DOI：10.7652/xjtuxb201706020

数据中心网络差分流传输控制协议研究

蔡岳平，张文鹏，罗森
（重庆大学通信工程学院，400030，重庆）

摘要：针对数据中心网络多对一通信流量产生TCP Incast拥塞导致吞吐量降低，以及小流易受大流影响难以满足应用截止时间要求等问题，提出了差分流传输控制协议(DFTCP)。DFTCP采用主动队列管理机制，利用显式拥塞通知机制传递拥塞信息，通过控制交换机队列长度来减少突发流分组的丢失以解决TCP Incast问题。DFTCP在终端服务器中对TCP流进行分类，当网络发生拥塞时，基于网络状态信息和流分类信息调节TCP拥塞窗口，通过对大流进行更大程度的拥塞退避从而减小其对网络中其他TCP流传输性能的影响，进而减少小流完成传输的时间，解决小流高延迟的问题。仿真实验表明：与传统TCP相比，DFTCP能够避免因TCP Incast拥塞导致的吞吐量崩溃；与数据中心传输控制协议(DCTCP)相比，DFTCP能够减少小流传输完成的时间，同时DFTCP能够实现共享同一链路的多个大流快速收敛，保证公平性。

关键词：云计算；数据中心网络；TCP Incast；拥塞控制；拥塞窗口
中图分类号：TP393 **文献标志码**：A **文章编号**：0253-987X(2017)06-0122-07

Differentiated Flow Transmission Control Protocol in Data Center Networks

CAI Yueping, ZHANG Wenpeng, LUO Sen
(College of Communication Engineering, Chongqing University, Chongqing 400030, China)

Abstract: Widely deployed applications in cloud computing with many-to-one communication pattern may cause TCP incast congestion problem, resulting in throughput collapse. Mice flows are easily influenced by elephant flows to miss the application deadline. To address these problems, we propose the differentiated flow transmission control protocol (DFTCP). The DFTCP adopts active queue management scheme and delivers the network congestion information through explicit congestion notifications. DFTCP reduces the number of discarding bursty small packets by controlling the queue length of switches to solve TCP incast problem. TCP flows are classified by DFTCP at the end-servers. When network congestion happens, DFTCP adjusts TCP congestion windows based on network states and classified flow information. Elephant flows have larger back-off time compared with mice flows. Simulation shows that DFTCP can effectively avoid TCP incast congestion compared with the traditional TCP, and it can reduce the completion time of mice flows compared with DCTCP. In addition, DFTCP can make multiple elephant flows converge quickly to guarantee the fairness of link sharing.

Keywords: cloud computing; data center network; TCP incast; congestion control; congestion window

---

收稿日期：2016-09-28。 作者简介：蔡岳平(1980—)，男，副教授。 基金项目：国家自然科学基金资助项目(61301119)；教育部留学归国人员启动基金资助项目(1020607820140002)。
网络出版时间：2017-03-04 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20170304.1717.004.html

随着云计算[1]的不断发展，作为云计算的基础设施和解决云计算海量数据传输与交换的关键网络，数据中心网络已成为各界关注和研究的热点[2]。利用传输控制协议（TCP）可靠的数据交付和拥塞控制机制，当前数据中心网络大量采用 TCP 协议实现服务器之间的数据传输[3-5]。然而，数据中心网络环境具有高带宽、低时延等特性，导致 TCP 在特定情况下性能急剧下降，最典型的现象就是 TCP Incast。随着对数据中心网络的深入研究，数据中心网络中的 TCP Incast 问题已经变成了一个实际的问题，广泛存在于分布式文件系统[6]、分布式存储集群[7]、MapReduce[8]等数据中心应用中。此外，数据中心网络中不同流量的传输性能需求不同，小流传输容易受到大流的影响，使得小流完成传输的时间延长，导致小流无法满足传输截止时间的要求。

