



最小木問題

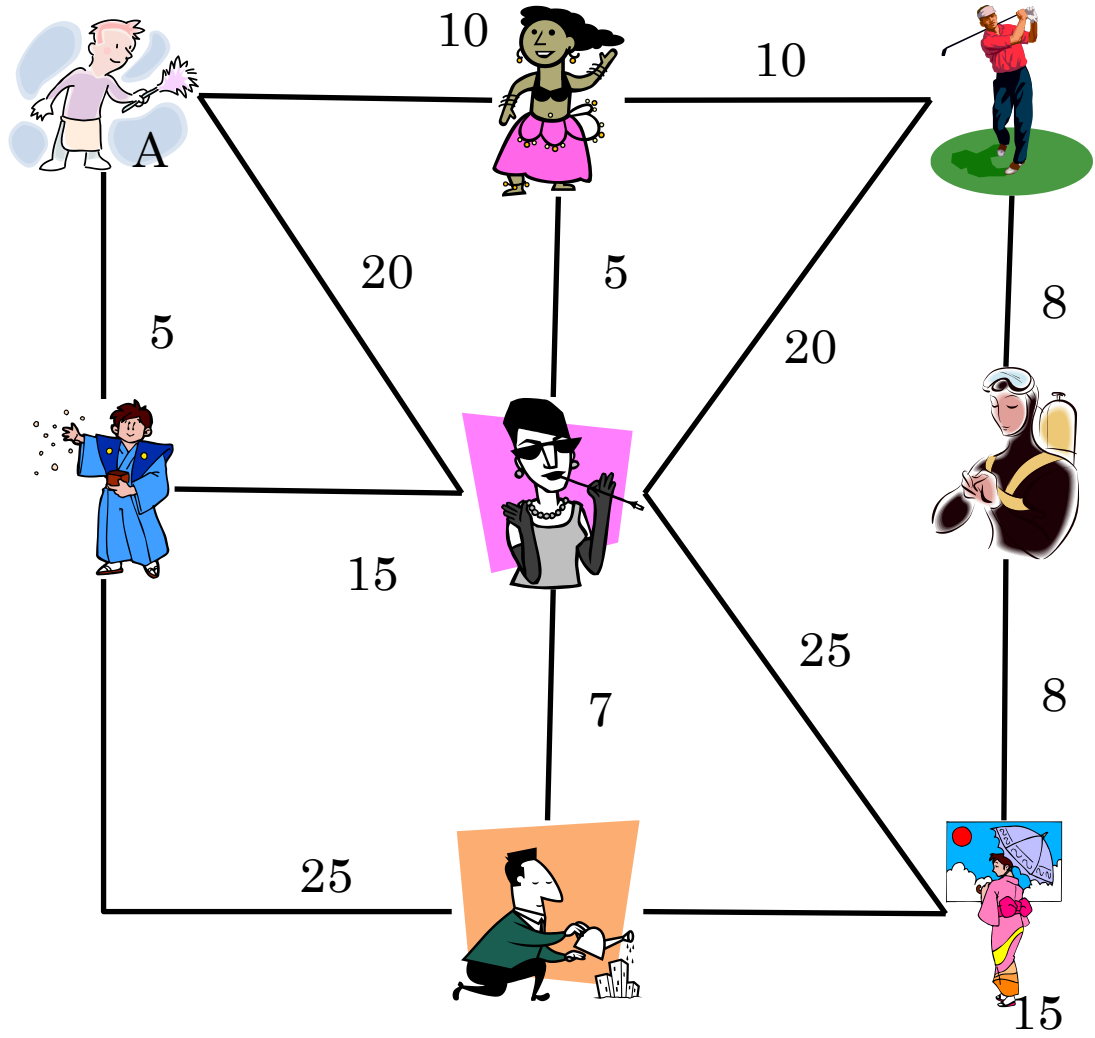
MINIMUM TREE PROBLEM

# 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

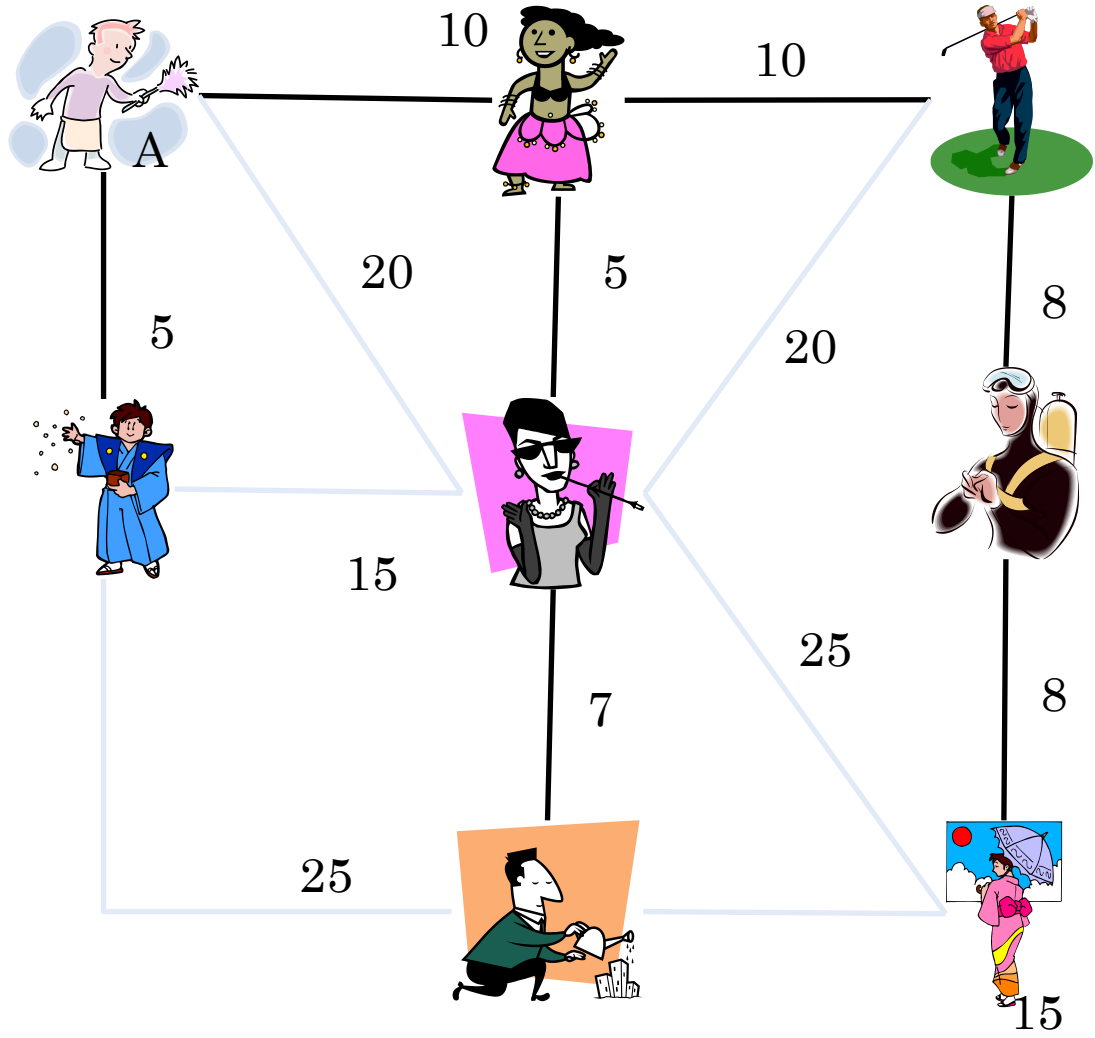


Aから全員に最も安く連絡する経路  
総経費を考える

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  $G=(V,A)$ 
  - 弧に向きが無い
- 重み関数  $w: A \rightarrow R$ 
  - 各弧に実数に対応: 重み、距離、etc.
- $G$  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



## 二つのアプローチ

### ○ 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

### ○ Jarník-Prim法

- 始点から開始して、連結した頂点の数を増やす
- 構成途中でも木になっている



# 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)

$T = \emptyset$

$H:G$  の弧の重みに関するヒープ

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

$a_{\text{new}} = \text{null}$

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

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

$a_{\text{new}} = a$

        }else{

$a = H.poll()$

        }

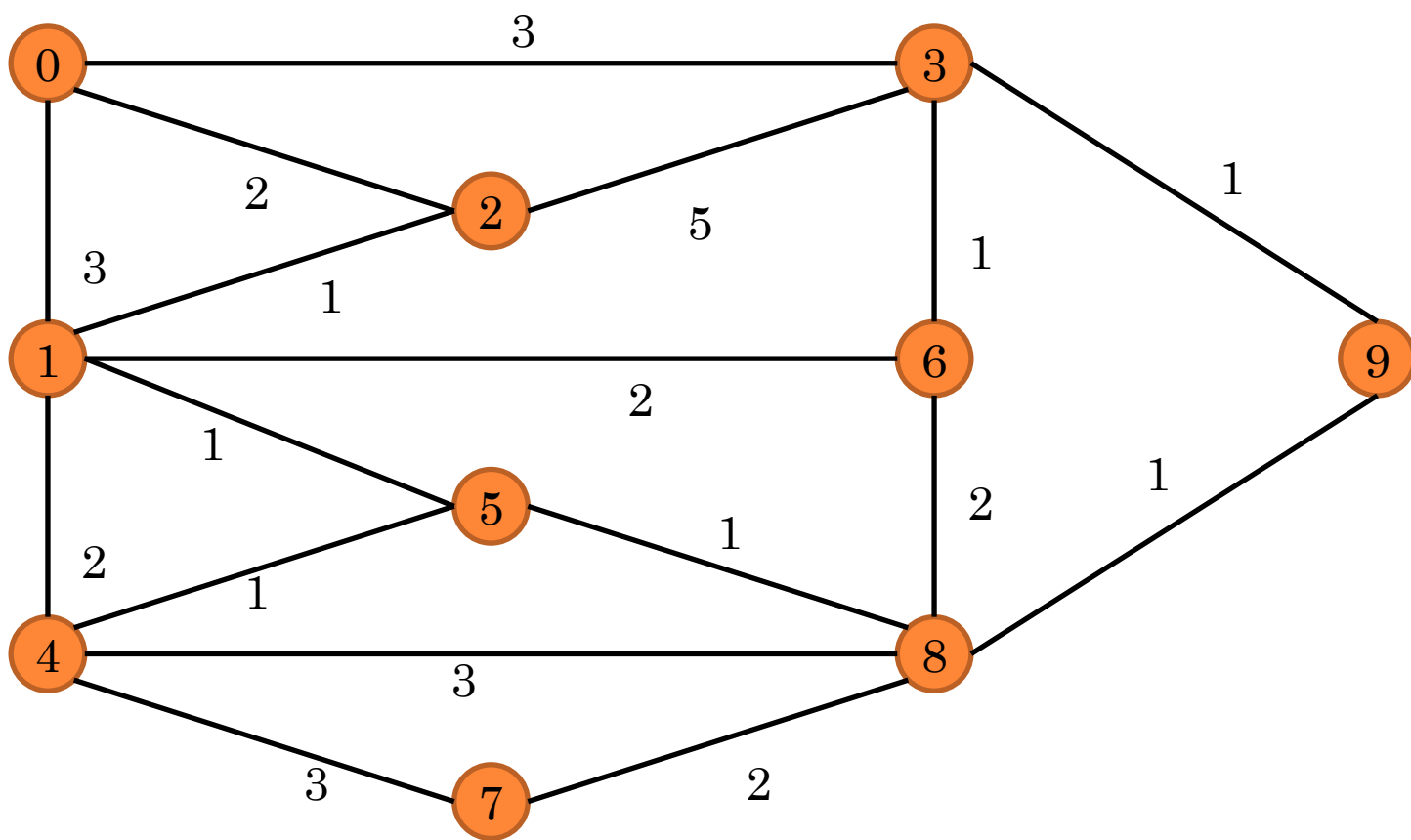
    }

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

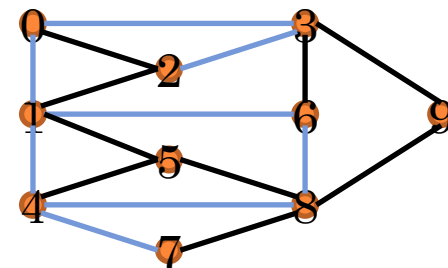
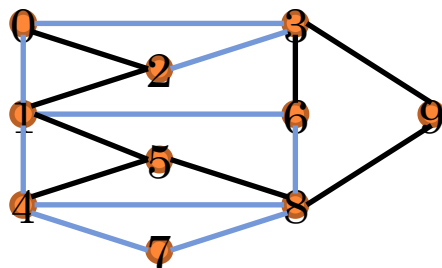
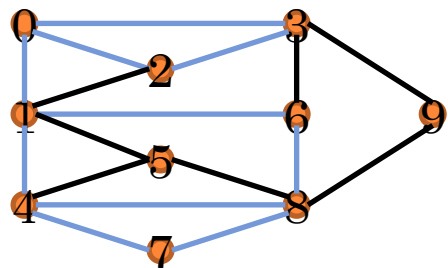
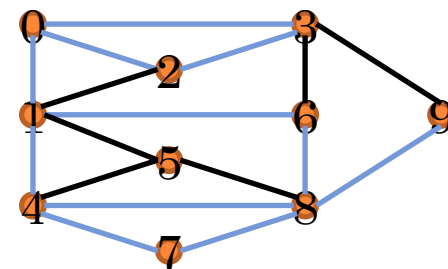
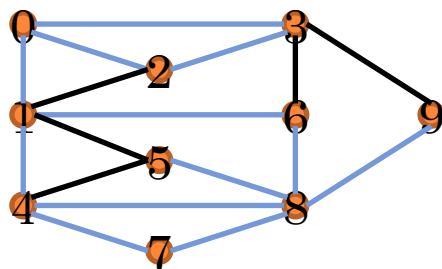
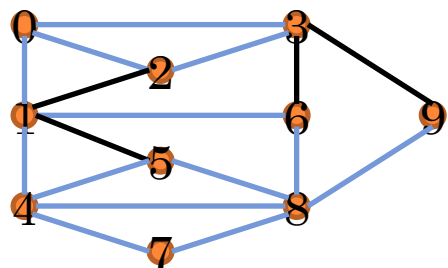
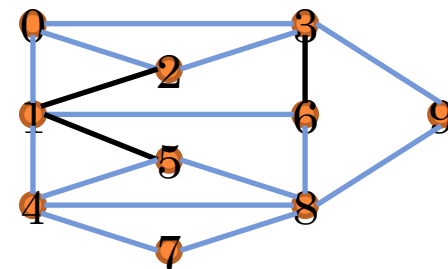
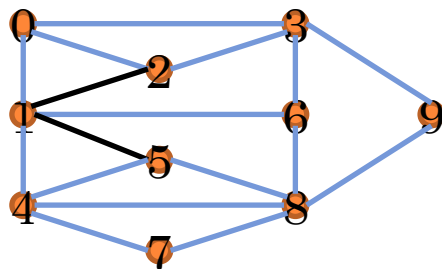
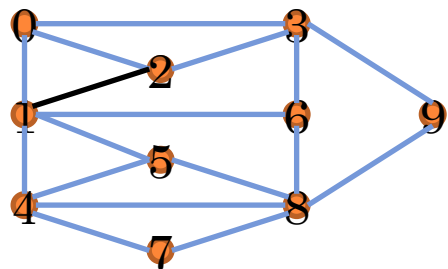
}



