

元计算系统的研究现状与趋势^{*}

The History and Future of the Research of Metasystem

桂小林 钱德沛

(西安交通大学计算机科学与技术系 西安710049)

Abstract A metasystem is a relatively new research field, its notation was firstly provided in 1992. To this day, some experiments system or prototypes are developed in oversea, including Globus, Legion, and Globe etc. In this article, the characteristics of metasystem are summed up, the study tendency about metacomputing is described, and the way to the metasystem based on Internet is also presented.

Keywords Metacomputing, Metasystem, Heterogeneous system, Study tendency

元计算系统(metasystem)^[1,2]是可以作为虚拟的整体而使用的地理上分散的异构计算资源,这些资源包括通过高速网络连接的异构计算机、数据库、科学仪器、文件和超计算系统等。元计算^[2]系统在硬件和软件等方面均有异构特性,适合具有不同内在并行性的复杂应用的执行。现存的绝大多数并行系统都是同构的,不具有这一优势。因此,研究异构的并行系统很有现实意义。国外在元计算系统方面开展了实质性的研究工作,美国的研究领先于其它国家。十年前开始预研,1992年^[2]正式提出概念设计,现在元计算已经成长为高性能计算的一个新的研究热点,并有不同设计目标的原型或实验系统产生。而在国内,元计算的研究处于初研阶段,和国外相比还有较大差距,还无可用的系统问世。

1 元计算的研究背景和意义

自从第一台电子计算机在1946年问世以来,计算机科学技术已经发生了巨大的变化。元器件的发展,无疑是过去几十年来推动计算机技术发展的重要动力,也是最活跃、最具影响的因素。尽管元器件制造技术在不断改进,集成电路的设计方法不断更新,但在未来十年内,元器件的速度不可能继续以以往的速度继续提高^[3]。据 SIA 预测,目前集成电路制造的主导技术—CMOS 技术,在2010年左右可能达到极限^[1]。当然,使用非 CMOS 制造技术,如量子技术、神经网络技术等,可以将元器件的速度再提升到一个新的台阶。但是,如果没有理论与技术上的重大突破,要提高元器件的速度最终将受到光速和氢原子直径的限制。而另一方面,

人类对计算机系统性能的需求永无止境,应用领域需要计算机解决的问题越来越复杂,规模越来越大。并行处理技术正是基于上述需要而提出的,它是提高计算机系统性能的关键技术。过去40年来,科学家提出了很多并行处理技术并设计了各种计算机系统,如 SMP 和 MPP 等,它们都极大地提高了计算机系统的整体速度。但由于 SMP、MPP 的处理器往往是同构的,而应用任务可能包含复杂的内在并行性(如 SIMD、MIMD 和向量等),不同的并行性可能适合不同的处理机来运行。在 SMP 和 MPP 系统中,大部分应用程序仅能获得其峰值性能的一小部分^[3]。传统的同构计算系统通常在一个给定的机器上使用一种并行编程模型,不能满足多于一种并行性的应用需求。任何单一类型的机器经常花费大部分的时间在不适合其执行的代码上。研究和开发支持多种内在并行应用的计算系统是摆在我们面前的重大挑战,其目的是提高计算效率,获取应用程序峰值性能的大部分。异构计算系统^[4]支持具有多内在并行性的应用,它可以提高应用程序实际执行性能与其峰值性能的比。一个异构系统通常包括若干异构的计算节点、互连的高速网络、通信接口以及编程环境等。异构处理^[4]可分为粗粒度异构处理和细粒度异构处理。细粒度异构是按指令分配和调度的,通信开销太大;而粗粒度异构是按指令块进行调度的,通信开销可以容忍,是具有开发前景的一种异构处理。粗粒度的异构处理又称为元计算,支持元计算的系统称为元计算系统,简称元系统。

Internet 作为流行的互联网络,具有分布范围广、传输速度相对较快等特点。在互联网环境下研究元

^{*} 国家863高技术计划资助项目(863-306-ZT01-06-2),西安交通大学博士论文基金资助项目。桂小林 讲师,博士生、研究兴趣为高性能计算、网络安全与容错。钱德沛 教授,博士生导师。

计算系统,具有重要的现实意义。元计算环境一方面使人们能聚集分散的计算能力,形成超级计算能力,解决诸如虚拟核爆炸、新药研制、气象预报和环境等重大科学研究和技术应用领域的问题;另一方面使人们共享广域网络中的异构资源,使各种资源得以充分利用。我国对高性能计算环境有强烈需求,这不仅表现在高端科学计算方面,同时表现在各行业对高端信息服务的需求方面。

