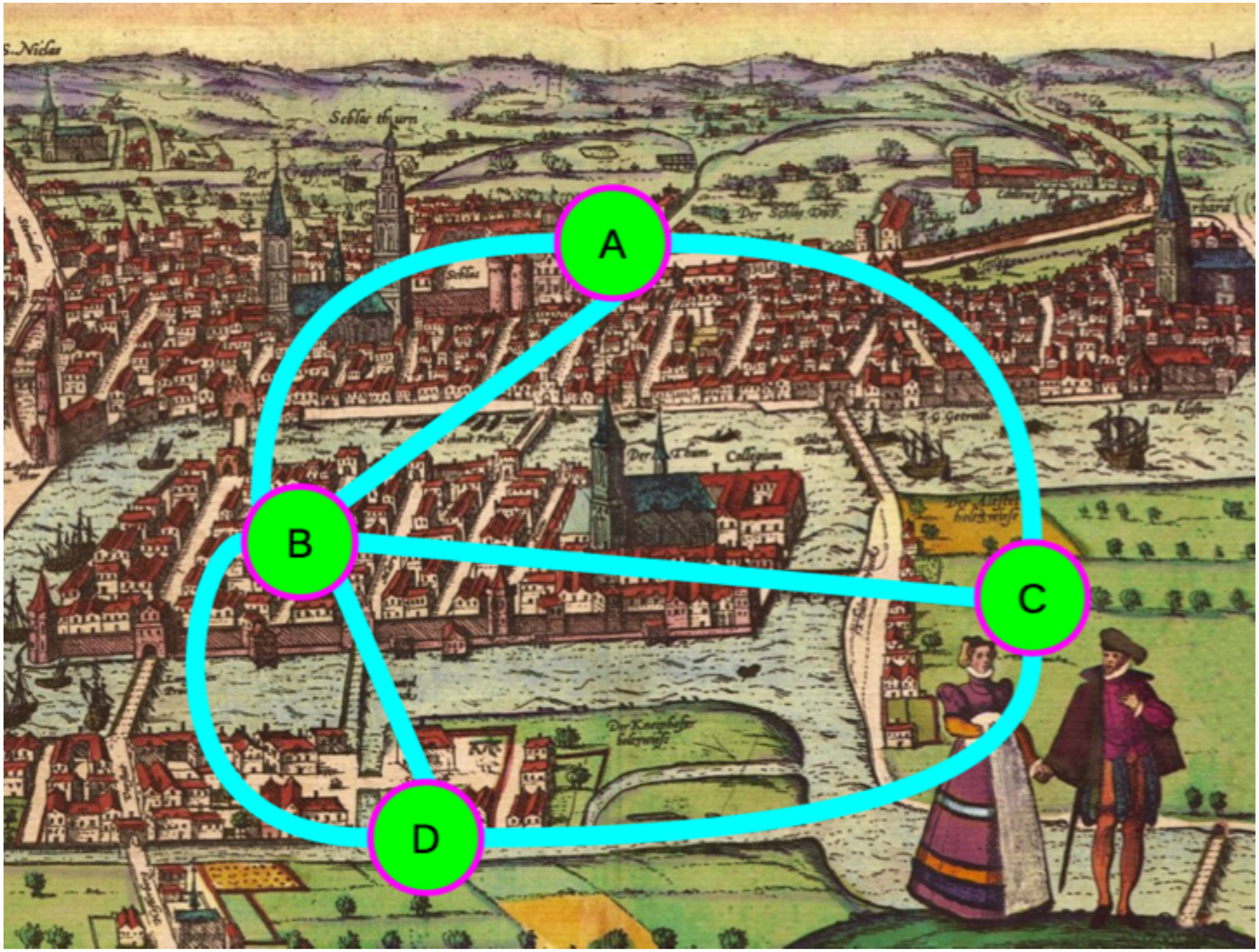



# CSCI 241

Lecture 19

A3 Overview, Intro to Graphs

# Announcements

- A3 has phases!
- Phase 1 will be released today or tomorrow.
- Phases 2 and 3 should be out by Tuesday.

# Goals

- Understand the architecture of A3
- Know the definition of a graph and basic associated terminology:
  - Node/vertex; edge/arc; directed, undirected; adjacent; (in/out-)degree; path; cycle;

# Goals (next time)

- Understand how to represent graphs and implement basic graph operations using:
  - Adjacency list
  - Adjacency matrix

# A3 has 4 phases.

# A3 has 4 phases.

It may sound scary.

# A3 has 4 phases.

It may sound scary.


It isn't so bad:

total lines of code is probably <= A2

nothing here is as tricky as AVL rebalance

I'm giving you the unit tests.

# A3 has 4 phases.

0. Write an ArrayList clone

# A3 has 4 phases.

0. Write an ArrayList clone
   (done!)

# A3 has 4 phases.

0. Write an ArrayList clone
   (done by Sunday!)

# A3 has 3 phases.

use AList to handle growing the array!

1. Write a min-heap to implement a priority queue with operations:
   - `boolean add(V value, P priority)`
   - `V peek();`
   - `V poll();`
2. Write a hash table.

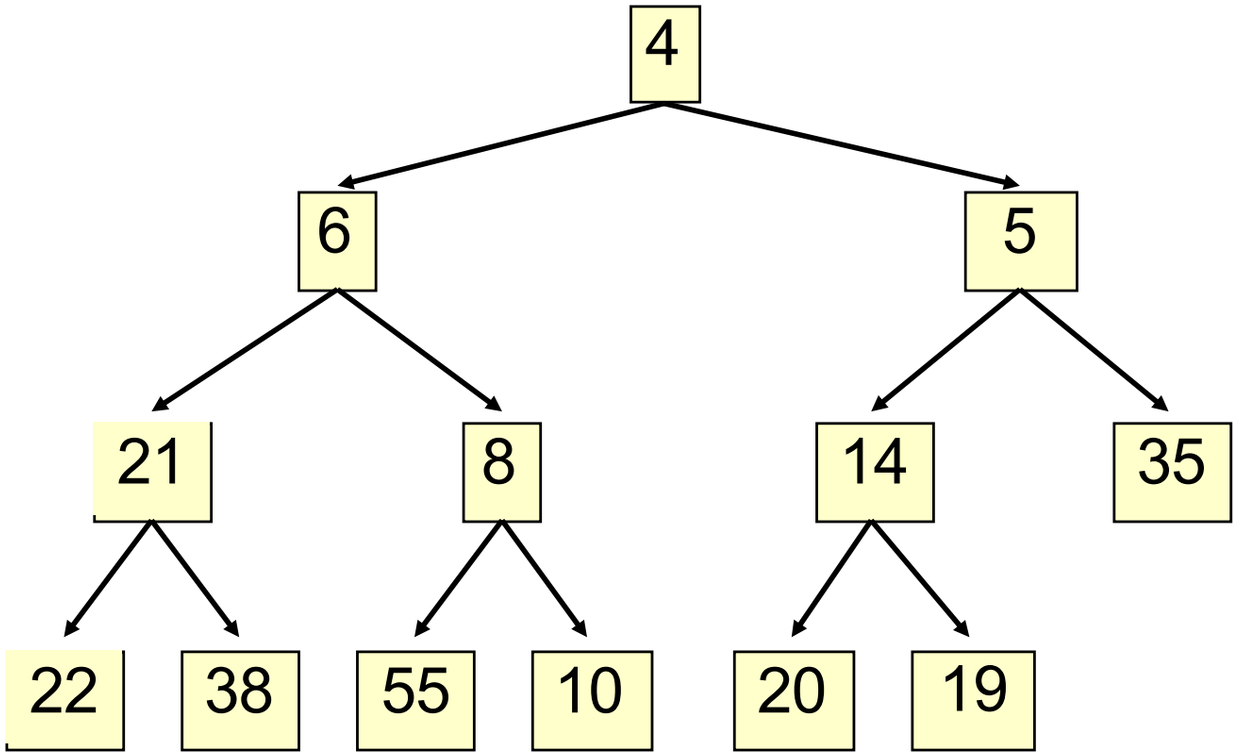
use AList to handle growing the array!

3. Use the hash table to augment the heap, making the following operations efficient:
   - `boolean contains(V v);`
   - `void changePriority(V v, P newP);`

# Phase 3 - Hash your Heap

In Phase 1 Heap:

- `contains` requires searching the whole tree.
- `changePriority` requires searching the whole tree, then bubbling down or up.



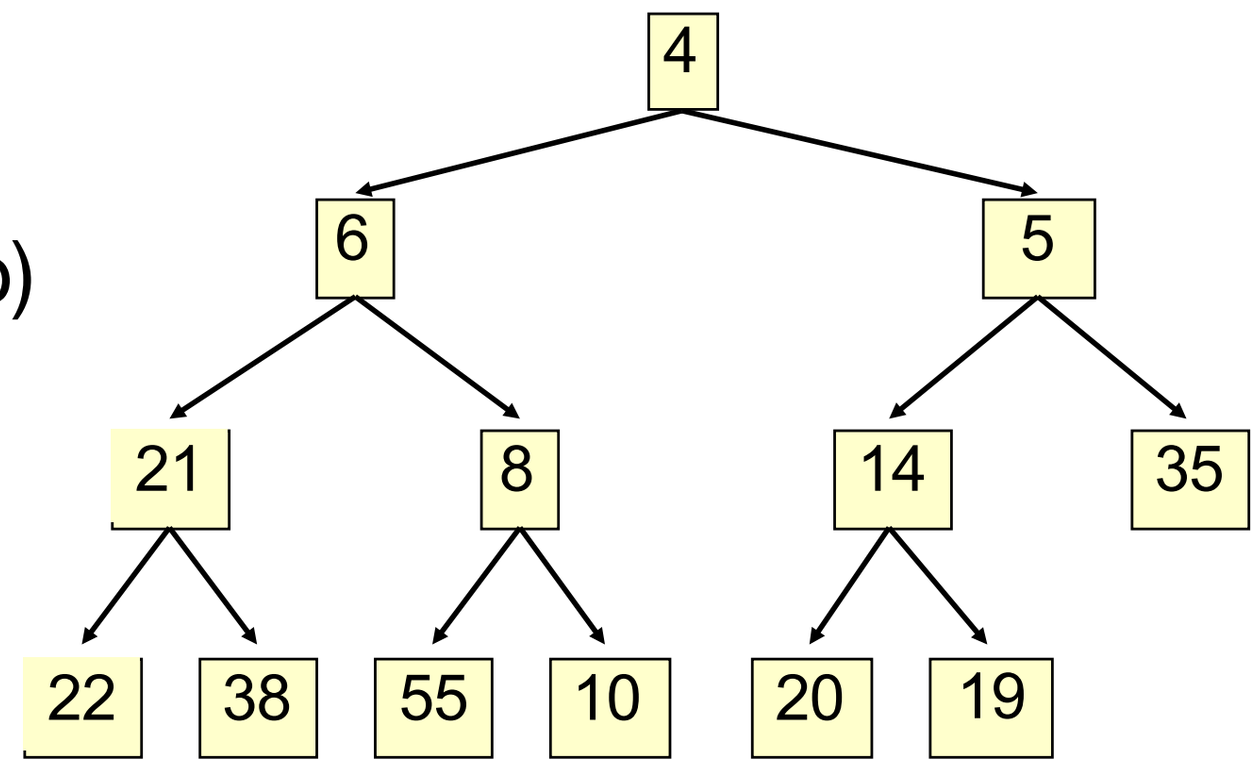
# Phase 3 - Hash your Heap

In Phase 3 Heap:

- Each node is stored in the heap and in a HashTable that tracks its index in the heap.

HashTable<V, Integer>:

| value | i (index in heap) |
|---|---|
| 4 | 0 |
| 8 | 4 |
| 6 | 1 |
| 38 | 8 |
| 35 | 6 |
| 21 | 3 |
| 10 | 10 |




| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heap: [ | 4 | 6 | 5 | 21 | 8 | 14 | 35 | 22 | 38 | 55 | 10 | 20 | 19 ] |

# Phase 3 - Hash your Heap

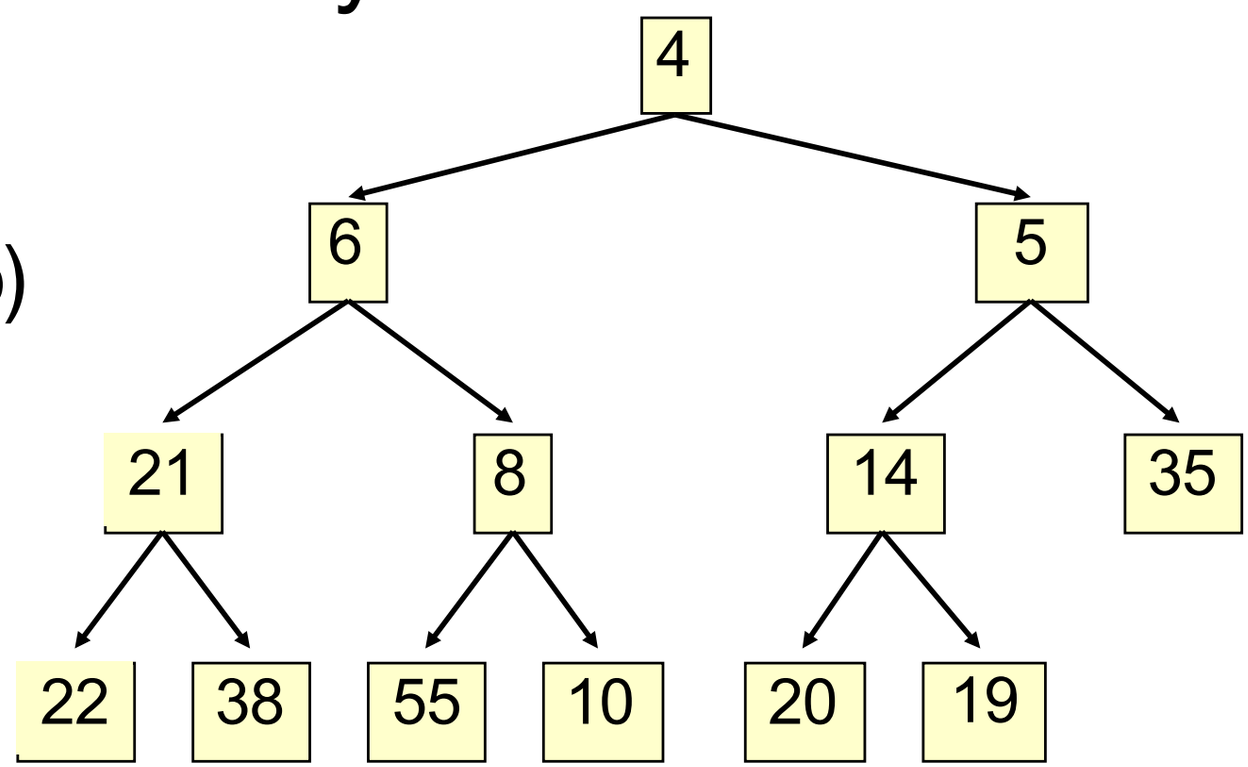
In Phase 3 Heap:

`boolean contains(V v):`

true iff map contains key v

HashTable<V, Integer>:

| value | i (index in heap) |
|---|---|
| 4 | 0 |
| 8 | 4 |
| 6 | 1 |
| 38 | 8 |
| 35 | 6 |
| 21 | 3 |
| 10 | 10 |




| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heap: [ | 4 | 6 | 5 | 21 | 8 | 14 | 35 | 22 | 38 | 55 | 10 | 20 | 19 ] |

# Phase 3 - Hash your Heap

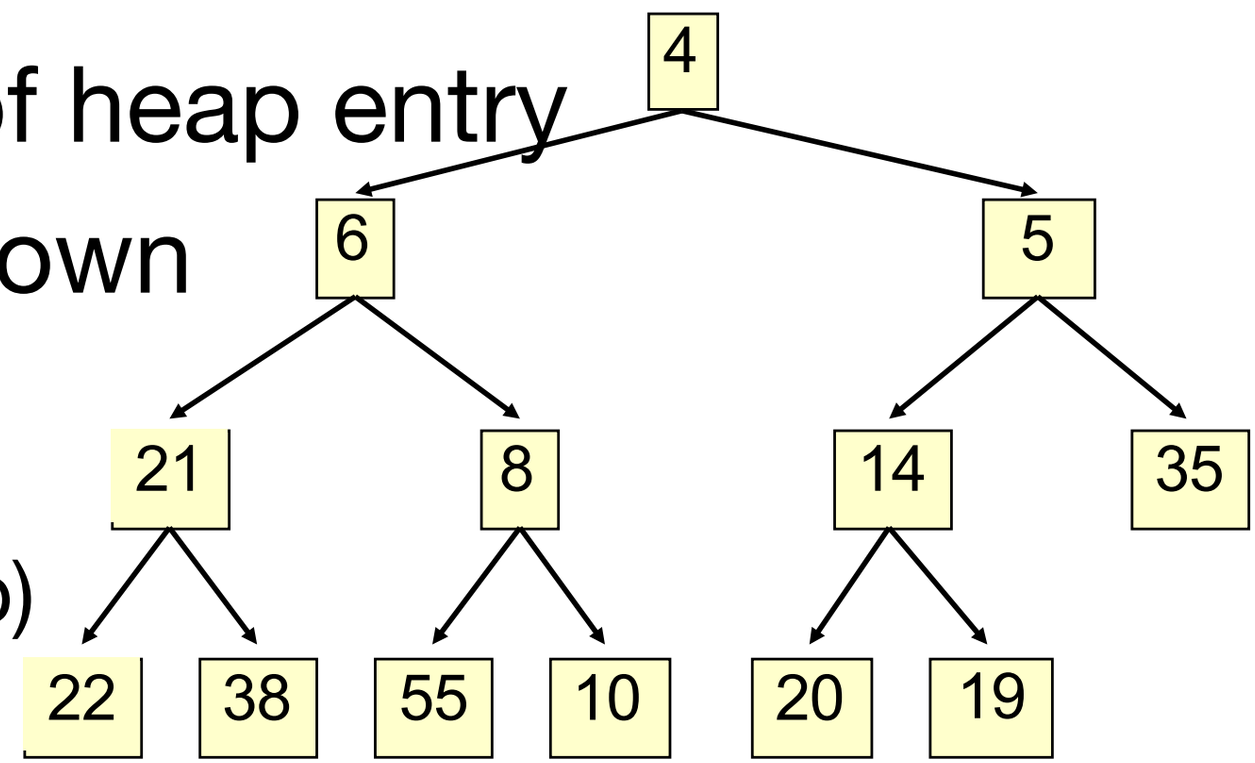
In Phase 3 Heap:

```
void changePriority(V v, P newP):
```

i = map.get(v);

