



最小木問題

Minimum Tree Problem

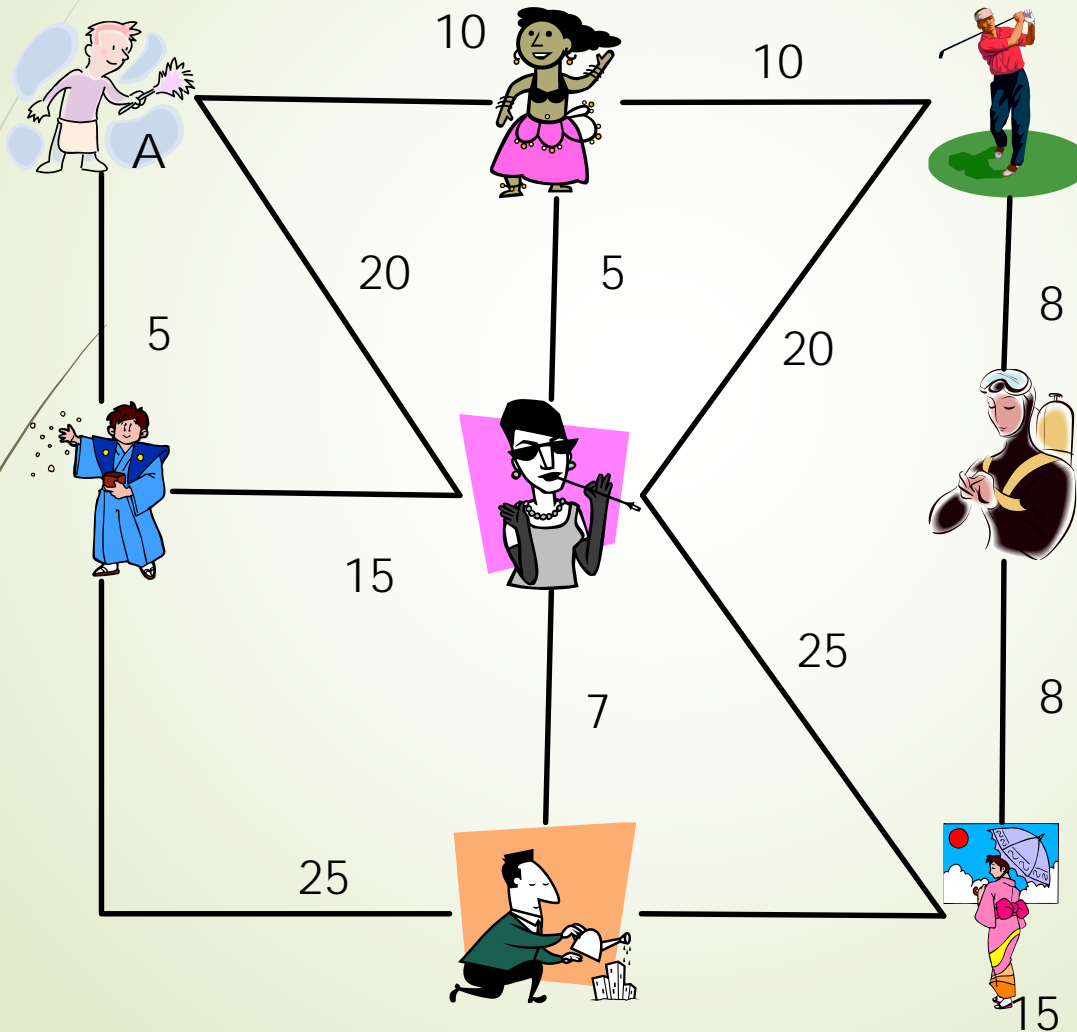
2017年度

担当：只木進一（工学系研究科）

ネットワーク(networks)

- ➡ 弧に長さ、重み、費用などの属性のあるグラフ
 - ➡ 都市とそれを結ぶ交通
 - ➡ 都市間の道路距離
 - ➡ 都市間の鉄道の運賃
 - ➡ 都市間の空路の最大輸送可能人数
 - ➡ コンピュータとネットワーク：帯域
 - ➡ 作業工程：所用時間、遅れ

