Table of Contents

- 1 The Structure of the World Wide Web
    - 1.1 Information Networks
    - 1.2 Hypertext
    - 1.3 Reachability in Directed Graphs
        - 1.3.1 Weakly Connected V Strongly Connected
- 2 Graphs as relations
    - 2.1 Digraphs and Mathematical Relations
    - 2.2 The Bow-Tie Structure of the WWW
- 3 Computing Bow-Tie Components
    - 3.1 Weakly Connected Components

CS4423-Networks: Week 11 (26+27 March 2025)

Part 2: Relations and Bow Ties

Niall Madden, School of Mathematical and Statistical Sciences
University of Galway

This Jupyter notebook, and PDF and HTML versions, can be found at https://www.niallmadden.ie/2425-CS4423/#Week11

This notebook was adapted by Niall Madden from one developed by Angela Carnevale.

In [1]:

```
import networkx as nx
import numpy as np
opts = { "with_labels": True,  "node_color": "#84003d", "font_color": "white", "arrows

import matplotlib.pyplot as plt

np.set_printoptions(precision=2)    # just display arrays to 2 decimal places
np.set_printoptions(suppress=True)
```

The Structure of the World Wide Web

So far, most of the networks that have been discussed most of the time consisted of people or organizations, connected by links representing opportunities for interactions.

For a different type of network, we consider the World Wide Web, or, simply "the web", which is an example of an **information network**.

Note: the "web" and the "internet" are not the same thing: the Internet is the infrastructure which supports the web (and lots of other things too).

Information Networks

Information networks connect pieces of information, like documents, or parts of documents, through links that represent references of some kind. Such links, in contrast to social relationships which are typically symmetric, only point in one direction. The underlying graph of an information network thus is a **directed graph**.

Information networks have existed before the WWW. Some prominent examples include:

- **Academic Publications.** In the scientific literature it is customary to give credit to sources that have been used in the form of references to those publications that contain those sources. This practice creates a network whose nodes are the publications, and whose links represent the references, pointing from the citing article back to the cited article. A large part of this network in the mathematical literature is captured on MathSciNet.

- **Technical Documentation.** The documentation of complex systems, such as computer software, typically consists of a collection of articles (manual pages), each describing one aspect of the system, frequently using cross-references to each other. Here the network consists of the manual pages, and the links represent those cross references. In a similar way, an encyclopedia (or a dictionary) organizes its content as a sequence of articles, sorted alphabetically, with supporting cross-references.

Hypertext

The **World Wide Web** arose out of the desire to make technical documentation more easily accessible by using the physical infrastructure of the rapidly growing internet. It was conceived by Tim Berners-Lee around 1990 as information management system at CERN.

In this system, documents are **web pages**, that anyone can create and store in a publicly accessible place on their computer. Moreover, it supplies a **web browser**, a piece of software that can retrieve the web pages from those public spaces, allowing others to easily access those documents.

Web pages contain **hypertext**, that is a mixture of plain text and **hyperlinks**. Here, a hyperlink (or just link) is a reference to another document that the reader can follow by simply clicking on it. Hyperlinks have a **source** (the document they are contained in) and a **target** (the document they reference). This creates a network of documents as nodes and hyperlinks as **directed edges** between them.

It is not an especially new idea. The term **hypertext** was coined by Ted Nelson in 1965/1966.

It will be useful to distinguish between **navigational links** (providing access to related pages) and **transactional links** (which exist more as a side effect--like ordering a book, or sending an email--than for the sake of leading to the next page). The distinction is not always clear, but transactional links are the kind that is of little interest for search engines. It's the navigational links that form the edges of the directed graph that turns the Web into an information network.

Reachability in Directed Graphs

As with undirected graphs, an interesting question in directed graphs is: which nodes can be reached from a given node?

A **directed graph** is a pair $G = (X, E)$ with **vertex set** X and **edge set** $E \subseteq X^2 = X \times X$. For an edge $(x, y) \in E$ we sometimes write $x \to y$.

Note that here an edge is an **ordered pair** as opposed to an edge in an undirected graph being a 2-element set.

A **path** in a directed graph $G = (X, E)$ is a sequence of nodes $(x_0, x_1, \ldots, x_l)$ with $x_{i-1} \to x_i$ for $i = 1, \ldots, l$. The number l is called the **length** of the path. We write $x \rightsquigarrow y$ if there exists a path (possibly of length 0) from x to y in G.

Weakly Connected V Strongly Connected

Definition. A directed graph G is **weakly connected** if, when considerd as undirected graph, it is connected. The **weakly connnected components** (WCCs) of G are its connected components, when considered as undirected graph.

