

# 多级文法在通用型中间件中的应用\*

程玲

包振东 黄上腾

(IBM 中国研究中心 北京 100085)

(上海交通大学 上海 200030)

**摘要** 在当今计算机技术发展的主流中,中间件技术是客户/服务器结构的关键技术.在对数据库专用中间件——RDA 技术的深入研究后,针对它灵活性低、可扩充性差的特点,本文提出了适于构造基于通用型中间件的分布式环境的新技术——多级文法.实践证明将多级文法的技术引入中间件的构造大大提高了中间件的灵活性,增强了它的可扩充性,并使中间件本身具有了一些新的特点.目前,一个基于该多级文法的通用型中间件的原型系统已可在 UNIX 平台运行.

**关键词** 多级文法,三维文法表,非确定性匹配,外部文法的控制机制.

**中图法分类号** TP301.2

目前,客户/服务器体系结构已逐步成为分布式系统环境的主导.客户/服务器接口,即中间件是整个系统的基础,其核心是分别位于接口两端的一对服务上下文逻辑协议自动机,即服务文法.中间件的构造方法和实用性能在很大程度上决定了分布式环境对各种应用的支持能力.

基于客户/服务器结构的分布式系统中,各服务实体在逻辑上相互独立,但往往又属于同一外部整体.在这种系统环境中,服务逻辑局部变动的可能性较大.而根据传统的自底向上的文法构造法构造的一整块单层二维文法表,由于各部分相互联系,牵一毫而动全身,不便于整个系统的修改,从而限制了中间件的通用性、灵活性,文法的自动生成环境也难以建立.针对这一不足,我们提出了基于三维文法表的多级文法,使得中间件的构造更具通用性及灵活性,也便于系统文法的自动生成.

本文讨论正规文法的结构化划分,揭示了多级文法的实质;并对多级文法的实现、意义进行了阐述.

## 1 多级文法的提出

### 1.1 正规文法

正规文法是整个工作的基石,我们利用正规文法来对各服务实体的服务逻辑进行描述.由于正规文法只含有以下 2 种形式的产生式:

\* 本文研究得到国家“八五”攻关项目基金资助.作者程玲,女,1970年生,硕士,主要研究领域为数据库.包振东,1966年生,讲师,主要研究领域为分布式计算与人工智能.黄上腾,1943年生,教授,主要研究领域为新型数据库.

本文通讯联系人:包振东,上海 200030,上海交通大学软件基地, E-mail: josephb@public.sta.net.cn

本文 1996-09-20 收到修改稿

$$A \rightarrow \alpha B \quad A \rightarrow \alpha$$

其中  $\alpha \in V_T^+(G)$ ,  $A, B \in V_N(G)$ . 所以对任何一个正规文法  $G$ , 可以设计一个 NFA, 它能够而且只能够识别  $G$  的语言. 我们把  $G$  的每个非终结符看作是 NFA 的一个状态, 令  $G$  的开始符号  $S$  对应 NFA 的初态. 另外, 为这个 NFA 新设一个唯一的终态. 对于形如  $A \rightarrow \alpha B$  的产生式, 画一条从状态  $A$  到状态  $B$  的弧, 此弧的标志为  $\alpha$ . 对于形如  $A \rightarrow \alpha$  的产生式, 从状态  $A$  画一条标志为  $\alpha$  的弧到达终态. 显然, 该 NFA 能够且只能够识别  $G$  所产生的语言. 类似地, 我们可以对任何 DFA 构造一个相应的正规文法  $G$ .

根据 DFA 图与正规文法的等价性以及 RDA 国际标准, 我们利用带操作 (Action) 的 DFA 状态图来表示服务实体的服务逻辑, 如图 1 所示, 其中详细的描述方法请见文献[1], primitive/action 便为其文法的终结符. 本文后面的讨论均基于这种 DFA 图.

