大规模并行计算机系统并行性能模拟技术研究

徐传福 车永刚 王正华

(国防科技大学计算机学院 长沙 410073)

(国防科技大学并行与分布处理国家重点实验室 长沙 410073)

摘 要 性能模拟技术是计算机系统性能评价的重要手段。介绍了面向大规模并行计算机系统以及消息传递应用程序的并行性能模拟技术,总结了相关的关键技术和国内外研究现状。对几个代表性的并行模拟器系统进行了详细介绍。结合并行计算机系统和应用的发展趋势,讨论了未来并行模拟器设计、实现面临的问题和可能的解决方案。

关键词 并行模拟,并行计算机,消息传递应用程序

中图法分类号 TP302 **文献标识码** A

Research on Parallel Performance Simulation of Large Scale Parallel Computer

XU Chuan-fu CHE Yong-gang WANG Zheng-hua

(School of Computer, National University of Defense Technology, Changsha 410073, China)

(National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha 410073, China)

Abstract Simulation is an important tool for performance evaluation of computer system. This paper focused on parallel performance simulation of large scale parallel computer systems and message passing applications. Firstly we summarized some key techniques and state of the art in the research area, then we gave a detailed introduction about several representative parallel simulators, and based on the trend of parallel computer system and its application we discussed some challenging problems and useful suggestions for further research of parallel simulator in the end.

Keywords Parallel simulation, Parallel computer, Message passing application

为追求更高性能,人们正建造包含成千上万处理器核的超大规模并行计算机系统,例如2008年上半年公布的TOP500[17]中排在第一位的IBM Roadrunner便包含122400个计算内核。如何理解并行应用在这类极大规模机器上的性能及其可扩展性,已成为当前计算机系统性能评测领域迫切需要解决的问题。性能模拟与分析模型、测试被认为是计算机系统性能评测的三大主要方法[1],而其中基于软件的性能模拟技术因为灵活高效等优点,日益受到学术界和产业界的关注。研究人员通常面向特定类型的目标体系结构和目标应用,采用高级编程语言实现模拟器软件系统,通过在宿主机平台上运行该系统,预测给定程序输入条件下目标应用在目标体系结构上的性能。当前,多数模拟器系统以串行方式实现,其主要缺点是模拟的速度太慢,通常比实际测试的执行速度慢4~5个数量级[22]。串行模拟器的性能问题严重限制了其所能模拟的目标系统和应用的规模。为此,人们近年来研究了各种高效的性能模拟加速技术[3],并行模拟便是其中重要的一种技术。它将模拟任务并行化,充分利用并行宿主机平台提供的大量资源加快模拟速度。当前,并行模拟器系统多面向并行目标体系结构及其应用,这是由于一方面并行目标系统及其应用的模拟需要更强大的计算能力和内存,在串行宿主机平台上模拟大规模并行目标系统及其应用在实际性能上是不可行的。另一方面,并行模拟可更充分发掘、利用目标并行系统和应用本身的并行性,获得较好的并行模拟性能加速比。由于消息传递接口(Message Passing Interface, MPI)已成为当前大规模并行计算机系统的编程标准,因此本文重点关注可在并行宿主机平台上模拟大规模并行体系结构及消息传递应用程序的并行模拟器。

并行模拟器具有重要的应用意义:在目标机器建造之前,体系结构研究人员可根据并行模拟器的性能预测结果调整、优化系统设计,并行算法设计者能够通过并行模拟器获得算法在问题规模和机器结构扩展时的性能数据;即使目标机器已经存在,用户仍可采用并行模拟器对并行应用进行灵活方便的离线调试、优化,缩减并行应用的性能调试周期。20世纪90年代以来,国外对并行模拟技术进行了广泛深入的研究,已经出现了若干各具特色的并行模拟器系统并获得了成功应用。国内在并行模拟方面的研究基本上始于2000年以后,主要集中在中科院计算所、江南计算所、国防科技大学等并行计算机系统研制单位,由于起步较晚,至今仍停留在跟踪探索阶段。

