文章编号:1004-5694(2004)01-0010-05

基于端到端时延保证的紧急分组优先算法*

周卫华,倪县乐,丁 炜
(北京邮电大学 宽带通信网络实验室,北京 100876)

摘 要:提出了一种能够提供端到端时延保证的多跳间时延协作Crossbar调度算法(紧急分组优先算法)。该算法以分组头中记录的剩余时延为权重对分组进行调度,通过控制分组在各跳上的时延不但能够保证分组的端到端时延,还能够平衡不同跳数分组的端到端时延。算法还能够使路由器避免维护每个流的状态信息以及对单个流进行复杂的队列管理和调度,由此增加了路由器的可扩展性。计算机仿真表明该算法具有较高的资源利用率,较低的端到端时延和时延抖动以及较低的分组丢弃率等特点。

关键词:紧急分组优先;交叉开关;时延抖动;分组丢弃率
中图分类号:TN393.0 文献标识码:A

0 引 言

当前的Crossbar调度算法可以分为2种:一种是基于轮循服务的公平调度算法,即尽可能公平地对每个VOQ进行调度,如iSLIP[1],PIM[2],2DDR[3]等;另一种是基于一定的权重对VOQ进行调度,如LQF[1](采用队列的长度为权重)和OCF[1](采用队头信元在该系统内的等待时间为权重)等。

基于多跳间协作的调度算法的目的是希望通过多跳间的协作调度来提供端到端的QoS保证。文献[4-8]对多跳间协作的必要性以及优点进行了理论上的分析。文献[9]提出了基于多跳间带宽协作的调度算法(CSFQ),并对其实现方案和性能进行了分析。该算法的主要思想是在分组头中记录该分组所在流的带宽信息用于队列调度,但是,它存在2个缺点:一个是该算法只适用于OQ交换结构,或者通过增大加速比才能够仿真OQ的Crossbar交换结构,因而实现难度较大;另一个是边缘路由器仍然需要维护每个流的状态信息,也就可能产生瓶颈。基于CSFQ算法的思想,提出了基于多跳间时延协作的调度算法以及实现方案。算法在分组头中记录分组的"剩余年龄"(即分组当前时刻到最大时延的时间差),并把它用于交换结构对分组的调度,相对于传统路由器对分组的独立调度,该方法能够保证分组按时到达目的地,同时,还能够从全网的角度对资源分配进行优化,满足尽可能多的分组的端到端时延保证。

1 问题分析

1.1 路由器的交换结构模型

图1给出了路由器交换结构模型,包括PQ,CVOQ,COQ,SEG和ASM组件。各组件功能如下:PQ_i为第i个输入端口上存储从链路上接收到的分组的队列;$CVOQ_{i,j}$是用于存储第i个输入端口到达第j个输出端口的信元的队列;$COQ_{i,j}$为第i个输出端口上存储来自第j个输入端口信元的队列;SEG是把PQ中的分组分割成信元并按照预定的算法把信元存入CVOQ;ASM是把COQ的信元重新组装成分组,并向输出链路发送。其中PQ和COQ是

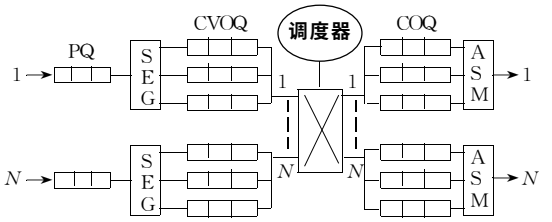

图1 路由器交换结构模型
Fig. 1 Switch model of router

* 收稿日期:2003-05-10 修订日期:2003-08-29
基金项目:教育部博士学科点专项科研基金(20020013011)、华为科技基金资助项目。
作者简介:周卫华(1976-),男,山东郓城人,博士研究生,主要研究方向为交换技术、调度算法、MPLS和GMPLS;丁 炜,教授,博士生导师,主要研究方向为高速交换和路由技术。

FIFO 队列,CVOQ 可以是 FIFO 或优先级队列。SEG 主要功能是把 PQ 队头的分组分割成信元,然后把信元按照FIFO 的顺序放在CVOQ 队列尾进行排队,或者根据所选择的权重按照不同优先级把信元插入到CVOQ 队列的合适位置。在使用iSLIP 的交换结构中,SEG 以 FIFO 的顺序对分割后的信元进行排队。当加速比为1 时,每个时隙最多有一个信元到达一个出端口,所以,每个时隙内 ASM 最多只需对一个分组进行重组。

