软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(5):1041−1052 [doi: 10.3724/SP.J.1001.2011.03840]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

MANET 中基于链路稳定性路由的跨层传输控制协议*

孙 杰+, 郭 伟

(电子科技大学 通信抗干扰技术国家级重点实验室,四川 成都 611731)

Cross Layer Transmission Control Protocol Interacting with Link Reliable Routing in MANET

SUN Jie+, GUO Wei

(National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu 611731, China)

+ Corresponding author: E-mail: sun.jie,sc@163.com, http://www.scie.uestc.edu.cn

Sun J, Guo W. Cross layer transmission control protocol interacting with link reliable routing in MANET. *Journal of Software*, 2011,22(5):1041−1052. http://www.jos.org.cn/1000-9825/3840.htm

Abstract: Routing fails frequently because of the node mobility of MANET. The regular TCP works badly in MANET because the congestion control algorithm takes all the problems in MANET (disconnections, route changes, etc.) as congestion. To improve the efficiency of end to end transmission, this paper proposes a cross-layer transmission control protocol that interacts with link reliable routing protocols, which are called TCP-PLRT (transmission control protocol based on probability of link residual lifetime). TCP-PLRT judges the stability of end to end connection through information about the probability of the link residual lifetime that feedback from routing layer. Three complementary mechanisms have been proposed: ROUTING SWITCH, if the link is not stable enough, SAVE THEN FORWARD, if the link is about to break, and ACK RECONFIRM, if the routing has been rebuilt. Therefore, TCP-PLRT can do prediction before routing failure, and do remediation after it happens. It has been shown by computer simulation that results of TCP-PLRT have significantly improved the packet loss, caused by routing failure, and have increased the throughput much more.

Key words: transmission control protocol; cross layer; routing failure; link reliable routing; probability of link residual lifetime

摘 要: 在 MANET 中,通信节点的移动会造成端到端通信路由的时常中断.传统 TCP 协议只有拥塞控制机制,对由于节点移动造成的数据包传输丢失和超时也作为拥塞处理,使得端到端传输性能低下.为解决这一问题,采用跨层设计思想,将传输控制与链路稳定性路由结合,提出了一种基于链路生存时间概率的传输控制协议(transmission control protocol based on probability of link residual lifetime,简称 TCP-PLRT).该协议通过在路由稳定时跨层收集路由层链路生存时间概率信息来实现对端到端连接稳定性的认知,并针对路由不稳定、路由中断、路由恢复,分别制定了路由切换、数据存储转发、ACK 再确认三大机制.这使得 TCP-PLRT 协议具有对由移动造成的路由中断进行提前预判和有效处理的能力.仿真结果表明,TCP-PLRT 协议能够极大地减小由节点移动带来的端到端传输性能的

* 基金项目: 国家重点基础研究发展计划(973)(2009CB320405); 国家科技重大专项(2010ZX03005-002)
收稿时间: 2009-08-13; 定稿时间: 2010-03-11

下降,减少分组重传,提高端到端的吞吐量.

关键词: 传输控制协议;跨层;路由中断;链路稳定性路由;链路生存时间概率
中图法分类号: TP393 文献标识码: A

移动自组织网络 MANET(mobile ad hoc networks)是一种不需要固定基础设施、通信节点可以自由移动的互联网络.每个通信节点同时具有通信终端和路由转发的双重功能.由于其独特的自组织性和移动性,使得 MANET 具有很高的灵活性和生存能力,很适合临时或机动的通信场合,如战场环境、抢险救灾等.

有线网中的端到端传输控制协议 TCP 协议直接应用到 MANET,其传输性能将极大地下降[1,2].TCP 协议为有线网设计,适应低误码且路由稳定的网络环境.其控制策略只在端节点上实现,控制的对象只有网络拥塞,不对网络状况进行主动认知,只被动等待丢包后作出反应.而在 MANET 中,开放的无线信道以及移动的通信节点等特性使得传输控制的对象变得更多,情况更复杂,仅依靠在端节点上的被动反应来制定控制策略已显得力不从心.让端到端传输中间节点参与传输控制,是改善传输性能的重要途径之一.同时,利用跨层设计思想,让传输控制协议获知其他协议层信息主动认知网络状况,将能够更准确地区分不同网络事件,从而正确作出相应的控制反应.

目前,国内外针对 MANET 中的传输控制协议已经有了一些研究.已有的研究表明,节点移动性造成的路由中断是造成 MANET 中传输性能低下的最主要原因[1,2].文献[3,4]分别提出了基于网络中间节点信息反馈的传输控制协议 TCP-F 和 TCP-ELFN,其基本思想是,依靠中间节点显式向源节点反馈路由中断和路由恢复信息,以解决 MANET 中的路由改变问题.文献[5]提出在 IP 层和 TCP 层之间添加名为 ATCP 的新层次来监视网络层和传输层的状态,以期实现对路由中断及误码丢包的检测.文献[6]提出了 Link RED 算法,其目标是通过主动控制链路层的丢包率来使得 TCP 的拥塞窗口保持在某一特定值 W^*附近,从而提高 TCP 吞吐量.文献[7]提出了一个端到端的友好传输协议.它通过在端节点对多个网络参数进行检测,以区分拥塞、信道误码、路由中断和断链.文献[8]提出利用中间节点的缓存能力来解决 MANET 中由于路由中断引起的包丢失问题.文献[9]认为,TCP 的拥塞窗口变化过于“剧烈”,提出将拥塞窗口每隔一个 RTT 增加的量由原来的 1 个 MSS 变为$\alpha(0<\alpha<1)$个,从而减缓拥塞窗口的变化速度,使得 TCP 用更小的流量来探测网络容量.

现有的 MANET 传输层改进协议很多是单独在某一协议层进行改进,不同协议层之间没有有机结合.有的虽然也跨层获取下层协议信息,但并没有针对下层协议的具体特点制定相应策略.并且,大多数改进协议是在路由中断之后才采取措施,控制策略的执行具有相当的滞后性.本文在跨层结合链路稳定性路由的基础上,利用路由协议中的链路生存时间概率信息提出了 TCP-PLRT 协议(transmission control protocol based on probability of link residual lifetime).TCP-PLRT 协议结合链路稳定性路由的具体特点,跨层收集端到端连接中各中间节点的链路生存时间概率信息,实现了对端到端连接稳定性的认知,从而能够对路由中断进行提前预判并有效处理.制定了路由切换、数据存储转发、ACK 再确认三大机制,分别针对节点处于路由不稳定、路由中断、路由重建这 3 种状态时进行处理.