本文首先给出了并行模拟器的总体结构,总结了其关键技术及国内外研究现状,然后选择3个并行模拟器系统LAPSE[20],MPI-SIM[9,10]以及BigSim[15,18,19]进行了详细介

到稿日期:2008-10-28 返修日期:2009-01-21 本文受国家863项目(2007AA01Z116),国家自然科学基金(60603055)资助。

徐传福(1980-),男,博士生,助理研究员,主要研究方向为系统性能评测等,E-mail:xuchuanfu@nudt.edu.cn;车永刚(1973-),男,博士,副研究员,主要研究方向为系统性能评测等;王正华(1962-),男,博士,教授,博士生导师,主要研究方向为系统性能评测等。

绍，最后结合并行计算机及其应用的发展趋势，讨论了未来并行模拟器设计、实现面临的问题和可能的解决方案。

1 并行模拟器总体结构

目前，并行模拟器多采用并行离散事件仿真（Parallel Discrete Event Simulation，PDES）[8]的方式实现，并行模拟器本身可以看作是PDES在计算机系统性能评测领域的具体应用。我们总结了基于PDES的并行模拟器的总体结构，参见图1。为模拟并行目标机器上的消息传递应用程序，目标应用程序的每个进程在目标计算内核上的运行由一个逻辑进程（Logic Process，LP）表示，每个LP维护自己的模拟时钟、事件队列以及相关的目标系统和应用的状态信息，LP通过处理具有时间戳的事件更新模拟时钟及状态信息。消息传递应用程序模拟中的事件可以分为两类：本地事件和消息通信事件。本地事件对应于目标应用程序中一些本地计算代码块的执行；消息通信事件对应于目标应用程序中消息通信语句（例如发送、接收消息的语句）的执行。与串行模拟器相比，并行模拟器的一个重要区别是各并行模拟进程同时推进模拟。因此，为保证结果的正确性和精度，LP之间需要进行时间同步。同时，为预测上述两类事件在目标并行机器上的执行时间，需要对并行目标体系结构部件进行性能建模。此外，并行模拟器在具体实现上也采用了一些与串行模拟不同的技术。下面从同步协议、部件性能模型及并行模拟器实现等3方面总结其关键技术及国内外研究现状。

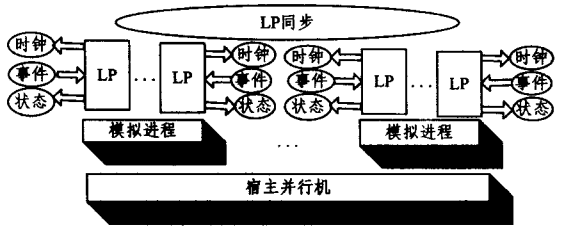

图1 基于PDES的并行模拟器总体结构

2 关键技术

2.1 并行模拟同步协议

并行模拟中各LP可能接收、处理来自其他LP的事件。为保证事件以其全局时间戳序被各LP处理，需要对并行运行的各LP进行同步。同步协议在PDES中已有广泛深入的研究[8]，目前基本上可以分为两大类：保守同步和乐观同步。

保守同步当且仅当LP能够保证事件“安全”的时候才允许处理该事件，所谓具有时间戳 t 的事件是安全的，指的是不可能有时间戳早于 t 的事件到达该LP。为确定事件是否“安全”，LP之间需要进行全局同步。目前广泛应用于并行模拟器中的保守同步协议称为quantum协议[5]。quantum协议要求每个LP周期性地以预先确定的时间间隔（称为模拟quantum）推进其模拟，只能处理时间戳小于当前quantum结束时的模拟时间的事件。每个quantum结束时各LP之间进行全局同步，Quantum的大小通常小于目标体系结构的通信延迟。其他保守协议包括空消息（Null Message）协议[5]、条件事件（Conditional Event）协议[6]、加速空消息（Accelerated Null Message）协议[7]等。保守协议的缺点是LP之间的全局同步开销较大，未能充分利用模拟中可用的并行性，LP可能会经常阻塞，优点是实现简单。多数并行模拟器（例如LAPSE和MPI-SIM等）采用了保守同步，具体实现中并行模拟器通常利用对目标应用程序的特征分析信息对保守协议进行了性能优化，重点考虑如何增加quantum大小、减少不必要的同步次数等。这种性能优化的效果与具体目标应用是密切相关的，某些情况下可以完全避免不必要的LP全局同步。

