ネットワーク設計論レポート 2

27019679 グレゴリウスブライアン

December 28, 2021

前書き

このレポートのすべてのグラフはpythonで生成され、添付されたnotebook(.ipynb)で記載されている。 グラフの表現、描画はnetworkxライブラリーを使用している。また、アルゴリズム自体を説明すると記載されていない課題は ライブラリーのアルゴリズム関数を用いる場合がある(演習7の連結度を決定するための最小カットなど)
また、このレポートで集合は「{}」だけでなく「[]」でも表記される。(プログラムとの互換性のため)

演習 7 問題 1 (グラフ基本)

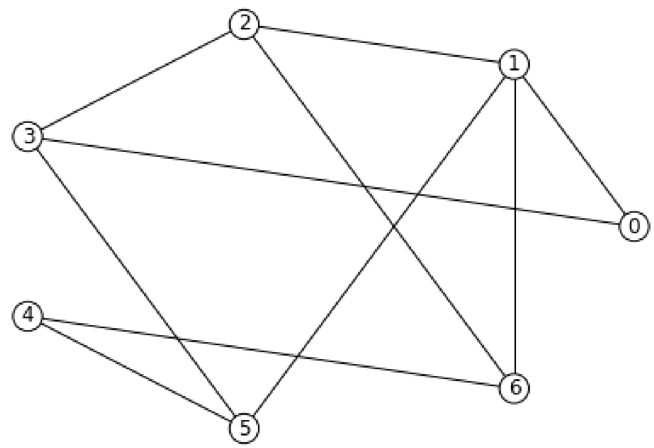

Figure 1: グラフの例

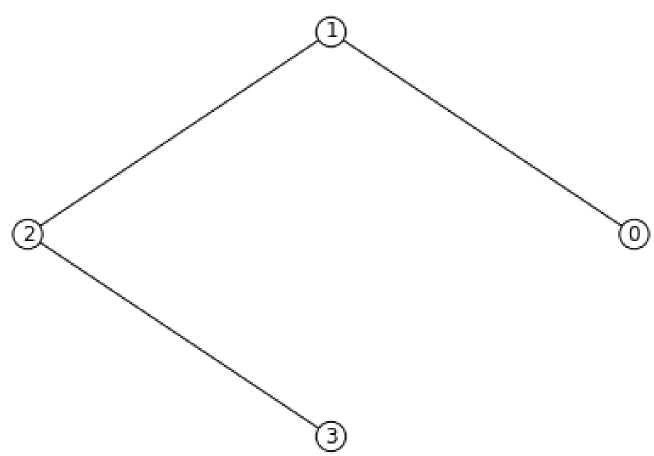

Figure 2: Figure 1 の部分グラフ例

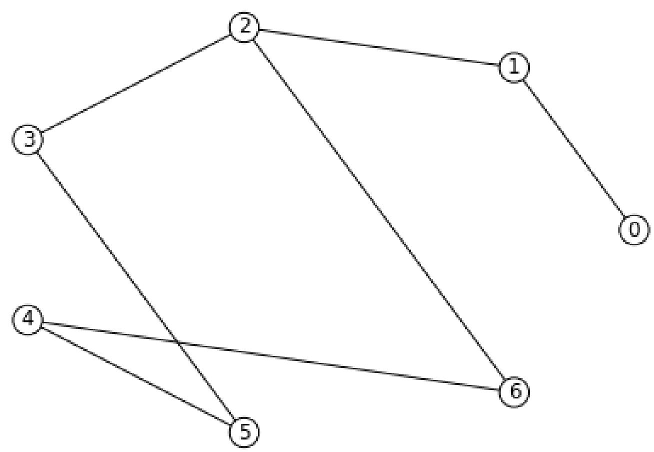

Figure 3: Figure 1 の全域部分グラフ例

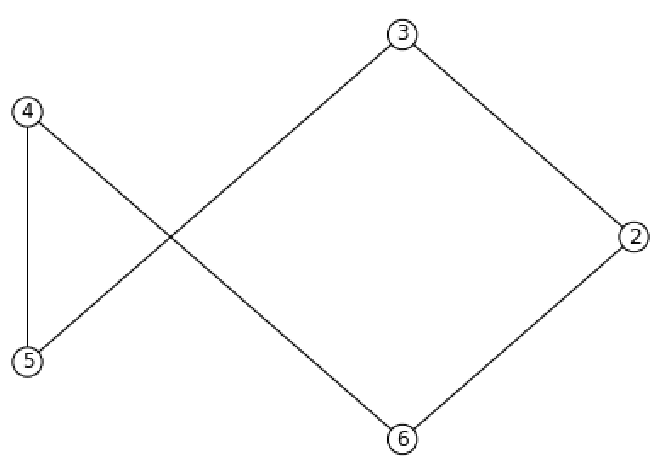

Figure 4: Figure 1 の頂点集合[2, 3, 4, 5, 6]で誘導される生成部分グラフ

演習７　問題２(頂点部分集合のカットサイズ)

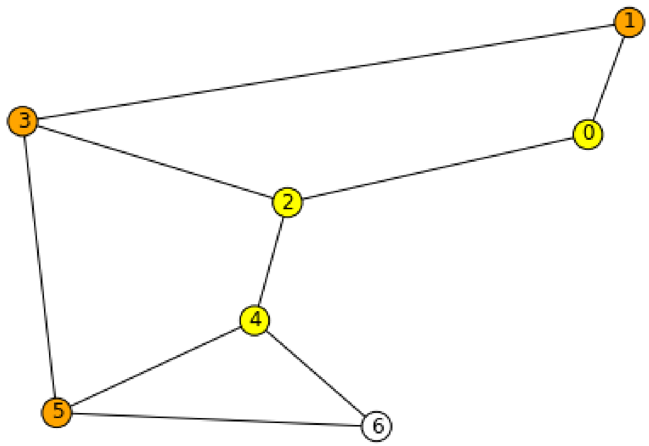

Figure 5: 頂点集合$[0, 2, 4]$と$[1, 3, 5]$が4辺連結のグラフ

Figure 5 で示される黄色の頂点集合$[0, 2, 4]$とオレンジ色の頂点集合$[1, 3, 5]$のカットサイズが4である。そのうち一例のカットはFigure 6である。

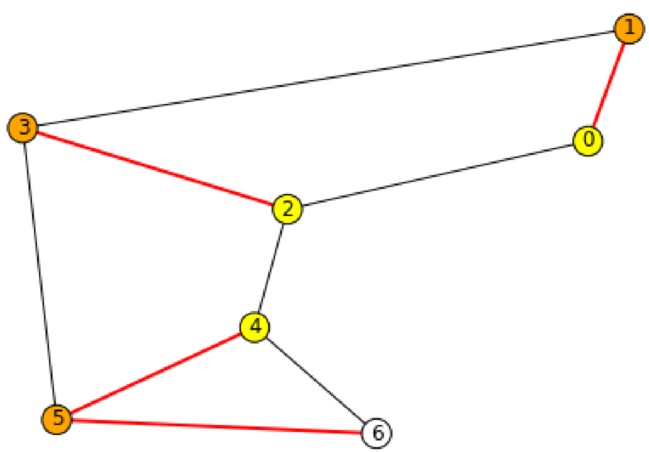

Figure 6: Figure 5 のカット例

演習 7　問題 3 (辺独立、点独立、辺素、内素)

