主动队列管理算法的分类器实现

任丰原, 林 闯, 黄小猛, 刘卫东
(清华大学计算机科学与技术系, 北京 100084)

摘 要: 作为端到端拥塞控制的增强机制, 主动队列管理(AQM) 通过在网络中间节点有目的地丢弃分组来维持较小的队列长度和较高的链路利用率. 已有的大多数主动队列管理算法沿用了随机早期探测(RED) 算法首创的概率丢弃机制. 本质上, 判决是否丢弃分组的过程是一个依赖于网络拥塞状态的决策过程, 因此, 概率决策不应该是唯一的方法. 在本文的研究中, 我们首先归纳了理想 AQM 算法所应具备的品质, 然后应用模式识别中分类器的设计思想提出了一种新颖简洁的主动队列管理策略实现框架, 并基于 Fisher 线性判别方法为 AQM 设计了一个两维两类分类器(TCC). 仿真试验表明 TCC 有效、敏捷、鲁棒, 扩展性好, 同时实现简单, 计算开销小, 有利于高速路由器的性能优化.

关键词: 主动队列管理; 拥塞控制; 分类器
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2004) 11-1796-05

A Classifier Implementation for Active Queue Management Algorithm

REN Feng-yuan, LIN Chuang, HUANG Xiao-meng, LIU Wei-dong
(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

Abstract: Active Queue Management (AQM) is an enhancement mechanism to end-to-end congestion control, which can maintain smaller queuing delay and higher throughput by purposefully dropping the packets at intermediate nodes. Almost all the existing AQM schemes follow the probability dropping mechanism originated from Random Early Detection (RED). It is essentially a decision process with aid of information about network congestion that AQM decides whether or not to drop an incoming packet, thus the probability discrimination should not be only one way. We firstly summarize the properties of an ideal AQM scheme, and then develop a novel framework for AQM scheme based on the two-category classifier, which is considerably simple and extensible. A two-dimensional two-category classifier (TCC) for AQM is designed using the Fisher Linear Discriminate approach. The simulation results show that TCC outperforms other well-known AQM schemes in the integrated performance, namely TCC is an effective, prompt and robust algorithm. Additionally, TCC scheme requires few CPU cycles, which makes it deploy in high-speed routers simply.

Key words: active queue management; congestion control; classifier

1 引言

Internet 由许许多多异构的自治域通过松耦合方式组成, 具有与生俱来的分散特性. 在 Internet 技术发展的初期, 考虑到业务发展的需要, 在资源的管理和应用方面, "端到端原则"倍受重视, 即与业务流相关的状态都应该尽量在终端系统上加以实现与维护. 该指导原则的直接结果就是拥塞控制的绝大多数功能都是在主机端实现的, 对网络中间节点所能发挥的作用考虑较少. 无庸置疑, 端到端原则提供的可扩展性在客观上有利于 Internet 的快速发展, 但随着应用要求的日益丰富和技术的不断发展, 研究者开始认识到要想完全依赖终端系统上的策略与算法很难满足诸如 QoS 这样复杂的应用需求. 于是, 开始将部分研究注意力转向网络中的路由器等中间节点设备, 期望通过增强它们的功能来实现端系统无法达到的技术目标. 就拥塞控制而言, 网络中间节点有可能更及时, 甚至提前准确预测网络的拥塞状态, 从而实施有效的资源管理策略, 使网络能够有效的避免拥塞, 或尽早从深度拥塞状态中恢复过来. B Braden[1] 等人在 IETF 提出 AQM 研究的目的即在于此.

