# 第1章

# 解答例

練習問題の解答例です.

**練習 2.1.1 の解答例**　省略

**練習 2.1.2 の解答例**　省略

**練習 2.2.1 の解答例**　省略

**練習 2.3.1 の解答例**　省略

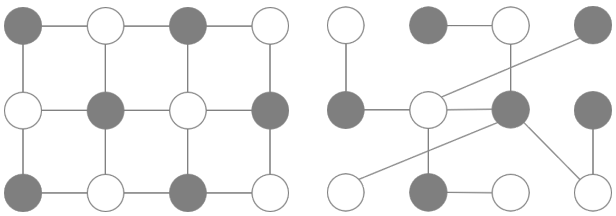
**練習 2.3.2 の解答例**　完成 Excel ファイル [ch2-2.xlsx] のシート [rw2] 参照

**練習 3.1.1 の解答例**　省略

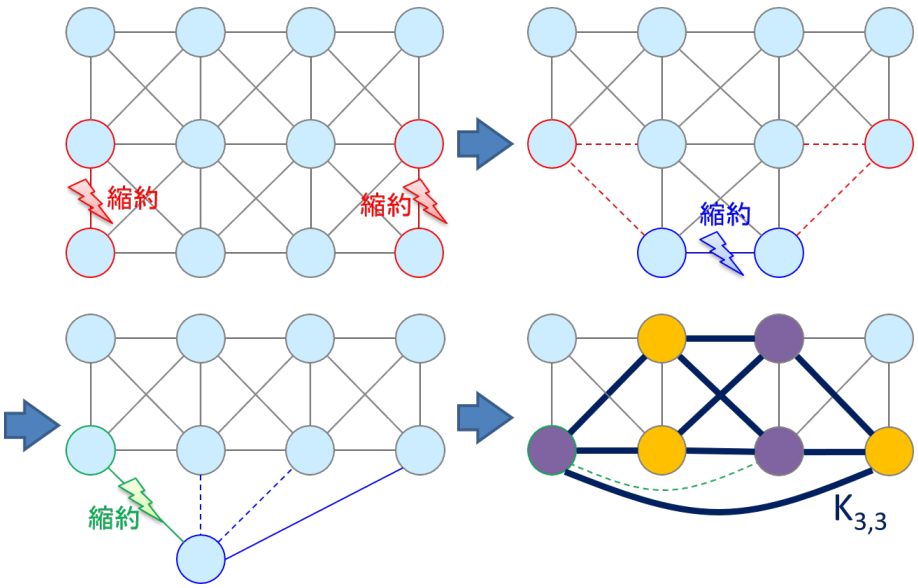
**練習 3.1.2 の解答例**　3 つのグラフ（左/中/右）の判定（○/×）は下表の通りです.

| | 木 | 平面 | 完全 | 二部 |
|---|---|---|---|---|
| 左のグラフ | × | × | × | × |
| 中央のグラフ | × | ○ | × | ○ |
| 右のグラフ | ○ | ○ | × | ○ |

中央と右のグラフが二部グラフであることは，ぱっと見分かりづらいと思いますが，下図の様に $V_1$（黒点）と $V_2$（白点）に分ければ定義を満たします．また，奇数長の閉路がないことからも，二部グラフであることが結論できます.



左のグラフの平面性判定は少々やっかいで，Kuratowski の定理 (1930)「グラフが平面に描画できるための必要十分条件は，$K_5, K_{3,3}$ のどちらも位相的マイナーとしてもたないこと」を使う必要があります．点を順に上手く**縮約 (shrink)** していくと，$K_{3,3}$ を含む事が分かり，平面描画出来ないことが判明します．詳細は，参考文献を参照してください.



**ch3.2 例題の解答例**　オイラー閉路とハミルトン閉路の例.

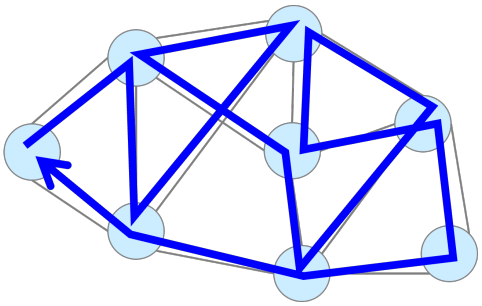


図 1.1　オイラー閉路の例

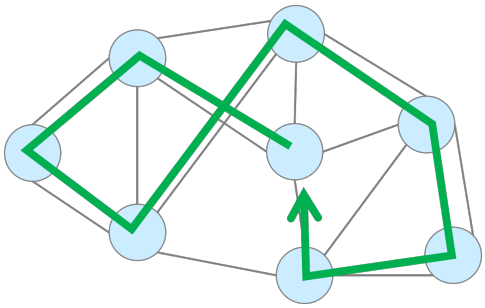


図 1.2　ハミルトン閉路の例

**練習 3.4.1 の解答例**

- $M_1 = \{\{A,Q\},\{B,S\},\{C,P\},\{C,R\},\{D,U\},\{E,U\},\{F,T\}\}$ はマッチングではありません．端点を共有している枝がある（「$\{C,P\}$ と $\{C,R\}$ が端点 $C$ を共有している」，「$\{D,U\}$ と $\{E,U\}$ が端点 $U$ を共有している」）からです．
- $M_2 = \{\{A,P\},\{B,T\},\{C,S\},\{D,Q\},\{E,U\}\}$ はマッチングです．マッチされていない点 $F,R$ があるので，完全マッチングではありません．

**練習 3.4.2 の解答例**　不安定です．ブロッキングペアの例 $\{D,Q\}$.

**練習 4.1.1 の解答例**　省略

**練習 4.1.2 の解答例**　完成 Excel ファイル [ch4-1.xlsx] のシート [cramer_chi2test] 参照

**練習 4.1.3 の解答例**　例として表 1.1〜1.3 を見てみましょう.

表 1.1　性別と嗜好 1

| | 紅茶 | 緑茶 | 珈琲 | 計 |
|---|---|---|---|---|
| 男 | 0 | 3 | 9 | 12 |
| 女 | 6 | 0 | 0 | 6 |
| 計 | 6 | 3 | 9 | 18 |

