ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.4, April 2006, pp.822−829
DOI: 10.1360/jos170822
© 2006 by *Journal of Software*. All rights reserved.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

EPFTS 中基于时槽加权的公平调度算法*

李 季+, 曾华燊

(西南交通大学 信息科学与技术学院,四川 成都 610031)

Timeslot Weighted Fair Scheduling in EPFTS

LI Ji+, ZENG Hua-Xin

(School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China)

+ Corresponding author: Phn: +86-28-87601745 ext 601, E-mail: leejee@163.com, http://sist.swjtu.edu.cn/

Li J, Zeng HX. Timeslot weighted fair scheduling in EPFTS. *Journal of Software*, 2006,17(4):822−829. http://www.jos.org.cn/1000-9825/17/822.htm

Abstract: Based on the technology of EPFTS (ethernet-like physical frame timeslot switching), this paper introduces a new scheduling algorithm call TWFS (timeslot weighted fair scheduling) which could be implemented in EPFTS nodes to fit the requirement of fast data switching and QoS (quality of service) guarantee in SUPANET (single physical layer user-data platform architecture network). By analyzing merits and shortcomings of the typical scheduling techniques such as iSlip (iteration round-robin match with slip) and BvN-switch (Birkhoff-von neumann switch), TWFS utilizes the same iteration mechanism as iSlip and takes the total reserved timeslots of an I/O port pair as the weight (priority) for data switching, thus overcomes the shortcoming of the slow response to traffic fluctuation of BvN-switch, while keeping the same calculation complexity in iteration to $O(\log_2 N)$ as with iSlip. Simulation results have shown that TWFS can made an optimal balance among effectiveness、fairness, and complexity and is most suitable for future high-speed EPFTS nodes in SUPANET.

Key words: scheduling algorithm; TWFS (timeslot weighted fair scheduling); EPFTS (ethernet-like physical frame timeslot switching); SUPANET (single physical layer user-data platform architecture network); input queuing

摘 要: 基于 EPFTS(ethernet-like physical frame timeslot switching)交换技术,提出了一种新型调度算法 TWFS(timeslot weighted fair scheduling),可实现于 EPFTS(ethernet-like physical frame timeslot switching)交换节点,满足 SUPANET(single physical layer user-data platform architecture network)网络中具备 QoS(quality of service)保障能力的快速数据转发的需要.通过分析两类典型的调度机制 iSlip(iteration round-robin match with slip)和 BvN-switch(Birkhoff-von neumann switch)的优缺点,TWFS 利用类似 iSlip 的迭代机制,以交换节点输入输出端口对上预定的时槽总数作为数据转发的度量权值(优先权),克服了 BvN-switch 对负载变化反应慢的缺点,同时又使算法时间复杂度保持在与 iSlip 相同的级别 $O(\log_2 N)$.仿真实验结果表明,TWFS 算法在算法有效性、公

* Supported by the National Natural Science Foundation of China under Grand No.60372065 (国家自然科学基金); the Key Laboratory of Broadband Optical Fiber Transmission and Communication Networks, Ministry of Education of China at UESTC (宽带光纤传输与通信网技术教育部重点实验室(电子科技大学)开放课题)

Received 2005-07-07; Accepted 2005-08-15

© 中国科学院软件研究所 http://www.jos.org.cn

平性和实现复杂度之间取得了很好的平衡,因而特别适合于 SUPAENT 中的 EPFTS 高速交换节点.

