

Map Folding Variations : Flat-foldability of  
Box-pleated Patterns and Validity of Overlapping  
Orders

March 2021

Yiyang Jia

Map Folding Variations : Flat-foldability of  
Box-pleated Patterns and Validity of Overlapping  
Orders

Graduate School of Systems and Information Engineering  
University of Tsukuba

March 2021

Yiyang Jia

# Abstract

The map folding problem is one of the most attractive unsolved problems in the field of computational origami. It is a problem about origami that determines whether a piece of paper can be made flat by folding along the edges in a given grid pattern or not. It relates traditional fields of computer science, e.g., algorithms and computational complexity, to the new field, geometric folding (called origami when the objects are two-dimensional). Recent developments in origami research include the fields of mathematics, theoretical physics, mechanisms, design, robotics, and biology.

The focus of this thesis is on several problems related to the map folding, which is on the boundary of P and NP. The map folding problem has been studied for over 30 years. Its difficulty comes from the subtle balance between "certain" and "uncertain" in the problem setting. The certain thing is the pattern that specifies the places to be folded. It is natural to believe that the more complex the pattern is, the more time it will take to determine whether it can be flat-folded or not. A recent result shows that this problem becomes NP-hard by simply adding a few diagonal lines to the grid pattern. On the other hand, the uncertain thing is the folding direction along the edges (mountain or valley) and folding methods (general fold or simple fold).

Because it is difficult to reveal the complexity of the map folding problem itself, we turned to consider related problems with some constraints or some variants to get closer to the complexity of the map folding problem. When trying to characterize a class of problems in terms of computational complexity, one of the most exciting things is to find where the boundary lies. The boundary between P and NP is supposed to divide the set of related problems of the map folding. Some constraints, e.g. simple folding, are raised from a practical point of view. If the complexity of a problem with such constraints turns out to be P, then we can say that we have obtained a useful folding model for the practical use.

We did the research on the flat-foldability of  $1 \times n$  and  $2 \times n$  box-pleats, a pattern formed by some edges of the grid patterns and some diagonals. Our results compensate most of the gap between a P: "deciding the flat-foldability of a  $1 \times n$  and a  $2 \times n$  map" and an NP: "deciding the flat-foldability of a  $4 \times n$  box-pleating pattern". Now, the only remained future work in the gap is "deciding the flat-foldability of a  $3 \times n$  box-pleat pattern".

In the problem of determining the flat-foldability of a pattern, the more constraints, the less possible flat-folded states. This makes it difficult to determine whether or not a known flat state can be reached. Thus, we also considered the problem of determining the validity of given overlapping orders, focusing on the duality that exists between the folding and the folded states.

As the extension, we then investigated to which extent the complexity would change

when general folds are replaced by simple folds. The result is that the complexity does not change when the input is the total overlapping order of all the faces in the folded state. Furthermore, we considered some cases where partial orders are given as inputs because MV-assignments are essentially partial orders given on the set of faces. A partial order provides little information about the overlap, and distinct folds can yield equivalent folded states. We proved that the computational complexity of determining the validity of the boundary overlapping orders (partial orders given on the faces in contact with the boundary) is the same as that of determining the validity of the total overlapping order. Some interesting open directions include: the reachable folded states using different folding models, the decision considering multiple partial orders together, and so on.

# Acknowledgement

In my doctoral period, many people supported me and offered me invaluable help to both my research and my life as a foreign student. I would like to express my sincere gratitude to all those who ever helped me during these three years.

Deepest thanks to my supervisor, Professor Jun Mitani, for all his generous help, his support and suggestions during my entire doctoral study, and the ideal conditions he provided for my research, also for his stimulating influence as both a scholar and a researcher. And above all, for his smiles full of gentleness. I am sure that there is one among those stars that guides me through the infinite unknown.

I appreciate the comments and suggestions of Professors Ryuhei Uehara and Yoshihiro Kanamori. Without their help, the specific portions of this thesis would not have been finished.

# Contents

|  |            |
|--|------------|
| <b>Abstract</b>  | <b>i</b>   |
| <b>Acknowledgement</b>   | <b>iii</b> |
| <b>1 Introduction</b>  | <b>1</b>   |
| 1.1 Historical Origins . . . . .                                       | 1          |
| 1.2 Background of Map Folding . . . . .                                | 1          |
| 1.3 Map folding and Relevant Topics . . . . .                          | 3          |
| 1.4 Principal Contribution . . . . .                                   | 3          |
| 1.5 Practical Value of Our Results . . . . .                           | 5          |
| 1.6 Outline of This Thesis . . . . .                                   | 6          |
| 1.7 Publications . . . . .   | 7          |
| <b>2 Related Research</b>  | <b>8</b>   |
| 2.1 Theoretical Research . . . . .                                     | 8          |
| 2.1.1 Results about flat-foldings and flattenings . . . . .            | 9          |
| 2.1.2 Other theoretical topics in computational origami . . . . .      | 15         |
| 2.2 Practicality . . . . .   | 16         |
| 2.2.1 Structural designs and mechanisms . . . . .                      | 17         |
| 2.2.2 Self-Folding mechanisms and robotics . . . . .                   | 18         |
| 2.2.3 Puzzles . . . . .  | 19         |
| 2.2.4 Other applications related to our research . . . . .             | 20         |
| <b>3 Topics in This Thesis</b>   | <b>21</b>  |
| 3.1 Three Kinds of Variations of Map Folding . . . . .                 | 21         |
| 3.2 Another Respective - Valid Overlapping Orders . . . . .            | 23         |
| <b>4 Flat-foldability of Box-pleating Patterns</b>                     | <b>24</b>  |
| 4.1 Box-pleating Patterns of Size $1 \times n$ . . . . .               | 24         |
| 4.1.1 Notations . . . . .  | 25         |
| 4.1.2 Patterns with all the creases and the MV assignment . . . . .    | 26         |
| 4.1.3 Patterns with part of the creases but no MV assignment . . . . . | 28         |
| 4.2 Box-pleating Patterns of Size $2 \times n$ . . . . .               | 33         |
| 4.2.1 Notations . . . . .  | 33         |
| 4.2.2 Flat-Foldability in every part . . . . .                         | 36         |
| 4.2.3 Linear-time algorithm for the decision problem . . . . .         | 46         |
| 4.2.4 An instance of size $2 \times 15$ . . . . .                      | 48         |

|          |   |            |
|----------|---|------------|
| 4.3      | Future work and applications . . . . .                              | 50         |
| <b>5</b> | <b>Valid Overlapping Order Problems</b>                             | <b>53</b>  |
| 5.1      | Valid Overlapping Order Problems in box-pleating Patterns . . . . . | 54         |
| 5.2      | Valid Total Overlapping Orders with Simple Folds . . . . .          | 55         |
| 5.2.1    | Notations . . . . .   | 56         |
| 5.2.2    | Outline . . . . .   | 58         |
| 5.2.3    | Deciding the validity of the given order . . . . .                  | 58         |
| 5.2.4    | Algorithm description . . . . .                                     | 69         |
| 5.3      | Valid Boundary Overlapping Orders with Simple Folds . . . . .       | 73         |
| 5.3.1    | Preliminaries . . . . .   | 73         |
| 5.3.2    | Outline . . . . .   | 76         |
| 5.3.3    | Interchangeable consecutive folds . . . . .                         | 78         |
| 5.3.4    | Algorithms of the decision problem . . . . .                        | 90         |
| 5.3.5    | Enumeration of valid whole simple folding sequences . . . . .       | 96         |
| <b>6</b> | <b>Conclusion and Future Work</b>                                   | <b>100</b> |

# List of Figures

|      |  |    |
|------|--|----|
| 1.1  | The map folding problem. . . . .   | 4  |
| 1.2  | A crease pattern not flat-foldable . . . . .   | 4  |
| 1.3  | Instances of decision problems on box-pleating patterns of size $1 \times n$ . . .     | 4  |
| 1.4  | Instance of a box-pleating pattern of size $2 \times 10$ . . . . .                     | 5  |
| 1.5  | A map is folded into a square of $m \times n$ layers. . . . .                          | 5  |
| 1.6  | Separation with respect to the horizontal center line. . . . .                         | 5  |
| 2.1  | An example of some-layers simple fold. . . . .   | 11 |
| 2.2  | The two kinds of basic simple-fold operations: crimps and end-folds. . . .             | 11 |
| 2.3  | Locally but not globally flat-foldable patterns . . . . .                              | 13 |
| 3.1  | Minimal unfoldable map of size $2 \times 5$ . . . . .                                  | 22 |
| 3.2  | One-dimensional maps with generalized lengths between creases. . . . .                 | 22 |
| 3.3  | The box-pleating patterns. . . . .   | 23 |
| 4.1  | Basic notations in a box-pleating crease pattern. . . . .                              | 25 |
| 4.2  | Possible combinations of Ms and Vs within a square. . . . .                            | 26 |
| 4.3  | Result state of the first step applied on the instance Figure 1.3. . . . .             | 26 |
| 4.4  | Flat-foldable and not flat-foldable crease patterns in a single square. . . .          | 28 |
| 4.5  | The initial MV assignment and folded state before the unfolding operations.            | 29 |
| 4.6  | Notations of the box-pleating patterns of size $1 \times n$ . . . . .                  | 29 |
| 4.7  | An input box-pleating pattern without any MV assignment . . . . .                      | 30 |
| 4.8  | Unfolding the vertical creases . . . . .   | 30 |
| 4.9  | Unfolding the diagonal creases . . . . .   | 31 |
| 4.10 | Locally flat-foldable but not globally flat-foldable patterns of size $3 \times 3$ . . | 33 |
| 4.11 | Definitions in a box-pleating pattern of size $2 \times n$ . . . . .                   | 34 |
| 4.12 | A flat-folded state in the shape of a zigzag and a front view. . . . .                 | 35 |
| 4.13 | Two possible cases for creases around a vertex in $\overline{V}_4$ . . . . .           | 37 |
| 4.14 | The no-center-line parts and the method to fold them. . . . .                          | 40 |
| 4.15 | Operations of combination foldings. . . . .  | 41 |
| 4.16 | MV assignments for Patterns 4-1 and 4-2 . . . . .                                      | 43 |
| 4.17 | Example of size $2 \times 15$ . . . . .  | 48 |
| 4.18 | Two parts separated by $S_8$ in Pattern 4-1. . . . .                                   | 48 |
| 4.19 | Three parts in $p_1$ with respect to the existence of the centerline. . . . .          | 49 |
| 4.20 | The flat-folded state of $p_1$ . . . . .   | 49 |
| 4.21 | The entire MV assignment corresponding to the input pattern. . . . .                   | 49 |
| 5.1  | Valid states of layers under the butterfly condition . . . . .                         | 55 |



|      |  |    |
|------|--|----|
| 5.2  | An example of $6 \times 8$ map. . . . .  | 57 |
| 5.3  | The checkerboard pattern of a $2 \times 2$ map and its corresponding folded state. . . . . | 59 |
| 5.4  | The corresponding one-dimensional map . . . . .  | 61 |
| 5.5  | The partial order of folds based on the $MV$ assignment . . . . .                          | 61 |
| 5.6  | The uniformity of the positions in an $m \times n$ map and a $1 \times n$ map. . . . .     | 62 |
| 5.7  | Inner orders of $p_k$ directing to different $g(k + 1, x, y)$ s . . . . .                  | 63 |
| 5.8  | Inner orders of $p_k$ directing to the same $g(k + 1, x, y)$ . . . . .                     | 65 |
| 5.9  | Interdependencies among groups . . . . .   | 67 |
| 5.10 | The notations and the coordinate system of the map. . . . .                                | 74 |
| 5.11 | An instance with two parallel boundary strips overlapping each other. . . . .              | 77 |
| 5.12 | Computing the orders of folds an instance 1D map. . . . .                                  | 80 |
| 5.13 | Two possible cases for interchangeable parallel folds. . . . .                             | 81 |
| 5.14 | The corner squares in boundary areas. . . . .  | 82 |
| 5.15 | Possible relative positions of the corner squares and four end-folds. . . . .              | 83 |
| 5.16 | The first three cases for four consecutive end-folds. . . . .                              | 83 |
| 5.17 | Cases satisfying and dissatisfying the interchangeable condition. . . . .                  | 85 |
| 5.18 | Unique overlapping orders. . . . .   | 86 |
| 5.19 | Classes 4 and 5: Two perpendicular pairs of parallel end-folds. . . . .                    | 87 |
| 5.20 | Class 6: Two interchangeable crimps. . . . .   | 87 |
| 5.21 | Class 7: Two interchangeable crimps and end-fold. . . . .                                  | 88 |
| 5.22 | Class 8: Two interchangeable end-folds. . . . .  | 88 |
| 5.23 | Merge occurs when two consecutive end-folds both involve corner squares. . . . .           | 88 |
| 5.24 | The reduction by folding the first crimps. . . . .   | 91 |
| 5.25 | Three cases in the first class. . . . .  | 92 |
| 5.26 | The rectangle formed by the four corner squares in a middle state. . . . .                 | 92 |
| 5.27 | An example of the implement of folds involving no corner squares. . . . .                  | 92 |
| 5.28 | A possible lock induced by Step 3. . . . .   | 96 |

# List of Tables

|     |   |    |
|-----|---|----|
| 4.1 | Thirty-six possible patterns and corresponding operations . . . . .       | 35 |
| 4.2 | MV assignments and overlaps in every folded state of the center-line part | 38 |
| 4.3 | Ten classes of combinations involved in the second phase. . . . .         | 42 |
| 4.4 | Ten classes of combinations involved in the second phase. . . . .         | 43 |
| 4.5 | The connections of a connection section and Pattern 5-2 . . . . .         | 45 |
| 4.6 | Connections of a connection section and a center-line part . . . . .      | 51 |

# Chapter 1

## Introduction

In this chapter, we aim to introduce the research field and relevant topics of map folding. The flow is described as follows. In Section 1.1, we will describe the historical origins of the map folding problem. In Section 1.2, we will introduce the background of the map folding problem, most about the research field of flat-folding. Then, in Section 1.3, we will give a concrete description of the map folding problem and the related problems we studied. In Section 1.4, we will list the contribution of our research. In Section 1.5, we will talk about the value of our results in practical productions and uses. Finally, we will give the outline of this thesis in Section 1.5 and then list our publications in Section 1.6.

### 1.1 Historical Origins

Geometric folding is historically associated with culture and entertainment. The word “Origami” is the Japanese for “paper folding”, which is the restricted geometric folding in a two-dimensional sense. Since many centuries ago, it has played an important role in pedagogy and mathematics. At the outset, the only pursuit of origami might be just for fun, whereas approaching this field more scientifically has increasingly become the main tendency. Especially in the last fifty years, the rise of both theoretical developments and practical applications based on origami is notable.

Traditionally, origami describes the works made by folding pieces of paper. According to Ref. [23], its historical origin can be dated back to 105 A.D., roughly coinciding with the invention of paper. The Japanese word reveals the popularity of itself in Japan. In many early pieces of literature, you can find that origami is historically developed and popularized since the Heian period, more than a thousand years ago. Even if we do not value its importance in science, it is considered not only as a game but an activity full of cultural value.

### 1.2 Background of Map Folding

When people’s interest concentrates on the mathematics of origami, the constructions achievable by origami are put into concern. Relevant researches have been developed from the classical construction problems in geometry. The firstly concerned problem is whether the construction problems solvable by compass and ruler can also be solved by

folding a piece of paper. For example, drawing a line through two given points with a ruler can also be done by folding a piece of paper along a line. In this field, the axiom system of elementary folding principles was firstly published in 1986 [54]. The axioms were built based on the basic operations. For example, the first axiom says that given two distinct points  $p_1$  and  $p_2$ , there is a unique fold that passes through both of them. For another example, the third axiom says that given two lines  $l_1$  and  $l_2$ , there is a fold that places  $l_1$  onto  $l_2$ . As a more completed version, the seven-axioms-system is much more well-known. In such a system, the first six axioms are known as Huzita's axioms, and the seventh is discovered by Koshiro Hatori and Robert J. Lang in 2001 [6]. Later, Jorge et al. gave some suggestions to make the system more perfect [70]. Recently, along with the development of abstract algebra, the constructability of origami has also been investigated in an algebraic way. In fact, compared to the compass and straightedge, origami naturally entails an operation-based axiom system involving more symmetric properties, which serves to associate the equivalent algebra of origami and the symmetric groups together. Constructible numbers of origami were introduced in Ref. [81]. They showed that the cardinal number of the set of constructible numbers of origami is larger than the set of constructible numbers of compass and straightedge. With the help of the equivalence between the algebraic definitions and the geometric definitions of the constructability of origami, geometric questions on the constructions of origami can be turned into algebraic questions. This encouraged Paulus to provide an elaborate description of what are and what are not constructible with origami [81]. Some practicable methods to trisect an angle, construct regular polygons, the golden number, the roots of cubic and quartic equations were provided as instances.

Folding the paper to get a line segment seems very natural. Conversely, drawing a line and then folding along the line is also considered. In the former process, by multiple folding operations, we can always obtain multiple line segments. While in the latter process, when multiple line segments exist, it is not always possible to fold a paper just along these line segments. The discussion about in which cases a piece of paper can be folded to the plane and just along the given line segments drives the abstraction to the *flat-folding problem*. And the set of given line segments is then viewed as a planar graph called the *crease pattern*. The description of a crease pattern as a graph  $G = \{V, E\}$  defines its *vertices* in  $V$  and *creases* in  $E$ . Generally, both the lengths of creases and the angles between creases influence the flat-foldability. But the problem can be simplified when the crease pattern has only one vertex, in which case the lengths of creases do not matter. This kind of flat-foldability is called *local flat-foldability*, suggesting that any neighborhood of a vertex can be considered in this way as long as it is small enough. The solution to this particular case turned out to be a filtering procedure based on a set of constraints. The details on these constraints will be introduced in Chapter 2. The local flat-foldability can be decided in time linear in the number of the creases [14].

However, when considering the flat-foldability of an entire pattern with multiple vertices, the computational complexity of the problem increases largely to make it NP-hard [14]. Based on these results, we concern crease patterns with multiple vertices but also less arbitrariness. The grid pattern is such a pattern and has been popularly used in our daily life. The problem is thus reduced to a new version, the map folding problem, which was proposed by Jack Edmonds in a personal talk in 1997.

### 1.3 Map folding and Relevant Topics

The original *map folding problem* was presented as a decision problem on the existence of possible foldings continuously reshaping a map of size  $m \times n$  (in the form of a rectangular grid pattern composed by  $m \times n$  squares) to a *flat-folded state* of size  $1 \times 1$  (a state that all the squares totally overlap each other) [23]. The concern is on the computational complexity to make the decision. As part of the input, every non-boundary edge of the grid pattern between two squares is assigned as either a *Mountain* (“M”) or a *Valley* (“V”) to represent the direction of the fold. An example of the input is shown in Figure 1.1 (1). Figure 1.1 (2) gives the directions indicated by Mountains and Valleys (in the following illustrations, red solid lines indicate mountains while blue dashed lines indicate valleys). The direction decides which side (either the front or the back) of the two squares is supposed to face each other after the fold. All the *Ms* and *Vs* compose the entire assignment of the map, which is called a *Mountain-Valley assignment* (an *MV assignment*).

The MV assignment in spirit gives a partial order on the set of the squares. It gives the unique overlapping order of every pair of faces sharing a crease in the final flat-folded state while the orders of the faces without any common crease are rather free. From this viewpoint, the decision on the flat-foldability can be reconsidered as finding special orders of the squares simultaneously following the partial order and the non-penetrating condition naturally induced by the setting of paper folding. As many problems involving partial orders or sequencing are proved intractable, the computational complexity of the original map folding problem is rather unclear and considered hard to analyze. After almost thirty years, the map folding problem still remains unsolved in a universal sense, even though some results were proposed for its variations.

To approach the essential map folding problem, we studied some of its variations in two main topics. The first topic is towards the variations of its patterns. In our researches, the crease patterns involve two sorts. The first sort is the grid pattern, which involves all the boundary edges of the squares in the map. The second sort is the *box-pleating patterns* which involve part of the diagonal and boundary edges of the squares.

Our second topic is about whether an input order of the faces forms a valid folded state where self-penetration does not exist or not. We consider this problem because as long as a valid folded state exists, the corresponding input pattern is flat-foldable, which means that the paper can be folded onto a plane when restricted folded along and only along the line segments. In fact, most of the natural patterns are not flat-foldable. An example is given in Figure 1.2.

The details on the relationship of both topics will be given in Chapter 2.

### 1.4 Principal Contribution

As mentioned before, our results can be concluded as two parts, corresponding to whether the input is a crease pattern or an overlapping order of squares. Now we list our concrete conclusions as follows.

For the decision problems with crease patterns as inputs, we have proved that:

(1-1) Section 4.1: Any grid pattern of size  $1 \times n$  with all the diagonals and an MV assignment satisfying local flat foldability can be flat-folded (see Figure 1.3 (1-1)). Finding

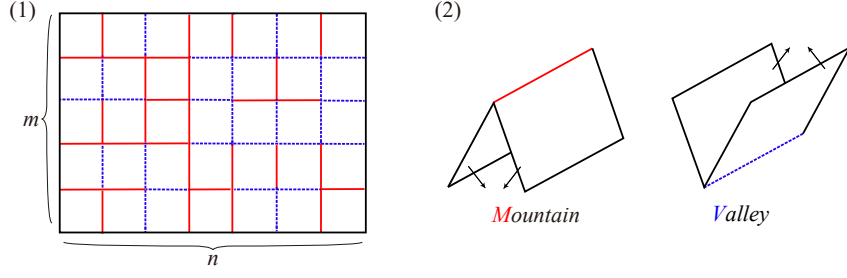


Figure 1.1: (1) Input of the map folding problem. (2) MV-assignment

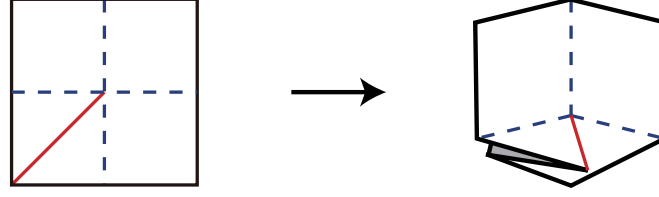


Figure 1.2: A crease pattern which is not flat-foldable

a method to fold it takes linear time.

(1-2) Section 4.1: Any box-pleating crease pattern of size  $1 \times n$  (see Figure 1.3 (1-2)) satisfying local flat foldability can be flat-folded. Finding a method to fold it takes linear time.

(2) Section 4.2: Any box-pleating crease pattern of size  $2 \times n$  (see Figure 1.4) satisfying local flat foldability can be flat-folded. Finding a method to fold it takes linear time.

For the decision problem about the validity of given overlapping orders of squares, we have the following results. The explanation of fold models will be given in Section 3.1.

(3) Section 5.1: It costs  $O(mn)$  time to decide the validity of the overlapping order of all the triangles of a grid-diagonal pattern in the general fold model.

(4) Section 5.2: Given an ordering of all the squares representing a flat-folded state of an map of size  $m \times n$ , its validity in the simple fold model can be decided in  $O(mn)$  time. Furthermore, within the same time, it gives a feasible whole simple folding sequence  $F$  when  $O$  is decided to be valid.

(5) Section 5.3: Given an ordering of only the  $2m + 2n - 2$  squares aligning on the boundary of an map of size  $m \times n$ , its validity to represent a flat-folded state in the simple fold model can be decided in  $O(m + n)$  time (the problem setting is indicated by Figure 1.5). Furthermore, within the same time, it gives a feasible whole simple folding

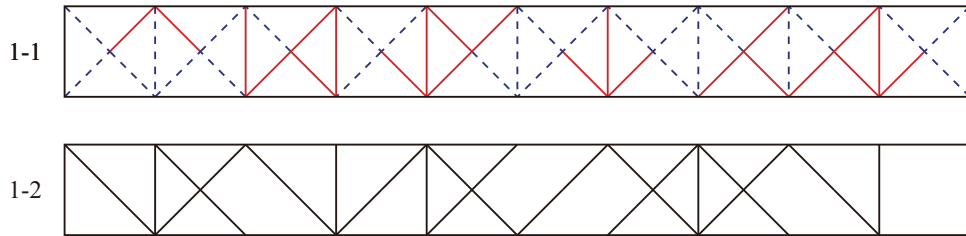


Figure 1.3: Instances of decision problems on box-pleating patterns of size  $1 \times n$ .

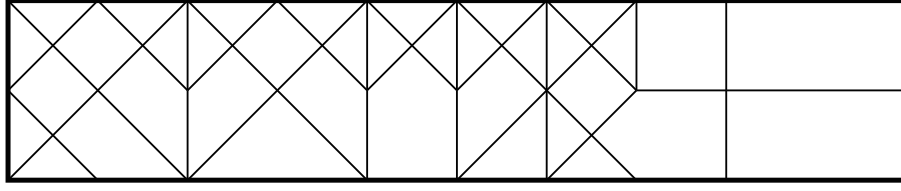


Figure 1.4: Instance of a box-pleating pattern of size  $2 \times 10$

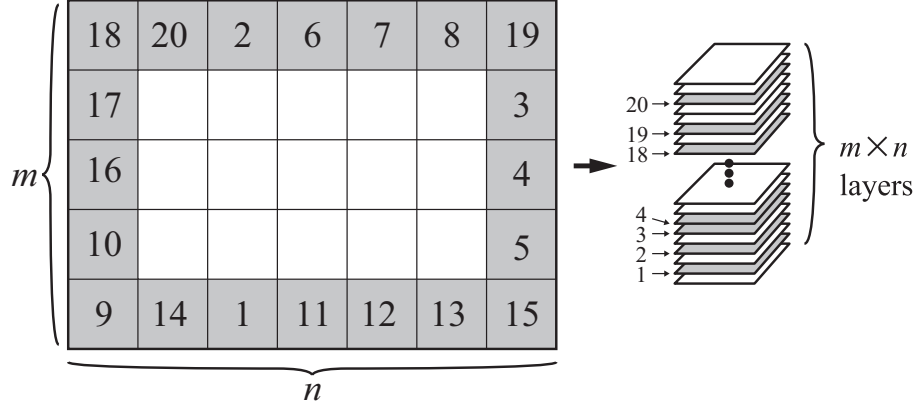


Figure 1.5: A map is folded into a square of  $m \times n$  layers. The overlapping order of the boundary squares (colored gray) in the folded state is given.

sequence  $F$  when  $O$  is decided to be valid.

Corresponding to the valid inputs of (5), we further give the following conclusion: There is an enumeration algorithm for valid whole simple folding sequences for a given overlapping order of the boundary squares. With an  $O(m + n)$  time precomputing, it enumerates each folded state in  $O(\max\{m, n\}mn)$  time delay.

## 1.5 Practical Value of Our Results

Theoretically, we concern about the map folding problem because it is not only a complex problem on the boundary of P and NP, but also can be considered as a computing model. Also, the folding operations, simple folds(the folding operations along a single

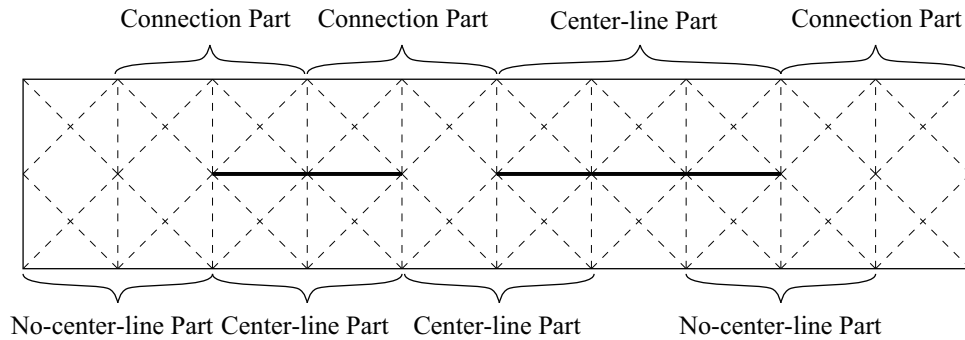


Figure 1.6: Separation with respect to the horizontal center line.

line segment and only applied on multiple layers above all the other layers, whose precise definition will be given later) and simple unfolds (the reverse folding operation of simple folds) considered in our researches, are usually used in the modelings rather than the general folds. For example, it is proved that a computational model considering the simple folds and simple unfolds on a  $1 \times n$  map with a final cut has at least the same or even more powerful performances compared to the Non-deterministic Turing Machine [7].

Most of our concerned problem turned out to have polynomial-time solutions, which brings the possibility of practical uses. In fact, many practical applications are tightly related to our topics, involving robotic designs, puzzle designs, application developing, and so on. These applications will be introduced in Section 2.2. The simple folds and unfolds are considered as basic operations in the practical productions and also the applicable tools when the productions are made by robots. To restrict the scale of the solution sets of puzzles, simple folds and simple unfolds are also desired because of the result after every fold can be described using a countable set. Also, if used to design puzzles, the partial order studied in our last research must give more entertainments than the general orderings.

As another kind of applications, box-pleats patterns are used as the most popular patterns to construct arbitrary polycubes. The structure of polycubes are popularly used in designs and architectures. Some works are introduced in Ref. [39].

## 1.6 Outline of This Thesis

We arrange our contents in six chapters as follows.

In Chapter 2, we will introduce closely related results on both the theoretical aspect and the practical aspect. Especially, we emphasize the critical early results, which give the historical background, and newly developed results, which reveal the potential and importance of our research field.

In Chapter 3, we will give an overview of our research. In this thesis, we intend to categorize our topics into two main directions. One is the decision on whether an input crease pattern can be flat-folded or not; the other is the decision on whether a given order of the faces corresponds to a valid flat-folded state of the crease pattern. Between the two directions, there are definitely many correspondences. However, what we emphasize is the most intrinsic one, that any given order of the faces, the input of the latter problem, can be viewed as a guess when solving the former problem by a non-deterministic Turing machine.

In Chapter 4, we will talk about two decision problems in the first direction, which take the box-pleating patterns of size  $1 \times n$  and  $2 \times n$  as the inputs, respectively. Some inputs also have variant MV assignments. We will explain the details of our solutions and propose the corresponding algorithms, which are all in linear time.

In Chapter 5, we will focus on the other direction and present three concrete problems. The first one takes the orders of faces of box-pleating patterns as input; the other two takes the orders of squares in regular grid patterns while restrict the folding operation to be simple folds. We further consider the orders of the squares aligning on the boundary instead of all the squares in the third problem. These varied problem settings are rather natural because of the following reasons. (1) Simple folds are much more popularly used



in practical uses; (2) The result set of a simple fold only has two elements, corresponding to that a part either would be folded over the other or be folded below the other, which makes it similar to the moving direction of the head in a Turing machine model; (3) MV assignments are essentially partial orders given on the set of faces. The first reasons inspired us to study the simple folds, and the third one inspired us to pay attention to partial orders (the order on the boundary squares in our study). These problems are all proved linear-time solvable. In distinct sections of Chapter 5, we will propose the corresponding algorithms, respectively.

In Chapter 6, we will summarize our conclusions and point out some open questions as future research fields.

## 1.7 Publications

This thesis is based on the following publications:

Journal papers (with peer review)

1. Yiyang Jia, Jun Mitani, Ryuhei Uehara, Efficient Algorithm for  $2 \times n$  Map Folding with a Box-pleated Crease Pattern, *Journal of Information Processing*, 2020, Volume 28, Pages 806-815.
2. Yiyang Jia, Jun Mitani, Ryuhei Uehara, Valid Orderings of Layers When Simple-Folding a Map, *Journal of Information Processing*, 2020, Volume 28, Pages 816-824.

International conference papers (with peer review)

1. Jia Y., Kanamori Y., Mitani J. (2019) Flat-Foldability for  $1 \times n$  Maps with Square/Diagonal Grid Patterns. In: Das G., Mandal P., Mukhopadhyaya K., Nakano S. (eds) *WALCOM: Algorithms and Computation. WALCOM 2019. Lecture Notes in Computer Science*, vol 11355. Springer, Cham.

International conference papers (without peer review)

1. Yiyang Jia, Jun Mitani and Ryuhei Uehara, Efficient Algorithm for  $2 \times n$  Map Folding with Diagonal Creases, *JCDCGGG 2019*

# Chapter 2

## Related Research

In this chapter, we will introduce the related research on both the theoretical level and on the practical uses in Section 2.1 and Section 2.2, respectively. The results on practical uses further reveal the potential value of our research. Then, in Section 2.3, we will introduce our topics and emphasize some closely associated results. Some definitions will also be given in this chapter.

### 2.1 Theoretical Research

The rudimentary concepts in the map folding associate it with the research field of geometric folding, especially at a two-dimensional level, called Origami. In most cases, the initial state before the folding is simply connected, flat, and thus can be modeled as a piece of paper. When the paper is reasonably folded following some certain rules and thus reaches some certain folded states, the folding process is called a *folding motion*.

*Crease patterns* play the most critical role in the folding motions. They decide along which line segments or curves (curve-folding) of the paper should be folded. A crease pattern can be viewed as a graph  $G = \{V, E\}$  embedded to the bounded plane, i.e., the paper.  $E$  is the set of all the creases, and  $V$  defines the intersecting points of these creases. Each region bounded by creases or boundary of the paper and with no crease inside is called a *face*. Generally, for the crease patterns whose  $E$  is composed of only line segments, a final folded state with every crease folded and every face flat is usually desired. For curve-foldings,  $E$  is composed of curved creases. Other than the creases, the faces also have to be transformed and thus could not remain in a flat form. The reason is that, essentially, a folding motion indicates continuous isometries of the planar paper, namely the continuous distance-preserving transformations from the initial unfolded state and a final state supposed to be reached.

We can also think about it in this way, the faces(with boundary) form a cover of the paper. In this cover, the intersection of any two elements is not empty if and only if they are neighbors in the initial unfolded state. The intersection of such two faces is the crease they share. An isometry has two requirements. One is that the elements in the cover are supposed to be mapped to the elements in the cover of the final folded state, where the mapping is  $C^2$  for non-curve foldings and  $C^1$  for curve-foldings. In fact, the elements in the cover of non-curve foldings are just planes and would be mapped to planes, while the curve-foldings map planes to curved planes with the non-zero curvature that makes

it not  $C^2$ . These mappings simultaneously preserve the distance along the faces.

Besides the two-dimensional papers, the folding of one-dimensional *linkages* (a connected graph formed by a set of line segments, which join at their endpoints), as well as certain transformations of some more complex objects in three-dimensional space, are also classified into the field of geometric folding. Some introductions for these themes will be given later.

Restricted to a smaller study field, the context of our studies is *flat foldings* of papers. A flat-folding refers to a mapping from the paper to its folded state satisfying that the image of each crease is a line segment with a dihedral angle measuring either  $\pi$  or  $-\pi$  and the image of each face is a congruent face. In the folded state, each face is also called a *layer*. The *flat-folding problem* asks whether a paper with a given crease pattern can be flat-folded [46]. When the objects are not restricted to papers, the term represents a geometrically multi-dimensional conception, which can be considered as a folding beginning from an  $n$ -dimensional object and ending with an  $n$ -dimensional shape, whereas the folding is completed in the  $n+1$ -dimension. In Section 2.1.1, we will introduce some critical results about these topics.

A relevant conception is *flatten*, which is also considered in many works. Compared with flat-folding, it considers foldings beginning from an  $n + 1$ -dimensional object and ending with an  $n$ -dimensional shape, via  $n + 1$ -dimensional folding motions. With local flatten operations, the *unfolding motions* (the reverse of folding motions) can be efficiently analyzed. One-dimensional flat-foldings and flattenings are further connected with some basic topics in the field of the linkage-folding. Also, the three-dimensional flat-folding and flattens have always been on the table since Albrecht Dürer proposed his famous problem about polyhedra flattening [23]. Some results will be given later in Section 2.1.1.

Our results are also classified to the computational origami considering the research methods. “Origami” is the Japanese word of “Paper Folding”. It may be rooted in the general acknowledgment that historically, origami was given birth as a pursuit for fun. Nevertheless, it developed remarkably along with the theoretical developments of mathematics, physics, and computational science in the latest two centuries. Especially in the last decade, the rise of both theoretical developments and practical applications based on origami was manifest. Computational origami is a study field which takes paper folding as the research object and sits between computer science and mathematics, using discrete mathematics, algorithms, geometry, and computational geometry to analyze the problems. Demaine et al. published a masterpiece in this field in 2007 [23]. With their efforts, this field has been eventually theoretically studied, although the applications ranging from robotics (made from panels and by foldings on the panels) to protein foldings have been studied in the practical sense from years ago. In the following, after the introductions about flat foldings, we will introduce some well-known theoretical results of other topics in the field of origami, involving both the computational origami and other theoretical results.

### 2.1.1 Results about flat-foldings and flattenings

The topic of flat-foldings and flattenings restrict the creases to be line segments. From both the theoretical perspective and the practical perspective, the decision on the flat-foldability of certain patterns has always been viewed as one of the most attractive topics

according to its locally analyzable property and usefulness in practical productions. In the following sections, we will introduce the results about flat-foldings and flattenings in one-dimensional, two-dimensional and three-dimensional background, respectively. Their practical applications will be introduced in the next section, together with all the other practical applications relative to our research. Also, even though these two conceptions can also be extended to manifolds in higher dimensions, we do not emphasize them because of lack of researching.

## One-dimensional flat-foldings and flattenings

In the one-dimensional case, the creases degenerate to crease points. A flat-folding means aligning all the bars of a planar linkage to the same line via a folding motion. Here, a planar linkage is a connected planar graph constructed by a set of one-dimensional segments joined at their endpoints with certain lengths. Each joint can be rotated by an angle ranged from 0 to  $2\pi$ . Specially, the segments are permitted to pass through one another freely. Some extensions further require certain joints to be pinned to the plane. Details about the terminologies and some basic properties are provided in Ref. [23], as both a generalization and a unification of some early works, such as [14, 53, 68], and so on.

A special case, where the input pattern is not a two-dimensional graph, but a one-dimensional graph, has been studied by Arkin et al. [9]. Their research is also famous as the one-dimensional version of the map folding problem. But more importantly, compared with the general fold model, a special kind of folding model, *simple fold model*, is for the first time introduced in their research. Their question is, given a one-dimensional crease pattern with every crease pre-assigned mountain or valley, is there a corresponding sequence of simple folds as a flat folding motion? The simple fold model has a predecessor named “pureland folding” [86]. But the definition has not been formalized until Arkin et al. gave the three different kinds of simple fold models. Instead of the general folding which only requires no tearing, no penetrating, and no stretching of the paper, the simple folding restricts every folding to fold some continuous layers on the top of the folded state, such that the next state is also flat. Three simple fold models are defined as *one-layer simple fold model*, *all-layers simple fold model*, and *some-layers simple fold model*, where the numbers of folded layers each time are restricted to one, the number of all the layers, and a number between one and the number of all the layers, respectively. An illustration is given in the top figure in Figure 2.1. From this illustration, we can tell that the simple folds in spirit represents the rigidity between every pair of continuous flat (partly)folded states. In some topics of this thesis, we also use the some-layers simple fold model to derive new variations.

In [9], whether a simple flat folding motion exists or not can be associated with the existence of general flat folding motions. The association origins from the number of dimensions. Later we will introduce that when the simple fold model is applied on papers but not line segments, it performances differently from the general fold model. The simple folding operations in Arkin et al.’s study were further decomposed into two sorts of basic simple folding elements, *crimps* and *end-folds*(see the two figures on the bottom of Figure 2.2). An end-fold is a fold at either the first or the last crease point with the first or final shorter than its neighbor interval. A crimp is a fold along a pair

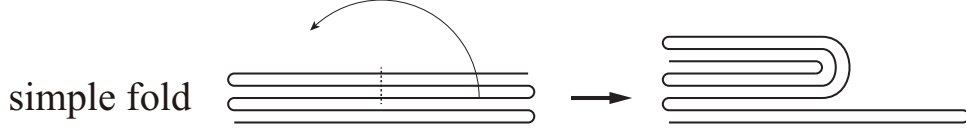


Figure 2.1: An example of some-layers simple fold.

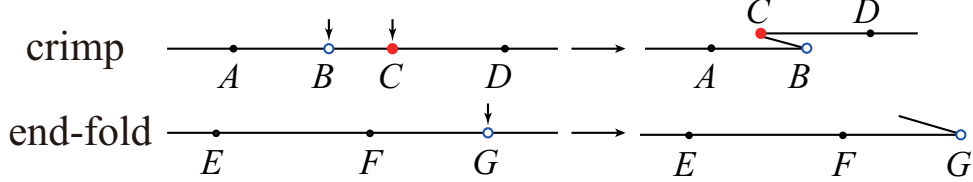


Figure 2.2: The two kinds of basic simple-fold operations: crimps and end-folds.

of consecutive crease points labeled “ $MV$ ” or “ $VM$ ”, where the length of the interval between the two creases is a local minimum value. Some applications of relevant folding models will be introduced in the next section.

The simple fold model is then extended to another folding model by Uehara. Here we refer to it as the *simple folding-unfolding model*, which combines simple folds and the reverse operation, *simple unfolds* as the basic folding operations [97].

An interesting structure inspired by the one-dimensional flat-folding of a one-dimensional crease pattern is the flat-folded state of the segment constructed based on the Cantor Set, designed by Robert Lang [23]. This design shows the potential of one-dimensional flat-folding to be connected with studies about fractions.

