COMP2521 23T3
Directed and Weighted Graphs

Kevin Luxa

cs2521@cse.unsw.edu.au

directed graphs
weighted graphs

Generalising Graphs

In graphs representing real-world scenarios,
edges are often directional
and have a sense of cost

Thus, we need to consider directed and weighted graphs

Directed Graphs

We've mostly considered *undirected* graphs:
an edge relates two vertices equivalently.

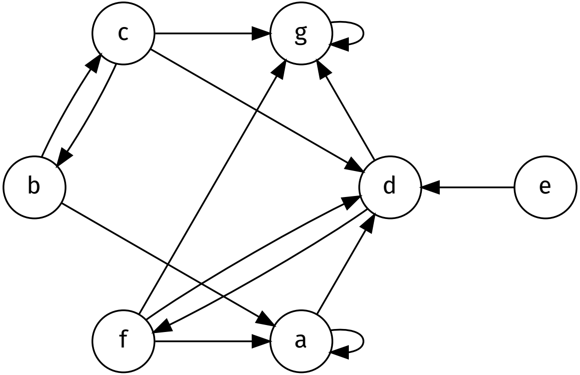
Some applications require us to consider
directional edges: $v \to w \neq w \to v$
e.g., 'follow' on Twitter, one-way streets, etc.

In an directed graph or digraph:
edges have direction;
self-loops are allowed.

Each edge (v, w) has a source v and a destination w.

Directed Graphs

Example



Directed Graphs

Common Domains

domain	vertex is...	edge is...
WWW	web page	hyperlink
chess	board state	legal move
scheduling	task	precedence
program	function	function call
journals	article	citation

Digraph Terminology (I)

in-degree or $d^{-1}(v)$: the number of directed edges leading into a vertex
out-degree or $d(v)$: the number of directed edges leading out of a vertex

Digraph Terminology (II)

directed path a sequence of vertices v_1, v_2, ...v_n such that v_i has an outgoing edge to v_{i+1}

directed cycle a directed path where the first and last vertices are the same

Digraph Terminology (III)

reachability indicates existence of directed path:
if a directed path $v, \ldots, w$ exists,
w is reachable from v

strongly connected indicates mutual reachability:
if both paths $v, \ldots, w$ and $w, \ldots, v$ exist,
v and w are strongly connected

strong connectivity every vertex reachable from every other vertex;

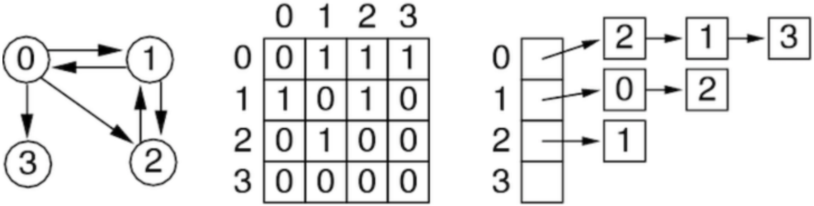
strongly-connected component maximal strongly-connected subgraph

Digraph Representation

Similar choices as for undirected graphs:

- adjacency matrix … asymmetric, sparse; less space efficient
- adjacency lists … fairly common solution
- edge lists … order of edge components matters

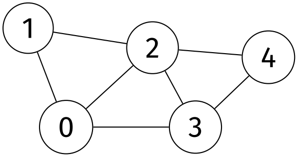
Can we make our undirected graph implementations directed? Yes!

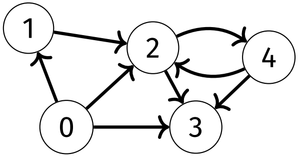


digraph *adj matrix* *adj lists*

Directed Graphs

Implementation: Adjacency Matrix





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

unweighted, undirected

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

unweighted, directed

Digraph Complexity

	storage	edge add	has edge	outdegree
adjacency matrix	$O\left(V^2\right)$	$O(1)$	$O(1)$	$O(V)$
adjacency list	$O(V+E)$	$O(d(v))$	$O(d(v))$	$O(d(v))$
array of edges	$O(E)$	$O(E)$	$O(E)$	$O(E)$

Overall, adjacency lists tend to be ideal:
real digraphs tend to be sparse
(large V, small average $d(v)$);
algorithms often iterate over v's edges

