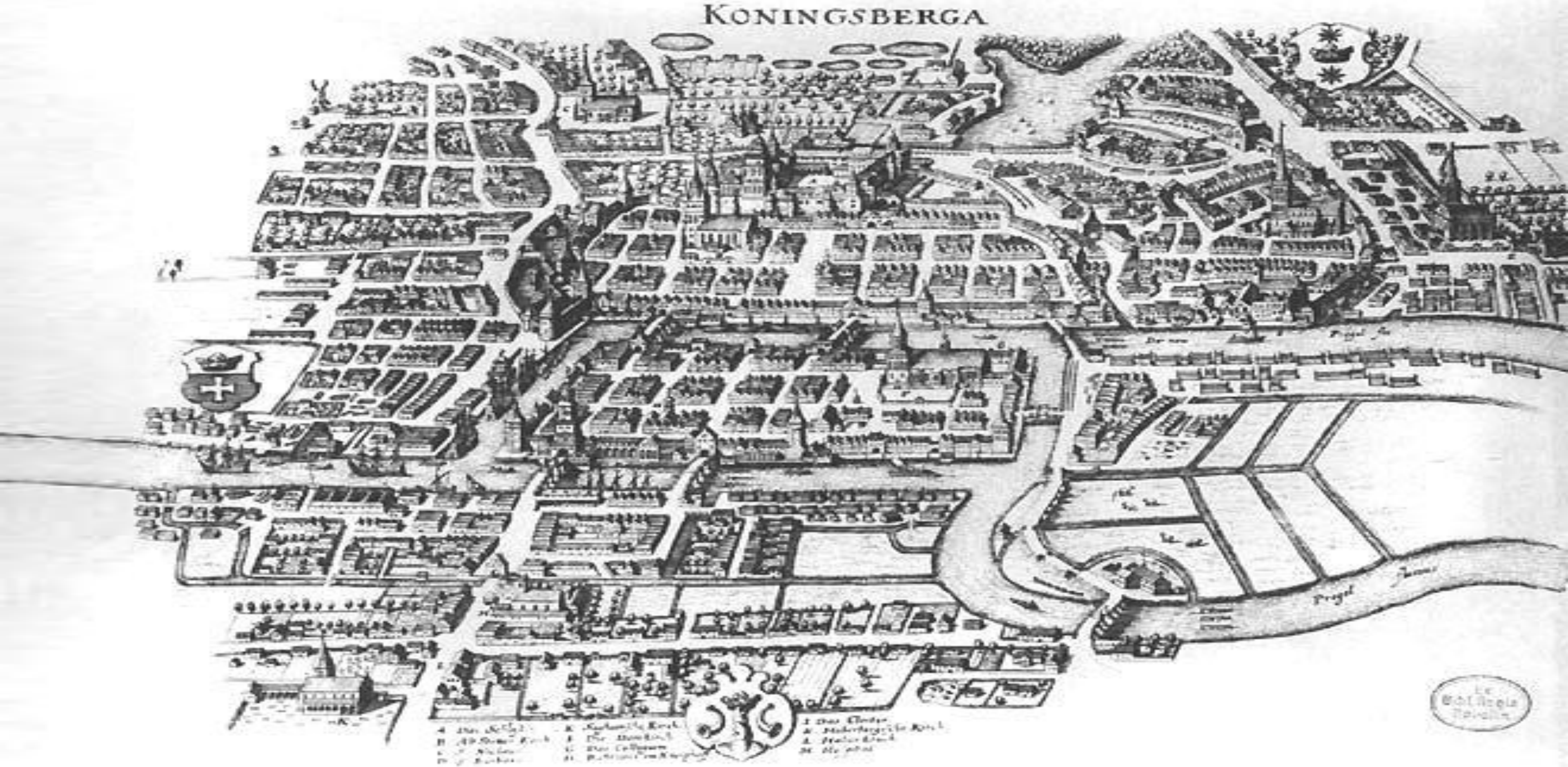
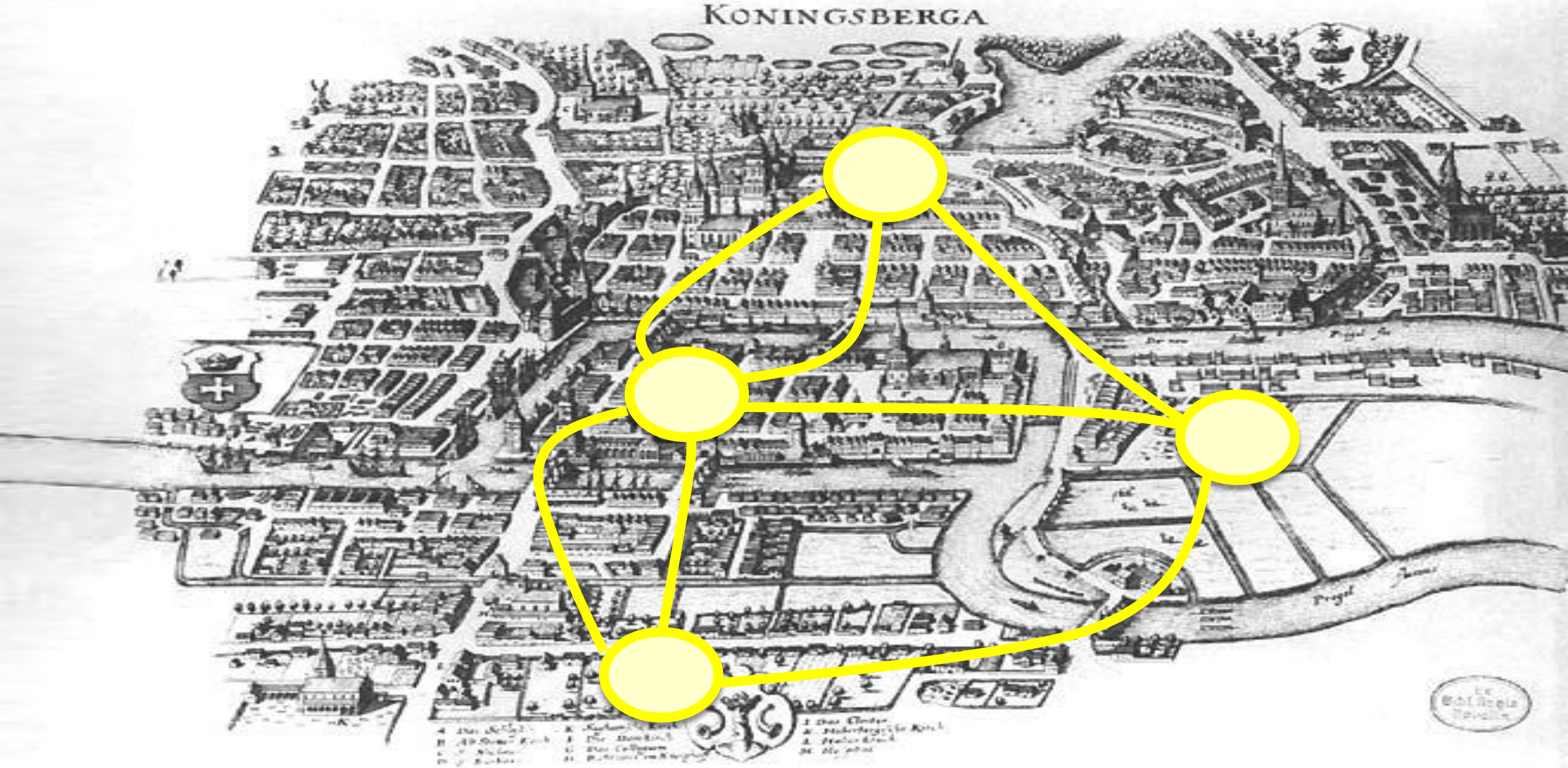
# Object-oriented programming and data-structures