There do not exist many results for folding a planar graph to a line at crease points on edges. But as its reverse, the unfold, or we say the flattening, has been rather minutely investigated. The one-dimensional flattenings are always recognized as a sub-problem of the *lock problem*, an investigation on whether a linkage has connected configuration space(*unlocked*) or not(*locked*). The problem also leads to another research topic about the characterization of linkages, always keeping the property locked and unlocked. Most of the early works are included in Ref. [60]. This topic is closely relevant to the *Carpenter’s rule problem*, which asks whether a simple planar polygon can be moved continuously to a position where all its vertices are in convex position so that the edge lengths and simplicity are preserved along the way. As an illustrious result, Connelly et al. gave the conclusion that planar linkages consisting of disjoint polygonal arcs and polygonal chains, with the property that no cycle surrounds another arc or cycle can be continuously moved so that the arcs become straight, the cycles become convex, and no bars cross while preserving the bar lengths [20]. Because during the flattening process, every bar cannot be bent, this problem also revealed the rigidity (similar with simple folds). This property is also discussed in Ref. [10] using graph theory languages. Later, because of the popularity in the practical productions, *rigid folding* is particularly abstracted as a multi-dimensional conception in the field of geometric folding, defined as a piece-wise linear developable surface(or line segments in one-dimensional cases) with a deployment mechanism. The condition to realize the mechanism is that all the faces and creases(or all the segments and crease points in one-dimensional cases) in the crease pattern can be substituted with

rigid panels and hinges(or rigid bars and joints) [91]. The geometrical properties and constraints, as well as some concrete structures, are also introduced and analyzed in Ref. [90,91,94]. One of the most intrinsic properties, which also ensures the popularity of rigid foldings in the practical productions, defined as *self-foldability*, was discussed in Ref. [93]. However, deciding such a favorable property, the rigidity of a given crease pattern, is proven to be NP-hard in Ref. [3] using two-dimensional instances, no matter all the creases or only part of the creases are supposed to be folded. An interesting open question is: does this problem remain NP-hard for one-dimensional instances?

## Two-dimensional flat-foldings and flattenings

The two-dimensional flat-folding is the background of our researches. In computational geometry, the most significant topic is the decision problem on the flat-foldability. Precisely, given a piece of paper and an embedded crease pattern, sometimes with an MV assignment and sometimes not, the decision problem on the flat-foldability asks if such a piece of paper can be folded to a plane with all the creases properly folded.

It was firstly clarified that making decisions on the single-vertex flat-foldability, i.e., local flat-foldability of a single vertex costs linear time [46]. In this simplest case, the basic necessary and sufficient conditions are given for both the cases with and without MV assignments. When the crease pattern involves only a single vertex, the folding model can be reduced to a disk with creases centered at the vertex and then further reduced to the boundary circle of the disk with some folding points, which is essentially a problem in the field of linkage folding.

If no MV assignment is assigned to the input crease pattern, the necessary and sufficient condition is a single condition for the angles as follows. For single-vertex crease patterns, satisfying the flat-foldability also means that there exists a feasible MV assignment for the pattern to be flat-folded.

**Condition 1: *Kawasaki's Theorem*** (Kawasaki [58], Justin [53], [47]): For a flat-foldable vertex, the alternate angles between its incident creases must sum to  $\pi$ .

When the MV assignment is preassigned, the conditions for a crease pattern to be flat-folded become more intricate. Bern et al. used a recursive algorithm to solve the decision problem on the flat-foldability of input crease patterns with MV assignments [14]. Moreover, for such patterns, some well-known conditions are listed below.

**Condition 2: *Maekawa's Theorem*** (Maekawa [57], Justin [53]): For a flat-foldable vertex, the numbers of related creases assigned to be mountains and valleys differ by  $\pm 2$ .

**Condition 3: *Even Degree Theorem*** [47]: For a flat-foldable vertex, the numbers of related creases assigned to be mountains and valleys are both always even.

**Condition 4: *Big-Little-Big Angle Theorem*** [47]: For a flat-foldable vertex, if an angle between its two neighbor creases is a strict local minimum, then the two creases bounding the angle should be assigned differently mountain-valley assignment.

**Condition 5: *Angle Theorem*** [47]: For a flat-foldable vertex, every angle between two neighbor creases is always smaller than  $\pi$ .

These conditions are taken as the theoretical basis of our research. However, in our researches, the crease patterns involve only two sorts, the grid pattern, which involves all the boundary edges of the squares in the pattern, and the *box-pleating patterns* which

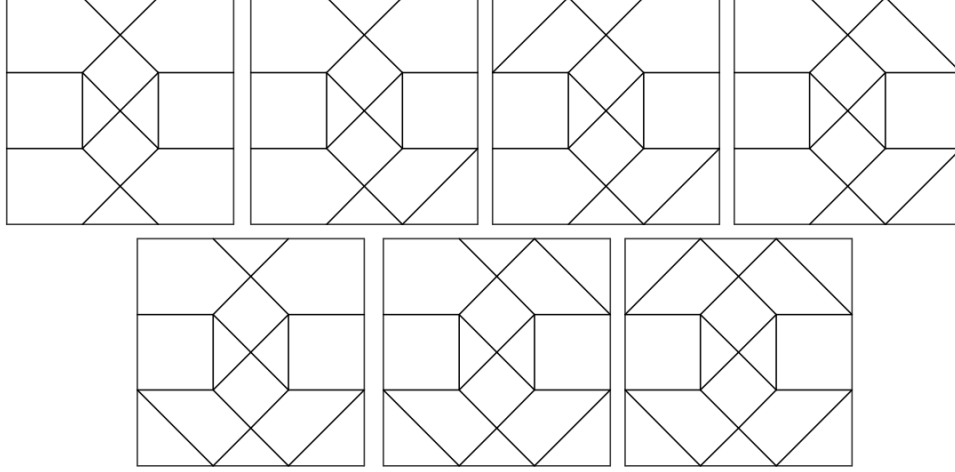


Figure 2.3: The patterns whose every vertex has an MV assignment satisfying the single-vertex flat-foldability but have no MV assignment satisfying the local flat-foldability at every vertex at the same time.

involve part of the diagonal and boundary edges of the squares. Since these crease patterns only induce  $\pi/2$  and  $\pi/4$  angles, Condition 1 and Condition 2 turns out to be locally sufficient for our main use.

The generalization to the usual patterns with more than one vertex derived two questions. One asks if every vertex can be simultaneously flat-folded within a small neighborhood, namely, if there exists a uniform MV assignment following which all the vertices can satisfy Condition 1. This question is called the question of *local flat-foldability of patterns*. The other one asks whether there exists an MV assignment following which the entire pattern can be practically folded to a plane or not. This question is called the question of *global flat-foldability*. The local flat-foldability of a pattern is only a necessary condition for the global flat-foldability of the pattern. Note that the local flat-foldability and the global flat-foldability are rather distinct. As a simple example, Figure 2.3 shows some box-pleating patterns of size  $3 \times 3$  (discovered by Matsukawa et al. [71]) whose every vertex has an MV assignment satisfying the single-vertex flat-foldability, whereas global MV assignments do not exist and thus the entire patterns are not locally flat-foldable. Figure 3.1 is also such an instance in the grid patterns.

In 1996, Bern and Hayes firstly proposed a linear-time algorithm for the decision problem of local flat-foldability of patterns. Then, they proved that the decision on the global flat-foldability of even an everywhere locally flat-foldable pattern is NP-hard [14]. The proof is achieved by a natural matching to the Not-all-equal 3-Satisfiability problem [40], where every single clause (a disjunction of three distinct literals) is matched with a single gadget, and the conjunction of all the clauses is matched to the combination of all the gadgets which form the entire crease pattern. As the simpleness to assign boolean values to make a clause true but the hardness to find a global truth assignment to make the conjunction of all the clauses satisfiable, it is rather comprehensible that to satisfy the flat-foldability of single gadgets, i.e., the local flat-foldability, is relatively easy while to satisfy the flat-foldability of the combination of gadgets, i.e., the global flat-foldability, is difficult.

It was not until 2015 that a small flaw in their proof was perceived by Akitaya et al. [4]. Departing from a similar consideration, Akitaya et al. fixed the proof using box-pleating crease patterns as a replacement, which reinforced the result to the NP-hardness for box-pleating crease patterns (patterns including only horizontal, vertical, and diagonal creases). In the same paper, they further provided the terminologies for the two-dimensional flat-folding, which is an extension of the terminologies introduced in Chapter 11 of [23] for the one-dimensional flat-folding. The properties Consistency (faces with their overlapping orders in the final flat-folded state should form a total order set), Face-Crease Non-crossing, and Crease-Crease Non-crossing, are defined using mathematical languages, respectively. As mentioned before, box-pleating patterns refer to the patterns as subgraphs of square and diagonal grids. Their instances in fact show that the decision on the global flat-foldability of box-pleating patterns of size  $m \times n$  with  $m \geq 4$  is NP-Hard.

These introduced researches concerning the flat-foldability have a common tendency that in these researches only the existence of a legal final flat-folded state, i.e., an ordering of all the faces with Consistency, Face-Crease Non-crossing, and Crease-Crease Non-crossing conditions satisfied has been discussed. However, whether the existence of the final flat-folded state can represent the existence of a folding motion leading to the folded state or not, kept unproved for a long time. Two pieces of research filled the crack in the single-vertex case and the multiple-vertices case, respectively by Streinu et al. [87], and Demaine et al. [30]. They proved that such a folding motion  $C(t)$  with  $C(0)$  as the initial flat piece of paper and  $C(1)$  as the final flat-folded state is path-connected. For the former one, the result holds even when the regions between creases are rigid(not foldable).

Some combinatorial results were given before the decision problem on the global flat-foldability of a multiple-vertices pattern was proved NP-hard. Hull did an enumeration of the MV assignment for single-vertex patterns and proposed some instances showing that the enumeration for a general multiple-vertices case is difficult [49].

For the intractable decision problem on the global flat-foldability, the next objective is to figure out if there exist some tractable special classes where the global flat-foldability can be simply decided in polynomial time. Both the restriction on the crease pattern and the restriction on the folding motions have been considered in our researches, as variations involving grid patterns or box-pleating patterns and the simple fold model. Details will be given in the following sections.

On the other hand, the flattenings have been mainly studied in the two-dimensional case. Since a bijection can be built between every flattening progress and its reverse, the folding progress, the question if every polyhedra can be flattened is equivalent to the question if every folded state of a polyhedral piece of paper is reached by a continuous folding process. Early research mainly provided solutions via disk packing and straight skeleton [23, 24]. They also gave some concrete applications and proofs in Ref. [11, 25]. In Ref. [35], they asked if there exists a net of a polycube which is exactly a rectangle with slits was probed. They gave a general conclusion, that there always exists a rectangle with slits that can fold to  $k$  different polycubes for any arbitrary positive integer  $k$ . For a special class, Bern et al. proved that all the polyhedral surfaces homeomorphic to a disk or a sphere can be flattened [15]. In Ref. [65], some particular sorts of polycubes whose totally flattened shapes after an unfolding along edges can tile the plane were raised. The authors proposed also the question that if or not all the edge-unfoldable polycubes satisfy Conway's criterion. Similarly, the flattenings of some typical polygons



are studied in Ref. [8, 17, 21, 44, 83, 99], and so on. When handling these typical sets, geometric properties has been all of the concern. For example, considering the angles, Biedl et al. showed that the flatten-ability problem can be solved in polynomial time if the dihedral angle at each crease is given, and it becomes NP-hard if the dihedral angles are unknown [16]. The interesting point is that they used only orthogonal foldings in their proof. Similar research combining graph theory concerns whether a plane graph with prescribed edge lengths and prescribed angles can be folded flat to lie in a line [1].

In Ref. [34], a survey has been done on this topic for some early results. Recently, Uehara et al. gave some concluded discussions in Ref. [98].

### Three-dimensional flat-foldings and flattenings

Generally, most of the three-dimensional foldings and curved foldings lead to non-flat folded states. In those foldings, it might be the folding process itself is concerned, rather than only the result overlap or shape. However, the flat-foldings and flattenings still have a vital position in three-dimensional cases, and in an even higher dimension because of the feasibility in practice. The first research concerning this topic can be dated back to 1989 by Kawasaki [58]. He studied flat origami in a  $d$ -dimensional Euclid space and provided a necessary condition for the flat-foldability. A question proposed by Demaine et al. in 2001 asks which kind of polyhedral complexes in  $d \geq 3$  dimensions can be flattened (i.e., have flat-folded states) [33]. This question has only been partly solved in Ref. [18, 36], and so on. Even though the theory for multi-dimensional flat-folding and flattenings is still immature, some mathematical models are already put into practical architectural designs [42].

### 2.1.2 Other theoretical topics in computational origami

Introduction in this part aims to give a concise overview of computational origami and other theoretical topics (mainly mathematical topics) in origami. The theme of this paper, map folding, is a branch developed from the computational origami and categorized in the flat-folding problems, as introduced in the last part. However, besides the flat-folding problems and flattening problems, there are still many important topics in computational origami on the theoretical level. Here we take an overview of these topics.

In the field of computational origami, numerous topics were yielded and introduced in an earlier literature [23], like the afore-mentioned flat-folding, flattening polyhedron, as well as some other problems like wrapping, cut-and-fold, and so forth. Since then, relevant topics and open problems have been studied in a broad range.

In one-dimensional foldings, the configuration space, i.e., the set of reachable points of a given linkage, is of vital concern because the configuration space indicates rigidity. In [23], they gave the conclusion that rigidity corresponds to an isolated point in configuration space and lockedness to a disconnected component in configuration space. Also, many early results are mentioned in this book. For other recent results, one can refer to the survey [96]. These results drive applications in the design of robots, mechanisms, bending machines, protein foldings, and so on.

In two-dimensional foldings, an important direction other than the flat-folding is origami design. It was described as problems of folding a piece of paper into a desired

folded shape or with some other certain properties. The most famous early design is the Tree Method by Lang [64]. Its significance directs from the idea of designing origami bases, which is also studied by Meguro [72, 73]. The core of such an idea is: first, creating a stick figure to represent the basic shape, then, using the circle packing to distribute the vertices of the stick figure into distinct areas, and finally, connecting the center of the disks to achieve the corresponding crease pattern. Since then, some other software and applications can be found in Ref. [48, 74–76]. Also, a problem related to both flat-folding and origami design, the cut-and-fold problem, is folding a piece of paper to align a prescribed graph such that the desired shape can be obtained by this folding and one complete straight cut along a segment. Some early results can be found in Ref. [51]. Recently, inherited the Cut-and-Fold problem [23] and the Tree-Method design, a generalization covering both, named as the hole problem, was studied in Ref. [32]. They showed that given a sheet of paper and a prescribed folding of its boundary, there always exists a consistently isometric and polynomial-time computable mapping of the polygon interior from the initial sheet to the final flat-folded state. Distinct from it, another interesting research in Ref. [7], did not view the cut-and-fold in a puzzle-like way, but rather as an abstract computational prototype. They proposed a computational model considering the simple folds and unfolds, straight-line cuts on a sheet of paper with holes as a (deterministic) fold-and-cut machine and compared its performance with Turing machines.

There are not so many studies about foldings beginning from a three-dimensional or multi-dimensional object. However, if we only concern the final folded state of the paper, curved foldings would certainly be categorized to three-dimensional folding since they cannot be flat-folded. The beginning of such research can be dated back to 70s [45], where special kinds of geometrical objects are used to generate a representation of the folded state of a piece of paper along some curved creases. Following Huffman’s idea, later research used *rule segments* to model the phenomena of curving on the surfaces (which are called faces when the creases are line segments). Furthermore, to describe to which extent the paper is folded, new terminology as *folding angles* are introduced. With these conceptions, the geometrical properties during the folding motion can partly be analyzed in a theoretical way using differential geometry and some other tools. We list some significant references in this field as follows [26, 27, 59, 77, 92].

Besides all the above-mentioned computational studies, some origami topics in the fields of pure mathematics, history and pedagogy are also flourishing. All these researches together present the diversity of the development of origami on the theoretical level.

## 2.2 Practicality

The applications of geometric foldings encompass a wide range of subjects such as robotics, architecture, mechanisms, designing systems, and so on. These developments have been significantly multiplied since the last decade. Considered as the two-dimensional congruent transformation, or as the combination and arrangement of faces, the primarily concerned geometric properties of origami patterns include flexibility, deployability, rigidity, etc. These are generally accounted as conventional factors to be considered in origami-based structures and mechanisms. The mathematics of origami is correspondingly viewed

as also a methodology to help to model the design and analyze the properties of practical productions. From this viewpoint, many research regarding origami geometry in the practical uses may be seen as outcomes of the origami in applied mathematics. In the settings of our topics, the linear-time results, the restricted folding operations in the problem settings, and the input crease patterns, all of them are closely related to the practical uses in the structural designs, mechanisms, robotics, puzzle designs, and so forth.

### 2.2.1 Structural designs and mechanisms

The flat-folding has been popularly employed in the mechanism designs because of its controllability. Under this prerequisite, it is unnecessary to add angle-setting components to the entire processor. This favorable property further shows the possibility to automatize the manufacture. For example, instead of the traditional manual processing, flat-folding makes the processing much more possible for robots. To provide the feasibility when put into practical productions, in some of our researches, we especially concerned the simple folding operations and their reverse, the simple unfolding operations. We have mentioned that even with the restriction on flat-folding, as long as the material is not soft enough, it is always not even possible to put general folding operations into real processings. On the opposite, although the simple folding leads to only a subset of the set of folded states corresponding to the general folding, the practical foldings can always be implemented along single straight lines by either man or robots.

On the other hand, because of its potential in transforming flat sheets of materials to three-dimensional structures, physical applications of origami primarily entail the mechanisms which are controllable, producible, and capable of being scientifically modeled and quantified. Also, the compatibility of the mechanism with particular materials is of increased importance. The foci of research about structural origami are varied according to diverse requirements. Even though there is no general rule to assess which design serves the requirement better, crease patterns with regular angles between creases are always much more deeply and broadly studied so as to benefit the practical requirements. Such patterns can be periodically arranged to change the scale while keeping most of the geometric properties. Therefore, some particular patterns have been usually chosen as bases because of their superior performance. For example, the most famous pattern, Miura Pattern, has been used firstly in the design of solar panel arrays for space satellites and then in furniture and architectural designs, metamorphic mechanisms, surface or curve approximations [55, 56, 95], and so on. Miura Pattern has been so broadly used mainly because of the rigidity of its folding operation and its one-degree-freedom property (the motion of one crease uniquely decides the motion of all the other creases). It is also considered as a way to fold a map, which is closely relevant to our topics. As another example, the Yoshimura Pattern [100], is often used in the design of origami crash boxes and applications such as transport vehicles because of its remarkable energy absorption capability [102, 104]. In fact, when the acute angle of Yoshimura Pattern is restricted to be of degrees 45, it also becomes a particular kind of box-pleating pattern. That means, most of the results which hold for general Yoshimura Patterns can also be viewed as the results of box-pleating pattern. As the applications of this kind of pattern, the mechanisms applied for energy absorption show another significant achievement of origami-derived

structures. They have become especially of importance as the increasing of requirement on devices saving people from vehicle collisions. This theme, the analysis of the collapse of certain structures, is historically important. The earliest research took cylindrical shells made from Yoshimura Patterns and some other patterns as the objects [5]. In Ref. [69], the energy absorption behaviors of plenty of different kinds of structures and materials were concluded, also involving the box-pleating patterns. As origami being associated with this topic and thin-wall structures being put under expectation, conventional tube structures made of these patterns have been broadly studied [67, 102, 104].

Beyond the above special kind, the general box-pleating patterns studied as the focus in half of our research also have many applications in the structural mechanism designs, especially for the designs in orthogonal shapes. For a pure shape design, a universal method is provided in Ref. [13], which uses box-pleating patterns to achieve all possible kinds of shapes combined by orthogonal cubes. In Ref. [89], the rigidity of some mechanisms based on box-pleating patterns was discussed, which reveals its possibility in the applications of self-folding designs (will be introduced in the next section). As the theoretic basis of mechanisms based on the box-pleating patterns, Ref. [50] analyzed some properties of the folding model.

An overview of the recent designs of origami-based mechanisms was presented in Ref. [103]. It was emphasized that the flat states as both the initial state and the final state are usually desired. However, as they benefit the practical application and development of origami-derived mechanisms to be under control, they also highly limit the configuration of designs. Therefore, the combination of the folding directions of creases thus can be considered as the foci to increase the variety of these designs. In our researches about the box-pleating patterns, instead of the MV-assigned crease patterns, we studied the unassigned patterns, which have higher freedoms when applying to practical uses. Our linear-time algorithms for obtaining a feasible folding process of the flat-foldable input patterns further reveals the applicability of them in practical applications.

## 2.2.2 Self-Folding mechanisms and robotics

The simple folding is essentially the realization of rigid folding operations in flat-foldings. And at the practical level, an important application of rigid folding is self-folding. Some introductions to their folding mechanisms and the relationship between them can be found in Ref. [89, 93]. Self-folding origami can be directly used in robotics and machines. Ref. [85] shows the possibilities to realize self-folding paper robots with low costs. The characteristics of a self-folding robot even make itself capable of conducting basic tasks and behaviors, including swimming, delivering, climbing, digging, and so on [77]. Surprisingly, their design uses the box-pleating patterns as the base, only with a little revision around one vertex. Ref. [38] gives a method for building self-folding machines.

Results in Ref. [82] shows that compared to the general origami patterns with multiple folding motions, the folding motion of a self-folding pattern should be usually unique. Furthermore, from the viewpoint of energy cost, self-folding requires the least energy among all kinds of foldings. On the other of the dual directions, from whether the folding motion is unique or not, we can partly imply whether the corresponding crease pattern can be self-folded or not. The uniqueness of the folding motion must lead to the uniqueness of the final folded state. In this opinion, it matches with our research on the valid folded

states using the simple fold model.

In some cases, a self-folding model is desired to be folded symmetrically in multiple directions. Designs based on Miura Pattern cannot satisfy this requirement, whereas other patterns like grid patterns are considered to be with good behavior. In Ref. [22], the self-folding performance of a grid pattern under a high temperature was tested with the model made from a biaxially oriented polystyrene sheet. They analyzed the effect of the grid geometry, the grid size, and the power of the laser on the bending curvature. Also, other than the pattern, the performance of self-folding also depends on the materials. In such cases, to make the folding easily controllable, the creases computably and regularly positioned in the pattern are desired. Also, since self-folding is always realized by temperature-control on heat-shrinkable polymer sheets or electricity-control of conducting and insulating materials, to make the finished products more flexible on shapes, properly removing some creases from the regular patterns becomes necessary. Box-pleating patterns just satisfy these needs. Our results about the flat-foldability of box-pleating patterns clarify the properties of these self-folding models. Referencing the research in Ref. [41], which concerned self-folding planes able to change the form to goal shapes of arbitrary local Gaussian curvature via uniform heating, the development of our results can also provide the theoretic basis for designing and analyzing such self-folding planes.

Self-folding is also closely related to robotic designs. Also, in the self-folding field, a platform to simulate robotic metamorphosis by origami exoskeletons was provided in Ref. [78]. In their method, they used the same simple folding operations as in our research. This result reveals a possible future development of our paper folding models in the cross-field of self-folding and robotics.

### 2.2.3 Puzzles

Puzzle designs based on origami is a very conventional application. For example, Dieleman et al. designed a Jigsaw puzzle based on the pluripotent origami [37]. Demaine et al. keep it a custom to use origami to design font puzzles in many ways [28, 29, 31]. In Ref. [66], origami is used to design a puzzle the same as David Mitchell 's Columbus Cube.

Both two themes in our researches can be directly used in producing puzzles. It is very natural to consider the two games below. (1) Print a box-pleating pattern on the paper and let a person determine if the pattern can be flat-folded along and only along the printed creases. (2) Print a grid pattern with its squares marked by numbers on the paper and let a person find a folded state such that these squares are ordered just as the numbers from bottom to top.

Especially, we want to note that our last topic about the boundary overlapping orders is extremely appropriate for the puzzle design. The boundary overlapping order itself is just very interesting. Besides, it is a partial order rather particular since compared with its linear-time solution, most of the partial orders are supposed to lead to intractable results.

As the presenter of such puzzles, we certainly know the answer because our corresponding algorithms are time linear in the size of the input. Such puzzles are also feasible because we use only simple patterns. These puzzles can also be implemented to

design not paper puzzles but digital puzzles as software or little Apps.

## 2.2.4 Other applications related to our research

In the above sections, we talked about the designs and applications related to the patterns and folding operations in our research. Their popular applications rely on favorable properties like flexibility, deployability, rigidity, tractability, and so forth. In this section, we will introduce some other possible practical use of our results. Since most of the problems we studied are linear-time solvable, they bring about the applicability in designs using kirigami, nanoscale materials, thick materials, and many others.

A methodology to improve the performance of a pattern is differentiating some particular parts from other parts of it. Either these parts are supposed to be made by different materials, or some cuttings are considered to be implemented on them. The latter methodology is also called *kirigami*. Usually the locations of creases are considered firstly. Ref. [88] gives a guidance of the kirigami used to transform flat surfaces into three-dimensional structures. In their introductions, grid patterns, box-pleating patterns, are used as the basic patterns together with generalized Yoshimura Patterns and Miura Patterns. Focusing on the material and thickness, in Ref. [43], the practical use of structures with non-negligible fold thickness or with maximum curvature was considered rather than a good performance of a pure model. With this premise, a model whose creases are not rigidly but smoothly folded was developed. In Ref. [19], tessellations of the kirigami patterns were used to create curved surfaces whilst Yuan et al. provided the circular origami tessellation to approximate flat surfaces [101]. For all the methods and topics involved in these researches, grid patterns and box-pleating patterns show their feasibility and potential.

Another field on the frontier that should be mentioned is the designs with nanoscale materials, with main objectives to generate specific shapes [84]. Usually, these designs are derived from DNA origami structures since the attention is mostly focused on the creases but not the faces. For example, a scaffolded DNA origami that can be autonomously designed showed enough potential to be applied to nanoscale materials science and nanotechnology [52]. While in such designs, repetitious patterns can only form limited sorts of shapes. The frame formed by the creases in box-pleating patterns with some unassigned ones can increase flexibility.

# Chapter 3

## Topics in This Thesis

Our topics in this thesis mostly belong to computational origami, which concerns the flat-foldability problems and flattening problems. We take crease patterns and orderings of the faces as inputs in our problem settings. The former corresponds to the initial state, an unfolded piece of paper. Thus the output of it is the decision(yes or no) on its flat-foldability. The latter corresponds to the final folded state of a piece of paper, and we aim to find out whether such a state really exists(without self-penetration). In this chapter, we will introduce the two big themes and the concrete settings of our topics systematically.

### 3.1 Three Kinds of Variations of Map Folding

Generally, the existed research concerns three kinds of variations of map folding, including restrictions on the size of the map, restrictions on the folding model as well as variations on the grid pattern. The first two are specializations from general cases to restricted cases and thus are making the problem becoming easier, while the last one is a generalization that makes the problem harder.

For the first one, limiting the size of maps, three cases as maps of size  $1 \times n$ ,  $2 \times n$ , and one-dimensional maps (which is not strictly a restriction), have been mainly studied. For maps of size  $m \times n$  with both  $m$  and  $n$  bigger than 3, there exists no result.

Maps of size  $1 \times n$  are trivial (all flat-foldable) with MV assignment as inputs, while the complexity increased rapidly in the maps of size  $2 \times n$ . In this case, Morgan [79] gave an  $O(n^9)$  time algorithm to solve the decision problem. This problem is surprisingly so not trivial that even for a map as smaller as of size  $2 \times 5$ , there exist inputs that cannot be flatly folded (see Figure 3.1).

The slightly harder version of maps of size  $1 \times n$ , one-dimensional maps with generalized lengths between creases as shown in Figure 3.2, are studied by Arkin et al. [9]. It was proved that for one-dimensional maps, the flat-foldability can be determined in  $O(n)$  time.

As introduced before, in their research, the simple fold model is for the first time introduced. Instead of the general folding which only requires no tearing, no penetrating, and no stretching of the paper, the simple folding restricts every folding to fold some continuous layers on the top of the folded state, such that the next state is also flat (see Figure 2.1). They proved that the existence of such a simple folding directly corresponds

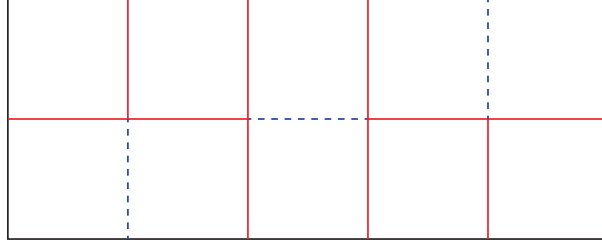


Figure 3.1: Minimal unfoldable map of size  $2 \times 5$ . The red solid line segments indicate mountains, whereas the blue dashed line segments indicate valleys.

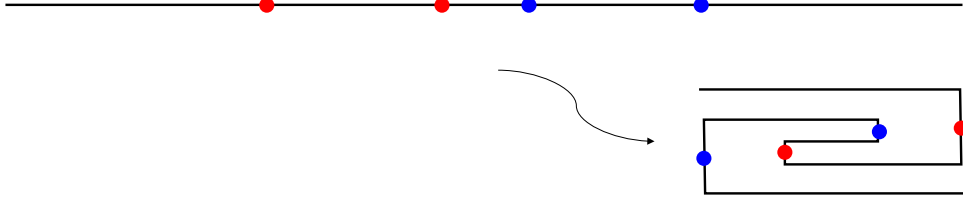


Figure 3.2: One-dimensional maps with generalized lengths between creases. The red points indicate mountains, whereas the blue points indicate valleys.

to the existence of general flat foldings. The simple folding in their study is further decomposed into crimps and end-folds. After their research, the simple fold model is extended to another folding model by Uehara. Here we refer to it as the *simple folding-unfolding model*, which combines simple folds and the reverse operation, *simple unfolds* [97]. Uehara applied this model to maps of size  $1 \times n$  and proved that this model is in fact as efficient as the general fold model in maps of size  $1 \times n$ . Moreover, he proposed a linear time algorithm to decide the flat-foldability of maps of size  $1 \times n$  [97].

The above-mentioned folding models also induced the variations of folding maps of size  $m \times n$  (the second direction of the variations). In maps of size  $m \times n$ , even though the flat-foldability in the general fold model no more keeps the equivalence with the simple fold model, whether an input map can be flat-folded to the unit size  $1 \times 1$  by simple folds turned out to be linear-time solvable [23].

The last sort of variation, generalizations of the grid patterns, lead to studies of box-pleating patterns (See Figure 3.3) and general patterns. Box-pleating patterns refer to the patterns with the patterns whose creases are only on a subgraph of a square and diagonal grid. The name comes from their most popular application- constructing arbitrary polycubes (boxes). Historically, the box-pleating patterns were developed by Neal Elias [63], and popularly used in art works and designs [48].

The first analysis on the complexity about box-pleating patterns was presented in Ref. [62]. Later, as aforementioned, the NP-hardness to decide the flat-foldability of box-pleating patterns is proved by Akitaya et al. [4] after an earlier version given for the more generalized arbitrary pattern by Bern and Hayes (although with a few flaws) [14]. Both NP-hard results hold no matter the MV assignment is predefined or not. A special case of the arbitrary patterns is that all the creases converge to one intersecting vertex, which makes the decision on (global) flat-foldability reduced to the decision on the *local flat-foldability* solvable in linear time. The complexity remains linear even when the question is changed to ask whether all the vertices in a given pattern can be locally flat-folded



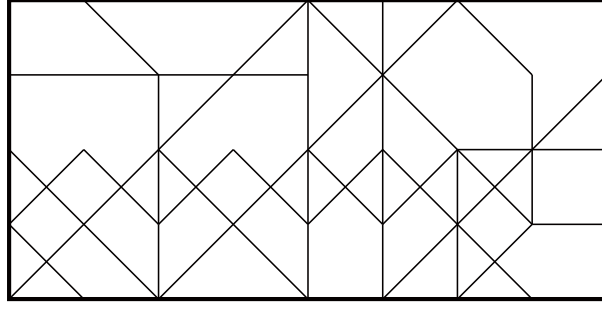


Figure 3.3: The box-pleating patterns.

simultaneously [14]. Also, when the objects are general patterns, the flat-foldability in the simple fold model is proved to be NP-hard [2].

The proof of Akitaya et al. gives the NP-hardness for box-pleating patterns with size larger than  $4 \times n$ . Our research thus mainly takes the  $1 \times n$  and  $2 \times n$  box-pleating patterns as the targets. The  $3 \times n$  case has been remained as an open problem.

## 3.2 Another Respective - Valid Overlapping Orders

As the computational complexity of the map folding problem remains unclear, only an exponential-time algorithm exists [80]. The strategy is enumerating all the possible orders of the  $m \times n$  squares and check each of them on whether it represents a non-intersection valid flat state. The positive ones are then called *valid overlapping orders*. Only in the case that no valid overlapping order exists, the decision on the flat-foldability would be no. In other words, as long as a valid overlapping order exists, the input map definitely can be flat-folded. The existence of the corresponding folding process is given by Demaine et al. [30]. The decision problem on whether an input order is a valid overlapping order or not is called the *valid overlapping order* problem. If we analyze the map folding problem with the non-deterministic Turing machine model, an order given on the set of the squares can be viewed as a guess given by the guessing module of the machine.

Without many existing studies for the valid overlapping order problem, we considered its combination with the simple fold model introduced in the last section. Note that although the restriction of simple folds makes the map folding getting simpler, it makes the valid overlapping order problem more complicated. We first considered the validity of input overlapping order in the simple fold model for maps of size  $m \times n$ . Since the MV-assignment is essentially a partial order given on the set of squares, we further considered the validity of partial overlapping orders and proposed the solution for the boundary orders (total orders given on the set of all the squares aligning on the boundary of the map) for maps of size  $m \times n$  in the simple fold model.

To conclude the value of our research, even though the original map folding problem seems intricate, our studies on the variations provided comparable instances to trace the computational complexity of it. And more importantly, many variations are rather interesting and valuable as research topics themselves.

# Chapter 4

## Flat-foldability of Box-pleating Patterns

Our research on the flat-foldability of box-pleating patterns focuses on if an input box-pleating pattern can be folded flat with general folding operations. Even though this decision problem is proven to be NP-hard for  $m \times n$  patterns for any  $\min\{m, n\} \geq 4$ , there may exist polynomial-time solutions for the flat foldability of box-pleating crease patterns when their sizes are restricted. We put our attention on the box-pleating patterns of smaller sizes to find whether the solution can get simpler. We solved the decision problems for box-pleating patterns of size  $1 \times n$  and  $2 \times n$  and proposed corresponding algorithms to output feasible folding processes when the answers to the decisions are “yes”, respectively. Our results show that every box-pleating pattern of size  $1 \times n$  and  $2 \times n$  is globally flat-foldable as long as it is locally flat-foldable. The  $3 \times n$  case is then left as an attractive open problem.

### 4.1 Box-pleating Patterns of Size $1 \times n$

The first topic we discussed takes the box-pleating patterns of size  $1 \times n$  as inputs. We proved that the flat-foldability of any input box-pleating pattern of size  $1 \times n$  could be decided in  $O(n)$  time. These patterns only involve a subset of the creases and no MV assignment. The conclusion essentially reveals the equality between local flat-foldability and global flat-foldability. We proved this by a folding-unfolding process. Firstly fold all the creases, even the ones that actually do not exist. Then, we use an unfolding process to unfold the vertical ones and diagonal ones in order, with some *flips* along the creases on the flat-folded state to make it always in the shape of a *zigzag*. A flip is an interchange of the MV assignment on the corresponding crease. By “zigzag”, we mean the certain states that when folding the map from one end to the other, the faces on the right side is always supposed to be folded above or to the same layer as the faces on the left side. This solution also constitutes part of the folding method in the box-pleating patterns of size  $2 \times n$ .

Besides, we also provided similar results for the box-pleating pattern consisting of all the vertical creases and all the diagonal creases and with an MV assignment of size  $1 \times n$ . We proved that by (1) folding the squares from one end to the other and (2) always seeing the overlapping faces as a single face in the reduced map, every locally flat-foldable

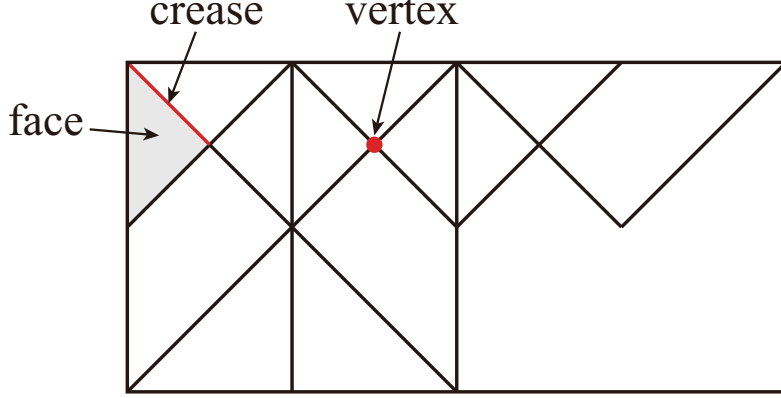


Figure 4.1: Basic notations in a box-pleating crease pattern.

pattern can be globally flat-folded.

#### 4.1.1 Notations

The input box-pleating crease pattern of size  $1 \times n$  is denoted by  $M$  with two distinctive sides, the front and the back. Following the terminology in Ref. [9], the line segments in the crease pattern are called *creases* and the endpoints of creases inside  $M$  are called *vertices*. The regions bounded by a set of creases or by part of the boundary of the paper, and without creases inside are called *faces*. Two faces are called *neighbors* if they share a crease. Two faces are said to be *adjacent* in a certain folded state (maybe a middle state) if their surfaces touch and overlap each other. A vertex can have single or multiple incident creases, while a crease incident to exactly two faces. These notations are illustrated in Figure 4.1. In our topics, sometimes MV assignments are given together with the input crease patterns.

The sequence of partly flat-folded states is indicated by  $R = (R_0, R_1, R_2, \dots, R_t)$ , where  $R_0$  represents the initial state of the input pattern before any fold and  $R_t$  represents the final folded state. When the creases cover all the diagonals and all the vertical edges between two neighbor squares, the final state would be in the shape of a triangle. Otherwise, the final state can be achieved by firstly folding all the diagonals and vertical edges and then unfolding some creases that do not really exist. These creases are called the *imaginary creases* in the following, while the creases that really exist are called *real creases*. Corresponding to each input pattern, we define the configuration space  $\mathcal{R}$  comprising of all the possible partly flat-folded states. Then, any fold connecting two partly flat-folded states can be described as a mapping  $f : \mathcal{R} \rightarrow \mathcal{R}$ . Specifically, in a given sequence, a mapping from the state  $R_{i-1}$  to  $R_i$  is denoted by  $f_i$  where  $f_i(R_{i-1}) = R_i$ . Therefore, the entire folding process  $F$  is defined as a sequence of mappings  $f_1, f_2, \dots, f_t$ . This is to take advantage of inverse functions when defining unfolding operations. For convenience, an unfold which undoes  $f_i$  is denoted by  $f_i^{-1}$ .

Since the local flat-foldable condition is a prerequisite of the global flat-foldability, for all the following decision problems on the flat-foldability of crease patterns, the initial step is a filtering of the locally flat-foldable patterns. Referencing Ref. [14], the method of filtering can be finished in time linear in  $n$  for any  $m \times n$  pattern with an constant  $m$ .

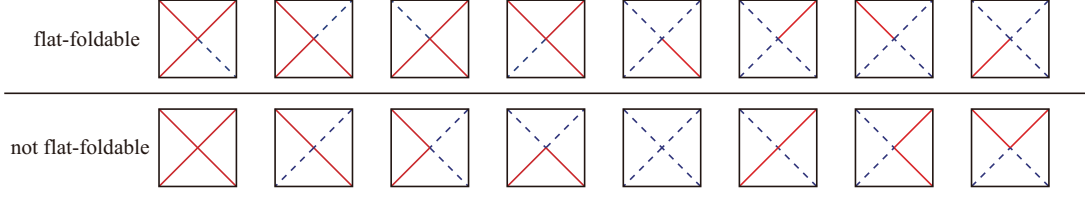


Figure 4.2: 16 possible combinations of Ms and Vs within a square. Eight MV assignments on the upper row are flat-foldable.

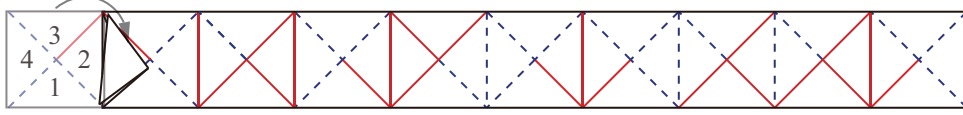


Figure 4.3: Result state of the first step applied on the instance Figure 1.3.

#### 4.1.2 Patterns with all the creases and the MV assignment

When all the creases and the MV assignment simultaneously exist and comprise the input, there are  $2^4 = 16$  kinds of possible combinations of Ms and Vs within a square. Based on Maekawa's Theorem, which turns out to be the necessary and sufficient condition for the flat-foldability of a single square, eight of the 16 patterns are testified to be the local flat-foldable ones (see Figure 4.2). Note that all these local flat-foldable patterns can be flat-folded via two simple folds, each along a diagonal.

A folding process inducing a reducible transformation on a locally flat-foldable input pattern is used as the principal tool in this problem. The description of the folding process is as follows.

In the first step, the diagonal creases of the first square and the vertical crease between the first and the second square are supposed to be folded in this order. The result is a strip of size  $1 \times (n - 1)$  with a triangular shape of four layers overlapping a triangle on one end. For example, the result state of this step applied on the instance Figure 1.3 (1-1) is illustrated in Figure 4.3.

