计算机专业导论

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

第6讲 程序与递归：组合-抽象-构造

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

本讲学习什么?

---程序与递归：组合-抽象-构造

战德臣

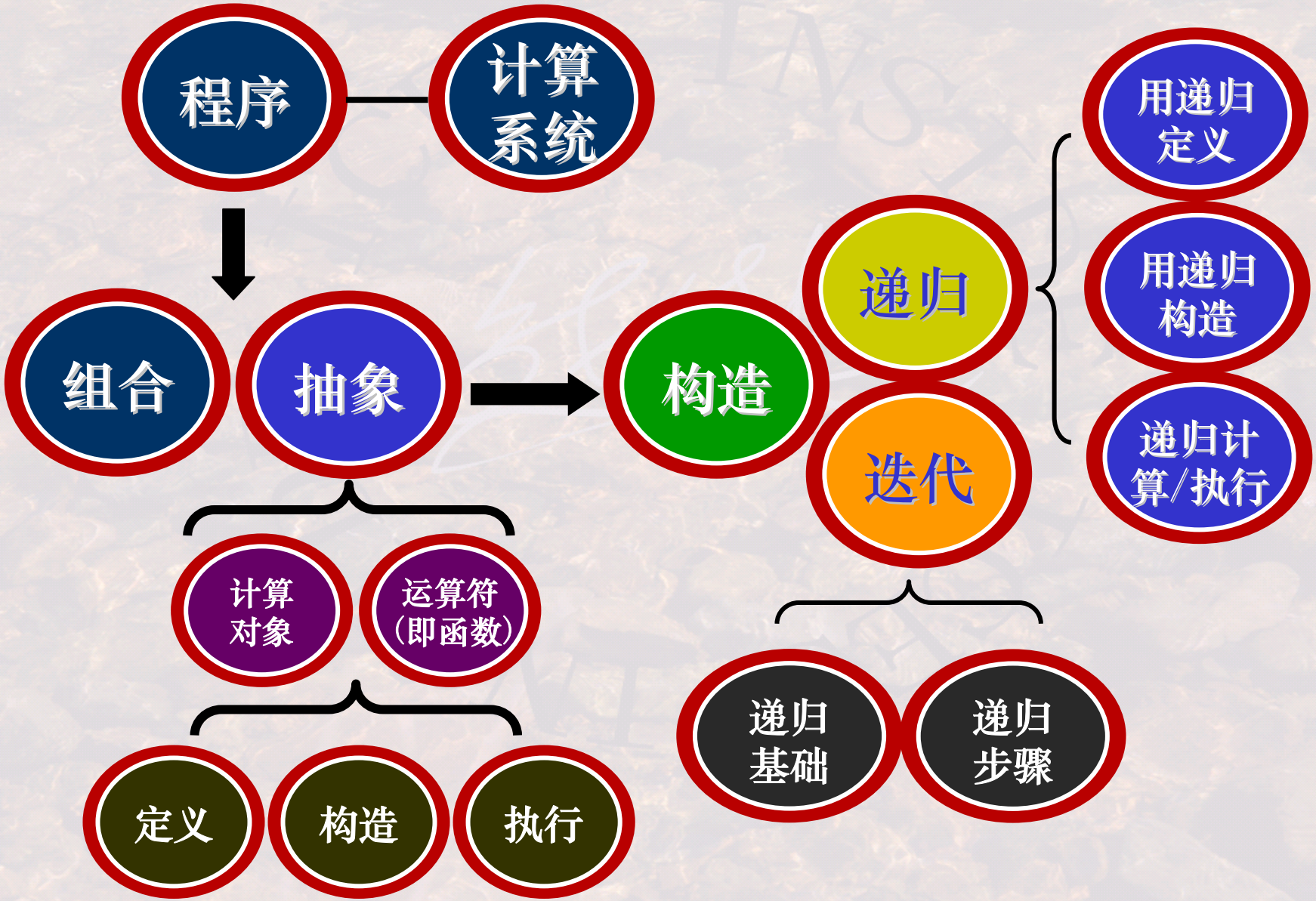
哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

程序与递归：组合-抽象-构造
本讲概述

ICES 战德臣 教授



计算系统与程序

---程序的作用和本质

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

战德臣 教授

什么是程序?

程序的本质是什么?

组合 抽象 → 构造

战德臣 教授

如何设计实现一个基本计算系统?

首先，设计并实现系统可以执行的基本动作(可实现的)，例如

- “与”动作
- “或”动作
- “非”动作
- “异或”动作

已知的基本事实是:

“加减乘除运算都可转换为加减法运算来实现”

“加减法运算又可以转换为逻辑运算来实现”

那么，复杂的动作呢?

- 系统需要提供复杂的动作
- 复杂的动作千变万化
- 复杂的动作随使用者使用目的的不同而变化

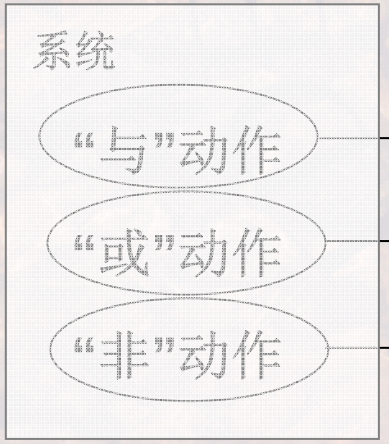
如何设计实现一个基本计算系统?

程序:由基本动作指令构造的，若干指令的一个组合或一个执行序列，用以实现复杂动作

复杂动作

((A AND B) AND C) OR (NOT C))

拆解开



AND

OR

NOT

X= A AND B
X= X AND C
Y= NOT C
X= X OR Y

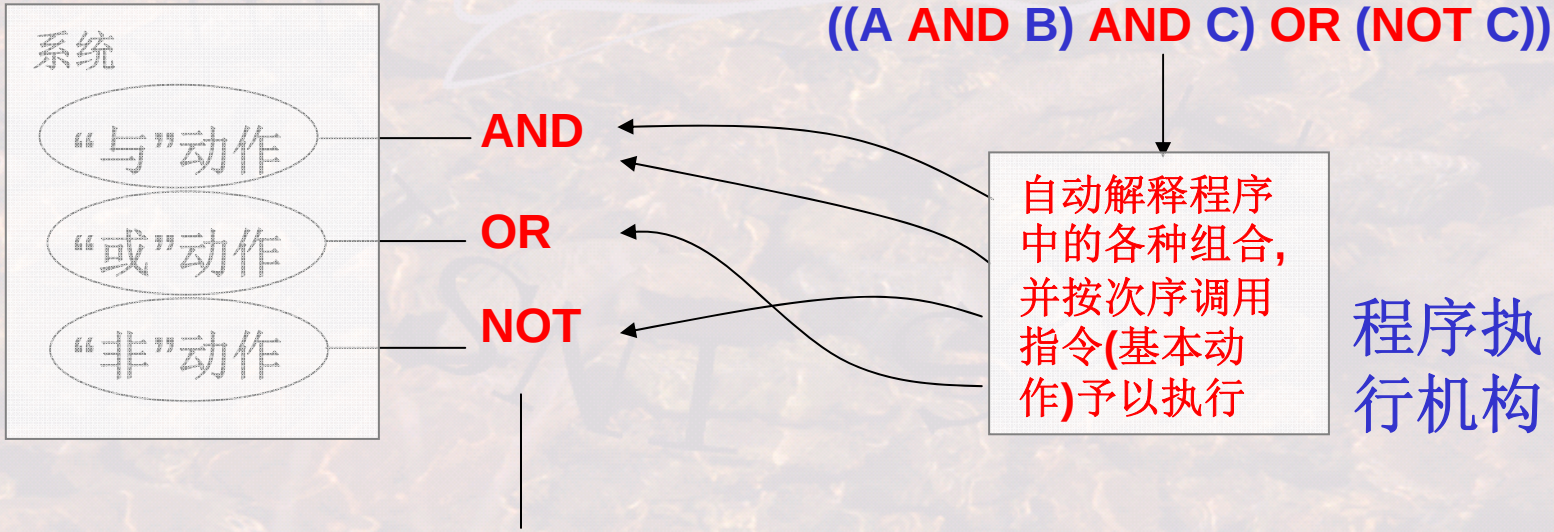
指令：控制基本动作执行的命令

如何设计实现一个基本的计算系统?