本文首先介绍链路稳定性路由.第 2 节详细介绍 TCP-PLRT 协议的设计.第 3 节通过仿真对协议性能进行分析.最后是本文的总结.

1 链路稳定性路由

在本文所讨论的 MANET 网络中,所有节点的配置相同.具有相同的无线覆盖范围、物理带宽、MAC 协议、路由协议等.节点间均是双向信道且收、发速率相等.

链路稳定性路由突破了传统以最短路径作为路由选择的依据,选择具有最大生存时间概率的链路来构建路由[10,11].文献[10]在随机行走模型(random walk)的基础上计算出任意相邻节点 i,j 在已知当前距离 $d_{i,j}$ 的条件下,在时间 s 内仍保持连接的概率 $P_{i,j}(s|d_{i,j})$,并提出了 THP-AORP 路由协议.THP-AORP 协议在传递路由请求分组 RREQ 时收集所经过链路 $i-1\leftrightarrow i$ 的生存时间概率 $P_{i-1,i}(s|d_{i-1,i})$,累乘得到节点 i 到源节点的 s 时间生存概率

$\psi_{src,i}(s)$.端到端传输中间节点根据$\psi_{src,i}(s)$更新自身路由表,目的节点根据$\psi_{src,des}(s)$最大且跳数最小原则选择最优路径.协议还有“目的节点对路由请求的二次应答”、“中间节点对路由的反向优化”等一系列机制,使其与 AODV 相比具有更小的链路断裂概率和更长的链路生存期.文献[11]同样基于随机行走模型计算链路稳定性,计算出在已知节点 i,j 邻居关系已建立时间 $t_{i,j}$ 的条件下,在时间 s 内二者仍保持连接的概率 $P_{i,j}(s|t_{i,j})$,并基于此制定出 CPM-PRRA 路由协议.

链路稳定性路由打破了传统路由协议链路只有通、断两种状态的模式,用链路生存时间概率来描述链路状态,更符合 MANET 无线动态网络的特性.同时,也为传输层获取端到端连接稳定状态提供了条件.传输控制协议可以通过监视端到端连接中各段链路的 s-时间生存概率 $P_{i,j}(s)$,实现对路由中断的预判,提前采取措施以减少对传输性能的影响.

2 基于链路稳定性路由的传输控制协议 TCP-PLRT

2.1 TCP-PLRT协议的设计

在 MANET 路由协议中,节点周期发送 Hello 报文以通告邻居关系.如果在一定时间内没有收到某邻居节点的 Hello 报文,则认为邻居关系不再成立,删除到该节点的路由表项.此时,若有数据需要转发至该节点,即认为发生了路由中断(断链),此节点即是断链节点.

TCP-PLRT 协议跨层结合链路稳定性路由,并不针对某一特定路由协议设计.只要路由协议能够提供链路生存时间概率信息,就可以与 TCP-PLRT 协议相结合.为了能够与 TCP-PLRT 协议形成有机整体,对路由协议作以下必要限制,但并不涉及路由协议自身的基本机制:(1) 路由协议不仅在路由建立阶段计算链路生存时间概率,转发传输层数据包时也做同样的计算;(2) 规定发生路由中断时采用断链节点发起路由查找的局部修复方式.

为了仅得到特定端到端连接中各段链路的稳定信息,TCP-PLRT 协议在 ACK 包头开辟一个新字段 SoC (status of connection)以传递链路稳定状态.为了减轻传输开销,SoC 中并不包含各段链路具体的生存时间概率,而仅用来表示整个端到端连接是否稳定.SoC 初始为 0,表示端到端连接稳定.端到端连接路径中某节点在转发传输层数据包(DATA 和 ACK)时,先计算到自身下一跳(更远离源节点)的链路生存时间概率 $P(s)$.若 $P(s)$小于断链概率 P_D(disconnect probability)(P_D 及 $P(s)$的具体计算方法因路由协议不同而不同,不是本文重点,将另文加以研究),则在此后转发 ACK 时修改 SoC=1 通知源节点端到端连接已不稳定;同时,该节点不等待链路实际断开,立即发起新路由查找.

如图 1 所示,设某节点 i 在 t_0 时刻触发 P_D,此时,当前路由还能维持时间 T_L,查找新路由需时 T_R,节点 i 在 t_0 之后第 1 次转发的 ACK 经时间 T_{ack} 到达发送端.设源节点的平均数据发送率为λ_s,则从 t_0 时刻开始到源节点收到该 ACK 为止共有$\lambda_s T_{ack}$的数据被发送.设数据包从源节点到达节点 i 需时 T_{tra},则在 t_0 之前源节点已发出但尚未被节点 i 收到的数据量为$\lambda_s T_{tra}$.因此,即使源节点在收到 SoC=1 的 ACK 后停止数据发送,若不想发生丢包,则节点 i 从 t_0 时刻开始仍需转发$\lambda_s(T_{ack}+T_{tra})$的数据包.

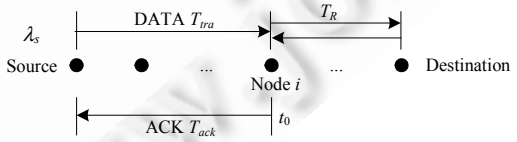


Fig.1 End-to-End connection where node i has routing failure

图 1 端到端连接示意(node i 处发生断链)

因此,要最大限度地减少路由中断所造成的丢包.TCP-PLRT 协议设计的根本是,各个节点分别确定适合自身的路由维持时间 T_L,尽量满足:(1) $T_L \geqslant T_R$,使得路由不中断;(2) 当不能满足 $T_L \geqslant T_R$ 时,当前路由至少能保证已

发出的$\lambda_s(T_{ack}+T_{tra})$数据包被传输(在所有节点收、发速率相等的假设下等价于 $T_L \geqslant (T_{ack}+T_{tra})$),或不被丢弃.

2.2 T_L的确定

为了确定合适的 T_L,所有节点都应该收集:(1) 本节点到目的节点的路由建立时间 T_R;(2) ACK 包从本节点到源节点与对应的 DATA 包从源节点返回的时间之和$(T_{ack}+T_{tra})$.为了能够提供不同端到端连接的链路稳定信息,各个节点对相邻的不同路由转发节点计算不同的 T_L,并仅在转发传输层数据包时计算 T_L,转发路由数据包时不计算.

