離散数理工学 第 2 回
数え上げの基礎：漸化式の立て方

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2015 年 10 月 20 日

最終更新：2015 年 10 月 18 日　15:29

スケジュール 前半 (予定)

1. 数え上げの基礎：二項係数と二項定理 (10/6)
- ⋆ 休講 (体育祭) (10/13)
2. 数え上げの基礎：漸化式の立て方 (10/20)
3. 数え上げの基礎：漸化式の解き方 (基礎) (10/27)
- ⋆ 祝日で休み (11/3)
4. 数え上げの基礎：漸化式の解き方 (発展) (11/10)
5. 離散代数：整数と有限体 (11/17)
6. 離散代数：多項式環 (11/24)
7. 離散代数：多項式環による有限体の構成 (12/1)
8. 離散代数：有限体の応用 (12/8)

注意：予定の変更もありうる

スケジュール 後半 (予定)

9. 離散確率論：確率の復習と確率不等式 (12/15)
- ⋆ 中間試験 (12/22)
10. 離散確率論：確率的離散システムの解析 (1/5)
11. 離散確率論：乱択データ構造とアルゴリズム (基礎) (1/12)
12. 離散確率論：乱択データ構造とアルゴリズム (発展) (1/19)
13. 離散確率論：マルコフ連鎖 (基礎) (1/26)
14. 離散確率論：マルコフ連鎖 (発展) (2/2)
- ⋆ 予備日 (2/9)
- ⋆ 期末試験 (2/16?)

注意：予定の変更もありうる

今日の目標

今日の目標

漸化式を立てられるようになる
- 組合せ構造の数え上げ
- アルゴリズムの計算量

格言

アルゴリズムの計算量解析の基礎は数え上げ

目次

1. 組合せ構造の数え上げ
   グラフにおける独立集合の数え上げ
2. アルゴリズムの計算量
3. 今日のまとめ

無向グラフ

無向グラフとは？

無向グラフとは，順序対 (V, E) で，
- V は集合
- $E \subseteq 2^V$ は V の 要素数 2 の部分集合の集合

であるもののこと

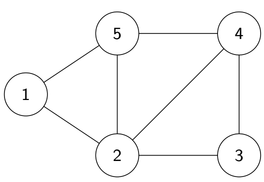
例：
- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$

注意

$\{2, 5\} = \{5, 2\}$　(集合では順序を不問)

無向グラフの図示

- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$



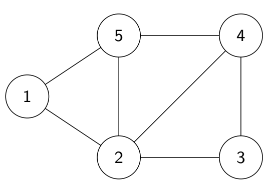
無向グラフの用語

無向グラフ $G = (V, E)$

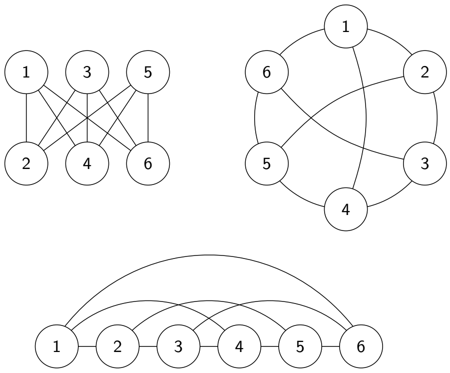
無向グラフの用語

- V の要素を G の頂点と呼ぶ
- E の要素を G の辺と呼ぶ
- V を G の頂点集合と呼ぶ
- E を G の辺集合と呼ぶ
- 辺 $\{u, v\} \in E$ に対して，u, v をその端点と呼ぶ
- 頂点 v が辺 e の端点であるとき，v は e に接続するという
- 頂点 u と v が辺を成すとき，u と v は隣接するという

- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\},$ $\{2, 5\}, \{3, 4\}, \{4, 5\}\}$
- 頂点 2, 3 は辺 $\{2, 3\}$ の端点
- 頂点 2 は辺 $\{2, 3\}$ に接続する
- 頂点 2 と頂点 3 は隣接する



1 つのグラフに対するいろいろな図示



用語に関する注意

無向グラフ

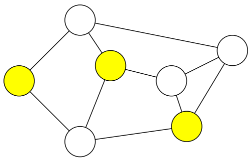
- 「頂点」の別名：「節点」,「ノード」,「点」
- 「辺」の別名：「枝」,「エッジ」

