Understanding Networks through Physical Metaphors

Daniel A. Spielman

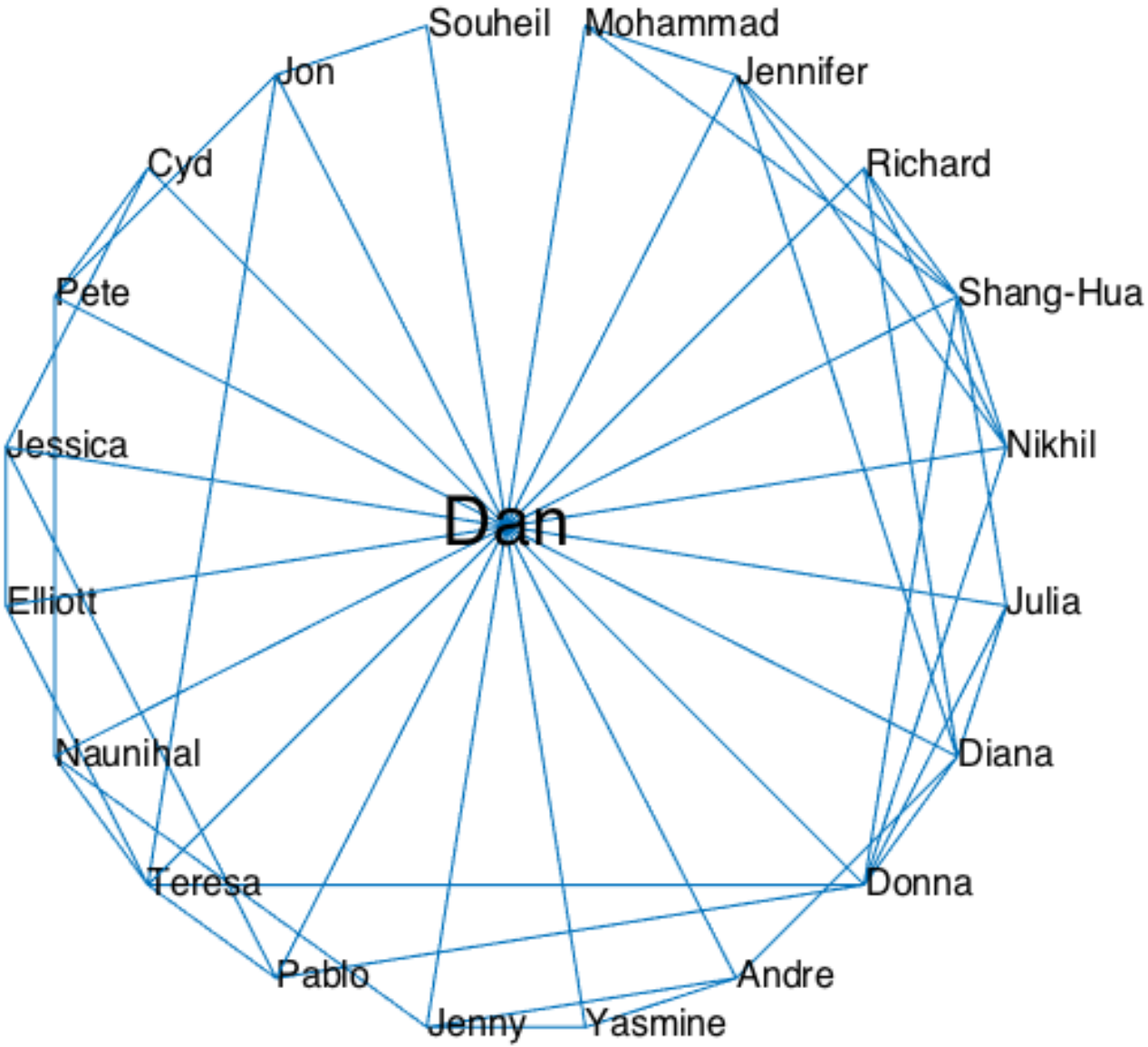
Yale Institute For
Network Science

Graphs and Networks

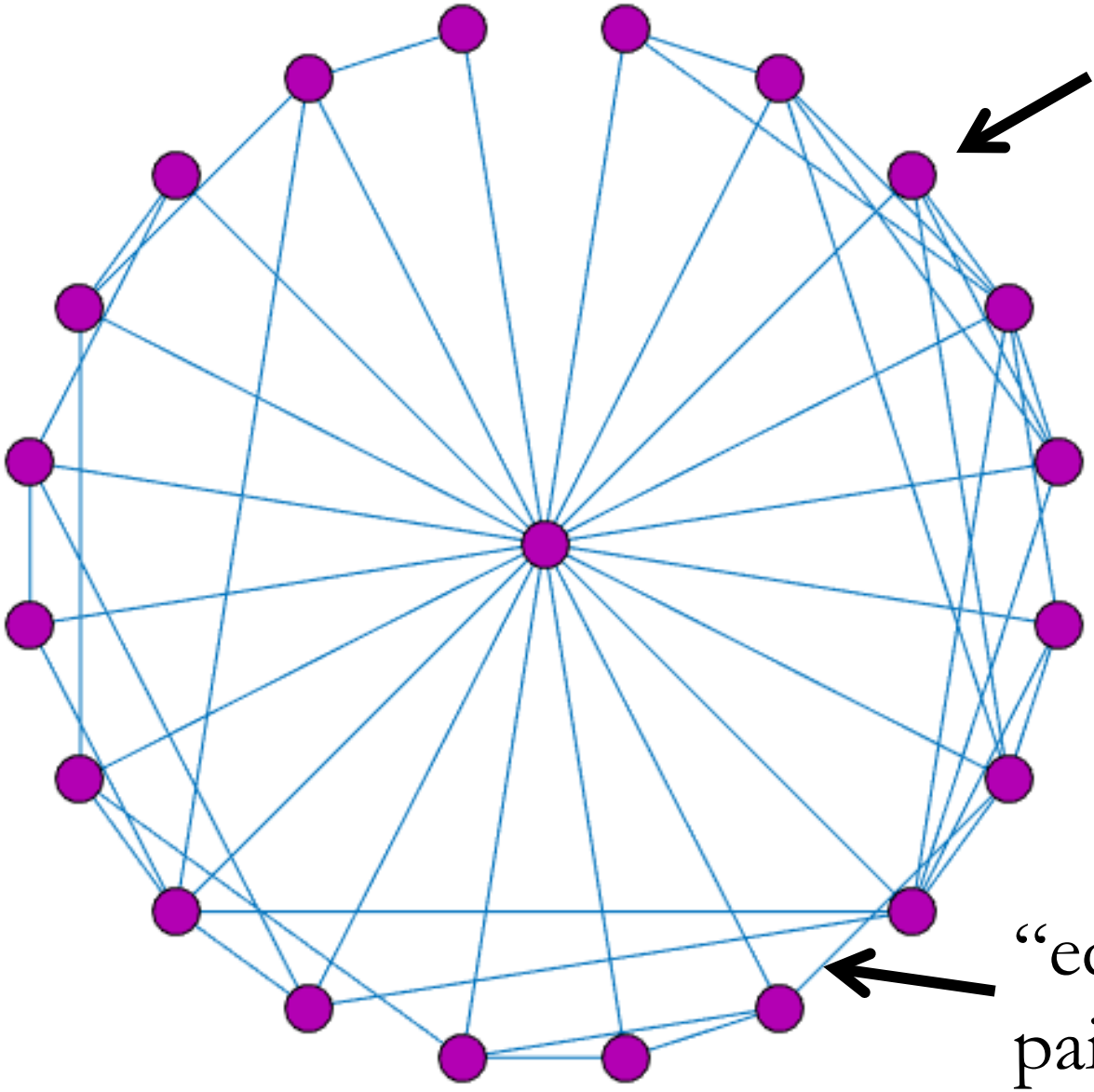
Analysis by Physical Metaphors

Algorithms (what I usually do)

A Social Network Graph



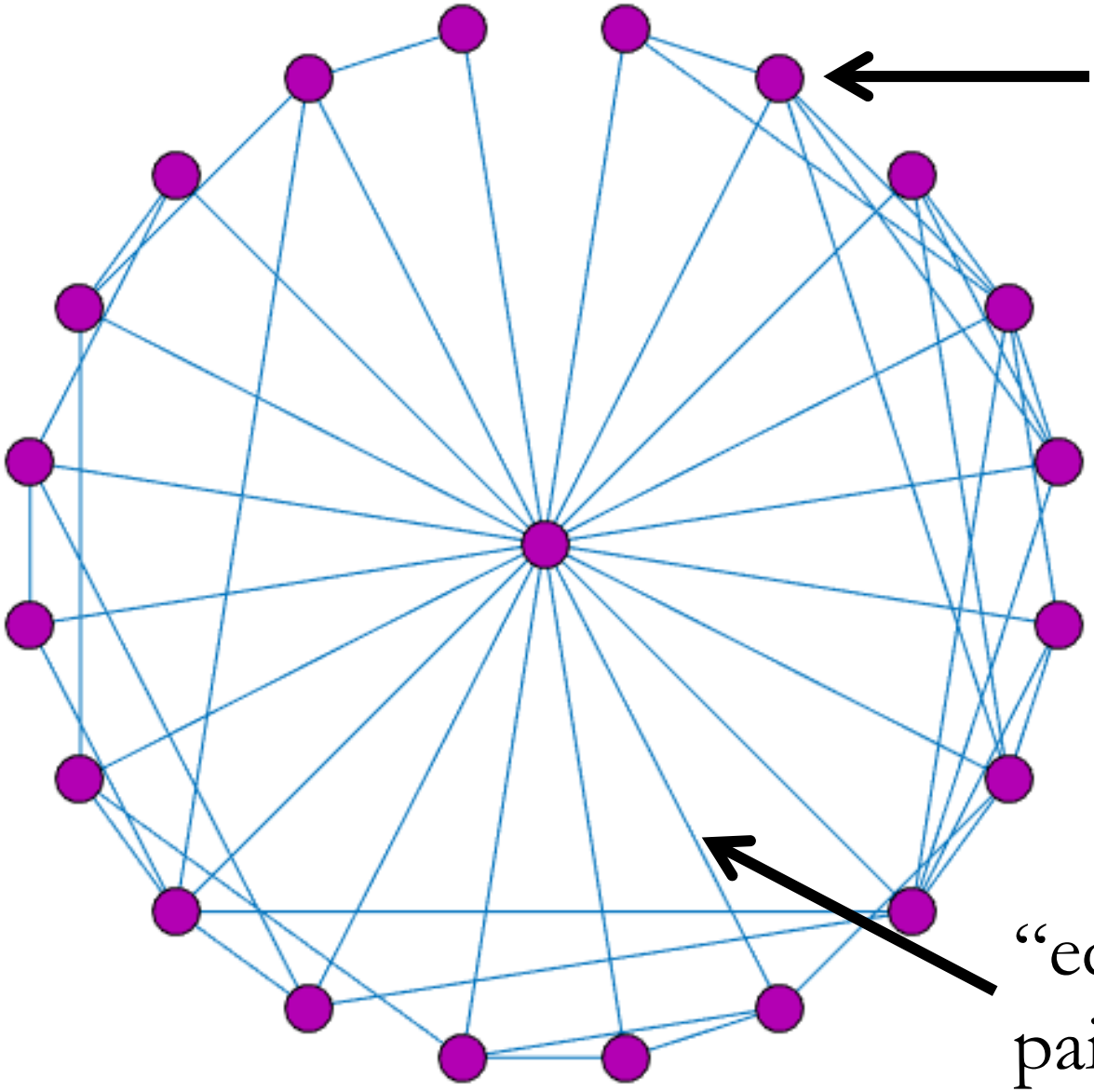
A Social Network Graph



"vertex"
or "node"

"edge" =
pair of nodes

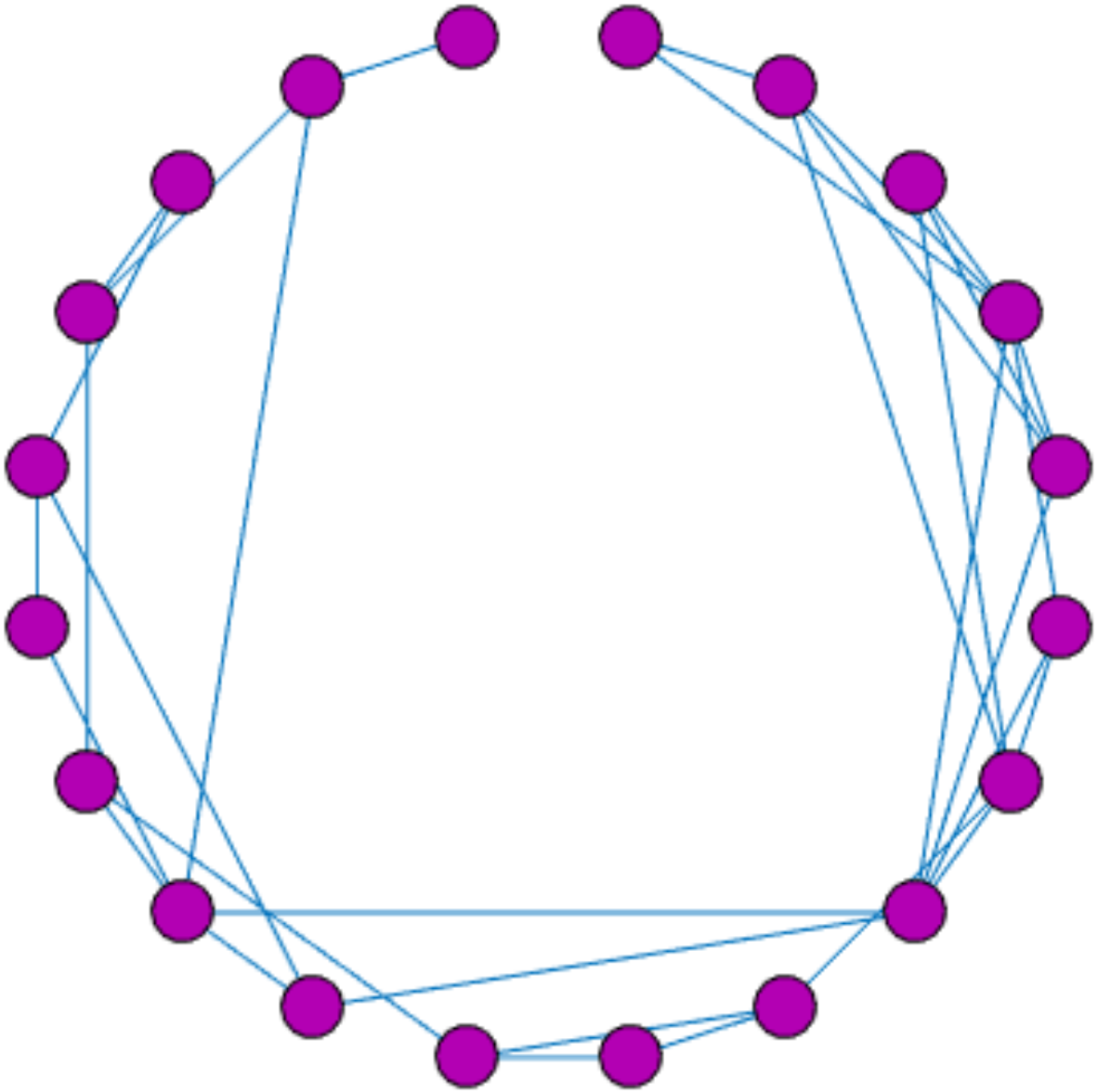
A Social Network Graph



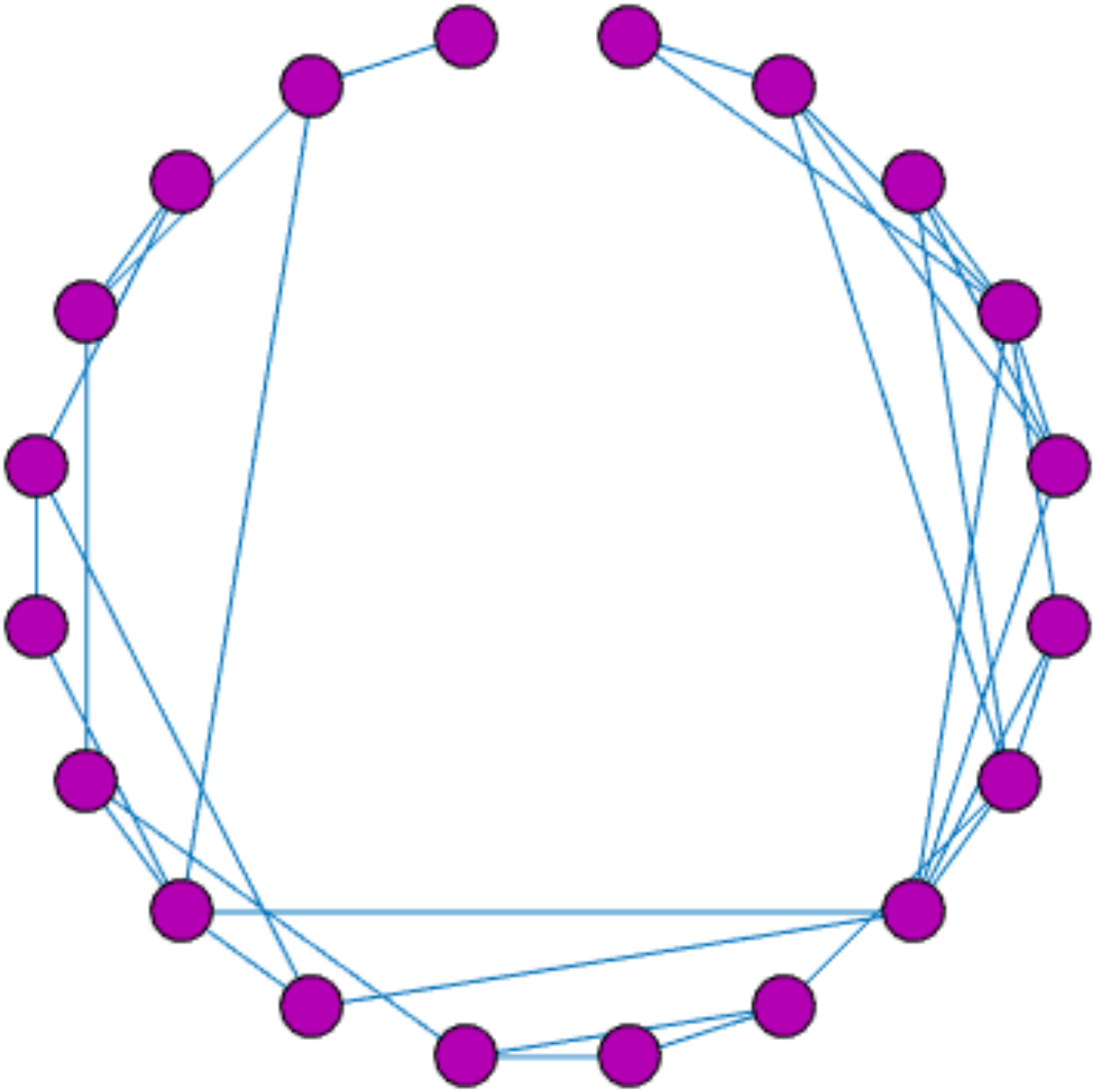
"vertex"
or "node"