目前，针对数据中心网络的 TCP 拥塞控制以及典型的 Incast 问题，存在多种尝试的解决方案。文献[9]提出通过增大以太网交换机缓存的方式减少分组丢失，但是该方式在实际实现中不具备较高的性价比。文献[10-11]均提出通过减小 TCP 的重传时间（RTO），将定时器时钟精度设置至微秒级别来减轻 TCP Incast 拥塞对网络性能的影响，但是设置微秒级的最小超时重传时间存在实现上的挑战，并且可能产生安全性问题。在新提出的拥塞控制算法方面，ICTCP 算法[12]是在接收侧基于通告窗口实施 Incast 拥塞避免的方案，而 DCTCP 算法[3]的核心思想是在不影响网络吞吐量的前提下，尽量保持交换机中队列长度较短。文献[13]提出了使用 UDP 来替换 TCP 进行数据传输，但是基于 TCP 的应用十分广泛，替换 TCP 将付出很大的代价。此外，文献[14-16]提出将 SDN（Software Defined Networks）技术应用于数据中心网络，基于全局网络状态信息动态地调节 TCP 流初始窗口大小、超时重传时间等参数，有利于实现更精确的拥塞控制。

1　TCP Incast 问题

Incast 是一种多对一的通信模式，如图 1 所示。在这种模式通信过程中，首先多个服务器发送端向同一个客户端并行地传输数据块，只有当所有发送端完成当前轮次的数据块传输后，客户端才会开始请求传输下一轮的数据块，这个过程被称为承载同步传输。由于一般商用交换机的缓冲区大小有限，同时突发的大量数据包容易造成接收方一侧的交换机缓冲过载，使得接收者无法正常接收数据。分组丢失造成发送端超时并重传分组。由于 TCP 超时的重传时间一般为毫秒级的，例如流行的 Linux 操作系统中 RTO 定时器的默认缺省值是 200 ms，虽然这个值相对于广域网较为适合，但是数据中心网络的往返时延一般为微秒级别的，因此 TCP 超时重传时间和网络时延之间严重不匹配。由于网络微秒级的往返时延变成了几百毫秒级的超时重传时延，因此接收方应用层所感受到的吞吐量比链路的实际容量降低了几个数量级，即 TCP 吞吐量崩溃。

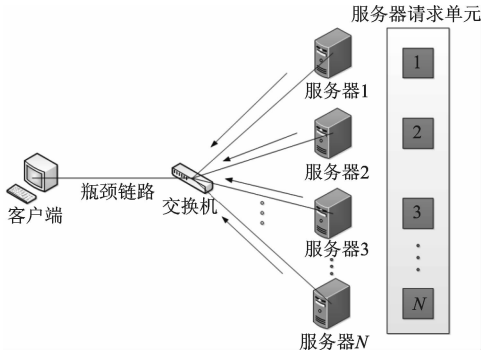
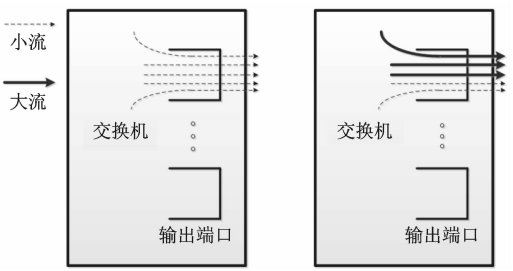


图 1　TCP Incast 模型示意图

在 Incast 拥塞过程中，大量并行的高突发小流造成的分组丢失是 TCP 吞吐量崩溃的主要原因，而在混合各种流量类型的数据中心网络环境中，大流传输是影响网络性能的主要因素，大流对链路带宽和交换机缓存资源的过分占用将导致其他 TCP 流分组丢失，从而降低网络处理高突发流量的能力以及影响小流传输，如图 2 所示。小流分组受大流传输影响导致的高延迟，将增大小流完成传输的时间，而延迟作为一项重要的网络性能指标将直接影响互联网在线服务，进一步影响公司业务收入。



（a）情形 1：高突发并行小流造成分组丢失　　（b）情形 2：大流影响小流的传输

图 2　数据中心网络流量相互影响情况

本文提出了基于流分类的传输控制协议用于数据中心网络的拥塞控制，称为差分流传输控制协议（DFTCP）。DFTCP 采用主动队列管理机制，利用显式拥塞通知 ECN（Explicit Congestion Notification）机制传递拥塞信息，在终端服务器中对 TCP

流进行分类。当网络发生拥塞时，通过对大流进行更大程度的拥塞退避从而减小对网络中其他 TCP 流传输性能的影响。首先，DFTCP 机制通过控制交换机队列长度来减小突发流分组的丢失来解决 TCP Incast 问题；其次，DFTCP 机制通过对主要影响网络性能的大流进行更大程度的拥塞退避来减少小流完成传输的时间，解决小流高延迟的问题。

