



抽象数据结构的 物理实现

⋮

4个例子

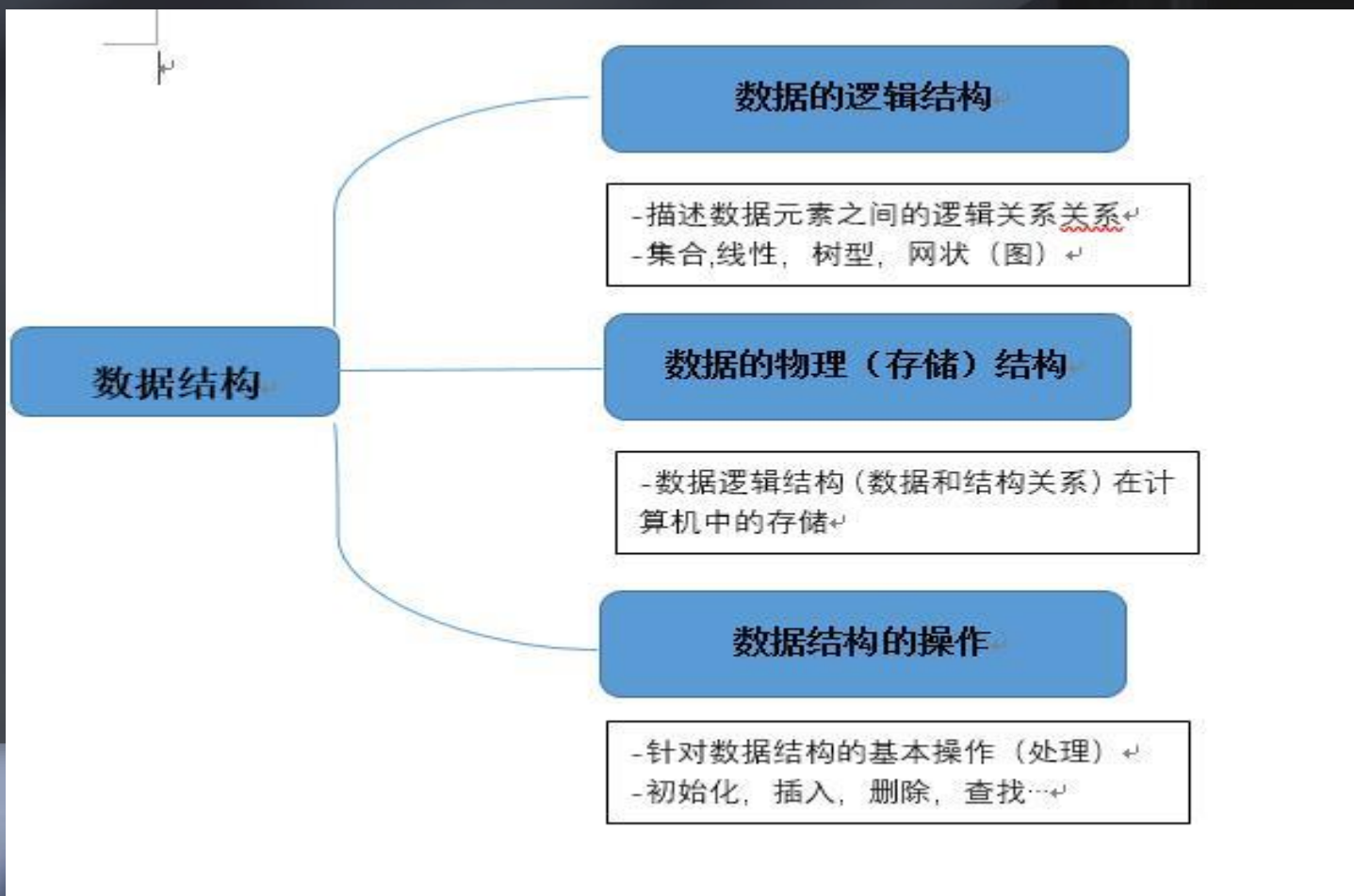
主要从一下四个方面来介绍

- ◆ 物理数据结构的概述
- ◆ 物理数据结构的定义
- ◆ 如何存储数据结构到物理数据结构中
- ◆ 如何构建（算法）
样例说明和算法描述

01. 二叉树的二叉链表表示法
02. 二叉树的左右兄弟结点表示法
03. 图的邻接矩阵
04. 图的邻接表



数据结构简述



◆ 数据的存储结构:

