Characteristing graphs

Julien Arino
University of Manitoba
julien.arino@umanitoba.ca

The University of Manitoba campuses are located on original lands of Anishinaabeg, Ininew, Anisininew, Dakota and Dene peoples, and on the National Homeland of the Red River Métis. We respect the Treaties that were made on these territories, we acknowledge the harms and mistakes of the past, and we dedicate ourselves to move forward in partnership with Indigenous communities in a spirit of Reconciliation and collaboration.

Outline

Why characterise graphs?

A few R preliminaries

FIGS-slides-admin/a_robot_looking_at_a_desolate_tree_as_if_they_were_about_to_pa

Measures specific to vertices

Measures at the graph level

Why characterise graphs?

A few `R` preliminaries

Measures specific to vertices

Measures at the graph level

Why characterise a graph

Graphs are everywhere!

To compare graphs, understand their properties, we need ways to describe their shape and characteristics

The global air transportation network

Example of spread of p-H1N1

Role of social networks in shaping disease transmission during a community outbreak of 2009 H1N1 pandemic influenza, Cauchemez *et al*, PNAS **108**(7):2825-2830 (2011)

Example of spread of MERS

Topological dynamics of the 2015 South Korea MERS-CoV spread-on-contact networks, Yang & Jung, Scientific Reports **10**:4327 (2020)

More disease transmission trees

Outbreak Trees has an extensive database of disease transmission trees

Some “measures” concern the vertices, others the graph as a whole

In all that follows, unless otherwise indicated, $G = (V, A)$ is a digraph. If undirected, we write $G = (V, E)$

Why characterise graphs?

A few R preliminaries

Measures specific to vertices

Measures at the graph level

R packages for analysing graphs

Two main packages: `network` and `igraph`

We will use `igraph`: if you learn how to use it in R, you can easily do the same in `Python`, `C/C++` or `Mathematica` !

So in the following, I will assume that we have used the command
`library(igraph)`

igraph documentation

These days, there is an issue with the igraph documentation site, whereas normally it is quite good

You can find it here

Do read the R vignette, though, as well as the manual

Setting up a graph

There are multiple ways to set up a graph in `igraph`. Of course, you will need

```
library(igraph)
```

Two main mechanisms:

1. Use a function to create a *known* graph
2. Implement your own graph, describing the vertices and the edges/arcs

Known graphs (a few)

- `make_lattice`
- `make_ring`
- `make_star`
- `make_tree`
- `make_line_graph`
- `make_full_graph`
- `make_bipartite_graph`
- `make_empty_graph`

Why characterise graphs?

A few `R` preliminaries

Measures specific to vertices

FIGS-slides-admin/Gemini_Generated_Image_4pxjh4pxjh4pxjh4.jpeg

Measures at the graph level

Measures specific to vertices

- **Centre of a graph**
- **Centrality – Betweenness and closeness**
- **Periphery of a graph**
- **Degree distribution**

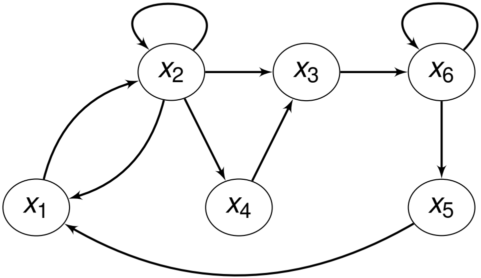
Geodesic distance

Definition 1 (Geodesic distance)

For $x, y \in V$, the **geodesic distance** $d(x, y)$ is the length of the shortest path from x to y, with $d(x, y) = \infty$ if no such path exists

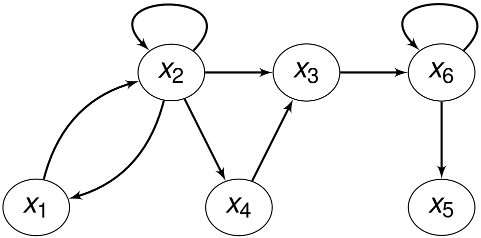
- $d(x_1, x_2) = 1$
- $d(x_1, x_3) = 2$
- $\ldots$