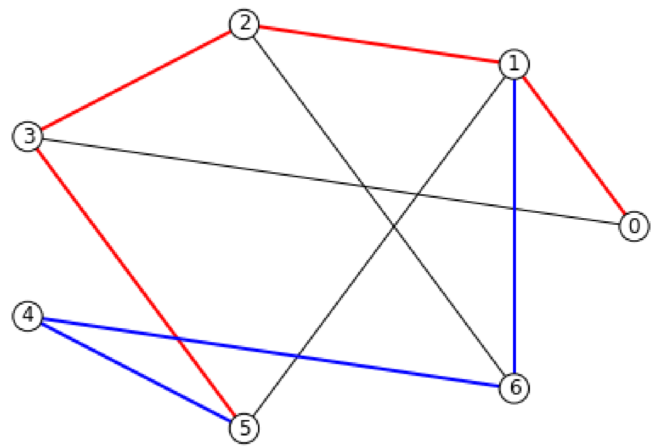

Figure 7: 互いに辺独立な経路集合 $[0, 1, 2, 3, 5]$ (赤)と $[1, 6, 4, 5]$ (青)

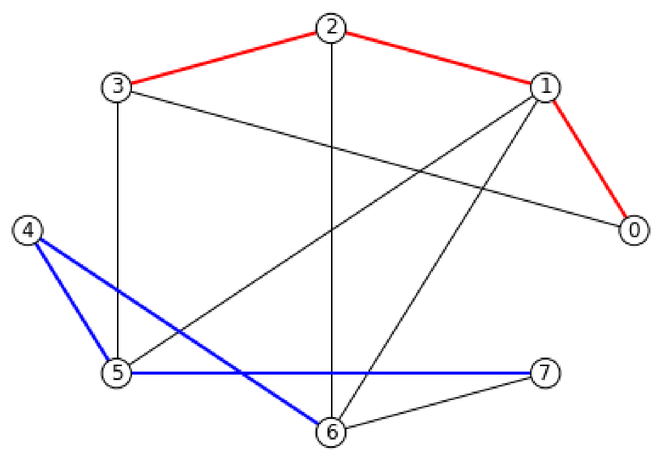

Figure 8: 互いに点独立な経路集合 $[0, 1, 2, 3]$ (赤)と $[6, 4, 5, 7]$ (青)

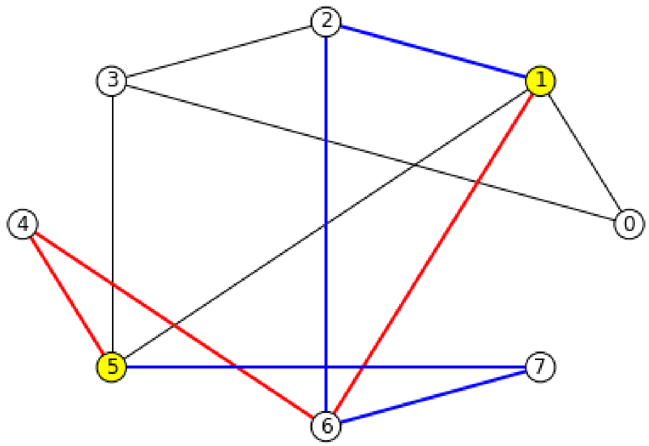

Figure 9: 互いに辺素な経路集合 $[1, 6, 4, 5]$ (赤)と $[1, 2, 6, 7, 5]$ (青)

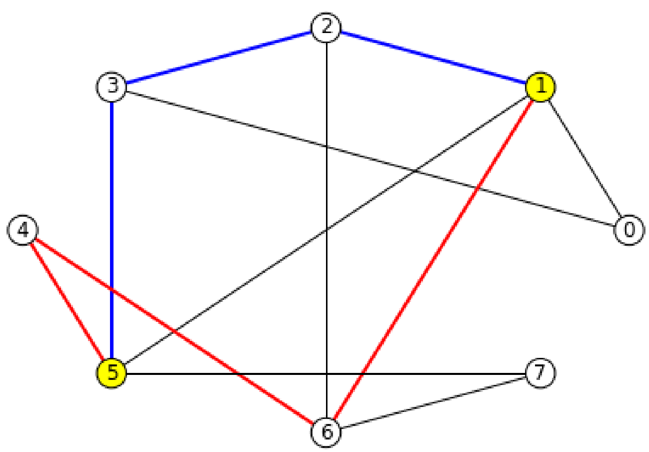

Figure 10: 互いに内素な経路集合 $[1, 6, 4, 5]$ (赤)と $[1, 2, 3, 5]$ (青)

演習７ 問題４ (頂点対の辺連結度)

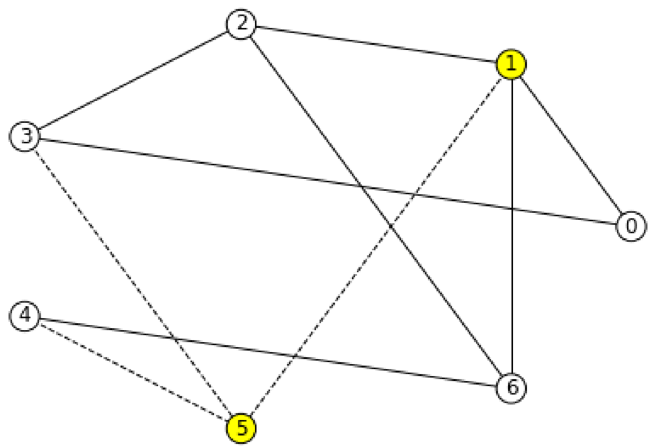

Figure 11: ３辺連結の頂点対(1, 5)とその最小カット（点線）

ただし、ここではすべての辺の容量は1であり、最小カットを決定するのにライブラリーの最大フロー・最小カット関数を用いた。

演習 7　問題 5 (グラフの連結度)

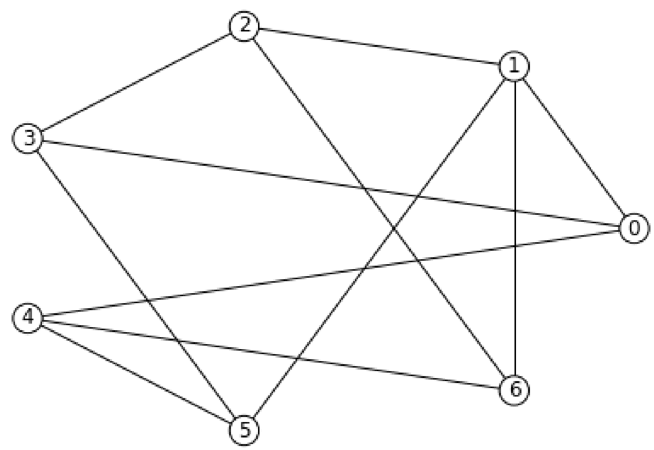

Figure 12: 連結度 3 の グラフ

上記の グラフはすべての頂点対の最小カットは 3 である。したがって、これは連結度 3 の グラフの一例である。 コードですべての頂点対の最小カットを調べている。

