# 一种综合缓存管理和主动队列管理的区分服务节点机制

李锁钢, 吴建平, 徐 恪
(清华大学计算机科学与技术系, 北京 100084)

摘 要: 随着网络和应用的飞速发展, Internet 不仅要提供尽力转发(Best Effort)的服务, 还要支持各种传输类型和多个优先级的 QoS 服务. 目前普遍认为区分服务体系机构是很有前途的提供 QoS 保证的 Internet 框架, 而网络节点机制是其关键技术之一. 我们提出了一种在网络节点上实现的区分服务机制 comBAQ(combining Buffer Management and Active Queue Management), 它综合了适当的缓存管理和主动队列管理方法. 我们详细介绍了它的服务框架和分组处理判决算法, 实验模拟结果显示, 它能满足我们提出来的五个设计目标, 可以在网络节点上实现多个丢失优先级的区分服务: 为需要可靠传输的多媒体应用提供"有保证"服务, 为传统 TCP 传输提供"无保证"服务.

关键词: 区分服务; 缓存管理; 动态阈值; 主动队列管理
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2005) 05-0847-05

# A Node Mechanism of Differentiated Services Combining Buffer Management and Active Queue Management

LI Suo-gang, WU Jiarr-ping, XU Ke
(*Department of Computer Science, Tsinghua University, Beijing* 100084, *China*)

**Abstract:** With the rapid development of networks and applications, Internet is required to not only provide the Best Effort service, but also supply services of various traffic classes and multiple priorities. DiffServ architecture is considered to be the promising Internet framework meeting QoS requirements. One of crucial elements in DiffServ is the network node scheme. In this paper, we propose a novel DiffServ node mechanism called combining Buffer Management and Active Queue Management (comBAQ). We detail comBAQ service framework and its packet judgment process. Simulation results demonstrate that comBAQ is capable of simultaneously achieving our five targets and provides service differentiation in terms of packet drop priority: the More Guaranteed (MG) service for reliable traffics and the Less Guaranteed (LG) service for traditional application flows.

**Key words:** DiffServ; buffer management; dynamic thresholds; active queue management

## 1 引言

现在的 Internet 只能提供"尽力转发(Best Effort, BE)"的服务, 而许多新的应用如流媒体等需要比 BE 更好的转发服务, Internet 正逐渐变得不能适应这些新的需求. Internet 工程任务组(IETF)为下一代互联网提出了区分服务框架[1,2]. 通过区分服务的体系结构, 服务提供商能够根据不同的性能为用户提供了一系列网络服务. 如果设计合理, 区分服务体系结构能支持很好的灵活性和扩展性, 同时能满足多媒体流应用的服务需求. IETF 区分服务工作组还详细说明了"确保转发(Assured Forwarding, AF)"类型的逐跳行为(Per Hop Behavior, PHB)[3].

在本文中, 我们提出一种在网络节点上提供区分服务的机制, 它综合了缓存管理和主动队列管理方法, 我们称它为 comBAQ(combining Buffer Management and Active Queue Management). comBAQ 能够为 IP 分组提供不同层次的丢失率保证, 既可以满足可靠传输流(如多媒体流)的传输要求, 也可以支持传统数据(如 TCP 流)应用的传输. 根据网络需求, comBAQ 必须同时实现下面五个目标:

(1) 网络的核心节点不用维护每流的任何状态信息;

(2) 支持丢失率的多优先级传输, 可以提供可靠的传输服务;

(3) 能对 TCP 传输实现拥塞控制, 并随机丢弃分组避免全程同步(Global Synchronization)问题;

(4) 在实现节点上, 每流的分组顺序不会更改;

(5) 动态适应网络变化, 实现尽量简单.

