

第八讲

CFG的应用与文法的二义性

2022/4/12

School of Software

1

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

2

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

3

CFG的应用

- Regular Expressions
 - Take a single instance of a string:
10/Apr/2021
 - Use regex to generalize the pattern:
 (..)/(...)/(...) → AB/789/A1B2
 (..)/(..+)/(..+) → ABCD/432/00
 (\d+)/([a-zA-Z]+)/(\d+) → 10/Apr/2021

Use parentheses to specify fields to extract.

2022/4/12

School of Software

4

CFG的应用

- Regular Expressions: Meta Characters

	Description	Example	Match	No Match
.	Any character except \n	...	abc	abn abcd
[]	Any character inside brackets	[cb.]ar	bar ,ar	jar
*	>=0 or more of last symbol	[pb]*ark	bbpark ark	dark
+	>=1 last symbol	[pb]+ark	bbpark bark	dark ark
?	Last symbol optional	s?he	she he	the
	Before or after	we us	we us	e s

2022/4/12

School of Software

5

CFG的应用

- Regular Expressions: Meta Characters

	Description	Example	Match	No Match
[]	Any character inside brackets	[A-Z.]ar	Bar .ar	jar
[^]	Any character not inside brackets	[^a-z]ar	Bar :ar	car
\	Escapes next character	\[hi\]	[hi]	hi
^	Beginning of line	^ark	ark two	dark
\$	End of line	ark\$	Noahs ark	arkk

2022/4/12

School of Software

6

CFG的应用

Regular Expressions

Shorthand Characters

Bracket Form	Shorthand	Description
[a-zA-Z0-9_]	\w	Alphanumeric character
[^\w]	\W	Not an alphanumeric char
[0-9]	\d	Digit
[^\d]	\D	Not a digit
[\t\n\r\f\p{Z}]	\s	Whitespace
[^\s]	\S	Not whitespace

2022/4/12

School of Software

7

CFG的应用

A BNF grammar of sentences in PL

```

S := <Sentence> ;
<Sentence> := <AtomicSentence> | <ComplexSentence> ;
<AtomicSentence> := 'TRUE' | 'FALSE' | 'P' | 'Q' | 'S' ;
<ComplexSentence> := '(' <Sentence> ')' |
    <Sentence> <Connective> <Sentence> |
    'NOT' <Sentence> ;
<Connective> := '^' | 'v' | '→' | '↔' ;

```

Grammar of sentences in PL is a CFG

2022/4/12

School of Software

8

CFG的应用

A BNF for FOL

```

S := <Sentence> ;
<Sentence> := <AtomicSentence> | <Sentence> <Connective> <Sentence> |
<Quantifier> <Variable> , ... <Sentence> | '(' <Sentence> ')' ;
<AtomicSentence> := <Predicate> '(' <Term> , ... ')' ;
<Term> := <Function> '(' <Term> , ... ')' | <Constant> | <Variable> ;
<Connective> := '^' | 'v' | '→' | '↔' ;
<Quantifier> := '∃' | '∀' ;
<Constant> := 'A' | 'XI' | 'John' | ... ;
<Variable> := 'a' | 'x' | 's' | ... ;
<Predicate> := 'Before' | 'HasColor' | '=' | ... ;
<Function> := 'Mother' | 'LeftLegOf' | ... ;

```

Grammar of FOL is also a CFG

2022/4/12

School of Software

9

CFG的应用

Syntax of LTL

- $\phi ::= p \mid \neg \phi \mid \phi \wedge \phi \mid (\phi)$
 $\mid \phi U \phi \mid X \phi \mid F \phi \mid G \phi \mid \phi W \phi \mid \phi R \phi$
- Basic temporal operators:
 X (O, "next time")
 U ("until")
 F (\Diamond , "sometime", "eventually")
 G (\Box , "always", "henceforth")
 W Weak-until
 R Release

It is easy to show that the Syntax of LTL is also the CFG

2022/4/12

School of Software

10

上下文无关文法的应用

- 程序设计语言语法描述和分析
Yacc (yet another compiler's compiler)
- 文档格式描述和处理
DTD (Document Type Definition)