演習 7 問題 6 (頂点対の最大辺素経路)

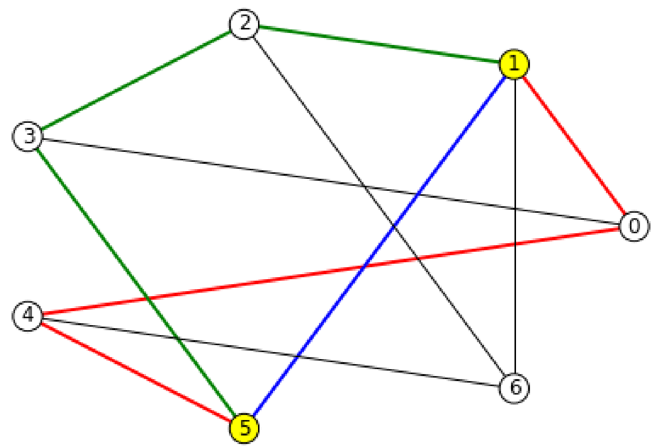

Figure 13: グラフと最大辺素経路集合

上記の グラフに対して、頂点対[1, 5]の辺素な経路集合の経路数が3であり、それぞれ、[1, 0, 4, 5](赤)、[1, 5](青)、[1, 2, 3, 5](緑)である。 そして、その同じサイズのカットは次通りである。

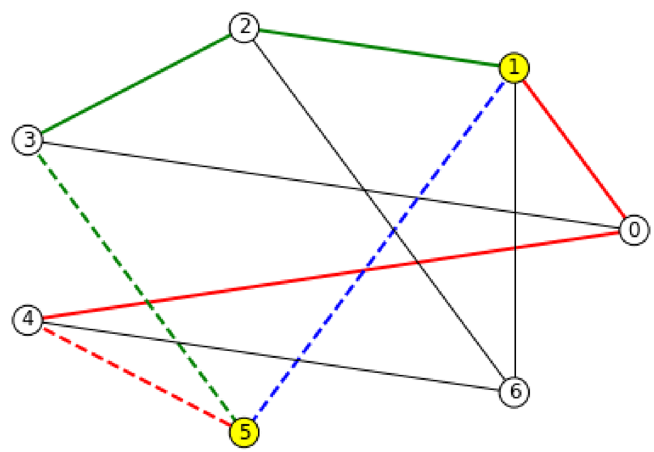

Figure 14: Figure 13のカット

演習 7 問題 7 (NA辺連結度)

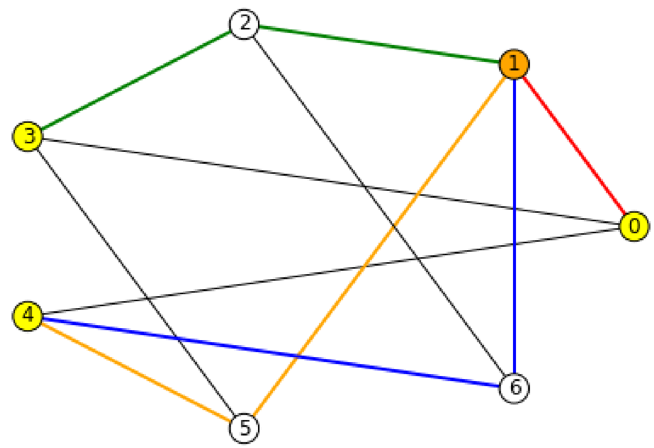

Figure 15: Node(1)-to-Area([0, 3, 4])連結度

上記の頂点1と領域[0, 3, 4]のNA辺連結は4であり、NA間の辺素経路をそれぞれの色で表せる。 M

演習８ 問題３(k辺連結保存))

Figure 12の２辺連結性保存を決めたい。コードでアルゴリズム自体を実装した。MA順序、辺順位決定の結果はFigure 16であり、 Figure 16で得られた辺順位が1と2を取るとFigure 17である2辺連結性保存全域部分グラフができる。ただし、MA順序決定の過程は Figure 18で示され、辺順位決定の過程は Figure 19で示される（図の大きさの関係で結果が先に示す）。辺順位決定は逆順で行っていることに注意（コードが書きやすい）。

