DCS290

Compilation Principle

编 译 原 理

第二章：语言和文法基础

郑馥丹
zhengfd5@mail.sysu.edu.cn

CONTENTS

目 录

1. 程序设计语言

- 程序设计语言包括：语法和语义
  - 语法(syntax)：是一组**规则**，用它可以形成和产生一个合适的程序
  - 语义(semantics)：定义程序的**意义**
    - ✓ 静态语义：程序在语义上要遵守的规则
      - **数组下标越界**
      - **声明和使用的函数没有定义**
      - **零作除数**
      - ……
    - ✓ 动态语义：表明程序要做什么

2. 文法[Grammar]的直观概念

- 如何来描述一种语言?
  - 如果语言是有穷的（只含有有穷多个句子）：可以将句子逐一列出来表示
  - 如果语言是无穷的，要找出语言的有穷表示
  - **文法[Grammar]**：
    - ✓ 是语言**语法**的描述工具，实现**用有穷的规则把语言的无穷句子集描述出来**
    - ✓ 严格定义句子的结构，是判断句子结构合法与否的依据
  - 例：“**我是大学生**”是汉语的一个句子

汉语句子的部分构成规则可表示为：

- <句子>::=<主语><谓语>
- <主语>::=<代词> | <名词>
- <代词>::= 我 | 你 | 他
- <名词>::= 王明 | 大学生 | 工人 | 英语
- <谓语>::=<动词><直接宾语>
- <动词>::= 是 | 学习
- <直接宾语>::=<代词> | <名词>

2. 文法[Grammar]的直观概念

- 由规则推导句子
  - 方法: 用一条规则，用**::=右端的符号串代替::=的左端**
    - ✓ **<句子>::=<主语><谓语>**
  - 表示: 用“ ⇒ ”表示推导，其含义是，使用一条规则，代替掉⇒左边的某个符号，产生⇒右端的符号串
  - 例如：句子“我是大学生”的推导过程如下：

<句子>

⇒ <主语><谓语>

⇒ <代词><谓语>

⇒ 我<谓语>

⇒ 我<动词><直接宾语>

⇒ 我是<直接宾语>

⇒ 我是<名词>

⇒ 我是大学生

汉语句子的部分构成规则可表示为:

- <句子>::=<主语><谓语>
- <主语>::=<代词> | <名词>
- <代词>::= 我 | 你 | 他
- <名词>::= 王明 | 大学生 | 工人 | 英语
- <谓语>::=<动词><直接宾语>
- <动词>::= 是 | 学习
- <直接宾语>::=<代词> | <名词>

CONTENTS

目 录

1. 字母表[Alphabet]（符号集[a set of symbols]）

- 定义：字母表是元素的非空有穷集合

  例：∑={0,1} A={a,b,c}

- 元素也称为符号，字母表也称符号集
- 不同的语言有不同的字母表
- 程序语言的字母表由**字母、数字和若干专用符号**组成

2. 符号串[String]

- 定义：符号串是由字母表中的符号组成的任何有穷序列

  例： 0,00,10,011是字母表∑={0, 1}上的符号串

  a,ab,aaca是A={a,b,c}上的符号串

- 在符号串中，符号是有顺序的，顺序不同，代表不同的符号串 **例: ab和ba不同**
- 不含任何符号的符号串称为空串，用ε表示

  注意：{ε}并不等于空集合{ } —— Φ

- 符号串长度：是符号串中含有符号的个数

  例：|abc|=3　　|ε|=0

- 符号串的头尾，固有头和固有尾：如果z=xy是一符号串，则x是z的头，y是z的尾。如果x是非空的，则y是固有尾；如果y非空，则x是固有头。

  例：z=abc，z的头：ε，a，ab，abc，除abc外，其他都是固有头；z的尾：ε，c，bc，abc，除abc外，其他的都是固有尾。

头、尾 ↔ 前缀、后缀
固有头、固有尾 ↔ 真前缀、真后缀

3．符号串的运算

- **符号串的连接**：设x 、 y是符号串，它们的连接是把y的符号写在x的符号之后得到的符号串xy

例如 x="ST"， y="abu" ， 则 xy="STabu"

