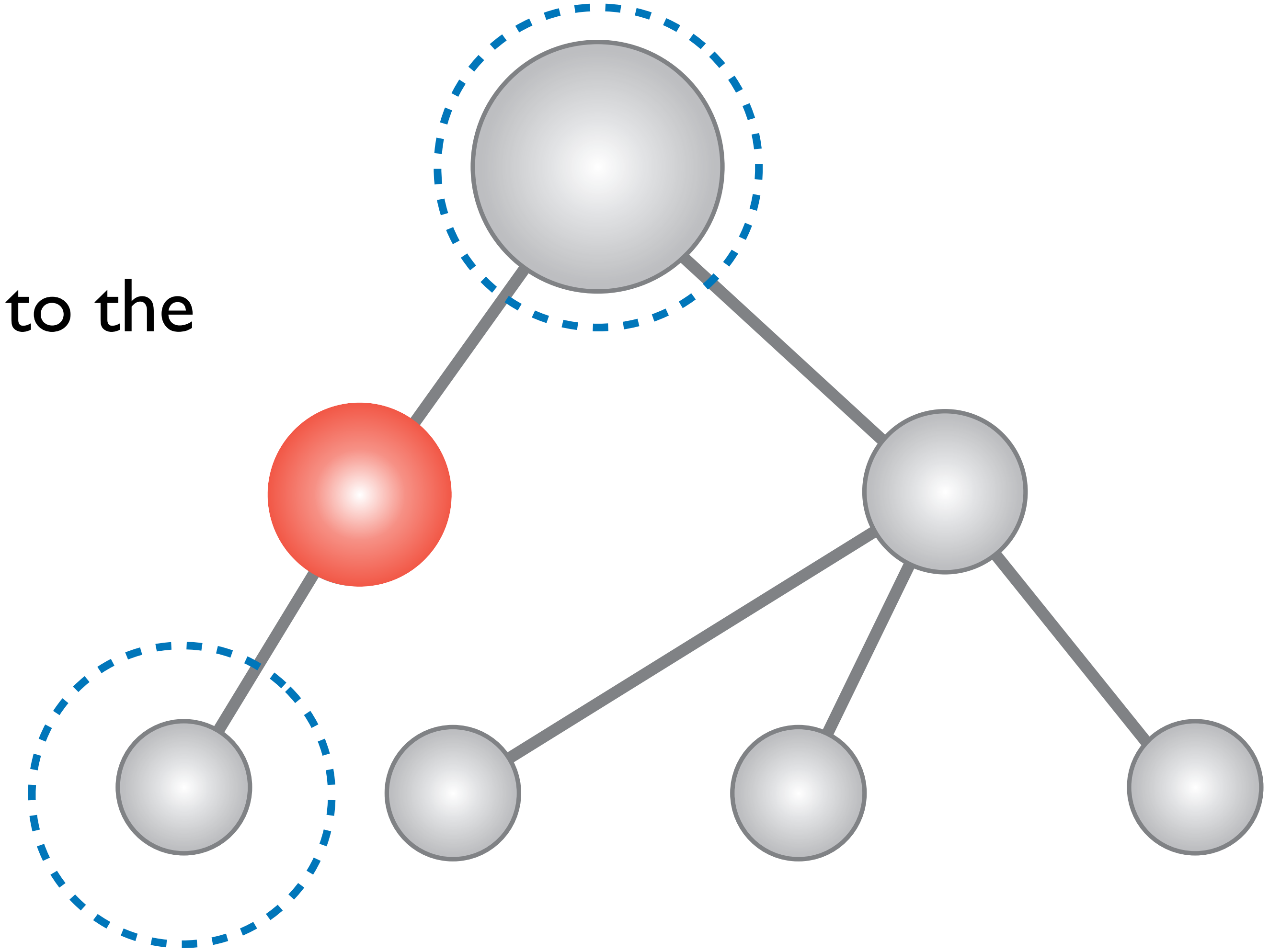
CSE 427 Computational Biology

Lecture 8
Graph diffusion

1st-order proximity only models local structure

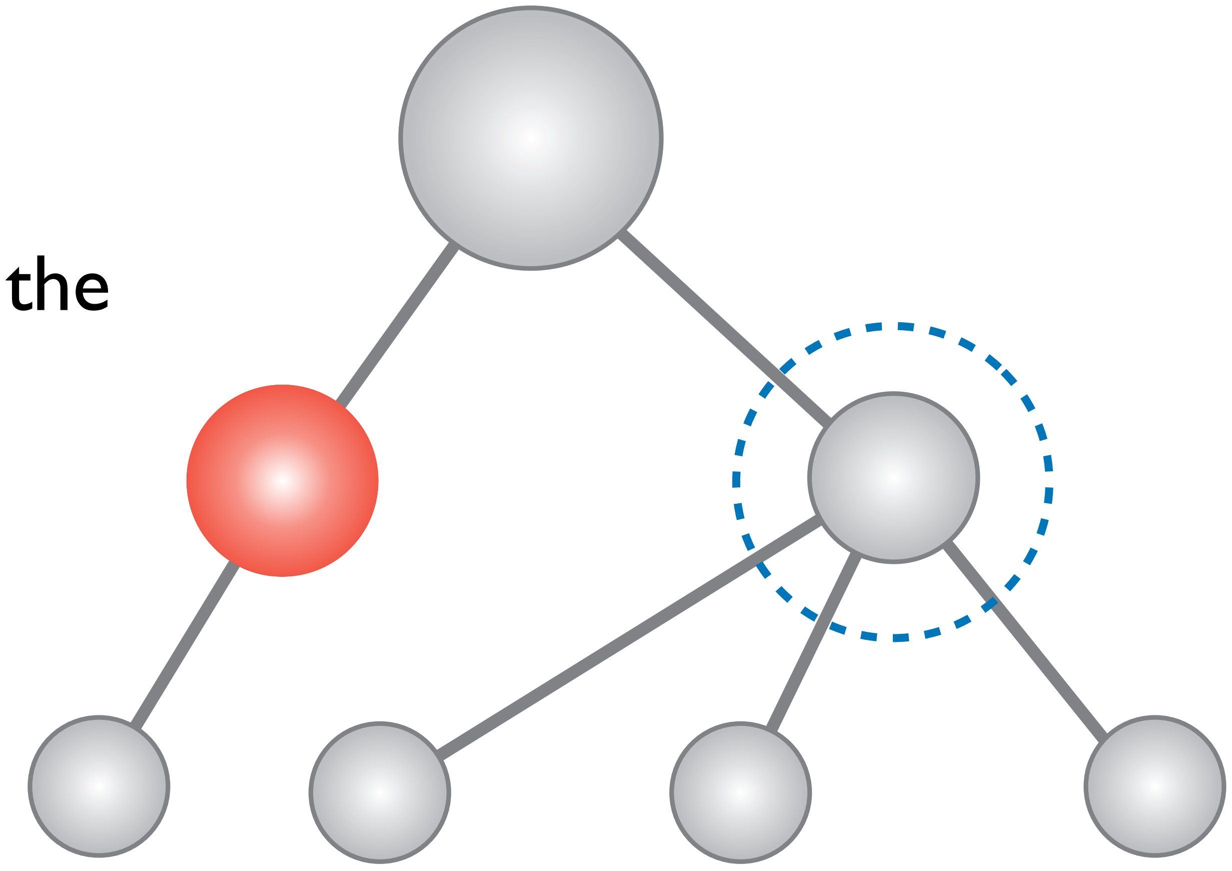
Q: Which node is topologically similar to the red node?



Direct neighbors: two nodes are considered similar if they share an edge

2nd-order proximity only models local structure

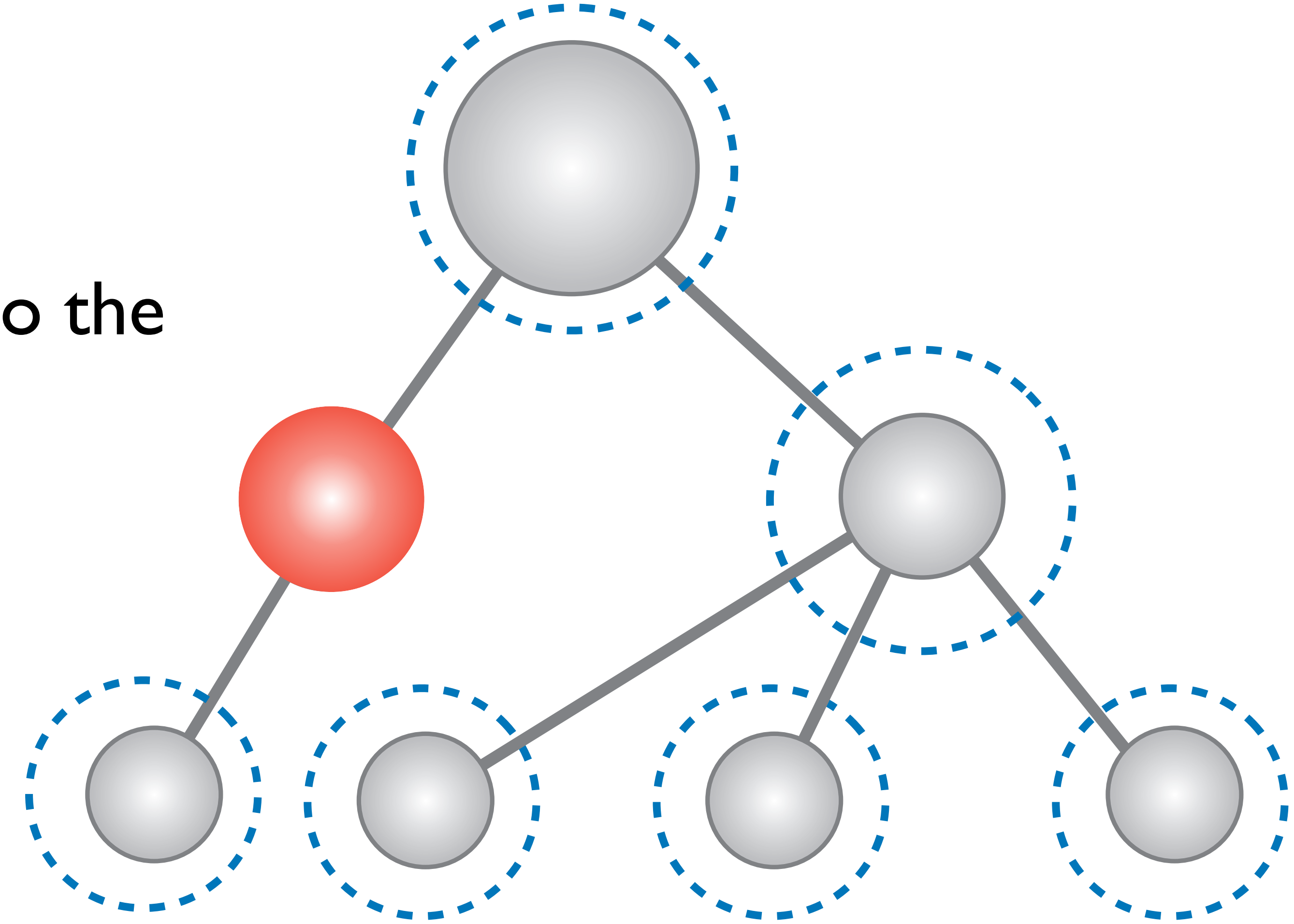
Q: Which node is topologically similar to the red node?



Neighbors' neighbors: two nodes are considered similar if they share many neighboring/adjacent nodes

Key idea:
high-order (nth-order) proximity models global structure

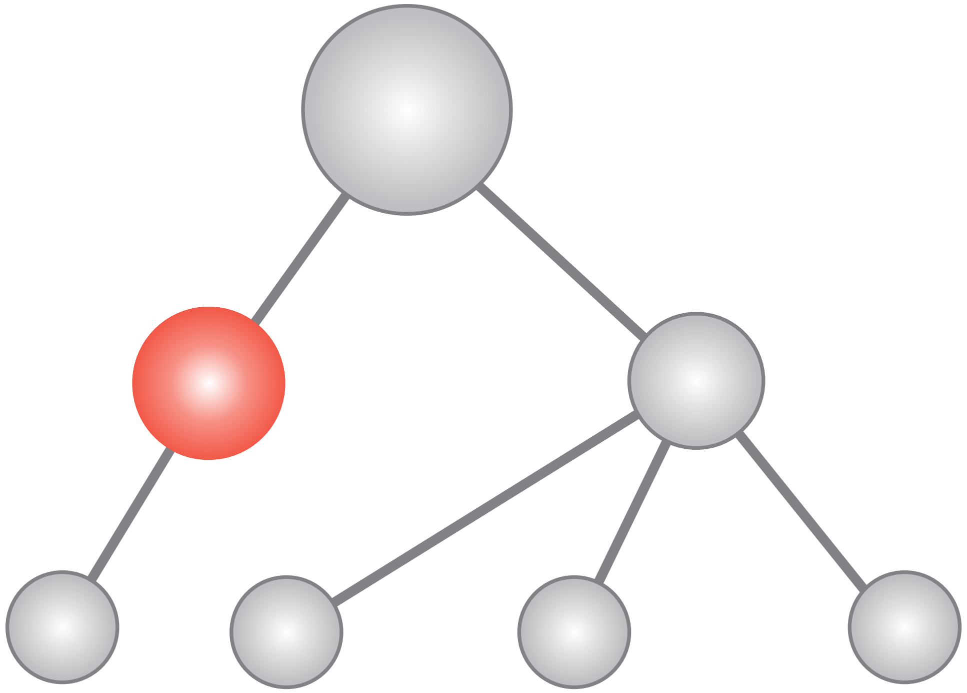
Q: Which node is topologically similar to the red node?



All nodes in the graph: two nodes are considered similar if their distances to all other nodes are similar

The Math:

use diffusion model on each node to calculate topological similarity

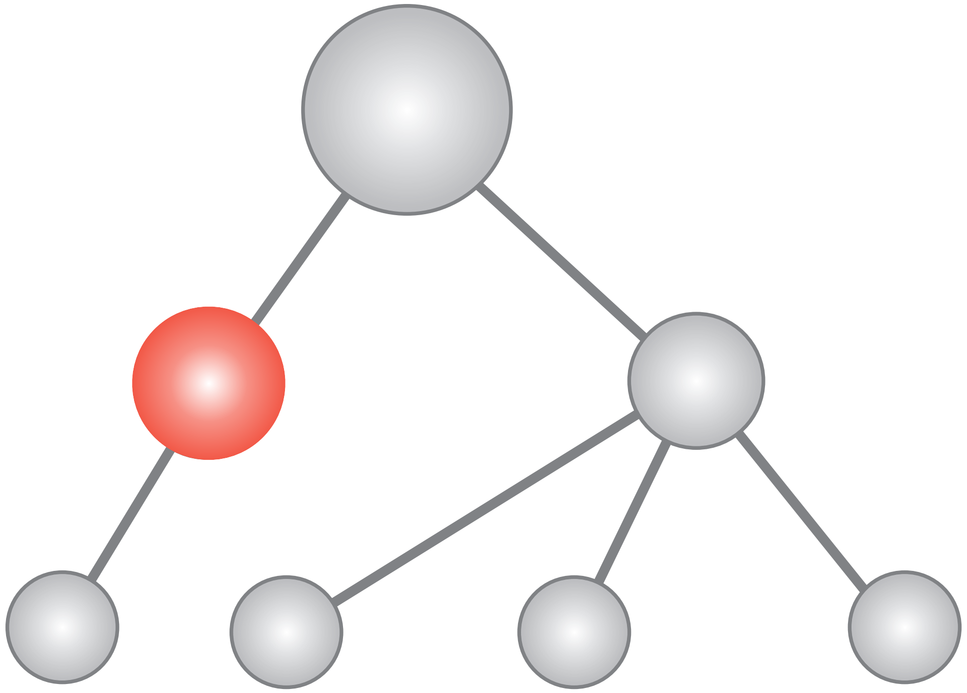


Input: adjacency matrix of the hierarchy (undirected)

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t\boldsymbol{B} + p_r\boldsymbol{I}$$

The Math:

use diffusion model on each node to calculate topological similarity



Input: adjacency matrix
of the hierarchy (undirected)

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^{t}\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node

Node i and node j are topologically similar if s_i is similar to s_j

The Math:

use diffusion model on each node to calculate topological similarity

Input: adjacency matrix
of the hierarchy (undirected)

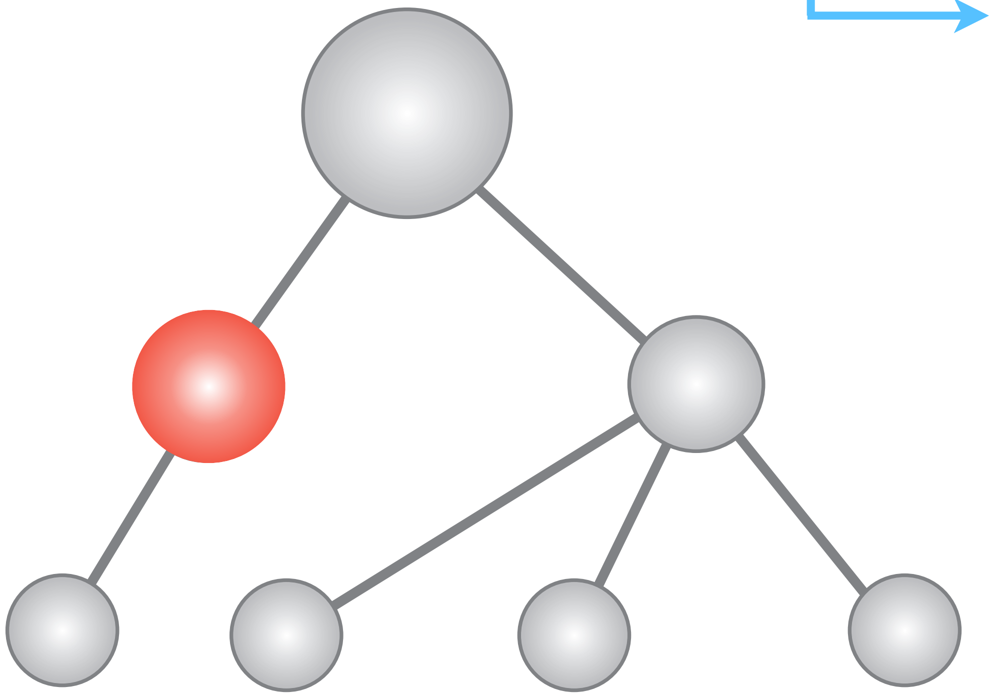
Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^{t}$.

Output: equilibrium distribution s_i starting from node i

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^{t}\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node



Node i and node j are topologically similar if $\boldsymbol{s}_i$ is similar to $\boldsymbol{s}_j$

The Math:

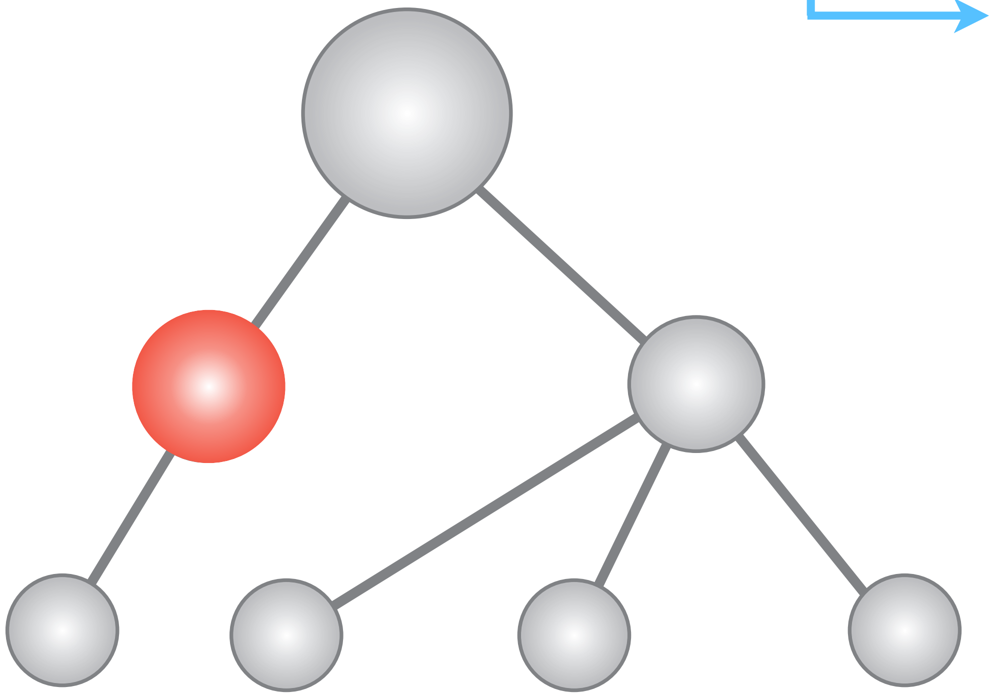
use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^{t}$.