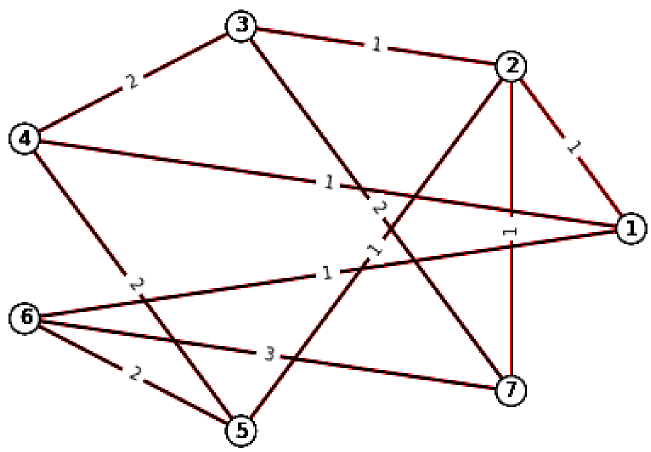

Figure 16: MA順序、辺順位決定結果

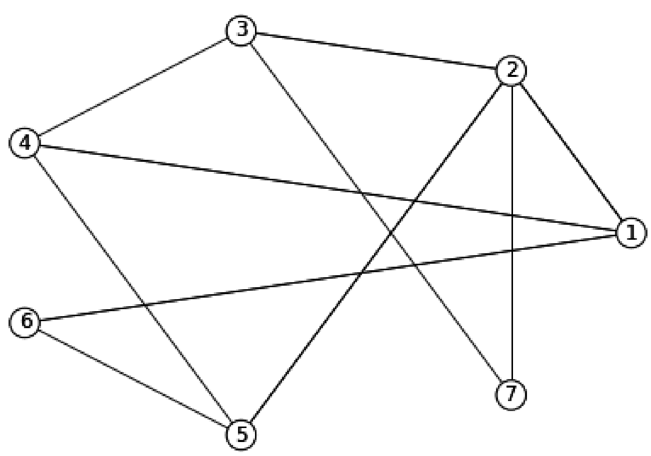

Figure 17: Figure 12の2辺連結性保存全域部分グラフ

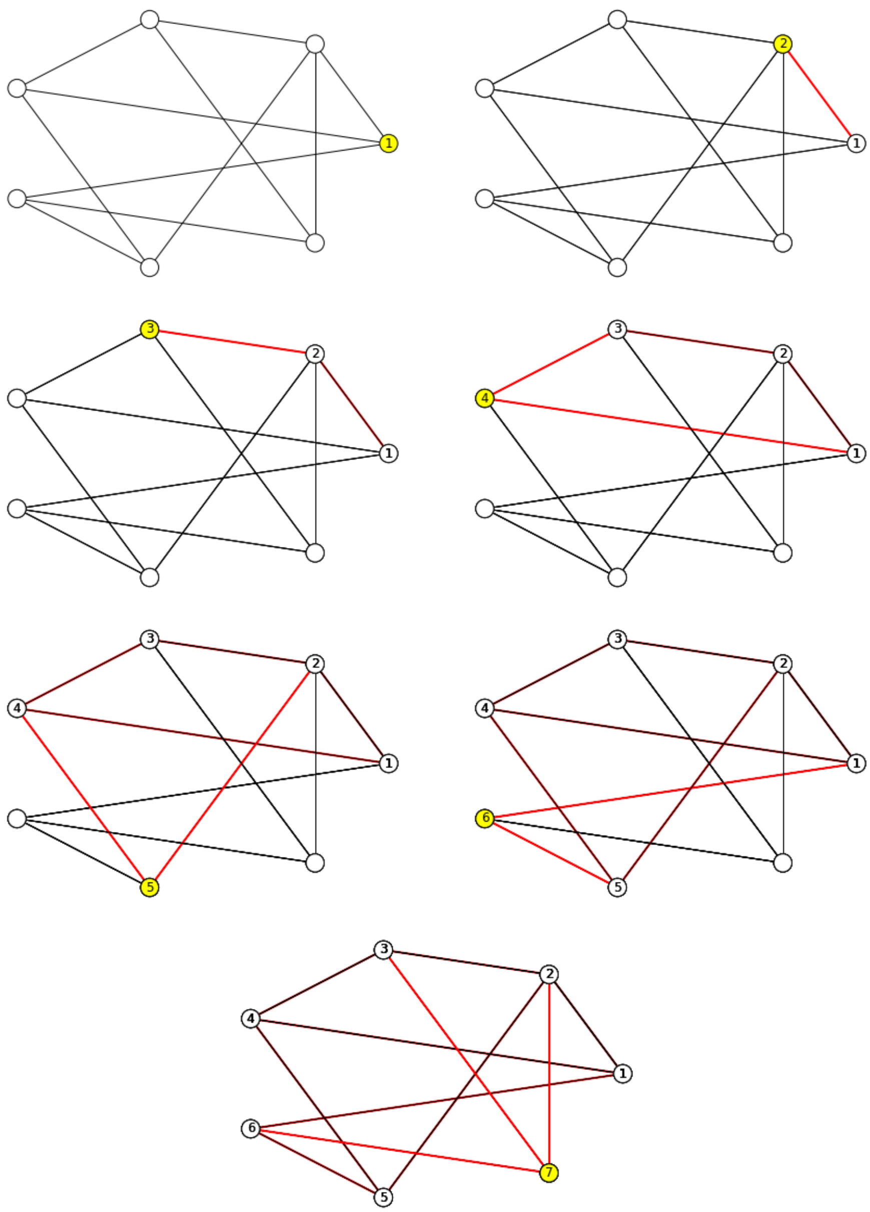

Figure 18: MA順序決定

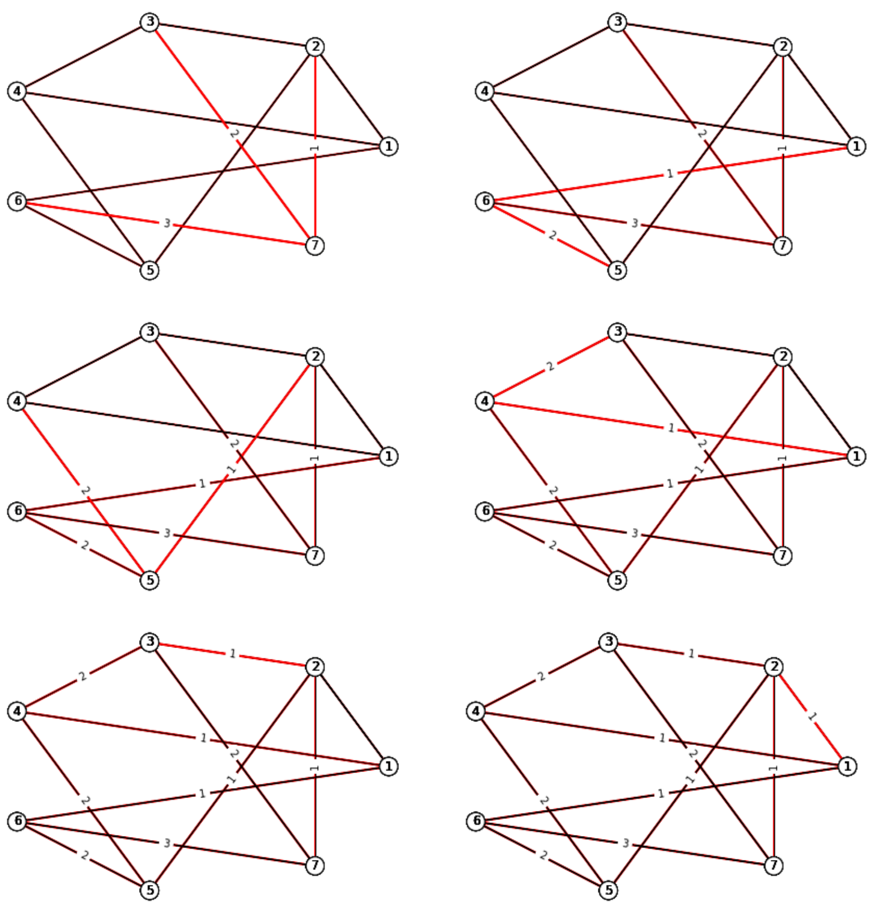

Figure 19: 辺の順位付け

演習 9 問題 1 (辺連結分解)

Figure 1のグラフに対して辺連結分解を行う。見やすくするように、このグラフをレイアウト変更したグラフはFigure 20である。このグラフの連結度は２である。言い換えれば、２辺連結成分はグラフの頂点集合と等しい(Figure 20 黄色頂点）。そして、このグラフの3辺連結成分はFigure 21通りである。同じ色の頂点は同じ成分である。つまり、$V = [[0], [1, 2, 3, 5, 6], [4]]$。この方法はコード中のk_edge_decomposition(G,k)で実装され、 k_edge_decomposition(G,3)は３辺連結成分を計算する形になっている。