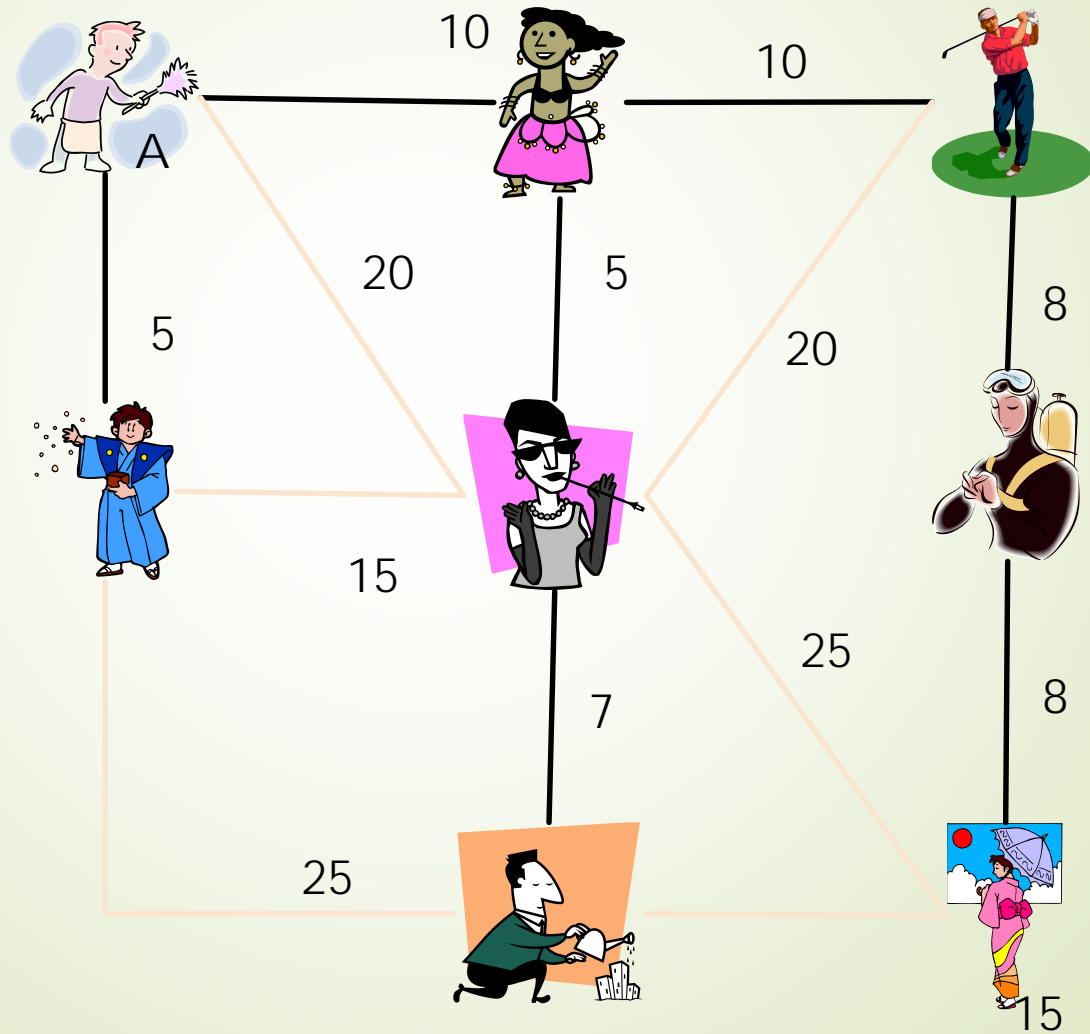
例：最安の連絡経路



Aから全員に最も
安く連絡する経路
総経費を考える

経路の経費が
定義されてい
る

例：最安の連絡経路：解



最小木問題(Minimum Tree Problem)

- 連結無向グラフ $G = (V, A)$
- 重み関数 $w: A \rightarrow R$
 - 各弧に実数に対応：重み、距離、etc.
 - 弧の重みは正 $\forall a: w(a) > 0$
- G の極大木 $T \subseteq A$
 - 重みが最小になる極大木 T を見付ける

$$\min_T w(T)$$

$$w(T) = \sum_{a \in T} w(a)$$

最小木問題の応用

- ▶ 油井（ゆせい）から精油所へパイプラインを引く
 - ▶ 最短(経費の最も安い)のパイプラインで一カ所に原油を集める
- ▶ 最小のコストでコンピュータを繋ぐ
- ▶ 通信コストを最小にして事業所を繋ぐ

