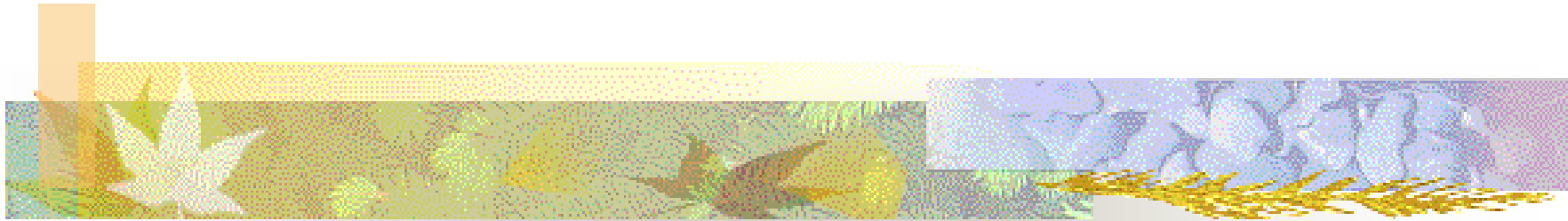


# 大規模知識処理特論 ガイダンス



北海道大学 情報科学研究院

堀山 貴史

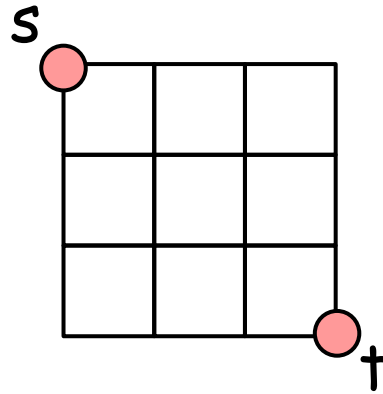
脊戸 和寿

# 大規模知識処理特論

- この授業の目標
  - 知識の編集・分類・解析・索引化等の知的な情報処理を行うために不可欠な知識処理の技法について学びます
- 大規模知識処理について、以下の観点から並行して or 順に 学習を進めます
  - 最適化技法
  - 論理関数と計算量理論の基礎
  - 厳密アルゴリズムと近似アルゴリズム
  - BDD/ZDDによる離散構造処理
- 適当な時期にレポート課題を課します

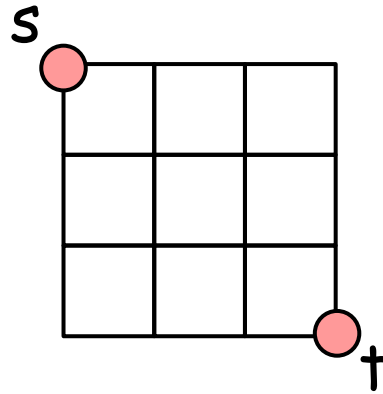
# Quiz : $s$ - $t$ パス (最短路)

Q :  $s$  から  $t$  への最短路を列挙しなさい (何通り?)



# Quiz: s-t パス (最短路)

Q: s から t への最短路を列挙しなさい (何通り?)



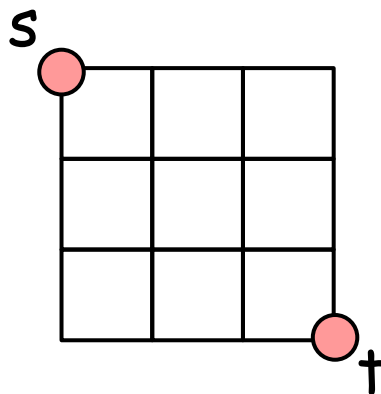
↓↓↓ → → → の組合せで、1 つの最短路

$$\text{最短路は、} {}_6C_3 = \frac{6 \cdot 5 \cdot 4}{3 \cdot 2 \cdot 1} = 20 \text{ 通り}$$

# Quiz: s-t パス ~~(最短路)~~

Q: s から t への経路を列挙しなさい (何通り?)