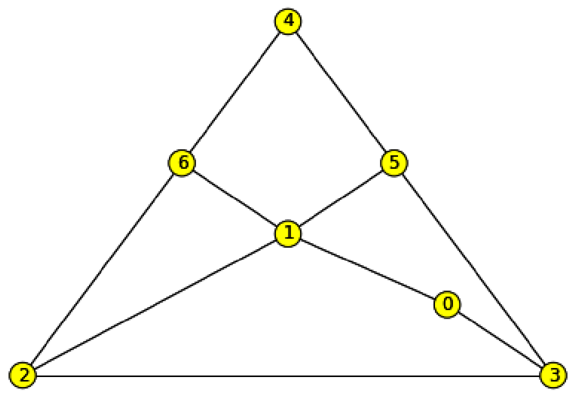

Figure 20: Figure 1 の２辺連結成分

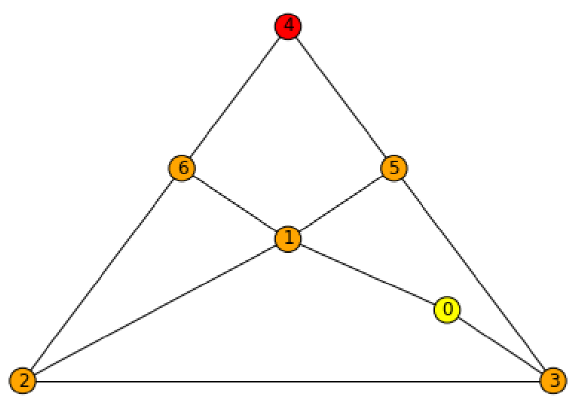

Figure 21: Figure 1 の３辺連結成分

演習9 問題３(木の辺付加)

連結性保存の時、２辺連結性保存のグラフがあるが、そのグラフの１辺連結性の保存を今回扱う木とする。深さ優先探索で 葉に順序をつけたのがFigure 22である。ただし、探索は赤頂点から開始されるとする。そして、アルゴリズム通り辺を付加すれば Figure 23の２辺連結グラフになる。

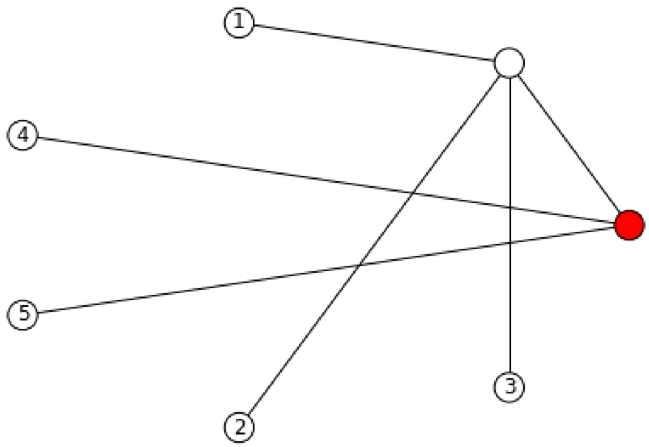

Figure 22: 赤頂点から深さ優先探索、葉には順序をつける

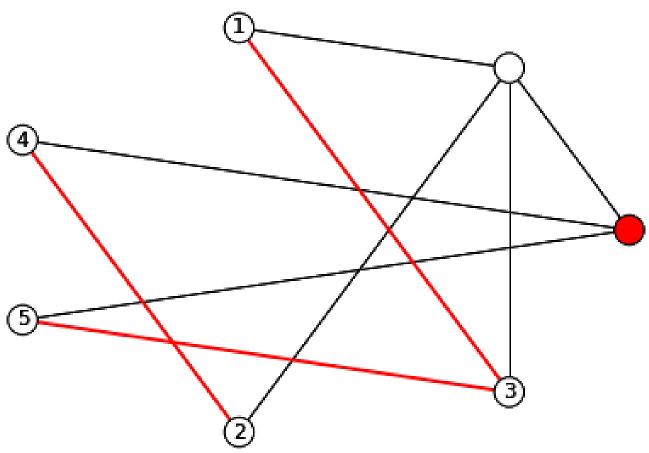

Figure 23: 木の辺付加（付加した辺は赤）

演習１０問題１(直径増加抑制)

Figure 24のグラフは直径４のグラフである。どの２つの辺を消しても直径が４のままのように保護辺を決めたい。 Figure 25がFigure 24の辺を頂点とするグラフであり。同時に２つの辺が切れたらグラフの直径が４より多くなったら辺を引く。 それの頂点被覆（黄色頂点の集合）が保護辺の集合となり、Figure 26で示す。

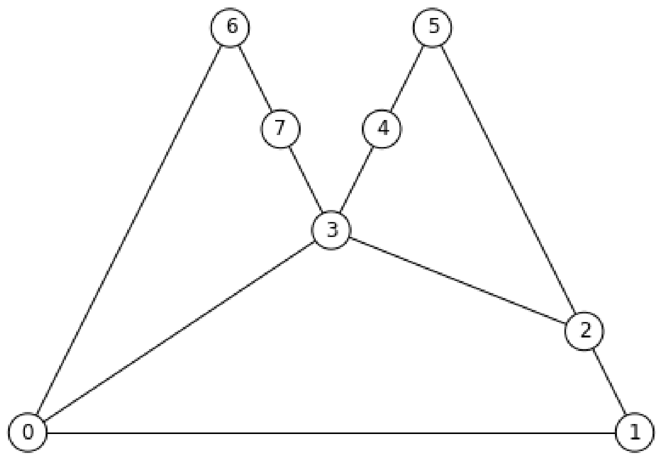

Figure 24: 直径４のグラフ

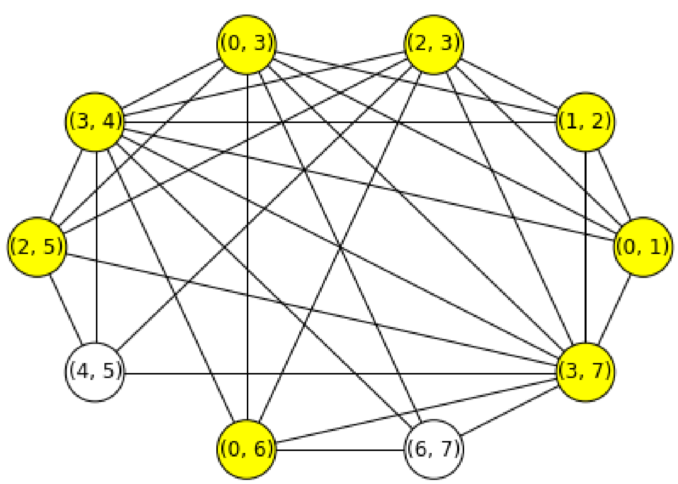

Figure 25: Figure 24の辺を頂点とするグラフ、黄色頂点は頂点被覆集合

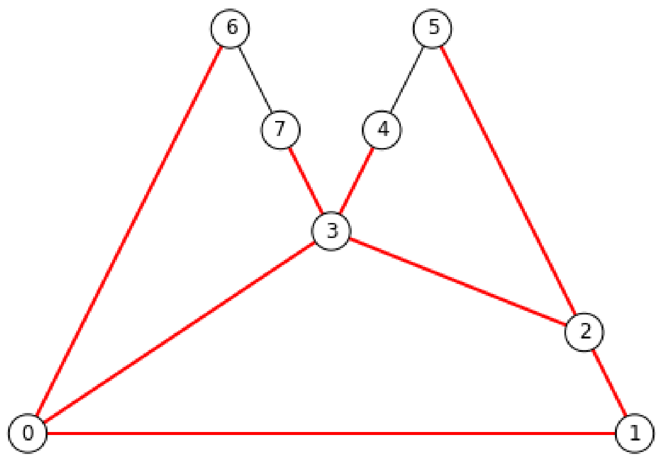

Figure 26: 元のグラフの保護辺

演習１１問題１と２(ダイクストラ法最短路)

