计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2019.20180752
56(9): 1965-1976, 2019

基于 DASH 流媒体的 TCP 拥塞控制算法优化

吴 桦 王 凌 程 光
(东南大学网络空间安全学院 南京 211189)
(计算机网络和信息集成教育部重点实验室(东南大学) 南京 211189)
(101005557@seu.edu.cn)

Optimization of TCP Congestion Control Algorithm in Dynamic Adaptive Streaming over HTTP

Wu Hua, Wang Ling, and Cheng Guang
(*School of Cyber Science and Engineering*, *Southeast University*, *Nanjing* 211189)
(*Key Laboratory of Computer Network and Information Integration* (*Southeast University*), *Ministry of Education*, *Nanjing* 211189)

Abstract With the rapid growth of Internet video traffic, streaming media transmission technologies are also developed rapidly. From the traditional real-time transport protocol (RTP) over user datagram protocol (UDP) to hyper text transfer protocol (HTTP) over transmission control protocol (TCP), major video service providers are constantly developing new media transmission technologies to attract more users. As one of the most popular adaptive streaming media transmission technologies, dynamic adaptive streaming over HTTP (DASH) has many advantages in improving users' experience. However, due to its fragmented transmission, the resulting ON-OFF transmission mode causes bursts of TCP traffic. The intermittent traffic bursts have an impact on other applications. When multiple users compete for the bandwidth at the same time, the players may estimate the network bandwidth incorrectly. This will cause the players switch among the different video resolutions frequently, and the users experience bad QoE (quality of experience). As the transport layer protocol, TCP congestion control algorithm plays a decisive role in the video transmission efficiency, but the traditional congestion control algorithm is not customized for the DASH streaming media transmission. This paper proposes a TCP congestion control algorithm TCP-HAS which adapts TCP Vegas to the DASH streaming media transmission. More specifically, bandwidth and video's bitrate are also used to set TCP congestion control parameters in TCP-HAS. Experimental results demonstrate that TCP-HAS can improve network QoS (quality of service) as well as QoE of the users when multiple users share the bottleneck link bandwidth.

Key words adaptive streaming; dynamic adaptive streaming over HTTP; TCP congestion control algorithm; quality of experience (QoE); quality of service (QoS)

收稿日期:2018-11-05;**修回日期**:2019-04-04

基金项目:国家重点研发计划项目(2017YFB0801703);国家自然科学基金项目(61602114,61872080,61502098);赛尔网络下一代互联网技术创新项目(NGII20150108,NGII20170406);江苏省自然科学基金项目(BK20151416)

This work was supported by the National Key Research and Development Program of China (2017YFB0801703), the National Natural Science Foundation of China (61602114, 61872080, 61502098), the CERNET Innovation Project (NGII20150108, NGII20170406), and the Natural Science Foundation of Jiangsu Province (BK20151416).

摘 要 随着互联网视频流量的快速增长,流媒体传输技术也日新月异,从传统的使用UDP传输协议的实时流媒体协议到使用TCP传输协议的HTTP协议,各大视频服务提供商都在为获得更多用户不断发展新的流媒体传输技术.超文本传输协议上的动态自适应流媒体作为目前最流行的自适应流媒体传输技术,在提高用户观影体验方面具有很多的优点.但是它的分片传输所形成的ON-OFF传输模式会造成TCP流的突发.这种间歇性的流突发会对其他的应用产生一定的影响.当多个客户端同时竞争带宽时会造成播放器错误估计网络带宽,从而产生视频分辨率频繁切换,对用户体验产生极大的负面影响.TCP作为传输层的协议,其拥塞控制算法对视频的传输效率起着决定性作用,由于传统的拥塞控制算法不能很好地适应DASH流媒体的传输,提出了TCP-HAS拥塞控制算法.该算法基于TCP Vegas进行了优化,将带宽估计值与视频码率相结合用于设置TCP拥塞控制参数.实验表明TCP-HAS能够提升网络QoS,并能在多个用户共享链路带宽时提升用户观影体验.

关键词 自适应流;超文本传输协议上的动态自适应流;传输控制协议拥塞控制算法;体验质量;服务质量

中图法分类号 TP393

根据2018年思科公司的全球互联网流量研究报告[1]显示,从2017—2022年,互联网视频流量将增长4倍,预计到2022年,IP视频流量占整个互联网流量的比例将高达82%.YouTube是全球最大的视频内容提供商之一,拥有超过10亿的用户,观看时长每天可达数亿小时,并产生数亿的观看次数,因此视频服务提供商也致力于为用户提供更好的观影体验以赢得更多的效益.随着用户观看视频需求量的增大,流媒体传输技术也日新月异,从传统的基于用户数据报协议(user datagram protocol, UDP)的实时传输协议(real-time transport protocol, RTP)转变为基于传输控制协议(transmission control protocol, TCP)的超文本传输协议(hyper text transfer protocol, HTTP)传输模式,各大视频服务提供商为获取更多的用户不断提出新的流媒体传输技术.码流自适应HAS(HTTP adaptive streaming)是当前应用比较广泛的流媒体传输技术,全球主要的视频服务提供商都采用了这个技术,而且有不同的实现方案,如Adobe公司基于Flash的HTTP动态流媒体方案、微软公司的平滑流媒体方案IIS(Internet information server) Smooth Streaming[2]、苹果公司的HLS(HTTP live streaming)[3]方案、MPEG与3GPP联合提出的基于HTTP的动态自适应流媒体传输技术DASH(dynamic adaptive streaming over HTTP)[4]协议.这些技术将视频编码为不同的分辨率,并切分为许多片段来进行传输,这种传输技术可以在不同分辨率分片之间进行无缝切换并提供高质量的业务体验.

除了HAS技术外,互联网视频服务提供商也采用了多种技术优化视频播放体验,如改善解码技术、应用层速率控制、网关流量整型、视频流量传输协议优化等,这些技术旨在提高网络资源利用率、改善瓶颈链路带宽竞争等问题,从而使用户获得良好的视频播放服务质量和服务体验.TCP作为HAS的传输层协议,是网络中提供端到端拥塞控制的主要组成部分,其传输性能是影响视频客户端播放体验的重要因素之一.为了适应不同的网络环境,许多拥塞控制算法已经被提了出来,例如NewReno[5],CUBIC[6],Westwood[7],CTCP[8],DCTCP[9],TCP-Illinois[10]等,但这些算法大都适用于一般的文件传输,不能很好地适应于自适应流媒体的ON-OFF传输特点.传统的TCP拥塞控制算法会在DASH视频传输空闲时间超时后重新进入慢启动阶段,造成网络利用率低,而DASH视频的突发也在一定程度上影响其他TCP流的传输,当多个HAS客户端在瓶颈链路产生带宽竞争时,ON-OFF传输模式也可能会使播放器错误地估测带宽,从而导致不公平的带宽竞争以及视频分辨率频繁切换,对客户端的体验质量(quality of experience, QoE)和网络服务质量(quality of service, QoS)产生影响,极大地降低了用户的观看体验.文献[11]根据流媒体的特点给出了在交换设备上的优化传输方案,由于需要对交换设备定制,实际环境中部署起来较为困难.