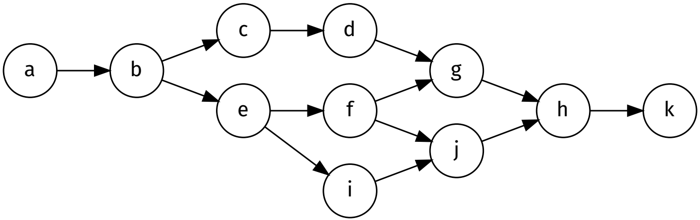
Directed Graphs

Problems and Applications

- Is there a directed path from s to t? (transitive closure)
- What is the shortest path from s to t? (shortest path search)
- Are all vertices mutually reachable? (strong connectivity)

- How can I organise a set of tasks? (topological sort)
- How can I crawl the web? (graph traversal)
- Which web pages are important? (PageRank)

Directed Acyclic Graphs



Is it a tree? Is it a graph?
No: it's a DAG, a directed acyclic graph.

Tree-like: each vertex has 'children'.
Graph-like: a child vertex may have multiple parents.

Directed Acyclic Graphs

Application: the Topological Sort

NOT EXAMINABLE (and not taught until '4128)

The most common application of a DAG is *topological sorting*:
ordering vertices such that, for any vertices u and v,
if u has a directed edge to v, then v comes after u in the ordering.

Directed Acyclic Graphs

Application: the Topological Sort

NOT EXAMINABLE (and not taught until '4128)

The most common application of a DAG is *topological sorting*:
ordering vertices such that, for any vertices u and v,
if u has a directed edge to v, then v comes after u in the ordering.

Computable with a DFS, tracking *post-order sequence*:
vertices only added after their children have been visited
$\Rightarrow$ a valid topological ordering

Directed Acyclic Graphs

Application: the Topological Sort

NOT EXAMINABLE (and not taught until '4128)

The most common application of a DAG is *topological sorting*:
ordering vertices such that, for any vertices u and v,
if u has a directed edge to v, then v comes after u in the ordering.

Computable with a DFS, tracking *post-order sequence*:
vertices only added after their children have been visited
$\Rightarrow$ a valid topological ordering

dependency problems: *make(1)*, spreadsheets
version-control systems: Git, Fossil, etc.

Digraph Traversal

Mostly the same algorithms as for undirected graphs:
DFS and BFS should all Just Work

e.g., Web crawling: visit every page on the web.
BFS with implicit graph;
on visit, scans page for content, keywords, links
... assumption: WWW is fully connected.

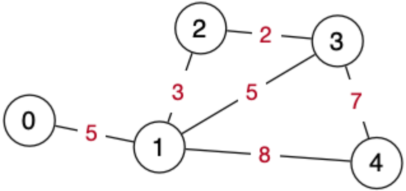
Weighted Graphs

Weighted Graphs

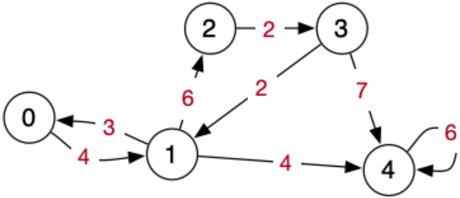
Some applications require us to consider a cost or weight assigned to a relation between two nodes.

In a weighted graph, each edge (s, t, w) has a weight w.

Weights can be used in both directed and undirected graphs.

