

## A-3) 仲間グループに権力者ができるのはなぜか？

---

社会システム科学 (10/26)

# 人間関係の不均衡

---

[仮定]

- ・ 各メンバー間のそれぞれの関係以外はすべて同じである

[問い]

- ・ このような場合にもメンバーの影響力に違いは生まれるのか？
- ・ それはなぜか？

# 社会ネットワーク分析

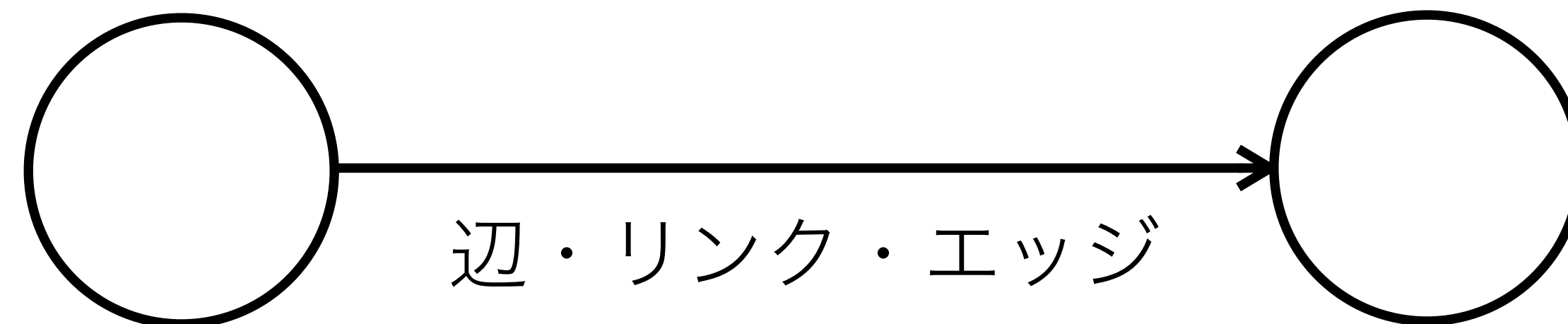
# 社会ネットワーク分析

---

