标志树文法及其语法分析*

方 林 程景云

(上海海运学院计算机系CAD研究室,上海 200135)

摘要 树文法是一种高维文法,能够用来表达二维以上复杂对象的构造规则.在模式识别、图式语言等领域有着广泛的应用前景.本文在树文法有关概念基础上提出了标志树、连接标志、标志树文法等概念,构造了标志树的匹配和识别算法,并成功解决了标志树文法的语法分析器构造问题.

关键词 模式识别,形式语言.

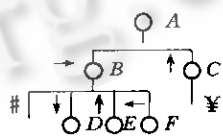
1 标志树

树是不含有回路的连通的有向图[1].标志树的概念是对树概念的推广.

定义 1. 标志树是 n 个结点的有限集合($n>0$),使得:

(1)有一个唯一指定为根的结点;(2)根结点有若干个标志,设为 m 个($m\geqslant 0$);(3)除根以外其余结点被分为 $p(0\leqslant p\leqslant m)$ 个不相交的集合 $T_1,T_2,\cdots,T_p$,其中每个集合又都是一棵标志树.我们称 $T_i(1\leqslant i\leqslant p)$ 为这个根结点的子树;(4)每个子树总与根的一个标志相对应,而根的标志至多与一个子树相对应.当 $m>p$ 时,必有 $m-p$ 个标志没有对应子树,这样的标志称为属于根结点的一个连接标志.

标志树概念的上述定义是递归的,因为其中又用到了标志树概念本身.图1是标志树的一个例子.其中每个结点下面画出若干根树枝,每根树枝表示一个标志(标志就画在树枝旁),如果这个标志有对应的子树,就把树枝与对应子树的根相连,否则不画任何东西.



(其中#是属于结点B的连接标志,¥是属于结点C的连接标志)

图1 标志树实例

如果一棵标志树中存在一个连接标志,它属于两个或两个以上不同结点,这样的标志树称为不确定的标志树,反之称为确定的标志树.图1所示的是确定的标志树,如果把结点 B 的标志#改为¥,可以得到一棵不确定的标志树.

* 本文1993-09-10收到,1994-04-07定稿

本文研究受到国家自然科学基金资助.作者**方林**,1970年生,硕士,主要研究领域为图示语言和用户接口.**程景云**,1941年生,教授,主要研究领域为图示语言,人机界面,*CAD* 支撑软件等.

本文通讯联系人:方林,南京 210093,南京大学计算机科学系

2 标志树文法

2.1 替 换

设 T_1 是一棵确定的标志树，T_2 是一棵标志树，N 是 T_2 上的一个结点，它有 n 个($n\geqslant 0$)标志 $L_1,L_2,\cdots,L_n$. 如果

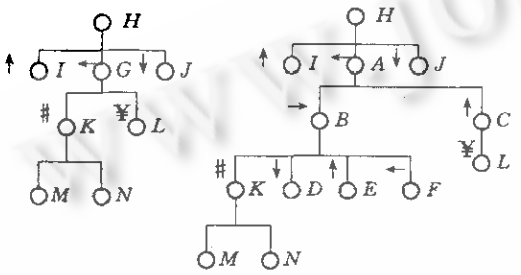
(1)每个 $L_i(i=1,2,\cdots,n)$都有对应的子树；

(2)每个 L_i 又都是 T_1 的一个连接标志，并且 T_1 除此外没有其他连接标志，那么称 T_1 满足结点 N 的替换条件. 这时可以用 T_1 去替换它，步骤是：

1. 把 N 的所有子树连到 T_1 中相应的连接标志(树枝)上得到一棵新标志树 T'_1；

2. 再把 T_2 的以 N 为根的子树换成 T'_1.

例如，用图 1 所示的标志树替换图 2(1)的结点 G 得到图 2(2).



(1)替换前　(2)替换后(G结点已被A替换，它的两棵子树已被分别连接到B,C两结点上)

图2 替换实例

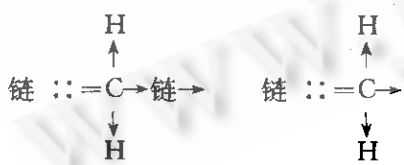
2.2 标志树文法

定义 2. 标志树文法 G 是一个五元组 (N,T,L,P,S)，其中(1)N 和 T 分别是非空有限的非终结符集和终结符集，N 与 T 不相交；(2)$S\in N$ 是一个非终结符，称为文法的开始符；(3)P 是产生式的集合，每个产生式由左部和右部两部分组成，左部是一个非终结符，右部是一棵以终结符和/或非终结符为结点的标志树. 左部和右部之间用符号"::="隔开. P 中至少有一个产生式的左部是 S，所有左部是 S 的产生式其右部标志树中没有任何连接标志；(4)L 是所有产生式右部标志树中标志的集合.

