ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.7, July 2008, pp.1847−1855
DOI: 10.3724/SP.J.1001.2008.01847

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

一种支持DiffServ模型的全分布式调度算法*

伊 鹏+, 扈红超, 于 婧, 汪斌强

(国家数字交换系统工程技术研究中心,河南 郑州 450002)

A Distributed DiffServ Supporting Scheduling Algorithm

YI Peng+, HU Hong-Chao, YU Jing, WANG Bin-Qiang

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

+ Corresponding author: E-mail: yipengndsc@gmail.com

Yi P, Hu HC, Yu J, Wang BQ. A distributed DiffServ supporting scheduling algorithm. *Journal of Software*, 2008,19(7):1847−1855. http://www.jos.org.cn/1000-9825/19/1847.htm

Abstract: Scheduling algorithm is very important for network design to implement per hop behaviors (PHBs) in DiffServ model. Most of the presented DiffServ supporting scheduling algorithms are based on output queued (OQ) switches or input queued (IQ) switches, which are not suitable to be used in high speed network. This paper proposes a distributed DiffServ supporting scheduling (DDSS) algorithm based on combined input-crosspoint -queued (CICQ) switches. Theoretical analysis illuminates that the DDSS algorithm can obtain good fairness. The DDSS algorithm adopts a two-stage flow control mechanism based on periodic statistic to achieve fair bandwidth allocation for expedited forwarding (EF) and assured forwarding (AF) traffic, and uses a priority scheduling mechanism to provide lower delay for EF traffic. The time complexity of the DDSS algorithm is only $O(\log N)$, hence is practical and scalable for high speed network. Simulation results show that DDSS algorithm can obtain good fairness and delay performance. It is more appropriate to be used to support the DiffServ model.
Key words: DiffServ model; quality of service (QoS); scheduling; flow control; buffered crossbar

摘 要: 调度算法设计对于网络路由设备实现区分服务(DiffServ)模型的单跳行为(per hop behavior,简称 PHB)至关重要.现有支持 DiffServ 模型的调度算法普遍基于输出排队(output queued,简称 OQ)或是输入排队(input queued,简称 IQ)交换结构进行设计,均无法在高速环境下提供高性能的调度.基于联合输入/交叉节点排队(combined input-crosspoint-queued,简称 CICQ)交换结构提出一种支持 DiffServ 模型的全分布式调度算法 DDSS (distributed DiffServ supporting scheduling),并通过理论分析对其公平性进行了验证.DDSS 算法采用基于预约带宽的逐级流量控制机制实现所有预约带宽在快速转发(expedited forwarding,简称 EF)业务与确保转发(assured forwarding,简称 AF)业务之间的分配,采用优先级调度机制为 EF 业务提供低延迟服务,算法复杂度为 $O(\log N)$.仿真结果表明,DDSS 算法具有良好的时延性能和公平特性,与现有算法相比,能够更好地支持 DiffServ 模型.
关键词: 区分服务模型;服务质量;调度;流量控制;带缓存交叉开关

* 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)); the National Basic Research Program of China under Grant No.2007CB307102 (国家重点基础研究发展计划(973))

Received 2007-06-17; Accepted 2007-09-13

中图法分类号: TP393 文献标识码: A

随着Internet的迅速膨胀,IP网络服务质量(QoS)问题日益受到人们的关注.为了满足Internet上多种业务对QoS的需求,Internet工程任务组(IETF)先后制定了两种QoS服务模型:综合服务(integrated services/resource reservation protocol,简称IntServ/RSVP)模型[1]和区分服务(differentiated services,简称DiffServ)模型[2].IntServ模型试图通过全程信令将原本面向无连接的Internet改造为面向连接的网络,需要进行端到端的资源预留,要求所有网络路由设备都支持必要的信令协议,并且实现RSVP、接纳控制、MF(multi-field)包分类和包调度.这种方式对网络路由设备要求较高,存在开销大、可扩展性差等问题[3].鉴于此,IETF提出了一种克服上述问题的区分服务(DiffServ)模型.在DiffServ模型中,边缘网络设备对每个分组进行分类并标记DS域,用DS域来携带IP分组对QoS的需求信息;核心网络设备根据分组头上的区分服务码点(differentiated service code point,简称DSCP)[4]选择所对应的转发处理.DiffServ模型充分考虑了IP网络的灵活性和可扩展性,将复杂的服务质量保证通过DSCP转换为简单的单跳行为(per hop behavior,简称PHB),从而极大地降低了实现复杂度.由于DiffServ模型具有良好的可扩展性,而且便于实现,因此更适合在IP网络上获得广泛的应用.目前DiffServ模型中定义了 3 类PHB:快速转发(expedited forwarding,简称EF)[5]、确保转发(assured forwarding,简称AF)[6]和尽力而为(best effort,简称BE).其中AF内部又定义了 4 个子类,每个子类中按照丢包策略又可分为 3 个优先级.不同种类的PHB用于为不同类型的DiffServ业务提供区分的服务.