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 3 \\ 1 & 0 & 1 & 1 & 3 & 2 \\ 3 & 4 & 0 & 5 & 2 & 1 \\ 4 & 5 & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & 4 \\ 2 & 3 & 4 & 4 & 1 & 0 \end{pmatrix}$$



- $d(x_5, x_1) = \infty$
- $d(x_3, x_1) = \infty$
- $\dots$

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 3 \\ 1 & 0 & 1 & 1 & 3 & 2 \\ \infty & \infty & 0 & \infty & 2 & 1 \\ \infty & \infty & 1 & 0 & 3 & 2 \\ \infty & \infty & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & 1 & 0 \end{pmatrix}$$



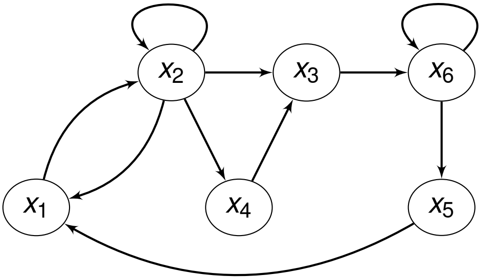
Eccentricity

Definition 2 (Vertex eccentricity)

The **eccentricity** $e(x)$ of vertex $x \in V$ is

$$e(x) = \max_{\substack{y \in V \\ y \neq x}} d(x, y)$$

$$\begin{pmatrix} 0 & 1 & 2 & 2 & \mathbf{4} & 3 \\ 1 & 0 & 1 & 1 & \mathbf{3} & 2 \\ 3 & 4 & 0 & \mathbf{5} & 2 & 1 \\ 4 & \mathbf{5} & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & \mathbf{4} \\ 2 & 3 & \mathbf{4} & \mathbf{4} & 1 & 0 \end{pmatrix}$$



Central points, radius and centre

Definition 3 (Central point)

A **central point** of G is a vertex x_0 with smallest eccentricity

Definition 4 (Radius)

The **radius** of G is $\rho(G) = e(x_0)$, where x_0 is a centre of G In other words,

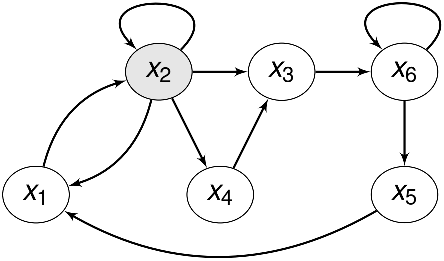
$$\rho(G) = \min_{x \in V} e(x)$$

Definition 5 (Centre)

The **centre** of G is the set of vertices that are central points of G, i.e.,

$$\{x \in V : e(x) = \rho(G)\}$$

$$\begin{pmatrix} 0 & 1 & 2 & 2 & \mathbf{4} & 3 \\ 1 & 0 & 1 & 1 & \mathbf{3} & 2 \\ 3 & 4 & 0 & \mathbf{5} & 2 & 1 \\ 4 & \mathbf{5} & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & \mathbf{4} \\ 2 & 3 & \mathbf{4} & \mathbf{4} & 1 & 0 \end{pmatrix}$$



Radius is 3, x_2 is a central point (the only one) and the centre is $\{x_2\}$

Measures specific to vertices

- Centre of a graph
- **Centrality – Betweenness and closeness**
- Periphery of a graph
- Degree distribution

How *central* is a vertex?

Centrality tries to answer the question: what are the most influent vertices?

