基于帧的 ATM 层次调度机制性能界及其仿真研究

姜宁康, 李毓麟
(上海交通大学光纤区域通信网国家重点实验室, 上海 200030)

摘 要: 对网络交换节点中所采用的调度机制的研究是网络提供 QoS 保证的一个重要课题. 在本文中, 结合基于帧调度器的简单性和 GPS(generalized processor sharing) 算法的良好性能, 提出了一种新颖的基于帧的层次调度算法 HFFQ(Hierarchical Frame-based Fair Queueing). HFFQ 能同时支持实时业务和非实时业务, 另外采用了简单计数的方法来替代复杂的系统虚拟时间的计算过程, 大大减少了算法的实现复杂度. 最后, 用理论分析和仿真的方法, 对 HFFQ 的公平性、服务率等指标的性能进行了论证. 结果显示, 它和 PGPS(Packet-by-packet GPS) 相比, 在性能上有很大的提高.

关键词: ATM (Asynchronous transfer mode); 信元调度; 公平队列; FCFS
中图分类号: TN915.2 文献标识码: A 文章编号: 0372-2112 (2001) 06-0770-04

Study of Performance Bound and Simulation of Frame-Based Hierarchical Fair Queueing Scheduling in ATM Network

JIANG Ning-kang, LI Yu-lin
(National Key Lab. of Fiber Optic Local Communication Networks S. J. T. U, Shanghai 200030, China)

Abstract: One important issue in the provision of QoS guarantees is the study of the scheduling disciplines to be employed at network switches. In this paper, we strive to combine the simplicity of frame-based schedulers with the performance characteristics of generalized processor sharing (GPS) disciplines, and propose a new scheduling mechanism named Hierarchical Frame-based Fair Queueing (HFFQ). HFFQ can support both real time traffic and non-real time traffic at the same time. In addition, it tremendously decrease the implement complexity by using counters to substitute the computing of system virtual time of other mechanism. At last, we evaluate some performance metrics of the HFFQ, such as fairness, service rate etc, using both analysis and simulation, and the results show that HFFQ has a better performance than that of PGPS.

Key words: ATM; VC scheduling; fair queueing; FCFS

1 分组调度算法背景及其存在的问题

目前, 大多数分组网络在节点中使用单一的先到先服务 (FCFS) 队列来进行分组调度. 由于实现简单, 而且在业务比较简单的情况下性能尚可, 因此 FCFS 得到了广泛的应用, 但是它不能区别对待不同的流, 从应用的角度来看, 对性能有很坏的影响. 支持多业务的网络应当使用比 FCFS 更高级的调度机制和合适的分组丢弃机制[2].

关于分组调度(packet scheduling) 算法的研究已经开展了多年, 从 90 年代以来, 特别是在集成服务网络的相关研究成为热点之后, 产生了以研究调度算法对网络集成服务的不同服务模式, 提供性能保证为出发点的另一种研究思路[3].

GPS(generalized processor sharing) 是一个理想化的服务器, 称为“通用处理器共享”, 是分组调度算法的基础. 它假定, 服务器在任何瞬间, 能同时为所有积压队列(在时刻 t, 一个队列中有需要服务的分组, 则称该队列在时刻 t 是积压队列) 提供服务. 并且, 业务流是无限可分的(即数据可以细分成任意小的单元). 但在实际的分组网络系统中, 任一时刻最多只能有一个连接得到服务, 并且, 只有在一个分组传输完成之后才能传输其它分组. 所以, 虽然 GPS 系统具有许多优良的性能, 但并不是一个实用的调度服务器.

PFQ(packet fair queueing) 算法是在分组系统中对 GPS 系统进行模拟、逼近, 以便既能在分组网络中实现, 又能最大限度地保留 GPS 系统的优点. 各种 PFQ 算法使用了类似 GPS 的基于虚拟时间函数的优先队列机制, 它们的区别主要在三方面[4]: 虚拟时间函数、分组选择策略和算法复杂度.

在已经提出的 PFQ 算法中, 很少既能在高速网络(如 ATM) 中实现, 同时又能满足下面三个目标:

(1) 支持大量的连接, 且各个连接的带宽需求变化很大;

(2) 调度速度很快, 622Mbit/s 或更高;

(3) 保持 GPS 的一些主要特性, 如延迟上界、公平性及最坏情况公平性.

