



(12)发明专利

(10)授权公告号 CN 106464598 B

(45)授权公告日 2019.04.23

(21)申请号 201580021233.2

(22)申请日 2015.03.31

(65)同一申请的已公布的文献号
申请公布号 CN 106464598 A

(43)申请公布日 2017.02.22

(30)优先权数据
14382146.0 2014.04.23 EP

(85)PCT国际申请进入国家阶段日
2016.10.21

(86)PCT国际申请的申请数据
PCT/EP2015/056962 2015.03.31

(87)PCT国际申请的公布数据
W02015/161990 EN 2015.10.29

(73)专利权人 贝匡特有限公司
地址 西班牙马德里市雷蒙德费尔南德斯维
拉韦尔德街61号5号门邮编28003

(72)发明人 约瑟·洛佩斯·塞拉诺
路易斯·皮内罗·布兰卡
吉列尔莫·洛佩斯·塞拉诺

(74)专利代理机构 北京德恒律治知识产权代理
有限公司 11409

代理人 章社杲 李伟

(51)Int.Cl.
H04L 12/801(2013.01)

(56)对比文件
US 2011002223 A1,2011.01.06,
US 7394762 B2,2008.07.01,
US 2006209898 A1,2006.09.21,
CN 101087244 A,2007.12.12,
CN 102104908 A,2011.06.22,

审查员 石霞

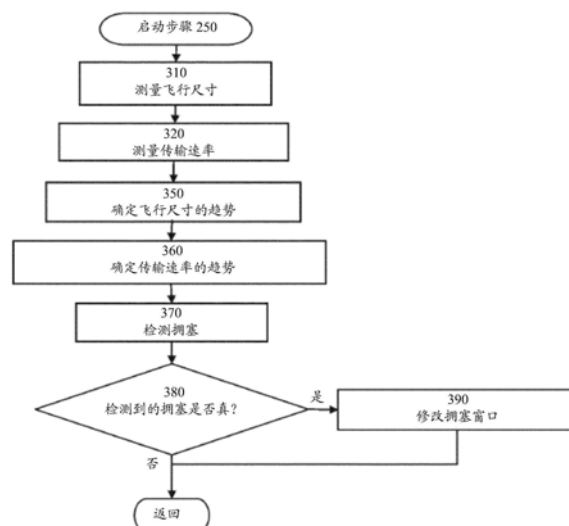
权利要求书4页 说明书27页 附图12页

(54)发明名称

用于基于传输速率梯度的网络拥塞控制的方法和装置

(57)摘要

本发明涉及用于网络的确认通信的拥塞控制。基于飞行尺寸和传输速率这二者的趋势检测拥塞,以根据检测结果来调整拥塞窗口。比如,所述拥塞检测能够区分具有不公平竞争或没有不公平竞争的拥塞。另外,测量的传输速率或其趋势经滤波以补偿时间变化。本发明可以在端节点或代理节点内实现。



1. 一种用于数据通信协议中的拥塞控制的方法,所述数据通信协议采用确认通信,在所述确认通信中传输节点将数据传输至接收节点而且所述接收节点确认接收到所述数据,其中,拥塞窗口指定由所述传输节点在接收到所有或部分所述数据的肯定确认之前传输的未确认数据的最大量,所述方法通过所述传输节点执行并包括:

测量飞行尺寸(310),其中所述飞行尺寸指示由所述传输节点发送的但尚未被所述接收节点确认的数据量;

测量传输速率(320),其中所述传输速率对应于与所述飞行尺寸相同的时刻;

确定所述飞行尺寸的趋势(350);

确定所述传输速率的趋势(360),其中,由传输速率梯度计算推导出所述趋势,其中,所述传输速率测量值和所述传输速率梯度计算的其中之一或这两者都被滤波以减少其临时变化;

根据确定的所述传输速率的趋势和所述飞行尺寸的趋势检测是否存在拥塞(370);

在肯定检测到所述拥塞时,修改所述拥塞窗口(390);并在考虑到所述拥塞窗口尺寸的同时传输数据(270)。

2. 根据权利要求1所述的方法,其中,

检测是否存在拥塞的步骤包括检测是否存在具有不公平竞争的拥塞;以及

在所述飞行尺寸的趋势是增加或是基本稳定并且所述传输速率的趋势在降低时,会肯定检测到所述具有不公平竞争的拥塞,反之,则结果相反。

3. 根据权利要求1所述的方法,其中,

检测是否存在拥塞的步骤包括检测是否存在没有不公平竞争的拥塞;以及

在所述飞行尺寸的趋势增加并且所述传输速率的趋势基本上稳定时,会肯定检测到所述没有不公平竞争的拥塞,反之,则结果相反。

4. 根据权利要求1所述的方法,其中,

检测是否存在拥塞的步骤包括检测是否存在具有不公平竞争的拥塞和没有不公平竞争的拥塞中的任一个,

在所述飞行尺寸的趋势是增加或基本稳定并且所述传输速率的趋势是降低时,会肯定检测到所述具有不公平竞争的拥塞,反之,则结果相反,以及

在所述飞行尺寸的趋势是增加并且所述传输速率的趋势基本稳定时,会肯定检测到所述没有不公平竞争的拥塞,反之,则结果相反。

5. 根据权利要求2或4所述的方法,其中,修改所述拥塞窗口的步骤包括在肯定检测到所述具有不公平竞争的拥塞时,增加所述拥塞窗口尺寸。

6. 根据权利要求3或4所述的方法,其中,修改所述拥塞窗口的步骤包括在肯定检测到所述没有不公平竞争的拥塞时,减少所述拥塞窗口尺寸。

7. 根据权利要求2或4所述的方法,其中,

修改所述拥塞窗口的步骤包括在肯定检测到所述具有不公平竞争的拥塞时,将当前拥塞控制状态改变为新的拥塞控制状态,其中,所述当前拥塞控制状态和所述新的拥塞控制状态分别指定当前的规则和新的规则,其以在拥塞和数据丢失都未被检测到时,用于在接收到数据确认时增加所述拥塞窗口尺寸,以及

所述新的规则指定以比所述当前规则更快的速率增加所述拥塞窗口尺寸。

8. 根据权利要求3或4所述的方法, 其中,

修改所述拥塞窗口的步骤包括在肯定检测到所述没有不公平竞争的拥塞时, 将当前拥塞控制状态改变为新的拥塞控制状态, 其中, 所述当前拥塞控制状态和新的拥塞控制状态分别指定当前的规则和新的规则, 以在拥塞和数据丢失都未被检测到时, 用于在接收到数据确认时增加所述拥塞窗口尺寸, 以及

所述新的规则指定以比所述当前规则更慢的速率增加所述拥塞窗口尺寸。

9. 根据权利要求8所述的方法, 其中,

所述通信协议为TCP,

如果所述当前拥塞控制状态为TCP慢启动, 则在肯定检测到所述没有不公平竞争的拥塞时, 执行从当前拥塞控制状态向新的拥塞控制状态的改变的操作, 否则不执行此操作, 以及

所述新的拥塞控制状态为TCP拥塞避免。

10. 根据权利要求6所述的方法, 其中,

所述通信协议为TCP, 以及

如果所述传输节点处于TCP拥塞避免拥塞控制状态, 则在肯定检测到所述没有不公平竞争的拥塞时, 执行减少所述拥塞窗口尺寸的操作, 否则不执行此操作。

11. 根据权利要求7所述的方法, 其中,

在确定所述飞行尺寸的趋势和所述传输速率的趋势的步骤中, 在时间周期内确定所述飞行尺寸的趋势和所述传输速率的趋势, 所述时间周期开始于从之前的拥塞控制状态到当前拥塞控制状态的最近的改变之前的并接近于此的时刻, 其中, 所述之前的拥塞控制状态和所述当前的拥塞控制状态分别指定之前的规则和当前的规则, 以在拥塞和数据丢失都未被检测到时, 用于在接收到数据确认时增加所述拥塞窗口尺寸, 而且所述之前的拥塞控制状态指定以所述当前拥塞控制状态更快的速率增加所述拥塞窗口尺寸。

12. 根据权利要求5所述的方法, 其中,

在确定所述飞行尺寸的趋势的步骤中, 在时间周期内确定所述飞行尺寸的趋势, 所述时间周期开始于由于检测到的拥塞而产生的所述拥塞窗口尺寸的最近减少之后并最接近于此的时刻, 以及

在确定所述传输速率的趋势的步骤中, 在时间周期内确定所述传输速率的趋势, 所述时间周期开始于由于检测到的拥塞而产生的所述拥塞窗口尺寸的所述最近减少之前并接近于此的时刻。

13. 根据权利要求5所述的方法, 其中,

所述通信协议为TCP, 以及

如果所述传输节点处于TCP拥塞避免拥塞控制状态, 则在肯定检测到所述具有不公平竞争的拥塞时, 执行增加所述拥塞窗口尺寸的操作, 否则不执行此操作。

14. 根据权利要求7所述的方法, 其中,

所述通信协议为TCP,

如果所述当前拥塞控制状态为TCP拥塞避免, 则在肯定检测到所述具有不公平竞争的拥塞时, 执行从当前拥塞控制状态向新的拥塞控制状态改变的操作, 否则不执行此操作, 以及

所述新的拥塞控制状态为TCP慢启动。

15. 一种用于在传输节点中控制数据通信协议中拥塞的装置, 所述数据通信协议采用确认通信, 在所述确认通信中传输节点将数据传输至接收节点而且所述接收节点确认接收到所述数据, 其中, 拥塞窗口指定所述传输节点在接收到所有或部分所述数据的肯定确认之前传输的未确认数据的最大量, 所述装置包括:

飞行尺寸测量单元(1210), 其用于测量飞行尺寸, 所述飞行尺寸指示由所述传输节点发送但尚未被所述接收节点确认的数据量;

传输速率测量单元(1220), 其用于测量传输速率, 所述传输速率对应于与所述飞行尺寸相同的时刻;

飞行尺寸趋势计算单元(1250), 其用于确定所述飞行尺寸的趋势;

传输速率趋势计算单元(1260), 其用于确定所述传输速率的趋势, 其中, 由传输速率梯度计算推导出所述趋势, 其中, 所述传输速率测量值和所述传输速率梯度计算的其中之一或这两者都被滤波以减少临时变化;

拥塞检测单元(1270), 其用于根据确定的所述传输速率的趋势和所述飞行尺寸的趋势来检测是否存在拥塞;

拥塞处理单元(1280), 其在所述拥塞被所述拥塞检测单元肯定检测到时, 用于修改所述拥塞窗口; 以及

传输单元(1290), 其用于在考虑所述拥塞窗口尺寸的同时传输数据。

16. 根据权利要求15所述的装置, 其中, 所述拥塞检测单元被配置为检测是否存在拥塞, 其包括检测是否存在具有不公平竞争的拥塞; 以及

在所述飞行尺寸的趋势是增加或基本稳定并且所述传输速率的趋势降低时, 肯定检测所述具有不公平竞争的拥塞, 反之则结果相反。

17. 根据权利要求15或16所述的装置, 其中, 所述拥塞检测单元被配置为:

检测是否存在拥塞, 其包括检测是否存在没有不公平竞争的拥塞; 以及

在所述飞行尺寸的趋势增加并且所述传输速率的趋势基本稳定时, 肯定检测所述没有不公平竞争的拥塞, 反之则结果相反。

18. 根据权利要求16所述的装置, 其中, 所述拥塞处理单元被配置为修改所述拥塞窗口, 其包括:

在所述拥塞检测单元肯定检测到所述具有不公平竞争的拥塞时, 增加所述拥塞窗口尺寸; 或者

在所述拥塞检测单元肯定检测到所述具有不公平竞争的拥塞时, 将当前拥塞控制状态改变为新的拥塞控制状态, 其中, 所述当前拥塞控制状态和新的拥塞控制状态分别指定当前的规则和新的规则, 以在拥塞和数据丢失都未被所述拥塞检测单元检测到的时候, 用于在接收到数据确认时增加所述拥塞窗口尺寸, 以及所述新的规则指定以比所述当前规则更快的速率增加所述拥塞窗口尺寸。

19. 根据权利要求17所述的装置, 其中, 所述拥塞处理单元被配置为修改所述拥塞窗口, 其包括:

在所述拥塞检测单元肯定检测到所述没有不公平竞争的拥塞时, 减少所述拥塞窗口尺寸;

在所述拥塞检测单元肯定检测到所述没有不公平竞争的拥塞时,将当前拥塞控制状态改变为新的拥塞控制状态,其中,所述当前拥塞控制状态和新的拥塞控制状态分别指定当前的规则和新的规则,以在拥塞和数据丢失都未被所述拥塞检测单元检测到时,用于在接收到数据确认时增加所述拥塞窗口尺寸,以及所述新的规则指定以比所述当前规则更慢的速率增加所述拥塞窗口尺寸。

20.根据权利要求18所述的装置,其中,

所述飞行尺寸趋势计算单元被配置为在时间周期内计算所述飞行尺寸的趋势,所述时间周期开始于由于由所述拥塞检测所检测到的拥塞而产生的所述拥塞窗口尺寸的最近的减少之后并最接近于此的时刻,以及

所述传输速率趋势计算单元被配置为在时间周期内计算所述传输速率的趋势,所述时间周期开始于由于由所述拥塞检测所检测到的拥塞而产生的所述拥塞窗口尺寸的最近的减少之前并最接近于此的时刻。

用于基于传输速率梯度的网络拥塞控制的方法和装置

技术领域

[0001] 本发明涉及网络通信领域,而且尤其涉及用于网络通信中的拥塞控制的方法和装置。

背景技术

[0002] 拥塞控制

[0003] 在网络节点或链路的可用传输容量低于其需要转发的数据速率的任何时候,数据通信网络中都会出现拥塞状况(congestion situation)。比如,图1A示出了数据传输节点110通过通信网络140向数据接收节点120发送数据流150,其中,所述通信网络包括互相连接的网络节点141、142和143和其他可能的网络节点的集合。在这种情况下,数据流150由网络节点141和142转发。图1A进一步示出了另一个数据传输节点111通过相同的网络140向数据接收节点121发送数据流151。然而,在这种情况下,数据流151由网络节点141和143转发。如果数据流150和151的总发送速率大于网络节点141的转发容量(数据流在所述节点中占用相同的空间(coincide)),则网络节点141将变得拥塞,而且数据流150和151都将经历拥塞状况。如果网络节点(如141)具有一些输入缓存容量,则拥塞状况就会引发其缓存容量被持续使用,直到被最终填满。一旦被填满,如果拥塞状况持续存在,则部分数据流150和151将被丢弃,因此,从数据接收节点120和121的角度来看会丢失部分数据流。使用拥塞控制机制来避免、缓解和处理拥塞状况。拥塞控制机制可以是任何层(一般是链路层、网络层、传递层或应用层)的功能部分并且可以位于端节点(如110或111)或中间节点(如141、142或143)。

[0004] 作为现今最广泛使用的通信协议之一,传输控制协议(TCP)使用复杂的拥塞控制机制。在TCP的发展过程中,拥塞控制领域出现了很多先进的技术。因此,下面将要说明TCP中的拥塞控制机制。然而,很多来自TCP的技术发展后来都被其他协议所采纳,如流控制传输协议(SCTP)。

[0005] 标准TCP协议拥塞控制

[0006] 传输控制协议(TCP)在数据通信网络中被广泛使用。TCP由J.Postel在“*IETF RFC 793:Transmission control protocol*,”中(1981年)进行了详细说明并且其全部内容结合于此作为参考,其中,该TCP提供了两个端点之间的可靠数据传输。在关于TCP技术的文献中,端点通常指“主机”。术语“可靠数据传输”指的是这样一个事实:TCP提供能够进行确认数据传输的自动重传请求(ARQ)机制。具体地,如图1B所示,数据传输节点110通过网络140传输数据段101(TCP数据报(datagram)的有效载荷),其中,数据接收节点120校验该数据段的正确到达。随后,数据接收节点120通过网络140将确认102发回数据传输节点110,其中,该确认肯定地确认数据的正确接收。根据所述反馈102或缺少该反馈,数据传输节点110可以重传所述数据。也可以在TCP段中传输该确认,并且该确认可以是累积的,也就是说,TCP段的确认隐含所有的先前连续TCP段的确认。

[0007] 数据包可能会丢失,这意味着其在预定的时间窗口(也就是,时间段)内没有到达

数据接收节点。另外,数据包会经历一些传输错误,在所述数据接收节点处通过标准方式检测所述错误,所述标准方式包括如循环冗余校验(CRC)等的错误检测/纠正码。可以由网络内增加的载荷和/或信道条件的恶化引起延迟或传输错误。由TCP提供的确认机制能够通过重传的方式恢复数据包的丢失和破坏的数据。

[0008] 然而,如果网络经历高载荷,则很多用户重复的重传会导致这种状况的进一步恶化并且网络会变得拥塞。为了避免所述状况和为了处理拥塞,TCP提供了可以在数据传输节点110和/数据接收节点120上实施的一些拥塞控制机制和策略。

[0009] 下文中,将使用标准TCP术语,如上面提到的RFC 793中并且在IETF RFC 5681“TCP Congestion Control,”(2009年9月)中使用所述术语,具体地:

[0010] 主机(Host):网络节点,其为TCP通信的端点。术语“主机”也将被用于应用本发明的其它协议,以表示从那些协议角度来看是通信端点的所述网络节点。

[0011] 连接(Connection):建立在两个主机之间的双向数据流,通过所述双向数据流本身的建立、流控制和拥塞控制机制被单独识别,而与其它数据流无关。

[0012] 拥塞窗口(Congestion Window,cwnd):拥塞窗口表示TCP主机发送超过最新确认的序列数量的连续数据的最大量,可以通过所述发送主机进行局部计算,而不考虑所述接收主机所通告的窗口。

[0013] 接收窗口(Receive Window,rwnd):接收窗口是由接收器在其发回的确认消息中通告给发射器的窗口尺寸。所述窗口尺寸指定接收器准备接受的超过最新确认的序列数量的连续数据的最大量。这是TCP实施流控制所使用的机制,也就是说,防止快速发射器无法适应慢速接收器。

[0014] 发送窗口(Send Window):发送窗口是cwnd和rwnd中的最小值。标准TCP拥塞控制是基于发射器的机制,因此其着重于设定合适的cwnd值和着重于假定cwnd决定发送窗口。

[0015] 慢启动(Slow Start):满启动是TCP拥塞控制状态。在慢启动状态下,TCP拥塞控制算法以指数的方式增加cwnd,其中,每当接收到确认,所述cwnd就增加大约一段尺寸。在使用增加cwnd的方式时,TCP连接中的TCP发送主机被认为处于慢启动。

[0016] 拥塞避免(Congestion avoidance):拥塞避免是TCP拥塞控制状态。在所述拥塞避免状态下,所述TCP拥塞控制算法以比慢启动更慢的方式增加所述cwnd。在标准Reno/New Reno TCP中,所述cwnd以 $1/cwnd$ 个字节(cwnd的倒数)增长以用于每个确认字节,也就是说,cwnd将在确认段之后,以一段的尺寸增长,其中,所述段的尺寸总计为cwnd个字节。不同的TCP变型具有不同的拥塞避免算法。在使用增加cwnd的方式时,TCP连接中的TCP发送主机被认为处于拥塞避免状态。

[0017] 慢启动阈值(Slow Start thresh,ss阈值):慢启动阈值限定慢启动状态和拥塞避免状态之间的转换:其是拥塞窗口值低于慢启动使用的值并超过拥塞窗口根据拥塞避免所增加的值。

[0018] 飞行尺寸(Flight size):飞行尺寸为由所述发射器传输并且未被确认的数据的量(通常以字节进行测量)。

[0019] 缓存过满(Buffer-bloat):缓存过满为通过连接过度使用传输途径中缓存的术语,所述连接使用比用来克服传输途径中内部延迟和带宽调整缓存的窗口更大的拥塞窗口。