以下是标志树文法的一个例子(我们把标志和树枝画在了一起)，它生成醇的分子式.

例 1：G=〈{醇,链},{H,O,C},{↑,→,↓},P,醇〉，其中 P 的定义是：

醇::=H→链→O→H



2.3 推导的形式定义

定义 3. G 是一个标志树文法. 设 x 是一棵标志树，如果 G 中存在一个产生式 p，它的左部 N 是 x 中一个结点. 如果 p 的右部标志树满足结点 N 的替换条件，y 表示替换结果，那么称在文法 G 中，x 可以直接推导 y，记作 $x\Rightarrow y$，y 称为 x 的一个直接推导.

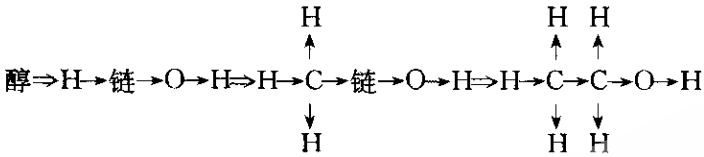
定义 4. 如果在标志树文法 G 中，存在

$$x=x_0\Rightarrow x_1\Rightarrow x2\Rightarrow\cdots\Rightarrow x_n=y(n\geqslant 0)$$

则称 y 能够由 x 推导出来.

定义 5. 设有标志树文法 G,S 是它的开始符,能由 S 推导出来的标志树称为文法 G 的一个树型,既没有非终结符又没有任何连接标志的树型称为终结树.

下面以例 1 中的标志树文法为例,给出乙醇分子式的推导过程:



3　匹配和识别

3.1　匹　配

标志树中的标志可以用来表达结点之间的相互关系,这种关系既可以是具体的(如空间位置关系),又可以是抽象的(如家族血统关系). 如果把结点看成是现实世界中构成对象的元素,那么当一棵标志树真实反应了一个对象的各个元素之间的相互关系时,我们称这棵标志树与这个对象匹配,反之,称为不匹配. 设"→"和"↓"分别表示"…在…左边"和"…在…上边"两个关系,则图 3(1)所示的标志树与图 3(2)所示的平面图形相匹配.

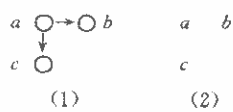


图3　标志树和图形的匹配

下面算法 *match* 可以判定给定终结树是否与指定对象相匹配. 它有两个参数,其中 LT 表示要匹配的终结树,$ELEMENT$ 表示当前元素. $ELEMENT$ 的初始值是要匹配对象中的一指定元素(称为起始元素).

算法 1:*match*(LT, $ELEMENT$)

如果 LT 中仅有一个结点

则　如果该结点的值与 $ELEMENT$ 的值相等

　　则　返回 TRUE

　　否则　返回 FALSE.

否则　设 R 是 LT 的根,R 的标志是 $L_1, L_2, \cdots, L_n$. 与 $L_i(i=1,2,\cdots,n)$ 对应的子树是 LT_i,那么

　　如果 $match(LT_i, next\_element(ELEMENT, L_i))$ 对于任意 $i=1,2,\cdots,n$ 都成立

　　则　返回 TRUE

　　否则　返回 FALSE. □

算法中 $next\_element(ELEMENT, Li)$ 是一个函数,返回和元素 $ELEMENT$ 一起满足标志 Li 所定义关系的另一个元素. 例如对于图 3(2)所示图形来说,有 $next\_element(a, \rightarrow)=b$, $next\_element(a, \downarrow)=c$.

3.2　识　别

标志树文法反映了一类标志树的共同特征,根据算法 1 我们可以判断一棵标志树是否与一个对象相匹配. 这样,标志树文法可用来概括一类对象的共同特征. 如果我们把这种具有共同特征的对象归属为一类的话,所谓识别就是判断指定对象属于哪一类的过程.

下面我们构造一个算法来实现任意标志树文法 G 对任意对象 S 的识别. 形参 LT 记录了由文法 G 的开始符所推导出来的一棵树型. 如果 LT 是终结树,则把它同对象匹配,匹配的结果就是识别的结果;否则应该寻找 LT 的一个直接推导 LT',如果用 LT' 不能识别 S,就

要寻找 LT 的另一个直接推导，再去识别，…，如此循环，直到能够识别 S（识别成功）或 LT 的所有推导都找完为止（识别失败），LT 的初始值为文法 G 的开始符.

