3 文法的形式化描述

杨策

3 文法的形式化描述

- 可计算性理论入门
- BNF范式
- 乔姆斯基文法

公理的形式化

- 欧几里德：几何原本
  - 第五公设
- 康托集合论
- 罗素悖论

需要避开这种定义方式

  - 符合某性质的元素总体是一个集合
- 希尔伯特纲领
  - 数学公理、证明形式化

原始递归

- 自然数域N上的函数
- 基础函数
  - 常函数$f(x)=0$
  - 后继函数$f(x)=x+1$
  - 投影函数$f(x_1, \ldots, x_n)=x_i$
- 复合规则
  - m元函数g, m个n元函数hi，复合成n元函数
  - $f(x_1, \ldots, x_n)=g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$
- 递归规则
  - $f(0)=c$
  - $f(n+1)=h(n, f(n))$

Primitive Recursive Functions

原始递归

- 自然数域N上的函数
- 基础函数
  - 常函数$f(x)=0$
  - 后继函数$f(x)=x+1$
  - 投影函数$f(x_1, \ldots, x_n)=x_i$
- 3个基础函数使用有限次复合和递归得到的函数为原始递归
- 原始递归函数是可计算的

- 复合规则
  - m元函数g, m个n元函数hi，复合成n元函数
  - $f(x_1, \ldots, x_n)=g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$
- 递归规则
  - k元函数g，k+2元函数h
  - $f(x_1, \ldots, x_k, 0)=g(x_1, \ldots, x_k)$
  - $f(x_1, \ldots, x_k, n+1)=$
  - $h(x_1, \ldots, x_k, n, f(x_1, \ldots, x_k, n))$

原始递归

- 加法 $f(x, y)=x+y$
- $g(x)=x$ 投影函数
- $f(x, y)$ 递归
  - $f(x, 0)=g(x)=x$
  - $f(x, y)=f(x, y-1)+1=\ldots=f(x, 0)+y$
  - $f(x, y)=h(x, y-1, f(x, y-1))$
  - $h(x, y-1, f(x, y-1))=f(x, y-1)+1$

- $h(x, y, z)=z+1$
  - $h(x, y, z)=p(q(x, y, z))$ 复合
  - $q(x, y, z)=z$ 投影函数
  - $p(z)=z+1$ 后续函数

原始递归

- 乘法 $f(x, y)=x*y$
- $g(x)=0$ 常函数
- $f(x, y)=f(x, y-1)+x$ 递归
  - $f(x, 0)=g(x)=0$
  - $f(x, y)=f(x, y-1)+x=\ldots=f(x, 0)+x*y$
  - $f(x, y)=h(x, y-1, f(x, y-1))$
  - $h(x, y-1, f(x, y-1))=f(x, y-1)+x$
- $h(x, y, z)=x+z$
  - $h(x, y, z)=p(q_1(x, y, z), q_2(x, y, z))$
  - $q_1(x, y, z)=x$ 投影函数
  - $q_2(x, y, z)=z$ 投影函数
  - $p(x, z)=x+z$ 加法函数

部分递归

- 原始递归的问题
  - 能覆盖的函数太少
  - 阿克曼函数不是原始递归函数
    - 增长比所有原始递归函数快

阿克曼函数 $A : \mathbb{N}^2 \to \mathbb{N}$ 由如下定义:

$$
\begin{array}{lll}
A(0,n) & = n+1 & \text{对于 } n \geq 0 \\
A(m+1,0) & = A(m,1) & \text{对于 } m \geq 0 \\
A(m+1,n+1) & = A(m, A(m+1,n)) & \text{对于 } m,n \geq 0
\end{array}
$$

如果第一位参数是固定的:

$$
\begin{aligned}
A(1,n) &= 2+(n+3)-3, \\
A(2,n) &= 2*(n+3)-3, \\
A(3,n) &= 2^{n+3}-3, \\
A(4,n) &= \underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{n+3\text{次}2}-3,
\end{aligned}
$$