程序:由基本动作指令构造的，若干指令的一个组合或一个执行序列，用以实现复杂动作

复杂动作

((A AND B) AND C) OR (NOT C))



指令：控制基本动作执行的命令

(4) 计算系统与程序?

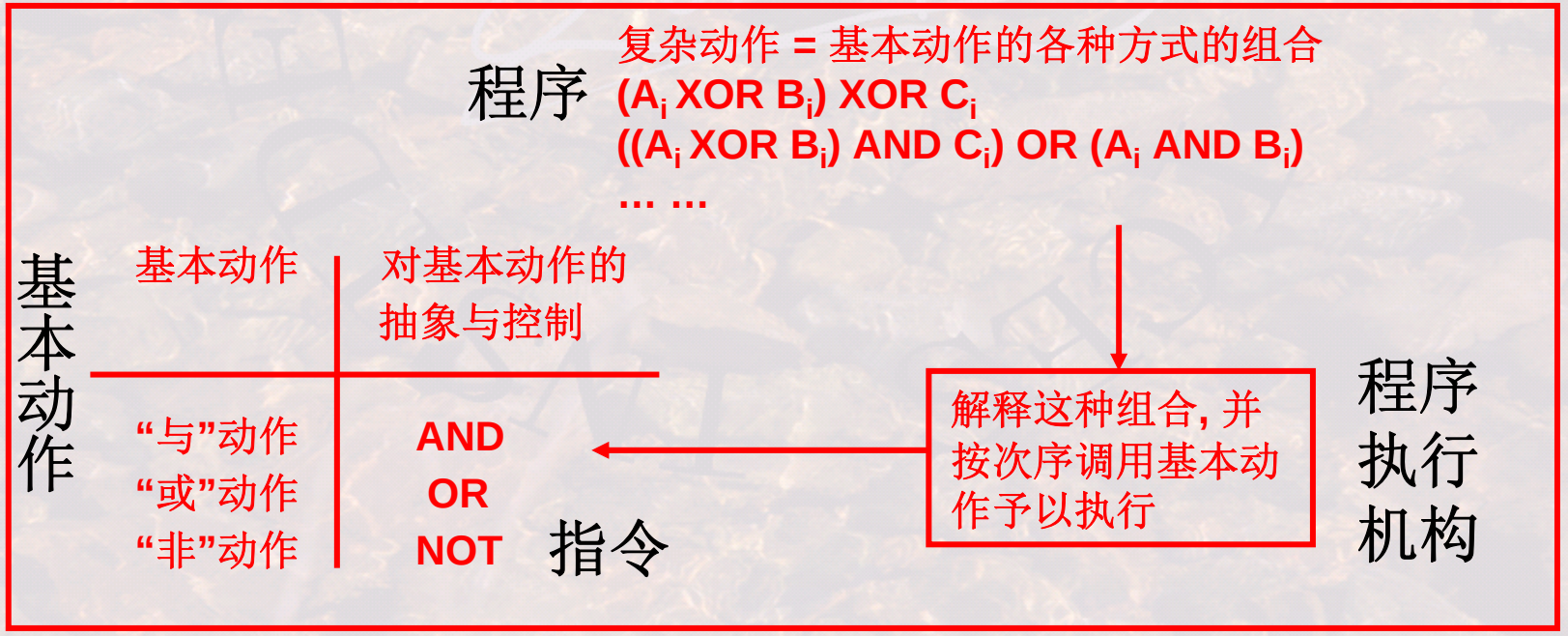
ICES 战德臣 教授

计算系统 = 基本动作 + 指令 + 程序执行机构

指令 = 对可执行基本动作的抽象，即控制基本动作执行的命令

程序 = 基本动作指令的一个组合或执行序列，用以实现复杂的动作

程序执行机构 = 负责解释程序即解释指令之间组合，并按次序调用指令即调用基本动作执行的机构

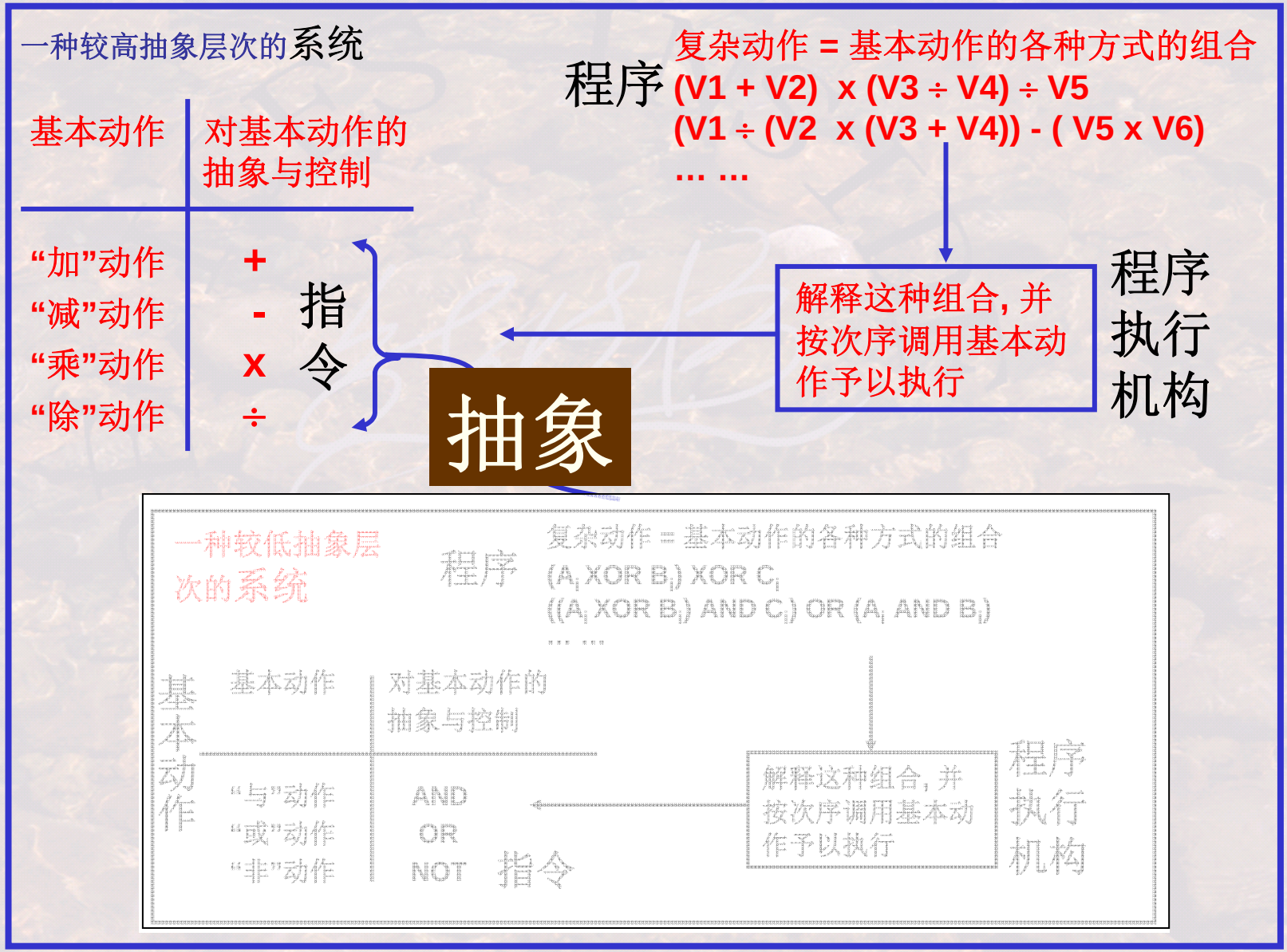


计算系统与程序-程序的作用和本质

(5) 程序：组合-抽象-构造?