对于 T_R 的收集,各个节点记录自身转发 RREQ 的时间,收到对应 RREP 时记录收到时间,二者之差即为路由建立时间(routing setup time,简称 RST)的一次采样值.由于重路由后到目的节点的跳数可能发生变化,因此,更具意义的参数是平均每跳路由建立时间(average routing setup time in each hop,简称 RSTEH).RREP 分组中包含跳数信息,节点收到 RREP 的同时记录距离目的节点的跳数 *hop*,按公式(1)计算 RSTEH 的一次采样.

$$Sam\_RSTEH=Sam\_RST/hop \tag{1}$$

对于$(T_{ack}+T_{tra})$(ACK and DATA round time,简称 ADRT),中间节点在转发数据包时跨层获取上层数据包类型,判断是否为 ACK 包.若是,则记录下转发时间和 ACK 的确认号(acknowledgment number),然后仅在第 1 次按顺序收到对应序号(sequence number)的 DATA 数据包时记录收到时间(忽略重传和乱序分组),二者之差即为 ADRT 一次采样.同时,需要记录的还有 ACK 的目的地址和目的端口.目的地址用以区别不同源节点,目的端口的作用是对同一源节点只记录一个端口的通信,以保证 ACK 与 DATA 序号对应的唯一性.

公式(2)、公式(3)所示为根据 T_R 和$(T_{ack}+T_{tra})$的采样值计算其典型值的公式.在公式(2)中,认为网络拓扑在一次路由查找时间内没有剧烈变化,因此使用最新的跳数信息 *last_hop* 来计算 T_R.两式均采用指数加权移动平均(exponential weighted moving average,简称 EWMA)来对二者的采样值进行处理.EWMA 是处理时间序列数据的一种常用方法,在通信领域很多地方都得以应用,TCP 协议中的 RTT 即采用这种方法.$(T_{ack}+T_{tra})$完全等效 RTT,T_R 也与 RTT 类似,记录一个端到端传输来回的时间;且同时,TCP-PLRT 中需要的是 T_R 和$(T_{ack}+T_{tra})$取值空间的上界值,因此,根据文献[12],公式(2)、公式(3)中α,β,ε的经典值应为 0.125,0.25,4.这样的取值将使得采样值和平均值之间的方差足够小,同时在较快收敛速度的条件下得到最简单的算法表达.

$$\begin{cases} Est\_RSTEH = (1-\alpha)\times Est\_RSTEH + \alpha \times Sam\_RSTEH \\ Dev\_RSTEH = (1-\beta)\times Dev\_RSTEH + \beta \times |Sam\_RSTEH - Est\_RSTEH| \\ RSTEH = Est\_RSTEH + \varepsilon \times Dev\_RSTEH \\ T_R = RSTEH \times last\_hop \end{cases} \tag{2}$$

$$\begin{cases} Est\_ADRT = (1-\alpha)\times Est\_ADRT + \alpha \times Sam\_ADRT \\ Dev\_ADRT = (1-\beta)\times Dev\_ADRT + \beta \times |Sam\_ADRT - Est\_ADRT| \\ T_{ack} + T_{tra} = Est\_ADRT + \varepsilon \times Dev\_ADRT \end{cases} \tag{3}$$

为了进一步地分析 T_L 的最佳取值方式,首先如表 1 所示列出发送端在收到 *SoC*=1 之后采取不同措施可能出现的情况.其中,λ_D 为假设源节点在收到 *SoC*=1 的 ACK 后继续发送的速率,λ_s 为正常状态下的速率.从表中可以看出,若能保证 $T_L \geqslant T_R$ 始终满足,即在路由中断前就切换到新路由,则路由中断不会出现,传输控制层也不需采取任何措施.而当 $T_L<T_R$ 时,T_L 的不同取值范围会对传输控制产生巨大影响.

Table 1 Situation when T_L in different interval

表 1 T_L 在不同取值区间的情况

Conditions		Time of routing failure	Queue length at routing failure node	
			Source node stop transmitting when received *SoC*=1	Source node transmitting at rate λ_D when received *SoC*=1
$T_L \geqslant T_R$		0	0	0
$T_L<T_R$	$(T_{ack}+T_{tra}) \leqslant T_L<T_R$	(T_R-T_L)	0	$\lambda_D(T_R-T_L)$
	$T_L<(T_{ack}+T_{tra})<T_R$	(T_R-T_L)	$\lambda_s(T_{ack}+T_{tra}-T_L)$	$\lambda_s(T_{ack}+T_{tra}-T_L)+\lambda_D(T_R-T_{ack}-T_{tra})$
	$T_L<T_R \leqslant (T_{ack}+T_{tra})$	(T_R-T_L)	$\lambda_s(T_R-T_L)$	$\lambda_s(T_R-T_L)$

第 2.1 节中已经论述,T_L 的设置应首先满足 $T_L \geqslant T_R$,不能满足时应使得 $T_L \geqslant (T_{ack}+T_{tra})$.但同时,$T_L$ 的取值不能过大,否则会造成频繁重路由;若取值过小则又会使得路由更新不及时,产生路由中断.因此,T_L 的取值必须是一个折中的方案.由于 T_R,$(T_{ack}+T_{tra})$分别与节点到源节点的跳数成反比和正比,当断链节点离源节点较远时,T_R 较小,$T_L \geqslant T_R$ 较容易满足,同时不会造成重路由过频繁;当离源节点较近时,若强制满足 $T_L \geqslant T_R$ 则会造成频繁的重路由,反而加重网络负担.而此时$\lambda_s(T_{ack}+T_{tra})$较小,不丢包条件更易满足.因此,分析表 1 可知,T_L 的最佳取值方式应如公式(4)所示,取 T_R 和$(T_{ack}+T_{tra})$中较小者.同时,发送端收到 *SoC*=1 后仍可继续发送(详见第 2.3 节).这虽然可能造成数据包在断链节点累积,但 TCP-PLRT 的“数据存储转发”机制可以解决这一问题.

$$T_L = \begin{cases} T_{ack}+T_{tra}, & (T_{ack}+T_{tra}) < T_R \\ T_R, & T_R \leqslant (T_{ack}+T_{tra}) \end{cases} \tag{4}$$

2.3 “路由切换”与“数据存储转发”

图 2 为 TCP-PLRT 在中间节点的状态转移图.如图所示,某端到端传输的中间节点 *i* 跨层获取上层数据包信息后,计算出自身与下一跳节点 *j* 的 T_L 满足 $P(T_L)<P_D$ 时,触发“路由切换(routing switch)”机制.此时,尽管 *i*,*j* 之间的链路并没有实际断开,但节点 *i* 立刻发起新路由查找,同时按旧路由转发数据,并跨层修改所转发 ACK 中 *SoC*=1.待新路由查找完成后,删除旧路由,切换到新路由进行数据转发.考虑到同一端到端路径上其他节点同时触发“路由切换”的情况,节点 *i* 会首先跨层获取 *SoC* 信息,判断 *SoC*≠2(定义见下文),才在此后转发 ACK 时修改 *SoC*=1.若在实际断链之前 *i* 没有收到需要转发的 ACK,表明此时没有数据传输或传输速率很低,则短时的路由通断不会造成数据传输上的严重后果,因此不特别通知源节点.路由切换机制将使整个端到端连接中的各段链路始终保持较高的稳定性,减少路由中断的发生.