关键词: 调度算法;基于时槽加权的公平调度;物理帧时槽交换;单物理层用户数据传输平台体系结构网络;输入排队

中图法分类号: TP393 文献标识码: A

Internet 主干网络带宽目前正由 Gbps 级别向 Tbps 级别发展,新型多媒体业务如 VoIP,IPTV 等在网络上的不断出现及广泛应用对网络传输速率与服务质量(quality of service,简称 QoS)提出了更高的要求.光通信技术的发展已解决了网络节点之间的链路速率问题,因此,使交换机/路由器支持高速率处理与 QoS 保证能力是实现下一代因特网的瓶颈之一.交换机/路由器中负责将输入链路上的数据转发到输出链路的部件称作交换结构(switching fabric).传统的基于输出排队的调度策略虽然具有较好的 QoS 保证能力,但要求交换结构具有 N 倍的加速比,因而可扩展性差,不适应高速网络环境;由于基于开关交换矩阵(crossbar)的交换结构采用输入排队使交换矩阵和缓存结构工作于在线速率上,已经成为今天骨干网络上高速交换设备的首选.现有研究表明,可以通过 VOQ(virtual output queue)技术来解决输入排队的 HOL(head of line)问题[1],可使开关交换矩阵达到 100%的吞吐率,但如何提高输入排队调度策略的 QoS 能力一直是业界面临的问题之一.在 Crossbar 结合 VOQ 的体系结构下,本文提出了一种基于时槽加权的公平调度算法(timeslot weighted fair scheduling algorithm,简称 TWFS).仿真结果表明,无论在均匀或突发分布的业务流负载下,TWFS 算法均可取得渐近 100%的吞吐率,并且能有效区分不同输入输出对之间的带宽需求,具有较好的 QoS 保证能力.

1 研究背景

本文的研究背景是西南交通大学研发中的单物理层用户数据传输平台体系结构网络(single physical layer user-data platform architecture network,简称 SUPANET)[2,3]中的新型交换技术——类以太网物理帧时槽交换(ethernet-like physical frame timeslot switching,简称 EPFTS)[4,5].EPFTS 是实现 SUPANET 的关键技术,结合了 DWDM 技术以光纤传输数据与电域内对数据进行交换与处理的技术特点,以类“以太网 MAC 帧”格式构建物理帧(ethernet-like physical frame,简称 EPF),将传输一个 EPF 的时间——“物理帧时槽(PFT)”作为复用物理信道(波长或线路)和进行交换的基本单元.采用面向连接的用户服务模式,EPFTS 为业务数据的传输提供两类虚线路(virtual line,简称 VL)服务——交换虚线路(switched virtual line,简称 SVL)或永久虚线路(permanent virtual line,简称 PVL)服务.SUPANET 网络中终端用户在传递业务数据之前,需要先经过虚线路建立阶段.在建立连接阶段,信控平台通过用户与网络服务提供者之间对服务质量的协商并建议一条端到端的虚通路(virtual path——由沿途虚线路联接而成,简称 VP).对要求保证数据传输服务质量的业务流,EPFTS 一旦承诺协商约定的服务质量,将以时槽为基础,为该业务流在单个波长内预订并保证单位时间内需要传输的时槽数.因此,如果能够严格控制各交换节点单位时间内每条虚线路的数据吞吐率,尽量减少时延抖动,则能够控制实时业务流数据传输质量.因此,基于 EPFTS 交换技术的特点研究高速数据调度策略正是本文研究的出发点.

在 SUPANET 中 EPFTS 交换域内,任意时刻各交换节点上业务流的数量及各自性能需求参数可知,也即各交换节点的业务流到达速率将是可知的.以下我们约定交换节点上各端口的最大吞吐带宽相同,记为 F 个时槽,并始终假设交换节点的规模为 $N\times N$,统一地以矩阵 S 来表示各交换节点上输入输出对的带宽需求:$S=(s_{ij})_{N\times N}$,其中 s_{ij} 表示输入端口 i 到输出端口 j 的所有业务流所需时槽数之和.例如,若采用业务流平均到达速率直接映射为其带宽需求的方案,则由业务流平均到达速率矩阵(记为 $R=(r_{ij})_{N\times N}$,其中 r_{ij} 表示入口 i 到出口 j 的聚合流到达速率)很容易地得到带宽需求矩阵 S,其中 $s_{ij}=r_{ij}\times F(1\le i,j\le N)$.若带宽需求矩阵 S 满足 $\sum_{i=1}^{N}s_{ij}\le F$, $\sum_{j=1}^{N}s_{ij}\le F$, $1\le i,j\le N$,则称其是允许的,否则称其是非允许的.