传统分组调度机制, 主要考虑的是分组交换网的情况, 因

收稿日期: 2000-02-23; 修回日期: 2000-12-07

此数据包的长度是影响排队公平性的重要因素. 而对 ATM 网络, 分组(信元)的长度是固定的, 大大简化了排队的复杂性. 针对 ATM 的特点, 我们设计了基于帧的层次公平队列调度算法 HFFQ(Hierarchical frame based fair queueing).

2 HFFQ 信元调度器的结构

图 1 是 HFFQ 算法的调度器示意图. 假定每个连接有一个独立的缓冲队列, 输出链路的总容量 C, 为 I 个"实时业务"连接和 $\sum_{j=1,\cdots,J}N_j$ 个"非实时业务"连接共享. 每个实时连接可以直接参加组调度器的输出调度. 非实时连接中具有相同速率要求的连接放在一组, 称为"调度组", 组调度器把"调度组"作为参与输出调度的一个队列.

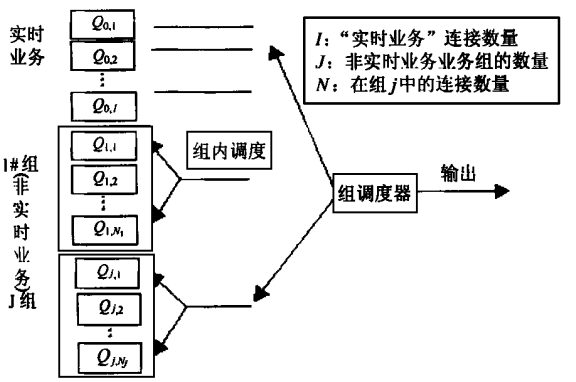

图 1　基于帧的层次公平队列调度器示意图

基于帧的调度器不需要维护分组的优先级队列, 它把时间轴分为定长或变长的一系列帧, 限定每一个连接在一个帧周期中允许传送的业务量, 以此来保证每个连接的带宽需求. HFFQ 也是一种基于帧的调度机制. 它不同于 WRR[5](Weighted round robin)和 DRR[6](Deficit round robin)的地方在于, 最大帧长是固定的, 不是共享链路的连接数量的函数, 因此, HFFQ 中的一个信元在最坏情况下的时延独立于连接的数量. 同时, HFFQ 是一种连续工作模式.

当"调度组"非积压时(组内所有连接队列都空), 其服务时隙用于实时连接中输出压力(队列长度/带宽)最大的连接. 当"调度组"部分空闲时, 空闲时隙为组内最长队列服务, 这样可以减少缓冲器溢出造成的信元丢失.

3 算法描述

信元的服务时间是一个时隙的长度, 一帧最多能传输 T_s 个信元. 调度过程分为两部分: 组调度过程和组内调度过程.

3.1 组调度过程

①*while* (不是全部队列都为空) {;

②$t = T_s$;//本轮最多可以传输的信元数;

③对实时业务: $tmp = r_{0,k} + rest_k$, 为每一个非空连接 k 服务 $Num = \min(Q_{0,k}\lfloor tmp + 0.5\rfloor, t)$ 个信元; 修改 $t = t - Num$;

$$rest_k = \begin{cases} 0.5, \text{ if } Num < \lfloor tmp + 0.5\rfloor \\ r_{0,k} + rest_k - \lfloor tmp + 0.5\rfloor, \text{ if } Num = \lfloor tmp + 0.5\rfloor \end{cases}$$

其中 $r_{j,k}$ 是组 j 中第 k 个连接的预留带宽, 若 $j = 0$, 表示实时业务, 则 $k = 1, \cdots, I$; 若 $j = 1, \cdots, J$, 则 $k = 1, \cdots, N_j$. I 是实时连接的数量; J 是交换机支持的非实业务速率种类, 也即调度组的数量; N_j 是组 j 中连接的数量, $j = 1, \cdots, J$; $Q_{j,k}$ 是连接的队列长度. $rest_k$ 是实时连接 k 带入下一帧的余额($k = 1, \cdots, I$);

