. Then, since there are four faces incident with B_i, three of those faces would be in a single unfolding component. But then the unfolding angle of that component is greater than $3\frac{2\pi}{3} = 2\pi$. Lemma 4.2.1 then implies that this unfolding contains a 1-local overlap.

□

Lemma 4.3.2. *Any cut tree for P_4 that avoids 1-local overlaps must cut from some B_i (without loss of generality, B_1) to C_1 and C_2. For the other three vertices B_i, the edges from B_i to the adjacent A_j vertices must be cut.*

Proof. By Lemma 4.3.1, each B_i must have a pair of opposing cuts. These cuts will either be to C_1 and C_2, or to the two adjacent A_j vertices. Now if all B_i have cuts to their adjacent A_i vertices then the cut tree will contain a cycle: $B_1, A_1, B_2, A_2, B_3, A_3, B_4, A_4, B_1$. Thus at least one B_i must have cuts to the points C_1 and C_2.

Suppose there are cuts from vertex B_i to C_1 and C_2, and also cuts from some other vertex B_j to C_1 and C_2. Then we get another cycle in the cut tree: B_i, C_1, B_j, C_2, B_i. We conclude that there can be cuts from only one B_i to both C_1 and C_2. There must therefore be cuts from the remaining B_i's to their adjacent A_j vertices, as required. □

Lemma 4.3.3. *Any cut tree for P_4 that avoids* 1*-local overlaps must cut at least two edges incident to each of C_1 and C_2.*

Proof. Each of C_1 and C_2 has negative curvature, so the result follows from Lemma 4.2.4. □

Lemma 4.3.4. *Any unfolding of P_4 must contain a* 1*-local overlap.*

Proof. Recall that a cut tree must be a spanning tree over the vertices of P_4. Since P_4 has 10 vertices, any cut tree for P_4 must have 9 cuts.

Lemma 4.3.2 describes 8 cuts that must be made, those being (B_1, C_1), (B_1, C_2), (B_2, A_1), (B_2, A_2), (B_3, A_2), (B_3, A_3), (B_4, A_3), and (B_4, A_4). But of these, only one cut is incident to each of C_1 and C_2. By Lemma 4.3.3, there must be an additional cut incident to each of C_1 and C_2. There is only one cut left to make, but since C_1 and C_2 are not adjacent this cut cannot be made incident with both. There is therefore no cut tree for P_4 for which the associated unfolding has no 1-local overlaps. □

Theorem 4.3.5. *There exists a simplicial, star-like polyhedron with* 16 *faces for which every unfolding contains a* 1*-local overlap*

Proof. The polyhedron $P_4(10, 1)$ is simplicial with 16 faces. By Lemma 4.3.4, any unfolding of $P_4(10, 1)$ contains a 1-local overlap. To see that $P_4(10, 1)$ is star-like, consider the origin $(0, 0, 0)$ in the interior of $P_4(10, 1)$. The interior of every face of $P_4(10, 1)$ is visible from this point. Thus $P_4(10, 1)$ satisfies the requirements of this theorem. □

4.4 A Small Ununfoldable Polyhedron

We now modify our example from the previous section to create a smaller polyhedron for which every edge-unfolding contains an overlap. The idea is to use only three spikes instead of four. As we shall see, if these spikes are arranged into a three-way symmetrical structure then the resulting polyhedron can be unfolded without overlap. However, if the spikes are perturbed to remove some symmetry, we obtain an ununfoldable polyhedron.

4.4.1 Symmetric Example

Here we present the polyhedron obtained via a similar construction as $P_4(\alpha, \beta)$, using only three spikes instead of four. We also simplify the spikes so that each is made of three faces instead of four: two on top and one on the bottom. We call this polyhedron $P_3(\alpha, \beta)$. See Figure 4.4 for an illustration of this polyhedron. We shall see that $P_3(\alpha, \beta)$ can be edge-unfolded without overlap for any choice of α and β.

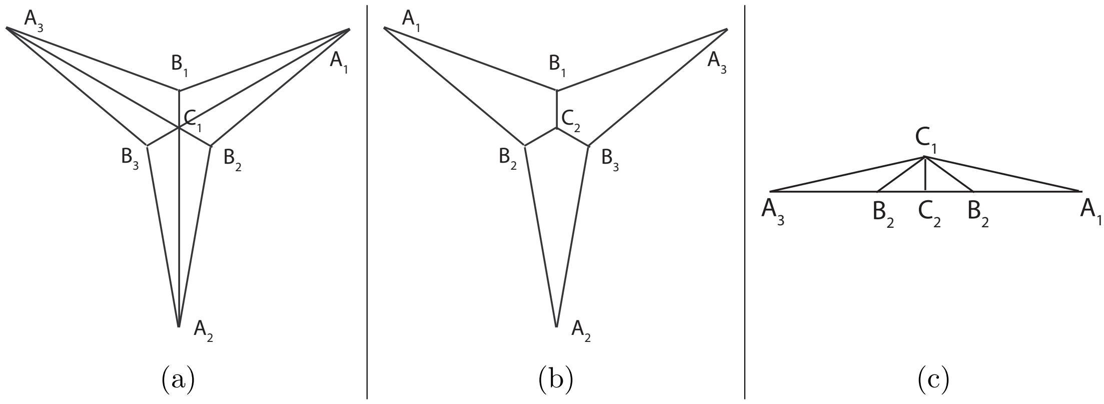


Figure 4.4: Different views of polyhedron P_3: (a) the top view, (b) the bottom view, and (c) a view from the side.

Just as with P_4, the points A_i and B_i lie in the xy-plane. The point C_1 lies at $(0, 0, \beta)$, but C_2 is now located at the origin. One can think of our simplification of the spikes as taking only the part of a spike that lies above the xy-plane. The faces incident with C_2 are now quadrilaterals that lie in the xy-plane.

This polyhedron has the undesirable property that adjacent faces (those incident with C_2) are coplanar. This degeneracy can be removed by perturbing each A_i slightly in the negative z direction and moving C_2 the appropriate amount in the positive z direction to preserve coplanarity. We shall leave a more formal analysis of this process until the next section, when we analyze a slightly modified polyhedron. For now, we shall leave $P_3(\alpha, \beta)$ degenerate.

Now consider the face angles of P_3. Just as in our analysis of P_4, this is easiest if we consider the limit case of $\alpha = \infty$ and $\beta = 0$. In this degenerate polyhedron, the face angles at A_i are 0, at B_i are $\frac{2\pi}{3}$, at C_1 are $\frac{\pi}{3}$, and at C_2 are $\frac{2\pi}{3}$. In the non-limit case, the face angles in the polyhedron approach these limit values (from below in the case of B_i, and from above in all other cases).

The polyhedron P_3 has a simple edge-unfolding for any values of the parameters α and β. For an illustration of this unfolding, see Figure 4.5. Note that, in this illustration, we have taken α to be sufficiently large and β sufficiently small that edges incident with any given A_i can be seen as arbitrarily close to parallel.

The intuition as to why this unfolding can occur is that the face angles at vertices B_i are all less than $\frac{2\pi}{3}$. Thus, in an unfolding, three of these faces can be adjacent without causing an overlap. Compare this to the situation with P_4 in Section 4.3, where the face angles at the vertices B_i were all greater than $\frac{2\pi}{3}$ (in fact, only slightly less than $\frac{3\pi}{4}$). That posed the restriction that

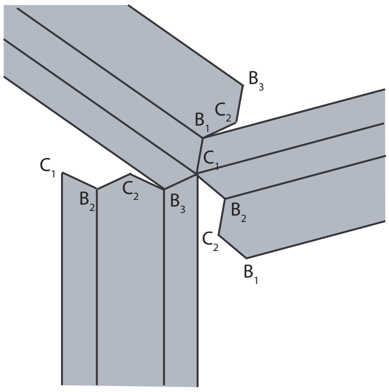


Figure 4.5: A simple unfolding for P_3

no three faces at B_i could be adjacent, and hence two opposing edges adjacent to B_i needed to be cut. In this example with three spikes, that restriction is no longer present, and hence there is more freedom in how cuts can be made. The trick to making this polyhedron ununfoldable seems to be to create face angles greater than $\frac{2\pi}{3}$ at the B_i vertices.

4.4.2 Asymmetric Example

This section will contain a number of arguments in which various angles and sums of angles are compared. Hence, for improved readability, we shall switch to expressing angles in degrees for the remainder of this section.

As discussed above, we would like to have face angles about the vertices B_i that are greater than $120°$. The geometry prevents this from being true for all three vertices. However, if we perturb the relative angles between the spikes, we can cause the face angles at two vertices, say B_2 and B_3, to increase while the face angles at the other decrease.

In more detail, consider the vectors from the origin to each vertex A_i. In P_3, the angles between these vectors are all $120°$, by symmetry. We shall now rotate vertices A_1 and A_3 toward A_2 about the origin, so that they each make angles of $120° - \phi$. The angle between A_1 and A_3 about the origin will therefore increase to $120° + 2\phi$. Identical spikes, of the form in P_3, now emanate from each A_i toward the origin, and the resulting polyhedron will be denoted $P_3^{\phi}(\alpha, \beta)$. See Figure 4.6 for an illustration.

It should be noticed that the faces incident with C_2 are coplanar, and the total face angle at C_2 is therefore $360°$. This is a slight degeneracy that is undesirable. We shall proceed to analyze this polyhedron as is, but at the end of this section we shall show how to modify $P_3^{\phi}(\alpha, \beta)$ to

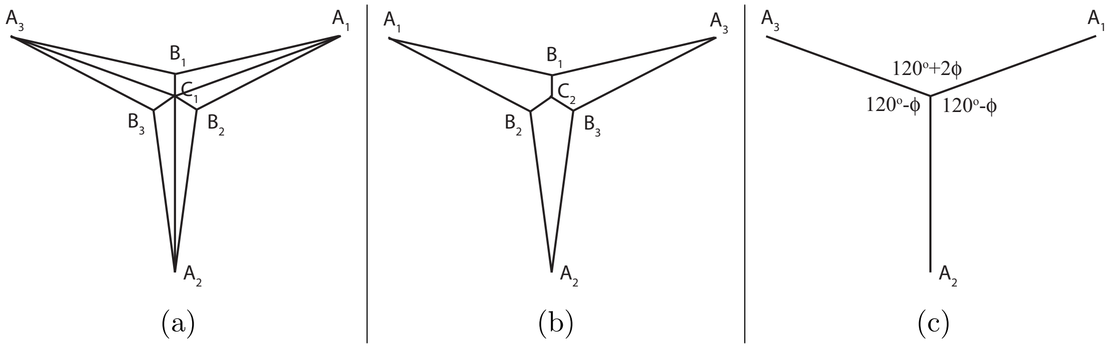


Figure 4.6: The polyhedron P_3^ϕ as seen from (a) the top and (b) the bottom. In (c) the positioning of vertices A_1, A_2, and A_3 is shown.

remove this degeneracy. A continuity argument will imply that the modified polyhedron is also unfoldable.

Now, just as with our analyses of the face angles in $P_4(\alpha, \beta)$ and $P_3(\alpha, \beta)$, we consider the limit case of $\alpha = \infty$ and $\beta = 0$. In this case the face angles at B_1 are $120° - \phi$ and the face angles at B_2 and B_3 are $120° + \frac{1}{2}\phi$. Thus, as $\alpha \to \infty$ and $\beta \to 0$, the face angles of $P_3^\phi(\alpha, \beta)$ will approach these limit values.

In particular, the choices of parameters α and β can be made so that the face angles at B_1 are arbitrarily close to $120° - \phi$. Similarly, the face angles at B_2 and B_3 can be made arbitrarily close to $120° + \frac{1}{2}\phi$, and hence greater than $120°$. Finally, for the non-limit case $\beta > 0$, the curvature at C_1 will always be negative. Note, however, that the curvature at C_2 is precisely 0.

4.4.3 Ununfoldability

We shall now show that there exist parameters α and β for which $P_3^\phi(\alpha, \beta)$ is ununfoldable. We shall use P_3^ϕ to mean "$P_3^\phi(\alpha, \beta)$ for sufficiently large α and sufficiently small β."

Lemma 4.4.1. *Any cut tree for P_3^ϕ that avoids* 1*-local overlaps must cut two opposing edges incident to each of B_2 and B_3.*

Proof. Since the face angles at B_2 and B_3 can be made arbitrarily close to $120° + \frac{1}{2}\phi$, they can certainly be made greater than $120°$. The result then follows as in the proof of Lemma 4.3.1. □

Lemma 4.4.2. *Any cut tree for P_3^ϕ that avoids* 1*-local overlaps must contain two edges incident to C_1 and B_1.*

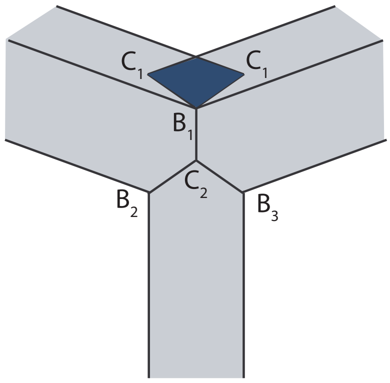


Figure 4.7: An overlap in an unfolding of P_3^ϕ, from Lemma 4.4.3. Note that the edge (B_1, C_2) is cut, but since the curvature at C_2 is 0 both images of B_1 occur at the same location.

Proof. The result follows from Corollary 4.2.4 and the fact that C_1 and B_1 have negative curvature. □

Lemma 4.4.3. *Any cut tree for P_3^ϕ that avoids overlaps must have cuts from one of B_2 or B_3 to both C_1 and C_2.*

Proof. Suppose a cut tree that does not have an overlap does not have the specified cuts. Then, by Lemma 4.4.1, the cut tree must have cuts (B_2, A_1), (B_2, A_2), (B_3, A_2), and (B_3, A_3). Recall that since the cut tree is a spanning tree over the 8 vertices of P_3^ϕ, it must have 7 edges.

Now there are 3 cuts left to make. Vertex B_1 must have two incident cuts, as must C_1 (by Lemma 4.4.2). There must also be a cut incident to C_2 (even though the curvature at C_2 is 0; see Section 2.2.2).

We conclude that there must be cuts (B_1, C_1) and (B_1, C_2), and the final cut must be incident with C_1. But then a portion of the unfolding will look as in Figure 4.7, which contains an overlap. We conclude that a cut tree without the given cuts will generate an overlap. □

We shall now again consider the highly degenerate case of $P_3^{10^\circ}(\infty, 0)$. This construct is a double-covering of an unbounded portion of the xy-plane. See Figure 4.8 for an illustration. Vertices C_1 and C_2 both occur at the origin and the vertices A_i are considered to lie at infinity. We shall show that overlaps with interior points occur in every unfolding of $P_3^{10^\circ}(\infty, 0)$. Then, as this is the limit of our true polyhedra, it will turn out that all the overlaps presented will still occur in $P_3^{10^\circ}(\alpha, \beta)$ for sufficiently extreme values of α and β.

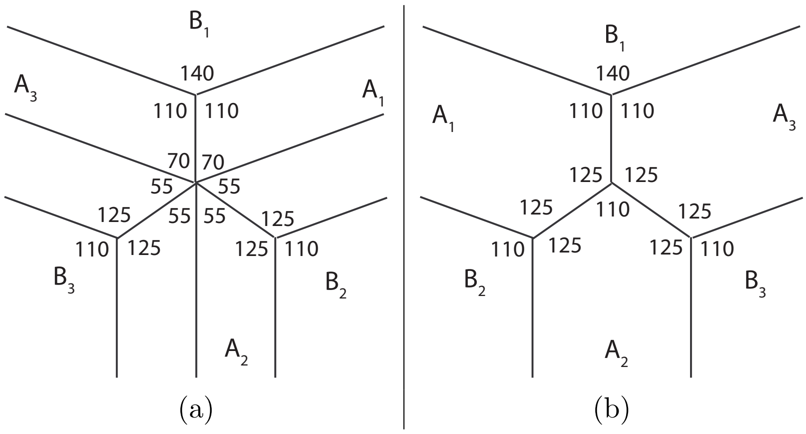


Figure 4.8: The centre portion of polyhedron $P_3^{10^\circ}(\infty, 0)$ with all face angles given (in degrees). The polyhedron is shown (a) from the top and (b) from the bottom.

Lemma 4.4.4. *Any unfolding of $P_3^{10^\circ}(\infty, 0)$ must contain an overlap with interior points.*

Proof. See Figure 4.8 for an illustration of $P_3^{10^\circ}(\infty, 0)$. This is a degenerate case where the polyhedron lies entirely in the xy-plane, and the vertices A_i lie at infinity. All edges to a given vertex A_i are therefore parallel.

Suppose there exists a cut tree for $P_3^{10^\circ}(\infty, 0)$ that generates a simple unfolding. Without loss of generality this tree must include edges (B_2, C_1) and (B_2, C_2) by Lemma 4.4.3. Then, by Lemma 4.4.1, the tree must also include edges (B_3, A_2) and (B_3, A_3) (since if B_3 were cut to C_1 and C_2 then the cut tree would contain cycle C_1, B_3, C_2, B_2, C_1).

Since our cut tree must contain 7 edges and we have specified 4, there are now 3 edges left to choose. By Lemma 4.4.2, at least two edges in the cut tree must be incident with B_1 and at least one must be incident with C_1. Also, since the cut tree is spanning, at least one edge must be incident with A_1.

We shall proceed by cases on the vertices adjacent to B_1 in the cut tree T. For each case we present a diagram illustrating the overlap that occurs. We have omitted the details of proving that these overlaps actually do occur as we have drawn them; the proofs are unenlightening and quite lengthy. Suffice it to say that all illustrations are unfoldings of $P_3^{10^\circ}(\infty, 0)$ and can be verified via simple geometry.

Case 1: C_1 and C_2 are adjacent to B_1 in T. Then our cut tree contains cycle B_1, C_1, B_3, C_2, B_1, a contradiction.

Case 2: A_1 and A_3 are adjacent to B_1 in T. The final cut must then be incident with

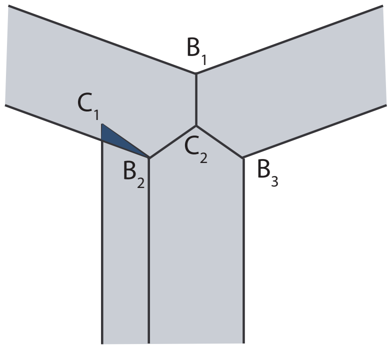


Figure 4.9: The overlap that occurs in the unfolding of P_3^ϕ in Case 2 of Lemma 4.4.4

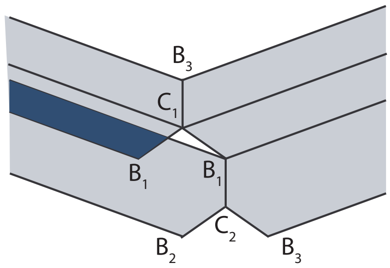


Figure 4.10: The overlap that occurs in the unfolding of P_3^ϕ in Case 4 of Lemma 4.4.4

C_1. But then, in particular, edges (B_1, C_2), (B_3, C_2) and (B_2, A_2) are not in the cut tree. Thus the unfolding will have the overlap illustrated in Figure 4.9, since each of the three face angles shown at B_2 are greater than $\frac{2\pi}{3}$.

Case 3: C_1 **and** A_3 **are adjacent to** B_1 **in** T**.** The final cut must then be incident with A_1. But then edges (B_1, C_2), (B_3, C_2) and (B_2, A_2) are not in the cut tree. As in the previous case, the unfolding will have the overlap illustrated in Figure 4.9.

Case 4: C_1 **and** A_1 **are adjacent to** B_1 **in** T**.** The final edge must be incident with one of B_3, A_2, or A_3 to form a connected tree. Note that if this final edge is not one of (B_3, C_2) or (B_2, A_2) then the overlap from the previous two cases, shown in Figure 4.9, will occur. However, if the edge is one of those, then the overlap shown in Figure 4.10 will occur. Thus, in either case, the corresponding unfolding will have an overlap.

Case 5: C_2 **and** A_1 **are adjacent to** B_1 **in** T**.** The final edge must be incident with C_1, and also with one of A_2, A_3, or B_2. In any of these cases, the overlap shown in Figure 4.11 will

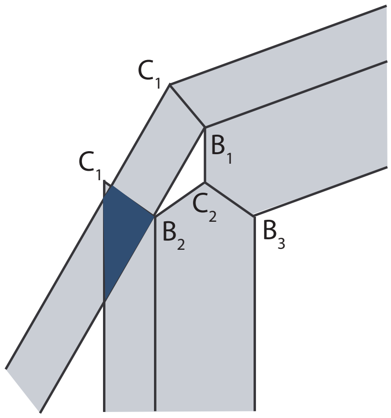


Figure 4.11: The overlap that occurs in the unfolding of P_3^{ϕ} in Case 5 of Lemma 4.4.4

occur.

Case 6: C_2 and A_3 are adjacent to B_1 in T. Then the final edge must be (C_1, A_1). The unfolding is then as shown in Figure 4.12. This unfolding has an overlap.

Thus, in all cases, the corresponding unfoldings are not simple. We conclude that there is no cut tree for P_3 that generates a simple unfolding, as required. $\square$

Theorem 4.4.5. *There exist parameters $0 < \beta < \infty$ and $0 < \alpha < \infty$ such that every unfolding of $P_3^{10^\circ}(\alpha, \beta)$ contains an overlap.*

Proof. By Lemma 4.4.4, any unfolding for $P_3^{10^\circ}(\infty, 0)$ contains an overlap with interior points. Enumerate all combinatorial unfoldings of $P_3^{10^\circ}(\alpha, \beta)$ as $U_1, U_2, \ldots, U_k$. Then, for each $1 \leq i \leq k$, unfolding U_i applied to $P_3^{10^\circ}(\infty, 0)$ contains an overlap with interior points. Then, by continuity, there exist $\epsilon_i > 0$ and $\Gamma_i > 0$ such that for all $\alpha > \Gamma_i$ and $0 < \beta < \epsilon_i$, the unfolding U_i applied to $P_3^{10^\circ}(\alpha, \beta)$ will contain an overlap.

