CS1113
Graphs in Computer Science

Lecturer:

Professor Barry O’Sullivan

Office: 2.65, Western Gateway Building

email: *b.osullivan@cs.ucc.ie*

http://osullivan.ucc.ie/teaching/cs1113/

An introduction to Graphs

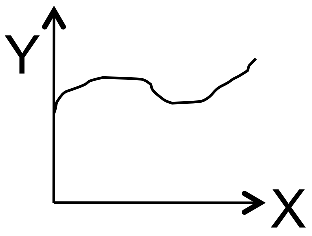
Graphs in computer science

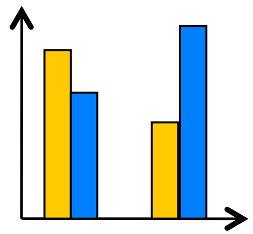
simple graphs
multigraphs
directed graphs

simple graph properties

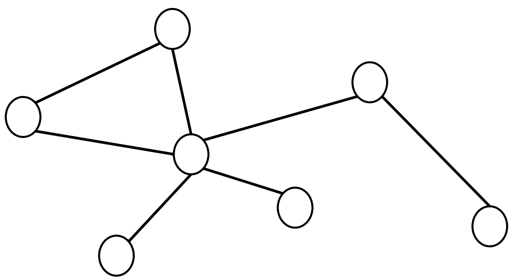
What is a graph?

1. a visual representation of the relationship between the values of two or more variables





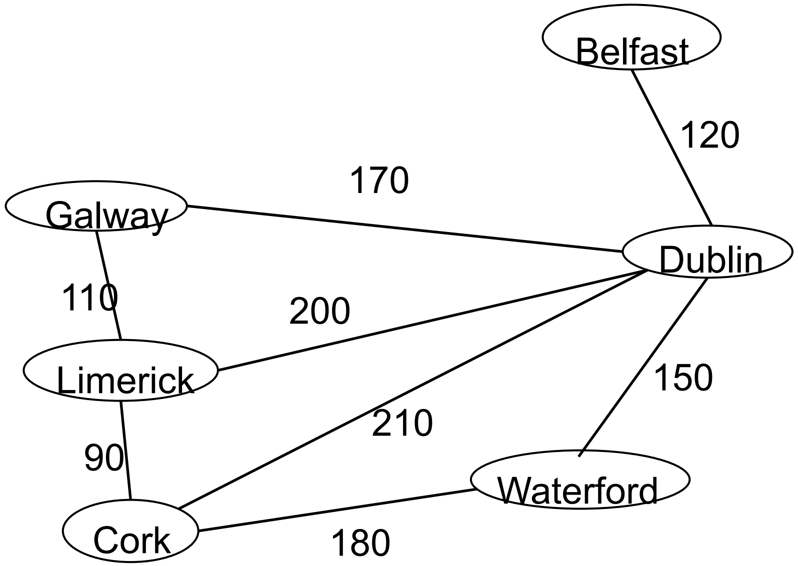
2. a representation of the relationships between multiple entities