## CS/ENGRD 2110
## SUMMER 2018

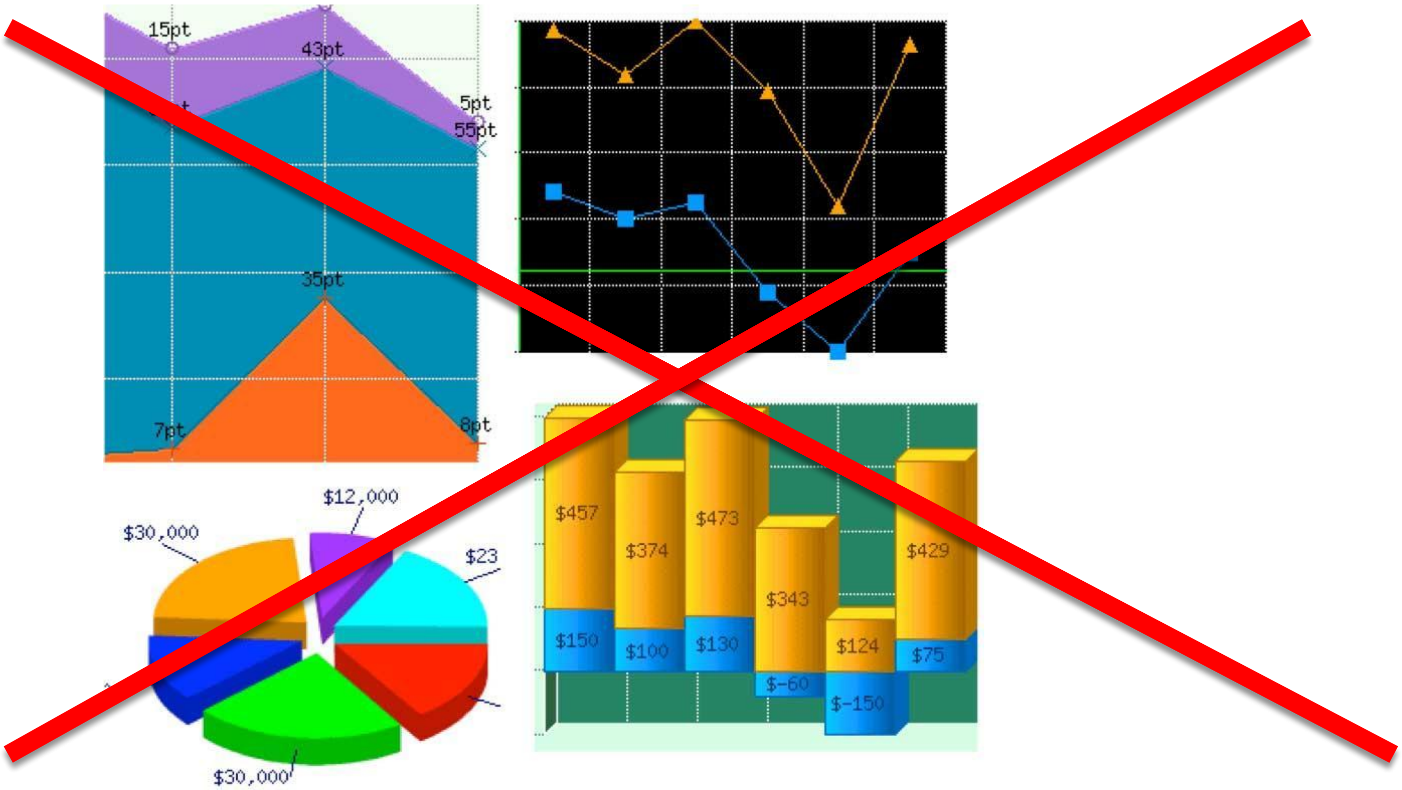
Lecture 11: Graphs

http://courses.cs.cornell.edu/cs2110/2018su


KONINGSBERGA

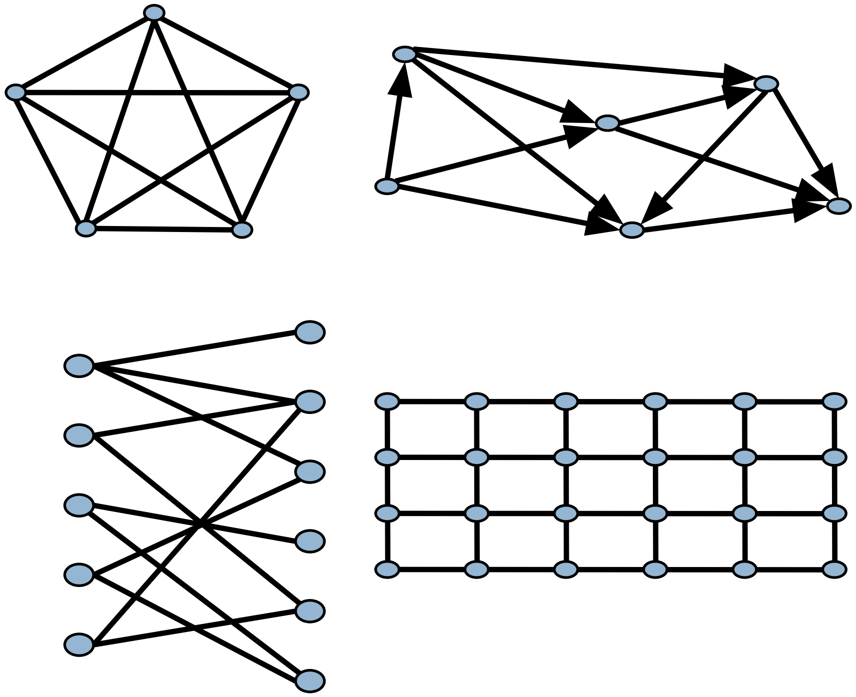

KONINGSBERGA