表 1.2　性別と嗜好 2

| | 紅茶 | 緑茶 | 珈琲 | 計 |
|---|---|---|---|---|
| 男 | 3 | 1 | 8 | 12 |
| 女 | 3 | 2 | 1 | 6 |
| 計 | 6 | 3 | 9 | 18 |

表 1.3　性別と嗜好 3

| | 紅茶 | 緑茶 | 珈琲 | 計 |
|---|---|---|---|---|
| 男 | 4 | 2 | 6 | 12 |
| 女 | 2 | 1 | 3 | 6 |
| 計 | 6 | 3 | 9 | 18 |

3 表とも周辺度数と総度数は共通です．表 1.3 が 3 表共通の理論度数となります．3 表とも，$n = 18$，$m = \min\{2-1, 3-1\} = 1$ です．ピアソンの $\chi^2$ 統計量は，それぞれ

$$\chi^2 = \frac{(0-4)^2}{4} + \frac{(3-2)^2}{2} + \frac{(9-6)^2}{6} + \frac{(6-2)^2}{2} + \frac{(0-1)^2}{1} + \frac{(0-3)^2}{3} = 18$$

$$\chi^2 = \frac{(3-4)^2}{4} + \frac{(1-2)^2}{2} + \frac{(8-6)^2}{6} + \frac{(3-2)^2}{2} + \frac{(2-1)^2}{1} + \frac{(1-3)^2}{3} = 17/4$$

$$\chi^2 = \frac{(4-4)^2}{4} + \frac{(2-2)^2}{2} + \frac{(6-6)^2}{6} + \frac{(2-2)^2}{2} + \frac{(1-1)^2}{1} + \frac{(3-3)^2}{3} = 0$$

で，クラメルの連関係数はそれぞれ，

$$V = \sqrt{\frac{18}{18 \cdot 1}} = 1$$

$$V = \sqrt{\frac{17/4}{18 \cdot 1}} = 0.49$$

$$V = \sqrt{\frac{0}{18 \cdot 1}} = 0$$

です．以上からわかる通り，表 1.1 のように項目ごとに各変量のみの値となれば，連関係数は 1，即ち，完全相関になり（女性は必ず紅茶を飲み，男性は必ず緑茶か珈琲を飲む，逆に，紅茶を飲んでいれば必ず女性で，緑茶か珈琲なら必ず男性），表 1.3 のように理論度数に一致すれば，連関係数は 0，即ち，無相関となります．

**練習 4.1.4 の解答例**　完成 Excel ファイル [ch4-1.xlsx] のシート [cramer_chi2test] 参照

**練習 4.2.1 の解答例**　リーグ別の「打率」の箱ひげ図，省略

**練習 4.2.2 の解答例**　リーグ別の「本塁打」と「盗塁」の散布図，省略

**練習 4.2.3 の解答例**

$$x \text{ の平均} : \bar{x} = \frac{1}{6}(0.247 + 0.242 + \cdots + 0.243) = 0.245$$

$$y \text{ の平均} : \bar{y} = \frac{1}{6}(0.566 + 0.540 + \cdots + 0.444) = 0.499$$

$$x, y \text{ の共分散} : cov_{x,y} = \frac{1}{6}\{(0.247 - 0.245)(0.566 - 0.499) + \cdots\} = 0.00006$$

$$x \text{ の標準偏差} : s_x = \sqrt{\frac{1}{6}\{(0.247 - 0.245)^2 + \cdots + (0.243 - 0.245)^2\}} = 0.00564$$

$$y \text{ の標準偏差} : s_y = \sqrt{\frac{1}{6}\{(0.566 - 0.499)^2 + \cdots + (0.444 - 0.499)^2\}} = 0.04491$$

$$x, y \text{ の相関係数} : \rho_{x,y} = \frac{0.00006}{0.00564 \cdot 0.04491} = 0.2351$$