[0020] 上面所述的TCP术语也可以用于其他协议中,所述协议可以使用与用于TCP拥塞控制的那些概念类似的概念。

[0021] TCP标准,如上面提到的IETF RFC 5681、IETF RFC 6582“*The New Reno modification to TCP’s fast recovery algorithm,*”(2012年)、IETF RFC 2018:“*TCP selective acknowledgment options*”(1996年10月)、和IETF RFC 6675:“*A Conservative Loss Recovery Algorithm Based on Selective Acknowledgment (SACK) for TCP*”(2012年)(其全部内容结合于此作为参考),所述标准包括拥塞控制算法,用于确定主机的最合适数据发送速率;和数据恢复机制,以允许有效重传由于拥塞或其他原因所丢失的数据。如最近官方的国际互联网工程任务组(IETF)RFC文件所指出的,TCP拥塞控制方法来源于所谓的“Reno”拥塞控制机制(例如,在V.Jacobson,“*Congestion avoidance and control,*”in ACM SIGCOMM Computer Communication Review,1988中所描述的),所述方法有一些提高数据包传输丢失的恢复的附加项并基于以下原则:

[0022] -在通信主机的TCP功能中实施拥塞控制机制,而不依赖于中间网络级和链路级节点。

[0023] -在正确数据段被接收时,从接收端点发送累积确认段,以指示接收的最高连续正确字节的序列数量。

[0024] -如果未被上面定义的“发送窗口”限制,则与cwnd相等的rwnd控制由发射器传输的数据量。

[0025] -在连接的开始时(和在重传超时之后),发射器以最小的拥塞窗口值启动,并且,使用上面所述的“慢启动算法”使其递增。

[0026] -如果不被rwnd或发射器的输出缓存所限制,则慢启动算法使发送率快速升高并超过网络容量,最终导致数据包丢失。

[0027] -在接收到三个复制的确认段时,发射器检测到丢失。在快速丢失恢复程序之后,将拥塞窗口设定为慢启动期间获得的最大值的一半,并将连接切换为拥塞避免,使得拥塞窗口比在慢启动期间增长更慢。

[0028] -在拥塞避免中,遵循加性增、乘性减(AIMD)方案。对应每个确认段以很小的固定量来增加拥塞窗口,并且在检测到拥塞之后以减半方式来减少所述拥塞窗口。在出现数据包丢失(接收到三个连续复制的确认)时,检测到拥塞。即使所述机制迫使拥塞出现,但已在数学上证明的是,一些TCP流共用瓶颈资源而且遵循与增长拥塞窗口类似的AIMD方案,并TCP流最终将以合理方式共用可用带宽。

[0029] -无法利用快速恢复程序恢复非常严重的数据包丢失,从而最终使得重传计时器在TCP发射器中被触发。所述超时将会引起第一次发送,但是在此之后启动慢启动阶段,通过减至一段的拥塞窗口尺寸重传非确认段。在严重拥塞的情况下,所述机制最终使所有的发射器大幅减小其发送速率,这样避免了发生完全拥塞性的崩溃。

[0030] 所述标准TCP(有时叫做Reno或NewReno)具有以下三个局限性:(1) 高速/长延迟网络的低性能,这是因为拥塞避免中的拥塞窗口的低增长率需要很长时间才能达到这些网络所需的拥塞窗口的大尺寸;(2) 由于基于丢失的机制来检测拥塞,过度使用网络缓存(缓存过满)会增加通信主机所经历的延迟,所述基于丢失的机制将拥塞窗口增加至路径中的网络节点的缓存是满的并且因此发送的数据包被丢掉的点,以及(3) 与并行TCP“贪婪”流的竞

争,也就是说,TCP流在其共用带宽瓶颈时,以一种比所述标准TCP更具攻击性的方式将其拥塞窗口升高,而且其将会将大部分可用带宽据为己用并使Reno TCP流“饥饿”。

[0031] 如上面所述,高速/长延迟网络的性能在标准TCP下受到局限,这是因为在这些情况下,拥塞避免中的线性拥塞窗口增长太慢,从而产生显著的未使用的容量。很多TCP变型已经提出了更具攻击性的拥塞窗口增长方案以用于这些情景,像被称为STCP、HSTCP、BIC-TCP、H-TCP、CUBIC和TCP-Hybla的TCP变型保持基于丢失的拥塞检测。所述变型通常成功改良了高速/长延迟网络中TCP的性能,但由于这些方案使用基于丢失的拥塞检测,所以这些方案无法解决缓冲过满的问题。在一些情况下,其也会具有关于与欠攻击性TCP(像标准Reno TCP)共用带宽瓶颈的问题,从而可能被拥塞窗口更具攻击性的增加压制住。

[0032] 基于延迟的拥塞检测以减少缓存过满

[0033] 存在通过利用测量的往返时间(RTT)分析连接的端对端延迟来检查网络拥塞的TCP变型,所述往返时间是从发送段到接收段的确认之间的时间。一些所述变型为TCP Vegas、TCP Vegas-A、TCP New Vegas和FAST-TCP。RTT测量值有时被转化为发送速率估计或传输途径中排队等待(缓冲)的段估计,但实际上,其用于制定决策(decision)的独立变量为RTT测量值。

[0034] 在基于延迟的TCP变型中,RTT的增加被视为拥塞开始的信号,在拥塞避免中,增加还是减少拥塞窗口的决策是基于所述RTT测量而作出的。所述方法通常能成功减少或消除拥塞丢失,因此减少了缓存过满和过度延迟。然而,在与并行的基于丢失的TCP变型竞争时,其遭受严重损害。原因就在于,基于延迟的TCP流早于基于丢失的TCP流检测到拥塞并相应地减少了其发送速率。基于丢失的变型不具有限制和保持增加其发送速率直到其填满所有的中间瓶颈缓存,从而导致用于基于延迟的流的容量变得越来越少。由于在今天的因特网中,压倒性数量的TCP使用基于丢失的拥塞控制,所以这已经称为采用完全基于延迟的变型主要阻碍。

[0035] 混合的基于丢失和基于延迟的变型

[0036] 在使用混合模型的一个早期提议中,已知的TCP-DUAL的混合模型将基于RTT测量的拥塞检测算法添加至基于丢失的Reno拥塞控制,从而触发拥塞窗口成倍的减少。所述方法可以解决缓冲过满为题,但是像上面所述的完全基于延迟的变型,由于基于RTT的延迟检测,使其不能与基于丢失的TCP变型竞争。

[0037] 一些其他的TCP变型(像Compound TCP、TCP Libra、TCP Africa、TCP Veno、YeAH-TCP和TCP Illinois)都使用了含有基于丢失的和基于延迟的拥塞控制的混合模型。基于延迟的拥塞检测被用来调节拥塞窗口增长的攻击性,使得在未检测到拥塞时,产生更多攻击性的增长,这通常解决了与高速/低延迟网络中的Reno/NewReno有关的性能问题。然而,在所有这些变型中,在基于RTT的度量估计存在拥塞时,拥塞窗口将会仍然保持增长(虽然更慢),直到出现数据包丢失,所以,即使在未与其他数据流竞争时,缓冲过满问题仍将持续。

[0038] 另一个TCP变型叫做TCP Vegas+,其是默认使用TCP Vegas的混合模型,但如果检测到有竞争的流时,则其切换至NewReno。这样应该避免了没有与其他数据流竞争时的缓存过满,但仍保留有一些未解决的Vegas问题,例如,高速/长延迟网络中的低性能问题。

[0039] 有带宽或速率估计的TCP变型

[0040] 一个叫做Tri-S的TCP变型是使用传输速率的时移的具有基于速率的拥塞检测的

早期TCP变型。然而,在没有适当滤波传输速率估计或没有测试测量速率的增长或稳定的统计方法的情况下,测量的RTT中的相对大的固有变量会破坏拥塞检测结果。此外,基于实时速率的拥塞检测无法区分完全拥塞情形和TCP流与类似攻击性的TCP流竞争的情形:在两种情形下,拥塞窗口都将增长而且测量的速率将维持不变。由于Tri-S减少了拥塞检测时的拥塞窗口,所以与基于延迟的TCP变型一样,其也无法与有基于丢失的拥塞控制的TCP流竞争。

[0041] TCP变型被表示为TCP-Westwood和TCP-Westwood+,其基于相互确认的时序的复杂测量值或仅基于接收到的确认的速率,在其拥塞控制机制中引入了显式带宽估计。通过合适的滤波,拥塞引发丢失发生时的这些估计用作TCP连接的可用带宽。随后,所述带宽估计和测量的最小RTT一起被用来确定最佳拥塞窗口。从此时起,如Reno的拥塞避免模式将最终使用拥塞窗口尺寸引起拥塞和数据包丢失(因此引发缓存过满),这个时候,将基于新的带宽估计计算新的最佳拥塞窗口。

[0042] 也还存在一些最近的TCP-Westwood的变型,其目的在于具有更具攻击性的窗口增长的高速/长延迟网络(如LogWestwood+、TCPW-A、TCP-AR和TCP Fusion),所述网络更好地调节正在变化的网络带宽,但仍遭受缓存过满的问题并且会压制并行的、欠攻击性的Reno流。

[0043] 从慢启动到拥塞避免改良的转换

[0044] 目前为止,所述的TCP变型集中在拥塞避免阶段期间的行为操作中。然而,尤其在满启动中花费其存在的显著部分进行少量下载时,慢启动到拥塞避免的转换可以是非常重要的。由于发送窗口呈指数式增长的同时将达到拥塞,所以只通过丢失检测慢启动中的拥塞会引起严重的缓存过满问题和数据包丢失。

[0045] 存在在慢启动中的数据包丢失之前检测拥塞并且随后改变为欠攻击性的拥塞避免算法的不同方法。这些方法中的一些使用相互确认的延迟的测量值,由于时间测量值决策和发射器中所需的复杂滤波,所述测量值可能不准确。TCP-Vegas提供了修改过的慢启动,实际上,由于慢启动流量的爆发,所述修改过的慢启动引起了过早的拥塞。“限制性慢启动”是依靠确定传输点的任意常数的实验性的IETF RFC。“自适应启动”作为TCPW-A变型的一部分,其使用估计的带宽以推导慢启动阈值参数,并因此强烈依赖于所述估计的质量。“混合启动”(如今在普遍的Linux操作系统的大多数版本中默认使用)基于RTT测量和相互确认的延迟使用两种启发式算法。除了在与有拥塞的并行的TCP数据流竞争时之外,所述“混合启动”可以起作用,由于并行的流量将从开始就增加RTT,所以从慢启动到拥塞避免的切换将出现过早,从而引起流量的减少。

[0046] 拥塞下与更具攻击性的TCP流之间的公平和竞争

[0047] 在大多数文献中,公平问题是关于更具攻击性的(与拥塞窗口增长有关)TCP变型怎样避免压制欠攻击性的TCP变型。然而,如果确定另一个TCP数据流正以一种更具攻击性的方式与TCP流竞争,则该TCP流变得更具攻击性同样重要。在其他数据流处于慢启动时,如果其他数据流来自相同TCP变型,则可以发生这种情形。解决所述问题的少数TCP变型的其中之一为TCPW-A,其具有如果估计会导致更高的带宽,则增加慢启动阈值参数的机制,但所述机制由另一个机制制约以检测不存在其他TCP流竞争。

[0048] 基于网络的方法

[0049] 目前为止,所述TCP拥塞控制方法依赖于终端主机中所实施的功能,主要依赖于发

射器侧。然而,一些方法依赖于中间网络节点(如路由器和交换机)的功能,所述中间网络节点提醒端点即将发生的拥塞(比如,TCPECN),或拥塞发生时,丢掉(drop)数据包(比如,随机早期检测队列管理算法)。新的主动队列管理算法CoDel(K.Nichols,V.Jacobson,“Controlling queue delay”,Communications of the ACM,vol.55,no.7,pp.42-50,2012)最近被提议用来具体解决缓存过满问题,其也是依赖于中间路由器或交换机中所部署的功能的一部分。由于必须以沿着端对端路径大量安装的路由器和交换机为基础来支持这些方法,所以这些方法的问题是很难部署这些方法。另一方面,如果两个主机支持该功能,则使用基于主机的解决方案是足够的,并且如果仅仅是发射器端功能或接收器端功能,则只需一个终端主机来实施以从中获益。

[0050] 总之,缓存过满是TCP通信中保留的重要问题,其可以引起不必要的延迟和资源利用,由于最广泛使用的TCP变型采用了基于丢失的拥塞检测,这同时阻止使用可以缓解缓存过满问题的基于延迟的变型的使用。几乎所有的使用基于带宽的和基于速率的算法以确定拥塞窗口的TCP变型仍使用基于丢失的拥塞检测,所以仍保留有缓存过满的问题。在少数方法中,基于速率的拥塞检测被用于减少拥塞窗口,测量滤波和稳定的统计方法正在消失,从而没有正确解决了测量值的变化性、没有很好检测到拥塞。实际上,只要使用基于丢失的TCP变型(并且今天基于丢失的TCP变型被广泛应用),对于与其竞争的TCP流来说,缓存过满就是不可避免的。然而,存在该连接不与容量瓶颈上的其他连接竞争的很多情形,其中,消除缓存过满将是非常有益的。

[0051] 此外,通常尝试在慢启动到拥塞避免转换中基于延迟度量来减少缓存过满,这在面对来自其他数据流的竞争时会导致低流量。总的来说,在TCP流没有适当地反应以与并行TCP数据流竞争时,缺乏检测拥塞和竞争的良好机制在一些情况下会引起缓存过满问题并且在其他情况下会导致性能问题。

发明内容

[0052] 基于上面总结的现有技术的缺点,有利于提供一种拥塞控制机制以在避免缓存过满的同时能够有效处理拥塞状况和并行数据流之间带宽的竞争。

[0053] 这通过独立权利要求书的特征来实现。

[0054] 本发明的有利的实施例是从属权利要求书的主题内容。

[0055] 本发明的特别方法是基于飞行尺寸和传输速率这二者的趋势检测拥塞,以根据检测结果调整拥塞窗口。

[0056] 所述方法具有可靠的拥塞检测和区分有公平竞争的拥塞与没有公平竞争的拥塞之间的可能性的优点,以及从不同处理中获益的状况的优点。在数据流不断地占用越来越多的拥塞限制的带宽时,与并行数据流的竞争被认为是不公平的。此外,结合确定传输速率的趋势和/或飞行尺寸的趋势所应用的滤波确保在相对于测量值的临时变化稳定地执行存在拥塞的决策。

[0057] 根据本发明的第一方面,提供一种数据通信协议中拥塞控制的方法,所述数据通信协议采用确认通信,其中,传输节点将数据传输给接收节点,接收节点确认接收到数据,其中,拥塞窗口指定所述传输节点在接收到所有或部分数据的肯定确认之前进行传输的未确认数据的最大量,所述方法通过所述传输节点执行并包括:测量飞行尺寸,其所述飞行尺

寸指示由所述传输节点发送的但尚未被所述接收节点确认的数据量；测量传输速率，其所述传输速率对应于与所述飞行尺寸相同的时刻；确定所述飞行尺寸的趋势；确定所述传输速率的趋势，其中，所述趋势来源于传输速率梯度计算，在所述计算中，所述传输速率测量和所述传输速率梯度计算的其中之一或两者都被滤波以减少其时间变化；根据所述传输速率的趋势和所述飞行尺寸确定的趋势检测是否存在拥塞；在肯定检测到所述拥塞时，修改所述拥塞窗口；并在考虑到所述拥塞窗口尺寸的同时传输数据。

[0058] 根据本发明的另一个方面，提供一种用于在传输节点中控制数据通信协议中拥塞的装置，所述数据通信协议采用确认通信，在所述确认通信中传输节点将数据传输至接收节点而且所述接收节点确认接收到所述数据，其中，拥塞窗口指定所述传输节点在接收到所有或部分所述数据的肯定确认之前进行传输的未确认数据的最大量，所述装置包括：飞行尺寸测量单元，其用于测量飞行尺寸，所述飞行尺寸指示由所述传输节点发送但尚未被所述接收节点确认的数据量；传输速率测量单元，其用于测量传输速率，所述传输速率对应于与所述飞行尺寸相同的时刻；飞行尺寸趋势计算单元，其用于确定所述飞行尺寸的趋势；传输速率趋势计算单元，其用于确定所述传输速率的趋势，其中，所述趋势来源于传输速率梯度计算，在所述传输速率梯度计算中，所述传输速率测量和所述传输速率梯度计算的其中之一或两者都被滤波以减少时间变化；拥塞检测单元，其用于根据确定的所述传输速率和所述飞行尺寸的趋势来检测是否存在拥塞；拥塞处理单元(1280)，其在所述拥塞被所述拥塞检测单元肯定检测到时，用于修改所述拥塞窗口；以及传输单元，其用于在考虑所述拥塞窗口尺寸的同时传输数据。

附图说明

[0059] 本发明的方法和装置的示例性实施例在附图中以示例的方式示出，其中，相同的参考数字表示相同或类似的元件，以及其中：

[0060] 图1A是示出网络通信的方框图；

[0061] 图1B是示出两个网络节点之间的确认通信的方框图；

[0062] 图1C是示出通过代理节点互连的两个网络节点之间的确认通信的方框图；

[0063] 图2是示出在处理传入(incoming)确认段时被调出的步骤的流程图，其示出本发明的实施例怎样适合现有的TCP实施方式；

[0064] 图3是示出本发明的方法的可能实施例中的主要步骤的流程图，其仅包括本发明的5个检测步骤中的一个；

[0065] 图4是示出本发明的方法的另一个实施例中的主要步骤的流程图，其以协作方式包括本发明的5个拥塞检测类型；

[0066] 图5是示出数据传输速率滤波(filtering)步骤的实施例中主要步骤的流程图；

[0067] 图6是示出传输速率梯度滤波步骤的实施例中主要步骤的流程图；

[0068] 图7是示出用于确定飞行尺寸的趋势的步骤的实施例中主要步骤的流程图；

[0069] 图8是示出用于确定传输速率的趋势的步骤的实施例中主要步骤的流程图；

[0070] 图9是示出用于检测拥塞的步骤的实施例中主要步骤的流程图；

[0071] 图10是示出用于修改拥塞窗口的步骤的实施例中主要步骤的流程图；

[0072] 图11是示出用于确定传输速率的趋势的步骤的可选实施例中主要步骤的流程图；

- [0073] 图12是示出实现本发明的装置的方框图。
- [0074] 图13是示出实现本发明的装置的操作系统核心中实施方式的方框图；
- [0075] 图14是示出实现本发明的装置的操作系统核心外实施方式的方框图；以及
- [0076] 图15是示出实现本发明的装置的方框图。

具体实施方式

[0077] 有效数据包切换的通信需要一些网络节点中的缓存,以允许切换为适应不同的链路速率和吸收暂时的传输爆发。然而,使用基于丢失的拥塞控制的协议倾向于使用传输路径中的所有可用缓存,超过有效通信必需的可用缓存的话,将会浪费内存资源和引起不必要的延迟,并导致缓存过满。实际上,由于在一些情形下,试图限制缓存过满的数据流将被那些未限制缓存过满的数据流所压制,所以与在拥塞条件下与其他通信数据流竞争时一样,缓存过满在特定情形下是不可避免的。然而,对于拥塞的资源来说,在不与其他的数据流竞争时(很常见的状况),可以避免缓存过满。

[0078] 本发明提供用于网络拥塞控制的方法和装置,所述方法和装置在可以避免缓冲过满问题时,解决缓冲过满问题而不以流量为代价。本发明增加了传输速率和飞行尺寸趋势的估计和拥塞检测机制作为另一项有利特征,所述拥塞检测机制是稳定的而不是具有可观察到的可变性。本发明的特别实施例指定另外的方法,以使用飞行尺寸和传输速率的测量趋势来检测拥塞,并能够区分不同拥塞状况。因此,对具有不同感知的进攻性的竞争数据流的拥塞状况可以进行不同处理。