2022/4/12

School of Software

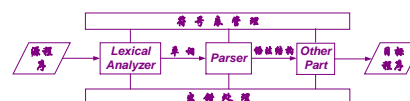
11

上下文无关文法的应用

- 编译/解释程序 (Compiler / Interpreter)



- 词法分析器 (Lexical Analyzer)与语法解析器(Parser)



2022/4/12

School of Software

12

Yacc 源程序式样

声明节

定义文法中使用的单词 (利用 %token 命令) 以及一些 C 代码相关的声明;

%%

语法规则节

定义语法规则及语义动作. Yacc中的产生式格式为非终结符: 符号串 {C语言表示的语义动作};

%%

支撑函数节

规则节用到的局部 C 函数定义

2022/4/12

School of Software

13

Yacc 语法规则描述

$$\begin{aligned} E &\rightarrow EOE \\ E &\rightarrow (E) \\ E &\rightarrow v \\ E &\rightarrow d \\ O &\rightarrow + \\ O &\rightarrow * \end{aligned}$$


```
%token VARIABLE DATA
%%
Exp: Exp Op Exp {...}
    '(' Exp ')' {...}
    'VARIABLE' {...}
    'DATA' {...}

Op: '+' {...}
    '*' {...}

%%
yylex() {...}
```

2022/4/12

School of Software

14

标记语言

- 标记语言 (markup languages)
 - HTML (HyperText Markup Language)

如 <P>The things I hate:

```
<OL>
<LI>Moldy bread
<LI>People who drive too slow in the fast lane
</OL>
```

The things I hate:

- Moldy bread
- People who drive too slow in the fast lane

2022/4/12

School of Software

15

标记语言

Case Study: Transcript

```
# You should be familiar enough to get what you want from the text
tables = page.find_all('table')
courses = tables[2]

header_row, 'data_rows, _' = coursestbody.find_all('tr', recursive=False)
headers = [cell.div.text for cell in header_row.find_all('th')]
print(headers)

def parseCell(cell):
    if cell.div:
        cell = cell.div
        if cell.script:
            return cell.contents[0].strip()
        else:
            return cell.text.strip()
    return cell.text.strip()

data = [[parseCell(cell) for cell in row.find_all('td')] for row in data_rows]
print(data[0])
print(data[1])
```



```
[['课程号', '课程名', '学分', '学时', '成绩', '绩点', '替代课程', '课程属性', '特殊课程标志', '学年学期', '考试时间'],
 ['12090043', '4', '军事理论与技能训练', '2', '32', 'B+', '3.6', '-', '必修', '-', '2015-2016-3', '20160820'],
 ['02070012', '91', '党的知识概论', '2', '32', 'P', 'N/A', '-', '任选', '-', '2016-2017-1', '20170115']]
```

2022/4/12

School of Software

16

标记语言

Convert HTML into DataFrame

```
import pandas as pd
# convert to pandas DataFrame
df = pd.DataFrame(columns=headers, data=data)
# check some rows
df.head()
```

课程号	课程名	学分	学时	成绩	绩点	替代课程	课程属性	特殊课程标志	学年学期	考试时间
0 12090043	军事理论与技能训练	4	32	B+	3.6		必修		2015-2016-3	20160820
1 02070012	党的知识概论	2	32	P	N/A		任选		2016-2017-1	20170115
2 10421055	微积分A(1)	5	80	A-	4.0		必修		2016-2017-1	20170115
3 10421094	线性代数(1)	4	78	A	4.0		必修		2016-2017-1	20170115
4 10450012	现代生物学导论	2	32	B+	3.6		限选		2016-2017-1	20170115

2022/4/12

School of Software

17

标记语言

- 标记语言 (markup languages)
 - HTML (HyperText Markup Language)
 - SGML (Standard Generalized Markup Language)
 - ISO 8879 国际标准
 - XML (eXtensible Markup Language)
 - SGML 的一个子集, 用户通过 DTD (Document Type Definition) 可以自定义 tag 的类型.