Definition. A directed graph G is **strongly connected** if, for each pair of vertices $x, y \in X$, there is a path from x to y in G, i.e., if $x \rightsquigarrow y$.

A **strongly connected component (SCC)** of a directed graph G is a subset C of X which is

(i) strongly connected, and

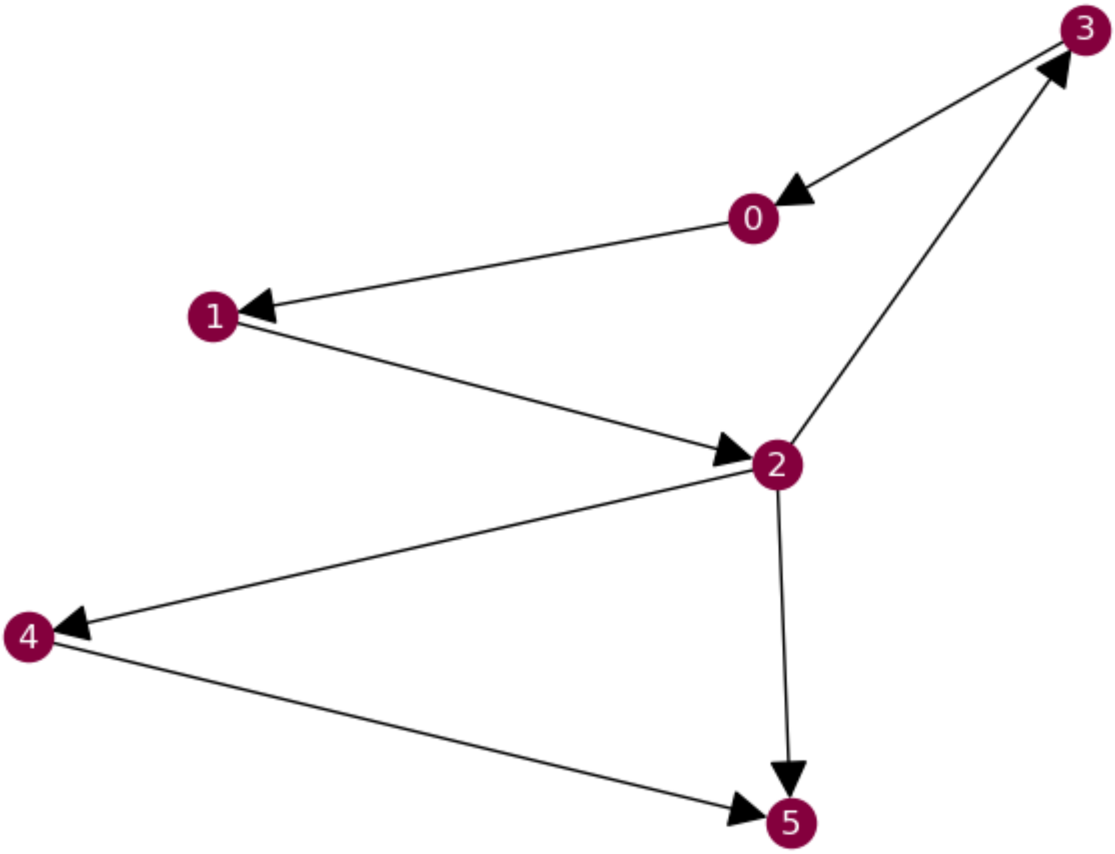
(ii) not part of a larger strongly connected subset of X.

In general, a directed graph is a collection of WCCs. Each WCC in turn is a collection of SCCs.

Example Identify the WCCs and SCCs in this graph:

In [2]:

```
G = nx.DiGraph([(0, 1), (1, 2), (2, 3), (3,0), (2,4),(2,5),(4,5)])
nx.draw(G, **opts)
```



In the previous example: Node 5 is an example of a **sink** - it has only incoming edges.

Graphs as relations

Definition. A **relation** from a set X to a set Y is (nothing but) a subset R of the Cartesian product $X \times Y = \{(x, y) : x \in X,\ y \in Y\}$. We say that $x \in X$ is **R-related** to $y \in Y$ whenever $(x, y) \in R$ and then write xRy.

- R is a **homogeneous** relation if $Y = X$ (usually is for us: X is the node set)
- (R) R is **reflexive** if xRx for all $x \in X$;
- (S) R is **symmetric** if xRy implies yRx for all $x, y \in X$;
- (T) R is **transitive** if xRy and yRz imply that xRz for all $x, y, z \in X$;
- (I) R is **irreflexive** if **not** xRx for all $x \in X$;
- (A) R is **antisymmetric** if xRy and yRx imply that $x = y$ for all $x, y \in X$.

