ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.4, April 2008, pp.1036−1050
DOI: 10.3724/SP.J.1001.2008.01036

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

高性能交换与调度仿真平台的设计与实现*

扈红超+, 伊 鹏, 郭云飞

(国家数字交换系统工程技术研究中心,河南 郑州 450002)

Design and Implementation of High Performance Simulation Platform for Switching and Scheduling

HU Hong-Chao+, YI Peng, GUO Yun-Fei

(National Digital Switching System Engineering and Technological R&D Center, Zhengzhou 450002, China)

+ Corresponding author: Phn: +86-371-63532677, Fax: +86-371-63941700, E-mail: huhongchao@gmail.com, http://www.ndsc.com.cn

Hu HC, Yi P, Guo YF. Design and implementation of high performance simulation platform for switching and scheduling. *Journal of Software*, 2008,19(4):1036−1050. http://www.jos.org.cn/1000-9825/19/1036.htm

Abstract: Simulation has become a significant way for performance evaluations in switching and scheduling, however, the existing simulation softwares have some limitations in inheritability and extensibility. Based on the current crossbar switching fabric, by employing system level design method, and object oriented technology, a simulation platform called SPES (switching performance evaluation system) for switching fabrics and scheduling policies' developments are designed and implemented. Input queuing, output queuing, combined input-output queuing and combined input-crosspoint queuing and corresponding scheduling policies are integrated. Inheritability and extensibility attributes are obtained by designing traffic sources, switching fabrics and scheduling policies separately, and it exhibits good performances for supporting multi-fabric, variable packets sizes and QoS model's simulations. By configuring the platform through a uniform view, users can fulfill their concrete simulation environment. Besides, it can carry out end to end performances' evaluations with little modification. Finally, this paper presents a simulation case based on combined input-crosspoint queuing switch, displaying the good performance of SPES.

Key words: queuing mechanism; management policy; object oriented; simulation platform; switching system

摘 要: 仿真实验已成为交换结构和调度策略性能评价的重要手段,而目前存在的交换结构与调度策略的仿真软件在可继承性与可扩展性方面还存在缺陷.基于Crossbar交换结构,建立数学模型,引入系统级设计方法,采用面向对象技术,设计并实现了用于研究交换结构和调度策略的仿真平台——SPES(switching performance evaluation system).该平台集成了输入排队、输出排队、联合输入输出排队、联合输入交叉点排队等多种交换结构以及相应调度策略.设计上实现了业务流、交换结构和调度策略三者之间的分离,具有良好的可继承、可扩展特性.用户通过与仿真平台之间的简单交互,完成模块的添加与仿真环境参数的配置,在支持变长业务、区分服务质量模型和多交换平面仿真方面具有良好的特性.通过简单扩展,该平台还可以实现网络级性能仿真.最后给出了基于该平台,在

* Supported by the National Natural Science Foundation of China under Grant No.60572042 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2005AA121210 (国家高技术研究发展计划(863))

Received 2006-06-30; Accepted 2006-10-10

CICQ(combined input and crosspoint queuing)交换结构下,对所提出的支持 DiffServ 模型的分布式调度策略 DS(DiffServ supporting algorithm)在不同业务流模型下的性能测试结果,并与输入、输出排队交换结构进行了比较,展示了 DS 良好的性能,验证了仿真平台的合理性.

关键词: 排队机制;管理策略;面向对象;仿真平台;交换系统

中图法分类号: TP393 **文献标识码:** A

目前,通信网络呈现两方面的发展态势:一方面,在骨干传输网上,光传输线路代替了传统的电传输线路,网络核心交换节点容量从 G 比特级向 T 比特级转型,10Gbps 乃至更高速率的商用线路接口技术稳步出现;另一方面,边沿网承载业务类型向着多元化方向发展,尤其是 IPTV 等多媒体业务将在下一代互联网(next generation network,简称 NGN)中扮演重要角色.因此,作为核心组网结点的交换机/路由器在交换容量上将面临新的挑战,交换结构与调度策略将成为互联网发展的瓶颈,寻求适合新形势发展需求的交换结构与调度策略成为亟待解决的问题.

然而,近年来,由于交换技术的发展,交换系统的行为变得十分复杂,单纯的理论分析很难得出系统模型的解析解,因此,目前业界主要通过仿真实验手段对交换技术的公平性和有效性进行评价,理论分析手段主要用于交换技术复杂性的评价以及其他性能指标的定性分析与粗略评估.由于能够更客观、更直观、更准确地描述系统行为,仿真实验已成为交换技术性能评价的重要手段,对设计方案的决策具有十分重要的意义.

本文提出一种系统级描述语言建模、面向对象技术实现的交换结构与调度策略的仿真平台——SPES(switching performance evaluation system),由于模块间采用“解耦合”的设计思路,使得 SPES 具备了最大限度的灵活性与可扩展特性.目前,该平台兼容现有的各种交换结构与调度策略,并为新的结构扩展与调度策略的设计预留了接口,用户通过与仿真平台之间的简单交互,完成模块的添加与仿真环境参数的配置,专用性、可扩展性强,为交换调度领域的技术创新和方案设计可提供一个重要的基础平台.

文章第 1 节简要介绍仿真平台背景与相关研究.第 2 节从总体上介绍该仿真平台的设计思路及系统抽象模型.第 3 节详细介绍该仿真平台的设计与实现及如何扩展到支持变长分组、QoS 以及多交换平面.第 4 节介绍在我们的仿真平台中集成的性能评估指标,并验证了该仿真平台的合理性.最后总结全文.

1 相关研究

根据交换机/路由器物理交换结构上构建的逻辑排队方式的不同,基于Crossbar(交叉开关)构建的交换结构主要可分为输出排队(output queuing,简称OQ)交换结构(如图 1(a)所示)、输入排队(input queuing,简称IQ)交换结构(如图 1(b)所示)以及联合输入输出排队(combine input and output queuing,简称CIOQ)交换结构(如图 1(c)所示).随着CMOS(complementary metal oxide semiconductor)工艺水平的提高,在交换单元内部存储一定容量的缓存成为可能[1],并已成为该领域的研究热点,其中最具影响力的为联合输入/交叉节点排队(combined input and crosspoint queuing,简称CICQ)交换结构[2](如图 1(d)所示).调度策略是以交换结构为基础构建的逻辑管理机制,是交换系统性能的重要保证.交换结构与调度策略共同决定整个交换系统的性能.

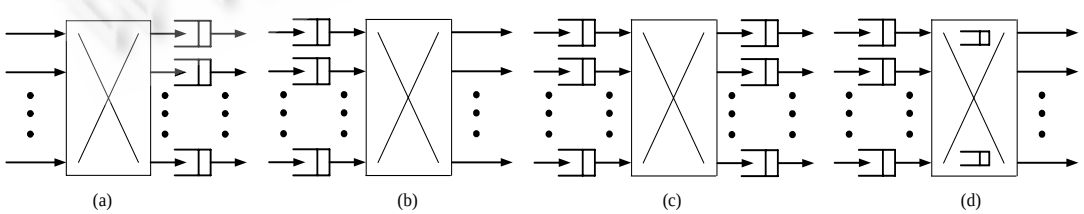

Fig.1 OQ, IQ, CIOQ and CICQ switch architectures

图 1 OQ,IQ,CIOQ 和 CICQ 交换结构

从工程实现的层面上讲,核心交换机/路由器的交换体系是一个复杂的分布式软、硬件实时系统,为了解决

工程需求与实现之间的差距,节约成本和缩短工程周期,业界的一般做法是根据功能需求提出解决方案,并通过理论分析和仿真实验对方案进行性能评估,进而决定其可实施性.然而,单纯的理论分析很难得出系统模型的解析解,因此,仿真验证就成为性能评估的关键环节.目前,国内外主要采用的网络仿真软件有 NS-II,OPNET 等,它们均包含了非常丰富的模块,几乎涉及到了网络技术的所有方面.这些大而全的网络仿真软件对于网络技术的研究和发展的确起到了很大的推动和促进作用,但是,它们的优势更多地体现在对高层协议的支持方面,在交换技术研究方面仅提供了少量简单的基本功能模块,而且由于牵涉到的内容过于庞杂,因此应用较为复杂.另外,这些网络仿真软件由于未能将经典的交换结构和调度算法系统地集成到软件平台,在研究新型交换技术时缺乏统一的参考标准和比较对象,虽然一些研究人员发布的软件包提供了部分典型调度算法,但设计思路和实现机制因人而异,因此在可继承性与可扩展性方面仍然存在缺陷.

