基于多服务类比例公平的分组调度算法*

江 勇** 吴建平
(清华大学计算机系, 北京 100084)

摘要 研究了同时满足多性能目标的资源管理模型和算法, 基于服务质量模型实现比例公平性原则, 考虑了包括延迟、丢失率在内的服务质量要求及对于综合的多服务类数据流非常重要的公平性问题, 提出了一种分组网络中的比例公平调度模型, 该调度模型综合描述了网络效率、用户 QoS 要求和系统公平性等多维目标. 分析论证了能有效满足比例公平的调度策略 PFS(proportional fairness scheduling), 同时在 Linux 上实现了该算法. 最后在模拟和实验测量的基础上对 PFS 调度算法的性能和系统开销进行了分析讨论.

关键词 比例公平性原则 分组调度 QoS 公平性

随着网络技术的不断发展, 各种新的网络应用对分组交换网络中调度策略的转发效率、带宽、延迟、丢失率以及系统公平性等都提出了新的要求, 如何同时满足这些要求是当前的研究难点. 其困难在于: 网络中存在资源和策略机制方面的限制, 如网络带宽资源、处理器速度, 以及控制策略的设计是否合理等等, 而且吞吐率、分组延迟和丢失率等多种服务质量要求很难同时满足, 有时候这些目标是相互抵触的, 比如过分要求减小丢失率可能导致分组延迟的大幅度增大, 因而任何一种实际的分组调度策略都只能对多个服务质量要求进行折衷.

这就引出了一个问题, 如何设计一种分组交换网络中的调度策略, 或者说资源分配机制, 使其能够有效而公平地分配网络资源, 而且更重要的问题在于不应该只考虑单个性能目标(如吞吐率或者延迟等等)的要求, 而应该考虑多个性能目标的、综合的要求, 而这在目前的网络技术研究中还是未解决的问题[1].

以前对网络中调度策略的研究工作主要集中在问题的某个方面, 比如某一个性能目标的要求或者某些特定领域的综合性能研究. 例如, 有几种新的流模型, 确定的[2, 3]或随机的[4~6], 被提出来作为端到端网络的分析和解决诸如延迟、吞吐率和存储空间等性能参数的方法; 在文献[7]中, 作者比较了吞吐率和延迟抖动对不同 IP 分组的影响, 进而提出了一种非对称尽力发送服务模型, 对两种 IP 分组提供不同的吞吐率和延迟抖动; 文献[8]中研究了传统强迫优化算法(classical constrained optimisation)和遗传算法(genetic algorithm)在吞吐率、公平性和时间复杂度方面的性能差异, 作者随后提出了一种综合的折衷方案, 但它主要关注的是带宽分配; 文献[9, 10]分析了 Web 服务器在服务实时性和网络吞吐量方面的综合要求, 建立了一种该系统的随机 Petri 网性能分析模型, 文献[11]提出了将网络分组传输延迟和丢失率控制的区分服务

2002-02-07 收稿, 2002-05-24 收修改稿
* 国家自然科学基金(批准号: 69725003, 90104002)、广东省自然科学基金(034308)和国家“八六三”(2001AA121013)资助项目
** E-mail: jyong@csnet1.cs.tsinghua.edu.cn

要求合并在一起的综合方案.

我们注意到, 到目前为止还缺乏有效的综合多种性能目标的分组调度策略. 在本文中, 我们力图在这方面做一些理论上的探讨. 提出了基于丢失率和转发延迟的比例公平调度策略(proportional fairness scheduling, PFS), 该调度策略在综合考虑网络效率、用户的服务质量(QoS)要求和系统公平性等多方面的性能目标的基础上提出了一种综合方案, 本文对比例公平调度策略及其可行性给出了严格的证明. 我们的研究工作也提供了对网络调度策略及性能研究有益的理论探索.

1 相关背景

1.1 比例公平原则

Internet 应用之间以及使用者之间有着非常不同的服务要求, 这使得目前的同一服务模型在某些情况下存在着较大的局限性. 在相对区分服务(relative differentiated services)[12]中, 网络流被组合成一个个的服务类(service classes), 这些服务类按照其分组转发质量要求进行排序以决定其排队延迟、分组丢失率等转发行为. 假定网络流被分类组合成排序的服务类, 类 i 应有更好于(或者至少不差于)类 $i-1(1<i<N)$ 的服务质量(排队延迟和分组丢失率). 注意“至少不低于”是必须的, 因为在低负载的情况下所有分组的服务质量是相同的, 即满足所有流的服务要求. 这样 Internet 用户或者应用程序能够选择最符合它们要求的质量和价格限制的服务类. 在这种情况下, 由于没有准入控制和资源预留, Internet 应用程序和使用者不能获得绝对的服务质量保证, 如端到端的延迟界限和带宽等, 但网络能保证高级别的类享有比低级别类相对更好的服务. 在文献[1]中,作者提出了一种比例公平性原则(proportional fairness principle)来达到相对区分服务的要求.

比例公平性原则按照网络管理者给定的区分参数按比例分配网络资源, 从而得到相应的服务性能. 若用 q_i 代表服务类 i 的性能量值, 则比例公平性原则给每对服务类加入如下的限制:

$$\frac{q_i}{q_j}=\frac{c_i}{c_j} \qquad (i,j=1,\cdots,N), \tag{1}$$

其中 $c_1<c_2<\cdots<c_N$ 是一般的服务质量区分参数. 因此, 即使每个类的服务质量随着其负载变化, 但类间的服务质量比值是不变的, 与负载无关.

