An undirected graph G consists of a set V of vertices and a set E of edges where each edge in E is associated with an unordered pair of vertices from V.

The degree of a vertex v is the number of edges incident to v.

If (u, v) is in E then u and v are adjacent.

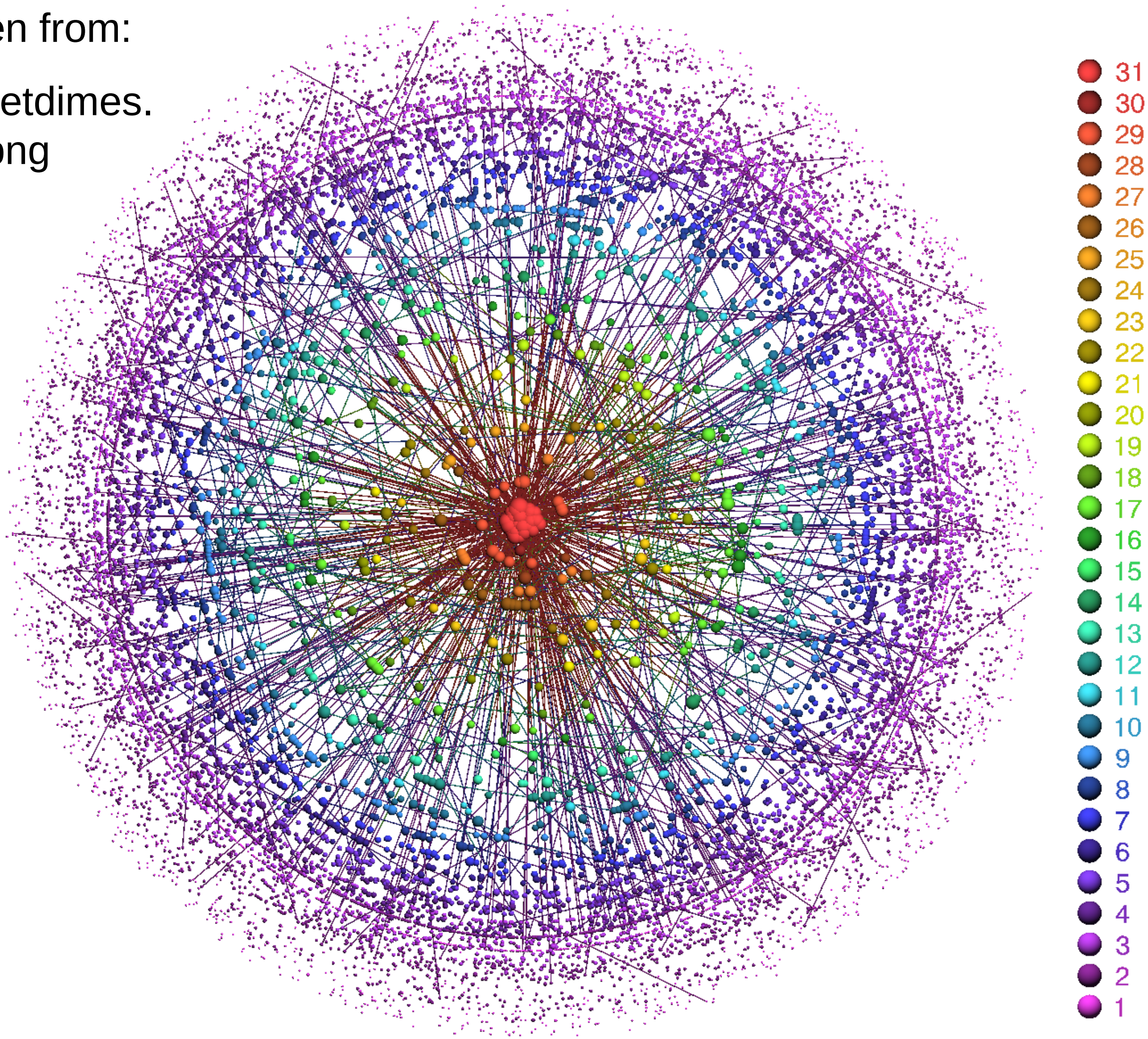
Text: uses less common notation {u,v} or uv for edges.

