**基于TCP协议的拥塞控制策略及优缺点分析**

罗凯 郑凯悦 洪宇

(南昌航空大学　江西 南昌 330000)

[摘要]：随着TCP协议在网络环境中的广泛使用，网络的通信量也不断增大，在网络中发送的数据报也就越多，网络中可用的资源(带宽、交换节点的缓存和处理机等)不能满足需求的资源，就会产生网络拥塞。同时，当发送的数据包出现差错时，有时可以直接简单的丢弃，但是简单的丢弃却会导致数据包的重传，这就会出现一些不必要的重传的数据包，导致网络信道的利用率下降。如何有效的解决网络拥塞问题和如何处理一些无法立即解码的数据包或出现差错的数据包，以及如何有效的减少重传次数，成为现如今解决网络拥塞问题的关键。本文就TCP拥塞问题和策略优缺点进行浅析，并提出一些简单的解决网络拥塞问题的解决办法。

[关键词]：计算机网络 TCP协议 拥塞控制

**Research and Discussion on Black-box testing method**

Luo kai Zheng KaiYue Hong Yu

(Nanchang Hangkong University　Jiangxi Nanchang 330000)

[Abstract]：With the widespread use of TCP protocol in the network environment, the network traffic is also increasing. The more data packets sent in the network, the resources available in the network (bandwidth, buffer of switching nodes, processors, etc.) can not meet the needs of resources, which will lead to network congestion. At the same time, when there are errors in sending packets, sometimes they can be simply discarded, but simple discarding will lead to the re-transmission of data packets, which will result in some unnecessary re-transmission of data packets, resulting in a decline in the utilization of network channels. How to effectively solve the network congestion problem and how to deal with some data packets that can not be decoded immediately or have errors, as well as how to effectively reduce the number of retransmissions, has become the key to solve the network congestion problem. In this paper, TCP congestion problem and strategy advantages and disadvantages are analyzed, and some simple solutions to solve network congestion problem are proposed.

[Keywords]: computer network TCP protocol congestion control;

# 引　言

随着计算机网络的飞速发展，Internet上的HTTP，FTP，TELNET，SMTP等数据业务都是使用的TCP协议，由此可见对TCP的拥塞控制是解决网络拥塞的重要方式。拥塞是一种持续过载的状态，用户所需求的资源超过网络的资源数量时，拥塞就会发生，导致网络效率降低，吞吐量下降。当负载达到膝点(Knee)[1]时，吞吐量增长缓慢，响应时间增长。若负载继续增加，便会发生丢包现象。当到达崖点(Cliff) [1]时，吞吐量快速降低，响应时间快速增加，如果发送端仍没有限制数据报的发送就会导致滞留在网络中的数据报持续增多，进而导致“拥塞崩溃”，是网络处于瘫痪状态的主要原因。

# TCP拥塞问题基本理论

由于网络资源具有共享这一属性，网络资源是有限的，而用户的数量随着科技的飞速发展快速增多，就会造成资源的短缺。拥塞的发生就是由于网络资源的缺乏引起的，但是单纯的增加网络资源并不能完全或者较高程度的避免用色的发生。拥塞现象的发生与互联网设计机制有密切联系。拥塞控制就是一种调节

## 网络拥塞产生原因

拥塞是如何形成的？看起来很简单的问题，实际上并不是所有人脑海中都有清晰的图景。概括来说，拥塞的主要成因是数据发送方投放到网络中的数据超出了网络的承载能力，用前文的概念就是Byte in flight 超出网络承载量。就像水管的流量取决于最细的部分，网络的承载能力取决于网络上可用带宽最小路径，也就是有瓶颈所在。网络设备接口往往都有缓冲区，超量的数据无法从瓶颈路径发送，只好在缓存区排队，而缓冲区的大小有限制，积压的数据量超出缓存区大小，就会发生丢弃，这个过程就像漏斗倒水一样。

产生拥塞的原因：网络中存在瓶颈，该瓶颈的吞吐量小于节点单位时间发送的数据量 。接收缓冲空间大于瓶颈处可用缓冲空间。

## 网络拥塞特点

场景的设计与分析已经广泛应用于软件测试领域，事件场景使得测试人员更加明白的理解系统。场景设计技术可以主动具体的描绘出程序中不同输入所触发的事件。同一事件在不同的引发顺序和不同的处理方法下形成了事件流。