1.2 调度算法对端到端时延和时延抖动的影响

基于轮循的 iSLIP 和 DRR 努力的目标是尽可能使调度资源公平分配到各端口,并没有考虑等待分组的"紧急程度",因而,并不能提供QoS 保证。基于加权匹配的 OCF 和 LQF 等算法使用局部的状态信息(队列头信元在系统内的等待时间或者队列的长度)作为权重,只能提供局部的 QoS 保证,而不能保证端到端的QoS,因此,多跳间无协作的调度算法带来如下问题:① 跳数不同的分组端到端时延抖动较大,跳数差异越大,时延抖动就越大,少跳数的分组会比最大时延要求提前到达,而多跳数的分组将超过最大时延要求到达而被丢弃,会造成网络资源的浪费;② 同一流的分组端到端时延抖动较大,虽然,同一流的分组经过的是相同的路径,但是由于路径上背景流量变化的影响,同一流的分组端到端的时延抖动也较大。因此,多跳间无协作的调度算法将会对网络资源利用率、分组时延、时延抖动以及分组丢弃率造成严重影响,如 VoIP 等对 QoS 性能要求较高的业务的服务质量就难以在维持较高的资源利用率的情况下得到保证。

2 算法描述和分析

基于 CSFQ 算法的思想,以如何利用有限的网络资源使尽可能多的分组满足端到端的 QoS 保证为研究目标,提出了适用于 Crossbar 交换结构的紧急分组优先算法(MUPF: most urgent packet first)。该算法使用分组的"剩余年龄"为权重对分组进行调度。随着分组剩余年龄的减小,分组优先级逐渐增加,就越优先得到调度。分组在系统内的排队时延就逐渐减小,端到端时延就得到保证,而且,不同分组的端到端时延也得到平衡,网络资源利用率被提高,更多的分组能够满足端到端时延要求。

2.1 MUPF 算法

MUPF 算法的核心思想是为分组提供端到端的QoS 保证,却不需要维护每个流的状态信息。该算法有2 个重要目的。第一,为了避免维护每个流的状态信息,每个分组都记录分组的"剩余年龄"(PktLeftAge),即分组距离最大时延要求的时间差。分组产生时,PktLeftAge 初始化为该业务对分组的时延要求, PktLeftAge<0 时,分组被丢弃。PktLeftAge 有记录在分组头中、记录在选项域中或者在第三层和第二层之间增加像MPLS 一样的新的一层等3 种方式记录[4]。第二,为了避免对每个流进行复杂的队列管理和调度,路由器的输入端使用优先级 CVOQ 队列。信元权重为信元的"剩余年龄"(CellLeftAge),CellLeftAge 越小,优先级越高。CellLeftAge = PktLeftAge − CellDelay,其中,PktLeftAge 为信元所在分组的"剩余年龄",CellDelay 为信元当前在系统内的等待时延。

MUPF 算法主要包括优先级排队、最大权重调度算法以及信元重组等功能,分别由路由器的 3 个主要部件(SEG、交换结构和调度算法以及 ASM)来完成。其功能如下。① SEG 把PktLeftAge>0 的分组分割成信元,并在信元头中记录分组进入系统的时间(用于计算CellDelay)以及分组的PktLeftAge。同时以信元的PktLeftAge 为权重使用链式Hash 法把信元插入CVOQ 中排队。② 调度算法使用iOCF (iterative modified oldest cell first)算法对各CVOQ 的信元进行调度。信元权重$w_{i,j}$为CellLeftAge。最大权重匹配算法有可能导致输入端被"饿死"的现象[4]。然而 iOCF 在任何情况下都不可能出现"饿死"现象,这是因为对于暂时没有得到服务信元,其权重会不断增加,直到最终得到服务为止。③ 接收到分组的最后一个信元后,ASM 对信元进行重组,并根据分组进入系统的时间,计算分组在系统内的时延(PktDelay)以及分组在后一段链路上的传输时延(PktTranDelay)、分组长度/链路速率。更新分组的"剩余年龄", PktLeftAge = PktLeftAge − PktDelay − PktTranDelay,如果 PktLeftAge <0 时,则丢弃分组,否则发送分组到链路上。

2.2 算法性能分析

考虑输出端口为 j 的 N 个队列 $\mathrm{CVOQ}_{i,j}(i=0,1,2,\cdots,N-1)$,而在不考虑输入竞争的情况下(即到达不同输出端口的最"紧急"分组位于同一输入端口),在每个时隙内,MUPF 调度算法将从该 N 个队列的头信元中选择最"紧急"的信元进行调度。此时,MUPF 具有以下性质:① 能够使分组最大时延最小