In the next step, the overlapping triangular faces are viewed as a single layer, and thus the size-shrunk strip is viewed as the same kind of pattern of size  $1 \times (n - 1)$ . Apply the folding on each square in order until all the squares are folded. Finally, a flat-folded state composed of  $4 \times n$  layers and in the size of a triangle (a quarter of the square) can be obtained.

The whole process involves  $3 \times n - 1$  times of simple folds (folding along a line segment). Among them,  $2 \times n$  times of simple folds are folds along the diagonals; the remaining  $n - 1$  times of simple folds are folds along the vertical creases. Hence, the time complexity is  $O(n)$ . We list the pseudo-code in Algorithm 1. When the input pattern is flat-foldable, this algorithm further outputs a valid overlapping order in the final state.

The main conclusion is described as Theorem 4.1.1.

**Theorem 4.1.1.** *Let  $M$  be a sheet of paper of size  $1 \times n$  with a box-pleating crease pattern with every possible vertical and diagonal crease, and with an MV assignment as well. Then, any  $M$  satisfying local flat foldability can be flat-folded. Moreover, finding a way to fold  $M$  takes linear time.*

**Input** : A box-pleating pattern  $M$  with all the possible creases and an MV assignment  
**Output**: Yes or No, when Yes, also output the overlapping order  $O$  of  $M$  in the final flat-folded state.

**initialization**

$O \leftarrow \emptyset$  // the decided overlapping order at each step

**if** *the MV assignment in the first square is locally flat-foldable* **then**

    Deciding the inner order of four faces in the first square according to the MV assignment, append the four faces to  $O$

**else**

**return** *No*

**end**

Remove the first square from  $M$

**foreach** *square  $S$  in  $M$*  **do**

    Update  $O$  according to the vertical crease left to  $S$

**if** *the MV assignment in  $S$  is locally flat-foldable* **then**

        Consider the folded layers below the leftmost triangle and the leftmost triangle in  $S$  as a single layer

        Decide the order of the triangle layers according to the MV assignment within this square, update  $O$

**else**

**return** *No*

**end**

**end**

**return**  $O$

**Algorithm 1:** Linear-time algorithm for deciding the flat-foldability of a box-pleating pattern of size  $1 \times n$  with all creases and an MV assignment.

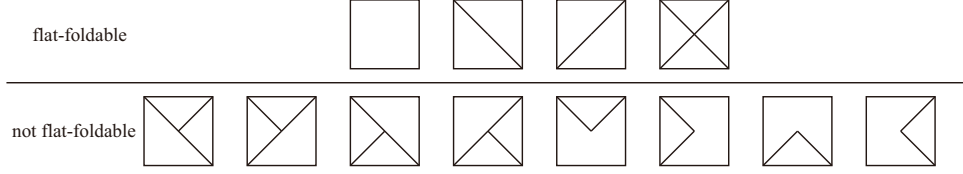


Figure 4.4: Flat-foldable and not flat-foldable crease patterns in a single square.

The correctness of Theorem 4.1.1 follows the folding process we gave above.

Trivial as this folding process is, it reveals some interesting points when considering other kinds of input patterns as well. In this  $1 \times n$  regular box-pleating pattern, every four creases inside a square form a group and are independent of the folding of the rest squares. The critical point for flat-foldability sometimes relies on this independence. Similar to this pattern, when considering every  $2 \times 2$  area in a locally flat-foldable  $2 \times n$  grid pattern (a regular map), even though the four squares can definitely be folded to consecutively overlapping layers, their folding influences the folding operation of the two squares in the next column at the same time. It may help to build a better understanding of why there exist some maps of size  $2 \times n$  not foldable from this perspective.

### 4.1.3 Patterns with part of the creases but no MV assignment

The condition of local flat-foldability is equivalent to restrictions on the diagonal creases in the box-pleating patterns of size  $1 \times n$  with part of the creases but without MV assignment. Only three possible cases hold, when there are 0, 2, or 4 creases inside a single square, and all the existing creases form a diagonal or two diagonals passing through the square (See Figure 4.4).

For the local flat-foldable cases, we proved that all of them could be flat-folded by providing a specific folding method. The key is to find a valid folding where self-penetration would not happen. Our method ensures this by taking the shape of a zigzag as the objective of every step. Briefly, our approach is a process as follows: first, fold the input pattern to an initial flat-folded state in a zigzag shape with imaginary creases; then, unfold the imaginary creases from the initial flat-folded state. To maintain the shape of a zigzag, after the unfolding, some real creases folded before may have to change their MV assignment. These changed creases can be computed during the same check, which is realized by a traverse of the creases in the input pattern, while for the clearness, following we describe the corresponding changing operation after each unfolding operation.

Using the result introduced in the last section, first, we assume that all the creases exist, such that we can apply a certain MV assignment (locally flat-foldable) to make the whole pattern globally flat-foldable with the imaginary creases. The specified MV assignment is illustrated in Figure 4.5.

In such an assignment, the notation of the creases are illustrated in Figure 4.6. The squares are labeled from  $s_1$  to  $s_n$  in order from one end to the other. Each vertical crease incident to  $s_i$  and  $s_{i+1}$  is denoted by  $v_i$ . Each four diagonal creases inside  $s_i$  are labeled  $d_{i,1}$ ,  $d_{i,2}$ ,  $d_{i,3}$ ,  $d_{i,4}$  in counterclockwise order. Vertical creases  $v_{2j}$  and diagonal creases  $d_{2j-1,2}$  and  $d_{2j,1}$  are labeled valleys, while the other creases are labeled mountains, as shown in Figure 4.5.

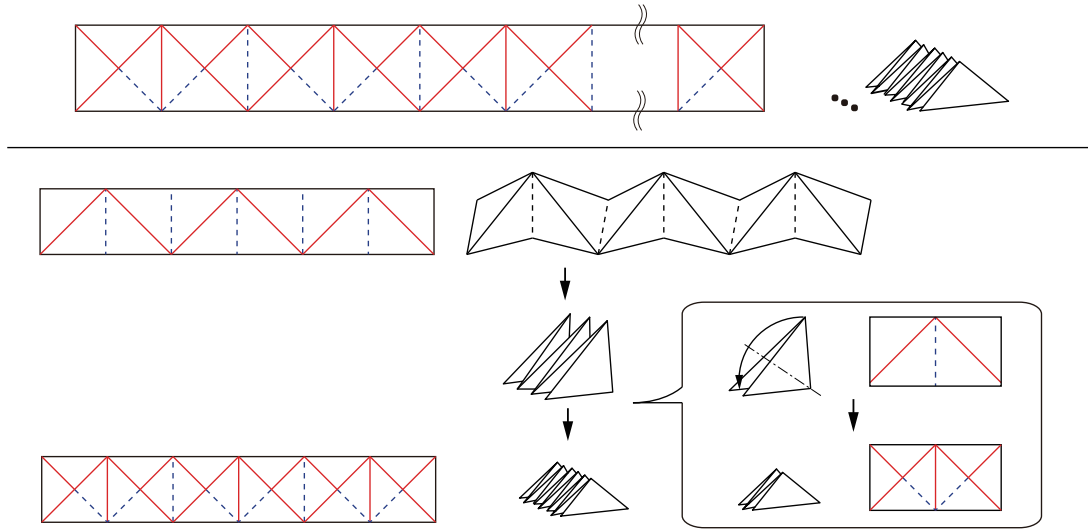


Figure 4.5: The initial MV assignment and folded state before the unfolding operations.

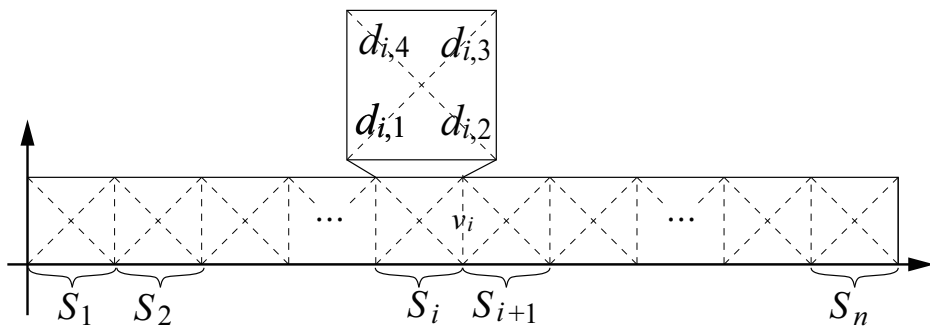


Figure 4.6: Notations of the box-pleating patterns of size  $1 \times n$ .

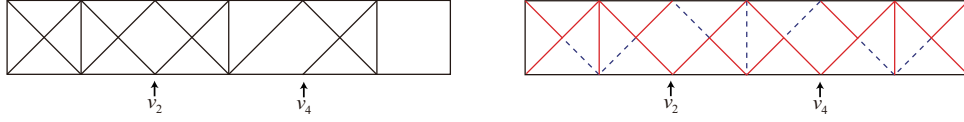


Figure 4.7: An input box-pleating pattern without any MV assignment and the pattern after the unfolding of vertical creases.

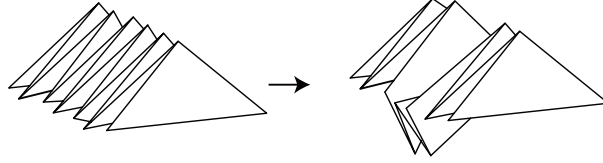


Figure 4.8: Flat-folded states before and after unfolding the vertical creases

With a little change from the folding method introduced in the last section, this time, we want to fold the pattern into the shape of a zigzag. Thus, our strategy is to fold all the diagonal creases firstly and then fold the vertical creases since it is clear that the vertical creases themselves form pleats. In the overlapping of the final flat-folded state, every four triangles in the same square form continuous adjacent layers. More precisely, the ordering of the layers in the final flat-folded state can be described as the triangles of  $s_1$ , the triangles of  $s_2$ ,  $\dots$ , the triangles of  $s_n$ . An illustration of the final flat-folded state is shown in Figure 4.5.

From such an initially over-folded state, we apply unfolding operations. The imaginary creases are supposed to be unfolded in order. The firstly unfolded ones are the vertical imaginary creases. We unfold them from one end to the other. To figure out how we could manage to unfold a vertical crease without breaking up the shape of zigzag (the overlapping order), we start the discussion in terms of one vertical crease. Before unfolding it, the layers are divided into two disjoint parts by it, and clearly, the unfolding operation causes a part to move to the same layers(in the overlapping) as the other part. In this case, we change the MV assignment for every vertical crease  $v_j$  ( $j \leq i$ ) and exchange the labels for every pair of diagonal creases on the same line but with different labels. Namely, the labels of  $\{d_{j,1}, d_{j,3}\}$  and  $\{d_{j+1,2}, d_{j+1,4}\}$  would be exchanged when  $j$  is odd and the labels of  $\{d_{j,2}, d_{j,4}\}$  and  $\{d_{j+1,1}, d_{j+1,3}\}$  would be exchanged when  $j$  is even. In the example illustrated in Figure 4.7, the crease  $v_2$  is firstly unfolded, and then  $v_4$ . The labels of the vertical creases and diagonal creases are correspondingly changed. The process of the unfolding is shown in Figure 4.8.

Now, only the imaginary diagonal creases remain to be unfolded. These creases have distinct properties with vertical creases since some unfolds would change the overlapping order of the faces globally, while some may not destroy the shape of zigzag. Our strategy correspondingly contains two steps as follows.

Step 1. Unfold the imaginary diagonal creases in a square  $s_i$ , which will let four continuous layers in the overlapping become two or a single layer.

Step 2. If an unfold makes the layers comprised by the squares  $s_j$  for some  $j$ s ( $\forall j, j > i$ , or  $\forall j, j < i$ ), which located on a certain side of the layers comprised by  $s_i$ , moving to the other side of the layers comprised by  $s_i$ , then change the MV assignment of the crease (only one) after (or before) the unfolded crease to keep the global order unchanged. Such



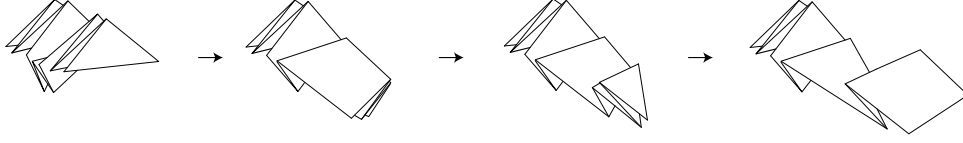


Figure 4.9: The unfolding process of the diagonal creases.

a process is exemplified in Figure 4.9.

The main conclusion is described as Theorem 4.1.2.

**Theorem 4.1.2.** *Let  $M$  be a sheet of paper of size  $1 \times n$  with a box-pleating crease pattern as a subgraph of a same-sized square and diagonal grid, but without an MV assignment. Then, if the crease pattern of  $M$  satisfying local flat-foldability everywhere, the  $M$  can be globally flat-folded, and it takes linear time to find such a folding.*

*Proof.* Our unfolding process ensures the property of non-penetration because the adjustment of MV assignment on the other creases keeps the shape of a zigzag after any unfold. And it is clear that any unfold would not induce penetrations within a single square. Therefore, as long as the local flat-foldability is satisfied, all the input patterns can be globally flat-folded.  $\square$

The folding strategy above gives a way to flat-fold any everywhere locally flat-foldable box-pleating pattern of size  $1 \times n$  without MV assignment. We provided the unfolding process for the simplicity of our proof. In the above description, every unfold step (unfold a crease) requires a traverse of the pattern to determine the changes for other creases, and thus the final flat-folded state is computed in  $O(n^2)$  time. However, the above description is for the purpose of a better understanding. In the algorithm design, the time complexity can be reduced to  $O(n)$  to obtain the overlapping order of layers in the final flat-folded state because we do not really have to change the label of a crease multiple times. The label of a crease can be decided at the time point when all the creases on one side of it are decided. The corresponding algorithm is given as Algorithm 2. In this algorithm, for a practically flat-foldable input, the output is its corresponding overlapping order of the faces in the final flat-folded state, which reveals both the MV assignment we assigned for the input pattern and the way to fold the pattern.

**Input** : A box-pleating pattern  $P$  without an MV assignment  
**Output**: Yes or No, when Yes, also output the overlapping order  $O$  of  $M$  in the final flat-folded state.

**initialization**

$O \leftarrow \emptyset$  // the decided overlapping order at each step

**if** *the MV assignment in the first square is locally flat-foldable* **then**

    Decide the label of creases in  $s_1$  and  $v_1$ , such that the overlapping order follows (the layers of  $s_1$ , the layers of other grids)

    Fold the creases in  $s_1$  and  $v_1$ , append the layers in  $s_1$  to  $O$

**else**

**return** *No*

**end**

**foreach** *vertical  $v_i$  with  $0 < i < n$*  **do**

**if** *the MV assignment in the first square is locally flat-foldable* **then**

        Decide the label of creases in  $s_{i+1}$ , such that the overlapping order globally follows (the layers of  $s_1$ , the layers of  $s_2, \dots$ , the layers of  $s_n$ )

        Fold the creases in  $s_{i+1}$  and append the layers in  $s_{i+1}$  to  $O$

**else**

**return** *No*

**end**

**end**

**return**  $O$

**Algorithm 2:** Linear-time algorithm for deciding the flat-foldability of a box-pleating pattern of size  $1 \times n$  with part of the creases and no MV assignment.

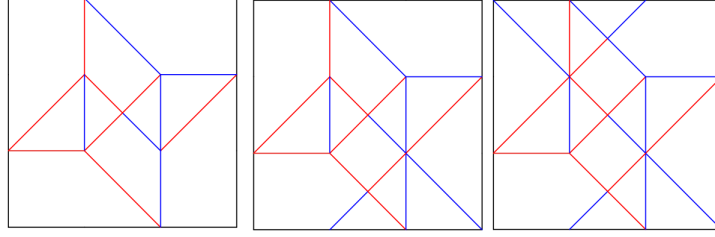


Figure 4.10: Locally flat-foldable but not globally flat-foldable patterns of size  $3 \times 3$ .

## 4.2 Box-pleating Patterns of Size $2 \times n$

As introduced in the last section, the decision problem on the flat-foldability of box-pleating of size  $1 \times n$  without any MV assignment was investigated before the  $2 \times n$  case. The methodology of “fold and unfold” has then been extended to this topic.

Now we only pick out the locally flat-foldable crease patterns, i.e., the crease patterns each can be assigned with at least one reasonable MV assignment such that in the MV assignment, the small neighborhood of the vertex can be folded flat.

In the  $1 \times n$  case, the patterns were described as part of a square and diagonal grid but with no MV assignment. It has been proved that every such locally flat-foldable instance is also globally flat foldable. On the other hand, for patterns of size  $3 \times n$  with exactly the same conditions, even if limited in a small square area, an area of size  $3 \times 3$ , instances that not globally flat-foldable exist, and three representatives are illustrated in Figure 4.10 (exemplified in Ref. [71]). For these patterns, appropriate MV assignments could be imposed without violating the conditions of local flat-foldability, whereas the inability to fold them is due to inevitable self-intersections.

Results show different appearances in  $1 \times n$  and  $3 \times n$  cases, then, what about the  $2 \times n$  cases? Based on both our conclusion for box-pleating patterns of size  $1 \times n$ , and the enumeration of the flat-folded states of box-pleating patterns of size  $2 \times n$  with  $n < 6$  achieved by Matsukawa [71], we proposed the conjecture that in box-pleating patterns of size  $2 \times n$  (See Figure 1.4), the global flat-foldability is also equal to the local flat-foldability at every vertex. To prove this conjecture, we gave the method to fold every input box-pleating pattern to a zigzag shape.

The main theorem is as follows:

**Theorem 4.2.1.** *Let  $P$  be a sheet of paper of size  $2 \times n$  with a box-pleating crease pattern as a subgraph of a same-sized square and diagonal grid but without an MV assignment. Then, any  $P$  satisfying local flat foldability can be flat-folded. Moreover, finding a way to fold the paper takes linear time.*

### 4.2.1 Notations

Our input box-pleating pattern of size  $2 \times n$  is denoted by  $P$ . Similar with the  $1 \times n$  case, here the box-pleating pattern is defined as a subgraph of a square and diagonal grid, i.e., a square grid with also diagonals (illustrated in Figure 1.4). The front and the back of  $P$  are distinguished. The line segments in the crease pattern are called *creases*, and the endpoints of creases inside  $P$  (not on the boundary) are called *vertices*.

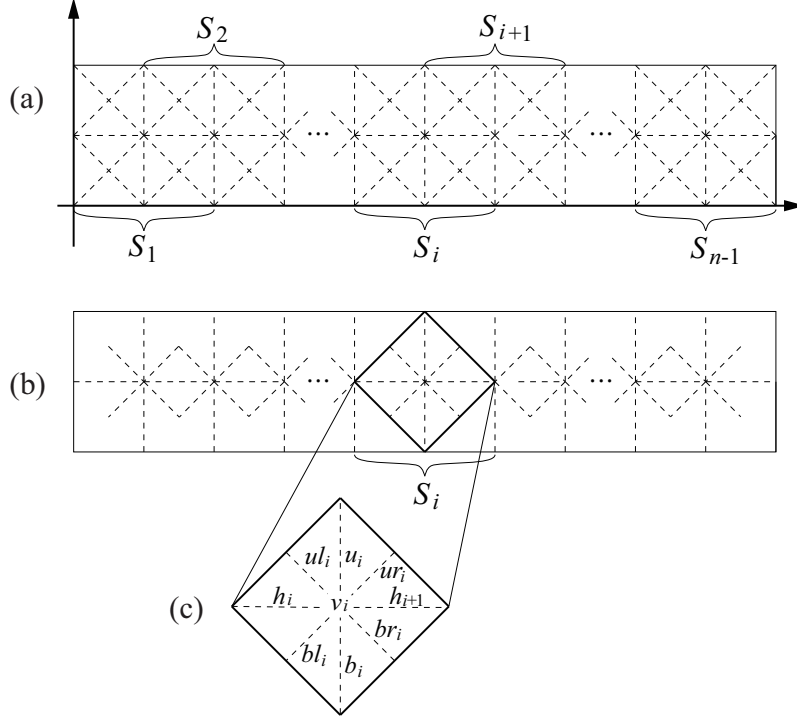









































Figure 4.11: Definitions in a box-pleating pattern of size  $2 \times n$ : (a) Indexing of sections. (b) Creases in  $C$ . (c) Labels of creases incident to  $v_i$ .

For convenience to describe, the input pattern  $P$  is posed in a right-handed Cartesian coordinate, the unit length of which is equal to the edge length of a unit square in  $P$ , and  $P$  is located entirely in the first quadrant with its lower-left corner located at the origin. Along the horizontal axis, a square of  $2 \times 2$  size centered at point  $(i, 1)$  ( $1 \leq i \leq n-1, i \in \mathbb{N}$ ) is defined as the  $i$ th *section*, denoted by  $S_i$  with its center point denoted by  $v_i$  (see Figure 4.11(a)).  $S_i$  and  $S_{i+1}$  form a pair of *neighbor sections*. Note that a pair of neighbor sections shares an area of size  $1 \times 2$ .

As illustrated in Figure 4.11(c), eight possible positions for the creases incident to a  $v_i$  are labeled  $h_i, u_i, b_i, ul_i, bl_i, ur_i$ , and  $br_i$ . Two horizontal positions are labelled  $h_i$  and  $h_{i+1}$ , respectively. For convenience, we also use these labels to describe the creases that really exist. For the vertices which can have eight possible creases around, i.e.,  $v_i$ s, we define the set of them all as  $\overline{V}_8$ . The set of these possible creases is denoted  $C$  and plays a more important role than other creases. Figure 4.11(c) illustrates the creases in  $C$  around a single vertex. Then, the set of remaining vertices with up to four possible creases around (i.e., the vertices located at points  $(i-1/2, 1/2)$  and  $(i-1/2, 3/2)$  for  $i$  with  $1 \leq i \leq n$ ) is denoted as  $\overline{V}_4$ .

In our method, an input box-pleating pattern  $P$  is firstly separated into three kind of parts as *center-line part*, the *no-center-line part* and the *connection section*. The classification is based on the existence of the two horizontal creases  $h_i$  and  $h_{i+1}$  incident to each  $v_i$  in  $\overline{V}_8$ . Precisely, a center-line part is a part of consecutive sections each with both creases  $h_i$  and  $h_{i+1}$ . A no-center-line part is a part of consecutive sections with no horizontal crease. A section joining two such parts is called a *connection section*, which shares its horizontal crease with a center-line part, i.e., with either crease  $h_i$  or  $h_{i+1}$ .

Table 4.1: Thirty-six possible patterns satisfying local flat foldability and associated operations.

| Collection | Center-line part  |   |   |   | Connection section  |   |   |   | No-center-line part  |  |   |   |   |   |
|------------|---|---|---|---|---|---|---|---|--|--|---|---|---|---|
| Pattern    |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|            |   |  |   |  |  |  |  |  |  |  |  |  |  |   |
|            |  |  |  |  |  |  |  |  |  |  |  |  |   |   |
|            |  |   |   |  |  |  |  |  |  |  |   |   |   |   |
| Operation  | Direct folding  |   |   |   |   |   |   |   | Combination folding  |  |   |   |   |   |

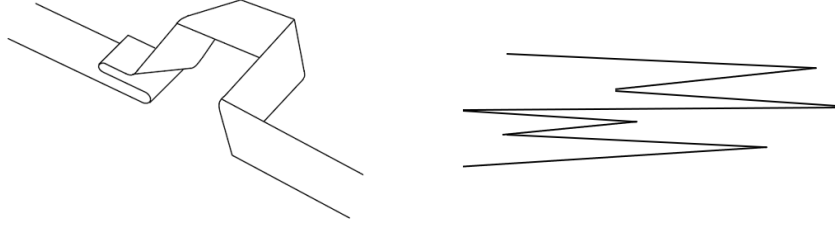


Figure 4.12: A flat-folded state in the shape of a zigzag and a front view.

Since the classification is also for the support of reducing the combinations of sections, we describe it within single sections. Then, 36 possible crease patterns within single sections are classified as illustrated in Table 4.1.

We have mentioned that our method lets every input box-pleating pattern be folded to a zigzag shape. This time we formally name this property as the *overlap principle* and give the description as follows. While folding  $P$  from  $S_1$  to  $S_{n-1}$  in order,  $S_{i+1}$  is always supposed to be folded to layers above  $S_i$  or to the same layer as  $S_i$ .

### Outline of our proof

Corresponding to the separation of three kinds of parts, our purpose, proving that the local flat-foldability is equivalent to the global flat-foldability in box-pleating patterns of size  $2 \times n$  is turned into a three-phases task. In other words, a sufficient condition for flat-foldability is specified as the conjunction of three conditions as (1) no self-intersection appears in any part of  $P$ , (2) the overlap principle is satisfied in every part, and (3) the parts are able to be connected without violating the overlap principle. If these conditions can be simultaneously satisfied, then  $P$  is globally flat-foldable. We showed that such a state definitely exists as long as  $P$  are everywhere locally flat-foldable. In a word, we provided a general method to fold  $P$  into the shape of a zigzag (see Figure 4.12).

For the classified 36 possible locally flat-foldable patterns in a single section, we provide two corresponding folding methods: *direct folding* (without unfolding) and *combination folding* (a combination of folding and unfolding operations). The summary in Table 4.1 concerns certain methods of folding corresponding to all these patterns. In the

following sections, we will explain how to specify the MV assignments for every center-line part and no-center-line part, respectively, in order to flat-fold these parts under the overlap principle. Then, we will show that every connection section can be effectively flat-folded to join a center-line part and a no-center-line part. The entire MV assignment is given step-by-step for every pattern in a section and their combinations.

In conclusion,  $P$  can be flat-folded if and only if its vertices are all locally flat-foldable. The decision on whether  $P$  can be flat-folded is solvable in linear time by checking the vertices one by one locally. For a positive input  $P$ , one of its flat-folded states is also specified in our folding process in the same linear-time.

## 4.2.2 Flat-Foldability in every part

In this section, the flat-foldability of all the possible cases in each part will be explained with different methods. By all the possible cases, we mean all the combinations of the 36 locally flat-foldable patterns illustrated in Table 4.1. Creases not incident to  $v_i$ s, i.e., some creases around the vertex in  $V_4$ , are left out in the illustrations of patterns in Table 4.1. Thus, aside from these explanations, there remains a pre-explanation on the reasonableness for why we can consider only the creases in  $C$ . The formal presentation is Lemma 4.2.2.

**Lemma 4.2.2.** *The flat foldability of  $P$  can be decided only by the creases in set  $C$ .*

*Proof.* Following the notations introduced in the last section, two neighbor sections  $S_i$  and  $S_{i+1}$  share an area of size  $1 \times 2$ . Such areas form a cover of the whole pattern except for its two ends,  $S_1$  and  $S_{n-1}$ . Thus, almost all the creases not in  $C$  are included in these areas and discussed within such areas. We thus consider half of the area, a single square in the upper row. Possible numbers of the creases within the square are zero, two (on one diagonal), and four (on two diagonal) creases cases. The zero creases case is trivial. For the two creases-case, since the local flat foldability forces the creases to be on the same line and with the same assignment, the entire assignment in the square is actually decided by  $C$ . For the case of four creases, two of the four creases are in  $C$ . Next, we will explain that all the assignments would be automatically decided once the assignments of the two creases in  $C$  is given.

Eight locally flat-foldable MV assignments are in existence within a single square. The symmetry between  $M$  and  $V$  induces a reduction of these assignments by half. The cases where  $ur_i$ s are assigned  $M$ s, as in Figure 4.13, are picked out and discussed. Four triangles separated by creases are identified by labels  $T_r$ ,  $T_u$ ,  $T_l$ , and  $T_b$ . Numbers written in the triangular areas indicate the overlapping orders of them in the folded states, with the assumption that  $T_b$  faces the front up. The overlap principle requires  $T_r$  overlapping (above)  $T_l$  finally. In Figure 4.13 1-(a) and 1-(b), we give the two possible assignments for the case that  $ur_i$  and  $ul_{i+1}$  are assigned  $M$  and  $V$ , respectively. While in 2-(a) and 2-(b), we provide the two possible assignments for the case that  $ur_i$  and  $ul_{i+1}$  are both assigned  $M$ . 2-(a) satisfies the requirement that  $T_r$  overlap  $T_l$  and is chosen in our method. When  $ur_i$  and  $ul_{i+1}$  are both assigned  $V$ , the assignment of the upper two creases in 2-(b) (the creases not in  $C$ ) will be chosen.

For the other half, the lower squares in the areas of size  $1 \times 2$ , creases in the bottom row can be handled in exactly the same manner. Finally, the remaining creases at the

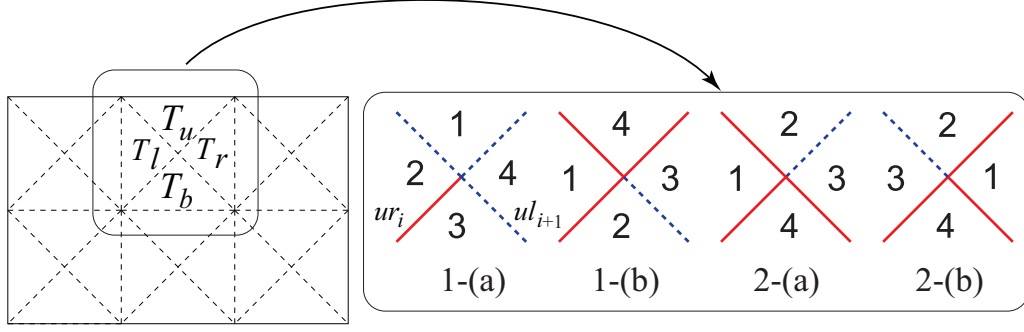


Figure 4.13: Two possible cases for creases around a vertex in  $\overline{V}_4$ .

two ends of size  $1 \times 2$  are basically handled in the same way. One of 1-(a), 1-(b), and 2-(a) would be the solution and is decided by the assignment of the crease in  $C$ .

Following the assignments chosen according to the above discussion, folding the creases which are not in  $C$  would not arise self-penetration because (1) such folds in a certain section do not cause intersection with other sections since the section is folded to different layers from other sections, and (2) the above discussion shows that such folds do not cause self-intersections within a single section. Furthermore, the overlap principle is always respected in our choices. Lemma 4.2.2 is thus proven.  $\square$

The Following explains the flat-foldability of different kinds of parts by presenting the flat-folded state of these parts, respectively.

### Flat-foldability of the center-line part

The folding operation for the center-line part is specified to be direct folding, which means that unfolds are not necessary in this case. A center-line part is a part comprised of continuous sections  $S_i$ s, each with both creases  $h_i$  and  $h_{i+1}$ . There are thus twelve possibilities for the pattern of each  $S_i$ , according to the classification in Table 4.1. The twelve possibilities induce 144 possible combinations of  $S_i$  and  $S_{i+1}$ . In other words, within a small region of size  $2 \times n$ , there are 144 possibilities of the crease patterns, and we must assign a reasonable MV assignment for each one.

The idea of using simply direct folding rather than more comprehensive operations comes from the following thoughts. First, if the patterns comprising a center-line part are all symmetric with respect to the horizontal center-line, then a reasonable flat-folded state of the part can be achieved by initially folding the center-line entirely as either M or V to reduce it to a flat-foldable pattern of size  $1 \times n$  (since the creases in the upper half and the creases in the lower half would totally overlap each other) which can be handled with the method introduced in the last section. Second, if the center-line is pre-folded, asymmetric patterns can be adjusted by only a little to get to the reasonable flat-folded states.

Our simplification of the center-line part is thus decided to be first folding the entire center-line as V, and then folding the patterns following the MV assignments illustrated in Table 4.2. The corresponding overlapping orders of the flat-folded states are also provided, certifying that our folding follows the overlap principle. There are two kinds of assignments, differentiated by either that the area directly under the left horizontal

Table 4.2: MV assignments and the overlapping order in every folded state. The numbers in ascending order indicate the corresponding layers from bottom to top.

| Case                                      | 1       | 2       | 3       | 4       | 5       | 6       | 7       | 8       |         |         |         |         |
|---|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| MV assignment<br>and<br>overlapping order | 1-1<br> | 1-2<br> | 1-5<br> | 1-3<br> | 1-4<br> | 1-6<br> | 2-1<br> | 2-5<br> | 2-2<br> | 2-3<br> | 2-6<br> | 2-4<br> |
|   |         |         |         |         |         |         |         |         |         |         |         |         |

crease faces the front up(the folding follows the upper illustrations) or the back of this area faces up(the folding follows the lower illustrations).

To summarize our conclusion for the center-line part, we have the following lemma.

**Lemma 4.2.3.** *Any center-line part in  $P$  can be flat-folded respecting the overlap principle providing that it is everywhere locally flat-foldable.*

*Proof.* The flat-folded state of a center-line part can be achieved by applying the above operations to the sections one by one from left to right. Specifically, each section should be folded as illustrated in the upper row in Table 4.2 when the area directly under the left horizontal crease faces the front up. Otherwise, each section should be folded to the states whose overlapping orders are shown in the lower row.

The overlapping orders in the folded states, as labeled in Table 4.2, show that for every section  $S_i$  (except for the sections in Pattern 1-1 ), its left half, which also belongs to  $S_{i-1}$  would be folded under its right half, which also belongs to  $S_{i+1}$ . Particularly, every  $S_i$  in Pattern 1-1 would put its right half in the same layer as its left half, which also obeys the overlap principle.  $\square$

In fact, the reasonableness of our assignment should be verified with also the consideration of both the ends of size  $1 \times 2$  of a center-line part. Both ends are connected with the no-center-line parts by connection sections. This part of the discussion will be finished in the following two sections.



### Flat-foldability of the no-center-line part

On the one hand, the non-existence of the center-line diminishes the hope for directly reduce a no-center-line part to a box-pleating pattern of size  $1 \times n$ . While on the other hand, it causes the similarity to the box-pleating patterns of size  $1 \times n$  since both of them have no horizontal crease. Therefore, this time we use a folding-unfolding process similar to the process we provided for the box-pleating pattern of size  $1 \times n$ , called the combination folding, to handle such parts.


Compared with the  $1 \times n$  case, there are one time more possible positions for the vertical creases, which naturally arises the complexity. As a simple pre-analysis, there are eight possible patterns involved in the no-center-line part in total. Thus, the combination should involve  $8 \times 8 = 64$  possible cases. In our method, we handle Pattern 4-1 and Pattern 4-2 by directing folding. The other six patterns are handled by the









combination folding. We classify the combinations of other six patterns into ten cases to reduce the cumbrousness.

Before introducing the way to folding-unfolding such patterns, we first remove two particular patterns, Pattern 4-1  and Pattern 4-2 , from this discussion. The reason is that it is not very easy to unfold them but is more natural to view them as separation sections of the entire input pattern. More precisely, both Pattern 4-1 and Pattern 4-2 can separate  $P$  into two individual parts without any interference with each other, which means that these individual parts can be handled as independent box-pleating patterns of size  $2 \times n$  (a rough speaking since they can be separated by creases not vertical and thus their ends can be irregular).

Next, for the other possible patterns belonging to a no-center-line part, we use combination folding instead of direct folding. Surely, corresponding to a certain no-center-line part, there may exist many ways to flat fold it. We use combination folding only for the convenience of handling general cases. In the following explanation, we denote a no-center-line part as  $\{S_i \mid a \leq i \leq b\}$ . For convenience, both  $a$  and  $b$  are assumed to be odd.

The same as the box-pleating patterns of size  $1 \times n$ , an initial state is prepared before all the unfolds. Instead of hypothesizing the existence of all the creases, we initially fold each section as Pattern 5-2 , and then modifies creases to the real input pattern. The modification includes folds, unfolds, and some changes on the MV assignment. Even though every  $S_i$  is initially folded as Pattern 5-2, the  $MV$  assignments are assigned alternately with respect to  $i$  to achieve the shape of zigzag. The concrete way of folding is illustrated in Figure 4.14, whose upper illustration shows the flat-folded state, and the lower illustration shows the three phases of the folding process. Each section belonging to  $\{S_i \mid a \leq i \leq b, i \text{ is odd}\}$  has its creases incident to  $v_i$  assigned three valleys and a mountain, while each section belonging to  $\{S_j \mid a \leq j \leq b, j \text{ is even}\}$  has its creases incident to  $v_j$  assigned three mountains and a valley in a no-center-part composed by  $\{S_i \mid a \leq i \leq b\}$ . Then, the unfolds, and adjustments are also supposed to be applied during two phases, corresponding to the different  $MV$  assignments. In the first phase, sections in  $\{S_i \mid a \leq i \leq b, i \text{ is odd}\}$  are folded to their actual patterns. In the second phase, the remaining sections in  $\{S_i \mid a \leq i \leq b, i \text{ is even}\}$  are adjusted. Note that the first phase is done under the condition that all the  $\{S_i \mid a \leq i \leq b, i \text{ is even}\}$  are pre-folded as Pattern 5-2, while in the second phase, sections in  $\{S_i \mid a \leq i \leq b, i \text{ is odd}\}$  are already folded to their actual patterns.

We list the adjustments for all the other possible patterns in the first phase as follows. Figure 4.15 indicates almost every adjustment from Pattern 5-2 , except the patterns with properties very similar to the listed ones. The  $MV$  assignments and the corresponding states are illustrated side by side in this illustration. The simplest one, a section in Pattern 5-4  only needs a single unfold within the upper half area. Then, a section in Pattern 5-3  and a section in Pattern 5-6  need unfolds along the vertical creases followed by folds along diagonal creases. Pattern 5-1  requires an additional change on the  $MV$  assignment of the sections right to it because unfolds along the vertical creases of this pattern would change the overlapping order while we desire to obey the overlap principle. The adjustment to sections on its right side can be realized by an alternation of the  $MV$  assignments between sections with odd and even indexes. The adjustment to sections in Pattern 5-5  is omitted in the illustration since such sections can be

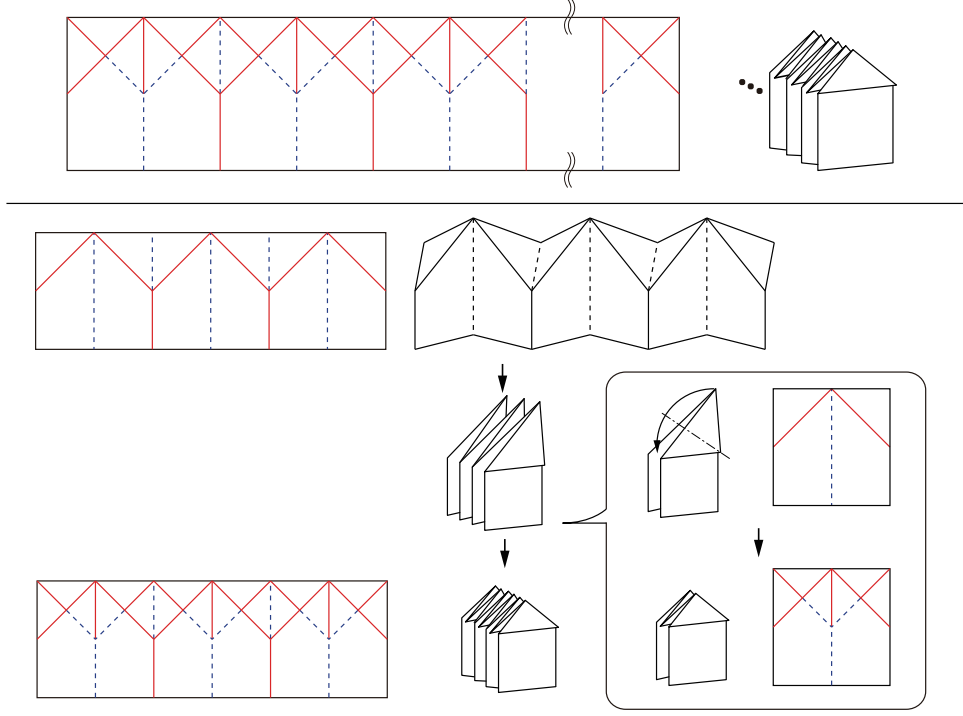


Figure 4.14: The no-center-line parts and the method to fold them. The upper illustration shows an initial flat-folded state of a no-center-line part and its  $MV$  assignment, and the lower illustration shows how to fold the part.

folded with firstly the same operation prepared for sections in Pattern 5-4  $\square$  and a next succeeding fold along its two diagonal creases, which is the same with the operations for sections in Pattern 5-2  $\nabla$ . It is rather clear that the above-mentioned combination folding operations would not affect the global overlapping order, which means they would not violate the overlap principle. The reasonableness of these operations in the first adjusting phase is presented in Lemma 4.2.4.

**Lemma 4.2.4.** *It is realizable to form the flat-folded states of sections in Patterns 5-1  $\square$ , 5-3  $\nabla$ , 5-4  $\square$ , 5-5  $\nabla$ , and 5-6  $\square$  from the flat-folded state of Pattern 5-2  $\nabla$  while maintaining the overlap principle in the first phase of combination folding (the handling for sections in  $\{S_i \mid a \leq i \leq b, i \text{ is even}\}$ ).*

Its correctness follows the above discussions.

In the second phase, every section in  $\{S_i \mid a \leq i \leq b, i \text{ is even}\}$  is adjusted in the case that its two neighbor sections are already adjusted to the right patterns. Since  $S_{i-1}$ ,  $S_i$ , and  $S_{i+1}$  form a continuous area of size  $2 \times 4$ , it is sufficient to consider all possible combinations of the six patterns in such areas of size  $2 \times 4$ . In the current flat-folded state,  $S_{i-1}$  and  $S_{i+1}$  have been already adjusted while  $S_i$  is still folded as a section in Pattern 5-2  $\nabla$ . The shape of the current flat-folded state is in a zigzag. With the same combination folding operations, the folded states  $S_i$ s are supposed to be modified from Pattern 5-2  $\nabla$  while keeping the overlap principle respected. Our strategy is to first adjust  $S_i$  to a tractable state according to  $S_{i-1}$  and  $S_{i+1}$  and then apply other folds to the pattern exactly the same with the input.