1. 定义: 若 $B(K,R)$ 是一个数据结构, 有一个映射 $S:K \rightarrow M$, 对于每一个 k 属于 K , 都有唯一的 z 属于 M (存储区域), 使得 $S(k)=z$,同时这个映射 S 应具有明显地或者隐含地体现关系 R 的能力, 则称对逻辑结构进行了存储。通俗的说, 就是如何在存储中实现结构

◆ 存储结构的实现方式:

1. 顺序方式: 借助元素在存储器汇总的相对位置来表示数据元素间的逻辑关系。特点: 逻辑上相邻, 物理位置上相邻
2. 链接方式: 借助指示元素存储地址的指针表示数据元素之间的逻辑关系。特点: 借助指针表示数据元素之间的逻辑关系
3. 索引方式: 特点 高效查询, 有序表的查询
4. 散列方式: 特点 高效查询, 时间复杂度为 $O(1)$, 但有冲突问题

二叉树的二叉链表表示法

1. 概述

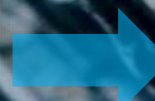
- 树的二叉链表实现方式，一种链接的存储方式
- 以二叉链表作为树的存储结构。通常就是用链表的方式来反映出二叉树元素之间的逻辑关系

2. 定义

- 二叉树的结点包含一个数据元素以及若干指向其子树的分支
- 二叉链表由二叉树的定义可知，由结点链接而成，每一个结点至少包含3个域，数据域，左右指针域。有时还可以在结点结构中增加一个指向其双亲的指针域。利用这种结点结构所得到的存储结构称为二叉链表。

3. 如何存储抽象数据结构到物理数据结构

- 二叉树是由根结点和左右子树构成，左右子树在结构上也是一棵二叉树。
- 存储：创建一个BinNode类，其包含根结点的数据域，左右孩子指针lc, rc, 分别用来指向左右子树。
- 存储上首先新建二叉树的根结点，确定数据域的内容，递归调用创建二叉树的函数，设置根结点左指针指向左子树，同样设置根结点右指针指向右子树，这样调用递归程序就可以实现用二叉链表存储一棵二叉树。



