在CICQ交换结构下实现分布式的WFQ类加权公平调度算法

王 荣 陈 越 邬江兴
(国家数字程控交换技术研究中心 解放军信息工程大学 郑州 450002)

摘 要 传统的基于 crossbar 的输入排队交换结构在提供良好的 QoS 方面存在很大的不足，而 CICQ(Combined Input and Crosspoint buffered Queuing)交换结构与传统的交换结构相比，不但能在各种输入流下提供接近输出排队的吞吐率，而且能提供良好的 QoS 支持。该文基于 CICQ 结构，提出了在输入排队条件下实现基于流的分布式 WFQ 类分组公平调度算法的方案，并通过仿真验证了这一方案的有效性。

关键词 CICQ 交换结构，WFQ 类调度算法，输入排队
中图分类号：TP393.07 **文献标识码**：A **文章编号**：1009-5896(2006)05-0805-05

Implementing Distributed Weighted Fair Scheduling Algorithm in CICQ Switches

Wang Rong Chen Yue Wu Jiang-xing
(*NDSC, PLA Information Engineering University, Zhengzhou* 450002, *China*)

Abstract Traditional input-queued switches based on crossbar are insufficient in providing good QoS performance. As a contrast, the CICQ(Combined Input and Cross-point buffered Queuing) switches can provide almost 100% throughput under different input traffic, the performance of which is very close to the OQ(Output-Queued) switch, and has the potentials to support good QoS. Based on the CICQ switches, a new scheme is put forward, which can realize distributed weighted fair schedule for the packets of variable length, and have both the scalability of input-queued switches and QoS performance of output-queued switches. The issue of updating the virtual time of back-pressured queues is also discussed. Simulation results show the scheme is very effective and has good performance.

Key words CICQ switch, GPS based(WFQ) packet fair scheduling algorithm, Input-queued switch

1 引言

目前 Internet 已经历了高速发展的阶段，主干链路的带宽已经达到 10Gbps，网络规模早已遍布世界各地。但是随着带宽的高速增长，IP 网络的关键问题——QoS 问题却一直没有得到很好的解决。在未来的 Internet 网络中，如果要对视频、语音和数据业务提供良好的支持，则必须解决 QoS 问题。

在高速IP网络中，带宽和QoS似乎是一对矛盾。在IETF提出的综合服务(Int-serv)框架中，保证服务可为单个流提供有严格端到端时延和低分组丢失率的电路型服务，这种服务需要在路由器中实现基于流的加权服务公平调度。WFQ(PGPS)[1], WF^{2}Q, SCFQ[2]等调度算法可实现基于流的加权公平调度，但这类算法都是在输出排队的环境下实现的。输出排队虽然可以提供良好的QoS支持，但难以在高速网络环境中应用。因为在输出排队的交换结构中，对于$N\times N$的交换结构，需要输出缓存和交换结构的加速比为N，受商用随机存贮器访问速率的限制，在高速交换结构中一般采用输入排队的交换结构。输入排队的交换结构对存贮器的带宽要求仅为输入链路速率的 2 倍，但在输入排队的交换结构下难以提供良好的QoS支持。在现有的基于输入排队的crossbar交换结构中，一般采用定长的分组交换方式，IP包在交换前先分割(segment)成定长的信元，交换结构的仲裁单元通过匹配算法找到输入和输出端的极大匹配，然后配置crossbar同步完成各端口间信元的交换。在这种交换方式下，为了实现100%的吞吐率，需要实现输入和输出端口的最大匹配或加权最大匹配[3]，实现QoS的要求和最大匹配的要求难以兼顾。现有的求极大匹配的输入排队调度算法如iSLIP等，其功能是通过迭代找到输入输出间的极大匹配，但难以提供相应的QoS支持，更无法提供基于流的带宽保证。

本文的讨论是基于一种新的交换结构CICQ(Combined Input and Cross-point buffered Queuing)[4], 这种交换结构在输入排队的情况，不但能够对i.i.d Bernoulli均匀流提供百分之百的吞吐率，还能对各种突发的、非均匀的流提供接近百分之百的吞吐率[5]，在吞吐率方面与输出排队(OQ)结构很接近，

