互联网端到端拥塞控制:研究综述*

章淼, 吴建平, 林闯

(清华大学 计算机科学与技术系 网络技术研究所,北京 100084)

E-mail: zm@csnet1.cs.tsinghua.edu.cn

http://netlab.cs.tsinghua.edu.cn

摘要: 随着互联网规模的增长,互连网上的用户和应用都在快速的增长,拥塞已经成为一个十分重要的问题.近年来,在拥塞控制领域开展了大量的研究工作.拥塞控制算法可以分为两个主要部分:在端系统上使用的源算法和在网络设备上使用的链路算法.在介绍拥塞控制算法的基本概念后,本文在源算法和链路算法两个方面总结了拥塞控制算法的研究现状,并分析了进一步的研究方向.

关 键 词: 互联网; 拥塞控制; 端到端

中图法分类号: TP393 **文献标识码:** A

1 引言

端到端拥塞控制是目前 Internet 的一个研究热点.在最初的 TCP 协议[1]中只有流控制(flow control)而没有拥塞控制,接收端利用 TCP 报头将接收能力通知发送端.这样的控制机制只考虑了接收端的接收能力,而没有考虑网络的传输能力,导致了网络崩溃(congestion collapse)的发生.1986 年 10 月,由于拥塞崩溃的发生,美国 LBL 到 UC Berkeley 的数据吞吐量从 32 Kbps 跌落到 40 bps[2].在那之后,在拥塞控制领域开展了大量的研究工作.拥塞控制算法对保证 Internet 的稳定具有十分重要的作用.

网络中的拥塞来源于网络资源和网络流量分布的不均衡性.拥塞不会随着网络处理能力的提高而消除.拥塞控制算法的分布性、网络的复杂性和对拥塞控制算法的性能要求又使拥塞控制算法的设计具有很高的难度.到目前为止,拥塞问题还没有得到很好的解决.

本文的组织如下:在第 2 部分中,介绍拥塞控制的基本概念,包括拥塞和拥塞控制的概念、Internet 的网络模型、Internet 中拥塞发生的原因;在第 3 部分中,介绍拥塞控制算法的概况,包括拥塞控制算法的评价方法、拥塞控制算法设计的困难和拥塞控制算法的研究概况;在第 4 部分中,介绍拥塞控制的源算法,包括“管子”模型、TCP 拥塞控制算法的发展和拥塞控制源算法的研究热点;在第 5 部分中,围绕“主动队列管理”算法介绍拥塞控制的链路算法,包括“主动队列管理”算法的研究概况、“主动队列管理”算法的发展、网络的流量特征对“主动队列管理”算法的影响和“主动队列管理”算法的反馈方式.第 6 部分对全文进行总结.

2 基本概念

2.1 拥塞和拥塞控制

当在网络中存在过多的报文时,网络的性能会下降,这种现象称为拥塞[3].[4]使用图 1 来描述拥塞的发生.

* 收稿日期: 2001-10-18; 修改日期: 2001-12-18

基金项目: 国家自然科学基金资助项目(90104002);国家重点基础研究发展规划 973 资助项目(G1999032707)

作者简介: 章淼(1976－),男,浙江缙云人,博士生,主要研究领域为计算机网络体系结构;吴建平(1953－),男,山东巨野人,博士,教授,博士生导师,主要研究领域为计算机网络体系结构,协议工程学,互联网络; 林闯(1948－),男,辽宁沈阳人,博士,教授,博士生导师,主要研究领域为计算机网络,系统性能评价,随机 Petri 网.

当负载较小时,吞吐量的增长和负载相比基本呈线性关系,延迟增长缓慢;在负载超过 Knee 之后,吞吐量增长缓慢,延迟增长较快;当负载超过 Cliff 之后,吞吐量急剧下降,延迟急剧上升.可以看出,负载在 Knee 附近时网络的使用效率最高.拥塞控制就是网络节点采取措施来避免拥塞的发生或者对拥塞的发生做出反应[5].在图 1 中就是使负载保持在 Knee 附近.和流控制相比,拥塞控制主要考虑端节点之间的网络环境,目的是使负载不超过网络的传送能力;而流控制主要考虑接收端,目的是使发送端的发送速率不超过接收端的接收能力.

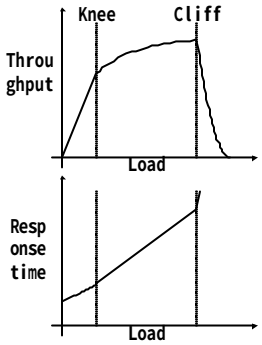

Fig. 1
图 1

拥塞控制算法包含拥塞避免(congestion avoidance)和拥塞控制(congestion control)这两种不同的机制[4].拥塞控制是“恢复”机制,它用于把网络从拥塞状态中恢复出来;拥塞避免是“预防”机制,它的目标是避免网络进入拥塞状态,使网络运行在高吞吐量、低延迟的状态下.