- 遠回り ok
- 同じ点を通るのは NG



- 何通りかを求める公式は、ありません

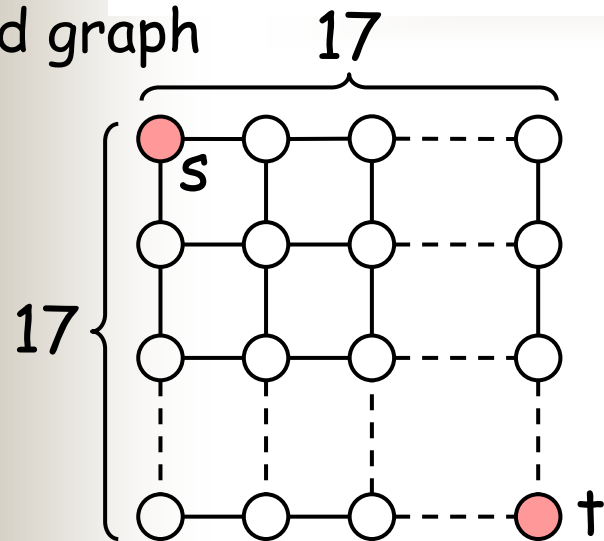
(ちなみに、答えは 184 通り)

# 列挙：条件を満たすものを全部

- 求められること
  - 効率的に列挙したい (時間計算量)
  - 結果をコンパクトに持ちたい (領域計算量)
  - 結果を簡単に利用したい
    - 何個ある？ / サンプルングしたい
    - 後で条件を色々変えて、結果を検索したい

# 例) s-t パス (同じ点を2回以上 通らない)

Grid graph



s-t パスの個数 6,344,814,  
611,237,963,971,310,297,540,  
795,524,400,449,443,986,866,  
480,693,646,369,387,855,336  
約  $6.3 \times 10^{61}$  (63 那由他)

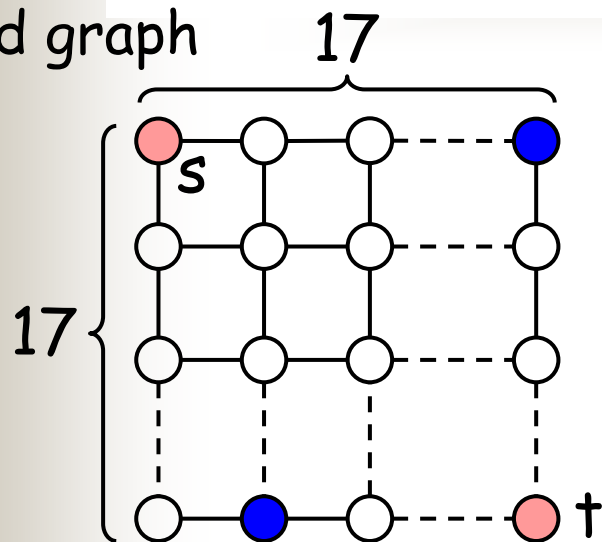
サンスクリット語で  
“十分に大きな数”

## ■ 求められること

- **効率的に**列挙したい (時間計算量)
- 結果を**コンパクト**に持ちたい (領域計算量)
- 結果を**簡単に**利用したい
  - 何個ある? / サンプルングしたい
  - 後で条件を色々変えて、結果を検索したい

# 例) s-t パス (同じ点を2回以上 通らない)

Grid graph



s-t パスの個数 6,344,814,  
611,237,963,971,310,297,540,  
795,524,400,449,443,986,866,  
480,693,646,369,387,855,336  
約  $6.3 \times 10^{61}$  (63 那由他)

## ■ 求められること

- **効率的に** 列挙したい (時間計算量)
- 結果を**コンパクト**に持ちたい (領域計算量)
- 結果を**簡単に**利用したい
  - 何個ある? / サンプルを抽出したい
  - 後で条件を色々変えて、結果を検索したい

サンスクリット語で  
“十分に大きな数”

