# 本講義のトピック

## そのA:展開図とそこから折れる凸立体の研究

- 展開図と立体のとても悩ましい関係:最大の未解決問題
- 与えられた「展開図」を折って作れる(凸)「立体」をどうやって計算するか？
  - 数学的な特徴づけ/アルゴリズム/計算パワー

## そのB:「折り」のアルゴリズムと計算量の関係

- 折り紙の基本操作 済
- 折り紙のアルゴリズムと計算量
  - 1次元の紙における効率のよい折り方(アルゴリズムと計算量)
    - 高速に折るアルゴリズム(折る回数を減らせるか？)
    - 「良い折り畳み状態」を評価する指標のモデル
  - 1次元の紙における計算不能性(計算の理論)
    - 計算モデル

# そのB:「折り」のアルゴリズムと計算量の関係

5. <u>時間計算量</u>
   - “Folding complexity”入門
     - 理論上、世界最速のジャバラ折りアルゴリズム
   - 意外とアルゴリズムの解析（組合せ論）と相性が良い！
     - 再帰解析とフィボナッチ数
     - 数え上げ手法による下界

6. 領域計算量（？）
   - 切手折り問題
   - 折り目幅最小化問題
     - NP完全問題、FPTアルゴリズムなど

7. （折り紙における決定不能問題）
   - 対角線論法と不完全性定理

# じゃばらを高速に折る
## --- folding complexity ---

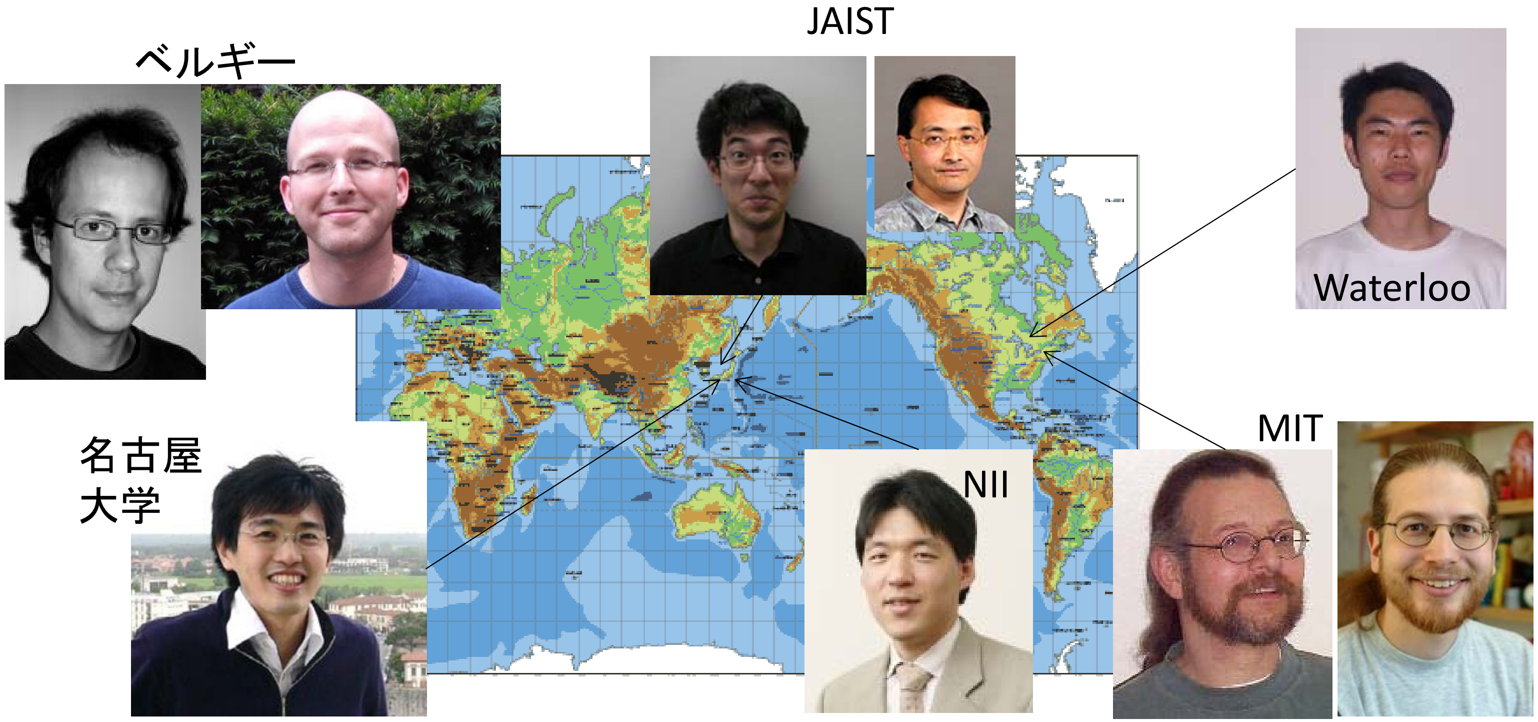
上原隆平（うえはらりゅうへい）
北陸先端科学技術大学院大学　情報科学研究科
Japan Advanced Institute of Science and Technology (JAIST)
uehara@jaist.ac.jp
http://www.jaist.ac.jp/~uehara