2.2 Internet的网络模型

拥塞现象的发生和 Internet 的设计机制有着密切的联系.Internet 的网络模型可以用以下几点来抽象[5]:

(1) 报文交换(packet-switched)网络.和电路交换(circuit-switched)相比,报文交换通过共享提高了资源的利用效率.但在共享方式下,如何保证用户的服务质量是一个很棘手的问题;在报文交换网络中可能出现报文“乱序”现象[6],对乱序报文的处理增加了端系统的复杂性.

(2) 无连接(connectionless)网络.Internet 的节点之间在发送数据之前不需要建立连接.无连接模型简化了网络的设计,在网络的中间节点上不需要保存和连接有关的状态信息.但是使用无连接模型难以引入“接纳控制”(admission control)算法,在用户需求大于网络资源时难以保证服务质量;在无连接模型中对数据发送源的追踪能力很差,给网络的安全带来了隐患;无连接也是网络中乱序报文出现的一个主要原因.

(3) best-effort 的服务模型.best-effort 即网络不对数据传输的服务质量提供保证.这个选择和早期网络中的应用有关[7].传统的网络应用主要是 FTP、Telnet、SMTP 等,它们对网络性能(带宽、延迟、丢失率等)的变化不敏感,best-effort 模型可以满足需要.但 best-effort 模型不能很好满足新出现的多媒体应用的要求,这些应用对延迟、速率等性能的变化比较敏感.这要求网络在原有服务模型的基础上进行扩充.

2.3 Internet中拥塞发生的原因

拥塞发生的原因是“需求”大于“供给”.网络中有限的资源由多个用户共享使用.由于没有“接纳控制”算法,网络无法根据资源的情况限制用户的数量;缺乏中央控制,网络也无法控制用户使用资源的数量.目前 Internet 上用户和应用的数量都在迅速增长.如果不使用某种机制协调资源的使用,必然会导致网络拥塞.虽然拥塞源于资源短缺,但增加资源并不能避免拥塞的发生,有时甚至会加重拥塞程度[8].例如:增加网关缓冲会增大报文通过网关的延迟,如果总延迟超过端系统重传时钟的值,就会导致报文重传,反而加重了拥塞.

拥塞总是发生在网络中资源“相对”短缺的位置.拥塞发生位置的不均衡反映了 Internet 的不均衡性.首先是资源分布的不均衡.图 2(a)[8]中带宽的分布是不均衡的,当以 1Mb/s 的速率从 S 向 D 发送数据时,在网关 R 会发生拥塞.其次是流量分布的不均衡.图 2(b)[8]中带宽的分布是均衡的,当 A 和 B 都以 1Mb/s 的速率向 C 发送数据时,在网关 R 也会发生拥塞.Internet 中资源和流量分布的不均衡都是广泛存在的,由此导致的拥塞不能使用增加资源的方法来解决.

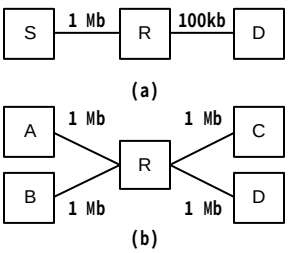

Fig. 2
图 2

3 拥塞控制算法的概况

3.1 拥塞控制算法设计的困难

拥塞控制算法的设计困难体现在以下方面[8][9]:

(1) 算法的分布性.拥塞控制算法的实现分布在多个网络节点中,必须使用不完整的信息完成控制,并使各节点协调工作,还必须考虑某些节点工作不

正常的情况.

(2) 网络环境的复杂性.Internet 中各处的网络性能有很大的差异,算法必须具有很好的适应性;另外由于 Internet 对报文的正确传输不提供保证,算法必须处理报文丢失、乱序到达等情况.

(3) 算法的性能要求.拥塞控制算法对性能有很高的要求,包括算法的公平性、效率、稳定性和收敛性等.某些性能目标之间存在矛盾,在算法设计时需要进行权衡.

(4) 算法的开销.拥塞控制算法必须尽量减少附加的网络流量,特别是在拥塞发生时.在使用反馈式的控制机制时,这个要求增加了算法设计的困难.算法还必须尽量降低在网络节点(特别是网关)上的计算复杂性.目前的策略是将大部分计算放在端节点完成,在网关上只进行少量的操作,这符合 Internet 的基本设计思想[10].

3.2 拥塞控制算法的评价方法

在设计和比较拥塞控制算法时,需要一定的评价方法.从用户的角度出发,可以比较端系统的吞吐率、丢失率和延迟等指标,这些是用户所关心的.由于拥塞控制算法对整个网络系统都有影响,在评价算法时更应该从整个系统的角度出发进行考虑.两个重要的评价指标是资源分配的效率和资源分配的公平性.

3.2.1 资源分配的效率

资源分配的效率可以用 Power 函数[4][5]来评价.Power 函数定义为:

$$\text{Power} = \text{Throughput}^{a} / \text{Response Time}$$

在上式中,一般取 a=1.如果评价偏重吞吐量,则取 a>1;如果评价偏重反应时间,则取 a<1.图 3[4]示意当负载位于 Knee 时 Power 取最大值.使用 Power 函数有一定的局限性.它主要基于 M/M/1 队列的网络,并假设队列的长度为无穷.Power 函数一般在单资源、单用户的情况下使用[4].