# 例1



# 例1: 解の探索の様子



ここまでは、重み1の弧

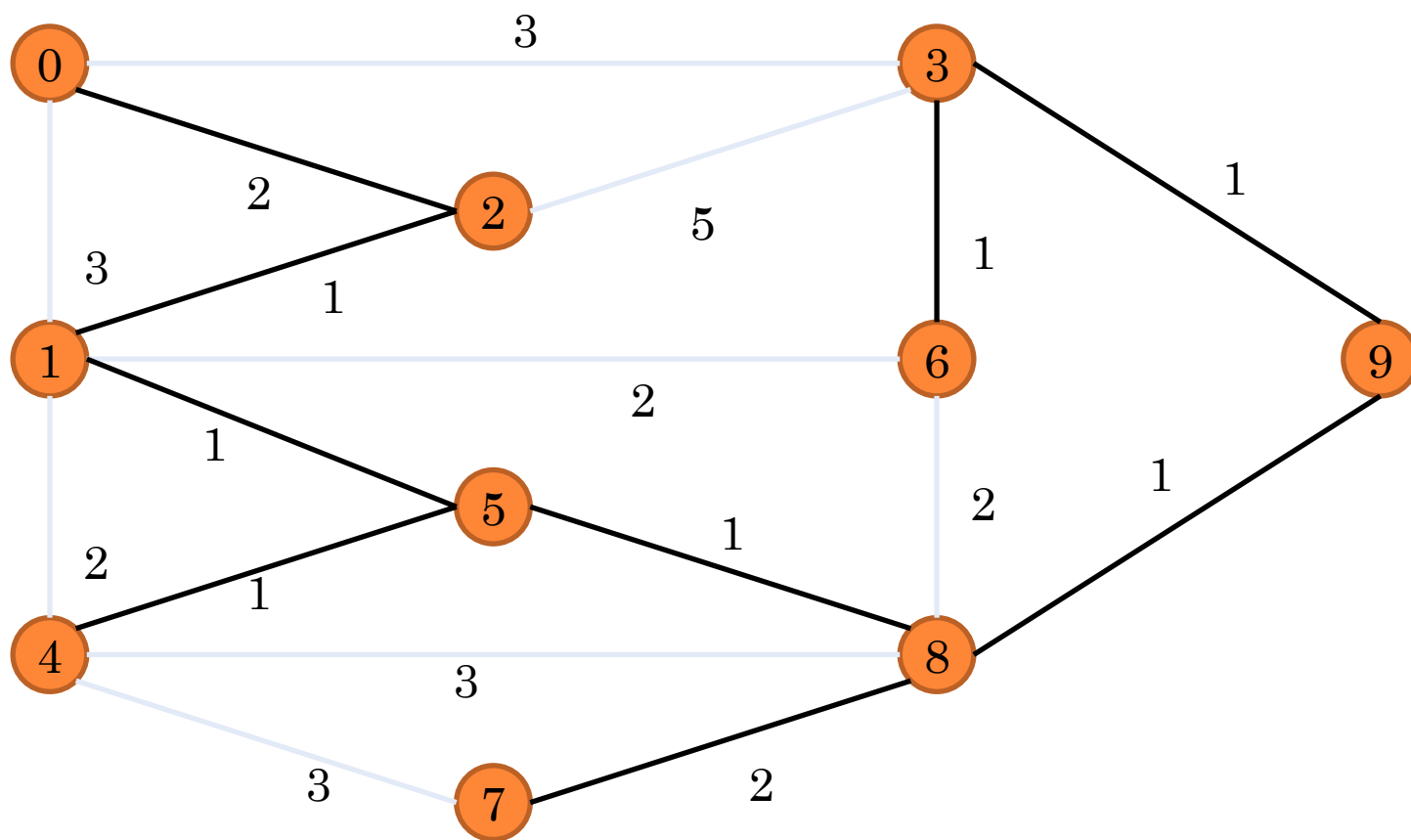
重み2の弧  $e(v_0, v_2)$

重み2の弧  $e(v_7, v_8)$

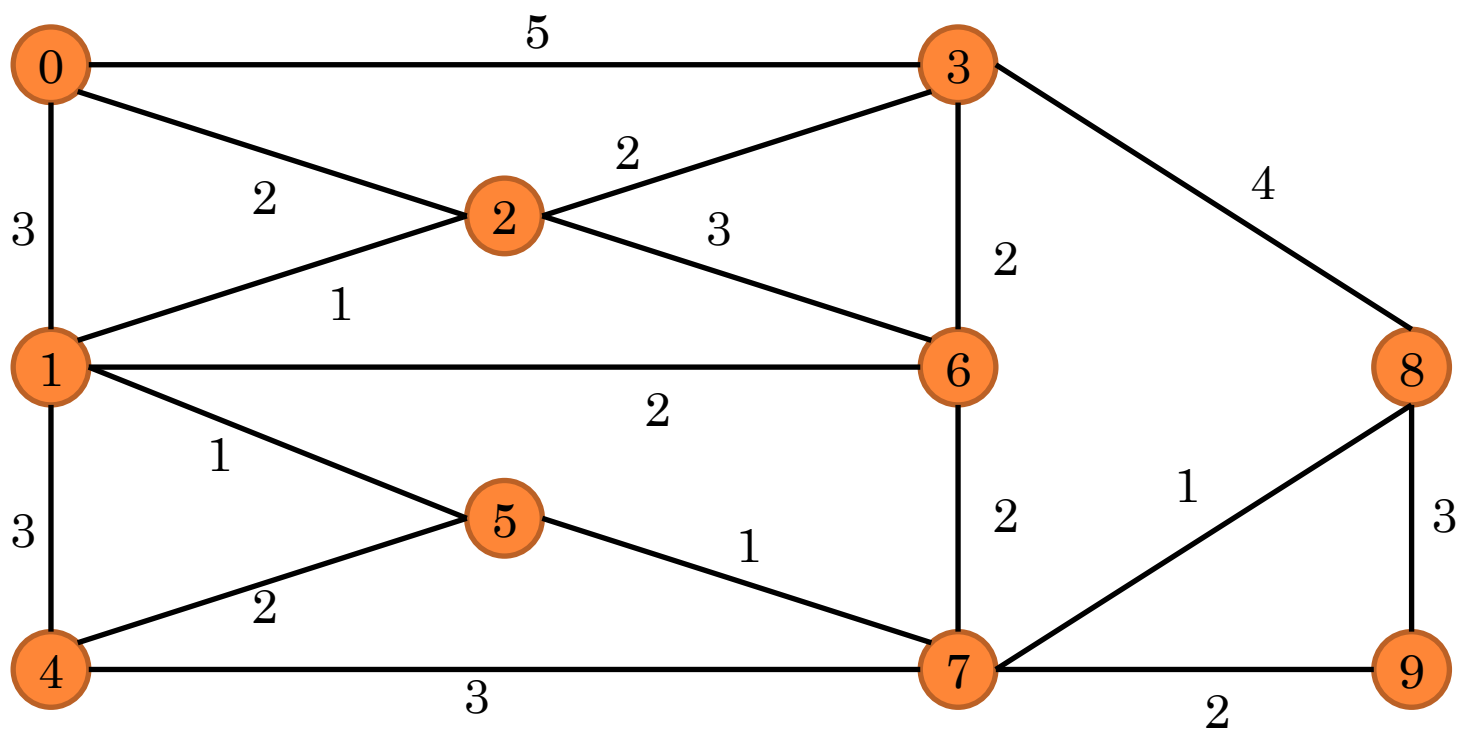


$e(v_1, v_6)$  を選択すると閉路ができる

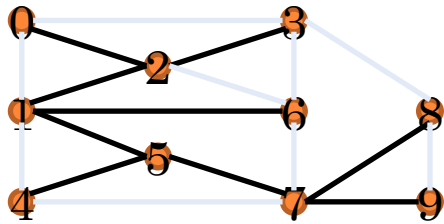
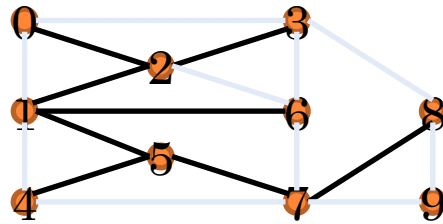
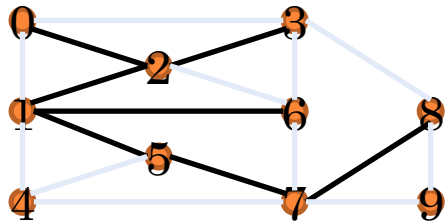
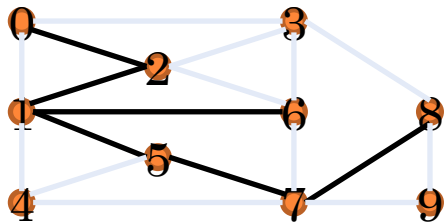
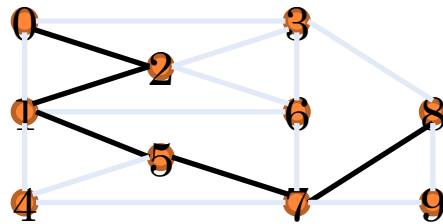
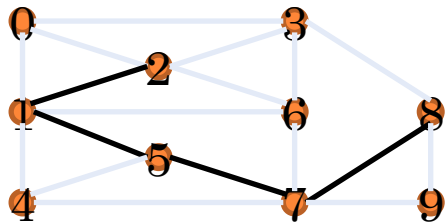
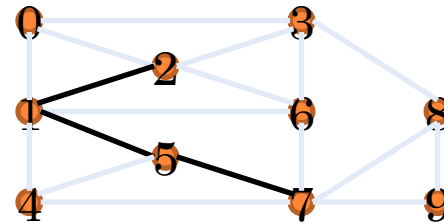
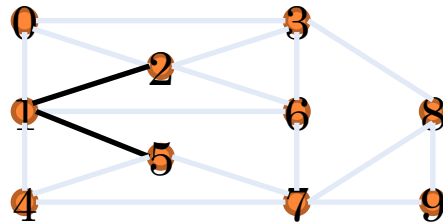
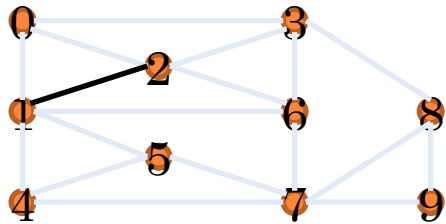
# 例1:結果

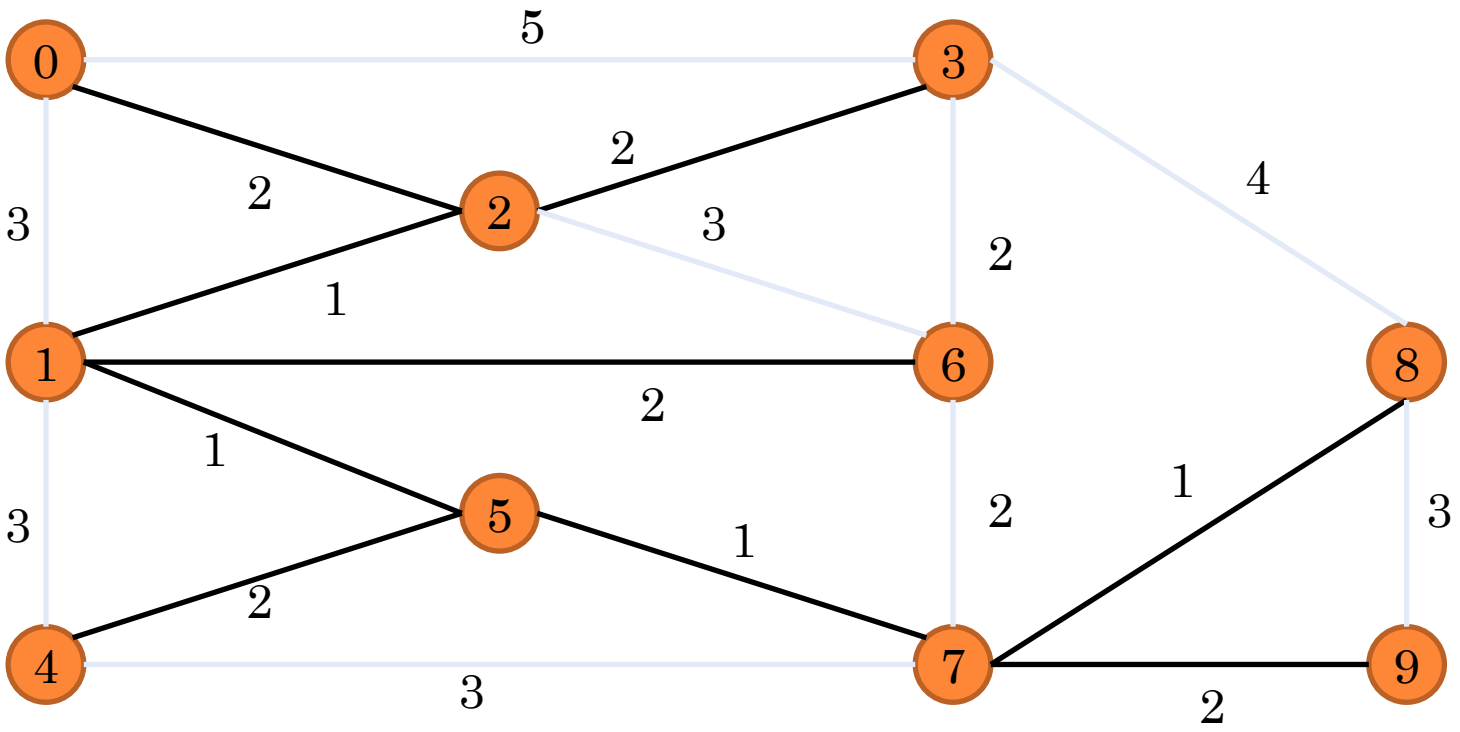


## 例2



## 例2: 解の探索の様子





# 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枝集合のうちで、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にある弧集合 $T$ が性質(\*)を満たしているとする。
  - 次に選択された弧を $a$ とする。
- $|T|+1$ 本の弧を持ちサーキットを有さない弧集合のうちで重みが最小のものを $S$ 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)