# These aren't the graphs we're looking for



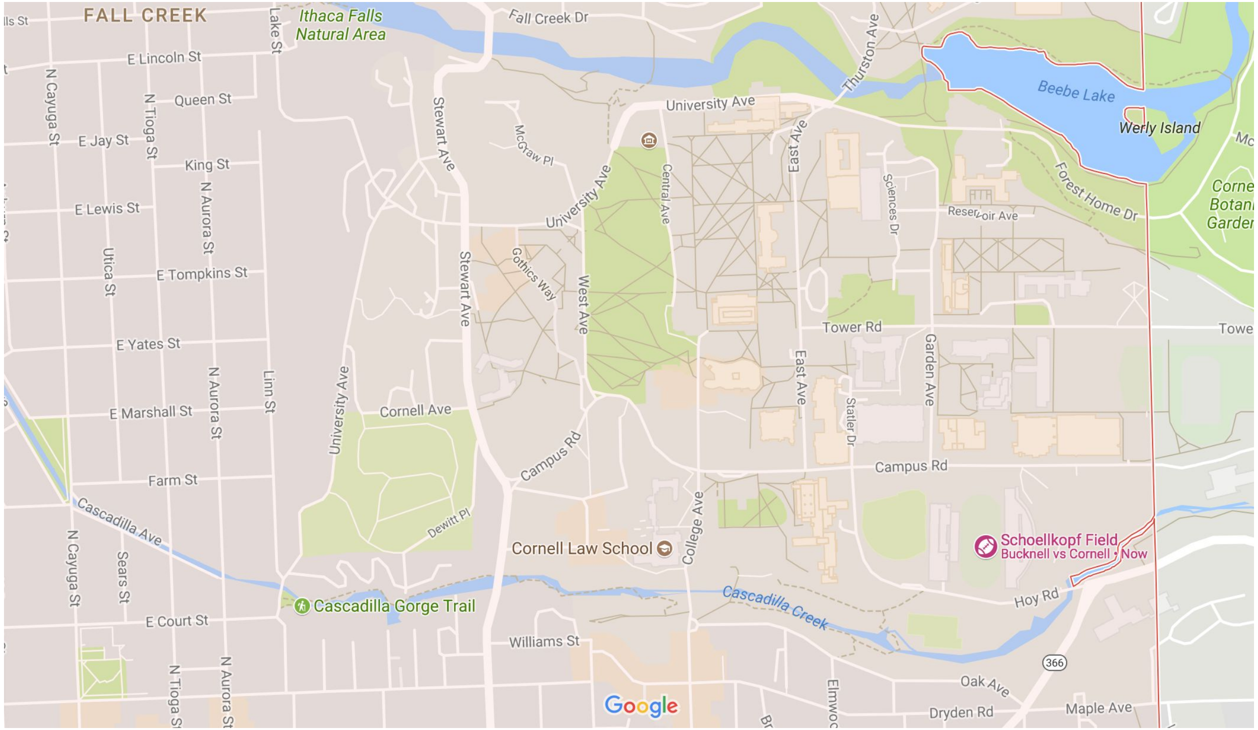
# Graphs

- A graph is a data structure
- A graph has
  - a set of vertices
  - a set of edges between vertices
- Graphs are a generalization of trees