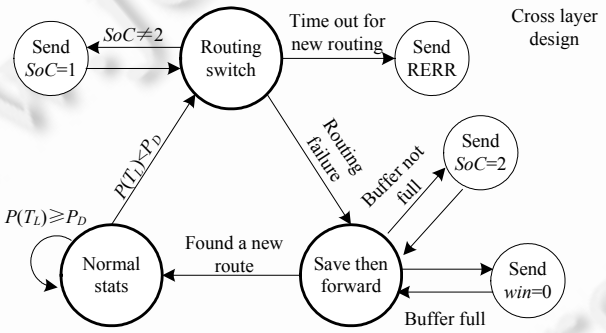


Fig.2 State transition diagram for TCP-PLRT intermediate nodes

图 2 TCP-PLRT 在中间节点的状态转移图

当已经发起新路由查找,但在新路由建立之前发生了实际断链时,节点 *i* 执行“数据存储转发(save then forward)”机制.节点 *i* 将无法转发的数据包存储在本地,并利用跨层机制对每一个存储的数据包向其源节点返回一个特殊的 ACK.该 ACK 将置 *SoC*=2,告知源节点该数据包被网络中某中间节点存储,尚未到达目的节点但并未丢失.该 ACK 并不将节点 *i* 自身的地址和端口写入发送地址和端口,而是使用其所对应的缓存数据包的目的地址和端口.这样,使得在源节点看来仍与目的节点保持着通信,使得在路由中断时仍可保持一定的吞吐量.这种特殊 ACK 有两大作用:一是使得源节点在路由中断时可以不中断发送,充分利用网络资源,并使得源节点的发送速率可控,当断链节点存储空间耗尽时还可以通知源节点停止发送(ACK“确认窗口(window)”置 0);二是和源节点间形成可靠传输,保证了数据包的完整和顺序,在路由重建之后可减少重传和乱序的产生.当新路由建立后,节点 *i* 并不特别显式通知源节点,直接将存储的数据包发往目的节点.对新到数据包不再发送 *SoC*=2 的 ACK,按新路由转发.当源节点收到来自目的节点对这些被缓存数据包的正常 ACK(*SoC*≠2)时,即知道新路由已建立.但如果节点 *i* 查找新路由超时,则立即显式发送路由错误信息 RERR 到源节点,同时丢弃存储的数据包.

考虑两种特殊情况:一种是由于节点移动性使得 $P(T_L)$在 P_D 附近震荡.这种情况同样是链路不稳定的表现,因此,一旦 $P(T_L)<P_D$ 同样进行新路由查找;另一种是同一端到端路径上多点同时触发“路由切换”的情况.即在从节点 i 触发 $P(T_L)<P_D$ 到完成新路由查找的时间内,与其在同一端到端路径上的其他节点 m 也触发 $P(T_L)<P_D$ 的情况.假设 m 的位置更靠近源节点,则 m 会收到 SoC=2 的 ACK.因为 SoC=2 已经表明路由中断,m 不再修改 SoC=1 直接转发该 ACK.m 的链路实际断开后,由于更接近源节点,它将接替 i 作“数据存储转发”.这种情况下可能有部分数据包被 i 存储,而 i 发出的 SoC=2 的 ACK 无法被源节点收到.TCP-PLRT 协议对此的处理是,源节点会重传这些数据包到 m.而 i 在路由恢复后仍然转发这些数据包,以保证一旦存储在 m 的数据包出现丢失,有部分冗余的数据包会到达目的节点.对由此造成的重复 ACK 问题,第 2.4 节中有相应的机制来解决.假设 m 的位置更远离源节点,则没有修改 SoC 的问题.若 m 存储了部分数据包,则处理方法同前.

2.4 TCP-PLRT源节点的控制策略

TCP-PLRT 源节点的状态转移图如图 3 所示.当 TCP-PLRT 源节点收到 SoC=1 的 ACK 时进入“冬眠(hibernation)”状态.源节点首先保存当前各种发送参数并冻结拥塞窗口不再增加发送速率,之后若收到正常的 SoC=0 的 ACK,则退出“冬眠”状态,恢复正常通信;若收到 SoC=2 的 ACK,则在保存发送参数之后按正常方式与断链节点通信,根据该 ACK 调整与断链节点间的传输速率,同时记录所有被 SoC=2 的 ACK 确认的数据包序号的取值区间(称为待确认区间(unacknowledged DATA sequence number interval,简称 unACK interval)).当收到真正来自目的节点的 SoC≠2 的 ACK 时(包括 SoC=0 和 SoC=1,都是来自目的节点的有效 ACK),因为认为短时间内网络拓扑没有剧烈变化,所以恢复之前保存的发送参数,恢复路由中断前的发送速率,并且根据待确认区间对被缓存数据包再一次进行 ACK 确认(ACK 再确认(ACK reconfirm)机制),确保所有数据包到达目的节点.被缓存包的 RTT 已不能反映当前网络状态,因此对符合待确认区间的 ACK 不计算 RTT,也不执行慢启动和快速重传,若丢包只简单重传.直到所有待确认区间都得到 SoC≠2 的 ACK 的再确认,源节点退出“冬眠”状态,恢复一般传输状态.源节点若在记录了待确认区间后收到了 RERR,则表明此时可能出现了网络分割,目的节点不可达,将待确认区间内的数据包作丢包处理,重新开始新路由查找.

