


何中市  
重庆大学计算机学院



# Algorithm Design & Analysis

## Introduction to Algorithm



# 《算法分析与设计》课程介绍

- 教材 《Introduction to Algorithms》
- 主讲内容
  - 1-2-3-4
  - 7-15-16-26-32
- 学习目标：掌握经典算法的基本原理，懂得如何寻求解决问题的方法
- 学习方法：多读、多练习、勤思考、勤动手

# 《算法分析与设计》课程介绍

- 教师：缩小反差
  - 教材课件-学生实战，自然社会-人，理想-现实
  - 代读经历-提高效率，课件-思想语言生活阅历
- 算法：学会计算
  - 3W{What&Why-教材、Where-现代计算&算法帝国}
  - 计算？ 计算机学院？ Y&M
- 理念：提炼三点
  - 掌握核心技术-理解基本概念-聚焦看点热点
- 要求：算法素质&计算能力
  - 分析技术、设计策略、描述方法
  - 听课笔记、作业练习、课堂测试
  - 分享教材PPT，预习带书带笔纸

# 知识体系@ AAD 课程

- 1、算法=A
- 2、算法分析=AA
- 3、算法设计=AD

# 知识体系@ AAD 课程

- 1、算法=A
  - 定义
  - 作用
  - 描述
- 2、算法分析=AA
  - 复杂度
  - 作用
  - 复杂度计算
- 3、算法设计=AD
  - 问题开始
  - 策略设计
  - 问题求解

# 知识体系@ AAD 课程-A

- 1、算法=A

- 定义

- 定义=求解问题的操作序列
    - 性质：功能性、确定性、正确性、简单性等**十大性质？**
    - 准则：正确性、工作量、简单性、最优性      **复杂性？**

- 作用

- 计算：现代计算&算法帝国
    - 计算+：

- 描述

- 问题描述两部分=输入/输出(实例)
    - 算法描述三步骤(初始-关键-重复)
    - 逐步求精三层次(思想-关键-伪代码)

# 知识体系@ AAD 课程-AA

- 2、算法分析=AA

- 复杂度

- 模型-简化：不计基本操作之快慢，不比机器性能之高低
    - 时间复杂度：  $T(n)$ =基本操作的次数
    - 空间复杂度：  $S(n)$ =基本变量之个数
    - 三者关系：  $t=T(n)/v$   $v$ --CPU主频 -1/s

- 作用

- 复杂度计算 CC

- 直接合计：主要的 基本操作的 次数
    - 递归方程求解：分治递归算法
    - 渐近性分析：简化比较

# 知识体系@ AAD 课程-AA-CC

- 2、算法分析=AA

- 复杂度计算：CC

- 直接合计：主要的 基本操作的 次数

- 典型问题：给定算法伪代码(递归、迭代在其中)，计算 $T(n)$ ；
      - 插入排序算法-复杂度

- 递归方程求解：分治递归算法

- 两类方程：等差递减、等比递减
      - 四种方法：直接展开TD(Expansion),代入法BU(Substitution),递归树(Recursion),主方法(MasterMethod): TopDown-BottomUp-图形直观-公式
      - 典型方程：  $T(n) = T(n-1) + O(n)$ ,  $T(1) = O(1)$
      - $T(n) = k T(n/2) + \Theta(n)$ ,  $T(1) = \Theta(1)$ ; 特殊  $T(n) = 2T(n/2) + n$ ,  $T(1) = 1$
      - $T(n) = T(n/4) + T(n/2) + \Theta(n^2)$ ,  $T(1) = \Theta(1)$ ;

- 渐近性分析：简化比较



# 知识体系@ AAD 课程-AA-CC

- 2、AA=算法分析
  - 复杂度计算：CC
    - 渐近性分析：简化比较
      - 理念：路遥知增速
      - 五个符号： $\Theta O \Omega - o \omega$ 
        - » 定义：
        - » 关系/性质&证明：
      - 三类代表函数：常数1、多项式 $n^a$ 、指数 $a^n$ .
      - 常见复杂度函数： $\log n, n, n \log n, n^2 \dots, 2^n \dots$  关联典型问题

# 知识体系@ AAD 课程-AD

- 3、AD=算法设计

- 问题开始

- 搜索问题：分治+递归
    - 排序问题：分治+递归
    - 优化问题

- 策略设计

- DC 分治+递归：（Ch4+Ch7）
    - DP 动规+递归：（Ch15）
    - GA 贪心+递归：（Ch16）
    - 动规&贪心：最大流问题（CH26）

# 知识体系@ AAD 课程-AD

- 3、AD=算法设计

- 问题开始

- 搜索问题：分治+递归

- 无序：排序后搜索，顺序查找

- 排序问题：分治+递归

- 问题描述：输入-输出（实例）

- 排序算法：插入、选择、冒泡、归并、快速,...

- 优化问题

- 动态规划：钢条切割问题、整数背包、矩阵链乘积、最长公共子序列、最优二叉搜索树 ...

