一种提供延迟保证的多级 FIFS 队列包调度算法*

杨明川[1,2+], 钱华林[1]

[1](中国科学院 计算机网络信息中心,北京 100080)

[2](中国科学院 计算技术研究所,北京 100080)

A Multi-Level FIFS Queue Packet Scheduling Algorithm to Provide Delay Guarantee

YANG Ming-Chuan[1,2+], QIAN Hua-Lin[1]

[1](Computer Network Information Center, The Chinese Academy of Sciences, Beijing 100080, China)

[2](The Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: 86-10-66001199 ext 2740, E-mail: yangmch@ctbri.com.cn

Received 2001-10-11; Accepted 2002-04-10

Yang MC, Qian HL. A multi-level FIFS queue packet scheduling algorithm to provide delay guarantee. *Journal of Software*, 2003,14(3):531~537.

Abstract: Packet scheduling algorithm is an important element to provide quality of service (QoS) guarantee. Traditional per-flow packet scheduling methods are often not able to support good scalability. While, the non-per-flow-differentiated methods usually can not provide service guarantee for very individual flow. DPS (dynamic packet state) provides a method to support guaranteed service without per-flow control. It can provide both the service performance and scalability. But its per-packet scheduling results in high computational complex are related with the number of packets. A packet scheduling algorithm with multi-level FIFS queues, which is based on DPS, is provided in this paper to get guaranteed service. And the constrained conditions to delay guarantee under this algorithm is also provided. The theoretical analysis and simulations show that the algorithm can schedule packets with constant time complexity and the same delay performance as DPS.

Key words: packet scheduling; QoS (quality of service); guaranteed service

摘 要: 包调度算法是提供服务质量保证的一个重要部分.传统的每流区分的包调度方法通常不能支持较好的扩展性,不适应当前网络带宽的迅速增长.而非每流区分的方法又不能提供每流的服务保证.动态包状态(dynamic packet state,简称 DPS)方法提供了一种在无须维护每流状态下提供保证服务的方法,该方法在保证服务质量的同时大大提高了扩展性.但是它仍然需要每包的调度,其复杂度和包的数量有关.在 DPS 的基础上提出了一种用多级 FIFS 队列提供延迟保证的包调度算法,并给出了该算法实现服务保证的约束条件.理论分析和仿真实验结果都表明:该算法可以实现常数时间的包调度复杂性,同时具有和 DPS 同样的延迟性能.

关键词: 包调度;服务质量;保证服务

* Supported by the National High-Tech Research and Development Plan of China under Grant Nos.2001AA112040, 2001AA112136 (国家高技术研究发展计划)

第一作者简介: 杨明川(1973－),男,重庆人,博士,工程师,主要研究领域为计算机网络体系结构,网络服务质量.

中图法分类号: TP393 文献标识码: A

现有的 IP 网络提供无服务质量(QoS)保证的尽力服务(best-effort)方式,这种服务方式为网络的设计提供了很大的灵活性和扩展性.但是,随着 Internet 日益成为未来全球统一的通信平台,需要支持大量的实时多媒体应用,如话音、视频、在线游戏等,对 QoS 的要求越来越迫切,原有的尽力服务方式不再能提供满意的支持.

实现每流的 QoS 保证的一个重要环节是包调度(scheduling).包调度算法在输出队列上决定当前哪个包进入链路发送.传统的包调度算法需要对每个流进行区分,并维护每流的状态,如加权公平队列 WFQ[1].每流包调度算法的主要问题是缺乏扩展性.它在两个方面消耗资源:一方面是需要对每个流的状态信息进行存储和处理,另一方面是根据状态信息对包调度进行决策[2].二者的复杂度都与当前流的个数有关.随着网络带宽的快速增长,骨干网络中流的数量大大增加(可达数十万甚至上百万个),同时每个包获得的处理时间相应减少.例如,在 OC-48 的链路上对每个包长为 32 字节的包,要求处理时间不超过 107ns.这两个方面都对包调度的扩展性提出了更高的要求,每流的调度显然很难满足这些要求.

