6.1图的基本概念

图的定义

图G由顶点集V和边集E组成,记为G=(V,E),其中 V(G)表示图G中顶点的有限非空集;E(G)表示图G中顶点之间的关系(边)集合

$V=\{v_1, v_2, \cdots, v_n\}$ 则用|V|表示图G中顶点的个数也称图G的阶　　$E=\{(u,v) \mid u \in V, v \in V\}$ 用|E|表示图G中边的条数

注意:线性表可以是空表,树可以是空树,但图不可以是空图

图的一些基本概念及术语

- 有向图
  - 若E是有向边(也称弧)的有限集合时,则图G为有向图。
  - 弧是顶点的有序对,记为<v,w>,其中v,w是顶点,v称为弧尾,w称为弧头,<v,w>称为从顶点v到顶点w的弧,也称v邻接到w, 或 w邻接自v
- 无向图
  - 若E是无向边(简称边)的有限集合时,则图G为无向图
  - 边是顶点的无序对,记为(v,,w)或(w,v),因为(v,w)=(w,v),其中v,w是顶点
  - 可以说顶点w和顶点v互为邻接点。边(v,w)依附于顶点w和v,或者说边(v,w)和顶点y,w相关联
- 简单图
  - 不存在重复边
  - 不存在顶点到自身的边,则称图G为简单图
- 多重图　若图G中某两个结点之间的边数多于一条,又允许顶点通过同一条边和自己关联,则G为多重图
- 完全图(也称简单完全图)
  - 对于无向图,|E|的取值范围是0到n(n-1)/2,有n(n-1)2条边的无向图称为完全图,在完全图中任意两个顶点之间都存在边
  - 对于有向图,|E|的取值范围是0到n(n-1),有n(n-1)条弧的有向图称为有向完全图,在有向完全图中任意两个顶点之间都存在方向相反的两条弧
- 子图　设有两个图G=(V,E)和G'=(V',E'),若V'是V的子集,且E'是E的子集,则称G'是G的子图。若有满足V(G')=V(G)的子图G',则称其为G的生成子图
- 连通、连通图和连通分量
  - 在无向图中,若从顶点v到顶点w有路径存在,则称v和w是连通的
  - 若图G中任意两个顶点都是连通的,则称图G为连通图,否则称为非连通图
  - 无向图中的极大连通子图称为连通分量
  - 若一个图有n个顶点,并且边数小于n-1,则此图必是非连通图
- 强连通图、强连通分量
  - 强连通图:在有向图中,若从顶点v到顶点w和从顶点w到顶点v之间都有路径,则称这两个顶点是强连通的若图中任何一对顶点都是强连通的,则称此图为强连通
  - 有向图中的极大强连通子图称为有向图的强连通分量
- 生成树、生成森林
  - 连通图的生成树是包含图中全部顶点的一个极小连通子图
  - 若图中顶点数为n,则它的生成树含有n-1条边
  - 对生成树而言,若砍去它的一条边,则会变成非连通图,若加上一条边则会形成一个回路
  - 在非连通图中,连通分量的生成树构成了非连通图的生成森林
- 顶点的度,入度和出度
  - 度:定义为以该顶点为一个端点的边的数目
  - 无向图:顶点v的度是指依附于该顶点的边的条数　无向图的全部顶点的度的和等于边数的2倍
  - 有向图:全部顶点的入度之和与出度之和相等,并且等于边数
- 边的权和网　每条边都可以标上具有某种含义的数值,该数值称为该边的权值。这种边上带有权值的图称为带权图,也称网
- 稠密图、稀疏图
  - 边数很少的图称为稀疏图,反之称为稠密图
  - 一般当图G满足 $|E|<|V|\log|V|$ 可以将G视为稀疏图
- 路径、路径长度和回路
  - 路径:两个顶点之间相连的边
  - 路径长度:路径上边的数目称
  - 回路或环:第一个顶点和最后一个顶点相同的路径
  - 若一个图有n个顶点,并且有大于n-1条边,则此图一定有环
- 简单路径、简单回路
  - 简单路径:顶点不重复出现的路径
  - 简单回路:除第一个顶点和最后一个顶点外,其余顶点不重复出现的回路
- 距离
  - 从顶点u出发到顶点v的最短路径若存在,则此路径的长度称为从u到v的距离
  - 若从u到v根本不存在路径,则记该距离为无穷
- 有向树　一个顶点的入度为0、其余顶点的入度均为1的有向图,称为有向树

