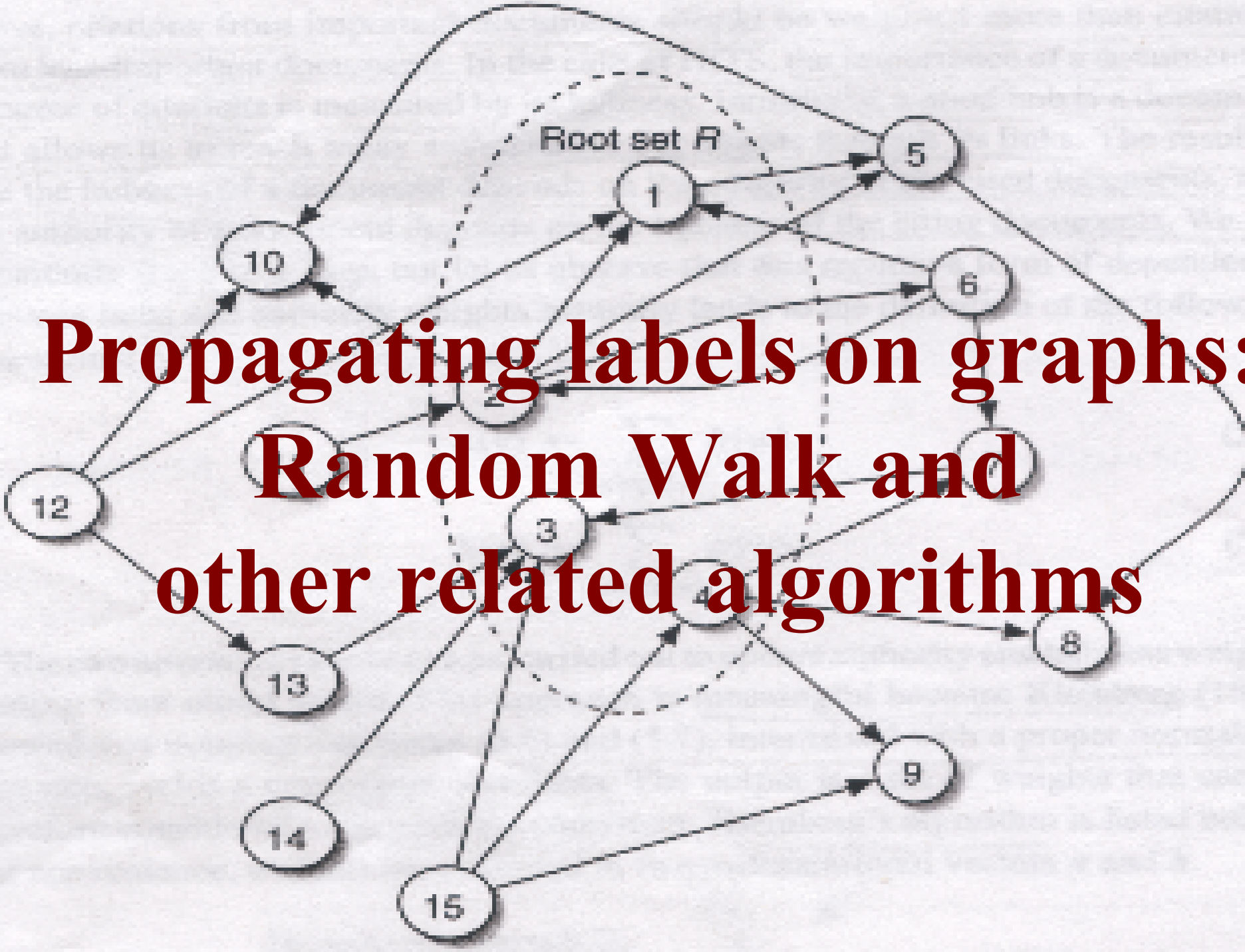


Propagating labels on graphs: Random Walk and other related algorithms

What is a Random Walk

- Given a graph and a starting point (node), we select a neighbor of it at random, and move to this neighbor;
- Then we select a neighbor of this node and move to it, and so on;
- The (random) sequence of nodes selected in this way is a *random walk* on the graph