We have seen central vertices and vertices on the periphery, let us consider two other measures of centrality

- Betweenness centrality
- Closeness centrality

Many other forms (we will come back to this, e.g., degree centrality)

Betweenness

Definition 6 (Betweenness)

$G = (V, A)$ a (di)graph. The **betweenness** of $v \in V$ is

$$b_{\mathcal{D}}(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

where

- σ_{st} is number of shortest geodesic paths from s to t
- $\sigma_{st}(v)$ is number of shortest geodesic paths from s to t through v

In other words

- For each pair of vertices (s, t), compute the shortest paths between them
- For each pair of vertices (s, t), determine the fraction of shortest paths that pass through vertex v
- Sum this fraction over all pairs of vertices (s, t)

Normalising betweenness

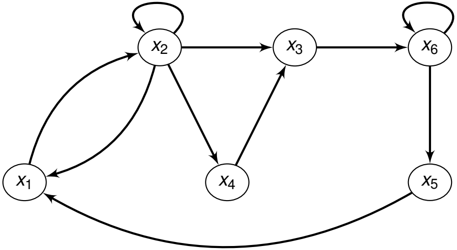
Betweenness may be normalized by dividing through the number of pairs of vertices not including v:

- for directed graphs, $(n-1)(n-2)$
- for undirected graphs, $(n-1)(n-2)/2$

Example of betweenness

```
distances(G, mode="out")
```

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 3 \\ 1 & 0 & 1 & 1 & 3 & 2 \\ 3 & 4 & 0 & 5 & 2 & 1 \\ 4 & 5 & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & 4 \\ 2 & 3 & 4 & 4 & 1 & 0 \end{pmatrix}$$



Number of shortest paths

Recall we found `distances(G, mode="out")`

$$\mathcal{D} = \begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 3 \\ 1 & 0 & 1 & 1 & 3 & 2 \\ 3 & 4 & 0 & 5 & 2 & 1 \\ 4 & 5 & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & 4 \\ 2 & 3 & 4 & 4 & 1 & 0 \end{pmatrix}$$

To find the number of shortest paths between pairs of vertices, we can use powers of the adjacency matrix

Write $\mathcal{D} = [d_{ij}]$, for a given (i, j) $(i \neq j)$, if $d_{ij} = k$, then pick the (i, j) in A^k

We find

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

Recall that betweenness of v is

$$b_{\mathcal{D}}(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

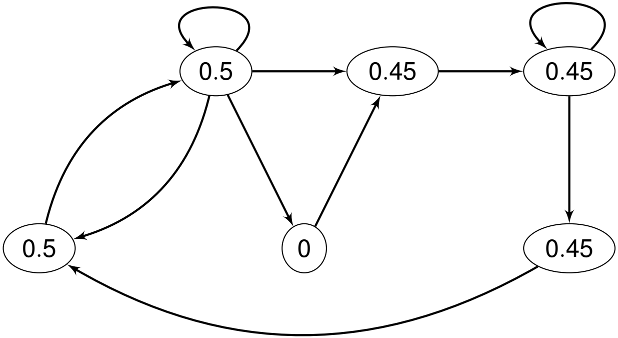
σ_{st} (# shortest paths from s to t) is found in the matrix above

What about $\sigma_{st}(v)$, # of those shortest paths that go through v?

We can use `all_shortest_paths(G, from = s, to = t, mode = "out")`

Example of betweenness

`betweenness(G, directed = FALSE, normalized = TRUE)`



Closeness

Definition 7

$G = (V, A)$. The **closeness** of $v \in V$ is

$$c_{\mathcal{D}}(v) = \frac{1}{n-1} \sum_{t \in V \setminus \{v\}} d_{\mathcal{D}}(v, t)$$

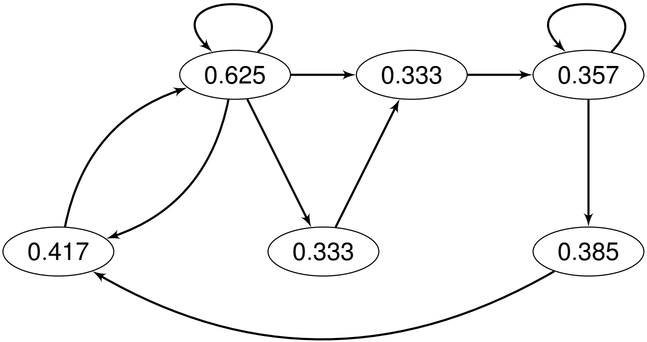
i.e., mean geodesic distance between a vertex v and all other vertices it has access to

