最適化課題 (解説編)

スタートアップゼミ2023 フィードバック

2023/5/8

M1 林由翔

問題概要

- 震災で破壊された**道路の最適復旧計画**を求めてください．

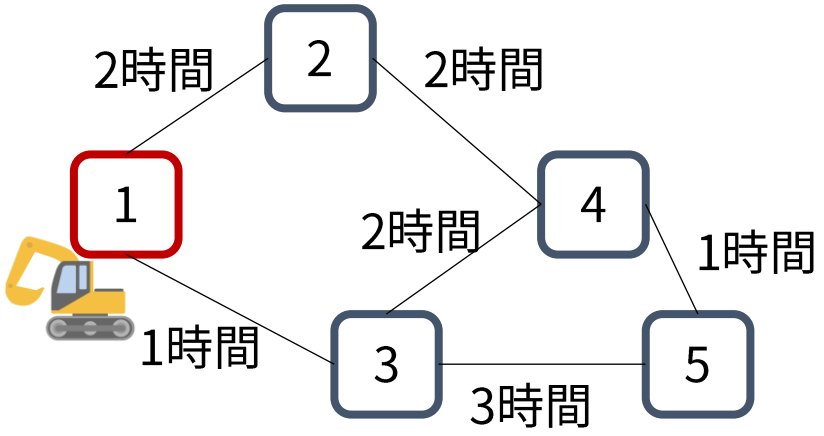
- 2つの問題を出すので，どちらかを解いてください．
  - 他の問題を自主的に考えて解いても良いです．

問題①

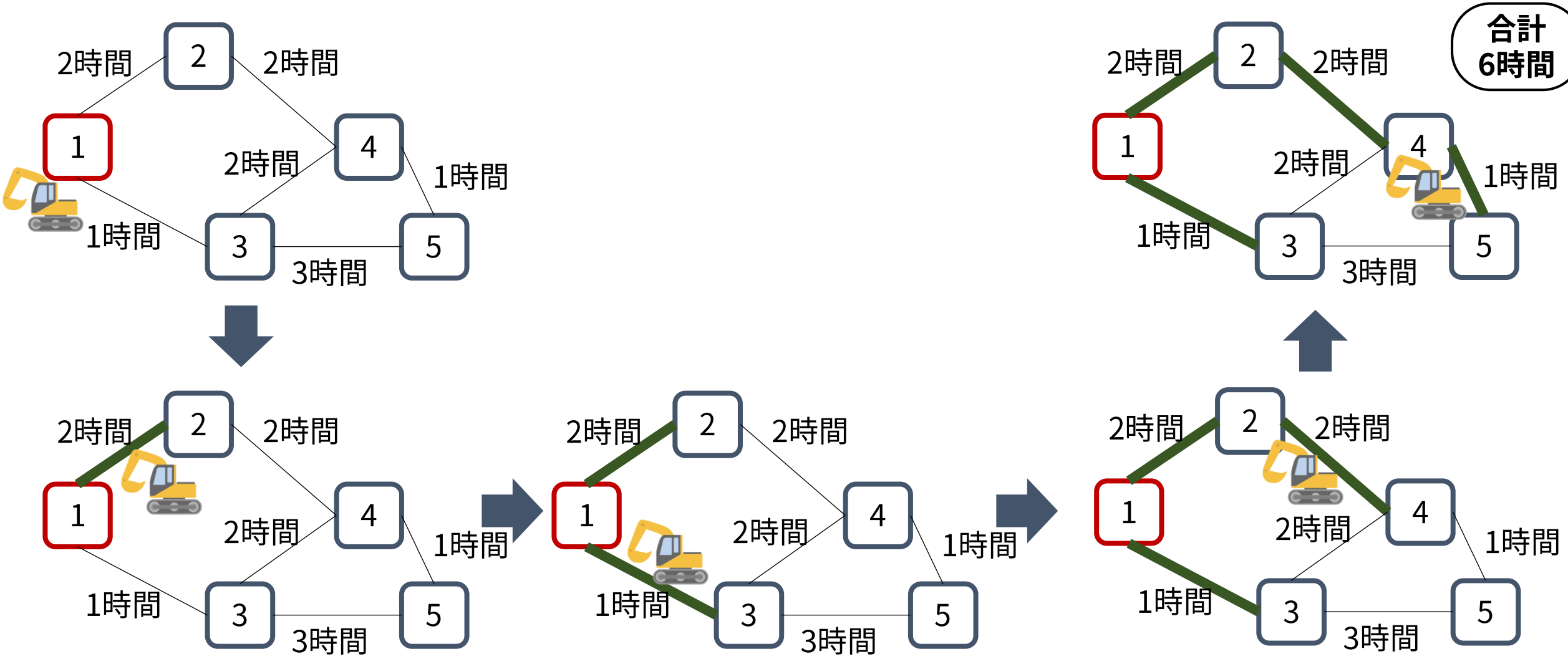
いくつかの集落と，各集落をつなぐ道があります．1番目の集落には防災拠点があります．

