第37卷 第15期
Vol.37 No.15

计 算 机 工 程
Computer Engineering

2011年8月
August 2011

·网络与通信·

文章编号：1000—3428(2011)15—0079—03 文献标识码：A 中图分类号：TP393

基于DupAck的Ad Hoc网络TCP性能改进

刘 磊，冯慧芳

(西北师范大学数学与信息科学学院，兰州 730070)

摘 要：为改进传输控制协议(TCP)在无线网络环境下的性能，分析Ad Hoc网络数据丢失的原因，提出一种区分无线丢包和拥塞丢包的算法。该算法通过在发送端检测返回的重复Ack的相对单向传输时延，探测到网络真实的拥塞状况，以便采取合理的拥塞控制措施。仿真结果表明，该算法能够正确区分无线丢包和拥塞丢包，改善Ad Hoc网络的TCP性能。

关键词：Ad Hoc网络；传输控制协议；无线丢包；拥塞丢包；性能改进

TCP Performance Improvement for Ad Hoc Network Based on DupAck

LIU Lei, FENG Hui-fang

(College of Mathematics and Information Science, Northwest Normal University, Lanzhou 730070, China)

【Abstract】In order to improve the Transmission Control Protocol(TCP) performance over wireless networks, the possible reasons of data loss for Ad Hoc network is discussed and the efficient of loss differentiation scheme is proposed. It designs the scheme of detection based on one-way trip time of duplicate Ack. According to the scheme, the sender can adopt immediately the appropriate congestion control. Simulation results show that the proposed scheme can improve the TCP performance efficiently.

【Key words】Ad Hoc network; Transmission Control Protocol(TCP); wireless lost packet; congestion lost packet; performance improvement

DOI: 10.3969/j.issn.1000-3428.2011.15.024

1 概述

有线网络的TCP拥塞控制协议认为，一旦数据包丢失就是网络拥塞导致的，马上启用拥塞控制过程，减小发送速率[1]。然而Ad Hoc网络由于其传输媒介为空气，影响空气传输的因素又比较复杂，因此在数据传输过程中，导致丢包的原因就有很多，大致可以分为如下4种[2-4]：(1)比特误码率。无线信道在传输过程中，不可避免地要受到周围环境的影响和干扰，当数据受到干扰后，接收端得到的数据就不可靠，认为丢包了，就会给发送端发送重复确认帧(DupAck)让其重发数据。(2)数据包乱序。因为路由传输的多路径，导致数据传输经过不同的路径，接收端如果收到的数据不是按序到达的，也认为丢包了，同样会发送DupAck让发送端重发数据。(3)用户移动导致路由切换。人们青睐无线通信设备的原因就是因为其方便快捷和可移动性。而无线的覆盖区域又是有限的，当用户从一个区域移动到另一个区域时，路由表就变化了，需要重新构造路由，在路由构造完成前发送的数据包就会丢失。(4)网络拥塞。用户对网络资源的“求”大于了网络能提供给用户的“供”后，网络就处于持续超载的状态，导致数据包丢失。一般情况下，将前3种丢包合并称为无线丢包，第4种丢包称为网络拥塞丢包[5]。

在传统的TCP拥塞控制机制中，当发送端连续收到至少3个DupAck或超时器超时，就认为该数据包丢失是由于网络拥塞导致的，因此启动拥塞控制机制。而在无线网络中，如果将丢包都认为是拥塞丢包，就有可能将无线丢包也误认为是拥塞丢包，TCP调用拥塞控制减小拥塞窗口，降低发送速率，就会使无线网络性能降低。也就是说，传统的TCP协议无法区分数据包丢失的真正原因，从而频繁地进行拥塞控制，其结果导致了无线TCP性能下降。为此，人们在这方面做了大量的探索和研究，以改进传统的TCP协议，使其适应现今日益普及的Ad Hoc网络环境。