Another definition is

$$c_{\mathcal{D}}(v) = \frac{1}{\sum_{t \in V \setminus \{v\}} d_{\mathcal{D}}(v, t)}$$

Example of (out) closeness

```
closeness(G, normalized = TRUE, mode=''out'')
```



Measures specific to vertices

- Centre of a graph
- Centrality – Betweenness and closeness
- **Periphery of a graph**
- Degree distribution

Diametre and periphery of a graph

Definition 8 (Diametre of a graph)

The **diametre** of G is

$$\delta(G) = \max_{\substack{x,y \in V \\ x \neq y}} d(x, y) = \max_{x \in V} e(x)$$

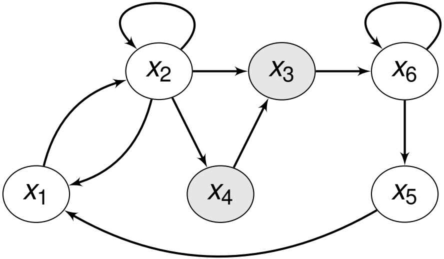
$\delta(G) < \infty \iff G$ strongly connected

Definition 9 (Periphery)

The **periphery** of a graph is the set of vertices whose eccentricity achieves the diametre, i.e.,

$$\{x \in V : e(x) = \delta(G)\}$$

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 3 \\ 1 & 0 & 1 & 1 & 3 & 2 \\ 3 & 4 & 0 & 5 & 2 & 1 \\ 4 & 5 & 1 & 0 & 3 & 2 \\ 1 & 2 & 3 & 3 & 0 & 4 \\ 2 & 3 & 4 & 4 & 1 & 0 \end{pmatrix}$$



Diametre is $\delta(G) = 5$ and periphery is $\{x_3, x_4\}$

Definition 10 (Antipodal vertices)

Vertices $x, y \in V$ are **antipodal** if $d(x, y) = \delta(G)$

Measures specific to vertices

- Centre of a graph
- Centrality – Betweenness and closeness
- Periphery of a graph
- **Degree distribution**

Degree distribution

Definition 11 (Arc incident to a vertex)

If a vertex x is the initial endpoint of an arc u, which is not a loop, the arc u is **incident out of vertex** x

The number of arcs incident out of x plus the number of loops attached to x is denoted $d_G^+(x)$ and is the **outer demi-degree** of x

An arc **incident into vertex** x and the **inner demi-degree** $d_G^-(x)$ are defined similarly

Definition 12 (Degree)

The **degree** of vertex x is the number of arcs with x as an endpoint, each loop being counted twice. The degree of x is denoted $d_G(x) = d_G^+(x) + d_G^-(x)$

If each vertex has the same degree, the graph is **regular**

Definition 13 (Isolated vertex)

A vertex of degree 0 is **isolated**.

Definition 14 (Average degree of G)

$d(G) = \frac{1}{|V|} \sum_{v \in V} deg_G(v)$.

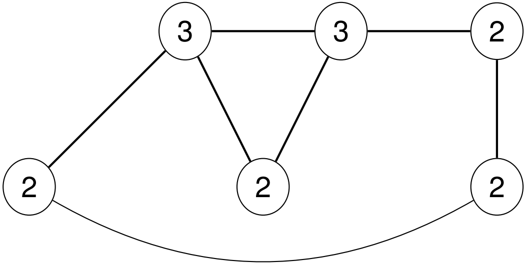
Definition 15 (Minimum degree of G)

$\delta(G) = \min\{deg_G(v) | v \in V\}$.

