第 26 卷第 5 期 电子与信息学报 Vol.26No.5
2004 年 5 月 Journal of Electronics & Information Technology May 2004

一种新的实现区分服务的可扩展缓存管理算法 [1]

魏蛟龙　　肖艳华　　张　驰

(华中科技大学电子与信息工程系　武汉　430074)

摘　要：　目前 Internet 网络中采用的缓存管理策略大多为丢尾 (Drop tail) 算法，并不能适应区分服务模型的要求；而 RED 及其改进算法 CHOKe 也不能提供公平性和相对优先级的保证．该文简要地分析了目前常见的缓存管理算法— RED 以及 CHOKe 所存在的不足，提出了一种基于区分服务模型的新型缓存管理算法— D-CHOKe ．该算法提供了公平性和相对优先级的保证，并且保持了现有 IP 网的可扩展性．仿真试验表明，该算法能够较好地满足区分服务的要求，为不同优先级的流分配不同的带宽．

关键词：　区分服务，丢弃优先级，可扩展性，缓存管理

中图分类号：　TN913.2　　**文献标识码**：　A　　**文章编号**：　1009-5896(2004)05-0795-05

A Novel Scalable Buffer Management Algorithm for DiffServ

Wei Jiao-long　　Xiao Yan-hua　　Zhang Chi

(*Dept of Electron. and Info., Huazhong Univ. of Sci. and Tech., Wuhan* 430074, *China*)

Abstract　Current Internet uses typically drop tail as its buffer management scheme. This scheme does not meet the need of DiffServ model, while RED and its improved algorithms also suffer from shortcoming in term of fairness and relative drop precedence. This paper analyzes several popular packet scheduling algorithms, proposes a novel scalable algorithm named D-CHOKe(Difference-CHOKe) based on DiffServ model. This algorithm achieves fairness and relative drop precedence, and keep the scalability of the IP network. Simulation demonstrates that D-CHOKe algorithm well meets the requirements of DiffServ.

Key words　DiffServ, Drop precedence, Scalability, Buffer management

1 引言

随着 Internet 规模的不断增大，各种各样的网络业务争相涌现，这些业务对于网络传输时延、延时抖动等特性的要求各有不同。这就要求网络管理者对不同的服务区别管理，而不能对所有的数据包一视同仁。而区分服务 (DiffsServ) 体系结构 [1] 正是针对这个问题而提出的一种解决方案。 DiffServ 提供的是一种区别对待不同业务的服务，为不同的业务设置不同的优先级和转发特性，但并不具体规定如何进行转发，也不设法消除拥塞，这样在 DiffServ 网络中完全有可能发生拥塞。在 DiffServ 网络中，按 DS 字段 (DS field) 将 IP 分组分成不同的服务级别 (Service level) ，不同服务级别的 IP 分组具有不同的丢弃优先级 (Drop precedence) 。分组的丢弃优先级越高，则当网络发生拥塞时，其被丢弃的几率也越高。采取怎样的缓存管理策略 (分组丢弃策略)[2,3] 对于网络的性能有着举足轻重的影响。

当前 Internet 中采用的绝大部分是丢尾 (Drop tail) 算法，即当路由器中的缓冲区满时，丢弃随后到达的所有分组。丢尾算法存在下列固有缺陷：没有优先级的概念；每个拥塞周期都有

[1] 2003-07-20 收到， 2004-02-13 改回

湖北省自然科学基金 (No.2001 ABB104) 、图像信息处理与智能控制教育部重点实验室开放基金 (TKLJ0204) 资助课题

可能会引发网络中的“全局同步” (Global synchronization) 现象；不区分 UDP 流和 TCP 流，使 TCP 流在资源竞争中处于不利状态，无公平性保证。