目前区分无线丢包和拥塞丢包的主要方法分为显式丢包检测与隐式丢包检测2类。显式丢包检测是使发送方的传输层通过意识到无线链路层的状态，从而区分出拥塞丢包和无线丢包[6]；隐式丢包检测是接收端或发送端通过某些参量的变化隐式地推断出丢包的类型[5-8]。隐式丢包检测技术因其实现简单、系统开销小而成为人们关注的焦点。本文在已有研究工作的基础上提出一种新的隐式丢包检测算法，以有效改善Ad Hoc网络的TCP性能。

2 TCP拥塞控制的改进方法

文献[5]提出基于单向时延ROTT(Relative One-way Trip Time)的丢包区分算法，即Spike-train算法，其思想是通过报文段从发送端到达接收端的单向时延ROTT，由接收端根据ROTT的值推断丢包的类型，然后将该信息反馈给发送端，发送端采取合理的TCP拥塞控制。

受该文献的启发，本文提出由发送端根据DupAck的ROTT来判断网络丢包类型，然后直接采取合理的TCP控制机制。和文献[5]算法不同的是，本文直接由发送端判断丢包类型，而不是由接收端判断，之后再将判断结论反馈给发送端。由于无线传输中存在的链路不对称性，因此和Spike-train算法相比，本文算法能够更加及时地判断网络的状态，进而

基金项目：甘肃省教育厅科研基金资助项目(0901-03)；西北师范大学科技创新工程基金资助项目(NWNU-KJCXGC-03-52)

作者简介：刘 磊(1983－)，男，硕士研究生，主研方向：网络性能评价；冯慧芳，副教授、博士

收稿日期：2011-01-25 **E-mail：**xiaosaliulei@126.com

提高无线网络资源的利用率。

由于确认数据包 ACK 较小，那么接收端在 ACK 中加入一个发送 ACK 时间的字段 $t_i', i=1,2,\cdots$，发送端接收时有一个收到 ACK 时间 $t_i'', i=1,2,\cdots$，两者之差表示一个 ACK 在链路上的单向传输延时。那么当出现丢包时，发送方可以得到 3 个 DupAck 的单向传输延时，分别为：$\Delta t_1=t_1''-t_1'$，$\Delta t_2=t_2''-t_2'$，$\Delta t_3=t_3''-t_3'$。

如果 Δt_1、Δt_2、Δt_3 三者的值比较接近时，此时的丢包认为是无线丢包，因为一旦信道出现干扰，产生比特误码率和数据包乱序时，接收端收到数据后，认为不是自己需要的数据，就直接发送 DupAck，且 DupAck 在返回过程中由于没有网络拥塞，不会有任何“更多”的延时，因此 3 个重复确认帧的单向延时比较接近。如果 $\Delta t1<\Delta t2<\Delta t3$，那么认为丢包是网络拥塞导致，因为网络拥塞是连锁反应，一旦网络发生拥塞，而不及时采取措施解除拥塞，网络将会更加拥塞，直接结果是 DupAck 在返回过程中的延时“越来越大”。

为此，丢包区分算法描述如下：记 $\Delta t_{\min}$ 为最小单向延时，Δt_{mean} 为单向延时均值，Δt_{dev} 为单向延时方差。如果 $\Delta t_{\min}\leqslant\Delta t_i\leqslant\Delta t_{\text{mean}}+\Delta t_{\text{dev}}/2,\ i=1,2,3$，则认为网络正常，数据丢包是无线丢包，需要重设定时器，保持拥塞窗口 Cwnd 和阈值 Ssthresh 的值不变，重传丢失的包。否则认为丢包为拥塞丢包，TCP 降低拥塞窗口 Cwnd 和阈值 Ssthresh 的值，进入 TCP 拥塞控制阶段，其状态转换流程如图 1 所示，Δt_{mean} 和 Δt_{dev} 分别采用参数为 α 滑动平均公式计算；α 取值 1/32 [8]。

$$\Delta t_{\text{mean}}=(1-\alpha)\times\Delta t_{\text{mean}}+\alpha\times\Delta t_i$$

$$\Delta t_{\text{dev}}=(1-2\alpha)\times\Delta t_{\text{dev}}+2\alpha\times|\Delta t_i-\Delta t_{\text{mean}}|$$

