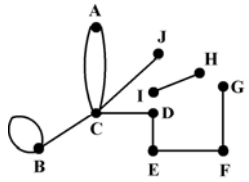
Graph concepts

Graphs are made up by vertices (nodes) and edges (links).
An edge connects two vertices, or a vertex with itself – loop.



AC, AC - multiple edges
BB – loop

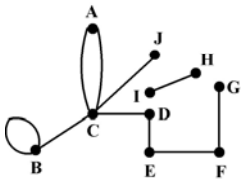
The shape of the graph does not matter, only the way the nodes are connected to each other.

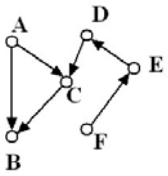
Simple graph - does not have loops (self-edges) and does not have multiple identical edges.

Further reading:
http://www.utm.edu/departments/math/graph/glossary.html

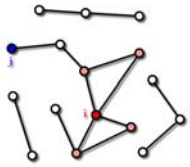
Symmetrical and directed graphs

Two distinct types of edges: symmetrical and directed (also called arcs).
Two different graph frameworks: graph, digraph = directed graph.
In the digraph framework a symmetrical edge means the superposition of two opposite directed edges.



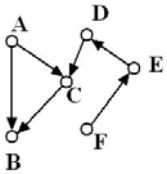


Node degrees



Node degree: the number of edges connected to the node. $k_i = 4$

In directed networks we can define an in- degree and out-degree. The (total) degree is the sum of in- and out-degree. $k_C^{in} = 2 \quad k_C^{out} = 1 \quad k_C = 3$

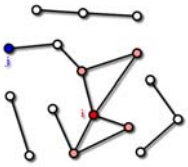


Source: a node with in-degree = 0.

Sink: a node with out-degree = 0.

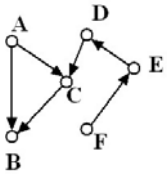
E.g. A, F are sources, B is a sink.

Average degree



$$\langle k \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i$$

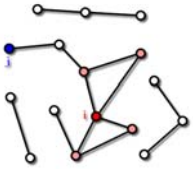
N – the number of nodes in the graph

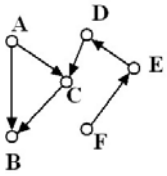


$$\langle k^{in} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{in}, \quad \langle k^{out} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{out}, \quad \langle k^{in} \rangle = \langle k^{out} \rangle$$

Q: What is the relation between the number of edges in a (non-directed) graph and the sum of node degrees? How about in a directed graph?

Statistics of node degrees





Average degree $\langle k \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2E}{N}$

$\langle k^{in} \rangle = \langle k^{out} \rangle = \frac{E}{N}$

The degree distribution $P(k)$ gives the fraction of nodes that have k edges.

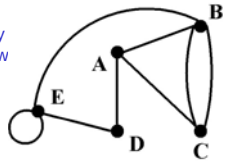
Similarly $P(k^{in})$ / $P(k^{out})$ gives the fraction of nodes that have in-degree k^{in} / out-degree k^{out}.

Ex. Calculate the degree distributions of the graphs in the left.

Paths and circuits

Adjacent nodes (vertices) – there is an edge joining them.

In the digraph framework the adjacency only defined in the direction of the arrow



Path: a sequence of nodes in which each node is adjacent to the next one. Edges can be part of a path only once.

In the digraph framework a symmetrical edge can be used once in one direction and once in the opposite direction.

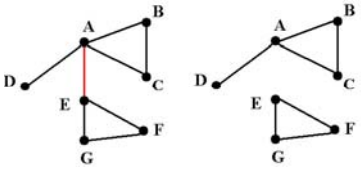
Circuit: a path that starts and ends at the same vertex .

Cycle: a circuit that does not revisit any nodes.

Ex. Give examples of circuits and cycles in the above graph

Connectivity of undirected graphs

Connected (undirected) graph: any two vertices can be joined by a path.
A disconnected graph is made up by two or more connected components.

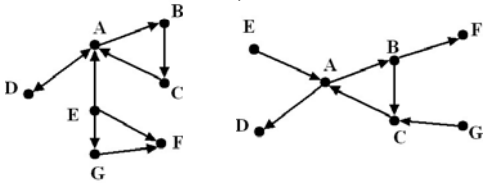


Bridge: if we erase it, the graph becomes disconnected.

Connectivity of directed graphs

Strongly connected directed graph: has a path from each node to every other node and vice versa (e.g. AB path and BA path).
Weakly connected directed graph: it is connected if we disregard the edge directions.

Strongly connected components can be identified, but not every node is part of a nontrivial strongly connected component.



