图灵可计算性

(Turing Computability)

熊 明
mingshone@163.com

South China Normal University

主要内容

主要内容

1. 问题来源
2. 图灵机
3. 可计算函数

Hilbert 的 Entscheidungsproblem

1928 年，Hilbert 在与其学生合著的书中，

- David Hilbert and Wilhelm Ackermann (1928). Grundzüge der theoretischen Logik (Principles of Mathematical Logic). Springer-Verlag.

提出如下 Entscheidungsproblem （判定问题）：

是否有一种“能行的”（effective）方法，用于判定一个一阶公式是否是有效的?

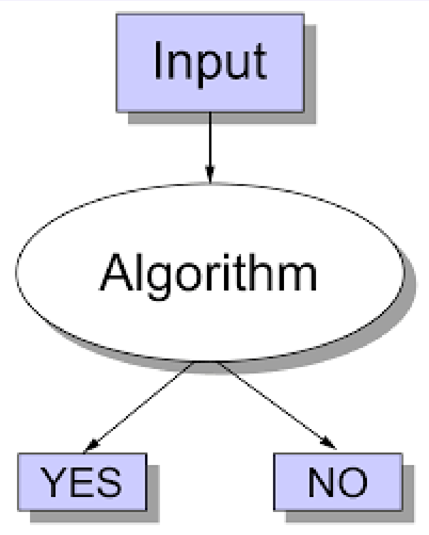
能行性

是否存在一种程序（或算法、机械方法），当输入一个一阶公式到程序中，

“能行性”要求程序作出以下两个选择之一：

- 程序回答“这个公式有效”;
- 程序回答“这个公式无效”。

能行性图示



已知的例子 1

- 判定一个命题逻辑公式（布尔公式）是否是有效的?

判定方法：真值表方法、解析树方法!

已知的例子 1

- 判定一个命题逻辑公式（布尔公式）是否是有效的?

判定方法：真值表方法、解析树方法!

已知的例子 2

- 判定一个正整数是否是素数?

判定方法：埃拉托斯特尼（Eratosthenes）筛法！

已知的例子 2

- 判定一个正整数是否是素数?

  判定方法：埃拉托斯特尼（Eratosthenes）筛法！

解析树

就一阶公式的判定而言，解析树仅仅是一个“半能行判定”方法：

当输入一个一阶公式到解析树程序中，解析树程序作出的回答有**三种**可能性：

- 程序回答“这个公式有效”；
- 程序回答“这个公式无效”；
- **程序回答“不知道！”（或者进入死机状态）。**

Hilbert 的 Entscheidungsproblem 的回答

1936 年，美国的 Church 和英国的 Turing 各自独立地回答了 Hilbert 的问题：

- Alonzo Church,"An unsolvable problem of elementary number theory", *American Journal of Mathematics*, 58 (1936), pp 345–363.
- Alan Turing, "On computable numbers, with an application to the Entscheidungsproblem", *Proceedings of the London Mathematical Society*, Series 2, 42 (1936-7), pp 230–265.

他们的回答是否定的！

Hilbert 的 Entscheidungsproblem 的回答

1936 年，美国的 Church 和英国的 Turing 各自独立地回答了 Hilbert 的问题:

- Alonzo Church,"An unsolvable problem of elementary number theory", *American Journal of Mathematics*, 58 (1936), pp 345–363.
- Alan Turing, "On computable numbers, with an application to the Entscheidungsproblem", *Proceedings of the London Mathematical Society*, Series 2, 42 (1936-7), pp 230–265.

他们的回答是否定的!

- 如果是肯定性的回答，只需给出一种算法即可。例如，在命题逻辑中，这种算法是存在的，我们事实上也给出了两种算法：真值表法和解析树法。
- 如果是否定性的回答，则必须证明**所有**算法能无法达到判定 Entscheidungsproblem 的目的。为此，首先必须回答：什么是算法（程序、机械方法）?

- 如果是肯定性的回答，只需给出一种算法即可。例如，在命题逻辑中，这种算法是存在的，我们事实上也给出了两种算法：真值表法和解析树法。
- 如果是否定性的回答，则必须证明**所有**算法能无法达到判定 Entscheidungsproblem 的目的。为此，首先必须回答：什么是算法（程序、机械方法）?