设计、分析和评价 AQM 算法已经成为近来网络研究的一个技术热点. 最初, RFC2309 推荐 RED[2] 为唯一的 AQM 实现算法, 但接下来的试验研究[3] 和理论分析[4~6] 表明: RED 在某些网络环境中表现出不稳定或不公平. 为了增强 RED 算法的鲁棒性, 研究者提出了一系列的 RED 变种算法, 典型的有 Stabilized RED[7]、Balanced RED[8]、Adaptive RED[9]、Self-Configuring RED[10]、Weighted RED[11] 和 gentle-RED[12] 等. 此外, 也产生了一些新的策略, 主要有 BLUE[13]、REM[14]、PI 控制器[15]、模糊逻辑控制器[16]、SMVS[17] 和 AVQ[18] 等. 总结上述已有的算法和策略, 除 AVQ 引入虚队列概念辅助分组丢弃决策之外, 所有的算法在确定是否丢弃分组时, 全部沿用了 RED 的概率丢弃机制, 不同之处主要表现在概率的计算和更新方法上. 毋庸置疑, 概率丢弃是实现 AQM 的一种有效手段, 但我们认为: 它

收稿日期: 2004-04-25; 修回日期: 2004-05-24
基金项目: 国家自然科学基金(No. 60273009, No. 90104002); 国家重点基础研究发展规划(973 计划)(No. 2003CB314804); 华为研究基金

不应该是唯一的, 也不一定是最优的. 确定是否丢弃新到达分组的过程本质上是一个基于一定优化目标的决策过程, 决策的正确性取决于依赖信息的充分性和可靠性. 概率丢弃机制是通过算法确定的函数将诸如队列长度、分组到达速率和链路利用率等观测变量描述的网络拥塞状态信息映射为 0 和 1 之间的数, 然后与实时生成的随机数进行比较, 从而决策是否丢弃新到达分组. 因为不同的状态观测变量往往具有不同的物理属性, 所以很难构造简单函数融合来自不同拥塞观测变量的信息, 并最终转化为一个简单的数. 正因如此, 在计算或更新分组丢弃概率时, 已有的大多数 AQM 算法往往只能考虑单个变量, 譬如 RED 及其变种算法中的平均队长或瞬时队长, BLUE 中的链路利用率, PI 控制器中的队长变化速率等. 另一方面, 生成随机数的计算开销在高性能路由器的设计中也是一个不可回避的问题. 在本文的研究中, 我们将放弃流行的概率分组丢弃机制, 基于模式识别理论为主动队列管理提出一种实现简单、扩展性好的新机制; 并应用 Fisher 线性判别函数方法[19] 给出一种具体的实现算法; 通过仿真试验, 分析比较新算法与已有典型算法的性能.

2　设计思想

2.1.1　理想 AQM 算法的品质

在讨论具体的算法设计之前, 我们首先来分析和归纳一下理想的 AQM 算法所应具备的品质, 它们是设计具体算法的指导原则. 因为 RED 在稳定性和公平性方面存在问题, 目前大多数的 AQM 研究格外重视这两方面的问题, 但一个真正可以实践的路由器队列管理算法需要同时满足多方面的要求, 技术指标固然重要, 但其他因素也需要给予一定的考虑. 综合各个方面, 我们认为: 一个理想的 AQM 算法需要同时具备以下品质.

(1) 有效性: AQM 的技术目标可以概括为: 在较高的链路利用率和较低的分组排队延时之间取得一个恰如其分的平衡. 较小的队列长度允许路由器有条件在不丢弃分组的前提下最大限度的吸纳突发性业务流. 在正常情形下, 分组经历的排队时延也会小许多, 这对实时性的交互式应用尤为重要. 当然, 片面追求小队长, 空队列概率加大, 导致链路利用率降低也是不期望的. 最有效的方法是无论网络状态如何, 负载如何变化, 始终将队列保持在一个较小的恒定值附近. 在对 RED 算法的改进中, S Floyd 引入目标队长概念正是基于这种考虑[9]. 这一认识已被越来越多的研究人员认可[15, 17, 20]. 从控制理论的角度分析, AQM 的技术目标是需要实现一个调节系统, 因此在评价算法有效性时, 收敛时间自然是一个不可忽视的指标. 连接的不断建立与拆除使得网络状态瞬息万变, 算法必须具有很好的响应性才能充分适应快速变化的网络环境.

(2) 鲁棒性: AQM 算法的鲁棒性包括两个方面. 一是稳定性, 即 AQM 算法自身必须稳定, 不能像 RED[3] 和 AVQ[20] 那样可能导致队列振荡. 二是抗扰性. AQM 算法应该有能力抵抗非响应性业务(如 UDP) 和短时间突发业务(如 HTPP 或 Telnet) 等扰动或噪声的干扰.

