核心传输网络节点分层资源分配策略的研究

宋美娜, 宋俊德, 张 益
(北京邮电大学 PCN&CAD 部级重点实验室, 北京 100876)

摘 要: 网络节点的资源分配策略是影响未来 IP 核心传输网络服务质量的关键因素. 本文针对网络节点的数据层面资源分配, 提出了具有混合算法结构的分层调度算法、分层动态门限缓存管理机制. 仿真结果表明采用分层资源分配策略, 可以使网络节点系统在优先级在输出端口间分布不均衡情况下的数据转发率接近理论值.

关键词: 核心传输网; 调度算法; 缓存管理; 服务质量
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2004) 12A-052-05

Study on the Hierarchical Resource Management Policy for Core Transportation Network Nodes

SONG Mei-na, SONG Jun-de, ZHANG Yi
(*Department Key Lab. of PCN&CAD, Beijing University of Posts and Telecommunications, Beijing* 100876, *China*)

Abstract: The resource management policy of network nodes is the key element to influence the Quality of Service (QoS) of the future IP core transportation network. In this paper, aimed to distribute the data path resource, a hierarchical scheduling algorithm with hybrid algorithm structure and a buffer management mechanism with hierarchically dynamic threshold are proposed. The simulation results show that when the hierarchical resource management policy is adopted, the forwarding rate will approximate the theoretical value under the situation with non-uniform priorities distribution among output ports.

Key words: core transportation network; scheduling algorithm; buffer management; QoS

1 引言

未来的通信网络具有更高的接入速率、更大的容量、更高的智能化、更高质量的多媒体通信等特点. 这就要求核心传输网络节点(交换机、路由器)在支持大容量、高速率数据传输的前提下, 通过合理分配资源来更好地满足不同业务对服务质量(QoS)的不同要求. 要真正解决服务质量问题, 需要从应用层到网络层多层联合起来考虑. 在 IP 网中, 采取"分类服务"来解决这个问题. 将业务根据用户签订的服务等级协议(SLA, Service Level Agreement)划分为类(又称为优先级), 为每一类业务分配合适的网络资源, 以保证该类业务的服务质量[1].

图 1 是 ITU 关于 NGN QoS 体系的建议[2], 控制层面需要具备接纳控制、QoS 路由、资源预留功能; 数据层面需要采取的措施包括: 缓存管理、拥塞控制、排队与调度、分组标记、流量均衡、流量管理、流量分类. 网络节点的资源分配可以保证对网络资源有效利用的同时为业务提供更好的服务质量. 节点资源主要包括存储资源(缓存大小)、链路带宽和计算能力(CPU 的处理能力). 由于计算能力通常由操作系统、网络处理器进行控制, 所以在数据层面中, 资源分配通常指的是对缓存

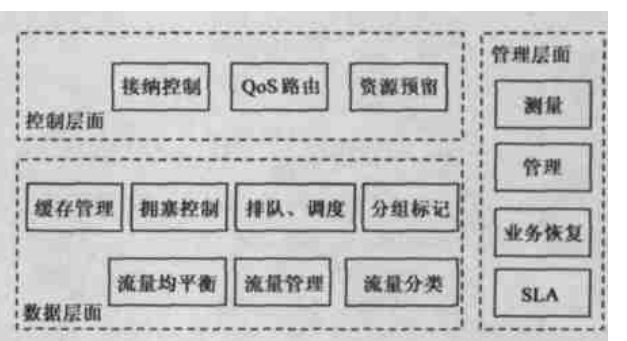

图 1 NGN 的 QoS 体系模型

大小和链路带宽的分配. 资源分配策略主要包括: 缓存管理机制、排队规则与队列调度算法, 它们之间是相互关联的, 不同的队列调度算法需要与不同的缓存管理、排队规则相配合.

本文的研究针对于多业务融合网络环境中节点数据层面的资源分配策略.

