第 35 卷 第 6 期 计 算 机 工 程 2009 年 3 月
Vol.35 No.6 Computer Engineering March 2009
·网络与通信· 文章编号：1000—3428(2009)06—0119—03 文献标识码：A 中图分类号：TP274.2

基于 IEEE802.16 Mesh 网络的拥塞控制方法

魏登宇[1]，李 云[1]，王海宝[2]，刘占军[1]
(1. 重庆邮电大学无线信息网络研究中心，重庆 400065；2. 重庆三峡学院教务处，重庆 404000)

摘 要：随着网络服务类型的增加和用户对带宽需求的增长，宽带无线接入技术受到关注。该文介绍 IEEE802.16 的 Mesh 结构，阐述在 Mesh 结构下的拥塞控制，提出一种动态改变传输等待时间指针和动态分配微时隙的拥塞控制方法。仿真结果证明，该方法可有效控制拥塞，增加吞吐量，降低丢包率和端到端时延。

关键词：IEEE802.16 技术；网状网；拥塞控制；指针；微时隙

Congestion Control Method Based on IEEE 802.16 Mesh Networks

WEI Deng-yu[1], LI Yun[1], WANG Hai-bao[2], LIU Zhan-jun[1]
(1. Research Center for Wireless Information Networks, Chongqing University of Posts & Telecommunications, Chongqing 400065;
2. Academic Administration Office, Chongqing Three Gorges University, Chongqing 404000)

【Abstract】With the increase of network service types and users broadband demand, Broadband Wireless Access(BWA) has become the focus of people's attention. This paper introduces the Mesh mode of IEEE802.16, discusses congestion control in Mesh mode, and presents a new method through dynamically adjusting the exponent of transmission holdoff time and allocating minislot to control congestion. Simulation results show that the congestion can be controlled effectively, throughput is increased, packet loss rate and end-to-end delay are decreased.

【Key words】IEEE802.16 technology; Mesh networks; congestion control; exponent; minislot

1 概述

宽带无线接入已成为通信业主流。目前已有很多国家向用户提供高速无线宽带接入，如欧洲的宽带无线接入网络、美国的本地多点分布式服务、多点多信道分布式系统和 IEEE802 工作组提出的宽带无线接入技术 IEEE802.16[1]。

IEEE802.16 定义了无线城域网的空中接口规范，开发了 2 GHz~66 GHz 频带的无线接入物理层和媒质接入控制层，以解决最后 1 km 的无线宽带接入。标准中包括 2 种网络结构：点对多点(PMP)和网状网(Mesh)。在 PMP 中，一个基站和多个用户站组成类似蜂窝网的网络；在 Mesh 中，所有节点以自组织方式构成，彼此对等，每个节点都像路由器一样转发数据给邻居节点。由于基于 PMP 结构的研究已趋于成熟，因此，本文主要研究 Mesh 结构。

目前，通用的 2 种无线 Mesh 网络分别基于 IEEE802.11 技术和 IEEE802.16 技术，前者已有示范网络和许多研究成果，而后者的研究还处于初步阶段，研究成果较少[2-4]，且主要集中在调制解调、信道编码、信道分配、移动性管理等方面，还没有相关研究讨论拥塞控制问题。因此，本文提出一种基于 IEEE802.16 Mesh 网络的拥塞控制方法。节点在发送数据请求前检测拥塞并动态调整传输等待时间指针。其上、下游节点在收到请求后也相应调整传输等待时间指针。该方法以逐跳调整的方式来缓解拥塞，提高吞吐量和缩短时延。同时本文还给出平均丢包率、吞吐量和端到端时延等的仿真验证和分析。