Figure 27のグラフに対して黄色頂点を根とする最短路木を作りたい。djikstraを用いた最短路木の探索 はFigure 29で示される。ただし、Figure 29の黄色頂点は今探索中の頂点であり、オレンジ色の頂点は次に探索する頂点である。また、各頂点に記載されている数字は暫定最短路であり、赤いエッジは全体の最短路木になるエッジを示す。 出来上がった最短路木はFigure 28で示される（頂点に付いている数字は根からの距離を示す）。

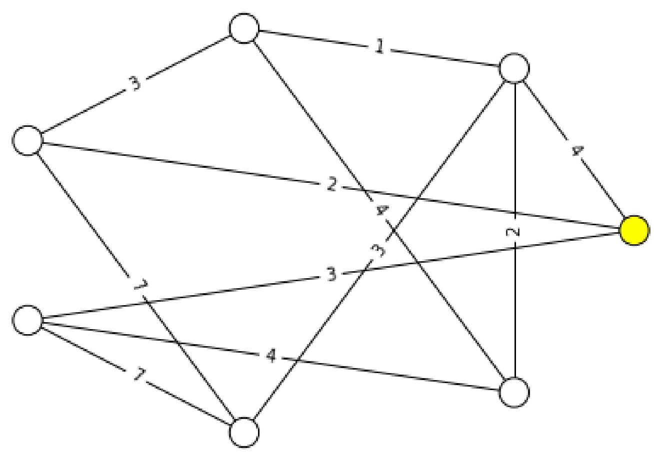

Figure 27: 重み付きグラフ

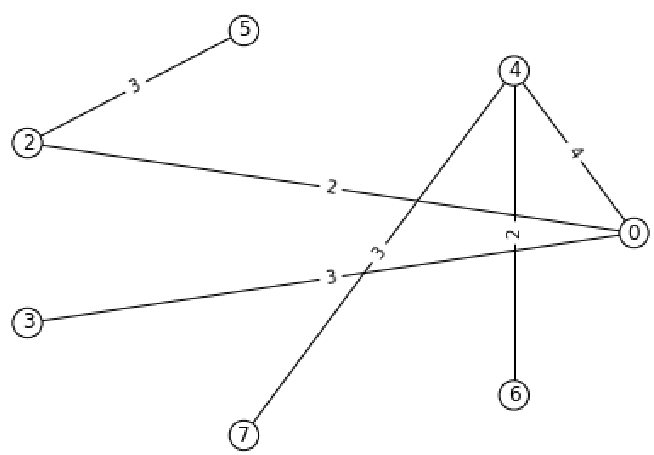

Figure 28: Figure 27の最短路木

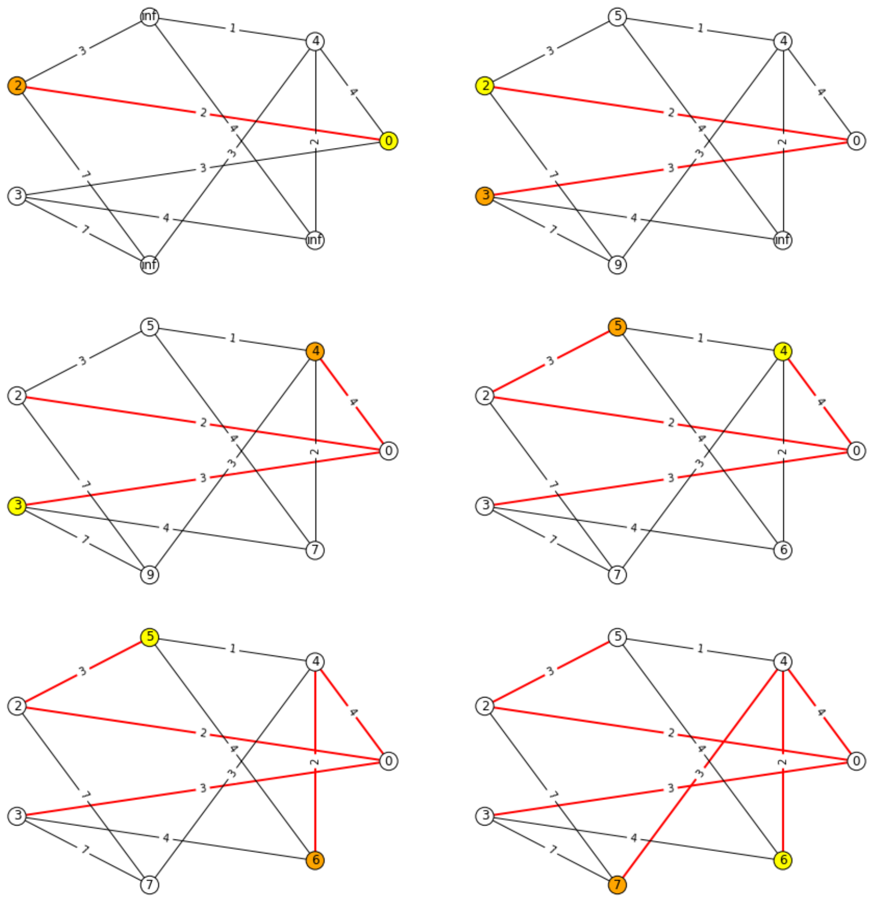

Figure 29: djikstra最短路過程

演習１１問題５(最大フロー)

ソースノードからシンクノードのパス有無を探すのにBFSを用いた。BFSも最大フローアルゴリズムも添付した.ipynbにゼロから実施した。 Figure 30のグラフに対して0から3の最大フローを求める。その最大フローはFigure 31で示され、最大フローは47であることが分かる。 最大フローを求める残余ネットワークの過程はFigure 32である。ここで、矢先に近い数字はその辺の容量である。