## 1.2 多级文法的思想——结构化方法

结构化方法是开发大型系统的必要方法, 尤其在分布式环境中, 多个服务实体相互协作完成复杂的功能. 客观上, 各服务实体相互独立, 功能单一, 本身便适合结构化设计 (SD) 的思想.<sup>[2]</sup> 所以, 若利用结构化方法对整个系统的服务上下文逻辑进行重构, 使原来描述整个服务上下文逻辑的平面单层文法“模块化”, 各独立的模块即局部文法通过拼装, 便形成了与原平面单层的全局文法相等价的多级文法. 这样可简化服务逻辑的描述, 并能保留服务上下文逻辑中的层次和递归特性. 在系统实现中, 我们通过三维文法

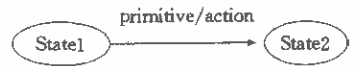


图1

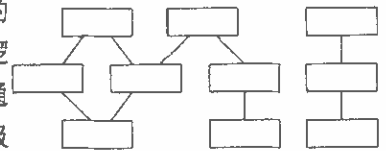


图2

自动机完成多级文法的实现. 使每一文法模块对应于一个服务实体的服务逻辑, 模块内部的各种联系为一服务实体的“内部文法”; 各文法模块之间的关联便作为服务实体的“外部文法”. “内部文法”是对各服务实体内部服务逻辑的描述, 逻辑关联较复杂. 而“外部文法”是对各服务实体之间的逻辑约束的一种描述, 逻辑关联较简单. 各“内部文法”通过“外部文法”相联系, 形成一种带偏序的有向无环图 (DAG) 结构. 这便是本文的核心——多级文法.

## 2 多级文法的构造及实现

多级文法的构造基于对正规文法的结构化划分. 由此形成的多级文法的结构是如图 2 所示的一种由多个具有偏序关系的有向无环图 (DAG) 组成的图集, 偏序关系由上至下. 同层文法间可以具有并行性. 一个局部文法可以被多个上层文法所共享, 即该文法所描述的服务实体的服务功能可以被多个服务实体所共享. 与原来的单层文法不同, 多级文法的关键是要提供进入下层局部文法及返回上层局部文法的条件制约机制, 即外部文法的控制机制. 从而保证服务上下文逻辑的完整性.

我们利用对文法自动机的 UNKNOWN 项的非确定性匹配<sup>[1]</sup>来实现进入下层自动机的控制. 由于局部文法之间的关联不同, 它们之间的这种条件制约机制也就有所不同. 下面我们将分不同的关联情况, 对正规文法的结构化划分及外部文法控制机制进行详细的讨论.

### 2.1 一般化关联

在一般化关联的情况下, 下层的局部文法被抽象为与之关联的上层文法中的 2 个状态

结点——入口状态和出口状态. 这与子过程类似. 上层的局部文法由入口状态进入下层的局部文法; 当下层的局部文法到达终态时, 由出口状态返回上层的局部文法. 可以有多个下层的局部文法对应于同一上层文法的同一对入口和出口状态, 此时在该关联的情况下, 它们都是一种或关系, 即每个下层的局部文法对应一条分支. 一般化关联根据入口状态和出口状态的关系可分 2 种情况: 入口状态与出口状态不统一(简称出入相异)及入口状态和出口状态统一(简称出入相同).

### 2.1.1 基于出入相异的文法划分及其控制机制

设  $G$  是正规文法, 作  $G$  的 DFA 图  $\Delta(G)$ . 设  $A, B \in V_N(G)$ , 即  $A, B$  为  $G$  的 2 个状态.

对  $\forall \alpha \in V_T^+(G), \forall C \in V_N(G)$ , 若  $G$  中存在产生式  $A \rightarrow \alpha C$ , 则必存在产生式  $B \rightarrow \alpha C$ , 反之亦然, 则称  $A, B$  的出口条件等价, 记作  $O(A) \equiv O(B)$ .