ICES 战德臣 教授

抽象：将经常使用的、可由低层次系统实现的一些复杂动作，进行**命名**，以作为高层次系统的指令被使用



运算式的组合-抽象与构造

战德臣

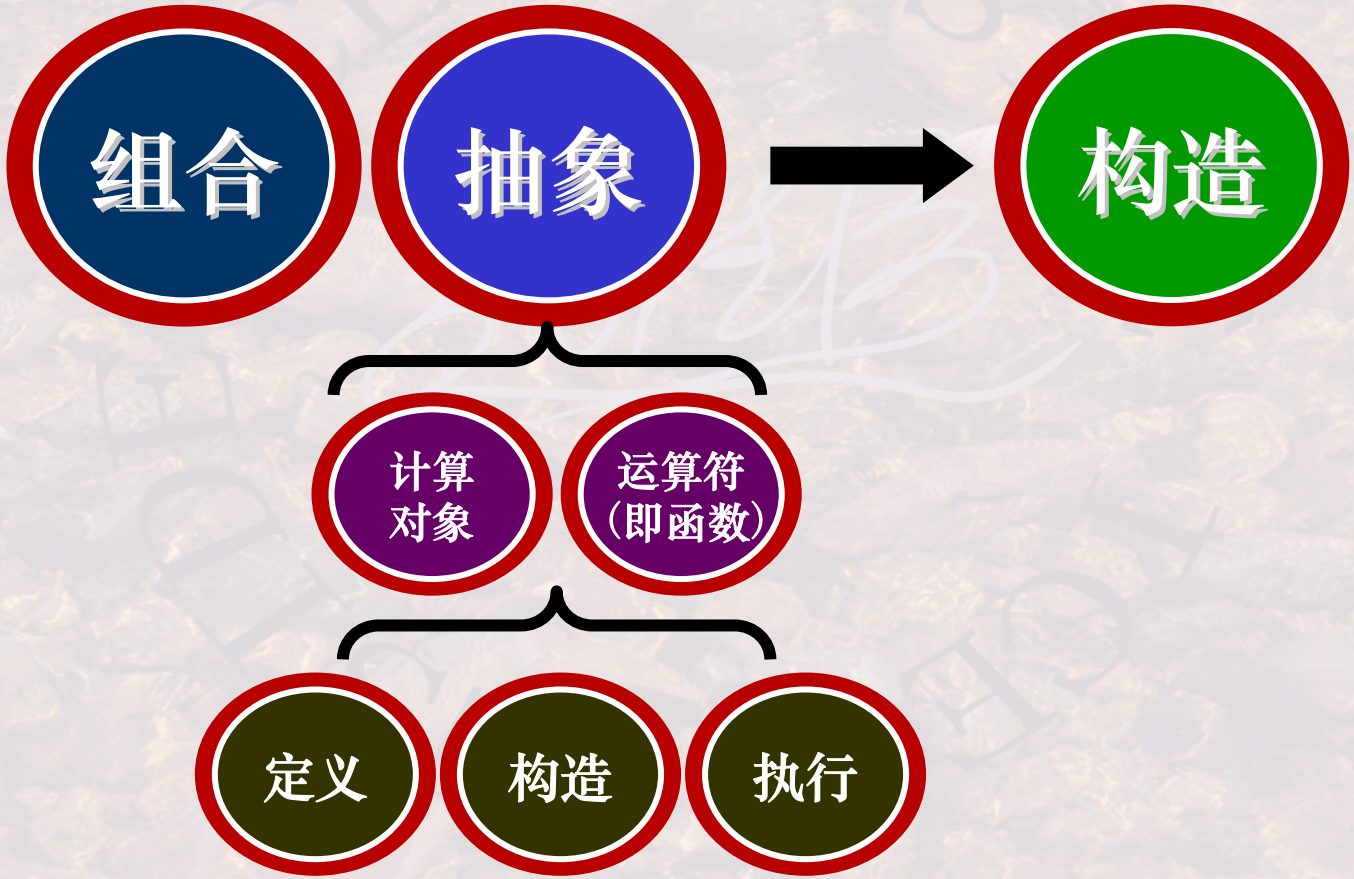
哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

运算式的组合-抽象与构造---程序构造示例
(7)小结

战德臣 教授



运算式的组合-抽象与构造

---程序构造示例I-计算对象的定义-构造与计算

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent
Computing for Enterprises & Services,
Harbin Institute of Technology

运算式的组合-抽象与构造---程序构造示例
(1)运算组合式?

战德臣 教授

由数值，到基本运算组合式

100

205

实际的数值

(100 + 205)

中缀表示法, 用运算符(即前述的指令)
将两个数值组合起来，运算符在中间

(+ 100 205)

前缀表示法, 用运算符(即前述的指令)
将两个数值组合起来，运算符在前面

将运算符表示的操作应用于后面的一组数值上，求出结果

(+ 100 205 300 400 51 304)

一个运算符可以表示连加，
连减等情况，

运算式的组合-抽象与构造---程序构造示例
(1)运算组合式?

战德臣 教授

由数值，到基本运算组合式

(+ 100 205)

(- 200 50)

(* 200 5)

(* 20 5 4 2)

(- 20 5 4 2)

(+ 20 5 4 2)

一起练习,书写程序, … …

战德臣 教授

运算组合式的“嵌套”及其计算过程

(+ 100 205)

(+ (+ 60 40) (- 305 100))

((* 3 (+ (* 2 4) (+ 3 5))) (+ (- 10 7) 6))*

计算过程

((* 3 (+ (* 2 4) (+ 3 5))) (+ (- 10 7) 6))*

((* 3 (+ 8 8)) (+ 3 6))*

((* 3 16) 9)*

(48 9)*

432

运算式的组合-抽象与构造---程序构造示例
(3)如何用名字简化运算组合式的构造?--抽象?

战德臣 教授

命名计算对象和构造中使用名字及计算中以计算对象替换名字

(define height 2) ——— *名字的定义：定义名字**height**与**2**关联，以后可以用**height**来表示**2***

一种类型的名字：数值型的名字

(+ (+ height 40) (- 305 height)) ——— *名字的使用*

(+ (* 50 height) (- 100 height))

*注意：不同类型的对象可以有不同的定义方法。这里统一用**define**来表示，在具体的程序设计语言中是用不同的方法来定义的*

战德臣 教授

命名计算对象和构造中使用名字及计算中以计算对象替换名字

```
(define pi 3.14159)
(define radius 10)
(*  pi  (* radius radius))
```

```
(*  pi  (*  radius radius))
(*  pi  (* 10  10))
(*  pi  100)
(* 3.14159 100)
```
计算

```
(define  circumference  (* 2 p
```

```
(*  circmference  20)
```

```
(*  circmference  20)
(* (* 2  pi  radius) 20)
(* (* 2  3.14159  10) 20)
(* 62.8318  20)
1256.636
```
计算

运算式的组合-抽象与构造

---程序构造示例II-运算符的定义-构造-与计算

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

运算式的组合-抽象与构造---程序构造示例
(4)如何定义、使用和执行新运算符?

战德臣 教授

命名新运算符和构造中使用新运算符及执行中以过程替换新运算符

```
(define (square  x)  (* x x))
```

x^2

名字的定义：定义名字square为一个新的运算，即过程或称函数
另一种类型的名字：运算符型的名字

新运算符，即过程名或函数名

形式参数，使用时将被实际参数所替代

过程体，用于表示新运算符的具体计算规则，其为关于形式参数x的一种计算组合。

```
(square  3)
(square  6)
```

名字的使用

注意：不同类型的对象可以有不同的定义方法。这里统一用define来表示，在具体的程序设计语言中是用不同的方法来定义的

运算式的组合-抽象与构造---程序构造示例
(4)如何定义、使用和执行新运算符?

战德臣 教授

命名新运算符和构造中使用新运算符及执行中以过程替换新运算符

(square 10) ——————— *名字的使用*

(square (+ 2 8))

(square (square 3))

(square (square (+ 2 5)))

(define (SumOfSquare x y) (+ (square x) (square y))

x^2+y^2

(SumOfSquare 3 4)

(+ (SumOfSquare 3 4) height)

运算式的组合-抽象与构造---程序构造示例
(4)如何定义、使用和执行新运算符?

战德臣 教授

命名新运算符和构造中使用新运算符及执行中以过程替换新运算符

(define (NewProc a) (SumOfSquare (+ a 1) (* a 2)))

$(a+1)^2+(a*2)^2$

(NewProc 3)

(NewProc (+ 3 1))

运算式的组合-抽象与构造---程序构造示例
(5)新运算符的计算/执行方法?

战德臣 教授

命名新运算符和构造中使用新运算符及执行中以过程替换新运算符

含名字的运算组合式的计算方法：求值、代入、计算

(NewProc (+ 3 1))的两种计算过程示意

```
(NewProc (+ 3 1))
(NewProc 4)
(SumOfSquare (+ 4 1) (* 4 2))
(SumOfSquare 5 8)
(+ (Square 5) (Square 8))
(+ (* 5 5) (* 8 8))
(+ 25 64)
89
```

先求值，再代入

运算式的组合-抽象与构造---程序构造示例
(5)新运算符的计算/执行方法?