- 贪心算法：活动选择、分数背包、哈夫曼编码（Huffman Codes） ...

- 动态规划&贪心：最大流(单源单汇、多源多汇)、可转化为最大流问题...

- 字符串匹配 Ch32?

- 策略设计

# 知识体系@ AAD 课程-AD

- 3、AD=算法设计

- 问题开始

- 策略设计

- 分治+递归：（Ch4+Ch7）

- 思想：求解就是简化、分解以求分治

- 适用条件：大可分小可治，递归如法炮制

- 典型问题：排序-快速排序&改进版@中位数、归并排序

- 动规+递归：（Ch15）

- 思想：问题优化-错综复杂-函数递归-自顶向下：自底向上(最优值)结果共享

- 条件：最优子结构

- 保证：反证法

- 关键：递归方程建立=选择何处开刀&问题关系

# 知识体系@ AAD 课程-AD

- 3、AD=算法设计

- 问题开始

- 策略设计

- 分治+递归：（Ch4+Ch7）

- 动规+递归：（Ch15）

- 贪心+递归：（Ch16）

- 思想：问题优化-错综复杂-简单操作-贪心选择：局部最优(达到)全局最优

- 条件：最优子结构+贪心选择性质

- 保证：反证法、构造法

- 关键：贪心指标选择=选择合适指标&指标取向

- 动规&贪心：最大流问题（CH26）

- 思想：先动规-从小到大逐步增加，兼贪心-每步贪心尽量最大

- FF算法：逐步增加@构造s-t增广路径P，贪心选择@增加P流量

- EK算法：逐步增加@构造s-t最短增广路径P，贪心选择@增加P流量

- 概念符号：残存网络、增广路径、割，

- 性质结论：关系、三等价定理（最大流-最小割-增广路）证明

# 算法描述：关键&难点

## ✓ 关键

算法：计算思维之方法

思维.表达...相辅相成...相得益彰...

## ✓ 难点

思维容易表达难，能说会道下笔难。

浮想联翩，思绪万千，交流不易，下笔困难 坚持练！

## ✓ 方法

程序语言、机器语言、自然语言、伪代码。

符号模块流程图，自然程序伪代码！

易于理解很重要，逻辑正确是王道。

# 算法描述-实例：粗细结合三个版本+三个

## ✓ 插入排序

见纸质笔记-手写黑板/电子版@20200219

问题描述：排序问题

算法描述：插入排序

算法-1版：方法描述（关键环节：找位置、插入）

算法-2版：步骤描述（步骤、符号：粗线条三步走）

算法-3版：伪代码（融合 步骤、符号、文字、代码）

算法-4版：伪代码（完整版：输入/输出/必要说明）

算法-5版：伪代码（优化版：合并找位置与插入）

算法-6版：伪代码（更优化版：合并、不增加新序列）

# 算法描述：实例

**算法/伪代码：**插入排序-4+版（完整版：输入/输出/必要的说明文字，易于理解）

输入：一个序列  $n$  个元素  $a_1, a_2, \dots, a_n$

输出：升序排列  $b_1, b_2, \dots, b_n$ , 满足  $b_1 \leq b_2 \leq \dots \leq b_n$

步 1 赋初始：  $b_1 \leftarrow a_1$

//迭代：逐个插入有序区  $b$ , 依次取出  $a$ , 从  $a_2 \dots a_n$

For  $j=2 \dots n$

步 2 找位置： //设  $k = a_j$  插入  $b_1 \dots b_{j-1}$  的适当位置...逐一对比中立位置

$i = j - 1$

While  $i > 0 \ \& \ b_i > a_j$

    Do  $i \leftarrow i - 1$

$k \leftarrow i$

步 3 插入： //将  $a_j$  插入到  $b_k$  与  $b_{k+1}$  之间 ... $b_k$  之后元素逐一后移，空位插入

For  $i = j - 1 : k + 1 : -1$

$b_{i+1} \leftarrow b_i$

$b_{k+1} \leftarrow a_j$



# 算法描述：实例

表 4.1 社交相似度算法伪代码

---

**算法：**社交相似度算法

---

**input:** 社交网络中的用户数据  $U$ ，用户-用户的关系数据  $R$

**output:** 用户社交相似度  $socialSim$

1.   **for every**  $U_a, U_b \in U$  **do**   //遍历每个用户
  2.       **for every**  $R_{ab} \in R$  **do**   //遍历每个用户的好友关系数据
  3.           **set**  $follower(a), follower(b)$
  4.            $socailSim_{ab} = \frac{|follower(a) \cap follower(b)|}{\sqrt{|follower(a)| |follower(b)|}}$    //计算社交相似度
  5.       **end for**
  6.   **end for**
-