change priority of heap entry

bubble it up or down

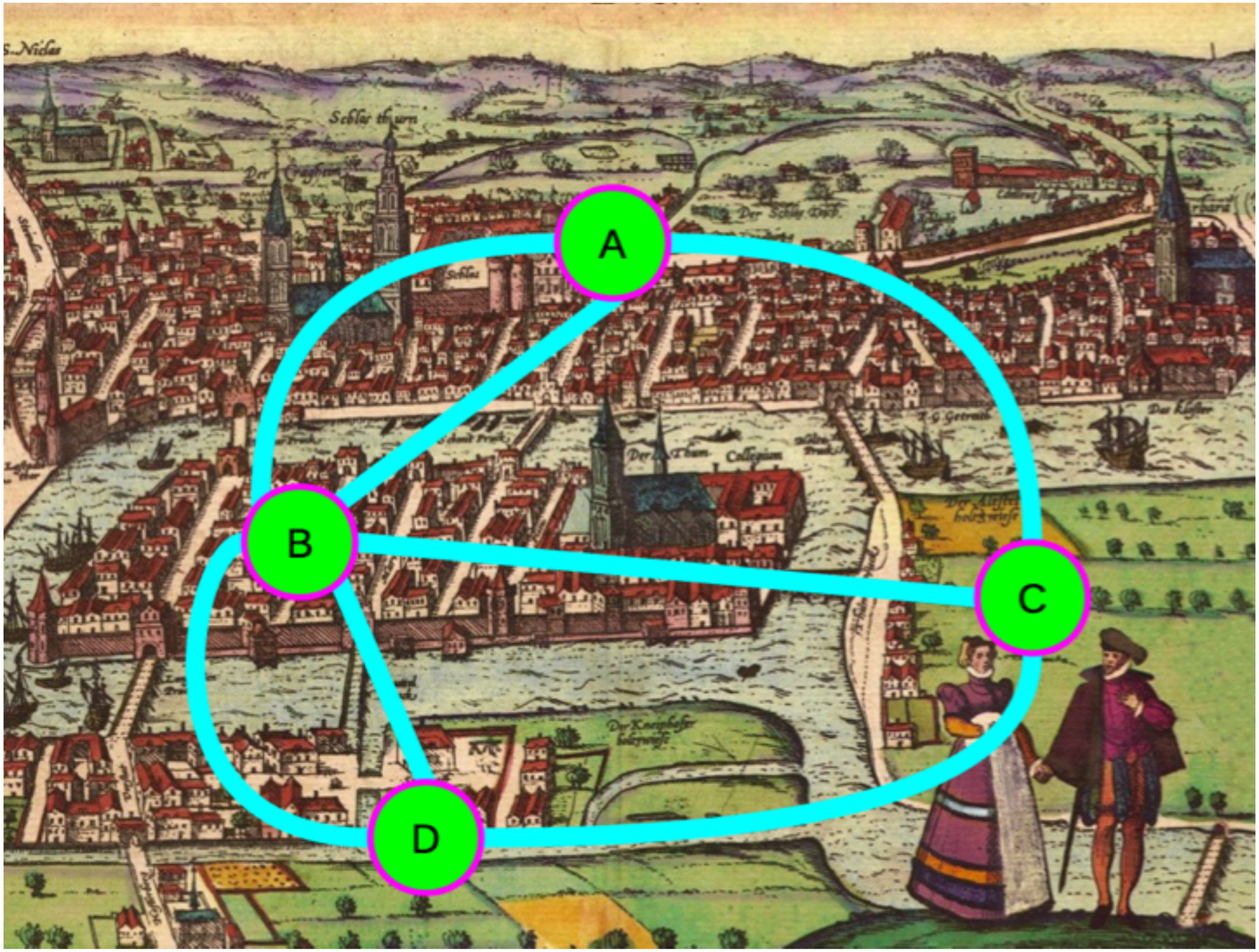



HashTable<V, Integer>:

| value | i (index in heap) |
|---|---|
| 4 | 0 |
| 8 | 4 |
| 6 | 1 |
| 38 | 8 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heap: | 4 | 6 | 5 | 21 | 8 | 14 | 35 | 22 | 38 | 55 | 10 | 20 | 19 |

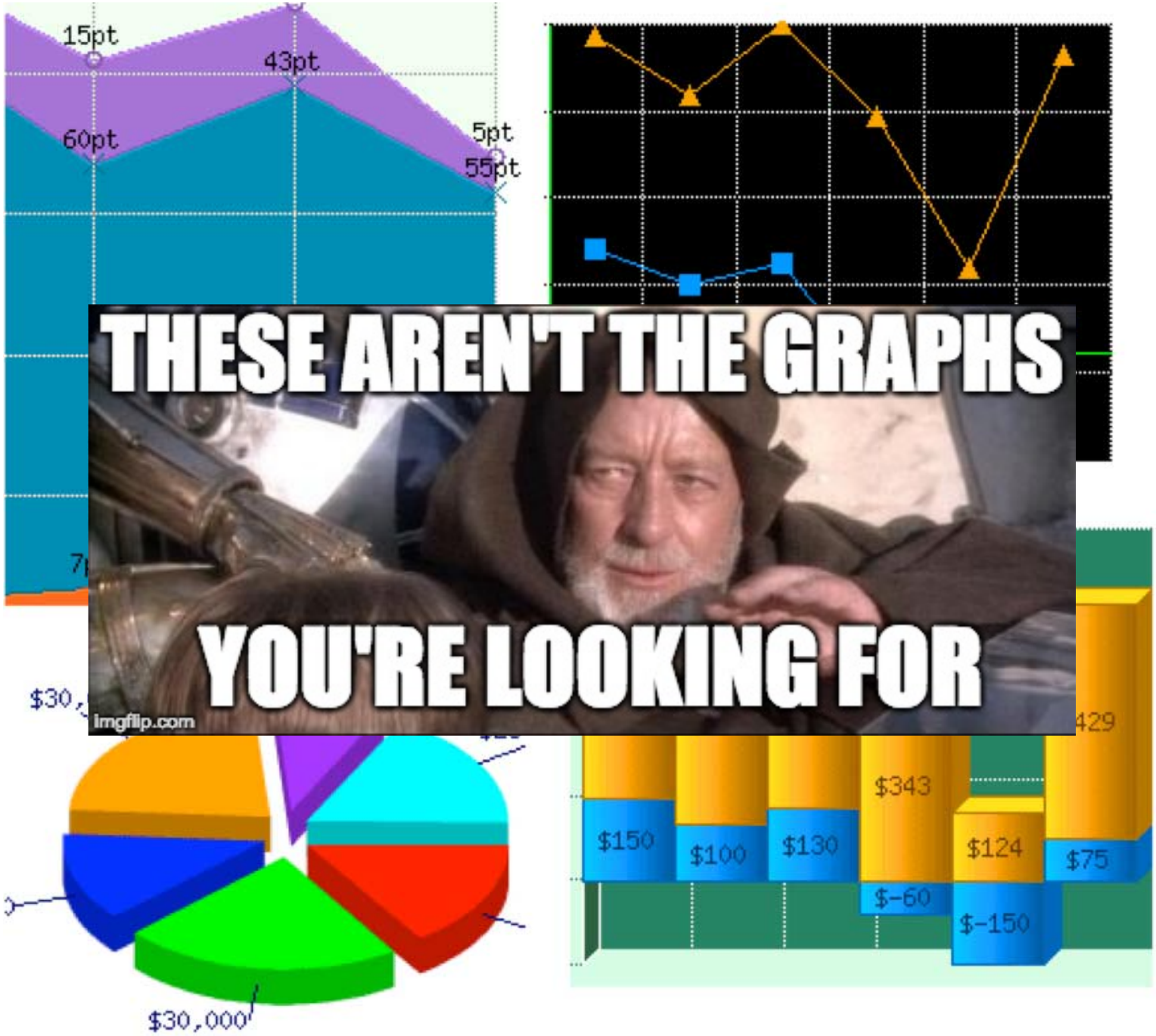
# Questions?