部分递归

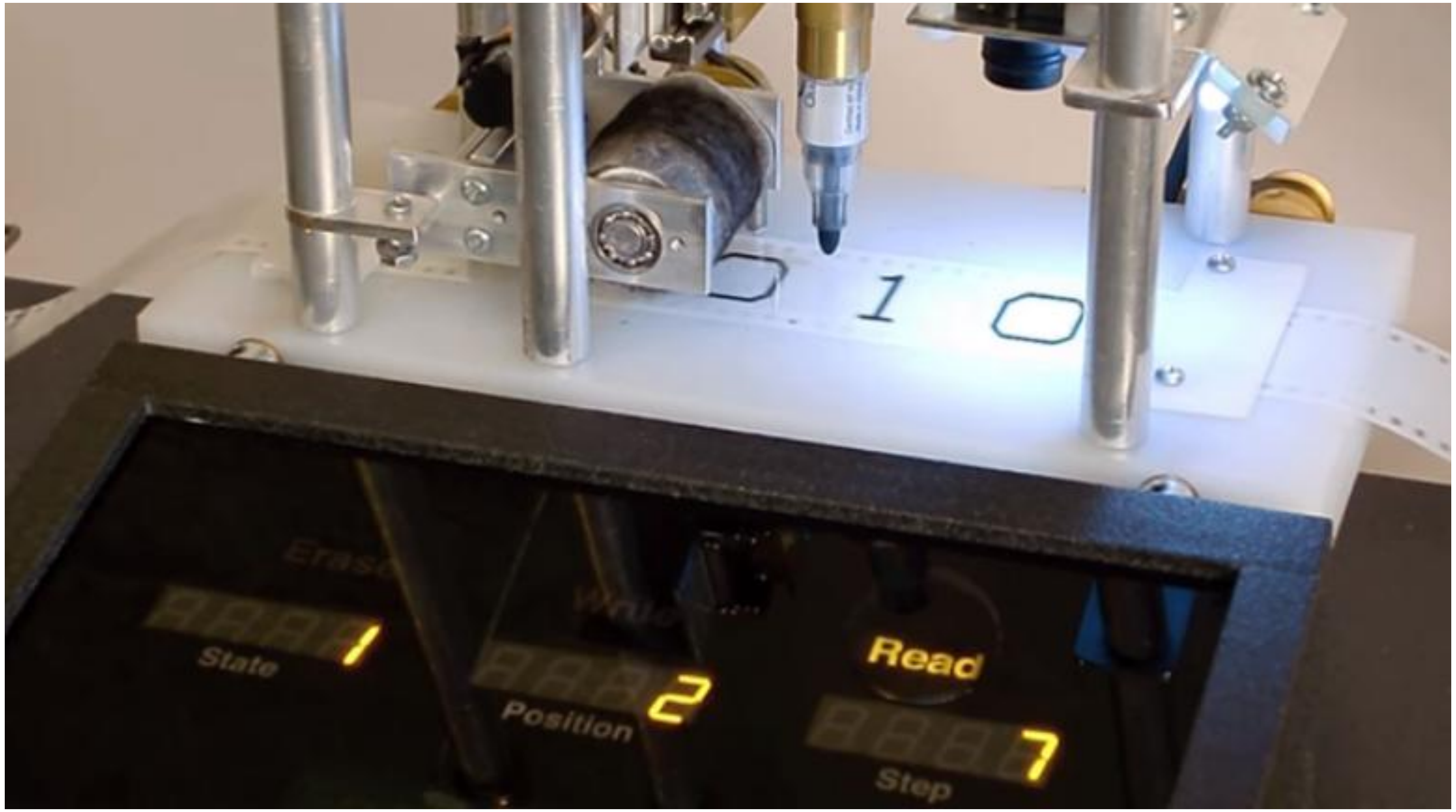
- 增加一条规则：μ规则
  - k+1元函数g，对任意$n_1, \ldots, n_k$都有解x使得$g(n_1, \ldots, n_k, x)=0$
  - 得到新的k元函数$f(n_1, \ldots, n_k)=\min\{x \mid g(n_1, \ldots, n_k, x)=0\}$
- 部分递归函数
  - 三种基本函数
  - 部分递归函数使用有限次复合、递归、μ规则进行构造
- 部分递归函数(哥德尔)=lambda演算(丘奇)=图灵机(图灵)