"edge" =
pair of nodes

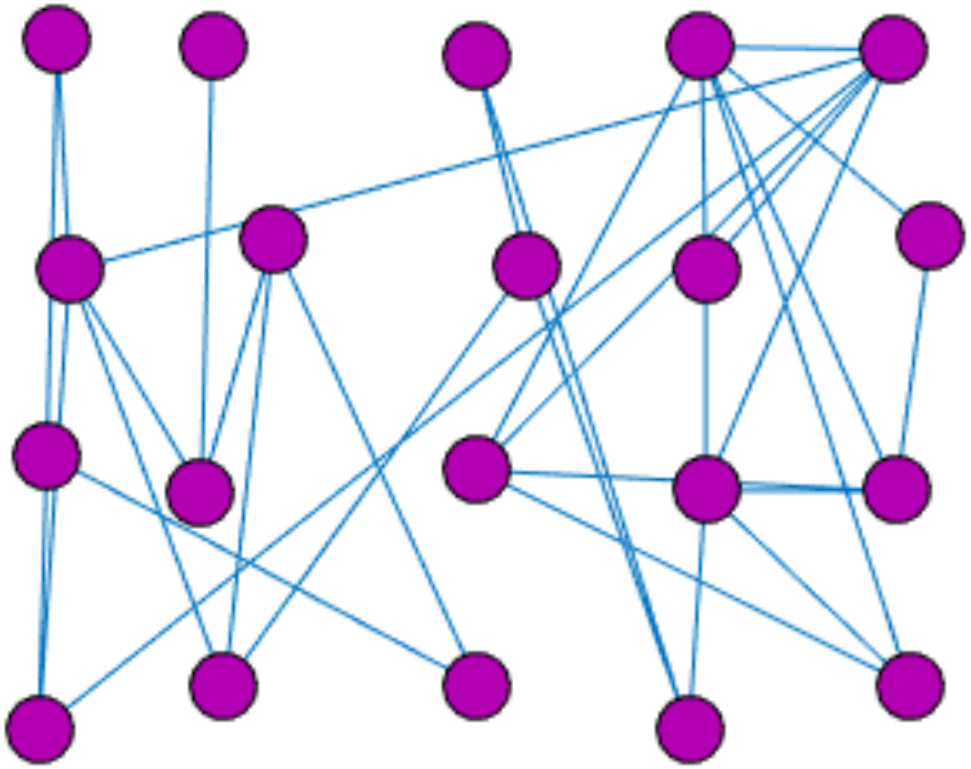
A Social Network Graph



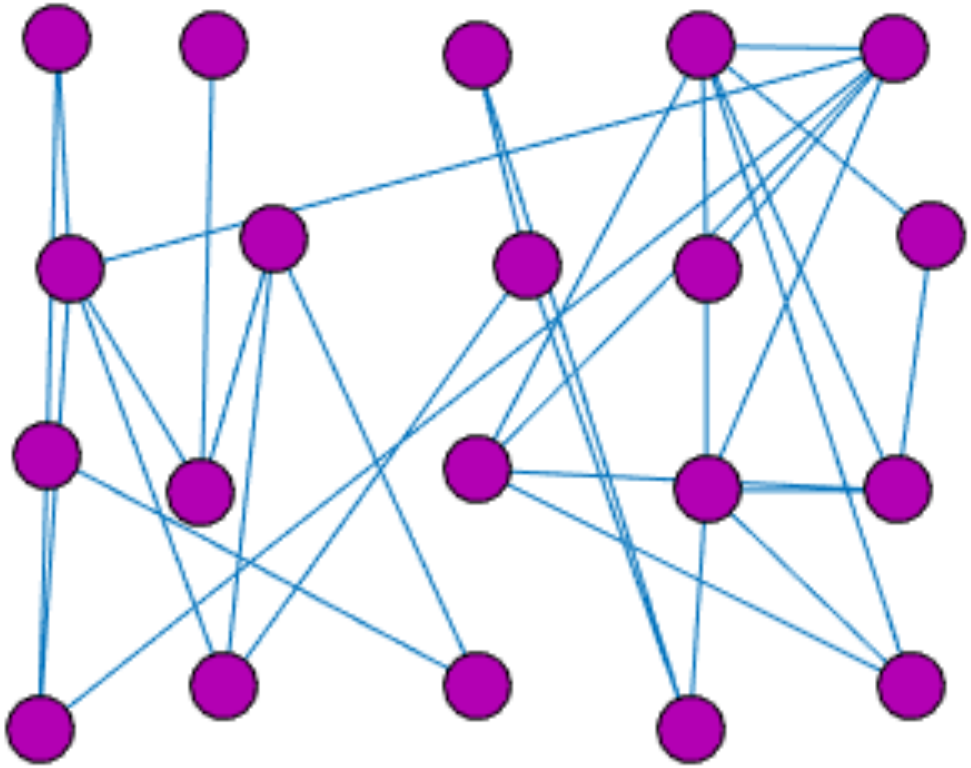
A Social Network Graph



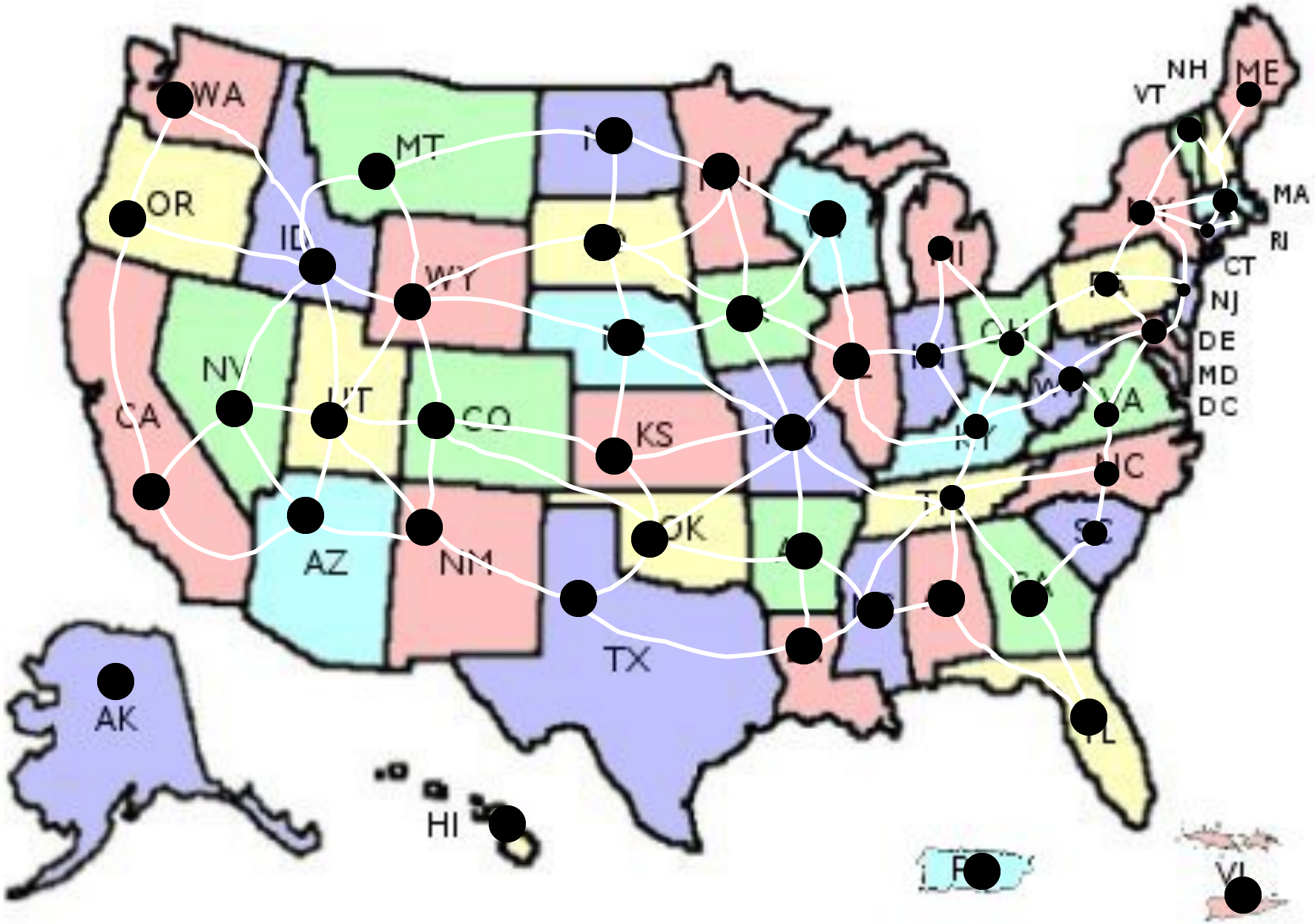
A Social Network Graph



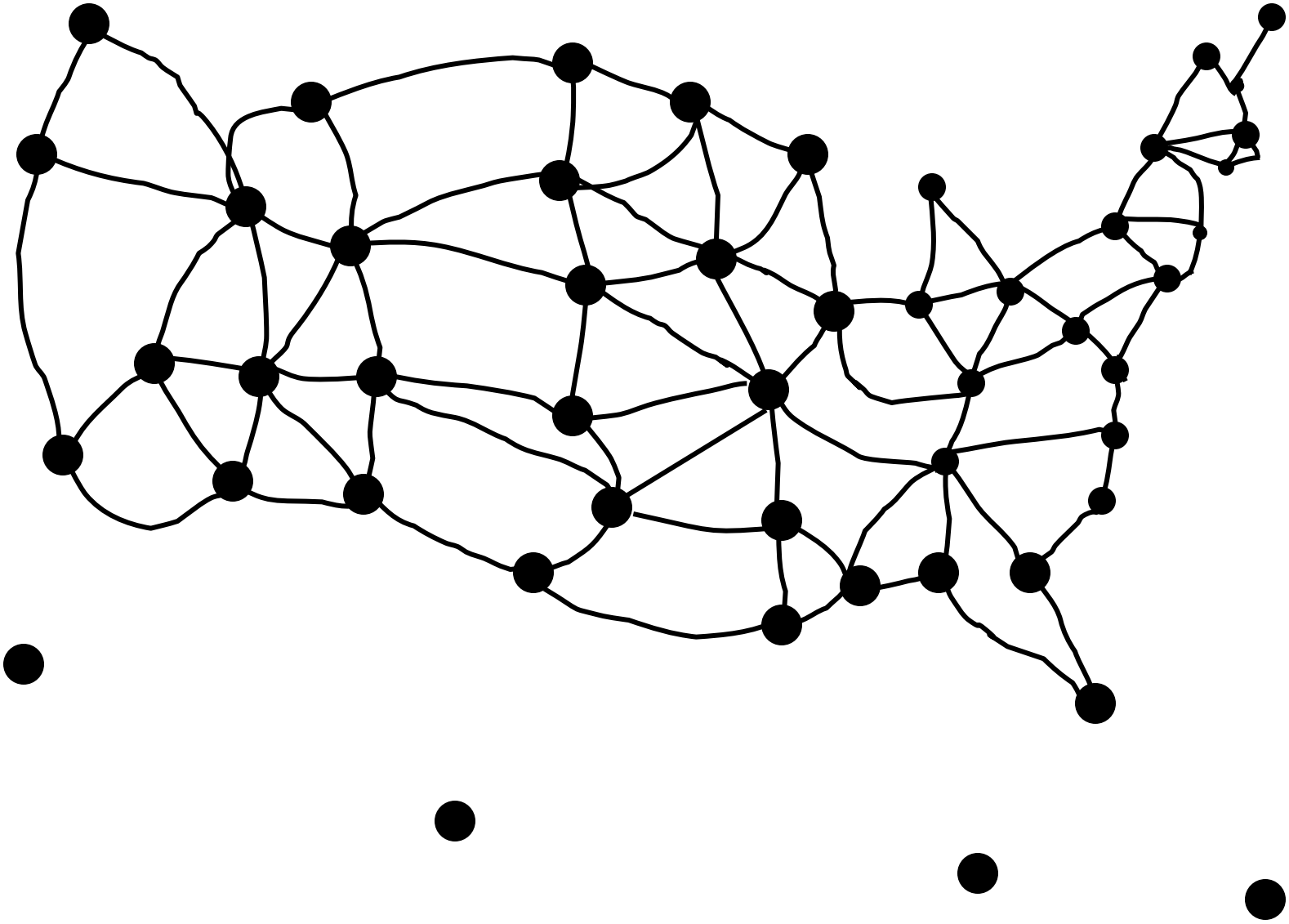
A Social Network Graph



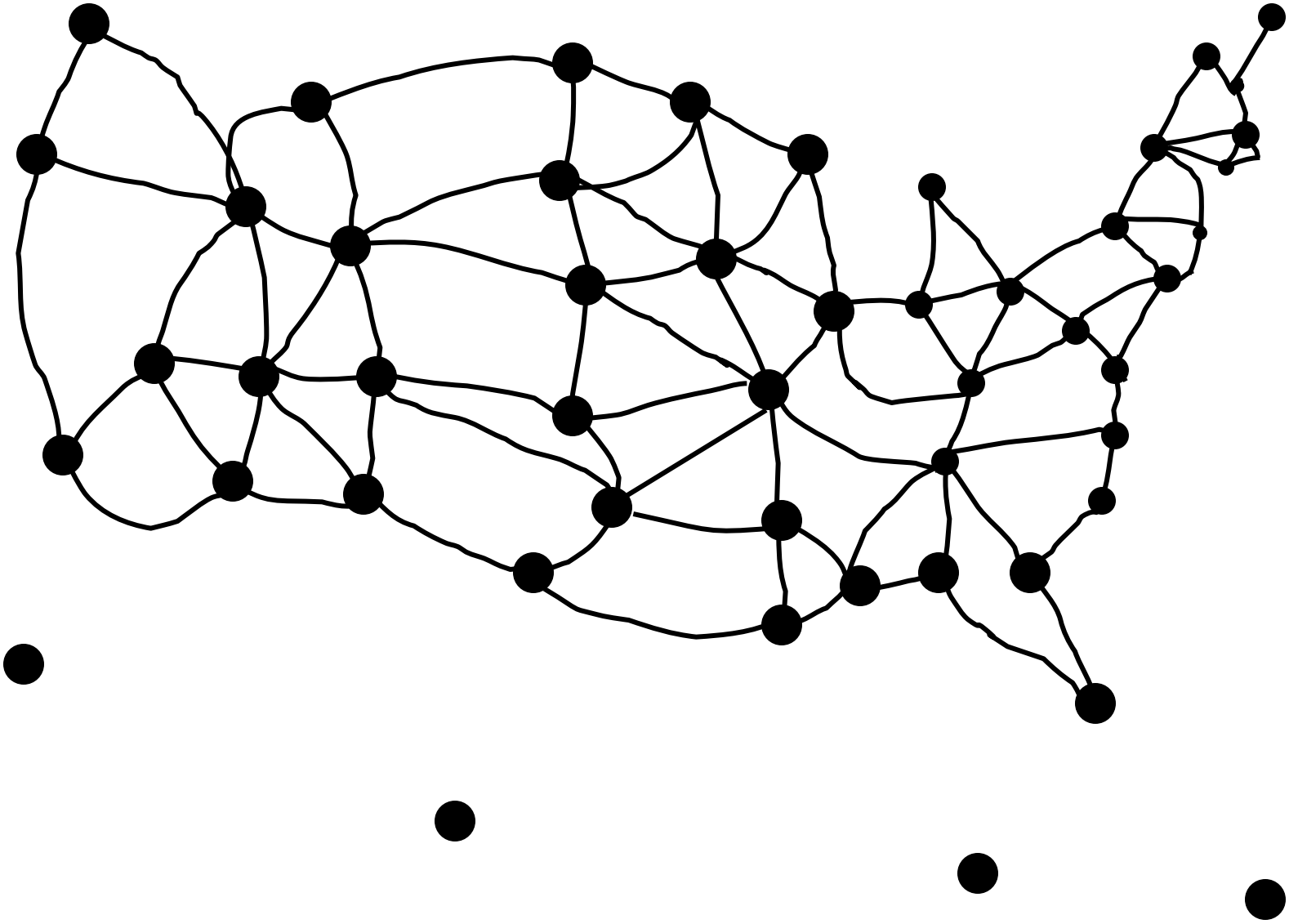
from: http://nrc.uchsc.edu/STATES/united-states-map.jpg



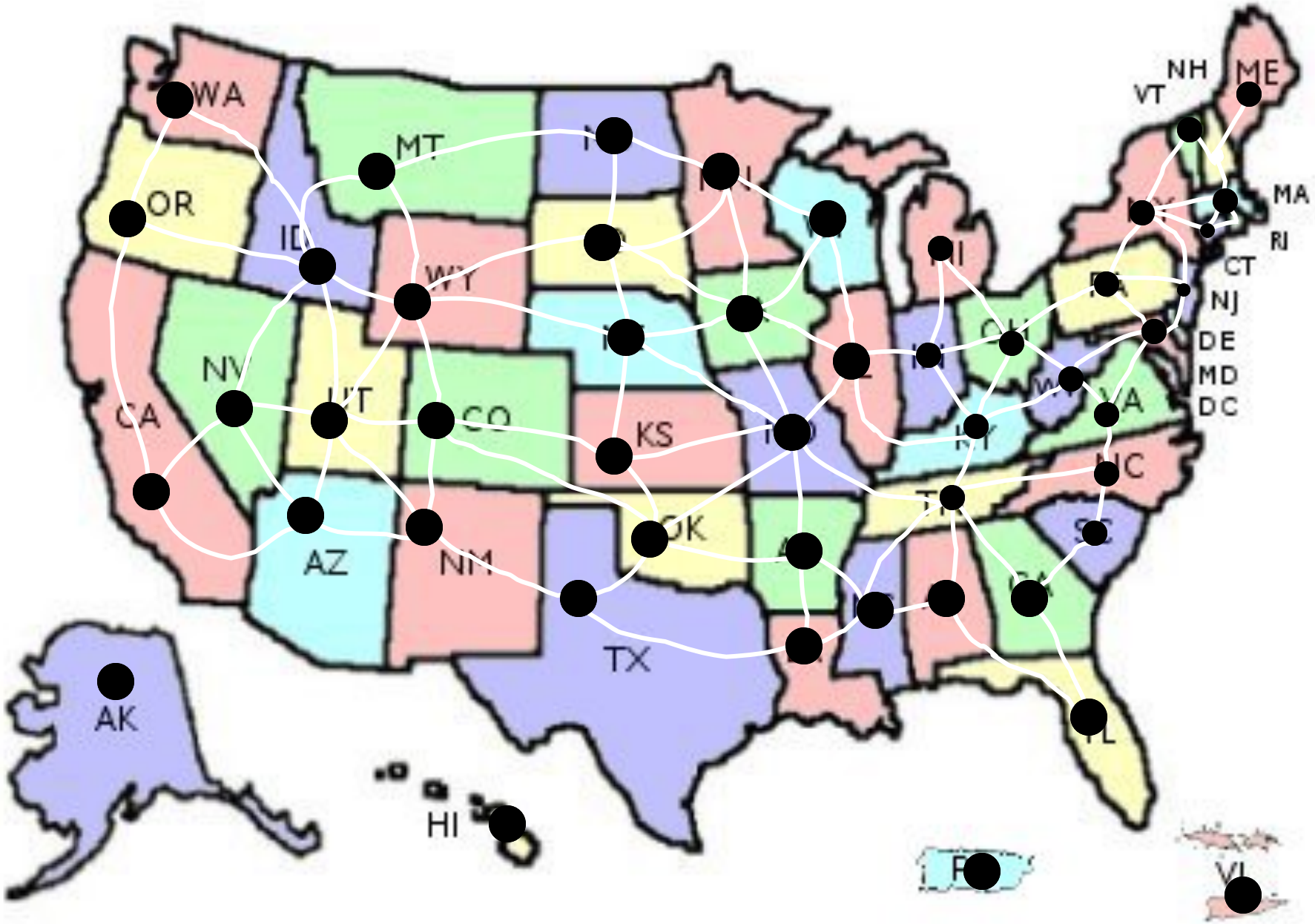
Vertices = States. Edges connect adjoining states.



Vertices = States. Edges connect adjoining states.

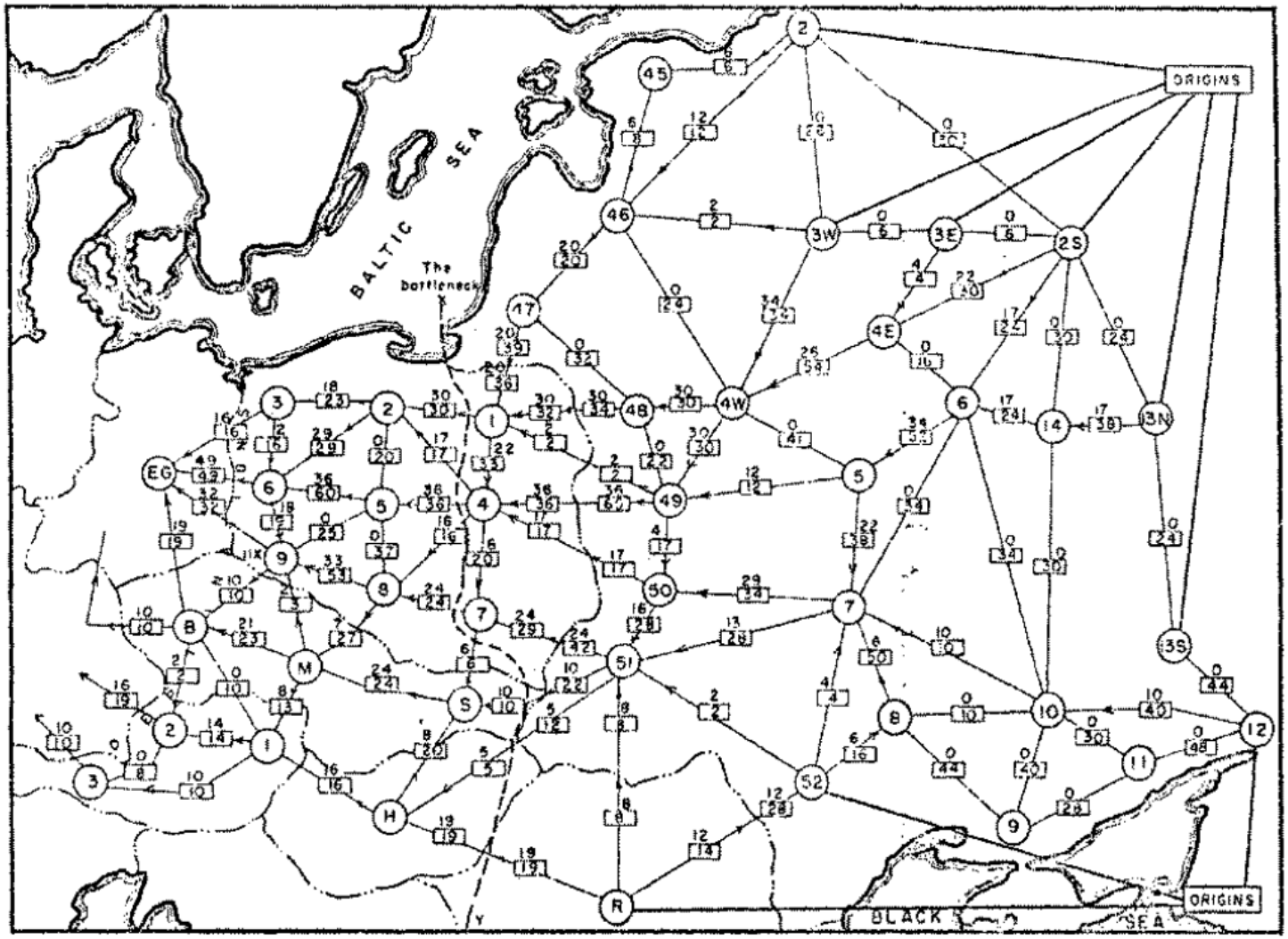


Planar = can draw without crossing edges



Planar implies can draw with 4 colors

A Transportation Network

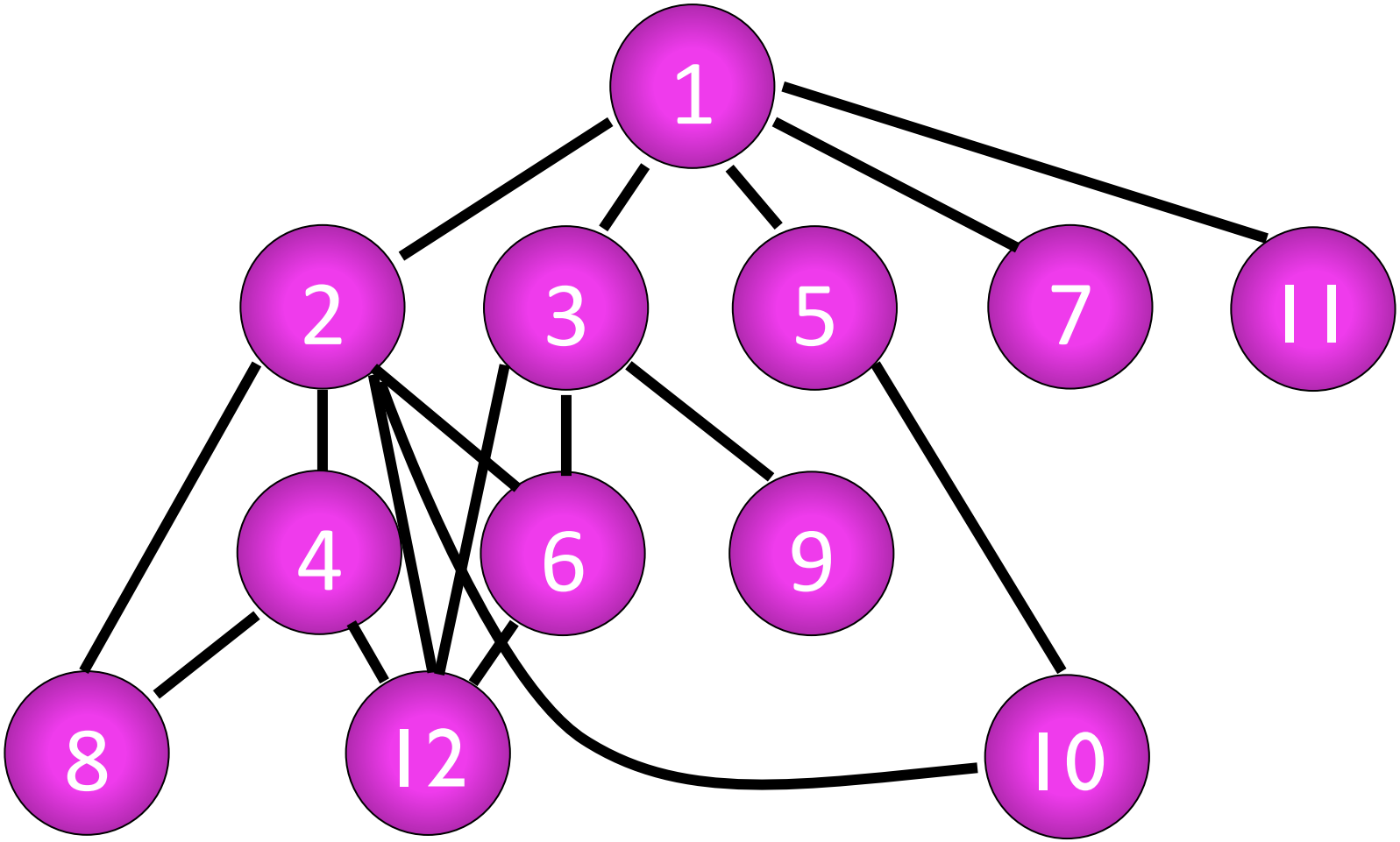


Harris and Ross, RAND Corporation, 1955

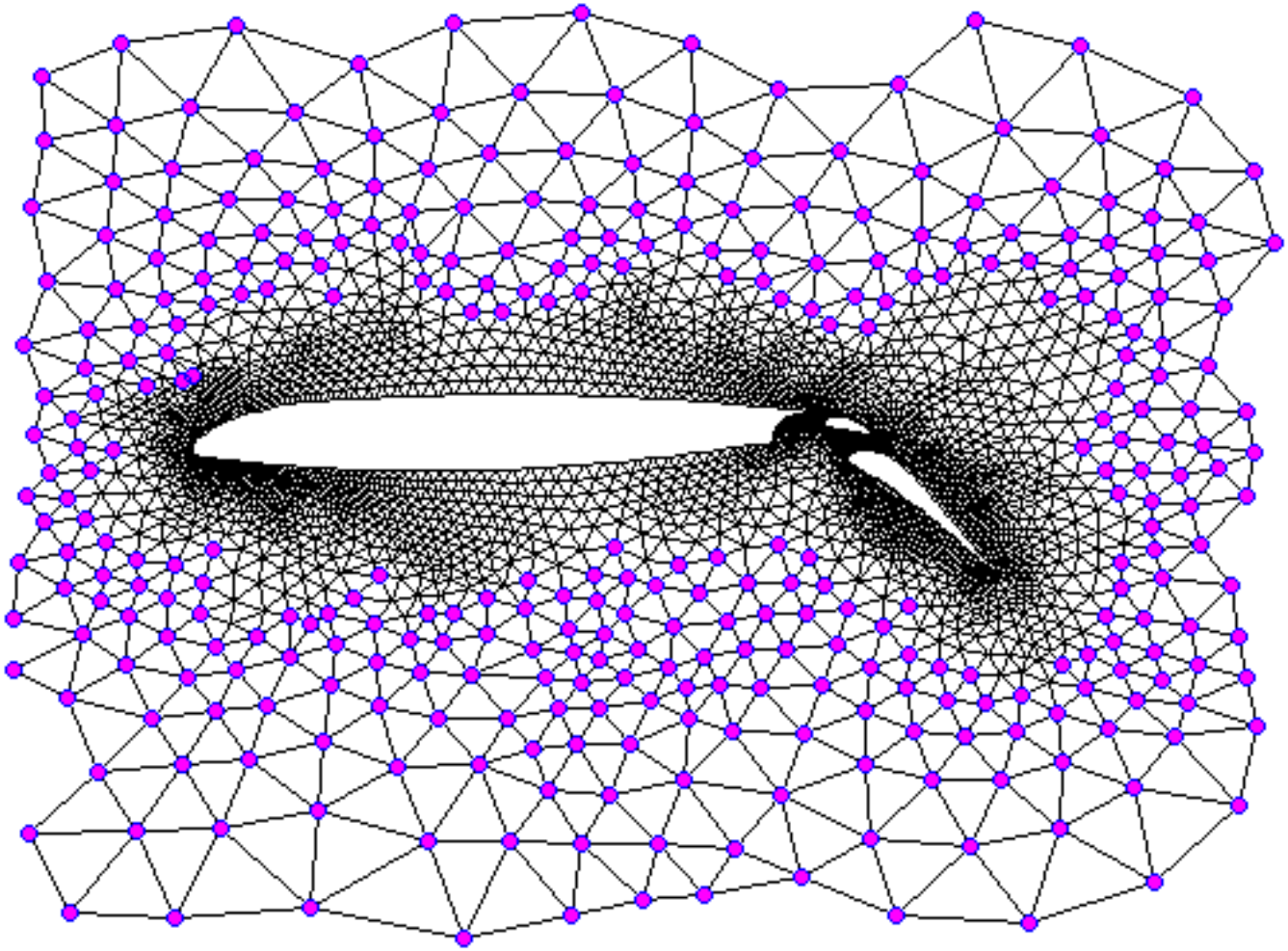
A Mathematician's Network

Vertices = Natural numbers

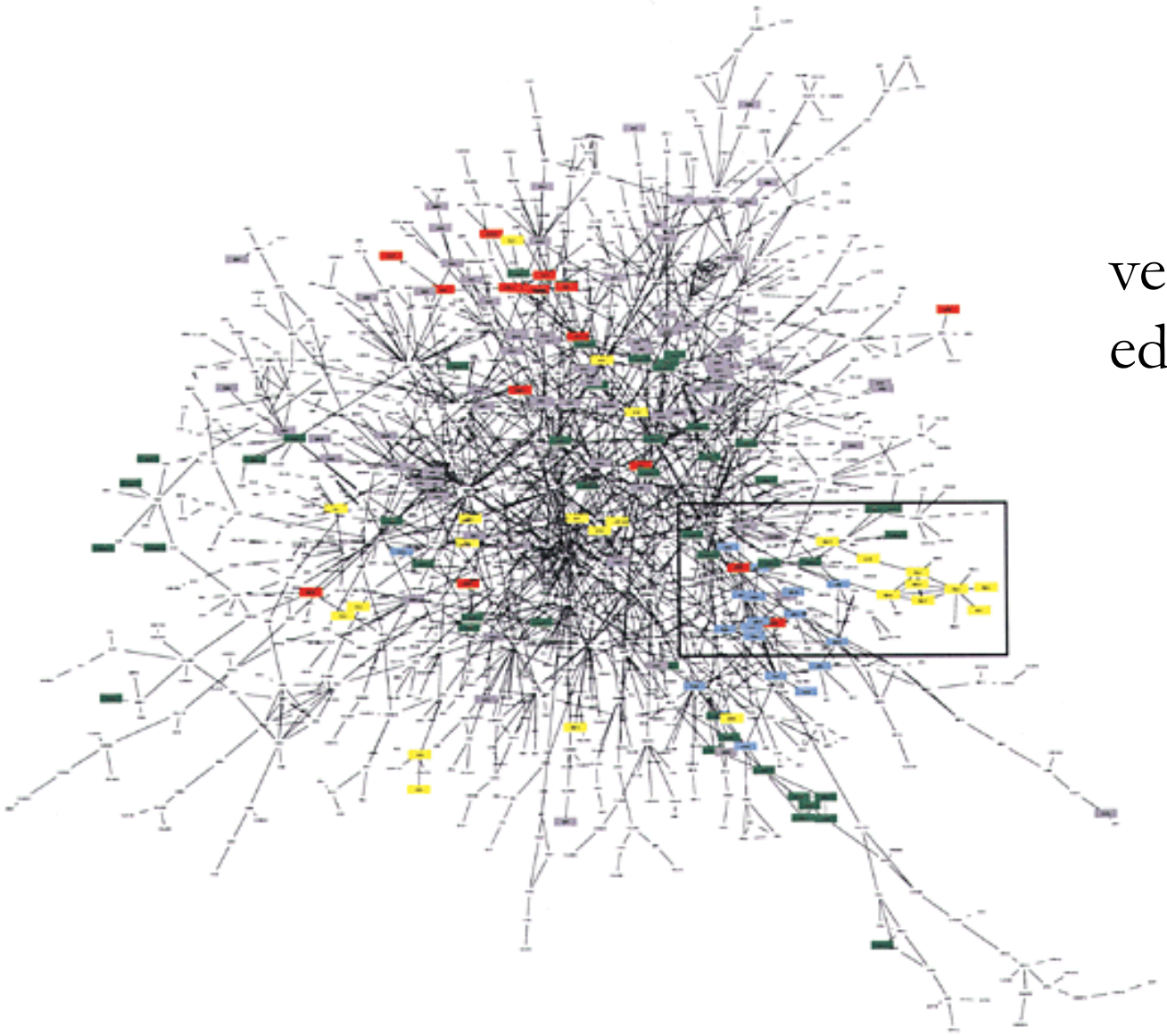
Edges between pairs where one divides another



The Graph of a Mesh



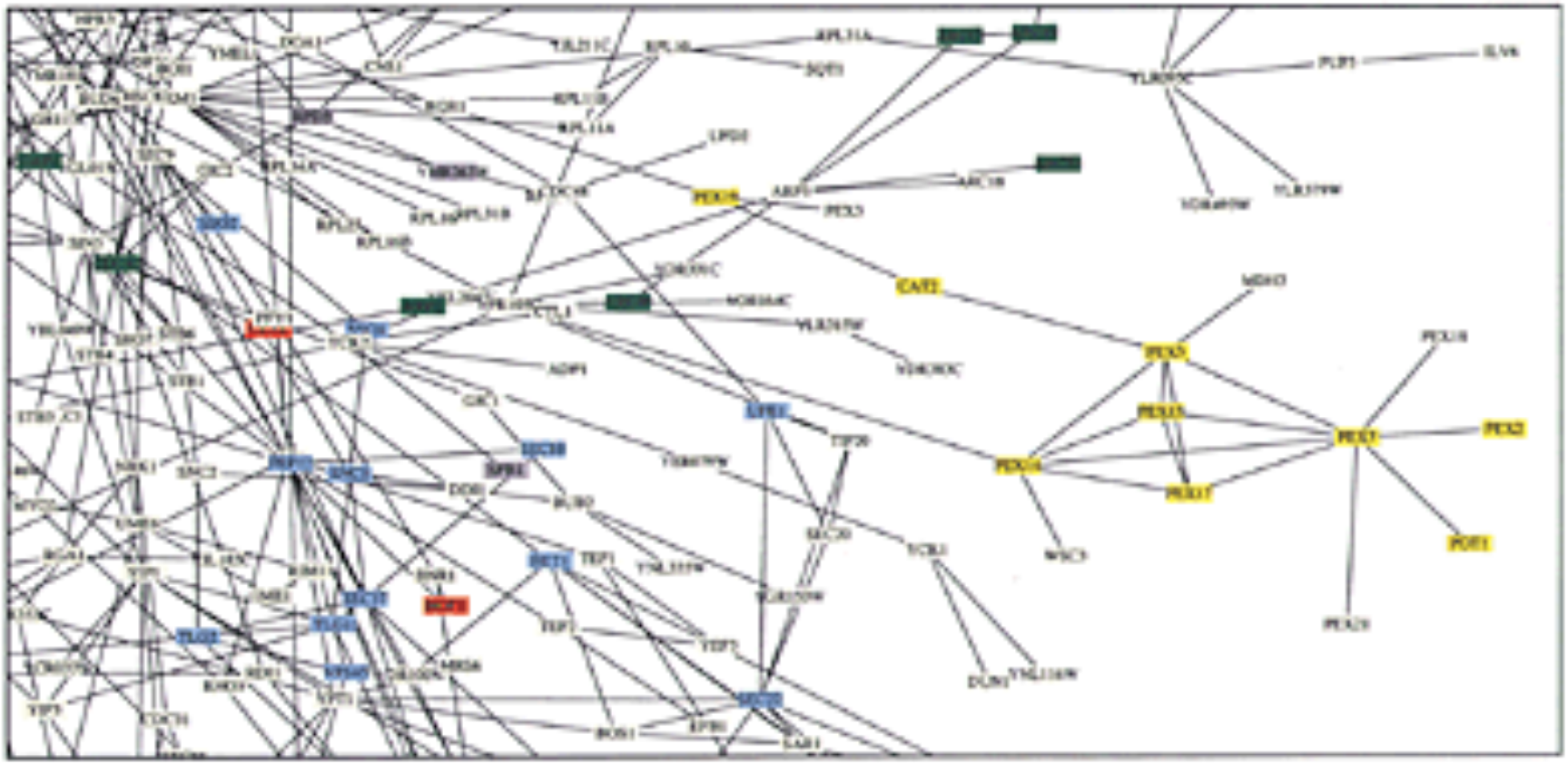
Protein-protein Interaction Networks



vertex = protein
edge if "interact"

Schwikowski, Uetz & Fields, Nature Biotechnology 2000

Protein-protein Interaction Networks

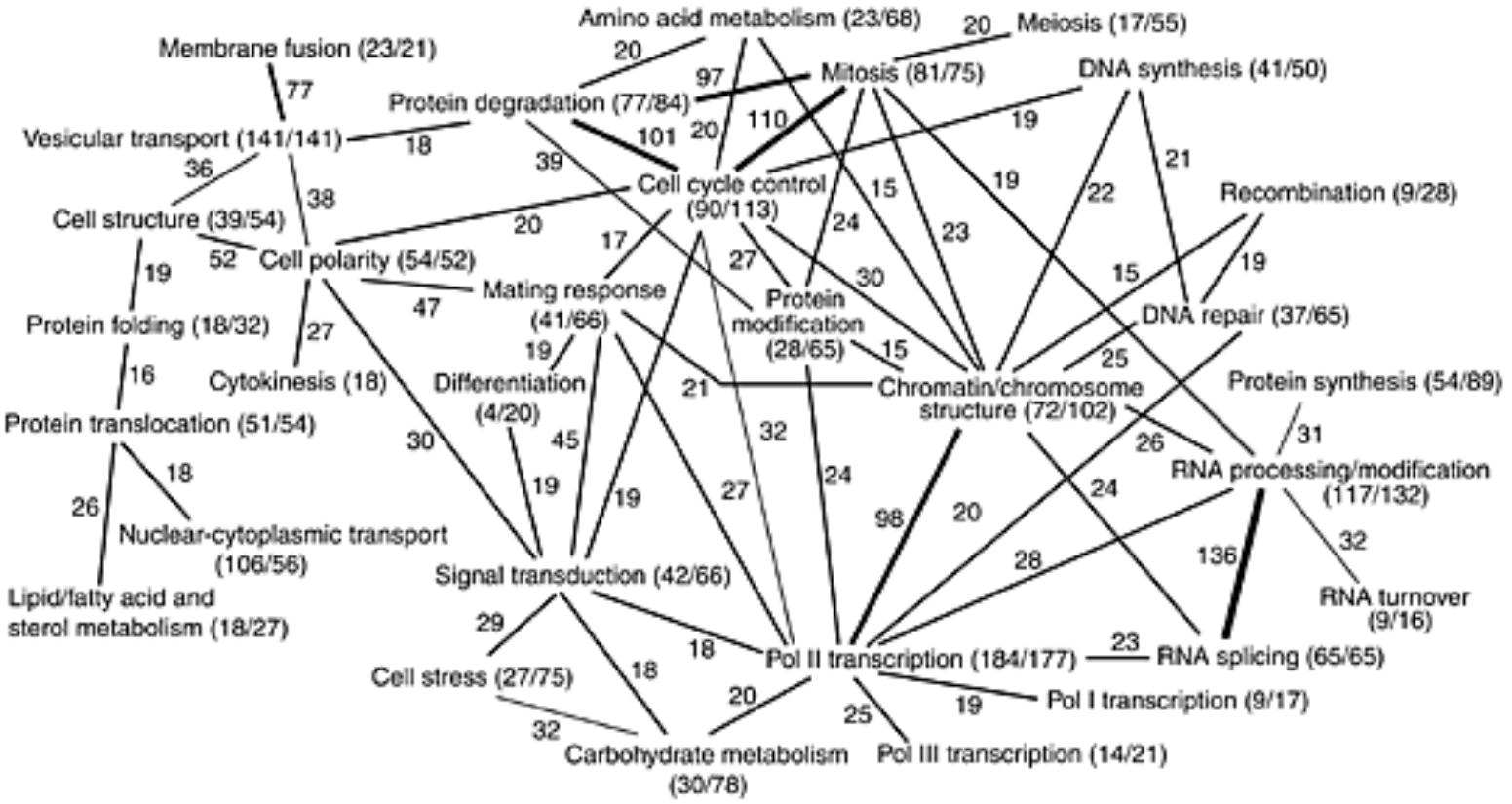


vertex = protein

edge experimentally observed interaction

Schwikowski, Uetz & Fields, Nature Biotechnology 2000

Protein-protein Interaction Networks



vertex = group of proteins

edges between groups with interacting proteins

Schwikowski, Uetz & Fields, Nature Biotechnology 2000

Other Networks

Web

Vertices = Web pages. Edges = links.