算法 2：*recognize*(LT)

若 LT 是一个终结树，

则 返回 *match*(LT，对象 S 的起始元素）的结果.

否则 1. 寻找 LT 的一个直接推导 LT'；

2. 若找到，

则 如果 *recognize*(LT')=*TRUE*

则 返回 *TRUE*；

否则 寻找 LT 的另一个直接推导 LT'，转 2.

否则 返回 *FALSE*. □

4 语法分析程序

事实上我们可以把匹配和识别过程合二为一从而为文法 G 构造特定的识别器即语法分析程序. 从实现角度看，这比构造一个通用的识别器要容易得多. 更重要的是避免了通用识别器中的组合爆炸问题.

我们可以为每一个产生式写一个子程序. 子程序有若干个形式参数记录结点信息，其中第一个参数记录进入子程序时的当前元素，另有若干个参数作为返回参数用. 子程序的模块结构一般是先匹配根，再匹配子树（多个子树的匹配可以并行进行），而子树也是先匹配根，后匹配子树如此递归下降直到树叶. 含有连接标志 L 的结点 N 被用来计算和 N 一起满足 L 所定义关系的另一个元素（通过子程序 *next_element* 实现），它被返回给子程序的调用者作为下一步语法分析的当前结点. 下面以例 1 中的标志树文法为例，使用 *PROLOG* 语言说明递归下降语法分析程序的构造过程.

```
/* 识别醇的子程序,CN 是当前元素 */
    醇(CN):—match(CN,"H"),                    /* 先匹配根,并且 */
            next_element(CN,"→",NN),           /* 得到 CN 右边的结点 NN. */
            链(NN,RN),                         /* 匹配子树"链→O→H"的根,返回
                                                  右边结点 RN */
            match(RN,"O"),                     /* 匹配子树"O→H"的根 */
            next_element(RN,"→",GN),           /* 得到 RN 右边的结点 GN. */
            match(GN,"H").                     /* 匹配树叶"H" */
/* 识别链的子程序,CN 是当前元素,RNODE 返回右边元素 */
    链(CN,RNODE):—match(CN,"C"),
                  next_element(CN,"↑",UN),
                  next_element(CN,"↓",DN),
                  next_element(CN,"→",RNODE),
                match(UN,"H"),
                match(DN,"H").
    链(CN,RNODE):—match(CN,"C"),
```

$next\_element(CN,"\uparrow",UN)$,
$next\_element(CN,"\downarrow",DN)$,
$next\_element(CN,"\rightarrow",RN)$,
$match(UN,"H")$,
$match(DN,"H")$,
链$(RN,RNODE)$.

值得注意的是,*next_element* 相当于串文法语法分析程序里的词法分析程序.与之不同的是,后者没有表示关系的标志的概念,隐含的是指当前元素的后继元素;而前者必须指明一种关系才能得到与当前元素相关的另一个元素.

5　结束语

标志树文法是对一般树文法的扩充.每个标志反映了构成对象的元素间的一种关系.这种关系可能是具体的,也可能是抽象的.这样,我们可以用标志树文法来抽象元素间具有复杂关系的一类事物的共同特征.标志树文法的重要意义还在于它的语法分析程序结构简明,可以用并行程序实现.另外,尽管本文提出的是上下文无关的标志树文法,但我们可以把有关概念扩展到上下文有关标志树文法中去,还可以提出更抽象的文法——标志图文法.对本文提出的匹配、识别算法和语法分析程序构造规则略作变动,它们就能适用于标志图文法.

参考文献

1　王兵山,王长英,周贤林等.离散数学.长沙:国防科技大学出版社,1985.
2　Gonzalez R C, Thomason M G, 濮群等译.句法模式识别.北京:清华大学出版社,1984.

LABELLED TREE GRAMMAR AND A PARSER FOR ITS SYNTAX

Fang Lin　Chen Jingyun

(*CAD Institute, Department of Computer Science, Shanghai Maritime University, Shanghai* 200135)

Abstract　Tree Grammar is a kind of highly dimensional grammar which can be used to represent the rules for creating 2_D and up objects. Tree Grammar will be extensively applied to Pattern Recognition and Visual Language . This paper introduces the concepts of Labelled Tree, Connection and Labelled Tree Grammar based on related concepts of Tree Grammar. It also presents two algorithms for matching and recognizing labelled trees and provide a method to generate the parser for labelled tree grammars.

Key words　Pattern recognition, formal language.