算法设计与分析

第一章 绪论

户保田
e-mail:hubaotian@hit.edu.cn

哈尔滨工业大学（深圳）计算机学院

课程信息

- 授课教师: 户保田 博士，副教授
- 办公室: 信息楼1518
- 邮箱: hubaotian@hit.edu.cn
- 助教：孙泽田、蒋深远
- 课程QQ群：扫码加入，以班级-姓名（如，计算机0班-户保田）格式实名

课程信息

课程教材： 《算法导论》第三版，殷建平等译，机械工业出版社，2012. 12.

学习资源： https://leetcode-cn.com/problemset/all/

课程考核方法

考核环节	所占分值	考核与评价细则
平时作业及考核	40	学生们上课考勤，对提问的回答等，占5分；课程测试与作业，每次单独评分，最后合计成绩占35分。
期末考试	60	卷面成绩100分，以卷面成绩按比例折算成实际得分；考试命题以大纲中的应知应会内容为主，并保证逐年有所变化。

本讲内容

1.1 为什么学习算法

1.2 什么是算法

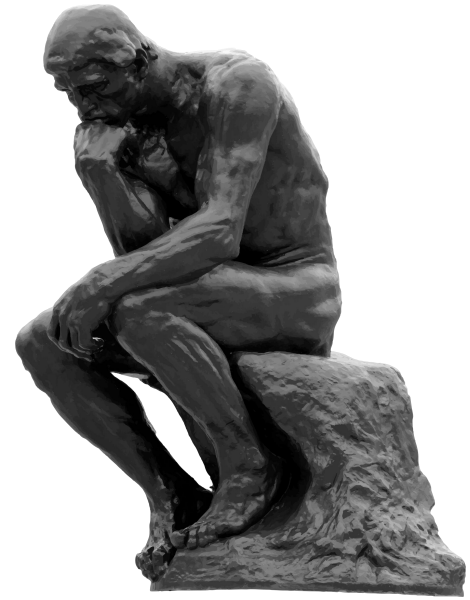
1.3 计算机科学中算法的位置

1.4 算法分析引论

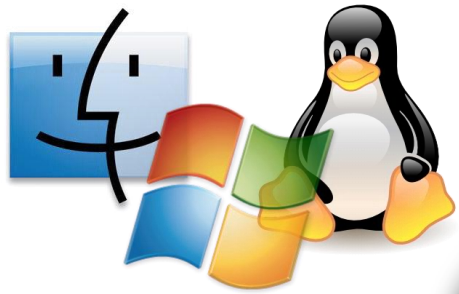
1.5 算法设计引论

为什么学习算法

- 它是计算机专业的核心必修专业课
  - 算法是计算机学科的基础
  - 算法是有用的
  - 算法是有趣的

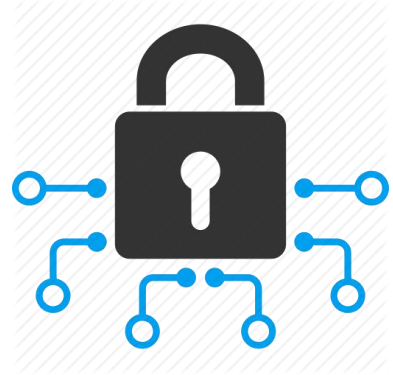


算法是基础

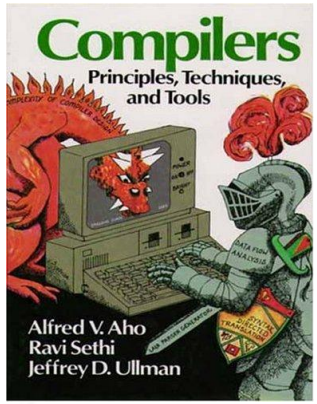


Operating Systems

Machine learning



Cryptography



Compilers

Networking