Trade networks

Vertices = Companies. Edges when trade.

Gene regulatory networks

Vertices = Genes.

Edge when one regulates another.

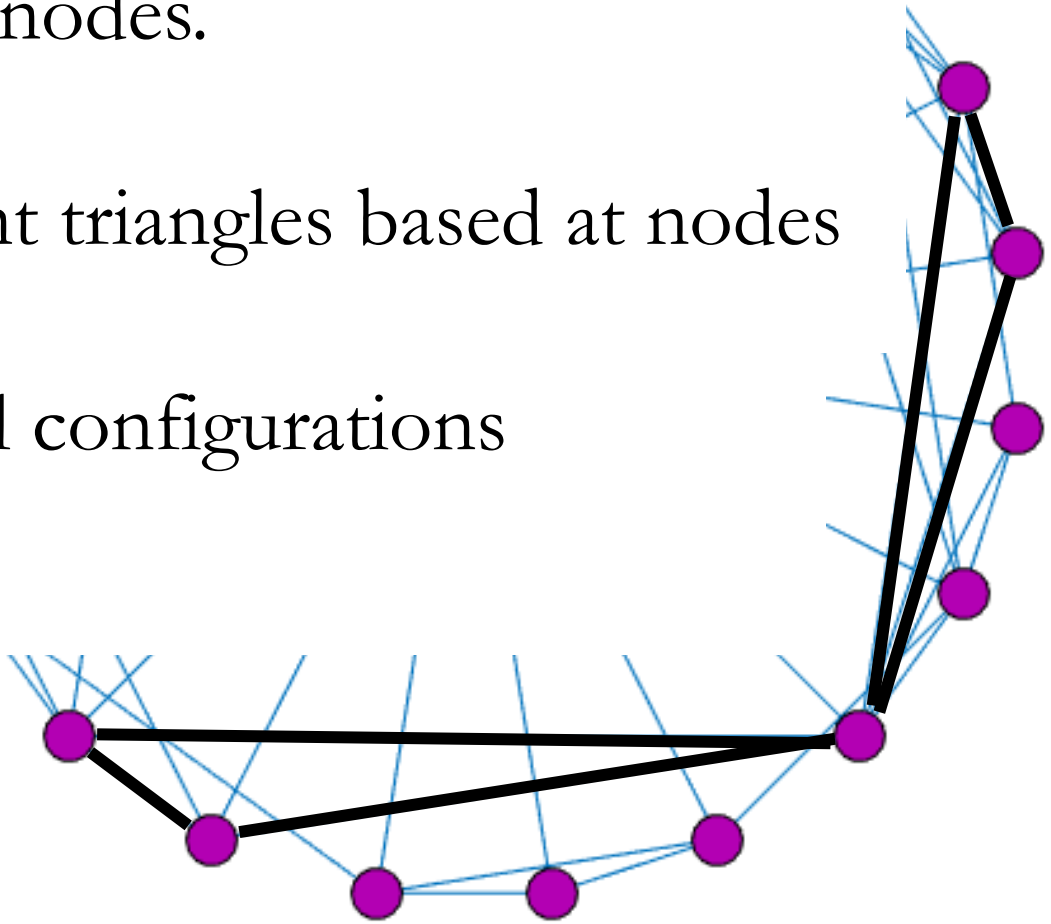
Etc.

Local Analysis of Networks

Examine degrees (number of attached edges) of nodes.

Count triangles based at nodes

Small configurations



Global Analysis of Networks

Diameter: how many steps between nodes

Drawing: understand overall structure

Clusters: how to group the nodes

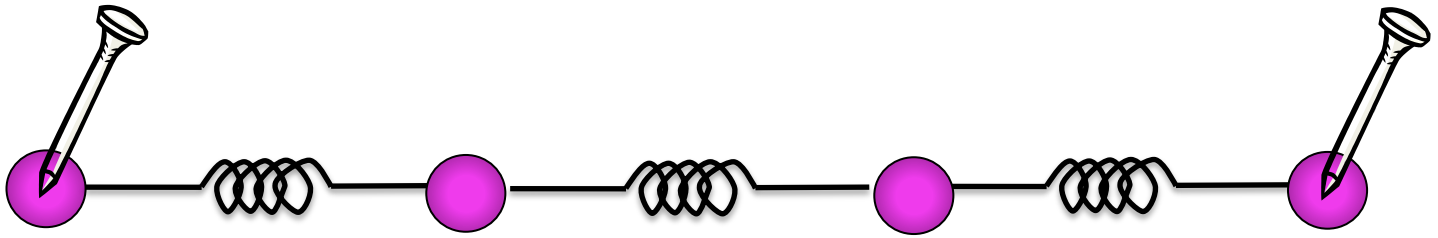
Inference: extrapolate from a few nodes

Importance: find most important nodes

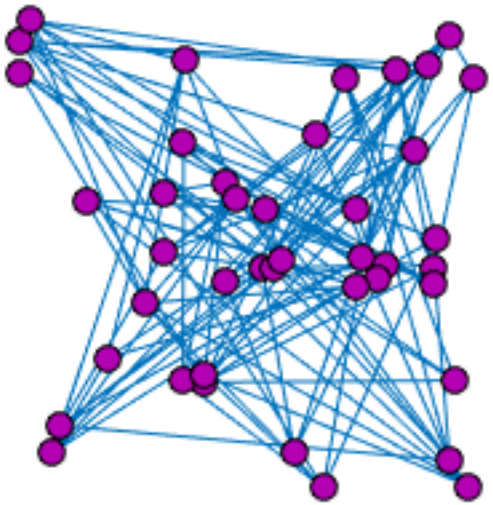
Graphs as Spring Networks

View edges as rubber bands or ideal linear springs

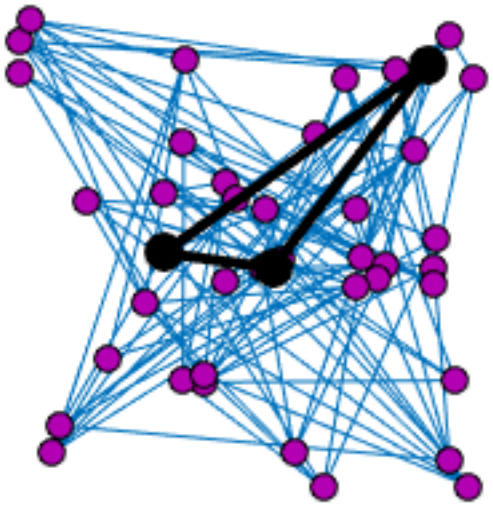
Nail down some vertices, let rest settle



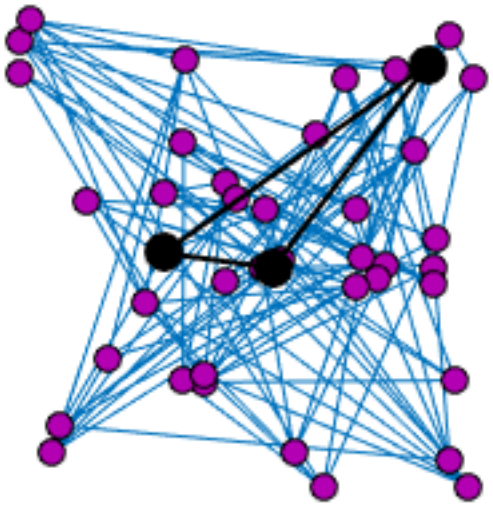
Drawing by Spring Networks



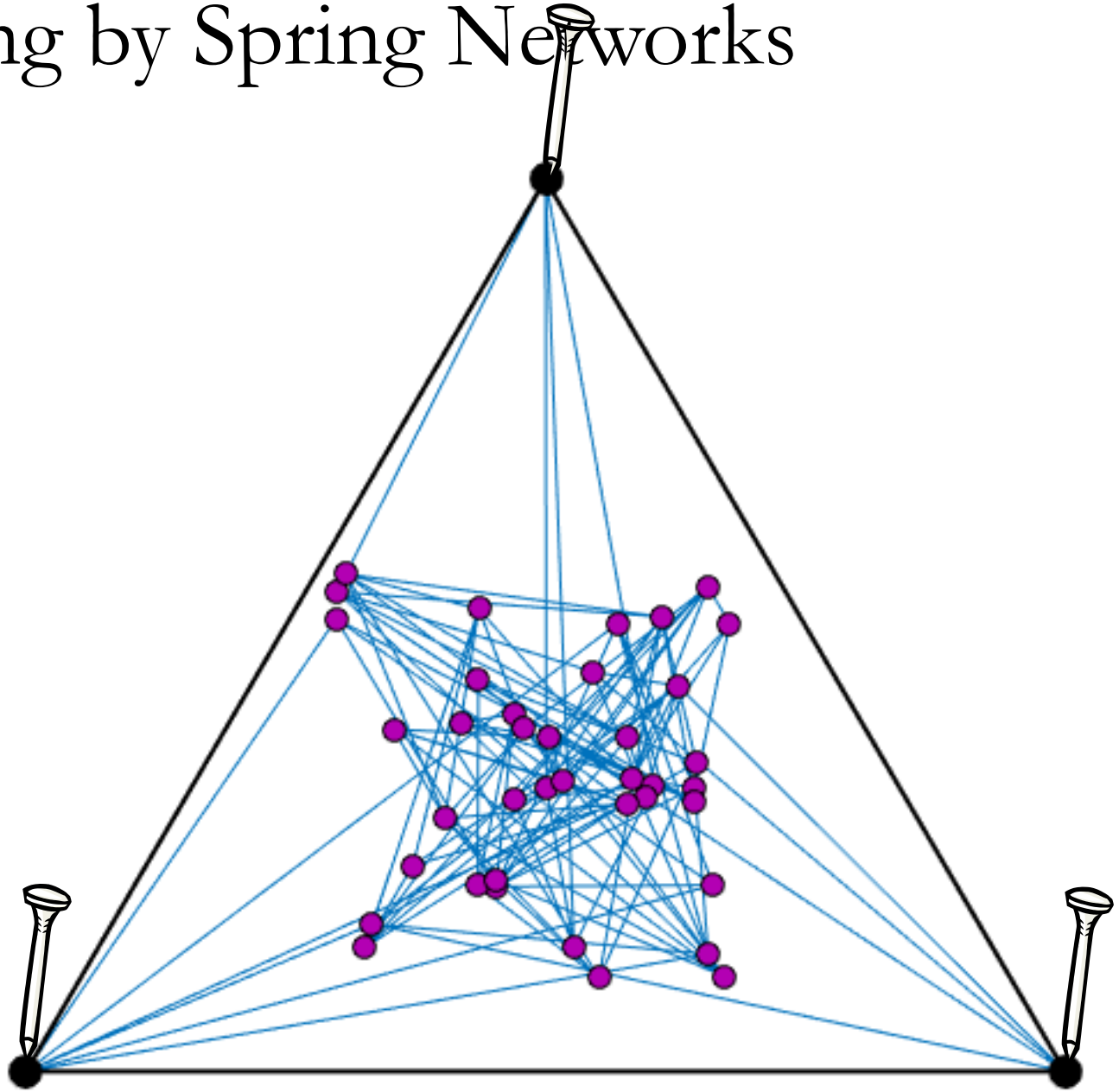
Drawing by Spring Networks



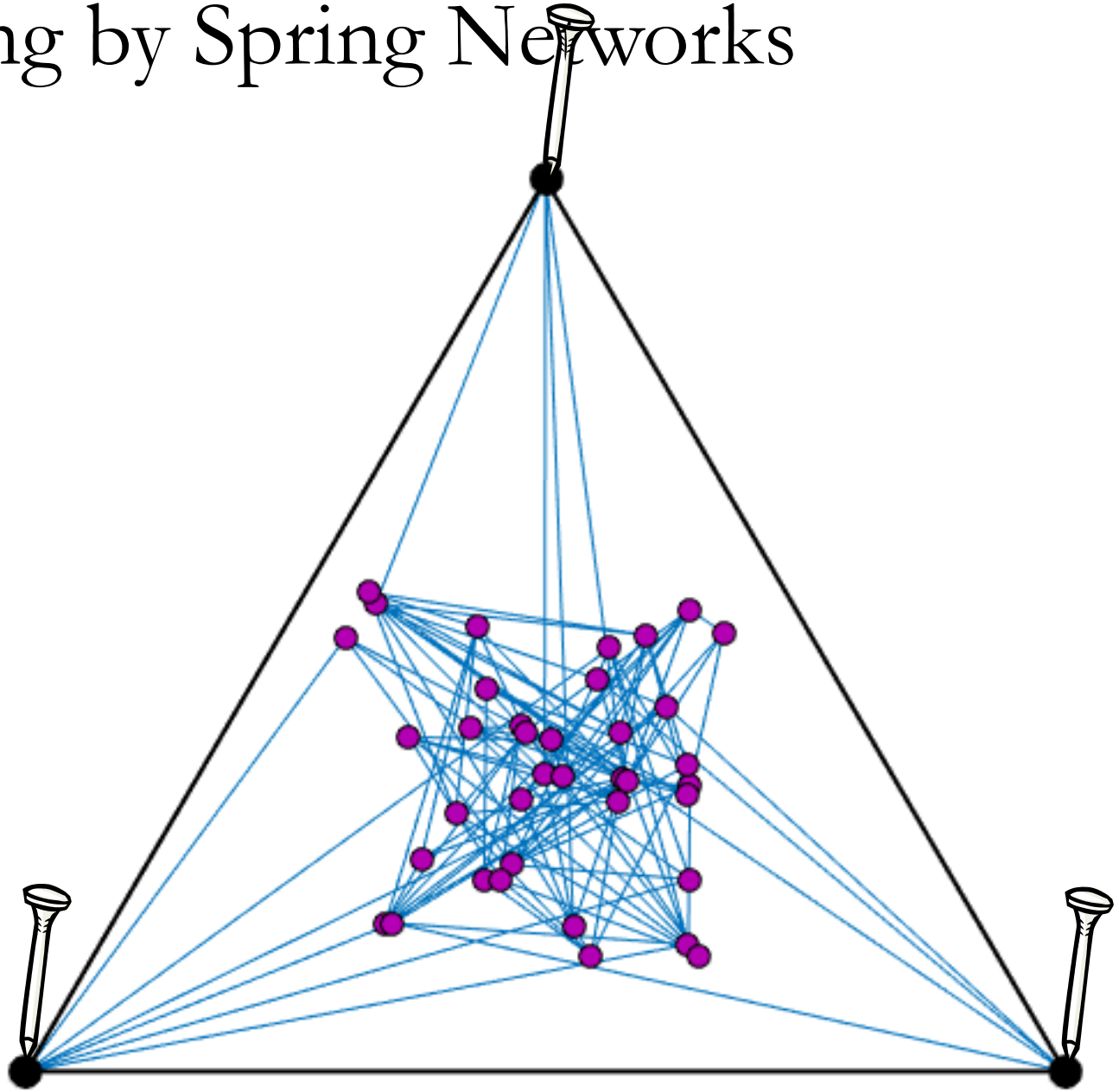
Drawing by Spring Networks



Drawing by Spring Networks

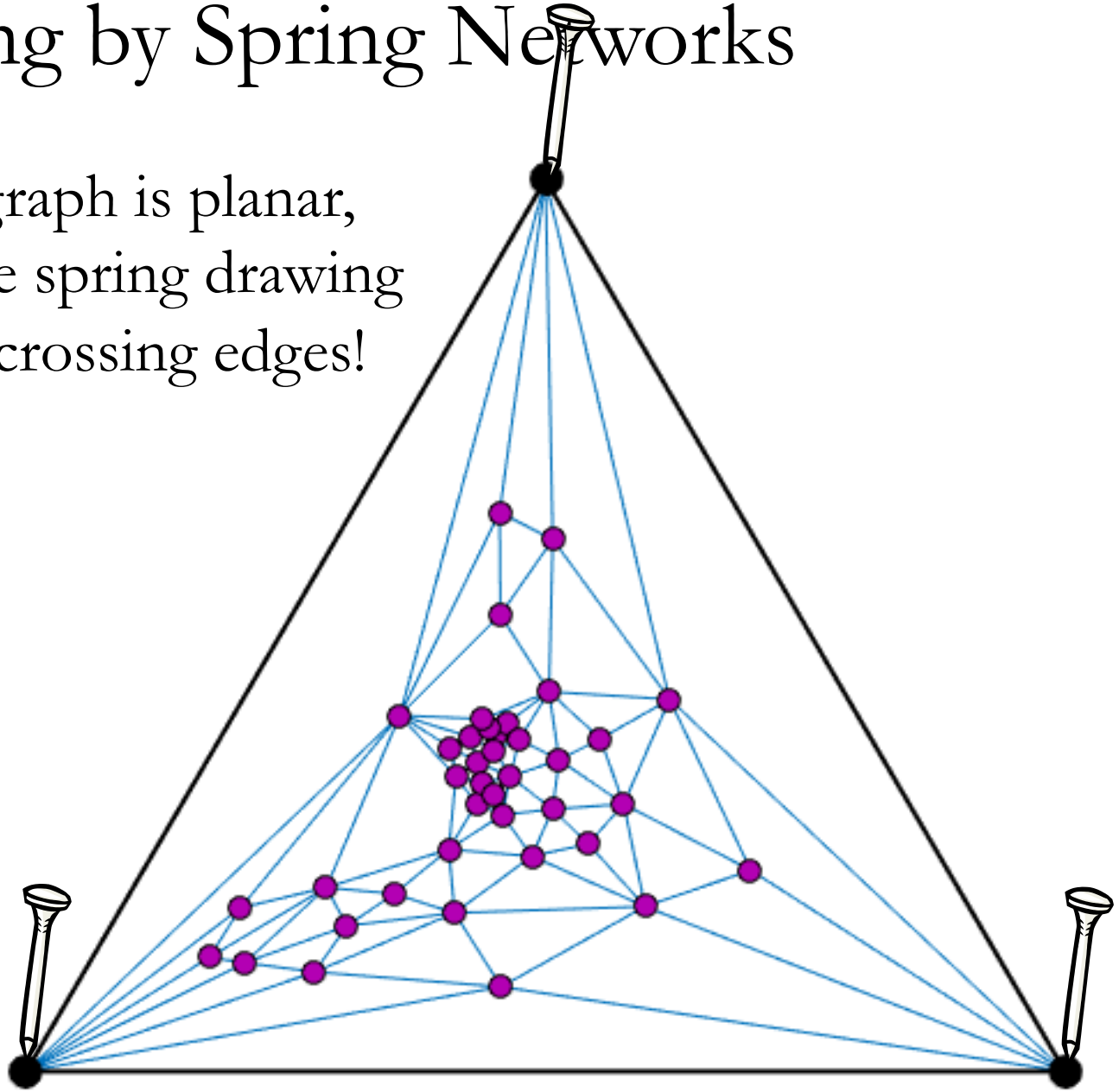


Drawing by Spring Networks

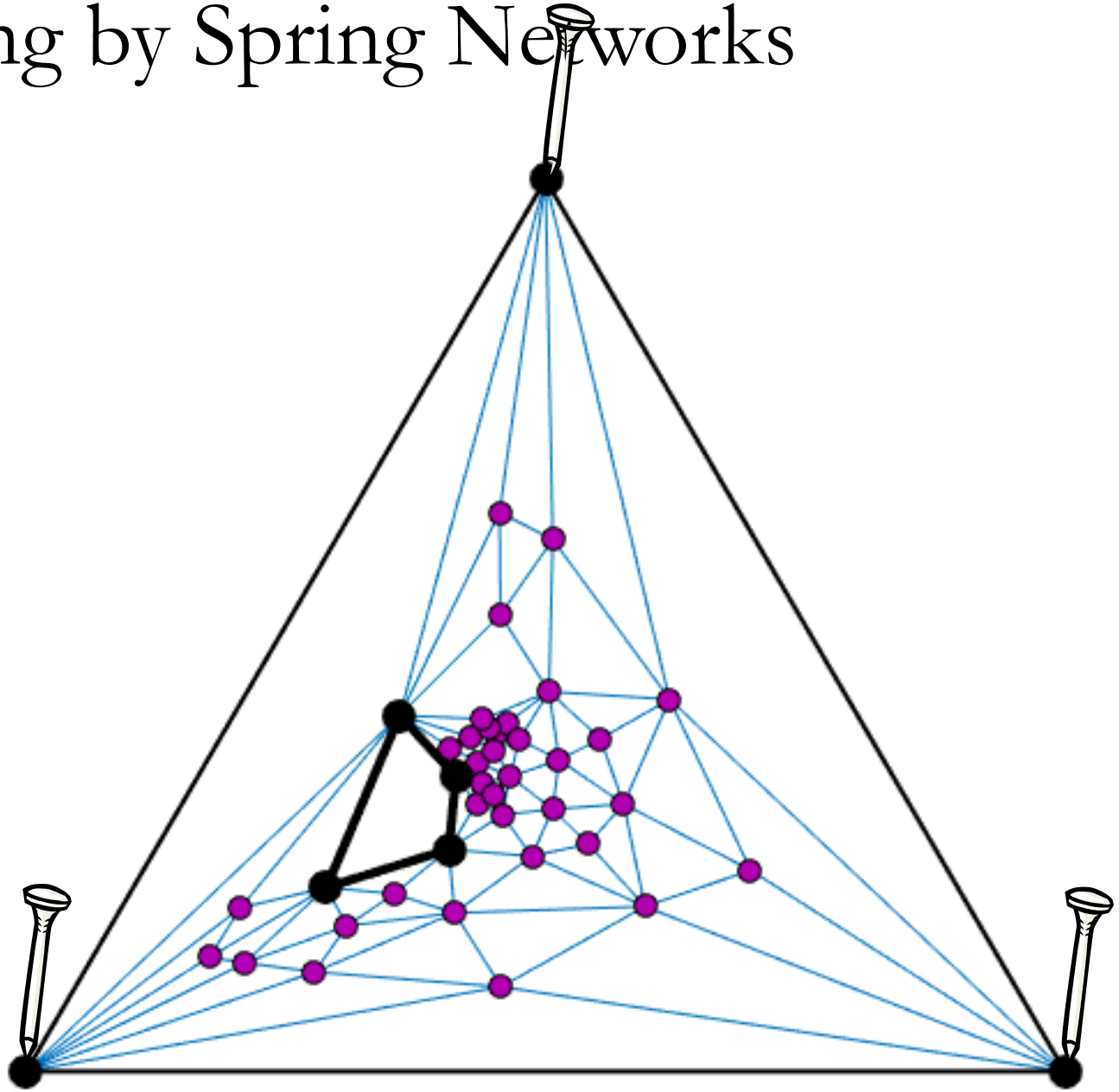


Drawing by Spring Networks

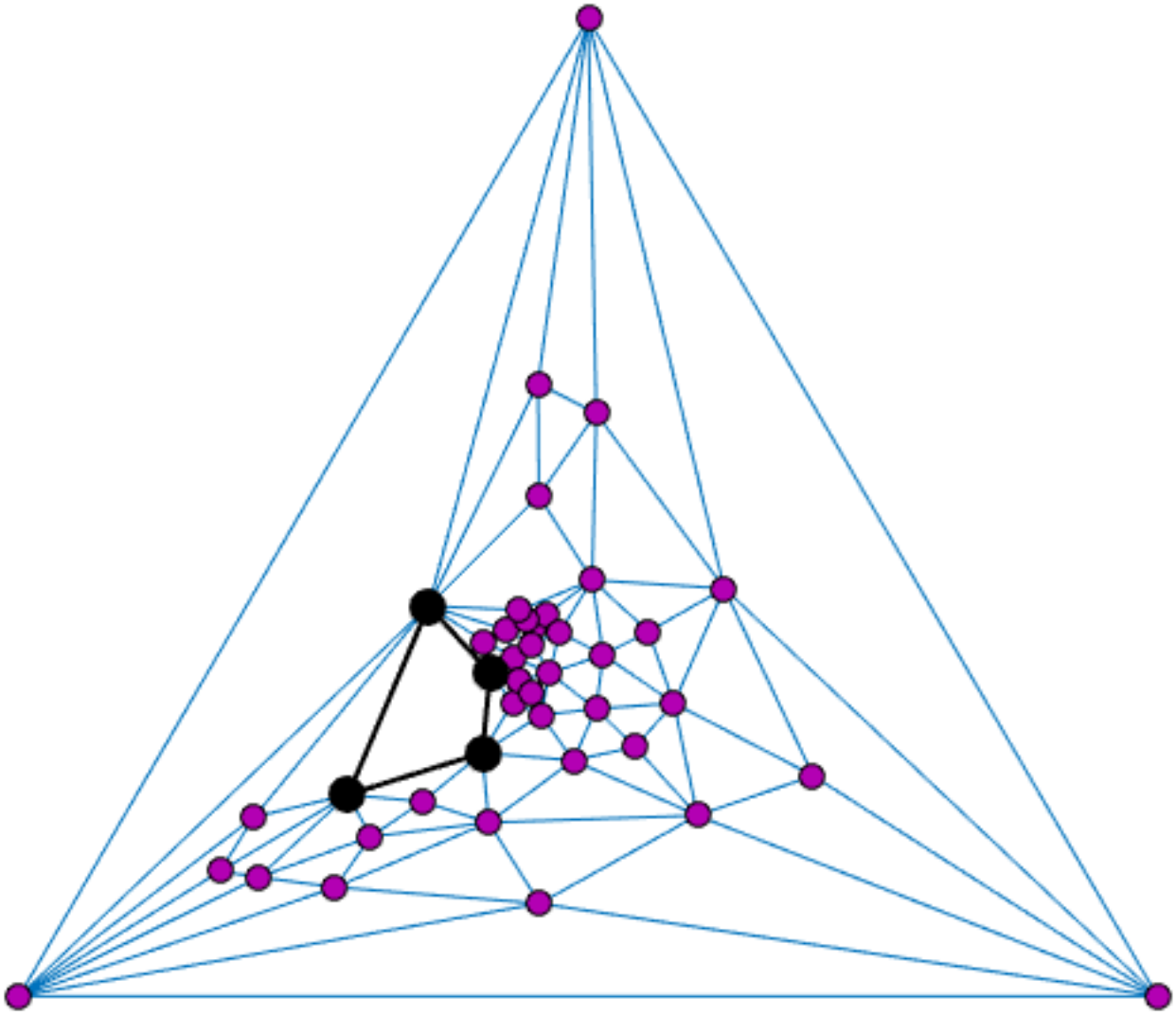
If the graph is planar, then the spring drawing has no crossing edges!



