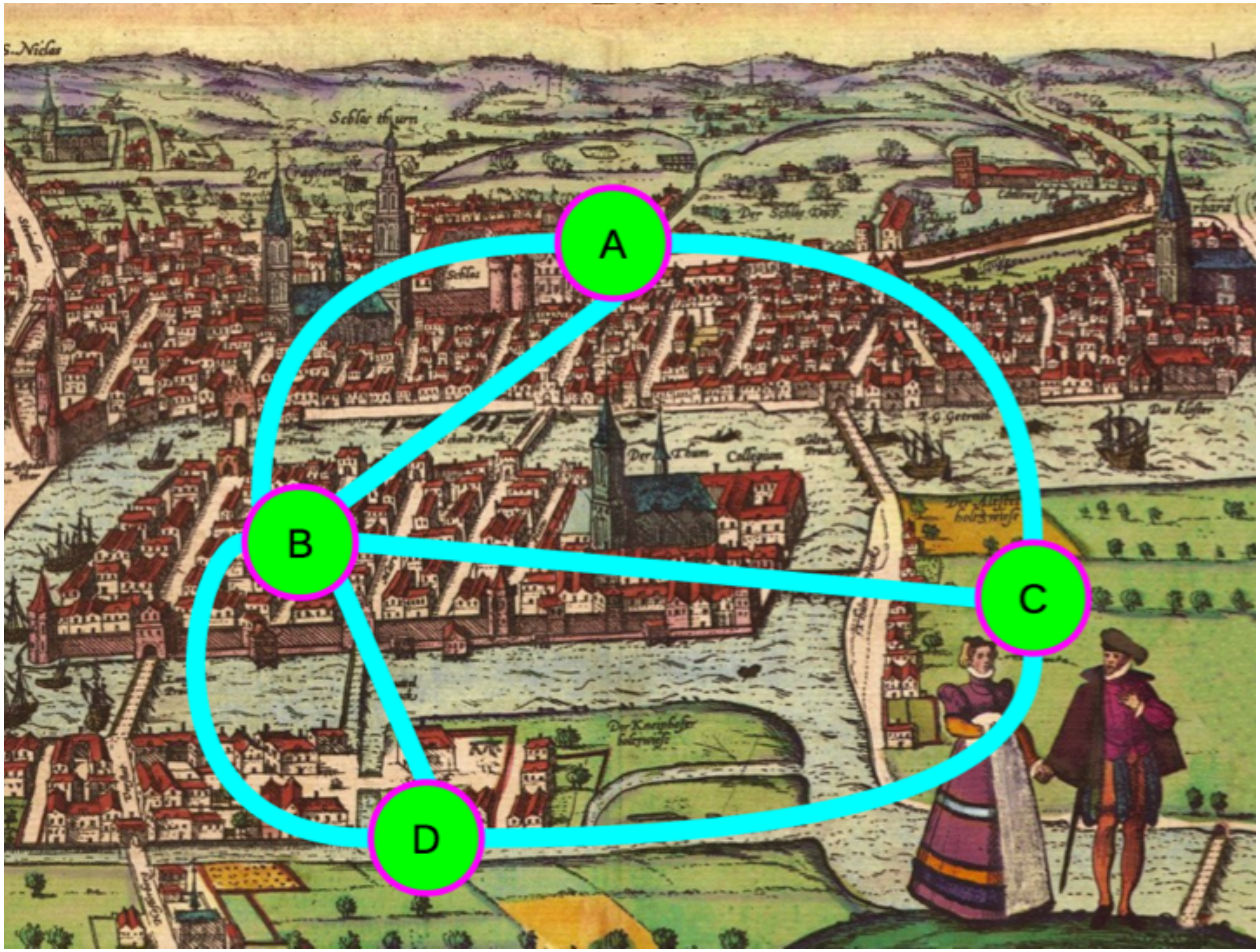


CSCI 241

Lecture 18

Intro to Graphs; Graph Representation; Graph Traversals

Announcements

- A2 grades are out.
  - Please pull the grading branch to see your feedback.
  - As usual, you can resubmit once for half unit test credit back.
- A3 is due Wednesday
  - My solution has 111 more lines than the skeleton.
  - The concepts-to-code ratio is high.
- A4 out Wednesday, due the following Wednesday
  - We’ll cover the algorithm on Wednesday and Friday.

Happenings

Wednesday, 3/6 – Peer Lecture Series: VIM Workshop – 5 pm in CF 420

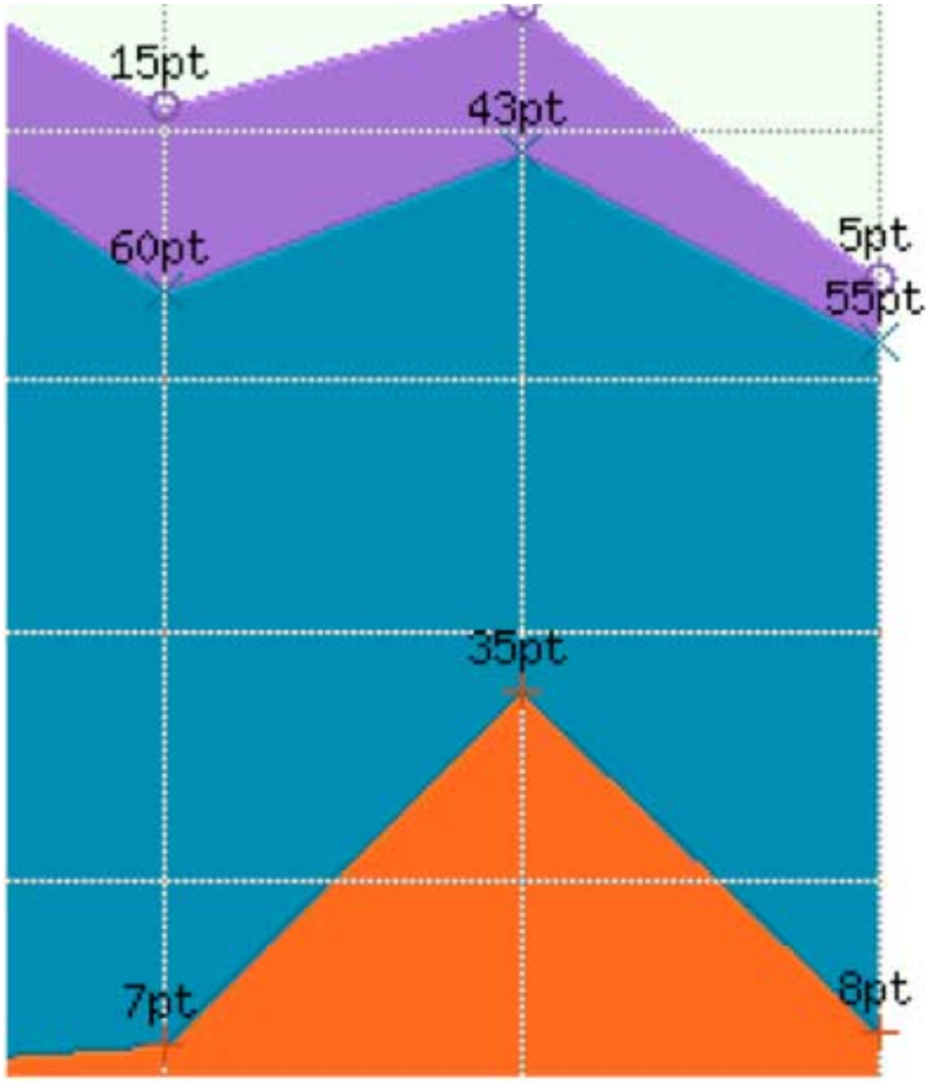
Wednesday, 3/6 – Cybersecurity Lecture Series: Cyber and Physical Security Standards in the Power Industry – 5 pm in CF 105

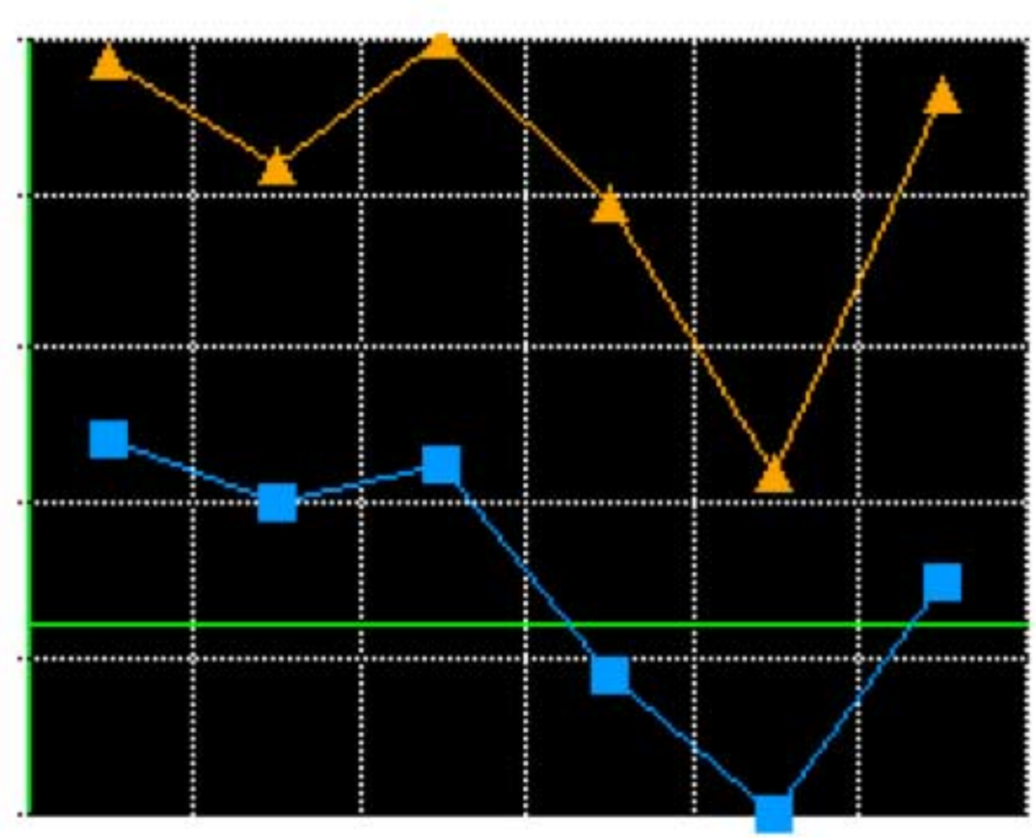
Thursday, 3/7 – Elevator Speech Workshop – 6 pm in CF 316