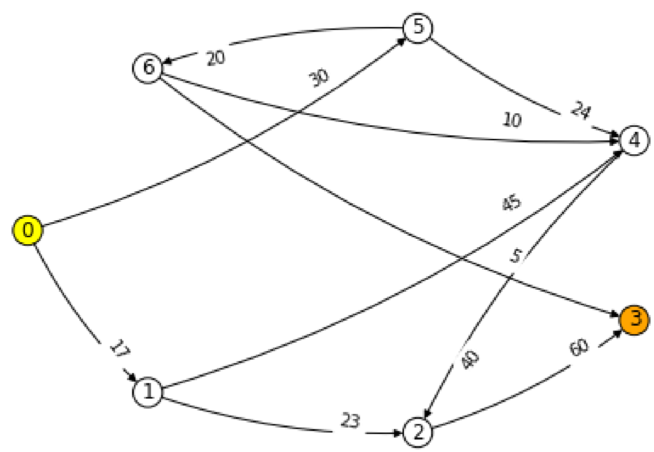

Figure 30: 容量付き有向グラフ

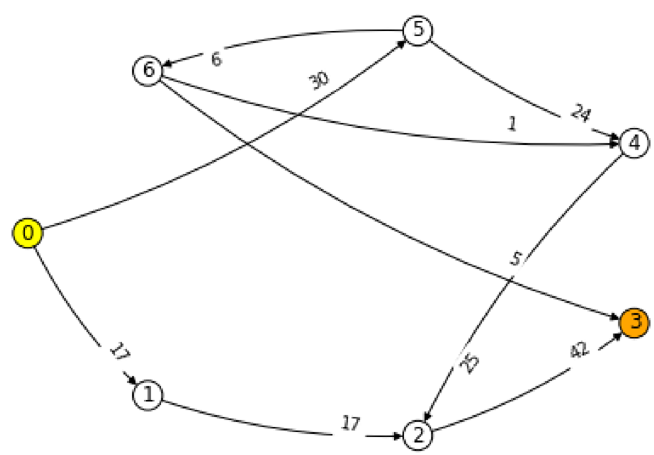

Figure 31: Figure 30の最大フロー

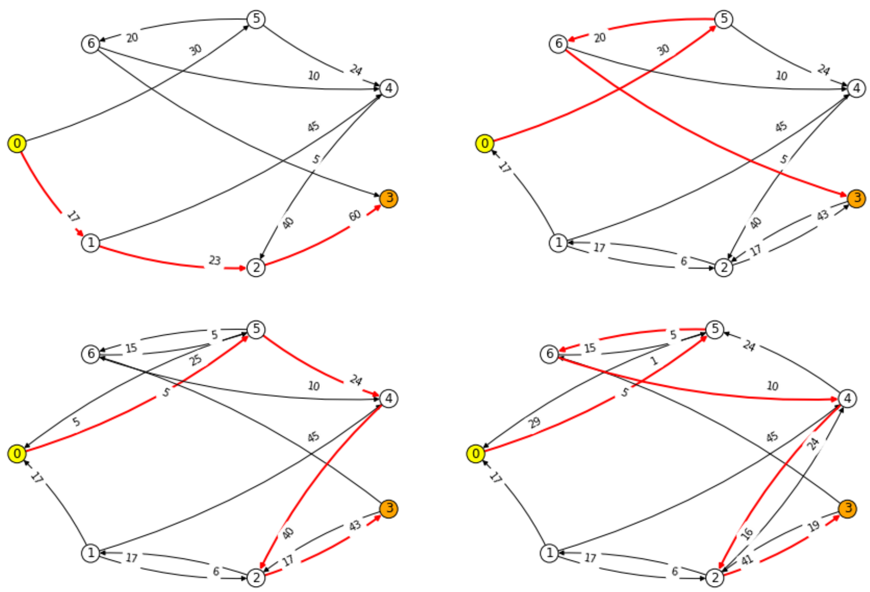

Figure 32: Ford-Fulkerson最大フロー過程

演習１１問題６(最小コストフロー)

Figure 33は容量とコスト付きの有向グラフであり、最大フローを計算したらFigure 34になる。しかし、Figure 35のように最大フローを求めたあとの残余グラフにはシンクノードにおいて負閉路が存在する。この閉路消したあとの最大フロー最小コストグラフはFigure 36である。

