In His name

Eulerian graph : contains cycle visiting all edges once. Every vertex degree is even
Hamilton graph :

- contains cycle visiting all vertices once ,
- Min vertex degree >= n/2 and order of graph n>=3
- |G| >= 3 and (chromatic number)k(G) >= a(G)(biggest independent set of vertices)
- 4-connected planar graph
- For every set S => G-S has at most S components
- Tutte graph is not hamiltonian

1) Barbell graph

- the ***n*-barbell graph** is a special type of undirected graph consisting of two non-overlapping *n*-vertex cliques together with a single edge that has an endpoint in each clique

2) Lollipop graph

Lollipop graph

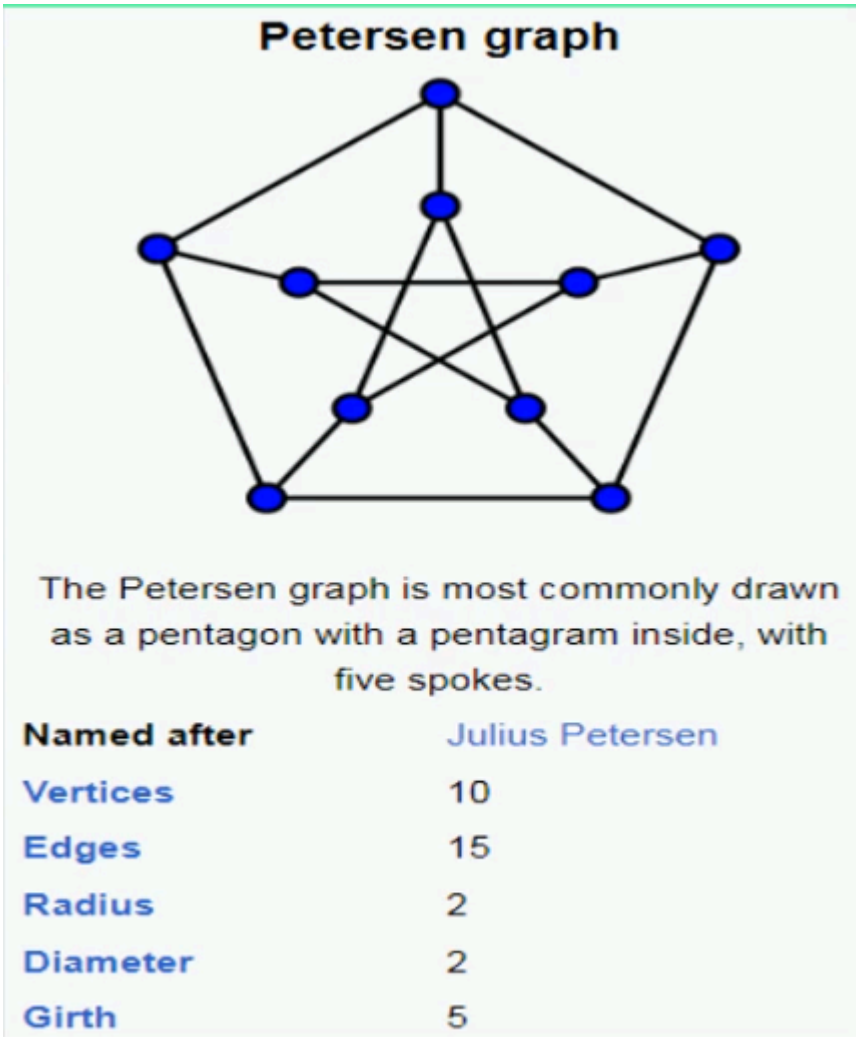
A *(8,4)*-lollipop graph

Vertices	$m+n$
Edges	$\binom{m}{2}+n$
Girth	$\begin{cases}\infty & m \le 2 \\ 3 & \text{otherwise}\end{cases}$
Properties	connected
Notation	$L_{m,n}$