Friday, 3/8 – AWC Women in Industry Panel – 4 pm in AW 210

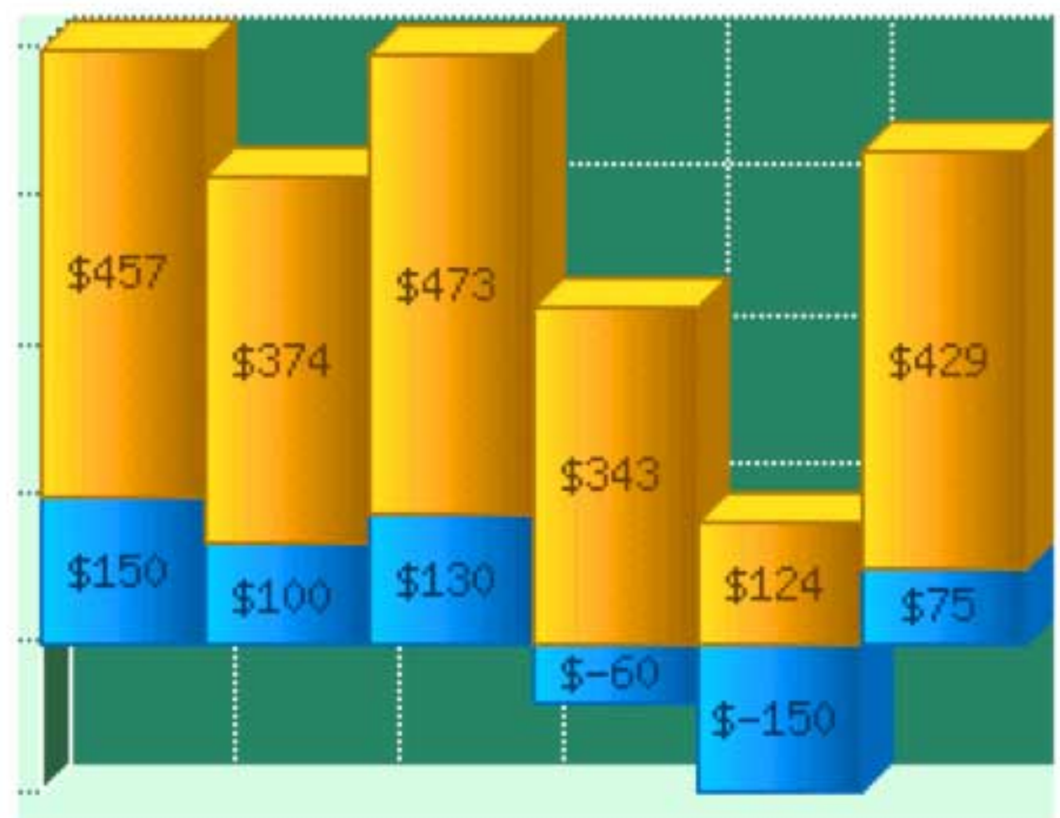
Goals

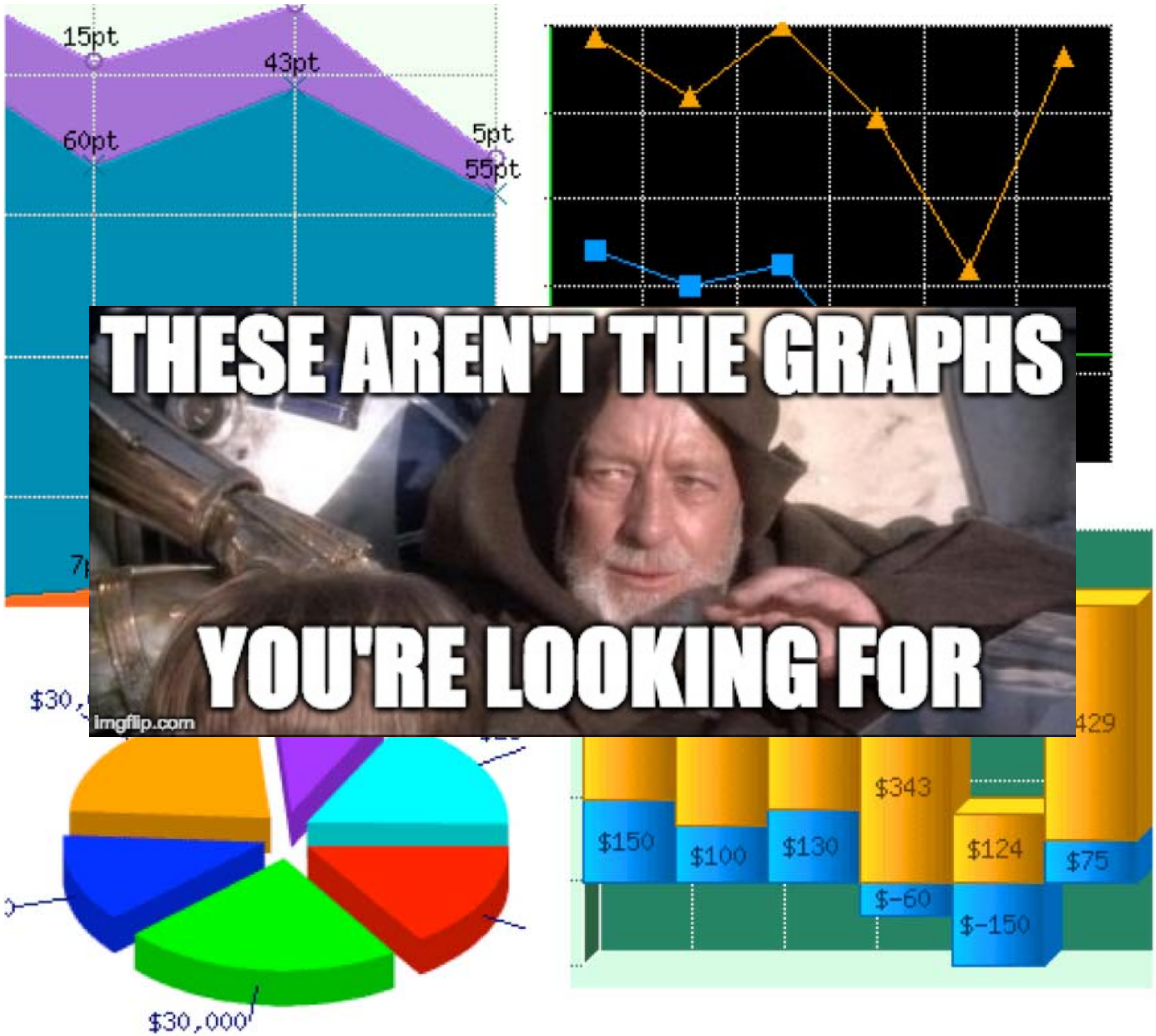
- Know the definition of a graph and basic associated terminology:
  - Node/vertex; edge/arc; directed, undirected; adjacent; (in/out-)degree; path; cycle;
- Understand how to represent a graph using:
  - adjacency list
  - adjacency matrix
- Be able to implement and analyze the runtime of simple graph operations on adjacency matrices and adjacency lists.
- Know how to implement breadth-first and depth-first graph traversals.


15pt
43pt
5pt
60pt
55pt
35pt
7pt
8pt

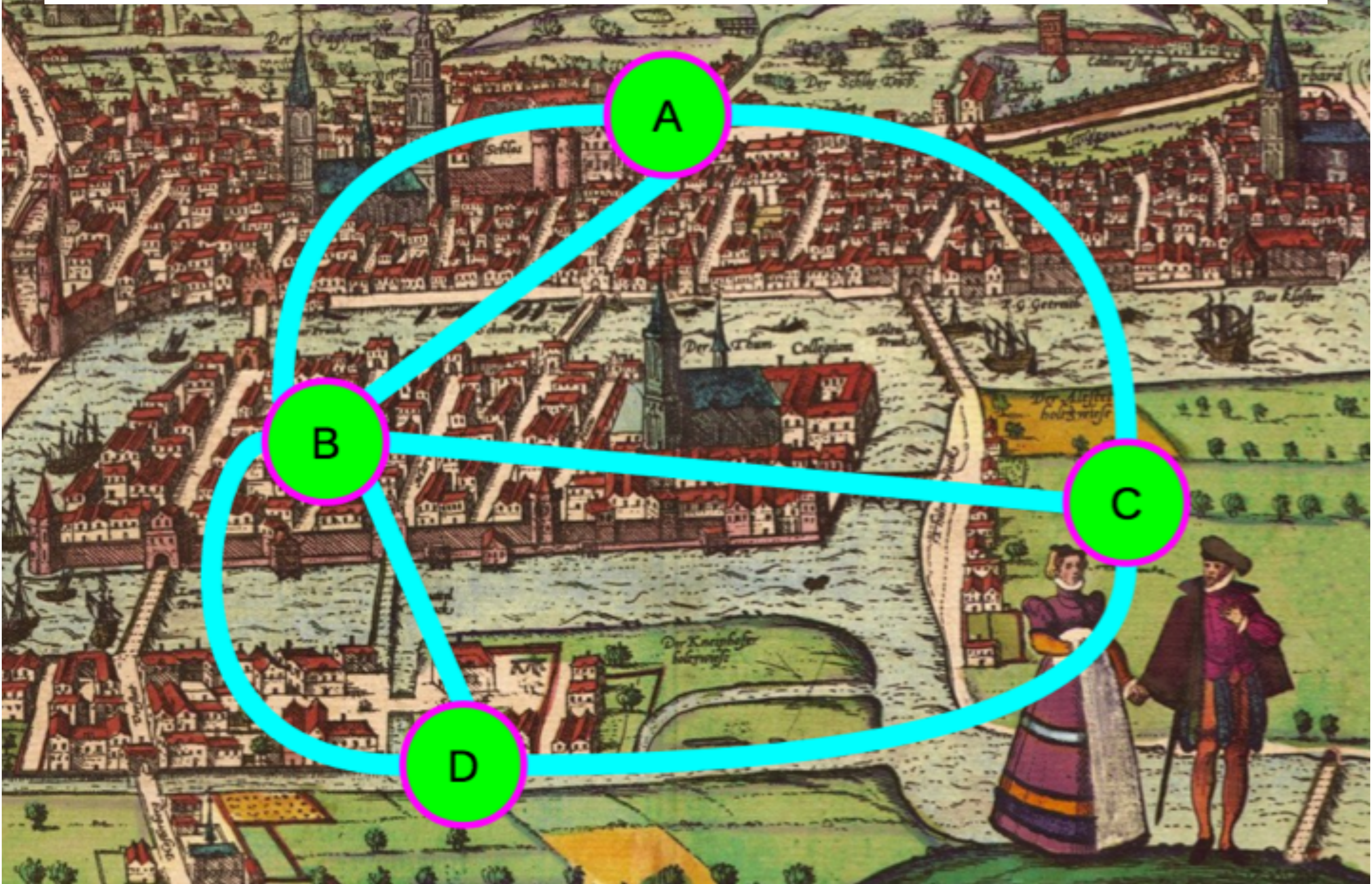



$12,000
$30,000
$23
$30,000

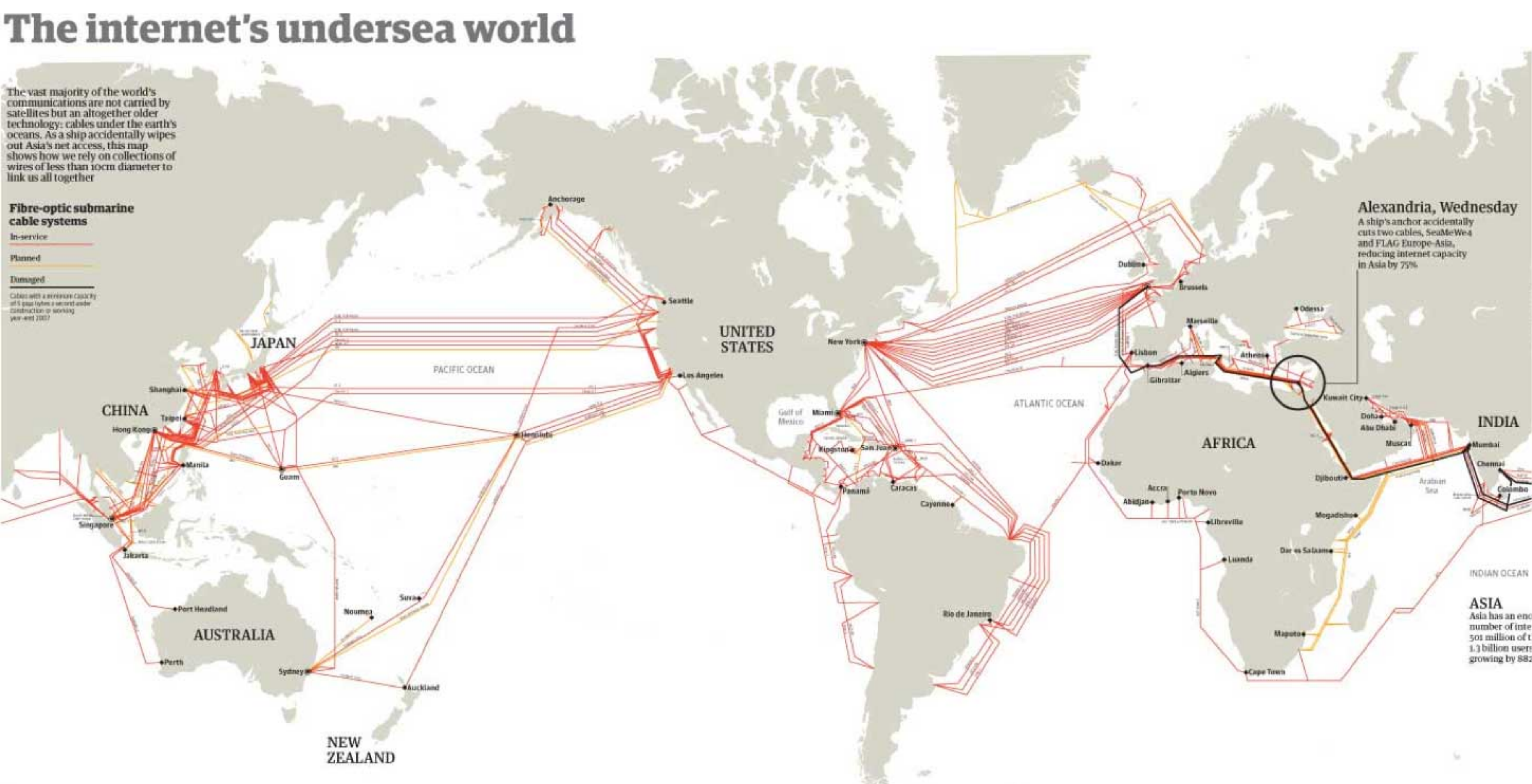

$457
$374
$473
$343
$124
$429
$150
$100
$130
$-60
$-150
$75


THESE AREN'T THE GRAPHS
YOU'RE LOOKING FOR
imgflip.com
15pt
43pt
60pt
5pt
55pt
$30,000
$150
$100
$130
$343
$124
$75
$-60
$-150
429

