



福昕高级PDF编辑器

高效 · 安全 · 专业

立即下载

点击购买



OFFICE格式互转



OCR文字识别



文本图像编辑



加密和签署



交互式动态表单



互联PDF文档



中国科学技术大学 计算机科学与技术系
University of Science and Technology of China
DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

算法设计与分析

Design and Analysis of Algorithms

主讲人 徐云

Fall 2012, USTC



Part 1 Foundation

Part 2 Sorting and Order Statistics

Part 3 Data Structure

Part 4 Advanced Design and Analysis Techniques

Part 5 Advanced Data Structures

Part 6 Graph Algorithms

Part 7 Selected Topics

...

chap 34 Computation Models and NP-Completeness

...

Part 8 Supplement



第34章 计算模型和NP完全

34.1 引言

34.2 图灵机模型

34.3 NP完全问题



34.1 引言

- 早期的计算模型
- 算法与计算模型的关系
- 计算模型的作用
- 图灵机与语言识别问题
- 通用的图灵机

早期的计算模型

- Recursive Function Theory - Kleene, Church, Turing, Post, 1930's
- Turing Machines - Turing, 1930's
- RAM Machines - von Neumann, 1940's
- Cellular Automata - von Neumann, 1950's
- Finite-state machines, pushdown automata
various people, 1950's
- VLSI models - 1970's
- Parallel RAMs, etc. - 1980's
随机存取机器

算法与计算模型的关系

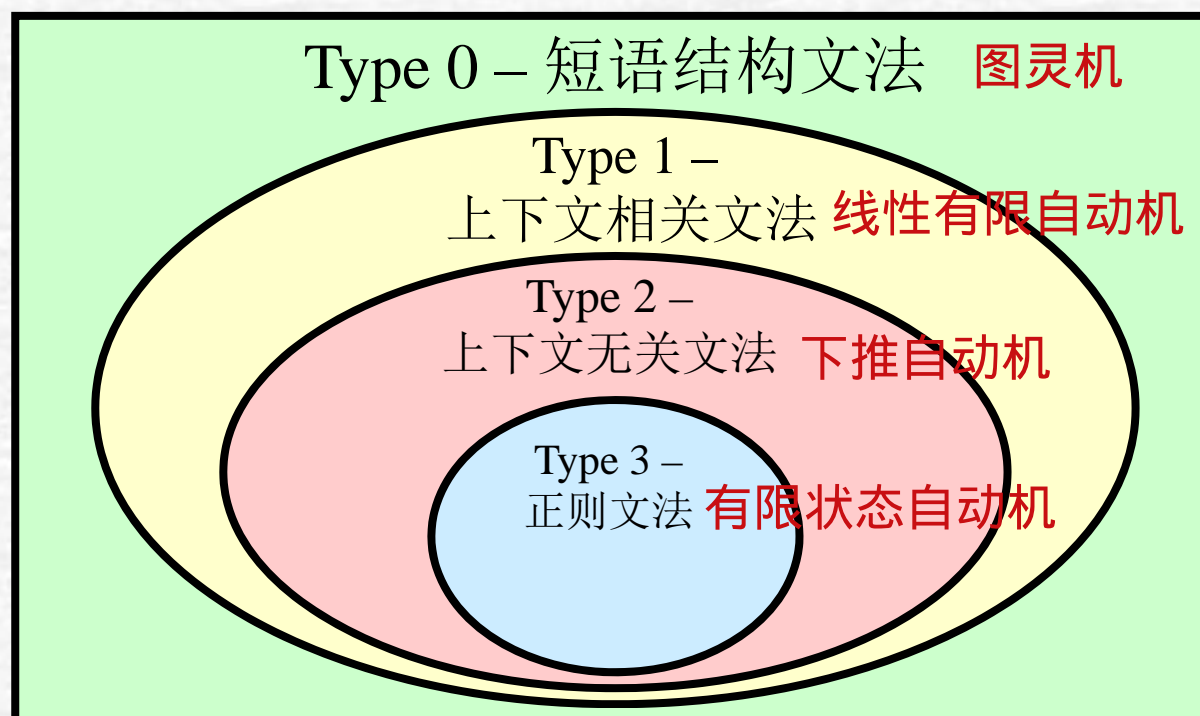
- 非形式地说，算法是为实现某个任务而构造的简单指令集。这是一个非严格的算法定义。
- 1900年，Hilbert在巴黎举行的世界数学家大会上提出了23个数学问题，其中第10个问题是要设计一个算法来测试多项式是否有整数根，他没有用算法这个术语，而用这样一句短语：“通过有限多次运算就可以决定的过程”。
 - 该问题是算法上不可解的，1970年由Yuri Matijasevic解决。
 - 关键在于算法的精确定义，后来由Church和Turing分别解决，从此称为丘奇-图灵论题。
 - Church使用 λ 演算定义算法和Turing使用图灵机定义算法。
在图灵机上有限时间可以完成
- 现在严格地讲，一个问题算法可解的等于该问题在图灵机上可解(可判断)。