战德臣 教授

命名新运算符和构造中使用新运算符及执行中以过程替换新运算符

含名字的运算组合式的计算方法：代入、求值、计算

(NewProc (+ 3 1))的两种计算过程示意

```
(NewProc (+ 3 1))
(SumOfSquare(+ (+ 3 1) 1) (* (+ 3 1) 2))
(+ (Square (+ (+ 3 1) 1) (Square (* (+ 3 1) 2)))
(+ (* (+ (+ 3 1) 1) (+ (+ 3 1) 1)) (* (* (+ 3 1) 2) (* (+ 3 1) 2)))
(+ (* (+ 4 1) (+ 4 1)) (* (* 4 2) (* 4 2)))
(+ (* 5 5) (* 8 8))
(+ 25 64)
89
```

先代入，后求值

代入阶段

求值阶段

运算式的组合-抽象与构造---程序构造示例
(6)训练一下?

战德臣 教授

◆问题1：用前缀表示法书写下述表达式

$$\frac{10 + 4 + (8- (12 - (6 + 4\div5)))}{3*(6-2)(12-7)}$$

◆问题2：请定义一个过程，求某一数值的立方

a^3

◆问题3：进一步以问题2定义的过程，再定义一个过程，求某两个数值的立方和。a^3+b^3 进一步求 5^3+8^3 ，并模拟给出计算过程。

递归的概念

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

递归的概念
(1)为什么需要递归?

战德臣 教授

递归(Recursion)

(* …(* (* (* (* 1 1) 2) 3) 4) … n)

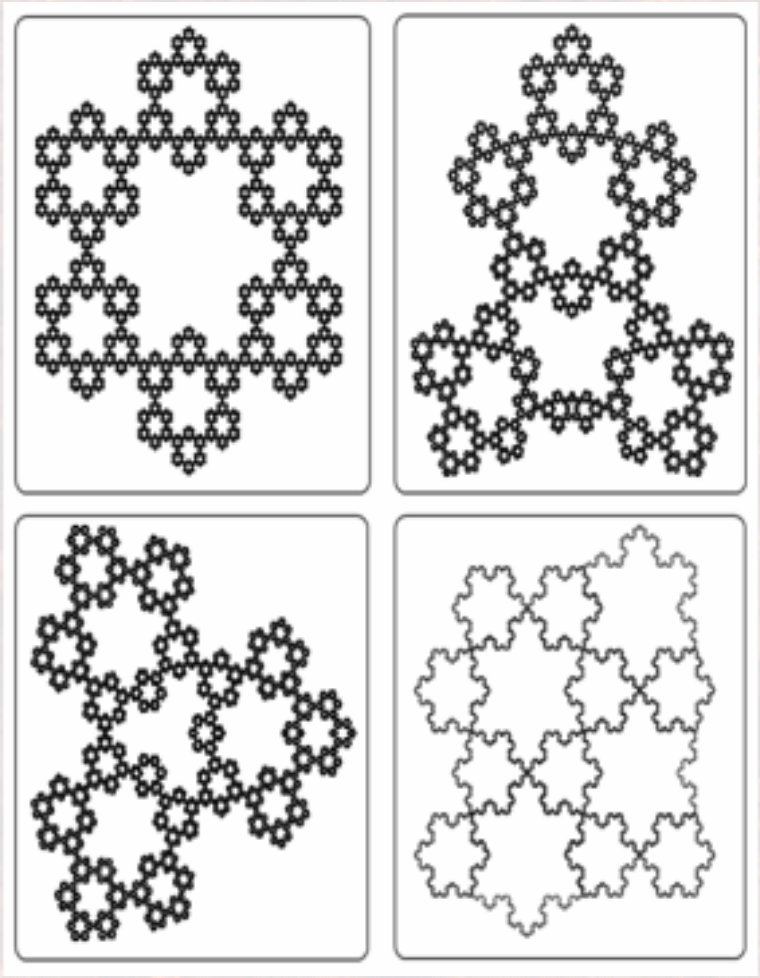
怎样在表达中既去掉省略号，而又能表达近乎无限的内容

“从前有座山，山里有座庙，庙里有个老和尚，正在给小和尚讲故事呢！故事是什么呢？(从前有座山，山里有座庙，庙里有个老和尚，正在给小和尚讲故事呢！故事是什么呢？(从前有座山，山里有座庙，庙里有个老和尚，正在给小和尚讲故事呢！故事是什么呢？(从前……)))”

递归的概念
(2)什么情况需要递归?

战德臣 教授

递归的典型视觉形式---自相似性事物的无限重复性构造



递归的概念
(3)如何表达延续不断却相似或重复的事物或过程?

战德臣 教授

数学中的递推式

◆一个数列的第n项a_n与该数列的其他一项或多项之间存在某种对应关系，被表达为一种公式，称为递推式

等差数列递推公式

$a_0=5$

$a_n=a_{n-1}+3$　当n>=1时

- 第1项(或前K项)的值是已知的—递推基础;
- 由第n项或前n项计算第n+1项—递推规则/递推步骤;
- 由前向后，可依次计算每一项

等差数列的产生

$a_0=5$

$a_1=a_0+3=8$

$a_2=a_1+3=11$

$a_3=a_2+3=14$

$a_4=a_3+3=17$

…… ……

递归的概念
(4)什么是递归?

战德臣 教授

递归是一种表达相似性对象及动作的无限性构造和执行的方法。

■递归基础：定义、构造和计算的起点，直接给出；

■递归步骤：由前n项或第n项定义第n+1项；由低阶f(k)且k<n，来构造高阶f(n+1)

◆递归是一种关于抽象的表达方法---用递归定义无限的相似事物

◆递归是一种算法或程序的构造技术---自身调用自身，高阶调用低阶，构造无限的计算步骤

◆递归是一种典型的计算/执行过程---由后向前代入，直至代入到递归基础，再由递归基础向后计算直至计算出最终结果，即由前向后计算

两种不同的递归函数
---递归与迭代

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

两种不同的递归函数--递归与迭代

(1)两种不同的递归函数?

递归和递推：比较下面两个示例

- Fibonacci数列，无穷数列1，1，2，3，5，8，13，21，34，55，……，称为Fibonacci数列。它可以递归地定义为：

$$F(n)=\begin{cases}1 & n=0\\ 1 & n=1\\ F(n-1)+F(n-2) & n>1\end{cases}$$

递归定义

F(0)=1;

F(1)=1;

F(2)=F(1)+F(0)=2;

F(3)=F(2)+F(1)= 3;

F(4)=F(3)+F(2)= 3+2=5;… …

递推计算/迭代计算/迭代执行

定义是递归的，但执行可以是递归的也可是迭代的

两种不同的递归函数--递归与迭代

(1)两种不同的递归函数?

战德臣 教授

递归和递推：比较下面两个示例

- 阿克曼递归函数---双递归函数
- 阿克曼给出了一个不是原始递归的可计算的全函数。表述如下：

$$A(1,0) = 2$$

$$A(0,m) = 1 \qquad m \geq 0$$

$$A(n,0) = n+2 \qquad n \geq 2$$

$$A(n,m) = A(A(n-1,m),m-1) \quad n,m \geq 1$$

递归定义

函数本身是递归的，
函数的变量也是递归的

m=0时， A(n,0)=n+2；

m=1时， A(n,1)=A(**A(n-1,1),0)=A(n-1,1)+2**， 和A(1,1)=2故A(n,1)=2*n

m=2时， A(n,2)=A(**A(n-1,2),1)=2A(n-1,2)**， 和A(1,2)=A(A(0,2),1)=A(1,1)=2，故A(n,2)= 2^n。

m=3时，类似的可以推出 $\underbrace{2^{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}}_{n}$

递归计算/递归执行

由后向前代入，再由前向后计算

两种不同的递归函数--递归与迭代

(1)两种不同的递归函数?

