**第4章 图学习指导**

1、时间

2020年4月21日周二 （1-4班3-4节，5-10班、数学、大数据专业5-6节）

2020年4月23日周四 （1-4班3-4节，5-10班、数学、大数据专业5-6节）

2020年4月28日周二 （1-4班3-4节，5-10班、数学、大数据专业5-6节）

2020年4月30日周四 （1-4班3-4节，5-10班、数学、大数据专业5-6节）

4次课共8学时。

2、内容

腾讯课堂上课；

QQ群交流练习题（中间根据需要采用QQ视频或腾讯视频讲解）；

投票方式答题（点名、检查上课人数）；

通过课程视频回放重点复习；

问答。

具体安排：

|  |  |  |
| --- | --- | --- |
| 1 | 4月21日 | (1) 图的概念和相关术语，图的存储结构 |
| 2 | 4月23日 | (2) 图的遍历及其操作； |
| 3 | 4月28日 | (3) 图的应用算法，关节点与双连通性求解算法、强连通性、最小生成树算法、 |
| 4 | 4月30日 | (4)拓扑排序算法、关键路径算法、最短路径算法等 |

**3、知识要点**

**一个概念 + 二个存储结构 + 二个遍历算法 + 四个经典算法**

**3.1 基本概念和术语**

**图**是一种数据结构，由二个集合V和E组成，记做G=(V,E)，其中V是数据元素的非空有限集合，E是V中二元关系（边）的集合。即V是顶点的集合，E是顶点的偶对，即边的集合。

**有向图**：图的每条边都是有序顶点对（即边是有方向的）

**无向图**：图的每条边都是无序顶点对（即边是无方向的）

在有n个顶点的无向图中，e的范围为0－n(n-1)/2。具有n(n-1)/2条边的无向图称为**无向完全图**。

在有n个顶点的有向图中，e的范围为0-n(n-1)，具有n(n-1)条边的有向图称为**有向完全图**。

**度**：图的边或弧的数目等于顶点度数之和的一半。

在有向图或无向图中，如果存在首尾相接并且无重复边（弧）的边（弧）序列，则称这个序列是一条从顶点v0到vn-1的一条路径，序列中的边（弧）数称为**路径长度**。

**简单路径**：在一条路径中，若除起点和终点以外，所有顶点彼此各不相同，则称该路径为简单路径。

**回路**：在一条路径中，若起点和终点都是同一顶点，则称该路径为回路。

**简单回路**：有简单路径组成的回路称为简单回路

**连通：**在无向图G中，若任意二个顶点之间都是连通的,即均存在路径，则称图G为连通图。否则称为非连通图。

**连通分量**：在非连通的无向图G中，极大连通子图称为无向图的连通分量。

**强连通**：在有向图G中，若顶点vi到顶点vj有路径存在，并且从顶点vj到顶点vi也有路径存在，则称vi到vj是强连通的。

**强连通图**：在有向图G中，若任意二个顶点之间都是强连通的，则称有向图G为强连通图。

**强连通分量**：在非强连通的有向图G中，极大强连通子图称为有向图的强连通分量。

**3.2 图的存储结构**

（1）邻接矩阵

用一维数组存储图中顶点的信息，用矩阵表示图中各顶点之间的邻接关系。在图的邻接矩阵存储方法具有以下特点：

①无向图的邻接矩阵一定是一个对称矩阵。因此，在具体存放邻接矩阵时只需存放上(或下)三角矩阵的元素即可。

②对于无向图，邻接矩阵的第i 行(或第i 列)非零元素(或非∞元素)的个数正好是第i 个顶点的度TD(vi)。

③对于有向图，邻接矩阵的第i 行(或第i 列)非零元素(或非∞元素)的个数正好是第i 个顶点的出度OD(vi) (或入度ID(vi)) 。

④用邻接矩阵方法存储图，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。这是用邻接矩阵存储图的局限性。

