When Can You Fold a Map?

Esther M. Arkin[1], Michael A. Bender[2], Erik D. Demaine[3], Martin L. Demaine[3], Joseph S. B. Mitchell[1], Saurabh Sethia[2], and Steven S. Skiena[2]

[1] Department of Applied Mathematics and Statistics, State University of New York, Stony Brook, NY 11794-3600, USA, email: {estie, jsbm}@ams.sunysb.edu.
[2] Department of Computer Science, State University of New York, Stony Brook, NY 11794-4400, USA, email: {bender, saurabh, skiena}@cs.sunysb.edu.
[3] Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada, email: {eddemaine, mldemaine}@uwaterloo.ca.

Abstract. We explore the following problem: given a collection of creases on a piece of paper, each assigned a folding direction of mountain or valley, is there a flat folding by a sequence of simple folds? There are several models of simple folds; the simplest *one-layer simple fold* rotates a portion of paper about a crease in the paper by $\pm 180°$. We first consider the analogous questions in one dimension lower—bending a segment into a flat object—which lead to interesting problems on strings. We develop efficient algorithms for the recognition of simply foldable 1-D crease patterns, and reconstruction of a sequence of simple folds. Indeed, we prove that a 1-D crease pattern is flat-foldable by any means precisely if it is by a sequence of one-layer simple folds.

Next we explore simple foldability in two dimensions, and find a surprising contrast: "map" folding and variants are polynomial, but slight generalizations are NP-complete. Specifically, we develop a linear-time algorithm for deciding foldability of an orthogonal crease pattern on a rectangular piece of paper, and prove that it is (weakly) NP-complete to decide foldability of (1) an orthogonal crease pattern on a orthogonal piece of paper, (2) a crease pattern of axis-parallel and diagonal (45-degree) creases on a square piece of paper, and (3) crease patterns without a mountain/valley assignment.

1 Introduction

> The easiest way to refold a road map is differently.
> — Jones's Rule of the Road (M. Gardner [6])

Perhaps the best-studied problem in origami mathematics is the characterization of flat-foldable crease patterns. A crease pattern is a straight-edge embedding of a graph on a polygonal piece of paper; a flat folding must fold along all of the edges of the graph, but no more. For example, two crease patterns are shown in Figure 1. The first one folds flat into a classic origami crane, whereas the second one cannot be folded flat (unless the paper is allowed to pass through itself), even though every vertex can be "locally" flat folded.

The algorithmic version of this problem is to determine whether a given crease pattern is flat-foldable. The crease pattern may also have a direction of "mountain" or

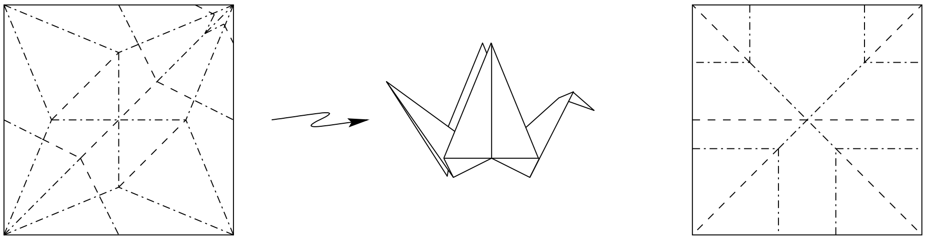

Fig. 1. Sample crease patterns. Left: the classic crane. Right: pattern of Hull [9], which cannot be folded flat, for any mountain-valley assignment.

"valley" assigned to each crease, which restricts the way in which the crease can be folded. (Our figures adhere to the standard origami convention that valleys are drawn as dashed lines and mountains are drawn as dot-dashed lines.)

It is known that the general problem of deciding flat foldability of a crease pattern is NP-hard (Bern and Hayes [2]). In this paper, we consider the important and very natural case of recognizing crease patterns that arise as the result of flat foldings using *simple foldings*. In this model, a flat folding is made by a sequence of *simple folds*, each of which folds one or more layers of paper along a single line segment. As we define in Section 2, there are different types of simple folds (termed "one-layer," "some-layers," and "all-layers"), depending on how many layers of paper are required or allowed to be folded along a crease.

Note that not every flat folding can be achieved by a simple folding. For example, the crane in Figure 1 (top) cannot be made by a simple folding. Also, the hardness gadgets of [2] require nonsimple folds.