2022/4/12

School of Software

18

标记语言

JavaScript Object Notation (JSON)

```
1 {
2   "glossary": {
3     "title": "example glossary",
4     "id": "Glossary",
5     "longdescript": "A long description of the glossary.",
6     "shortdescript": "A short description of the glossary.",
7     "date": "2003-10-10",
8     "author": "John Doe",
9     "editor": "Jane Doe",
10    "url": "http://www.example.com/glossary.html",
11    "version": "1.0",
12    "copyright": "Copyright 2003, Example Corp.",
13    "license": "All rights reserved.",
14    "terms": "All rights reserved.",
15    "notes": "This glossary is for internal use only.",
16    "examples": [
17      {
18        "id": "example1",
19        "text": "An example of a term."
20      },
21      {
22        "id": "example2",
23        "text": "Another example of a term."
24      }
25    ]
26  }
27 }
```

```
import json
with open('std.json', 'r') as f:
    data = json.load(f)
```

- Recursive datatype
- Data inside of data
- Value is a:
 - A basic type:
 - String
 - Number
 - True/False
 - Null
 - Array of Values
 - A dictionary of Key: Value pairs

2022/4/12

School of Software

19

标记语言

- XML is a standard for hierarchical representations of data

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
2 <glossary title="example glossary" id="Glossary">
3   <longdescript>A long description of the glossary.</longdescript>
4   <shortdescript>A short description of the glossary.</shortdescript>
5   <date>2003-10-10</date>
6   <author>John Doe</author>
7   <editor>Jane Doe</editor>
8   <url>http://www.example.com/glossary.html</url>
9   <version>1.0</version>
10  <copyright>Copyright 2003, Example Corp.</copyright>
11  <license>All rights reserved.</license>
12  <terms>All rights reserved.</terms>
13  <notes>This glossary is for internal use only.</notes>
14  <examples>
15    <example id="example1">An example of a term.</example>
16    <example id="example2">Another example of a term.</example>
17  </examples>
18 </glossary>
```

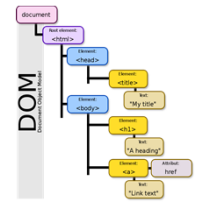
```
from bs4 import BeautifulSoup
file = open('std.xml', encoding='utf-8').read()
soup = BeautifulSoup(file, 'xml')
```

2022/4/12

School of Software

20

Handle XML file with bs4 as you wish!



标记语言和文档类型定义

• DTD 的格式

```
<!DOCTYPE name-of-DTD
[ list of element definitions ]>
<!ELEMENT element-name
(description of the element)>
```

description of the elements 实质上是一个正则表示:

- 基本表示式可以是其它 *element-name*;
- 可以是一个代表任意文本的特殊项 *#PCDATA*.
- 正则表示的算符记号类似于 *UNIX* 正则表示式.

2022/4/12

School of Software

21

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

22

DTD 格式

```
<!DOCTYPE name-of-DTD
[ list of element definitions ]>
```

```
<!ELEMENT element-name
(description of the element)>
```

2022/4/12

School of Software

23

DTD 例

```
<!DOCTYPE PcSpecs [
  <!ELEMENT PCS (PC*)>
  <!ELEMENT PC (MODEL, PRICE, PROCESSOR, RAM, DISK+)>
  <!ELEMENT MODEL (#PCDATA)>
  <!ELEMENT PRICE (#PCDATA)>
  <!ELEMENT PROCESSOR (MANF, MODEL, SPEED)>
  <!ELEMENT MANF (#PCDATA)>
  <!ELEMENT SPEED (#PCDATA)>
  <!ELEMENT RAM (#PCDATA)>
  <!ELEMENT DISK (HARDDISK | CD | DVD)>
  <!ELEMENT HARDDISK (MANF, MODEL, SIZE)>
  <!ELEMENT SIZE (#PCDATA)>
  <!ELEMENT CD (SPEED)>
  <!ELEMENT DVD (SPEED)>
]>
```

2022/4/12

School of Software

24

DTD 例

