第一章、基本遍历

一、深度优先搜索

二、广度优先搜索

此图遍历中最基本的俩种算法，BFS，DFS，入选本图算法十大算法，自是无可争议。
因为，这俩种搜索算法，应用实为广泛而重要。

关于此BFS、DFS算法，更多，请参考：
经典算法研究系列：四、教你通透彻底理解：BFS和DFS优先搜索算法
http://blog.csdn.net/v_JULY_v/archive/2011/01/01/6111353.aspx

三、A*搜索算法

DFS和BFS在展开子结点时均属于盲目型搜索，也就是说，
它不会选择哪个结点在下一次搜索中更优而去跳转到该结点进行下一步的搜索。
在运气不好的情形中，均需要试探完整个解集空间, 显然，只能适用于问题规模不大的搜索问题中。

A*算法，作为启发式算法中很重要的一种，被广泛应用在最优路径求解和一些策略设计的问题中。
而A*算法最为核心的部分，就在于它的一个估值函数的设计上：
$f(n)=g(n)+h(n)$
其中f(n)是每个可能试探点的估值，它有两部分组成：
一部分，为g(n)，它表示从起始搜索点到当前点的代价（通常用某结点在搜索树中的深度来表示）。
一部分，即h(n)，它表示启发式搜索中最为重要的一部分，即当前结点到目标结点的估值。

更多，请参考：
经典算法研究系列：一、A*搜索算法
http://blog.csdn.net/v_JULY_v/archive/2010/12/23/6093380.aspx

附：
Flood Fill
LeeMaRS、wtzyb4446：
图形学中Flood Fill是满水法填充，是用来填充区域的。
就好比在一个地方一直到水，水会往四周满延开，直到高地阻挡。

Flood Fill就是从一个点开始往四周寻找相同的点填充，直到有不同的点为止。
我们用的Flood Fill和这个差不多原理,就是BFS的一种形式.

假设在(i,j)滴好大一滴红墨水,然后水开始漫开,向它的上下左右染色,也就是(i-1,j),(i+1,j),(i,j-1),(i,j+1)这四个点.然后在分别再从这四个点开始向周围染色...直到碰到某种边界为止.

把这个转化为BFS的思想,就是队列中初始元素是(i,j),然后把(i,j)扩展状态,得到(i-1,j),(i+1,j),(i,j-1),(i,j+1)这四个状态,加入队列.把(i,j)出列,继续扩展下一个结点...如此

第二章、最短路径算法

四、Dijkstra

Dijkstra 算法，又叫迪科斯彻算法（Dijkstra），
算法解决的是有向图中单个源点到其他顶点的最短路径问题。

此Dijkstra 算法已在本BLOG内俩篇文章中，有所具体阐述，请参见：
I、经典算法研究系列：二、Dijkstra 算法初探
http://blog.csdn.net/v_JULY_v/archive/2010/12/24/6096981.aspx
II、经典算法研究系列：二之续、彻底理解Dijkstra算法
http://blog.csdn.net/v_JULY_v/archive/2011/02/13/6182419.aspx

五、Bellman-Ford

Bellman-Ford：
求单源最短路，可以判断有无负权回路（若有，则不存在最短路），
时效性较好，时间复杂度O（VE）。

附：

SPFA：
是Bellman-Ford的队列优化，时效性相对好，时间复杂度O（kE）。（k<<V）。

六、Floyd-Warshall

Floyd-Warshall：求多源、无负权边的最短路。用矩阵记录图。时效性较差，时间复杂度O(V^3)。此算法是解决任意两点间的最短路径的一种算法，可以正确处理有向图或负权的最短路径问题。

更多，请参考：
几个最短路径算法比较：
http://blog.csdn.net/v_JULY_v/archive/2011/02/12/6181485.aspx

附：Kneser图

Kneser图是与图的分数染色有关的算法。
给定正整数a，b，a≥2b，Kneser图Ka:b是以如下方式定义的一个图：
其顶点是从给定的a个元素的集合中选出的b个元素构成的子集，两顶点间有边当且仅当这两个顶点为不交的集合。

第三章、最小生成树

七、Prim

八、Kruskal

此最小(权值)生成树的俩种算法，日后，会在本BLOG内 具体而深入阐述。

第四章、图匹配

九、匈牙利算法

匈牙利算法是众多用于解决线性任务分配问题的算法之一，是用来解决二分图最大匹配问题的经典算法，可以在多项式时间内解决问题，由匈牙利数学家Jack Edmonds于1965年提出。
这个算法，比较生疏，下面，稍微阐述下：

I、匈牙利算法应用问题的描述：
设G=(V,E)是一个无向图。如顶点集V可分区为两个互不相交的子集V1,V2之并，并且图中每条边依附的两个顶点都分属于这两个不同的子集。则称图G为二分图。二分图也可记为G=(V1,V2,E)。

给定一个二分图G，在G的一个子图M中，M的边集{E}中的任意两条边都不依附于同一个顶点，则称M是一个匹配。 选择这样的子集中边数最大的子集称为图的最大匹配问题(maximal matching problem)

如果一个匹配中，图中的每个顶点都和图中某条边相关联，则称此匹配为完全匹配，也称作完备，完美匹配。

II、算法描述
求最大匹配的一种显而易见的算法是：先找出全部匹配，然后保留匹配数最多的。但是这个算法的时间复杂度为边数的指数级函数。因此，需要寻求一种更加高效的算法。

下面介绍用增广路求最大匹配的方法(称作匈牙利算法，匈牙利数学家Edmonds于1965年提出)。
增广路的定义(也称增广轨或交错轨)：
若P是图G中一条连通两个未匹配顶点的路径，并且属于M的边和不属于M的边(即已匹配和待匹配的边)在P上交替出现，则称P为相对于M的一条增广路径。

由增广路的定义可以推出下述三个结论：
1－P的路径长度必定为奇数，第一条边和最后一条边都不属于M。
2－将M和P进行异或操作(去同存异)可以得到一个更大的匹配M’。
3－M为G的最大匹配当且仅当不存在M的增广路径。

算法轮廓：
(1)置M为空
(2)找出一条增广路径P，通过异或操作获得更大的匹配M’代替M
(3)重复(2)操作直到找不出增广路径为止

III、时间复杂度与空间复杂度
时间复杂度
邻接矩阵：最坏为O(n^3)　　邻接表：O(mn)
空间复杂度：
邻接矩阵：O(n^2)　　邻接表：O(m+n)

附：
Edmonds's matching

第五章、强连通分支算法与网络流

十、Ford-Fulkerson

最大流量算法(Ford-Fulkerson Algorithm),
也叫做贝尔曼-福特算法，被用于作为一个距离向量路由协议例如RIP, BGP, ISO IDRP, NOVELL IPX的算法。

附：Edmonds-Karp、Dinic、Push-relabel、maximum flow

强连通分支算法:
Kosaraju、 Gabow、 Tarjan。