**練習 5.2.1 の解答例**　「新規印刷レイアウト」での作成例

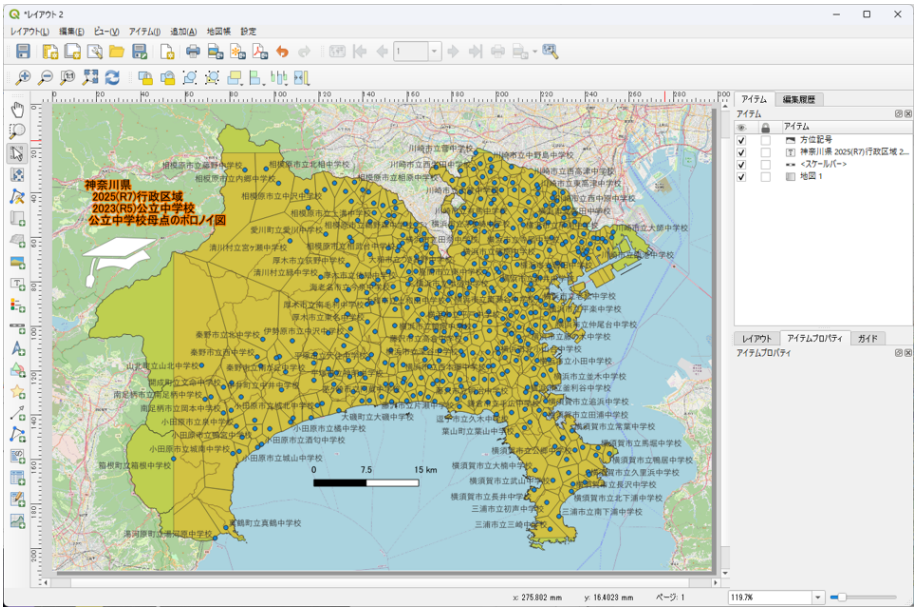



図 1.3　神奈川県の公立中学校母点のボロノイ図を印刷レイアウトで装飾

**練習 5.2.2 の解答例**　実施例

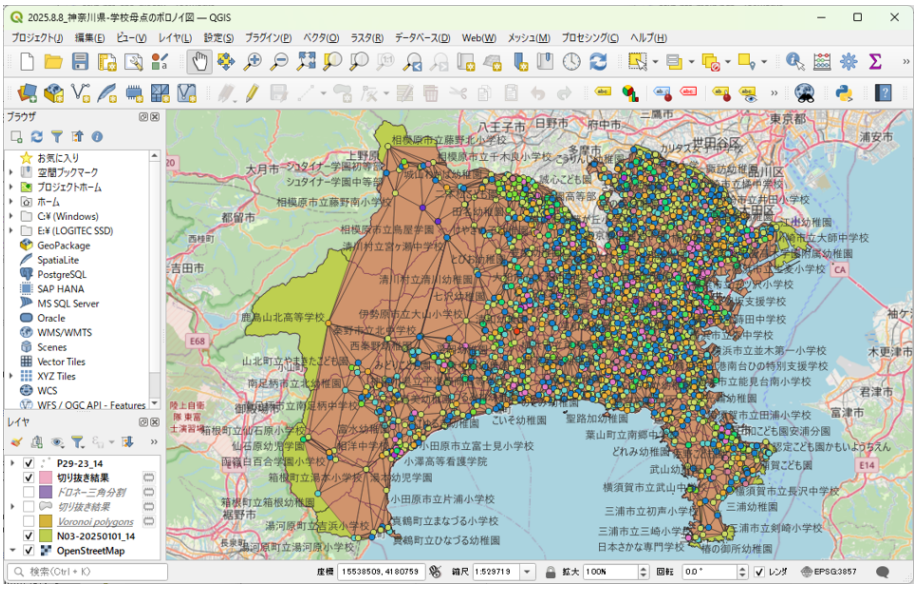


図 1.4　神奈川県の各種学校母点のドロネー図

**練習 6.2.1 の解答例**

1. 囚人のジレンマ型のゲームの各値を $B, S, T, W$ で表すと次の表のとおりです.

| $\alpha\backslash\beta$ | C | D |
|---|---|---|
| C | $(S, S)$ | $(W, B)$ |
| D | $(B, W)$ | $(T, T)$ |

このとき，各値の大小関係は $B > S > T > W$ となっています.
（※ $B$:best, $S$:2nd best, $T$:3rd best, $W$:worst）
$S = \frac{B+W}{2}$ のとき，標準的な囚人のジレンマとよびます．この意味は，双方が協調を繰り返す場合 (C,C) → (C,C) → (C,C) →…と，協調・裏切りを交互に繰り返す場合 (D,C) → (C,D) → (D,C) → (C,D) →…の利得が同じだということです．$S > \frac{B+W}{2}$ や $S < \frac{B+W}{2}$ の場合と比較して，より協調や裏切りが選ばれやすくなるかどうか考察しましょう.
なお，2 人のプレイヤーの 4 つの値は，大小がこの順なら共通の値とする必要はありませんが，同じにしてある方が，各プレイヤーに対する情報として明確です．異なっていると，それ自身が戦略決定に影響を及ぼす可能性があります．例えば，以下の利得表で与えられるゲームを繰返した場合を考えれば分かると思います.

| $\alpha\backslash\beta$ | C | D |
|---|---|---|
| C | $(6, 3)$ | $(4, 9)$ |
| D | $(7, 0)$ | $(5, 2)$ |

$\beta$ の方がより裏切り D を選ぶ動機が高くなるでしょうし，$\alpha$ は協調 C で構わないと思うかもしれません.

2. 省略

**練習 7.2.1 の解答例**　$x_2$ が自由変数，$x_3$ が非正なので，各々 $x_2 = \tilde{x}_2 - \hat{x}_2, x_3 = -\hat{x}_3$ と置き換えます．

$$
\left|\begin{array}{rl}
\min. & 3x_1 - 4(\tilde{x}_2 - \hat{x}_2) + 2(-\hat{x}_3) + 7x_4 \\
\text{s.t.} & 2x_1 + 5(\tilde{x}_2 - \hat{x}_2) - 3(-\hat{x}_3) + x_4 \leq 1 \\
& 3x_1 - (\tilde{x}_2 - \hat{x}_2) + 3(-\hat{x}_3) - 2x_4 \geq 2 \\
& -x_1 + 2(\tilde{x}_2 - \hat{x}_2) - 4(-\hat{x}_3) = 5 \\
& x_1 \geq 0, \tilde{x}_2 \geq 0, \hat{x}_2 \geq 0, \hat{x}_3 \geq 0, x_4 \geq 0
\end{array}\right.
$$

$$
\rightarrow \left|\begin{array}{rl}
\min. & 3x_1 - 4\tilde{x}_2 + 4\hat{x}_2 - 2\hat{x}_3 + 7x_4 \\
\text{s.t.} & 2x_1 + 5\tilde{x}_2 - 5\hat{x}_2 + 3\hat{x}_3 + x_4 \leq 1 \\
& 3x_1 - \tilde{x}_2 + \hat{x}_2 - 3\hat{x}_3 - 2x_4 \geq 2 \\
& -x_1 + 2\tilde{x}_2 - 2\hat{x}_2 + 4\hat{x}_3 = 5 \\
& x_1 \geq 0, \tilde{x}_2 \geq 0, \hat{x}_2 \geq 0, \hat{x}_3 \geq 0, x_4 \geq 0
\end{array}\right.
$$

目的関数が最小化（$\min.$）なので $-1$ 倍して最大化（$\max.$）にし，2 番目の不等式制約を $-1$ 倍し，3 番目の等号制約を 2 つの不等式制約へ置き換えると標準形になります．

$$
\left|\begin{array}{rl}
\max. & -3x_1 + 4\tilde{x}_2 - 4\hat{x}_2 + 2\hat{x}_3 - 7x_4 \\
\text{s.t.} & 2x_1 + 5\tilde{x}_2 - 5\hat{x}_2 + 3\hat{x}_3 + x_4 \leq 1 \\
& -3x_1 + \tilde{x}_2 - \hat{x}_2 + 3\hat{x}_3 + 2x_4 \leq -2 \\
& -x_1 + 2\tilde{x}_2 - 2\hat{x}_2 + 4\hat{x}_3 \leq 5 \\
& x_1 - 2\tilde{x}_2 + 2\hat{x}_2 - 4\hat{x}_3 \leq -5 \\
& x_1 \geq 0, \tilde{x}_2 \geq 0, \hat{x}_2 \geq 0, \hat{x}_3 \geq 0, x_4 \geq 0
\end{array}\right.
$$