Kruskal法 (貪欲法、Greedy法)

- 重み最小の弧を順に選ぶ
- 構成途中は木になっていない (部分木の集合)
- 閉路ができないように制限しながら弧を選択する

貪欲アルゴリズム (Greedy algorithm, Kruskal algorithm)

$T = \emptyset$

$H:G$ は弧の重みに関するヒープ:最初に全弧を登録

while (T は G の極大木ではない){

$a = H.poll()$ //ヒープから最小要素を取得

$a_{new} = null$

 while ($a_{new} == null$) {

 if ($T \cup \{a\}$ は閉路を持たない) { $a_{new} = a$ }

 else { $a = H.poll()$ }

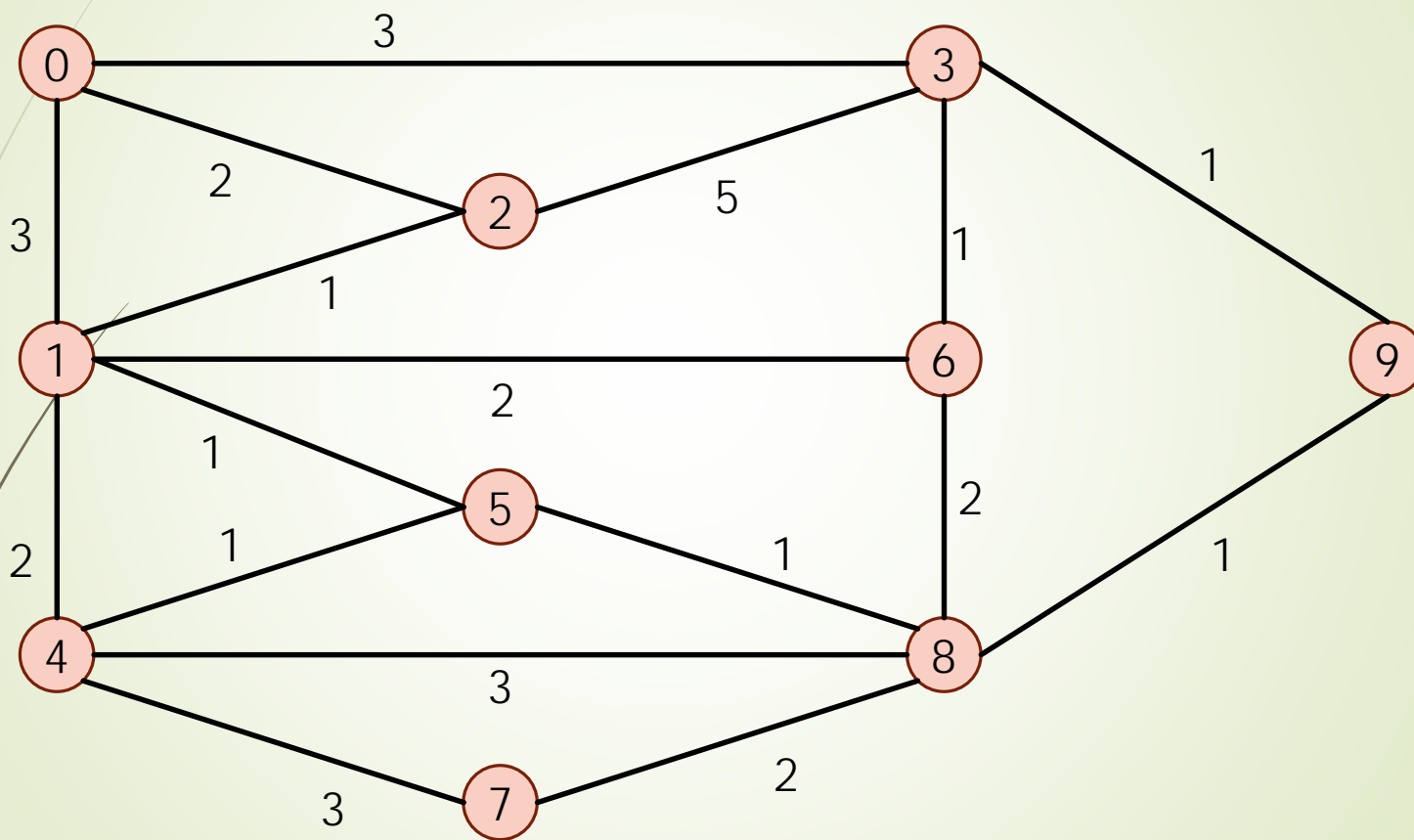
 }

$T = T \cup \{a_{new}\}$