(3) 可扩展性: Internet 的快速发展表现在许多方面: 用户数急剧增加, 带宽迅猛提升, 局域网和广域网规模不断扩大, 理想的 AQM 算法必须在各个维度上具有可扩展性. 不仅能工作在平均往返时延(RTT, Round Trip Time) 为几十毫秒的 LAN 或 WAN 中, 而且要能稳定工作于有数百毫秒, 甚至秒数量级 RTT 的 WAN 环境下; 既可以支持带宽为数兆比特的专用线路, 也可以支持吉比特的光通路; 在用户较少的接入网与连接密集的核心网中都可以有效工作, 不会因为需要维护单个流的状态信息而影响算法的性能和实现的复杂度.

2.1.2　设计原理

TCP/ AQM 构成的闭环系统是控制理论中一个典型的调节系统. AQM 调节器通过有目的丢弃或标记一些分组产生控制信号, TCP 终端系统加以响应, 最终将路由器队列保持在一个期望值附近. 一般而言, 路由器在进行决策分组丢弃与否的过程中依赖的信息越充分, 所作的决定便越科学. 为探测可能发生的拥塞, 路由器可以利用各种各样的观测信息, 包括队列长度、链路利用率和分组到达速率等. 假定观察并预测网络潜在拥塞的状态变量有 n 个, 它们一起构成了一个 n 维状态空间. 该空间中的任意一点对应于路由器的一种工作状态, AQM 算法的功能既是辅助该状态下的路由器作出接收或者丢弃(包括标记) 新到达分组的决策, 没有其他可能的选择. 从这一意义上讲, AQM 调节器等价于一个具有多维度的两类分类器, 因此, 我们可以通过设计分类器来得到所需的 AQM 算法, 也就是在 AQM 技术目标的约束下, 通过监督学习, 构造超平面将 n 维状态空间分为两部分. 作为示例, 下面我们为 AQM 设计一个二维的两类分类器.

3　算法设计

首先, 我们需要确定状态观测变量. 队列长度被众多拥塞控制机制用来感知网络的拥塞状态. 它的确非常有效, 一方面, 它容易获得. 另一方面, 它的大小与网络负载状态有着非常直接和密切的联系. 但完全依赖于此而推断网络的拥塞状态, 可能会得到错误的结论. 譬如, 在一个缓冲大小为 200KB 的路由器队列中存在两种状态, 一种状态队列长度为 150KB, 队列长度的变化速率为- 70KB/ s. 另一状态的队列长度为 70KB, 变化速率为+ 70KB/ s. 哪一种状态更容易发生拥塞是非常明显的, 仅仅依赖队列长度无法得到正确的结论. 上面的例子同时也告诉我们, 队列长度的二阶信息对预测可能发生的拥塞也是非常重要的, 它将会使主动队列管理的措施更加及时主动. 基于上述分析, 我们选择瞬时队列长度 q 和它的变化速率 $q 作为网络拥塞状态的观测变量, 它们构成了一个 2 维状态平面. 为得到观测变量的值, 以周期 T 采样队列长度, 得到时间序列 q(kT) (k= 1, 2, , ; t= kT). 为方便起见, 定义差分 $q(kT) = q(kT) - q((k- 1) T) 为队列长度的变化速率.

为进行监督学习, 我们需要一些训练样本, 它们构成集合 Ω1= {(ei(kT), $ei(kT)) , i= 1, 2, 3, , , n} 和集合 Ω2= {(ej(kT), $ej(kT)) , j= 1, 2, 3, , , m}, 其中, 偏差 e(kT) = q(kT) - q0(q0 为期望队列长度), 偏差的变化速率 $e(kT) = e(kT) - e((k- 1) T) = $q(kT), 这里 Ω1 和 Ω2 分别表示接收和丢弃分组的状态集合, 也就是说, 当路由器工作于 Ω1 中元素确定的某一状态时, 它将接收新到达分组; 而 Ω2 集合中的任意状