Output: equilibrium distribution s_i starting from node i

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^{t}\boldsymbol{B} + p_r\boldsymbol{I}$$



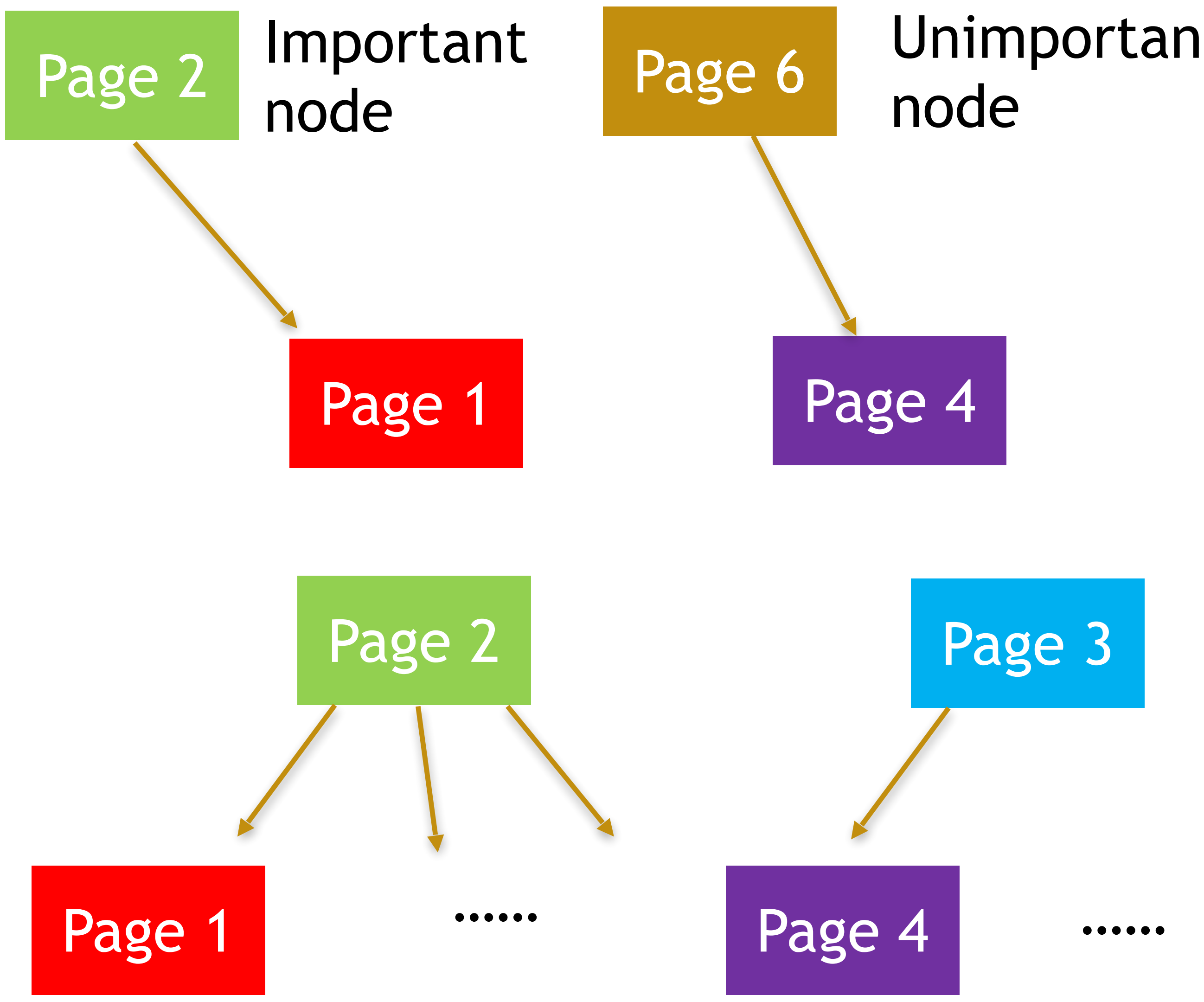
Random walk to neighbors

Restart to the original node

Motivation

- Given a set of web pages with links between them, we would like to rank the pages in order of importance.

- We can model this as a graph problem where web pages are vertices and links are edges.



Page 2

Important node

Page 1

Page 6

Unimportant node

Page 4

Page 2

Page 1

......

Page 3

Page 4

......

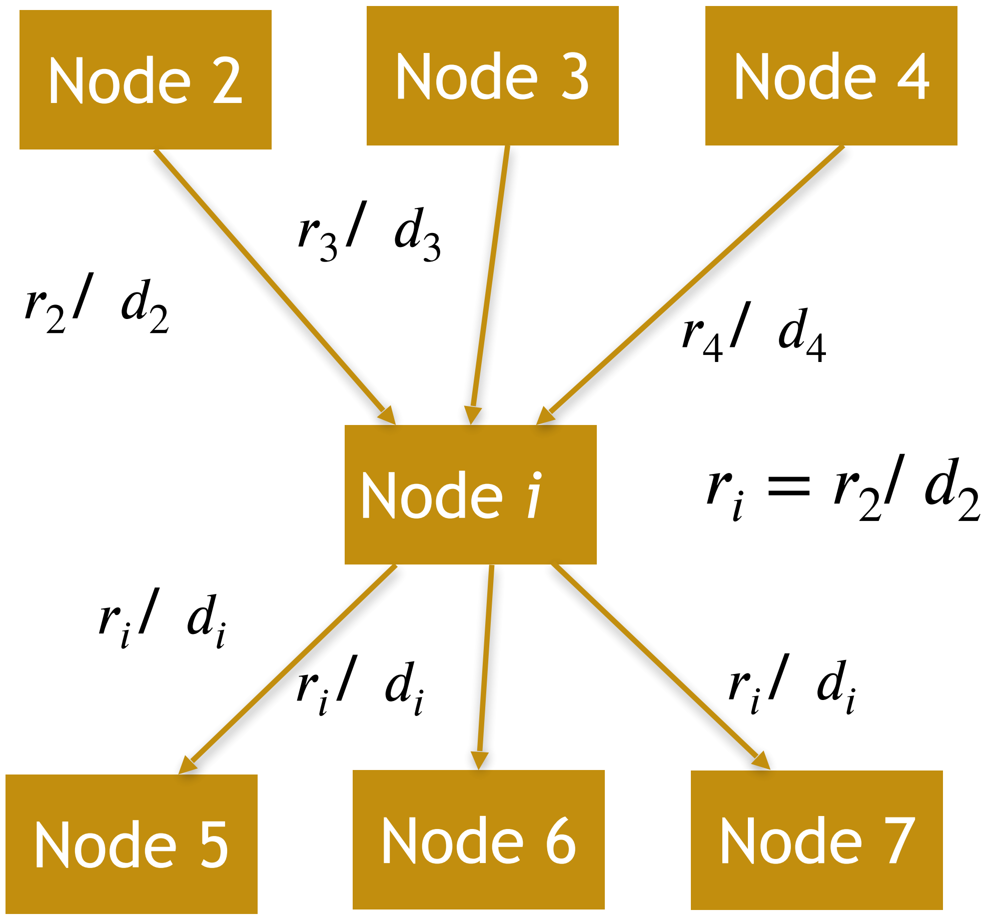
1. A link from an important edge is more significant than a link from an unimportant web page.

2. Being linked from a page with many outgoing links is less significant than being linked from a page with few outgoing

Random walk

Every node votes for its neighbors and gets votes from neighbors

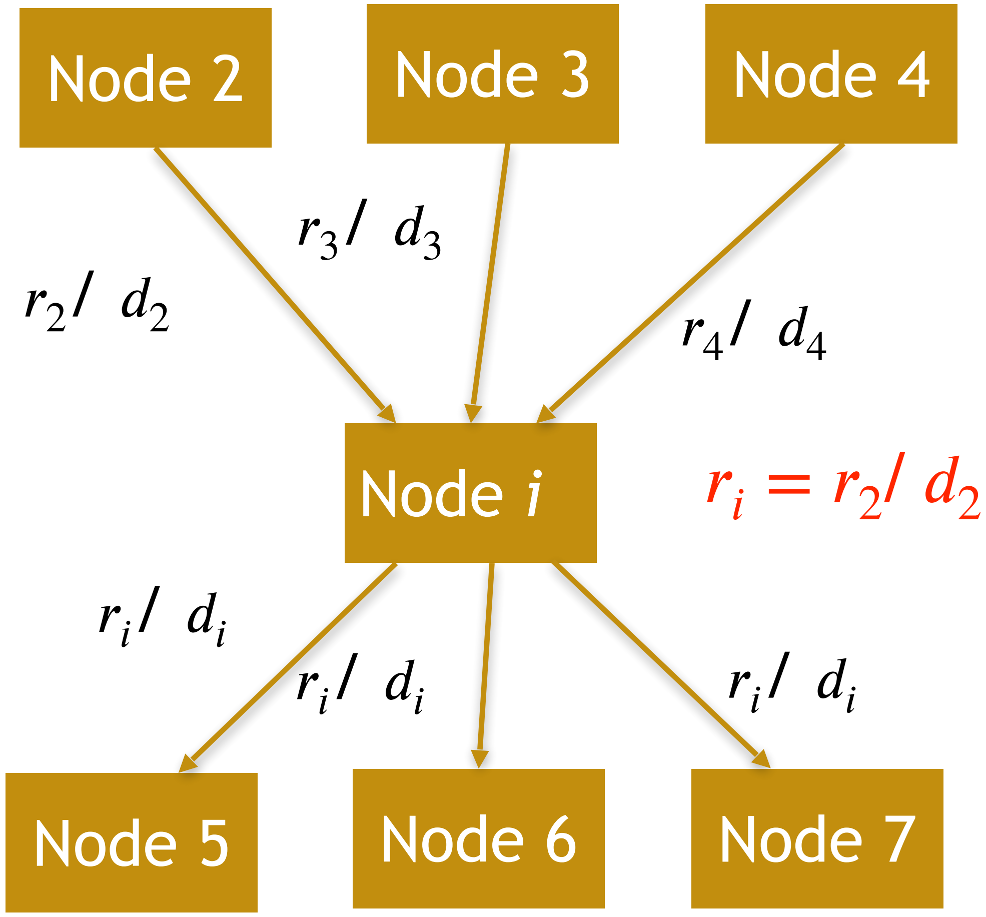
- Each link's vote is proportional to the importance of its source node
- If node i with importance r_i has d_i out-links, each link gets r_i / d_i votes
- node i's own importance r_i is the sum of the votes on its in-links



$$r_i = r_2/ d_2 + r_3/ d_3 + r_4/ d_4$$

Random walk

Every node votes for its neighbors and gets votes from neighbors



Node 2

Node 3

Node 4

$r_2/\ d_2$

$r_3/\ d_3$

$r_4/\ d_4$

Node *i*

$r_i = r_2/\ d_2 + r_3/\ d_3 + r_4/\ d_4$

$r_i/\ d_i$

$r_i/\ d_i$

$r_i/\ d_i$

$r_5 = r_i/\ d_i + \ldots$

Node 5

Node 6

Node 7

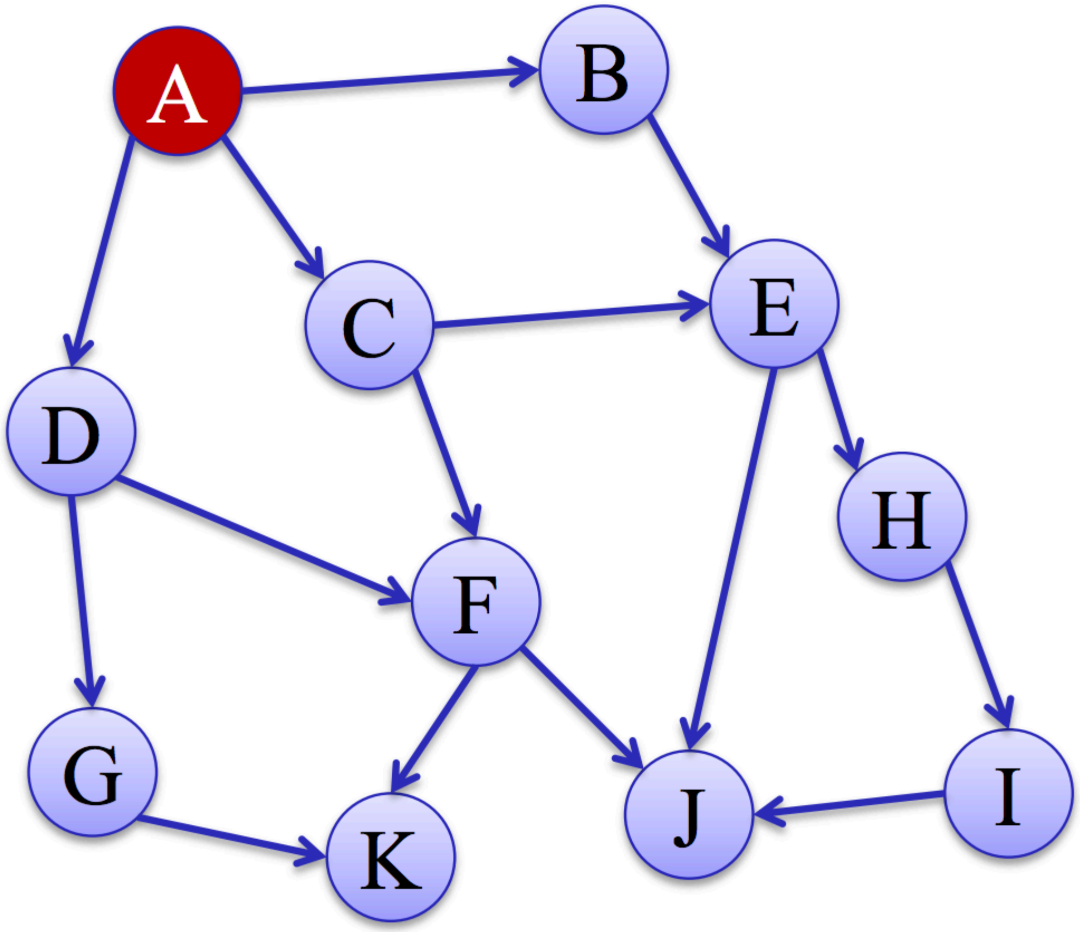
$r_7 = r_i/\ d_i + \ldots$

$r_6 = r_i/\ d_i + \ldots$

How can you solve so many equations together?

What is a Random Walk

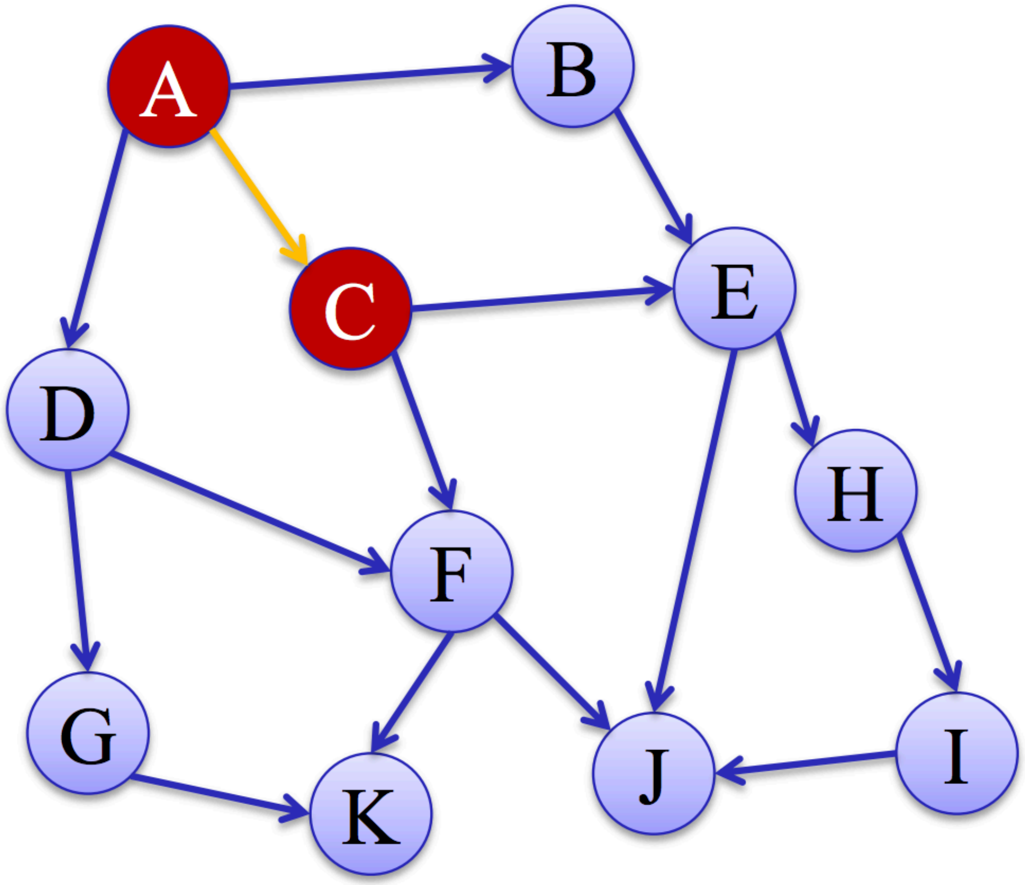
Given a graph and a starting node, we select a neighbor of it at random, and move to this neighbor



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

What is a Random Walk

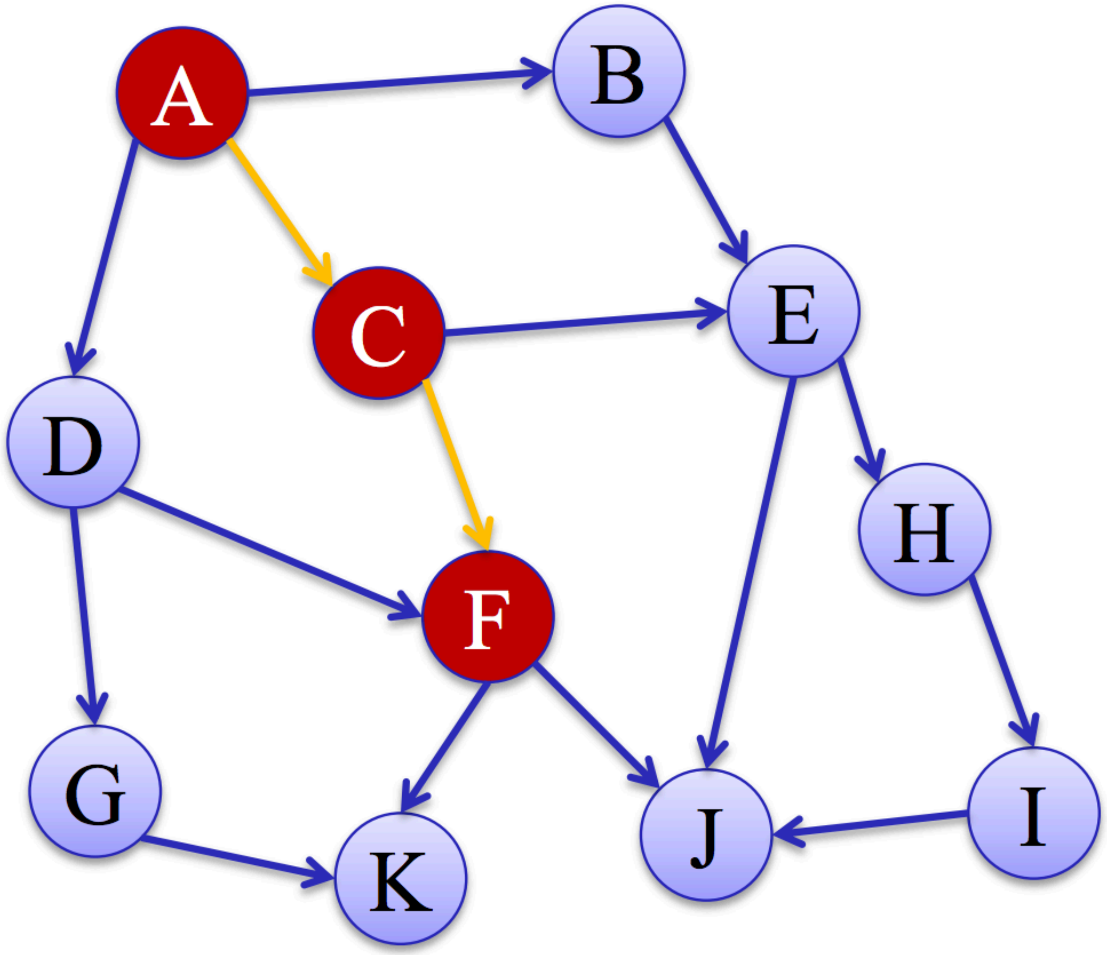
We select a neighbor of it at random, and move to this neighbor



