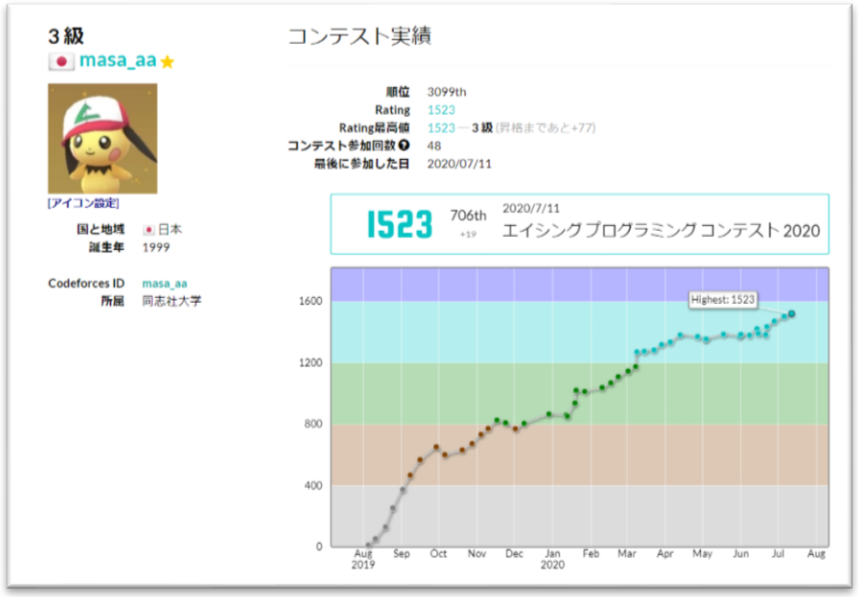
2020･07･23　ゼミ資料

グラフ理論入門

まさぁぁ

自己紹介

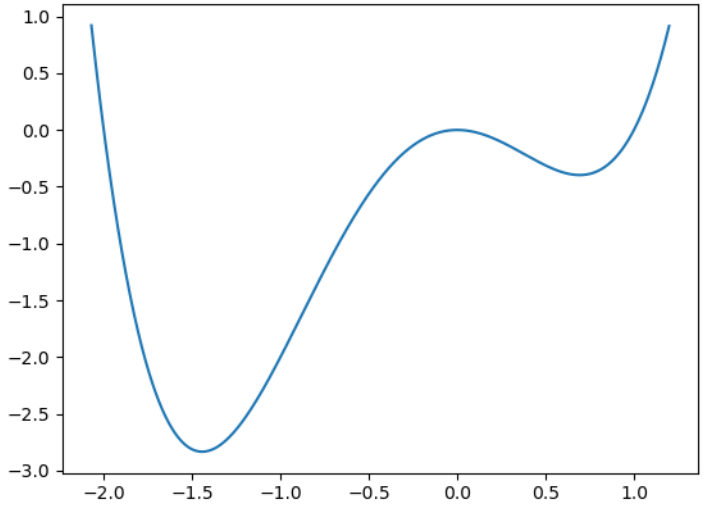
- 理工学部数理システム学科の4年生です.
- 競技プログラミングで上位5%~10%ぐらい
（データ構造とかアルゴリズムにちょっと詳しい）
- 機械学習は始めたての状態
- 大学がなくて生活が崩壊してます．タスケテ