## 証明: 準備

- $T$  は  $|T|$  本の弧を持つサーキット(閉路)の無い弧集合で、重みが最小である。
  - $S$  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。

$$\forall b \in S, w(S \setminus \{b\}) = w(S) - w(b) \geq w(T)$$

- 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、  $w(a') < w(a)$
  - 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。つまり、 $a$  を加えるより小さい木が存在する。
- よって矛盾する



## 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

```
 $T \leftarrow A$   
 $w_{\max} = 0$   
While ( $T$  は  $G$  の極大木ではない ) {  
    forall ( $a \in T$  ) {  
        if ( $T \setminus \{a\}$  は  $G$  の極大木を含む) {  
            if ( $w(a) > w_{\max}$  ) {  
                 $a_{\text{selected}} = a$   
                 $w_{\max} = w(a)$   
            }  
        }  
    }  
    }  
    }  
 $T \leftarrow T \setminus \{a_{\text{selected}}\}$   
}
```



# JARNÍK-PRIM法

- 始点から開始して、連結した頂点の数を増やす
- 構成途中でも木になっている
- 途中の木から、未連結の頂点への弧のうちから、最小の弧を選んで、枝を伸ばす



# JARNÍK-PRIMアルゴリズム

任意の頂点  $v \in V$  を選び、 $U = \{v\}$ 、 $T = \emptyset$  とする

While ( $U \neq V$ ) {

$U$  と  $V \setminus U$  を結ぶ弧のうち最小の重みのものを  $a$  とする

$a$  の  $V \setminus U$  側の端点を  $w$  とする

$U \leftarrow U \cup \{w\}$

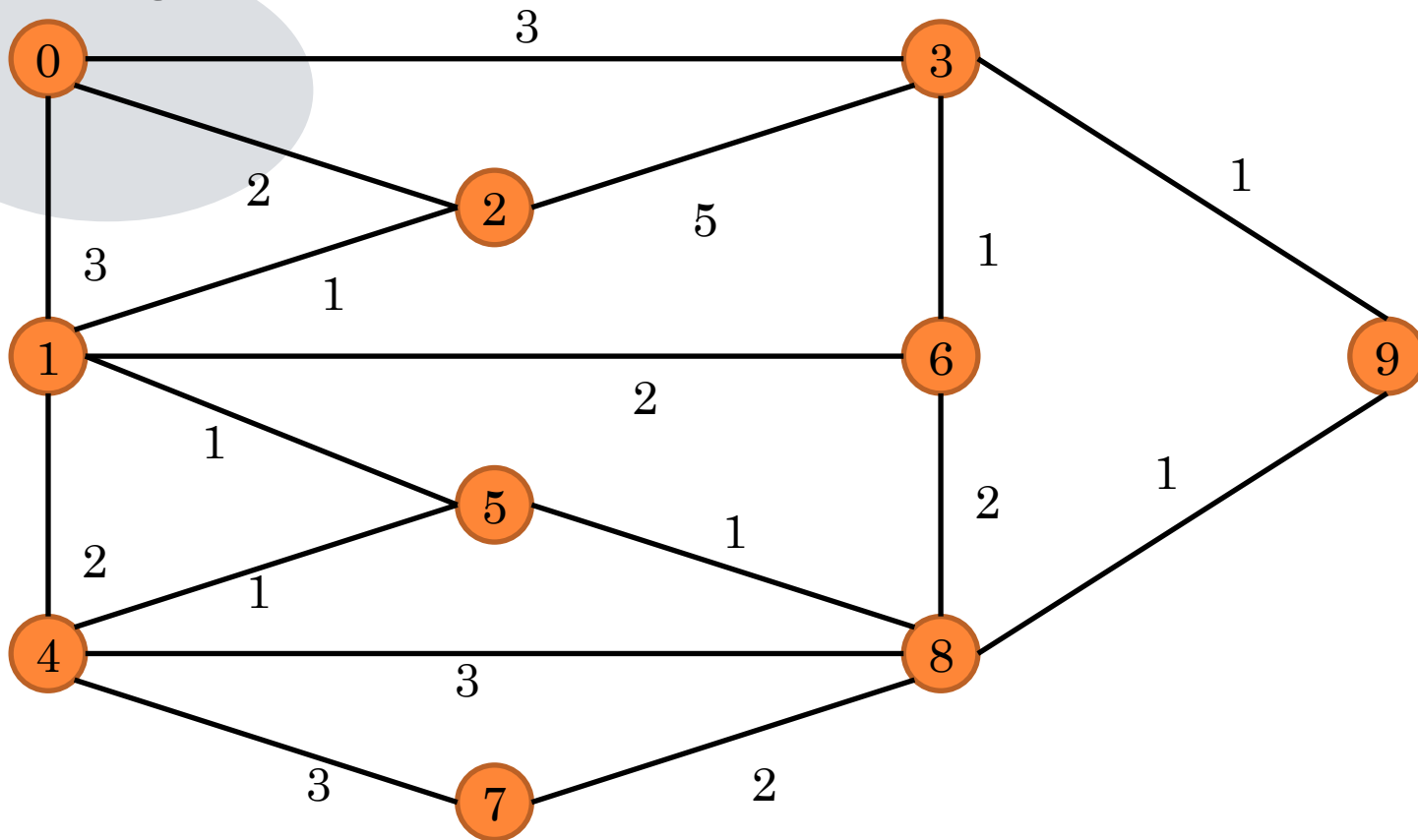
$T \leftarrow T \cup \{a\}$

}

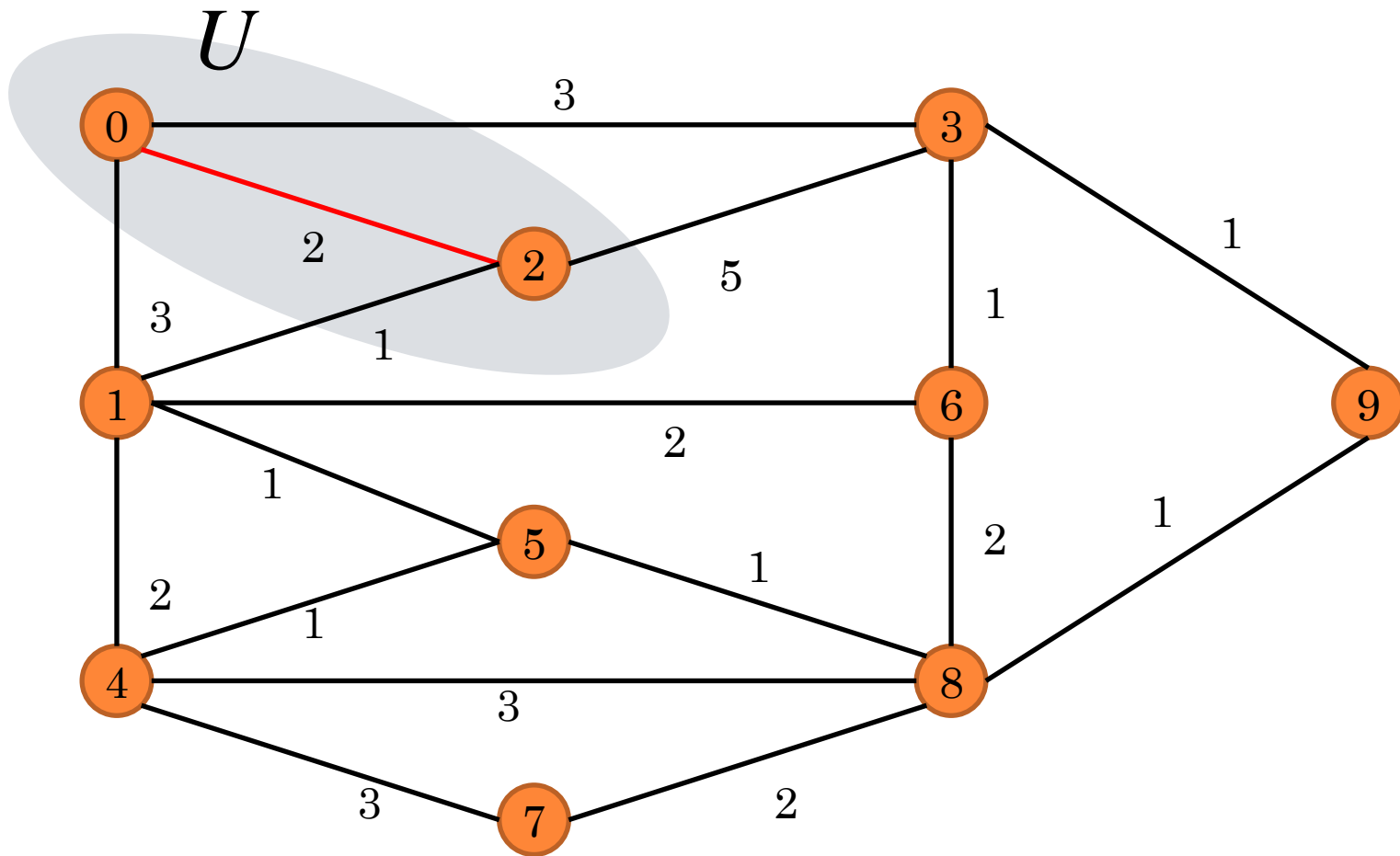


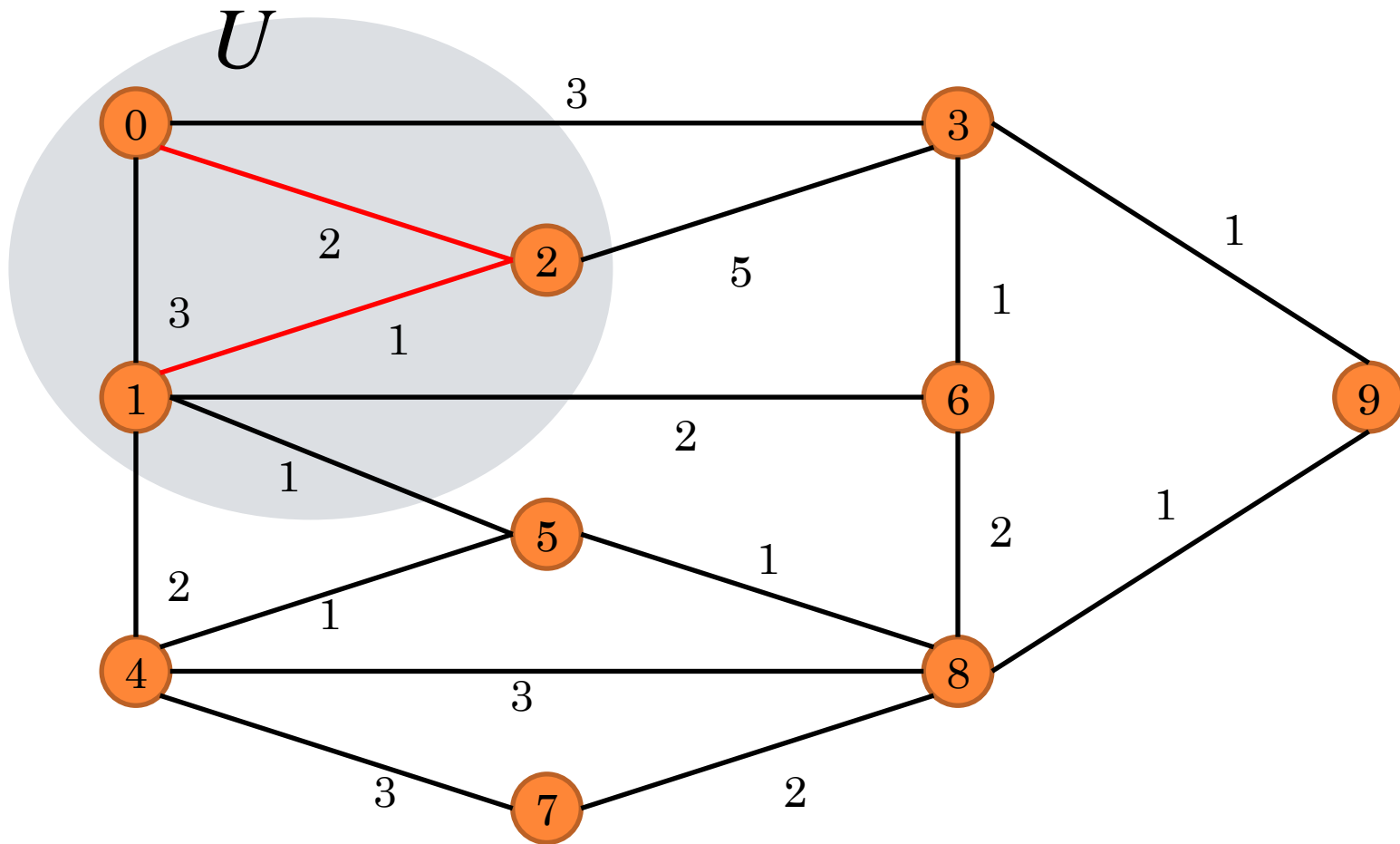
例1

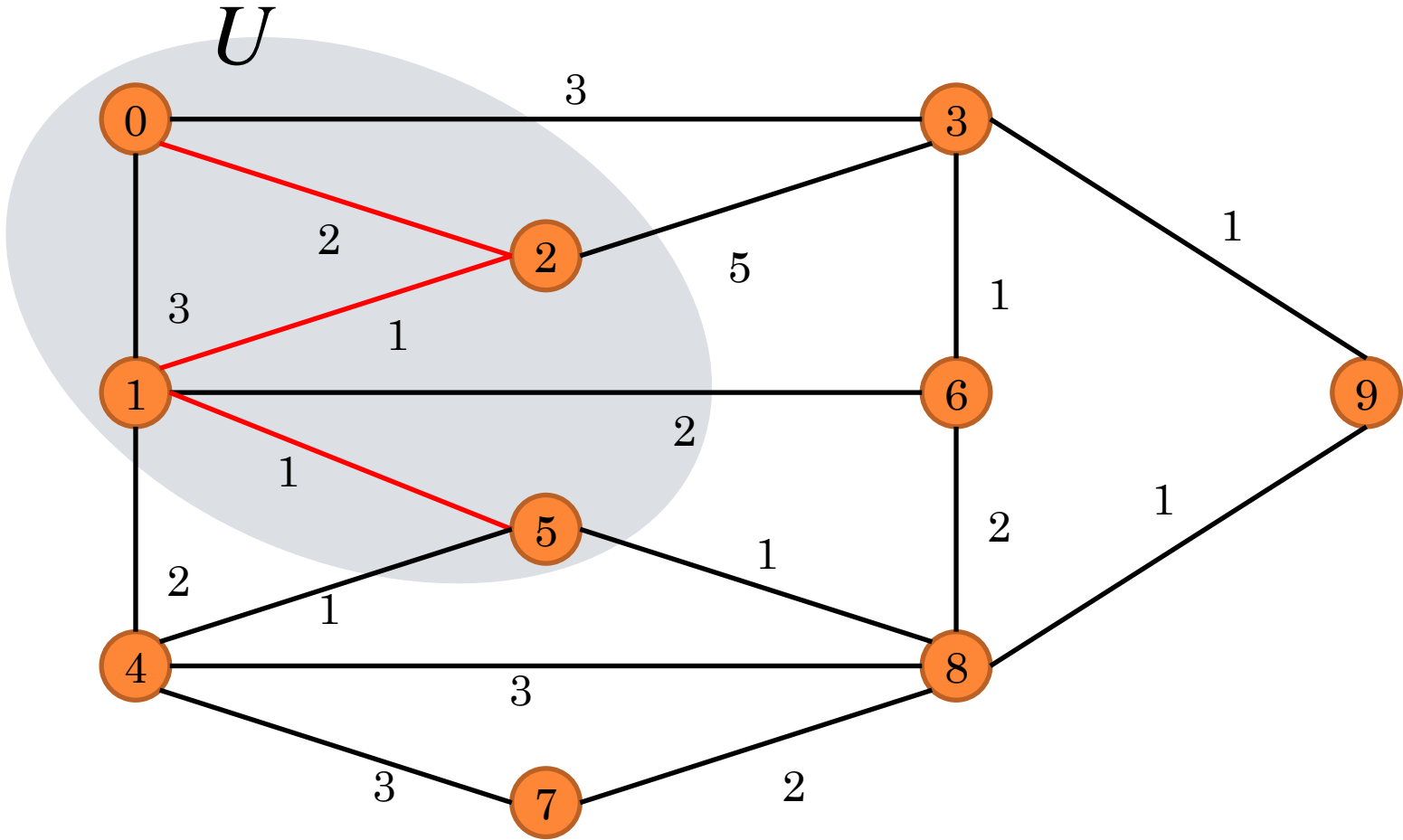
$U$

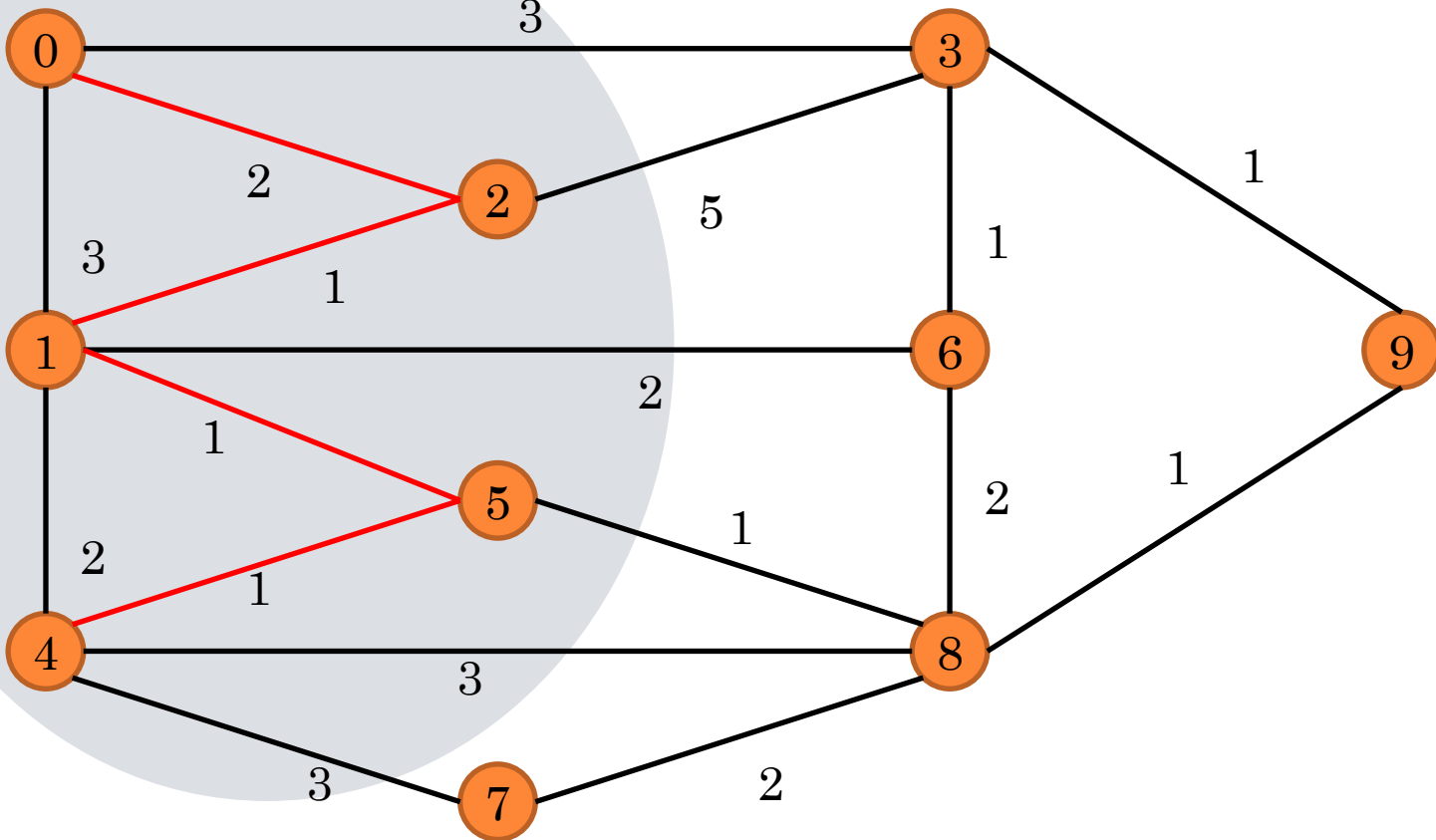
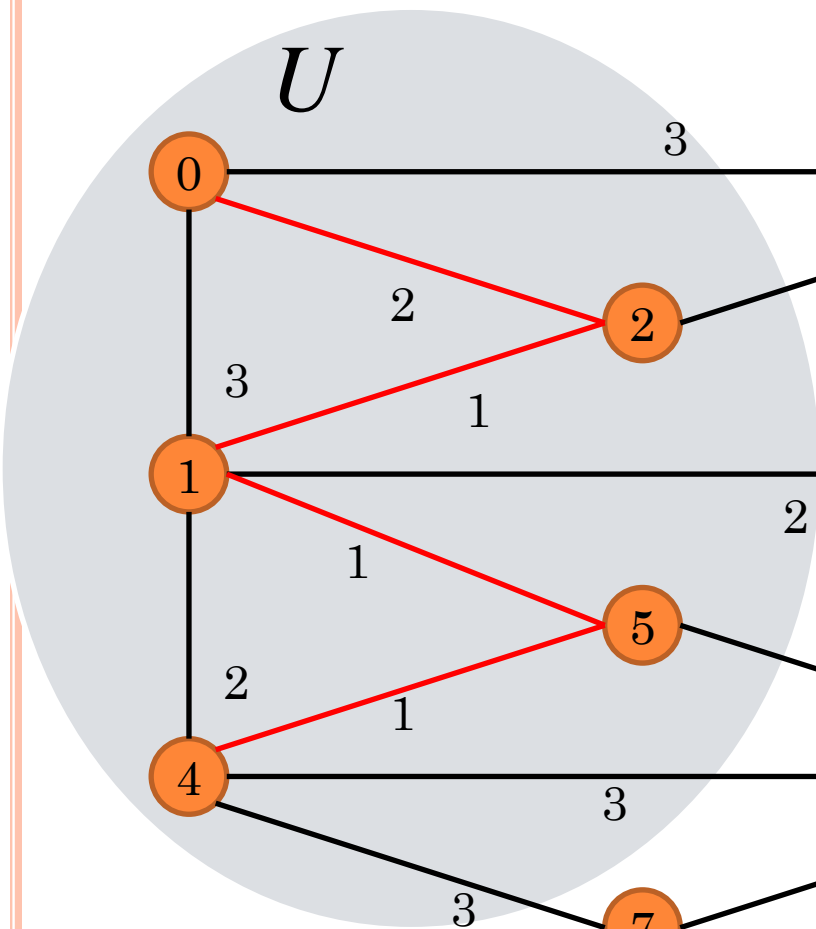