计算模型的作用

- 对于一个计算任务，有两个问题要解决：
 - 该计算任务能否在一个计算机上实现？
 - 该计算任务在一个计算机上实现的复杂度？
- 计算模型可以帮助我们解决上述问题，本课程我们主要学习图灵机模型。
- 计算模型的主要作用：
 - **可计算性**：将问题按计算模型的可计算性进行分类。也就是回答一个问题在某种计算模型上是否可计算，而不计较其计算时间的长短。
 - **计算复杂性**：将问题按计算模型的计算复杂性(时间和空间)进行分类。**多项式时间、指数时间**
 - **可编程性**：在计算模型下算法的实现。

图灵机与语言识别问题 (1)

- 语言识别问题和包含关系：



图灵机与语言识别问题 (2)

- 有限状态自动机能够识别正则文法生成的语言；
- 下推自动机能够识别上下文无关文法生成的语言；
- 线性有界自动机能够识别上下文相关文法生成的语言；
- 然而上述自动机不能识别短语结构文法生成的语言。图灵机能够识别短语结构文法生成的所有语言，是一种能力很强的计算模型。

通用的图灵机 (1)

- TM's we saw executed a specific algorithm
 - A different TM needed for a different alg.
 - Or, re-wire the machine
- Turing (in 1936) foresaw the *stored-program computer*
 - Flexibility to execute different algorithms
 - Turing describes a Universal TM
- “a TM is a general model of computation”
 - Means: *any algorithmic procedure that can be carried out (by a human or a computer) can be carried out by a TM*
- First formulated by Alonzo Church (1936)
 - Referred to as Church's thesis also
 - Not a precise statement because “algorithmic procedure” is undefined → cannot prove

通用的图灵机 (2)

- The thesis is generally accepted, because
 - Nature of model indicates all steps crucial to human computation can be carried out
 - No one has proposed an “algorithmic procedure” that cannot be implemented in TM
 - Various enhancements do not enhance the computing power
 - Other theoretical models have been shown to be equivalent to a TM

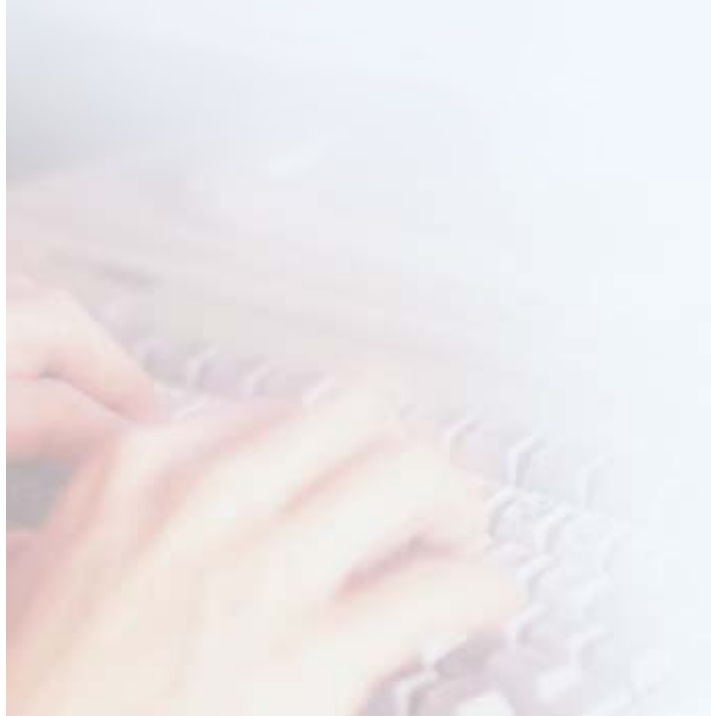


第34章 NP完全

34.1 引言

34.2 图灵机模型

34.3 NP完全问题



34.2 图灵机模型

- 确定型图灵机
- 确定型图灵机计算示例
- 非确定型图灵机

确定型图灵机DTM (1)

