Graph Theory
An Introduction

https://github.com/mohawk2/presentations

Introduction

- Things
- Connections between things
- Not charts

Introduction

- Why
- What
- How (in computing)
- Algorithms (why bought ticket)
- Resources
- Exercises

Why

- Network routing (paths)
- Social media
- Machine learning
- Genetics
- Software build systems such as make (example of hypergraph)

What

- Directed
- Undirected
- Hypergraphs (both above)

UNEXPECTED MATHS

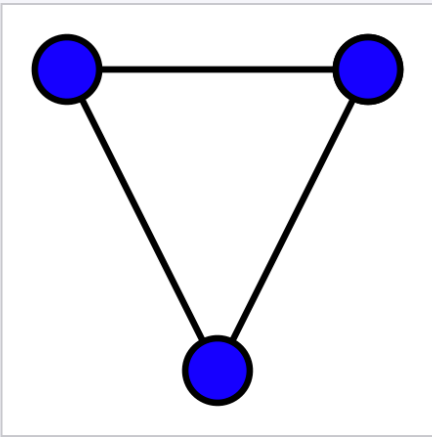
Graph [edit]

In one restricted but very common sense of the term,[1][2] a **graph** is an ordered pair $G = (V, E)$ comprising:

- V, a set of **vertices** (also called **nodes** or **points**);
- $E \subseteq \{\{x, y\} \mid x, y \in V \text{ and } x \neq y\}$, a set of **edges** (also called **links** or **lines**), which are unordered pairs of vertices (that is, an edge is associated with two distinct vertices).

To avoid ambiguity, this type of object may be called precisely an **undirected simple graph**.

In the edge $\{x, y\}$, the vertices x and y are called the **endpoints** of the edge. The edge is said to **join** x and y and to be **incident** on x and on y. A vertex may exist in a graph and not belong to an edge. **Multiple edges**, not allowed under the definition above, are two or more edges that join the same two vertices.



A graph with three vertices and three edges.

