ECEN 758
Data Mining and Analysis

Nick Duffield
Department of Electrical & Computer Engineering
Texas A&M University

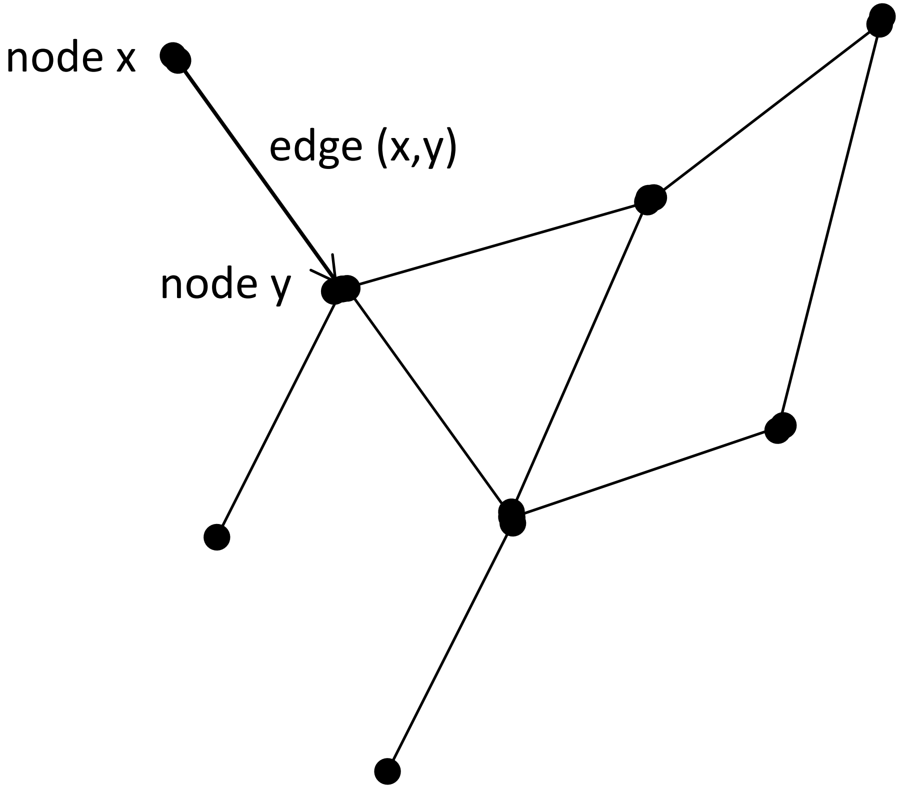
Random Walks on Graphs & Pagerank

TEXAS A&M UNIVERSITY
Department of Electrical
& Computer Engineering

TEXAS A&M
Institute of
Data Science

Graphs

- **Graph G = (V,E)**
  - V = vertices, a.k.a. nodes
  - E = edges, a.k.a. links
- **Edge = node pair**
  - Edge (x,y) joins nodes x, y
- **Undirected edge (x,y)**
  - No order between x and y
- **Directed edge (x,y)**
  - x and y ordered
  - Initial node x
  - Terminal node y

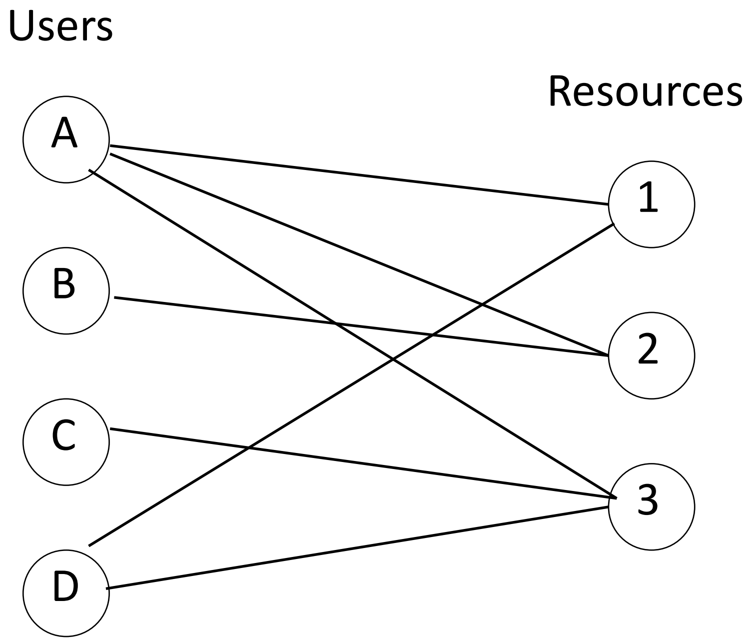


Graphs in the Internet

- **Web graph**
  - Vertex = web page
  - Directed edge (x,y) = hyperlink on page x referring to page y
- **Citation graph**
  - Vertex = publication
  - Edge (x,y) if x cites y
- **Social networks**
  - Vertex = user; edge = social relationship
  - Twitter: directed edge (x.y) when x follows y
  - Facebook: undirected edge (x,y) when x and y are friends
- **IP Communications graph**
  - Vertices = IP addresses
  - Edges: directed edge (x,y) if packets observed with SrcIP x and DstIP y