- ・ 集団における要素間の関係（ネットワーク）のグラフによる分析

# ネットワークとグラフ

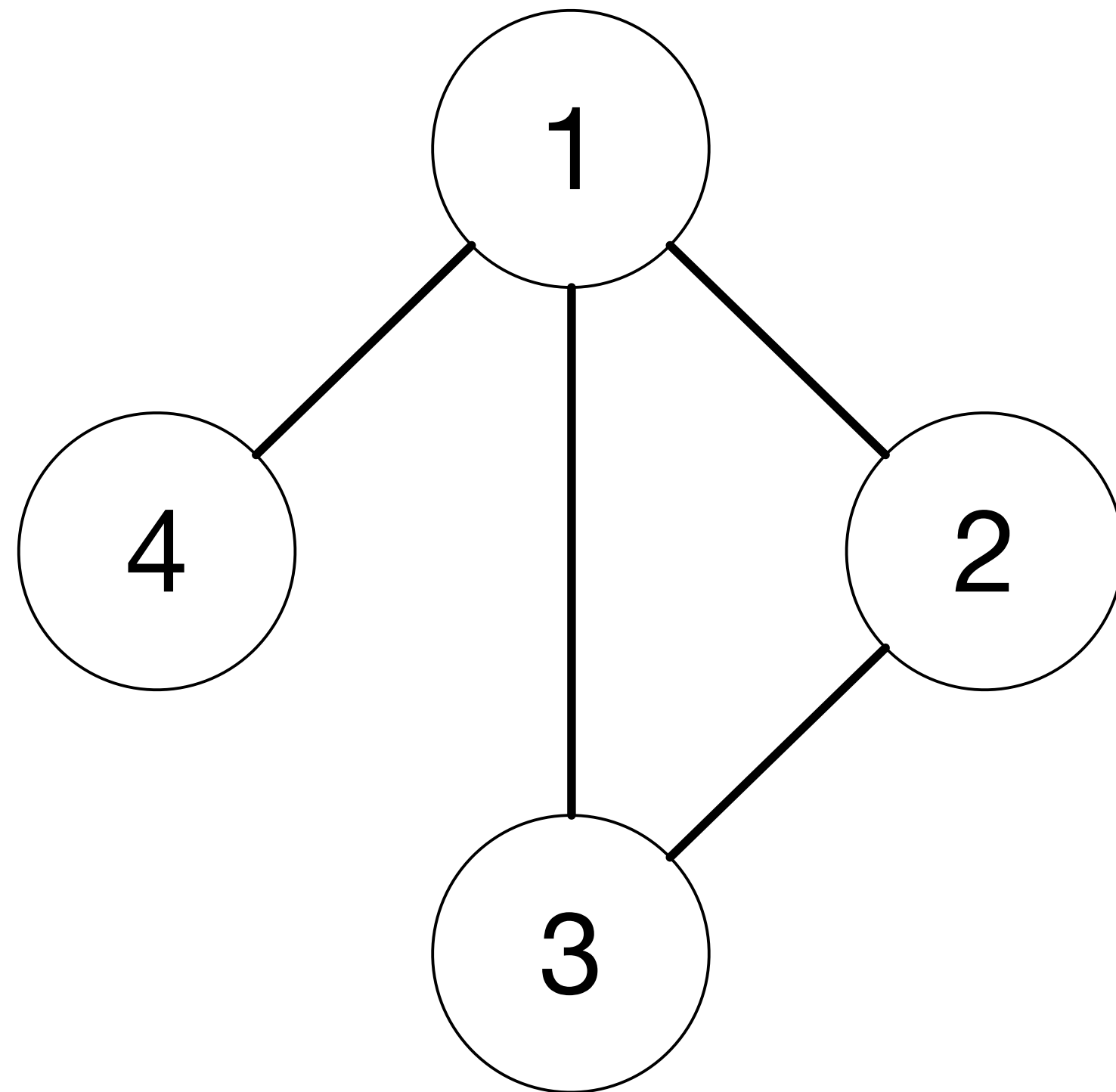
- ネットワーク（関係性） $\Rightarrow$  グラフ（数学モデル）



ノード・頂点 (vertex)

# グラフの表現

- 4ノードの無向グラフ



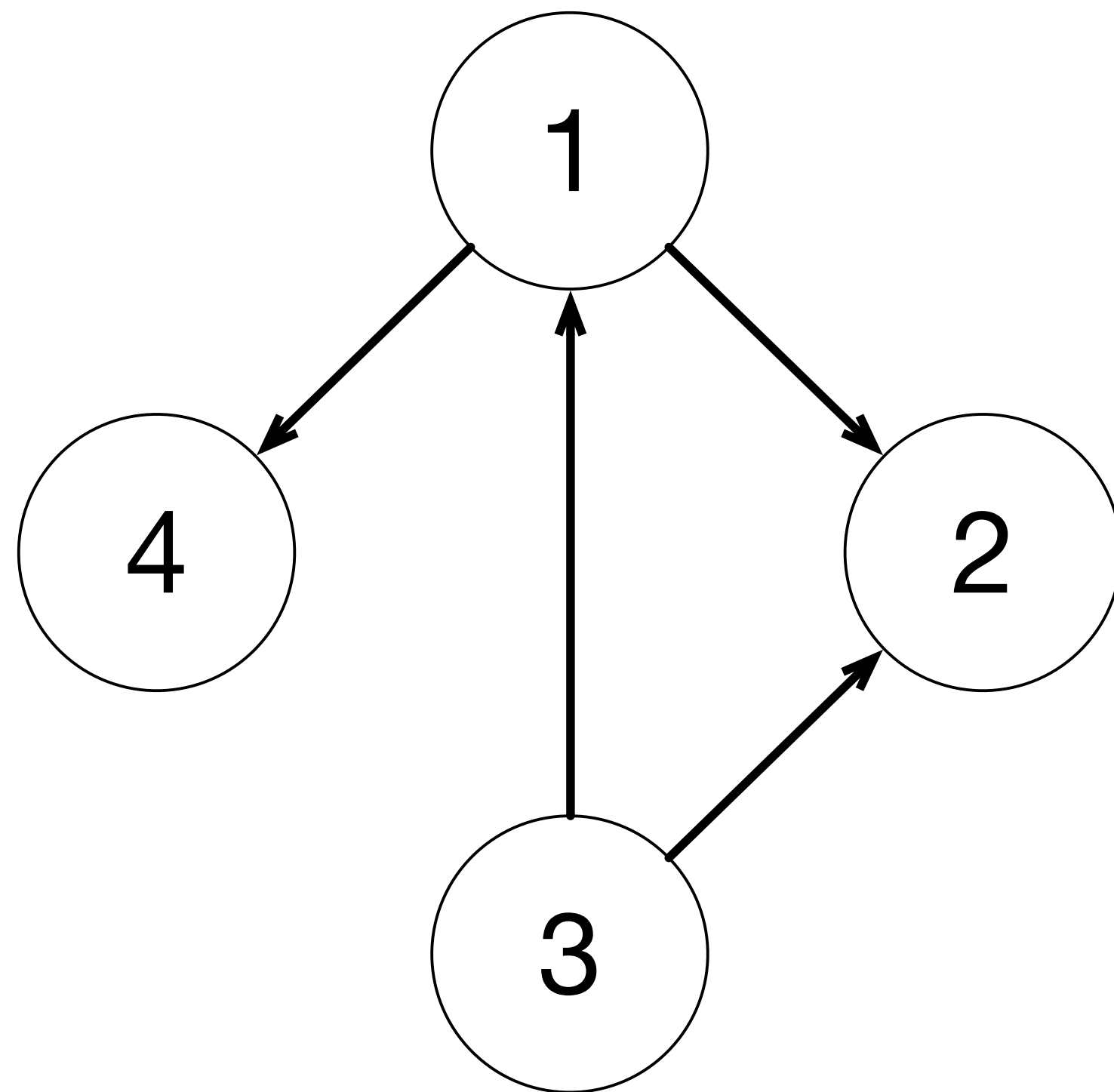
ネットワーク図  
(network diagram)