© 中国科学院软件研究所 http://www.jos.org.cn

2 相关研究

基于 Crossbar+VOQ 结构采用输入排队的调度算法主要解决的是二分图的快速匹配问题,其中一类算法采用根据帧/分组排队状态进行调度的策略,典型的如 iSlip(iteration round-robin match with slip)[6],由于其实现复杂度低,已进入实际应用.iSlip 算法采用迭代形式,每次迭代包含 3 个步骤:

1) 请求(request):所有未匹配的输入端向有数据请求的输出端发送连接请求;
2) 响应(grant):尚未匹配的输出端在接受到多个连接请求后,从当前指针指向的输入端起,按轮询的原则选择排在最前面的输入端,批准其请求,并通知每一个向它发送请求的输入端其请求是否被接受.只有在第 3 步中,这个批准被相应的输入端确认后,并且这是第一次迭代,轮询指针才由当前位置向后移动一个位置;
3) 确认(accept):每个输入端在接受到多个批准后,根据其自身指针指向的位置开始,选择排在最前面的输出端,向该输出端发送确认信号,并且将指针移动到下一个位置.

iSlip 算法一般在迭代 $\log_2 N$ 次后即收敛,时间复杂度为 $O(\log_2 N)$,算法运行时只需各入口上虚拟输出队列是否为空的 1 位信息,因此空间复杂度为 $O(N^2\times 1)$ bit.尽管 iSlip 算法简单、系统吞吐率高,但无法有效区分不同业务连接的带宽需求,难以满足业务流对服务质量的不同要求特性.

另一类输入排队调度算法采用基于平均到达速率矩阵分解的调度策略,典型的有 Idling-HRR (idling hierarchical round robin)[7],BvN-switch(Birkhoff-von neumann switch)[8,9].在文献[7]中,作者将应用业务分为 CBR(constant bit rate)及 Best Effort 两种类型.根据每条 CBR 业务的传输速率,可得到各输入输出对上聚集流的带宽需求,选取合适帧长 f(个时隙),通过 Slepian-Duguid Algorithm 算法计算出一个规模为 $N\times f$ 的时隙分配矩阵(slot assignment matrix),该矩阵每个列向量对应一个时隙的交换矩阵的匹配模式.以此矩阵作为分组调度依据,每 f 个时隙循环一次,每个周期内各输入输出对获得的匹配次数与其需求带宽相当,从而保证其带宽需求.图 1 所示是在帧长 f=6,N=3 时,左边的需求矩阵 R 及其对应的 3×6 阶时隙分配矩阵(阴影部分),在时隙 k=4 时,匹配模式为 1-2,2-3,3-1.另外,Idling-HRR 算法选择在时隙分配矩阵空闲位置以及当入口 CBR 业务等待队列为空时,该入口可调度 Best Effort 业务数据.Idling-HRR 算法实际上由两个阶段来完成:第 1 阶段是计算时隙分配矩阵阶段,平均耗时为 $O(N^2 f)$;第 2 阶段是按顺序选择匹配模式,时间复杂度为 $O(1)$.

$$R=\begin{bmatrix}2 & 3 & 0\\ 1 & 0 & 3\\ 2 & 1 & 2\end{bmatrix}$$

Slot k	1	2	3	4	5	6
Input 1		1	1	2	2	2
Input 2	1			3	3	3
Input 3	2	3	3	1	1	

Fig.1 Sample of request matrix and appropriate slot assignment matrix

图 1 需求矩阵与时隙分配矩阵示例

类似地,BvN-switch 调度机制也分为两个阶段完成.第 1 阶段假设交换节点上平均到达速率矩阵 $R=(r_{ij})_{N\times N}$ 已知且是允许的.根据 Birkhoff 及 von Neumann 分解定理,最多存在 $K(\le N^2-2N+2)$个实常数 ϕ_i 和 K 个置换矩阵 $P_i, i=1,2,\ldots,K$,可将速率矩阵 R 的双随机矩阵 $\tilde{R}$ 表示为如下的线性代数和:

$$\tilde{R}=(\tilde{r}_{ij})_{N\times N}=\sum_{i=1}^{K}\phi_i P_i,\quad \sum_{i=1}^{N}\phi_i=1\ (\text{满足}\ 0\le r_{ij}\le\tilde{r}_{ij}\le 1,\quad \sum_{i=1}^{N}\tilde{r}_{ij}=1,\quad \sum_{j=1}^{N}\tilde{r}_{ij}=1).$$

每个调度时间片内,调度器仅需要考虑选择其中一个置换矩阵,据此对业务流进行调度和控制开关交换矩阵的连通模式.因此,BvN-switch 将到达速率矩阵分解后,使业务流的调度问题演变为 K 个置换矩阵的调度问题.在第 2 阶段,BvN-switch 采用了类似于 WFQ(即 P-GPS[10])的调度机制,以系数 ϕ_i 为权值对置换矩阵 P_i 进行调度.BvN-switch 调度机制的时间复杂度在第 1 阶段为 $O(N^{4.5})$,第 2 阶段为 $O(\log_2 N)$.