独立集合

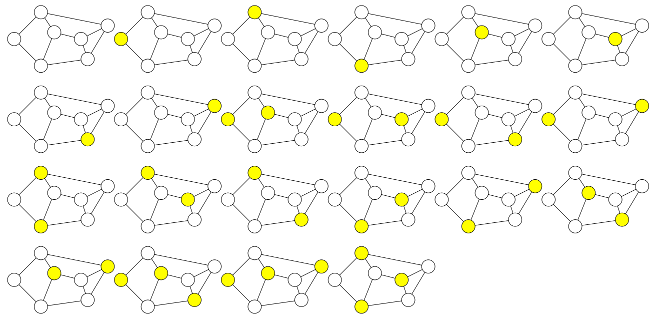
無向グラフ $G = (V, E)$

独立集合とは？

G の独立集合とは，頂点部分集合 $I \subseteq V$ で，
任意の異なる 2 頂点 $u, v \in I$ に対して $\{u, v\} \notin E$



すべての独立集合 (独立集合全体)

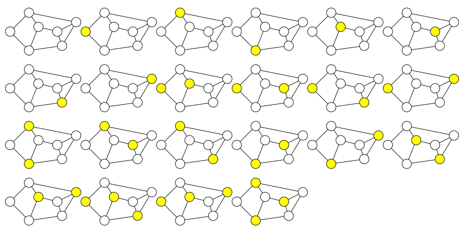


22 個

目標

やりたいこと

与えられた無向グラフにおける独立集合の数を計算したい

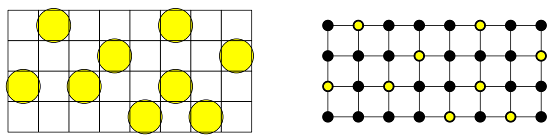


22 個

目標：なぜ計算したい？

統計力学における「ハードコア格子気体模型」

- 系を無向グラフ $G = (V, E)$ としてモデル化する
- 各 $v \in V$ が状態 $\sigma_v \in \{0, 1\}$ を持つ
  - $\sigma_v = 0 \Leftrightarrow v$ に気体分子が存在しない
  - $\sigma_v = 1 \Leftrightarrow v$ に気体分子が存在する
- $\sigma_v = 1$ となる $v \in V$ の集合が独立集合である
  $\Leftrightarrow$ 気体分子同士が重なり合わない
- 系において許される状態の総数 $=$ 独立集合の総数
- $\rightsquigarrow$ 系の分配関数の計算 $\rightsquigarrow$ 系の振舞いのシミュレーション



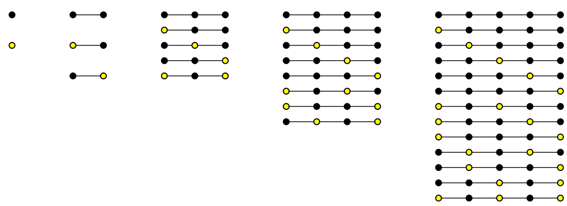
例：道

道と呼ばれる無向グラフ

目標

グラフ P_n における独立集合の総数を計算する

例：道 — 手でやってみる

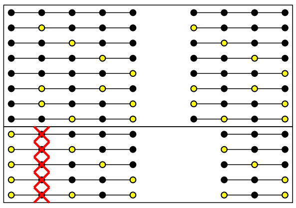


n	1	2	3	4	5
独立集合の総数	2	3	5	8	13

例：道 — 系統立ててやってみる

グラフ P_5 を考えると，独立集合は次の 2 種類

- (A) 左端の頂点を要素として含まないもの
  = 左端の頂点を除去してできる P_4 の独立集合
- (B) 左端の頂点を要素として含むもの
  = 左側の 2 頂点を除去してできる P_3 の独立集合 ∪{ 左端の頂点 }

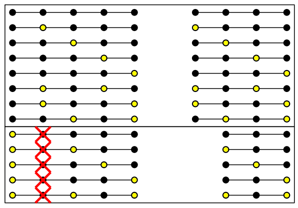


つまり，
P_5 の独立集合の総数 = P_4 の独立集合の総数 + P_3 の独立集合の総数

例：道 — 系統立ててやってみる (一般化)