Definition 16 (Maximum degree of G)

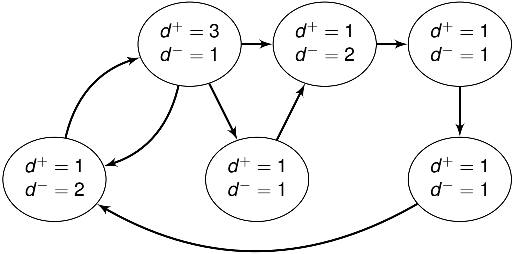
$\Delta(G) = \max\{deg_G(v) | v \in V\}$.

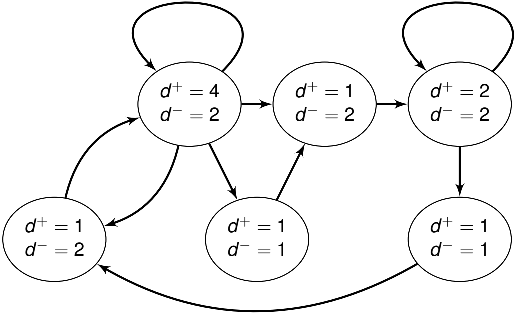
Degrees in an undirected graph



Here, vertices are labelled using the degree

Degrees in a directed graph





What to consider about degrees?

Degrees are often considered as a measure of popularity

Often write $k(i)$ (or k_i) for "degree of vertex i", $k^-(i)$ and $k^+(i)$ for in- and out-degree

- Minimum and maximum degree
- Minimum and maximum in/out-degree. E.g., if you consider the global air transportation network and the in/out-degree of airports, in-degree is a measure of a location's "popularity" as a travel destination
- Range of degrees in a graph: are there large discrepancies in connectivity between vertices in the graph?
- Average degree (often denoted $\langle k \rangle$ because of physicists)
- Average in/out-degree
- Variance of the degrees or in/out-degrees

- Average (nearest) neighbour degree, to encode for *preferential attachment* (one prefers to hang out with popular people)

$$k_i^{nn} = \frac{1}{k(i)} \sum_{j \in \mathcal{N}(i)} k(j)$$

or, in terms of the adjacency matrix $A = [a_{ij}]$,

$$k_i^{nn} = \frac{1}{k(i)} \sum_j a_{ij} k(j)$$

- *Excess degree*: take nearest neighbour degree but do not consider the edge/arc followed to get to the neighbour
- Degree, nearest neighbour and excess degree distributions

Degrees in `igraph`

- `degree` gives the degrees of the vertices
- `degree_distribution` gives numeric vector of the same length as the maximum degree plus one. The first element is the relative frequency zero degree vertices, the second vertices with degree one, etc.
- `knn` calculate the average nearest neighbor degree of the given vertices and the same quantity in the function of vertex degree
- `strength` sums up the edge weights of the adjacent edges for each vertex

Degree from adjacency matrix

Suppose adjacency matrix take the form $A = [a_{ij}]$ with $a_{ij} = 1$ if there is an arc from the vertex indexed i to the vertex indexed j and 0 otherwise. (Could be the other way round, using A^T, just make sure)

Let $\boldsymbol{e} = (1, \ldots, 1)^T$ be the vector of all ones

$A\boldsymbol{e} = (d_G^+(1), \ldots, d_G^+(1))^T$ (out-degree)

$\boldsymbol{e}^T A = (d_G^-(1), \ldots, d_G^-(1))$ (in-degree)

Why characterise graphs?

A few R preliminaries

Measures specific to vertices

FIGS-slides-admin/Gemini_Generated_Image_hfenashfenashfen.jpeg

Measures at the graph level

Measures at the graph level

- **Circumference & Girth**
- Graph density
- Graph connectivity
- Cliques
- k-cores

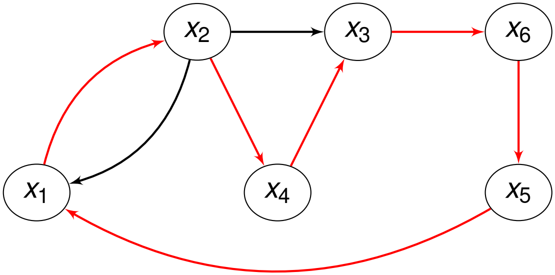
FIGS-slides-admin/Gemini_Generated_Image_a36uz1a36uz1a36u.jpeg

Circumference

Definition 17 (Circumference)

In an undirected (resp. directed) graph, the total number of edges (resp. arcs) in the longest cycle of graph G is the **circumference** of G

Circumference is 6.