考虑基于排队延迟的公平性, 采用排队延迟作为比例公平性原则的性能参数. 若用 $\hat{d}_i$ 代表服务类 i 分组的队列延迟界限, 比例公平性原则可以表述为对所有服务类 i 和 j 有

$$F=\frac{\hat{d}_i}{\delta_i}=\frac{\hat{d}_j}{\delta_j} \qquad (i,j=1,\cdots,N), \tag{2}$$

其中参数$\{\delta_i\}$是用户要求的延迟区分参数(delay differentiation parameters, DDPs), 由于高级别类有更好的服务性能, 故有 $\delta_1>\delta_2>\cdots>\delta_N>0$.

同样, 用 $\hat{l}_i$ 代表类 i 的丢失率限制, 基于分组丢失率的比例公平性原则要求服务类间的丢失率满足

$$F=\frac{\hat{l}_i}{\sigma_i}=\frac{\hat{l}_j}{\sigma_j} \qquad (i,j=1,\cdots,N), \tag{3}$$

其中, σ_i是丢失率区分参数(loss rate differentiation parameters, LDPs), 它们的大小顺序为$\sigma_1 > \sigma_2 > \cdots > \sigma_N > 0$.

1.2　服务描述函数

服务描述函数的概念最早出现在 Parekh 和 Gallager 的工作[13]中, 他们在其调度算法中引入了一种通用的服务描述函数, 这种方案的一个优点在于它能把对服务质量的要求通过一个简单的描述函数表示出来, 并将一个网络的连接的服务特性和其他的网络连接区别开来; 它的另一个重要特点是给了服务器更大的灵活性来为达到不同的延迟和吞吐率要求分配系统资源.

Cruz 在文献[14, 15]提出了一种更为宽松的服务曲线的概念, 它采用了更宽松的服务定义来作为对服务特性的通用描述框架; 另外在文献[16, 17]中给出了更严格但较缺少一般性的服务定义. 本文中采用的服务描述的定义也可见文献[18, 19].

2　比例公平函数

本文假定系统时间分成一个个的小的时间片, 编号为 0, 1, 2, …. 考虑将所有分组分为 M 个服务类, 并假定在每个时间片内网络服务器(交换机或路由器)能服务 c 个分组, c 称作服务器的服务能力. 注意在本文中我们考虑了分组丢失.

我们用 $R_i^{\text{in}'}[t]$ 代表在时间片 t 服务类 i 中到达服务器的所有分组数, $L_i[t]$代表该服务类在 t 中丢失的分组数, 令 $R_i^{\text{in}}[t]=R_i^{\text{in}'}[t]-L_i[t]$表示到达服务器并最终得到服务的分组数; 用 $R_i^{\text{out}}[t]$ 代表在时间片 t服务类 i离开服务器的分组数, $Q_i[t]$代表在时间片 t末暂存在服务器中的分组数, 其中 t 是一个非负整数. 不失一般性, 令 $R_i^{\text{in}}[0]=R_i^{\text{in}'}[0]=0$, $R_i^{\text{out}}[0]=0$, $Q_i[0]=0$, $L_i[0]=0$.

定义 $R_i^{\text{in}}[s,t]$ 为在时间间隔[s, t]内到达的分组数, $R_i^{\text{in}}[s,t]=\sum_{m=s}^{t} R_i^{\text{in}}[m]$, 其中 s 和 t 均为非负整数. 若 $s > t$, 定义 $R_i^{\text{in}}[s,t]=0$. 同样, $R_i^{\text{out}}[s,t]$定义为在时间间隔[s, t]内离开的分组数. 为了简化, 在后面我们集中讨论一个给定的服务类并省略下标 i.

假定一开始服务器中没有分组. 因而在时间片 t 结束时服务器中的暂存分组数为

$$Q[t] = R^{\text{in}}[1, t]-R^{\text{out}}[1, t]\geqslant 0, \tag{4}$$

相对于 t 的丢失率 $l(t)$定义为

$$l(t)=\frac{L[0,t]}{R^{\text{in}'}[0,t]}, \tag{5}$$

相对于 t 的延迟 $d[t]$定义为

$$d[t]=\min\{\Delta: \Delta\geqslant 0 \text{ and } R^{\text{in}}[1, t]\leqslant R^{\text{out}}[1, t+\Delta]\}. \tag{6}$$

注意到如果该服务类的分组离开服务器的顺序和其到达顺序一致(FIFO), 那么 $d[t]$的上限为在 t 中到达的分组的延迟.

定义 1 (突发性限制)　给定一个非减函数 $b(\cdot)$作为到达函数, 我们说输入网络流 $R^{\text{in}'}$ 是 b 形的, 若对所有满足 $s\leqslant r$ 的 s 和 t 有 $R^{\text{in}'}[s+1, t]\leqslant b(t-s)$成立. 对于某些特殊情形, 如 $b(x)=\sigma+\rho x$, 我们说 R^{in}是(σ, ρ)形的.