尽管与 Idling-HRR 相比,BvN-switch 调度机制取消了对帧长的依赖,空间复杂度大为降低,但两种调度算法都具有相同的特点,即当交换节点业务到达过程变化时,两者都需要重新运行算法的第 1 部分,由于时间复杂度

© 中国科学院软件研究所 http://www.jos.org.cn

较高,在业务负载频繁变化的环境中,算法将不能快速适应网络负载变化的需要.因此,为了适应业务到达速率矩阵的快速变化的场合,本文提出一种新的基于业务矩阵的公平调度算法,力求在满足对不同业务流进行区分服务的同时,能够快速地对系统中业务到达速率的变化作出响应.

3 TWFS 调度算法

根据 EPFTS 交换技术的特点,由应用业务发起的连接请求在服务质量协商阶段需要向网络服务提供者即沿途交换节点递交服务质量需求参数,其中包括该连接所需物理帧时槽数量.被网络接受的连接请求,其相应的服务质量需求参数如预约带宽、时延抖动上限等参数将会保存在沿途各 EPFTS 交换节点上.因此,当各虚连接建立后,EPFTS 交换节点上每一输入输出端口对上的物理帧时槽需求总数量 s_{ij} 是可知的,TWFS 调度算法的出发点,就是通过保证交换节点上每一输入输出端口对内聚集流对物理帧时槽的总体需求数量来保证业务流的带宽需求.

设交换节点上各输入输出对的带宽需求已知,也即带宽需求矩阵 $S=(s_{ij})_{N\times N}$ 是确定的.记 s_i 为输入端口 i 上业务流时槽需求数量之和,s_j 为输出端口 j 上业务流时槽需求数量之和,$1\le i,j\le N$.调度算法根据以下 3 个方面的参数来运行:

① 当前负载最大的端口时槽需求数量记为 f=maximum$\{s_i,s_j\}$.f 需随端口负载的变化而同步更新;

② 权重矩阵记为 weight_matrix=$(w_{ij})_{N\times N}$,时刻 0 时权重系数 w_{ij} 赋值 $w_{ij}^0 = s_{ij}$

③ 允许匹配标记矩阵记为 mark_matrix=$(m_{ij})_{N\times N}$,时刻 0 时所有标记符 m_{ij} 全赋值 1.

在 TWFS 调度算法中,当 $m_{ij}>0$ 时,称输入端口 i 允许向输出端口 j 发送匹配请求.以上参数将分布存储在调度器的输入端与输出端上,也即输入端 i(或输出端 j)将保存参量 f 以及对应参数 s_{ij},w_{ij} 和 m_{ij},j=1,…,N(或 i=1,…,N).算法运行时分两个阶段进行:

第 1 阶段与 iSlip 算法类似,以迭代的形式进行,迭代开始前所有端口设置为匹配未结束状态.迭代过程包含以下 3 个步骤:

1) 请求(request):所有匹配未结束的输入端,在本地寻找一个允许匹配并且权重最大的未匹配输出端:若存在,则向该输出端发送请求;否则设置本输入端为匹配结束状态.
2) 响应(grant):若有输入端匹配请求到达,则输出端在所有的请求入口中寻找权重最大者,批准其请求;并且,输出端通知所有输入端其请求是否被批准;输出端将对应的标记矩阵元素 m_{ij} 值减少 1.
3) 确认(accept):输入端接收到响应信号后,判断匹配请求是否被接受,若是,则设置本输入端匹配结束,并记录与之匹配的输出端口号,将对应的标记矩阵元素 f_{ij} 值减少 1;否则,设置发送响应信号的输出端为已匹配状态.

在第 1 阶段中,当输入端 i(或输出端 j)需要选择权重最大者时,各输出端 j(或输入端 i)的权重值将按如下公式计算:权重=$m_{ij}\times f+ w_{ij}^k$,k 是指第 k 个时隙.注意到 f 的值总比 w_{ij}^k 的值要大,因此通过权重比较相当于优先选择 m_{ij} 的值最大者,m_{ij} 值相同时,则选择 w_{ij}^k 最大者.