Resource Graphs e.g. Online retailers

- **Bipartite graph**
  - Two types of node: users and resources
  - Edges only between nodes of different types
- **Edge (x,y) between user x and resource y**
  - User x has {used, purchased, reviewed} resource y



Internet Graph Size

- **These graphs can be enormous**
- **Webgraph:**
  - Trillions of pages indexed
- **Social networks:**
  - Billions of users, trillions of social relationship
- **Communications graphs:**
  - Trillions of flows daily
- **…and growing**

Attributes of Graphical Objects

- **Topological attributes**
  - Example: Node degree: n_x = number of edges containing x
  - Directed graph
    - In-degree n_x^{in} = # { edges (y,x): terminating at x}
    - Out-degree n_x^{out} = # {edges (x,y} starting at x }
  - Example: Shortest path length between nodes x, y
- **Non-topological attributes**
  - Node annotations:
    - Communication graph: node IP address
    - Web graph: page content or abstract thereof
    - Resource graph:
      - User information (physical address, demographics)
      - Resource information (price, review statistics)

Graph ranking queries

- **Rank order nodes for a particular query**
  - Find top k web pages about a topic
  - Top k friend recommendations when joining social network
- **Using both topological and non-topological attributes**
  - Topological component to rank
    - Relevance as measured by (in) degree
  - Semantic relevance
    - Does page content match query topic?
  - Overlap: relevance also inferred from content of related

Definitions

- **Graph with n nodes**

- **nxn Adjacency matrix A.**
  - A(i,j) = weight on edge from i to j, e.g. 1
  - If the graph is undirected A(i,j)=A(j,i)
    - A is symmetric

- **nxn Transition matrix P.**
  - P(i,j) = probability of transition from node i to node j