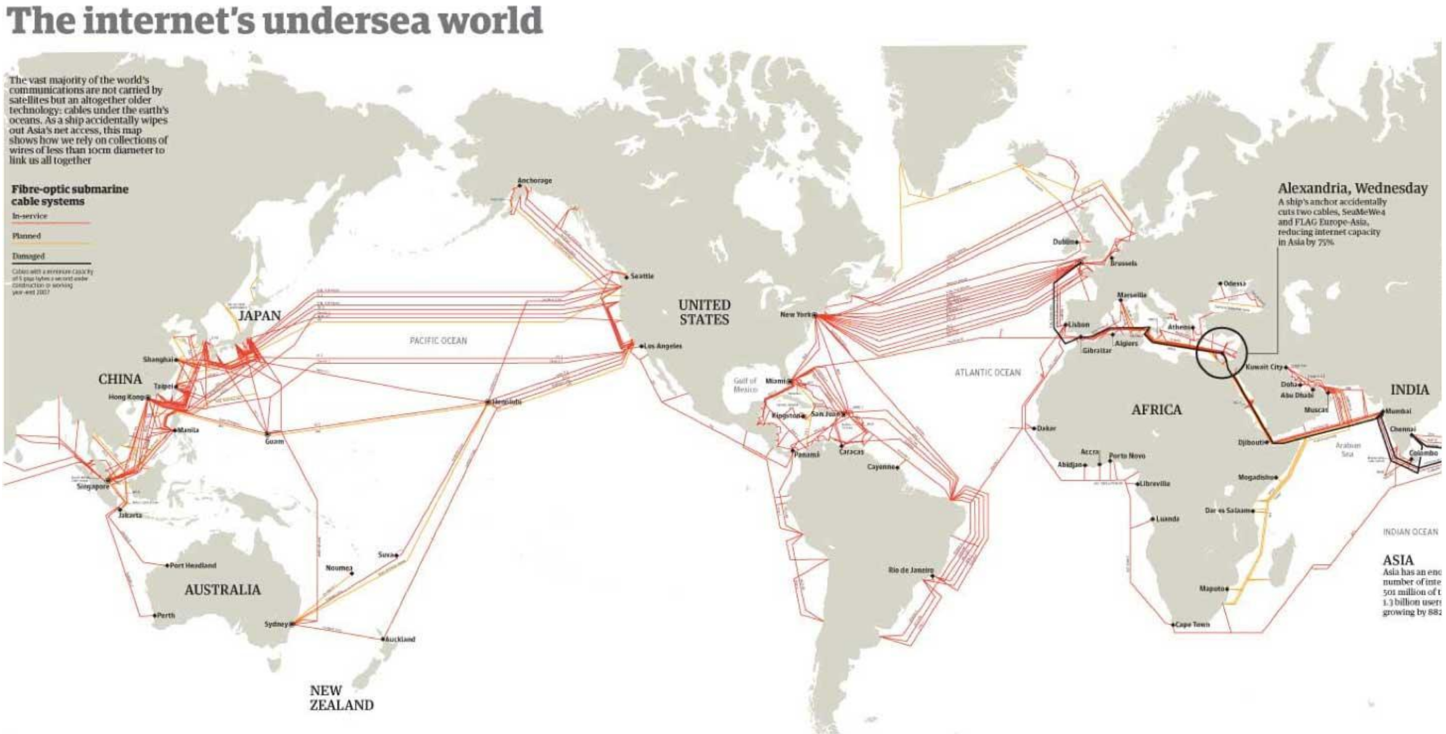


# This is a graph

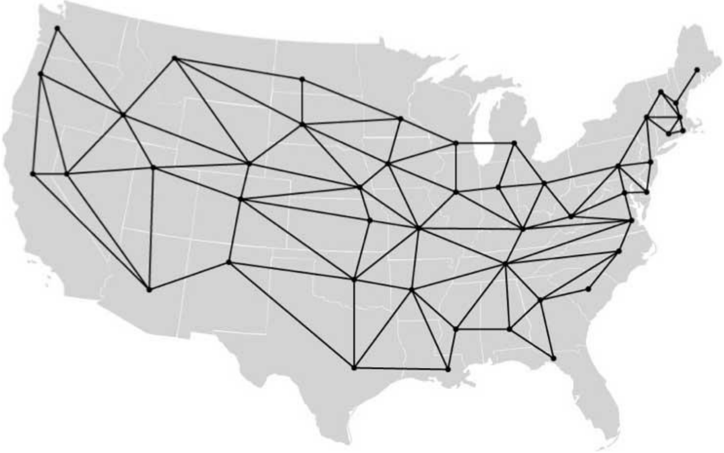
# Another transport graph

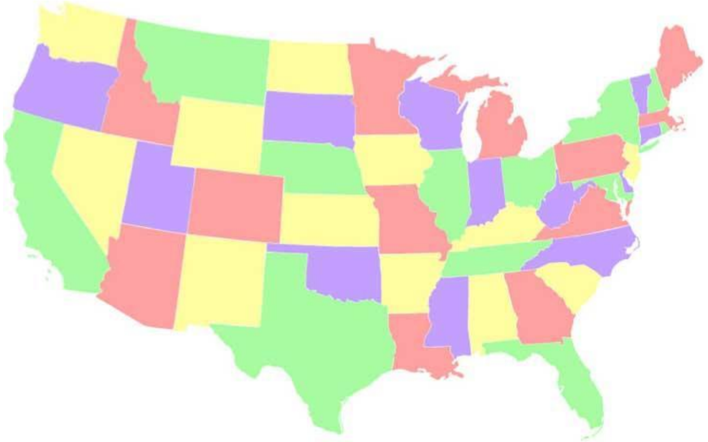


# This is a graph



# Viewing the map of states as a