计算机科学 1999 Vol.26 №.7

P-Ada：一个基于工作站机群的 Ada95 编译系统

P-Ada: A Parallel Ada95 Compile System on a Workstation Cluster

陆 嘉 温冬婵 王鼎兴
（清华大学计算机系 北京 100084）

Abstract This paper describes an approach to implement a parallel Ada95 compile system on a workstation cluster-P-Ada. P-Ada consists of a source to source precompiler, a runtime system and a procedural interface. It can convert single Ada95 program with tasking syntax into a set of programs which can execute in a coordinated fashion using message-passing primitives.

Keywords Ada95, Workstation cluster, Pre-compiler, Run-time system

Ada95 语言是一种功能十分强大的高级程序设计语言，其诸多特征，例如强类型检查、数据抽象以及面向对象等，反映了现代软件工程的要求，便于程序员开发高可靠性、高可移植性的大规模实时软件系统。在并行处理方面，Ada95 语言提供了任务描述机制用以支持并发程序设计。在现有的 Ada95 编译器中，任务机制一般都是在单机上或共享内存的多机系统上利用多线程实现的，因而在不存在共享内存的松散耦合的系统中无法使用。

另一方面，随着 VLSI 技术和计算机网络技术的发展，分布计算日益流行。计算机机群系统，以其巨大的计算潜力、良好的性能价格比和可伸缩性，以及灵活的体系结构而倍受青睐。如何使机群系统的优点与 Ada 语言良好的语言特征结合起来成为了一个重要的课题。本文介绍一个基于工作站机群的 Ada95 编译系统 P-Ada 的设计与实现。

1 几种 Ada 编译系统简介

Highly Parallel Ada 是美国纽约大学开发的运行于 Ultracomputer 上的一种 Ada 并行编译系统。纽约大学的 Ultracomputer 是一种 MIMD 的计算机，是基于一种并行计算的理想模型 paracomputer 设计的。在 paracomputer 模型中，并行计算机包含有许多执行单位(PE)，这些 PE 共享一个公共的主存储器。每个 PE 还拥有自己的本地存储器用于进行独立计算。Ultracomputer 的 PE 间使用 Ω-开关网络连接。

Highly Parallel Ada 系统中并行的基本单位是 task。系统将并发任务分成两类，全局并发任务与局部并发任务，局部并发任务是并行计算的执行者，局部任务拥有自己独立的 PE 与地址空间，相互之间无法互相访问私有数据。全局并发任务用于局部任务之间的同步与参数传递。局部任务对全局任务的访问被自动顺序化。Highly Parallel Ada 是通过预编译实现的，并没有对标准 Ada 语言进行扩充。预编译器将全局任务转化为共享主存中的数据结构，将全局任务接口调用转化为对共享主存的访问，并利用 fetch_and_add 原语实现其间的同步与互斥关系。

由于 Ultracomputer 是一种共享主存的计算机，其 Ada 语言实现较为容易。

Parallel Ada System(PAS) 是瑞典 Lund 大学开发的并行 Ada 系统，包含四个部分：Ada 预编译器、运行时系统、并发任务管理内核以及底层硬件系统。在 PAS 中，一个包含多个 task 的单一 Ada 程序可以在多个处理单元上并行执行。task 与共享内存由运行时系统进行控制，对程序员透明。PAS 底层的硬件系统是一个具有共享内存结构的 MIMD 并行计算机。该计算机由多个处理单元连接在公共总线上构成。每个处理单元包括一个代码执行处理器、一个通讯处理器和一部分共享主存。

PAS 中并发的基本单位是 task，所有的 task 共享同一地址空间。当一个新任务派生时，系统在全局堆中分配一块地址空间，包含 task 控制块、task 私有栈与 task 私有堆，并将该数据结构分给某一处理单元。出于效率考虑，该数据结构一般分配在就近的

处理单元上。

PAS 中并发 task 间的同步与通讯通过共享内存来完成，系统提供了 LOCK 与 UNLOCK 两个原语来保证并发访问的正确性。系统维护一个 task 就绪队列进行 task 调度与动态负载平衡。

PAS 系统由于任务同步与通信由硬件完成，并且共享内存的体系结构十分适合 Ada 的语义，实现效率较高。

Ada Across Machines(AAM) 是美国密西根大学机器人实验室 Ada 研究小组设计(未实现)的一个分布式实时 Ada 编译系统，其目的是使单一的 Ada 程序自动分布执行。AAM 支持任意粒度的并行对象，程序包、子程序、任务与数据项都可以被分布并行。AAM 的核心是一个预编译器。