2004-10-14 收到, 2005-04-07 改回
国家 863 高科技发展计划重点项目 (2003AA103510)资助课题

基于这种CICQ交换结构本文提出了实现分布式的WFQ类加权公平调度算法的方案。这里所谓的分布式是指各个输入和输出端口的调度器之间不需要相互间的复杂的协作和通信，而是相互之间相对独立的工作，从而具有实现的简单性和可扩展性。

2 CICQ 结构及其实现

2.1 CICQ 结构的特点

传统的crossbar交换结构，其缓存位于交换结构的输入或输出端口，在crossbar交换结构中是没有缓存的。而CICQ交换结构如图 1 所示，除了在输入端有缓存外，在每个交叉点上(crosspoint)都有一个小的缓存，称为交叉点缓存。对于$N\times N$的交换结构而言， 一共有N^2个交叉点从而有N^2个缓存，这种crossbar又称buffered crossbar[5]。在CICQ交换结构的输入端和传统的输入排队crossbar一样采用虚拟输出排队结构(VOQ)，以消除队头阻塞。

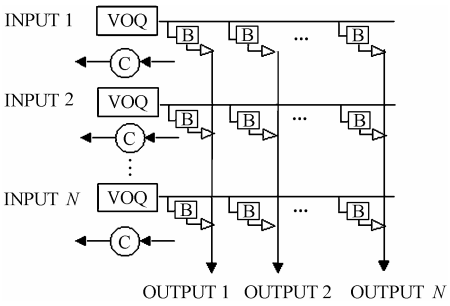

图 1 Buffered crossbar 交换结构

Fig.1 Buffered crossbar switch fabric

CICQ 结构和传统 crossbar 结构的不同导致两者有显著的区别：

(1)传统的 crossbar 由于没有交叉点缓存，所有输入和输出端的连接按照事先确定好的配置同步进行操作。每个输入端只能连接到一个输出端，每个输出端也只能连接到一个输入端。而 buffered crossbar 由于在每个交叉点上有缓存，所有输入和输出端不必同步的进行交换，每个输入端和每个输出端可以相互独立地与交叉点缓存进行交换。这样减少了同步所带来的问题，例如在同步情况下，每个输入端都属于不同的时钟域，各个时钟域间的信号需要进行同步。

(2)传统 crossbar 由于同步地进行交换，需要一个专门的调度算法，实现输入和输出端的最大匹配，这是一个双向图匹配问题，其算法复杂度为 $O(n^{2.5})$ 。在实际使用中为了在一个信元的时间内，求出匹配矩阵，一般采用求极大匹配的迭代算法。当链路速率提高，信元时间变短时，性能良好的匹配算法就难以实现，匹配算法的运算时间成为制约因素。而在 buffered crossbar 结构中，由于取消了输入和输出的同步限制，每个输入和输出端独立地操作，从而不需要实现输入和输出端的匹配算法，使交换结构在更高的速率下运行并支持良好的 QoS 成为可能。

(3)传统的 crossbar 需要同步的完成各端口的交换，输入的变长 IP 分组需要分割成定长的信元，这种定长的信元会浪费一部分带宽，从而要求交换结构有一定的加速比以补偿这部分带宽。对 buffered crossbar 结构就不存在这种问题，CICQ 结构可以直接实现变长的分组交换，交换结构的带宽可以得到更充分的利用。

(4)传统的基于输入排队结构的 crossbar 难以提供良好的 QoS 支持，像 WFQ 这样的基于流的加权公平调度算法难以实现。而在 buffered crossbar 交换结构中可以实现更好的 QoS 性能，本文就是讨论在输入排队的 CICQ 结构下，实现分布式的 WFQ 类加权公平调度方案。

2.2 CICQ 结构的实现

如果把输入缓存全部设置在crosspoint中，由于节点数是N^2，这样的结构显然难以实现。可以考虑只在crosspoint上设置小的缓存，把大量缓存设置在输入端的VOQ队列中，通过适当的流控机制，使crosspoint中的缓存不会溢出(图 1)。