显然 $\varepsilon x = x\varepsilon = x$

- **符号串的方幂**：把符号串a自身连接n次得到的符号串$a^n = aa...aa$

例1 $a^0=\varepsilon$, $a^1=a$, $a^2=aa$

例2 $x=AB$, $x^0 = \varepsilon$, $x^1 = AB$, $x^2 = ABAB...$

4. 符号串集合

- 定义：若集合A中所有元素都是某字母表Σ上的符号串，则称A为字母表Σ上的符号串集合。
- 符号串集合的乘积：符号串集合A和B的乘积定义为：
  - AB ={xy|x∈A且y∈B}，即AB是由A中的串x和B中的串y连接而成的所有串xy组成的集合。

例：若集合A = {ab,cde}　B = {0,1}

则 AB = {ab0,ab1,cde0,cde1}

显然 {ε}A = A{ε} = A

4. 符号串集合

- 符号串集合的方幂： 设A是符号串的集合，则称A^i为符号串集A的方幂，其中i是非负整数。具体定义如下:
  - A^0 ={ε}, A^1 = A, A^2 = AA, A^K = AA......A(k个)

例：

若集合A={a, b}　则A^0={ε}　　A^1=A={a, b}

A^2=AA={a, b}{a, b}={aa, ab, ba, bb}

A^3=AAA={a, b}{a, b}{a, b}={aaa, aab, aba, abb, baa, bab, bba, bbb}

5. 集合的闭包[Closure]

- 闭包[Kleene Closure]
  - 集合Σ的闭包Σ*定义如下： $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$

    例：设有字母表Σ={0, 1}

    则$\Sigma^*=\Sigma^0\cup\Sigma^1\cup\Sigma^2\cup\ldots$

    ={ε,0,1,00,01,10,11,000,…}

    即Σ*表示Σ上所有有穷长的串的集合。
- 正闭包[Positive Closure]
  - $\Sigma^+ = \Sigma^1\cup\Sigma^2\cup\Sigma^3\cup\ldots$称为Σ的正闭包。
  - Σ^+ 表示Σ上的除ε外的所有用穷长串的集合

    $\Sigma^* = \Sigma^0\cup\Sigma^+$

    $\Sigma^+ = \Sigma\Sigma^* = \Sigma^*\ \Sigma$

6. 语言[Language]

- 例如：$\Sigma=\{a,b\}$ $\Sigma^*=\{\varepsilon,a,b,aa,ab,ba,bb,aaa,aab,\ldots\}$

1. 集合$\{ab,aabb,aaabbb,\ldots,a^nb^n,\ldots\}$或$\{w|w\in\Sigma^*$且$w=a^nb^n,n\geq1\}$为字母表Σ上的一个语言。
2. 集合$\{a,aa,aaa,\ldots\}$或$\{w|w\in\Sigma^*$且$w=a^n,n\geq1\}$为字母表Σ上的一个语言。
3. $\{\varepsilon\}$是一个语言。
4. Φ即{ }是一个语言。
5. ……

Σ*上任意字符串的集合均是该字母表Σ上的语言

CONTENTS

目 录

1. 文法的定义

- 文法定义：
  - 文法G定义为**四元组(V_N，V_T，P，S)**
    - ✓ V_N (Nonternimal)：非终结符集
    - ✓ V_T (Terminal)：终结符集
    - ✓ P (Production)：**产生式（规则）**集合
    - ✓ S (Start)：开始符号或识别符号

- V_N，V_T 和 P 是非空有穷集
- $V=V_N \cup V_T$，V称为文法G的字母表
- $V_N \cap V_T = \Phi$
- S是一个非终结符，且至少要在一条产生式的左部出现