```
<!DOCTYPE PcSpec [
  <ELEMENT PC (MODEL, PRICE, PROCESSOR, RAM,
    DISK+)>
  <ELEMENT MODEL (#PCDATA)>
  <ELEMENT PRICE (#PCDATA)>
  <ELEMENT PROCESSOR (MANF, MODEL, SPEED)>
  <ELEMENT MANF (#PCDATA)>
  <ELEMENT SPEED (#PCDATA)>
  <ELEMENT RAM (#PCDATA)>
  <ELEMENT DISK (HARDDISK | CD | DVD)>
  <ELEMENT HARDDISK (MANF,MODEL,SIZE)>
  <ELEMENT SIZE (#PCDATA)>
  <ELEMENT CD (SPEED)>
  <ELEMENT DVD (SPEED)>
]>
```

```
<PC>
<PC>
  <MODEL>4560</MODEL>
  <PRICE>$2295</PRICE>
  <PROCESSOR>
    <MANF>Intel</MANF>
    <MODEL>Pentium</MODEL>
    <SPEED>800MHz</SPEED>
  <PROCESSOR>
    <MANF>Maxtor</MANF>
    <MODEL>Diamond</MODEL>
    <SIZE>30.5Gb</SIZE>
  <HARDDISK></DISK>
  <DISK><CD>
    <SPEED>32x</SPEED>
  <CD></DISK>
</PC> <PC> ... </PC>
</PCS>
```

2022/4/12

School of Software

25

HTML部分语法

Part of an HTML Grammar

Char $\rightarrow a \mid A \mid \dots$ Text $\rightarrow \epsilon \mid \text{Char Text}$ Doc $\rightarrow \epsilon \mid \text{Element Doc}$

Element $\rightarrow \text{Text} \mid \text{ Doc } \mid$
 $\text{<P> Doc} \mid \text{ List } \mid \dots$

ListItem $\rightarrow \text{ Doc}$ List $\rightarrow \epsilon \mid \text{ListItem List}$

```
<P>The things I <EM>hate</EM>:
<OL>
  <LI>Moldy bread
  <LI>People who drive too slow in the fast lane
</OL>
```

2022/4/12

School of Software

26

DTD转化CFG

DTD 实际上是一个上下文无关文法

将右部含正则表示的产生式转化为一般上下文无关文法的产生式。

2022/4/12

School of Software

27

DTD转化CFG

