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

2024 年 10 月 29 日

最終更新：2024 年 10 月 20 日　12:49

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

無向グラフ

定義：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 は V の 要素数 2 の部分集合の集合

であるもののこと

例：

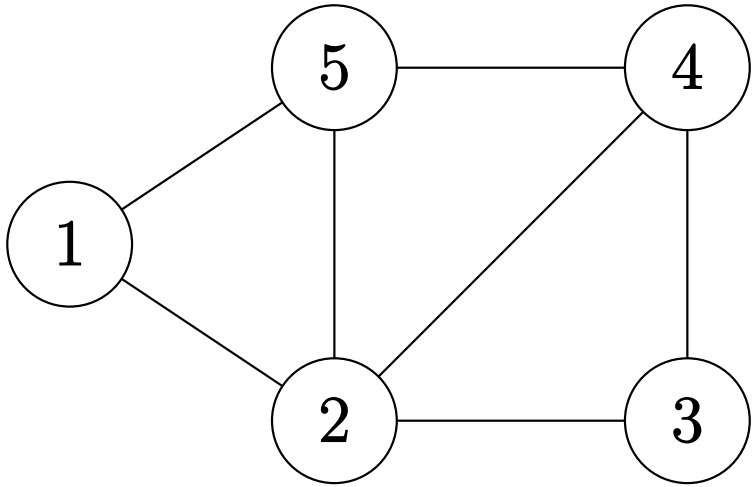
- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$

注意

$\{2, 5\} = \{5, 2\}$　(集合では順序を不問)

無向グラフの図示

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



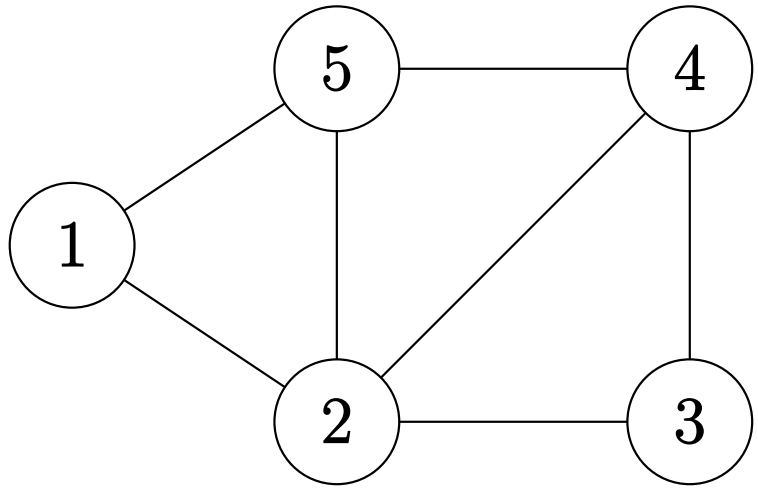
無向グラフの用語

無向グラフ $G = (V, E)$

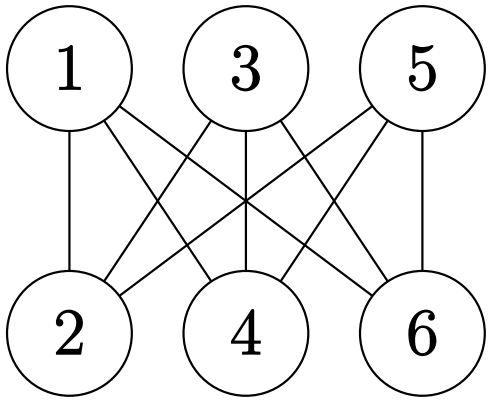
定義：無向グラフの用語

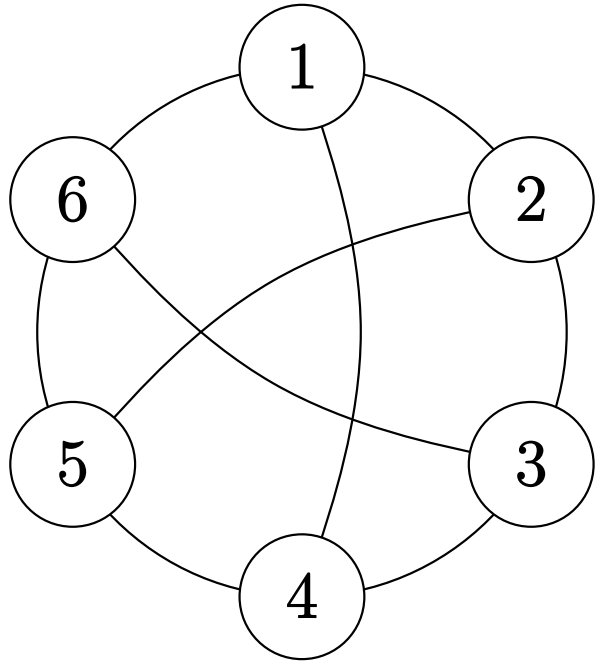
- V の要素を G の**頂点**と呼ぶ
- E の要素を G の**辺**と呼ぶ
- V を G の**頂点集合**と呼ぶ
- E を G の**辺集合**と呼ぶ
- 辺 $\{u, v\} \in E$ に対して，u, v をその**端点**と呼ぶ
- 頂点 v が辺 e の端点であるとき，v は e に**接続**するという
- 頂点 u と v が辺を成すとき，u と v は**隣接**するという

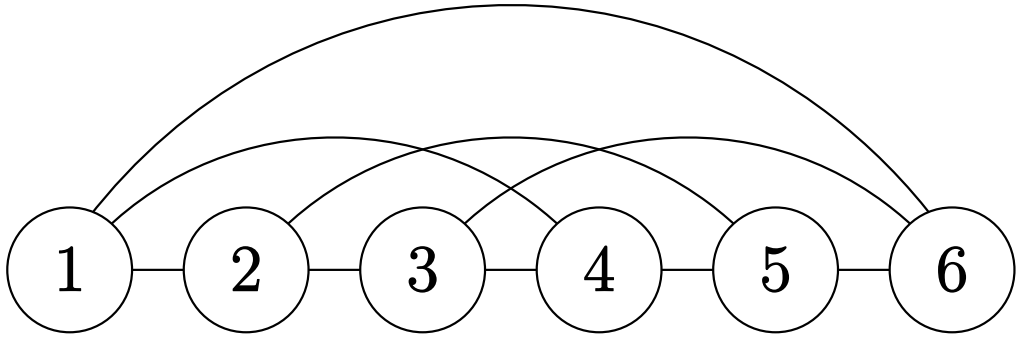
- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$
- 頂点 $2, 3$ は辺 $\{2, 3\}$ の端点
- 頂点 2 は辺 $\{2, 3\}$ に接続する
- 頂点 2 と頂点 3 は隣接する



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

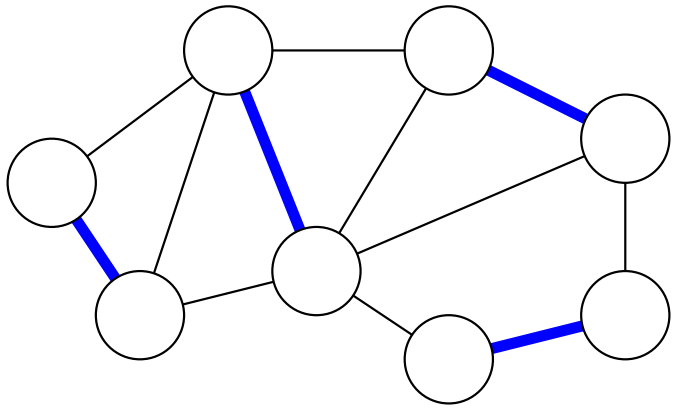
- 「頂点」の別名：「節点」，「ノード」，「点」
- 「辺」の別名：「枝」，「エッジ」