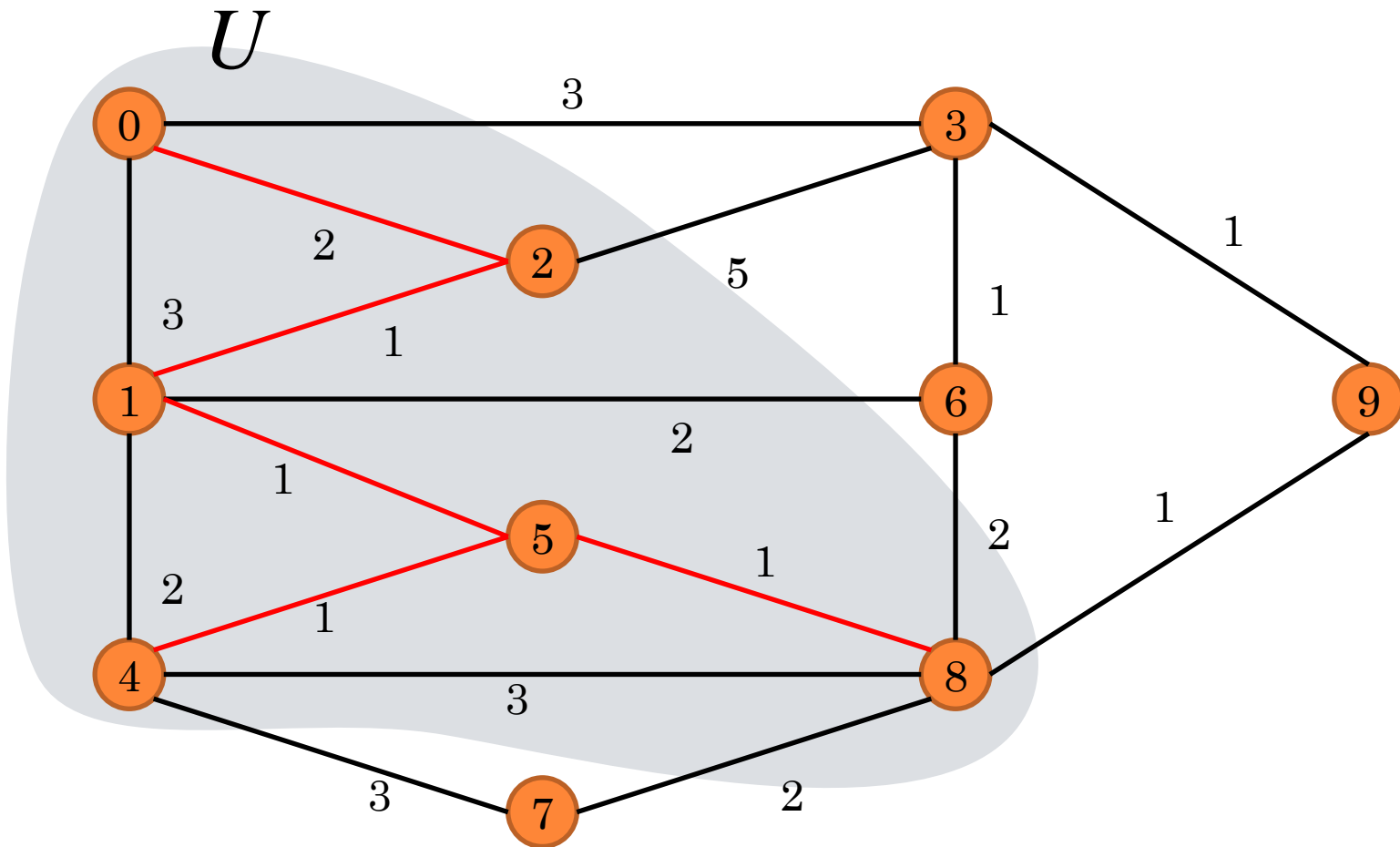


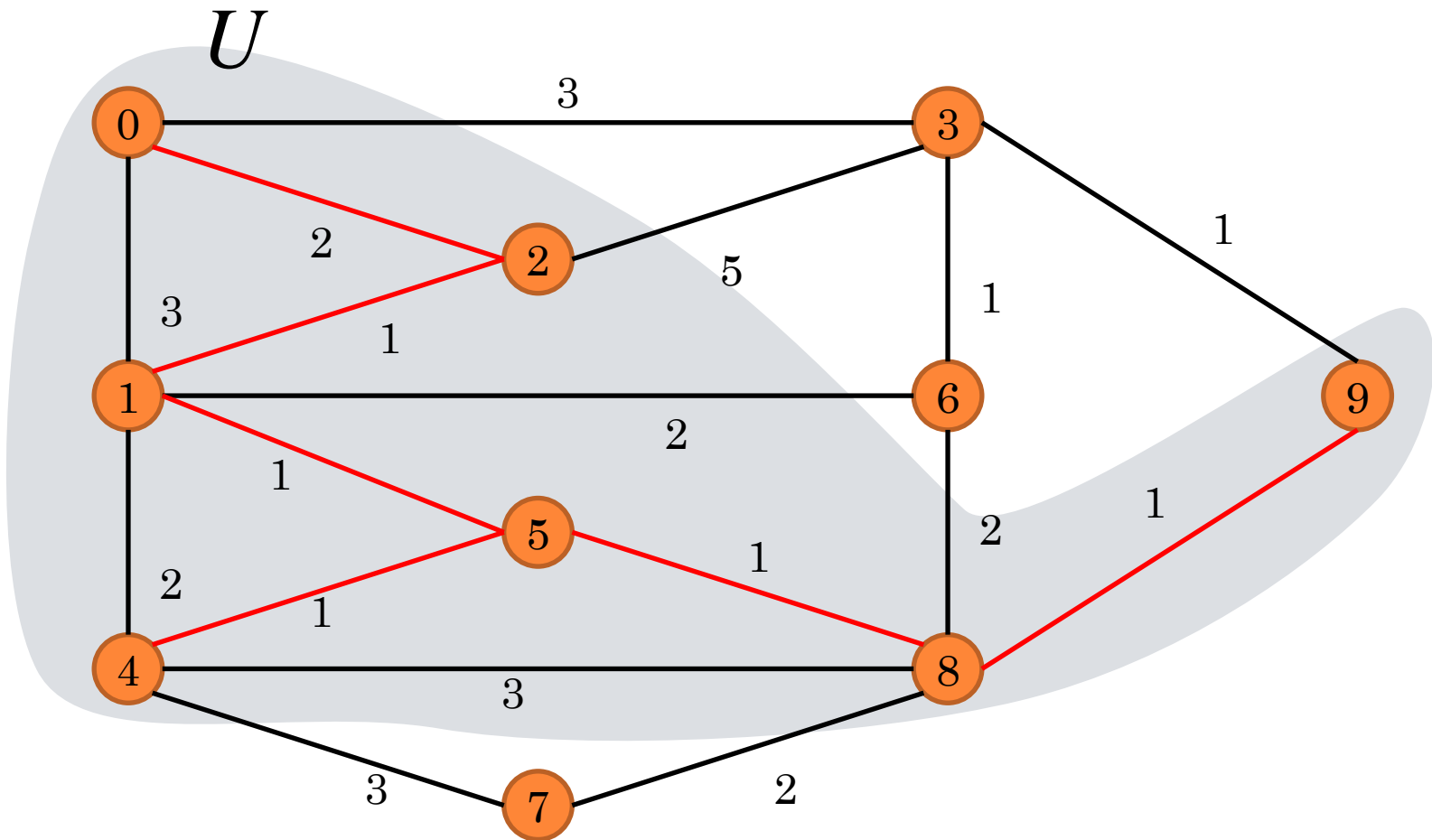


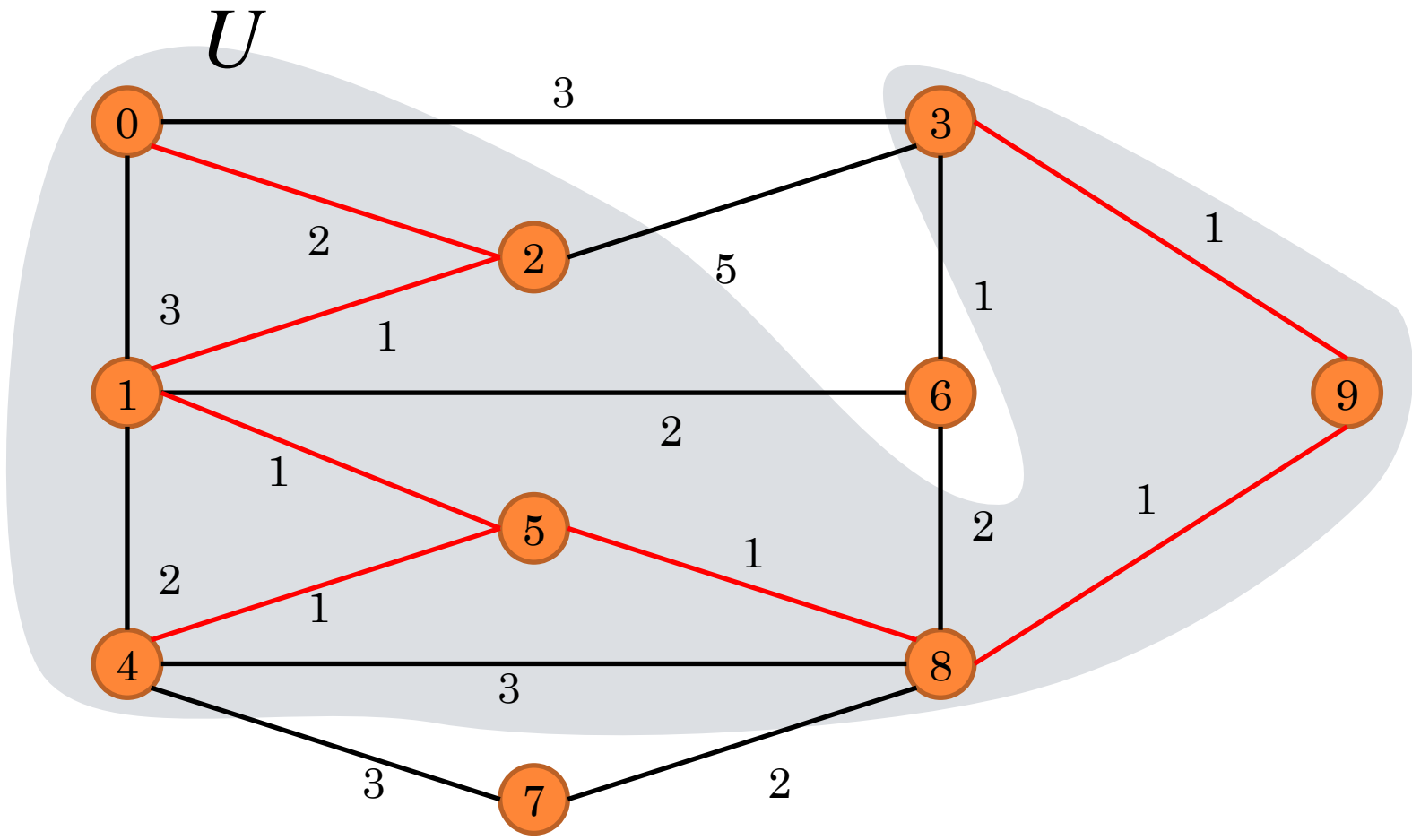


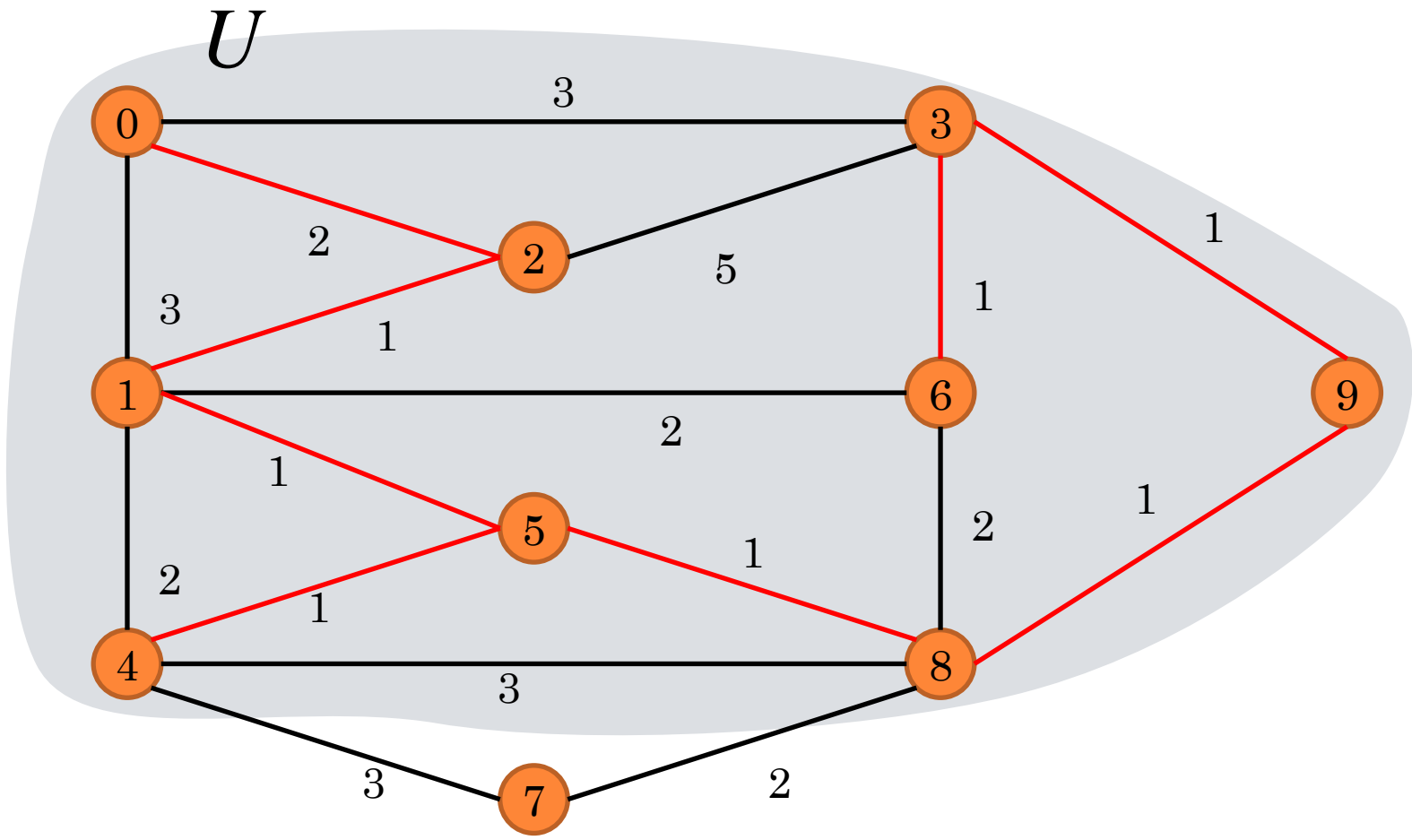






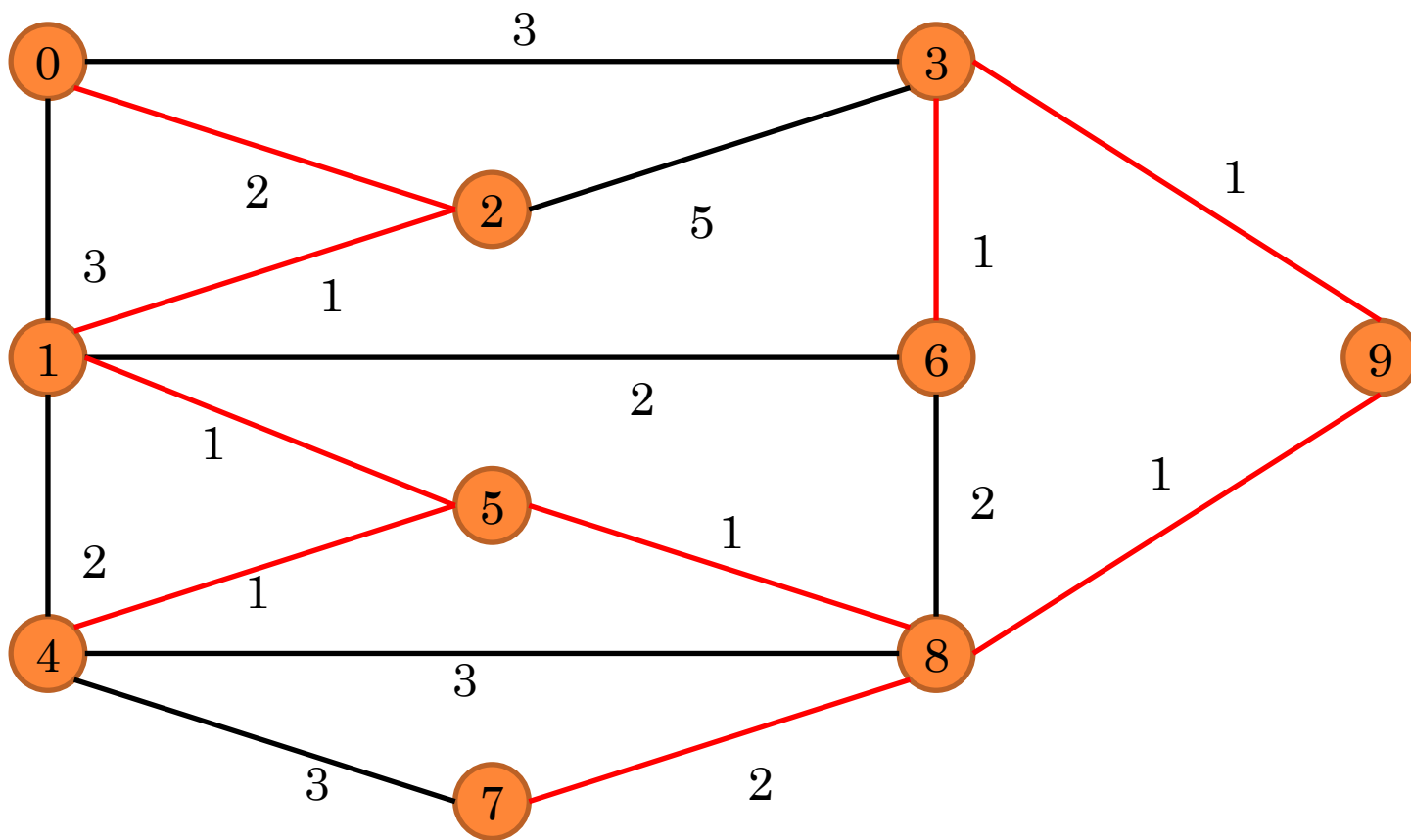




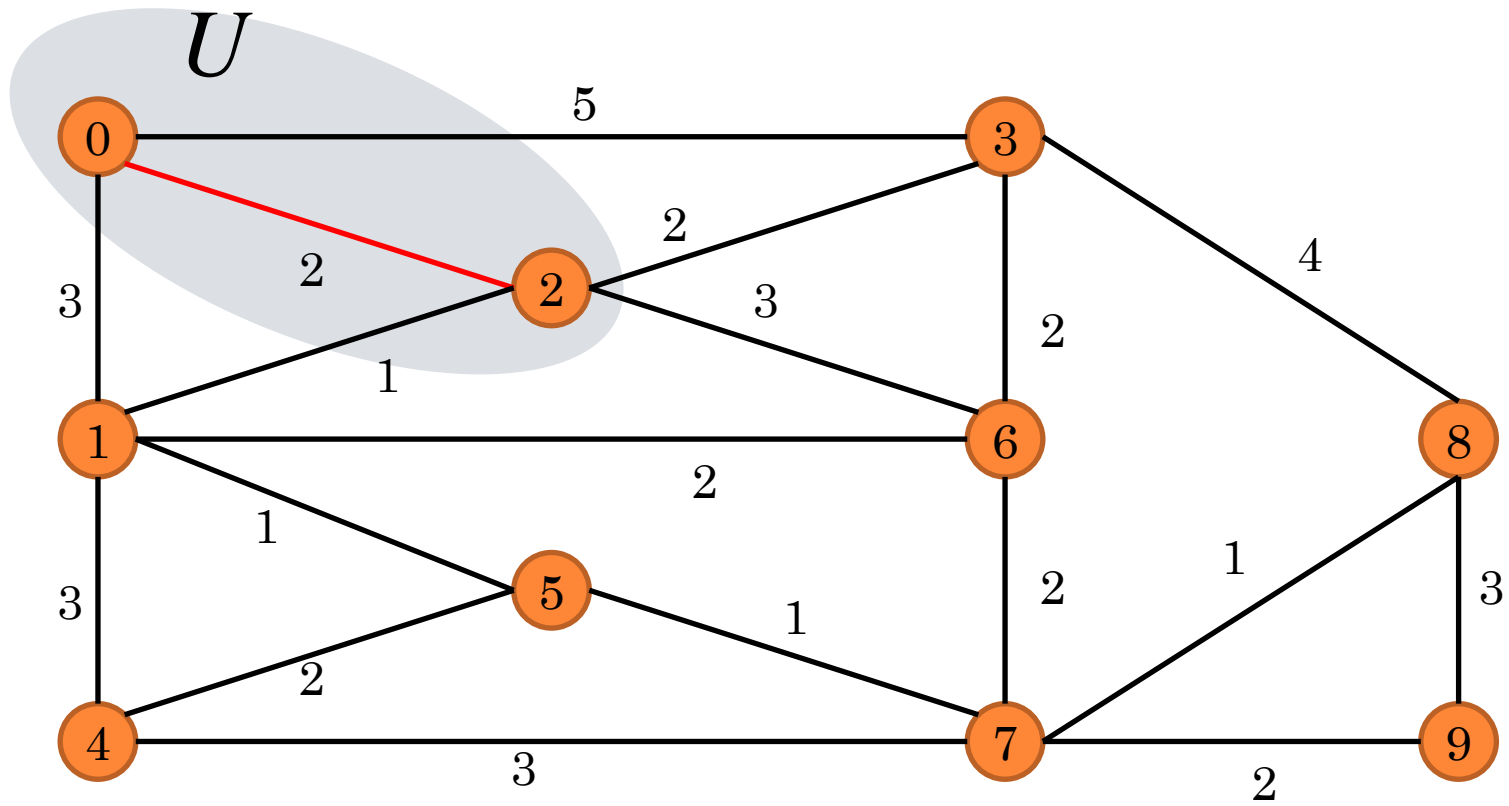




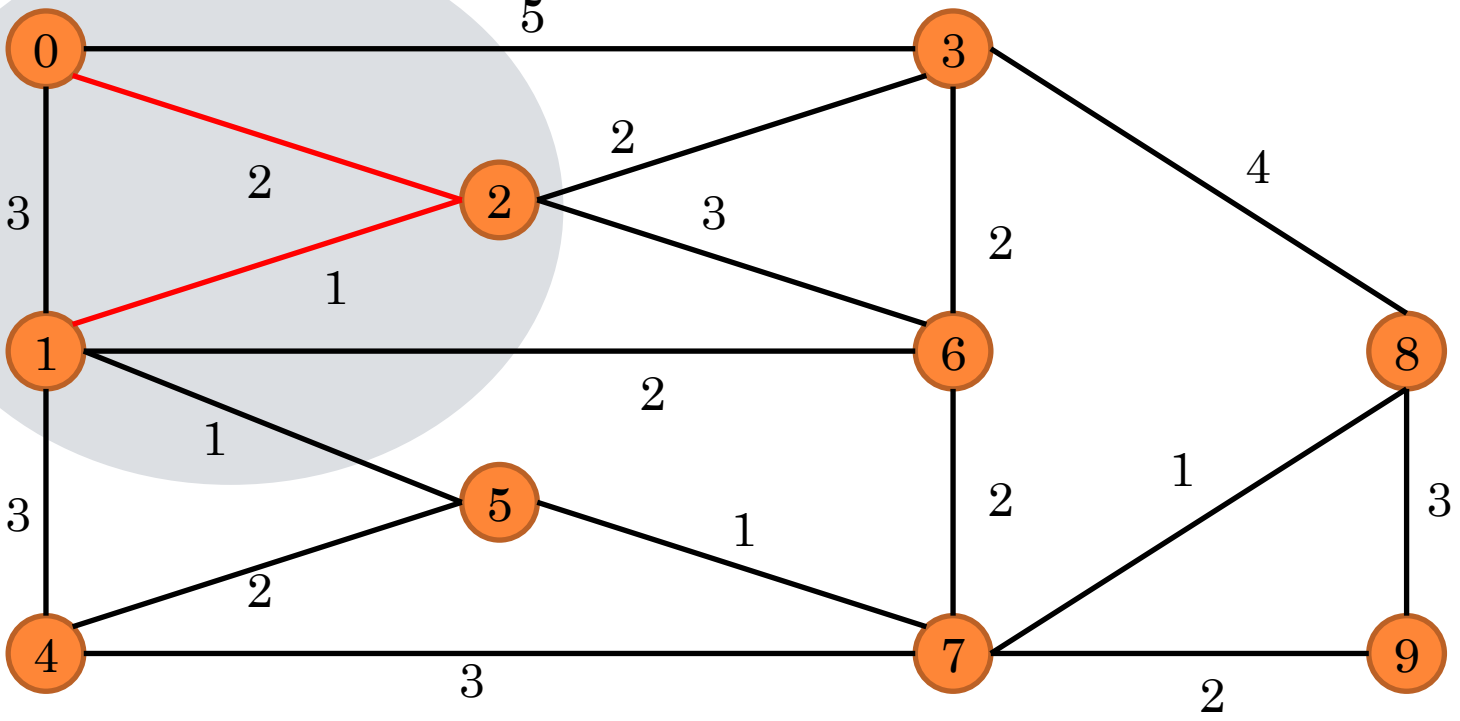
# 例1:最小木



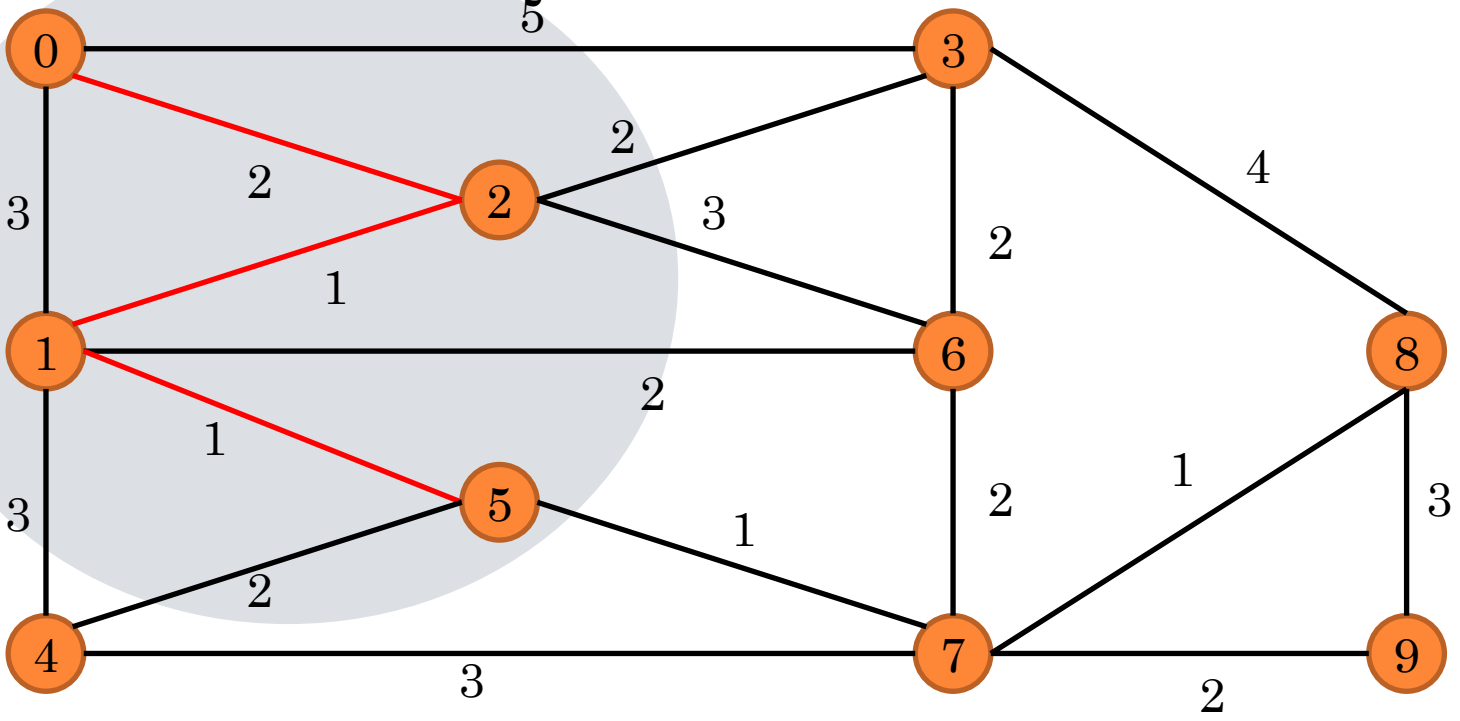
# 例2



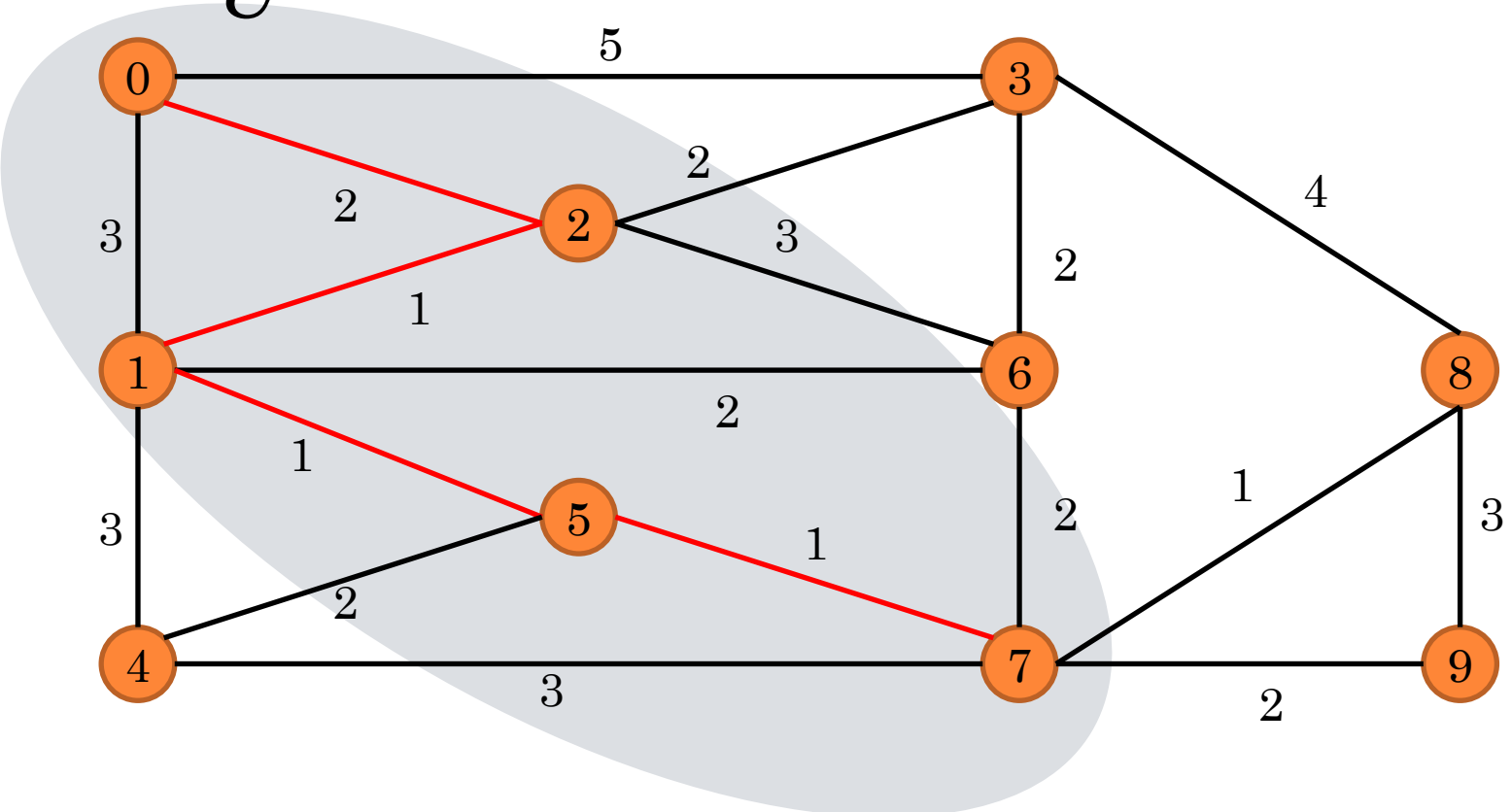
$U$



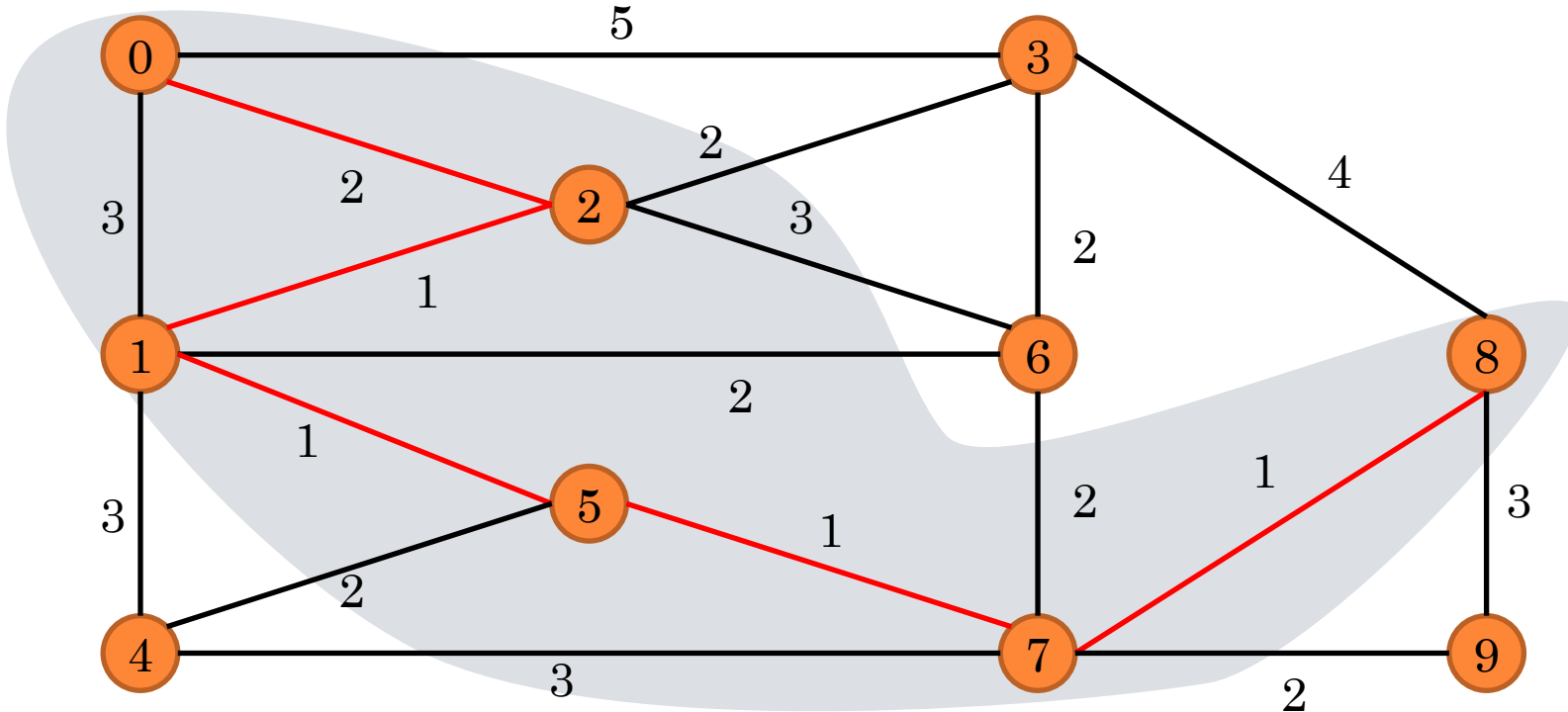
$U$



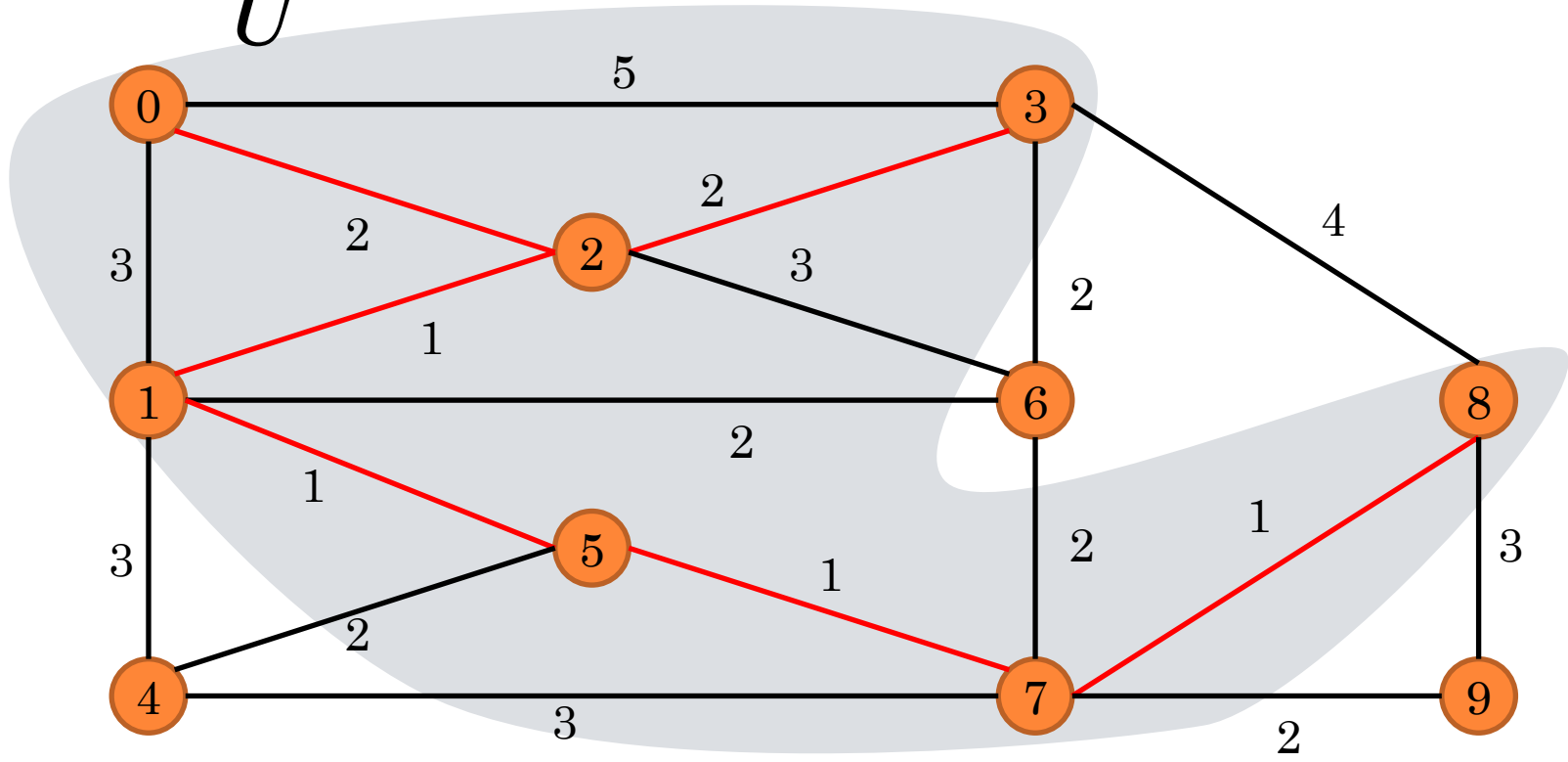