graph



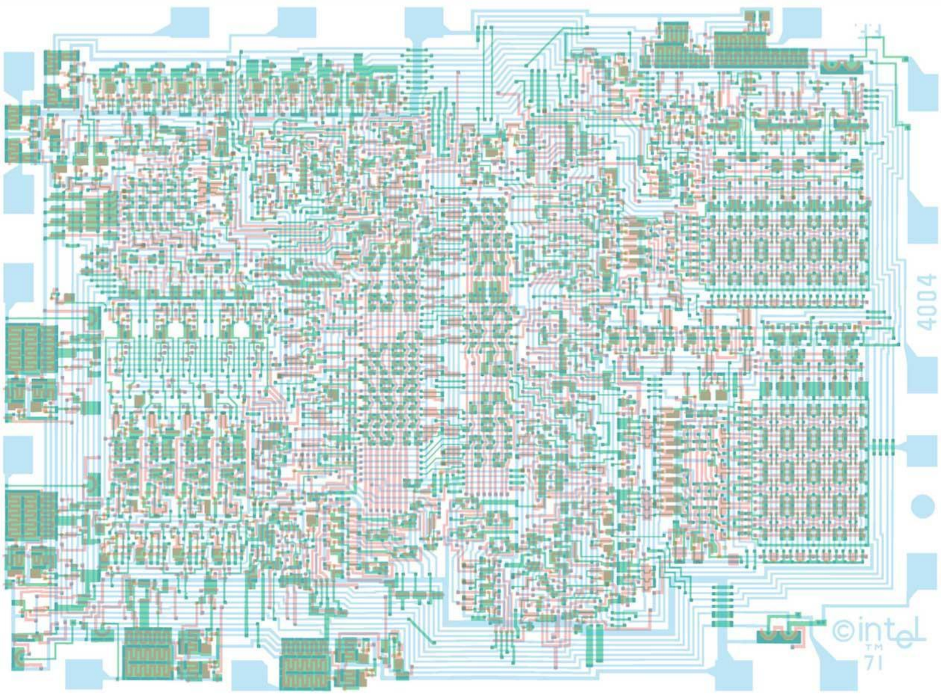


http://www.cs.cmu.edu/~bryant/boolean/maps.html

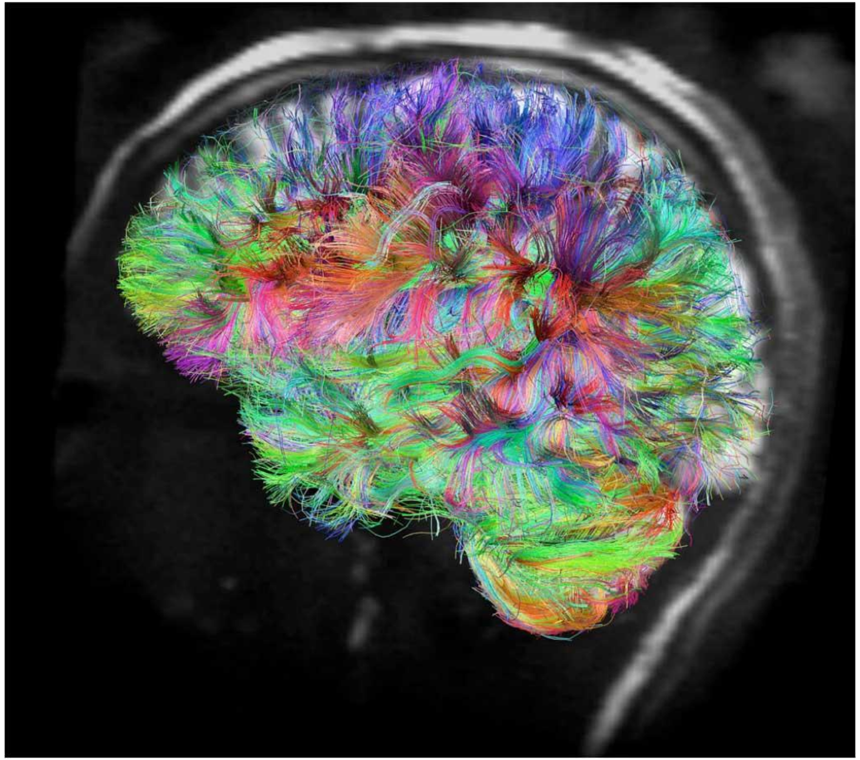
Each state is a point on the graph, and neighboring states are connected by an edge.