AAM 中跨节点数据访问是通过代理(Agent)实现的。每一个分布对象在系统中有一个宿主节点，该节点上的对象被称作主对象(master)。同时在所有可访问该对象的节点上都有一个代理。所有对分布对象的访问通过代理进行。

AAM 是一个实验系统。其并行粒度很小并且多代理方式开销很大，并行效率不会很高。

2 P-Ada 并行编译系统及其结构

P-Ada 并行编译系统是基于分布式工作站机群的 Ada 编译系统。该系统是为了解决下列问题而设计的：

1)Ada 95 标准语言文本中虽然提供了并发任务 Task 的描述以及 Task 间的会合通讯机制，但实际上在已有的串行编译器中，Task 是利用多线程来实现的，即不同的 Task 实际上是在一台机器上分时执行，无法利用机群系统多机并行处理的特点。

2)由于 Ada95 语言提供了 C 语言的标准语言接口，将 PVM，MPI 等消息传递库引入 Ada95 是十分方便的。但在这种情况下，并行编程常常需要在较低的层次上进行，典型情况下开发者需要显式地将数据分布到不同的物理处理机上，为访问非局部数据需显式地进行消息传递。这种编程模式需要对实际机器的体系结构有很好的了解，并且容易出错。

3)此外，标准的 Ada95 语言中并没有提供图形界面接口，为了更好地支持开发具有良好用户界面的用户程序，提供必要的图形界面库也是必须的。

针对这些问题，P-Ada 并行编译系统实现了如下的功能：

· 支持 Ada95 语言全集(ISO/IEC 8652:1995)。

· 在 Ada 语言提供的 Task 描述机制基础上，支持多 Task 在机群系统上的自动并行执行；支持 Task 间的定时、条件接口调用；支持 Task 的动态负载平衡与 Task 复用。

· 支持保护数据类型(Protected Types)。保护数据类型对象可为多机上并行的 Task 所共享。对保护数据类型的访问将被自动顺序化。

· 提供基于消息传递的 PVM 通信原语库。

· 支持 Xwindow，Motif 窗口编程。

P-Ada 并行编译系统主要由四部分构成：P-Ada 并行编译器、动态执行系统、功能调用接口以及通讯及图形库。其结构如图 1 所示。其中预编译器、动态执行系统、功能调用接口以及 PVM 通信原语接口与 Motif 图形界面接口将自行开发。

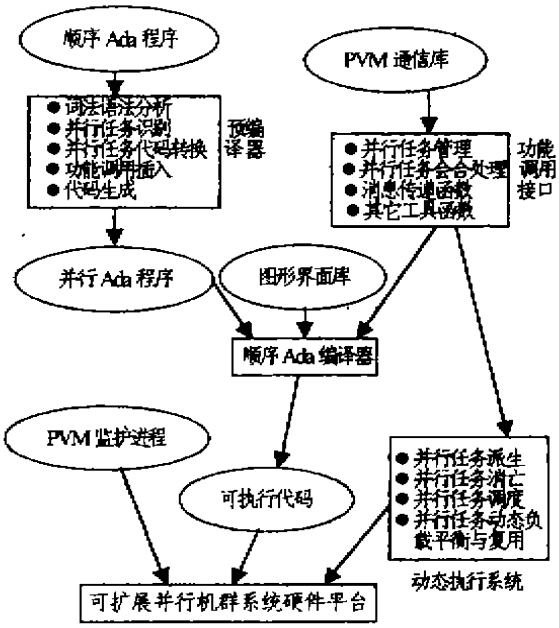

图 1 P-Ada 并行编译系统结构

P-Ada 并行编译器由预编译器和普通的串行 Ada 语言编译器构成。其中预编译器是一个源到源的程序并行化处理器，其输入是使用 Ada95 语言编写的具有 task 描述的并发 Ada 程序，通过词法语法分析、并行任务识别、并行任务代码转换、功能调用插入等步骤生成包含 PVM 通信原语的并行 Ada 程序，然后经过串行 Ada 编译器编译生成可执行代码。

动态执行系统是独立于用户程序运行的特殊进程，它负责在机群系统空间内对并行任务进行维护，

为并行任务提供动态服务。它对并行任务的派生、消亡、调度以及并行任务的负载平衡与复用进行管理，其作用很象 PVM 系统中的 pvmd。

功能调用接口是系统提供给预编译器的底层调用接口。它主要分为并行任务管理接口、并行任务通讯接口、消息传递函数与其它工具函数。其主要作用是对底层的通讯细节进行封装，使其对用户透明。当系统底层硬件平台发生变化时，只需改写功能调用函数库，即可进行系统移植。

