

doi:10.3969/j.issn.1673-4785.2011.02.010

机会网络中一种提高TCP性能的跨层改进算法

左朝树¹,雷仕英²,李云²

(1. 保密通信重点实验室, 四川成都610041; 2. 重庆邮电大学无线信息网络研究中心, 重庆400065)

摘要:针对机会网络中TCP与传染路由结合时, 传染路由的“洪泛”特性和机会网络的间断特性会使得TCP性能变差的问题, 提出了一种基于传染路由协议和TCP/Reno协议的跨层改进算法——ACK-EPI. 该算法对连接开始阶段的慢开始门限值进行修改, 避免因网络错误进入拥塞避免阶段而导致拥塞窗口增长速度过于缓慢. 同时, 为了避免网络中已经成功交付但仍存储在网络中其他节点的数据包的副本继续在网络中存在并扩散, 造成网络资源浪费, 算法还利用ACK作为到达通告来删除这些冗余数据包. 通过深入的仿真分析和比较, 结果表明ACK-EPI算法能明显改善TCP性能.

关键词: 机会网络; TCP/Reno; 传染路由协议; ACK-EPI

中图分类号: TP393 **文献标识码:** A **文章编号:** 16734785(2011) 02-0152-08

A cross-layer design for improving TCP performance in opportunistic network

ZUO Chaoshu¹, LEI Shiyang², LI Yun²

(1. Science and Technology on Communication Security Laboratory, Chengdu 610041, China; 2. CWIN, Chongqing University of Posts and Telecommunications, Chongqing 400065, China)

Abstract: In opportunistic networks, the infectivity of epidemic routing and intermitent connectivity of network would degrade the performance of TCP when TCP works with epidemic routing. In order to improve the performance of TCP in opportunistic networks, a cross-layer design, named ACK-EPI, which is based on epidemic routing and TCP/Reno, is proposed. The algorithm renews the threshold of slow-star at the beginning of TCP connection to prevent the network from incorrectly turning into congestion avoidance phase so that the network can keep the regular increment speed of congestion window. At the same time, to avoid the packets that have already been delivered being forwarded again in the network, the design makes use of ACK to delete the redundant packets. The stimulation results show that the proposed algorithm can ultimately increase TCP performance in opportunistic networks.

Keywords: opportunistic networks; TCP/Reno; epidemic routing; ACK-EPI

机会网络!被定义为一种受限网络.它是一种不需要源节点和目标节点之间存在完整链路,利用节点移动带来的相遇机会实现通信的自组织网络.很多应用领域都无法建立结构化的全连通网络,导致传统的多跳自组织网络协议无法运行,而机会网络能够更好地满足这些应用需求.机会网络应用领域包括:野生动物追踪、手持设备入网、车载网络^[3]和偏远地区网络传输^[4]等.目前,人们将研究的重点放在机会网络的路由协议,并普遍采用“存

储-携带-转发”的路由模式来实现节点间通信,具体包括基于冗余的路由、基于效用的路由、基于冗余效用混合的路由和基于主动运动的路由.在基于冗余的路由协议中,主要有基于复制和基于编码的方式,如: Spray and wait^[1]、擦除编码路由(eraser-coding based routing)^[2]和传染路由协议(epidemic routing)^[5]等.

传染路由的基本思路是当2个节点相遇时,彼此交换各自缓存中的不同数据,使数据包最终被复制到网络中的所有节点.同网络层的路由协议相比,机会网络的传输层机制研究还很少.TCP^[6]是互联网(internet)中一种重要的传输层协议,被用于很多互联网应用中,如:文件传输协议(FTP)、电子邮件

收稿日期: 2010-09-15.

基金项目: 国家自然科学基金资助项目(61071118, 60702055); 重庆

市科委自然科学基金资助项目(CSTC2009BB2279).

通信作者: 左朝树.E-mail: cs_zuo@163.com

(email)和网络服务(web)等。目前,尽管有较多文献研究了移动自组织网络(MANET: Mobile Ad hoc NETworks)和传感器网络中 TCP 的性能并提出了改进方法^[12-15],但这些改进主要是针对无线网络的信道误码率高、节点的缓存及能量有限等方面提出的,而机会网络中传输层的研究相对较少^[16-18],尤其是针对 TCP 性能的研究。

在文献[19]中,作者研究了机会网络中 TCP 与传染路由结合时的性能。文献指出,在机会网络中,当节点移动速度比较大时,传染路由比 DSR 路由更有效,能与 TCP 更好结合。但是,在 TCP 连接开始阶段,由于节点的移动性,会导致网络错误地进入拥塞避免阶段而使得拥塞窗口增长过慢,减低数据发送速度;同时,在机会网络中使用传染路由也会因为其产生的重复的数据包而降低 TCP 的性能。针对上述问题,提出了一种基于传染路由和 TCP/Reno 协议的跨层改进算法——ACK-EPI,并对该改进算法进行了仿真,结果表明,该改进算法可以有效地提高网络性能。