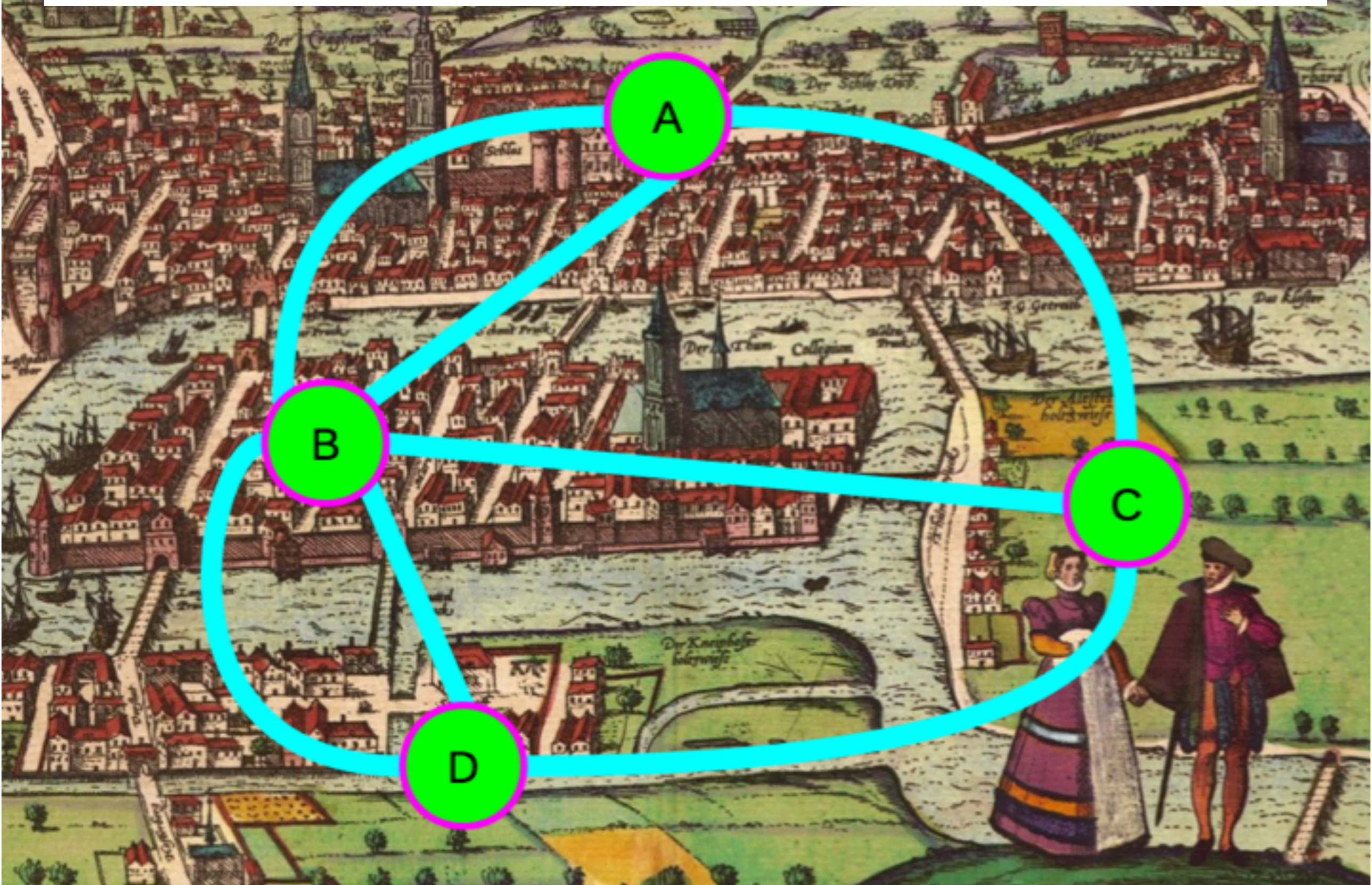

# CSCI 241

Lecture 19

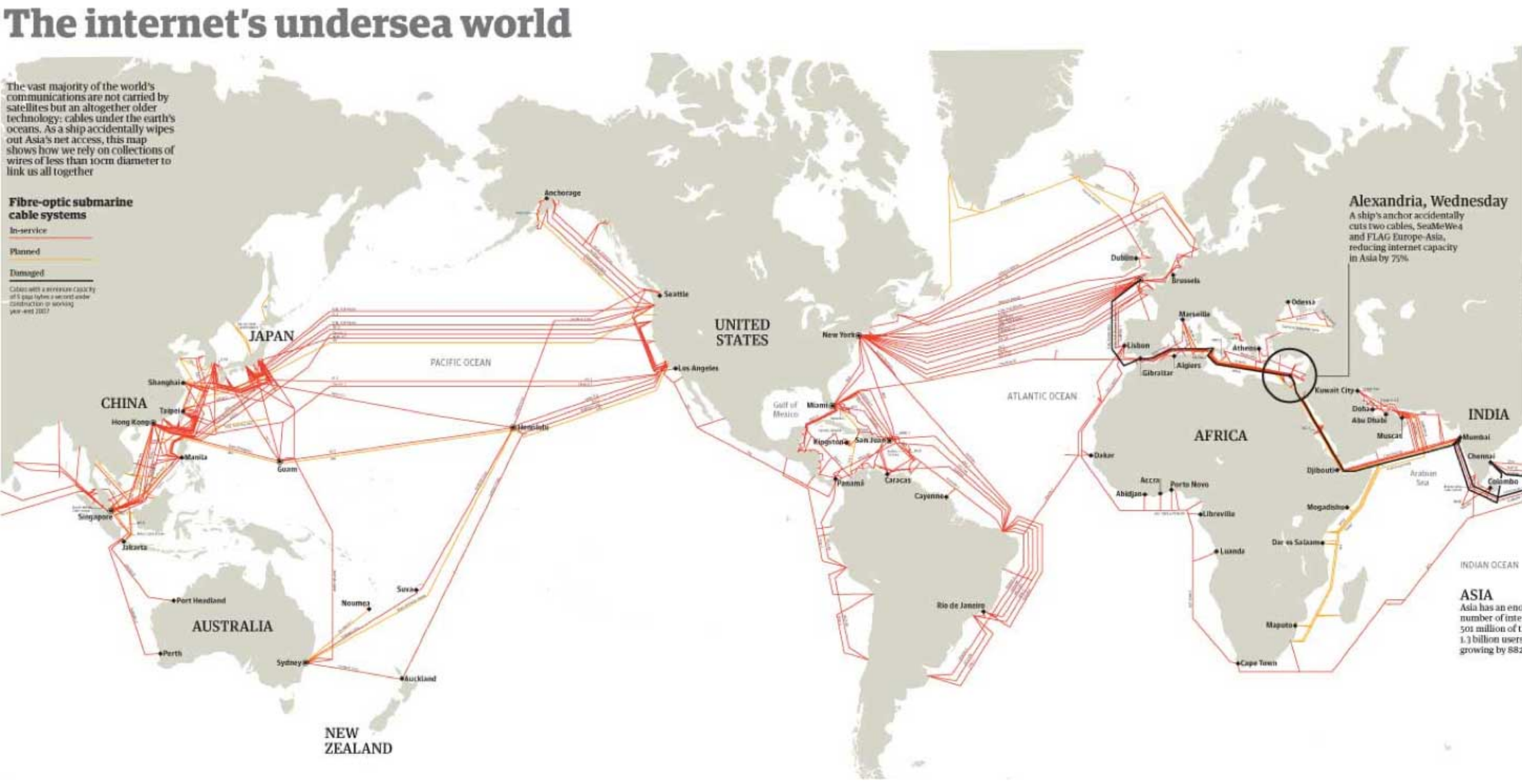
~~A3 Overview,~~ Intro to Graphs


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

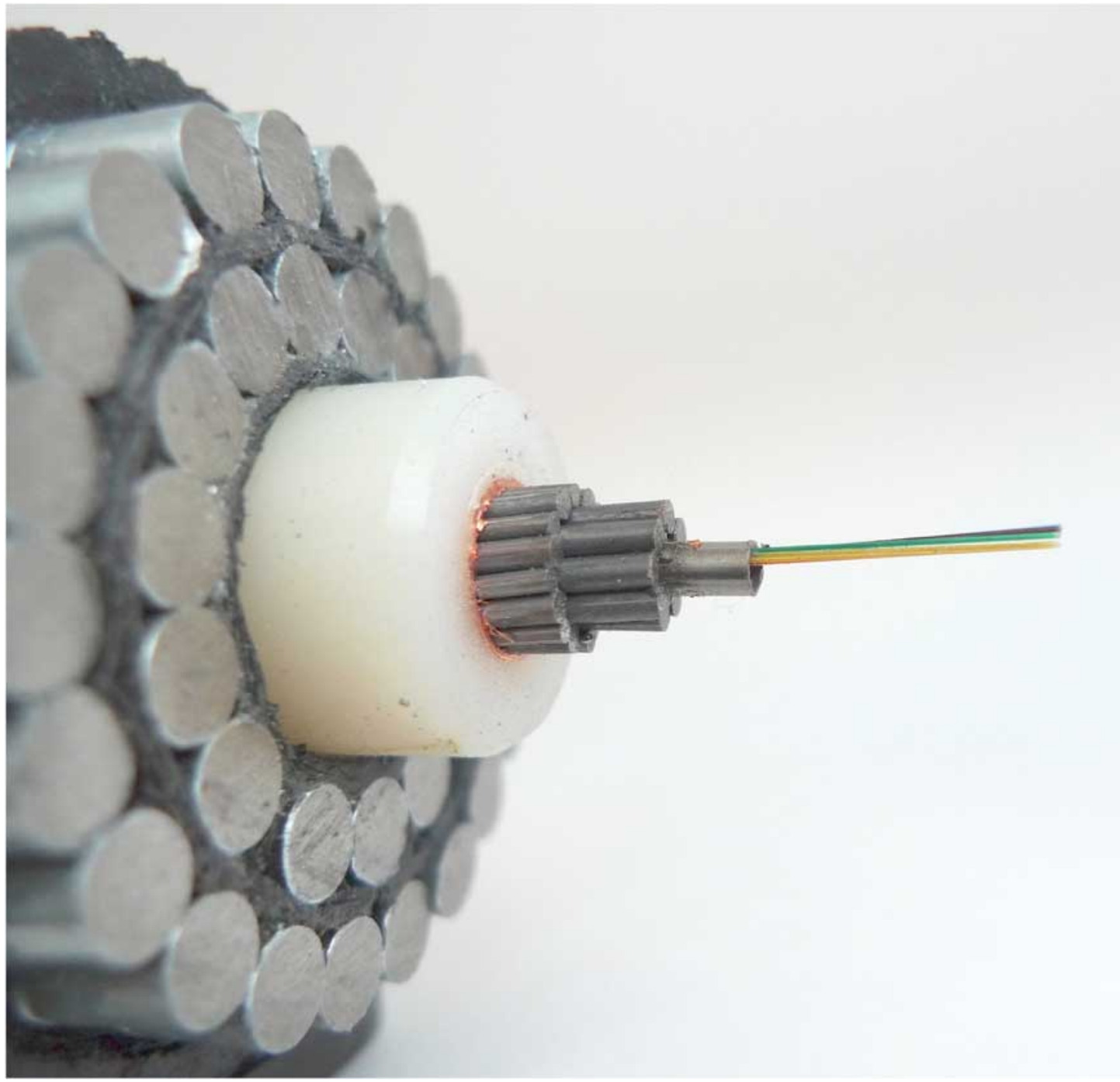
**Graph: a bunch of points connected by lines.**
**The lines may have directions, or not.**