（2）邻接表

是图的一种顺序存储与链式存储结合的存储方法，类似于树的孩子链表表示法。就是对于图G 中的每个顶点vi，将所有邻接于vi 的顶点vj 链成一个单链表，这个单链表就称为顶点vi 的邻接表，再将所有点的邻接表表头放到数组中，就构成了图的邻接表。

图的邻接表存储方法具有以下特点：

①若无向图中有n 个顶点、e 条边，则它的邻接表需n 个头结点和2e 个表结点。稀疏图用邻接表表示比邻接矩阵节省存储空间，当和边相关的信息较多时更是如此。

②在无向图的邻接表中，顶点vi 的度恰为第i 个链表中的结点数；在有向图中，第i 个链表中的结点个数只是顶点vi 的出度，为求入度，必须遍历整个邻接表。在所有链表中其邻接点域的值为i 的结点的个数是顶点vi 的入度。有时，为了便于确定顶点的入度或以顶点vi 为头的弧，可以建立一个有向图的逆邻接表，即对每个顶点vi 建立一个链接以vi 为头的弧的链表。在建立邻接表或逆邻接表时，若输入的顶点信息即为顶点的编号，则建立邻接表的复杂度为O(n+e)，否则，需要通过查找才能得到顶点在图中位置，则时间复杂度为O(n×e)。

③在邻接表上容易找到任一顶点的第一个邻接点和下一个邻接点，但要判定任意两个顶点( vi 和vj)之间是否有边或弧相连，则需搜索第i 个或第j 个链表，因此，不及邻接矩阵方便。

**3.3 图的遍历**

为了保证图中的各顶点在遍历过程中访问且仅访问一次，需要为每个顶点设一个访问标志，因此设置一个访问标志数组visited[n]，用于标示图中每个顶点是否被访问过。

1. 深度优先遍历（递归算法）

类似于树的先根遍历，是树的先根遍历的推广。

遍历图的过程实质上是对每个顶点查找其邻接点的过程，其耗费的时间则取决于所采用的存储结构。当以邻接矩阵为图的存储结构时，查找每个顶点的邻接点所需时间为O(n2) ，其中n 为图中顶点数。而当以邻接表作图的存储结构时，找邻接点所需时间为O(e) ，其中e 为元向图中边的数或有向图中弧的数。由此，当以邻接表作存储结构时，深度优先搜索遍历图的时间复杂吏为O(n+e)。

（2）广度优先遍历（非递归算法）

类似于树的按层次遍历的过程

广度优先搜索遍历图的过程实质是通过边或弧找邻接点的过程，其时间复杂度和深度优先搜索遍历相同，两者的不同之处仅仅在于对顶点访问的顺序不同。

**3.4最小生成树**

图的生成树是不唯一的，从不同的顶点出发得到的生成树是不同的，但是如果图是带权图，则权值之和最小的生成树是称为**最小生成树**（Prim 算法、Kruskal 算法，两个算法的比较）。

• Prim 算法：取图中任意一个顶点v 作为生成树的根，之后往生成树上添加新的顶点韧。在添加的顶点w 和已经在生成树上的顶点v 之间必定存在一条边，并且该边的权值在所有连通顶点v 和w 之间的边中取值最小。之后继续往生成树上添加顶点，直至生成树上含有n-1 个顶点为止。

• Kruskal 算法：先构造一个只含n 个顶点的子图SG，然后从权值最小的边开始，若它的添加不使子图SG 中产生回路，则在子图SG 上加上这条边，如此重复，直至加上n-l 条边为止。

**3.5有向无环图**

一个无回环的有向图称为有向无环图，简称为DAG图。

**拓扑排序算法**（递归与非递归算法）。

利用广度优先搜索算法，从度数为的顶点开始搜索。整个算法的时间复杂度为O(e+n)。拓扑排序的序列可能不唯一。

当有向图中无环时，也可用深度优先遍历的方法进行拓扑排序，按DFS 算法的先后次序记录下的顶点序列为逆向的拓扑有序序列。

**3.6关键路径（AOE网）**

在带权的有向图中，用结点表示事件，用边表示活动，边上权表示活动的开销（如持续时间），则称此有向图为边表示活动的网络,简称AOE网。

AOE网研究的主要问题：

如果用AOE 网表示一项工程，那么仅仅考虑各个子工程之间的优先关系还不够，更多地是关心整个工程完成的最短时间是多少，哪些活动的延迟将影响整个工程进度,而加速这些活动能否提高整个工程的效率,因此AOE网有待研究的问题是:

(1)完成整个工程至少需要多少时间？

(2)哪些活动是影响工程进度的关键活动？

在AOE网中，从源点到汇点的带权路径长度最长的路径称为**关键路径**。

**关键活动**：在关键路径上的弧称为关键活动。

**3.7单源最短路径**

求某源点到其余各顶点的**最短路径（Dijkstra算法）**；

任意两个顶点间的最短路径（Floyd算法）；

传递闭包（Warshall算法）；

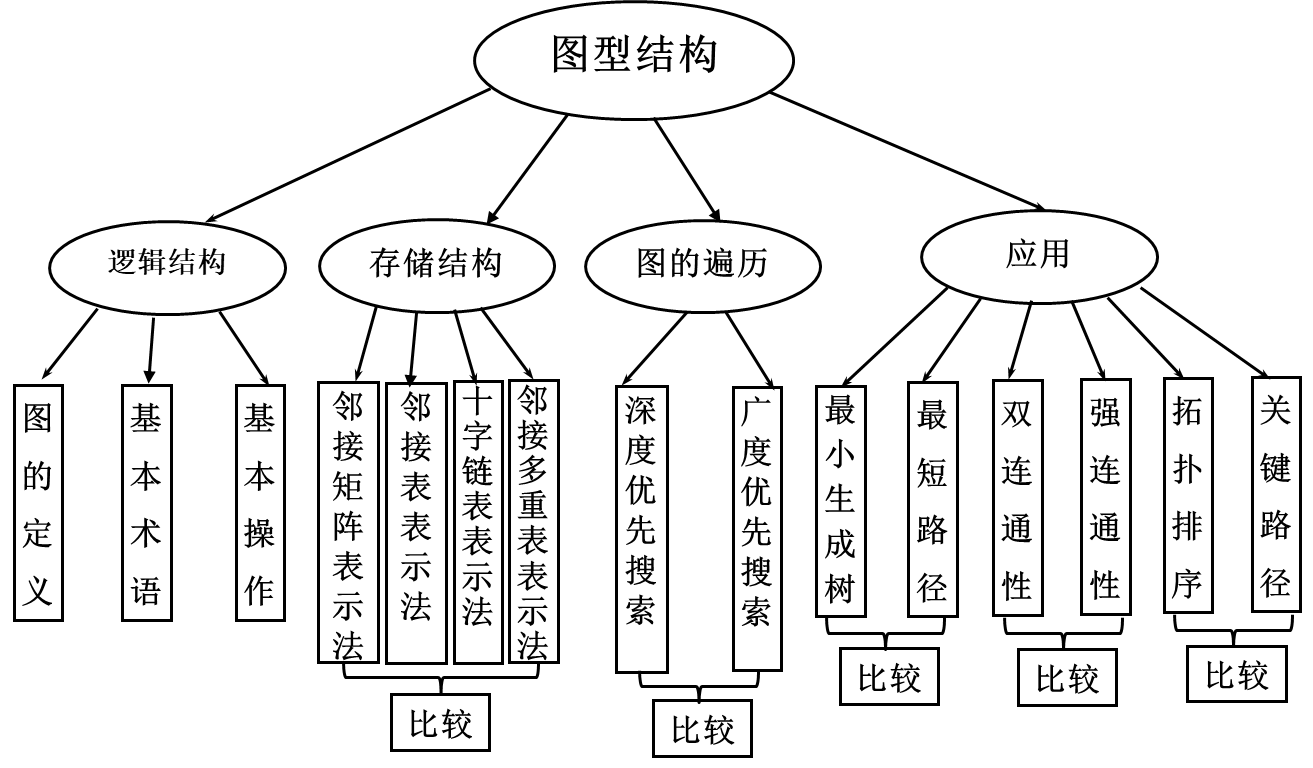
有向图的中心点及其求解。

• Dijkstra 算法：依最短路径的长度递增的次序求得各条路径。设置辅助数组

Dist，其中每个分量Dist[k]表示当前所求得的从源点到其余各顶点k 的最短 路径。Dist[k] = <源点到顶点k 的弧上的权值>，或者=<源点到其他顶点的路径长度>+<其他顶点到顶点k 的弧上的权值>。时间复杂度是O(n2)。

• Floyd 算法：从vi 至vj 的所有可能存在的路径中，选出一条长度最短的路径。时间复杂度 是O(n3)。

**3.8 知识点小结**



**4、习题**

**一、选择题**

（1）在一个图中，所有顶点的度数之和等于图的边数的（ ）倍。

A．1/2 B．1 C．2 D．4

（2）在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的（ ）倍。