Now take $\epsilon = \min_i\{\epsilon_i\}$ and $\Gamma = \max_i\{\Gamma_i\}$. Set $\alpha = \Gamma + 1$ and $\beta = \frac{1}{2}\epsilon$. Then for every U_i we have that $\alpha > \Gamma_i$ and $0 < \beta < \epsilon_i$, so U_i will contain an overlap when applied to $P_3^{10^\circ}(\alpha, \beta)$. Thus any edge-unfolding of this polyhedron will contain an overlap, as required. $\square$

4.4.4 Removing Coplanar Faces

We now modify polyhedron P_3^{ϕ} to remove the coplanarity of the faces incident with C_2. Consider moving point C_2 so that, instead of lying at $(0, 0, 0)$, it lies at $(0, 0, \gamma)$ for some $0 < \gamma < \beta$. We

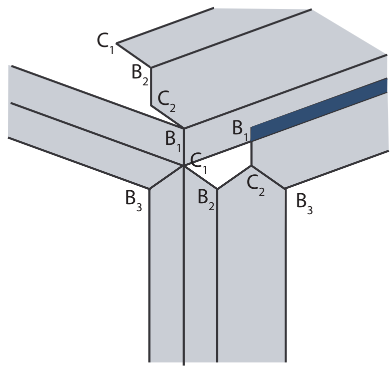


Figure 4.12: The overlap that occurs in the unfolding of P_3^{ϕ} in Case 6 of Lemma 4.4.4

understand γ to be arbitrarily small (and an arbitrarily small fraction of β). We would then move each A_i below the xy-plane so that each face incident with C_2 remains planar. This modified polyhedron will be called $P_3^{\phi}(\alpha, \beta, \gamma)$. Note that no adjacent faces are coplanar in this polyhedron.

If α and β are fixed, for any ϵ we can choose $0 < \gamma < \epsilon$ such that the displacement of each A_i is less than ϵ. A continuity argument now implies that there are parameters such that $P_3^{\phi}(\alpha, \beta, \gamma)$ has no simple unfolding.

In more detail, Theorem 4.4.5 implies that all unfoldings of $P_3^{10°}(\alpha, \beta)$ contain overlaps with interior points for certain parameters α and β. The new parameter γ can be chosen small enough that any unfolding of $P_3^{10°}(\alpha, \beta, \gamma)$ is arbitrarily similar to the corresponding unfolding of $P_3^{10°}(\alpha, \beta)$ (that is, all vertices in an unfolding of $P_3^{10°}(\alpha, \beta, \gamma)$ will be arbitrarily close to their corresponding images in $P_3^{10°}(\alpha, \beta)$). Thus we can take γ small enough that every unfolding of $P_3^{10°}(\alpha, \beta, \gamma)$ has an overlap, as required.

Theorem 4.4.6. *There exists a starlike ununfoldable polyhedron with* 9 *faces.*

Proof. It was demonstrated above that $P_3^{10°}(\alpha, \beta, \gamma)$ is a polyhedron that satisfies the required conditions, except for being starlike. To see that $P_3^{10°}(\alpha, \beta, \gamma)$ is starlike, consider the point $(0, 0, \delta)$ for any $\gamma < \delta < \beta$. The quadrilateral faces are all visible from this point, since it lies above C_2. The triangular faces are also all visible from this point, since it lies below C_1. Thus $P_3^{10°}(\alpha, \beta, \gamma)$ is starlike as required. □

4.5 Arbitrary Cuts

In Section 4.3 we saw an example of a polyhedron for which every edge-unfolding contains a 1-local overlap. Indeed, this form of overlap is quite trivial: it corresponds to a vertex in the unfolding with total face angle greater than 2π.

We now extend our consideration from edge-unfoldings to arbitrary unfoldings. That is, we allow cuts to cross faces. It is tempting to believe that a polyhedron for which every edge-unfolding has a vertex with interior angle greater than 2π may have a similar property for general unfoldings as well. Such a result would be exciting, as it would resolve the open problem of whether all polyhedra can be cut across faces and unfolded into the plane without overlap [5]. Alas, we shall show that the trivial sort of overlap that must occur in an unfolding of $P_4(10, 1)$ can always be avoided with arbitrary unfoldings. More specifically, we shall prove the following theorem:

Theorem 4.5.1. *Any polyhedron P of genus 0 can be cut along its surface and unfolded such that no vertex in the unfolding has total face angle greater than 2π.*

For the remainder of this section, let P be any polyhedron of genus 0 with n vertices. We shall construct an arbitrary cut tree C upon the surface of P. Recall that this means C is a spanning tree for the vertices of P and the cuts of C are allowed to cross faces of P.

The idea of our construction is to make cuts into every vertex within a small neighbourhood. These cuts will be simple line segments made uniformly around each vertex. Enough cuts will be made so that every unfolding angle is no more than 2π. These little cuts are then connected together into a tree.

Our proof is now split into three parts. First, we describe the combinatorics of the tree to be constructed. Then we consider the manner in which we cut into a vertex. Finally, we form an embedding of our tree on the surface of the polyhedron.

4.5.1 Combinatorics of the Cut Tree

For each vertex v, let $f(v)$ be the total face angle at v. Let $w'(v) = \left\lceil \frac{f(v)}{2\pi} \right\rceil$.

Lemma 4.5.2.

$$\sum_{v \in P} w'(v) \leq 2n - 2.$$

Proof. Recall by Euler's Formula that the total curvature over all vertices of P must be 4π. Then the total face angle over all n vertices satisfies

$$\sum_{v \in P} f(v) = 2\pi n - 4\pi.$$

But then

$$\sum_{v \in P} \frac{f(v)}{2\pi} = n - 2$$
$$\sum_{v \in P} \left\lceil \frac{f(v)}{2\pi} \right\rceil \leq n + n - 2 \quad \text{(since } P \text{ has } n \text{ vertices)}$$
$$\sum_{v \in P} w'(v) \leq 2n - 2$$

as required. □

We want to create a tree over the vertices in P such that the degree of each vertex v is at least $w'(v)$. We do so by noting the following simple result.

Lemma 4.5.3. *Suppose we have a set of positive integers $w_1, \ldots, w_n$ such that $\sum_{i=1}^{n} w_i = 2n-2$. Then there is a tree T on n vertices $v_1, \ldots, v_n$ such that the degree of v_i is w_i for all $1 \leq i \leq n$.*

Proof. We proceed by induction on n. The result trivially follows for $n = 1$.

For general n, we construct a rooted tree T. Take the root to be vertex v_1 and let $R = w_1$. Note that $R \geq 1$ and we must have at least $R-1$ values among $\{w_2, \ldots, w_n\}$ that are equal to 1 (because the sum of all w_i is $2n-2$). Assume without loss of generality (by relabeling) that these values are $w_2, \ldots, w_R$. We then set the vertices $v_2, \ldots, v_R$ to be children of v_1, all of degree 1.

Now if $w_i = 1$ for all $i > R$ then we must have $n = R+1$, so simply take v_{R+1} to be a child of v_1 and we are done. Otherwise, suppose (again, by relabeling) that $w_n > 1$. Then $\{w_{R+1}, \ldots, w_n\}$ is a set of $n - R$ values that satisfies $\sum_{i=R+1}^{n} w_i = 2(n-R) - 2 + 1$. By induction we therefore know that there is a tree T' over vertices $\{v_{R+1}, \ldots, v_n\}$ such that the degree of each v_i is w_i for $i < n$, and v_n has degree $w_n - 1$. We take T' to be a subtree of T, with v_n a child of v_1. Then the degree of v_n in T is now $w_n - 1 + 1 = w_n$. We conclude that T is the desired tree. □

Corollary 4.5.4. *There is a tree T over the vertices of P such that the degree of v in T is at least $w'(v)$ for each $v \in P$.*

Proof. Label the vertices of P as $v_1, \ldots, v_n$. Take $w_i = w'(v_i)$ for all $2 \leq i \leq n$, and $w_1 = w'(v_1) + \left[\sum_{v \in P} w'(v) - (2n-2)\right]$. The result then follows from Lemma 4.5.3. □

Let T be the tree from Corollary 4.5.4. Let $w(v)$ be the degree of vertex v in tree T. Note that T is simply a combinatorial tree describing which vertices are adjacent; we have not yet embedded T onto the surface of P, which is the focus of the remainder of our proof.

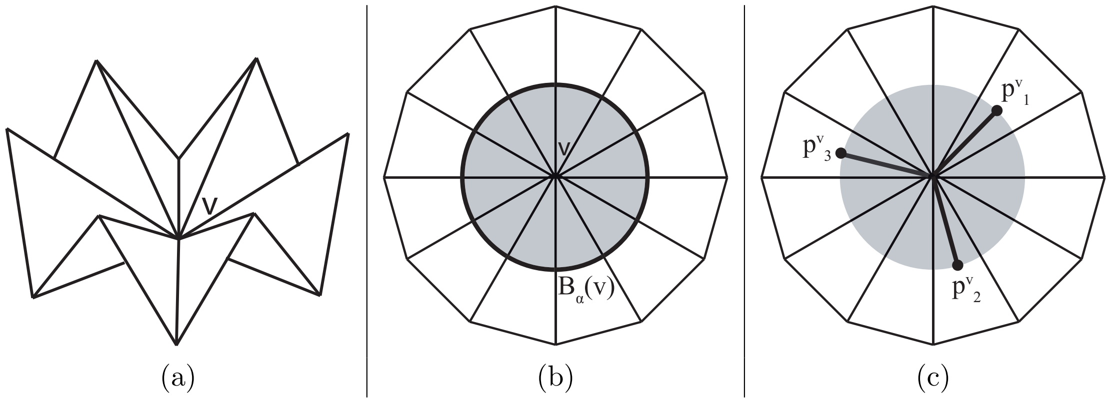


Figure 4.13: An example of the construction of points p_i^v in Theorem 4.5.1. In (a) we have a vertex v with a total face angle of, say, 5π. In (b) this vertex is shown from above, and $B_\alpha(v)$ is shaded. The split of $B_\alpha(v)$ into sectors, marked by points p_i^v on the boundary of $B_\alpha(v)$, is shown in (c). Since $\lceil \frac{5\pi}{2\pi} \rceil = 3$, this region is split into three sectors, each having total face angle $\frac{5\pi}{3} < 2\pi$.

4.5.2 Cutting Into Vertices

We now wish to consider a set of small cuts that separate a small neighbourhood around each vertex into components with equal face angle. Let us introduce some terminology. Given any $\epsilon > 0$, we say that the *closed ϵ-neighbourhood* of a vertex v, $B_\epsilon(v)$, is the set of points on the surface of P whose distance from v is at most ϵ. $B_\epsilon(v)$ will always be a closed region.

We now wish to define a value $\alpha > 0$ that depends on P. Take α to be any value small enough that no two neighbourhoods $B_\alpha(v)$ and $B_\alpha(w)$ intersect, where v and w are vertices of P. Also set α small enough that the only edges that intersect with $B_\alpha(v)$ are incident with v, for all vertices v of P. In other words, $B_\alpha(v)$ lies entirely upon faces incident with v.

Now, for each v, we define $w(v)$ points $p_1^v, \ldots, p_{w(v)}^v$ on the boundary of $B_\alpha(v)$. Let l_i^v be the line segment from v to p_i^v. Then l_i has length α for each i. Note that each l_i^v will lie on the surface of P, since the only edges of P that intersect with $B_\alpha(w)$ are incident with v. We place points p_i^v such that the face angle at v between l_i^v and l_{i+1}^v is $\frac{f(v)}{w(v)}$ for each i (and also between $l_{w(v)}^v$ and l_1^v).

It is certainly possible to find such points. First, place p_1^v arbitrarily on the boundary of $B_\alpha(v)$. Now travel around the boundary (in, say, the clockwise direction) until an angle of $\frac{f(v)}{w(v)}$ from l_1^v is reached; place p_2^v at that location. Continue in this way until all points are placed. Since the total face angle is $f(v)$ and there are $w(v)$ points to be placed, it will be possible to place all of our points in this fashion. See Figure 4.13 for an illustration.

4.5.3 Embedding of the Cut Tree

We are now ready to construct our cut tree. We wish to draw tree T on the surface of P. This tree will be drawn so that edges approach their endpoint vertices only along the lines l_k^v.

This problem is very similar to that of embedding a planar graph in the plane with fixed vertex locations. We will use a solution to that problem in order to construct our embedding of the tree T.

We construct a new tree T as follows. Suppose T contains edge $e = (v, w)$. Then T' will have edges (v, p_i^v), (p_i^v, p_j^w), and (w, p_j^w) where i and j are chosen so that p_j^w and p_i^v are unique to this edge e. This can certainly be done, since the number of points p_i^v for vertex v is equal to the degree of v in T. Note that T' is similar to T; we have simply fragmented the edges of T.

We now want to embed T' on the surface of P, with the restriction that the edges of the form (v, p_i^v) must be straight line segments. This restriction guarantees that the edges of T' split the face angle at each vertex v uniformly. This is very similar to the problem of embedding a tree in the plane with fixed vertex locations, when the embedding of certain edges is already given. See Figure 4.14 for an illustration of this process.

It is known that a solution to the problem of embedding a tree in the plane with predescribed vertex locations will always exist [28]. However, the surface of P is topologically equivalent to a sphere, not the plane. However, this problem is trivially fixed. We simply remove some point from the surface of P (which does not lie on any of the line segments already specified for T'). The resulting surface is equivalent to a plane, so we know an embedding C of tree T' must exist upon that surface. But this curve is also an embedding of T' on the surface of P, as required.

But this embedding of T' is also an embedding of T: we simply reinterpret each path of edges (v, p_i^v), (p_i^v, p_j^w), (w, p_j^w) as a single edge (v, w). We have therefore constructed our desired embedding of T.

Now our embedding of T is a cut tree such that the angle between two consecutive cuts to vertex v is precisely $\frac{f(v)}{w(v)}$. Recall $w(v) \geq \left\lceil \frac{f(v)}{2\pi} \right\rceil$. We conclude that the angle between any two cuts to v is no more than 2π. The unfolding that corresponds to our cut tree therefore has no unfolding angles greater than 2π.

This completes our proof of Theorem 4.5.1.

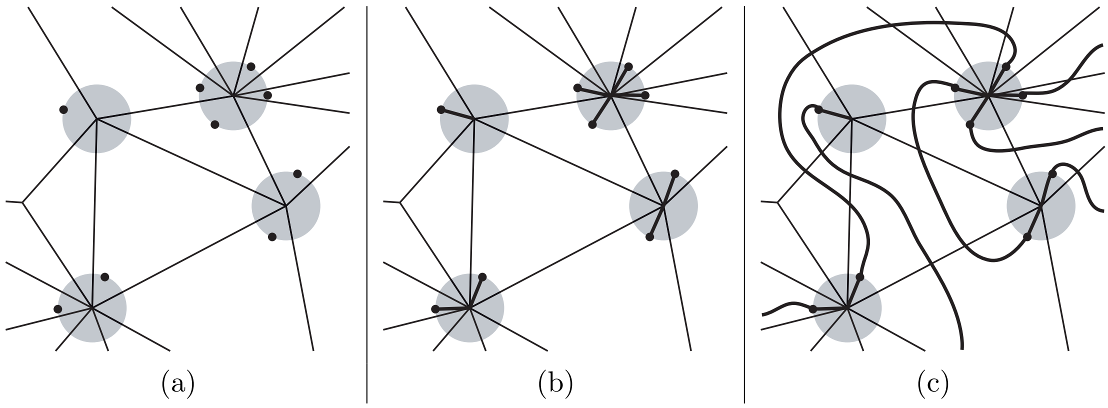


Figure 4.14: The construction of a tree embedding. (a) A portion of the surface of a polyhedron P with α-neighbourhoods shaded and points p_i^v shown. (b) The straight line segments l_i^v. (c) The additional curves connecting the straight line segments to complete the embedding of T'.

Chapter 5

Unfolding Convex Polyhedra

5.1 Introduction

In this chapter we continue to analyze local overlaps in polyhedron unfoldings. However, we now consider unfoldings of convex polyhedra.

This work is motivated by Shephard's conjecture, restated here as an open problem:

Problem 5.1.1. *Can every convex polyhedron be cut along its edges and unfolded into the plane such that the resulting surface does not self-intersect?*

In other words, the problem is to determine whether every convex polyhedron has a simple unfolding. It has long been believed that the answer is yes, but there have been distressingly few proven results on this topic. A thesis by Schlickenrieder [30] proposed a promising algorithm for unfolding convex polyhedra without overlap, but left open the task of proving whether or not it is successful in all cases. One of the main results of this chapter is that Schlickenrieder's algorithm is not successful for all convex polyhedra.

We shall examine this question from the point of view of local overlaps. Recall from the previous chapter that a k-local overlap is one in which the overlapping faces are connected by a path of at most k vertices on the boundary of the unfolding.

In Chapter 4 we analyzed 1-local overlaps in unfoldings of polyhedra. However, Corollary 4.2.3 states that no convex polyhedron contains a 1-local overlap. We shall therefore analyze 2-local overlaps in convex polyhedron unfoldings. In particular, we wish to characterize some conditions for cut trees that cause 2-local overlaps to occur. We shall then use these results to construct examples of convex polyhedra for which certain types of unfoldings will always contain overlaps. In particular, we shall construct a convex polyhedron for which cutting along any shortest path tree creates an unfolding with an overlap. We then construct a convex polyhedron for which

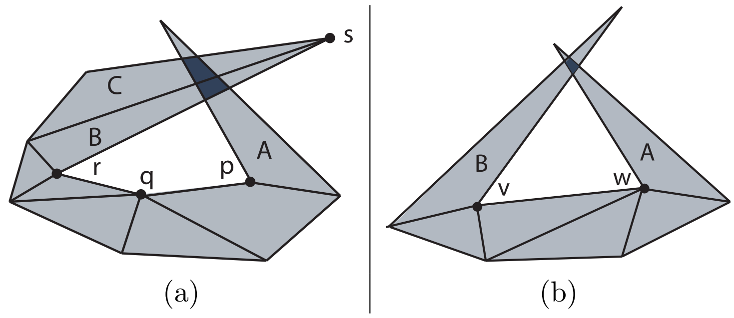


Figure 5.1: Examples of k-local overlaps for (a) $k = 3, 4$ and (b) $k = 2$.

Schlickenrieder's algorithm will always generate an overlap. Finally, we consider a more general class of unfoldings: the *normal order* unfoldings. Again using 2-local overlaps we shall show that there is a convex polyhedron for which every normal order unfolding contains an overlap.

5.2 2-Local Overlaps

We defined k-local overlaps in the previous chapter, but we shall repeat the definition here for completeness. Suppose a polyhedron unfolding has an overlap between two faces, f_1 and f_2. This overlap is called *k-local* if there are at most k vertices in the shortest path of the unfolding that starts with a vertex incident with f_1 and ends with a vertex incident to f_2. In particular, any overlap which is k-local is also r-local for any $r > k$. In Figure 5.1(a) the overlap between faces A and B is 3-local, corresponding to points p, q, and r. The overlap between faces A and C is 4-local, as it involves point s as well. Figure 5.1(b) shows an example of a 2-local overlap.

We shall now develop conditions for cut trees on convex polyhedra that will result in 2-local overlaps. We begin by providing a sufficient condition on unfoldings.

Lemma 5.2.1. *Suppose P' is an unfolding of a convex polyhedron. Let e_1, e_2, and e_3 be incident edges on the boundary of P', where e_1 and e_2 have common vertex v and e_2 and e_3 have common vertex w. Further suppose that $|e_3| = |e_2|$. Let ϕ be the exterior angle at v, and let θ be the exterior angle at w. If*

1. $\theta + 2\phi < \pi$, *and*

2. $|e_1| \geq |e_2| \frac{\sin\theta}{\sin(\pi-\theta-\phi)}$

then P' will contain a 2-local overlap.

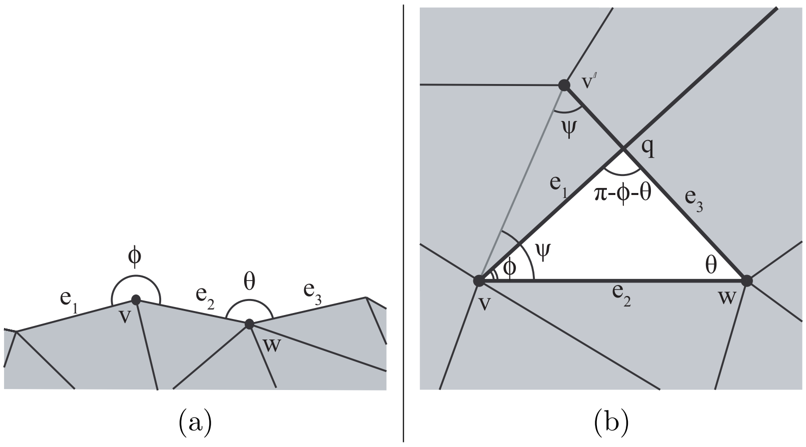


Figure 5.2: Unfoldings in Lemma 5.2.1. Shaded areas represent interiors of faces. (a) The configuration of edges, vertices, and angles in the statement. (b) A 2-local overlap, showing derivation of the edge length condition. Note that the line drawn from v to v' is not an edge; it is meant to illustrate angle ψ.

Proof. See Figure 5.2(a) for an illustration of the statement of this lemma.

Note first that $\theta \leq \pi$ and $\phi \leq \frac{\pi}{2}$ by the first condition in the claim.