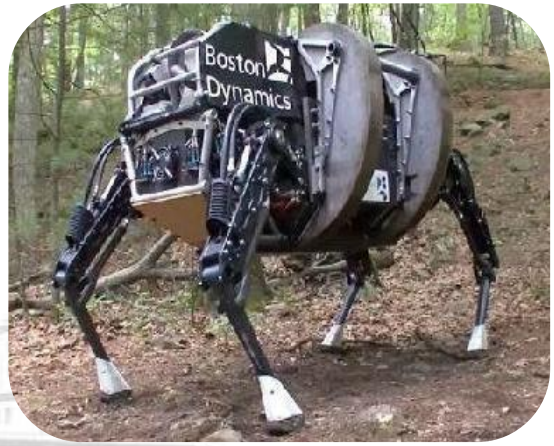
Computational Biology

算法很有用

- 当我们能够获得的数据越来越大，算法就越来越重要
- 学好算法能帮你找到自己喜欢的工作

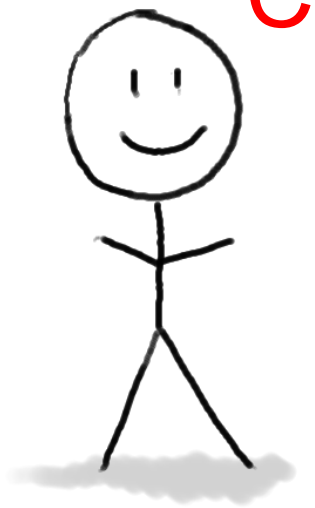
算法很有趣

- 算法设计即是艺术也是科学
- 会让你能够解决一些很酷的问题
- 一个不断发展的年轻领域，很多重要的问题值得探索



算法设计者的问题

Can I do better?



Algorithm designer

学者谈算法

算法是计算机科学领域最重要的基石之一。许多学生认为学计算机就是学各种编程语言，或者认为，学习最新的语言、技术、标准就是最好的铺路方法。编程语言虽然该学，但是学习计算机算法和理论更重要，因为计算机语言和开发平台日新月异，但万变不离其宗的是那些算法和理论。整天赶时髦的人最后只懂得招式，没有功力，是不可能成为高手的。——李开复

（https://cloud.tencent.com/developer/article/1055096）

李航，字节跳动研究负责人、ACL Fellow、IEEE Fellow、ACM 杰出科学家。

课程目标

- 具有扎实的算法设计与分析能力
- 具有良好的算法思维

Learn to think “algorithmically”

本讲内容

1.1 为什么学习算法

1.2 什么是算法

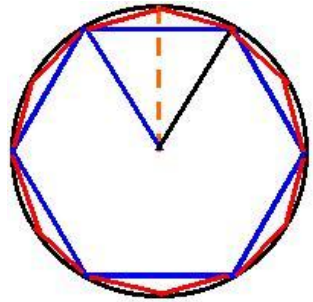
1.3 计算机科学中算法的位置

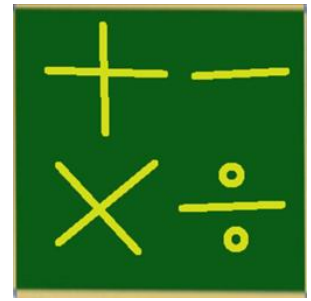
1.4 算法分析引论

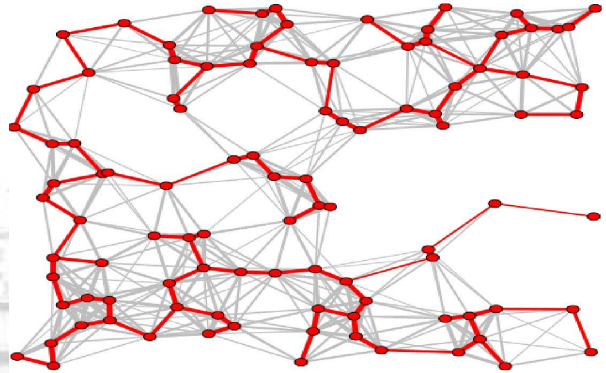
1.5 算法设计引论

什么是算法?