Do the same thing for a map of the world showing countries

# A circuit graph (Intel 4004)

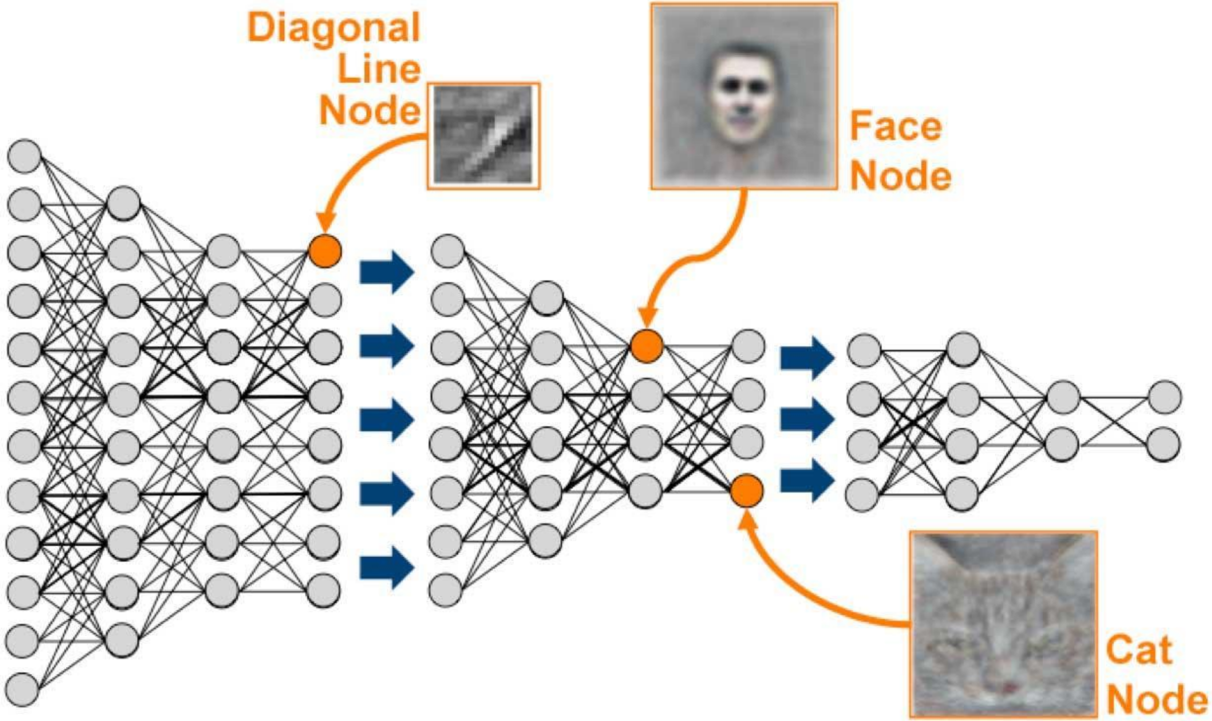


# This is a graph

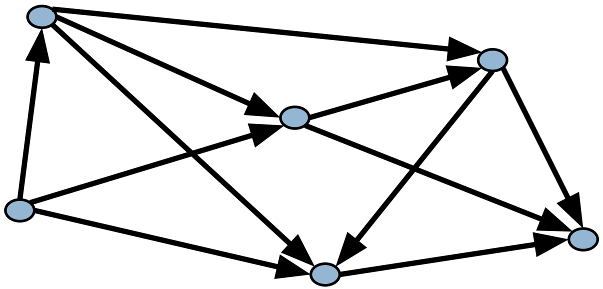


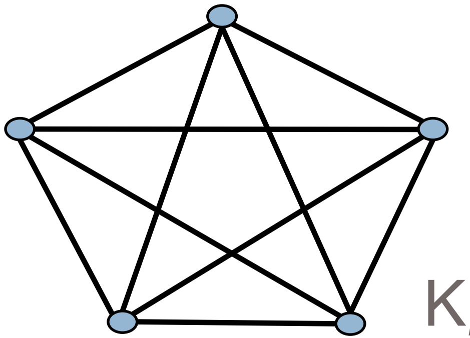
V.J. Wedeen and L.L. Wald, Martinos Center for Biomedical Imaging at

# This is a graph(ical model) that has learned to recognize cats

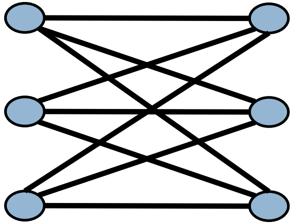


# Graphs

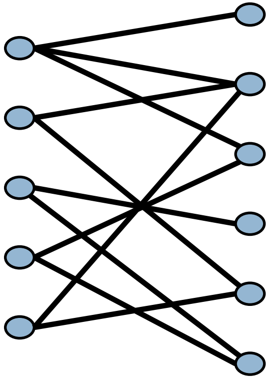


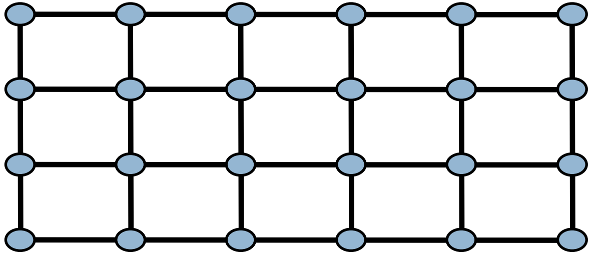


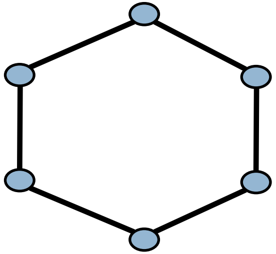
$K_5$



$K_{3,3}$







# Undirected graphs

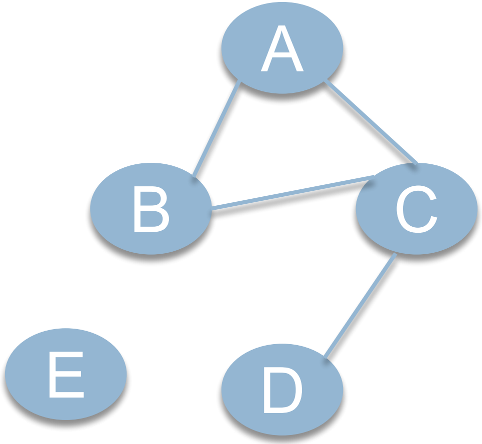
- A undirected graph is a pair $(V, E)$ where
  - $V$ is a (finite) set
  - $E$ is a set of pairs $(u, v)$ where $u,v \in V$
    - Often require $u \neq v$ (i.e. no self-loops)

- Element of $V$ is called a vertex or node
- Element of $E$ is called an edge or arc

- $|V|$ = size of $V$, often denoted by $n$
- $|E|$ = size of $E$, often denoted by $m$

# Undirected graphs

- A undirected graph is a pair ($V$, $E$) where
  - $V$ is a (finite) set
  - $E$ is a set of pairs ($u$, $v$) where $u,v \in V$
    - Often require $u \neq v$ (i.e. no self-loops)




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

$|V| = 5$
$|E| = 4$

- Element of $V$ is called a vertex or node
- Element of $E$ is called an edge or arc

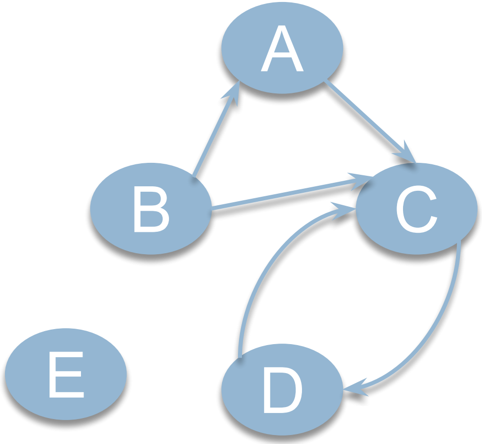
- $|V|$ = size of $V$, often denoted by $n$
- $|E|$ = size of $E$, often denoted by $m$

# Directed graphs

- A directed graph (digraph) is a lot like an undirected graph
  - $V$ is a (finite) set
  - $E$ is a set of ***ordered*** pairs $(u, v)$ where $u,v \in V$

- Every undirected graph can be easily converted to an equivalent directed graph via a simple transformation:
  - Replace every undirected edge with two directed edges in opposite directions