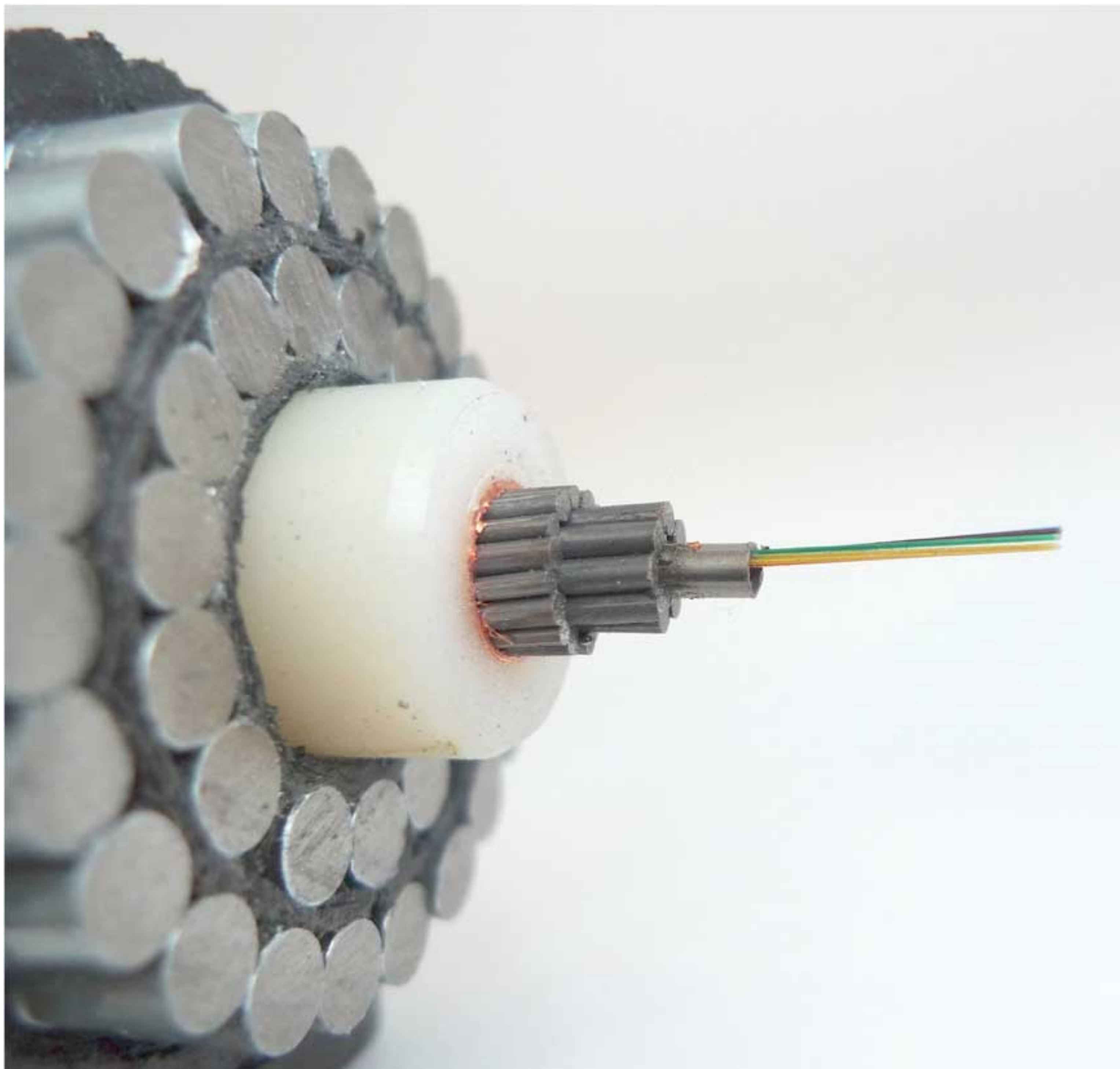
Graph: a bunch of points connected by lines.
The lines may have directions, or not.



This is a graph:

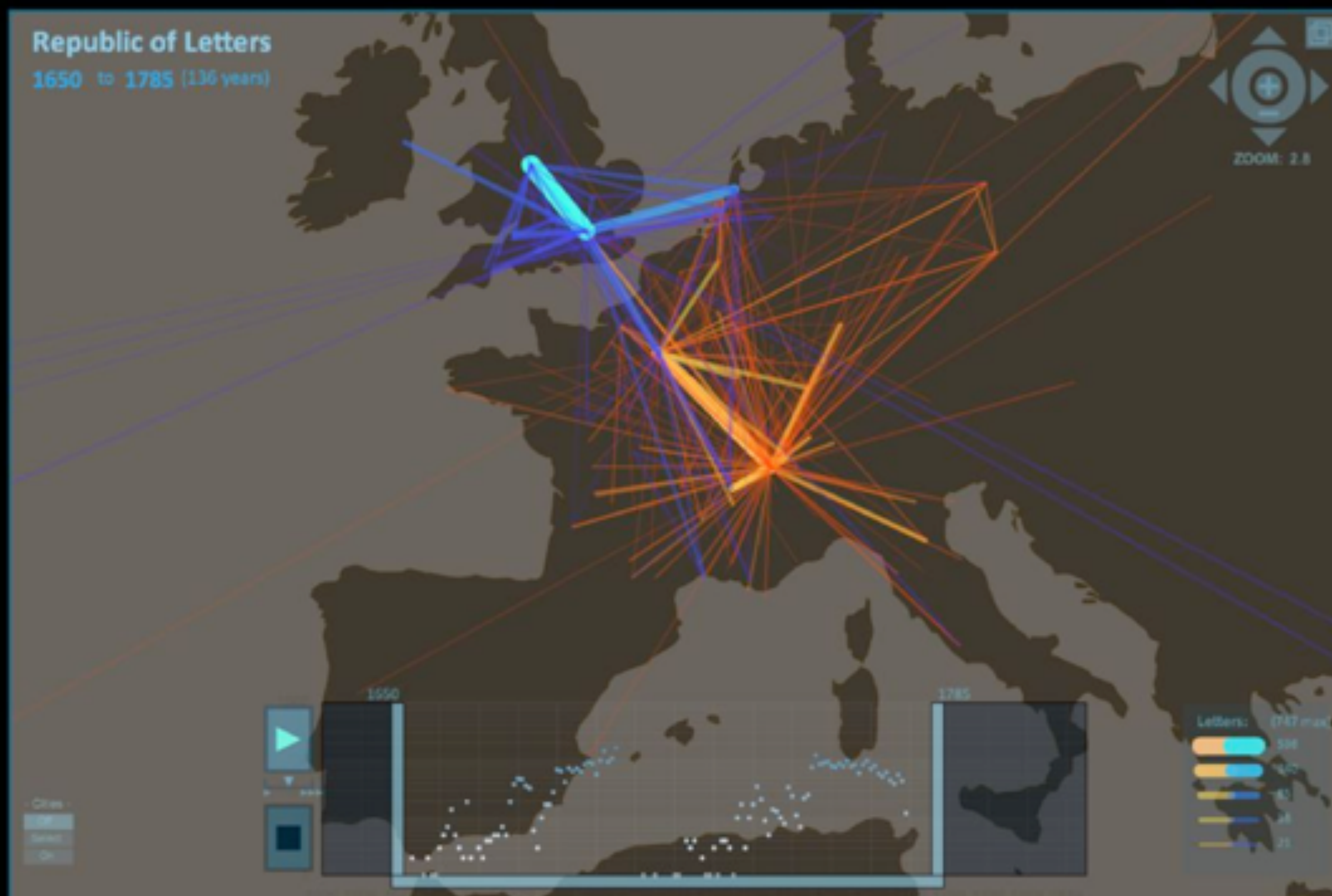


The edges are made of these:



Social Networks

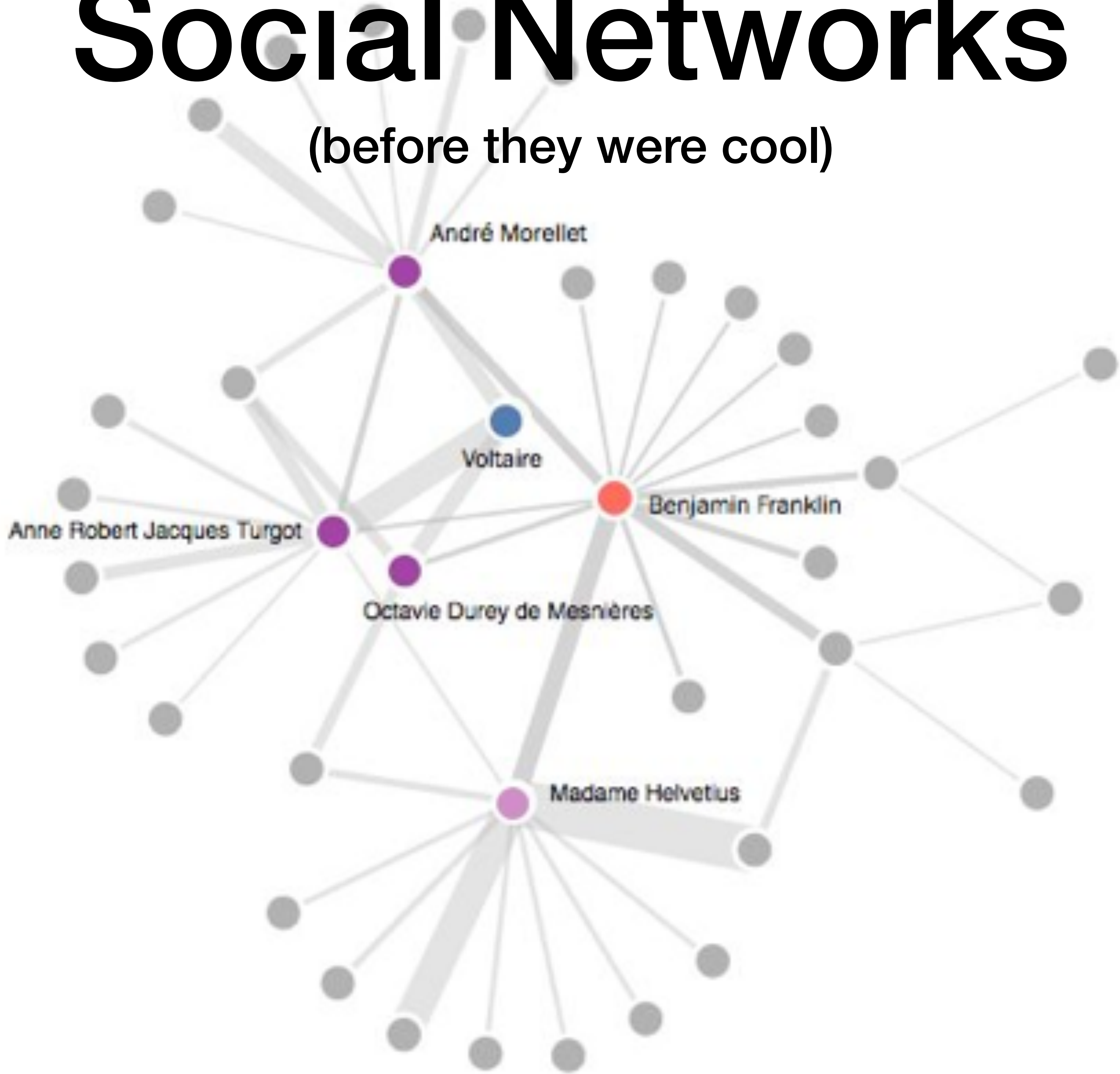
(before they were cool)

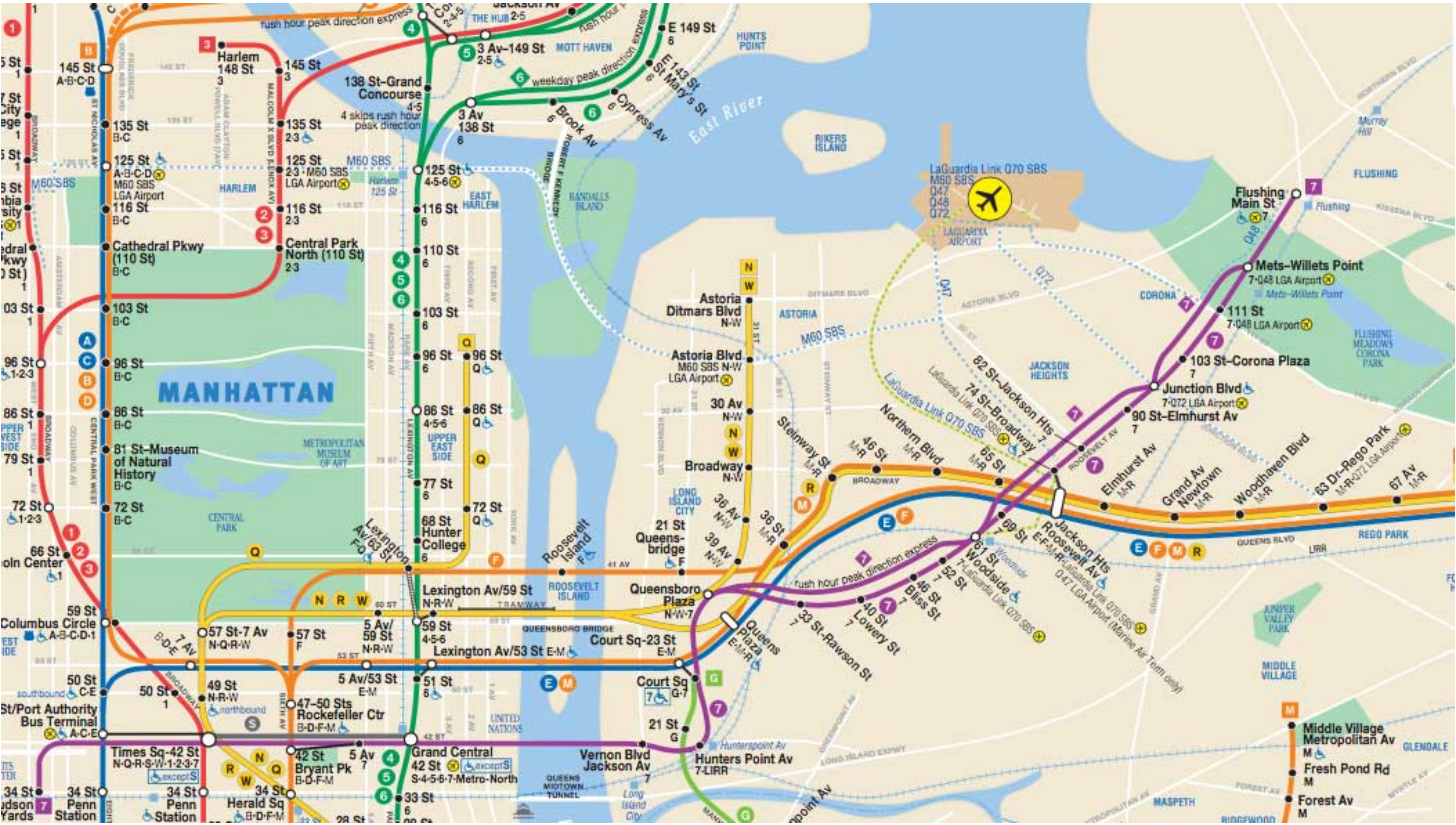


Locke's (blue) and Voltaire's (yellow) correspondence.
Only letters for which complete location information is available are shown.
Data courtesy the Electronic Enlightenment Project, University of Oxford.