2 研究背景

如图 2 所示, 来自输入端口的分组, 在网络处理器中被切分为定长的信元, 根据目的线卡等信息在输入缓存中排队, 等

收稿日期: 2004-09-28; 修回日期: 2004-10-28
基金项目: 美国 Intel 研究院资助项目(No. 14765)

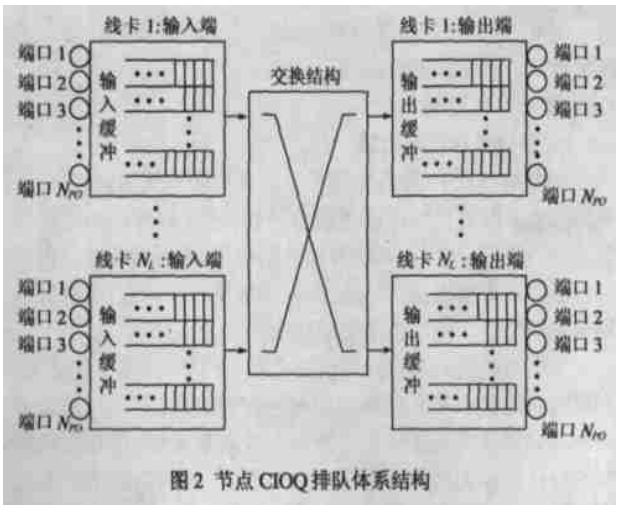

图 2 节点 CIOQ 排队体系结构

待调度算法调度后输出到相应的输出线卡上, 将信元重组成分组后, 在输出缓存中根据目的端口等信息进行排队, 经调度算法调度后从相应输出端口输出. 这样, 整个系统就以一个固定的时间长度向前推进, 称之为时隙(cell time). N_L、N_P 分别为节点的线卡和端口个数.

组播是指来自一个输入端口的一个数据分组需要传送到多个目的输出端口. 组播比例是组播流量占所有网络流量(即单播流量和组播流量的总和) 的比例.

输入负载定义为线卡的输入速率与线速的比值. 信元时延指该信元最后一位到达网络节点到信元第一位离开的时间. 信元转发速率是指在特定的输入负载下, 按照目的地址成功转发信元的速率. 转发速率不明显考虑信元的丢失, 反映所测试输入负载下节点的转发速率. 为了方便表示, 在系统仿真中, 定义" 数据转发率" 为转发速率与输入负载的比值.

3 问题的提出

在网络节点拥塞或者发生输入、输出端口竞争时, 通常采用队列技术, 使得分组按一定的排队策略暂时缓存到队列中, 然后再按相应的调度策略把分组从队列中取出并发送出去. 为了给不同的业务提供不同的服务质量(QoS) 保证, 通常把不同的业务放到不同的队列里, 再利用队列调度算法对所有的业务队列进行调度. 理想的队列调度算法应该不仅能够给某些业务提供带宽和时延等方面的保证, 还能够隔离恶意业务流, 提供业务之间的公平性保证[3]. 根据所作用的排队方式不同, 队列调度算法可以被分为两大类: 输入队列调度算法[4,5] 和输出队列调度算法[6]. 本文所涉及到的队列调度算法, 如无特别说明, 都是指输出队列调度算法.

在实践中得到广泛应用的调度算法大致可以分为三类: 优先级调度算法; 分组公平排队(Packet Fair Queueing, PFQ) 算法[7], 包括加权公平排队算法(Weighted Fair Queueing, WFQ)[8,9], WF^2Q[10], WF^2Q+[11]; 基于轮询(Round Robin, RR) 的调度算法, 如加权轮询 WRR[12]. 上述三类算法均为不分层的调度算法. 它们在进行优先级区分服务时存在这样的问题: 为了简化分析, 让我们来考虑一个 2 张线卡, 每张线卡 1 个物理端口, 2 个优先级的系统. 调度权值设定为: 高优先级占 90% 的带宽, 低优先级占 10% 的带宽. 考虑一个极端情况: 到输出端口 1 的流量包含全部两个优先级, 到输出端口 2 的流量仅包含低优先级, 即优先级在端口间的分布不均衡. 若所有队列统一用 WFQ 进行调度, 则有 20/11 的流量到端口 1, 2/11 的流量到端口 2, 这样端口 2 的利用率仅为 2/11, 系统的数据转发率为 13/22 ≈ 59%. 可见, 优先级在输出端口间分布的不均衡导致数据转发率下降.