```
<ELEMENT PROCESSOR (MANF, MODEL, SPEED)>
  Processor  $\rightarrow \text{Manf Model Speed}$ 
<ELEMENT DISK (HARDDISK | CD | DVD)>
  Disk  $\rightarrow \text{HardDisk} \mid \text{CD} \mid \text{DVD}$ 
<ELEMENT PC (MODEL, PRICE, PROCESSOR, RAM,
  DISK+)>
  PC  $\rightarrow \text{Model Price Processor Ram Disks}$ 
  Disks  $\rightarrow \text{Disk} \mid \text{Disk Disks}$ 
```

2022/4/12

School of Software

28

CFG的应用与文法的二义性

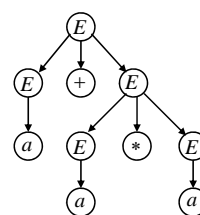
- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

29

例



(1) (2) (3) (4)
 $E \rightarrow E + E \mid E * E \mid (E) \mid a$
 $w = a + a * a$
 $E \Rightarrow^1 E + E \Rightarrow^4 a + E \Rightarrow^2 a + E * E$
 $\Rightarrow^4 a + a * E \Rightarrow^4 a + a * a$

最左推导

2022/4/12

School of Software

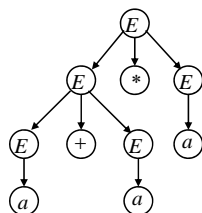
30

例

(1) (2) (3) (4)

 $E \rightarrow E + E \mid E * E \mid (E) \mid a$ $w = a + a * a$ $E \Rightarrow^2 E * E \Rightarrow^1 E + E * E \Rightarrow^4 a + E * E$ $\Rightarrow^4 a + a * E \Rightarrow^4 a + a * a$

最左推导



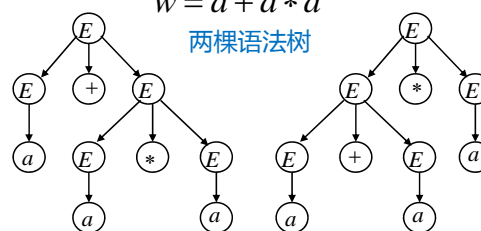
2022/4/12

School of Software

31

 $E \rightarrow E + E \mid E * E \mid (E) \mid a$ $w = a + a * a$

两棵语法树

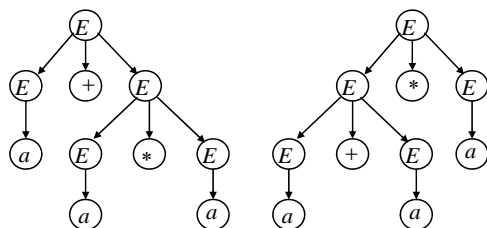


2022/4/12

School of Software

32

文法的二义性

存在字符串 $w = a + a * a$ ，有两棵不同的语法树文法： $E \rightarrow E + E \mid E * E \mid (E) \mid a$ 是二义的

2022/4/12

School of Software

33

文法的二义性

换言之：

文法G有二义性表明了：

存在某个字符串，有至少两种最左（或者最右）推导。



2022/4/12

School of Software

34

文法G： $E \rightarrow E + E \mid E * E \mid (E) \mid a$ $E \Rightarrow^1 E + E \Rightarrow^4 a + E \Rightarrow^2 a + E * E$ $\Rightarrow^4 a + a * E \Rightarrow^4 a + a * a$ $E \Rightarrow^2 E * E \Rightarrow^1 E + E * E \Rightarrow^4 a + E * E$ $\Rightarrow^4 a + a * E \Rightarrow^4 a + a * a$ 由于 $a + a * a$ 有两种不同的最左推导，则文法G是二义的。

2022/4/12

School of Software

35

文法的二义性

• 二义文法概念

CFG $G = (V, T, S, P)$ 称为二义的，如果对某个 $w \in T^*$ ，存在根结点都为开始符号 S 的两棵不同的分析树，其产物都是 w 。CFG G ，如果对每一 $w \in T^*$ ，仅存在一棵这样的分析树，则 G 为无二义的文法。

2022/4/12

School of Software

36

文法的二义性

• 定理

对 CFG $G = (V, T, S, P)$ 和 $w \in T^*$, w 具有两棵不同的分析树, 当且仅当存在两个不同的从开始符号 S 到 w 的最左推导。

• 文法二义性的另一种定义

CFG $G = (V, T, S, P)$ 称为二义的, 如果存在某个 $w \in T^*$, 有两个不同的从开始符号 S 到 w 的最左推导。

用途: 方便证明文法的无二义性。

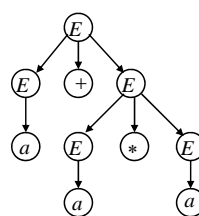