	1	2	3	4
1	0	1	1	1
2	1	0	1	0
3	1	1	0	0
4	1	0	0	0

隣接行列  
(adjacency matrix)

# グラフの表現

- 4ノードの有向グラフ



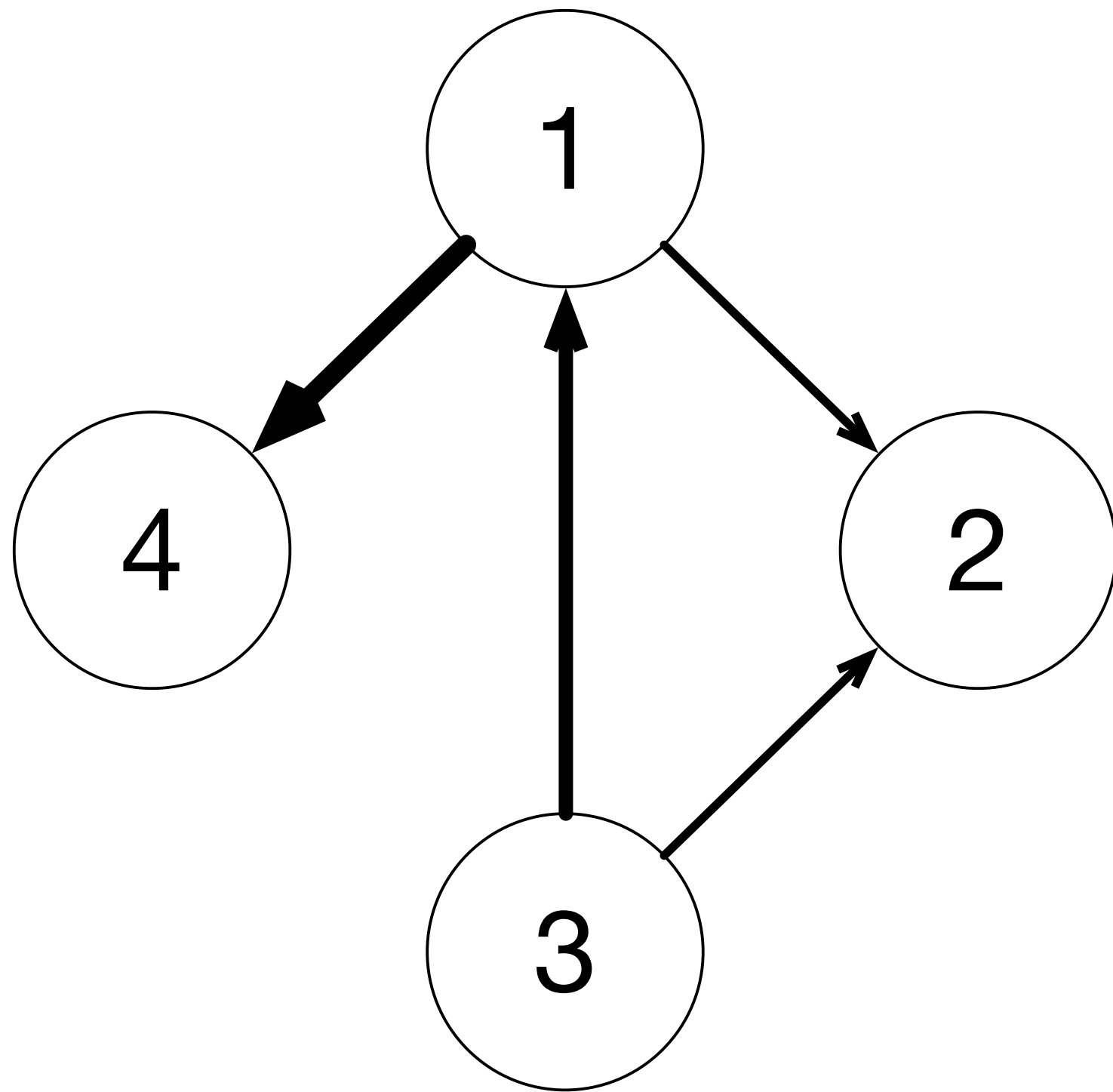
ネットワーク図  
(network diagram)