然而,这些仿真软件的设计与实现思路值得我们借鉴,尤其是 NS-2,其模块化的设计思路及其代码的开放性使得我们可以在很多模块的实现上具有重要的参考价值.从对模块之间进行“解耦合”的角度出发,我们最大限度地保持了仿真平台 SPES 各个模块的独立性,模块之间通过接口调用相互访问,并为新的扩展预留了接口,灵活性、可扩展性强.目前,在我们的平台下集成了基于 Crossbar 的输出排队、输入排队、联合输入输出排队和联合输入交叉点排队交换结构以及相应的调度策略,用户可以通过统一的交互界面配置自己的仿真环境,如图 2 所示.

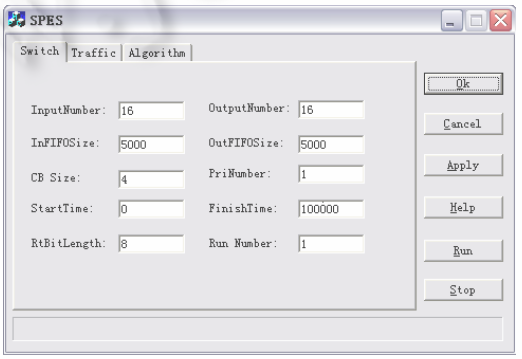

Fig.2 System level configuration view

图 2 系统交互视图

2 系统分析与设计

2.1 系统分析

在交换系统中,数据流按照一定的接纳控制策略进入输入端口,经输入端口预处理(如分类、整形、排队等)后按照预定的调度策略传送到交换单元,最后由交换单元交换到输出端口.为了方便描述交换系统,我们首先给出如下定义:

定义 1. 用来构建交换系统的可选模块将其定义为子系统.

定义 2. 在交换系统中,数据的处理可能由多个交换单元并行进行,我们将由交换单元及其排队方式和调度策略完成的逻辑功能实现称为交换平面.

虽然交换系统采用的交换结构和调度机制各异,但抽象到模型的层面上,它们又存在着一些共性,如它们都是由输入端口、交换单元、输出端口和调度策略有机结合构成,其差异主要体现在这些组成部件的具体实现机制上,如是否设置缓存、缓存排队机制、调度机制和队列管理策略等.为了设计出具有良好可扩展性的交换技术性能仿真评价系统,本节首先从共性层面将不同的交换系统抽象为一个公共的模型,它由输入端子系统、交换单元子系统、输出端子系统和调度子系统组成.在交换系统模型的基础上添加业务源子系统和仿真控制子系统,即可构成交换系统性能仿真系统的结构模型,如图 3 所示.其中,虚线所围模块为子系统,包括输入子系统、

Crossbar 交换单元子系统和输出子系统以及调度策略子系统,它们相互组合构成不同类型的交换平面;由交换平面进而构建更为复杂的交换系统(如并行分组交换系统).子系统除具备自身的功能之外,还向外提供接口,接口提供了子系统之间相互访问交互的途径,以数据流驱动的方式协调工作,完成整个交换系统功能.

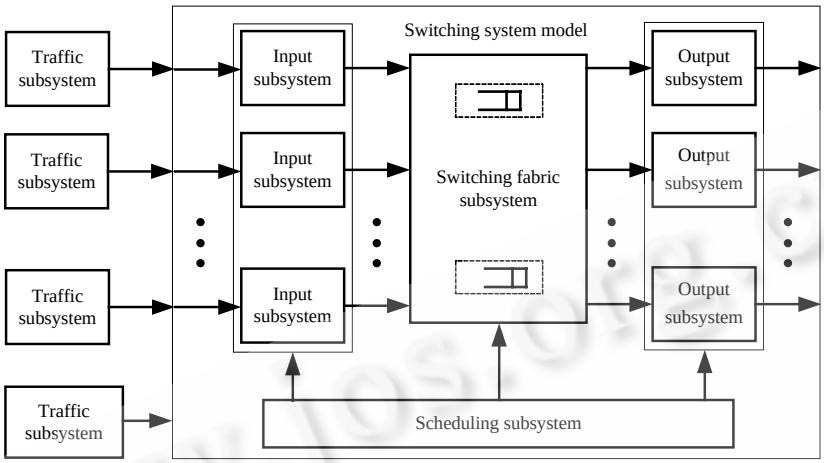

Fig.3 Architecture model for simulation system

图 3 仿真系统的结构模型

2.2 面向对象设计

SPES 系统结构模型中的每个子系统具有相对独立的功能和接口,这一特性非常适合采用面向对象的方法学设计.子系统采用面向对象技术设计,由不同的类和对象封装而成,并以基本类库的形式提供给系统.用户不仅可以借助基本类库搭建典型交换结构和调度算法的仿真验证环境,而且还可以通过面向对象的设计技术(继承等),在基本类库的基础上开发复杂的组件,对更为复杂的交换系统进行仿真验证.图 4 给出了 SPES 平台系统类的结构图,它反映了系统中类的层次结构关系.

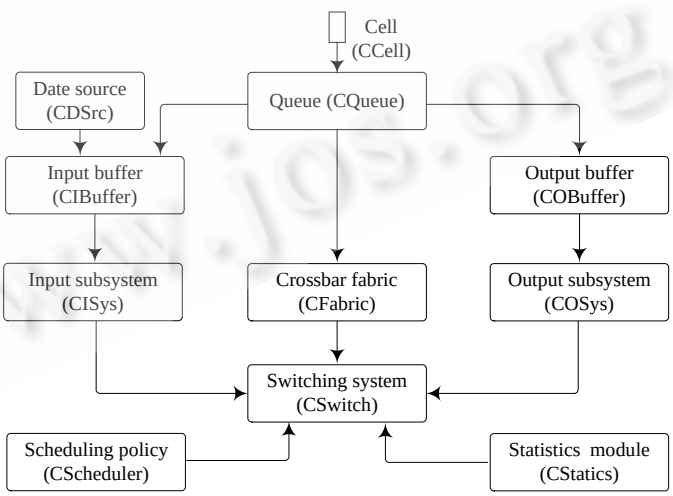

Fig.4 Switching system classes hierarchy view

图 4 交换系统类层次结构图

将图 3 所示系统中的各个模块进行细化,并采用类的对象进行表述,从而得到如图 5 所示的交换系统对象结构图.虚框为可选模块,与图 3 所述的结构相对应.从面向对象的角度来说,只需添加或者去除类的对象即可实

现模块的添加与去除,完成具体仿真环境的配置.在下面的几小节里,我们在假设分组交换系统是基于信元(cell)、时间被分割成时隙(slot)、一个时隙为单个信元的传输时间的条件下,详细介绍各个子系统的实现,包括各自的属性、方法及它们之间的交互机制.最后介绍如何将该仿真平台进行扩展,以支持变长分组交换、区分服务质量模型和多交换平面的分组交换系统的仿真,进而支持到网络级的分组交换系统仿真.