https://blog.csdn.net/weixin_60738001/article/details/135345582

图灵机

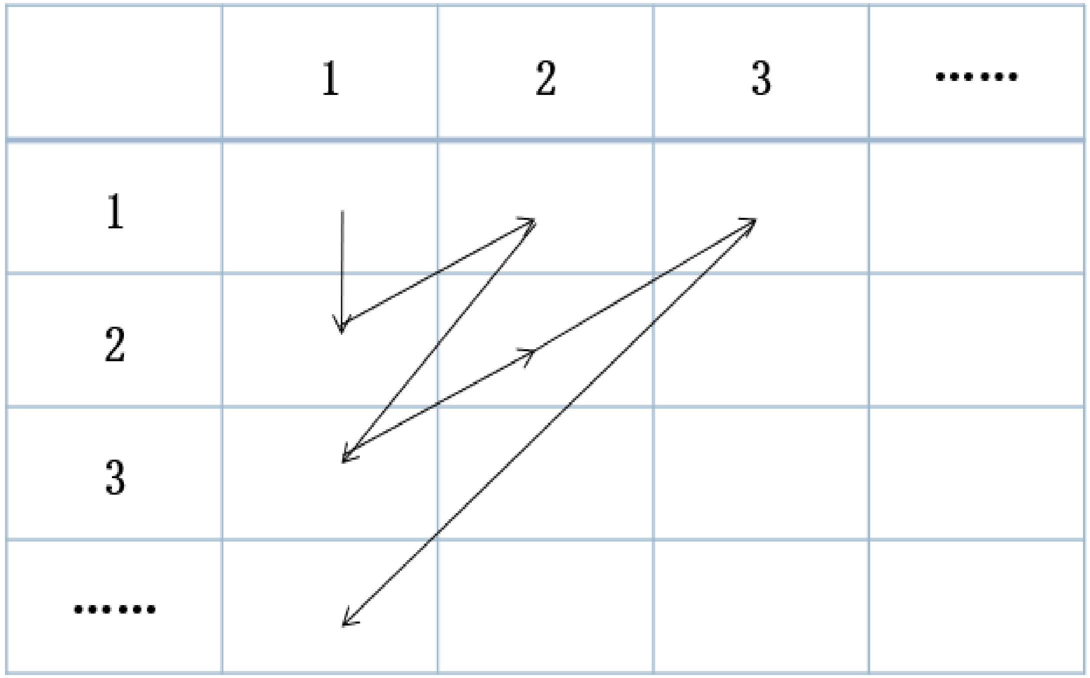
- 数据
  - 纸带
  - 内部状态
- 操作
  - 读/写
  - 左移/右移
- 通用图灵机

编译器/解释器



图灵机

- 无穷集比大小
  - f(A) → B，f是单射，则认为|A|≤|B|
  - |A|≤|B| & |B|≤|A|，则认为|A|=|B|
  - 可数集：|A|=|N|，N是自然数集合
  - 有理数集合可数



图灵机

- 实数是否可数？对角线法

	1	2	3	4	5	……
1	0	1	1	0	1	
2	0	1	1	1	1	
3	1	0	0	1	1	
4	0	1	1	1	0	
……						

图灵机

- 停机问题
  - 是否存在一个图灵机，判定其他任意一个图灵机在任意一种输入下是否会停机
  - 图灵机：可数
  - 输入：可数
  - 对角线法：不存在通用算法判定停机

BNF范式

- <if语句> ::= if (<表达式>) 语句
- 产生式：$A \rightarrow iES$
- 大写字母表示非终结符
- 小写字母表示终结符
- 同一个非终结符的多个产生式用|表示或
- $A \rightarrow iES|iESeS$

