

第四章 数据结构

- 4.1 基本概念
- 4.2 线性表
- 4.3 栈和队列
- 4.4 树
- 4.5 二叉树
- 4.6 二叉树的线性表示及生成
- 4.7 任意次树与二叉树之间的转换
- 4.8 图

Algorithm + *Data Structure* = *Programs*

程序设计	为计算机处理问题编制一组指令集
算法	处理问题的方法
数据结构	问题的数学模型

第四章 数据结构

- 4.1 基本概念
- 4.2 线性表
- 4.3 栈和队列
- 4.4 树
- 4.5 二叉树
- 4.6 二叉树的线性表示及生成
- 4.7 任意次树与二叉树之间的转换
- 4.8 图

第四章 数据结构

- 4.1 基本概念
 - 4.1.0 数据结构的研究内容
 - 4.1.1 数据结构的定义
 - 4.1.2 结点定义
 - 4.1.3 数据结构的存储方式
 - 4.1.4 数据结构的分类
 - 4.1.5 基本操作
 - 4.1.6 数据结构的学习内容
- 4.2 线性表
- 4.3 栈和队列
- 4.4 树
- 4.5 二叉树
- 4.6 二叉树的线性表示及生成
- 4.7 任意次树与二叉树之间的转换

➤4.1 基本概念

➤4.1.0 数据结构的研究内容

➤数据(对象)的性质

什么样的数据/集合

➤数据联系

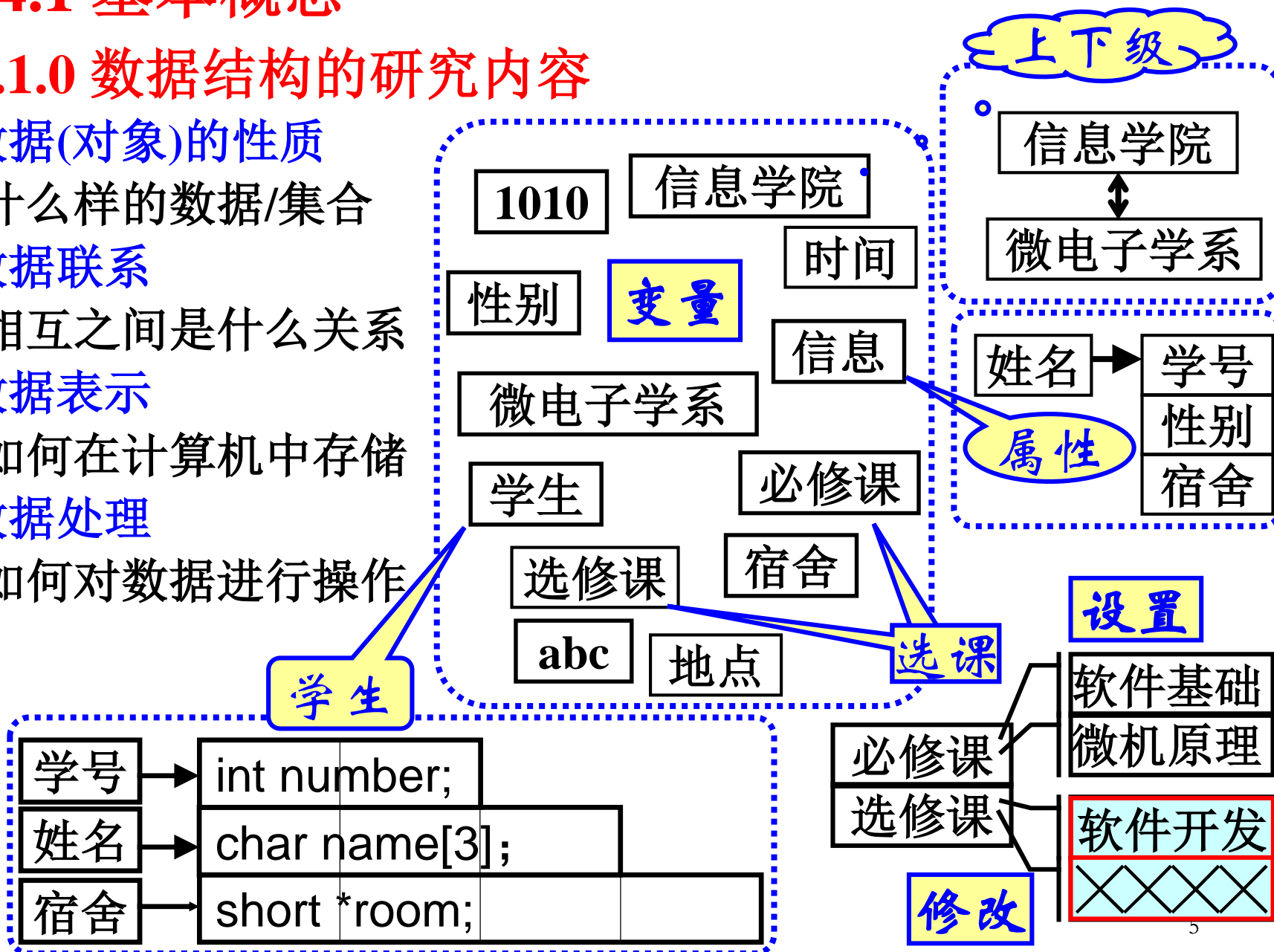
相互之间是什么关系

➤数据表示

如何在计算机中存储

➤数据处理

如何对数据进行操作



➤4.1.1 数据结构的定义

➤数据结构的定义

设 $B=(K, R)$ 是一个二元组， K 是数据的有限集合， R 是 K 中各数据的有限关系集合，则称 B 是数据结构。

其中， K 的成员 k_1, k_2, \dots ，又称为数据元素(或结点)，记为

$$K=\{k_1, k_2, \dots, k_n\}$$

R 的成员 r_1, r_2, \dots ，称为关系，记为

$$R=\{r_1, r_2, \dots, r_m \mid r_i = (k_s, k_t), k_s, k_t \in K\}$$

表示在结点 k_s, k_t 之间存在关系 r_i 。

➤4.1.1 数据结构的定义

【例4-1.1】数据结构示例。

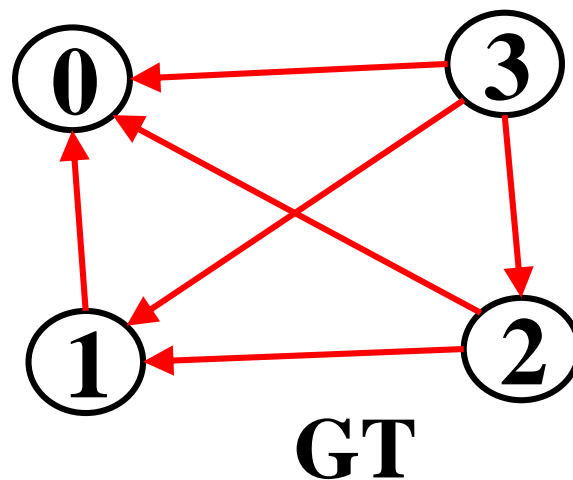
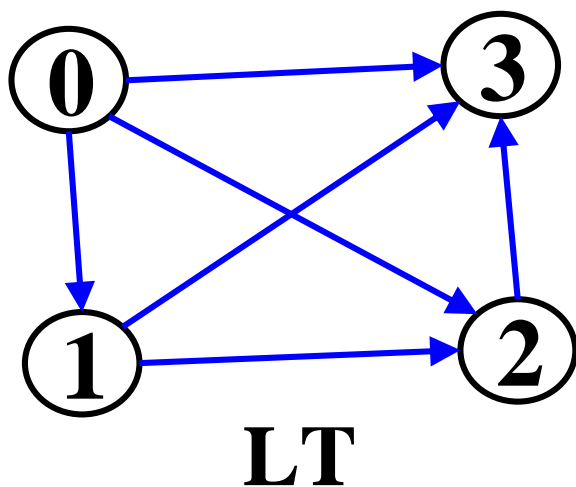
0~3可组成集合K，在集合 $K=\{0\sim3\}$ 上可存在GT(大于)和LT(小于)两种关系。