# This is a graph:



# The edges are made of these:



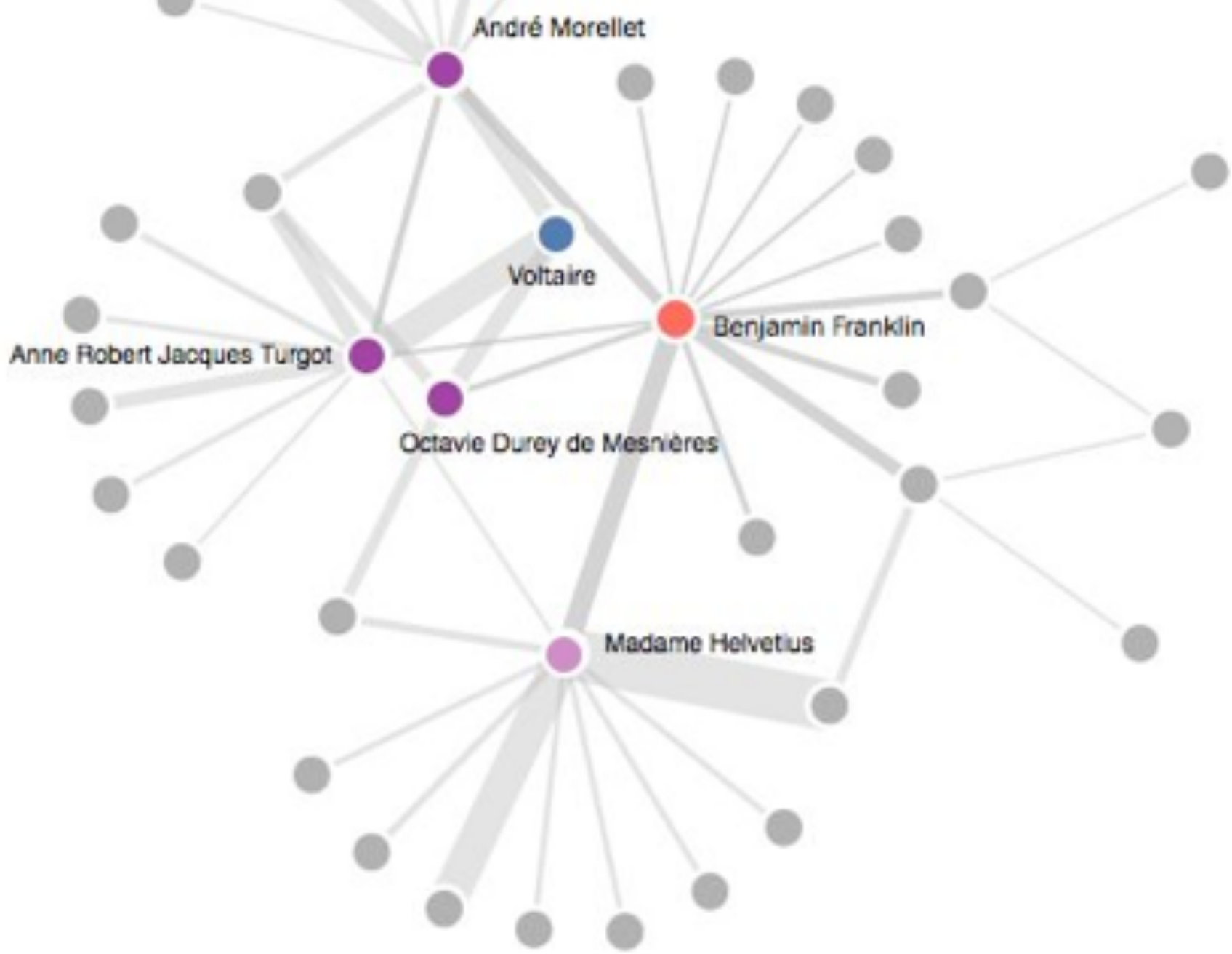
# Social Networks

(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

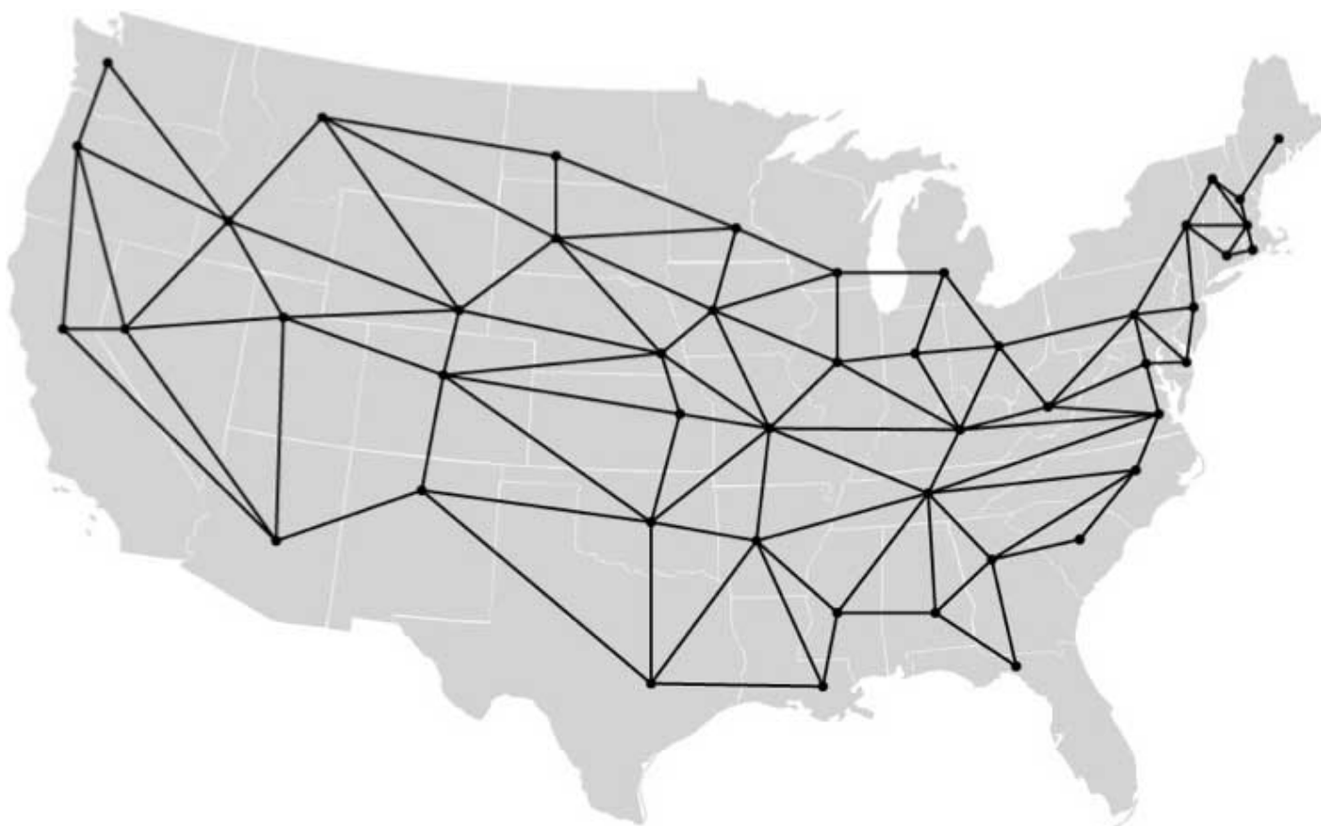
(before they were cool)