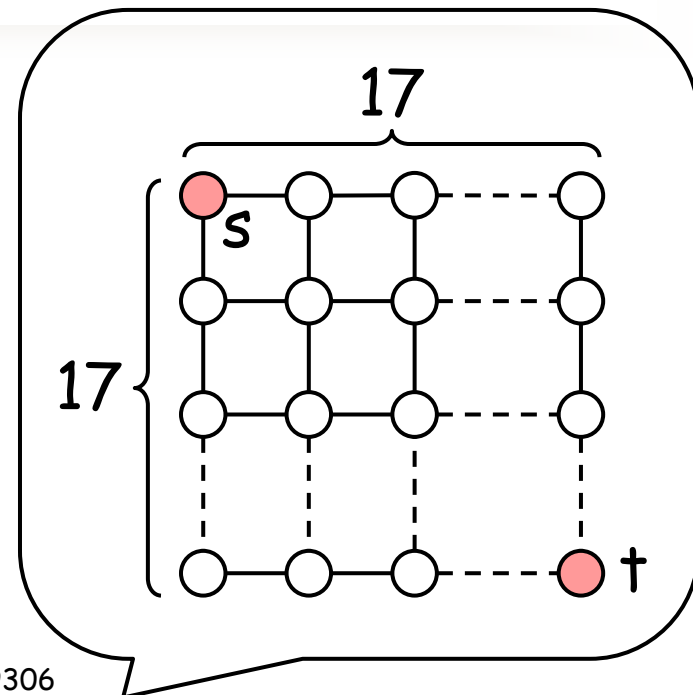
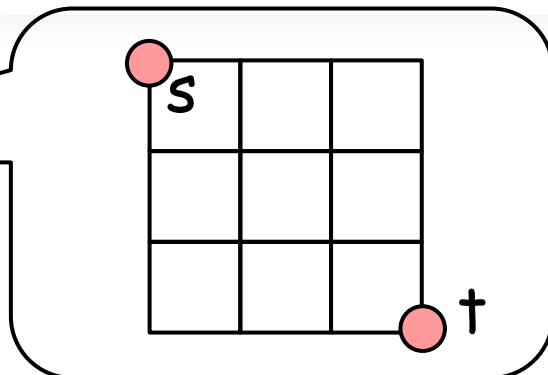
長さ50以下の  
パスは何個?

最長の  
パスは?

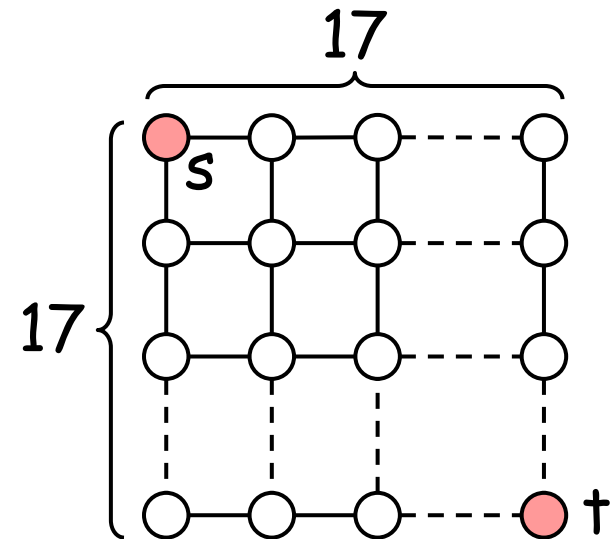
● を2つとも通る  
パスは何個?