为了解决扩展性问题,近年来提出了一些方法.文献[2]提出了一种通过缓冲区管理来实现速率保证的方法.该方法通过限制每个流的包在缓冲区中的占有率来控制其速率.其优点是简化了调度决策的复杂度,并能实现公平性.但是仍然需要维护每流的状态.另一种方法是不再区分每个流,而是对流按服务质量需求分类,采用汇集的方法调度.如在区分服务(diffserv)模型[3]中采用的方法.它的缺点是很难提供对每个流的服务保证.本质上,这些方法都是在扩展性和性能上作了一定的折衷.为了既保证扩展性,又能实现服务保证,文献[4]提出了一种基于动态包状态(dynamic packet state,简称 DPS)的方法.它通过在包中携带调度信息来实现无状态的调度.DPS 可以提供与 WFQ 相同的服务保证.包括基于速率的延迟保证和公平性.本文的工作基于 DPS 方法.

DPS 的一个主要缺点是,它需要在队列中对每包进行调度决策,这依赖于当前队列中所有包的合法时间(eligible time)和截止时间(deadline).所以它实际上没有降低调度决策的复杂度,而只是降低了每流管理的复杂度.如前所述,每包调度决策的复杂度在高速网络中支持线速的转发非常重要.为了进一步提高调度的扩展性,本文在 DPS 的基础上提出了一种用多级先来先服务(MFIFS)队列实现无状态包调度的算法.这种方法利用了基于帧(frame)的非持续工作(non-work-conserving)方法的思想,基本的方法是按时隙(slot)建立多个级连的 FIFS 队列,到来的包根据其合法时间和截至时间被送到不同队列中进行调度.MFIFS 的方法可以实现常数的时间调度复杂度,与当前队列中包的个数和流的个数无关,因此具有更好的扩展性,同时能够获得和 DPS 相同的性能,即可以获得和 WFQ 相当的延迟保证和公平性.

使用多个 FIFS 队列实现包调度的思想在其他包调度方法中也有应用.如文献[5]在文献[2]的基础上提出了用多个段(section)进行基于缓冲区控制的速率保证.其结果表明:分段可以极大地减小对缓冲区容量的需求,并有效地减小延迟.文献[6]提出了一种用多个循环 FIFO 优先队列(RPQ)模拟最早截止时间优先(EDF)调度的方法.

本文第 1 节介绍相关的工作.第 2 节讨论 MFIFS 方法的基本模型和性能分析.第 3 节讨论本文的方法在实现上的考虑.第 4 节给出了一些仿真实验的结果和分析.最后是结论和未来的工作.

1 相关工作

DPS 基于一个域模型 SCORE[4],它在网络域中区分边界节点和核心节点,边界节点初始化每个包中携带的用于动态计算每个包的合法时间和截止时间的必要信息.核心节点根据这些值进行每包的调度.同时根据调度的状态更新这些值.DPS 的核心是 CJVC(core-jitter-VC),它是一种消去了每流状态依赖的延迟抖动虚拟时钟调度算法(jitter-VC)[7~9].Jitter-VC 是一种非持续工作的调度算法,与传统的虚拟时钟相比,它同时定义了包的合法时间和截止时间.在包的合法时间没有到来之前,包被阻塞,即使此时系统空闲(保持非工作持续).另外,它考虑了多个传输节点的情况,保证包在沿路径的节点中转发时,保持一定的恒定延迟时间,目的是保证抖动不会因为传输距离的延长而增加,其计算方法如下:

记流 i 的第 k 个包为 p^k_i,它在传输路径上的第 j 个节点的合法时间 $e^k_{i,j}$ 和截止时间 $d^k_{i,j}$ 满足:

$$e^1_{i,j}=a^1_{i,j},\ e^k_{i,j}=\max(a^k_{i,j}+g^k_{i,j-1},d^{k-1}_{i,j}),\ i,j\geq 1,k>1, \tag{1}$$

$$d^k_{i,j}=e^k_{i,j}+\frac{l^k_i}{r_i},\ i,j,k\geq 1. \tag{2}$$

其中 $a^k_{i,j}$ 为 p^k_i 在节点 j 的抵达时间;$g^k_{i,j}$ 为 p^k_i 在截止时间前提前发送的时间量,有 $g^k_{i,j}=d^k_{i,j}-s^k_{i,j}$;$s^k_{i,j}$ 为 p^k_i 在节点 j 的开始发送时间;l^k_i 为 p^k_i 的长度;r_i 为流 i 的服务率.