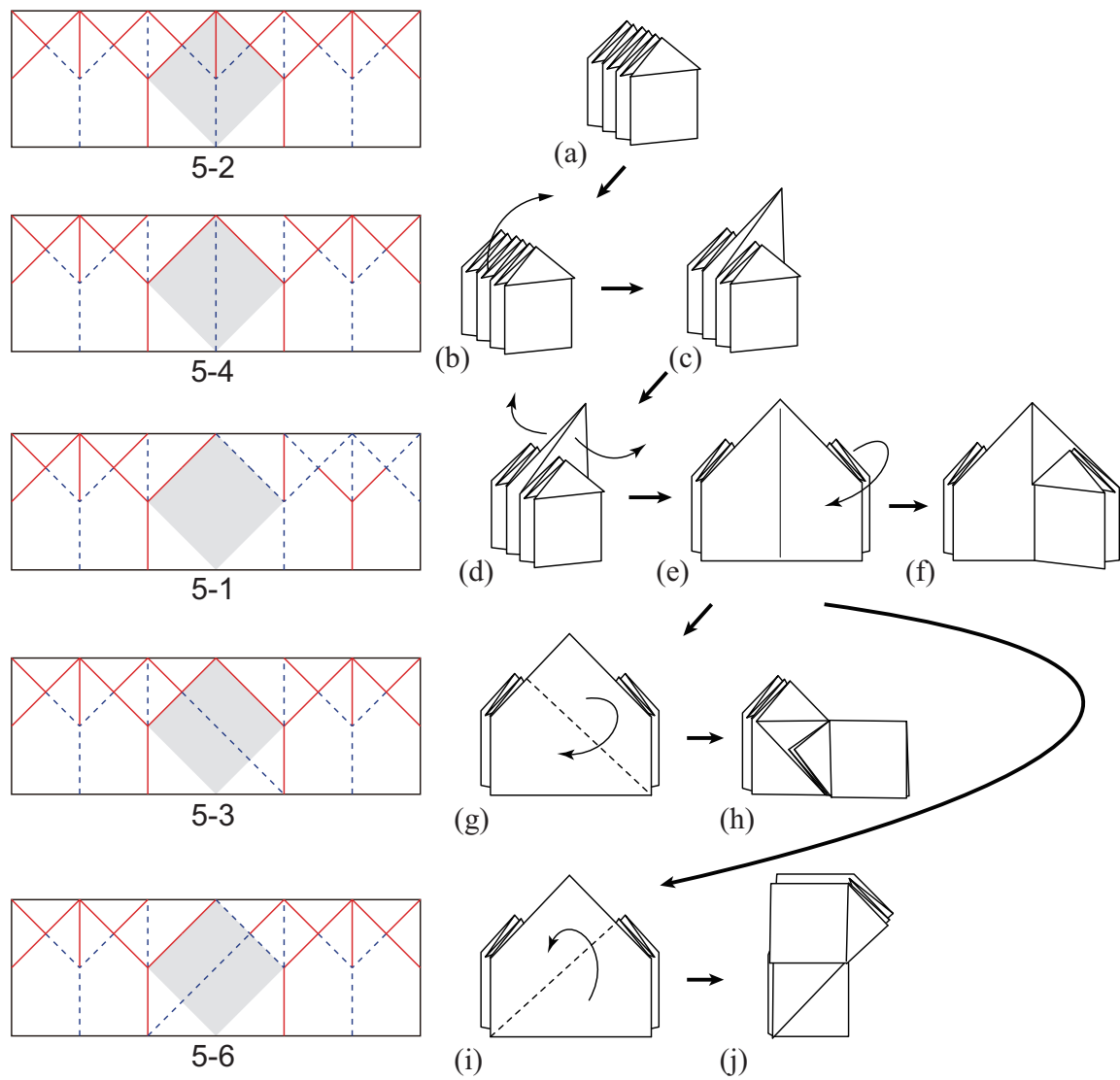


Figure 4.15: Operations of combination foldings.

Table 4.3: Ten classes of combinations involved in the second phase.

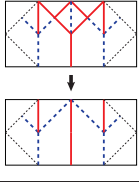
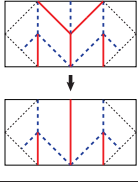
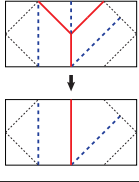
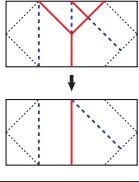
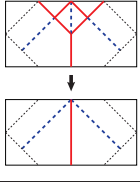
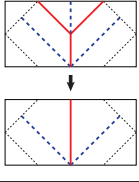
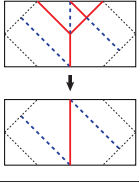
| Case | Combinations of patterns ( $S_{i-1}, S_{i+1}$ )   |
|------|---|
| 1    | $(\{5-1\Box, 5-4\Box\}, \{5-1\Box, 5-4\Box\})$  |
| 2    | $(5-2\Box, 5-2\Box)$  |
| 3    | $(5-2\Box, \{5-1\Box, 5-3\Box, 5-4\Box, 5-5\Box, 5-6\Box\}),$<br>$(\{5-1\Box, 5-3\Box, 5-4\Box, 5-5\Box, 5-6\Box\}, 5-2\Box)$ |
| 4    | $(5-5\Box, 5-5\Box)$  |
| 5    | $(5-5\Box, \{5-1\Box, 5-3\Box, 5-4\Box, 5-6\Box\}),$<br>$(\{5-1\Box, 5-3\Box, 5-4\Box, 5-6\Box\}, 5-5\Box)$                   |
| 6    | $(\{5-1\Box, 5-4\Box\}, 5-6\Box), (5-6\Box, \{5-1\Box, 5-4\Box\})$  |
| 7    | $(\{5-1\Box, 5-4\Box\}, 5-3\Box), (5-3\Box, \{5-1\Box, 5-4\Box\})$  |
| 8    | $(5-6\Box, 5-3\Box)$  |
| 9    | $(5-3\Box, 5-6\Box)$  |
| 10   | $(5-3\Box, 5-3\Box), (5-6\Box, 5-6\Box)$  |

For the considered regions of size  $2 \times 4$ , all the possible combinations in a no-center-line part within these regions are firstly categorized. Precisely, the six possible patterns 5-1 to 5-6 lead to 36 ( $6 \times 6$ ) possible combinations. These combinations are classified into ten classes for their similarities and difference during the folding. The classification is presented in Table 4.3. Combinations belonging to the same case are supposed to be handled with almost the same folds and unfolds. To explain our notations,  $()$  indicates the combination, and  $\{\}$  indicates the set comprised of choosable elements. For example, in Case 1, four kinds of combinations as (5-1, 5-4), (5-1, 5-1), (5-4, 5-1), and (5-4, 5-4) are included; In Case 2, only the combination of (5-2, 5-2) is categorized; In Case 3, all the combinations of Pattern 5-2 and any other patterns, no matter in which order, are included. Explanations of other cases follow the same way.

Then, as before-mentioned, according to the folded state of  $S_{i-1}$  and  $S_{i+1}$ , adjustments of  $S_i$  as a prepared state for the next folds and unfolds are illustrated by the  $MV$  assignment given on the upper side in Table 4.4. The  $MV$  assignment on the lower side in every case explains the next folds and unfolds. The folded states follow the overlap principle. The method of folding provided here is, in fact, an adjustment from the folded state obtained at the first phase to a flat state based on the real pattern under consideration of the overlap principle and represented by the  $MV$  assignments. These illustrated assignments correspond to the case in which every leftmost area faces the front up. During a real folding process, once the leftmost area turns the direction to face the back side up, the assignments should be reversed to get the correct handlings.

The handlings correspond to different cases are explained as follows. For most cases, the cases other than Cases 1, 3, and 5, the handlings include the first adjustment on  $S_i$  to the corresponding assignment (illustrated in Table 4.4), and then apply the same folds and unfolds as in the first phase. Then, for the particular Cases 1, 3, and 5, the handlings are respectively introduced below.

Table 4.4: Ten classes of combinations involved in the second phase of unfolding the no-center-line part.

| Case   | 2   | 4   | 6   | 7   | 8  | 9   | 10  |
|--|---|---|---|---|--|---|---|
| Patterns with MV assignment before and after the operation |  |  |  |  |  |  |  |

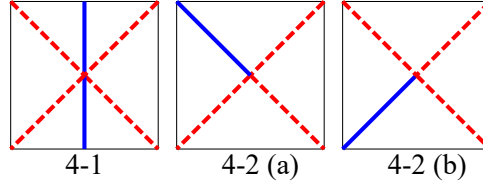




Figure 4.16:  $MV$  assignments for Patterns 4-1  and 4-2 .




In Case 1, the handling is performing a trivial unfolding operation on  $S_i$  as in the first phase. This operation is always feasible because the unfold involves no part intersecting the crease near  $v_i$ .

Case 3: The operations are almost the same as Case 2. The only difference is that when there is no corresponding crease in another part, the diagonal crease in  $S_i$  will also be deleted.

Case 5: The operations are almost the same as Case 4. Similar to Case 3, the only difference is that when sometimes unnecessary diagonal creases in  $S_i$  exist, they would be eliminated.

Until now, all the cases and their corresponding handlings are listed and explained. By the above case analysis, we can assert Lemma 4.2.5. Its proof follows the presentations above.

**Lemma 4.2.5.** *All possible combinations of patterns from Patterns 5 – 1 to 5 – 6 in a no-center-line part can be flat-folded.*



To conclude the proof for the flat-foldability of a no-center-line part, we have to finally include Patterns 4-1  and 4-2  into our discussion. Both would be handled by the direct-folding operation. Under the restriction of the overlap principle, in Figure 4.16, we give the  $MV$  assignments in the case that the leftmost triangle faces the front up. Sections in Patterns 4-1 and 4-2 are supposed to be folded before other sections. The reason is that after folding such a section,  $P$  can be respectively regarded as two independent parts joined by vertical creases or two diagonal creases assigned the same and along the same line. Then, the independent parts can be viewed as a new box-pleating pattern and be handled in the same way as the initial input pattern.  $MV$  assignment for the other case, where the leftmost triangle faces the back side up, can be obtained by interchanging the roles of  $M$ s and  $V$ s. In both cases, two choices are feasible for Pattern 4-2 , with the difference on which line directs the separation.

By now, all the handlings of the possible patterns involved in a no-center-line part are detailed. Therefore, as the final conclusion for the no-center-line part, we claim Lemma 4.2.6.


**Lemma 4.2.6.** *When independently consider a no-center-line part of  $P$ , its global flat-foldability can be ensured by its local flat-foldability.*

### Flat-foldability of the connection section

In the above explanations, we have discussed the equivalence between the local flat-foldability and the global flat-foldability in center-lined parts and no-center-line parts. In a certain input pattern, these parts are disjoint with each other, and their union composes the entire pattern. The remaining task is to prove that when these parts are connected together, no penetration would be produced, and the shape of the zigzag would be maintained. Now, our focus is the connection section, the section that connects a center-lined part and a no-center-line part. Note that the MV assignment (the way of folding) of the connect section has been decided, half as a center-lined part, and the other half as a no-center-line part.

The proof is partitioned into two parts to exhaust all the possible combinations involving a connection section within an area of size  $2 \times 3$ . In the above explanations, we have already introduced that both Patterns 4-1  and 4-2  can be seen as separations and thus their connectability with other sections are clear and is omitted in the following discussion. In the first part, the discussed area (of size  $2 \times 3$ ) is comprised of a connection section and a section at the end of a no-center-line part. In the second part, the discussed area is comprised of a connection section and a section at the end of a center-lined part. As a combinatorial analysis, regardless of the MV assignment, the number of possible patterns within these areas are 96 ( $16 \times 6$ ) and 192 ( $16 \times 12$ ), respectively.

Our method reduces them to eight and 20 by choosing representatives from very similar cases. The corresponding folding methods are described by the MV assignments and the overlapping orders in the final flat-folded states, as illustrated in Tables 4.5 and 4.6. These representatives are chosen with respect to symmetry and the initial states defined for the no-center-line part.

For the combination of a connection section and a no-center-line part, it is sufficient to consider only the combinations involving Pattern 5-2  because every flat-folded state we defined for the other patterns in a no-center-line part (Patterns 5-1, 5-3, 5-4, 5-5, and 5-6) can be obtained from Pattern 5-2 while maintaining the overlap principle (Lemma 4.2.5). Based on this analysis, we give the MV assignments and the flat-folded states for the possible combinations involving Pattern 5-2 at the right side, i.e., the combinations in the pattern (a connection section, a section in Pattern 5-2), in Table 4.5. The reverse case that the combinations are in the pattern (a section in Pattern 5-2, a connection section) can be seen as turning the back side of the paper over and can be simply dealt with by interchanging all the Ms with Vs. Two different assignments for Pattern 5-2 in these illustrations distinctively decide the initial folding of the connected no-center-lined part (the parity of sections). The conclusion is presented as Lemma 4.2.7.

**Lemma 4.2.7.** *Any two neighbor sections  $S_i$  and  $S_{i+1}$  belonging to a connection section and a no-center-line part, respectively, can always be folded.*

Table 4.5:  $MV$  assignments and overlapping orders in every folded state for the connections of a connection section and Pattern 5-2.





| Case                               | 1  | 2  | 3  | 4  |
|------------------------------------|--|--|--|--|
| MV assignment and overlap relation | <br>$3-1 \rightarrow 5-2$<br><br>$3-2 \rightarrow 5-2$ | <br>$3-2 \rightarrow 5-2$<br><br>$3-6 \rightarrow 5-2$ | <br>$3-3 \rightarrow 5-2$<br>or<br><br>$3-3 \rightarrow 5-2$<br><br>$3-7 \rightarrow 5-2$<br>or<br><br>$3-7 \rightarrow 5-2$ | <br>$3-4 \rightarrow 5-2$<br><br>$3-8 \rightarrow 5-2$ |



For the other kind of combination, the combination of a connection section and a center-lined part, all the possible cases are classified into 20 cases. Representatives of these cases are illustrated in Table 4.6. The connection section is on the right side, and the section from the center-lined part is on the left side. The same as the combination of a connection section and a no-center-line part, the reverse case of this kind of combination, that the combinations are in the pattern (a connection section, a section from the center-lined part) can be dealt with by interchanging all the Ms with Vs. In these illustrations, the lower-left corners of the combined areas of size  $2 \times 3$  are assumed to face the front up.

The meaning of the columns in Table 4.6 should be recognized as follows. Patterns in the same grid have similar properties and thus are supposed to be folded in the similar manner. For example, in the first kind of combination in Case 1, when combining either Pattern 1-1 or Pattern 2-5 with any one of Patterns 3-1, 3-2, 3-5, 3-6, 3-9, 3-10, 3-13, and 3-14, they can always be folded in a manner very similar with the manner we provided on the first row of the third column, the one for the combination (Pattern 1-1, Pattern 3-1).



For every class in the first column, i.e., the classes of sections from the center-lined part, only the folding manner of the representative pattern is illustrated in the third column. For all the other unlisted pattern, the flat-folded states of their relevant combinations can be obtained by folding these patterns in the above-mentioned manners illustrated in Table 4.2.

Following, we will give the explanations for the unlisted patterns from the connection section.

For Case 1 in Table 4.6, we have: all the combinations involving Pattern 3-2  should have the creases  $h_i, ul_i, u_i, br_i$  in Pattern 3-2 assigned as  $h_i, ul_i, ur_i, b_i$  in Pattern 3-1  (the case that the lower-left corner faces the back up can be dealt by changing the assignments of  $b_i$  and  $ur_i$ ); All the combinations involving Patterns 3-5  or 3-6  should have their assignments are specified as the mirrors of Patterns 3-1 and 3-2 with respect to the centerline.

For Case 2 in Table 4.6, we have: all the combinations involving Pattern 3-7  should be assigned by mirroring the assignments for Pattern 3-3  with respect to the centerline (the case that the lower-left corner faces the back up can be dealt by changing

the assignments of  $u_i, ul_i, ur_i$ , and  $br_i$ ).

For Case 3, the  $MV$  assignment of Pattern 3-4  illustrated in the table can be easily connected to any pattern from a center-line part assigned as in Table 4.2. The reason is that there exists no diagonal crease in the left half of Pattern 3-4, which would not cause any intersection with its left neighbor section; The combinations involving Pattern 3-8  can be handled by mirroring the assignment of Pattern 3-4 with respect to the centerline.

In the illustrations, the right half of a connection section is supposed to be shared by a no-center-line part. Different assignments are presented for Pattern 5-2, which distinctively decide the initial folding of the connected no-center-lined part. By here, we have exhausted all the cases. Correspondingly, we give the conclusions as Lemma 4.2.8 and Lemma 4.2.9.

**Lemma 4.2.8.** *Any two neighbor sections  $S_i$  and  $S_{i+1}$  belonging to a center-line part and a connection section, respectively, can always be globally flat-folded to the shape of zigzag.*

**Lemma 4.2.9.** *A connection section can always be globally flat-folded as the connection of a no-center-line part and a center-line part.*

In the above analysis respectively done in (1), (2), and (3), we have obtained the conclusions that as long as the local flat-foldability is satisfied, all the no-center-line parts and center-line parts can be flat-folded and be connected by the connection section, at the same time respecting the overlap principle. Therefore, any box-pleating pattern of size  $2 \times n$  can be globally flat-folded, providing that it is locally flat-foldable.

### 4.2.3 Linear-time algorithm for the decision problem

As a direct conclusion of the last section, the decision on the flat-foldability of any input box-pleating pattern of size  $2 \times n$  can definitely be given in time linear in  $n$  because of the equivalence between the local flat-foldability and global flat-foldability. On this level, an existing algorithm given in Ref. [14] deciding the local flat-foldability can also be viewed as a solution to our decision problem. However, we provide an algorithm that takes one step further. Rather than only the answer to the decision problem, corresponding to any flat-foldable input pattern, our algorithm also provides a concrete manner to fold it. The algorithm description is given in Algorithm 3. For the integrity of the input data, we use a sequence of sections with the information of creases rather than a raw pattern.

There remain only a few details to explain. First, to avoid repetition when recording the horizontal creases, only the last section,  $S_n$ , has eight parameters, while every other section has seven. As a standard, all the shared horizontal creases are assigned to the right-side sections. These input parameters indicate the existence and assignments of creases in each section. We denote the state of a crease, non-existence, existence with no assignment, the assignment  $M$ , and the assignment  $V$ , by an integer in  $\{0, 1, 2, 3\}$ , respectively. When the output is “False”, it describes that the input box-pleating pattern is neither globally flat-foldable or locally flat-foldable. Otherwise, the output would be  $F$ , which indicates a sequence of folding operations leading the input pattern to a flat-folded state in a zigzag shape. Finally, about the computational complexity, in our computation, it requires a constant time to find the corresponding operation defined by



**Input** : The section sequence  $\{S_i\}$  s.t.,  $S_i = (h_i, u_i, b_i, ul_i, bl_i, ur_i, br_i)$  with  $1 \leq i \leq n-2$ ,  $S_{n-1} = (h_{n-1}, u_{n-1}, b_{n-1}, ul_{n-1}, bl_{n-1}, ur_{n-1}, br_{n-1}, h_n)$

**Output:** A sequence of operations  $F$  for every section along the horizontal axis

**initialization**

$e \leftarrow (0, 0, 0, 0, 0, 0, 0)$  // the present pattern

$P \leftarrow \emptyset$  // the sequence of parts

$F \leftarrow \emptyset$  // the sequence of operations

$i \leftarrow 1$

$n \leftarrow |\{S_i\}|$

**while**  $i \leq n-2$  **do**

    update  $e$  as the parameters for  $S_i$

**if**  $e$  represents either Pattern 4-1 or Pattern 4-2 **then**

        divide the present part into two parts with respect to  $e$ , and recall the operations involved in this while loop for both parts

**else**

        decide the part to which part  $S_i$  belongs, append this part to  $P$

**end**

$i \leftarrow i+1$

**end**

append  $S_{n-1}$  to  $P$

**foreach**  $p$  in  $P$  **do**

**if**  $p$  is locally flat-foldable **then**

**if**  $p$  is a center-line part **then**

            append the corresponding direct folding operation to  $F$

**end**

**if**  $p$  is a no-center-line part **then**

            append the corresponding direct folding operations for Patterns 4-1 and 4-2 and the corresponding combination folding operation for other patterns to  $F$

**end**

**if**  $p$  is a connection section **then**

            append the corresponding operation of the connection to  $F$

**end**

**else**

**return** *False*

**end**

**end**

**return**  $F$

**Algorithm 3:** Linear-time algorithm for computing the folding process of a given  $2 \times n$  map.

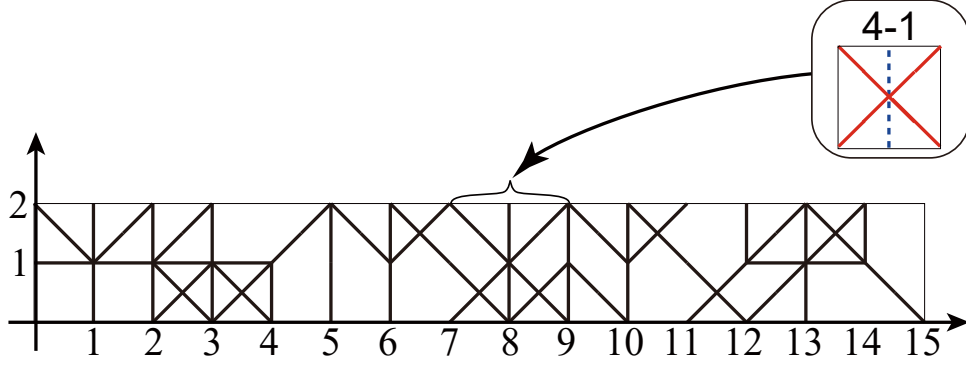


Figure 4.17: Example of size  $2 \times 15$ .

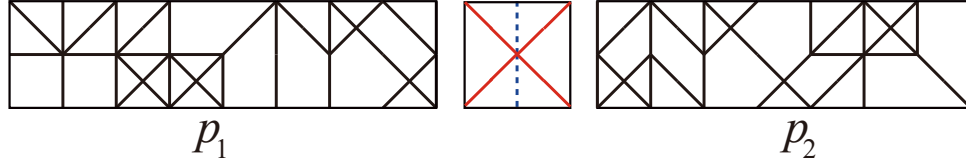



Figure 4.18: Two parts separated by  $S_8$  in Pattern 4-1.




the MV assignments and the overlapping orders for each section, and there are  $n-1(O(n))$  sections. Therefore, Algorithm 3 costs time linear in  $n$  to return either a folding sequence or a negative answer to the decision problem. The main conclusion of this work, Theorem 4.1.1, is proven.

#### 4.2.4 An instance of size $2 \times 15$

Even though we have proven the locally flat-foldable parts in different kinds can be all globally flat-folded and connected to each other, in the above explanations, we mentioned many details so as to complete the proof. But as an instruction about the method of folding a certain locally flat-foldable pattern, it may be difficult to follow. To make it clearer, in this part, we exemplify the concrete folding method of a certain input pattern of size  $2 \times 15$ , as illustrated in Figure 4.17.

Via the first traverse of the input pattern which is represented by sections, one would find the separation point  $S_8$  in Pattern 4-1 . Thus the input pattern is at the first step separated into two parts by  $S_8$ , whose MV assignment is illustrated in Figure 4.17. The separated two parts,  $p_1$  and  $p_2$  as shown in Figure 4.18, are viewed as new inputs, and thus, the same handlings should be recalled for both of them.

Based on the existence of the centerline,  $p_1$  is separated into three parts, namely, one center-line part, one no-center-line part in left-to-right order, and the connection section between them, as shown in Figure 4.19.

The center-line part of  $p_1$  is composed by sections in Pattern 1-5 , Pattern 1-4 , and Pattern 1-2 . The section in Pattern 1-5 is assigned the same as the assignment in the upper row of Table 4.2. For sections in Patterns 1-4 and 1-2, their MV assignments are given according to the second row because their back would face upward in the final zigzag shape. These MV assignments complete the folding manner of the center-line part

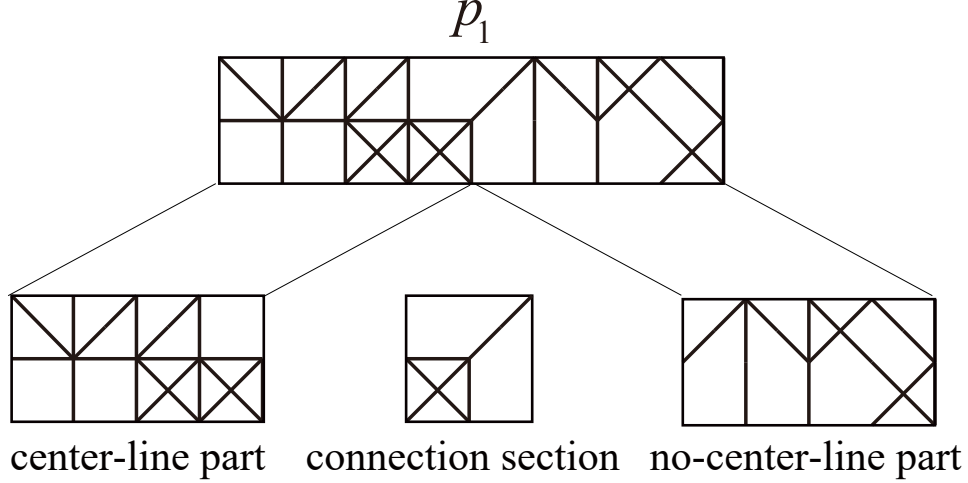


Figure 4.19: Three parts in  $p_1$  with respect to the existence of the centerline.

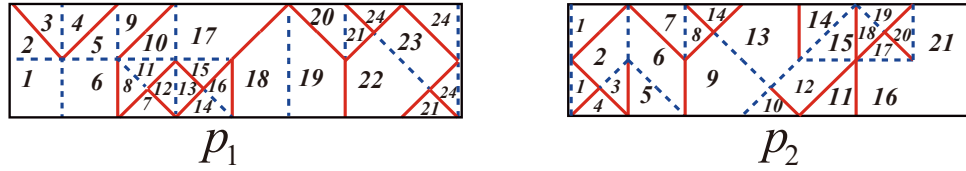


Figure 4.20: The final assignment and the overlapping order of the flat-folded state of  $p_1$ .

of  $p_1$ . The final assignment of the center-line part of  $p_1$  is shown on the left-hand side of Figure 4.20. The numbers indicate overlapping orders.

The connection section of  $p_1$  matches Case 1 in Table 6. Therefore, it is folded in a similar manner with the combination (Pattern 1-5 ( $\square$ ), Pattern 3-1 ( $\square$ )). The difference is, we have to firstly change Pattern 3-1 to Pattern 3-2 with its creases  $h_i, ul_i, u_i, br_i$  assigned in the same manner as  $h_i, ul_i, ur_i, b_i$  in Pattern 3-1, and next mirror the pattern with respect to the centerline.

For the no-center-line part of  $p_1$ , its initial state is well-defined as the folded state composed by Pattern 5-2 ( $\square$ ). Then, we use the revisions given in Figure 4.15 to revise the third section to Pattern 5-3 ( $\square$ ). With the other two unfolds to remove the creases that do not really exist, the final assignment of the no-center-line part of  $p_1$  is shown on the left-hand side of Figure 4.20. The numbers indicate overlapping orders.

After the handling of  $p_1$ ,  $S_8$  is assigned as in Figure 4.17 to maintain the overlap principle. Next, the handling for  $p_2$  is based on the same idea with  $p_1$ . The  $MV$  assignment and the overlapping order of  $p_2$  are shown in the right-hand side of Figure 4.20.

The entire  $MV$  assignment in Figure 4.21 is achieved by joining  $p_1$  and  $p_2$ . We can

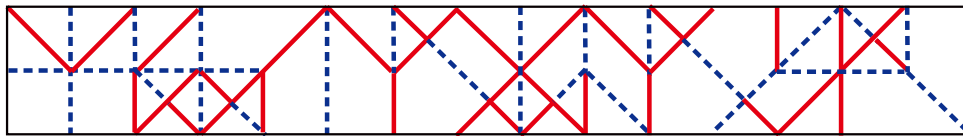


Figure 4.21: The entire  $MV$  assignment corresponding to the input pattern.

see from the overlapping orders in the illustration that this instance is folded into a final flat-folded state without self-intersection.

### 4.3 Future work and applications

Inspired by the existing conclusion which says that making decisions on the flat-foldability of box-pleating patterns of size  $m \times n$  where  $m \geq 4$  is NP-hard, the objective of our research was set as clarifying how much time cost would it be when the size of the input box-pleating patterns is restricted to be 1, 2, or 3. Our work fills two-thirds of the gap, remaining the flat-foldability of box-pleating patterns of size  $3 \times n$  as the only open question.



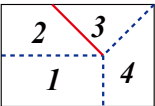

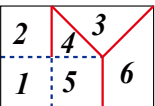

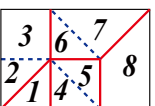

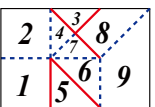

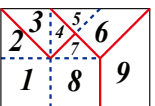

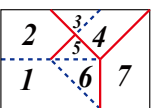

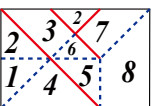

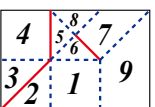

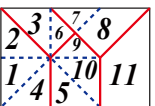

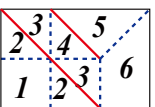








In both problems about box-pleating patterns of size  $1 \times n$  and  $2 \times n$ , we provided the linear-time algorithms. If we only focus on the decision problem itself but not the folding sequence as the output corresponding to the flat-foldable input patterns, the linear-time results are in fact natural conjectures inheriting the main conclusion: In box-pleating patterns of size  $1 \times n$  and  $2 \times n$ , global flat-foldability equals local flat-foldability.







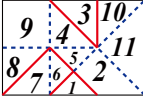


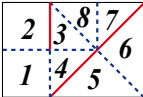


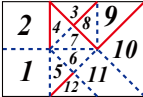


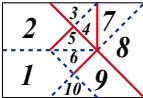




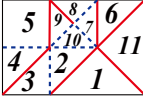

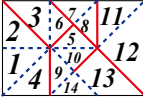







Research in Ref. [71] provided the practical results of some sized-restricted patterns. Using an exponential-time algorithm, they enumerated the flat-folded states of box-pleating patterns of size  $2 \times n$  for  $n \leq 6$ . Their results provided the first-step proof of our conclusion that all the box-pleating patterns of size  $2 \times n$  can be flat-folded, providing that the local flat-foldability is satisfied.

In the  $1 \times n$  and  $2 \times n$  cases, we detected the flat-foldability in square areas of size  $1 \times 1$  and  $2 \times 2$ . With a pre-filtering on the local flat-foldability, we found that every remained pattern restricted in the square areas, i.e., the locally flat-foldable patterns of size  $1 \times 1$  and  $2 \times 2$ , can also be globally flat-folded. However, this does not hold for square areas of size  $m \times m$  when  $m \geq 3$ . From the results in Ref. [71], we know that there exist three such patterns, as illustrated in Figure 4.10. In other words, these three patterns of size  $3 \times 3$  are patterns whose local flat-foldability does not equal global flat-foldability.

The set of all the patterns can be divided into three disjoint subsets: the subset of patterns not locally flat-foldable (which definitely also not globally flat-foldable), the subset of patterns locally but not globally flat-foldable, and the subset of patterns globally flat-foldable (which means they must be firstly locally flat-foldable). Among these three subsets, the first and the last one have the equivalence between local flat-foldability and globally flat-foldability, as our focus is the discordance of the second subset. For patterns with sizes beyond  $4 \times 4$ , there exist more than thousands of patterns divided into the second subset. The enormous number to an extent practically reveals the intractability of the decision of flat-foldability of box-pleating patterns of size  $4 \times n$ . But for a pattern of size  $3 \times n$  with only three instances classified into the second subset, what would the complexity be?

Table 4.6: Twenty representative types of connections of a connection section and a center-line part.

| Case | Patterns from the center-line part   | Patterns from the connection section  | MV assignment and overlap relationship for a typical example                                   |
|------|--|---|--|
| 1    | <br>1-1   | <br>2-5    | 1-1→ 3-1    |
|      | <br>1-3   |   | 1-3→ 3-1    |
|      | <br>1-2   |   | 1-2→ 3-1    |
|      | <br>1-4   |   | 1-4→ 3-1   |
|      | <br>1-5 |   | 1-5→ 3-1  |
|      | <br>2-1 |   | 2-1→ 3-1  |
|      | <br>2-2 |   | 2-2→ 3-1  |
|      | <br>2-3 |   | 2-3→ 3-1  |
|      | <br>2-4 |   | 2-4→ 3-1  |
|      | <br>2-6 |   | 2-6→ 3-1  |
|      |  | <br>3-1    |  |
|      |  | <br>3-2    |  |
|      |  | <br>3-5  |  |
|      |  | <br>3-6  |  |
|      |  | <br>3-9  |  |
|      |  | <br>3-10 |  |
|      |  | <br>3-13 |  |
|      |  | <br>3-14 |  |

|   |  |  |   |   |  |   |
|---|--|--|---|---|--|---|
| 2 | <br>1-1   | <br>2-1 |   | <br>2-5    |  | <p>1-1 → 3-3</p>   |
|   | <br>1-2   | <br>1-5 |   |   |  | <p>1-2 → 3-3</p>   |
|   | <br>1-3   | <br>1-6 |   |   |  | <p>1-3 → 3-3</p>   |
|   | <br>1-4   |  | <br>3-3    |   |  | <p>1-4 → 3-3</p>   |
|   | <br>2-1  |  | <br>3-7    |   |  | <p>2-1 → 3-3</p>    |
|   | <br>2-2 |  | <br>3-11 |   |  | <p>2-2 → 3-3</p>   |
|   | <br>2-3 |  |   |   |  | <p>2-3 → 3-3</p>   |
|   | <br>2-4 |  |   |   |  | <p>2-4 → 3-3</p>   |
|   | <br>2-6 |  |   |   |  | <p>2-6 → 3-3</p>   |
| 3 | All possible patterns  |  | <br>3-4  | <br>3-8  |  | <p>3-4</p>  <p>It is easy to flat fold all the crease patterns following an assignment of 3-4 in this manner because there is no other crease involved in the left-half section.</p> |
|   |  |  | <br>3-12 | <br>3-16 |  |   |

# Chapter 5

## Valid Overlapping Order Problems

In the traditional map folding problem, the decision is supposed to be given on the flat-foldability of a given regular grid pattern with an MV assignment. Even though we do not know whether there do exist polynomial-time solutions or not, intuitive solutions in exponential time can be easily given. Among the solutions, one is listing all the total orders of the squares in the grid pattern following the MV assignment at first. Then, for each order, making the decision on whether it represents a valid flat-folded state of the map, which means no penetration. As long as there exists one order corresponding to a valid flat-folded state, the input would be decided flat-foldable. This fact can also be rephrased as: Only in the case that all the total orders of squares cannot represent a flat-folded state, the input would be decided non-flat-foldable. In such a solution, an order of the squares could be recognized as a conjecture of the non-deterministic Turing Machine. The second topic of our research is about these orders. As mentioned before, we call these orders overlapping orders and say that the orders corresponding to flat-folded states are valid. Particularly, in some cases, instead of the total order on the set of squares, we also consider some special kind of partial orders, i.e., total orders on a certain subset of the set of squares. For example, the orders on the set of squares on the boundary of the map are called the boundary overlapping orders. We thus call the orders given onto the set of all the squares *the total overlapping orders* so as to keep the clarity.

As an overview, three problems are considered to comprise the mainstream of researches about the overlapping order (or we say the folded state). They are the decision problem on the validity of an input overlapping order, the enumeration problem for listing all the valid overlapping orders, and the counting problem on the total number of the valid overlapping orders. To the same objectives, the complexity they appear is from simple to difficult in this order. For example, the decision on the validity of an input overlapping order in a map of size  $m \times n$  can be made in linear time, while the enumeration of such valid overlapping orders arises to the time complexity  $O(mn^{mn+1})$  [80]. By now, there is no result in the counting problem. However, the study in Ref. [61], whose focus is counting the valid folded states of maps of size  $1 \times n$ , is an adequate instruction on how intractable the counting problem is.

Research about the decision problem on the validity of total overlapping orders in Ref. [80] implies the triviality of general folds. For general folds, it is sufficient to filter for the no self-penetration condition. The process of filtering can be limited in time linear in the size of the input. Therefore, the decision on the validity of both the total

overlapping order and any partial overlapping orders can always be concluded in linear time.

In this part, our research includes three topics. We studied the decision problems on all of them and the enumeration problem on the third topic. The first one changes the grid pattern of regular maps to the all-creases-existing box-pleating patterns. Because the arrangement of triangles is different from the squares in grid patterns physically, the filtering process also has to be changed to apply to the box-pleating patterns. Then, as the second and the third topic, since general folds induce triviality, we studied the validity of both the total overlapping order and a particular kind of overlapping order, the boundary overlapping order with the restriction of simple folding. More precisely, all the folds during the whole folding process have to be simple folds instead of the general ones.

## 5.1 Valid Overlapping Order Problems in box-pleating Patterns

The decision on the validity of given overlapping orders of the faces in the general fold model, as we mentioned, can be decided by eliminating the self-penetrations. We here give the conclusions for both the cases when MV assignment is given and not given.

Considering the overlapping relations of the faces, namely which face should be folded over which face and which face should be folded below which face, we can define a total order  $<$  on these faces, specified as follows. In a valid final flat-folded state, all of the triangles(faces) comprise the set  $T = \{t_1, t_2, \dots, t_{4mn}\}$ . For any  $t_a, t_b, t_c$  in  $T$ , we have:

- (1) Not  $t_a < t_a$ .
- (2) If not  $t_a < t_b$ , then  $t_b < t_a$ .
- (3) If  $t_a < t_b$  and  $t_b < t_c$ , then  $t_a < t_c$ .

Clearly,  $<$  is a strict partial order on the set of all the triangle faces.

Now, we build a directed graph  $G = \{V, E\}$  to indicate the triangles and the overlapping relations of them. Every triangle is abstracted to a vertex in  $V$ . Then, the above-defined total order  $<$  can be abstracted as directed edges. We can find that every valid overlapping order should have a  $G$  without any circuits. Namely, only acyclic graphs can correspond to valid overlapping orders. We call this necessary condition the *acyclic condition*. It only ensures that the input order is an overlap of all the triangles but has no relation with the self-penetrations.

The self-penetration can be located on the three edges of the final triangle shape. The layers do not have penetrations if and only if any pair of connected parts do not intersect with another pair along the same edge. This condition is described as the *butterfly condition* in Ref. [80]. We follow their terminology and illustrate the corresponding states for the box-pleating patterns in Figure 5.1.

When the input includes the MV assignment, the acyclic condition and the butterfly condition together construct the necessary and sufficient condition of the validity of given orders. When the input does not include the MV assignment, the acyclic condition is self-evident. Hence, our method to solve this decision problem follows the below description.

For the case that only takes the order on the set of triangles as input, we consider a pair of triangle neighbors in the all-creases-existing box-pleating pattern as a unit. The



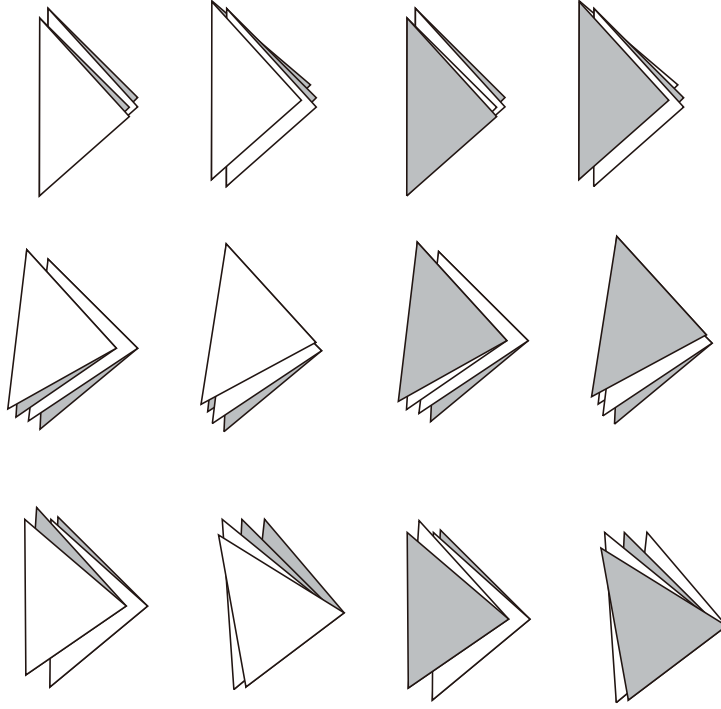


Figure 5.1: Valid states of layers under the butterfly condition.

check is about if two units with their creases (between each pair of neighbors) located the same would form self-penetration. If the check corresponds to positive, the given order is decided to be valid.

If the input involves not only the order on the set of triangles but also a certainly defined MV assignment, we further have to check the acyclic condition. We assign edges to  $G$  according to the order  $<$  decided by the MV assignment and then check if the edges do not form cycles in  $G$ . Then, we have to check the uniformity between the MV assignment and the given order. Since the MV assignment can be uniquely decided by the overlapping of neighbor triangles, we check if the given order on all the triangles follows the MV assignment. If both checks correspond to positive, the given order is decided to be valid.

