COMP20003
Algorithms and Data Structures
Introduction to Graphs

Kris Ehinger
Department of Computing and Information Systems
University of Melbourne
Semester 2

Application: Image synthesis

A. A. Efros & W. T Freeman. Image quilting for texture synthesis. SIGGRAPH 2002

The key to solving many algorithmic problems is to think of them in terms of graphs. Graph theory provides a language for talking about the properties of relationships, and it is amazing how often messy applied problems have a simple description and solution in terms of classical graph properties.

Designing truly novel graph algorithms is a very difficult task. The key to using graph algorithms effectively in applications lies in correctly modeling your problem so you can take advantage of existing algorithms. Becoming familiar with many different algorithmic graph *problems* is more important than understanding the details of particular graph algorithms, particularly since Part II of this book will point you to an implementation as soon as you know the name of your problem.

Here we present basic data structures and traversal operations for graphs, which will enable you to cobble together solutions for basic graph problems. Chapter 6 will present more advanced graph algorithms that find minimum spanning trees, shortest paths, and network flows, but we stress the primary importance of correctly modeling your problem. Time spent browsing through the catalog now will leave you better informed of your options when a real job arises.

S. S. Skiena. The Algorithm Design Manual.

ing s, particularly since Part II of with manyaph algorithms, particularly since Part II of this book von as you know the name of yeanding thenentation as soon as you know the name of your obleires and traversal operations foris book wisic data structures and traversal operations for gra s, whiutions for basic graph probler problem.bble together solutions for basic graph problems. aptgorithms that find minimum saphs, whivanced graph algorithms that find minimum span ir pran take advantage of existing algorithms. Becoming familiar with mams for basic graph with algorithmic graph *problems* is more important than understanding thms that find minin undiof particular graph algorithms, particularly since Part II of this book wiress the primary im boot to an implementation as soon as you know the name of your problem. ing through the ca probwe present basic data structures and traversal operations for graphs, whiof this boniliar with phs ble you to cobble together solutions for basic graph problems. Chapter your prohnderstand Chapnt more advanced graph algorithms that find minimum spanning treeor graphs, of this bo ving aths, and network flows, but we stress the primary important tof correctbms. Chapvour pro theory provides ions lies ins lies in correctly mophs, whichrly since Part II of spanning for graphs it is amazing lg algorithalgorithms. Becoming Chapter know the name of yance of coblems. Ch ution in terms is more imore important thaning treeaversal operations for, now will spannin novel graph algorithms is a very difficult task. The key r basic graph problems. Chapteble toge ffectively in applications lies in correctly modeling your that find minimum spanning tvanced g dvantage of existing algorithms. Becoming familiar wi the primary importance of correctwork flo ic graph *problems* is more important than understan through the catalog now will lem. Time r graph algorithms, particularly since Part II of this bs. plementation as soon as you know the name of your prask. The key to usingey to usiney to usin basic data structures and traversal operations for graphsodeling your problenur problemr proble cobble together solutions for basic graph problems. Clg familiar with mair with manyith mai advanced graph algorithms that find minimum spanninn understanding thtanding thanding t network flows, but we stress the primary importance of rt II of this book wills book wil book w

https://commons.wikimedia.org/wiki/File:Franklin%27s_Gull_Flock_(3005531251O).jpg
COMP 90016 Computational Genomics 1-6

https://commons.wikimedia.org/wiki/File:Franklin%27s_Gull_Flock_(3005531251O).jpg
COMP 90016 Computational Genomics 1-8

Graph definition

Graph:

- a representation of a set of objects
- and the relationships between them

Graph definition

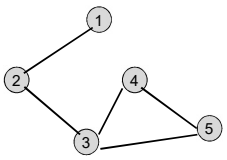
Graph G = {V,E} Set of:

- Vertices V: can contain information
- Edges E (links between vertices): can have direction and/or weight

Compared to trees and linked lists:

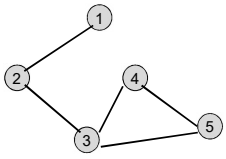
- vertices = nodes
- edges = links

Undirected graph



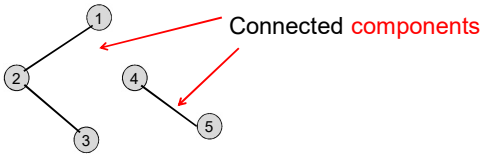
Edges have no direction specified

Connected Undirected graph

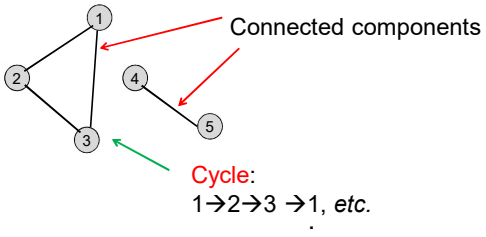


Every pair of vertices is connected (possibly indirectly)

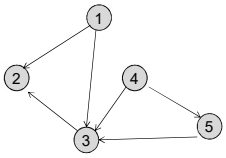
Unconnected Undirected graph



Unconnected Undirected graph with cycle

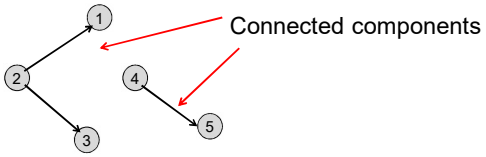


Directed graph



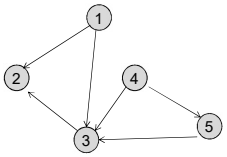
- Edge direction is specified
- Links are not symmetrical

Acyclic, unconnected directed graph



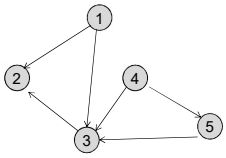
.

Directed graph



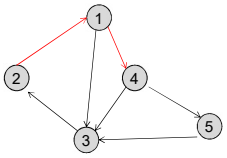
Reachability:
Can you get from Vertex 1 to Vertex 5?

Weakly connected Directed graph



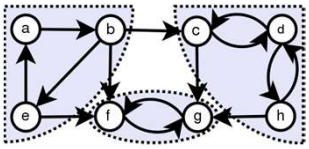
Replace all directed edges with undirected edges, to obtain a connected (undirected) graph

Strongly connected Directed graph



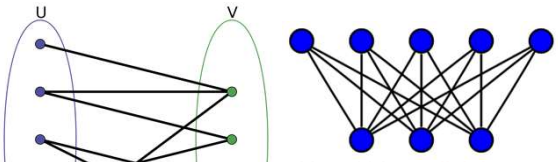
If every vertex is reachable from every other vertex

Strongly connected Components in a Digraph



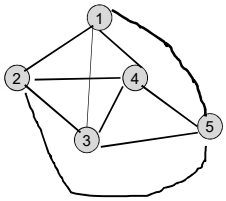
If every vertex is reachable from every other vertex within the same component

Bipartite Graph



- U and V are disjoint sets of vertexes
- Every vertex in U connects to a vertex in V, and viceversa

Complete graph



V(V-1)/2 edges

Trees and Graphs

Tree, undirected graph that is:

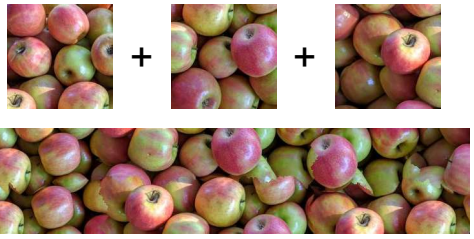
- Connected
- Acyclic

- *n.b.* Any two vertices are connected by exactly one simple path
- *n.b.* All vertices are connected

Demos…

Application: image synthesis

- How does it use graphs?



Application: image synthesis

- To find the best splice, represent overlapping images as a graph:
  - Edge weight = difference in colour
  - Problem = find minimum path