第 2 阶段为权重更新阶段.当第 k 个时槽调度结束时,权重矩阵需按以下规则进行更新: $w_{ij}^k =(w_{ij}^{k-1} +s_{ij})$,如果更新后 $w_{ij}^k>f$,则相应的 m_{ij} 值增加 1,并且 $w_{ij}^k = w_{ij}^k \mod f$(其中 mod 为取模运算).

显然,TWFS 调度算法在第 1 阶段迭代过程中迭代次数最多为 N,为了测试实际运行时算法的收敛速度,我们用 C 语言实现了 TWFS 算法,当交换节点规模 N 分别取值 4,8,16,32,64,128 时进行实验,每次实验中选择随机生成带宽需求矩阵 S,调度算法每次运行 1 000 000 时隙.每次实验中,对迭代次数进行统计,取 3 次运行的结果进行平均,结果见表 1.

© 中国科学院软件研究所 http://www.jos.org.cn

Table 1 Statistics of iteration count in TWFS

表 1 TWFS 调度算法迭代次数统计

N	k	Ratio of $k \le \log_2 N$ (%)	Max(m_{ij})
4	1.93	90.55	2
8	2.24	99.53	2
16	2.61	99.67	2
32	3.15	99.80	2
64	3.67	100	2
128	4.33	100	2

从表 1 可以看出,在 99%以上的交换时隙中,调度算法第 1 阶段迭代 $\log_2 N$ 次即收敛,N 越大,这一比例越高.因此,在实际应用中可设定 TWFS 调度算法的迭代次数为 $\log_2 N$.表 1 中第 4 列记录参数 m_{ij} 在算法运行过程中可能出现的最大值,各次实验结果中,其值不超过 2.综上所述,TWFS 调度算法在时间和空间上的复杂度如下:(1) 更新参数 f 的时间为 $O(\log_2 N)$;(2) 迭代过程用时为 $O(\log_2 N)$;(3) 参量 f、带宽需求参数 s_{ij}、权重参数 w_{ij} 以及匹配标识符 m_{ij} 将分布存储在调度器的输入端与输出端上,若设参数 f,s_{ij},w_{ij} 宽度为 l bit,则每个端口上存储以上参数需占用空间 $N\times(l+l+2)+l$ bits,总存储空间为 $N^2\times(4l+4)+2N\times l$ bits.

表 2 是算法 TWFS 与 Idling-HRR,BvN-switch 以及 iSlip 在时间和空间上的复杂度比较.与 Idling-HRR 以及 BvN-switch 调度机制相比,TWFS 算法中参数 f 的更新过程所需计算过程简单,易于硬件实现,更适合于高速网络环境中,但在调度阶段,由于 TWFS 调度算法分为迭代过程和权重更新过程两步,而且迭代过程涉及到权重比较运算,在实际应用中实现难度将高于 BvN-switch 算法和 iSlip 算法.

Table 2 Comparison of scheduling algorithm complexity

表 2 调度算法时空复杂度比较

Algorithms	Updating time	Scheduling time	Space complexity
iSlip	None	$O(\log_2 N)$	$O(N^2)$
Idling-HRR	$O(N^2 f)$	$O(1)$	$O(f\ N\log_2 N)$
BvN-switch	$O(N^{4.5})$	$O(\log_2 N)$	$O(N^3\log_2 N)$
TWFS	$O(\log_2 N)$	$O(\log_2 N)$	$O(N^2 4l)$

4 性能仿真

在本节中,我们用仿真软件建立了一个规模为 8×8 的交换节点仿真模型,端口编号为 0~7,采用定长帧/分组交换方式,对以上几类调度算法的有效性、业务流区分能力进行评测.由于 Idling-HRR 与 BvN-switch 算法较为相似,因此我们只选择了 iSlip,BvN-switch 与 TWFS 调度算法进行比较.所有仿真实验中,业务源都采用贝努里到达过程,交换出口根据指定的概率分布产生,设置 VOQ 中所有子队列容量为 1 000 个帧/分组.

4.1 一致贝努里数据流到达模式下的性能测试

一致贝努里业务模式下,各输入端口(以下简称入口)到达的帧/分组出口满足均匀分布概率特性,仿真实验在不同系统负载条件下进行,统计 3 种调度算法下,分组在调度过程中经历的平均时延和最大时延以及系统吞吐率.