为此,本文结合自适应流媒体传输技术DASH,对TCP拥塞控制算法的优化展开研究,主要贡献有:基于TCP Vegas提出一种在服务器端实现的拥塞控制算法TCP-HAS.它在自适应流媒体传输发生空闲后选择最接近当前流所占带宽值的码率,根据码率计算出最小窗口值,同时将慢启动门限值设置为当前窗口值,进入拥塞避免阶段.这种做法不但

能够防止传统 TCP 拥塞控制算法在空闲超时后将拥塞窗口降为最低,也能够防止慢启动快速增加拥塞窗口到带宽时延积产生的突发,使得 TCP 拥塞窗口维持在一个相对稳定的值,客户端播放器会结合上一分片的吞吐量请求一个不高但相对稳定的分辨率分片,从而降低视频分辨率的频繁波动.实验表明在进行 DASH 视频传输时,TCP-HAS 能够提升网络 QoS 和用户的 QoE,并能在多个用户共享链路带宽时提升用户观影体验.

1　DASH 流媒体技术简介

本节将对基于 HTTP 的动态自适应流媒体传输技术 DASH 进行详细介绍,同时对其在传输过程中可能产生的问题进行分析.

1.1　DASH 流媒体技术

目前主流的 HAS 传输技术有 DASH 和 HLS 视频传输技术.DASH 是独立于视频服务提供商的国际标准,于 2012 年发布,适用于安卓终端设备;而 HLS 主要应用在苹果公司开发的 iOS 系统,为 iOS 设备(如 iPhone,iPad)提供音视频直播和点播方案.

DASH 架构主要分为 3 个部分(如图 1 所示):视频内容生成模块、流媒体服务器模块和 DASH 客户端播放器.内容生成部分按照从低到高的码率水平对视频进行编码切片并生成媒体展示描述(media presentation description, MPD)文件,MPD 文件记录视频分片的信息以及资源 URL.流媒体服务器提供 HTTP 服务和视频服务,负责内容的存储和分发.DASH 播放器主要是对 MPD 文件的解析以及对视频片段的请求.用户首先请求 MPD 文件,播放器对其解析,然后根据当前的网络状况来请求特定的分片,通过分片的 URL 来建立 HTTP 连接,同时视频播放过程中客户端也会根据网络状况来选择合适的分片,其传输过程如图 2 所示.

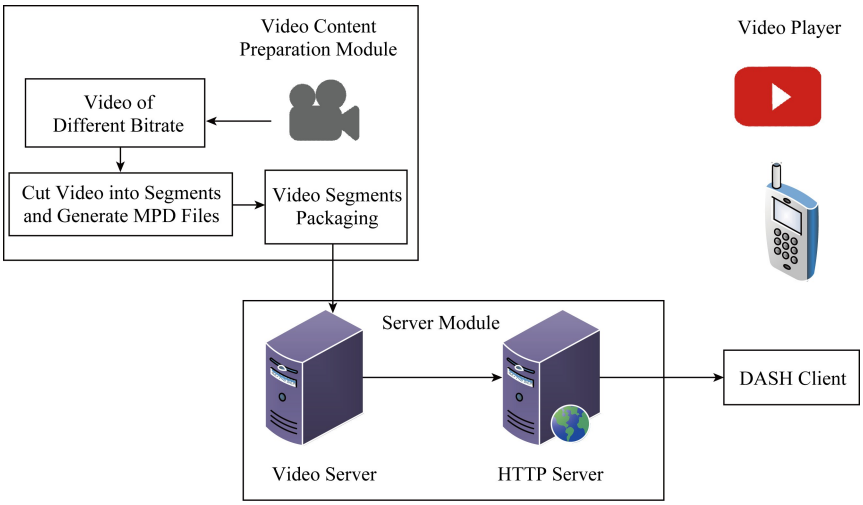


Fig. 1　DASH transmission technology architecture

图 1　DASH 传输技术架构

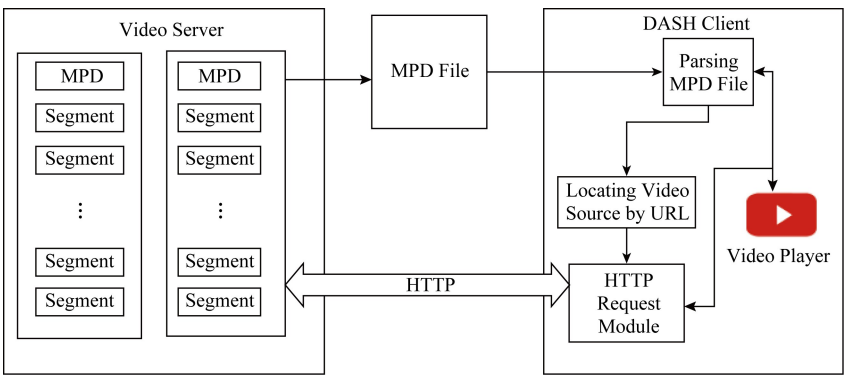


Fig. 2　DASH streaming media transmission process

图 2　DASH 流媒体传输过程

1.2 DASH 播放器带宽竞争分析

由于 DASH 流媒体服务器端的分段存储和客户端播放器缓存的存在,当两者建立 TCP 连接后,客户端会依次进入如图 3 所示的 2 个阶段[12]:

1) 初始缓冲阶段.这一阶段播放器在请求分片时为了尽可能快地达到缓冲区的大小,只要前一个片段缓冲完毕,就会立即请求下一个片段,直到缓冲区满,然后进入下一阶段.

2) 稳态阶段.这一阶段播放器缓冲区被分片填满的大小恒定不变.假设一个分片时长为 T(单位为 s),播放器每间隔时长 T 从服务器端下载分片,网速较好的时候,一次分片传输结束到下一个分片传输开始间隔时长 t, 下载时长为 $T-t$,在某些情况下也有可能刚接收到前一个分片就请求下一个.这种情况播放器的下载会形成一个 ON-OFF 模式,ON 状态下载分片(持续时长为 $T-t$),OFF 状态处于空闲状态(持续时长为 t).

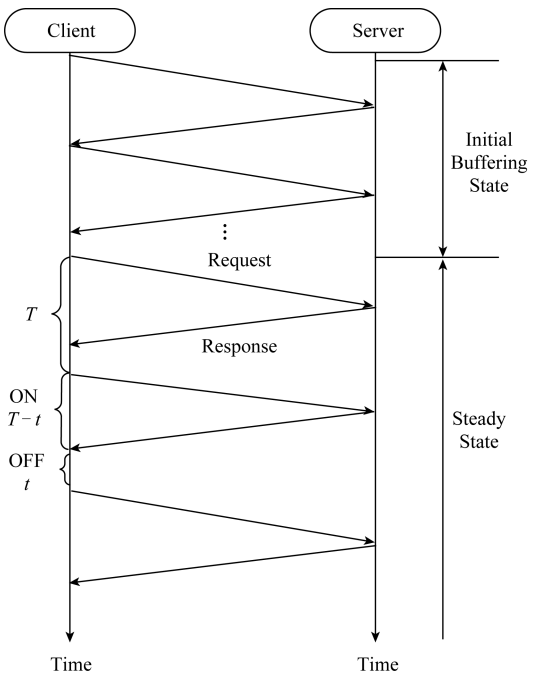
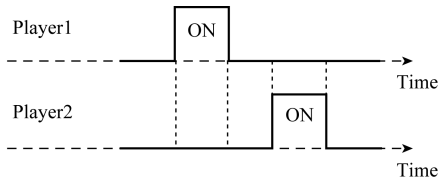