乐观同步允许LP不考虑安全性便执行事件队列中时间戳最早的事件。此时，LP需要周期性地运行检查点（checkpoint）操作，保存自己的模拟状态。若出现时间戳顺序错误（例如LP接收到时间戳小于已处理的事件的时间戳的事件），模拟会回滚到早期的检查点，重新执行被回滚的事件。比较著名的乐观同步方法包括时间偏差（Time Warp）[12]方法和周期时间桶（breathing time bucket）[13]方法等。乐观同步方法的开销在于检查点存储、错误回滚以及重新执行被回滚的事件。乐观同步能够充分利用事件执行的并行性，但实现较为困难，已有的并行模拟器中只有BigSim采用了乐观同步方法。BigSim充分利用目标程序内部的确定性对乐观同步进行了性能优化，模拟中仅在必要时对事件时间戳进行重新修正，无需进行实际的事件回滚和重新执行操作。

2.2 部件性能模型

应用代码在目标机器上执行时间的预测涉及到对目标机器部件的性能建模。通常性能模型越复杂，预测的精度越高，但同时模拟开销也越大。目前，针对大规模并行机器及消息传递应用，多数并行模拟器仅考虑了计算部件和互联网络部件等关键部件的性能模型，分别针对本地代码块和消息通信代码预测其执行时间。

1）本地代码块执行时间的预测：为预测本地代码块的执行时间，最精确的方法是在目标系统的时钟精确模拟器上运行该代码。但由于模拟速度太慢，多数并行模拟器并不支持时钟精确模拟。目前多数并行模拟器均采用本地代码块在宿主机平台上执行的墙上时间乘以适当的缩放因子（scale factor），预测其在目标机平台上的执行时间。这种方法的优点是实现简单，尤其适合在较小配置规模机器上预测同一类型的较大配置规模机器；缺点是确定缩放因子的理论依据不够充分，通常仅仅依据CPU主频速度之比。但由于多进程运行时的切换、OS调度以及Cache行为等各种因素的影响，这种方法本身的精确性难以保证。

其他方法还包括：1）针对每个本地代码块提供一个执行时间预测值，该预测值可以通过各种方法获取，因而其优点是简单灵活，但缺点是需要用户手工修改目标应用，在代码块前后增加对时间预测API的调用。2）基于宿主机硬件性能计数器统计代码块中各种类型指令的数量，然后结合各种启发式方法预测这些指令操作在目标机器上的执行时间。这种方法可基于统计的指令类型进行性能模型的扩展，例如可根据访存指令信息和简单地存储层次性能模型预测目标应用的访存性能，但要求宿主机平台支持硬件性能计数器及其访问接口库。

2）消息通信时间的预测：消息通信时间的预测需要基于一个互联网络模型。目前多数并行模拟器采用简单网络性能模型，忽略了网络竞争，通信时间由延迟、开销、带宽及消息大小等决定。并行模拟器也可以实现详细的竞争网络模型，其中可以包括对构成并行系统内部互联网络的基本部件（如交