2022/4/12

School of Software

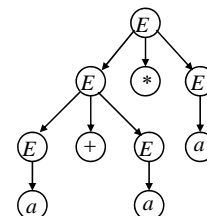
37

文法的二义性

为什么要讨论文法的二义性?


 $a + a * a$

取 $a = 2$

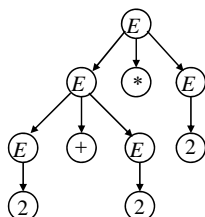
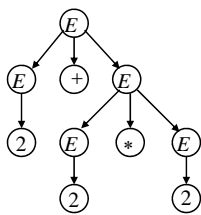


2022/4/12

School of Software

38

文法的二义性

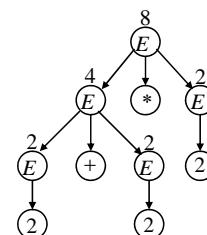
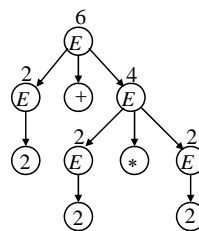
 $2 + 2 * 2$


2022/4/12

School of Software

39

文法的二义性

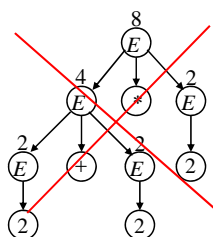
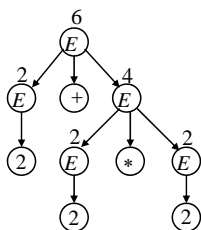
 $2 + 2 * 2 = 6$
 $2 + 2 * 2 = 8$


2022/4/12

School of Software

40

文法的二义性

 $2 + 2 * 2 = 6$ 正确结果


2022/4/12

School of Software

41

文法的二义性

• 二义性的判定

一个 CFG 是否为二义的问题是不可判定的, 即不存在解决该问题的算法。

2022/4/12

School of Software

42

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

43

二义性消除

没有通用的办法可以消除文法的二义性。对某些特定的文法，可以找到消除二义性的办法。

2022/4/12

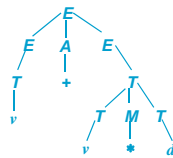
School of Software

44

二义性消除

对于右上图的文法，采用算符优先级方法将其变换为左下图的文法，对于该文法，串：
 $v + v * d$
存在唯一的分析树。

$$\begin{aligned} E &\rightarrow EAE \mid T \\ T &\rightarrow TMT \mid (E) \mid v \\ &\mid d \\ A &\rightarrow + \\ M &\rightarrow * \end{aligned}$$

$$\begin{aligned} E &\rightarrow EOE \mid (E) \mid v \mid d \\ O &\rightarrow + \mid * \end{aligned}$$


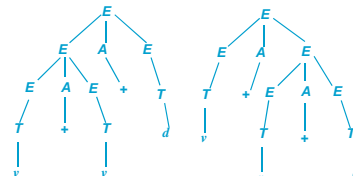
2022/4/12

School of Software

45

二义性消除

右上图的文法仍然是二义文法，串：
 $v + v + d$
存在不同的分析树(下图)。

$$\begin{aligned} E &\rightarrow EAE \mid T \\ T &\rightarrow TMT \mid (E) \mid v \mid d \\ A &\rightarrow + \\ M &\rightarrow * \end{aligned}$$


2022/4/12

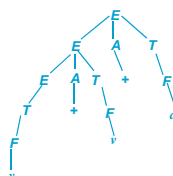
School of Software

46

二义性消除

采用左结合方法将右上图的文法变换为左下图，串：
 $v + v + d$
存在唯一的分析树（右下图）

$$\begin{aligned} E &\rightarrow EAT \mid T \\ T &\rightarrow TMF \mid F \\ F &\rightarrow (E) \mid v \mid d \\ A &\rightarrow + \\ M &\rightarrow * \end{aligned}$$

$$\begin{aligned} E &\rightarrow EAE \mid T \\ T &\rightarrow TMT \mid (E) \mid v \mid d \\ A &\rightarrow + \\ M &\rightarrow * \end{aligned}$$


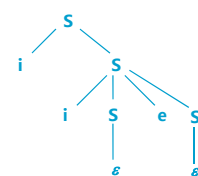
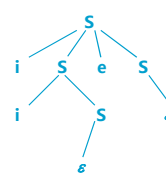
2022/4/12

School of Software

47

二义性消除

悬挂else二义性

$$S \rightarrow \varepsilon \mid iS \mid iSeS$$
字符串: iie 

2022/4/12

School of Software

48

二义性消除

悬挂else二义性

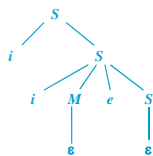
$$S \rightarrow \varepsilon | iS | iSeS$$

采用最近嵌套方法消除悬挂 else 二义性