1 现有算法评述

DiffServ模型要求DiffServ域中的每个网络节点提供针对不同DSCP值的PHB服务,而PHB的实现在很大程度上依赖于网络路由设备中调度算法的设计[7].由于输出排队(output queued,简称OQ)交换结构可以获得高的吞吐量,而且其调度机制可独立工作于各个输出端口,复杂度较低,易于实现对DiffServ模型的支持,因而多数支持DiffServ模型的调度算法都是基于OQ交换结构进行设计的,典型的,如优先级排队(priority queuing,简称PQ)[8]、加权轮询服务(weighted round-robin,简称WRR)[9]以及联合优先级排队加权轮询(priority queuing weighted round-robin,简称PQWRR)[10]等.然而OQ交换结构自身存在N倍加速问题[11],在高速环境下难以实现,因此这类支持DiffServ模型的调度算法在高速环境下的应用受到限制[12].为了克服这一缺陷,研究人员考虑基于输入排队(input queued,简称IQ)交换结构设计支持DiffServ模型的调度算法.虽然IQ交换结构无须加速,可以在高速环境下实现,但IQ交换结构的调度需要对交换结构的所有端口进行全局考虑,必须采用复杂的集中式调度算法才能获得良好的性能,不易于工程实现.目前基于IQ交换结构支持DiffServ模型的调度算法如动态DiffServ调度算法(dynamic DiffServ scheduling,简称DDS)[13]和分级DiffServ调度算法(hierachical DiffServ scheduling,简称HDS)[14],虽然通过采用迭代方式逼近最大匹配可以在一定程度上降低算法复杂度,但它们仅能在均匀业务条件下获得较好的性能.对于非均匀业务,DDS和HDS算法都会导致IQ交换结构吞吐量降低,当业务负载较重时,其时延性能将急剧恶化.因此基于IQ交换结构支持DiffServ模型的调度算法在实际应用中也存在着很大的局限性.

近年来,随着芯片设计技术的进步和芯片工艺水平的提高,带缓存交叉开关成为交换结构领域的研究热点.目前,基于联合输入/交叉节点排队(combined input-crosspoint-queued,简称CICQ)交换结构已经提出多种简单、高效的调度算法[15−17],但它们仅关注如何获取高吞吐量和低时延,无法针对不同业务提供区分服务.在提供服务质量保障方面,Magill等人证明了CICQ交换结构在 2 倍加速条件下可以通过分布式调度算法模拟支持k个优先级的先入先出输出排队交换结构(first come first serve-output queuing,简称FCFS-OQ)[18],Chuang等人证明了在3倍加速条件下CICQ交换结构可以通过分布式调度算法实现模拟采用任意调度算法的OQ交换结构[19],我们证明了当空分复用扩展因子为 2 时CICQ交换结构无须加速即可模拟采用任意调度算法的OQ交换结构[20].这些研究虽然从理论上论证了CICQ交换结构具有提供服务质量保障的能力,但是模拟OQ交换结构的实现代价过高,并且缺乏简单、高效的区分服务调度方案.

为了更好地支持 DiffServ 模型,本文结合交换结构领域的新进展,基于 CICQ 交换结构提出一种支持 DiffServ 模型的全分布式调度算法 DDSS(distributed DiffServ supporting scheduling).DDSS 算法通过在 CICQ 交换结构中引入支持 DiffServ 模型的排队机制,结合 CICQ 交换结构的特点,采用基于预约带宽的逐级流量控制机制,实现所有预约带宽在 EF 业务与 AF 业务之间的分配,并通过优先级调度机制为 EF 业务提供低延迟服务,理论分析表明,它具有良好的公平特性.由于 DDSS 算法采用全分布式的调度机制,可以并行工作于交换结构的各个输入输出端口,算法复杂度仅为 $O(\log N)$,而且 DDSS 算法可以直接支持变长包的处理,避免了切片重组造成的额外开销,因此实现起来十分简单,文中给出了一种参考硬件实现方案.为了进一步验证 DDSS 算法的性能,本文通过仿真对 PQWRR,DDS 和 DDSS 算法的公平性和时延性能进行了比较,仿真结果表明,DDSS 算法始终具有良好的时延性能和公平特性,能够更好地支持 DiffServ 模型.

2 交换结构及其排队机制

随着芯片技术及其工艺水平的发展,带缓存交叉开关成为交换结构领域的研究热点,其中一种典型的结构为联合输入/交叉节点排队(CICQ)交换结构.CICQ 交换结构无须加速,可以在高速环境下实现,具有良好的可扩展性.图 1 给出了 $N\times N$ 的 CICQ 交换结构及其支持 DiffServ 模型排队机制的示意图,它采用带缓存交叉开关作为核心交换单元,在交换结构的每个输入端口分别设置一个输入缓存队列.为了避免队头阻塞,每个输入缓存队列在逻辑上被分为 N 个虚拟输出队列(virtual output queuing,简称 VOQ),每个虚拟输出队列用于缓存到达一个输出端口的报文.由于 DiffServ 模型中定义了对应不同应用需求的多种 PHB,为了提供满足不同应用需求的服务,每个虚拟输出队列在逻辑上被分成 K 个虚拟优先级队列(virtual priority queuing,简称 VPQ),每个虚拟优先级队列用于缓存一个虚拟输出队列当中具有相同优先级的报文.与此同时,带缓存交叉开关内部的每个交叉节点缓存也采用类似的机制进行区分优先级的排队.对于支持 DiffServ 模型的调度,K 被设置为 6,其中优先级从 1~6 分别对应 EF,AF1,AF2,AF3,AF4,BE 这 6 种类型的 DiffServ 业务.