文献[10]的分析表明,Jitter-VC 支持与虚拟时钟相同的延迟保证.它通过保持在同一个流的不同包之间、同一个包在不同节点之间的延迟,既保证了延迟抖动性能,又避免了由于过快发送而导致的突发,并使得每个流的流量在每个节点是可以预测的.这为包在边界节点提前计算截止时间提供了依据.但是,观察式(1)可以发现,第 k 个包的合法时间的计算依赖于第 k–1 个包的截止时间.为了实现无状态的包调度,必须消除这种包之间的依赖性,这是 CJVC 的主要思想.为了消除对流状态的依赖性,CJVC 引入了一个变量 δ^k_i,这个变量表示包 p^k_i 的合法时间与抵达时间的累计差.如下式:

$$e^k_{i,j}=a^k_{i,j}+g^k_{i,j-1}+\delta^k_i, \tag{3}$$

$$a^k_{i,j}+g^k_{i,j-1}+\delta^k_i\geq d^{k-1}_{i,j}. \tag{4}$$

如果在沿路径的每个节点 j(共有 h 个节点)都满足式(4),就使得式(1)的计算不依赖于上一个包的截止时间.保持在最后一个节点等式(4)的成立,就可以递推出各个包的 δ^k_i 值.令 $\delta^1_i=0$,有

$$\delta^k_i=\max(0,\delta^{k-1}_i+\frac{l^{k-1}_i-l^k_i}{r_i}-\frac{e^k_{i,1}-e^{k-1}_{i,1}-l^{k-1}_i/r_i}{h-1}),k>1,h>1. \tag{5}$$

这样,由式(3)可得,包的合法时间计算仅取决于 $a^k_{i,j}$, δ^k_i 和 $g^k_{i,j-1}$.而这些值都可以从包本身的信息获得,这样就避免了维护每流的状态.根据式(3)和式(5)可以对包进行无状态的调度.具体的方法是:① 在入口节点计算好 δ^k_i 的值,在发送到下一节点时由 $g^k_{i,j}=d^k_{i,j}-s^k_{i,j}$ 计算 $g^k_{i,0}$ 的值;② 在每个中间节点,由式(3)计算包的合法时间,由式(2)计算截止时间,并在转发时更新 $g^k_{i,j-1}$ 的值.对于已经计算好合法时间和截止时间的包,采用 Jitter-VC 的调度方法.由于这时每个包是独立调度的,不需要再识别流,所以调度的复杂度和当前队列中包的个数有关.一般地,如果仅对截止时间进行排序,每个包到来时处理的复杂度为 $\log(n)$,n 为包的个数.如果再考虑包的合法时间,复杂度会更高.

2 基于 MFIFS 的方法

本文的方法采用了 CJVC 的调度机制,为简单起见,我们采用了和 CJVC 同样的符号表示,见第 1 节.

2.1 MFIFS调度的基本方法和核心问题

MFIFS的基本方法是把可能的最大延迟时间划分成时隙(slot),每个时隙对应一个 FIFS 队列,构成多级 FIFS 队列,即 MFIFS 队列.当包抵达时,系统根据 CJVC 的机制计算出包的合法时间和截止时间,然后将其送入对应时隙的队列中.当每个时隙队列在该时隙成为当前时隙时,可以发送包,包按 FIFS 的方式发送.