2 元计算系统的研究现状

近年来,国外在异构计算环境方面开展了实质性的研究工作,美国的研究工作大大领先于其它国家。十年前开始预研,1992年正式提出概念设计,现在异构计算已经成长为高性能计算的一个新的研究热点。美国各政府部门启动了几个与元计算有关的重大研究发展计划。其中包括:(1)美国 NSF 的 PACI 计划(Grid 计划)由美国自然科学基金会支持,是80年代的超级计算中心项目的继续,其目标是试验未来10~15年美国社会的信息基础设施。PACI 计划支持了两个计算联盟,即 Illinois 大学超级计算机应用中心牵头的 NCSA 和圣地亚哥大学超级计算机中心(UCSD)牵头的 NPACI,其下各连100多个结点。(2)美国能源部的 ASCI 计划是为核武器储存和虚拟核爆研究而发展的计划。Los Alamos, Sandia 和 Livermore 三个国家实验室作为该计划的超级结点,其目标是至2004年达到每秒100万亿次浮点数的计算能力,以满足全系统、全物理、全三维的模拟需要。(3)美国国防部高性能计算现代化 HPCMP 网络是美国国防部建立的高性能计算的基础设施,供全国各地4000多名军事科技人员面向国防的研究和开发使用。该网络包括4个主计算中心和13个分中心,数十个远程中心。大部分计算能力集成在主中心。四个主中心于1998年共装备了23套计算机系统,速度从20亿次到5000亿次不等。该基础设施大大改变了美国国防科学研究的面貌。

经过近几年的研究,元计算在国外也有一定的基础,有不同设计目标的原型或实验系统产生。I-Way^[5]和 Globus^[6]是比较典型的元计算实验系统,它们的目标是为远程用户提供可视化的访问高性能计算平台的接口。其高性能计算平台由现有的美国若干国家实验室和大学的并行计算系统通过高速网络连接。Globus 是元计算系统的基础构件,提供可供用户使用的 C++ 库扩充函数,支持远程文件、远程 I/O、资源管理和 Nexus 通信管理等服务。Globus 通过协同操作和在一个广域分布式环境中扩展执行程序有效地利用了现存高性能计算系统,但随着服务需求的增加,缺少统一编程接口和模型使用户使用 Globus 变得相对困难。而

Legion^[7]系统强调工作站簇(或机群)上的并行任务的高效运行。Legion 使用 C++ 的扩充性语言 Mentat 实现,支持基于论证的用户安全服务、面向对象的虚拟文件服务和 MMPS 通信服务。Globe^[8]是在荷兰 Vrije 大学发展起来的元计算系统设计计划。它和 Globus 和 Legion 有许多相同的设计目标。Globe 是运行在现有主操作系统和通用网络协议之上的中间件系统, Globus 未提供有力的信息安全服务和用户安全论证服务。在国内,高性能计算方面的研究也有相当基础,建立了国家高性能计算中心,但在元计算方面的研究,还处于初研阶段,和国外还有较大差距,尚无可用的系统问世。

3 元计算系统的研究趋势与目标

元计算系统作为分布式和并行计算的支持平台,具有分布式环境和并行计算的许多共同特点。现有分布式环境和并行系统的设计技术为研究元计算环境提供了强有力的基础。并行计算模型的研究,并行执行环境、并行编译技术的成熟,通用并行编程环境 MPI、PVM 的使用,为开发元计算提供了有力保证。分布式文件系统和网络文件系统的研究,特别是 OSF/DCE 以及 CORBA 的研究,为研究元计算系统提供了有力的指导和借鉴。但元计算的设计原理和它们不同。目前成为工业标准的 DCE 和 CORBA 也是分布式计算工具,但它们只支持分布式计算,不支持并行计算;它们使用客户机/服务器模型,而没有使用并行计算模型,它们主要是为商务开发的,而不是为了进行高性能计算的。而元计算环境则不同,它具有分布式计算与并行计算两方面的好处,既可用于商务开发,又可用来进行高性能计算。

元计算系统的研究将趋向基于 Internet 的广域异构环境,并支持成千上万的属于不同管理域的计算机节点参与计算和资源共享。在未来,开发一个性能优良的元计算系统将遵循如下的设计目标:

- 提供单一映象空间:用户通过单一映象空间透明地操作和使用元计算环境。单一映象空间屏蔽硬件的边界,实现异构数据格式的转换,提供虚拟文件服务,把分散在各地的主机节点映射成一个统一入口的虚拟机器。

- 保证用户站点自治:元计算的节点是 Internet 网络上的异构节点,参与元系统的这些节点可能属于不同的管理域。为了保证节点拥有者的合法权益,元计算必须保证各个节点的自主权。在设计元计算系统时,要考虑到每个节点的最少安装,实现瘦型服务是十分重要的,也是系统实现扩展的关键所在。