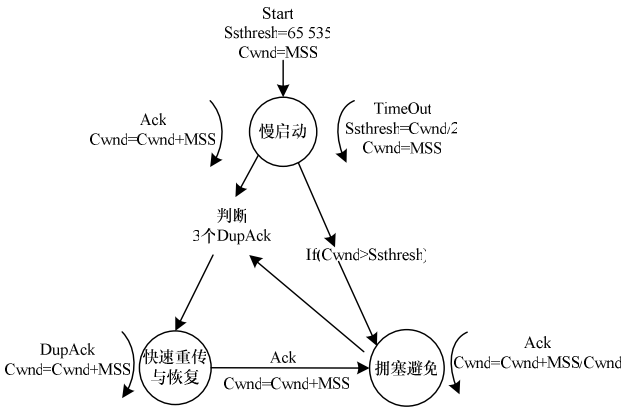

图 1 改进方法的状态流程

在图 1 中，发送方收到 3 个 DupAck 后，对拥塞窗口调整算法如下。其中，Δt_1 用代码中的变量 Dack_delay[1]表示；Δt_2 用 Dack_delay[2]表示；Δt_3 用 Dack_delay[3]表示；Δt_{mean} 和 Δt_{dev} 则分别用 Dackmean 和 Dackdev 表示。

```
Dack_delay[1]=receive[1]-send[1]
Dack_delay[2]=receive[2]-send[2]
Dack_delay[3]=receive[3]-send[3]
If((receives 3 DupAcks)
&&(Dack_delay[1]>Dackmean+Dackdev/2)
&&(Dack_delay[2]>Dackmean+Dackdev/2)
&&(Dack_delay[3]>Dackmean+Dackdev/2))
{
Ssthresh=Cwnd/2
Cwnd=Cwnd/2
}
```

3 仿真实验及 TCP 性能分析

3.1 仿真实验设计

TCP-Reno 协议是目前使用最广泛且较为成熟的算法。该算法所包含的慢启动、拥塞避免和快速重传、快速恢复机制，是现有的众多算法的基础[1,9]。

从 TCP-Reno 运行机制中很容易看出，为了维持一个动态平衡，必须周期性地产生一定量的丢失，再加上 AIMD 机制减少快、增长慢，尤其是在大窗口环境下，由于一个数据报的丢失所带来的窗口缩小要花费很长的时间来恢复，这样，带宽利用率不可能很高，且随着网络链路带宽的不断提升，这种弊端将越来越明显。

基于此种原因，本文将改进的思路运用于 TCP-Reno 算法中，通过比较以达到改善其性能的目的。

本文采用 NS-2 仿真软件[10]讨论本文提出的算法对网络性能的影响。采用本文算法的 TCP 拥塞控制协议记为 TCP-Reno-DupAck。仿真场景：网络规模为 50 个节点，仿真场景为 500 m×500 m 的矩形区域，数据包大小为 1 000 Byte，其他参数设置如表 1 所示。

表 1 主要仿真参数

参量名	数值
数据包大小	1 000 Byte
MAC 帧头	272 bit
确认帧大小	112 bit+PHY header
请求发送帧	160 bit+PHY header
确认发送帧	112 bit+PHY header
信道速率	11 Mb/s
传播时延	1 μs
帧间隔	20 μs
短帧帧间间隔	10 μs
分布帧帧间间隔	50 μs
路由协议	DSDV