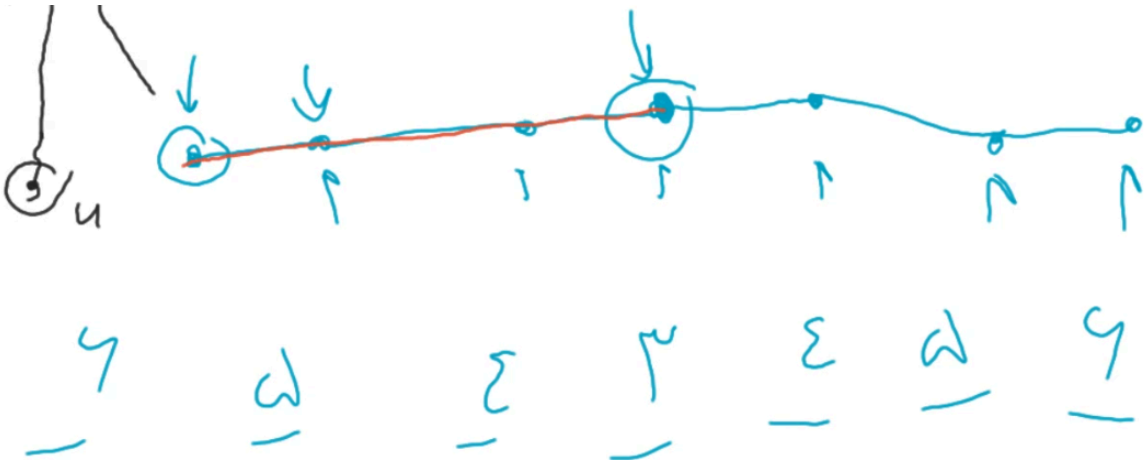
3) Tadpole graph

- the **(*m*,*n*)-tadpole graph** is a special type of graph consisting of a cycle graph on *m* (at least 3) vertices and a path graph on *n* vertices, connected with a bridge

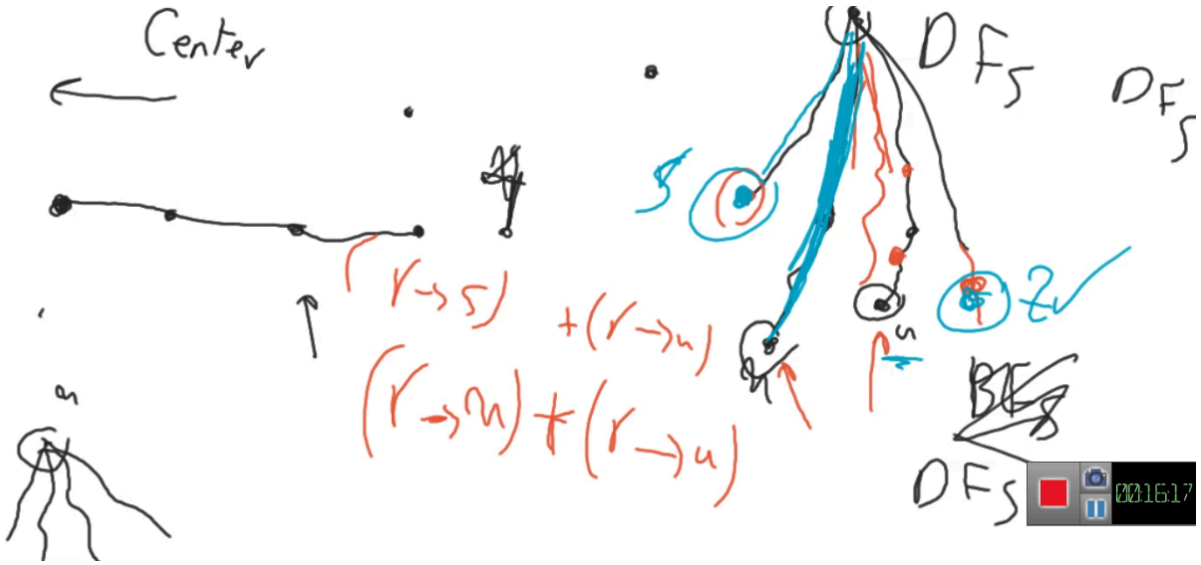
Cubic graph = 3-regular graph



Diameter = longest path in graph, radius = min max distance from all vertices
Center of graph and raduis of graph:

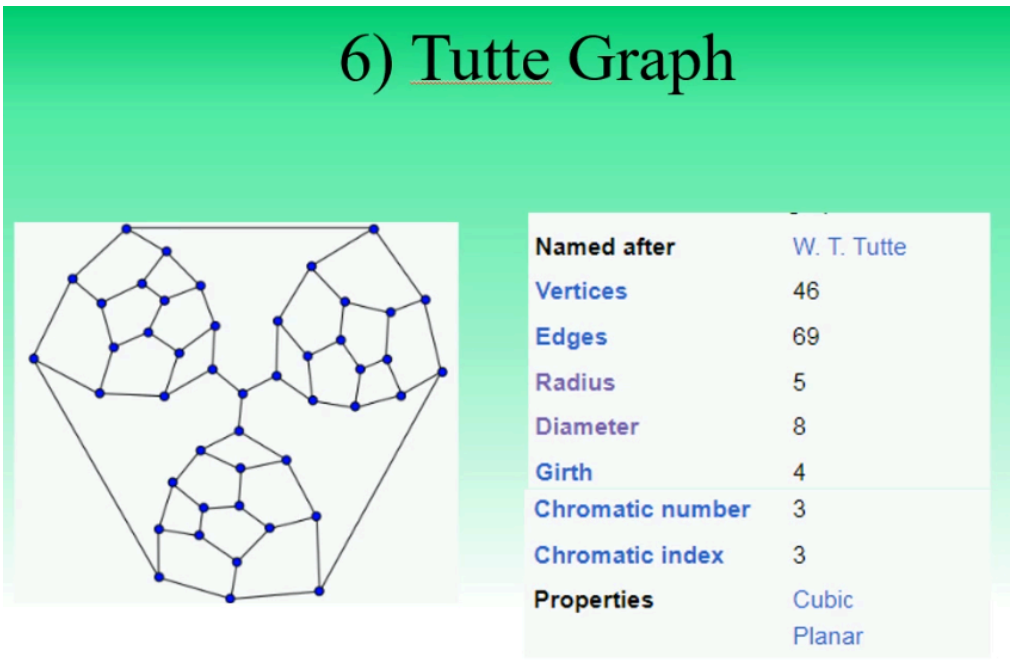


Diameter of graph



Integr Embedded => edge weight = real distance

Tutte => 3-regular



Bach Edge => edge from child to ancestor
Cross edge => from child to cousin (not ancestor , not on the same branch)

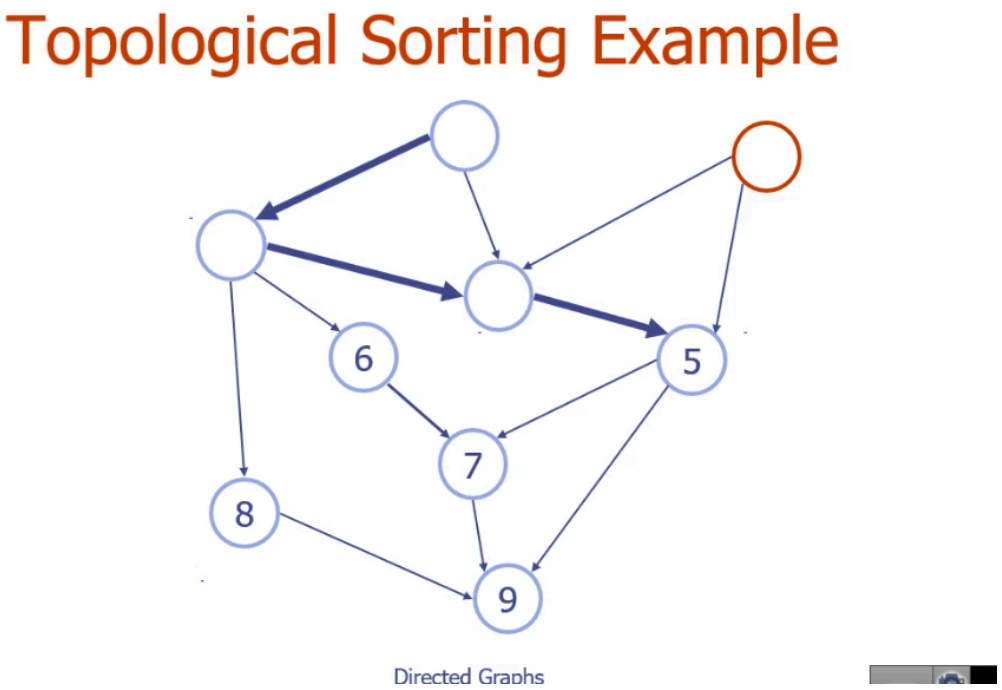
DFS => back edge no cross edge, BFS=> cross edge(max one level difference) no back edge
Tree => connected, n vertices, n-1 edges jungle => graph with no cycles

DFS dont give shortest path

SCC (Strongly connected component) in directed graphs => exists path between every pair of vertices