2 DFTCP 机制

本节对 DFTCP 的工作机制进行详细介绍。DFTCP 的拥塞窗口调节流程如图 3 所示。

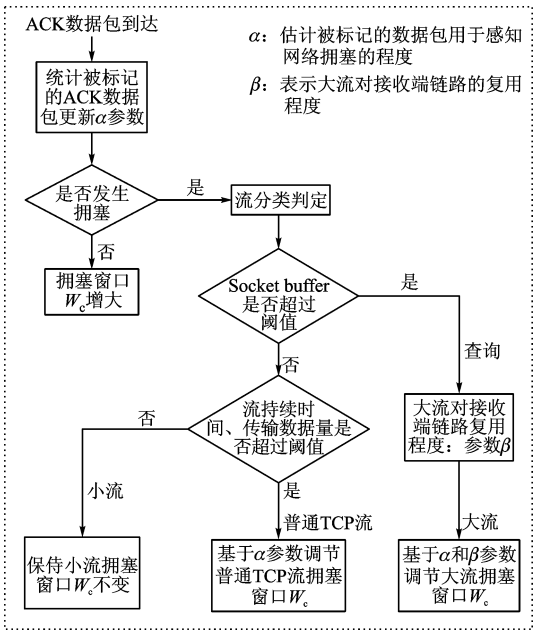


图 3 DFTCP 的拥塞窗口调节流程

DFTCP 的核心思想为通过主动队列管理机制控制交换机队列长度，减小突发流分组的丢失来解决 TCP Incast 问题。通过区分不同传输需求的 TCP 流，对于影响网络性能的大流，对其拥塞窗口 W_c 进行更大程度调节的方式来提高网络处理高突发流量的能力，同时减少小流完成传输时间，解决小流高延迟的问题。DFTCP 机制由 3 个部分组成，分别为流分类机制、拥塞信息传递、拥塞退避机制。

2.1 流分类机制

DFTCP 在终端服务器中部署流分类机制，以实现对不同传输需求 TCP 流的区分。流分类的主要目的是当网络发生拥塞时，通过对影响网络性能的大流进行更大程度的拥塞退避来减小其对网络中其他 TCP 流传输的影响。数据中心作为单一管理域，包含了终端主机操作系统的实现，在终端服务器中以分布式机制实现流分类机制，有利于提高网络的可扩展性和减小网络开销，避免了网络中的交换机和控制器为了维护大量流状态信息成为限制网络规模的瓶颈。DFTCP 将所有 TCP 流划分为 3 类，分别为低时延传输需求倾向的小流、高吞吐量传输需求倾向的大流和无明显传输需求倾向的普通 TCP 流。

小流：文献[3]指出，对时延敏感小流传输的数据量在 100 kB 到 1 MB 的范围之间，而数据中心网络流量的截止时间要求的变化范围在 10～100 ms 之间，因此使用静态阈值划分的方式，DFTCP 将小流定义为满足流持续时间小于 50 ms 并且总的传输数据量小于的 500 kB 两个条件的 TCP 流，持续时间和传输数据量的阈值均设置为变化范围内的中间值。

大流：DFTCP 机制将大流定义为套接字缓存大小超过阈值 100 kB 的 TCP 流。文献[17]指出，在终端服务器中，基于监控套接字缓存大小是否超过阈值作为标志来检测大流相对于通过轮询、抽样等方式有利于控制开销。文献[18]将发送数据量超出服务器接入链路容量 10%（100 Mbit·s^{-1}对应 1 Gbit·s^{-1}的链路）的流定义为大流。与静态阈值设定检测大流的方式相比，通过动态监测 TCP 套接字缓存大小的方式可以提高检测出影响网络性能大流的准确性，使得大流的检测与 TCP 流持续时间和已完成传输的数据量大小无关，可以避免将持续时间较长、已传输数据量较大但是不活跃的 TCP 流归类为大流。

普通 TCP 流：除了低时延传输需求倾向的小流和高吞吐量传输需求倾向的大流，其他无明显传输需求倾向的 TCP 流被定义为普通 TCP 流。

2.2 显式拥塞通知 ECN

对于 TCP 的拥塞控制机制主要可以分为两大类型，第 1 种方式是基于准确估计往返时延作为标志来感知链路的拥塞程度，第 2 种方式是采用主动队列管理（AQM）来显式地传递拥塞反馈信息。不同于广域互联网，由于数据中心网络具有微秒级的时延，较小的时延起伏变化难以准确反映网络拥塞情况，除此之外对于终端服务器，更高精度的时钟实现也十分具有挑战性。因此，DFTCP 使用 ECN 机制来传递网络拥塞信息。