	1	2	3	4
1	0	1	0	1
2	0	0	0	0
3	1	1	0	0
4	0	0	0	0

隣接行列  
(adjacency matrix)

# グラフの表現

- 4ノードの重み付き有向グラフ



ネットワーク図  
(network diagram)

	1	2	3	4
1	0	1	0	5
2	0	0	0	0
3	3	1	0	0
4	0	0	0	0

隣接行列  
(adjacency matrix)



# グラフの指標

---

- 次数：頂点に接続する辺の数（入次数  $\leftrightarrow$  出次数）
- 長さ：2頂点間を経由する辺の数
- 距離：長さのうち最小のもの
- グラフの直径：グラフの最大頂点間距離

# 中心性：頂点の重要性 (1/2)

---

- 次数中心性：頂点の次数（有向グラフの場合は入次数）
- 固有ベクトル中心性
  - 頂点  $i$  の中心性  $x_i$

$A$ は隣接行列（ $A_{ij}$ は頂点 $i$ と頂点 $j$ の辺の重み）

$$x_i(n+1) = \sum_j A_{ij} x_j(n)$$

- このままだと発散するので正規化（合計を1にする）

$$x_i(n) = \frac{x_i(n)}{\sum_j x_j(n)}$$

- $n \rightarrow \infty$  の極限（ $x_i$ が変化しなくなる状態）における  $x_i$

$$Ax = \lambda x$$

## 中心性 (2/2)

---

- カッツの中心性
  - 次数が 0 のノードにも微小な中心性を与える

$$x_i(n+1) = \alpha \sum_j A_{ij} x_j(n) + \beta$$

- ボナチッチの中心性
  - 各頂点の中心性をその時点の最大中心性で規格化する (最大を1とする)

