Computational Origami

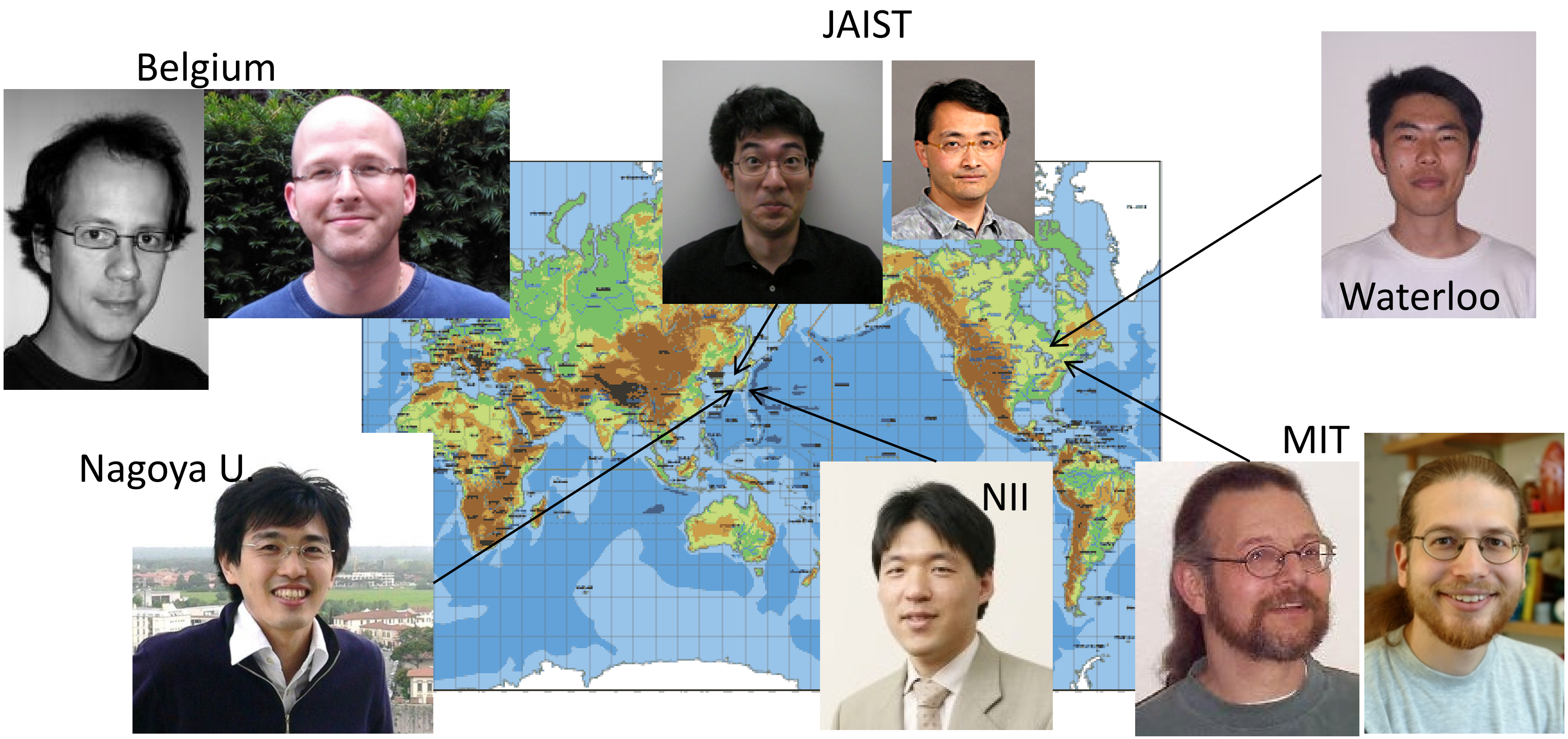
Ryuhei Uehara
Japan Advanced Institute of Science and Technology (JAIST)
School of Information Science
uehara@jaist.ac.jp
http://www.jaist.ac.jp/~uehara