由前面的延迟比例公平性原则[1]有 $\frac{d_i}{\delta_i}=\frac{d_j}{\delta_j}=\tilde{d}$, 其中 $\tilde{d}$ 为延迟比例公平参数, 其值将在

后面的比例公平调度策略的可行性分析中讨论, 在此我们将其看作一个已知量. 容易发现, 若服务器对每个服务类 i 均满足延迟条件 $d_i^{\max}=\delta_i\tilde{d}$, 则服务器在服务类间满足延迟比例公平性原则((2)式).

定义 2 (延迟比例函数) 假设服务类 i 的网络流符合某种 b 形, 要求服务器产生的延迟最大为 $d_i^{\max}=\delta_i\tilde{d}$ 个时间片. 假定 $P_i^D(\cdot)$ 是一非减函数,

$$P_i^D(t)=\begin{cases}0, & \text{if } 0\leqslant t\leqslant d_i^{\max}-1;\\ b(t-d_i^{\max}), & \text{if } t\geqslant d_i^{\max}.\end{cases} \tag{7}$$

若对任意 t 存在 $s\leqslant t$ 使 $Q_i[s]=0$ 且 $R_i^{\text{out}}[s+1,t]\geqslant P_i^D(t-s)$, 则我们说服务器保证延迟比例函数 $P_i^D(\cdot)$.

容易发现, $P_i^D(t-s)$ 描述了在给定的时间间隔[s+1, t]中必要的最小离开服务器分组数, 其中 t 是任意给定的时间片, s 是某个不大于 t 的时间片, 且在该时间片末暂存分组数为零.

接下来的两条定理[14]显示若一个到达的数据流符合定义 1 的限制并且保证延迟比例函数, 则延迟和暂存分组数是有界的. 在文献[20, 21]中有过类似的描述.

定理 1 (延迟上界) 考虑一个保证延迟比例函数 $P^D(\cdot)$ 的服务器, 并假定输入流 R^{in} 是 b 形的. 则对每一个 t, 延迟 $d[t]$有上界

$$d[t]\leqslant \max_{s:s\geqslant 1}\min\{\Delta:\Delta\geqslant 0 \quad \text{and} \quad b(s)\leqslant P^D(s+\Delta)\}. \tag{8}$$

容易发现, 延迟 $d[t]$的上界为 $d_i^{\max}=\delta_i\tilde{d}$.

定理 2 (暂存分组数上界) 假定服务器有 $P^D(\cdot)$的延迟比例函数保证且输入流 R^{in} 是 b 形的, 则对 t, 暂存分组数 $Q[t]$有上界

$$Q[t]\leqslant \max_{s:s\geqslant 0}\{b(s)-P^D(s)\}. \tag{9}$$

由延迟比例公平原则和延迟比例函数的定义以及定理 1 可知: 若服务器对每一服务类保证延迟比例函数 $P^D(\cdot)$且输入流 R^{in} 是 b 形的, 则服务器满足延迟比例公平性原则.

接下来, 我们分析服务器的丢失率控制.

定理 3 (丢失率控制) 假定 $R^{\text{in}'}[t]$符合到达函数 $b(\cdot)$并且服务器保证延迟比例函数 $P^D(\cdot)$. 若分配给服务类 i 的缓冲区为 B 且存在一个常数 $\hat{l}$ 使得

$$P^D(t)\geqslant b(t)(1-\hat{l})-B, \quad \forall t\geqslant 0, \tag{10}$$

则丢失率 $l(t)$上限为 $\hat{l}$.

由定理 3 可得要保证 $\hat{l}$ 的丢失率上限, 需要为服务类保留 $B\geqslant b(t)(1-\hat{l})-P^D(t)$的缓冲区空间. 根据比例公平性原则[1]有 $\frac{l_i}{\sigma_i}=\frac{l_j}{\sigma_j}=\tilde{l}$, 可得服务类 i 应保持的丢失率上限为 $\hat{l}_i=\sigma_i\tilde{l}$. 其中 $\tilde{l}$ 称为丢失率比例公平参数.

定理 4 (丢失率比例公平) 假定服务类 i 的 $R_i^{\text{in}}[t]$ 符合到达函数 $b_i(\cdot)$并且服务器保证延迟比例函数 $P_i^D(\cdot)$. 若分配给服务类 i 的缓冲区 B_i 满足

$$B_i=\max\{b_i(t)(1-\sigma_i\tilde{l})-P_i^D(t), \ t\geqslant 0\}, \tag{11}$$

则服务器对服务类的丢失率满足比例公平性原则.

由于所有服务类缓冲区 B_i 的总和要小于系统总的缓冲区空间 B_{total}, 即 $\sum_{i=1}^{M} B_i \leqslant B_{\text{total}}$, 故丢失率比例公平参数 $\tilde{l}$ 应满足 $\tilde{l} \geqslant \dfrac{\sum_{i=1}^{M}\left[b_i(t)-P^D{}_i(t)\right]-B_{\text{total}}}{\sum_{i=1}^{M}\sigma_i b_i(t)}$.

3　比例公平调度策略

我们研究如图 1 所示的 M 个服务类的网络传输控制模型.