震災により全ての道が壊れてしまいました．そこで防災拠点から1台の重機を発進させ，道を修繕します．

道ごとに修繕にかかる時間が異なります．また修繕済みの道路は所要時間0で重機が移動できます．そこで，**全ての集落が防災拠点と道で結ばれるまでにかかる時間を最小化**するように，修繕する道を決めてください．

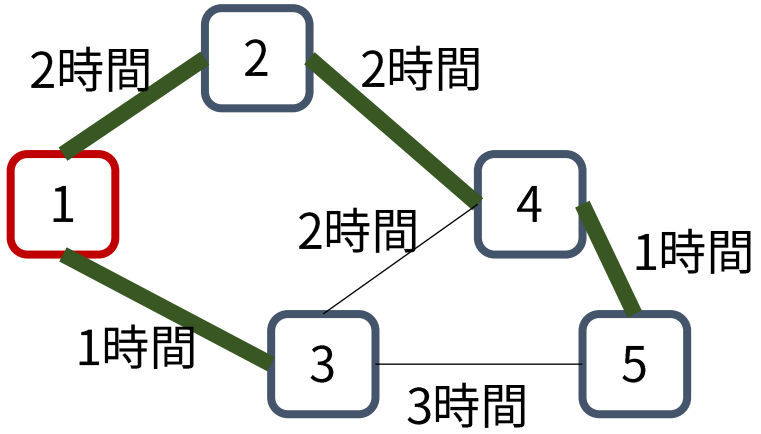


問題①—例



問題②

各集落が防災拠点と結ばれるまでの所要時間の平均を最小化する場合はどうすればよいか.



- 集落1: 0時間後に防災拠点と接続
- 集落2: 2時間後に防災拠点と接続
- 集落3: 3時間後に防災拠点と接続
- 集落4: 5時間後に防災拠点と接続
- 集落5: 6時間後に防災拠点と接続

→平均3.2時間

1. 整数計画法
2. 最小木問題
3. 動的計画法

整数計画法

キーワード: 目的関数, 制約条件, PuLP, 木, 連結

整数計画法

定式化してやればPuLPなどライブラリで解ける.

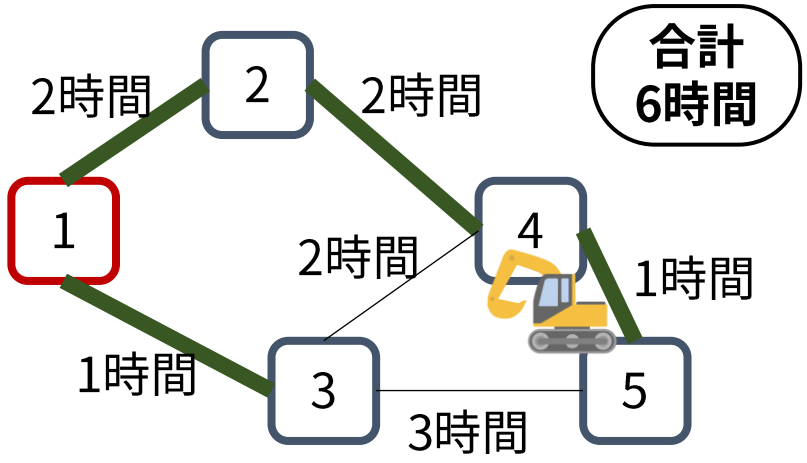
目的関数と**制約条件**を決める

整数計画法 問題1

問題1: 全ての集落が防災拠点と道で結ばれるまでの時間を最小化する.