Let v' be the vertex besides w incident with e_3. Now consider the isosceles triangle formed by v, v', and w. This triangle has angle θ at w, and angle $\psi := \frac{2\pi-\theta}{2}$ at v and v'. But we know that $\theta + 2\phi \leq \pi$, so $\phi \leq \frac{2\pi-\theta}{2} = \psi$. This means that v will be on the interior side of the line containing edge e_1. Thus edge e_1 will intersect e_3, assuming e_1 is sufficiently long.

We now determine the required length of e_1. Extend edge e_1 from v until it intersects e_3. Call that point of intersection q. Consider now the triangle formed by v, w, and q. The angle at q will be $\pi - \theta - \phi$. See Figure 5.1(c). Then, by the sine rule (and since $|e_2| = |e_3|$), we have that

$$\frac{|q - v|}{\sin \theta} = \frac{|e_3|}{\sin(\pi - \theta - \phi)}.$$

We conclude that e_1 will contain point q, and hence intersect e_3, if

$$|e_1| \geq |q - v| = |e_3| \frac{\sin \theta}{\sin(\pi - \theta - \phi)}$$

as required. □

We shall now present another condition that is sufficient to guarantee presence of a 2-local

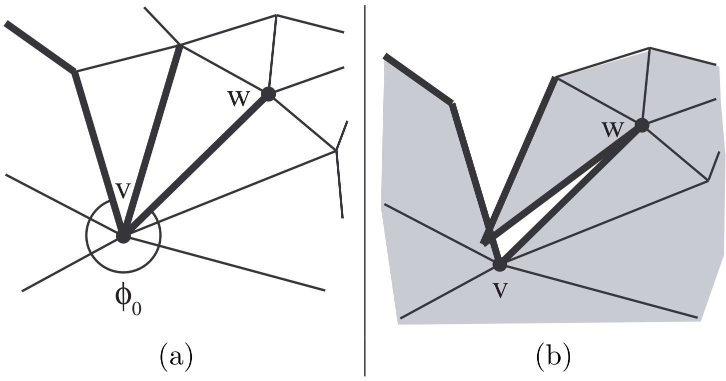


Figure 5.3: (a) A portion of the surface of a polyhedron, illustrating the conditions of Lemma 5.2.2. Cut edges are shown in bold and $\phi_0 > \frac{3\pi}{2}$. (b) The resulting 2-local overlap.

overlap. However, this condition applies to a polyhedron and cut tree directly, as opposed to Lemma 5.2.1 which applies to an unfolding.

Lemma 5.2.2. *Let P be a convex polyhedron with cut tree C. Suppose $w \in V(P)$ has degree 1 in C, and is adjacent to $v \in V(P)$ in C. Suppose further that there is an unfolding angle ϕ_0 at v bounded by (v, w) with $\phi_0 > \frac{3\pi}{2}$. Then there exists an angle θ_0 that depends on C and ϕ_0 such that the unfolding implied by C will contain a 2-local overlap if the curvature at w is less than θ_0.*

Proof. See Figure 5.3 for an illustration of the statement of this lemma. Let P' be the unfolding implied by C. Since w is incident with only one edge in C, it will have a single image in P', say w'. The exterior angle at w' will be precisely the curvature at w, say θ. The edge of C incident with w will have two images in P', say e_1 and e_2. They will satisfy $|e_1| = |e_2|$ and both will be incident with w'.

As w is incident with v, both e_1 and e_2 will be incident with images of v, say v_1 and v_2 respectively. The faces adjacent to v_1 and v_2 are precisely the unfolding groups of v bounded by (v, w). Thus, one of v_1 and v_2 (without loss of generality, v_1) will have total face angle greater than $\frac{3\pi}{2}$, and hence exterior angle less than $\frac{\pi}{2}$. Let ϕ be the exterior angle at v_1. Let e' be the edge incident to v_1 on the exterior of the unfolding, besides e_1.

Then if $0 < \theta < \pi - 2\phi$, we have that

$$\theta + 2\phi < \pi. \tag{5.1}$$

Note further that if $\sin\theta < \frac{|e'|}{|e_1|}\sin(\pi-\theta-\phi)$, then

$$|e'| > |e_1|\frac{\sin\theta}{\sin(\pi-\theta-\phi)}. \tag{5.2}$$

But as $\theta \to 0$, we have $\sin\theta \to 0$ and $\sin(\pi-\theta-\phi) \to \sin(\pi-\theta) > 0$. We conclude that there exists some $\theta_1 > 0$ such that $0 < \theta < \theta_1$ implies $\sin\theta < \frac{|e_1|}{|e'|}\sin(\pi-\theta-\phi)$.

Take $\theta_0 = \min\{\pi - 2\phi, \theta_1\}$. Then we conclude that if the curvature at w is less than θ_0 then both (5.1) and (5.2) hold. Thus the conditions of Lemma 5.2.1 are satisfied, so P' will indeed contain a 2-local overlap. □

5.3 Counterexamples to Unfolding Algorithms

Over the past decade, a number of conjectures have arisen proposing that some algorithm or another always generates simple unfoldings for convex polyhedra. All such conjectures have so far been false, but demonstrating that they are false is often quite difficult. There has been no particularly good way to construct examples of convex polyhedra that contain overlaps under a specified unfolding algorithm or class of unfoldings.

In this section we use Lemma 5.2.2 to construct examples of convex polyhedra. Each convex polyhedron will be such that every unfolding of a certain class will have an overlap. In the first three examples, the polyhedra will be presented to refute a particular unfolding algorithm. In the fourth example, a broad class of unfoldings not motivated by any particular algorithm will be discounted.

5.3.1 Shortest Path Trees

Given a polyhedron P and a vertex $v \in V(P)$, the *shortest path tree at v* is the tree formed by taking the union of the shortest paths from each vertex $w \in V(P)$ to v along the edges of P. It is well known that this subgraph is, indeed, a tree [18]; call it $SPT(v)$.

Counterexample for One Vertex

Fukuda made the following conjecture [18]:

Conjecture 5.3.1 (Fukuda)**.** *For every convex polyhedron P and every vertex $v \in V(P)$, the cut tree $SPT(v)$ forms a simple unfolding of P.*

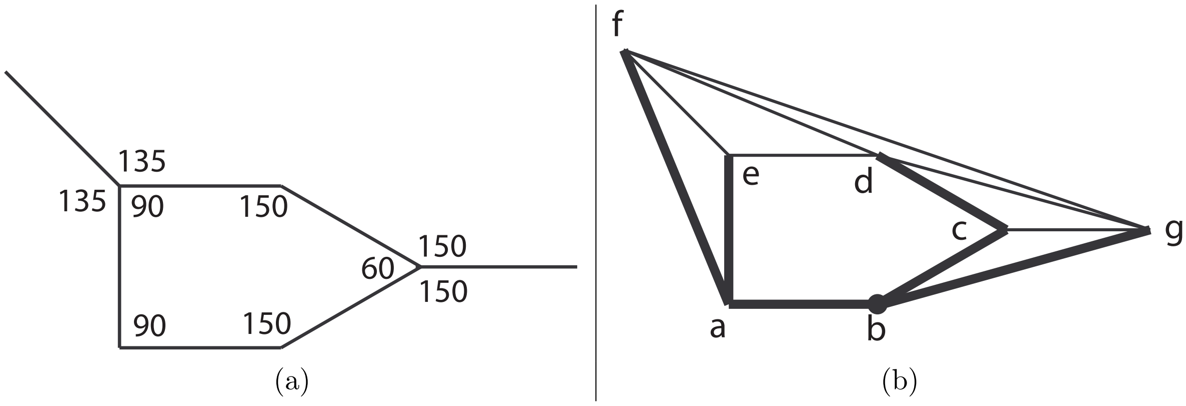


Figure 5.4: The planar figure used to disprove Conjecture 5.3.1. (a) The underlying structure. All line segments are of length 1 and angles are shown in degrees. (b) The completed figure. The bold line segments form $SPT(b)$.

This conjecture was resolved negatively by Schlickenrieder. However, Schlickenrieder's proof was empirical: he did not provide a formal construction of a counterexample. We fill this gap by constructing a polyhedron that disproves Conjecture 5.3.1.

A note before we begin: in the next section we shall show that a stronger conjecture, Conjecture 5.3.2, is false. This will imply that Conjecture 5.3.1 is false as well. In a sense, this makes the construction in this section redundant. However, the construction in this section is much simpler than those that will come later. We are therefore including this section to introduce gently the underlying ideas of our method of proof.

We shall construct a convex polyhedron that negatively resolves Conjecture 5.3.1. Consider the embedded graph shown in Figure 5.4(b). The tree $SPT(b)$ is illustrated in that figure. The important thing to note is the construction around vertices c and d. The example was created such that our tree includes edges (b, c) and (c, d) but not (c, g). The result is that faces (b, c, g) and (c, d, g) together form a component with angle greater than $270°$ at c. Further, vertex d has degree 1 in our tree. These properties are reminiscent of the requirements in Lemma 5.2.2.

We now wish to turn this graph into a convex polyhedron. We do this by first converting it into a convex terrain. Conceptually, we take vertices c, d, and e and raise them off the page by small distances, say α_c, α_d, and α_e. Note that α_d and α_e are determined by α_c, since face (a, b, c, d, e) must remain planar, so we need only specify α_c. Note that this construction works (i.e. it is possible for all faces to remain planar) because no face has more than two vertices upon the boundary of the graph. Finally, we add a single face (a, b, g, f) on the bottom of this terrain to complete a convex polyhedron $P(\alpha_c)$ that depends on α_c.

Note that as $\alpha_c \to 0$ we will have $\alpha_d \to 0$ and $\alpha_e \to 0$ as well. We can therefore choose α_c to be very small and the lengths of edges in the construction will be as close as we desire to the lengths in the original planar figure. In particular, if α_c is small enough, $SPT(b)$ for our polyhedron will be precisely the tree illustrated in Figure 5.4(b).

But now consider the properties of $SPT(b)$ as a cut tree upon our polyhedron. Faces (b, c, g) and (c, d, g) together form an unfolding group with angle greater than 270° at c (as long as α_c is small enough that the total face angle at c is sufficiently close to 360°). Also, vertex d has degree 1 in the cut tree. All that remains to satisfy the requirements of Lemma 5.2.2 is for the curvatures at vertex d to be sufficiently small. But recall that we can choose α_c to be as small as desired. We can therefore choose α_c to be small enough that the curvature at d is as small as required by Lemma 5.2.2. But then Lemma 5.2.2 implies that cutting along $SPT(b)$ will create an unfolding that contains a 2-local overlap.

We conclude that there exists a value of α_c such that the convex polyhedron $P(\alpha_c)$ is a counterexample to Conjecture 5.3.1.

Counterexample for All Vertices

In the previous section we showed that Conjecture 5.3.1 is false. Now consider the following conjecture:

Conjecture 5.3.2. *For every convex polyhedron P there exists a vertex v such that $SPT(v)$ forms a simple unfolding.*

This conjecture is stronger than Conjecture 5.3.1 because it allows a choice among the vertices for the root of the shortest path tree. In this section we construct a convex polyhedron that demonstrates for the first time that Conjecture 5.3.2 is false.

The idea of the construction is to create a convex terrain very similar to the one presented in Section 5.3.1. However, the tree $SPT(v)$ must satisfy the properties of Lemma 5.2.2 no matter which vertex v is chosen. If we were to use the construction of Figure 5.4(b) and v were chosen to be one of the interior vertices (say c in Figure 5.4(b)), our tree would not turn out as desired. To get around this, we use (a variant of) the previous construction as a widget, and we will use multiple copies of it in our new construction. Thus, no matter where our vertex v is chosen, there will be some copy of the widget that will unfold in a manner to create a 2-local unfolding.

The planar graph we use as a starting point is illustrated in Figure 5.5. Note that the graph in Figure 5.5(a) is similar to the construction of Section 5.3.1. Our final planar figure in Figure 5.5(b) has four of these constructions as pieces.

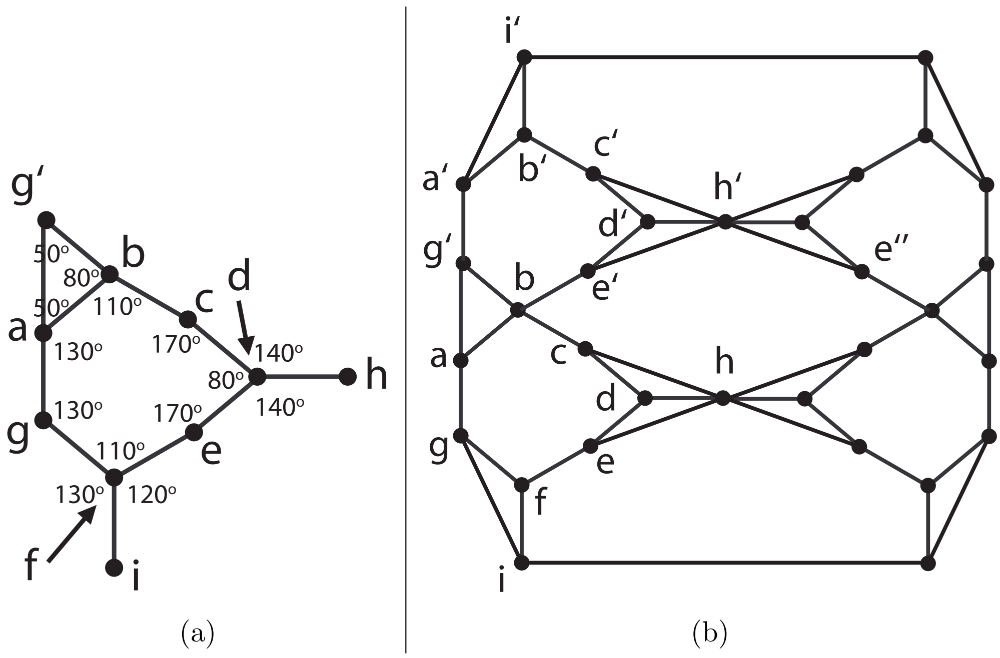


Figure 5.5: The construction of our terrain used to disprove Conjecture 5.3.2. (a) The main component. All line segments have length 1, except (a, g') which is longer than 1. (b) The completed terrain.

Now consider turning Figure 5.5(b) into a convex terrain by raising every internal vertex above the page so that all faces remain planar and no vertex is raised more than some parameter α. Turn this terrain into a polyhedron by adding a single face on the bottom, incident to all edges on the boundary of the terrain. Let $P(\alpha)$ be such a convex polyhedron.

We now show that $P(\alpha)$ forms a counterexample to Conjecture 5.3.2.

Theorem 5.3.3. *There is a value of α such that every shortest path tree of $P(\alpha)$ induces an unfolding that contains a 2-local overlap.*

Proof. We shall show that the result is true for sufficiently small $\alpha > 0$.

Let v be any vertex of $P(\alpha)$. Let $L = \{a, b, c, d, e, f, g, h, i\}$ be the lower-left vertices in Figure 5.5(b). By symmetry, we can assume that $v \in L$. We now consider the nature of $SPT(v)$ in the upper-left portion of the graph.

Claim 5.3.4. *The shortest path from any $w \notin L$ to $v \in L$ is the union of the shortest path from w to one of a, b, h, or i, and the shortest path from that vertex to v.*

Proof. Any path from w to v must pass through one of a, b, h, or i. If the path has minimum length then its subpaths must have minimum length as well, so the result follows. □

Claim 5.3.5. *$SPT(v)$ must contain edges (c', d') and (d', e').*

Proof. By Claim 5.3.4, the shortest path from c' to v must contain the shortest path from c' to one of a, b, h, or i. But each of these paths contains edges (c', d') and (d', e'). We conclude that (c', d') and (d', e') must be in $SPT(v)$. □

Claim 5.3.6. *$SPT(v)$ will not contain edges (b', c') and (c', h').*

Proof. Take some vertex $w \notin L$. Consider the shortest path from w to v. We can think of this path as a sequence of directed moves from one vertex to another, starting at w and ending at v. By Claim 5.3.4, this path consists of the shortest path from w to one of a, b, h, or i, followed by the shortest path from that vertex to v.

Further suppose for contradiction that our shortest path contains a move from b' to c'. It follows that the shortest path from b' to one of a, b, h, or i begins with a move to c'. However, this is not true: the shortest path from b' to any of a, b, h, or i begins with a move to a'.

Suppose instead that our shortest path contains a move from c' to b'. Then the shortest path from c' to one of a, b, h, or i must begin with a move to b'. Again, this is not true: the shortest path from c' to any of a, b, h, or i begins with a move to d' (since the edge (a, g') has length greater than 1).

We conclude that the path from w to v does not contain a move from b' to c' or from c' to b'. Therefore edge (b', c') is not included in $SPT(v)$.

A similar argument shows that $SPT(v)$ does not contain (c', h'). We have already shown that the shortest path from c' to any of a, b, h, or i does not begin with a move to h'. On the other hand, shortest paths from h' to a, b, or i begin with a move to e', and the shortest path from h' to h begins with a move to either e' or e'' (as both directions lead to paths of equal length).

Thus $SPT(v)$ does not contain (b', c') or (c', h'), as required. □

Now choose α small enough that $P(\alpha)$ has the same shortest path trees as the planar graph of Figure 5.5(b). Then Claim 5.3.5 and Claim 5.3.6 imply that if we take $SPT(v)$ as a cut tree, we will cut (c', d') and (d', e') but not (b', c') or (c', h'). Thus c' has degree 1 in the cut tree, and faces (c', d', h') and (d', e', h') form an unfolding group incident with (c', d') with unfolding angle greater than 270°.

Lemma 5.2.2 now implies that a 2-local overlap will occur, as long as the curvatures at c' and d' are sufficiently small. But these curvatures are determined by α, which we can make arbitrarily small. We therefore choose α small enough that our curvatures meet the requirements of Lemma 5.2.2. Then $SPT(v)$ will contain a 2-local overlap, as required. □

5.3.2 The Steepest Edge Algorithm

Schlickenrieder performed empirical tests to compare different types of unfolding algorithms [30]. He put forth a conjecture regarding a particular directional algorithm: the Steepest Edge algorithm. In this section we shall describe this algorithm, present Schlickenrieder's conjecture, and prove that it is false.

Algorithm Description

The Steepest Edge algorithm proceeds as follows. Take as input a convex polyhedron P. Pick a unit direction vector ζ. We shall informally refer to the direction of ζ as "up". Without loss of generality we can assume that $\zeta = (0, 0, 1)$ (by reorienting space). Let v_+ be the vertex of the polyhedron with maximum z-coordinate.

Now for each vertex v in $V(P) - \{v_+\}$, and for each edge (v,w), consider the unit vector $d(v, w) := \frac{w-v}{|v,w|}$. That is, $d(v, w)$ is the unit vector that follows edge (v, w) beginning at v. We shall say that the *steepest edge at* v is the edge (v, w') for which the z-coordinate of $d(v, w')$ is maximal. The Steepest Cut algorithm chooses the cut tree C to be the set of steepest edges for each vertex in $V(P) - \{v_+\}$. Heuristically, we are cutting "the most upward" that we can at each vertex. Schlickenrieder shows that C is indeed a cut tree [30].

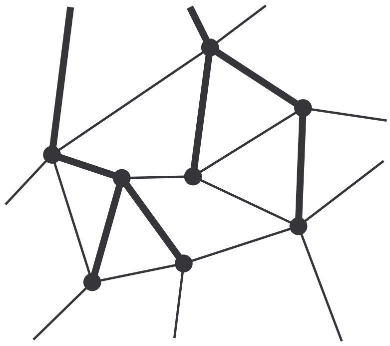


Figure 5.6: Illustration of steepest edges, assuming ζ faces the top of the page.

See Figure 5.6 for an illustration; this figure illustrates a portion of the surface of a polyhedron, where ζ points to the top of the page and curvatures are assumed to be small. The steepest edge for each vertex in the illustration is drawn in bold.

In his empirical tests, Schlickenrieder found that this algorithm generated a simple unfolding for roughly 93% of all tested polyhedra when ζ was chosen at random [30]. For those that had an overlap, the algorithm would simply choose another vector for ζ and try again. In this way, a simple overlap was found for all tested polyhedra within a few iterations.

Based on these results, Schlickenrieder put forth the following conjecture, reworded into our notation.

Conjecture 5.3.7 (Schlickenrieder)**.** *For every convex polyhedron P there exists some unit vector ζ such that the Steepest Edge algorithm with direction ζ generates a simple unfolding.*

Unfortunately, this promising conjecture is false, as we shall now show.

5.3.3 Counterexample

We shall disprove Conjecture 5.3.7 by constructing a counterexample. That is, we construct a polyhedron for which the Steepest Edge algorithm generates an unfolding with a 2-local overlap for every possible direction vector ζ.

The counterexample to Conjecture 5.3.7 is considerably more involved than the previous two. The main difficulty is that the algorithm in Conjecture 5.3.7 allows choice of a continuous parameter (a direction).

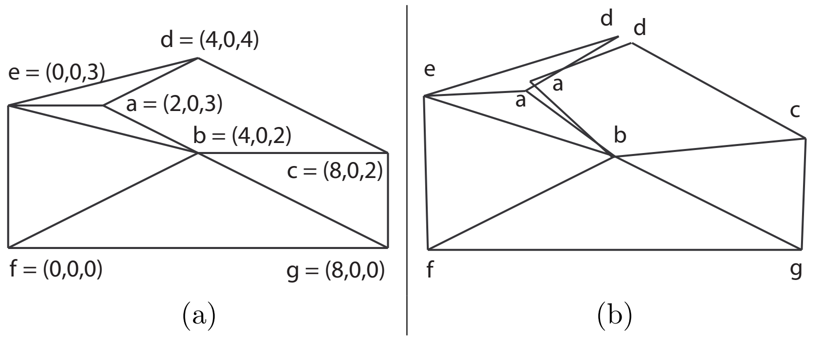


Figure 5.7: (a) The planar graph M_1^*, with $(0, 0, 1)$ directed toward the top of the page. (b) The steepest edge unfolding of $M_1(\alpha)$ for small α and direction vector $\zeta = (0, 0, 1)$.