A．1/2 B．1 C．2 D．4

（3）具有n个顶点的有向图最多有（ ）条边。

A．n B．n(n-1) C．n(n+1) D．n2

（4）n个顶点的连通图用邻接距阵表示时，该距阵至少有（ ）个非零元素。

A．n B．2(n-1) C．n/2 D．n2

（5）G是一个非连通无向图，共有28条边，则该图至少有（ ）个顶点。

A．7 B．8 C．9 D．10

（6）若从无向图的任意一个顶点出发进行一次深度优先搜索可以访问图中所有的顶点，则该图一定是（ ）图。

A．非连通 B．连通 C．强连通 D．有向

（7）下面（　）算法适合构造一个稠密图G的最小生成树。

A． Prim算法 B．Kruskal算法 C．Floyd算法 D．Dijkstra算法

（8）用邻接表表示图进行广度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

（9）用邻接表表示图进行深度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

（10）深度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

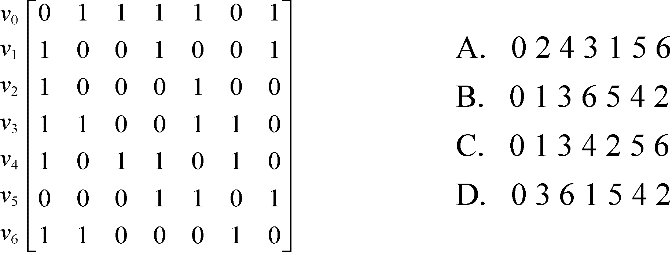
（11）广度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