目标(1)和(5)是为了提高方法的可扩展性, 目标(2)和

收稿日期: 2004-05-17; 修回日期: 2004-07-31
基金项目: 国家自然科学基金(No. 60303006, No. 60203025); 973 基金(No. 2003CB314801)

(3)是实现区分服务的必要条件,而目标(4)是遵照 IETF 的规定,不能对属于同一个流的分组重新排序[4]. 分组重排序会使实时传输产生抖动,使TCP 传输的性能下降. 我们选择在同一个队列中对分组进行存储的服务机制.

comBAQ 区分服务节点机制综合了缓存管理方法和主动队列管理方法. 这两种方法都是网络节点(如路由器)的重要技术,它们都是在分组进入网络节点排队时管理节点内存空间的方式. 我们选用了适当的缓存管理和主动队列管理方法,模拟实验说明 comBAQ 能够实现上面的五个目标.

本文组织如下: 先介绍研究问题的背景,简述了 DiffServ 体系结构的特点和选择适当的缓存管理与主动队列管理方法的原因;然后根据上面的五个目标,详细介绍了 comBAQ 节点机制的服务类型和优先级定义,介绍了它的分组处理流程和在路由器上的实现细节;接下来我们对 comBAQ 机制的性能进行了实验模拟. 最后我们做出结论,并讨论了进一步的研究方向.

## 2 背景介绍

### 2.1 区分服务

在 IP 网络研究 QoS 问题的过程中,IETF 首先提出集成服务(IntServ)体系结构. 但 IntServ 是基于每流(per- flow)的、状态相关的,在复杂、大规模网络中很难实现. 而且 IntServ 具有某种面向连接的特性,这与 IP 网络面向无连接的特性相悖. 然后 IETF 提出了区分服务(DiffServ)体系结构. 在 DiffServ 域内的网络节点只需进行简单的调度转发,而流状态信息的保存与流监控机制等只在边界节点实现,保证了核心节点状态无关. DiffServ 的服务对象是流聚集(aggregate)而非单流,单流信息只在网络边界保存和处理. DiffServ 框架将网络核心节点的复杂性转移到边界节点,实现了很好的扩展性.

### 2.2 缓存管理

现有的缓存管理算法中大部分都是基于输出端口排队的、共享缓存结构. 这种结构能保持网络节点中分组的顺序,实现目标(4). 当有足够的缓存空间时,到达的分组可能进入缓存;否则就要丢弃刚到达的分组,或者丢弃缓存中的分组以腾出空间接收刚到达的分组.

缓存管理分为静态策略、动态策略和 Pushout 策略,还有可提供不同级别服务的多优先级策略[7]. 静态策略实现简单,也有扩展支持多优先级传输的方法,但是它不能反映网络的变化情况,不符合我们的目标(5). 动态策略是启发式的自适应控制系统,典型方法是动态阈值法(Dynamic Threshold, DT),能根据网络情况动态修改每个队列在缓存中的长度阈值,适应网络传输的变化. 但是为了实现目标(2),我们需要支持多优先级的缓存管理策略.

在具有多优先级特性的动态阈值策略中,多优先级最佳动态阈值方法[5]可以向不同类型的网络流提供不同优先级的 QoS 保证,缓存利用率较高,控制参数容易设置. Pushout 策略通过丢弃缓存中的某些分组允许刚到达的分组进入缓存[8],它与多优先级最佳动态阈值方法相结合可以共同实现目标(2)和目标(5),为要求可靠传输的应用提供"有保证"的服务.

### 2.3 主动队列管理

主动队列管理(Active Queue Management, AQM)在拥塞控制研究中是一个很重要的主题. AQM 的传统技术是队尾丢弃机制. 但队尾丢弃会引起TCP 全程同步问题,而且不能提供服务区分. RED[9](Random Early Detection)是著名的 AQM 算法. 它在拥塞发生时随机丢弃分组,避免TCP 的全程同步问题. 但 RED 的计算周期过短,参数设置也不够理想. 同时,RED 只针对TCP 传输,不能为多媒体应用流提供服务区分,这不符合我们的目标(2). RED 的扩展方法 RIO(RED In and Out)和 WRED(Weighted RED)也不能满足我们的要求.