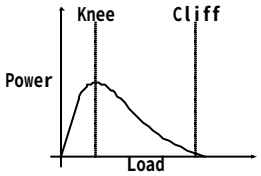

Fig. 3
图 3

3.2.2 资源分配的公平性

多用户情况下需要考虑资源分配的公平性.公平性评价的主要方法包括 Max-Min Fairness[11],Fairness Index[12]和 Proportional Fairness[13]等.

Max-min fairness 被非正式的定义为:每个用户的吞吐量至少和其它共享相同瓶颈的用户的吞吐量相同.Max-Min Fairness 是一种理想的状况,但是它不能给出公平的程度.

Fairness Index 提供了一个计算公式,可以计算公平的程度.它定义为:

$$F(x) = \frac{(\sum x_i)^2}{n(\sum x_i^2)}$$

Fairness Index 的计算结果位于 0 和 1 之间,并且结果不受衡量单位的影响.它的一个性质是:如果 n 个用户中只有 k 个用户平均共享资源,而另 $(n - k)$ 个用户没有任何资源,计算结果为 k/n.

一些研究者认为,如果考虑用户的“效用函数”(utility function),在一些情况下使用 Max-Min Fairness 评价并不是最理想的.针对对数的效用函数,[13]引入了 Proportional Fairness 的概念.Proportional Fairness 定义为:向量 X 满足 Proportional Fairness,如果对于其它任何向量 Y 都满足:

$$\sum_{i \in I} \frac{y_i - x_i}{x_i} \le 0$$

由于公平性是针对资源分配而言的,所以在评价前首先要确定“资源”的含义.目前大多数研究在评价公平性时都针对吞吐量,这是从用户的角度出发考虑的,并不完全适合网络中的资源状况.网络中的资源包括链路带宽、网关的缓冲和网关的处理能力等,在考察公平性时应当将这些资源的分配情况综合考虑.

3.3 拥塞控制算法的研究概况

从控制理论的角度,拥塞控制算法可以分为开环控制和闭环控制两大类[14].当流量特征可以准确规定、性能要求可以事先获得时,适于使用开环控制;当流量特征不能准确描述或者当系统不提供资源预留时,适于使用

闭环控制.Internet 中主要采用闭环控制方式.

闭环的拥塞控制分为以下三个阶段[3]:检测网络中拥塞的发生;将拥塞信息报告到拥塞控制点;拥塞控制点根据拥塞信息进行调整以消除拥塞.闭环的拥塞控制可以动态适应网络的变化,但它的一个缺陷是算法性能受到反馈延迟的严重影响[14].当拥塞发生点和控制点之间的延迟很大时,算法性能会严重下降.

根据算法的实现位置,可以将拥塞控制算法分为两大类:链路算法(link algorithm)和源算法(source algorithm)[15].链路算法在网络设备(如路由器和交换机)中执行,作用是检测网络拥塞的发生,产生拥塞反馈信息;源算法在主机和网络边缘设备中执行,作用是根据反馈信息调整发送速率.拥塞控制算法设计的关键问题是如何生成反馈信息和如何对反馈信息进行响应.

源算法中使用最广泛的是 TCP 协议中的拥塞控制算法.TCP 是目前在 Internet 中使用最广泛的传输协议.根据 MCI 的统计,总字节数的 95%和总报文数的 90%使用 TCP 传输[16].近年来 TCP 中采用了很多新的算法,包括慢启动(slow start)、拥塞避免、快速重传(fast retransmit)、快速恢复(fast recovery)、选择性应答(SACK)等,大大提高了网络传输的性能.TCP 中使用的拥塞控制算法已经成为保证 Internet 稳定性的重要因素[17].

链路算法的研究目前集中在“主动队列管理”(active queue management, AQM)算法方面.和传统的“队尾丢弃”(DropTail)相比,AQM 在网络设备的缓冲溢出之前就丢弃或标记报文[18].AQM 的主要优点是:

(1) 减少网关的报文丢失.使用 AQM 可以保持较小的队列长度,从而增强网关容纳突发流量的能力.

(2) 减小报文通过网关的延迟.减小平均队列长度可以有效的减小报文在网络设备中的排队延迟.

(3) 避免 lock-out 行为[19]的发生.

AQM 的一个代表是 RED(random early detection)[20].研究表明 RED 比 DropTail 具有更好的性能.但是 RED 的性能对算法的参数设置十分敏感,至今没有在 Internet 中没有得到广泛的使用.

4 拥塞控制的源算法

4.1 “管子”模型

在拥塞控制源算法的研究中,使用“管子”模型来抽象两个端节点之间的网络.“管子”的性能(包括带宽、延迟、丢失率等)在一定时间范围内是相对稳定的,这样端系统可以使用历史信息来估计未来的情况.“管子”模型的一个重要特性是“报文守恒”(packet conservation)[2],控制的目标是使网络中报文的个数保持恒定;在报文离开网络前,不向网络中加入新的报文.

使用“管子”模型要求端节点之间的网络性能是相对稳定的.[21][22][23]指出端系统之间的网络性能在分钟量级上是相对稳定的.另一个因素是路由的稳定性.[24]指出端节点之间的路由是相对稳定的,测试数据中保持 10 分钟以上的路由超过 95%,其中 68%的路由保持 1 天以上.以上都说明使用“管子”模型是合适的.