[0079] 可以在TCP协议内容实施本发明,并且一些实施例也构想合并到TCP拥塞处理算法中。尽管如此,本发明绝不限制在TCP中的实施并且适用于提供确认传输和使用拥塞控制的任何通信协议,如流控制传输协议(SCTP)、数据报拥塞控制协议(DCCP)或其他协议。此外,本发明适用于任何协议层并不限于传输层。

[0080] 根据本发明,飞行尺寸和传输速率都被测量并都被用于确定其趋势,通过这种方式来确定是否存在拥塞状况。基于被测量的拥塞,选择修改传输窗口尺寸的策略。

[0081] 传输速率的趋势来源于其梯度。所述梯度可以是采集所述趋势的任何度量。比如,所述梯度可以是两个时间点所测量的值之间的差异、两个时间点的估计之间的差异(如测量集合的平均值、中值和指数加权平均值)、拟合测量值序列的曲线的斜率的估计、趋势估计统计测试的几个连续梯度计算或度量的(加权)平均值(如众所周知的Jonckheere的趋势测试)。总之,所述梯度可以是指示值序列具有向上趋势、稳定趋势还是向下趋势的任何度量。在任何情况下,从可以潜在包括非常显著数量的噪音的测量值开始,尤其在传输速率测量值的情况下,反复地实时测量所述度量。因此,输入传输速率测量值和传输速率梯度(趋势度量)中的两者之一或两者都被滤波,以减少噪音的影响,该噪声会以其他方式阻碍拥塞检测。

[0082] 类似地,可以确定飞行尺寸的趋势作为任何测量以表示飞行尺寸的发展或实时滤波的飞行尺寸。特别地,可以使用如上所述的梯度。

[0083] 在方法和装置中实现的本发明提供在瓶颈时与基于丢失的TCP流竞争的可能性,在这种情况下,其倾向于与竞争流的进攻性相匹配,而不是压制竞争流。在在瓶颈时不与“贪婪”数据流竞争时,一旦达到传输速率物理限制,倾向于限制其拥塞窗口,因此,减少缓

存过满而不用危害流量。从发射器角度来看,所述方法用于测量飞行尺寸(也就是,使用的窗口)和通过所述飞行尺寸获得的传输速率,随后使用统计稳定技术确定其趋势(时间发展),并最终在检测到连接处于预定拥塞状况时,作用于拥塞窗口。根据有利的实施例,存在如下所述的5个具体拥塞状况(condition,又称为条件或状态)。本发明适用于TCP发射器(主机),其被添加检测前面所述的状况的能力,并且在检测到所述状况其中之一的事件时,适当地改变了拥塞窗口或控制拥塞窗口增长的算法。为了对测量值(尤其是传输速率测量值)中存在的大量变化性计数,本发明采用统计学上稳定技术并对传输速率趋势的估计进行滤波。

[0084] 可以检测到和操作的5种具体状况为:(1)慢启动中没有不公平竞争的拥塞,(2)拥塞避免中没有不公平竞争的拥塞,(3)拥塞避免中具有不公平竞争的拥塞,(4)拥塞窗口减少之后具有不公平竞争的拥塞,以及(5)在慢启动到拥塞避免转换之后,具有不公平竞争的拥塞。应所述注意的是,本发明不限于估计上述所有状况和适当地处理其状况。然而,根据本发明,可以通过估计飞行尺寸的趋势和传输速率趋势,以区分上述的任何上述状况。因此,对上述5种状况中的每一种的检测和处理都构成一个独立的实施例。此外,本领域的技术人员清楚的是,这些特别实施例可以互相结合。

[0085] 虽然上面所述状况采用的术语指的是术语“慢启动”和“拥塞避免”,但是本发明不限于TCP并被用于其他拥塞控制机制或装置中所对应的状态(如上面背景部分或对应的TCP说明书所确定的)。此外,通常可以检测拥塞状态而不用区分如慢启动或拥塞避免这样的拥塞状态。尽管如此,但是对状态的区分进一步提升了拥塞控制机制对应网络状况的更出色的调整度。

[0086] 术语“不公平竞争”与执行拥塞检测的发送主机以及其当前状态有关,也就是,其指示在当前状态中,竞争性数据比发送主机更具攻击性。术语“更具攻击性”的意思是竞争性数据流增加其拥塞窗口(或其传输速率)的速率比发送主机要快。因此,发送主机可以检测状况(1),也就是,在与也处于TCP慢启动状态下的其他数据流竞争时,慢启动中没有不公平竞争的拥塞。由于其正与发送主机一样具有攻击性,所以竞争将可能不视为不公平的。然而,如果发送主机切换至拥塞避免,由于所述数据流随后将比发送主机更具攻击性,则所述发送主机将保持在慢启动状态的竞争流视为“不公平竞争”。

[0087] 术语“拥塞”也与发送主机有关,就意义而言,其指的是根据测量和估计的某些数据传输参数,由发送主机检测的拥塞症状。因此,如果满足了基于测量的参数的某些预定的状况,则拥塞仍被视为肯定检测到。

[0088] 有利地,测量的参数为飞行尺寸和传输速率以及随着时间推移估计的这两个参数发展(也就是,趋势)。

[0089] 本发明的一个实施例可以通过检测飞行尺寸中同时的增加和传输速率的稳定性(肯定)检测状况(2),也就是,拥塞避免状态中没有不公平竞争的拥塞。如果肯定检测到状况(2),则通过减少拥塞窗口来处理该拥塞。在缺少接收器窗口或主机缓存限制时,拥塞窗口将支配随后可以支配飞行尺寸的发送窗口。因此,减少拥塞窗口将倾向于减少飞行尺寸,这反过来将倾向于缓解缓存过满,

[0090] 可选地或除此之外,本发明的实施例可以通过检测飞行尺寸的的同时的增加和传输速率的稳定性(肯定)检测状况(1),也就是,在慢启动中没有不公平竞争的拥塞,通过慢启

动到拥塞避免状态的改变来处理该拥塞。这样将倾向于使飞行尺寸增长变慢并因此倾向于缓解缓存过满。

[0091] 如果不实施慢启动和拥塞避免之间的区别,则可以仍然由检测飞行尺寸的增加和同一时间基本未改变的传输速率,以执行对没有不公平竞争的拥塞的肯定检测。所述处理可以包括减少拥塞窗口或减少拥塞窗口增长的速度。

[0092] 可选地或除此之外,本发明的实施例可以检测状况(3),也就是,在拥塞避免中具有不公平竞争状况的拥塞,其中,存在拥塞并且存在以更具攻击性方式(如通过使拥塞窗口增长更快的方式)竞争容量瓶颈资源的其他数据流。通过检测飞行尺寸同时的增加和传输速率的降低,在拥塞避免(CA)状态下可以检测所述具有不公平竞争的拥塞。更具攻击性地增加拥塞窗口或拥塞避免到慢启动的切换可以有益地处理CA状态中的不公平竞争的肯定检测。

[0093] 在缺少竞争性数据流时,基于速率的拥塞检测(如基于延迟的算法)可以检测到其发生时的拥塞,所述拥塞发生通常比拥塞引发的丢失发生的更早,并且可以在此时通过减少拥塞窗口或将慢启动转换至拥塞避免来避免缓存过满。基于飞行尺寸和传输速率的实时发展,检测拥塞的优点之一在于:尤其是在拥塞避免中,区分具有不公平竞争的拥塞和没有不公平竞争的拥塞。相应地,每个情况下采取的措施可以完全不同,从而提升拥塞处理的效果。

[0094] 如果存在使用基于丢失的拥塞检测的竞争流,则不可能避免缓存过满。如果在(通过其自有方法的流发射器)检测到拥塞时,数据流中的其中之一限制其拥塞窗口,则随后其将被基于丢失的竞争数据流所压制,从而竞争数据流保持增长其飞行尺寸,直到所有网络缓存变满。在拥塞避免中,在达到瓶颈容量时,如果并行的数据流更具攻击性地增长,则根据上面所述的实施例中基于速率的不公平竞争检测则将检测到降低的传输速率与增加的飞行尺寸,并且拥塞窗口将被增加或切换至更具攻击性的算法,如慢启动。这两项操作将使数据流更具攻击性并帮助其与更具攻击性的数据流更好地竞争。

[0095] 在主机检测到拥塞时(通过本发明的方法或元件,或者通过像基于延迟的机制一样的其他方法,或者通过接收到三次复制确认),除了会出现拥塞状况之外,也会出现一个或多个数据流以类似的攻击性窗口增长速率竞争可用带宽。到此时,类似更具攻击性数据流可能已经与大致恒定的传输速率共用拥塞瓶颈,与此同时,各自的飞行尺寸保持并行增长(使用可用网络缓存)。作为拥塞检测的结果,在减少拥塞窗口或将慢启动的连接切换至拥塞避免时,目前欠攻击性的数据流将损失一些被竞争性数据流所占用的传输速率。由于从数据流角度看,减少攻击性是不公平的,所以在本发明中这种状况就叫做“不公平竞争”。可以通过发送侧后验(*posteriori*)估计降低拥塞窗口或切换至拥塞避免的直接结果来检测具有不公平竞争状况的拥塞。

[0096] 本发明的一个实施例可以在减少拥塞窗口之后(在拥塞的肯定检测之后)的短时间段内,在飞行尺寸不小于拥塞窗口减少的数量,同时在传输速率更低时,肯定地检测在拥塞窗口减少之后“具有不公平竞争的拥塞”(也就是,上面所述的状况(4))。在检测状况(4)时,可以有益地不减少拥塞窗口,也就是,可以对其进行相反操作。例如,如果竞争数据流在拥塞避免中以相同速率增长其拥塞窗口,则上述的基于速率的没有不公平竞争的拥塞的检测(2)最终肯定检测拥塞并减少拥塞窗口,从而由于其他数据流将填满已经放弃的缓存,所

以将减缓减少拥塞窗口的数据流。然而,根据实施例处理状况(4),可以检测传输速率的减少,并且通过将扣除的值添加回给拥塞窗口对拥塞窗口的减少进行相反操作。

[0097] 可选地或除此之外,本发明的实施例可以在慢启动转换至拥塞避免之后的短时间段(在拥塞检测之后)内飞行尺寸不是更小、慢启动转换至拥塞避免之后存在传输速率的减少时,肯定检测在将慢启动转换至拥塞避免之后“具有不公平竞争的拥塞”(对应于上述的状况(5))。如果检测到状况(5),有利地,可以将连接切换回慢启动。以这样的方式,验证慢启动到拥塞避免的转换的检测能够确定如果慢启动中存在竞争的基于丢失的数据流,则存在传输速率的减少,这种情况下数据流返回慢启动。在后一个情况下,由于慢启动算法增长非常快,所以有益于暂时保持有值的变量,所述值表示如果未出现到拥塞避免的切换时,则拥塞窗口将会达到的尺寸。相应地,如果需要切换回至慢启动,则可以将拥塞窗口设定为存储在所述变量中的值。

[0098] 通过检测上述5种状况和其相关操作的能力将本发明结合到TCP实施方式,提出了解决缓存过满问题的TCP实施方式。然而,一些TCP实施方式可以获益于仅仅结合5种状况检测能力(和相关操作)的子集。同样实施方式应用于可以在其他层上实施的而不是在传输层上实施的其他协议的拥塞控制机制。

[0099] 在其他可能性的情况下,如图1C所示,本发明的方法可以被实施为增加TCP连接的末端主机的TCP功能或添加TCP代理的TCP功能。TCP代理节点130是用作经由TCP通信的两个末端主机160和170之间的中间节点的网络节点。TCP代理130相对于末端主机160和170以透明的方式,终止与两个末端主机的TCP连接。此类代理节点130可以以透明的方式将一个末端主机发送的信息转发至另一个末端主机(图1C中的105和106),或可以试图通过以某种方式修改所述信息以添加值。类似地,本发明的方法可以应用于除了可以使用代理节点的TCP之外的其他协议。由于不需要改变末端主机160和170的配置或实施方式以从本发明的优点获益,所以提供了在代理节点130内实现本发明是有益的。

[0100] 应所述注意的是,TCP包括除拥塞控制之外的很多方面。本发明不需要改变所述其他方面。因此,本发明可以轻易适应已有的具有基于丢失的拥塞检测的TCP实施方式,并且在以下将详细说明的处理过程中的某一处添加某些步骤。

[0101] 图2是示出根据本发明在接收到210新的确认段(新ACK,new ACK)时执行处理逻辑的流程图。

[0102] 所述图同样指示本发明的实施例怎样适合已有的TCP实施方式。“新ACK”指的是累计第一次确认先前由所述新ACK段的目标主机所发送的TCP用户数据的ACK段(也就是,所述数据此前未被累计确认过)。应所述注意的是,即使在每次接收到新ACK之后,有益于执行所述方法,但本发明不限于这种实施方式。一般来说,实施本发明,从而执行比每次接收新ACK更少的次数执行所述方法。即使在接收到新ACK时,也可以不执行该方法,而是根据内部时钟周期性地执行所述方法,从而能够估计传输速率。

[0103] 除了源端口、目标端口、序列号、窗口尺寸和校验和之外,TCP协议头部其中还包括如ACK的标记(指示确认字段的显著性(significance of the acknowledgement field))、FIN(没有来自发射器的更多数据)、SYN(同步序列号)、RST(连接的重启)等(如上面引用的RFC 793详细所述的)。在TCP实施方式中,新ACK为有效TCP段,有效TCP段的特征为:具有在TCP头部设定的标记并不具有TCP头部设定的FIN、SYN和RST中任一个。步骤220指的是处理

TCP协议(或其他协议)执行的新ACK段,直到RTT的估计。一般来说,步骤220可以包括从网络接口读出段、对其数据层和网络层头部进行解码、验证段是否是正确形成的TCP段等等。步骤220中的术语“SND.UNA”指的是TCP连接状态参数,即,是未被累计确认的最低字节的序列号,其被更新为接收的新ACK消息中的确认序列。参数拥塞窗口表示拥塞窗口尺寸的字节。作为所述逻辑的一部分,TCP主机通常计算RTT(步骤230)。RTT是往返时间:自发送特定TCP段开始,直到第一确认段被接收的时间,所述第一确认段确认发送段并估计通信的主机间的端对端往返延迟。

[0104] 在计算RTT的步骤230之后,执行拥塞和竞争检测步骤250。如果TCP拥塞控制处于慢启动或拥塞避免(或与这些模式等同的模式)但不处于快速回复或快速重传模式,则只执行这些步骤。这通过决定步骤240示出。在一些TCP实施方式中,没必要为所有新确认(ACK)计算RTT。尽管如此,还是在这种情况下,在相同的步骤250中可以执行拥塞和竞争检测,可以使用从计算RTT的之前的(比如,最近的)新ACK获得的最近的RTT值。在本发明的实施例中,假定在拥塞和竞争检测250之前,在步骤220中设定SND.UNA。然而,也可在步骤250之后设定SND.UNA,在所述情况下,步骤250将使用新ACK中的确认序列而不是SND.UNA。

[0105] 步骤260包括通过TCP实施方式所需要的用于新ACK的所有剩余的处理逻辑。所述处理不是本发明的一部分,因此也不是不需的。分别在步骤220和260处对新ACK进行TCP处理的方式不会影响本发明:任何步骤220或260可以包括,例如,更新与TCP相关的计数或根据拥塞避免或慢启动增加拥塞窗口。在步骤270中,考虑拥塞窗口尺寸的同时传输数据,其包括估计发送窗口(作为拥塞窗口、接收窗口和发送主机中输出缓存限制的的函数),随后发送出TCP段,所述TCP段的末端序列号小于SND.UDA和发送窗口的总和。所述这些段来自TCP段的队列,该TCP段的队列其以与接收的新ACK的队列相反的方向等待被传输。在步骤270之后,可以可选性地执行RTT估计260之后的新ACK处理的一部分。

[0106] 拥塞控制实施已可以包括确定丢失发生之前,从慢启动到拥塞避免的转换的方法,也就是,在慢启动期间用于将慢启动阈值(ss_thresh)的值与当前拥塞窗口的值设定为在相同的慢启动阶段期间的测量值的函数的方法。在所述情况下,根据本发明的实施例,在接收新ACK时被调用的全部已有方法可以被慢启动中拥塞控制的对应步骤代替。

[0107] 如上所述,拥塞检测是基于对传输速率和飞行尺寸的测量。传输速率可以是发送速率,或是确认速率。发送速率是发送新TCP用户数据的速率。确认速率是确认新TCP用户数据的速率。

[0108] 另外,在发送包括新用户数据的TCP段(即,TCP数据包)时,如果确认速率被用作传输速率,则优选在发送段时记录序列号和SND.UNA值,从而在接收到发送数据的确认时,可以读出记录的SND.UNA值,以指示发送确认数据时的SND.UNA值。可以记录用于所有被发送的新数据段或仅作为其子集的值,但是在后一种情况下,计算确认字符的精度将会丢失。

[0109] 图3示出了本发明的方法的一个可能实施例,其实施了5种可能拥塞检测步骤的其中任何一个步骤(和各自的相关操作)。具体地,提供一种用于采用确认通信的通信协议中拥塞控制的方法,在所述确认通信中传输节点将数据传输至接收节点而且所述接收节点确认接收到所述数据,其中,拥塞窗口指定所述传输节点在接收到所有或部分所述数据的肯定确认之前可以传输的未确认数据的最大量。所述方法包括测量飞行尺寸(步骤310),其中,所述飞行尺寸指示由所述传输节点发送的但尚未被所述接收节点确认的数据量;测量

传输速率(步骤320),其中,所述传输速率对应于与所述飞行尺寸相同的时刻;确定所述飞行尺寸的趋势(步骤350);确定所述传输速率的趋势(步骤360);以及根据确定的所述传输速率的趋势和所述飞行尺寸的趋势检测判定是否存在拥塞(步骤370)。如果在判断步骤370(检测步骤)中,肯定检测到拥塞,则进行修改拥塞窗口的步骤390。随后在步骤270中,在考虑拥塞窗口尺寸的同时,传输待传输的数据。在图2的情况下,图3示出了在计算RTT(步骤230)之后,作为新ACK的处理的部分,执行拥塞和竞争检测步骤250的可能实施方式。

[0110] 除了如图3所包括的步骤外,存在在如图3所述步骤之后执行的又一个步骤,即,如以上所述的在考虑拥塞窗口尺寸的同时传输数据(步骤270)。图3所示的实施例的前两个步骤为:测量飞行尺寸310和测量传输速率320。

[0111] 测量飞行尺寸步骤310获得进一步用于检测拥塞并且可能用于区分具有不公平竞争的拥塞和没有不公平竞争的拥塞的飞行尺寸。可以如下所述测量(计算)飞行尺寸。如果使用数据传输速率的发送速率,则可以在新ACK被处理时,通过SND.NXT和SND.UNA之间的差来获得飞行尺寸的适当的样本。作为处理新ACK的最终结果,将正常发送新段。因此,一旦这些新段已被发送出,可能会使用对SND.NXT的估计以代替SND.NXT。SND.NXT为TCP连接状态参数、接下来的未发送段的第一序列号。