完全マッチング

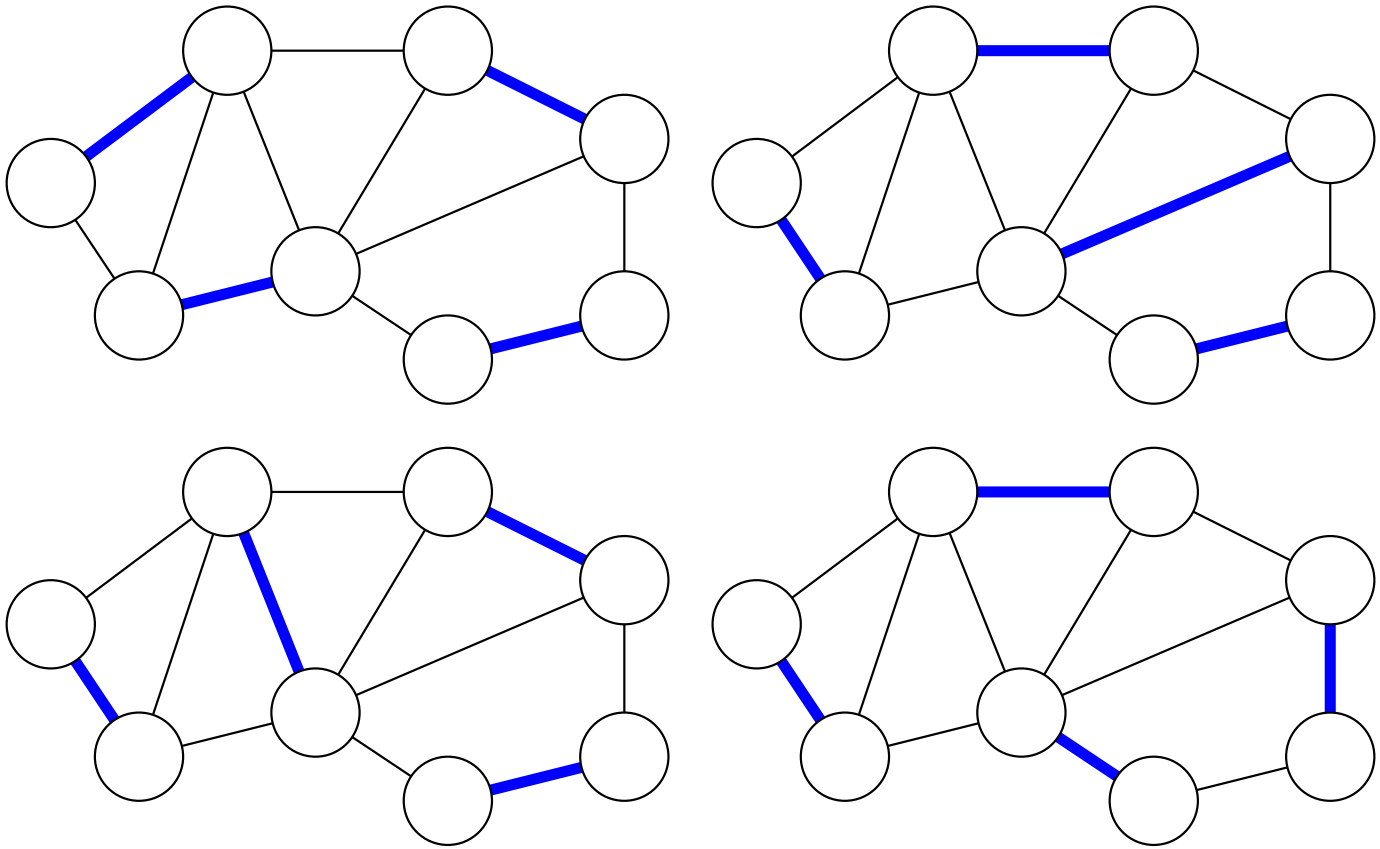
無向グラフ $G = (V, E)$

定義：完全マッチングとは？

G の **完全マッチング** とは，辺部分集合 $M \subseteq E$ で，
各頂点 $v \in V$ に対して，v に接続する M の辺がただ 1 つ存在するもの



すべての完全マッチング (完全マッチング全体)

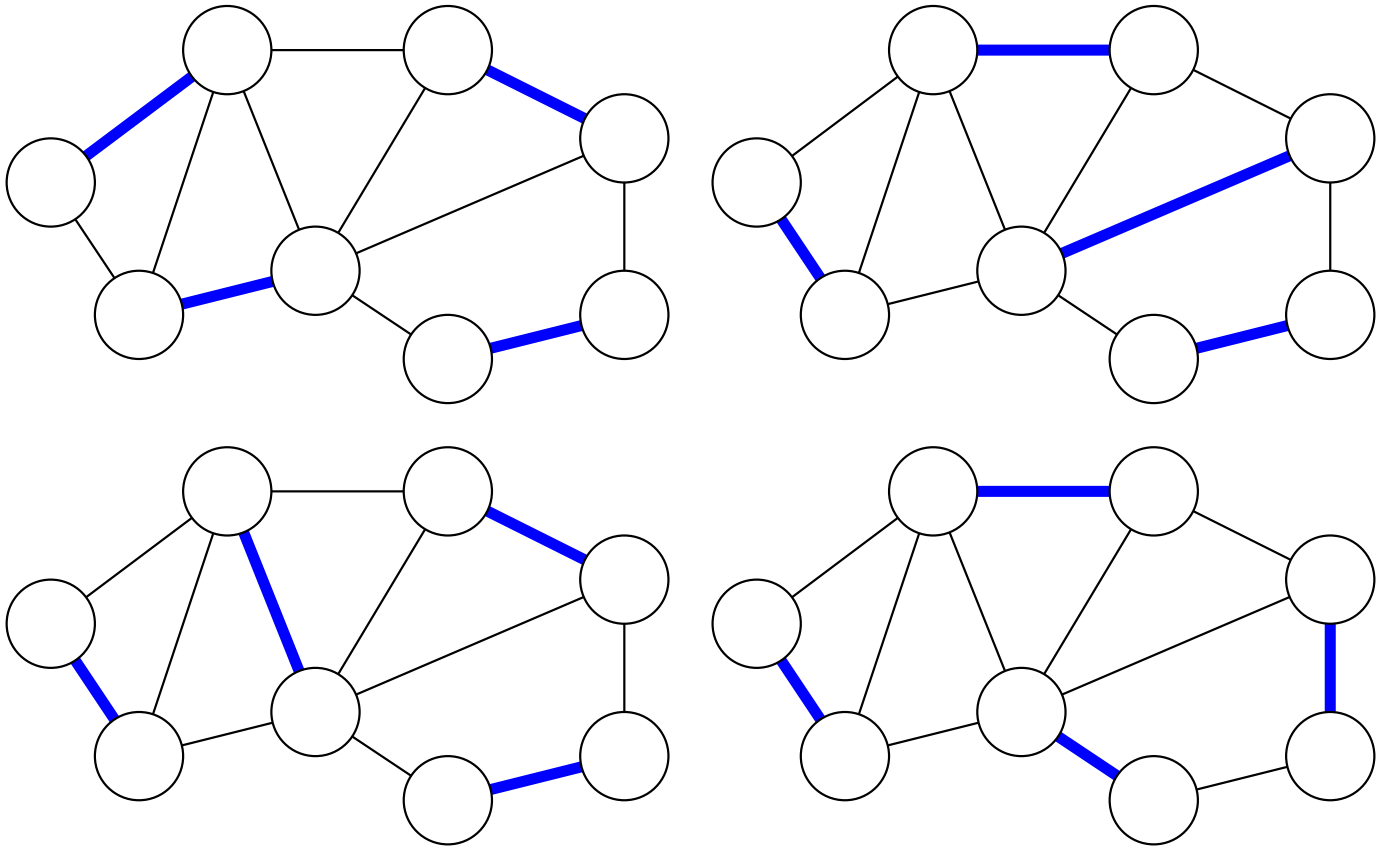


4 個

目標

やりたいこと

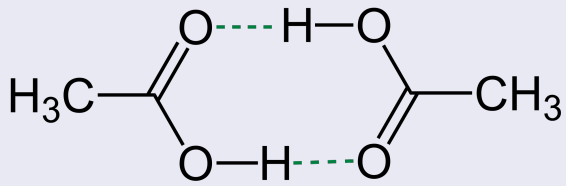
与えられた無向グラフにおける完全マッチングの総数を計算したい

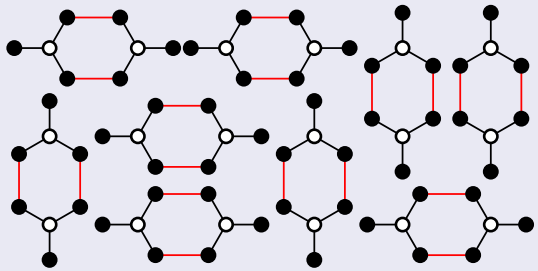


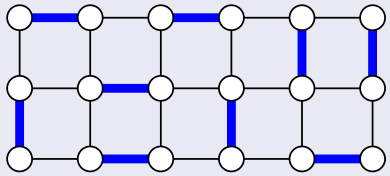
4 個

目標：なぜ計算したい？

物理学は，数え上げ組合せ論の重要な応用分野







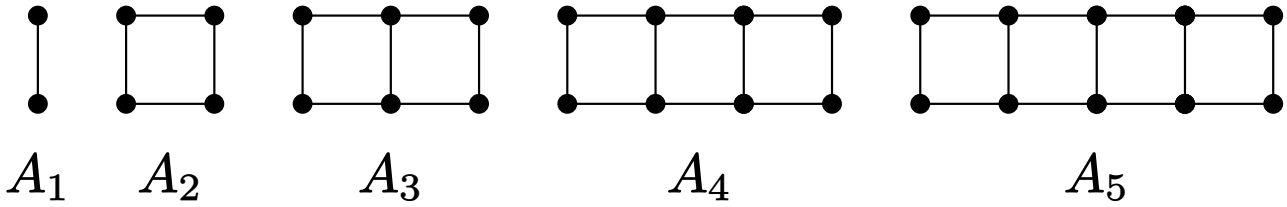
https://commons.wikimedia.org/wiki/File:Acetic_Acid_Hydrogenbridge_V.1.svg

(酢酸の二量体)