图 2 示出每个 crosspoint 的参考逻辑结构，存贮器由一个双端口的 SRAM 组成，因为要同时完成读写操作，所以使用双端口 SRAM。SOP 表示分组到来，EOP 表示分组结束，每个分组通过 w 位宽的总线接入。在每个分组的头一个字节中包含一个多播的位图，规定接收分组的交叉点缓存。由于写入和读出的时钟域不同，因此通过由两个 D 触发器组成的同步电路完成 new packet 信号到输出时钟域的同步。

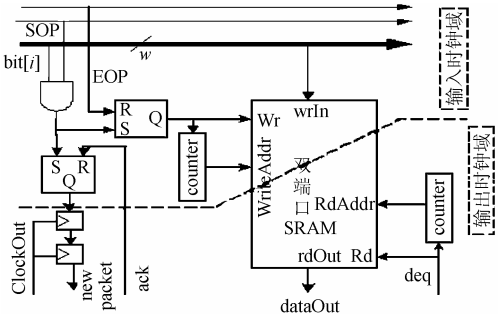

图 2 Crosspoint 的逻辑结构

Fig.2 Crosspoint logic structure

3 在 CICQ 结构下实现基于输入排队的分布式分组公平调度

3.1 WFQ 类的分组公平调度算法

WFQ算法最初是由Demers[1]等人提出的，该算法的基本思想是模仿相应的流体模型的通用处理器共享(GPS)服务器，在流体模型中，假设各个队列无限可分，服务器可以同时服务所有的队列。设 ϕ_1， ϕ_2，…， ϕ_N 为正实数， $S_i(\tau,t)$ 为服务器在(τ,t)时间内给流i提供的服务，GPS服务器定义为对每个在(τ,t)期间队列长度＞0 的流，所提供的服务满足下式：

$$\frac{S_i(\tau,t)}{S_j(\tau,t)} \geq \frac{\phi_i}{\phi_j},\ j=1,2,\cdots,N \tag{1}$$

则对流 i 提供的服务的速率满足

$$r_i = \frac{\phi_i}{\sum_j \phi_j} r \tag{2}$$

GPS 服务器是一种理想的服务模式，每个流可得到所预约的服务速率。WFQ 调度算法在分组的环境下模仿 GPS 服务器，设在 GPS 服务器下分组的离开时间为 F_p，则 WFQ 按照各个流队头中分组的 F_p 的升序调度分组输出。在实际实现中，WFQ 算法用虚拟时间来模仿 GPS 服务器的服务时间。设在 GPS 服务器下每个分组到达或离开为一个事件，设 t_j 为第 j 个事件的时间，则在(t_{j-1},t_j)间隔内流的数目是固定的，定义这些流的集合为 B_j。考虑系统的任何一个忙期，设开始时间为 0，虚拟时间定义如下：

$$\left.\begin{aligned} &V(0)=0 \\ &V(t_{j-1}+\tau)=V(t_{j-1})+\frac{\tau}{\sum_{i\in B_j}\phi_i},\ \tau \leq t_j - t_{j-1},\ j=2,3,\cdots \end{aligned}\right\} \tag{3}$$

设第 i 个流的第 k 个包的到达时间为 a_i^k，包长为 L_i^k，则这个包开始和结束服务的虚拟时间 S_i^k 和 F_i^k 为

$$\left.\begin{aligned} S_i^k &= \max\{F_i^{k-1}, V(a_i^k)\} \\ F_i^k &= S_i^k + L_i^k/\phi_i \end{aligned}\right\} \tag{4}$$

在WFQ算法的基础上，人们提出了一系列类似的算法，如WF^2Q+[6]，SCFQ等，这一类算法统称为类GPS算法。这些算法之间的区别主要在两个方面：系统虚拟时间的计算和分组选择的策略。对系统虚拟时间计算的改进，使其计算复杂度下降，从而有利于算法的实现。按照对分组出发选择的策略有，最小开始时间优先(SSF)，最小结束时间优先(SFF)，最小可用结束时间优先(SEFF)等。总之，这一类算法都以模仿理想的GPS调度为目标，从而实现流之间的加权公平调度。