Fig. 3 Client and server request-response process

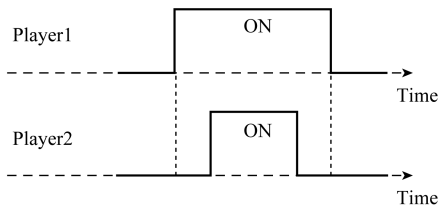
图 3 客户服务器请求-响应时间图

DASH 视频的分片和 ON-OFF 传输模式会使服务器端应用程序处于速率限制和空闲状态,进而影响 TCP 的传输性能.对于许多一直有数据要发送的 TCP 流来说,已经发送但还没有被确认的数据量 *FlightSize* 会随着拥塞窗口 *cwnd* 的增加而增加.但是对于自适应流媒体这类应用程序来说,由于其分片传输时表现出的 ON-OFF 特点,可能会存在信道空闲或无法以 *cwnd* 所允许的最大速率发送的状态,在这种情况下,2 个往返时间(round trip time, *RTT*)内的 *FlightSize* 可能会从远小于 *cwnd* 的值变化到 *cwnd* 所允许的最大值,由此引发数据包突发而带来的网络拥塞状况.同时由于空闲持续的时间超过重传超时(retransmission timeout, *RTO*)[13]后,传统的 TCP 协议栈会重新进入慢启动,造成拥塞窗口变小,不能充分利用可用带宽.与传统的流媒体传输技术不同的是,当 2 个或多个 DASH 播放器在请求分片时,由于多个播放器 ON-OFF 时间段的重叠,它们可能会不正确地估测所应该获得的带宽值,这样会带来播放不稳定、带宽的不公平竞争和利用不充分等问题.

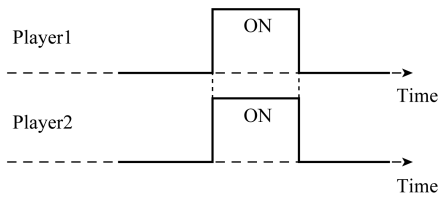
假设 2 个 DASH 播放器共享瓶颈链路的总容量为 C,它们接收端所获得的吞吐量分别为 f_1 和 f_2,并且每个播放器都已经处于稳态阶段.忽略 TCP 协议栈的影响,在下载片段时,理论上 2 个播放器所获得的吞吐量都应该为 $f=C/2$,但是由于 ON-OFF 模式的存在,2 个播放器所获得的吞吐量可能为 $f_1=f_2>f$,如图 4(a)所示,2 个播放器的 ON 状态没有重叠部分,都高于它们的公平共享值,基于所估测的 TCP 吞吐量,2 个播放器都将请求较高的码率分片,此时可能发生网络拥塞,导致吞吐量下降,反过来它们又会切换为较低的码率,由于 ON-OFF 存在于整个播放期,这种情况将会反复



(a) The ON period of the two players does not overlap



(b) The ON period of Player2 falls within the ON period of player1



(c) The ON period of the two players coincides completely

Fig. 4 Three cases of two players about ON-OFF

图 4 2 个播放器 ON-OFF 的 3 种情况图

发生,从而会造成视频播放的不稳定.图 4(b)中是另外一种情形,播放器 2 的 ON 时间段落在了播放器 1 的 ON 时间段内,当播放器 1 请求的分片码率大于播放器 2 请求的分片码率时,这种情况就有可能发生.此时播放器 2 所估测的吞吐量为 f,而播放器 1 估测的大于 f,两者可能最终获得一个稳定但不公平的带宽值,播放器会趋于请求较高码率的分片.图 4(c)是 2 个播放器的 ON 时间段完全重合的情形,假设在服务器端 1 个视频片段有 2 种码率 b_1 和 b_2,其中 $b_1<C/2$,$b_1+b_2<C$,$b_2>C/2$,这种情况下当 $b_1 \ll b_2$ 且 $b_1+b_2 \approx C$ 时,2 个播放器都会趋于请求 b_1,使得带宽利用不充分.

前面的例子说明了 2 个 DASH 播放器的一些竞争场景.其他一些因素也有可能对视频的播放产生不稳定、竞争不公平和带宽利用率不充分等问题,例如播放器码率自适应算法、网络带宽波动以及视频编码技术等.本文将在 TCP 层通过优化拥塞控制算法来减弱空闲状态对客户端播放器码率切换的影响和服务器端 TCP 流突发对其他流的影响.

2　TCP-HAS 算法设计

本节将结合 DASH 的传输特点,基于 TCP Vegas[14] 拥塞控制算法给出 TCP-HAS 的设计思路.TCP-HAS 主要分为 4 个功能模块:1)初始拥塞窗口设置;2)带宽估测;3)最佳码率分片选择;4)OFF 阶段重新设定拥塞窗口和慢启动门限值.具体的架构如图 5 所示:

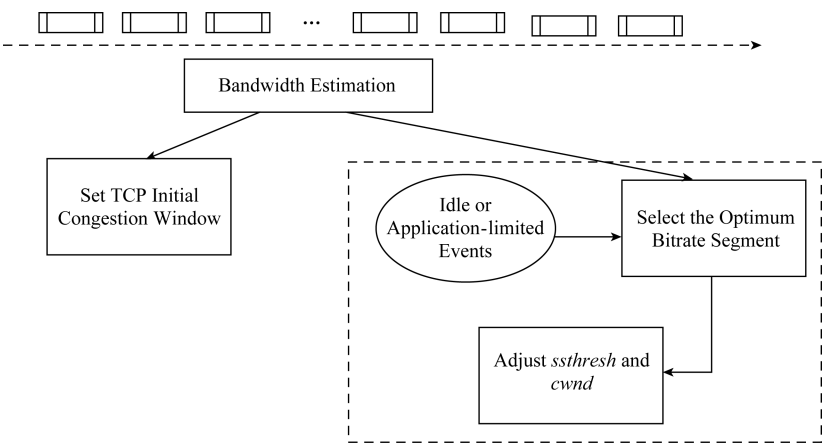


Fig. 5　TCP-HAS architecture

图 5　TCP-HAS 总体架构

带宽估测是在整个视频传输过程中都在进行的,视频传输开始时根据服务器端视频片段码率来选择初始拥塞窗口值,慢启动的窗口增长方式是指数上升,进入拥塞避免阶段后采取基于时延的拥塞控制算法,同时在遇到网络拥塞事件和空闲事件时,根据带宽估测值重新调整拥塞窗口 $cwnd$ 和慢启动门限 $ssthresh$.

2.1　初始拥塞窗口设置

TCP-HAS 在慢启动阶段的 $cwnd$ 呈指数增加.由于客户端的初始缓冲时延对于用户的体验有着非常重要的影响,因此 TCP-HAS 根据当前服务器端的索引文件来获取分片的最低码率,根据码率来设置初始拥塞窗口,如式(1)所示:

$$cwnd=\frac{min\_bitrate\times RTT}{8\times MSS}, \tag{1}$$

其中,$min\_bitrate$ 为视频分段的最小播放码率;MSS 为 TCP 最大报文长度,以太网中 $MSS=1\,460$ B.TCP 连接初始拥塞窗口 TCP_INIT_CWND 取 $cwnd$ 和 10 中的较大值.

$$TCP\_INIT\_CWND=\max(cwnd,10). \tag{2}$$

2.2　带宽估计

TCP-HAS 采用 TIBET[15] 来进行带宽估计,通过估测网络中的数据包数来计算可用带宽值.