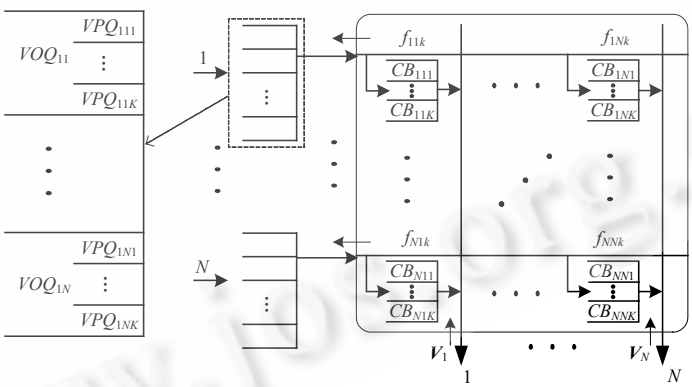

Fig.1 CIOQ switch and queuing mechanism

图 1 CICQ 交换结构及其排队机制

如图 1 所示,下文采用 VOQ_{ij} 标识输入端口为 i 输出端口为 j 的虚拟输出队列,用 VPQ_{ijk} 标识虚拟输出队列 VOQ_{ij} 中优先级为 k 的虚拟优先级队列,用 CB_{ijk} 标识与虚拟优先级队列 VPQ_{ijk} 和交换结构输出端口 j 相关联的交叉节点缓存队列,其中每个交叉节点缓存队列可以缓存一个最长包.为了防止交叉节点缓存溢出导致报文丢失,对于每个 CB_{ijk} 维护一个流控状态信号 f_{ijk} 用于对 VPQ_{ijk} 进行流量控制,当 $f_{ijk}=1$ 时,禁止 VPQ_{ijk} 中的报文输出;当 $f_{ijk}=0$ 时,允许 VPQ_{ijk} 中的报文输出.为了实现不同类型DiffServ业务之间的带宽分配,对于每个输出端口 j 维护一个 K 维流控状态向量 V_j,用于对输出端口 j 进行流量控制.当 $V_{jk}=1$ 时,说明输出端口为 j 优先级为 k 的一类DiffServ业务已经获得了其预先约定的带宽,此时所有 CB_{ijk} (i=1,2,...,N)禁止输出报文;当 $V_{jk}=0$ 时,说明输出端口为 j 优先级为 k 的一类DiffServ业务已获得带宽小于其预先约定带宽,此时所有 CB_{ijk} (i=1,2,…,N)允许输出报文.

由于 CICQ 交换结构在核心交换单元内部引入了交叉节点缓存,因此 CICQ 交换结构的调度被分为输入调

度和输出调度两个部分.其中输入调度部分负责对输入缓存队列中的报文进行调度,依据一定的准则选择合适的报文送往交叉节点缓存队列,输出调度部分负责对交叉节点缓存队列中的报文进行调度,依据一定的准则选择合适的报文送往交换结构的输出端口.在选择判决过程中,不同的判决准则对应着不同的调度算法.

3 DDSS 调度算法

由于调度算法用于决定交换结构在拥塞时优先服务的报文,因此它对于网络路由设备实现DiffServ模型中针对不同 DSCP 值的 PHB 服务至关重要.本节提出一种 DDSS 调度算法,并通过公平指数对调度算法的公平性进行理论分析,最后给出 DDSS 调度算法的硬件电路设计.

3.1 调度算法描述

DDSS调度算法主要包含基于预约带宽的逐级流量控制机制、输入调度策略和输出调度策略 3 个部分.基于预约带宽的逐级流量控制机制用于产生和维护交叉节点缓存的流控状态信号f_{ijk}和输出端口的流控状态向量$\boldsymbol{V}_j$,输入调度策略根据流控状态信号f_{ijk}的状态对输入缓存队列中的报文进行区分优先级的调度,输出调度策略根据流控状态向量$\boldsymbol{V}_j$的状态对交叉节点缓存队列中的报文进行区分优先级的调度.DiffServ模型为EF业务定义了峰值信息速率(peak information rate,简称PIR)用于约束EF业务获得的带宽,为 4 种AF业务定义了承诺信息速率(committed information rate,简称CIR)用于保证各种AF业务获得的带宽,其中$AF(k)$业务的承诺信息速率标记为$CIR(k)$.DDSS调度算法根据这两类参数针对不同优先级的DiffServ业务设置其预约带宽,标记为R_{jk},其中$R_{j1}=PIR$,$R_{jk}=CIR(k-1)(k=2,\ldots,K-1)$.算法的基本思想是,通过基于预约带宽的逐级流量控制机制实现所有预约带宽在EF业务与AF业务之间的分配,通过优先级调度机制为EF业务提供低延迟服务.