統計力学における「二量体モデル」

- 系を無向グラフ $G = (V, E)$ としてモデル化する
- 系において許される状態の総数 $=$ 完全マッチングの総数
- $\rightsquigarrow$ 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：2 段の格子

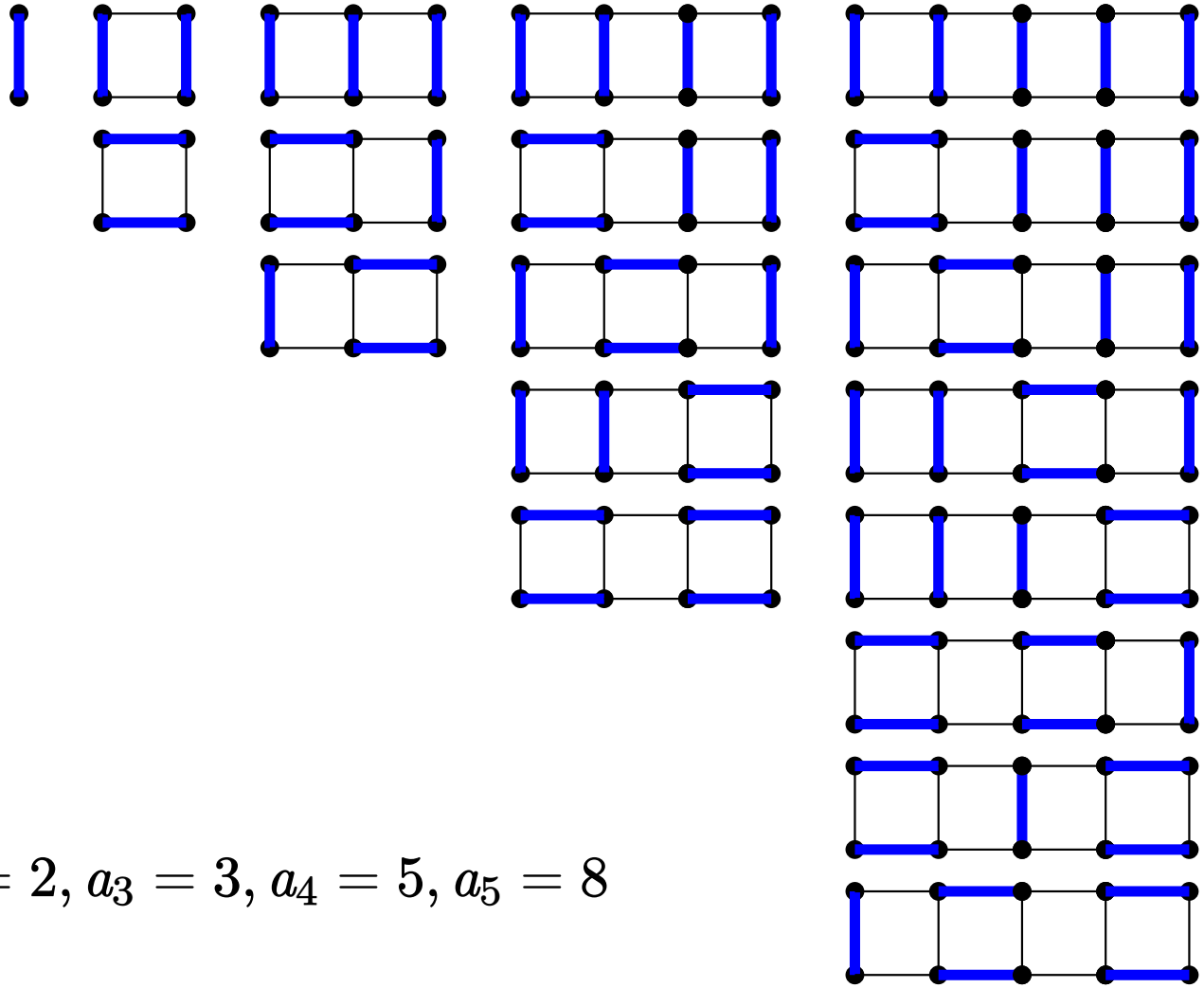


目標

グラフ A_n における完全マッチングの総数を計算する

例：2 段の格子 — 手でやってみる

$a_n = A_n$ の完全マッチングの総数

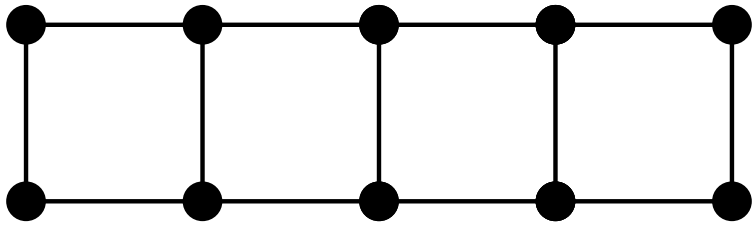


$a_1 = 1, a_2 = 2, a_3 = 3, a_4 = 5, a_5 = 8$

例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

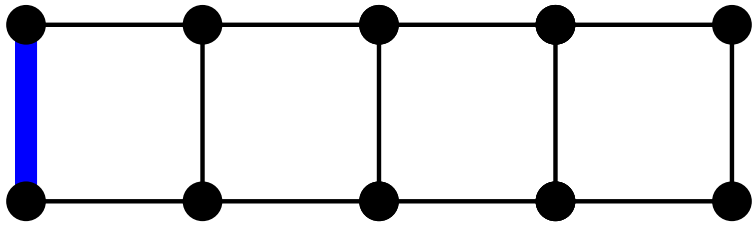
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

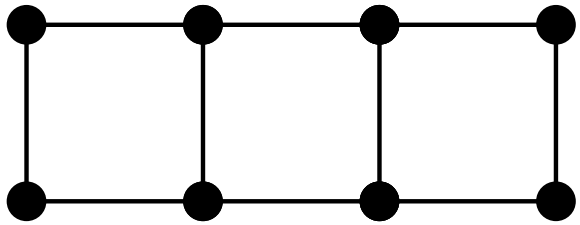
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

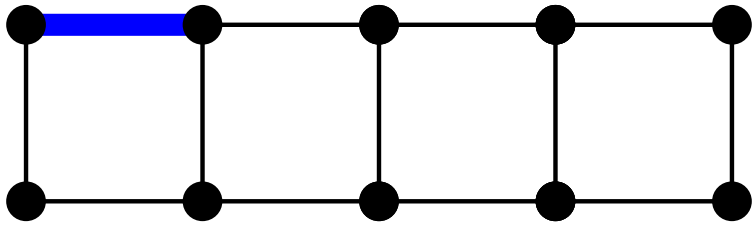
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_4 の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

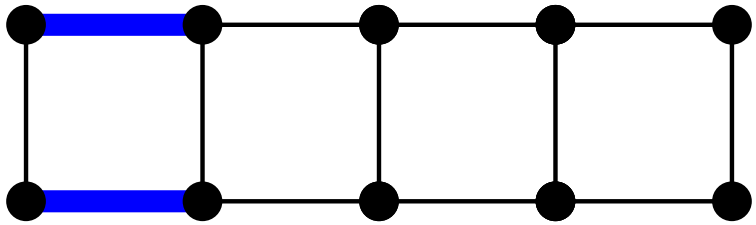
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_4 の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

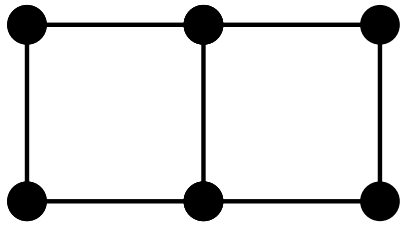
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_4 の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

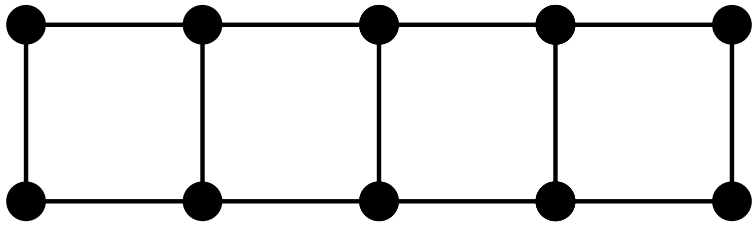
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_4 の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_3 の完全マッチング



例：2 段の格子 — 系統立ててやってみる

グラフ A_5 を考えると，完全マッチングは次の 2 種類

- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_4 の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_3 の完全マッチング

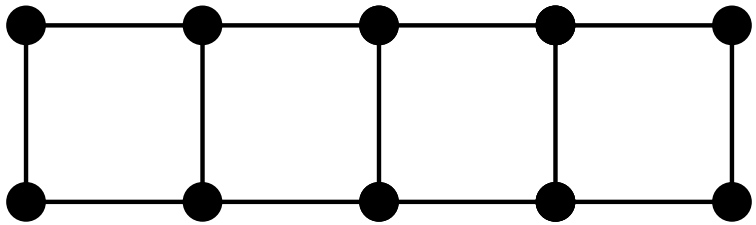


つまり，$a_5 = a_4 + a_3$

例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

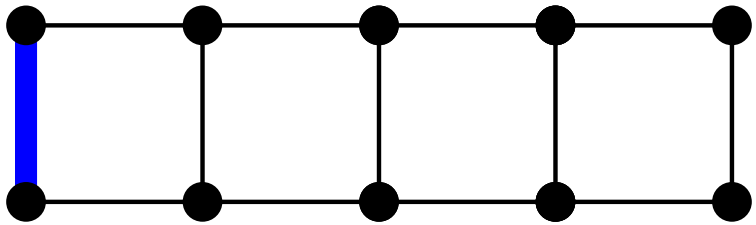
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