# s-† パスは、何通り？

- 1 2
- 2 12
- 3 184
- 4 8512
- 5 1262816
- 6 575780564
- 7 789360053252
- 8 3266598486981642
- 9 41044208702632496804
- 10 1568758030464750013214100
- 11 182413291514248049241470885236
- 12 64528039343270018963357185158482118
- 13 69450664761521361664274701548907358996488
- 14 227449714676812739631826459327989863387613323440
- 15 2266745568862672746374567396713098934866324885408319028
- 16 68745445609149931587631563132489232824587945968099457285419306
- 17 6344814611237963971310297540795524400449443986866480693646369387855336
- 18 1782112840842065129893384946652325275167838065704767655931452474605826692782532
- 19 1523344971704879993080742810319229690899454255323294555776029866737355060592877569255844
- 20 3962892199823037560207299517133362502106339705739463771515237113377010682364035706704472064940398



# s-† パスは、何通り？



21  
3137475105013710272042053813738221451310331219369872365306135199134643337938938579396557699224602131646  
3868

22  
755970286667345339661519123315222619353103732072409481167391410479517925792743631234987038883317634987  
271171404439792

23  
5543542935523747700991431848906143793069037997096433133255695864648400840733488554456638692402087571  
1242060085408513482933945720

24  
123717122312070647583387448626735708323730419890129435396787270804849516955159304856413945507921530371  
91858028212512280926600304581386791094

25  
84029748578811334710070837454368091272960542937753835498247426239370284978982152569291785770839709601  
21625602506027316549718402106494049978375604247408

26  
173699315862792729311754404212364989003722295882881406046637037209103424132761347627892181934980061070  
82296223143380491348290026721931129627708738890853908108906396

# 大規模知識処理アルゴリズム

- **問題の解き方** (ざっくり言うと 作戦) を考える
- どうして アルゴリズムは **重要** ?
  - **実装**を頑張って、スピードが **2倍, 3倍** ...  
**アルゴリズム**を変えると **1000倍** ...
    - より大規模/実用的な対象が扱える
  - **基礎理論**が、様々な**応用分野**とつながる
    - 例) ■ 「選挙区の区割り」と「避難所の割当て」
    - 「展開図」と「細胞で折り紙」
    - など

# 選挙区の区割り

大阪府だと、489穰3281杼3930垓3925京  
22兆5190億1210万1206通りの区割り  
(普通にやると、ハードディスクに入らない)  
0.34秒で求める