Today's Topic

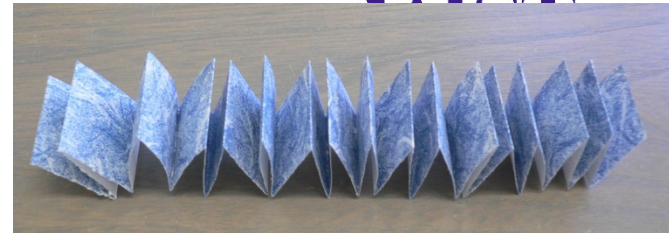
5. Time Complexity
   - "Folding complexity"
     - Theoretically, the world fastest algorithm for pleat folding
   - We can use some techniques in TCS.
     - Recursive analysis and Fibonacci sequence
     - Lower bound by counting method
6. Space Complexity (?)
   - Stamp Folding Problem
   - Minimization of Crease width
     - NP-complete problem, FPT algorithm
7. Undecidable Origami Problem
   - Diagonalization and undecidability

Reference:

J. Cardinal, E. D. Demaine, M. L. Demaine, S. Imahori, T. Ito, M. Kiyomi, S. Langerman, <u>R. Uehara</u>, and T. Uno: Algorithmic Folding Complexity, *Graphs and Combinatorics*, Vol. 27, pp. 341-351, 2011.



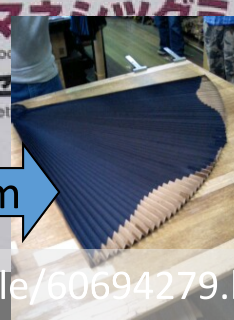
Pleat folding is…



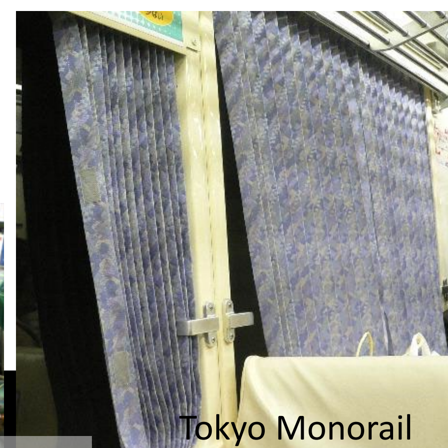
- Repeating of mountain and valley foldings
- Basic operation in some origami
- Many applications

Fold and…

steam



http://km-sewing.seesaa.net/article/60694279.html



Tokyo Monorail

Bus to JAIST

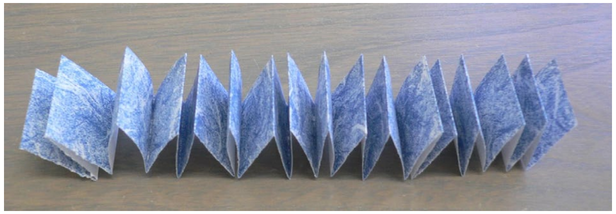
Pleat folding

Repeating “folding in half” is the best way to make many creases.

- Pleat folding (in 1D)

- Naïve algorithm: n time folding is a trivial solution
- We have to fold at least $\log n$ times to make n creases

- More efficient ways…?
- General Mountain/Valley pattern?



- proposed at Open Problem Session on CCCG 2008 by R. Uehara.
- Many people get together with some ideas.

• Complexity of Pleat F[...]

Model :
Paper has 0 thickness

[Main Motivation] Do we have to make n foldings to make a pleat folding with n creases??

1. The answer is "No"!
   - ***Any pattern*** **can be made by** $\lfloor n/2 \rfloor + \lceil \log n \rceil$ **foldings**
2. Can we make a pleat folding in $o(n)$ foldings?
   - Yes!! …it can be folded in $O(\log^2 n)$ foldings.
3. Lower bound; $\log n$
   - (We stated $\Omega(\log^2 n/\log\log n)$ lower bound for pleat folding!!)

• Complexity of Pleat Folding

[Next Motivation] What about general pleat folding problem for a given M/V pattern of length n?

- ***Any pattern*** **can be made by** $\lfloor n/2 \rfloor + \lceil \log n \rceil$ **foldings**

1. Upper bound:
   Any M/V pattern can be folded by $(4+\varepsilon)\dfrac{n}{\log n} + o\left(\dfrac{n}{\log n}\right)$ foldings

2. Lower bound:
   Almost all mountain/valley patterns require $\dfrac{n}{3+\log n}$ foldings

[Note] Ordinary pleat folding is exceptionally easy pattern!