图 2 中左图是 3 种调度算法在不同系统负载下的平均时延(实线部分)及最大经历时延(虚线部分),右图所示为调度算法系统吞吐率.由图 2 可知:各种负载条件下,TWFS 与 BvN-switch 调度机制性能接近;iSlip 算法在系统负载低时延时性能较为优越,但系统负载高时,并无优势.另外,由图 2 右图可知,各调度算法在系统负载不超过 99.5%时,系统吞吐率为 100%.因此,系统负载不超过 99.5%时,各调度算法均处于稳定状态,所取得的时延统计结果可信.

© 中国科学院软件研究所 http://www.jos.org.cn

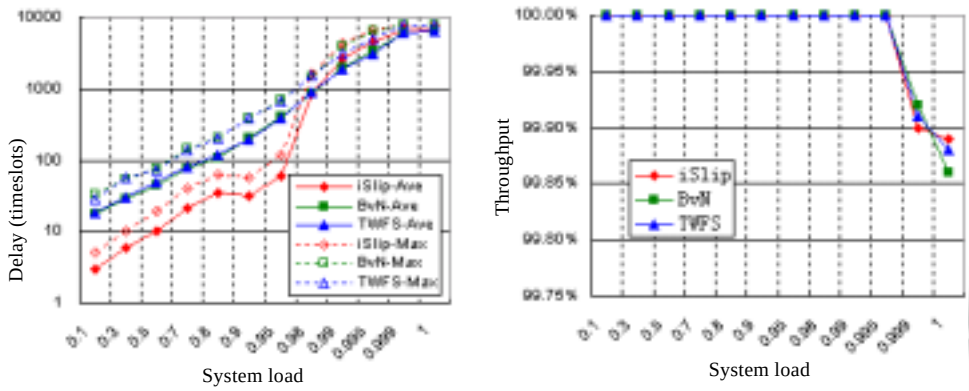

Fig.2 Simulation result under Bernoulli i.i.d and uniform distribution

图 2 一致贝努里模式下仿真实验结果

4.2 在突发性数据流到达模式下的性能测试

为了比较 3 种调度算法对不同输入输出对上的业务的区分服务能力,我们设计了如下两种应用场景:

1) 0 号~7 号入口都产生发往 0 号输出端口的信元.在 BvN-switch 及 TWFS 调度算法中,0 号输出端口(以下简称出口)按如下比例为各入口分配带宽:s_0={10%,25%,11.25%,10%,10%,10%,10%,10%};但在仿真实验中,让 6,7 号入口产生比预约带宽分别多 2 倍和 1 倍的业务流,即分别设置它们的负载比例为 30%和 20%,以制造 0 号出口超载的现象.

2) 从入口 i 到达业务数据中,按(11/18=0.6111111)的比例发往相应出口 i,其余数据均匀分布到其他出口.仿真时设置各端口负载为 90%.

图 3 是突发场景 1)下的仿真实验结果,左图显示出 0 号输出端口上记录的由各入口到达业务数据经历的平均时延,右图是各入口实际吞吐率相对于其预约带宽的变化率,负值表示减少的比例,正值表示超过的比例.综合图 3 可知,TWFS 调度算法有效隔离了不同业务流的带宽需求,并且保证了正常业务(0~5 号入口)不受到异常业务(6 号与 7 号入口)的影响,使正常业务的时延保持在合理的范围内;BvN-switch 调度机制倾向于优先服务预约带宽高的业务,从而影响了其他带宽要求相对较低的业务流的吞吐率与时延;iSlip 算法则根本无法有效区分不同业务的带宽需求,带宽要求高的业务流受到异常业务的干扰导致时延性能很差.

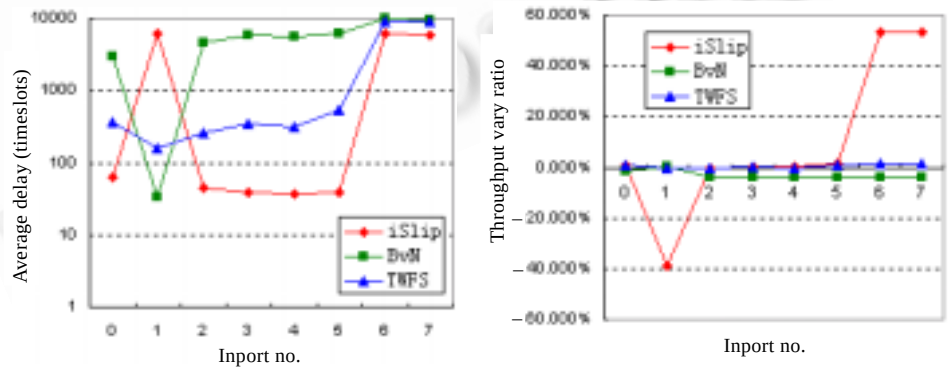

Fig.3 Average delay and throughput deviation ratio of data arrived at out port 0 under pattern 1)