化;② 能够使分组最小时延最大化;③ 能够使分组时延抖动最小化。下面分别证明这 3 个性质。

证明 在不考虑输入竞争的情况下,调度算法将在输出端口为 j 的 N 个队列 $\mathrm{CVOQ}_{i,j}(i=0,1,2,\cdots,N-1)$ 中选择最"紧急"的信元进行调度,因此,该 N 个队列实际上就是按照"紧急"优先级排队的一个队列。考虑在 T_0 时间队列中 2 个分组(叫做 A 和 B)的"剩余年龄"分别为 l_A 和 l_B,且 $l_A<l_B$。假设存在一个和 MUPF 调度算法不同调度结果 S,其最大时延要比 MUPF 还小。根据定义,分组 B 将最先得到服务。如果每个信元的服务时间相同(为 T),最大允许时延为 $d_{\max}$,则分组 B 的时延为:

$$d_B = d_{\max} + T - l_B \tag{1}$$

由于系统为非抢占队列,则分组 A 将在 T_0+nT 时得到服务,其中 n 为正整数,则分组 A 的时延为:

$$d_A = d_{\max} + (n+1)T - l_A \tag{2}$$

现在考虑另一个和 S 很相似的调度结果 S',只是分组 A 首先在时间 T_0 得到服务,而分组 B 将在 T_0+nT 得到服务。由于服务时间相等,其他等待的分组将不会受到影响。在 S'' 调度时可得到以下结果:

$$d'_A=d_{\max}+T-l_A, d'_B=d_{\max}+(n+1)T-l_B \tag{3}$$

由式(1),(2),(3)可得:

$$d'_A < d_A \text{ 和 } d'_B < d_A \tag{4}$$

由式(4)可得

$$\max(d'_A,d'_B) < \max(d_A,d_B)$$

由于其他的分组并没有受到影响,因此 S'' 调度时的最大时延要小于或等于 S 调度时的最大时延。这个过程一直持续到没有任何分组违背 MUPF 调度原则。因此 MUPF 调度时的最大时延要小于或等于其他的调度 S 的最大时延。性质①得到证明。

和性质①证明一样,由式(1),(2),(3)可得:

$$d'_A > d_B \text{ 和 } d'_B > d_B \tag{5}$$

由式(5)可得:

$$\min(d'_A,d'_B) > \min(d_A,d_B) \tag{6}$$

性质②得证。

以下证明性质③。在 S 的调度下分组的时延抖动方差为:

$$v^2(d) = \frac{1}{k-1}\sum_{i=1}^{k}(d_i - \frac{1}{k}\sum_{j=1}^{k}d_j) = \frac{1}{k-1}\sum_{i=1}^{k}(d_i)^2 - \frac{1}{k(k-1)}(\sum_{j=1}^{k}d_j)^2 \tag{7}$$

由式(1),(2),(3)得:

$$d'_A + d'_B = d_A + d_B \tag{8}$$

继而可得:

$$\sum_{i=1}^{k}d_i = \sum_{i=1}^{k}d'_i \tag{9}$$

因为其它分组都没有受到影响,因此,由式(4),(5),(6)得:

$$(d_A)^2 + (d_B)^2 > (d'_A)^2 + (d'_B)^2 \tag{10}$$

另外,式(7)的第一项在 S 中要大于 S',因此:

$$v^2(d) > v^2(d') \tag{11}$$

由此,性质③得证。

以上性质是在假设没有输入竞争的情况下得到,而 iOCF 算法并不能避免输入竞争的发生,因而,实际中 MUPF 算法并不能严格符合以上性质,但是,在分组在独立同分布的均匀到达情况下发生输入竞争的概率很小,即使发生输入竞争,iOCF 也将选择接近"最紧急"的分组进行调度。因此,MUPF 算法仍然对于平衡分组端到端时延、降低分组时延抖动有很好的作用,仿真结果也证明了这一点。