The problem we study in this paper is that of determining whether a given crease pattern (usually with specified mountain and valley assignments) can be folded flat by a sequence of simple folds, and if so, to construct such a sequence of folds.

Several of our results are based on the special case in which the creases in the piece of paper are all parallel to one another. This case can be seen to be equivalent to a *one-dimensional* folding problem of folding a line segment ("paper") according to a set of prescribed crease points (possibly labeled "mountain" or "valley"). We will therefore refer to this special case, which has a rich structure of its own, as the "1-D" case to distinguish it from the general 2-D problem. In contrast to the 2-D problem, we show that 1-D flat foldability is equivalent to 1-D simple foldability.

Motivation. In addition to its inherent interest in the mathematics of origami, our study is motivated by applications in sheet metal and paper product manufacturing, where one is interested in determining if a given structure can be manufactured using a given machine. (See references cited below.) While origamists can develop particular skill in performing nonsimple folds to make beautiful artwork, practical problems of manufacturing with sheet goods require simple and constrained folding operations. Our goal is to develop a first suite of results that may be helpful towards a fuller algorithmic understanding of the several manufacturing problems that arise, e.g., in making three-dimensional cardboard and sheet-metal structures.

Related Work. Our problem is related to the classic combinatorics question of *map folding* [15]. This question asks for the *number* of foldings of a particular crease pattern, namely an $m \times n$ rectangular grid, by a sequence of simple folds. See also the discussion in Gardner's book [6]. In contrast with this combinatorics question, we study the algorithmic complexity of the decision problem, also in some more general instances of crease patterns.

The mathematical and algorithmic problems arising in the study of flat origami have been examined by several researchers, e.g., Hull [9], Justin [10], Kawasaki [12], and Lang [13]. Of particular relevance to our work is the paper of Bern and Hayes [2], in which the general problem of deciding flat foldability of a crease pattern is shown to be strongly NP-hard. Demaine et al. [4] used computational geometry techniques to show that any polygonal (connected) silhouette can be obtained by simple folds from a rectangular piece of paper.

There has been quite a bit of work on the related problems of manufacturability of sheet metal parts (see, e.g., [21]) and folding cartons (see, e.g., [14]). Existing CAD/CAM techniques (including BendCad and PART-S) rely on worst-case exponential-time state space searches (using the A^* algorithm). In general, the problem of bend sequence generation is a challenging (and provably hard [1]) coordinated motion planning problem. For example, Lu and Akella [14] utilize a novel configuration-space formulation of the folding sequence problem for folding cartons using fixtures; their search, however, is still worst-case exponential time. Our work differs from the prior work on sheet metal and cardboard bending in that the structures we are folding are ultimately "flat" in their folded states (all bend angles in the input crease pattern are $\pm 180°$, according to a mountain-valley assignment that is part of the input crease pattern). Also, we are concerned only with the feasibility of the motion of the (stiff) material that is being folded – does it collide with itself during the folding motion? We are not addressing here the issues of reachability by the tools that perform the folding. As we show, even with the restrictions that come with the problems we study, there is a rich mathematical and algorithmic theory of foldability.

Summary of Our Results.[1]

(1) We analyze the 1-D one-layer and some-layers cases, giving a full characterization of flat-foldability and an $O(n)$ algorithm for deciding foldability and producing a folding sequence, if one exists.
(2) We analyze the 1-D all-layers case as a "string folding" problem. In addition to a simple $O(n^2)$ algorithm, we give an algorithm utilizing suffix trees that requires time linear in the bit complexity of the input, and a randomized algorithm with expected $O(n)$ running time.
(3) We give a linear-time algorithm for deciding foldability of orthogonal crease patterns on a rectangular paper (the "map folding problem"), in the one-, some-, and all-layers cases, based on our 1-D results.

[1] Due to space limitations, many of the proofs and details are omitted from this extended abstract. The reader is referred to the full paper.

(4) We prove that it is (weakly) NP-complete to decide foldability of an orthogonal crease pattern on a piece of paper that is more general than a rectangle: a simple orthogonal polygon.
(5) We also prove that it is (weakly) NP-complete to decide foldability of a rectangular piece of paper with a crease pattern that includes *diagonal* creases (at 45-degrees), in addition to axis-parallel creases.
(6) We show that it is (weakly) NP-complete to decide foldability of a orthogonal piece of paper having a crease pattern for which no mountain-valley assignment is given.