In-component: nodes that can reach the scc,
out-component: nodes that can be reached from the scc.

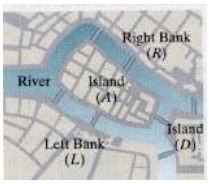
Exercises

1. Draw a graph or digraph with 4 nodes such that each node has degree 1 / 2 / 3. Try to use a variety of edges: symmetrical, directed, multiple edges, loops.

2. You have N nodes and need to build a connected graph from them. Each time you add an edge you must pay $1. What is the minimum amount of money needed to build the graph?

3. You are constructing a disconnected graph from N nodes. For each edge you add you receive $1. You are not allowed to use directed edges, loops or multiple edges, and you must stop before the graph becomes connected. What is the most money you can make?

Exercise: the bridges of Konigsberg



Walking problem –
traverse each bridge
do not recross any bridge
return to the starting point

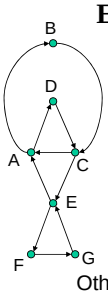
Euler circuit: return to the starting point by traveling each edge of the graph once and only once.

Is there a solution to the Konigsberg bridge walking problem?

Euler's theorem:
(a) If a graph has any vertices of odd degree, it cannot have an Euler circuit.
(b) If a graph is connected and every vertex has an even degree, it has at least one Euler circuit.

How would we need to modify the graph so it has an Euler circuit?

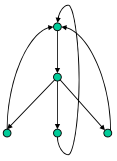
Euler circuits in directed graphs



If a digraph is strongly connected and the in-degree of each node is equal to its out-degree, then there is an Euler circuit

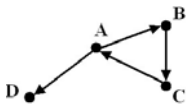
Q: Give one possible Euler circuit

Otherwise there is no Euler circuit.
This is because in a circuit we need to enter each node as many times as we leave it.



Distances between nodes

The distance between two nodes is defined as the number of edges along the shortest path connecting them.
If the two nodes are disconnected, the distance is infinity.
In digraphs each path needs to follow the direction of the arrows.
Thus in a digraph the distance from node A to B (on an AB path) is generally different from the distance from node B to A (on a BA path).



Ex. Calculate the distances among node pairs for the above graphs.

How to record distances

Tip: fill out the matrix

	A	B	C	D
A	--	l_{AB}	l_{AC}	l_{AD}
B	l_{BA}	--	l_{BC}	l_{BD}
C	l_{CA}	l_{CB}	--	l_{CD}
D	l_{DA}	l_{DB}	l_{DC}	--

Q: How many entries will you need for an N- node graph?

A: N(N-1) in a digraph, N(N-1)/2 in a symmetrical graph. Let's use the notation

$$N_{pairs} = \binom{N}{2} = \frac{N(N-1)}{2}$$

Diameter and average distance

Graph diameter: the maximum distance between any pair of nodes in the graph. Note: not the longest path.

Average path length/distance for a connected graph (component) or a strongly connected (component of a) digraph.

$$\langle l \rangle \equiv \frac{1}{2N_{pairs}} \sum_{i,j \neq i} l_{ij}$$

, where l_{ij} is the distance from node *i* to node *j*, $N_{pairs} = \binom{N}{2} = \frac{N(N-1)}{2}$ and N is the number of nodes in the graph or component.

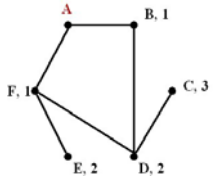
Since in a (symmetrical) graph $l_{ij} = l_{ji,}$ we only need to count them once $\langle l \rangle \equiv \frac{1}{N_{pairs}} \sum_{i,j>i} l_{ij}$

Algorithm for finding distances- breadth first search

Distance between node u and node v:

1. Start at u.
2. Find the nodes adjacent to u. Mark them as at distance 1. Put them in a queue.
3. Take the first node, w, out of the queue. Find the unmarked nodes adjacent to it in the graph. Mark them with the label of w +1. Put them in the queue.
4. Repeat until you find v or there are no more nodes in the queue.
5. The distance between u and v is the label of v or, if v does not have a label, infinity.

Ex. Apply the algorithm to find the distance between A and C



Other applications of breadth first search

Find the connected components of a graph:

1. Start from a node u, label with 1
2. Find all nodes reachable from u, label with 1
3. Choose an unmarked node v, label with 2
4. Find all nodes reachable from v, label with 2
5. Repeat with increasing labels until no more unmarked nodes

Calculate average distance of a connected graph:

1. Put the nodes in an ordered list
2. Use BFS to find distances between the first node and all other nodes, cumulate them
3. Use BFS to find distances between the second node and all other nodes except the first, cumulate them
4. Repeat as you go down the list
5. Divide cumulated distance by the number of node pairs

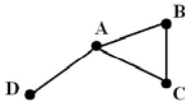
Note that BFS can only find the reachable nodes!