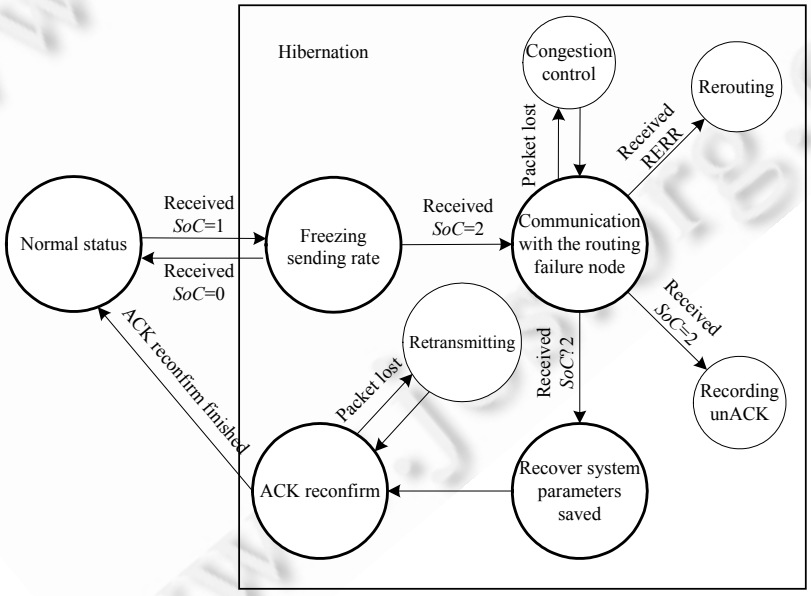


Fig.3 State transition diagram for TCP-PLRT source node

图 3 TCP-PLRT 源节点的状态转移图

由于有路由切换、数据存储转发等一系列机制,TCP-PLRT 认为路由中断造成的丢包只在“冬眠”阶段发生.当收到 *SoC*=2 的 ACK 时,源节点和断链节点之间是可靠传输,因此,该阶段若发生丢包则按拥塞处理.源节点在“ACK 再确认”阶段会收到大量的 ACK.为了更好地解决 ACK 重复和乱序问题,TCP-PLRT 在该阶段启用选择确认(selective acknowledgment,简称 SACK)[13]选项.ACK 再确认机制会需求较大存储空间,但可以减小报文乱序的影响,这与 SACK 的特性相吻合.这一要求在当前技术条件下不难实现.

3 TCP-PLRT 的协议性能分析

本文将使用网络仿真软件 NS 分别在静态网络环境和随机动态网络环境下考察 TCP-PLRT 协议的性能.NS2 由 UC Berkeley 开发,采用模块化设计具有很好的可扩展性,是当今网络研究领域应用最广泛的网络仿真软件之一[14,15].

在相同网络场景下将 TCP-PLRT 与 TCP-ELFN[4]和 TCP newReno[16]进行对比.前者是实际运用最多的 TCP 协议版本,一般将其作为传输控制协议改进的基础;后者是采用路由中断后显式通知源节点的典型代表,这种策略在已有的 MANET 传输层改进协议中是最常见的思路.

3.1 协议性能度量参数

仿真中主要采用以下性能度量参数:

- 协议开销比(protocol cost ratio):以 TCP newReno 协议为基准,改进协议所涉及的所有协议层的数据包之和与相同网络场景下 TCP newReno 相同协议层所产生数据包之和的比例.如 TCP-PLRT 涉及网络层和传输层的改进,其协议开销比为
  $sizeof(RREQ+RREP+RERR+DATA+ACK)_{TCP\text{-}PLRT}/sizeof(RREQ+RREP+RERR+DATA+ACK)_{TCP\ newReno}$.
- 路由中断时间(time of routing failure):从所需的路由表项被删除到新路由表项建立的时间.该参数反映路由协议的收敛速度.路由中断时间越短,对数据传输的影响越小.
- 断链节点队列长度(the queue length at routing failure node):断链节点在断链之前会累积大量数据包.这些数据包若丢弃则是对网络资源的浪费,若存储则又可能造成存储区溢出.对该参数的合理控制是传输控制协议应该考虑的问题.
- 分组重传率(packet retransmit ratio):由于丢包、RTO 超时、重复 ACK 等造成的所有重传包的数量与总的发送数据包的数量之比.每一次重传都是对网络资源的浪费,对于传输控制协议而言,该参数比丢包率更有意义.
- 平均端到端吞吐量(average end-to-end throughput):平均单位时间内被目的节点接收的 TCP DATA 数据包的比特数.
- 平均端到端时延(average end-to-end delay):数据包从源节点发出到被目的节点完全接收所经过的平均时间.在 MANET 中,该参数更多地反映的是网络的通畅程度.

3.2 基本仿真参数设置

仿真实验的软硬件环境如下:CPU:AMD Athlon 64 X2 4800+ 2.5GHz,内存:2G,操作系统:Ubuntu 8.10,仿真软件:NS 2.33.基本仿真参数设置见表 2.

3.3 仿真实验与分析

静态网络将采用网格(grid)结构,随机网络将采用 random walk 移动模型,这两种网络结构都是 MANET 研究中最常用的模型[6,9,17,18].本文的仿真结果都是多次不同随机种子仿真结果的平均.

仿真实验 1. 静态网络结构下的协议性能对比.

如图 4 所示建立一个 11×11 的网格网络,收发两端分别位于网络中间左右两端.模拟一个足够大的静态网络,消除网络范围对路由协议的影响.节点间距离为 200m,各个节点仅能与上下左右 4 个节点通信.

为了突出路由中断对传输控制协议的影响,仿真中仅使用 1 个 FTP 连接来产生数据传输,数据到达率 10 分

组/秒,数据净荷 1 000byte.一个 FTP 连接不会造成拥塞丢包.仿真中的无线信道没有加入干扰,且 820.11 DCF 具有链路层确认机制,因此也不会产生信道误码丢包.所有丢包或 RTO 超时事件都路由中断引起.从仿真时间 0 秒开始数据传输,在第 5 秒,当前路由上某节点以最大速度为 10m/s 的 random walk[10]模型开始移动,仿真时间持续到出现断链为止.同一速度下仿真重复 13 次,每次移动不同的中间节点,以得到距离远节点不同跳数的断链对传输协议的影响.