战德臣 教授

递归和递推：比较下面两个示例

- 阿克曼递归函数---双递归函数---另一种形式

$$A(m,n)=\begin{cases} n+1 & \text{若} m=0 \\ A((m-1),1) & \text{若} n=0 \\ A(m-1,A(m,n-1)) & \text{若} m,n>0 \end{cases}$$

A(1,2) = A(0,A(1,1)) = A(0, A(0,A(1,0))) = A(0, A(0,A(0,1)))=A(0,A(0,2))=A(0,3)=4。
A(1,3) =A(0, A(1,2))=A(0. <...代入前式计算过程>)=A(0,4)=4+1=5。
...
A(1,n) =A(0, A(1,n-1))=A(0. <...代入前式计算过程>)=A(0,n+1)=n+2。
A(2,1) = A(1，A(2，0)) = A(1，A(1，1)) =A(1，A(0，A(1，0)))
= A(1，A(0，A(0，1))) = A(1，A(0，2)) = A(1，3) =A(0，A(1，2))
= A(0，A(0，A(1，1))) = A(0，A(0，A(0，A(1，0))))
= A(0，A(0，A(0，A(0,1))) = A(0，A(0，A(0，2))) = A(0，A(0，3))
= A(0，4) = 5。

两种不同的递归函数--递归与迭代
(2)递归和迭代有什么差别?

递归和迭代(递推)

◆迭代(递推)：可以自递归基础开始，由前向后依次计算或直接计算；

◆递归：可以自递归基础开始，由前向后依次计算或直接计算；但有些，只能由后向前代入，直到递归基础，寻找一条路径，然后再由前向后计算。

◆递归包含了递推(迭代)，但递推(迭代)不能覆盖递归。

运用递归与迭代

战德臣

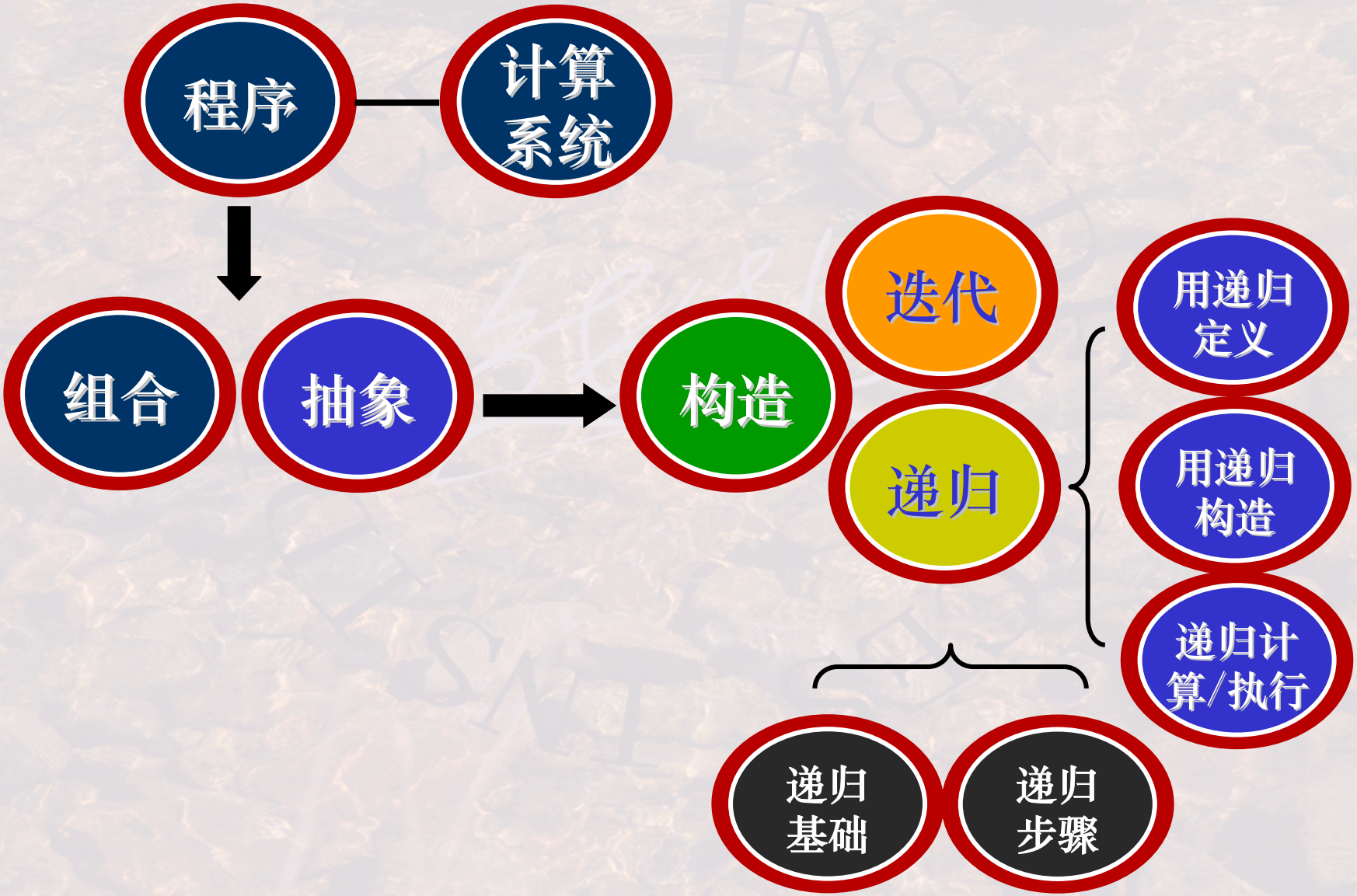
哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

运用递归和迭代
(0)概述

战德臣 教授



运用递归与迭代
—无限的自相似性对象的定义

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

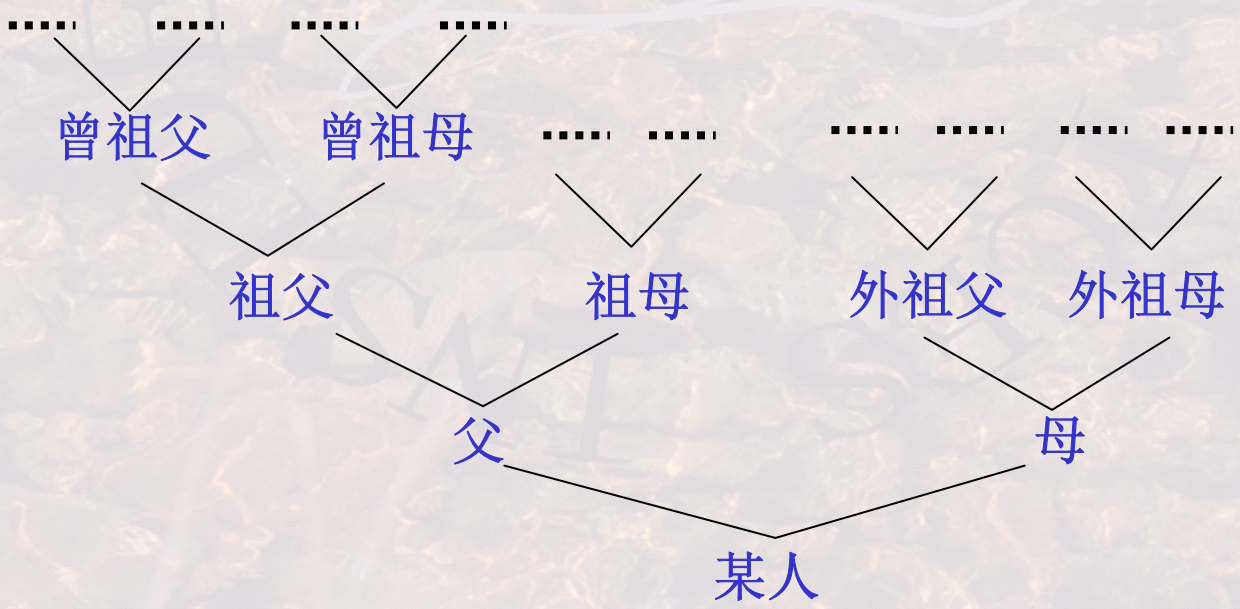
Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