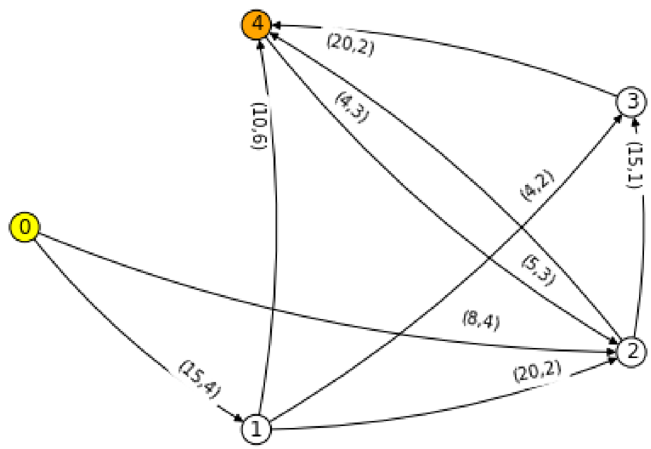

Figure 33: 容量とコスト付き有向グラフ

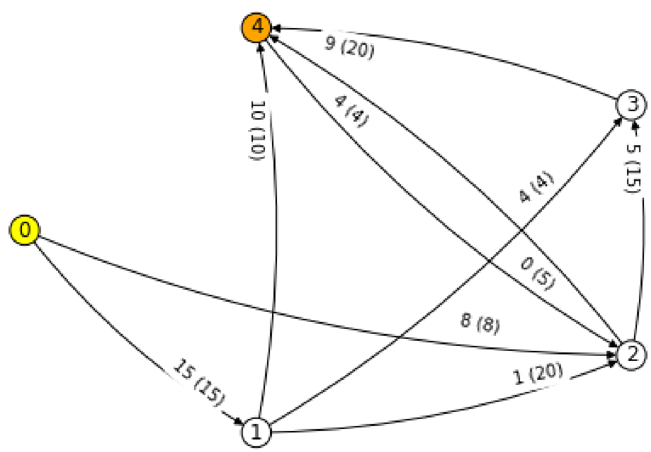

Figure 34: 初期最大フロー

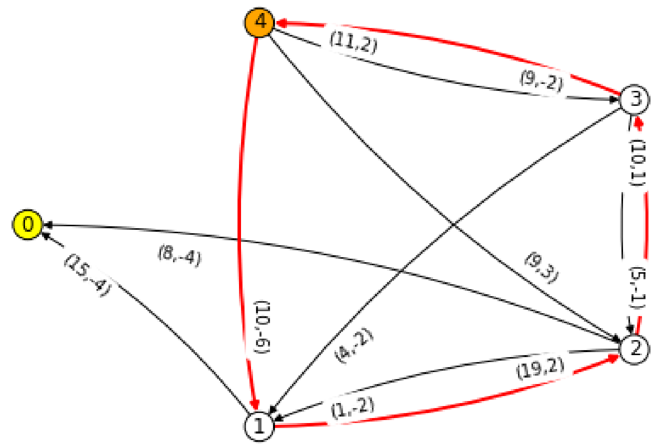

Figure 35: 残余グラフに負閉路存在

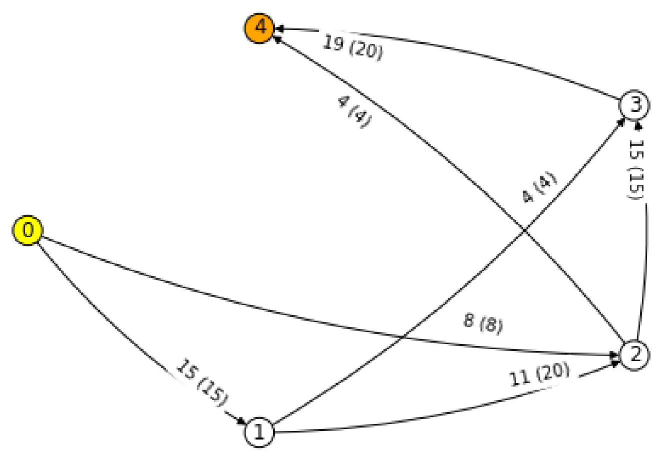

Figure 36: 最小コスト最大フロー

演習１１問題７＆８（多品種最大フロー）

Figure 37のグラフを考える。Figure 37上にFigure 38のようなソースノード(S)とシンクノード(T)の対が与えられたとき、多品種最大フローを求める。 多品種最大フロー問題を線形計画問題として定式化してpythonのscipyライブラリーを用いて線形計画問題を解いた。その結果はFigure 38であり、多品種最大フローが17であることがわかる。 この問題は以下の線形計画問題として定式化できる。ただしここで、x_{i-j}は(i, j)に流れるフローである。

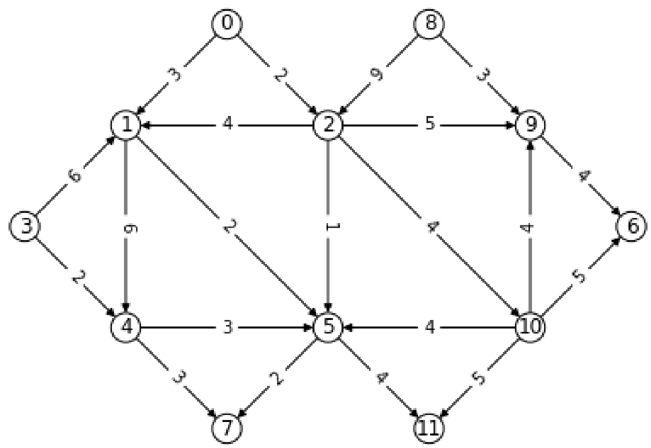

Figure 37: 容量付き有向グラフ

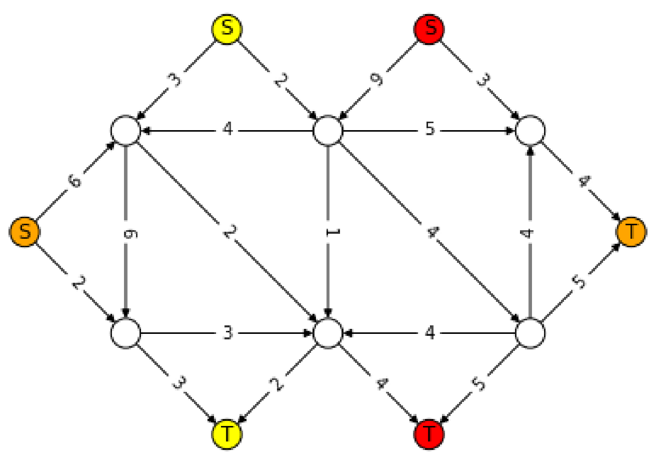

Figure 38: ソースとシンクノードペア

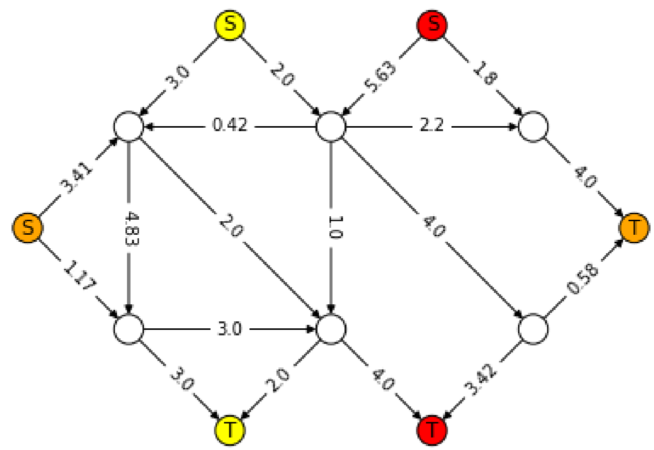

Figure 39: 多品種最大フロー

目的変数（最大化）：

$$z = x_{0-1} + x_{0-2} + x_{3-1} + x_{3-4} + x_{8-2} + x_{8-9}$$

制約条件：

$$x_{0-1} + x_{0-2} = x_{4-7} + x_{5-7}$$

$$x_{3-1} + x_{3-4} = x_{9-6} + x_{10-6}$$

$$x_{8-2} + x_{8-9} = x_{10-11} + x_{5-11}$$

$$x_{1-4} + x_{1-5} = x_{0-1} + x_{3-1} + x_{2-1}$$

$$x_{2-5} + x_{2-9} + x_{2-10} + x_{2-1} = x_{0-2} + x_{8-2}$$

$$x_{4-5} + x_{4-7} = x_{3-4} + x_{1-4}$$

$$x_{5-7} + x_{5-11} = x_{1-5} + x_{2-5} + x_{4-5} + x_{10-5}$$

$$x_{9-6} = x_{8-9} + x_{2-9} + x_{10-9}$$

$$x_{10-6} + x_{10-11} + x_{10-9} + x_{10-5} = x_{2-10}$$

$$0 \leq x_{0-1} \leq 3$$

$$0 \leq x_{0-2} \leq 2$$

$$0 \leq x_{1-4} \leq 6$$

$$0 \leq x_{1-5} \leq 2$$

$$0 \leq x_{2-5} \leq 1$$

$$0 \leq x_{2-9} \leq 5$$

$$0 \leq x_{2-10} \leq 4$$

$$0 \le x_{2-1} \le 4$$
$$0 \le x_{3-1} \le 6$$
$$0 \le x_{3-4} \le 2$$
$$0 \le x_{4-5} \le 3$$
$$0 \le x_{4-7} \le 3$$
$$0 \le x_{5-7} \le 2$$
$$0 \le x_{5-11} \le 4$$
$$0 \le x_{8-2} \le 9$$
$$0 \le x_{8-9} \le 3$$
$$0 \le x_{9-6} \le 4$$
$$0 \le x_{10-6} \le 5$$
$$0 \le x_{10-11} \le 5$$
$$0 \le x_{10-9} \le 4$$
$$0 \le x_{10-5} \le 4$$