    $= A(i,j)/\sum_j A(i,j)$
  - P is row stochastic: $\sum_j P(i,j) = 1$

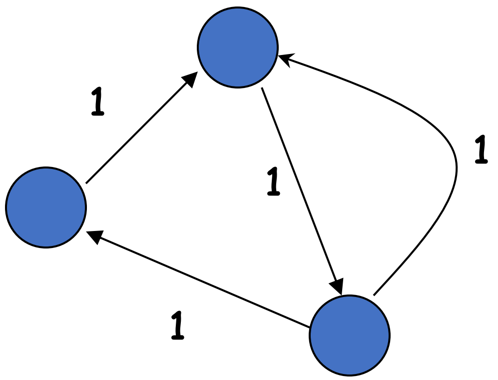
Definitions

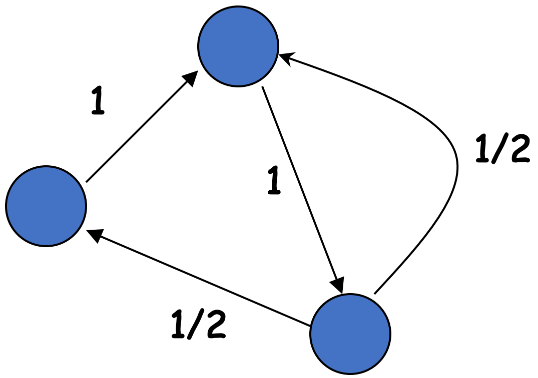
0	1	0
0	0	1
1	1	0

Adjacency matrix A

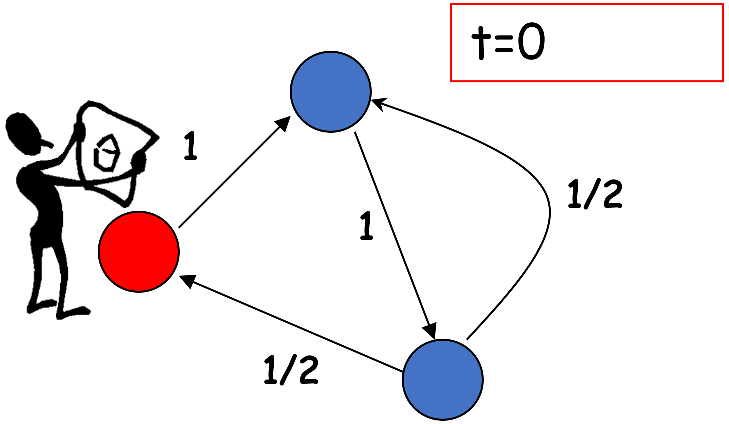
0	1	0
0	0	1
1/2	1/2	0

Transition matrix P

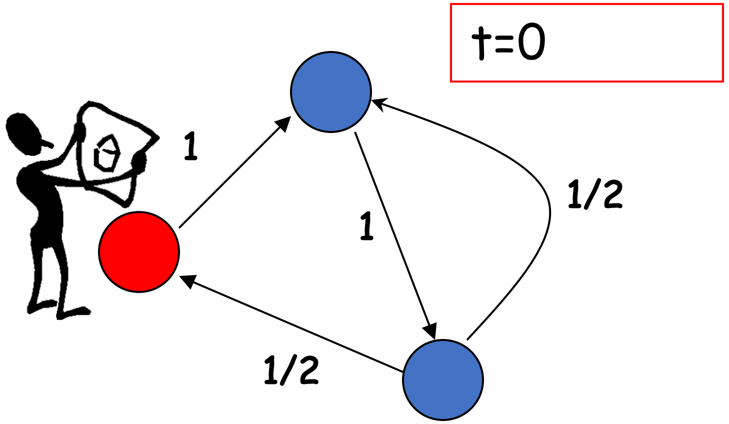


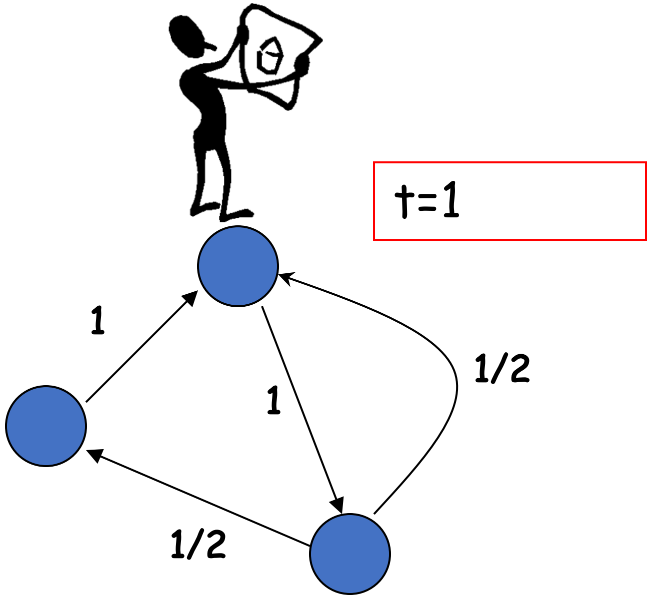


What is a random walk

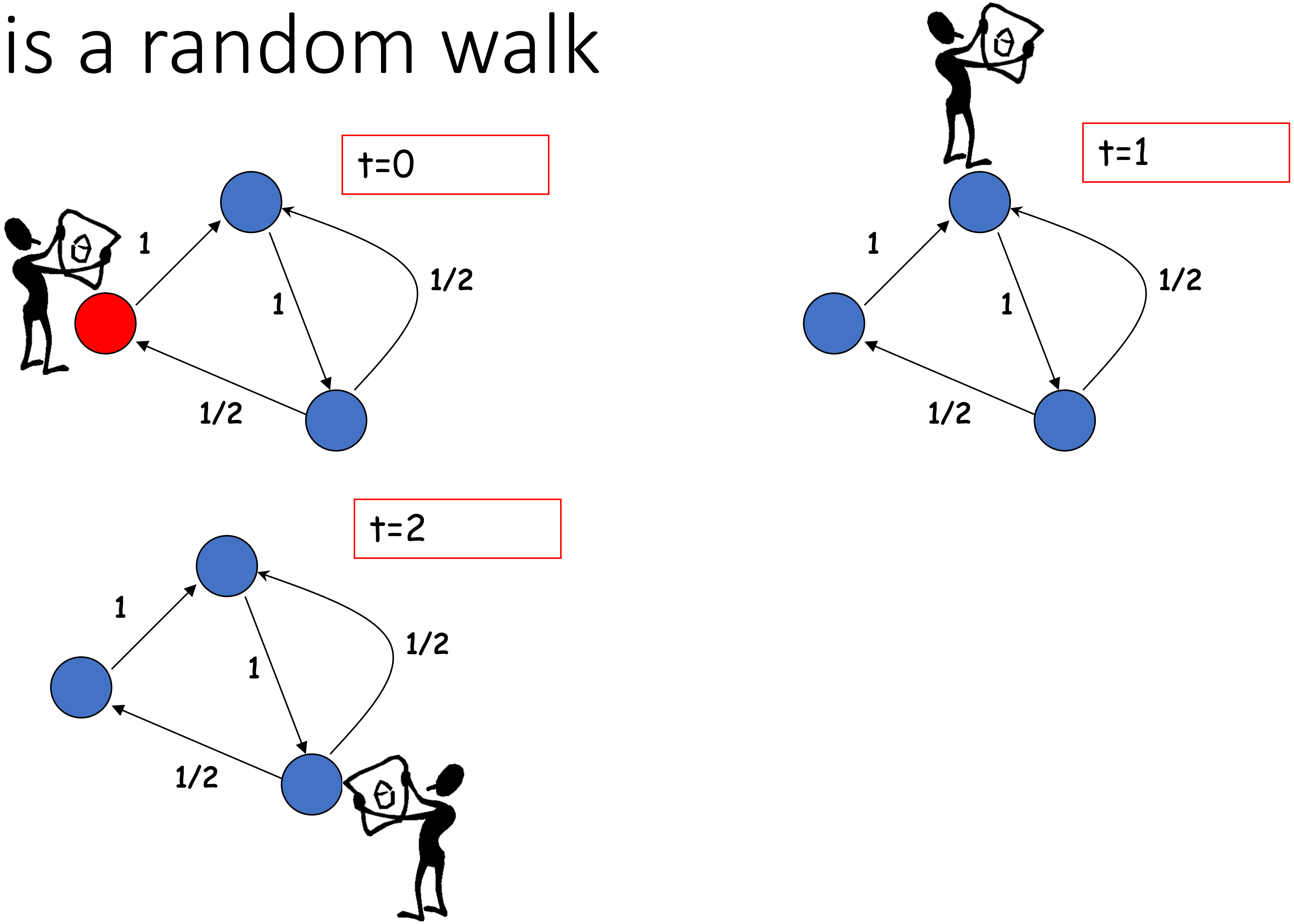


What is a random walk

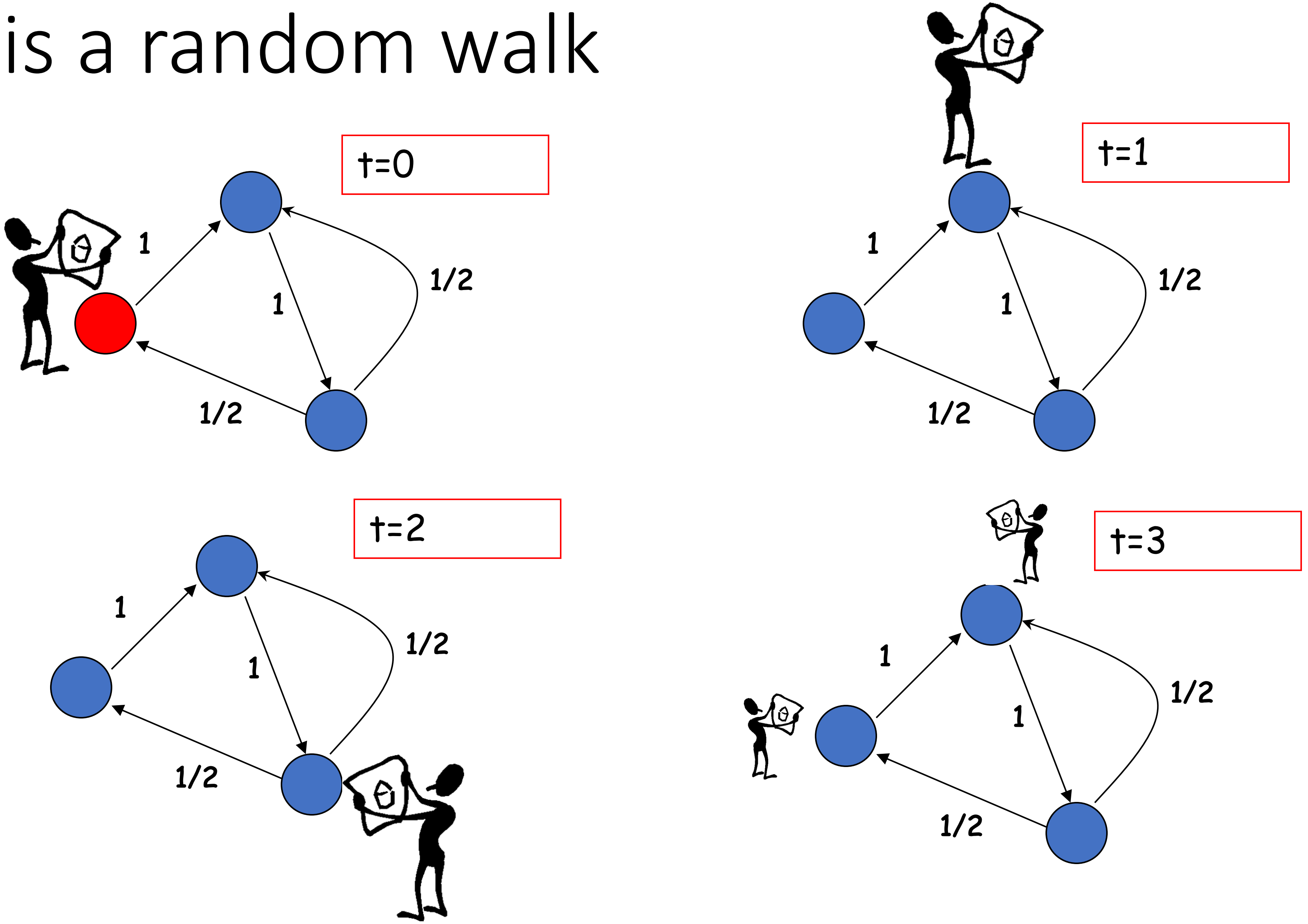




What is a random walk



What is a random walk



Probability Distributions

- $x_t(i)$ = **probability that the surfer is at node *i* at time *t***
- $x_{t+1}(i) = \sum_j$**(Probability of being at node j)*Pr(j->i)** $= \sum_j x_t(j) * P(j,i)$
- $x_{t+1} = x_t P = x_{t-1} * P * P = x_{t-2} * P * P * P = \ldots = x_0 P^{t+1}$
- **What happens when the walker keeps walking for a long time?**

Stationary Distribution

- **When the walker keeps walking for a long time**

- **When the distribution does not change anymore**
  - i.e. $x_{T+1} = x_T$

- **For "well-behaved" graphs this does not depend on the start distribution!!**

What is a stationary distribution?