在场景测试中，测试人员的角色是软件的使用者，可以从不同角度，顺序运行软件，并将所有的场景可能性罗列形测试用例。基于场景的测试与一般的功能性测试有着很大的区别，场景测试主要是针对程序中各个功能的能否连续并且正确的完成。提取测试场景的分类[1]主要是：(1)顺序结构 (2)平行结构 (3)循环结构

## 互联网基本模型

一个程序的测试用例是无穷多的，所以需要选取某些具有代表性的用例。在较大的软件系统，测试员不可穷举每一个用例来测试，这样人力成本和时间成本耗费太多，在实际测试过程中，一般将软件规模和测试用例个数近似看作线性关系。通过在程序变量的取值范围内按照矩形法进行初步划分，保证每个关键功能测试点都可以被测试，减少线性随机取值产生的测试用例无法发现缺陷的情况，提高发现缺陷的可能性。

# TCP基于源主机的拥塞控制机制

网络拥塞控制是一个复杂的系统，单单从某一方面来处理拥塞无法达到最佳的效果，甚至有可能加剧网络的拥堵状况，TCP拥塞控制机制主要是针对发送方和接收方来进行调整。下面是各种TCP版本的拥塞控制。如图1 [1]所示。

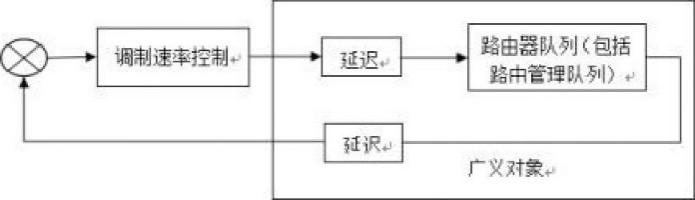


图1. 基于源主机的TCP拥塞控制机制模型

## TCP Tahoe

最早版本的TCP称为Tahoe，其主要有三个机制去控制数据流和拥塞窗口：slow start(慢开始SS)、congestion avoidance (拥塞避免CA)和fast retransmit(快速重传FS)，SS机制:当TCP连接建立时，把拥塞窗口(cwnd)的大小初始化，并设为一个MSS(maximum segment size)，同时把慢开始门限ssthresh (slow start threshold)设为 64 KB。CA 机制: 为了在发生拥塞的情形下控制流量TCP Tahoe 使用AIMD机制。AIMD:只要有一个数据包丢失就认为网络发生拥塞，Tahoe会把ssthrsh 设为目前的拥塞窗口(cwnd) 的一半。并且回到SS的状态，之后拥塞窗口(cwnd) 继续以指数成长；当到达ssthresh 时拥塞窗口(cwnd)会以线性成长来避免拥塞。FS 机制:当收到三个重复的ack 时，不必等到Retransmit Timeout(超市重传时间RTO)，会认为包丢失，并且马上重传。

## TCP Reno

TCP Reno为了提高拥塞恢复的效率，在Tahoe的基础上增添了“Fast Recovery(快速恢复FR)”机制。是目前使用最广泛的版本。FR机制:当收到三个重复的ack 或是超过了RTO 且尚未收到某个数据报的ack，Reno 会认为有数据报遗失了，并且认定网络发生拥塞。Reno 会把ssthresh 设为目前拥塞窗口(cwnd)的一半，但并不会回到SS的状态，而是设定拥塞窗口(cwnd) 为ssthresh，之后拥塞窗口(cwnd) 则维持线性成长。Reno避免了通信链路在重传之后的空闲，提高了网络的吞吐量的传输效率。

## TCP New Reno

New Reno TCP在Reno TCP的基础上对快速恢复算法进行修改，只有一个数据包丢失的情况下，其机制和Reno是一样的；当同时有多个包丢失时就显示出了它的优势。

Reno快速恢复算法中发送方收到一个新的ACK就退出快速恢复状态，New Reno算法中只有当所有报文都被应答后才退出快速恢复状态。New Reno TCP添加了恢复应答判断功能，以增强TCP终端通过ACK报文信息分析报文传输状况的能力。

使TCP终端可以把一次拥塞丢失多个报文的情形与多次拥塞的情形区分开来，进而在每一次拥塞发生后拥塞窗口仅减半一次，从而提高了TCP的顽健性和吞吐量。

## TCP Sack

相比较于New Reno虽然可以解决大量数据报丢失，但是在一个RTT时间内只能处理一个数据报的错误。为了更加高效的处理大量数据包丢失问题。Sack增加了选择确认(Selective Acknowledgements)和选择重传(Selective Retransmission) SACK中加入了一个SACK选项（TCP option field），允许接收端在返回Duplicate ACK时，将已经收到的数据区段（连续收到的数据范围）返回给发送端，数据区段与数据区段之间的间隔就是接收端没有收到的数据。发送端就知道哪些数据包已经收到，哪些该重传，因此SACK的发送端可以在一个RTT时间内重传多个数据包。