λ演算

e ::= x　　变量

　| λx. e　　抽象

　| e_1 e_2　　应用

x	$(\lambda x.x)y = y$	相当于f(x)=x，求f(y)
xy	$(\lambda x.xy)z = zy$	f(x)=xy，f(z)=zy
$\lambda x.x$	$(\lambda x.xx)(\lambda y.y) = (\lambda y.y)(\lambda y.y) = (\lambda y.y)$	
$\lambda y.\lambda x.xy$	$(\lambda x.xx)(\lambda x.xx) = (\lambda x.xx)(\lambda x.xx)$	

applied λ演算

- 增加其他算符、类型
- 加法运算和数字
  - $(\lambda x. x + 6)2 = 2 + 6 = 8$
  - $(\lambda x. x\ 100)(\lambda y. y + 1) = (\lambda y. y + 1)100 = 100 + 1 = 1$
- 布尔运算
- 条件语句

上下文无关文法

- 四元组 $G = (V_T, V_N, S, P)$
- **终结符 A finite terminal vocabulary** V_T
  - 不可再分
  - 如：基本字、标识符、常数、算符和界符等
- **非终结符 A finite set of nonterminal vocabulary** V_N
  - 代表语法范畴，也称语法变量，一定符号串的集合
  - 如：表达式、赋值句、分程序、过程等

上下文无关文法

- 四元组**G = (V_T, V_N, S, P)**
- **开始符号 A start symbol S ∈ V_N that starts all derivations**
  - 特殊的非终结符
- **产生式 P, a finite set of productions (rewriting rules) of the form P →α | β**
  - 左部 P ∈ V_N，右部 α, β ∈Σ^*

Σ^*表示Σ上的所有可能符号串

$\Sigma = V_T \cup V_N$

符号记号

- 字母表∑
  - 符号的有限集
  - ∑={a, 0, 1}
  - 空集Φ
- 符号串
  - 符号的有限序列
  - 空串ε
- 符号串连接操作
  - x=0, y=1, xy=01
  - $a^0=\varepsilon$, $a^1=a$, $a^2=aa$
- 集合乘积
  - XY={xy|x∈X且y∈Y}
  - A={a, b, c, d}, B={0, 1}, AB=?

例子：自然数集合

- 1位自然数：$N=\{0, 1\}$
- 2位自然数：$M=\{00, 01, 10, 11\}=\{0, 1\}\ \{0, 1\}=N\ N$
- 3位自然数 $L = N\ M$ 或者 $L = M\ N$
- 集合的幂 $N^0=\{\varepsilon\}$, $N^1=N$, $N^k=N^{k-1}N=NN^{k-1}$ 递归定义
- 正闭包$N^+=N\cup N^2\cup N^3\cup\ldots\ldots$
- 克林闭包$N^*=N^0\cup N^+$

上下文无关文法

- 它所定义的语法范畴（或语法单位）完全独立于这种范畴可能出现的环境之外
- 不宜描述自然语言
  - 自然语言中，句子和词等往往与上下文紧密相关
- 四个组成部分
  - 一组终结符号，一组非终结符号
  - 一个开始符号，一组产生式

乔姆斯基文法

- 上下文无关文法的一般化
- 定义清晰自洽
  - 避免出现罗素悖论
- 语言描述能力强
  - 能描述大多数程序设计语言
- 递归定义

乔姆斯基文法

对任一产生式α→β

- 0型 短语文法
  - 递归可枚举

$\alpha \in (V_N \cup V_T)^*$且至少含有一个非终结符
$\beta \in (V_N \cup V_T)^*$

- 1型 上下文有关文法
  - 产生式左边可以有多个符号

除$S \to \varepsilon$外,对任一产生式$\alpha \to \beta$都有$|\alpha| \le |\beta|$
S不得出现在任何产生式的右部

- 2型 上下文无关文法
  - 大多数程序语言的语法

$\alpha \in V_N$
$\beta \in (V_N \cup V_T)^*$

- 3型 正则/正规文法
  - 有限自动机可以识别

$A \to aB$或$A \to a$
$A \in V_N$
$B \in V_N$
$a \in V_T^*$

随着对产生式的约束条件逐渐增强，文法描述语言的能力逐渐减弱