the standard computer science use

Examples

representing main roads for route planning



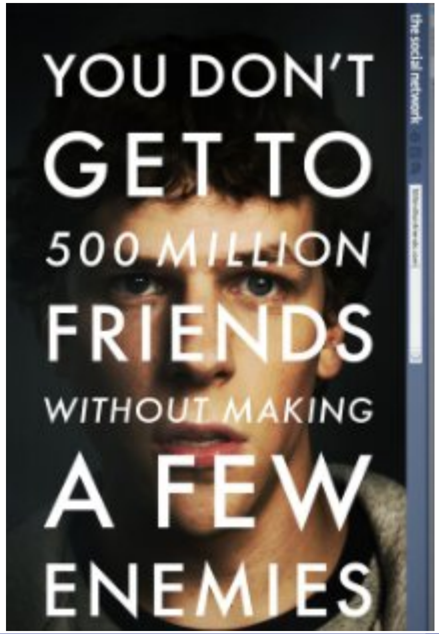
augmented with travel times

note: numbers attached to links

Social Networks

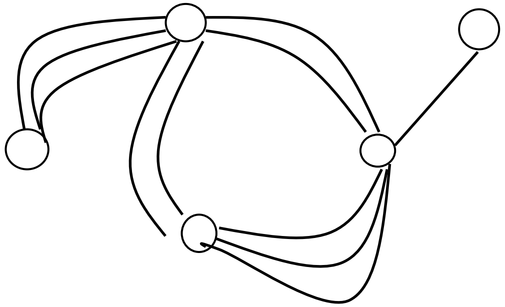
A social network is a graph where the objects are people, and the links show a social relationship.

"FOAF (an acronym of Friend of a Friend) is a machine-readable ontology describing persons, their activities and their relations to other people and objects.
FOAF allows groups of people to describe social networks ...
FOAF is an extension to RDF Resource Description Framework and is defined using OWL Web Ontology Language. "
(From wikipedia)



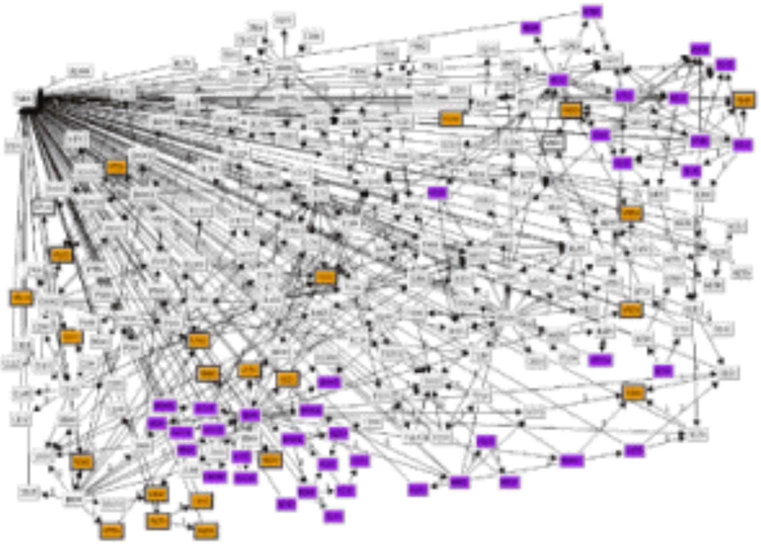
How far apart can two people be? How many links do I need to follow to get from one website to another? Which websites (or people) are the most connected?

Call graphs



Each link represents a call between two telephone numbers

Note: multiple links between nodes



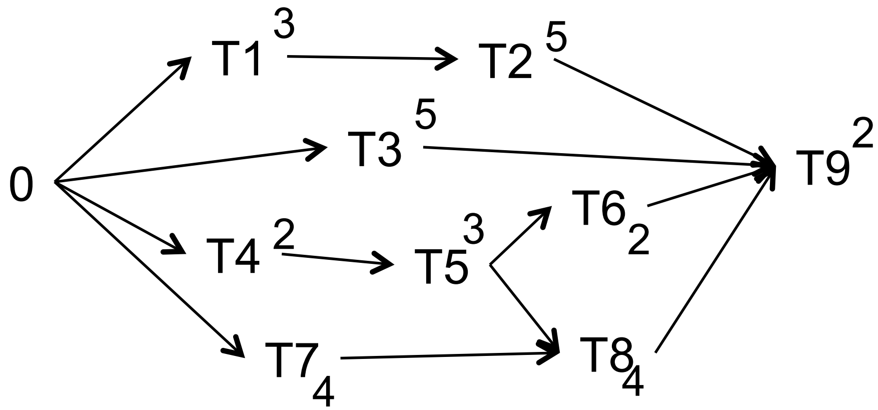
Each link represents a function call between two different modules in a large software system

Recursive functions will require an edge from a node to itself

Image taken from "Release: Reconstruction of Legacy Systems for Evolutionary Change", by Munro Burd and Young, The Centre for Software Maintenance, University of Durham

Project Planning

A set of tasks to complete. Graph shows precedence order.