[0112] 如果使用确认速率作为数据传输速率,则将在确认段被发送时,估计飞行尺寸。为了能够进行这样的估计,本发明的实施方式可以在发送相应段时,保留SND.UNA参数的记录(也就是,存储),以用于所有发送的未确认段或用于其子集。随后,可以,通过从处理新ACK时的SND.UNA中减去在发送的被确认的最后段时所存储的SND.UNA来获取发送的但未被确认的字节数的样本。可选地,为了获得最正确的估计,代替使用发送被确认的最后段时(存储)的SND.UNA,在发送第一段被确认时的SND.UNA用于计算(尤其在使用延迟的确认时,其可以不同于被确认的最后段)。此外,在发送段时存储SND.UNA参数时可以有利的,在连续发送几段作为接收的ACK段的响应,作为突发性增加,以存储发送多少个连续段,从而在计算飞行尺寸时,可以将同样的飞行尺寸分配给作为突发性增加而发送出的任何段的确认,也就是说,所述飞行尺寸对应于作为突发性增加的一部分所发送的最后段。

[0113] 可选地,可以使用拥塞窗口参数cwnd本身作为飞行尺寸的估计,从而产生类似于使用确认速率的结果。然而,在通过接收侧通知的窗口或由发射器组中的缓存极限值限制飞行尺寸时,拥塞窗口可以非常不准确地表示飞行尺寸。

[0114] 在步骤320中,计算的测量的传输速率可以作为由测量的RTT所划分的上述计算的飞行尺寸。数据发送速率或确认速率的获得依赖于使用哪个飞行尺寸。由于使用数据确认速率更好地反应了端对端传输速率以及由于其为较低噪音的测量值,所以使用数据确认速率在拥塞和不公平竞争检测中产生更好的结果,但是其也可以是更具计算性的和存储密集性的。因此,两个选项都可以用于实施本发明。在没有显著丢失的稳定状况下,由于TCP中固有确认“定时”以及只可以确认已被发送的事实,所以发送速率和确认速率是非常相似的。然而,当传输速率中存在快速变化时,与在慢启动中一样,发送速率可以完全不同于确认速率。在此状况下,与应用发送速率相比,使用确认速率将可能产生更好的慢启动拥塞检测的结果。由于通常使用确认速率产生更好的结果,所以所述的实施例使用确认速率及其对应的飞行尺寸,但是如上所述,简单直接的是,使用改变飞行尺寸估计的发送速率来修改确认速率。

[0115] 根据如图3所述的实施例,在测量飞行尺寸和传输速率(步骤310和320)之后,可以确定飞行尺寸和传输速率这二者的实时发展,也就是,趋势(步骤350和360)。

[0116] 图7示出了确定飞行尺寸的趋势的一种可能实施方式(步骤350)。在附图中,飞行尺寸趋势(flightSizeTrend)表示可以分配以下值的变量:“INDETERMINATE(未确定)”、“UP(向上)”、“DOWN(向下)”和“STABLE(稳定)”,以及在步骤710中最初被设定为“未确定”的默认值。飞行尺寸测量变量存储来自步骤310中的测量值,并且在步骤715中,如果该飞行尺寸测量变量小于可配置值flightSizeMin(飞行尺寸最小值),则离开(leave)飞行尺寸趋势估计程序(其中,飞行尺寸趋势设定为“未确定”)。飞行尺寸最小值可以设定为不利于确定趋势或检测拥塞,比如设为 $20 \times \text{MSS}$ 。MSS代表最大段尺寸,以对应于与在单个TCP段中(也就是,在TCP有效载荷内)传输或接收的数据的最大量(通常以八位字节或字节表示)。

[0117] 在图7和示例性实施例的几个其他附图中使用变量拥塞类型,该变量拥塞类型(congType)表示被检测到的拥塞类型并可以具有以下值:“congSS(慢启动中的拥塞)”(用于检测慢启动中没有不公平竞争的拥塞)、“congNoUnfair(没有不公平竞争的拥塞)”(用于检测拥塞避免中没有不公平竞争的拥塞)、“congUnfair(具有不公平竞争的拥塞)”(用于检测拥塞避免中具有不公平竞争的拥塞)、“afterSS(慢启动之后)”(用于检测从慢启动切换到拥塞避免之后具有不公平竞争的拥塞)以及“afterCwnd(在减少拥塞窗口之后)”(用于检测由于慢启动中的拥塞检测而减少拥塞窗口之后具有不公平竞争的拥塞)。如图3所示的实施例中,假定拥塞类型已被设定为其多个可能值中的一个值,以反映待检测拥塞的类型。应所述注意的是,也可以实施本发明用于上述拥塞类型的其中一个,在此情况下,仅检测所述特殊拥塞类型的存在或不存在。如图4所示的实施例,其中,以协作的方式使用5种拥塞检测的类型,假定拥塞类型在连接于慢启动中开始时,最初被设定为“慢启动中的拥塞”,随后根据网络的当前状态,适当地将其改变为其他值。

[0118] 在图7示出了步骤350的实施例,假定已设定拥塞类型。如果在步骤720中,将其设定为“afterCwnd”或“afterSS”,随后在步骤725中,将SND.UNA值与seqRef变量进行比较,所述seqRef变量在检测到拥塞时存储接下来未发送的连续段的序列号,拥塞窗口因此被减少或者拥塞控制状态因此从慢启动被切换至拥塞避免。如果SND.UNA低于seqRef,则离开飞行尺寸趋势估计程序(其中,飞行尺寸趋势被设定为“未确定”)。如果SND.UNA不低于seqRef,则在步骤735中则检查validRef的值,所述变量表示有效参考值是否已经选择为趋势估计的参考。如果validRe为“FALSE(假)”,以及在步骤740中,拥塞类型被验证为“afterCwnd”,则随后,在步骤750中,将flightSizeRef变量设定为存储在flightSizePrev中的值,也就是,刚好在seqRef指示拥塞检测之前,被设定为flightSizeMeasured值。假定flightSizePrev在当前测量的飞行尺寸值之前,具有测量的飞行尺寸值,并且刚好在更新flightSizeMeasured变量之前,在步骤310中存储所述值。如果在步骤740中拥塞类型不是“afterCwnd”(也就是,如果是“afterSS”),则随后,在步骤745中,将flightSizeRef变量设定为当前flightSizeMeasured值。在步骤745和750这两个步骤之后,离开飞行尺寸趋势估计程序(其中,飞行尺寸趋势被设定为“未确定”)。如果在步骤735中,validRef为“TURE(真)”(也就是,存在趋势确定的有效参考飞行尺寸),但是在步骤760中,SND.UNA低于的seqTest,则也离开飞行尺寸趋势估计程序(其中,飞行尺寸趋势被设定为“未确定”)。所述seqTest为存储序列号的变量,此后在拥塞类型为“afterCwnd”或“afterSS”的情况下,将执

行趋势确定。

[0119] 如果在步骤720中,拥塞类型不是“afterCwnd”或“afterSS”,则在步骤795中,估计validRef,并且如果validRef不是真,则随后在步骤745中,将flightSizeRef设定为当前flightSizeMeasured值并离开飞行尺寸趋势估计程序(其中,flightSizeTrend被设定为“未确定”)。

[0120] 如果在步骤760中,SND.UNA不低于seqTest,或者如果在步骤795中,validRef为真,则随后将执行步骤765,以计算存储阈值的变量flightSizeThresh,其中,在此特别的实施方式中基于所述阈值确定飞行尺寸的趋势并以字节进行测量。可以计算所述阈值为参考飞行尺寸(存储在flightSizeRef中)的固定比率,也由于执行除以8作为二进制记录位移,所以所述比率的1/8为适当的值。随后,所得到的值可以被钳位在最小值flightSizeDiffMin(例如,5*MSS为适当的)和最大值flightSizeDiffMax(例如15*MSS为适当的)之间。

[0121] 在步骤770中,通过更多上述计算的阈值,检查飞行尺寸的当前值是否低于其参考值,并且如果是,则在步骤775中,将变量flightSizeTrend设定为“向下”以及将当前值作为飞行尺寸(存储在flightSizeRef中)和传输速率(存储在rateRef变量中)这二者的新参考值。rateRef变量与flightSizeRef变量类似,但其用作传输速率参考值以在步骤360中计算传输速率的趋势。如果在步骤770中,通过多个上述计算的阈值,确定飞行尺寸不低于其参考值,则随后,在步骤780中,通过多个相同的阈值,检查飞行尺寸的当前值是否高于其参考值,并且如果是,则在步骤780中,将变量flightSizeTrend设定为“向上”。最后,如果在步骤770中,通过多个前面提到的阈值,确定飞行尺寸的当前值不高于其参考值,则随后在步骤790中,将变量flightSizeTrend设定为“稳定的”。在步骤775、785和790中的任一步骤之后,离开飞行尺寸趋势估计程序(其中,flightSizeTrend设定为其确定的值,其指示“向上”意味着增加飞行尺寸趋势或者“向下”意味着降低飞行尺寸趋势或“稳定”意味着稳定,也就是,基本未改变的飞行尺寸趋势)。进一步在步骤370中使用以这样的方式确定的趋势,以检测拥塞。

[0122] 图8示出了确定传输速率的趋势的一个可能实施方式(步骤360)。只要在步骤340中执行了滤波速率梯度时,第一步(步骤330)滤波传输速率是可选的(也就是,步骤330和步骤340两者之一或两者都必须被实施以对测量的传输速率中的变化计数)。使用哪个滤波步骤将取决于哪个度量用于传输速率梯度。传输速率(发送速率和确认速率这两者)可以展示大量变化,除非对计算的梯度执行滤波,否则建议将其滤波以从测量值中去除噪音。广泛使用的和计算性有效的低通滤波器是称作递推预估误差算法(recursive prediction error algorithm)的这类基本平均值估计器,也被称做指数加权移动平均值(EWMA, Exponentially Weighted Moving Average)。将滤波的传输速率更新为新值(RFn)作为先前值的函数,新测量值Rmeasured和不变常量g:

[0123]
$$RFn = (1-g) \cdot RFn-1 + g \cdot Rmeasured = RFn-1 + g \cdot (Rmeasured - RFn-1)$$

[0124] 符号“ $\cdot$ ”表示乘,在附图中其也表示为“ $*$ ”。例如,在每次接收到新ACK获得样本时,1/16是用于g的适当的值,但是对于较低频率的样本,较高的值可能是有益的。尽管如此,本发明不限于所述这些值。然而,可以通过测试,选择引起更稳定拥塞控制机制的值。也可以使用其他滤波算法,如二阶低通滤波器。

[0125] 在图5中示出了传输速率滤波330的主要步骤的示例性实施方式,其中,在第一步骤505中,在更新rateFiltered之前,将rateFiltered的当前值的副本存储ratePrev中。在图5中,rateFilterRestart表示可以具有值“真”或“假”的变量。变量rateMeasured是存储先前在步骤320中所测量的传输速率的变量。变量rateFiltered存储对传输速率测量值滤波所获得的结果。如果在步骤510中,restartRateFilter的值为真,则在步骤520中,通过将rateFiltered设定为rateMeasured以及将restartRateFilter设定为假,重置滤波。否则(如果restartRateFilter为假),则在步骤530中,如下更新变量rateFiltered:

[0126] $\text{rateFiltered} = \text{rateFiltered} + g \cdot (\text{rateMeasured} - \text{rateFiltered})$

[0127] 应所述注意的是,上述滤波的实施方式仅是有利的实例。然而,本发明绝不限于所述特别的实施例。对于本发明的目的,可以应用任何类型的滤波,比如包括任何(加权)平均值滤波或任何低通滤波。

[0128] 再次参照图8,rateTrend表示可以分配以下值的变量:“未确定”、“向上”、“向下”和“稳定”,以及在步骤810中,最初被设定为“未确定”的默认值。在步骤330中,rateFiltered变量存储传输速率的滤波值(或如果未使用滤波,则存在来自步骤320的测量值)。在步骤810之后,如果在步骤815中,将拥塞类型设定为“afterCwnd”或“afterSS”,步骤820则将SND.UNA值与seqRef变量比较。如果SND.UNA低于seqRef,则离开飞行尺寸趋势估计程序(其中,rateTrend被设定为“未确定”)。如果SND.UNA不低于seqRef,步骤825则检查validRef的值。如果validRe为“假”,则随后在步骤830中,将ratePrev变量设定为存储在ratePrev中的值,也就是,恰好在seqRef指示拥塞检测之前,设定为rateFiltered值。如果使用步骤330滤波速率测量值,则假定ratePrev存储之前滤波的速率值,或如果未使用330滤波速率测量值,则存储之前速率测量值(在这种情况下,应恰好在步骤320收集新rateMeasured值之前,更新ratePrev变量)。另外,在步骤830中,一旦已将参考传输速率选择用于趋势估计,将restartRateFilter和restartGradFilter变量设定为真作为重启滤波的方式;将validRef变量设定为“真”,以指示得到传输速率和飞行尺寸参考值这两者,以及由于在拥塞类型为“afterCwnd”或“afterSS”时,SND.NXT为执行趋势计算的序列值,所以SND.NXT将被存储在seqTest变量中。在步骤830之后,离开传输速率趋势估计程序(其中,rateTrend被设定为“未确定”)。如果在步骤825中,validRef为“真”(也就是存在用于趋势确定的有效参考传输速率),但是在步骤835中,SND.UNA低于seqTest,则也离开传输速率趋势估计程序(其中,rateTrend被设定为“未确定”)。

[0129] 如果在步骤815中,拥塞类型不是“afterCwnd”或“afterSS”,则在步骤840中,估计validRef,并且如果varidRef不是真,则随后在步骤845中,将rateRef变量设定为的当前rateFiltered值,同时将validRef变量设定为“真”,以指示得到传输速率和飞行尺寸参考值这两者,并且离开传输速率趋势估计程序(其中,flightSizeTrend被设定为“未确定”)。

[0130] 如果在步骤835中,SND.UNA不低于seqTest,或如果在步骤840中,validRef为真,则随后步骤850将计算传输速率梯度并将其存储在rateGradient变量中。在步骤850、340之后,可以执行滤波速率梯度(将在随后说明步骤340的内部细节)。如前所述,必须实施步骤330和步骤340中的其中一个或这两者,以对测量的传输速率的变化计数。在步骤340(或如果未执行传输速率滤波,则在步骤850)之后,步骤855将计算变量gradientThreshUp和变量gradientThreshDown,其中,这两个变量将基于哪个传输速率的趋势被确定存储阈值。在步

骤850中,可以以不同方式计算传输速率梯度,这将影响步骤340中所述梯度的滤波方式和步骤855中趋势确定的阈值的计算方式。

[0131] 一种计算传输速率的梯度的可选方式是作为当前传输速率(如果使用步骤330滤波传输速率,则为rateFiltered,否则为rateMeasured)和参考值(rateRef)之间的差值,也就是:

[0132] $\text{rateGradient} = \text{rateFiltered} - \text{rateRef}$

[0133] 所述可选方式在一次性趋势计算的情况时尤其有利,如在拥塞类型为“afterCwnd”或“afterSS”情况下、其中,目的是观察传输速率在具体事件之后是否已降低(所述事件可以是慢启动到拥塞避免的转换,或是由于之前拥塞检测而产生的拥塞窗口减少)。

[0134] 在计算梯度作为当前速率和参考速率之间的差值时,可以直接对传输速率(通过步骤330得到)或在步骤340中对rateGradient变量执行滤波,但是,由于即使SND.UNA低于seqTest也将执行滤波,所以在步骤330中滤波时,应获得更好的结果。在这种情况下,可以计算向上趋势确定的阈值gradientThreshUp作为可以阈值飞行尺寸与应用于参考传输速率(存储在rateRef中)的参考飞行尺寸的比率的分数(比如,1/2)。由于将使用阈值来测试传输速率的趋势是否稳定,而且由于阈值过低,会产生更多错误的稳定检测,所以随后可以迫使所得到的值低于参考传输速率的特定比率(比如,1/16)。确定是否存在向下趋势的阈值gradientThreshDown可以有利地被计算为2倍gradientThreshUp,但带有负号。2倍或大于1的一些其他因子,这是为了具有错误地检测向下趋势的较低可能性。总之,可以执行gradientThreshUp和gradientThreshDown计算如下(符号“ $\cdot$ ”表示乘法以及“/”表示除法):

[0135] $\text{gradientThreshUp} = (\text{flightSizeThresh} / \text{flightSizeRef}) / 2 \cdot \text{rateRef}$

[0136] $\text{gradientThreshUp} = \min(\text{gradientThreshUp}, \text{rateRef} / 16)$

[0137] $\text{gradientThreshDown} = -2 \cdot \text{gradientThreshUp}$

[0138] 可选地,可以将梯度阈值与测量的传输速率的变化性关联起来。在这种情况下,需要变化估计,如平均绝对偏差。可以结合步骤330中的滤波来计算传输速率的平均偏差,并使用指数加权平均值(例如,其中因子 $a = 1/16$)将其存储在变量rateMeanDev中。随后可以使用所述变量的倍数(比如,用于向上趋势阈值的2和用于向下趋势阈值的4)来计算传输速率阈值。正如在先前的计算中,最大值强加在阈值中是有益的。总之(“ $| |$ ”表示绝对值):

[0139] $\text{rateMeanDev} = \text{rateMeanDev} + a \cdot (|\text{rateFiltered} - \text{rateMeasured}| - \text{rateMeanDev})$

[0140] $\text{gradientThreshUp} = (2 \cdot \text{rateMeanDev}, \text{rateRef} / 16)$

[0141] $\text{gradientThreshDown} = -2 \cdot \text{gradientThreshUp}$

[0142] 计算梯度的另一可选方式是确定与飞行尺寸每一次改变相关的传输速率的改变。由于会存在具有相同的测量飞行尺寸和不同测量的传输速率的几个连续的新ACK(例如,在拥塞避免中,实际拥塞窗口一次增加每个拥塞窗口字节,这以同样的方式反映在飞行尺寸增加的方式中),所以最初滤波将包括确定用于每个飞行尺寸值的平均传输速率值(例如,这些平均值可以存储在rateFiltered变量中)。随后,速率梯度值可以是平均传输速率的改变除以飞行尺寸从之前不同飞行尺寸值的改变的商,也就是:

[0143] $\text{rateGradient} = (\text{rateFiltered} - \text{ratePrev}) / (\text{flightSizeMeasured} -$

flightSizePrev)

[0144] 在这种情况下,在飞行尺寸具有flightSizePrev值时,ratePrev值将对应于计算的平均传输速率,同时在飞行尺寸为flightSizeMeasured时,rateFiltered对应于计算的平均传输速率。对于飞行尺寸的每次改变,将计算一次所述值。实际上,通过 $(ratePrev/flightSizePrev)$ 与梯度相除来规范化所述梯度。所以,规范化的值1将对应于传输速率增加的状况,作为百分数比率,与飞行尺寸相同,同时值0将指示稳定的趋势以及负值将指示向下趋势。然而,以这种方式计算的梯度将展示相对大的变化,所以对于减少所观察到的变化来说,滤波速率梯度步骤340将是非常有利的。如图6所示实施所述步骤340,从而直接复制图5所示的传输速率滤波实施例。在这种情况下,适当的阈值值是用于gradientThreshUp的0.5,和用于gradientThreshDown的-1。

[0145] 步骤860检查传输速率的(可能滤波的)梯度的当前值是否低于其上述计算的gradientThreshDown阈值,如果是,则在步骤865中,将变量rateTrend设定为“向下”。如果在步骤860中,确定滤波的传输速率不低于其上述计算的gradientThreshDown阈值,则随后在步骤870中检查滤波的传输速率的当前值是否高于其gradientThreshUp阈值,如果是,则在步骤875中,将变量rateTrend设定为“向上”。最后,如果步骤870确定滤波的传输速率的当前值不高于其gradientThreshUp阈值,则随后,在步骤880中,将变量rateTrend设定为“稳定”。在步骤865、875和880中任何一个步骤之后,离开传输速率趋势估计程序(其中,rateTrend被设定为其确定的值:向上、向下或稳定)