定义： $B=(K, R)$, $K=\{0, 1, 2, 3\}$, $R=\{GT, LT\}$,

令 $GT = \{(1, 0), (2, 0), (2, 1), \dots\}$,

$LT = \{(0, 1), (0, 2), (1, 2), \dots\}$,

则称 $B=(K, R)$ 是一种数据结构。



➤4.1.2 结点定义

设 $B=(K, R)$ 是一种数据结构，可有以下定义：

α_k	δ_k	p_k
------------	------------	-------

结点 k 的存储单元

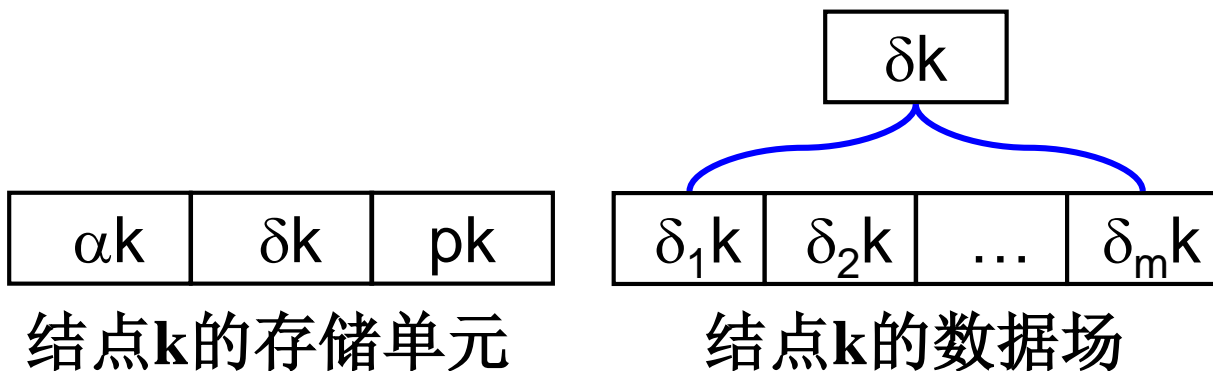
➤结点的存储单元

结点 k 的存储单元由地址 α_k ，数据场 δ_k 和指针场 p_k 组成，记为：

$$k = \{\alpha_k, \delta_k, p_k\}。$$

➤4.1.2 结点定义

设 $B=(K, R)$ 是一种数据结构，可有以下定义：



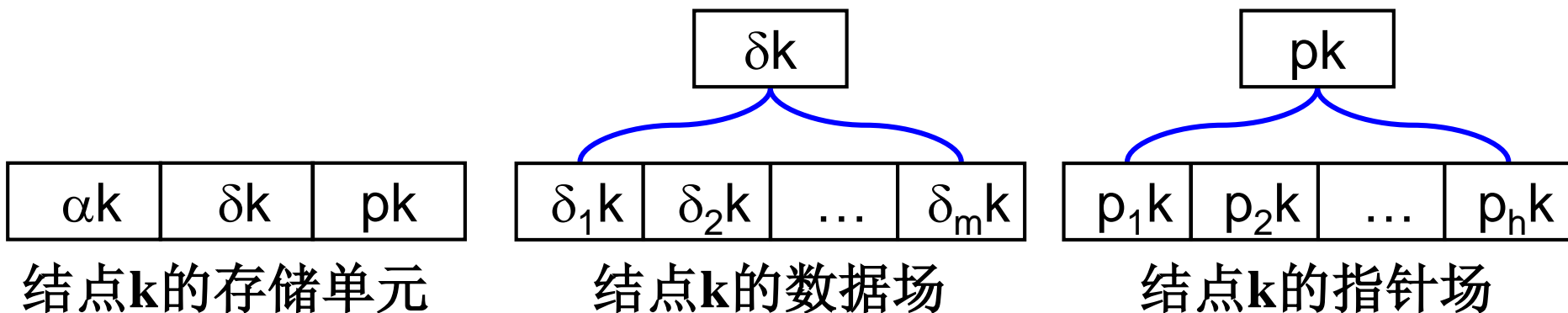
➤结点的数据场

若结点 k 的数据场 δk 有 m 个分量，即每个结点都有 m 个不同的数据，则记为：

$$\delta k = \{\delta_i k \mid i=1, \dots, m\} = \{\delta_1 k, \delta_2 k, \dots, \delta_m k\}$$

➤4.1.2 结点定义

设 $B=(K, R)$ 是一种数据结构，可有以下定义：



➤结点的指针场

若结点 k 的指针场 $p k$ 有 h 个分量，每个指针场分量都对应于 R 的一个关系 r_j , ($j=1, \dots, h$), 则记为：

$$p k = \{p_j k \mid j=1, \dots, h\} = \{p_1 k, p_2 k, \dots, p_h k\}$$

➤4.1.2 结点定义

➤结点的指针场

● 指针场分量

假定对应于一种关系 r_j 的指针场 p_jk ，需要指向 $t(j)$ 个结点，即：

$$p_jk = \{p_{jk}k \mid k=1, \dots, t(j)\} = \{p_{j1}k, \dots, p_{jt(j)}k\},$$

则 pk 又可以进一步记为

$$pk = \{p_{11}k, \dots, p_{1t(1)}k, p_{21}k, \dots, p_{2t(2)}k, \dots, p_{h1}k, \dots, p_{ht(h)}k\}$$

从而

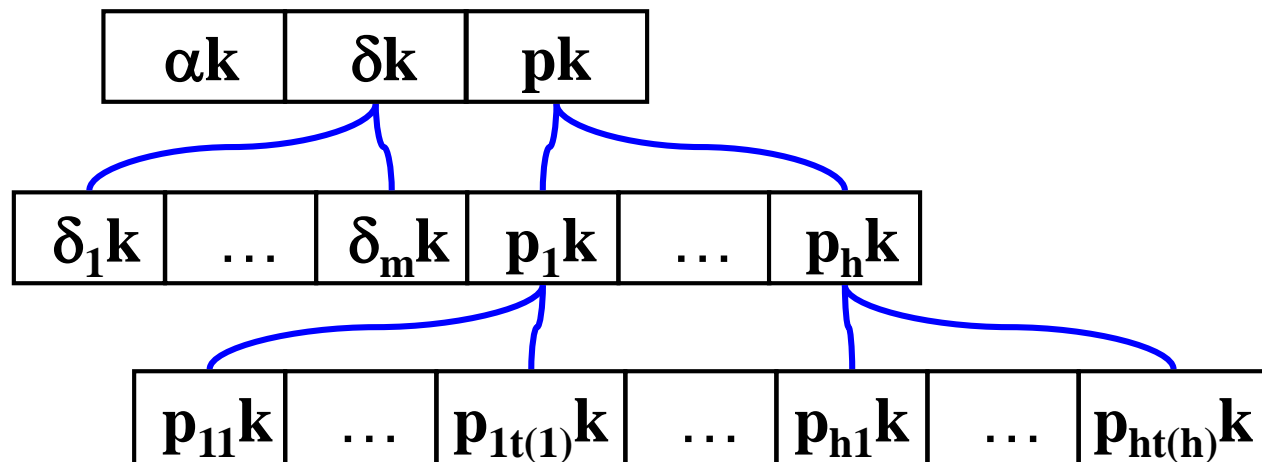
$$k = \{\alpha k, \delta_1k, \delta_2k, \dots, \delta_mk, p_{11}k, \dots, p_{1t(1)}k, p_{21}k, \dots, p_{2t(2)}k, \dots, p_{h1}k, \dots, p_{ht(h)}k\}$$