Example 1

Given a (simple, **undirected**) graph $G = (X, E)$, we can define a relation R_1 on X, as being xR_1y if $(x, y) \in E$. Then R is **symmetric** and **irreflexive**.

Example 2

Given a (simple, **undirected**) graph $G = (X, E)$, let R on X be the relation: xR_2y if there is a path from x to y in G. What are the properties of this relation?

Does R_2 contain R_1?

Transitive Closure

Given a relation, R, its **transitive closure**, R^+ is the smallest transitive relation that includes R.

Reflexive Closure

Given a relation, R, its **reflexive closure**, $R^=$ is the smallest reflexive relation that includes R.

Examples:

Digraphs and Mathematical Relations

When a directed graph G is regarded as a **relation** on the set X, strongly connected components can be described as the **equivalence classes** of an equivalence relation that is obtained as follows.

First note that the relation $x \rightsquigarrow y$ is the reflexive and transitive closure of the edge relation $x \to y$.

Thus, by construction it is reflexive and transitive.

(There might be nodes x and y with $x \rightsquigarrow y$ and $y \rightsquigarrow x$, though it won't be all of them).

So this allows us to define a new relation as follows.

Definition. We set $x \equiv y$ if $x \rightsquigarrow y \textit{ and } y \rightsquigarrow x$.

This **is** an equivalence relation we get equivalence classes that partition our graph. These equivalence classes are the **strongly connected components** of G. We denote the class of $x \in X$ by $[x]$.

Moreover, there is a partial order relation $\leq$ (a relation which is reflexive, transitive and anti-symmetric) on the set of equivalence classes:

$[x] \leq [y]$ if $x \rightsquigarrow y$.

We can say something about the structure of the WWW in terms of these equivalence classes and of the partial order on them.

The Bow-Tie Structure of the WWW

Like the giant component in a simple graph, it turns out that a directed graph with sufficiently many edges has a **giant SCC**.

The remainder of the graph consists of four more sets of components of nodes, as follows:

1. IN: upstream components, the set of all components C with $C <$ SCC.
2. OUT: downstream components, the set of all components C with $C >$ SCC.
3. tendrils: the set of all components C with either $C >$ IN and $C \not<$ OUT or $C <$ OUT and $C \not>$ IN; and tubes: components C with $C >$ IN, $C <$ OUT but $C \not<$ SCC.
4. disconnected components.

Thus, in any directed graph with a distinguished SCC, the WCC in which it is contained necessarily has the following global bow-tie structure:

Research on available data on the Web in 1999 has confirmed this bow tie structure for the World Wide Web, with a **large Giant SCC** covering less than $\frac{1}{3}$ of the vertex set, and the three parts IN, OUT and the tendrils and tubes roughly of the same size. One can assume that this proportion has not changed much over time, although the advent of social media has brought many new types of links and documents to the Web.

Computing Bow-Tie Components

Example. Let's start with a reasonably large random **directed graph**, using the Erdős-Rényi $G(n, m)$ model:

In [3]:

```
n, m = 100, 120
G = nx.gnm_random_graph(n, m, directed=True)
```

Weakly Connected Components

The weakly connected components of a directed graph G can be determined by BFS, as before, counting as "neighbors" of a node x **both** its *successors* and it *predecessors* in the graph.

A single component, the weakly connected component of node x, is found as follows.

In [4]:

```
def weak_component(G, x):
    nodes = {x}
    queue = [x]
    for y in queue:
        G.nodes[y]["seen"] = True
        for z in set(G.successors(y)) | set(G.predecessors(y)): ## preds+succs are the
            if z not in nodes:
                nodes.add(z)
                queue.append(z)
    return nodes
```

The list of all weakly connected components is computed by looping over all the nodes of `G`, computing the components of "unseen" nodes and collecting them in a list. The final result is sorted by decreasing length before it is returned.

In [5]:
```
def weak_components(G):
    wccs = []     # initialize

    # find each node's wcc
    for x in G:
        if not G.nodes[x].get("seen"):
            wccs.append(weak_component(G, x))

    # clean up after yourself
    for x in G:
        del G.nodes[x]["seen"]

    # return sorted list of wccs
    return sorted(wccs, key=len, reverse=True)
```

In [6]:
```
wccs = weak_components(G)
len(wccs)
```

Out[6]:
```
13
```

In [7]:
```
[len(c) for c in wccs]
```

Out[7]:
```
[88, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```