観察1: 工事する順序は最小化する所要時間に関係しない.

- 集落 3→2→4→5 と行っても 2→4→5→3 と行っても良い



整数計画法 問題1

順序を無視し，繋ぐ道路を選ぶだけの最小化問題として記述する．

x_i：道路iを使用するかどうか (0 or 1)

c_i：道路iを修繕するのにかかる時間

目的関数 $\min_{\boldsymbol{x}} \boldsymbol{cx} = \min_{\boldsymbol{x}} \sum_i c_i x_i$

整数計画法 問題1

順序を無視し，繋ぐ道路を選ぶだけの最小化問題として記述する．

x_i：i番目の道路を修繕するかどうか

c_i：i番目の道路を修繕するのにかかる時間

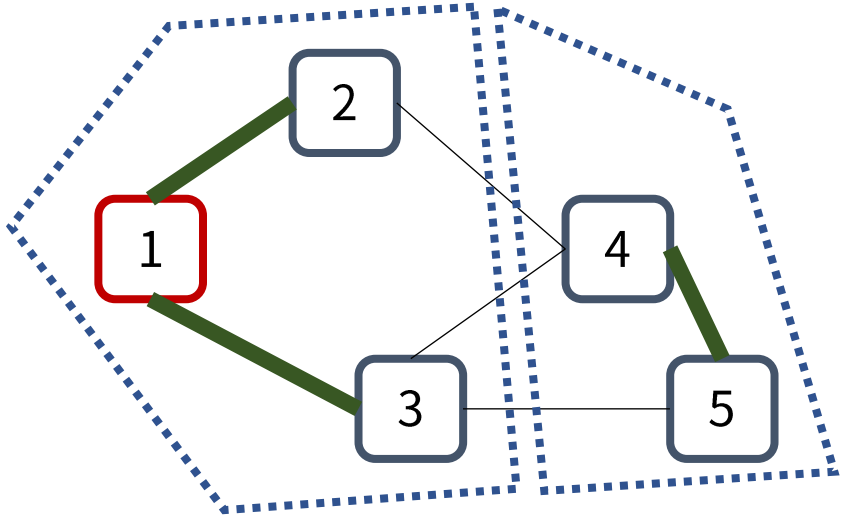
制約条件

防災拠点(集落1)から他のすべての集落に行くことができる (連結性条件)

↑どう記述する？

整数計画法 問題1

アウトな場合を考える



二つに割れてたらアウト！

整数計画法 問題1

集落の集合を $V = \{1,2,3,4,5\}$ とおく．

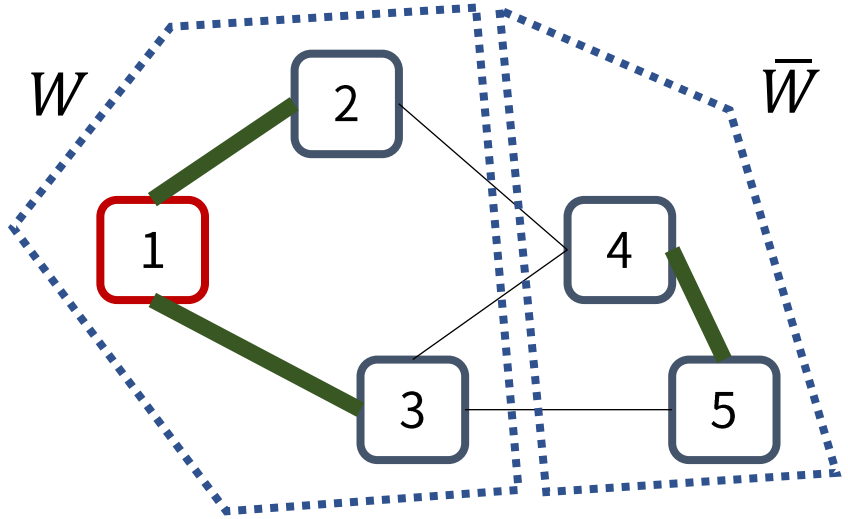
ある $W \subset V$ であって，W と $\bar{W}$ をつなぐ道路が一本もないようなWが存在したらアウト．