➤4.1.2 结点定义

➤结点的指针场

●空置的指针场

用 ϕ 或NULL表示空置的指针场。例如，记为 $p_i k = \phi$ 或 $p_i k = \text{NULL}$ 。



$p_i k$
 ϕ
或
 $p_i k$
NULL

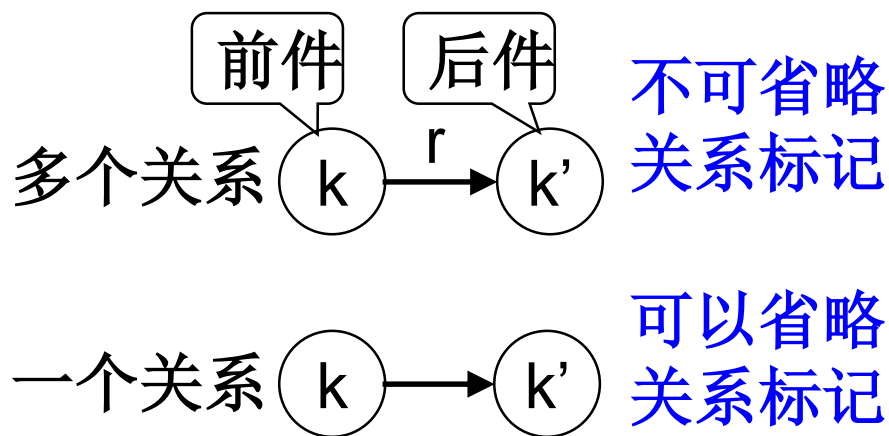
➤4.1.2 结点定义

➤结点性质

●前件和后件

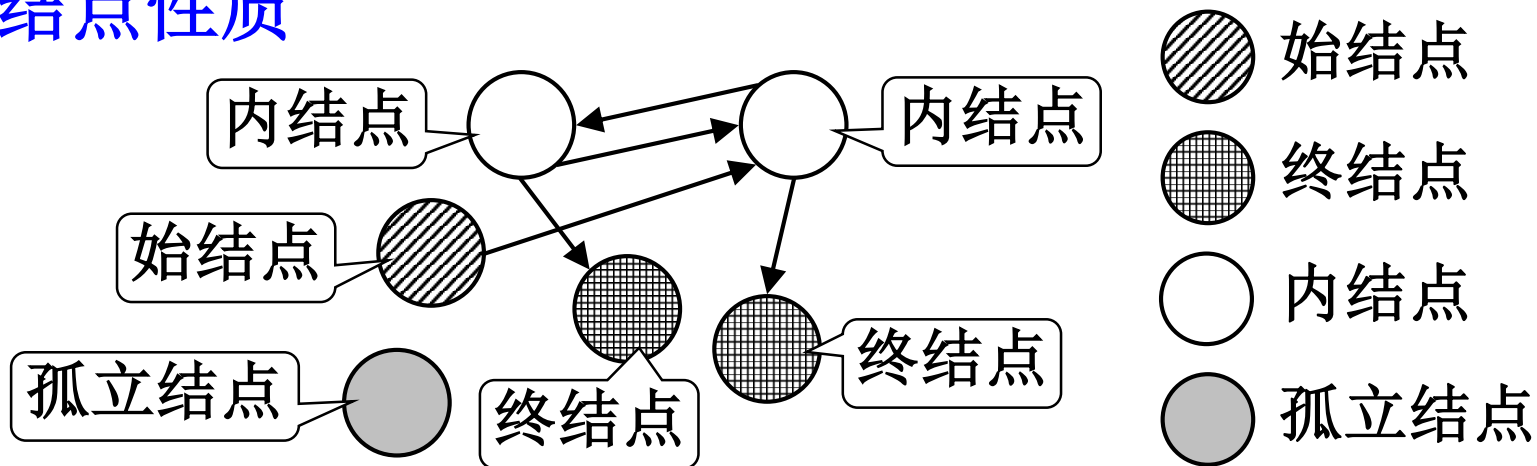
设 $r = (k, k')$ ，其中 $r \in R, (k, k' \in K)$ ，则称 k 是 k' 关于关系 r 的前件， k' 是 k 关于关系 r 的后件。图中用箭头从 k 指向 k' 。