将文法变换为下面的文法

$$\begin{aligned} S &\rightarrow \varepsilon | iS | iMeS \\ M &\rightarrow \varepsilon | iMeM \end{aligned}$$

串 iie 存在唯一的分析树 (右图)



2022/4/12

School of Software

49

固有二义语言

如果上下文无关语言 L 的所有文法都是二义的, 则称 L 是固有二义的 (*inherently ambiguous*)

某些上下文无关语言只有二义文法

2022/4/12

School of Software

50

固有二义语言

例:

$$L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\}$$

$$\begin{aligned} S &\rightarrow S_1 | S_2 & S_1 &\rightarrow S_1 c | A & S_2 &\rightarrow a S_2 | B \\ A &\rightarrow a A b | \varepsilon & B &\rightarrow b B c | \varepsilon \end{aligned}$$

2022/4/12

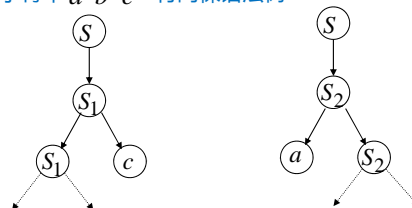
School of Software

51

固有二义语言

例:

字符串 $a^n b^n c^n$ 有两棵语法树



2022/4/12

School of Software

52

固有二义语言

例:

语言

$$L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\}$$

是固有二义的

2022/4/12

School of Software

53

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- CFG的构造实例

2022/4/12

School of Software

54

上下文无关文法的构造

- 给定一个上下文无关语言 L ，要构造其上下文无关文法 G 是一项困难的工作，因为没有针对这一问题的统一解法。
- 如果语言 L 是正则的，文法 G 的构造比较容易。
- 对于非正则语言，构造文法要困难的多。
 - 需要考虑文法中的递归结构。
 - 使用新定义的变量，必须保证递归的完整和结束。

2022/4/12

School of Software

55

上下文无关文法的构造

例 1：构造上下文无关文法 G ，接受 L

$$L = \{w = w^R \mid w \in \{0, 1\}^*\}.$$

解：

$$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \varepsilon$$

2022/4/12

School of Software

56

上下文无关文法的构造

例 2：构造上下文无关文法 G ，接受 L

$$L = \{ww^R \mid w \in \{0, 1\}^*\}.$$

解：

$$S \rightarrow 0S0 \mid 1S1 \mid \varepsilon$$

$$\{ww^R \mid w \in \{0, 1\}^*\} \subset \{w \in \{0, 1\}^* \mid w = w^R\}.$$

区别在于少了集合 $\{0, 1\}$ 。

2022/4/12

School of Software

57

上下文无关文法的构造

引理：如果 w 有 n 个 0 和 $n+1$ 个 1，那么存在一个 1 满足 $w = \alpha 1 \beta$ ，使得 α 和 β 中 0 和 1 的数量相等。证明：令 $w = a_1 a_2 \dots a_{2n+1}$ ， $a_i \in \{0, 1\}$ 。如果 $a_1 = 1$ ，那么 $w = 1\beta$ 。结论得证。如果 $a_1 = 0$ ，定义函数 $f: f(0) = 0$ ，对于 $i = 1, 2, \dots, 2n$ ，

$$f(i) = \begin{cases} f(i-1) - 1, & \text{if } a_i = 0 \\ f(i-1) + 1, & \text{if } a_i = 1 \end{cases}$$

2022/4/12

School of Software

58

上下文无关文法的构造

令 $\alpha = a_1 a_2 \dots a_k$ ， $1 \leq k \leq 2n+1$ ，那么：

$$f(k) = n_1(\alpha) - n_0(\alpha)$$

并且 $f(k+1)$ 比 $f(k)$ 多一或者少一。因为

$$f(1) = -1, \quad f(2n+1) = 1,$$

存在最小的 k ，满足 $3 \leq k \leq 2n+1$ ，使得：

$$a_k = 1, \quad f(k) = 1 \quad \text{且} \quad f(k-1) = 0$$