1 跨层改进算法:ACK-EPI

利用 TCP 协议中的 ACK 来设计数据包的到达通告。当一个数据包到达目的节点以后,存储在网络中其他节点中的该数据包的复本还会在网络中继续存在并扩散,对于网络资源稀缺的机会网络来说是一种浪费。同时,由于节点的移动性,在 TCP 连接建立的初始时刻经常会出现数据发送超时,导致网络错误地进入拥塞避免阶段而使得拥塞窗口增长过于缓慢,使得网络性能偏低。本文首先在 Epidemic 的基础上利用 TCP 接收端发送的 ACK,为其加入到到达通告机制来解决已交付的复本浪费资源的问题,然后将初始时刻错误的拥塞避免过程修复为慢开始过程以解决拥塞窗口增长缓慢的问题,这样同时也提高了协议的工作性能。在下文中,将加入了到达通告和修复了拥塞窗口的跨层改进算法称为 ACK-EPI。

1.1 ACK-EPI 基本思想

1.1.1 ACK 到达通告

在 ACK-EPI 协议中,为了尽量减少路由开销,每一个无线节点除了保存 Epidemic 分组的复本和 SV 数组^[20]外,还保存一个 delivered 列表,该列表以比特数组的方式存储,其比特值 1 或 0 分别表示此数据是否已经成功到达目的节点。

有 2 种情况表明数据包已经成功交付,并分别处理:

1) 目的节点:目的节点收到数据包后将数据包在 delivered 列表中对应该比特位置 1;

2) 中间节点:利用 TCP 接收端发送的 ACK 的序

列号将中间节点缓存中那些已经到达了目的节点的数据包删除。即将收到的 ACK 包的序列号与缓存中的数据包的序列号比较,将缓存中序号比收到的 ACK 序号小的 TCP 包(表明这些 TCP 数据包已经到达了目的节点)在 delivered 列表中对应该比特位置 1。

delivered 列表被封装在请求(QUERY)分组和更新(UPDATE)分组中来传播,节点收到 delivered 列表后删除缓存中已交付的数据包并更新自己的 delivered 列表,在计算自己需要的数据 iwant 时参考 delivered 列表,不再请求已交付数据包。随着 delivered 列表在全网的传播,网络中已交付的数据包将不再继续被复制传播而且逐渐被删除。

1.1.2 修复拥塞窗口

在连接建立的初始阶段,由于节点的移动性导致发送的数据长时间不能到达目的节点,网络会出现超时。此时,慢开始门限值 ssthresh 被更新后的值为 2。当数据发送端在首次收到 ACK 确认后,cwnd 的值会增长为 2,而当 cwnd > ssthresh 时就会停止使用慢开始算法而改用拥塞避免算法,这会使得拥塞窗口 cwnd 的值增长过于缓慢(如表 1),从而降低了数据发送速率,导致网络吞吐量偏低。

表 1 cwnd 的变化
Table 1 Changes of cwnd

时间/s	收到 确认	发送 的数据 包	窗口 数据 (TCP 序列号)	cwnd 值 (MSS)	ssthresh 值 (MSS)	网络 特殊 状态	拥塞 控制 算法
82.557	x	tcp0	0	1	ssthresh_	x	慢 开始
88.557	x	tcp0	0	1	2	超时	慢 开始
100.557	x	tcp0	0	1	2	超时	慢 开始
113.450	ack0	tcp1 tcp2	1,2	2	2	x	拥塞 避免
113.687	ack1	tcp3	2,3	2	2	x	拥塞 避免
113.704	ack2	tcp4	3,4	2	2	x	拥塞 避免
113.921	ack3	tcp5 tcp6	4,5,6	3	2	x	拥塞 避免
113.966	ack4	tcp7	5,6,7	3	2	x	拥塞 避免
114.158	ack5	tcp8	6,7,8	3	2	x	拥塞 避免
114.176	ack6	tcp9 tcp10	7,8,9,10	4	2	x	拥塞 避免

但这种在数据发送的初始阶段的超时重传并非网络真正出现拥塞,故不必错误地进入拥塞避免阶段,因此需要调整 $ssthresh$ 的值以便使网络恢复到慢开始状态。

1.2 ACK-EPI 算法描述

1.2.1 ACK 到达通告算法描述

对于 QUERY 和 UPDATE 分组^[20],在分组头中也各自加入了与摘要向量数组 $SV[SUMMSIZE]$ 相似的新数组 $qryack[SUMMSIZE]$ 和 $upack[SUMMSIZE]$ 、 $qryack[SUMMSIZE]$ 和 $upack[SUMMSIZE]$ 用来传递 delivered 列表。ACK-EPI 的算法有如下 4 个阶段(如图 1)。