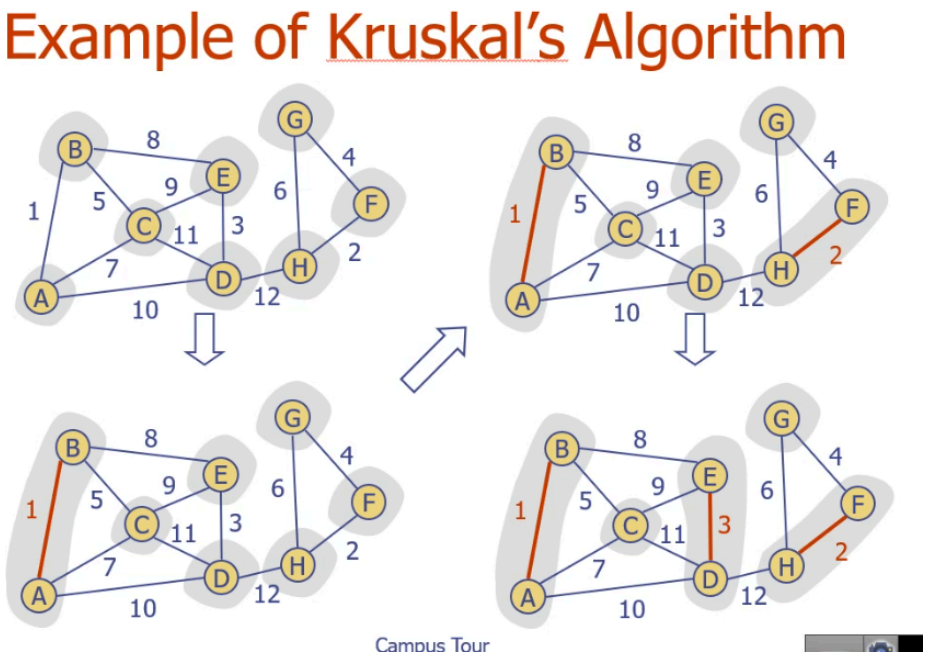
DAG => no SCC , no directed cycle

Topological sort => reverse DFS



MST

- Kruskal



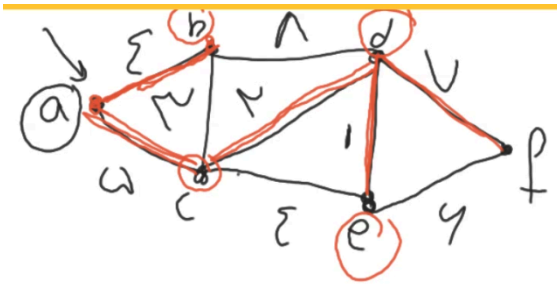
Partition-Based Implementation

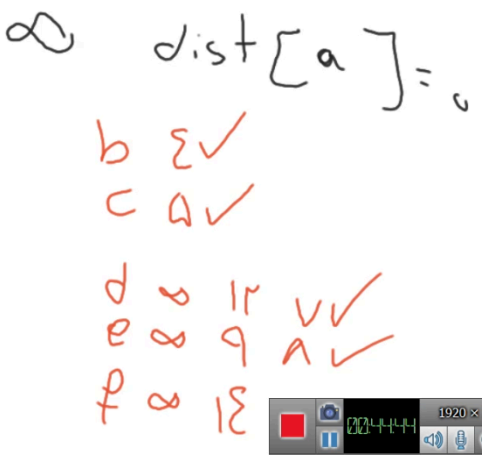
- Partition-based version of Kruskal's Algorithm
  - Cluster merges as unions
  - Cluster locations as finds
- Running time $O((n+m)\log n)$
  - Priority Queue operations: $O(m \log n)$
  - Union-Find operations: $O(n \log n)$

-

- Prim (MST) and Dikjstra (Min distance from a vertex) => only if smaller -> update not same

`

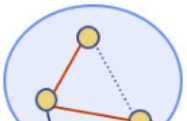




Baruvka
Connect components with smallest possible edges until obtain one component(first for each vertex then for each components)

Example of Baruvka's Algorithm (animated)

Slide by Matt Stallmann included with permission.