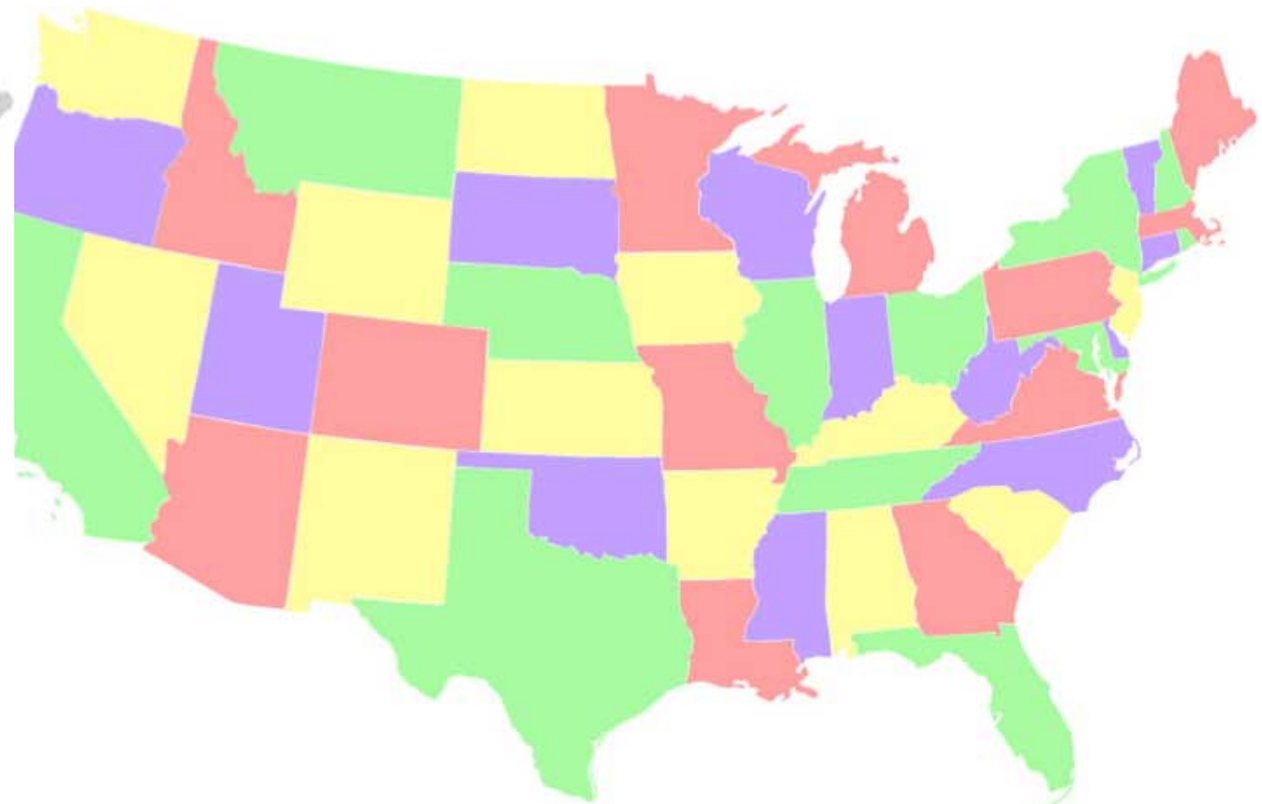



MANHATTAN
HARLEM
EAST HARLEM
UPPER EAST SIDE
CENTRAL PARK
METROPOLITAN MUSEUM OF ART
MOTT HAVEN
HUNTS POINT
RIKERS ISLAND
RANDALLS ISLAND
ROOSEVELT ISLAND
East River
ASTORIA
LONG ISLAND CITY
JACKSON HEIGHTS
CORONA
FLUSHING
FLUSHING MEADOWS CORONA PARK
JUNIPER VALLEY PARK
MIDDLE VILLAGE
MASPETH
GLENDALE
REGO PARK
Murray Hill
LAGUARDIA AIRPORT
LaGuardia Link Q70 SBS
M60 SBS
Q47
Q48
Q72
THE HUB
Harlem 148 St
3
145 St
A·B·C·D
135 St
B·C
125 St
A·B·C·D
M60 SBS LGA Airport
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
50 St
C·E
42 St/Port Authority Bus Terminal
A·C·E
Times Sq-42 St
N·Q·R·S·W·1·2·3·7
34 St Penn Station
34 St Herald Sq
B·D·F·M
Hudson Yards
7
145 St
3
135 St
2·3
125 St
2·3 · M60 SBS LGA Airport
116 St
2·3
Central Park North (110 St)
2·3
96 St
1·2·3
86 St
1
79 St
1
72 St
1·2·3
66 St Lincoln Center
1
50 St
1
57 St-7 Av
N·Q·R·W
49 St
N·R·W
southbound
northbound
57 St
F
47-50 Sts Rockefeller Ctr
B·D·F·M
42 St Bryant Pk
B·D·F·M
5 Av
7
5 Av/59 St
N·R·W
5 Av/53 St
E·M
Lexington Av/63 St
F·Q
Lexington Av/59 St
N·R·W
Lexington Av/53 St
E·M
59 St
4·5·6
51 St
6
Grand Central 42 St
S·4·5·6·7·Metro-North
except S
33 St
6
68 St Hunter College
6
77 St
6
86 St
4·5·6
96 St
6
103 St
6
110 St
6
116 St
6
125 St
4·5·6
96 St
Q
86 St
Q
72 St
Q
138 St-Grand Concourse
4·5
4 skips rush hour peak direction
3 Av 138 St
6
3 Av-149 St
2·5
E 149 St
6
E 143 St St Mary's St
6
Cypress Av
6
Brook Av
6
weekday peak direction express
rush hour peak direction express
Jackson Av
2·5
Roosevelt Island
F
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
Astoria Ditmars Blvd
N·W
Astoria Blvd
M60 SBS N·W
LGA Airport
30 Av
N·W
Broadway
N·W
36 Av
N·W
39 Av
N·W
36 St
M·R
Queens Plaza
E·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 · LaGuardia Link Q70 SBS
Q47 LGA Airport (Marine Air Term only)
Woodside
61 St Woodside
7·LaGuardia Link Q70 SBS
LIRR
52 St
7
46 St Bliss St
7
40 St Lowery St
7
33 St-Rawson St
7
69 St
7
74 St-Broadway
7·LaGuardia Link Q70 SBS
82 St-Jackson Hts
7
90 St-Elmhurst Av
7
Junction Blvd
7·Q72 LGA Airport
103 St-Corona Plaza
7
111 St
7·Q48 LGA Airport
Mets-Willets Point
7·Q48 LGA Airport
Flushing Main St
7
Elmhurst Av
M·R
Grand Av Newtown
M·R
Woodhaven Blvd
M·R
63 Dr-Rego Park
M·R·Q72 LGA Airport
67 Av
M·R
Middle Village Metropolitan Av
M
Fresh Pond Rd
M
Forest Av
M
TRAMWAY
QUEENSBORO BRIDGE
QUEENS MIDTOWN TUNNEL
UNITED NATIONS
BROADWAY
QUEENS BLVD
ROOSEVELT AV
DITMARS BLVD
ASTORIA BLVD
LONG ISLAND EXPWY
METROPOLITAN AV
rush hour peak direction express

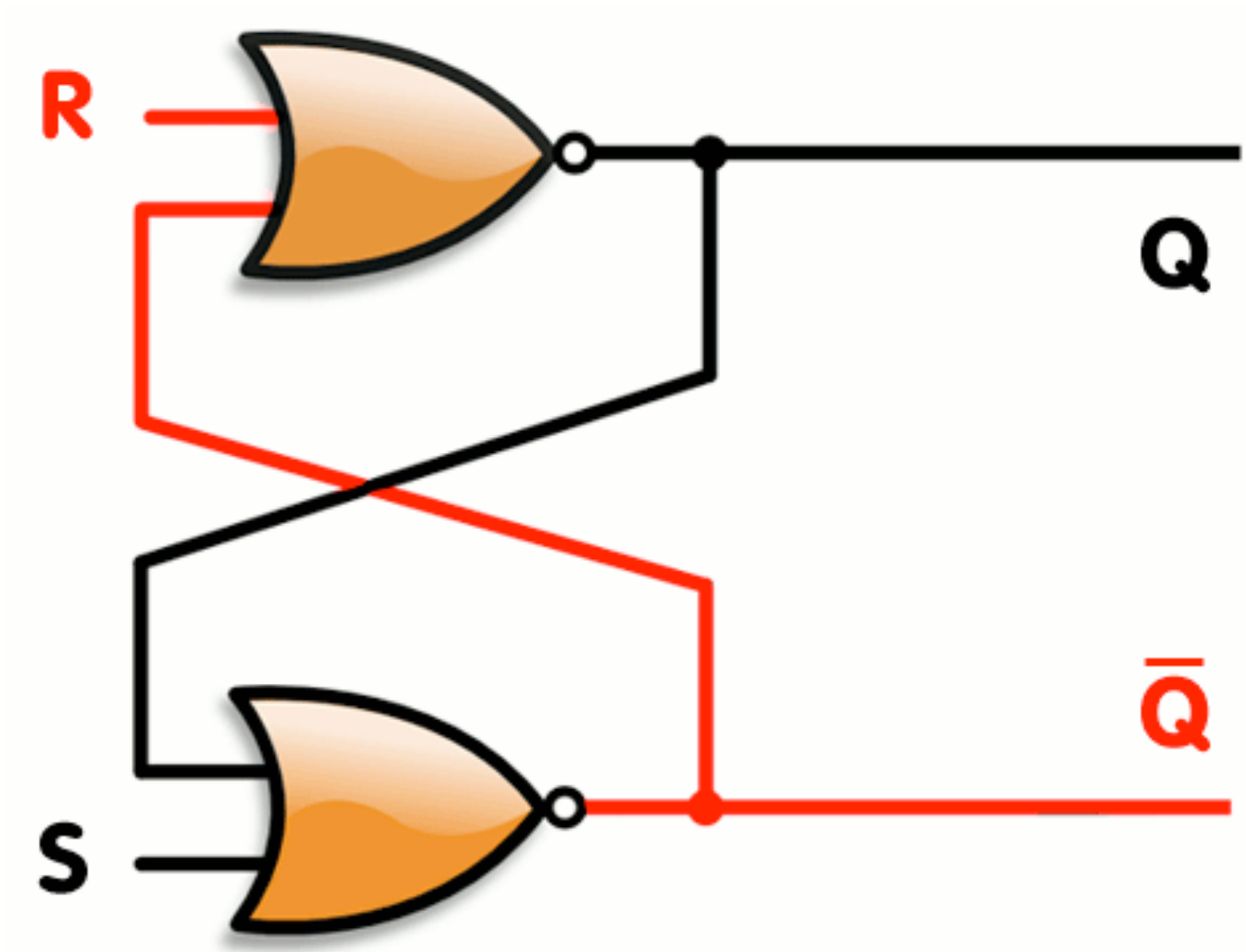
# The USA as a graph:

- Neighboring states are connected by edges.