*U*



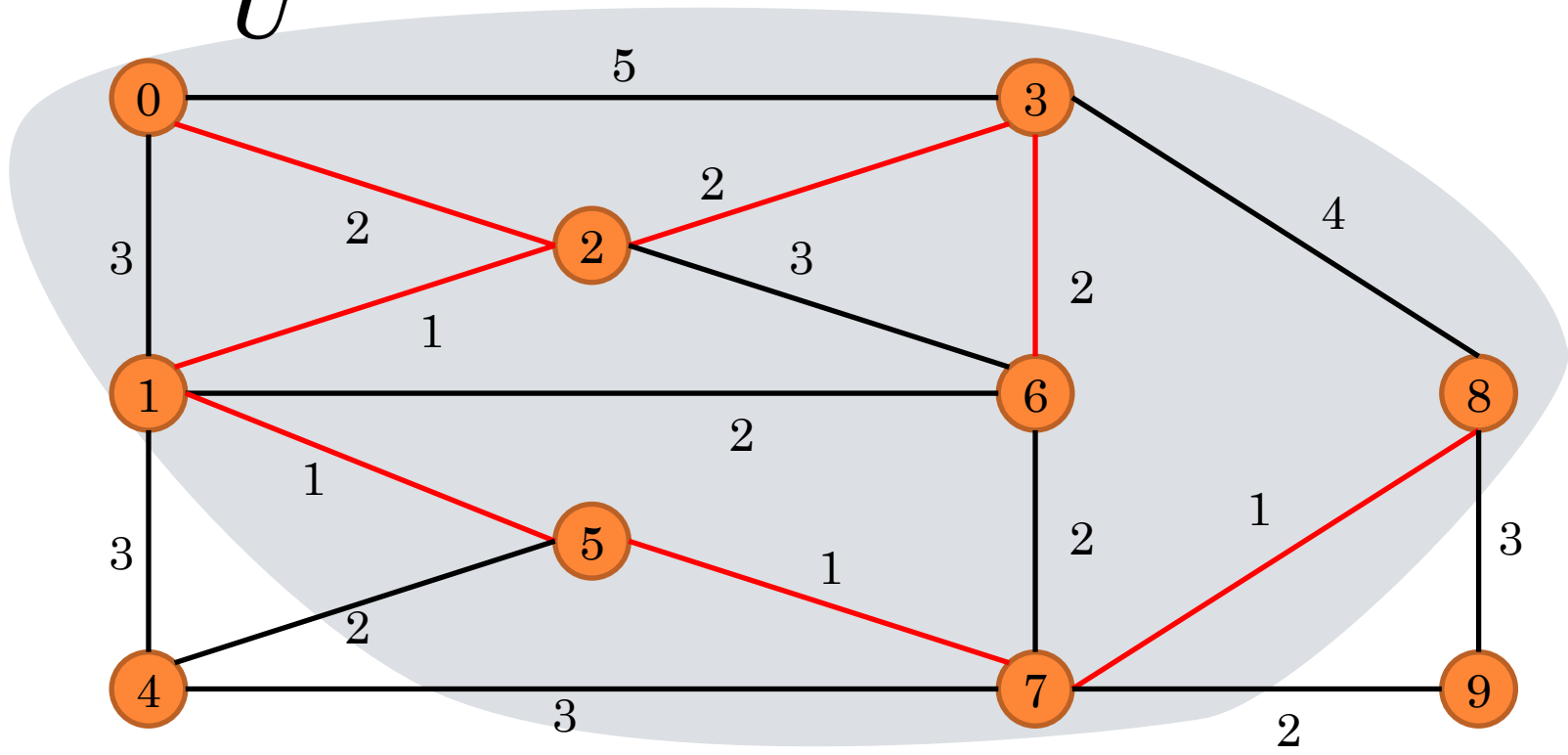
$U$



$U$

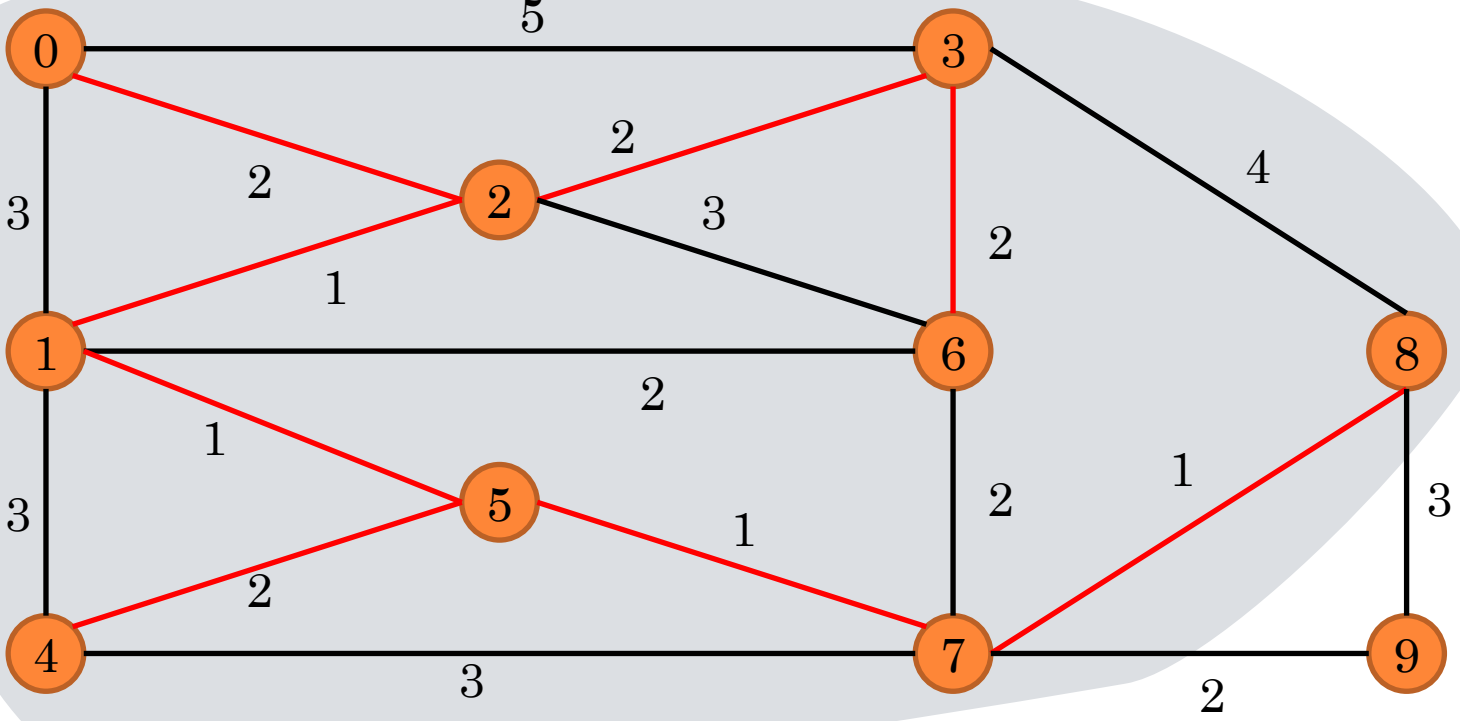


$U$

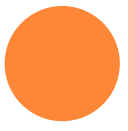
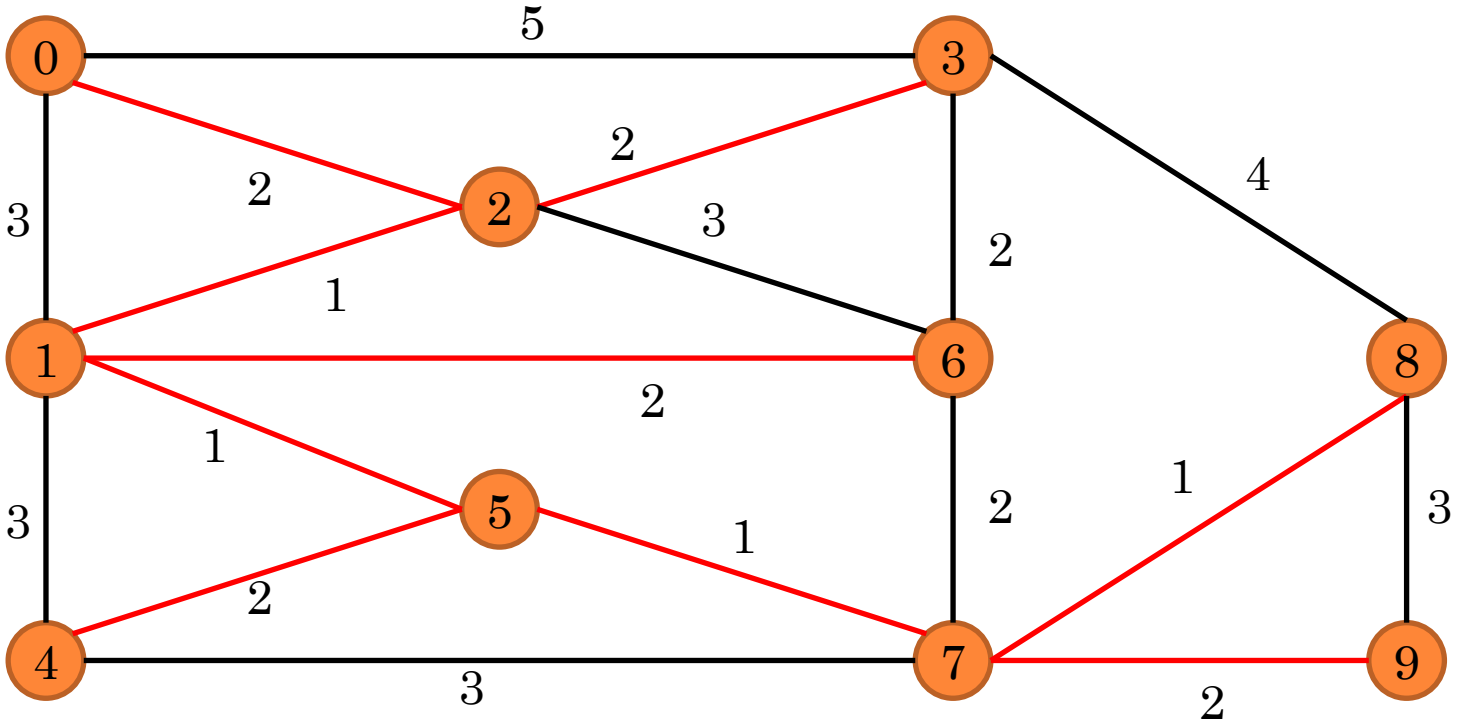




$U$



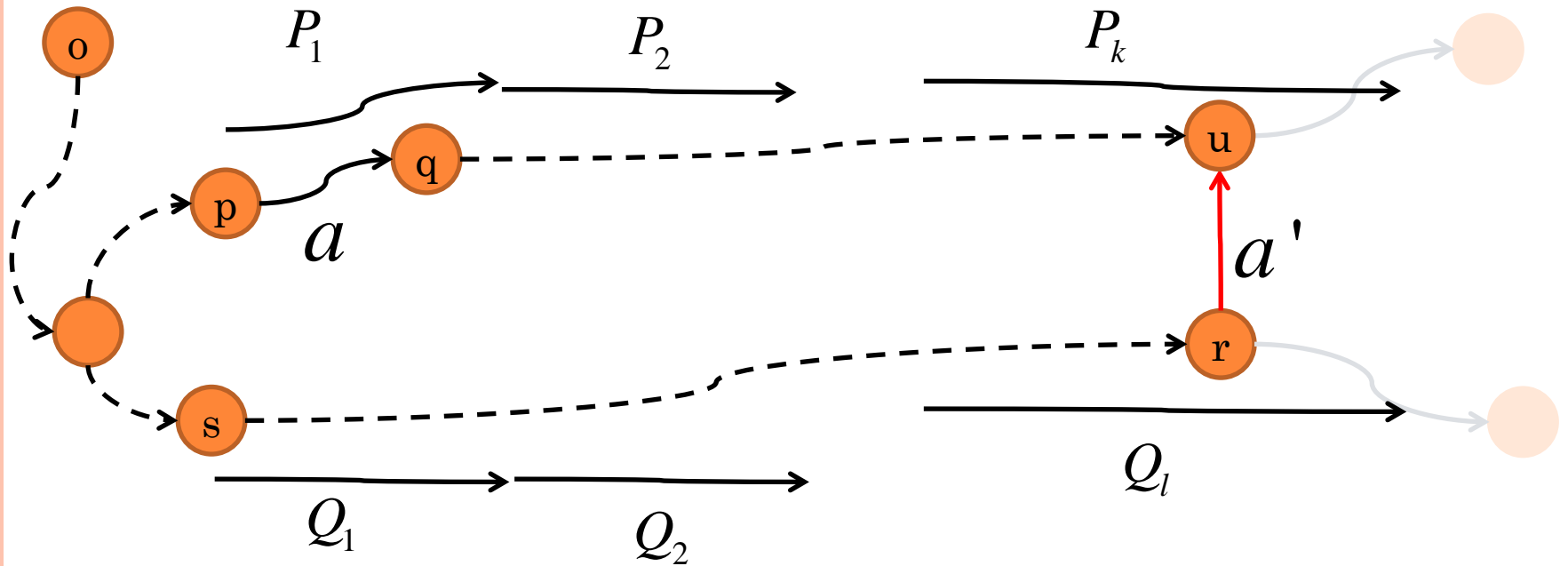
$U$



# JARNÍK-PRIMアルゴリズムが正しいこと

- Jarník-Primアルゴリズム実行中の木 $T$ は、 $U$ によって誘導される $G$ の部分グラフ $G(U)$ における最小木になっている。
- 証明
  - $T$ のある枝 $a$ を $T$ に含まれない枝 $a'$ に置き換えることで、より小さい木ができる ( $w(a') < w(a)$ ) ことを仮定して、矛盾を導く。





- $O$  を根とする木  $T$  において、弧  $a$  の代わりに弧  $a'$  としたほうが、重みが小さくなると仮定する。