ECN 工作的过程如下。①在交换机处进行数据包标记。设置标记的门限参数值为 K，当一个数据包到达交换机输出队列后，如果即时队列长度大于门限 K，则该数据包的 IP 头部被交换机用 CE（Congestion Experienced）代码点标记，否则不进行标记。②在接收端反馈拥塞信息。接收端对于每一

个带有 CE 标记的数据包，通过在 ACK 确认数据包中设置 ECE(ECN-Echo)标记，将拥塞信息反馈给源端。文献[11]指出，在数据中心网络环境下，TCP 的延迟确认机制(delayed ACKs)会影响网络性能，因此 DFTCP 关闭掉 TCP 的延迟确认机制，对于每一个 TCP 分组，接收端立即返回 ACK 数据包。③在发送源端执行拥塞控制。具体的拥塞退避算法本文将在下一小节进行详细介绍。

对于 ECN 标记门限参数 K，较高的门限值设置有利于维持较高的网络吞吐量，而较低的门限值设定则有利于改善网络处理突发流量的能力。为了不影响 TCP 吞吐量，K 的设定应该与链路的带宽时延积相关。文献[11,19]均指出，在数据中心网络中，对于使用链路带宽 C 为 1 Gbit·s^{-1} 接入网络的主机，其大部分数据包的往返时延 T 小于 400 ms。因此，带宽时延积 P 约为 50 kbit·s^{-1}，计算式为

$$P = CT \tag{1}$$

其中以太网默认的最大传输单元为 1.5 kB，此时链路大约承载 33 个最大尺寸的数据包。由于多个 TCP 连接共享瓶颈链路，如果所有 TCP 流的拥塞窗口为 W_T，为了避免丢包，必须使得瓶颈链路所有 TCP 流的拥塞窗口之和不超过链路带宽时延积加上队列长度，即满足

$$W_T \leqslant CT + K \tag{2}$$

其中文献[19]建议门限 K 设定在 20 到 30 个数据包之间，因此实验设置 ECN 标记门限参数 $K=20$。

2.3　拥塞退避机制

传统 TCP 对于带有 ECE 标记的 ACK 数据包采取的方式是固定地将对应 TCP 流的拥塞窗口减半，其拥塞窗口 W_c 的更新公式为

$$W_{c\_new} \leftarrow 1/2W_{c\_old} \tag{3}$$

这会使得链路利用率较大幅度地下降，从而导致链路资源浪费，并且可能导致部分小流传输完成的时间延长。因此，DFTCP 机制在流分类的基础上，对于不同传输需求的 TCP 流，在调节拥塞窗口的过程中分别执行不同程度的拥塞退避。

对于具有低时延传输需求倾向的小流，由于其持续时间和传输的数据量都很小，该类型流量不是造成网络拥塞的主要原因，相反该类型流量容易受到其他类型流量传输的影响，造成小流分组时延上升，增加完成传输时间，甚至导致错过截止时间的要求。因此，当收到带有 ECE 标记的 ACK 数据包时，DFTCP 保持时延敏感 TCP 流的拥塞窗口不变，其拥塞窗口 W_c 的更新公式为

$$W_{c\_new} \leftarrow W_{c\_old} \tag{4}$$

对于无明显传输需求倾向的普通 TCP 流，利用 DCTCP[3] 的方式调节拥塞窗口。发送端维持一个参数 α 用于估计被标记的数据包，以感知网络拥塞的程度，其中参数 α 的更新公式为

$$\alpha_{new} \leftarrow (1-g)\alpha_{old} + gF \tag{5}$$

式中：参数 F 表示上一个窗口数据中被标记数据包的比例；参数 g 表示加权平滑系数(实验设置为 1/16)。普通 TCP 流拥塞窗口 W_c 的更新公式为

$$W_{c\_new} \leftarrow W_{c\_old}(1-\alpha/2) \tag{6}$$

其中参数 α 的变化区间为[0 1]，α 接近 0 表示网络拥塞程度较低，α 越接近 1 表示网络拥塞程度越高。

对于高吞吐量传输需求倾向的大流，DFTCP 基于网络拥塞程度和大流对接收端链路的复用程度进行拥塞窗口调节。在大流拥塞退避过程中，利用 SDN 技术，使用控制器维护全局大流信息，目的是向终端服务器传递对应接收端服务器链路并存大流的数量信息 N。定义参数 β 用于表示大流对接收端链路的复用程度，其计算公式为