每个逻辑分组队列对应着一个服务类. 每一个到达分组根据它的分类标识入队这些队列中的一个. 分组调度器(packet scheduler)决定哪个类将被服务, 以获取类间必要的延迟区分. 另一个模块, 我们叫它缓冲控制器(backlog controller), 管理在转发引擎中等待发送的分组的缓冲队列, 它决定是否应该有分组被丢弃. 最简单的缓冲控制器是尾部丢弃(drop-tail)的缓冲管理, 当没有可用的缓冲资源时丢弃分组. 更好的可以采用如 RED[22]的丢弃策略. 当一个分组不得不被丢弃时, 分组丢弃模块(packet dropper)选取一个缓冲队列. 也就是说缓冲控制器管理转发引擎整个的缓冲容量, 而分组丢弃模块控制类间的分组丢失率区分. 丢弃模块从相应队列的末尾移除一个分组.

考虑一个 M 个服务类的服务器. 假定服务类 i 需要服务器保证丢失率限制以及延迟比例函数 $P_i^D(\cdot)$, i=1, …, M. 我们希望设计一种调度策略按照比例公平性原则提供延迟界限保证和丢失率界限保证.

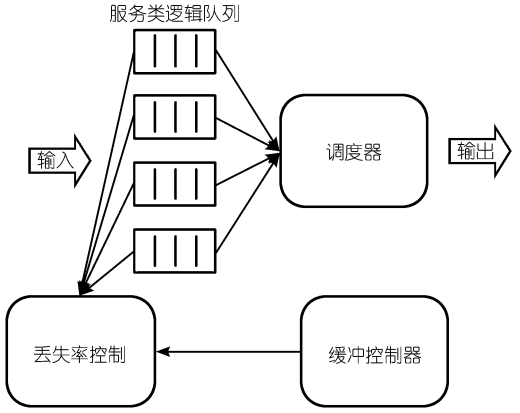

图 1　M 个服务类的网络传输控制模型

根据定理 4 中丢失率比例公平的条件, 在输入侧由缓冲控制器和丢失率控制对每服务类队列的缓冲区大小和分组丢弃策略按照(11)式给予限制, 从而对服务器内的服务类提供比例公平的丢失率界限保证.

而对调度器的设计遵从延迟比例公平的原则. 为简化起见, 我们假设 $P_i^D(\cdot)$取整数值. 一个服务器称为空的条件是所有服务类在服务器中的暂存分组数在时间片 t 末均为 0, 也就是说 $Q_i[t]$=0, i=1, …, M. 对于一个非负整数 t, 定义 $\tau(t)$为不大于 t 的最后一个服务器为空的时间片, 也即

$$\tau(t) = \max\{s: s\leqslant t, \quad Q_i[s]=0, i=1, \cdots, M\}. \tag{12}$$

我们说服务器对服务类 i 保证目标输出函数 $Z_i(\cdot)$, 若对每 t 有

$$R_i^{\text{out}}[\tau(t)+1, t] \geqslant Z_i(t), \tag{13}$$

其中

$$Z_i(t)=\min_{\substack{s:\tau(t)\leqslant s\leqslant t\\ Q_i[s]=0}}\{R_i^{\text{out}}[\tau(t)+1, s]+P_i^D(t-s)\}. \tag{14}$$

引理 1　服务类 i 保证延迟比例函数 $P_i^D(\cdot)$的充分条件是该服务类保证目标输出函数 $Z_i(\cdot)$.

引理 1 启发我们思考比例公平调度策略(PFS). 考虑一种策略给每个到达分组分配一个满足(13)式的限期(deadline), 并按限期的顺序发送分组. 目标输出 $Z_i(t)$ 在 t 前是不可知的, 但在不考虑暂存分组数清零时可以做一个估计. 其中每个服务类的缓冲区大小和丢弃策略按照定理 4 设计.

设服务类 i 在时间片 u 内到达分组的序号是在[$R_i^{\text{in}}[\tau(u-1)+1,\ u-1]+1$, $R_i^{\text{in}}[\tau(u-1)+1,\ u]$]间的惟一的整数.

注意到若在每个时间片内最多有一个分组到达时, 则到达序号简化为服务类 i 自上一次服务器空后的到达分组数. 完整的调度策略描述如下:

定义 3 按照(11)式计算和分配各服务类的缓冲区, 根据缓冲空间和丢弃策略决定到达分组丢弃或入队, 对进入服务器的每个分组分配一个限期, 按持续工作方式有最早限期的分组先得到服务, 特别地, 在某个时间片 u 内, 有 $\sum_{i=1}^{M}\left(Q_i[u-1]+R_i^{\text{in}}[u]\right)$ 个分组可以离开服务器, 并有最多 c 个最早限期的分组在时间片 u 中被选中离开. 服务类 i 中在时间片 u 到达的分组被分配如下的限期:

$$D_i=\min\{t:\ t\geqslant u \quad \text{and} \quad Z_i(t:\ u-1)\geqslant n_i\}, \tag{15}$$

其中

$$Z_i(t;\ u)=\min_{\substack{s:\tau(u)\leqslant s\leqslant u\\ Q_i[s]=0}}\{R_i^{\text{out}}[\tau(u)+1,s]+P_i^D(t-s)\}, \tag{16}$$

n_i 为分组的到达序号.

注意 $Z_i(u;\ u)=Z_i(u)$ 为(14)式中定义的目标输出函数. 另外, $Z_i(t;\ u)$ 是 $Z_i(t)$ 在时间片 u 的估计值.