在一张线卡支持多个物理端口、多个优先级的情况下, 任一时刻到某个端口的输入流量的组成多种多样, 为了充分利用每个物理端口的输出带宽, 需要在任意输入流特性组合的条件下, 最大化系统数据转发率, 同时保持可接受的平均时延性能.

分层调度算法通过把需要调度的队列分成若干个组, 先在组之间进行加权调度, 再在被调度到的组内部进行调度, 组内又可以进一步划分组, …. 这样就实现了按层次带宽分配, 首先忽略组的内部组成, 在最顶层的组之间分配带宽, 然后在组内进行细化的带宽分配. 这样, 通过按输出端口分组, 组内再按优先级进行 WFQ 调度, 就解决了由于优先级在输出端口间分布不均衡而导致数据转发率的下降问题.

文献[11, 13] 分别提出了 H-GPS、H-PFQ、H-FSC 分层调度算法, 但由于实现复杂度过高, 在实际系统中没有应用.

4 核心传输网络节点资源分配策略

4.1 带宽分配策略

4.1.1 具有混合算法结构的分层调度算法

具有混合算法结构的分层调度算法所采用的排队和调度的层次结构为如图 3 所示的树状结构. 在分层算法中, 只有叶子节点对应实际的物理队列, 其余的层次节点对应一个逻辑队列. 每一个节点是它所有子节点的调度器. 根节点中存放当前可发送信元的指针. 根节点有两个子树: 左子树为绝对优先级队列调度器, 对应于节点中需要传递的反压控制等内部控制信息; 右子树为普通优先级队列调度器, 对应于来自输入线

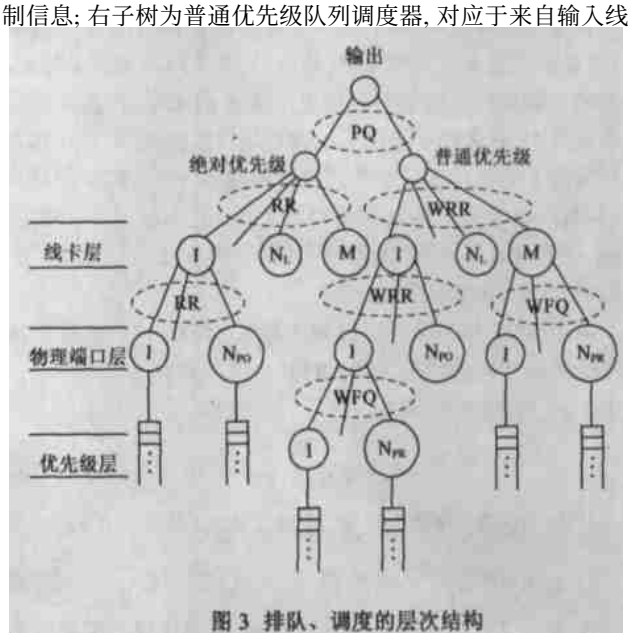

图 3 排队、调度的层次结构

卡的网络流量. 左、右子树之间采用优先级调度, 即只要左子树根节点非空, 就调度左子树, 否则调度右子树.

左子树的根下分为线卡层、物理端口层. 线卡层中有 N_L + 1 个节点, 其中包括 N_L 个与输出线卡对应的逻辑队列和一个绝对优先级组播队列, 队列之间采用 RR 进行调度; 对应于每个输出线卡节点, 有 N_{PO} 个与该线卡的每个输出物理端口对应的队列, 它们之间采用 RR 进行调度.