4.2 TCP拥塞控制算法的发展

下面通过 TCP 的发展说明算法的演进.

(1) TCP Tahoe.Tahoe 是 TCP 的早期版本,包括 3 个最基本的拥塞控制算法:“慢启动”,“拥塞避免”和“快速重传”[2][25].“快速重传”根据 3 个重复的应答报文来判断报文的丢失,减少了超时重传的发生.

(2) TCP Reno.Reno 在 Tahoe 的基础上增加了“快速恢复”[25].“快速恢复”使用“管子”模型的“报文守恒”特性.发送方每收到一个重复的应答,就认为已经有一个报文离开网络,于是将发送方的拥塞窗口加一.

(3) TCP NewReno[26].NewReno 对 Reno 中“快速恢复”算法进行了补充.它考虑了一个发送窗口内多个报文丢失的情况.在“快速恢复”算法中,发送方收到一个不重复的应答后就退出“快速恢复”状态.而在 NewReno 中,只有当所有报文都被应答后才退出“快速恢复”状态.

(4) SACK [27].SACK 也关注一个窗口内多个报文的丢失,它使用“选择性重复”(selective repeat)[3]策略.

从以上算法中,可以总结出 TCP 拥塞控制的几个特点:

(1) 将拥塞控制分为“慢启动”和“拥塞避免”两个阶段.“慢启动”用于探测网络的带宽,使用指数增长的方式;“拥塞避免”试图避免拥塞的发生,使用 AIMD(additive increase multiplicative decrease)[12]方式.

(2) 假设报文的丢失由网络拥塞引起.TCP 算法用报文丢失判断拥塞的发生.“快速重传”和 SACK 都可以检测报文的丢失,但当这些机制失效时,“重传时钟”超时是发现报文丢失的最终机制.

(3) 从解决一个发送窗口内单个报文丢失到解决多个报文的丢失.

4.3 拥塞控制源算法的研究热点

目前拥塞控制源算法的研究热点包括[28]:

(1) “慢启动”过程的改进[29][30].“慢启动”阶段对短数据传输更重要,“拥塞避免”阶段对长数据传输更重要.目前 Internet 上的一个主要应用—HTTP 的流量主要是短数据.这方面的研究包括:增大拥塞窗口的初始值,[31]推荐将初始拥塞窗口的值由 1 MSS(maximum segment size)增加到 4 MSS;将“慢启动”过程分为多段,逐步减小窗口增长的速度,平滑从“慢启动”到“拥塞避免”的过渡[32];在多个连接或主机间共享网络状况信息,根据网络的状况决定初始的阈值,相关研究包括 SPAND[21]和“TCP Fast Start”[33].

(2) 基于速率的控制策略[34][35].TCP 使用的窗口控制策略有一些缺陷:容易导致报文的突发;速率受到窗口大小的限制;一个窗口内多个报文的丢失不容易恢复等.为此一些研究者提出 RBP (rate-based pacing),将窗口控制和速率控制结合起来以克服以上缺陷.但[36]对 RBP 的效果提出了质疑,指出拥塞发生时 RBP 会导致多个连接同时丢失报文,从而出现“全局同步”(global synchronization)现象,严重降低网络的吞吐量.另外 RBP 的实现需要大量高精度的时钟,消耗资源较多.

(3) ACK 过滤(ACK filter).它的目的是保持 TCP 的“自时钟”(self clocking)机制[2].“自时钟”机制可减轻突发报文对网络的冲击,而“ACK 压缩”(ACK compression)[37]破坏了“自时钟”机制.[38]建议在网络中增加 PEP(performance enhance proxy)来确保 ACK 报文之间的间隔.

(4) 减少不必要的“超时重传”和“快速重传”.这些重传发生时,TCP 都会减小拥塞窗口,从而降低传输的速率.RTT(round trip time)测量的准确性和乱序报文都会影响 TCP 做出正确的判断. Eifel 算法[39]通过在应答报文中增加特殊信息来解决这个问题.

(5) ECN(explicit congestion notification)的使用[40][41].端系统通过报文中网关设定的标志位检测拥塞,而不完全依赖报文丢失来判断拥塞的发生.在无线网络环境中,ECN 的使用有利于将报文损坏(packet corruption)和拥塞导致的报文丢失区分开.

(6) TCP-Friendly 的拥塞控制[42][43].很多多媒体的传输要求速率不能剧烈变化,但是也要对网络中的拥塞做出反应,并和 TCP 的传输保持公平.TCP-Friendly 定义为[17]:长时间的吞吐量不超过相同条件下 TCP 连接的吞吐量.针对这个需求提出了 TCP-Friendly 的拥塞控制.它使用基于速率的控制,速率的计算建立在 TCP 吞吐量模型[44]的基础上.

(7) 特殊网络环境中的拥塞控制.包括 TCP 在无线链路[45]、卫星链路[46]和“非对称”链路(asymmetric link)[47]上的拥塞控制.这些链路的特点为高延迟、高丢失率、两个方向传输链路的性能不相同,这些环境中的拥塞控制算法需要增加特殊的考虑.