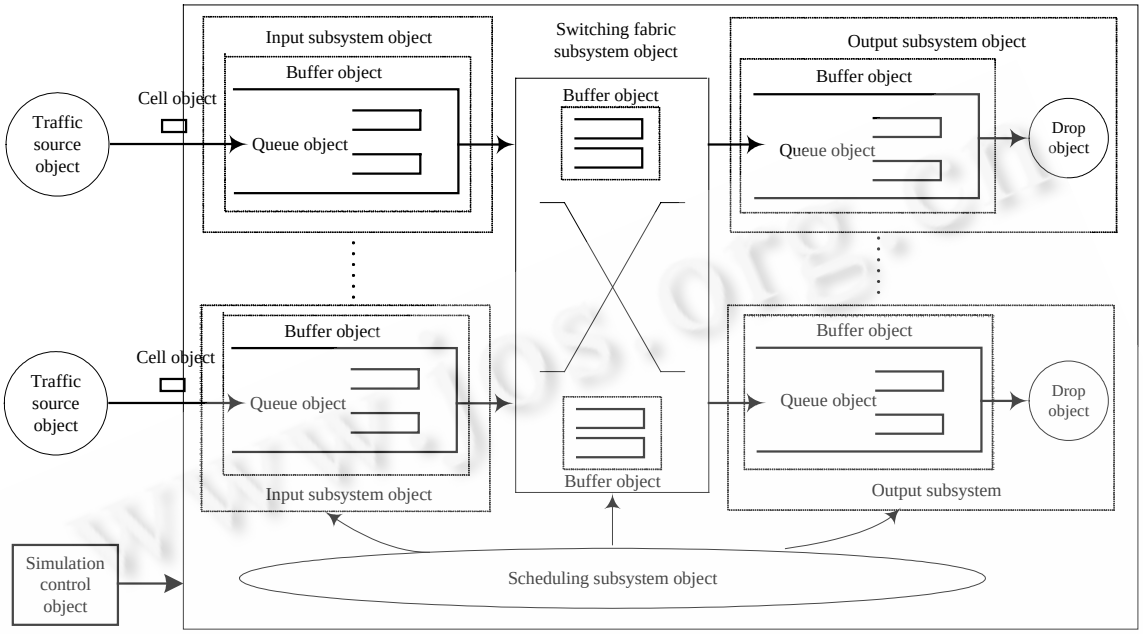

Fig.5 Switching system architecture object view

图 5 交换系统类对象结构视图

3 系统实现

3.1 业务流子系统

业务流的产生有两种方法:一种是利用现实存在的数据,比如从现实网络中提取数据分组并按照一定格式以文件形式记录下来,然后依照对应格式从文件中读取;另外一种是依据业务源特性建立业务源模型,按照模型产生分组.这里,我们主要介绍业务源建模产生业务流的方法.

用业务源建模的方法产生的业务流,应能刻画出现实网络中的业务流特性,按照NPF(network processing forum)的建议和文档[3],我们实现了Bernoulli业务流和突发业务流的建模,并对它们进行了封装设计,最终以接口的形式提供给系统调用.为了便于描述,我们用离散时间随机过程$A_i(t)$表示输入端i的到达过程,到达率为$\lambda_i(0\le\lambda_i\le1)$.$A_{i,j}(t)$为输入端口$i$到目的端口$j$的到达过程,到达率为$\lambda_{i,j}$,则到达过程的集合$A(t)=\{A_{i,j}(t),1\le i\le N,1\le j\le N\}$.若$\lambda_{i,j}$满足$\sum_i\lambda_{i,j}<1,\sum_j\lambda_{i,j}<1$,我们称$A(t)$为允许的,否则为非允许的.

3.1.1 Bernoulli 业务源的建模

Bernoulli业务源具有无记忆特性,在每个时隙内,分组的产生以一定的概率准则进行控制,相邻时隙内分组是否产生是独立的,我们按照IEEE标准实现了均匀随机数据源来生成[0,1]区间的随机数R_{num},并与到达率λ进行比较.如图 6 所示,若$R_{num}<\lambda$,分组产生;否则,分组不产生.因此,对于Bernoulli业务源模型,仅需一个控制到达强度的参数λ.

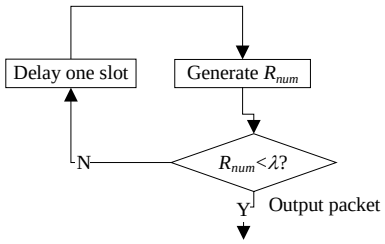

Fig.6 Flowchart for Bernoulli traffic generation

图 6 Bernoulli 业务流实现流程图

3.1.2 ON-OFF 突发业务源的建模

现实网络中的业务流在多数情况下是具有突发性的,因而简单的Bernoulli业务流模型已经不能刻画真实网络中的业务流特性.而马尔可夫调制的ON-OFF模型能够很好地刻画出这种突发性[3],ON,OFF之间的状态转移情况如图 7 所示.其中,p_1,p_2为概率转移参数.

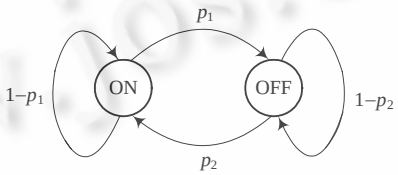

Fig.7 States conversion for two-states Markov chain

图 7 两状态 Markov 链状态转移图

业务流仅在ON时期内产生,ON,OFF的时间段服从几何分布.ON时间段内突发强度表示在一个ON周期内产生的分组数,用参数L_{on}表示,OFF时间段内突发强度表示在一个OFF周期内产生的空闲分组数,用L_{off}表示,ON时间段和OFF时间段几何分布参数用E_{on},E_{off}表示,则它们之间的关系满足公式:

$$E_{on}=L_{on}/(1+L_{on}),L_{off}=L_{on}\times(1-\lambda)/\lambda,E_{off}=L_{off}/(1+L_{off}).$$

3.1.3 分组对象的设计

将不同的业务源模型封装成业务源类,由属性和接口两部分构成.对于Bernoulli业务源,属性主要是指到达业务强度λ;对于ON-OFF模型,还有突发强度L_{on},L_{off}.向外部提供的接口主要包括属性的设置接口和分组的产生接口.系统在设置完业务源属性后,调用分组的产生接口来产生分组,分组格式见表 1.

Table 1 Packet format

表 1 分组结构图

Number of bits	32	32	32	32	16	16	16
Field name	Sequence	Birth time	Time label	Time label	Input port	Output port	Priority

其中,分组序号对产生的业务分组进行编号.时间标签是分组在子系统间传递时填充的.出端口号和优先级决定业务流负载的分配,目的端口的分布可以是均匀的或非均匀的.常用的均匀模型有Uniform业务流,非均匀模型有Unbalance业务流和Asymmetric业务流.对于一个$N\times N$的交换系统,假设i,j分别表示某输入端口号、出端口号,λ表示各个输入端口总业务强度,$\lambda_{i,j}$为输入端口i到达输出端口j的业务强度,则 3 种业务流模型[4]可以描述为:(1) Uniform业务流:$\lambda_{i,j}=\lambda(1-\omega/N)$;(2) Unbalanced业务流:如果$j=i$,$\lambda_{i,j}=\lambda(\omega+(1-\omega)/N)$),否则,$\lambda_{i,j}=\lambda(1-\omega)/N$; (3) Asymmetric业务流:如果$j=i$,$\lambda_{i,j}=0$,否则,$\lambda_{i,j}=\lambda(r^{|j-i|}-r^{|j-i-1|})/(r^{N-1}-1)$.

3.2 输入、输出子系统

从业务分组的处理层面上看,输入、输出子系统具有相似的功能:按照一定的接纳控制策略接收、缓存业务分组,并按照一定的调度策略将分组发送出去.在描述设计实现时,我们以输入子系统作为讨论对象.输入系

统是否预置缓存与具体的排队机制有关,这里讨论内部预置缓存的情形,对于不预置缓存的情形,只需去除对应类的对象即可.

3.2.1 缓存管理

在如图 3 所示的交换系统模型中,每个输入端口预留一定量的缓存,并采用了虚拟输出排队(virtual output queuing,简称 VOQ)的方式存储突发业务的分组.每个输入端口的缓存容量为 c(cells),如果采取缓存均匀分配的机制,则每个 VOQ 队列的容量为 c/N(cells),缓存容量的大小在一定程度上影响丢包率和分组时延.