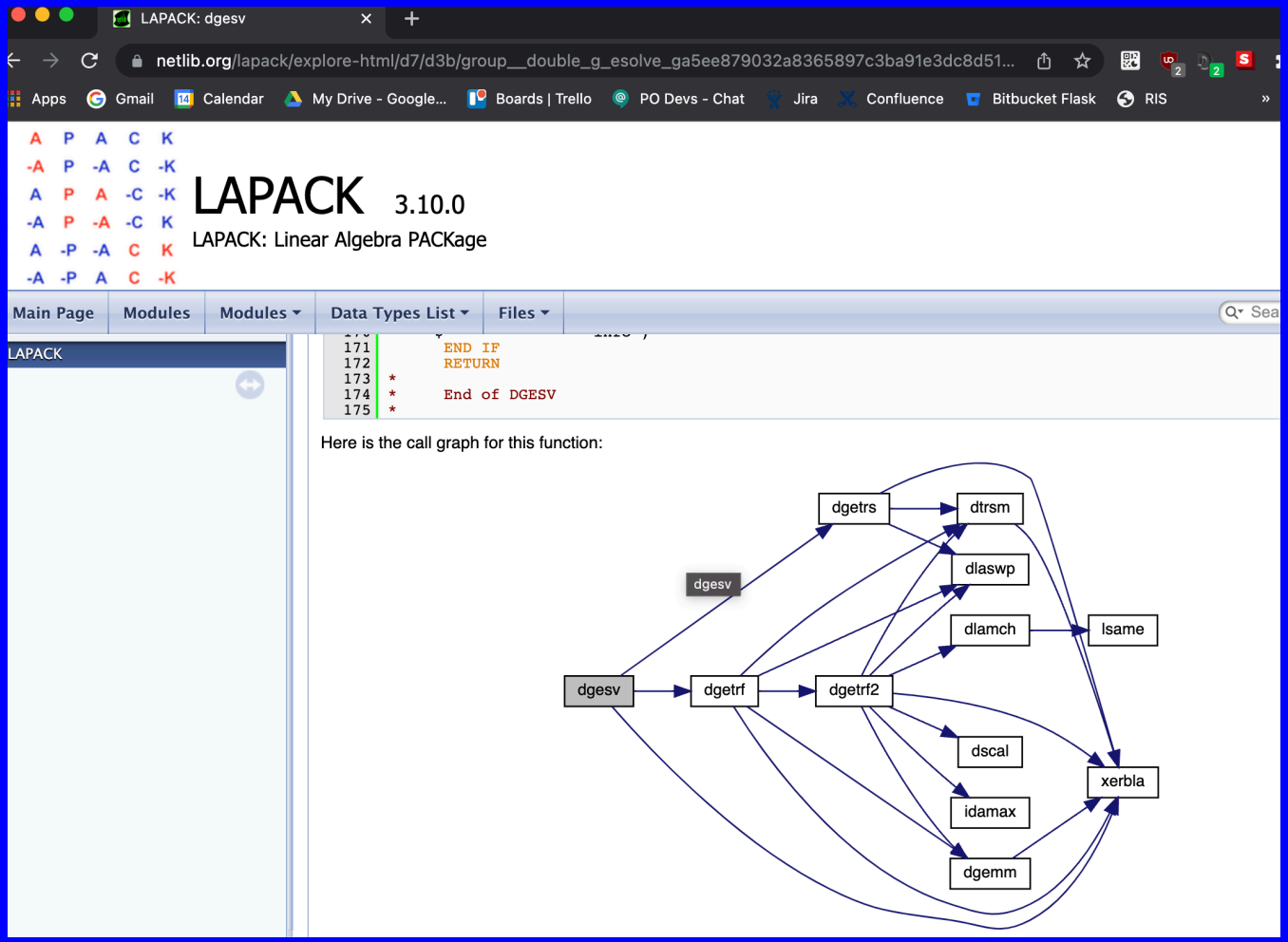
In one more general sense of the term allowing multiple edges,[3][4] a **graph** is an ordered triple $G = (V, E, \phi)$ comprising:

- V, a set of **vertices** (also called **nodes** or **points**);
- E, a set of **edges** (also called **links** or **lines**);
- $\phi : E \to \{\{x, y\} \mid x, y \in V \text{ and } x \neq y\}$, an **incidence function** mapping every edge to an unordered pair of vertices (that is, an edge is associated with two distinct vertices).

To avoid ambiguity, this type of object may be called precisely an **undirected multigraph**.