主要な論文

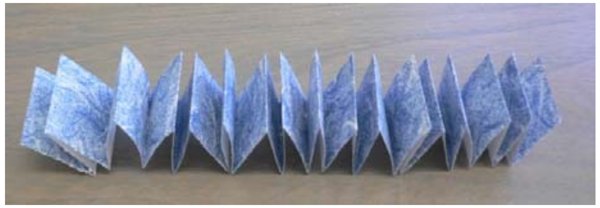
J. Cardinal, E. D. Demaine, M. L. Demaine, S. Imahori, T. Ito, M. Kiyomi, S. Langerman, R. Uehara, and T. Uno: Algorithmic Folding Complexity, *Graphs and Combinatorics*, Vol. 27, pp. 341-351, 2011.

1. 時間…折り(基本演算)の回数

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




# じゃばら折り = 1次元折り紙



- 山折りと谷折りが等間隔に交互に続く
- 折り紙の基本ツール
- 応用も…?

➢山/谷を一般化したときの複雑さとは?

「(M/Vによる)2進数文字列上の操作の問題」と考えると、コンピュータサイエンスと相性がよい

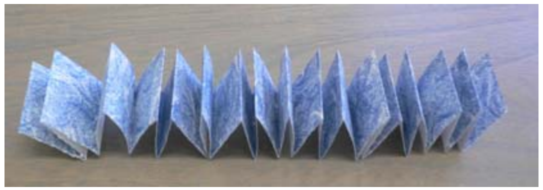

# じゃばら折り

- じゃばら折り(1次元折り紙)

- 単純な方法： $n$ 回折ればよい
- $n$ 個の折り目をつけるには $\log n$ 回は折らないといけない

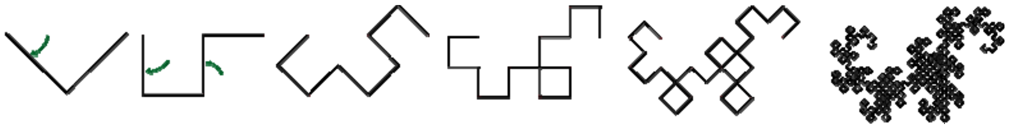
「中央で半分に折る」ことが
もっとも多くの折り目を
つけることができる

- もっと効率よく折り目をつけることはできるのか…?
- 一般のM/Vパターンに対してはどうか？



- CCCG 2008 の "Open Problem Session" にて
  上原が提案（Ron Graham が絶賛してくれた!!）
- 何度かの発表と結果の改善を経て多くの共著者を
  得て論文に至る

ドラゴン曲線



# じゃばら折りの複雑さ

モデル：紙の厚みは0

**[最大の疑問]** $n$ 個の折り目からなるじゃばら折りは
$n$ 回折らなければ作れないのか？

最初の「大発見」であり，
研究を始めるきっかけと
もなるアルゴリズム！

1. 答えは“No”!
   - **どんなM/Vパターンでも高々 $\lfloor n/2 \rfloor + \lceil \log n \rceil$ 回折れば作れる！**
   - では $o(n)$ 回ではどうか?
   - Yes!! …わずか $O(\log^2 n)$ 回の折りで作る方法がある

2. 下界; $\log n$
   - じゃばら折りは $\Omega(\log^2 n / \log\log n)$ 回未満の折りでは作れない!!