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

What is a Random Walk

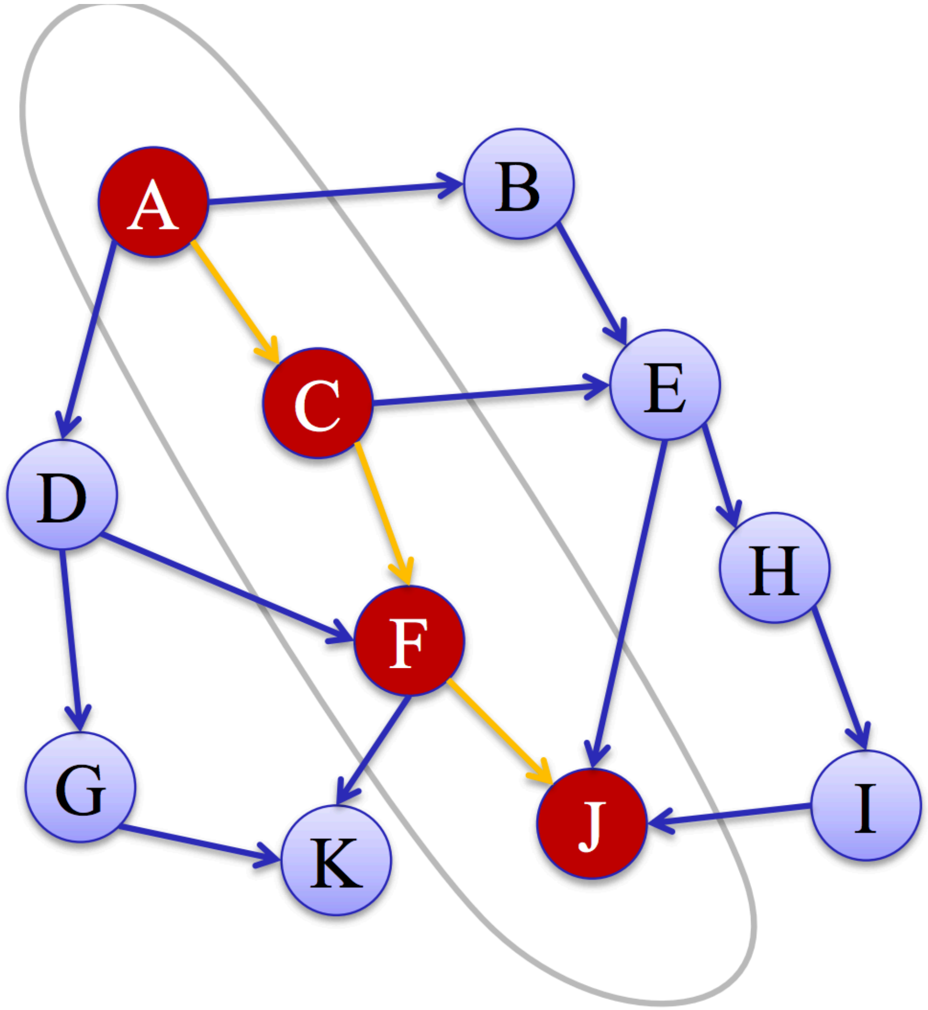
Then we select a neighbor of this node and move to it, and so on.



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

What is a Random Walk

The (random) sequence of nodes selected this way is a random walk on the graph



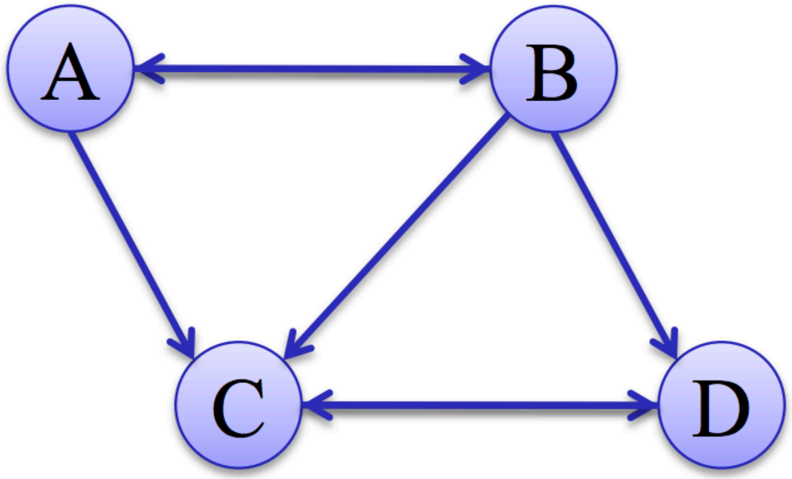
Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Adjacency Matrix vs. Transition Matrix

- A transition matrix is a stochastic matrix where each element a_{ij} represents the probability of moving from i to j, with each row summing to 1.

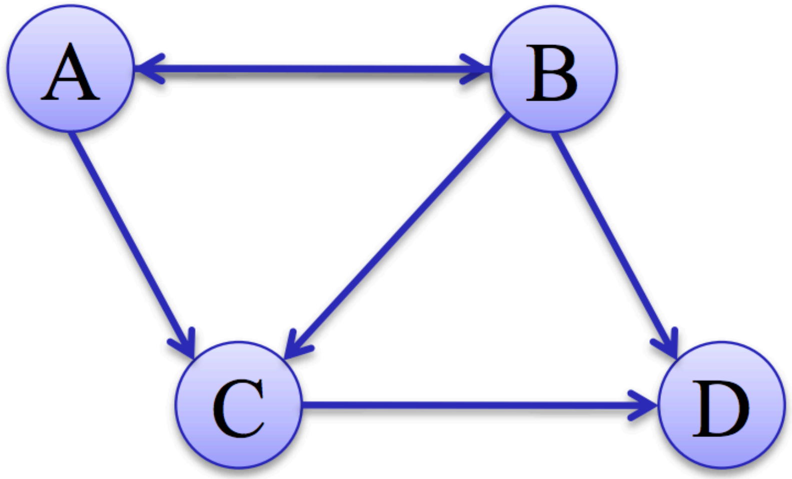
Adjacency Matrix

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



Transition Matrix

$$\begin{bmatrix} 0 & 1/2 & 1/2 & 0 \\ 1/3 & 0 & 1/3 & 1/3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Markov chains

- A Markov chain describes a discrete time stochastic process over a set of states

$$S = \{s_1, s_2, \dots s_n\}$$

according to a transition probability matrix

$$P = \{P_{ij}\}$$

P_{ij} = probability of moving to state j when at state i

- Markov Chains are memoryless: The next state of the chain depends only at the current state

Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Stationary Distribution

- $x_t(i)$ = probability that the surfer is at node *i* at time *t*

- $x_{t+1}(j) = \sum_i x_t(i) \,.\, P_{ij}$
- $x_{t+1} = x_t P = x_{t-1} P P = \mathrm{x}_0 \mathrm{P}^\mathrm{t}$

- What happens when the surfer keeps walking for a long time?
  - We get a stationary distribution

Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Stationary Distribution

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

- When the surfer keeps walking for a long time, the distribution does not change any more: $x_{t+1}(i) = x_t(i)$

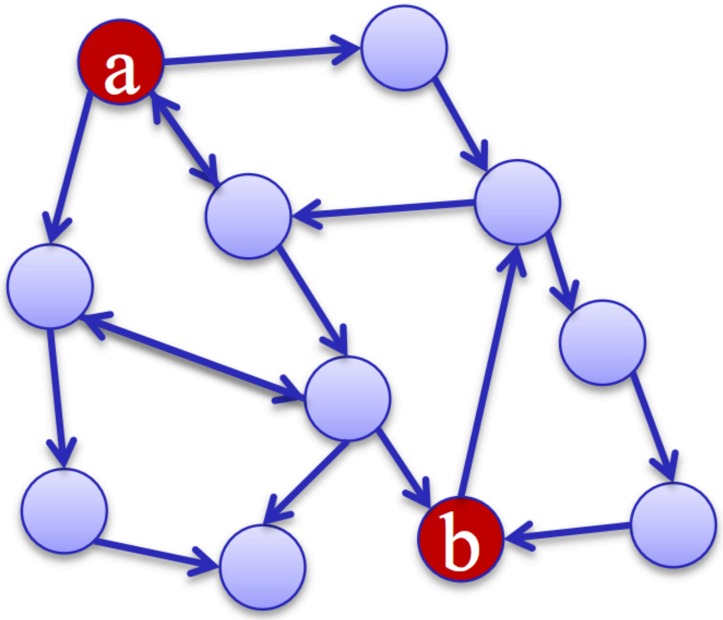
- For “well-behaved” graphs this does not depend on the start distribution

Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Hitting Time

- How long does it take to hit node *b* in a random walk starting at node *a* ?

- Hitting time from node i to node j
  - Expected number of hops to hit node *j* starting at node *i*.
  - Not symmetric
  - $h(i,j) = 1 + \Sigma_{k \in adj(i)} P(i,k)\, h(k,j)$

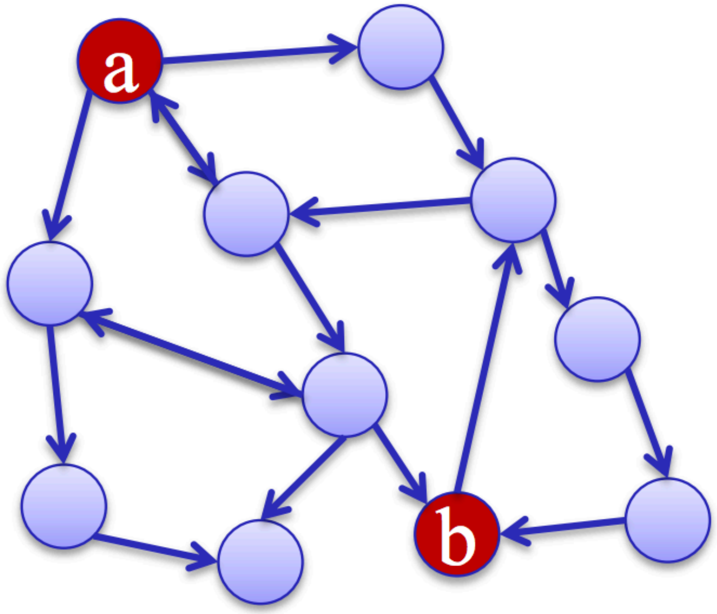


Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Commute Time

- How long does it take to hit node *b* in a random walk starting at node *a* and come back to *a*?

- Commute time from node *i* to node *j*
  - Expected number of hops to hit node *j* starting at node *i* and come back to *i*.
  - Symmetric
  - $c(i,j) = h(i,j) + h(j,i)$



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

- Random walk
  - Starting from different node results in the same probability distribution
- Random walk with restart
  - Starting from different node results in different probability distribution

Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^t$.

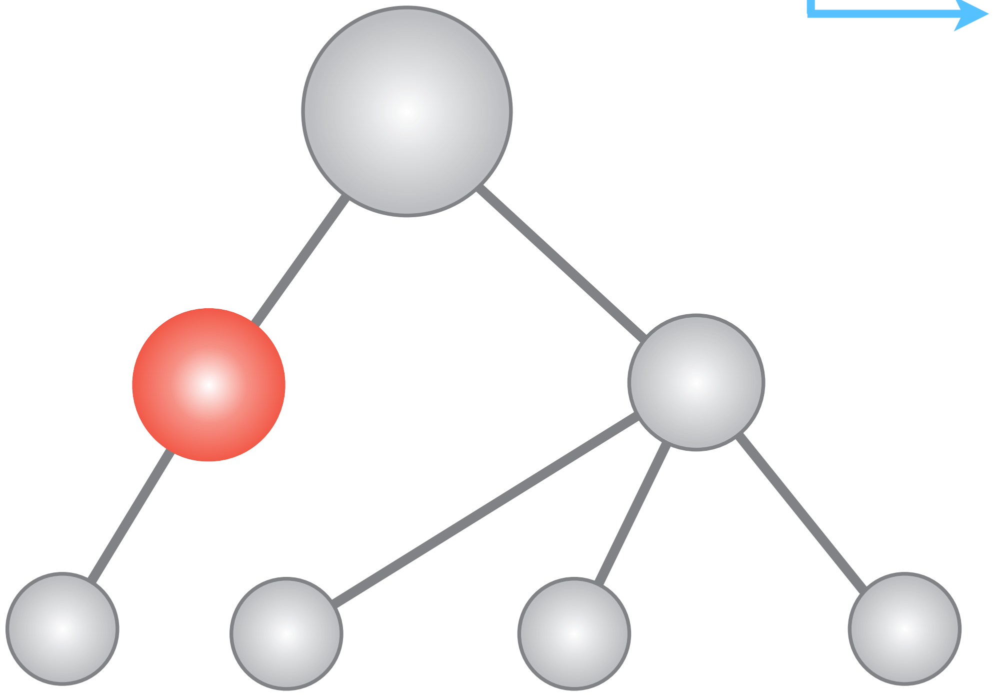
Output: equilibrium distribution s_i starting from node i

Input: adjacency matrix of the hierarchy (undirected)

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

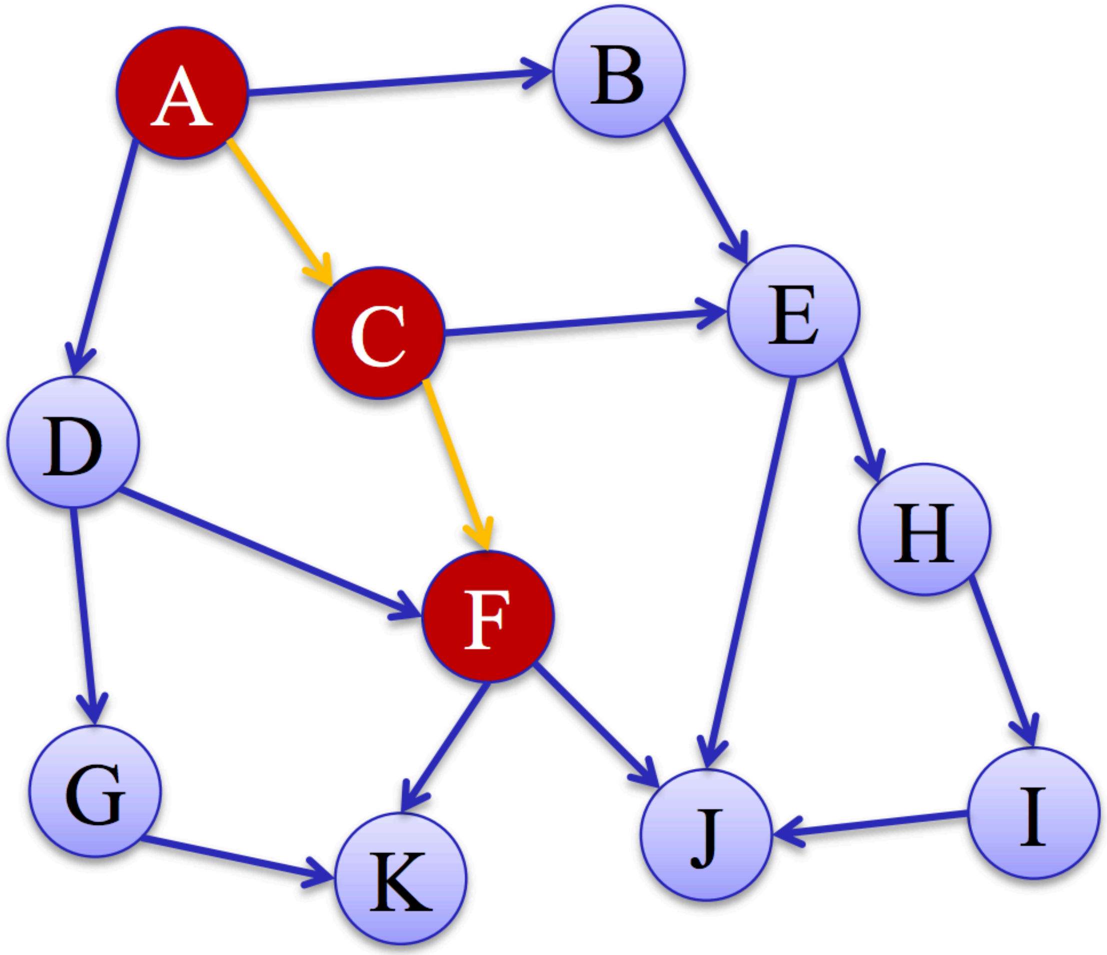
Restart to the original node



Source: Network-based stratification of tumor mutations

Random Walk

Then we select a neighbor of this node and move to it, and so on.



Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking

Random walk

The flow equations can be written:

$$r = Mr$$

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix} = \begin{bmatrix} 1/d_1 , & 0 , & \dots , & 1/d_N \\ 1/d_1 , & 1/d_2 , & \dots , & 0 \\ \vdots & \vdots & & \vdots \\ 0 , & 1/d_2 , & \dots , & 1/d_N \\ \vdots & \vdots & & \vdots \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix}$$

M is a Markov matrix since each column sums equal to 1

How to solve this ?