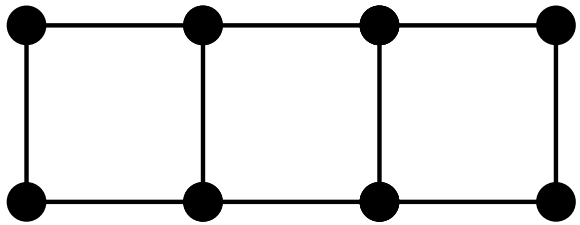
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

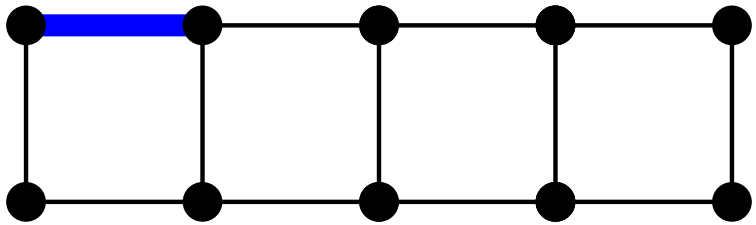
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

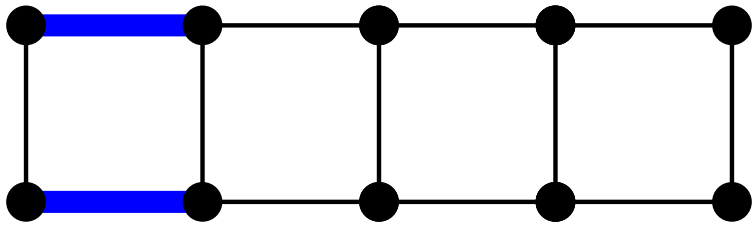
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

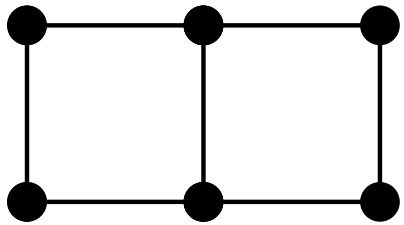
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

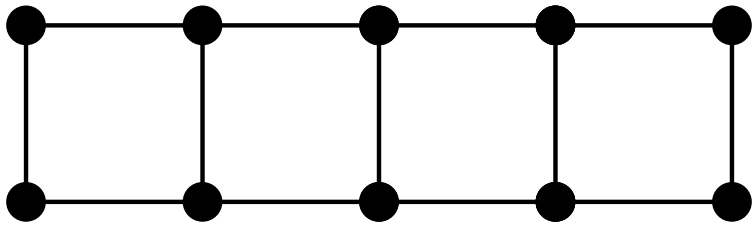
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-2} の完全マッチング



例：2 段の格子 — 系統立ててやってみる (一般化)

グラフ A_n を考えると，完全マッチングは次の 2 種類 $(n \geq 3)$

- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る A_{n-2} の完全マッチング



つまり，$a_n = a_{n-1} + a_{n-2}$ $(n \geq 3)$

例：2 段の格子 — まとめ

$a_n =$ グラフ A_n における完全マッチングの総数　とするとき

漸化式

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

これでプログラムは書ける (解くのは次回)

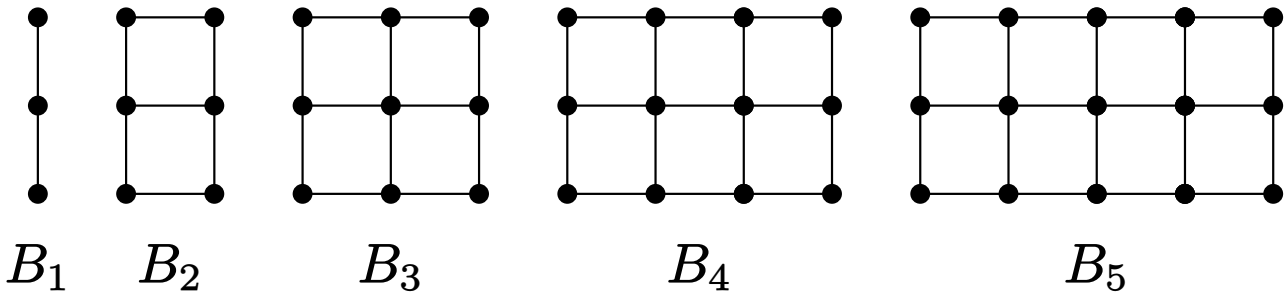
例：2 段の格子 — プログラム実行結果

n	a_n
1	1
2	2
3	3
4	5
5	8
6	13
7	21
8	34
9	55
10	89
11	144
12	233
13	377
14	610
15	987

n	a_n
16	1597
17	2584
18	4181
19	6765
20	10946
21	17711
22	28657
23	46368
24	75025
25	121393
26	196418
27	317811
28	514229
29	832040
30	1346269

例：3 段の格子

次のグラフを考える (B_n と書くことにする)

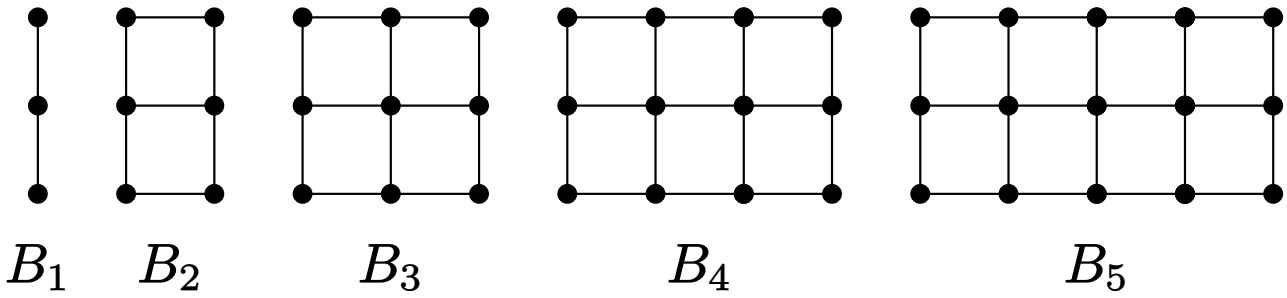


目標

グラフ B_n における完全マッチングの総数 b_n を計算する

例：3 段の格子

次のグラフを考える (B_n と書くことにする)