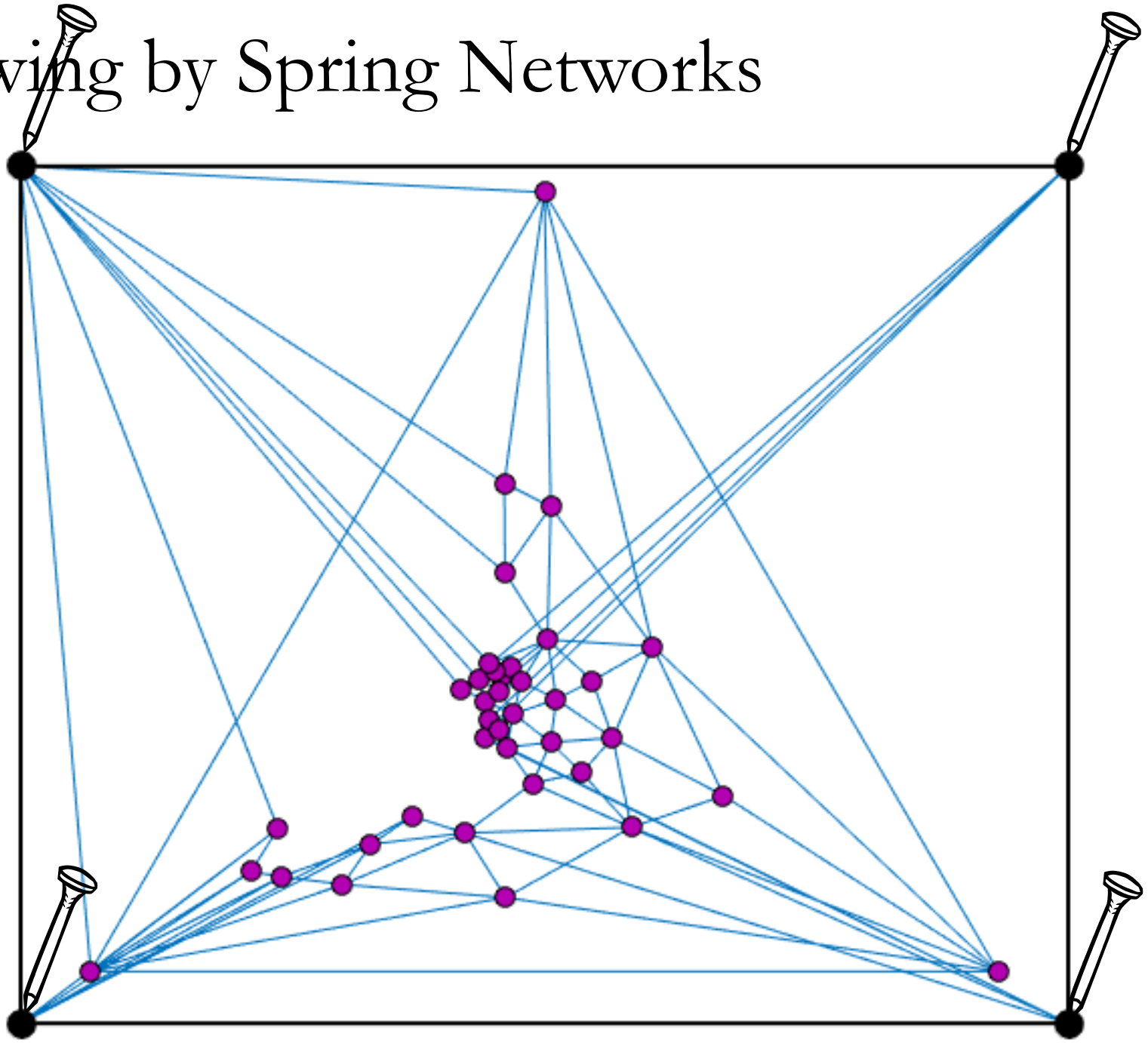
Drawing by Spring Networks



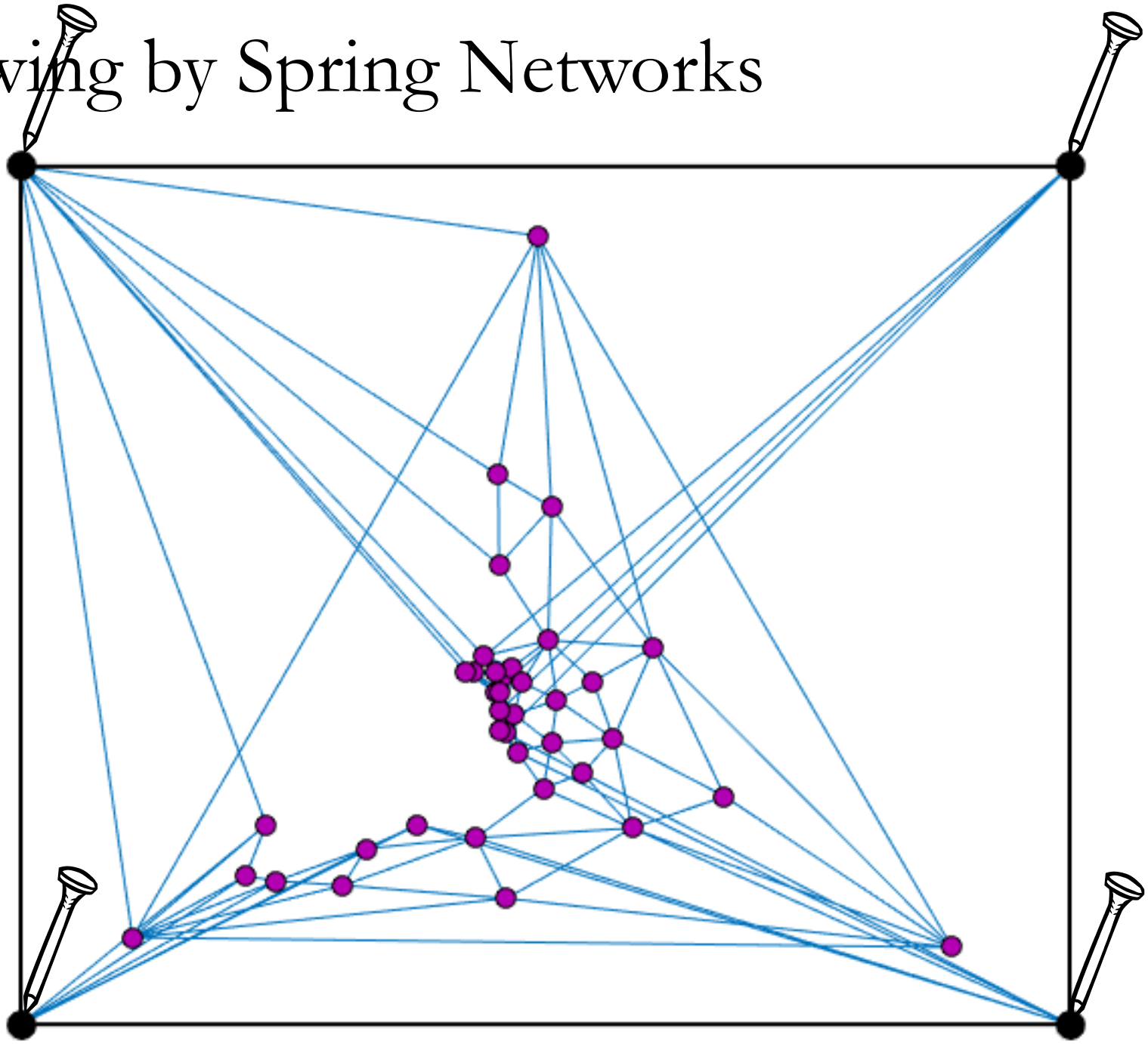
Drawing by Spring Networks



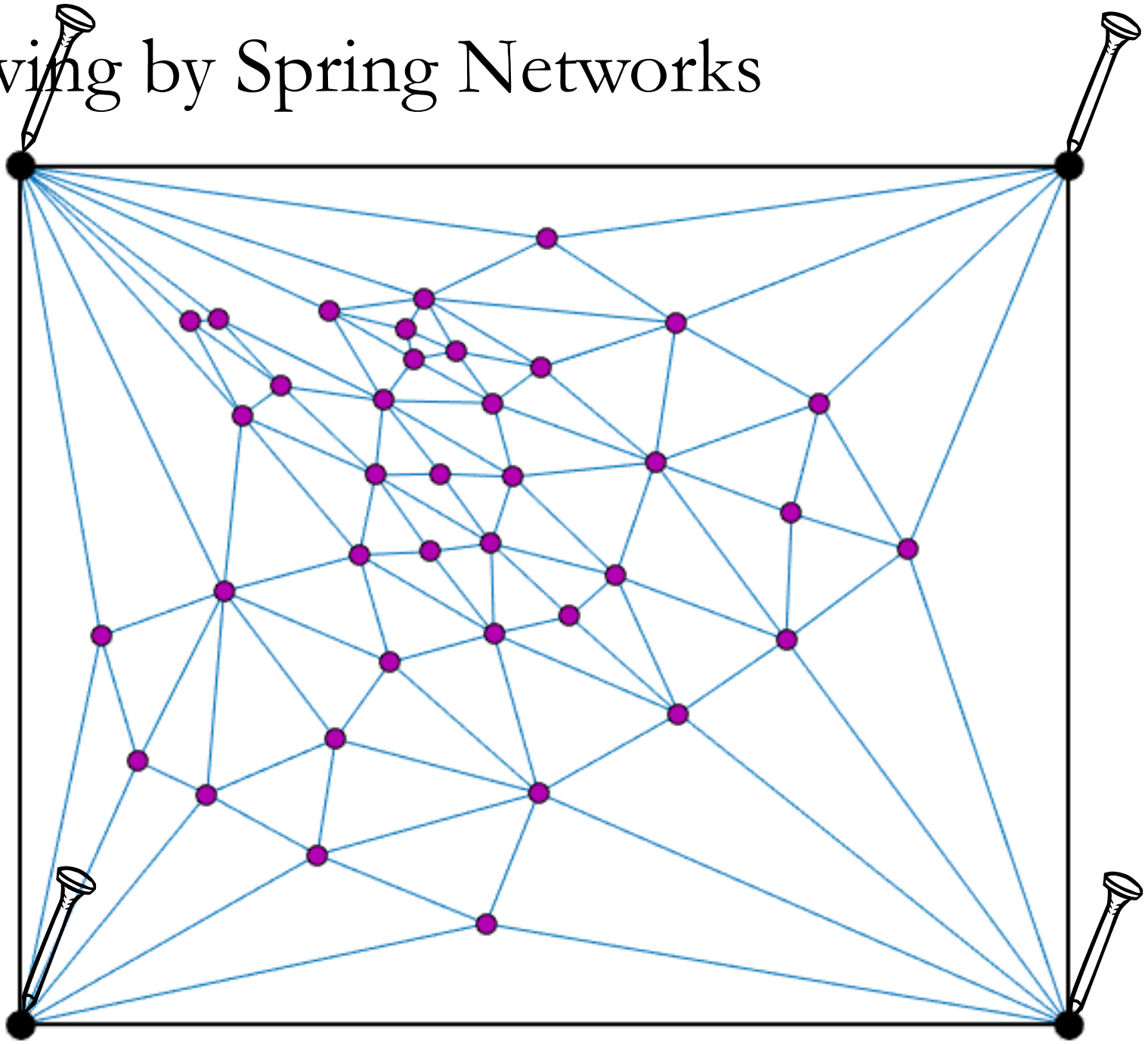
Drawing by Spring Networks



Drawing by Spring Networks

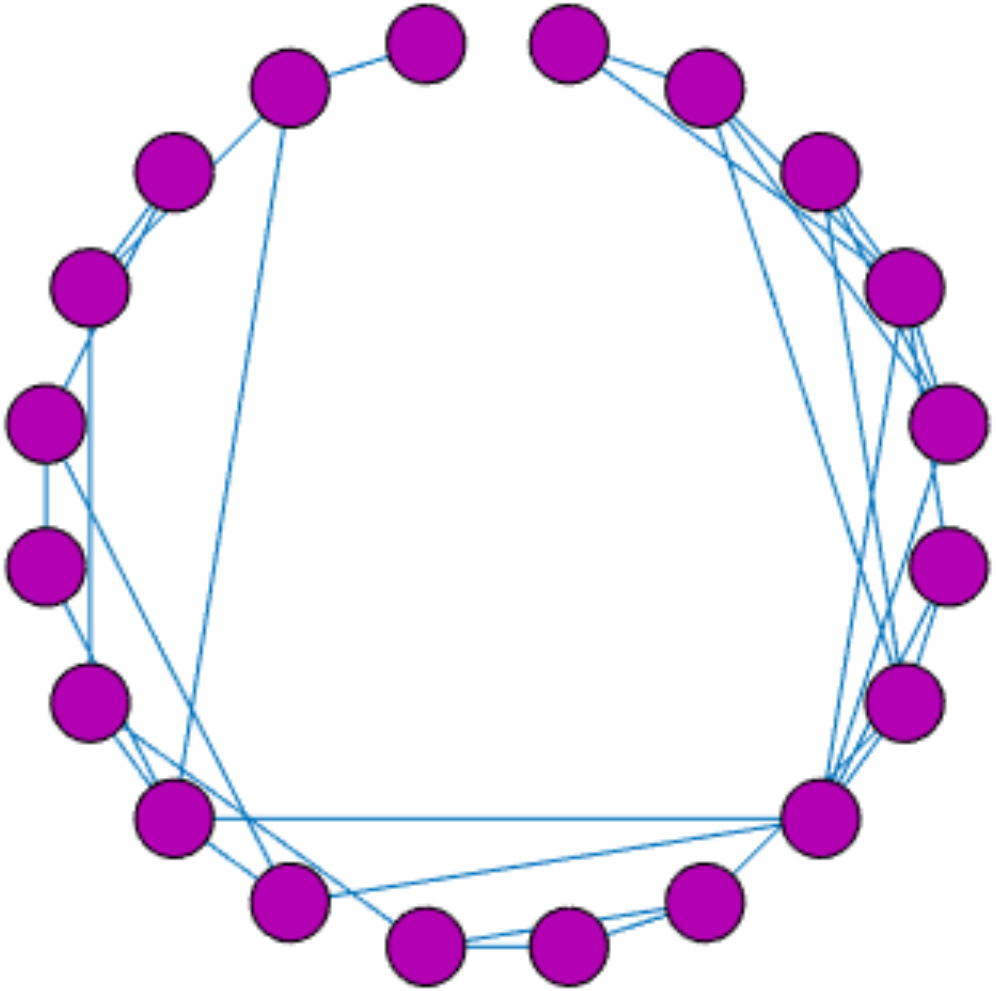


Drawing by Spring Networks



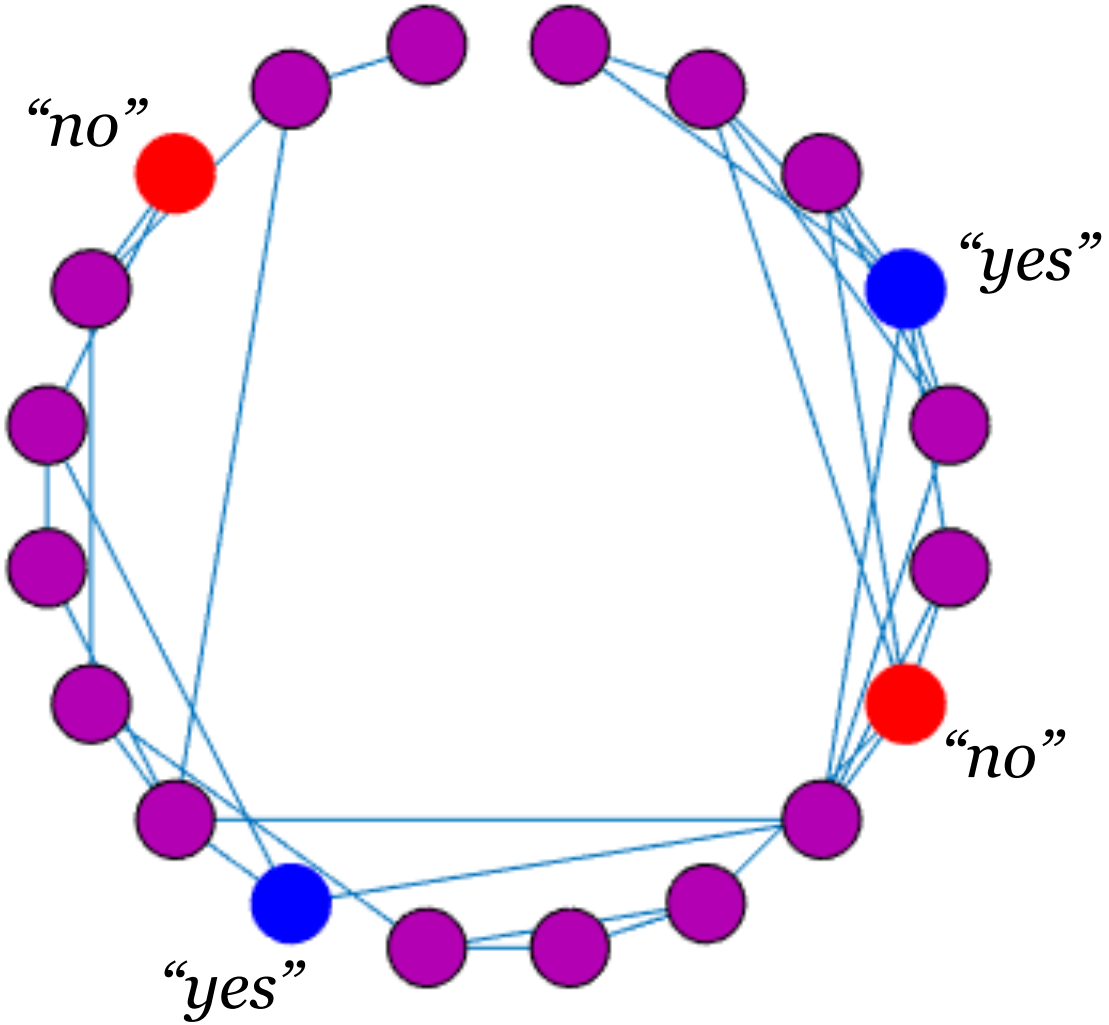
Inference by Spring Networks

Assuming friends are similar, infer from limited data.



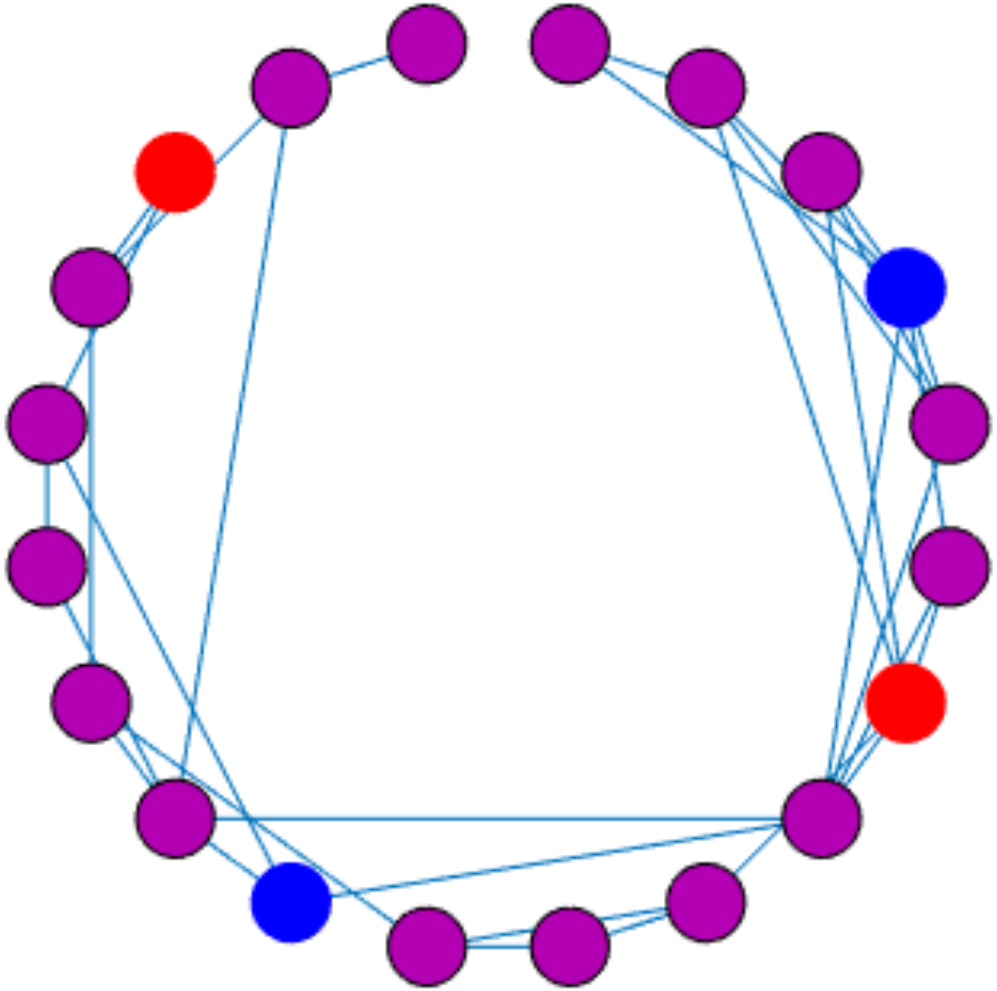
Inference by Spring Networks

Will you donate to X?



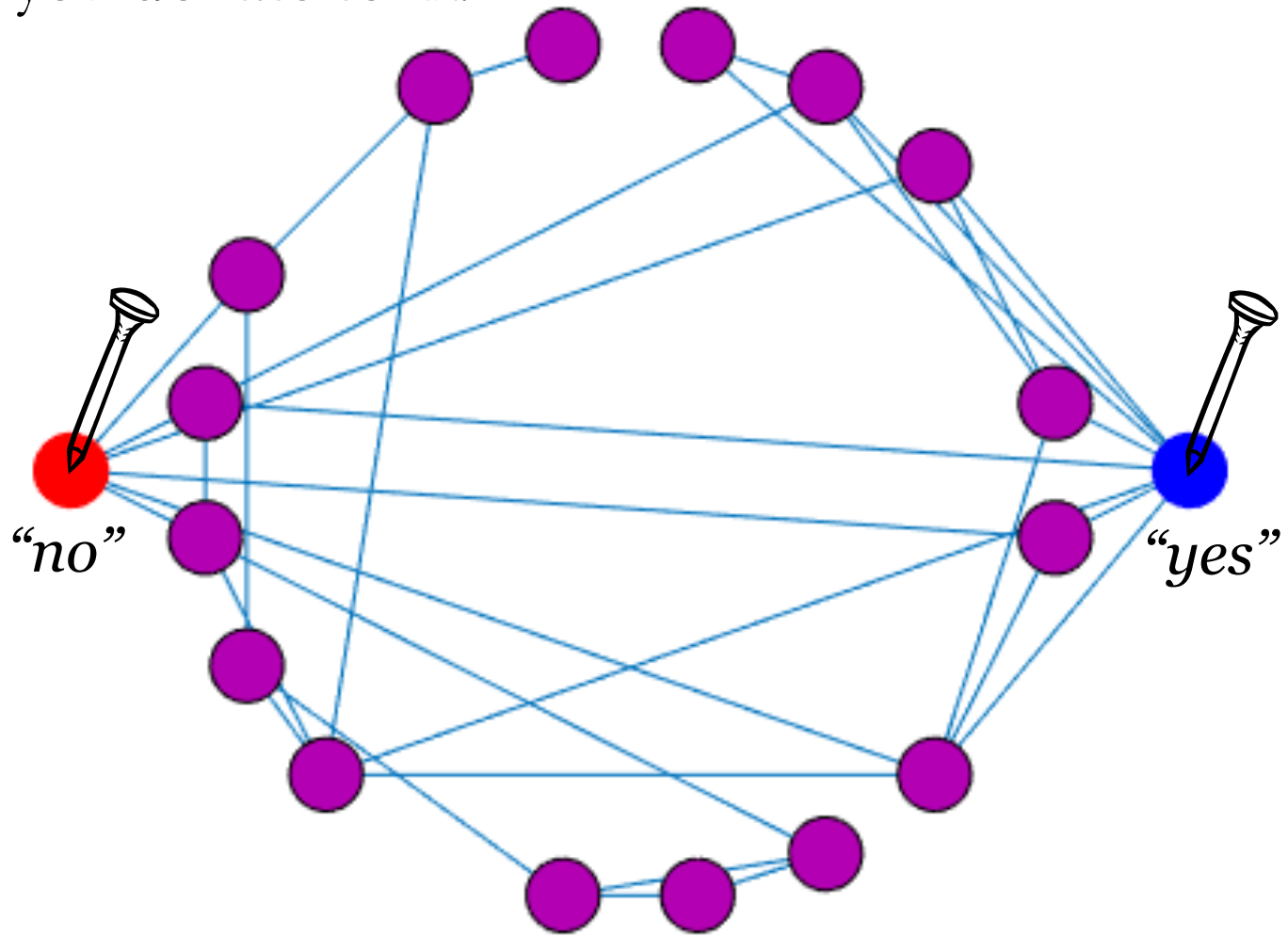
Inference by Spring Networks

Will you donate to X?



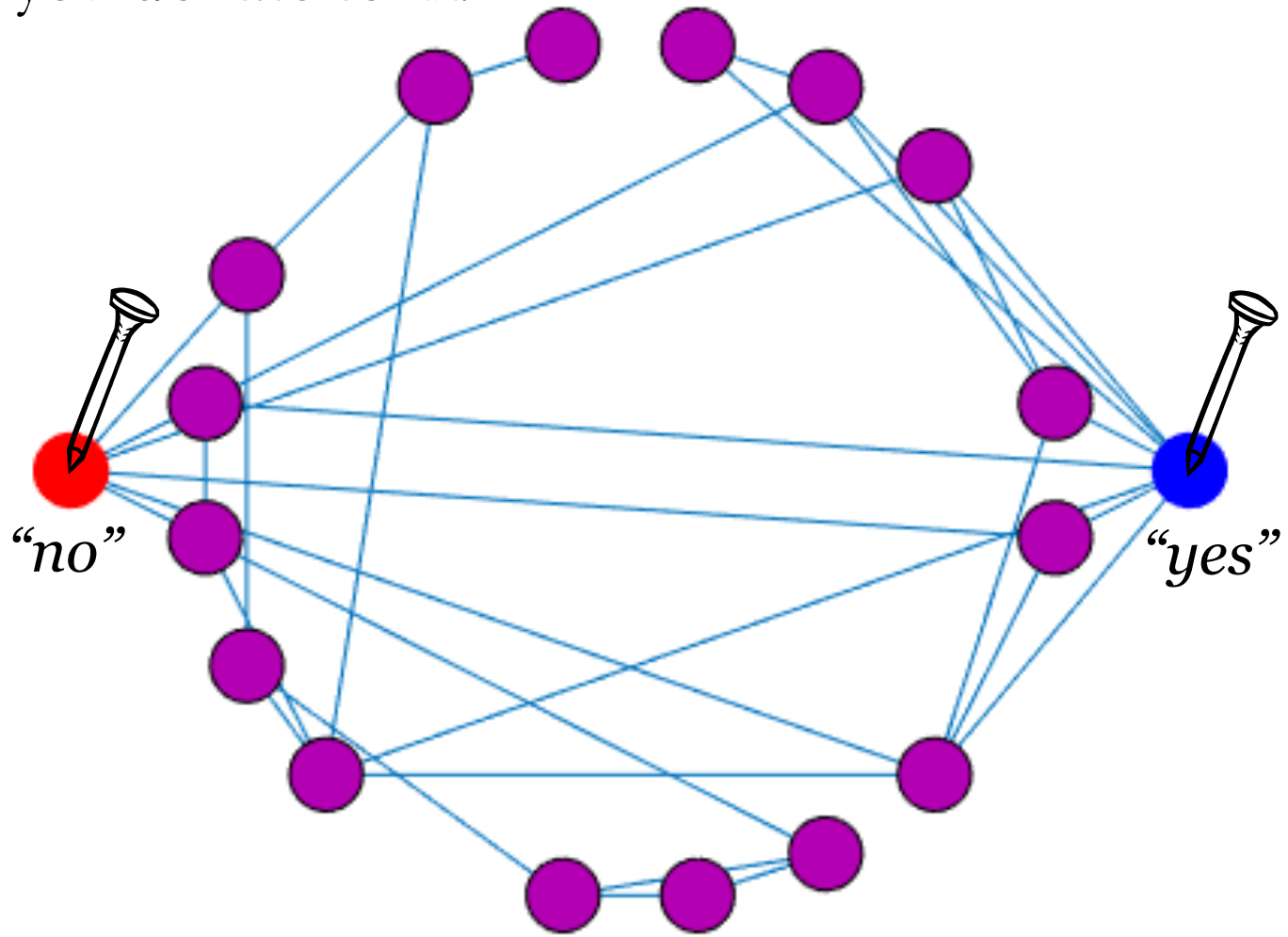
Inference by Spring Networks

Will you donate to X?



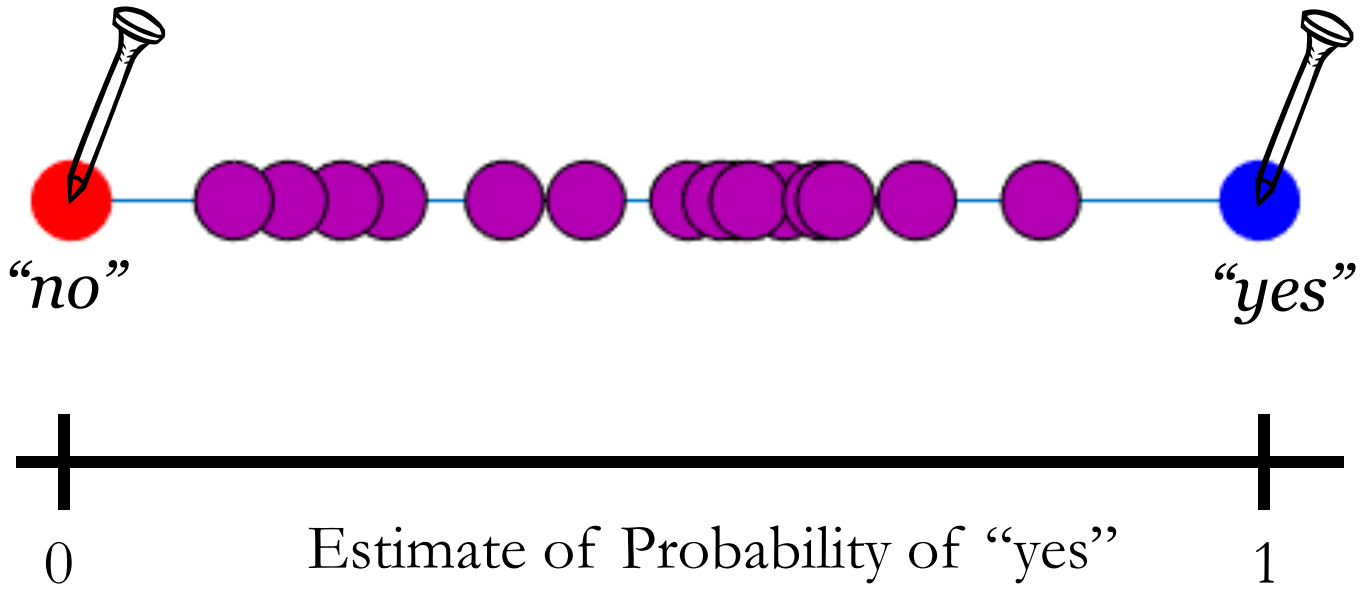
Inference by Spring Networks

Will you donate to X?