④对非实时业务: $tmp = total_j + g\_rest_j$, 为每一个非空组 j($j = 1, \cdots, J$) 服务 $Num = \min(\lfloor tmp\rfloor, t)$ 个信元, 修改 $t = t - Num$; $g\_rest_j = tmp - \lfloor tmp\rfloor$. 其中 $total_j$ 是组 j 的带宽和($j = 1, \cdots, J$); g_rest_j 是组 j 带入下一帧的带宽余额($j = 1, \cdots, J$);

⑤在实时业务连接中分配剩余带宽(t 个时隙, $t > 0$)

While (not all queue empty and $t > 0$) {

从非空实时连接中找一个队列 k, 使得对所有其余实时连接 i ($k, i = 1, \cdots, I, i \neq k$), 有 $Q_{0,k}/r_{0,k} \geq Q_{0,i}/r_{0,i}$; 为队列 k 传输一个信元; $t = t - 1$; $rest_k = rest_k - 1$; }

⑥ };// 转到②继续执行.

3.2 组内调度过程

①组 j 内有 N_j 个连接, 在一帧内最多得到 $Num = \lfloor r_{j,k} + g\_rest_j\rfloor$ 个时隙;

②采用循环调度机制, 记录头队列和尾队列. 新连接加入到尾部, 从头队列开始服务, 每个时隙为一个队列传输一个信元, $Num = Num - 1$, 直到 $Num = 0$.

③若遇某队列为空, 则为最长队列额外服务一个信元. 在服务的过程中, 经过比较, 更新最长队列指针, 使其指向新的最长队列.

4 HFFQ 算法性能分析

4.1 公平性分析

即使两个调度算法能提供相同的时延保证, 它们的公平性也可能有很大的差异. 例如, 有些调度器会对一定时间内得到太多服务的连接采取一些惩罚性策略. 在许多文献中可以看到关于如何分析分组调度算法的公平性的方法. 文献[7]对公平性作了如下定义

定义　一个调度算法, 在时段 $(\tau, t]$ 内, 积压连接 i 和 j 得到的服务量分别为 $S_i(\tau, t)$ 和 $S_j(\tau, t)$, 速率分别为 c_i 和 c_j, 如果这两个连接的标称化服务量之间的差别是有限的, 即 $|S_i(\tau, t)/c_i - S_j(\tau, t)/c_j| \leq B$, 其中 B 是一个常量, 则称调度算法的不公平性是有界的.

GPS 的不公平性是 0, 而在基于分组的调度器中, 任一时刻调度器只能为一个分组服务, 并且在一个分组服务完毕之后才能为下一个分组服务, 所以短期不公平性总是存在的, 下面证明 HFFQ 的短期不公平性有上界.

引理　对一个 HFFQ 调度器, 一个连续积压连接 $C_{j,k}$ 在时段 $(t_0, t_n]$ 之间的服务率 $c_{j,k}$ 为 $c_{j,k} \geq \frac{R_{j,k}}{\sum_{j=0,k=1,\cdots,I} R_{j,k} + \sum_{j=1,\cdots,k=1,\cdots,N_j} R_{j,k}} T$, 其中 $R_{j,k}$ 是组 j 的第 k 个连接的预留带宽, $j = 0$ 时, $k = 1, \cdots, I$, I 是实时连接的数量; $j = 1, \cdots, J$ 时, $k = 1, \cdots, N_j$, J 是组的数量; N_j 是组 j 中连接的数量; T 是最大帧长; t_0 是一帧的开始; t_n 是从 t_0 之后的第 n 帧的结束.

证明　考虑一个完整的帧周期, 有

$$\sum_{j=0,k=1,\ldots,I} R_{j,k} + \sum_{j=1,\ldots,J,k=1,\ldots,N_j} R_{j,k} \geq \sum_{j=0,k\in B_I} R_{j,k} + \sum_{j\in B_J,k=1,\ldots,N_j} R_{j,k}$$

其中: B_I 和 B_J 是积压连接和调度组; 又因为 $C_{j,k}$ 在 $(t_0, t_n]$ 期间总是积压的, 并且调度器是连续工作的, 非积压连接的带宽被用来为积压连接服务, 所以有

$$c_{j,k} = \frac{R_{j,k}}{\sum_{j=0,k\in B_I} R_{j,k} + \sum_{j\in B_J,k=1,\ldots,N_j} R_{j,k}} T \geq \frac{R_{j,k}}{\sum_{j=0,k=1,\ldots,J} R_{j,k} + \sum_{j=1,\ldots,J,k=1,\ldots,N_j} R_{j,k}} T = \frac{R_{j,k}}{T_s} T$$

定理 对一个 HFFQ 调度器, 其公平性满足:

$$\left| \frac{S_i(\tau, t)}{c_i} - \frac{S_j(\tau, t)}{c_j} \right| \leq 2\times \frac{\text{分配给所有的连接的带宽之总和}}{T} \tag{1}$$

证明 首先, 考虑时间间隔为 $(t_0, t_n]$ 的情况.