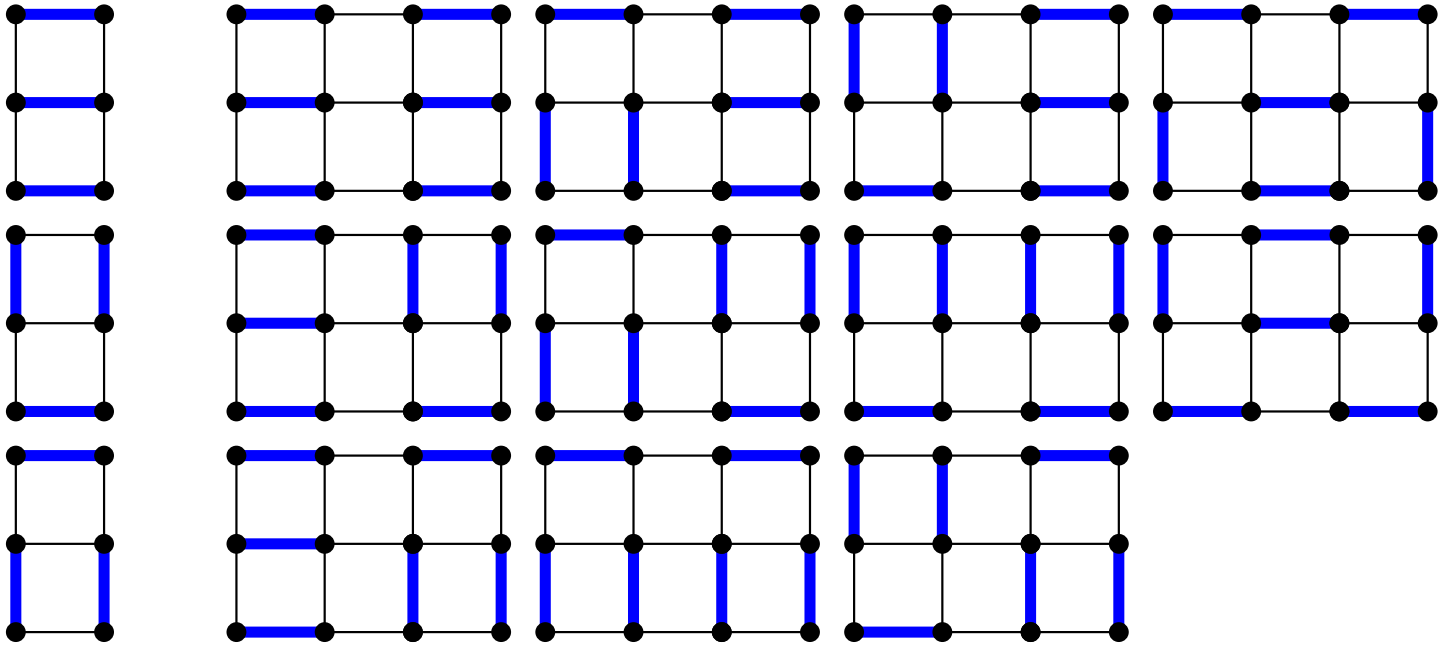
目標

グラフ B_n における完全マッチングの総数 b_n を計算する

注：n が奇数 $\Rightarrow$ $b_n = 0$

例：3 段の格子 — 手でやってみる

$b_n = B_n$ の完全マッチングの総数

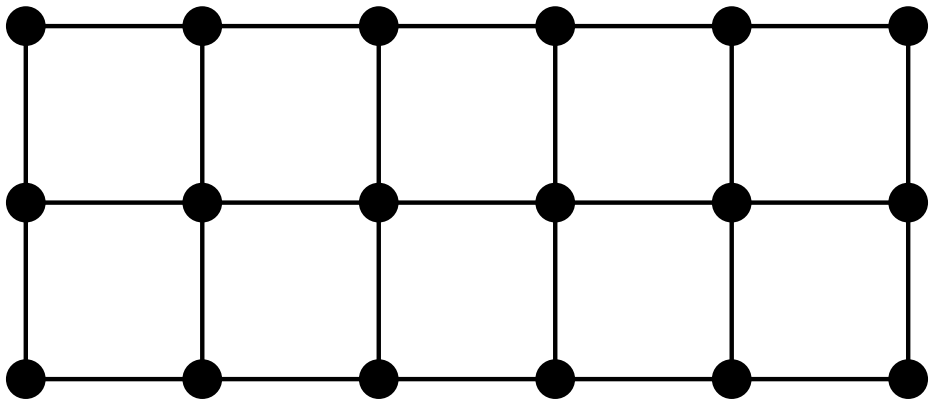


$b_1 = 0, b_2 = 3, b_3 = 0, b_4 = 11, b_5 = 0$

例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

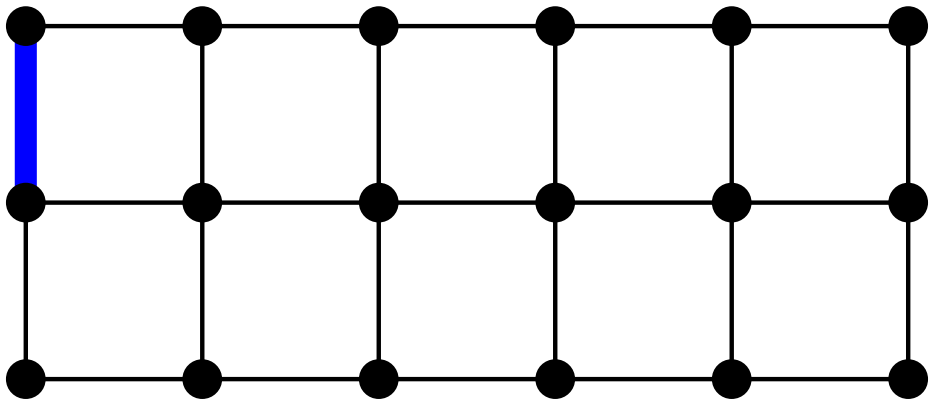
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

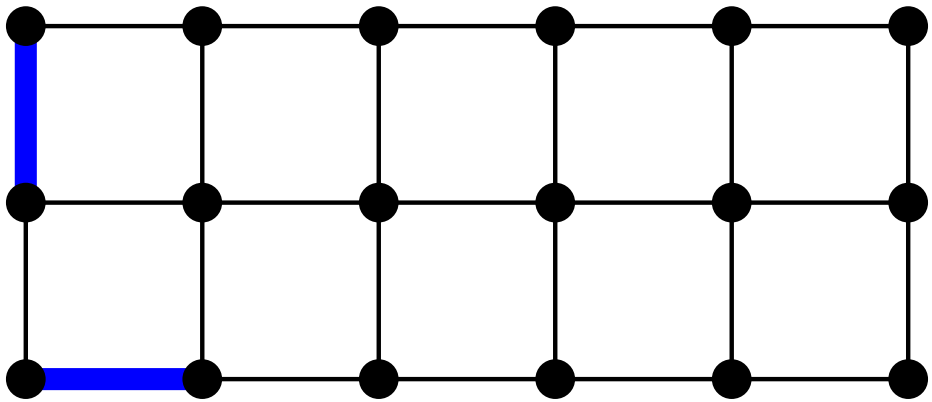
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

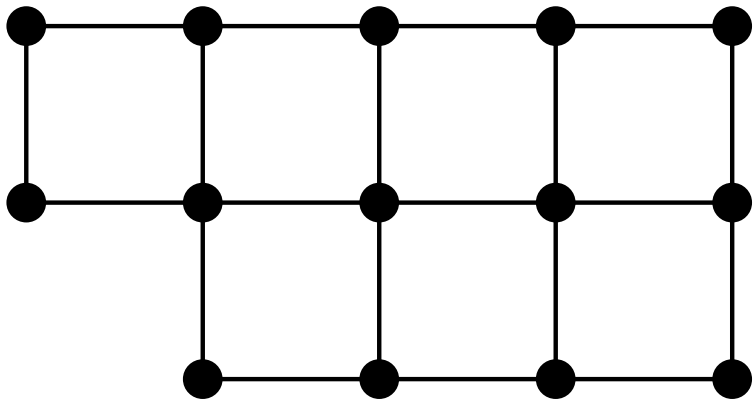
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

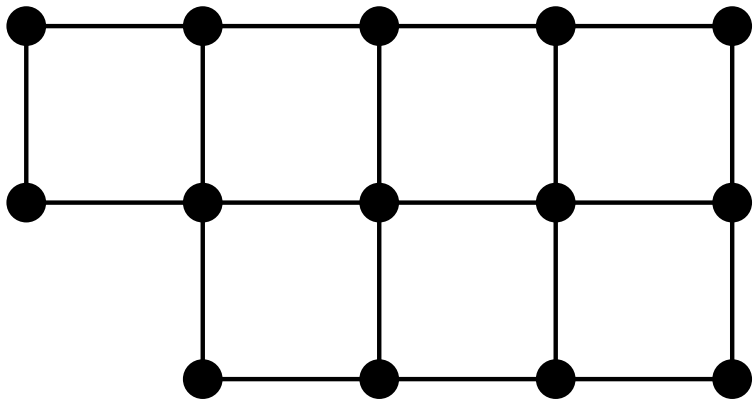
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

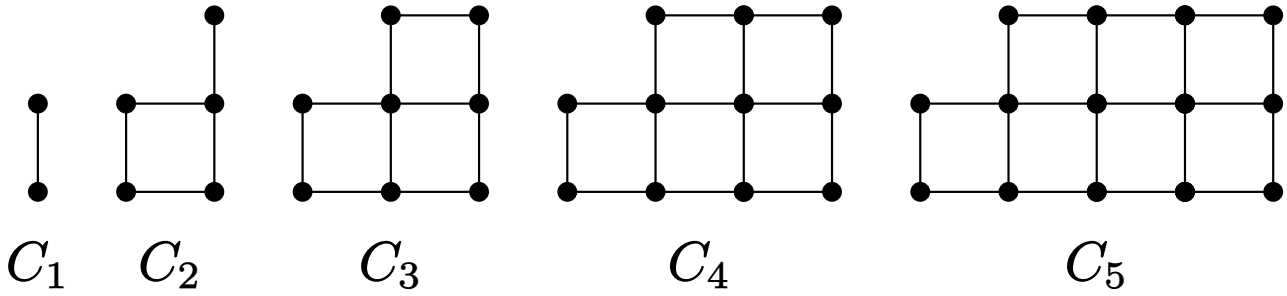
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



問題点：小さくなったグラフが 3 段の格子ではない

例：欠けた 3 段の格子

次のグラフを考える (C_n と書くことにする)