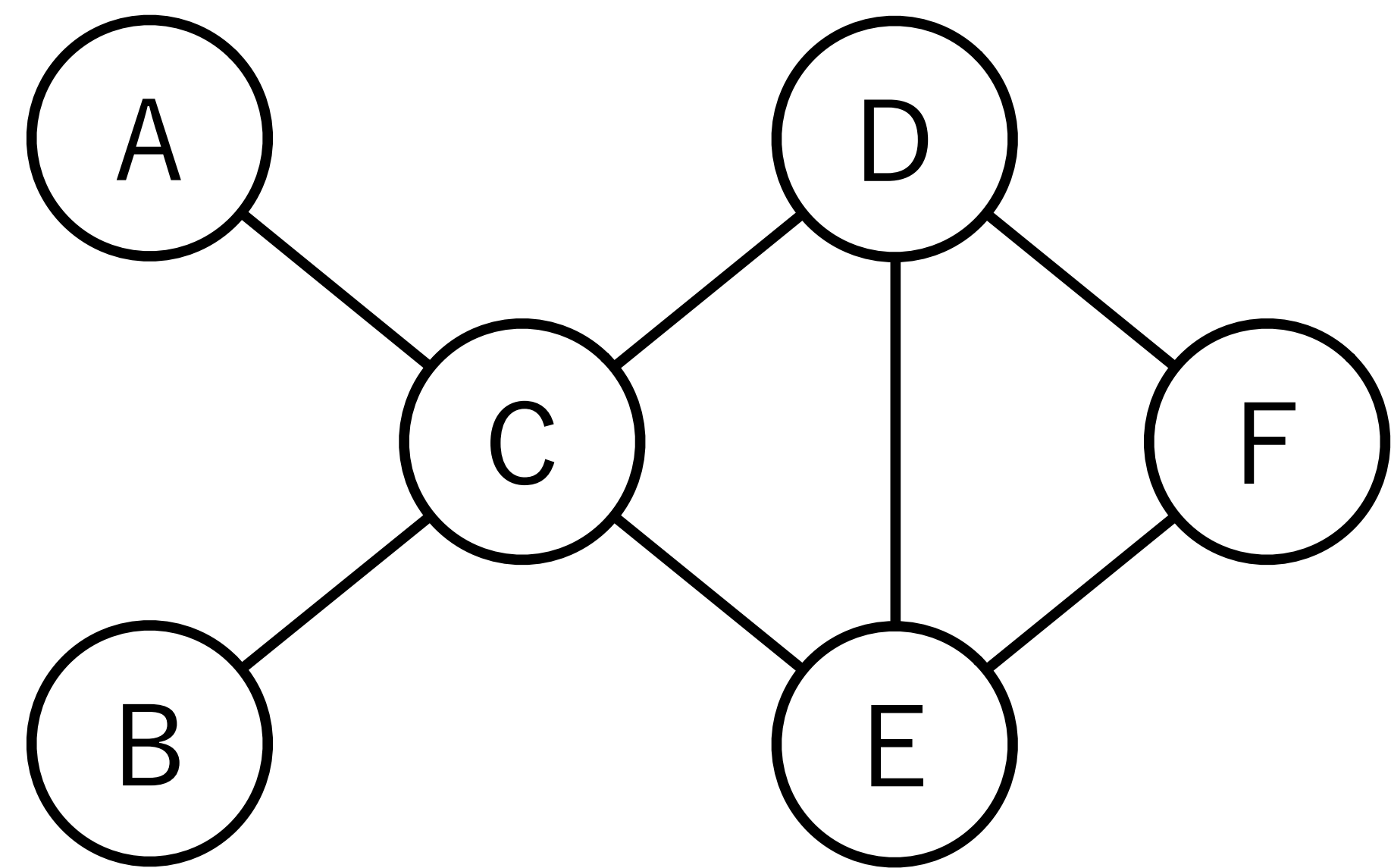
$$x_i(n+1) = \frac{1}{\max_i x_i(n)} \sum_j A_{ij} x_j(n)$$

- PageRank
  - 中心性を出次数で規格化する → 出次数の多い頂点の重みを軽く

$$x_i(n+1) = \sum_j A_{ij} \frac{x_j(n)}{k_j}$$

# Jupyterによる中心性の計算

# 次のネットワークの固有ベクトル中心性を調べる



ネットワーク図

	A	B	C	D	E	F
A	0	0	1	0	0	0
B	0	0	1	0	0	0
C	1	1	0	1	1	0
D	0	0	1	0	1	1
E	0	0	1	1	0	1
F	0	0	0	1	1	0

隣接行列

# 1. 準備

---

- 必要なパッケージの読み込み

```
import numpy as np  
import matplotlib.pyplot as plt
```

## 2. ネットワークの設定

---

- ・ 隣接行列を設定

```
A = np.matrix([
    [0,0,1,0,0,0],
    [0,0,1,0,0,0],
    [1,1,0,1,1,0],
    [0,0,1,0,1,1],
    [0,0,1,1,0,1],
    [0,0,0,1,1,0]
])
```

### 3. シミュレーションの初期設定

---

<code>N = 10</code>	←	何回繰り返すか（回数が多いと収束していく）
<code>x = [0] * (N+1)</code>	←	データの保存場所
<code>x[0] = np.matrix([1,1,1,1,1,1]).T</code>	←	中心性の初期値（みんな1）