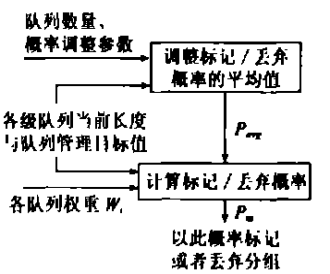

图 1 WSAP 算法流程

加权简单自适应比例控制算法[6](Weighted Simple Adaptive Proportional, WSAP)是一种自适应的 AQM 算法. WSAP 算法流程如图 1 所示,它不像 RED 那样计算频繁,对每个优先级设置参数少,扩展性好. 因此,我们选择 WSAP 对可重传数据流进行优先级管理和拥塞控制.

## 3 comBAQ 服务框架

### 3.1 分组服务类型与优先级定义

comBAQ 节点机制设置两类服务: 有保证的服务和无保证的服务,分别记为 MG(More Guaranteed)和 LG(Less Guaranteed). 需要MG 服务的分组是要保证可靠传输的分组,比如多媒体流中影响接收端的视频质量的重要信息或者控制信息分组;需要LG 服务的分组不要求可靠传输的分组,比如多媒体流中对应用层视频质量不重要的分组,或者是支持重传的分组流(传统的TCP 分组).

对于每种 MG 和 LG 服务,按照丢失率的高低划分优先级. MG 服务和 LG 服务的丢失优先级分别为 $MG[i]$ 和 $LG[j]$, $0< i\leqslant N$, $0< j\leqslant M$. 相应地,各个服务优先级的丢失率为 $Loss(MG[i])$ 和 $Loss(LG[j])$. $i$ 或者 $j$ 越小,丢失优先级越高,即丢失率越低. 按照目标(2),comBAQ 需要实现:

(1) $Loss(MG[i]) \leqslant Loss(LG[j])$, 对任意的 $0< i\leqslant N$, $0< j\leqslant M$;

(2) $Loss(MG[i]) < Loss(MG[j])$, 如果 $i< j$, $0< i\leqslant N$, $0< j\leqslant M$;

(3) $Loss(LG[i]) < Loss(LG[j])$, 如果 $i< j$, $0< i\leqslant N$, $0< j\leqslant M$.

可以在 IP 分组头部指定位置设置 DSCP(区分服务标记值)与 $MG[i]$ 和 $LG[j]$ 相匹配,对于 IPv4 是分组头部的 TOS 域,而对于 IPv6 分组是

表 1 comBAQ 机制的优先级定义与 AF PHB 对应关系

| 服务类型 \ 丢失优先级 | 1 | 2 | 3 |
|---|---|---|---|
| 有保证服装 MG | 001-010 (AF11) | 001-100 (AF12) | 001-110 (AF13) |
| 无保证服装 LG | 001-010 (AF21) | 010-100 (AF22) | 010-110 (AF23) |

头部的Flow Label 域,如图 2 所示. 特别地, 我们根据 AF PHB 的定义举例说明: 定义两种 AF PHB 类型的服务 MG 和 LG, 它们对应的 DSCP 值参见表 1.

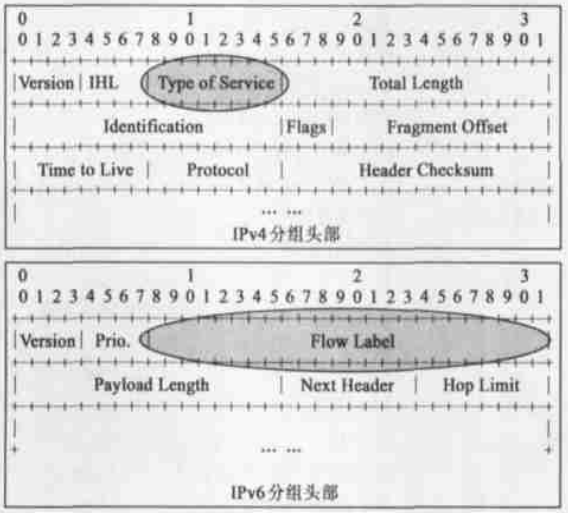


图 2　DSCP 在 IPv4/IPv6 分组头部中的位置

在一个节点上发生拥塞时, MG 服务的分组应该具有比 LG 分组更低的丢失率. 在缓存满时, 如果 LG 分组存在于缓存中, 刚到达的MG 分组就不能被丢弃, LG 分组应该腾出足够的空间以接收 MG 分组. 在享有相同类型服务的分组中高优先级分组的丢失率要低于低优先级分组的丢失率. 比如在 MG 分组中, MG[ i] 的分组丢失率应低于 MG[ j] 分组丢失率( $i < j$). 在缓存满时 MG 分组的服务不会受到 LG 分组的干扰. 因此, MG 服务可以在拥塞时为分组提供最大可能的可靠保证.