## 5.2 Valid Total Overlapping Orders with Simple Folds

This research is about the decision problem on the validity of total overlapping orders of maps(grid patterns) of size  $m \times n$  in the simple fold model. Concretely, it asks whether an input overlapping order  $O$  of all the squares of a map of size  $m \times n$  can represent a final flat-folded state of the map that is reachable with only simple folds.

Simple folds are distinguished from the general folds by limiting that (1) a simple fold is along a unique line segment and (2) only some continuously adjacent layers in the flat-folded state(whose surfaces touch each other) are folded along a single *crease line*(defined as a set of creases aligning on the same line and passing through the map). Clearly, the requirement of simple folds shrinks the set of valid flat-folded states. Without this requirement, the validity of a given overlapping order can be decided using filtering

on only one condition, that the input overlapping order does not represent a state with self-penetration. However, the restriction of simple folds asks for more filtering and makes the problem more complicated. It is interesting to think of what roles the simple folds play in the map folding problems introduced in the last chapter and in the valid overlapping problems here. In [9], they proved that for input as a grid pattern with MV assignment, whether it can be flat-folded or not with only simple folds can be decided in linear time( $O(mn)$ ). General folds as folding operations are intractable. To handle them, we must look at the folded states. In contrast, even though simple folds put more restrictions upon the folded states, they are easy to handle as folding operations. The linear-time algorithm in Ref. [9] following a process that finds the possible simple folds in the MV assignment at each step to reduce the map and the assignment until the map is reduced to a size of  $1 \times 1$ .

However, the valid overlapping order problem cannot be handled with this single-direction solution because even the MV assignment is computed out, there may exist an exponential number of flat-folded states corresponding to the same MV assignment. In other words, if we follow the idea of [9], we have to check whether the input is in the set composed of all the flat-folded states corresponding to the same MV assignment with the input, which costs exponential time. Therefore, instead of the filtering on the final folded states as in Ref. [80] and the computation from the MV assignment in Ref. [9], we use a two-direction method: finding out the possible simple folds based on the MV assignment and decide the validity and the ordering of them based on the input overlapping order. With such a method, we finally provide a linear-time solution.

### 5.2.1 Notations

A map with  $m \times n$  congruent squares arrayed on  $m$  rows and  $n$  columns is denoted by  $M_{m,n}$ . The front and the back of  $M_{m,n}$  are distinguished. In  $M_{m,n}$ , the *crease pattern* is a grid pattern consisting of all the edges of the squares other than the ones on the boundary of the map. These edges are called *creases* and their non-boundary endpoints are called *vertices* (still following the terminology in Ref. [9]). The *Mountain-Valley assignment*(*MV assignment*) is a mapping from the set of creases to the set  $\{M, V\}$  which models the way to fold every crease along either direction. In  $M_{m,n}$ , each vertex has degree 4. Therefore, a vertex of it is locally flat-foldable if and only if three of them are assigned the same, and the remaining one is assigned differently.

We use a right-handed Cartesian coordinate whose unit length is equal to the side length of the squares to formalize the squares and creases of  $M_{m,n}$ . As illustrated in Figure 5.2,  $M_{m,n}$  is assumed to be at the first quadrant with its bottom row aligned with the  $x$ -axis from the origin. The square whose lower-left vertex is located at  $(i, j)$  before any fold is referred to as  $s_{i,j}$  ( $0 \leq i < n, 0 \leq j < m$ ). *Neighbor squares* represents either a pair of squares  $s_{i,j}, s_{i+1,j}$ , or a pair of squares  $s_{i,j}, s_{i,j+1}$ . Compared to neighbor squares defined by the initial state of the map, we say a pair of squares is *adjacent* in a certain folded state (maybe a middle state) if their surfaces touch and overlap each other. Without loss of generality,  $s_{0,0}$  is supposed to always face the front side up during the whole folding process. Then, for creases,  $h_{i,j}$  refers to a horizontal crease whose left endpoint is located at  $(i, j)$  and  $v_{i,j}$  refers to a vertical crease whose bottom end is located at  $(i, j)$ . Since simple folds are always along a line segment. For convenience, We use  $h_a$

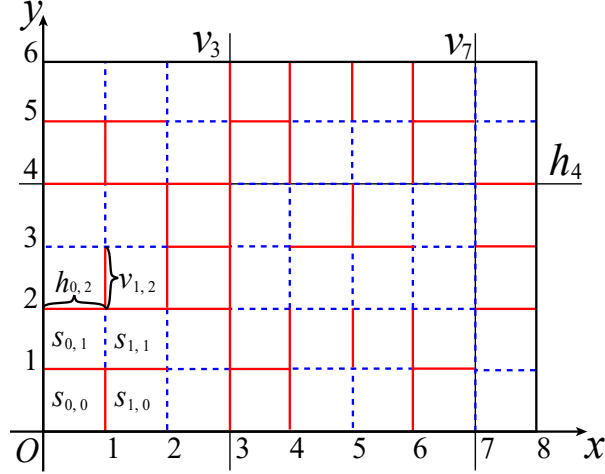


Figure 5.2: An example of  $6 \times 8$  map.

to indicate the set of all the creases located on  $y = a, 1 \leq a \leq m - 1$  and  $v_b$  to indicate the set of all the creases on  $x = b, 1 \leq b \leq n - 1$  and call such  $h_a$ s and  $v_b$ s *crease lines*. Figure 5.2 gives an example map of size  $6 \times 8$ .

Next, about the foldings, we use *simple folds* as mentioned above and illustrated in Figure 2.1. The following definitions are about the foldings. *Simple folds* focused in this research are exactly the same with the *some-layers simple fold* model in Ref. [9]. A simple fold is along a crease line, and it avoids interference on other layers. [9] also provided two other simple fold models, the single-layer simple fold model, and the all-layer simple fold model, while we do not concern other models of simple folds in this research. We then define the whole folding process, a sequence composed of simple folds, as the *simple folding sequence*. Mathematically, it is a continuous reconfiguration of the map from its initial state  $R_0$  (a single sheet of paper) to a target flat state  $R_t$  of the  $1 \times 1$  size without tearing, stretching, or self-penetrating, and every reconfiguration is finished by only simple folds. We then have the observation that every partly flat-folded state during the folding is always in the shape of a rectangle.

An *valid overlapping order* is a list of the layers ordered from bottom to top in a valid final flat-folded state reachable by only simple folds. It is represented by a fixed sequence of the elements in  $\{s_{i,j} | 0 \leq i < n \text{ and } 0 \leq j < m\}$ .

By now, we can define the problem *Valid Total Overlapping Order in a Simple Fold Model (VTOS)* as below.

(VTOS) When an overlapping order  $O$  of all the squares of  $M_{m,n}$  is given, is  $O$  a valid ordering reachable by a simple folding starting from  $R_0$ ?

In the final flat-folded state  $R_t$ , each square forms a single layer. As mentioned, two squares are adjacent in a partly or the final flat-folded state  $R_t$  if their surfaces touch each other. When a pair of squares  $s_{a_1,b_1}$  and  $s_{a_2,b_2}$  are adjacent and  $s_{a_1,b_1}$  is supposed to be folded below  $s_{a_2,b_2}$  in  $R_t$ , the *adjacency relation* between them is denoted by  $s_{a_1,b_1} \leftrightarrow s_{a_2,b_2}$ . As an extension, the adjacency relation among a set of squares  $\{s_1, s_2, \dots, s_i\}$  in which all the pair of  $s_a$  and  $s_{a+1}$  have adjacency relation  $s_a \leftrightarrow s_{a+1}$  is indicated by the tuple  $(s_1, s_2, \dots, s_i)$ . The maximal tuple is a closure of  $\leftrightarrow$  by  $\leftrightarrow^*$ , which can be constructed as follows: (1)  $s_i \leftrightarrow^* s_i$ ; (2)  $s_i \leftrightarrow^* s_k$  if  $s_i \leftrightarrow^* s_j$  and  $s_j \leftrightarrow s_k$ .

In this research, we use the *crimps* and *end-folds* as the basic folding operations to describe the folding process. As illustrated in Figure 2.1, an end-fold is a folding of either the first or the last crease in the one-dimensional map, whereas a crimp is a folding of a pair of adjacent creases with different labels. Crimps along two crease lines can be viewed as a combination of simple folds along single crease lines. We use such descriptions only for the purpose of clarity and a better connection with our next research on the boundary overlapping order. All of our conclusions still hold even if the folds are described by the simple folds along single crease lines but not crimps (along two crease lines). It is clear that any simple folding applied on  $M_{m,n}$  involve at most  $m + n - 2$  crimps and end folds.

### 5.2.2 Outline

In our method, the procedure to decide the validity of the input order of all the squares  $O$  is specified as follows:

- (1) Computing the MV assignment from  $O$  and checking if it satisfies the local flat-foldability at every vertex; if not, the algorithm should return false immediately. Otherwise, the algorithm should proceed to the next step.
- (2) Checking a necessary condition of the MV assignment to be flat-foldable with only simple folds. The condition is a check for the satisfaction of two facts: (a) every simple fold applies on a single crease line in the current map (the map is reduced to a smaller one by each step of folding), which means that all the creases on such a crease line should have the same MV assignment; (b) every simple fold is either along horizontal lines or vertical lines. If the condition is not satisfied, the algorithm should return false. Otherwise, we construct an incomplete order of simple folds  $P = (p_0, p_1, \dots, p_{t-1})$  with  $t \leq m + n - 1$  according to this MV assignment. Each  $p_k$  describes the maximal set of multiple parallel crease lines folded simultaneously.
- (3) Constructing a directed graph  $G$  to describe the adjacency relations of the squares.  $G$  records the adjacency relations that can be obtained from  $P$ . Each vertex of  $G$  indicates a unique square, and each edge represents the adjacency relation between a pair of squares in the final flat-folded state. As a partial order on the set of simple folds,  $P$  provides part of the information to decide the validity of  $O$ . After this step,  $G$  becomes a graph that uniquely indicates groups of squares (squares in each group have to be folded simultaneously), all the adjacency relations within each group, and the *interdependencies* among groups.
- (4) Traversing  $O$  while adding the remaining edges to  $G$  with respect to the interdependencies among groups and checking if  $O$  follows the construction of  $G$ .

In these steps, (2) and (3) can be concluded at the same time, and only some edges in  $G$  are assigned before applying (4). This procedure can be realized in time linear in the size of the input.

### 5.2.3 Deciding the validity of the given order

#### (1) Computing the MV Assignment

As introduced before, our method uses a two-direction computation. The first one is finding out the possible simple folds based on the MV assignment. Thus, firstly, we have to figure out the MV assignment. It is concluded through a traverse of  $O$ . The label

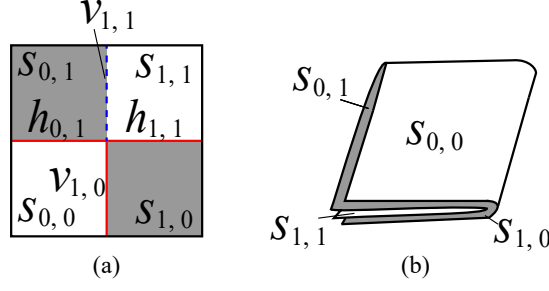


Figure 5.3: The checkerboard pattern of a  $2 \times 2$  map and its corresponding folded state.

of any crease is uniquely decided by the overlapping relation between its two incident squares.

In our assumption,  $s_{0,0}$  always face front side up. This induces that  $s_{i,j}$  would face front side up (down) when  $i + j$  is even (odd) in  $R_t$ . This property can also be described as a checkerboard pattern, as illustrated in Figure 5.3.

The computation of the MV assignment is made in the following manner. Consider an arbitrary  $s_{i,j}$  facing front side up in  $R_t$ , if it is positioned below  $s_{i+1,j}$  (or one of  $s_{i-1,j}$ ,  $s_{i,j+1}$ ,  $s_{i,j-1}$ ), the crease between them must be labeled V. If it is positioned over  $s_{i+1,j}$  (or one of  $s_{i-1,j}$ ,  $s_{i,j+1}$ ,  $s_{i,j-1}$ ), the crease must be labeled M. Following this way, the labels of all the creases can be figured out through a traverse of  $O$ . A simple example of a  $2 \times 2$  map is illustrated in Figure 5.3. The overlapping order  $(s_{1,0}, s_{1,1}, s_{0,1}, s_{0,0})$  of the folded state in (b) is given as the input. From the fact that  $s_{0,0}$  is positioned above  $s_{1,0}$  and  $s_{0,1}$ , we can find out that  $h_{0,1}$  and  $v_{1,0}$  are labeled Ms. Because  $s_{1,1}$  faces the front side up and is positioned below  $s_{0,1}$  while above  $s_{1,0}$ , we know that  $h_{1,1}$  is labeled M and  $v_{1,1}$  is labeled V. The MV assignment on all the creases becomes clear. By such a manner, the MV assignment is obtained in  $O(mn)$  time, linear in the size of the input.

Sometimes when  $O$  is not a valid overlapping order, at this step, it may correspond to an MV assignment not flat-foldable. Since the condition for checking global flat-foldability remains unknown, we here only check if such an MV assignment is (everywhere) locally flat-foldable. If it is not everywhere locally flat-foldable, we immediately return false. Even though this check can be avoided in the first step, it is unavoidable during the later computations. For clearness, we arrange this procedure here. The check for local flat-foldability can be concluded in  $O(mn)$  time by checking the MV assignment [14].

## (2) The incomplete order $P$ with respect to horizontal and vertical folds

The MV assignment can partly decide the order of simple folds. In this part, we construct an incomplete order of simple folds  $P = (p_0, p_1, \dots, p_{t-1})$  with  $t \leq m + n - 1$  according to the MV assignment. Here the simple folds are represented by the crease lines and they will be concreted to crimps and end-folds in the after procedure. This incomplete order is supposed to have two properties: (1) each  $p_k$  where  $k < t - 1$  is either a maximal set of some  $h_a$ s or a maximal set of some  $v_b$ s perpendicular to those involved in  $p_{k+1}$  and (2) the crease lines in  $p_k$  are folded directly after  $p_{k-1}$  and before  $p_{k+1}$ . The above indication induces the uniqueness of elements in  $p_k$ , no matter a single crease line or multiple crease lines. With the permission of empty set as  $p_0$ , any  $P$  can be integrated in a form where

any  $p_{2i}$  is composed of  $h_a$ s and  $p_{2i+1}$  is composed of  $v_b$ s. Also,  $|p_0| + |p_2| + \dots = m - 1$  and  $|p_1| + |p_3| + \dots = n - 1$  since there exist  $m - 1$  horizontal crease lines and  $n - 1$  vertical crease lines in an  $m \times n$  map. When such  $P$  does not exist, it means that the MV assignment is not simple-foldable. Then, we feedback that  $O$  is invalid and end the process.

Roughly speaking, the method introduced in Section 14.2 in Ref. [23] can be applied to construct  $P$  with a slight revision. Their method concerns the flat-foldability of a given MV assignment by simple folds. By Lemma 14.2.1 in Section 14.2 of [23], we know that the crease lines each with all the creases on it labeled the same must be folded at first. These creases comprise  $p_0$ . Even though the inner order of  $p_0$  remains unknown, we can still reduce the map with respect to  $p_0$ . The inner order of any  $p_i$  does not affect the shape of  $R_{i+1}$  but only affects the resulting overlapping order of the involved squares in such layers. This means that  $M_{m,n}$  can always be folded to the shape of the largest rectangle separated by the folds. We cannot read out the overlapping order of the layers in the partially folded state  $R_1$ , but if we view  $R_1$  as a new map, its MV assignment is clear. Then, for the new map, we can use the same checking method to obtain  $p_1$ . Repeating this process and finally either the incomplete order of crease lines  $P = (p_0, p_1, \dots, p_{t-1})$  can be computed out or at some steps, we cannot find any crease line labeled all the same, which means that the MV assignment is not simple-foldable. In the former case, the MV assignment of  $M_{m,n}$  is flat-foldable in the simple fold model, the computation of  $P$  should end as  $M_{m,n}$  is reduced to a map of size  $1 \times 1$ . In the latter case, we conclude that the input is invalid. Because the traverse involves all the  $(m - 1) \times n + (n - 1) \times m = 2mn - m - n$  creases, this computation costs  $O(mn)$  time.

Following is an example of the above process. For the instance shown in Figure 5.2, such an incomplete order  $P$  on the set of its crease lines can be obtained as follows. The entire folding until it is reduced to the size  $1 \times 1$  consists of 5 steps. The first three steps are illustrated in (a), (b) and (c) in Figure 5.5, respectively. In the first step, only  $v_3$  and  $v_7$  are entirely labeled M and V, respectively. Hence, the map should be folded along them, i.e.,  $p_0 = \{v_3, v_7\}$ . The earlier check on the local flat-foldability is equivalent to the check of the feasibility of the simple folds here. If the local flat-foldability was not checked, at this step, we should further check the labels of creases located symmetrically with respect to the folded crease lines. Concretely, in this step, pairs of creases in the two areas of size  $6 \times 3$  adjacent to  $v_3$  and the two columns adjacent to  $v_7$  are respectively verified to match each other. Next, we check if we can sequence  $v_3$  and  $v_7$  to achieve a feasible folding without self-penetration, which can be finished by viewing the map as a one-dimensional map and the crease lines as crease points. The corresponding one-dimensional map and the state after folding  $v_3$  and  $v_7$  are illustrated in Figure 5.4. In this case, the answer is yes since they form two end-folds whose order does not matter. When the input is changed and the MV assignment we computed corresponds to the same one-dimensional map on the left side of Figure 5.4 but has  $p_0 = \{v_3, v_4\}$ , both as mountain folds, then self-intersection would happen.

Similarly, in the second step, we find that  $p_1 = \{h_2, h_4\}$  in the reduced map, which forms a crimp. Different from  $p_0$  that induce only one possible overlapping order of the layers, this time, different inner orders of  $p_1$  would induce different overlapping orders. This practically indicates that a given MV assignment may give more than one overlapping order. Then, in the third step,  $v_2, v_4$  are located on the same line segment, and the

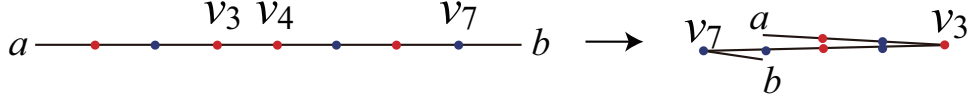


Figure 5.4: The corresponding one-dimensional map before and after folding  $v_3$  and  $v_7$ .

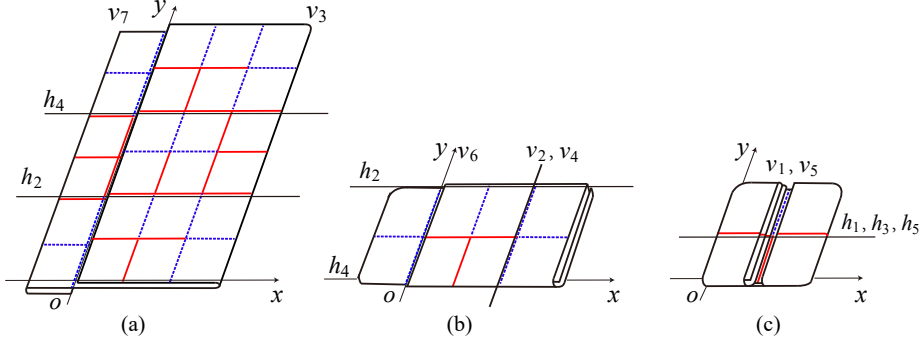


Figure 5.5: Obtaining the partial order based on a known  $MV$  assignment via a simple folding.

folding is along  $p_2 = \{v_2, v_4, v_6\}$  and involves two end folds. The fourth step and the last step are omitted in the illustrations. They are composed of  $p_3 = \{h_1, h_3, h_5\}$  and  $p_4 = \{v_1, v_5\}$ , respectively. Since after  $p_4$ , the map is reduced to size  $1 \times 1$ , we know that this  $MV$  assignment can be flat-folded in the simple fold model with  $P = (p_0, p_1, p_2, p_3, p_4)$  along all the  $6 + 8 - 2 = 12$  crease lines.

### (3) Preparation for the following computations

We avoided some notations and definitions in the last section because they would not make sense until the operations in (1) are specified. Now, we introduce the necessary ones for later description. The initial unfolded state of the map is defined  $R_0$ . Then, we use a sequence  $\mathcal{R} = (R_0, R_1, R_2, \dots, R_t)$  to indicate the partly flat-folded states with respect to the simple folding sequence  $P = (p_0, p_1, \dots, p_{t-1})$ . In this sequence, each  $R_{j+1}$  indicates the flat-folded state directly after folding the folds in  $p_j$ . As introduced before, the final flat-folded state of size  $1 \times 1$  is denoted by  $R_t$ . We also assume that the location of  $s_{0,0}$  in every  $R_k$  is kept as  $(0, 0)$ .

We have also defined the closure  $\leftrightarrow^*$  of the adjacency relation  $\leftrightarrow$  to describe the adjacency relation of a set of squares. These squares can be aligned as a sequence which indicates their overlapping order. However, in the case that the inner order of the folds in every  $p_j$  is not decided, we can only know which squares are supposed to be involved in the base set of squares of  $\leftrightarrow^*$  but cannot capture the information of their overlapping orders at each step. To further emphasize the elements supposed to be involved in the  $\leftrightarrow^*$  in every certain  $R_j$ , we use a 3-tuple  $g(j, x, y)$  for  $j, x, y \in \mathbb{Z}$ , which is called a *square group*, as a description of the squares located at the coordinate  $(x, y)$  in  $R_j$ . Specifically, in  $R_0$ ,  $g(0, x, y)$ s are assigned in the following manner. For every  $(x, y)$  with  $0 \leq x < n, 0 \leq y < m$ , let  $g(0, x, y) = \{s_{x,y}\}$ . For other coordinates  $(x, y)$ s where no

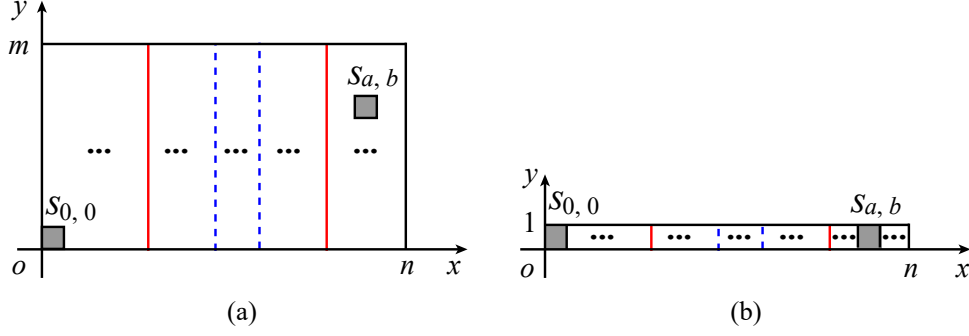


Figure 5.6: The uniformity of the positions in an  $m \times n$  map and a  $1 \times n$  map.

square locates initially, let  $g(0, x, y) = \emptyset$ . Because we have assumed that the location of  $s_{0,0}$  in every  $R_k$  is  $(0, 0)$ ,  $s_{0,0}$  belongs to every  $g(j, 0, 0)$  in the partly flat-folded state  $R_j$ . Unless especially noted, in the following descriptions, we use the notation  $p_k$  to refer to an arbitrary element in  $P$  and use  $(x, y)$  to refer to an arbitrary coordinate. After folding the crease lines of  $p_k$ , the partly flat-folded state  $R_k$  would become  $R_{k+1}$ . Also, to avoid unnecessary computations, we only consider the non-empty square groups. With these limitations,  $g(k+1, x, y)$  uniquely indicates the squares supposed to be folded to the coordinate  $(x, y)$  in  $R_{k+1}$ . It is necessary to give the above facts a theoretical description and proof. We provide these in the following Observation 5.2.1 and Lemma 5.2.2.

**Observation 5.2.1.** *The squares involved in the same rows (columns) in  $R_i$  would never be separated into different rows (columns) in  $R_j$  for any  $j > i$ .*

**Lemma 5.2.2.** *Whether the inner order of the folds involved in  $p_k$  is clear or unclear, squares involved in  $g(k+1, x, y)$  are uniquely decided since they are independently decided by  $P$ . The inner order of  $p_k$  only affects the overlapping order of the squares involved in  $g(k+1, x, y)$ , i.e., the indices of the squares in the sequence.*

*Proof.* The proof uses the illustration Figure 5.6 (a). Without loss of generality, we assume that  $p_k$  is comprised of vertical crease lines. Then, for a square  $s_{a,b}$  in  $g(k, x', y')$ , we give some explanations about its corresponding  $g(k+1, x, y)$ .

It is clear that  $y' = y = b$  holds for  $s_{a,b}$ . We focus on the value of  $x$  for  $s_{a,b}$ , which can be computed using a one-dimensional model illustrated in (b). The folding is composed of a sequence of crimps and end folds [9]. Corollary 12.1.5 of [23] clarifies that the value of  $x$  for  $s_{a,b}$  is certain after folding all the crease lines in  $p_k$ . The same analysis can be applied to the case when the crease lines in  $p_k$  are horizontal. Therefore, by the recursion of  $k$ , the squares involved in any  $g(k+1, x, y)$  are certain. Lemma 5.2.2 is proven.  $\square$

Lemma 5.2.2 indicates that the set of squares located at  $(x, y)$  in  $R_{k+1}$  is comprised of the set of squares located at some certain  $(x', y')$ s in  $R_k$ . With this lemma, next we define  $g(k+1, x, y)$  as  $g(k+1, x, y) = \{g(k, x', y') \mid g(k, x', y') \text{ is supposed to be folded to } (x, y) \text{ by } p_k\}$ . Namely,  $g(k+1, x, y)$  is a set comprised of sets. Each of its element is a  $g(k, x', y')$  supposed to be folded to  $(x, y)$  by the folds involved in  $p_k$ . Distinctive properties of folds decide whether they can lead to a certain overlapping order or not. For example, the folds  $f_1$  and  $f_2$  in Figure 5.7 lead to two possible overlapping orders, while the folds  $f_1$  and  $f_2$



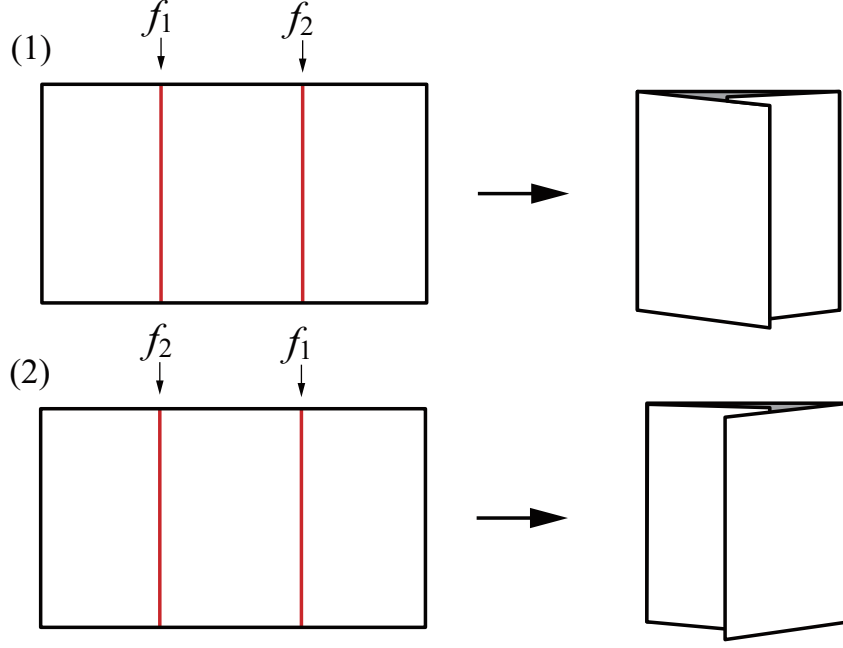


Figure 5.7: Different orders of folds leading to different possible overlapping orders.

in both cases in Figure 5.8 lead to the unique possible overlappings. When discussing a  $g(k+1, x, y)$ , if the overlapping order of its elements is uniquely decided in  $R_{k+1}$ , then we say that the inner order of  $g(k+1, x, y)$  is *clear*, otherwise we say that the inner order of  $g(k+1, x, y)$  is *unclear*.

In the above definitions, the *adjacency relation* is defined to describe the relationship between squares. Here, noticing that a pair of  $g(k, x', y')$ s involved in the same  $g(k+1, x, y)$  may have their relative overlapping decided even if the adjacency relation of all the squares in a  $g(k+1, x, y)$  remains unclear, the definition of *adjacency relation* is thus extended to the relationship between a pair of groups. If the uppermost square in  $g(k, x'_1, y'_1)$  touches the lowermost square in  $g(k, x'_2, y'_2)$  in  $R_u$  where  $u > k$ , we say that they are adjacent and denote them by  $g(k, x'_1, y'_1) \leftrightarrow g(k, x'_2, y'_2)$  or  $g(k, x'_2, y'_2) \leftrightarrow g(k, x'_1, y'_1)$  according to which side they face up in  $R_t$ . Then, for the adjacency relation between groups during the whole folding, we have Lemma 5.2.3.

**Lemma 5.2.3.** *In any partly flat-folded state  $R_k$ , a pair of adjacent squares would still be adjacent in  $R_t$ , even though the adjacency relation between them may be converse according to whether the sides they face up in  $R_k$  and  $R_t$  are the same.*

*Proof.* Following the folding process, a pair of adjacent squares in a partly flat-folded state would always be folded together by the subsequent folds. The simple fold model folds continuous layers either above or below all the other layers. Such a definition ensures that no other layers can be inserted between them. Thus, a pair of adjacent squares in  $R_k$  are also adjacent in  $R_{k+1}$ . Because both  $g(k+1, x, y)$  and the adjacency relation are defined recursively, the adjacency between a pair of squares in  $R_k$  would be kept in  $R_t$ . Their overlapping order in  $R_t$  is decided by their initial coordinates, which can be checked using a checkerboard pattern, where the white and black panels correspond to their facing-up sides. With these facts, Lemma 5.2.3 is proven.  $\square$

Besides the adjacency relation, there is another type of relation between groups, which we call the *interdependency*. It describes the groups affected by the same folds. When the overlapping of one changes, the change on the overlapping other groups should be in accordance with it. Terminologically, the crease lines of  $p_k$  divide  $R_k$  into multiple rectangles. Each of them is viewed as a single layer in  $R_{k+1}$  and such rectangles comprises a set, denoted by  $L_k = \{l(k, i)\}$  with  $i = \{0, 1, \dots, |p_k|\}$ . We consider a  $g(k+1, x, y)$  composed of  $g(k, x', y')$ s from different  $l(k, i)$ s. In the partly flat-folded state  $R_{k+1}$ , once the inner order of  $g(k+1, x, y)$  becomes clear, the overlapping order of  $l(k, i)$ s also become clear. Furthermore, if  $g(k+1, x_2, y_2)$  is composed of  $g(k, x'_2, y'_2)$ s from the same rectangle  $l(k, i)$ s, then the inner order of  $g(k+1, x_2, y_2)$  also becomes clear. For such  $g(k, x', y')$  and  $g(k, x'_2, y'_2)$  related by the same  $l(k, i)$ s, we say that  $g(k, x', y')$  and  $g(k, x'_2, y'_2)$  are *associated* and call the relations among such groups the *interdependencies*.

With the terminology defined and redefined in the last paragraph, we can now describe how we compute a folding process corresponding to a valid input. We have introduced how to construct the incomplete order  $P$ , which is a partial order of simple folds decided by their directions. But to decide the validity of  $O$ , it is not enough to compute based on  $P$  itself, since part of the information about interdependencies included in  $O$  is not recorded by  $P$ .  $O$  should be verified by checking all the adjacency relations, whereas not all the adjacency relations can be decided only by  $P$ . To solve this problem, we further introduce a directed graph  $G = \{V, E\}$  in the next step.

#### (4) The Directed Graph $G$ and the Validity Checking

The directed graph  $G$  is constructed from two directions, based on  $P$  and  $O$ , respectively. Each edge in  $E$  represents the adjacency relation between a pair of squares in  $R_t$ . When the construction of  $G$  is concluded,  $E$  must include  $m \times n - 1$  directed edges. These edges form a Hamiltonian path. The validity of the input  $O$  is decided according to the adjacency relations in  $G$ . Initially,  $G_0 = \{V_0, E_0\}$  is constructed as a graph with  $V_0 = \{s_{i,j} \mid 0 \leq i < n, 0 \leq j < m\}$  and  $E_0 = \emptyset$ . Each square  $s_{i,j}$  is abstracted as a node  $s_{i,j}$  of  $G_0$ . Since  $G$  is constructed step by step, we denote its currently specified state achieved directly after folding each  $p_k$  by  $G_{k+1} = \{V_{k+1}, E_{k+1}\}$ . Sometimes the directions of elements in  $E$  are not fixed and such states represent all the possible  $R_{k+1}$ s. In other words, every  $G_{k+1}$  represents a certain state preceding  $G$ . It records the adjacency relations decided by  $P$  and involved in  $R_{k+1}$ .

Corresponding to the final state,  $R_t$ ,  $G_t$  is obtained when all the elements of  $P$  are considered. Note that  $G_t$  does not represent the final state of  $G$ . In the next step, we traverse  $O$  and complete  $G$  according to  $O$ . If  $G$  indicates a valid state reachable by simple folds and  $O$  is verified to be consistent with it, we claim that  $O$  is valid.

The adjacency relations decided by  $p_k$  are indicated in  $G$  by assigning edges to  $G_k$ . As mentioned before, the adjacency relation of two squares known to be adjacent in  $R_t$  can be figured out by a parity check on their initial coordinates. Thus in the following, we only consider the adjacency relations in  $R_t$  when assigning the edges. In every  $G_k$ , a directed edge from  $s_{a,b}$  to  $s_{c,d}$  represents the adjacency relation  $(s_{a,b}, s_{c,d})$  in  $R_t$ .

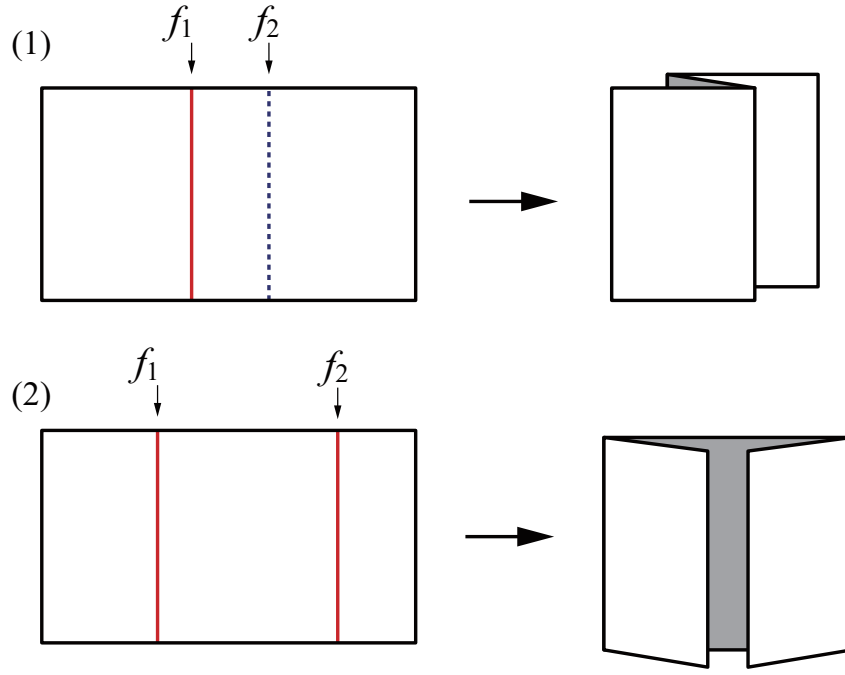


Figure 5.8: Inner orders of  $p_k$  directing to the same  $g(k+1, x, y)$ .

#### (5) Construction of the adjacency relation graph $G$ from $P$

$G$  is achieved through three phases. As above-mentioned,  $G$  is supposed to be specified from two different directions considering  $P$  and  $O$ , respectively. In the first two phases, we consider  $P$  and include all the possibilities in  $G$ , while in the third phase, we reference  $O$  to rule out the possibilities which is not in accordance with the input. More precisely, in the first phase, we consider the adjacency relations inside the groups with clear inner orders and record such adjacency relations as edges. In the second phase, the interdependencies among groups with unclear inner orders are recorded, not as edges in  $G$ , but as indices of the elements of associated groups. In the last phase, we construct  $G$  referencing  $G_t$  and  $O$  by assigning the edges according to the recorded interdependencies and check the consistency between  $O$  and  $G$  so as to determine the validity of the given overlapping order. Every  $G_{k+1}$  is constructed from  $G_k$  according to  $p_k$ .  $G_t$  is obtained before traversing  $O$ .

Now we give the specified instructions for each phase. In the first phase, we concern  $g(k+1, x, y)$ s whose inner orders are clear. Note that  $g(k, x', y')$ s comprising  $g(k+1, x, y)$  can have unclear inner orders. The adjacency relations involved in such  $g(k+1, x, y)$ s are assigned as (undirected) edges in  $E$ .

Since the shape after each fold is clear, the  $g(k, x', y')$ s which comprise  $g(k+1, x, y)$  are certain for each  $k$ . A  $p_k$  can involve both ordered folds and disordered folds. Whether the folds are ordered or not and the order of the ordered folds can be decided with the same method mentioned in Section 14.2 in Ref. [23]. The inner order of  $p_k$  influences the inner order of  $g(k+1, x, y)$  but not completely. To explain this, we note that both orders  $(f_1, f_2)$  and  $(f_2, f_1)$  of  $p_k$  in the two cases shown in Figure 5.8 directs to the same  $g(k+1, x, y)$ . Above all, there are three possible cases where the inner order of  $g(k+1, x, y)$  is clear.

Case 1.  $|p_k| = 1$ .

Case 2. The order of folds in a certain  $p_k$  is uniquely decided.

Case 3.  $g(k, x', y')$ s relocated by the folds in  $p_k$  do not overlap each other.

The above analysis brings us the task of assigning edges in  $E$  properly so that all the possible adjacency relations involved in  $g(k+1, x, y)$  can be represented and specified in the subsequent handlings. Our method of assigning edges for a  $g(k, x', y')$  with an unclear inner order is thus arranged as follows.

Two nodes are attached to every  $g(k, x', y')$  with an unclear inner order, including a *top pseudo node* and a *bottom pseudo node*. They are respectively denoted by  $g_t(k, x', y')$  and  $g_b(k, x', y')$ , indicating the uppermost and lowermost element of  $g(k, x', y')$  finally (means the overlapping order in  $R_t$ , which might be conversed comparing to the one in  $R_k$ ). With these two pseudo nodes which does not represent squares, the adjacency relations between  $g(k, x', y')$  and its directly upper and lower adjacent groups in  $R_{k+1}$  can be specified in  $G$  by assigning an incoming edge to  $g_b(k, x', y')$  and an outgoing edge from  $g_t(k, x', y')$ . In the same manner, all the edges between the corresponding adjacent nodes in  $G$  can be specified according to  $P$ . If the inner order in the next partly folded state, i.e., the inner order of  $g(k+1, x, y)$  is unclear, this edge assigning process is skipped. However, we still attach the two pseudo nodes  $g_t(k+1, x, y)$  and  $g_b(k+1, x, y)$  to  $G$  for the subsequent process. In the third phase, when we specify the edges and their directions according to  $O$ , all the pseudo nodes would be removed as soon as the inner orders of their groups become clear.

Along with the construction, all the adjacency relations involved in the groups with clear inner orders would be included in  $G_t$ . For the unclear ones, the corresponding elements in  $E$  will be specified during the operations in the third phase based on  $O$ .

To make it clearer, we explain this assigning process using the simple example shown on the left-hand side of Figure 5.9 (a). In this instance,  $G$  is initially specified as six square nodes  $s_{i,j}$  ( $i = 0, 1, 2, j = 0, 1$ ), as illustrated in the right figure of (a). Also, we let  $g(0, i, j) = \{s_{i,j}\}$  initially. Since the inner order of  $p_0 = \{v_1, v_2\}$  remains unclear, in the state  $R_1$  (of size  $1 \times 2$ ) directly after folding  $p_0$ ,  $g(1, 0, 0)$  and  $g(1, 0, 1)$  are specified as  $g(1, 0, 0) = \{g(0, 0, 0), g(0, 1, 0), g(0, 2, 0)\}$  and  $g(1, 0, 1) = \{g(0, 0, 1), g(0, 1, 1), g(0, 2, 1)\}$ , respectively. We use the indication of sets because the inner orders of both  $g(1, 0, 0)$  and  $g(1, 0, 1)$  remain unclear. Then, the top pseudo nodes  $g_t(1, 0, i)$  and the bottom pseudo nodes  $g_b(1, 0, i)$  are attached to represent  $g(1, 0, i)$  with  $i = 0, 1$ , respectively. In the next step,  $p_1 = \{h_1\}$  folds the map to  $R_t = R_2$ . Such a  $p_1$  leads to a state where  $g(1, 0, 1)$  is upper adjacent to  $g(1, 0, 0)$  in  $R_t$  with the assumption that  $s_{0,0}$  faces its front side up. Corresponding to this adjacency relation, an edge is added from  $g_t(1, 0, 0)$  to  $g_b(1, 0, 1)$ .