- 图灵机模型(Turing 1936): 最通用的计算模型
 - 能做现代计算机所能做的一切;
 - 但实现的效率和方式有所不同。
- 思想源于: 模仿人在一个无限长的带格子的纸带上写字
 - 人使用一个带橡皮头的铅笔来写字;
 - 写字从某个格子开始, 或者仅仅读取该格子中的符号, 或者擦取该格子中的符号并写上一个新的符号;
 - 写字以这种连续方式进行: 沿着纸带的两个方向, 向相邻的格子进行写字和读字。

确定型图灵机DTM (2)

有限状态控制器 算法

..... 1 0 B 1 0 1 1 0 0 B 0 1 1 0 0 1 B

- 两端无限长的带子，带子可读可写，带子作为输入设备、存储设备和输出设备；
- 有限状态控制器只含有限个状态，其中有开始状态、接受状态和拒绝状态；
- 读写头可左移右移，取决于当前状态和当前格子中的符号；
- 一个单步移动的构成：当前格子中读或写，改变状态，根据状态确定左移/右移/停机；

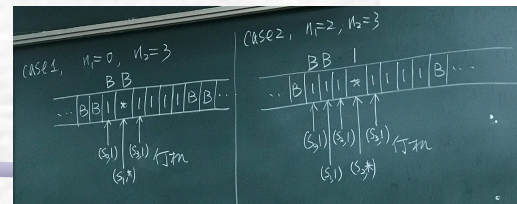
确定型图灵机DTM (3)

- 图灵机的形式定义:

A TM is $T=(S, I, f, s_0, s_{\text{accept}}, s_{\text{reject}})$, where

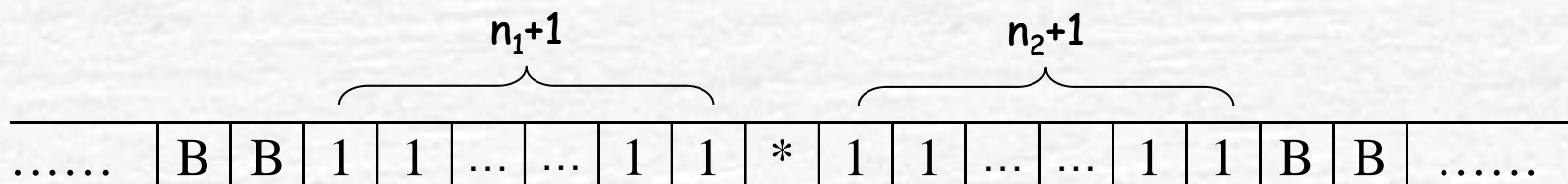
- S : a finite set of states 有限状态集
- I : a finite input alphabet (excluding blank B) 有限字符集
- f : is the transition function: $S \times I \rightarrow S \times I \times \{R, L\}$ [映射 f may not be defined for some points]
- s_0 : the start state 开始状态
- s_{accept} : the accept state true
- s_{reject} : the reject state false