### 3.2　节点机制

我们假设端节点拥有关于应用的完整知识, 端节点可以将分组标记为 MG[ i] 或者 LG[ j] 服务类型. 我们还假设所有的边界节点具有传输调节功能( 如标记、整型和管制). 在区分服务域中的核心节点不会把来不同的传输流分成不同队列. 下面, 我们首先给出 comBAQ 机制的算法流程, 然后说明分组进入缓存数据结构和组织结构.

**3.2.1**　分组接收判决算法　设总的缓存 $B_{ALL}$, $B$(MG[ i]) 表示优先级为 $i$ 的MG 分组占用的缓存数量, $B$(MG) 表示MG 类型的所有优先级分组占用的缓存数量, $B$(LG) 表示 LG 类型的所有优先级分组占用的缓存数量, $B_{FREE} = B_{ALL} - B$(MG) $- B$(LG) 是未用缓存数. 设 $T_{MG}$ 是优先级为 $i$ 的 MG 型分组可占用缓存的阈值. 表 2 显示了 comBAQ 机制在一个分组到达时根据缓存当前状态判断是否接收该分组.

表 2　comBAQ 机制根据缓存状态决定是否接收分组

| 分组 P 到达时的状态 ( 假设其优先级为 $i$) | 状态说明 | 接收/丢弃 | |
|---|---|---|---|
| | | P 的类型为 MG | P 的类型为 LG |
| $B_{FREE} = B_{ALL}$ | 缓存中没有分组排队 | 直接接收 P | 由 *WSAP* 决定是否接收 P |
| $B_{FREE} > 0$ && $B$(MG[ i]) $< Th$(MG[ i]) | 有空闲缓存, 并且 MG[ i] 型分组占用缓存未超过阈值 | 直接接收 P | |
| $B_{FREE} = 0$ && $B$(MG[ i]) $< Th$(MG[ i]) | 缓存满, 并且 MG[ i] 型分组占用缓存未超过阈值. 即缓存中存在 LG 型分组 | 将缓存中的 LG 型分组 Pushout, 腾出足够空间接收 P | |
| $B$(MG[ i]) $> Th$(MG[ i]) | MG[ i] 型分组占用缓存超过阈值 | 将 P 与所有其他分组一起根据" 多优先级最佳动态阈值算法" 决定是否接收 | |

当一个MG 分组到达节点时, comBAQ 机制会努力接收它进入缓存, 必要时将缓存中的 LG 分组剔出来. 另一方面, 当一个LG 分组到达时, 只有当缓存有足够的空间时才允许它进入, 并使用 WSAP 算法以概率决定是否接收它. 因此 comBAQ 机制实现了按照服务和优先级的区分服务即目标(2), 同时避免了 *TCP* 传输的全程同步问题即目标(3).

**3.2.2**　缓存队列组织结构　在 comBAQ 的实现中, 需要维护两个缓存的统计变量 $B$(LG) 和 $B_{FREE}$( 都以字节计). 其中, $B$(LG) 用于跟踪所有被 LG 分组占用的缓存数量; $B_{FREE}$ 是空闲缓存空间的数量. 还要维护两个双向链表: 所有分组链表 $L_{ALL}$ 和 LG 分组链表 $L_{LG}$, 以及指向它们的队头和队尾的四个指针: $PH_{ALL}$、$PH_{LG}$ 和 $PT_{ALL}$、$PT_{LG}$. 链表 $L_{LG}$ 是嵌入在 $L_{ALL}$ 内部, 如图 3 所示.