态都会使路由器丢弃分组. 我们依据经验给定一组初始值, 然后通过大量的仿真试验进行调整与优化, 最终得到如下结果:

Υ_1= {(- 28, 18), (- 28, 0), (- 28, 18), (- 21, 27), (- 21, 9), (- 21, - 9), (- 21, - 27), (- 14, 18), (- 14, 0), (- 14, - 18), (- 7, 9), (- 7, - 9), (- 7, - 27), (0, 18), (0, 0), (0, - 18), (7, 9), (7, - 9), (7, - 27), (14, 0), (14, - 18), (21, - 27)}

Υ_2= {(- 7, 27), (7, 27), (14, 18), (21, 27), (21, 9), (21, - 9), (28, 18), (28, 0), (28, - 18), (35, 27), (35, 9), (35, - 9), (35, - 27), (28, 18), (28, 0), (28, - 18)}

为直观起见, 将它们表示在图 1 中.

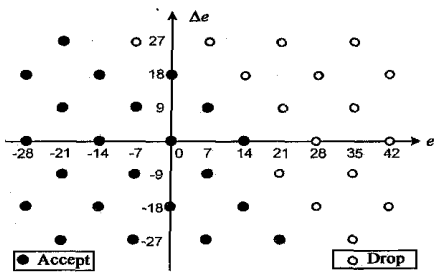

图 1 训练样本

建立了训练样本集合, 接着我们应用经典的 Fisher 线性判别方法[19]为 AQM 设计二维线性分类器, 即寻找一个合适的矢量 w, 使样本在其上的投影具有很好的分离度. 为此, 定义各类样本均值 m_i 和样本内离散度 s_i

$$m_i = \frac{1}{n_i} \sum_{y \in \Upsilon_i} y, \qquad i = 1, 2 \tag{1}$$

$$s_i^2 = \sum_{y \in \Upsilon_i} (y - m_i)^2, \qquad i = 1, 2 \tag{2}$$

那么使 Fisher 准则函数

$$J(w) = \frac{|m_1 - m_2|^2}{s_1^2 + s_2^2} \tag{3}$$

最大化的投影方向既为 w 的解, 它使得类均值之差 $(m_1 - m_2)$ 达到最大, 同时保证了各样本内部尽量密集. 由于篇幅所限, 这里直接给出 w 的显式解, 详细的推导与证明参见文献[19]. 如下定义样本均值向量 m_i, 样本类内离散度矩阵 S_i 和总类内离散度矩阵 S_w:

$$m_i = \frac{1}{n_i} \sum_{x \in \Upsilon_i} x, \qquad i = 1, 2 \tag{4}$$

$$S_i = \sum_{x \in \Upsilon_i} (x - m_i)(x - m_i)^t, \qquad i = 1, 2 \tag{5}$$

$$S_w = S_1 + S_2 \tag{6}$$

则, w 的解为:

$$w = S_w^{-1}(m_1 - m_2) \tag{7}$$

选定分界阀值点 w_0 为:

$$w_0 = \frac{\frac{1}{n_1} \sum_{x \in \Upsilon_1} w^t x + \frac{1}{n_2} \sum_{x \in \Upsilon_2} w^t x}{2} \tag{8}$$

则两类分类器的决策规则如下:

$$w^t x \begin{cases} < w_0 & x \in \Upsilon_1 \\ \geq w_0 & x \in \Upsilon_2 \end{cases}, \quad (w^t \text{ 为 } w \text{ 的转置向量}) \tag{9}$$

将图 1 中的样本数据代入式(4)~(9), 得到:

$$w = \{0.0058, 0.0028\}$$
$$w_0 = 0.0589 \tag{10}$$

至此, 我们得到了所需要的二维两类分类器, 为方便论述, 将它称为 TCC(Two-category Classifier)算法, 图 2 给出了实现 TCC 的伪代码.

```
/* TCC Algorithm */
w1 = 0.0058; w2 = 0.0028; w0 = 0.0589
q(k): queue length;        T: sampling time
q0: expected queue length

At each packet arrival epoch do
    if(now_time() > last_time + T)
        x1 = q(k) - q0;    x2 = q(k) - q(k-1)
        last_time = now_time()
    else
        null
    if(w1 @ x1 + w2 @ x2) > w0
        drop packet
    else
        enqueue packet
```