输入端口将业务源产生的分组根据目的端口进行分类(classify)、接纳业务源分组并进行缓存以等待调度.缓存管理定义了队列的逻辑组织形式和分组的入队、出队机制.图 8 描述了输入端口的概念模型,可以看出,队列是构成了该模块组成的基本单元,包括入队机制和出队机制以及分组在队列中的组织形式等.

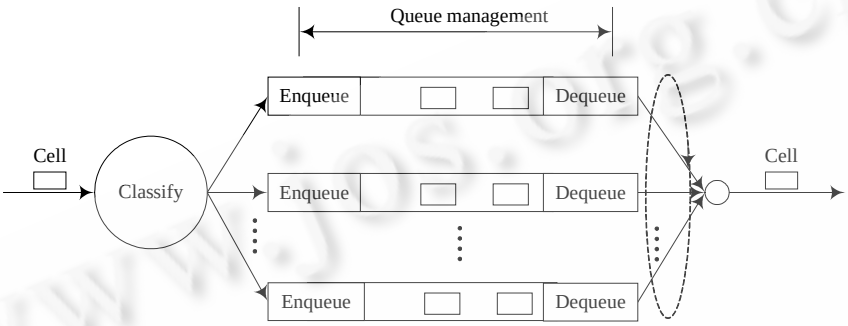

Fig.8 Input subsystems conceptual model

图 8 输入端口子系统概念模型

3.2.2 队列的封装

一个基本队列 q 由瞬时长度 $q(t)$、容量 C、接纳控制策略 P 以及队列管理方式 M 等来描述,即

$$q \xleftarrow{def} R^V, \quad V=[q(t),C,P,M]^T,$$

其中,R是定义在V上的四维向量空间.队列的主要功能是按照一定接纳控制策略接收业务分组,将分组按照队列管理方式M缓存起来,并按照调度策略的规定选择分组发送到交换平面.如图 9 所示的队列概念模型,接纳控制策略P是队列的管理层面,根据队列当前状态(如$q(t)$)等来决定是否允许分组进入队列,目前有RED(random early drop)[5],Drop Tail等;从队列管理方式M角度出发,队列可分为先进先出(first in first out,简称FIFO)队列、先进后出(last in first out,简称LIFO)队列等.一个队列类的实现应具备$q(t)$,C,P和M这些基本属性和方法,通过继承扩展进而得到具有更为复杂功能的队列.

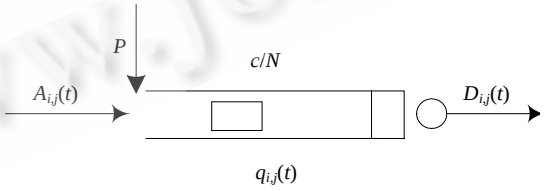

Fig.9 Queue conceptual model

图 9 队列概念模型

3.3 交换单元子系统

交换单元是交换系统的核心部件,目前在商业领域中广泛应用的是基于 Crossbar 的交换单元,描述交换单元的参数有输入端口个数 N、输出端口个数 M 和吞吐量 C 等.

基本功能的交换单元应具有接收、发送分组的功能,更为复杂的交换单元(如带缓存 Crossbar)还应具有调

度功能.交换单元从输入端口 i 接收分组,交换后送到输出端口 j,并且每次交换时,i 和 j 之间仅可以建立一个连接.假设某个时隙内共 m 个分组从交换单元调度到输出端口,则在该时隙内交换单元的利用率 η 定义为

$$\eta=m/M.$$

如果在每个交叉点预留缓存队列,该交换结构就演化为带缓存的 Crossbar 交换单元,共有 $N\times M$ 个交叉点子队列,队列的封装格式如第 3.2.2 节中的描述,从而将常规的 Crossbar 交换单元从逻辑上分割为 N 个 $M\times 1$ 和 M 个 $N\times 1$ 的子交换结构,分布式调度策略的实现成为可能,在 SPES 平台下我们将交换单元子系统封装为 CFabric 类,它可以方便地用于实现由多个交换单元扩展而构成的交换系统.

3.4 调度策略子系统

由于带缓存交叉开关是当前的研究热点,因此,本节主要介绍联合输入交叉节点交换结构其调度算法的设计准则.我们考虑一个$N\times N$的交换系统结构,如图 3 所示,数据分组以信元为单位,每个输入端口维护N个VOQ队列,因而输入端共有N^2个VOQ队列,对应N^2个交叉点队列.CICQ采用分级与分布式调度:输入调度时将VOQ队列中的分组传送到带缓存的Crossbar,输出调度时将Crossbar缓存中的分组输出.我们首先给出描述如下CICQ基本系统的若干定义:

定义 3. 集合$V=\{VOQ_{i,j}|0\le i\le N,0\le j\le N\}$,其中,$VOQ_{i,j}$缓存从$i$端口输入到$j$端口的数据分组,$q_{i,j}(t)$表示$t$时刻系统中$VOQ_{i,j}$的长度,$0\le t\le T$,其中,$T$为系统仿真时间,$0\le q_{i,j}(t)\le C_q$,$C_q$为VOQ队列容量.

定义 4. 集合$C=\{CB_{i,j}|0\le i\le N,0\le j\le N\}$,其中,$CB_{i,j}$对输入端口$i$到达输出端口$j$的分组进行二次缓存,与$VOQ_{i,j}$一一对应.用$l_{i,j}(t)$表示$t$时刻系统中$CB_{i,j}$队列长度,$0\le t\le T$,其中,$T$为系统仿真时间,$0\le l_{i,j}(t)\le C_l$,$C_l$为$CB_{i,j}$的容量.

定义 5. 在输入调度阶段,我们称$VOQ_{i,j}$在t时刻是“候选的(eligible)”,若满足:$0<q_{i,j}(t)\le C_q$,$0\le l_{i,j}(t)<C_l$,记为$E(VOQ_{i,j})$,则输入端口i候选VOQ的集合$E_{i=i}=\{E(VOQ_{i,j})|i=i,1\le j\le N\}$,或者$E_{i=i}=\{VOQ_{i,j}|0<q_{i,j}(t)\le C_q,0\le l_{i,j}(t)<C_l,i=i,1\le j\le N\}$.

定义 6. 在输出调度阶段,我们称$CB_{i,j}$在t时刻是“候选的”,若满足:$0<l_{i,j}(t)\le C_l$,记为$E(CB_{i,j})$,则输出端口j的“候选”CB集合$E'_{j=j}=\{E(CB_{i,j})|j=j,1\le i\le N\}$,或者,$E'_{j=j}=\{CB_{i,j}|0<l_{i,j}(t)\le C_l,j=j,1\le i\le N\}$.

CICQ交换结构采用的是分级、分布式调度策略,输入调度策略用f_{in}表示,f_{in}是$E=(E_1,E_2,\ldots,E_N)$的函数;输出调度策略用f_{out}表示,f_{out}是$E'=(E'_1,E'_2,\ldots,E'_N)$的函数,即$f_{in}(E)$,$f_{out}(E')$.$J$表示目标函数(如时延、丢包率等),则$J$为$f_{in}$,$f_{out}$的函数,用$J(f_{in},f_{out})$表示.在每个时隙内,输入$i$至多匹配(match)一个交叉点$CB_{i=i,j}$,输出$j$也至多可以和一个$CB'_{i,j=j}$匹配.用$x_{i,j}(t)$表示在$t$时刻、输入调度阶段输入$i$和输出$j$之间的匹配关系,$y_{i',j'}(t)$表示输出调度阶段$i'$和$j'$之间的匹配关系.那么,调度策略的数学模型可以描述为

$$\min_{f\in F}(\max)\ J(f)$$

st.

$$\begin{cases}\sum_j x_{i,j}=0,1\ \sum_{i'} y_{i',j'}=0,1\\ F=[f_{in}(E_1,E_2,\ldots,E_N),f_{out}(E'_1,E'_2,\ldots,E'_N)]\\ E'_j=\{CB_{i,j}\mid 0<l_{i,j}(t)\le C_l,1\le i\le N,1\le j\le N\}\\ E_i=\{VOQ_{i,j}\mid 0<q_{i,j}(t)\le C_q,0\le l_{i,j}(t)<C_l,1\le i\le N,1\le j\le N\}\end{cases}.$$

因此,调度策略的设计就是在一定时域内、在交换结构动态状态的约束下选择一种执行算法,使系统所追求的目标函数(吞吐量、平均时延等)的统计性能成为最优.