https://en.wikipedia.org/wiki/Graph_theory

Directed

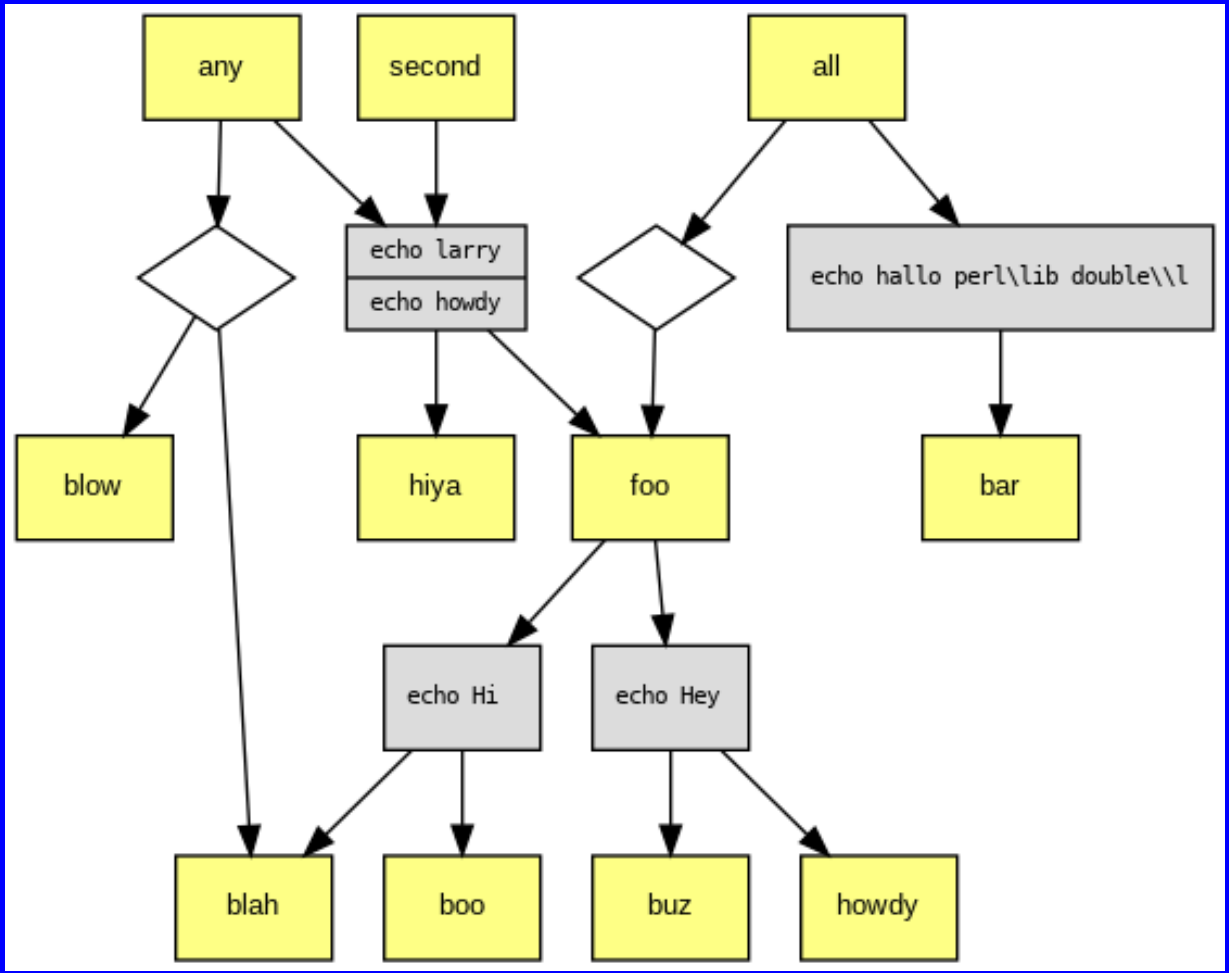


https://www.netlib.org/lapack/explore-html/d7/d3b/group__double_g_esolve_ga5ee879032a8365897c3ba91e3dc8d512.html

Undirected


Angel
Old Street
Hoxton
Farringdon
Liverpool Street
Shoreditch High Street
Barbican
Moorgate
Chancery Lane
St. Paul's
Bank
Aldgate East
Aldgate
Cannon Street
Monument
Tower Hill
Tower Gateway
River Thames
Rotherhithe
London
Bermondsey
Canada

Directed hypergraph



https://gitlab.com/graphviz/graphviz/-/issues/1911

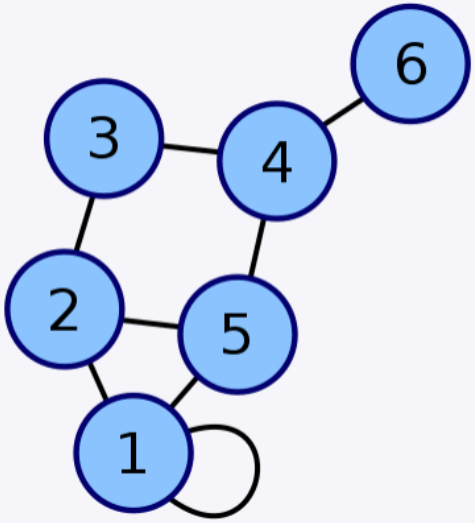
Graph classification

- Directed? Cyclic? Multi? Hyper?
- Social media
- Transport networks
- Software build systems