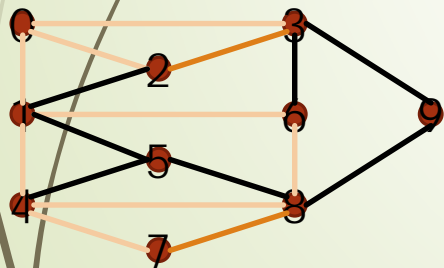
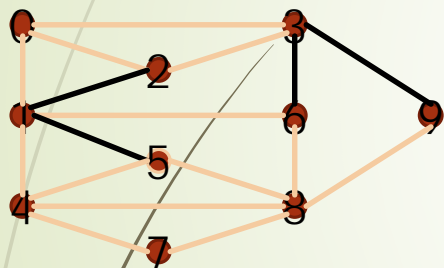
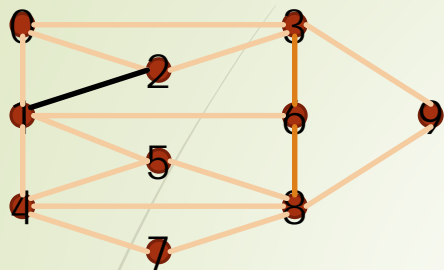
}

ヒープ使用は必須ではないが、ここでは最小重みの弧を探すためにヒープを利用

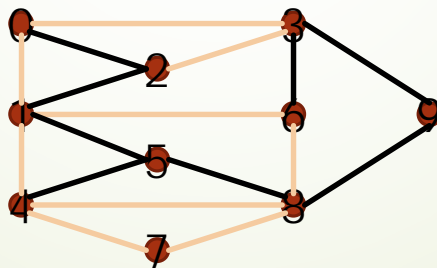
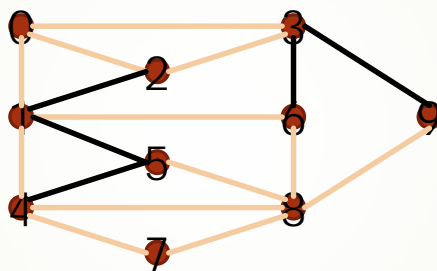
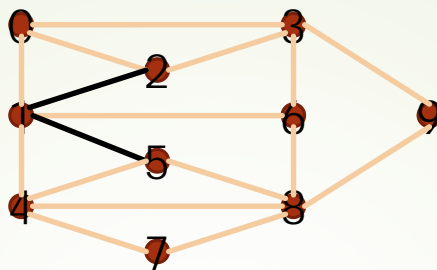
例1



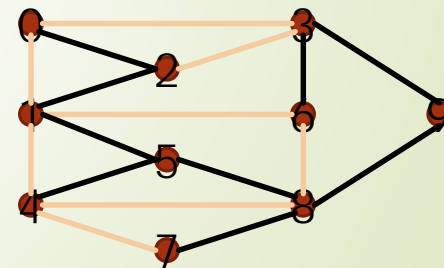
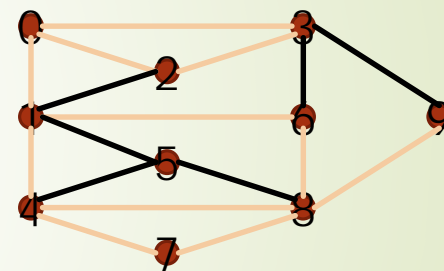
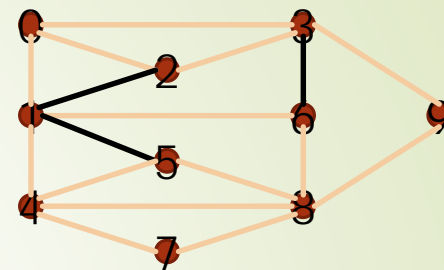
例1：解の探索の様子