- **在数学和计算机科学之中，算法/算则法（Algorithm）为一个计算的具体步骤，常用于计算、数据处理和自动推理。**
- **算法的例子**
  - 刘徽割圆术
  - 四则运算
  - 最小生成树
  - 快速排序







计算的定义

- 可由一个给定**计算模型**机械地执行的**规则或计算步骤序列**称为该计算模型的一个**计算**
  - 一个计算机程序是一个计算
  - 计算可能永远不停止——不是算法

算法的定义

- 算法是一个满足下列条件的**计算**：
  - **有穷性/终止性**：有限步内必须停止
  - **确定性**：每一步都是严格定义和确定的动作
  - **能行性**：每一个动作都能够被精确地机械执行
  - **输入**：有一个满足给定约束条件的输入
  - **输出**：满足给定约束条件的结果

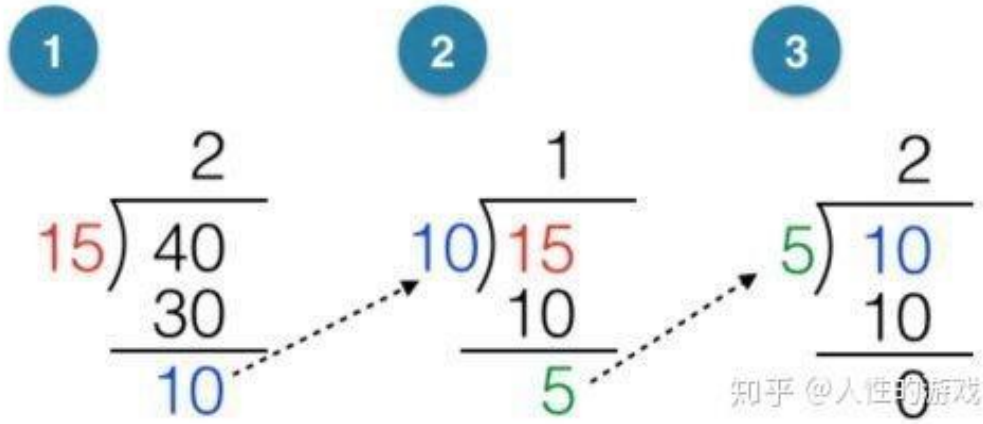
关于算法

- **“算法”的来源**
  - 中文名称：周髀（bì）算经
  - 英文名称：“Algorithm”来自于9世纪波斯数学家花拉子米（al-Khwarizmi），“算法”原为“algorism”，即“al-Khwarizmi”的音转，意思是“花拉子米”的运算法则。在18世纪演变为“algorithm”
- **最早的算法：**欧几里德的“求最大公因子算法”

关于算法

//注意：这里不用考虑m和n的大小问题。

```
int euclid(int m, int n)
{
    int r;
    do
    {
        r = m % n;
        m = n;
        n = r;
    }
    while(r!=0);
    return m;
}
```



欧几里得（约公元前330年—公元前275年），古希腊数学家。

问题的定义

- **算法的目的是求解问题，什么是问题?**

  设Input和Output是两个集合。一个问题是一个关系$R \subseteq Input \times Output$，Input称为问题R的输入集合，Input的每个元素称为R的一个输入，Output称为问题R的输出或结果集合，Output的每个元素称为R的一个结果。

- **问题定义了输入和输出的关系**

问题的例子

- **SORT问题定义如下：**
  - 输入集合Input={$<a_1,\ldots,a_n>$ | a_i是整数}
  - 输出集合Output={$<b_1,\ldots,b_n>$ | b_i是整数, $b_1 \le \ldots \le b_n$}
  - SORT={($<a_1,\ldots,a_n>$, $<b_1,\ldots,b_n>$) | $<a_1,\ldots,a_n> \in$ Input, $<b_1,\ldots,b_n> \in$ Output, $\{a_1, \ldots, a_n\} = \{b_1, \ldots, b_n\}$}
- **问题实例**
  - 问题P的一个实例是P的一个二元组
  - 注意：一个算法面向一个问题，而不是仅求解一个问题的一个或几个实例

