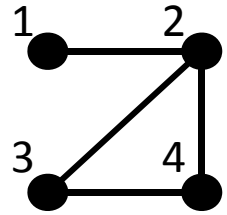
Quiz 4

graph 1



$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$p_i(t) = \sum_j \frac{A_{ij}}{k_j} p_j(t-1)$$

$$\mathbf{p}(t) = \mathbf{A}\mathbf{D}^{-1}\mathbf{p}(t-1)$$

$$\mathbf{D}^{-1} = \begin{pmatrix} 1/k_1 & 0 & 0 & \cdots \\ 0 & 1/k_2 & 0 & \cdots \\ 0 & 0 & 1/k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

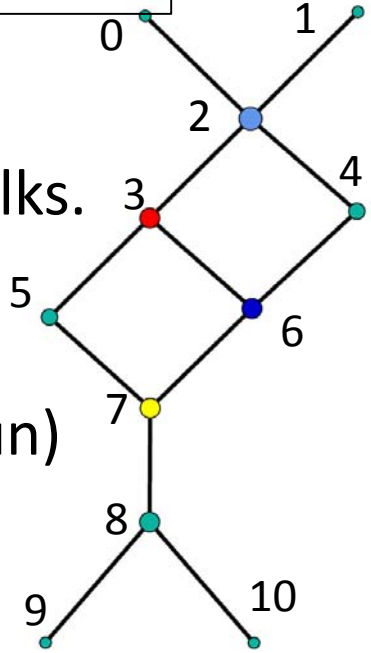
What happens when a walker keeps walking for a long time on a graph?

$p_i(t)$: probability that the walk is at vertex i at time t

1. graph 1: find AD^{-1}
2. graph 1: find $p_0(\infty)$, $p_1(\infty)$, $p_2(\infty)$, and $p_3(\infty)$
3. graph 2: make a program of simulating random walks. show its results and answer the most visited node.

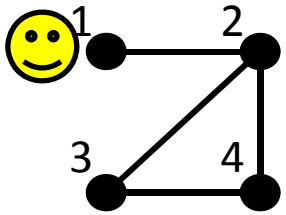
- Submit from Tokyo Tech OCW-i
- Deadline: ??:??(Japan Standard Time) on Dec. 16(Sun)

graph 2



random walk on a graph

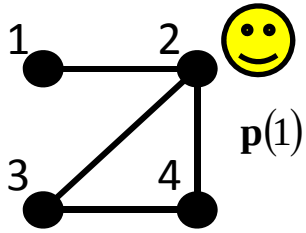
- t=0



$$\mathbf{p}(0)=\begin{pmatrix}1\\0\\0\\0\end{pmatrix} \qquad \begin{aligned} p_1(0)&=1\\ p_2(0)&=0\\ p_3(0)&=0\\ p_4(0)&=0 \end{aligned}$$

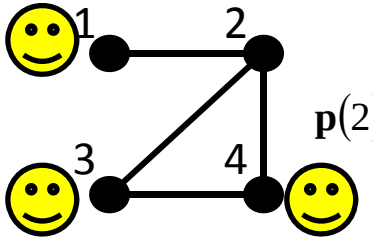
start vertex does not matter

- t=1



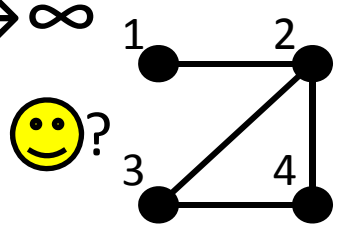
$$\mathbf{p}(1)=\begin{pmatrix}0\\1\\0\\0\end{pmatrix} \qquad \begin{aligned} p_1(1)&=0\\ p_2(1)&=1\\ p_3(1)&=0\\ p_4(1)&=0 \end{aligned}$$

- t=2



$$\mathbf{p}(2)=\begin{pmatrix}1/3\\0\\1/3\\1/3\end{pmatrix} \qquad \begin{aligned} p_1(2)&=1/3\\ p_2(2)&=0\\ p_3(2)&=1/3\\ p_4(2)&=1/3 \end{aligned}$$

- $t\to\infty$

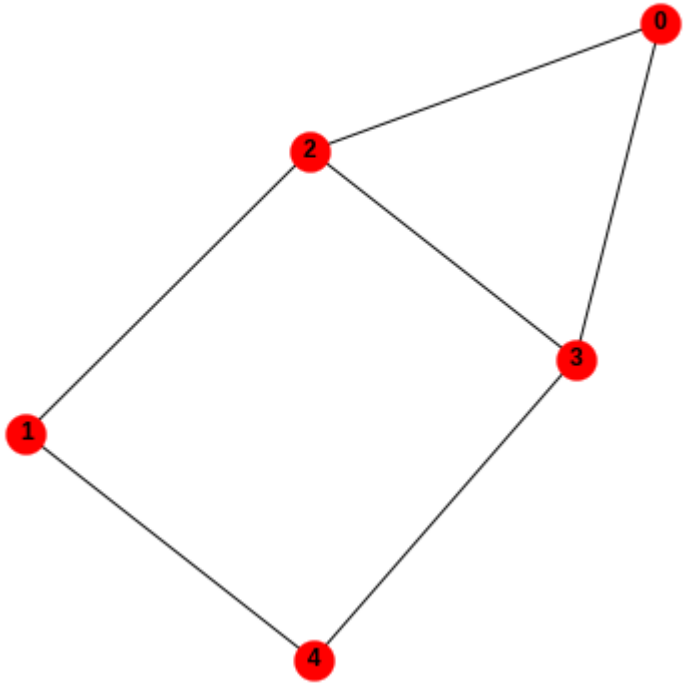


$$\mathbf{p}(\infty)=\begin{pmatrix}?\\?\\?\\?\end{pmatrix} \qquad \begin{aligned} p_1(\infty)&=?\\ p_2(\infty)&=?\\ p_3(\infty)&=?\\ p_4(\infty)&=? \end{aligned}$$

sum should be one

```python
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(5, 5))
G = nx.gnp_random_graph(5, 0.5)
nx.draw_spring(G, node_size=400, node_color='red', with_labels=True, font_weight='bold')
if nx.is_connected(G):
  A = nx.adjacency_matrix(G).todense()
  A = np.array(A, dtype = np.float64)
  c = np.array([0, 0, 0, 0, 0]) # counter
  walkLength = 100000 # random walk length
  id = 0 # starting node (0-th)
  visited = [id]
  c[id] = c[id] + 1
  for k in range(walkLength):
    p = A[id,:]
    id = np.random.choice(np.flatnonzero(p == p.max())) # random choice
    visited.append(id)
    c[id] = c[id] + 1
  print(c)
else:
  print("not connected ")
```

```
[16687 16759 24938 24920 16697]
```




```
not connected
```