$$\beta = N/N_{max} \tag{7}$$

式中：参数 N 表示该大流对应接收端服务器链路当前并存的大流数量；N_{max} 表示服务器并存流的最大数量；参数 β 变化区间为[0,1]。文献[3]通过实验发现，并存流数量小于 150 的服务器节点数量的累积分布函数值(CDF)接近于 1，即数据中心绝大部分服务器节点的并存流数量小于 150，因此实验固定的设置 N_{max} 为 150，当参数 N 超过 150 时，设置 β 值为 1。

由于大流是造成网络拥塞、影响网络中其他流传输的重要因素，因此在大流拥塞退避过程中，DFTCP 在感知到的网络拥塞程度 α 参数的基础上，将表示大流对接收端链路复用程度的 β 参数作用于 α 参数的指数部分，从而基于网络拥塞程度和大流对接收端链路的复用程度两个方面的信息对大流进行更大程度的拥塞窗口调节。拥塞窗口 W_c 的更新公式为

$$W_{c\_new} \leftarrow W_{c\_old}(1-\alpha^{\beta}/2) \tag{8}$$

由于 β 参数的变化区间为[0,1]，因此大流相对于其他 TCP 流的拥塞退避程度更大。β 接近 0 表示大流对接收端链路的复用程度低，即当网络发生拥塞时并存的大流数量较少，因此应对单个大流进行更大程度的拥塞窗口调节，从而避免拥塞。相反 β 接近 1 则表示大流对接收端链路的复用程度高，此时只需对单个大流进行较小程度的拥塞窗口调

节,在实现控制拥塞的同时可以避免链路吞吐量大幅下降。β 参数更新过程如图 4 所示。

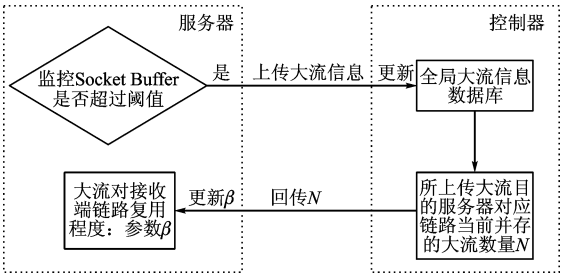


图 4 β 参数更新过程

在大流拥塞窗口的调节过程中,首先当终端服务器检测出大流后,初始化 β 参数为 1,然后将该大流信息上传至控制器;对于套接字缓存大小持续超过设定阈值的大流,为了控制网络开销避免服务器与控制器持续通信,设置 1 s 的间隔时间;控制器负责维护全局大流信息,记录发往每一目的服务器的大流数量,当收到服务器上传的大流信息后,控制器更新全局大流信息,然后向该服务器回传该大流对应目的服务器当前并存的大流数量 N,控制器对大流条目设置 1 s 的生存时间;最后,终端服务器基于 N 值计算 β 参数用于调节大流的拥塞窗口。

3 实验结果与分析

实验采用 NS2(Network Simulation Version 2)仿真软件,并且将使用传统 TCP 中的 New Reno 版本,对 DCTCP 机制与 DFTCP 机制进行对比,来展示 DFTCP 在解决 TCP Incast 拥塞和小流分组高延迟等问题时的性能优势。

在实验过程中,设置所有的链路带宽为 1 Gbit·s^{-1},时延为 50 μs,对应数据中心高带宽、低时延的网络环境。此外,设置 ECN 标记门限参数 K 为 20,而在多对一通信模式中,设置每一台服务器请求固定大小的数据量为 256 kB。主要的实验仿真参数如表 1 所示。