[0146] 上面所述的趋势确定实施方式(如图7和图8所示)只是示例。应所述注意的是,本发明也可以采用其他实施方式,其中,稍后在该文档中对其进行详细说明以用于确定传输速率的趋势(使用统计上非参数方法和模型残差)。一般来说,可以通过即将飞行尺寸或传输速率的梯度与阈值比较,以估计飞行尺寸或传输速率的趋势。比如,可以计算梯度,分别作为对飞行尺寸或传输速率的当前值和先前值之间的差值。可以计算滤波的先前值和滤波的当前值之间的差值,以引起变化稳定的结果。然而,也可以计算先前滤波的值和测量的当前值之间的差值,以导致对短暂改变的更快反应。有利地,“先前值”为直接在前的值(滤波的或测量的)。然而,这不意味着限制本发明,先前值也可以是任何的先前值。此外,也可以通过考虑一个以上的测量的或滤波的先前值来执行趋势的计算,从而提供更稳定的趋势估计的优点。

[0147] 图9示出了拥塞检测的可能实施方式(对应于步骤370)。在步骤995中,如果拥塞类型为“afterSS”(在步骤905中),则将MSS字节添加至shadowCwnd变量。如果已经保留在慢启动状态(congType被设置为“afterSS”表示最近已经处于慢启动状态),则shadowCwnd变量存储已达到拥塞窗口的值,以在必要时还原为shadowCwnd变量。如果在步骤910中,将拥塞类型设定为“afterCwnd”,或如果在步骤995之后,将拥塞类型设定为“afterSS”(也就是,在拥塞窗口减少时或在慢启动到拥塞避免切换之后,试图检测具有不公平竞争的拥塞时),则步骤915检查之前确定(在步骤350中)的飞行尺寸的趋势(flightSizeTrend)为“稳定”还是“向上”,如果不是(也就是,是“向下”或“不确定”),则随后在步骤930中,将congestionDetected变量设定为假并终止具有否定拥塞检测(未检测到拥塞)结果的拥塞检测程序。在步骤915中,如果flightSizeTrend为“稳定”或“向上”,则随后在步骤920中,将validRef变量设定为假,从而迫使将随后的趋势确定步骤(步骤350和360)设定新参考值。

在步骤920之后,如果在步骤925中,之前确定(在步骤360中)的传输速率的趋势(rateTrend)为“向下”,则随后在步骤935中将congestionDetected变量设定为“真”并终止具有肯定拥塞检测(检测到网络中的拥塞)结果的拥塞检测程序。否则,如果在步骤925中,rateTrend不为“向下”(也就是,如果为向上、稳定或未确定),则随后在步骤930中,将congestionDetected变量设定为“假”并终止有拥塞检测程序(否定检测)。

[0148] 如果在步骤910中,拥塞类型既不为“afterCwnd”也不为“afterSS”,但在步骤940中,拥塞类型为“congUnfair”(也就是,在拥塞避免范围内试图检测具有不公平竞争的拥塞时),则在步骤945中,检查之前确定(在步骤350中)的飞行尺寸的趋势(flightSizeTrend)是否为“向上”,如果不是(也就是,如果为“稳定”、“向下”或“未确定”),则在步骤960中将congestionDetected变量设定为“假”并终止拥塞检测程序。如果在步骤945中,flightSizeTrend为“向上”,则随后在步骤950中,将validRef变量设定为假,从而迫使随后的趋势确定步骤(步骤350和360)设定新参考值。在步骤950之后,如果在步骤955中,之前确定(在步骤360中)的传输速率的趋势(rateTrend)为“向下”,则随后在步骤965中,将congestionDetected变量设定为“真”并终止拥塞检测程序。否则,如果在步骤955中,rateTrend不是“向下”(也就是,如果为“向上”、“稳定”或“未确定”),则随后在步骤960中,将congestionDetected变量设定为“假”并终止拥塞检测程序(否定检测)。

[0149] 如果在步骤940中,拥塞类型不为“congUnfair”(也就是,在慢启动或拥塞避免范围内试图检测具有不公平竞争的拥塞时),则在步骤970中,将检查之前确定(在步骤350中)的飞行尺寸的趋势(flightSizeTrend)是否为“向上”,如果不是(也就是,如果为“稳定”、“向下”或“未确定”),则随后在步骤990中将congestionDetected变量设定为假并终止拥塞检测程序。如果在步骤970中,flightSizeTrend为“向上”,则随后在步骤975中将validRef变量设定为假,从而迫使随后的趋势确定步骤(步骤350和360)设定新参考值。在步骤975之后,如果在步骤980中,之前确定(在步骤360中)的传输速率的趋势(rateTrend)为“稳定”,则随后在步骤985中将congestionDetected变量设定为“真”并终止拥塞检测程序。否则,如果在步骤980中,rateTrend不为“稳定”(也就是,如果为向上、向下或未确定),则随后在步骤990中,将congestionDetected变量设定为假并终止有拥塞检测程序。

[0150] 图9示出了拥塞状况的检测并考虑所有上面所述的拥塞种类(分别在慢启动或拥塞避免中的拥塞,有或没有竞争的拥塞)。然而,本发明也可以只实施检测和处理这些拥塞字节中的一个或其子集。相应地,可以只实施如图9所示步骤的子集。比如,可以只实施步骤970到990,以检测没有不公平竞争的拥塞。比如,可以只实施步骤945到965,以检测有不公平竞争的拥塞。特别地,在根据本发明实施例的方法中,检测是否存在拥塞的步骤370包括检测是否存在有不公平竞争的拥塞。在所述飞行尺寸的趋势在增加(步骤945)并且所述传输速率的趋势在降低(步骤955)时,肯定会检测到具有不公平竞争的所述拥塞(步骤965),反之,则结果相反。另外或除了所述方法之外,在方法250中,检测是否存在拥塞的步骤370包括检测是否存在没有不公平竞争的拥塞;并且,在所述飞行尺寸的趋势在增加(步骤970)并且所述传输速率的趋势基本上相等(步骤980)时,肯定会检测到没有不公平竞争的所述拥塞,反之,则结果相反。

[0151] 图10示出了拥塞窗口修改的可能实施方式(对应于步骤390)。在图3和图4所示的实施例中,只有在拥塞检测步骤370中,先前检测到一些种类的拥塞时,才执行所述步骤。随

后,步骤390包含由步骤370所执行的检测5种可能拥塞类型的每一个类型的操作并具体表现为拥塞类型变量。如果在步骤1010中,拥塞类型为“CongSS”(也就是,比如,如果在步骤935中,在慢启动中检测到没有不公平竞争的拥塞),则随后,步骤1015将慢启动阈值变量(慢启动阈值)设定为拥塞窗口(cwnd)的当前值,因此将拥塞控制状态切换至拥塞避免。在步骤1015之后,步骤1020也将shadowCnd变量设定为拥塞窗口,从而,如果慢启动中保留有拥塞,则所述变量可以被用以监测可以检测到何种拥塞窗口(在所述实施例中,在随后的步骤370中,更新shadowCwnd)。然而,只有在检测到慢启动中的没有不公平竞争的拥塞之后,步骤1015才是必要的,将拥塞类型设定为“afterSS”用以检测恰好在慢启动切换至拥塞避免之后是否存在有不公平竞争的拥塞。因此,在如图3所示实施例的一般情况下,步骤1015将不是必需的,但是对于如图4所示的更具体的实施例来说,所述步骤是必需的。

[0152] 如果在步骤1025中,拥塞类型为“congNoUnfair”(也就是,如果在拥塞避免中,已检测到没有不公平竞争的拥塞),则随后,步骤1030以deltaCwnd的量减少拥塞窗口。字节的数量(deltaCwnd)可以为拥塞窗口固定的比例(比如,20%),或为用以确定飞行尺寸的趋势(flightSizeThresh)的飞行尺寸阈值的比例。在后一种情况下,建议其为稍微更大的数量(比如,1.2x flightSizeThresh),所以,如果已检测到拥塞,则将拥塞窗口减少为大于真实拥塞触发,因此,使其在连续几轮拥塞窗口增长的拥塞触发点周围和由于拥塞检测而产生的减少周围上下浮动。

[0153] 步骤1035在步骤1020或1030之后,但其只有在检测到没有不公平竞争的拥塞之后才是必要的,如果将拥塞类型设定为“afterSS”或“afterCwnd”,以分别检测在慢启动到拥塞避免的切换之后或拥塞窗口减少之后,是否存在具有不公平竞争的拥塞。因此,在如图3所示的实施例中,步骤1035将不是必需,但是在4所示的实施例中实施所述步骤。步骤1035包括将变量seqRef设定为是序列号的SND.NXT,在所述序列号处,将选择飞行尺寸和传输速率的参考值,以确定其趋势(实际上,如图7和图8所示,除了在拥塞类型为afterCwnd,所述值在飞行尺寸的seqRef之后马上得到之外,所述值在seqRef之前得到)。

[0154] 如果在步骤1040中,拥塞类型为“congUnfair”(也就是,如果已在拥塞避免中检测到有不公平竞争的拥塞),则随后步骤1045将慢启动阈值变量(慢启动阈值)设定为更大的值,比如,3倍拥塞窗口的当前值(cwnd),因此将拥塞控制状态切换至慢启动。

[0155] 如果在步骤1050中,拥塞类型为“afterSS”(也就是,如果在慢启动切换至拥塞避免之后,已检测到有不公平竞争的拥塞),则随后,步骤1055将拥塞窗口设定为shadowCwnd变量的值(如果慢启动中停留有连接,则其反映的是拥塞窗口将获得的值)。步骤1055也将慢启动阈值变量(慢启动阈值)设定为更大的值,比如3倍拥塞窗口的值(Cwnd),因此将拥塞控制状态切换至慢启动。

[0156] 如果在步骤1060中,拥塞类型为“afterCwnd”(也就是,如果在由于拥塞检测产生的拥塞窗口的减少之后,已检测到有不公平竞争的拥塞),则随后,步骤1065将先前存储在deltaCwnd变量中的值添加至拥塞窗口,从而反映了在先前时间点检测到拥塞时拥塞窗口的减少。这样的方式下,拥塞窗口的减少为“undone”。

[0157] 对于如图3所示的实施例来说,使用如图5到图10所示的步骤330、340、350、360、370和390,初始化一些使用的变量是有益的,也就是说,无论在启动连接还是在存在TRO超时或不活动超时时,都应所述将restartGradFilter、restartRateFilter和validRef变量

初始化为“假”。另外,在连接启动时,都应拥塞类型设定所需的拥塞检测类型。

[0158] 图10示出了对所有上面所述的拥塞状况的处理,特别示出了拥塞窗口的修改390。然而,可以只针对单个状况或其子集来实施本发明。一般来说,修改拥塞窗口的步骤390包括在肯定检测到有不公平竞争的拥塞(步骤1040)时,增加慢启动阈值(步骤1045),这样将会将连接切换至慢启动。图10示例性示出了慢启动阈值增加为当前拥塞窗口的3倍。然而,这仅是示例并且可以通过不同的倍数或其他方式来执行增加。在TCP的情况下,如果传输节点处于TCP拥塞避免拥塞控制状态,则有利地,在肯定检测到有不公平竞争的拥塞时执行增加拥塞窗口尺寸的操作,否则不执行此操作。

[0159] 可选地或另外地,修改拥塞窗口的步骤390可以包括在肯定检测到没有不公平竞争的拥塞(步骤1025)时,减少拥塞窗口尺寸(步骤1030)。如果传输节点处于TCP拥塞避免拥塞控制状态,在使用TCP时,则在肯定检测到没有不公平竞争的拥塞时执行减少拥塞窗口尺寸的操作(步骤1010中的“NO”表示不在慢启动中),否则,不执行此操作。

[0160] 仍然可选地或除了先前所述的拥塞窗口处理方法之外,修改拥塞窗口的步骤390可以包括在肯定检测到有不公平竞争的所述拥塞时,将当前拥塞控制状态改变为新的拥塞控制状态,其中,当前拥塞控制状态和新的拥塞控制状态分别指定当前和新的规则,以用于在拥塞和数据丢失都未被检测到时,接收到数据确认时增加所述拥塞窗口尺寸,以及新的规则指定比当前规则更快速率增加所述拥塞窗口尺寸。在TCP的情况下,如果当前拥塞控制状态为TCP拥塞避免,则在肯定检测到有不公平竞争的所述拥塞时,执行从当前拥塞控制状态向新的拥塞控制状态改变的操作,否则,不执行此操作,以及新拥塞控制状态对应于TCP慢启动。在TCP的情况下,如果当前拥塞控制状态为TCP拥塞避免,则在肯定检测到有不公平竞争的所述拥塞时,有利地执行从当前拥塞控制状态向新的拥塞控制状态改变的操作,否则,不执行此操作,以及新的拥塞控制状态为TCP慢启动。

[0161] 仍然可选地或除了之前所述的实施方式之外,修改拥塞窗口的步骤390可以包括在肯定检测到有不公平竞争的拥塞时,将当前拥塞控制状态改变为新的拥塞控制状态,其中,当前拥塞控制状态和新的拥塞控制状态分别指定当前和新的规则,其以用于在拥塞和数据丢失都未被检测到时,接收数据确认时增加拥塞窗口尺寸,以及新的规则指定比当前规则更慢的速率增加所述拥塞窗口尺寸。在TCP的情况下,如果当前拥塞控制状态为TCP慢启动,则在肯定检测到有不公平竞争的所述拥塞时,执行从当前拥塞控制状态向新的拥塞控制状态改变的操作是有利的,否则,不执行此操作,以及新的拥塞控制状态为TCP拥塞避免状态。

[0162] 图4示出了本发明的方法的一个可能实施例,其包括协作方式下的5种类型的拥塞检测步骤(和其各自有关的操作)。图4示出了在计算(230)RTT之后,假定执行拥塞和竞争检测步骤250的实施方式作为新ACK的处理的一部分。除了图4所包括的步骤之外,还存在为本发明的一部分的一个以上的步骤,其在如图4所述的步骤之后执行(参见图2):上面所述在考虑拥塞窗口尺寸的同时传输数据(270)。

[0163] 如图4所示的本发明的实施例使用如图3所述的实施例中的所有步骤(310、320、350、360、370、380和390)并添加了更多的步骤,以管理指定使用的拥塞检测类型的拥塞类型变量。在这样的方式下,所述实施例将在慢启动中和检测到拥塞时,检测没有不公平竞争的拥塞,将连接切换至拥塞避免。在切换至拥塞避免时,随后参考切换发生的时间,确定是

否存在有不公平竞争的拥塞,并且如果是存在,则将连接切换回慢启动。一旦在拥塞避免中,将通过减少拥塞窗口对待对没有不公平竞争的拥塞的检查。由于拥塞产生的拥塞窗口的减少将触发特殊的检查,以参考拥塞窗口减少的时间来验证不存在具有不公平竞争的拥塞,如果发现存在,则将迫使不减少拥塞窗口。如果在拥塞避免中,未检测到没有不公平竞争的拥塞,则也将检查具有不公平竞争的拥塞,此时将连接切换至慢启动。

[0164] 下面将详细说明图4所示的实施例的步骤。如果如图2所述的部分实施例,在慢启动或拥塞避免中,将调用如图4所示的步骤用于每个新ACK消息。如果在步骤405中,连接处于慢启动中,则在步骤410中,拥塞类型设定为“conSS”,以确定在慢启动中检测到没有不公平竞争的拥塞。如果未检测到,则将保留拥塞类型之前的值。在两种情况下,都将执行下面的两个步骤以测量飞行尺寸310和测量传输速率,可以如上所述实施这两个步骤(在说明如图3所示实施例的这些相同步骤时)。接下来的步骤将是确定飞行尺寸的趋势350(通过如图7所示的一个可能实施例)、测量传输速率趋势360(通过如图8所示的一个可能实施例)和检测拥塞370(通过如图9所示的一个可能实施例),作为如图3所示实施例的说明的部分对所有这些步骤进行详细说明。

[0165] 随后,如果在步骤380中,拥塞检测变量为真(也就是,在步骤370中,检测到一些种类的拥塞),则执行步骤390(通过如图9所示的一个可能实施例),以适当地修改拥塞窗口。随后如果在步骤450中,拥塞类型为“afterCwnd”(拥塞窗口减少之后的有不公平竞争的拥塞),则步骤455将拥塞类型设定为“没有不公平竞争的拥塞”(拥塞避免中没有不公平竞争的拥塞检测)。然而,如果拥塞类型为“conSS”(慢启动中没有不公平竞争的拥塞检测),则从步骤460跳至步骤465,其中将拥塞类型设定为“afterSS”(切换至慢启动之后有不公平竞争的拥塞检测)。最后,如果拥塞类型为“conNoUnfair”(拥塞避免中没有不公平竞争的拥塞检测),则从步骤470跳至步骤475,其中将拥塞类型设定为“afterCwnd”(拥塞窗口减少之后具有不公平竞争的拥塞检测)。

[0166] 另一方面,如图4所示,如果在步骤380中,拥塞检测为假,也就是未检测到任何拥塞,则步骤415则在拥塞类型为“afterSS”或“afterCwnd”的情况下,将验证SND.UNA是否大于seqTest,并且如果是,则在420中,将拥塞类型设定为“conNoUnfair”。这样就确保了仅执行关于具体具体事件(到拥塞避免的转换和由于拥塞产生的拥塞窗口的减少)的有不公平竞争的拥塞检测,直到seqTest所指定的序列号,并且在此之后,将拥塞检测模式切换至拥塞避免中没有不公平竞争的拥塞检测。实际上,如果步骤415中的验证不是肯定的而且如果425确定拥塞类型为“conNoUnfair”(也就是,在拥塞避免中,已测试但未发现没有不公平竞争的拥塞),则随后,步骤430将拥塞类型设定为“conUnfair”(拥塞避免中有不公平竞争的拥塞),并且随后运行另一个拥塞检测步骤370(通过如图9所示的一个可能实施例),其中利用相同的趋势信息但是在此情况下检查不同类型的拥塞。如果步骤435验证所述新的拥塞检测已是否定,则随后,步骤440将拥塞类型设定回“conNoUnfair”,使得对于接下来新ACK来说,将再次测试没有不公平竞争的拥塞,并且如果未发现,则测试有不公平竞争的拥塞。如果步骤435发现检测到有不公平竞争的拥塞,则随后步骤390(通过如图10所示的一个可能实施例)将适当地修改拥塞窗口,最后步骤445将拥塞类型设定回“conNoUnfair”,如果,与如图10所述的步骤390的实施例一样,在一些情况下,将连接切换至慢启动,则在步骤405检测到拥塞在慢启动中时,拥塞类型将不改变为“conSS”,直到接下来的新ACK。

[0167] 对于如图4所示的实施例来说,使用如图7到图10所示的步骤350、360、370和390,初始化一些使用的变量是有益的,也就是说,无论在启动连接还是在存在TRO超时或不活动超时,都应在将restartGradFilter、restartRateFilter和validRef变量初始化为“假”的同时,将拥塞类型初始化为“congSS”。

[0168] 使用统计上非常量方法确定传输速率趋势

[0169] 由于滤波的传输速率可以展现大量变化性并且所述变化性可以不遵循明确的统计分布,所以使用统计上非常量(non-parametric)方法以确定传输速率的趋势可以是有益的。在这样的情况下,与其有不同的步骤360的可能实施例可以仍然基于图8,其中,关于上述实施方式具有大量差异。主要的差异将是可以使用称作rateSample的新变量来代替rateFiltered变量,rateSample为rateFiltered的N个(比如,5个)样本的矢量或阵列。例如,在步骤330之后和步骤810之前,一个可能起始步骤将值分配给N分量rateSample。在这样的方式下,rateSample将具有rateFiltered的最后N个值或最后值的子集。在任何情况下,假定图8中rateFiltered的每个例子现在都是rateSample的例子,另外需要的改变是ratePrev和rateRef变量也将不得不变为N-分量矢量或阵列变量。在步骤830中,在将rateFiltered的值分配给rateRef时,并且在步骤845中,在将ratePrev的值分配给rateRef时,这些分配将是矢量/阵列分配,也就是,将分配矢量或阵列的所有分量。另一个需要的改变是将不使用步骤850(计算速率梯度)和步骤340(滤波速率梯度),而是直接计算仍将是标量变量(也就是,不会变为矢量/阵列)的gradientThreshUp和GradientThreshDown,所述gradientThreshUp和GradientThreshDown被计算为阈值飞行尺寸和参考飞行尺寸的比率的分数(也就是,flightSizeThresh/flightSizeRef)并被应用于平均参考传输速率(来源于存储在rateRef中的样本)。这与步骤855中一个可能实施例所述的类似,但是其使用rateRef的样本平均值,也就是:

[0170] $\text{gradientThreshUp} = ((\text{flightSizeThresh}/\text{flightSizeRef})/2) \cdot \text{mean}(\text{rateRef})$

[0171] $\text{gradientThreshUp} = \min(\text{gradientThreshUp}, \text{mean}(\text{rateRef})/16)$

[0172] $\text{gradientThreshDown} = -2 \cdot \text{gradientThreshUp}$

[0173] 最重要的改变在步骤860和870中。如果通过检查rateSample矢量是否大于从gradientThreshUp到rateRef的每个N分量的总和,则步骤860将检查是否肯定检查到传输速率的梯度。如果rateSample矢量小于从rateRef的每个N分量中减去gradientThreshUp所得到的矢量,则步骤870将检查是否否定检查到传输速率的梯度。这留下了两个统计上非常量方法使用的矢量不等式测试(">"和"<"),特别地,可以使用众所周知的Mann-Whitney U test(也被称作Mann-Whitney-Wilcoxon、Wilcoxon rank-sum test或Wilcoxon-Mann-Whitney test),以确定N值的两个样本(每个由N-分量矢量/阵列表示)是否由N-分量是更大(来自于较大平均值的群集)、更小(来自于较小平均值的群集)或等于(来自于相同群集表示)。

[0174] 通过将层级分配给两个比较的矢量的N个值中的每个可以轻易执行所述测试,也就是,在序列中对 $2 \cdot N$ 个值进行排序并将序列的序号分配其值(和分配给相同序号)。随后,对于其N个分量中的每个,使用低平均值的矢量/阵列以对所观察到的具有较小层级的其他矢量/阵列(对于等于所述层级的任何矢量/阵列,计委半数)计数。这些计数的总和为U统计量。根据90%可靠区间的一侧测试的标准表格适当的样本计数值 $N=5$,在这种情况下,U值

为5或更小意味着样本(矢量/阵列)小于其他样本,同时值为20或更大意味着样本(矢量/阵列)高于其他样本。

[0175] 使用模型残差确定传输速率趋势

[0176] 确定传输速率的趋势的另一可选方式是使用统计上的测试检测传输速率测量值中的趋势,随后使用测试产生的度量作为梯度,其中,随后度量被滤波(例如,通过随着时间的推移累积或取平均值)作为梯度。一种可能的所述测试用于检测向上趋势,其将两个模型拟合测量的数据,一个模型为没有拥塞的期望动作,另一个也是没有拥塞的期望动作。然后比较两个模型良好地拟合测量数据的度量可以作为从传输速率的趋势推导的梯度。例如,可以将每个传输速率测量值与预测传输速率测量值与先前值相同的模型比较(也就是,拥塞状况预期的随意游走模型),随后与预测传输速率从最后值开始以与测量的飞行尺寸增加(可以在非拥塞状况中预期)相同的比例增加的模型比较。在每一种情况下,这样将产生残差,从而取所述残差的平方,随后互相相除,以产生指示哪种模型更好地拟合测量的数据的度量。

[0177] 图11示出了一种可能实施方式,除了现在将要说明的一些方面之外,其与图8等同。在之前段落所述的梯度只可以被用以检测传输速率向上的趋势,所以需要另一个梯度以测量趋势是否向下。因此,在如图11所示的一个可能实施例中,由步骤851产生的梯度将是双值变量:其将产生向下趋势测试的梯度(存储在称作rateGradientDown的变量中)和向上趋势测试的梯度(存储在称作rateGradientUp的变量中)。在步骤895中,将分别对这些梯度中的每一个滤波,随后步骤856将产生向上趋势测试的一个阈值(存储在称作gradientThreshUp的变量中)和向下趋势测试的另一个阈值(存储在称作gradientThreshDown的变量中)。

[0178] 对于向上趋势测试来说,可以使用所述的比较两个模型的拟合的度量,例如,平方残差的商数:

[0179]
$$\text{rateGradientUp} = (\text{rateMeasured} - \text{ratePrev})^2 / [\text{rateMeasured} - \text{ratePrev} \cdot (1 + (\text{flightSizeMeasured} - \text{flightSizePrev}) / \text{flightSizePrev})]^2$$

[0180] 对于向下趋势测试来说,可能使用之前所述的步骤360的示例性实施例中已使用的梯度:

[0181]
$$\text{rateGradientDown} = \text{rateFiltered} - \text{rateRef}$$

[0182] 随后,在步骤895中,滤波rateGradientUp和rateGradientDown可以使用两个梯度的指数加权移动平均值(如上所述)。另一种滤波方法可能用于rateGradientUp,与单独累加两个模型的每个平方残差的总和一样,随后通过划分平方残差的累加和来更新滤波的梯度。

[0183] 在步骤895之后,步骤856可以计算两个阈值:

[0184]
$$\text{gradientThreshDown} = ((\text{flightSizeThresh} / \text{flightSizeRef}) / 2) \cdot \text{rateRef}$$

[0185]
$$\text{gradientThreshDown} = -2 \cdot \min(\text{gradientThreshDown}, \text{rateRef} / 16)$$

[0186]
$$\text{gradientThreshUp} = 2$$

[0187] 已之前步骤360中的示例性实施例使用用于向下趋势计算的阈值,同时在预测拥塞的模型中的平方残差(误差)大于(2倍大)没有拥塞的模型中的平方残差(误差)时,将达到gradientThreshUp所使用的阈值2。

[0188] 除了存在向上趋势确定的具体梯度和向下趋势确定的其他梯度之外,在步骤856

之后,步骤861和871将以与先前在步骤360中的其他实施例所述的几乎相同的方式,比较阈值并确定趋势。

[0189] 本发明的实施例包括被配置为执行上面所述的任何方法的装置。在图12中示出所述装置的示例。

[0190] 比如,提供一种装置,以用于在传输节点(如发送主机110、111或代理130)中控制采用确认通信的数据通信协议中拥塞,在所述确认通信中传输节点将数据传输至接收节点而且所述接收节点确认接收到所述数据,其中,拥塞窗口指定未确认数据的最大量,可以所述传输节点在接收到所有或部分所述数据的肯定确认之前进行传输所述未确认数据。所述装置包括用于测量飞行尺寸的飞行尺寸测量单元1210,所述飞行尺寸指示由所述传输节点发送但尚未被所述接收节点确认的数据量;用于测量传输速率的传输速率测量单元1220,所述传输速率对应于与所述飞行尺寸相同的时刻;飞行尺寸趋势计算单元1250,其用于确定所述飞行尺寸的趋势;传输速率趋势计算单元1260,其用于确定所述传输速率的趋势;拥塞检测单元1270,其用于根据确定的所述传输速率的趋势和所述飞行尺寸的趋势来检测是否存在拥塞;拥塞处理单元1280,其在所述拥塞被所述拥塞检测单元肯定检测到时,用于修改所述拥塞窗口;以及传输单元1290,其用于在考虑所述拥塞窗口尺寸的同时传输数据。

[0191] 特别地,在所述装置实施TCP时,所述装置有利地包括逻辑性(也可能是物理性,也就是,通过单个处理设备实施)包含上述单元的TCP逻辑处理单元1200以及拥塞和竞争检测单元1295。

[0192] TCP逻辑处理单元1200包括需要使用TCP协议发送和接收数据的逻辑。根据TCP标准,如IETF RFC 793和其他上面引用的标准,在这两种情况下,TCP逻辑处理单元中的接收单元1205包括接收TCP段的TCP功能,同时传输单元1290包括发送TCP段的TCP功能。特别地,传输单元也包括TCP拥塞窗口和TCP发送窗口的强制执行,从而只有在参数允许时,传输数据包。

[0193] 除了上面所述的传送单元,拥塞和竞争检测单元1295包括本发明中的其他元件:飞行尺寸测量单元1210、传输速率测量单元1220、飞行尺寸趋势计算单元1250、传输速率趋势计算单元1260、拥塞检测单元1270和拥塞处理单元1280。TCP逻辑处理单元1200对于由接收单元接收的每个新ACK(之前确定的)调用拥塞以及拥塞和竞争检测单元的功能,以用于慢启动或拥塞避免状态中的连接,所述新ACK可能包括新ACK接收到数据段时的SND.UNA变量的值。拥塞以及拥塞和竞争检测单元具有SND.NXT的当前值、最后测量的RTT、拥塞窗口以及TCP逻辑处理单元中的TCP连接的ssThresh变量的入口(例如,通过内存),从而包括改变存储的拥塞窗口和ssThresh的值的的可能性。另外,拥塞和竞争检测单元的内部单元可以通过存储在内存中的变量互相通信。

[0194] 如图13所示,可以实施TCP逻辑处理单元1200和拥塞以及拥塞和竞争检测单元1295,以作为操作系统1310的核心的一部分。操作系统核心包括网络接口驱动和软件库,所述驱动和库可以通过数据网络将数据包发送至其他系统并从其他系统接收数据包。在所述实施方式中,TCP处理逻辑单元1200可以是大部分操作系统(比如,Linux、Windows、BSD、OSX、Android、IOS)核心的TCP功能,所述操作系统核心适于如上所述调动拥塞和拥塞检测单元1295以用于新ACK,使得访问也如上所述的SND.UNA、SND.NXT、最后测量的RTT、拥塞窗口和ssThresh。在这种情况下,除了在发送出段时存储SND.UNA的值之外,处理逻辑单元

1200中的传输单元1290将不需要作任何调整。

[0195] 可选地,如图14所示,在“user-space”中,可以实施TCP逻辑处理单元1200和拥塞以及拥塞和竞争检测单元1295,以不作为操作系统1410的部分核心。在实施方式中,操作系统核心包括网络接口驱动和软件库,其可以通过数据网络将数据包发送至其他系统并从其他系统接收数据包。在这样的情况下,TCP处理逻辑单元1200在“user-space”(核心之外)中实施TCP协议,其包括根据TCP协议发送出数据段的传输单元1290,也就是,如之前所述的考虑连接的拥塞窗口和发送窗口,通过系统库1420系统库与操作系统核心的接口连接。在这种情况下,TCP处理逻辑单元1200在发送出段时,存储SND.UNA的值。

[0196] 可以通过配置上面所述的标准的执行以实现包括在拥塞以及拥塞和竞争检测单元1295中的单元的实施方式,作为本发明的方法的示例性实施例一部分。特别地,飞行尺寸测量单元1210被配置为执行步骤310,测量飞行尺寸;传输速率测量单元1220被配置为执行步骤320,测量传输速率;飞行尺寸趋势计算单元1250被配置为执行步骤350,确定飞行尺寸的趋势;传输速率趋势计算单元1260被配置为执行步骤360,确定传输速率的趋势;拥塞检测单元1270被配置为执行步骤370,检测拥塞;以及拥塞处理单元1280被配置为执行步骤390,只有在拥塞检测单元检测到拥塞时,修改拥塞窗口。

[0197] 此外,如图15所示,可以在对应于处理设备的装置中采用本发明,该设备包括一个或多个处理器1510、内存1520或内存集合以及所有互相连接的具有内存存储指令的一个或多个网络接口1530,在由处理执行所述指令时,指令引起处理器执行包括上面所述方法的任何步骤的操作。特别地,处理器可以被配置为接受新ACK消息(本发明的方法的部分实施例所述和如图2所示的步骤220);新ACK消息的处理可能包括SND.UNA参数更新(本发明的方法的部分实施例所述和如图2所示的步骤220);测量RTT更新(本发明的方法的部分实施例所述和如图2所示的步骤230);如果接收的新ACK在慢启动或拥塞避免中,则随后执行拥塞和竞争检测更新(本发明的方法的部分实施例所述和如图2所示的步骤250);TCP更新所需的新ACK消息的另一个处理(本发明的方法的部分实施例所述和如图2所示的步骤260);以及在考虑拥塞窗口尺寸更新的同时,传输数据(本发明的方法的部分实施例所述和如图2所示的步骤270)。图15所示的装置可以将执行上面所述任何方法的指令存储在内存1520或多个所述内存中。随后,所述指令可以由处理器或多个所述处理器执行,可以在通信的基础上执行拥塞控制并且如上面所述对其进行估计(测量、滤波、处理以获得趋势、测试情形),所述通信如通过网络接口1530传输或接收确认和数据。

[0198] 可以在如图3所述实施例的详细说明的具体实施拥塞和竞争检测。例如,其可以如图4所述的实施例的详细说明的进行具体实施。

[0199] 例如,装置实施方式的处理器可以是通用目的处理器、微型控制器、用于手机的优化处理器或平板设备、定制的ASIC设备、FPGA设备或其他类似的设备。内存可以是RAM、ROM、EPROM、EEPROM、硬盘、固态硬盘或类似的存储设备或任何它们的组合。网络接口可以使得通过有线端口(比如,以太网、电缆、光纤、ISDN、ADSL)或无线端口(比如,Wifi、Wimax、GPRS、UMTS、HSPA、LTE、微波链路),以向网络发送数据和从其接收数据。例如,采用本发明的装置可以通信的网络是LAN、WAN、卫星、3G无线、4G无线、电缆、回程线路等。

[0200] 上面示例的装置实施TCP通信。然而,但是本发明不限于此,也可以通过使用确认通信的任何其他通信协议使用本发明。

[0201] 此外,可以将如上所述的装置应用至端点,如主机(如图1A所示),也可以应用至代理节点(如图1B所示)。在后者的情况下,例如,在TCP代理下,采用本发明的装置将拥塞和完成检测和控制应用至发送功能。

[0202] 此外,可以作为存储程序的数据载体实施本发明,所述程序包括能够在运行电脑时,使电脑执行任何上面所述方法的步骤的指令。数据载体可以是可携式数据载体,如可通过USB接口连接至电脑的闪存、如CD、DVD、蓝光等的光学介质。数据载体也可以是硬件驱动或其他任何存储媒介。

[0203] 总之,本发明涉及用于网络的确认通信的拥塞控制。基于飞行尺寸和传输速率二者的趋势检测拥塞,以根据检测结果调整拥塞窗口。比如,所述拥塞检测能够区分有不公平竞争或没有不公平竞争的拥塞。另外,可以对测量的传输速率滤波以补偿时间变化。本发明可以在端节点或代理内实现。