**練習 7.2.2 の解答例**　$(x_1, x_2, x_3) = (0, 0, 1)$ は実行可能解の 1 つです．この解から，$x_1 = x_2 = 0$ としたまま $x_3$ を幾らでも大きくでき，それによって目的関数を幾らでも増大できるので非有界です．

**練習 7.2.3**　練習 7.2.1 と同じ問題なので，練習 7.2.1 の解答例（標準形に変換した問題）から双対問題を作っても良いですが，一部そのままで作れます．非正変数 $x_3$ は非負変数 $\hat{x}_3$ に置き換え，目的関数の最大化と不等号制約の向きは揃えておきます．自由変数 $x_2$ と等号制約はそのままです．すると以下となります．

$$
\left|\begin{array}{rl}
\max. & -3x_1 + 4x_2 + 2\hat{x}_3 - 7x_4 \\
\text{s.t.} & 2x_1 + 5x_2 + 3\hat{x}_3 + x_4 \leq 1 \\
& -3x_1 + x_2 + 3\hat{x}_3 + 2x_4 \leq -2 \\
& -x_1 + 2x_2 + 4\hat{x}_3 = 5 \\
& x_1 \geq 0, \hat{x}_3 \geq 0, x_4 \geq 0
\end{array}\right.
$$

これを主問題として考えると，以下の双対問題ができます．

$$
\begin{aligned}
\min.\quad & y_1 - 2y_2 + 5y_3 \\
\text{s.t.}\quad & 2y_1 - 3y_2 - y_3 \geq -3 \\
& 5y_1 + y_2 + 2y_3 = 4 \\
& 3y_1 + 3y_2 + 4y_3 \geq 2 \\
& y_1 + 2y_2 \geq -7 \\
& y_1 \geq 0, y_2 \geq 0
\end{aligned}
$$

標準形からの変換規則に加えて，等号制約に対応する変数は自由変数になることを知っていれば導けます．主問題の 3 つ目の制約は等号なので，対応する双対問題の第 3 変数 $y_3$ は自由変数になり，主問題の自由変数 $x_2$ に対応する双対問題の 2 つ目の制約は等号になります．なぜこれで良いのかは，一度完全に標準形に変換してから双対問題を導いてみたり，本文で述べた上界としての双対問題の導出から分かると思います．

**練習 7.2.4 の解答例** 双対問題 (7.3) の 5 本の制約に対応する重みを非負変数 $x_\alpha, x_\beta, x_\gamma, x_\delta, x_\epsilon$ とし，加重和をとると

$$
\begin{aligned}
& x_\alpha(\ 3y_A + y_B + 2y_D) && 6x_\alpha \\
+\ & x_\beta(\ 2y_B + 4y_C + y_D) && +7x_\beta \\
+\ & x_\gamma(\ y_A + 2y_C\ ) \geq && +3x_\gamma \\
+\ & x_\delta(\ 3y_A + 3y_B + 5y_C + y_D) && +10x_\delta \\
+\ & x_\epsilon(\ y_A + 2y_B + 2y_D) && +5x_\epsilon
\end{aligned}
\tag{1.1}
$$

となります．左辺を $y_i (i \in \{A, B, C, D\})$ で括り直して

$$
\begin{aligned}
(\ & 3x_\alpha + x_\gamma + 3x_\delta + x_\epsilon)y_A && 6x_\alpha \\
+(\ & x_\alpha + 2x_\beta + 3x_\delta + 2x_\epsilon)y_B && +7x_\beta \\
+(\ & 4x_\beta + 2x_\gamma + 5x_\delta\ )y_C \geq && +3x_\gamma \\
+(\ & 2x_\alpha + x_\beta + x_\delta + 2x_\epsilon)y_D && +10x_\delta \\
& && +5x_\epsilon
\end{aligned}
$$

です．双対問題 (7.3) の目的関数 $80y_A + 75y_B + 95y_C + 70y_D$ と左辺を項毎に係数比較し

$$
\begin{aligned}
80 &\geq 3x_\alpha + x_\gamma + 3x_\delta + x_\epsilon \\
75 &\geq x_\alpha + 2x_\beta + 3x_\delta + 2x_\epsilon \\
95 &\geq 4x_\beta + 2x_\gamma + 5x_\delta \\
70 &\geq 2x_\alpha + x_\beta + x_\delta + 2x_\epsilon
\end{aligned}
$$

を満たせば，式 (1.1) の右辺が双対問題 (7.3) の下界になります．このとき，最も良い下界を求める問題が主問題 (7.1) です．

**練習 7.2.5 の解答例** 主問題 (7.2) の制約について非負変数 $\boldsymbol{y}$ で加重和をとり，$\boldsymbol{y}^T(A\boldsymbol{x}) \leq \boldsymbol{y}^T\boldsymbol{b}$ です．左辺を $\boldsymbol{x}$ で括り直し $(\boldsymbol{y}^T A)\boldsymbol{x} = (A^T\boldsymbol{y})^T\boldsymbol{x} \leq \boldsymbol{b}^T\boldsymbol{y}$ です．変数 $\boldsymbol{x}$ が非負なので，主問題の目的関数 $\boldsymbol{c}^T\boldsymbol{x}$ と左辺の係数を比較して，$\boldsymbol{c}^T \leq (A^T\boldsymbol{y})^T$ ならば，右辺 $\boldsymbol{y}^T\boldsymbol{b}$ が上界になります．最も良い上界を見つけたいので，$\min. \boldsymbol{b}^T\boldsymbol{y}$ s.t. $A^T\boldsymbol{y} \geq \boldsymbol{c}, \boldsymbol{y} \geq \boldsymbol{0}$