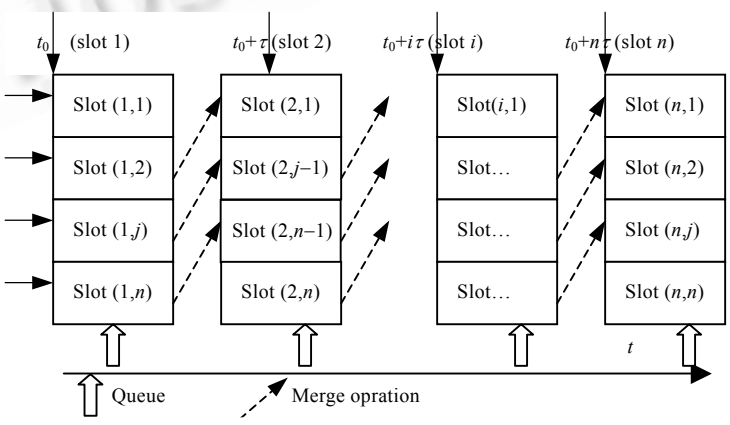

Fig.1 Multi-FIFS queue

图 1 多级 FIFS 队列

由于 MFIFS 调度方法使得进入同一个时隙的包是无序的,为了满足截止时间的要求,我们使用增加相对发送速率的方法,方法是通过许可控

制提高链路带宽与所有流的总服务率之比.由于进入每个时隙队列的流量受 CJVC 的非持续工作调度的限制,因此增加相对发送速率可以缩短包的发送时间,从而抵消因队列中包的无序排列带来的影响.但是,增加相对发送速率显然会限制支持的流的数量.后面将分析这个限制的应用背景.进一步的分析表明,仅仅增加发送的速率还不足以保证所有包的截止时间.考虑某个包具有相对较小的包长和较大的服务率,根据式(2),它的合法时间与截止时间的差是比较小的,如果这个包刚好又排在了队列的末尾,则即使在很高的链路带宽和总服务率比的情况下,其截止时间也很难保证.解决的方法是把包放到开始时间比它的合法时间小的时隙,这样将导致可能使包在其合法时间到来之前发送.但是考虑到,一方面提前发送的时间被限制在一个时隙范围内,另一方面,包提前发送的时间被记入参数 g(式(3)),在下一个节点,提前量会得到纠正.这两个因素都限制了由于提前发送产生的累计效应,因此,一定的提前发送是可行的.MFIFS 调度的另一个问题是,如果包的包长和包所在的流的服务速率之比(l_p/r_i)变化比较大,则会导致延迟性能的降低.我们采用了进一步给每个 FIFS 时隙队列分级的方法,即把具有不同的 l_p/r_i 的包分开.如图 1 所示,t_0 为任意一个开始时间,τ为时隙的长度.Slot(i,j)表示时隙队列 i 的第 j 级队列.实际上,这些队列构成了一个矩阵.整个矩阵可以用一个缓冲区来实现.缓冲区由不同时隙的队列级连而成,并用不同的指针区分内部级数.

2.2 MFIFS调度算法

当包进入队列时,首先判断包的合法时间,决定包插入哪个时隙的队列(纵向队列),然后判断包的截止时间,决定把包放入队列的哪一级,准确地说:

定义时隙 m 为$[t_0,t_0+m\tau]$,若它为当前时隙,则满足:

$$t_0+(m-1)\tau\le t<t_0+m\tau. \tag{6}$$

这里,t 为当前时间,τ为时隙长度.对任意流 i 的第 k 个包 p^k_i,插入时隙 Slot (m,n)标志的队列,其中:

$$m=\left\lceil\frac{e^k_{i,j}-t_0}{\tau}\right\rceil,n=\left\lceil\frac{d^k_{i,j}-t_0}{\tau}\right\rceil-m+1. \tag{7}$$

调度是按队列时隙的顺序进行的非持续工作调度.当队列的时隙成为对当前时隙时,该队列依次按队列的级数进行 FIFS 调度.如果在时隙结束时,还有某个(些)级的队列没有发送完包(但必须保证第 1 级队列发送完),则按截止时间进行队列合并.由式(7)可以看出,队列矩阵中沿对角线方向的队列有同样的截止时间范围,因此,可以把余下的各级队列合并到下一个时隙队列的较高级队列(如图 1 所示,只需要按对角线合并即可).合并时把前一个时隙的队列放到前面,因为它们有较小的合法时间.

对包携带的信息更新如式(2)~式(5),即 CJVC 描述的算法.

2.3 约束条件和性能分析

为了保证每个包的延迟性能,必须保证每个包的发送时间小于包的截止时间.实际上,由于在时隙结束时没有发送完的队列可以合并到下一个时隙的队列中去,所以只要保证当前时隙的第 1 级队列能在该时隙结束时发送完成即可.

定义 1. k 为链路总带宽 C 和所有流的服务率 r_i 之和的比,即 $k=\dfrac{C}{\sum_i r_i},k>1$.

定义 2. 忙周期 E$[t_1,t_2]$表示对于周期内任意 t,$t_1\le t<t_2$,系统忙(有合法的包可以发送).t_1 是该忙周期的开始时间,t_2 是该忙周期的结束时间.包 p 所在的忙周期表示 s_p 所在的忙周期,这里 s_p 表示包 p 的发送时间.