针对丢尾算法的固有缺陷，S. Floyd 和 V. Jacobson 提出了 RED(Random Early Detection) 算法 [4] 。 RED 算法的基本思想是通过监控路由器输出端口队列的平均长度来探测拥塞，一旦发现拥塞逼近，就随机地选择连接来通知拥塞，使他们在队列溢出导致丢包之前减小拥塞窗口，降低发送数据速度，从而缓解网络拥塞。

RED 算法的优点是比较简单，易于实现，消除了全局同步现象，较大地提高了物理线路的利用率，并且保证了一定的公平性。但是 RED 算法仍存在一定的缺陷：首先， RED 算法没有分组优先级的概念，不能适应各种用户的不同要求；其次，该方法并不能独立地做到避免拥塞的发生，当拥塞发生时也无法保证对不同数据流的公平性，它要依赖用户终端协作与配合才能真正发挥作用。并且不能够解决相对优先级问题和提供公平性保证。

目前的研究工作集中在如何在 Internet 中实现区分服务体系结构。已提出的算法存在的问题是： (1) 它们都要求网络是状态相关的，即要求路由器保持每个流的状态信息。因而这些方法都破坏了 Internet 赖以成功的基本特性：可扩展性、鲁棒性。 (2) 它们都使得路由器变得十分的复杂。 (3) 它们要求整个网络的软件和硬件有较大的改变，这种改变所需的投资对于大范围的广域网是不可想象的。

由于路由器对分组的控制主要体现在缓存管理和队列调度上，通过修改缓存管理算法可以间接地控制输出链路带宽分配，而且易于软件实现。因而本文提出一种可扩展的缓存管理算法 D-CHOKe (Difference-CHOKe) 来实现区分服务体系结构。

2 优先级缓存管理算法

2.1 路由器调度算法 (CHOKe)

为了解决 RED 的公平性问题， Rong Pan 等人在 RED 的基础上提出了一种新的路由器调度算法—— CHOKe[5,6] 。 CHOKe 的主要目的也是保护适应流，惩罚非适应流，保证网络上数据流之间的公平性。其实现过程是，当一个包到达拥塞的路由器时， CHOKe 从 FIFO 队列中随机地挑出一个包进行比较。如果它们属于同一个流，则这两个包都被丢弃；否则，被挑出的包依然留下，而刚到达的包则依某种概率被丢弃，此概率的计算和 RED 中一样。

由于非适应流的包在缓存中占据的比例较大，同时非适应流到达的速率也较高，进行比较的几率大。因而，非适应流被丢弃的概率将会在很大程度上大于适应流，从而实现了对非适应流的惩罚。

CHOKe 算法的实现比较简单，不需要保留网络上各流的状态，同 RED 相比，能够比较好地实现对非适应流的抑制，从而保证网络上的公平性。但是 CHOKe 仍旧没能解决相对优先级的问题。

2.2 优先级缓存管理算法描述

针对区分服务模型中“逐跳行为” (Per-Hop-Behavior, PHB) 的要求和弥补现有分组丢弃算法中存在的缺陷，本文提出一种新的分组丢弃算法： D-CHOKe ，在 CHOKe 算法的基础上实现区分服务。其基本思想是将状态相关的控制转移到网络边界节点，即保持网络内部简单，将复杂性推到网络边缘。在网络边界节点实施流状态信息的保存与监控机制，由边缘路由器进行流的分类、整形、聚集等。假设 DiffServ 的优先级设置为 $i(i=0,1,2,\cdots$ 优先级递增) ，由边缘路由器将相应的优先级标签插入到分组包头中。在内部节点实施适用于全网业务的一套“逐点行为”，且只进行简单的调度转发，内部节点是状态无关的。核心路由器仅根据进入的聚合流的优先级来对不同的流实施不同的丢弃算法。 D-CHOKe 是核心路由器上的调度算法，不保留任何流的状态信息，因而具有很好的可扩展性。