5 拥塞控制的链路算法

5.1 AQM 算法的研究概况

RED 是最著名 AQM 算法.它通过以一定概率丢失或标记报文通知端系统网络的拥塞情况.RED 使用平均队列长度度量网络的拥塞程度,然后以线性方式将拥塞信息反馈给端系统.RED 使用最小阈值(min_{th})、最大阈值(max_{th})和最大概率(max_p)等几个参数,概率的计算可以用图 4 来表示.模拟结果显示 RED 的性能优于 DropTail[20],但是它存在两个主要缺陷[48][15].RED 对参数设置很敏感,改变参数对性能影响很大.到目前为止,这些参数还没有明确的设定方法.另一个问题是随着网络中“流”(Flow,指一个 TCP 连接)数目的增加,网关的平均队列长度会逐渐增加.一些研究者

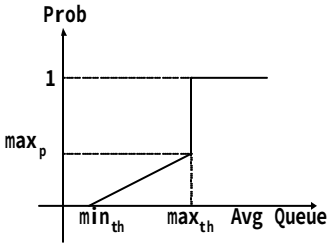

Fig. 4
图 4

使用真实实现对 RED 进行试验[49][50],也反映出相同的问题.

针对第二个问题一些研究提出了解决方案,包括 ARED[51]、SRED[52]和 BLUE[53]等.它们的主要思路是根据网络中负载的情况对标记/丢失概率进行动态调整.ARED 基于 RED 算法,其主要思想是根据网络负载的情况调整 max_p.当平均队列小于 min_{th},就减小 max_p;当平均队列大于 max_{th},就增大 max_p.在 ARED 中,不需要使用"单流信息"(per-flow information).SRED 的主要思想是通过估计网络中流的个数来调整报文标记/丢失概率.在 SRED 中流的个数通过概率统计的方法获得,所以也不需要使用"单流信息".BLUE 的主要思想是通过链路空闲和缓冲溢出的状况来调整报文标记/丢失概率.如果缓冲溢出,就增大概率;如果线路空闲,就减小概率.

最近,在 AQM 算法方面又提出了 REM(Random Early Marking)[48]、PI[54]和 AVQ(Adaptive Virtual Queue)[55]等.它们的基本思想都是在 AQM 中使用 PI 控制器[56].相关的内容将在 5.2 中进行介绍.

5.2 AQM算法的发展

这里我们以 DropTail、RED 和 PI 为例分析 AQM 算法的发展.

在 DropTail 中,使用 ON/OFF 控制器来生成反馈,反馈的值只能为 0 或者 1.DropTail 使用常数 B(B 是缓存大小)来判断拥塞的发生:

$$p_i = \begin{cases} 0, & \text{if } q_i \le B \\ 1, & \text{if } q_i > B \end{cases}$$

其中 p_i 是报文标记/丢失概率,q_i 是队列长度.DropTail 的最大问题是反馈变化比较猛烈,容易造成系统振荡.

RED 使用"Proportional 控制器+低通滤波器"的方法计算反馈.Proportional 控制器的基本形式为:

$$p_i = kq_i$$

其中 k 是计算使用的比例系数.为了减小"瞬时抖动"对反馈计算的影响,RED 引入了低通滤波器,将上式中的 q_i 使用平均队列长度 q_{ave} 代替.平均队列长度使用 EWMA(Exponential Weighted Moving Average)计算:

$$q_{ave} = (1-w)q_{ave} + wq_i$$

其中 w 是计算的权重.Proportional 控制器的优点是实现简单、反应速度快,它的缺点是控制存在"稳态误差"(steady-state error),这是平均队列随网络流量增长的主要原因.另外 RED 使用低通滤波器降低了系统的反应速度;固定的 w 不能适应不同速率的网络链路.

在 PI 中使用了 PI 控制器.PI 控制器的形式为:

$$p_{i+1} = p_i + a(q_{i+1} - q^*) + b\Delta q_{i+1}$$
$$\Delta q_{i+1} = q_{i+1} - q_i$$

其中 q^*是控制的目标,a 和 b 是比例系数.通过在反馈计算中引入积分项,PI 控制器可以有效的消除 Proportional 控制器中出现的"稳态误差",从而保证队列长度的稳定.[54]给出了确定 PI 控制器参数的方法.REM 和 AVQ 中也使用了 PI 控制器.在 AVQ 中被控制的对象是队列的入口速率,而不是队列长度.REM 除了使用 PI 控制器外还引入了指数函数,但是并没有明确说明使用指数函数的优势.在 AQM 算法中使用 PI 控制器虽然可以消除"稳态误差",但同时也会减慢系统的反应速度.当网络中的流量发生很大变化时,使用 PI 控制器需要的收敛时间要远远长于使用 Proportional 控制器的情况.这是 PI、REM 和 AVQ 都存在的问题.