下面介绍两条引理. 它们表明若服务器采用 PFS 策略调度分组, 并且没有错过限期的情况, 那么服务类 i 保证延迟比例函数 $P_i^D(\cdot)$, $i=1,\ \cdots,\ M$. 在此, 分组错过限期是指该分组在限期过后才离开服务器.

引理 2 设服务器实现 PFS 调度策略, 并且没有错过限期的情况, 那么对于任意的非负整数 t, 服务类 i 中到达并被分配限期在时间段[$\tau(t)+1,\ t$]内的分组数至少为 $Z_i(t)$.

引理 3 假设服务器采用 PFS 调度策略, 并且限期均未错过, 那么, 服务类 i 保证了延迟比例函数 $P_i^D(\cdot)$, $i=1,\ \cdots,\ M$.

定理 5 (可行性分析) 假定一容量为 c 的服务器给 M 个服务类提供服务. 若对所有非负整数 t 有 $\sum_{i=1}^{M}P_i^D(t)\leqslant ct$, 则定义 3 中描述的 PFS 策略保证服务类 i 的延迟比例函数 $P_i^D(\cdot)$, $i=1,\ \cdots,\ M$.

下面我们讨论延迟比例公平参数的取值, 由定理 5 可得, 延迟比例函数的可行性条件为 $\sum_{i=1}^{M}P_i^D(t)\leqslant ct$, 又由延迟比例函数的定义: $P_i^D(t)=b_i(t-\delta_i\tilde{d})$, 故前述的延迟比例公平参数必须满足条件: $\sum_{i=1}^{M}b_i(t-\delta_i\tilde{d})\leqslant ct$.

4 PFS 算法实现

在这一节中, 我们将根据(15)式推导在 PFS 调度策略中计算限期的算法.

在服务类 i 中到达时刻为 u 的分组 $n_i = R_i^{\text{in}}[1,u-1]+l$ 的限期可由

$$\begin{aligned} D_i(n_i) &= \min\{t:\ t\geqslant u \quad \text{and} \quad Z_i(t;\ u-1)\geqslant n_i\} \\ &= \max\{u,\ Y_i(l;\ u)\} \end{aligned} \tag{17}$$

得到，其中 $Y_i(l;\ u) = \min\{t:\ Z_i(t;\ u-1)\geqslant R_i^{\text{in}}\ [1,\ u-1]+l\}$.

我们定义

$$S_i^{-1}(n)=\min\{m:\ m\geqslant 1 \quad \text{and} \quad P_i^D(m)\geqslant n\}. \tag{18}$$

下面推导 $Y_i(l;\ u)$的递归算法. 对于 $l\geqslant 1$，有

$$\begin{aligned} Y_i\left(l;\ u\right) &= \min\{t: Z_i(t;\ u-1)\geqslant R_i^{\text{in}}[1,u-1]+l\} \\ &= \min\{t: \min_{\tau(u-1)\leqslant s\leqslant u-1}\{R_i^{\text{in}}[1,s]+P_i^D(t-s)\}\geqslant R_i^{\text{in}}[1,\ u-1]+l\} \\ &= \max_{\tau(u-1)\leqslant s\leqslant u-1}\{\min\{R_i^{\text{in}}[1,s]+P_i^D(t-s)\}R_i^{\text{in}}[1,\ u-1]+l\} \\ &= \max_{\tau(u-1)\leqslant s\leqslant u-1}\{s+S_i^{-1}(l+R_i^{\text{in}}[s+1,u-1])\}. \end{aligned} \tag{19}$$

假定服务器在时间片 τ_0 末系统重置，即 $\tau(\tau_0)=\tau_0$，再假定直到时间片 u_1 服务类 i 中均没有分组到达，也即 $R_i^{\text{in}}[\tau_0+1,\ u_1-1]=0<R_i^{\text{in}}[u_1]$，则根据(19)式可得

$$Y_i(l;\ u_1) = u_1-1+S_i^{-1}(l). \tag{20}$$

递归处理，设在时间片 u_m 到 u_{m+1} 之间服务类 i 没有分组到达，$R_i^{\text{in}}[u_m+1,\ u_{m+1}-1]=0<R_i^{\text{in}}[u_m]$，服务器在$[u_m,\ u_{m+1}-1]$内也没有重置，$\tau(u_m-1)=\tau(u_{m+1}-1)$. 此时，由(19)式有

$$\begin{aligned} &Y_i(l;\ u_{m+1}) \\ &= \max\{Y_i(l+R_i^{\text{in}}[u_m,u_{m+1}-1];u_m),\quad \max_{u_m\leqslant s\leqslant u_{m+1}-1}\{s+S_i^{-1}(l+R_i^{\text{in}}[s+1,u_{m+1}-1])\}\} \\ &= \max\{Y_i(l+R_i^{\text{in}}[u_m];\ u_m),\ u_{m+1}-1+S_i^{-1}(l)\}. \end{aligned} \tag{21}$$

因而，为实现服务类 i 限期的计算. 可以采用(20)和(21)式递归算出在每个时间片 u 内到达分组的 $Y_i(\cdot\ ;\ u)$，然后根据(18)式算出限期. 一般情况下，这比较困难，除非 $Y_i(\cdot\ ;\ u)$能够用几个参数表示.