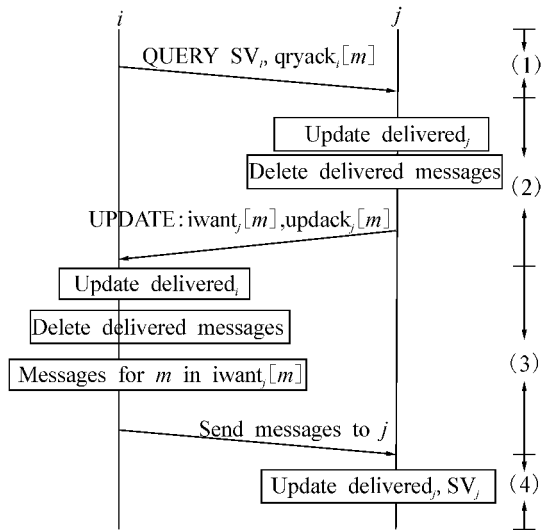


图 1 ACK-EPI 算法原理

Fig. 1 Block diagram of ACK-EPI algorithm

1) 节点 i 向节点 j 发送 QUERY 分组。

节点 i 和 j 相遇后,节点 i 向 j 发送 QUERY 分组, QUERY 分组中包括有:自己存储报文的摘要向量 $SV_i[m]$ 以及已交付数据包列表 $delivered_i[m]$ 的副本 $qryack_i[m]$ 。

2) 节点 j 接收 QUERY 分组并发送 UPDATE 分组。

节点 j 接收到 $SV_i[m]$ 和 $qryack_i[m]$ 后, j 将 $qryack_i[m]$ 和本节点的 $delivered_j[m]$ 按式(1)进行按位或,将消息 m 在 $delivered_j[m]$ 中的对应位按位置 1,以更新本地节点中保存的 $delivered_j[m]$ 。

$$delivered_j[m] = delivered_j[m] \parallel qryack_i[m].$$

(1)

更新完 $delivered_j[m]$ 后,节点 j 根据最新的 $delivered_j[m]$ 列表将 $delivered_j[m]$ 中相应位为 1 的数据包从节点缓存列表 $copytable$ 和分组中的缓存列

表 $copylist$ 中删除,节点 j 按式(2)计算其所需的数据包请求消息。节点 j 所需要的数据 $iwant_j[m]$ 为其自己没有而节点 i 有,并且是还没有交付到目的节点的也不是 j 节点先前请求过的数据。

$$iwant_j[m] = SV_i[m] \&$$

$$ihave_i[m] \& prevask_j[m] \& delivered_j[m]. \quad (2)$$

节点 j 将 $delivered_j[m]$ 复制到 UPDATE 分组中的 $upack_j[m]$,再将封装 $iwant_j[m]$ 和 $upack_j[m]$ 的 UPDATE 分组发送给节点 i 。

3) 节点 i 接收 UPDATE 分组,发送 EPI 分组给节点 j 。

节点 i 接收到 UPDATE 分组后,节点 i 接收到表示 $delivered_j[m]$ 的 $upack_j[m]$ 后,先按式(3)更新本地节点的 $delivered_i[m]$ 列表。

$$delivered_i[m] = delivered_i[m] \parallel upack_j[m].$$

(3)

更新完 $delivered_i[m]$ 后,节点 i 根据最新的 $delivered_i[m]$ 列表将 $delivered_i[m]$ 中相应位为 1 的数据包从节点缓存列表 $copytable$ 和分组中的缓存列表 $copylist$ 中删除。节点 i 将 $iwant_j[m]$ 中相应位为 1 的数据包 m 逐条封装在 EPI 分组中发送给 j 。

4) 节点 j 接收 EPI 分组。

节点 j 收到数据包后,依次作如下处理:

如果该 EPI 分组的目的是节点 j ,说明 EPI 分组正确地由源节点传送到目的节点,则将该数据包在 $SV_j[m]$ 和 $delivered_j[m]$ 中的对应位置 1,并将数据传递给上层协议。

如果该 EPI 分组的目的是节点不是 j ,但是需要 j 进行存储转发,则将该数据包在 $SV_j[m]$ 中的对应位置 1,等待该分组转发。但是在转发之前要先判断此包是否已经交付给了目的节点但仍存储在中间节点的消息副本,利用 TCP 到达目的节点后,接收端发送的在网络中传输的 ACK 确认消息来判断,具体方法如下:建立一个新的二维数组 $check[SUMMSIZE + 3][3]$ 来存储数据包的序列号和数据包类型,设定 Max 为收到的 ACK 的最大序列号,如果来的这个包是新的数据包则提取此包的 ID 号、序列号和数据包类型,并执行以下步骤:

1) 如果此包是 TCP 包并且其序列号不大于 Max ,表明此 TCP 包已经到达了目的节点(因为有更大序列号的 ACK 已经在网络中传播),则直接将该数据包在 $delivered_j[m]$ 中的对应位置 1。

2)如果此包的序列号大于 Max 并且是 TCP 包,则直接将此包的 ID 号、序列号和数据包类型对应地存储在数组 check 中。

3)如果此包的序列号大于 Max 并且是 ACK 包,则将此包的 ID 号、序列号和数据包类型对应地存储在数组 check 中,并更新 Max,即将此包的序列号作为新的 Max. 然后从上次遍历过的下一处开始遍历 check 数组中存储的序列号,依次找出比 Max 小的所有 TCP 数据包(表明这些数据包已经到达了目的节点,不必再在网络中传输),并将这些数据包在 delivered_j 列表中的对应位置 1。

在算法的第 3 阶段中,节点将 delivered_j[m] 中相应位为 1 的数据包从节点缓存列表 copytable 和分组中的缓存列表 copyleft 中删除,这样就将网络中已经到达了目的节点的数据包删除。同时,在第 4 阶段,将 ACK 作为到达通告机制跨层加入到传染路由中,根据 ACK 序列号将已经成功到达了目的节点但仍存储在中间节点的数据包或者其复本删除,达到节约网络资源,提高网络性能的目的。

1.2.2 修复拥塞窗口算法描述

伪代码如下。

```
当一个包到达发送端时
if(此时sssthresh的值小于3)
{if(它是ACK包&&首次收到此包&&序列号为1)
    将sssthresh的值恢复为初始值
}
```

图 2 修复 CWND 算法

Fig.2 Algorithm of repairing CWND

在连接建立的初始阶段,如果由于节点的移动性导致数据发送的不断超时重传,表明并非网络真正出现拥塞,故不必进入拥塞避免阶段。在数据发送端首次收到 ACK 确认时,就将超时重传后减小的慢开始门限值 sssthresh 恢复为初始值,这样可以使得网络不再错误地处于拥塞避免阶段,而是继续调用慢开始算法,让拥塞窗口正常增长。

2 仿真结果与分析

在仿真时,采用 NS-2 作为仿真模拟器,网络中的 30 个节点依据随机节点移动模型进行移动,仿真的物理层和数据链路层采用 IEEE 802.11,在节点 2 和节点 3 之间建立一条 TCP 连接,其上承载 FTP 流量,具体参数设置见表 2。

表 2 仿真参数

Table 2 Simulation parameters

参数	值
信道速率	1Mbit/s
TCP 包的大小	512bytes
窗口尺寸	500 m × 500 m
仿真时间	1000 s
缓冲大小	100 packets
传输距离	50 m

在仿真中,主要对以下性能参数进行评估。

1)吞吐量(throughput):从目的节点正确接收第一个数据包开始到接收最后一个数据包为止,每秒成功接收的数据量,统计公式如式(4),统计单位为 kb/s。

$$\text{吞吐量} = \frac{\text{数据包数量}}{(t_{\text{last}} - t_1) \times 1000} \times \text{包的长度} \times 8. \quad (4)$$