グラフ P_n を考えると，独立集合は次の 2 種類 (ただし，$n \geq 3$)

- (A) 左端の頂点を要素として含まないもの
  = 左端の頂点を除去してできる P_{n-1} の独立集合
- (B) 左端の頂点を要素として含むもの
  = 左側の 2 頂点を除去してできる P_{n-2} の独立集合 ∪{ 左端の頂点 }



つまり，$n \geq 3$ のとき，
P_n の独立集合の総数 = P_{n-1} の独立集合の総数 +
P_{n-2} の独立集合の総数

例：道 — まとめ

a_n = グラフ P_n における独立集合の総数　とする

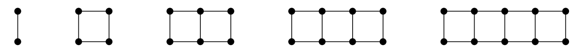
漸化式

$$a_n = \begin{cases} 2 & (n = 1 \text{ のとき}) \\ 3 & (n = 2 \text{ のとき}) \\ a_{n-1} + a_{n-2} & (n \geq 3 \text{ のとき}) \end{cases}$$

これを解くのは次回

例：$P_n \times P_2$

次のグラフを考える (G_n と書くことにする)



G_1　G_2　G_3　G_4　G_5

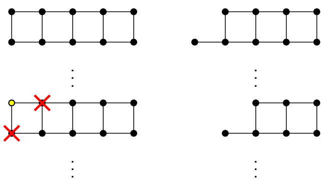
目標

グラフ G_n における独立集合の総数を計算する

例：$P_n \times P_2$ — 系統立ててやってみる

グラフ G_n を考えると，独立集合は次の 2 種類

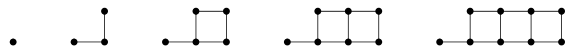
- (A) 左上端の頂点を要素として含まないもの
  = 左上端の頂点を除去してできるグラフの独立集合
- (B) 左上端の頂点を要素として含むもの
  = 左上の 3 頂点を除去してできるグラフの独立集合 ∪{ 左端の頂点 }



問題点：小さくなったグラフが G_k の形をしていない

例：$P_n \times P_2$ から得られたグラフ

次のグラフを考える (H_n と書くことにする)



H_1　H_2　H_3　H_4　H_5

目標

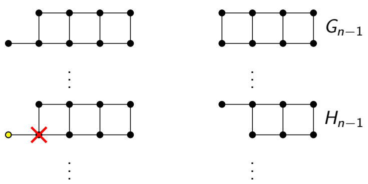
グラフ H_n における独立集合の総数を計算する

注：$n \geq 2$ のとき，
G_n の独立集合の総数 = H_n の独立集合の総数 +
H_{n-1} の独立集合の総数

例：$P_n \times P_2$ から得られたグラフ — 系統立てて考える

グラフ H_n を考えると，独立集合は次の 2 種類

- (A) 左端の頂点を要素として含まないもの
  = 左端の頂点を除去してできるグラフの独立集合
- (B) 左端の頂点を要素として含むもの
  = 左下の 2 頂点を除去してできるグラフの独立集合 ∪{ 左端の頂点 }



つまり，$n \geq 2$ のとき，
H_n の独立集合の総数 = G_{n-1} の独立集合の総数 +
H_{n-1} の独立集合の総数

例：$P_n \times P_2$ から得られたグラフ — まとめ

次のように定義

- b_n = グラフ G_n における独立集合の総数
- c_n = グラフ H_n における独立集合の総数

漸化式

$$b_n = \begin{cases} 3 & (n = 1 \text{ のとき}) \\ c_n + c_{n-1} & (n \geq 2 \text{ のとき}) \end{cases}$$

$$c_n = \begin{cases} 2 & (n = 1 \text{ のとき}) \\ b_{n-1} + c_{n-1} & (n \geq 2 \text{ のとき}) \end{cases}$$

これを解くのは次回

目次

単純な再帰アルゴリズム

アルゴリズム A

```
1: def fnct(n)
2:   print "a"
3:   if n > 2
4:     fnct(n-1)
5:     fnct(n-2)
6:   end
7: end
```