Difficulty/Interest come from two kinds of *Parities*:
- "Face/back" determined by layers
- Stackable points having the same parity

• De

Input: Paper of length $n+1$ and a string s in $\{M, V\}^n$

Output: Well-creased paper according to s at regular intervals.

Basic operations

1. Flat {mountain/valley} fold {all/some} papers at an integer point (= simple folding)
2. Unfold {all/rewind/any} crease points (= reverse of simple foldings)

Rules

1. Each crease point remembers the last folded direction
2. Paper is rigid except those crease points

Goal: Minimize the number of folding operations

Note: We ignore the cost of unfoldings

• Upper bound of Unit FP (1)

- Any pattern can be made by $\lfloor n/2 \rfloor + \lceil \log n \rceil$ foldings

1. *M/V* fold at center point according to the assignment
2. Check the center point of the folded paper, and count the number of *M*s and *V*s (we have to take care that odd depth papers are reversed)
3. *M/V* fold at center point taking majority
4. Repeat steps 2 and 3
5. Unfold all (cf. on any model)
6. Fix all incorrect crease points one by one

Steps 1~4 require $\log n$ and step 6 requires $n/2$ foldings

Upper bound of Pleat Folding(1)

[Observation]
If $f(n)$ foldings achieve n mountain foldings,
n pleat foldings can be achieved by $2\,f(n/2)$ foldings.

The following strategy works;

- Make $f(n/2)$ mountain foldings at odd points;
- Reverse the paper;
- Make $f(n/2)$ mountain foldings at even points.

We will consider the
"**mountain folding problem**"

CF: O($n^{0.69}$) algorithm

[Th] Mountain folding problem can be solved in O($n^{0.69}$) time

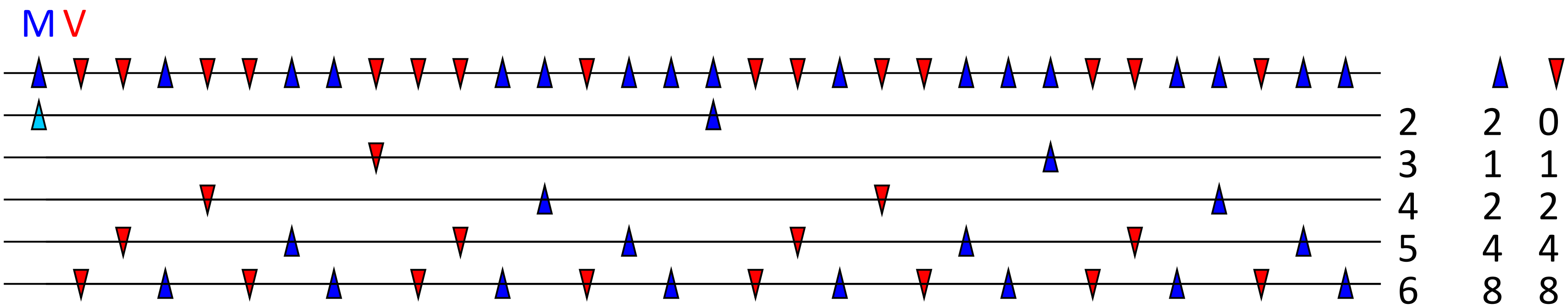
[Proof] Let n=2^k, and use the following algorithm;