当一个新分组到来时， D-CHOKe 按照以下步骤对分组进行处理：

步骤 1　D-CHOKe 算法采用“指数加权滑动平均”的方法计算队列平均长度：

$$q\_\text{new} = (1-\omega)\cdot q\_\text{old} + \omega \cdot q\_\text{inst}$$

其中 $\omega(0<\omega<1)$ 是系统参数——指数平滑的权重系数， q_new 是更新后的平均队列长度估计值， q_old 为新分组到达前的平均队列长度估计值， q_inst 为新分组到达时的瞬时队列长度。

步骤 2　将求得的平均队列长度与缓存区的最小门限 (min_th) 和最大门限 (max_th) 比较，当拥塞程度 (以平均队列长度 q_new 表示) 小于 min_th 时，分组不丢弃；而当 q_new 大于 max_th 时，则所有到达的分组全部被丢弃。

步骤 3　当 q_new 介于 min_th ， max_th 之间时，首先检查分组的包头中的优先级标签 i ，根据不同的优先级确定分组的处理方式，对不同优先级的分组以 $1/(2^i)$ 的概率实行抽样比较。抽样比较的方法是：令 R_j 为区间：

$$\left[\min\_\text{th}+\frac{j-1}{k}(\max\_\text{th}-\min\_\text{th}),\ \min\_\text{th}+\frac{j}{k}(\max\_\text{th}-\min\_\text{th})\right],\qquad j=1,2,\cdots,k$$

当 $q\_\text{new}\in R_j$ 时，从缓存区中随机抽取 $2j$ 个分组与到达的分组比较，只要其中有分组与新到达的分组属于同一个流 (具有相同的 ID) ，那么丢弃有相同 ID 的所有分组。

步骤 4　如果新分组没有在抽样比较中被丢弃，则根据平均队长以概率 P 丢弃新分组：

$$P=\frac{P_{\max}(q\_\text{new}-\min\_\text{th})}{(\max\_\text{th}-\min\_\text{th})}$$

即当平均队长 q_new 从 min_th 变化到 max_th 时，丢弃概率 P 从 0 线性地变化到 $P_{\max}$ 。

这样既可以实现对不同优先级的分组的丢弃概率的控制，从而达到按比例分配网络资源的目的，同时也可以实现对网络的拥塞控制。

3　仿真试验及其结果

我们采用 NS-2[7] 软件进行了仿真，并将编写的 D-CHOKe 实现模块编译到 NS 系统中。试验的目的是检验算法的公平性和相对优先级的问题，为此我们设计了 3 个试验， (1) 当各流处于同一优先级时，检查对不同流能否公平地分配带宽； (2) 当不同优先级的流竞争拥塞带宽时，能否依据优先级按照一定比例对带宽进行分配； (3) 当不同优先级的聚合流竞争拥塞带宽时，能否依据优先级对带宽进行按比例分配，同时保证聚合流之间的公平性。

试验 1　网络的拓扑结构如图 1 所示，瓶颈链路位于两个核心路由器之间。从 $s1,s2,s3$ 等 3 个网络终端发出 3 个具有相同优先级的 UDP 流，竞争核心路由器间带宽为 5 Mb/s 的瓶颈链路， 3 个节点上 3 个 UDP 流的发送速率分别为 3 Mb/s, 5 Mb/s, 7 Mb/s 。 3 个 UDP 流均通过