Outline

We begin by constructing a convex terrain for which the Steepest Edge algorithm does not work for a particular choice of ζ, say ζ_0. That is, if the Steepest Edge algorithm is applied to our terrain with $\zeta = \zeta_0$, an overlap will occur. We further show that our ζ need not be precisely ζ_0; it is sufficient for ζ to be within some small angle ϕ of ζ_0. Furthermore, this ϕ is independent of scaling, translation, and rotation of the terrain (and of the corresponding vector ζ_0). Finally, we shall construct a polyhedron by gluing together many copies of this terrain in various orientations. The result will be that every possible choice of ζ will be within ζ_0 for some copy of the terrain, and hence that terrain will form an overlap. Thus, no matter what ζ is chosen, the resulting unfolding of the polyhedron will contain an overlap.

The Terrain

Consider the planar graph M_1^* illustrated in Figure 5.7(a). We are taking $\zeta_0 = (0, 0, 1)$ in this illustration. This graph is thought of as lying in the xz-plane, with the positive z-axis facing the top of the page and the positive y-axis directed out of the page. One thing to note is that the angle $\angle dab$ is less than $\frac{\pi}{2}$. We can convert this graph into a convex terrain by raising the interior vertices a and b. In particular, given parameter $\alpha > 0$, we denote by $M_1(\alpha)$ the convex terrain formed by raising the vertices a and b to a height of α. Thus the coordinates of a and b will be $(2, \alpha, 3)$ and $(4, \alpha, 2)$, whilst the remaining vertices have y-coordinate 0. Note that the quadrilateral a, b, c, d remains planar under this modification, since the edges (a, b) and (c, d) are parallel. Also, as $\alpha \to 0$, the curvatures at a and b become arbitrarily small.

Lemma 5.3.8. *Suppose that $M_1(\alpha)$ forms part of a polyhedron P. Then there exists an $\alpha_0 > 0$ and $\phi_0 > 0$ such that when the Steepest Cut algorithm is applied to P with ζ within an angle of ϕ from $(0, 0, 1)$, the corresponding unfolding of P will contain a 2-local overlap whenever $\alpha \leq \alpha_0$ and $\phi \leq \phi_0$. Further, our choice of ϕ_0 is independent of any scaling operations performed upon $M_1(\alpha)$. That is, ϕ_0 does not depend on the size of $M_1(\alpha)$.*

Proof. First suppose that we take $\zeta = (0, 0, 1)$. Choose α small enough so that the relative steepness of edges in $M_1(\alpha)$ is identical to that in M_1^*.

Now $M_1(\alpha)$ is embedded in a polyhedron P. This means that the external vertices of $M_1(\alpha)$ (i.e., all but a and b) may have additional incident edges in P. Vertices a and b, however, will have no other additional incident edges. Thus we can determine which edges will be steepest from vertices a and b.

Note that, of all edges adjacent to b, edge (a, b) is steepest. This edge will therefore be cut. Similarly, edge (a, d) is the steepest from vertex a, so it will be cut.

For the remaining vertices we cannot determine which incident edge will be steepest, since steeper edges may be added when $M_1(\alpha)$ is embedded in P. However, consider the edges (b, c), (b, g), (b, f), (b, e), and (a, e). For each vertex of each of these edges, there is another incident edge that is steeper. Thus, none of these edges will be cut by the Steepest Cut algorithm, regardless of what additional edges may be added.

We conclude that b has degree 1 in C. Recall that the angle $\angle dab$ in M_1^* is less than $\frac{\pi}{2}$. Thus, if the curvature at a is small enough in $M_1(\alpha)$, the face angles at a of faces (a, d, e) and (a, b, e) will sum to more than $\frac{3\pi}{2}$. That is, the unfolding angle at a bounded by (a, b) and (a, d) is greater than $\frac{3\pi}{2}$ for sufficiently small α. Thus, by Lemma 5.2.2, a 2-local overlap occurs when we unfold if the curvature at a and at b is small enough (and hence if α is small enough). Hence a 2-local overlap occurs when α is small enough to satisfy all of the above conditions. We take α_0 to be some such small value; so any $0 < \alpha < \alpha_0$ will be sufficient. See Figure 5.7(b).

Note that, given a fixed $\alpha < \alpha_0$, our overlap was caused only by the cutting of (a, b) and (a, d), and the fact that (b, c), (b, g), (b, f), (b, e), and (a, e) were not cut. Thus, for any direction vector ζ such that these cuts and non-cuts occur, the resulting unfolding will contain an overlap. Now note that if ζ_0 is perturbed by a small enough amount in any direction, this same pattern of cuts and non-cuts will occur. This implies that there is some open range $D(\zeta_0)$ of choices for direction vectors on the unit sphere that will cause an overlap. We can therefore find an angle ϕ_0 such that if the angle between ζ and ζ_0 is within ϕ_0 then the illustrated 2-local overlap will occur when ζ is chosen as the direction in the Steepest Edge algorithm. Since $D(\zeta_0)$, and hence ϕ_0, depends only on the orientations of edges and not upon their lengths, the value ϕ_0 is independent of the size of $M_1(\alpha)$. □

We shall now take M_1 to mean $M_1(\alpha_0)$, where α_0 is as defined in the statement of Lemma 5.3.8.

Our eventual goal is to glue many copies of M_1 together into a polyhedron. In order to make this task easier, we shall first embed M_1 into a triangle. Our terrain will then have a triangular border and the gluing will be easier to visualize.

We embed M_1 into an isosceles triangle, as shown in Figure 5.8(a). Note that the edge (q, r) is shorter than edges (p, q) and (p, r). Lower the vertices of the triangle by α_0, so they occur with a y-coordinate of $-\alpha_0$. Call the resulting convex terrain M_2.

Lemma 5.3.9. *Suppose that M_2 is embedded in a convex polyhedron P and the Steepest Cut algorithm is applied to P with ζ within an angle of ϕ_0 from $(0, 0, 1)$. Then the corresponding unfolding of M_2 will contain a 2-local overlap.*

Proof. Such an embedding can be thought of as an embedding of M_1 in polyhedron P, since M_1 is embedded in M_2. The result then follows from Lemma 5.3.8. □

Transforming the Terrain

We now wish to consider instances of M_2 in different orientations. Informally, we simply rotate M_2 in three dimensions and apply the same change of orientation on direction vector ζ. Then the rotated instance of M_2 will unfold in the same way when the Steepest Cut algorithm is applied with the rotated direction vector.

More formally, suppose we have an instance of a triangle $T = (p, q, r)$ in $\mathbf{R}^3$, similar to the bounding triangle in M_2. This can be thought of as a 3-dimensional rotation (plus translation and uniform scale) of the embedding triangle in M_2. Let $c(T)$ be the unit vector that results from applying this rotation to $(0, 0, 1)$. Note that this is well-defined, since (as T is isosceles) one can derive the necessary rotation from the vertices of T.

Now suppose a copy of M_2 is embedded in T. Then the resulting convex terrain would unfold with a 2-local overlap if the Steepest Edge algorithm were applied with direction vector within angle ϕ_0 of $c(T)$. Here ϕ_0 is chosen as in the statement of Lemma 5.3.8. Recall that ϕ_0 is independent of the orientation and size of M_2, so indeed the same value ϕ_0 applies no matter how we transform our terrain M_2, and hence no matter what triangle T is chosen.

The Desired Polyhedron

First some definitions. Let S be the surface of the unit sphere. Choose ϕ_0 as in the statement of Lemma 5.3.8. Given unit vector ζ, let $D(\zeta)$ be the set of all unit vectors within angle ϕ_0 of ζ. Then $D(\zeta)$ can be thought of as an open disc lying upon S.

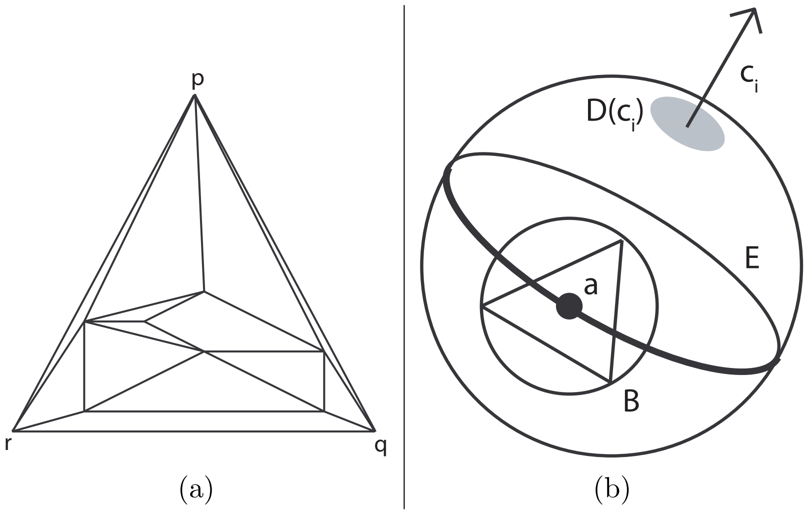


Figure 5.8: Illustrations for Lemma 5.3.9. (a) Embedding of $M_1(\alpha)$ into a triangle. (b) Placing triangle i on the unit sphere.

We are now ready to construct our polyhedron. The idea of the construction is to glue together many copies of M_2 so that, for their corresponding direction vectors $\zeta_1, \ldots, \zeta_k$, the $D(\zeta_i)$ cover all of S. Thus, for any choice of direction vector ζ, ζ will fall within some $D(\zeta_i)$, and hence the i^{th} copy of M_2 will have an overlap when we unfold, by Lemma 5.3.9.

We begin by showing that, given any finite set of direction vectors, we can glue together the *boundaries* of many copies of M_2 to match those direction vectors. We will end up with a spherical polyhedron; that is, a polyhedron with all vertices lying upon the surface of a sphere.

Lemma 5.3.10. *Suppose $k \geq 0$, and $\{c_i\}_{i=1}^k$ is a set of unit vectors. Then there exists a spherical polyhedron P with a subset of faces $\{f_i\}_{i=1}^k \subset F(P)$ such that each f_i is similar to the bounding triangle in $M_2(\alpha)$ and $c(f_i) = c_i$ for all i.*

Proof. We shall proceed by induction on k, proving the result with the additional constraint that each f_i has a diameter of at most $\frac{1}{2^i}$. Recall that the diameter of a polygon is the maximum distance between any two points upon it. This proof will also make use of convex hull. Given a set A of polygons in space, we shall take $CH(A)$ to mean the convex hull of all vertices of the polygons.

First consider $k = 0$. Then the regular tetrahedron, with vertices lying upon the unit sphere, trivially satisfies the conditions of the lemma.

Next consider $k = 1$. Place vertices p, q, and r on the unit sphere such that the resulting triangle T is similar to that in $M_2(\alpha)$, and rotate so that $c(T) = c_1$. Add any other vertex s on the surface of the sphere, so that s is not coplanar with T. Then the convex hull of p, q, r, and s is a convex polyhedron satisfying the conditions of the lemma.

Now suppose $k > 1$, and the result is true for $n = k-1$. Let P' be a spherical polyhedron that satisfies the conditions of the lemma for vectors $\{c_i\}_{i=1}^{k-1}$. Take the set of faces $A_{k-1} := \{f_i\}_{i=1}^{k-1}$ of P' corresponding to $\{c_i\}_{i=1}^{k-1}$. Then since A_{k-1} is a set of faces of a polyhedron, $CH(A_{k-1})$ contains all the faces in $f_1, \ldots, f_{k-1}$. We now wish to add vertices p, q, and r to form face f_k, such that if $A_k = A_{k-1} \cup \{f_k\}$, we shall have that the $CH(A_k)$ contains all faces $f_1, \ldots, f_k$. This is equivalent to having the spherical region covered by f_k not intersect the spherical regions covered by each f_i.

Well, let E be the equator of S that lies on a plane perpendicular to c_k. We wish to find a point $a \in E$ such that a disc lying on S centered at a of diameter less than $\frac{1}{2^k}$ does not intersect any faces f_i for $1 \leq i < k$. But such a point certainly exists; E has length 2π, and each face f_i has diameter at most $\frac{1}{2^i}$. Thus the total length of E that could be covered by faces f_i is at most

$$\sum_{i=1}^{k-1} \frac{1}{2^i} < 1 < 2\pi.$$

There is therefore a point a upon E that is exterior to every face f_i. See Figure 5.8(b).

Let B be a disc centered at a that is small enough not to intersect any f_i, and has diameter less than $\frac{1}{2^i}$. Note that such a disc exists, since the set of points of S exterior to all f_i is an open region. Now place points p, q, r upon S, within B, such that they form a triangle T that is similar to the bounding rectangle of $M_2(\alpha)$, with $c(T_k) = c_k$. Such a triangle exists: we can make this triangle arbitrarily small, and vector c_k is tangent to S at a, so we simply take the plane of T to be parallel to the tangent plane at a.

Take $f_k = T$. Then f_k lies in a region of S that does not intersect any other faces f_i, so the convex hull of $f_1, \ldots, f_k$ contains faces $f_1, \ldots, f_k$. In addition, $c(f_k) = c_k$ and f_k has vertices upon S, as required. We therefore take our polyhedron to be the convex hull of $f_1, \ldots, f_k$ and the conditions of the lemma are satisfied.

□

We are now nearly done. We need only define our set of direction vectors such that all possible choices for the Steepest Edge algorithm are covered, build the polyhedron from Lemma 5.3.10, and paste copies of M_2 upon the boundaries given in that polyhedron.

Theorem 5.3.11. *There exists a polyhedron P such that the Steepest Edge algorithm generates an unfolding with a 2-local overlap for any choice of direction vector ζ.*

Proof. Choose any set of direction vectors $\{\zeta_i\}_{i=1}^k$ such that $\cup_{i=1}^k D(\zeta_i) = S$. Such a finite set exists, since S is a compact set. By Lemma 5.3.10 there exists a spherical polyhedron P_0 containing triangular faces $f_1, \ldots, f_k$, each similar to the bounding rectangle of $M_2(\alpha)$, such that $c(f_i) = \zeta_i$ for each i. Form polyhedron P_1 by replacing each f_i in P_0 with a copy of $M_2(\alpha)$. Here we take α small enough that Lemma 5.3.9 applies (recall that M_2 in Lemma 5.3.9 depends on α), and also small enough that the polyhedron remains convex. That is, $\alpha < \alpha_0$ and α is small enough that the angles between the faces of $M_2(\alpha)$ and the plane are negligible compared to the face angles of P_0.

But now for any $\zeta \in S$, ζ will lie in some $D(\zeta_i)$. Thus, by Lemma 5.3.9, the embedding of $M_2(\alpha)$ in face f_i will unfold to create a 2-local overlap when the Steepest Edge algorithm is applied with direction ζ. Thus, for any direction vector ζ, the steepest edge algorithm will unfold polyhedron P_1 to generate a 2-local overlap, as required. $\square$

5.3.4 Normal Order Unfoldings

We now give a final example of the use of 2-local overlaps to analyze a class of unfoldings. We shall define a class of unfoldings: the *Normal Order* unfoldings. At the outset of our research, we were of the opinion that every convex polyhedron has a simple normal order unfolding. However, this is not the case: we shall prove that there exists a polyhedron P such that every normal order unfolding of P contains an overlap.

Definition

Let P be a convex polyhedron. Choose a direction vector ζ. Reorient space so that $\zeta = (0, 0, 1)$. Choose any $f \in F(P)$, and let n be the outward-facing unit normal for f. Denote by $z(f)$ the z-coordinate of n. We say $z(f)$ is the *height* of f.

Now consider a cut tree C of P, with corresponding adjacency tree A and unfolding P'. We say that the unfolding P' is a *normal order* unfolding if, for all $\delta \in [-1, 1]$, the set $\{f \in F(P) : z(f) \leq \delta\}$ is connected in A. We also say in this case that A is in normal order.

Informally, a normal order unfolding is constructed by first choosing a face f in P with minimum $z(f)$, then attaching faces to the adjacency tree one by one in ascending order by height. Note that there may be many normal order unfoldings for a polyhedron P and fixed direction vector ζ; a given face may have many lower faces to which it may be attached in A, and each choice leads to a different normal order unfolding. See Figure 5.9.

Lemma 5.3.12. *Suppose adjacency tree A of convex polyhedron P is in normal order. Then, for every face $f \in F(P)$, either*

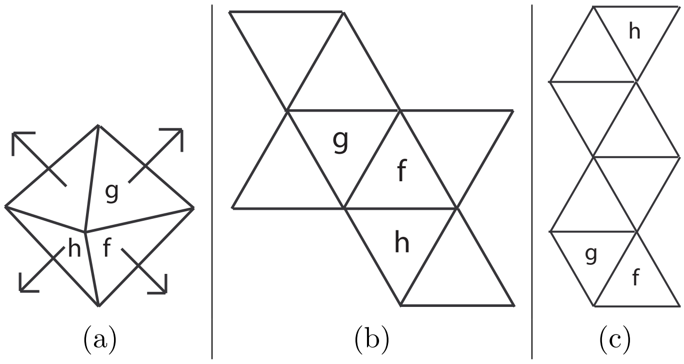


Figure 5.9: An example of normal order unfolding. (a) A tetrahedron with face normals, oriented with $(0, 0, 1)$ facing the top of the page. (b) A normal order unfolding of the tetrahedron, rooted at face f. (c) An unfolding that is not in normal order with respect to direction $(0, 0, 1)$, as one of f or h will violate Lemma 5.3.12.

1. *f is the unique face with minimal $z(f)$, or*
2. *f is adjacent in A to another face g with $z(g) \leq z(f)$.*

Proof. Suppose that A is in normal order but $f_0 \in F(P)$ is not adjacent in A to any face g with $z(g) \leq z(f_0)$. We shall show that f_0 must be the unique face in $F(P)$ with minimal $z(f_0)$.

Suppose that there is some $f_1 \in F(P)$ with $z(f_1) \leq z(f_0)$. Take the set $S = \{f \in F(P) : z(f) \leq z(f_0)\}$. Then f_0 is not adjacent to any other face in S. But $f_1 \in S$, $f_1 \neq f_0$. We conclude that S is not connected in A, contradicting the normal order of A.

Thus $z(f_0) > z(f)$ for all $f \in F(P)$, and hence f_0 is the unique face that minimizes $z(f_0)$ as required. $\square$

A Polyhedron With No Simple Normal Order Unfolding

We shall now construct a polyhedron for which every normal order unfolding contains an overlap. These overlaps will not necessarily be 2-local, but every normal order unfolding that does not contain a 2-local overlap contains a 3-local overlap.

The method of construction is very similar to that for the counterexample to Conjecture 5.3.7. In particular, it is based upon the idea of forming a construct of faces that will form a k-local overlap under a given orientation, then forming a polyhedron built with many instances of it.

Consider the planar graph M_3 illustrated in Figure 5.10(a). Here the positive z-axis is thought of as pointing toward the top of the page, and the positive y-axis points out of the page. The

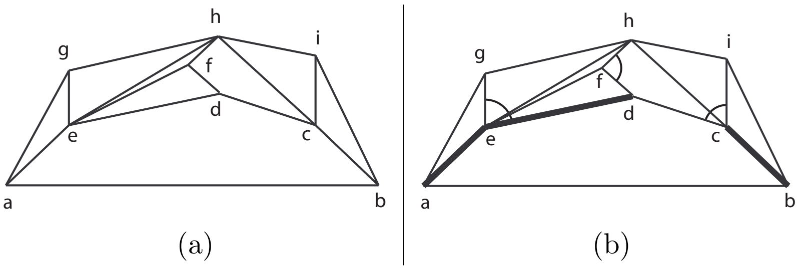


Figure 5.10: The planar graph M_3. In (b) the edges that will not be cut in a normal order unfolding are marked, as are notable angles that are less than $\frac{\pi}{2}$.

important thing to notice about this graph is that the angles $\angle icd$, $\angle dfh$, and $\angle deg$ are all less than $\frac{\pi}{2}$, see Figure 5.10(b).

Consider raising the interior vertices of M_3 in such a way that each face of M_3 remains planar. More specifically, take some $\delta > 0$ and raise point d so that its y-coordinate is δ. Now raise vertices e and c so that the polygon (a, b, c, d, e) remains planar. Finally, raise vertex f so that the polygon (c, d, f, h) remains planar.

Suppose the largest y-coordinate of any vertex raised in this way is α; then we call the resulting convex terrain $M_3(\alpha)$. Note that $\alpha > 0$, and that as our choice of δ approaches 0, so does α. Hence α can be made arbitrarily small, and thus the curvature at all interior vertices of $M_3(\alpha)$ can be made arbitrarily small by taking α to be arbitrarily close to 0.

Lemma 5.3.13. *Suppose ζ is a unit vector that has angle at most ϕ from $(0, 0, 1)$. Then any normal order unfolding of $M_3(\alpha)$ with respect to ζ contains an overlap, when α and ϕ are sufficiently small.*

Proof. First suppose that $\zeta = (0, 0, 1)$. Note that for any adjacent faces f_1 and f_2 in $M_3(\alpha)$ with common edge e, if f_1 lies above e in Figure 5.10(a) then we will have $z(f_1) > z(f_2)$.

Now consider Figure 5.10(b). In this illustration of $M_3(\alpha)$, the bold edges will not be cut in a normal order unfolding. These are the situations in which a face is incident to only one lower face, and thus must be adjacent to that face in any normal order unfolding by Lemma 5.3.12.

Let F denote the face (c, d, f, h). Note that there is a choice regarding edges incident with F to cut. In a Normal Order unfolding, one of edges (d, f) or (c, d) must not be cut.