Numbers show time to complete the task

Note: nodes now have a label

How quickly can I finish all my tasks?
How many resources would I need?

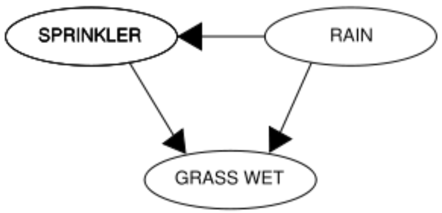
Note: links now have a direction

If I have 2 resources, how quickly can I finish?

Bayesian Networks and Influence Diagrams

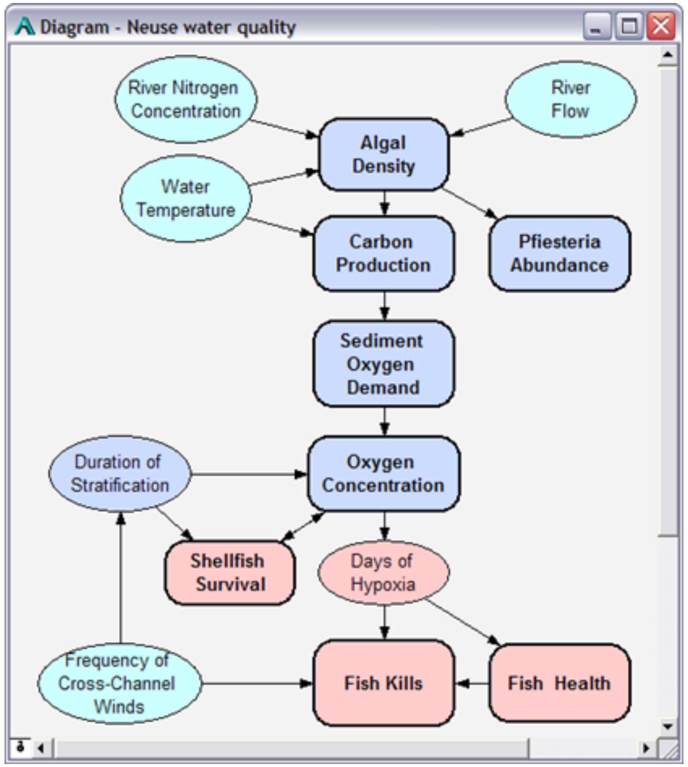
representing the presence of an influence between two parameters

RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99



RAIN T	RAIN F
0.2	0.8

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

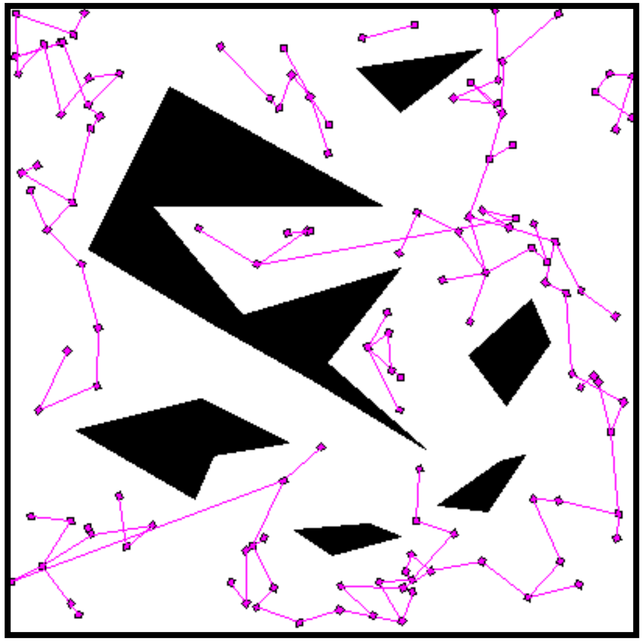


Note: complex information now stored with each node

image taken from
http://www.lumina.com/casestudies/NeuseEstuary.htm

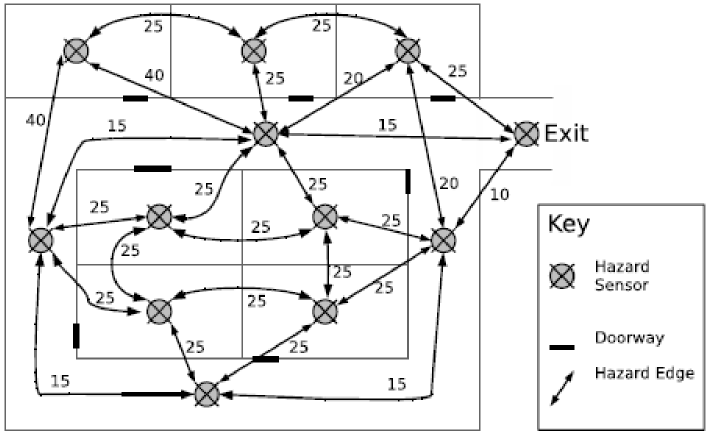
Path planning in computer games

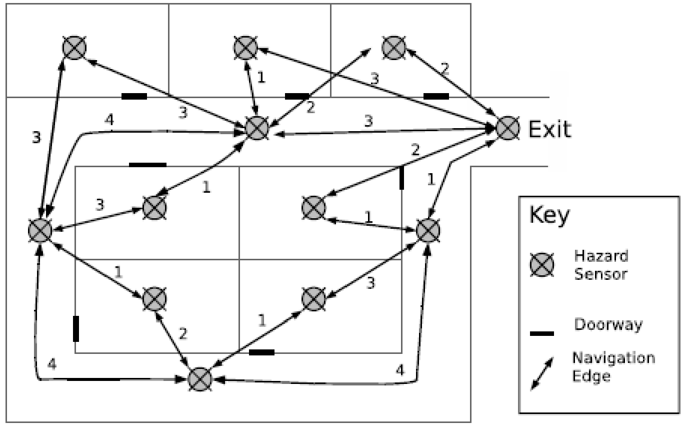
- intelligent agents must plan fast routes through an environment to achieve tasks or to defeat opponents
- environment is represented internally as a graph (which might be changing)

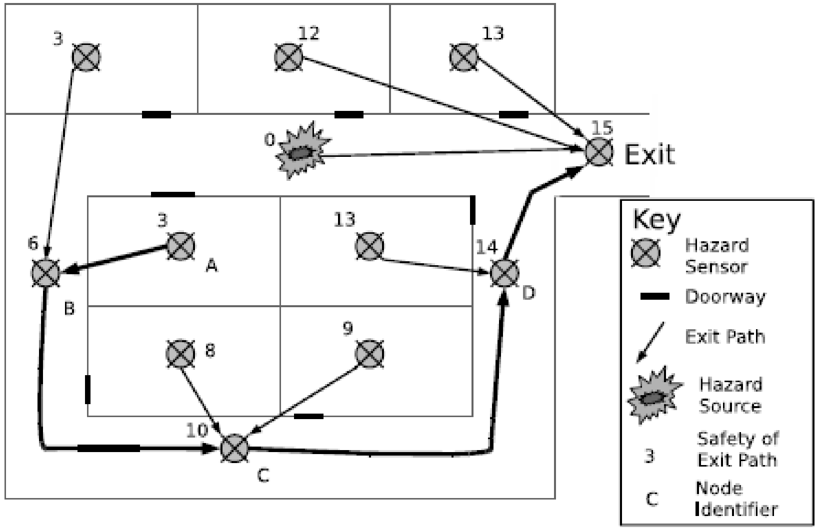


images taken from "Motion planning in games" talk by Marc Overmars at Intl workshop on Motion Planning in Virtual Environments", 2005
- see http://people.cs.uu.nl/markov/
- see also thesis on Quake bots http://www.kbs.twi.tudelft.nl/docs/MSc/2001/Waveren_Jean-Paul_van/thesis.pdf

Preparing Evacuation Routes







images taken from "Emergency Evacuation using Wireless Sensor Networks", by Matthew Barnes, Hugh Leather and D. K. Arvind