目標

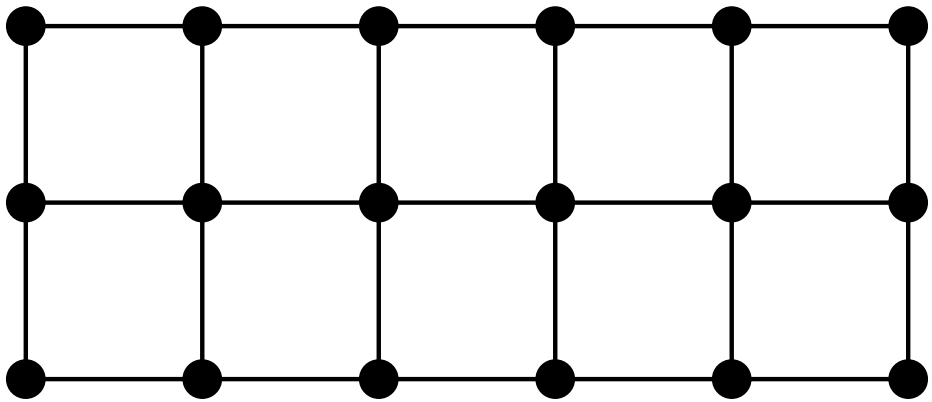
グラフ C_n における完全マッチングの総数 c_n を計算する

注：n が偶数 $\Rightarrow$ $c_n = 0$

例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

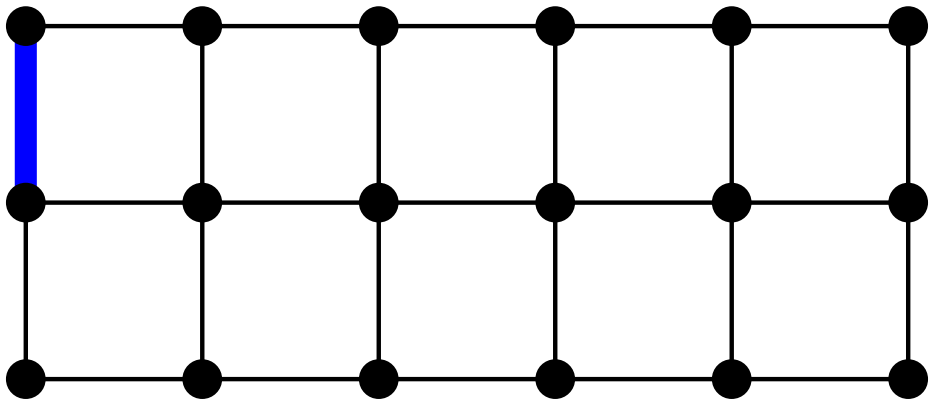
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

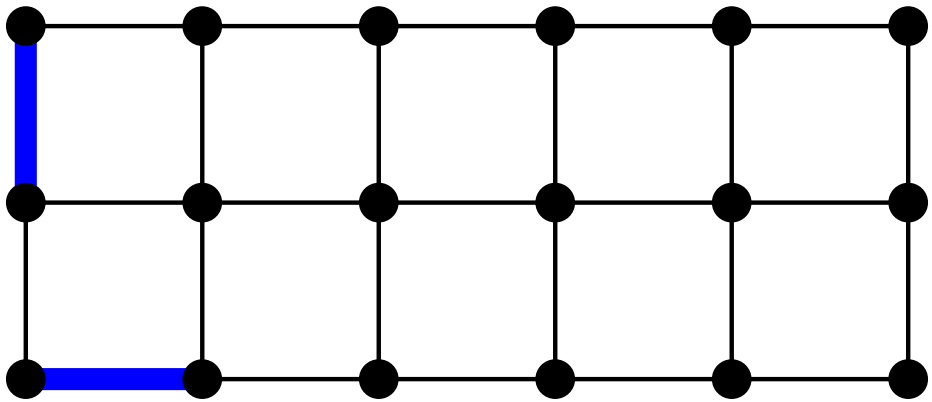
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

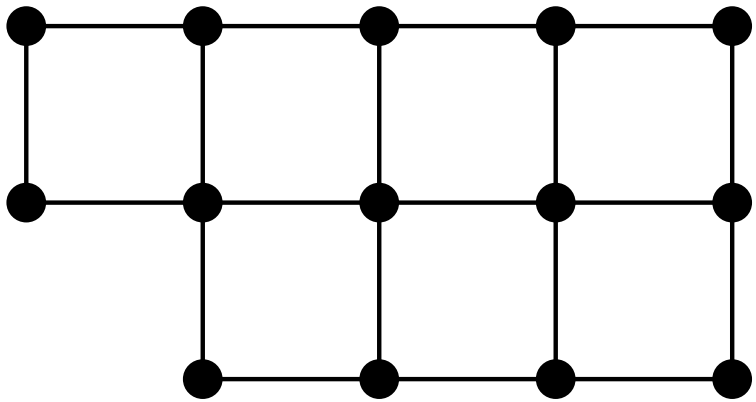
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

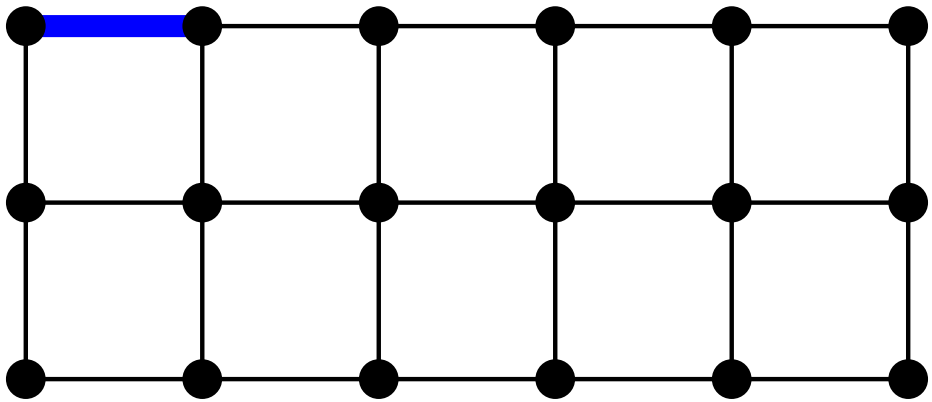
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

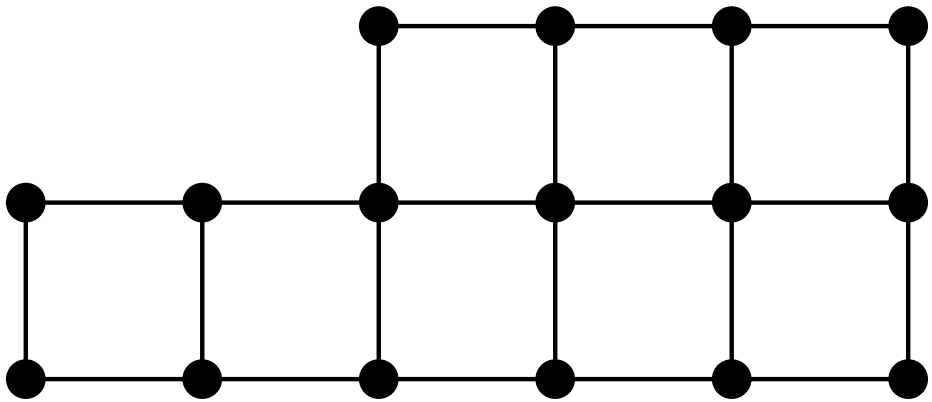
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

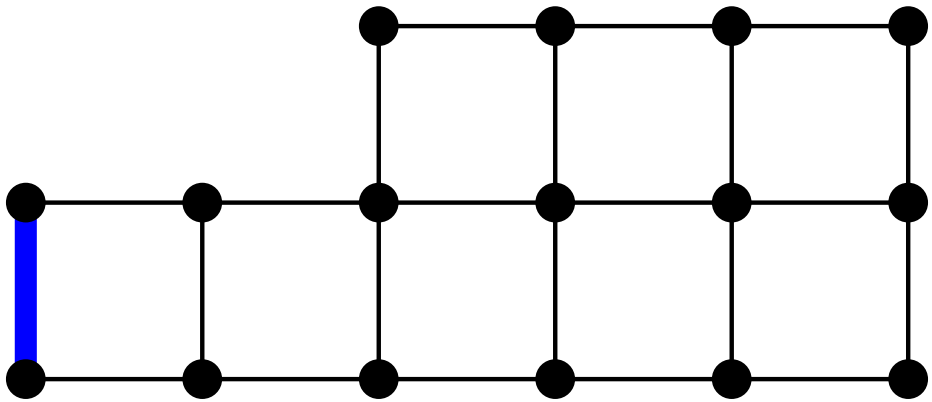
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

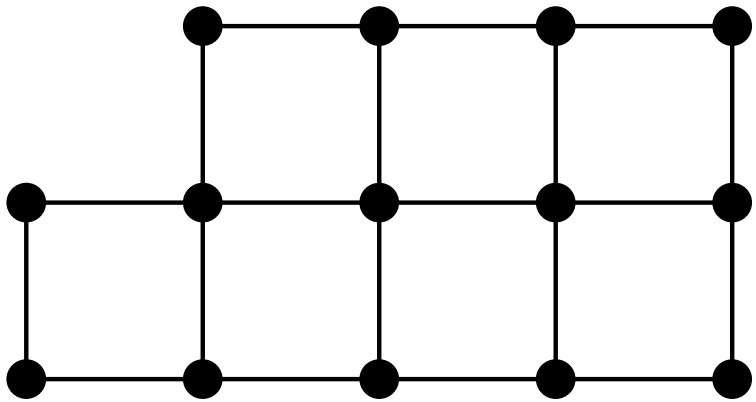
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