图 2 TCC 算法的伪代码

4 仿真实验

我们在 ns2.19b 网络仿真平台[21]上实现了 TCC 算法来验证和评价它的性能. 采用图 3 所示的哑铃型拓扑结构, 节点 A 的缓冲为 300packet(分组缺省大小为 1000bytes), 瓶颈链路的容量和传播时延设定为(L, S), 其他链路均为(10Mbps, 5ms). 业务源分为三类, 第一类包括 N_1 个贪婪持久性的 FTP 业务; 第二类由 N_2 个突发性的 HTPP 业务组成, 其中每连接有 10 个会话业务, 每个会话业务需要传输 3 幅页面文件; 第三类包括 N_3 个服从负指数 ON/OFF 模型的非弹性 UDP 业务, 空闲和突发时间间隔的均值分别为 10s 和 1s. 突发其间业务生成速率为 40Kbps. 为便于比较, 我们分别用几种流行的 AQM 算法控制队列 A. 它们的参数配置如下:

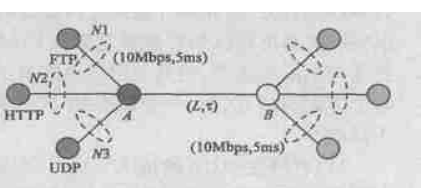

图 3 仿真拓扑结构

RED: 最大/最小门限值分别设定为 75/15packets, 其余参数参照文献[22]设定.

PI: 控制器参数依照文[15]设定. 采样频率和期望队列长度分别给定为 160Hz 和 50packets.

AVQ: 链路利用率的期望值 C 和平滑因子 A 分别设定为 0.98 和 0.15.

TCC: 设定期望队列长度 q_0 为 50packets, 采样周期为传输 q_0 个分组所需时间.

为了在尽可能现实的网络环境中评价包括, 包括有效性、响应性和鲁棒性等在内的算法综合性能. 我们给定 N_1= N_2=

200, N_3= 20, (L, S) = (15Mbps, 50ms). FTP 业务源被进一步均分为两组, t= 0 秒时启动第一组; t= 150 秒时第二组启动, 再过 150s 后停止第二组 FTP 业务, 启动 HTPP 业务, UDP 业务在 t= 450s 时启动, 整个仿真试验持续 600s. 跟踪瞬时队列长度的变化, 将结果描述在图 4~ 图 7 中.