を求めれば良いですが，これは双対問題 (7.4) そのものです．
逆に，双対問題 (7.4) の制約について非負変数 $\boldsymbol{x}$ で加重和を取り，$\boldsymbol{x}^T(A^T\boldsymbol{y}) \geq \boldsymbol{x}^T\boldsymbol{c}$ です．左辺を $\boldsymbol{y}$ で括り直し $(\boldsymbol{x}^TA^T)\boldsymbol{y} = (A\boldsymbol{x})^T\boldsymbol{y} \geq \boldsymbol{c}^T\boldsymbol{x}$ です．変数 $\boldsymbol{y}$ が非負なので，双対問題の目的関数 $\boldsymbol{b}^T\boldsymbol{y}$ と左辺の係数を比較して，$\boldsymbol{b}^T \geq (A\boldsymbol{x})^T$ ならば，右辺 $\boldsymbol{c}^T\boldsymbol{x}$ が下界になります．最も良い下界を見つけたいので，$\max . \boldsymbol{c}^T\boldsymbol{x}$ s.t. $A\boldsymbol{x} \leq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}$ を求めれば良いですが，これは主問題 (7.2) そのものです．

**練習 7.2.6 の解答例**

1. 食材 $i$ の摂取量を $x_i$ とし，変数は $x_\alpha, x_\beta, x_\gamma, x_\delta, x_\epsilon$ の 5 つ．定式化は以下です．

$$
\begin{array}{lrcrcrcrcrcl}
\min . & 52x_\alpha & + & 64x_\beta & + & 32x_\gamma & + & 46x_\delta & + & 38x_\epsilon & & \\
\text{s.t.} & 2.5x_\alpha & + & 0.7x_\beta & + & 1.3x_\gamma & + & 3.1x_\delta & + & 4.1x_\epsilon & \geq & 70 \\
 & 11x_\alpha & + & 21x_\beta & + & 5x_\gamma & + & 13x_\delta & + & 12x_\epsilon & \geq & 300 \\
 & 0.6x_\alpha & + & 4.2x_\beta & + & 2.5x_\gamma & + & 5.1x_\delta & + & 0.8x_\epsilon & \geq & 50 \\
 & 23x_\alpha & + & 16x_\beta & + & 15x_\gamma & & & + & 12x_\epsilon & \geq & 150 \\
 & x_\alpha & , & x_\beta & , & x_\gamma & , & x_\delta & , & x_\epsilon & \geq & 0
\end{array}
$$

2. 双対問題は，主問題の制約 4 本に対応する双対変数を $y_A, y_B, y_C, y_D$ として

$$
\begin{array}{lrcrcrcrcl}
\max . & 70y_A & + & 300y_B & + & 50y_C & + & 150y_D & & \\
\text{s.t.} & 2.5y_A & + & 11y_B & + & 0.6y_C & + & 23y_D & \leq & 52 \\
 & 0.7y_A & + & 21y_B & + & 4.2y_C & + & 16y_D & \leq & 64 \\
 & 1.3y_A & + & 5x_B & + & 2.5y_C & + & 15y_D & \leq & 32 \\
 & 3.1y_A & + & 13y_B & + & 5.1y_C & & & \leq & 46 \\
 & 4.1y_A & + & 12y_B & + & 0.8y_C & + & 12y_D & \leq & 38 \\
 & y_A & , & y_B & , & y_C & , & y_D & \geq & 0
\end{array}
$$

   となります．
3. 図 1.5 の通りとなります．

```
c = [52, 64, 32, 46, 38]
b = [70, 300, 50, 150]
A = [[2.5, 0.7, 1.3, 3.1, 4.1],
     [11, 21, 5, 13, 12],
     [0.6, 4.2, 2.5, 5.1, 0.8],
     [23, 16, 15, 0, 12]]
J = range(len(c))
I = range(len(b))

from mip.model import *
m = Model("nutrient")  # モデルの設定：線形最適化
x = [m.add_var(var_type="C", lb=0) for j in J]       # 変数宣言：モデル m に変数を追加
m.objective = minimize(xsum(c[j] * x[j] for j in J)) # 目的関数の設定：モデル m に目的関数を追加
for i in I:
  m += xsum(A[i][j] * x[j] for j in J) >= b[i]       # 制約条件の設定：モデル m に制約条件を追加
m.optimize()   # 最適化（求解）の実行

if m.status.value==0:   # もし，最適解が求まったなら
  print("最適解:")     # 最適解を表示
  for j in J:
    print(" x[",j+1,"] = ", x[j].x)
  print("最適値:", m.objective_value, "=", m.objective) # 目的関数値を表示
else:                   # もし，最適解が求まらなかったなら
  print("error:最適解は求まりませんでした")  # エラーメッセージを表示
```

```
最適解:
 x[ 1 ] =  0.0
 x[ 2 ] =  4.114849392284658
 x[ 3 ] =  0.0
 x[ 4 ] =  4.36498150431566
 x[ 5 ] =  13.070283600493218
最適値: 960.8102871234807 = + 52.0var(0) + 64.0var(1) + 32.0var(2) + 46.0var(3) + 38.0var(4)
```

図 1.5　練習 7.2.6 栄養摂取問題のモデル・求解と結果

**練習 7.3.1 の解答例**　問題をグラフに描画するプログラム（図 1.6）と結果（図 1.7）です．また，問題を安定集合として定式化し，最適解を求めて結果をグラフに描画する Python プログラムは図 1.8 の通りで，図 1.9 は得られた最適解のグラフです．

```
%matplotlib inline
import networkx as nx

G = nx.Graph()                              # 空の無向グラフ作成
G.add_nodes_from([1,2,3,4,5,6,7,8,9,10]) # 点集合：設定・追加
G.add_edges_from([(1,6),(1,10),(2,3),(2,5),(2,6),(2,7),(2,8),(2,10),
 (3,6),(3,10),(4,10),(6,7),(6,8),(6,9),(7,8),(7,10),(8,10)]) #枝集合
pos = nx.circular_layout(G)                 # 点の位置(circular_layout)
nx.draw(G, pos, with_labels=True)
```