Social Networks

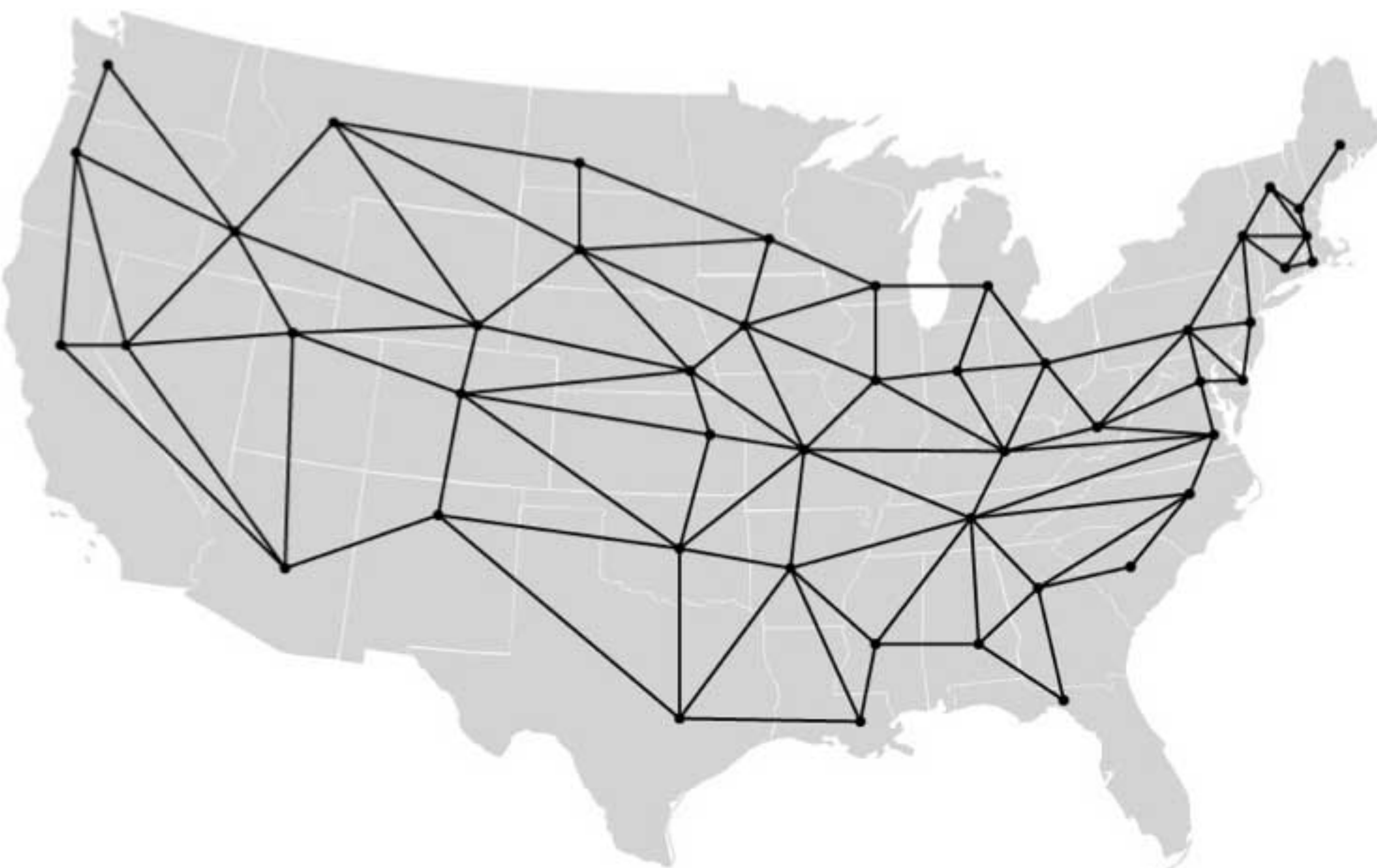
(before they were cool)

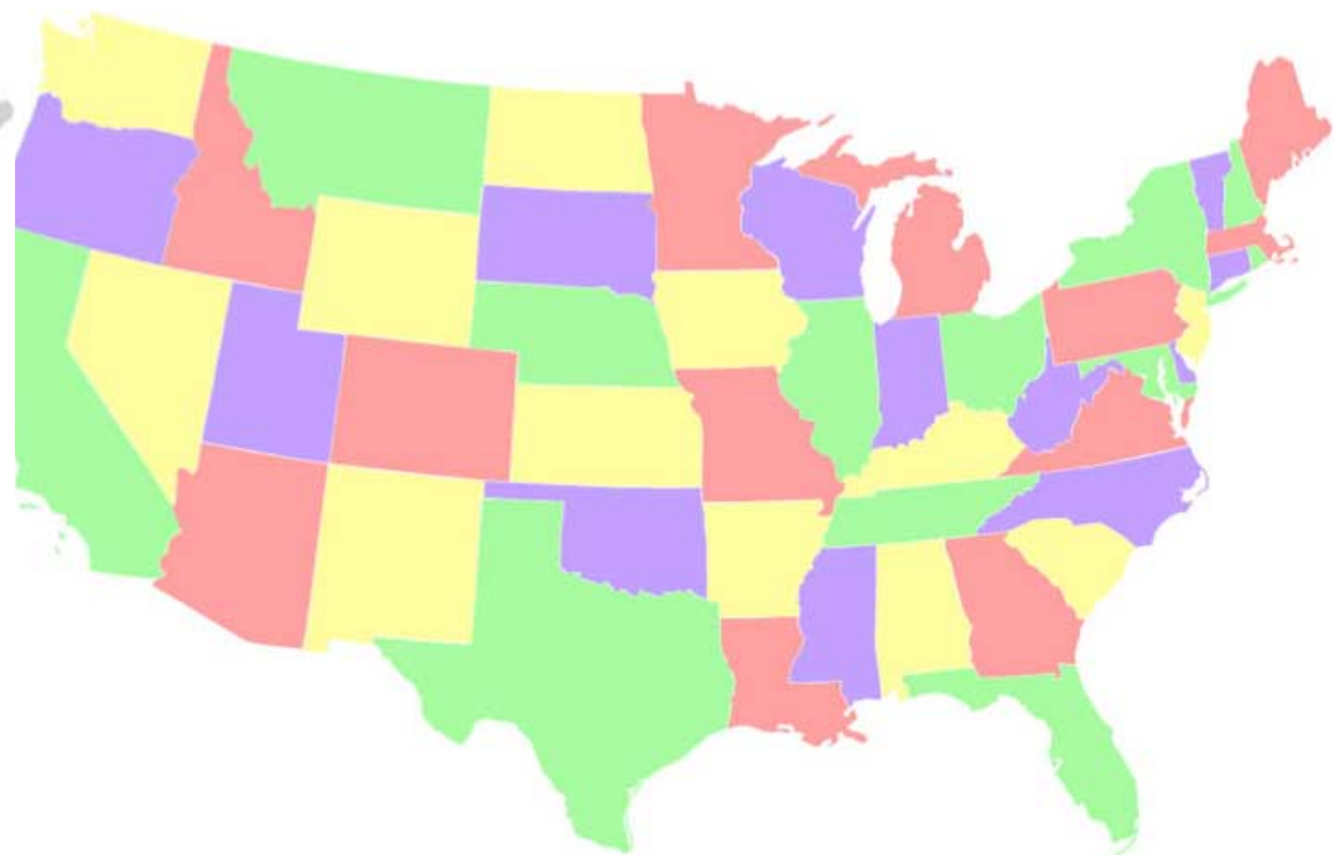



MANHATTAN
HARLEM
EAST HARLEM
UPPER EAST SIDE
CENTRAL PARK
METROPOLITAN MUSEUM OF ART
MOTT HAVEN
THE HUB
HUNTS POINT
East River
RIKERS ISLAND
RANDALLS ISLAND
ROOSEVELT ISLAND
UNITED NATIONS
LONG ISLAND CITY
ASTORIA
JACKSON HEIGHTS
CORONA
FLUSHING
FLUSHING MEADOWS CORONA PARK
LAGUARDIA AIRPORT
LaGuardia Link Q70 SBS
M60 SBS
JUNIPER VALLEY PARK
MIDDLE VILLAGE
MASPETH
REGO PARK
GLENDALE
QUEENS MIDTOWN TUNNEL
QUEENSBORO BRIDGE
TRAMWAY
145 St A·B·C·D
135 St B·C
125 St A·B·C·D
116 St B·C
Cathedral Pkwy (110 St) B·C
103 St B·C
96 St B·C
86 St B·C
81 St–Museum of Natural History B·C
72 St B·C
59 St Columbus Circle A·B·C·D·1
Harlem 148 St 3
145 St 3
135 St 2·3
125 St 2·3
116 St 2·3
Central Park North (110 St) 2·3
138 St-Grand Concourse 4·5
3 Av–149 St 2·5
E 149 St 6
E 143 St St Mary's St
Cypress Av 6
Brook Av 6
3 Av 138 St 6
125 St 4·5·6
116 St 6
110 St 6
103 St 6
96 St 6
86 St 4·5·6
77 St 6
68 St Hunter College 6
96 St Q
86 St Q
72 St Q
Lexington Av/63 St F·Q
Roosevelt Island F
Lexington Av/59 St N·R·W
59 St 4·5·6
5 Av/59 St N·R·W
57 St-7 Av N·Q·R·W
57 St F
7 Av B·D·E
Lexington Av/53 St E·M
5 Av/53 St E·M
51 St 6
50 St C·E
50 St 1
49 St N·R·W
47–50 Sts Rockefeller Ctr B·D·F·M
Port Authority Bus Terminal A·C·E
Times Sq-42 St N·Q·R·S·W·1·2·3·7
42 St Bryant Pk B·D·F·M
5 Av 7
Grand Central 42 St S·4·5·6·7·Metro-North
33 St 6
34 St Herald Sq B·D·F·M
34 St Penn Station
34 St Hudson Yards 7
Astoria Ditmars Blvd N·W
Astoria Blvd N·W
30 Av N·W
Broadway N·W
36 Av N·W
39 Av N·W
Queensboro Plaza N·W·7
21 St Queensbridge F
Court Sq-23 St E·M
Court Sq G·7
21 St G
Vernon Blvd Jackson Av 7
Hunters Point Av 7·LIRR
Queens Plaza E·M·R
36 St M·R
Steinway St M·R
46 St M·R
Northern Blvd M·R
65 St M·R
Jackson Hts Roosevelt Av E·F·M·R·7
74 St-Broadway 7
82 St-Jackson Hts 7
90 St-Elmhurst Av 7
Junction Blvd 7
103 St-Corona Plaza 7
111 St 7
Mets-Willets Point 7
Flushing Main St 7
69 St 7
61 St Woodside 7·LIRR
52 St 7
46 St Bliss St 7
40 St Lowery St 7
33 St-Rawson St 7
Elmhurst Av M·R
Grand Av Newtown M·R
Woodhaven Blvd M·R
63 Dr-Rego Park M·R
67 Av M·R
Middle Village Metropolitan Av M
Fresh Pond Rd M
Forest Av M
rush hour peak direction express
weekday peak direction express
4 skips rush hour peak direction