我们维护下面的数据结构: 缓存中的每个数据单元包括一个物理 IP 分组和四个指针, 其中两个指针用于链接链表 $L_{ALL}$, 两个指针用于链接链表 $L_{LG}$. 链表 $L_{ALL}$ 连接缓存中的所有分组( 包括 MG 服务和 LG 服务分组). 当新到达的分组加入到队列尾或者输出链路传送队列头的分组时, 它将被更新; 链表 $L_{LG}$ 连接嵌入到 $L_{ALL}$ 链表的 LG 服务分组的链表. 当新到达的 LG 分组加入到队列尾, 或者一个 LG 分组被输出或被 Pushout 机制丢弃时, $L_{LG}$ 将被更新.

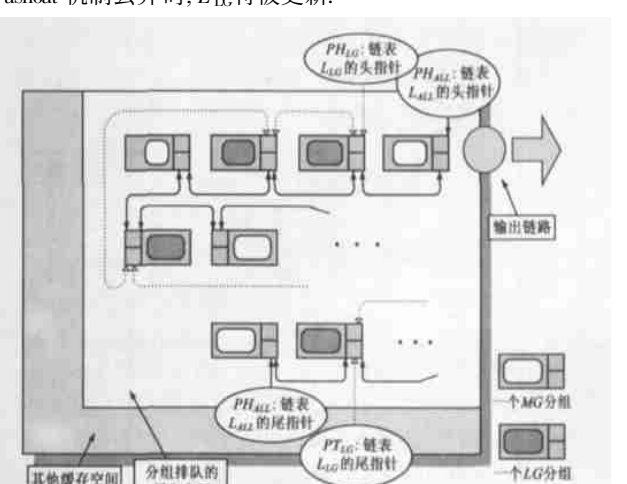


图 3　comBAQ 的队列在缓存空间中的组织结构

$L_{ALL}$和 $L_{LG}$都是双向链表. $L_{LG}$有一个头部指针, 将要丢弃的分组就是它指向的 LG 分组. 这样便于定位待删除 LG 分组. 使用单向链表也可以实现这些操作, 但是要花费更多的处理时间.

## 4 实验模拟结果

### 4.1 实验场景

我们使用 NS-2 模拟器[10]对 comBAQ 算法进行了实验模拟. 实验建立的链路拓扑参见图 4. 共有四个源节点, 四个目的节点, 在源节点上创建应用(突发流应用或者 FTP 传输), 每个源节点上的应用发出不同优先级的分组. 在实验中每个源节点上应用的数量是可变的.

实验设置参见表 3, 在表中 TCP 版本和 UDP 传输模型都是使用 ns-2 提供的 Agent. 给 TCP 分组加标记的方式是直接丢弃. 实验中缓存大小 Ω、UDP 源的速率 $r_{ON}$和 MG 型分组占用缓存的阈值 $T_{MG}$都可以根据实验设置调整. 相应的, WSAP 的队列目标长度 $Q_{TAR}=T_{MG}*\Omega$, 多优先级最佳 DT 算法的可用缓存即 $0.9(1-T_{MG})*\Omega$.

表 3 comBAQ 机制的实验参数

| | | | |
|---|---|---|---|
| 链路 | 端节点到网络节点 | 带宽 | 64Mbps |
| | | 延迟 | 40ms |
| | 网络节点之间 | 带宽 | 128Mbps |
| | | 延迟 | 5ms |
| 网络节点 | comBAQ | 缓存大小 | Ω |
| | | MG 分组占用缓存阈值 | $T_{MG}$ |
| | WSAP | 队列平均目标长度比例 | 0.5 |
| | | 标记概率初值 | 0.1 |
| | | 丢失率区分参数比 $w_{LG1}$: $w_{LG2}$ | 1:4 |
| | 多优先级最佳 *DT* | 可用缓存比例 | 0.9 |
| | | 队列阈值初值 | 1000 字节 |
| 端节点 | TCP(ns-2) | TCP 版本 | Reno |
| | | 标记分组方式 | 直接丢弃 |
| | | 窗口大小 | 150 |
| | | 分组大小 | 1000 字节 |
| | UDP(ns-2) | 传输模型 | Pareto |
| | | 分组大小 | 536 字节 |
| | | ON 状态平均时间 $E(T_{ON})$ | 100ms |
| | | OFF 状态平均时间 $E(T_{OFF})$ | 150ms |
| | | ON 状态分组发射速率 | $r_{ON}$ |
| | | Pareto 的 shape 参数 | 1.5 |