Power Iteration method

Initialize: $r^0 = [\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}]^T$

While $||r^{k+1} - r^k||_2$ > 0.0001:

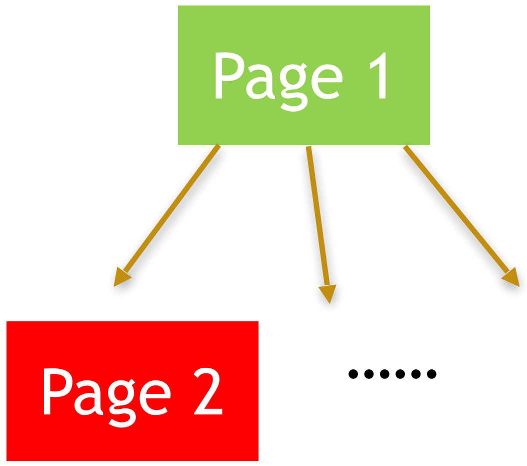
$$r^{k+1} = Mr^k$$

$$\begin{array}{ccccccc}
1/d_1 & , & 0 & , & \ldots & , & 1/d_N \\
1/d_1 & , & 1/d_2 & , & \ldots & , & 0 \\
\vdots & & \vdots & & & & \vdots \\
0 & , & 1/d_2 & , & \ldots & , & 1/d_N \\
\vdots & & \vdots & & & & \vdots
\end{array}$$

Random walk interpretation

The vector r can be reinterpreted as a probability vector to visit each website

- Imagine a random web surfer
  - At any time *k*, surfer has a probability vector r^k to visit a web page following the out-link.
  - Process repeats indefinitely

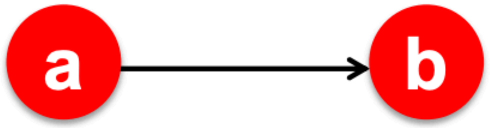


$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix} = \begin{bmatrix} 0 & , & 1/d_2 & , & \dots & , & 1/d_N \\ 1/d_1 & , & 0 & , & \dots & , & 0 \\ \vdots & & \vdots & & & & \vdots \\ 1/d_1 & , & 1/d_2 & , & \dots & , & 1/d_N \\ \vdots & & \vdots & & & & \vdots \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix}$$

$$r = Mr$$

Problem of random walk

Dead ends



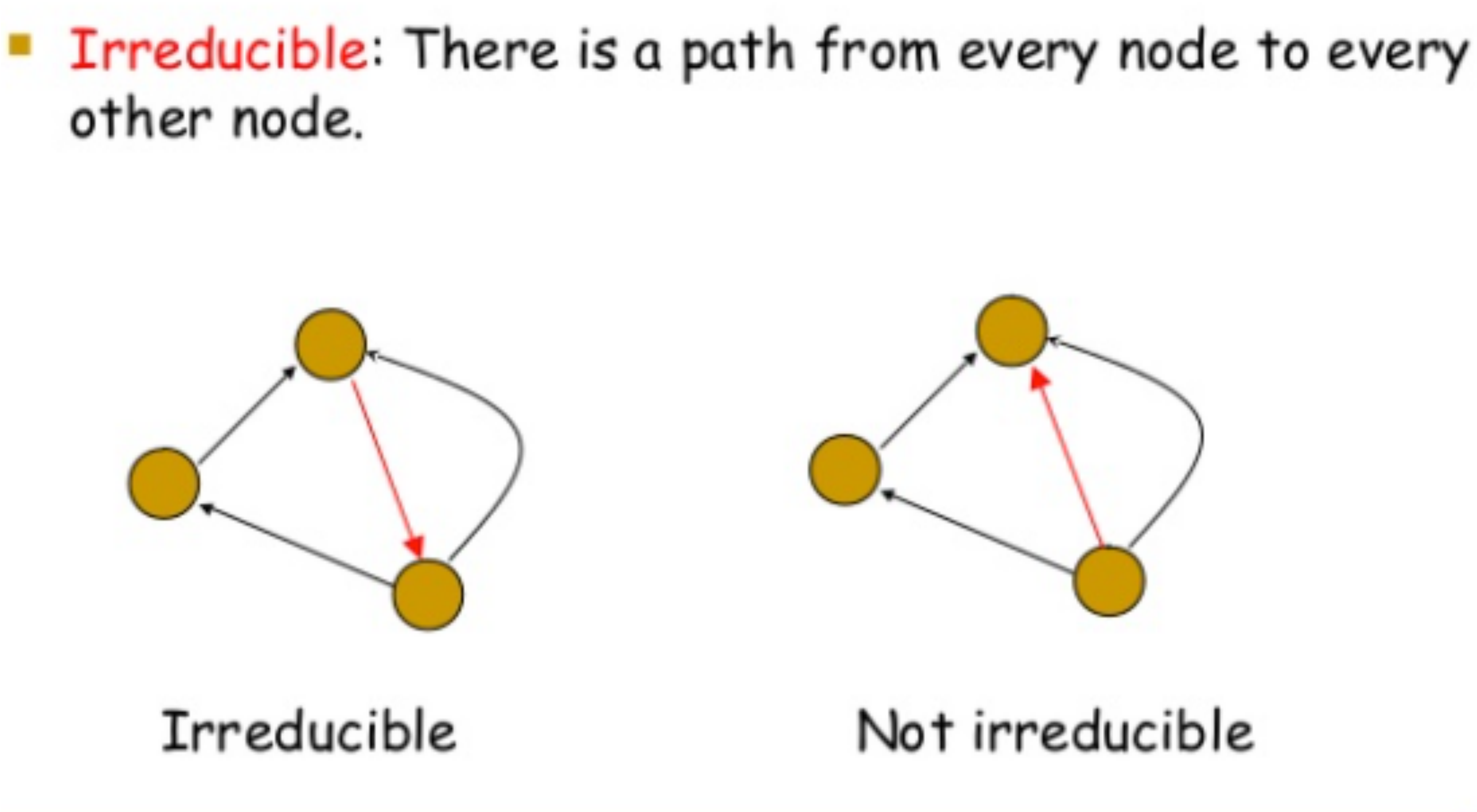
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

- **Example:**

Iteration:		0,	1,	2,	3…
r_a	=	1	0	0	0
r_b		0	1	0	0

Dead-ends are a problem because the matrix is not column stochastic so our initial assumptions are not met.

Random walk has stationary distribution when the graph is irreducible and aperiodic



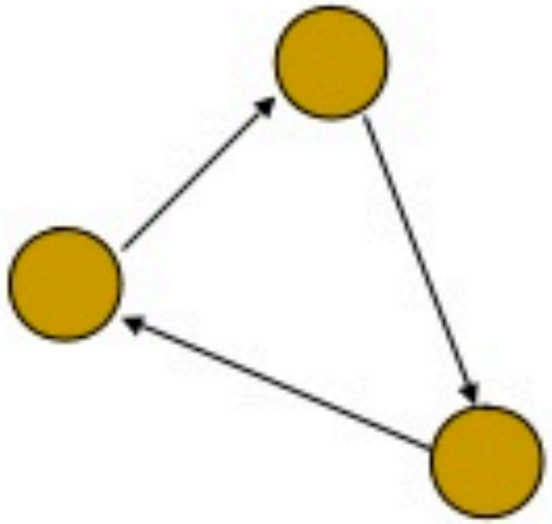
- Irreducible: There is a path from every node to every other node.

Irreducible

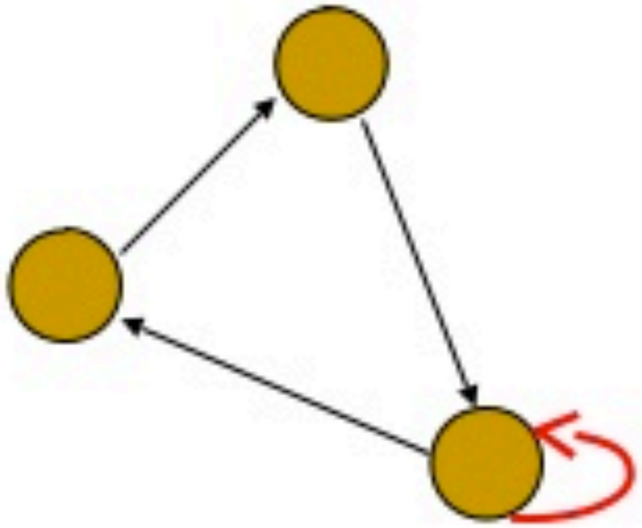
Not irreducible

Source: Random walk on graphs: an overview

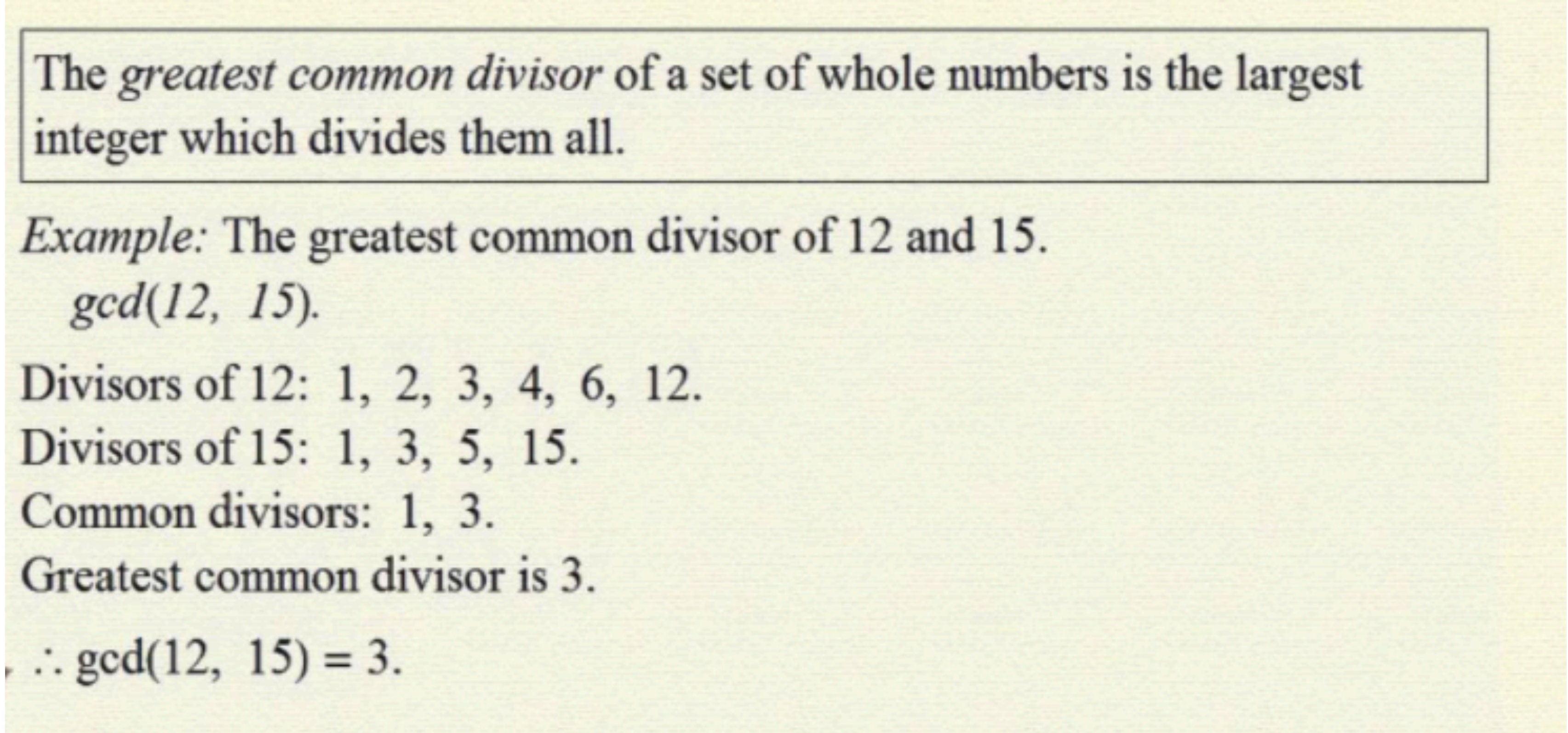
- Aperiodic: The GCD of all cycle lengths is *1*. The GCD is also called period.



Periodicity is 3



Aperiodic



The *greatest common divisor* of a set of whole numbers is the largest integer which divides them all.

Example: The greatest common divisor of 12 and 15.

$gcd(12,\ 15)$.

Divisors of 12: 1, 2, 3, 4, 6, 12.

Divisors of 15: 1, 3, 5, 15.

Common divisors: 1, 3.

Greatest common divisor is 3.

$\therefore$ gcd(12, 15) = 3.

Source: Random walk on graphs: an overview

Solution: jump to a random node

At each time step, the random surfer has two options

- With prob. β, follow a link at random
- With prob. $1-\beta$, jump to a random page
- Common values for β are in the range 0.8 to 0.9

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix} = \beta \begin{bmatrix} 1/d_1 , & 0 , & \ldots \\ 1/d_1 , & 1/d_2 , & \ldots \\ \vdots & \vdots & \\ 0 , & 1/d_2 , & \ldots \\ \vdots & \vdots & \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \\ \vdots \\ r_N \end{bmatrix} + (1-\beta) \begin{bmatrix} 1/N \\ 1/N \\ \vdots \\ 1/N \\ \vdots \\ 1/N \end{bmatrix}$$

Difference from random walk

Random walk

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \end{bmatrix} = \beta \begin{bmatrix} 1/d_1 , & 0 , & \dots \\ 1/d_1 , & 1/d_2 , & \dots \\ \vdots & \vdots & \\ 0 , & 1/d_2 , & \dots \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \end{bmatrix} + (1-\beta) \begin{bmatrix} 1/N \\ 1/N \\ \vdots \\ 1/N \end{bmatrix}$$

Random walk with restart

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \end{bmatrix} = \beta \begin{bmatrix} 1/d_1 , & 0 , & \dots \\ 1/d_1 , & 1/d_2 , & \dots \\ \vdots & \vdots & \\ 0 , & 1/d_2 , & \dots \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_j \end{bmatrix} + (1-\beta) \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_j \end{bmatrix} \Rightarrow \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

Matrix representation v.s. vector representation

Input: adjacency matrix of the hierarchy (undirected)

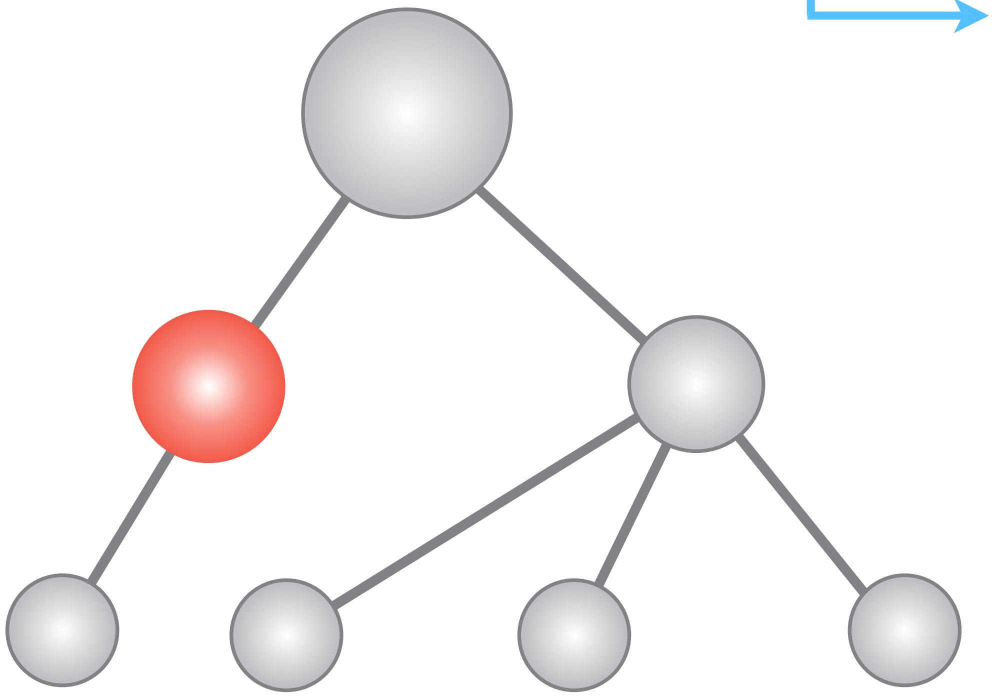
Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^{t}$.

Output: equilibrium distribution s_i starting from node i

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^{t}\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node



Node i and node j are topologically similar if s_i is similar to s_j

Random walk with restart

- Random walk with restart is the same as random walk other than the fact that jumps are back to one of a given set of starting vertices.

- In a way, the walk in Random walk with restart is biased towards (or personalized for) this set of starting vertices and is more localized compared to the random walk.

Functions of random walk

1. Smooth the whole graphs

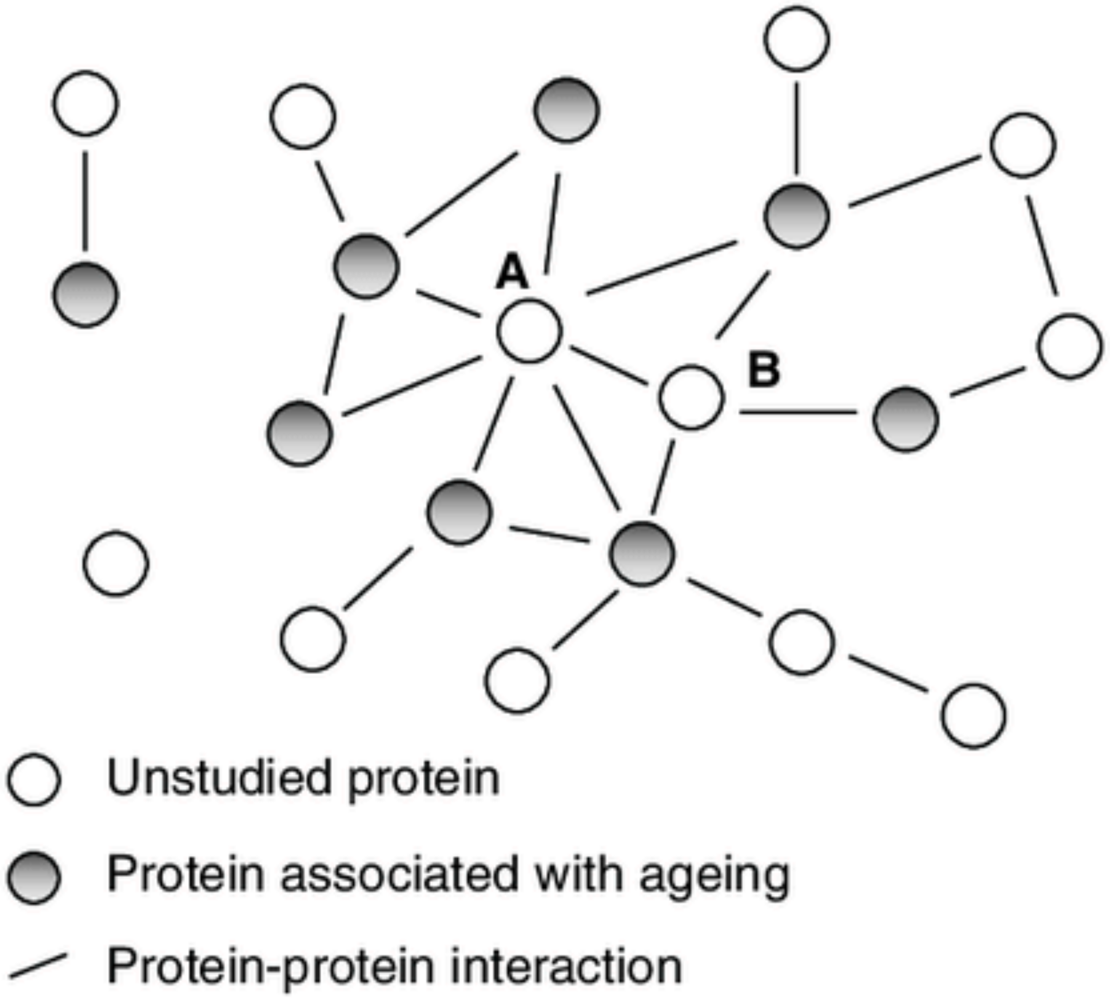
2. Assign importance score

3. Quantify the distance of two nodes

4. Want to integrate information beyond the neighbors

Guilt-by-association rule

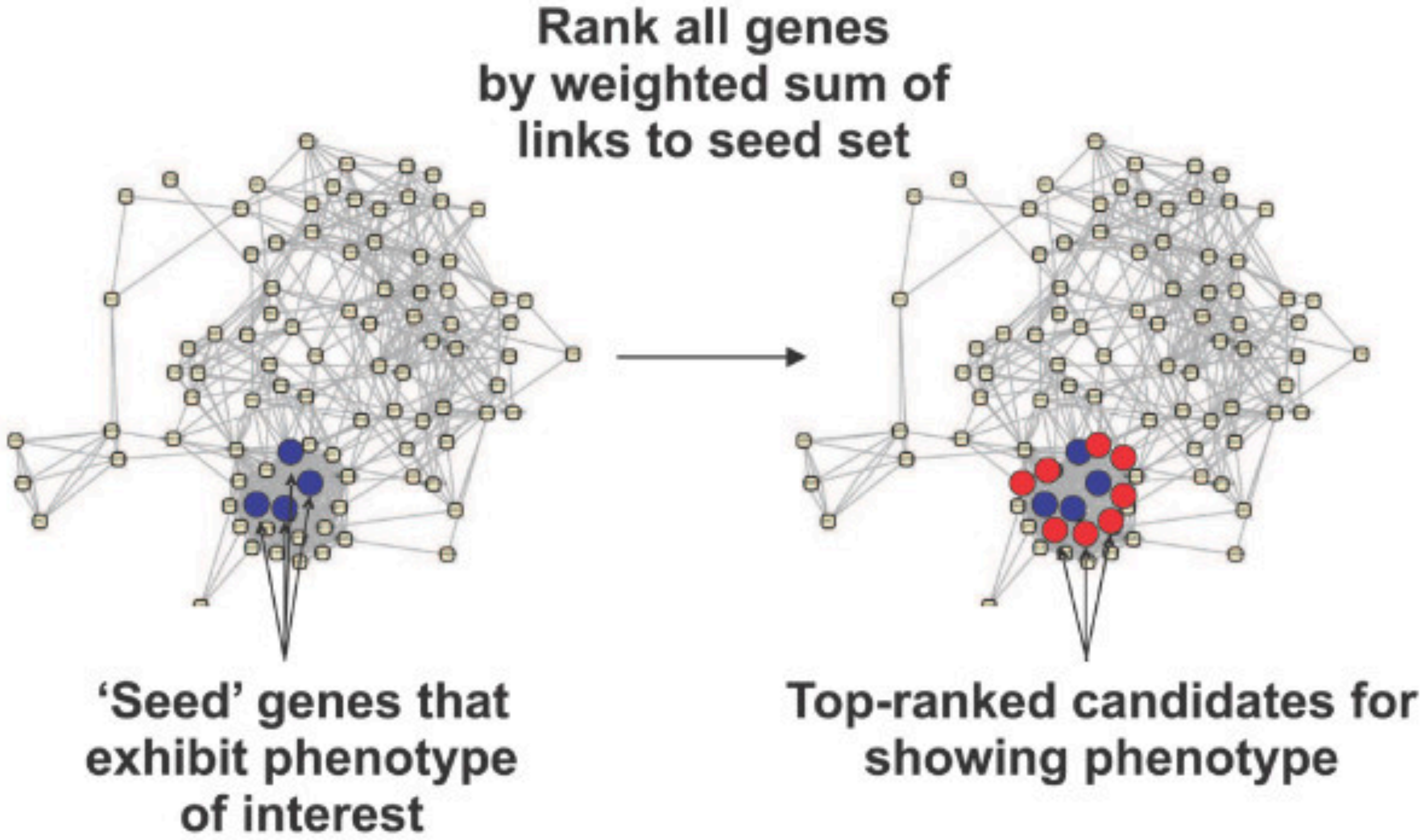
- Assign a label to a node using its neighbor's labels