1. Fold the leftmost point to make length 2^k-1

2. Fold in half at the central point

2. Repeat [2] up to length 1

4. Open all…

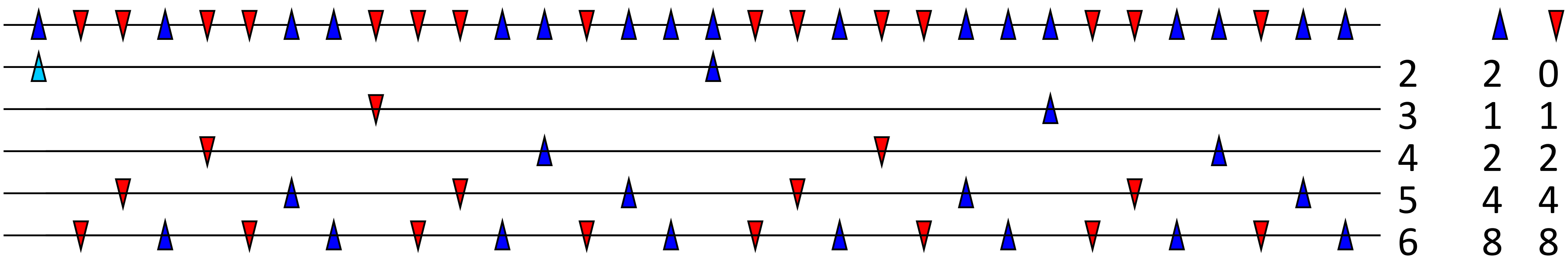


By k+1 folding, we have 2^{k-1}+1 mountains & 2^{k-1}-1 valleys

CF: O($n^{0.69}$) algorithm

[Th] Mountain folding problem can be solved in O($n^{0.69}$) time

[Proof]



2^{k-1}-1 valleys can be split into k-1 independent and uniform layers!!

$$\therefore\ f(2^k) = 1 + k + f(2^{k-2}) + f(2^{k-3}) + \cdots + f(4) + f(2) + f(1)$$

$$f(2^{k-1}) = k + \qquad f(2^{k-3}) + \cdots + f(4) + f(2) + f(1)$$

$$f(2^k) - f(2^{k-1}) = f(2^{k-2}) + 1$$

$$(f(2^k) + 1) = (f(2^{k-1}) + 1) + (f(2^{k-2}) + 1)$$

Fibonacci sequence for k !

CF: O($n^{0.69}$) algorithm

[Th] Mountain folding problem can be solved in O($n^{0.69}$) time

[Proof]

Fibonacci sequence for k !

$$(f(2^k)+1)=(f(2^{k-1})+1)+(f(2^{k-2})+1)$$

[Fibonacci sequence]

F_0=0, F_1=1, F_i=F_{i-1}+F_{i-2} (i>1)

0,1,1,2,3,5,8,13,21,34,…

$$F_i=\frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^i-\left(\frac{1-\sqrt{5}}{2}\right)^i\right)$$

Initial state:

$$f(2^0)=1, f(2^1)=2, f(2^2)=4$$

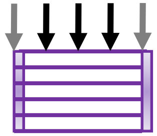
Thus $f(2^k)+1=F_{k+3}$

$$\therefore\ f(n)=f(2^k)=\frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{k+3}-\left(\frac{1-\sqrt{5}}{2}\right)^{k+3}\right)-1$$

$$=O\left(\left(\frac{1+\sqrt{5}}{2}\right)^{\log n}\right)=O\left(n^{\log\frac{1+\sqrt{5}}{2}}\right)=O(n^{0.694242})$$

□

Mountain folding in $\log^2 n$ foldings



Step 1;

1. Fold in half until it becomes of length [vvv] ($\log n$-2 foldings)
2. Mountain fold 3 times and obtain [MMM]
3. Unfold; vMMMvvvvvMMMvvvvvMMMvvvvvMMMvvvvv…

Step 2;

[MvvvvvM]

1. Fold in half until all "vvvvv"s are piled up ($\log n$-3 foldings)
2. Mountain fold 5 times [MMMMMMM], and unfold
3. vMvMMMMMMMvMvvvvvMvMMMMMMMvMvvvvvMvM

Step 3; Repeat step 2 until just one "vvvvv" remains

vMvMMMvMMMvMMMMMMMvMMMvMMMvMvvvvvMvM

Step 4; Mountain fold all irregular vs step by step.

- #iterations of Steps 2~3; $\log n$
- #valleys at step 4; $\log n$

#foldings in total~ $(\log n)^2$

• Lower bound of Unit FP

[Thm] Almost all patterns but $o(2^n)$ exceptions require $\Omega(n/\log n)$ foldings.