## (6) The specification of the interdependencies among groups based on $P$

After the assigning of adjacency relations, in this section, we explain how to record the interdependencies among associated groups with unclear inner orders by giving indices to the nodes. In practical foldings, the interdependencies are induced from the simple folds, which means the consistency of the inner orders of these associated groups. The details of the indexing can be described as follows: we assign indices to the  $g(k, x', y')$ s which comprise the same  $g(k+1, x, y)$  with an unclear inner order. We index  $g(k, x', y')$  by  $i$  if it is in the above-introduced rectangle  $l(k, i)$  before folding  $p_k$ . Firstly, an arbitrary feasible

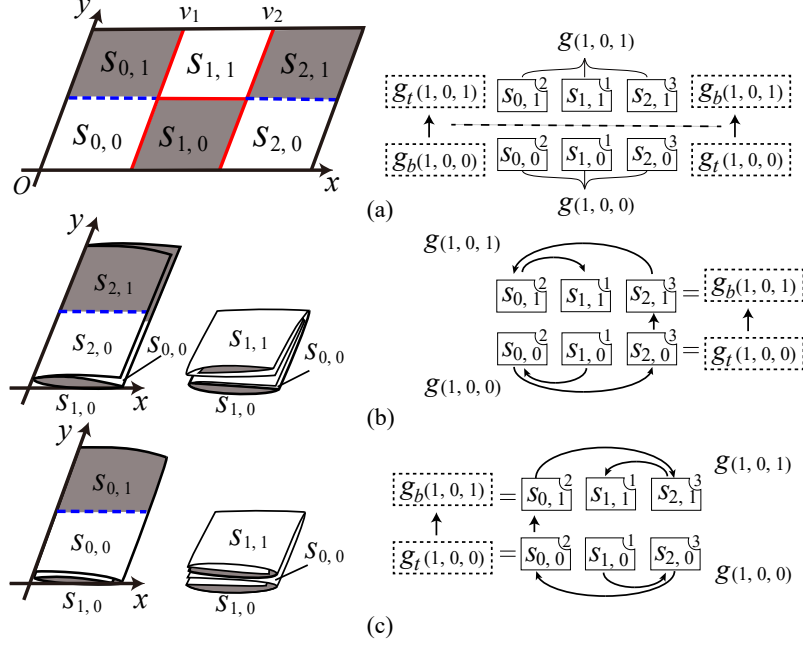


Figure 5.9: The interdependencies among groups. The numbers on the upper-right corner of nodes indicate their indices.

order of  $p_k$  is chosen by the method in Chapter 14.2 of [23]. Because they only discussed the one-dimensional case. We here view each rectangle divided by the crease lines in  $p_k$  as a single segment along the axis perpendicular to the crease lines. A feasible overlapping order of these segments (rectangles) can be obtained. We index these rectangles from bottom to top to obtain  $L_k = \{l(k, i) | i = 0, 1, \dots, |p_k|\}$ . According to the location where  $g(k, x', y')$  is folded to by  $p_k$ , the index of any  $g(k, x', y')$  can be specified as an  $i$ .

As an example, we give the concrete operation for the instance illustrated in Figure 5.9. The figure on the right side of (b) illustrates the indexing in  $G$  when the partly flat-folded state follows the illustration given on the left side of (b). In such a state,  $p_0$  is sequenced as  $(v_1, v_2)$ . Then, the elements in  $g(1, 0, 0)$  are ordered by an ordered set  $(g(0, 1, 0), g(0, 0, 0), g(0, 2, 0))$ . The elements in  $g(1, 0, 1)$  are ordered by an ordered set  $(g(0, 1, 1), g(0, 0, 1), g(0, 2, 1))$ . In both groups, the ordinal number of an element is just recorded as its index.

With these indices, even if the folds in  $p_k$  can be ordered differently, as long as  $R_t$  is reachable by simple folds, the adjacency relations among associated groups are consistent with each other. In the third phase, when we try to specify the unassigned edges according to  $O$ , these indices will play a critical role.

## (7) The completion of adjacency relation graph by traversing $O$

In this section, we introduce the operations supposed to be finished in the third phase, namely, how to complete the adjacency relation graph  $G$  according to  $O$  and the interdependencies specified following the method introduced in the last section. As mentioned before, all the edges remaining unspecified correspond to the unclear inner orders of groups. In the third phase,  $G$  is concluded from  $G_t$  by concluding the edge assigning

corresponding to every group with an unclear inner order. And the check for the validity of  $O$  is also implemented at the same time. Along with the construction of  $G$ , the pseudo nodes  $g_t(k+1, x, y)$  and  $g_b(k+1, x, y)$  would be removed as soon as the inner order of  $g(k+1, x, y)$  becomes clear. Moreover, the edges incident to  $g_t(k+1, x, y)$  and  $g_b(k+1, x, y)$  are respectively replaced by the edges incident to its uppermost and lowermost elements. Therefore, finally, a Hamiltonian path with exactly  $m \times n - 1$  directed edges should be induced in the final state of  $G$ .

This step would finish the decision on the validity of  $O$ , and thus a traverse of  $O$  is necessary. For every adjacent pair of squares  $(s_{u,v}, s_{u',v'})$  in  $O$ , if their adjacency relation is decided valid, both the edge representing their adjacency relation and the edges representing the adjacency relations of their associated groups should be assigned. And we classify the state of  $G$  as two possibilities. Possibility 1 corresponds to the case where the adjacency relation between  $s_{u,v}$  and  $s_{u',v'}$  is already decided and specified in  $G$ . It is further specified as two sub-possibilities 1a and 1b as follows.

Possibility 1a: there already exists a corresponding directed edge between  $s_{u,v}$  and  $s_{u',v'}$ . In such a case, the edge should have been assigned according to another adjacency relation  $(s_{a,b}, s_{a',b'})$  and the interdependencies between them and  $s_{a,b}, s_{a',b'}$ . We do not have to apply other operations

Possibility 1b: a directed edge is already assigned from  $s_{u,v}$  to a group which includes  $s_{u',v'}$ . The inner order of the group is still unclear at this point. In such a case, we further specify  $s_{u',v'}$  as the lowermost element of this group.

These two possibilities cover all the cases in Possibility 1 because in the edge assigning process, before  $(s_{u,v}, s_{u',v'})$  in  $O$  is traversed, we have already specified the uppermost element of the group including  $s_{u,v}$  as  $s_{u,v}$ .

Next, Possibility 2 corresponds to the case that the adjacency relation between  $s_{u,v}$  and  $s_{u',v'}$  remains undecided yet. We have to assign the edge between them and all the other edges of the associated adjacency relations. The latter should be found out according to the interdependencies among groups. As introduced before, the adjacency relation between a pair of  $g(k, x', y')$ s of the same  $g(k+1, x, y)$  decides both the MV assignment and the ordinal number of a certain crease line in  $p_k$  uniquely. Thus, the adjacency relation between corresponding rectangles  $l(k, i_1)$  and  $l(k, i_2)$  would become clear, too. In other words, the adjacency relation between  $s_{u,v}$  and  $s_{u',v'}$  decides the adjacency relations between all the pairs of  $g(k, a', b')$ s in  $l(k, i_1)$  and  $l(k, i_2)$  belonging to the same  $g(k+1, a, b)$ .

During the indexing for the groups  $G$  in the second phase, the interdependencies among the unclear adjacency relations are recorded by the indices. Therefore, every adjacency relation in  $O$ , which is unspecified in  $G$  can be viewed as new information involving a set of adjacency relations. We add edges both between the nodes corresponding to this pair of  $g(k, x', y')$ s and all their interdependent pairs  $g(k, a, b)$ s. Their directions are specified with respect to the sides they face up in  $R_t$ . To make it clearer, we give an example at the end of this section.

Such an assigning ensure the interdependent adjacency relations to be represented by the newly added edges in all the associated groups. Whether  $O$  follows the relations added in other groups or not is also checked along the traverse of  $O$ .

Now we give the conditions for an input  $O$  to be valid.

Condition 1.  $O$  is always in accordance with  $G$ ;

Condition 2. the inner order of every group is valid under the context of simple folds.

For Condition 2, whether the inner order follows the simple folds or not can be verified with the method provided in Ref. [9] by viewing the map as a one-dimensional map in the direction perpendicular to the creases at each step. On account of the interdependencies among the groups, each time, we only have to check  $L_k$  as an entire if the inner order of  $g(k, x', y')$  is unclear. Therefore, in fact, only the edge assignment in one group is checked because it represents all its associated groups.

Given by Lemma 5.2.3, we have proved that once an adjacency relation becomes definite, it would not be changed during the after folding. Therefore, a valid  $R_t$  should respect all the adjacency relations indicated by  $G$  step by step. If  $O$  is invalid, it would definitely be checked out through our specification. Compared to the valid case in Possibility 1, some adjacency relations would be pre-assigned in  $G$  but not in accordance with  $O$ . In addition, since our handlings strictly follow the definition of simple folds with no other restriction, any valid order corresponding to a reasonable  $G$  is accepted by our method. Hence, the correctness of our method follows.

A set of concrete operations applied to the instance in Figure 5.9 are given as an example. Operations in the first two phases are introduced in the last two paragraphs, and now, in the third phase, we are going to simulate the check. Two possible foldings and their corresponding edge assignments of  $G$  are illustrated in (b) and (c). Specifically, (b) shows the case when the first pair traversed in  $O$  is  $(s_{1,0}, s_{0,0})$ . This means  $p_0$  is sequenced as  $(v_1, v_2)$ . First, an edge is assigned from  $s_{1,0}$  to  $s_{0,0}$ , namely the nodes indexed 1 and 2 in  $g(1, 0, 0)$ . Then, we assign an edge to their interdependent nodes in  $g(1, 0, 1)$  (with the same indices) from  $s_{0,1}$  to  $s_{1,1}$  (the direction is reversed because of the parity). The next pair traversed in  $O$  should be  $(s_{0,0}, s_{2,0})$  because  $g(1, 0, 0)$  is supposed to be folded under  $g(1, 0, 1)$ . Otherwise,  $O$  is invalid. The corresponding edge assignment is from  $s_{0,0}$  (indexed 2) to  $s_{2,0}$  (indexed 3) in  $g(1, 0, 0)$  and from  $s_{2,1}$  (indexed 3) to  $s_{0,1}$  (indexed 2) in  $g(1, 0, 1)$ . In addition, as  $s_{1,0}$  and  $s_{2,0}$  are clarified as the lowermost and the uppermost elements of  $g(1, 0, 0)$ , the edges incident to  $g_t(1, 0, 0)$  and  $g_b(1, 0, 0)$  are reassigned to  $s_{2,0}$  and  $s_{1,0}$ , respectively. Then  $g_t(1, 0, 0)$  and  $g_b(1, 0, 0)$  are removed from  $G$ . At this point, all the edges in  $E$  are assigned. Then we traverse the remaining elements in  $O$  to check if their adjacency relations follow  $G$ . Another possible folding is when the first pair traversed in  $O$  is  $(s_{1,0}, s_{2,0})$ , corresponding to the case that  $p_0$  is sequenced as  $(v_2, v_1)$  illustrated in (c). The edges assigned to the nodes in  $g(1, 0, 0)$  include the one between  $(s_{1,0}, s_{2,0})$  and the other between  $(s_{2,0}, s_{0,0})$ . At the same time, the edges incident to the nodes with the same indices in  $g(1, 0, 1)$  are assigned. With the directions conversed, the edges assigned to the nodes in  $g(1, 0, 1)$  are specified as one between  $(s_{0,1}, s_{2,1})$  and one between  $(s_{2,1}, s_{1,1})$ . The edges reassignment is shown on the right side of (b) and (c).

## 5.2.4 Algorithm description

Algorithm 4 describes the mentioned process. We use the coordinates of the squares in  $R_0$  to represent them in the input. In this description, the  $i$ th member in  $O$  is indicated by  $O_i$ . From the flow of the algorithm, we can conclude that it can be accomplished in polynomial time.

Next, we discuss that Algorithm 4 can be realized in  $O(mn)$  time. Note that to achieve a linear-time realization, the coordinates of  $s_{0,0}$  are not standardized to  $(0, 0)$  in

each step anymore.

For the first two steps, the  $MV$  assignment can be obtained by traversing  $O$  once in  $O(mn)$  time, and the computation of the incomplete order  $P$  for simple folds costs  $O(mn)$  time by analyzing the labels of all the creases only once (the result in Ref. [23]). Next, we consider the time complexity of the three phases introduced in Section 4.3. Since every group represents the squares positioned the same when folding a certain  $p_k$ , they can be figured out along with the traverse of  $P$  and cost the same time as simple folding a map, i.e.,  $O(mn)$  time referencing the method in Ref. [23]. Furthermore, this process can simultaneously figure out whether the inner orders of the groups are clear or not. We now discuss the time complexity when constructing  $G$ . When representing the groups with unclear inner orders, there would be no more than  $2 \times m \times n$  pseudo nodes added, and thus there are  $O(mn)$  nodes in any state of  $G$ . The edge-assigning of adjacency relations in the first phase can be implemented along with the traverse of  $P$  and involves no more than  $O(mn)$  single edge additions. Thus, this phase costs  $O(mn)$  time using a simple BFS algorithm. In the second phase, the indexing of elements of groups with unclear inner orders essentially equals to arranging a reasonable order for the crease lines in  $p_k$ s, which cost the same time as arranging  $P$ , i.e.,  $O(mn)$  time.

In the third phase, for all the adjacency relations in  $O$ , whether they are in accordance with  $G$  or not can be concluded by traversing  $O$  once, which costs  $O(mn)$  time. The construction of  $G$  is also completed during this traversing. There remains only the time consumption of checking the edge assignments in groups with unclear inner orders. The orders of these groups are clarified by the overlapping orders in  $O$  and are required to be checked for the simple-foldability. This check is applied to the corresponding  $L_k$ s by seeing them as one-dimensional maps and using the method in Ref. [97]. According to their result, the process costs linear time to the factor of the number of elements. Thus, the total consumption of all these kinds of checks during the whole process costs  $O(m+n)$  time since there are  $m + n - 2$  crease lines.

In conclusion, the realization of Algorithm 4 costs  $O(mn)$  time.



**Input** : A sequence  $O$  of the ordering of layers from bottom to top // indicating  $m \times n$  numbered squares

**Output**: A boolean value // the validity of  $O$  under the context of simple folds

**Begin**

```

  initialization
   $P \leftarrow \emptyset$  // the incomplete order sequence
   $G \leftarrow$  A directed graph with  $m \times n$  nodes
  Specify the MV assignment by traversing  $O$ 
  //  $O(mn)$ 
  if the MV assignment is not locally flat-foldable then
    | return false // The map cannot be flat-folded.  $O(mn)$ 
  end
  Construct  $P$  as an incomplete order by referring the MV assignment //  $O(mn)$ 
  if the construction failed then
    | return false // The map cannot be folded by simple folds.
  end
  foreach  $p_k$  in  $P$  do
    // Partly specify the directed graph  $G$  ( $O(mn)$ )
    if  $p_k$  uniquely decides the adjacency relation between two squares or groups already exist then
      | Add edges representing the corresponding adjacency relations after folding  $p_k$  to  $G$ 
    else
      | Make group nodes based on their positions after the folding according to  $p_k$  // The nodes correspond to the sets  $g(k+1, x, y)$ s for the admissible  $x$ s and  $y$ s, and this operation is done only if the order of  $p_k$  is unclear
      | For each group, add a top pseudo node and a bottom pseudo node to  $G$ 
      | Index the elements of each group from bottom to top according to an arbitrary flat-folded state after folding  $p_k$ 
    end
  end

```

**end**

```

Delete the isolated pseudo nodes in  $G$ 
for  $i = 0$  to  $m \times n - 1$  do
  if  $O_i$  is not in any group  $S$  then
    if  $O_{i+1}$  is in a group  $g$  and the next adjacent node of  $O_i$  is the bottom
      pseudo node of  $g$  then
        assign all the incident edges of the bottom pseudo node to  $O_{i+1}$ 
        and then remove this pseudo node from  $G$ 
      next  $i$ 
    else
      if the next adjacent node of  $O_i$  is  $O_{i+1}$  then
        | next  $i$ 
      else
        | return false
      end
    end
  else
    if  $O_{i+1}$  and  $O_i$  are in the same group then
      if  $O_i$  is not adjacent to  $O_{i+1}$  in  $G$  then
        assign the edge from  $O_i$  to  $O_{i+1}$  and the edges between the
        same-indexed nodes in other groups // the direction is decided
        by the side the squares face up
      next  $i$ 
      end
    else
      if  $O_{i+1}$  is in another group then
        // It means that all the edges in the group including  $O_i$  are
        specified since squares of the same group are adjacent in the
        final order; otherwise  $O$  is invalid
        if  $O_i$  is in a group whose arrangement in  $G$  is invalid under the
          context of simple folds then
            | return false // checked with the method in Ref. [97]
            |  $O(\max\{m, n\})$ 
          end
          assign all the incident edges of the top pseudo node of the
          group containing  $O_i$  and then remove this pseudo node from  $G$ 
          assign all the incident edges of the bottom pseudo node of the
          group containing  $O_{i+1}$  to  $O_{i+1}$ , and then remove this pseudo
          node from  $G$ 
        next  $i$ 
      else
        | return false //  $O_{i+1}$  is not in any group
      end
    end
  end
end
return true
end

```

**Algorithm 4:** A linear-time algorithm for VTOS.

## 5.3 Valid Boundary Overlapping Orders with Simple Folds

In the input of the original map folding problem, the MV assignment actually plays the role of a partial order on the set of the squares. Correspondingly, we intend to figure out how partial orders influence the computational complexity in the decision problem of valid overlapping orders. In the last research about the valid total overlapping, we provided an  $O(mn)$  time algorithm to solve the decision problem about the validity of overlapping orders given on all the squares of an  $m \times n$  map in the simple fold model. However, referencing the general intractable results about the valid orders in accordance with input orders given on subsets (special or general partial orders), the decision would become much more intricate when the input order is given onto a subset of the squares. To figure out if there exists any tractable result when the context is map folding and with such inputs, we consider the boundary overlapping orders.

It is clear that this problem in the general fold model can always be solved by excluding self-penetrations as introduced in Section 3.1. In the simple fold model, the first topic we studied is the partial order with the most abundant information, the boundary overlapping order. As illustrated in Figure 1.5, a boundary overlapping order  $O$  is an order given on only the squares aligning on the boundary. In the valid boundary overlapping order problem, the input is a boundary overlapping order of a map of size  $m \times n$ ; The output is the decision on whether the simple fold model can reach a valid final flat-folded state where the input order is consistent with the indexes of the squares from bottom to top. Our conclusion is that we can decide the validity of a given boundary overlapping order and find a feasible way to fold it by simple folds for a given valid overlapping order in  $O(m+n)$  time. The way to fold the map is called a *whole simple folding sequence* and represented by a sequence of simple folds.

On the other hand, we also considered the enumeration of valid whole simple folding sequences when the input is a valid boundary overlapping order. We proposed the algorithm to enumerate all the other feasible folding sequences for the same input. With an  $O(m+n)$  time precomputing, our algorithm enumerates each folded state in  $O(\max\{m, n\}mn)$  time delay.

### 5.3.1 Preliminaries

For clarity, the notations and definitions used in this paper are divided into three parts.

#### (1) Notations of map

The basic definitions and notations of the map, including vertices, creases, crease lines, and the coordinates, are the same as the definitions and notations we used for the last problem, the valid total overlapping order problem. Each vertex in  $M_{m,n}$  has degree four, and it is known that a vertex is locally flat-foldable if and only if exactly three of its incident creases are assigned the same [53] [57] [58]. The notations for such a map  $M_{m,n}$  are illustrated in Figure 5.10.  $M_{m,n}$  is assumed to be at the first quadrant of a right-handed Cartesian coordinate whose unit length is equal to the side length of the squares. The bottom row of  $M_{m,n}$  is aligned with the  $x$ -axis from the origin. We use

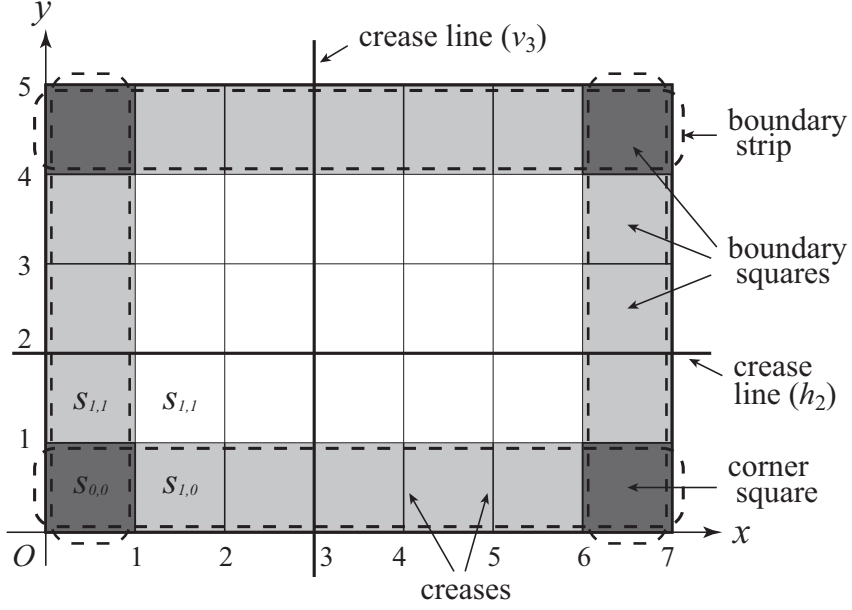


Figure 5.10: The notations and the coordinate system of the map.

$s_{i,j}$  ( $0 \leq i < m, 0 \leq j < n$ ) to refer to the square whose lower-left vertex is located at  $(i, j)$  before any fold. Specifically, our research concerns the squares aligning on the boundary of  $M_{m,n}$  (shaded), i.e., the set  $\{s_{i,j} \mid i \in \{0, m-1\} \text{ or } j \in \{0, n-1\}\}$ . We call these squares *boundary squares*. A *boundary strip* is a set of boundary squares aligning on the same side. *Corner squares* are the four squares initially located at the corner of the map, namely,  $s_{0,0}, s_{m-1,0}, s_{0,n-1}, s_{m-1,n-1}$ . Without loss of generality,  $s_{0,0}$  is supposed to be fixed to  $(0, 0)$  and always faces the front up during the whole folding process.

## (2) Definitions about folding

As mentioned before, Arkin et al. defined two kinds of simple folding operations, *end-folds* and *crimps*, for a given MV assignment [9]. We here redescribe their definitions because of their importance to this research. In a 1D map, as illustrated in Figure 2.1, an end-fold is a fold at either the first or the last crease point. The interval between the last crease point and its corresponding end of the map is not longer than its neighbor interval. A crimp is a fold along a pair of adjacent crease points labeled “MV” or “VM”. The length of the interval between the two crease points is a local minimum value. By such definitions, once an MV assignment is given, the end-folds and crimps are uniquely decided. We use the same operation to indicate our folding process. However, we always neglect the creases remaining unfolded at each step, which means that at each step, the crimps and end-folds are redefined using the current crease lines to be folded. By mathematical induction, it can be clarified that any valid partly folded state is reachable by crimps and end-folds [23].

Both crimps and end-folds can be considered as conditional simple folds. For a valid input order, we intend to find a folding process  $F$  composed of crimps and end-folds such that during the folding, once the surfaces of two squares touch each other, the two squares would never be separated by the subsequent folding operations. By treating the touching

squares as a single square, the size of the map would be reduced correspondingly. This non-separating property is crucial in our approach. Furthermore, it forces the crimps and end-folds in  $F$  to be applied to all the layers where the corresponding crease lines exist. In the following, we will give the interchangeable conditions of simple folds so that other feasible folding processes, which correspond to the same boundary order while dissatisfies this non-separating property, can be produced from  $F$ .

On the other hand, we call the unconditional simple folds along single lines *general simple folds* to distinguish them. Compared to crimps and end-folds, general simple folds have no restriction on the lengths of intervals. In the following, a *simple fold* refers to either a crimp or an end-fold unless expressly stated otherwise.

We define a *simple folding sequence* as a folding sequence composed of crimps and end-folds, which consecutively reconfigure the map from the initial(or totally unfolded) state of the map to a final flat state of size  $1 \times 1$  is called a *whole simple folding sequence*. Every partly flat-folded state during the folding is in the shape of a rectangle and thus can be viewed as a smaller new map.

The sequence of partly flat-folded states restricted on the set of boundary squares is indicated by  $R = (R_0, R_1, R_2, \dots, R_t)$ , where  $R_t$  represents the final folded state of size  $1 \times 1$ . We define the set  $\mathcal{R}$  of all the possible partly flat-folded states. Then, any end-fold or crimp can be described as a mapping  $f : \mathcal{R} \rightarrow \mathcal{R}$ . Specifically, in a given sequence, a mapping from the state  $R_{i-1}$  to  $R_i$  is denoted by  $f_i$  where  $f_i(R_{i-1}) = R_i$ . For convenience, we indicate such a  $f_i$  when it is restricted on the singleton  $\{R_{i-1}\}$  by the set of crimps and end-folds (both can be indicated by some  $h_a$ s and  $v_b$ s) involved in the fold, i.e.,  $f_i = \{h_a$ s and  $v_b$ s of the crimps and end-folds  $f_i$  involves $\}$ . We say a pair of consecutive folds  $f_i$  and  $f_{i+1}$  are *interchangeable* if  $f_i \circ f_{i+1}(R_{i-1}) = f_{i+1} \circ f_i(R_{i-1}) = R_{i+1}$ , where  $\circ$  means the composite of mappings. A folding sequence from  $R_i$  to  $R_j$  with  $i < j$  is denoted by  $(f_{i+1}, f_{i+2}, \dots, f_j)$  and corresponds to  $f_j \circ f_{j-1} \circ \dots \circ f_{i+1}$  mapping  $R_i$  to  $R_j$ . Then, a whole simple folding sequence  $F$  is abstracted as the composite of all the  $f_i$ s, where  $1 \leq i \leq t$ . That is,  $R_0$  is mapped to  $R_t$  by  $F$ . Let  $\mathcal{F}$  be the set of feasible whole simple folding sequences;  $\mathcal{F}$  is the solution space of the decision problem on the validity of the input order.

From another viewpoint,  $F$  can be partitioned into some sub-sequences each consisting only parallel folds. We denote these sub-sequences by  $P = (p_1, p_2, \dots, p_u)$  with  $1 \leq u \leq t$  which satisfies that (1) each  $p_k$  is a maximal set of some parallel  $f_i$ s which are perpendicular to those composing  $p_{k+1}$  and (2) the crease lines in  $p_k$  are folded directly after  $p_{k-1}$  and before  $p_{k+1}$ . The sub-sequences in  $P$  are uniquely decided by  $F$  and vice versa. Then, we have Observation 5.3.1.

**Observation 5.3.1.** *The elements in  $p_0, p_2, p_4, \dots, p_{2i}$  are parallel folds; The elements in  $p_1, p_3, p_5, \dots, p_{2i+1}$  are parallel folds, which are perpendicular to the elements in  $p_0, p_2, p_4, \dots, p_{2i}$ .*

Without loss of generality, we assume that all elements  $p_{2i}$  are parallel to  $x$ -axis, and all  $p_{2i+1}$  are parallel to  $y$ -axis.

The term *neighbor squares* and *adjacent* follows the definitions before. Namely, a pair of neighbor squares refers to a pair  $s_{a,b}, s_{a+1,b}$  or a pair  $s_{a,b}, s_{a,b+1}$  in the initial state. Also, when a partly flat-folded state  $R_i$  is viewed as a new map, two squares sharing a crease of the new map are called *neighbor squares in  $R_i$* . In any partly or totally flat-folded state,

we say a pair of squares  $s_i$  and  $s_j$  whose surfaces touching each other are adjacent, and denoted by  $s_i \leftrightarrow s_j$ . Again, the closure of  $\leftrightarrow$  by  $\leftrightarrow^*$  can then be constructed as follows: (1)  $s_i \leftrightarrow^* s_i$ ; (2)  $s_i \leftrightarrow^* s_k$  if  $s_i \leftrightarrow^* s_j$  and  $s_j \leftrightarrow s_k$ .

For a square  $s_j$  in a folded state  $R_i$ , we can consider the closure of squares adjacent to  $s_j$  as the set of squares  $\{s_l | s_j \leftrightarrow^* s_l\}$ . As mentioned before, adjacent squares would never become non-adjacent again in  $F$ . Thus, the closure of  $s_j$  in  $R_i$  would be a subset of the closure of  $s_j$  in  $R_k$  for any  $k > i$ . Let  $O_i(a, b)$  denote the set of squares  $\{s_l | s_{a,b} \leftrightarrow^* s_l\}$  in the state  $R_i$ . For a given  $R_i$ , when the ordering of the squares in the set  $O_i(a, b) = \{s_1, \dots, s_l\}$  is fixed, we sometimes use an ordered set  $(s_1, s_2, \dots, s_l)$  such that  $s_j \leftrightarrow s_{j+1}$  to describe the order. The ordering of any two neighbor squares in  $R_t$  is decided by the MV assignment [80].

### (3) Definitions in this problem

We use a *valid overlapping order* to refer to a feasible overlapping order  $O_t(0, 0)$  of the map in the simple fold model. A *valid boundary overlapping order* is a total valid overlapping order given on the set of boundary squares of  $M_{m,n}$ .

Our topic is the validity of the boundary overlapping order of  $m \times n$  maps. The problem is described as follows. Given an order  $O$  of the boundary squares, is  $O$  a valid boundary overlapping order in the simple fold model? Also, when  $O$  is valid, we ask which whole simple folding sequence would fold  $M_{m,n}$  to the corresponding flat-folded state of  $O$ .

We pose our conclusion as Theorem 5.3.2.

**Theorem 5.3.2.** *There exists an  $O(m+n)$  time algorithm that decides the validity of the boundary overlapping order  $O$  in the simple fold model and gives a feasible whole simple folding sequence  $F$  when  $O$  is decided to be valid.*

For the corresponding enumeration problem, i.e., the enumeration of valid whole simple folding sequences corresponding to a valid input  $O$ , we have Theorem 5.3.3.

**Theorem 5.3.3.** *There is an enumeration algorithm for valid whole simple folding sequences for a given overlapping order of the boundary squares. With an  $O(m+n)$  time precomputing, it enumerates each folded state in  $O(\max\{m, n\}mn)$  time delay.*

## 5.3.2 Outline

In this research, we concern about the boundary overlapping order in the simple fold model. Two problems are considered, including (1) the decision on the validity of a given order of the boundary squares in the simple fold model and (2) the enumeration of valid whole simple folding sequences when the answer of (1) is positive.

The method to solve (1) is briefly summarized as follows. First, the MV assignment on every crease shared by a pair of adjacent boundary squares is computed by traversing  $O$ . Then, as mentioned before,  $F$  is obtained by computing the sub-sequences of parallel folds in  $F$ . These sub-sequences are represented by  $P = (p_1, p_2, \dots, p_u)$ . If we consider every boundary strip as a 1D map,  $p_k$  can correspondingly be viewed as a simple folding sequence that folds two parallel boundary strips in the same way. Therefore, we decide the simple folds in each  $p_k$  by finding the same folds on two horizontal and two vertical

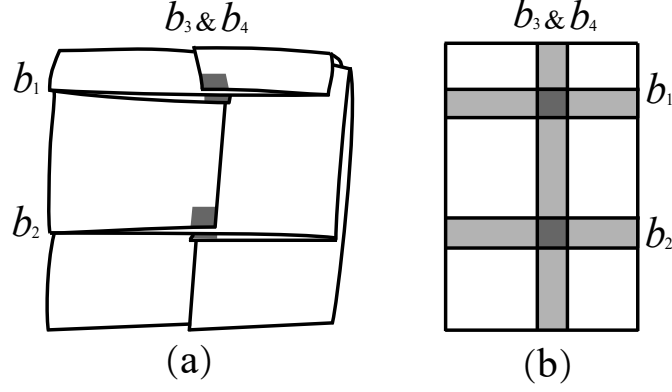


Figure 5.11: An instance with two parallel boundary strips overlapping each other. (a) shows the state of the map. (b) is the top view.

boundary strips, respectively. In this way, we can obtain a whole simple folding sequence  $F$  for a valid  $O$ .

However, since there may exist other whole simple folding sequences inducing the same boundary overlapping order with  $F$ , we have to give the proof that if no  $F$  can be obtained with our method, then there exists no valid whole simple folding sequence leading to the given order. That means  $F$  can represent all the whole simple folding sequences corresponding to the same boundary overlapping order. We prove this by proving that all the other whole simple folding sequences can be achieved from  $F$ . The strategy is considering the interchangeable folds in  $F$  so that any other valid whole simple folding sequences can be obtained by interchanging these folds in  $F$ . By Cayley's theorem [12], all the valid whole simple folding sequences can be obtained by interchanging pairs of the consecutive interchangeable folds from a valid whole simple folding sequence. Therefore, it is sufficient to consider only the interchangeability between consecutive simple folds.

Concretely, the proof is concluded by two parts. In the first part, we analyze the condition that makes consecutive folds interchangeable. In the second part, we assign an equivalence relation on  $\mathcal{F}$  based on the interchangeable condition. If two whole simple folding sequences  $F_1$  and  $F_2$  are equivalent,  $F_1$  and  $F_2$  will lead to the same boundary overlapping order. In this way,  $F$  uniquely and exactly corresponds to an equivalence class, meaning that its existence represents the existence of all the other foldings leading to the same boundary overlapping order. Then, deciding the validity of  $O$  is turned into deciding the existence of  $F$ .

In the following sections, we will firstly explain the concrete manner to determine the inner order of folds in  $p_k$ . Then for a clear explanation, we will propose the interchangeable conditions before the concrete method to decide  $F$ . The interchangeable conditions between a pair of parallel folds and a pair of perpendicular folds will be introduced, respectively.

The case that two parallel boundary strips totally overlap each other (as illustrated in Figure 5.11) is not under the discussion of the interchange. The reason is that such a case can be handled with the same operation for a single  $p_k$ . When parallel boundary strips  $b_3$  and  $b_4$  overlap each other,  $b_1$  and  $b_2$  are supposed to be handled with the operation for two parallel boundary strips, which will be given in the next section. The boundary

strips  $b_3$  and  $b_4$  can be viewed as a single 1D map, which also can be handled with the same operation. The handling of this case can be simply decided by considering the overlapping formed on  $\{b_1, b_2\}$  and  $b_3$  together.

Based on these conditions,  $F$  can thus be chosen to represent all the foldings leading to the same boundary overlapping order. The method to compute  $F$  is given in Section 5.3.3. In Section 5.3.4, we describe the algorithm of the decision problem. In the same section, we also introduce the algorithm to enumerate all the feasible whole simple folding sequences by interchanging the interchangeable simple folds.

### 5.3.3 Interchangeable consecutive folds

The interchangeable conditions of consecutive folds are provided in this section. The discussion contains both consecutive parallel folds and consecutive perpendicular folds. The classes we provide exhaust all the possible consecutive interchangeable cases.

For the former, we concern a certain  $p_k$ , a sequence of parallel simple folds. Firstly we propose the method to find an inner order of  $p_k$  according to the input  $O$ . Then, we present the condition for interchangeable consecutive parallel folds. Note that after each interchange, the update of crimps and end-folds is always considered. In such a manner, all the available combinations of crimps and end-folds in a  $p_k$  can be exhausted. This is for the purpose that other feasible foldings composed of different crimps and end-folds with  $F$  can also be obtained by the interchange. By this exhaustion, when concerning the interchangeability between folds in  $p_k$  and folds in  $p_{k+1}$ , i.e., the perpendicular folds, it is sufficient to consider crimps and end-folds instead of general simple folds. At the final of this section, we will analyze these perpendicular folds and give the condition for consecutive perpendicular folds to be interchangeable.

#### Recognizing consecutive parallel folds

Given two states  $R_i$  and  $R_j$ , we decide the reachability of  $R_j$  from  $R_i$  by only parallel folds. Because of the similarity, we here only discuss the folds in  $p_k$  for an odd  $k$ .

If we view the partly flat-folded state as a map, the creases on the two boundary strips folded by  $p_k$  should have the same assignment (may be different from the initial map if one of them faces the front up and the other faces the back up) and induce the same or reversed overlapping order on the two boundary strips. The consistency of both the assignment and overlap on the two boundary strips can be checked by a parity check of the coordinates [80].

To simplify the explanation, in the following, we discuss the folds on a single boundary strip. We give the description of the two states after a crimp and after an end-fold by referring Figure 5.5. The size of the map before the fold is assumed to be  $1 \times n'$  with its left end located on  $x = 0$ . We denote the coordinates of points  $A$  to  $G$  as  $x_A$  to  $x_G$ , which are all integers. Then, we have  $x_B - x_A > x_C - x_B < x_D - x_C$  for the crimp and  $n' - x_G \leq x_G - x_F$  for the end-fold. The equal symbol is permitted only for end-folds for the convenience of the handling of finding a feasible whole simple folding sequence. The two states are respectively described by the following formulas, where  $o(x, 0)$  indicates the squares folded to the coordinate  $(x, 0)$ , and tuples indicate the order of the squares from bottom to top.



For the crimp, we have

$$o(x, 0) = \begin{cases} (s_{x,0}), & x < 2x_B - x_C \\ (s_{x,0}, s_{2x_B-x-1,0}, s_{x+2x_C-2x_B,0}), & 2x_B - x_C \leq x < x_B \\ (s_{x+2x_C-2x_B,0}), & x_B \leq x < 2x_B + n' - 2x_C \end{cases}$$

and for the end-fold, we have

$$o(x, 0) = \begin{cases} (s_{x,0}), & x < 2x_G - n' \\ (s_{x,0}, s_{2x_G-x-1,0}), & 2x_G - n' \leq x < x_G \end{cases}$$

By repeating the following three steps, it can be determined whether or not  $R_j$  can be reached from  $R_i$  by  $p_k$ . If reachable, the elements and the inner order of  $p_k$  are also decided in this process.

Step 1. Update the 1D map representing a boundary strip only with the creases whose neighbor squares are in the same  $O_j(a, b)$  in  $R_j$ .

Step 2. Find crimps and end-folds on the 1D map by referring the formulas of  $o(x, 0)$ . If no fold exists,  $R_j$  is not reachable.

Step 3. Reduce the map to a new (smaller) map by applying the folding operations found in Step 2.

This process can be done in linear time of the 1D map since it can be done by a standard graph traverse algorithm, e.g., breadth-first search or depth-first search algorithm. An example of this process for a  $1 \times 10$  map is introduced here.

For the conciseness, we use numbers 0 to 9 to label the squares of the  $1 \times 10$  map  $M$  from its left end to its right end. The input order  $O$  is given as: (9, 2, 3, 4, 1, 0, 5, 8, 7, 6). The objective is to determine whether or not  $O$  is a valid overlapping order of  $M$  in the simple fold model, and if it is, a feasible whole simple folding sequence is also desired.

The process of computation is as follows: First, the MV assignment of  $M$  is computed using the method introduced in Ref. [80], as illustrated in Figure 5.12 (a). Then, by a traverse of the map, all the pairs of neighbor squares are recorded. In this instance, they are {2, 3}, {3, 4}, {1, 0}, {8, 7}, {7, 6}. Except for the creases between every pair, all the other creases are removed from the map, which forms a new map as illustrated in Figure 5.12 (b).

Next, by locally considering every single crease and every pair of nearest creases, the candidates of crimps and end-folds are the crimp forming (2, 3, 4), the crimp forming (8, 7, 6), and the end-fold forming (1, 0). For the crimps, referencing the formula provided in Section 4.1, the existence of the adjacent relation  $(s_x, s_{2x_B-x-1}, s_{x+2x_C-2x_B})$  or its reverse should be checked. For (2, 3, 4),  $x=2$ ,  $x_B=3$  and  $x_C=4$ , which holds the formula and thus is decided to be firstly folded. The check for (8, 7, 6) follows the same way. Similarly, the existence of  $(s_x, s_{2x_G-x-1})$  or its reverse should be checked for the end-fold. For (1, 0),  $x=0$  and  $x_G=1$ , which holds the reverse of the formula and indicates that this end-fold is firstly folded. The folded state is shown in Figure 5.12 (b). These three folds are disordered. After folding them, the map is reduced to a new map, as illustrated in Figure 5.12 (c). For convenience, we use the labels of the top layers to indicate the

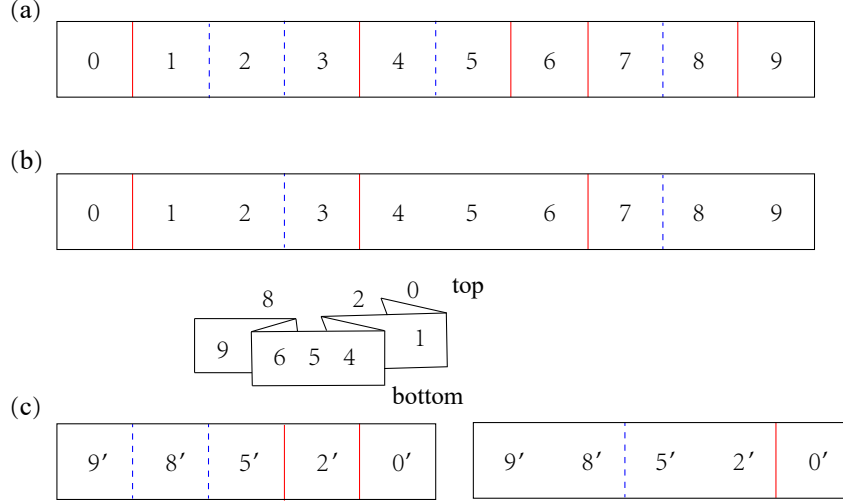


Figure 5.12: Computing the orders of folds an instance 1D map.

squares in the new map.  $O$  is correspondingly updated to  $(9', 2', 0', 5', 8')$ . The MV assignment is changed along with the change of the side that these squares face up.