- **The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.**

- **Remember that we can write the probability distribution at a node as**
  - $x_{t+1} = x_t P$

- **For the stationary distribution *v^0* we have**
  - $v^0 = v^0 P$

- **v^0 is left eigenvector of the transition matrix with eigenvalue 1**

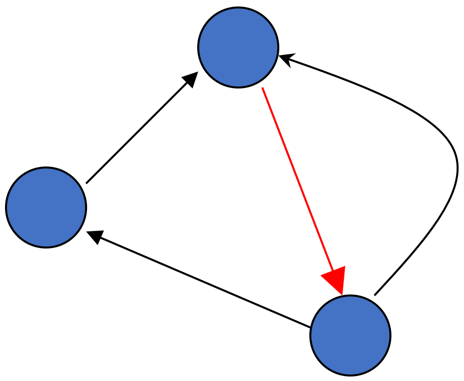
Questions

- **Does a stationary distribution always exist? Is it unique?**
  - Yes, if the graph is “well-behaved”.

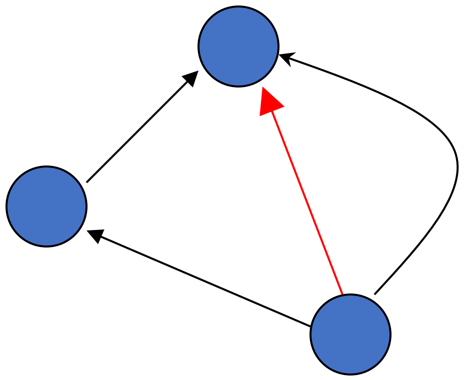
- **How fast will the random walker approach this stationary distribution?**
  - Mixing Time

Well-behaved graphs

- **Irreducible: There is a path from every node to every other node.**



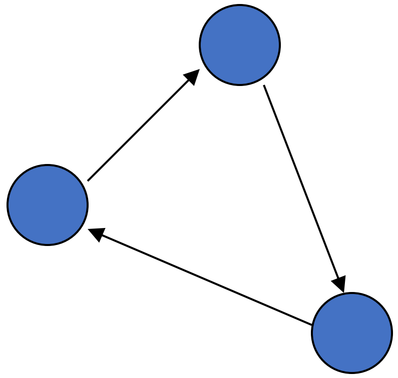
Irreducible



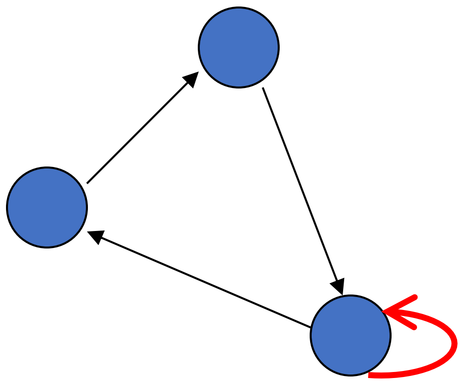
Not irreducible

Well behaved graphs

- **Aperiodic: The Greatest Common Divisor (GCD) of all cycle lengths is *1.* This GCD is also called period.**



Periodicity is 3



Aperiodic

Perron-Frobenius Theorem

- **Assume P: a real square matrix with strictly positive entries**
- **PF Theorem:**
  - P has eigenvalue λ of maximal | λ | such that
  - λ is real and positive
  - λ is simple
    - only one linearly independent eigenvector v with eigenvalue λ
  - The eigenvector v has strictly positive entries
- **If P has nonnegative entries, λ may not be unique**
  - Example: nxn identity matrix I(n)
  - has n linear independent eigenvectors of eigenvalue 1

Implications of Perron-Frobenius Theorem

- **Suppose random walk is irreducible and aperiodic.**
- **The PF eigenvalue of the transition matrix will be equal to 1 and all the other eigenvalues will have absolute value strictly less than 1.**
  - Let the eigenvalues of P be $\{\sigma_i | i=0:n-1\}$ in non-increasing order of σ_i .
  - $\sigma_0 = 1 > |\sigma_1| > |\sigma_2| >= \ldots\ldots >= |\sigma_n|$
- **These results imply that for a well-behaved graph there exists a unique stationary distribution.**

Test out these ideas on undirected graphs

- **Suppose G an undirected connected graph**
- **Unit edge weights: A(i,j) = A(j,i) = 1 if (i,j) is an edge**
- **Transition probability P(i,j) = 1 / n_i**
  - If at i, then equally likely to transition to any of the n_i neighbors
- **If G is connected, then P irreducible**
  - There is a path connecting any two nodes
- **Suppose shortest path between i and j has length m**
  - Can move from vertex i to j in m time steps: $(P^m)_{ij} > 0$

What is stationary distribution?

- **$P_{ij} = 1/n_i$ if $j \in N(i)$ = set of neighbors if i, 0 otherwise**
- **Stationary distribution v: $v_j = \Sigma_i v_i P_{ij}$**
- **Prove that v_j proportional to n_j:**
- **$\Sigma_i n_i P_{ij} = \Sigma_{i \in N(j)} n_i / n_i = \Sigma_{i \in N(j)} 1 = n_j$**
- **Intuition:**
    - The more neighbors a node has, the more likely it is to be visited.

