Last week

- Definition of tree-decomposition
- Algorithm for Independent Set on Graph with small treewidth,

  NP-Hard → $O(2^{\text{treewidth}} \| V \|)$

- Fixed-Parameter Tractability (FPT)

  Assume that input has some quantity d

  tree-width

  degree (maximum #links associating to a node)

  We want to find an algorithm with time complexity

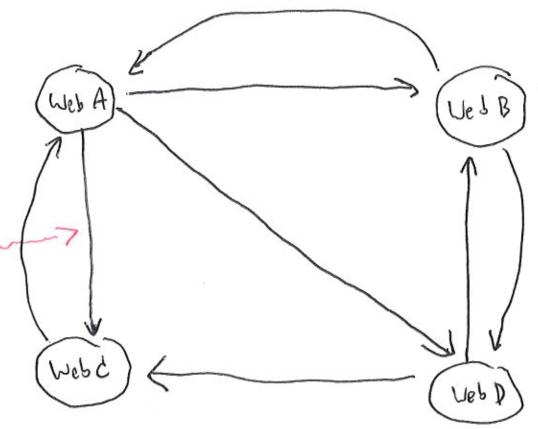
  $f(d)$ [polynomial of input size]

  ↳ f can be arbitrarily large function such as 2^d or d^d.

  Because d is small, $f(d)$ will not be that large.

Web graph

[Leskovec et al 2014, Chapter 5]



We can access web C by clicking a link on web A

Set of nodes = set of webs with a searched keywords

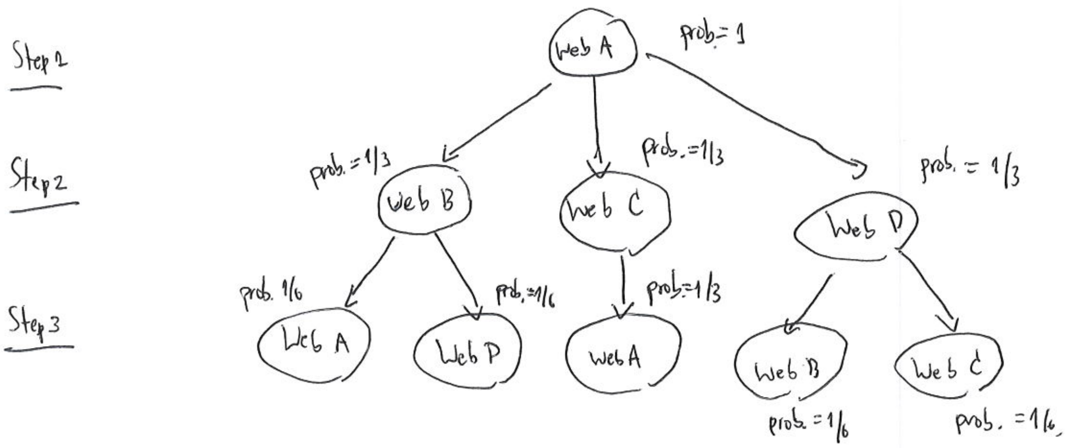
set of links = $\{(u,v)$: v can be accessed by clicking a link at $u\}$

not set but tuple

Page Rank : A way to calculate importance of nodes
webs

- A drunk man is surfing the internet.
- He will click a link on pages randomly with uniform probability.

→ We will calculate the probability of being in each page after infinite steps.



$$\text{Prob}[\text{Web A}] = 1/6 + 1/3 = 1/2$$

$$\Pr[\text{Web B}] = \Pr[\text{Web C}] = \Pr[\text{Web D}] = 1/6$$

Closed form

$$P_{t+1}(A) = P_t(B)/2 + P_t(C)$$

prob. for web A at time t+1

$$P_{t+1}(B) = P_t(A)/3 + P_t(D)/2$$

$$P_{t+1}(C) = P_t(A)/3 + P_t(D)/2$$

$$P_{t+1}(D) = P_t(A)/3 + P_t(B)/2$$

Assume that, at infinite steps, $P_{t+1}(A) = P_t(A)$, $P_{t+1}(B) = P_t(B)$, ...

$$P(A) = P(B)/2 + P(C)$$

$$P(B) = P(A)/3 + P(D)/2$$

$$P(C) = P(A)/3 + P(D)/2$$

$$P(D) = P(A)/3 + P(B)/2$$

$$1. \quad \lambda \begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix} = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix}$$

$$\cdot\, V \qquad\qquad M \cdot V$$

$\therefore \begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix}$ is an eigenvector of M corresponding to eigenvalue $\lambda = 1$

$\therefore \begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix} = \begin{bmatrix} 3/9 \\ 2/9 \\ 2/9 \\ 2/9 \end{bmatrix}$

- Web A has more important than Web B, C, D
- Webs B, C, D have equal importances.

Property of PageRank

- You will have more scores when you have links from a page with large score.

$$P(\text{your web}) = \frac{1}{\#\text{ linked from } A} p(A) + \frac{1}{\#\text{ linked from } B} p(B) + \dots + \frac{1}{\#\text{ linked from } P} p(P)$$

when p(A) is large
P(your web) tends to be large.

$\{A, \dots, P\}$: Set of webs with links to your web

- You will have a larger score if a distinguished person endorses you than when 1,000 unknown persons endorse you.
- PageRank will prevent the case that someone create 1,000 webs that have links to their own pages.

Personalized PageRank

- Google will provide different results for different users

Conventional Page Rank : $p = Mp$

a vector that contains probabilities of each web

Matrix that include the fact that the link is clicked randomly.

Personalized PageRank ; $p = (1-\beta)Mp + \beta q$

personalized vector

- Users sometimes go to an address box and press URLs of the web they want to visit.
- The probability distribution of webs accessed by the address box is q.

<u>Ex</u> $q = [0.1 \; 0.3 \; 0.5 \; 0.2]^t$

Users may like Web C but not Web A

- The probability that users go to the address box is β.

<u>Ex</u> $\beta = 0.5$

$$\begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix} = 0.5 \begin{bmatrix} 0 & 1/2 & 1/4 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} p(A) \\ p(B) \\ p(C) \\ p(D) \end{bmatrix} + 0.5 \begin{bmatrix} 0.1 \\ 0.3 \\ 0.5 \\ 0.2 \end{bmatrix}$$

$$I \cdot p = (1-\beta) \cdot M \cdot p + \beta \cdot q$$

$$I \cdot p = (M - \beta M) \cdot p + \beta q$$

$$I \cdot p - (M - \beta M) \cdot p = \beta q$$

$$(I - M + \beta M) \cdot p = \beta q$$

$$p = (\underbrace{I - M + \beta M}_{})^{-1} \cdot (\beta q)$$

can be solved by inversing this matrix

LU decomposition

$$I - M + \beta M = LU$$

$$LUp = \beta q$$

$$p = U^{-1} L^{-1} (\beta q)$$