Economics of Information Networks

Stephen Turnbull

Division of Policy and Planning Sciences

Lecture 3b: December 3, 2020

Abstract

We look at some basic structures of graphs, and applications to social networks, including Granovetter's famous argument about *The Strength of Weak Ties* and approximate network partitioning.

In Part 2, we review the basic terminology for structures in graphs, and then consider some of the basic quantitative measures of graph structure.

Basic notions of networks

- Network structure is described using graphs.
  - Networks often have additional properties attached to them.
- Graphs consist of
  - *nodes*, also called *objects*, which may be interpreted in many ways, *e.g.*, people or places, and
  - *links*, usually called *arrows* in directed graphs, which represent relationships of various kinds. Links are also often called *edges*.
  - Additional properties of networks may be attached to either nodes or links.

Basic types of graphs

- Graphs may be restricted to zero or one link between each node (the usual case, and there is no special term for this case), or may have zero or more links. To distinguish from the usual case, this is called a *multigraph*.
- Links may be automatically symmetric, which is call an *undirected graph*, or they may be in principle asymmetric, called a *directed graph*.
  - In a directed graph, even if not a multigraph, there may be two links between a pair of nodes, going in opposite directions.
- A graph with numerical property attached to each link is called a *weighted graph*, and one with a qualitative property attached to each link is called a *colored graph*.
  - There is no particular name for graphs with properties attached to nodes. Graph structure is defined by the links, so attaching properties to nodes is not very useful in graph theory.

Basic structures in graphs

- A *path* in a graph is a sequence of links, each of which shares an endpoint with its next link.
- In a directed graph, *path* usually means a *directed path*, in which each pair of links is connected head to tail. But in discussions of "connectedness," *path* means an *undirected path.*
- When A and B are the endpoints of a path p, we say "A and B are *connected* by p." If A is the tail and B is the head of a directed path p, we say "B is *reachable* from A via p."
- The *length* of a path is the number of links in the path.
- A *cycle* is a path where each node can be considered to be both the beginning and the end of the path. A cycle of length 1 is called a *loop*.
- A *simple path* is a path that contains no cycles.

Connectedness

- Two nodes are *connected* if there is an undirected path from one to the other.
- A set of nodes is *connected* if for each node in the set there is a path to every other node in the set.
- A set of nodes is a *connected component* if (1) the set is connected, and (2) no node in the set is connected to a node not in the set. More concisely, it is a *maximal connected set*.
  - "Connected component" is often shortened to just "component." There is no other kind.
- A *connected graph* consists of a single connected component. A *disconnected graph* contains more than one component.
- In social and economic networks, we often observe the phenomenon of a single *giant component*, even if the graph is technically disconnected because of some tiny components or solitary nodes.

Betweenness

- A node A is *pivotal* for B and C if it lies on every shortest path between B and C.
- A node A is a *gatekeeper* for B and C if it lies on every path between B and C.
- A node A is a *local gatekeeper* for neighbors B and C if there is no link between B and C.