右子树的根下首先分为线卡层, 线卡层中队列结构与左子树类似, 有 N_L 个单播队列节点与一个组播队列节点, N_L 个单播节点和组播节点间采用 WRR 调度. 组播信元按照其优先级排 N_{PR} 个队, 采用 WFQ 算法进行调度; 线卡层中的每一个单播节点都包括 N_{PO} 个物理端口层的子节点, 采用 WRR 调度算法; 每一个物理节点下, 又包括 N_{IR} 个优先级节点, 采用 WFQ 算法进行调度.

4.1.2 分层调度算法的性能分析

参考文献[11]中指出分层调度算法具有这样的时延特性: 每一个队列的延迟上界等于该队列的父节点调度算法的时延上界加上从父节点到根的每级调度算法的最坏情况公平指数(WFI).

$$L_i = L_{i's\,Parent\,Node\,Scheduling} + \sum_{k=i's\,parent\,nodes}^{Root} WFI_{Node\,k's\,Algorithm} \quad (1)$$

按这个性质进行具有混合结构的分层算法节点上的调度算法选择:

(1) 优先级之间的调度算法

优先级队列是分层算法中的叶子节点, 因此优先级之间的调度算法直接作用到叶子节点上. 在(1)的时延上界表达式中, 它是第一项 $L_{i's\,Parent\,Node\,Scheduling}$. 这一层的调度需要最小化算法本身的延迟上界; 且优先级之间的带宽配置可能因应用的不同而变化, 考虑到这些因素, 不使用延迟上界与权值分布相关的 WRR 等简单算法, 而使用 WFQ, 它有好的时延上界.

(2) 物理端口之间的调度算法

该层调度算法的 WFI 对分层算法的整体时延上界有影响. 考虑实现复杂度低的 WRR 算法, 它的 WFI 与帧长 F 相关, 队列的个数和权值的分布直接决定了 F 的大小. 在高速网络节点设计中, 通常同一张线卡的物理端口线速相同, 即可以看做特定线卡的物理端口层队列权值相同, 从而将 WRR 简化为 RR, F 由一张线卡上的物理端口数目(一般不超过 16)决定. 显然, 物理端口层采用 RR 算法, 具有小的 WFI.

(3) 线卡层的调度算法

采用 WRR 作为线卡层的调度算法. 不失一般性, 考虑所有线卡线速相同的情况, 这时 WRR 中的权值由多播比例 r_m、线卡个数 N_L 确定, 关系为

$$r_m = \frac{W_m}{N_L W_u + W_m} \quad (2)$$

其中 W_m, W_u 分别为组播、单播在 WRR 算法中的权重.

以 $r_m = 40\%$, $N_L = 16$ 为例, 带入式(2)中得 $\frac{W_u}{W_m} = \frac{3}{2}$, 取 $W_u = 3$, $W_m = 2$ 可得 $F = 3\times 16 + 2 = 50$. 由于权值分配简单和队列数目固定, 所以此情况下 WRR 具有小的 WFI.

因此, 上文设计的混合算法结构分层调度算法在给定系统参数后, 能够得到时延上界.

4.2 缓存分配策略

4.2.1 缓存管理机制概述

在网络节点中, 缓存管理机制用来改善系统的丢包率, 在不同的业务间提供缓存占用的公平性. 缓存管理机制对缓存资源加以控制, 与队列调度算法共同作用, 成为网络节点提供区分服务的重要手段. 与分层队列调度算法相对应, 缓存管理机制必须按层次提供公平的缓存占用.

本文所涉及的缓存管理机制是基于门限的, 基本含义为: 如果当前缓存中某个队列的信元超过特定的门限值, 那么未来时隙到达该队列的信元将被阻塞或者被丢弃. 缓存管理机制的设计目标就是得到这个合适的特定门限: 门限值必须设计的足够大, 以便对队列提供保证的服务; 门限值又必须设计的足够小, 以便尽早的检测到该队列已经过多占用了缓存.