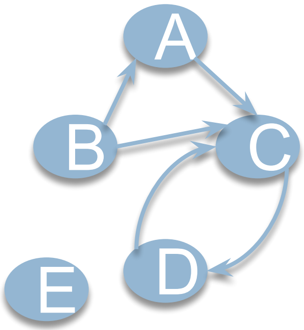
- … but not vice versa



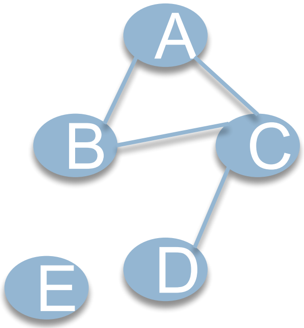

$\boldsymbol{V} = \{A, B, C, D, E\}$
$\boldsymbol{E} = \{(A, C), (B, A),$
$(B, C), (C, D),$
$(D, C)\}$
$|\boldsymbol{V}| = 5$
$|\boldsymbol{E}| = 5$

# Graph terminology

- Vertices $u$ and $v$ are called
  - the source and sink of the directed edge $(u, v)$, respectively
  - the endpoints of $(u, v)$ or $\{u, v\}$

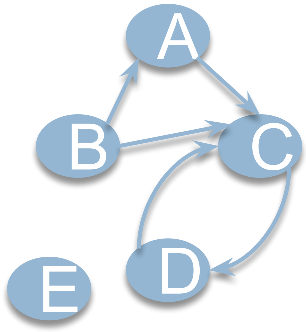



- Two vertices are adjacent if they are connected by an edge



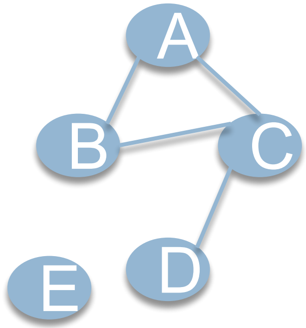
# Graph terminology

- The outdegree of a vertex $u$ in a directed graph is the number of edges for which $u$ is the source



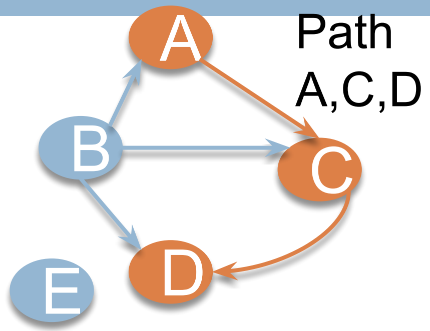

- The indegree of a vertex $v$ in a directed graph is the number of edges for which $v$ is the sink

- The degree of a vertex $u$ in an undirected graph is the number of edges of which $u$ is an endpoint

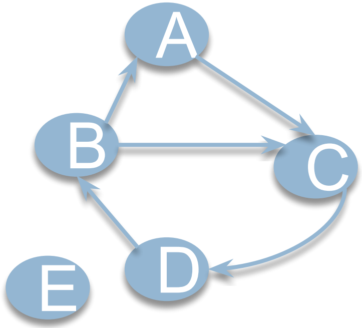


# More graph terminology

- A path is a sequence $v_0, v_1, v_2, \ldots, v_p$ of vertices such that for $0 \leq i < p$,
  - $(v_i, v_{i+1}) \in E$ if the graph is directed
  - $\{v_i, v_{i+1}\} \in E$ if the graph is undirected
- The length of a path is its number of edges
- A path is simple if it doesn't repeat any vertices



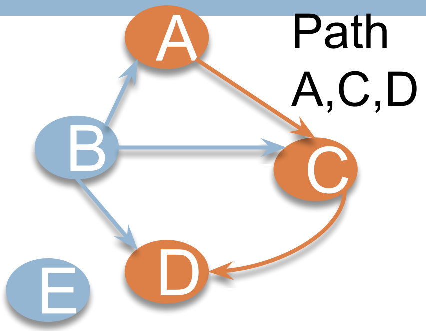
DAG



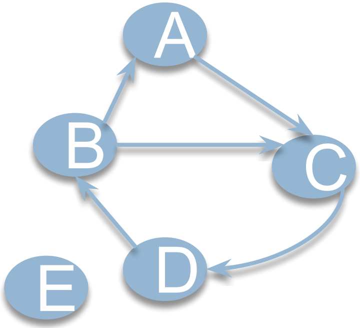
Not a DAG

# More graph terminology

- A cycle is a path $v_0, v_1, v_2, \ldots, v_p$ such that $v_0 = v_p$
- A cycle is simple if it does not repeat any vertices except the first and last
- A graph is acyclic if it has no cycles
- A ***d***irected ***a***cyclic ***g***raph is called a DAG




DAG

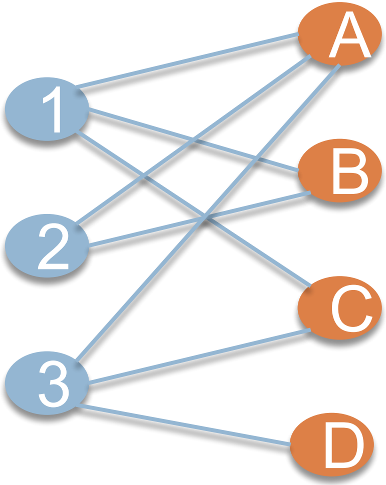



Not a DAG

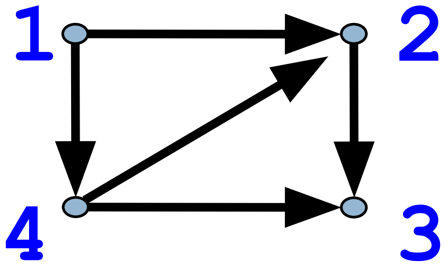
# Bipartite graphs

- A directed or undirected graph is bipartite if the vertices can be partitioned into two sets such that no edge connects two vertices in the same set