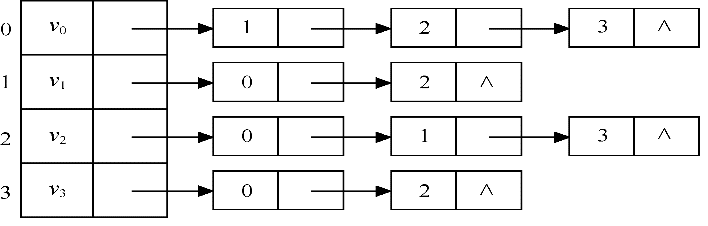
（12）图的BFS生成树的树高比DFS生成树的树高（ ）。

A．小 B．相等 C．小或相等 D．大或相等

（13）已知图的邻接矩阵如下图所示，则从顶点*v*0出发按深度优先遍历的结果是（ ）。



（14）已知图的邻接表如下图所示，则从顶点*v*0出发按广度优先遍历的结果是（ ），按深度优先遍历的结果是（ ）。



A．0 1 3 2 B．0 2 3 1 C．0 3 2 1 D．0 1 2 3

（15）下面（ ）方法可以判断出一个有向图是否有环。

A．深度优先遍历 B．拓扑排序 C．求最短路径 D．求关键路径

**二、填空题：**

（1）n个顶点的连通图至少\_\_ \_\_条边。

（2）在无权图G的邻接矩阵A中，若(vi,vj)或＜vi,vj＞属于图G的边集合，则对应元素A[i][j]等于\_\_ \_\_，否则等于\_ \_\_。

（3）在无向图G的邻接矩阵A中，若A[i][j]等于1，则A[j][i ]等于\_\_ \_\_。

（4）已知图G的邻接表如下图所示，其从顶点v1出发的深度有限搜索序列为\_ \_，其从顶点v1出发的宽度优先搜索序列为\_ 。

v1

v3

v2

v4

v5

v6

v2

v5

v4

v3

v5

^

^

v6

v4

v6

v3

（5）已知一个有向图的邻接矩阵表示，计算第i个结点的入度的方法是\_ \_。

（6）已知一个图的邻接矩阵表示，删除所有从第i个结点出发的边的方法是\_ \_。

（7）如果含n个顶点的图形成一个环，则它有 棵生成树。

（8）一个非连通无向图，共有28条边，则该图至少有 个顶点。

（9）遍历图的过程实质上是 。BFS遍历图的时间复杂度为 ，DFS遍历图的时间复杂度为 ，两者不同之处在于 ，反映在数据结构上的差别是 。

（10）一个图的 表示法是唯一的，而 表示法是不唯一的。

（11）有向图中的结点前驱后继关系的特征是 。

（12）若无向图G的顶点度数最小值大于等于 时，G至少有一条回路。

（13）根据图的存储结构进行某种次序的遍历，得到的顶点序列是 的。

**三、应用题**

（1）已知图a所示的有向图，请给出：

① 每个顶点的入度和出度； ② 邻接矩阵；

③ 邻接表； ④ 逆邻接表。

  图a 有向图 图b 无向网

（2）已知如图b所示的无向网，请给出：

① 邻接矩阵；

② 邻接表；

③ 最小生成树

（3）已知图的邻接矩阵如图c所示。试分别画出自顶点1出发进行遍历所得的深度优先生成树和广度优先生成树。

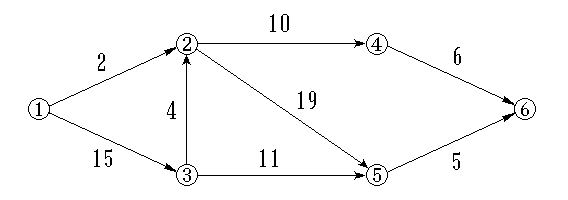
（4）有向网如图d所示，试用迪杰斯特拉算法求出从顶点a到其他各顶点间的最短路径。

图c 邻接矩阵 图d 有向网

（5）试对图e所示的AOE-网：

① 求这个工程最早可能在什么时间结束；

② 求每个活动的最早开始时间和最迟开始时间；

③ 确定哪些活动是关键活动

图e AOE-网

**四、算法设计题**

（1）分别以邻接矩阵和邻接表作为存储结构，实现以下图的基本操作：

① 增加一个新顶点v，InsertVex(G, v)；

② 删除顶点v及其相关的边，DeleteVex(G, v);

③ 增加一条边<v，w>，InsertArc(G, v, w);

④ 删除一条边<v，w>，DeleteArc(G, v, w)。

（2）一个连通图采用邻接表作为存储结构，设计一个算法，实现从顶点v出发的深度优先遍历的非递归过程。

（3）设计一个算法，求图G中距离顶点v的最短路径长度最大的一个顶点，设v可达其余各个顶点。

（4）试基于图的深度优先搜索策略写一算法，判别以邻接表方式存储的有向图中是否存在由顶点vi到顶点vj的路径（i≠j）。

（5）采用邻接表存储结构，编写一个算法，判别无向图中任意给定的两个顶点之间是否存在一条长度为k的简单路

（6）设计算法，将一个无向图的邻接矩阵转换为邻接表。

（7）设计算法，将一个无向图的邻接表转换成邻接矩阵。