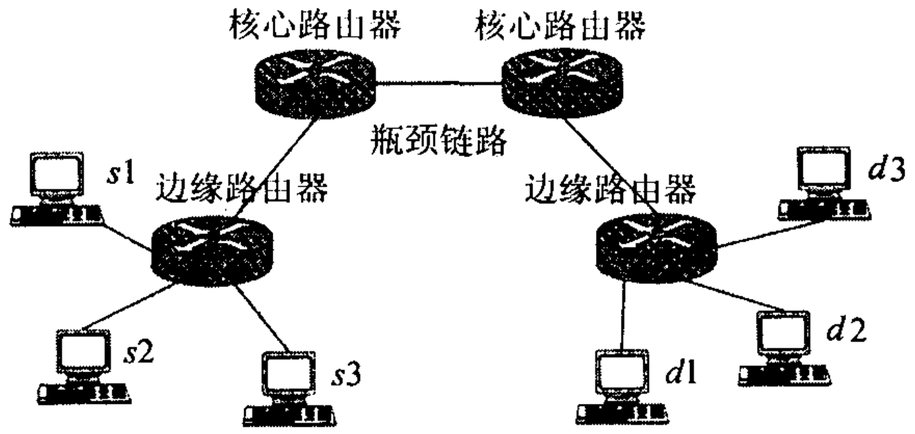

图 1　网络拓扑结构图

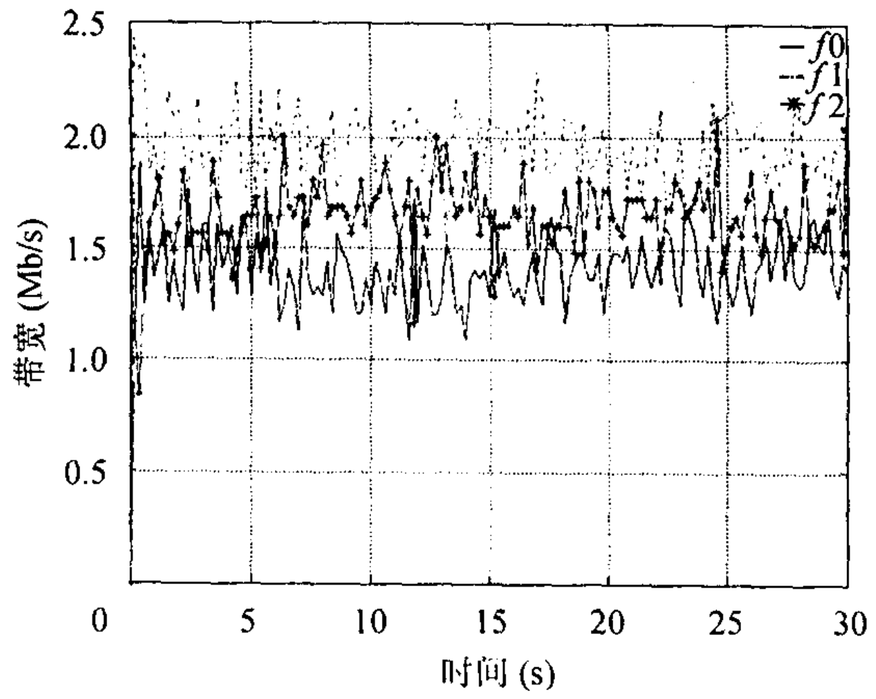

图 2 同一优先级、不同速率流的带宽分配

1 条 10 Mb/s 接入链路连接到边缘路由器，边缘路由器与核心路由器之间的链路带宽为 15 Mb/s 。仿真时间为 30 s 。试验结果如图 2 所示。

图中 x 轴为时间 (s)， y 轴为实际获得的带宽 (Mb/s) 。$f0, f1, f2$ 分别代表发送速率为 3 Mb/s, 5 Mb/s, 7 Mb/s 的 3 个 UDP 流所获得的实际带宽。从图中可以看出， 3 个不同速率的流分得的带宽相差不大，对速率高的流的抑制能力较强，能够实现流之间的公平性。

试验 2 仍采用上面试验的网络拓扑结构。 3 个不同优先级的 UDP 流竞争这个带宽为 5 Mb/s 的瓶颈链路。优先级设置为 0, 1, 2 。 3 个 UDP 流均通过 1 条 10 Mb/s 接入链路连接到边缘路由器，边缘路由器与核心路由器之间的链路带宽为 15 Mb/s 。仿真时间为 30 s 。试验结果如图 3 所示。

