Graph Basics

- Graphs
- Properties of Graphs
- Graph Terminology

❖ Graphs

Many applications require

- a collection of items (i.e. a set)
- relationships/connections between items

Examples:

- maps: items are cities, connections are roads
- web: items are pages, connections are hyperlinks

Collection types you're familiar with

- lists ... linear sequence of items (COMP1511)
- trees ... branched hierachy of items (Weeks 02/03)

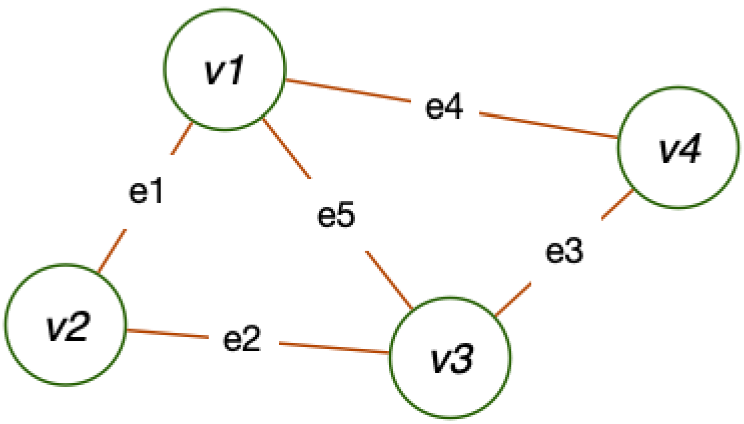
Graphs are more general ... allow arbitrary connections

❖ ... Graphs

A graph $G = (V,E)$

- V is a set of vertices
- E is a set of edges (subset of $V \times V$)

Example:



$V = \{ v1, v2, v3, v4 \}$

$E = \{ e1, e2, e3, e4, e5 \}$

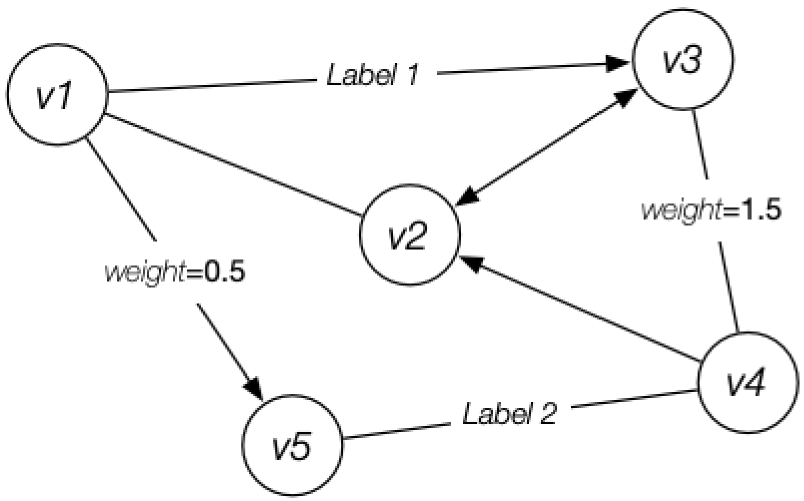
or

$E = \{ (v1,v2), (v2,v3), (v3,v4), (v1,v4), (v1,v3) \}$

❖ ... Graphs

Nodes are distinguished by a unique identifier

Edges may be (optionally) directed, labelled and/or weighted



❖ ... Graphs

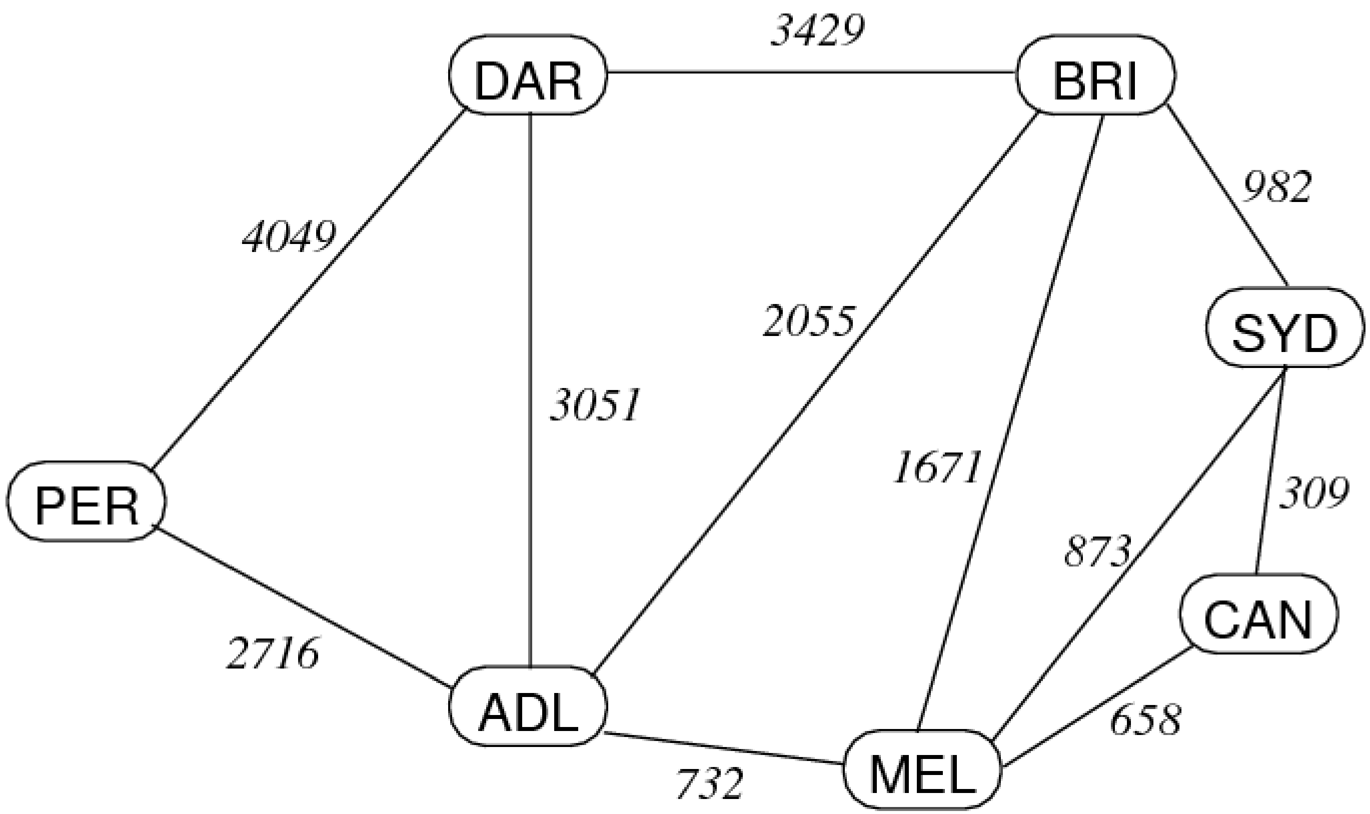
A real example: Australian road distances

Distance	Adelaide	Brisbane	Canberra	Darwin	Melbourne	Perth	Sydney
Adelaide	-	2055	1390	3051	732	2716	1605
Brisbane	2055	-	1291	3429	1671	4771	982
Canberra	1390	1291	-	4441	658	4106	309
Darwin	3051	3429	4441	-	3783	4049	4411
Melbourne	732	1671	658	3783	-	3448	873
Perth	2716	4771	4106	4049	3448	-	3972
Sydney	1605	982	309	4411	873	3972	-

Notes: vertices are cities, edges are distance between cities, symmetric

❖ ... Graphs

Alternative representation of above:



❖ ... Graphs

Questions we might ask about a graph:

- is there a way to get from item A to item B?
- what is the best way to get from A to B?
- which items are directly connected (A↔B)?

Graph algorithms are generally more complex than tree/list ones:

- no implicit order of items
- graphs may contain cycles
- concrete representation is less obvious
- algorithm complexity depends on connection complexity

❖ Properties of Graphs

Terminology: *|V|* and *|E|* (cardinality) normally written just as *V* and *E*.

A graph with *V* vertices has at most *V(V-1)/2* edges.

The ratio *E:V* can vary considerably.