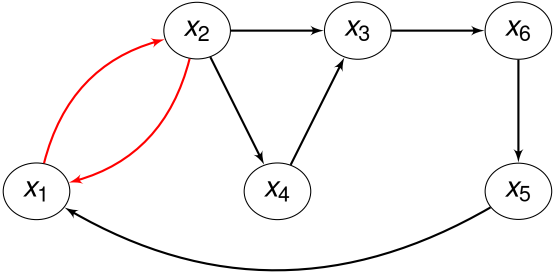


Girth

Definition 18 (Girth)

The total number of edges in the shortest cycle of graph G is the **girth** $g(G)$

Girth is 2.



Measures at the graph level

- Circumference & Girth
- **Graph density**
- Graph connectivity
- Cliques
- k-cores

FIGS-slides-admin/Gemini_Generated_Image_m4586wm4586wm458.jpeg

Completeness

Definition 19 (Complete undirected graph)

An undirected graph is complete if every two of its vertices are adjacent.

Definition 20 (Complete digraph)

A digraph $D(V, A)$ is complete if $\forall u, v \in V$, $uv \in A$.

In case of simple graphs, completeness effectively means that “information” can be transmitted from every vertex to every other vertex quickly (1 step)

It can be useful to know how far away we are from being complete

Number of edges/arcs in a complete graph

$G = (V, E)$ undirected and simple of order n has at most

$$\frac{n(n-1)}{2}$$

edges, while $G = (V, A)$ directed and simple of order n has at most

$$n(n-1)$$

arcs

Density of a graph

Definition 21 (Density)

The fraction of maximum number of edges or arcs present in the graph is the **density** of the graph.

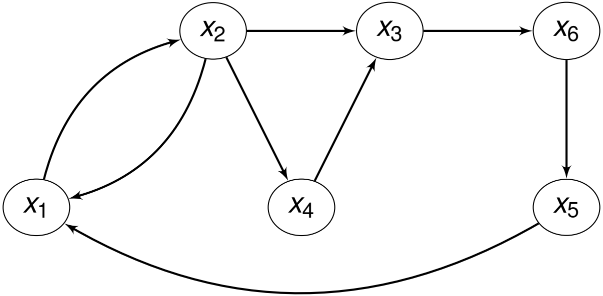
If the graph has p edges or arcs, then its density is, respectively,

$$\frac{2p}{n(n-1)}$$

or

$$\frac{p}{n(n-1)}$$

Example of density



Graph has order 6 and thus a max of 30 arcs. Here, 8 arcs $\Longrightarrow$ density 0.267 (26.7% of arcs are present)

Measures at the graph level

- Circumference & Girth
- Graph density
- **Graph connectivity**
- Cliques
- k-cores

FIGS-slides-admin/Gemini_Generated_Image_q8u933q8u933q8u9.jpeg

Connectedness

We have already seen connectedness (quasi- or strong in the oriented case)

Connectedness is important in terms of characteristing graph properties, as it shows the capacity of the graph to convey information to all the members of the graph (the vertices)

Definition 22 (Connected graph)

A **connected graph** is a graph that contains a chain $\mu[x, y]$ for each pair x, y of distinct vertices

Denote $x \equiv y$ the relation "$x = y$, or $x \neq y$ and there exists a chain in G connecting x and y". $\equiv$ is an equivalence relation since

1. $x \equiv y$ [reflexivity]
2. $x \equiv y \implies y \equiv x$ [symmetry]
3. $x \equiv y, y \equiv z \implies x \equiv z$ [transitivity]