5.3 网络流量特征对AQM算法的影响

网络流量特征对 AQM 算法的设计和有效性有很重要的意义.大部分 AQM 算法使用模拟器(如 ns[57])进行验证,模拟时多数采用 FTP 流量,这并不符合网络实际测量得到的结果.[16]的测量结果中,75%的字节、70%的报文和 75%的"流"是 Web 流量,FTP 只占 5%的字节、3%的报文和小于 1%的"流"."流"是网络资源的用户,拥塞控制算法负责在多个"流"之间分配资源.在平衡状态下,如果网络中"流"的数量增加或者资源的数量减少,在"流"之间必须重新进行资源分配,在重分配完成之前会暂时出现"需求"大于"供给",从而发生

拥塞.假设网络资源的数量是相对稳定的,如果“流”的数量不断剧烈的变化,就会不断出现拥塞.这要求 AQM 算法具有很好的反应速度.目前,Internet 网络流量特征的研究还不够充分.[58]提出 Internet 中“流”的到达比泊松分布更具有突发性和不关联性.这种流量特征对 AQM 算法的设计和有效性是一个很大的挑战.

5.4 AQM算法的反馈方式

AQM 计算出反馈大小后,需要将反馈传递给端系统.网关可以采用的反馈方式包括“丢弃”(dropping)和“标记”(marking).“丢弃”是所有网关都支持的操作,传统的 TCP 算法只使用报文丢失作为拥塞发生的指示.“标记”方式“显式”的通知端系统,可以避免 TCP 发送端调用超时处理.试验结果显示“标记”比“丢弃”具有更好的性能[48].“标记”方式的缺点是要求网关提供特殊支持,但随着 ECN 的标准化和广泛采用[59][60],这个问题将得到解决.“源抑制”(source quench,下面用 SQ 表示)[61]也是反馈的一种方式.当网络流量超过网关的处理能力,网关在丢弃报文的同时可以向数据源发送 SQ 报文,由数据源对发送速率进行调整.直观上看 SQ 和 ECN 相比可以提供更快的反馈.[41]和[62]中对 SQ 和 ECN 进行了比较,最近有关这个问题也有讨论[63].ECN 在以下方面优于 SQ: (1) 拥塞发生时 ECN 不增加网络的负担,SQ 会增加反方向网络链路的流量; (2) SQ 在 ICMP 报文处理方面会给网关增加太多的负担; (3) 报文在 ECN 方式中只是被“标记”,可减少报文的丢失,并可以避免发送方等待超时重传,这对于短连接更为重要; (4) ECN 对“基于发送端”(sender-based)和“基于接收端”(receiver-based)的拥塞控制算法都适用,而 SQ 只能用于“基于发送端”的拥塞控制算法.总之以目前的理解,ECN 和 SQ 相比是一个更好的选择.

6 总结

本文从源算法和链路算法两个方面对互联网端到端拥塞控制的研究现状进行了总结.本文在介绍研究现状的同时,讨论了拥塞控制领域的研究热点.这些研究热点可以总结为以下几大类:

(1) 对 TCP 协议本身的改进,包括对 TCP 中各种机制的改进和对 TCP 在各种网络环境下优化的研究.

(2) 对多媒体传输拥塞控制的研究,包括传输算法的设计和传输公平性的研究.

(3) 对“主动队列管理”算法的研究.

近些年来,非线性规划理论[13][64]和系统控制理论[54]被引入到拥塞控制的研究中来,一些研究者尝试使用严格的数学模型来描述端系统和网关组成的系统.这对拥塞控制的研究有很大的推动作用.

网络流量特征对拥塞控制中链路算法的设计和成功有很重要的意义.对实际网络流量的测量和分析还有待于进一步深化.

致谢 徐明伟博士、徐恪博士、盛立杰同志和《软件学报》的审稿人对本文的完成提出了很多有益的建议,在此一并表示感谢.

References:

[1] Postel, J. Transmission Control Protocol. RFC 793, 1981.

[2] Jacobson, V. Congestion Avoidance and Control. ACM Computer Communication Review, 1988, 18(4): 314~329.

[3] Tanenbaum, A.S. Computer Networks. 3rd ed., Prentice Hall, Inc., 1996.

[4] Jain, R., Ramakrishnan, K.K., Chiu, Dah-Ming. Congestion Avoidance in Computer Networks with a Connectionless Network Layer. Technical Report, Digital Equipment Corporation DEC-TR-506, 1988. http://www.cis.ohio-state.edu/~jain.

[5] Peterson, L.L., Davie, B.S. Computer Networks: A System Approach, Morgan Kaufmann Publishers, 2000.

[6] Bennett, J.C.R., Partridge, C., Shectman, N. Packet Reordering is Not Pathological Network Behavior. IEEE/ACM Transactions on Networking, 1999, 7(6): 789~798.

[7] Shenker, S. Fundamental Design Issues for the Future Internet. IEEE Journal on Selected Areas in Communications, 1995, 13(7): 1176~1188.

[8] Jain, R. Congestion Control in Computer Networks: Issues and Trends. IEEE Network Magazine, 1990, 4(3): 24~30.

[9] Balakrishnan, H. M.I.T.6.899 Computer Networks (Fall 2000). Tutorial Slides, 2000. http://nms.lcs.mit.edu/6.899/.

[10] Saltzer, J., Reed, D., Clark, D. End-to-end Arguments in System Design. ACM Transactions on Computer Systems, 1984, 2(4): 195~206.