A simple graph has no loops or multiple edges.

Exercise: prove by induction that a simple graph G on n vertices has at most n(n-1)/2 edges.

Internet taken from:

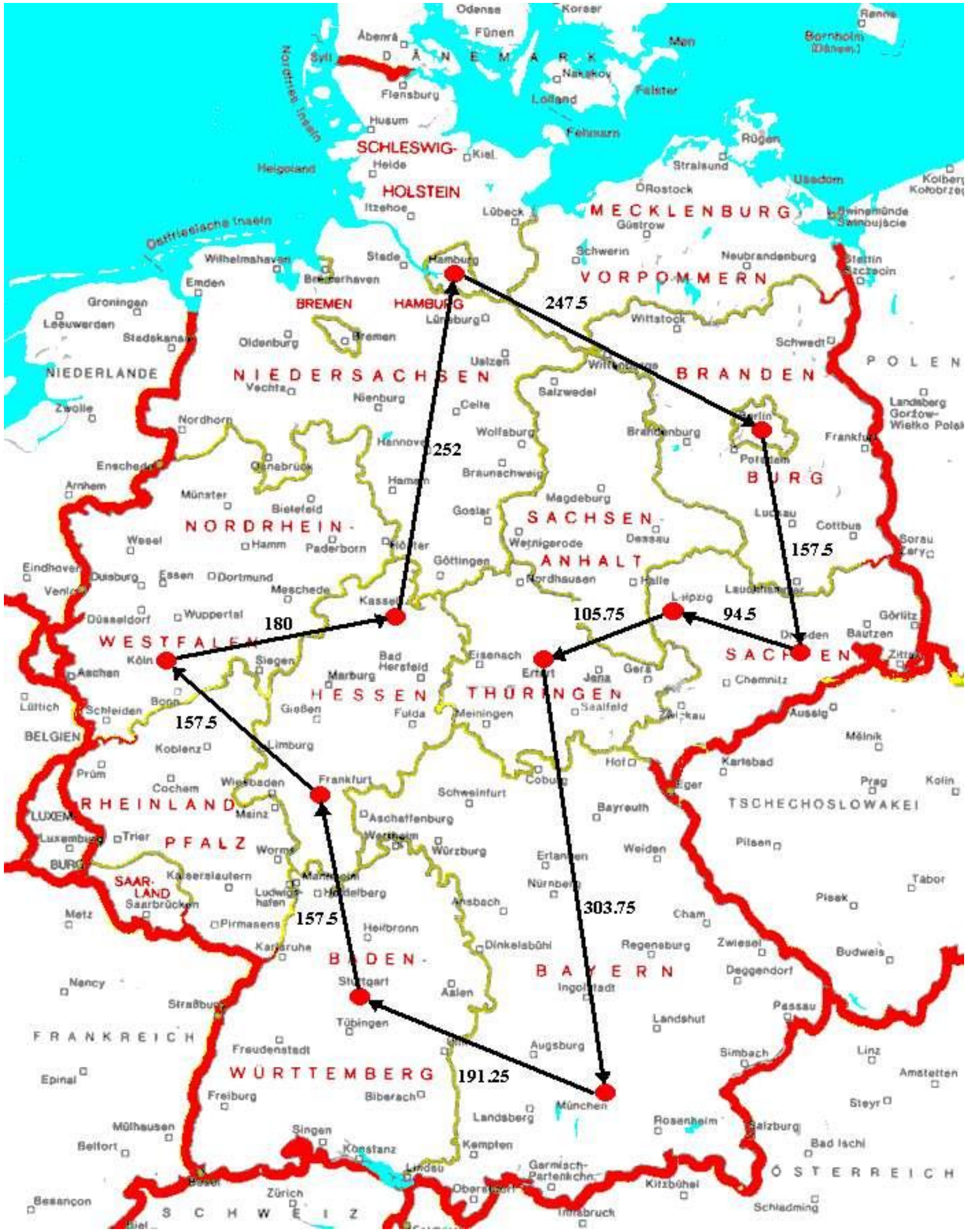
http://www.netdimes.org/asmap.png

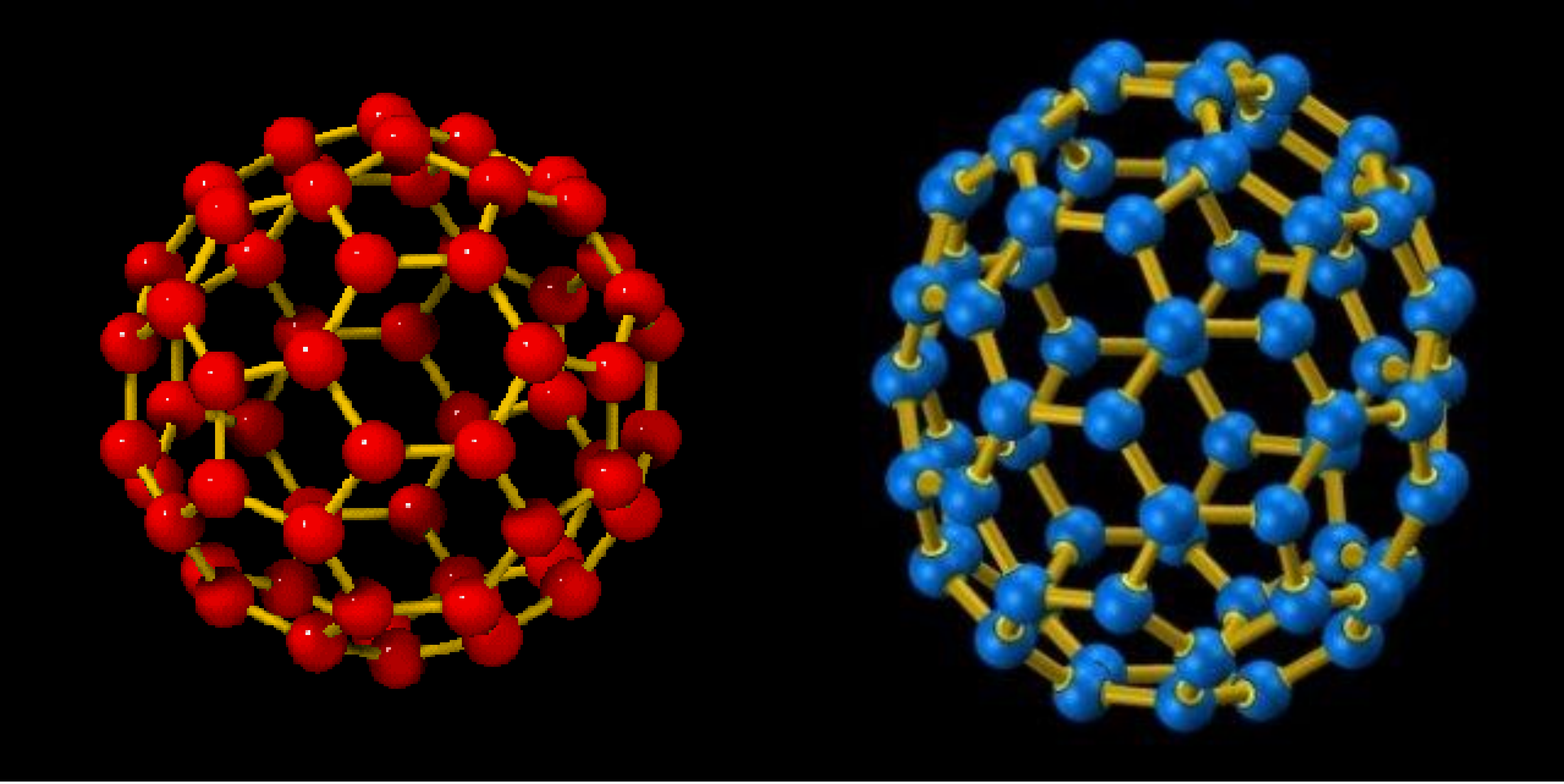


Travelling Salesman