2 Definitions

We are concerned with foldings in one and two dimensions. A one-dimensional piece of paper is a (line) *segment* in $\mathbf{R}^1$. A two-dimensional piece of paper is a (connected) *polygon* in $\mathbf{R}^2$, possibly with holes. In both cases, the paper is folded through one dimension higher than the object; thus, segments are folded through $\mathbf{R}^2$ and polygons are folded through $\mathbf{R}^3$. *Creases* have one less dimension; thus, a crease is a point on a segment and a line segment on a polygon.

A *crease pattern* is a collection of creases on the piece of paper, no two of which intersect except at a common endpoint. A *folding* of a crease pattern is an isometric embedding of the piece of paper, bent along every crease in the crease pattern (and not bent along any segment that is not a crease). In particular, each facet of paper must be mapped to a congruent copy, the connectivity between facets must be preserved, and the paper cannot pass through itself. See Figure 2.

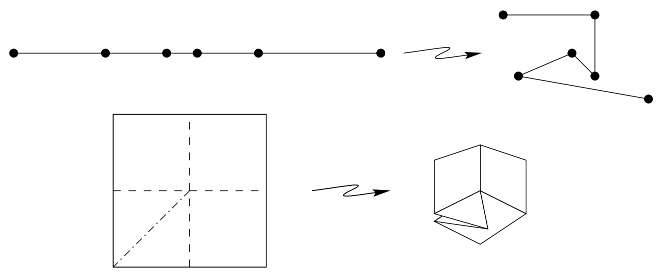

Fig. 2. Sample nonflat foldings in one and two dimensions.

A *flat folding* has the additional property that it lies in the same space as the unfolded piece of paper. That is, a flat folding of a segment lies in $\mathbf{R}^1$, and a flat folding of a polygon lies in $\mathbf{R}^2$. In reality, there can be multiple layers of paper at a point, so the folding really occupies a finite number of infinitesimally close copies of $\mathbf{R}^1$ or $\mathbf{R}^2$. More formally, a flat folding can be specified by a function mapping the vertices to their folded positions, together with a partial order of the facets of paper that specifies their overlap order [2, 9, 13].

If we orient the piece of paper to have a top and bottom side, we can talk about the *direction* of a crease in a flat folding. A *mountain* brings together the bottom sides of the two adjacent facets of paper, and a *valley* brings together the top sides. A *mountain-valley assignment* is a function from the creases in a crease pattern to $\{M, V\}$. Together, a crease pattern and a mountain-valley assignment form a *mountain-valley pattern*.

This paper is concerned with the following generic question.

Problem 1. Simple Folding: Given a mountain-valley pattern, is there a simple folding satisfying the specified mountains and valleys? If so, construct such a simple folding.

There are three natural versions of this problem, depending on the type of "simple folds" allowed. In general, a *simple folding* is a sequence of simple folds. Each simple fold takes a flat-folded piece of paper, and folds it into another flat folding using additional creases. There are three types of simple folds: one-layer, all-layers, and some-layers.

A *one-layer simple fold* f is a crease on the folded piece of paper, together with a direction. If we look at the unfolded piece of paper, then f partitions it into two parts, call them A and B. Performing f corresponds to rotating A about f by $\pm 180°$, where $\pm$ depends on the fold direction, if this does not cause the paper to cross itself. This makes just a single crease, which is what we mean by folding one layer of paper.

An *all-layers simple fold* f is also a crease together with a direction. Now we consider the partition of the flat folding (instead of the unfolded piece of paper) by f into two parts, call them A and B again. Performing f corresponds to rotating (all layers of) A about f by $\pm 180°$. This makes a crease through all of the layers of paper at f. Note that this type of fold can never cause the paper to cross itself.

Finally, a *some-layers simple fold* f is the most general. It takes some of the top [bottom] layers of A, and rotates them about f by $180°$ $[-180°]$, provided the paper does not cross itself.

3 1-D: One-Layer and Some-Layers

This section is concerned with the 1-D one-layer simple-fold problem. We will prove the surprising result that we only need to search for one of two local operations to perform. The two operations are called *crimps* and *end folds*, and are shown in Figure 3.