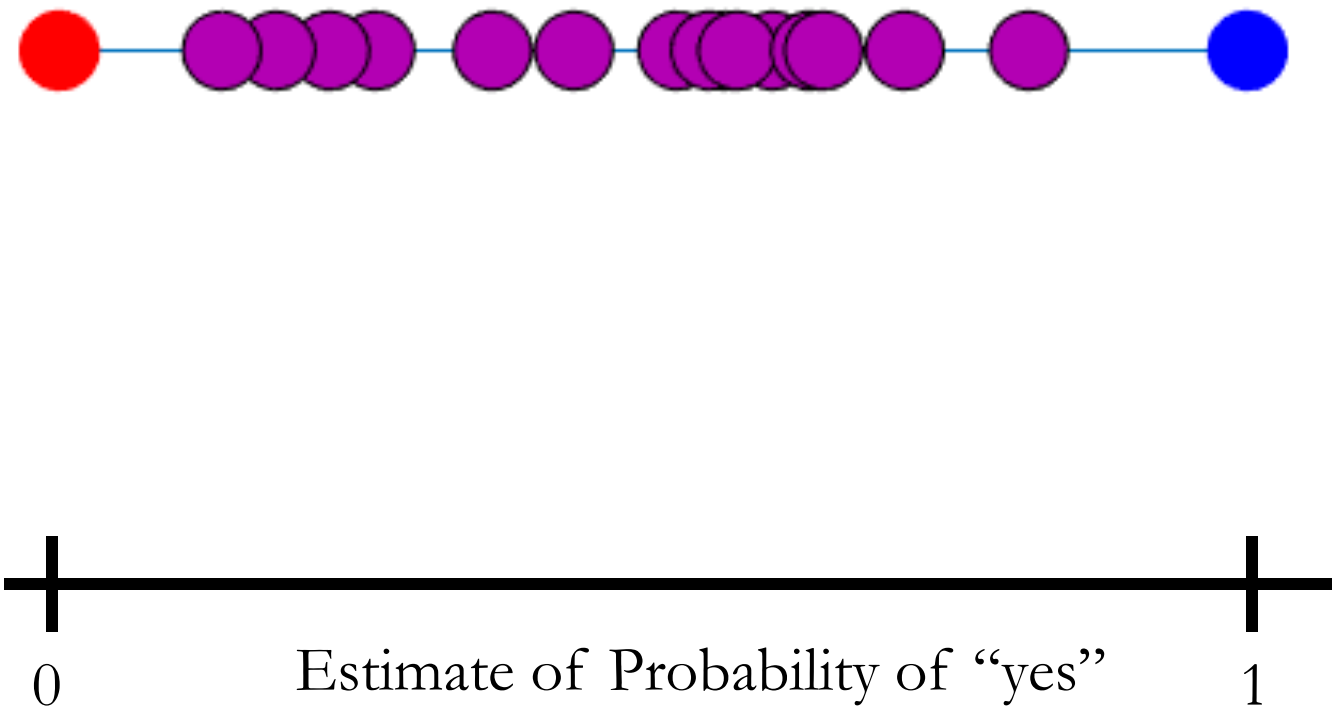
Inference by Spring Networks

Will you donate to X?

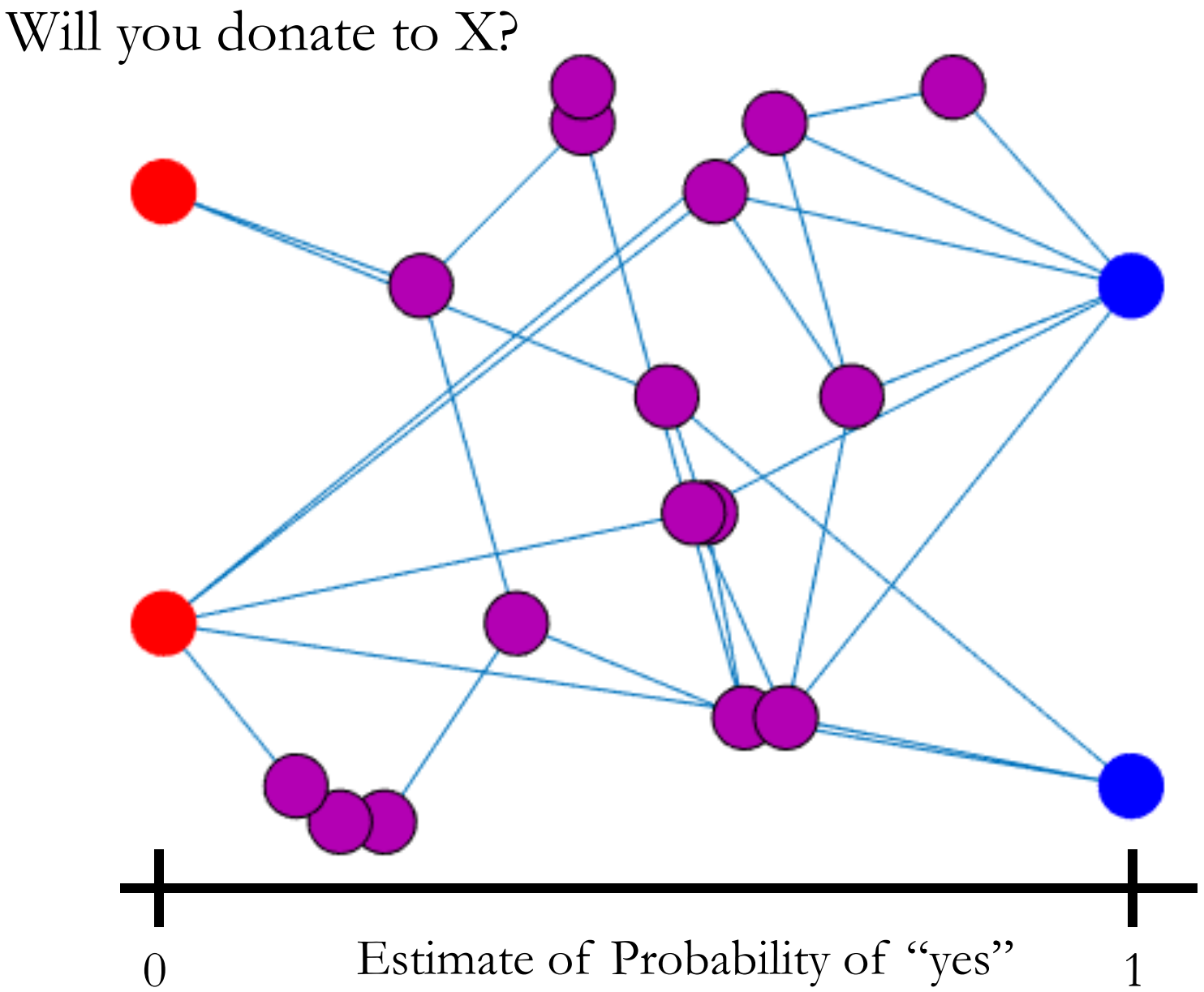


Inference by Spring Networks

Will you donate to X?



Inference by Spring Networks



Some Networks

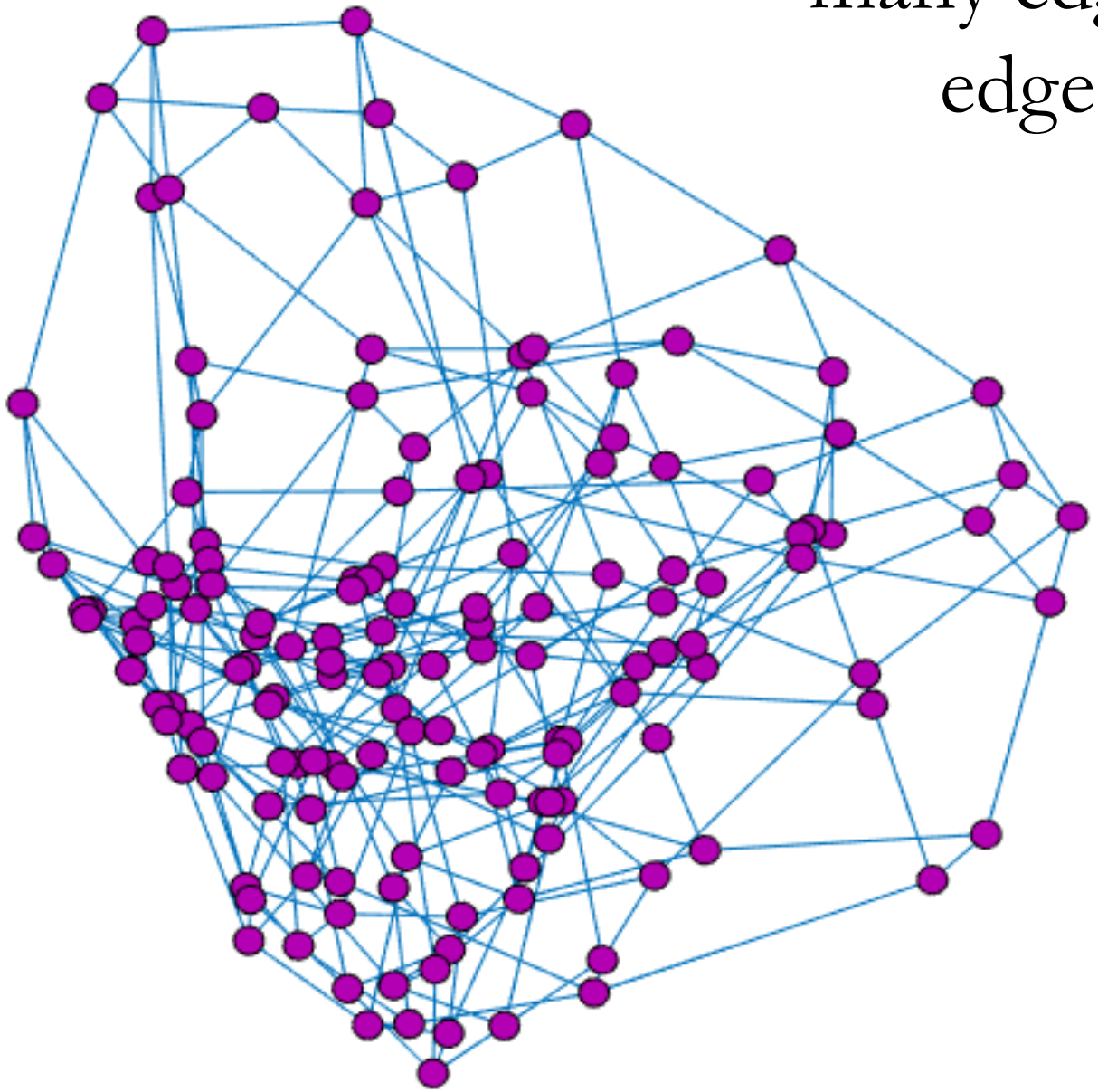
can not

be nicely drawn.

A “bad” drawing: mostly edges,

many edge crossings

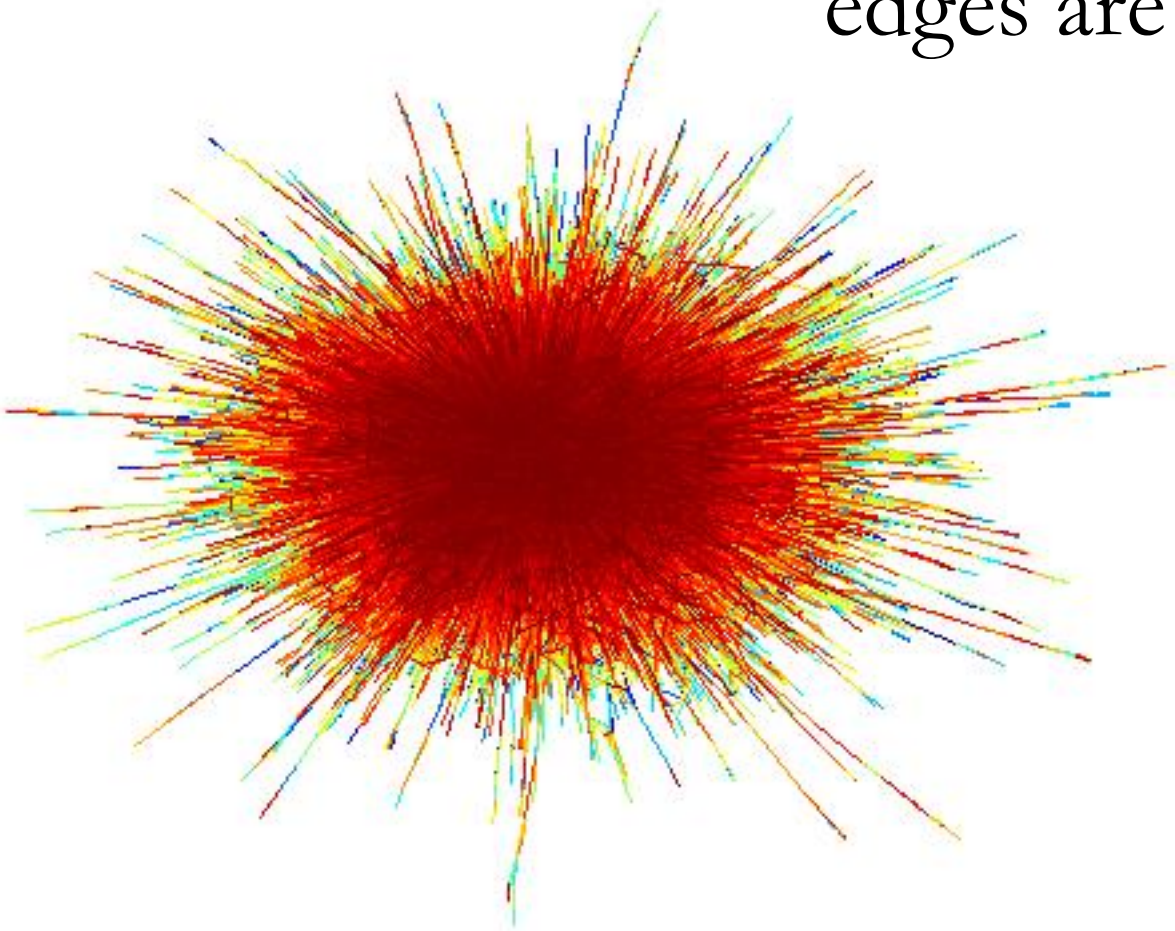
edges are long



A “bad” drawing: mostly edges,

many edge crossings

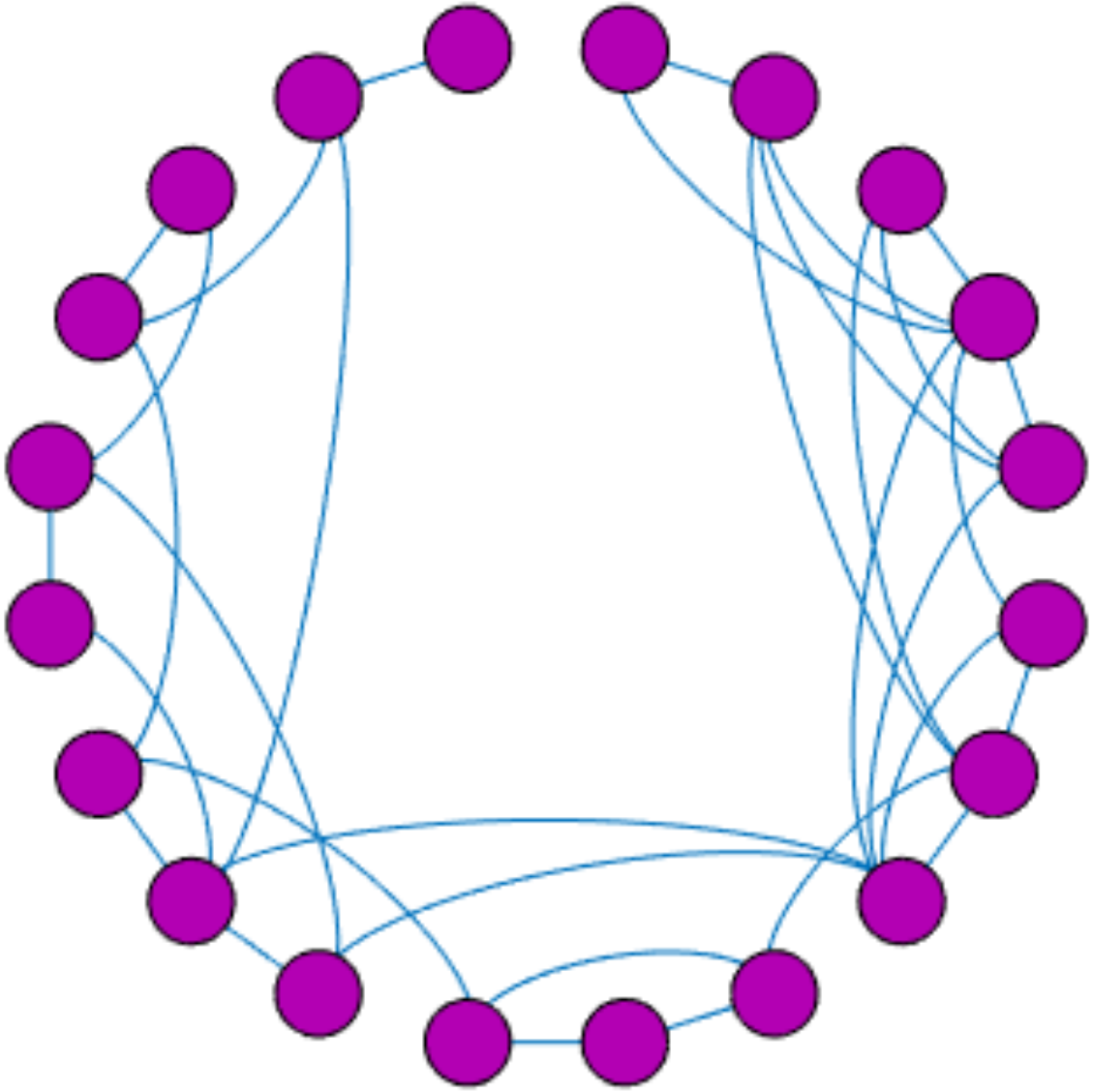
edges are long



Some Networks
can not
be nicely drawn.

So,
we group their
nodes into clusters.

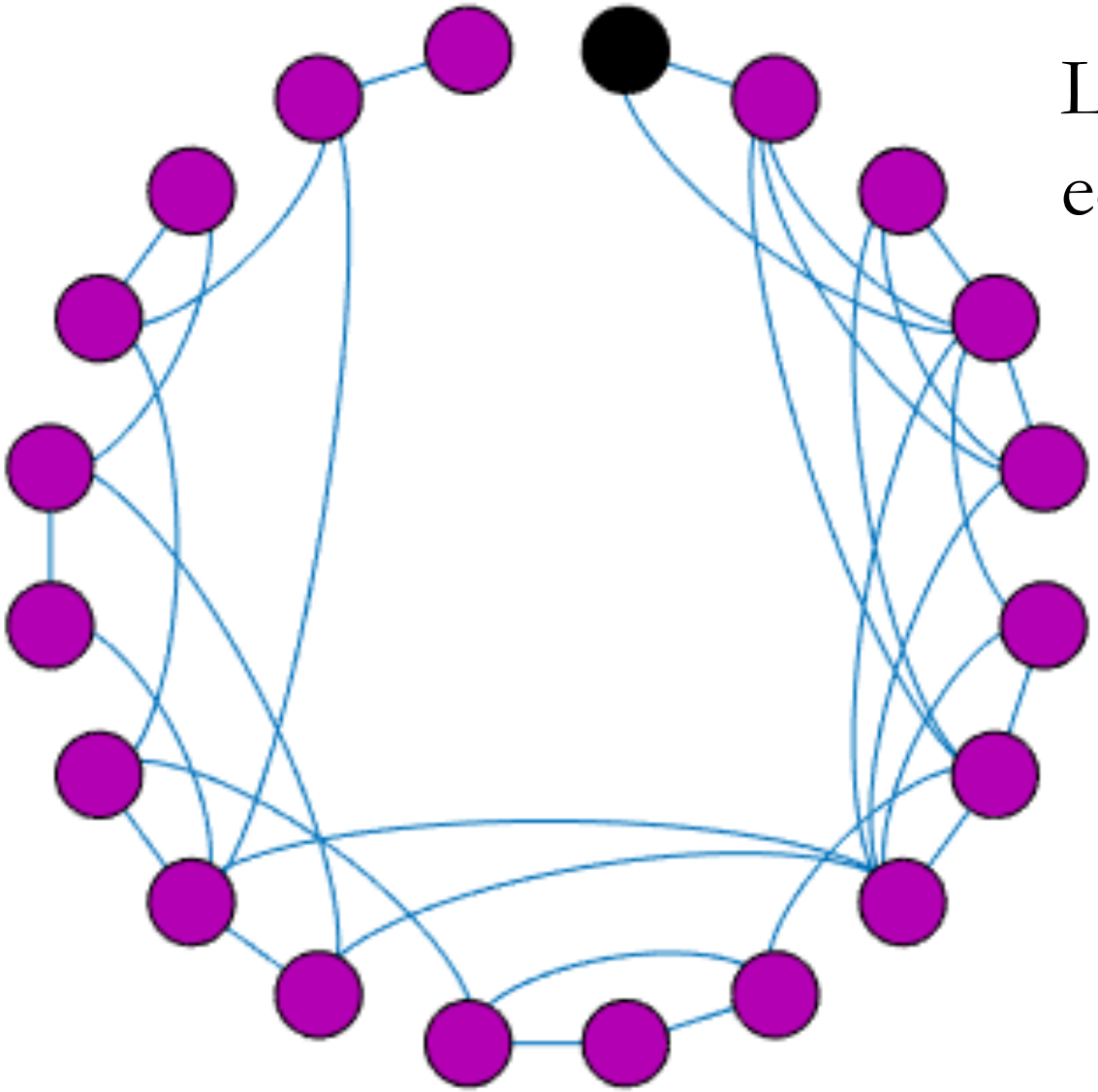
Diffusion in Graphs



Diffusion in Graphs

Put stuff at a node.

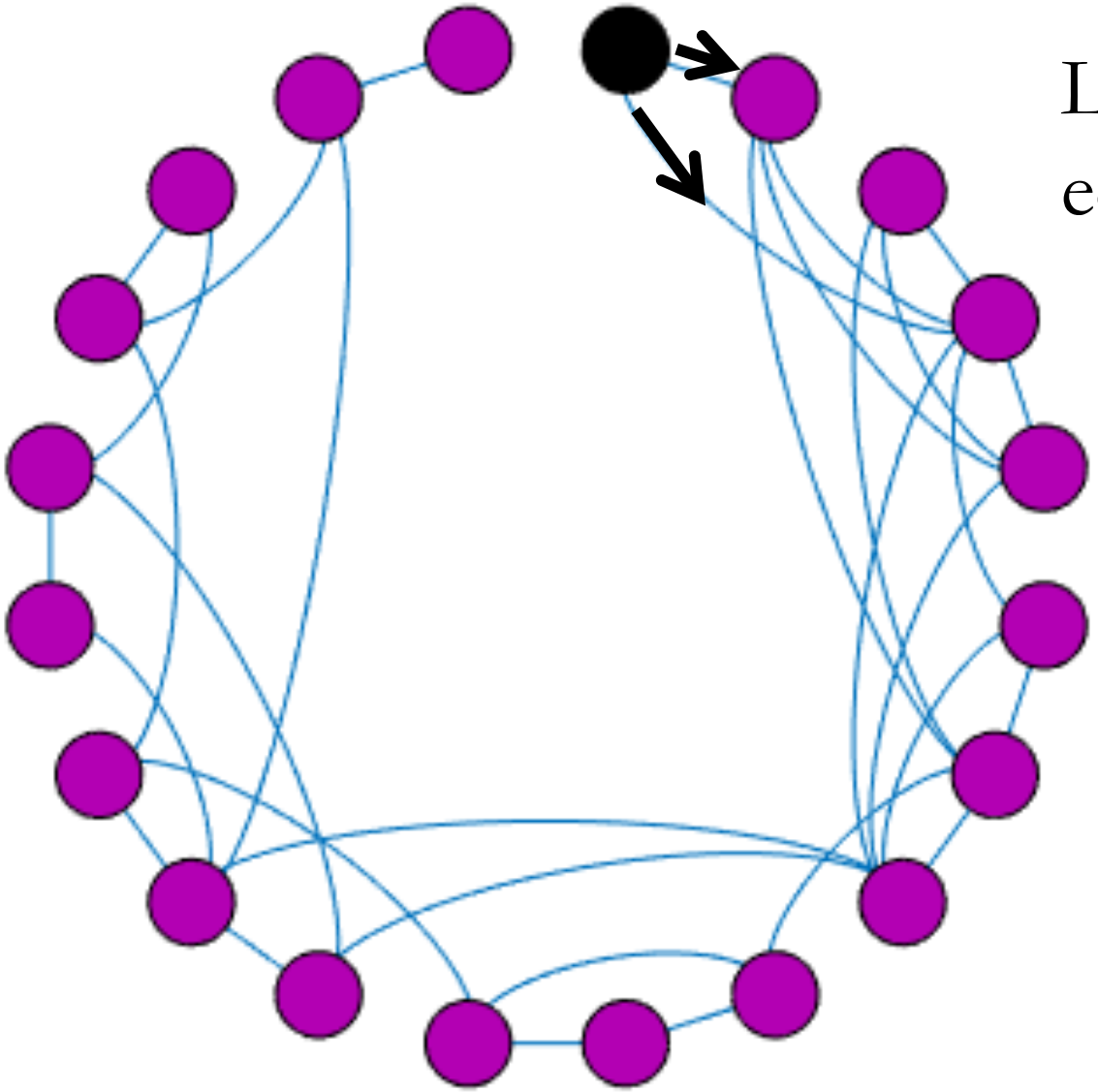
Let stuff flow along edges to other nodes



Diffusion in Graphs

Put stuff at a node.