若  $A \rightarrow^* B$  成立, 则称  $B$  为  $A$  可到达的状态, 令  $\Sigma(A)$  为  $A$  所能到达的所有状态的集合.

若有  $A \rightarrow \alpha B$  成立, 则称  $B$  为  $A$  直接导出状态. 令  $N(A)$  为  $A$  所能直接导出的状态集.

若  $\exists T_1, T_2, \dots, T_n \in \Sigma(A), T_1, T_2, \dots, T_n$  均非  $\Delta(G)$  的终态, 且有  $O(T_1) \equiv O(T_2) \equiv \dots \equiv O(T_n)$  成立, 则令  $\Gamma = \{T_1, T_2, \dots, T_n\}$ .

令  $F_A(B)$  为由状态  $A$  出发, 首次到达状态  $B$  所经历的所有状态结点, 对于  $\Gamma$  这样的状态集合,  $F_A(\Gamma)$  表示  $A$  出发首次到达  $\Gamma$  中所有状态所经历的全部状态.

令  $D$  为  $F_A(\Gamma)$  所对应的 DFA 子图.

若  $\exists \Gamma' \subseteq \Gamma$ , 且满足以下条件:

- (1)  $V_T(D) \cap V_T(G-D) = \emptyset$
- (2)  $V_N(D) \cap V_N(G-D) = \emptyset$
- (3)  $\forall X \in V_N(G-D)$ , 则  $N(X) \cap F_A(\Gamma') \subseteq \{A\}$
- (4)  $\forall X \in V_N(D)$ , 则  $\Sigma(X) \subseteq F_A(\Gamma') \cup \Gamma'$

这样, 可将  $D$  抽象为一下层的局部文法. 其中可以将  $A$  作为其初态, 即上层文法进入其入口状态, 也可在下层文法中另定义初态;  $\Gamma'$  为其终态集, 由于它们的出口条件一致, 因此在其上层局部文法中, 可将  $\Gamma'$  合成为一个状态, 即出口状态. 这样该局部文法便被抽象为两个状态结点. 上述条件中的 (1)、(2) 表示该局部文法中的终结符和非终结符不得与其它局部文法中的相同. 从而保证了局部文法的相对独立性. 条件 (3)、(4) 要求文法的入口和出口唯一, 从而使各文法模块间的联系简单化, 控制方便.

让我们来看一个例子, 图 3 为一文法  $G$  的描述. 根据上述划分方法, 可将该文法分解为 3 个文法模块, 如图 4、5.

局部文法  $G_1$  被抽象为上层文法中的状态  $B$  和  $O1$ ,  $G_2$  则对应状态  $B$  和  $O2$ . 其中  $B$  为这 2 个下层局部文法共同的入口状态.

这种情况下的外部文法的控制机制具有一般的特点, 上层文法在入口状态下通过发生非确定性匹配<sup>[1]</sup>, 进入其相应的下层局部文法进行确定性的匹配搜索. 这种非确定性的匹配是一种或关系, 记为  $\pi$  型非确定性匹配, 在该型的非确定匹配下, 对应于同一入口状态下的同层的局部文法, 只是不同的分支, 它们之间不具有并行性. 同时, 当每个下层局部文法到达终态时, 便从与其入口文法相对应的出口状态返回上层文法. 因此在每个下层局部文法中需

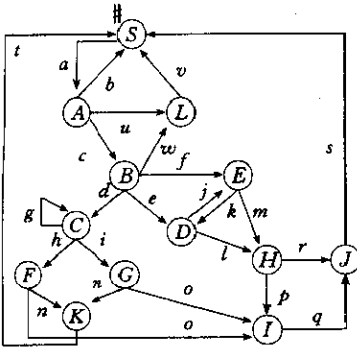


图3 正文语法 G

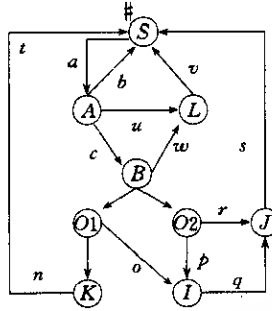


图4 上层语法 G<sub>0</sub>

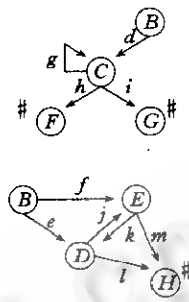


图5 下层局部语法 G<sub>1</sub>, G<sub>2</sub>

对其出口状态加以记录.