Case 1: edge (c, d) is cut. Then a portion of the unfolding is as illustrated in Figure 5.11(a). Recall that angle $\angle icd$ is less than $\frac{\pi}{2}$ in M_3. Thus, if α is sufficiently small, the sum of the angles of faces (i, c, b) and (a, b, c, d, e) at vertex c will be greater than $\frac{3\pi}{2}$. But then, by Lemma 5.2.2,

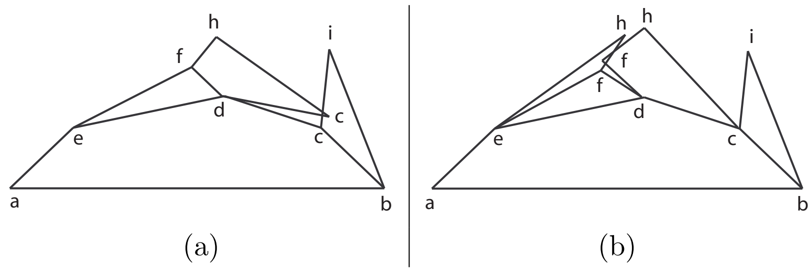


Figure 5.11: The 2-local overlaps that occur in normal order unfoldings of $M_3(\alpha)$

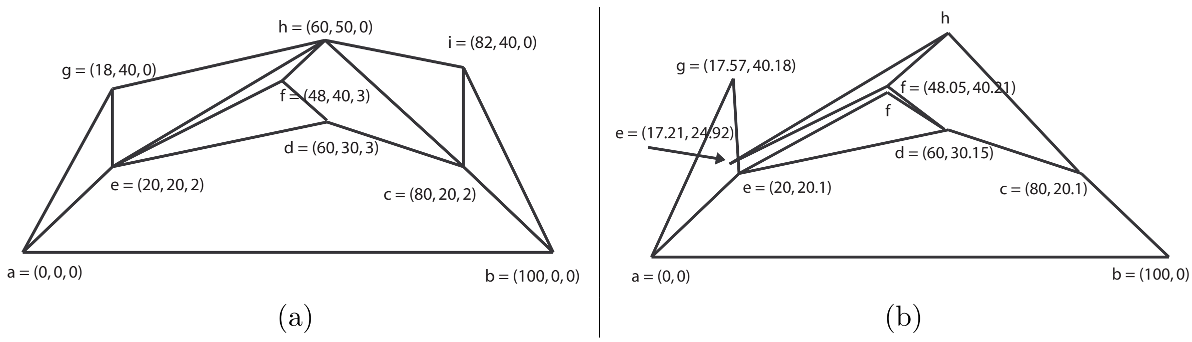


Figure 5.12: (a) A particular instance of $M_3(\alpha)$. (b) The 3-local overlap that can occur in a Normal Order unfolding of $M_3(\alpha)$, using the embedding from (a).

a 2-local overlap will occur in this unfolding of $M_3(\alpha)$ (again for sufficiently small α). See Figure 5.11(a) for an illustration.

Case 2: edge (d, f) **is cut.** We now have two subcases.

Case 2.1: edge (f, h) **is cut.** Then our situation is similar to that discussed in Lemma 5.3.9. That is, the angles at f in faces (d, e, f) and (e, f, h) sum to more than $\frac{3\pi}{2}$ when α is sufficiently small. So, by Lemma 5.2.2, a 2-local overlap will occur in this unfolding when α is sufficiently small. See Figure 5.11(b).

Case 2.2: edge (f, h) **is not cut.** Then edge (f, e) must be cut. But then, taking curvatures sufficiently small, there will be an overlap between faces (a, e, g) and (e, f, h). See Figure 5.12.

This situation requires particular attention, since the occurrence of an overlap does not follow immediately from Lemma 5.3.9. In particular, a 2-local overlap does not occur. We therefore proceed by explicitly determining the coordinates of the vertices in the unfolding to demonstrate that an overlap occurs. See Figure 5.12 for this information. The result is a 3-local overlap between faces (a, e, g) and (e, f, h).

We conclude that there is no way to unfold terrain M while respecting the normal order induced by ζ. Note that no edges in M_3 are parallel to the z-axis. Thus we do not have any boundary conditions upon the ordering of heights of faces in $M_3(\alpha)$. There is therefore an open range of direction vectors ζ for which M cannot be simply unfolded. This implies that there is some angle ϕ such that if ζ is within ϕ of $(0, 0, 1)$, then any normal order unfolding of $M_3(\alpha)$ with direction ζ will contain an overlap. □

Theorem 5.3.14. *There exists a convex polyhedron P such that every normal order unfolding of P contains an overlap.*

Proof. The proof of this theorem is identical to that of Theorem 5.3.11. We therefore only sketch the ideas.

We first embed $M_3(\alpha)$ into an isosceles triangle to simplify the construction. Then we argue that the value of ϕ in Lemma 5.3.13 is independent of the orientation and scaling of $M_3(\alpha)$. We are therefore free to rotate, translate, and uniformly scale $M_3(\alpha)$ and Lemma 5.3.13 will still apply (with vector ζ_0 in place of $(0, 0, 1)$ to denote the orientation of $M_3(\alpha)$).

We then form a polyhedron by placing instances of $M_3(\alpha)$ upon the unit sphere. These instances will be placed so that for each choice of ζ there is some instance of $M_3(\alpha)$ such that ζ is within angle ϕ of the corresponding ζ_0. But then this instance of $M_3(\alpha)$ creates an overlap in the unfolding, by Lemma 5.3.13. We conclude that the polyhedron will unfold with an overlap in any normal order unfolding. □

Chapter 6

Reconstructing Polygons and Polyhedra

In this chapter we investigate the problem of reconstructing a polygon or a polyhedron from edge information. We consider two main variants of this problem: reconstruction from edge lengths and reconstruction from edge vectors. We also investigate the effect of imposing restrictions such as convexity, orthogonality, and non-degeneracy upon the constructed objects. It is proved that most of these problems are NP-complete, although some are only weakly NP-complete.

6.1 Introduction

The results in this chapter are most directly motivated by a summary of open problems in the area of polyhedron reconstruction by Demaine and Erickson [15]. Among other questions, Demaine and Erickson ask whether it is NP-hard to determine whether a polyhedron can be constructed from a given multiset of vectors as edges. That is, we wish to reconstruct a polyhedron given the lengths and (undirected) orientations of its edges.

The equivalent problem for polygons is somewhat simpler since the edges of a polygon always form a simple cycle. In other words, a sequence of vectors can form the edges of a polygon if and only if they can be arranged into a simple closed chain. The key step in finding such a closed chain is an assignment of direction to each vector, such that the sum of the resulting vectors is zero. In other words, we must split our (positive) vectors into two subsets with equal sum. This problem is reminiscent of the well-studied NP-hard problem Partition. Indeed, this similarity is key for our results: we use it as a foothold for our NP-hardness proofs.

An associated problem is that of reconstructing polygons and polyhedra from edge lengths

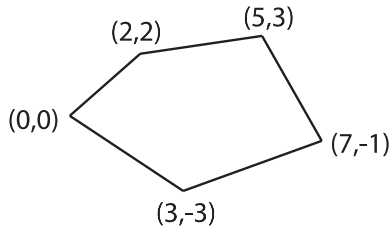


Figure 6.1: A sample polygon

only. A well-known result is that it is easy to determine whether a convex polygon can be constructed with a given multiset of edge lengths ([22], Lemma 3.1). However, this result depends (informally speaking) on the large degree of freedom inherent in configuring a closed polygonal chain. When restrictions such as orthogonality are added, the decision problem becomes NP-hard, as we shall prove.

In this chapter we shall analyze the complexity of reconstructing various classes of polygons and polyhedra from either edge lengths or edge lengths and orientations. A full summary of our results is presented in Table 6.1. Among our key results is that the problem of reconstructing polygons from edge vectors is weakly NP-complete, the problem of reconstructing polyhedra from edge vectors is NP-complete when degeneracies are allowed, and the problem of reconstructing polyhedra from edge lengths is NP-hard when degeneracies are allowed.

6.2 Preliminaries

6.2.1 Lengths and Orientations

We begin by making explicit the notions of "edge length" and "edge orientation" via an example. Consider the polygon P shown in Figure 6.1. We wish to extract information about the edges of this polygon. There are many pieces of data that could be stored; the locations of all endpoints, the combinatorial structure of which edge is incident with which, the angles between edges, etc. All of this information together would completely specify our polygon, so reconstructing the polygon from this information is trivial. The more interesting problem is to specify only a small amount of information, then attempt to reconstruct the polygon.

The least information we shall consider is the length of each edge. For the sample polygon, this information would be given as the multiset $Len(P) = \{\sqrt{8}, \sqrt{10}, \sqrt{18}, \sqrt{20}, \sqrt{20}\}$. This information is referred to as the *edge lengths*.

In addition to the length of each edge, we may also be given information about how the edges are to be oriented. This corresponds to storing a vector for each edge of the polygon. However, for each edge, there are two possible vectors: one positive and one negative (recall that a vector is positive when its first non-zero coordinate is positive). We shall take all vectors to be positive. For our sample polygon, this information would be given as the multiset $Vec(P) = \{(2,-4),(2,1),(2,2),(3,-3),(4,2)\}$. This information is referred to as the *edge orientations* or *edge vectors*.

If we do not require that all vectors be positive, it is possible to encode an additional bit of information for each edge in the choice of positive or negative vectors. In particular, we could use the signs of the vectors to determine the standard directions of the edges from the standard traversal described in Lemma 2.3.1. For our example, this corresponds to the multiset $\{(2,-4),(2,1),(2,2),(-3,3),(-4,-2)\}$. We call this information the *edge directions*. In other words, the edge direction information is *directed*, whereas the edge orientation information $Vec(P)$ is *undirected*.

We will not be considering the problem of reconstruction from edge directions in this thesis. We have defined edge directions here merely to distinguish them from edge orientations. We will also be making use of edge directions in some of our proofs, so it is especially important not to confuse the two.

6.2.2 Equivalence of Convex and General Polygons

Our first result is a simple consequence of Lemma 2.3.1 applied to reconstruction from vectors. The idea is that since every polygon is a chain of edges and all closed chains of edges in a particular order form convex polygons (by Lemma 2.3.1), one can always reorder the edges of any polygon to form a convex polygon.

Lemma 6.2.1. *If collinear edges are allowed, a polygon can be reconstructed from a sequence of edge vectors if and only if a convex polygon can be reconstructed from those vectors.*

Proof. We prove the non-trivial direction. Suppose that a non-convex polygon can be constructed from a given sequence $(V_1,\ldots,V_n)$ of edge vectors. Take the standard order and direction of those vectors, $(W_1,\ldots,W_n)$. Now reorder these vectors to the order specified in Lemma 2.3.1. That is, reorder the vectors to all positive vectors in decreasing order by slope, followed by all negative vectors in decreasing order by slope. Form a chain from the vectors in this new order. Then the resulting chain, say $(W_{\sigma(1)},\ldots,W_{\sigma(n)})$ where σ is a permutation, is still closed, and now corresponds to a convex polygon by Lemma 2.3.1. Note, however, that the original polygon may have been non-degenerate, and this reordering process may have introduced degeneracies. □

6.2.3 Sets of Edge Lengths

These next two lemmas are technical results regarding the distributions of edge lengths in polyhedra. The first is a rough analog of the result for polygons: that the largest edge length is not larger than the sum of the others. In the case of polyhedra, if there is one edge with a long length, then there must be 2 disjoint sets of edges such that each set has total length at least as large as that long edge. The second result is for orthogonal polyhedra, where it will turn out that 3 such disjoint sets are guaranteed.

Edge Lengths in Polyhedra

Lemma 6.2.2. *Given a polyhedron P and edge e_0 of P, reorient P such that e_0 is parallel to the x-axis. Suppose $|e_0| = k$. Then there are disjoint sets A_1, A_2 of edges of P such that*

$$\sum_{e \in A_i} |v(e).x| \geq k$$

for $i = 1, 2$.

Proof. The edge e_0 must be adjacent to two faces of P, say f_1 and f_2. These faces can have no other edge in common, since otherwise they would share interior points which contradicts the definition of a polyhedron. Let A_1 be the set of all edges adjacent to f_1 besides e_0. Define A_2 similarly with respect to f_2. We then have that A_1 and A_2 are disjoint.

But now, since $\{e_0\} \cup A_1$ forms a polygon, the vectors of $\{e_0\} \cup A_1$ form a closed chain and therefore sum to 0 under a standard orientation. Hence

$$\sum_{e \in A_1} |v(e).x| \geq |v(e_0).x| = k.$$

The same result holds for the edges in A_2. □

Edge Lengths in Orthogonal Polyhedra

The result of this section is similar to Lemma 6.2.2, but is rather stronger. In an orthogonal polyhedron we can construct three sets of edges with length at least any given edge length, rather than just two. The proof of this result is far more involved than that of Lemma 6.2.2, and will be done via a series of claims.

Lemma 6.2.3. *Suppose P is an orthogonal polyhedron and e_0 is an edge of P with $|e_0| = k$. Then there are disjoint sets A_1, A_2, A_3 of edges of P parallel to e_0 such that*

$$\sum_{e \in A_i} |e| \geq k$$

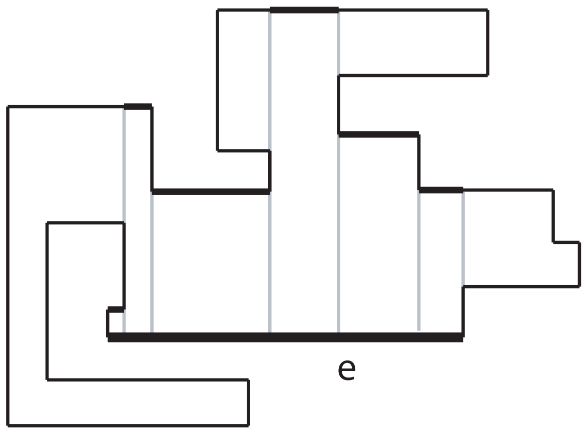


Figure 6.2: Line segments opposite an edge e. If e is the labeled thick edge in the drawn polygon P, the line segments in bold are $L(P, e)$. The grayed lines are drawn as guides.

for $i = 1, 2, 3$.

Let us first provide a sketch of the proof. The idea is similar to that in Lemma 6.2.2. That is, we take edges in the two faces incident with e_0 and argue that their sum is larger than $|e_0|$. However, we can take this argument one step further. Take one of those two faces, and for every edge considered we can apply the same operation again: choose another face incident with that edge, and sum up *its* edges. This roughly gives us our third set, but does not guarantee disjointness of all edges. We get around this problem by considering segments of edges, instead of working with edges directly. Note that the reason we require such a complex argument is that we are not restricting P to be non-degenerate; faces similar to that in Figure 6.2 can occur, possibly interlocked with coplanar faces.

Note that this result does not hold for non-orthogonal polyhedra. Consider a long, skinny triangular prism, and take e_0 to be one of the long edges. Then there are only 2 other edges as long as e_0, and no third set can be formed. The difference for orthogonal polyhedra is that they are limited in their possible face orientations, so in the following proof we can make strong arguments about disjointness of sets of edges.

Proof. Let us first make a few definitions. Suppose a polygon P_0 has an edge e. At each point p of e, take the ray orthogonal to e directed toward the interior of P_0 and let p' be the first point on an edge of P_0 (other than e) intersected by this ray. We say that p' is the *point opposing* e *at point* p. Let $A(P_0, e)$ be the set of all points opposing e (at some point in e) in polygon P_0.

Note that the set $A(P_0, e)$ forms a finite set of line segments, corresponding to segments of edges in P_0. Let $L(P_0, e)$ be that set of line segments corresponding to $A(P_0, e)$. See Figure 6.2 for an illustration of $L(P, e)$. We also define $A(P_0, r)$ and $L(P_0, r)$ where r is a segment of an edge e. These are simply the subsets of $A(P_0, e)$ and $L(P_0, e)$ corresponding to points opposite points in r.

Claim 6.2.4. *Suppose orthogonal polygon P contains edge e, and point p' is the point opposing e at point p. Then p' lies upon an edge e' parallel to e, and p is the point opposing e' at p'.*

Proof. We first prove the parallel requirement. Every edge of P is parallel with e or orthogonal to e. If edge e' is parallel to e we are done. If an edge e' is orthogonal to e', the only way in which it can have a point p' opposing e is if p' is on a corner: p' is the vertex incident with e' that is closest to e, and p' is also incident with an edge parallel to e. Thus, in either case, p' must be a point upon an edge parallel with e.

Let e' be the parallel edge containing p'. Then the line orthogonal to e' at p' is identical to the line orthogonal to e at p. Since we know this line has no intersections with the boundary of P between p and p', we conclude that p must be the point opposite e' at p', as required. □

Claim 6.2.5. *If orthogonal polygon P contains edge e, and r_0 is a line segment contained in e, then $\sum_{r \in L(P, r_0)} |r| = |r_0|$. In particular, $\sum_{r \in L(P,e)} |r| = |e|$.*

Proof. By Claim 6.2.4, each segment $r \in L(P, r_0)$ is parallel to e. The line segments of $L(P, r_0)$ are therefore simply translations of corresponding line segments contained in r_0 of the same length, and every point of r_0 lies upon exactly one such line segment. We conclude that the sum of the lengths of these segments is precisely the length of r_0. □

Now we prove the desired result. See Figure 6.3 for a simple example of the three sets of edges that we are interested in. We shall define these sets of edges formally.

There must be two disjoint faces adjacent to e_0 in P, call them f_1 and f_2. Orient P in $\mathbf{R}^3$ so that e_0 has endpoints $(0, 0, 0)$ and $(k, 0, 0)$, and the interior of f_1 lies in the positive y direction relative to e_0. Note then that either f_2 lies upon the xz-plane or the interior of f_2 lies in the negative y direction relative to e_0.

Now consider $A(f_1, e_0)$ and $A(f_2, e_0)$. These will form the first two of our desired three sets. Then for each line segment $r \in L(f_1, e_0)$, let f_r be the face incident with r besides f_1. Consider $A(f_r, r)$ for each such r. The union of these sets will form the last of our desired sets.

The bulk of the remaining proof will be in showing disjointness (or near disjointness) of these sets.

Claim 6.2.6. *No edge contains segments in both $L(f_1, e_0)$ and $L(f_2, e_0)$.*

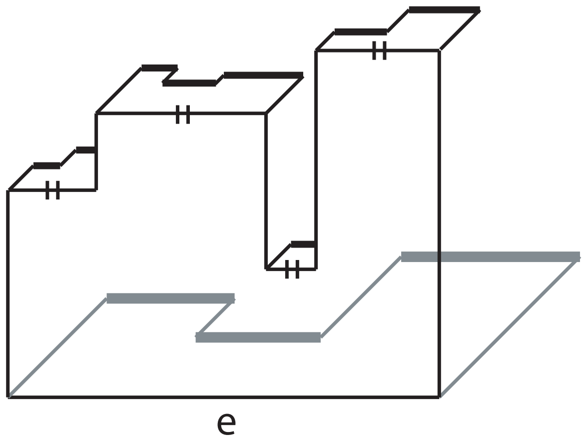


Figure 6.3: A portion of an orthogonal polyhedron showing the three edge sets of interest in Lemma 6.2.3, relative to edge e. One set consists of the hashed edges, another of the thick black edges, and the last of the thick gray edges (shown through the face in the forefront).

Proof. All points in $A(f_1, e_0)$ lie in the (strictly) positive y halfspace, as the inward-facing normal of e_0 for polygon f_1 is $(0, 1, 0)$. Further, the points in $A(f_2, e_0)$ either lie upon the xz-plane or in the negative y halfspace. We conclude that $A(f_1, e_0)$ and $A(f_2, e_0)$ are distinct.

All edges containing segments in $L(f_1, e_0)$ or $L(f_2, e_0)$ are parallel to e_0, by Claim 6.2.4. Thus each such edge is parallel to the xz-plane, and cannot contain points both in the positive y halfspace and not in the positive y halfspace. So no edge can contain points in both $A(f_1, e_0)$ and $A(f_2, e_0)$, as required. □

Claim 6.2.7. *No edge contains segments in both $L(f_r, r)$ and $L(f_2, e_0)$ for any $r \in L(f_1, e_0)$.*

Proof. Note that the interior of f_1 lies in the negative y direction from r. Then the interior of f_r cannot lie in the negative y direction from r (since then it would intersect f_1). We conclude that f_r is either parallel to the xz-plane, or the interior of f_r is in the positive y direction from r. Then we have that $A(f_r, r)$ must contain only points in the positive y halfspace, so $A(f_r, r)$ and $A(f_2, e)$ are disjoint for all $r \in L(f_1, e_0)$.

Note also that r must be parallel to e_0, by Claim 6.2.4. This implies that no edge can contain segments in both $L(f_r, r)$ and $L(f_2, e_0)$, by an argument identical to that in Claim 6.2.6. □

Claim 6.2.8. *No edge contains segments in both $L(f_r, r)$ and $L(f_1, e_0)$ for any $r \in L(f_1, e_0)$.*

Proof. Suppose edge e parallel to e_0 contained a point in $A(f_r, r)$ and a point in $A(f_1, e_0)$. So in particular e is a common edge between f_r and f_1. But line segment r lies upon another edge in common between f_1 and f_r. We conclude that f_1 and f_r must be coplanar.