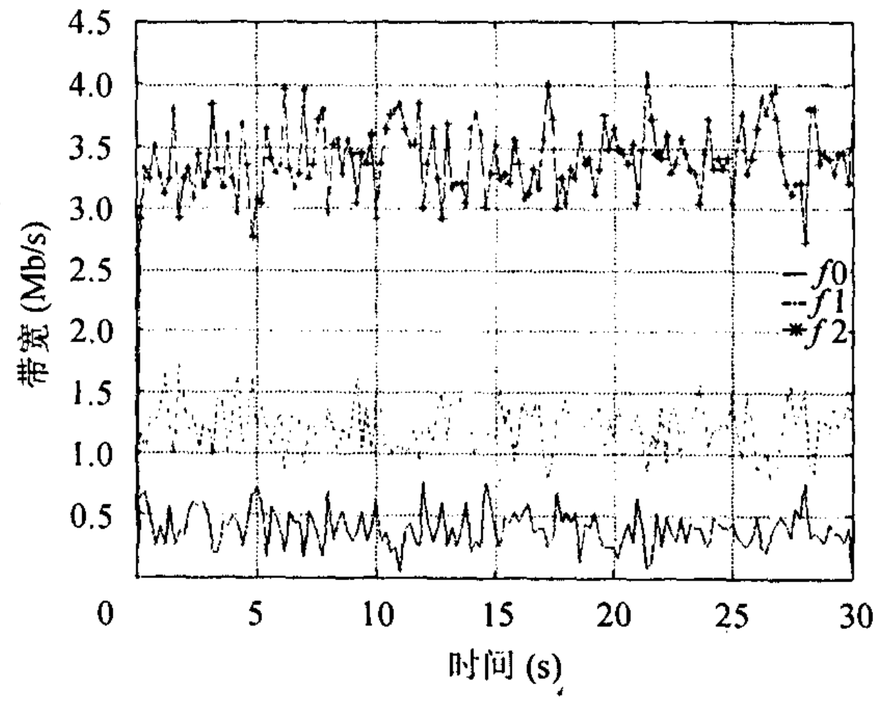

图 3 不同优先级、相同速率流的带宽分配

图中 $f0, f1, f2$ 分别代表优先级为 0, 1, 2 ，发送速率为 5 Mb/s 的 3 个 UDP 流所获得的实际带宽。从图中可以明显看出，在拥塞发生的情况下，3 个流分得的带宽有很大的差异，优先级高的流分得带宽明显高于优先级低的流，而这正是区分服务所要达到的目标。并且，可以通过调整优先级 i 的大小，来调节各流之间分得带宽的比例。

试验 3 仍采用上面试验的网络拓扑结构。 3 个不同优先级的聚合流竞争带宽为 5 Mb/s 的瓶颈链路，聚合流的优先级设置为 0, 1, 2, 每个聚合流由两个不同速率的 UDP 流组成，速率分别为 3 Mb/s, 5 Mb/s 。 3 个 UDP 流均通过一条 10 Mb/s 接入链路连接到边缘路由器，边缘路由器与核心路由器之间的链路带宽为 25 Mb/s 。仿真时间为 30 s 。试验结果如图 4 所示。

表 1 试验 3 的参数设置

流标号	$f0$	$f1$	$f2$	$f3$	$f4$	$f5$
发送速率 (Mb/s)	5	3	5	3	5	3
优先级	0	1	2	0	1	2

图中 $f0 \sim f5$ 分别代表优先级与发送速率各不相同的 6 个 UDP 流所获得的实际带宽，各个流的具体参数设置见表 1 。从上图可知，当存在不同优先级的聚合流时，聚合流之间将会按照优先级的高低来分配带宽，确保了区分服务，而相同优先级之间的流则可以实现公平的分配带宽。