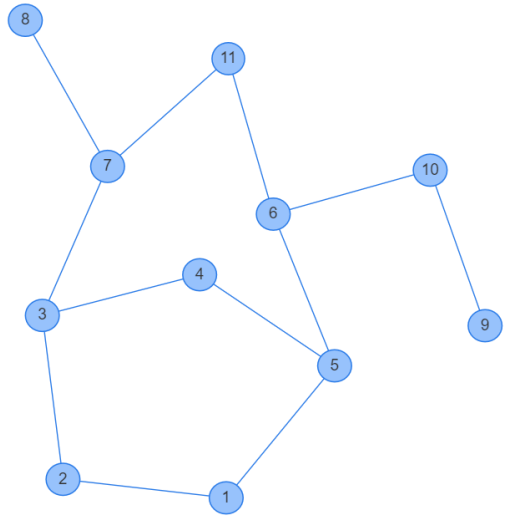
グラフ理論ってなんぞや

グラフ：対象物を丸，関係を線で表すもの

グラフ理論 ：グラフの持つ様々な性質を解明し、
日常の様々な場面で利用することを目的とする数学分野の1つ。



グラフ理論で扱わない例



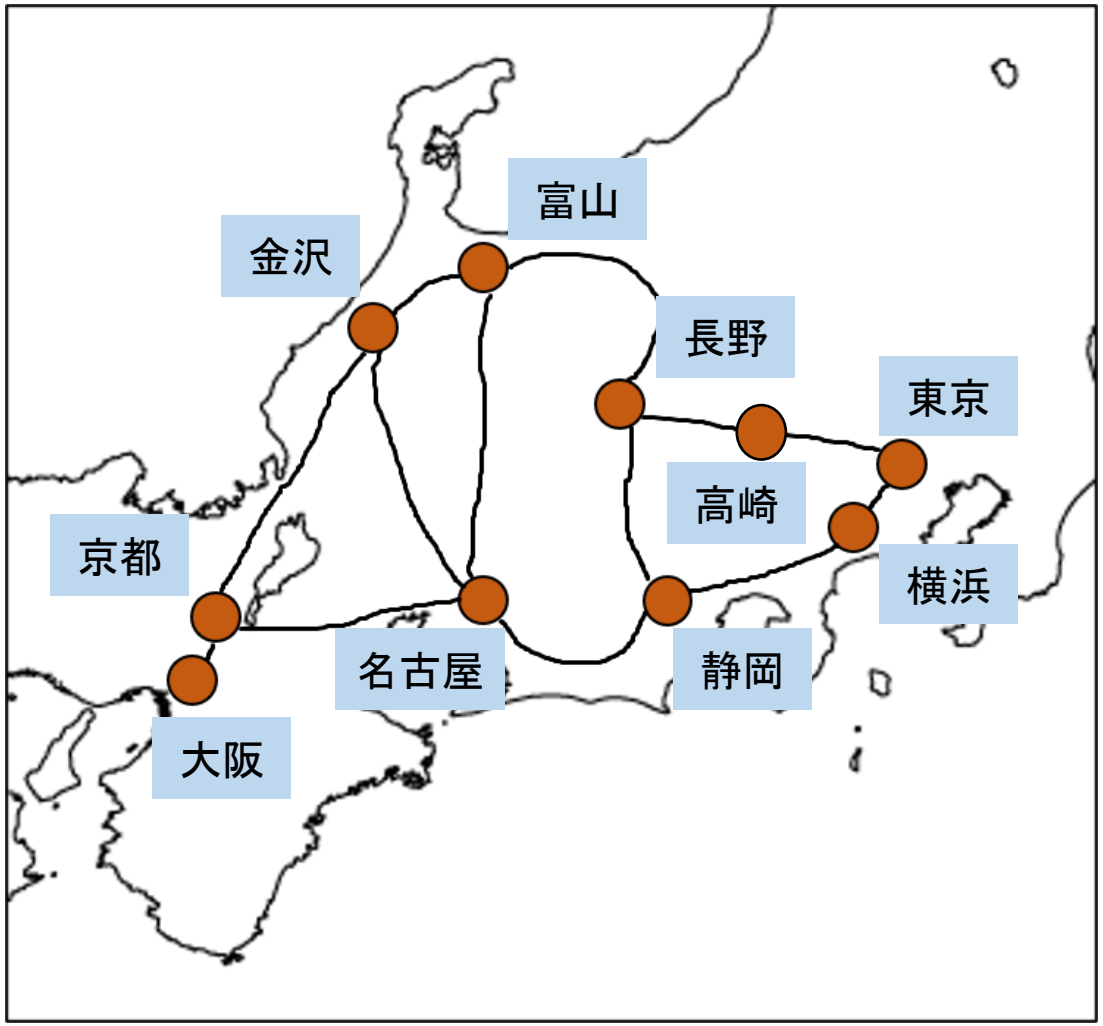
グラフ理論で扱うグラフの例

応用例（流れだけ見てみる)

