编程语言通识

语言分类

自然语言

> 自然地随文化演化的语言，例如汉语、英语、日语等...

形式语言

> 用精确的数学或机器可处理的公式定义的语言，是为了特定应用而人为设计的语言。例如数学家用的数字和运算符号、化学家用的分子式等...

乔姆斯基谱系

> 乔姆斯基谱系是计算机科学中刻画形式文法表达能力的一个分类谱系，它是由诺姆·乔姆斯基于1956年提出的。

文法分类

乔姆斯基谱系将形式文法分为以下**四类**：

Ⅰ. **0-型文法**（无限制文法或短语结构文法）产生可被图灵机识别的语言。

规则：设G = (VN, VT, P, S)，如果它的每个产生式α → β是这样一种结构：α ∈ (VN ∪ VT)*且至少含有一个非终结符，而β ∈ (VN ∪ VT)*，则G是一个0型文法。

自动机：图灵机

Ⅱ. **1-型文法**（上下文相关文法）产生上下文相关语言。

规则：在0型文法的基础上，每一个α → β都满足：

i. |β| >= |α|（这里的|α|和|β|表示的是α和β的长度）

特例：α→ε也满足1型文法

自动机：线性有界自动机

Ⅲ. **2-型文法**（上下文无关文法）产生上下文无关语言。

规则：在1型文法的基础上，每一个α → β都满足：

i. |α| = 1，且α是非终结符

ii. |β|长度有限

自动机：下推自动机

Ⅳ. **3-型文法**（正规文法）产生正规语言。

规则：在2型文法的基础上，每一个α → β都满足：

i. 1 <= |β| <= 2，且|β| = 1时β为终结符**或**非终结符，|β| = 2时β为终结符**和**非终结符。

ii. β的线性递归必须保持一致，都为左或者都为右

自动机：有限状态自动机

示例

🤔 如何判断属于哪一类型文法呢?

💡 其实从上述规则中可以看出：0、1、2、3型文法是具有包含关系的。所以我们一般从3-型文法开始逐步往上判断其类型。

举例：待补充...

巴克斯范式

> Backus-Naur Form，简称BNF。是一种形式化的语法表示方法，是用来描述**上下文无关语法**的一种形式体系，它是一种典型的元语言。

基础概念

终结符：不可再分的字符或串

非终结符：由终结符和至少一个非终结符组成的串，用尖括号<>括起

语法规则

- ::= "被定义为"
- | "或"
- + "重复一次或多次"
- * "重复0或多次"
- () "分组"
- <> 包含的是必选项
- [] 包含的是可选项
- {} 包含的是可重复0至无数字的项
- 双引号内的字代表字符本身
- 双引号外的字代表语法部分

示例

```
<Int> ::= 0 | ((1|2|3|4|5|6|7|8|9)<Int>*)    // 整数
/**
*    四则运算
*/
```

```
<Expression> ::= <AdditiveExpression>
<AdditiveExpression> ::=
                <AdditiveExpression> "+" <MultipleExpression> |
                <AdditiveExpression> "-" <MultipleExpression> |
                <MultipleExpression>
<MultipleExpression> ::=
                <MultipleExpression> "×" <PrimaryExpression> |
                <MultipleExpression> "/" <PrimaryExpression> |
                <PrimaryExpression>
<PrimaryExpression> ::= <Int> | "("<Expression>")"
```

扩展

EBNF、ABNF
待补充...

图灵问题

图灵机

> 图灵机是图灵在1936年发表的《论可计算数及其在判定性问题上的应用》（《On Computable Numbers, with an Application to the Entscheidungsproblem》）中提出的数学模型。

图灵完备

> 简单来说，**能够抽象成图灵机**的系统或编程语言就是图灵完备的；一切**可计算的问题**图灵机都能计算，因此满足这样要求的逻辑系统、装置或者编程语言就叫图灵完备的。

一般来说，一门编程语言实现了goto、if、while或是lambda表达式，我们就会判定这门编程语言是图灵完备的。

类型系统

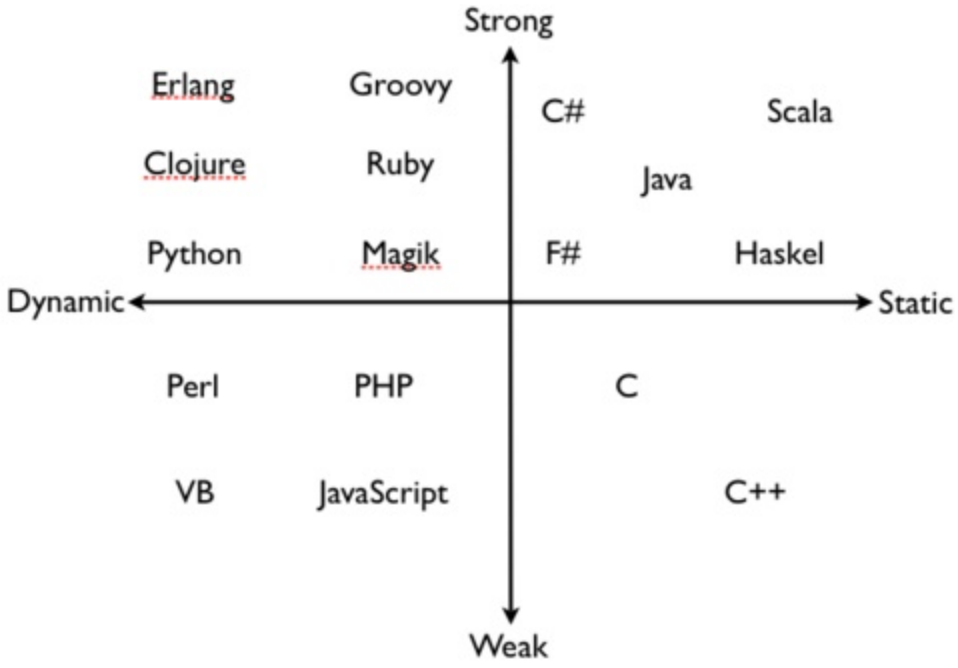
动态类型和静态类型

> **动态类型语言**是指数据类型的检查是在**运行时**做的。用动态类型语言编程时，不用给变量指定数据类型，该语言会在你第一次赋值给变量时，在内部记录数据类型。动态类型语言中，变量没有类型，而值有类型。
> **静态类型语言**是指数据类型的检查是在**运行前**（如编译阶段）做的。

强类型和弱类型

> **强类型**：当定义一个变量是某个类型，如果该变量不经过显式转换，它就永远是那个类型，如果把它当做其他类型来用，就会报错。
> **弱类型**：你想把这个变量当做什么类型来用，就当做什么类型来用，语言的解析器会自动进行隐式转换。

主要通过是否存在**隐式类型转换**去判断强弱类型。



复合类型

> 复合类型（compound type）是指基于其他类型定义的类型

JS中的复合类型

Object、Array、Function

子类型

逆变

> 如果某个参数类型可以由其基类替换，那么这个类型就是支持**逆变**的。

协变

> 如果某个返回的类型可以由其派生类型替换，那么这个类型就是支持**协变**的。