假设在时间段 P 内有 $L_1L_2\cdots L_n$ 共 n 个数据包传输通过,L_i 为每个数据包的长度,则占用的网络带宽可以用 $\frac{1}{P}\sum_{i=1}^{n}L_i$ 来表示,平均占用带宽 Bwe 为

$$Bwe=\frac{n\bar{L}}{P}=\frac{\bar{L}}{\frac{P}{n}}, \tag{3}$$

其中,$\bar{L}=\frac{1}{n}\sum_{i=1}^{n}L_i$ 为所有数据包的平均长度,$\frac{P}{n}$ 为

报文平均间隔时间.所以该方案的基本思想就是通过测得平均数据包长度和平均时间间隔来得到带宽估计值.由于拥塞的产生可能是由于可用带宽的低频分量以及延迟响应选项造成的,为了得到一个较为平滑的带宽值,使用低通滤波的方法处理结果.

2.3 最佳码率选择

DASH 最佳码率的选择基于测得的带宽 Bwe,同时需要在服务器端来获得本视频的码率取值范围,视频的码率取值存在于视频的索引文件中,需要从索引文件中获取这些值并存储到数组中,最佳分片码率选择的伪代码如算法 1 所示:

算法 1. 最佳分片码率选择.

① for $i=Max\_EncodingRate$ to 0, $i--$
② if $Bwe \geqslant EncodingRate[i]$ then
③ break;
④ end if
⑤ end for
⑥ $QLevel \leftarrow i$.

算法 1 中 $Max\_EncodingRate$ 为最高的视频码率等级,依次从最高阶向下进行判断,选择小于带宽估计值 Bwe 的最高码率等级,同时需要考虑该模块执行的时间点,本方案中最佳分片码率估计将会在网络拥塞发生后和空闲后执行.因为拥塞发生证明网络状况对当前的视频传输不是很理想,应当调整发送窗口,当 DASH 传输出现空闲时间段时,意味着网络状况发生变化,因此为了适应这种变化,在传输空闲后,该模块也会被执行.

2.4 OFF 阶段重新设定拥塞窗口和慢启动门限值

DASH 视频的传输并不是无间断地传输数据,可能会存在不发送数据的空闲状态,也可能存在应用程序所发的数据量小于拥塞窗口的值,此时发送端的拥塞窗口会一直增长,导致其不能很好地反映当前网络带宽利用情况,当一旦有大量数据要发送时会立即占满整个窗口,造成一定程度的突发,CWV[16]机制就是为了防止应用程序窗口增长过大导致突发而提出的.

CWV 机制主要分为 2 个部分:1)当应用程序处于空闲状态时,重新计算拥塞窗口,防止其增长过大造成数据的突发从而导致网络出现拥塞状况;2)当网络上的未确认数据包的数量远小于当前拥塞窗口值时,重新调整其大小.在第 1 种情况中,TCP 计算上一次发送数据距离本次发送数据的时间间隔,当空闲时间间隔超过 RTO 后,Linux 内核实现中的函数 $tcp\_cwnd\_restart$ 会减小拥塞窗口的值,重新设置慢启动门限值,超时几个 RTO,拥塞窗口就会减半几次,最小会降为 10 个 MSS,同时设置 $ssthresh=\max(ssthresh,\frac{3}{4}cwnd)$.第 2 种情况下,判断网络是否处于满载状态的依据条件是网络中的数据包量达到窗口一半就可以,如果未达到拥塞窗口的一半,重新调整慢启动门限值和拥塞窗口:

$$ssthresh=\max(ssthresh,\frac{3}{4}cwnd), \tag{4}$$

$$cwnd=\frac{1}{2}(cwnd+cwnd\_used), \tag{5}$$

其中 $cwnd\_used$ 为程序实际使用的窗口大小.

虽然该机制在一定程度上能够避免应用程序的数据突发问题,但是对于时长通常为几秒的 DASH 视频分片来说,其空闲时长远超过 RTO,导致其跟传统 TCP 一样拥塞窗口最终降为初始值,重新进入慢启动阶段,当拥塞窗口重新爬升到网络带宽阈值时,通常需要几百毫秒到几秒,产生一定的时延.因此 TCP-HAS 定义了 2 种事件:流媒体传输空闲事件(idle event)和应用程序限制事件(application-limited event),在检测到这 2 个事件时都进入 TCP_HAS 进行处理.

1) 流媒体传输空闲事件的检测.流媒体服务器在收到新的请求时,计算该时刻距上一次数据报文发送的时间差,并判断是否发生超时,若发生超时,表明出现流媒体传输空闲事件.

2) 应用程序限制事件的检测.流媒体服务器在用户连接数目超出负荷能力时,对数据的读取成为瓶颈,如果 TCP 连接中未确认的数据报文数目小于拥塞窗口的一半,则发生了应用程序限制事件.

检测到流媒体传输空闲事件和应用程序限制事件后,进入 TCP_HAS 拥塞控制算法.TCP_HAS 会结合当前所估测的网络带宽和分片码率,选择小于带宽估计值 Bwe 的最高码率,并重置 $ssthresh$ 和 $cwnd$.TCP-HAS 算法将 $ssthresh$ 的值作调整:

$$ssthresh=EncodingRate[Qlevel]\times RTT\times\lambda, \tag{6}$$

其中 $EncodingRate[QLevel]$ 是根据可用带宽估计出的最优码率,设置系数 $\lambda(\lambda>1)$ 是为了确保传输码率大于需要的播放码率,本文设置 $\lambda=1.2$.设置 $cwnd=ssthresh$,这样做的好处是避免了在空闲时间段后将拥塞窗口减为初始值,以便减小传输下一个分片时所存在的时延,同时 $ssthresh$ 能够使得客户端在长时间的空闲后获得一个合理的速率.慢启动门限值和拥塞窗口的调整是同步的.TCP-HAS 算法的伪代码如算法 2 所示:

算法 2. TCP-HAS 算法.

① if 流媒体传输空闲事件或者应用程序限制事件被检测到 then

② 估计可用带宽 Bwe;

③ 选择最佳码率 $QLevel$;

④ $ssthresh \leftarrow EncodingRate[QLevel] \times RTT \times \lambda$;

⑤ $cwnd \leftarrow ssthresh$;

⑥ end if

如果流媒体传输空闲事件和应用程序限制事件都没有被检测到,则 TCP-HAS 算法退化为 TCP Vegas 算法.

3　实验与分析

本节将搭建实际的网络环境来验证分析 TCP-HAS 的合理性.由于当前 Linux 内核默认的拥塞控制算法为 CUBIC,因此首先在无随机丢包的网络环境下,分别在单条和多条流传输时,从拥塞窗口、网络 QoS 和客户端 QoE[17] 对比分析 TCP-HAS 和 CUBIC 的传输性能.

3.1　实验环境部署

本实验平台采用服务器客户端的模式,主要由 2 台 Linux 主机和 2 台路由器组成.其中 1 台 PC 机作为客户端,另外 1 台作为服务器端,并在中间设置 1 个路由器来模拟端到端的往返时延和丢包率.Linux 系统采用 CentOS6.9,内核源代码版本使用 4.10.此外本实验平台客户端播放器采用 VLC[18],视频服务器采用 Nginx.具体的网络环境拓扑如图 6 所示.

实验采用路由器中的 sch_netem 模块来模拟实际网络环境,模块中的 tc 命令可以根据实验需求来设置不同的时延,通过 ethtool 工具来设置网卡速率.

本实验采用 ITEC 的 DASH Dataset[19] 数据来进行实验,视频为大雄兔 BigBuckBunny,它是由开源软件 Blender 所制作的动画短片,时长为 10 min 左右,包含 320×240,480×360,854×480,1 280×720,1 920×1 080 这 5 种分辨率,片段时长有 1 s,2 s,4 s,6 s,10 s,15 s 共 6 种,时长为 4 s 的码率有 46 Kbps,89 Kbps,…,4.2 Mbps,如图 7 所示.