For the new map, two candidate end folds in the new map are decided as the one corresponding to  $(2', 0')$  and the one corresponding to  $(5', 8')$ .  $(2', 0')$  holds the formula and is decided to be firstly folded in this step while  $(5', 8')$  does not hold and is checked in the new map after folding the end-fold  $(2', 0')$ . It holds in the new map and is then folded. As mentioned in Section 5.2, using the graph structure in the realization can reduce the check for every fold only once.

Following the similar process, we can finally decide the validity of  $O$  and obtain the folding process as: (the creases are indicated by its neighbor squares) ( $\{c_1=(2, 3, 4), c_2=(6, 7, 8), e_1=(0, 1)\}$ ,  $e_2=(1, 2)$ ,  $e_3=(5, 6)$ ,  $e_4=\{(4, 5), (8, 9)\}$ ). The folds in the same brace can be arbitrarily ordered.

### Interchangeable parallel folds

The interchangeable condition of the parallel folds inside  $p_k$  is given in this section. This condition is supposed to be used to produce other feasible foldings leading to the same boundary overlapping order. Assuming that a valid overlapping order  $O$  and one of its valid whole simple folding sequence  $F$  is known, every valid whole simple folding sequence corresponding to the same  $O$  should be obtainable by interchanging some interchangeable folds in  $F$ . By the method provided in the last section, the crease lines in  $p_k$  would be folded as certain crimps and end-folds. However, another valid whole simple folding sequence may have a  $p_k$  with the same crease lines but folded as different crimps and end-folds. Therefore, here we consider interchangeable general simple folds (each along one crease line) to exhaust all the possibilities of valid whole simple folding sequences.

The following lemma gives the necessary and sufficient condition for a pair of parallel folds in a  $p_k$  to be interchangeable.

**Lemma 5.3.4.** *A pair of consecutive parallel folds  $l_a$  and  $l_{a+1}$  are interchangeable if and only if (1) or (2) holds.*

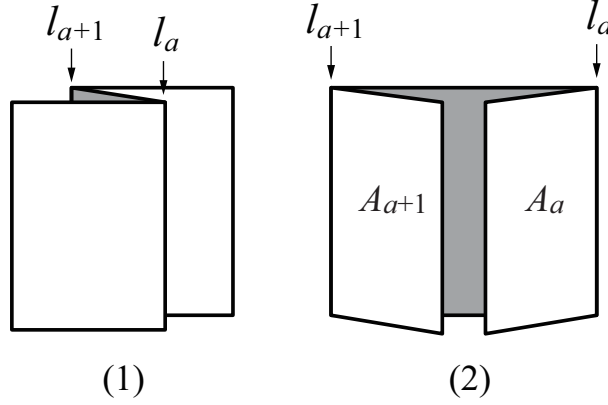


Figure 5.13: Two possible cases for interchangeable parallel folds.

- (1)  $l_a$  and  $l_{a+1}$  are labelled “MV” or “VM”.  
(2) The labels of  $l_a$  and  $l_{a+1}$  are the same, let  $A_a = \{(i, j) | O_a(i, j) \neq O_{a-1}(i, j)\}$ ,  $A_{a+1} = \{(i, j) | O_{a+1}(i, j) \neq O_a(i, j)\}$ , and the intersection of  $A_a$  and  $A_{a+1}$  is empty.

Case (2) means that the overlapping part formed by folding along  $l_a$  does not overlap with the overlapping part formed by folding along  $l_{a+1}$ . Note that since  $l_a$  and  $l_{a+1}$  are general simple folds, they may only affect the overlapping on some but not all the layers. Both cases are illustrated in Figure 5.13. The proof is omitted since the lemma is rather straight forward. We note that if  $A_a$  and  $A_{a+1}$  have overlapping, the order of  $l_a$  and  $l_{a+1}$  is fixed. That means they are not interchangeable.

By this lemma, we can find the interchangeable pairs of folds in  $p_k$ . The interchange may form new crimps and end-folds in  $p_k$ . The computation also costs time linear in the size of the 1D map. Every available order inside  $p_k$  can be obtained by interchanging these pairs finite times.

### Interchangeable perpendicular folds

The interchangeable condition of consecutive perpendicular folds is discussed in this section. For convenience, we say a simple fold  $f_a$  involves a square  $s_{i,j}$  if  $O_a(i, j) \neq O_{a-1}(i, j)$ . By the condition that adjacent squares would never become non-adjacent again in  $F$  in our simple fold model, the overlapping of corner squares indicates the order of horizontal folds and vertical folds. The most crucial factor for the interchangeability of perpendicular folds is whether the folds involve corner squares or not. Based on this factor and the MV assignment around the corner squares, we classify the interchangeable cases into eight classes as illustrated in Figures 5.16, 5.19, 5.20, 5.21, 5.22.

These classes concern partly folded states of the map. In partly folded states, the corner squares may not lay on the four corners and may not maintain their initial relative positions. For example, three possible states of the map are illustrated in the upper row in Figure 5.14, where the back of the map is colored gray. In (a), the relative position of  $s_{0,0}$  and  $s_{m-1,0}$  is changed. In (b), all the corner squares are no longer laying on the corners of the partly flat-folded map. In (c), both the positions of corner squares on the partly flat-folded map and their relative positions are changed. The top views, the

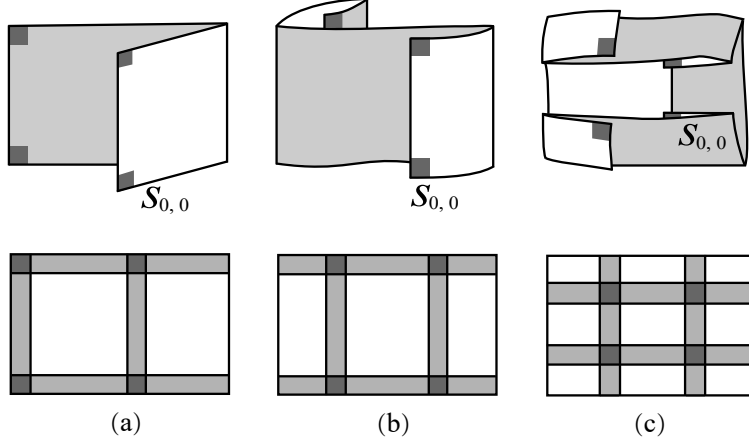


Figure 5.14: The corner squares in boundary areas.

locations of the boundary strips, and the locations of corner squares are illustrated in the bottom row.

For these eight classes, we have Lemma 5.3.5. The proof is given through the following sections.

**Lemma 5.3.5.** (1) *Every case satisfying the condition of one of these eight classes is an interchangeable case;*

(2) *Every interchangeable case can be classified into one of these eight classes.*

Before the proof, we introduce a notion called *merge* to explain that our interchangeable condition keeps the boundary overlapping order. A pair  $(S_a, O_a)$  is used to describe the folded state of a boundary strip, where  $S_a$  is the set containing all the squares on this boundary strip, and  $O_a$  is the total order on  $S_a$ , which represents the overlapping of the squares. For two pair  $(S_a, O_a)$  and  $(S_b, O_b)$ , if a folding sequence induces a new tuple  $(S_a \cup S_b, O_c)$  where  $O_c$  is an order on  $S_a \cup S_b$  (two boundary strips) and satisfies both  $O_a$  and  $O_b$ , then we say this folding sequence *merges* the two boundary strips.

### (i) Proof of Classes 1-3

Firstly, we discuss the first three classes. Four possible relative positions of the corner squares and four end-folds are illustrated in Figure 5.15.

For these consecutive end-folds, with symmetric cases omitted, the only three interchangeable MV assignments on the creases shared by two boundary strips are illustrated in Figure 5.16. The reason is that every two creases around a corner square must be labeled differently to ensure interchangeability. The positions of creases are taken only for convenience. The two creases around each corner square represent all the possible combinations (Figure 5.15) of perpendicular creases. For each class, there exist two possible ways to order the four end-folds. Possible orders and their corresponding MV assignments on the crease lines are given in columns 3 and 4. The order of folds is indicated by 1, 2, 3, 4. Their interchangeable conditions are listed below.

Class 1. At least one of (1, 2) and (3, 4) satisfies the interchangeable conditions the same to the parallel folds.

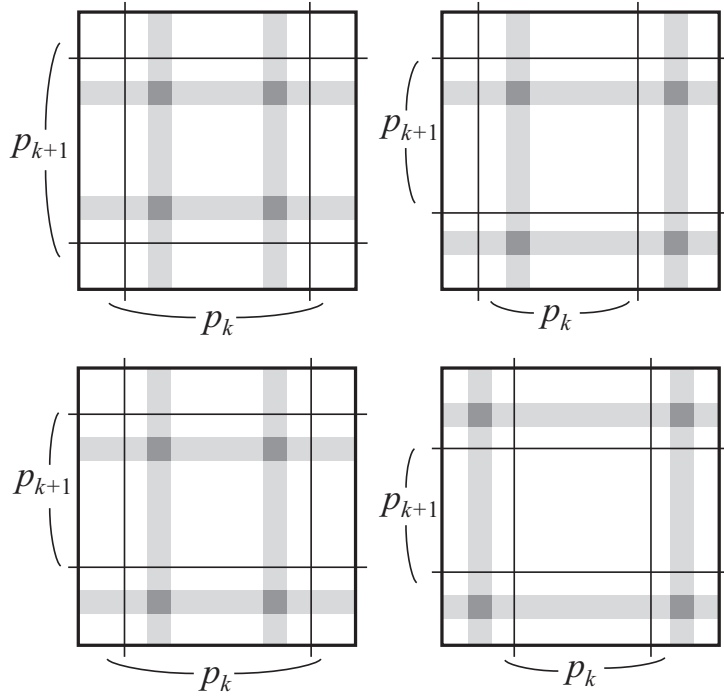


Figure 5.15: Possible relative positions of the corner squares and four end-folds (two are in  $p_k$  and the other two are in  $p_{k+1}$ .)

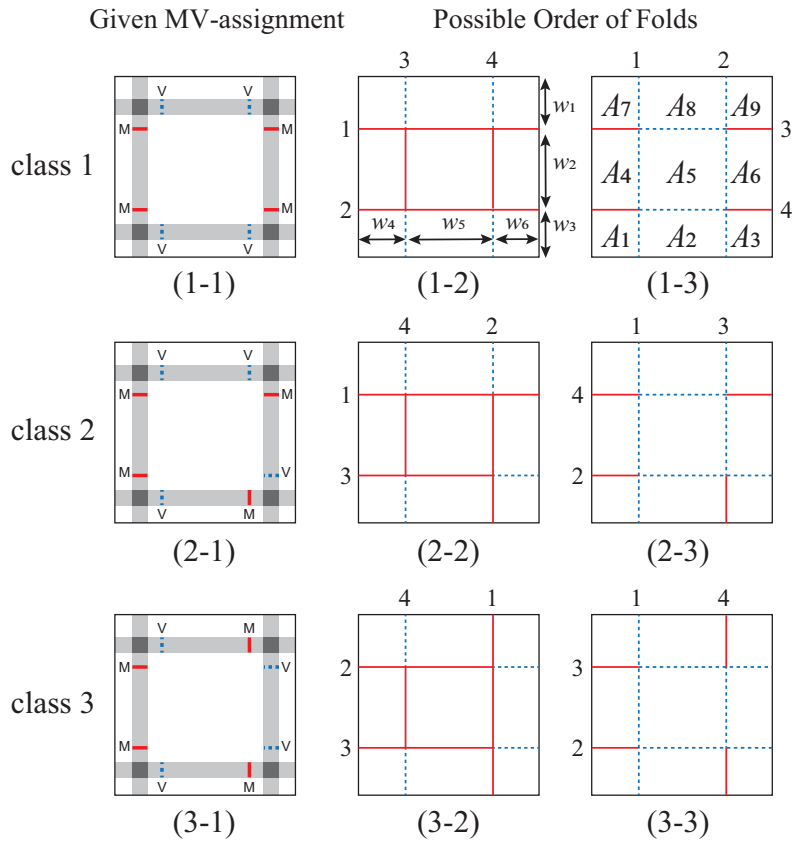


Figure 5.16: The first three cases for four consecutive end-folds.

Class 2. Both (1, 3) and (2, 4) satisfy the interchangeable conditions the same to the parallel folds.

Class 3. 1 and 4 satisfy the interchangeable conditions the same to the parallel folds.

Then we have Lemma 5.3.6.

**Lemma 5.3.6.** *(1) Interchangeability: in Classes 1, 2, 3 illustrated in the first column in Figure 5.16, when the interchangeable conditions are satisfied, the orders of the four end-folds illustrated in the last two columns in Figure 5.16 lead to the same boundary overlapping order. (2) For the partly folded states as illustrated in in the first column in Figure 5.16, the four end-folds cannot be interchanged when the interchangeable conditions are violated.*

*Proof.* (1) It is clear that the interchange of folds is feasible if it (a) does not change the adjacent relation  $\leftrightarrow$  on respective boundary strips, and (b) involves no merging. The certain MV assignment ensures the induced  $O_i(a, b)$  restricted on respective boundary strips to be always the same. We use the notations  $A_1$  to  $A_9$  to indicate different areas. These areas are labeled as in the middle-upper illustration in Figure 5.17. We first consider the case that the four corner squares respectively locate in  $A_1$ ,  $A_3$ ,  $A_7$  and  $A_9$  is explained here (the last case in Figure 5.15).

Taking Class 1 for example, assuming that  $A_1$  always faces the front up. The two folded states corresponding to the orders provided in the second and third columns in Figure 5.16 are shown in the second row in Figure 5.17. The overlap order on respective boundary strips can be described by the 3-tuples:  $(\{A_1, A_7\}, A_4)$ ,  $(A_2, \{A_1, A_3\})$ ,  $(\{A_3, A_9\}, A_6)$  and  $(A_8, \{A_7, A_9\})$ . It is clear that the interchangeable conditions avoid the boundary overlapping orders on respective boundary strips being merged in all the classes. Correspondingly, the order of the four end-folds in Columns 3 and the order in Columns 4 lead to the same boundary overlapping order on respective boundary strips. The other cases are similar and straightforward.

(2) On the contrary, cases as illustrated in Figure 5.18 describe the uninterchangeable cases when the interchangeable conditions are violated. The boundary strips are colored grey in the first figure in the third column. The MV assignments on the boundary strips are the same as the three interchangeable classes, whereas their interchangeable conditions are not satisfied. As an example of the partly flat-folded states, the lower two rows in Figure 5.17 illustrate the folded states induced by the same folds with Class 1. The corresponding folded states are supposed to have overlapping orders of boundary strips, as indicated by the numbers labeled in different areas in Figure 5.18 C and R. Every folding sequence uniquely orders the numbered areas. In other words, each folding sequence merges the 3-tuples, i.e., the overlapping orders on four boundary strips, to an 8-tuple. It means, when corner squares locate at the shadowed areas but not satisfy the interchangeable conditions, the order of consecutive end-folds can be uniquely decided. By this fact, the necessity of the interchangeable conditions in these cases is concluded.  $\square$

## (ii) Proof of Classes 4 and 5

The next two, Classes 4 and 5, are illustrated in Figure 5.19 (1) and (2), respectively. Their description and conditions are detailed as follows:

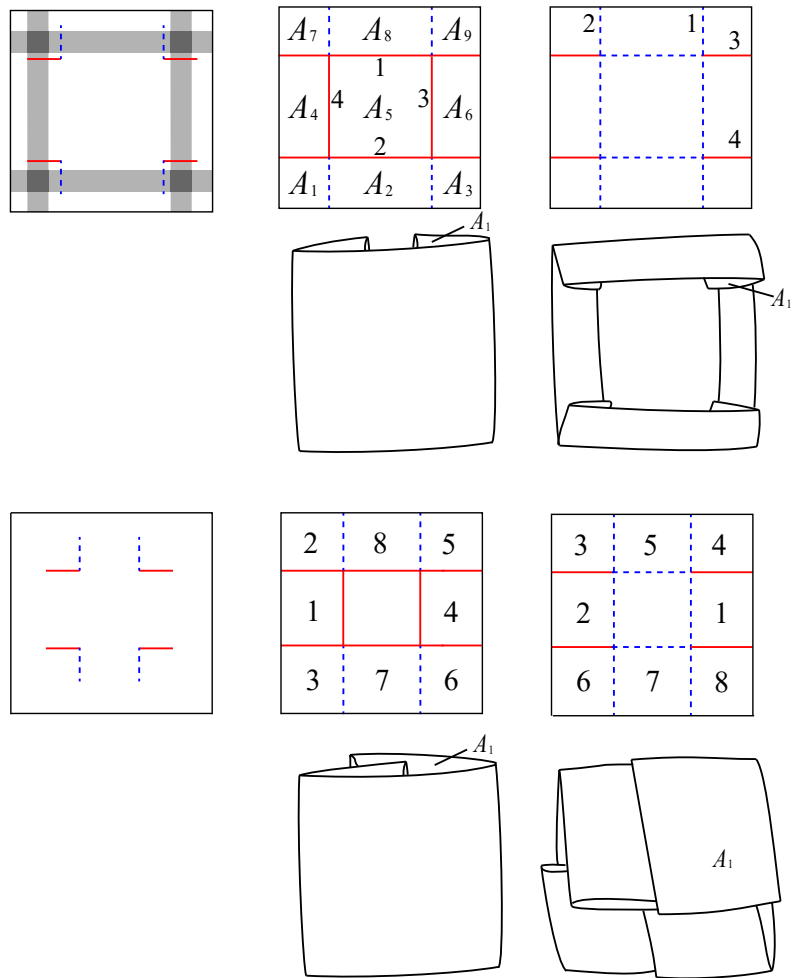


Figure 5.17: An example of the difference between the cases with the interchangeable condition satisfied and dissatisfied.

|       | L                   | C   | R                   |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
|-------|---------------------|---|---------------------|--|-------|--|---|---|---|-------|---|--|---|---|---|---|-------|---|-------|--|-------|--|---|---|---|-------|---|--|---|---|---|---|-------|
| Cases | Given MV-assignment | Overlapping Order 1   | Overlapping Order 2 |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | <table><tr><td colspan="2"><math>e_4</math></td><td><math>e_3</math></td><td></td></tr><tr><td>2</td><td>8</td><td>5</td><td rowspan="2"><math>e_1</math></td></tr><tr><td>1</td><td></td><td>4</td></tr><tr><td>3</td><td>7</td><td>6</td><td><math>e_2</math></td></tr></table> | $e_4$               |  | $e_3$ |  | 2 | 8 | 5 | $e_1$ | 1 |  | 4 | 3 | 7 | 6 | $e_2$ | <table><tr><td colspan="2"><math>e_2</math></td><td><math>e_1</math></td><td></td></tr><tr><td>3</td><td>5</td><td>4</td><td rowspan="2"><math>e_3</math></td></tr><tr><td>2</td><td></td><td>1</td></tr><tr><td>6</td><td>7</td><td>8</td><td><math>e_4</math></td></tr></table> | $e_2$ |  | $e_1$ |  | 3 | 5 | 4 | $e_3$ | 2 |  | 1 | 6 | 7 | 8 | $e_4$ |
| $e_4$ |                     | $e_3$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     | 8                   | 5   | $e_1$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | 4   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 3     | 7                   | 6   | $e_2$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| $e_2$ |                     | $e_1$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 3     | 5                   | 4   | $e_3$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     |                     | 1   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 6     | 7                   | 8   | $e_4$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     |                     | <table><tr><td colspan="2"><math>e_4</math></td><td><math>e_2</math></td><td></td></tr><tr><td>2</td><td>8</td><td>7</td><td rowspan="2"><math>e_1</math></td></tr><tr><td>1</td><td></td><td>6</td></tr><tr><td>3</td><td>4</td><td>5</td><td><math>e_3</math></td></tr></table> | $e_4$               |  | $e_2$ |  | 2 | 8 | 7 | $e_1$ | 1 |  | 6 | 3 | 4 | 5 | $e_3$ | <table><tr><td colspan="2"><math>e_1</math></td><td><math>e_3</math></td><td></td></tr><tr><td>7</td><td>8</td><td>6</td><td rowspan="2"><math>e_4</math></td></tr><tr><td>1</td><td></td><td>5</td></tr><tr><td>2</td><td>3</td><td>4</td><td><math>e_2</math></td></tr></table> | $e_1$ |  | $e_3$ |  | 7 | 8 | 6 | $e_4$ | 1 |  | 5 | 2 | 3 | 4 | $e_2$ |
| $e_4$ |                     | $e_2$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     | 8                   | 7   | $e_1$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | 6   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 3     | 4                   | 5   | $e_3$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| $e_1$ |                     | $e_3$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 7     | 8                   | 6   | $e_4$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | 5   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     | 3                   | 4   | $e_2$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 3     |                     | <table><tr><td colspan="2"><math>e_4</math></td><td><math>e_1</math></td><td></td></tr><tr><td>2</td><td>6</td><td>7</td><td rowspan="2"><math>e_2</math></td></tr><tr><td>1</td><td></td><td>8</td></tr><tr><td>3</td><td>4</td><td>5</td><td><math>e_3</math></td></tr></table> | $e_4$               |  | $e_1$ |  | 2 | 6 | 7 | $e_2$ | 1 |  | 8 | 3 | 4 | 5 | $e_3$ | <table><tr><td colspan="2"><math>e_1</math></td><td><math>e_4</math></td><td></td></tr><tr><td>4</td><td>5</td><td>6</td><td rowspan="2"><math>e_3</math></td></tr><tr><td>1</td><td></td><td>8</td></tr><tr><td>2</td><td>3</td><td>7</td><td><math>e_2</math></td></tr></table> | $e_1$ |  | $e_4$ |  | 4 | 5 | 6 | $e_3$ | 1 |  | 8 | 2 | 3 | 7 | $e_2$ |
| $e_4$ |                     | $e_1$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     | 6                   | 7   | $e_2$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | 8   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 3     | 4                   | 5   | $e_3$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| $e_1$ |                     | $e_4$   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 4     | 5                   | 6   | $e_3$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 1     |                     | 8   |                     |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |
| 2     | 3                   | 7   | $e_2$               |  |       |  |   |   |   |       |   |  |   |   |   |   |       |   |       |  |       |  |   |   |   |       |   |  |   |   |   |   |       |

Figure 5.18: The unique overlapping orders of boundary squares correspond to the cases when corner square has its adjacent two squares belonging to two perpendicular boundary strips.

Class 4.  $(e_1, e_2)$  and  $(e_3, e_4)$  respectively forms pairs of differently labeled parallel end-folds or crimps. Each tuple involves the same two corner squares.  $e_1$  and  $e_3$  are labeled differently.

Class 5.  $(e_1, e_2)$  is a pair of parallel interchangeable end-folds with the same labels.  $(e_3, e_4)$  is a pair of differently labeled parallel end-folds or a crimp.  $e_3$  and  $e_4$  involve the same two corner squares.  $e_1$  and  $e_3$  are labeled differently.

Then, we have the following lemma. Different from Classes 1-3, there is no additional condition for the interchangeability of Classes 4 and 5.

**Lemma 5.3.7. Interchangeability:** *The interchange of the two tuples  $(e_1, e_2)$  and  $(e_3, e_4)$  in Classes 4 and 5 would not affect the boundary overlapping.*

The proof of the interchangeability of Class 5 is omitted since the analysis is the same as for Class 3. The following gives the proof for Class 4.

*Proof.* As illustrated in Figure 5.19 (1), it is clear that the interchange of the two pairs of parallel folds maintains the overlapping relation that the squares on the left boundary strip are always locating over the squares on the lower boundary strip. It means that the interchange causes no merge of these two boundary strips. Also, these folds do not affect the overlapping of the upper-right corner square, indicating that the other two boundary strips would not be merged. Thus, this interchange would not affect the boundary overlapping.  $\square$



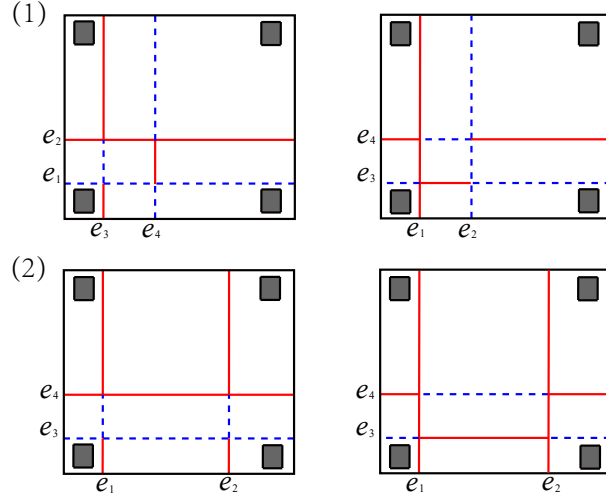


Figure 5.19: Classes 4 and 5: Two perpendicular pairs of parallel end-folds, where one pair of end-folds involves the same pair of corner squares.

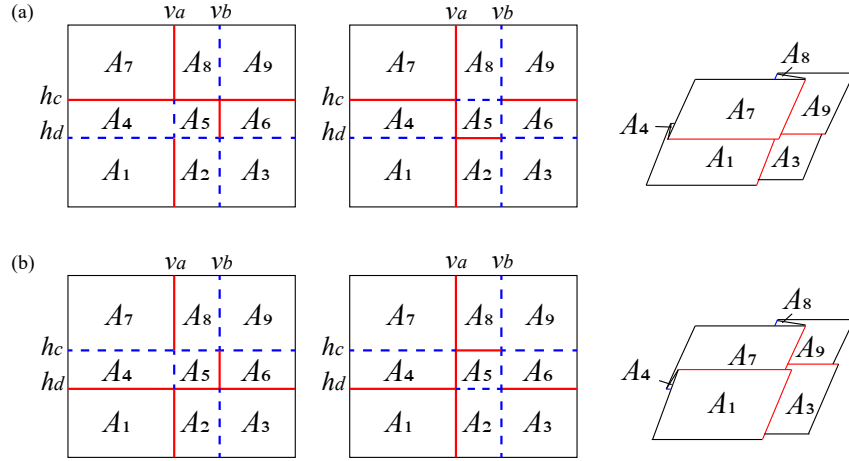


Figure 5.20: Class 6: Two interchangeable crimps.

### (iii) Proof of Classes 6-8

The last three, Classes 6, 7, and 8, concern two consecutive perpendicular folds  $f_{i-1}$  and  $f_i$ . Note that in these classes, we do not consider the cases already discussed. The three classes are classified according to the crimps and the end-folds.

Class 6. Both  $f_{i-1}$  and  $f_i$  are crimps.

Class 7.  $f_{i-1}$  is an end-fold and  $f_i$  is a crimp.

Class 8. Both  $f_{i-1}$  and  $f_i$  are end-folds.

Then we give the interchangeable condition as: at least one of  $f_{i-1}$  and  $f_i$  does not involve corner squares. Figures 5.20, 5.21 and 5.22 illustrate the interchangeable cases of these three classes.

For these three classes, we have Lemma 5.3.8.

**Lemma 5.3.8.** (1) *Interchangeability: in Classes 6, 7 and 8, when the interchangeable condition is satisfied, the interchange of  $f_{i-1}$  and  $f_i$  in Classes 6, 7 and 8 would not affect*

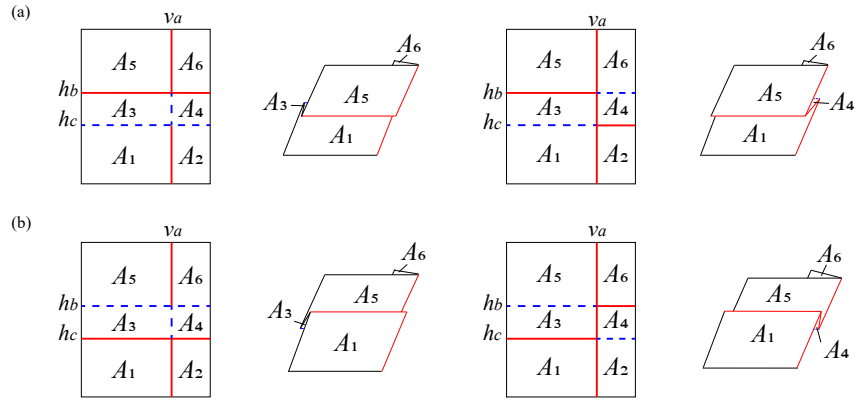


Figure 5.21: Class 7: Two interchangeable crimps and end-fold.

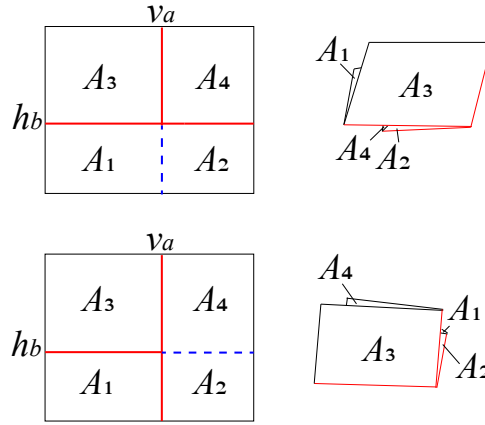


Figure 5.22: Class 8: Two interchangeable end-folds.

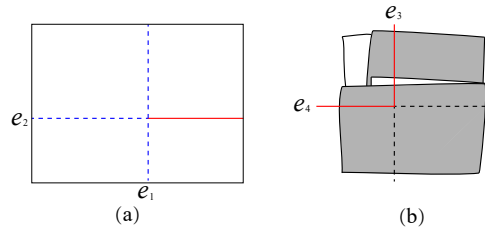


Figure 5.23: Merge occurs when two consecutive end-folds both involve corner squares.

the boundary overlapping. (2) *Necessity of the interchangeable condition: in Classes 6, 7 and 8,  $f_{i-1}$  and  $f_i$  cannot be interchanged when the interchangeable conditions are violated.*

*Proof.* (1) We first provide an overall explanation before the details. Viewing  $R_{i-2}$  as a reduced map, it is clear that in each class, the MV assignment induces the same overlapping order on every respective boundary strip.  $f_{i-1}$  and  $f_i$  are applied on perpendicular boundary strips. At least one of them involves no corner square. Therefore, no overlapping order of a corner square would be simultaneously affected by  $f_{i-1}$  and  $f_i$ . Then, since the only one intersection of two boundary strips is a corner square, the overlapping orders on different boundary strips would not be merged by  $f_{i-1}$  and  $f_i$ . Then, we provide a detailed proof for Class 6. Without loss of generality, we assume that  $f_{i-1} = \{h_a, h_b\}$ ,  $f_i = \{v_c, v_d\}$ , and  $s_{0,0}$  is located at  $A_1$ .  $A_1$  to  $A_9$  in Figure 5.20 indicate the areas separated by  $h_a$ ,  $h_b$ ,  $v_c$ , and  $v_d$ . The certain MV assignment limits the locations of boundary strips. They must locate at all or some of  $A_1$ ,  $A_3$ ,  $A_7$  and  $A_9$ . With the symmetric cases omitted, there are two possible MV assignments, as illustrated in Figure 5.20(a) and (b).

For (a), the overlapping order on every boundary strip is uniquely determined, because the MV assignment uniquely determines the overlap on respective boundary strips as  $(A_1, A_4, A_7)$ ,  $(A_9, A_6, A_3)$ ,  $(A_3, A_2, A_1)$  and  $(A_9, A_8, A_7)$ . These tuples are not merged by  $f_{i-1}$  and  $f_i$ . With no boundary square simultaneously involved in two tuples, the interchangeability is proven. The analysis for (b) is similar.

Class 7 also has two possible MV assignments as (a) and (b) illustrated in Figure 5.21. Each MV assignment has two possible ways to order the folds. Class 8 has only one possible MV assignment with two possible ways to order the folds, as illustrated in Figure 5.22.

(2) We take an instance of Class 8 violating the interchangeable condition to explain the necessity of the condition. As illustrated in Figure 5.23, state (b) is obtained by folding a single piece of paper along  $e_1$  and  $e_2$  as illustrated in (a). For the flat-folded state (b), the two orders of end-folds,  $(e_3, e_4)$  and  $(e_4, e_3)$ , induce different  $O_i(0, n-1)$ s. This fact means that the two end-folds are not interchangeable. Similarly, the other instances of these three classes violating the interchangeable condition also indicate the uninterchangeable property of the two folds.  $\square$

## ivCompleteness of Proof

In the above three sections, we have concluded the proof of (1) in Lemma 5.3.5. Also, we have proved that the interchangeable conditions we provided for Classes 1-8 are necessary. Therefore, to complete the proof of Lemma 5.3.5, we have to prove that every interchangeable case in a partly flat-folded state must belong to one of Classes 1-8.

For a pattern with only a pair of perpendicular crease lines, the four creases are definitely assigned either three “M”s and a “V” or three “V”s and an “M”. The interchange of the two crease lines would change the labels of two creases, which can also be considered as the prohibited change of the assignment on two perpendicular boundary strips. By mathematical induction, the interchange of an odd number of pairs of perpendicular crease lines would definitely change the MV assignment on the boundary strips, while the interchange of an even number of pairs of perpendicular crease lines maintains the original MV assignment on the boundary strips. It forces every interchange of horizontal

fold and vertical folds to be implemented on both an even number of horizontal crease lines and an even number of vertical crease lines. Thus, following, we consider the interchangeable perpendicular creases of the minimal even number such that any feasible interchange between  $p_k$  and  $p_{k+1}$  can be obtained by a combination of these minimal interchanges.

The crimps always affect an even number of layers, so we consider a pair of perpendicular crimps in Class 6.

In a partly flat-folded state, it is possible for an end-fold along a single line to fold an odd number or an even number of layers, which correspondingly involves an odd number or an even number of crease lines. For the end-folds involving an odd number of crease lines, we consider two such consecutive end-folds together to keep the parity. In this case, all the possible interchangeable cases can be classified to (a) two pairs of end-folds (Classes 1, 2, 3) and (b) a pair of end-folds and a crimp (Classes 4, 5). In a middle state where the end-fold corresponds to an even number of layers, the possible interchangeable cases can be classified to (a) two end-folds (Class 8) and (b) an end-fold and a crimp (Class 7).

Because all the possible interchangeable cases are considered, Lemma 5.3.5(2) is proven. Lemmas 5.3.6, 5.3.7 and 5.3.8 conclude the interchangeable condition of consecutive perpendicular folds on the eight classes. In the next section, we will give a folding process based on these classes.

### 5.3.4 Algorithms of the decision problem

In this section, we use the interchangeable condition of consecutive folds to define an equivalence relation on  $\mathcal{F}$ . Then, deciding the validity of  $O$  is turned into deciding whether a representative  $F$  exists or not. We will first define the equivalence relation based on the equivalence relation, and then we will provide the method to find  $F$  as a representative of the corresponding equivalence class. Finally, the algorithm description of the decision problem will be given.

#### The equivalence relation on $\mathcal{F}$

Based on the interchangeable conditions presented for parallel folds and the conditions for perpendicular folds introduced before, an equivalence relation can be defined as below: Taking any two elements  $F_1$  and  $F_2$  in  $\mathcal{F}$ , if  $F_2$  can be induced from  $F_1$  by interchanging the interchangeable folds, then we denote  $F_1 \sim F_2$ . It is easy to check that  $\sim$  is an equivalence relation. Let  $A$  be the set of all the representatives of the equivalence classes, and  $B$  be the set of all the valid  $R_i$ s. This equivalence relation induces an onto mapping from  $A$  to  $B$ , making it possible to check the validity of  $O$  by checking if a corresponding representative whole simple folding sequence  $F$  exists.

In addition, the number of elements in an equivalent class can be exponential to  $m+n$  according to the number of possible interchanges. It means that (1) a valid boundary overlapping order may correspond to an exponential amount of whole simple folding sequences, and (2) a valid boundary overlapping order may correspond to an exponential amount of valid total overlapping orders.

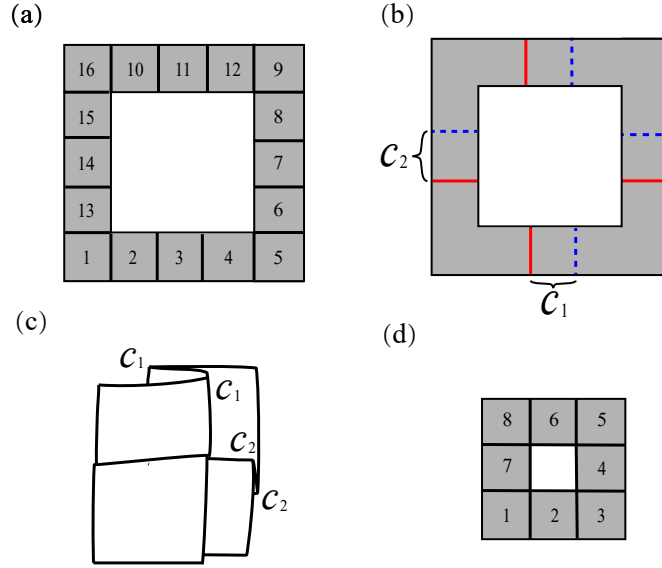


Figure 5.24: The reduction by folding the first crimps.

### First step: crimps

The concrete process to find  $F$  starts from  $R_0$ . The crimps and end-folds are decided step-by-step. Firstly, we find all the available crimps and fold them before the end-folds. This is reasonable because these crimps do not involve corner squares. By Classes 6 and 7 introduced in the last section, they are interchangeable with any end-fold. We first check out the crimps on parallel sides and fold them if they exist. Then, the map can be reduced to a smaller one by applying these crimps, keeping the boundary strips still the boundary strips of the new map. We repeat this process until no more crimp can be found. After folding these crimps, four corner squares should still locate at the four corners. An instance is illustrated in Figure 5.24. (a) gives the overlapping order of all the boundary squares. The first two available crimps  $c_1$  and  $c_2$  on the two parallel sides are illustrated in (b). After folding them, the map is reduced to a new map in (c), whose new overlapping order corresponds to (d).

### Second step: end-folds

The first feasible fold of the reduced map must be an end-fold and must involve corner squares (the same in both the reduced map and the old map) because it is the first end-fold. It is enough to decide whether the end-fold is along a horizontal or vertical crease line. This is because, based on the results we have for  $p_k$ , we can decide the feasible fold (crimp or end-fold) when whether it is horizontal or vertical is known.

The possible cases of the first end-fold are divided into two kinds (i) and (ii). In (i), at least one of the four corner squares satisfies that its two adjacent squares are on the same boundary strip. Figure 5.25 exemplifies the cases which takes  $s_{0,0}$  as such a corner square. When a corner square is the first or the last element in the overlapping order, which has only one adjacent square, we also consider the case as (i). In (ii), the two adjacent squares of any corner square are on perpendicular boundary strips.

For (i), we first consider  $s_{0,0}$ . Without loss of generality, we assume that  $s_{0,0}$  is in  $(s_{i,0}$ ,

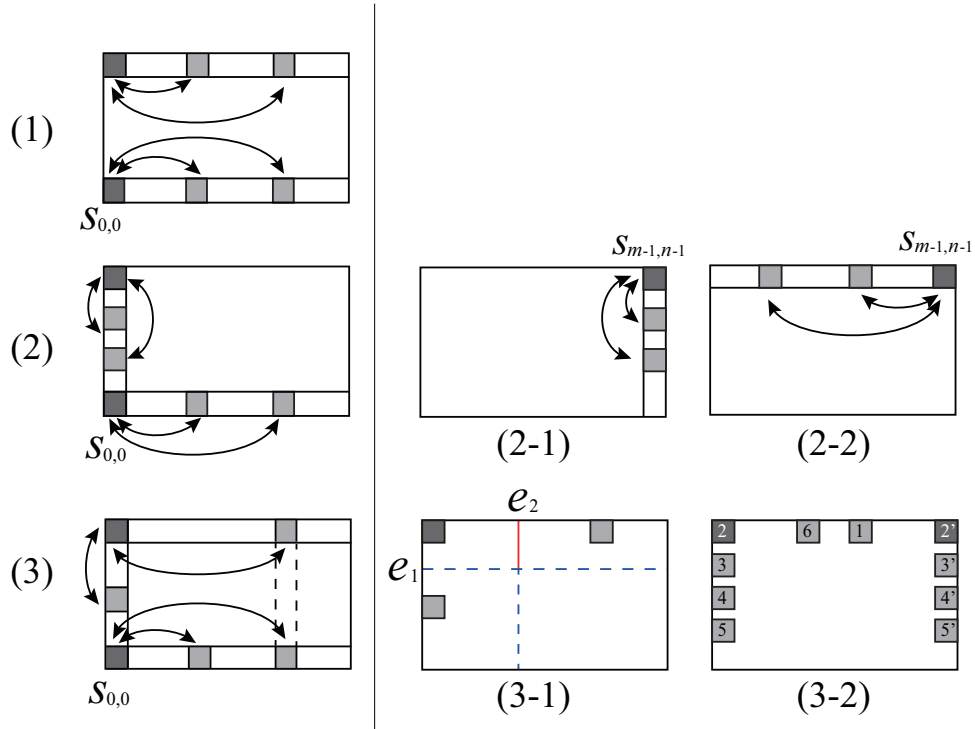


Figure 5.25: Three cases in the first class.

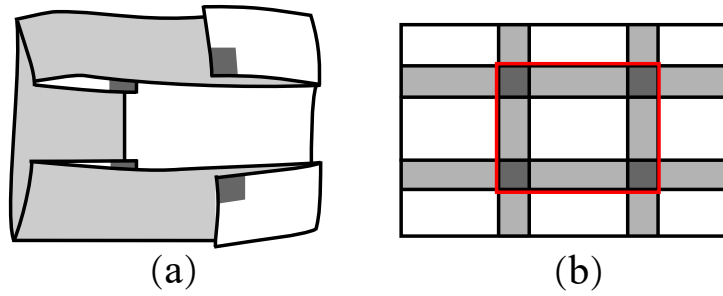


Figure 5.26: The rectangle formed by the four corner squares in a middle state.