定理 1. 在 MFIFS 调度下,保证所有包的截止时间的充分条件是 $\dfrac{\tau}{k}\le\left(1-\dfrac{1}{k}\right)\min\left(\dfrac{\min(l_i)}{r_i}\right)-\dfrac{l_{p'}}{C}$.

这里,$\min(l_i)$为流 i 的最小包长,r_i 为流 i 的服务率,τ为时隙长度,C 为链路带宽,$l_{p'}$为一个特定的包的长度(它可以是任意流的任意包).详细的证明见附录.

2.4 约束条件的适用背景

根据对 CJVC 的分析,对任意一个包,下游节点的合法时间与上游节点的合法时间之差不超过 $\max(l_i)/r_i$,由于包可能提前τ发送,所以在本文的方法中这个值是 $\max(l_i)/r_i+\tau$.考虑到 $d_p-e_p\le\max(l_i)/r_i$,所以对任意的包,MFIFS

队列的时隙数不超过 m,对任意时隙,队列的级数不超过 n.

$$m=\frac{\max\left(\frac{\max(l_i)}{r_i}\right)}{\tau}+1,\quad n=\frac{\max\left(\frac{\max(l_i)}{r_i}\right)}{\tau},\tag{8}$$

即队列的级数不超过 $m-1$;记: $\max=\max\left(\frac{\max(l_i)}{r_i}\right),\min=\min\left(\frac{\min(l_i)}{r_i}\right)$. 有 $m=\max/\tau+1$,所以,$\tau=\max/(m-1)$.由定理 1,有 $\max/k(m-1)\le(1-1/k)\min-l_p/C$.如果,$C>>\max(r_i)$,可得到满足定理 1 的约束条件:

$$(k-1)(m-1)\ge\max/\min.\tag{9}$$

所以,通过适当地选取 k,m 的值,可以保证 MFIFS 调度的延迟性能.

3 实现考虑和复杂性分析

在实现上,由于我们的算法是建立在 DPS 的基础上的,因此我们采用和 DPS 同样的网络模型,即类似 SCORE 的模型.MFIFS 主要体现在内部节点的调度方法上.

首先考虑如何确定队列的级数和利用率.根据在上一节得到约束条件(9).我们首先分析 max 和 min 的取值范围.考虑到要求提供服务质量保证的应用主要是要求速率保证的多媒体应用,因此,可以限定其最大最小的速率.如我们以同时传送话音和视频为例,一路话音的速率最小在 5KB 左右,相应的包长也很小(通常小于 40 字节),而视频的速率通常不会超过 2MB,相应的包长也较大(通常大于 800 字节),综合起来,max/min 可以安全限定在 20 以下.其次,我们设定 m 和 k.如果我们令 k=2(50%的利用率),则当队列个数为 21 时即满足约束.对于一个提供多种类型服务的节点来说,保持一定的 k 值是合理的,因为多服务网络同时还支持一定非服务保证流量(如 best-effort).本质上,MFIFS 通过牺牲流量的延迟性能换取了的调度复杂性的降低.尽管如此,调度算法的非持续工作性决定了在任何特定的时期内发送的流量是有限的,所以不可能发生带宽被保证服务流长期占用的情况.因此,非保证服务的流量的延迟性能的损失被控制在一定范围之内.值得指出的是,式(9)的约束仅仅是基于最坏情况考虑,在通常情况下,约束要弱得多.

在实现上另一个需要考虑的问题是队列的管理.如前所述,队列可以由指针进行管理,并按照时隙进行队列轮转,即用循环队列实现,这样可以避免复杂的队列操作.

在实现考虑的基础上,我们从时间和空间两个方面对算法的复杂度进行分析.

(1) 时间复杂度分析.由算法可知,MFIFS 调度方法的时间开销主要由 3 部分构成,第 1 部分是包到达时入队的时间,这可以通过一个简单的计算完成(见(式)7),为常数复杂度;第 2 部分是更新包的状态信息,这在包出队时进行,消耗的时间也是常数复杂度;第 3 部分是当时隙结束时的队列合并,合并以周期 τ进行.每次最多需要合并 $m-1$ 个队列,每个队列的合并通过指针进行,合并的复杂度为 $O(m)$.队列的合并和包的调度(包括入队和出队)可以并行完成,除非周期 τ非常小.队列的合并对包的调度没有太大的影响.因此,MFIFS 对每包的调度只需要常数时间 $O(1)$的复杂度.