## TCP Vegas

相对于其他TCP版本，tcp的vegas算法是基于delay的一个拥塞控制算法，所谓基于delay也就是用RTT测量网络状况的拥塞控制算法。Vegas就是通过观察以前的TCP连接中RTT值的改变情况来控制拥塞窗口cwnd。如果发现RTT变大，那么Vegas就认为网络发生拥塞，并开始减小cwnd；如果RTT变小，Vegas则解除拥塞，再次增加cwnd。这样，cwnd在理想情况下就会稳定在一个合适的值上，并且这个算法不仅作用于拥塞避免状态，而且还作用于slow start状态。使用RTT测量网络状况进行拥塞控制的算法好处在于拥塞机制的触发只与RTT的改变有关，而与包的具体传输时延无关。由于它没有采用包丢失来判断网络可用带宽，而改以用RTT的改变来判断，可以较好地推测出网络带宽使用情况，并且对小缓存的适应性较强，其公平性、效率都较好。

# TCP拥塞控制策略的不足

## 存在时延

端系统从发生拥塞到实施控制之间有着明显的时延，而且端系统缺乏对数据传输过程的动态了解，无法预知网络资源的使用情况，目前主要是通过降低发送到网络的数据流量来减小网络负载从而缓解拥塞。

## ACK Congestion现象

TCP源使用了ACK自计时技术控制数据包的发送速度。发送方根据接收到的ACK确认来触发新数据的发送，所以ACK访人行为也影响到了发送方的速度，但是ACK自计时技术在非对称网中往往会失效。

正常情况下ACK,到达源端的间隔为该链路上数据包的传输时间,如果在返回路径中ACK聚集在路由器的缓冲区中,这样ACK到达源端的间隔就成为ACK的传输时间,由于ACK比数据包要小得多,所以会造成源端以超出路由器承受能力的速度向网络发送数据包。

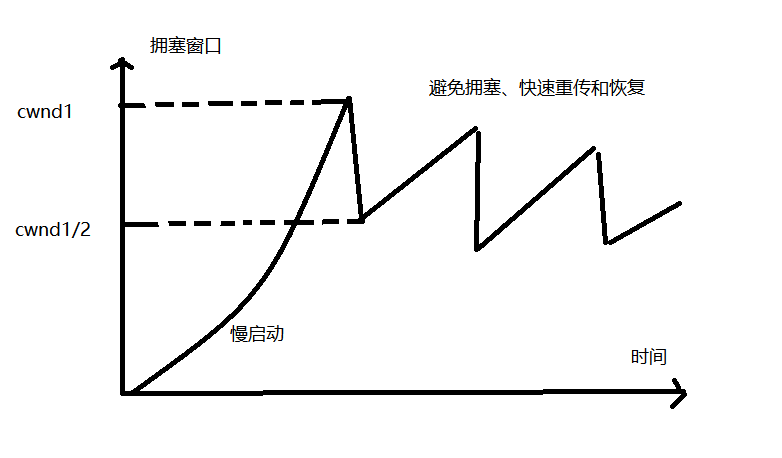


图2 快速重传和快速恢复

## 慢启动算法中存在的问题

首先,数据发送从一个数据包开始,要经过多个RTT才能达到较大的吞吐量,这对流量小但是链路延迟又比较大的TCP流很不利。其次,发送方不了解网络的带宽,但采用指数增长的方式发送数据造成了数据突发,从而引起瓶颈链路的拥塞。

## TCP Reno算法的局限性

TCP Reno算法在源端检测到拥塞后,重传从丢失数据包至检测到丢失时发送的全部数据包,而实际上其中有些数据包是正确传到接受端的,不必重传。TCP Reno在检测单个包丢失时效果不错,但是如果在一个数据发送窗口中出现多个包丢失,由于一个RTT只能重传一个数据包,因此它必须等待重传超时才能继续传输数据,导致它在性能上具有较大的局限性。

## 公平性问题

Internet采取的基于源端的拥塞控制策略是不公平的,TCP拥塞控制中的公平性问题主要表现在以下两个方面：

首先,TCP中拥塞窗口大小可以表示为RIT的函数,比如慢启动阶段cwnd随RTT呈指数级增长,拥塞避免阶段cwnd随RTT线性增长。这样对于长延迟的连接来讲,由于其窗口增长速度慢,因而在同样的条件下获得的带宽也就越少,特别是在慢启动过程中,RTT长的连接的劣势就表现得更为明显。