目前,SPES 集成的交换结构与管理策略见表 2.

Table 2 Switching architectures and scheduling policies integrated

表 2 目前集成交换结构与调度策略

Switching fabric	Queue management policy	Port scheduling policy
OQ	Red,Drop tai, etc.	Round robin based algorithms as RR, DRR[6], WRR[7], RRR[8]. GPS based algorithm as SFQ[9], WFQ[10,11], WF^{2}Q[12], WF^{2}Q+[13]
IQ	Red,Drop tail, etc.	Weighted algoritms as LQF[14], OCF[15], LPF[16], iLQF[17], iOCF[17], iLPF[16]. Non-weighted algorithms as iSLIP[18], WFA[19], PIM[20]
CIOQ	Red,Drop tail, etc.	MWM, MSM, etc.
CICQ	Red,Drop tail, etc.	RR-RR, OCF-OCF[21], LQF-RR[22], LQF-OCF[22], MCBF[23], SCBF[24], etc.

3.5 仿真时间控制

在业务源建模中,分组的产生控制是基于随机数的.因而单次运行结果存在偏差,需要多次运行获得相对稳定的仿真数据.

我们借助数理统计的方法来判决某次仿真的运行次数.用X表示仿真中待测试的某一性能指标,θ为X的数学期望,x_i为第i次仿真结果,$\bar{x}$表示n次仿真得到的X的均值,则

$$\bar{x}=\frac{1}{n}\sum_{i=1}^{n}x_i \tag{1}$$

且$\bar{x}$为θ的无偏估计量[25].为了确定n,我们给出MSE(mean square error)的定义.用σ表示$\bar{x}$的标准差,则MSE定义为$MSE=\sigma^2/n$,并用样本方差S^2作为σ^2的估计值,S^2为

$$S^2=\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n-1} \tag{2}$$

则

$$MSE=\frac{s^2}{n} \tag{3}$$

如果给出MSE的上限d^2,就可以通过迭代的方式得到仿真次数n.

(1) 设置参数,初始化 n=1;

(2) 运行SPES,并由式(1)~式(3)计算出MSE,同时与d^2比较,若$MSE\le d^2$,退出;否则,转(3);

(3) $n=n+1$,转(2).

3.6 扩展到支持变长分组、QoS、多交换单元

3.6.1 变长分组仿真

变长分组的预处理,目前普遍采用的是“切片”方式,切片为一个ATM(asynchronous transfer mode)信元的大小,即 64(bytes).用L_p(bytes)表示某一变长分组的长度,则经过切片后产生的信元个数n为$n=\lceil L_p/64\rceil$.

3.6.2 扩展到支持 QoS

QoS 的支持包括两个方面:一是交换结构,二是调度策略.两个方面具有一定的耦合性,交换结构不同,其具有 QoS 保证的调度策略的设计也有所不同.总体上来讲,实现 QoS 保证,时间复杂度和空间复杂度是衡量系统实现复杂性的两个重要指标,二者不可兼得.一般的解决办法是,在设计过程中将两者加以折衷,获得较好的 QoS 性能.在仿真平台的设计中,我们将交换结构与调度策略进行分离设计,迎合了时间复杂度与空间复杂度的关系.同时,时域加速或者是空域加速都能够轻易地在仿真平台上分离实现.

3.6.3 支持多交换单元

多交换单元是指在一个分组交换系统中交换单元呈现“三维”立体式交换网络,如图 10 所示.二维交换平面扩展为三维交换网络,这种交换结构需要在输出端预设缓存来实现业务流汇聚功能.由于我们在设计过程中将各子系统进行了面向对象的封装,从结构上讲,增加一个平面等同于增加一个类的对象;从调度策略的角度来说,对分组的输入调度需要额外考虑去往的平面编号参数,在输出端口对业务流进行汇聚,因而这种结构扩展在我们的平台上的实现也很简单.

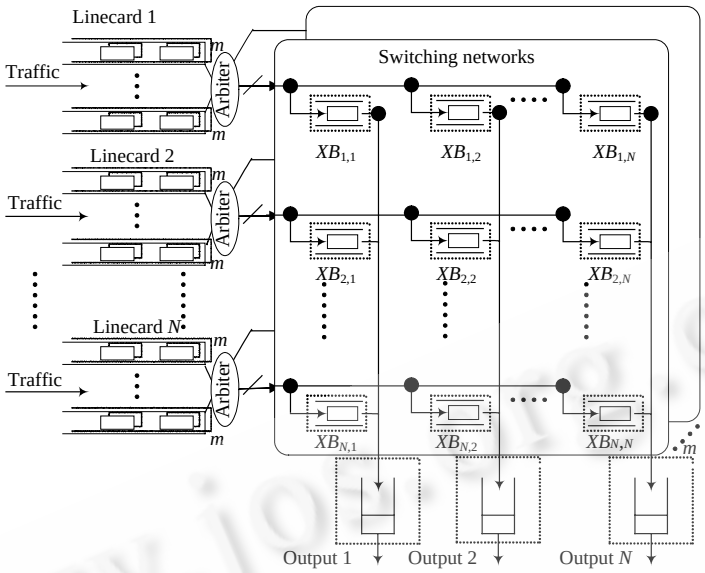

Fig.10 Multi-Crossbar fabric architecture

图 10 多交换平面分组系统

可以看出,由于采用了面向对象的技术进行模块化的设计,我们的仿真平台无论从支持变长分组交换、QoS,还是多交换单元方面都具有良好的可扩展性能,其灵活性为迅速开发新的交换结构与调度策略提供了重要保证.

4 性能指标与仿真验证

4.1 性能指标

SPES 系统可支持时延、时延抖动、稳定性、丢包率、吞吐量和带宽分配比等性能指标的获取,由于这些性能指标的获得都是基于分组统计得到的,所以,我们把各种性能指标封装为 CStatistics 类,并为新的性能指标的计算预留了接口.下面,我们介绍在仿真平台下各种性能指标的统计规则.

4.1.1 时 延

时延有分组平均时延、分组最大时延等.在一次仿真过程中,分组交换系统共交换的数据分组数为n,d_i表示第$i(0\le i\le n-1)$个分组的时延,那么分组平均时延(D_a)和分组最大时延(D_m)定义为

$$D_a=\frac{1}{n}\sum_{i=1}^{n}d_i,\quad D_m=\max(d_1,d_2,\ldots,d_n).$$

4.1.2 时延抖动

时延抖动是QoS保证的重要指标,用来描述相邻分组之间的时延变化程度,若用D_n表示n+1 个分组的时延抖动向量,那么D_n定义为$D_n=[d_1-d_0,d_2-d_1,\ldots,d_n-d_{n-1}]^T$.

4.1.3 丢包率

丢包率用来描述一定时间段内系统丢弃分组数目n_d与到达交换系统分组数目n_a之间的比值,即n_d/n_a.丢包率用来衡量系统接纳分组的能力.

4.1.4 吞吐量

吞吐量是指一个交换系统在没有丢包的前提下,允许接收分组的最大到达速率,是衡量一个交换系统交换能力的重要指标.

4.1.5 稳定性

稳定性采用队列的占有率来衡量,见文献[23].用$||L(n)||_x$表示在某服务策略x下、任意时刻n时所有VOQ的占有率,则$||L(n)||_x$定义为$\|L(n)\|_x=\sqrt{VOQ_{1,1}(n)^2+...+VOQ_{1,N}(n)^2+...+VOQ_{N,1}(n)^2+...+VOQ_{N,N}(n)^2}$.