ここまでは、重み1の弧



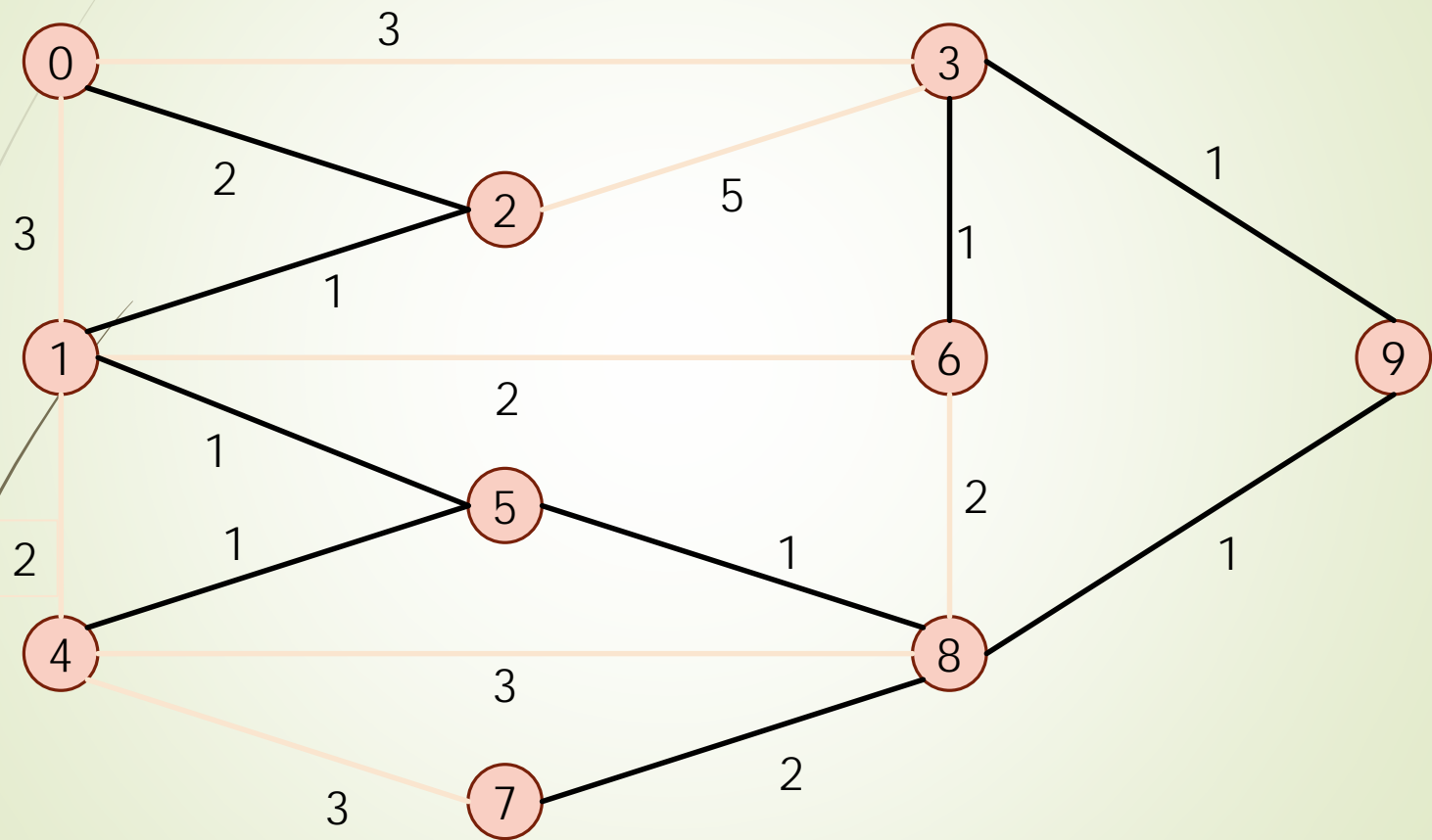
重み2の弧 $e(v_0, v_2)$



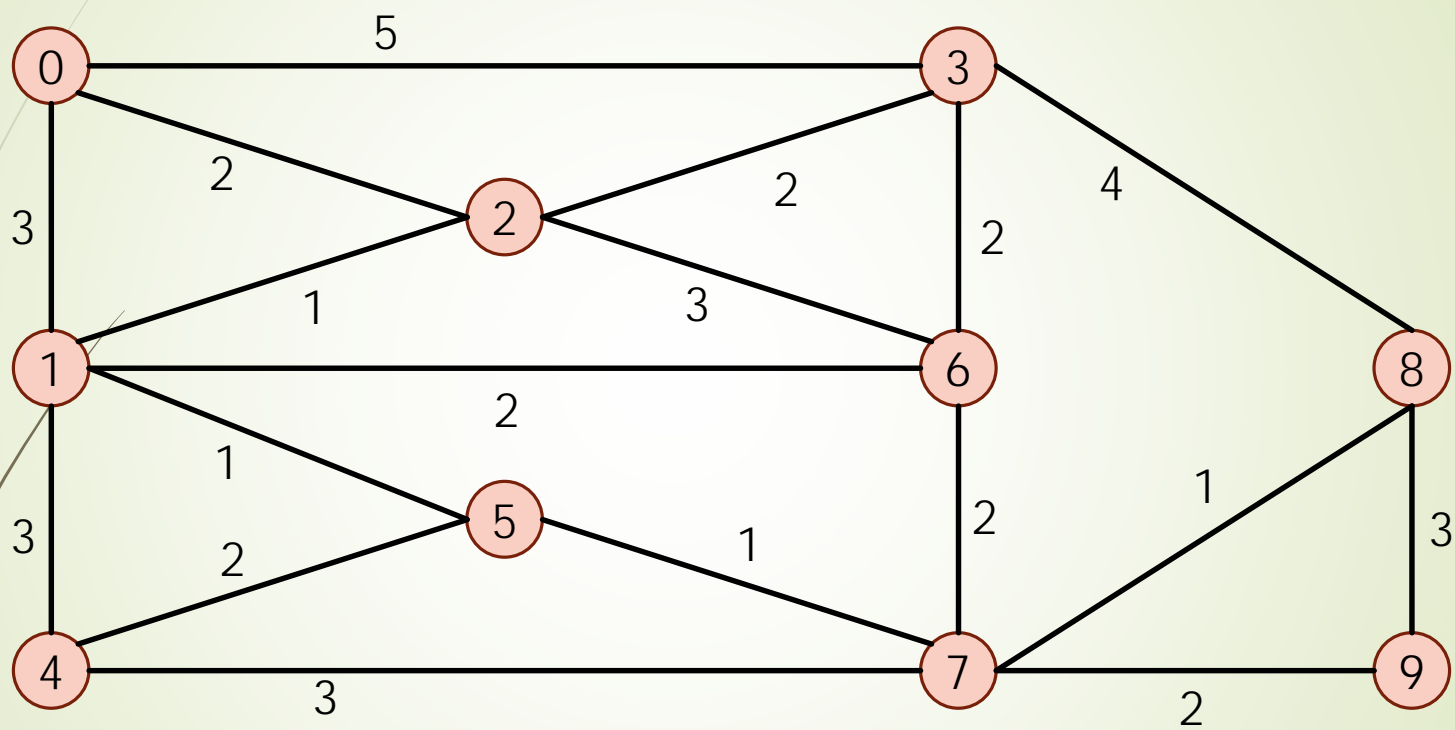
重み2の弧 $e(v_7, v_8)$

$e(v_1, v_6)$ を選択すると閉路ができる

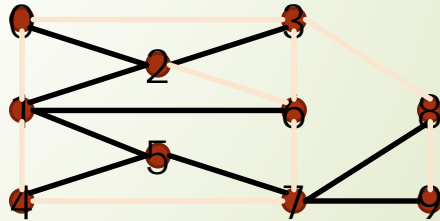
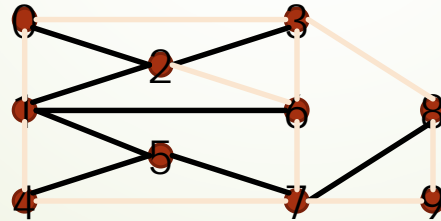
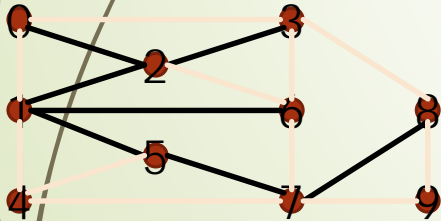