通信与图形界面接口包含两部分，PVM 标准通信原语接口与 Motif 图形界面接口。

3 P-Ada 程序的编译与执行

P-Ada 程序是一个单一的、包含 task 与 protected type 描述的 Ada 程序。P-Ada 没有扩充 Ada95 语言。所有的 P-Ada 程序用已有的单机编译器也可编译执行，只是无法利用机群系统的并行计算能力。

如前所述，P-Ada 并行的基本单位是 task 与 protected type，由于这两者都不是可以独立编译执行的语言单位，系统必须对其进行转换，这包含两方面的内容：①程序划分。在程序中找到可并行的语法成分，并将其转化为相应的数据结构与程序代码。Pada 95 的预编译过程将把一个单一的 Ada 程序划分为多个可分布并行执行的程序。②代码变换。在程序中找到这些并行语法成分相互访问的显式的或隐式的同步与通讯语句，并将其转化为对功能调用接口的调用。

在 P-Ada 中，并行对象实例是有其类属的。每一类并行对象对应于一个可独立编译执行的 Ada 程序，该程序是由 P-Ada 预编译器根据源程序中并行对象描述自动生成。每一个活动的并行对象实例对应于一个执行该程序的 PVM 进程，在激活点由动态执行系统创建。其相互关系如图 2 所示。

Ada 语言是一种嵌套式的程序设计语言，并行对象是有层次关系的，子对象必须由父对象通过说明语句或 new 语句进行创建。父对象中保存子对象的任务控制块 TCB，其中包含子对象类属名称，子对象状态，子对象网络位置，子对象全局 ID 号等。父对象根据子对象的 TCB 调用子对象接口，进行任务同步与通讯。并行对象间也可通过访问全局共享数据进行相互作用，但系统只负责维护访问保护数据类型的一致性。P-Ada 程序运行时必须先启动动态执行系统。动态执行系统将检测可用的节点计算资源，建立各种控制数据结构和完成系统初始化。

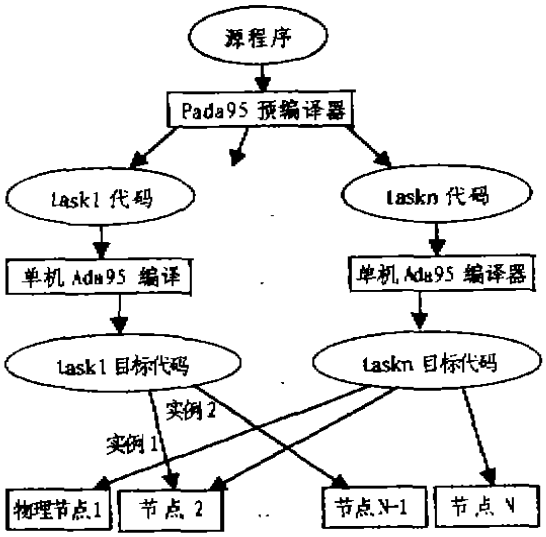

图 2 P-Ada 程序编译执行示意

P-Ada 程序由一个 Host 程序开始。Host 程序检测动态执行系统的存在并建立与其的连接关系，而后向动态执行系统注册自己的存在。Host 程序在执行前首先创建并激活所有的全局保护数据类型(protected type)对象，而后创建在自己声明部分定义的并发任务(task)对象。这些并发任务对象在被激活后还可创建自己的子对象…，整个系统开始并行执行。由于同类的并行对象的目标代码是完全相同的，其不同之处只是层次关系与激活后所接收的消息不同，因而并行对象进程是可以复用的。当一个并行对象运行结束后，其状态被置为"终止"，动态执行系统将其对应的 PVM 进程的并行对象数据进行重置，重新初始化对象消息队列，然后将其挂在就绪进程队列上。当又有创建该类并行对象的申请时，只需建立正确的连接关系即可。

P-Ada 程序的终止也必须根据并行对象的依赖关系进行。父任务只有在所有子任务终止后方可终止。

4 性能评测

我们选择了矩阵乘、分数维、CAD 三个实际程序对 P-Ada 进行了评测。矩阵乘是典型的向量计算程序，我们基于 MASTER/SLAVE 模型用 Ada95 语言进行编写。分数维程序原是在 GENESYS 并行环境下，用来测试 Transputer 多处理器的基于 MASTER/SLAVE 的 C 语言程序，我们用 Ada95