Source: Aging research in the post-genome era: New technologies for an old problem

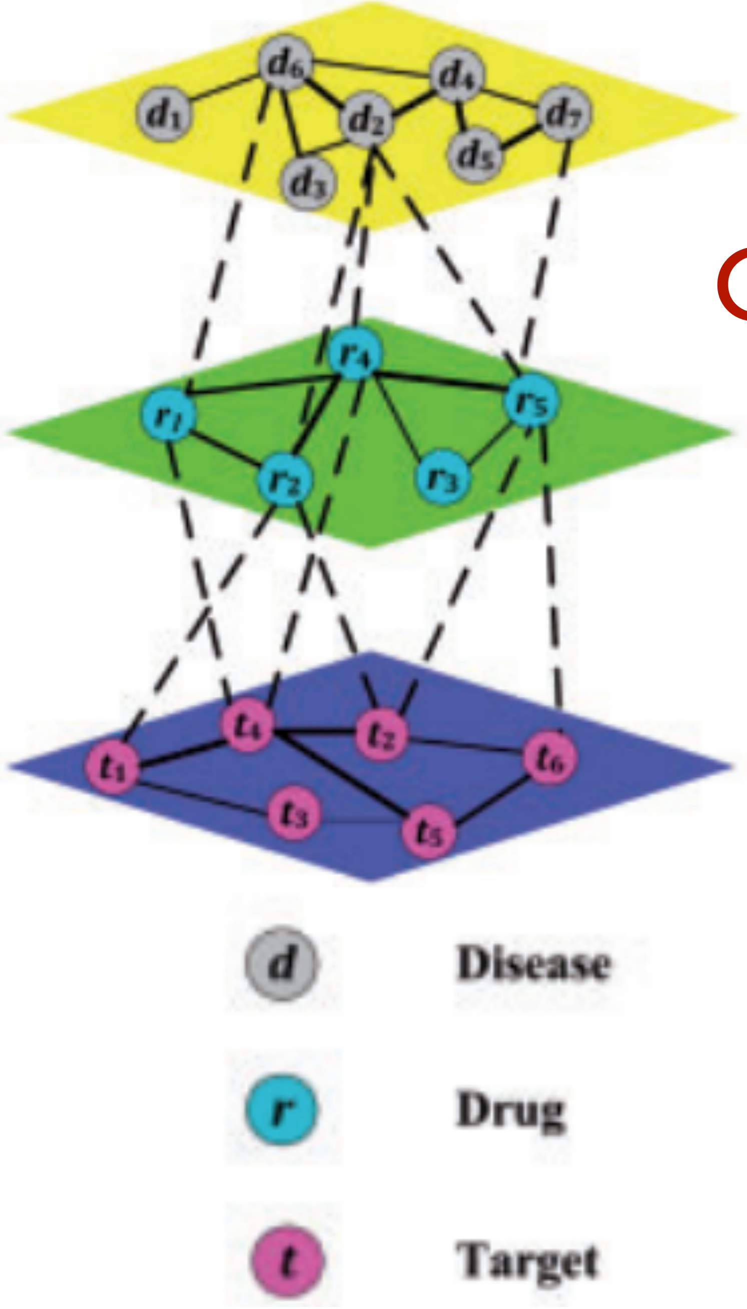
Guilt-by-association rule

- Fast and scalable to large networks
- But only utilize information of 1st-order neighbors

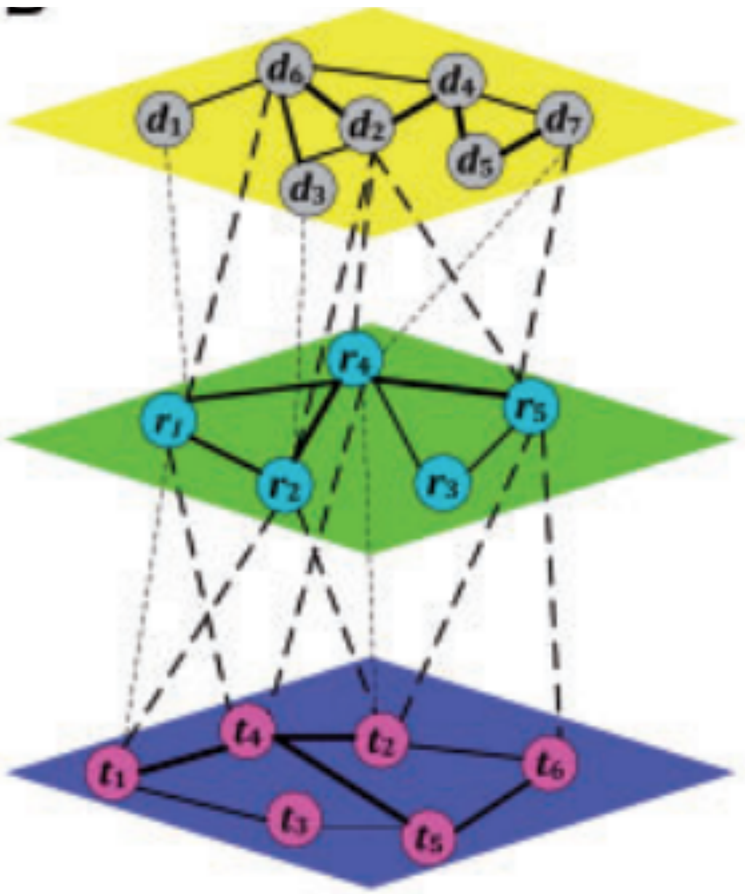


Source: Broad network-based predictability of Saccharomyces cerevisiae gene loss-of-function phenotypes

Drug target identification using guilt-by-association



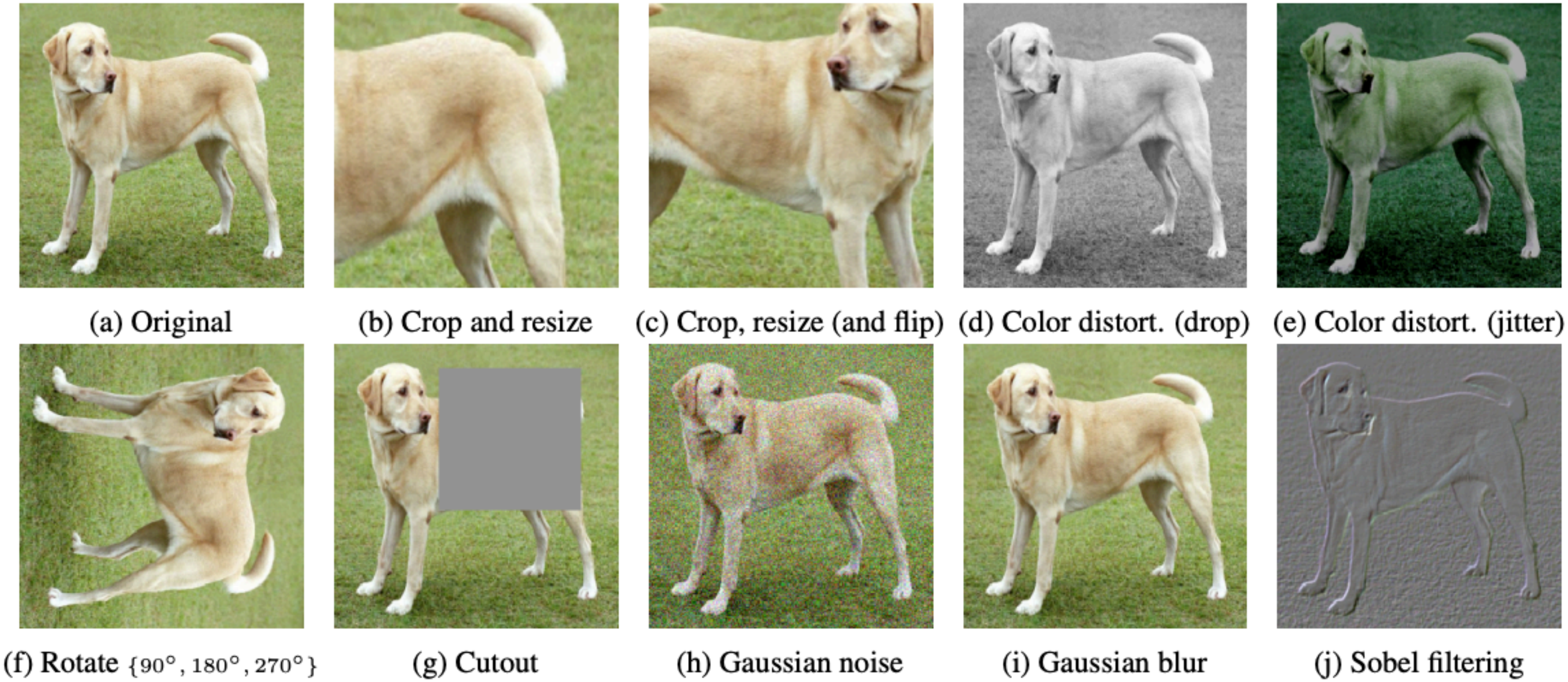
Guilt-by-association rule



Source: Drug repositioning by integrating target information through a heterogeneous network model

Network embedding and graph neural network

SimCLR: Contrastive Learning using data augmentation



(a) Original

(b) Crop and resize

(c) Crop, resize (and flip)

(d) Color distort. (drop)

(e) Color distort. (jitter)

(f) Rotate $\{90^\circ, 180^\circ, 270^\circ\}$

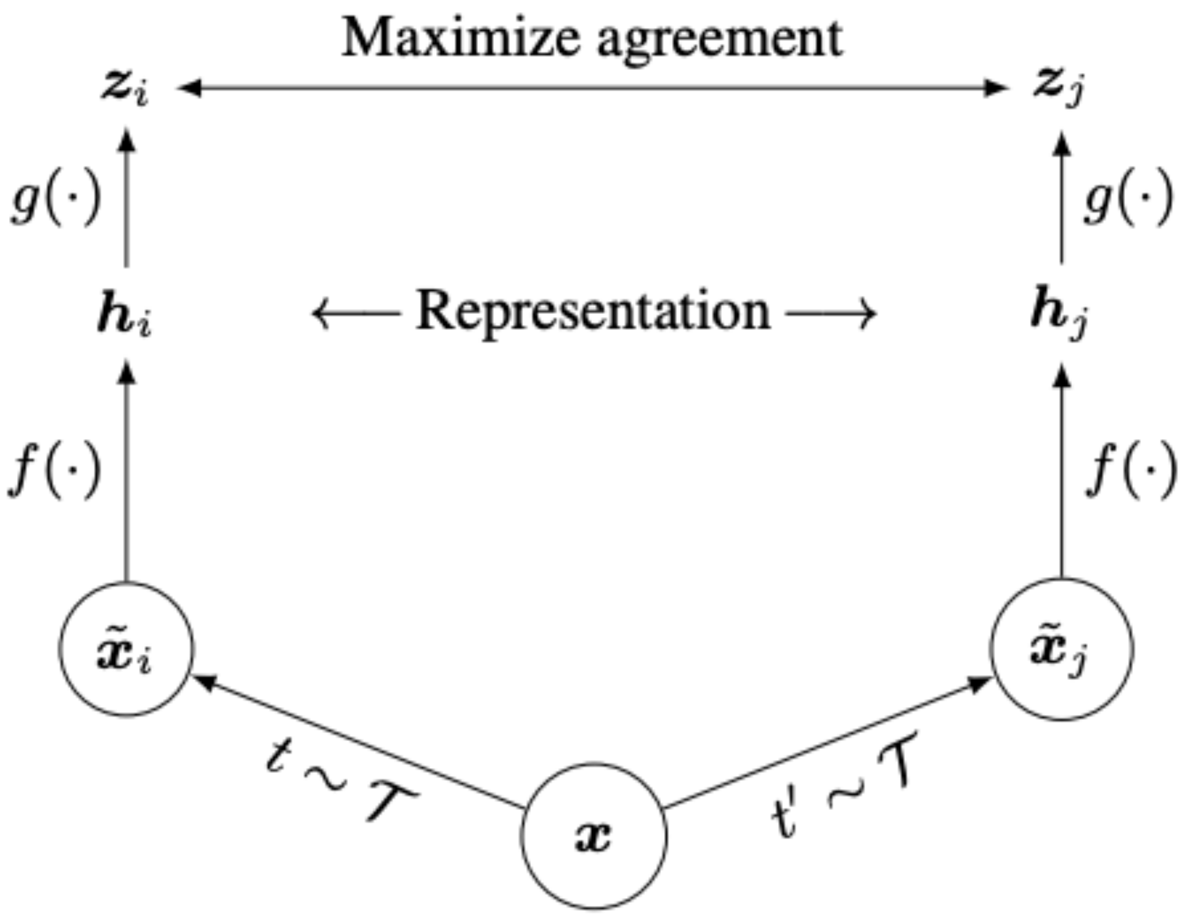
(g) Cutout

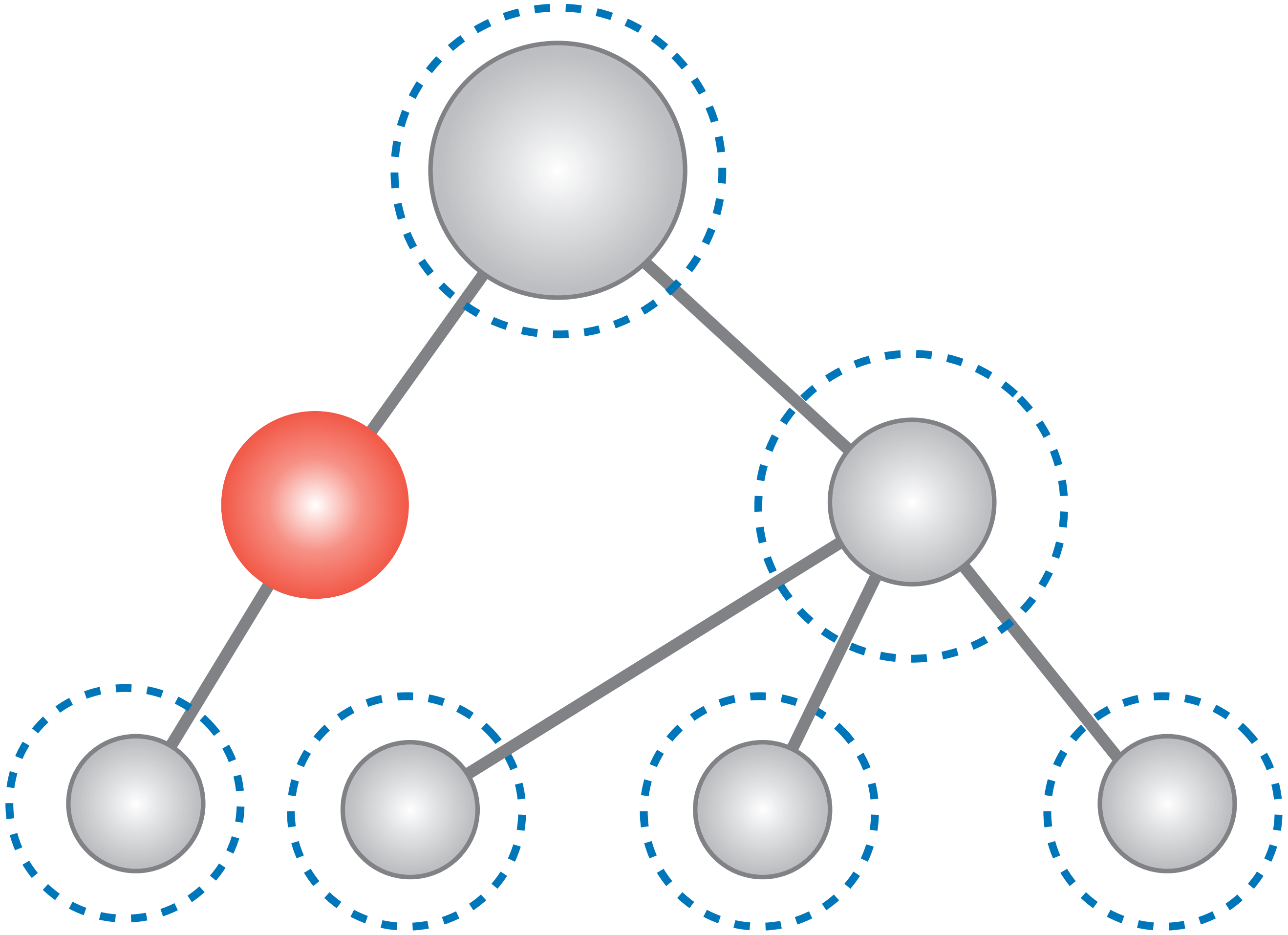
(h) Gaussian noise

(i) Gaussian blur

(j) Sobel filtering

Using network is like data augmentation



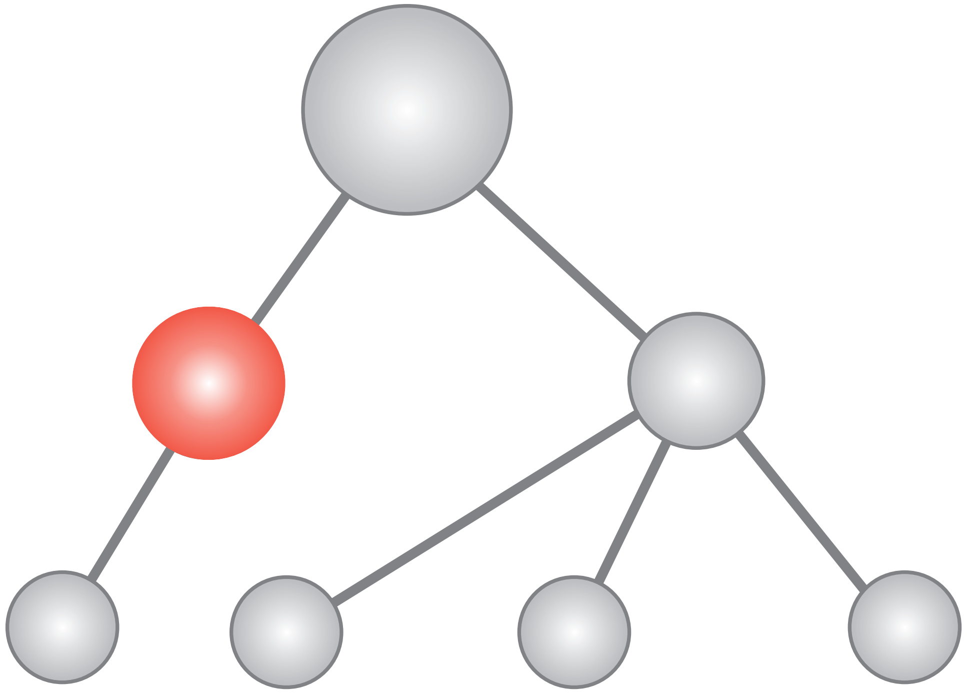


The Math:

use diffusion model on each node to calculate topological similarity

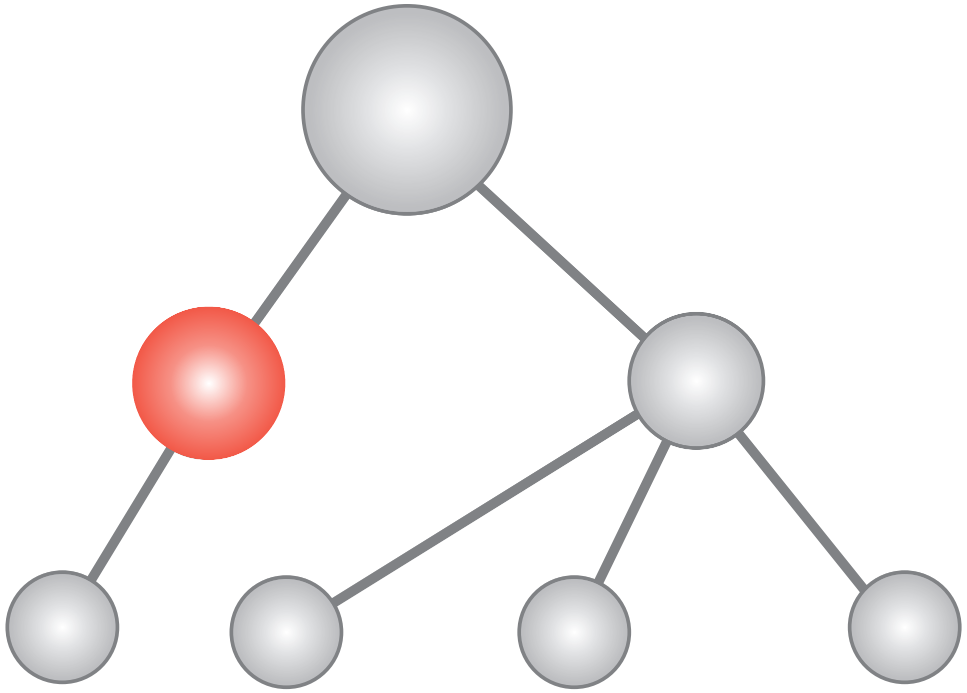
Input: adjacency matrix
of the hierarchy (undirected)

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t \boldsymbol{B} + p_r \boldsymbol{I}$$



The Math:

use diffusion model on each node to calculate topological similarity



Input: adjacency matrix of the hierarchy (undirected)

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node

Node i and node j are topologically similar if s_i is similar to s_j

The Math:

use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

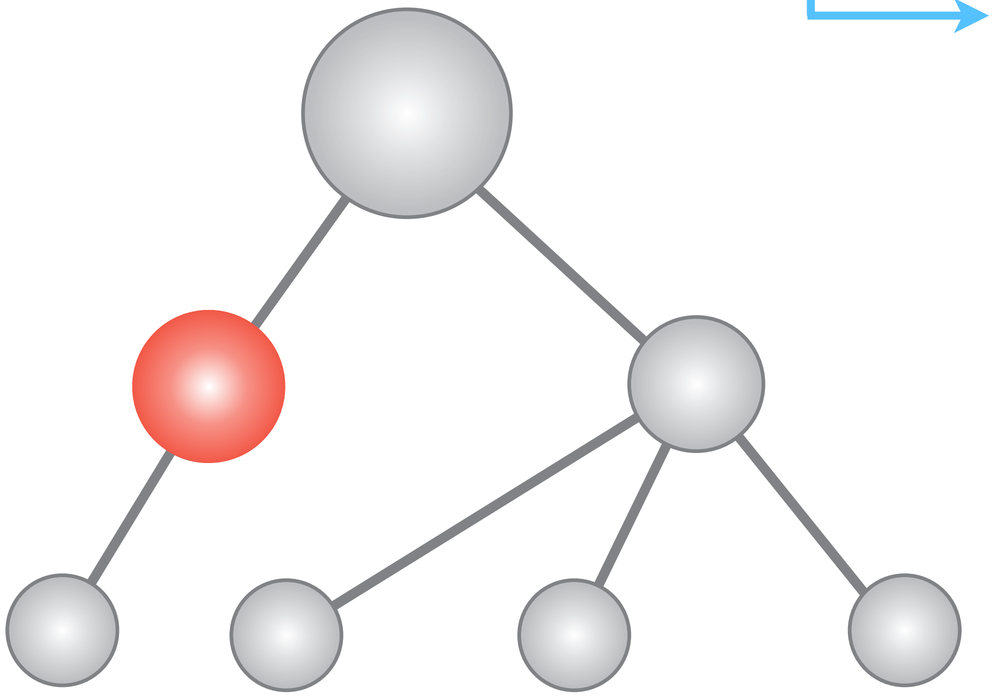
Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^t$.

Output: equilibrium distribution s_i starting from node i

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node



Node i and node j are topologically similar if s_i is similar to s_j

The Math:

use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

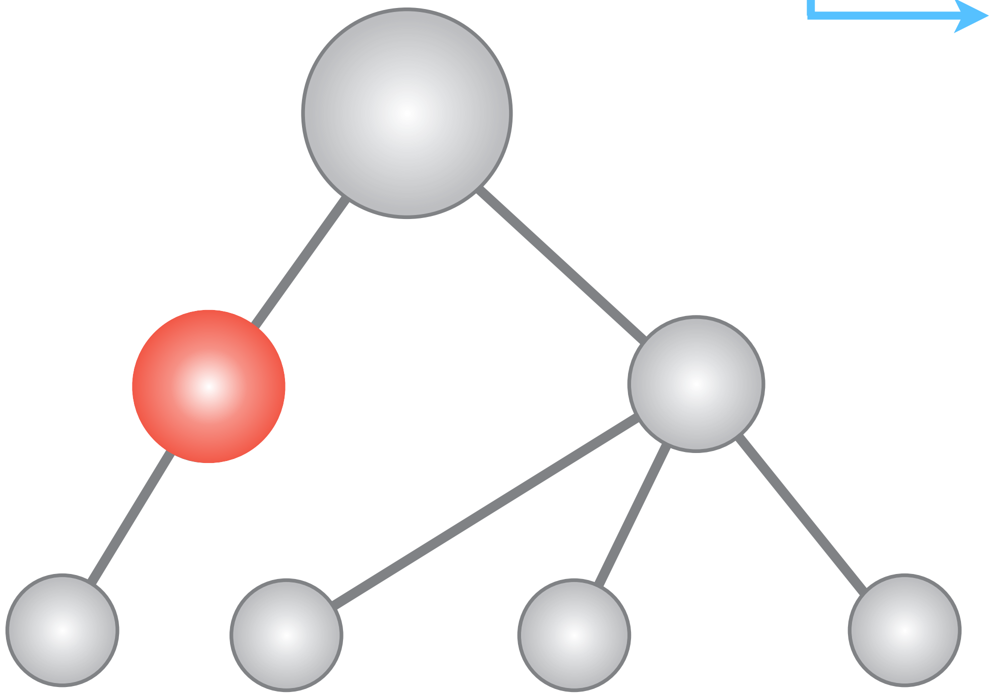
Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^t$.

Output: equilibrium distribution s_i starting from node i

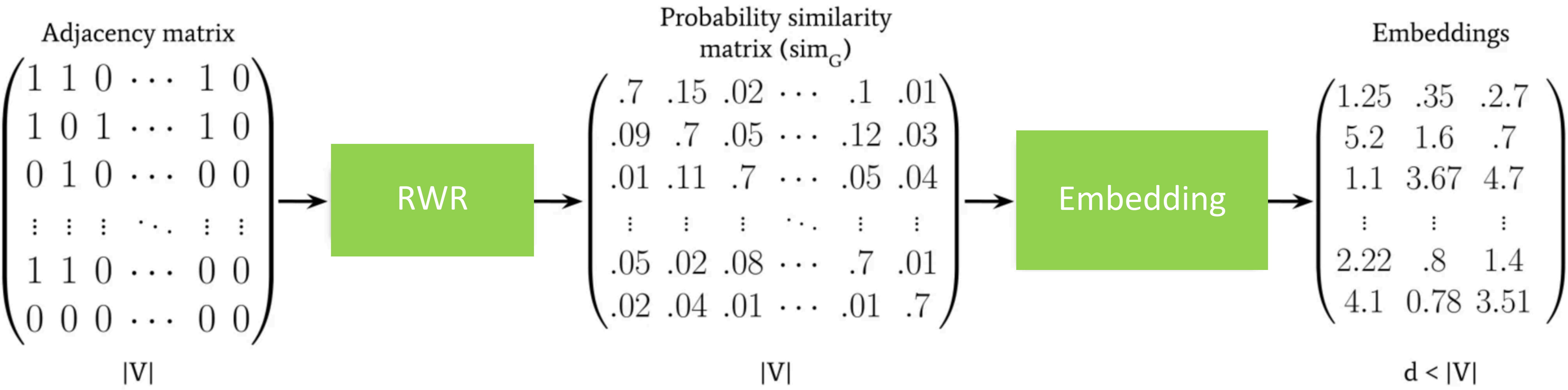
$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^t\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node



From advanced matrix to random walk probability matrix

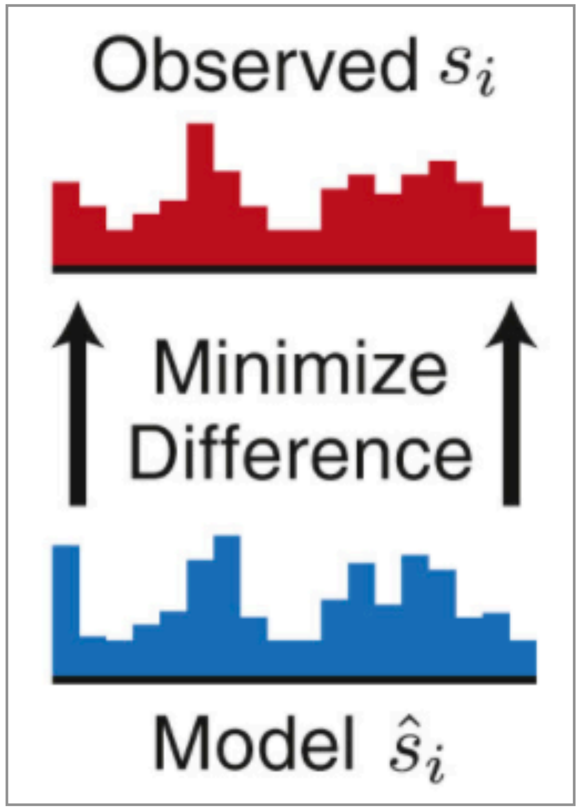


$$\begin{pmatrix} 1 & 1 & 0 & \cdots & 1 & 0 \\ 1 & 0 & 1 & \cdots & 1 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \rightarrow \boxed{\text{RWR}} \rightarrow \begin{pmatrix} .7 & .15 & .02 & \cdots & .1 & .01 \\ .09 & .7 & .05 & \cdots & .12 & .03 \\ .01 & .11 & .7 & \cdots & .05 & .04 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ .05 & .02 & .08 & \cdots & .7 & .01 \\ .02 & .04 & .01 & \cdots & .01 & .7 \end{pmatrix} \rightarrow \boxed{\text{Embedding}} \rightarrow \begin{pmatrix} 1.25 & .35 & .2.7 \\ 5.2 & 1.6 & .7 \\ 1.1 & 3.67 & 4.7 \\ \vdots & \vdots & \vdots \\ 2.22 & .8 & 1.4 \\ 4.1 & 0.78 & 3.51 \end{pmatrix}$$

Image adapted from MultiVERSE: a multiplex and multiplex-heterogeneous network embedding approach

Network embedding: decomposing diffusion matrix

Optimization goal: find class embedding $\{\boldsymbol{x}_i\}$ and context embedding $\{\boldsymbol{u}_i\}$ so that $\{\hat{\boldsymbol{s}}_i\}$ is close to $\{\mathbf{s}_i\}$.

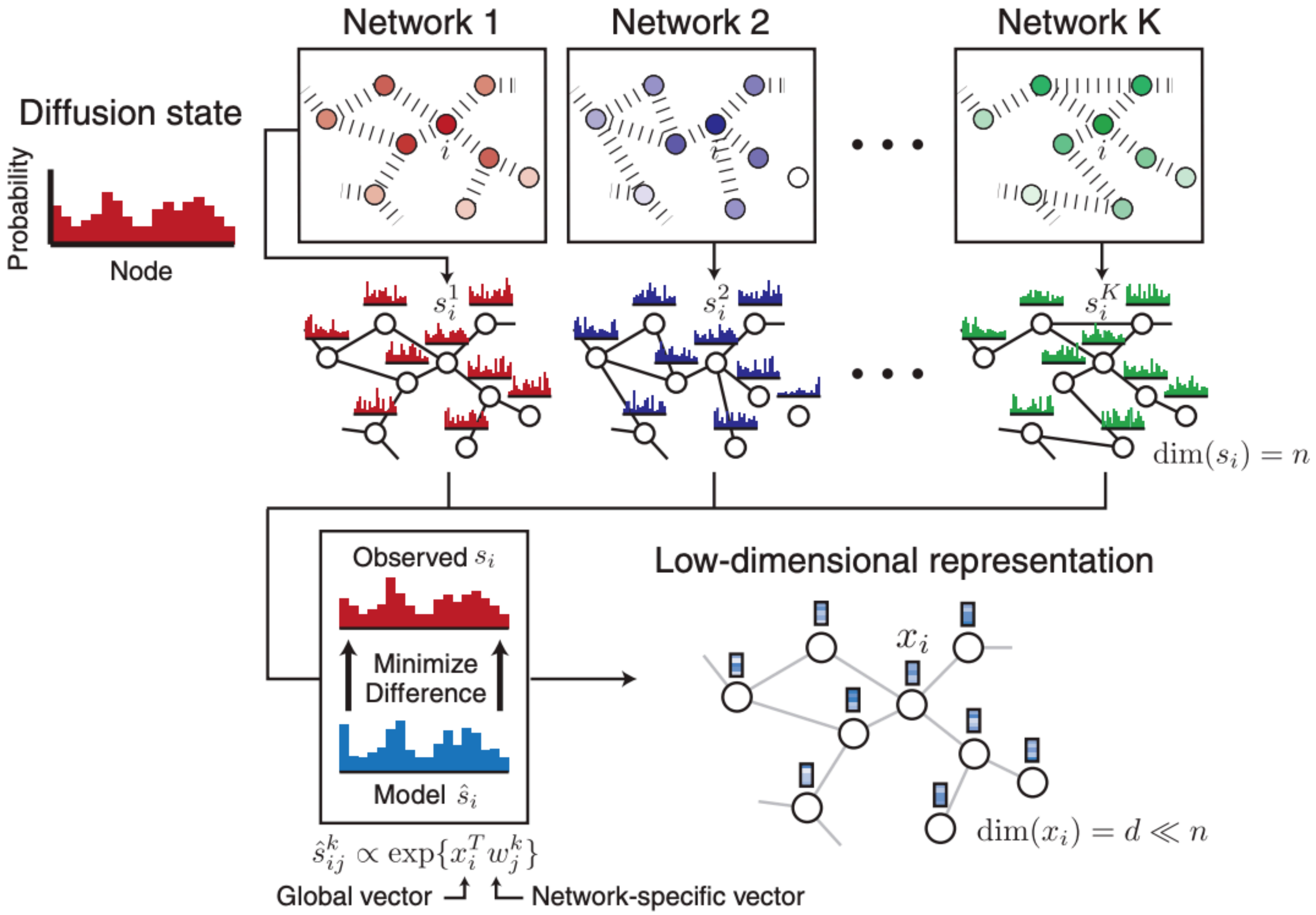


$$\hat{s}_{ij} = \frac{e^{\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{u}_j}}{\sum_k e^{\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{u}_k}} \approx e^{\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{u}_j}$$

Drop the partition function for fast approximation

Loss function: $\min_{\boldsymbol{x},\boldsymbol{u}} C(\boldsymbol{S}, \hat{\mathbf{S}}) = \sum_{i=1}^{n}\sum_{j=1}^{n}(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{u}_j - \log s_{ij})^2$

Network embedding for network integration



Compact Integration of Multi-Network Topology for Functional Analysis of Genes

Recap

- Guilt-by-association
  - Only use 1st-order neighbors' information
  - A good baseline that works very well in many applications.
- Random walk
  - Find the most important node. Consider all nodes in the graph.
  - Might not converge due to dead end issue
- Random walk with restart
  - Solved dead end issue.
  - Does not have node features. Only one importance score for each node.
- Network embedding
  - Reduce RWR matrix to low-dimensional
  - Better for large and noisy network