换机、通道、网卡等)的性能建模以及消息数据包的路由策略、网络拓扑等算法、结构的实现。详细的互联网络模型有助于深入研究应用随互联网络参数变化时的性能特征。

2.3 模拟器实现技术

早期的模拟器在实现上多采用 trace 驱动方式[22],目标是在功能模拟器上执行产生的 trace 文件,作为性能模拟器的输入。这种方式的优点是灵活简单,缺点是 trace 文件的存储、加载开销较大,且仅记录了应用实际执行的路径,无法反映推断执行等特征。20 世纪 90 年代以后,执行驱动的模拟器[22]逐渐成为主流。执行驱动又分为解释执行和直接执行两种方式。解释执行逐条解释、模拟应用中的所有指令,当前很多时钟精确模拟器便采用了这种方式。其缺点是速度慢,难以扩展到大规模目标系统的模拟。一般仅用于处理器、Cache 等关键体系结构部件的模拟;直接执行模拟器不模拟每条应用指令,仅对感兴趣的部分体系结构特征进行建模。例如,就并行体系结构模拟而言,用户可能并不关心计算部件,但对新型存储子系统、互联网络等特征很关注,此时用户便可以直接利用宿主机处理器资源执行计算代码,仅仅模拟尚不存在的存储或网络子系统行为。直接执行驱动的模拟实现方式有效地利用了宿主机提供的可用资源,避免了模拟用户并不关心的部分,效率较高,其缺点是要求宿主机平台和目标机平台相同,或者至少比较接近。

已有的并行模拟器通常采用了直接执行的实现方式。采用直接执行模拟消息传递应用程序时,并行模拟器通常包括一个 MPI 接口模拟库,本地代码由模拟器在宿主机平台上通过直接执行的方式模拟,而消息通信代码由 MPI 接口模拟库处理,应用代码与并行模拟器一起编译、链接,应用的编写、执行方式基本不变。模拟中,应用的控制逻辑在应用和模拟器代码之间进行切换,代码中涉及到的对目标体系结构信息的访问将被模拟器捕获(trap)进行处理,由当时的模拟状态确定返回结果。图 2 给出了直接执行驱动的并行模拟器的工作流程。

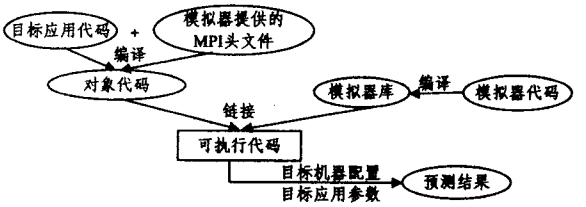

图 2 直接执行驱动的并行模拟器工作流程

此外,由于目标机器的计算内核数远多于宿主机,因此每个宿主机计算内核通常采用多控制流的方式模拟多个 LP。常见的实现机制包括进程、内核线程、用户线程等,三者的主要区别在于性能开销,包括创建、切换的开销以及占用的内存空间开销等。通常而言,进程开销最大,内核线程次之,用户线程开销最小。LP 的实现机制对并行模拟器的整体性能影响很大。一方面,很多 OS 限制允许创建的进程、内核线程的数量,这从理论上限定了并行模拟器在特定宿主机平台上所能模拟的目标机器规模;另一方面,开销太大(尤其是内存开销),导致在小规模并行宿主机上模拟大规模目标机器从性能上不再可行。

3 评价指标和典型系统介绍

3.1 评价指标

设计和实现模拟器时,通常需要权衡精度、性能和灵活性 3 个目标。具体到并行模拟器而言,这 3 个目标的含义及其测试验证方法又有所不同。

精度:为验证并行模拟器的精度,通常在一个或多个已有的并行机平台上运行目标应用程序,测得实际执行时间 T_{actu},以上述机器平台作为目标系统配置模拟运行目标应用程序,获得预测执行时间 T_{pred},则并行模拟器的精度以 T_{actu} 与 T_{pred} 的误差百分比衡量,可通过式(1)计算。并行模拟器分析、设计、实现等各阶段均可能引入产生误差的因素,通常模拟模型越详细,精度越高,但精度可能随不同的具体的目标应用和目标系统配置而差别较大,目前,多数并行模拟器精度误差在 10%以内。