2 IEEE802.16 Mesh 模式的调度

2.1 Mesh 帧结构

Mesh 是一种新型的网络拓扑结构，它采用时分复用方式，没有严格的上下行之分。Mesh 帧采用时分复用的帧结构，由控制子帧和数据子帧组成，如图 1 所示。

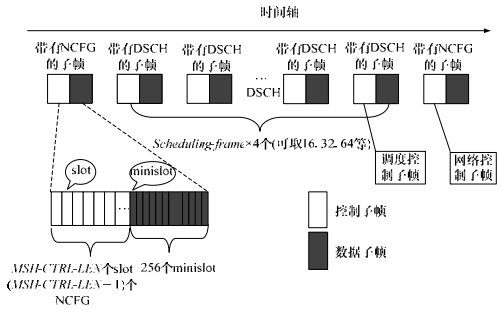

图 1 Mesh 帧结构

控制子帧分为若干个时隙单元 slot。控制子帧长度固定，等于 *MSH-CTRL-LEN*×7 个 OFDM 符号，其中 *MSH-CTRL-LEN* 表示 slot 的个数，由网络配置消息 MSH-NCFG 中的网络描述符信息单元 Network DescriptorIE 给出。

控制子帧分为网络控制子帧和调度控制子帧。网络控制子帧用于网络控制，发送网络接入消息 MSH-NENT 和网络配

基金项目：新世纪优秀人才支持计划基金资助项目(教技司[2006]348 号)；重庆市教委科技基金资助项目(KJ070521)；华为高校科学基金资助项目(YJCB20070615)；华为合作基金资助项目(YBWL2008033)

作者简介：魏登宇(1984－)，女，硕士研究生，主研方向：宽带无线接入；李 云，教授、博士；王海宝，教授、硕士；刘占军，讲师、硕士

收稿日期：2008-07-10 E-mail：haochishu1984@hotmail.com

置消息 MSH-NCFG，主要功能是创建和保持不同系统间的一致。在网络控制子帧中，只有第 1 个 slot 用于传送 MSH-NENT，其余的 MSH-CTRL-LEN-1 个 slot 用于传送 MSH-NCFG。网络控制子帧周期性发送，周期为 scheduling-frame×4+1，其中，scheduling-frame 由 MSH-NCFG 消息中的 Network DescriptorIE 给出。

调度控制子帧发送集中式调度消息 MSH-CSCH 和分布式调度消息 MSH-DSCH，用于对链路上资源数量的分配，完成系统间数据发送的协调调度。

数据子帧分为 256 个微时隙，用于传输数据。

2.2 控制子帧的调度

控制子帧调度的关键在于 MSH-NCFG 和 MSH-DSCH 的调度。它们具有相同的调度规则，本文以 MSH-DSCH 为例。在 Mesh 分布式调度中，MSH-DSCH 的调度思想在于确定每个节点传输消息的时间，形成一张邻居调度表广播给所有相邻节点，节点根据收到的调度表信息竞争下一次传输 MSH-DSCH 的时间 Next-Xmt-Time(t_{NX})更新调度表，等待传输时间将表广播出去。节点的 t_{NX} 的取值范围和传输等待时间 Xmt-Holdoff-Time(t_{NH})分别如式(1)和式(2)所示。

$$2^{exp}\cdot t_a < t_{NX} \le 2^{exp}\cdot(t_a+1) \tag{1}$$

$$t_{NH} = 2^{(exp+4)} \tag{2}$$

其中，*exp* 表示传输等待指针 *Xmt-Holdoff-Exponent*；t_a 表示下一次传输时间系数 *Next-Xmt-Mx*。节点的 t_{NX} 通过 IEEE802.16 中给出的函数 *MeshElection*()来实现。