$|S_i(t_0, t_n)/c_i - S_j(t_0, t_n)/c_j|$, 利用前述引理的结论, 将 c_i、c_j 的值代入式(1)左端, 则

$$\text{左端} \leq \left| \frac{S_i(t_0, t_n)}{R_i \times T/T_s} - \frac{S_j(t_0, t_n)}{R_j \times T/T_s} \right| = \left| \frac{S_i(t_0, t_n)}{R_i} - \frac{S_j(t_0, t_n)}{R_j} \right| \frac{T_s}{T} = P \times \frac{T_s}{T}$$

R_i、R_j 是连接 i 和 j 分配的额定带宽, 因此可以认为 P 是一个很小的常量.

第二, 考虑任意时段 $(\tau, t]$. 由于此时不一定在帧的边界, 有可能一个连接已得到本帧的服务份额, 而另一个连接还没有得到本帧的服务份额. 所以, 极端情况下, 式(2)成立.

$$\left| \frac{S_i(t_0, t_n)}{c_i} - \frac{S_j(t_0, t_n)}{c_j} \right| \leq 2P \times \frac{T_s}{T} \tag{2}$$

4.2 算法复杂性

在 HFFQ 机制中, 一个信元得到服务, 所需要的操作开销是 $O(1)$. 在组调度过程的离去过程中, 只需要简单的单步计算, 计数变量之间是独立的, 可以用一些简单的专用硬件来并行执行. 因此, 计数器的更新操作时间复杂度为 $O(1)$.

组内调度过程采用循环调度机制, 从前一个队列转向下一个队列只需移动对应的头队列计数器和尾队列计数器, 其操作复杂度是 $O(1)$, 维护最长队列指针的开销也是 $O(1)$.

在剩余带宽分配过程中, 需要在积压实时队列中找出输出压力(队列长度/带宽)最大的队列, 开销小于 $O(I)$. 所以, HFFQ 总的算法复杂度是 $O(I)$.

5 仿真研究

我们使用比较权威的排队系统分析仿真环境 C++ SIM 来进行仿真工作, 采用 ON/OFF 业务源生成器, 时隙长度等于一个 ATM 信元的传输时间. 在每一个业务突发期(ON), 以峰值信元速率 P 和平均突发度 B, 产生按几何分布的随机数量的信元. 在一个突发期之后, 是一个寂静期(OFF), 此时业务源不产生信元. 寂静期长度服从负指数分布, 平均长度为 I, 这样, 平均信元产生速率为: $A^{-1} = I/B + 1/P$, 在仿真过程中, 保持 B 不变, 通过改变 P 和 I 来调节负载率. 每一个业务源都使用 (σ, ρ) 漏桶流量描述模型, 在仿真中, 有 20 个连接共享带宽为 622Mbps 的链路. 固定 B (= 20cells), 为了比较性能, 研究了在同等业务条件下, PGPS 调度器、HFFQ 和文献[8]提出的 MFQ_FB 调度机制的性能表现.

5.1 信元丢失率固定下负载和缓冲器需求

图 2、图 3 示出了为得到一定信元丢失率的情况下, HFFQ 和 PGPS 两种调度器在不同负载下对缓冲容量的要求. 无论信元丢失率是 10^{-3}, 还是 10^{-4}, 在相同负载下 HFFQ 调度器所需的缓冲器数量总是少于 PGPS 所需要的缓冲量. 为了更清楚地显示 HFFQ 比 PGPS 性能的提高, 图 4 示出了相对改进——折线是仿真结果, 平滑线是经拟合后的结果. 相对改进定义为 $\frac{\text{PGPS 需要的缓冲量} - \text{HFFQ 需要的缓冲量}}{\text{PGPS 需要的缓冲量}}$. 当负载较轻时, HFFQ 比 PGPS 的相对改进比较小. 但是, 随着负载的增加, 改进也越来越大. 原因是当负载较低时缓冲器占用率不大, 溢出的概率相对较小, 此时不能显示 HFFQ 的优越, 当负载增大时, 缓冲占用率增加, 由于业务的突发性, 在短时间内一部分缓冲溢出而另一部分连接有剩余带宽的概率也增大了. HFFQ 调度算法能优化合理地分配剩余带宽, 所以性能有较大的提高.