$$|T_{pred}-T_{actu}|/T_{actu}\times 100\% \quad (1)$$

性能:通常从速度及其可扩展性方面衡量并行模拟器的性能。加速比是常用的指标。设并行模拟器在单个处理器上串行模拟给定目标系统配置下的目标应用时所需的时间为 T_1,采用 N 个处理器进行并行模拟所需的时间为 $T(N)$,则并行模拟器加速比 $S(N)$ 定义为

$$S(N)=T_1/T(N)$$

此外,并行模拟器在给定宿主机平台上所能模拟的最大目标系统规模也是衡量其可扩展性的重要指标。多数情况下,并行模拟器的加速比很难达到线性,效率不高。但对于某些受限于内存的目标应用程序的并行模拟,也可能会获得超线性加速比[19]。

灵活性:并行模拟器的灵活性通常指的是支持的目标系统的可配置性和宿主机平台的可移植性等功能性指标,难以量化。当前,并行模拟器通常仅能支持少量甚至单一类型的目标系统配置,且很多执行驱动并行模拟器在实现上依赖于并行宿主机平台提供的特定功能[21],从而导致灵活性较差。

3.2 典型系统介绍

20 世纪 90 年代以来陆续出现了不少针对并行体系结构的模拟器,其中多数是只支持串行模拟的系统,如 SIMOS[16],GEMS[4]等;也有支持并行模拟的系统,如 Wisconsin Wind Tunnel[21],Augmint[11]等。这些模拟器的目标体系结构多为共享存储多处理器系统,且难以扩展到对大规模并行系统的模拟。本文重点关注支持大规模并行目标体系结构及消息传递应用程序的并行模拟器,这里选取了 LAPSE,MPI-SIM 和 BigSim 等 3 个系统进行详细介绍。

3.2.1 LAPSE(Large Application Parallel Simulation Environment)

LAPSE 是 NASA 90 年代资助的项目,目标是实现 Intel Paragon 平台上消息传递应用代码的可扩展性分析和性能分析。LAPSE 实现时 MPI 尚未成为消息传递应用的编程标准,它模拟的是基于 Intel 的消息传递库 nx[20]开发的目标应用。LAPSE 以直接执行方式实现,支持 C,Fortran 或两者混合代码应用,并行模拟采用称为 Whoa(Window based Halting On Appointments)的保守同步方法。Whoa 基于对目标应用的运行时分析确定模拟 quantum 的大小,较 quantum 协议有一定的性能改进。LAPSE 最大的缺点是宿主机平台和目标机平台均为 Intel Paragon,可移植性差。此外,LAPASE 采用 OSF-1 Unix 线程实现逻辑进程,OS 调度的切换开销较大,这导致它能够实际模拟的目标机器规模有限,性能可扩展

性不强。测试结果表明多数情况下 LAPASE 的精度误差小于 10%，采用 64 处理器时能够获得较高的加速比。

3.2.2 MPI-SIM

MPI-SIM 是加州大学 COMPASS（COMponent-based PArallel System Simulator）[10]项目的一部分，该项目包含了 MPI 通信库模拟器 MPI-SIM、并行 I/O 模拟器 PIO-SIM 以及并行文件系统模拟器 PFS-SIM 等几个组件。其中 MPI-SIM 提供了大部分常用的 MPI 通信接口函数的模拟，可以模拟未经修改的 MPI 应用程序。COMPASS 是目前唯一提到支持并行 I/O 和并行文件系统应用模拟的项目，因而它支持的目标应用不仅包括计算密集型应用，还包括采用 MPI-I/O 实现的 I/O 密集型应用，但目前仅能从其网站中找到 MPI-SIM 的代码实现和相关参考文档。MPI-SIM 采用直接执行的实现方式，并行模拟支持一系列保守同步协议。设计 MPI-SIM 时强调支持大规模目标应用程序的模拟和可移植性，目前支持的宿主机平台包括 Intel Paragon，SGI Origin 2000 和 IBM SP 等。测试结果表明，MPI-SIM 在 16 宿主机平台上可获得 3.2 到 11.9 的加速比。