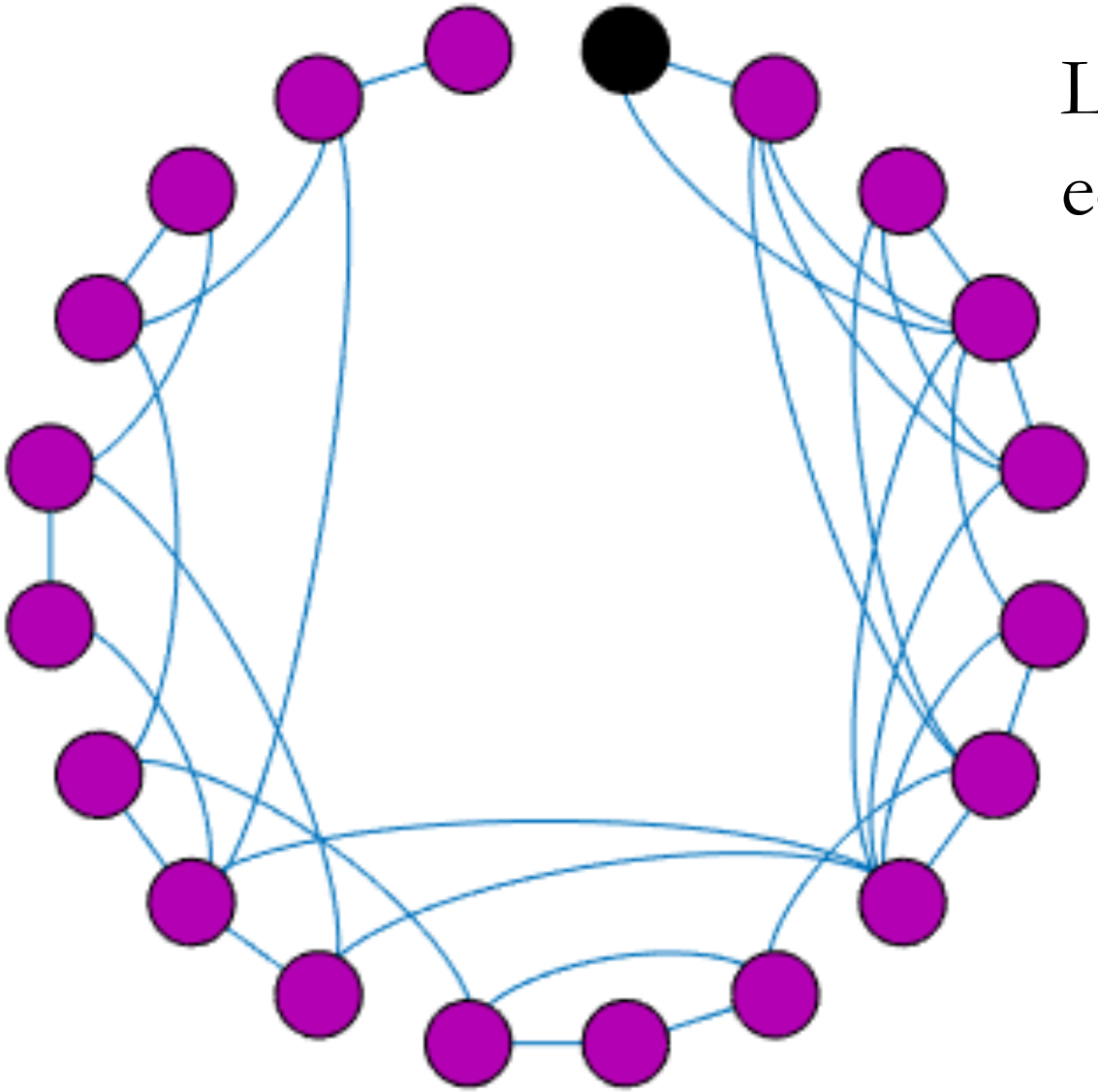
Let stuff flow along edges to other nodes



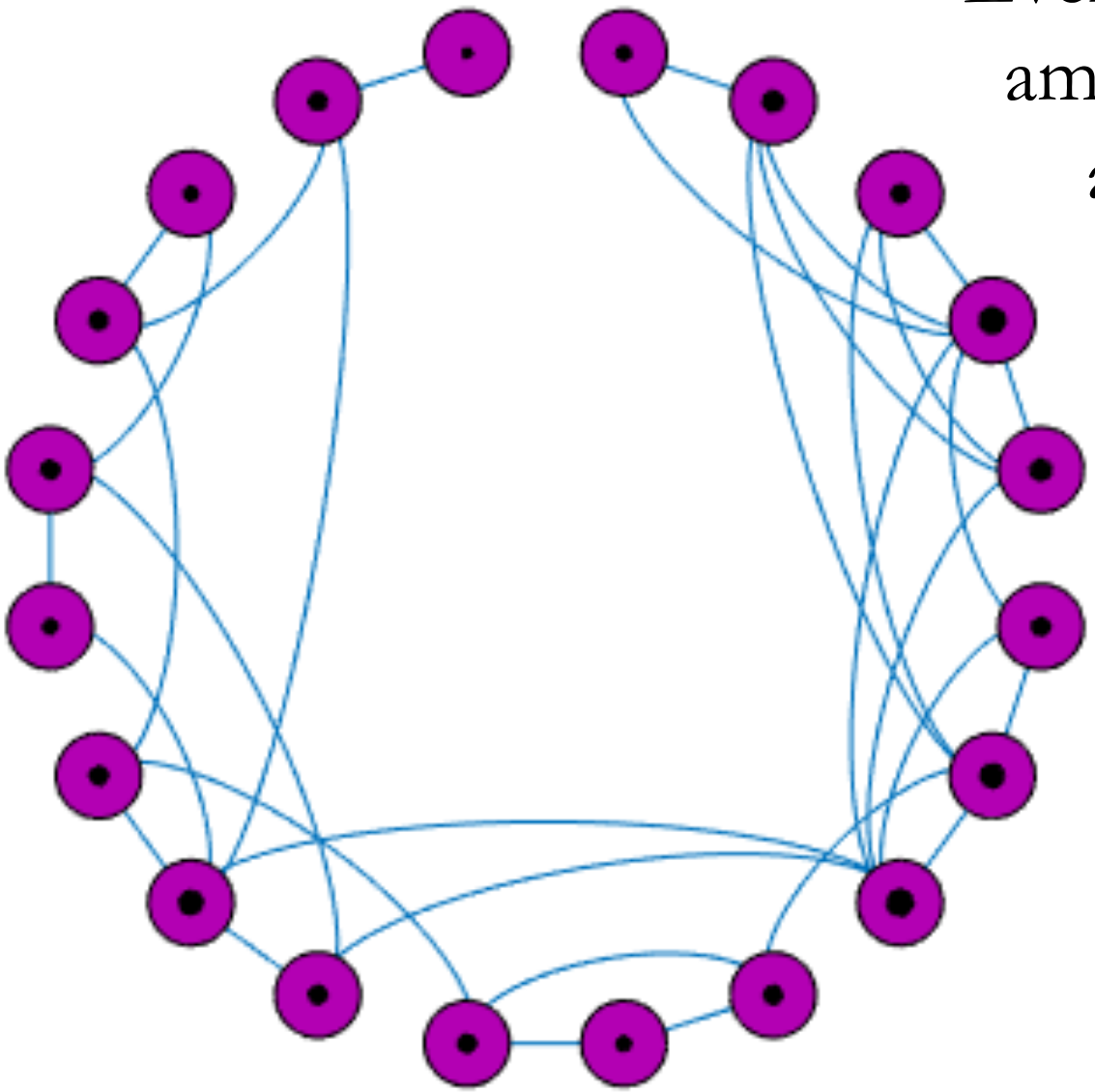
Diffusion in Graphs

Put stuff at a node.

Let stuff flow along edges to other nodes



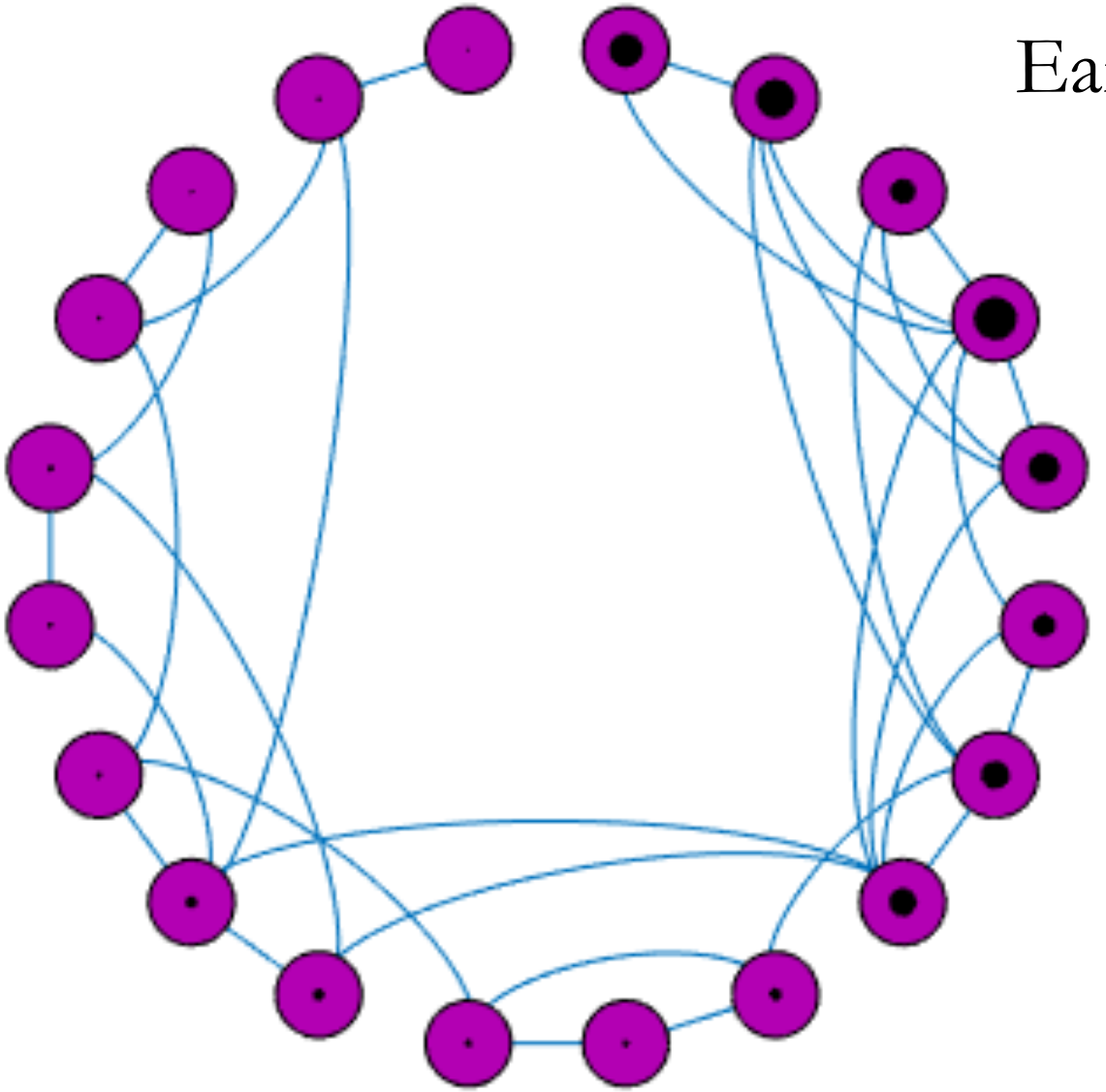
Diffusion in Graphs



Eventually…
amount of stuff
at nodes

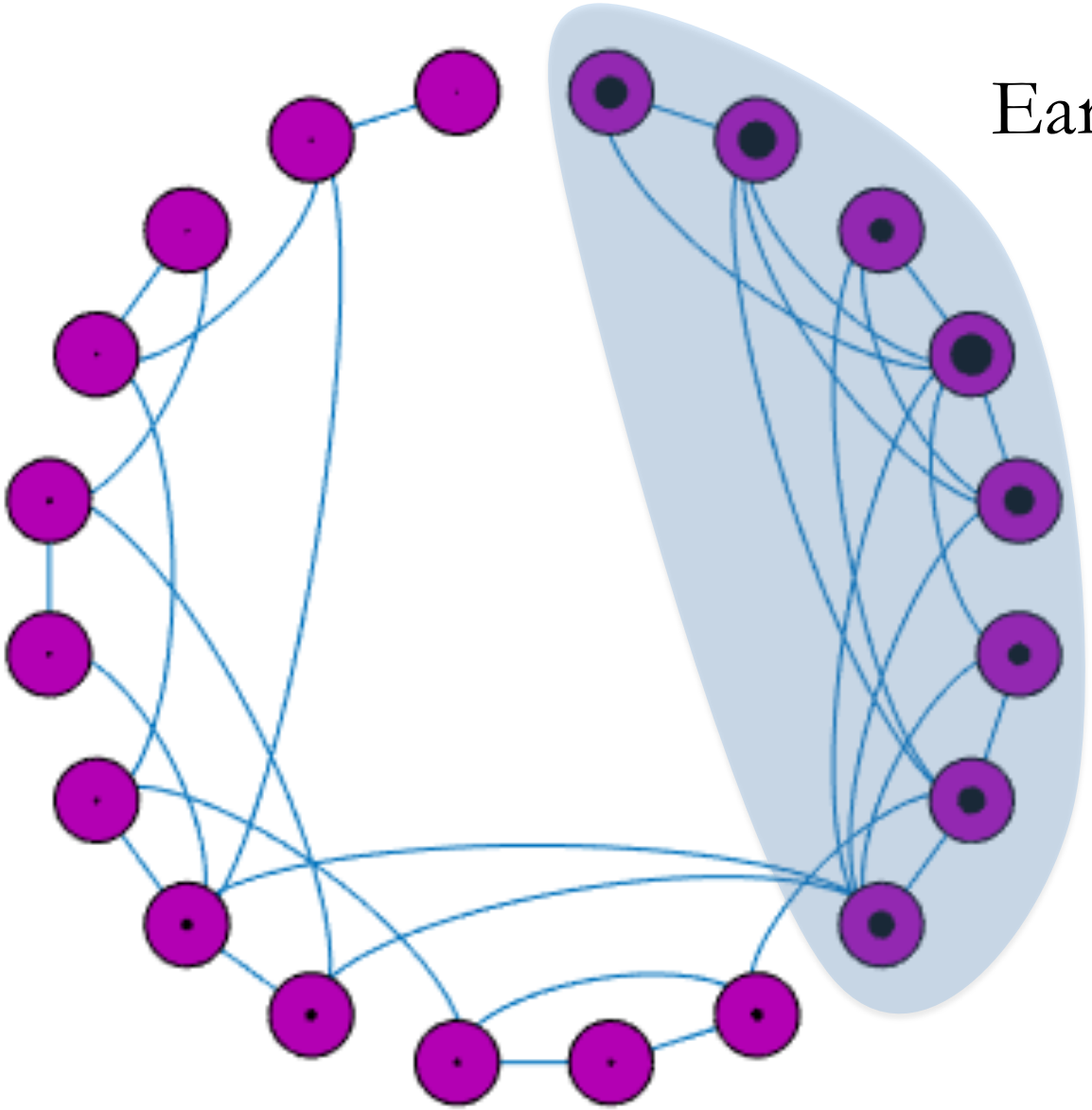
is proportional to
their number
of edges

Clustering by Diffusion in Graphs



Earlier in the process the nodes with the most stuff are clusters.

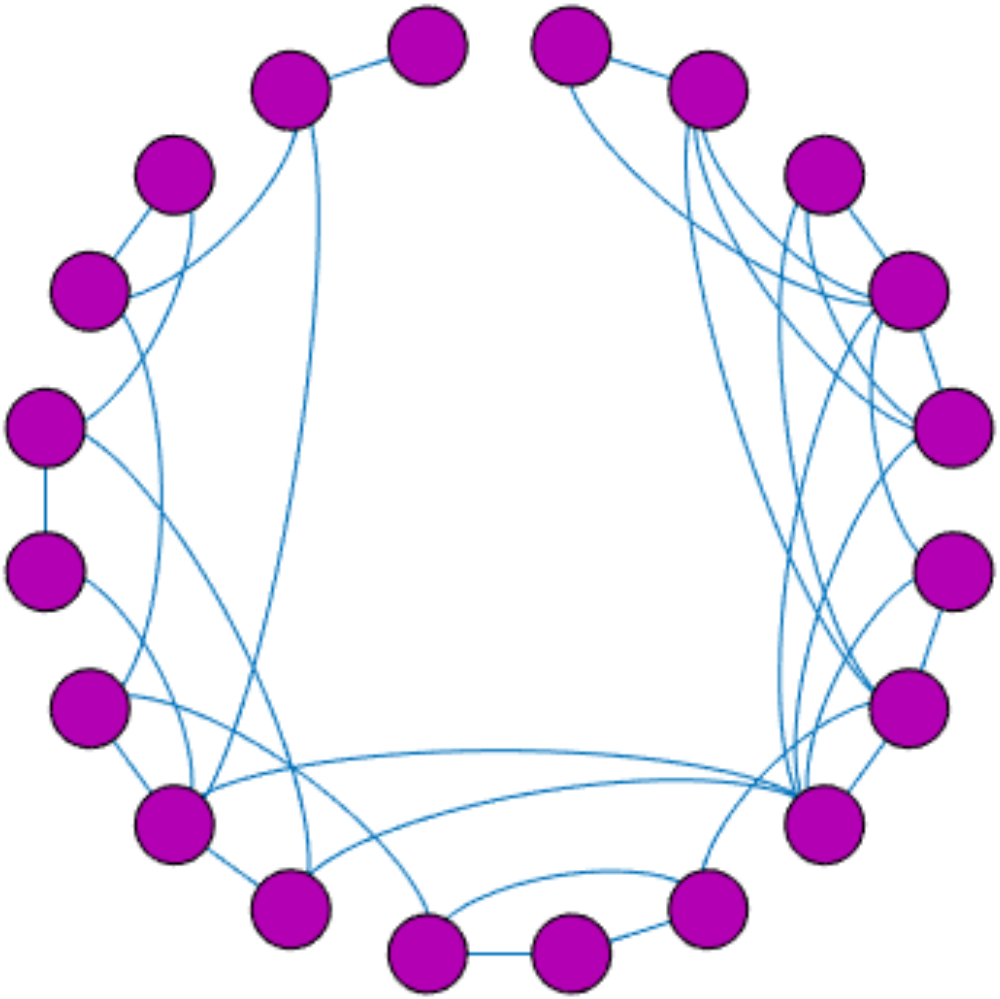
Clustering by Diffusion in Graphs



Earlier in the process the nodes with the most stuff are clusters.

If there are clusters, this finds them!

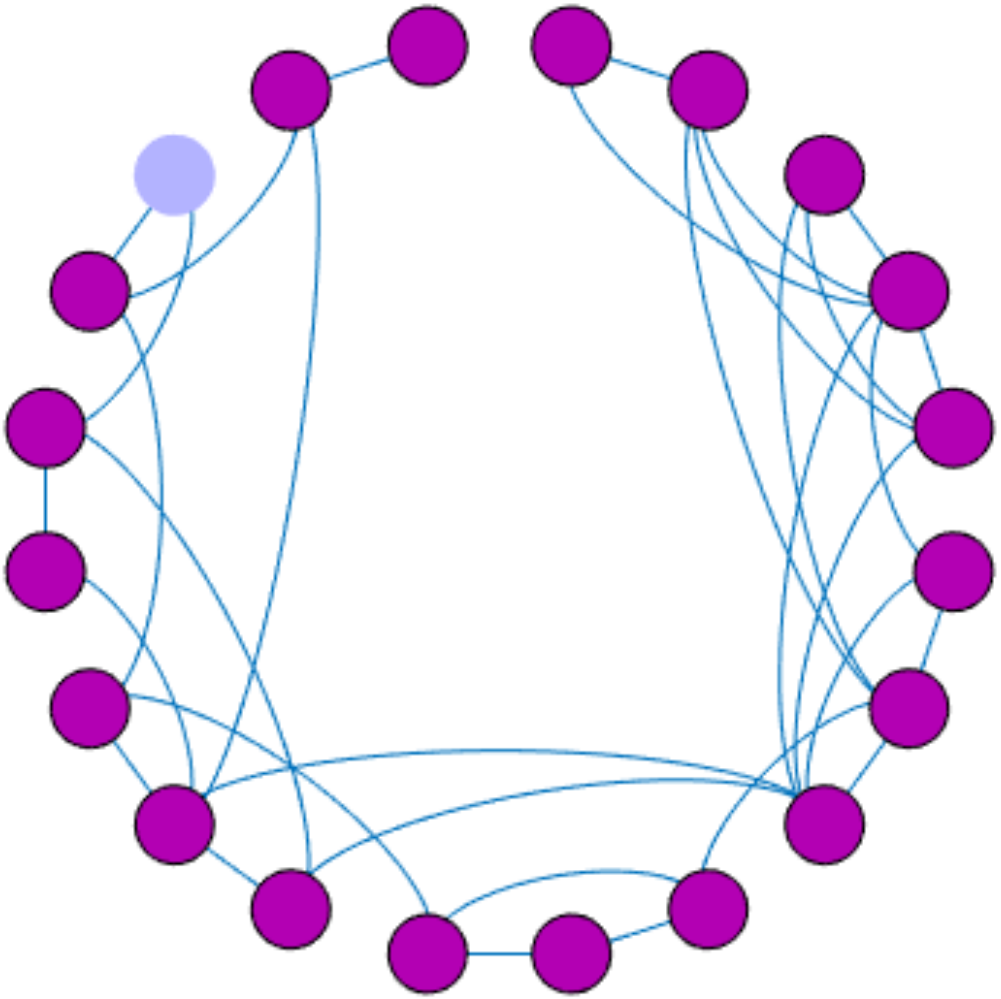
Importance: Personal PageRank



Spill paint at a node.

Paint both spreads and dries.

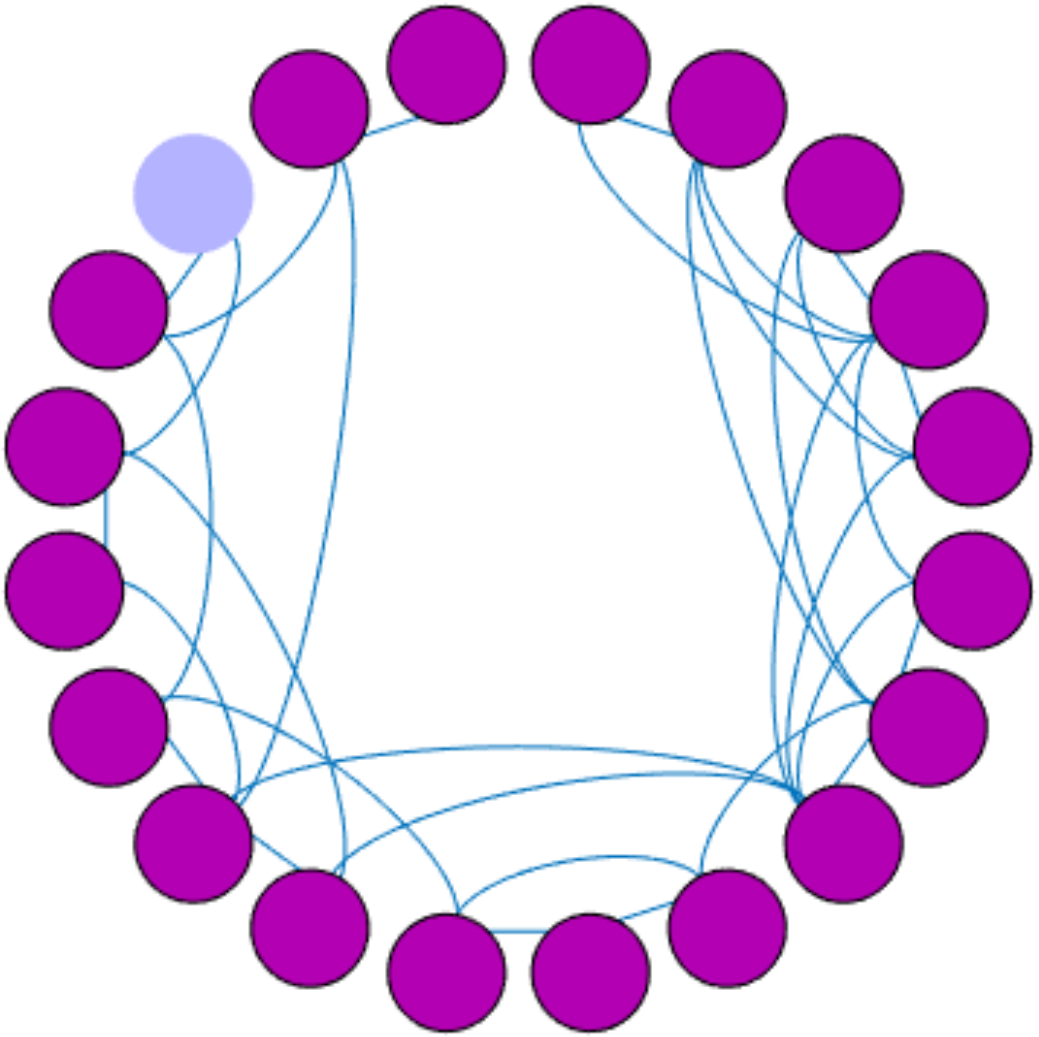
Importance: Personal PageRank



Spill paint at a node.

Paint both spreads and dries.

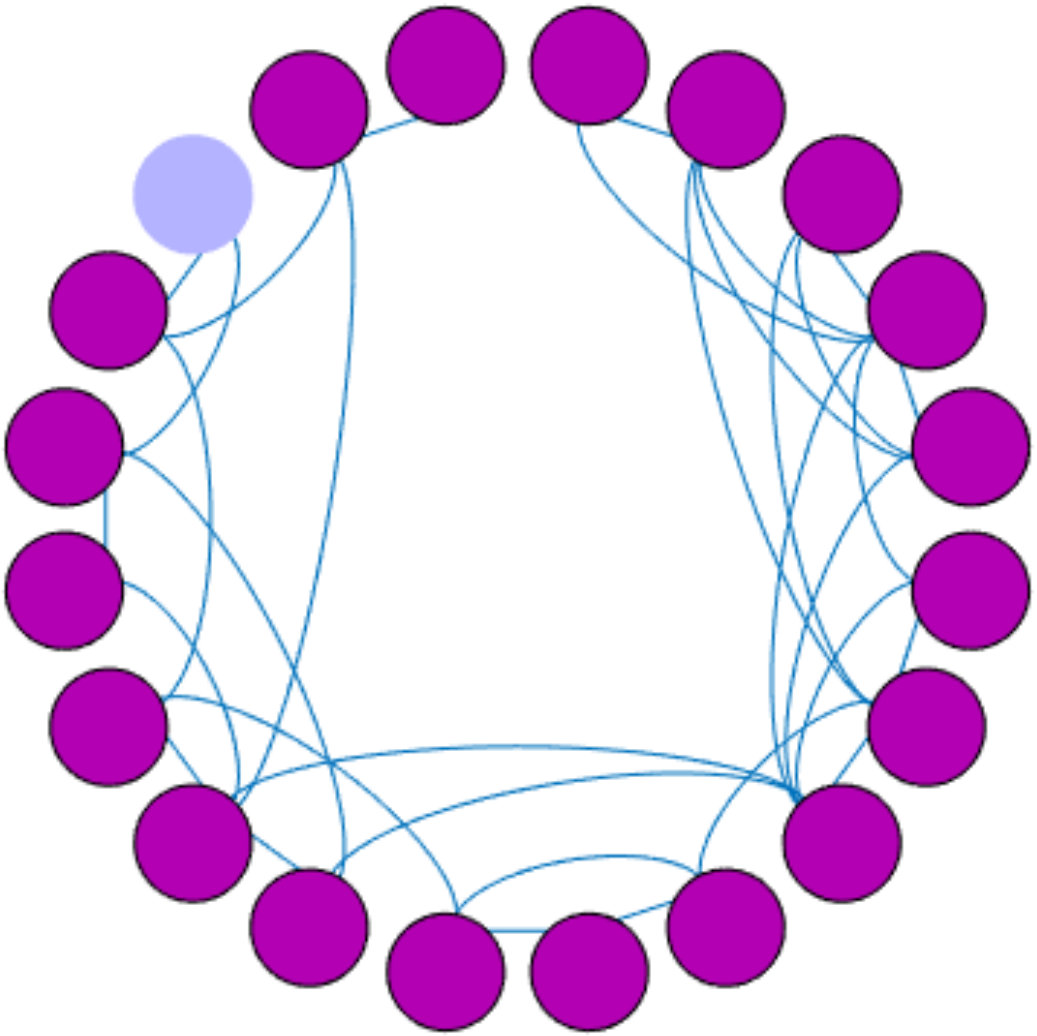
Importance: Personal PageRank



Spill paint at a node.

Paint both spreads and dries.

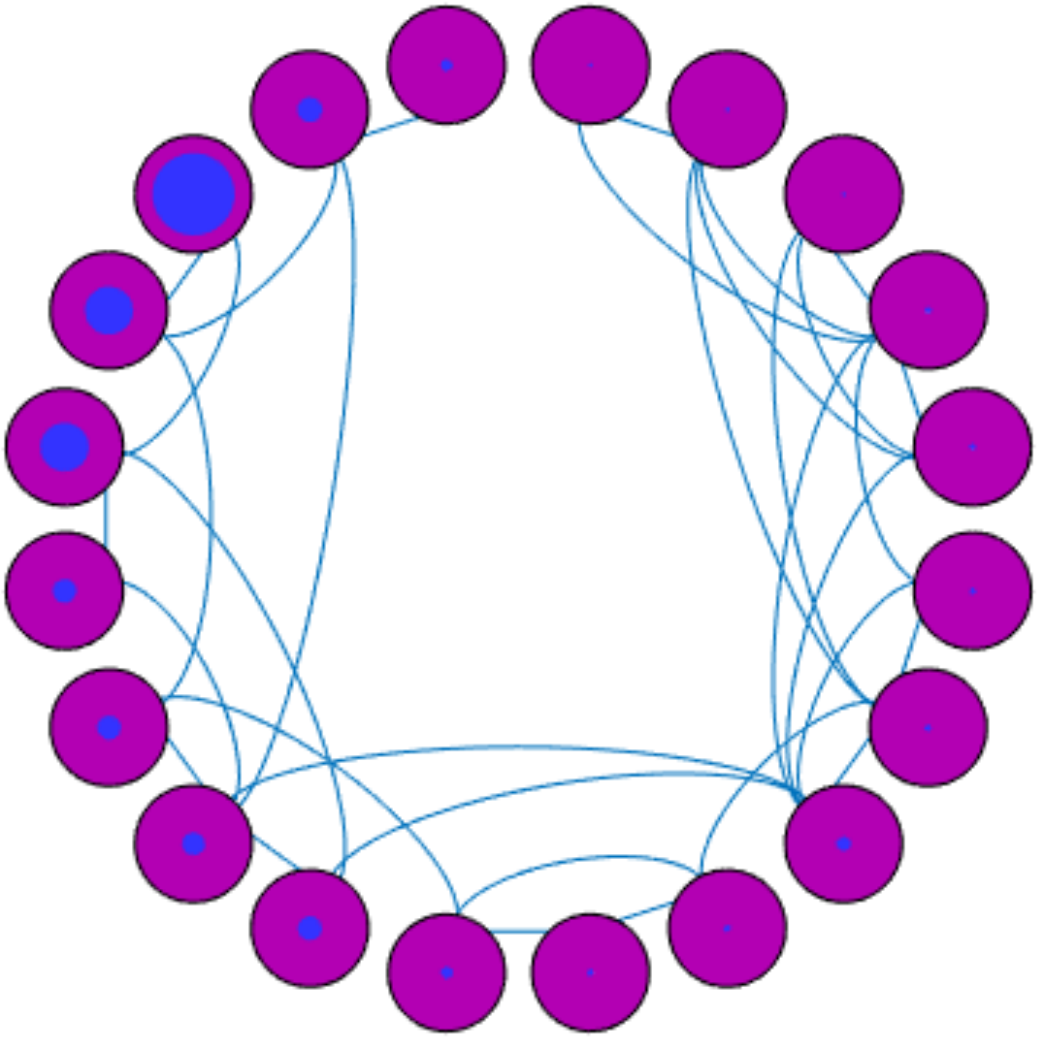
Importance: Personal PageRank



Spill paint at a node.
Paint both spreads and dries.