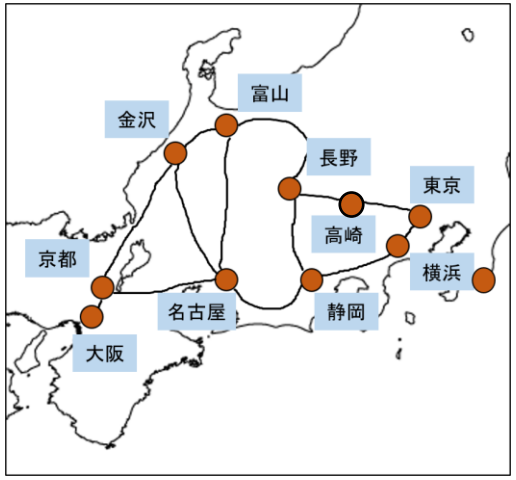
路線の候補が(都市A, 都市B, 距離)のように与えられる.
任意の都市同士が行き来可能になるためには,
少なくとも何km分の線路が必要か?

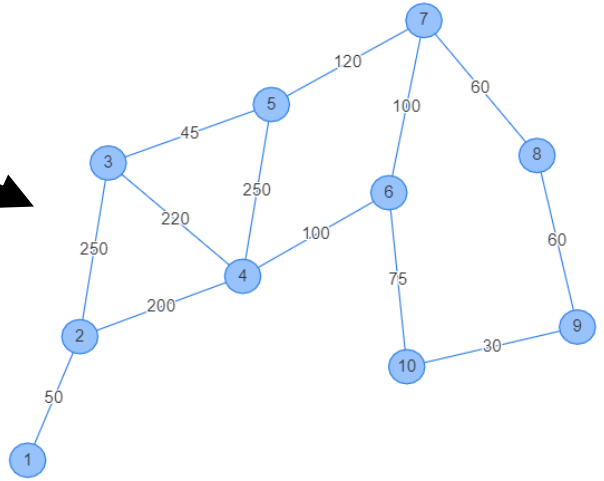
例

大阪-京都 50
京都-金沢 250
京都-名古屋 200
名古屋-静岡 120
名古屋-金沢 220
金沢-富山 45
富山-名古屋 250
富山-長野 120
長野-静岡 100
長野-高崎 60
高崎-東京 60
静岡-横浜 75
横浜-東京 30

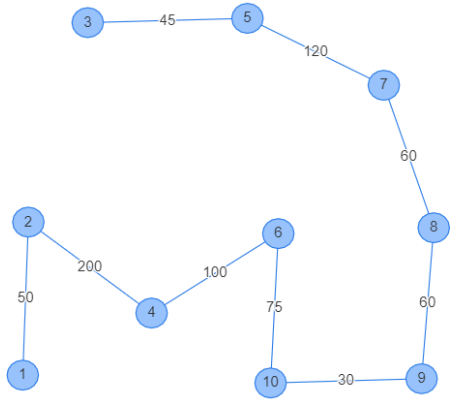


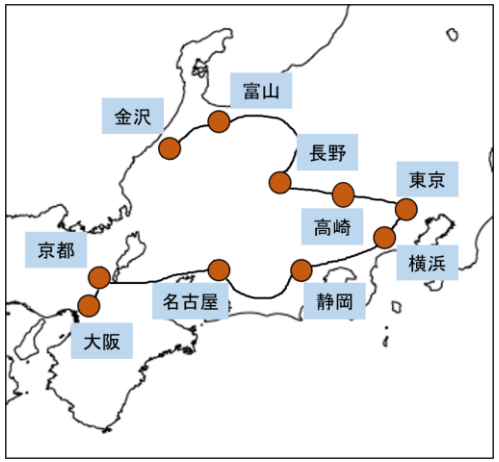
応用例 (流れだけ見てみる)





1. グラフに落とし込む
2. 実は最小全域木問題を解くと解ける.
3. 元に戻す





今回の目標

- グラフの用語を理解する.
- 深さ優先探索を使ってグラフの連結成分の数を数える.
- 幅優先探索を使って迷路の最短経路を求める.