- 产生式（规则）：
  - 产生式是一个有序对(α,β)，通常写作α→β（或α::=β）（读作：α定义为β）

〈句子〉::=〈主语〉〈谓语〉

P中产生式形如：α→β，其中：$\alpha \in V^+$且至少含一个非终结符，$\beta \in V^*$

1. 文法的定义

- 例1：文法$G=(V_N,V_T,P,S)$，其中：
  - V_N ={句子，主语，代词，名词，谓语，动词，直接宾语}
  - V_T ={我，你，他，王明，大学生，工人，英语，是，学习}
  - P = {

    <句子> → <主语><谓语>

    <主语> → <代词> | <名词>

    <代词> → 我 | 你 | 他

    <名词> → 王明 | 大学生 | 工人 | 英语

    <谓语> → <动词><直接宾语>

    <动词> → 是 | 学习

    <直接宾语> → <代词> | <名词>

    }
  - S =句子

➢ <句子>::=<主语><谓语>
➢ <主语>::=<代词> | <名词>
➢ <代词>::= 我 | 你 | 他
➢ <名词>::= 王明 | 大学生 | 工人 | 英语
➢ <谓语>::=<动词><直接宾语>
➢ <动词>::= 是 | 学习
➢ <直接宾语>::=<代词> | <名词>

1. 文法的定义

- 例2：文法$G=(V_N,V_T,P,S)$，其中：
  - V_N ={S}， V_T ={0, 1}，
  - P={S→0S1,S→01}，开始符为S

- 例3：文法$G=(V_N,V_T,P,S)$，其中：
  - V_N ={标识符，字母，数字}，
  - V_T ={a,b,c,…,x,y,z,0,1,…,9}，
  - P={

    <标识符>→<字母>，<标识符>→<标识符><字母>

    <标识符>→<标识符><数字>，

    <字母>→a,…, <字母>→z，

    <数字>→0,…,<数字>→9 }，
  - S=<标识符>

随堂练习(1)

- 为只包含数字、加号和减号的表达式，例如9-2+5,3-1,7等构造一个文法

随堂练习(1)

- 为只包含数字、加号和减号的表达式，例如9-2+5,3-1,7等构造一个文法
  - G[式子]:
    - ✓ <式子> →<数字>|<式子><运算符><式子>
    - ✓ <数字> → 0|1|2|3|4…|9
    - ✓ <数字> → <数字><数字>
    - ✓ <运算符> → +|-

2. 文法的简化表示法

- 简化：通常不用将文法的四元组表示出来，**只写出产生式**
- 约定:
  - 默认第一条产生式的左部的符号是开始符号，或用**G[S]**表示**S**是**开始符号**；
  - 用**大写字母**（或用尖括号括起来）表示**非终结符**；
  - 用**小写字母**表示**终结符**；
  - 左部相同的产生式**A→α, A→β**可以记为；**A→α|β**，其中“|”表示“或”
- 例如:

文法G[S]:

S→A|SA|SD

A→a|b|…|z

D→0|1|…|9

V_N={S, A, D}
V_T={a,b,…,z,0,1,…,9}
P ={
S→A|SA|SD
A→a|b|…|z
D→0|1|…|9 }
S——开始符号

3. 推导[Derivation]与归约[Reduction]

(1) 直接推导和直接归约

– α→β是文法G的产生式，若有v，w满足：v=γ**α**δ,w= γ**β**δ，其中γ,δ∈V*，则称：
  - ✓ v**直接推导**到w,
  - ✓ 或称： w**直接归约**到v,
  - ✓ 记作： v ⇒ w

– 直接推导：用产生式的右部替换掉产生式的左部

– 直接归约：用产生式的左部替换掉产生式的右部

例 文法G： S→0S1，S→01 有直接推导:

0S1	**⇒00S11**	(S→0S1)
0S1	**⇒0011**	(S→01)
00S11	**⇒000S111**	(S→0S1)
000S111	**⇒00001111**	(S→01)
S	**⇒0S1**	(S→0S1)

3. 推导[Derivation]与归约[Reduction]

(2) 推导和归约

- 若存在$v=w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n=w,(n>0)$（即v经过**多步**推到到w），则称:
  - ✓ v推导出w,
  - ✓ 或称：w归约到v,
  - ✓ 记为：**$v \stackrel{+}{\Rightarrow} w$**
- 若有$v \stackrel{+}{\Rightarrow} w$，或$v=w$，则记作**$v \stackrel{*}{\Rightarrow} w$**

$\alpha \Rightarrow 0\beta,\ \beta \Rightarrow 2\gamma$

$\alpha \Rightarrow 0\beta \Rightarrow 02\gamma$

$\alpha \stackrel{+}{\Rightarrow} 02\gamma$

$\alpha \stackrel{*}{\Rightarrow} 02\gamma$

例 文法G： S→0S1， S→01

$$\underline{S} \Rightarrow 0\underline{S}1 \Rightarrow 00\underline{S}11 \Rightarrow 000\underline{S}111 \Rightarrow 00001111$$

$$S \stackrel{+}{\Rightarrow} 00001111$$

$$S \stackrel{*}{\Rightarrow} 00001111$$

4. 句型、句子、语言的定义

（1）句型和句子

– 句型：由文法**开始符号S推导出**的符号串α（即S＝*＞α），称为文法G[S]的**句型**。

– 句子：**仅由终结符组成**的句型α（即S＝*＞α，α∈V_T*），称为文法G[S]的**句子**。

例 文法G： S→0S1， S→01

$\underline{S} \Rightarrow 0\underline{S}1 \Rightarrow 00\underline{S}11 \Rightarrow 000\underline{S}111 \Rightarrow 00001111$

则： 句型 句型 句型 句型 句型

句子

4. 句型、句子、语言的定义

（2）语言的定义

– 语言：文法G[S]的一切**句子的集合**称为语言，记做L(G)

例 文法G：S→0S1，S→01

$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 0^3S1^3 \Rightarrow \ldots\ldots \Rightarrow 0^{n-1}S1^{n-1} \Rightarrow 0^n1^n$

$L(G)=\{0^n1^n|n\geq1\}$

4. 句型、句子、语言的定义

（2）语言的定义

$G=(V_N,V_T,P,S)$, $V_N=\{S,B,E\}$, $V_T=\{a,b,e\}$

(1) $S \to aSBE$	使用(1) n-1次，得到$S \overset{*}{\Rightarrow} a^{n-1}S(BE)^{n-1}$
(2) $S \to aBE$	使用(2) 1次，得到$S \overset{*}{\Rightarrow} a^n(BE)^n$
(3) $EB \to BE$	使用(3) 若干次，得到$S \overset{*}{\Rightarrow} a^nB^nE^n$
(4) $aB \to ab$	使用(4) 1次，得到$S \overset{*}{\Rightarrow} a^nbB^{n-1}E^n$
(5) $bB \to bb$	使用(5) n-1次，得到$S \overset{*}{\Rightarrow} a^nb^nE^n$
(6) $bE \to be$	使用(6) 1次，得到$S \overset{*}{\Rightarrow} a^nb^neE^{n-1}$
(7) $eE \to ee$	使用(7) n-1次，得到$S \overset{*}{\Rightarrow} a^nb^ne^n$

所以G产生的语言$L(G)=\{a^nb^ne^n|n\geq 1\}$

5. 文法的等价

- 若L(G1)=L(G2)，即，若两个文法所定义的语言是一样的，则称文法G1和G2是等价的。

例如：文法　G1[A]：A→0R A→01 R→A1

G2[S]：S→0S1 S→01

所定义的语言都是0^n1^n

因此，两文法等价。