Note that r must lie between e and e_0, since otherwise the points of e between r and e_0 would have been in $A(f_1, e_0)$, so r would not be in $L(f_1, e_0)$. But now the interiors of faces f_1 and f_r must lie in the same direction from e, since both r and e_0 lie in that direction! This is a contradiction. ☐

Claim 6.2.9. *There are only finitely many points that lie in two different sets $A(f_r, r)$ and $A(f_{r'}, r')$ for $r, r' \in L(f_1, e_0)$.*

Proof. Suppose that $A(f_r, r)$ and $A(f_{r'}, r')$ contain a point p in common for some $r, r' \in L(f_1, e_0)$. Say p line on the interior of an edge e. Consider the point opposing e at p. By Claim 6.2.4, this point must lie on both r' and r. This implies that p is the point opposing r and r' at a common endpoint – that is, a vertex of the polyhedron. If, on the other hand, p does not lie on the interior of an edge, then e is a vertex of the polyhedron.

So for each pair $r, r' \in L(f_1, e_0)$, $A(f_r, r) \cap A(f_{r'}, r')$ is finite. Since there are only finitely many such pairs, the result follows. ☐

We can now finally define our sets of edges. Let A_1 be the set of edges from which $L(f_1, e_0)$ contains a segment. In other words, if $r \in L(f_1, e_0)$ where r is a segment on edge e, then $e \in A_1$. Let A_2 be the set of edges from which $L(f_2, e_0)$ contains a segment. Let A_3 be the set of edges from which $L(f_r, r)$ contains a segment for some $r \in L(f_1, e_0)$. Then Claims 6.2.6, 6.2.7, and 6.2.8 imply that A_1, A_2, and A_3 are pairwise disjoint.

Now Claim 6.2.9 implies that the line segments opposing all line segments opposing e_0 are disjoint, except for possibly some set of points that is finite (and hence has measure 0). We can then apply Claim 6.2.5 twice to conclude that

$$\sum_{e \in A_3} |e| \geq \sum_{r \in L(f_1, e_0)} \sum_{r' \in L(f_r, r)} |r'| = \sum_{r \in L(f_1, e_0)} |r| = k.$$

Further, Claim 6.2.5 can be applied to $L(f_1, e_0)$ and $L(f_2, e_0)$ to get that

$$\sum_{e \in A_1} |e| \geq \sum_{r \in L(f_1, e_0)} |r| = k$$

and

$$\sum_{e \in A_2} |e| \geq \sum_{r \in L(f_2, e_0)} |r| = k$$

as required.

☐

6.3 Problems For Reduction

In this section we shall introduce the problems that we will use for reductions in our NP-hardness proofs. Most are known to be NP-complete, although for one variant this will need to be proven.

The main problem we employ is PARTITION. An instance of PARTITION is a sequence of n positive integers $(w_1, \ldots, w_n)$. The problem is to determine whether these integers can be partitioned into disjoint sets A_1 and A_2 such that

$$\sum_{w \in A_1} w = \sum_{w \in A_2} w.$$

It is well known that this problem is NP-complete [17]. It should also be noted that if we define S by $2S = \sum_{i=1}^{n} w_i$, this problem is equivalent to finding a subset of the w_i that sum to S.

We now list some useful variants of PARTITION.

EQUAL-CARDINALITY-PARTITION: In this variant we require that $|A_1| = |A_2|$. This problem is known to be NP-hard; see [17].

UNIQUE-VALUES-PARTITION: This problem is identical to PARTITION, except that the inputs w_i are necessarily distinct. The proof that this problem is NP-hard follows from the proof that PARTITION is NP-hard [17]. Instead of restating this proof formally, we defer to the proof that the next variant is NP-hard, which implies by trivial reduction that UNIQUE-VALUES-PARTITION is NP-hard.

EQUAL-CARDINALITY-UNIQUE-VALUES-PARTITION: In this variant we require that the input values all be unique, and that the solution satisfy $|A_1| = |A_2|$. The proof is a simple extension of the Partition reduction given in [17], but we include it formally here for completeness.

Theorem 6.3.1. EQUAL-CARDINALITY-UNIQUE-VALUES-PARTITION *(ECUVP) is NP-complete.*

Proof. It is not difficult to see that ECUVP is in NP. A certificate can be provided in the form of a particular subset, A_1, and its sum and cardinality can be checked in polynomial time.

To show NP-hardness, we shall reduce from THREE-DIMENSIONAL MATCHING (3DM). An instance of 3DM is three disjoint sets $U = \{u_1, \ldots, u_n\}, V = \{v_1, \ldots, v_n\}, W = \{w_1, \ldots, w_n\}$ and a set of triples $M = \{m_1, \ldots, m_k\}$, where $M \subset U \times V \times W$. The problem is to determine whether some subset of M is a perfect matching; that is, whether there exists $M' \subset M$ such that each item in U, V, or W occurs in precisely one $m \in M'$. The 3DM problem is NP-Complete; there is a known reduction from 3-SAT [17].

Suppose we have an instance U, V, W, M of 3DM. We can assume that at least one value in U, V, or W is included in 3 or more matchings (otherwise the problem can be solved in polynomial time [17]). Without loss of generality (by renaming sets and/or reindexing), say u_1 is included

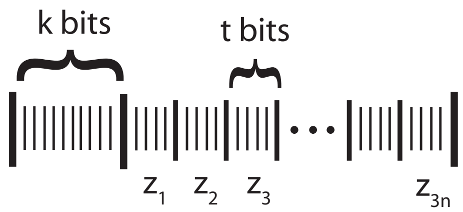


Figure 6.4: Bit format of an integer corresponding to a triple m_i in the reduction from 3DM. The blocks $z_1, \ldots, z_{3n}$ correspond to the elements of U, V, and W. These blocks are ordered so that the entry corresponding to z_1 occurs in at least three triples.

in 3 or more matchings. We now create an instance of ECUVP that has a solution if and only if there is a solution to our instance of 3DM. That is, we shall construct a set of input values for ECUVP. We build these values by specifying their bits in base 2, with most significant digits to the left.

Let $t = \lfloor \lg(k+1) \rfloor$. We first construct one integer weight x_i for each $m_i \in M$, as follows. The integer x_i consists of $3n$ blocks of t bits each, with k additional bits on the left. Each block corresponds to an item in U, V, or W. Call these blocks $z_1, z_2, \ldots, z_{3n}$, where the first n blocks correspond to the values in U in order of index, the next n correspond to the values in V, and the last n to the values in W. In particular, z_1 corresponds to u_1. See Figure 6.4.

Initially, set all bits of every block to 0. Then, if $m_i = (u, v, w)$, set the rightmost bit of each block corresponding to u, v, and w to 1. Finally, we set bit i of the leftmost k bits to 1. The resulting bit string is the binary representation of x_i.

We now construct another integer y_i for each m_i. This is done by only setting bit i of the leftmost k bits to 1, and leaving all bits in the $3n$ z_i blocks zero. In other words, $y_i = x_i$ AND $1^k 0^{3tn}$.

An important thing to notice about our construction is that since there are only k triples in M, and each value in U, V, or W can occur only once in a triple, there are at most k values x_i with a 1 in any particular block z_i. Thus, since a block z_i can represent integers at least as large as k, there will be no overflow between blocks if all the x_i and y_i are added together.

Take S to be the sum of all the x_i and y_i. Denote by A the integer formed by setting the rightmost bit of each block to 1, and setting all of the leftmost k bits to 1. Define two additional values $b_1 = 2S - A$ and $b_2 = S + A$. Our instance of the ECUVP is then b_1, b_2, and the x_i's and y_i's.

To demonstrate that this is a valid instance of ECUVP, we need to show distinctness of the values.

Claim 6.3.2. *The values b_1, b_2, $x_1, \ldots, x_k$ and $y_1, \ldots, y_k$ are all distinct.*

Proof. We first show that $S > 2A$. Consider the block z_1. Left of that blocks, S and $2A$ are identical: both are the sum of two instances of k 1's. Within the block z_1, $2A$ will have a value of 2 in that block, but S will have a value of at least 3 (since u_1 occurs in at least 3 matchings).

Thus $b_1 > b_2$ and certainly b_2 is larger than any x_i or y_i. All x_i and y_i are distinct, by differences in the k leftmost bits and the fact that each x_i will have exactly three blocks with non-zero entries, whereas each y_i has none. We conclude that all of our values are unique. $\square$

We are now ready to show that we have a reduction from 3DM to ECUVP.

Since the total sum of all values in our instance is $4S$, and there are $2n + 2$ values given, a solution to our instance of ECUVP corresponds to finding two disjoint subsets of $n + 1$ values that sum to $2S$.

Suppose that there is a solution M' to our instance of 3DM. Then we take x_i for every $m_i \in M'$, and y_i for every $m_i \notin M'$. These will sum to A, as each entry occurring in exactly one matching implies that there is exactly one 1 for each block in the sum, and one of x_i or y_i for each i implies that the leftmost k bits are all 1 in the sum. Thus these values plus b_1 will equal $2S$. This gives a subset with $n + 1$ values that sum to $2S$, as required.

Now suppose that there is a solution to our instance of ECUVP. As $b_1 + b_2 = 3S > 2S$, we know that b_1 and b_2 occur in different subsets. Consider the subset containing $b_1 = 2S - A$, say A_1. The remaining values in A_1 must sum to A. Since there can be no carry between blocks, there must be precisely one value with a 1 in each block of our integer representation. Also, since there are only two weights with bit i of the leftmost k bits being 1, and all these bits must be 1 in the sum, we must have exactly one weight with bit i of the leftmost k bits being 1, for each i.

We conclude that the remaining n values in this subsets are precisely one of x_i or y_i for each i. Every block z_i has a non-zero entry in precisely one of the x_i's chosen for this subset. That is, every value in U, V, and W is represented exactly once in the triples m_i for each x_i in this subset. A solution to our instance of 3DM is therefore $\{m_i : x_i \in A_i\}$.

As a final note, we argue that the size of integers in the instance to ECUVP is not too large. The instance of 3DM is given by the entries of the triples, which are $3k$ values each referring to one of n entries in U, V, or W. Thus the total problem size is $O(k \lg n)$. Note also that each value in U, V, or W must occur in some triple, otherwise the problem is trivial. We therefore have $n = O(k)$, so problem size is $O(k \lg k)$. Our constructed integers have $k + 3n \lg k = O(k + n \lg k) = O(k \lg k)$ bits. We have $2n + 1$ such integers, for a total problem size of $O(k^2 \lg n)$, which is certainly polynomial in the original problem size.

$\square$

6.4 Reconstructing from Edge Vectors

In this section we consider the problem of reconstructing a polygon or polyhedron from a set of edge vectors. That is, we are given vectors $V_1, \ldots, V_n$ and wish to determine whether they can be placed to form the edges of a polygon or polyhedron. We assume that each vector has integer coordinates, each representable in k bits.

It is important to recall that we consider the edge vectors as being undirected, as discussed in Section 6.2.1. That is, the problem remains the same if we negate some of the input vectors. Our vectors therefore denote the length and orientation of edges in $\mathbf{R}^2$ or $\mathbf{R}^3$ but do not represent any information about that edge's position relative to any given traversal of our constructed object.

6.4.1 Polygons

We begin by attempting to reconstruct convex polygons from edge vectors. This problem will be shown to be NP-complete. We then consider variants of the problem by allowing degeneracy or non-convexity, or by enforcing orthogonality, or some combination of these. It will turn out that all of these variants are NP-complete as well.

We also consider whether these problems are strongly or weakly NP-complete. It will turn out that convex, orthogonal, and degenerate polygons can be reconstructed in pseudo-polynomial time, and these problems are therefore only weakly NP-complete. The question as to whether the decision problem for general polygons is strongly or weakly NP-complete is left open.

Reconstructing Polygons and the Class NP

We will show that all variants of deciding whether a polygon (not necessarily convex) can be reconstructed from edge vectors are in NP. Informally, this follows because a set of vectors form a polygon if and only if they form a simple closed chain, by definition. An oracle can therefore simply provide this chain as a certificate.

Lemma 6.4.1. *The problem of determining whether a polygon can be reconstructed from a sequence $(V_1, \ldots, V_n)$ of edge vectors is in NP.*

Proof. Suppose the vectors can be placed together to form a polygon P. Let $(W_1, \ldots, W_n)$ be the vectors in standard order and orientation (where each W_i is either V_j or $-V_j$ for some j).

We supply $(W_1, \ldots, W_n)$ as a certificate for this solution. To verify that these vectors indeed trace out a polygon, we need only check that the implied chain is closed and simple. The chain is closed if and only if the vectors sum to 0, which is easily verified. For simplicity, we must verify that the corresponding line segments intersect only at incident vertices. Since each vector has integer coordinates, each line segment has integer coordinate vertices as well; the endpoints for the line segment corresponding to W_j are $\sum_{i=1}^{j-1} W_i$ and $\sum_{i=1}^{j} W_i$. We can therefore compute the endpoints of all the line segments, then use any known polynomial-time algorithm to find all intersections between segments (since intersection points will have rational coordinates).

Thus our certificate can be verified to correspond to a polygon in polynomial time, as required. □

It remains to show that the requirements of each variant of our reconstruction problem can also be verified from our given certificate in polynomial time.

Lemma 6.4.2. *The problem of determining whether a polygon can be reconstructed from a sequence $(V_1, \ldots, V_n)$ of edge vectors is in NP, even if we restrict the polygon to be one or more of orthogonal, convex, and non-degenerate.*

Proof. We use the same certificate as in Lemma 6.4.1. We need only show that each of the specified restrictions can be verified in polynomial time for our certificate.

Orthogonality can be checked simply by verifying that each vector has exactly one non-zero coordinate. Non-degeneracy is verified by ensuring that no two consecutive vectors have the same slope.

The test for convexity follows from Lemma 2.3.1. We need only test that the certificate vectors are in the order stated in that lemma, which can be done in polynomial time. □

Convex Polygons

Our first problem is one of reconstructing convex polygons from edge vectors. We assume here that our polygon must be non-degenerate, meaning that no two adjacent edges can be collinear.

Lemma 6.4.3. *It is NP-hard to determine whether a set of* 2*-dimensional vectors can form the edges of a non-degenerate convex polygon.*

Proof. We shall reduce from Equal-Cardinality-Unique-Values-Partition (ECUVP). Suppose we have an instance $(w_i)_{i=1}^{2n}$ of ECUVP. Define S by $2S = \sum_{i=1}^{2n} w_i$. Then we supply the following vectors to our reconstruction problem:

1. $X_i = (w_i, 1)$ for $1 \leq i \leq 2n$
2. $B_1 = B_2 = (S, n)$

We claim that these vectors form the edges of a convex polygon if and only if there is a solution to our instance of ECUVP.

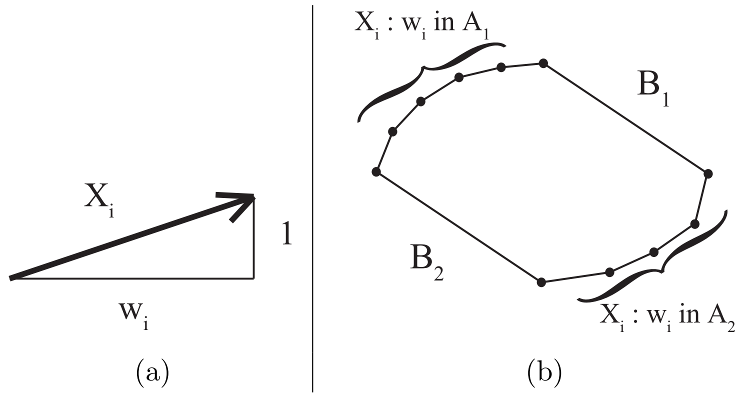


Figure 6.5: An example of the construction in the proof of Lemma 6.4.3. (a) The construction of the vector X_i from a value w_i. (b) The polygon resulting from a solution to Partition on the values (w_i).

Suppose that our instance of ECUVP has a solution A_1, A_2. Then we form our polygon as a chain of vectors $(W_i)_{i=1}^{2n+2}$, as follows:

- $W_1, \ldots, W_n$ are the vectors X_i corresponding to values of A_1, sorted by increasing w_i value.
- $W_{n+1} = B_1$.
- $W_{n+2}, \ldots, W_{2n+1}$ are the vectors $-X_i$ corresponding to values of A_2, sorted by increasing w_i value.
- $W_{2n+2} = B_2$.

See Figure 6.5 for an illustration of this construction.

Then we note that the sum of these vectors is 0, so they form a closed chain. Also, note that they are ordered by slope as required by Lemma 2.3.1, so they form a convex polygon. We conclude that this is a solution to our reconstruction problem, as required.

For the reverse direction, suppose that our constructed vectors form the edges of a convex polygon P. Let $W_1, \ldots, W_n$ be the vectors X_i in standard orientation, and similarly let B_1', B_2' be vectors B_1, B_2 in standard orientation. Let $A_1 = \{w_i : W_i = X_i\}$ and $A_2 = \{w_i : W_i = -X_i\}$. We claim that A_1, A_2 is a solution to ECUVP.

Note first that the sum of all W_i and B_i' must be 0, since they form a closed chain.

Claim 6.4.4. *Either $B_1' = -B_1$ and $B_2' = B_2$, or $B_1' = B_1$ and $B_2' = -B_2$.*

Proof. Suppose for contradiction that $B_1' = B_1$ and $B_2' = B_2$. Then $B_1' + B_2' = (2S, -2n)$. To yield a sum of 0 on the x-coordinate, we must have $W_i = -X_i$ for all i. But this does not achieve a sum of 0 on the y-coordinate, a contradiction. Similarly, we cannot have $B_1' = -B_1$ and $B_2' = -B_2$. This concludes the proof of the claim. □

Without loss of generality, assume $B_1' = B_1$ and $B_2' = -B_2$. Then $B_1' + B_2' = 0$, so we must have

$$\sum_{i=1}^{2n} W_i = (0, 0)$$

and hence

$$\sum_{W_i = X_i} X_i - \sum_{W_i = -X_i} X_i = (0, 0)$$

which implies (by examining x-coordinates and y-coordinates separately)

$$\sum_{w_i \in A_1} w_i = \sum_{w_i \in A_2} w_i$$

and

$$\sum_{w_i \in A_1} 1 = \sum_{w_i \in A_2} 1$$

and hence $|A_1| = |A_2|$. We conclude that our partition of the w_i's forms a solution to our instance of ECUVP.

□

We have now shown that determining whether a non-degenerate convex polygon can be constructed from a set of edge vectors is NP-hard. This particular problem is only weakly NP-hard, however. There is a pseudo-polynomial algorithm for solving it, which proceeds as follows.

Lemma 6.4.5. *There is a pseudo-polytime algorithm to determine whether a convex polygon can be reconstructed from a sequence of edge vectors.*

Proof. Suppose we are given vectors $V_1, \ldots, V_n$. We can assume that all of these vectors are non-negative (simply negate any vectors for which this is not true, since initial orientation is irrelevant). Let $S = \frac{1}{2} \sum_{i=1}^{n} V_i$. Our problem is now to find some subset of the V_i that sums to S. We shall use dynamic programming to find this subset. Our pseudo-polynomial parameter will be m, defined as

$$m = \max \left\{ S.x, \sum_{i=1}^{n} |V_i.y| \right\}.$$

Recall that a *pseudo-polynomial parameter* is a value that is polynomial in the input size if the input values are expressed in unary, but exponential in the input size if the input values are expressed in binary. This value m will be used as a parameter in the time complexity of our algorithm.

First we check that non-degeneracy is possible. If any 3 of the V_i have the same slope, there is no possible non-degenerate convex polygon so our algorithm returns false. This follows from Lemma 2.3.1: each vector must be in either the upper or lower chain, and the vectors in those chains must be ordered by slope, so if there are three parallel vectors then at least two must be adjacent and collinear which contradicts non-degeneracy.

If there are no 3 parallel vectors, relabel the V_i so that any two vectors with the same slope are labeled V_i, V_i'. We say in this case that V_i' is the *twin* of V_i. Note now all V_i have unique slopes (i.e. excluding the V_i'); say there are $n' \leq n$ such vectors. Finally, take $V_i' = (0, 0)$ if V_i has no twin.

We now perform dynamic programming. We fill a three-dimensional binary table, indexed by $e(i, j, k)$, for $1 \leq i \leq n'$, $0 \leq j \leq m$, and $-m \leq k \leq m$. We wish to fill this table such that $e(i, j, k) = 1$ if some subset of vectors with indices at most i can sum to (j, k), where exactly one of any V_l, V_l' pair must be used for each $1 \leq l \leq i$. Otherwise, $e(i, j, k) = 0$.

We fill the table as follows. Initially we set $e(1, V_i.x, V_i.y) = e(1, V_i'.x, V_i'.y) = 1$, all other $e(1, j, k) = 0$. Then for each subsequent value i, and for all j and k, set $e(i + 1, j, k) = 1$ if $e(i, j - V_i.x, k - V_i.x) = 1$ or $e(i, j - V_i'.x, k - V_i'.x) = 1$. When we are finished filling the table (i.e. we reach $i = n'$), we return true if and only if $(n', S.x, S.y) = 1$. There are at most $2nm^2$ entries in the table, so total running time is $O(nm^2)$.

To justify this algorithm, suppose there is some positive set of vectors adding to S. These will form the upper chain as in Lemma 2.3.1. We order them by decreasing slope, with any vertical vectors being placed first. Note that since only one of each V_i, V_i' is chosen for each i, the slopes in the upper chain will all be unique. For all values not taken, take their negatives and form the lower chain from them. Again, order by decreasing slope as prescribed by Lemma 2.3.1. These vectors must add to $-S$, since the sum of all the original vectors is $2S$. We conclude that the resulting chain is closed. Finally, Lemma 2.3.1 now gives us that the chain corresponds to a convex polygon, as required. □

Pairing the above argument with the result of Lemma 6.4.2, we conclude the following theorem.

Theorem 6.4.6. *Determining whether a set of* 2*-dimensional vectors can form the edges of a non-degenerate convex polygon is a weakly NP-complete problem.*

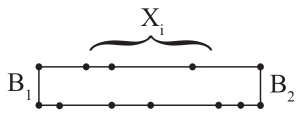


Figure 6.6: Forming a degenerate orthogonal polygon from an instance of PARTITION.