3.2　分布式 WFQ 类公平调度算法的实现

在输出排队结构中由于只有一个资源竞争点，即在输出端口，因此只需要在每个输出端口中实现一个WFQ公平调度器即可。在CICQ结构中，分组队列缓存在输入端口，在每个crosspoint缓存中也有分组，因此要想模仿输出排队结构，需要在多个竞争点进行资源调度[4]，即在每个输入端的VOQ队列中，交换结构的每个输入端口和输出端口中。如图 3 所示。

在每个VOQ队列V_{ij}中设置一个WFQ调度器$S_{i,j}$，负责调度每个VOQ队列中的所有流。同一输入端的所有调度器输出通过调度器 S_i^{in} 输出到crossbar的缓存中，crossbar中的所有到同一输出端的缓存，由调度器 S_j^{out} 调度输出。在上述

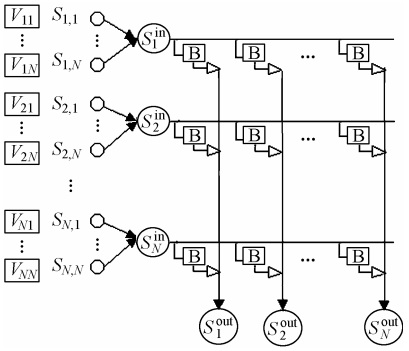

图 3　在 CICQ 结构下实现分布式 WFQ 调度算法构架

Fig.3 Distributed weighted fair queuing schedule in CICQ

各个调度器中为了实现分布式的调度，即各个VOQ队列中的调度器不需要互相通信和协作，各个输入端口和输出端口的调度器也不需要互相通信和协作。我们采用如下的调度策略：每个VOQ中的调度器$S_{i,j}$只对相应VOQ中的所有流进行调度；输入端口的 S_i^{in} 调度器，对输入端口i中的N个VOQ队列作为流进行调度；各个VOQ队列的权重等于相应VOQ队列中各个预约流的权重之和。如果一个VOQ队列作为整，体它的带宽得到保证，则其中的每个流的带宽也相应能得到保证。输出端口调度器 S_j^{out} 对到输出端口j的N个crosspoint缓存作为流进行调度。各个crosspoint缓存队列的权重等于相应输入端中到输出端j的VOQ队列中各预约流的权重之和。同样VOQ队列整体的带宽得到保证，从而VOQ队列中的各流的带宽也得到保证。上述的各调度器$S_{i,j}$，S_i^{in} 和 S_j^{out} 均采用WFQ类的调度器，可以是WFQ，WF^2Q+或SCFQ等。由于WF^2Q+具有类GPS调度算法中最好的性能，同时又有最小的实现复杂度，因此本文中假设各个调度器均采用WF^2Q+调度算法。

这种分布式的调度结构除了有分布式的特点外，在结构上还具有显然的优点。每个 VOQ 队列都有自己的调度器，这些调度器可以在每个输入端的线卡上实现，从而可以将大量的调度流的计算量分布到每个线卡上。而在交换结构中，每个输入端只对 N 个 VOQ 队列进行调度，每个输出端只对 N 个 crosspoint 缓存进行调度，调度队列的数目是固定的，这样可以大大减化调度算法的实现，便于用硬件实现，可以在一块芯片上实现交换结构的功能。这种结构具有工程实现上明显的优势。

4　反压队列的虚拟时间函数更新

在 CICQ 交换结构中，由于每个 crosspoint 缓存只能实现较小容量的缓存，大量的分组缓存在输入端口中，crosspoint 缓存和输入端缓存需通过反压信号进行协调控制。当一个 crossbar 缓存发出反压信号时，相应输入端口中的与该 crossbar 缓存相对应的 VOQ 队列就暂时不参与调度，这时该队列的虚拟时间函数维持不变；当反压信号解除时，该队列的虚拟时间函数如何更新，是一个需要考虑的问题。由于在反压期间，其它队列的虚拟时间函数都在增加，而反压