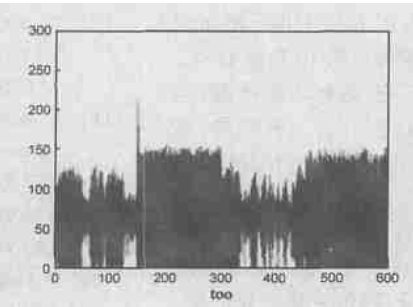

图 4　RED

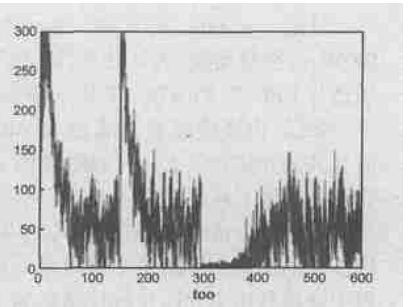

图 5　PI

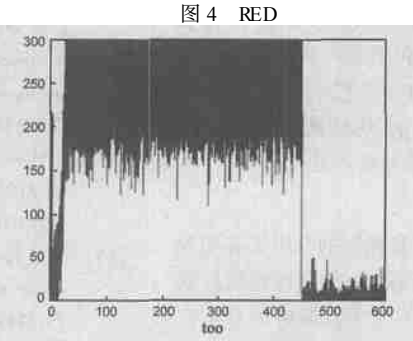

图 6　AVQ

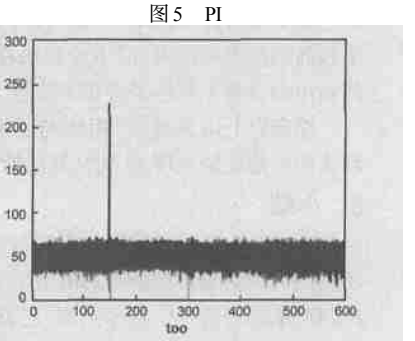

图 7　TCC

观察图 4, RED 敏感负载的缺陷再一次得到证实, 当持久性的 FTP 业务在 t= 150s 增加到 200 时, 队列开始表现出幅度较大的振荡. 此外, 非响应性 UDP 业务流的干扰产生的负面影响也很明显. PI 控制器响应性差的弱点在图 5 中表现地比较明显, 300s 时一部分 FTP 结束会话, 但因 PI 控制器响应迟缓, 迟迟不能将分组丢弃概率调节到一个较小的稳定值附近, 致使过多分组被丢弃, 队列被清空, 链路得不到充分利用. 新业务加入导致过长的暂态调节过程虽不会影响链路利用率, 但却无法有效控制队列长度, 进而控制排队延时, 图 5 中 PI 用了 50 多秒的时间完成这一过程算不上理想. AVQ 没有明确的队长控制目标, 为了保持 98% 的链路利用率, 对队长的控制很保守, 大多数时间瞬时队长接近满队列, 这不利于实现 AQM 平衡高链路利用率和低排队延时的技术目标. 另一方面, AVQ 非常敏感非响应性业务流的干扰, 几乎很难到达预期的技术目标, 这一点图 6 中 450s 以后的队长变化曲线反映的很清楚. 反观图 7, 队列长度自始至终在期望值附近抖动, 只是在状态切换过程中出现了短暂的尖峰. UDP 业务流的干扰使队长摆动幅度有微弱的增加, 但不易觉察, 证明 TCC 算法有很强的抗干扰能力. 整个动态仿真试验表明: TCC 是一种有效、敏捷和稳健的 AQM 实现算法.

接下来, 我们将从负载、瓶颈链路容量和往返时间等三个方面进一步考察 TCC 算法的可扩展性. 因为依据文献[4]中的结论, 这三个参数是影响 AQM 算法稳定性的主要因素. 首先, 令(L, S) = (15Mbps, 20ms), 其余参数保持不变, 改变 FTP 连接数, 重复实验, 跟踪相关数据, 计算瓶颈链路利用率、队长均值及其标准偏差, 将结果表示在图 8 和图 9 中. 观察图 8, TCC 算法几乎不敏感负载变化, RED 却使链路利用率随着会话数的增加而减小.

表面上看, PI 控制器和 AVQ 也独立于负载, 但图 9 中的队长标准偏差值(图中竖线)却很大, 这说明队列存在着严重振荡, 加剧了控制延时及其抖动的难度, 但 TCC 却能始终以较小的标准偏差值将队列维持在期望值附近.