[Proof] A simple counting argument:

- # patterns with n creases > $2^n/4 = 2^{n-2}$ ({surface/reverse}×{front/back})
- # patterns after k foldings <

$$(2\times n)\times(n+1)\times(2\times n)\times(n+1)\times\ldots\times(n+1)\times(2\times n)$$

M/V, Position, Possible unfoldings

$$<(2n(n+1))^k$$

- We cannot fold most patterns after at most k foldings if

$$\sum_{i=0}^{k}(2n(n+1))^i \le (2n(n+1)+1)^k < 2^{n-2}$$

- Letting

$n \ge 2, k = O\left(\frac{n}{\log n}\right)$ we have $(2n(n+1)+1)^k = o(2^n)$ □

Any pattern can be folded in $cn/\log n$ foldings

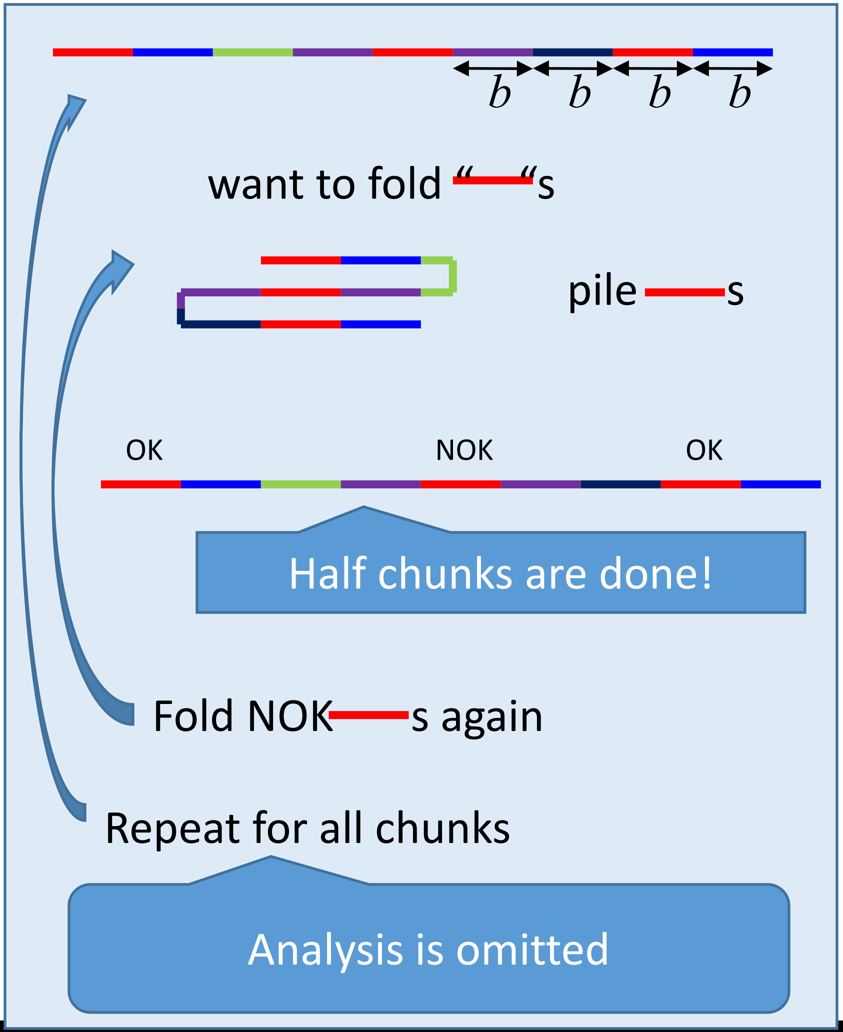
Select suitable b depending on n.

Prelim.

- Split into *chunks* of size b;
  1. Each chunk is small and easy to fold
  2. #kinds of different bs are not so big

Main alg.

- For each possible b
  1. pile the chunks of pattern b and mountain fold them
  2. fix the reverse chunks
  3. fix the boundaries



• Open Problems

- Pleat foldings
  - Make upper bound $O(\log^2 n)$ and lower bound $\Omega(\log^2 n / \underline{\log\log n})$ closer
- "**Almost all** patterns are difficult", but…
  - **<u>No explicit M/V pattern</u>** that requires $(cn/\log n)$ foldings
- When "unfolding cost" is counted in…
  - Minimize #foldings + #unfoldings

- Extension to 2 dimensional and general intervals