Table 2 Simulation parameters

表 2 仿真模拟参数

Parameters	Description	Value
Range	Simulation area size	3000mX3000m
Topology	Network topology	chain/grid/random
Mobility model	Node mobility model	random walk (for THP-AORP)[10]
Node number	Node number	90/100
Anttype	Antenna type	Antenna/OmniAntenna (Omni antenna)
Radio propagation model	Radio propagation model	Propagation/FreeSpace (free space model)
Transmit radius	Radio transmit radius	250m
Phy bandwidth	Wireless channel bandwidth	2e6 bit
ifqType	Type of the interface queue	Queue/DropTail/PriQueue
ifqLen	Max length of the interface queue	50
MAC protocol	MAC protocol	IEEE 802.11 DCF
Routing protocol	Routing protocol	THP-AORP

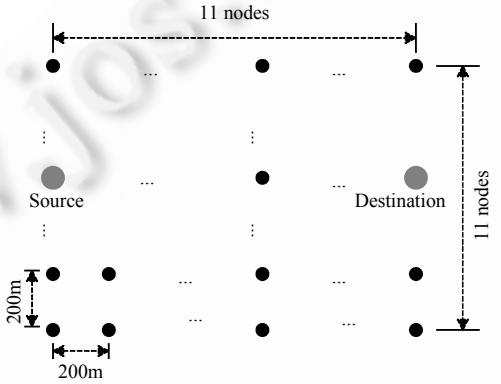


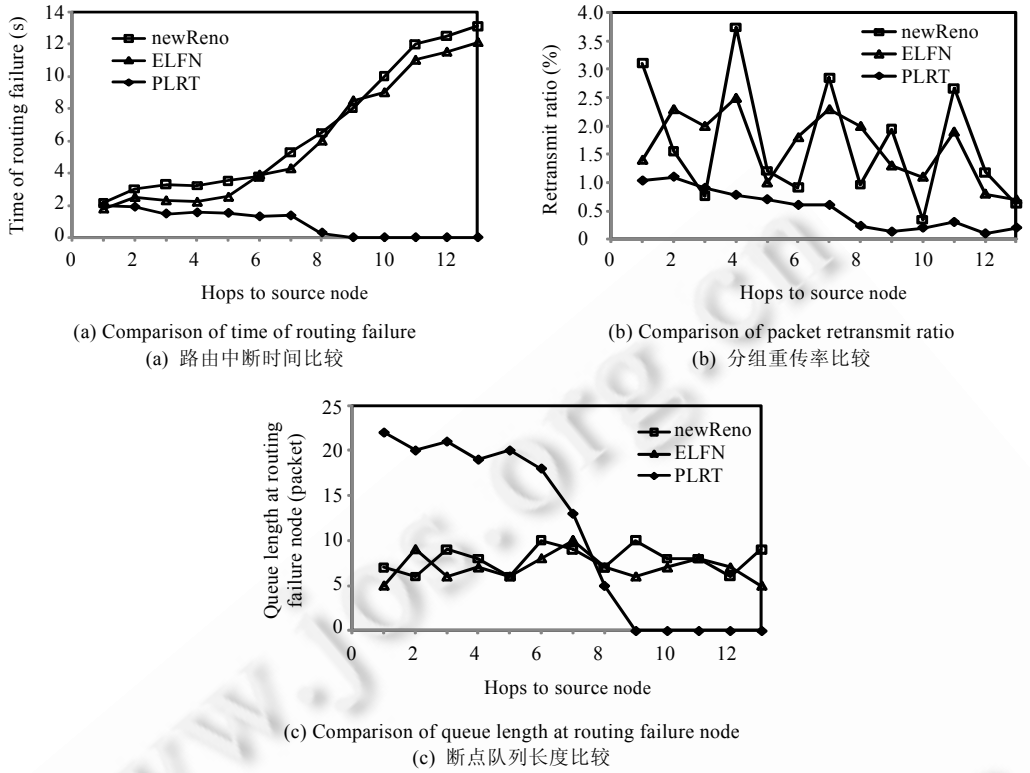
Fig.4 11×11 grid network topology

图 4 11×11 网格网络拓扑

图 5(a)是路由中断时间的对比结果.可以看出,TCP-PLRT 在路由中断时间上做出了显著改进,且随着跳数的增加表现出巨大优势.TCP newReno 的路由中断时间最长,且随着跳数的增加急剧上升.TCP-ELFN 没有对路由中断的发现做出改进,仍只是被动等待路由中断,路由中断时间大致和 TCP newReno 相同.TCP-PLRT 的“路由切换”机制实现了对路由重建时间相对精确的预测,很好地控制了路由中断时间.且 T_L 的合理设置使得 TCP-PLRT 在断链节点距离源节点较远时路由中断时间反而为 0.图 5(b)所示为 3 种协议的分组重传率比较.重传分组会引起慢启动造成传输速率的下降,而且在 MANET 这种资源受限的网络中是对网络资源的极大浪费.因此,虽然 TCP-PLRT 的数据存储转发机制使它拥有比其他两种协议更大的队列长度(如图 5(c)所示),但却极大地降低了分组重传率,减少了对网络资源的浪费.如图 5(c)所示为 3 种协议的断链节点队列长度比较.由于断链节点和源节点间是可靠传输,所以并不会造成断链节点缓存区溢出而丢包.同时,在距离源节点较远时,由于保证了路由不中断,所以反而队列长度为 0.而 TCP-ELFN 和 TCP newReno 没有对断链节点队列进行优化,任由丢包的产生造成了较高的分组重传率;同时,无论何处丢包都会有一定的排队.

仿真中,由于路由协议 THP-AORP 基于 random walk 移动模型[10],因此在随机动态网络仿真中使用该模型.单段直线运动的速率服从[0,v_{max}]上的均分,v_{max} 分别取值 5m/s,10m/s,15m/s,20m/s,单段直线移动时间服从参数

为 10s 的指数分布,移动方向服从 0~2π的均分.节点数目为 100 个,随机分布在网络场景中.随机选择 10 对节点进行数据传输,分组产生时间间隔服从 0.25s 泊松分布,分组数据净荷 1 000byte.其他参数见表 2.