補集合

制約条件 $\forall W \subset V: \sum_{i \in \delta W} x_i \geq 1$

x_i: i番目の道路を修繕するかどうか
どれか1本以上直ってたらOK

ただし δW はW と $\bar{W}$ をつなぐリンクの集合．



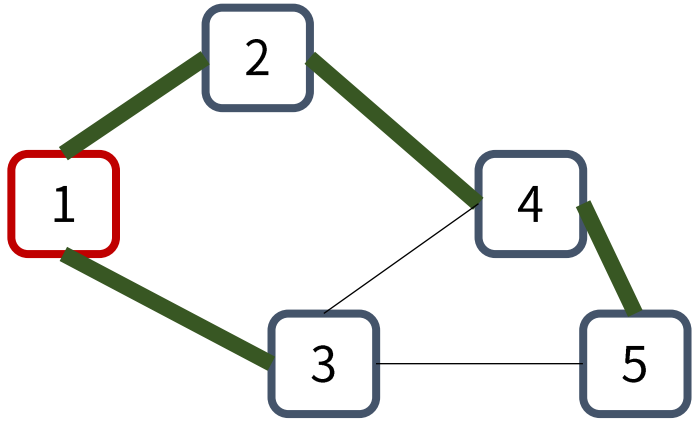
整数計画法 問題1

補足

- 実装上は，全ての部分集合について$\sum_{i\in\delta W} x_i \geq 1$を制約条件として書く
  - 制約条件の数はN集落なら最悪$O(2^N)$個
- $\forall W \subset V: \sum_{i\in\delta W} x_i \geq 1$ が十分条件であることは構成的に示せます．

整数計画法 問題1

- 順序はどう構成するか
  →深さ優先探索などで実装可能.



整数計画法 問題2

問題2: 各集落が防災拠点と道で結ばれるまでの時間の平均を最小化する.

観察2-1: 平均の代わりに合計を最小化すればよい.

観察2-2: 直す時刻ではなく直す順序に着目する.

i番目に修繕する道路の修繕所要時間をt_iとする.

目的関数(合計値)は,

$$t_1 + (t_1 + t_2) + (t_1 + t_2 + t_3) + \cdots + (t_1 + \cdots + t_N)$$
$$= Nt_1 + (N-1)t_2 + \cdots + t_N$$

整数計画法 問題2

$x_{i,j}$：道路iをj番目に使用するかどうか (0 or 1)

c_i：道路iを修繕するのにかかる時間

目的関数

$$\min_{s} \sum_{i} s_i c_i$$

$$s.t.$$

制約条件

$$\forall i:\ s_i = \sum_{j} (N - j + 1) x_{i,j}$$ sの定義

$$\forall j: \sum_{j} x_{i,j} \leq 1$$ 道路は高々1回しか選ばれない

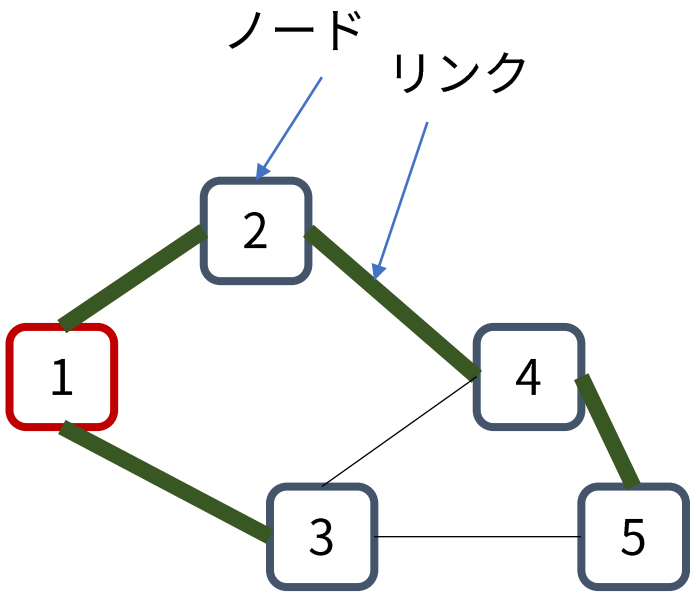
$$\forall W \subset V: \sum_{i \in \delta W} \sum_{j} s_{i,j} \geq 1$$ 連結性条件

最小全域木 (問題1のみ)