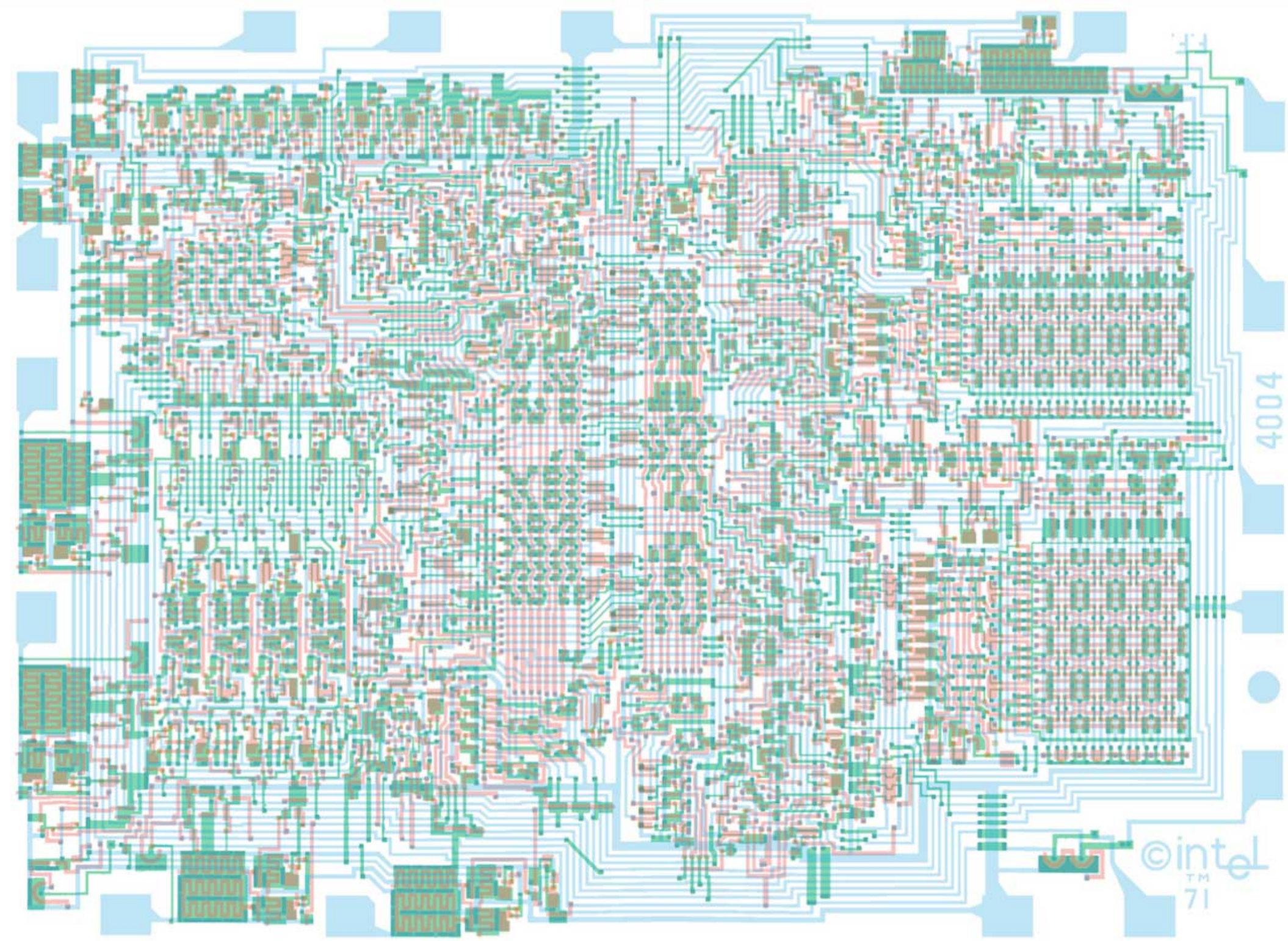




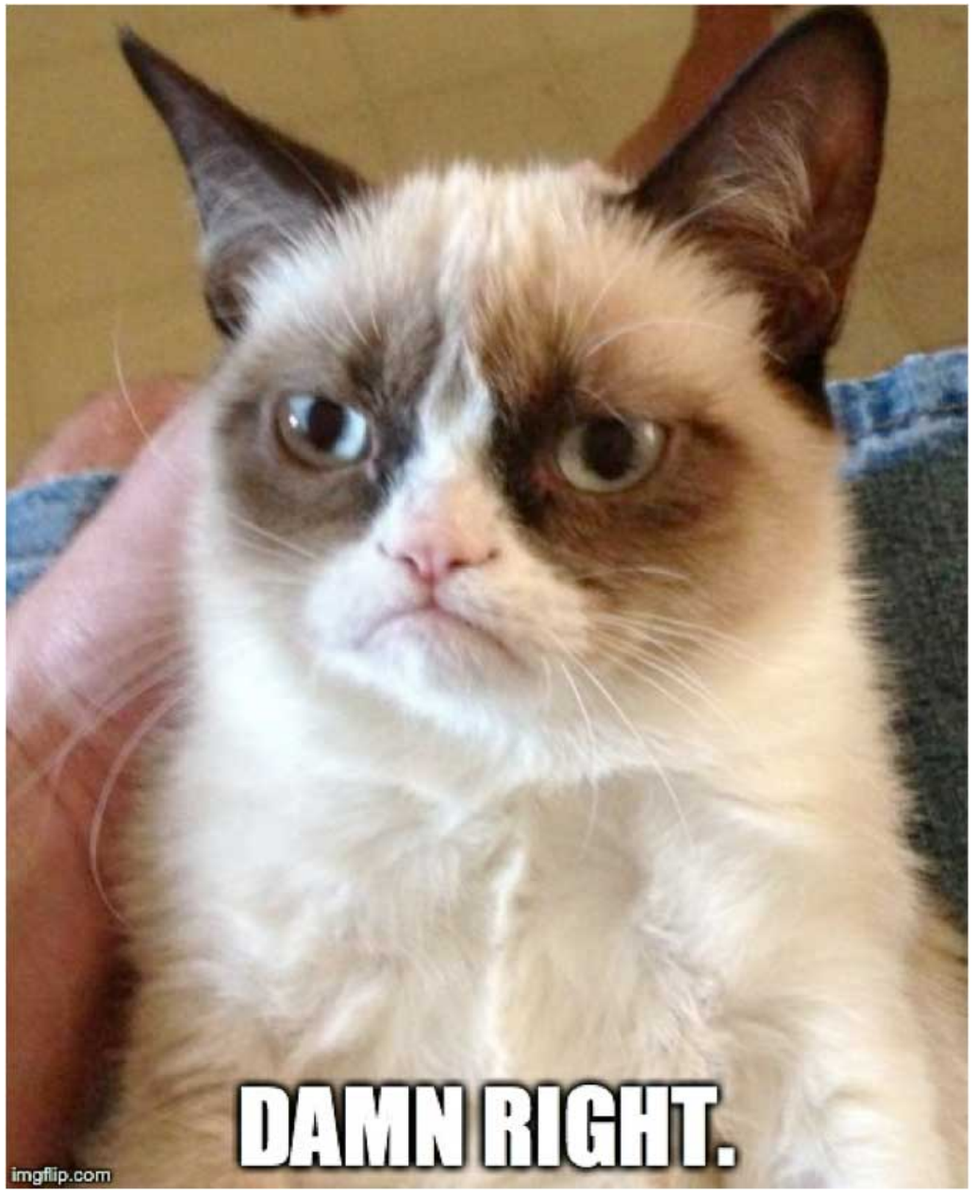
# Electrical circuit



# A bigger electrical circuit

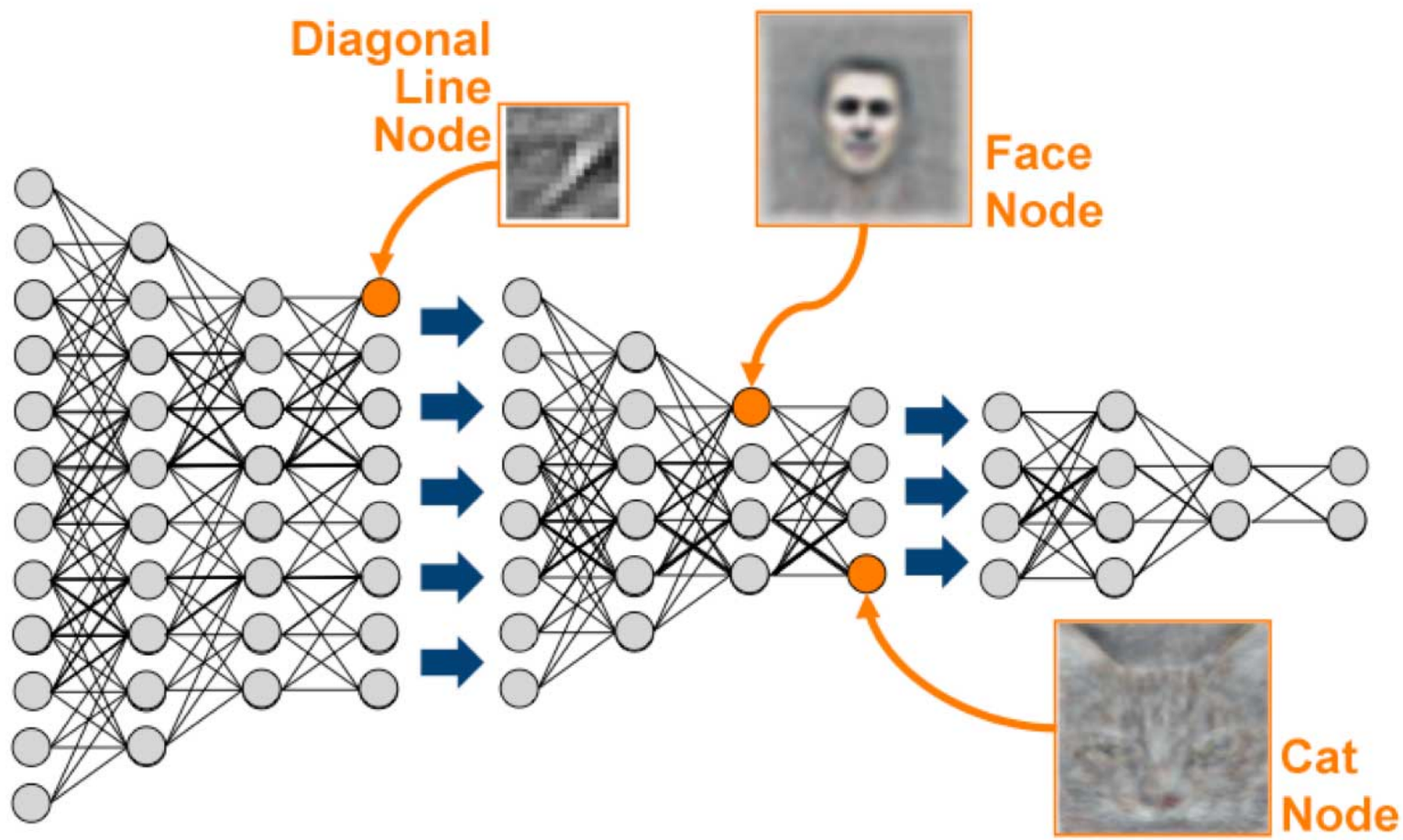


# This is not a graph:



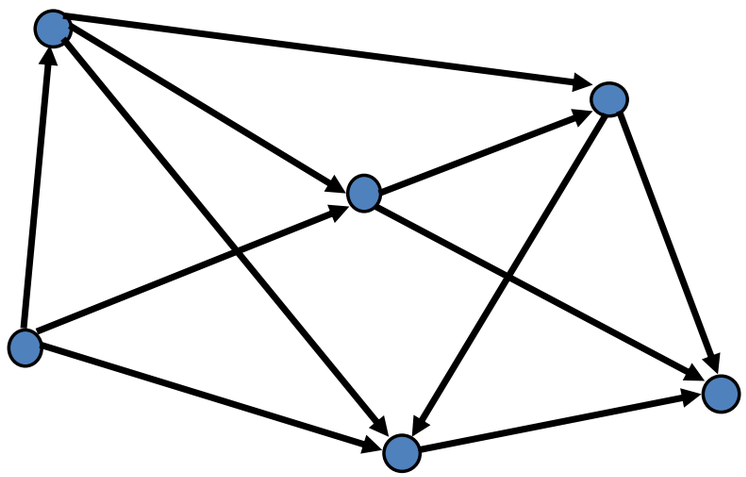

# it is a cat.

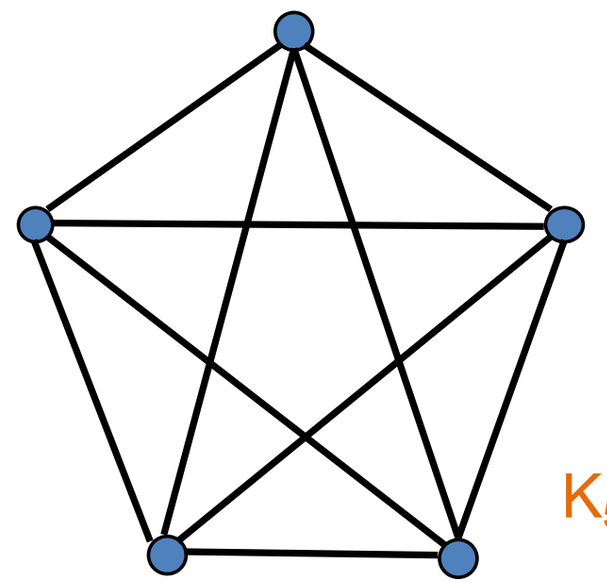
# This is a graph



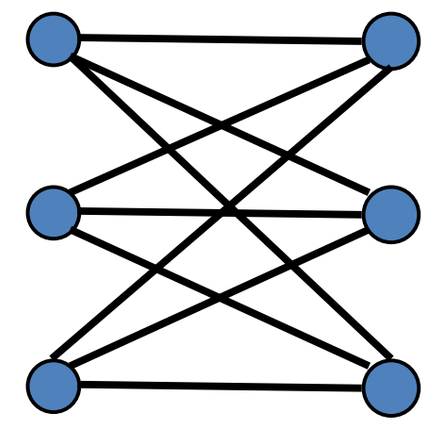

# that can recognize cats.

# Graphs: Abstract View

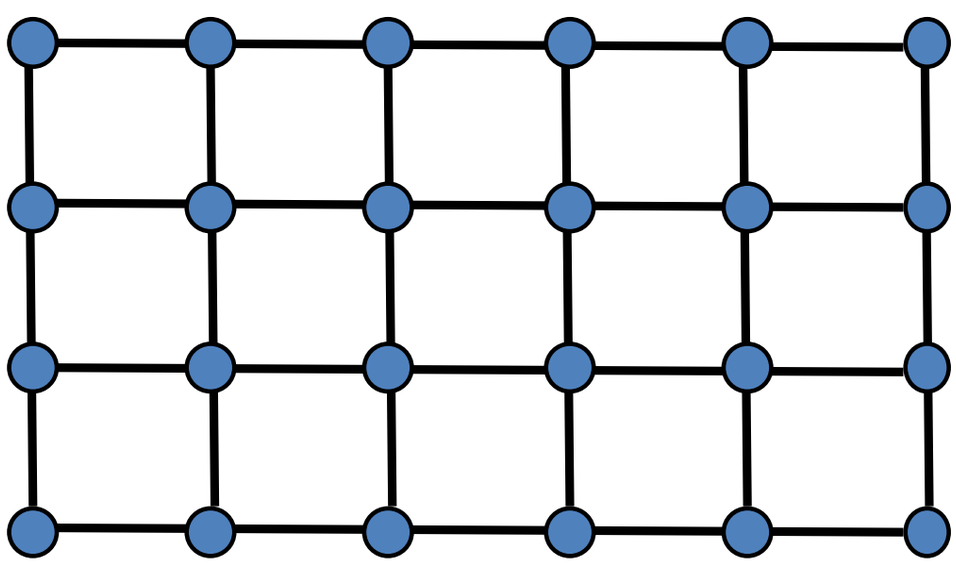


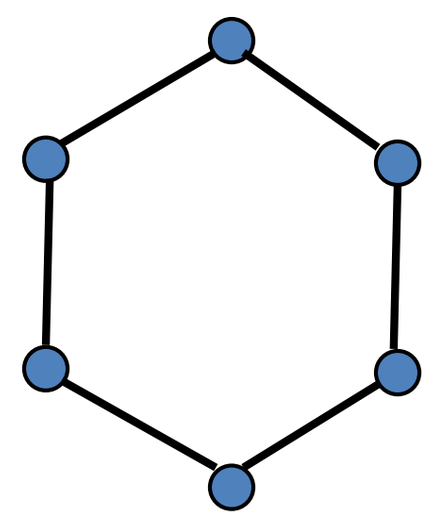


$K_5$



$K_{3,3}$



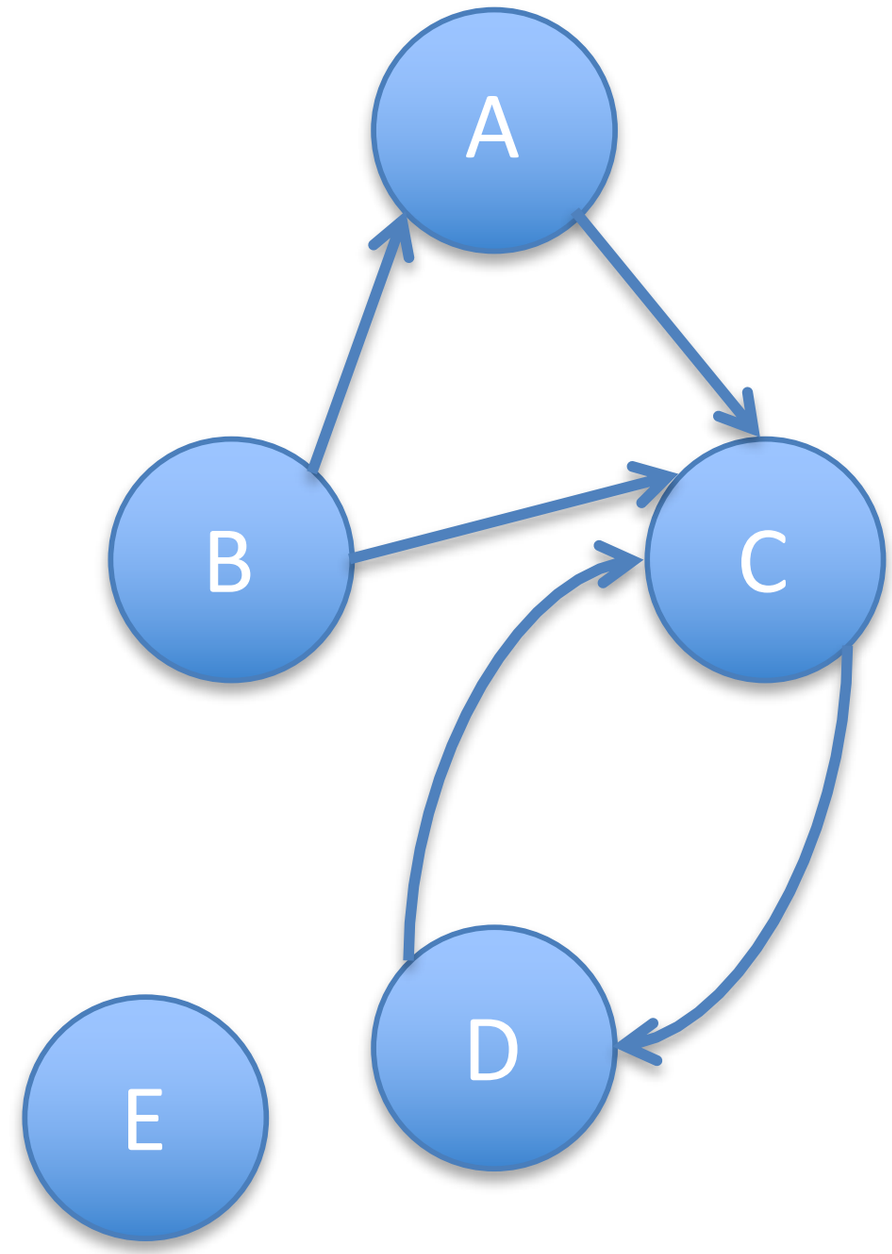


# Graphs, Formally

- 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): $u \neq v$ (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