図 1.6　問題をグラフ描画する Python プログラム

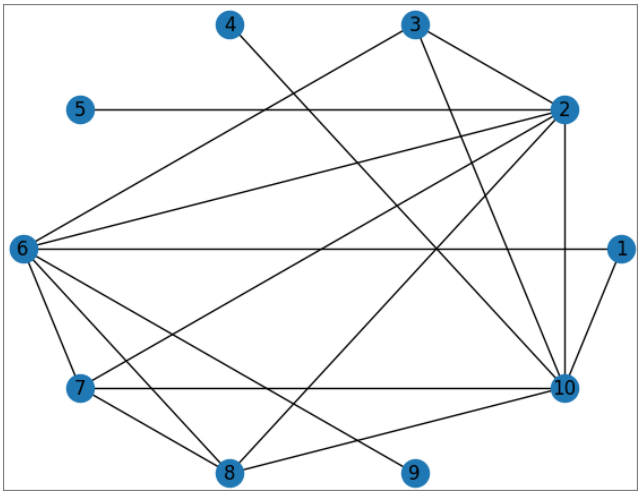



図 1.7　グラフで描画

```
from mip.model import *
m = Model("StableSetEx")  # モデルの設定：安定集合（練習問題）
V = range(G.number_of_nodes())
x = [m.add_var(var_type="B") for v in V]      # 0-1変数
m.objective = maximize(xsum(x[v] for v in V)) # 目的関数
for (i,j) in G.edges():
  m += x[i-1] + x[j-1] <= 1 # 制約：枝(i,j)の高々1点が安定集合に
m.optimize()   # 最適化（求解）の実行

optV = []
print("最適解:")     # 最適解を表示
for v in V:
  if x[v].x==1:
    optV.append(v+1)
    print(" x[",v+1,"] = ", x[v].x)
print("最適値:", m.objective_value, "=", m.objective) # 目的関数値を表示

GR = G.copy()   # 元のグラフ G を描写用グラフ GR へコピー
GR.add_nodes_from(optV, color='red')  # 安定集合に含まれる点の色を赤に
vcol_dict = nx.get_node_attributes(GR, "color")       # それ以外の点を灰色に
vcol = [vcol_dict[v] if v in vcol_dict else 'lightgray' for v in GR.nodes()]
nx.draw(GR, pos, with_labels=True, node_color=vcol)   # グラフ描画
```

```
最適解:
 x[ 1 ] =  1.0
 x[ 3 ] =  1.0
 x[ 4 ] =  1.0
 x[ 5 ] =  1.0
 x[ 7 ] =  1.0
 x[ 9 ] =  1.0
最適値: 6.0 = + var(0) + var(1) + var(2) + var(3) + var(4) + var(5) + var(6)
```

図 1.8　練習 7.3.1 問題のモデルと最適解の求解結果とグラフ描画する Python プグラムと最適解

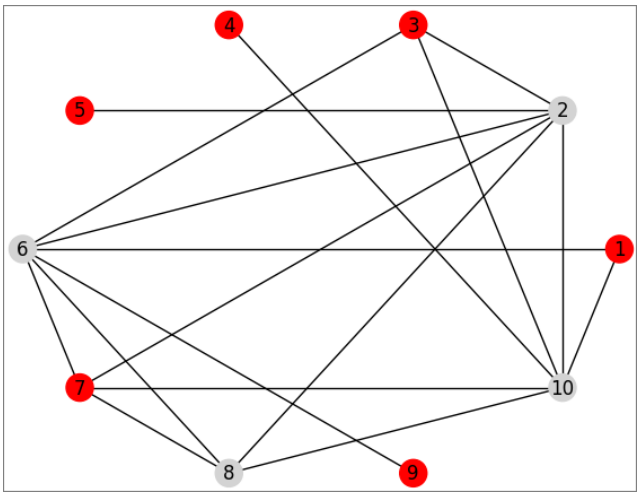
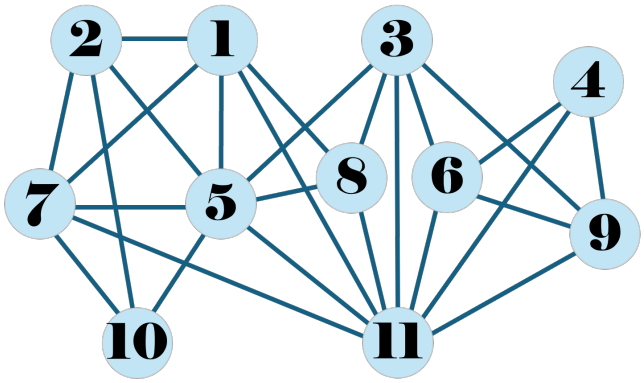



図 1.9　練習 7.3.1 最適解のグラフ描画

**練習 7.3.2 の解答例**

1. 各仕事を点 $i$ とし，時間帯が重なっている 2 つの仕事 $i, j$ 間に枝 $(i, j)$ を張るグラフ $G$ をつくると，下図の通りとなります．




   このグラフの点彩色問題（彩色数最小）を解くと，求める答えが得られます．なぜなら，割当色が従業員に該当し，彩色数が必要人数になるからです．このジョブスケジューリング問題を点彩色問題としてグラフ描画する Python プログラムは図 1.10 で，その描画結果は図 1.11 です．
2. データ（$V, E$）が異なるだけで，定式化は (7.9) と同じです．
3. **練習 7.3.2** のモデルは本文の例題と全く同じなので，結果のみ示します．得られた最適解は図 1.12 と図 1.13 です．

```
%matplotlib inline
import networkx as nx

G = nx.Graph()                                  # 空の無
G.add_nodes_from([1,2,3,4,5,6,7,8,9,10,11]) # 点集合
G.add_edges_from([(1,2),(1,5),(1,7),(1,8),(1,11),
 (2,5),(2,7),(2,10),(3,5),(3,6),(3,8),(3,9),(3,11),
 (4,6),(4,9),(4,11),(5,7),(5,8),(5,10),(5,11),
 (6,9),(6,11),(7,10),(7,11),(8,11),(9,11)]) # 枝集合
nx.draw(G,with_labels=True)
K = V = range(G.number_of_nodes())
```