[11] Jaffe, J. M. Bottleneck Flow Control. IEEE Transactions on Communication, 1981, 29(7): 954~962.

[12] Chiu, Dah-Ming, Jain, R. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. Computer Networks and ISDN Systems, 1989, 17(1): 1~14.

[13] Kelly, F.P., Maulloo, A., Tan, D. Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability. Journal of Operations Research Society, 1998, 49(3): 237~252.

[14] Kalampoukas, L. Congestion Management in High Speed Networks [Ph.D. Thesis], UCSC, 1997.

[15] Low, S.H. TCP Congestion Controls: Algorithms and Models. Tutorial Slides, 2000. http://netlab.caltech.edu.

[16] Thompson, K., Miller, G.J., Wilder, R. Wide-Area Internet Traffic Patterns and Characteristics. IEEE Network, 1997, 11(6): 10~23.

[17] Floyd, S., Fall, K. Promoting the Use of End-to-End Congestion Control in the Internet. IEEE/ACM Transactions on Networking, 1999, 7(4): 458~472.

[18] Braden, B., Clark, D., Crowcroft, J., et al. Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC 2309, 1994.

[19] Floyd, S., Jacobson, V. On Traffic Phase Effects in Packet-Switched Gateways. Internetworking: Research and Experience, 1992, 3(3): 115~156.

[20] Floyd, S., Jacobson, V. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, 1993, 1(4): 397~413.

[21] Seshan, S., Stemm, M., Katz, R.H. Network Measurement Architecture for Adaptive Applications. In: Sidi, M. ed. Proceedings of IEEE INFOCOM. Tel Avlv, Israel: IEEE Communications Society, 2000. 285~294.

[22] Balakrishman, H., Seshan, S., Stemm, M., Katz, R.H. Analyzing Stability in Wide-Area Network Performance. Performance Evaluation Review, 1997, 25(1): 2~12.

[23] Paxson, V. Measurements and Analysis of End-to-End Internet Dynamics [Ph.D. Thesis], U.C. Berkeley, 1996.

[24] Paxson, V. End-to-End Routing Behavior in the Internet. IEEE/ACM Transactions on Networking, 1997, 5(5): 601~615.

[25] Stevens, W. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. RFC 2001, 1997.

[26] Floyd, S., Henderson, T. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC2582, 1999.

[27] Mathis, M., Mahdavi, J., Floyd, S., Romanow, A. TCP Selective Acknowledgment Options. RFC 2018, 1996.

[28] Floyd, S. A Report on Some Recent Developments in TCP Congestion Control. IEEE Communication Magazine, 2001, 39(4): 84~90.

[29] Hoe, J.C. Improving the Start-up Behavior of a Congestion Control Scheme for TCP. Computer Communication Review, 1996, 26(4): 270~280.

[30] Zhang, Y., Qiu, L., Keshav, S. Optimizing TCP Start-up Performance. Technical Report, Cornell, 1999. http://www.cs.cornell.edu/cnrg/papers.html.

[31] Allman, M., Floyd, S., Partridge, C. Increasing TCP's Initial Window. RFC 2414, 1998.

[32] Wang, H, Xin, H., Reeves, D.S., Shin, K.G. A Simple Refinement of Slow-start of TCP Congestion Control. In: Sherif, S.H. ed. Proceedings of IEEE Symposium on Computers and Communications (ISCC'00). Los Alamitos, California, USA: IEEE Computer Society, 2000. 98~105.

[33] Padmanabhan, V.N. Addressing the Challenges of Web Data Transport [Ph.D. Thesis], U.C. Berkeley, 1998.

[34] Visweswaraiah, V, Heidemann, J. Improving Restart of Idle TCP Connections. Technical Report, USC TR 97-661, 1997. http://www.isi.edu/~johnh/papers/.

[35] Ke, J., Willianmson, C. Towards a Rate-Based TCP Protocol for the Web. In: Boukerche, A. ed. Proceedings of IEEE International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'00). San Fransisco, California, USA: IEEE Computer Society, 2000. 36~45.

[36] Aggarwal, A., Savage, S., Anderson, T., Understanding the Performance of TCP Pacing. In: Sidi, M. ed. Proceedings of IEEE INFOCOM. Tel Avlv, Israel: IEEE Communications Society, 2000. 1157~1165.

[37] Floyd, S. Connections with Multiple Congested Gateways in Packet-Switched Networks Part 2: Two-way Traffic. Unpublished Draft, 1991. http://www.aciri.org/floyd.

[38] Border, J., Kojo, M., Griner, J., Montenegro, G., Shelby, Z. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135, 2001.

[39] Ludwig, R., Katz, R.H. The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions. ACM Computer Communication Review, 2000, 30(1): 23~36.

[40] Ramakrishnan, K.K., Jain, R. A Binary Feedback Scheme for Congestion Avoidance in Computer Networks. ACM Transactions on Computer Systems, 1990, 8(2): 158~181.

[41] Floyd, S. TCP and Explicit Congestion Notification. ACM Computer Communication Review, 1994, 24(5): 10-23.