算法定义大事记

- 前奏
  - Gödel (1931)：原始递归函数（用于证明不完全性定理）
- 对算法的规定
  - 1936 年：
    - Church: λ-演算（编程语言（如：Java、python、Lisp 等）的基石）
    - **Turing：Turing 机**
    - Gödel and Kleene: 部分递归函数
  - 1936 年之后
    - Post (1943): Post 系统
    - Markov (1951)：Markov 算法
    - Shepherdson and Sturgis (1963): 无穷存储机
    - …
    - 以上手段规定出来的算法都是等价的！

假想对话

- 诸逻辑学家（问 Gödel）：这么多的算法规定，哪一种最符合我们关于算法的直观？
- Gödel：我只服 Turing 规定的！
- 诸逻辑学家：我们都服你！

假想对话

- 诸逻辑学家（问 Gödel）：这么多的算法规定，哪一种最符合我们关于算法的直观?
- Gödel：我只服 Turing 规定的！
- 诸逻辑学家：我们都服你！

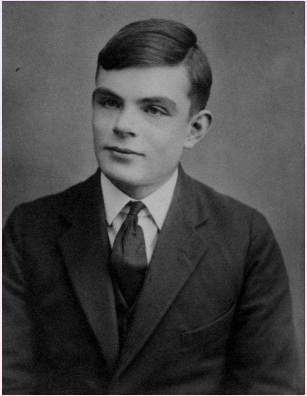
假想对话

- 诸逻辑学家（问 Gödel）：这么多的算法规定，哪一种最符合我们关于算法的直观？
- Gödel：我只服 Turing 规定的！
- 诸逻辑学家：我们都服你！

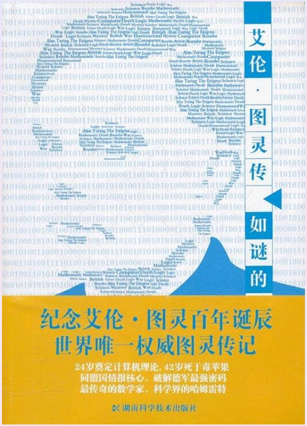
主要内容

阿兰 · 图灵（Alan Turing）

- Born: 23 June 1912 in Paddington, London
- Died: 7 June 1954 in Wilmslow, Cheshire
- A British pioneering computer scientist, mathematician, logician, cryptanalyst and theoretical biologist

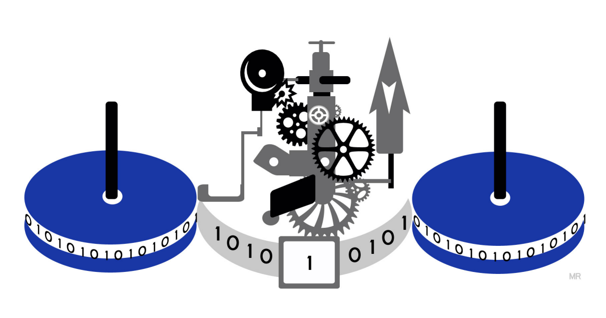


模仿游戏：The Imitation of Game

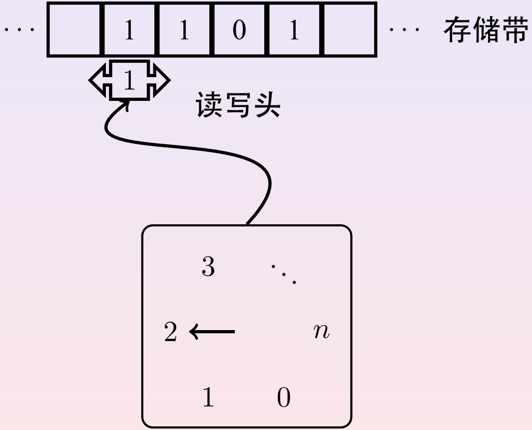


影片改编自：安德鲁·霍奇斯，《艾伦·图灵传—如谜的解谜者》，孙天齐（译），湖南科学技术出版社，2012。

图灵机的硬件模型之一



图灵机的硬件模型之二



图灵机的软件

- 图灵机软件 = 程序

图灵机的程序

- 程序由指令构成。
- 指令的形式：

< 当前状态 >< 当前符号 >< 新符号 >< 移动方向 >< 新状态 >

- 指令的意义：如果读写头处在 < 当前状态 >，在正扫描的格子中读到 < 当前符号 >，那么在扫描到的格子中用 < 新符号 > 代替 < 当前符号 >，并按 < 移动方向 > 进行移动，同时状态改变为 < 新状态 >。
- 表示方法：状态—自然数，符号—数字或字母（_ 表示空格），移动方向—l（左移一格）、r（右移一格）、*（保持不动）。