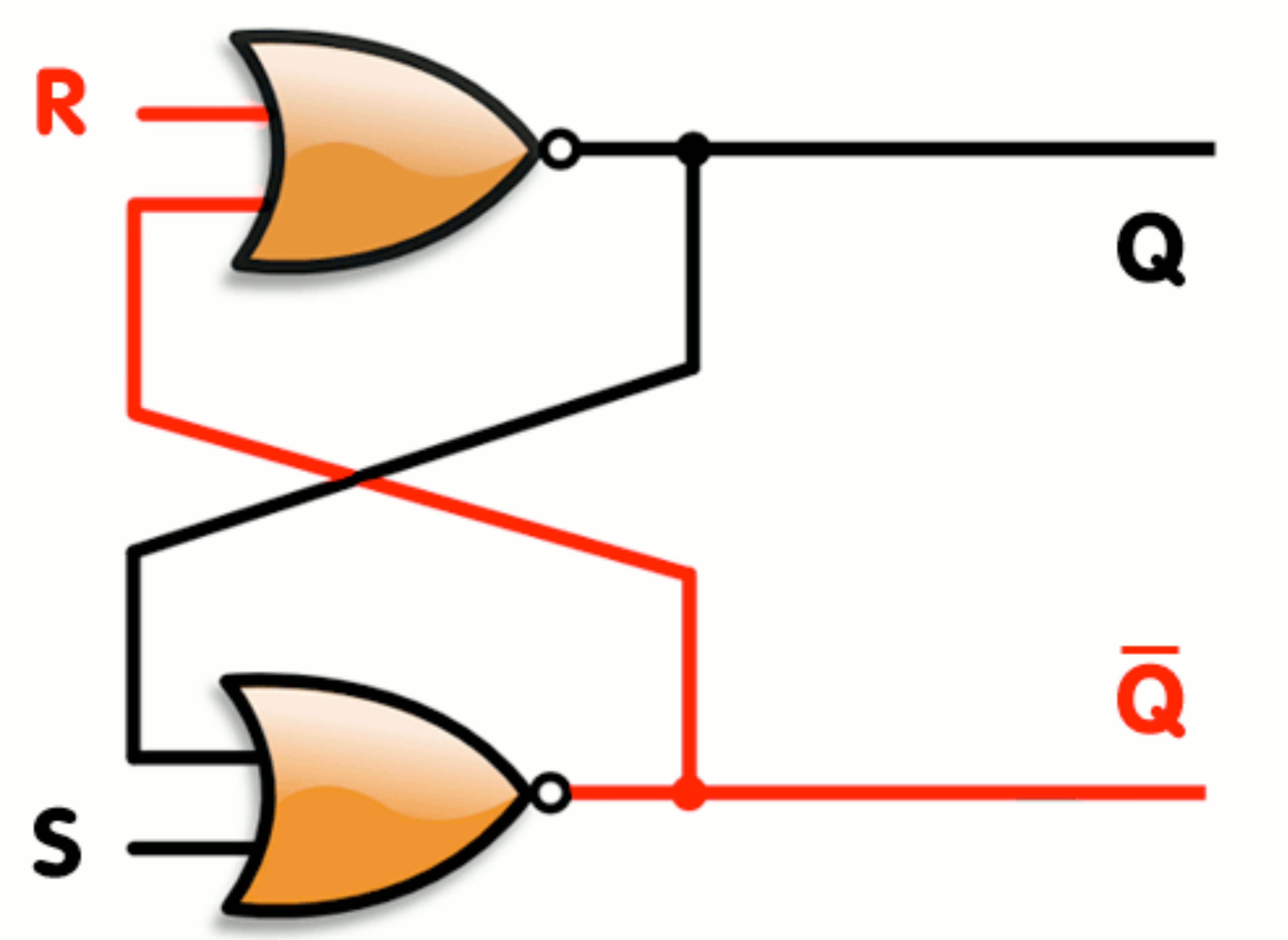
The USA as a graph:

- Neighboring states are connected by edges.

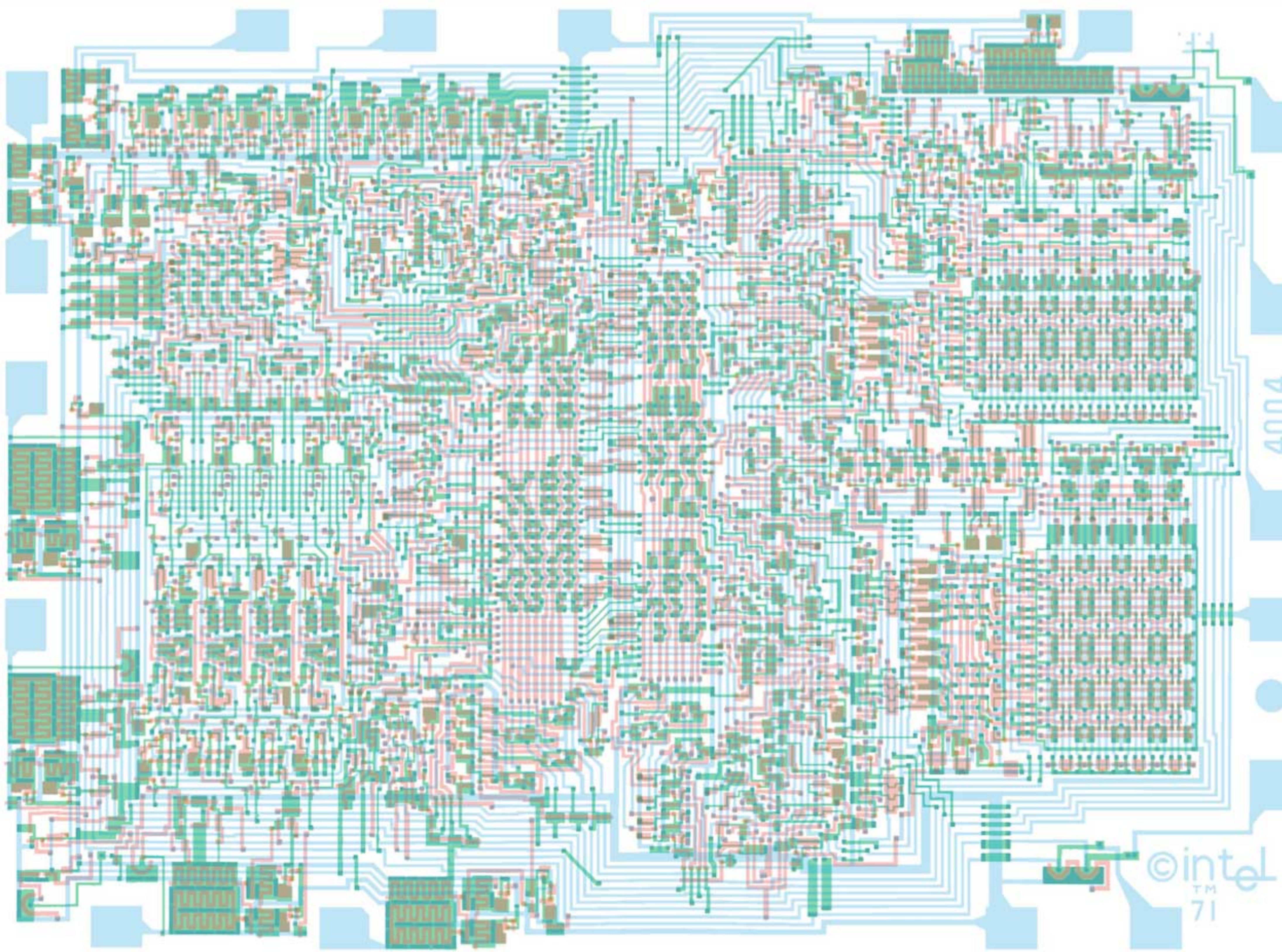




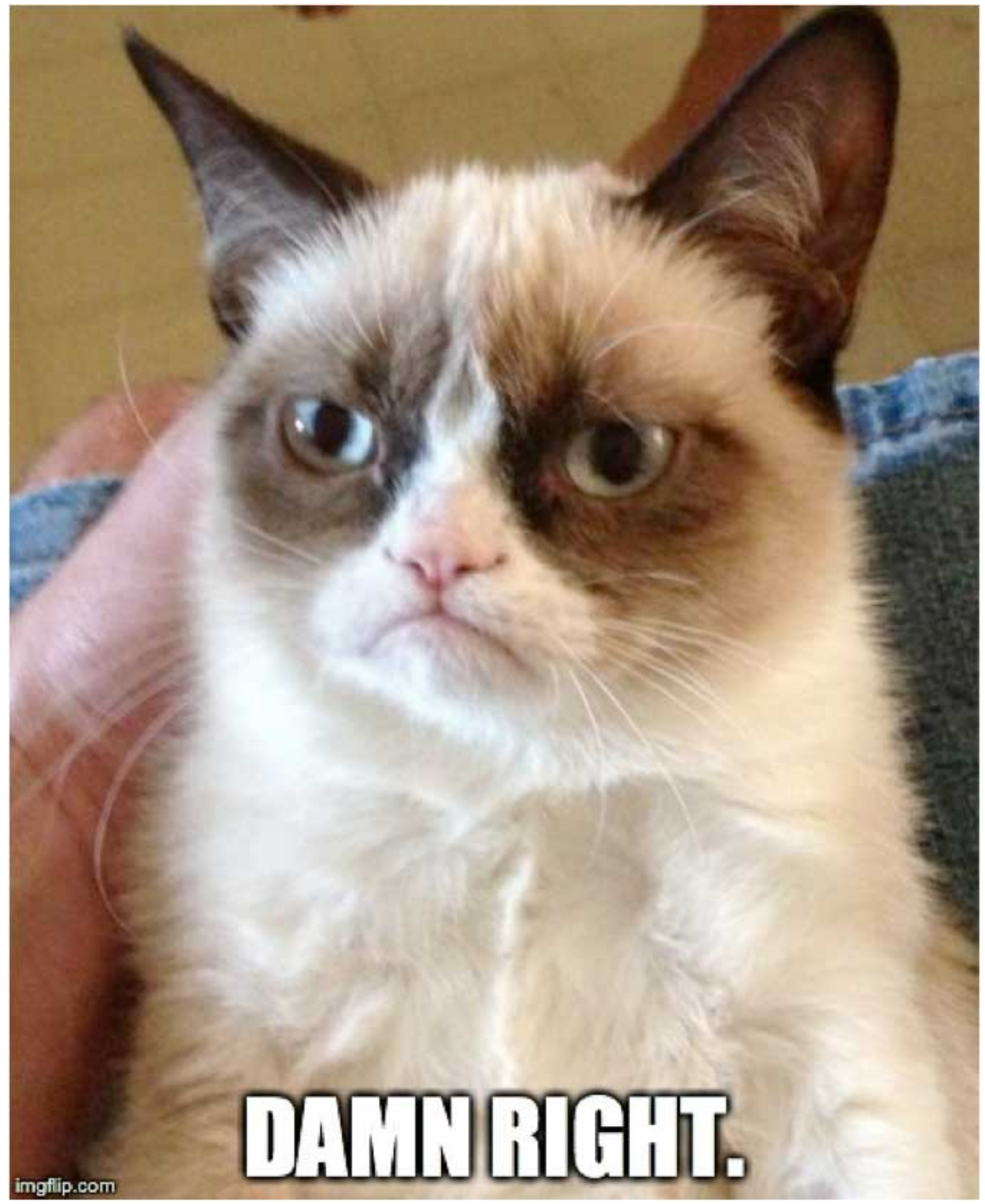
Electrical circuit



A bigger electrical circuit

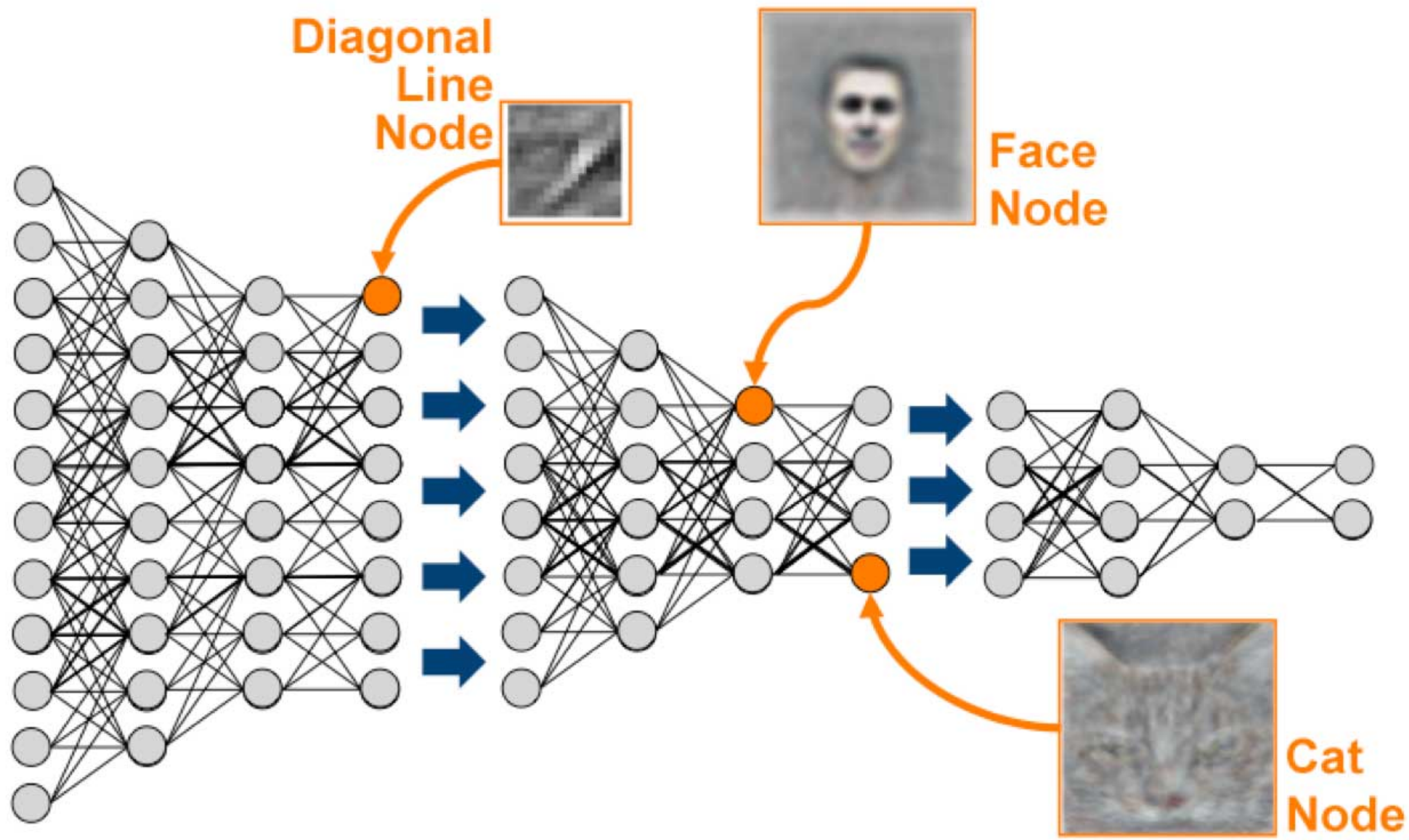


This is not a graph:



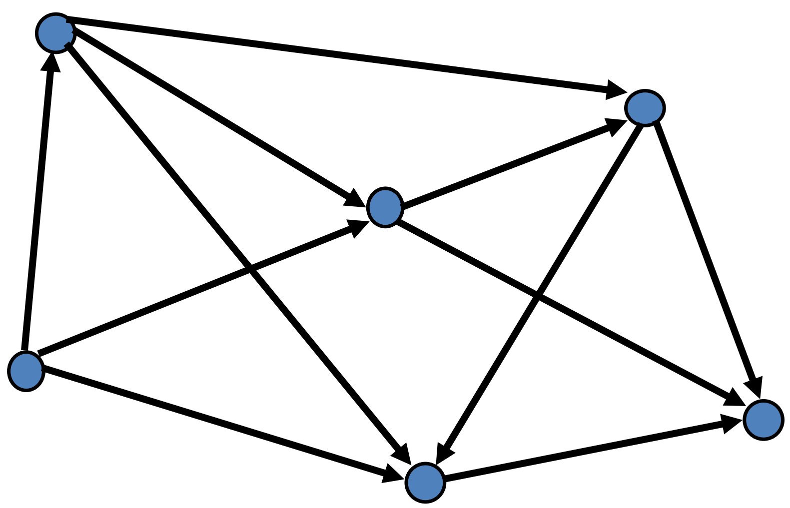
it is a cat.

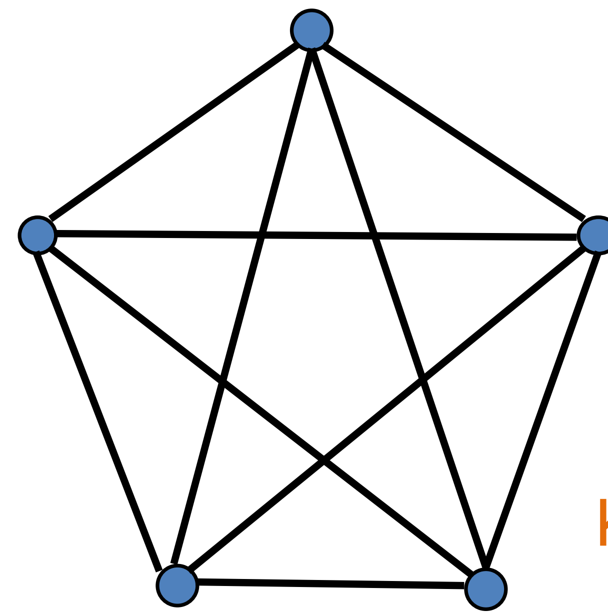
This is a graph



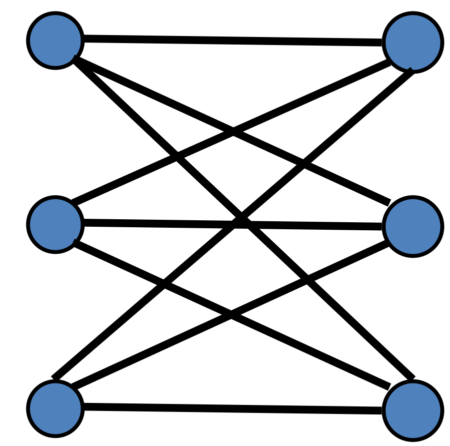
that can recognize cats.

Graphs: Abstract View

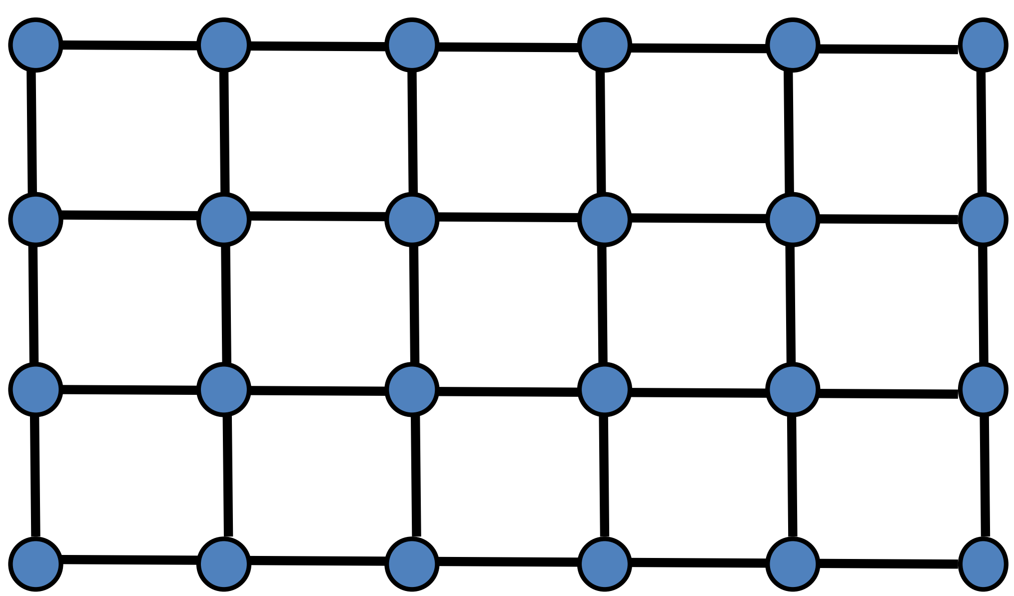


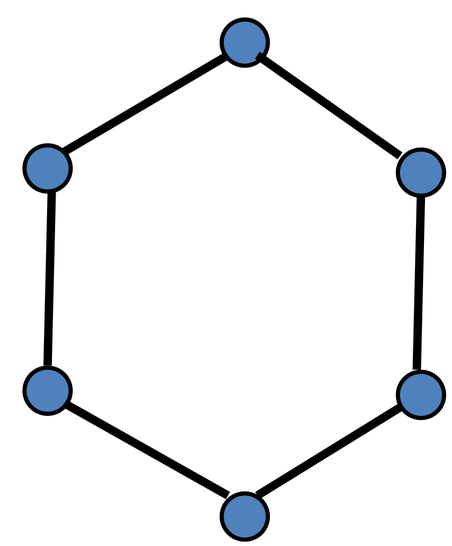


K_5



$K_{3,3}$





Graphs, Formally

- A directed graph (digraph) is a pair (V, E) where:
  - V is a (finite) set
  - E is a set of **ordered** pairs (u, v) where u, v are in V
  - Often (not always): u ≠ v (i.e. no edges from a vertex to itself)
- An element in V is called a vertex or node
- Elements in E are called edges or arcs
- |V| = size of V (traditionally called n)
- |E| = size of E (traditionally called m)

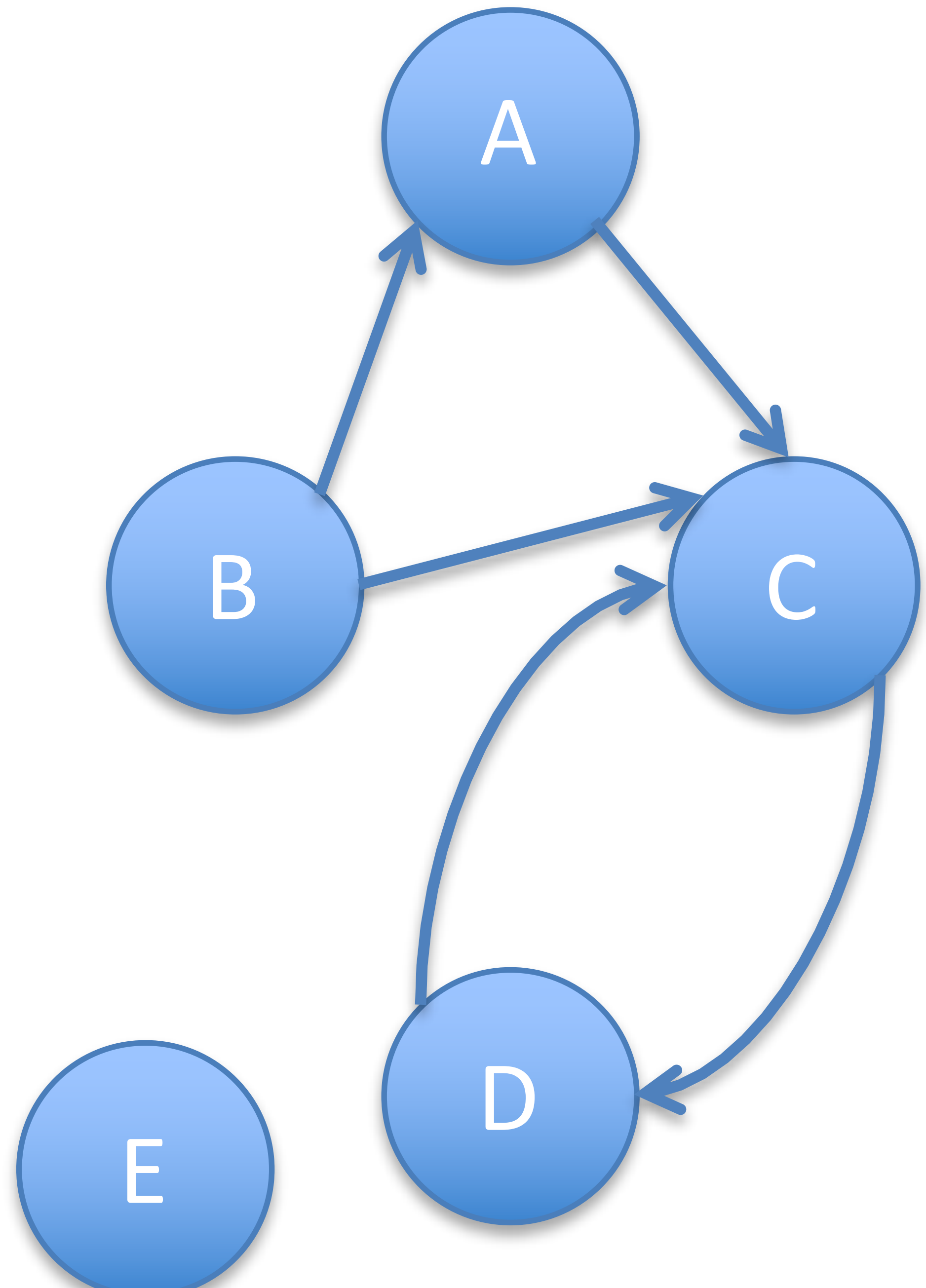
An example directed graph

$$\boldsymbol{V} = \{A, B, C, D, E\}$$

$$\boldsymbol{E} = \{(A, C), (B, A), (B, C), (C, D), (D, C)\}$$

$$|\boldsymbol{V}| = 5$$

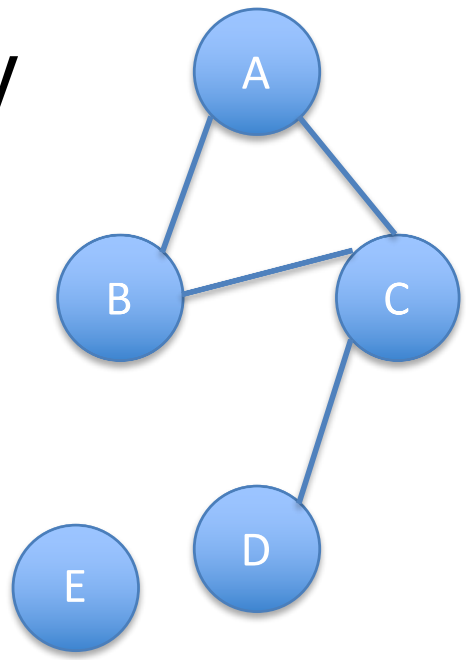
$$|\boldsymbol{E}| = 5$$



Graphs, Formally

- An **un**directed graph is a just like a digraph, but
  - E is a set of **un**ordered pairs (u, v) where u, v are in V

$$
\begin{aligned}
V &= \{A, B, C, D, E\} \\
E &= \{\{A,C\}, \{B,A\}, \\
&\qquad \{B,C\}, \{C,D\}\} \\
|V| &= 5 \\
|E| &= 4
\end{aligned}
$$



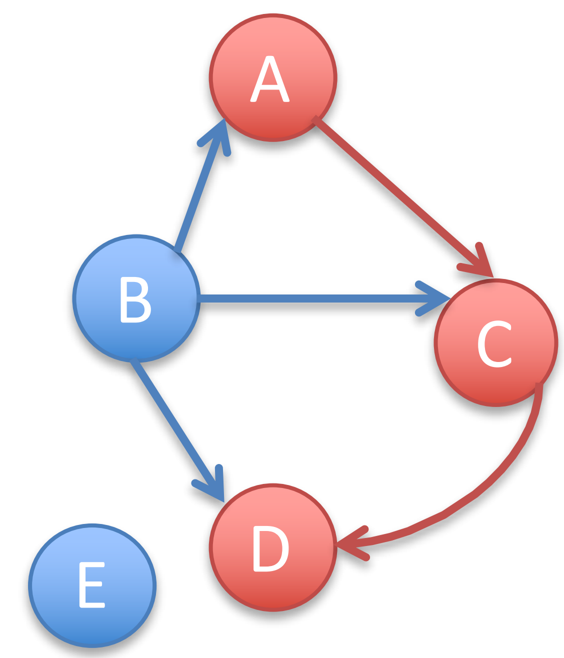
- An **un**directed graph can be converted to an equivalent **directed** graph:
  - Replace each undirected edge with two directed edges in opposite directions
- A **directed** graph can't always be converted to an **undirected** graph.