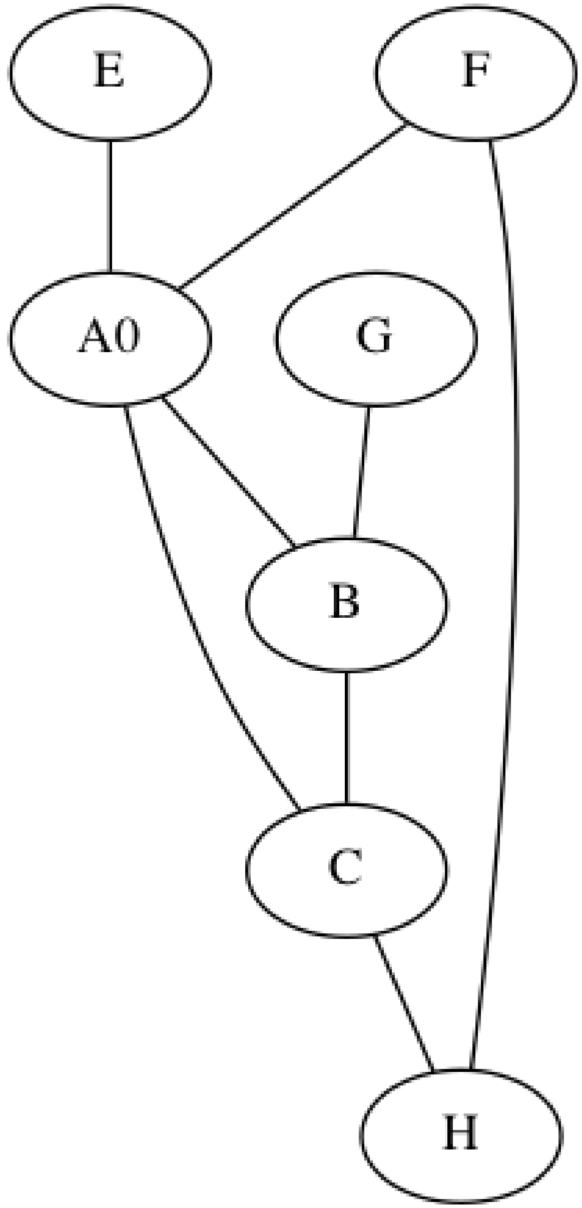
Bridges

- A *bridge* is a link if deleting it would cause its component to become disconnected.
- A *local bridge* is a link such that deleting it increases the distance between its endpoints to more than two, or is a bridge.
- The *span* of a local bridge is the length of the shortest alternative route between its endpoints.

Distance

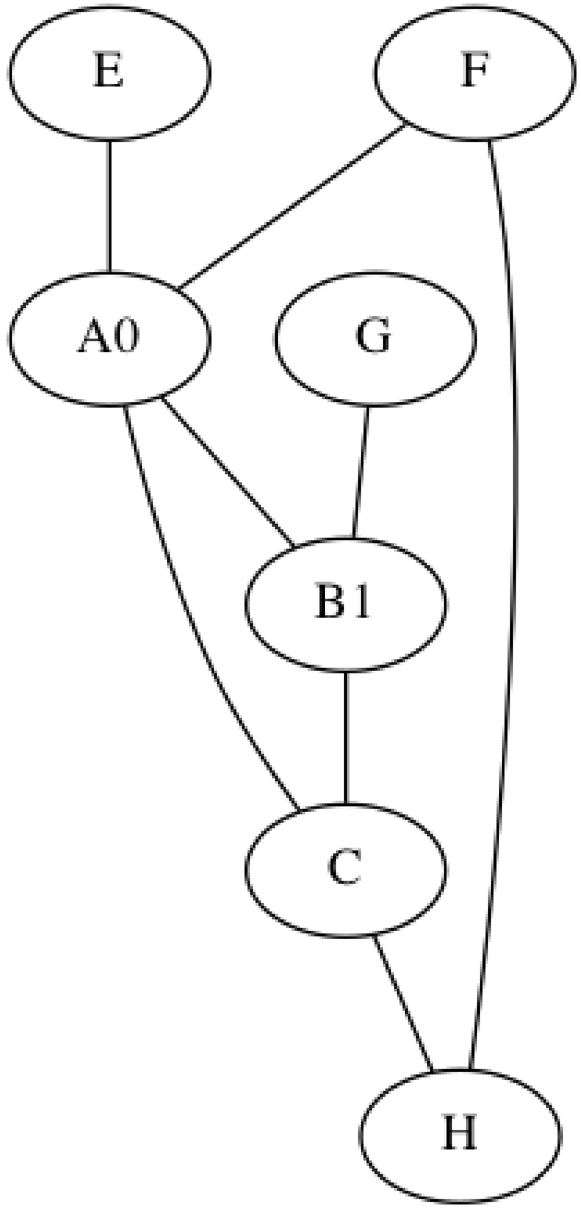
- The *distance* between two nodes is the length of the shortest path connecting them. (This path is necessarily a simple path, and need not be unique.)
- By convention, the distance from a node to itself is defined by the *empty path* which starts and ends at the node and contains no links. The distance therefore is 0. (Note: there is a different empty path for each node.)
- The *diameter* of a component is the maximal distance taken over all pairs of nodes in the component.
  - The diameter of a disconnected graph doesn't exist: the distance between nodes in different components is undefined. (There is no "shortest path" if there is no path at all.)
  - The diameter of connected sets of nodes that aren't components isn't useful. It's unclear what to do with paths that go outside the component.