## 一票の格差

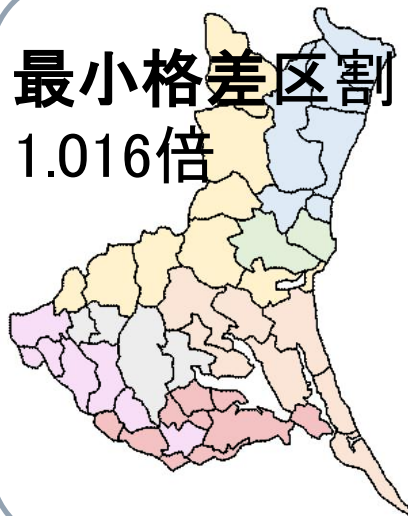
例: 茨城県

41市区郡  
7選挙区

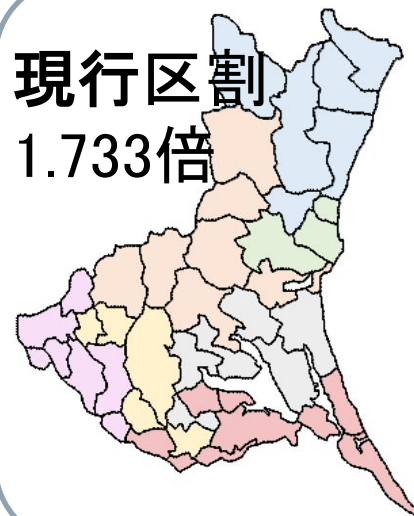


国勢調査の結果をもとに、  
格差が小さくなるように、区割りをする

最小格差区割  
1.016倍

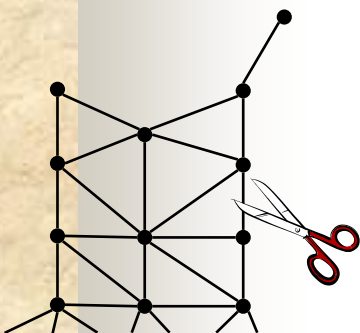


現行区割  
1.733倍

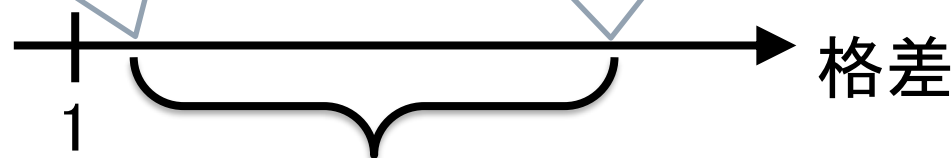


問題の取り組み方

- ・ 一気に考えるのは大変
- ・ 少しずつ小さな目標に分ける



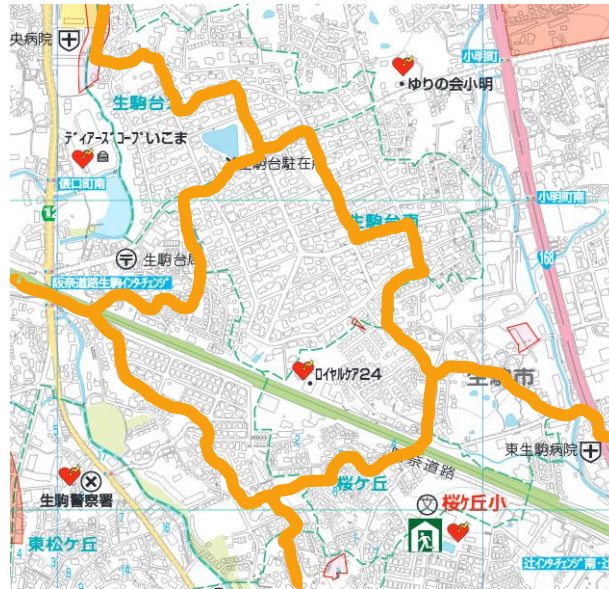
たとえば、  
グラフを切り分ける



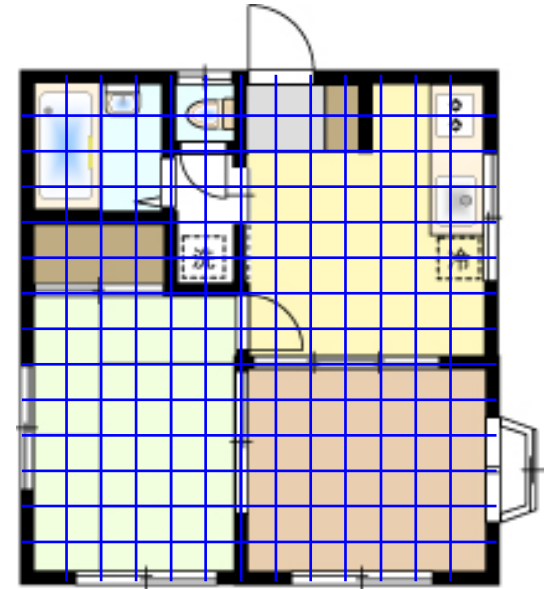
この間の様子はどうなっているのか  
解は多いのか、少ないのか

# 選挙区の区割りの応用

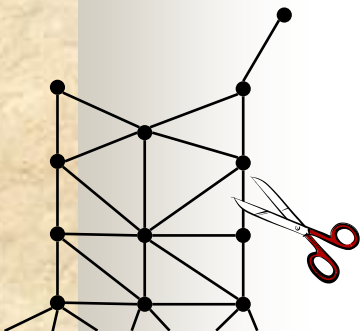
- 同様のアイデアが、他の分野にも応用できる



被災時の避難計画  
(避難場所の割当)



フロアプラン  
(間取り図)