基于预约带宽的逐级流量控制机制为每个AF业务分配一个计数器,以一个固定时长T为间隔,周期性地对不同种类的AF业务在一个T时间周期内已获得的输出带宽进行统计.周期T的选取要求T与最小预约带宽R_{jk}的乘积至少大于一个最大包长,即$T\cdot R_{jk}>\mathrm{Max}$.在每一个$T$时间周期的初始时刻,所有AF业务的计数器被清零并开始计数.对于EF业务,采用基于周期统计进行流量控制会增大EF业务的时延抖动,因此为EF业务分配的计数器从初始时刻开始进行持续计数.输出端口j优先级为k的DiffServ业务获得调度的字节数被标记为C_{jk}.输出调度应用的流控状态向量$\boldsymbol{V}_j$通过对周期性统计量C_{jk}和预约带宽值R_{jk}的比较进行更新.在任意时刻t,当$k=1$ 时,如果$C_{j1}\geq t\cdot R_{j1}$,则$V_{jk}=1$,否则$V_{jk}=0$;当$k=2$ 到K时,如果$C_{jk}\geq T\cdot R_{jk}$,则$V_{jk}=1$,否则$V_{jk}=0$.输入调度应用的流控状态信号f_{ijk}直接根据对应交叉节点缓存队列CB_{ijk}是否处于满状态进行更新,如果CB_{ijk}已被写满,则$f_{ijk}=1$,否则$f_{ijk}=0$.由于$V_{jk}=1$时所有$CB_{ijk}(i=1,2,\ldots,N)$禁止输出报文,而每个交叉节点缓存队列只能缓存 1 个最长包,因此在输出端基于预约带宽进行流量控制会很快导致交叉节点缓存队列被写满,从而反馈到输入端口,实现基于预约带宽的逐级流量控制.DDSS调度算法采用这一流控机制可以根据不同类型DiffServ业务的预约带宽实现EF业务与AF业务之间的带宽分配.以下分输入调度和输出调度两个部分对DDSS调度算法进行描述:

- 输入调度

对于每个输入端口i:根据流控状态信号f_{ijk}和优先级标识将输入端口的所有虚拟优先级队列划分为K个集合$\mathbf{M}_{ik}(k=1,2,\ldots,K)$,集合$\mathbf{M}_{ik}$包含输入端口$i$优先级为$k$且对应流控状态信号$f_{ijk}=0$ 的所有非空虚拟优先级队列.在K个集合并行地选取每个集合中具有最大长度的虚拟优先级队列作为候选队列,然后在候选队列中选取具有最高优先级的虚拟优先级队列获得服务,并更新对应的流控状态信号f_{ijk}.

- 输出调度

对于每个输出端口j:根据优先级标识将与输出端口j关联的所有交叉节点缓存队列划分为K个集合$\mathbf{S}_{jk}(k=1,2,\ldots,K)$,集合$\mathbf{S}_{jk}$包含输出端口$j$优先级为$k$的所有交叉节点缓存队列.在$K$个集合并行地选取每个集合中具有最大长度的交叉节点缓存队列作为候选队列,如果流控状态向量$\boldsymbol{V}_j$的前$K-1$ 个元素全部为 1,则从集合$\mathbf{S}_{jk}(k=2,\ldots,K)$选出的候选队列中选取具有最大长度的交叉节点缓存队列获得服务,否则在候选队列中选取具有最高优先级的交叉节点缓存队列获得服务,然后更新对应的流控状态向量$\boldsymbol{V}_j$.

DDSS 调度算法中在 K 个集合并行选取最长队列的复杂度为 $O(\log N)$,在候选队列中选取获得服务的队列

复杂度为 $O(\log K)$,由于 K 是常数,并且输入调度和输出调度可以在交换结构的输入端口和输出端口并行工作,因此 DDSS 调度算法的时间复杂度仅为 $O(\log N)$,具有良好的可扩展性.

3.2 公平性分析

公平性是指链路带宽必须以公平、合理的方式分配给共享链路带宽的业务流.因此高性能的调度算法应可以隔离不同的业务流,这样即便发生高突发型业务也不至于影响其他正常业务流.单位时间内不同业务流由于预约带宽不同因而获得的服务量也不相同,因此通常采用归一化服务量——服务公平指数来衡量调度算法的公平性.对于任意一种调度算法,第k个业务流的服务公平指数定义为第k个业务流获得的实际带宽比FQ_k和理想带宽比IFQ_k(应得份额)的比值,公平指数越接近 1,说明调度算法公平性越好.

由于在DiffServ模型中BE业务没有预约带宽,只需为其提供尽力而为的服务,因此本文仅讨论所有预约带宽在EF业务和所有AF业务之间分配的公平性.由于仅当交换结构处于拥塞导致不同业务流存在带宽争用时才需要考虑公平性,以下我们简单地假设输出端口j所有EF业务和AF业务的到达速率均超出其预约带宽.令$Sent_{k,n}$为第 1~第n个周期优先级为k的DiffServ业务获得预约带宽的字节数,$Sent_n$为第 1~第n个周期所有EF业务和AF业务获得预约带宽的字节数.优先级为k的DiffServ业务获得的实际带宽比、理想带宽比以及公平指数分别在式(1)~式(3)中定义.

$$FQ_k = \frac{Sent_{k,n}}{Sent_n} \tag{1}$$

$$IFQ_k = \frac{R_{jk}}{\sum_{k=1}^{K-1} R_{jk}} \tag{2}$$

$$FairnessIndexk_k = \frac{FQ_k}{IFQ_k} \tag{3}$$

由于 DDSS 调度算法在调度一个整包时才对流控状态向量进行更新,任何一种优先级的 DiffServ 业务在一个周期中应该获得预约带宽和实际获得预约带宽的字节数之差最大为 Max,因此,

$$n \cdot R_{jk} \cdot T \le Sent_{k,n} \le n \cdot R_{jk} \cdot T + n \cdot \mathrm{Max} \tag{4}$$

$$n \cdot \sum_{k=1}^{K-1} R_{jk} \cdot T \le Sent_n \le n \cdot \sum_{k=1}^{K-1} R_{jk} \cdot T + n \cdot (K-1)\mathrm{Max} \tag{5}$$