一种可能情况是 $S_i^{-1}(l)=|\Delta_i^0+\eta_i l|$ 对 $l>M_i$ 成立，M_i 为某个整数，而 Δ_i^0 和 η_i 是常数. 此时，$Y_i(l;\ u_{m+1})=\lceil\Delta_i(u_{m+1})+\eta_i l\rceil$ 对 $l>M_i$ 成立，其中 $\Delta_i(u_{m+1})$ 能通过 $\Delta_i(u_m)(m\geqslant 1)$ 递归得到. 其递归过程如下:

对于 $m=1$ 和 $l>M_i$，由(20)式可得

$$\begin{aligned} Y_i\left(l;u_1\right) &= u_1-1+S_i^{-1}\left(l\right) = u_1-1+\left|\Delta_i^0+\eta_i l\right| \\ &= \left\lceil\left(u_1-1+\Delta_i^0\right)+\eta_i l\right\rceil \\ &= \left\lceil\Delta_i\left(u_1\right)+\eta_i l\right\rceil, \end{aligned}$$

其中

$$\Delta_i(u_1)=u_1-1+\Delta_i^0. \tag{22}$$

假设 $Y_i\left(l;u_m\right)=\left\lceil\Delta_i\left(u_m\right)+\eta_i l\right\rceil$ 对于 $m\geqslant 1$ 和 $l>M_i$ 成立，根据(21)式可以得到

$$
\begin{aligned}
&Y_i(l;u_{m+1})\\
&=\max\left\{Y_i\left(l+R_i^{\text{in}}[u_m];u_m\right),u_{m+1}-1+S_i^{-1}(l)\right\}\\
&=\max\left\{\left\lceil \Delta_i(u_m)+\eta_i\left(l+R_i^{\text{in}}[u_m]\right)\right\rceil,u_{m+1}-1+\left\lceil \Delta_i^0+\eta_i l\right\rceil\right\}\\
&=\left\lceil \max\left\{\Delta_i(u_m)+\eta_i\left(l+R_i^{\text{in}}[u_m]\right),u_{m+1}-1+\left\lceil \Delta_i^0+\eta_i l\right\rceil\right\}\right\rceil\\
&=\left\lceil \Delta_i(u_{m+1})+\eta_i l\right\rceil,
\end{aligned}
$$

其中

$$\Delta_i(u_{m+1})=\max\left\{\Delta_i(u_m)+\eta_i R_i^{\text{in}}[u_m],u_{m+1}-1+\Delta_i^0\right\}. \tag{23}$$

$Y_i(l;u)(l=1,2,\cdots,M_i)$的值存储在一个有 M_i 个元素的表里，递归更新. 这就得到了图 2 中的计算分组限期的算法. 该算法需要 $O(M_i)$ 存储空间保存服务类 i 的状态变量，以及在每个有服务类 i 分组到达的时间片里进行 $O(M_i)$ 次比较和内存访问操作. 在足够的硬件支持下，比较和内存访问操作可以 $O(1)$时间并行流水操作.

上述的 $S_i^{-1}(l)=\left\lceil \Delta_i^0+\eta_i l\right\rceil$ 情况对于一般的数据流类型均可以满足(如 CBR, ON/OFF, Poisson 流等)，而对于更一般的情况，也就是延迟比例函数 $P_i^D(x)$ 形式任意的情况，此时 M_i 的大小简单说依赖于延迟比例函数在时间轴的变化，故可以采用分段线性函数对延迟比例函数进行近似 [1].

5 性能与分析

我们在模拟器和 Linux 2.4.2 内核中实现了 PFS 调度策略. 模拟器和 Linux 中的实现代码是基本一致的，惟一区别是在 Linux 的实现中采用 Intel Pentium Ⅲ处理器的 CPU 时钟周期作为时钟计数器来进行分组的限期计算操作.

评价 PFS 的模拟实验运行在网络模拟系统 ns-2[2]上，该系统提供了在分组级别的网络协议、缓冲管理和调度算法的不同实现. 我们将结合传输协议和流量模型考察 PFS 算法的行为特性.

在 Linux 中，除调度器(scheduler)之外，我们也实现了一种 Ipv4 分组的分类器(classifier).

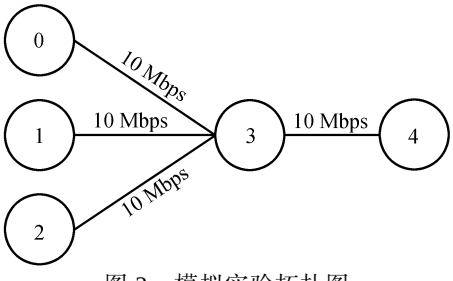

图 2 模拟实验拓扑图

我们将通过模拟和实验测量来评价 PFS 算法. 所采用的硬件配置为，双 Intel Pentium Ⅲ 800MHz CPU, 512KB L2 cache, 1GB of RAM, 以及 3COM Etherlink XL PCI (3C905-TX)的网络接口卡. 我们在运行过程中由系统根据 CPU 时钟周期计数器自动记录事件日志(如 enqueue 和 dequeue)，然后对系统输出的记录结果进行分析处理. 在接下来的内容中，包括 PFS 两方面的性能结果: 1)PFS 提供比例公平性的能力; 2)本文算法实现的系统开销.