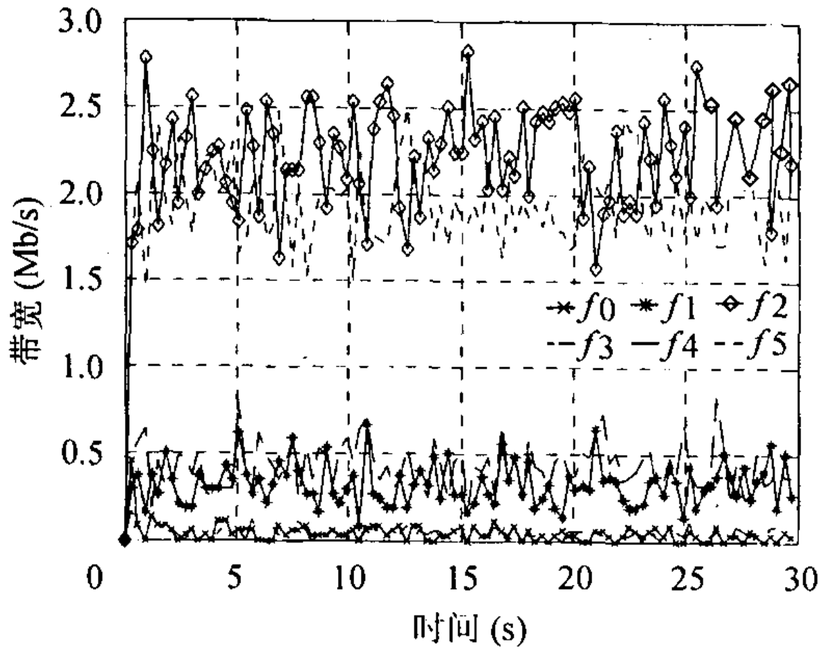

图 4　不同优先级的聚合流的带宽分配

4 结束语

本文针对区分服务的要求，在 CHOKe 的基础上提出了一种改进的缓存管理算法—— D-CHOKe 。D-CHOKe 算法能准确地反映网络的拥塞状况，保证不同优先级用户所需的带宽，较好地实现了相同优先级流之间的公平性。同时在参数设置上与 CHOKe 算法相同，只需对 CHOKe 算法做很小的修改便可以实现，因而在实现和推广上具有很大的优势。因此，D-CHOKe 算法弥补了 RED 算法及其改进算法 CHOKe 在公平性、优先级问题上的不足，是一种更适合于 DiffServ 模型的缓存管理策略。

参　考　文　献

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

[2] Pan R, Nair C, Yang B, Prabhakar B. Packet dropping schemes, some examples and analysis. Proceedings of the 39th Annual Allerton Conference on Communication, Control and Computing, Illinois, USA, October 2001: 563–572.

[3] Pan R, Breslau L, Prabhakar B, Shenker S. Flow table-based design to approximate fairness. Proceedings of Hot Interconnect X., Stanford Symposium on High Performance Interconnects, Stanford, Calif., August 2002: 37–42.

[4] Sally Floyd, Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1993, 1(4): 397–413.

[5] Pan R, Prabhakar B, Psounis K. CHOKe, a stateless active queue management scheme for approximating fair bandwidth allocation. Proceedings of the IEEE INFOCOM, Tel Aviv, Israel, March 2000, 2: 942–951.

[6] Tang Ao, Wang Jiantao, and Low Steven H. Understanding CHOKe. In Proc. of IEEE Infocom, San Francisco, April 2003: 82–93.

[7] The ns Manual. The VINT Project. http://www.isi.edu/nsnam/ns/doc/index.html.

魏蛟龙：　男，1965 年生，在职博士生，副教授，现主要从事广域网拥塞控制、IP 网服务质量保证、人工智能与专家系统、故障诊断等方面的研究.

肖艳华：　女，1978 年生，硕士生，现主要研究领域为网络拥塞和流量控制.

张　弛：　男，1977 年生，博士生，现主要研究领域为网络拥塞和流量控制.