グラフとは

グラフ G は

- 頂点集合

$$V = \{v_1, v_2, \cdots, v_n\}$$

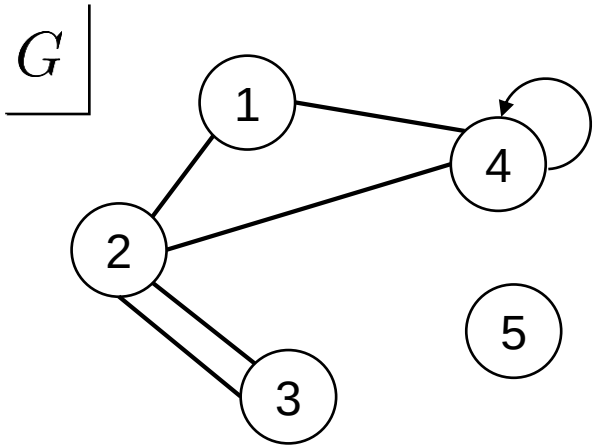
- 辺集合

$$E = \{e_1, e_2, \cdots, e_m\}$$

の組として定義され, $G = (V, E)$ と表される.

辺 e はある v_i, v_j が存在して $e = (v_i, v_j)$ と表される.

例



この場合,

$$V = \{1, 2, 3, 4, 5\}$$

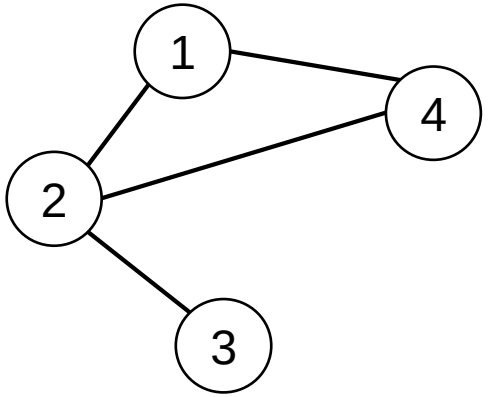
$$E = \{(1, 2), (2, 3), (2, 3), (2, 4), (1, 4), (4, 4)\}$$

また (2, 3), (2, 3) のようなものを多重辺, (4, 4)のようなものを自己ループという.