一般利用二叉树的递归定义可以实现用二叉链表实现二叉树的存储，针对具体的实现也可以稍作调整

📷 4. 如何构建（样例说明和算法描述）

样例说明

根据二叉树的后序遍历，中序遍历，将二叉树的结点信息存入二叉链表中，然后构建出二叉树。

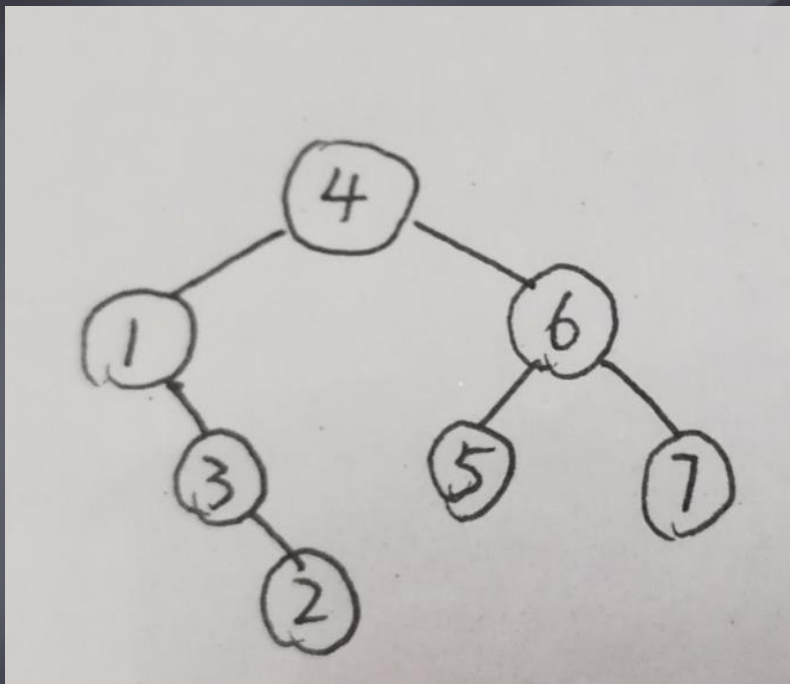
7（结点个数）

2 3 1 5 7 6 4（后序）

1 2 3 4 5 6 7（中序）

根据根结点的位置确定出中序中左右子树，然后左右子树利用递归的想法和上面一样确定出。看样例中的后序遍历确定出根结点4，在中序遍历中找到根结点4，进而确定出左子树序列1, 2, 3，右子树序列5, 6, 7，然后看左子树，根据后序确定出根结点1，在中序中找到，确定出左子树无左孩子，右孩子序列为2, 3，如此依次寻找，每次找到将其对应的值存入二叉链表中

构建出的二叉树型
结构如下：



算法描述

- 创建二叉树的二插链表存储结构
- 以左子树右子树的数组中存储的位置关系读入的递归算法
- 按先序遍历顺序读入每个节点值，创建二叉链表

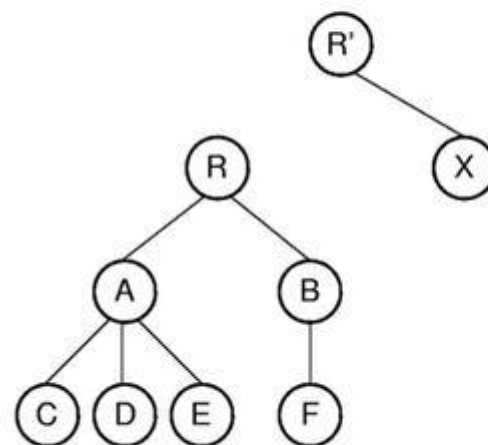


二叉树的左右兄弟结点表示法

1. 概述

- 子兄弟结点表示法是一种简单的树的表示法，基于数组来实现。
- 数组中存储的元素是一个结点类型，是一种线性的存储结构。
- 每个结点在存储上只需要固定的存储空间，在某些情况下空间开销较大（树的结点个数较多）
- 左右兄弟中要求父亲相同，左右兄弟分别是表示同一层中的左右结点

左子结点/右兄弟结点表示法



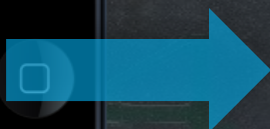
Left	Val	Par	Right
1	R	/	/
3	A	0	2
6	B	0	/
/	C	1	4
/	D	1	5
/	E	1	/
/	F	2	/
8	R'	/	/
/	X	7	/

2. 定义

- 该存储结构中用一个数组来存储结点，用一组地址连续的存储单元依次自上而下存储每一个结点
- 左右兄弟结点表示法中，结点的结构包括存储结点值的变量，指向父节点，以及最左结点和右兄弟结点的指针。二叉树的基本操作可以根据读取结点中的一个值来实现