若$||L(n)||<\infty$,我们称该交换结构在服务策略 x 是稳定的.稳定性是交换系统性能的重要指标,反映了调度策略的健壮(robust)性能.

4.1.6 带宽分配比

带宽分配比是衡量公平性的重要指标,用来描述不同优先级业务在一定时间内获得的归一化带宽.用n_p表示优先级为p的业务在n个时隙内调度输出的分组数,则优先级为p的业务获得的带宽分配比为n_p/n.

4.2 仿真验证

由于CICQ交换结构的优良性能并已成为业界研究的热点,因此本节基于CICQ交换结构,论述如何将基本的CICQ结构进行扩展以保证QoS策略,并提出支持DiffServ模型的分布式调度策略DS.同时与DDS(dynamic DiffServ scheduling)[26],PQWRR(priority queuing weighted round robin)[27]进行性能对比,表明了DS的优越性能,也验证了仿真平台的合理性.

4.2.1 结构扩展

DiffServ模型将不同的业务流映射为 6 类:EF(expedited forwarding),{$AF_i|1\le i\le 4$}(assured forwarding),BE(best effort)[28],并为每类提供不同的服务质量保证.

在交换系统输入端i,为了独立缓存 6 类不同的业务流,每个$VOQ_{i,j}$队列需要扩展为$P(P=6)$个子队列.$VOQ_{i,j,p}$缓存输入端口为i、输出端口为j、$p(1\le p\le 6)$类的业务分组.相应地,每个交叉点缓存$CB_{i,j}$也扩展为P个子队列,变为$CB_{i,j,p}$.扩展后的队列结构如图 11 所示.

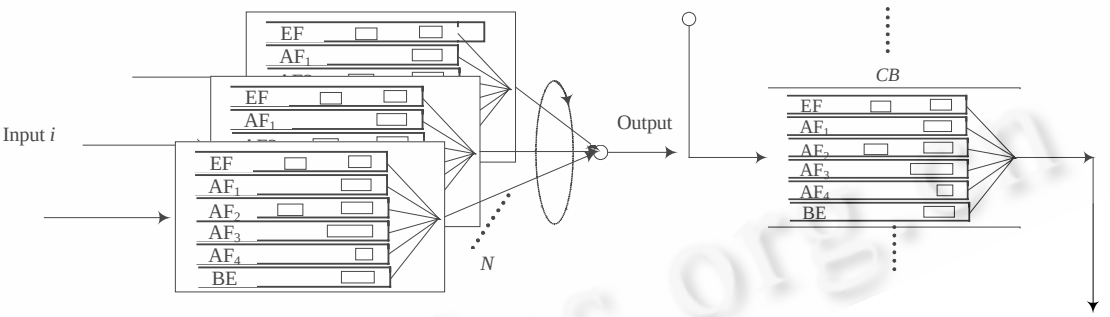

Fig.11 VOQ and CB structures after extension

图 11 扩展后 VOQ 与 CB 的结构

4.2.2 DS 调度策略

在带宽分配上,我们简单采用文献[26]中的动态带宽分配(dynamic bandwidth allocation)机制,将输出端口的带宽Bw分割为预留带宽和剩余带宽两部分.EF与{$AF_i|1\le i\le 4$}业务共享预留带宽,剩余带宽在{$AF_i|1\le i\le 4$}与BE之间分配.用$B_{w,p}$表示在任意输出端口j,为$p(1\le p\le 5)$类业务流预留的带宽,则$B_{w,p}$满足

$$\sum_{p=1}^{P}B_{w,p}=1,\sum_{p=1}^{P-1}B_{w,p}<1.$$

用$B_{j,p}(n)$表示在某输出端口j、时隙n,p类业务流获得的实际统计带宽.$B_{j,p}(n)$的计算规则如下:对$p=1$,$B_{j,1}(n)=c_{j,1}(n)/n$;对 $1<p\le 5$,$B_{j,p}(n)=c_{j,p}(n)/(n \bmod F)$.其中,$c_{j,1}(n)$表示截止到时隙$n$、输出端口$j$调度EF类的业务分组数;$c_{j,p}(n)(1<p\le 5)$表示在当前"帧"$k(k=\lceil n/F\rceil)$,内调度$p$类业务流的分组数,$F$为单位"帧"所占用的时隙数.

对第3.4节描述的模型进行扩展定义,$v_{i,j,p}(n)$用来指示$VOQ_{i,j,p}$在时隙n是否为空,并用0表示空;$f_{i,j,p}(n)$指示在时隙n时$CB_{i,j,p}$中的分组数.定义$E_{i,p}=\{VOQ_{i,j,p}|0<v_{i,j,p}(n),0\le f_{i,j,p}(n)<S,1\le j\le N\}$, $E'_{j,p}=\{CB_{i,j,p}|0<f_{i,j,p}(n)\le S,B_{j,p}(n)<B_{w,p},1\le i\le N\}$,则我们的调度策略DS描述如下:

F_{in}:对任意输入i,从p=1 开始迭代,直到找到某一$VOQ_{i,j,p}$,满足:{$VOQ_{i,j,p}\in E_{i,p}$}∩{$VOQ_{i,j,p}$|队头分组的出生时间标签在该类分组中是最小的}.

F_{out}:对任意输出j,从p=1(p<6)开始迭代,直到找到某一$CB_{i,j,p}$,满足:{$CB_{i,j,p}\in E'_{j,p}$}∩{$CB_{i,j,p}$|队头分组的出生时间标签在该类分组中是最小的},如果找到,返回;否则,在 $2\le p\le 6, 1\le i\le N$范围内搜索队头分组具有最小出生时间标签的$CB_{i,j,p}$.

4.2.3 仿真结果与验证

我们在均匀突发业务流下($\lambda_{i,j}=\lambda/N,\forall i,j$)与非均匀突发业务流下($\lambda_{i,i}=\lambda 2/3, \lambda_{i,(i+1)\bmod n}=\lambda/3$,并对于$j\ne i$或$|i+1|$,$\lambda_{i,j}=0$).利用SPES,对DS,DDS和PQWRR进行了性能测试.为了方便比较,我们将相关参数设置成与文献[26]中一样的情景,即EF和AF业务分组平均到达率为 18%,24%,20%,16%,12%,对应的预约带宽为 19.8%,24%,20%,16%,12%,带宽统计与时延测量分别在 4×4 与 16×16 交换结构下.

理论上讲,SPES只需很低的主机配置即可以实现仿真.然而,仿真实验需要反复进行才能得到相对稳定的统计结果.为了保证仿真所能接受的时间忍受度,我们推荐的最低主机配置为CPU 1GHz,512M内存.在配置为Pentium(R) 4 CPU 2.60GHz,1.00G内存的主机上运行我们的仿真软件,单次仿真周期为 10^6时隙,计算机的运行时间为 30s,应用程序占用内存为 1M.

图 12 为 DS,DDS 和 PQWRR 的带宽分配曲线图.可以看出,DS 具有与 DDS 相似的带宽分配性能,远优于 PQWRR,但是 DS 采用的是并行与分布式的调度策略,因而复杂度比 DDS 要低,硬件上更易实现.