3.2 丢包区分算法准确度

为了检验新的丢包区分算法的准确性，定义性能评价指标，即准确率 A_w 和 A_c：

$$A_w=\frac{\text{新方案识别出来的无线丢包数}}{\text{实际的无线丢包数}}$$

$$A_c=\frac{\text{新方案识别出来的拥塞丢包数}}{\text{实际的拥塞丢包数}}$$

在相同的设定下(误码率 pe=0.15 时)，重复做了 7 次模拟实验，统计出准确率 A_w 和 A_c，如图 2 所示。

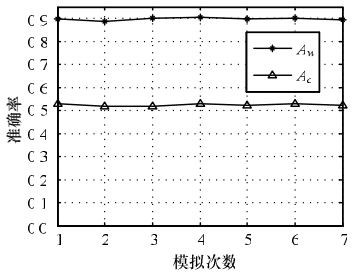

图 2 根据模拟次数统计得到的准确率

从图 2 中可以看出拥塞丢包的区分准确性均在 90%左右，无线丢包的区分准确性在 52%，也就是说如果采用 TCP-Reno 协议，则系统将这些无线丢包均认为是拥塞丢包而启动拥塞避免算法，这种错误的判断导致系统低吞吐量，采用 TCP-Reno-DupAck 算法后，至少能区分 52%的无线丢包，这时 TCP 不会盲目启动拥塞避免算法，而是保持原来的拥塞窗口 Cwnd 和阈值 Ssthresh 不变，继续发送数据，这样就使整个系统的吞吐量不会下降。

图 3 是不同数据包大小情况下 A_w 和 A_c 的值，可以看出数据包大小对区分算法没有太大影响。

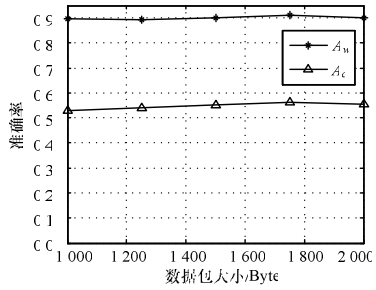

图 3　根据数据包大小统计得到的准确率

3.3　TCP 性能分析

图 4~图 6 是 TCP-Reno-DupAck 算法和 TCP-Reno 算法在不同信道误码率(pe=0.1，pe=0.2，pe=0.3)下的拥塞窗口变化情况。从图中可直观地看出，和 TCP-Reno 相比较，TCP-Reno-DupAck 并没有频繁地调用拥塞控制，拥塞窗口的波动比较小。

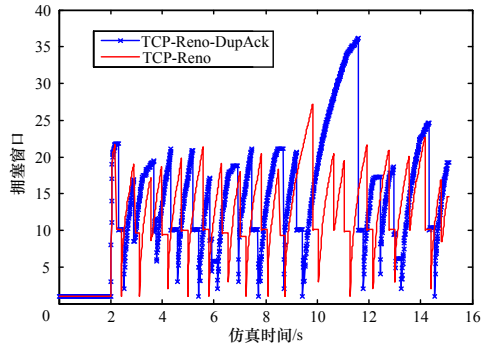

图 4　拥塞窗口的变化情况(pe=0.1)

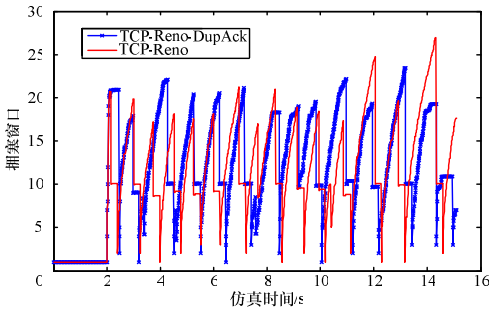

图 5　拥塞窗口的变化情况(pe=0.2)

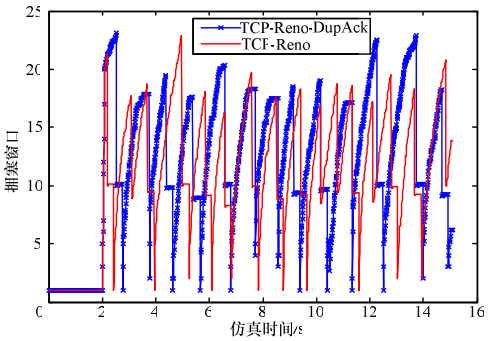

图 6　拥塞窗口的变化情况(pe=0.3)