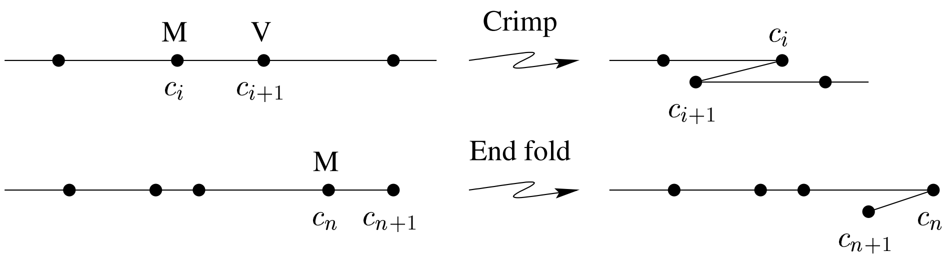

Fig. 3. The two local operations for one-dimensional one-layer folds.

More formally, let $c_1, \ldots, c_n$ denote the creases on the segment, oriented so that c_i is left of c_j for $i < j$. Let c_0 $[c_{n+1}]$ denote the left [right] end of the segment. Despite the

"c" notation (which is used for convenience), c_0 and c_{n+1} are not considered *creases*; instead they are called the *ends*.

First, a pair (c_i, c_{i+1}) of consecutive creases is *crimpable* if c_i and c_{i+1} have opposite directions and $|c_{i-1} - c_i| \geq |c_i - c_{i+1}| \leq |c_{i+1} - c_{i+2}|$. *Crimping* such a pair corresponds to folding c_i and then folding c_{i+1}, using one-layer simple folds.

Second, c_0 is a *foldable end* if $|c_0 - c_1| \leq |c_1 - c_2|$, and c_{n+1} is a *foldable end* if $|c_{n-1} - c_n| \geq |c_n - c_{n+1}|$. *Folding* such an end corresponds to performing a one-layer simple fold at the nearest crease (crease c_1 for end c_0, and crease c_n for end c_{n+1}).

We claim that one of the two local operations exists in any flat-foldable 1-D mountain-valley pattern. We claim further that an operation exists for any pattern satisfying a certain "mingling property." Specifically, a 1-D mountain-valley pattern is called *mingling* if for every sequence $c_i, c_{i+1}, \ldots, c_j$ of consecutive creases with the same direction, either (1) $|c_{i-1} - c_i| \leq |c_i - c_{i+1}|$; or (2) $|c_{j-1} - c_j| \geq |c_j - c_{j+1}|$. We call this the mingling property because, for maximal sequences of consecutive creases with the same direction, it says that there are folds of the opposite direction nearby. In this sense, the mountain-valley pattern is "crowded" (mountains and valleys must "mingle" together).

First we show (in the full paper) that mingling mountain-valley patterns include flat-foldable patterns:

Lemma 1. *Every flat-foldable 1-D mountain-valley pattern is mingling.*

Next we show (again, see the full paper) that having the mingling property suffices to imply the existence of a single crimpable pair or foldable end.

Lemma 2. *Any mingling 1-D mountain-valley pattern has either a crimpable pair or a foldable end.*

Ideally, we could show at this point that performing either of the two local operations preserves the mingling property, and hence a mountain-valley pattern is mingling precisely if it is flat-foldable. Unfortunately this is false; see the full paper. Instead, we must prove that flat foldability is preserved by each of the two local operations; i.e., if we treat the folded object from a single crimp as a new segment, it is flat-foldable.

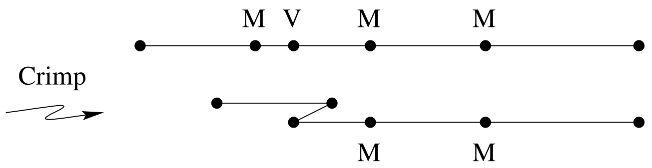

Fig. 4. A mingling mountain-valley pattern that when crimped is no longer mingling and hence not flat-foldable. Indeed, the original mountain-valley pattern is not flat-foldable.