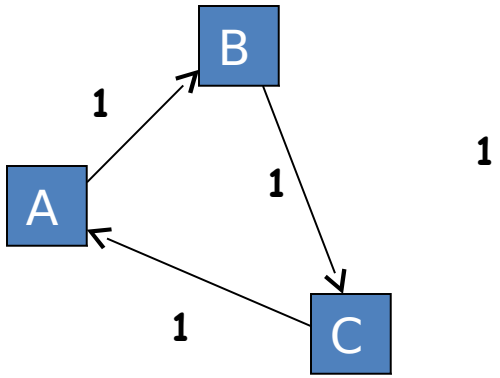
An example

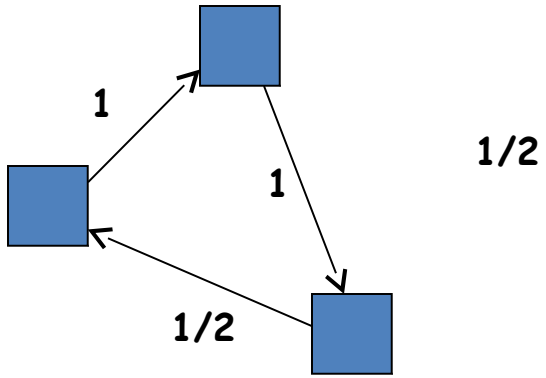
0	1	0
0	0	1
1	1	0

Adjacency matrix W

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

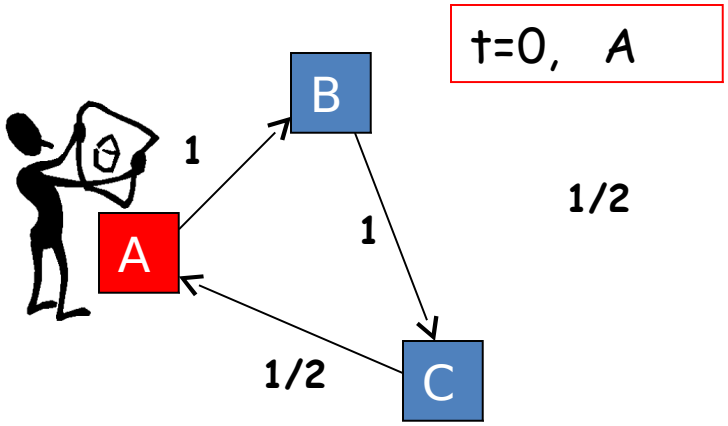
Transition matrix Q





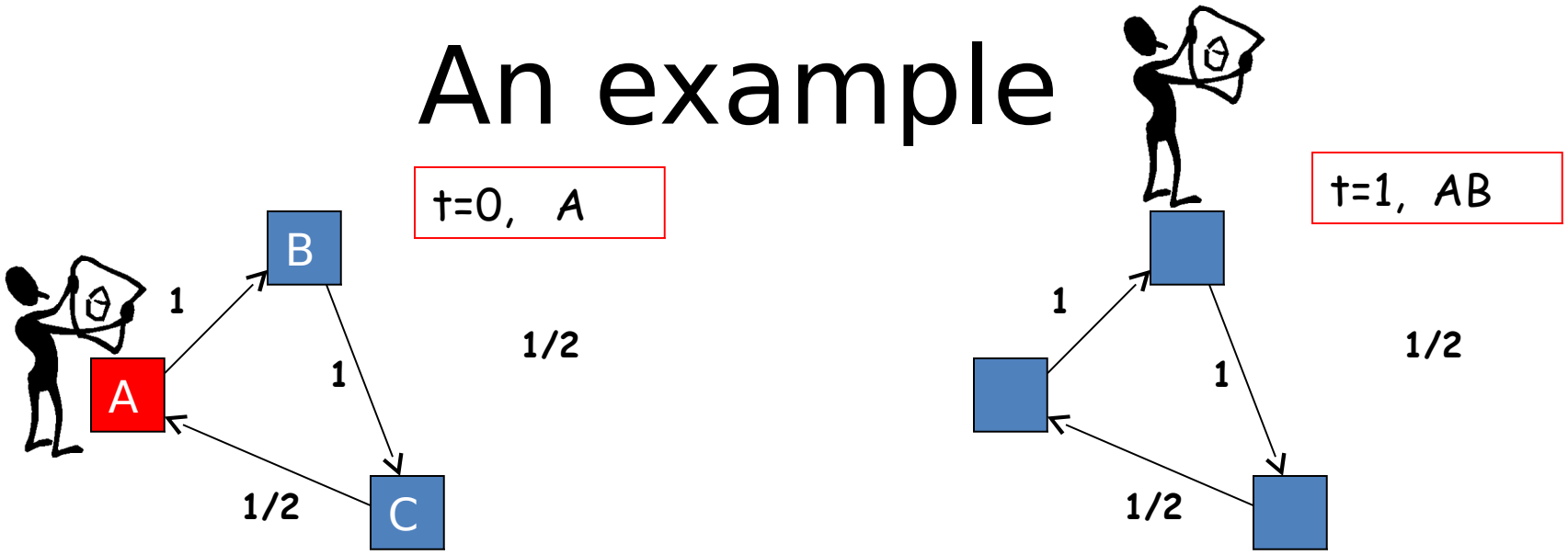
Slide from Purnamitra Sarkar, *Random Walks on Graphs: An Overview*