# 一般化じゃばら折りの複雑さ

証明技法が面白い

**[次の疑問]** 与えられた長さ $n$ の任意の M/V パターンを折る問題の複雑さはどうか？

□ **<u>どんなM/Vパターンでも高々 $\lfloor n/2 \rfloor + \lceil \log n \rceil$ 回折れば作れる!!</u>**

1. 上界:
どんなM/Vパターンも $(4+\varepsilon)\dfrac{n}{\log n} + o\left(\dfrac{n}{\log n}\right)$ 回の折りで作れる

計算量のオーダーは同じ

2. 下界:
ほぼすべてのM/Vパターンは $\dfrac{n}{3+\log n}$ 回以上折らないと作れない

[つまり] じゃばら折りは例外的に簡単なパターンであった！

# • 定義：単位長折り問題

2種類のパリティがあるところが{難しい|面白い}
- “表/裏”は層のパリティで決まる
- 同じパリティでないと重ねて折れない

入力: 長さ $n+1$ の紙と $\{M, V\}^n$ の文字列 $s$
出力: $s$ にしたがって等間隔に折り目のついた紙

<u>基本操作</u>

1. 整数点で{一部/全部}の紙を{山/谷}折りにして平坦にする(=単純折りモデル)
2. {すべて/任意/直前}の折り目で開く(=単純折りの逆操作)

<u>規則(仮定)</u>

1. それぞれの折り目は<u>最後に折られた方向を記録している</u>
2. 紙は折り目を除いて剛体

<u>目標</u>: <u>折り操作の回数の最小化</u>

注意: 紙を開く操作のコストは気にしない

- 単位長折りの上界(1)

□ どんなパターンも $\lfloor n/2 \rfloor + \lceil \log n \rceil$ 回で折れる

1. 指定にしたがって紙を中心で半分に折る
2. 折られた紙の中心を見て、$M$と$V$の多数決をとる (裏返った紙に注意)
3. <u>多数決に従って中心で半分に折る</u>
4. 2と3を繰り返す
5. 全部広げる
6. 間違った折り目を逐一直す

ステップ1~4の折りは $\log n$ 回でステップ6は高々 $n/2$ 回の折り

□

# じゃばら折りの上界(1)

[観測]
もし $f(n)$ 回の折りで「$n$ 個の山折り」が作れるなら、
$n$ 個のじゃばら折りは $2\,f(n/2)$ 回の折りで作れる

以下の方法でよい:

- 偶数点だけに着目して $f(n/2)$ 回折って全部山折りにする;
- 紙を裏返す;
- 奇数点だけに着目して $f(n/2)$ 回折って全部山折りにする.