2)数据交付率(delivery ratio):成功传递到目的节点的数据占所有发送数据的比率,统计公式如式(5)。

$$\text{数据交付率} = \frac{\text{收到包的数量}}{\text{所有发送包数量}}. \quad (5)$$

3)端到端的平均交付时延(average delay):数据包从源节点成功传递到目的节点的平均时间,统计公式如式(6),统计单位为 s。

$$\text{平均交付时延} = \frac{\sum_{i=1}^n t_i}{\text{所有收到包的数量}} \quad (6)$$

4)端到端平均跳数(average hop):所有从源节点成功传送到目的节点数据包所经跳数的平均值,如式(7)。

$$\text{端到端平均跳数} = \frac{\sum_{i=1}^n h_i}{\text{所有收到包的数量}}. \quad (7)$$

5)数据转发率(epifwd ratio):在整个仿真过程中,所有被转发数据包占所有发送数据包的比率见式(8)。

$$\text{数据转发率} = \frac{\text{转发数据包数量}}{\text{所有发送包数量}}. \quad (8)$$

为了更好地体现修改后的算法,对以下 2 个场景进行仿真分析。

场景 1:其他参数保持不变时,在 30 个节点组成的机会网络中,仿真分析节点移动速度的变化对性能参数的影响情况。

场景 2:其他参数保持不变时,保持节点移动速度为 15m/s,仿真分析网络中节点数的变化对性能参数的影响情况。

首先,从仿真统计文件中得到的修改算法后的 cwnd 变化情况,如表 3。

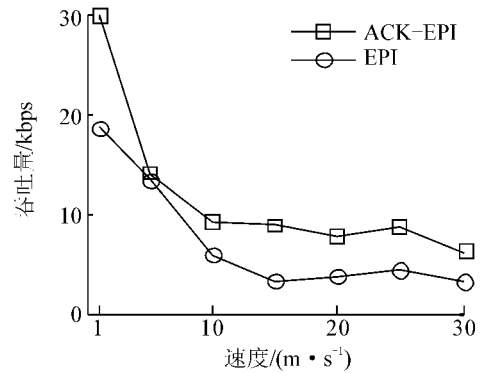
表 3 修改后 cwnd 的变化

Table 3 Changes of cwnd with improved algorithm

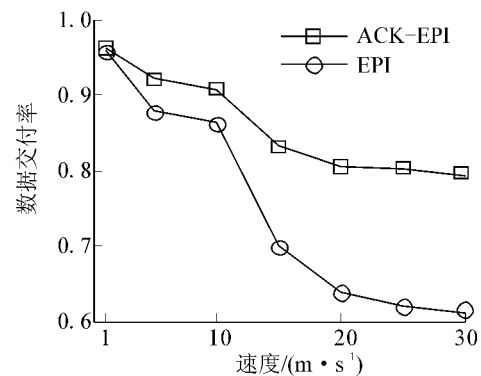
时刻/s	收到 确认	发送 的数据 包	窗口 中数 据包 (TCP 序 列号)	cwnd 值 (MSS)	Ssthresh 值	网络 特殊 状态	拥塞 控制算法
82.557	x	tcp0	0	1	ssthresh_	x	慢 开始
88.557	x	tcp0	0	1	2	超时	慢 开始
100.557	x	tcp0	0	1	2	超时	慢 开始
119.212	ack0	tcp1	1,2	2	恢复为 ssthresh_	x	慢 开始
119.429	ack1	tcp2	2,3,4	3	ssthresh_	x	慢 开始
119.441	ack2	tcp3	3,4,5,6	4	ssthresh_	x	慢 开始
119.645	ack3	tcp4	4,5,6,7,8	5	ssthresh_	x	慢 开始
119.932	ack14	tcp5	15~30	16	ssthresh_	x	慢 开始
120.045	ack14	tcp6	15~30	16	ssthresh_	x	慢 开始
120.055	ack14	tcp7	15~30	16	ssthresh_	x	慢 开始
120.064	ack14	tcp8	15~30	11	8	重传	快 恢复
120.157	ack15	tcp9	16~30	8	8	x	拥塞 避免
—	—	—	—	—	—	—	—
120.251	ack24	tcp10	25~33	9	8	x	拥塞 避免
—	—	—	—	—	—	—	—

从表中可看到,在初始连接时,如果网络出现超时,则在发送端首次收到确认时将慢启动门限值恢复到初始值,因为此刻网络并未出现拥塞。在 119.212 s 时,首次收到确认,将 ssthresh 恢复至初始值 ssthresh_,即为 65 535 字节。随后,网络不会错误地调用拥塞避免算法,这样可以使得 cwnd 较快地增长,充分利用网络资源。

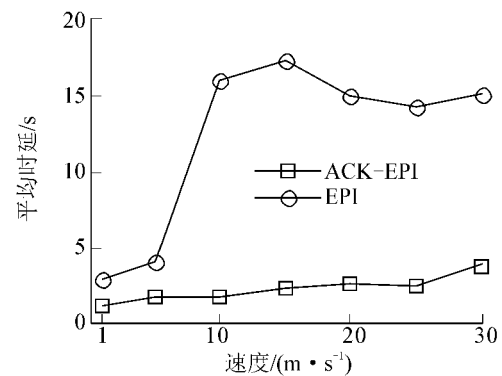
其次,根据仿真统计文件得到修改前后吞吐量、数据交付率、平均时延、平均跳数和数据转发率的对比图,如图 3 和图 4。从图可以看出,改进后的算法性能优于修改前的。