An example



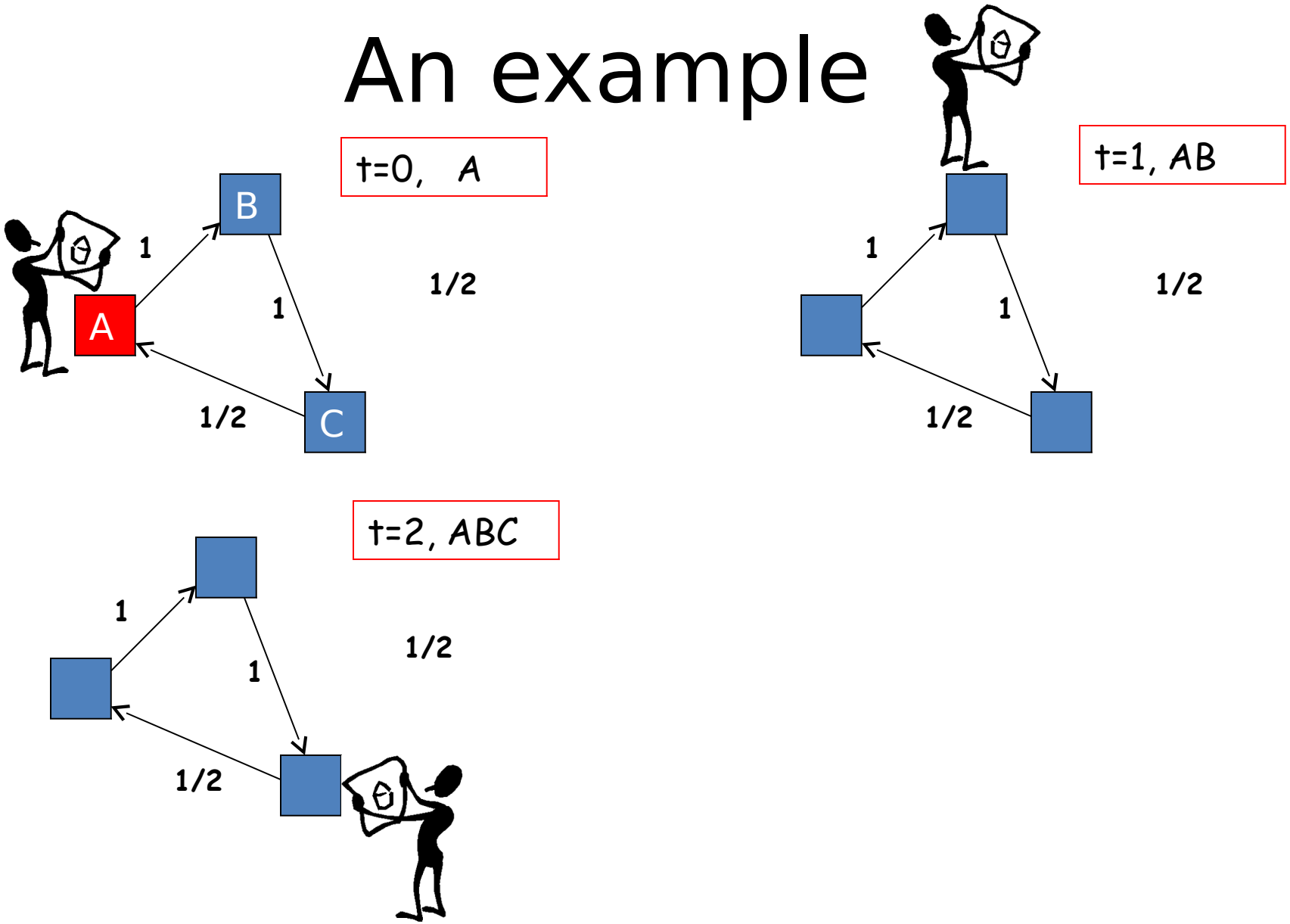
Slide from Purnamitra Sarkar, *Random Walks on Graphs: An Overview*