質問

fnct(n) を実行したとき,「a」は何個出力されるか？

単純な再帰アルゴリズム：例

n	a の数	n	a の数	n	a の数	n	a の数
1	1	11	177	21	21891	31	2692537
2	1	12	287	22	35421	32	4356617
3	3	13	465	23	57313	33	7049155
4	5	14	753	24	92735	34	11405773
5	9	15	1219	25	150049	35	18454929
6	15	16	1973	26	242785	36	29860703
7	25	17	3193	27	392835	37	48315633
8	41	18	5167	28	635621	38	78176337
9	67	19	8361	29	1028457	39	126491971
10	109	20	13529	30	1664079	40	204668309

単純な再帰アルゴリズム

アルゴリズム A

```
1: def fnct(n)
2:   print "a"
3:   if n > 2
4:     fnct(n-1)
5:     fnct(n-2)
6:   end
7: end
```

漸化式に向けて

f_n = fnct(n) を実行したときに出力される a の数

単純な再帰アルゴリズム

アルゴリズム A

```
1: def fnct(n)
2:   print "a"
3:   if n > 2
4:     fnct(n-1)
5:     fnct(n-2)
6:   end
7: end
```

漸化式に向けて

- 2 行目：n が何であろうと必ず 1 つは a が出力される
- 4 行目と 5 行目：再帰呼び出し

単純な再帰アルゴリズム

アルゴリズム A

```
1: def fnct(n)
2:   print "a"
3:   if n > 2
4:     fnct(n-1)
5:     fnct(n-2)
6:   end
7: end
```

漸化式

$$f_n = \begin{cases} 1 & (n \leq 2 \text{ のとき}) \\ 1 + f_{n-1} + f_{n-2} & (n \geq 3 \text{ のとき}) \end{cases}$$

ユークリッドのアルゴリズム — 最大公約数の計算

ユークリッドのアルゴリズム (正当性は演習問題)

```
1: def gcd(a, b) # precondition: a >= b
2:   print "G"
3:   if b == 0
4:     return a
5:   else
6:     gcd(b, a % b)
7:   end
8: end
```

a % b = a を b で割った余り (数学では $a \bmod b$ と書く)

質問

gcd(a, b) を実行したとき,「G」は何個出力されるか？

厳密に求めるのは難しいので，上界を求めたい

(最悪の場合における保証)

ユークリッドのアルゴリズム：ちょっと観察 (1)

a	b	G の数
14	11	5
143	11	2
1432	11	4
14325	11	5
143259	11	5
1432591	11	5
14325910	11	4
143259106	11	3
1432591067	11	5
14325910676	11	2
143259106765	11	4
1432591067659	11	5
14325910676592	11	5
143259106765923	11	4

ユークリッドのアルゴリズム：ちょっと観察 (2)

a	b	G の数
14	13	3
143	13	2
1432	13	4
14325	13	4
143259	13	4
1432591	13	4
14325910	13	3
143259106	13	4
1432591067	13	5
14325910676	13	4
143259106765	13	7
1432591067659	13	5
14325910676592	13	7
143259106765923	13	6

アルゴリズムの計算量

ユークリッドのアルゴリズム：解析に向けて

ユークリッドのアルゴリズム

```
def gcd(a, b) # precondition: a >= b
  print "G"
  if b == 0
    return a
  else
    gcd(b, a % b)
  end
end
```

考える量

$$g_n = \max_{a \geq 1, b \leq n} \{\mathtt{gcd}(a, b) \text{ の実行で出力される G の数}\}$$

直感：$g_n =$「$b \leq n$ に限った場合の最悪時計算量」

欲しいもの

g_n の上界

アルゴリズムの計算量

ユークリッドのアルゴリズム：計算量解析 — 補題

補題

自然数 $a, b \geq 1$ に対して，$a \geq b$ のとき，

$$a \bmod b \leq \left\lfloor \frac{a}{2} \right\rfloor$$

証明：$a = bq + r$ とする (ただし，$0 \leq r < b$)

- このとき，$a \bmod b = r$
- $a \geq b$ より，$q \geq 1$
- $b \leq \left\lfloor \frac{a}{2} \right\rfloor$ のとき，$r < b \leq \left\lfloor \frac{a}{2} \right\rfloor$
- $b > \left\lfloor \frac{a}{2} \right\rfloor$ のとき，$r = a - bq \leq a - b < a - \left\lfloor \frac{a}{2} \right\rfloor = \left\lceil \frac{a}{2} \right\rceil$
- したがって，このとき，$r \leq \left\lfloor \frac{a}{2} \right\rfloor$ □