Graph efficiency

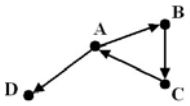
To avoid infinities in graphs that are not connected and digraphs that are not strongly connected, one can define a graph efficiency (= average inverse distance)

$$\eta \equiv \frac{1}{2N_{pairs}} \sum_{i,j \neq i} \frac{1}{l_{ij}}$$

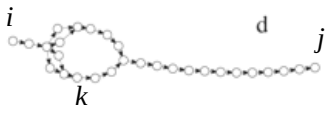
N_{pairs} is the number of node pairs



Ex.: Calculate the average distance and efficiency of the graphs on the right.



Betweenness centrality (load)



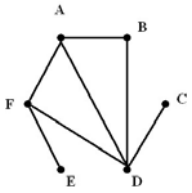
For all node pairs (i, j):

- Find all the shortest paths between nodes i and j - $C(i,j)$
- Determine how many of these pass through node k - $C_k(i,j)$

The betweenness centrality of node k is

$$g_k = \sum_{i \neq j} \frac{C_k(i,j)}{C(i,j)}$$

L. C. Freeman, Sociometry 40, 35 (1977)



$$g_k = \sum_{i \neq j} \frac{C_k(i,j)}{C(i,j)}$$

$C(i,j)$ –nr. of shortest paths btw. i,j
$C_k(i,j)$ – nr. of these paths that contain k

Ex1: Calculate the betweenness centrality of the nodes in this graph. Do not count being the starting or ending point of a path ($k \neq i$, $k \neq j$) .

Tip: Construct a node pair (half)matrix and fill it with the nodes between each node pair.

Ex.2. Determine the betweenness centrality distribution for the graph.

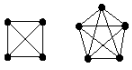
Common subgraphs

Subgraph: a subset of nodes of the original graph and of edges connecting them. Does not have to contain all the edges of a node included in the subgraph.

Trees: contain no circuits; N nodes and N-1 edges.

Cycles: circuits where nodes are not revisited; N nodes and N edges

Cliques: completely connected subgraphs; N nodes and N(N-1)/2 edges



Note difference between connected and completely connected!

Special directed subgraphs

Bi-fan

Feed-forward loop: two nonintersecting directed paths between a start and endpoint

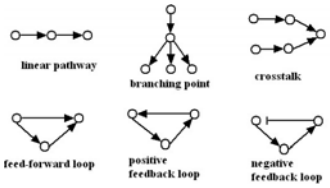
Bi-parallel: two nonintersecting paths of identical length between a start and endpoint

Feed-back loop: a directed cycle

S. S. Shen-Orr, R. Milo, S. Mangan and U. Alon, Nature Genetics (2002)

Network topology and dynamics

Network motifs can illustrate regulatory relationships



Further, dynamic details are needed to describe how multiple inputs on a node are integrated - additive action (e.g. same product for two chemical reactions)
- synergy (e.g. transcriptional regulation)

Weighted networks

In some applications it is necessary quantify edges with weights, corresponding, e.g., to a traversal cost or a geographical distance.

Then the shortest path between two nodes is redefined in terms of weight, e.g. “the path with lowest cost”, or “most efficient path”.

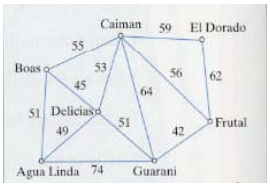
To find the distance between two nodes in a weighted network, calculate the sum of edge weights on each path (this is the path weight), then select the path with lowest weight

$$l_{ij} = w_{ik} + w_{kl} + \ldots + w_{mj}$$

where *i k l..m j* is the path with minimum weight
w_{ij} is the weight of edge *ij*

Kruskal’s algorithm to find the minimum spanning tree of a graph

MST: a tree that contains all nodes of the graph and has minimal edge weight



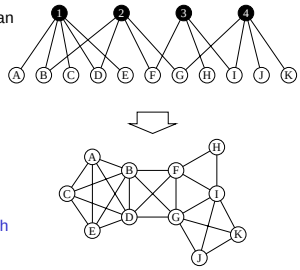
- Find the cheapest edge in the graph and mark it.
- Continue selecting the cheapest remaining edge at each step, but do not select edges that create circuits.
- When the number of edges is one less than the number of vertices, STOP.

Bipartite graphs

Group structure (e.g. in social networks) can be incorporated into a bipartite graph.