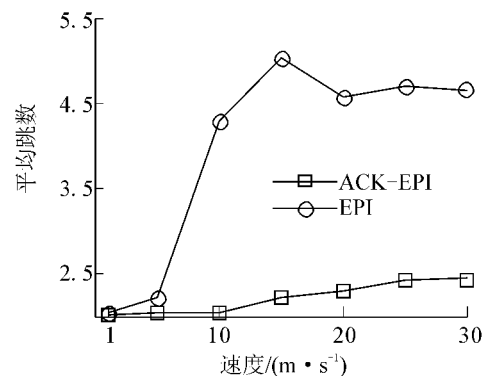
(a) 吞吐量随速度变化



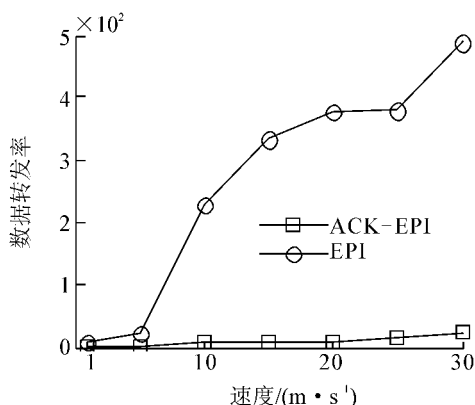
(b) 数据交付率随速度变化



(c) 平均时延随速度变化



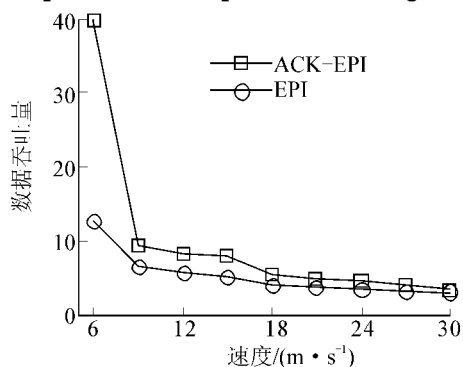
(d) 平均跳数随速度变化



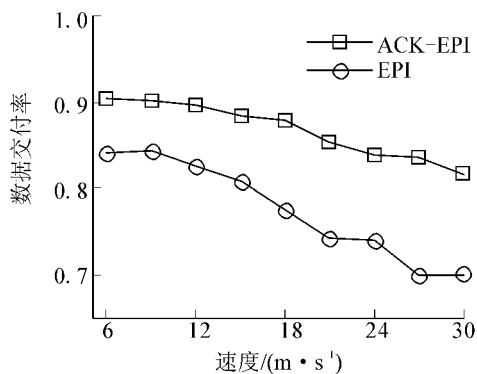
(e) 数据转发率随速度变化

图3 修改前后各个参数随速度变化的比较

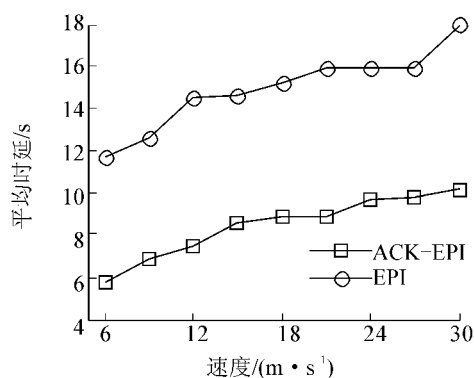
Fig. 3 The before and after the modified algorithm performance comparison with changes of speed



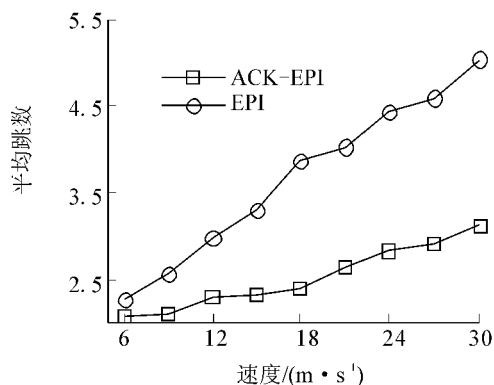
(a) 吞吐量随移动节点数变化



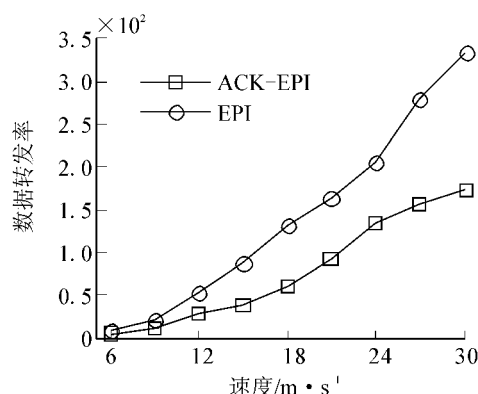
(b) 数据交付率随移动节点数变化



(c) 平均时延随移动节点数变化



(d) 平均跳数随移动节点数变化



(e) 数据转发率随移动速度变化

图4 修改前后各个参数随移动节点数变化的比较

Fig. 4 The before and after the modified algorithm performance comparison with changes of number of node