(2) 空间复杂度分析.系统占用的存储资源分为两部分,一部分是包队列缓冲区,一部分是存放队列维护的指针.与 DPS 方法相比,由于有 τ时间的提前发送,在最坏情况下,会增加 $\tau c/k=\max(m-1)c/k$ 的包队列长度.另外,需要维护 $m(m-1)$个队列的指针.通过仔细控制 m 的值,可以得到一个合适的空间消耗.

总之,MFIFS 的开销是常数性质的,与当前队列中包的个数和流的个数无关,具有很好的扩展性.

4 实验及分析

前面的分析表明:影响 MFIFS 调度延迟性能的因素主要有最大和最小包长与流的服务率之比、带宽利用率和队列级数等等.带宽利用率通过一定的许可控制来限定,队列级数在队列配置时确定,它们都是相对固定的值.在实验中,我们主要考察包长与流服务率的比值变化对性能的影响,并与 DPS 的结果相比较.

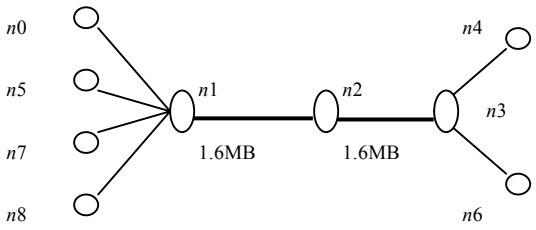

Fig.2 Simulation topology
图 2 仿真拓扑结构

采用 NS-2 对我们的算法进行仿真实验,在实验中采用图 2 的拓扑.为了简单起见,我们把源节点设为边界入口节点(实现包状态的初始化工作),把目标节点设为边界的出口节点.中间节点设为域的核心节点(实现 MFIFS 队列调度).核心节点之间的链路带宽为 1.6MB,边界节点和核心节点之间的链路带宽为 1MB.我们建立了 4 个流 $f1$, $f2$, $f3$, $f4$,流的服务率在 10KB~90KB 之间变化,流的包长范围则根据流的服务率在 500B~2000B 之间变化.具体的配置见表 1.

Table 1 Configure for flows

表 1 流配置

Flow	Traversing path	Rate (KB)	Maximum packet length (B)	Minimum packet length (B)	Ratio of packet length to service rate
$f1$	$n0$-$n1$-$n2$-$n3$-$n4$	50	2 000	1 500	0.03~0.04
$f2$	$n5$-$n1$-$n2$-$n3$-$n4$	50	1 000	500	0.01~0.02
$f3$	$n7$-$n1$-$n2$-$n3$-$n6$	90	2 000	1 000	0.011~0.022
$f4$	$n8$-$n1$-$n2$-$n3$-$n6$	10	500	100	0.01~0.05

其他设置包括:带宽利用率为 0.5,级数为 11,其中核心节点 $n1$,$n2$,$n3$ 是瓶颈节点.DPS 和 MFIFS 的每个流的各个包的延迟时间如图 3 和图 4 所示.从图中可以看出,DPS 和 MFIFS 队列的延迟性能非常相似.流 1、流 2 和流 3 在两种方式下延迟抖动性能都很好,因为它们具有相对狭窄的最小最大包长和服务率比的范围.流 4 的延迟抖动较大,因为它的包长和服务率比值变化较大.从图中可以看出,与 DPS 相比,MFIFS 的平均延迟稍低,因为它允许适当地提前发送.总之,实验表明,MFIFS 方式在极大地降低了复杂性的同时并没有失去 DPS 方法原有的优良特性,这与理论分析是一致的.