战德臣 教授

示例：“某人祖先”的递归定义

(1)某人的双亲是他的祖先（递归基础）。

(2)某人**祖先**的双亲同样是某人的**祖先**（递归步骤）。



战德臣 教授

示例：算术表达式的递归定义

首先给出递归基础的定义：

(1)任何一个常数C是一个算术表达式；

(2)任何一个变量V是一个算术表达式；

再给出递归步骤：

(3)如F、G是算术表达式，则下列运算：

F+G，F-G，F*G，F/G是算术表达式；

(4)如F是表达式，则(F)亦是算术表达式。

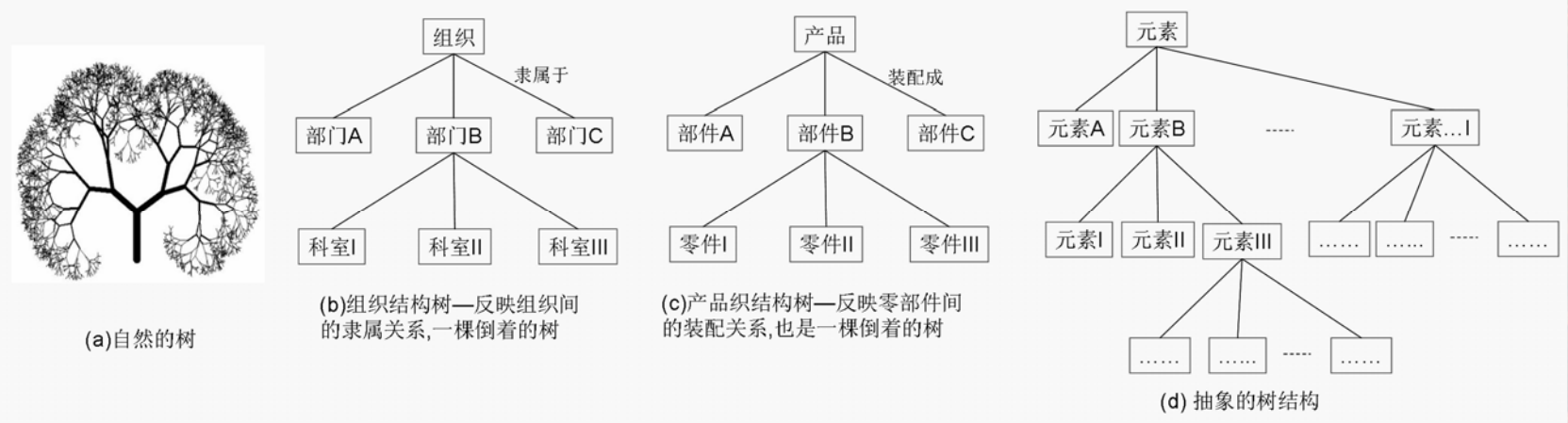
(5)括号内表达式优先计算，“*”与“/”运算优先于“+”与“-”运算。

(6)算术表达式仅限于以上形式。

(… (((100 + (X + Y)) * (Z-Y)) + Z) …)

战德臣 教授

示例：树的形式化递归定义



(a)自然的树

(b)组织结构树—反映组织间的隶属关系,一棵倒着的树

(c)产品织结构树—反映零部件间的装配关系,也是一棵倒着的树

(d) 抽象的树结构

树是包含若干个元素的有穷集合，每个元素称为结点。其中：

(1)有且仅有一个特定的称为根的结点；**(**递归基础**)**

(2)除根结点外的其余结点可被分为**k**个互不相交的集合$T_1, T_2, \cdots, T_k (k \geq 0)$，其中每一个集合$T_i$本身也是一棵树，被称其为根的子树。**(**递归步骤**)**

运用递归与迭代
—递归算法与程序的构造

战德臣

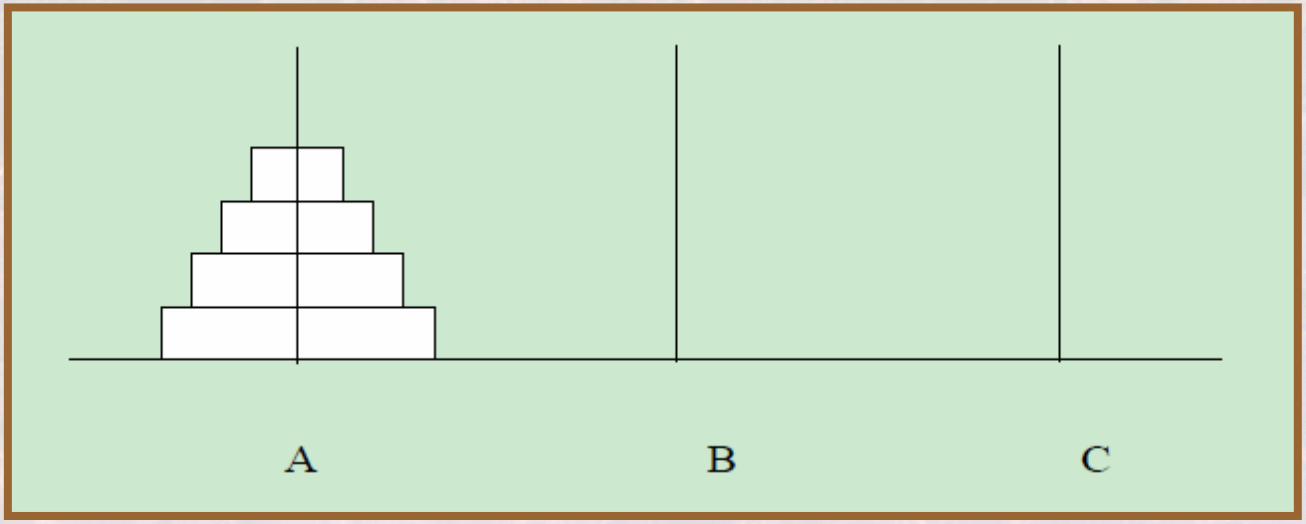
哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

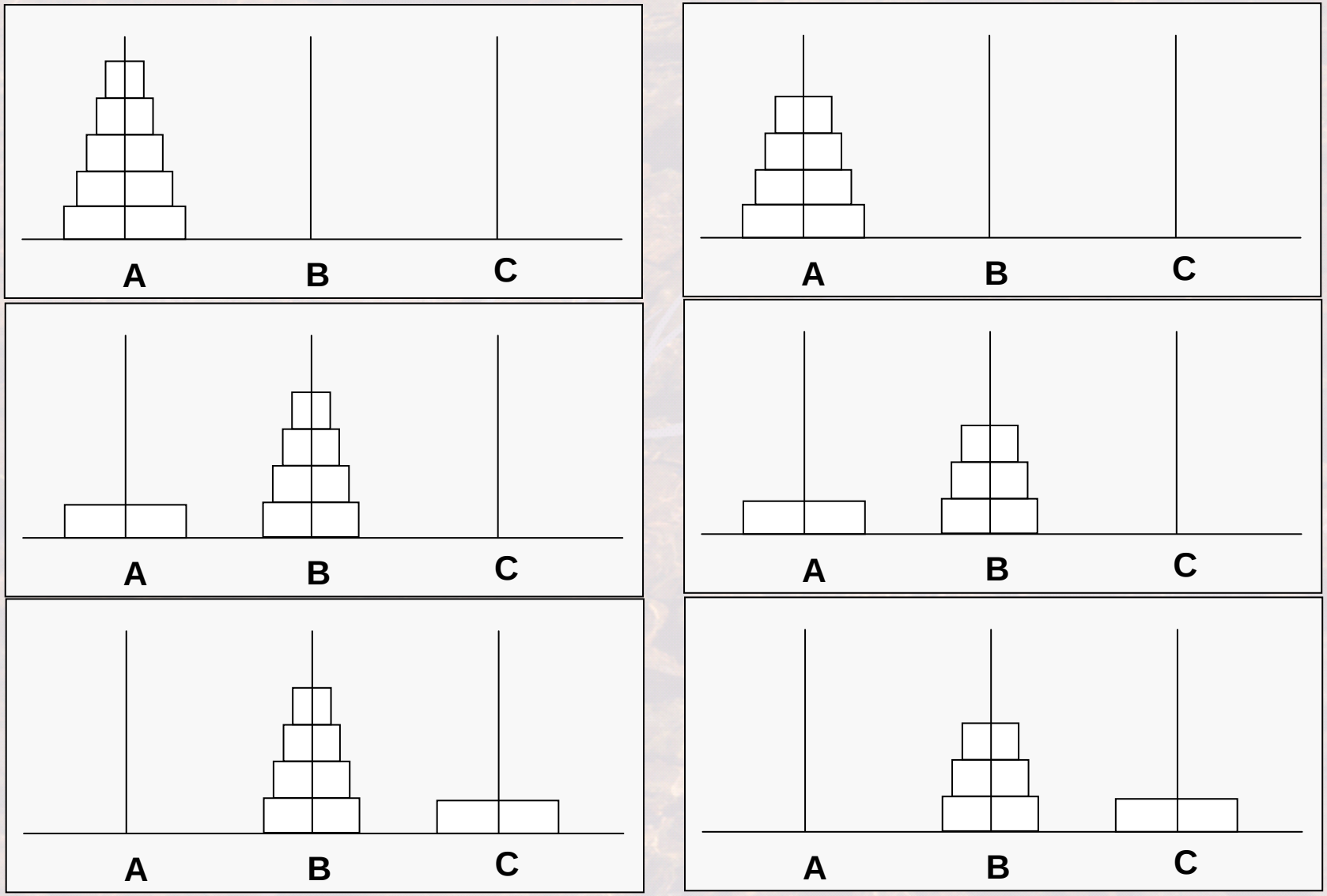
战德臣 教授

梵天塔(汉诺塔)问题：有三根柱子，梵天将64个直径大小不一的金盘子按照从大到小的顺序依次套放在第一根柱子上形成一座金塔，要求每次只能移动一个盘子，盘子只能在三根柱子上来回移动不能放在他处，在移动过程中三根柱子上的盘子必须始终保持大盘在下小盘在上。