当 R 中只有一种关系 r 时，可以省略关系的名称，并直接称 k 是 k' 的前件， k' 是 k 的后件。



➤4.1.2 结点定义

➤结点性质

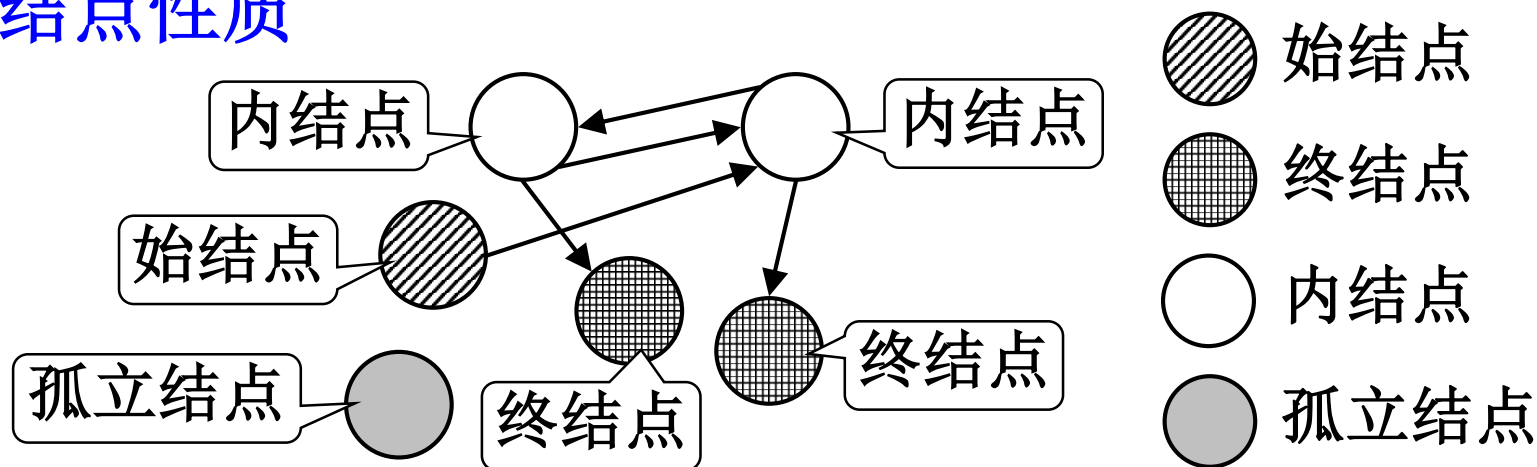


● 孤立结点

若在 K 中对于关系 r 而言，既无前件又无后件的 k 称为孤立结点。

➤4.1.2 结点定义

➤结点性质



- 孤立结点

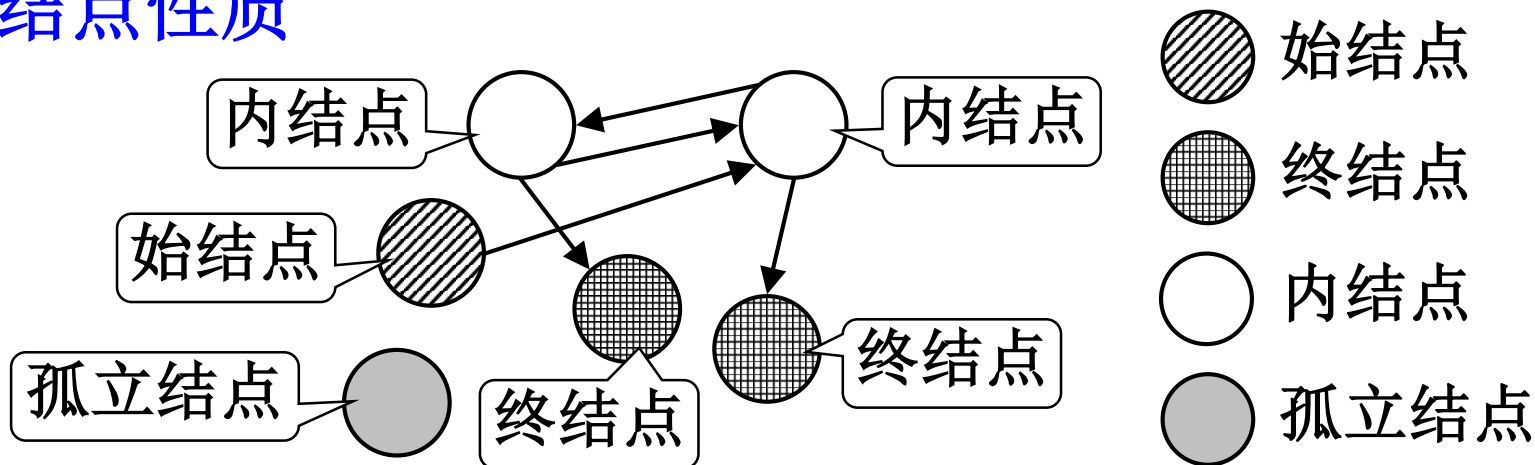
对于非孤立的结点，定义：

- 始结点

若在 K 中不存在任何 k' ，使得成立 $(k', k) \in r$ ，则称 k 为 r 的始结点。

➤4.1.2 结点定义

➤结点性质



- 孤立结点

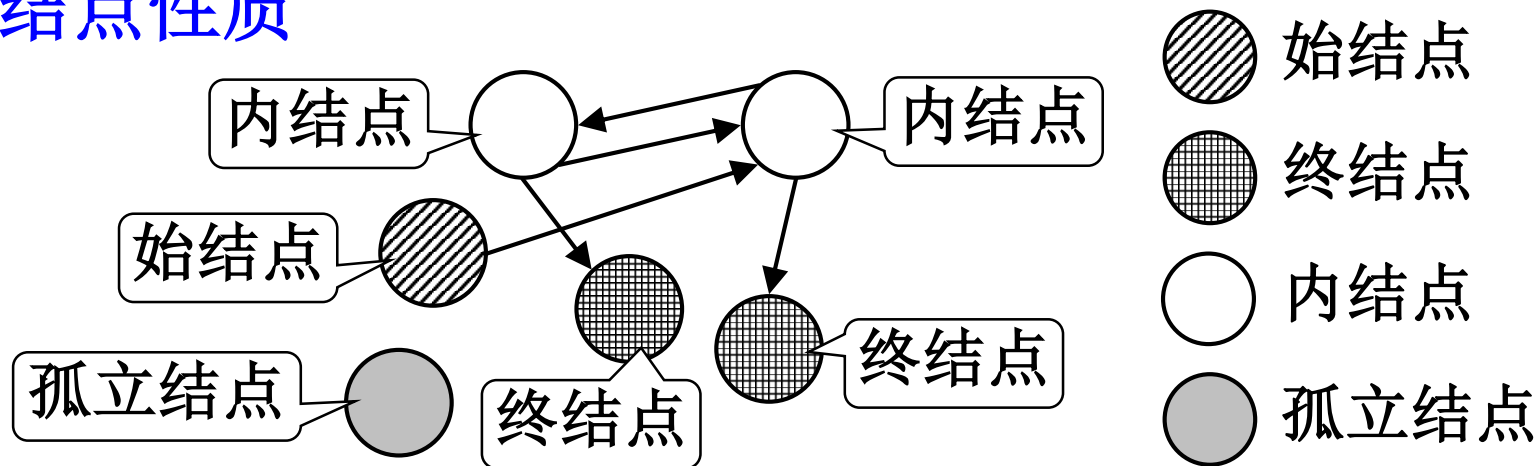
对于非孤立的结点，定义：

- 始结点
- 终结点

若在 K 中不存在任何 k' ，使得成立 $(k, k') \in r$ ，则称 k 为 r 的终结点。

➤4.1.2 结点定义

➤结点性质



- 孤立结点

对于非孤立的结点，定义：

- 始结点
- 终结点
- 内结点

既非始结点又非终结点的称为r的内结点。

【例4-1.2】 结点性质示例

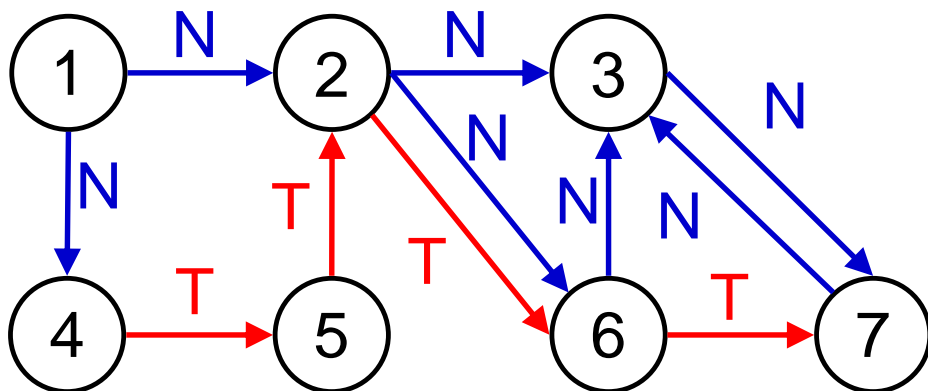
定义 $B=(K,R)$ ，其中

$K=\{1, 2, 3, 4, 5, 6, 7\}$,

$R = \{N,T\}$,

$N = \{(1,2), (1,4), (2,3), (2,6), (3,7), (6,3), (7,3)\}$,

$T = \{(2,6), (4,5), (5,2), (6,7)\}$ 。



	关系N	关系T
始结点		
终结点		
内结点		
孤立结点		

结点	关系N		关系T	
	前件	后件	前件	后件
1				
2				
3				
4				
5				
6				
7				

【例4-1.2】结点性质示例

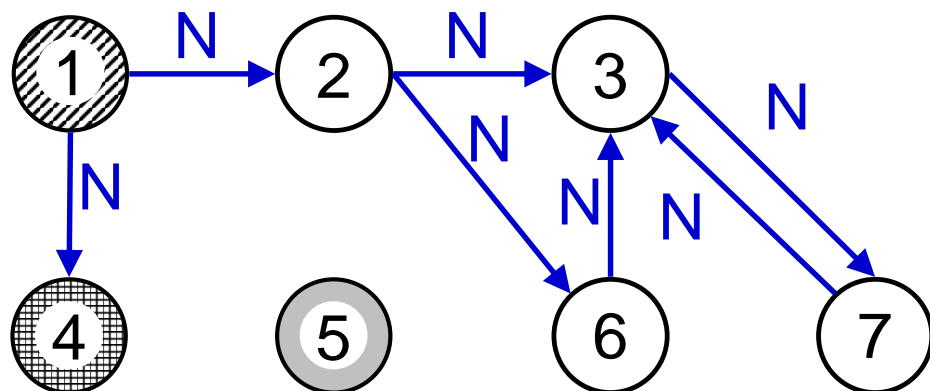
定义 $B=(K,R)$ ，其中

$K=\{1, 2, 3, 4, 5, 6, 7\}$,

$R = \{N,T\}$,

$N = \{(1,2), (1,4), (2,3), (2,6), (3,7), (6,3), (7,3)\}$,

$T = \{(2,6), (4,5), (5,2), (6,7)\}$ 。



始结点 终结点
 内结点 孤立结点

	关系N	关系T
始结点	1	
终结点	4	
内结点	2,3,6,7	
孤立结点	5	

结点	关系N		关系T	
	前件	后件	前件	后件
1		2,4		
2	1	3,6		
3	2,6,7	7		
4	1			
5				