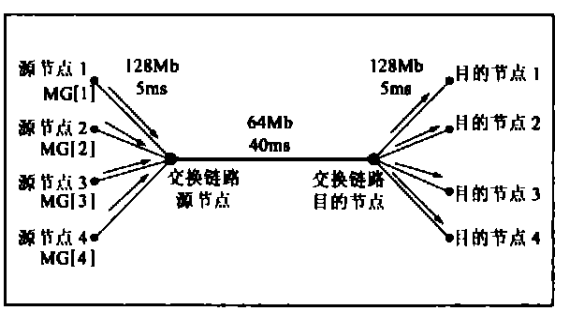


图 4 实验链路拓扑图

### 4.2 实验结果

在实验一中, Ω= 40kB, $r_{ON}$= 100kbps, $T_{MG}$= 0.5, 每个优先级 40 个应用. 实验结果参见图 5. 在实验二中, 设置 $T_{MG}$= 0.14, 其余参数同上. 实验结果参见图 6. 由于 MG 分组的可使用缓存数量减少, 所以出现了丢失. MG1 丢失率要小于 MG2. 在实验三中, Ω= 400kB, $r_{ON}$= 200kbps, $T_{MG}$= 0.7, 每个优先级 80 个应用. 实验结果参见图 7.

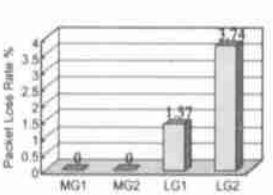


图 5 comBAQ 实验一中各优先级的分组丢失率

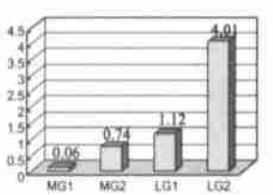


图 6 comBAQ 实验二中各优先级的分组丢失率

从以上三个实验可以看出, comBAQ 机制实现了丢失率区分, 在节点缓存充足情况下保证 MG 类型的分组不丢弃, 在缓存数量不足以应付链路负载情况下, 丢失率低于 LG; 各优先级丢失率之间关系是 Loss(MG1) < Loss(MG2) < Loss(LG1) < Loss(LG2), 其中, Loss(MG1) < Loss(MG2)是由多优先级最佳动态阈值算法保证的, Loss(LG1) < Loss(LG2)是通过 WSAP 算法保证的, Loss(MG) < Loss(LG)则是 Pushout 策略的贡献.

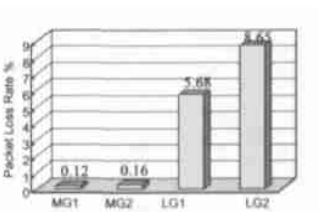


图 7 comBAQ 实验三中各优先级的分组丢失率

图 8 是在实验三的条件下, 一段时间 comBAQ 机制处理分组的曲线. 可以看出, 当单位时间的 LG 分组到达数量增加到最大时, 缓存被大量使用, MG 分组出现丢失最大值, 同时缓存中的 LG 分组被剔出(pushout). 这说明我们的方法保证了 MG 分组优先 LG 占用缓存, 减小 MG 的分组丢失率.