- 提供强有力安全服务:由于元系统的节点具有分散性、流动性,每类节点拥有的安全功能有所差别,因

此,对元系统的节点管理就更显困难。在设计中,每个节点的安全性必须得到加强,至少不能破坏或降低节点原有的安全特性。

·支持高效容错功能:设计元计算系统应该考虑系统的容错功能、系统例外处理功能和软件调试与排错功能。

·支持统一资源管理:由于异构环境的节点各式各样,元系统必须支持多种体系结构,在元系统中,最好没有中心节点,整个系统是全分布的。通过中间件来保证异构硬件和软件的互操作性,进行资源的统一管理和调度。

·支持应用程序在元系统上合理调度:在元系统中,除了考虑指派任务给机器之外,还要考虑机器的异构特性,实现任务有选择的映射和调度,一个应用程序分解的子任务间往往存在依赖关系,依赖任务在异构机器上的调度是元系统设计的重点。

结论 元计算是一个相对较新的研究领域,开发一个高效、好用的元计算系统还存在一定的困难。一个元计算系统必须和同构系统一样,具有友好的人机交互界面、统一的编程环境、可靠的通信协议和高效的任务调度算法,并提供与流行的并行编码软件如PVM的接口,开发一个基于校园网络的元计算系统是我们的第一研究目标。

参考文献

1 Grimshaw A, Ferrari A, et al. Metasystems. Communica-

tions of the ACM, 1998, 41(11): 46~55

- 2 Smarr L, Cadlett C E. Metacomputing. Communications of the ACM, 1992, 35(6): 44~52
- 3 Messina P, Culler D, et al. Architecture. Communications of the ACM, 1998, 41(11): 36~45
- 4 Freund R F. Heterogeneous Processing. Computer, 1993, 26(6): 13~17
- 5 DeFanti T A, Foster I, et al. Overview of the I-WAY: Wide area visual supercomputing. International Journal of Supercomputer Applications and High Performance Computing, 1996, 10(2/3): 123~130
- 6 Foster I, Kesselman C. Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications and High Performance Computing, 1997, 11(2): 115~128
- 7 Grimshaw A S, Wulf W. The Legion Vision of a Worldwide Virtual Computer. Communications of the ACM, 1997, 40(1): 39~45
- 8 Steen M, Homburg P, et al. The Architectural Design of Globe: A Wide-Area Distributed System. [Internal report IR-422]. Vrije Universiteit, March, 1997
- 9 Foster I, Kesselman C, et al. Nexus: Runtime Support for Task-Parallel Programming Languages. Argonne National Laboratories. <http://www.mcs.anl.gov/nexus/paper/>
- 10 Grimshaw A S. Easy to Use Object-Oriented Parallel Programming with Mentat. IEEE Computer, May, 1993: 39~51

(上接第101)

助测试和调试人员分析错误结果和原因,为进一步修改和完善软件提供技术支持。

结束语 基于切片技术的软件测试是一种全新的软件测试技术,它把对整个程序的测试转化为只对某个程序切片的测试。与传统的测试方法相比,它具有效率高、准确性好等优点。基于切片的测试用例设计与切片准则有关,测试模块内切片也就是进行单元测试,测试用例的设计只与该模块有关;测试模块间切片也就是进行组装测试,测试用例的设计必须考虑到模块之间的联系;测试软件体系结构切片也就是进行确认方面的测试,测试用例的设计须符合需求分析的要求;接口测试(系统测试)可通过计算接口切片的方法进行相关测试,当然是通过接口描述语言规约切片以后进行测试。程序切片可用于错误查找,定位。

参考文献

1 朱鸿, 金陵紫. 软件测试和质量保障技术. 科学出版社, 1997

- 2 Podgurski A, Clarke L. A formal model of program dependence and its implications for software testing, debugging, and maintenance. IEEE Transactions on Software Engineering, 1990, 16(9): 965~979
- 3 Harman M, Danicic S. Using program slicing to simplify testing. Software Testing, Verification and Reliability, 1995, 5: 143~162
- 4 Kamkar M. Interprocedural Dynamic Slicing with Applications to Debugging and Testing. [PhD thesis]. Linkoping University. S-581 83 Linkoping, Sweden, 1993
- 5 Kamkar M, Fritzson P, Shahmehri N. Interprocedural dynamic slicing applied to interprocedural data flow testing. In: Proc. of the Conf. on Software Maintenance -93, 1993: 386~395
- 6 Gupta R, Harrold M J, Sofia M L. An approach to regression testing using slicing. In: Proc. of the IEEE Conf. on Software Maintenance, 1992: 299~308
- 7 Bates S, Horwitz S. Incremental program testing using program dependence graphs. In: Conf. Record of the Twentieth ACM Symposium on Principles of Programming Languages, ACM, 1993