- 上の枝で、弧  $a$  を先頭に連続して伸びた道を  $P_1$  とし、その後、下の枝で連続して伸びた道を  $Q_1$  とする。その後、 $P_2, Q_2$  と交互に伸びるとする。他の枝は無視する。
- 弧  $a'$  の両端の頂点は道  $P_k$  及び  $Q_l$  に属しているとする。

$P_i$  を構成する弧を  $\{a_0^i, a_1^i, \dots, a_{n(i)}^i\}$  、  $Q_i$  を構成する弧を

$\{b_0^i, b_1^i, \dots, b_{n(i)}^i\}$  とする。  $P_i$  の後  $Q_i$  が伸びることから

$$\forall i, \forall j, w(a_j^i) \leq w(b_0^i), w(b_j^i) \leq w(a_0^{i+1})$$

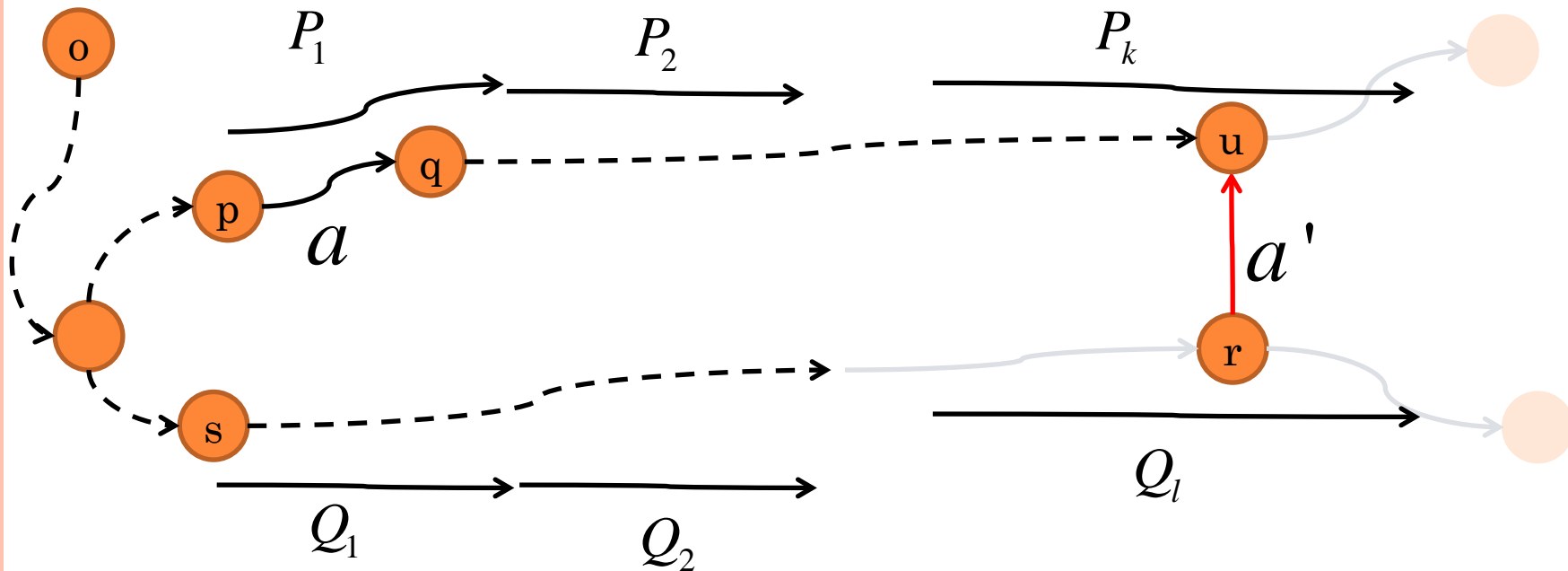
先頭の弧に注目すると、以下が成り立つ