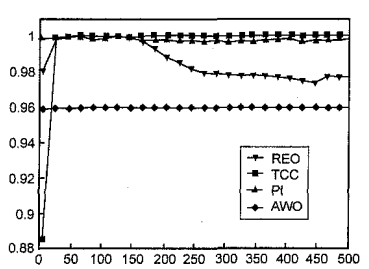

图 8　利用率 / 负载

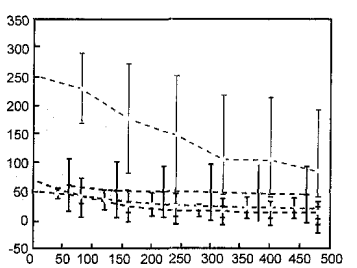

图 9　队长统计值 / 负载

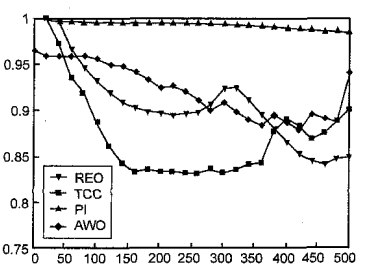

图 10　利用率 / 链路容量

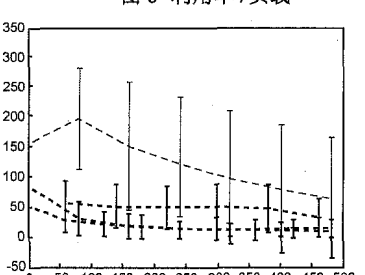

图 11　队长统计值 / 链路容量

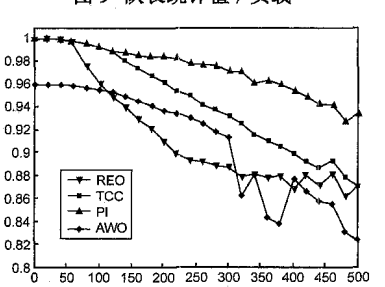

图 12　利用率 / 延时

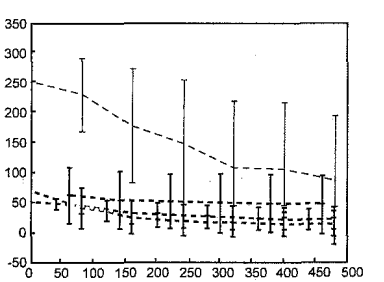

图 13　队长统计值 / 延时

设定 S= 20ms, N_1= 100, N_2= N_3= 0, L 从 1Mbps 增加到 500Mbps, 重复实验, 统计结果, 将它们刻画在图 10 和图 11 中. 相比与 RED、PI 和 AVQ, 在某些情形下, 链路利用率稍低, 但可以接受, 但当链路容量超过 350Mbps 之后, 有上升趋势, 最后与其他算法持平. 另一方面, 就图 11 中队长统计结果而言, TCC 明显优于其他算法.

给定 L= 15Mbps, N_1= 100, N_2= N_3= 0. 图 12 和图 13 分别描述的是瓶颈链路的广播时延 S 从 0ms 增加到 500ms 的过程中链路利用率和队长统计结果的变化曲线. 在所有算法中, 时延加大给链路利用率带来的影响都是负面的, 尽管 PI 保持了较高的链路利用率, 但 TCC 算法在利用率和队长统计值之间给出的合理平衡结果应该更符合 AQM 的技术目标.

综合以上仿真试验, 相比与已有的 AQM 典型算法, TCC 算法的性能更接近理想 AQM 算法所应具备的品质.

5 小结

主动队列管理通过网络中间节点有控制的分组丢弃实现了较低的排队延时和较高的有效吞吐量, 是 TCP 端到端拥塞控制近来研究的一个技术热点. 已有的大多数策略和算法在判定分组丢弃时大都沿袭了 RED 的概率丢弃机制. 我们认为该机制是有效的, 但不是唯一的. 判定是否丢弃分组的过程本质上是一个依赖网络拥塞状态的决策过程, 应该有不同于概率丢弃的其他可供选择的方法. 这一思路启发我们提出了基于多维两类分类器的 AQM 实现框架, 它方便融合多种信息给出更加合理的决策. 应用模式识别中的 Fisher 线性判别方法设计的 TCC 算法在大量仿真试验中被证明综合性能优于 RED、PI 和 AVQ 等典型算法, 而且扩充性强, 简单易实现, 有利于高速路由器的性能优化.

参考文献:

[1] B Braden, et al. Recommendations on Queue Management and Congestion Avoidance in the Internet[S]. RFC2309, 1998.

[2] S Floyd, V Jacobson. Random early detection gateways for congestion avoidance[J]. IEEE/ACM Transactions on networking, 1993, 1(4): 397- 413.

[3] M Christiansen, K Jeffay, D Ott, et al. Tuning RED for web traffic[A]. Proc. of ACM SIGCOMM 2000[C]. Stockholm, Sweden, 2000. 139- 150.