- if *E* is closer to V^2, the graph is dense
- if *E* is closer to *V*, the graph is sparse
  - Example: web pages and hyperlinks

Knowing whether a graph is sparse or dense is important

- may affect choice of data structures to represent graph
- may affect choice of algorithms to process graph

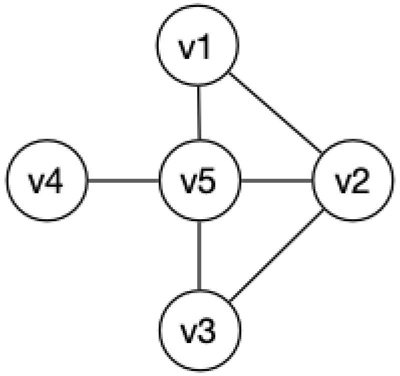
❖ Graph Terminology

For an edge *e* that connects vertices *v* and *w*

- *v* and *w* are adjacent (neighbours)
- *e* is incident on both *v* and *w*

Degree of a vertex *v*

- number of edges incident on *e*



degree(v1) = 2
degree(v2) = 3
degree(v3) = 2
degree(v4) = 1
degree(v5) = 4

Synonyms:

- vertex = node
- edge = arc = link (Note: some people use arc for *directed* edges)

❖ ... Graph Terminology

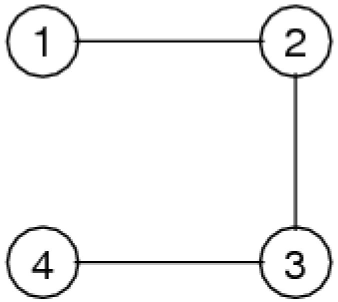
Path: a sequence of vertices where

- each vertex has an edge to its predecessor

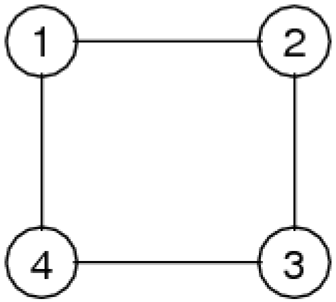
Cycle: a path where

- last vertex in path is same as first vertex in path

Length of path or cycle = #edges



Path: 1–2, 2–3, 3–4



Cycle: 1–2, 2–3, 3–4, 4–1

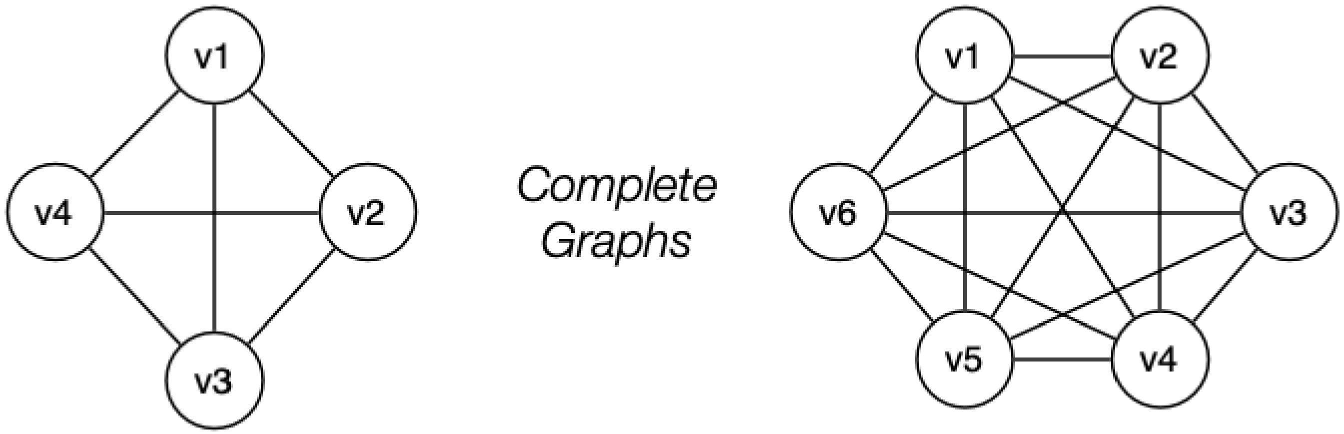
❖ ... Graph Terminology

Connected graph

- there is a *path* from each vertex to every other vertex
- if a graph is not connected, it has ≥2 connected components