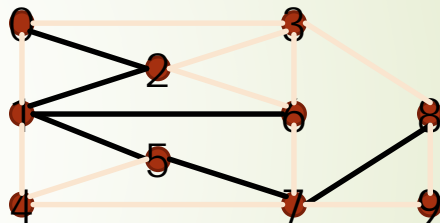
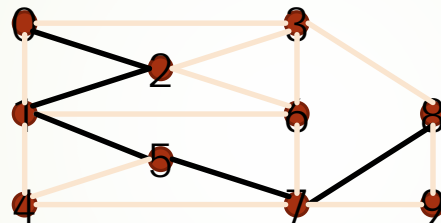
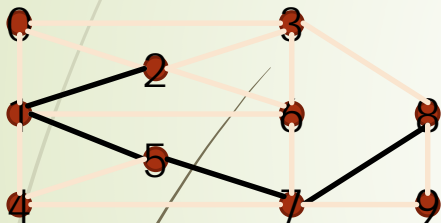
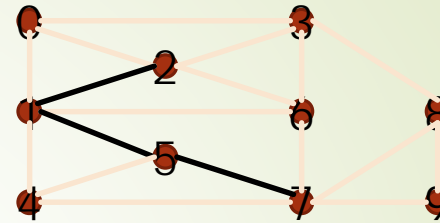
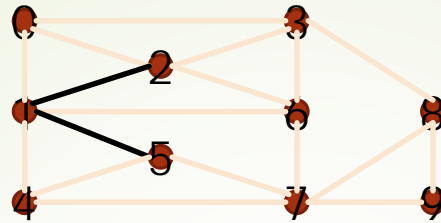
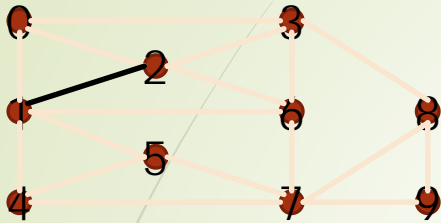
例1：結果