An example



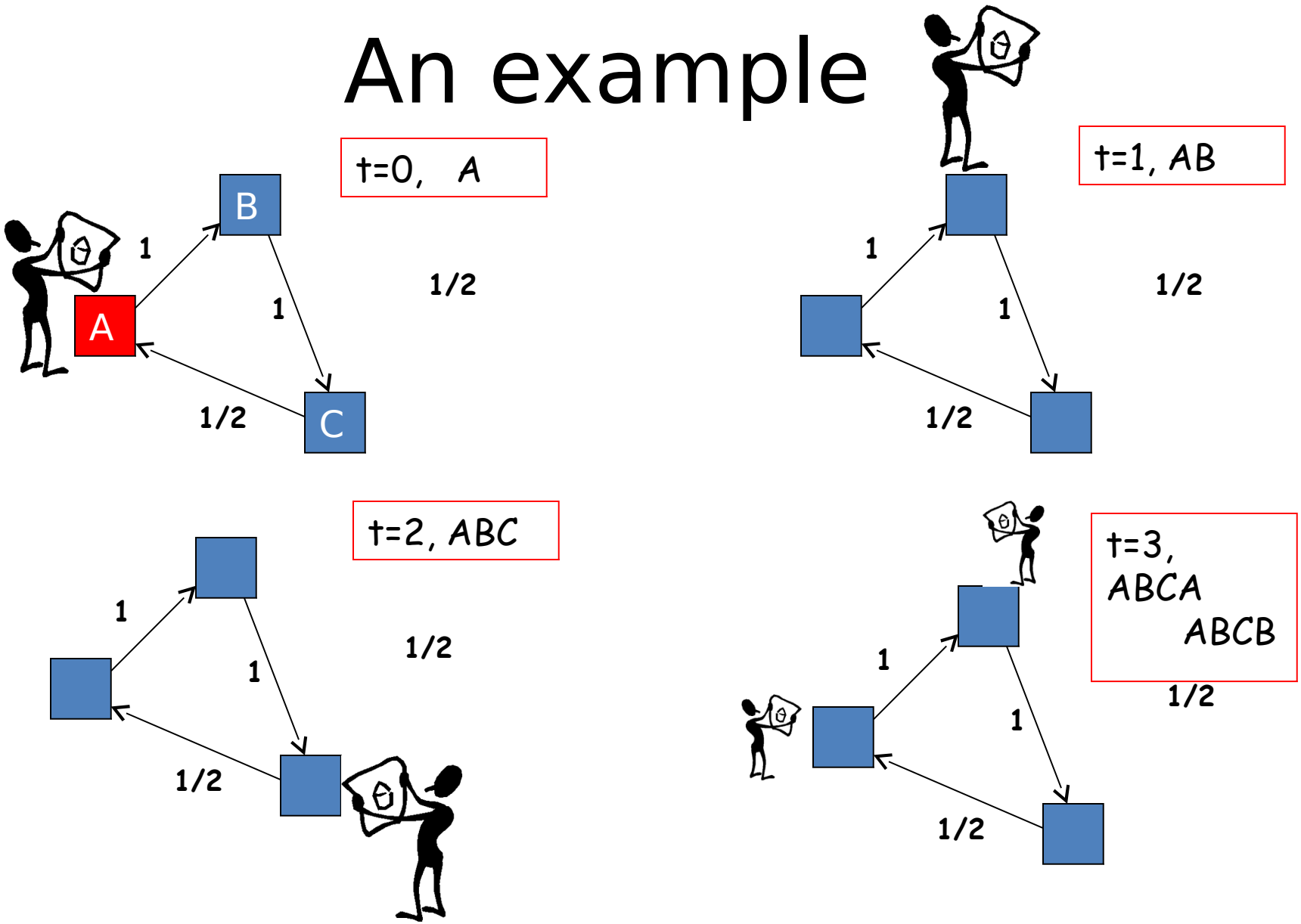
Slide from Purnamitra Sarkar, *Random Walks on Graphs: An Overview*

