6.

$G = \{V, E\}$

$V(G)$: vertex set

$E(G)$: edge set

degree: num of edges

vertex types:

odd or even degrees

$\sum_{V_i \in V(G)} degree(V_i) = |E(G)| \times 2$

0 or 2 nodes with odd degrees (偶数個)

→ visits every edges exactly once

* 奇数点不会是奇数個(画/不出来)

* 起点. 終点必為奇数 degree 的点

6 degrees of separation from the small world exp.

Simple path: a path passes through any vertex only once

Simple cycle: a cycle passes through the other vertices only once

Most common implementations of a graph

1. Adjacency matrix

2. Adjacency list - for a directed graph

DFS and BFS

* Depth-First Search Traversal

由图某一点开始搜寻, 先探鄰接迎上来走过的点, 盡量往深处搜寻, 直至最後, backtracking 前一節点, 持续走訪未走過的 node, 直至找到目标或遍歷全部 node.

→ 遞迴

→ iterative form uses a stack

* Breadth-First Search Traversal

以某顶点為起始点, 首先拜訪該顶点, 再走訪該顶点之所有相鄰顶点, 接下来下一层, 直至拜訪完所有顶点.

→ First visited, first explored

→ queue

→ recursive is possible, but not simple

7.

Topological order:

A list of vertices in a directed graph without cycles (DAG)

Directed Acyclic Graph

Topological sort:

Arranging the vertices into a topological order

Spanning tree 生成樹

從一張图取出一棵樹. 含图上所有点, 當一張图完成連通, 則擁有生成樹; 反之, 則无.

Prufer序列:

將一帶有節点編号之无根樹转化為一個序列之过程

Minimum Spanning Tree: Kruskal's Algorithm

求無向图的最小(大)生成樹. 若不连通. 則求最小(大)生成森林

-- Prim's Algo.

求无向图的最小(大)生成樹其中一棵

* 仿效 Dijkstra's Algo...

差異: D. Algo. 數次找不在樹上, 離根最近的点

P. Algo. ", 離樹最近的点.

8.

Activity-on-Edge (AOE) Network

* Directed edge : activity (task) to be performed
* vertex : event to signal the completion of certain activities
* Edge weight : the time required to perform an activity
* Path length : the total time from the start to the last event
* Critical path : a path with the longest length.

Critical Path Method (CPM)

* 关键路径法
* 是一种計劃管理方法

Maximum flow :

一張图，給定一源头与匯点，所有可能的flow中，流量最大者便為Maximum flow，可能有許多個

Ford-Fulkerson Algo.

→ residual graph

* residual capacity:

$c_f(u,v) = c(u,v) - f(u,v)$,

$c_f(v,u) = c(v,u) - f(v,u)$

Edmonds-Karp Algo.

→ Heuristic to find augmenting path

9.

External Sort (外部排序)

資料量过大，无法一次將所有资料置於memory，必须藉外部储存体储存，再做排序

k-way Merge

* 2-way merge

64 runs → $\log_2 64 = 6$ passes

16 runs → $\log_2 16 = 4$ passes

* k-way merge on m runs needs $\log_k m$ passes
* Higher-order merge can reduce I/O time

B-tree index

* Balance m-way search tree
  - = B-tree of order m
  - → Given the order m and tree height h, the number of keys N in the B-tree $\leq m^h - 1$

Index v.s. Data

* Insertion
  1. Add the data record → get the location in file
  2. Add the index entry
* Deletion
  1. Remove the index entry → get the location in file
  2. Remove the data record.

Variation of Hash file

* Hash indexing Methods
  - Static Hash
    → Fixed-length hash table
  - Extensible Hash
    → Hash table size is doubled if necessary
  - Linear hash
    Hash table size grows linearly