たとえば、  
グラフを切り分ける

# Folding Mechanism (折りメカニズム)



商品のパッケージ

# Folding Mechanism (折りメカニズム)



商品のパッケージ



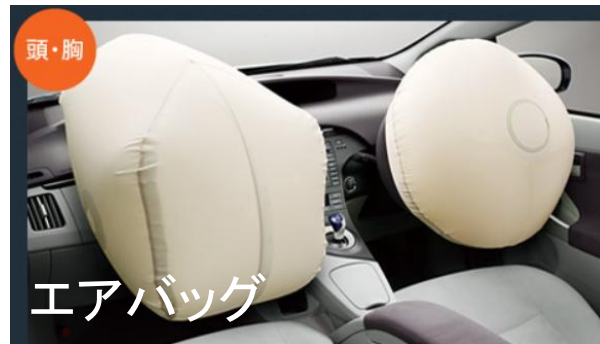
人工衛星のパネル  
(ミウラ折り)

建築

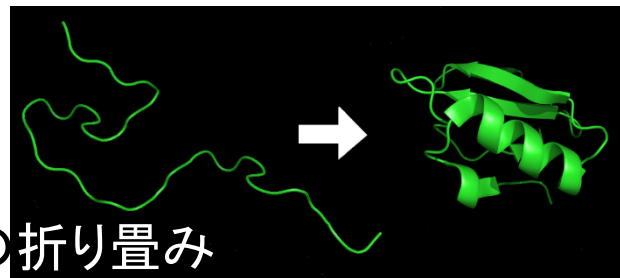


缶飲料

(ヨシムラ パターン)



エアバッグ



タンパク質の折り畳み

## Origami 研究に、大型予算も

### Algorithmic design of self-folding polyhedra

Shivendra Pandey<sup>a</sup>, Margaret Ewing<sup>b</sup>, Andrew Kunaš<sup>c</sup>, Nghi Nguyen<sup>d</sup>, David H. Gracias<sup>a,e,1</sup>, and Govind Menon<sup>f,1</sup>

<sup>a</sup>Department of Chemical and Biomolecular Engineering, The Johns Hopkins University, Baltimore, MD 21218; <sup>b</sup>School of Mathematics, University of Minnesota, Minneapolis, MN 55455; <sup>c</sup>Department of Computer Science, Brown University, Providence, RI 02912; <sup>d</sup>Department of Mathematics and Statistics, University of Massachusetts, Amherst, MA 01003; <sup>e</sup>Department of Chemistry, The Johns Hopkins University, Baltimore, MD 21218; and <sup>f</sup>Division of Applied Mathematics, Brown University, Providence, RI 02906

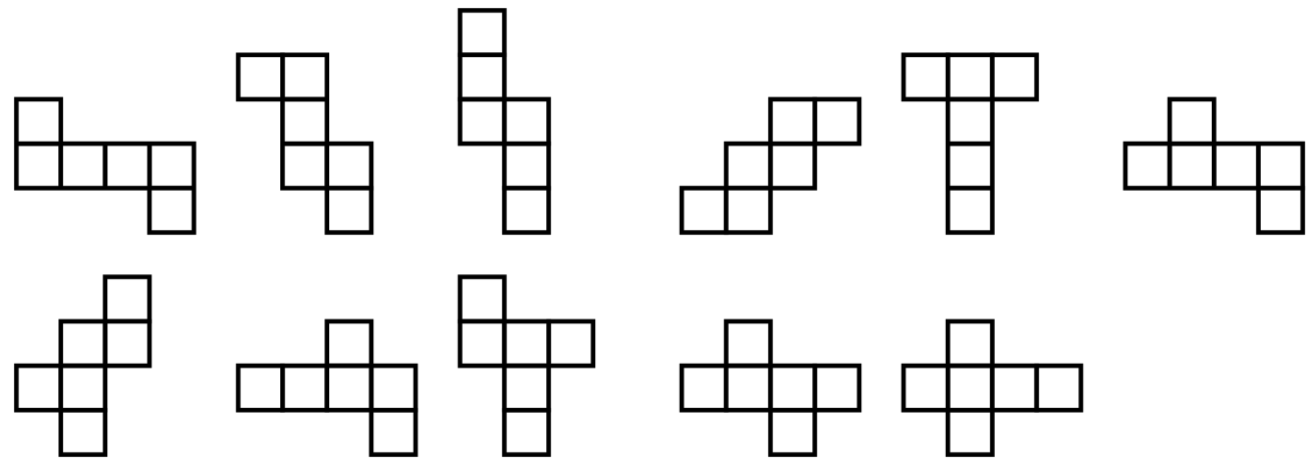
Edited by Ken A. Dill, Stony Brook University, NY