Breadth-First Search: Start

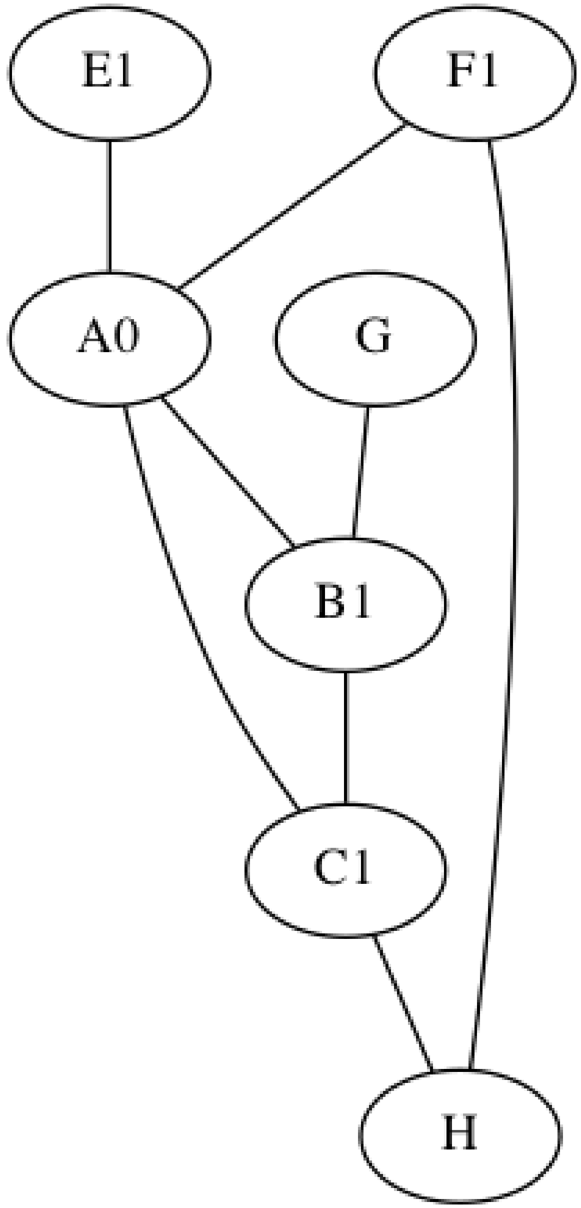


Breadth-first search is an efficient algorithm to (1) determine the distance from one node to another, and (2) determine all distances between all pairs of nodes.