应用于网络节点中的缓存管理机制可以分为静态门限策略[14,15](Static Threshold, ST)、动态门限策略(Dynamic Threshold, DT)[16]、Push Out 策略[17,18](PO). 同时为了满足不同业务类型的需求, 三种门限策略还可以跟优先级结合, 形成多优先级的缓存管理机制. ST 适应网络流量的能力差, PO 实现过于复杂, 因此 DT 在实际中应用广泛.

动态门限策略在系统运行的过程中, 根据缓存中剩余空间的大小来确定队列的门限值. 在时刻 t, $T(t)$ 为该时刻的门限值, $L(t)$ 为该时刻所有端口的队列长度和, 为该时刻 i 端口的队列, 则 $T(t)$ 可由式(3)确定:

$$T(t) = f(M - L(t)) = f\left(M - \sum_{i=1}^{N} L_i(t)\right) \quad (3)$$

如果 $L_i(t) \geqslant T(t)$, 则到达该队列的信元将被阻塞或者丢弃.

为了算法易于实现, 通常取 $f(x) = \alpha * x$, 则有

$$T(t) = \alpha(M - L(t)) = \alpha\left(M - \sum_{i=1}^{N} L_i(t)\right) \quad (4)$$

动态门限策略具有很好的自适应能力, 当端口负载变化时, 门限会随之变化. 例如, 当某端口负载增加时, 其队列长度增加, 因此, $T(t)$ 减小, 很快达到门限; 而经过一段时间的阻塞后, 由于有信元输出, 队列变短, $T(t)$ 增大, 该端口的信元又可以得到服务.

4.2.2 分层的缓存管理机制

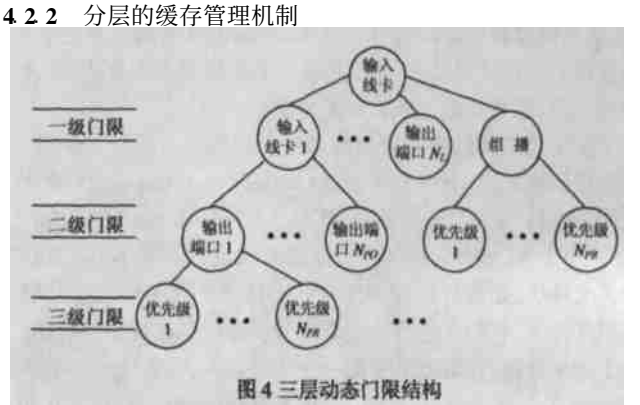

图4 三层动态门限结构

根据排队、调度规则, 缓存中的队列可以用 $Q(i,j,k)$ 表示, 其中 $i=0,1,\cdots,N_L$ 是信元目的线卡的编号, $j=0,1,\cdots,N_{PO}$ 是信元目的物理端口的编号, $k=0,1,\cdots,N_{PR}$ 是信元优先级的编号. 三层动态门限值的设定规则如下:

(1) 第一层门限值 $T_i(t)$ 是针对按目的线卡排列的队列而设计, 假定线卡之间的共享参数为 α_i, $i=0,1,\cdots,N_L$, B 为缓存大小(按信元数目计), 则有

$$T_i(t)=\alpha_i^*(B-L(t)) \tag{5}$$

(2) 第二层门限值 $T_{i,j}(t)$ 针对属于特定目的线卡的物理端口而设计, 假定目的线卡 i 物理端口 j 的共享参数为 $\phi_{i,j}$, 则有