Lemma 3. *Folding a foldable end preserves foldability.*

Lemma 4. *Crimping a crimpable pair preserves flat foldability.*

Combining all of the previous results, we have the following:

Theorem 1. *Any flat-foldable 1-D mountain-valley pattern can be folded by a sequence of crimps and end folds.*

A particularly interesting consequence of this theorem is the following:

Corollary 1. *The following are equivalent for a 1-D mountain-valley pattern P:*

1. *P has a flat folding.*
2. *P has a some-layers simple folding.*
3. *P has a one-layer simple folding.*

Finally, let us show (in the full paper) that Theorem 1 leads to a simple linear-time algorithm:

Theorem 2. *The 1-D one-layer and some-layers simple-fold problems can be solved in $O(n)$ worst-case time on a machine supporting arithmetic on the input lengths.*

4 1-D: All-Layers Simple Folds

The 1-D all-layers simple-fold problem can be cast as an interesting "string folding" problem. (This folding problem is not to be confused with the well-known protein/string folding problem in biology [3].) The input mountain-valley pattern can be thought of as a string of lengths interspersed with mountain and valley creases. Specifically, we will assume that the input lengths are specified as integers or equivalently rational numbers. (Irrational numbers can be replaced by close rational approximations, provided the sorted order of the lengths is preserved.)

Thus, an input string is of the form $d_0\ c_1\ d_1\ c_2 \cdots c_{n-1}\ d_{n-1}\ c_n\ d_n$, where each $c_i \in \{M, V\}$ and each d_i is a positive rational number. We call each c_i and d_i a *symbol* of the string. It will be helpful to introduce some more uniform notation for symbols. For a string S of length n, we denote the ith symbol by $S[i]$, where $1 \leq i \leq n$.

When we make an all-layers simple fold, we cannot "cover up" a crease except with a matching crease (which when unfolded is in fact the other direction), because otherwise this crease will be impossible to fold later. To formalize this condition, we define the *complement* of symbols in the string: $\mathrm{comp}(d_i) = d_i$, $\mathrm{comp}(M) = V$, and $\mathrm{comp}(V) = M$. Finally, we call a crease (M or V symbol) $S[i]$ *allowable* if $S[i-x] = \mathrm{comp}(S[i+x])$ for all $1 \leq x \leq \min(i-1, n-i)$, except that $S[1]$ and $S[n]$ (the ends) are allowed to be shorter than their complements.

Lemma 5. *A mountain-valley pattern can be folded by a sequence of all-layers simple folds precisely if there is an allowable fold, and the result after performing that fold has an allowable fold, and so on, until all creases of the segment have been folded.*

By Lemma 5, the problem of testing foldability reduces to repeatedly finding allowable folds in the string. Testing whether a fold at position i is allowable can clearly be done in $O(1 + \min\{i-1, n-i\})$ time, by testing the boundary conditions and whether $S[i-x] = \mathrm{comp}(S[i+x])$ for $1 \leq x \leq \min(i-1, n-i)$. Explicitly testing all creases in this manner would yield an $O(n^2)$-time algorithm for finding an allowable fold (if one exists). Repeating this $O(n)$ times results in a naive $O(n^3)$ algorithm for testing foldability.

This cubic bound can be improved by being a bit more careful. In $O(n^2)$ time, we can determine for each crease $S[i]$ the largest value of k for which $S[i-x] = \mathrm{comp}(S[i+x])$ for all $1 \leq x \leq k$. Using this information it is easy to test whether a crease $S[i]$ is allowable. After making one of these allowable folds, we can in $O(n)$ time update the value of x for each crease, and hence maintain the collection of allowable folds in linear time. This gives an overall $O(n^2)$ bound, which we now proceed to improve further.

We present two efficient algorithms for folding strings. The algorithm in Section 4.1 is based on suffix trees and runs in time linear in the bit complexity of the input. In Section 4.2, we use randomization to obtain a simpler algorithm that runs in $O(n)$ time.

4.1 Suffix-Tree Algorithm

In the full paper we prove the following:

Theorem 3. *A string S of length n can be tested for all-layers simple foldability, in time that is dominated by that to construct a suffix tree on S.*

The difficulty with the time bound is that sorting the alphabet seems to be required. Other than the time to sort the alphabet, it is possible to construct a suffix tree in $O(n)$ time [5]. To sort the alphabet in the comparison model, $O(n \log n')$ time suffices, where n' is the number of distinct input lengths. In particular, if the input lengths are encoded in binary, then the algorithm is linear in this bit complexity. On a RAM, the current state-of-the-art algorithm for integer sorting [20] uses $O(n(\log \log n)^2)$ time and linear space.