客户端采用的播放器为 VLC,可通过 URL 来访问服务器进行视频的点播.为了获得视频播放过程中的窗口、时延等信息,本实验在视频服务器端使用 Linux 的内核模块 tcp_probe 来监听特定 TCP 连接的各个参数.

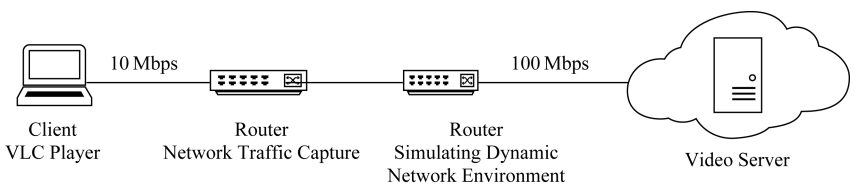


Fig. 6　Experimental topology

图 6　实验拓扑图

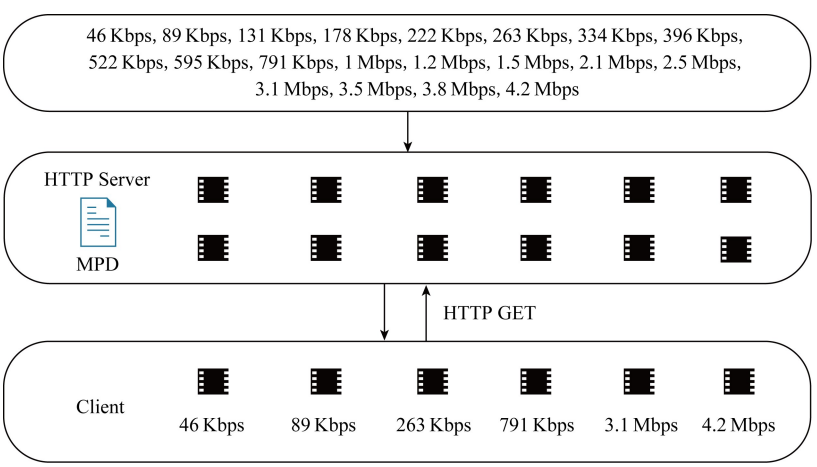


Fig. 7　Video slicing and transmission process

图 7　视频切片及传输过程

3.2 实验结果分析

发送端拥塞窗口反映了能够有多少数据发送到网络中,过大的拥塞窗口会造成 TCP 流的突发,过小的拥塞窗口又限制了 TCP 流的发送速率,因此设置合适的拥塞窗口大小对 TCP 流的传输性能具有重要意义.同时网络 QoS 和客户端 QoE 也是衡量 TCP 传输性能的关键指标.因此本节在无随机丢包的网络环境下进行实验,分别在单条和多条流的情况下,从 *cwnd* 的变化和网络 QoS 参数对 TCP-HAS 和 CUBIC 进行对比分析.最后通过客户端 QoE 指标、码率切换频率来对比分析 TCP-HAS 和 CUBIC.

3.2.1 拥塞窗口变化分析

1) 10 Mbps,无随机丢包率,单条流

图 8 是 TCP-HAS 与 CUBIC 在带宽为 10 Mbps、随机丢包率为 0%条件下,单个用户播放视频时发送端的拥塞窗口变化图.可以看出 CUBIC 比较激进,前 50 s 阶段 *cwnd* 上升到 80,丢包发生后降为 60,波动幅度比较大,随后在 DASH 流传输业务 ON-OFF 模式下出现了更为剧烈的窗口波动,空闲后降为初始值重新进入慢启动,并且间隔时间刚好为片段时长 4 s.而 TCP-HAS 前 50 s 阶段 *cwnd* 始终在 45 附近,并且波动幅度较小,50 s 之后在 ON-OFF 的传输模式下窗口变化也比较平稳.

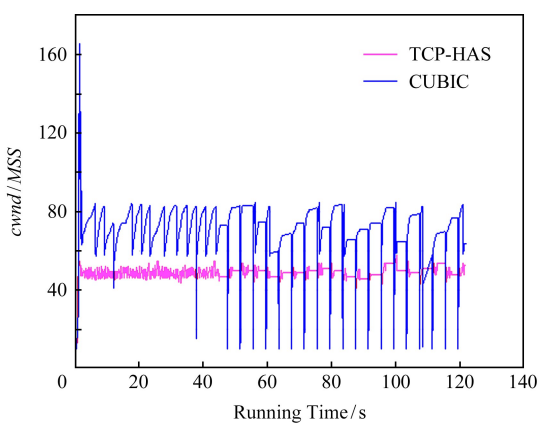


Fig. 8 *cwnd* variation of TCP-HAS and CUBIC

图 8 TCP-HAS 和 CUBIC 的 *cwnd* 变化图

2) 10 Mbps,无随机丢包率,两条流

图 9 和图 10 分别是 CUBIC 和 TCP-HAS 在 2 条流共享带宽时的 *cwnd* 变化值.两者在 DASH 流发生空闲后 *cwnd* 都发生了剧烈的波动,由于 2 条流发生空闲的时间段不同,因此 *cwnd* 交替增长.但是 TCP-HAS 比 CUBIC 的整体波动幅度要小得多.

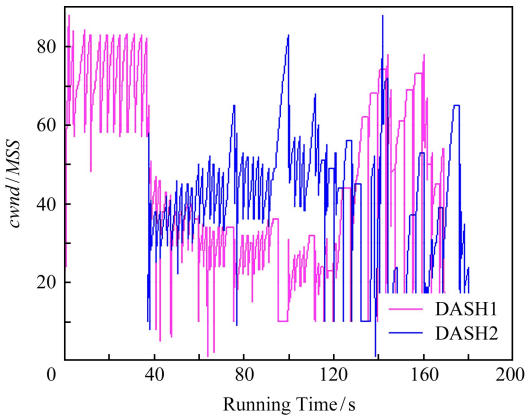


Fig. 9 *cwnd* variation of two competitive CUBIC flows

图 9 2 条 CUBIC 竞争流的 *cwnd* 变化图

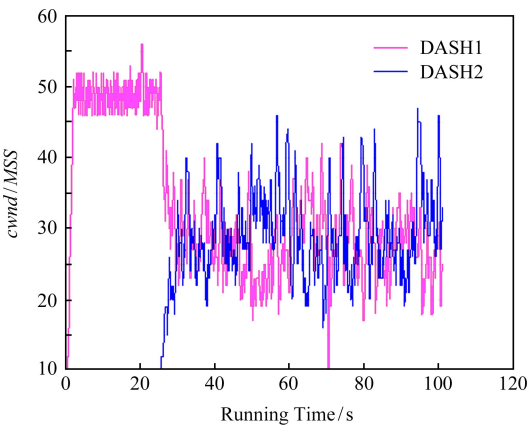


Fig. 10 *cwnd* variation of two competitive TCP-HAS flows

图 10 2 条 TCP-HAS 竞争流的 *cwnd* 变化图

3.2.2 QoS 指标分析

QoS 是衡量网络整体传输性能的重要指标,包括网络丢包率、时延、时延抖动等.本节将从丢包率、时延和时延抖动这 3 个方面对 TCP-HAS 和 CUBIC 进行对比分析.