6	2	3		
7	3	3		

【例4-1.2】结点性质示例





定义 $B1=(K,R)$ ，其中

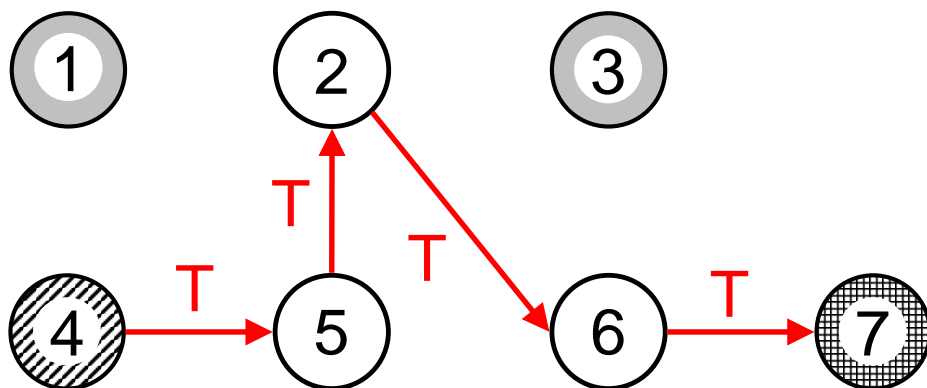
$K=\{1, 2, 3, 4, 5, 6, 7\}$,

$R = \{N,T\}$,

$N = \{(1,2), (1,4), (2,3), (2,6), (3,7), (6,3), (7,3)\}$,

$T = \{(2,6), (4,5), (5,2), (6,7)\}$ 。

 始结点  终结点
 内结点  孤立结点



	关系N	关系T
始结点	1	4
终结点	4	7
内结点	2,3,6,7	2,5,6
孤立结点	5	1,3

结点	关系N		关系T	
	前件	后件	前件	后件
1		2,4		
2	1	3,6	5	6
3	2,6,7	7		
4	1			5
5			4	2
6	2	3	2	7
7	3	3	6	

➤4.1.3 数据结构的存储方式

假定 r 是 $B=(K,R)$ 中的一种关系，实现 r 的存储方式有以下两种：

➤顺序存储

若对于任何一对结点 (k,k') 都成立 $\alpha k' = \alpha k + s$ (s 为存储单元的大小,固定长度), 则称 r 是用顺序方式存储的,简称顺序存储。

➤链接存储

若对于任何一对结点 (k,k') 都成立 $\alpha k' \in pk$, 则称 r 是用链接方式存储的, 简称链接存储。

顺序存储
 $s=2$

$\alpha k_1 = \text{FF00}$

$\alpha k_2 = \text{FF02}$

$\alpha k_3 = \text{FF04}$

链接存储

$\alpha k_1 = \text{FF60}$

$pk_1 = \text{FF10}$

$\alpha k_2 = \text{FF10}$

$pk_2 = \text{FF00}$

$\alpha k_3 = \text{FF00}$

$pk_3 = \text{FF40}$

➤4.1.3 数据结构的存储方式

结 点	关系 N		关系 T	
	前件	后件	前件	后件
1		2,4		
2	1	3,6	5	6
3	2,6,7	7		
4	1			5
5			4	2
6	2	3	2	7
7	3	3	6	

例如：

在N中各结点的最多后件数大于1，则只能用链接存储。

在T中，各结点的最多后件数不超过1，则既可用顺序存储，也可用链接存储。

【例4-1.3】存储单元的图形表示

在数据结构 $B1=(K, R)$ 中，每个结点 k 的数据场只有一个分量 δk 。

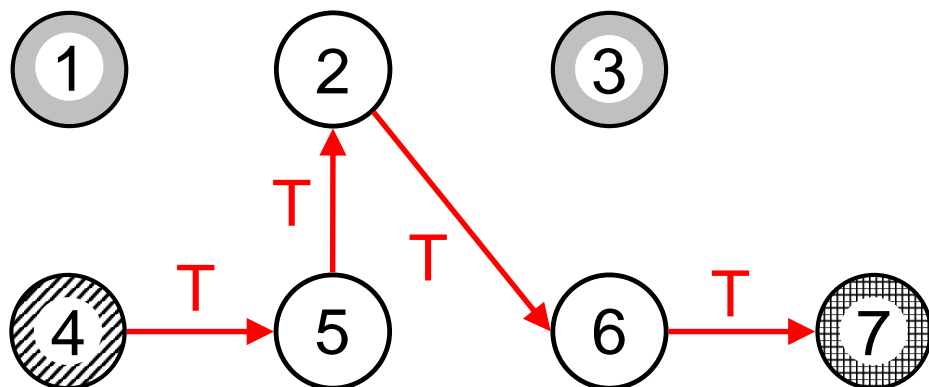
由于所有结点关于关系 N 的后件最多有2个，关于关系 T 的后件均为1个，则 k 的存储单元为：