図 1.10　**練習 7.3.2** の問題をグラフ描画する Python プログラム

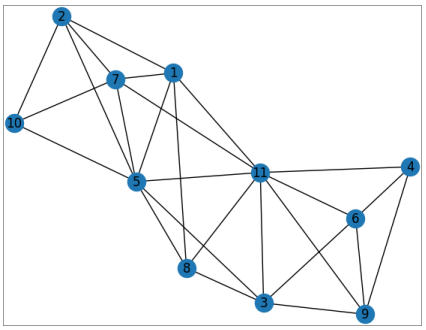



図 1.11　描画結果

```
最適解:
 y[ 1 ]= 1.0 x[ 1 , 1 ]= 1.0 x[ 3 , 1 ]= 1.0 x[ 4 , 1 ]= 1.0 x[ 10 , 1 ]= 1.0
 y[ 2 ]= 1.0 x[ 6 , 2 ]= 1.0 x[ 7 , 2 ]= 1.0 x[ 8 , 2 ]= 1.0
 y[ 3 ]= 1.0 x[ 2 , 3 ]= 1.0 x[ 11 , 3 ]= 1.0
 y[ 4 ]= 1.0 x[ 5 , 4 ]= 1.0 x[ 9 , 4 ]= 1.0
最適値: 4.0 = + var(121) + var(122) + var(123) + var(124) + var(125) + var(126
```

図 1.12　**練習 7.3.2** の最適解

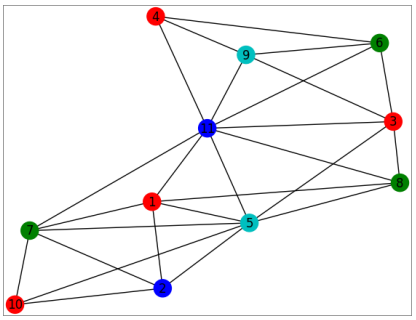



図 1.13　最適解をグラフで視覚化

**練習 8.1.1 の解答例**　各作業の先行関係をグラフで描画する Python プログラムと実行結果は図 1.14 で，解いた結果をグラフで描画したものが図 1.15 です．図 1.15 の各点 (作業) の位置（$x$ 座標）を開始時刻にして描き直すと図 1.16 となります．

```
%matplotlib inline
import networkx as nx

G = nx.DiGraph()      # 空の有向グラフ作成：生産工程の先行関係を表すグラフ
G.add_nodes_from([1,2,3,4,5,6,7,8,9,10,11,12,13,14])  # 点集合：設定・追加
G.add_edges_from([(1,2),(2,3),(3,4),(2,5),(5,6),(5,7),(1,8),(2,9),(4,9),
      (6,9),(7,10),(8,10),(9,11),(9,12),(10,13),(12,13),(11,14),(13,14)])
pos = {1:(0,3),2:(1,4),3:(2,5),4:(3,5),5:(2,3),6:(3,3),7:(3,2),8:(2,1),
                    9:(4,4),10:(4,2),11:(5,4),12:(5,3),13:(6,2),14:(7,3)}
nx.draw(G, pos, with_labels=True)

c = [5.4,6.2,4.9,2.4,5.7,9.4,7.6,7.3,6.8,4.8,6.5,5.1,8.1,8.2] # 生産時間
s = 0.10  # 工程間の余裕時間
I = range(len(c))

from mip.model import *
m = Model("ProductionPlanning")  # モデルの設定：生産スケジューリング
t = [m.add_var(var_type="C", lb=0) for i in I] # 変数宣言
m.objective = minimize(xsum(t[i] for i in I))  # 目的関数
for (i,j) in G.edges():
  m += t[j-1] - t[i-1] >= c[i-1] + s           # 制約
m.optimize()   # 最適化（求解）の実行

print("最適解:")     # 最適解を表示
for i in I:
  print(" t[",i+1,"] = ", t[i].x)
print("最適値:", m.objective_value, "=", m.objective) # 目的関数値を表示
```

```
最適解:
 t[ 1 ] =  0.0
 t[ 2 ] =  5.5
 t[ 3 ] =  11.8
 t[ 4 ] =  16.8
 t[ 5 ] =  11.8
 t[ 6 ] =  17.6
 t[ 7 ] =  17.6
 t[ 8 ] =  5.5
 t[ 9 ] =  27.1
 t[ 10 ] =  25.3
 t[ 11 ] =  34.0
 t[ 12 ] =  34.0
 t[ 13 ] =  39.2
 t[ 14 ] =  47.400000000000006
最適値: 293.6 = + var(0) + var(1) + var(2) + var(3) + var(4) + var(5) + va
```