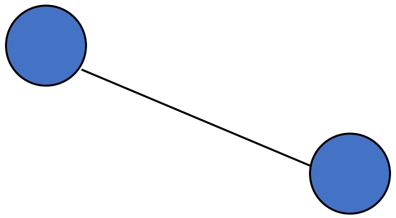
Convergence to stationary distribution

- **Need aperiodic in addition to irreducible**

- **If periodic: $P^m = I$ (the identity matrix) for some m**

- **$uP^m = u$ for any initial u,**
  - uP^n does not converge to stationary distribution v^o as $n \rightarrow \infty$

- **Simple example: period = 2**



$$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad P^2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Ranking algorithms on the web

- **Pagerank (Page & Brin, 1998)**
- **Directed graph**
  - A webpage is important if other important pages point to it
- **Transition matrix $P_{ij} = 1/n^{out}(i)$ for j in $N^{out}(i)$**
  - $N^{in}(i)$ = set of inbound neighbors to i, $n^{in}(i) = \#N^{in}(i)$
  - $N^{out}(j)$ = set of outbound neighbors from j, $n^{out}(j) = \#N^{out}(j)$
- **Expect stationary distribution to obey:**
  - Using j in $N^{in}(i)$ iff i in $N^{out}(j)$:

$$v(i) = \sum_{j \in N^{in}(i)} \frac{v(j)}{n^{out}(j)}$$

Iterative computation of pagerank

- **Aim:**
  - rank pages i according to stationary distribution v^0
- **Method**
  - Iterative computation
  - Start with initial page rank x_o
  - Divide page rank $x_{0,i}$ at page i over all vertices linked to from i
    - i.e. over all j in $N^{out}(i)$
  - Iterate
- **This is just iteration $x_{t+1} = x_t P$**
- **$x_t \rightarrow$ stationary v^0 under Perron-Frobenius conditions**

Pagerank & Perron-Frobenius

- **Perron Frobenius needs graph irreducible & aperiodic.**

- **But how can we guarantee that for the web graph?**
  - Do it with a small restart probability *c*.

- **At any time-step the random walker**
  - jumps (teleport) to any other node with probability ***c***
  - jumps to its direct neighbors with total probability ***1-c***.

$$\tilde{\mathbf{P}} = (1-c)\mathbf{P} + c\mathbf{U} \quad \text{for} \quad \mathbf{U}_{ij} = \frac{1}{n} \forall i, j$$

- **Effectively:**
  - Introduce fully connected graph with link weights c

Power iteration

- **Power Iteration is an algorithm for computing the stationary distribution.**
- **Start with any distribution x_0**
- **Keep computing $x_{t+1} = x_t P$**
- **Stop when x_{t+1} and x_t are almost the same.**

Power iteration

- **Why should this work?**

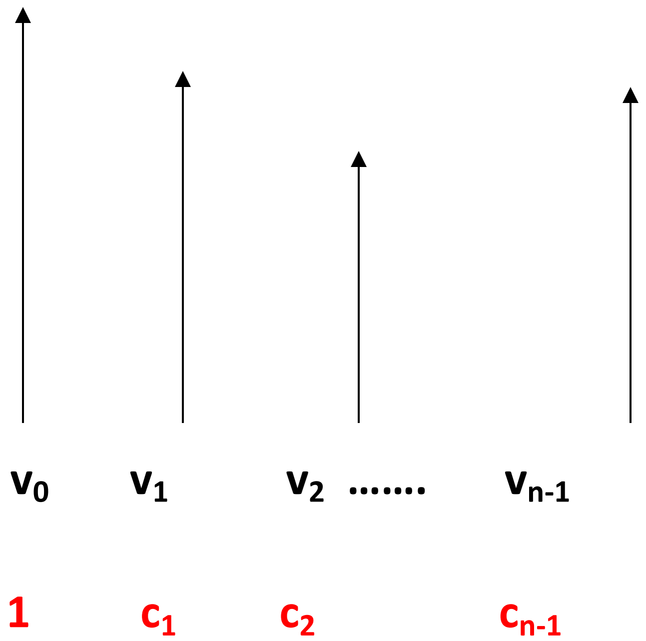
- **Write x_0 as a linear combination of the left eigenvectors $\{v_0, v_1, \ldots, v_{n-1}\}$ of P**