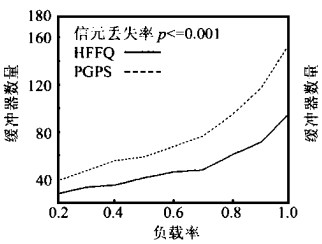

图 2 信元丢失率< 10^{-3}, 缓冲容量和负载

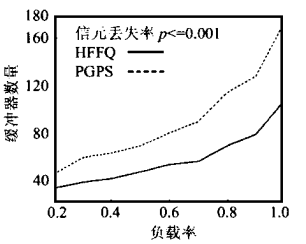

图 3 信元丢失率< 10^{-4}, 缓冲容量和负载

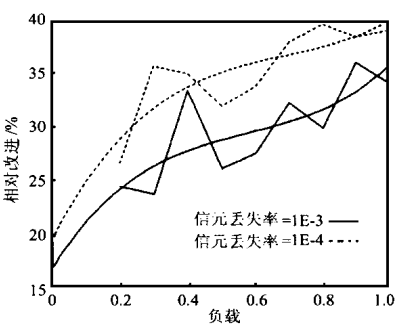

图 4 HFFQ 在不同负载下的相对改进

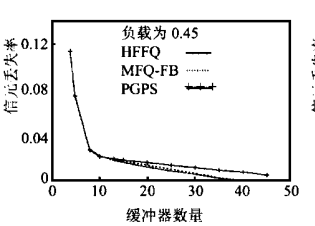

图 5 信元丢失率与缓冲的关系

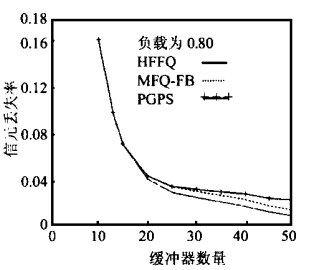

图 6 信元丢失率与缓冲的关系

5.2 缓冲器大小对信元丢失率的影响

首先, 研究均匀共享带宽的情况. 总计有 20 个连接来共享 622Mbps 的带宽, 假定每个连接得到相同的带宽份额, 连接 $0^{\#} \sim 4^{\#}$ 为实时业务, 其余的分为三组, 每组 5 个连接.

图 5、6 分别在中等负载和较重负载条件下, 缓冲器数量对信元丢失率影响很大. 当缓冲器很少时(负载等于 0.45 时, 缓冲器小于 8 个信元; 负载等于 0.80 时, 缓冲器小于 20 个信元), 有可能一次突发的信元就有一些要被丢弃, 此时, 信元丢失率很高, 调度机制起不了太大的作用, 因此, PGPS、MFQ_FB 和 HFFQ 三种算法的性能非常接近.

当缓冲容量超过了一定的量, 调度机制开始起作用. 从图 5、6 可以看到, 在中等负载和较重负载条件下, 信元丢失性能非常相似, MFQ_FB 的性能比 PGPS 要好. HFFQ 比 MFQ_FB 又要好. 随着缓冲量的增加, 性能的改善愈来愈多. 主要原因是当缓冲器增加时, 更多的突发信元能够缓存起来, HFFQ 调度机制能使那些快要溢出的队列迅速恢复常态, 让所有的队列尽可能地保持低的缓冲占用率, 这样, 下一个突发到达时, 溢出的概率会大大减少.

5.3 实时业务和非实时业务的公平性

表 1 显示了在一定负载(= 0.45) 下, 三种调度机制各自 4 组连接的信元丢失率和缓冲之间的关系. 从表中可以看到, MFQ_FB 和 PGPS 在任何缓冲容量下, 各组连接具有同等数量级的信元丢失率. 而 HFFQ, 当缓冲数量达到一定值后, 调度机制发生明显效果, $0^{\#}$ 组 $G0$ 的信元丢失率比其它组降低许多. 在其它负载情况下, 也有相同的结论.