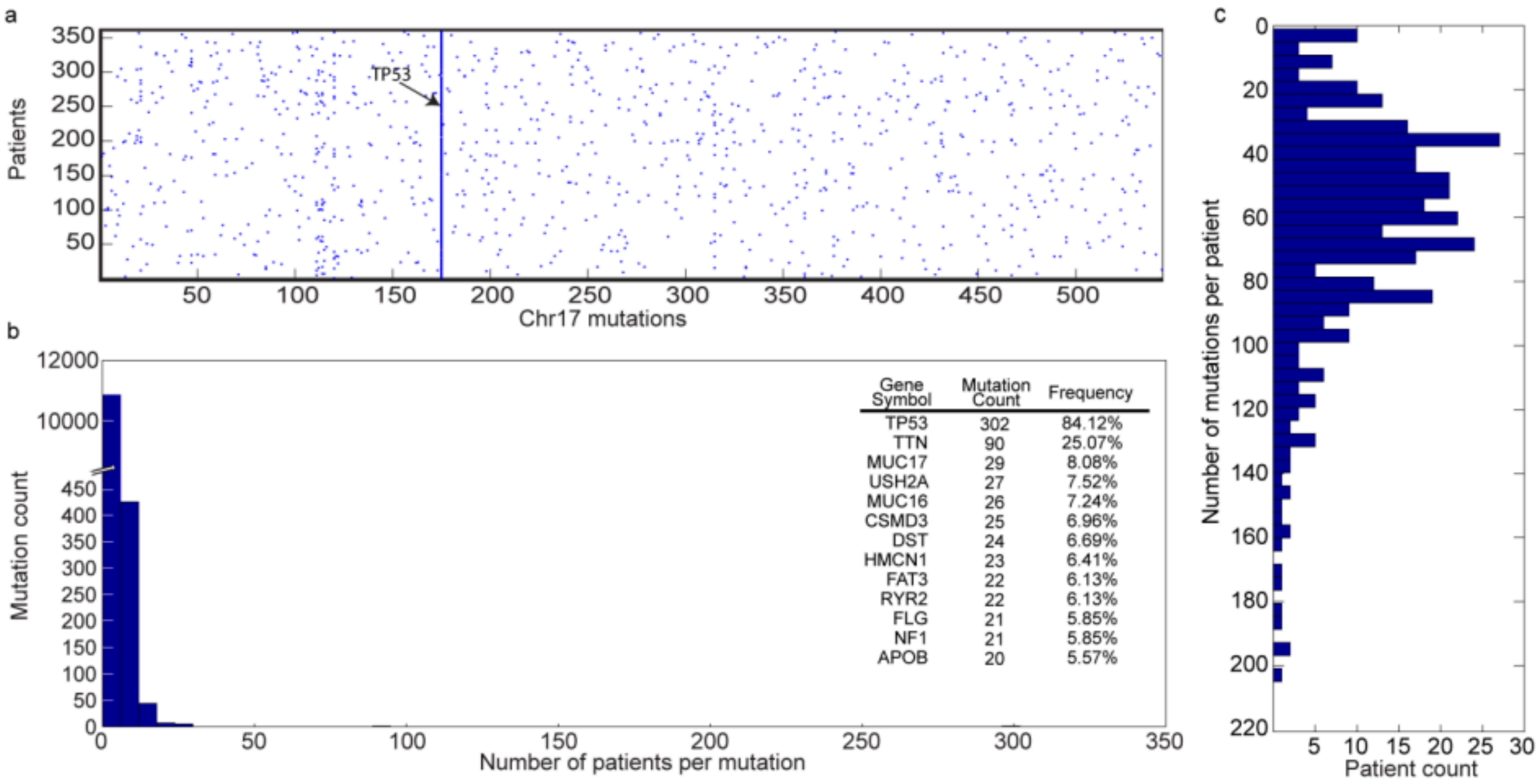
Goal

- Tumor stratification: to divide a heterogeneous population into clinically and biologically meaningful subtypes based on molecular profiles

Somatic mutation profile

- Compare the mutations of tumors
- Sparse

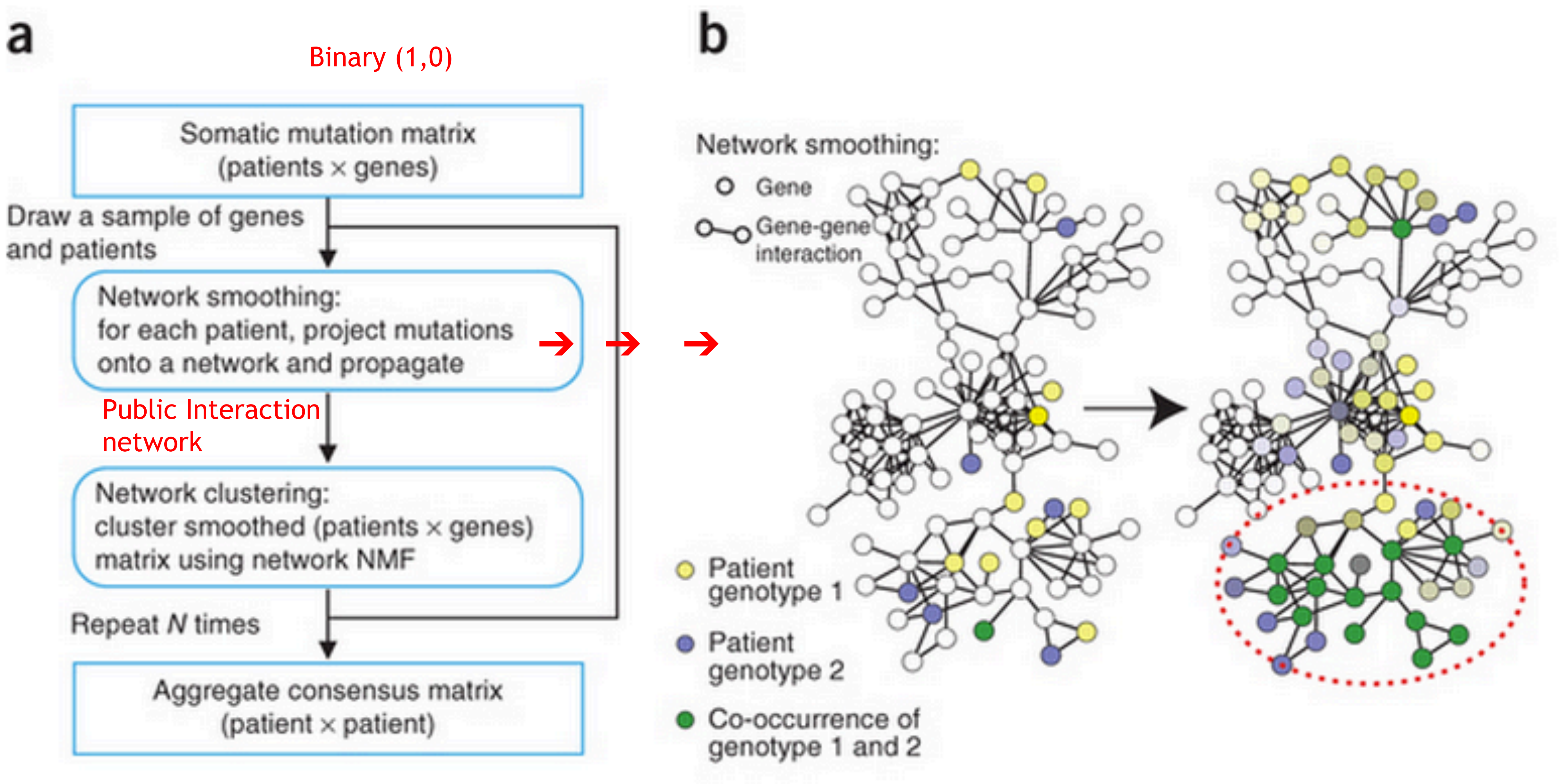
Supplementary Figure 1



Gene Symbol	Mutation Count	Frequency
TP53	302	84.12%
TTN	90	25.07%
MUC17	29	8.08%
USH2A	27	7.52%
MUC16	26	7.24%
CSMD3	25	6.96%
DST	24	6.69%
HMCN1	23	6.41%
FAT3	22	6.13%
RYR2	22	6.13%
FLG	21	5.85%
NF1	21	5.85%
APOB	20	5.57%

Source: Network-based stratification of tumor mutations

Overview of network-based stratification



Binary (1,0)

Somatic mutation matrix (patients × genes)

Draw a sample of genes and patients

Network smoothing: for each patient, project mutations onto a network and propagate

Public Interaction network

Network clustering: cluster smoothed (patients × genes) matrix using network NMF

Repeat *N* times

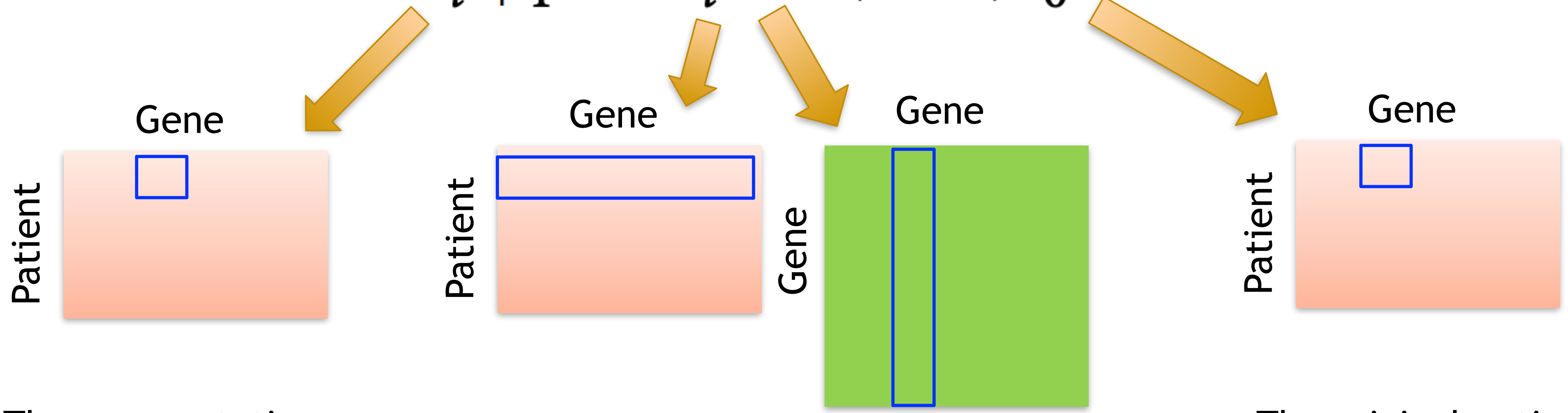
Aggregate consensus matrix (patient × patient)

Source: Network-based stratification of tumor mutations

Algorithm

Performing random walk with restart for each patient

$$F_{t+1} = \alpha F_t A + (1-\alpha) F_0$$



The new mutation status for patient 1, gene j

First consider the mutations for all the genes of patient 1

Then consider all the neighbors of gene j

The original patient mutation matrix

Source: Network-based stratification of tumor mutations

The Math:

use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

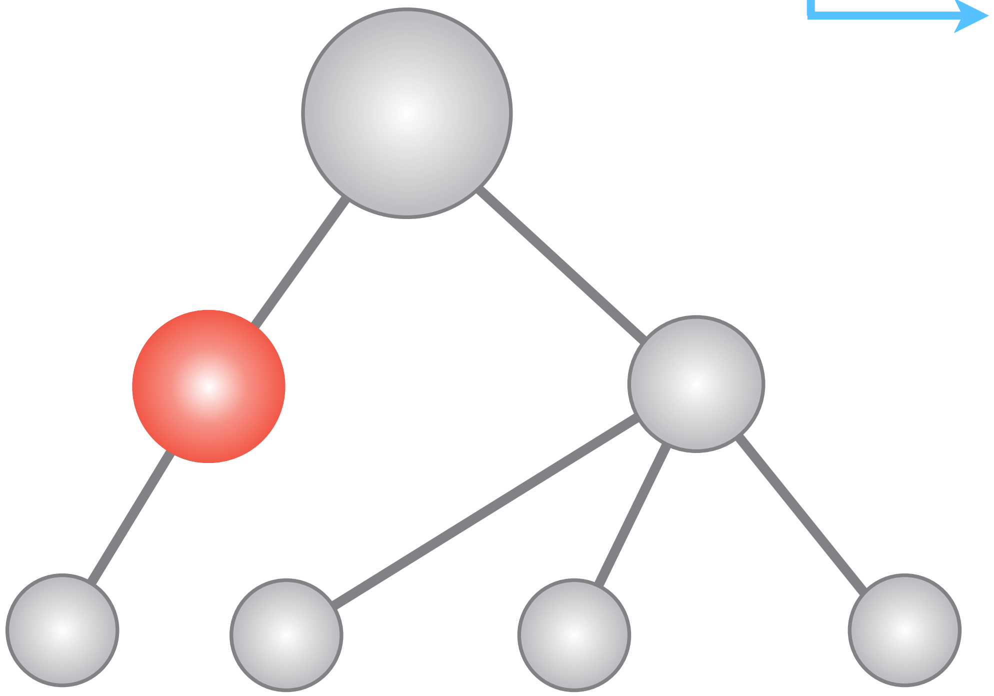
Iterate $t = \{0,1,\ldots\}$ until $\boldsymbol{S}^{t+1} \approx \boldsymbol{S}^{t}$.

Output: equilibrium distribution s_i starting from node i

$$\boldsymbol{S}^{t+1} = (1 - p_r)\boldsymbol{S}^{t}\boldsymbol{B} + p_r\boldsymbol{I}$$

Random walk to neighbors

Restart to the original node



Network smoothing

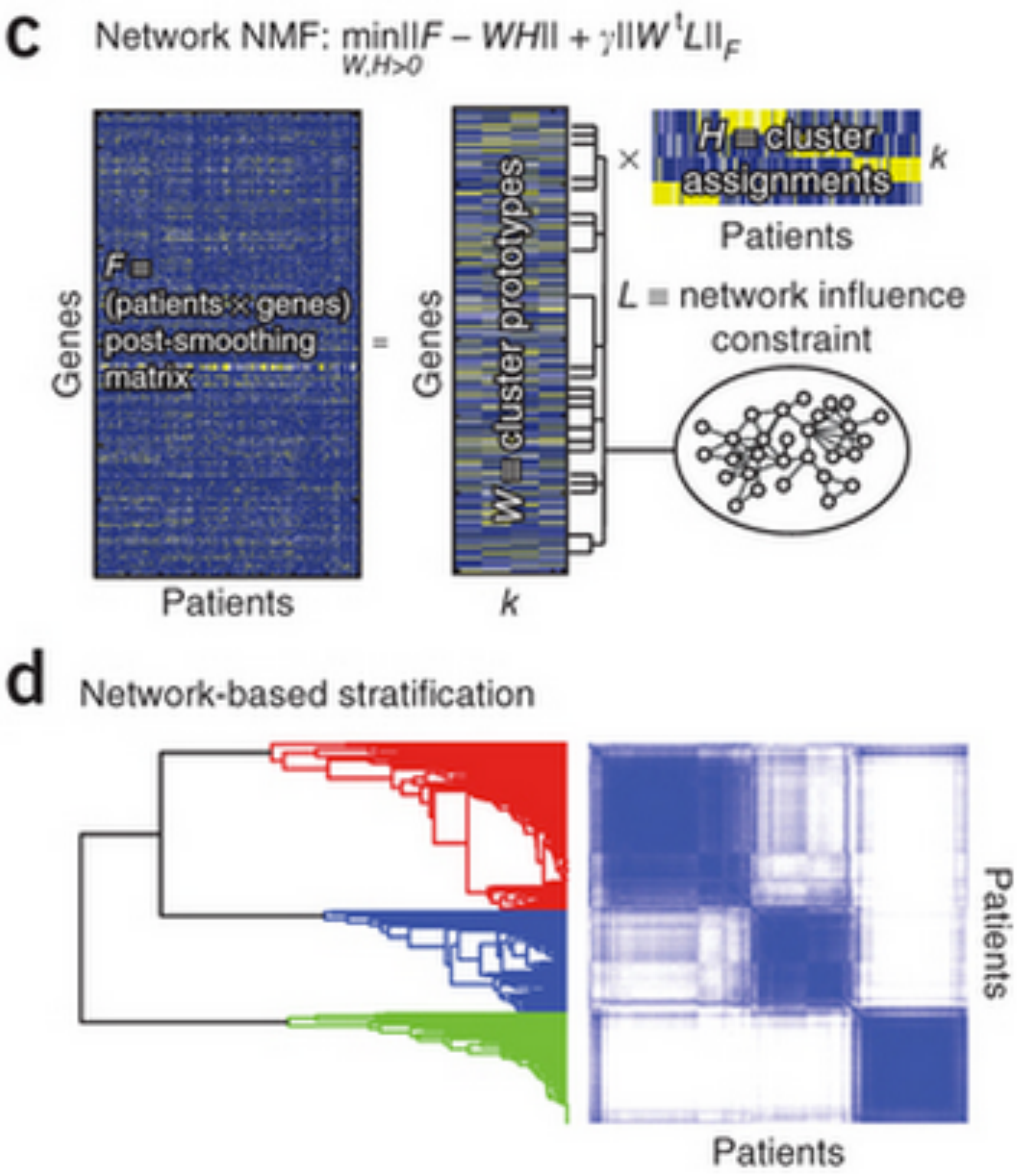
- $F_{t+1} = \alpha F_t A + (1-\alpha) F_0$

F_0: patients * genes matrix

A: adjacency matrix of the gene interaction network (STRING, HumanNet and PathwayCommons)

α: tuning factor that determines how far a mutation signal can diffuse

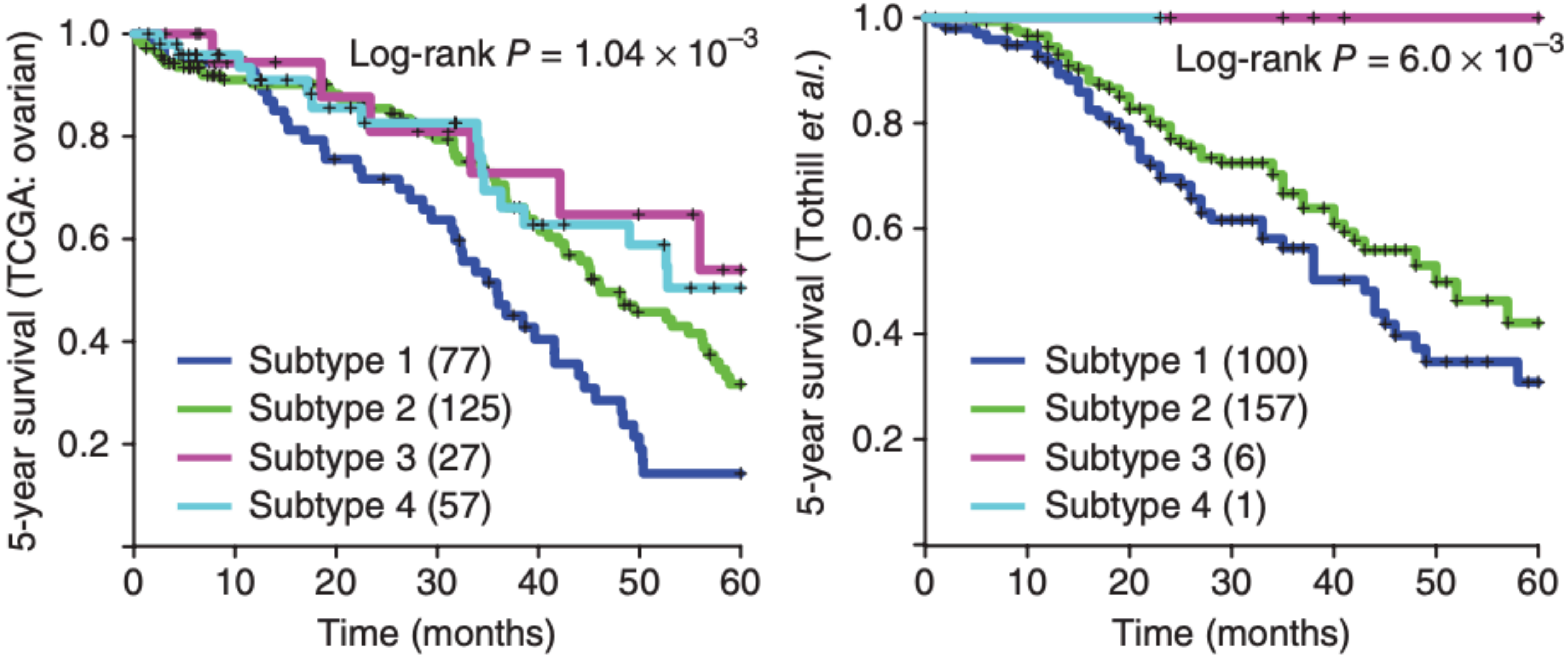
Source: Network-based stratification of tumor mutations


c
Network NMF: $\min_{W,H>0} \|F - WH\| + \gamma \|W^{t}L\|_{F}$
F = (patients × genes) post-smoothing matrix
Genes
Patients
=
Genes
W = cluster prototypes
k
×
H = cluster assignments
k
Patients
L = network influence constraint
d
Network-based stratification
Patients
Patients

Source: Network-based stratification of tumor mutations

Biological significance

Different clusters of patients have different survival rates !



Source: Network-based stratification of tumor mutations

How to incorporate node features into RWR

- The restart probability on each node is a vector now
- $F_{t+1} = \alpha F_t A + (1-\alpha)F_0$

F_0: patients * genes matrix

A: normalized adjacency matrix of the gene interaction network

α: restart probability

Question: what is the limitation of incorporating node features in this way?