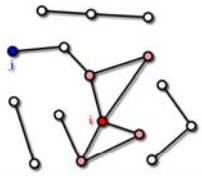
A bipartite graph has two types of nodes:
group nodes (black, numbered)
member nodes (white, lettered)
Edges are possible only between different types of nodes: membership in group.

An alternative representation connects all members in a given group - each group becomes a completely connected subgraph



D. Watts, P. S. Dodds, M. E. J. Newman, Science 296 (2002)
M. E. J. Newman, S. Strogatz, D. Watts, Phys. Rev. E 64, 026118 (2001)

Local order and clustering



Cliques (completely connected subgraphs)

$$k = N-1,\ n = \frac{N(N-1)}{2}$$

How close the neighborhood of a node is to a clique?

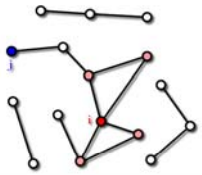
Edges among first neighbors of node *i*

Clustering coefficient $C_i \equiv \frac{n_i}{k_i(k_i-1)/2}, \quad k_i \neq 0,1$ or

$$C_i \equiv \frac{\textit{nr. of triangles connected to } i}{\textit{nr. of triples centered on } i}$$

Q: How would you generalize the concept of clustering coefficient to directed graphs?

Cumulating clustering coefficients



$$C_i \equiv \frac{n_i}{k_i(k_i-1)/2}, \quad k_i \neq 0,1$$

$$C_i \equiv \frac{\textit{nr. of triangles connected to } i}{\textit{nr. of triples centered on } i}$$

Average clustering coefficient $C \equiv \frac{1}{N}\sum_{i=1}^{N} C_i$

Clustering –degree function $C(k)$: for each degree represented in the graph calculate the average clustering coefficient of the nodes with that degree.

Ex. Determine the average clustering coefficient, clustering distribution and clustering-degree function of the above graph.

Ex. 1
N nodes are connected by N edges such that they form a cycle. This is also called a ring lattice.
How does the maximum distance between nodes (the diameter) depend on N? How about the average distance?

Ex. 2
On the ring lattice from above every second neighbor is connected by an edge. What is the clustering coefficient of the nodes?

Ex. 3
Construct a square lattice (grid) L edges long. How does the maximum distance between nodes depend on L ?

Regular lattices, ex. 1, 1D lattice (ring)

$k = 4$ for each node

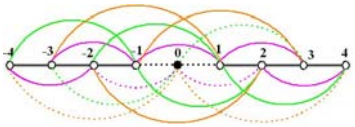
$C = \frac{1}{2}$ for each node if N > 6

$$1 + \sum_{l=1}^{l_{max}} 4 \approx N \Rightarrow l_{max} \approx \frac{N}{4} \qquad < l > = \frac{4\sum_{l=1}^{l_{max}} l}{N} \Rightarrow < l > \approx \frac{N}{8}$$

The average path-length varies as $< l > \approx N$

Constant degree, constant clustering coefficient.

Clustering coefficient of 1D lattice



N>12, the origin (black) node is connected to 4 nodes on each side.

Edges among neighbors: $n = 6 + 5 + 4 + 3 = 18$

first neighbors on lattice

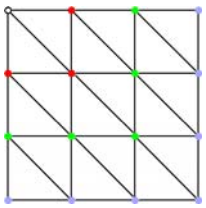
second neighbors

$$C = \frac{9}{14}$$

In general, $C = \frac{3}{2}\frac{k-1}{2k-1}$, where 2k is the degree of each node,

and N>3k, k>1

Regular lattices, ex. 2, 2D lattice



$k = 6$ for inside nodes

$C = \frac{6}{15}$ for inside nodes

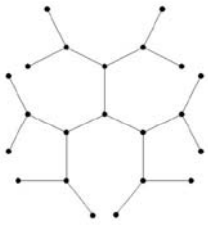
$$1 + \sum_{l=1}^{l_{max}} 6l \approx N \Rightarrow l_{max} \propto N^{0.5}$$

$$< l > \approx L \approx N^{1/2}$$

In general, the average distance varies as $< l > \approx N^{1/D}$

where D is the dimensionality of the lattice. Constant degree (coordination number), constant clustering coefficient.

Regular lattices, ex. 3, the Cayley tree



$k = 3$ *for inside nodes*

$k = 1$ *for surface nodes*

$< k >= 2$

$C = 0$

$$1 + 3\sum_{l=1}^{l_{max}} 2^{l-1} \approx N \Rightarrow l_{max} \propto \frac{\log N}{\log < k >}$$

$$< l > \approx \frac{\log N}{\log < k >}$$

Distances vary logarithmically with N. Constant degree, no clustering.