BFS => shortest distance from a vertex

Belman ford update each vertex distant value with min (its distance and its neighbor value + neighbor distance) O(VE)

```
Bellman-Ford(G,w, s)
   d[s] :=0; Set the others to ∞

 Repeat |V| stages:
   for each edge (u,v) ∈ E[G]
      d[v] := min{ d[v], d[u]+w(u,v); }  //relax(u,v)


 At the end of the algorithm, can detect negative cycles by:

 for each edge (u,v) ∈ E[G]
   if d[v] > d[u]+w(u,v)
      Return Negative cycle

 return No negative cycle
```

The Floyd-Warshall algorithm

A more clever dynamic programming algorithm

Before, $d_{i,j}^{(m)}$ = shortest paths of lengths ≤ m

Next: $d_{i,j}^{(m)}$ = shortest paths from i to j such that
all INTERMEDIATE vertices are ≤ m

$d^{(0)} = W$

$$d^{(m)}_{i,j} = \begin{cases} w(i,j) \\ \min\left(d^{(m-1)}_{i,j}\ ,\ d^{(m-1)}_{i,m} + d^{(m-1)}_{m,j}\right) \end{cases} \quad \text{if } m \geq 1.$$

Floyd-Warshall(W)

$D^{(0)} := W;$

for m = 1 to n
  for every i,j :
    $d^{(m)}_{i,j} = \min (d^{(m-1)}_{i,j} , d^{(m-1)}_{i,m} + d^{(m-1)}_{m,j})$

Return $D^{(n)}$

Time $\Theta(|V|^3)$

Erdős-Rényi-Gilbert Model (continued)

Note: Assume that the nodes are numbered 1 through n.

Algorithm for ER model graph generation:

for $i = 1$ **to** $n - 1$ **do** {
  for $j = i + 1$ **to** n **do** {
    Add edge $\{i, j\}$ with probability p.
  }
}

- So, by linearity of expectation, $\mathrm{E}[\mathrm{Degree}(v)] = p(n-1)$.

Let $\pi_k(v)$ denote the probability that node v has degree $= k$ $(0 \leq k \leq n-1)$. Then,

$$\pi_k(v) \quad = \quad \binom{n-1}{k} p^k (1-p)^{n-1-k}.$$

This called the **binomial distribution**.

This is the same probability as getting k heads from $n-1$ tosses of a coin, where the probability of heads $= p$.

Some non-trivial properties: The following results due to Erdős and Rényi are **asymptotic** (i.e., they hold for large n).

Condition	**Property of** $G(n,p)$
$p < 1/n$	Almost surely has **no** connected component of size larger than $c_1 \log_2 n$ for some constant c_1.
$p = 1/n$	Almost surely has a giant component of size at least $c_2\, n^{2/3}$ for some constant c_2.
$p > 1/n$	Almost surely has a giant component of size at least αn for some constant α $(0 < \alpha < 1)$. All other components will almost surely have size $\leq \beta \log_2 n$ for some constant β.
$p = 1/2$	With high probability, the size of the largest clique is $\approx 2 \log_2 n$.

Note: Step of the algorithm implements the "rich get richer" idea.

Example:

- Let $m = 1$; that is, each new node will get one edge.
- There are 4 nodes (numbered 1, 2, 3 and 4) and the new one is node 5.
- Let the degrees of nodes 1, 2, 3 and 4 be 3, 3, 2 and 2 respectively.
- Current sum of degrees $= 3 + 3 + 2 + 2 = 10$.
- For node 5:
  - $\Pr\{\text{Edge to node 1}\} = 3/10$.
  - $\Pr\{\text{Edge to node 2}\} = 3/10$.
  - $\Pr\{\text{Edge to node 3}\} = 2/10$.
  - $\Pr\{\text{Edge to node 4}\} = 2/10$.

Watts-Strogatz Model (continued)

Inputs:

- The number of nodes: n.
- An even integer K, the average node degree in the resulting graph.
- The rewiring probability β.
- **Assumption:** $n \gg K \gg \ln n \gg 1$.

Output: An undirected graph with the following properties.

- The graph has n nodes and $nK/2$ edges. (Thus, the average node degree is K.)
- With high probability, the average distance between any pair of nodes is $\ln(n)/\ln(K)$.

Watts-Strogatz Model (continued)

Notes:

- If $\beta = 0$, there is no rewiring and the diameter remains large.
- If $\beta = 1$, every edge gets rewired; it is known that such graphs are similar to graphs under the ER model.
- If $C(0)$ represents the average clustering coefficient of the initial graph, empirical evidence suggests that the average clustering coefficient $C(\beta)$ after rewiring is given by

$$C(\beta) \;=\; C(0)\,(1-\beta)^3 \;.$$

  If β is small, the clustering coefficient does not decrease much due to rewriting.

- **Newman-Watts Model:** Instead of rewiring, add edges between randomly chosen pairs of nodes with with probability $= \beta$.
  - This version is easier to implement.
  - The resulting model has properties similar to the Watts-Strogatz model.