指令的例子

- 指令 0 _ 1 1 1：如果读写头当前状态的是 0，当前扫描的格子为空格，那么在扫描到的格子中写入 1，然后左移一格，状态变为 1。
- 指令 0 1 1 r 0：如果读写头当前状态的是 0 并在当前扫描的格子中读到 1，那么保持这个格子中内容不变（先擦掉 1 再写入 1），然后右移一格，状态保持不变。

程序 1 及其执行情况

程序 1

0 1 1 r 0

0 _ 1 1 1

1 1 1 1 1

1 _ _ r 2

转到网页： Turing Machine Simulator

程序 1 及其执行情况

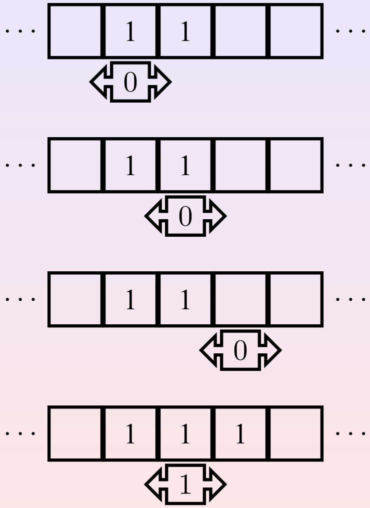
程序 1

0 1 1 r 0

0 _ 1 1 1

1 1 1 1 1

1 _ _ r 2



程序 2 及其执行情况

程序 2

0 _ 1 r 0

转到网页： Turing Machine Simulator

图灵机与程序

图灵机只是因为程序不同而不同，以后约定：图灵机的程序看做是图灵机本身。

这意味着：写出一个图灵机程序，就等于给出了一个图灵机。

主要内容

1. 问题来源
2. 图灵机
3. 可计算函数

图灵机计算后继函数

- 后继函数：$f(n) = n + 1$

图灵机 1

0 1 1 r 0

0 _ 1 1 1

1 1 1 1 1

1 _ _ r 2

计算过程

- 输入：n，代码：11...1（$n + 1$ 个 1）
- 开动图灵机 1
- 输出：$n + 1$，代码：11...1（$n + 2$ 个 1）

图灵机计算加法函数

- 加法函数：$f(n, m) = n + m$

图灵机 2

0 1 1 r 0
0 0 0 r 0
0 _ _ 1 1
1 1 _ 1 2
1 0 1 * 2
2 0 1 1 2
2 1 1 1 2
2 _ _ r 3
3 1 _ r 4

计算过程

- 输入：
  n 与 m，代码：11...1（$n+1$ 个 1），
  11...1（$m+1$ 个 1)
- 开动图灵机 2
- 输出：
  $n+m$，代码：11...1（$n+m+1$ 个 1）

图灵可计算函数：一元函数情形

- 函数：$f(x) = y$

如果存在图灵机，使得

- 输入 $x+1$ 个 1（其余格子都为空格）
- 执行这个图灵机，
- 输出为 $y+1$ 个 1（其余格子都为空格）

那么就称这个图灵机**计算**了函数 f。

图灵可计算函数：二元函数情形

- 函数：$f(x_1, x_2) = y$

如果存在图灵机，使得

- 输入 $x_1 + 1$ 个 1 和 $x_2 + 1$ 个 1（中间用 0 分开，其余格子都为空格）
- 执行这个图灵机，
- 输出为 $y + 1$ 个 1（其余格子都为空格）

那么就称这个图灵机**计算**了函数 f。

图灵可计算函数

任何一个函数，如果存在一个图灵机来计算它，就称这个函数是**图灵可计算的**，常常简称可计算的。

丘齐-图灵论题

丘齐-图灵论题

- （直观）算法 = 图灵机上的算法。
- 算法可计算 = 图灵可计算。

丘齐-图灵论题

丘齐-图灵论题

- （直观）算法 = 图灵机上的算法。
- 算法可计算 = 图灵可计算。

丘齐-图灵论题的一个应用

丘齐-图灵论题

算法可计算 = 图灵可计算。

函数

$$f(x) = \begin{cases} 1 & \text{若 } x \text{ 是素数;} \\ 0 & \text{否则。} \end{cases}$$

是图灵可计算的。

Thanks for your attention!

Q & A