3. 如何存储抽象数据结构到物理数据结构

首先需要创建一个结构体数组，数组中存储的元素是结点，结点可以用类或者结构体来实现，每个结点包括存储结点值的变量，指向父节点，以及最左结点和右兄弟结点的指针。

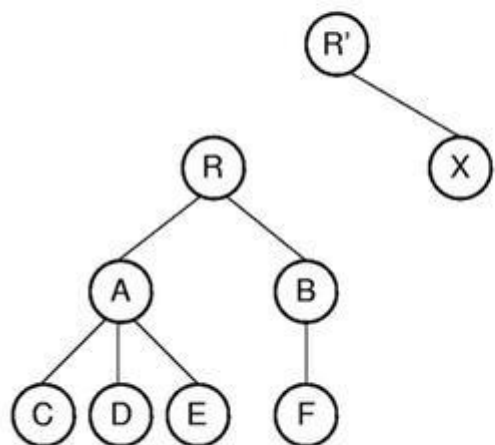


根据输入遍历每一个结点，找到这个结点的父节点，和左右结点指针，将其存入结构体数组中，完成对每一个结点的存入。根据数组中每一个结点和其左右结点的关系，可以还原出这棵二叉树

📷 4. 如何构建（样例说明和算法描述）

样例说明

左子结点/右兄弟结点表示法



Left	Val	Par	Right
1	R	/	/
3	A	0	2
6	B	0	/
/	C	1	4
/	D	1	5
/	E	1	/
/	F	2	/
8	R'	/	/
/	X	7	/

如左图所示有两颗树，首先读入第一个结点R，其左子结点是A，在R左边存入1，R没有父节点和右结点，右边两个位置为空。遍历下一个结点A，左结点是C，在A左边存入3，父节点是R，在A的右边第一个位置存入0，右结点是B，在A的右边第二个位置存入2。其他结点按照同样的方法来存储。这样就可以将这两个二叉树存在静态链表中。

算法描述

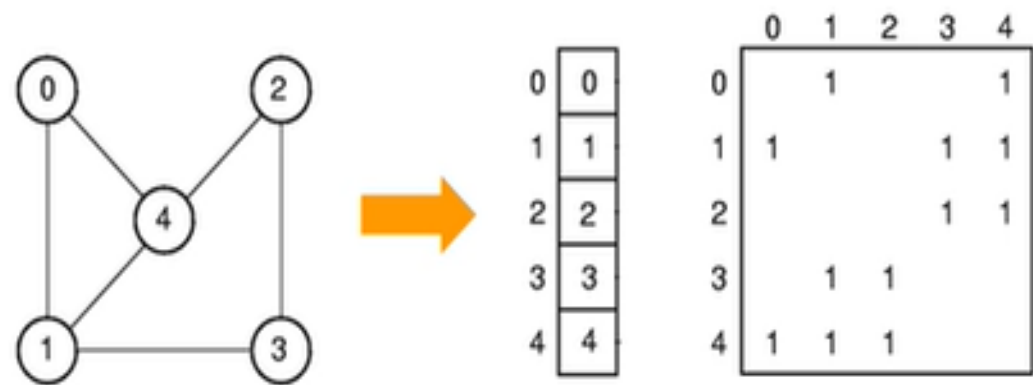
- 创建一个结构体数组，用来存储二叉树中每一个结点，结点中包括了该结点的父节点和左右结点的指针，用来联系每个结点之间的关系
- 通过遍历，将每一个结点存入数组中。
- 在对二叉树进行基本操作时，可以通过读取数组中的结点的值来实现。



图的邻接矩阵

1. 概述

图的邻接矩阵表示法也称为邻接相邻矩阵或者二维数组表示法。其特点是用两个二维数组分别来存储数据元素（顶点）的信息和数据元素之间的关系（边或弧）的信息





2. 定义

- 图的邻接矩阵能有效反映出任意两点之间有无连接关系。
- 实际上就是一个 $V \times V$ 的标记数组，假设 $V=n$ ，各顶点依次记为 $v_0, v_1, v_2, \dots, v_{n-1}$ ，则邻接矩阵的第 i 行包括以 v_i 为起点的边。如果从 v_i 到 v_j 存在一条边，则对第 i 行的第 j 个元素进行标记，否则不进行标记。这个邻接矩阵的内部存储的就是图各个顶点之间有无边的标记。