表 1 仿真实验参数

参数	数值
链路带宽/Gbit·s^{-1}	1
链路时延/μs	50
最大传输单元/B	1 500
服务器请求单元/kB	256
门限 K	20

3.1 交换机队列长度变化

实验首先在同一台交换机下设置 6 台服务器,其中将 1 台服务器作为接收端,其他 5 台服务器作为发送端,且各自产生 1 个大流共享接收端瓶颈链路。链路带宽均设置为 1 Gbit·s^{-1},链路时延设置为 50 μs,因此往返时延约为 200 μs。对于传统 TCP,队列模式设置为队尾丢弃,而对于 DFTCP,设置门限值 K 为 20。每隔 50 ms 的时间,检测一次接收端对应的交换机即时队列长度,总的仿真时间为 5 s,仿真结果如图 5 所示。

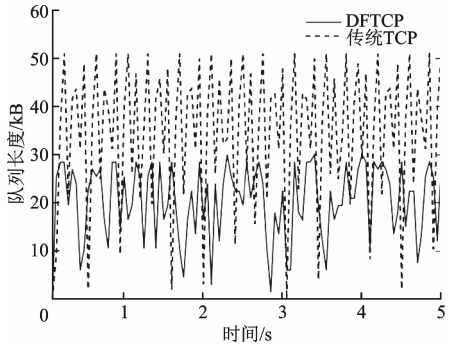


图 5 交换机队列长度变化

仿真结果表明,当多个 TCP 大流竞争同一瓶颈链路时,在接收端对应的交换机队列处,由于传统 TCP 在 AIMD(Additive Increase Multiple Decrease)算法作用下,队列长度变化具有明显的锯齿现象并且变化幅度较大,而使用主动队列管理机制的 DFTCP 机制能够有效控制队列长度波动的幅度,保持相对更加稳定和较短的队列长度。控制交换机队列长度,有利于提高网络处理高突发流量的能力以及减小高突发流量造成的分组丢失。

3.2 TCP Incast 拥塞控制

根据文献[9]建立了数据中心网络环境下典型的 Incast 通信模式,构建了同一台交换机下具有多对一通信模式的多台服务器间的同步读过程。多台服务器发送端向同一个客户端并行地传输数据块,只有当所有发送端完成当前轮的数据块传输后,客户端才会开始请求传输下一轮的数据块。链路带宽均设置为 1 Gbit·s^{-1},链路时延设置为 50 μs,服务器请求单元固定设置为 256 kB,将参与多对一通信的服务器数量作为变量,检测客户端接收到的数据,计算在整个传输过程中客户端应用层感知到的平均吞吐量。仿真结果如图 6 所示。

仿真结果表明,随着参与同步读过程的服务器数量上升,使用传统 TCP 进行多对一通信的客户端应用层吞吐量会出现大幅下降,即 Incast 拥塞。如前所述,由于突发流量造成接收端链路过载,从而导致 TCP 分组丢失。由于同步读过程的特点是只有当所有发送端完成当前轮的数据块传输后,客户端

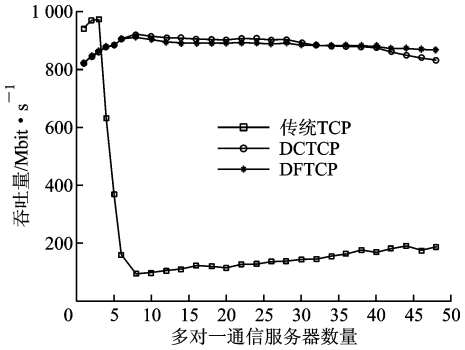


图 6　TCP Incast 拥塞控制对比

才会开始请求传输下一轮的数据块。因此，在分组丢失后的超时重传时间之内会阻塞部分发送端继续进行数据发送，由于 TCP 毫秒级的超时重传时间与数据中心网络微秒级的时延严重失配，从而将导致吞吐量大幅下降。对于采用主动队列管理机制的 DFTCP 和 DCTCP，对 TCP Incast 拥塞的控制取得了相似的实验结果。其原因在于，当交换机队列长度超过阈值时将会传递拥塞信息，促使发送源端提前进行拥塞控制，减少丢包和 TCP 超时的发生，因此能够支持更多数量的服务器同时进行多对一通信而不会出现吞吐量大幅下降的情况。