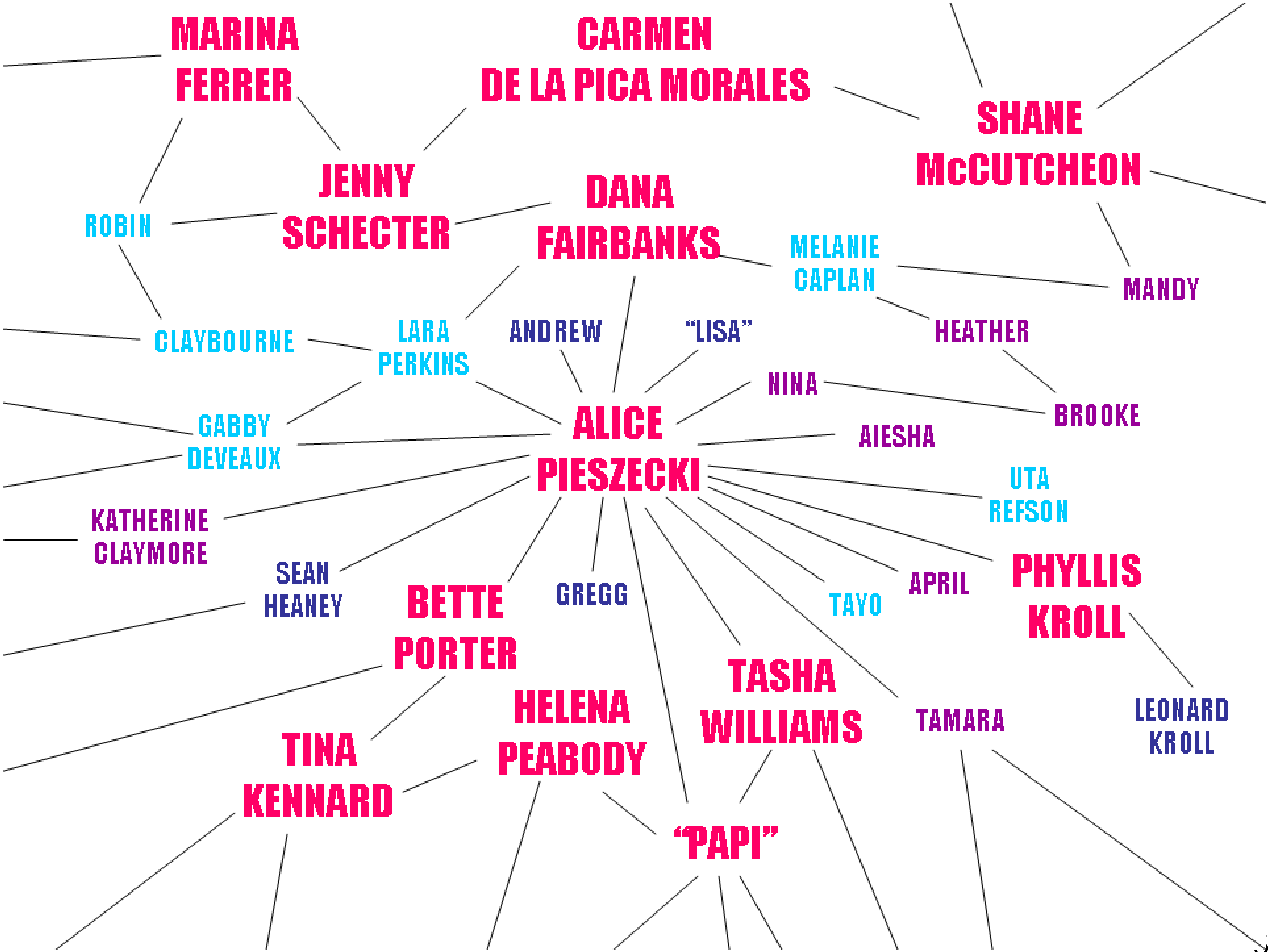
From:

Ehsan Moeinzadeh
Guildford, Surrey,
United Kingdom





Graphs representing chemical molecules


MARINA FERRER
CARMEN DE LA PICA MORALES
SHANE McCUTCHEON
JENNY SCHECTER
DANA FAIRBANKS
ROBIN
MELANIE CAPLAN
MANDY
CLAYBOURNE
LARA PERKINS
ANDREW
"LISA"
HEATHER
NINA
BROOKE
ALICE PIESZECKI
AIESHA
GABBY DEVEAUX
UTA REFSON
KATHERINE CLAYMORE
SEAN HEANEY
GREGG
APRIL
TAYO
PHYLLIS KROLL
BETTE PORTER
TASHA WILLIAMS
HELENA PEABODY
TAMARA
LEONARD KROLL
TINA KENNARD
"PAPI"

# Data Structures for Graphs

How can graphs be stored in the computer?