从图3可以看到,随着节点移动速度的增加,网络吞吐量和数据交付率不断降低,平均时延、平均跳数和数据转发率逐渐增大,其原因是由于节点移动速度增加,网络连接的间断性增强,会导致超时和重传增加,从而降低吞吐量和数据交付率,增加平均时延、平均跳数和数据转发率.但使用ACK-EPI算法得到的各个网络性能都优于使用修改前算法得到的性能.首先,通过改变初始连接阶段的慢启动门限值,可以避免网络不再错误地进入拥塞避免阶段,从而提高发送端发送数据的速度;其次,通过ACK的到达通告,删除网络中已经成功交付的数据包以节约网络资源.这些都在很大程度上提高了网络性能.

图4是网络中移动节点数对网络性能的影响,网络中移动节点越多,占用网络资源也越多,故吞吐量和交付率是呈现下降趋势,而平均时延、平均跳数和数据转发率则呈上升趋势.同样,改进后的算法比改进前的算法更有效.

3 结束语

在 TCP 连接开始阶段,由于节点的移动性,会导致网络错误地进入拥塞避免阶段而使得拥塞窗口增长过慢,减低数据发送速度;同时,在机会网络中使用传染路由也会因为其产生的重复的数据包而降低 TCP 的性能.本文针对该问题,提出了一种基于传染路由和 TCP 的跨层改进算法——ACK-EPI.该算法对连接开始阶段的慢开始门限值进行修改,以避免因网络错误地进入拥塞避免阶段而导致的拥塞窗口增长速度过于缓慢.同时,为了避免网络中已经成功交付但仍存储在网络中其他节点的数据包的副本继续在网络中存在并扩散,造成网络资源浪费,算法还利用 ACK 作为到达通告来删除这些冗余数据包.通过深入的仿真分析和比较,表明 ACK-EPI 算法能明显改善 TCP 性能.

参考文献:

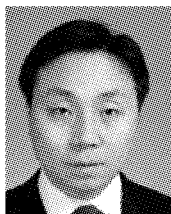
- [1] PELUSI L, PASSARELLA A, CONTI M. Opportunistic networking: data forwarding in disconnected mobile ad hoc network[J]. IEEE Communications Magazine, 2006, 44(11): 134-141.
- [2] 熊永平,孙利民,牛建伟,刘燕. 机会网络[J]. 软件学报, 2009, 20(1): 124-137.
XIONG Yongping, SUN Limin, NIU Jianwei, LIU Yan. Opportunistic networks[J]. Journal of Software, 2009, 20(1): 124-137.
- [3] KEVIN FALL K. A delay-tolerant network architecture for challenged internets[C]//Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. New York, USA: ACM Press, 2003: 27-34.
- [4] JUANG P, OKI H. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet[C]//Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems. New York, USA: ACM Press, 2002: 96-107.
- [5] HUN B, BYCHKOVSKY V. CarTel: a distributed mobile sensor computing system[C]//Proceedings of the 4th International Conference on Embedded Networked Sensor Systems. New York, USA: ACM Press, 2006: 125-138.
- [6] PENTLAND A, FLETCHER R, HASSON A. DakNet: rethinking connectivity in developing nations[J]. IEEE Computer, 2004, 37(1): 78-83.
- [7] SPYROPOULOS T, PSOUNIS K, RAGHAVENDRA C S. Spray and wait: an efficient routing scheme for intermittently connected mobile networks[C]//Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking. New York, USA: ACM Press, 2005: 252-259.
- [8] WANG Y, JAIN S, MARTONOSI M, FALL K. Erasure-coding based routing for opportunistic networks[C]//Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking. New York: ACM Press, 2005: 229-236.
- [9] VAHDAT A, BECKER D. Epidemic routing for partially connected ad hoc networks[R]. Duke University: Technical Report CS-200006, 2000.
- [10] RAMANATHAN R, HANSEN R, BASU P, HAIN R R, KRISHNAN R. Prioritized epidemic routing for opportunistic networks[C]//Proceedings of the 1st International MobiSys Workshop on Mobile Opportunistic Networking. New York, USA: ACM Press, 2007: 62-66.
- [11] ALLMAN M, PAXSON V, STEVENS W. RFC 2581, TCP Congestion Control[S]. New York: IETF, 1999.
- [12] TRIANTAFYLIDOU D, AGHA K A. Adaptive setting of TCP's maximum window in ad hoc multihop networks with a single flow[C]//Proceedings of IEEE Wireless Communications and Networking Conference. Budapest, Hungary: WCNC, 2009: 1-6.
- [13] XIAO Hannan, ZHANG Ying, JAMES M, et al. Modeling and analysis of TCP performance in wireless multihop networks[J]. Wireless Sensor Network, 2010, 2(7): 493-503.
- [14] AHMED A, PATRICK M, DAVID R. Improving distributed TCP caching for wireless sensor networks[C]//Proceedings of the 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop. Juan Les Pins, France: Med-Hoc-Net, 2010: 1-6.
- [15] YOON W, VAIDYA N. Routing exploiting multiple heterogeneous wireless interfaces: a TCP performance study[J]. Computer Communications, 2010, 33(1): 23-34.
- [16] RAMADAS M, BURLEIGH S, FARRELL S. Licklider transmission protocol-specification [EB/OL]. [2010-11-18]. <http://wenku.baidu.com/view/7a86dc1a6bd97f192279e994.html>, Internet-draft, 2007.
- [17] FARRELL S, CAHILL V. Ltp-t: a generic delay tolerant transport protocol[R]. University of Dublin, Technical reports: TCD-CS-2005-69, 2007.
- [18] FARRELL S, RAMADAS M, BURLEIGH S. Licklider transmission protocol-security extensions [EB/OL]. [2010-08-22]. <http://wenku.baidu.com/view/c394efd87f1922791688e894.html>, Internet-draft, 2007.
- [19] LI Yun, LEI Shiyang, YOU Xiaohu. Performance of TCP in intermittently connected wireless networks: analysis and improvement[C]//Proceedings of the 2010 IEEE Global