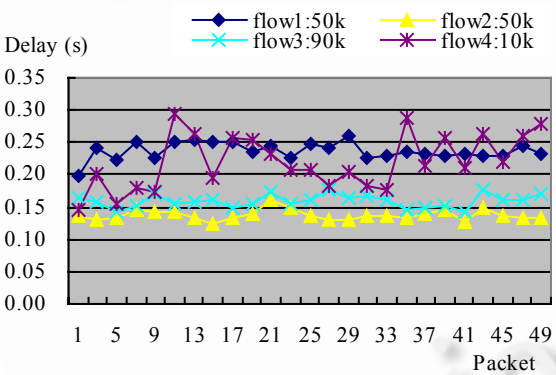

Fig.3 Delay for DPS
图 3 DPS 的延迟

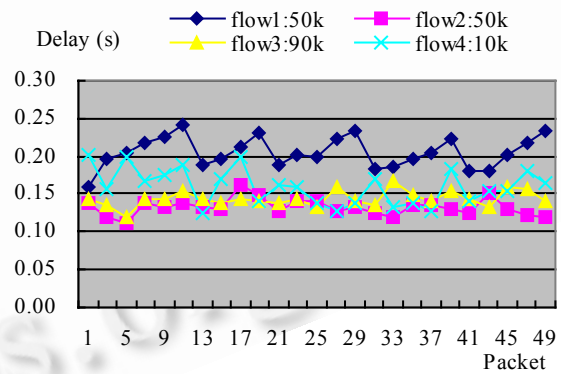

Fig.4 Delay for MFIFS
图 4 MFIFS 的延迟

5 结论以及将来的工作

本文在 DPS 方法的基础上提出了一种用 MFIFS 队列实现延迟保证的调度算法,这种算法实现了常数的时间复杂度和很好的扩展性,同时具有与 WFQ 相同的服务质量性能.本文的方法可以应用在 DPS 可以应用的全部环境下[4],例如在支持服务保证的高速网络中,也可以和其他的服务质量模型(如区分服务)集成.

MFIFS 方法的一个主要问题是需要一些约束条件(见定理 1),但是,考虑到本文的所有讨论都是基于最坏情况下的性能分析,所以如果我们允许一个小范围的截止时间越界的话,可以使这些约束大大减弱.本文没有讨论这个问题,我们将在后面的工作中对此作进一步的分析.

References:

[1] Parekh AKJ, Gallager RG. A generalized processor sharing approach to flow control in integrated service networks: the single node case. IEEE/ACM Transactions on Networking, 1993,1(3):344~357.

[2] Guérin R, Kamat S, Peris V, Rajan R. Scalable QoS provision though buffer management. In: Proceedings of the ACM SIGCOMM'98. New York: ACM Press, 1998. 29~40.
[3] Blake S, Black D, Carlson M, Davies E, Wang Z, Weiss, W. An architecture for differentiated services. RFC 2475, 1998.
[4] Stoica I, Zhang H. Providing guaranteed services without per flow management. In: Proceedings of the SIGCOMM'99. New York: ACM Press, 1999. 81~94.
[5] Cheung SY, Pencea CS. Pipelined sections: a new buffer management discipline for scalable QoS provision. In: Proceedings of the IEEE INFOCOMM 2001. Los Alamitos: IEEE Computer Society Press, 2000. 1530~1538.
[6] Liebeherr J, Wrege DE. A versatile packet multiplexer for quality-of-service networks. In: Proceedings of the 4th International Symposium on High Performance Distributed Computing (HPDC-4). Los Alamitos: IEEE Computer Society Press, 1995. 148~155.
[7] Zhang H. Service disciplines for guaranteed performance service in packet-switching networks. Proceedings of the IEEE, 1995, 83(10):1374~1396.
[8] Zhang L. Virtual clock: a new traffic control algorithm for packet switching networks. In: Proceeding of the ACM SIGCOMM'90. New York: ACM Press, 1990. 19~29.
[9] Xie GG, Lam SS. Delay guarantee of virtual clock server. IEEE/ACM Transactions on Networking, 1995,3(6):683~689.
[10] Georgiadis L, Guerin R, Perin V, Sivarajan K. Efficient network QoS provisioning based on per node traffic shaping. IEEE/ACM Transactions on Networking, 1996,4(4):482~501.

附录. 定理 1 的证明.

引理 1. 对于任意包 p,若满足:(1) $t_1 \le e_p$;(2) $e_p+l_p/r_i \ge 0$ $l_p/r_i \ge 0$,有 $e_p+l_p/r_i-(1-1/k)t_1 \ge (1-1/k)l_p/r_i$.