队列维持不变，在反压解除后，反压队列的虚拟时间函数较小，就会产生该队列被优先调度的现象。我们以下面的例子进行分析。

假设某一输入端口中有 3 个队列，队列 0 的权重为 r_0，队列 1 为 r_1，队列 2 为 r_2 (权重均归一化)。队列 0 的输出端口为 0，在交换机构的输出端口 0 还有其他队列与队列 0 竞争。用$V(t)$表示调度器的系统虚拟时间函数，$S_i(t)$表示队列i的开始虚拟时间函数，$F_i(t)$表示队列i的结束虚拟时间函数。设 3 个队列的分组长度均为L，假设在时刻 t_1 队列 0 被反压，按照WF^{2}Q+的调度策略，有 $S_0(t_1)<V(t_1)$。假设在时刻 t_2 反压被解除，在时间段$[t_1,t_2]$内，队列 0 和队列 1 有分组在等待，队列 1 没有被反压，队列 2 处于空闲状态，到 t_2 时刻队列 1 正好服务完一个分组。则此时段内，队列 1 一直被服务，系统虚拟时间的变化为 $V(t_2)-V(t_1)=L/r_1$。设在时刻 t_2，队列 2 到达一个分组P，有 $S_2(t_2)\ge V(t_2)$，队列 1 变为空闲，而队列 0 仍然有分组在等待。

$$\left.\begin{aligned} & S_2(t_2)\ge V(t_2)\Rightarrow F_2(t_2)\ge V(t_2)+L/r_2 \\ & \qquad\qquad \Rightarrow F_2(t_2)\ge V(t_1)+L/r_1+L/r_2 \\ & S_0(t_1)<V(t_1) \end{aligned}\right\}\Rightarrow$$

$$F_2(t_2)\ge S_0(t_1)+L/r_1+L/r_2\ge F_0(t_1)+L/r_1+L/r_2-L/r_0$$

因为在 t_1 到 t_2 期间队列 0 被反压，虚拟时间函数不变，有 $F_0(t_1)=F_0(t_2)$，则有 $F_2(t_2)-F_0(t_2)\ge L(1/r_1+1/r_2-1/r_0)$。若令 $r_2=r_0$，则有 $F_2(t_2)-F_0(t_2)\ge L\bullet 1/r_1$，即队列 2 的时延与 r_1有关，而队列 2 到达时，只有队列 0 和队列 2 参与调度。此时的时延应仅与自身的预约带宽有关，即与r_0和r_2有关。因此如果反压结束后，反压队列的虚拟时间函数保持不变，则会造成其它队列的时延增加。

针对以上问题，我们采用如下的办法更新反压队列的虚拟时间，当反压队列解除反压后，将反压队列的虚拟结束时间与系统虚拟时间比较，如反压队列的虚拟结束时间小于系统虚拟时间，则将队列的开始虚拟时间设为等于系统虚拟时间，并相应改变队列的虚拟结束时间；如反压队列的虚拟结束时间大于或等于系统虚拟时间，则维持不变。即有 $F_0(t_2)>V(t_2)$，由 $F_2(t_2)>V(t_2)+L/r_2$，有 $F_2(t_2)-F_0(t_2)>L/r_2$，可见队列 2 的时延不再与 r_1 有关。

5 仿真环境和仿真结果分析

我们模仿一个 4×4 的交换结构，每个端口的链路输入速率为 1Gbps, 每个输入端的流按照到不同的输出端排在不同的 VOQ 队列中，令 $f(i,j)$代表从输入端 i 到输出端 j 的流。

仿真 1 假定流 $f(1,1)$,$f(2,1)$,$f(3,1)$,$f(4,1)$的预约带宽分别为 40%，30%，20%，10%，且这 4 个流的到达速率都相同，其它流的到达速率为 5%。