$k = \{\alpha k, \delta k, (p_{N1}k, p_{N2}k), p_Tk\}$ 。

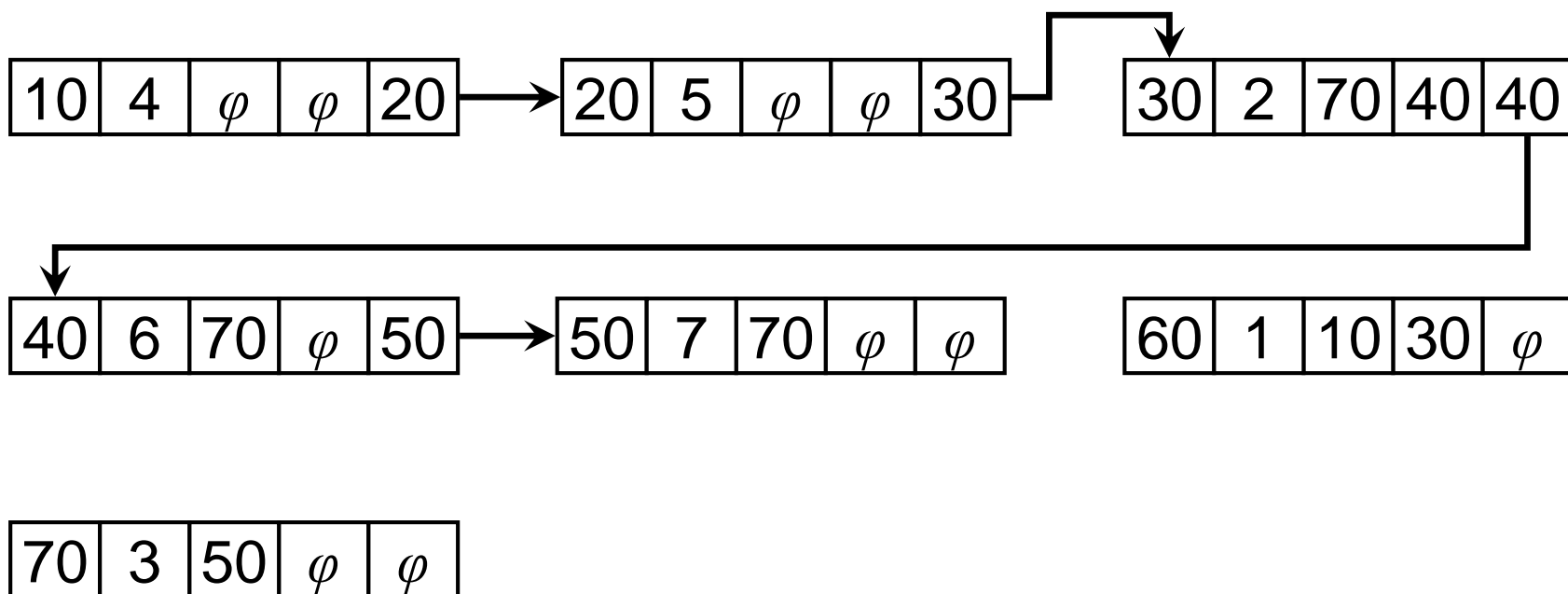


现以链接方式存储关系 N ，以顺序方式存储关系 T ， $B1=(K, R)$ 的存储单元如图所示（虚设存储单位的地址）。

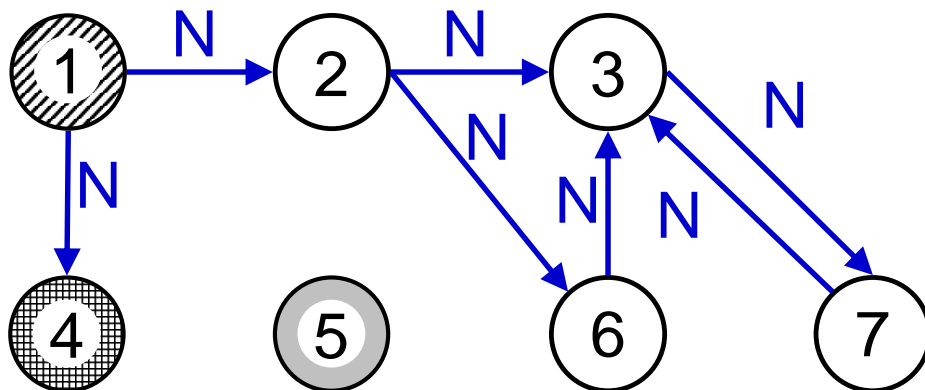
【例4-1.3】存储单元的图形表示



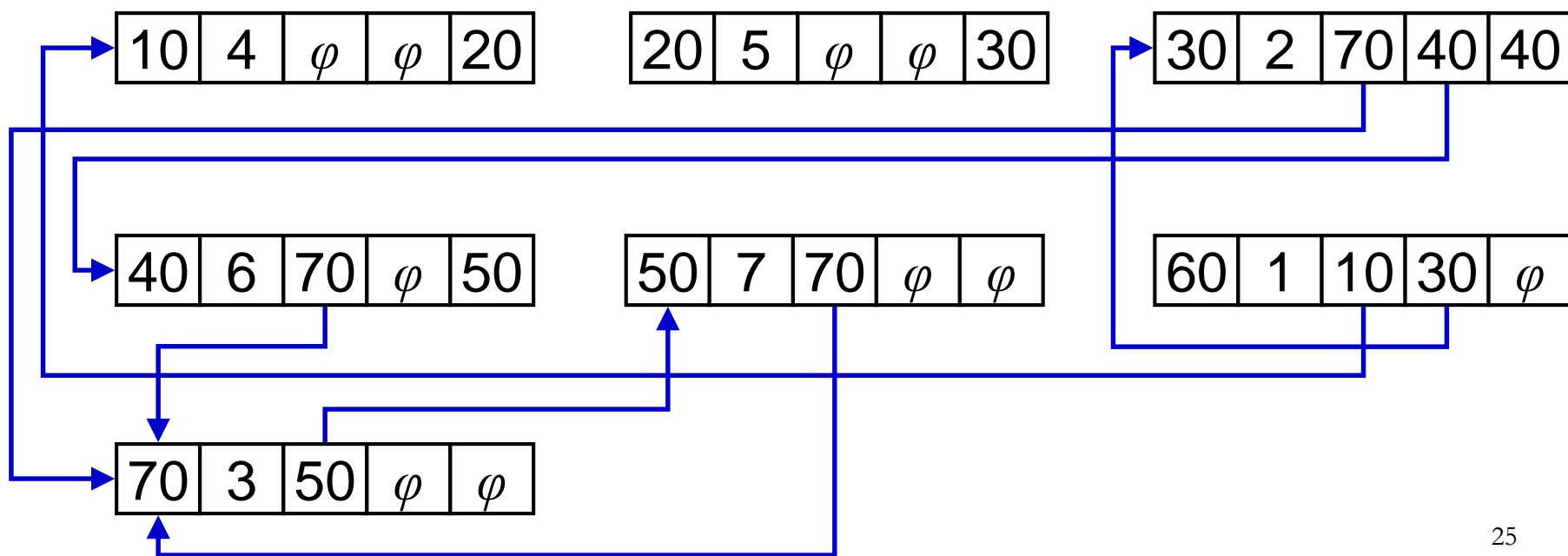
现以链接方式存储关系N，以顺序方式存储关系T， $B1=(K, R)$ 的存储单元如图所示（虚设存储单位的地址）。