令 $\alpha = a_1 a_2 \dots a_{k-1}$ 。那么 α 中 0 和 1 的数量相等。

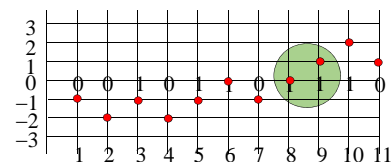
证毕！

2022/4/12

School of Software

59

上下文无关文法的构造

例如， $w = 00101101110$ 。考虑 $f(n)$ 的图像。

$f(1)$ 的值是 -1。有 $f(8) = 0$ 且 $f(9) = 1$ ， $a_9 = 1$ 。且 $\alpha = 00101101$ 拥有同样数量的 0 和 1。

2022/4/12

School of Software

60

CFG的应用与文法的二义性

- CFG的应用
- CFG的转换
- 文法二义性
- 二义性的消去方法
- CFG的构造方法
- **CFG的构造实例**

2022/4/12

School of Software

61

CFG构造实例

例 3: 构造 CFG G 满足

$$L = L(G) = \{w \in \{0, 1\}^* \mid n_0(w) = n_1(w)\}.$$

解答:

方法 1: 先看如下的 CFG

$$S \rightarrow 01S \mid 0S1 \mid S01 \mid 10S \mid 1S0 \mid S10 \mid \varepsilon$$

虽然这个文法生成的字符串中有相同数量的 0 和 1, 但是无法生成字符串 00111100.

该文法是不正确. 怎样描述该语言的 CFG 呢?

2022/4/12

School of Software

62

CFG构造实例

方法 2:

字符串 w 开头可能是 0, 在之后的某个位置有 1. 在 0 和 1 之间有一个或者更多的 0,1 对.

如果字符串的开头是 1, 构造方法类似.

构造如下的文法:

$$S \rightarrow 0S1S \mid 1S0S \mid \varepsilon$$

但是, 我们只知道 $L(G) \subset L$.

2022/4/12

School of Software

63

CFG构造实例

通过归纳法证明 $L \subset L(G)$:1) $\varepsilon \in L$,

$$S \xRightarrow{*} \varepsilon. \text{ 因此, } \varepsilon \in L(G)$$

2) $w \in L$, 并且 $|w| \leq 2n$, 假设

$$S \xRightarrow{*} w, \quad w \in L(G)$$

3) 如果 $v \in L$, 且 $|v| = 2(n+1)$.

2022/4/12

School of Software

64

CFG构造实例

- a) 如果 $v = 0w \in L$. 那么 w 中有 n 个 0 和 $n+1$ 个 1. 根据引理, 存在 1 满足 $w = \alpha 1 \beta$, 且 α 和 β 中 0 和 1 数量相等. 则有

$$S \Rightarrow 0S1S \Rightarrow 0\alpha 1S \Rightarrow 0\alpha 1\beta = v \in L(G).$$

- b) 如果 $v = 1w \in L$. 那么 w 中有 n 个 1 和 $n+1$ 个 0. 与 a) 同理, 有

$$S \Rightarrow 1S0S \Rightarrow 1\alpha 0S \Rightarrow 1\alpha 0\beta = v \in L(G).$$

证毕!

2022/4/12

School of Software

65

CFG构造实例

方法3:

另一种解决例3的方法. CFG 如下所示:

$$S \rightarrow 0B \mid 1A \mid \varepsilon, \quad B \rightarrow 1S \mid 0BB, \quad A \rightarrow 0S \mid 1AA$$

对于任意字符串 w , 有三种情况:

- (a) w 中 0 的数量和 1 的数量相等, 通过 S 生成;
- (b) w 中 0 的数量比 1 的数量多一个, 通过 A 生成;
- (c) w 中 0 的数量比 1 的数量少一个, 通过 B 生成.

2022/4/12

School of Software

66

CFG构造实例

方法4:

文法 G :

$$S \rightarrow SS \mid OS1 \mid 1S0 \mid \varepsilon$$

2022/4/12

School of Software

67

课后练习

◇ 必做题:

- P-204 Ex.5.3.3
- P-205 Ex.5.3.4(b),(c)
- P-205 Ex.5.3.5
- P-214 Ex.5.4.2
- P-214 Ex.5.4.3
- P-215 Ex.5.4.7 (a)

◇ 思考题:

- P-215 Ex.5.4.7 (b)

2022/4/12

School of Software

68

*That's all for today.**Thank You*

2022/4/12

School of Software

69