How does this affect the time complexity of algorithms for graphs?

A cycle of a graph is an alternating sequence of vertices and edges of the form $v_0$, $(v_0, v_1)$, $v_1$, $(v_1, v_2)$, $v_2$, $(v_2, v_3)$, … ,$v_{k-1}$, $(v_{k-1}, v_k)$, $v_k$ where except for $v_0 = v_k$ the vertices are distinct.

Exercise: define path, define connected.

A tree is a connected graph with no cycles.

A subgraph H of a graph G is a graph with $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.

H is spanning if $V(H) = V(G)$.

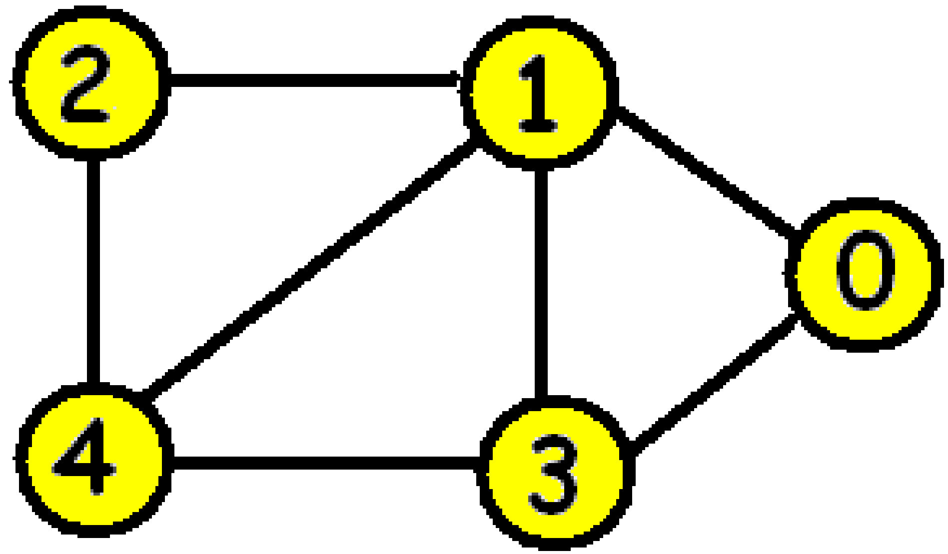
Spanning tree- spanning subgraph which is a tree.

# Strange Algorithms

Input: a graph G
Question: does G have a spanning tree?

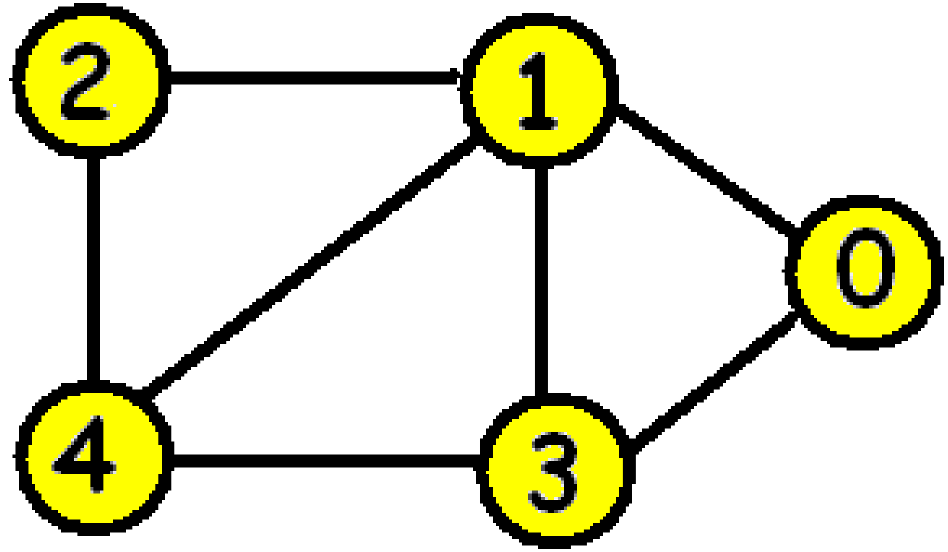
This can be answered by computing a determinant of a matrix and checking to see if it is zero or not.