算法演示

A[1,……,n] = 5,2,4,6,1,3

A[1,……,n] = 5,2,4,6,1,3

A[1,……,n] = 2,5,4,6,1,3

A[1,……,n] = 2,4,5,6,1,3

A[1,……,n] = 2,4,5,6,1,3

A[1,……,n] = 1,2,4,5,6,3

A[1,……,n] = 1,2,3,4,5,6

算法描述

Insertion-sort

Input: **A[1,.....,n]，n个数的数组**

Output: **A[1,.....,n]，n个排好序的数的数组**

Insertion-sort(A)

过程：Insertion-sort(A)

```
1. FOR j=2  To n  Do
2.         key←A[j];
3.         i←j-1
4.         WHILE i>0 AND A[i]>key  Do
5.                   A[i+1]←A[i];
6.                   i←i-1;
7.         A[i+1]←key;
8. Return A
```

本讲内容

1.1 为什么学习算法

1.2 什么是算法

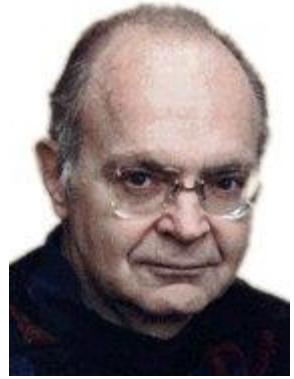
1.3 计算机科学中算法的位置

1.4 算法分析引论

1.5 算法设计引论

算法是计算机科学基础的重要主题

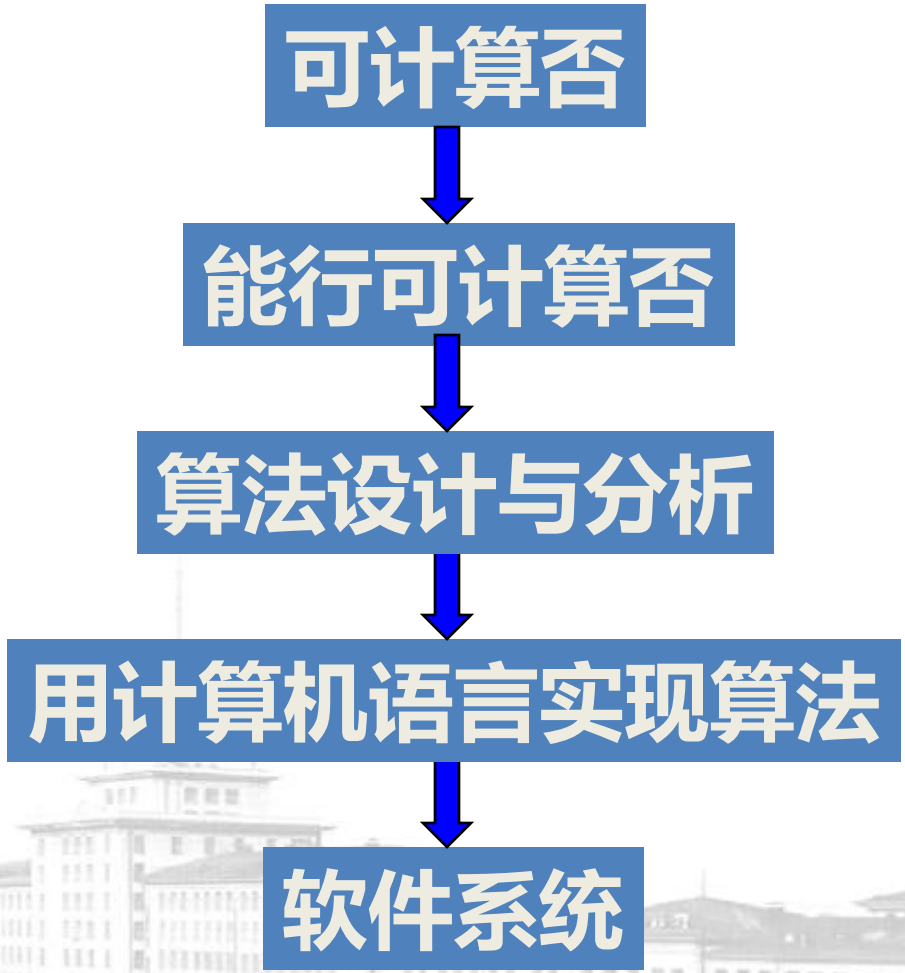
- **70年代前**
  - •计算机科学基础的主题没有被清楚地认清。
- **•70年代**
  - •Knuth出版了《The Art of Computer Programming》
  - •以算法研究为主线,确立了算法为计算机科学基础的重要主题
  - •1974年获得图灵奖。