Degenerate Polygons

We now modify our problem to allow degeneracies in our polygons. It turns out that this variation is also NP-hard, via a simpler construction than that for non-degenerate convex polygons. We begin by considering the special case of convex orthogonal polygons. In other words, we consider box-shaped polygons, possibly with collinear edges.

Lemma 6.4.7. *It is NP-hard to determine whether a set of* 2*-dimensional vectors can form the edges of a (possibly degenerate) convex orthogonal polygon.*

Proof. Given 2-dimensional vectors $V_1, \ldots, V_n$, we wish to determine whether they can be placed in $\mathbf{R}^2$ to form the edges of a convex orthogonal polygon, where degeneracies (i.e. adjacent collinear edges) are allowed.

Given an instance $(w_i)_{i=1}^n$ of PARTITION, we supply the following vectors:

1. $X_i = (w_i, 0)$ for $1 \leq i \leq n$
2. $B_1 = B_2 = (0, 1)$

We now claim that these vectors form a (possibly degenerate) convex orthogonal polygon if and only if there is a solution to our instance of PARTITION. Suppose that A_1 and A_2 are a solution to PARTITION. Then we form our polygon as a rectangle with height 1 and width S; see Figure 6.6. This rectangle has B_1 and B_2 being the opposing edges of length 1, the edges of A_1 placed collinearly to form one side of length S (i.e. the top of the box), and the edges of A_2 placed collinearly to form the other side of length S (i.e. the bottom).

Suppose now that the X_i and B_i are arranged to form a polygon P. Take $(W_1, \ldots, W_n)$ to be the vectors X_i in standard direction. For each i, place w_i in A_1 if $W_i = X_i$. Otherwise (i.e. $W_i = -X_i$) place w_i in A_2. Then A_1 and A_2 form a partition of (v_i), and indeed the sums of values in A_1 and A_2 must be equal since the total displacement along the x-coordinate of the circuit made around P is 0. See Figure 6.7 for a general example of this approach (for a non-convex polygon, in this case).

□

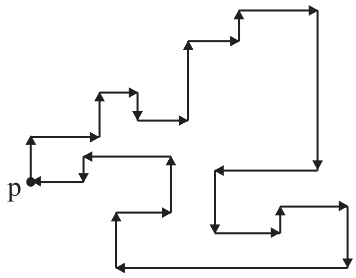


Figure 6.7: Obtaining a solution to PARTITION from a reconstructed orthogonal polygon.

But now Lemma 6.2.1 and the definition of polygon orthogonality imply the following result.

Lemma 6.4.8. *Deciding whether the following polygon types can be reconstructed from edge vectors is NP-Complete, when collinear adjacent edges are allowed.*

1. *Convex Polygons*
2. *Orthogonal Polygons*
3. *General Polygons*

Proof. By definition, any polygon reconstructed from axis-aligned vectors must be orthogonal. We can therefore drop the orthogonality requirement from the statement of Lemma 6.4.7 as it is implied by any reduction to orthogonal vectors. Also, by Lemma 6.2.1, a general (resp., orthogonal) polygon can be reconstructed if and only if a convex (resp., convex orthogonal) polygon can be reconstructed when degeneracies are allowed. Thus Lemma 6.4.7 implies NP-hardness of all the required polygon classes. □

In fact, these are all weakly NP-complete: there are pseudo-polytime algorithms for each of the decision problems.

Lemma 6.4.9. *There are pseudo-polytime algorithms to determine whether a set of vectors form the edge vectors for a convex, orthogonal, or general polygon, when degeneracies are allowed.*

Proof. We shall modify the algorithm from Lemma 6.4.5 to relax its non-degeneracy and convexity conditions.

For the case of convex polygons, we proceed with the algorithm from Lemma 6.4.5 but omit the notion of twins. Instead, we set each $V_i' = (0,0)$. We also no longer test whether there are three parallel vectors. The correctness of this algorithm follows in exactly the same manner as in Lemma 6.4.5.

This algorithm also answers the question for general polygons, since we can apply the algorithm to general polygons by Lemma 6.2.1.

Finally, for orthogonal polygons, we use our modified algorithm for convex polygons but additionally check that each vector has exactly one non-zero coordinate; otherwise no orthogonal polygon can be constructed. □

Putting all these results together, we get the following characterization of our reconstruction problems.

Theorem 6.4.10. *The problem of deciding whether the following polygon types can be reconstructed from edge vectors, when collinear adjacent edges are allowed, is NP-complete.*

1. *Convex Polygons*
2. *Orthogonal Polygons*
3. *General Polygons*

Non-Degenerate, Non-Convex Polygons

We now consider the reconstruction of non-convex polygons where collinear edges are not allowed. There are two main variants: orthogonal polygons and general non-convex polygons.

Lemma 6.4.11. *It is NP-hard to determine whether a set of* 2*-dimensional vectors can form the edges of a non-degenerate orthogonal polygon.*

Proof. We proceed by performing a Turing reduction from PARTITION. That is, our reduction uses multiple instances of the reconstruction problem. Given an instance $(w_i)_{i=1}^{n}$ of PARTITION we create three instances of this reconstruction problem by supplying the following vectors with $k = 1, 2, 3$:

1. $X_i = (w_i, 0)$ for $1 \leq i \leq n$
2. $Y_i = (0, 1)$ for $1 \leq i \leq n - 2$
3. $B_1 = (0, 1), B_2 = (0, k)$

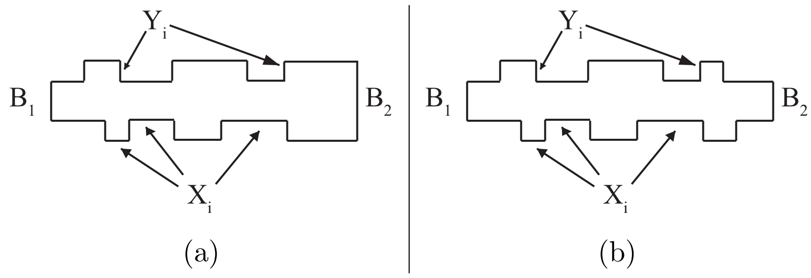


Figure 6.8: An example of non-degenerate orthogonal polygon reconstruction from an instance of Partition. The cases where (a) $k = 1$ and (b) $k = 3$ are shown.

We now claim that these vectors form a (possibly degenerate) orthogonal polygon for one of $k = 1, 2, 3$ if and only if there is a solution to our instance of Partition. Thus, to solve Partition, we need only run three instances of our reconstruction problem, which causes a contradiction if this problem can be solved in polynomial time.

Suppose that A_1 and A_2 are a solution to Partition. Then we form our polygon as a "lumpy rectangle;" see Figure 6.8. Place B_1 as the left side of the polygon. Now for each w_i in A_1, place X_i along the top of the polygon. Attach to this X_i a vector from Y_i, alternately angling up and down, starting upwards. Perform the same operation for the bottom of the polygon, using the values from A_2 and starting downward. The result is that the endpoints of the upper and lower perimeters are vertically aligned (they both have x-coordinate S). However, the distance between the two endpoints depends on the parity of A_1 and A_2.

- If $|A_1|$ and $|A_2|$ are odd then a vector of length 1 is needed.
- If $|A_1|$ and $|A_2|$ are even then a vector of length 3 is needed.
- If exactly one of $|A_1|$ or $|A_2|$ is odd then a vector of length 2 is needed.

Thus, for one value of k, the vector B_2 will be the appropriate length and can be placed as the right end of our rectangular structure.

Suppose now that the vectors can be arranged to form a polygon P for one of the values of k. Then we can form a partition A_1 and A_2 solving Partition by using the standard order of edge vectors for P, just as was done for degenerate orthogonal polygons in Lemma 6.4.7. See Figure 6.7 for an example.

□

Our use of a Turing reduction for the preceding proof is unfortunate. It is tempting to try to remove this requirement by reducing from Equal-Cardinality-Partition so that the correct

k to use can be derived from n. However, we have not found a way to derive a solution to Equal-Cardinality-Partition from an instance of orthogonal polygon reconstruction.

It is a simple matter to remove the orthogonality condition of Lemma 6.4.11, giving us the following theorem.

Theorem 6.4.12. *It is NP-complete to determine whether a set of 2-dimensional vectors can form the edges of a non-degenerate polygon. The problem remains NP-complete if we restrict the polygon to be orthogonal.*

Proof. The orthogonality of the reconstructed polygon in Lemma 6.4.11 can be guaranteed simply by providing orthogonal vectors to the reconstruction problem. The orthogonality condition can therefore be removed from the statement of that lemma and the result follows. □

We leave open the question of whether these problems are weakly or strongly NP-complete.

6.4.2 Polyhedra

In this section we extend the hardness proofs for polygons to three dimensions. Most of the results for polygons carry over to polyhedra, though with proofs involving more complex constructions.

In particular, the decision problems remain in NP for polyhedra.

Lemma 6.4.13. *The problem of deciding whether a sequence of vectors can form the edges of a polyhedron is in NP.*

Proof. Supposing that a given sequence of vectors $(V_i)_{i=1}^n$ can form a polyhedron P, we provide the following certificate. Translate P so that it has a vertex v located at $(0, 0, 0)$. Then all vertices of P occur at lattice points, bounded by the sizes of the edge vectors provided, and can therefore be represented in polynomial space. The certificate is therefore the polyhedron represented as a list of vertices, edge connectivity information, and face connectivity information. A winged-edge data structure would be a sufficient certificate [26].

To verify this construction, we need only match each edge (p_1, p_2) in this polyhedron with exactly one input vector which equals either $p_1 - p_2$ or $p_2 - p_1$. Note that this can be done in polynomial time: ensure that the input vectors are all positive (by negating any that are not), then only compare whichever of $p_1 - p_2$ or $p_2 - p_1$ is positive.

We need now only verify that the given construction is a valid polyhedron. There are well-known polytime algorithms for such verification. We need only verify that each edge is in two faces, the surface is connected, the cyclic sequence of faces around every vertex is connected, and no two faces intersect other than at common edges. See, for example, the algorithms presented by O'Rourke [26]. □

In addition, many properties about polyhedra can be verified in polynomial time given a reasonable data structure as a certificate. This implies the following.

Lemma 6.4.14. *The problem of deciding whether a sequence of vectors can form the edges of a polyhedron is in NP, even when we require that the polyhedron be one or more of convex, orthogonal, and non-degenerate.*

Proof. We need only verify the required properties given a winged-edge data structure as a certificate. Testing for orthogonality is easy; simply check that all faces are parallel to one of the axis-aligned planes. For non-degeneracy, check that no two incident faces are coplanar and that no two edges incident along the boundary of a face are collinear. To test for convexity, we need only test that all dihedral angles are no more than π, which is done by comparing the planes of the two faces incident with each edge of the polyhedron. □

Convex Polyhedra

We now show how to extend the NP-hardness construction for convex polygons to convex polyhedra.

Theorem 6.4.15. *It is NP-complete to determine whether a set of* 3*-dimensional vectors can form the edges of a non-degenerate convex polyhedron.*

Proof. The fact that the problem is in NP follows from Lemma 6.4.14. It remains to show NP-hardness.

We shall reduce from Equal-Cardinality-Unique-Values-Partition (ECUVP). Suppose we are given an instance of ECUVP, and A is the set of 2-dimensional vectors given in the proof of Lemma 6.4.3. We supply the following vectors to the polyhedral construction problem:

1. $P_i = Q_i = (x, y, 0)$ for each $a_i = (x, y) \in A$
2. $B_i = (0, 0, 1)$ for each $a_i \in A$

We claim that a polyhedron can be reconstructed from these vectors if and only if there is a solution to our instance of ECUVP.

Suppose that we are given a solution A_1, A_2 to our instance of ECUVP. Then we can construct a polygon p_1 from the vectors P_i and an identical polygon p_2 from the vectors Q_i as in Lemma 6.4.3, and connect these polygons into a prism using the vectors B_i. See Figure 6.9. A polyhedron can therefore be constructed from these vectors, as required.

On the other hand, suppose we can construct polyhedron P from these vectors. The B_i are the only vectors with a non-zero z-coordinate, so each face of P is either parallel to the x, y-plane

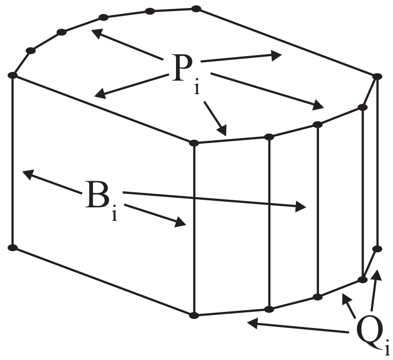


Figure 6.9: A convex polyhedron constructed from a solution to an instance of ECUVP.

or to the z-axis. Since P is non-degenerate, every vertex must be adjacent to a vector from B_i. Indeed, each vertex must be adjacent to exactly one; if a vertex v were adjacent to two, one going up and the other going down, then the corresponding faces would be coplanar (since our polyhedron must be convex).

We conclude that P must be a prism. See Figure 6.9 for a visualization of this polyhedron. That is, it corresponds of two identical polygons p_1 and p_2 parallel to the x, y-plane connected by z-axis-parallel edges at each vertex. But, since p_1 and p_2 are identical (and hence have identical edges) we can assume without loss of generality that the edges of p_1 are precisely P_i and the edges of p_2 are precisely Q_1. But now p_1 is a convex non-degenerate polygon on the edges in A, and therefore implies a solution to our instance of ECUVP, as required.

□

Degenerate Convex Polyhedra

As with our argument for degenerate polygons, we shall first consider orthogonal polyhedra.

Lemma 6.4.16. *It is NP-hard to determine whether a set of* 3*-dimensional vectors can form the edges of a (possibly degenerate) convex orthogonal polyhedron. This is true even if we remove the convexity requirement.*

Proof. We proceed in a manner similar to Lemma 6.4.15. However, the construction for the previous proof is not as simple to prove correct when degeneracies are allowed, since the resulting polyhedron need not necessarily be a prism. However, we can force creation of a prism by modifying the input to the reconstruction problem.

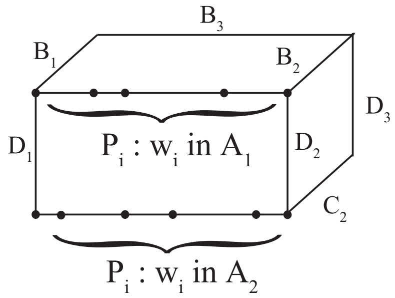


Figure 6.10: Forming a degenerate orthogonal polyhedron from an instance of PARTITION.

Suppose we are given an instance $(w_i)_{i=1}^n$ of PARTITION. Let $2S = \sum_{i=1}^n w_i$. Then we construct the following vectors.

1. $P_i = (0, w_i, 0)$ for each $1 \leq i \leq n$
2. $B_1 = B_2 = C_1 = C_2 = (1, 0, 0)$
3. $B_3 = C_3 = (0, S, 0)$
4. $D_1 = D_2 = D_3 = D_4 = (0, 0, 1)$

Suppose our instance of PARTITION has a solution. Then an orthogonal, convex, degenerate polyhedron can be constructed as in Figure 6.10. We basically group the P_i's into two sets, forming edge-constructs of length S, and form a $(1, 1, S)$ box.

For the reverse direction, suppose these vectors can form a convex orthogonal polyhedron P. Then certainly vector B_3 must be present in P. Since it is parallel to the y-axis of length S and the only other vectors with non-zero y-component are the P_i and C_3, we must be able to separate these vectors into 3 sets, each with total y-component at least S, by Lemma 6.2.3. The only way to do so is to take C_3 as one set, and partition the P_i into two sets, each with sum of w_i's being S. This implies a solution to PARTITION.

Note that the convexity requirement upon the polyhedron was not used to form our solution to PARTITION. Our argument implies that the reconstructed polyhedron *must* be convex, whether or not we required it to be so. Since a reconstructed convex orthogonal polyhedron is trivially an orthogonal polyhedron, we conclude that the NP-hardness result follows even if the convexity requirement is removed.

□

A polyhedron reconstructed from axis-aligned vectors must be orthogonal (since the cross-products of non-collinear pairs of these vectors, and hence face normals, will be axis-aligned as well). We can therefore drop the orthogonality constraint from the statement of Lemma 6.4.16, since it will be implied by an input consisting of orthogonal vectors. We therefore conclude the following result.

Lemma 6.4.17. *It is NP-hard to determine whether a set of* 3*-dimensional vectors can form the edges of a (possibly degenerate) polyhedron. The problem remains NP-hard when we require the polyhedron to be orthogonal and/or convex.*

Putting together all these results, plus Lemma 6.4.14, gives the following characterization.

Theorem 6.4.18. *The problem of determining whether a set of* 3*-dimensional vectors can form the edges of a polyhedron is NP-complete when degeneracies are allowed. The decision problem remains NP-complete even when we restrict the polyhedron to be convex or orthogonal.*

6.4.3 Open Problems

The complexity of reconstructing general and orthogonal non-degenerate polyhedra from edge vectors remains open. We have shown that these problems are in NP, but have not yet found a reduction to demonstrate that they are NP-complete.

The issue is that the proof of 6.4.15 relies upon the fact that the reconstructed polyhedron must be a prism. When the convexity requirement is removed, this is no longer true; consider Figure 3.3. Our approach of reducing the polyhedral version of the problem to two instances of the polygonal version is therefore not directly applicable.

6.5 Reconstructing from Edge Lengths

In this section we consider problems similar to reconstruction from edge vectors, but here we are given only a set of numbers and we are asked whether they can form the lengths of edges in a given construct (polygon or polyhedron). This problem can be thought of as being given a pile of sticks, each of fixed length, and asked whether they can be arranged to form the edges of a polygon or polyhedron. Unlike the edge vector reconstruction problem, the sticks can be arbitrarily rotated as well as translated.

6.5.1 Polygons

Here we are given a sequence of values $(l_i)_{i=1}^{n}$ and are asked whether they can form the lengths of edges of a given polygon. It is a well-known result that this question can be answered in linear

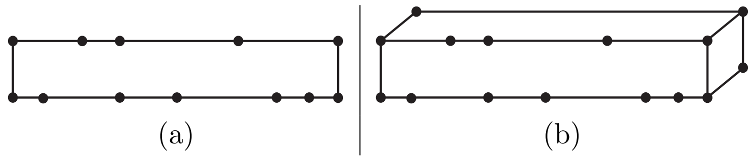


Figure 6.11: Reconstructing an orthogonal (a) polygon and (b) polyhedron from an instance of PARTITION.

time for general or convex polygons, either with or without degeneracies allowed.

Theorem 6.5.1 (Lemma 3.1 of [22])**.** *Given a sequence of values (l_i), one can determine in linear time whether they can form the lengths of the edges of a polygon, either necessarily convex or not.*

However, if we restrict to orthogonal polygons, this result is no longer true.

Lemma 6.5.2. *It is NP-hard to determine whether a sequence of values can form the edge lengths of an orthogonal (possibly degenerate) polygon.*

Proof. Suppose we are given an instance of PARTITION, (w_i). We provide the following values to our decision problem: $(3w_1, \ldots, 3w_n, 1, 1)$. We claim that these values can form the lengths of the edges of an orthogonal polygon if and only if there is a solution to our instance of PARTITION.

If a solution A_1, A_2 to PARTITION exists, we can use it to form a polygon. Simply form a box with height 1 and width $3S$ by taking all edges corresponding to A_1 and placing them collinearly horizontal, and similarly for the edges corresponding to A_2. The vertical sides of the box are formed by the edges of length 1. See Figure 6.11.

If a polygon P can be formed, we simply traverse the edges of the polygon to partition the vertical edges into two sets V_1, V_2 with the same sum of lengths, and similarly for the horizontal edges into H_1, H_2. In order for the sums to be equal, the two edges of length 1 must either be both horizontal or both vertical, and in opposite sides of the corresponding partition. Thus they can be removed from our partitions and equality still holds, so we have partitioned our values into four sets H_1, H_2, V_1, V_2 such that $V_1 = V_2$ and $H_1 = H_2$. Thus take $A_1 = V_1 \cup H_1$, $A_2 = V_2 \cup H_2$ to get a solution to PARTITION.

□

However, as with edge vectors, there is a pseudo-polytime algorithm to reconstruct an orthogonal degenerate polygon from edge lengths.

Lemma 6.5.3. *There is a pseudo-polynomial time algorithm that will determine if a (possibly degenerate) orthogonal polygon can be reconstructed from a multiset of edge lengths.*

Proof. An orthogonal polygon can be constructed with given edge lengths if and only if there is an associated multiset of orthogonal vectors with the appropriate lengths that can be used to reconstruct an orthogonal polygon. But by Lemma 6.2.1, this occurs if and only if a convex orthogonal polygon can be constructed. We can therefore limit ourselves to the problem of reconstructing a convex orthogonal polygon with degeneracies. Such a polygon is a box with edges possibly subdivided.

If such a polygon exists, it must have a vector associated with each given edge length. For edge length l_i, the associated vector must be one of $(l_i, 0)$,$(-l_i, 0)$,$(0, l_i)$, or $(0, -l_i)$. Our approach is to use dynamic programming, where for each i exactly one of the above vectors must be chosen, and we wish to determine if a final vector sum of $(0, 0)$ can be reached.

Formally, suppose our input lengths are $(l_i)_{i=1}^{n}$. Let our pseudo-polynomial parameter be $m = \sum_{i=1}^{n} l_i$. Recall that a pseudo-polynomial parameter is a value that is polynomial in the input size when input values are represented in unary, but exponential in the input size when values are represented in binary. We use m as a parameter in the time complexity of our algorithm.