# 排序问题与排序算法

排序：为什么？

排序算法：知多少？

几种基于比较的排序算法思想&复杂度 $T(n)$

归并排序： $T(n)=O(n\log n)$

插入排序： $T(n)=O(n^2)$

选择排序： $T(n)=O(n^2)$

快速排序： $T(n)=O(n^2), O(n\log n)?$

堆排序： $T(n)=O(n^2), O(n\log n)?$

排序算法： $T(n)$ 的下界？ 渐近最优？  $O(n\log n)?$

其他排序算法：计数、基数、桶排序...  $O(n)?$

顺序统计量： $O(n^2), O(n)?$

# 简单实现 @快速排序算法: QUICKSORT

- **QUICKSORT** (A, p, r)  
    **IF**  $p < r$   
    **THEN**  $q \leftarrow$  **PARTITION** (A, p, r)  
        **QUICKSORT** (A, p, q-1)  
        **QUICKSORT** (A, q+1, r)  
     $n=r-p+1, k=q-p,$   
     $\Theta(1)$   
     $\Theta(n)$   
     $T_Q(q-p)$   
     $T_Q(r-q)$
- Initial call: **QUICKSORT**(A, 1, n)
- 写出算法的时间复杂度  $T_Q(n)$  的递归方程
- $T_Q(n) = \Theta(1) + \Theta(n) + T_Q(k) + T_Q(n-k-1)$   
     $= \Theta(n) + T_Q(k) + T_Q(n-k-1) = 2T(n/2) + \Theta(n)$   
     $k = ? \quad n/2, 1$

# 改进@快速排序算法

- 1、如何改进快速排序算法？使得  $T(n)=O(n\log n)$ 
  - 三步：选主元：  $x \leftarrow A[1,n]$  的中位数  $\Theta(n)?$  设计此算法
  - 划分：  $\text{左} \leq x \leq \text{右边}$   $\Theta(n)$
  - 递归调用：  $T(n/2)+T(n/2)=2T(n/2)$
  - $T(n)=2T(n/2)+\Theta(n) = \Theta(n\log n)$
- 2、如果：找中位数算法的时间复杂度为  $\Theta(n \log n)$ 
  - 那么：  $T(n)=2T(n/2)+\Theta(n\log n) = \Theta(?)$  求解此方程
- 1. 如何改进快速排序算法？使得最坏时间复杂度  $T(n)=O(n\log n)$
- 2. 如果主元采用中位数，假设找中位数的算法的时间复杂度为  $O(n\log n)$ , 那么快速排序算法的时间复杂度  $T(n)$  是多少？

# 三种算法设计策略对比：DP-GA-DC

- 最优化问题
- DP：一组选择、达到最优（子问题、相同问题-重复求解）
- DP-GA-DC共性：求解{组合子问题之解、构建原问题之解}
- DP三步骤：
  - 1、建立递归方程：刻画最优结构并建立递归方程@最优解/值
  - 2、**计算最优值**：自底向上（避免重复，计划-规划-动态规划）
  - 3、构造最优解：利用过程信息（更新最优值 & 更新最优方案）
- GA两步骤：
  - 1、确定贪心指标：确定贪心指标与贪心取向
  - 2、**顺序选择对象**：按照贪心指标取向排序、并顺序选择合适对象
- MF最大流-两步走&循环
  - 1、**寻找可增长路径**
  - 2、**沿路径增加流**

# 最大流算法：步骤

---

initialize network with null flow;

## **Method FindFlow**

if augmenting paths exist then

    find augmenting path;

    increase flow;

    recursive call to FindFlow;

# AAD 重点@过去考试

- 1、算法复杂度&渐进分析
- 2、动态规划
- 3、贪心算法
- 4、最大流

# AAD 重点@过去考试

## • 1、算法复杂度&渐进分析

- 1) 给出一段代码，算出复杂度
- 2) 递归方程求解：递归树、归纳法（替代法）证明、代入法：  
 $T(n) = 3T(n/4) + n^2$
- 3) 算法复杂度渐近分析
  - 证明：  $O(f(n)) + O(g(n)) = O(\min\{f(n) + g(n)\})$ ,
  - $\Theta(f(n)) + \Theta(g(n)) = \Theta(f(n) + g(n))$
  - 判断：渐近性递增：  $2^{(n/\log n)}$ ,  $n^2 - n^{1.5}$ ,  $2n^{1.9}$ ,  $\log(\log n + n^{1.9})$ ,  $n^{(n/\log n)}$
- 4) 合并排序：充分解读，不同情况的 $T(n)$ 的递归方程及结果（联系三个符号）

## • 2、动态规划

- 1) 问题：0-1 背包问题、矩阵列相乘问题、整数相加合并问题...
- 2) 写递归方程、写算法伪代码、分析算法复杂性



# AAD 重点@过去考试

- 1、算法复杂度&渐进分析
- 2、动态规划
- 3、贪心算法
  - 1) 问题：特殊0-1 背包问题@重量价值有特殊关系
  - 2) 证明：满足贪心选择性质（局部最优选择满足贪心原则）
- 4、最大流
  - 1) 证明性质： $f(V, t) = |f|$ ,  $f(V, V)=0$ .
  - 2) 给出具体流网络，画出残存网络、找出最短增广路径（广度优先遍历）。