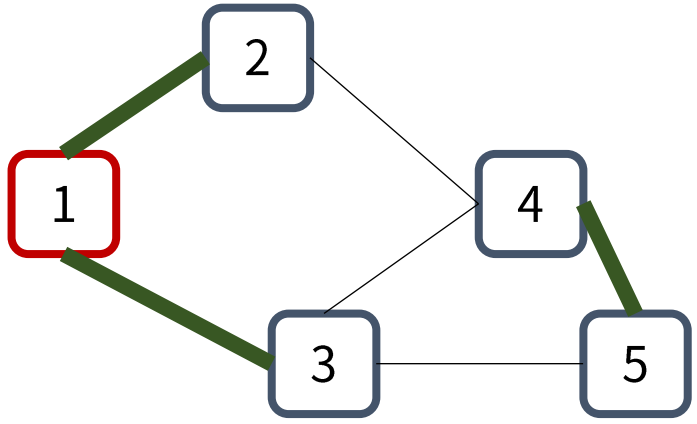
キーワード: クラスカル法, プリム法, マトロイド, 離散凸解析

最小全域木

- グラフが与えられるので，全てのノードが連結であるようにしつつ，リンクコストが最小になるようにリンクを選ぶ問題.
- 問題1の順序がないバージョンそのもの



連結である



連結でない

最小全域木

クラスカル法または**プリム法**と呼ばれるアルゴリズムにより，リンクの数にほぼ比例する時間で計算可能．高速！

クラスカル法

リンクを小さい順に見ていく．もしそのリンクの両端の頂点が連結でなければ，そのリンクを採用する．
https://www.youtube.com/watch?v=mN__iQ2zfjs

プリム法

幅優先探索の要領で，その時点で使える一番短いリンクを採用していく．
https://www.youtube.com/watch?v=oDnlIP5pe5o

問題1はプリム法を用いるとそのまま解ける．
クラスカル法の場合は**深さ優先探索**等で順序を復元する必要あり

クラスカル法とマトロイド

この後のスライドでは，クラスカル法の正当性を示すため**マトロイド**という概念を導入します．

- プリム法の正当性証明にも使えるらしいが難しいのでここでは扱いません．
  - https://mizuwater0.hatenablog.com/entry/2021/06/17/214506 このブログで触れている．
- 正当性の証明の主要部は https://kopricky.github.io/code/Graph_SP_MST/matroid.pdf による．

最小全域木とマトロイド

最小全域木問題が効率的に解けるのは，**マトロイド**という都合のいい構造のおかげ．

マトロイド

マトロイドとは，有限集合Eとその部分族$F \subset 2^E$の組であって，以下を満たすもの．

1. $0 \in F$
2. $X \in F$ かつ $Y \subseteq X$ ならば $Y \in F$
3. $X, Y \in F$ かつ $|X| > |Y|$ ならば，ある $e \in X - Y$ であって$Y \cup \{e\} \in F$となるものが存在する．

マトロイドの例：ベクトルマトロイド

行列 $E = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots \boldsymbol{a}_N]$からいくつか列ベクトルを抜き取ってくる．
matrix

$$F = \{X \subset E \mid X\text{は線形独立}\}$$

とすると，(E, F)はマトロイド．
matroid

例)

$$E = \begin{pmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

としたとき，

$$F = \{\emptyset, \{\boldsymbol{a}_1\}, \{\boldsymbol{a}_2\}, \{\boldsymbol{a}_3\}, \{\boldsymbol{a}_4\}, \{\boldsymbol{a}_1, \boldsymbol{a}_2\}, \{\boldsymbol{a}_1, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_3, \boldsymbol{a}_4\}\}$$

$\{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_1\} \in F$ の時たしかに $\{\boldsymbol{a}_1, \boldsymbol{a}_2\} \in F$になっている．
X　Y　$a_2 \in X - Y$　$Y \cup \{a_2\}$