表 1　负载为 0.45, 三种调度机制下各组的信元丢失率比较

缓冲器	HFFQ(%)				MFQ_FB(%)				PGPS(%)			
	$G0$	$G1$	$G2$	$G3$	$G0$	$G1$	$G2$	$G3$	$G0$	$G1$	$G2$	$G3$
5	2.03	1.92	1.77	1.81	1.97	1.89	1.72	1.74	2.01	1.91	1.77	1.79
10	0.57	0.67	0.61	0.48	0.57	0.63	0.52	0.53	0.61	0.62	0.52	0.46
15	0.22	0.24	0.51	0.42	0.42	0.51	0.44	0.45	0.48	0.54	0.55	0.32
20	0.09	0.39	0.41	0.40	0.34	0.40	0.34	0.38	0.40	0.44	0.49	0.35
30	0	0.20	0.21	0.20	0.18	0.21	0.12	0.18	0.38	0.42	0.18	0.23
35	0	0.12	0.10	0.1	0.1	0.1	0.06	0.1	0.22	0.35	0.24	0.17

6　结束语

本文提出的基于帧的层次公平队列 HFFQ, 结合了基于帧调度器的简单性和 GPS 的良好性能, 用计数的方法替代系统虚拟时间的计算, 总的算法复杂度为 $O(1)$, 能同时支持实时业务和非实时业务. 仿真研究采用 20 个连接共享 622Mbps 的链路, 结果显示, HFFQ 比 PGPS 有更好的性能, 如当缓冲器为 40(信元) (负载= 0.45) 或 65(信元) (负载= 0.8) 时, HFFQ 的信元丢失率为零, 同时在调度组之间能保持较好的公平性. HFFQ 的层次结构, 使它能支持大量的连接, 具有很好的可扩展性. 基于 HFFQ 所表现出来的性能可以认为, 它具有一定的实用前景.

参考文献:

[1] Parekh A, Gallager R. A generalized processor sharing approach to flow control-the single node case [A]. Proc IEEE infocom' 92, Genova [C], 1992: 915- 924.

[2] 景志刚, 李乐民, 孙海荣. RED 分组丢弃算法性能研究 [J]. 电子学报, 2000, 28(4): 4- 9.

[3] 王宏宇, 顾冠群. 集成服务网络中的分组调度算法研究综述 [J]. 计算机学报, 1999, 22(10): 1090- 1099.

[4] Stephens D C, Bennett J C R, Zhang H. Implementing scheduling algorithms in high-speend networks [J]. J-SAC, 1999, 17(6): 1145- 1158.

[5] Katevenis M, Sidiropoulos S, Courcoubetis C. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip [J]. J-SAC, October. 1991, 9(8).

[6] Shreedhar M and Varghese G. Efficient fair queueing using deficit round robin [J]. In Proceedings of SIGCOMM' 95, September 1995: 231- 242.

[7] Golestani S J. A self-clocked fair queueing scheme for broadband applications [A]. Proc. IEEE INFOCOM' 94 [C], Toronto, Canada, June 1994: 636- 646.

[8] Lai M O, Tang H K. Modified fair queueing for finite buffer in ATM networks [A]. IEEE international conference on Communications 99' [C]. June, 1999: 193- 198.

作者简介:

姜宁康　男. 1965 年出生江苏镇江, 1993 年于合肥工业大学计算机系获得硕士学位, 后在江苏理工大学从事教学科研工作, 讲师. 现在上海交通大学通信与信息系统专业攻读博士学位. 主要研究兴趣: 软件工程、宽带接入设备、分组调度算法、MPLS 等, 已发表学术论文十余篇.

李毓麟　男. 1941 年 3 月生于上海, 1965 年毕业于清华大学无线电电子学系. 1982 年获上海交通大学电子工程系硕士学位. 现任上海交通大学教授和博士生导师、上海交通大学区域光纤通信网国家重点实验室网络研究室主任, 长期从事局域网的科研和教学工作. 曾获上海市育才奖、上海市科技成果二等奖. 著有《光纤区域网的原理和设计》, 发表相关论文六十余篇.