### 2.1.2 基于出入相同的文法划分及其控制机制

这是上一种情况的特例,即当一局部文法所对应的入口与出口状态相一致的情况,这样,该局部文法便被抽象为一个状态.此状态既为其初态,亦为其终态,相关的上层文法可以由该状态进入这个下层文法,在到达终态时,由该状态返回原上层文法.这种情况下的文法划分方法与上一种情况相同.这种情况也十分普遍.其出入的条件控制机制较简单,将其非确定性匹配记为  $\theta$  型,它也是一种或关系.由于该局部文法的入口状态和出口状态一致,所以该局部文法不必记录其出口状态.

### 2.2 继承性关联

当一个局部文法被抽象为一个状态结点时,该局部文法中的状态可继承该状态在上层文法中的部分转换路径.此时,该状态结点称为此局部文法的父结点.令该情况下的非确定性匹配为  $\sigma$  型.

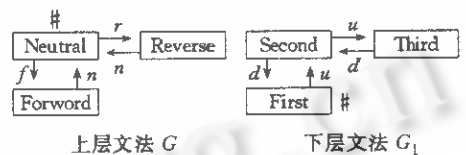


图6

图 6 为自动传输的状态图.其中上层文法为方向

控制的文法,其中 Forward 状态是下层速度控制文法的抽象,设子文法继承其父状态的所有转换路径.在任何前档(Forward)中选“n”都引起其到 Neutral 的转换.从 Forward 到 Neutral 的转换隐含 3 种所继承的转换.在其外部文法控制机制中,需要记录所继承的路径.在外部文法中,通过将记录该路径所经历的状态标志为同一 ID 来实现.具有相同 ID 的状态之间可相互转换而不必关心下层文法的状态.实质上,当上层文法满足某一条件后,便进入下层文法,此时,上层文法与下层文法之间可以部分并行.对于共享的局部文法,可以进行多重继承.

### 2.3 并发性关联

并发性是目前系统实现中一种高效方法,尤其在分布式环境中,各个服务功能均由不同的服务实体来承担,为并发的实现带来了方便.对应于同一对人口和出口状态的一组局部文法,与一般化关联不同,它们之间的关系为与关系.

例如:有收音机服务逻辑的文法如图 7 所示.其中 a, 打开; b, 关闭; c, 调高音量; d, 调低音量; e, 调高频段; f, 调低频段; F, 关机状态; LP1, 低音频段 1; LP2, 低音频段 2; LP3, 低音频段 3; HP1, 高频段 1; HP2, 高频段 2; HP3, 高频段 3.