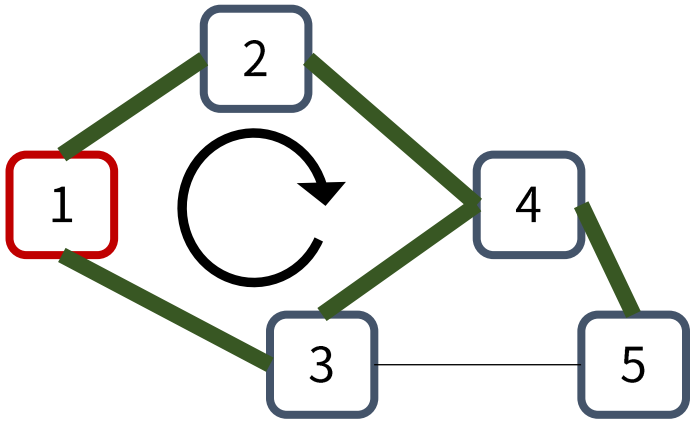
マトロイドの例：閉路マトロイド

グラフの辺集合Eから何本か選んでくる．

$$F = \{X \subset E \mid X\text{は閉路をつくらない}\}$$

とすると，(E, F)はマトロイド．

※閉路



最小全域木とマトロイド

Fの要素であってサイズが極大のものを，そのマトロイドの**基**という．

例)

$$E = \begin{pmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

としたとき，

$$F = \{\emptyset, \{\boldsymbol{a}_1\}, \{\boldsymbol{a}_2\}, \{\boldsymbol{a}_3\}, \{\boldsymbol{a}_4\}, \{\boldsymbol{a}_1, \boldsymbol{a}_2\}, \{\boldsymbol{a}_1, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_3, \boldsymbol{a}_4\}\}$$

基は$\{\boldsymbol{a}_1, \boldsymbol{a}_2\}, \{\boldsymbol{a}_1, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_3, \boldsymbol{a}_4\}$.

基交換定理

基交換定理

B_1, B_2 が (E, F) の異なる基とする．

$$\forall x \in B_1 - B_2 \; \exists y \in B_2 - B_1 \text{ s.t. } B_1 - \{x\} + \{y\} \text{ は基}$$

例)

$$E = \begin{pmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

としたとき，

$$F = \{\emptyset, \{\boldsymbol{a}_1\}, \{\boldsymbol{a}_2\}, \{\boldsymbol{a}_3\}, \{\boldsymbol{a}_4\}, \{\boldsymbol{a}_1, \boldsymbol{a}_2\}, \{\boldsymbol{a}_1, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_3, \boldsymbol{a}_4\}\}$$

基は$\{\boldsymbol{a}_1, \boldsymbol{a}_2\}, \{\boldsymbol{a}_1, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_3\}, \{\boldsymbol{a}_2, \boldsymbol{a}_4\}, \{\boldsymbol{a}_3, \boldsymbol{a}_4\}$.

$B_1 = \{\boldsymbol{a}_1, \boldsymbol{a}_2\}, B_2 = \{\boldsymbol{a}_3, \boldsymbol{a}_4\}, x = \boldsymbol{a}_2$とすると，$y = \boldsymbol{a}_3$ を選べば $B_1 - \{x\} + \{y\} = \{\boldsymbol{a}_1, \boldsymbol{a}_3\}$で基

基交換定理

基交換定理

B_1, B_2 が (E, F) の異なる基とする．

$$\forall x \in B_1 - B_2 \; \exists y \in B_2 - B_1 \text{ s.t. } B_1 - \{x\} + \{y\} \text{ は基}$$

証明

性質2より$B_1 - \{x\} \in F$．また$|B_1| = |B_2|$ (性質3から背理法で示せる)
なので $B_2, B_1 - \{x\}$ に対して性質3を適用すればよい． ■

マトロイド(再掲)

マトロイドとは，有限集合Eとその部分族$F \subset 2^E$の組であって，以下を満たすもの．

1. $0 \in F$
2. $X \in F$ かつ $Y \subseteq X$ ならば $Y \in F$
3. $X, Y \in F$ かつ $|X| > |Y|$ ならば，ある $e \in X - Y$ であって $Y \cup \{e\} \in F$となるものが存在する．

最小全域木とマトロイド

グラフの辺集合Eから何本か選んでくる．

$$F = \{X \subset E \mid X\text{は閉路をつくらない}\}$$

とすると，(E, F)はマトロイド．

Fの要素であってサイズ最大のものは，明らかに全てのノードをつなぐ=全域木．

→最小全域木問題は**基での最小化問題**．

最小基問題

Eの各要素eにコスト$c(e)$が決められているとする．

$$\min_{B} \sum_{b \in B} c(b)$$

s.t. B はマトロイドの基

クラスカル法とその証明

クラスカル法：最小基問題を解くアルゴリズム

- E の要素 e を $c(e)$ の小さい順にソートした列を $e_1, e_2, \ldots e_n$ とする．
- 集合 $X \leftarrow \emptyset$ で初期化する．
- 以下の操作を $i = 1,2, \ldots, n$ に対して順に実行する．
  - 操作：もし $X \cup \{e_i\} \in F$ ならば，$X \leftarrow X \cup \{e_i\}$