4.2 Randomized Algorithm

In this section we describe a simple randomized algorithm that solves the 1-D all-layers simple-fold problem in $O(n)$ time. There are two parts to the algorithm:

1. assigning labels to the input lengths so that two lengths are equal precisely if they have the same label; and
2. finding and making allowable folds.

The first part is essentially element uniqueness, and can be solved in linear expected time using hashing. For example, the dynamic hashing method described by Motwani and Raghavan [16] supports insertions and existence queries in $O(1)$ expected time. We can use this data structure as follows. For each input length, check whether it is already in the hash table. If it is not, we assign it a new unique identifier, and add it to the hash table. If it is, we use the existing unique identifier for that value (stored in the hash table). Let n' denote the number of distinct labels found in this process (or 2, whichever is larger).

For the second part, we will show that each performed fold can be found in $O(1 + r)$ time, where r is the number of creases removed by the discovered fold (in other words, the minimum length to an end of the segment to be folded). However, it is possible that the algorithm makes a mistake, and that some of the reported folds are not actually possible. Fortunately, mistakes can be detected quickly, and after $O(1)$ expected iterations the pattern will be folded. (Unless of course the pattern is not flat-foldable, in which case the algorithm reports this fact correctly.)

In the full paper we give details of the algorithm, and conclude:

Theorem 4. *The 1-D all-layers simple-fold problem can be solved in $O(n)$ expected time on a machine supporting random numbers and hashing of the input lengths.*

5 Orthogonal Simple Folds in 2-D

In this section, we generalize our results for 1-D simple folds to *orthogonal* 2-D crease patterns, which consist only of horizontal and vertical folds on a rectangular piece of paper, where horizontal and vertical are defined by the sides of the rectangular paper. In such a pattern, the creases must go all the way through the paper, because every vertex of a flat-foldable crease pattern has degree at least four [2, 9]. Hence, the crease pattern is a *map* or grid of creases. Recall from Section 3 that the opposite holds in 1-D: one-layer and some-layers folds are equivalent to general flat-foldability.

In this section we handle all three cases of simple folds: one-, some-, and all-layers folds. To know what time bounds we desire, we must first discuss encoding the input. A natural encoding of maps specifies the height of each row and the width of each column, thereby using $n_1 + n_2$ space for an $n_1 \times n_2$ grid. The mountain-valley assignment, however, requires $\Theta(n_1 n_2)$ space to specify the direction for each edge of the grid. Hence, our goal of linear time amounts to being linear in $n = n_1 n_2$.

It is easy to see that in exactly one of the two orientations, vertical or horizontal, there must be at least one crease line, all the way across the paper, that is entirely valley or mountain. (If there is no such crease, the pattern is unfoldable. And it cannot be that there is both a horizontal crease and a vertical crease all the way through the paper, since their intersection would be a vertex that is locally unfoldable.) Without loss of generality, assume it is horizontal. Let the set of these horizontal fold lines be $\mathcal{H}$.

We claim that all fold lines in $\mathcal{H}$ must be folded before any other fold. This is so because (1) folding along any vertical fold line v will lead to a mismatch of creases at the intersection of v with any unfolded elements of $\mathcal{H}$ and (2) horizontal folds not in $\mathcal{H}$ are not entirely mountain or valley and hence cannot be folded before some vertical fold is made. Thus we have a corresponding 1-D problem (one-, some- or all-layer folds) to solve with added necessary condition that the non-$\mathcal{H}$ folds must match up appropriately after all the folds in $\mathcal{H}$ are made. (The time spent of checking for this necessary condition can be attributed to the non-$\mathcal{H}$ folds that vanish after every fold.) Since $\mathcal{H}$ contains at least one fold, performing the $\mathcal{H}$ folds (strictly) reduces the size of the problem, and we continue. The base case consists of just horizontal or vertical folds, which corresponds to a 1-D problem. In summary we have:

Lemma 6. *If a crease pattern is foldable, it remains foldable after the folds in $\mathcal{H}$ have been made in any feasible way considering $\mathcal{H}$ to be a 1-D problem and ignoring other creases.*

To find $\mathcal{H}$ quickly we maintain the number of mountain and valley creases for each row and column of creases. We maintain these numbers as we make folds in $\mathcal{H}$. To do this we traverse all the creases that will vanish after a fold and decrement the corresponding numbers. The cost of this traversal is attributed to the vanishing creases. Every time the number of mountain or valley creases hits zero in a column or a row, we add the row or column to a list to be used as the new $\mathcal{H}$ in the next step. Thus,