2.3 数据子帧的调度

在 Mesh 分布式调度中，IEEE802.16 运用 3 次握手来建立发送数据前的连接，如图 2 所示，以实现数据子帧中微时隙的调度和分配。

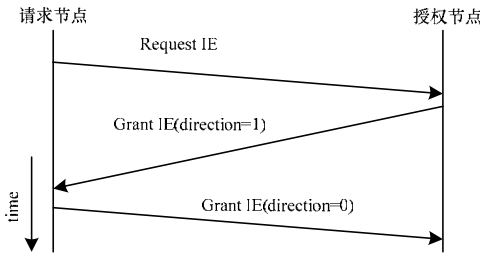

图 2　3 次握手过程

(1)请求：待发送数据的请求节点使用 MSH-DSCH 的请求信元 RequestIE 说明链路标识 *LinkID*、发送数据大小 *DemandLevel* 和发送数据持续帧个数 *DemandPresistence* 等。

(2)回复：授权节点根据请求，寻找适合的微时隙，向请求节点发出确认信元(GrantIE, *direction*=1)，说明授权的微时隙的位置。

(3)确认：请求节点收到 GrantIE 后，拷贝授权信息，发送 GrantIE(*direction*=0 表明是发给授权方)作为确认信息完成 3 次握手。

3 本文提出的拥塞控制方法

在上述基础上，本文提出一种基于 IEEE802.16 分布式的 Mesh 拥塞控制方法，通过自适应调整 IEEE802.16MAC 层的资源分配来缓解网络拥塞，实现对网络性能的优化。该方法包括：(1)本地节点在发送数据前的拥塞检测和拥塞处理；(2)上游节点的拥塞处理；(3)下游节点的拥塞处理。

3.1 本地拥塞检测和拥塞处理

本地节点发送数据请求前首先综合运用网络层队列的占用率 *BU*、调度子帧中控制时隙的占用率 *CSU* 和数据子帧中微时隙的占用率 *DSU* 来联合检测拥塞，用 *CSt* 表示检测到的拥塞状态变量，取值为 0, 1, 2, 3，其中 0 为初始值、1 表示拥塞；2 表示轻负荷；3 表示网络良好。同时，定义 $BU_{\max}$，$BU_{\min}$ 为 *BU* 的上、下限，$CSU_{\max}$、$CSU_{\min}$ 为 *CSU* 的上、下限，$DSU_{\max}$、$DSU_{\min}$ 为 *DSU* 的上、下限。

若 *BU*，*CSU*，*DSU* 中任意一个大于上限，即 $BU > BU_{\max} \,\|\, CSU > CSU_{\max} \,\|\, DSU > DSU_{\max}$，说明网络即将发生拥塞，则判定本地节点拥塞，设置 *CSt*=1。

若 *BU*, *CSU*, *DSU* 均小于下限，即 $BU < BU_{\min}$ &&$CSU < CSU_{\min}$ &&$DSU < DSU_{\min}$，说明网络轻负荷，则判定本地节点轻负荷，设置 $CSt = 2$。否则，说明网络状态良好，判定本地节点状态良好，设置 *CSt*=3。

为计算 *BU*，定义 BU_{temp} 为队列的瞬时占用率，每隔一定的时间 *t*，对目前队列的占用率进行统计，并将统计值赋值给 BU_{temp}，同时按如下更新：

$$BU_{new} = \alpha\cdot \mathrm{BU}_{\mathrm{old}} + (1-\alpha)\cdot BU_{temp} \tag{3}$$

其中，α 为队列占用率的平滑因子，BU_{old} 为上次计算得到的队列平均占用率；BU_{new} 为当前的队列平均占用率。

同理，对 *CSU* 和 *DSU* 按式(4)和式(5)进行更新。

$$CSU_{new} = \beta\cdot CSU_{old} + (1-\beta)\cdot CSU_{temp} \tag{4}$$

$$DSU_{new} = \gamma\cdot DSU_{old} + (1-\gamma)\cdot DSU_{temp} \tag{5}$$

其中，CSU_{temp} 为调度子帧中控制时隙的瞬时占用率；CSU_{old} 为上次计算得到的调度子帧中控制时隙的平均占用率；CSU_{new} 为当前的调度子帧中控制时隙的平均占用率；β 为调度子帧中控制时隙占用率的平滑因子；DSU_{temp} 为数据子帧中微时隙的瞬时占用率；DSU_{old} 为上次计算得到的数据子帧中微时隙的平均占用率；DSU_{new} 为当前的数据子帧中微时隙的平均占用率；γ 为数据子帧中微时隙占用率的平滑因子。