$$T_{i,j}(t)=\lambda_{i,j}^*T_i(t) \tag{6}$$

(3) 第三层门限值 $T_{i,j,k}(t)$ 针对特定目的线卡的特定物理端口上不同的优先级而设计, 假定优先级 k 的共享参数为 λ_k, 则有

$$T_{i,j,k}(t)=\lambda_k^*T_{i,j}(t) \tag{7}$$

综合式(5) ~ (7) 得

$$T_{i,j,k}(t)=\lambda_t^*\phi_{i,j}^*\alpha_i^*(B-L(t)) \tag{8}$$

5　网络节点性能仿真与结果分析

5.1　流量模型描述

流量模型是指到达输入端口的网络流量模型, 通常由两个随机过程来刻画: 一个用来表示相继分组到达间隔的统计规律, 称为"到达模型"; 另一个表示分组目的端口的分配规律, 称为"目的端口分布模型". 如无特别说明, 本文以下仿真结果均基于如下流量模型: 到达模型为截尾重尾分布模型[19] (Truncated Power Tail, TPT), 信元的到达时间具有自相似特性; 每个输入端口所生成的流量包括 8 个优先级, 各个优先级所占的比例为[1/8、1/8、1/8、1/8、1/8、1/8、1/8、1/8], 分层调度算法中 WFQ 的权值设定相应为[0.16、0.15、0.14、0.13、0.12、0.11、0.10、0.09]; 为了造成输出端口间优先级分布的不均衡, 所有输入端口的目的端口分布模型采用文献[20]所述的热点分布, 目的端口集中在几个特定的输出端口上. 假定的系统参数为: 16 张线卡, 每张线卡上有 16 个端口; 仿真时间为 1 秒.

5.2　动态分层缓存管理机制仿真分析

为了对节点的性能进行仿真分析, 首先需要确定式(8)中的动态门限参数. 我们做了如下合理简化:

(1) 由于实际的系统中, 所有线卡具有相同的线速, 考虑到线卡之间的公平性, 可以认为 $\alpha_i=\alpha$, $i=1,\cdots,16$ 则有

$$T_i(t)=\alpha^*(B-L(t)) \tag{9}$$

(2) 假定所有线卡具有相同数目、相同线速的物理端口, 那么 $\phi_{i,j}=\phi$, 则有

$$T_{i,j}(t)=\phi^*\alpha^*(B-L(t)) \tag{10}$$

(3) 优先级层的共享参数与优先级所分配的带宽资源成比例, 即 $T_{i,j,k}(k)=\lambda_k^*\phi^*\alpha^*(B-L(t))$ 中, $\lambda_k=2r_k$, $k=1,\cdots,8$. 依据分层算法中的带宽分配一[0.16、0.15、0.14、0.13、0.12、0.11、0.10、0.09], [λ_k], $k=0,1,\cdots,7$ 的相应取值为[0.32、0.3、0.28、0.26、0.24、0.22、0.2、0.18].

为了便于硬件实现, 将 α、ϕ 均取为 2 的倍数, 在实际仿真中参考[21], 取 $\alpha=[1/16,1/8,1/4,1/2,1,2,4,8,16]$, $\alpha=1/16$ 表示 16 个线卡之间对当前的剩余缓存完全独占, α 越大线卡之间的共享程度越大; $\phi=[1,1/2,1/4,1/8,1/16]$, $\phi=1/16$ 表示每个线卡的 16 个物理端口之间缓存完全独立, ϕ 越大物理端口之间的共享程度越大. 如图 5、6 分别为目的端口服从热点分布与均匀分布时, 缓存的利用率(纵轴值) 随共享参数 α(横轴值)、ϕ(不同曲线) 的变化情况.

观察图 5、图 6, 我们可以得出如下结论:

- 当固定 ϕ 值时, 缓存利用率随 α 增加而单调增加;
- 当固定 α 值时, 缓存利用率随 ϕ 的增加而单调增加;
- 目的端口为均匀分布比热点分布情况下的缓存利用率低.

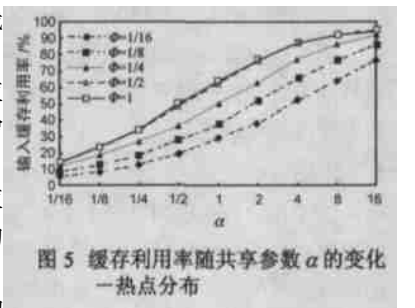

图 5　缓存利用率随共享参数 α 的变化—热点分布

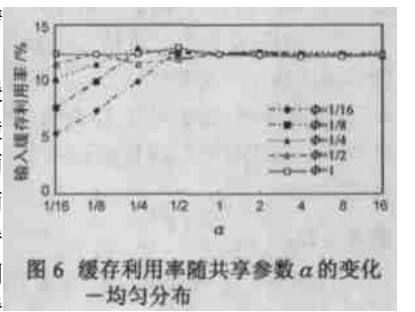

图 6　缓存利用率随共享参数 α 的变化—均匀分布

在突发量大时, 缓存利用率太高, 容易造成某个队列过长, 从而增加了信元的时延; 而缓存利用率太低, 会导致信元丢失率增大. 因此, 取 $\alpha=1$, $\phi=1/2$, 缓存利用率大约在 50% ~ 70%.