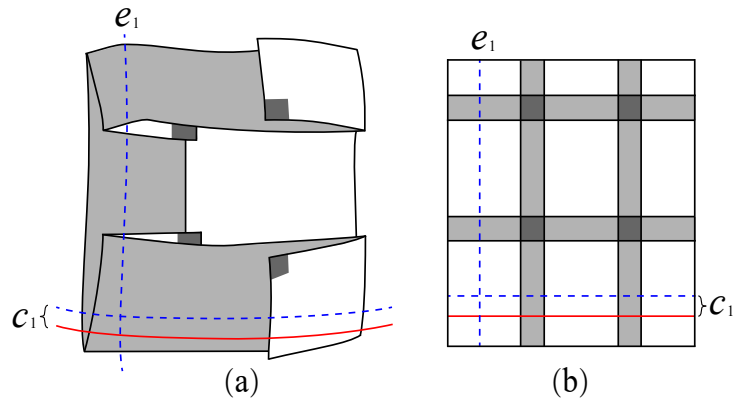


Figure 5.27: An example of the implement of folds involving no corner squares.

$s_{0,0}, s_{j,0}$ .  $(s_{i,0}, s_{0,0}, s_{j,0})$  indicates that the first end-fold involving  $s_{0,0}$  must be vertical. It is necessary to consider the squares adjacent to other corner squares, and we here take  $s_{0,n-1}$  first.

For the convenience of description, we define the *consistency* between two adjacent pairs of squares on parallel boundary strips. If

- (a)  $(s_{i,n-1}, s_{0,n-1})$  (or  $(s_{0,n-1}, s_{j,n-1})$ ) holds when  $n$  is an odd;
- (b)  $(s_{0,n-1}, s_{i,n-1})$  (or  $(s_{j,n-1}, s_{0,n-1})$ ) holds when  $n$  is an even,

we say that the overlapping of  $\{s_{0,n-1}, s_{i,n-1}\}$  (or  $\{s_{0,n-1}, s_{j,n-1}\}$ ) is consistent with  $\{s_{0,0}, s_{i,0}\}$  ( $\{s_{0,0}, s_{j,0}\}$ ); If the reverse holds, we say the overlappings of the corresponding pairs are *inconsistent*. When both pairs satisfy the consistency, we say that  $\{s_{i,n-1}, s_{0,n-1}, s_{j,n-1}\}$  is consistent with  $\{s_{i,0}, s_{0,0}, s_{j,0}\}$ . In this manner, the definition of consistency can further be extended to sets of arbitrary number of squares.

Corresponding to different closures of  $s_{0,n-1}$ , three valid cases omitting the symmetries are summarized in the first column of Figure 5.25. In some cases, a further check is necessary. Corresponding to respective cases, the next check and the decision on the direction of the first end-fold are given in Figure 5.25 and explained as follows:

Case (1). In the overlapping,  $\{s_{i,n-1}, s_{0,n-1}, s_{j,n-1}\}$  is ordered consistently with  $\{s_{i,0}, s_{0,0}, s_{j,0}\}$ . In this case, the first end-fold must be vertical. Otherwise, at least one of  $s_{0,0}$  and  $s_{0,n-1}$  should be adjacent with a boundary square  $s_{0,y}$  (for some  $y$ ).

Case (2). The overlapping includes  $(s_{0,k}, s_{0,n-1}, s_{0,l})$ . Then, at least one horizontal fold involving  $s_{0,n-1}$  precedes the end-fold involving  $s_{0,0}$ . Otherwise,  $s_{0,n-1}$  should be adjacent with a boundary square  $s_{i,n-1}$  or  $s_{j,n-1}$  in the overlapping. However, there may exist end-folds involving only  $s_{m-1,0}$  and  $s_{m-1,n-1}$  that precede this horizontal fold. A further consideration on the closure of  $s_{m-1,n-1}$  is necessary. Two of the three possible cases are illustrated in Figure 5.25:

(2-1). The adjacent squares of  $s_{m-1,n-1}$  are  $s_{m-1,k}$  and  $s_{m-1,l}$  (in either order). It indicates that the horizontal folds precede the vertical folds. Otherwise, at least either  $s_{0,n-1}$  or  $s_{m+1,n-1}$  should be adjacent with a boundary square  $s_{x,n-1}$  (for some  $x$ ).

(2-2). If the closure of  $s_{m-1,n-1}$  is  $(s_{i',n-1}, s_{m-1,n-1}, s_{j',n-1})$ , then the vertical folds involving  $s_{m-1,n-1}$  precede the horizontal folds involving  $s_{0,n-1}$ . The first end-fold should be vertical. Otherwise, either  $s_{0,0}$  should be adjacent with a boundary square  $s_{0,y}$  or  $s_{m-1,n-1}$  should be adjacent with a boundary square  $s_{m-1,y'}$  (for some  $y$  and  $y'$ ).

To avoid repetition, another case (2-3) adjacent squares of  $s_{m-1,n-1}$  belong to different boundary strips, is not discussed here. It can be classified into Case (3) by rotating the map by an angle of 90 degrees along the counterclockwise direction.

Case (3).  $s_{0,n-1}$  is in  $(s_{a,n-1}, s_{0,n-1}, s_{0,b})$  or  $(s_{0,b}, s_{0,n-1}, s_{a,n-1})$  where  $a = i$  or  $a = j$ . It is clear that the first one of all the folds involving  $s_{0,0}$  and  $s_{m-1,0}$  must be vertical, otherwise  $s_{0,0}$  should touch a  $s_{0,y}$ . There are only two cases where the first fold is horizontal.

(3-1) The first fold is horizontal if there exists an adjacent relation on  $\{s_{u,n-1}, s_{v,n-1}\}$  where  $u = 0$  or  $m - 1$  and is inconsistent with  $\{s_{u,0}, s_{v,0}\}$ . In other words, there exists a corner square on the upper boundary strip whose adjacent relation is inconsistent with its parallel square on the lower boundary strip. This means that the folds on the two horizontal parallel strips are not the same for the reason that the upper boundary strip is overturned to the other side before any vertical folds as the figure labelled 3-1 in the second column of Figure 5.25,  $e_2$  induces the adjacent relation of  $\{s_{0,n-1}, s_{a,n-1}\}$  in and  $\{s_{0,0}, s_{a,0}\}$  and is thus decided to be folded after  $e_1$ . The inconsistent closure corresponds

to that an odd number of horizontal folds precede the vertical folds.

(3-2) If there exists a closure  $S=(s_{u,n-1}, s_{0,a_1}, s_{0,a_2}, \dots, s_{0,a_k}, s_{v,n-1})$  ( $k > 2$ ) including  $s_{0,n-1}$  as the second or penultimate element in this tuple (as the sequence of squares (1, 2, 3, 4, 5, 6) in the figure labelled 3-2 in the third column of Figure 5.25), and there also exists a closure on  $\{s_{m-1,a_1}, s_{m-1,a_2}, \dots, s_{m-1,a_k}\}$  consistent with this sub-sequence (as (2', 3', 4', 5')) in the figure labelled 3-2 in the third column of Figure 5.25), then there exists an even number of horizontal folds precede the vertical folds.  $k > 2$  ensures that the number of horizontal folds is at least two. The closure on  $\{s_{0,a_1}, s_{0,a_2}, \dots, s_{0,a_k}\}$  and  $\{s_{m-1,a_1}, s_{m-1,a_2}, \dots, s_{m-1,a_k}\}$  (at least three pairs of squares on two vertical boundary strips are folded together) indicates that these horizontal folds either are folded before the vertical ones, or form zigzag for each corner square with the vertical folds in the folded state. The later case corresponds to Classes 4 and 5. By the interchangeability, the horizontal fold can be applied at first. This case corresponds to that an even number of horizontal folds precede the vertical folds.

The above analysis exhausts both cases that the proceeding horizontal folds are in an odd number and in an even number. In all the other cases, the first end-fold is decided to be vertical.

(ii) is the case that each corner square has its two adjacent squares on perpendicular boundary strips. The closures of the four corner squares (we consider each corner square with its adjacent two squares as the closure of the corner square here) are formed by at least four end-folds. Figure 5.16 gives all the instances where the closures of the four corner squares are formed by four consecutive end-folds. Each end-fold assigns new adjacent squares to the pair of corner squares it involves.

When more than four end-folds form the closures of the four corner squares, there must exist some end-folds only assign one or no new adjacent square to the pair of corner squares it involves. Because the first end-fold must assign new adjacent squares to two corner squares, these end-folds are not possible to be the first fold. Correspondingly, by checking the consistency of adjacent relations of pairs of parallel corner squares, at most four end-folds can become candidates for the first fold. In the following, we only discuss the case that the first four end-folds causes all the relations between the four corner squares and their adjacent squares. To form all these relations, every two creases around a corner square must be labeled differently. Thus Figure 5.16 exhausts all the possible cases of the MV assignments.

The first check is on whether the two parallel pairs of folds respectively have unique orders or not. If the case matches with Figure 5.16, then there exist two choices to order these end-folds as shown in the second and the third columns Figure 5.16. We choose an arbitrary one. Otherwise, the order of the four folds can be uniquely decided by the overlapping order of boundary strips, as illustrated in Figure 5.18. Their MV assignments in L1-L3 are the same as Classes 1, 2, 3 in Figure 5.16, whereas the interchangeable conditions are not satisfied. The shadowed two horizontal and two vertical strips in Figure 5.18 L1 indicate the locations of boundary strips. The overlapping order of boundary strips after the four end-folds is indicated by the indices written inside the map in Figure 5.18 C and R. Correspondingly, the known closures on the boundary strips uniquely decide the order of these four end-folds.

When the direction of the current fold is decided, the first end-fold can be decided using the afore-mentioned method. After the first end-fold, we have to find the crimps



before applying other end-folds. The reason is that some boundary squares change their upward side and may form new feasible crimps.

### Third step: folds on partly folded states

After the steps introduced above, the original boundary strips may no longer locate at the boundary of the current reduced map. Unlike in  $R_0$ , there may exist some crimps involving corner squares and some end-folds involving no corner squares this time. According to the interchangeability in Classes 6 to 8, if there are some folds only influencing the overlapping outside the rectangle formed by the four corner squares (as illustrated by the red line segments in Figure 5.26) in a middle state, we can first apply these folds. For example, in the instance presented in Figure 5.27, the end-fold  $e_1$  and the crimp  $c_1$  are firstly folded. Such end-folds and crimps are founded with the method introduced in the first paragraph of the last section.

Next is a repeating process of the check as for  $R_0$  until the map is reduced to a state where either pair of parallel boundary strips overlap each other. When deciding the order of perpendicular folds, if they form the cases in Classes 1 to 8, we can order the folds as any of the orders introduced in the last section. Otherwise, the order of the folds is always uniquely decided by the overlapping of the boundary strips. The decision is similar to the analysis illustrated in Figure 5.18, just with crimps included.

The above checking process ends when a pair of parallel boundary strips totally overlap each other. This state can be viewed as a 1D map (introduced in the last section) and then handled with the same method as the check on a pair of parallel boundary strips. If  $O$  is a valid boundary overlapping order in the simple fold model, we can finally reduce the map to the size of  $1 \times 1$ . A corresponding representative folding  $F$  is composed of the uniquely decided folds and the arbitrary choices for interchangeable folds.

### Algorithm description

We give Algorithm 6 for the decision on the validity of  $O$ , i.e., an array of  $2m + 2n - 4$  squares. The index of a square in the array indicates its order in the boundary overlap in the simple fold model. When  $O$  is valid, the algorithm outputs a representative whole simple folding sequence  $F$ .

The time complexity of this algorithm is  $O(m + n)$ . The computation of the MV assignment on the boundary strips is realized by checking every pair of neighbor squares, which costs  $O(m + n)$  time. By a traverse of  $O$ , the input order of all the boundary squares can be saved as a directed graph  $G$ . In  $G$ , each node represents a square. Each edge represents the adjacent relation between a pair of squares. Also, we assign the value one to each edge as its weight. During the folding process, once an adjacent relation is induced by the folds we find, we reassign the value zero to the corresponding edge. The reduction of the map is indicated by the decrease in the values of the edges, each  $R_i$  is then represented by  $G$  after some reduction. A valid final state  $R_t$  corresponds to  $G$  with the total weight zero on all the edges. At every step, the new neighbor squares can be found by checking the value of the path connecting two nodes.

When we start folding, the first search for the possible crimps is realized through two steps. The first step is finding out the nodes representing the neighbor squares which are adjacent in  $G$ . The next step is, corresponding to the squares found on two

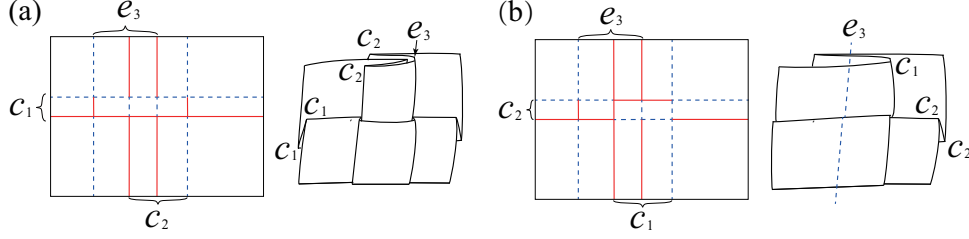


Figure 5.28: A possible lock induced by Step 3.  $e_3$  is an end-fold after  $c_1$  and  $c_2$ . The interchange of  $c_1$  and  $c_2$  makes a later fold  $e_3$  becoming unavailable.

parallel boundary strips, deciding whether or not the same crimps exist on the two strips according to the formula we gave in the first paragraph of the last section. The first step needs constant time, and the second one needs  $O(m)$  and  $O(n)$  time for horizontal and vertical crimps, respectively. This process is applied multiple times along the folding. Because repetitious search on the adjacent relation can be avoided by referencing the weights of paths in  $G$ , the total search for crimps in all the steps costs  $O(m + n)$  time.

Next, each time we decide the end-fold, the check for the closures of the four corner squares costs constant time. This is because we only have to consider the two adjacent squares of every corner square. During the reduction of the map based on the decided folds, every adjacent relation is considered in the computation only once. Thus, the ordering of the end-folds and crimps totally costs  $O(m + n)$  time.

In conclusion, the realization of Algorithm 6 costs  $O(m + n)$  time.

### 5.3.5 Enumeration of valid whole simple folding sequences

In this section, we propose an algorithm to enumerate all the valid whole simple folding sequences matching a valid boundary overlapping order. It uses  $F$  obtained by Algorithm 6. Other valid whole simple folding sequences are induced from  $F$  following the five steps below. We also give the concrete algorithm description as Algorithm 8.

First, the procedure is given as follows.

- S1. Compute  $P = (p_1, p_2, \dots, p_l)$  by grouping the parallel folds in  $F$ ;
- S2. Find all the interchangeable pairs of folds in each  $p_i$ ;
- S3. Record all the different folding sequence obtained by interchanging the pairs found in Step 2;
- S4. For each sequence obtained in S3, output the sequence as a valid whole simple folding sequence; Then, find the perpendicular interchangeable folds in the sequence.
- S5. Record all the different simple folding sequences by interchanging the folds checked out in S4. For each of them, check the feasibility, i.e., whether some folds would form a lock on the layers to be folded after or not (a possible lock is illustrated in Figure 5.28. The interchange of  $c_1$  and  $c_2$  makes a later fold  $e_3$  on the shadowed layers becoming unavailable. The lock may be induced essentially because the interchange inside a  $p_i$  may turn the folds into general simple folds. On the opposite,  $F$  would not induce any lock since it includes only crimps and end-folds). If there exists no such lock, output the sequence as a valid whole simple folding sequence; then reapply Steps 1 to 4.

Because the equivalence relation is built on the whole solution space  $\mathcal{F}$ , it is clear

**Input** : A total order  $O$  on the set of boundary squares//indicating  
 $2m + 2n - 4$  numbered squares

**Output:** A boolean value // the validity of  $O$  in the simple fold model

**Begin**

```

initialization
 $G \leftarrow$  A directed graph with  $2m + 2n - 4$  nodes // the folded state of the map
 $F \leftarrow \emptyset$  // the folds applied to the map in order
Compute the MV assignment on the boundary strips according to  $O$ 
if  $|F| = 0$  then
    Find the first horizontal and vertical folds (Section 4.1)
    if no feasible folds can be found then
        | return false //  $O$  is invalid
    end
    while crimp exists do
        | Append the crimps to  $F$ , update  $G$  by assigning zero to the edges
        | incident to the new adjacent pairs of nodes induced by these crimps
    end
    Find End-Folds( $F, G$ )
end
while no parallel boundary strips totally overlap each other according to  $G$  do
    | Find out the crimps or end-folds involving no corner squares in the
    | current state of the map indicated by  $G$ , append them to  $F$ , update  $G$ 
    | by assigning zero to the edges incident to the new adjacent pairs
    | Find End-Folds( $F, G$ )
end
while not all the edges in  $G$  are assigned zero do
    | Find out the folds involving no corner squares, append them to  $F$ , update
    |  $G$ 
    | Find the next crimps or end-folds according to the adjacent squares of the
    | corner squares in the current map
    if no feasible folds can be found then
        | return false //  $O$  is invalid
    else
        | Decide the unique order of folds, append them to  $F$ , update  $G$ 
    end
    return true
end

```

```

Function Find_End-Folds( $F, G$ ):
  Check the closure of the adjacent relation of corner squares
  if Classes 1, 2, 3 exists then
    Decide the first four consecutive end-folds. Assign either of the
    corresponding feasible order to them and append them to  $F$ , update
     $G$ 
  else
    if the first end-fold can be uniquely decided or is arbitrarily decided by
    the interchangeability of Classes 4 and 5 then
      Append the first end-fold to  $F$ , update  $G$  and check for the crimps
      if crimps exist in the current state then
        Append the crimps to  $F$ , update  $G$ 
      else
        Append the next end-folds to  $F$ , update  $G$ 
      end
    else
      return false //  $O$  is invalid
    end
  end
End Function
end

```

**Algorithm 5:** Algorithm of the decision problem of the boundary overlapping order

that this method provides all the valid whole simple folding sequences. The analysis for the time complexity is as follows. Firstly, because  $F$  is comprised by  $m + n$  crease lines, S1 costs  $O(m + n)$  time. Corresponding to S2 and S5, finding the interchangeable pairs of folds from an existing folding sequence costs  $O(m + n)$  time each time. The interchangeability is decided according to the analysis in the last section, respectively, for parallel and perpendicular boundary strips. In S3, each interchange costs constant time to obtain a new folding sequence. Then, corresponding to S5, each check of the feasibility can be concluded in  $O(mn)$  time by a traverse of the corresponding order of all the squares to decide whether it represents a valid overlapping. When none of the interchanges of perpendicular folds produces a valid whole simple folding sequence, there would be at most  $O(\max\{m, n\})$  times of interchanges to change back to the valid whole simple folding sequence before the interchanges. Because the interchanges of parallel folds are always feasible, we can conclude that every time delay can be limited in  $O(\max\{m, n\}mn)$  time.

In conclusion, for the enumeration of valid whole simple folding sequences, Algorithm 2 takes  $O(m + n)$  time on precomputing and enumerates each folded state in  $O(\max\{m, n\}mn)$  time delay.

**Input** : A valid whole simple folding sequence  $F$   
**Output**: Other valid whole simple folding sequences  $F'$   
**begin**  
     $P \leftarrow \emptyset$  // consecutive parallel folds  
     $F' \leftarrow \emptyset$  // the produced whole simple folding sequence  
    **while**  $F' \neq F$  **do**  
         $F' \leftarrow F$   
        **if**  $F'$  involves interchangeable perpendicular pairs **then**  
            **foreach** interchangeable perpendicular pair in  $F'$  **do**  
                Update  $F'$  by interchanging the perpendicular pair in  $F'$   
                Check the validity of  $F'$   
                **if**  $F'$  is valid **then**  
                    Output  $F'$   
                **end**  
                Compute(or update)  $P = (p_1, p_2, \dots, p_l)$  by grouping the parallel folds in  $F'$   
                **foreach**  $p_i$  in  $P$  **do**  
                    **if**  $p_i$  involves interchangeable parallel pairs **then**  
                        **foreach** interchangeable parallel pair in  $p_i$  **do**  
                            Update  $F'$  by interchanging the pair in  $F'$   
                            Check the validity of  $F'$   
                            **if**  $F'$  is valid **then**  
                                Output  $F'$   
                            **end**  
                        **end**  
                    **end**  
                **end**  
        **end**  
    **end**  
    **return** 1 // finish the progress  
**end**

**Algorithm 6:** Algorithm of the enumeration problem of the boundary overlapping order

# Chapter 6

## Conclusion and Future Work

Our work involves two main variations of map folding problems, with one as the decision problem on the flat-foldability of box-pleating patterns and the other one as the decision problem on the validity of overlapping orders in the final flat-folded states. For the first variation, we proved that the flat-foldability of both box-pleating patterns of size  $1 \times n$  and  $2 \times n$  is the same as their local flat-foldability and thus can be decided in time linear in  $n$ . These results filled in part of the unknown area about the computational complexity of the decision on the flat-foldability of box-pleating patterns under the size  $4 \times n$ , remaining the only gap about box-pleating patterns of size  $3 \times n$ . For the second variation, we proposed the linear-time algorithms for the decision problems on the validity of total overlapping orders of a box-pleating pattern with all the creases in the general fold model, as well as the validity of both total and boundary overlapping of a map in the simple fold model. Also, we considered the distinct reachable overlapping orders in different folding models and discussed their relationship. Finally, we presented some results on partial overlapping orders in maps of size  $1 \times n$ .

In the following, we will propose some open directions as future works of the two variations, respectively. About the flat-foldability of given patterns, we propose two open directions.

Open Direction 1: What is the computational complexity of the decision problem on the flat-foldability of  $3 \times n$  box-pleating patterns?

Open Direction 2: What is the computational complexity of the decision problem on the flat-foldability of  $1 \times n$ ,  $2 \times n$ , and  $3 \times n$  box-pleating patterns with also MV-assignments?

The next two open directions are about the validity of given overlapping orders.

Open Direction 3: What is the computational complexity of the decision problem on the validity of a given partial order (in a general sense)?

Open Direction 4: What is the computational complexity of the decision problem on the validity of multiple partial orders? In other words, what is the computational complexity to decide whether there exists a flat-folded state following a set of partial orders simultaneously?

Above all, the most intricate and interesting open problem is still the map folding problem itself.

# Bibliography

- [1] Zachary Abel, Erik D Demaine, Martin L Demaine, David Eppstein, Anna Lubiw, and Ryuhei Uehara. Flat foldings of plane graphs with prescribed angles and edge lengths. In *International Symposium on Graph Drawing*, pages 272–283. Springer, 2014.
- [2] H. A. Akitaya, E. D. Demaine, and J. S. Ku. Simple Folding is Really Hard. *Journal of Information Processing*, 25:580–589, 2017.
- [3] Hugo Akitaya, Erik D Demaine, Takashi Horiyama, Thomas C Hull, Jason S Ku, and Tomohiro Tachi. Rigid foldability is np-hard. *arXiv preprint arXiv:1812.01160*, 2018.
- [4] Hugo A Akitaya, Kenneth C Cheung, Erik D Demaine, Takashi Horiyama, Thomas C Hull, Jason S Ku, Tomohiro Tachi, and Ryuhei Uehara. Box pleating is hard. In *Japanese Conference on Discrete and Computational Geometry and Graphs*, pages 167–179. Springer, 2015.
- [5] J M Alexander. An approximate analysis of the collapse of thin cylindrical shells under axial loading. *The Quarterly Journal of Mechanics and Applied Mathematics*, 13(1):10–15, 1960.
- [6] Roger C Alperin and Robert J Lang. One-, two-, and multi-fold origami axioms. *Origami*, 4:371–393, 2009.
- [7] Byoungkwon An, Erik D Demaine, Martin L Demaine, and Jason S Ku. Computing 3sat on a fold-and-cut machine. In *CCCG*, pages 208–213, 2017.
- [8] Yoshiaki Araki, Takashi Horiyama, and Ryuhei Uehara. Common unfolding of regular tetrahedron and johnson-zalgaller solid. *Journal of Graph Algorithms and Applications*, 2016.
- [9] E. M. Arkin, M. A. Bender, E. D. Demaine, M. L. Demaine, J. S. B. Mitchell, S. Sethia, and S. S. Skiena. When Can You Fold a Map? *Computational Geometry: Theory and Applications*, 29(1):23–46, 2002.
- [10] Leonard Asimow and Ben Roth. The rigidity of graphs. *Transactions of the American Mathematical Society*, 245:279–289, 1978.
- [11] Devin J Balkcom, Erik D Demaine, Martin L Demaine, John A Ochsendorf, and Zhong You. Folding paper shopping bags. In *Origami4: Proceedings of the 4th International Meeting of Origami Science, Math, and Education (OSME 2006)*, pages 315–334, 2009.
- [12] B. Baumslag, B. Chandler, and Schaum’s outline Series. *Theory and problems of group theory*. McGraw-Hill New York, 1968.

- [13] Nadia M Benbernou, Erik D Demaine, Martin L Demaine, and Aviv Ovadya. Universal hinge patterns to fold orthogonal shapes. In *Origami5: Proceedings of the 5th International Conference on Origami in Science, Mathematics and Education*, pages 405–420, 2010.
- [14] M. Bern and B. Hayes. The Complexity of Flat Origami. In *Ann. ACM-SIAM Symposium on Discrete Algorithms*, pages 175–183. ACM, 1996.
- [15] Marshall Bern, Erik D Demaine, David Eppstein, Eric Kuo, Andrea Mantler, and Jack Snoeyink. Ununfoldable polyhedra with convex faces. *Computational Geometry*, 24(2):51–62, 2003.
- [16] Therese Biedl, Anna Lubiw, and Julie Sun. When can a net fold to a polyhedron? *Computational Geometry*, 31(3):207–218, 2005.
- [17] Amartya Shankha Biswas and Erik D Demaine. Common development of prisms, anti-prisms, tetrahedra, and wedges. In *CCCG*, pages 202–207, 2017.
- [18] Li Chenglei and Zhou Jingqi. A general algorithm of flattening convex prismatoids.
- [19] Gary PT Choi, Levi H Dudte, and L Mahadevan. Programming shape using kirigami tessellations. *Nature materials*, 18(9):999–1004, 2019.
- [20] Robert Connelly, Erik D Demaine, and Günter Rote. Straightening polygonal arcs and convexifying polygonal cycles. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 432–442. IEEE, 2000.
- [21] Mirela Damian, Robin Flatland, Henk Meijer, and Joseph O’Rourke. Unfolding well-separated orthotrees. In *15th Annu. Fall Workshop Comput. Geom.*, pages 23–25. Citeseer, 2005.
- [22] Christian Danielson, Ali Mehrnezhad, Ashkan YekrangSafakar, and Kidong Park. Fabrication and characterization of self-folding thermoplastic sheets using unbalanced thermal shrinkage. *Soft Matter*, 13(23):4224–4230, 2017.
- [23] E. D. Demaine and Joseph O’Rourke. *Geometric folding algorithms*. Cambridge university press Cambridge, 2007.
- [24] Erik D Demaine. Origami, linkages, and polyhedra: folding with algorithms. In *European Symposium on Algorithms*, pages 1–1. Springer, 2006.
- [25] Erik D Demaine, Martin L Demaine, and David Eppstein. Acutely triangulated, stacked, and very unfoldable polyhedra. *arXiv preprint arXiv:2007.14525*, 2020.
- [26] Erik D Demaine, Martin L Demaine, and Duks Koschitz. Reconstructing david huffman’s legacy in curved-crease folding. In *Origami 5: Fifth International Meeting of Origami Science, Mathematics, and Education*, page 39. Taylor & Francis Group, 2011.
- [27] Erik D Demaine, Martin L Demaine, Duks Koschitz, and Tomohiro Tachi. Curved crease folding: a review on art, design and mathematics. In *Proceedings of the IABSE-IASS Symposium: Taller, Longer, Lighter*, pages 20–23. Citeseer, 2011.
- [28] Erik D Demaine, Martin L Demaine, and Jason Ku. Folding any orthogonal maze. In *Origami5: Proceedings of the 5th International Conference on Origami in Science, Mathematics and Education (OSME 2010)*, pages 449–454, 2011.



- [29] Erik D Demaine, Martin L Demaine, Tomoko Taniguchi, Ryuhei Uehara, et al. Impossible folding font. In *Bridges 2019 Conference Proceedings*, pages 51–58. Tessellations Publishing, 2019.
- [30] Erik D Demaine, Satyan L Devadoss, Joseph SB Mitchell, and Joseph O’Rourke. Continuous foldability of polygonal paper. In *CCCG*, pages 64–67, 2004.
- [31] Erik D Demaine et al. Origami maze puzzle font. 2010.
- [32] Erik D Demaine and Jason S Ku. Filling a hole in a crease pattern: Isometric mapping from prescribed boundary folding. *Origami6: I. Mathematics*, pages 177–188, 2015.
- [33] Erik D Demaine and Joseph O’Rourke. Open problems from cccg 2008. In *CCCG*, pages 185–187, 2001.
- [34] Erik D Demaine and Joseph O’Rourke. A survey of folding and unfolding in computational geometry. *Combinatorial and computational geometry*, 52:167–211, 2005.
- [35] Martin L Demaine, Robert Hearn, Jason Ku, and Ryuhei Uehara. Rectangular unfoldings of polycubes.
- [36] Kristin DeSplinter, Satyan L Devadoss, Jordan Readyhough, and Bryce Wimberly. Unfolding cubes: nets, packings, partitions, chords. *arXiv preprint arXiv:2007.13266*, 2020.
- [37] Peter Dieleman, Niek Vasmel, Scott Waitukaitis, and Martin van Hecke. Jigsaw puzzle design of pluripotent origami. *Nature Physics*, 16(1):63–68, 2020.
- [38] Samuel Felton, Michael Tolley, Erik Demaine, Daniela Rus, and Robert Wood. A method for building self-folding machines. *Science*, 345(6197):644–646, 2014.
- [39] ALBERTO FRISOLI. A study about novel methodologies for the generation of polycube structures for hexahedral meshing. 2019.
- [40] Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.
- [41] Milton R Garza, Edwin A Peraza-Hernandez, and Darren J Hartl. Self-folding origami surfaces of non-zero gaussian curvature. In *Behavior and Mechanics of Multifunctional Materials XIII*, volume 10968, page 109680R. International Society for Optics and Photonics, 2019.
- [42] ALIM HAMEED and RASHED H YASEEN. Using of paper folding art and computer software to produce architectural models. *Journal of Engineering Science and Technology*, 14(6):3194–3212, 2019.
- [43] Edwin A Peraza Hernandez, Darren J Hartl, and Dimitris C Lagoudas. Kinematics of origami structures with smooth folds. In *Active Origami*, pages 201–268. Springer, 2019.
- [44] Takashi Horiyama and Wataru Shoji. The number of different unfoldings of polyhedra. In *International Symposium on Algorithms and Computation*, pages 623–633. Springer, 2013.
- [45] David A. Huffman. Curvature and creases: A primer on paper. *IEEE Transactions on computers*, (10):1010–1019, 1976.

- [46] T. Hull. The combinatorics of flat folds: a survey. In *Third International Meeting of Origami Science*, pages 29–38, 2002.
- [47] Thomas Hull. On the mathematics of flat origamis. *Congressus numerantium*, pages 215–224, 1994.
- [48] Thomas C Hull. Origami design secrets: mathematical methods for an ancient art. *The Mathematical Intelligencer*, 27(2):92–95, 2005.
- [49] Thomas C Hull. Counting mountain-valley assignments for flat folds. *arXiv preprint arXiv:1410.5022*, 2014.
- [50] Thomas C Hull et al. Modelling the folding of paper into three dimensions using affine transformations. *Linear Algebra and its applications*, 348(1-3):273–282, 2002.
- [51] Paul Jackson. *Cut and fold techniques for Pop-Up designs*. Laurence King Publishing, 2014.
- [52] Hyungmin Jun, Fei Zhang, Tyson Shepherd, Sakul Ratanalert, Xiaodong Qi, Hao Yan, and Mark Bathe. Autonomously designed free-form 2d dna origami. *Science advances*, 5(1):eaav0655, 2019.
- [53] J. Justin. Towards a mathematical theory of origami. In *International Meeting of Origami Science and Scientific Origami*, pages 15–29. K. Miura (Ed.), 1996.
- [54] Jacques Justin. Résolution par le pliage de l’ équation du troisieme degré et applications géométriques. In *Proceedings of the first international meeting of origami science and technology*, pages 251–261. Ferrara, Italy, 1989.
- [55] Soroush Kamrava, Ranajay Ghosh, Yu Yang, and Ashkan Vaziri. Slender origami with complex 3d folding shapes. *EPL (Europhysics Letters)*, 124(5):58001, 2018.
- [56] Soroush Kamrava, Davood Mousanezhad, Samuel M Felton, and Ashkan Vaziri. Programmable origami strings. *Advanced Materials Technologies*, 3(3):1700276, 2018.
- [57] K. Kasahara and T. Takahama. *Origami for the Connoisseur*. Japan Publications Inc., 1987.
- [58] T. Kawasaki. On the relation between mountain-creases and valley-creases on a flat origami. In *International Meeting of Origami Science and Technology*, pages 229–237. H. Huzita, 1989.
- [59] Yannick L Kergosien, Hironobu Gotoda, and Tosiyasu L Kunii. Bending and creasing virtual paper. *IEEE Computer graphics and applications*, 14(1):40–48, 1994.
- [60] Rob Kirby and Edited Rob Kirby. Problems in low-dimensional topology. In *Proceedings of Georgia Topology Conference, Part 2*. Citeseer, 1995.
- [61] John E Koehler. Folding a strip of stamps. *Journal of Combinatorial Theory*, 5(2):135–152, 1968.
- [62] Robert J Lang. Origami: Complexity increasing. *Engineering & Science*, 52(2):16–23, 1989.

- [63] Robert J Lang. Computational origami: from flapping birds to space telescopes. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 159–162, 2009.
- [64] Robert J Lang and K Miura. The tree method of origami design. In *Proceedings of the 2nd International Meeting of Origami Science and Scientific Origami*, pages 73–82, 1994.
- [65] Stefan Langerman and Andrew Winslow. Polycube unfoldings satisfying conway’s criterion. In *Japan Conference on Discrete and Computational Geometry and Graphs (JCDCG3 2016)*, 2016.
- [66] Zsolt Lengvarszky et al. An origami puzzle of intersecting cubes. In *Proceedings of Bridges 2010: Mathematics, Music, Art, Architecture, Culture*, pages 367–370. Tessellations Publishing, 2010.
- [67] Zhejian Li, Wensu Chen, Hong Hao, Jian Cui, and Yanchao Shi. Experimental study of multi-layer folded truncated structures under dynamic crushing. *International Journal of Impact Engineering*, 131:111–122, 2019.
- [68] Tomás Lozano-Pérez and Michael A Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.
- [69] Guoxing Lu and TX Yu. *Energy absorption of structures and materials*. Elsevier, 2003.
- [70] Jorge C Lucero. On the elementary single-fold operations of origami: reflections and incidence constraints on the plane. *arXiv preprint arXiv:1610.09923*, 2016.
- [71] Y. Matsukawa, Y. Yamamoto, and J. Mitani. Enumeration of Flat-foldable Crease Patterns in the Square/Diagonal Grid and Their Folded Shapes. *Journal for Geometry and Graphics*, 21(2):169–178, 2017.
- [72] Toshiyuki Meguro. Jitsuyou origami sekkeihou [practical methods of origami designs]. *Origami Tanteidan Shinbun*, 2(7-14):1991–1992, 1991.
- [73] Toshiyuki Meguro. ’ tobu kuwagatamushi ’ -to ryoikienbunshiho [ ’ flying stag beetle ’ and the circular area molecule method], 1994.
- [74] Jun Mitani. Oripa (origami pattern editor), 2005.
- [75] Jun Mitani. A design method for 3d origami based on rotational sweep. *Computer-Aided Design and Applications*, 6(1):69–79, 2009.
- [76] Jun Mitani. Column-shaped origami design based on mirror reflections. *Journal for Geometry and Graphics*, 16(2):185–194, 2012.
- [77] Shuhei Miyashita, Isabella DiDio, Ishwarya Ananthabhotla, Byoungkwon An, Cynthia Sung, Slava Arabagi, and Daniela Rus. Folding angle regulation by curved crease design for self-assembling origami propellers. *Journal of Mechanisms and Robotics*, 7(2), 2015.
- [78] Shuhei Miyashita, Steven Guitron, Shuguang Li, and Daniela Rus. Robotic metamorphosis by origami exoskeletons. *Science Robotics*, 2(10):eaao4369, 2017.
- [79] T. Morgan. *Map folding*. Master thesis, Massachusetts Institute of Technology, 2012.
- [80] Rahnuma Islam Nishat. *Map folding*. PhD thesis, 2013.

- [81] Kay Paulus. Geometrische konstruktionen und origami. *arXiv preprint arXiv:1810.06852*.
- [82] Matthew B Pinson, Menachem Stern, Alexandra Carruthers Ferrero, Thomas A Witten, Elizabeth Chen, and Arvind Murugan. Self-folding origami at any energy scale. *Nature communications*, 8(1):1–8, 2017.
- [83] Lydie Richaume, Eric Andres, Gaëlle Largeteau-Skapin, and Rita Zrour. Unfolding h-convex manhattan towers. 2018.
- [84] Uladzimir Sayevich. Synthesis, surface design and assembling of colloidal semiconductor nanocrystals. 2016.
- [85] Hiroki Shigemune, Shingo Maeda, Yusuke Hara, Naoki Hosoya, and Shuji Hashimoto. Origami robot: a self-folding paper robot with an electrothermal actuator created by printing. *IEEE/ASME Transactions on Mechatronics*, 21(6):2746–2754, 2016.
- [86] John S Smith. Pureland origami 1, 2, and 3. *British Origami Society. Booklets*, 14:29.
- [87] Ileana Streinu and Walter Whiteley. The spherical carpenter ’ s rule problem and conical origami folds. In *Proc. 11th Annual Fall Workshop on Computational Geometry*, 2001.
- [88] Daniel M Sussman, Yigil Cho, Toen Castle, Xingting Gong, Euiyeon Jung, Shu Yang, and Randall D Kamien. Algorithmic lattice kirigami: A route to pluripotent materials. *Proceedings of the National Academy of Sciences*, 112(24):7449–7453, 2015.
- [89] Tomohiro Tachi. Simulation of rigid origami. *Origami*, 4(08):175–187, 2009.
- [90] Tomohiro Tachi. Freeform rigid-foldable structure using bidirectionally flat-foldable planar quadrilateral mesh. *Advances in architectural geometry*, 14(2):203–215, 2010.
- [91] Tomohiro Tachi. Geometric considerations for the design of rigid origami structures. In *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*, volume 12, pages 458–460. Elsevier Ltd, 2010.
- [92] Tomohiro Tachi and Gregory Epps. Designing one-dof mechanisms for architecture by rationalizing curved folding. In *International Symposium on Algorithmic Design for Architecture and Urban Design (ALGODE-AIJ). Tokyo*, volume 5, page 6, 2011.
- [93] Tomohiro Tachi and Thomas C Hull. Self-foldability of rigid origami. *Journal of Mechanisms and Robotics*, 9(2), 2017.
- [94] Tomohiro Tachi and Koryo Miura. Rigid-foldable cylinders and cells. *Journal of the international association for shell and spatial structures*, 53(4):217–226, 2012.
- [95] Mia Tsiamis, Alfonso Oliva, and Michele Calvano. Algorithmic design and analysis of architectural origami. *Nexus Network Journal*, 20(1):59–73, 2018.
- [96] Nicholas Turner, Bill Goodwine, and Mihir Sen. A review of origami applications in mechanical engineering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 230(14):2345–2362, 2016.
- [97] Ryuhei Uehara. Stamp foldings with a given mountain-valley assignment. In *Origami 5*, pages 599–612. AK Peters/CRC Press, 2016.
- [98] Ryuhei Uehara. Unfolding. In *Introduction to Computational Origami*, pages 3–7. Springer, 2020.

- [99] Ryuhei Uehara. Zipper-unfolding. In *Introduction to Computational Origami*, pages 143–153. Springer, 2020.
- [100] Toshiyuki Yoshimura, Naoko Miura, Shinji Okazaki, Minoru Toriumi, and Hiroshi Shiraishi. Method for forming a pattern and forming a thin film used in pattern formation, April 30 1996. US Patent 5,512,328.
- [101] Hang Yuan, JH Pikul, and Cynthia Sung. Programmable 3-d surfaces using origami tessellations. In *Proc. 7th Int. Meeting Origami Sci., Math. Educ.*, pages 893–906, 2018.
- [102] Lin Yuan, Haoyuan Shi, Jiayao Ma, and Zhong You. Quasi-static impact of origami crash boxes with various profiles. *Thin-Walled Structures*, 141:435–446, 2019.
- [103] Wuxiang Zhang, Shengnan Lu, and Xilun Ding. Recent development on innovation design of reconfigurable mechanisms in china. *Frontiers of Mechanical Engineering*, 14(1):15–20, 2019.
- [104] Caihua Zhou, Bo Wang, Jiayao Ma, and Zhong You. Dynamic axial crushing of origami crash boxes. *International journal of mechanical sciences*, 118:1–12, 2016.