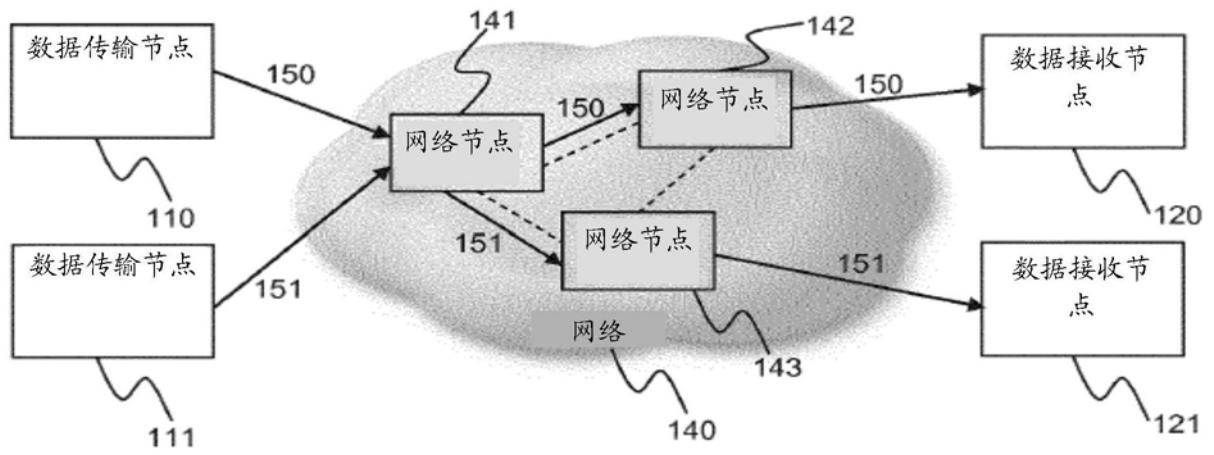


图1A

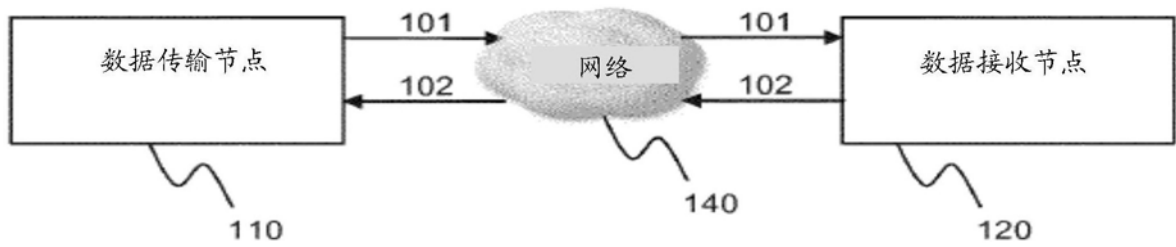


图1B

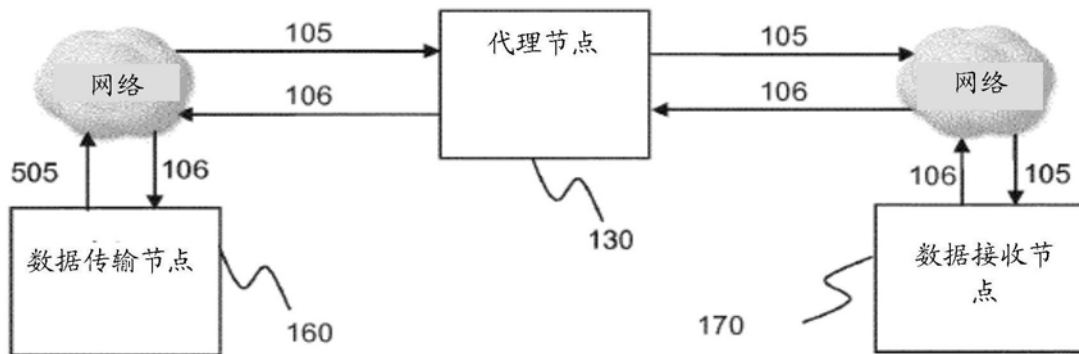


图1C

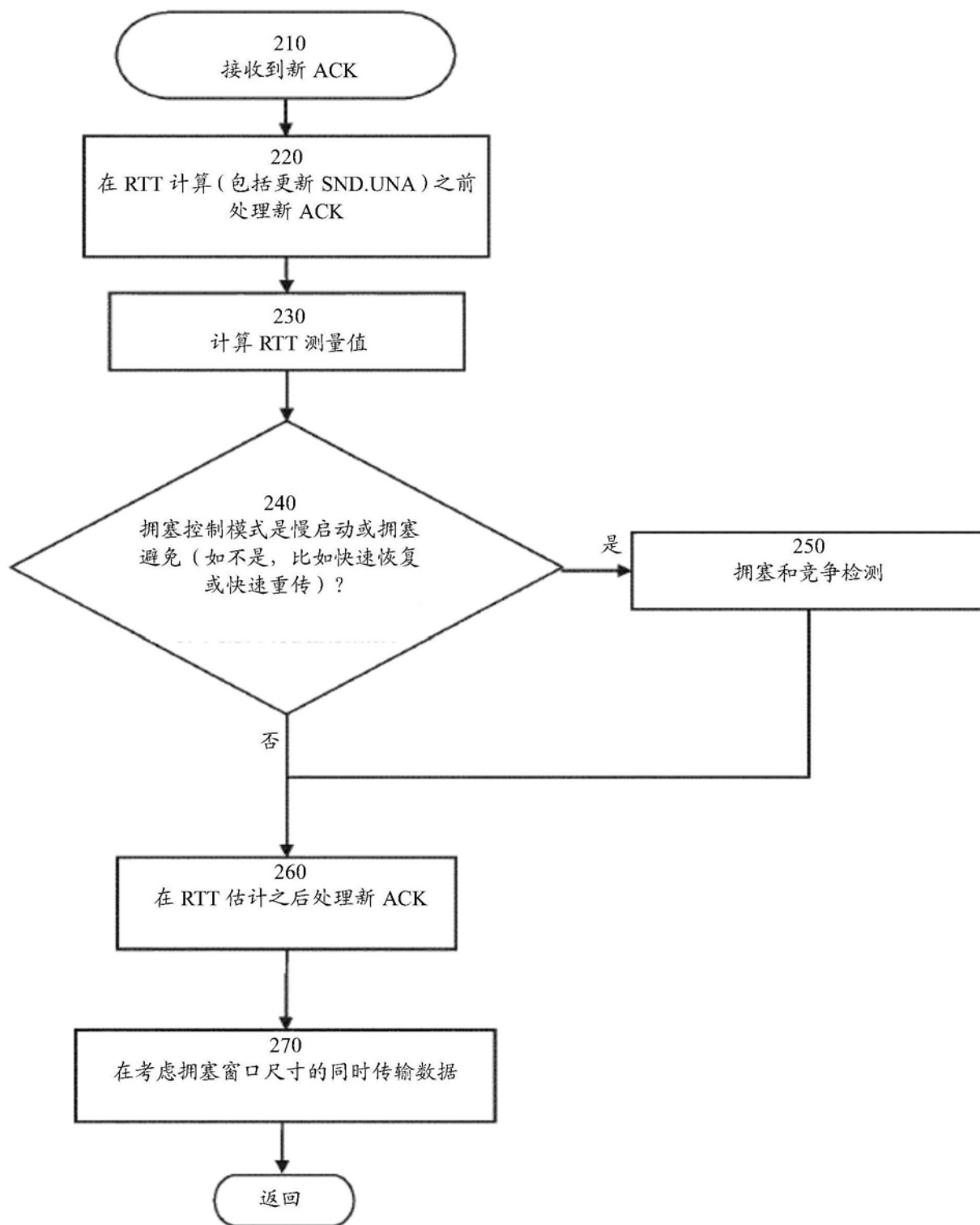


图2

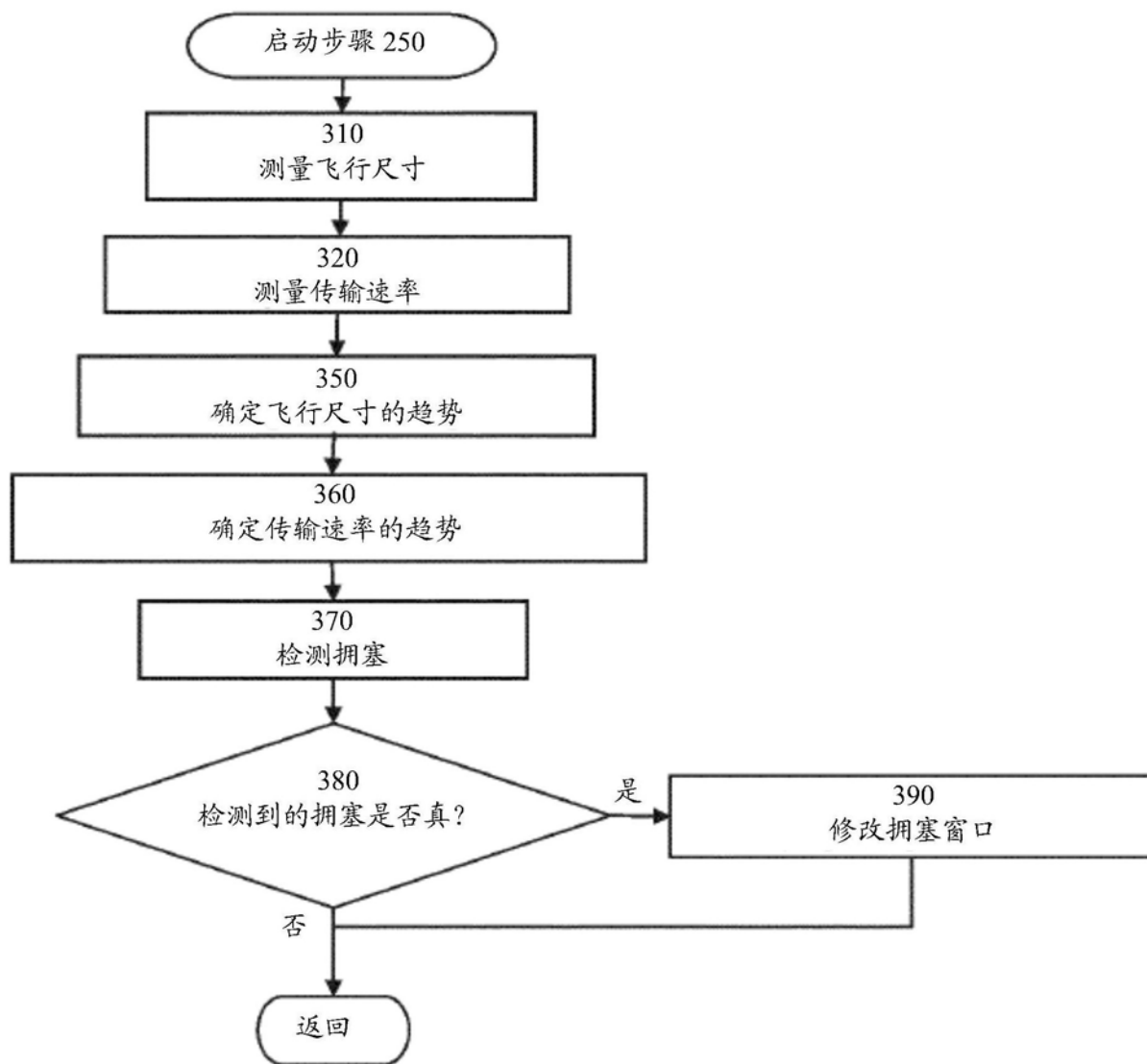


图3

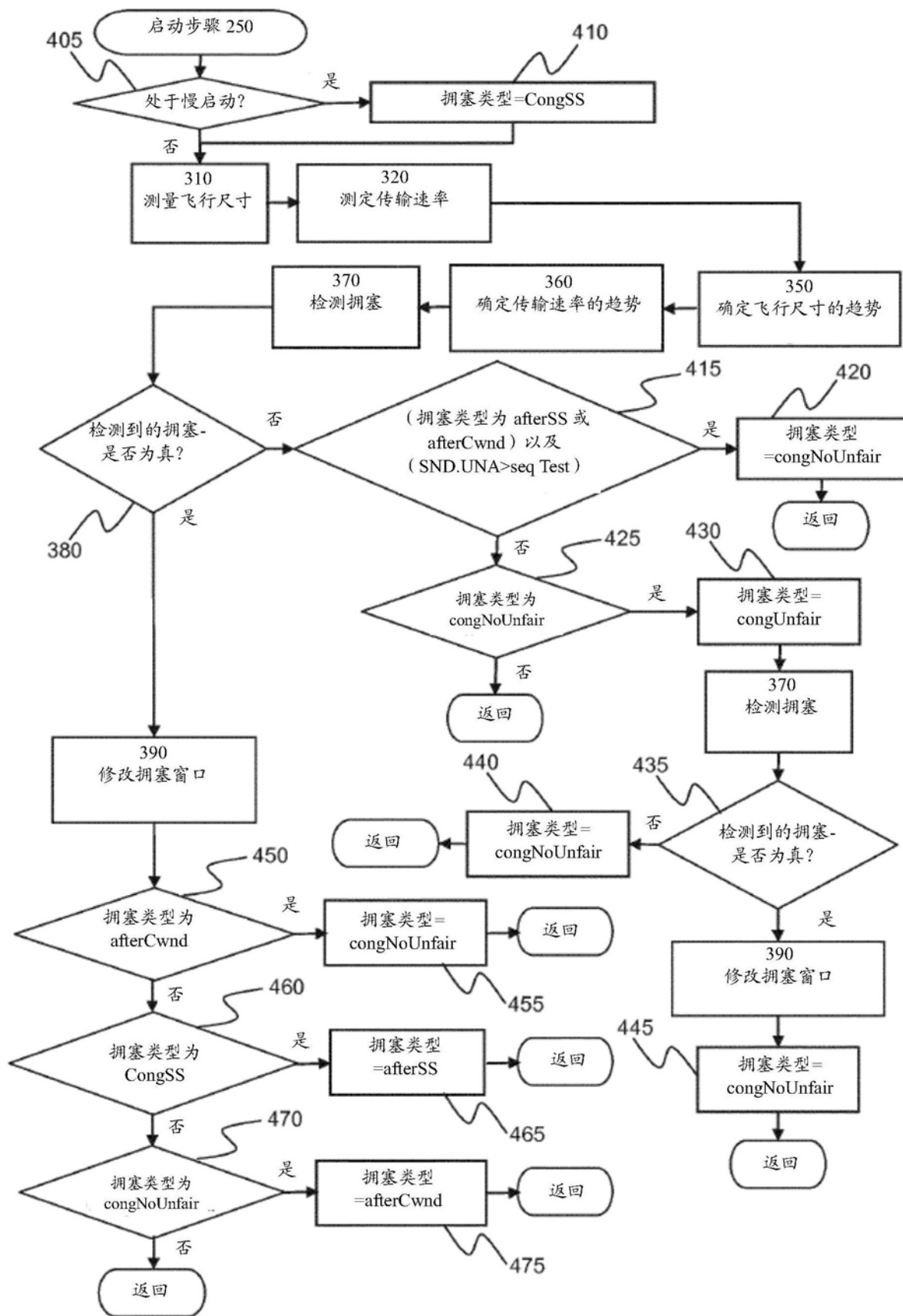


图4

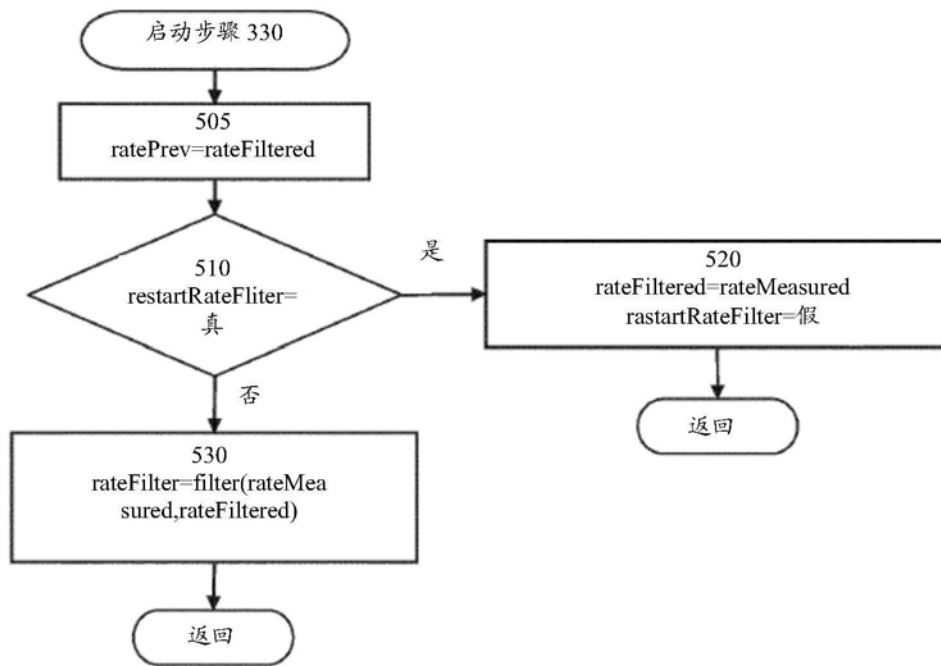


图5

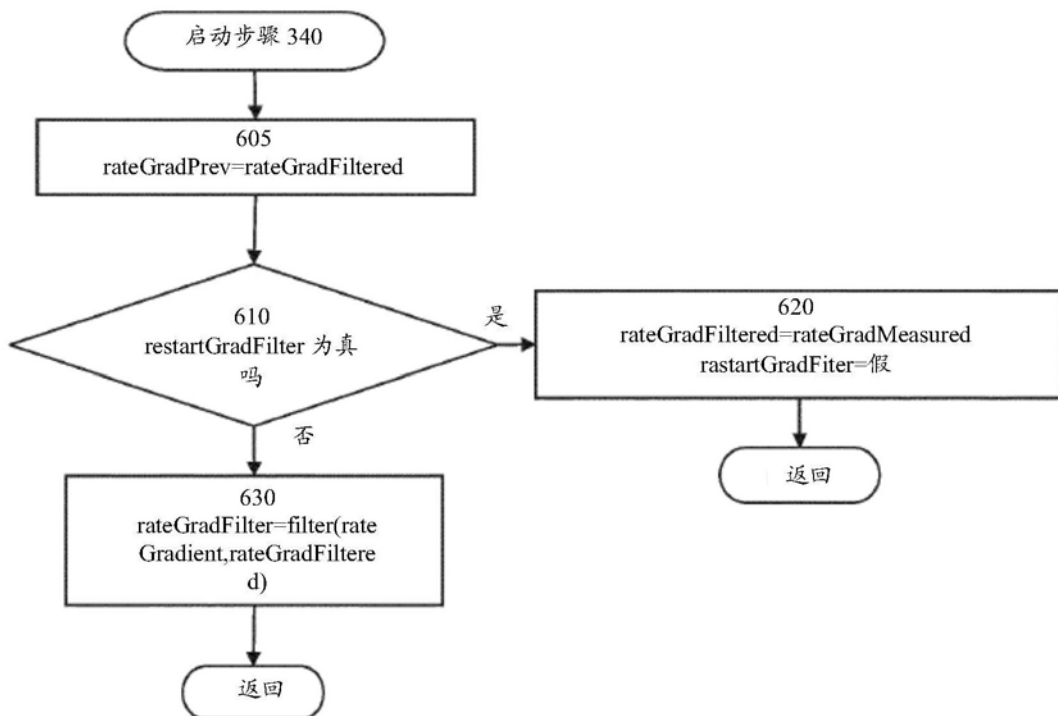


图6

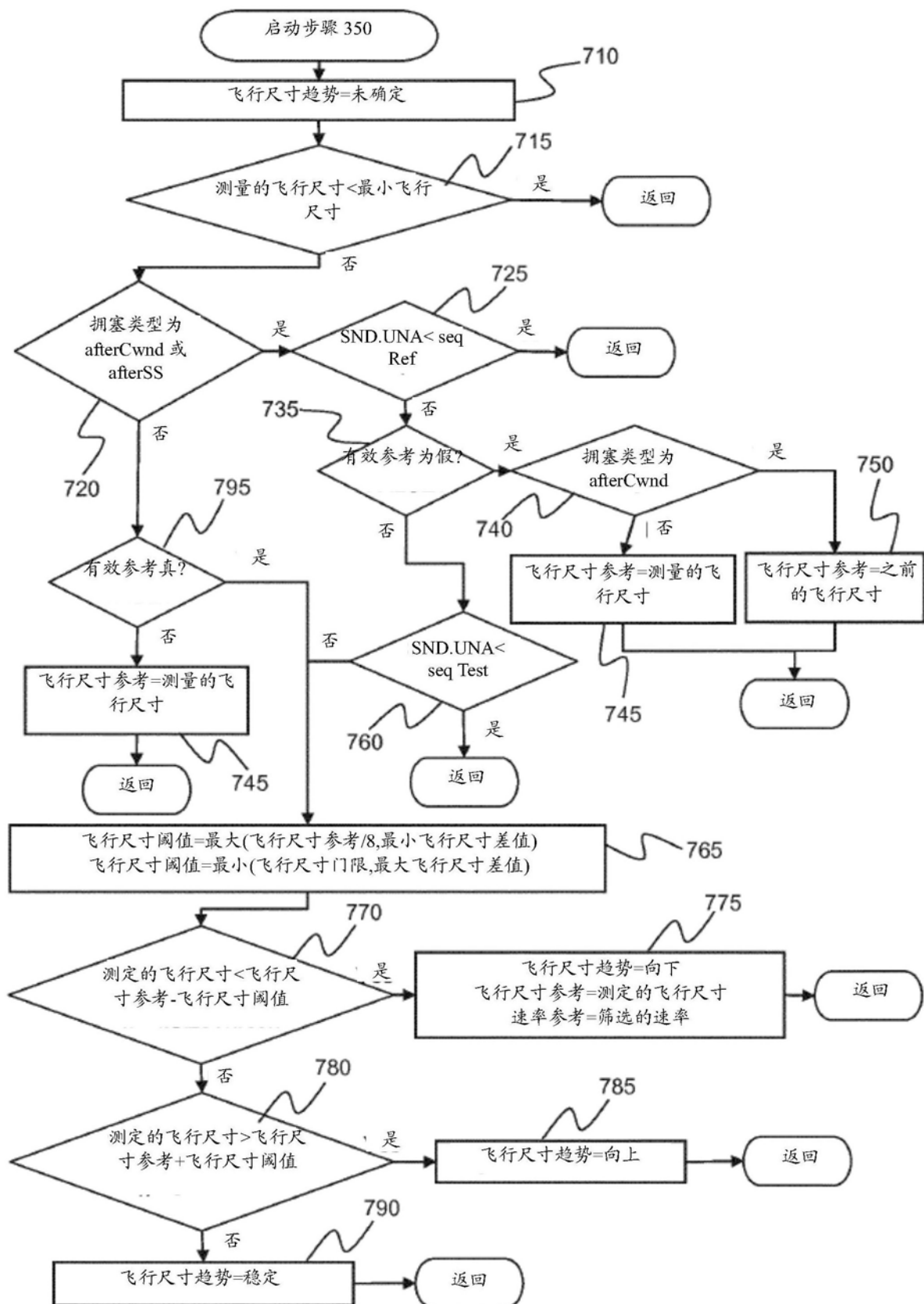