3. 如何存储抽象数据结构到物理数据结构

定义两个数组，一个一维数组，用来存储顶点的信息，另外一个二维数组用来存储任意两个顶点之间有无边的连接关系，数组首先初始化为全0。

利用循环将顶点信息存入一维数组中，根据顶点顺序与数组下标的关系，在两点之间有边相连的两个点之间，给相应的二维数组中的位置出标记为1，表示有边。

在求解图的问题时，可以采用二重循环来看点与点之间是否有边相连，以便可以完成图的遍历，最短路径的求解等问题。

📷 4. 如何构建（样例说明和算法描述）

样例说明

输入第一行表示图的结点数和边数；输入第二行表示顶点信息；输入第三行表示各个结点之间存在的关系以及边的权值

```
4 3
A B C D
A B 1
B C 1
B D 1
```

- 首先根据结点数，定义一个存储顶点信息的数组vexs, 将顶点信息存入里面，再定义一个数组m存储边的关系，写一个函数可以寻找顶点与对应序号关系，在输入每一组顶点的时候，将边的信息记录在邻接矩阵。
- A, B分别对应0, 1, m[0][1], m[1][0]值赋值为1, B, C分别对应1, 2, m[1][2], m[2][1]赋值为1, B, D分别对应1, 3, m[1][3], m[3][1]赋值为1, 其余位置为0, 这样就创建了图，将图的非线性结构存入了一个数组。

构建出的图在矩阵
中：

0	1	0	0
0	0	1	1
0	0	0	0
0	0	0	0

算法描述

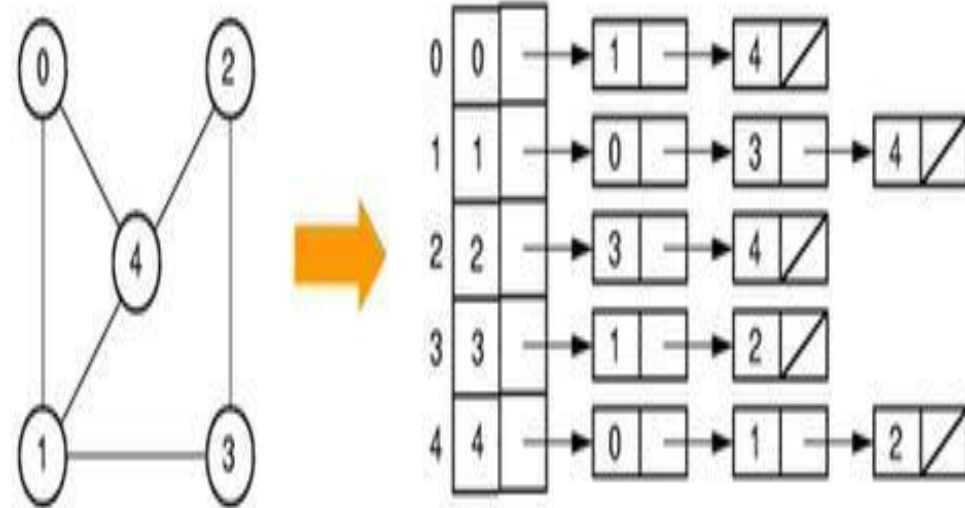
- 创建两个数组，一个用来存储顶点信息，另外一个用来存储图中顶点之间的关系
- 根据输入判断两个顶点之间有无边，有边就在相应二维数组中标记为1，标记结束后，数组中其余位置填入0



图的邻接表表示法

1. 概述

- 邻接表，存储方法跟树的孩子链表示法相类似，是一种顺序分配和链式分配相结合的存储结构。如这个表头结点所对应的顶点存在相邻顶点，则把相邻顶点依次存放于表头结点所指向的单向链表中。
- 对于无向图来说，使用邻接表进行存储也会出现数据冗余，表头结点A所指链表中存在一个指向C的表结点的同时，表头结点C所指链表也会存在一个指向A的表结点。





2. 定义

- 邻接表是图的一种链式存储结构。
- 在邻接表中，对图中的每一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 v_i 的边。
- 每个结点由三个域构成，其中邻接点域指示于顶点 v_i 邻接的点在图中的位置，链域指示下一条边或者弧的结点，数据域存储边或者弧的相关信息，比如权值。