For lower bound arguments, it is essential to not make too many assumptions about how an algorithm can solve a problem.

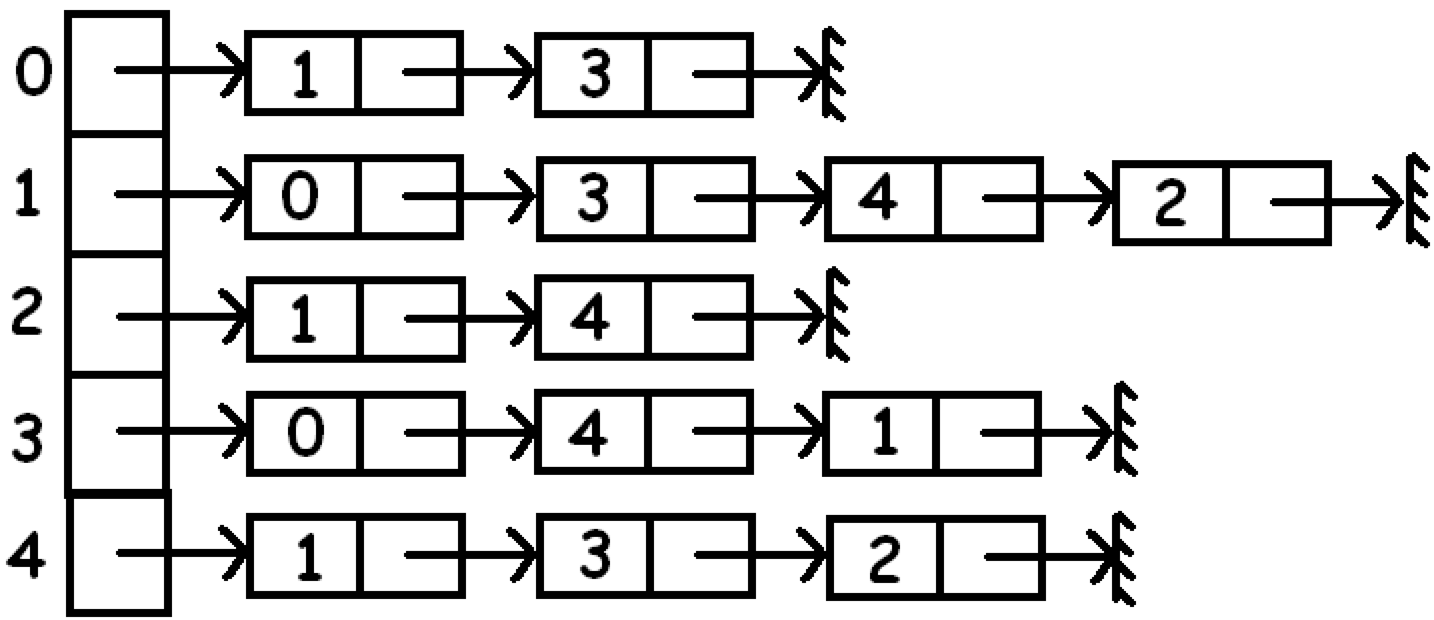



Adjacency matrix:

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




## Adjacency list:



## Adjacency lists:

Lists can be stored:

1. sorted,
2. in arbitrary order,
3. in some other specific order- for example a rotation system has the neighbours of each vertex listed in clockwise order in some planar embedding of a graph (a picture drawn on the plane with no edges crossing).

## Data structures for graphs:

n= number of vertices

m= number of edges

Adjacency matrix: Space $\Theta(n^2)$

Adjacency list: Space $\Theta(n + m)$

How long does it take to do these operations:

1. Insert an edge?
2. Delete an edge?
3. Determine if an edge is present?
4. Traverse all the edges of a graph?