3 计算机仿真和性能分析

图 2 是仿真使用的网络拓扑结构模型。路由器采用 16×16 的交换规模。路由器之间使用 8 条链路

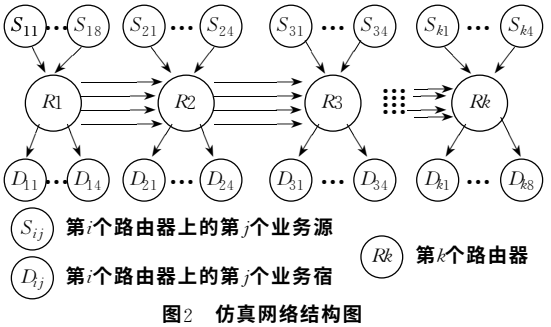

图2 仿真网络结构图

Fig. 2 Network architecture in simulation

相连接,其余端口则分别连接业务源和业务宿。业务源产生泊松到达的分组,分组目的出端口为(0,15)均匀分布。分组长度分布采用 IEEE 工作组公布的混合分组长度统计结果,见表 1。为了保证收到分组的目的出端口都是(0,15)均匀分布,每个路由器的输入端口对所收到分组的目的出端口重新进行(0,15)间均匀分配。仿真的目的是检验 MUPF 算法对具有不同跳数分组的端到端时延的调节作用,因此对目的出端口重新分配并不会对结果有影响。

由于能够达到目的地的分组肯定都能够满足其对时延的要求,因此,仿真重点是研究调度算法对时延抖动以及分组丢弃率的影响。为了方便分析和比较,文章对 MUPF、iSLIP 以及 iOCF 在相同的条件

下分别做了仿真。仿真网络由20个路由器组成,链路速率为10 Gbit/s。CVOQ、PQ和COQ都采用无限长队列。仿真时间段选择1 s。仿真结果图采用对数坐标,时延抖动以信元时隙(slot)为单位(slot=64×8/10 Gs)。

表1 分组长度概率分布表

Tab. 1 Probability distribution of packet length

概率	分组长度(bytes)	概率	分组长度(bytes)
0.197	100	0.687	900
0.406	200	0.719	1000
0.464	300	0.915	1100
0.498	400	0.916	1200
0.519	500	0.917	1300
0.628	600	0.919	1400
0.647	700	1.000	1500
0.667	800		

3.1 分组丢弃率分析

图3示出的是1 s内通过交换网络的所有分组分别在时延门限200和700(时隙)时的平均分组丢弃率。从图3中可以看出各调度算法随时延门限的放松分组丢弃率也都有降低。随负载的增加分组丢弃率也都有增加,而且低时延门限时分组丢弃率随负载的增加而增长更快。但是,在各种情况下,MUPF都比其它的2个算法的丢弃率要低。原因是MUPF算法能够根据分组的"紧急"程度有效地降低多跳数分组的时延,增加少跳数分组的时延,从而使更多的分组满足端到端时延的要求。另一方面,为了仿真的方便,选择各路由器之间使用多条并行10 Gbit/s的高速链路相互连接,这样就在一定程度上限制了MUPF算法对分组时延平衡的作用,在由不同吞吐量的路由器以及不同链路速率组成的网络中,MUPF会更有明显的优势。

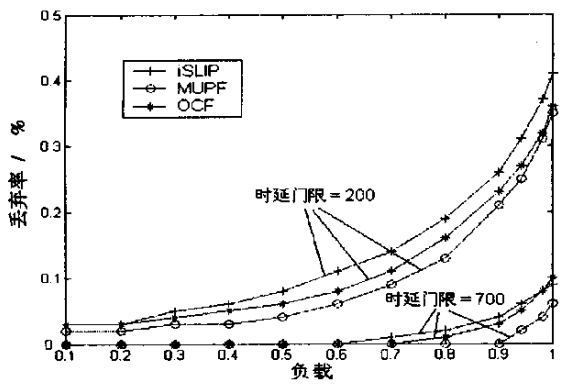

图3 不同时延门限时分组丢弃率随负载的变化

Fig. 3 Packet discarded ratio vs threshold

3.2 时延抖动分析

时延抖动是QoS另一个重要参数,反映了网络资源的利用率。时延抖动越小说明各分组时延差别较小,各分组占用的网络资源也就比较平均;反之,网络资源也就分配不公。图4和图5示出的是1 s内通过交换网络的所有分组端到端时延的抖动,可以看出在各种情况下MUPF要比iSLIP和iOCF有更低的时延抖动,也就是有更好的资源利用率。

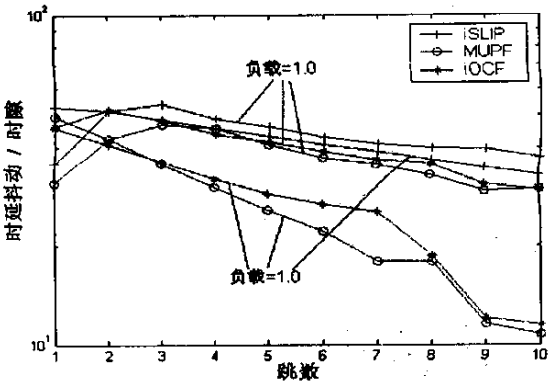

图4 不同跳数时端到端时延抖动(时延门限=200)