3.2.3 BigSim

BigSim 来源于伊利诺伊大学 Urbana-Champaign 分校为 BlueGene/C 开发的模拟器，因此最初又称为 BlueGene 模拟器。BigSim 包括两个独立运行的部分：一个并行功能仿真器和一个详细的并行网络模拟器。目标应用程序首先在功能仿真器上运行，产生 trace，网络模拟器以该 trace 为输入，对互联网络进行详细的性能模拟。尽管 BigSim 最初以 BlueGene 为目标机器，但目前 BigSim 支持的是一个三维结点拓扑结构的通用机器功能模型，用户可以配置每维的结点数及每个结点内部包含的处理器数。详细的网络性能模拟是 BigSim 的一大特色，目前支持的互联网络模型包括 blue gene 网络、red storm 网络等。在实现上，BigSim 采用切换开销很小的轻量级用户线程 CthThread[14]模拟目标进程，因而性能可扩展性较好，在 96 个处理器的 ASCI RED 机器上可模拟具有上百万处理器规模的目标系统，这是目前已知的并行模拟器所能支持的最大规模目标系统。测试结果表明，BigSim 在 256 节点的宿主机上获得了 150 的加速比，精度误差在 6%左右。

4 发展方向

尽管人们从 20 世纪 90 年代以来便对并行模拟技术进行了深入的研究，且出现了不少各具特色的并行模拟器系统，但并未出现像微处理器模拟器中的 SimpleScalar[2]那样被广泛应用的系统。这一方面是由于并行模拟器的目标体系结构、应用和宿主机平台定位为并行平台环境带来的限制；另一方面，并行模拟器系统本身也面临一些尚需要完善的地方。

首先，现有并行模拟器的可扩展性普遍较差，主要表现在功能可扩展性和性能可扩展性两方面。就功能可扩展性而言，目前多数并行模拟器仅支持单一类型目标机器和其中部分关键体系结构部件的模拟，模拟模型的可配置性差，难以适应未来并行体系结构朝着多态、混合体系结构发展的趋势。未来并行模拟器不仅需要全面支持各种已有的体系结构部件，还要重点解决如何方便地加入对新型体系结构部件（如专用 CPU 加速部件等）的模拟支持，可以研究如何从并行模拟器软件体系结构的设计和实现上提供这种功能可扩展性支持。就性能可扩展性而言，尽管并行模拟器从理论上不限制所模拟的目标系统和应用的规模，但由于巨大的计算、内存等开销，使得在当前较小规模的宿主机并行平台上模拟未来新一代具有多达成千上万个处理器的并行计算系统从性能上变得实际不可行。未来一方面可以研究如何有效充分利用更大规模宿主机平台，提高模拟速度；另一方面可以考虑从具体实现技术以及同步协议等方面研究如何对并行模拟器进行性能优化。

其次，相对于时钟精确的微体系结构模拟器而言，并行模拟器的精度仍然较差，且受具体的目标应用和同步协议等因素的影响较大。这一方面是并行应用本身的不确定性造成的，另一方面针对并行模拟器系统本身的精度验证工作也不够深入全面。未来一方面可以研究如何在并行模拟器中采用更加精确的部件性能模型，同时采用标准的并行 benchmark 及真实应用对模拟器进行更全面的精度验证，尽量减少由于模拟器设计、实现而导致的误差。

尽管并行模拟技术还有很多需要完善的地方，但并行模拟作为一种有效提升模拟性能的技术手段日益受到关注。尤其是当前计算机系统及其构成越来越复杂，串行模拟的性能无法满足研究人员充分探索设计空间的需求，而多核、集群等多种并行计算环境的日益普及也为并行模拟技术的发展和应用提供了更好的可供选择的运行平台。因此，并行模拟技术仍然具有广阔的发展前景。

参 考 文 献

[1] Jain R. The Art of Computer Systems Performance analysis [M]. Wiley, 1991

[2] Austin T, Larson E. SimpleScalar: An infrastructure for computer system modeling[J]. IEEE Computer, 2002, 35(2): 56-67