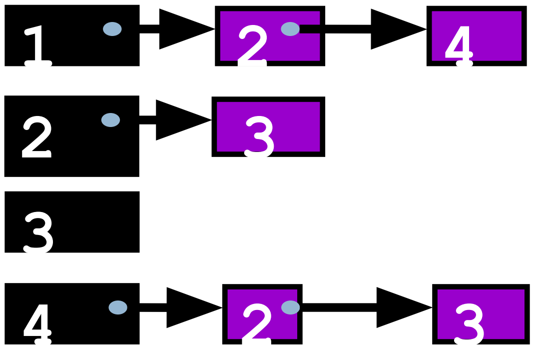
- The following are equivalent
  - *G* is bipartite
  - *G* is 2-colorable
  - *G* has no cycles of odd length



# Representations of graphs




Adjacency List




Adjacency Matrix

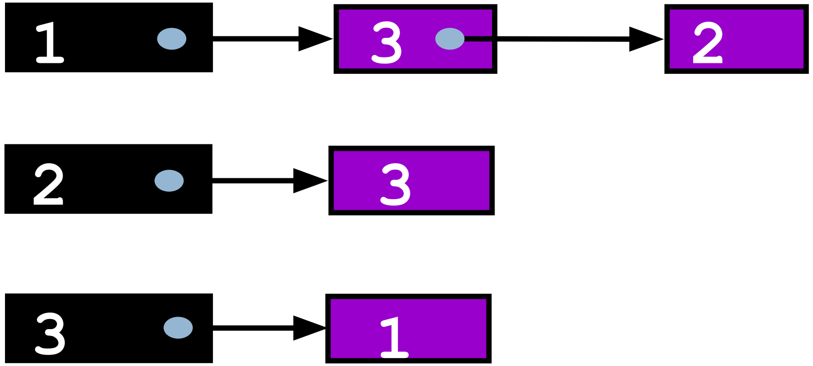
| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | 0 |

# Graph Quiz

Which of the following two graphs are DAGs?

Directed Acyclic Graph

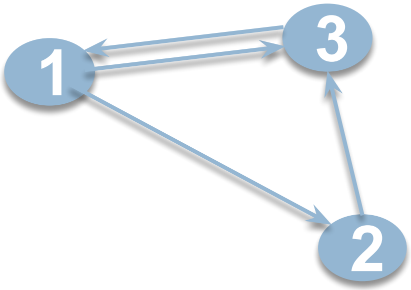
Graph 1:

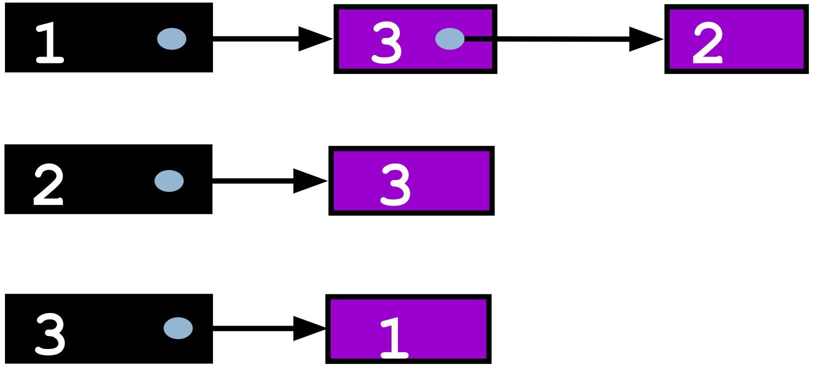


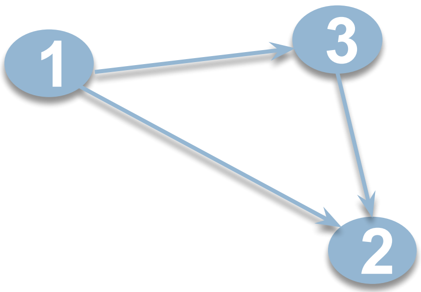

Graph 2:

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 2 | 0 | 0 | 0 |
| 3 | 0 | 1 | 0 |

# Graph Quiz




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

# Adjacency matrix or adjacency list?

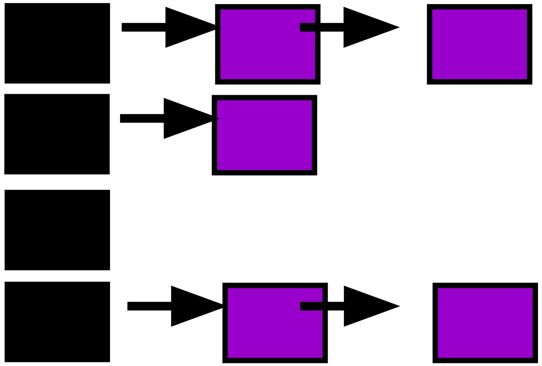
- $v$ = number of vertices
- $e$ = number of edges
- $d(u)$ = degree of $u$ = no. edges leaving $u$

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

- Adjacency Matrix
  - Uses space $O(v^2)$
  - Enumerate all edges in time $O(v^2)$
  - Answer “Is there an edge from $u1$ to $u2$?” in $O(1)$ time
  - Better for dense graphs (lots of edges)

# Adjacency matrix or adjacency list?

- $v$ = number of vertices
- $e$ = number of edges
- $d(u)$ = degree of $u$ = no. edges leaving $u$



- Adjacency List
  - Uses space $O(v + e)$
  - Enumerate all edges in time $O(v + e)$
  - Answer "Is there an edge from $u1$ to $u2$?" in $O(d(u1))$ time
  - Better for sparse graphs (fewer edges)

# What can we do on graphs?

- Search
  - Depth-first search
  - Breadth-first search
- Shortest paths
  - Dijkstra's algorithm
- Minimum spanning trees
  - Jarnik/Prim/Dijkstra algorithm
  - Kruskal's algorithm