Graph Terminology: Adjacency, Degree

- Two vertices are adjacent if they are connected by an edge
- Nodes u and v are called the source and sink of the **directed** edge (u, v)
- Nodes u and v are endpoints of an edge (u, v) (directed or undirected)
- The outdegree of a vertex u in a **directed** graph is the number of edges for which u is the source
- The indegree of a vertex v in a **directed** graph is the number of edges for which v is the sink
- The degree of a vertex u in an **undirected** graph is the number of edges of which u is an endpoint

Graph Teminology: Paths, Cycles

- A path is a sequence of vertices where each consecutive pair are adjacent.
- In a directed graph, paths must follow the direction of the edges (nodes must be ordered source then sink).



Path A,C,D

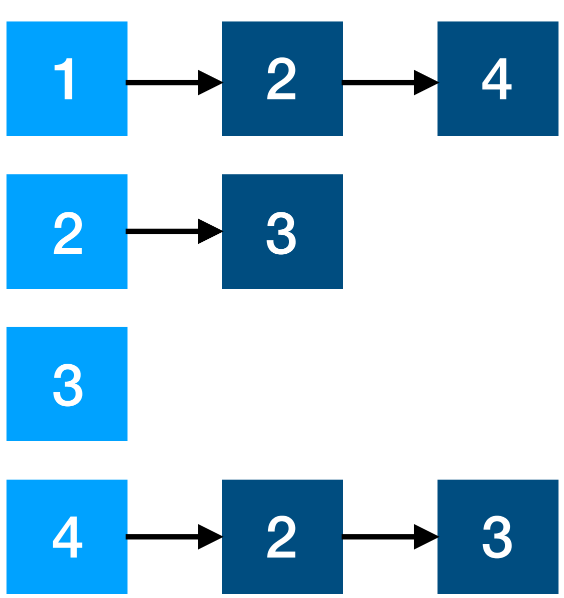
- A cycle is a path that ends where it started, e.g.: x, y, z, x
- A graph is acyclic if it has no cycles.

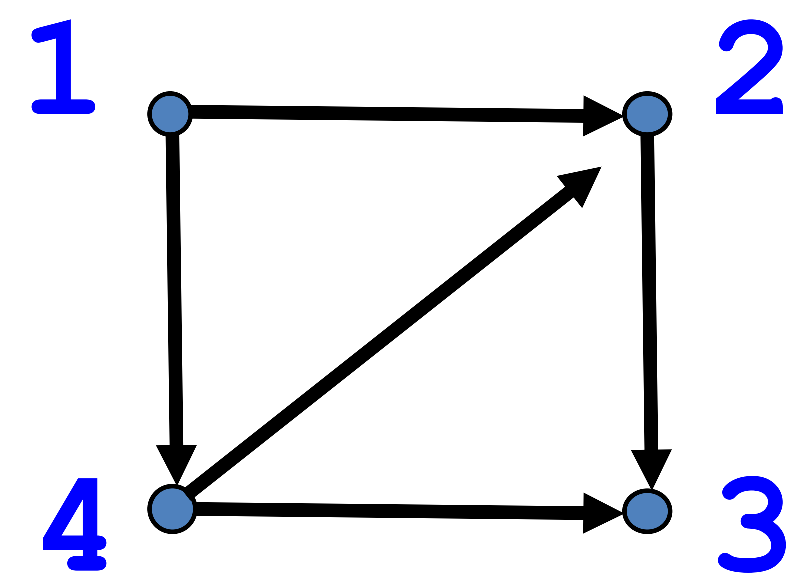
Representing Graphs: Adjacency Lists

```
public class GraphNode {
    // fields storing information
    // about this node

    List<GraphNode> neighbors;
}
```

Node: Neighbors:



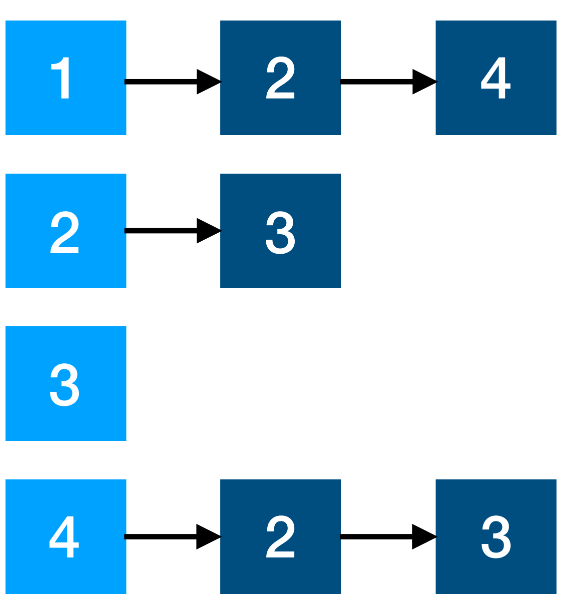


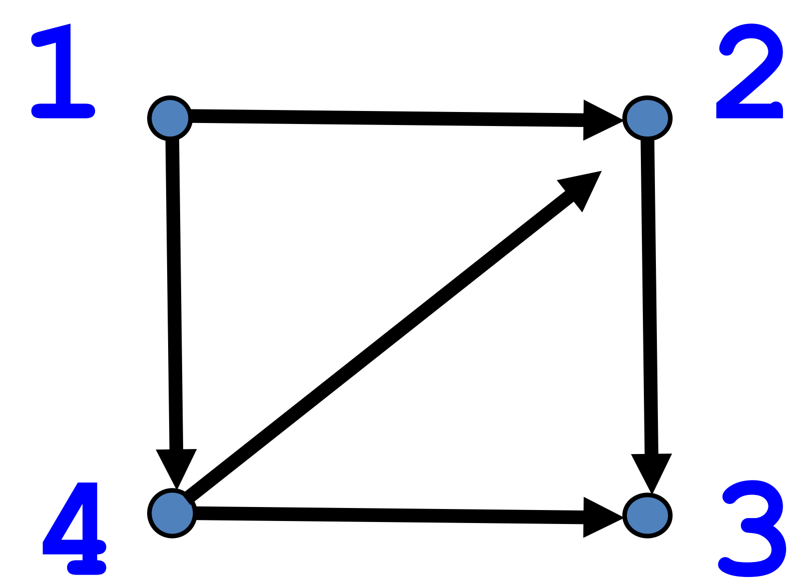
Representing Graphs: Adjacency Matrix

```
public class Graph {
    boolean[][] adjacent; // size n x n
}
```

Adjacency lists:

Node: Neighbors:



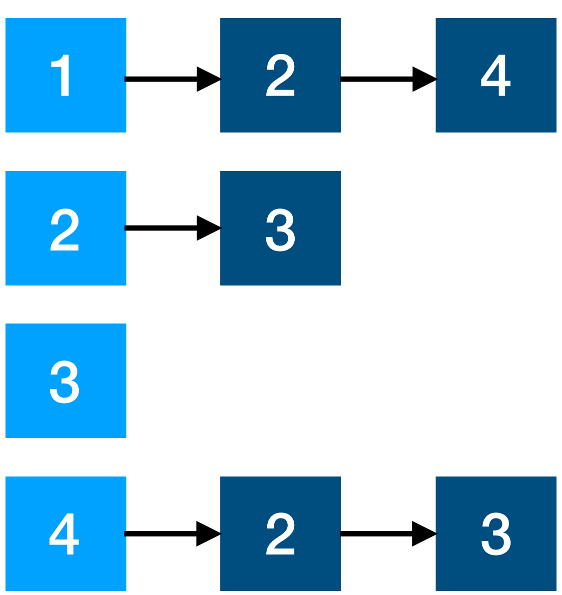


Representing Graphs: Adjacency Matrix

```
public class Graph {
    boolean[][] adjacent; // size n x n
}
```

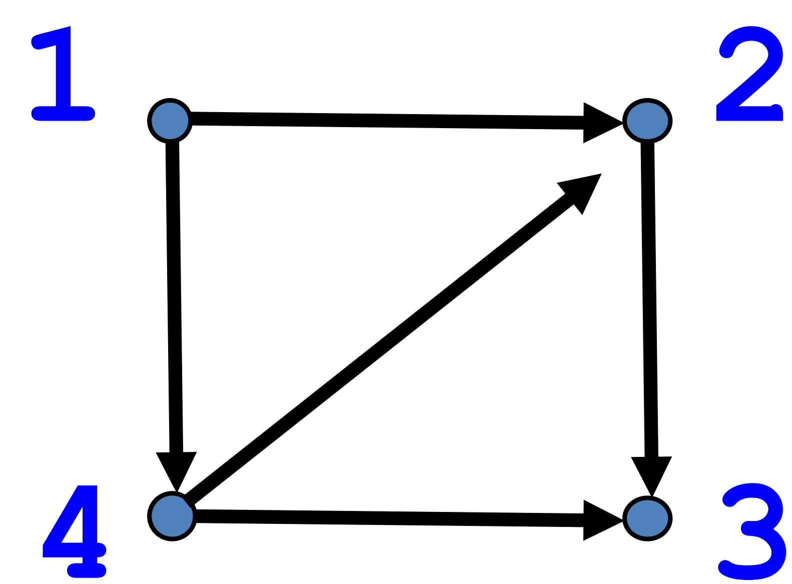
Adjacency lists:

Node: Neighbors:



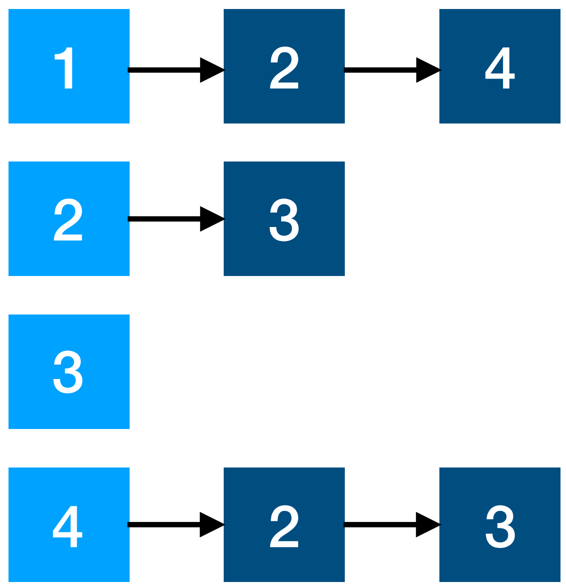
Adjacency Matrix:

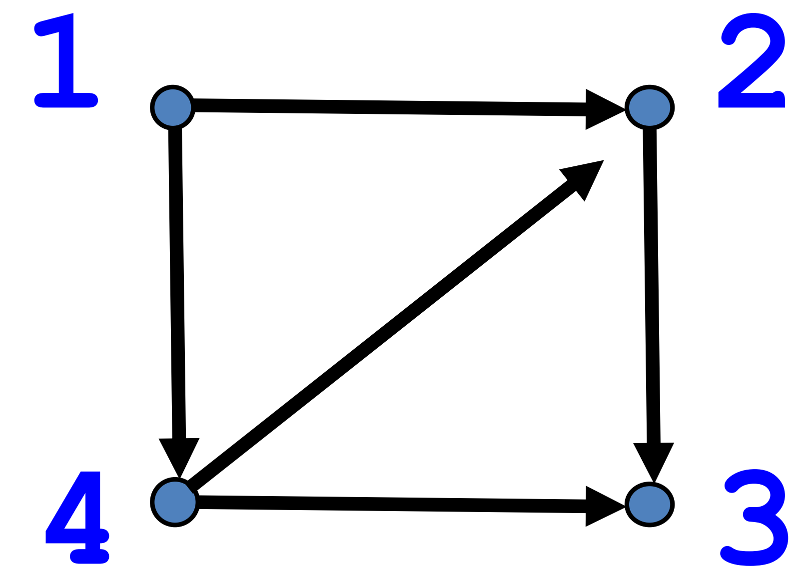
	1	2	3	4
1	0	1	0	1
2	0	0	1	0
3	0	0	0	0
4	0	1	1	0



Adjacency lists:

Node: Neighbors:



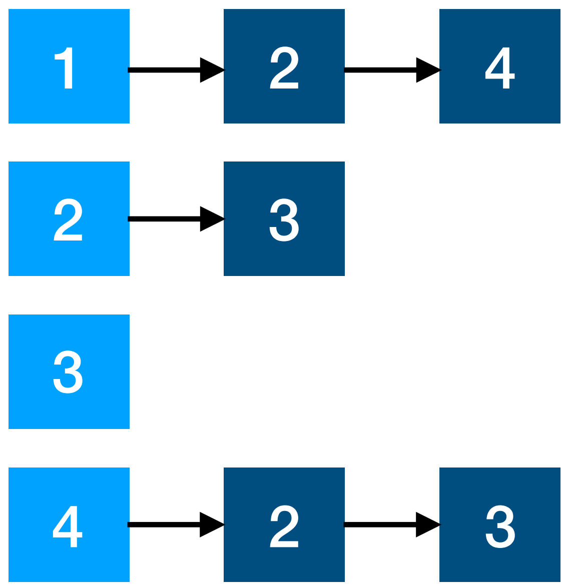


- Let $n = |V|$ and $m = |E|$; let $d(u)$ = degree of u

- ABCD: How much space does it take to store G as an adjacency list vs. adjacency matrix?
  A. List: $O(n^2+e)$; Matrix: $O(n^2)$
  B. List: $O(n+e)$; Matrix: $O(n + e)$
  C. List: $O(n^2)$; Matrix: $O(n + e^2)$
  D. List: $O(n+e)$; Matrix: $O(n^2)$

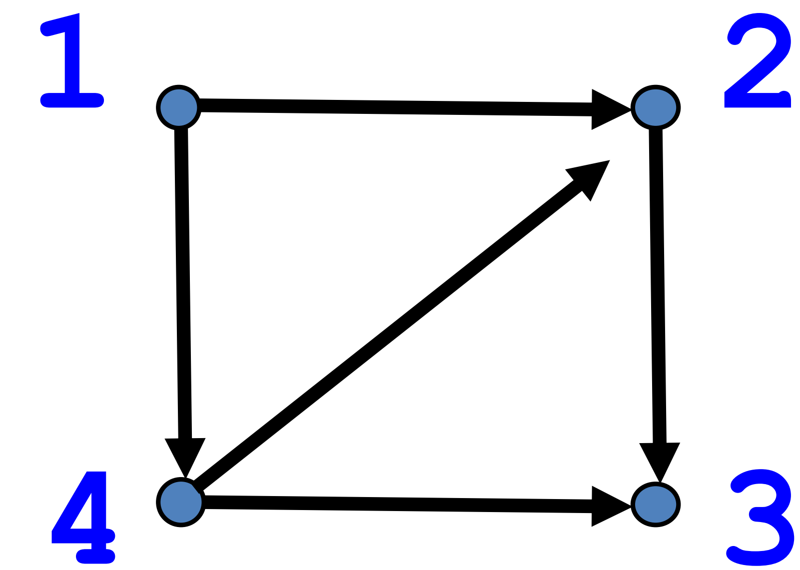
Adjacency lists:

Node: Neighbors:



Adjacency Matrix:

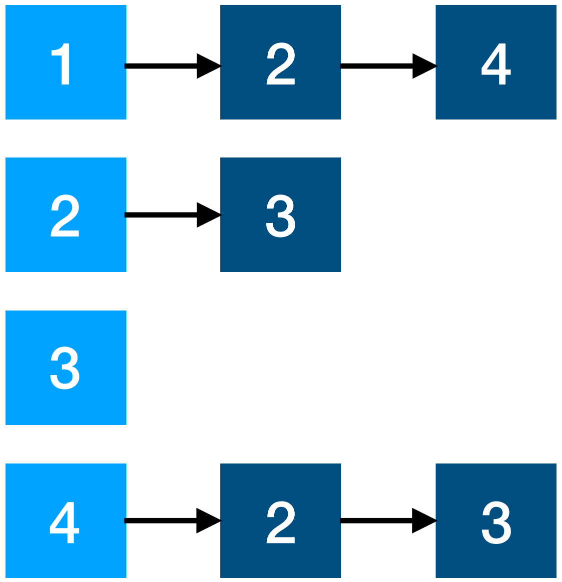
$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 1 & 1 & 0 \end{array}$$

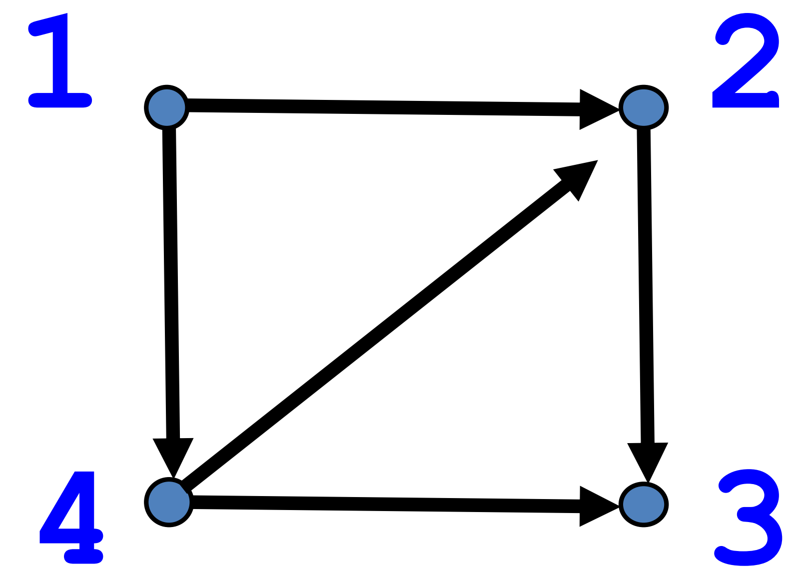


- Let n = |V| and m = |E|; let d(u) = degree of u
- ABCD: How much space does it take to store G as an adjacency list vs. adjacency matrix?
  A. List: $O(n^2+e)$; Matrix: $O(n^2)$
  B. List: $O(n+e)$; Matrix: $O(n + e)$
  C. List: $O(n^2)$; Matrix: $O(n + e^2)$
  D. List: $O(n+e)$; Matrix: $O(n^2)$

Adjacency lists:

Node: Neighbors:



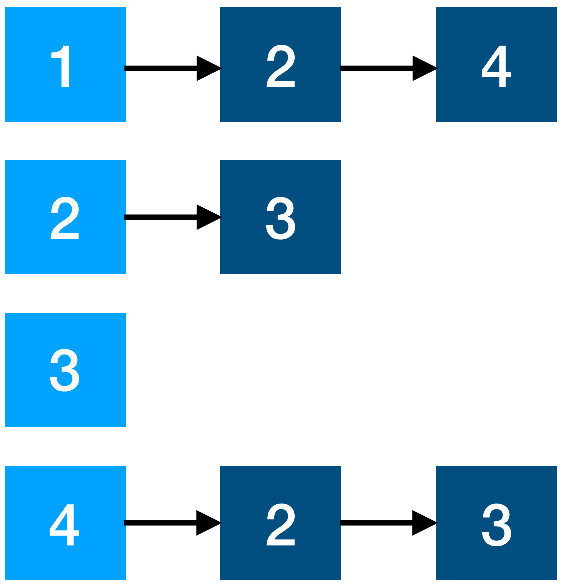


- Let n = |V| and m = |E|; let d(u) = degree of u

- ABCD: What's the runtime of iterating over all edges?
  A. List: $O(n^2)$; Matrix: $O(n^2)$
  B. List: $O(n+e)$; Matrix: $O(n^2)$
  C. List: $O(n + e)$; Matrix: $O(n + e)$
  D. List: $O(n+e)$; Matrix: $O(n^2 + e)$

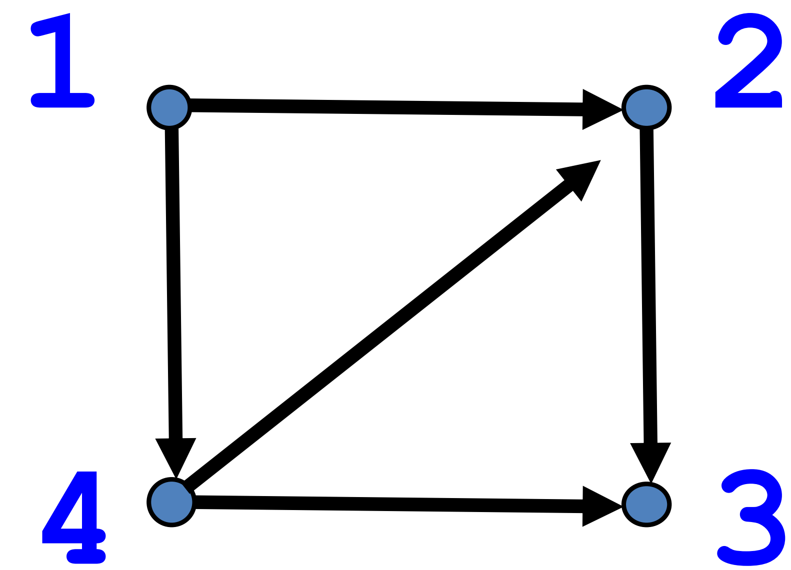
Adjacency lists:

Node: Neighbors:



Adjacency Matrix:

$$
\begin{array}{c|cccc}
 & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} \\
\hline
\mathbf{1} & 0 & 1 & 0 & 1 \\
\mathbf{2} & 0 & 0 & 1 & 0 \\
\mathbf{3} & 0 & 0 & 0 & 0 \\
\mathbf{4} & 0 & 1 & 1 & 0
\end{array}
$$



- Let n = |V| and m = |E|; let d(u) = degree of u

- ABCD: What's the runtime of iterating over all edges?
    A. List: $O(n^2)$; Matrix: $O(n^2)$
    B. List: $O(n+e)$; Matrix: $O(n^2)$
    C. List: $O(n + e)$; Matrix: $O(n + e)$
    D. List: $O(n+e)$; Matrix: $O(n^2 + e)$

Adjacency Matrix vs Adjacency List

- Reminder: $n = |V|$ and $m = |E|$; let $d(u)$ = degree of u
- Adjacency matrix:
  - Storage space: $O(n^2)$
  - Iterate over edges: $O(n^2)$ time
  - Check if there's an edge from u to v: $O(1)$
  - Good for dense graphs
    - e.g., if n^2 is close to n^2, you need n^2 storage anyway.

Adjacency Matrix vs Adjacency List

- Reminder: $n = |V|$ and $m = |E|$; let $d(u)$ = degree of u

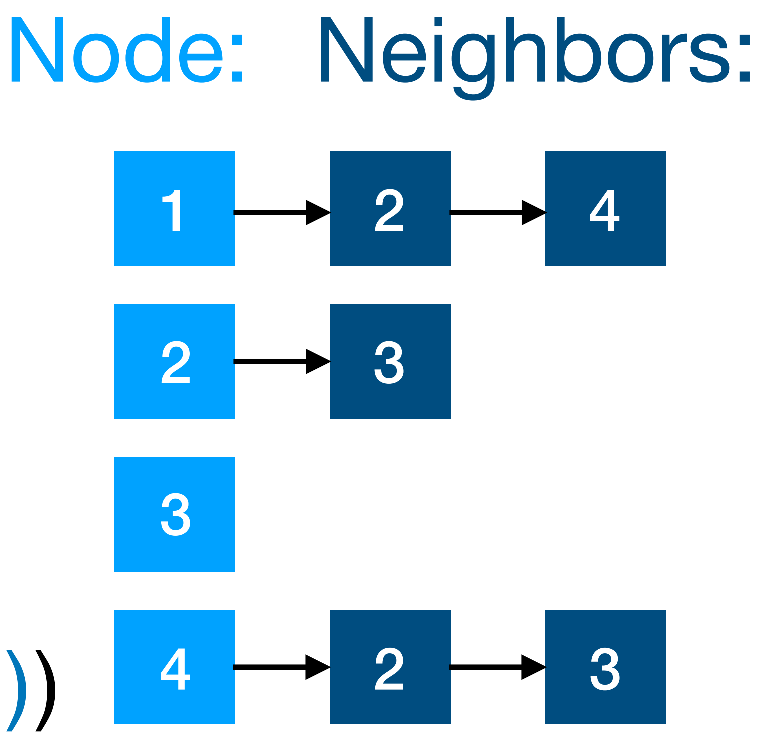
- Adjacency matrix:

$$
\begin{array}{c|cccc}
 & 1 & 2 & 3 & 4 \\
\hline
1 & 0 & 1 & 0 & 1 \\
2 & 0 & 0 & 1 & 0 \\
3 & 0 & 0 & 0 & 0 \\
4 & 0 & 1 & 1 & 0
\end{array}
$$

  - Storage space: $O(n^2)$
  - Iterate over edges: $O(n^2)$ time
  - Check if there's an edge from u to v: $O(1)$
  - Good for dense graphs
    - e.g., if n^2 is close to n^2, you need n^2 storage anyway.

Adjacency Matrix vs Adjacency List

- Reminder: n = |V| and m = |E|; let d(u) = degree of u
- Adjacency list:
  - Storage space: O(n + e)
  - Iterate over edges: O(n + e) time
  - Check if there's an edge from u to v: O(d(u))
  - Good for more sparse graphs:
    - e.g., if |E| is close to n, n + e ~= 2n, which is O(n)



Graph Algorithms

You can take entire graduate-level courses on graph algorithms. In this class:

- Search/traversal: search for a particular node or traverse all nodes (Lab 9)
    - Breadth-first
    - Depth-first
- Shortest Paths (A4)
- Spanning trees

Graph Algorithms

You can take entire graduate-level courses on graph algorithms. In this class:

- Search/traversal: search for a particular node or traverse all nodes (Lab 9)
  - Breadth-first
  - Depth-first
- Shortest Paths (A4)
- Spanning trees