$$\frac{n \cdot R_{jk} \cdot T}{n \cdot \sum_{k=1}^{K-1} R_{jk} \cdot T + n \cdot (K-1)\mathrm{Max}} \le FQ_k = \frac{Sent_{k,n}}{Sent_n} \le \frac{n \cdot R_{jk} \cdot T + n \cdot \mathrm{Max}}{n \cdot \sum_{k=1}^{K-1} R_{jk} \cdot T} \tag{6}$$

$$\frac{\sum_{k=1}^{K-1} R_{jk} \cdot T}{\sum_{k=1}^{K-1} R_{jk} \cdot T + (K-1)\mathrm{Max}} \le FairnessIndexk_k = \frac{FQ_k}{IFQ_k} \le \frac{R_{jk} \cdot T + \mathrm{Max}}{R_{jk} \cdot T} \tag{7}$$

由于 $\mathrm{Min}(R_{jk} \cdot T) \ge \mathrm{Max}$,因而 $\sum_{k=1}^{K-1} R_{jk} \cdot T \ge (K-1)\mathrm{Max}$, 所以,

$$\frac{1}{2} \le FairnessIndexk_k \le 2 \tag{8}$$

在实际应用中,周期T与最小预约带宽R_{jk}的乘积会远大于一个最大包长,服务公平指数将接近于 1,因此DDSS调度算法具有良好的公平性.这一特性将在仿真分析中得到进一步验证.

3.3 电路设计

DDSS 调度算法采用全分布式的调度机制将调度判决的实现分布到各个输入输出端口,可以通过 N 个相同的输入调度机和 N 个相同的输出调度机并行工作,实现 CICQ 交换结构的调度,因而简化了硬件电路的设计.

图 2 给出了输入调度机和输出调度机的电路设计.由于输入端流控状态信号可以直接由对应交叉节点缓存队列的满指示信号产生,输出端口流控状态向量的更新电路可以通过简单的计数器电路和比较器电路实现,因此图中略去了此部分电路的实现细节.图中用L_{ijk}标记虚拟优先级队列VPQ_{ijk}的对比长度,用C_{ijk}标记交叉节点缓存队列CB_{ijk}的长度.当f_{ijk}=1 时,L_{ijk}为优先级队列VPQ_{ijk}的实际长度,当f_{ijk}=1 时,L_{ijk}=0.这里,可以通过简单的二

选一电路实现,因此未在图中描述.

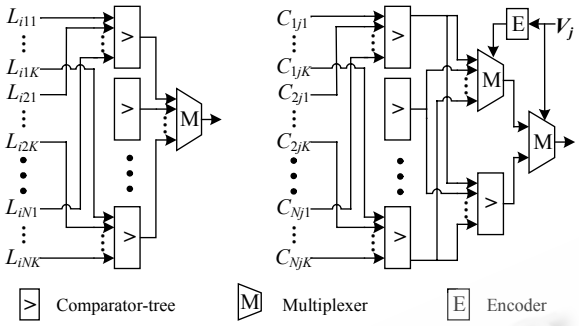

Fig.2 Circuit design of DDSS schedule algorithm

图 2 DDSS 调度算法电路设计

每个输入端口的输入调度机由 K 个 N 输入的树形比较器和一个 K 输入的多路复用器组成,其中 N 输入的树形比较器用于在具有相同优先级的虚拟优先级队列中选取最长队列,K 输入的多路复用器用于选取最高优先级的候选队列.每个输出端口的输出调度机由 K 个 N 输入的树形比较器、一个 K 输入的多路复用器、一个 K 输入的树形比较器和一个二选一复用器组成,其中 N 输入的树形比较器用于在具有相同优先级的交叉节点缓存队列中选取最长队列,K 输入的多路复用器用于从未超过预约带宽的 EF 和 AF 业务中选取最高优先级的候选队列,K 输入的树形比较器用于从 AF 和 BE 业务中选取具有最大长度的候选队列,最终由二选一复用器根据流控状态向量选取获得服务的队列.假设调度判决的所有队列最大长度为 L,则 N 输入的树形比较器的硬件复杂度为 $O(\log N\cdot\log L)$,K 输入的树形比较器和多路复用器具有 $O(\log K\cdot\log L)$,输入调度机的硬件复杂度为 $O(K\cdot\log N\cdot\log L+\log K\cdot\log L)$,输出调度机的硬件复杂度为 $O(K\cdot\log N\cdot\log L+2\log K\cdot\log L+\log L)$.由于 K 是常数,因此 DDSS 算法的硬件复杂度为 $O(\log N\cdot\log L)$,十分方便高速硬件实现.

4 仿真分析

本节通过仿真从公平性和有效性两个方面对DDSS调度算法进行性能验证.算法的公平性通过不同优先级的DiffServ业务获得的归一化带宽来衡量,算法的有效性通过不同DiffServ业务的平均时延性能来验证.由于基于OQ交换结构的PQWRR算法和基于IQ交换结构的DDS算法在同类算法中颇具性能优势,我们分别通过编程实现了PQWRR算法、DDS算法和DDSS算法,以便于进行性能比较.所有的仿真都假定到达报文为定长包,并且采用了与文献[7]相似的仿真参数,即业务到达过程通过ON-OFF模型产生,突发长度为 32,每个输入端口EF业务与AF业务的比例依次为 18%,24%,20%,16%,12%,R_{j1}到R_{jk}依次设置为 0.198,0.24,0.20,0.16,0.12.下文将定长包标记为*cell*,仿真所采用的时间粒度为传送一个*cell*所需的时间,也称为一个时隙,在配置为Pentium(R) 4 CPU 2.60GHz,1.00G内存的主机上运行上述算法的仿真程序,单次仿真周期为 10^6时隙,计算机的运行时间为 30s.