检测完拥塞，本地节点根据 *CSt* 动态调整 *exp*，并设置发送出去的拥塞变量 *St*。若 $CSt = 1$，则减小 *exp*，优先竞争到下一次发送时隙，以控制拥塞并设置 $St = 1$。在这种情况下，要根据不同业务采取不同的方式减小 *exp*。实时业务按式(6)计算，非实时业务按式(7)计算。

$$exp = exp - r_0 \tag{6}$$

$$exp = exp - r_1 \tag{7}$$

其中，r_0, r_1 分别为本地节点处于实时拥塞和非实时拥塞时对 *exp* 的修正值。

若 $CSt = 2$，则根据式(8)来增大 *exp* 并设置 $St = 2$，储存更多的数据发送给下游节点，以提高网络利用率：

$$exp = exp \times 2 + 1 \tag{8}$$

若 $CSt = 3$，则保持 *exp* 不变，设置 $St = 3$。

最后，本地节点发送带有调整后的 *exp* 和 *St* 的 MSH-DSCH 上、下游节点。

3.2 上游拥塞处理

上游节点接收到本地发送的 MSH-DSCH 后，提取 *St* 信息，并做出相应的处理。若 $St = 1$，说明本地节点即将发生拥塞，则上游节点按式(8)来增大 *exp*，以降低上游节点的数据发送速率；若 $St = 2$，说明本地节点轻负荷，则上游节点按式(9)减小 *exp*，以增加上游节点的数据发送速率：

$$exp = exp - r_2 \tag{9}$$

其中，r_2 为上游节点当本地节点处于轻负荷状态时对 *exp* 的修正值。

若 $St = 3$，说明本地节点状态良好，则上游节点保持 *exp* 不变。

3.3 下游拥塞处理

当下游节点接收到MSH-DSCH时，提取 *St* 信息，调整 *exp* 值：若 $St = 1$，说明本地节点即将发生拥塞，则下游节点根据式(10)减小 *exp*，将微时隙的分配情况尽快发送给本地节点，以便尽快接收本地节点的数据，缓解拥塞：

$$exp = exp - r_3 \tag{10}$$

其中，r_3 为下游节点当本地节点处于拥塞状态时对 *exp* 的修正值。

若 $St = 2$，说明本地节点轻负荷，则下游节点根据式(8)增大 *exp*，以免本地节点的数据过早发送，加重轻负荷；若 $St = 3$，说明本地节点状态良好，则下游节点保持 *exp* 不变。

同时，为更好地缓解本地节点的拥塞，下游节点还要根据 *St* 的值采用不同的策略分配微时隙给本地节点：若 $St = 1$，则从一帧中第 1 个空闲的微时隙开始向后依次选择 *r* 个空闲的微时隙；若 $St = 2$，则从一帧中最后一个空闲的微时隙开始向前依次选择 *r* 个空闲的微时隙；否则，从一帧中空闲的微时隙中随机选择 *r* 个空闲的微时隙。其中，*r* 表示一帧中连续的微时隙数，通过式(11)计算：

$$r = Demand\_level / Demand\_Persistence \tag{11}$$

其中，*Demand_level* 为发送数据的大小；*Demand_Persistence* 为发送数据的持续帧个数。它们都包含在MSH-DSCH 消息的 RequestIE 中。

最后，下游节点将微时隙的分配信息加载到 MSH-DSCH 消息的 GrantIE 中，发送出去并回复给本地节点。

4 性能验证

为验证该方法的性能，将它与传统 TCP 拥塞控制进行仿真比较，并给出仿真环境和结果。

4.1 仿真环境

本文使用 NS-2[5]作为仿真平台，它是一个面向对象的、离散事件驱动的网络模拟器，支持多种网络，因此，基于此搭建的新的 Mesh 平台，基本实现了 IEEE802.16 协议中规定的协调分布式调度和 3 次握手等。在图 3 所示的网络拓扑中进行仿真，节点发送距离为 150 m。