1) Saha D, Mukherjee S, Tripathi S K. Multirate scheduling of VBR video traffic in ATM networks. Ph. D. dissertation, Dept Elect Comput Eng, Univ California at San Diego, 1996

2) Ucb/lbnl/vint network simulator - ns (version 2)

5.1　比例公平性

我们的模拟网络的拓扑结构如图 2. 节点 0, 1 和 2 通过 10 Mbps 的链路与节点 3 相连, 而节点 3 和 4 通过 10 Mbps 链路连结. 我们有三个服务类 0, 1 和 2. 从节点 0, 1, 2 中出来的数据流分别对应上述的三个服务类, 也就是说从节点 i 出来的数据流为服务类 i, i=0, 1, 2. 节点 3 中的缓冲区大小为 200 KBytes.

考察三个服务类数据流的网络模型. 在我们的实验中, 每个服务类包括一个或多个数据流, 其分布和分配的带宽如表 1 所示. 音频(Audio)流每 20 ms 发送 160 Bytes 分组, 而视频(Video)流每 33 ms 发送 8 KBytes 的分组, 其他数据流发送 4 KBytes 分组, 而 FTP 数据流是持续发送的.

表 1　实验中的数据流

服务类 ID	流类型	分配速率	时间间隔	分组长度	入结点
0	Audio	64 Kbps	20 ms	160 Bytes	0
1	Video	2 Mbps	33 ms	8 KBytes	1
	On-Off	5 Mbps	5 s	4 KBytes	2
2	Poisson	2 Mbps	None	4 KBytes	2
	FTP	5 Mbps	None	4 KBytes	2

要描述 PFS 在延迟比例区分和提供实时应用更小延迟的能力, 我们给音频数据流设定延迟参数为 5, 给视频数据流设定延迟参数 10, 其他数据流的延迟参数为 100. 为了达到这个目标, 需要给音频流赋以延迟比例函数 $P_0^D(t)=b_0(t-5\tilde{d})$, 以及设定视频流延迟比例函数 $P_1^D(t)=b_1(t-10\tilde{d})$, 其中 $\tilde{d}$ 的值由定理 5 确定, 其他数据流的延迟比例函数类似. 所有服务类和数据流的延迟比例函数均为线性的.

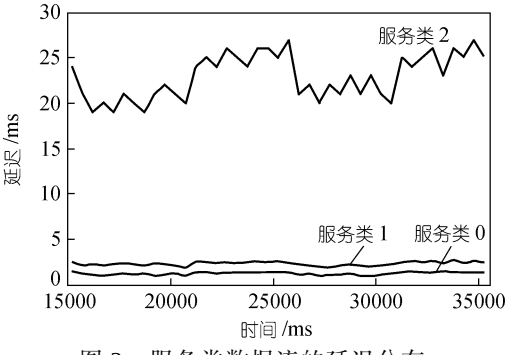

图 3　服务类数据流的延迟分布

图 3 显示了各服务类数据流的延迟分布. 从图中可以看出, PFS 算法能较好地满足延迟的比例区分. 其中的延迟的周期性变化表现了 ON-OFF 数据流的影响, 在 ON-OFF 数据流激活时各服务类的延迟均增加了, 但音频流和视频流相应于其实时性要求增加的幅度较小. 这反映了 PFS 算法能够根据网络拥塞情况实时调节各服务类的延迟, 保证其比例公平性, 同时我们看出它对实时性要求较高的音频流和视频流也能在网络拥塞时提供优先的延迟性能.

为考察服务类数据流间的丢失率比例公平, 将音频数据流和视频数据流的丢失率参数(LFP)均设为 $\sigma_0=\sigma_1=5$, 而将其他数据流的丢失率参数(LFP)设为 $\sigma_2=100$. 其他实验设置如前所述. 最后的模拟实验结果如表 2 所示, 从表中数据我们可以看出, PFS 算法基本上达到了服务类间的丢失率比例公平. 但结果并不很令人满意, 这是由于丢失率是一个统计数值, 要进行实时调节很困难. 如何更好地保证丢失率比例公平是我们下一步重点研究的内容.

表 2 服务类流的丢失率比较

服务类 ID	发送分组数	接收分组数	丢失率
服务类 0	5120	5053	1.3%
服务类 1	3200	3154	1.43%
服务类 2	38400	29970	22%

5.2 算法实现的系统开销

在我们的 PFS 算法实现中主要有三类系统开销: 分组分类(packet classification), 入队列(enqueue)和出队列(dequeue).

首先在 Linux 实现中测量了分组分类的系统开销. 为减少分类算法的系统开销, 我们采用了一种哈希算法. 在低负载的情况下, 只有一条数据流的头分组需要全部的分类开销, 而接下来的分组都可以通过该数据流的哈希值进行分类. 最坏情况下随着服务类数据流的增加系统开销也不断增加, 我们测得的平均哈希分类时间为 3 μs.

对于入队列和出队列的系统开销的测量, 我们采用实验服务器(dual 800 MHz Pentium Ⅲ system with 512 KB L2 cache and 1GB of memory running the unchanged Linux 2.4.2 kernel)测量了 Linux 下的实现情况.