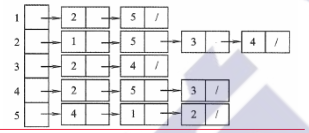
6.2图的存储及基本操作

邻接矩阵法

- 邻接矩阵存储
  - 是指用一个一维数组存储图中顶点的信息,用一个二维数组存储图中边的信息（即各顶点之间的邻接关系）
  - 存储顶点之间邻接关系的二维数组称为邻接矩阵
  - 结构示意
    $$A[i][j]=\begin{cases}1, & \text{若}(v_i,v_j)\text{或}<v_i,v_j>\text{是}E(G)\text{中的边}\\0, & \text{若}(v_i,v_j)\text{或}<v_i,v_j>\text{不是}E(G)\text{中的边}\end{cases}$$
  - 带权图示意
    $$A[i][j]=\begin{cases}w_{ij}, & \text{若}(v_i,v_j)\text{或}<v_i,v_j>\text{是}E(G)\text{中的边}\\0\text{或}\infty, & \text{若}(v_i,v_j)\text{或}<v_i,v_j>\text{不是}E(G)\text{中的边}\end{cases}$$
  - 邻接矩阵表示法的空间复杂度为O(n²),其中n为图的顶点数|V|
- 特点
  - 无向图的邻接矩阵一定是一个对称矩阵（并且唯一）。因此,在实际存储邻接矩阵时只需存储上（或下）三角矩阵的元素
  - 对于无向图,邻接矩阵的第i行（或第i列）非零元素(或非0元素）的个数正好是第i 个顶点的度 $TD(v_i)$
  - 对于有向图,邻接矩阵的第i行（或第i列）非零元素（或非o元素）的个数正好是第i 个顶点的出度 $OD(v_i)$
  - 用邻接矩阵法存储图,很容易确定图中任意两个顶点之间是否有边相连。但是,要确定图中有多少条边,则必须按行、按列对每个元素进行检测,所花费的时间代价很大
  - 稠密图适合使用邻接矩阵的存储表示

邻接表法

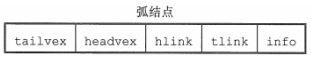
- 当一个图为稀疏图时，使用邻接矩阵法要浪费大量的存储空间，而图的邻接表法结合了顺序存储和链式存储方法，减少了不必要的浪费
- 结构
  - 对图G中的每个顶点建立一个单链表,第i个单链表中的结点表示依附于顶点 vi的边,这个单链表就称为顶点vi的边表（对于有向图则称为出边表）
  - 边表的头指针和顶点的数据信息采用顺序存储（称为顶点表）,所以在邻接表中存在两种结点:顶点表结点和边表结点
- 示意图



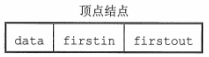
- 特点
  - 若G为无向图,则所需的存储空间为O(|V| + 2|E|);若G为有向图,则所需的存储空间为O|V|+|E|）
  - 对于稀疏图,采用邻接表表示将极大地节省存储空间
  - 在邻接表中,给定一顶点,能很容易地找出它的所有邻边,因为只需要读取它的邻接表
  - 在有向图的邻接表表示中,求一个给定顶点的出度只需计算其邻接表中的结点个数
  - 但求其顶点的入度则需要遍历全部的邻接表
  - 图的邻接表表示并不唯一

十字链表

- 十字链表是有向图的一种链式存储结构。在十字链表中,对应于有向图中的每条弧有一个结点,对应于每个顶点也有一个结点
- 时间复杂度O(|V| + |E|)
- 弧结点域结构
  - 尾域（tailvex）弧尾
  - 头域（headvex）：弧头
  - 链域（hlink）：指向弧头相同的下一条弧
  - 链域（tlink）：指向弧尾相同的下一条弧
  - info 域：指向该弧的相关信息



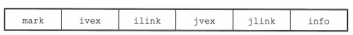
- 顶点结点域结构
  - data域存放顶点相关的数据信息
  - firstin和firstout 两个域分别指向以该顶点为弧头或弧尾的第一个弧结点