Complete graph K_V

- there is an *edge* from each vertex to every other vertex
- in a complete graph, $E = V(V-1)/2$



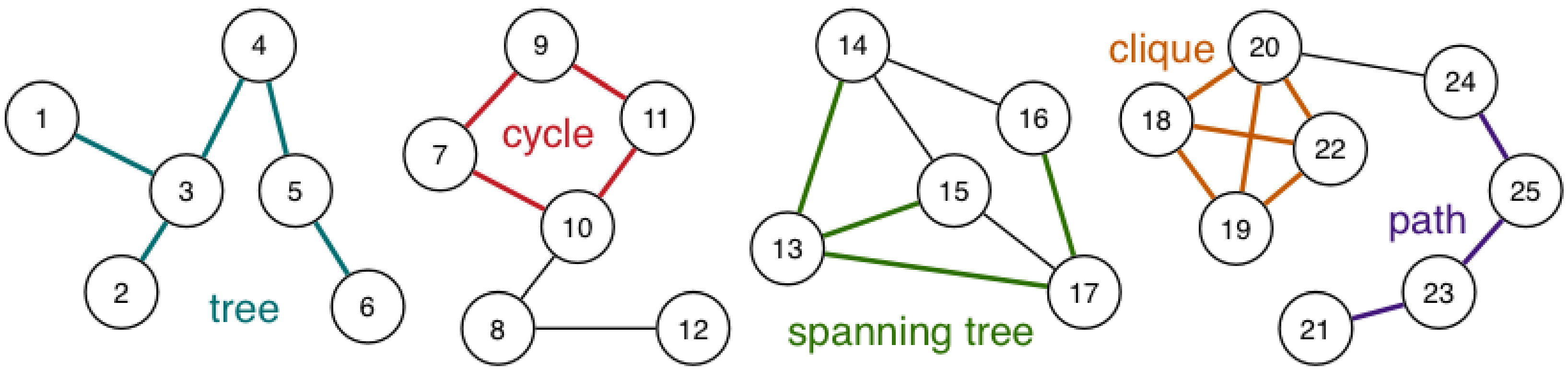
❖ ... Graph Terminology

Tree: connected (sub)graph with no cycles

Spanning tree: tree containing all vertices

Clique: complete subgraph

Consider the following *single graph*:



This graph has 25 vertices, 32 edges, and 4 connected components

Note: The entire graph has no spanning tree; what is shown in green is a spanning tree of the third connected component

❖ ... Graph Terminology

A spanning tree of connected graph $G = (V,E)$

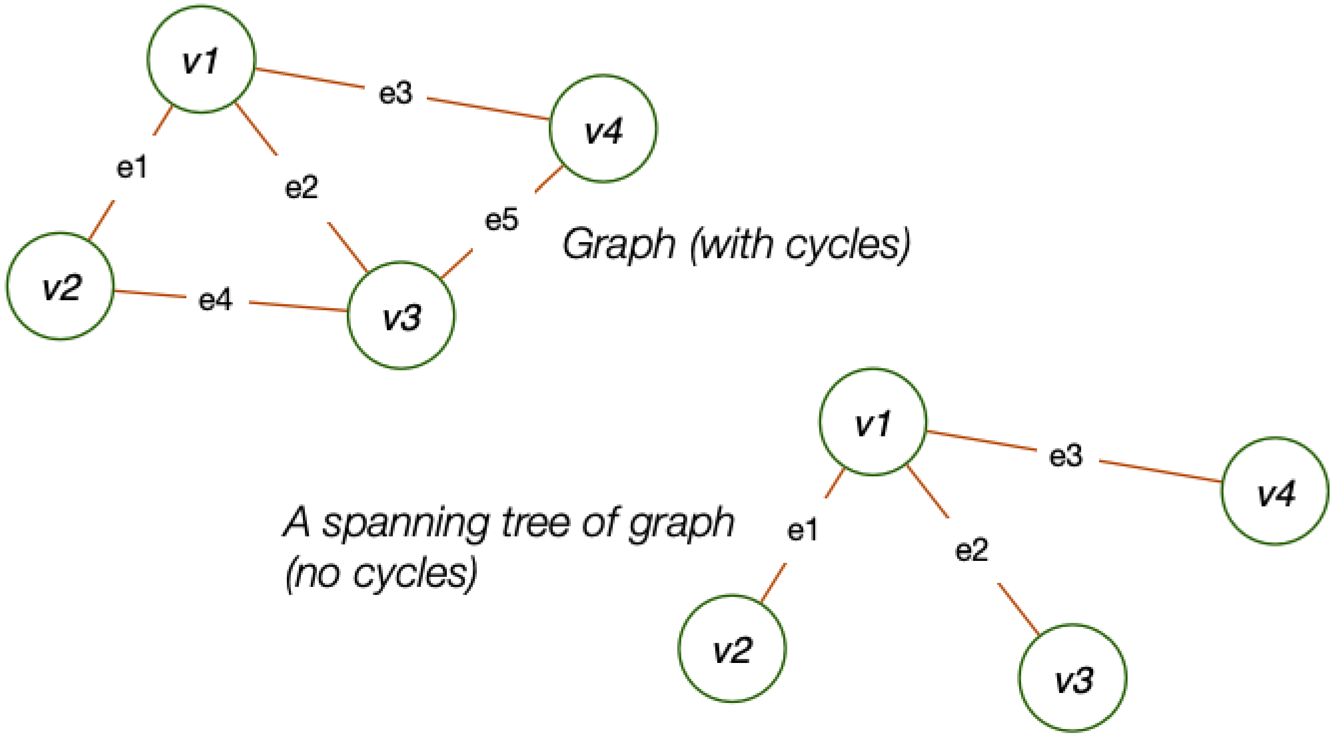
- is a subgraph of G containing all of V
- and is a single tree (connected, no cycles)