其次,TCP提供端到端的可靠传输服务,在检测到拥塞后会自觉地降低发送速度,随着网络中不遵守TCP协议的各种数据流的出现,TCP流在与这些数据流共存时,其竞争力较弱,往往会丧失应有的带宽,因此公平地共享带宽也是一个需要密切关注的问题[2]。

# TCP拥塞控制改进

## 优化重传次数

Ahlswede 等在2000年首次提出了网络编码的概念[3]，并从理论上证明：基于网络编码的有线网络能够达到多播传输容量的理论上限。

研究表明，IDNC传输策略具有很多的优势[4]：比如降低重传次数、提高网络吞吐量、减少解码的延迟时间等等/但是传统的IDNC策略对网络编码数据包有着严格的限制，即网络数据包必须被所有的网络终端立即可解码，所以也被称为立即可解码网络编码(Strict Instantly Decodable Network Coding，SIDNC)。

而为了追求更高效的重传策略，提出了广义立即可解网络编码((Generalized Instantly Decodable Network Coding，GIDNC)的概念，此策略不像传统用的IDNC那样严格的要求限制接受数据包的立即可解码性，以及不以编码数据包时以最大化立即可解码的终端的数量为目标，而是在终端仍然坚持即收即解码的原则，正确接受但不能立即解码的编码包通常将会被丢弃。

## 数据包是否被丢弃的原则

对于终端成功接收到的数据包但是不能立即解码的网络编码数据包将被缓存起来，等待将来的解码机会而不是简单的立即丢弃该编码包。因此，引入了网络编码缓存[5](Cache-based Instantly Decodable Network Coding，CIDNC)的概念。

## TCP拥塞控制算法的改进

由于收到重复的ACK可能会导致错误的系统问题，根据网 络负载、吞吐量和延迟的关系，把Rtt的变化和三个重复的ACK 共同控制慢启动的阈值和拥塞窗口的大小。 新的往返延迟会影响之后旧的RTT，这里可引入一个控制因素P。当新的RTT#旧的RTT时，P=0；当新的RTT>旧的RTT时，P=1；当收到3个重复的ACK时候控制慢启动阈值以及拥塞窗 口大小的算法。这个算法中的P值为0或者1，和三个重复的ACK共同控制慢启动阈值和拥塞窗口的大小，使系统能更加准 确地判断数据包有没有丢失，只有增加延迟时间也收到3三个重复的ACK时，能减少慢启动阈值和拥塞窗口为之前的一半，大大提高了资源的利用率[6]。

# 结语

本文阐述了一些对于拥塞控制中存在的问题,特别是在网络控制方面的问题。揭示了现在所面临的拥塞控制问题，进行了一些问题的总结以及一些基本的解决拥塞控制问题的思路和方法，而且对于TCP的优点与缺点简单的进行了阐述，给出了现有的改进技术，如优化重传次数，算法改进等。

在这个网络快速发展的时代，互联网的规模越来越大，网络用户的数量急剧攀升，现有的拥塞控制策略已经逐渐无法满足网络中各种应用的提出的复杂的要求，拥塞已经成为了迫在眉睫的问题。在将来，还需对拥塞控制的策略进行进一步的研究和改进,以便更好地符合各种网络应用的需要、适应网络动态变化的特性,使网络运行在最佳状态，使得用户拥有更好的体验。

**参考文献**

1. 陈文娟.TCP/IP计算机网络拥塞控制问题浅析[J].甘肃科技,2018,34(07):12-15..
2. 赵志豪. 网络TCP/AQM拥塞控制策略[D].江南大学,2005
3. Ahlswede Ｒ，Cai N，Li S-Y Ｒ，et al． Network information flow［J］.IEEE Transactions on Information Theory，2000， 46 6(2):1204-1216.
4. Le A，Tehrani A S，Dimakis A G，et al． Instantly decod

able network codes for realtime applications［C］∥IEEE International Symposium on Network Coding (NetCod)， Calgary，AB，Canada，2013:1-6．

1. 牛腾,张冬梅,许魁,王飞.最小化重传次数的无线网络编码广播重传算法[J].信号处理,2017,33(10):1368-1376.
2. 王荣平.计算机网络拥塞控制算法研究[J].科技风,2016(11):131-132.