[4] C V Hollot, V Misra, D F Towsley, et al. A control theoretic analysis of RED[A]. Proc. of INFOCOM2001[C]. Anchorage, Alaska, USA, 2001. 1510- 1519.

[5] 任丰原, 林闯, 王福豹. RED 算法的稳定性: 基于非线性控制理论的分析[J]. 计算机学报, 2002, 25(12): 1302- 1307.

[6] Eyad Abed, Priya Ranjan. Nonlinear instabilities in TCP2RED[A]. Proc. of INFOCOM2002[C]. San Francisco, USA, 2002. 249- 258.

[7] Teunis J Ott, T V Lakshman, et al. SRED: Stabilized RED[A]. Proc. of IEEE INFOCOM1999[C]. New York, USA, 1999. 1346- 1355.

[8] F Anjum, L Tassiulas. Fair bandwidth sharing among adaptive and non2adaptive flows in the internet[A]. Proc. of IEEE INFOCOM1999[C]. New York, USA, 1999. 1412- 1420.

[9] S Floyd, R Gummadi, S Shenker. Adaptive RED: An algorithm for increasing the robustness of RED. s active queue management[DB/OL]. http://www.icir.org/floyd/papers.html.

[10] W Feng, D Kandlur, D Saha, K Shin. Self2configuring RED gateway[A]. Pro. of IEEE INFOCOM 1999[C]. New York, USA, 1999. 1320 - 1328.

[11] U Bodin, O Schelen, S Pink. Load2tolerant differentiation with active queue management[A]. SIGCOMM Computer Communication Review[C]. USA, 2000. 30(3). 4- 19.

[12] S Floyd. Recommendation on using the / gentle. 0 variant of RED[DB/OL]. http://www.aciri.org/floyd/red/gentle.html.

[13] W Feng, D Kandlur, D Saha, et al. Stochastic fair blue: A queue management algorithm for enforcing fairness[A]. Proc. of IEEE INFOCOM2001[C]. Anchorage, Alaska, USA, 2001. 1520- 1529.

[14] S Athuraliya, D E Lapsley, S H Low. An enhanced random early marking algorithm for Internet flow control[A]. Proc. of IEEE INFOCOM2000[C]. Tel Aviv, Israel, 2000. 1425- 1434.

[15] C Hollot, V Misra, D Towsley, et al. On designing improved controllers for AQM routers supporting TCP flows[A]. Proc. of IEEE INFOCOM2001[C]. Anchorage, Alaska, USA, 2001. 1726- 1734.

[16] F Y Ren, Y Ren, X M Shan. Design of a fuzzy controller for active queue management[J]. Journal of Computer Communications, 2002, 25: 874- 883.

[17] F Y Ren, L Chuang, X H Yin. A robust active queue management algorithm based on sliding mode variable structure control[A]. Proc. of IEEE INFOCOM2002[C]. San Francisco, USA, 2002(1). 13- 20.

[18] S Kunniyur, R Srikant. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management[A]. Proc. of ACM SIGCOMM 2001[C]. San Diego, USA, 2001. 123- 134.

[19] R O Duda, P E Hart, D G Stork. Pattern Classification(2nd Edition)[M]. New York, USA: Wiley2Interscience Publication, 2000.

[20] D Katabi, M Handley, C Rohrs. Congestion control for future high bandwidth2delay product environments[A]. Proc. of ACM SIGCOMM2002[C]. Pittsburgh, USA, 2002. 89- 102.

[21] UCN/LBL/VINT. Network Simulator2NS2[DB/OL]. http://www2mash.cs.berkeley.edu/ns.

[22] RED parameters[DB/OL]. http://www.icir.org/floyd/red.html#parameters.

作者简介:

任丰原 男, 1970 年生于甘肃临洮, 清华大学计算机系讲师, 主要研究方向为网络流量的控制和管理, 传感器网络和测控网络等. Email: renfy@csnet1.cs.tsinghua.edu.cn.

林 闯 男, 1948 年生于辽宁省, 清华大学计算机系教授, 博士生导师, 主要研究方向为计算机网络, 系统性能评价, 随机 Petri 网, 逻辑推演和推理系统.