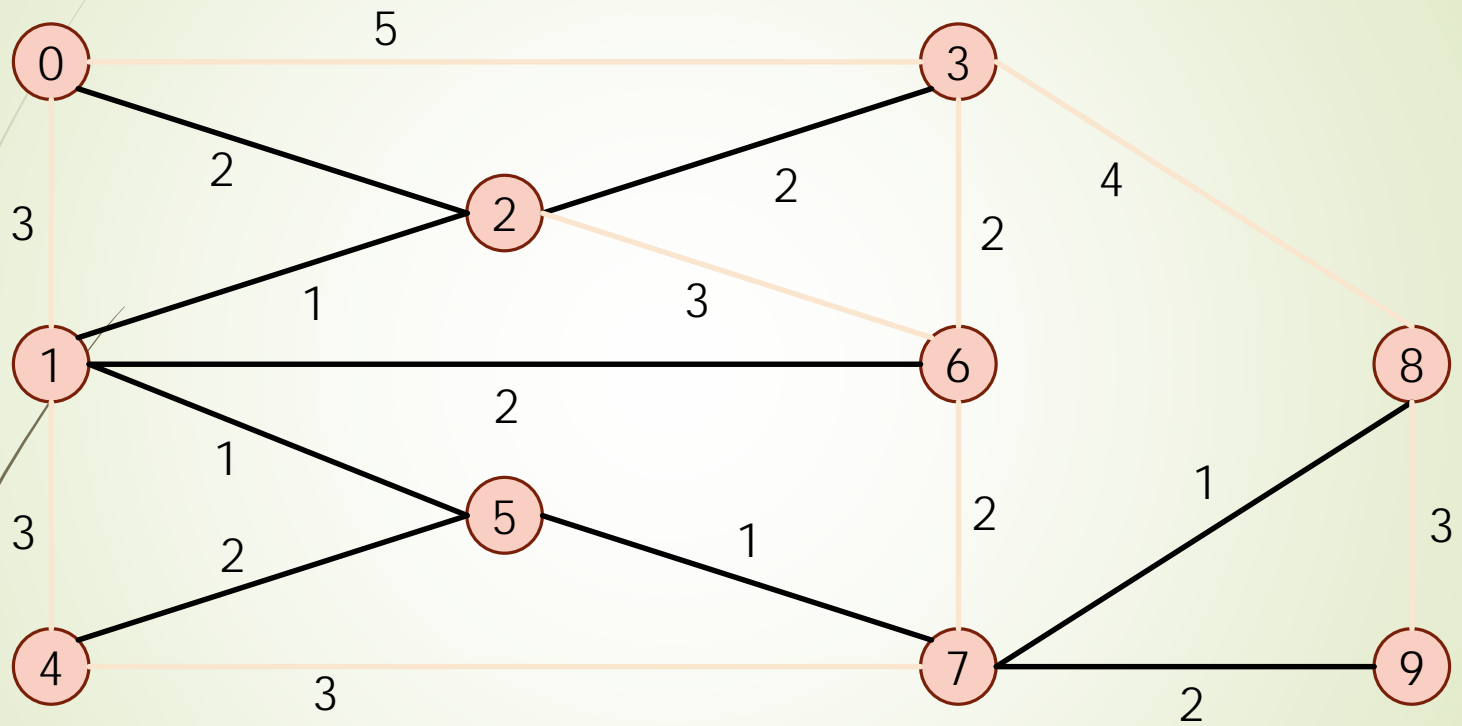
例2



例2：解の探索の様子



例2：結果



貪欲アルゴリズムが正しい理由

- 次の定理を証明すれば良い

「貪欲アルゴリズム実行中で得られる T は、弧数 $|T|$ を持ち、サーキットを含まない弧集合のうちで、その重みが最小である。」

- 要するに、アルゴリズムの各段階で最小の重みのグラフであることを示す。
- 証明では、上記を性質(*)と呼ぶことにする。

数学的帰納法による証明

- ▶ $T = \emptyset$ の時、自明
- ▶ 操作を i 回行って、次の弧を選択する直前の弧集合 T が条件(*)を満たしていると仮定する。
- ▶ 次に選択された弧を a とする： $a \in A \setminus T$
- ▶ $|T| + 1$ の弧を持ち、サーキットを有さない弧集合のうちで、重みの和が最小のものを S とする。
 - ▶ $W(S) = W(T) + w(a)$ ならば性質(*)が成り立つ。
 - ▶ $W(S) < W(T) + w(a)$ は矛盾する（起こりえないことを示す）。

証明：準備

- ▶ T は $|T|$ 本の弧を持つサーキット(閉路)の無い弧集合の中で、重みが最小である。
 - ▶ 従って、 S から任意の弧 b を取り除いた弧集合 ($|T|$ 本の弧) の重みは T の重みより小さいことはない。
 - ▶ $\forall b \in S, w(S \setminus \{b\}) = w(S) - w(b) \geq w(T)$

証明：準備

- このことから、 S に含まれる任意の弧 b と T にこれから追加する弧 a の重みの大小関係がわかる。

$$w(S) < w(T) + w(a) \leq w(S) - w(b) + w(a)$$

↓

$$w(b) < w(a)$$

証明：矛盾の導出