図 1.14　8.1 節 練習 8.1.1 生産スケジューリング問題の描画とモデル

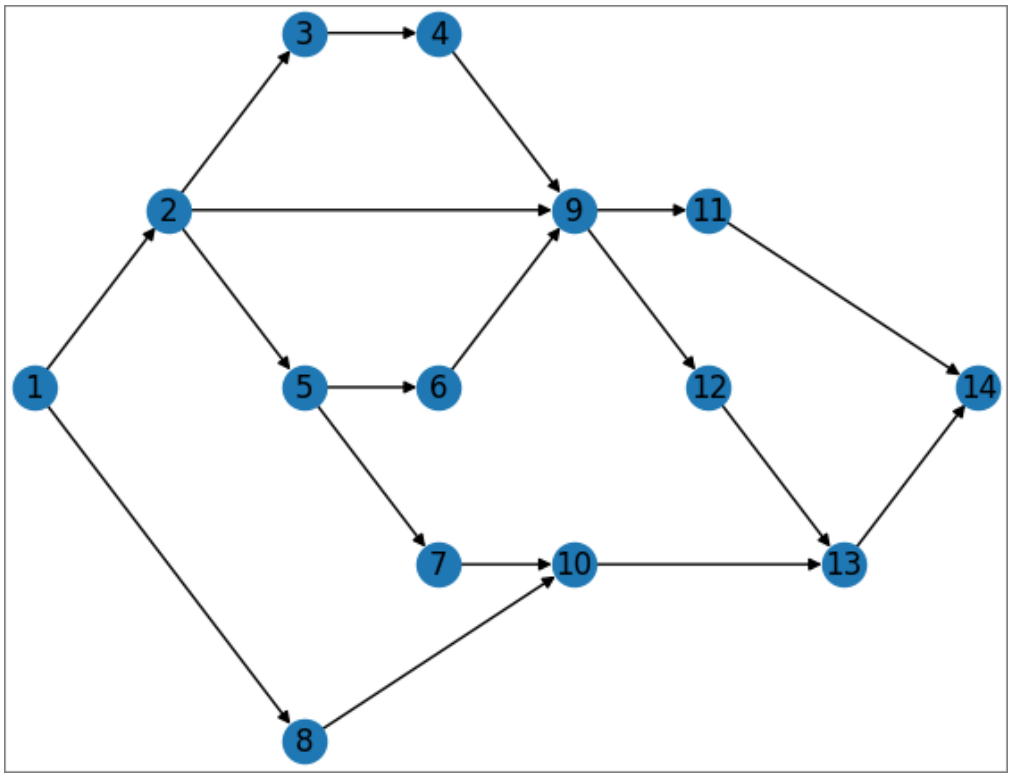



図 1.15　8.1 節 練習 8.1.1 生産スケジューリング問題の求解結果を描画

```
for i in I:
  pos[i+1] = (t[i].x, pos[i+1][1])  # 位置(x座標)をt[i].xで置換
nx.draw(G, pos, with_labels=True, node_color='red')
```

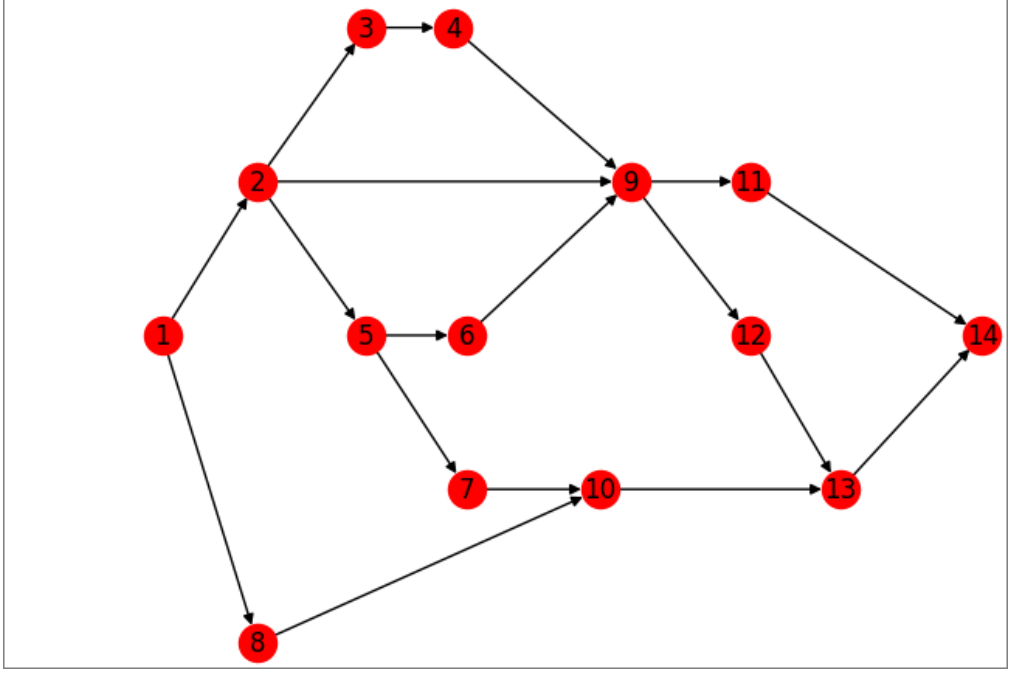



図 1.16　各点（作業）の位置（$x$ 座標）を開始時刻にして最適解を再描画

**練習 8.2.1 の解答例**　省略

**練習 8.3.1 の解答例**

1. チーム数が奇数の場合は，架空のチームを 1 チーム加えて偶数にして求めればよいからです（チーム数が 2 の場合は，わざわざ定式化して解くまでもないでしょう）．架空チームと対戦するチームは休みとなります．このとき，架空チームのホーム/アウェイを考慮した方が良い場合もあるでしょう．例えば，架空チームは全試合をホームで行うことにすれば，対戦相手（休みのチーム）はアウェイに 1 回行ったことに出来ます．
2. どのチームがどこの本拠地で対戦する場合でも，全チームの移動距離の総和は $\sum_{i=1}^{10}\sum_{j=1}^{10} d_{ij}$ となります．ただし，$d_{ij} = d_{ji} (\forall i, j \in T), d_{ii} = 0 (\forall i \in T)$ です．
3. ホームゲームでは応援者が多く，アウェイの場合は応援にいける人が少なくなります．アウェイ側は移動が必要で，時間と費用が掛かります．移動や宿泊で疲労します．その間十分な練習も出来ません．ホーム側は入場料や付随店舗の利益が得られます（契約によるでしょうが）．など．
4. ホーム-アウェイ・パターンが同じ 2 チームは対戦が出来ないからです．
5. ブレイクが 0 となるのは，ホームゲームで始まり HA が交互になる HAHA... と，アウェイゲームで始まり AH が交互になる AHAH... の 2 パターンのみで，前設問の答えより，このパターンが使えるのは 2 チームだけです．それ以外のチームは必ずブレイク 1 以上となるので，ブレイク数総和の下界は $2n-2$ となります．同様に，1 チームのブレイク数が最大の $|S|-1 (= 2n-2)$ となるのは，全てホームゲームか全てアウェイゲームの 2 パターンのみで，これが割り当てられるのは最大でも 2 チームです．それ以外は $2n-3$ 以下となるので，ブレイク数総和の自明な上界は $2(2n-2)+(2n-2)(2n-3) = (2n-2)(2n-1)$ です．例えば $n = 3 (6$ チーム$)$ の場合，下界は $2 \times 3 - 2 = 4$ で，上界は $(2 \times 3 - 2)(2 \times 3 - 1) = 20$ です．