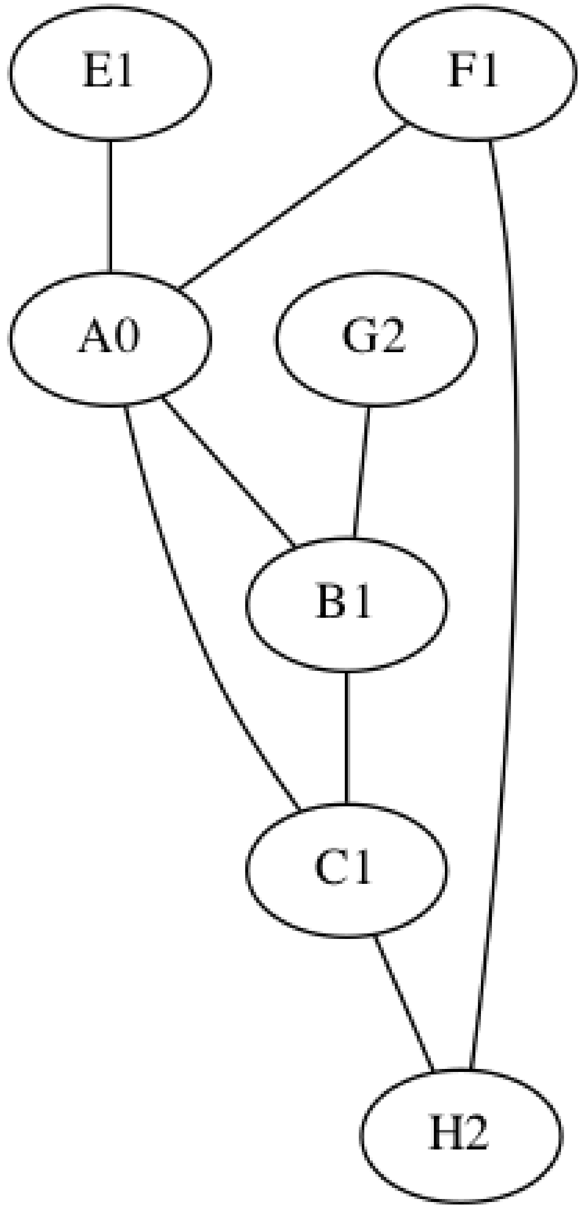
Breadth-First Search: First Neighbor



Breadth-First Search: All Neighbors



Breadth-First Search: All Distance=2 Nodes



At distance = 2, all nodes have been found. This implies that if a target node exists, it has been found.

Quantitive Analysis of Graphs

- To do economics, we need to compare values. Generally we want to make *marginal* comparisons, trading a little of one value for some of another, until we achieve balance. So we need to have quantitive measures of values. What can we measure in graphs?
- We start with *undirected* graphs with at most one link between objects.
- We have considered the *star graph* with a special central node that can be used to model a market, and the *complete graph* that underlies Metcalfe's quantitative model of "network externalities."
  - In these special graphs, we *count* the *number of links* or the *number of nodes* to determine value. In economics we often transform these values by a *cost* or *utility* function.
- Real information networks have many properties that depend on the exact configuration of links.

Connected Graphs

- A set of nodes in a graph is *connected* if for all nodes x and y in the set there is an undirected path from x to y.
  - A graph is *connected* if the set of all nodes is connected.
    * Note: in a directed graph, the nodes need not be linked head to tail. The graph may be connected even though there are pairs of nodes with no path between them.
    * Example: every path, considered as a subgraph, is connected.
  - A set of nodes is *strongly connected* if it (1) is connected and (2) has no link which if removed the graph is no longer connected.
    * Example: Every cycle is strongly connected. Remove any link, but its nodes are still connected "the long way around!"