我们首先验证 3 种算法在交换结构输出端口过载条件下对各类 DiffServ 业务分配带宽的公平性.此时,所有仿真平台均采用 4×4 的交换结构,交换结构所有输入端口到达业务均以输出端口 1 为目的端口,业务负载量ρ从 0.1 变化到 1.图 3 给出了 PQWRR 算法、DDS 算法和 DDSS 算法的带宽分配情况.由图 3 可知,采用 PQWRR 算法时,过载的 EF 业务会抢占其他业务的带宽,对其他业务的服务质量会造成不良影响,而 DDS 算法和 DDSS 算法均表现出良好的公平特性.

下面通过对 3 种算法在均匀业务($\lambda_{ij}=\rho/N\ \forall i,j$)和非均匀业务($\lambda_{ii}=0.7\rho,\lambda_{i|i+1|}=0.3\rho$)条件下时延特性的仿真对算法的有效性进行验证,仿真平台采用 16×16 的交换结构.对于均匀业务,图 4 给出了 3 种算法中EF业务的平均时延曲线,图 5 为EF业务在负载为 0.7 时的时延抖动分布,图 6 和图 7 分别给出了AF业务和BE业务的平均时延对比.由于AF业务种类较多,为了便于比较,我们仅给出了AF1 和AF3 的时延曲线.