1) 丢包率

图 11 所示为单个用户播放视频时 100 s 内 CUBIC 和 TCP-HAS 的丢包率变化图,可以看出 CUBIC 在初始时刻的丢包率较高,随后趋于稳定,而 TCP-HAS 在整个传输过程中几乎没有丢包.主要是因为 CUBIC 的竞争性比较大,在初始时刻希望尽快占用可用带宽,因此丢包率也高.

图 12 和图 13 分别是 CUBIC 和 TCP-HAS 在 2 个用户共享带宽时的丢包率变化图.CUBIC 和 TCP-HAS 的第 2 个用户都是在 30 s 后开启,并出现了短暂的高丢包,随后 CUBIC 两条流的丢包率逐渐趋于一致,而 TCP-HAS 第 1 条流丢包率始终为 0.总体上 CUBIC 的丢包率更高.

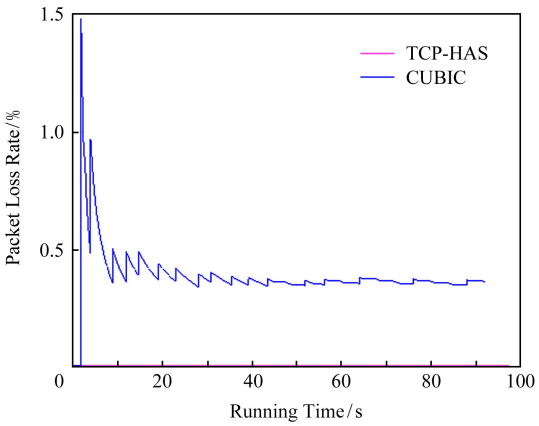


Fig. 11　Single stream packet loss rate of CUBIC and TCP-HAS

图 11　CUBIC 和 TCP-HAS 单条流丢包率变化图

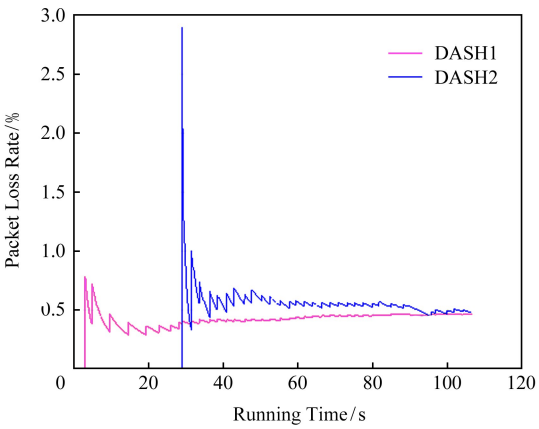


Fig. 12　Packet loss rate of two CUBIC DASH flows

图 12　2 条 CUBIC DASH 流的丢包率变化图

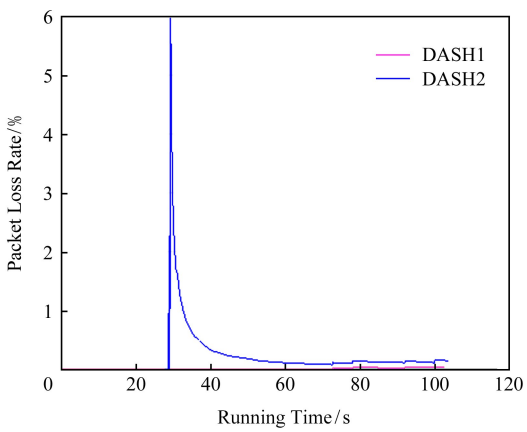


Fig. 13　Packet loss rate of two TCP-HAS DASH flows

图 13　2 条 TCP-HAS DASH 流的丢包率变化图

根据图 11～13 的实验结果,TCP-HAS 在丢包率这一指标上要优于 CUBIC.

2) 往返时延和抖动

图 14 是 CUBIC 和 TCP-HAS 在单个用户播放视频时的 *RTT* 概率累积分布图.可以很明显看出 CUBIC 的往返时延在 80～100 ms 区间所占的比例为 60%左右,而 TCP-HAS 的往返时延几乎全部落在 55～65 ms 之间,可见 CUBIC 的总体 *RTT* 值要远大于 TCP-HAS 的 *RTT* 值.

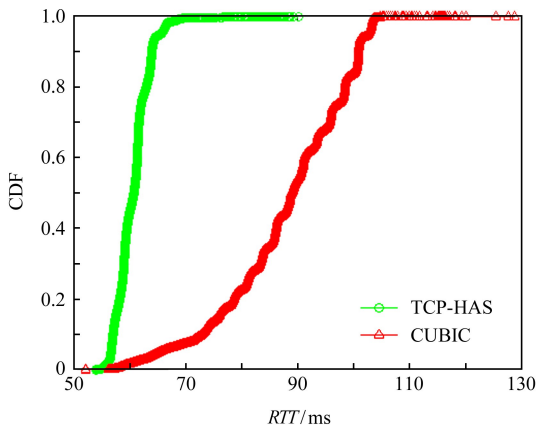


Fig. 14　*RTT* CDF of TCP-HAS and CUBIC

图 14　TCP-HAS 和 CUBIC 往返时延 CDF 图

图 15 和图 16 分别是 CUBIC 和 TCP-HAS 在 2 个用户共享带宽时的 *RTT* 概率累积分布图,结果与图 14 类似,唯一不同之处是 CUBIC 第 2 条流的往返时延要小于第 1 条流,而 TCP-HAS 恰好相反.

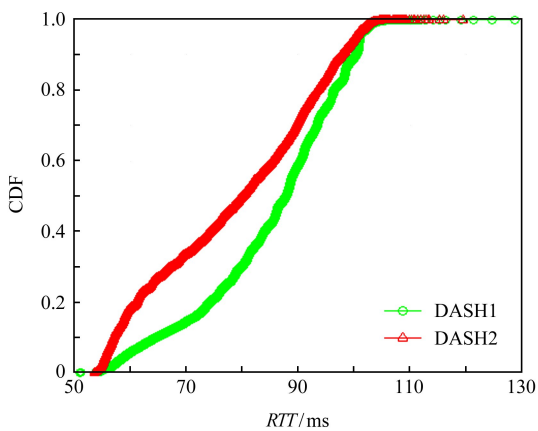


Fig. 15　*RTT* CDF of two competing CUBIC DASH flows

图 15　CUBIC 的 2 条 DASH 竞争流的往返时延 CDF 图

图 17 是 CUBIC 和 TCP-HAS 在单个用户播放视频时的 *RTT* 变化图,可以看出 CUBIC 的时延抖动范围为 60～120 ms,TCP-HAS 的抖动范围为 60～80 ms.