3.3　小流完成传输时间

实验首先建立两个 TCP 大流共享同一链路，然后再在同一链路下建立一个总共传输为 20 kB 的 TCP 连接作为小流，直到接收端完成 20 kB 的数据传输后才释放该连接。然后，连续重复上述过程建立小流并监测其完成传输时间，整个过程中两个大流连接保持不变。链路带宽均设置为 1 Gbit·s^{-1}，链路时延设置为 50 μs。在传统 TCP、DCTCP 和 DFTCP 3 种拥塞控制机制下，统计 1 000 个传输 20 kB 数据量的小流完成传输时间的累积分布函数，结果如图 7 所示。

从图 7 可以发现：传统 TCP90% 的小流在大约 8 ms 的时间内完成 20 kB 的数据传输；DCTCP 机制 90% 的小流在大约 4 ms 的时间内完成 20 kB 的数据传输；DFTCP90% 的小流在大约 3.6 ms 的时间内完成 20 kB 的数据传输。传统 TCP 小流完成传输时间的分布范围较大，而 DFTCP 机制控制小流完成传输时间的性能优于 DCTCP 和传统 TCP。其主要原因在于，DFTCP 和 DCTCP 在主动队列管理机制的作用下能够有效控制小流分组的丢失，而当拥塞发生时，DFTCP 在终端服务器处调节拥塞窗口的过程中，对于大流进行了更大程度的拥塞退避，因此有利于小流更加快速的完成数据传输。

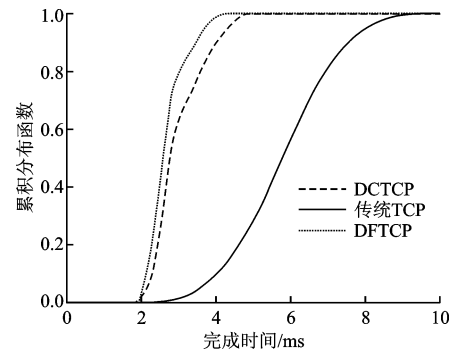


图 7　小流完成传输时间的对比

3.4　收敛性

实验建立 5 个 DFTCP 大流共享同一链路，每一个流顺序地开始和结束传输。每一个流的持续时长为 50 s，流开始数据传输的间隔时间为 10 s，仿真结果如图 8 所示。

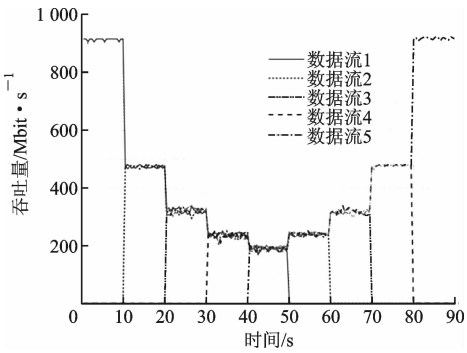


图 8　DFTCP 的收敛性

仿真结果表明，使用 DFTCP 进行数据传输时，随着流启动和结束传输，多个生存时间较长且传输需求大的 DFTCP 流在竞争同一链路时能够快速收敛到对链路公平共享的状态。

3.5　小流划分阈值选取

实验首先建立两个大流共享同一链路，然后再在同一链路下建立 100 个小流，每一个小流传输的数据量在 100 kB 至 1 MB 的范围内随机选取，每当一个小流完成传输后重复上述过程建立新的连接，并随机产生上述范围内待传输的数据量，整个过程中两个大流连接保持不变。划分小流的持续时间阈值为 50 ms，统计在不同小流划分阈值设定下，小流完成传输时间的累积分布函数以及相应的链路平均吞吐量，结果如图 9 所示。

从图 9 结果可以发现，对于传输数据量均匀分布的小流，随着小流划分阈值的上升（图例中从 100 kB 至 1 MB），小流完成传输时间整体呈现减少趋势，而链路利用率呈下降趋势（图例中从 913.2 Mbit·s^{-1} 降至 862.5 Mbit·s^{-1}）。当小流划分阈