Definition 23 (Connected component of a graph)

The classes of the equivalence relation $\equiv$ partition V into connected sub-graphs of G called **connected components**

Articulation set

Definition 24 (Articulation set)

For a connected graph, a set A of vertices is called an **articulation set** (or a **cutset**) if the subgraph of G generated by $V - A$ is not connected

`articulation_points(G)` in `igraph` (assumes the graph is undirected, makes it so if not)

Strongly connected graphs

$G = (V, U)$ connected. A **path of length 0** is any sequence $\{x\}$ consisting of a single vertex $x \in V$

For $x, y \in V$, let $x \equiv y$ be the relation "there is a path $\mu_1[x, y]$ from x to y as well as a path $\mu_2[y, x]$ from y to x". This is an equivalence relation (it is reflexive, symmetric and transitive)

Definition 25 (Strong components)

Sets of the form

$$A(x_0) = \{x : x \in V, x \equiv x_0\}$$

are equivalence classes; they partition V and are the **strongly connected components** of G

Definition 26 (Strongly connected graph)

G **strongly connected** if it has a single strong component

Definition 27 (Minimally connected graph)

G is **minimally connected** if it is strongly connected and removal of any arc destroys strong-connectedness

Definition 28 (Contraction)

$G = (V, U)$. The **contraction** of the set $A \subset V$ of vertices consists in replacing A by a single vertex a and replacing each arc into (resp. out of) A by an arc with same index into (resp. out of) a

Quasi-strong connectedness

Definition 29 (Quasi-strong connectedness)

G **quasi-strongly connected** if $\forall x, y \in V$, exists $z \in V$ (denoted $z(x, y)$ to emphasize dependence on x, y) from which there is a path to x and a path to y

Strongly connected $\implies$ quasi-strongly connected (take $z(x, y) = x$); converse not true

Quasi-strongly connected $\implies$ connected

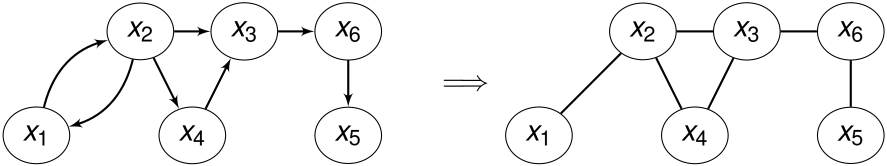
Lemma 30

$G = (V, U)$ has a root $\iff$ G quasi-strongly connected

Weak-connectedness

Definition 31 (Weakly connected graph)

$G = (V, U)$ **weakly connected** if $G = (V, E)$ connected, where E is obtained from U by ignoring the direction of arcs



Weak components

Define for $x, y \in V$ the relation $x \equiv y$ as "$x = y$ or $x \neq y$ and there is a chain in G connecting x and y" [like for components in an undirected graph, except the graph is directed here]

This defines an equivalence relation

Definition 32 (Weak components)

Sets of the form

$$A(x_0) = \{x : x \in V, x \equiv x_0\}$$

are equivalence classes partitioning V into the **weakly connected components** of G

$G = (V, U)$ is weakly connected if there is a single weak component

Components in `igraph`

- `is_connected` decides whether the graph is weakly or strongly connected
- `components` finds the maximal (weakly or strongly) connected components of a graph
- `count_components` does almost the same as components but returns only the number of clusters found instead of returning the actual clusters
- `component_distribution` creates a histogram for the maximal connected component sizes
- `decompose` creates a separate graph for each component of a graph
- `subcomponent` finds all vertices reachable from a given vertex, or the opposite: all vertices from which a given vertex is reachable via a directed path

Measures at the graph level

- Circumference & Girth
- Graph density
- Graph connectivity
- **Cliques**
- *k*-cores

Cliques

Definition 33 (Clique in undirected graphs)

$G = (V, E)$ a simple undirected graph. A **clique** is a subgraph G' of G such that all vertices in G' are adjacent