補題

X がある最小基 B^* の部分集合であるとする．ある基 Bであって，$X \cup \{e\} \subset B \subset B^* \cup \{e\}$なるものが存在する．

証明

$X \cup \{e\} \subset B'$ なる B' は基の定義(極大な独立集合)より存在する．この B' が $B^* \cup \{e\}$ に含まれないとき，$x \in B' - (B^* \cup \{e\})$ なる x を適当に選ぶことができる．基交換定理により新たな基 $B'' = B' - \{x\} + \{y\}$ (ただし $y \in B^* - B'$)を構成することができる．この交換1回ごとに基に含まれる$B^* \cup \{e\}$の要素を1つ増やすことができるため，いずれは$B \subset B^* \cup \{e\}$に到達できる．

■

クラスカル法とその証明

クラスカル法：最小基問題を解くアルゴリズム

- E の要素 e を $c(e)$ の小さい順にソートした列を $e_1, e_2, \ldots e_n$ とする．
- 集合 $X \leftarrow \emptyset$ で初期化する．
- 以下の操作を $i = 1,2, \ldots, n$ に対して順に実行する．
  - 操作：もし $X \cup \{e_i\} \in F$ ならば，$X \leftarrow X \cup \{e_i\}$

クラスカル法の正当性の証明

X がある最小基 B^* の部分集合であることを，Xがgreedyであると呼ぶことにする．greedyなXに対して操作をした後でも X がgreedyであることを示せばよい．

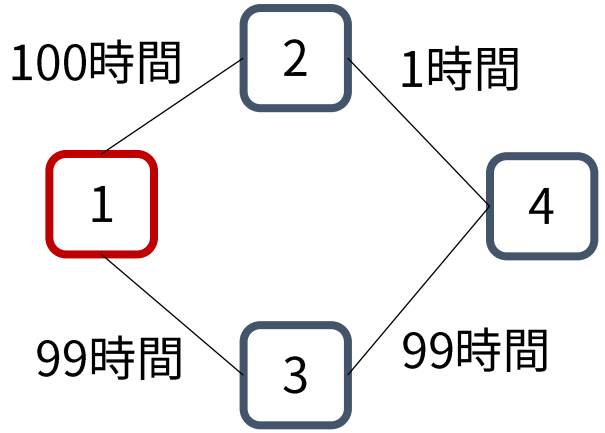
補題より，ある基 Bであって $X \cup \{e\} \subset B \subset B^* \cup \{e\}$なるものが存在する．このときある$e'$であって$B - \{e\} + \{e'\} = B^*$ なるものが存在する．eはソート済みだったので $c(e) \leq c(e')$である．ゆえにBのコストはB^*以下であり，Bはgreedy．よって$X \cup \{e\}$ もgreedy．■

動的計画法

キーワード: bit DP

動的計画法 (問題2)

- プリム法そのままで一見上手く行きそうだが失敗する．



- プリム法だと1→3→4→2が出力されるが，問題2では1→2→4→3でつなぐのが最適．

動的計画法 (問題2)

bit DP：ある集合の部分集合を状態として管理する

長さ2^Nの配列 dp を用意し，

- dp[b] = 集落の部分集合bのみをつなぐときの最小コスト

とすると，dpの要素をbの小さい順に計算していくことで解ける．

```
dp = [INF] * (1 << N)  # 1<<N=2のN乗
dp[0] = 0
for b in range(1 << N):
  for i in range(N):
    if not (b >> i) & 1:
      continue  # ノードiが集合bに含まれないのであればスキップ
  for adj_node, cost in G[i]:
    dp[b ^ (1 << adj_node)] = min(dp[b ^ (1 << adj_node)],
                                  dp[b] + cost)
```

動的計画法 (問題2)

- bit DP の計算量は指数オーダー ($O(N2^N)$など)になる．まともに計算できるのは$N \leq 17$程度．