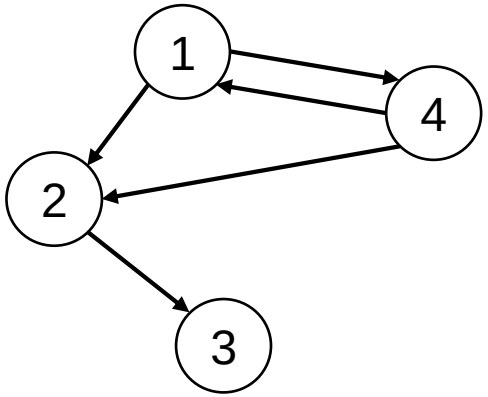
グラフの種類

- 無向グラフ：辺に向きがないグラフ
- 有向グラフ：辺に向きがあるグラフ
- 重み付きグラフ：辺に重みがあるグラフ

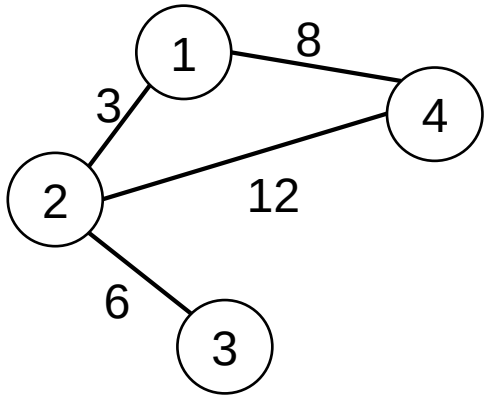
  重みの例：距離, 時間, 価格など



無向グラフ

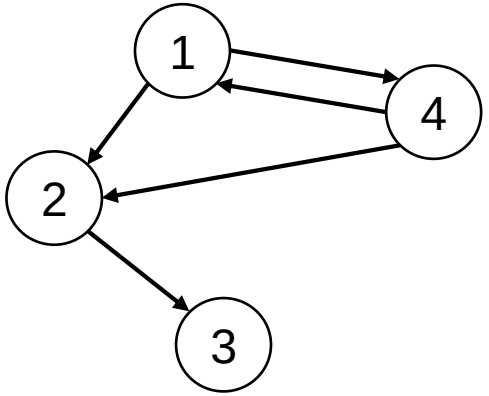


有向グラフ



グラフの用語

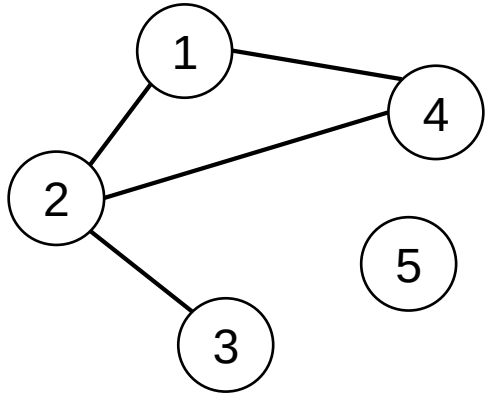
- 路：グラフ G 上の頂点 u, v について, u から v へと
  隣接する頂点をたどって到達できるとき, その経路を
  $u - v$ 路という. (辺の重複は不可)
  u を始点, v を終点と呼ぶ.



- パス：同じ頂点を二度通らない路.

- 閉路(サイクル)：路のうち始点と終点が等しいもの.

- 連結：G の任意の2頂点 $u, v \in V$ に対して,
  $u - v$ パスか $v - u$ パスが存在する時.



グラフの活用

- SNS…
  ツイッターのフォロー, フォロワーなどを分析して,
  - おすすめのユーザーを表示する.
  - コミュニティを検出する.
  - 影響力の強い人を検出する.

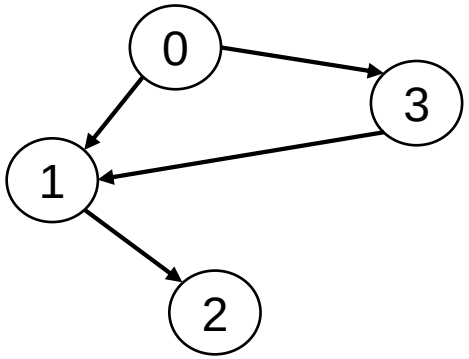
- 交通ネットワーク

- ゲームの局面tree

などあげるとキリがない…

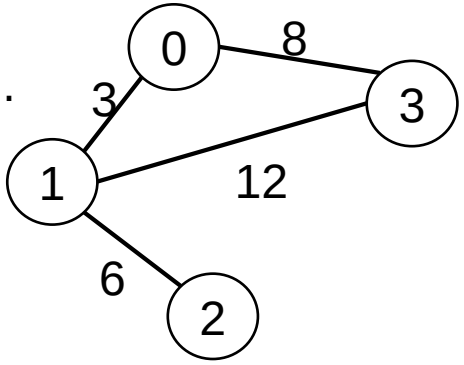
データの持ち方

--- コンピュータはグラフを画像で認識できない.



	0	1	2	3
0	0	1	0	1
1	0	0	1	1
2	0	0	0	0
3	0	0	0	0

- 隣接行列
  - g[i][j] = i番目とj番目の関係
  - 2頂点間に辺があるかどうかを$O(1)$で判定ができる.
  - メモリが $O(|V|^2)$ 必要で$|V| = 10^6$ とかでかなり厳しい.
  - 多重辺の管理が難しい.