Definition 34 (n-clique)

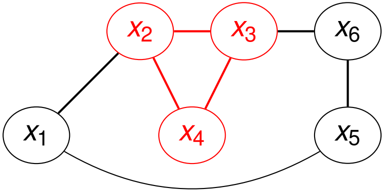
A simple, complete graph on n vertices is called an n-**clique** and is often denoted K_n

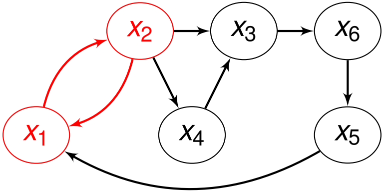
Definition 35 (Clique in directed graphs)

$G = (V, U)$ a simple directed graph. A **clique** is a subgraph G' of G such that all vertices in G' are mutually adjacent

Definition 36 (Maximal clique)

A **maximal clique** is a clique that cannot be extended by adding another adjacent vertex


x_2
x_3
x_6
x_1
x_4
x_5


x_2
x_3
x_6
x_1
x_4
x_5

Cliques in `igraph`

- `cliques` find all complete subgraphs in the input graph, obeying the size limitations given in the min and max arguments
- `largest_cliques` finds all largest cliques in the input graph
- `max_cliques` finds all maximal cliques in the input graph (The largest cliques are always maximal, but a maximal clique is not necessarily the largest)
- `count_max_cliques` counts the maximal cliques
- `clique_num` calculates the size of the largest clique(s)

Measures at the graph level

- Circumference & Girth
- Graph density
- Graph connectivity
- Cliques
- ***k*-cores**

k-core

Definition 37 (k-core of a graph)

$G = (V, U)$ a graph. The **k-core** of G is a maximal subgraph in which each vertex has degree at least k

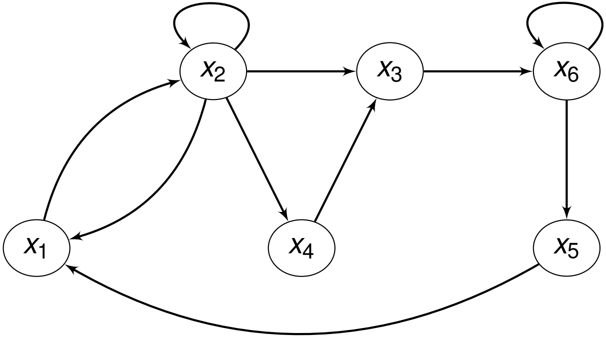
Definition 38 (Coreness of a vertex)

$G = (V, U)$ a graph, $x \in V$. The **coreness** of x is k if x belongs to the k-core of G but not to the $k + 1$ core of G

For directed graphs, in-cores or out-cores depending on whether in-degree or out-degree is used

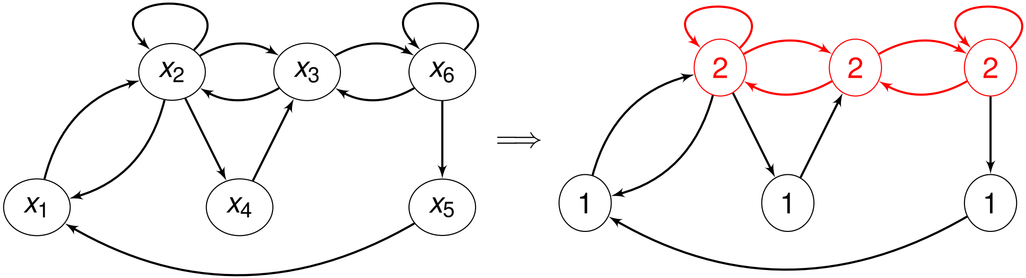
In `igraph`: `coreness`

Coreness in the directed case



G has only a 1-in-core and 1-out-core: there is no (maximal) subgraph in which the in- or out-degree is larger than 1

In-coreness in the directed case



Coreness in the undirected case