$$\forall i, w(a_0^i) \leq w(b_0^i) \leq w(a_0^{i+1})$$

つまり、各道の先頭の弧の重みは以下を満たす。

$$\forall i, w(a) \leq w(a_0^i), w(a) \leq w(b_0^i)$$



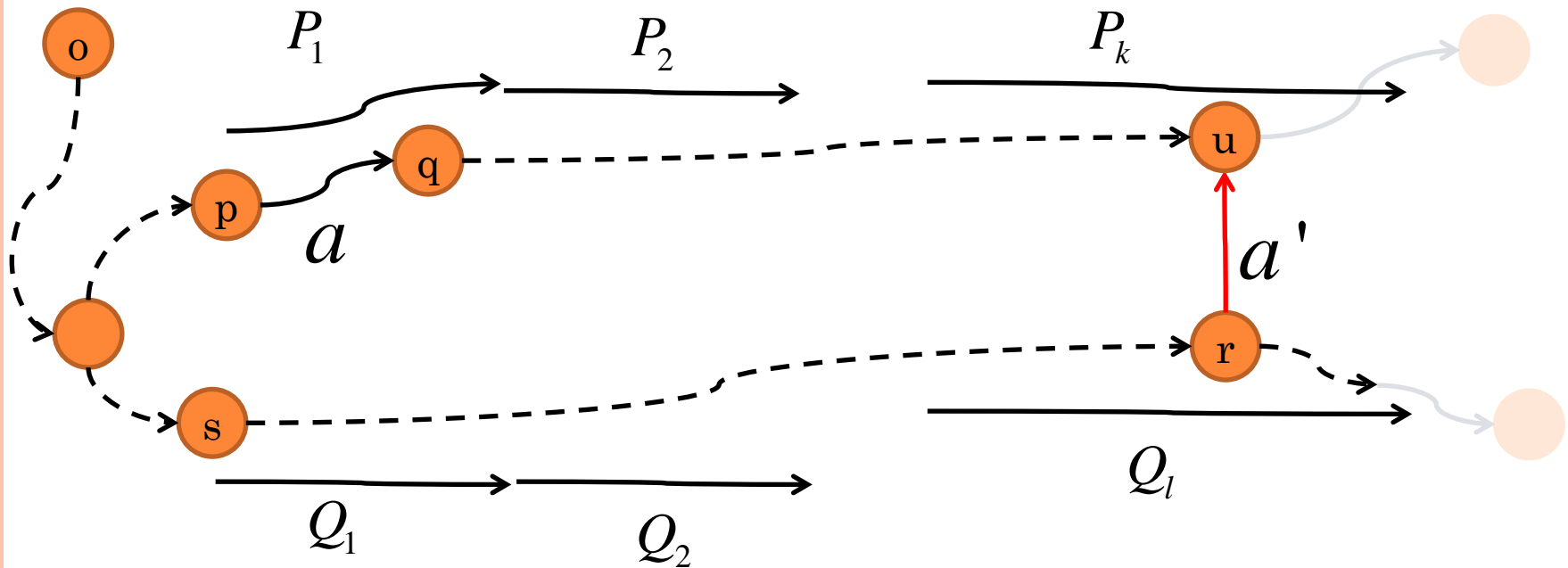


$k \leq l$  の場合、つまり上の枝が頂点  $u$  まで伸びたとき、下の枝は未だ頂点  $r$  に伸びていない場合を考える。このとき、上の道  $P_k$  が伸びるときに、枝  $a'$  がアルゴリズムによって採用されなかったことから

$$w(a) \leq w(b^k_o) \leq w(a')$$

となり矛盾する。





逆の  $k > l$  の場合、つまり上の枝が頂点  $u$  まで伸びたとき、下の枝は頂点  $r$  を過ぎて伸びていた場合を考える。下の道  $Q_l$  が伸びるときに、枝  $a'$  がアルゴリズムによって採用されなかったことから

$$w(a) \leq w(a^k_o) \leq w(a')$$

となり矛盾する。