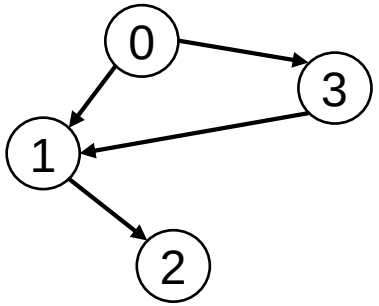


	0	1	2	3
0	0	3	INF	8
1	3	0	6	12
2	INF	6	0	INF
3	8	12	INF	0

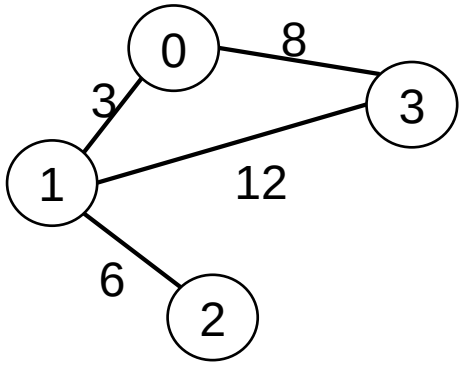
データの持ち方

--- コンピュータはグラフを画像で認識できない.

- 隣接リスト
  - 頂点v に隣接する頂点をリスト(場合によってはset)で管理する.
  - メモリが $O(|V| + |E|)$ でグラフが十分に疎のとき,メモリ消費が少ない.
  - こっちが主流 ?(要検証)



頂点	隣接する頂点リスト
0	1, 3
1	2
2	
3	1



頂点	隣接する(頂点,重み)リスト
0	(1, 3), (3, 8)
1	(2, 6), (0, 3), (3, 12)
2	(1, 6)
3	(1,12), (0, 8)

グラフの連結成分の数を数える

N 頂点, M 辺の多重辺と自己閉路のない無向グラフ G が与えられる.
i 番目の辺 $(1 \leq i \leq b)$ は a_i と b_i を結ぶ.
G の連結成分の数を求めよ.
入力は以下の形式で標準入力で与えられる.