An example



Slide from Purnamitra Sarkar, *Random Walks on Graphs: An Overview*

An example



Slide from Purnamitra Sarkar, *Random Walks on Graphs: An Overview*

Random walks and Markov chains

- A *Markov chain* describes a stochastic process over a set of states according to a transition probability matrix
- Markov chains are *memoryless*
- *Random walks correspond to Markov chains*:
  - The set of states is the set of nodes in the graph
  - The elements of the transition probability matrix are the probabilities to follow and edge from one node to another

Random Walk algorithm

Input:
- the adjacency matrix $\boldsymbol{W}$ of a graph $G=<V,E>$
- A subset of nodes V_C having property C

• Initialization of nodes:
if $v \in V_C$ then $p_0(v) = 1 / |V_C|$ else $p_0(v)=0$

• Set transition matrix: $\boldsymbol{Q} = \boldsymbol{D}^{-1}\boldsymbol{W}$
where $\boldsymbol{D}$ is a diagonal matrix with

$$d_{ii} = \sum_j w_{ij}$$

• Iteratively update until convergence or until $t=k$
$\boldsymbol{p}_t = \boldsymbol{Q}^T \boldsymbol{p}_{t-1}$

Output: $\boldsymbol{p}_t$

Random Walk with restart

Input:
- $\boldsymbol{W}$: weight matrix of the graph
- $V_M \subset V$: genes belonging to a cancer module M
- ϵ: convergence parameter
- θ: restart probability

begin algorithm

01: for each $i \in V_M$ $p_i^o := 1/V_M$

02: for each $i \notin V_M$ $p_i^o := 0$

03: for each $i \in V$ $d_{ii} := \sum_j w_{ij}$

04: $\boldsymbol{Q} := \boldsymbol{D}^{-1}\boldsymbol{W}$

05: t:=0

06: repeat

07: t := t + 1

07: $\boldsymbol{p}^t = (1-\theta)\,\boldsymbol{Q}^T\boldsymbol{p}^{t-1} + \theta\,\boldsymbol{p}^o$