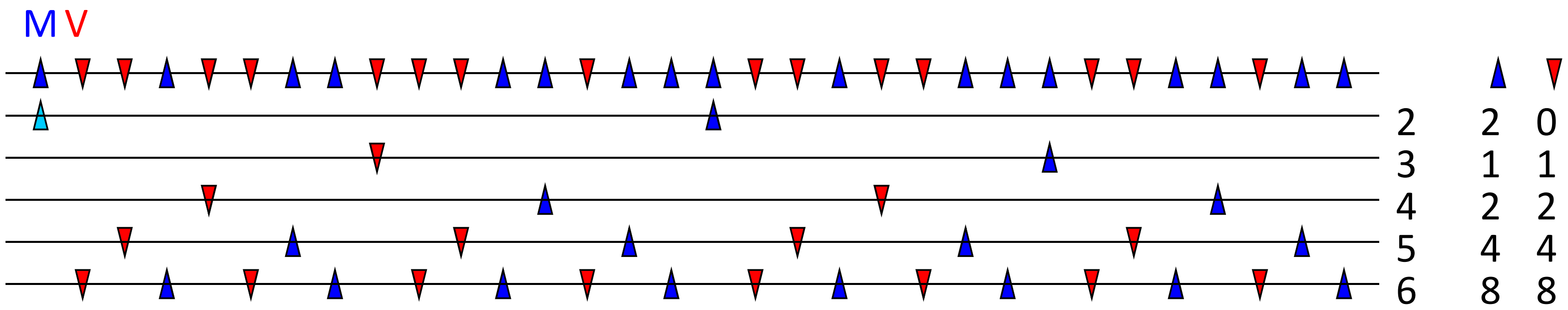
よって以下では「山折り問題」を考える

# 参考:「山折り問題」のO($n^{0.69}$)アルゴリズム

## [定理] 山折り問題を解くO($n^{0.69}$) アルゴリズム

[証明] $n=2^k$ として以下のアルゴリズムを使う

1. 左端を山折りにして、長さを $2^k$-1 にする
2. 中央で山折りにする
3. 長さ1になるまで2を繰り返す
4. 全部開いて…

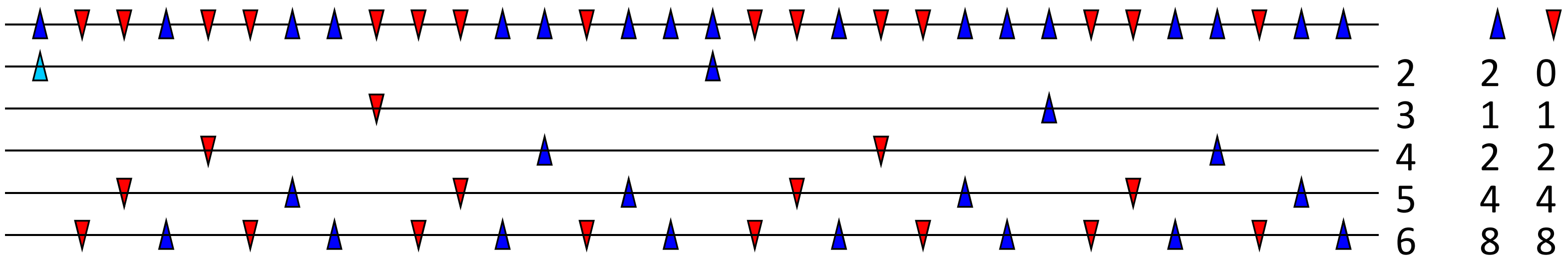



$k$+1回折って $2^{k\text{-}1}$+1 個の山と $2^{k\text{-}1}$-1 個の谷ができる

# 参考:「山折り問題」の$O(n^{0.69})$アルゴリズム

## [定理] 山折り問題を解く$O(n^{0.69})$ アルゴリズム

[証明]




**$2^{k\text{-}1}$-1 個の谷は独立で等間隔な $k$-1 個の層に分けられる!!**
**それぞれ独立に再帰的に解ける！**

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

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

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

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

$k$ に関するフィボナッチ数列！

# 参考:「山折り問題」のO($n^{0.69}$)アルゴリズム

## [定理] 山折り問題を解くO($n^{0.69}$) アルゴリズム

[証明]

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

$k$ に関するフィボナッチ数列！

[フィボナッチ数列]

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

0,1,1,2,3,5,8,13,21,34,…

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

初期条件:

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

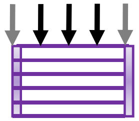
よって

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

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

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

# 山折り問題を $\log^2 n$ 回の折りで解くアルゴリズム



ステップ1;

1. 「半分に折る」を繰り返して以下をえる ($\log n$-2 回の折り)：[vvv]
2. 3回山折りして以下を得る：[MMM]
3. 開く; vMMMvvvvvMMMvvvvvMMMvvvvvMMMvvvvv...

ステップ2;

[MvvvvvM]

1. すべての “vvvvv”が重なるように半分折りを繰り返す($\log n$-3 回折る)
2. 5回山折りして以下を得る：[MMMMMMM]
3. 開く; vMvMMMMMMMvMvvvvvMvMMMMMMMvMvvvvvMvM

ステップ3; “vvvvv” が一つ残るまでステップ 2 を繰り返して以下を得る：

vMvMMMvMMMvMMMMMMMvMMMvMMMvMvvvvvMvM

ステップ4; すべてのとびとびの v を一つずつ折る.