图 18 和图 19 分别是 CUBIC 和 TCP-HAS 在 2 个用户播放视频时的 RTT 变化图,总体上 CUBIC 的抖动频率还是要比 TCP-HAS 大,但 CUBIC 在

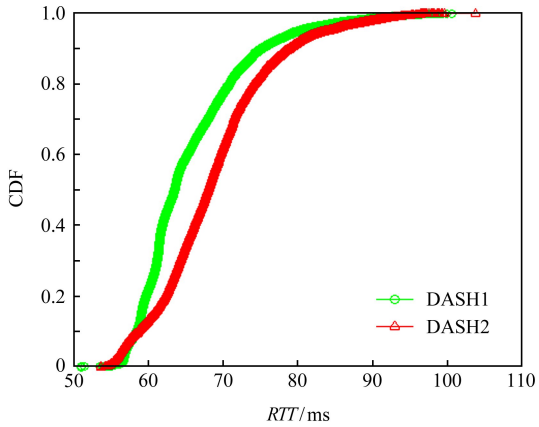


Fig. 16 *RTT* CDF of two competing TCP-HAS DASH flows

图 16 TCP-HAS 的 2 条 DASH 竞争流的往返时延 CDF 图

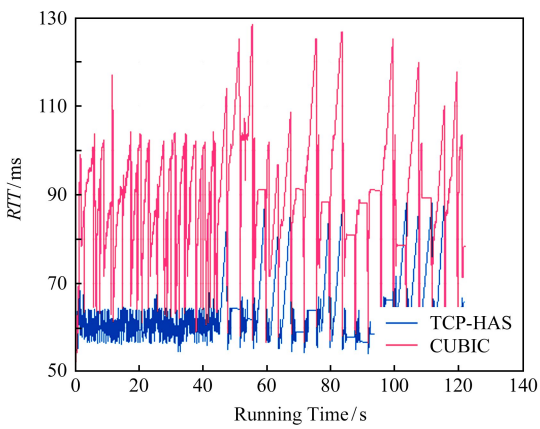


Fig. 17 *RTT* variation of TCP-HAS and CUBIC

图 17 TCP-HAS 和 CUBIC 往返时延变化图

第 2 条流开启后抖动范围几乎不变，而 TCP-HAS 却发生了剧烈的抖动.

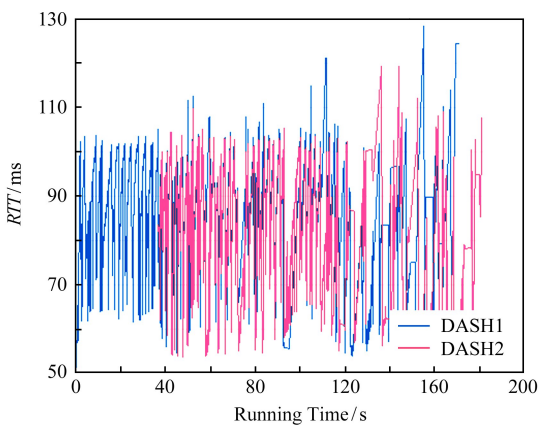


Fig. 18 *RTT* variation of two competing CUBIC DASH flows

图 18 CUBIC 的 2 条 DASH 竞争流的 *RTT* 变化图

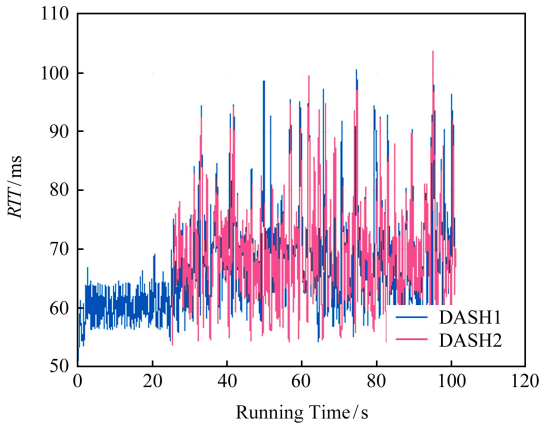


Fig. 19 *RTT* variation of two competing TCP-HAS DASH flows

图 19 TCP-HAS 的 2 条 DASH 竞争流的 *RTT* 变化图

在时延和时延抖动方面，TCP-HAS 的整体性能要优于 CUBIC.

3.2.3 QoE 指标分析

对自适应流媒体来说，影响 QoE 的主要因素有初始缓冲延迟、视频卡顿、码率或者分辨率切换次数、吞吐量等.初始缓冲时延是从客户端发出 HTTP 请求到开始播放的时间长度，一般来说实时流媒体或短视频流对初始缓冲时延更为敏感.视频码率和分辨率的切换次数也是影响客户端 QoE 的最重要因素，在网络状况发生改变时，视频码率和分辨率会进行切换以避免缓冲区过载，以此来保证视频的流畅播放，但如果切换得太频繁，并不断地从较高码率切换到较低码率的视频片段，这种情况就会使用户的观影体验大大降低，因此视频码率也需要在适应网络变化的同时维持在一个合理的波动范围内.吞吐量也是衡量 QoE 大小的一个指标，但需要结合其他因素来综合评估，吞吐量越高也不能说明用户的观影体验越好.本文综合各方面因素，选择视频码率切换次数来作为衡量 QoE 的指标.

码率信息存储在 HTTP GET 请求中，因此可以通过分析 pcap 数据包的 HTTP 请求来统计码率切换次数.

1) 单个用户

如图 20 和图 21 所示，当客户端为单个用户时，由于视频片段最大码率为 4 Mbps，小于带宽 10 Mbps，因此这种情况下 TCP-HAS 和 CUBIC 的码率变化趋势一样，从最低码率迅速切换为最大码率并维持稳定不变.

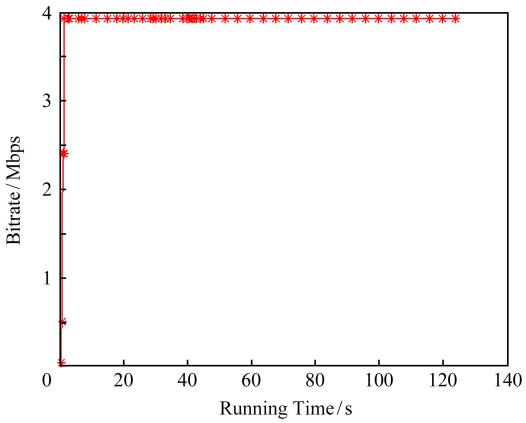


Fig. 20　Single user bitrate switching of CUBIC

图 20　CUBIC 单个用户码率切换图

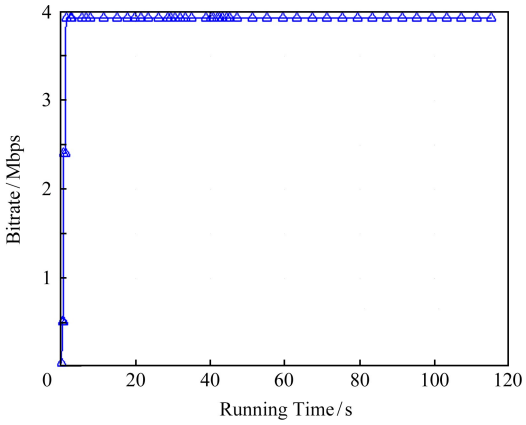


Fig. 21　Single user bitrate switching of TCP-HAS

图 21　TCP-HAS 单个用户码率切换图

2) 2 个用户

当 2 个用户同时播放时,采用 CUBIC 拥塞控制算法时,2 个播放器的码率变化此起彼伏,产生了一种交叉变化的现象,如图 22 所示.这种现象正是由于分片传输的 ON-OFF 所模式造成的.当用户 1 处于 OFF 时间段时,用户 2 估测到的带宽值比较大,会请求较高码率分片,而一旦用户 1 要请求分片时,其估测到的网络带宽值比用户 2 所估测到的要小,于是切换为较低的码率,同时用户 2 也受用户 1 的影响频繁切换码率.如图 23 所示,采用 TCP-HAS 拥塞控制算法时,由于在空闲和应用程序限制 2 种状态下对慢启动门限值重新设置,使得两者的码率波动范围变小.