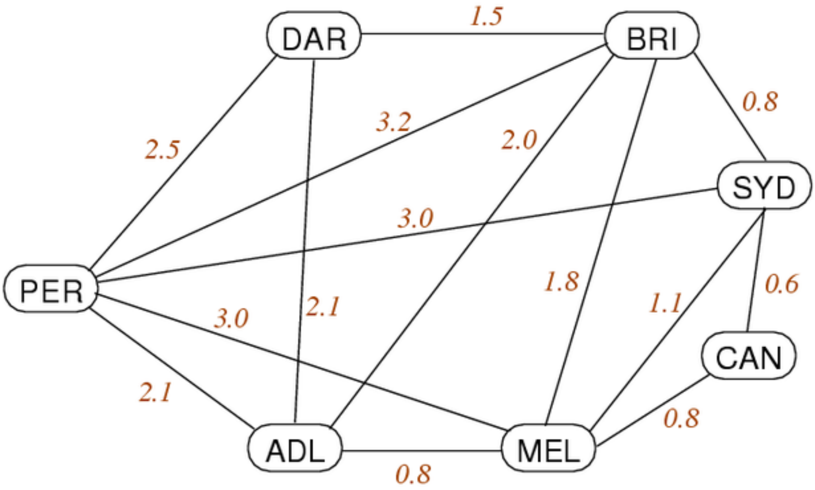


Weighted Graph



Directed Weighted Graph

Example: Major airline routes in Australia



Weighted Graphs

Implementation

Adjacency matrix:

- store *weight* in each cell, not just true/false.
- need some “no edge exists” value: zero might be a valid weight.

Adjacency list

- add weight to each list node

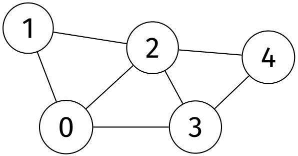
Edge list:

- add weight to each edge

Works for directed and undirected graphs!

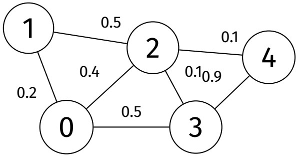
Weighted Graphs

Implementation: Adjacency Matrix



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

unweighted, undirected



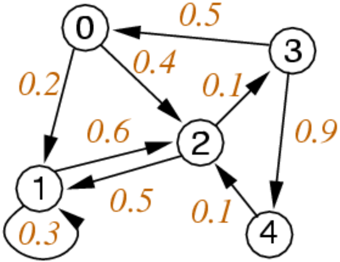
$$\begin{bmatrix} - & 0.2 & 0.4 & 0.5 & - \\ 0.2 & - & 0.5 & - & - \\ 0.4 & 0.5 & - & 0.1 & 0.1 \\ 0.5 & - & 0.1 & - & 0.9 \\ - & - & 0.1 & 0.9 & - \end{bmatrix}$$

weighted, undirected

Weighted Graphs

Implementation: Adjacency Matrix

Weighted directed graph:



Weighted Digraph

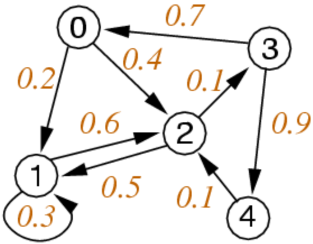
	0	1	2	3	4
0	*	0.2	0.4	*	*
1	*	0.3	0.6	*	*
2	*	0.5	*	0.1	*
3	0.5	*	*	*	0.9
4	*	*	0.1	*	*

Adjacency Matrix

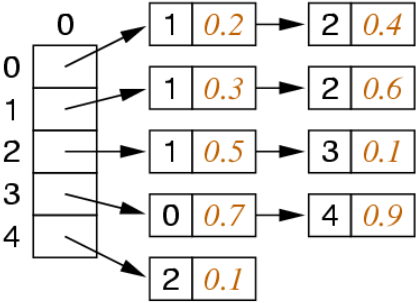
Weighted Graphs

Implementation: Adjacency List

Weighted directed graph:



Weighted Digraph

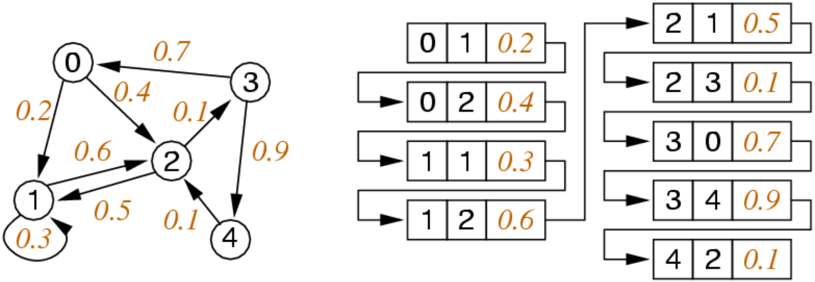


Adjacency Lists

Weighted Graphs

Implementation: Array of Edges

Weighted directed graph:



Weighted Digraph   *Edge List*

Feedback

https://forms.office.com/r/aPF09YHZ3X