【例4-1.3】存储单元的图形表示



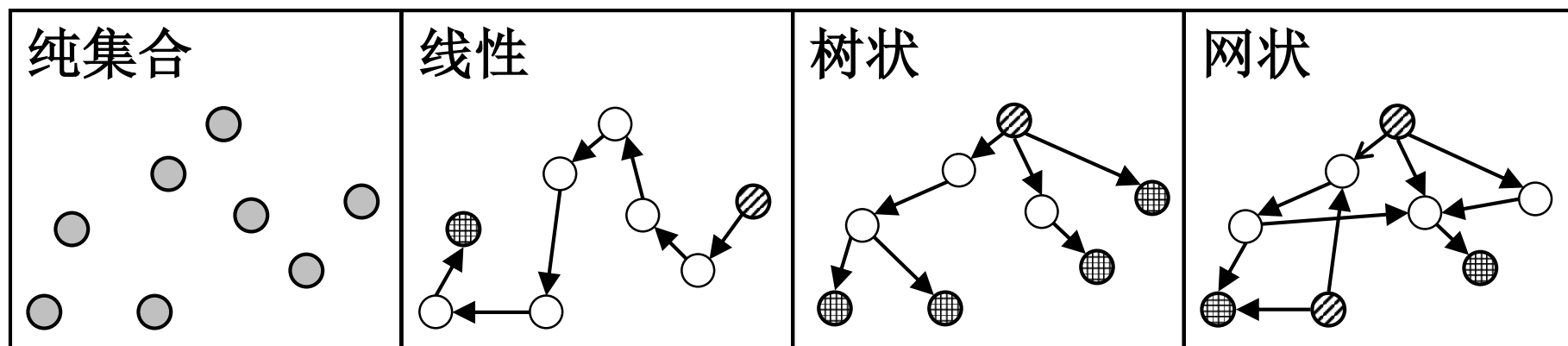
现以链接方式存储关系 N ，以顺序方式存储关系 T ， $B1=(K, R)$ 的存储单元如图所示（虚设存储单位的地址）。



➤4.1.4 数据结构的分类

根据结点的相互关系来分类

- (1) **纯集合**：结点间相互无关系，全部为孤立结点。
- (2) **线性**：只有一个始结点和一个终结点，各结点的前件和后件数至多一个，无孤立结点。
- (3) **树状**：只有一个始结点，各结点的前件数至多一个，无孤立结点。
- (4) **网状**：始结点和终结点的数目任意，各结点的前件和后件数任意，无孤立结点。

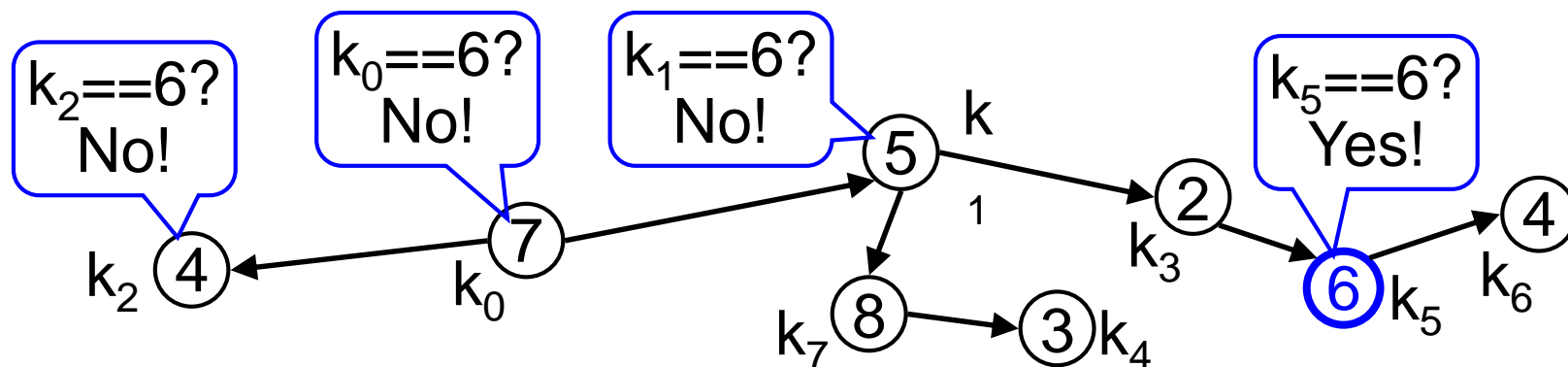


▨ 始结点 ▩ 终结点 ○ 内结点 ● 孤立结点

➤4.1.5 基本操作

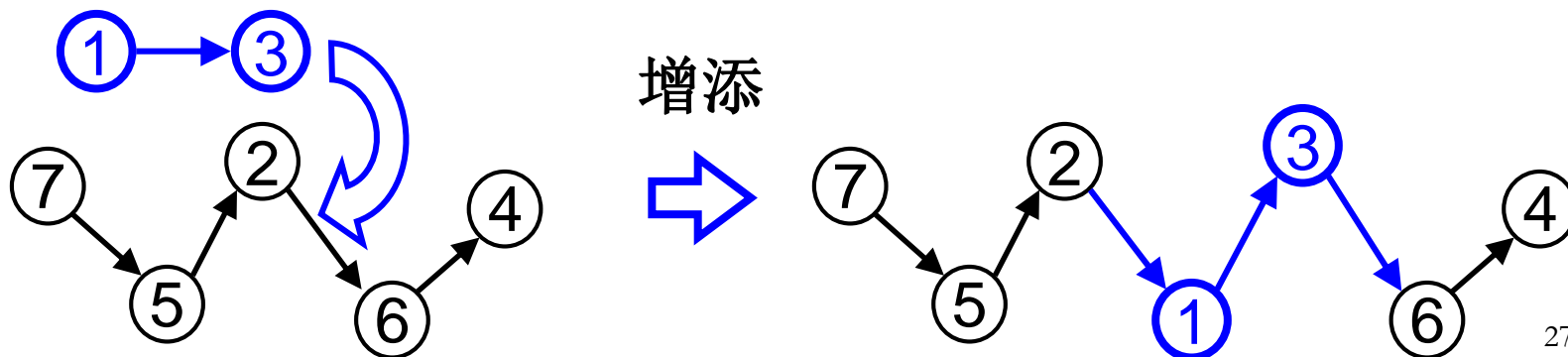
数据结构的基本操作包括查找，增添，删除和排序。

- (1) **查找**：根据地址、序号或某个数据场的值来寻找某个结点。
例如，查找结点值为 $\text{key}=6$ 的结点，即查找 $k_i==6$ ，得 $i=5$ 。