Theorem 5. *The problem of deciding simple foldability of an orthogonal crease pattern on a rectangular piece of paper can be solved in linear time.*

6 Hardness of Simple Folds in 2-D

In this section we prove that the problem of deciding whether a 2-D axis-parallel mountain-valley pattern can be simply folded is (weakly) NP-hard, if we allow the initial paper to be an arbitrary orthogonal polygon. We also show that it is (weakly) NP-hard to decide whether a mountain-valley pattern on a square piece of paper can be folded by some-layers simple folds, if the creases are allowed to be axis-parallel *plus* at a 45-degree angle.

Both hardness proofs are based on a reduction from an instance of PARTITION: given a set X of n integers $a_1, a_2, \ldots, a_n$ whose sum is A, does there exist a set $S \subset X$ such that $\sum_{a \in S} a = A/2$? For convenience we define the set $\bar{S} = X \setminus S$. Also, without loss of generality, we assume that $a_1 \in S$.

The PARTITION problem is known to be (weakly) NP-hard [7]. We transform an instance of the PARTITION problem into an orthogonal 2-D crease pattern on a orthogonal polygon, as shown in Figure 5.

In the figure, all creases are valleys. There is a staircase of width ε corresponding to $a_1, \ldots, a_n$, where $0 < \varepsilon < 2/3$. There is a step in the staircase of length a_i corresponding to each element a_i in X. L is a constant greater than $A/2$. Also $W_2 > W_1$. Also let there be a coordinate system with horizontal x-axis and vertical y-axis.

Lemma 7. *If the* PARTITION *instance has a solution, then the crease pattern in Figure 5 is simply foldable.*

Lemma 8. *If the crease pattern in Figure 5 is simply foldable, there exists a solution to the* PARTITION *instance.*

Lemmas 7 and 8 imply the following theorem.

Theorem 6. *The problem of deciding simple foldability of a orthogonal paper with an orthogonal crease pattern is (weakly) NP-complete.*

In the full paper, we prove the following theorem, which shows that even on a rectangular piece of paper it is hard to decide foldability if, besides axis-parallel, there are creases in diagonal directions (45 degrees with respect to the axes):

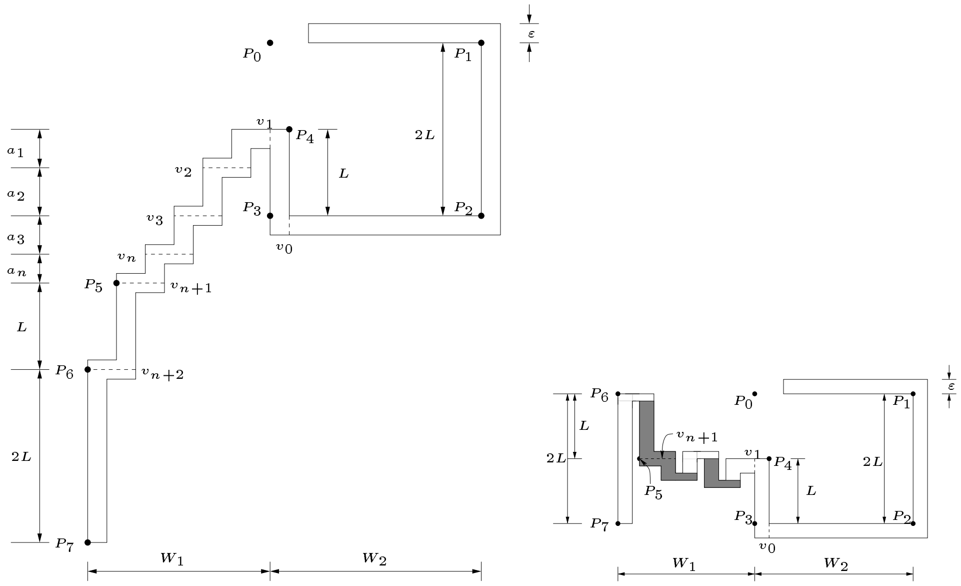

Fig. 5. Top: Hardness reduction from PARTITION problem. Bottom: Semi-folded staircase confined between y coordinates of P_1 and P_2. The top side of the paper is drawn white and the other side is drawn gray.