08: until $(||\boldsymbol{p}^t - \boldsymbol{p}^{t-1}|| < \epsilon)$

09: for each $i \in V$

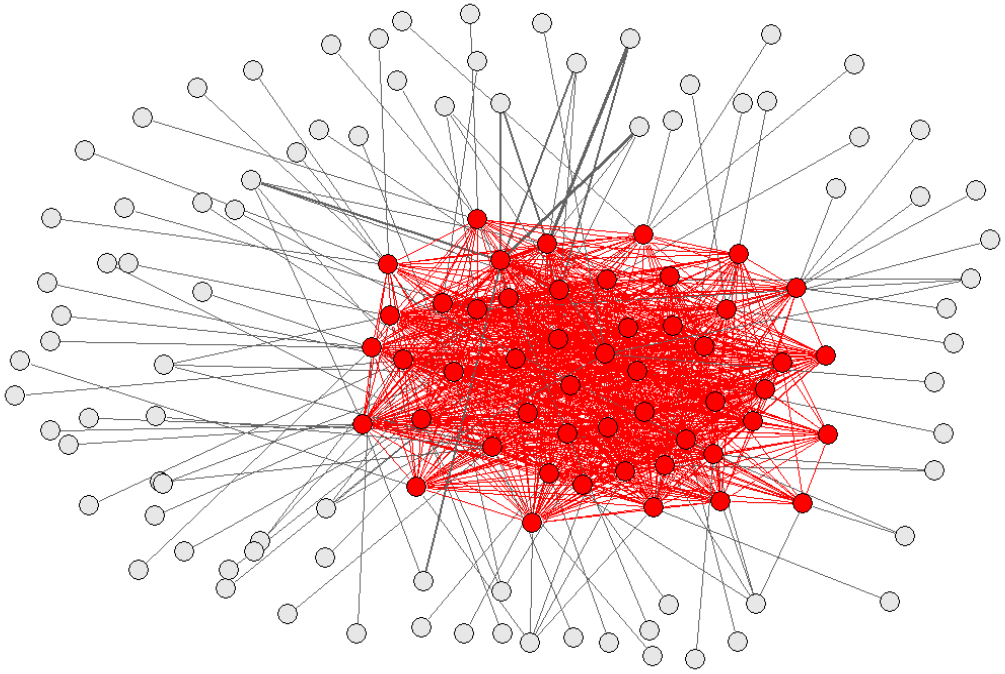
10: $p_i^t := p_i^t / \sum_j p_j^t$

end algorithm.

Output: the probability vector $\boldsymbol{p}^t$

Random Walk algorithm to rank genes w.r.t to a given "property" C

- A subset V_C of a set of genes V have "a priori" known property C
- Can we rank the other genes in the set $V \setminus V_C$ w.r.t their likelihood to belong to V_C ?



Random walk algorithm

C can be e.g. a *disease* (gene disease prioritization) or a GO term (gene function prediction)

Label propagation algorithm

Algorithm 11.1 Label propagation (Zhu and Ghahramani [2002])

Compute affinity matrix $\mathbf{W}$ from (11.1)
Compute the diagonal degree matrix $\mathbf{D}$ by $\mathbf{D}_{ii} \leftarrow \sum_j W_{ij}$
Initialize $\hat{Y}^{(0)} \leftarrow (y_1, \ldots, y_l, 0, 0, \ldots, 0)$
Iterate
1. $\hat{Y}^{(t+1)} \leftarrow \mathbf{D}^{-1}\mathbf{W}\hat{Y}^{(t)}$
2. $\hat{Y}_l^{(t+1)} \leftarrow Y_l$

until convergence to $\hat{Y}^{(\infty)}$
Label point x_i by the sign of $\hat{y}_i^{(\infty)}$

- Examples can be split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$
- The algorithm tries to maximizes the consistency of the unlabeled examples with the topology of the graph
- The algorithm forces the labels on the labeled data: $(\hat{Y}_l = Y_l)$
- The algorithm iterates till to the convergence

Label spreading algorithm

Algorithm 11.3 Label spreading (Zhou et al. [2004])

Compute the affinity matrix $\mathbf{W}$ from (11.1) for $i \neq j$ (and $\mathbf{W}_{ii} \leftarrow 0$)
Compute the diagonal degree matrix $\mathbf{D}$ by $\mathbf{D}_{ii} \leftarrow \sum_j W_{ij}$
Compute the normalized graph Laplacian $\mathcal{L} \leftarrow \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$
Initialize $\hat{Y}^{(0)} \leftarrow (y_1, \ldots, y_l, 0, 0, \ldots, 0)$
Choose a parameter $\alpha \in [0, 1)$
Iterate $\hat{Y}^{(t+1)} \leftarrow \alpha\mathcal{L}\hat{Y}^{(t)} + (1-\alpha)\hat{Y}^{(0)}$ until convergence to $\hat{Y}^{(\infty)}$
Label point x_i by the sign of $\hat{y}_i^{(\infty)}$