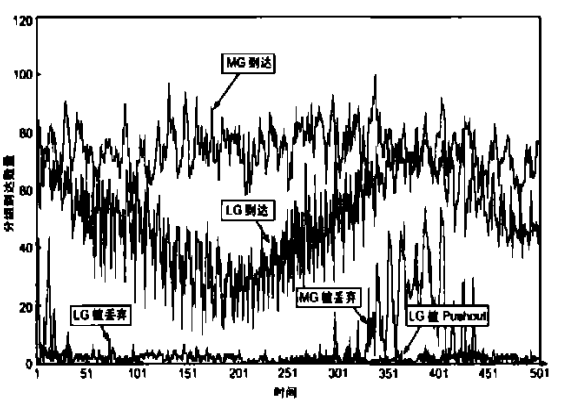


图 8 一段时间内 comBAQ 机制处理分组的曲线

## 5 结论以及进一步的研究

在现有文献中, Aweya 等[11]将动态阈值方法引入 RED 机制, 建立 DRED 算法. 这个算法是一个主动队列管理算法, 计

算过程比较复杂, 并且不能为混合的传输流提供区分服务. Hou 等[12]提出了 SPRED 区分服务机制适合多媒体流应用, 对 TCP 使用 RED 机制, 在拥塞时利用 Pushout 方法保护实时流. 但这个方案不能在多媒体流内部或者 TCP 传输内提供更多的优先级区分服务.

本文提出了区分服务实现机制 comBAQ, 它综合了多优先级最佳动态缓存管理算法和 WSAP 算法. 我们首先提出了五个设计目标, 然后详细说明了 comBAQ 机制的特点和实现. 从模拟实验可以看出, comBAQ 机制符合我们的设计目标, 能够在网络节点为需要可靠保证的流应用和 TCP 流提供区分服务, 并且在同一种传输中能提供不同级别的服务, 是一种很好的区分服务节点机制.

对于 comBAQ 机制, 缓存管理控制的缓存数量与主动队列管理控制的缓存数量的比例关系如何确定还是一个有待深入研究的问题. 在模拟实验中, 我们采用了根据经验静态设置参数的方式. 另一个自然的想法是根据控制需要, 设定两种管理方法各自控制的缓存空间. 如果能够根据分组到达的统计信息, 动态的设置这个参数, 可以进一步提高区分服务机制的缓存使用率和提高性能.

参考文献:


[1] Blake S, et al. An architecture for differentiated services[S]. IETF, 1998, RFC 2475.

[2] Nichols K, et al. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers[S]. IETF, 1998, RFC 2474.

[3] Heinanen J, et al. Assured forwarding PHB group[S]. IETF, 1999, RFC 2597.

[4] Nichols K, et al. A two-bit differentiated services architecture for the Internet[S]. IETF, 1999, RFC 2638.

[5] 李锁钢, 等. 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.

[6] Zhang M, et al. WSAP: Provide loss rate differentiation with active queue management[A]. Proc. of IEEE ICCT' 03[C]. Beijing, 2003, 385.

[7] 李锁钢, 等. 分组交换设备中的缓存管理算法研究综述[R]. 北京: 清华大学计算机系网络所, 2003.

[8] Cidon I, et al. Optimal buffer sharing[J]. IEEE J Selected Areas Commun. 1995, 13: 1229- 1240.

[9] Floyd S, et al. Random early detection gateways for congestion voidance [J]. IEEE/ ACM Trans Networking, 1993, 1: 397- 413.

[10] NS-2[DB/ OL]. http://www.isi.edu/nsnam/ns/.

[11] Aweya J, et al. Multi-level active queue management with dynamic thresholds[J]. Computer Communications Journal (Elsevier Science), 2002, 25(8): 756- 771.

[12] Hou Y T, et al. A differentiated services architecture for multimedia streaming in next generation Internet[J]. Computer Networks Journal (Elsevier Science), 2000, 32(2): 185- 209.


作者简介:


李锁钢　男, 1978 年 7 月生, 内蒙古包头人, 博士研究生, 研究方向是分布式路由器操作系统和缓存管理算法. E-mail: lsg@csnetl.cs.tsinghua.edu.cn.

吴建平　男, 1953 年 10 月生, 山西太原人, 获博士学位, 教授, 博士生导师, 研究方向是计算机网络体系结构, 计算机网络协议测试, 形式化技术.

徐　恪　男, 1974 年 12 月生, 江苏洪泽人, 博士, 讲师, 主要研究方向为新一代互联网络体系结构, 高性能路由器体系结构, 实时分布式操作系统.