How to incorporate node features into RWR

- The restart probability on each node is a vector now
- $F_{t+1} = \alpha F_t A + (1-\alpha)F_0$

F_0: patients * genes matrix

A: normalized adjacency matrix of the gene interaction network

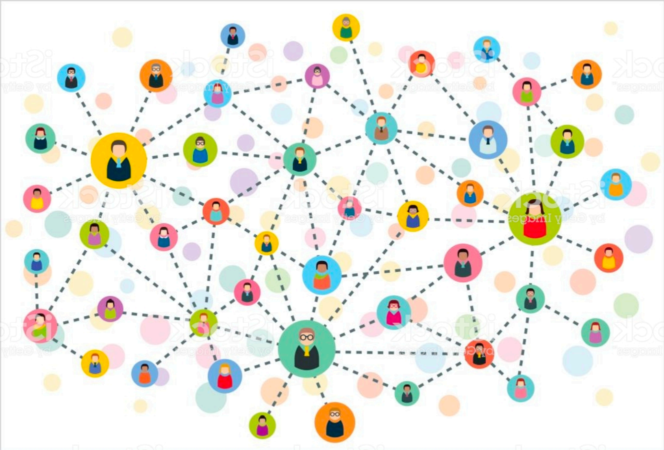
α: restart probability

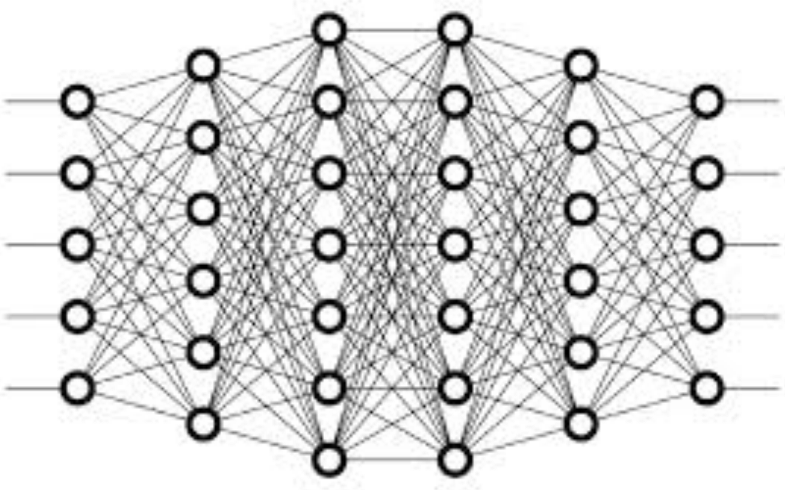
Question: what is the limitation of incorporating node features in this way?

Different node features are modeled independently.

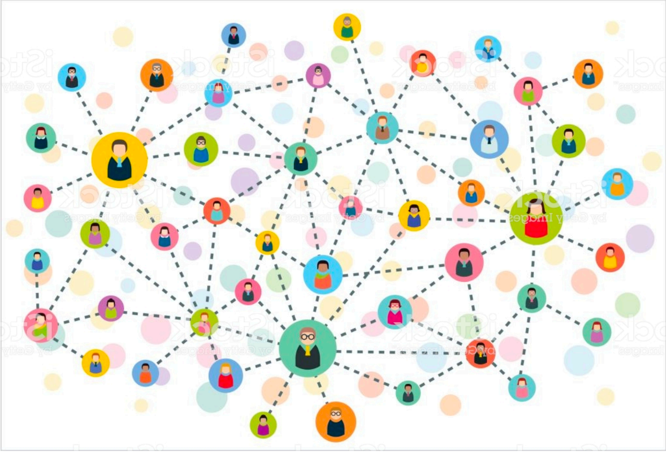
This is why we need graph neural network which will model the dependency between features

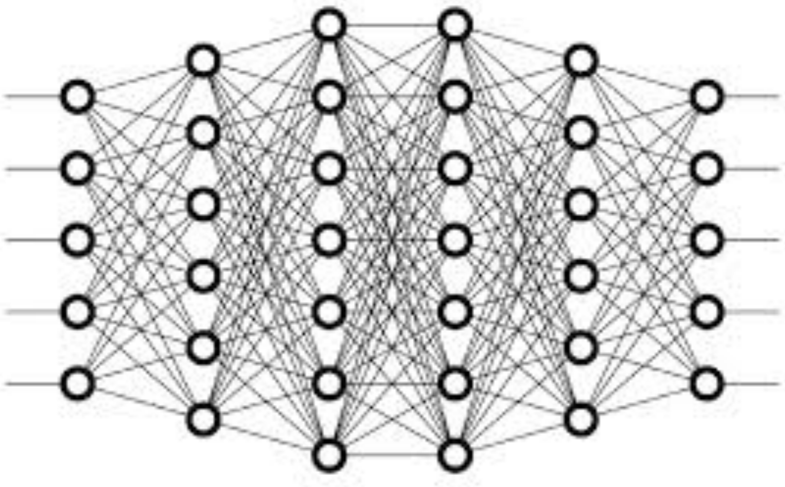
Deep Learning on Graphs

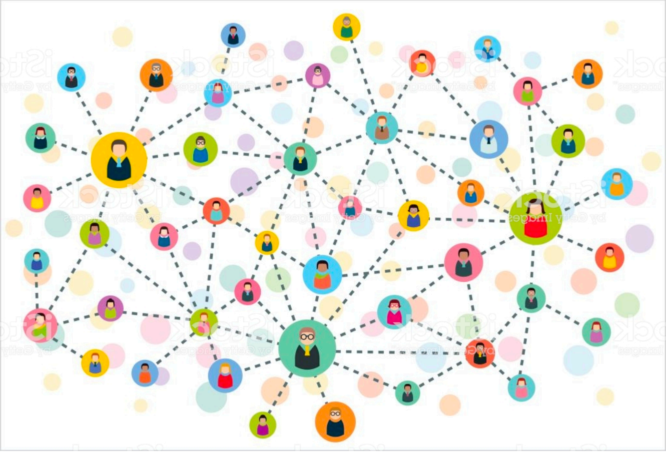




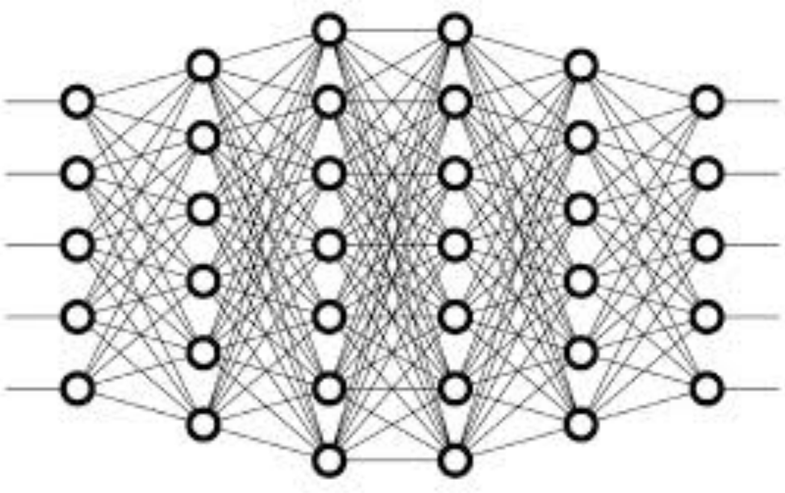
Labels

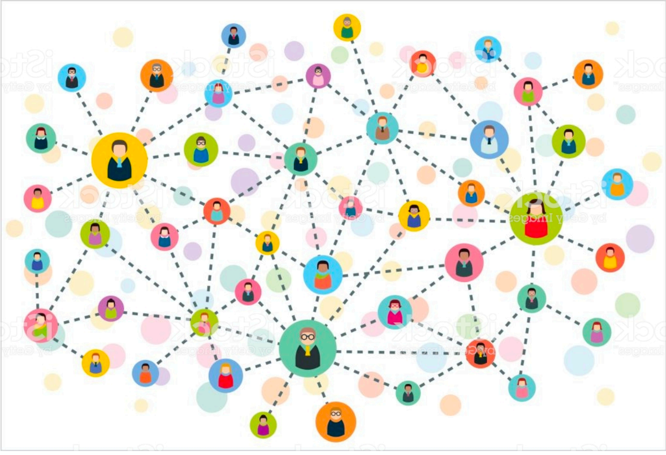






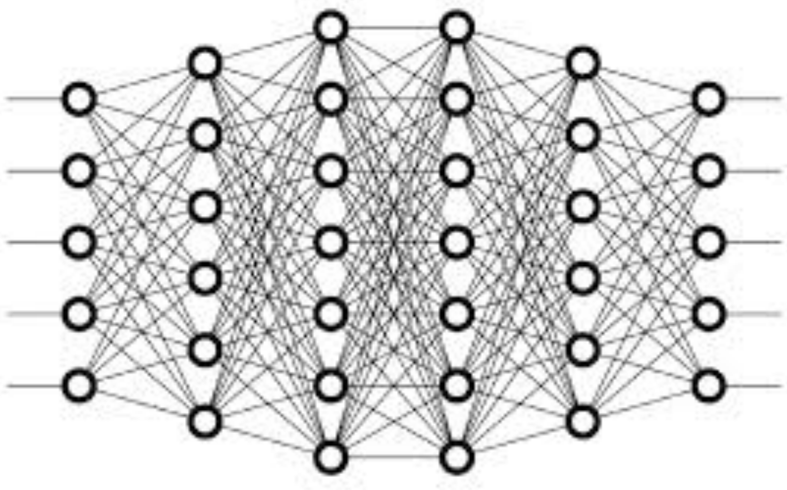
Random value

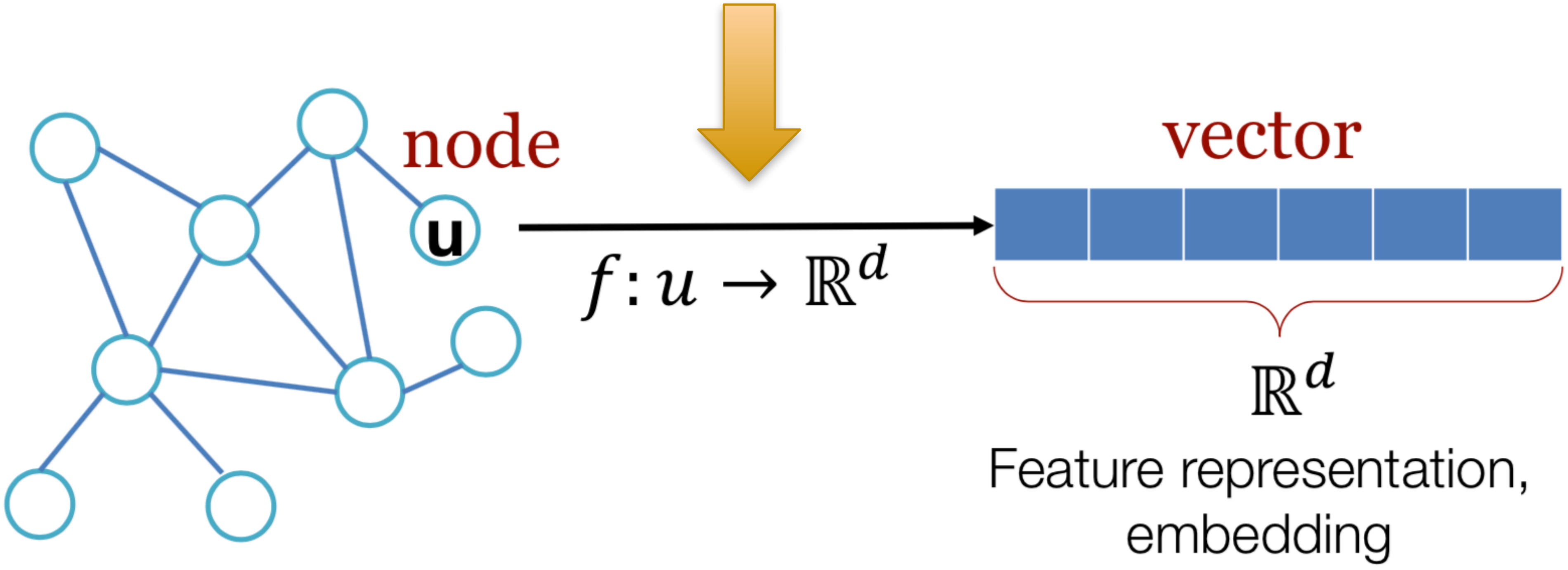




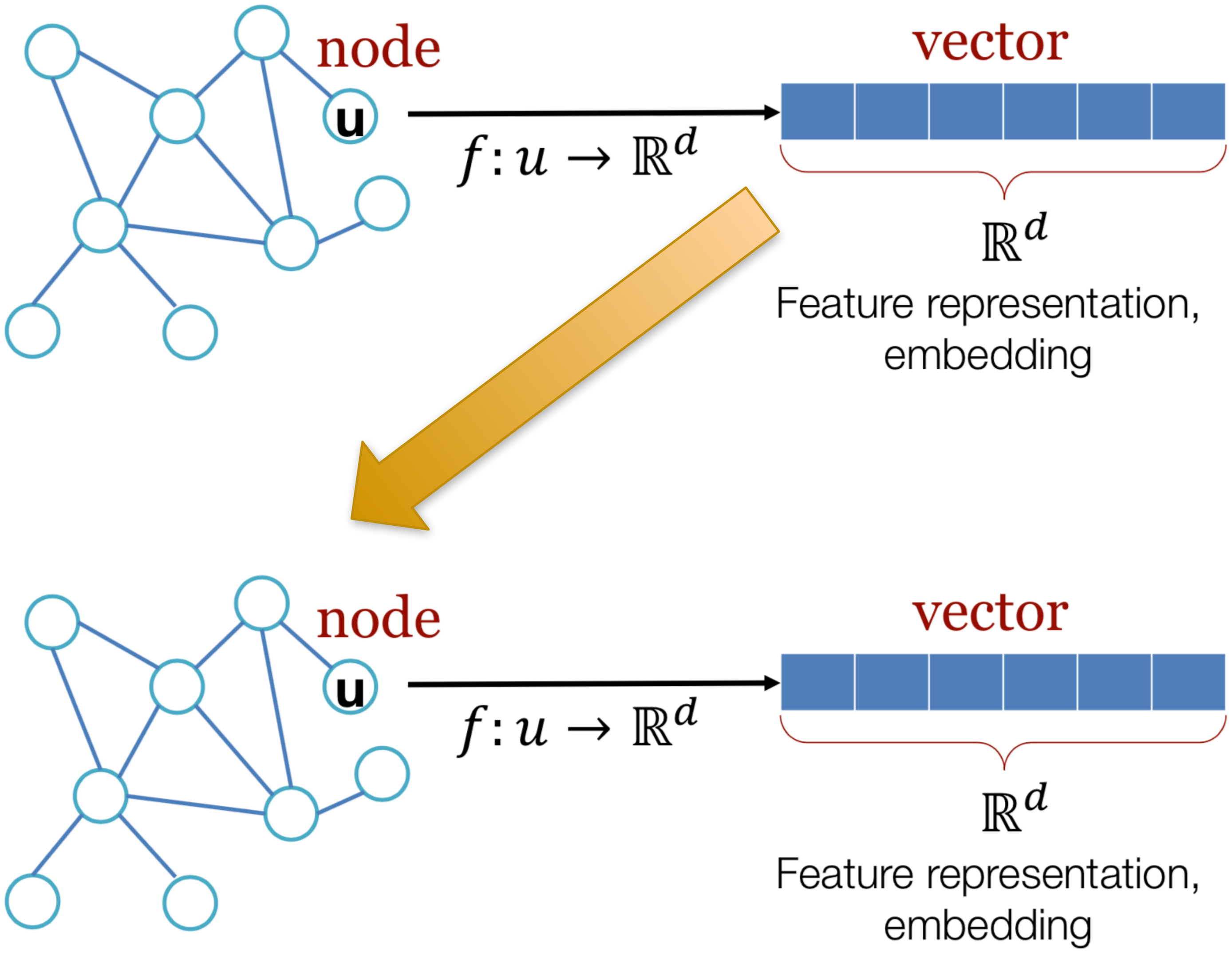
Deep Learning on Graphs

- Each node has its own features
- The embedding of each node is a function of its own features and its neighbors’ features.





Stack multiple layers

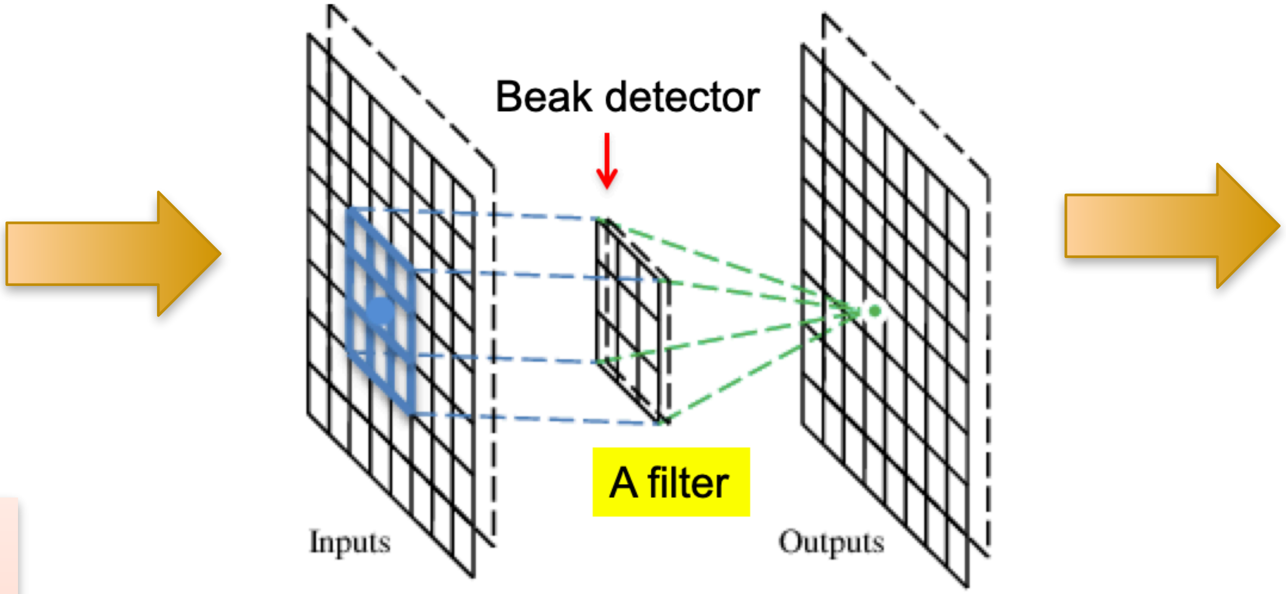


First layer

For the second layer, the feature is the hidden output by the first layer

One common mistake: Graphs are not Images

Images



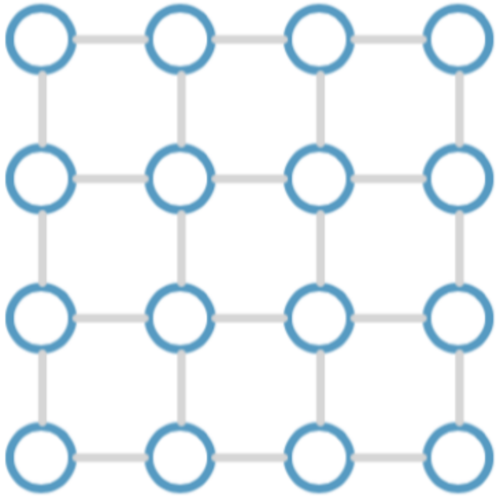
Label

$$\begin{matrix} 0 & , & 0 & , & \dots & , & 1 \\ 1 & , & 0 & , & \dots & , & 0 \\ \vdots & & \vdots & & & & \vdots \\ 0 & , & 1 & , & \dots & , & 1 \\ \vdots & & \vdots & & & & \vdots \end{matrix}$$

Adjacent matrix of a Graph

But Images can be modeled as Graphs

Images



2D grid

Graphical

Neural Network