ADA和PROLOG的比较

黄　明（吉林大学计机算科学系）

摘要

Ada语言是一种功能极强的程序设计语言，已成为80年代最有影响和最有代表性的一种高级语言；Prolog语言是建立在符号逻辑基础上的简单而功能却很强的程序设计语言。由于日本的FGCS计划把Prolog语言作为系统的核心语言，所以该语言目前已受到计算机界的极大重视。本文对这两种语言在通讯和同步方面、并行处理方面、变量共享方面、匹配方面、错误处理等方面的特点进行了比较和分析。如果在设计新的语言时把上述两种语言各自的优点有机结合起来，那么这种新语言的生命力一定很强。

Ada语言适用于数值计算和系统程序设计，并能满足实时分析及并行操作的要求。Ada的优势并不在于语言本身有多么好，而是它为软件工程实践的应用提供了良好的工具。而Prolog语言的应用范围很广，可用于专家系统设计、智能软件设计、解方程、情报检索等方面。

在这篇文章中，通过对这两种语言在许多方面进行比较和分析，特别是对通讯和同步及并行处理方面的比较，试图为设计新的同步工具和程序设计语言提供参考。

1　通讯和同步

Ada任务间的通讯方式和Prolog谓词间的通讯方式是类似的。

在Ada中，任务是可以与其它程序设计单元（子程序和程序包）并行操作的程序单元，任务是并行处理的基本工具。一个任务的生存期包括：任务的活化、任务的执行和任务的结束。任务间的同步机构有两种：一种是一个处理单位只有当由它启动的任务都结束时才告终止；另一种是汇合机构，下面举一个两任务间汇合的情况：

有两个任务L和M，L是被调用任务，M是调用任务。M中有CALL语句，L中有ENTRY（入口）和ACCEPT语句。入口用于任务间的通讯，ACCEPT语句指明当入口调用时要执行的动作。两个任务L和M活化后可并行执行，这有两种情况：第一，如果任务M首先到达CALL语句处，则M被挂起，等待任务L到达ACCEPT语句处；第二，如果任务L首先到达ACCEPT语句处，则L被挂起，等待M到达CALL语句处。两个任务在汇合时必须是确定的和激化的，汇合时两个任务可通过通讯变量传递参数，参数可为IN、OUT和IN-OUT型。

在Prolog中，两个谓词间的通讯是在调用请求调用被调用谓词时发生的。这个请求激活了被调用谓词，这隐含了对通讯变量的处理。Prolog的通讯变量只有IN和OUT型。约束变量不能由被调用谓词改变其值，只有自由变量可被赋值。

2　并行处理

Ada适用于嵌入式计算机系统领域。在该领域中要求某些不同的动作必须并行执行，Ada为这些不同的动作提供了并行执行的设施，这就是Ada中的任务，这些不同的动作可用几个不同的任务来处理。任务定义了可与其它任务并发执行的动作，n个并发任务的动作序列在时间上可以是重迭的，也可以是交叉的。若有n台处理机和n个并发任务，则这n个任务的动作在时间上是重迭的；若只有一台处理机，则这n个任务只能轮流执行。这两种执行方式实质上是一样的，只不过前者是真正的并行，而后者是逻辑上的

并行，关键是任务在概念上是并行执行的。n个任务可以互不干扰地工作，也可以通过传递消息来通讯。

Prolog程序中的一组合取条件是顺序执行的，即从左到右对合取条件顺序求值，并且把已求得的变量值传送给右边的条件。但Prolog包含有许多潜在的并行性，每种并行性都暗示一种实现进程管理的具体方式。这些并行性是：

OR并行：并行处理所有匹配子句。

AND并行：并行处理子句内的若干目标。

流并行：并行处理逻辑程序中各组子目标数据流。

Prolog中这些潜在的并行性使它能够成为适合并行处理的程序设计语言。目前已实现了几种并发逻辑程序设计语言，如并发PROLOG、PARLOG、EPLLOG、DELTA-PROLOG、LOGLISP等，其中有代表性的是并发PROLOG和PARLOG。并发PROLOG和PARLOG有许多共同点，如都提供了开发AND、OR并行性的手段，都使用流通讯等，但两者仍有明显的差别，主要差别在于表述进程间通讯限制的方法。并发PROLOG使用只读加注而PARLOG使用mode作为通讯限制，两者具有相同的表现力，只是mode概念更加自然和容易使用。

3 共享变量

Prolog和Ada的共享变量在语义上存在着差异。Ada对共享变量的使用具有限制，可在两个通讯的任务间使用共享变量。Prolog中共享变量的概念与Ada中共享变量的概念不同。在Prolog中，两个自由变量可以匹配，导致两个变量共享，两个共享变量，一旦一个被赋值，则另一个也为同值。另一方面，在Prolog中任何一个变量离开它所在的子句就没有任何值的概念，从这个意义上讲。Prolog中的变量都是局部变量。

4 匹配

当两个Ada任务间通讯时发生匹配。Prolog中的匹配有以下几种：

(1) 一个整数或原子只能与同类型的整数或原子匹配；

(2) 一个自由变量可与任何目标匹配，此变量被赋为那个目标；

(3) 如果算符和自变量个数相同，则二结构匹配，且对应的自变量也匹配。

在确定什么时候得到正确的选择方面，Ada不同于Prolog。在Ada中，每一个单一调用对应一入口，最多只有一个ACCEPT语句被执行。当ACCEPT语句执行完后，调用任务恢复执行。在Prolog中，只有当一个谓词的所有子目标都被满足时才得到正确的选择。

5 错误处理

Ada中对错误的处理和Prolog中对失败的处理隐含着相似之处。在Ada中，异常是用来处理错误或一些特殊情况的工具。当一个任务M调用另一个任务L时，如果L已经结束，则在M中将引发异常。导致停止执行引发异常的模块，由异常处理程序代替该模块继续执行。在Prolog中，一个子目标的失败使调用过程中正在处理的子目标立即回溯。回溯可能产生多个解，引入了不确定性。在Ada语言中，控制非确定性时终止和延迟语句比选择语句为好。在Prolog中，控制非确定性时谓词fail和cut比回溯为好。

Prolog和Ada分别是为不同目的而设计的程序设计语言，但两者有某些相似之处，如果把两者的优点结合起来构造一种新的语言，那么这种新语言的适用范围将更加广泛。

参考文献

1. Cathy C. Neube, Comparisons and Contrasts of Ada Tasking Model and Prolog Predication, Software Engineering Notes, 1988.4.

2. Grady Booch, Software Engineering with Ada, 1981.

3. "Turbo-Prolog 1.0"—the natural Language of artificial intelligence, Borland INC, 1986.7.

4. 万迭等编译，ADA语言，1987。