- Similar to the Label propagation algorithm, but:
- The normalized graph Laplacian is used instead
- The algorithm does not force the labeled data (useful with noisy data)
- At each step a contribution of the initial labeling is considered (convex combination)
- It can be shown that a different cost criterion is minimized

The previous semi-supervised algorithms minimize a quadratic cost function

Examples split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$

The previous semi-supervised algorithms minimize a quadratic cost function

Examples split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$

A. Consistency with the initial labeling: $\sum_{i=1}^{l} (\hat{y}_i - y_i)^2 = \|\hat{Y}_l - Y_l\|^2.$

The previous semi-supervised algorithms minimize a quadratic cost function

Examples split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$

A. Consistency with the initial labeling:

$$\sum_{i=1}^{l} (\hat{y}_i - y_i)^2 = \|\hat{Y}_l - Y_l\|^2.$$

B. Consistency with the geometry of the data (internal consistency):

$$\begin{aligned}
\frac{1}{2} \sum_{i,j=1}^{n} \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2 &= \frac{1}{2} \left(2 \sum_{i=1}^{n} \hat{y}_i^2 \sum_{j=1}^{n} \mathbf{W}_{ij} - 2 \sum_{i,j=1}^{n} \mathbf{W}_{ij} \hat{y}_i \hat{y}_j \right) \\
&= \hat{Y}^\top (\mathbf{D} - \mathbf{W}) \hat{Y} \\
&= \hat{Y}^\top L \hat{Y}
\end{aligned}$$

The previous semi-supervised algorithms minimize a quadratic cost function

Examples split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$

A. Consistency with the initial labeling: $\sum_{i=1}^{l} (\hat{y}_i - y_i)^2 = \|\hat{Y}_l - Y_l\|^2.$

B. Consistency with the geometry of the data (internal consistency):

$$\begin{aligned} \frac{1}{2} \sum_{i,j=1}^{n} \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2 &= \frac{1}{2} \left(2 \sum_{i=1}^{n} \hat{y}_i^2 \sum_{j=1}^{n} \mathbf{W}_{ij} - 2 \sum_{i,j=1}^{n} \mathbf{W}_{ij} \hat{y}_i \hat{y}_j \right) \\ &= \hat{Y}^\top (\mathbf{D} - \mathbf{W}) \hat{Y} \\ &= \hat{Y}^\top L \hat{Y} \end{aligned}$$

Putting together A and B we can obtain a cost function to be minimized:

$$C(\hat{Y}) = \|\hat{Y}_l - Y_l\|^2 + \mu \hat{Y}^\top L \hat{Y} + \mu\epsilon \|\hat{Y}\|^2$$

The previous semi-supervised algorithms minimize a quadratic cost function

Examples split in labeled and unlabeled: $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$

A. Consistency with the initial labeling:

$$\sum_{i=1}^{l} (\hat{y}_i - y_i)^2 = \|\hat{Y}_l - Y_l\|^2.$$

B. Consistency with the geometry of the data (internal consistency):

$$\begin{aligned} \frac{1}{2} \sum_{i,j=1}^{n} \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2 &= \frac{1}{2} \left(2 \sum_{i=1}^{n} \hat{y}_i^2 \sum_{j=1}^{n} \mathbf{W}_{ij} - 2 \sum_{i,j=1}^{n} \mathbf{W}_{ij} \hat{y}_i \hat{y}_j \right) \\ &= \hat{Y}^\top (\mathbf{D} - \mathbf{W}) \hat{Y} \\ &= \hat{Y}^\top L \hat{Y} \end{aligned}$$

Putting together A and B we can obtain a cost function to be minimized:

$$C(\hat{Y}) = \|\hat{Y}_l - Y_l\|^2 + \mu \hat{Y}^\top L \hat{Y} + \mu\epsilon \|\hat{Y}\|^2$$

It can be shown that previous network-based algorithms minimize a quadratic cost function.