Fig. 4 Packet jitter vs hops(when threshold=200)

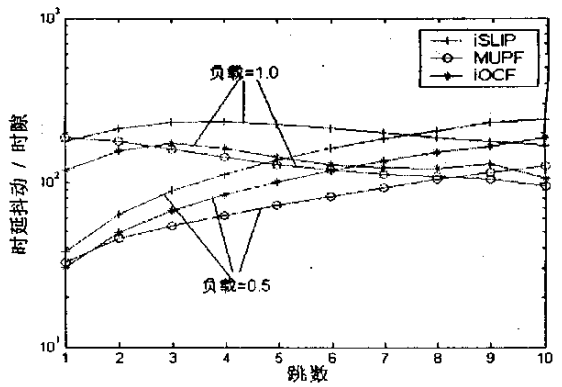

图5 不同跳数时端到端时延抖动(时延门限=1000)

Fig. 5 Packet jitter vs hops(when threshold=1000)

4 结 论

本文提出的基于多跳间协作的紧急分组优先算法以分组中记录的"剩余年龄"为权重,对分组进行调度。算法通过控制分组在每一跳上的时延来达到对端到端时延的调节,从而使网络资源尽可能公平分配,使分组尽可能满足时延要求。相对于iSLIP和iOCF,算法具有较高的资源利用率、较低的分组端到端时延抖动以及较低分组丢弃率等优点,但是,当不同QoS要求的分组具有相同的"剩余年龄"时,算法将认为两者具有相同的优先级进行公平调度,因此该算法目前只适用于网络中承载一种业务的情

况。如何有效支持多种业务将是进一步研究的方向。

参考文献:

[1] MCKEOWN N. Scheduling algorithms for input-queued cell switches [D]. Ph. D. dissertation, Univ. California, Berkeley. CA. May 1995.

[2] ANDERSON T. High speed switch scheduling for local area network[J]. ACM Trans on Computer Systems. 1993,(5):319-352.

[3] LAMAIRE R. O, SERPANOS D N. Two-dimensional round-robin schedulers for packet switches with multiple input queues [J]. IEEE/ACM Transactions on Networking, 1994,2(5):471-482.

[4] CHEN T, WALRAND J, MESSERSCHMITT D. Dynamic priority protocols for packet voice[J]. IEEE Journal on Selected Areas in Communications,1989, 7(5):632-643.

[5] CLARK D, SHENKER S, ZHANG L. Supporting real-time applications in an integrated services packet network: Architecture and mechanism [C]. In Proceedings of ACM SIGCOMM'92, 1992, Baltimore, Maryland,14-26.

[6] ANDREWS M. Probabilistic end-to-end delay bounds for earliest deadline first scheduling [C]. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, 2000.

[7] ANDREWS M,ZHANG L. Minimizing end-to-end delay in high-speed networks with a simple coordinated schedule [C]. In Proceedings of IEEE INFOCOM '99, New York, NY, 1999.

[8] LI Chengzhi, EDWARD W Knightly. Coordinated multihop scheduling: A framework for end-to-end services [J]. IEEE/ACM Transactions on Networking, 2002,10(6):776-789.

[9] STOICA I,SHENKER S,ZHANG H. Core-stateless fair queuing: Achieving approximately fair bandwidth allocations in high speed networks [R], June 1998. Technical Report CMU-CS-98-136, Carnegie Mellon University.

(**编辑:龙能芬**)

Most urgent packet algorithm in precedence based on end-to-end delay guarantees

ZHOU Wei-hua, NI Xian-le, DING Wei

(*Broadband Communication Networks Laboratory, Beijing University of Posts and Telecommunications, Beijing* 100876,*P. R. China*)

Abstract:In this paper, a delay coordination based multihop crossbar scheduling algorithm is presented, which is the most urgent packet. This algorithm can provide end-to-end delay guarantee of packets. This algorithm uses the left delay recorded in the head of packet as the weight for packets scheduling. Through controlling the delay of packet in different hops, this algorithm can provide end-to-end delay guarantee of packets, and adjust the end-to-end delay of packets with different hops. With this algorithm, the router can avoid maintaining per flow state and avoid complicated per flow buffering and scheduling. So the scalability of the router is improved. Simulation shows that the algorithm has many advantages in the better resource utilization, low jitter of the end-to-end delay and low discarded ratio of the packet.

Key words:most urgent packet; crossbar; jitter of packet delay; discarded ratio of packet