- ステップ2~3の繰り返し回数; $\log n$
- ステップ4での v の個数; $\log n$

折りの回数のトータル~ $(\log n)^2$

# 単位長折り問題の下界

[定理] o($2^n$)個の例外を除くほぼすべてのパターンは

$\Omega(n/\log n)$ 回折らないと作れない

{表/裏}×{前/後}

[証明] 単純な**数え上げ法**による:

- $n$ 個の折り目のパターンの数 $> 2^n/4 = 2^{n-2}$
- $k$ 回折ったことで作れるパターンの数 <

山/谷

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

位置

可能な展開

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

- よって以下が成立する場合、高々 $k$ 回の折りではすべてのパターンは作れない: $\sum_{i=0}^{k}(2n(n+1))^i \le (2n(n+1)+1)^k < 2^{n-2}$

ここで $n \ge 2, k = O\left(\frac{n}{\log n}\right)$ とおくと $(2n(n+1)+1)^k = o(2^n)$ をえる。

□

# 任意のパターンを $cn/\log n$ 回の折りで作るアルゴリズム（概要）

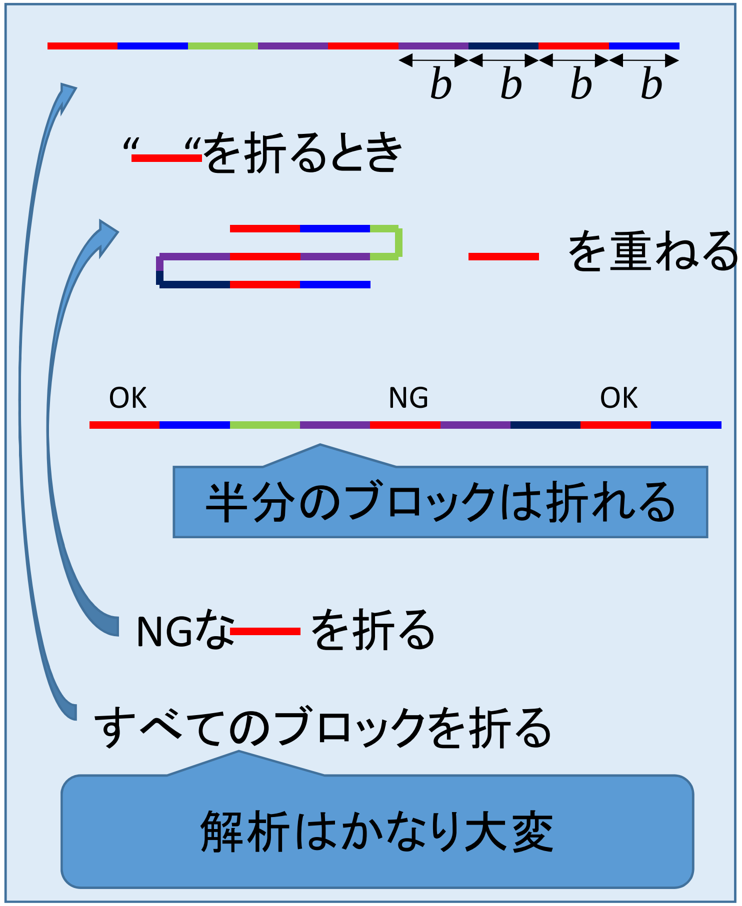
$n$に応じて適切に $b$ を選ぶ.

## 準備:

- パターンを大きさ$b$のブロックに分割;
  1. 各ブロックは $O(n/\log n)$ 回で折れる
  2. $2^b$ はそれほど大きくない

## アルゴリズム:

- 可能なそれぞれのブロックパターンに対して
  1. 同じパターンをもつブロックが重なるように半分折りする
  2. 裏返っているブロックを直す
  3. 重ねるために折った所を直す



# 未解決問題

- じゃばら折り
  - 上界 $O(\log^2 n)$ と下界 $\Omega(\log^2 n / \underline{\log\log n})$ を近づける
- 「ほぼすべてのパターンは難しい」と言うけれど…
  - $(cn/\log n)$ 回の折りが本当に必要な**<u>具体的な M/V パターン</u>**の構成方法はわかっていない
- 紙を開くコストは無視しているけど…
  - 「折る回数」＋「開く回数」を最小化するとよいかも
  - (たかだか折った回数しか開けないけど…)
- 一般の間隔や2次元への拡張も…