- (2) **增添**：根据某种关系，在保持数据结构的特点不变情况下，增加结点或结点组。

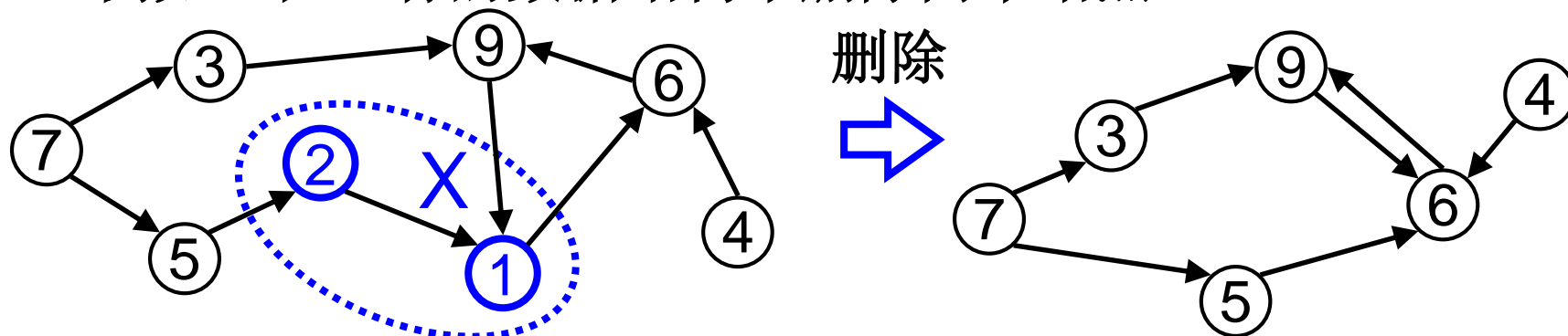
例如，在已有的数据结构中增添两个结点：



➤4.1.5 基本操作

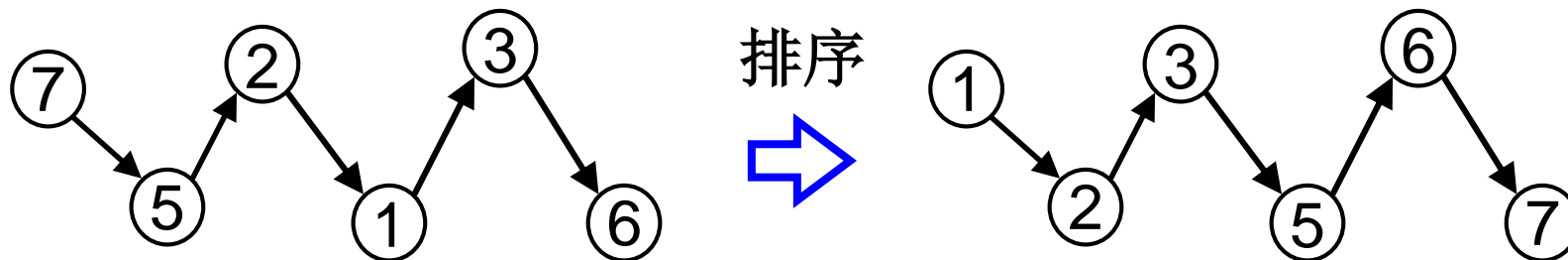
(3) **删除**：在保持数据结构的特点不变情况下，删去某些结点或结点组。

例如，在已有的数据结构中删除两个结点：



(4) **排序**：按某种方式使各结点有序排列。

例如，使数据结构中的结点按照从小到大排序：



➤ 4.1.5 基本操作

➤ 算法复杂度

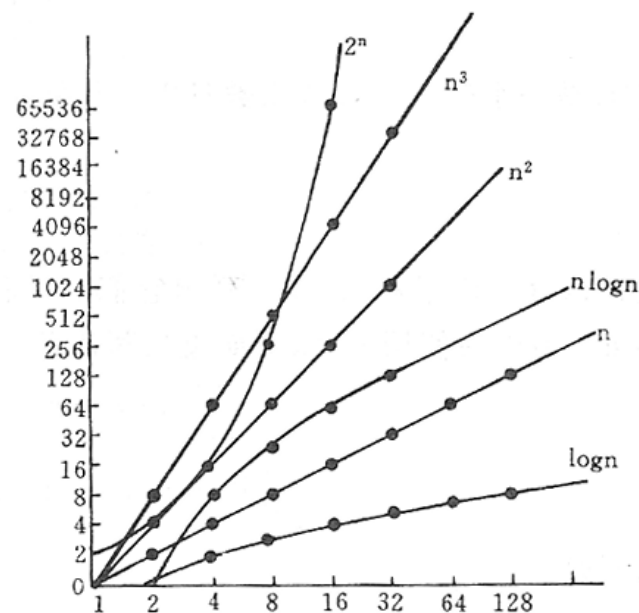
- 当问题规模（即要处理的数据）增长时，基本操作要重复执行的次数（时间）必定也会增长，
- 我们关心的是这个执行次数以什么样的数量级增长。
- 大O表示法，“O”表示order
 - n 表示要处理数据规模
 - 算法时间（空间）复杂度表示成 n 的函数 $O(f(n))$ ，表征算法时间随数据规模的增长趋势，如：

常数复杂度 $O(C)$

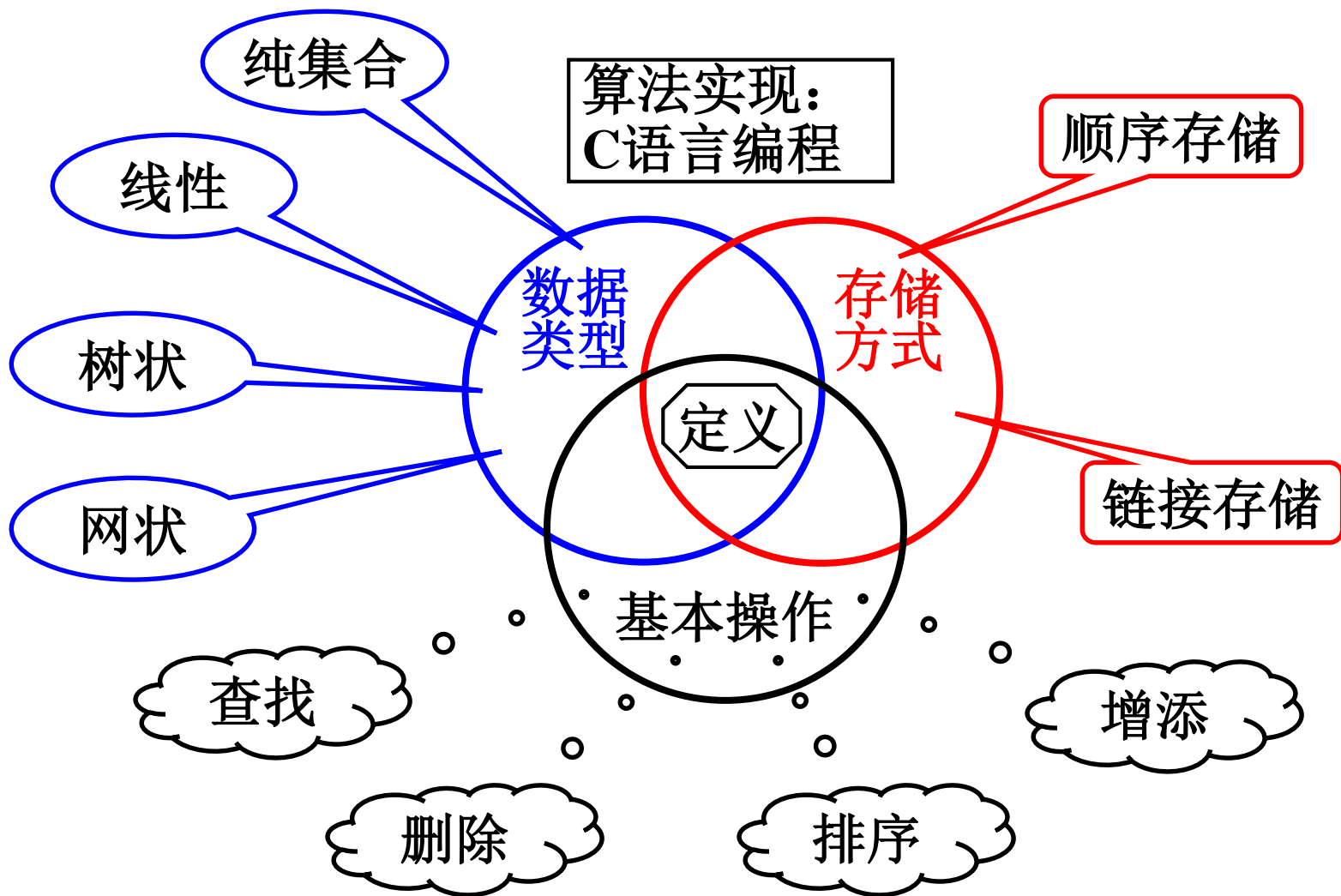
线性复杂度 $O(n)$

平方复杂度 $O(n^2)$

...



➤4.1.6 数据结构的学习内容



➤ 作业

3-3(学生成绩)

➤ 上机题