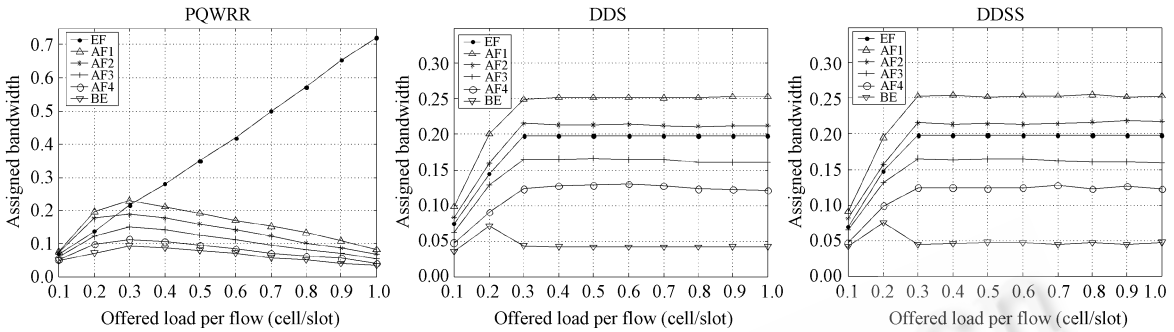

Fig.3 Comparison of bandwidth allocation

图 3 带宽分配特性对比

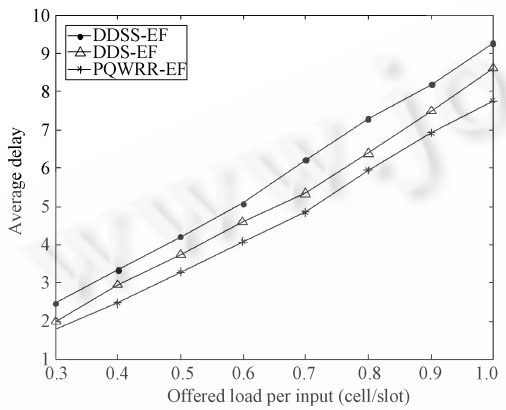

Fig.4 Average delay of EF under uniform traffic

图 4 均匀条件下 EF 平均时延

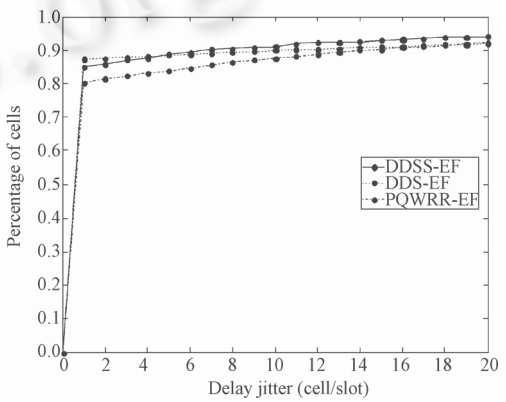

Fig.5 Delay jitter of EF under uniform traffic

图 5 均匀条件下 EF 时延抖动

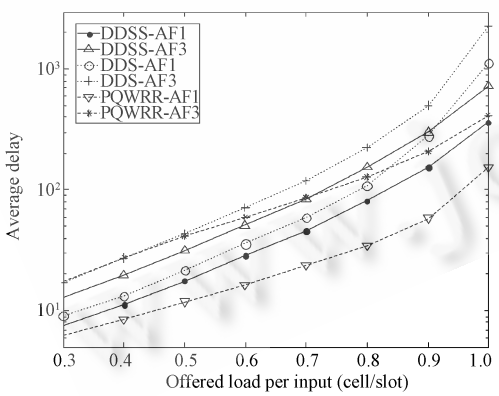

Fig.6 Average delay of AF under uniform traffic

图 6 均匀条件下 AF 平均时延

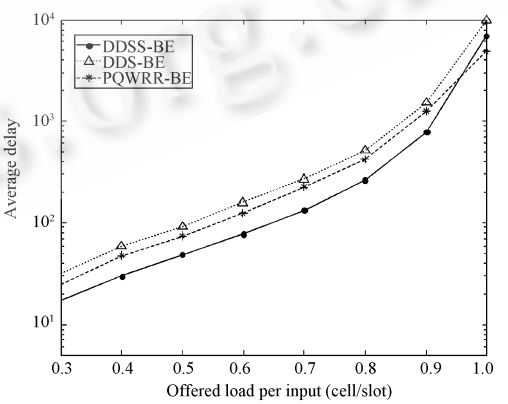

Fig.7 Average delay of BE under uniform traffic

图 7 均匀条件下 BE 平均时延

对于非均匀业务,图 8 给出了 3 种算法中 EF 业务的平均时延曲线,图 9 为 EF 业务在负载为 0.7 时的时延抖动分布情况,图 10 和图 11 分别给出了 AF 业务和 BE 业务的平均时延对比情况.

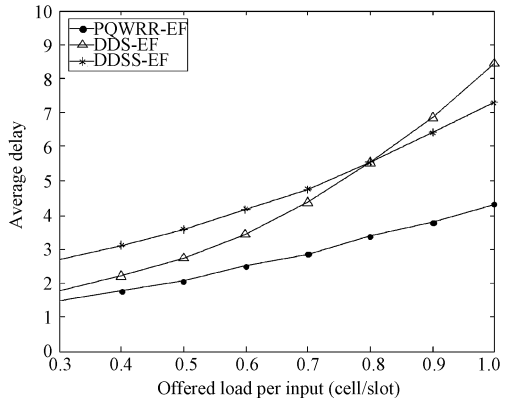

Fig.8 Average delay of EF under non-uniform traffic
图 8 非均匀条件下 EF 平均时延

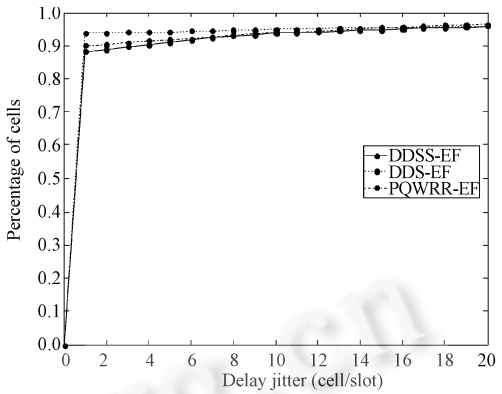

Fig.9 Delay jitter of EF under non-uniform traffic
图 9 非均匀条件下 EF 时延抖动

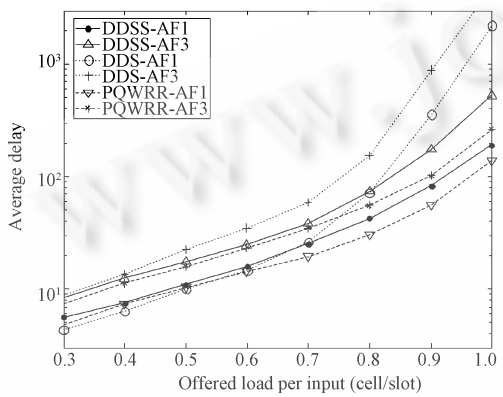

Fig.10 Average delay of AF under non-uniform traffic
图 10 非均匀条件下 AF 平均时延

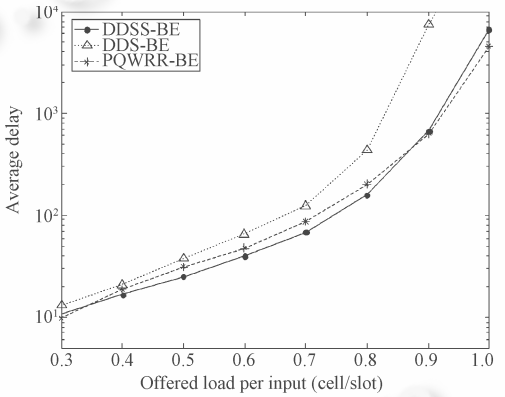

Fig.11 Average delay of BE under non-uniform traffic
图 11 非均匀条件下 BE 平均时延

由图可知,3 种算法在均匀业务和非均匀业务条件下均能为 EF 业务提供良好的服务,并且在均匀业务条件下,AF 业务和 BE 业务也可以获得较好的时延特性,但对于非均匀业务,采用 DDS 算法会导致 AF 业务和 BE 业务在负载较重时的时延特性急剧恶化,而采用 PQWRR 算法和 DDSS 算法依然可以获得良好的性能.综合比较上述仿真结果,只有 DDSS 算法在公平性和有效性两个方面均表现出良好的性能,因此能够更好地支持 DiffServ 模型.