确定型图灵机计算示例



- 示例：构造一个图灵机求两个非负整数 n_1 与 n_2 的和

解：初始带上的数据如下：



通过状态变化五元组(即 f)，使得停机时带上产生连续的 n_1+n_2+1 个1

下面设计状态变化五元组有：

$(s_0, 1, s_1, B, R)$ // 当前状态和数据 $(s_0, 1) \rightarrow (s_1, B)$ ，并右移

$(s_1, *, s_3, B, R)$ // s_3 不定义状态转移函数，使得机器停机

$(s_1, 1, s_2, B, R)$

$(s_2, 1, s_2, 1, R)$ // 当前状态和数据为 $(s_2, 1)$ 时，右移

$(s_2, *, s_3, 1, R)$ // 当前状态和数据为 $(s_2, *)$ 变换到 $(s_3, 1)$

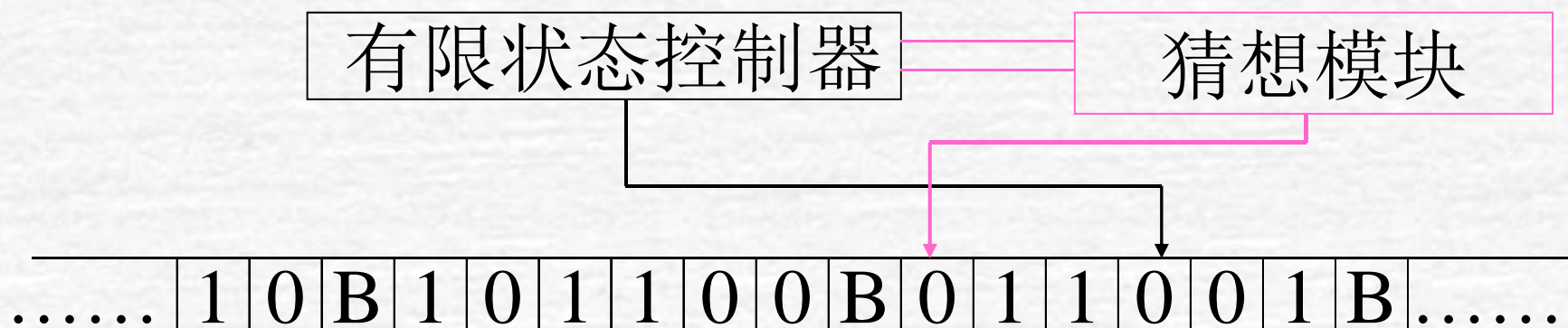
开始时机器读写头指向最左端的1，状态为 s_0 ，以后状态变化序列为：

$n_1 > 0$: $(s_0, 1, s_1, B, R), (s_1, 1, s_2, B, R), (s_2, 1, s_2, 1, R), \dots, (s_2, 1, s_2, 1, R), (s_2, *, s_3, 1, R)$

$n_1 = 0$: $(s_0, 1, s_1, B, R), (s_1, *, s_3, B, R)$

非确定型图灵机NTM

假想



- 分为猜想阶段和验证阶段；
- 不现实的计算
 - 现实中的计算方式都是确定的
- 解SAT问题的一个非确定型算法
 - 第一步：猜测一个变量的真值赋值；
 - 第二步：检查该赋值是否满足
- 非确定型算法的计算时间：
 - 各种可能的计算过程的最短时间

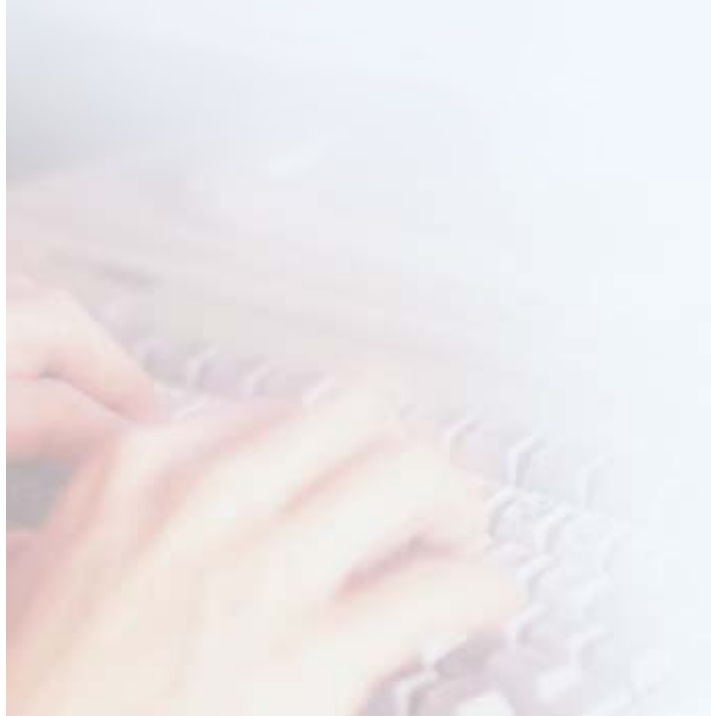


第34章 NP完全

34.1 引言

34.2 图灵机模型

34.3 NP完全问题



P问题和NP问题

- 判定问题：只有肯定和否定两种答案
 - 优化问题可以化作判定问题处理；由判断问题的多次迭代得到
- P问题：
 - 具有多项式时间算法的判定问题类；
- NP问题：
 - 在非确定型图灵机上多项式时间可解的问题；
 - 在确定型图灵机上多项式时间可验证的问题；
- 一个结论：P类包含在NP类中
- NP类问题在确定图灵机上指数时间可解，如果P<>NP_{一部分难的问题}