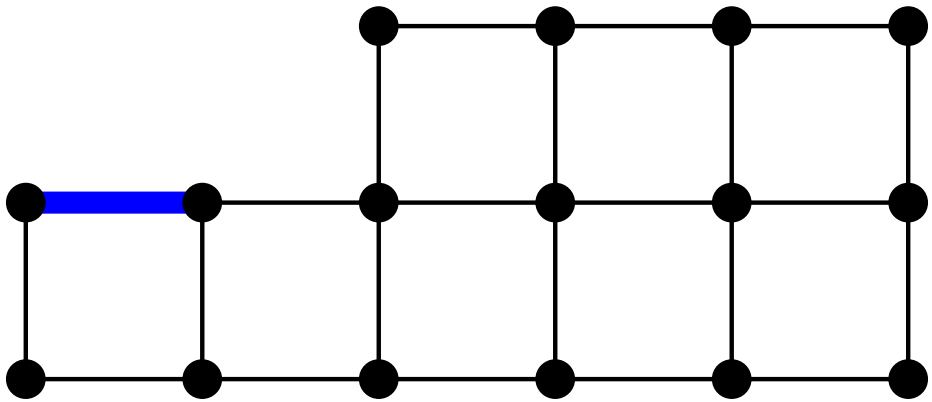
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング か



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

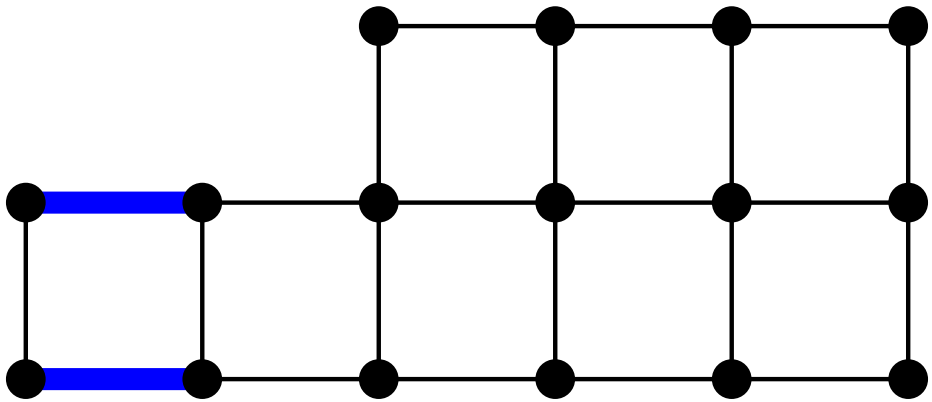
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング か



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

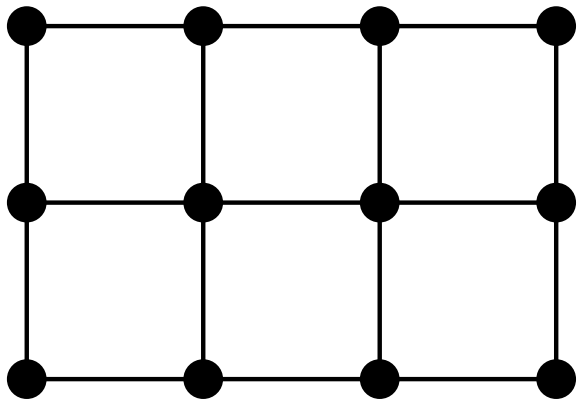
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング か



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

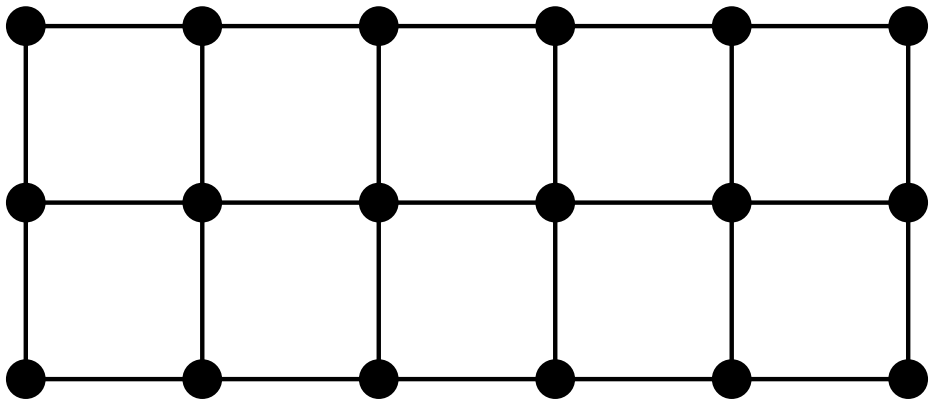
- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング か
  右側に残る B_{n-2} の完全マッチング



例：3 段の格子 — 系統立ててやってみる

グラフ B_n を考えると，完全マッチングは次の 2 種類 ($n \geq 4$，偶数)

- (A) 左上の頂点に接続する辺 が 縦の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング
- (B) 左上の頂点に接続する辺 が 横の辺 のもの
  $\rightsquigarrow$ 他の辺は 右側に残る C_{n-1} の完全マッチング か
  右側に残る B_{n-2} の完全マッチング



つまり，$b_n = b_{n-2} + 2c_{n-1}$ ($n \geq 4$，偶数)

例：3 段の格子 — まとめ

- $b_n =$ グラフ B_n における完全マッチングの総数
- $c_n =$ グラフ C_n における完全マッチングの総数

とするとき

漸化式

$$b_n = \begin{cases} 0 & (n \text{ が奇数のとき}) \\ 3 & (n = 2 \text{ のとき}) \\ b_{n-2} + 2c_{n-1} & (n \geq 4,\ \text{偶数のとき}) \end{cases}$$

$$c_n = \begin{cases} 0 & (n \text{ が偶数のとき}) \\ 1 & (n = 1 \text{ のとき}) \\ b_{n-1} + c_{n-2} & (n \geq 3,\ \text{奇数のとき}) \end{cases} \quad \text{(演習問題)}$$

これでプログラムは書ける (解くのは次回)

例：3 段の格子 — プログラム実行結果

n	$b(n)$	$c(n)$
1	0	1
2	3	0
3	0	4
4	11	0
5	0	15
6	41	0
7	0	56
8	153	0
9	0	209
10	571	0
11	0	780
12	2131	0
13	0	2911
14	7953	0
15	0	10864

n	$b(n)$	$c(n)$
16	29681	0
17	0	40545
18	110771	0
19	0	151316
20	413403	0
21	0	564719
22	1542841	0
23	0	2107560
24	5757961	0
25	0	7865521
26	21489003	0
27	0	29354524
28	80198051	0
29	0	109552575
30	299303201	0

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

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

目次

最大公約数の計算

問題：最大公約数の計算

- 入力：非負整数 a, b (ただし，$a \geq b$)
- 出力：a と b の最大公約数

最大公約数の計算

問題：最大公約数の計算

- 入力：非負整数 a, b (ただし，$a \geq b$)
- 出力：a と b の最大公約数

定義：約数

非負整数 $a \geq 0$ の **約数** とは，次を満たす非負整数 $d \geq 0$

ある非負整数 $a' \geq 0$ が存在して，　$a = a'd$

定義：最大公約数

非負整数 $a, b \geq 0$ の **最大公約数** とは次を満たす非負整数 $d^* \geq 0$

- d^* は a, b の共通の約数 (公約数) である
- a, b の任意の公約数 d に対して，d は d^* の約数である

最大公約数：例

例

$$
\begin{aligned}
12 \text{ と } 18 \text{ の最大公約数} &= 6 \\
15 \text{ と } 1 \text{ の最大公約数} &= 1 \\
4 \text{ と } 0 \text{ の最大公約数} &= 4 \\
0 \text{ と } 0 \text{ の最大公約数} &= 0
\end{aligned}
$$

定義：約数

非負整数 $a \geq 0$ の **約数** とは，次を満たす非負整数 $d \geq 0$

ある非負整数 $a' \geq 0$ が存在して，　$a = a'd$

定義：最大公約数

非負整数 $a, b \geq 0$ の **最大公約数** とは次を満たす非負整数 $d^* \geq 0$

- d^* は a, b の共通の約数 (公約数) である
- a, b の任意の公約数 d に対して，d は d^* の約数である

ユークリッドのアルゴリズム：最大公約数を計算するアルゴリズム

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 >= 0
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
19	11	6
919	11	5
6919	11	2
46919	11	5
546919	11	4
8546919	11	6
28546919	11	4
728546919	11	3
8728546919	11	5
38728546919	11	6
538728546919	11	4
1538728546919	11	5
81538728546919	11	5

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
41	20	3
441	20	3
7441	20	3
57441	20	3
457441	20	3
1457441	20	3
11457441	20	3
511457441	20	3
3511457441	20	3
53511457441	20	3
453511457441	20	3
2453511457441	20	3
22453511457441	20	3

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

```
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
```