5 结束语

本文基于联合输入/交叉节点排队交换结构提出一种支持 DiffServ 模型的全分布式调度算法 DDSS,并且给出了参考硬件实现方案.与其他支持 DiffServ 模型的调度算法相比,DDSS 算法的优势主要有 3 点:(1) DDSS 算法采用全分布式的调度机制,可以并行工作于交换结构的各个输入、输出端口,算法复杂度仅为 $O(\log N)$,适用于高速硬件实现.(2) DDSS 算法可以直接支持变长包的处理,避免了切片重组造成的额外开销.(3) DDSS 算法基于 CICQ 交换结构设计,比基于 OQ 交换结构和 IQ 交换结构的调度算法具有更好的可扩展性.理论分析和仿真实验一致说明,DDSS 算法具有良好的时延性能和公平特性,能够更好地支持 DiffServ 模型.

致谢 在此,我们向对本文的工作给予支持和建议的同行,尤其是国家数字交换技术研究中心的兰巨龙教授、张兴民教授领导的研发团队中的同学和老师表示感谢.

References:

[1] Braden R, Clark D, Shenker S. Integrated services in the Internet architecture: An overview. RFC1633, 1994.

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

[3] Carpenter B, Nichols K. Differentiated services in the Internet. Proc. of the IEEE, 2002,90(9):1479−1494.

[4] Nichols K, Blake S, Baker F, Black D. Definition of the differentiated service field (DS field) in the IPv4 and IPv6 headers. RFC 2474, 1998.

[5] Jacobson V, Nichols K, Poduri K. An expedited forwarding PHB. RFC 2598, 1999.

[6] Heinanen J, Baker F, Weiss W, Wroclawski J. Assured forwarding PHB group. RFC 2597, 1999.

[7] Liu QR. Research of the implementation mechanism of terabit high performance router [Ph.D. Thesis]. Zhengzhou: Information Engineering University, 2004 (in Chinese with English abstract)

[8] Heinanen J, Baker F, Weiss W, Wroclawski J. Assured forwarding PHB group. IETF RFC 2597, 1999.

[9] Kwak J-Y, Nam J-S, Kim DH. A modified dynamic weighted round robin cell scheduling algorithm. ETRI Journal Trans. on Communications, 2002,24(5):360−372.

[10] Mao J, Moh WM, Wei B. PQWRR scheduling algorithm in supporting of DiffServ. In: Estola K-P, ed. Proc. of the ICC, Vol.3. 2001. 679−684.

[11] Kesidis G, McKeown N. Output-Buffer ATM packet switching for integrated-services communication networks. In: Proc. of the IEEE ICC'97. Montreal, 1997. 1684−1688. http://citeseer.ist.psu.edu/kesidis97outputbuffer.html

[12] Peng YI, Wang BQ, Guo YF, LI H. Providing QoS guarantees in a novel switch architecture. Journal of Electronic, 2007,5(7): 28−35 (in Chinese with English abstract).

[13] Yang M, Lu E, Zheng SQ. Scheduling with dynamic bandwidth share for DiffServ classes. In: Proc. of the ICCCN. 2003. 319−324.

[14] Yang M, Wang J, Lu E, Zheng SQ. Hierarchical scheduling for DiffServ classes. In: Proc. of the IEEE Globecom. 2004. 707−712. http://faculty.salisbury.edu/~ealu/Papers/hds_globecom2004.pdf

[15] Rojas-Cessa R, Oki E, Jing Z, Chao JH. On the combined input-crosspoint buffered switch with round-robin arbitration. IEEE Trans. on Commun, 2005,53(11):1945−1951.

[16] Zheng YF, Shao C. An efficient round-robin algorithm for combined input-crosspoint-queued switches. In: Dini P, ed. Proc. of the IEEE ICAS/ICNS 2005. Papeete: IEEE Computer Society, 2005. 23−28.

[17] Luo JZ, Lee Y, Wu J. DRR: A fast high-throughput scheduling algorithm for combined input-crosspoint-queued (CICQ) switches. In: George R, ed. Proc. of the IEEE MASCOTS 2005. Atlanta: IEEE Computer Society, 2005. 329−332.

[18] Magill B, Rohrs C, Stevenson R. Output-Queued switch emulation by fabrics with limited memory. IEEE Journal on Selected Areas in Communications, 2003,4(21):606−615.

[19] Iyer S, Chuang ST, McKeown N. Practical algorithms for performance guarantees in buffered crossbars. In: Proc. of the IEEE INFOCOM 2005. Miami, 2005. 981−991. http://tiny-tera.stanford.edu/~nickm/papers/Infocom05_bufxbar.pdf

[20] Yi P, Wang BQ, Guo YF. Providing QoS guarantees in buffered crossbars with space-division multiplexing expansion. In: Kero T, ed. Proc. of the IEEE GLOBECOM 2006. San Francisco, 2006. 1−6.

附中文参考文献:

[7] 刘勤让.*T* 比特高性能路由器 QoS 实现机制研究[博士学位论文].郑州:信息工程大学,2004.

[12] 伊鹏,汪斌强,郭云飞,李挥.一种可提供 QoS 保障的新型交换结构.电子学报,2007,5(7):28−35.

伊鹏(1977－),男,湖北黄冈人,博士,讲师,主要研究领域为路由交换技术.

于婧(1979－),女,博士生,主要研究领域为对等网络路由协议.

扈红超(1982－),男,博士生,主要研究领域为交换结构,调度算法.

汪斌强(1963－),男,教授,博士生导师,主要研究领域为宽带信息网,高速路由器核心技术.