A spanning forest of non-connected graph $G = (V,E)$

- is a subgraph of G containing all of V
- and is a set of trees (not connected, no cycles),
  - with one tree for each *connected component*

❖ ... Graph Terminology

Can form spanning tree from graph by removing edges



Many possible spanning trees can be formed. Which is "best"?

❖ ... Graph Terminology

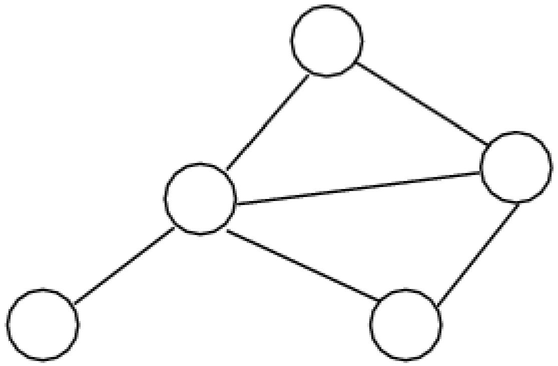
Undirected graph

- *edge(u,v) = edge(v,u)*, no self-loops (i.e. no *edge(v,v)*)

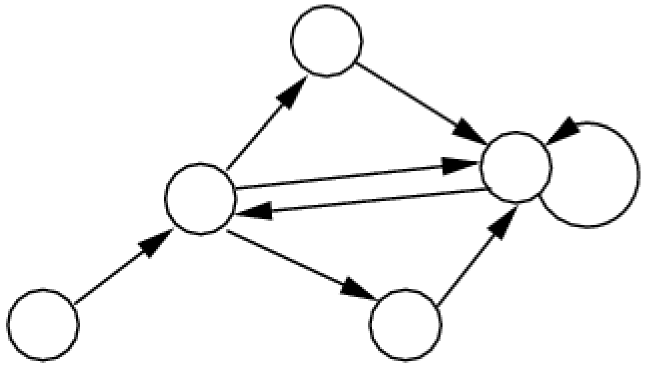
Directed graph

- *edge(u,v) ≠ edge(v,u)*, can have self-loops (i.e. *edge(v,v)*)

Examples:



Undirected graph



Directed graph

❖ ... Graph Terminology

Other types of graphs ...

Weighted graph

- each edge has an associated value (weight)
- e.g. road map (weights on edges are distances between cities)

Multi-graph

- allow multiple edges between two vertices
- e.g. function call graph (**`f()`** calls **`g()`** in several places)

Labelled graph

- edges have associated labels
- can be used to add semantic information

Produced: 21 Jun 2020