[42] Floyd, S., Handley, M., Padhye, J., Widmer, J. Equation-Based Congestion Control for Unicast Applications. ACM Computer Communication Review, 2000, 30(4): 43~56.

[43] Widmer, J., Denda, R., Mauve, M. A Survey on TCP-Friendly Congestion Control. IEEE Network, 2001, 15(3): 28~37.

[44] Padhye, J., Firoiu, V., Towsley, D.F., Kurose, J.F. Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation. IEEE/ACM Transactions on Networking, 2000, 8(2): 133~45.

[45] Balakrishnan, H., Padmanabhan, V.N., Seshan, S., Katz, R.H. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. IEEE/ACM Transactions on Networking, 1997, 5(6): 756~769.

[46] Partridge, C., Shepard, T. J. TCP/IP Performance over Satellite Links. IEEE Network, 1997, 11(5): 44~49.

[47] Kalampoukas, L., Varma, A., Ramakrishnan, K.K. Improving TCP Throughput over Two-Way Asymmetric Link: Analysis and Solutions. Performance Evaluation Review, 1998, 26(1): 78~89.

[48] Athuraliya, S., Li, V.H., Low, S.H., Yin, Q. REM: Active Queue Management. IEEE Network, 2001, 15(3): 48~53.

[49] Christiansen, M., Jeffay, K., Ott, D., Smith, F.D. Tuning RED for Web Traffic. ACM Computer Communication Review, 2000, 30(4): 139~150.

[50] Diot, C., Iannaccone, G., May, M. Aggregate Traffic Performance with Active Queue Management and Drop from Tail. Technical Report, Sprint ATL TR01-ATL-012501, 2001. http://www.sprintlabs.com/People/diot/publications.html.

[51] Feng, W., Kandlur, D., Saha, D., Shin, K. A Self-Configuring RED Gateway. In: Doshi, B. ed. Proceedings of IEEE INFOCOM. New York, USA: IEEE Communications Society, 1999. 1320~1328.

[52] Ott, T.J., Lakshman, T.V., Wong, L.H. SRED: Stabilized RED. In: Doshi, B. ed. Proceedings of IEEE INFOCOM. New York, USA: IEEE Communications Society, 1999. 1346~1355.

[53] Feng, W., Kandlur, D., Saha, D., Shin, K. Blue: A New Class of Active Queue Management Algorithms. Technical Report, U. Michigan CSE-TR-387-99, 1999. http://www.eecs.umich.edu/~wuchang/blue/.

[54] Hollot, C.V., Misra, V., Towsley, D., Gong, W. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In: Sengupta, B. ed. Proceedings of IEEE INFOCOM. Anchorage, Alaska , USA: IEEE Communications Society, 2001. 1726~1734.

[55] Kunniyur, S., Srikant, R. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. ACM Computer Communication Review, 2001, 31(4): 123~134.

[56] Franklin, G.F., Powell, J.D., Emami-Naeini, A. Feedback Control of Dynamic Systems, Addison-Wesley, 1995.

[57] UCB/LBNL/VINT Network Simulator. http://www.isi.edu/nsnam/ns.

[58] Ryu, R., Cheney, D., Braun, H.W. Internet Flow Characterization: Adaptive Timeout Strategy and Statistical Modeling. In: Uijterwaal, H. ed. Proceedings of PAM'2001, Amsterdam, Netherlands: RIPE NCC, 2001. http://www.ripe.net/pam2001/.

[59] Ramakrishnan, K.K., Floyd, S. A Proposal to add Explicit Congestion Notification (ECN) to IP. RFC 2481, 1999.

[60] Ramakrishnan, K.K., Floyd, S., Black, D., The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, 2001.

[61] Postel, J., Internet Control Message Protocol. RFC 792, 1981.

[62] Floyd, S., Personal Notes on ECN vs. Source Quench, 1998. http://www.aciri.org/floyd/ecn.html.

[63] Archive of end2end-interest mailing list, 2001. http://www.postel.org/pipermail/end2end-interest/.

[64] Low, S.H., Lapsley, D.E. Optimization Flow Control, I: Basic Algorithm and Convergence. IEEE/ACM Transactions on Networking, 1999, 7(6): 861-875.

Internet End-to-End Congestion Control: A Survey*

ZHANG Miao, WU Jian-ping, LIN Chuang

(*Institute of Computer Network Technology, Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

E-mail: zm@csnet1.cs.tsinghua.edu.cn

http://netlab.cs.tsinghua.edu.cn

Abstract: With the evolvement of the Internet, the number of users and applications using Internet increases very quickly. Congestion has become an important issue. To keep the stability of the whole network, congestion control algorithms have been extensively studied. There are two primary components in congestion control: one is the source algorithm executed by host computers and edge devices, the other is the link algorithm executed by network devices. After introducing the basic concepts of congestion control algorithm, this paper summarizes the research work on source algorithm and link algorithm of end-to-end congestion control. Research directions and open problems in this area are also discussed.

Key words: Internet; congestion control; end-to-end

* Received October 18, 2001; accepted December 18, 2001

Supported by the National Natural Science Foundation of China under Grant Nos. 90104002; the National Grand Fundamental Research 973 Program of China under Grant No. G1999032707