(a) Comparison of time of routing failure
(a) 路由中断时间比较

(b) Comparison of packet retransmit ratio
(b) 分组重传率比较

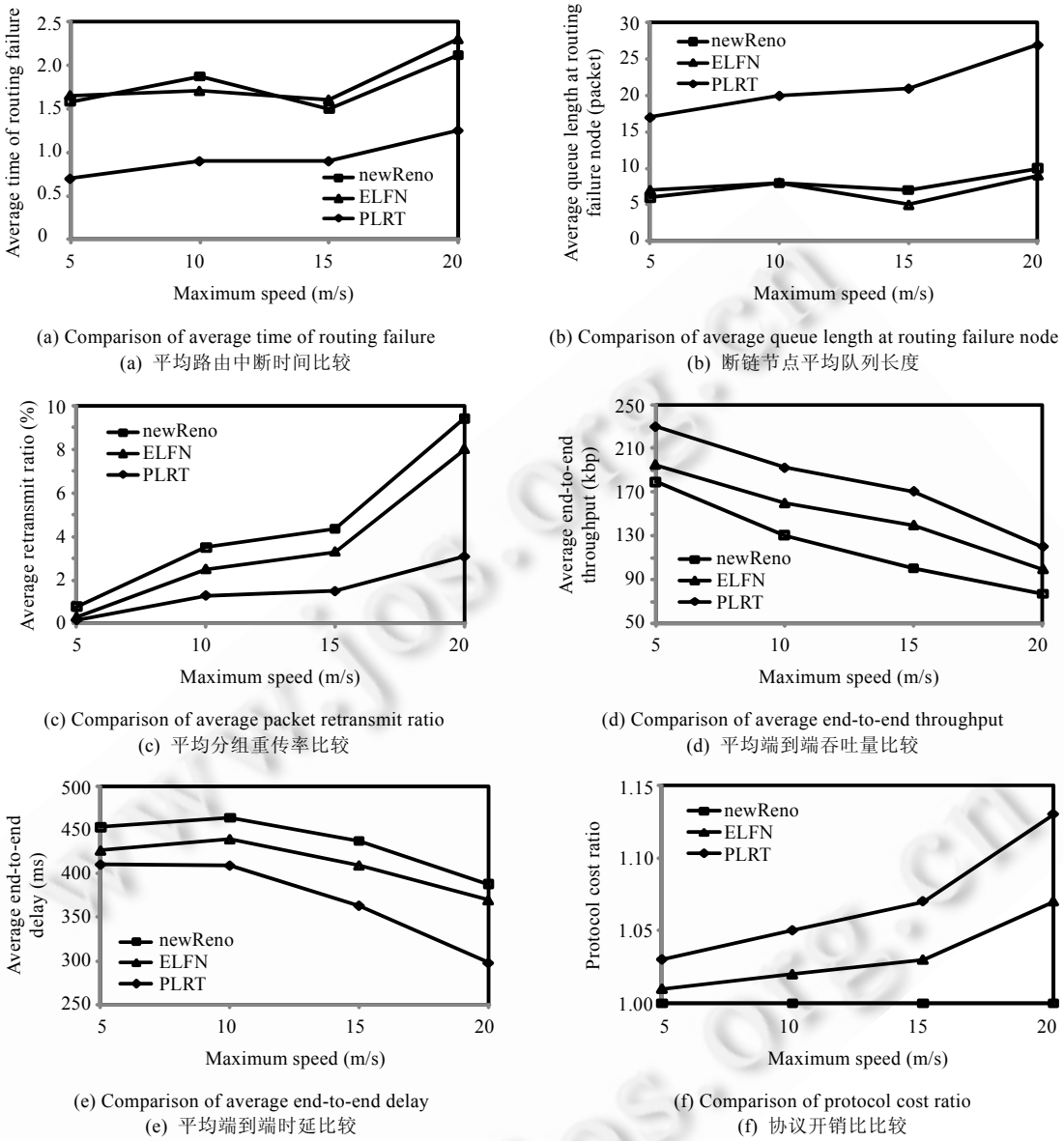
(c) Comparison of queue length at routing failure node
(c) 断点队列长度比较

Fig.5 Comparison of different TCP schemes in static topology
图 5 静态网络拓扑下不同传输控制协议的性能比较

仿真实验 2. 随机动态网络下的协议性能对比.

仿真结果如图 6 所示.TCP-PLRT 通过与链路稳定性路由协议跨层结合,实现了对路由中断的预判,从而最大限度地避免了路由中断的发生,同时控制了路由中断时间.而 TCP-ELFN 和 TCP newReno 只是被动地等待路由中断之后才作出反应.因此如图 6(a)所示,TCP-PLRT 的平均路由中断时间远低于其他两种协议.在图 6(b)中,TCP-PLRT 有着比其他两种协议更大的断链节点队列,这是 TCP-PLRT 的数据存储转发机制充分利用中间节点存储能力的结果.该机制在不导致断链节点存储空间溢出的前提下,能够极大地降低分组重传率(如图 6(c)所示),从而提高对网络资源的有效利用.而 TCP-ELFN 和 TCP newReno 在路由中断后,断链节点将会找不到路由的数据包丢弃,因此虽然有较小的队列长度,但却导致了更多的重传,浪费了网络资源.图 6(d)是 3 种协议吞吐量的对比.由于减少了丢包,使得慢启动的次数减少,同时在路由中断时不停止数据发送,使得 TCP-PLRT 有着更高的吞吐量.图 6(e)是 3 种协议平均端到端时延的仿真结果.路由切换机制缩短了路由中断时间,路由中断时不丢包而执行“数据存储”,使得 TCP-PLRT 能够在路由重建后最短时间内将数据包发往目的节点,从而降低了端到端时延.由以上对比可以看出,TCP-PLRT 协议在各项指标上都有更好的表现,但协议性能的提高却必然以增加协议开销为代价.如图 6(f)所示,TCP-PLRT 的协议开销比高于 TCP newReno 和 TCP-ELFN.TCP-PLRT 协议为了保证较高的链路稳定性,在链路实际断开之前就发起路由查找,必然导致发送更多的路由包;数据存储转发机制产生的 *SoC*=2 的 ACK 也是新增加的协议开销.但总体而言,与 TCP newReno 相比,TCP-PLRT 增加的协议开销在低速情况下(v_{max}=5m/s,10m/s)不超过 5%,在高速情况下(v_{max}=15m/s,20m/s)不超过 15%,与整体传输性能的显

著提升相比是可以接受的.



(a) Comparison of average time of routing failure
(a) 平均路由中断时间比较

(b) Comparison of average queue length at routing failure node
(b) 断链节点平均队列长度

(c) Comparison of average packet retransmit ratio
(c) 平均分组重传率比较

(d) Comparison of average end-to-end throughput
(d) 平均端到端吞吐量比较

(e) Comparison of average end-to-end delay
(e) 平均端到端时延比较

(f) Comparison of protocol cost ratio
(f) 协议开销比比较

Fig.6 Comparison of different TCP schemes in mobile random topology
图 6 随机动态拓扑下不同传输控制协议的性能比较

3.4 两种网络场景仿真结果对比

对比静态网络场景和随机动态网络场景的仿真结果,可以得到以下结论:

(1) 在静态网络下,通过端到端连接不同位置断链的仿真实验,TCP-PLRT 协议表现出了对路由中断很好的适应性,路由中断时间和分组重传率都有了显著的下降.路由中断时间至少在 TCP newReno 的 50%以下,分组重传率约为 TCP newReno 的 21%~47%.尤其是在断链节点距离源节点较远时,由于保证了 $T_L \geqslant T_R$,使得路由不中断,TCP-PLRT 表现出了其他两种协议无法比拟的优势.