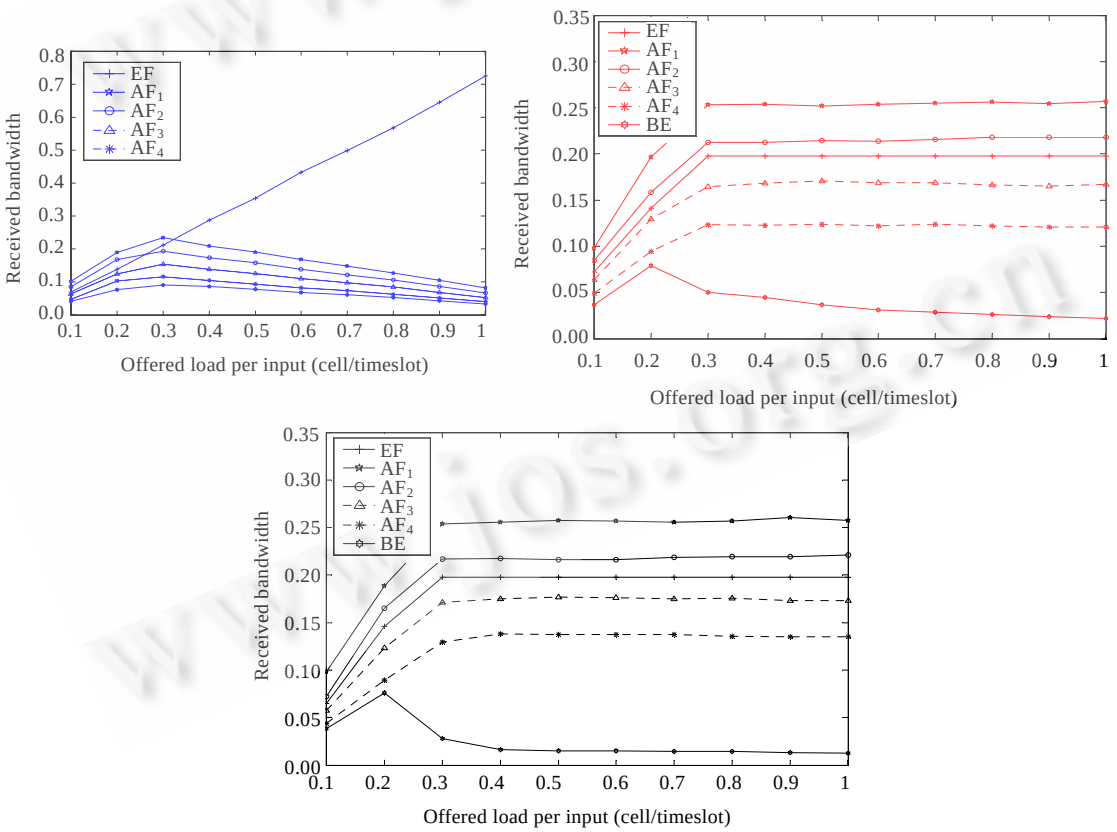

Fig.12 Bandwidth allocation for PQWRR, DDS and DS

图 12 PQWRR,DDS,DS 带宽分配对比图

图 13~图 15 描述了 PQWRR,DDS 和 DS 时延性能曲线.可以看出,DS 的 EF 业务时延性能与 DDS 十分接近,

比 PQWRR 略差,而 DS 调度策略的 AF 时延性能要优于 DDS.

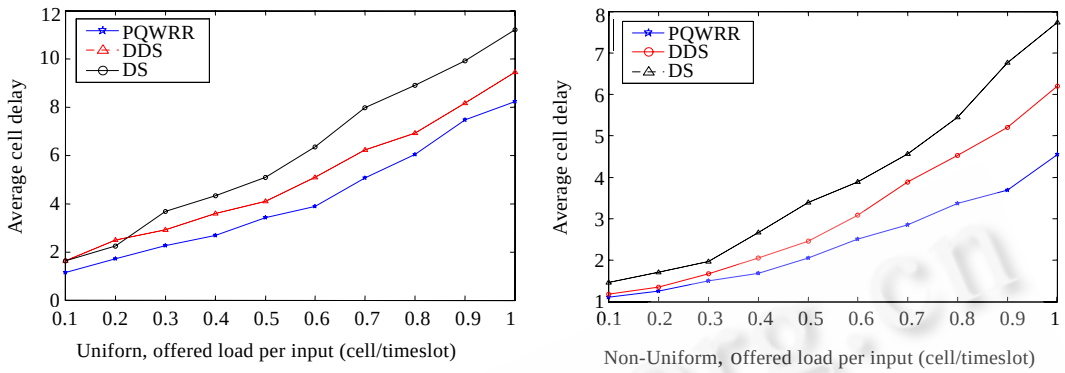

Fig.13 EF delay performance under uniform and non-uniform traffic

图 13 均匀与非均匀条件下 EF 业务平均时延

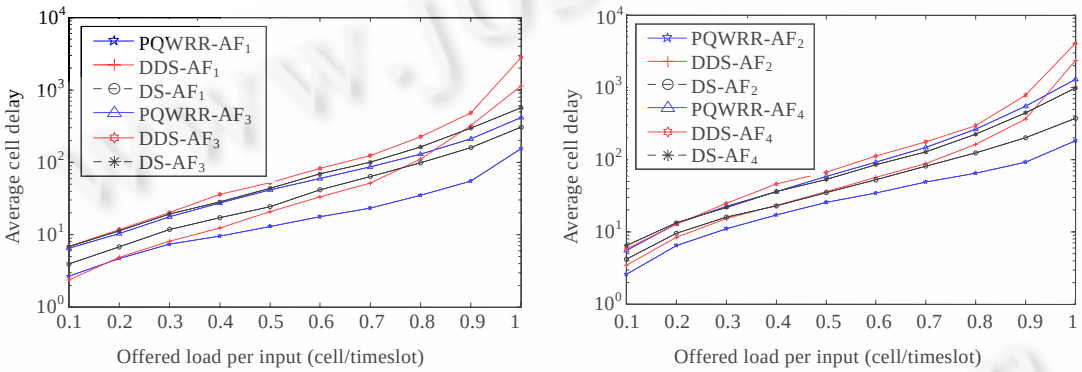

Fig.14 PQWRR,DDS and DS's AF delay performance under uniform traffic

图 14 均匀条件下 PQWRR,DDS,DS 的 AF 业务的时延性能

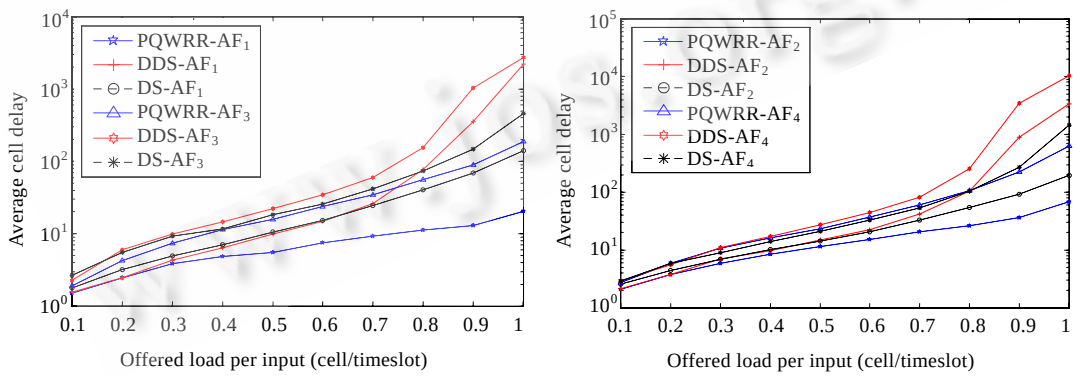

Fig.15 PQWRR,DDS and DS's AF delay performance under non-uniform traffic

图 15 非均匀条件下 PQWRR,DDS,DS 的 AF 业务的时延性能

图 16 描述了均匀与非均匀条件下,PQWRR,DDS,DS 的 EF 业务时延抖动曲线图,可以看出,DS 的时延抖动性能最优.