Theorem 7. *It is (weakly) NP-complete to decide the foldability of an (axis-parallel) square sheet of paper with a crease pattern having axis-parallel creases and creases at the diagonal angles of 45 degrees with respect to the axes, for both all-layers and some-layers simple folds.*

The problem is open for the one-layer case.

7 No Mountain-Valley Assignments

An interesting case to consider is when all creases do not have mountain-valley assignment: Any crease can be folded in either direction. Even with this flexibility, we are able to show that the problem is hard (see the full paper for the proof):

Theorem 8. *The problem of deciding the foldability of a orthogonal paper with a crease pattern that does not have mountain-valley assignment is (weakly) NP-complete, for both the all-layers and some-layers cases.*

The problem is open for the one-layer case.

Acknowledgments We thank Jack Edmonds for helpful discussions which inspired this research. E. Arkin acknowledges support from the NSF (CCR-9732221) and HRL Labs. M. Bender acknowledges support from HRL Labs. J. Mitchell acknowledges support from HRL Labs, the NSF (CCR-9732221), NASA (NAG2-1325), Northrop-Grumman, Sandia, Seagull Technology, and Sun Microsystems.

References

1. E. M. Arkin, S. P. Fekete, J. S. B. Mitchell, and S. S. Skiena. On the manufacturability of paperclips and sheet metal structures. In *Proc. of the 17th European Workshop on Computational Geometry*, pages 187–190, 2001.
2. M. Bern and B. Hayes. The complexity of flat origami. In *Proc. of the 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 175–183, 1996.
3. P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. *J. of Computational Biology*, 5(3), 1998.
4. E. D. Demaine, M. L. Demaine, and J. S. B. Mitchell. Folding flat silhouettes and wrapping polyhedral packages: New results in computational origami. In *Proc. of the 15th ACM Symposium on Computational Geometry*, 1999.
5. M. Farach. Optimal suffix tree construction with large alphabets. In *Proc. of the 38th Symp. on Foundations of Computer Science*, pages 137–143, 1997.
6. M. Gardner. The combinatorics of paper folding. In *Wheels, Life and Other Mathematical Amusements*, Chapter 7, pp. 60–73. W. H. Freeman and Company, 1983.
7. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.
8. D. Harel and R. E. Tarjan. Fast algorithms for finding nearest common ancestors. *SIAM J. on Computing*, 13(2):338–355, 1984.
9. T. Hull. On the mathematics of flat origamis. *Congressum Numerantium*, 100:215–224, 1994.
10. J. Justin. Towards a mathematical theory of origami. In Koryo Miura, editor, *Proc. of the 2nd International Meeting of Origami Science and Scientific Origami*, pages 15–29, 1994.
11. R. M. Karp and M. O. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, 1987.
12. T. Kawasaki. On the relation between mountain-creases and valley-creases of a flat origami. In H. Huzita, editor, *Proc. of the 1st International Meeting of Origami Science and Technology*, pages 229–237, Ferrara, Italy, December 1989. An unabridged Japanese version appeared in *Sasebo College of Technology Report*, 27:153–157, 1990.
13. R. J. Lang. A computational algorithm for origami design. In *Proc. of the 12th ACM Symposium on Computational Geometry*, pages 98–105, 1996.
14. L. Lu and S. Akella. Folding cartons with fixtures: A motion planning approach. *IEEE Trans. on Robotics and Automation*, 16(4):346–356, 2000.
15. W. F. Lunnon. Multi-dimensional map-folding. *The Computer Journal*, 14(1):75–80, 1971.
16. R. Motwani and P. Raghavan. *Randomized Algorithms*, Chapter 8.4, pages 213–221. Cambridge University Press, 1995.
17. B. Schieber and U. Vishkin. On finding lowest common ancestors: Simplification and parallelization. *SIAM J. on Computing*, 17(6):1253–1262, 1988.
18. J. S. Smith. Origami profiles. *British Origami*, 58, 1976.
19. J. S. Smith. *Pureland Origami 1, 2, and 3*. British Origami Society. Booklets 14, 29, and 43, 1980, 1988, and 1993.
20. M. Thorup. Faster deterministic sorting and priority queues in linear space. In *Proc. of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 550–555, 1998.
21. C-H. Wang. *Manufacturability-driven decomposition of sheet metal*. PhD thesis, Carnegie Mellon University 1997. Technical report CMU-RI-TR-97-35.