N M

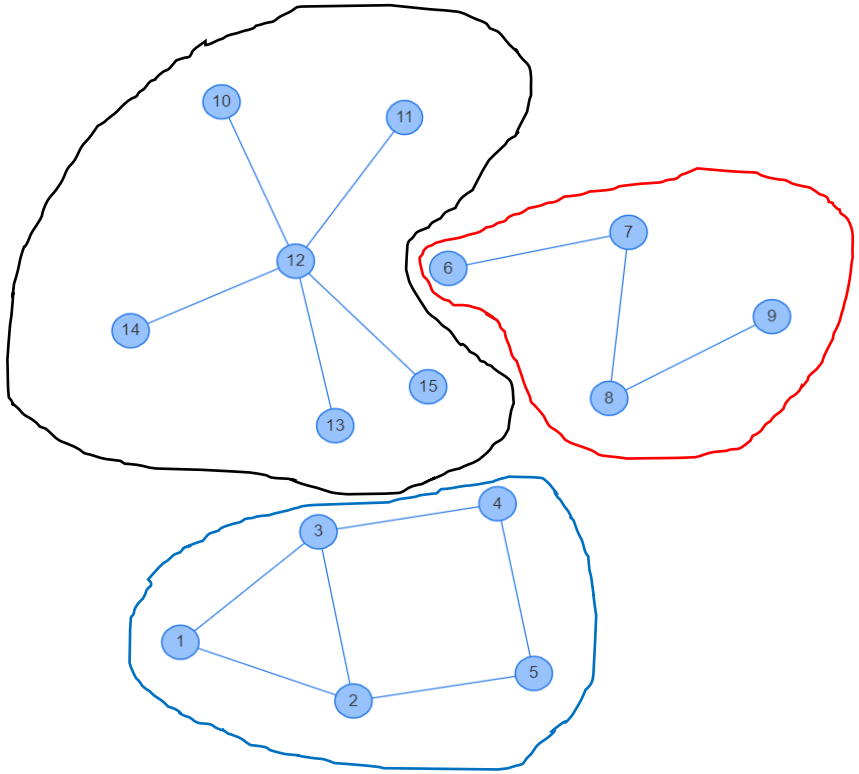
a_1 b_1

a_2 b_2

$\vdots$

a_n b_n

グラフの連結成分の数を数える



- 連結成分とは…

  任意の2頂点間にパスが存在するような

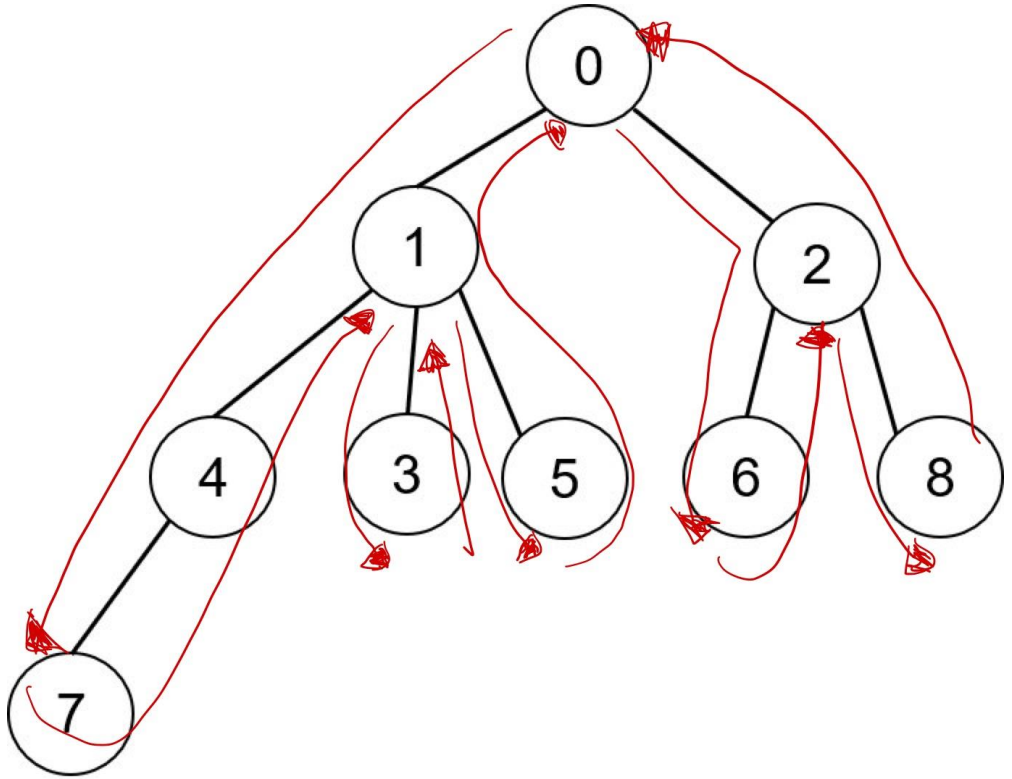
  部分グラフのうち極大なもの

～アルゴリズム～

1. 「未探索」の頂点について以下を行う .
   - 同じ連結成分に含まれる頂点を全て「探索済み」にする

2. 1を行った回数が連結成分の数

深さ優先探索



- 取りあえずいけるところまで行ってみて，
  いけるところが無くなったら引き返す．

- こうすることにより（無向グラフの）連結成分はすべて見ることができる．

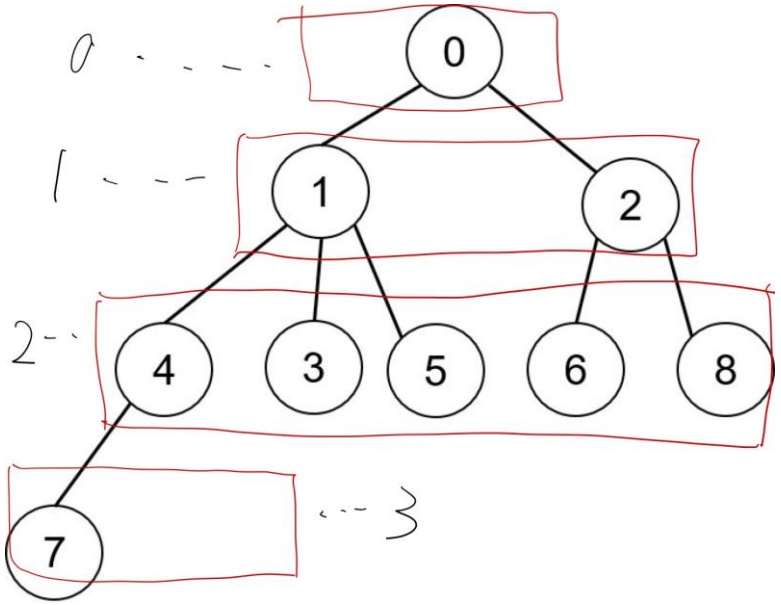
- 実装してみる．

迷路を解く

N 頂点, M 辺の多重辺と自己閉路のない無向グラフ G が与えられる.
i 番目の辺 $(1 \leq i \leq b)$ は a_i と b_i を結ぶ.
頂点 s と頂点 t の距離を求めよ.
ただし, 辺の重みはすべて 1 である.
入力は以下の形式で標準入力で与えられる.

N M
s t
a_1 b_1