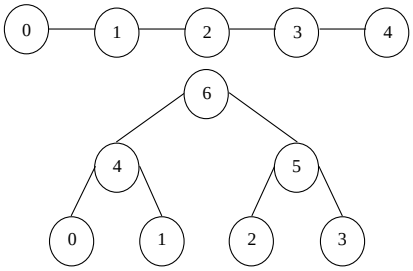

图 3　网络拓扑

应用层采用 FTP 业务，传输层采用 TCP 协议，路由层采用 AODV 协议。表 1 为主要的仿真参数。模拟时间为 100 s，TCP 包大小为 350 字节，缓冲区长度为 20，信道类型为无线信道，数据信道速率为 1 Mb/s，天线为单向天线。

表 1　仿真参数

参数	配置
Topo Space	1 000×1 000
Frame length (v)	8 ms
MSH_CTRL_LEN	8
MSH-DSCH_NUM	8
Scheduling Frames	4
Buffer size	20
Data bandwidth	1 Mb·s^{-1}
Simulation time	100 s
TCP packet size	350 Byte

4.2 仿真结果

仿真中 TCP with MAC Congestion Control 表示 MAC 拥塞控制，TCP without MAC Congestion Control 表示传统 TCP 拥塞控制表示。

在线形拓扑中，节点 0 和 4 间配置一条 FTP 流。图 4 给出了线形拓扑中端到端时延随着仿真时间的增加而发生的变化曲线。可见，从第 7 s 开始，使用 MAC 拥塞控制的时延比传统 TCP 拥塞控制的时延小，且使用后的最大时延才 0.37 s，是可容忍的。

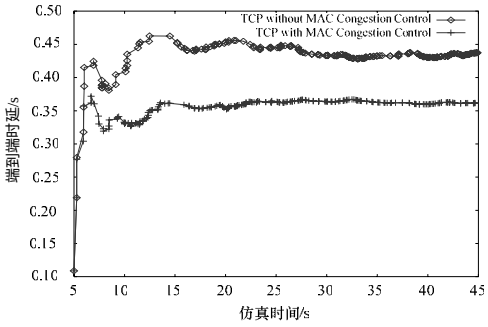

图 4　线形拓扑中的端到端时延

在树型拓扑中，节点 0, 1, 2, 3 分别配置了到节点 6 的相同 FTP 业务。图 5 给出整个网络平均丢包率，图 6 给出平均吞吐量的变化曲线。

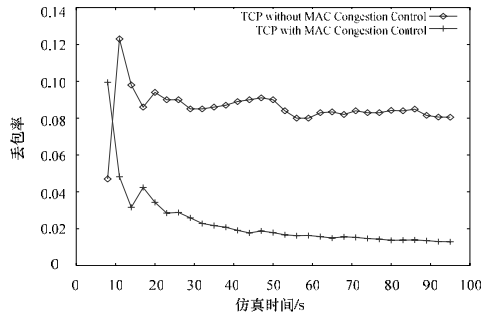

图 5　丢包率

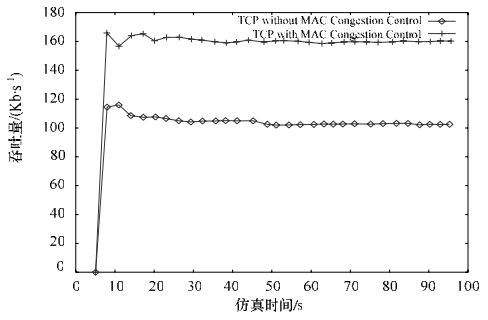

图 6　吞吐量

(下转第 124 页)