(2) 在随机动态网络下,通过多次仿真的平均结果,证明了 TCP-PLRT 协议中路由切换、数据存储转发、

ACK 再确认机制的有效性,能够有效地减少因路由中断造成的丢包(TCP newReno 的 50%左右)以及路由中断时间(TCP newReno 的 30%左右),提高端到端吞吐量(比 TCP newReno 提高 30%以上)并降低端到端时延(比 TCP newReno 提高 10%以上).虽然不可避免地带来更多的协议开销,但整体传输性能的提升是显著的.

4 总 结

本文针对 MANET 网络节点移动性的特点设计了 TCP-PLRT 协议,以解决 MANET 中由于节点移动造成端到端连接中断而带来的端到端传输性能下降问题.TCP-PLRT 协议采用跨层优化思想,增加中间节点在端到端传输中的作用,结合链路稳定性路由对可能出现的路由中断进行预判,采取路由切换、数据存储转发、ACK 再确认这一系列措施来预防和减小路由中断对传输性能的影响.仿真实验结果表明,TCP-PLRT 协议能够显著提高端到端传输性能,降低分组重传率,提高吞吐量并减小端到端传输时延,同时,并不明显提高系统开销.

References:

[1] Ahuja A, Agarwal S, Singh JP, Shorey R. Performance of TCP over different routing protocols in mobile ad-hoc networks. In: Proc. of the 51st IEEE Vehicular Technology Conf. Proc. (VTC 2000). Tokyo: Springer-Verlag, 2000. 2315−2319.

[2] Fu ZH, Meng XQ, Lu SW. How bad TCP can perform in mobile ad hoc networks. In: Proc. of the IEEE Symp. on Computers and Computers (IEEE ISCC 2002). IEEE Press, 2002. 298−303. [doi: 10.1109/ISCC.2002.1021693]

[3] Chandran K, Raghunathan S, Venkatesan S, Prakash R. A feedback-based scheme for improving TCP performance in ad hoc wireless networks. IEEE Personal Communications Magazine, 2001,8(1):34−39.

[4] Monks JP, Sinha P, Bharghavan V. Limitations of TCP-ELFN for ad hoc networks. In: Proc. of the 7th Int'l Workshop on Mobile Multimedia Communications (MoMuC 2000). IEEE Press, 2000. 1−6.

[5] Liu J, Singh S. ATCP: TCP for mobile ad hoc networks. IEEE Journal on Selected Areas in Communications, 2001,19(7): 1300−1315.

[6] Fu ZH, Zerfos P, Luo HY, Lu SW, Zhang LX, Gerla M. The impact of multihop wireless channel on TCP throughput and loss. In: Proc. of the 22nd Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM 2003), Vol.3. IEEE Press, 2003. 1744−1753.

[7] Fu ZH, Greenstein B, Meng XQ, Lu SW. Design and implementation of a TCP-friendly transport protocol for ad hoc networks. In: Proc. of the 10th IEEE Int'I Conf. on Network Protocols (ICNP 2002). IEEE Press, 2002. 216−225. [doi: 10.1109/ICNP.2002.1181409]

[8] Kim D, Toh C, Choi Y. TCP-BuS: Improving TCP performance in wireless ad hoc networks. Journal of Communications and Networks, 2001,3(2):175−186.

[9] Nahm K, Helmy A, Kuo CCJ. Cross-Layer interaction of TCP and ad hoc routing protocols in multihop IEEE 802.11 networks. IEEE Trans. on Mobile Computing, 2008,7(4):458−469.

[10] Liu J, Guo W, Xiao BL, Huang F. Path holding probability based ad hoc on-demand routing protocol. Journal of Software, 2007, 18(3):693−701 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/693.htm [doi: 10.1360/jos180693]

[11] Tang W, Guo W. A path reliable routing protocol in mobile ad hoc networks. In: Proc. of the 2008 4th Int'l Conf. on Mobile Ad-hoc and Sensor Networks. IEEE Press, 2008. 203−207. [doi: 10.1109/MSN.2008.10]

[12] Jacobson V, Karels M. Congestion avoidance and control. ACM SIGCOMM Computer Communication Review, 1995,25(1): 157−187.

[13] Mathis M, Mahdavi J. TCP selective acknowledgement options. RFC 2018, 1996. http://tools.ietf.org/html/rfc2018

[14] The networks simulator ns-2. http://nsnam.isi.edu/nsnam/index.php/Main_Page

[15] Breslau L, Estrin D, Fall K, Floyd S, Heidemann J, Helmy A, Huang P, McCanne S, Varadhan K, Xu Y, Yu HB. Advances in network simulation. IEEE Computer, 2000,33(5):59−67. [doi: 10.1109/2.841785]

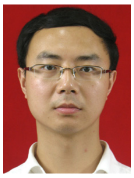
[16] Floyd S, Henderson T, Gurtov A. The NewReno modification to TCP's fast recovery algorithm. RFC 3782, 2004. http://tools.ietf.org/html/rfc3782

[17] Fu ZH, Meng XQ, Lu SW. A transport protocol for supporting multimedia streaming in mobile ad hoc networks. IEEE Journal on Selected Areas in Communications, 2004,21(10):1615−1626. [doi: 10.1109/JSAC.2003.816461]

[18] Chen K, Xue Y, Shah S, Nahrstedt K. Understanding bandwidth-delay product in mobile ad hoc networks. Elsevier Computer Communications, 2004,27(10):923−934.

附中文参考文献:

[10] 刘军,郭伟,肖百龙,黄飞.移动自组网基于路径维持概率的按需路由协议.软件学报,2007,18(3):693−701. http://www.jos.org.cn/1000-9825/18/693.htm [doi: 10.1360/jos180693]



孙杰(1982－),男,四川绵阳人,博士生,主要研究领域为自组织网络协议.

郭伟(1964－),男,教授,博士生导师,主要研究领域为移动通信网,信号与信息处理.

* *