We proceed by filling a three-dimensional table of values. The entries of this table are denoted $e(i, j, k)$, where $1 \le i \le n$, $-m \le j \le m$, $-m \le k \le m$. Our algorithm will fill this table so that $e(i, j, k)$ is 1 precisely when there are orthogonal vectors with lengths $l_1, \ldots, l_i$ that sum to (j, k).

Our initial setting is $e(1, 0, l_1) = e(1, 0, -l_1) = e(1, l_1, 0) = e(1, -l_1, 0) = 1$, and all other $e(1, j, k)$ are set to 0. Then for each i (incrementally), we set $e(i, j, k) = 1$ if and only if one of $e(i-1, j-l_i, k), e(i-1, j+l_i, k), e(i-1, j, k-l_i)$, or $e(i-1, j, k+l_i)$ is 1. When the entire table is filled in this way, the relevant entry is $e(n, 0, 0)$. If $e(n, 0, 0) = 1$ then we return true, otherwise return false.

The total runtime of this algorithm is $O(nm^2)$, as there are $4nm^2$ entries in the table. As justification for this algorithm, note that if indeed we have $e(n, 0, 0) = 1$ then there is an assignment of orthogonal vectors to each length that sum to $(0, 0)$; we simply place these in the order specified by the standard traversal of a polygon to obtain a box. More specifically, we place all vectors of the form $(0, l_i)$, then $(l_i, 0)$, then $(0, -l_i)$, and finally $(-l_i, 0)$. If $e(n, 0, 0) = 0$, then there is no such box, and hence by Lemma 6.2.1 there can be no orthogonal polygon constructed with the given edge lengths. □

When the possibility of degeneracies is removed, the problem remains NP-hard. However, the question of whether or not there is a pseudo-polytime algorithm is left open.

Lemma 6.5.4. *It is NP-hard to determine whether a sequence of values can form the edge lengths of an orthogonal, non-degenerate polygon.*

Proof. The proof is very similar to that for Lemma 6.4.11 (non-degenerate orthogonal polygons reconstructed from edge vectors). In particular, we require a Turing reduction with three separate cases. See Figure 6.8 for an illustration.

Given an instance (w_i) of PARTITION, we supply values $(2nw_1, \ldots, 2nw_n, a_1, \ldots, a_n, b, k)$ where $a_i = 1$ for each i, $b = 1$, and k is one of $1, 2, 3$. Note that the numerical complexity of the input is increased by a factor of n, but the input size is still polynomial in the size of the PARTITION instance. We now claim that an orthogonal, non-degenerate polygon can be constructed with these edge lengths if and only if there is a solution to our instance of PARTITION.

If a solution to this instance of PARTITION exists, we form a polygon in the same way as in the edge vector case. See Figure 6.8. We use vertical edges of length a_i to offset the horizontal edges of length v_i. The edge of length b forms one vertical end our our pseudo-rectangle, and the edge of length k forms the other.

We need multiple cases for k in order to cover the parity cases of the solution sets A_1 and A_2 for PARTITION. We omit the details of this argument, as they are identical to those in Lemma 6.4.11.

If a polygon P can be formed, a solution to PARTITION is implied by a similar argument to that for Lemma 6.5.2. That is, we split the horizontal and vertical edges into those that have positive direction and those that have negative direction in the standard traversal. Since all of the a_i, b, and k sum to less than $2n$, and all the w_i are multiples of $2n$, we must have that the positively-directed horizontal w_i edges must have total sum equal to that of the negatively-directed edges. A similar result holds for the vertically oriented edges. These two partitions can be combined to form a partition of the values w_i into two subsets with equal sum, as required. □

Lemma 6.5.5. *The problem of determining whether an orthogonal polygon can be reconstructed with given edge lengths is in NP. This is true whether or not collinear edges are allowed.*

Proof. The thing to note is that an orthogonal polygon with integer-length edges will have all vertices on lattice points. Thus, as a certificate, we can simply provide the line segments corresponding to each edge in a lattice-aligned embedding of a reconstructed polygon. We would provide these line segments in standard order.

To verify the certificate, we verify that the edges are all orthogonal, have the required lengths (which requires only subtraction as the edges are all orthogonal), form a closed connected chain, and do not intersect other than at links of the aforementioned chain. These are all trivial to verify except the last. However, there are polytime algorithms to compute the intersection points of line segments, so one of these can be used. If degeneracies are not allowed, it remains only to check that no two adjacent edges (in our chain; the order given in the standard order of the certificate) are both horizontal or both vertical. □

Putting these lemmas together, we get the following characterization for the orthogonal variant of this reconstruction problem.

Theorem 6.5.6. *The problem of determining whether an orthogonal polygon can be reconstructed from a given sequence of edge lengths is NP-complete. The problem is weakly NP-complete when collinear adjacent edges are allowed.*

6.5.2 Orthogonal Polyhedra

In this section we consider the case of constructing degenerate orthogonal polyhedra from edge lengths. We analyze this case by using a technical lemma from earlier in the chapter. The problem for non-degenerate orthogonal polyhedra is still open.

Theorem 6.5.7. *The problem of determining whether a sequence of values can form the edge lengths of an orthogonal (possibly degenerate) polyhedron is NP-complete.*

Proof. To show that the problem is in NP, we simply provide the constructed polyhedron as a certificate, in a standard data structure (such as the winged-edge data structure). Since the polyhedron is orthogonal and has integer length edges, the polyhedron can be embedded such that all vertices occur at lattice points. Then this certificate can be verified just as in Lemma 6.4.13.

For NP-hardness we reduce from PARTITION. Given an instance $(w_i)_{i=1}^n$ of PARTITION, supply the following lengths to our reconstruction problem:

1. $10w_1, \ldots, 10w_n$
2. $a_i = 1$ for $1 \leq i \leq 8$
3. $b_1 = b_2 = 10S$.

If a solution to our instance of PARTITION exists, we form our polyhedron as a box with dimensions $10S$, 1 and 1. Use the partition implied by the solution to form two of the edges of length $10S$, with b_1 and b_2 giving the other two. See Figure 6.11(b) for an illustration of this construction.

Suppose that an orthogonal polyhedron can be constructed from the given edge lengths. It must have some edge of length $b_i = 10S$. Then there must be three other disjoint sets of edges with lengths summing to $10S$, by Lemma 6.2.3. The only way for this to happen is to have one set contain b_2, then partition the v_i's into two sets each of total length $10S$. Note that a_i could be part of these sets, but are not large enough to allow omission of some v_i. Thus a solution to PARTITION is implied by our partition of the v_i's.

□

6.5.3 Degenerate Polyhedra

We now release the restriction of orthogonality from our polyhedra, but still retain the possibility of degeneracy. It is not yet known whether this problem is in NP; we discuss this issue in Section 6.5.4. However, we can show that this problem is NP-hard.

Theorem 6.5.8. *It is NP-hard to determine whether a sequence of values can form the edges lengths of a (possibly degenerate) polyhedron. The problem remains NP-hard when we require the polyhedron to be convex.*

Proof. Suppose we have an instance $(v_i)_{i=1}^n$ of PARTITION. We provide the following lengths.

1. $10v_1, \ldots, 10v_n$
2. $a = 20S - 5$
3. $b_1 = b_2 = 10S - 4$
4. $c = 1$.

Suppose our instance of PARTITION has a solution A_1, A_2. Then we form a 4-faced polyhedron, as illustrated in Figure 6.12. Place an edge of length a. Now form a triangle adjacent to a using b_1 as one side and the edges of lengths in A_1 arranged collinearly as the other side. Form another such triangle adjacent to a using b_2 and the lengths in A_2, such that b_1 and b_2 are adjacent. Finally, rotate these triangles about a such that their vertices not incident with a are precisely 1 from each other; use the edge of length $c = 1$ to connect them.

Now suppose that a polyhedron can be formed with our given edge lengths. Such a polyhedron must have an edge of length a. Reorient the polyhedron such that that edge is parallel to the x-axis. Then by Lemma 6.2.2, there are two disjoint sets B_1, B_2 of edges such that

$$\sum_{e \in B_i} |v(e).x| \geq a$$

and hence

$$\sum_{e \in B_i} |v(e)| \geq 20S - 5.$$

The only way in which this could happen is that one set includes b_1 and a subset of the $10v_i$'s that sum to $10S$, and the other face includes b_2 and a disjoint set of the v_i's that also sums to $10S$. This is because b_1 and b_2 together are less than $20S - 5$, so the values $10v_i$ in each set must sum to at least $20S - 5 - b_1 - c = 10S - 2$. Hence they must sum to at least $10S$, since they

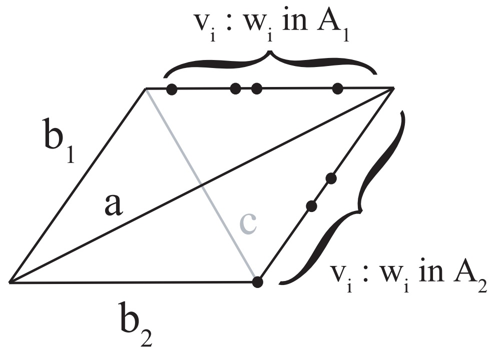


Figure 6.12: Forming a degenerate tetrahedral polyhedron from an instance of Partition. The lightened edge of length c is in the background.

are all multiples of 10. But then this partition of the values $10v_i$ across our two sets implies a solution to our instance of Partition, as required.

Note that we do not use convexity in our argument, so indeed the NP-hardness result remains whether or not we require the constructed polyhedron be convex. □

6.5.4 Open Problems

A number of problems are left open by our research. First, we have not shown that reconstructing polyhedra from edge lengths is in NP for non-orthogonal polyhedra. The main issue is that while edge lengths may be easy to represent (i.e. integers), the orientations of the corresponding edges may have very high complexity. For example, an edge of unit length could be oriented in a polyhedron so that its vertex endpoints are irrational. A polyhedron can always be reoriented so that that particular edge has easily representable endpoints (such as $(0, 0, 0)$ and $(0, 0, 1)$), but such an operation cannot be applied independently to all edges. It seems that in order to show that this problem lies in NP, one would need to find a manner of representing a polyhedron that does not depend upon the complexity of edge orientations.

Also, we have not determined whether the problem of reconstructing a general, non-degenerate polyhedron from a set of edge lengths is NP-hard. The issue with proving this result appears to be that a non-degenerate polyhedron will tend to have many edges, but so far we have no good way to restrict the placement of these edges. The additional edges needed to allow non-degeneracy interfere with any arguments regarding equivalence to a solution to Partition. It would seem that a completely different approach will be required to resolve this particular variant of the edge length reconstruction problem.

			Edge Vectors	**Edge Lengths**
Polygons	Degenerate	Convex	Weakly NP-Complete	Polynomial
		Orthogonal	Weakly NP-Complete	Weakly NP-Complete
		General	Weakly NP-Complete	Polynomial
	Non-Degenerate	Convex	Weakly NP-Complete	Polynomial
		Orthogonal	NP-Complete	NP-Complete
		General	NP-Complete	Polynomial
Polyhedra	Degenerate	Convex	NP-Complete	NP-Hard
		Orthogonal	NP-Complete	NP-Complete
		General	NP-Complete	NP-Hard
	Non-Degenerate	Convex	NP-Complete	?
		Orthogonal	NP	NP
		General	NP	?

Table 6.1: Summary of the computational complexity results presented in this chapter

6.6 Summary

We have analyzed the computational complexity of various reconstruction problems. Table 6.1 summarizes the results presented in this chapter. As shown in that table, many questions still remain open. For those problems that were shown to be NP-hard, the question still remains as to whether they are in NP and whether they admit pseudo-polytime algorithms. Some variants involving polyhedra were not analyzed at all; the computational complexity of these problems is still completely open.

Chapter 7

Conclusion

The work in this thesis was split into two categories: unfolding and reconstruction. To maintain a degree of symmetry, we shall split our conclusion in the same way.

Unfolding. One of our main contributions is the construction of the asymmetrical 3-pointed star $P_3^\phi(\alpha, \beta)$: an ununfoldable polyhedron with 9 convex faces. This immediately implies an open question: is 9 the smallest possible? We think that it is not, if only because of the intuitive simplicity of our constructed example. It is likely that some trick or another may be used to remove some faces from the 3-pointed star while retaining the properties that imply its ununfoldability. However, we feel that a tight bound is being approached: the seemingly key properties of ununfoldability, negative curvature at many vertices and very high curvature at others, simply cannot be realized in a polyhedron with too few faces. While the problem of finding an ununfoldable convex-faced polyhedron with fewer than 9 faces is interesting, of much more significance would be a proof that a certain number of faces is the true minimum, for such a proof would (hopefully) shed more light on the precise properties of polyhedra that imply ununfoldability.

This thesis also showed that convex-faced polyhedra cannot avoid overlaps caused by having too much material around a single vertex. A polyhedron was constructed such that every unfolding has a vertex with face angle greater than 2π. It was further shown that these sorts of overlaps *can* be avoided for all genus-0 polyhedra when one is allowed to cut into faces. An interesting avenue of future research is thus to determine precisely what types of overlaps can and cannot be avoided with edge-unfoldings and with general unfoldings.

The thesis then turned to convex polyhedra. Our study of unfoldings of convex polyhedra was motivated by Shephard's conjecture that all convex polyhedra can be edge-unfolded without overlap. Others have strengthened this conjecture to claim that specific unfolding methods will avoid overlap. Our contribution was to disprove some of these conjectures for specific unfolding methods. In particular, we formally constructed a convex polyhedron for which cutting along a certain minimum path tree creates an overlap, disproving a conjecture by Fukuda. We then constructed a convex polyhedron for which cutting along *any* minimum path tree creates an overlap. Next, we constructed a polyhedron for which any steepest edge cut tree creates an overlapping unfolding, disproving a conjecture by Schlickenrieder. Finally, we constructed a polyhedron for which every normal order unfolding contains an overlap.

All of our example convex polyhedra were constructed by appealing to a particular type of overlap, which we called a 2-local overlap. The next step in this line of research is to finish an analysis of the avoidability of 2-local overlaps. Can every convex polyhedron be edge-unfolded so that no 2-local overlaps occur? Or is there a convex polyhedron for which every edge-unfolding contains a 2-local overlap? Such an example would disprove Shephard's conjecture, whereas an edge-unfolding method that avoids 2-local overlaps might be an important step towards proving Shephard's conjecture.

Reconstruction. In the research on reconstruction we consider two classes of reconstruction problems: reconstructing from edge orientations and reconstructing from edge lengths. In the first, one is given a collection of edges in space, and must determine whether they can be rearranged *by translation only* to form a polygon or polyhedron. This problem was shown to be NP-complete in almost all cases: whether we discuss polygons or polyhedra, and even when additional constraints such as convexity, orthogonality, and (in the case of polygons) non-degeneracy are introduced. The complexity remains open for polyhedra that are required to be non-degenerate and not required to be convex. We additionally showed that some variants of this problem for polygons (namely when degeneracy is allowed or convexity is required) are weakly NP-complete by providing pseudo-polytime algorithms to solve them. Determining whether the remaining variants of this problem are strongly or weakly NP-hard remains open.

The second problem, reconstructing polyhedra from edge lengths, is the more classical of the two and is what motivated our research on reconstruction problems. In this reconstruction problem, one is given a collection of edges in space and must determine whether they can be rearraged by translation and/or rotation to form a polygon or polyhedron. It was already known that polynomial algorithms exist to determine whether general or convex polygons can be reconstructed from such information. We demonstrated that this problem is NP-complete for orthogonal polygons. For polyhedra, we demonstrated that if degeneracies (coplanar adjacent faces or degenerate faces) are allowed then the problem is NP-hard. However, many open problems remain. The complexity of this reconstruction problem is not known when the polyhedra are constrained to be non-degenerate. Also, for cases where the polyhedron is not required to be orthogonal, it is not even known if the problem is in NP. More research will be necessary before we fully understand the complexity of this long-standing decision problem.

Bibliography

[1] P. K. Agarwal, B. Aronov, J. O'Rourke, C. A. Schevon. Star Unfolding of a Polytope with Applications. *SIAM Journal on Computing*, 26(6):1689–1713.

[2] A. D. Alexandrov. *Convex Polyhedra* [Translated from Russian]. Springer Monographs in Mathematics, Berlin, 2005. Russian edition originally published by Gosudarstv. Izdat. Takhn.-Teor. Lit., Moscow-Leningrad, 1950.

[3] B. Aronov and J. O'Rourke. Nonoverlap of the star unfolding. *Discrete Computational Geometry*, 8:219–250, 1992.

[4] M. Bern, E. D. Demaine, D. Eppstein and E. Kuo. Ununfoldable polyhedra. In *Proceedings of the 11th Canadian Conference on Computational Geometry*, Vancouver, B.C., August 15-19, 1999.

[5] M. Bern, E. D. Demaine, D. Eppstein, E. Kuo, A. Mantler and J. Snoeyink. Ununfoldable polyhedra with convex faces. *Computational Geometry: Theory and Applications.* 24(2):51–62, 2003.

[6] T. Biedl, E. Demaine, M. Demaine, A. Lubiw, M. Overmars, J. O'Rourke, S. Robbins, and S. Whitesides. Unfolding some classes of orthogonal polyhedra. In *Proceedings of the 10th Canadian Conference on Computational Geometry*, Montréal, Canada, August 1998.

[7] T. Biedl, A. Lubiw, and J. Sun. When can a net fold to a polyhedron? In *Proceedings of the 11th Canadian Conference on Computational Geometry*, Vancouver, Canada, August 1999.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms* (Second Edition), McGraw-Hill, 2001.

[9] P. R. Cromwell. *Polyhedra.* Cambridge University Press, Cambridge, 1997.

[10] E. Demaine. Folding and Unfolding. Ph. D. Thesis, University of Waterloo, 2001.

[11] M. Damian, R. Flatland, H. Meijer, and J. O'Rourke. Unfolding well-separated orthotrees. *15th Annual Fall Workshop on Computational Geometry*, University of Pensylvania, November 2005. [http://www.research.att.com/˜krishnas/fwcg05/abstracts/6.pdf].

[12] M. Damian, R. Flatland, and J. O'Rourke. Unfolding Manhattan Towers. In *Proceedings of the 17th Canadian Conference on Computational Geometry*, 211–214, Windsor, Ontario, August 2005.

[13] E. Demaine, M. Demaine, A. Lubiw, J. O'Rourke, and I. Pashchenko. Metamorphosis of the cube. In *Proceedings of the 15th Annual ACM Symposium on Computational Geometry*, 409–410, 1999. Video and abstract.

[14] E. D. Demaine, D. Eppstein, J. G. Erickson, G. W. Hart, and J. O'Rourke. Vertex-unfoldings of simplicial manifolds. *Proceedings of the 18th Annual ACM Symposium on Computational Geometry*, 237–243, 2002.

[15] E. D. Demaine and J. Erickson. *Open Problems on Polytope Reconstruction.* Manuscript, July 1999. [http://theory.csail.mit.edu/˜edemaine/papers/PolytopeReconstruction/].

[16] E. D. Demaine, J. Iacono, and S. Langerman. Grid vertex-unfolding orthostacks. In *Revised Selected Papers from the Japan Conference on Discrete and Computational Geometry*, Lecture Notes in Computer Science, volume 3742, 76–82, Tokyo, Japan, October 2004.

[17] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* Freeman, 1979.

[18] K. Fukuda. Strange Unfoldings of Convex Polytopes, 1997. World Wide Web, available at [http://www.ifor.math.ethz.ch/˜fukuda/unfold_home/unfold_open.html].

[19] B. Grünbaum. A starshaped polyhedron with no net. *Geombinatorics*, 11:43–48, 2001.

[20] B. Grünbaum. No-net polyhedra. *Geombinatorics*, 11:111–114, 2002.

[21] P. Hoffman. Barebones Algebraic Topology. Course Notes in Algebraic Topology, University of Waterloo, 2004.

[22] W. J. Lenhart and S. H. Whitesides. Reconfiguring closed polygonal chains in Euclidean d-space. *Discrete Comput. Geom.*, 13:123–140, 1995.

[23] A. Lubiw and J. O'Rourke. When can a polygon fold to a polytope? Technical Report 048, Smith College, 1996.

[24] M. Namiki and K. Fukuda. Unfolding 3-dimensional convex polytopes: A package for Mathematica 1.2 or 2.0. Mathematica Notebook, Univ. of Tokyo, 1993.

[25] M. Namiki and K. Fukuda. *UnfoldPolytope: A Mathematica package for unfolding general convex polytopes,* 1992. World Wide Web, available at [ftp://ftp.ifor.math.ethz.ch/pub/fukuda/mathematica].

[26] J. O'Rourke. *Computational Geometry in C* (Second Edition). Cambridge University Press, 1998.

[27] J. O'Rourke. Folding and Unfolding in Computational Geometry. Lecture Notes In Computer Science, Vol. 1763, 258–266, Springer-Verlag, Berlin, 2000.

[28] J. Pach and R. Wenger. Embedding planar graphs at fixed vertex locations. In *Graphs Combin.*, 17:717–728, 2001.

[29] J. J. Rotman. *An Introduction to Algebraic Topology.* Springer-Verlag, New York-Berlin, 1988.

[30] W. Schlickenrieder. Nets of Polyhedra. Master's Thesis, Technische Universität Berlin, June 1997.

[31] M. Sharir and A. Schorr. On shortest paths in polyhedral spaces. *SIAM J. Comput.*, 15:193–215, 1986.

[32] G. C. Shephard. Convex polytopes with convex nets. *Mathematical Proceedings of the Cambridge Philosophical Society*, 78:389–403, 1975.

[33] M. Spriggs. Personal Communication, 2005.

[34] A. S. Tarasov. Polyhedra that do not admit natural unfoldings. [In Russian] Uspekhi Mat. Nauk 54, no. 3(327):185–186, 1999. Translation: Polyhedra with no natural unfoldings. Russian Math. Surveys 54, 3:656–657, 1999.