图 3 场景 1)下各入口发往 0 号出口的数据经历的时延及吞吐率变化

图 4 是在突发模式 2)下进行仿真实验 1 小时,交换系统在 3 种调度算法下分组经历的调度时延以及系统吞吐率随时间变化的示意图,其中时延取各采样周期中所有样本的最大值.

© 中国科学院软件研究所 http://www.jos.org.cn

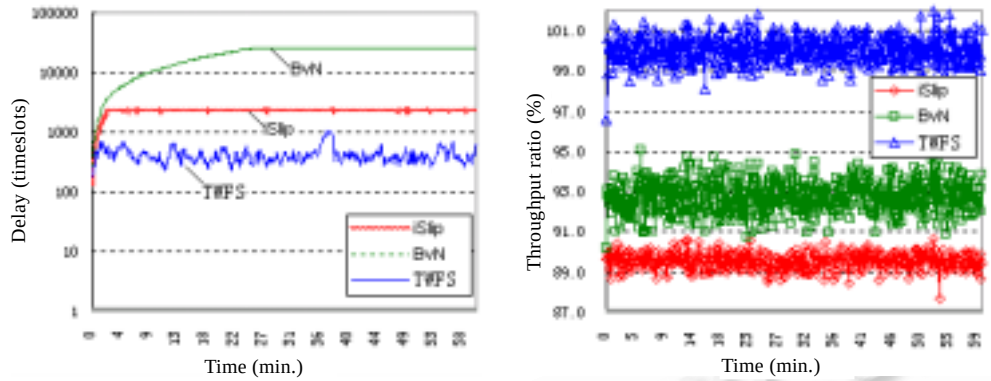

Fig.4 Comparison of delay and throughput under three algorithms

图 4 突发场景 2)下分组时延及系统吞吐率比较

实验结果表明,iSlip 以及 BvN-switch 调度算法系统吞吐率小于 100%,原因在于部分队列等待调度时间长,分组迅速充满缓存从而引起系统丢帧.TWFS 调度算法下各输入输出对的业务吞吐率为 100%,时延也保持在 1 000 个时隙之内.我们还发现场景 2)中 BvN-switch 的拥塞情况比 iSlip 算法严重,这一现象可由图 5 来解释.

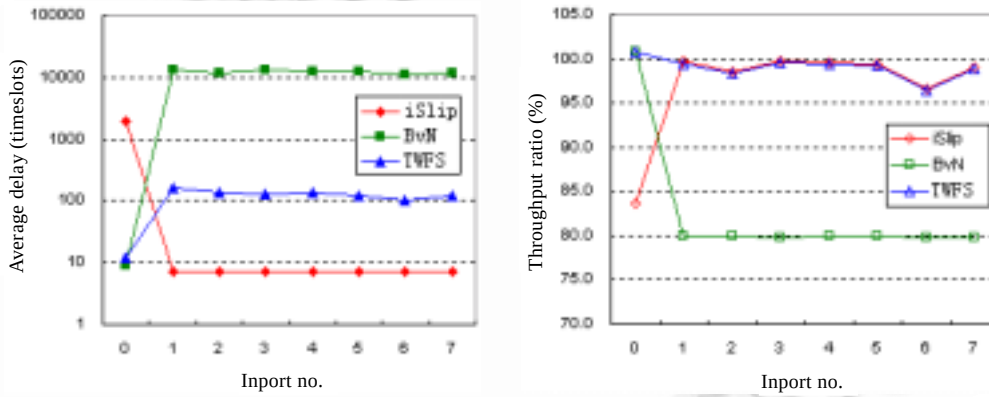

Fig.5 Average delay and throughput of data arrived at out port 0 under pattern 2)

图 5 场景 2)下各入口发往 0 号出口的数据经历的时延及吞吐率变化

图 5 显示了在 0 号出口上对来自不同入口的分组的时延和吞吐率统计结果,进一步表明了 TWFS 算法在时延及系统吞吐率上的优越性;同时也映证了上一节中我们关于 BvN-switch 倾向于分配带宽高的业务而导致相对带宽低的业务流服务质量普遍较差的特点,表现在图 5 上就是 0 号入口到 0 号出口的业务流服务质量明显好于其他入口到 0 号出口的业务流;而 iSlip 算法正好相反,因占用带宽低的业务服务质量优先得到满足,相对带宽要求高的业务性能则无法满足,正如图 5 中显示的 0 号入口到 0 号出口的业务流服务质量明显低于其他入口到达的业务流.