[3] Yi J, Kodakara S. Characterizing and comparing prevailing simulation techniques[C]// Proc of the 11th int'l Symp on High-Performance Computer Architecture (HPCA-11 2005). 2005

[4] Martin M, Sorin D. Multifacet's general execution-driven multiprocessor simulator (gems) tool set[N]. Computer Architecture News, 2005

[5] Misra J. Distributed Discrete Event Simulation[J]. ACM Computing Surveys, 1986, 18(1): 39-65

[6] Chandy K M, Sherman R. The conditional event approach to distributed simulation[C]// Distributed Simulation Conference. Miami, 1989

[7] Bagrodia R, Meyer R, Takai M, et al. Parsec: A Parallel Simulation Environment for Complex Systems[J]. Computer, 1998, 31(10): 77-85

[8] Fujimoto R M. Parallel discrete event simulation[J]. Commun. ACM, 1990, 33(10): 30-53

[9] Bagrodia R, Deelman E, Docy S. Performance Prediction of Large Parallel Applications Using Parallel Simulations[C]// ACM SIGPLAN Symposiumon Principles and Practice of Parallel Programming (PPOPP). May 1999

[10] Prakash S, Bagrodia R L. Mpi-sim: Using parallel simulation to evaluate mpi programs[C]// Proceedings of IEEE Winter Simulation Conference. 1998

[11] Sharma A, Nguyen A-T, Torrellas J. Augmint: a multiprocessor simulation environment for intel x86 architectures[R]. Urbana-Champaign: Center for Supercomputing Research and Development, University of Illinois, 1996

（下转第 35 页）

$$K \geqslant \begin{cases} \dfrac{\ln(\alpha)-\ln(1+\rho\alpha-\rho)}{\ln(\rho)}, & \rho \neq 1 \\ \dfrac{1-\alpha}{\alpha}, & \rho = 1 \end{cases} \quad (7)$$

5 模型验证

使用 NS-2[8]仿真软件对 DOCSIS 上行信道做了仿真，其主要仿真参数[6]如表 2 所列。

表 2 仿真参数

参数	取值	参数	取值
W_0	2	L_r	16Bytes
m'	7	L_d	500Bytes
m(最大回退次数减 1)	15	n	10
N_c(平均请求时隙个数)	32	λ	10,20

首先验证通过本模型中接入不同数量的 CM 与 CM 能达到的平均服务速率的关系。计算结果根据式(1)、式(2)、式(5)得出，仿真结果通过 NS-2 仿真软件得出。除 CM 的个数外的参数设置如表 2 所列。计算结果与仿真结果的对比如图 7 所示。

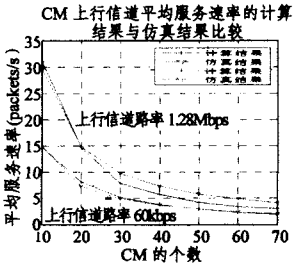

图 7 CM 数量与上行信道平均服务速率关系

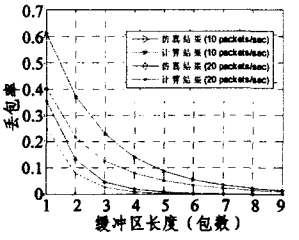

图 8 缓冲区溢出概率

从图中可以得知，本算法能够较为准确地估计上行通道的平均服务速率，而这种平均服务速率的计算精度受 CM 的个数的影响较小。计算结果与仿真结果一致，证明了平均服务速率计算模型是能够合理地反映 DOCSIS 上行信道特性的。

接下来，将对于缓冲区的大小计算结果进行验证。对平均到达速率是 10 packets/sec 及 20 packets/sec 的模型的缓冲区大小与丢包概率的关系进行了验证。通过仿真结果(如图 8 所示)可以看出，我们的算法可以较为准确地反映 DOCSIS 上行信道中缓冲区大小和丢包率之间的关系，尤其是能够反映较低丢包率下对缓冲区大小的要求，具有合理的应用价值。