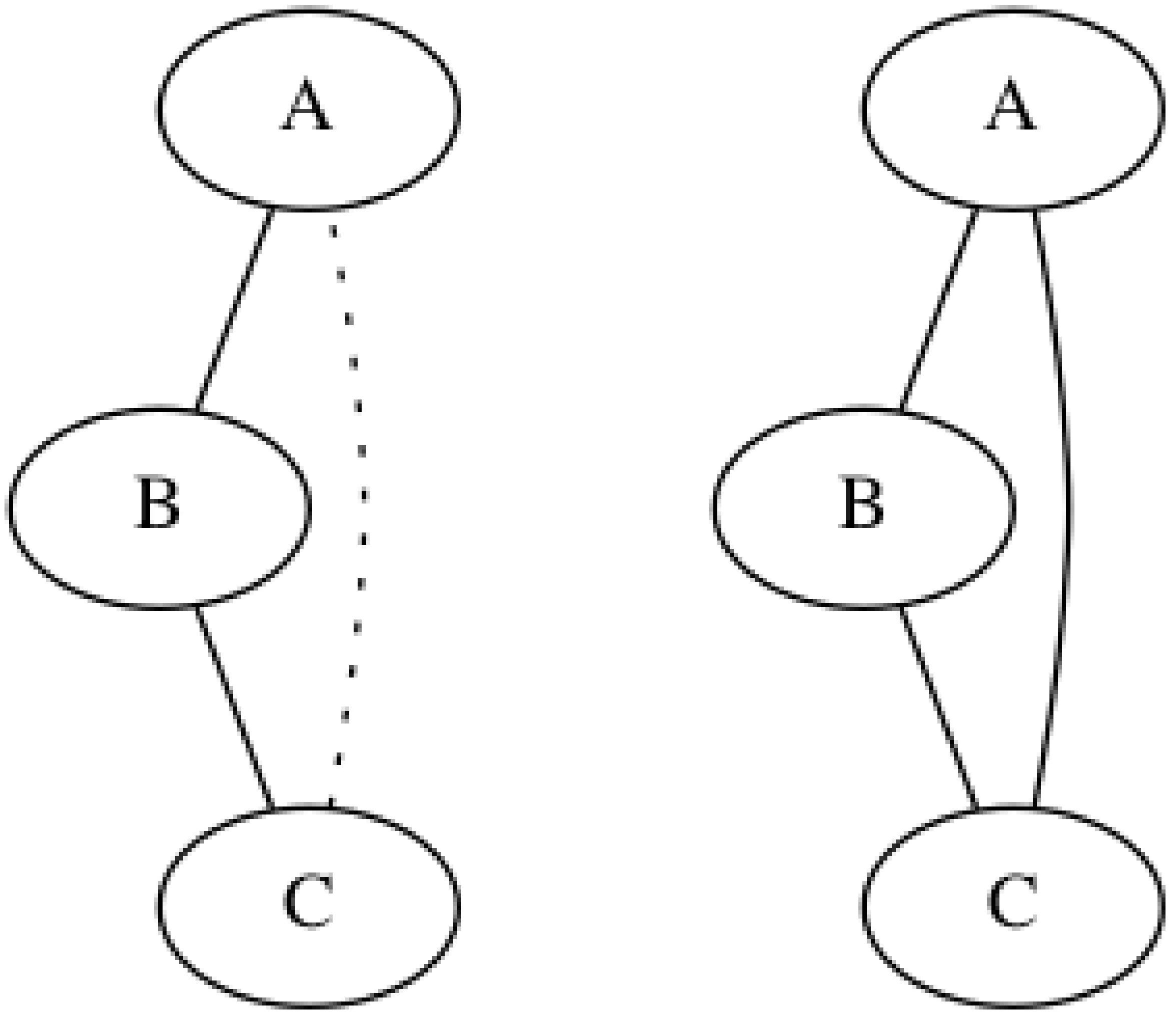
Triadic Closure

- We can start considering "completeness" in a non-trivial way when we have at least three nodes.
  - We typically ignore loops in information networks—you don't need to tell yourself what you already know.
  - With only two nodes, you can *define* completeness, but it's not very interesting. If the graph is incomplete, it has no connections!
- In fact, it's useful to look at all the connected subsets of exactly three nodes in a graph, which gives the idea of triadic closure.

Triadic closure: definition

- Consider three nodes in a graph. If A is connected to B and B is connected to C, is ABC is the shortest path between A and C?
- In social situations, there is the phenomenon of *introduction*: when A knows B and B knows C, it is common that B introduces A to C, creating a new link from A to C.
- *Triadic closure* refers to both the process of creating a triangle (*e.g.*, via introduction), and to a state where three nodes and their links form a triangle.
- The *clustering coefficient* of a node is the fraction of pairs of neighbors of the node which are neighbors of each other. It measures the amount of triadic closure around the node.
- We can also consider the fraction of all triads that are closed in the whole graph (or in any connected component), but this is not so interesting.

Triadic closure: whole graph



Triadic closure: subgraph