5.3　资源分配策略仿真分析

缓存机制确定后, 资源分配策略就是指利用调度算法对网络带宽进行分配. 对于非分层算法, 在输入缓存的不同优先级队列之间采取 WRR 调度算法; 分层调度算法与上文 3.1.2 中描述的具有混合算法结构的分层调度算法相同.

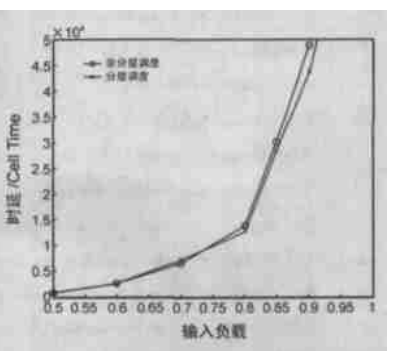

图 7　平均时延随输入负载的变化

从图 7、8 的仿真结果中可以看出, 在输入负载较轻(小于 80%) 时, 两种调度算法的传输延时、数据转发率方面性能差别不大; 在重负载的条件下, 分层算法的平均时延性能略好, 而数据转发率得到了明显的改善. 以输入负载为 1.0 为例来分析系统数据转发率急剧下降的原因: 流量模型每秒钟产生 2.5Gbits 的数据, 16 个流量模型的总流量为 16× 2.5=

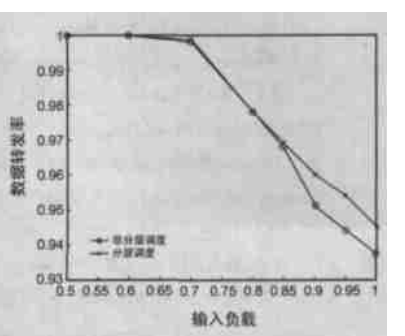

图 8　数据转发率随输入负载的变化

40Gbits,在系统无阻塞的情况下,子端口 1 到 14 的在出口处不会发生阻塞,吞吐量为 100%.在子端口 0 和 15(热点端口),1 秒钟会有 2.5× (1.4246- 1) = 1.0615Gbits 的数据阻塞在系统中,从而造成数据转发率的下降.此条件下,理论上的最大数据转发率为:

$$Throughput_{Max} = (40 - 1.0615 \times 2)/40 = 94.6\% \quad (11)$$

图 6 的仿真结果表明采用分层资源分配策略,可以使网络节点系统在优先级在输出端口间分布不均衡情况下的数据转发率接近理论值.

6 结论

为了在核心传输网络节点数据层面为未来的多业务通信网提供更高的服务质量,本文提出了分层的资源分配策略,包括两方面机制:(1)根据业务、输出线卡、优先级进行排队并经过具有混合排队结构的分层调度算法调度输出;(2)按输出线卡、输出端口、优先级三维信息来分配缓存资源.该策略不但能够解决优先级在端口间分布的不均衡而造成的数据转发率下降,而且简单、易于实现.仿真结果表明在输入负载较重(大于 80%)时,仍然可以保证较高的数据转发率和较好的时延性能.

参考文献:

[1] 蒋林涛. 下一代 IP 网与第五带路由器[J]. 中国数据通信, 2003, 5(5): 5- 7.

[2] 邬贺铨. 下一代互联网和下一代网[J]. 世界电信, 2004, 17(6): 31- 36.