- **Remember that v_0 is the stationary distribution.**

- $x_0 = v_0 + c_1 v_1 + c_2 v_2 + \ldots + c_{n-1} v_{n-1}$

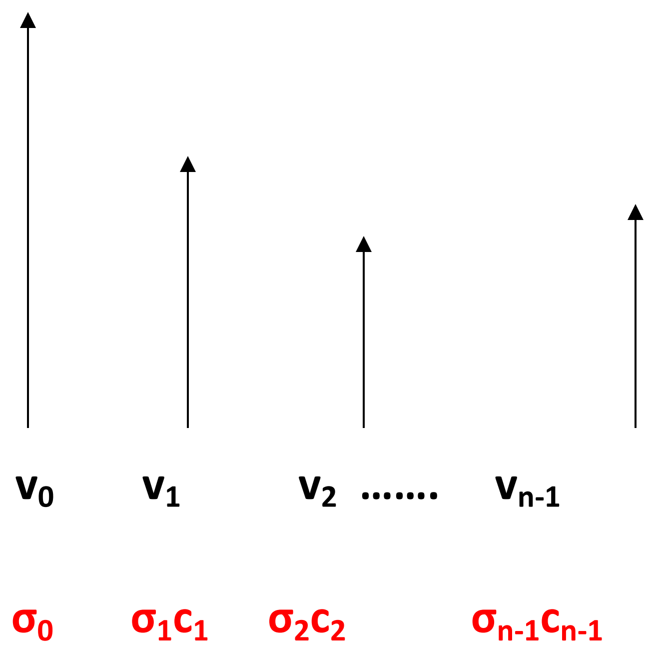
Power iteration

$\mathbf{x}_0$



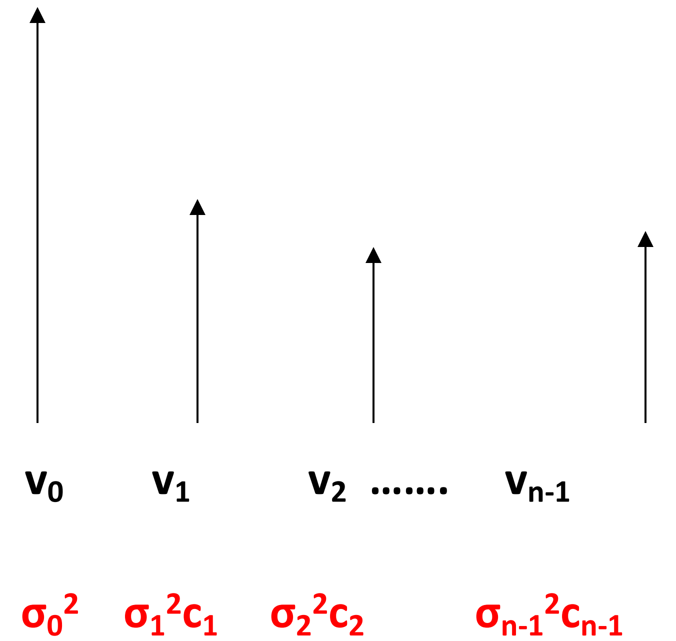
Power iteration

$$\mathbf{x}_1 = \mathbf{x}_0 \tilde{\mathbf{P}}$$



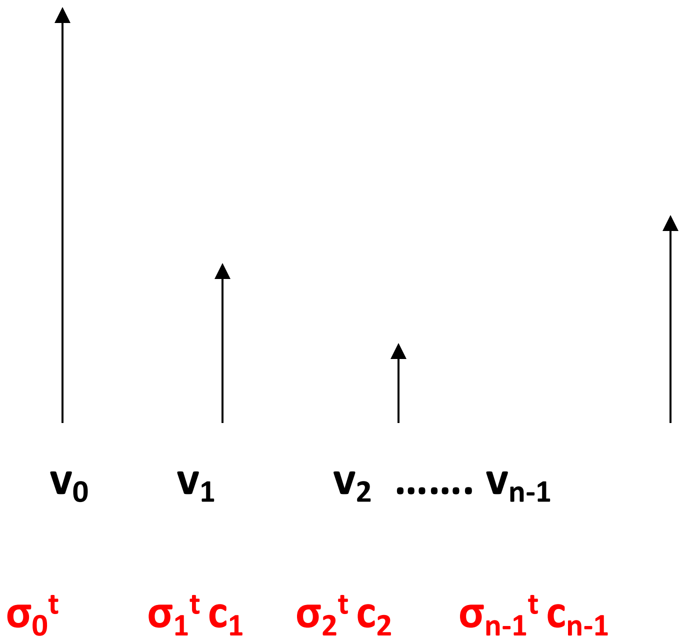
Power iteration

$$\mathbf{x}_2 = \mathbf{x}_1 \tilde{\mathbf{P}} = \mathbf{x}_0 \tilde{\mathbf{P}}^2$$



Power iteration

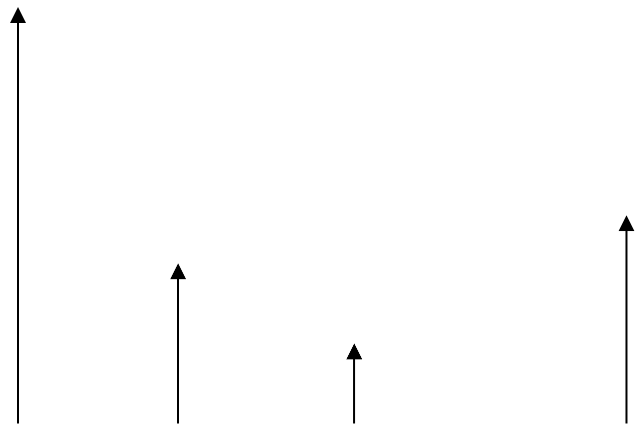
$$\mathbf{x}_t = \mathbf{x}_0 \tilde{\mathbf{P}}^t$$