How

- Matrix: adjacency (undirected = symmetrical)
- Matrix, sparse/hypergraph: incidence
- Linked lists of V, E
- Dictionary of E

Adjacency matrix

Labeled graph	Adjacency matrix
	$\begin{pmatrix} 2 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$ Coordinates are 1–6.

https://en.wikipedia.org/wiki/Adjacency_matrix

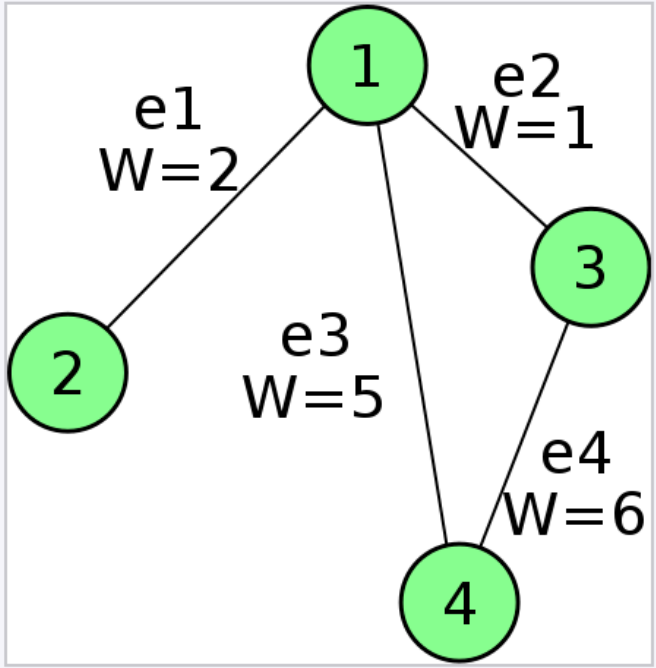
Incidence matrix

Weighted graphs [edit]

A weighted graph can be represented using the weight of the edge in place of a 1. For example, the incidence matrix of the graph to the right is:

	e_1	e_2	e_3	e_4
1	2	1	5	0
2	2	0	0	0
3	0	1	0	6
4	0	0	5	6

$$= \begin{bmatrix} 2 & 1 & 5 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 6 \\ 0 & 0 & 5 & 6 \end{bmatrix}.$$



A weighted undirected graph

https://en.wikipedia.org/wiki/Incidence_matrix

Graph as dictionary (pseudocode)

```
class Graph:
  nodes = []
  edges = {}
  def add_node(v):
    nodes.push(v)
  def add_edge(v1, v2):
    if not v1 in node:
      throw(“invalid node {v1}”)
    if v2 in nodes:
      throw(“invalid node {v2}”)
    edges[v1][v2] = 1
```

Algorithms

- Floyd-Warshall
- Topological sorting

Floyd-Warshall

```
let dist be a |V| × |V| array of minimum distances
          initialized to ∞ (infinity)
for each edge (u, v) do
    dist[u][v] ← w(u, v)  // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] ← 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] ← dist[i][k] + dist[k][j]
            end if
```

https://en.wikipedia.org/wiki/Floyd%E2%80%93Warshall_algorithm
https://metacpan.org/pod/PDL::MATLAB#COMPARISON:-FLOYD-WARSHALL-ALGORITHM

Topological sorting

```
L ← Empty list that will contain the sorted elements
S ← Set of all nodes with no incoming edge

while S is not empty do
    remove a node n from S
    add n to L
    for each node m with an edge e from n to m do
        remove edge e from the graph
        if m has no other incoming edges then
            insert m into S

if graph has edges then
    return error   (graph has at least one cycle)
else
    return L   (a topologically sorted order)
```