语言进行了重写。CAD 实例是采用光线跟踪算法对几何造型进行还原的程序，原是一个用 C 语言实现的串行程序，我们采用 Ada95 语言基于 MASTER/SLAVE 模型进行了重写。

评测实例源程序是一个完整的带有 Task 描述的 Ada 程序，可以用单机的 Ada 编译器直接生成可执行代码，我们将这个可执行代码的运行时间作为单机运行时间。

评测实例源程序经过 P-Ada 预编译器后转换成为多个程序，主要有 Host 程序（由主程序变换而来）、taskdefs.adb（由源程序中 task 定义变换而来）及节点程序（对应于 task 类型）。节点程序的加载、task 接口调用变换等操作由系统自动进行，用户无需对程序进行任何改动。底层硬件系统配置，如使用物理节点个数等对用户程序是透明的，P-Ada 程序既可在 4 个节点的机群上执行，也可在 8 个节点的机群上执行，无需对程序做任何调整或重新编译。我们将 P-Ada 处理后的一组可执行程序的运行时间作为多机运行时间。

我们在一个由 8 台 Sparc20 经 10M 以太网连接而成的机群系统上对 P-Ada 并行编译系统进行了测试，测试结果如下：

	执行时间		加速比	并行效率
	单机（秒）	多机（秒）		
矩阵乘实例 浮点，600×600	398.191	56.213	7.08	88.5%
分数维实例 1024×680	62.183	9.299	6.69	83.6%
CAD 实例 1024×680	120.213	17.609	6.83	85.3%

由测试结果可以得出如下结论：①P-Ada 并行编译系统具有较好的相对加速效果。并行效率一般在 85% 左右。②并行加速效果与问题的可分性直接相关。一般来说，计算量大且通讯少的问题容易得到较好的加速效果。

小结 P-Ada 编译系统通过设计实现运行时系统和预编译系统，以及其它相关系统，在工作站机群系统上支持 Ada95 语言的并行编程，并行加速效果良好。Ada95 程序员可以不需修改原有应用程序，而可以利用工作站机群系统的计算资源和能力，使其具有良好的可移植性。

参 考 文 献

1 Ada95 Reference Manual: Language and Standard Libraries ISO/IEC 8652:1995, Information Technology-Programming Languages-Ada

2 Gentleman W M, et al. Administrators and Multiprocessor Rendezvous Mechanisms. Software-Practice and Experience, 1992, 22(1): 1～39

3 Schonberg E, et al. Highly Parallel Ada-Ada on an Ultracomputer, Ada in Use. In: Proc Int. Ada Conf. 1985. 58～71

4 Lundberg L. A Parallel Ada System on an Experimental Multiprocessor. Software-Practice and Experience, 1989, 19(8): 787～800

5 Inverardi P, Mazzanti F. Experimenting with Dynamic Linking with Ada. Software-Practice and Experience, 1993, 23(1): 1～14

6 Mundie D A, Fisher D A. Parallel Processing in Ada. Computer, 1986(Aug.): 20～25

7 Schonberg E, Banner B. The GNAT Project: A GNU-Ada 9X Compiler. In: Proc. of Tri-Ada'94. Baltimore, Maryland, 1994

8 Geist A, et al. PVM 3 User's Guide and Reference Manual. Oak Ridge National Laboratory, May 1994

（上接第 62 页）

我们只需对其进行标准化。标准化后的语言必须能够以时间函数的方式修改实体的状态；必须具有原语，以独立于平台的方式与特定平台的数据结构进行交互（最可能的方式是通过一浏览器应用层）；还必须易于实现。

结语 实体及其行为是虚拟现实研究的重要议题之一。它对于日渐兴起的分布式虚拟现实研究更具有重要的意义。分布式虚拟现实系统涉及了大量种类各异、需要进行实时交互的实体，而实时交互则要求在实体之间高效地传递有关实体"行为"的信息。因此，深入研究实体及其"行为"可以帮助我们合理利用有限的系统及网络资源，建立优化的系统模型和消息机制。

参 考 文 献

1 汪成为、高文、王行仁．灵境（虚拟现实）技术的理论、实现及应用．清华大学出版社，广西科学技术出版社

2 张明治，等．虚拟生物的知觉构架．计算机学报，1996, 19(9 月增刊): 90～98

3 Stytz R M. Distributed Virtual Environments. IEEE Computer Graphics and Application, 1996(May): 19～31