证明:令 $f=e_p+l_p/r_i-(1-1/k)t_1$,因为 $t_1 \le e_p$,且 $e_p \ge -l_p/r_i$,所以 $e_p \ge \max(-l_p/r_i, t_1)$,当 $t_1 \ge -l_p/r_i$,有 $e_p \ge t_1$,故 $f=e_p+l_p/r_i-(1-1/k)t_1 \ge e_p+l_p/r_i-(1-1/k)e_p \ge l_p/r_i+e_p/k \ge (1-1/k)l_p/r_i$.当 $t_1 \le -l_p/r_i$ 时,有 $e_p \ge -l_p/r_i$,故 $f=e_p+l_p/r_i-(1-1/k)t_1 \ge -(1-1/k)t_1 \ge (1-1/k)l_p/r_i$. □

定理 1. 在 MFIFS 调度下,保证所有包的截止时间的充分条件是 $\frac{\tau}{k} \le \left(1-\frac{1}{k}\right)\min\left(\frac{\min(l_i)}{r_i}\right)-\frac{l_{p'}}{c}$.

证明:令当前的时间为 t,当前的时隙为 m,时隙长度为 τ,由式(6)有 $t_0+(m-1)\tau \le t < t_0+m\tau$.我们讨论 slot($m$,1)中的包的延迟性能:对于 slot($m$,1)中的任意包 p^s,满足:

$$t_0+(m-1)\tau \le d_{ps} < t_0+m\tau. \tag{10}$$

令 t_1 为该 p^s 所在的忙周期 E 的开始时间.d_{ps} 为包 p^s 的截止时间.若 t_1 位于时隙 $n(n \le m)$,即 $t_0+(n-1)\tau < t_1 \le t_0+n\tau$,则在 t_1 到 s_p 期间,位于 p^s 所在的忙周期中且比 p^s 先发送的任意包 p 满足:

$$t_1 \le e_p, e_p \text{ 为包 } p \text{ 的合法时间}. \tag{11}$$

假设这些包同时满足:

$$d_p < t_0+m\tau. \tag{12}$$

实际上,假设 p'是不满足条件(12)的最后一个包(在 p^s 所在的忙周期中),令 p'所在的时隙为 $n'(n \le n' \le m-1)$,则有:在 p'之后发送的所有包 p 满足 $t_0+n'\tau \le e_p$.因此,我们可以把 $t_1'=t_0+n'\tau$作为新的周期的开始点($t_1'>t_1$),这个新的周期中除 p'之外,所有的包满足式(12)和式(13):

$$t_1' \le e_p. \tag{13}$$

令 $t_1=t_1'$,则有从 t_1 开始的忙周期的所有包满足式(11),最多有一个包不满足式(12),不妨令这个包为 p'.对于任意的流 i,令包序列 $p_1,p_2,\ldots,p_f$ 为在时间段(t_1,a_p)到达的该流满足条件(11)和(12)的包,有 $e_{p_1} \ge t_1, d_{p_f} \le t_0+m\tau$,不难证明:这些包的长度和小于等于 $r_i(t_0+m\tau-t_1)$,对所有流,所有这样的包(包含 p^s 在内)的包长和不大于

$$\Sigma r_i(t_0+m\tau-t_1)+l_{p'}=(t_0+m\tau-t_1)c/k+l_{p'}, \tag{14}$$

则 p^s 的发送完成时间不超过 $t_1+(t_0+m\tau-t_1)/k+l_{p'}/c$,因为 p^s 的截止时间为 $d_{ps}=e_{ps}+l_{ps}/r_i$,必须保证:

$$t_1+(t_0+m\tau-t_1)/k+l_{p'}/c \le e_{ps}+l_{ps}/r_i \tag{15}$$

不影响结果,为了分析方便,不妨令 $t_0=0$,$m=1$(t_0,m 都是相对值),则式(15)变成

$$t_1+(\tau-t_1)/k+l_{p'}/c \le e_{ps}+l_{ps}/r_i, \tag{16}$$

令 $\tau/k \le (1-1/k)l_{ps}/r_i-l_{p'}/c$,因为包 p^s 满足 $d_{ps}=e_{ps}+l_{ps}/r_i \ge 0$(式(10)),$t_1 \le e_{ps}$(式(11)),即满足引理 1 的条件,由引理 1 可得:$\tau/k \le (1-1/k)l_{ps}/r_i-l_{p'}/c \le e_{ps}+l_{ps}/r_i+(1-1/k)(-t_1)-l_{p'}/c$.所以,$t_1+(\tau-t_1)/k+l_{p'}/c \le e_{ps}+l_{ps}/r_i$.式(16)成立. □