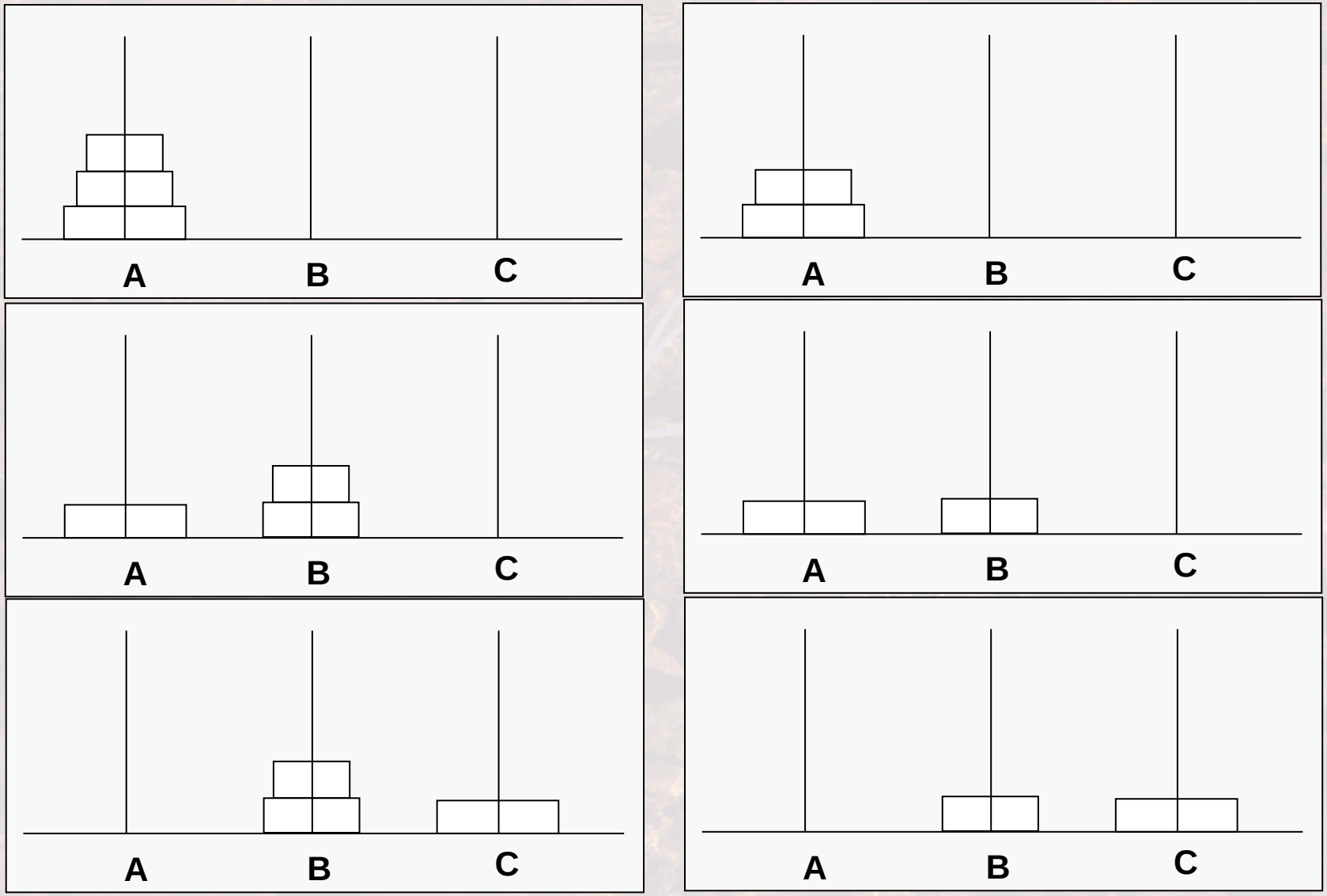
运用递归和迭代
(2)递归算法与程序的构造?

战德臣 教授



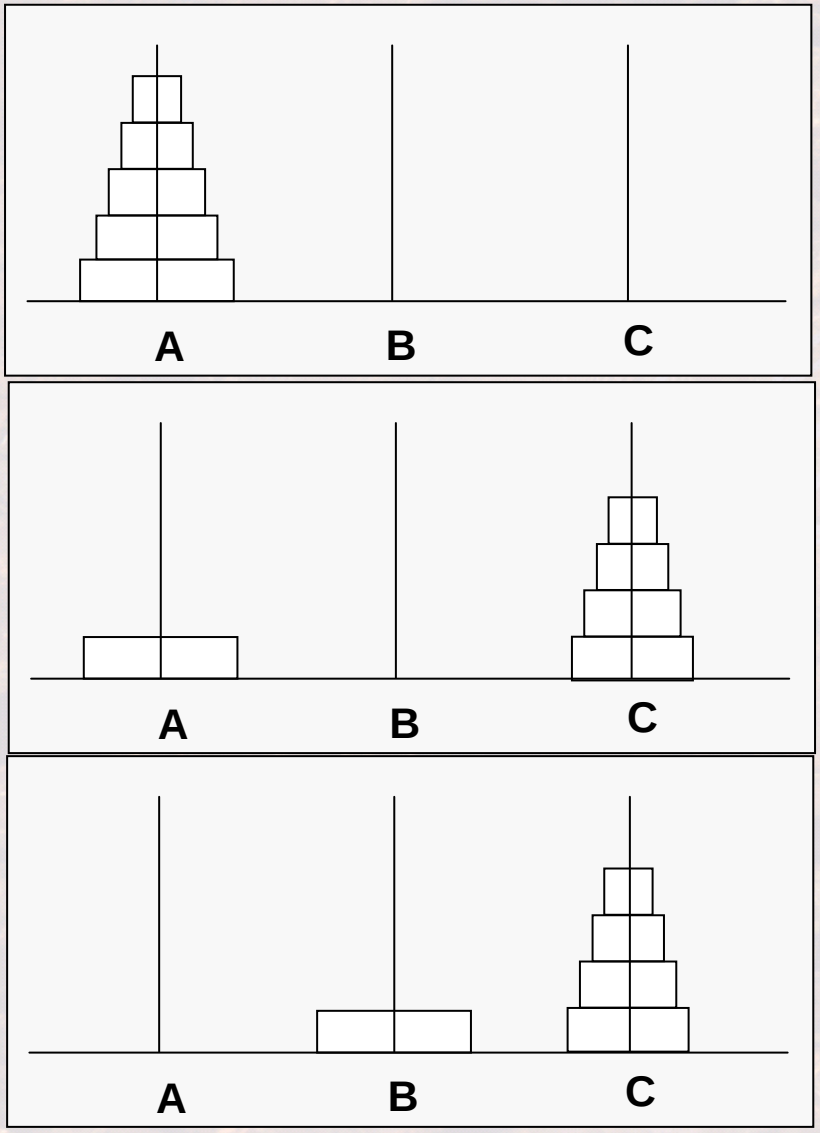
运用递归和迭代
(2)递归算法与程序的构造?

战德臣 教授



运用递归和迭代
(2)递归算法与程序的构造?