计算难度的比较——归约

- 多项式时间归约 (Karp归约 1972);
- 问题 A 的实例 I 多项式时间内转化为问题 B 的实例 $f(I)$ ，对于 A 的输入 I 的回答与其对应的 B 的输入 $f(I)$ 一致，则称 A 可多项式归约于 B ，记为 $A \leq_p B$ ； A 的难度不会高于 B
- 如果 B 可以多项式时间求解，则 A 也可以多项式时间求解；

NP完全问题 (1)

- 定义:

问题 p 是NP完全 (NPC) 问题,

$p \in NPC : 1. p \in NP; 2. \forall q \in NP, \text{有 } q \leq_p p$

- NP完全问题是NP问题中”最难“的问题

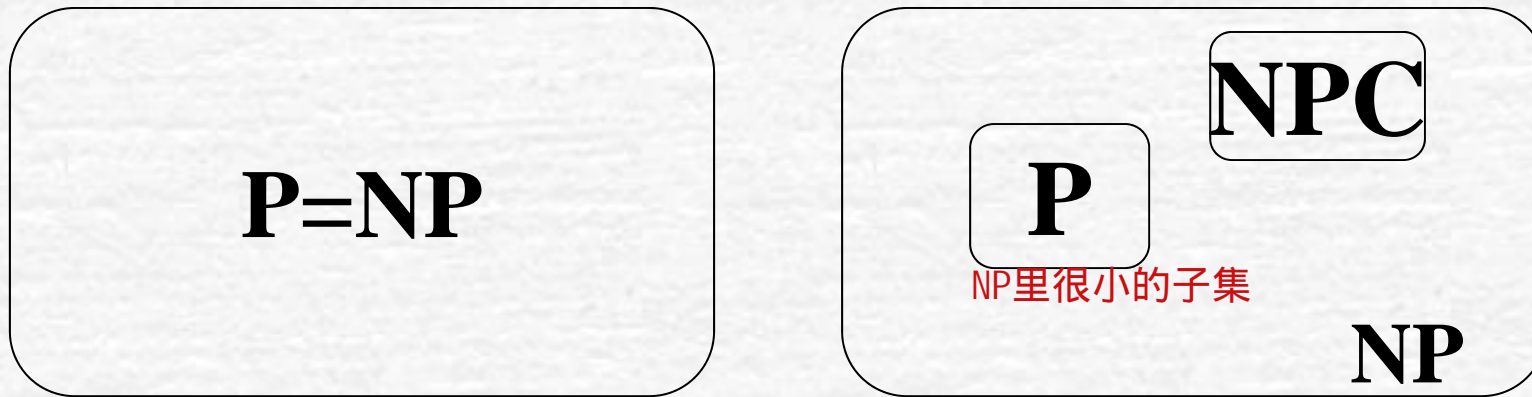
NP完全问题 (2)

- 第一个NP完全问题 (Cook-levin定理 1971)
 - 可满足性问题3SAT是第一个NP完全问题
- 如果一个NP完全问题多项式时间归约到另一个NP问题，则该问题也是NP完全的
- 六个NP完全问题 (Karp 1972)
 - 3SAT, 3DM, VC, 团, HC, 划分
- 更多的NP完全问题
 - 1979年: 300多个
 - 1998年: 2000多个

优化问题: NPHard

$P=?NP$ (P-NP问题)

$$P \cap NPC \neq \phi \Rightarrow P = NP$$



现在的估计

如果 $P \neq NP$ ，则NPC问题无有效算法

如何处理NP完全问题

- 实际中的NP完全问题不会消失;
- 证明难度并不会使问题得到解决;
- 如何处理NP完全问题:
 - 近似算法
 - 随机算法
 - 并行计算
 - ...
 - 新型的计算模型(计算机): DNA计算, 量子计算, 等



End of Ch34