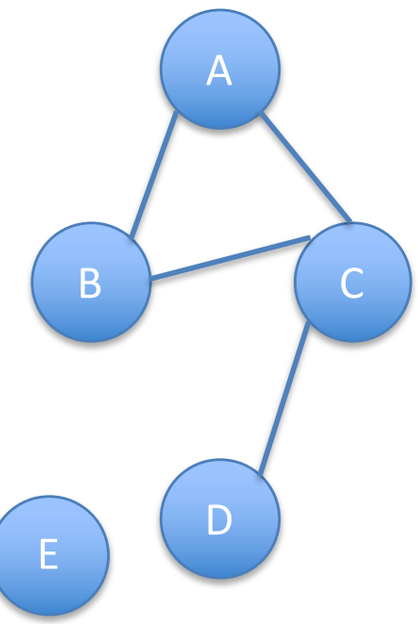
$$\boldsymbol{V} = \{A, B, C, D, E\}$$
$$\boldsymbol{E} = \{(A, C), (B, A), (B, C), (C, D), (D, C)\}$$
$$|\boldsymbol{V}| = 5$$
$$|\boldsymbol{E}| = 5$$



# 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

$$\boldsymbol{V} = \{A, B, C, D, E\}$$
$$\boldsymbol{E} = \{\{A,C\}, \{B,A\}, \{B,C\}, \{C,D\}\}$$
$$|\boldsymbol{V}| = 5$$
$$|\boldsymbol{E}| = 4$$



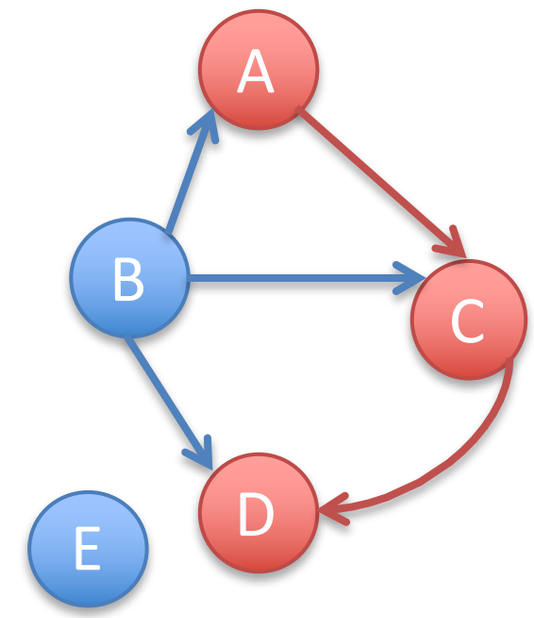

- An **un**directed graph can be converted to an equivalent **directed** graph:
  - Replace each undirected edge with two directed edges in opposite directions
- A **directed** graph can't always be converted to an **undirected** graph.

# 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.