- **•70年代后**
  - •算法作为计算机科学核心推动了计算机科学技术飞速发展

计算机科学的体系

- 解决一个计算问题的过程



计算机科学的体系

- **可计算理论**
  - ➢计算模型
  - ➢可计算问题/不可计算问题
  - ➢计算模型的等价性--图灵/Church命题
- **计算复杂性理论**
  - ➢研究在给定的计算模型下研究问题的复杂性
  - ➢固有复杂性:上界、下界、平均
  - ➢复杂性问题的分类: P=NP?
  - ➢抽象复杂性研究

计算机科学的体系

- **算法设计和分析**
  - ➢可计算问题的算法的设计与分析
  - ➢设计算法的理论、方法和技术
  - ➢分析算法的理论、方法和技术
- **计算机软件**
  - ➢系统软件
  - ➢工具软件
  - ➢应用软件

本讲内容

1.1 为什么学习算法

1.2 什么是算法

1.3 计算机科学中算法的位置

1.4 算法分析引论

1.5 算法设计引论

算法的正确性分析

- **算法正确性**
  - 一个算法是正确的，如果它对于每一个输入都最终停止，而且产生正确的输出。
  - 不正确算法：

    ①不停止(在某个输入上)

    ②对所有输入都停止，但对某输入产生不正确结果
  - 近似算法

    ①对所有输入都停止

    ②产生近似正确的解或产生不多的不正确解

算法的正确性分析

- **算法正确性证明**
  - 证明算法对所有输入都停止
  - 证明对每个输入都产生正确结果
  - 调试程序≠算法正确性证明

程序调试只能证明程序有错，

不能证明程序无错误!

插入排序的正确性

- **循环不变量:**在每次循环的开始，子数组A[1..j-1] 包含原来数组中A[1..j-1] 但是已经有序

证明:

- 初始化 : j=2, A[1..j-1]=A[1..1]=A[1], 已经有序
- 维护：每一层循环维护循环不变量
- 终止: j=n+1, 所以，A[1..j-1]=A[1..n] 有序

算法的复杂性分析

- **目的：**
  - 预测算法对不同输入所需资源量
- **复杂性测度：**
  - 时间，空间， I/O等, 是输入大小的函数
- **用途：**
  - 为求解一个问题选择最佳算法、最佳设备
- **需要的数学基础**
  - 离散数学，组合数学，概率论，代数等
- **需要的数学能力**
  - 建立算法复杂性的数学模型
  - 数学模型化简

算法复杂性分析的度量

- **输入的大小**
  - 设Input是问题R的输入集合，R的输入大小是一个函数 F：Input→N，N是正整数集合。

  示例：
    - 矩阵问题的输入大小=矩阵的维数
    - 图论问题的输入大小=图的边数/结点数

算法复杂性分析的度量

- **时间复杂性:**一个算法对特定输入的时间复杂性是该算法对该输入产生结果需要的原子操作或“步”数
- **注意:**
  - 时间复杂性是输入大小的函数
  - 我们假设每一步的执行需要常数时间，实际上每步需要的时间量可能不同