考える量

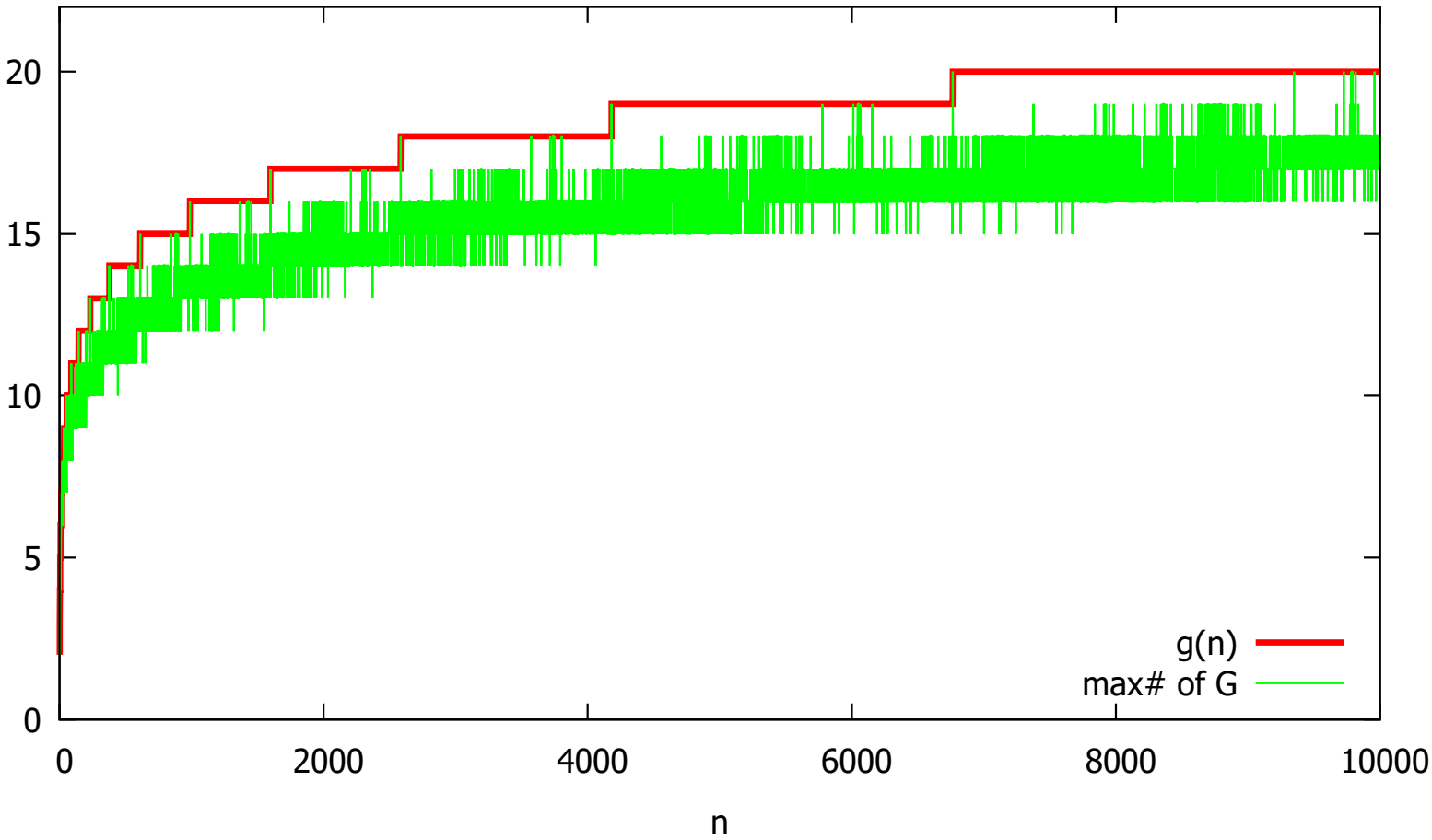
$$g_n = \max_{a \geq 1, b \leq n} \{\texttt{gcd}(a,\ b) \text{ の実行で出力される } \texttt{G} \text{ の数 }\}$$

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

ユークリッドのアルゴリズム：解析に向けて (図)



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

細い線 = $\max_{a \geq 1}$ {gcd(a, n) の実行で出力される G の数 }

ユークリッドのアルゴリズム：計算量解析 — 補題 A

考える量

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

補題 A

任意の正整数 n に対して，$g_n \leq g_{n+1}$

証明：「$g_n = \texttt{gcd}(a, b)$ の実行で出力される $\texttt{G}$ の数」となる
$a \geq 1$ と $b \leq n$ を考えると，

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

したがって，$g_n \leq g_{n+1}$ □

ユークリッドのアルゴリズム：計算量解析 — 補題 B

補題 B

正整数 $a, b \geq 1$ 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，$a \bmod b = r$

ユークリッドのアルゴリズム：計算量解析 — 補題 B

補題 B

正整数 $a, b \geq 1$ 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，$q \geq 1$

ユークリッドのアルゴリズム：計算量解析 — 補題 B

補題 B

正整数 $a, b \geq 1$ 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，$r < b \leq \left\lfloor \frac{a}{2} \right\rfloor$

ユークリッドのアルゴリズム：計算量解析 — 補題 B

補題 B

正整数 $a, b \geq 1$ 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，$r = a - bq \leq a - b < a - \left\lfloor \frac{a}{2} \right\rfloor = \left\lceil \frac{a}{2} \right\rceil$

注 (演習問題)：任意の非負整数 n に対して，$n - \left\lfloor \frac{n}{2} \right\rfloor = \left\lceil \frac{n}{2} \right\rceil$

ユークリッドのアルゴリズム：計算量解析 — 補題 B

補題 B

正整数 $a, b \geq 1$ 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)：任意の非負整数 n に対して，$n - \left\lfloor \frac{n}{2} \right\rfloor = \left\lceil \frac{n}{2} \right\rceil$

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

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

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

となる a, b を考えると...

ユークリッドのアルゴリズム：計算量解析 (1)

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

ユークリッドのアルゴリズム：計算量解析 (1)

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

ユークリッドのアルゴリズム：計算量解析 (1)

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

ここで，場合分け

ユークリッドのアルゴリズム：計算量解析 (1)

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

ここで，場合分け

- $a \bmod b = 0$ のとき，$g_n = 2$

ユークリッドのアルゴリズム：計算量解析 (1)

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

ここで，場合分け

- $a \bmod b = 0$ のとき，$g_n = 2$
  ($\because$ $\texttt{gcd}(b, a \bmod b)$ はもう再帰呼び出しをしない)

ユークリッドのアルゴリズム：計算量解析 (1)

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

ここで，場合分け

- $a \bmod b = 0$ のとき，$g_n = 2$
  ($\because$ $\texttt{gcd}(b, a \bmod b)$ 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

ユークリッドのアルゴリズム：計算量解析 (2)

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

ユークリッドのアルゴリズム：計算量解析 (2)

$$
\begin{aligned}
g_n &= \texttt{gcd}(a, b) \text{ の実行で出力される } \texttt{G} \text{ の数} \\
&= 1 + \texttt{gcd}(b, a \bmod b) \text{ の実行で出力される } \texttt{G} \text{ の数} \\
&= 2 + \texttt{gcd}(a \bmod b, b \bmod (a \bmod b)) \text{ の実行で出力される } \texttt{G} \text{ の数}
\end{aligned}
$$

ユークリッドのアルゴリズム：計算量解析 (2)

$$
\begin{aligned}
g_n &= \texttt{gcd}(a, b) \text{ の実行で出力される } \texttt{G} \text{ の数} \\
&= 1 + \texttt{gcd}(b, a \bmod b) \text{ の実行で出力される } \texttt{G} \text{ の数} \\
&= 2 + \texttt{gcd}(a \bmod b, b \bmod (a \bmod b)) \text{ の実行で出力される } \texttt{G} \text{ の数}
\end{aligned}
$$

注意

補題 B より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

ユークリッドのアルゴリズム：計算量解析 (2)

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

注意

補題 B より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

ユークリッドのアルゴリズム：計算量解析 (2)

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

注意

補題 B より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

ユークリッドのアルゴリズム：計算量解析 (2)

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

注意

補題 B より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

ユークリッドのアルゴリズム：計算量解析 (2)

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

注意

補題 B より，$b \bmod (a \bmod b) \leq \left\lfloor \frac{b}{2} \right\rfloor$

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

ユークリッドのアルゴリズム：計算量解析 (2)

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

注意

補題 B 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 $g_n \leq 2 + g_{\lfloor n/2 \rfloor}$

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

未解決問題：コラッツ予想

次のアルゴリズムを考える

```
1: def collatz(n)
2:   print n
3:   if n % 2 == 0
4:     collatz(n/2)
5:   else
6:     collatz(3*n+1)
7:   end
8: end
```

これは止まらないが…

コラッツ予想 (未解決)

任意の正整数 n に対して，`collatz(`n`)` は必ずいつか「1」を出力する

$n \leq 2^{70}$ のときは正しいと報告されている (Barina '23)

http://www.ericr.nl/wondrous/

目次

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