[3] C Semeria. Supporting Differentiated Service Classes: Queue Scheduling Disciplines [EB/OL]. http://www.juniper.net/solutions/literature/white_papers/200020.pdf.

[4] N McKeown. Scheduling Cells in an input-queued switch[D]. USA: PhD Thesis, University of California at Berkeley, May 1995.

[5] N McKeown. Fast Switched Backplane for a Gigabit Switched Router[EB/OL]. http://www.cnaf.infn.it/~ferrari/tfngn/doc/fasts_wp.pdf.

[6] A K Parekh, R G Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case[J]. IEEE/ACM Trans on Networking, June 1993, 1(3): 344- 357.

[7] C Minkenberg, T Engbersen. A combined input and output queued packet-switched system based on PRIZMA switch-on-a-chip technology [J]. IEEE Communication Magazine, Dec. 2000, 38(12): 70- 77.

[8] D C Stephens. Implementing scheduling algorithms in high-speed networks[J]. IEEE Journal on Selected Areas in Communications, June 1999, 17(6): 1145- 1159.

[9] A J Demers, S Keshav, S Shenker. Analysis and simulation of a fair queuing algorithm[A]. in Proc ACM SIGCOMM'1989[C]. Austin, TX, USA: 1989. 1- 12.

[10] J C R Bennett, H Zhang. WF^2Q: worst-case fair weighted fair queueing [A]. INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE [C]. San Francisco, CA, USA: 1996.

[11] J C R Bennett, H Zhang. Hierarchical packet fair queueing algorithms [A]. Proc ACM SIGCOMM'96[C]. California USA: 1996. 143- 156.

[12] M Katevenis, S Sidiropoulos, C Courcoubetis. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip[J]. IEEE Journal on Selected Areas in Communications, 1991, 9(8): 1265- 1279.

[13] I Stiliadis, A Varma. Rate-Proportional Servers: A Design Methodology for Fair Queueing Algorithms[J]. IEEE/ACM Tran Networking, 1998, 6(2): 164- 174.

[14] M Irland. Buffer management in a packet switch[J]. IEEE Transactions on Communications, 1978, 26(3): 328- 337.

[15] F Kamoun, L Kleinrock. Analysis of shared finite storage in a computer network node environment under general traffic conditions[J]. IEEE Transactions on Communications, 1980, 28(7): 992- 1003.

[16] 李钢锁, 徐恪, 吴建平. 面向变长分组的多优先级动态阈值缓存管理算法研究[J]. 电子学报, 2002, 30(8): 1188- 1191.

[17] A K Thareja, A K Agrawala. On the Design of Optimal Policy for Sharing Finite Buffers[J]. IEEE Trans Commun, 1984, 32(6): 737- 740.

[18] S X Wei, E J Coyle, M T Hsiao. An optimal buffer management policy for high performance packet switching[A]. in IEEE GLOBECOM'91 [C]. Pheonix Arizona, USA: IEEE, 1991, 924- 928.

[19] Lipsky L, Hatem J E. Buffer problems in telecommunications systems [A]. 5th International Conference on Telecommunication Systems[C]. Nashville: Tennessee, 1997.

[20] C Kolias, L Kleinrock. The Dual-Banyan (DB) Switch: A high-performance buffered-banyan ATM switch[A]. in the proceedings of the International Conference on Communications (ICC)'97[C]. Montreal, Canada: 1997.

[21] A K Choudhury, E L Hahne. Dynamic queue length thresholds for shared-memory packet switches[J]. IEEE/ACM Transactions on Communications, 1998, 6(2): 130- 40.

作者简介:

宋美娜 女, 1974 年生于山东威海, 讲师, 北京邮电大学, 主要研究方向为网络交换技术、下一代互联网. E-mail: mnsong@bupt.edu.cn.

宋俊德 男, 1935 生于河北沧州, 教授, 北京邮电大学, 博士生导师, 主要研究方向为 CAD、CTI 以及移动通信.

张 益 男, 1977 生于湖北黄岗, 工程师, 北京控制工程研究所, 主要研究方向为软件工程、下一代互联网.