Graphs in Computer Science

Graphs are everywhere in computer science:

- in the representation of application problems
- in general algorithms and data structures

We will have to work out answers to questions like:

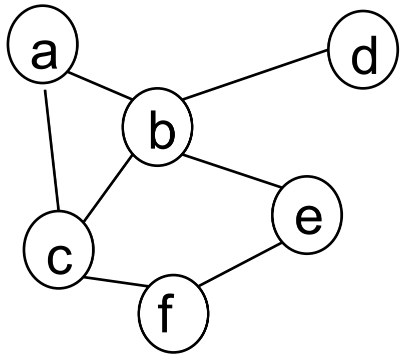
- are all the nodes connected to the rest of the graph?
- what is the shortest path between two particular nodes?
- on average, how many links separate any two nodes?

We need to

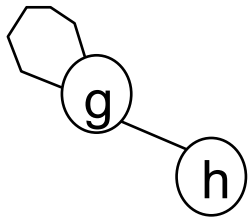
- agree a consistent language for talking about graphs
- work out how to represent them in programs
- understand the main algorithms for computing with graphs

Simple Graphs

A simple graph G is a pair (V,E), where
- V is a set of vertices (representing the objects)
- E is a set of edges, where each edge in E is a set of 1 or 2 vertices (representing the links between vertices)



G = (V,E), where
V = {a,b,c,d,e,f}
E = {{a,b},{a,c},{b,c},{b,d},{b,e},{c,f},{e,f}}



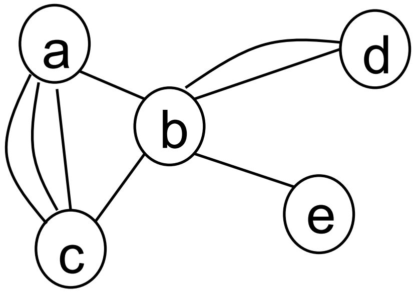
G' = (V',E'), where
V' = {g,h}
E' = {{g},{g,h}}

Multigraphs

A multigraph G is a pair (V,E), where

V is a set of vertices (representing the objects)

E is a *bag* of edges, where each edge in E is a set of 1 or 2 vertices (representing the links between vertices)



G = (V,E), where

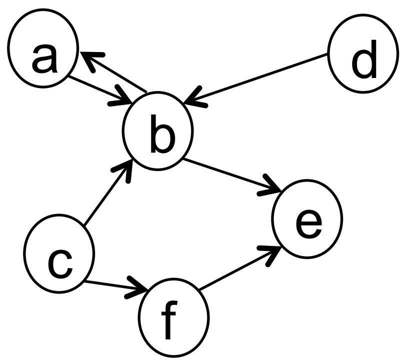
V = {a,b,c,d,e}

E = {{a,b},{a,c},{a,c},{a,c},{b,c},{b,d},{b,d},{b,e}}

A bag, or multiset, is a set in which repeated elements are allowed

Directed Graphs

A directed graph G is a pair (V,E), where
V is a set of vertices (representing the objects)
E is a set of edges, where each edge in E is an ordered pair of vertices (representing the links between vertices)



G = (V,E), where
V = {a,b,c,d,e,f}
E = {(a,b),(b,a),(b,e),(c,b),(c,f),(d,b),(f,e)}

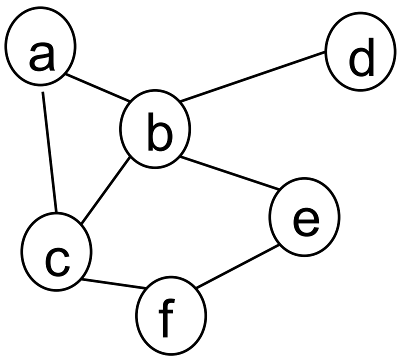
Note: the order in the ordered pair matters.
(a,b) and (b,a) are different edges

Adjacent vertices

Assume we have a simple graph G = (V,E).