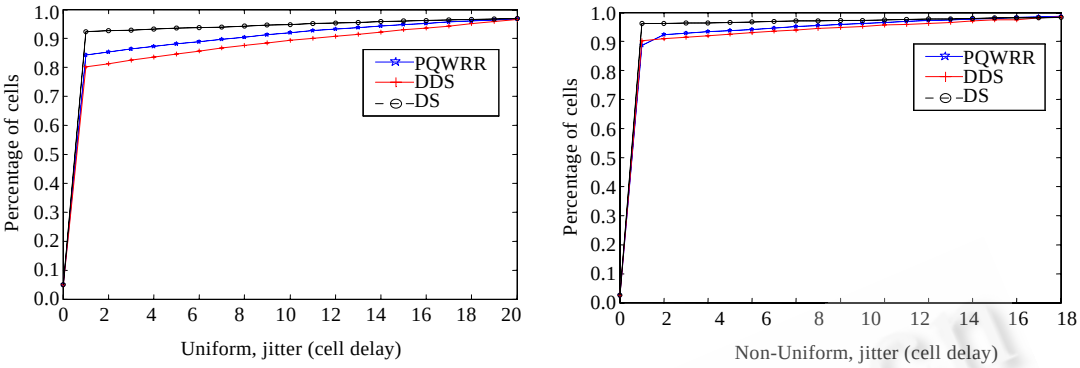

Fig.16 EF delay jitter performance under uniform and non-uniform traffic

图 16 均匀与非均匀条件下,EF 业务时延抖动曲线对比图

我们将 SPES 下得到的 DDS 和 PQWRR 的数据与文献[26,27]的仿真数据进行比较,结果一致,表明了 SPES 仿真平台的合理性.同时,仿真结果表明,DS 的性能特性与输入排队的 DDS、输出排队的 PQWRR 比较和理论分析是一致的.验证了我们的仿真平台.

5 结束语

交换技术的性能评价手段主要有理论分析和仿真实验两种.近年来,交换技术的发展使得单纯的理论分析手段很难描述交换系统的行为,因此,业界越来越多地采用仿真实验手段对交换技术的公平性和有效性进行评价.针对目前网络仿真软件应用于交换技术性能评价过于复杂、并且在可继承性与可扩展性方面存在缺陷,本文通过对交换系统建立统一的数学模型,采用系统级设计方法和面向对象技术设计并实现了一种专门的交换技术性能评价系统——SPES.该系统集成了多种典型的交换结构及其相应的典型调度算法,可以通过图形化界面进行系统参数设置,从而方便地构建不同的仿真环境,实现不同结构和算法的性能仿真.SPES 系统在设计上实现了业务流、排队机制和调度策略三者间的分离,具有良好的可继承和可扩展性,为新型交换技术的性能评估与验证提供了研究基础.

References:

[1] Texas Instruments. GS400.15-μ_m CMOS, Standard Cell/Gate Array. 2001. http://www.ti.com/

[2] Tamir Y, Frazier G. High performance multi-queue buffers for VLSI communication switches. In: Siegel HJ, ed. Proc. of the 15th Annual Symp. Computer Architecture. Honolulu: IEEE Communications Society, 1988. 343−354.

[3] Network Processing Forum Benchmarking Working Group. Traffic Models, 2003.

[4] Rojas-Cessa R, Oki E, Jing Z, Chao HJ. On the combined input-crosspoint buffered switch with round-robin arbitration. IEEE Trans. on Communications, 2005,11:1945−1951.

[5] Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. IEEE/ACM Trans. on Networking, 1993,1(4): 397−413.

[6] Shreedhar M, Varghese G. Efficient fair queuing using deficit round-robin. IEEE/ACM Trans. on Networking, 1996,4(3):375−385.

[7] Katevenis M, Sidiropoulos S, Courcoubetis C. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. IEEE Journal on Selected Areas in Communications, 1991,9(8):1265−1279.

[8] Garg R, Chen X. RRR: Recursive round robin scheduler. Computer Networks, 1999,31:1951−1966.

[9] Goyal P, Vin HM. Start-Time fair queueing: A scheduling algorithm for integrated services packet switching networks. IEEE/ACM Trans. on Networking, 1997,5(5):690−703.

[10] Demers AJ, Keshav S, Shenker S. Analysis and simulation of a fair queueing algorithm. In: Landweber LH, ed. Proc. of the ACM SIGCOMM ’99. Austin: ACM Press, 1989. 1−12.

[11] Bennett J CR, Zhang H. Why WFQ is not good enough for integrated services networks. In: Tokuda H, ed. Proc. of the NOSSDAV’96. Japan, 1996. http://www.cnaf.infn.it/~ferrari/sched.html

[12] Bennett R, Zhang H. WF^{2}Q: Worst-Case fair weighted fair queueing. In: Sohraby K, ed. Proc. of the IEEE INFOCOM'96. San Francisco: IEEE Press, 1996. 120−128.

[13] Bennett R, Zhang H. Hierarchical packet fair queueing algorithms. IEEE/ACM Trans. on Networking, 1997,5(5):675−689.

[14] McKeown N, Anantharam V, Walrand J. Achieving 100% throughput in an input-queued switch. In: Sohraby K, ed. Proc. of the IEEE INFOCOM'96. San Francisco: IEEE Communications Society, 1996. 296−302.

[15] McKeown N, Mekkittikul A. Starvation free algorithm for achieving 100% throughput in an input queued switch. In: Lee D, ed. Proc. of the ICCCN'96. Rockville: IEEE Communications Society, 1996. 226−229.

[16] Mekkittikul A, McKeown N. A practical scheduling algorithm to achieve 100% throughput in input-queued switches. In: Akyildiz I, ed. Proc. of the IEEE INFOCOM'98. San Francisco: IEEE Communications Society, 1998. 792−799.

[17] McKeown N. Scheduling algorithms for input-queue cell switches [Ph.D. Thesis]. University California Berkeley, 1995. http://Klamath.stanford.edu/~nickm/

[18] McKeown N. The *i*SIIP scheduling algorithm for input-queued switches. IEEE/ACM Trans. on Networking, 1999,7(2):188−201.

[19] Tamir Y, Chi HC. Symmetric crossbar arbiters for VLSI communication switches. IEEE Trans. on Parallel and Distributed Systems, 1993,4(1):13−27.

[20] Anderson TE, Owicki SS, Saxe JB, Thacker CP. High speed switch scheduling for local area networks. ACM Trans. on Computer System, 1993,11(4):319−352.

[21] Nabeshima M. Performance evaluation of a combined input-and crosspoint-queued switch. IEICE Trans. on Communications, 2000, E83-B(3):737−741.

[22] Javidi T, Magill R, Hrabik T. A high-throughput scheduling algorithm for a buffered crossbar switch fabric. In: Antikainen J, ed. Proc. of the IEEE ICC 2001. Helsinki: IEEE Press, 2001. 1581−1587.

[23] Mhamdi L, Hamdi M. MCBF: A high-performance scheduling algorithm for buffered crossbar switches. IEEE Communications Letters, 2003,9:451−453.

[24] Zhang X, Bhuyan LN. An efficient algorithm for combined input-crosspoint-queued (CICQ) switches. In: Shah R, ed. Proc. of the IEEE Globecom 2004. IEEE Press, 2004. 1168−1173.

[25] Sheng Z, Xie SQ, Pan CY. Probability Theory & Mathematical Statistics. Beijing: High Education Press, 2004 (in Chinese).

[26] Yang M, Lu E, Zheng SQ. Scheduling with dynamic bandwidth allocation for DiffServ classes. In: Proc. of the ICCCN 2003. Dallas: IEEE Press, 2003. 319−324.

[27] Mao J, Moh WM, Wei B. PQWRR scheduling algorithm in supporting of DiffServ. In: Abramsky S, Hofmann M, eds. Proc. of the ICC 2001. IEEE Press, 2001. 679−684.

[28] Blake S, Black D, Carison M, Davies E, Wang Z, Weiss W. An architecture for differentiated services. IETF RFC 2475, 1998.

附中文参考文献:

[25] 盛骤,谢式千,潘承毅.概率论与数理统计.北京:高等教育出版社,2004.

扈红超(1982－),男,河南商丘人,硕士,主要研究领域为高性能路由器.

郭云飞(1963－),男,教授,博士生导师,CCF高级会员,主要研究领域为高性能交换技术,下一代网络.

伊鹏(1977－),男,博士,讲师,主要研究领域为高性能交换,调度算法.