5 结 论

由于 EPFTS 本身的技术特点,所有允许入网的业务流(对应的是虚连接)向沿途交换节点递交了其性能需求参数,其中包含了最重要的参数预约带宽即单位时间内需要保证的物理帧时槽数量.因此,各交换交点易与基于本地所有活动的虚连接形成带宽需求矩阵,并随连接的建立和拆除对其保持同步更新,这正成为 TWFS 调度算法进行调度考虑的出发点.TWFS 算法结合了 iSlip 与 BvN-switch 两类调度机制的特点,一方面类似于 BvN-switch 以保障不同输入输出对之间的公平为目标,另一方面在调度过程中如同 iSlip 算法采用快速迭代的过程,以适应高速交换网络中交换节点内对分组进行快速仲裁的需要.仿真结果表明,TWFS 算法综合了系统吞

© 中国科学院软件研究所 http://www.jos.org.cn

吐率高、异类业务区分服务能力强、算法复杂度低的特点,即使当业务负载发生变化时,调度参数 f 的更新计算复杂度也很低,速度也很快,可以与一次调度过程同步完成,因此对系统整体性能影响小.

TWFS 算法是我们在 EPFTS 交换技术的研究过程中提出来的,算法仿真中关于 f_{ij} 的值不超过 2 的现象需要进行深入分析.另外,调度机制如何提供同一输入输出对内部不同微流之间的服务公平性和业务区分能力,也是我们下一步考虑的方向之一.

References:

[1] Anderson T, Owicki S, Saxe J, Thacker C. High-Speed switch scheduling for local-area networks. ACM Trans. on Computer Syst., 1993,11(4):319−352.

[2] Zeng HX, Dou J, Xu DY. Single physical layer *u*-plane architecture (SUPA) for next generation Internet. Comprehensive Report on VoIP and Enhanced IP Communications Services, IEC, 2004. 197−227.

[3] Zeng HX, Dou J, Xu DY. Replace MPLS with EPFTS to build a SUPANET. In: LI VOK, Hamdi M, eds. Proc. of the HPSR 2005. Hong Kong: IEEE Press, 2005.

[4] Zeng HX, Xu DY, Dou J. On physical frame time-slot switching over DWDM. In: Fan PZ, Shen H, eds. Proc. of the PACAT 2003. Chengdu: IEEE Press, 2003. 286−291.

[5] Zeng HX, Xu DY, Dou J. Promotion of physical frame timeslot switching (PFTS) over DWDM. Annual Review of Communications, 2004,57,809−826.

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

[7] Hung A, Kesidis G, Mckeown N. ATM input-buffered switches with guaranteed-rate property. In: Kristine L, ed. Proc. of the IEEE ISCC'98. Athens: IEEE Press, 1998. 331−335.

[8] Chang CS, Chen WJ, Huang HY. On service guarantees for input buffered crossbar switches: A capacity decomposition approach by Birkhoff and von Neumann. In: Bhatti SN, Crowcroft J, Diot C, eds. IEEE IWQoS'99. London: IEEE Press, 1999. 79−86.

[9] Chang BCS, Chen WJ, Huang HY. Birkhoff-Von neumann input buffered corssbar switches. In: Sidi M, ed. Proc. of the IEEE INFOCOM 2000. Tel Aviv: IEEE Communications Society, 2000. 1614−1623.

[10] Parekh A, Gallager R. A generalized processor sharing approach to flow control in Integrated services networks: The single-node case. IEEE/ACM Trans. on Networking, 1993,1(3):344−357.

李季(1978−),男,安徽怀宁人,博士生,主要研究领域为计算机网络体系结构,高速路由与交换技术.

曾华燊(1945－),男,研究员,博士生导师,主要研究领域为下一代网络体系结构,高速交换技术,网络测试技术.

© 中国科学院软件研究所 http://www.jos.org.cn