算法复杂性分析的度量

- **空间复杂性:**一个算法对特定输入的空间复杂性是该算法对该输入产生结果所需要的存储空间大小。

算法复杂性分析的度量

- **最坏复杂性**
  - 设Input是输入集合，complexity(X)是算法A的复杂性函数，size(y)是输入y的大小，A的最坏复杂性是

$$\mathbf{Max}\{\text{Complexity}(\text{size}(y)) \mid y \in \text{Input}\}$$

- **最小复杂性**

$$\mathbf{Min}\{\text{Complexity}(\text{size}(y)) \mid y \in \text{Input}\}$$

- **平均复杂性**
  - 设$y \in Input$，y出现的概率是p_y，A的平均复杂性为：

$$\sum_{y \in Input} complexity(size(y)) \times p_y$$

算法分析的模型

- 随机访问模型 (Random-Access-Model ,RAM)
  - ➢单处理机，串行执行，无并发
  - ➢基本数据类型
  - ➢基本操作(每个操作常数时间)
- 并行多处理机模型(PRAM)

插入排序的分析

INSERTION-SORT(A)	代价	次数
1. **for** j = 2 to length[A]	c_1	n
2. **do** $key \leftarrow A[j]$	c_2	n-1
3. //insert A[j] to sorted sequence A[1..j-1]	0	n-1
4. $i \leftarrow j$-1	c_4	n-1
5. **while** i >0 and A[i]>key	c_5	$\sum_{j=2}^{n} t_j$
6. **do** A[i+1] $\leftarrow$ A[i]	c_6	$\sum_{j=2}^{n} (t_j-1)$
7. $i \leftarrow i$-1	c_7	$\sum_{j=2}^{n} (t_j-1)$
8. A[i+1] $\leftarrow$ key	c_8	$n-1$

(t_j 是对j来说循环中执行的次数)

总时间代价$T(n)$ = 代价的和 × 每行执行次数

$$=c_1(n\text{-}1) + c_2(n\text{-}1) + c_4(n\text{-}1) + c_5\sum_{j=2}^{n} t_j + c_6\sum_{j=2}^{n}(t_j\text{-}1) + c_7\sum_{j=2}^{n}(t_j\text{-}1) + c_8(n\text{-}1)$$

插入排序的分析(续)

- **最好代价: 有序的数组**
  - t_j=1, 且6和7行执行0次
  - $T(n) = c_1n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$

    $=(c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8) = cn + c'$
- **最坏代价: 逆序数组**
  - $t_j=j$, $\sum_{j=2}^{n} t_j = \sum_{j=2}^{n} j = n(n+1)/2-1$, 且$\sum_{j=2}^{n}(t_j-1) = \sum_{j=2}^{n}(j-1) = n(n-1)/2$
  - $T(n) = c_1n + c_2(n-1) + c_4(n-1) + c_5(n(n+1)/2 -1) + c_6(n(n-1)/2) + c_7(n(n-1)/2)+ c_8(n-1) =((c_5 + c_6 + c_7)/2)n^2 +(c_1 + c_2 + c_4 +c_5/2-c_6/2-c_7/2+c_8)n-(c_2 + c_4 + c_5 + c_8) =an^2+bn+c$
- **平均代价: 随机数**
  - 平均来看，$t_j = j/2$, $T(n)$ 与n^2同阶，和最坏情况相同。

本讲内容

1.1 为什么学习算法

1.2 什么是算法

1.3 计算机科学中算法的位置

1.4 算法分析引论

1.5 算法设计引论

算法设计模式

- 暴力搜索
- 分治法
- 图搜索与枚举
  - 分支界限
  - 回溯
- 随机化方法

算法实现方法

- 递归与迭代
- 顺序、并行与分布式
- 确定性与非确定性
- 近似求解与精确求解
- 量子算法

最优化算法设计方法

- 线性规划
- 动态规划
- 贪心法
- 启发式方法