Amount of dried paint, measures importance relative to initial node

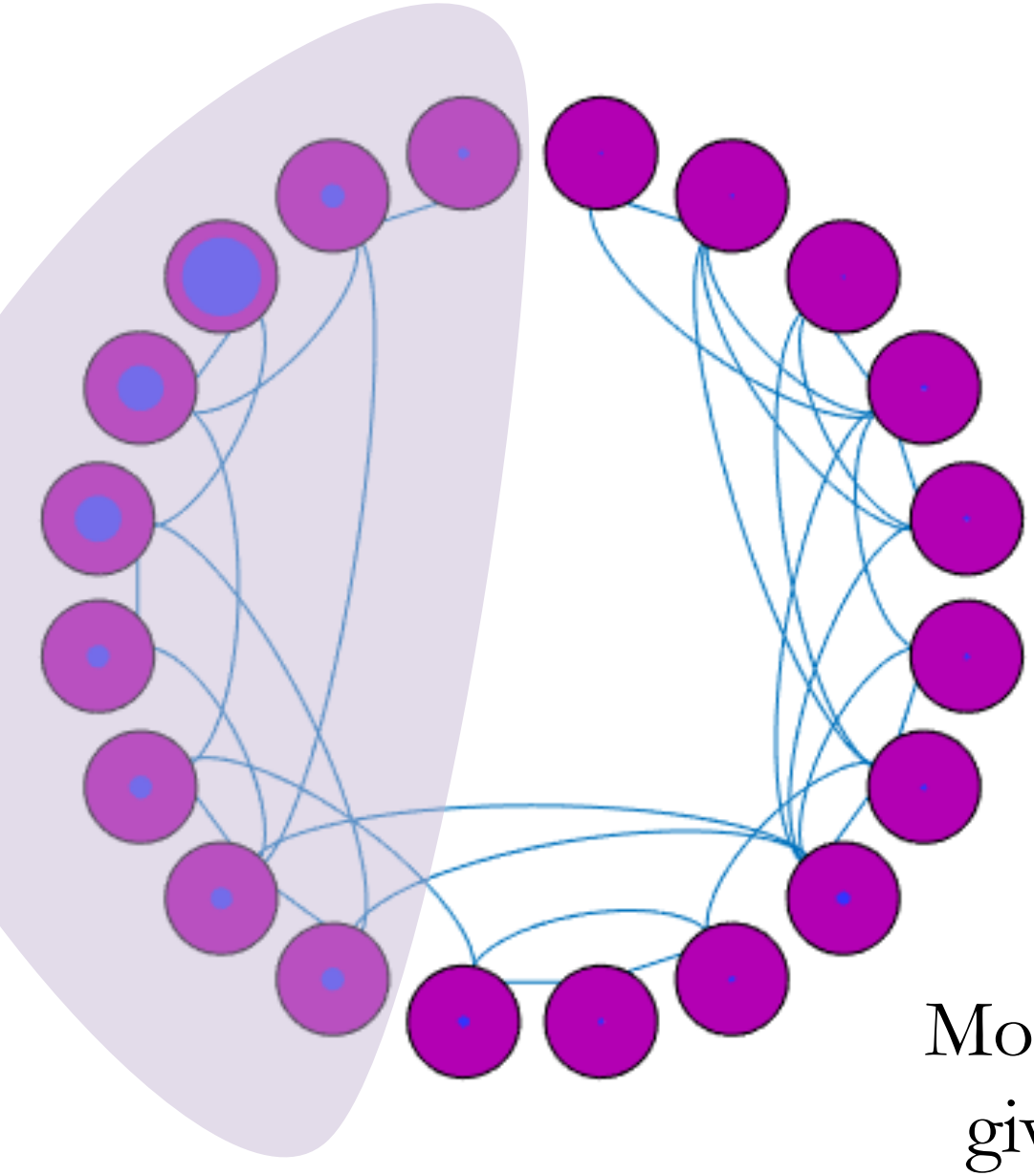
Importance: Personal PageRank



Spill paint at a node.
Paint both spreads and dries.

Amount of dried paint, measures importance relative to initial node

Importance: Personal PageRank



Spill paint at a node.
Paint both spreads and dries.

Amount of dried paint, measures importance relative to initial node

Most important nodes give clusters, if they exist.

PageRank, used by Google to rank web

Similar idea.

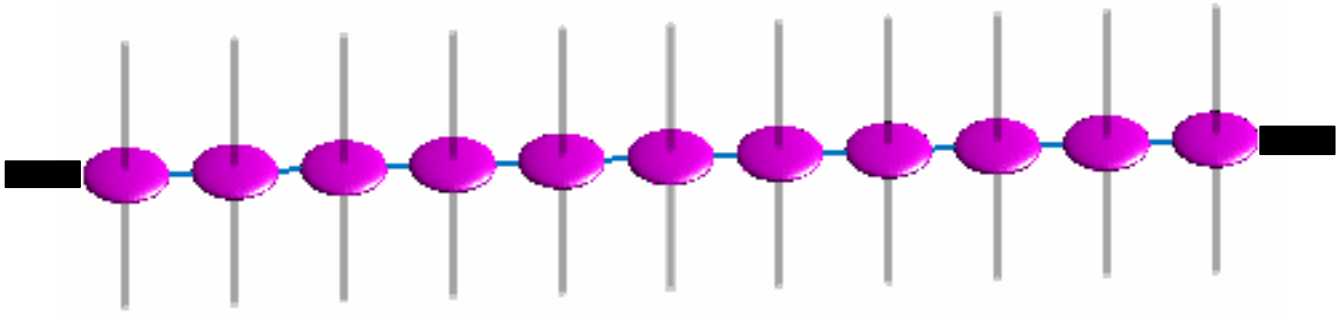
But,
paint only flows in the
directions of links.