图 7 是在不同信道误码率下，采用 TCP-Reno-DupAck 算法和 TCP-Reno 算法时系统吞吐量的变化情况。从图中可以看出，当无线信道误码率为 0.1 和 0.2 时，TCP-Reno-DupAck 算法的吞吐量明显高于 TCP-Reno 算法的。随着误码率的增加，两协议的吞吐量均呈下降趋势，当误码率为 0.3 时，无线信息质量很差，2 种算法吞吐量几乎相等。

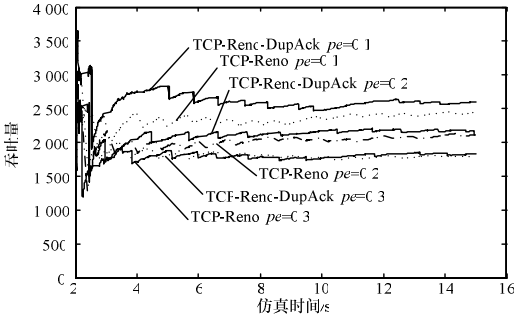

图 7　吞吐量的变化情况

由图 7 可知，本文通过重复 ACK 比较准确地判断出拥塞丢包和无线丢包，避免了直接将 TCP 应用于无线环境下所带来的诸如带宽利用率不高、吞吐量下降、系统时延加大等问题，并且由于重复 ACK 数据量小、传输速度快，因此其误码的概率也小，使可靠性得到提高，从而会有效降低移动主机不必要的能源消耗和系统的额外开销。

4　结束语

在无线信道在传输过程中，不可避免地要受到周围环境的影响和干扰，信道误码率在很大程度上影响了系统的性能。根据重复 ACK 的单程传输时延，发送端能够比较及时地分析出网络状态。本文提出的无线丢包区分算法是通过在发送端检测返回的重复 ACK 相对单向传输时延，判断网络状态的真实的拥塞状况，从而达到改进无线 TCP 性能的目的。仿真实验验证了该方案的有效性。

参考文献

[1] Allman M, Paxson V, Stevens W. TCP Congestion Control[S]. RFC 2581, 1999.

[2] Ye Tian, Kai Xu, Ansari N. TCP in Wireless Environments: Problems and Solutions[J]. IEEE Communications Magazine, 2005, 43(3): 27-32.

[3] 冯彦君, 孙利明, 钱华林, 等. MANNET 中 TCP 改进研究综述[J]. 软件学报, 2005, 16(3): 434-443.

[4] Thangam S, Kirubakaran E. A Survey on Cross-layer Based Approach for Improving TCP Performance in Multi Hop Mobile Adhoc Networks[C]//Proc. of International Conference on Education Technology and Computer. Singapore: [s. n.], 2009: 294-298.

[5] Tobe Y, Aida H, Tamura Y. Detection of Change in One-way Delay for Analyzing the Path Status[C]//Proc. of the PAM'00. Hamilton, USA: [s. n.], 2000.

[6] Floyd S, Mahdavi J, Mathis M, et al. An Extension to the Selective Acknowledgement(SACK) Option for TCP[S]. RFC 2883, 2000.

[7] Wang Ren,Valla M, Sanadidi M Y, et al. Adaptive Bandwidth Share Estimation in TCP Westwood[C]//Proc. of GLOBECOM'02. Taipei, China: [s. n.], 2002: 2604-2608.

[8] Song Cen, Cosman P, Voelker G. End-to-end Differentiation of Congestion and Wireless Losses[J]. ACM Transactions on Networking, 2003, 11(5): 703-717.

[9] 张晓琴, 黄玉清, 梁　靓. Ad Hoc 网络中多种移动模型的 TCP 性能分析[J]. 计算机工程, 2009, 35(23): 95-97.

[10] 徐雷鸣, 庞　博, 赵　耀. NS 与网络模拟[M]. 北京: 人民邮电出版社, 2003.

编辑　任吉慧