- ▶ S と T はサーキットを含まず $|S| = |T| + 1$
- ▶ ある $a' \in S \setminus T$ に対して $T \cup \{a'\}$ はサーキットを含まない。
 - ▶ a' は S の弧であることから、 $w(a') < w(a)$
 - ▶ なぜなら、 T は(*)を満たすから
 - ▶ これより $w(T \cup \{a'\}) = w(T) + w(a') < w(T) + w(a)$
 - ▶ これは a の選び方に反する。
- ▶ よって矛盾する。
 - ▶ つまり、そのような S は存在せず、手続きに従って構成した $T \cup \{a\}$ は、最小木である。

補足と注意

- ▶ $T \cup \{a'\}$ がサーキットを含まない $a' \in S \setminus T$ が必ずあること。
 - ▶ S が T に含まれない頂点を含む場合には自明
 - ▶ T が木であれば、 S は T に含まれない頂点を含む
 - ▶ S と T が同じ頂点を結ぶ場合には、 T は部分木の集合であり、 S は T の非連結要素を結ぶ弧を有する

補足と注意

- ある弧を選択した際に、それが閉路を作らないことの確認が必要
 - 加えようとして弧 a の両端の頂点 (v, w)
 - T 内に v から w への道があるかを調べる
 - 深さ優先、幅優先の探索アルゴリズムが必要

最大補木を求める方法

- ➡ 最小木を求めるために、重み最大の補木 $A \setminus T$ を求める

```
T ← A
H // 弧のヒープ。ただし、最大要素が先頭
while ( T は G の極大木ではない ) {
    a = H.poll // ヒープ内の最大要素
    if ( T \ {a} は G の極大木を含む ) {
        T ← T \ {a}
    }
}
```