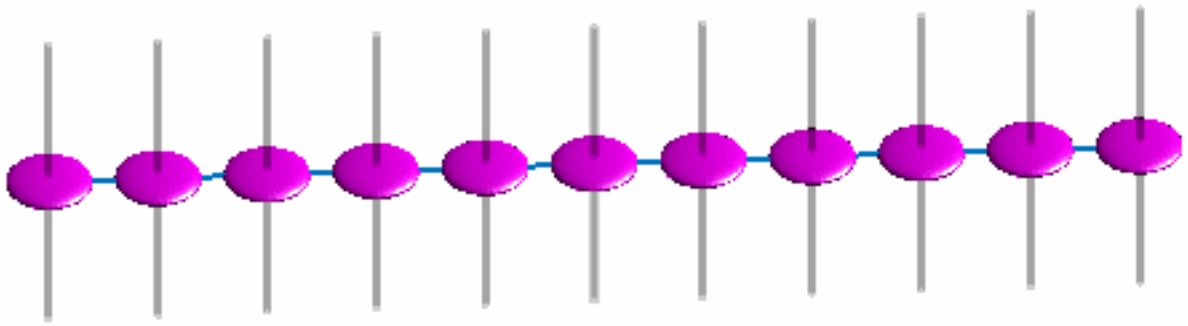
Vibrations / Eigenvectors

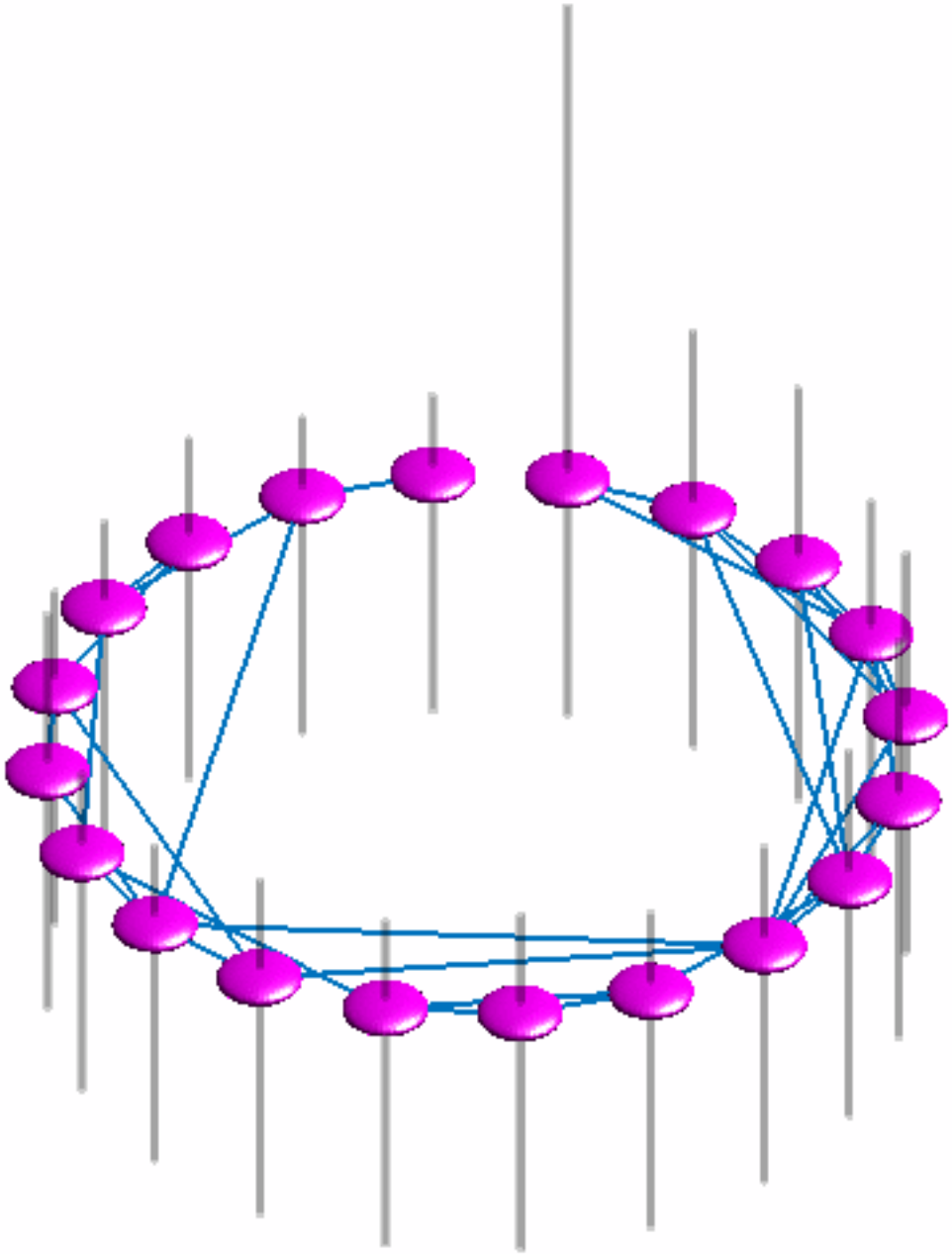
The springs never stop vibrating

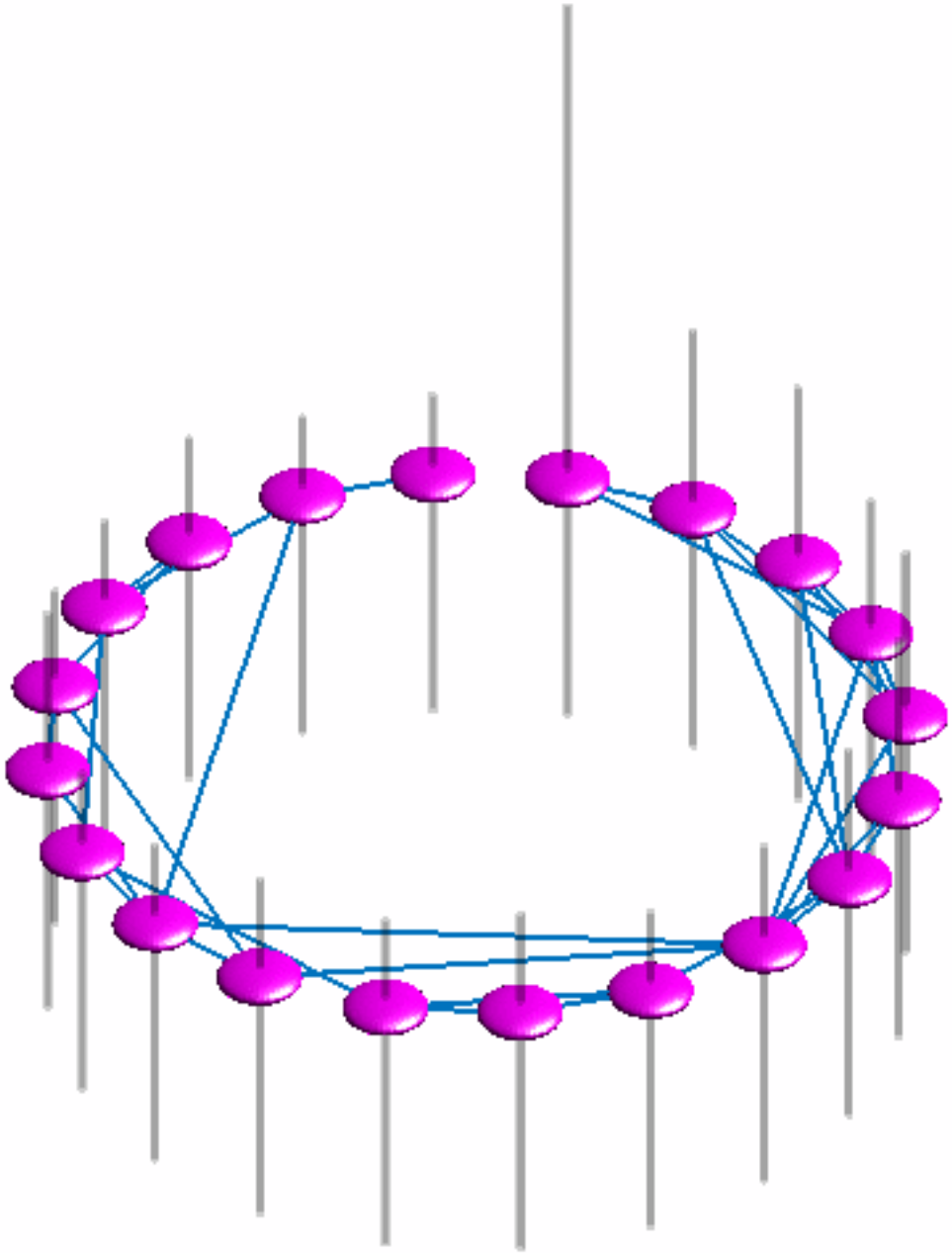
The “stuff” never stops moving

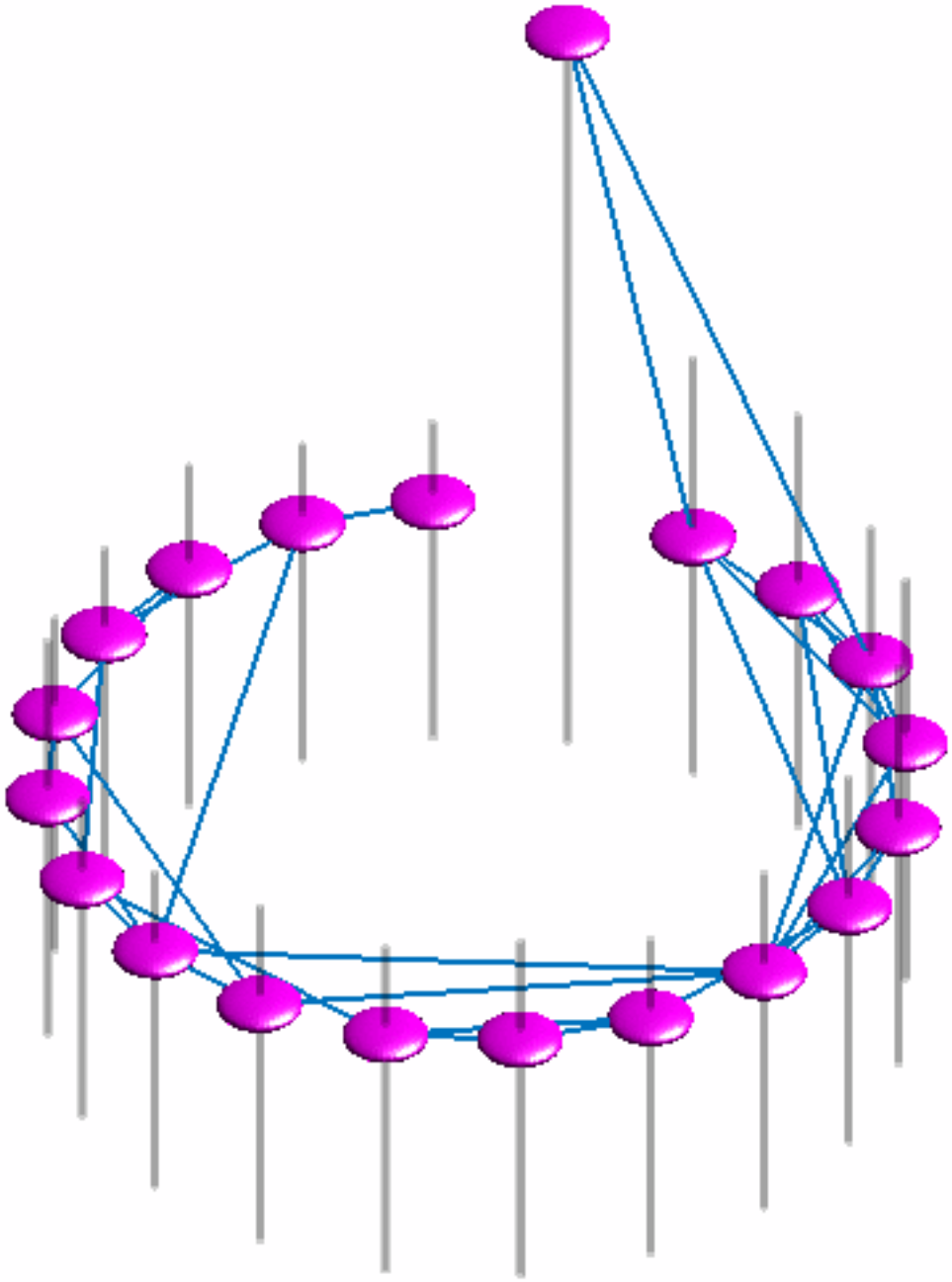
The motions are small

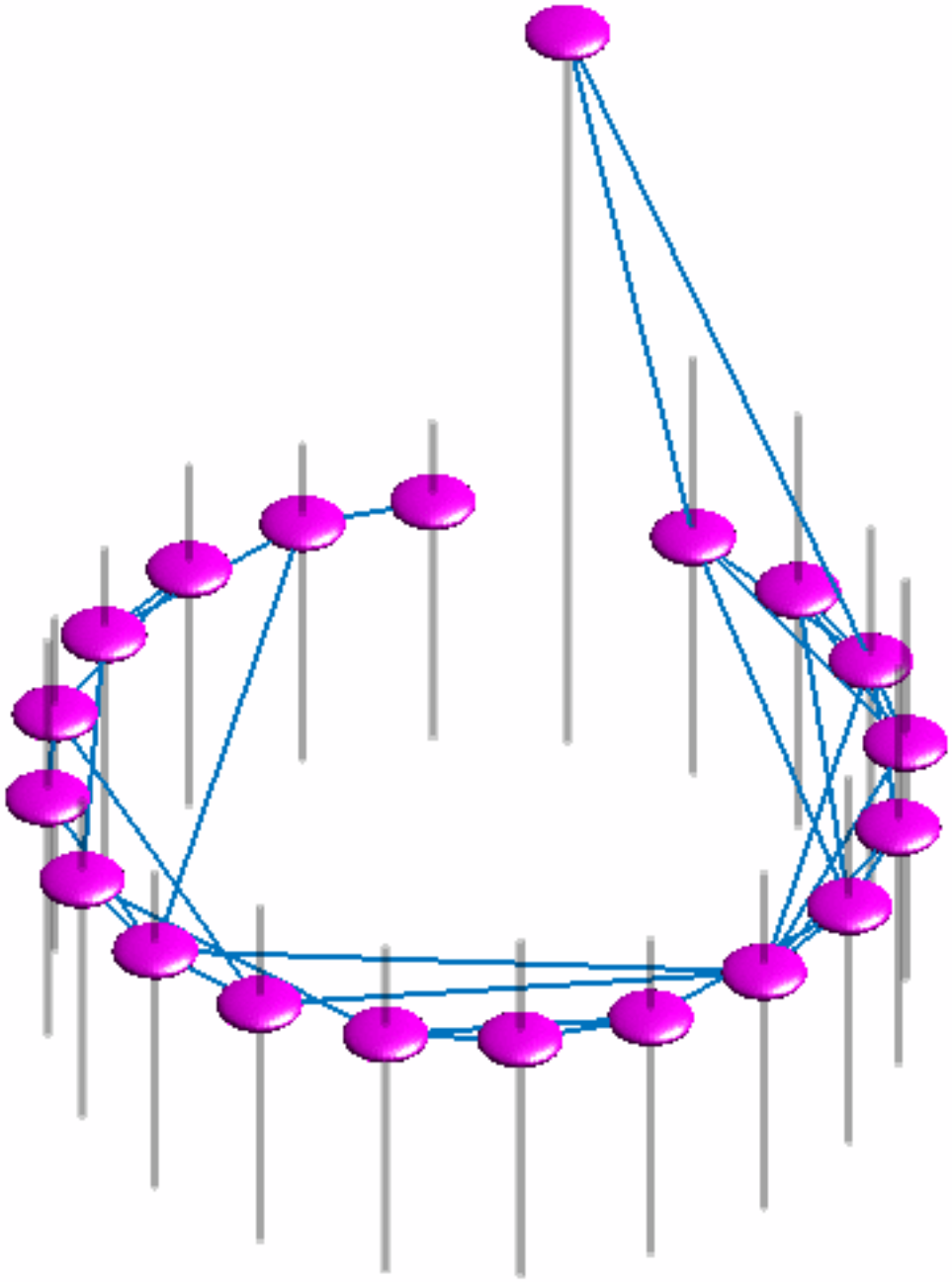




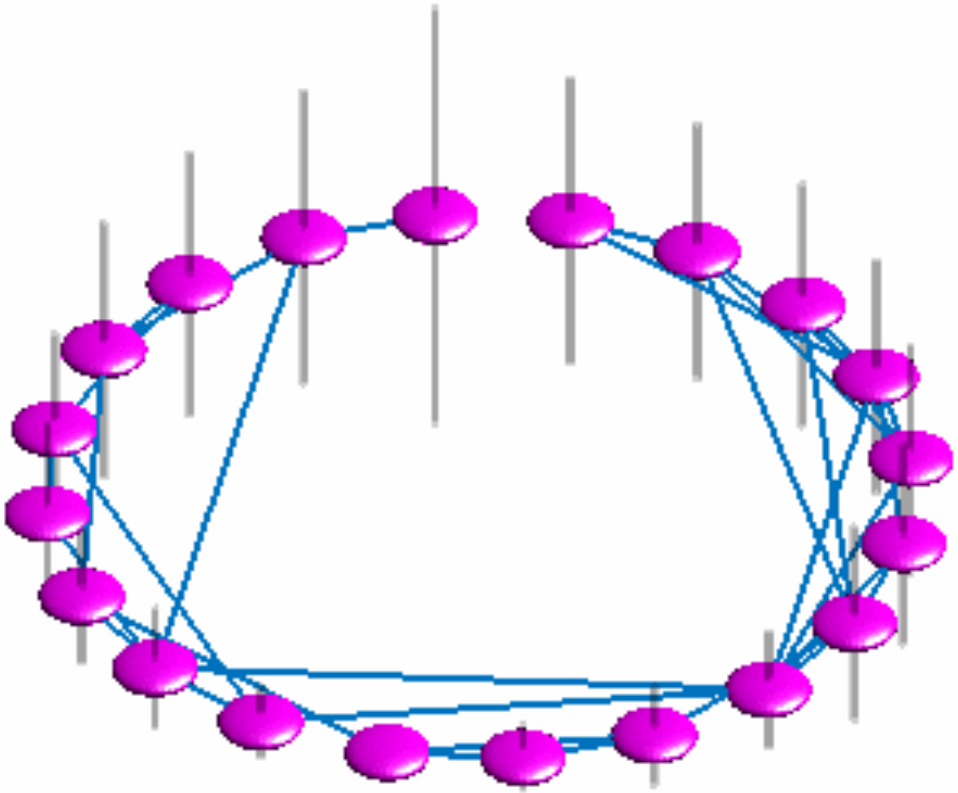






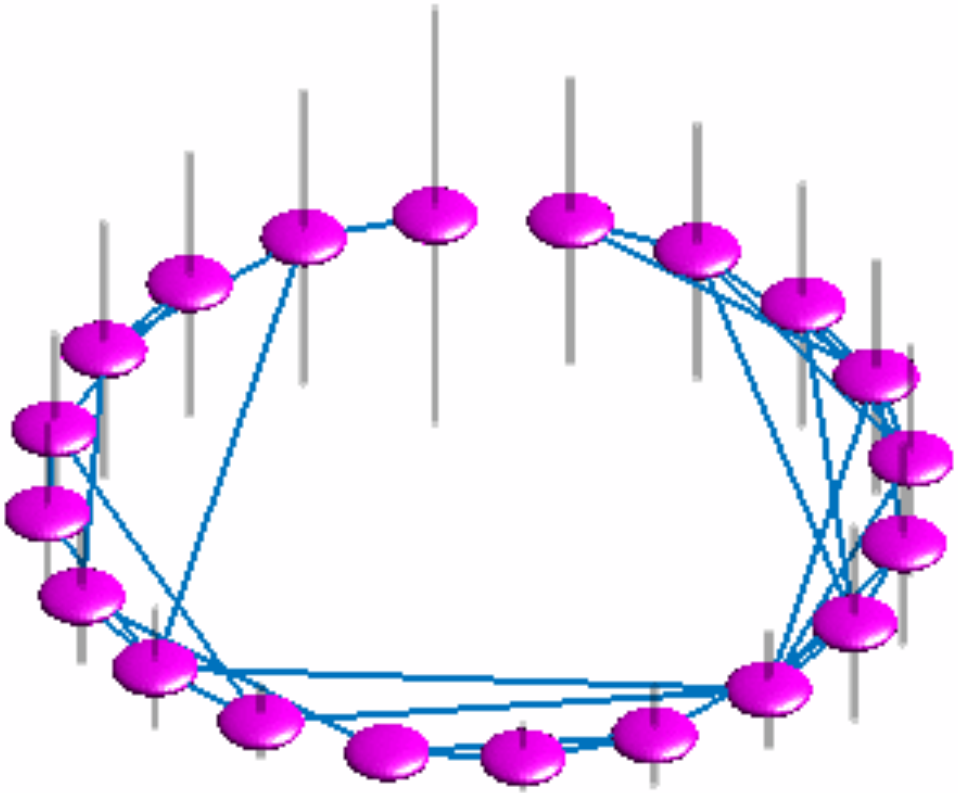


The fundamental mode

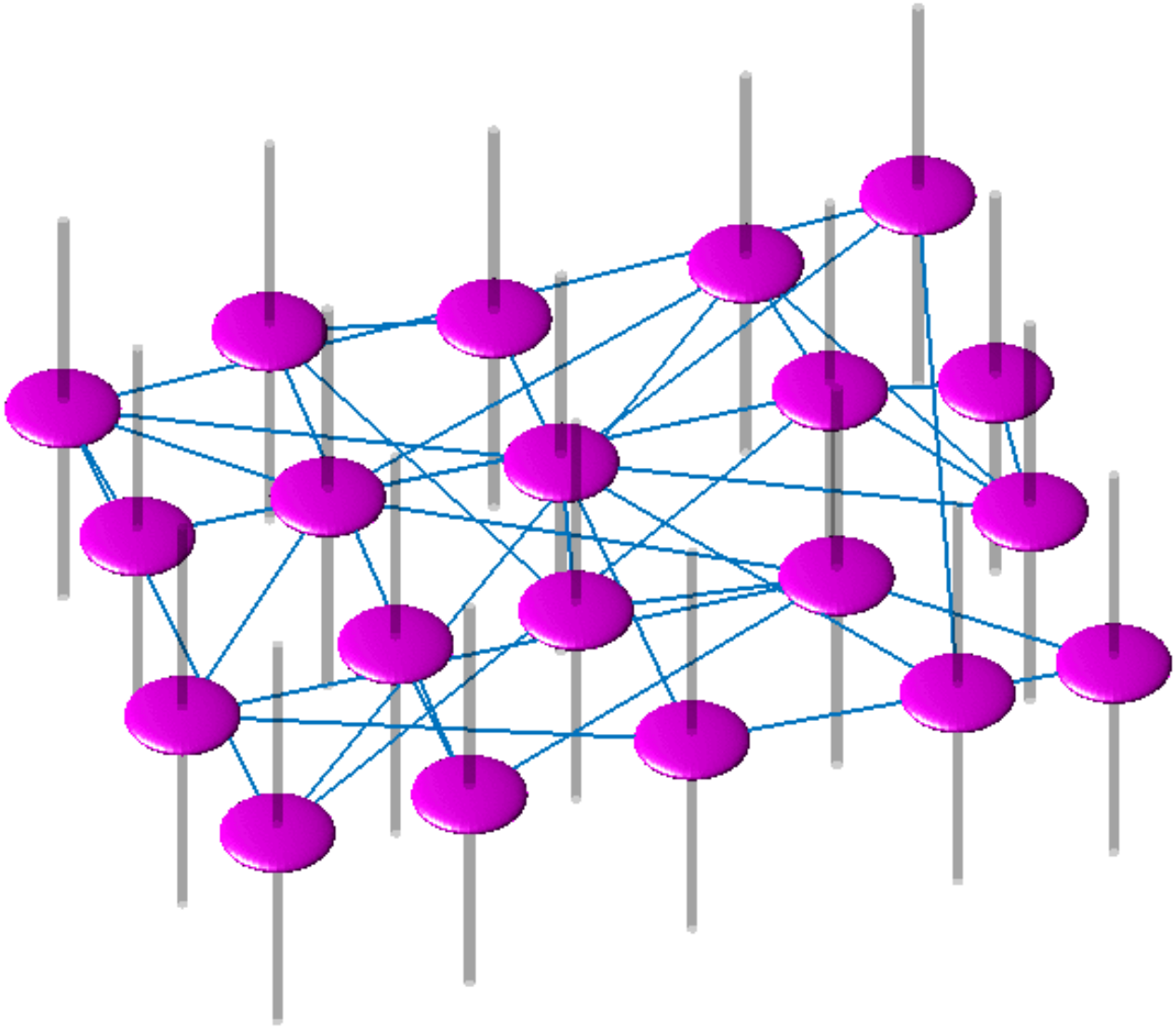


The fundamental mode

I used this to choose the order and draw the network

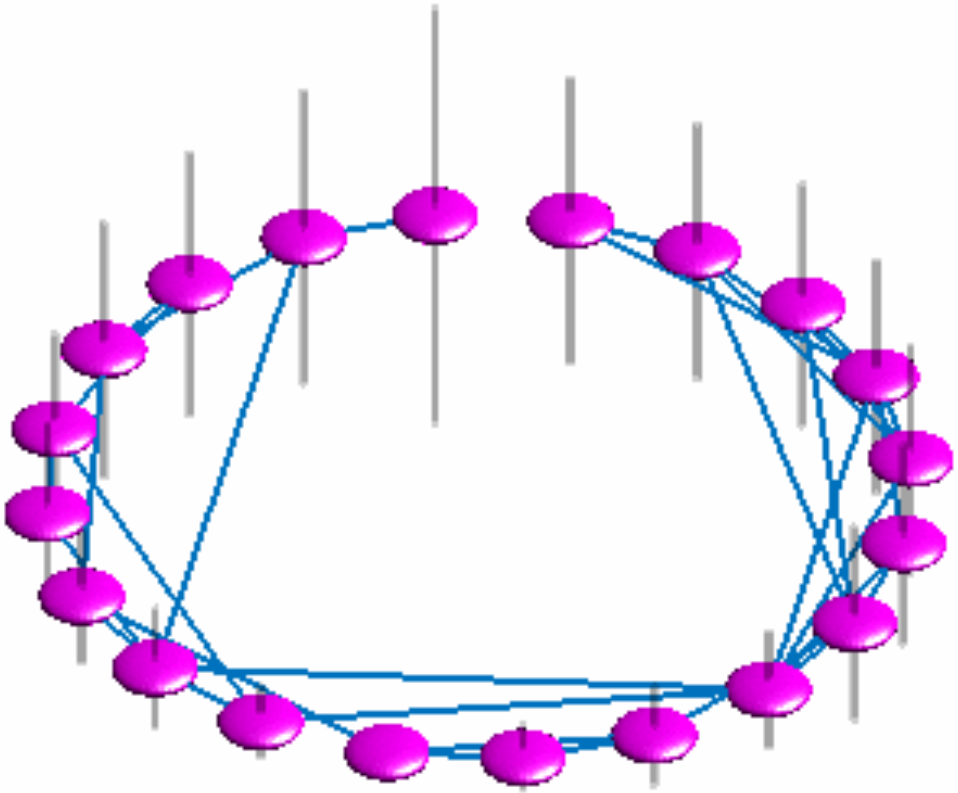


The fundamental mode



The fundamental mode

I used this to choose the order and draw the network



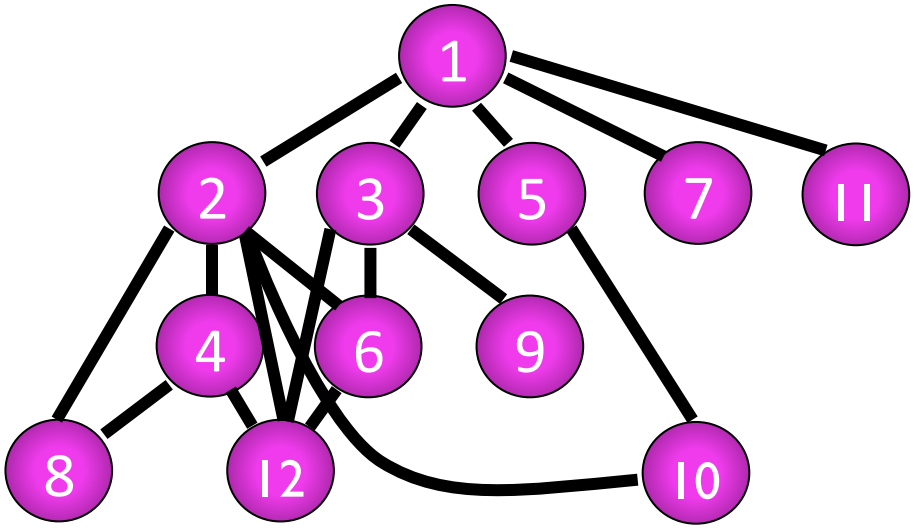
Algorithmic Problems: how to quickly compute

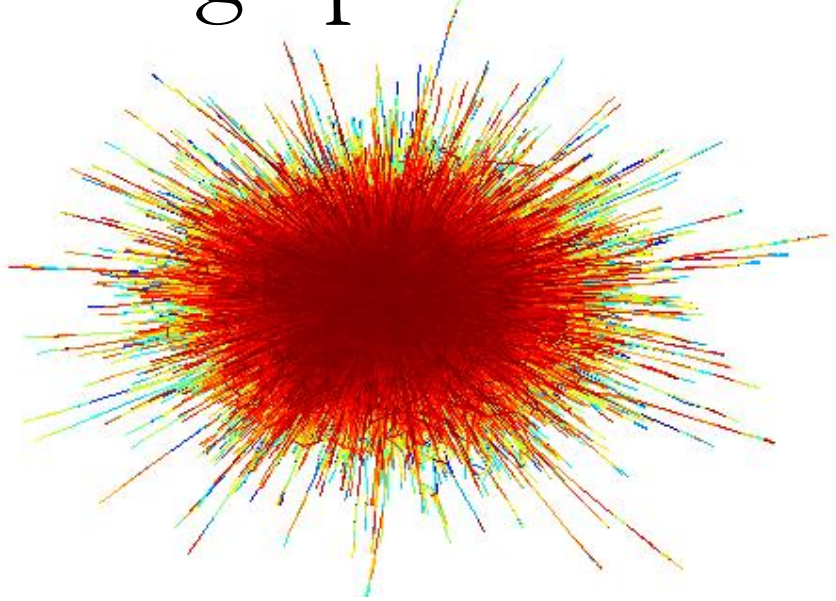
The stable configuration of springs =
solve linear equations

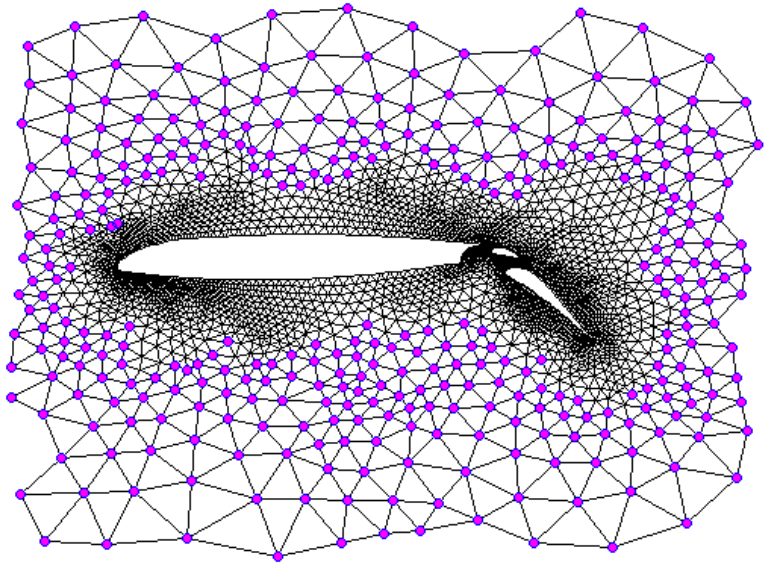
The vibrations / fundamental modes =
compute eigenvectors

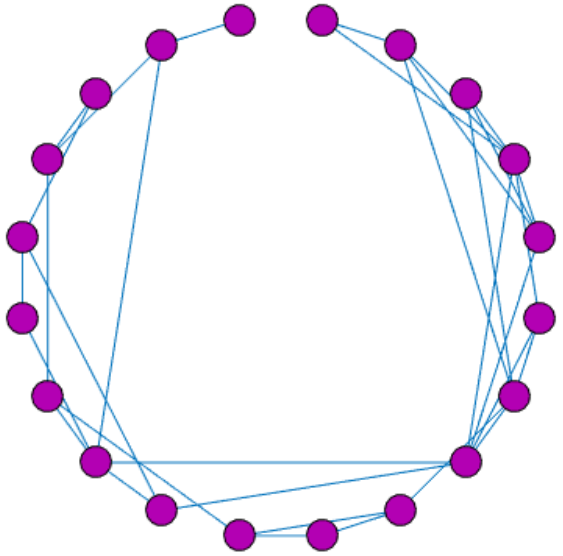
State of diffusion after a long time,
without just simulating and waiting

The difficult cases: Chimeric graphs

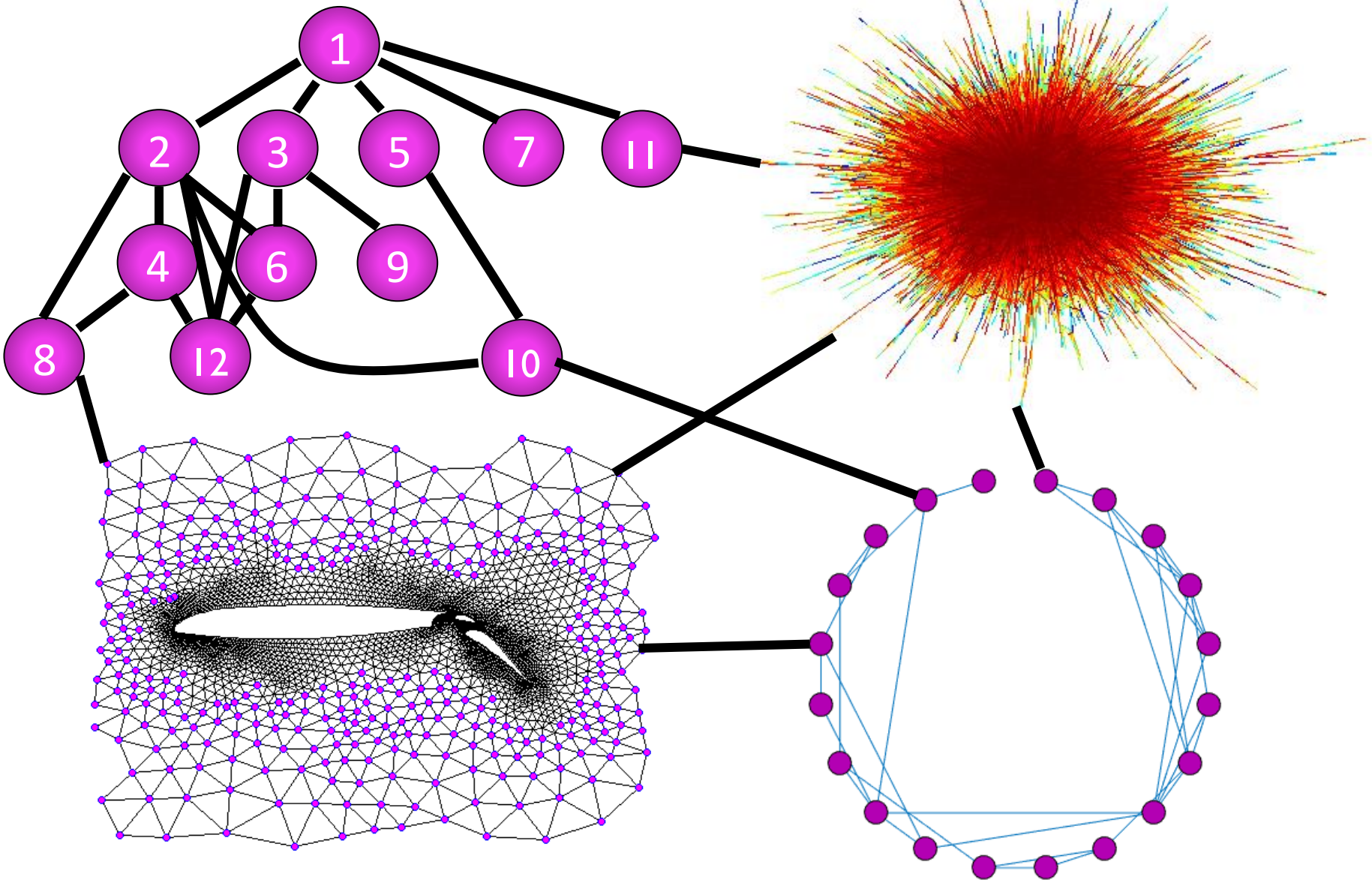








The difficult cases: Chimeric graphs



Finding the Stable Configuration

is a problem of solving linear equations

$$3x - y - z = 1$$

$$-x + 2y - z = 0$$

$$-2x - y + 4z = -1$$

Can solve exactly by Gaussian Elimination

Finding the Stable Configuration

is a problem of solving linear equations

$$3x - y - z = 1$$
$$-x + 2y - z = 0$$
$$-2x - y + 4z = -1$$

Can solve exactly by Gaussian Elimination

Can solve approximately by simulating physics

Finding the Stable Configuration

is a problem of solving linear equations

$$3x - y - z = 1$$
$$-x + 2y - z = 0$$
$$-2x - y + 4z = -1$$

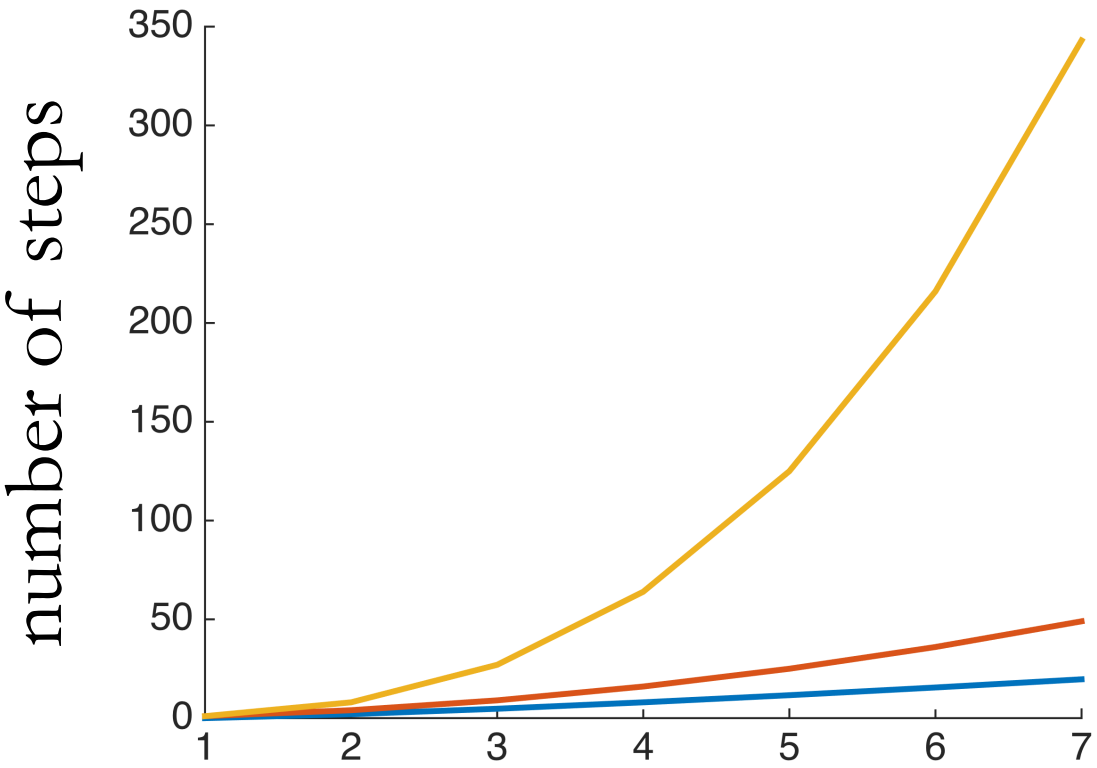
Can solve exactly by Gaussian Elimination

Can solve approximately by simulating physics

Or, by much fancier algorithms (preconditioning)

Running Times of Algorithms

Measure by n, the number of variables



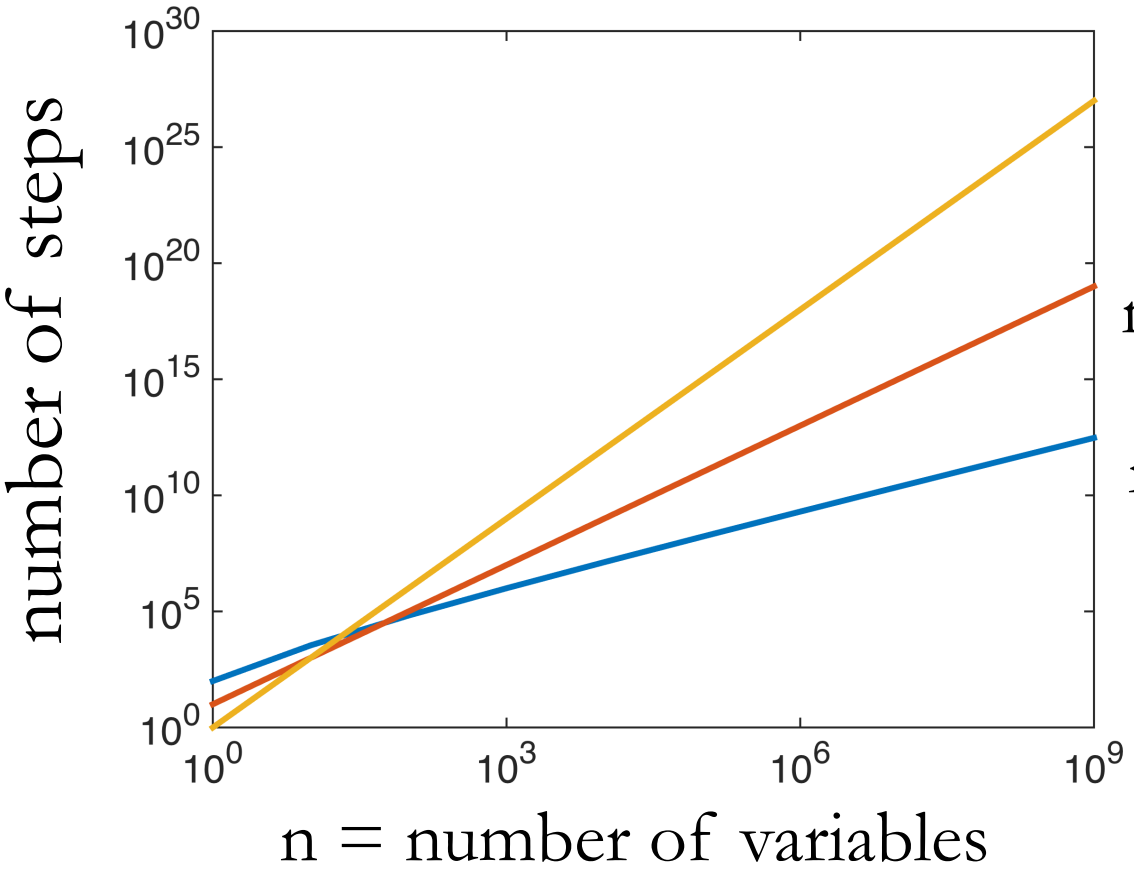
n^3 : elimination

n^2 : simulation

n log n: preconditioning

Running Times of Algorithms

Measure by n, the number of variables



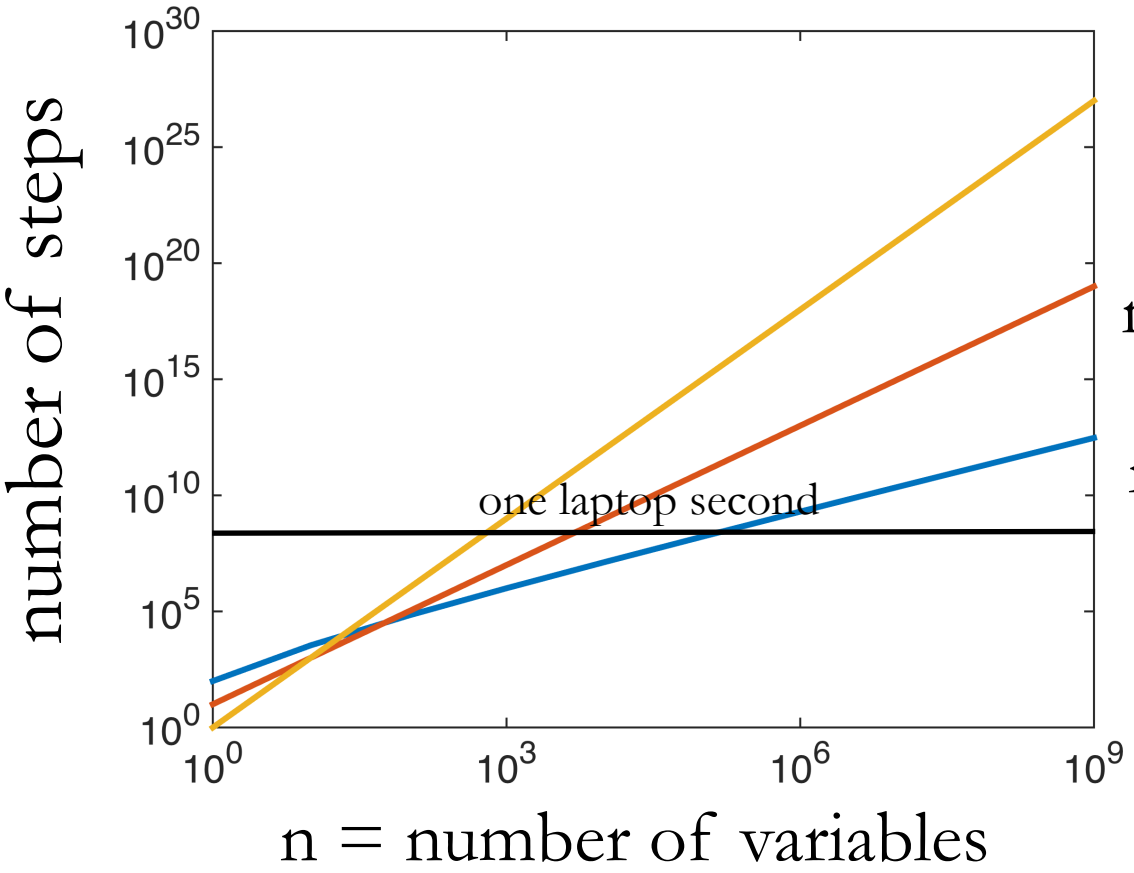
n^3 : elimination

n^2 : simulation

n log n: preconditioning

Running Times of Algorithms

Measure by n, the number of variables



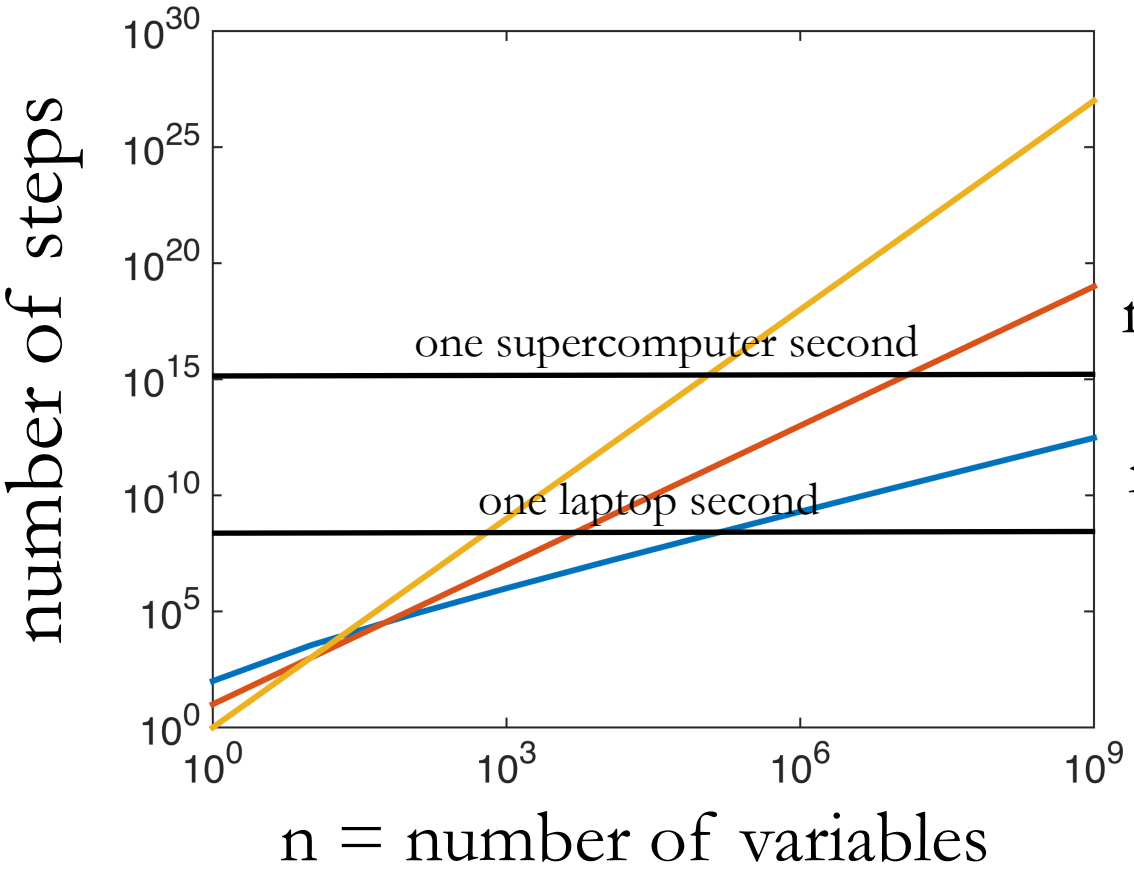
n^3 : elimination

n^2 : simulation

n log n: preconditioning

Running Times of Algorithms

Measure by n, the number of variables



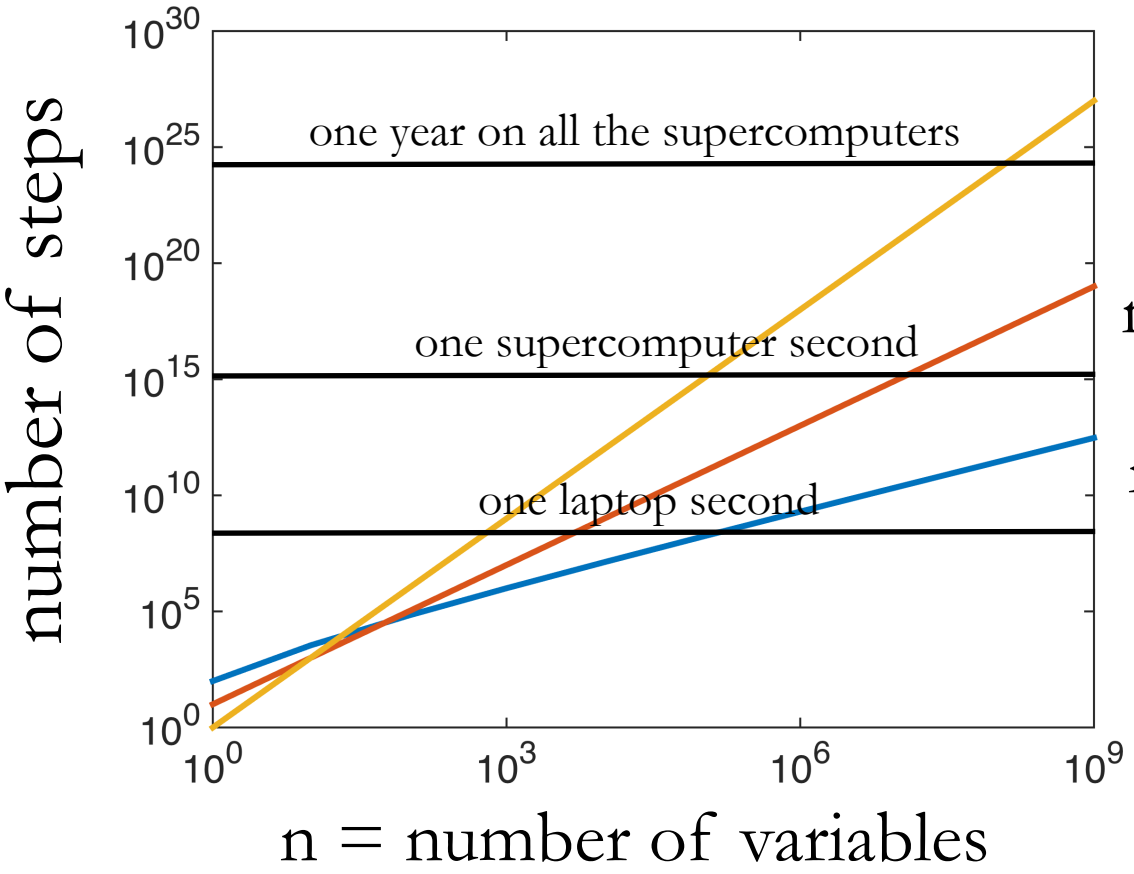
n^3 : elimination

n^2 : simulation

n log n: preconditioning

Running Times of Algorithms

Measure by n, the number of variables



n^3 : elimination

n^2 : simulation

n log n: preconditioning

To learn more (and all the caveats)

See

my web page on Laplacians, Clustering, etc.

or

lecture notes from "Spectral Graph Theory"
and "Graphs and Networks"