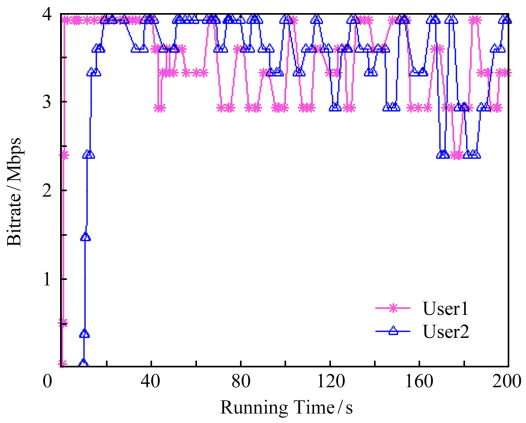


Fig. 22　Two users bitrate switching of CUBIC

图 22　CUBIC 的 2 个用户码率切换图

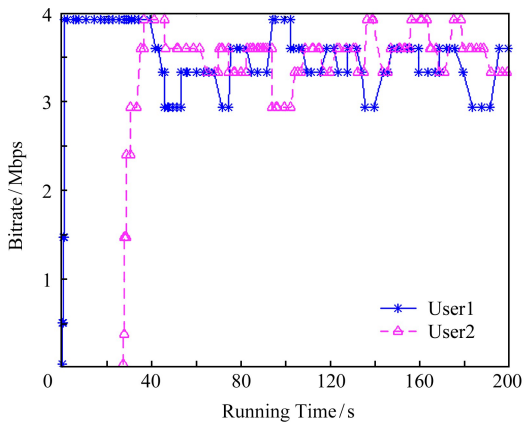


Fig. 23　Two users bitrate switching of TCP-HAS

图 23　TCP-HAS 的 2 个用户码率切换图

4　结束语

本文优化了 TCP 拥塞控制算法来适应 DASH 视频传输,将带宽与码率相结合,基于 TCP Vegas 提出了 TCP-HAS,并在 Linux 服务器端进行修改,客户端无需任何修改.TCP-HAS 在 DASH 流媒体传输出现空闲后,通过估测的带宽值来选择最佳分片码率,并根据码率重新计算 *cwnd* 和 *ssthresh*,能够较好地适应 DASH 流媒体 ON-OFF 的传输特点.相比 CUBIC 来说,虽然 TCP-HAS 比较保守,但它具有较小的往返时延和丢包率,而且 TCP 流窗口波动也比较小,能够在多个用户同时竞争带宽时获得较好的 QoS 和 QoE.由于实验环境的限制,本文无法针对大量客户端进行实验分析,这是本文的缺点之一.同时 TCP-HAS 流在与 CUBIC 流竞争时仍处于劣势,这也是基于时延拥塞控制算法的缺点.本文后续的研究工作将基于上述 2 个问题展开.

当前国内外许多大型视频服务提供商如 YouTube、Netflix、腾讯、优酷和搜狐等都将 DASH 作为主流的视频传输技术,而 YouTube 所属的 Google 公司针对视频传输也对 TCP 协议作了许多

创新性修改,并在 YouTube 服务器上部署应用.针对自适应流媒体的传输协议优化是十分必要的,并且未来还有许多新的研究和改进的空间.

参 考 文 献

[1] CISCO. Cisco visual networking index: Forecast and trends, 2017—2022 [EB/OL]. [2019-02-17]. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[2] Microsoft. Smooth streaming technical overview [EB/OL]. [2019-02-17]. https://docs.microsoft.com/en-us/iis/media/on-demand-smooth-streaming/smooth-streaming-technical-overview

[3] Pantos R, May W. RFC8216 HTTP Live Streaming [S/OL]. Fremont, CA: IETF, 2017 [2019-02-17]. https://tools.ietf.org/html/rfc8216

[4] Sodagar A. The MPEG-DASH standard for multimedia streaming over the Internet [J]. IEEE Multimedia, 2011, 18(4): 62-67

[5] Floyd S, Henderson T. RFC2582 the NewReno Modification to TCP's Fast Recovery Algorithm [S/OL]. Fremont, CA: IETF, 1999 [2019-02-17]. https://tools.ietf.org/html/rfc2582

[6] Ha S, Rhee I, Xu Lisong. CUBIC: A new TCP-friendly high-speed TCP variant [J]. ACM SIGOPS Operating Systems Review, 2008, 42(5): 64-74

[7] Casetti C, Gerla M, Mascolo S, et al. TCP Westwood: End-to-end congestion control for wired/wireless networks [J]. Wireless Networks, 2002, 8(5): 467-479

[8] Tan Kun, Song Jingmin, Zhang Qian, et al. A compound TCP approach for high-speed and long distance network [C/OL] //Proc of IEEE INFOCOM 2006. Piscataway, NJ: IEEE, 2006 [2019-02-17]. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2005-86.pdf

[9] Alizadeh M, Greenberg A, Maltz D A, et al. Data center TCP (DCTCP) [J]. ACM SIGCOMM Computer Communication Review, 2011, 41(4): 63-74

[10] Liu Shao, Başar T, Srikant R. TCP-Illinois: A loss-and delay-based congestion control algorithm for high-speed networks [J]. Performance Evaluation, 2008, 65(6/7): 417-440

[11] Wang Xueshun, Yu Shaohua, Xu Ning, et al. A dynamic threshold schedule algorithm for media traffic [J]. Journal of Compute Research and Development, 2011, 48(1): 110-117 (in Chinese)
(汪学舜,余少华,徐宁,等. 适用于流媒体传输的动态门限调度算法[J]. 计算机研究与发展,2011, 48(1): 110-117)

[12] Akhshabi S, Anantakrishnan L, Begen A C, et al. What happens when HTTP adaptive streaming players compete for bandwidth? [C] //Proc of the 22nd Int Workshop on Network and Operating System Support for Digital Audio and Video. New York: ACM, 2012: 9-14

[13] Paxson V, Allman M, Chu J, et al. RFC6298 Computing TCP's Retransmission Timer [S/OL]. Fremont, CA: IETF, 2011 [2019-02-17]. https://tools.ietf.org/html/rfc6298

[14] Brakmo L S, Peterson L L. TCP Vegas: End to end congestion avoidance on a global Internet [J]. IEEE Journal on Selected Areas in Communications, 1995, 13(8): 1465-1480

[15] Capone A, Martignon F. Bandwidth estimates in the TCP congestion control scheme [G] //LNCS 2170: Proc of the 2001 Thyrrhenian Int Workshop on Digital Communications. Berlin: Springer, 2001: 614-626

[16] Handley M, Padhye J, Floyd S. RFC2861 TCP Congestion Window Validation [S/OL]. Fremont, CA: IETF, 2000 [2019-02-17]. https://tools.ietf.org/html/rfc2861

[17] ITU-T. Vocabulary and effects of transmission parameters on customer opinion of transmission quality [EB/OL]. [2019-02-17]. https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-P.10-201711-I!! PDF-E&type=items.pdf

[18] Müller C, Timmerer C. A VLC media player plugin enabling dynamic adaptive streaming over HTTP [C] //Proc of the 19th ACM Int Conf on Multimedia. New York: ACM, 2011: 723-726

[19] Lederer S, Müller C, Timmerer C. Dynamic adaptive streaming over HTTP dataset [C] //Proc of the 3rd Multimedia Systems Conf. New York: ACM, 2012: 89-94

Wu Hua, born in 1973. PhD, associate professor. Her main research interests include network security, network measurement and network management.

Wang Ling, born in 1992. Received his master degree from Southeast University in 2018. His main research interests include TCP performance improvement, network measurement.

Cheng Guang, born in 1973. PhD, professor. PhD supervisor. His main research interests include network measurement, network security, network behavior analysis, and future network technologies.