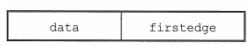
- 只能存有向图

邻接多重表

- 邻接多重表是无向图的另一种链式存储结构
- 时间复杂度O(|V| + |E|)
- 边结构
  - mark为标志域,可用以标记该条边是否被搜索过
  - ivex和jvex为该边依附的两个顶点在图中的位置
  - ilink指向下一条依附于顶点ivex的边;
  - jlink指向下一条依附于顶点jyvex 的边
  - info为指向和边相关的各种信息的指针域

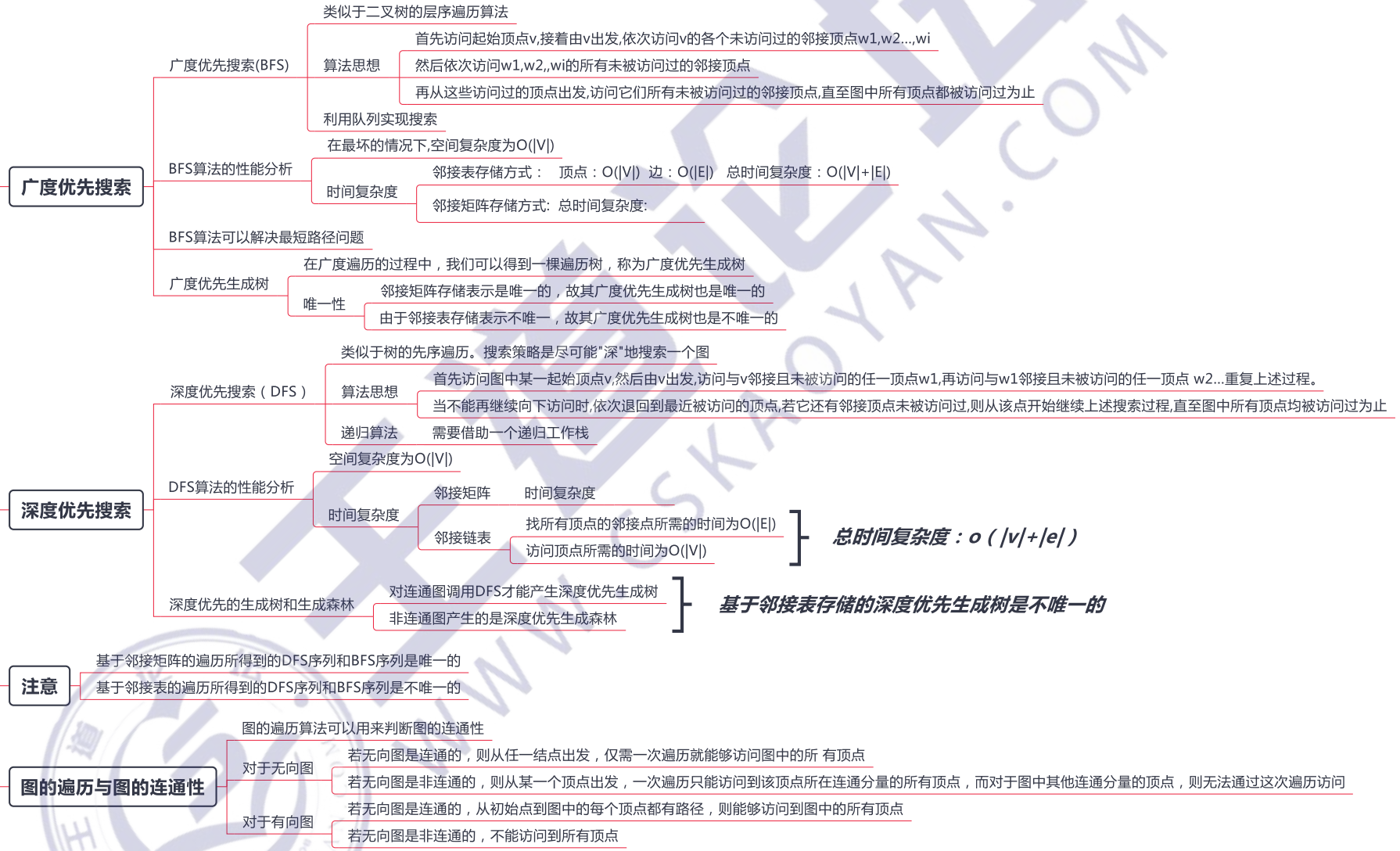


- 顶点结构
  - data域存储该顶点的相关信息
  - firstedge域指示第一条依附于该顶点的边



- 只能存无向图

6.3图的遍历



6.4图的应用(上)

6.4图的应用(下)