结束语 本文基于 DOCSIS 规范，分析了 CM(Cable Modem)的上行带宽分配的竞争请求机制，提出了上行信道的马尔可夫链模型，以此为基础建立了 CM 端的上行信道排队模型，给出了其发送缓冲区大小的估计方法，并给出了本文模型的准确性验证。通过 NS-2 仿真验证，本文方法可以较为准确地估计上行缓冲区的大小与溢出概率的关系，从而能够为上行缓冲区的大小设置提供理论参考依据。

参 考 文 献

[1] ETSI [S]. EN 300429 V1.2.1: Digital Video Broadcasting (DVB)—Framing structure, channel coding and modulation for cable systems, April 1998

[2] Reede I, Brandt M, Karaoguz J, et al. IEEE Project 802.14/a draft 3 revision 3: Cable-TV access method and physical layer specification[S]. Unapproved IEEE 802.14 Project Draft Specification. Oct. 1998

[3] CableLabs, Data-Over-Cable Service Interface Specifications Radio Frequency Interface Specification[S]. CM-SP-RFIv1.1-C01-050907. September 2005

[4] Wu H, Peng Y, Long K, et al. Performance of reliable transport protocol over IEEE 802.11 wireless LAN: Analysis and enhancement. IEEE Infocom'2002. New York, June 2002

[5] 唐应辉，唐小我. 排队论——基础与分析技术[M]. 北京：科学出版社，2006：59

[6] Murphy R. A Simulation study of DOCSIS Upstream Channel Bandwidth Allocation Strategies for Minimal User Response Time[D]. San Antonio: University of Texas, Dec. 2004

[7] 孙晓东，冯振明，陆明泉. HFC 网络上行信道的 MAC 层协议性能分析[J]. 电子学报，2002(2)：187-190

[8] http://www.isi.edu/nsnam/ns/

(上接第 10 页)

[12] Jefferson D, Beckman B, Wieland F, et al. Time warp operating system[C]//Proceedings of the 11th ACM Symposium on Operating System Principles. 1987：77-93

[13] Steinman J S. Breathing time warp[C]//Proceedings of the 7th Workshop on Parallel and Distributed Simulation. ACM Press, 1993：109-118

[14] Saboo N, Singla A K, Unger J M, et al. Emulating petaflops machines and blue gene[C]//Workshop on Massively Parallel Processing (IPDPS'01). San Francisco, CA, 2001

[15] Zheng Gengbin, Kakulapati G, Kal'e L V. BigSim: A parallel simulator for performance prediction of extremely large parallel machines[C]//18th International Parallel and Distributed Processing Symposium (IPDPS). Santa Fe, New Mexico, April 2004

[16] Rosenblum M, Herrod S A, Witchel E, et al. Complete computer system simulation: The SimOS approach[J]. IEEE Parallel and Distributed Technology: Systems and Applications, 1995, 3(4): 34-43

[17] Top 500supercomputersite[EB/OL]. http://www.top500.org/, 2008

[18] Zheng Gengbin, Wilmarth T, Jagadishprasad P, et al. Simulation-based performance prediction for large parallel machines[J]. International Journal of Parallel Programming, 2005, 33：183-207

[19] Zheng Gengbin. Achieving high performance on extremely large parallel machines: performance prediction and load balancing [D]. Urbana-Champaign: Department of Computer Science, University of Illinois, 2005

[20] Dickens P M, Heidelberger P, Nicol D M. A distributed memory lapse: parallel simulation of message-passing programs[J]. SIGSIM Simul. Dig., 1994, 24(1)：32-38

[21] Reinhardt S K, Hill M D, Larus J R, et al. The wisconsin wind tunnel: Virtual prototyping of parallel computers[M]. Measurement and Modeling of Computer Systems, 1993：48-60

[22] 喻之斌，金海，邹南海. 计算机体系结构软件模拟技术研究[J]. 软件学报，2008，19(4)：1051-1068