## 4. 実験

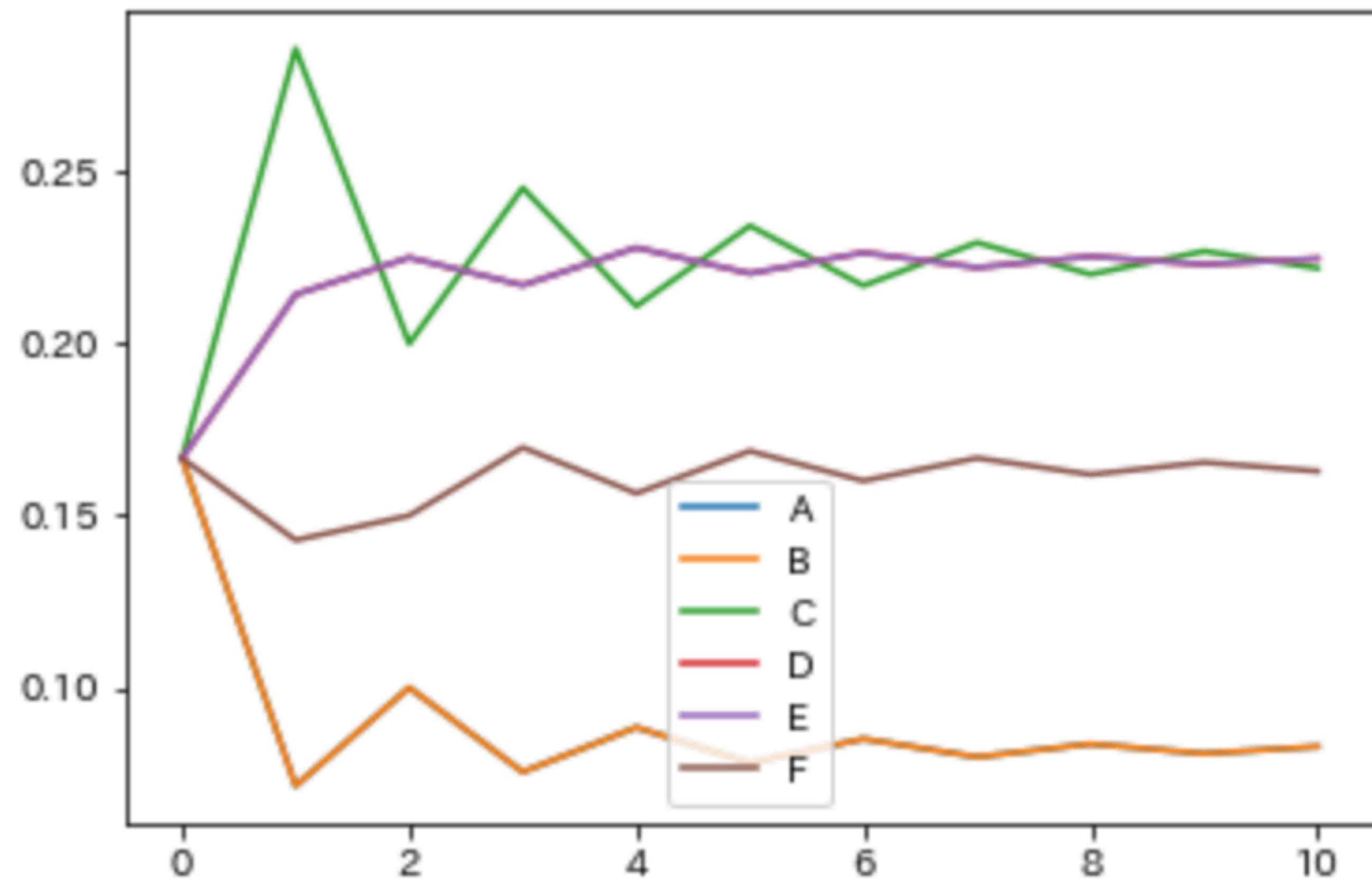
```
x[0] = x[0] / np.sum(x[0])  
for n in range(N):  
    x[n+1] = A * x[n]  
    x[n+1] = x[n+1] / np.sum(x[n+1])
```

正規化（合計で割る）

$x(n+1) = A \cdot x(n)$  の計算

## 5. プロット

```
x = np.array(x)
plt.plot(x.reshape(N+1,6))
plt.legend(['A','B','C','D','E','F'])
```



## 6. 1人分の中心性の推移のプロット

`plt.plot(x[:,3])` ← 何人目 (0から開始) をプロットするか (この例では4人目)