a_2 b_2
$\vdots$
a_n b_n

迷路を解く



- 数学的帰納法で考える.

0. 頂点 s からの距離のリスト d を
   $d = [\infty, \infty, \cdots, \infty]$
   で初期化する.

1. $d[s] = 0$ である.

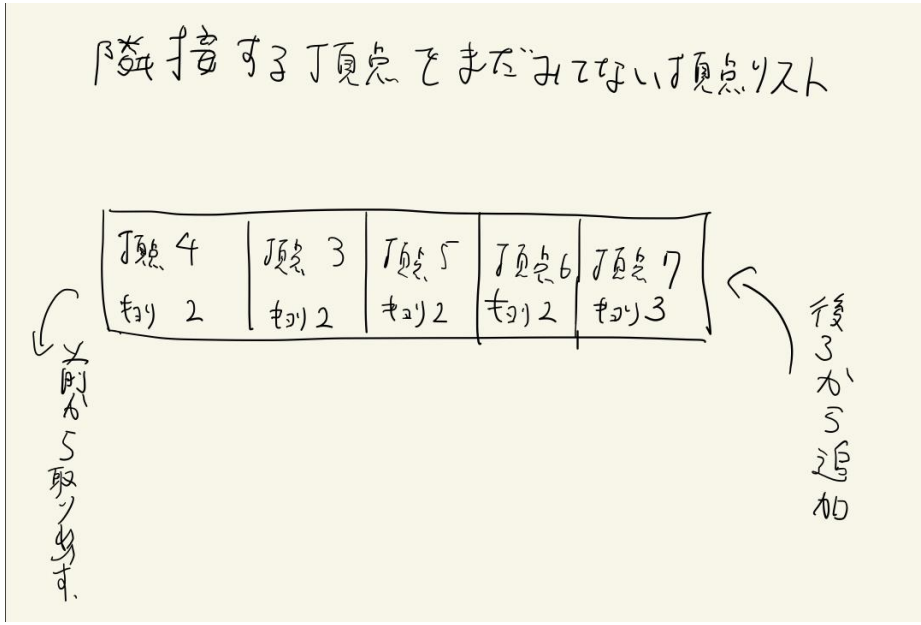
2. 頂点 s からの距離が k までの頂点が決まっているとする.

   まだ頂点 s からの距離が定まっていなくて,
   距離 k に隣接する頂点は頂点 s からの距離は$k+1$である.

- このような探索を幅優先探索という.

実装上の注意

- list型のpop(0)は $O\left(len(List)\right)$ かかる.
- pop(0)とappend()が $O(1)$ でできるデータ構造が欲しい.
- 多くの言語にはqueやdequeがある.
- 経路復元は今回は省略する.
- 実装してみる.



まとめ

- グラフ理論の用語や表現方法を見てきた.
- 深さ優先探索や幅優先探索探索といった全探索の手法を見てきた.
- グラフは形状によって性質がガラッと変わるので全探索は有効.
- 一見グラフに関係なさそうなものをグラフに落とし込んで解決できるのが魅力.

質問タイム