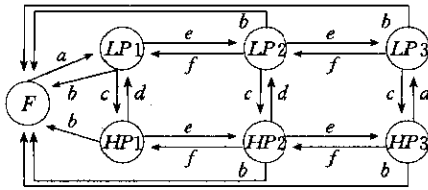


图7 收音机服务逻辑的文法

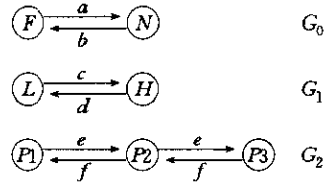


图8 分划后的文法

经研究发现,我们可以用如图 8 中的 3 个文法  $G_0, G_1, G_2$  更为方便地代替上述文法  $G$ , 其中的关键便是音量调节与频段调节在本质上相互独立且并行的. 其中  $G_1, G_2$  被抽象为一个状态结点  $N$ , 且文法  $G_0$  中包含了这个状态结点.  $G_1$  为调音文法,  $G_2$  为调频文法, 对并行性关联来讲, 整个系统的状态不可能用单个文法的单个状态来表示, 它是所有对象的一种组合. 例中, 原文法的 6 种状态表现为文法中状态和文法中状态的组合,  $(L, P1)$  对应于  $LP1$ ;  $(L, P2)$  对应于  $LP2$ ;  $(L, P3)$  对应于  $LP3$ ;  $(H, P1)$  对应于  $HP1$ ;  $(H, P2)$  对应于  $HP2$ ;  $(H, P3)$  对应于  $HP3$ . 由此我们可以看出这种并行性的特点使文法的规模明显变小, 且可支持并行性, 提高了系统的效率.

在这种关联下, 外部文法的控制机制也具有一些新的特点. 首先, 多个原语可同时在一入口状态发生非确定性匹配, 记为  $\xi$  型非确定性匹配. 进入下层多个局部文法中并行进行确定性匹配搜索. 对于出口条件控制, 则增加了对并发活动的同步控制机制. 它要求只有当所有的并发活动均结束后, 才允许其出口状态发生进一步迁移.

以上我们分 3 种情况对多级文法的构造及其外部文法的控制机制作了详细的分析. 在这里我们还要强调指出, 外部文法的控制机制是实现多级文法的核心, 它除了要包含上述的几种情况外, 还需考虑一种情况, 即由于一个局部文法模块可以被多个上层文法所共享, 在这些上层文法中该局部文法的入口与出口都是一致的, 为了避免冲突及二义性, 每个局部文法都需记录当前进入其上层文法的信息, 任意时刻只允许一个上层文法进入.

### 3 多级文法的意义

在一个分布式环境中, 每个服务实体的服务逻辑对应一个局部文法, 它们之间的关联便通过外部文法进行拼装. 当一个文法发生变动时, 只需对局部文法及相关的外部文法进行修改, 而不影响整个系统的文法. 当环境中要增添一个服务实体时, 只需将其局部文法通过定义相应的外部文法控制机制, 将其接插到环境中即可, 从而实现了即插即用的思想, 并为文法的自动生成创造了环境. 另外, 多级文法也带来了并行性和自导向性, 使整个系统的灵活性大大提高. 对于具体的系统, 我们认为上述几种关联情况是完全的, 过于复杂的关联会导致外部文法的控制机制过于复杂, 代价过高. 对于这种情况不如将它们合为一个文法.

### 4 结束语

本文对多级文法的构造及实现作了详细的讨论, 并对其在分布式环境中所发挥的作用进行了阐述. 为我们构造的基于通用型中间件的分布式环境提供了坚实的理论基础. 同时, 也为将来构造分布式对象环境提供了方便.

## 参考文献

- 1 包振东. 开放式并行自导向客户/服务器接口, 计算机工程, 1995, (增刊).
- 2 唐稚松. LBLR(K)文法与文法分划. 计算机学报, 1980, (1), 11~15.

## THE CONSTRUCTION AND IMPLEMENTATION OF MULTI-LEVEL GRAMMAR

CHENG Ling

(IBM China Research Center Beijing 100085)

BAO Zhendong HUANG Shangteng

(Shanghai Jiaotong University Shanghai 200030)

**Abstract** In this paper, a kind of multi-level grammar is introduced. It is the theoretical basis of plugable middleware based on distributed environment. Beginning with the distributed environment based on the client/server architecture, the authors propose the multi-level grammar based on 3-D syntax analysis table, and then expound its supporting mechanism thoroughly, so as to implement the plug and play protocol automation automatically.

**Key words** Multi-level grammar, 3-D syntax table, unknown matching, external grammar controlling mechanism.

**Class number** TP301.2