注 (演習問題)：任意の自然数 n に対して，$n - \lfloor \frac{n}{2} \rfloor = \lceil \frac{n}{2} \rceil$

アルゴリズムの計算量

ユークリッドのアルゴリズム：計算量解析に向けて

ユークリッドのアルゴリズム

```
def gcd(a, b) # precondition: a >= b
  print "G"
  if b == 0
    return a
  else
    gcd(b, a % b)
  end
end
```

$$g_n = \mathtt{gcd}(a, b) \text{ の実行で出力される G の数}$$

となる a, b を考えると… (つまり，$b = n$)

アルゴリズムの計算量

ユークリッドのアルゴリズム：計算量解析 (1)

$$\begin{aligned} g_n &= \mathtt{gcd}(a, b) \text{ の実行で出力される G の数} \\ &= 1 + \mathtt{gcd}(b, a \bmod b) \text{ の実行で出力される G の数} \end{aligned}$$

ここで，場合分け

- $a \bmod b = 0$ のとき，$g_n = 2$
  ($\because$ gcd$(b, a \bmod b)$ はもう再帰呼び出しをしない)
- $a \bmod b \neq 0$ のとき，次のページ

アルゴリズムの計算量

ユークリッドのアルゴリズム：計算量解析 (2)

$$\begin{aligned} g_n &= \mathtt{gcd}(a, b) \text{ の実行で出力される G の数} \\ &= 1 + \mathtt{gcd}(b, a \bmod b) \text{ の実行で出力される G の数} \\ &= 2 + \mathtt{gcd}(a \bmod b, b \bmod (a \bmod b)) \text{ の実行で出力される G の数} \\ &\leq 2 + \max_{a' \geq 1, b' \leq \lfloor b/2 \rfloor} \{\mathtt{gcd}(a', b') \text{ の実行で出力される G の数}\} \\ &= 2 + g_{\lfloor b/2 \rfloor} = 2 + g_{\lfloor n/2 \rfloor} \end{aligned}$$

注意

先ほどの補題より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

つまり，$n \geq 1$ のとき，どちらの場合でも $g_n \leq 2 + g_{\lfloor n/2 \rfloor}$

アルゴリズムの計算量

ユークリッドのアルゴリズム：計算量解析 (結論)

得られた漸化式 (不等式であることに注意)

$$g_n \begin{cases} = 1 & n = 0 \text{ のとき} \\ \leq 2 + g_{\lfloor n/2 \rfloor} & n \geq 1 \text{ のとき} \end{cases}$$

ここからどう進めるかは次回

アルゴリズムの計算量

未解決問題：コラッツ予想

次のアルゴリズムを考える

```
def collatz(n)
  print n
  if n % 2 == 0
    collatz(n/2)
  else
    collatz(3*n+1)
  end
end
```

これは止まらないが…

コラッツ予想 (未解決)

任意の n に対して，collatz(n) は必ずいつか「1」を出力する

$n \leq 20 \times 2^{58}$ のときは正しいと分かっている (Oliveira e Silva '10)

目次

1. 組合せ構造の数え上げ
   グラフにおける独立集合の数え上げ
2. アルゴリズムの計算量
3. 今日のまとめ

この講義の概要

主題

次の 3 つを道具として
離散システム／アルゴリズムの設計と解析に関する方法論を学習する

- 数え上げ組合せ論
- 代数系
- 離散確率論

キャッチフレーズ：「離散数学を使う」

達成目標：以下の 3 項目をすべて達成することを目標とする

1. 数え上げ組合せ論，代数系，離散確率論における用語を正しく使うことができる
2. 数え上げ組合せ論，代数系，離散確率論における典型的な論法を用いて，証明を行うことができる
3. 数え上げ組合せ論，代数系，離散確率論を用いて，離散システム／アルゴリズムの設計と解析ができる

今日の目標

今日の目標

漸化式を立てられるようになる

- 組合せ構造の数え上げ
- アルゴリズムの計算量

格言

アルゴリズムの計算量解析の基礎は数え上げ

残った時間の使い方

- 演習問題をやる
  - 相談推奨 (ひとりでやらない)
- 質問をする
  - 教員と TA は巡回
- 退室時，小さな紙に感想など書いて提出する ← 重要
  - 内容は何でも OK
  - 匿名で OK