战德臣 教授




```
Main()
{  //假设有5个盘子的汉诺塔
     Hanoi(5, “A”, “B”, “C”);
}

int Hanoi (int N, int X, int Y, int Z)
{  //该函数是将N个盘子从X柱，以Z柱做中转，移动到Y柱上
   If  N > 1 Then
   {
    //先把n-1个盘子从X放到Z上(以Y做中转)
        Hanoi (N - 1, X, Z, Y);
    //然后把X上最下面的盘子放到Y上
        Printf(“%d→%d”, X, Y);
    //接着把n-1个盘子从Z上放到Y上(以X做中转)
        Hanoi (N - 1, Z, Y, X);
   }
   Else
   { //只有一个盘子时，直接把它从X放到Y上
        Printf(“%d→%d”, X, Y);
   }
}
```

运用递归与迭代
—递归与迭代程序的执行

战德臣

哈尔滨工业大学　教授.博士生导师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

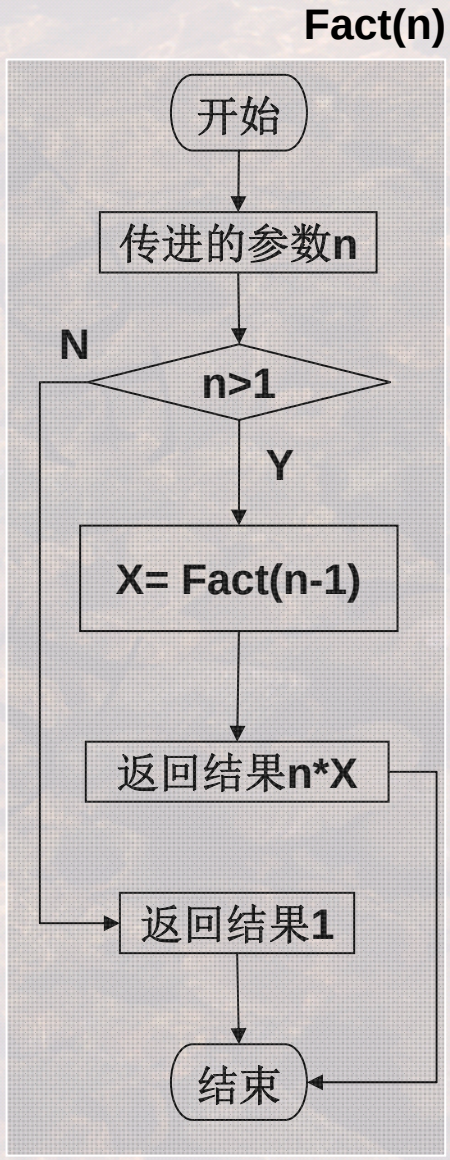
战德臣 教授

具有无限的自相似性步骤的表达，自身调用自身，高阶调用递阶

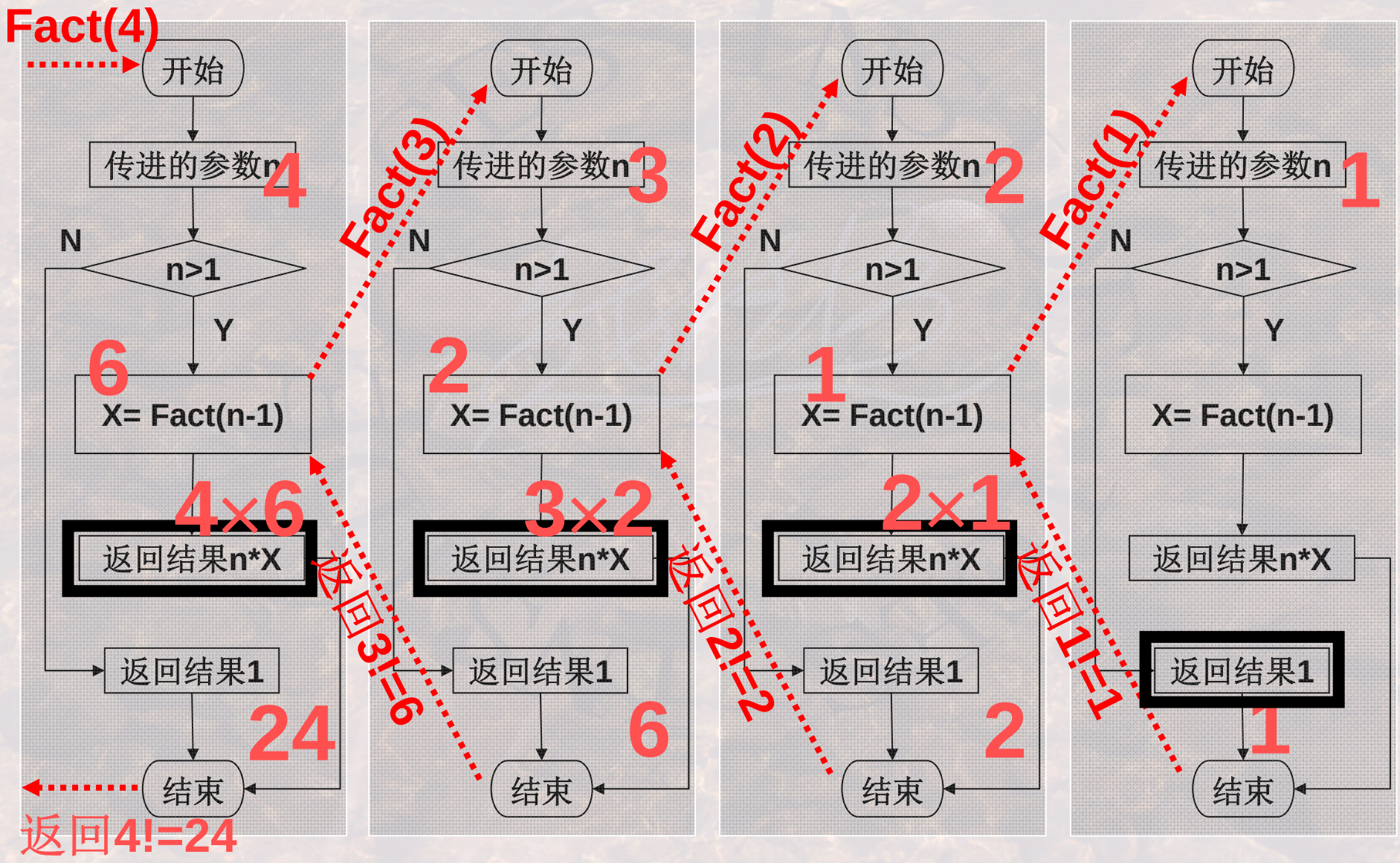
示例：求**n!**的算法或程序 *--用递归方法构造*

$$n!=\begin{cases}1 & \text{当} n\le 1 \text{时}\\ n\times(n-1)! & \text{当} n>1 \text{时}\end{cases}$$

```
long int Fact(int n)
{    long int x;
     If (n > 1)
     { x = Fact(n-1);
  /*递归调用*/
        return n*x;   }
     else return 1;
  /*递归基础*/
}
```



运用递归和迭代
(3)递归与迭代程序的执行



运用递归和迭代
(3)递归与迭代程序的执行

战德臣 教授

具有无限的自相似性步骤的表达，循环-替代-递推—迭代

示例：求n!的算法或程序 *--用迭代方法构造*

$$n!=\begin{cases}1 & \text{当} n\le 1\text{时}\\ 1\times 2\times\ldots(n-1)\times n & \text{当} n>1\text{时}\end{cases}$$

```
long int Fact(int n)
{   int  counter;
    long product=1;
    for  counter=1 to n step 1
    { product = product * counter;  }
  /*迭代*/
return product;
}
```

Counter	Product
	1
1	1
2	2
3	6
4	24
5	120
6	720

运用递归与迭代
(4)回顾差分机器的迭代执行

战德臣 教授

```
/*类C语言表达的计算规则—程序
Main()
{
  int k, n, square_nminus1,square_n, alpha_nminus1, alpha_n, beta_n;
  input k;
  square_nminus1=1; square_n=4; alpha_nminus1=1;
  for n=2 to k-1
  {
     alpha_n = square_n – square_nminus1;
     beta_n = alpha_n – alpha_nminus1;
     square_nplus1 = square_n + alpha_n + beta_n;
     square_nminus1 = square_n;
     square_n = square_nplus1;
     alpha_nminus1 = alpha_n;
  }
  output square_n;
}
```

n	square_nplus1	square_n	square_nminus1	alpha_n	alpha_nminus1	beta_n
2	9	4	1	3	1	2
3	16	9	4	5	3	2
4	25	16	9	7	5	2
5	36	25	16	9	7	2
		36	25		9	

运用递归和迭代
(5)小结