Self-assembly has emerged as a paradigm for the rationalization of complex three-dimensional structures. A few principles that guide a priori design of self-assembling structures. We extend this theory to the geometric principles that govern the self-assembly of submillimeter-scale higher polyhedra from flat sheets. In particular, we computationally search for the most compact and then test these nets in the laboratory. Our findings are that (i) compactness is a simple principle for maximizing the yield of self-assembly, (ii) shortest paths from 2D nets to 3D polyhedra are important for rationalizing folding pathways. Our work provides a method for experimental and theoretical analysis of self-assembly pathways in self-assembly.

microfabrication | origami | programmable materials

# 展開図の研究

サイコロの展開図は 11 種類



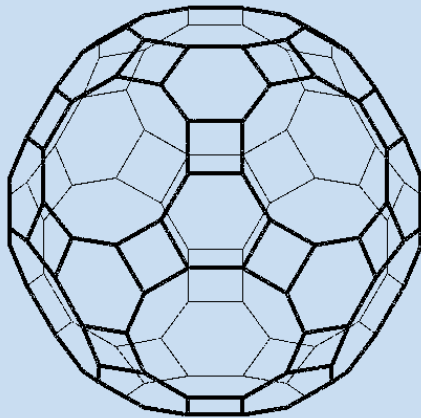
Q.2 サッカーボールの展開図は何種類？



A. 3,127,432,220,939,473,920  
(312京 7432兆 2209億 3947万 3920)

# 展開図の研究

サイコロの展開図は 11 種類



181,577,189,197,376,045,  
928,994,520,239,942,164,480

181澗 5771溝 8919穰 7376秭 459垓  
2899京 4520兆 2399億 4216万 4480



A. 3,127,432,220,939,473,920  
(312京 7432兆 2209億 3947万 3920)

# この授業では： 離散構造の観点から

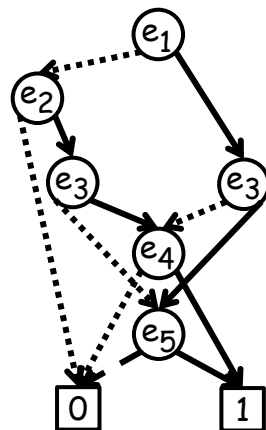
- アルゴリズムで、社会を支える  
(列挙/最適化 アルゴリズムを中心に)
- 社会とのつながりは色々

## 離散構造

離散構造 (組合せ、グラフなど) を  
どう表し、どう扱うか

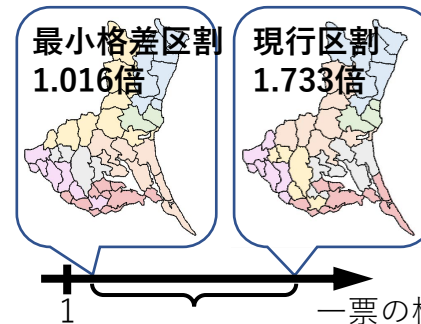
離散構造の表現  
構築アルゴリズム

$$\left\{ \begin{array}{l} \{e_1 e_3 e_5\} \\ \{e_2 e_3 e_4\} \\ \{e_1 e_4\} \\ \{e_2 e_5\} \end{array} \right\}$$

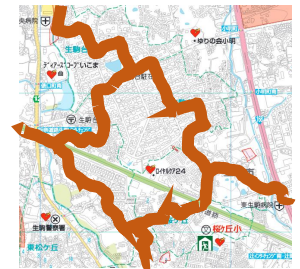


## 分野横断的なつながり

選挙区の区割り



被災時の避難計画



避難場所の割当て

計算折り紙

細胞の形成への応用