图7

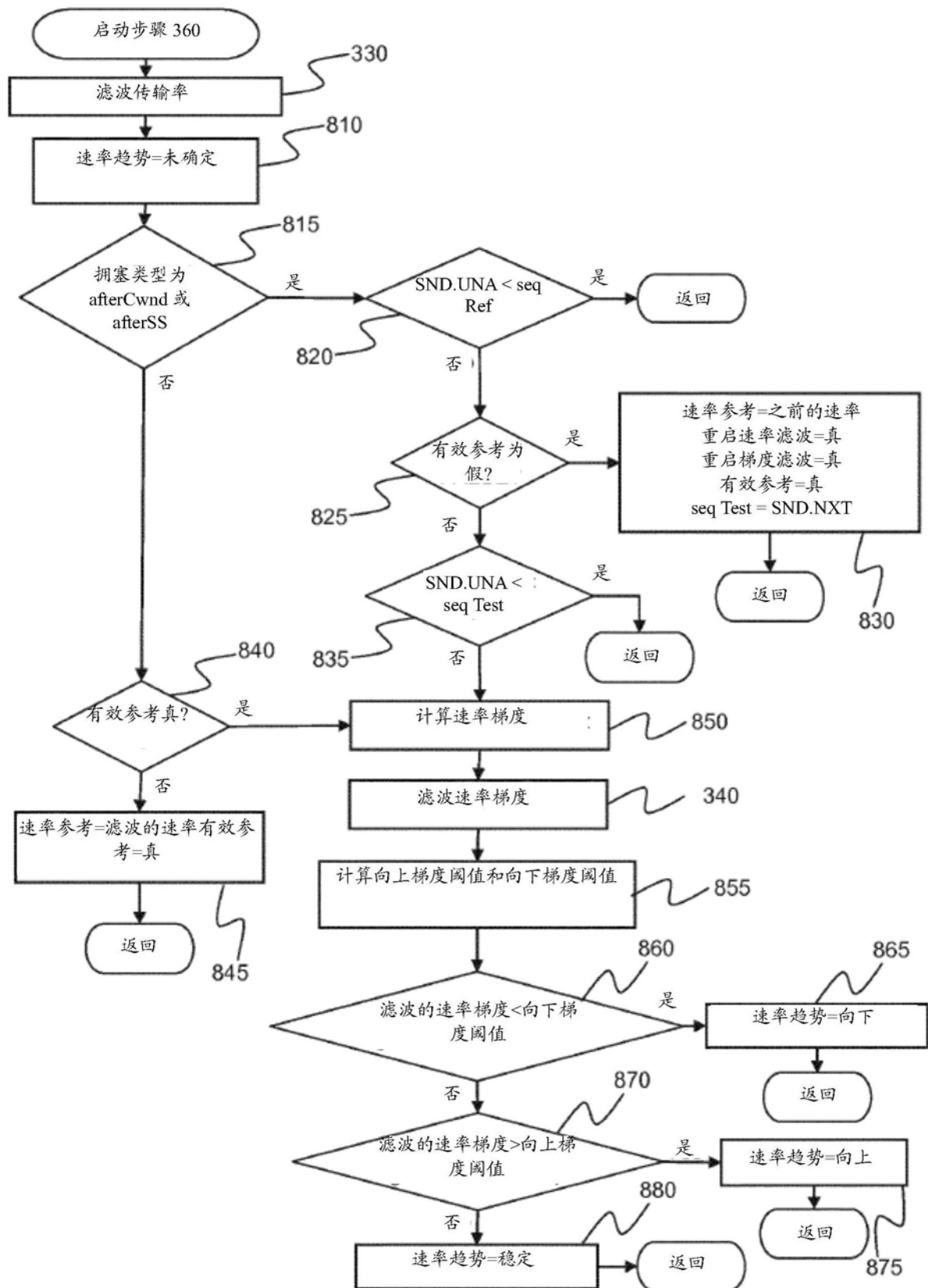


图8

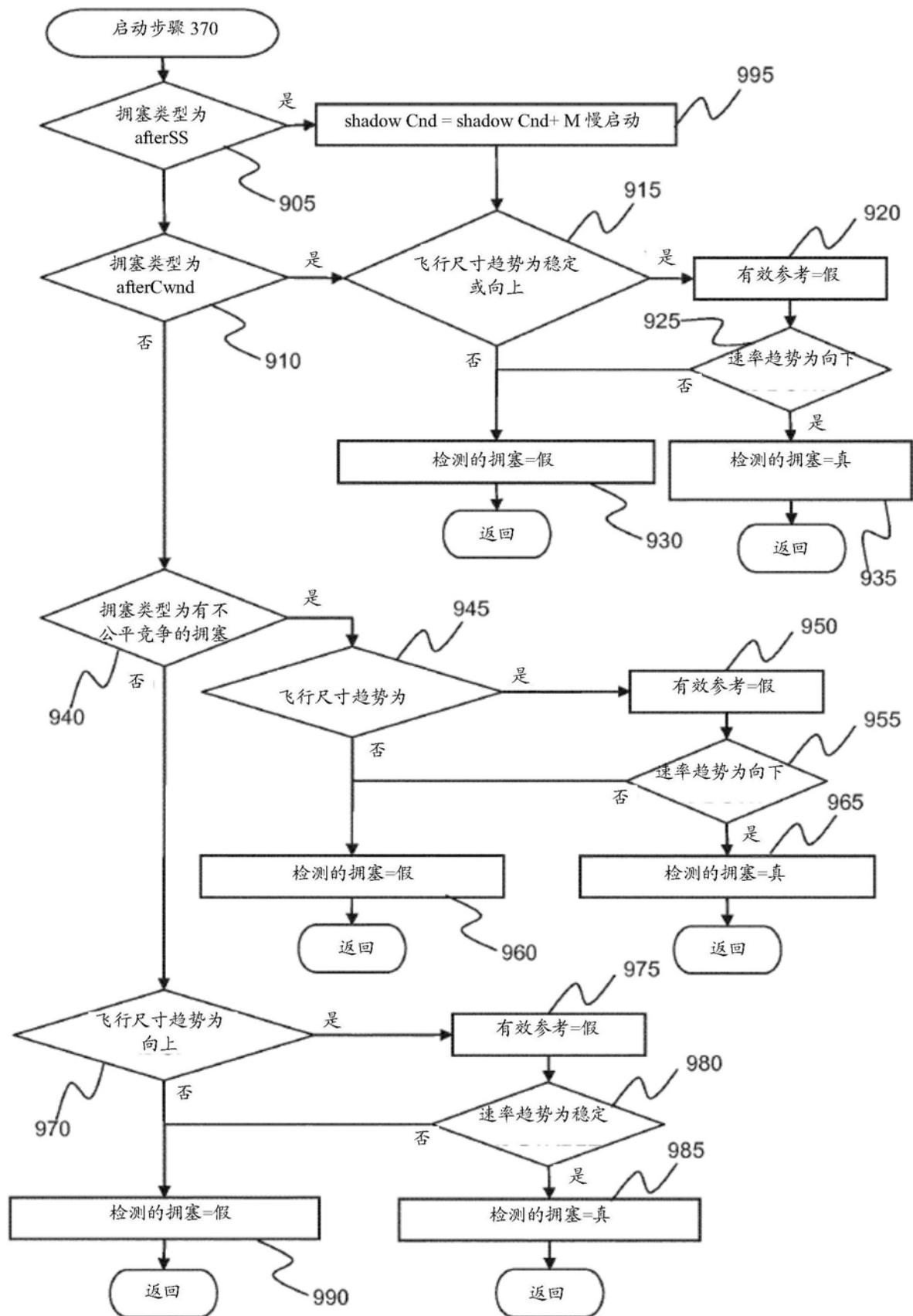


图9

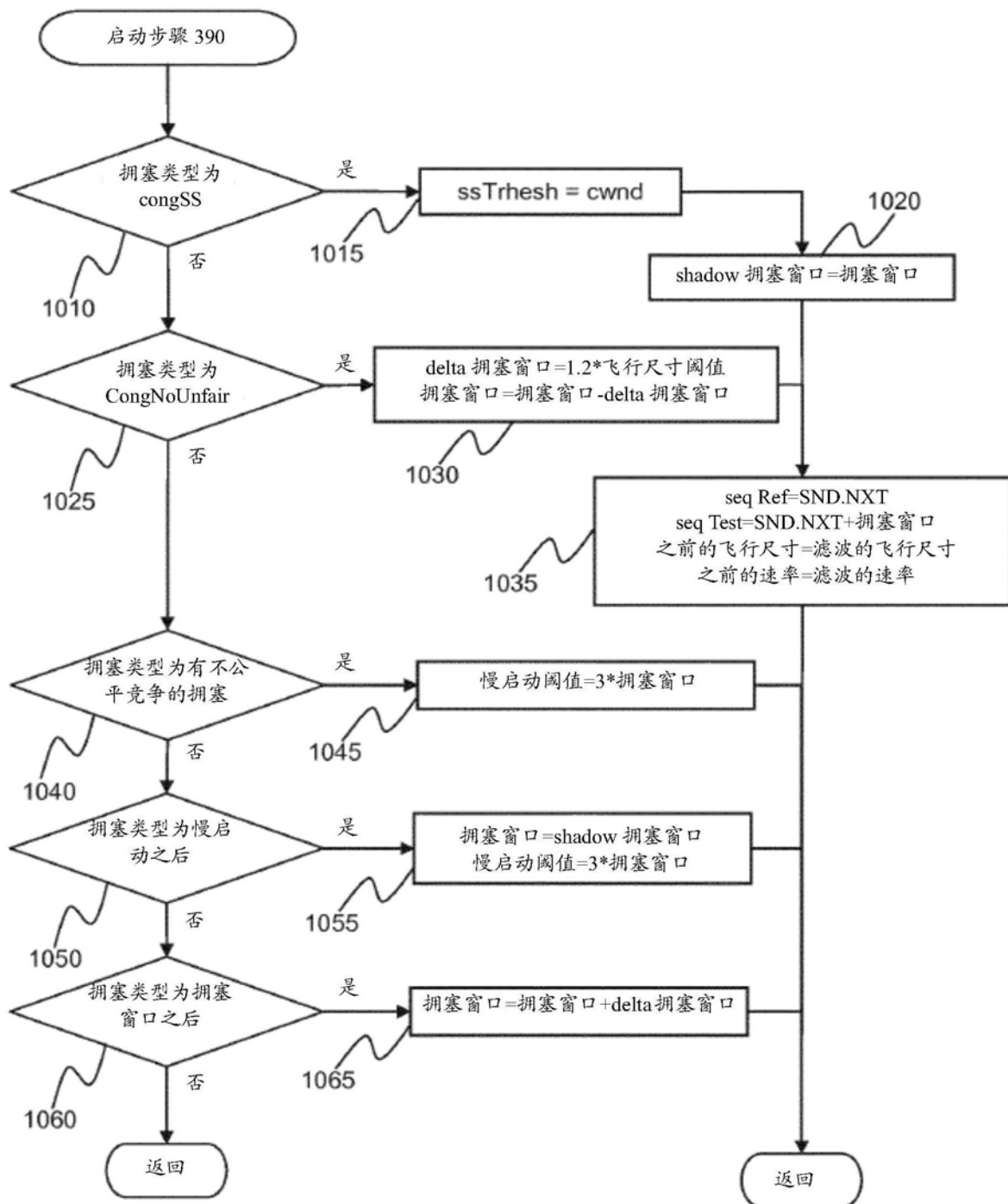


图10

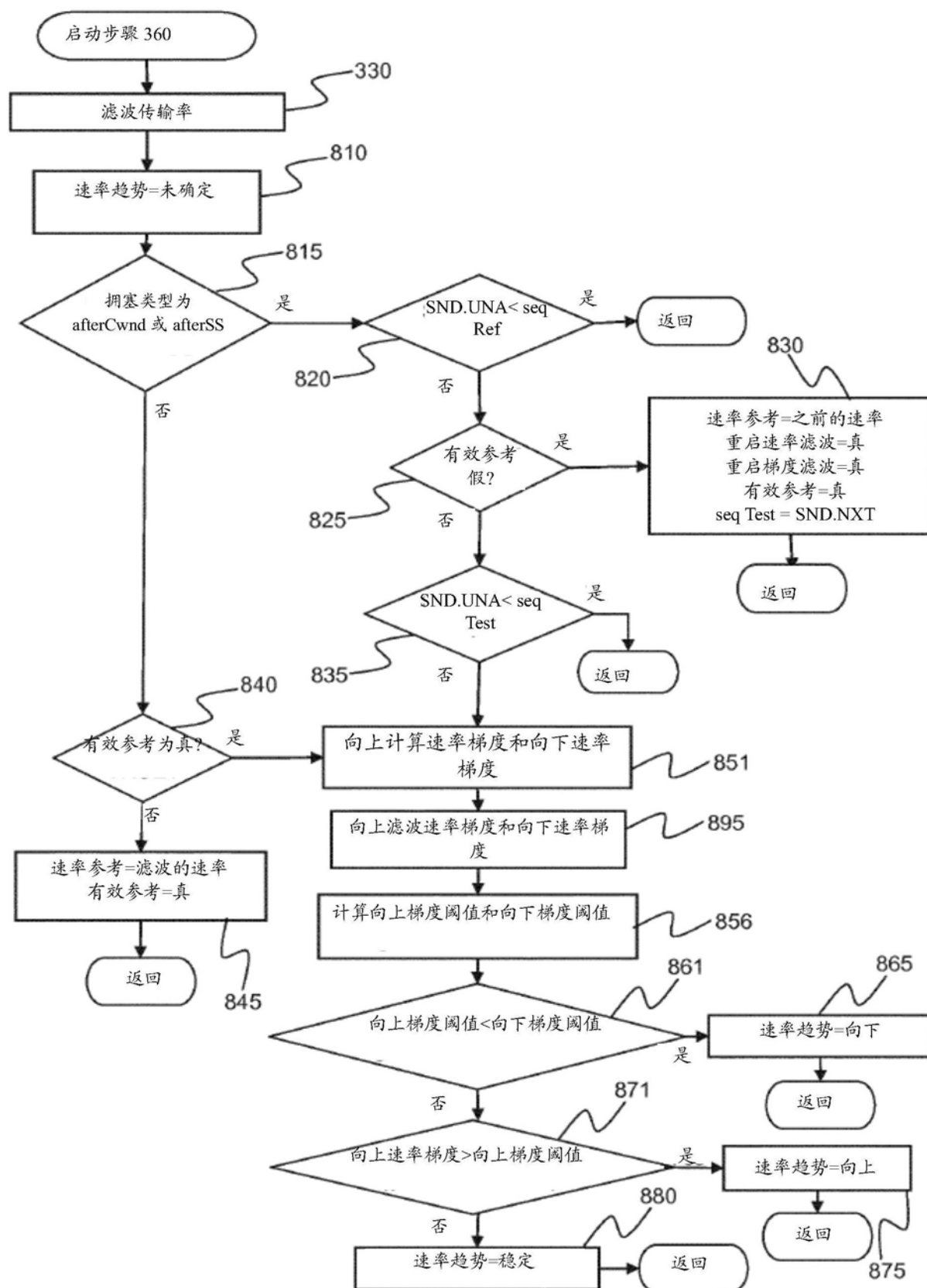


图11

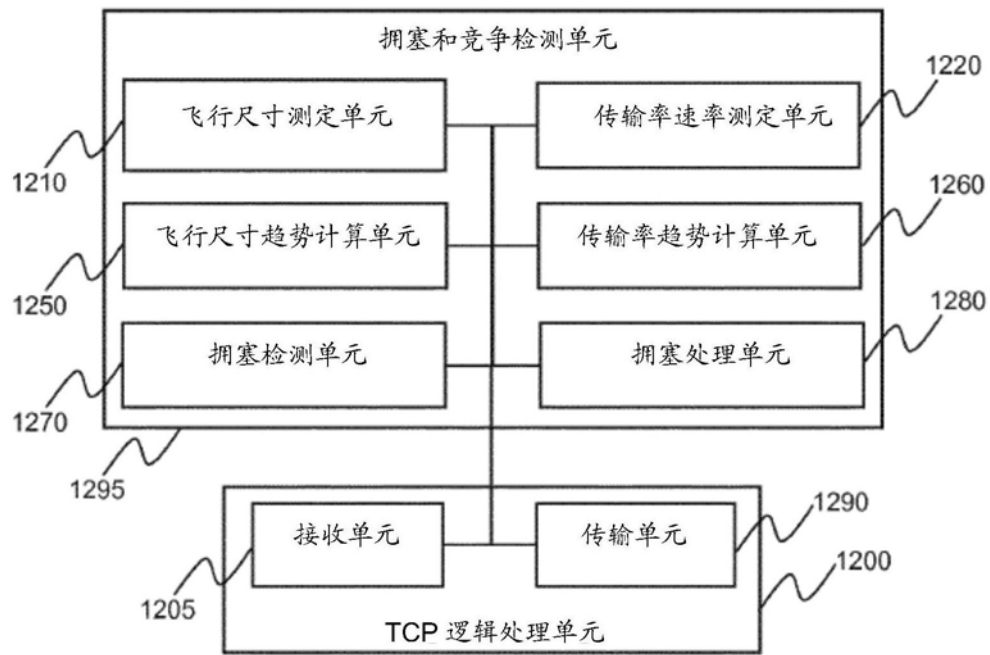


图12

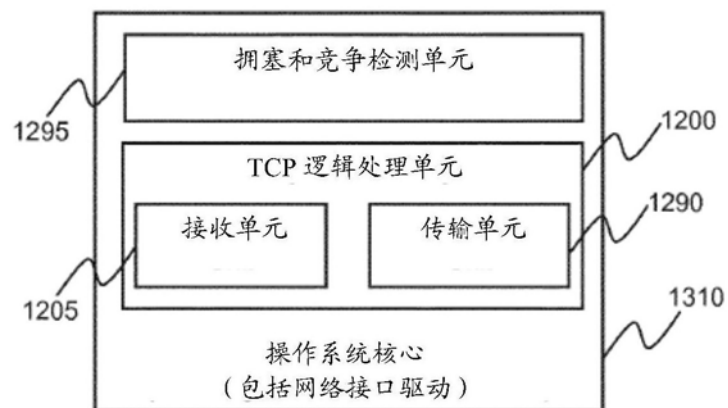


图13

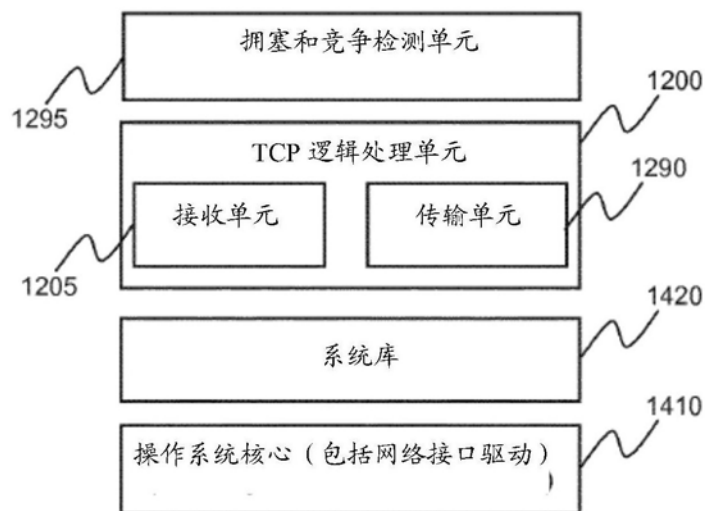


图14

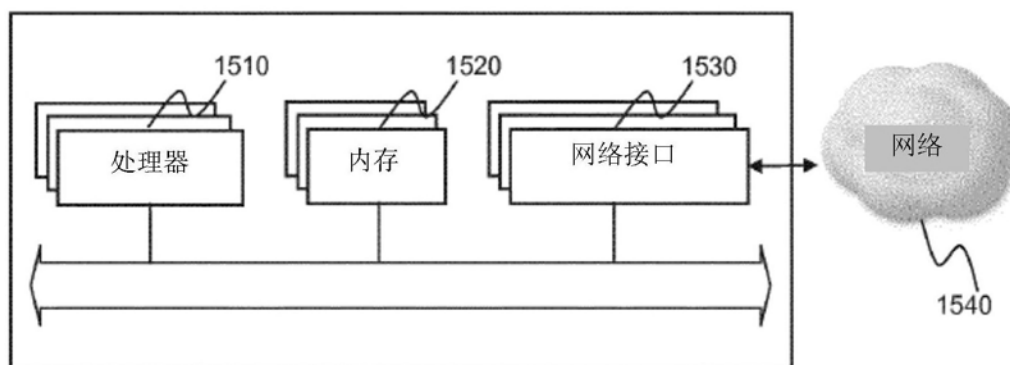


图15