Power iteration

$$\mathbf{x}_t = \mathbf{x}_0 \tilde{\mathbf{P}}^{t}$$

$\boxed{\sigma_0 = 1 > \sigma_1 \geq \ldots \geq \sigma_n}$



$\mathbf{v}_0$ $\mathbf{v}_1$ $\mathbf{v}_2$ ……. $\mathbf{v}_{n-1}$

$\mathbf{1}$ $\sigma_1^t c_1$ $\sigma_2^t c_2$ $\sigma_{n-1}^t c_{n-1}$

Power iteration

$\mathbf{x}_{\infty}$

$\boxed{\sigma_0 = 1 > \sigma_1 \geq \ldots \geq \sigma_n}$

$\mathbf{v}_0$ $\mathbf{v}_1$ $\mathbf{v}_2$ .... $\mathbf{v}_{n-1}$

1 0 0 0

Convergence Analysis

- $x_0 = v_0 + c_1v_1 + c_2v_2 + \ldots + c_{n-1}v_{n-1}$
- x_t $= x_0P^t$
  $= (v_0 + c_1v_1 + c_2v_2 + \ldots + c_{n-1}v_{n-1})P^t$
  $= v_0 + c_1v_1\sigma_1{}^t + c_2v_2\,\sigma_2{}^t + \ldots + c_{n-1}v_{n-1}\sigma_{n-1}{}^t$
- $||x_0P^t - v_0|| \leq a\,|\lambda|^t$
  - $\lambda = \sigma_1$ is eigenvalue with second largest magnitude
- **The smaller the second largest eigenvalue (in magnitude), the faster the mixing.**
- **For λ<1 there exists an unique stationary distribution, namely the first left eigenvector of the transition matrix.**

Pagerank and convergence

- **The transition matrix pagerank uses is**

$$\tilde{P} = (1-c)P + cU$$

- **Strictly positive entries:**
  - Perron-Frobenious→1 is simple eigenvalue, others $|\lambda| < 1$
- **The second largest eigenvalue $\leq$ (1-c)**
  - Convergence rate determined by c
- **Trade-off**
  - Larger c: faster convergence, but result less specific to original P
  - Trivial example: $c = 1 \rightarrow x_\infty = x_1$ = uniform distribution
- **Nice! This means pagerank computation will converge fast.**

Pagerank generalized

- **Pagerank:**
  - seek stationary distribution as solution to $v = v\tilde{P}$

$$v = v\tilde{P} = (1-c)vP + cvU = (1\text{-}c)vP + cr$$

- $\mathbf{r_i = (vU)_i = \Sigma_j\, v_j U_{ji} = n^{-1}\, \Sigma_j\, v_j = 1}$
  - uniform in i, does not change over time

- **What happens if r is non-uniform?**

Personalized Pagerank

- **Non-uniform r → non-uniform teleportation distribution**

- **Regard r as non-uniform preference vector**
  - e.g. specific to an user.

- **Resulting v gives "personalized views" of the web.**

- **Convergence?**
  - $\tilde{P}$ = (1-c)P +cU becomes $(1-c)P + c1^T r$
  - Can show second largest eigenvalue $|\lambda| < 1 - c$ as before

Personalized Page rank

- **How to determine appropriate r for user**
  - During user query: inferring preferences from query
  - Recompute pagerank? Costly
- **Precomputation**
  - Preclassify pages according to category
  - Each category c → preference vector r_c
    - Teleportation preferentially to random page within category
  - Precompute pagerank v_c for each category
- **Query runtime**
  - Map query q to category c(q) and use pagerank $v_{c(q)}$

Multiple Categories

- **Page rank v_r is linear function of preference vector r**
  - Formally: $v = (1-c)vP + r \rightarrow v = r\,(1-(1-c)\,P)^{-1}$

$$r=\begin{pmatrix} a \\ 0 \\ 1-a \end{pmatrix} \Rightarrow v(r)=av(r_0)+(1-a)v(r_2) \qquad r_0=\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, r_2=\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- **At Query runtime:**
  - Attribute query q to class c with weight $\alpha_c > 0 : \Sigma_c\, \alpha_c = 1$
  - Return page rank vector as linear combination of precomputed page rank vectors v_c over categories

$$v = \ : \Sigma_c\, \alpha_c\, v_c$$

Rank stability

- **How does the ranking change when the link structure changes?**
- **The web-graph is changing continuously.**
- **How does that affect page-rank?**

Rank stability

$$\tilde{\mathbf{P}} = (1-c)\mathbf{P} + c\mathbf{U}$$

- **Theorem: if v is the left eigenvector of $\tilde{\mathrm{P}}$. Let the pages i_1, i_2,…, i_k be changed in any way, and let v' be the new pagerank. Then**

$$\| \mathbf{v} - \mathbf{v}' \|_1 \leq \frac{\sum_{j=1}^{k} \mathbf{v}(i_j)}{c}$$

- **So if *c* is not too close to *0*, the ranks are stable and still have good time convergence**

Acknowledgements and References

- **Based in part on slides by Purnamrita Sarkar, CMU**
- **The anatomy of a large-scale hypertextual Web search engine, Brin and Page, 1998**
- **The Second Eigenvalue of the Google Matrix, Haveliwala &. Kamvar, Stanford University Technical Report, 2003.**
- **Topic-sensitive PageRank, Haveliwala, WWW2002,**
- **Scaling Personalized Web Search, Jeh & Widom. WWW, 2003**
- **Random Walks on Graphs: A Survey, Laszlo Lovasz**
- **Reversible Markov Chains and Random Walks on Graphs, Aldous & Fill**