Two vertices, v_1 and v_2, are adjacent if there is an edge $\{v_1,v_2\}$ in E.

If edge x = $\{v_1,v_2\}$, then x is incident on v_1 and v_2. v_1 and v_2 are the endpoints of e.

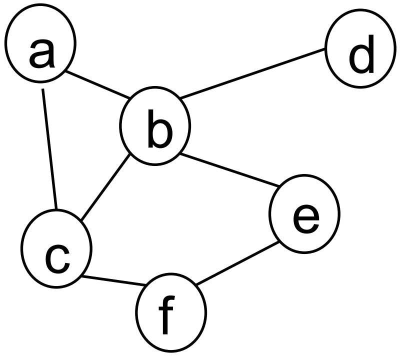


G = (V,E), where
V = {a,b,c,d,e,f}
E = {{a,b},{a,c},{b,c},{b,d},{b,e},{c,f},{e,f}}

Vertices *a* and *b* are adjacent.
Vertices *c* and *d* are not adjacent.

Vertex degree

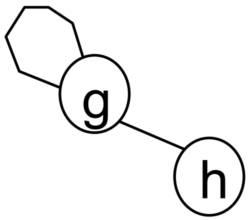
The degree of a vertex, v, is the number of times edges are incident on v (so an edge from v to itself counts twice)



G = (V,E), where
V = {a,b,c,d,e,f}
E = {{a,b},{a,c},{b,c},{b,d},{b,e},{c,f},{e,f}}

The degree of vertex *a* is 2, and degree of vertex *b* is 4.

The function deg:V→N returns the degree of any vertex.



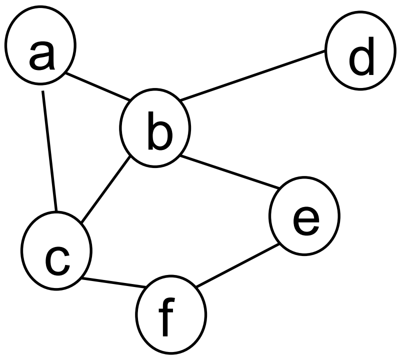
G' = (V',E'), where
V' = {g,h}
E' = {{g},{g,h}}

The degree of vertex *g* is 3, and degree of vertex *h* is 1.

The sum of the degrees of a simple graph

The sum of the degrees of a simple graph is equal to twice the number of edges in the graph.

if G = (V,E), and |E| = k, then $$\sum_{v \in V} \deg(v) = 2k$$



G = (V,E), where
V = {a,b,c,d,e,f}
E = {{a,b},{a,c},{b,c},{b,d},{b,e},{c,f},{e,f}}

deg(a)=2, deg(b)=4, deg(c)=3, deg(d)=1, deg(e)=2, deg(f)=2
2+4+3+1+2+2=14 = 2*7
There are 7 edges

The Handshaking Lemma

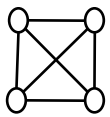
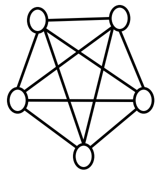
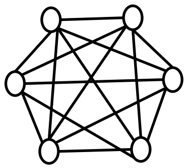
The number of vertices with odd degree is even.

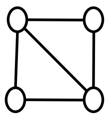
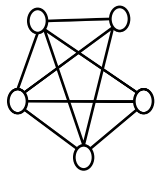
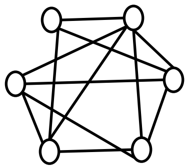
Proof

The sum of the degrees of all vertices is 2k, which is even.
Partition the vertices into two groups – the vertices with even degree, and the vertices with odd degree.
The sum of the degrees of even-degree vertices must be even, say 2x.
Let the sum of the degrees of odd-degree vertices be y.
Then y+2x=2k. So y=2k-2x = 2(k-x), which is even.
But all the numbers I added up to get y were odd numbers. So there must have been an even number of them.

Complete graphs

A complete graph is one where every vertex is adjacent to every other vertex.

Complete graphs:   

Not complete graphs:   

Next lecture ...

paths, subgraphs and connectivity