夏可为

2015301200168

卓工班

数据结构

7.1 基于邻接矩阵：空间复杂度为$O(n^2)$，用一个$n \times n$的矩阵来表示有n个结点的图，第i行第j个表示结点v_i和v_j之间的关系。在无权图中，用0和1分别表示两个结点之间有无连接关系，带权图中有∞表示两结点间无连接关系，0表示结点和结点自身，数字表示结点间权值

基于邻接表：仅存储空间与图的结点和边的数量均有关，储存稀疏图时占用空间远小于基于邻接矩阵。用一个结点表储存图的所有结点，每个结点类内部还有两个线性表分别储存该结点的相邻结点与对应的权值

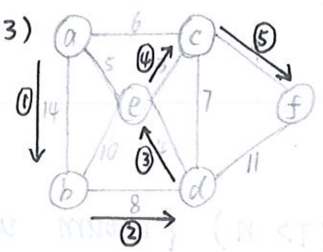
7.2 分别用0,1,2,3,4,5表示(编号)结点a,b,c,d,e,f

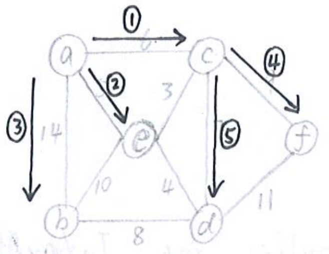
(1) $$\begin{pmatrix} 0 & 14 & 6 & \infty & 5 & \infty \\ 14 & 0 & \infty & 8 & 10 & \infty \\ 6 & \infty & 0 & 7 & 3 & 9 \\ \infty & 8 & 7 & 0 & 4 & 11 \\ 5 & 10 & 3 & 4 & 0 & \infty \\ \infty & \infty & 9 & 11 & \infty & 0 \end{pmatrix}$$

(2) a − 3　b − 3　c − 4

d − 4　e − 4　f − 2

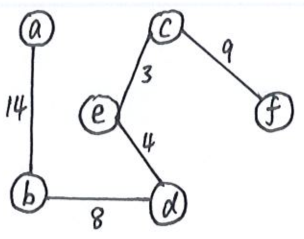
(3)



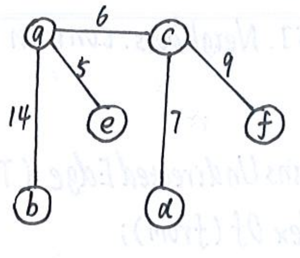


一种深度优先：a,b,d,e,c,f　　一种广度优先：a,c,e,b,f,d

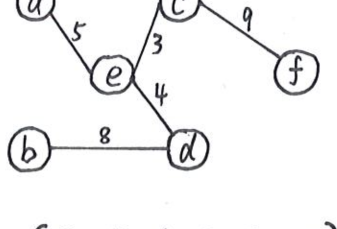
(4) 一棵深度优先生成树



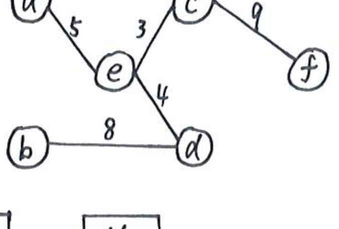
一棵广度优先生成树



(5) kruskal 算法

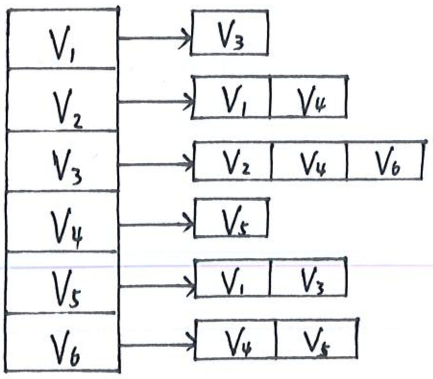


Prim 算法（从结点A开始）



7.3 (1) $$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

(2)

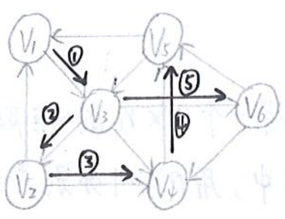


(3)

结点	入度	出度	度
V_1	2	1	3
V_2	1	2	3
V_3	2	3	5
V_4	3	1	4
V_5	2	2	4
V_6	1	2	3

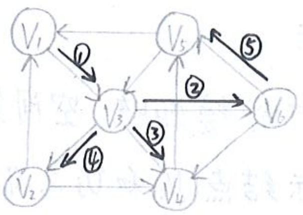
(5)

深度优先



$V_1, V_3, V_2, V_4, V_5, V_6$

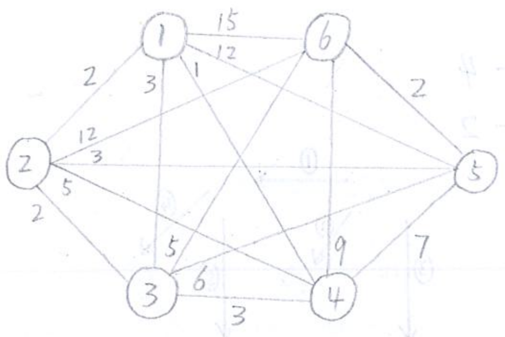
广度优先



$V_1, V_3, V_6, V_4, V_2, V_5$

(4) 图的强连通分量就是其自身

7.4



最短路径：$1 \xrightarrow{2} 2 \xrightarrow{3} 5 \xrightarrow{2} 6$

$2+3+2=7$

7.5 `public int IndexOf(Vertex<T> n) {return vertexList.IndexOf(n);}`

7.6
```
public bool ContainsDirectedEdge(T from, T to){
    int i= IndexOf(from);
    return nodes[i].Neighbors.contain(new GraphNode<T>(to));
}
public bool ContainsUndirectedEdge(T from, T to){
    int i = IndexOf(from);
    return nodes[i].Neighbors.contain(new GraphNode<T>(to));
}
```