Telecommunications Conference. Miami, USA: GLOBE-COM, 2010: 1.

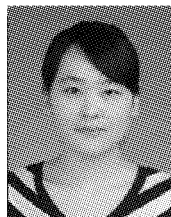
[20]何靖桐. 稀疏 AdHoc 网络中路由算法的研究[D]. 成都:成都电子科技大学, 2006: 17-22.

HE Jingtong. Research for routing in intermittent ad-hoc-networks[D]. Chengdu: University of Electronic Science and Technology, 2006: 17-22.

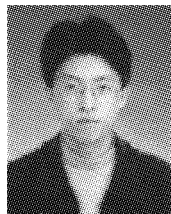
作者简介:



左朝树,男,1972年生,高级工程师,硕士生导师,博士,主要研究方向为通信网络及信息安全,发表学术论文20余篇,其中被EI检索10余篇。



雷仕英,女,1985年生,硕士研究生,主要研究方向为宽带无线接入网技术。



李云,男,1974年生,教授,博士生导师,主要研究方向为无线移动通信,发表学术论文150余篇,其中被SCI、EI检索80余篇。

2011 中国智能(产业)博览会

China International Intelligent Industries Expo 2011

2011. 9. 16—18 北京全国农业展览馆

为贯彻落实温家宝总理提出“感知中国”的科技纲领,推进国家“十二五”建设的新举措,激发广大科技工作者投身建设创新型国家的积极性和创造热情,进一步推动智能科学技术担当起信息化向智能化高端发展进程中的历史重任,欣逢纪念中国人工智能学会成立30周年和世界人工智能科学诞生55周年,经中国科学技术协会批准,得到国家科学技术部、工业和信息化部、教育部、住房和城乡建设部、中国科学院、中国工程院、国家自然科学基金委员会悉心指导,由中国人工智能学会主办,中国电子学会、中国计算机学会、中国通信学会、中国自动化学会、中国系统仿真学会、中国仪器仪表学会、中国图象图形学学会、中国中文信息学会、中国计算机用户协会支持合作,北京东方炬峰展览有限公司承办的“首届中国智能博览会”(简称智博会)将于2011年9月16—18日在北京全国农业展览馆盛大开幕。届时,本届“智博会”组委会将诚邀科技、教育、农业、机械、轻工、通信、电力、交通、建筑、金融、卫生、安全、国防等专业领域的政府官员、专家学者、国际友人和企业精英参会,共同见证与分享“智能体验·智慧生活”的核心主题。

本届智博会将秉承和弘扬理解、交流、创新与合作的智博理念,向社会各界传达智能科技和智慧生活的最新信息,展现丰富多彩的智能新型能源、创新成果、新兴产业以及智慧生活体验,搭建智能产业与科技民生的桥梁,激发广大热爱智能科学技术的有志之士相互交流,以及实现更美好生活的强烈愿望,深入探讨智能化进程中的多元时代现象,广泛推进学术、科技、工程、产业以及资本合作的多边共赢之路,关注全球重大问题给国际社会和科技民生带来的挑战和影响,共同展望物联网、云计算、智能电网、智能交通、智能通信、智能计算机、智能机器人、智能建筑等智能化领域的美好未来。

网址: www.ciiie.com.cn.