我们测量了实验中从第 10000 分组到第 20000 分组的(ⅰ)平均入队列时间, (ⅱ)选中分组的平均出队列时间, 以及(ⅲ)平均每分组排队开销(由系统总时间除以分组数得到).

在实验中我们主要研究了服务类数目增加对系统开销的影响. 当服务类数目从 1 到 1000 按每次增加 100 变化时, 结果显示入出队列时间和每分组排队开销均随着服务类数目呈线性增长. 这和我们预期的一样, 因为每增加一个服务类在更新 PFS 的状态时间上都增加一个固定开销.

6 结论

本文研究了高速分组交换网络中调度策略的综合性能要求, 提出了一种有效地综合网络效率、用户服务质量(QoS)要求和系统公平性等多性能目标的综合调度策略, 并对比例公平调度策略进行了详细分析论证. 本文的主要贡献包括: 1)提出基于比例公平性原则的丢失率和延迟比例函数; 2)提出综合考虑分组延迟和丢失率的用户要求与系统公平性的比例公平调度策略(PFS), 并给出对其详细的分析论证; 3)推导论证了比例公平调度策略的实现算法及其复杂性; 4)对比例公平调度策略在模拟和实验环境下进行了性能分析和讨论. 由于多目标性能调度策略研究的复杂性, 目前还缺乏有效的分组网络综合性能调度策略, 本文的研究是对这方面研究的理论探索, 研究成果可应用于对分组网络调度策略的设计、实现和性能分析优化.

高速分组交换网络调度策略中多个性能目标之间的相互联系和影响, 比例公平调度策略如何向网络链路扩展, 以及在随机状态下的服务描述函数[4~6]等方面还有许多问题仍然是未解决的, 这些问题是作者进一步的研究方向.

参 考 文 献

1 Jiang Y, Lin C, Wu J. Integrated performance evaluating criteria for network traffic control. In: Proceedings of IEEE Symposium on Computers and Communications 2001. Tunisia: IEEE Communications Society Press, 2001

2 Cruz R L. A calculus for network delay, Part I: Network elements in isolation. IEEE Trans on Information Theory, 1991, 37(1): 114~131
3 Cruz R L. A calculus for network delay, Part II: Network analysis. IEEE Trans on Information Theory, 1991, 37(1): 132~141
4 Kurose J. On computing per-session performance bounds in high-speed multi-hop computer networks. In: Proc of ACM Sigmetrics and Performance ′92. New York: ACM Press, 1992. 128~134
5 Yaron O, Sidi M. Performance and stability of communication networks via robust exponential bounds. IEEE/ACM Trans on Networking, 1993, 1(3): 372~385
6 Chang C S. Stability, queue length, and delay of deterministic and stochastic queueing networks. IEEE Trans on Automatic Control, 1994, 39(5): 913~931
7 Hurley P, Boudec J L. A Proposal for an asymmetric best-effort service. In: Proceedings of IWQOS'99. London, 1999. 129~132
8 Pitsillides A, Stylianou G, Pattichis C, et al. Bandwidth allocation for virtual paths (BAVP): Investigation of performance of classical constrained and genetic algorithm based optimisation techniques. In: Proceedings of INFOCOM'2000, Tel Aviv, 2000. 1379~1387
9 林　闯. Web 服务器集群请求分配和选择的性能分析. 计算机学报, 2000, 23(5): 500~508
10 Lin C, Shan Z, Yang Y. Integrated schemes of request dispatching and selecting in web server clusters. In: Proceedings of WCC2000. Beijing: Publishing House of Electronics Industry, 2000, 922~930
11 Lin C, Sheng L, Wu J, et al. An integrative scheme of differentiated services. In: Proceedings of MASCOTS 2000. San Francisco, 2000. 441~448
12 Dovrolis C, Stiliadis D. Relative differentiated services in the internet: Issues and mechanisms. In: Proceedings of ACM SIGMETRICS'99/ Atlata, 1999. 204~205
13 Parekh A K, Gallager R G. 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
14 Cruz R L. Quality of service guarantees in virtual circuit switched networks. IEEE Journal on Selected Areas in Communications, 1995, 13(6): 1048~1056
15 Cruz R L. Service burstiness and dynamic burstiness measures: a framework. Journal of High Speed Networks, 1992, 1(2): 105~127
16 Hung A, Kesidis G. Bankwidth scheduling for wide-area ATM networks using virtual finishing times. IEEE/ACM Transactions on Networking, 1996, 4(1): 49~54
17 Stiliadis D, Varma A. Latency-rate servers: A general model for analysis of traffic scheduling algorithms. IEEE/ACM Transactions on Networking, 1998, 6(5): 611~624
18 Boudec J Y L. Connectionless data service in an ATM-based customer premises network. Computer Networks and ISDN Systems, 1994, 26(11): 1409~1424
19 Agrawal R, Rajan R. Performance bonds for flow control protocols. IEEE/ACM Transactions on Networking, 1999, 7(3): 310~323
20 Reich E. On the integrodifferential equation of Takacs I. Annals of Mathematical Statististics, 1958, 29: 563~570
21 Roberts J W, Virtamo J T. The superposition of periodic cell arrival streams in an ATM multiplexer. IEEE Trans on Communications, 1991, 39(2): 298~303
22 Parekh A K, Gallager R G. 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