3. 如何存储抽象数据结构到物理数据结构

建立一个边类Edge, 用来存储边或者弧的相关信息（包括权值），在一个结构体数组中，定义一个变量date存储顶点信息，每个顶点对应一个子结点链表，存储和当前结点相关的所有点

根据输入，首先用isEdge()判断两个顶点v1, v2之间是否存在边相连, 若有边相连则去除当前链接，再插入当前边，如果不存在边相连，移至表尾，再插入当前边。

无向图的话，则插入当前结点后还需要，交换v1, v2位置，再次进行相同的操作

📷 4. 如何构建（样例说明和算法描述）

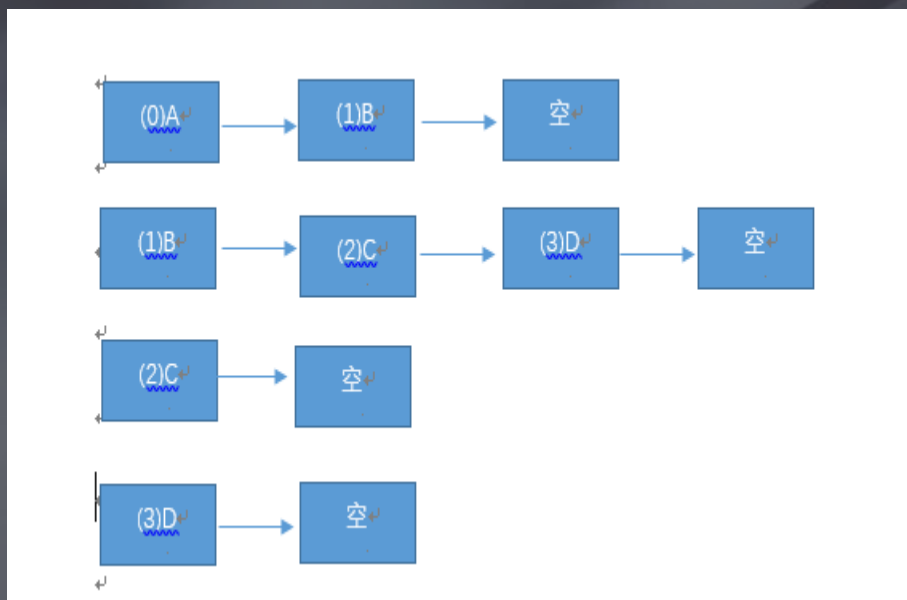
样例说明

输入第一行表示图的结点数和边数；输入第二行表示顶点信息；输入第三行表示各个结点之间存在的关系以及边的权值

```
4 3
A B C D
A B 1
B C 1
B D 1
```

- 首先根据结点数创建一个结构体数组ghv1，输入的顶点信息存入结构体数组中的数据域，每一对顶点，找到对应顶点的链表，将另外一个顶点存储链表中，交换两个顶点，进行相同的操作。
- 每一对顶点都存储结构体后得到图的邻接表存储结构

构建出的图的邻接表结构如下：



算法描述

- 创建一个结构体数组，内部包含存储数据域顶点信息的变量和顶点对应的链表
- 按照每一对顶点之间的关系，将和每一个顶点有关的点都存入这个顶点的链表中
- 交换两个顶点的位置，在其对称位置也进行存储





湖南大学

HUNAN UNIVERSITY

谢谢观看

参考文献

【1】 Clifford A.Shaffer 著 张铭 刘晓丹 等译 。 数据结构与算法分析【M】。北京：电子工业出版社，2013： 5-8.

【2】 Clifford A.Shaffer 著 张铭 刘晓丹 等译 。 数据结构与算法分析【M】。北京：电子工业出版社，2013： 138-139.

【3】 严蔚名 吴伟民 著 。 数据结构（C语言版）【M】。北京：清华大学出版社，1997： 160-164.

【4】 百度百科