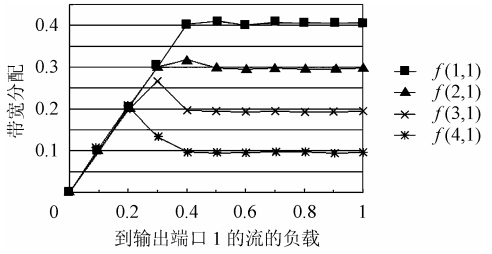

图 4 到输出端口 1 的各流的带宽分配

Fig.4 The bandwidth assignment of flows to output 1

我们改变 4 个流的输入载荷，考察在输出端的带宽分配，由图 4 可见，当输入的载荷在 25%以下时，各个流得到的带宽相同；当输入载荷大于 25%时，各个流分配到的带宽开始不同; 当输入载荷大于 40%时，每个流按照预约的带宽得到服务。我们来看当输入载荷在 25%和 40%之间的情形，当载荷为 30%时，流 $f(1,1)$的预约带宽为 40%，实际使用的只有 30%，剩余 10%的带宽，流 $f(3,1)$和 $f(4,1)$得到的带宽分别为 26.8%和 13.4%，分别多得到了 6.8%和 3.4%的带宽，可见剩余的带宽按照预约带宽作为权重在两个流之间分配，符合加权公平分配的原则。

仿真 2 假设 $f(1,1)$, $f(1,2)$, $f(1,3)$, $f(1,4)$的预约带宽为 40%，30%，20%，10%，$f(2,1)$, $f(3,1)$, $f(4,1)$预约的带宽同上，除 $f(1,1)$到达的速率为 300Mbps 外，其余各流到达的速率都为 400Mbps。由表 1 可见，在输入端口 1 的各竞争流中，剩余带宽按照预约的比例在各流中公平分配，在输出端口 1 的各竞争流中，剩余带宽按照预约的比例在各流中公平分配。

表 1 各业务流获得的额外带宽

Table 1 Over-allocated bandwidth of each flow

业务流	预约带宽 (Mbps)	吞吐率(Mbps)	额外获得的带宽(Mbps)
$f(1,2)$	300	350.26	50.26
$f(1,3)$	200	233.10	33.10
$f(1,4)$	100	116.62	16.62
$f(2,1)$	300	350.28	50.28
$f(3,1)$	200	233.12	33.12
$f(4,1)$	100	116.57	16.57

6 结束语

传统的 crossbar 交换结构广泛地用于各种现代路由器中，但传统的 crossbar 交换结构在提供良好的 QoS 方面存在着很大不足，本文在传统的交换结构基础上讨论了一种新的交换结构——CICQ 交换结构，这种交换结构相比传统的交换结构不但在各种输入流下能提供更好的吞吐率、更高的效率，其吞吐率接近输出排队交换结构，而且能够实现良好的 QoS 能力。本文提出了在 CICQ 交换结构下实现分布式的 WFQ 类加权公平调度算法的方案，并通过仿真验证了这一

方案的有效性；讨论了在存在反压信号的情况下，WFQ 类调度算法的虚拟时间函数更新的问题，并给出了解决的办法。

参 考 文 献

[1] 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.

[2] Zhang H. Service disciplines for guaranteed performance service in packet-switching networks[J]. *IEEE Commun. Mag.*, 1995, 83(10): 1374－1396.

[3] McKeown N, Anantharam V, Walrand J. Achieving 100% throughput in an input-queued switch[A]. Proc. InfoCom'96[C], San Francisco, CA, 1996: 296－302.

[4] Stephens D, Zhang H. Implementing distributed packet fair queueing in a scalable switch architecture. Proc. INFOCOM, 1998, [Online], Available: http://www-2.cs.cmu.edu/ hzhang/ papers/ INFOCOM98b.pdf

[5] Katevenis M, Passas G, D. Simos, Papaefstathiou I, Chrysos N. Variable packet size buffered crossbar (CICQ) switches. Proc. IEEE International Conference on Communications (ICC 2004), Paris, France, 2004: 255－278.

[6] Bennett J, Zhang Hui. Hierarchical packet fair queueing algorithms. In: Proc ACM SIGCOMM' 96, Stanford, CA, 1996: 143－156.

王　荣：　男，1968 年生，工程师，博士生，专业为 IP 网络 QoS、交换结构及调度算法.

陈　越：　男，1965 年生，副教授，博士生，专业为网络路由协议.

邬江兴：　男，教授，博士生导师，中国工程院院士，从事通信和计算机网格方面的研究.