https://en.wikipedia.org/wiki/Topological_sorting

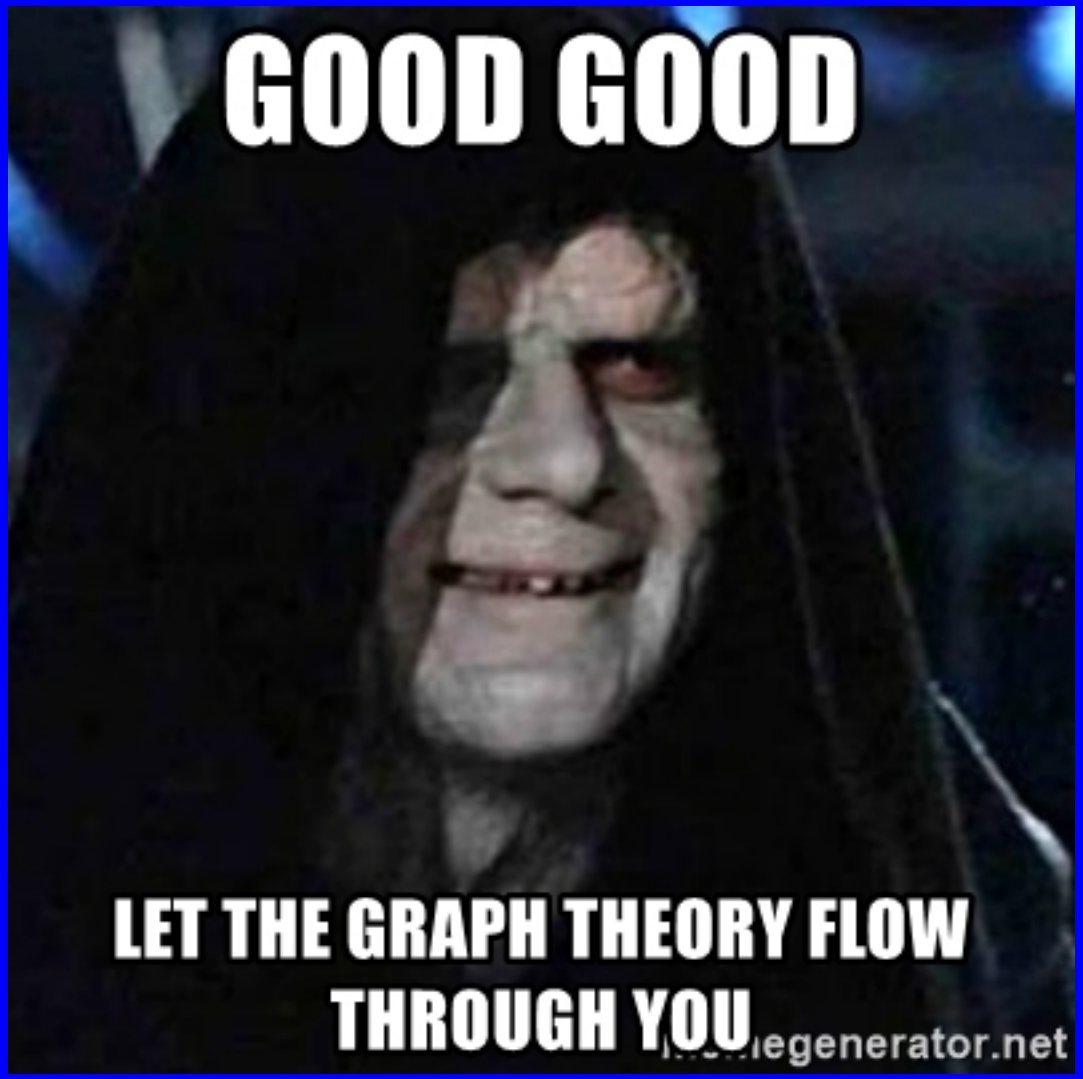
Summary

- Graphs: things and connections between things
- What, why, how
- Algorithms

Resources

- https://en.wikipedia.org/wiki/Graph_theory
- https://en.wikipedia.org/wiki/Hypergraph
- https://graphviz-perl.github.io/
- https://github.com/mohawk2/presentations

Questions?


GOOD GOOD
LET THE GRAPH THEORY FLOW
THROUGH YOU
egenerator.net

Exercises

- Floyd-Warshall
- Topological sorting