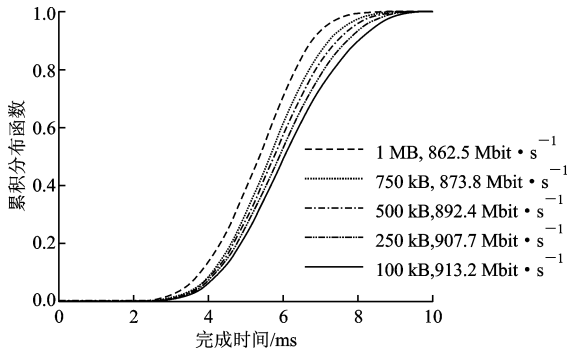


图 9 不同小流划分阈值下的传输时间对比

值设定增大时,一方面提高了小流判定的正确性,同时也使得网络中发生拥塞或者可能发生拥塞时拥有高优先级传输的流的数量增加,这将会导致大流和部分普通 TCP 流更进一步的拥塞退避,进而降低网络吞吐量。因此,在小流传输数据量均匀分布的情况下,小流划分阈值的选取是在网络吞吐量和小流完成时间之间的权衡。

对于小流划分阈值的选取问题,在特定数据中心网络环境中,在应用传输需求已知的情况下,应该保证大多数小流的传输性能。此外,DFTCP 机制对于小流判定中加入流持续时间小于 50 ms 的目的是为了提高小流判定的准确性,避免将网络中持续时间较长但是不活跃的 TCP 流判定为小流。优化的流持续时间阈值的设定也与特定网络环境中所部署应用对应的小流截止时间分布相关。

4 结 论

数据中心网络具有高带宽、低时延的特性,广泛存在的多对一通信模式可能导致 TCP Incast 拥塞造成吞吐量崩溃,并且在混合各种传输性能需求流量的网络环境中,小流的传输容易受到大流的影响,从而形成小流高延迟问题。本文提出了基于流分类的传输控制协议用于数据中心网络的拥塞控制,称为差分流传输控制协议 DFTCP。DFTCP 机制由 3 个部分组成,分别为流分类机制、拥塞信息传递、拥塞退避机制。DFTCP 通过主动队列管理机制控制交换机队列长度,减小突发流分组的丢失来解决 TCP Incast 问题,通过区分不同传输需求的 TCP 流,对于影响网络性能的大流,对其拥塞窗口进行更大程度调节的方式来提高网络处理高突发流量的能力,同时减少小流完成传输的时间,解决小流高延迟的问题。实验结果表明,DFTCP 能够避免 TCP Incast 拥塞导致的吞吐量崩溃,同时能够减少小流完成传输的时间。

参考文献:

[1] ARMBRUST M, FOX A, GRIFFITH R, et al. Above the clouds: a Berkeley view of cloud computing: UCB/EECS-2009-28 [R/OL]. [2016-08-20]. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.

[2] 李丹, 陈贵海, 任丰原, 等. 数据中心网络的研究进展与趋势 [C]// 2011 中国计算机科学技术年度报告: A 卷. 北京: 机械工业出版社, 2012: 54-55.

[3] ALIZADEH M, GRENNBERG A, MALTZ D, et al. Data center TCP (DCTCP) [J]. ACM SIGCOMM Computer Communication Review, 2010, 40(4): 63-74.

[4] CHEN K, BAI W, ALIZADEH M. Scheduling mix-flows in commodity datacenters with Karuna [C]// ACM SIGCOMM'16. New York, USA: ACM, 2016: 174-187.

[5] JIANG C, LI D, XX M. LTTP: an LT-code based transport protocol for many-to-one communication in data centers [J]. IEEE Journal on Selected Areas in Communications, 2014, 32(1): 52-64.

[6] GHEMAWAT S, GOBIOFF H, LEUNG S T. The Google file system [J]. ACM Sigops Operating Systems Review, 2003, 37(5): 29-43.

[7] CHANG F, DEAN J, GHEMAWAT S, et al. Bigtable: a distributed storage system for structured data [J]. ACM Transactions on Computer Systems, 2008, 26(2): 4.

[8] DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1): 107-113.

[9] PHANISHAYEE A, KREVAT E, VASUDEVAN V, et al. Measurement and analysis of TCP throughput collapse in cluster-based storage systems [C]// USENIX FAST 2008. Berkeley, CA, USA: USENIX Association, 2008: 175-188.

[10] CHEN Y, GRIFFITH R, LIU J, et al. Understanding TCP incast through put collapse in data center networks [C]// ACM Workshop on Research on Enterprise Networking. New York, USA: ACM, 2009: 73-82.

[11] VASUDEVAN V, PHANISHAYEE A, SHAH H, et al. Safe and effective fine-grained TCP retransmissions for datacenter communication [C]// ACM SIGCOMM 2009. New York, USA: ACM, 2009: 303-314.

(下转第 152 页)