第 35 卷 第 9 期 计 算 机 工 程 2009 年 5 月
Vol.35 No.9 Computer Engineering May 2009
·博士论文· 文章编号：1000—3428(2009)09—0004—03 文献标识码：A 中图分类号：TP393

基于 OMNeT++的 Ad Hoc 网络跨层协议栈仿真

王卫疆[1]，李腊元[2]，郑 锋[1]
(1. 空军雷达学院信息与指挥自动化系，武汉 430019；2. 武汉理工大学计算机科学与技术学院，武汉 430063)

摘 要：针对 OMNeT++的 MFW 组件中物理层和 MAC 层模块的错误进行修正和功能改进，实现基于 RBAB 的 IEEE 802.11b 协议速率自适应数据传输，建立移动 Ad Hoc 网络跨层协议栈模型的网络仿真框架，实现将 Ad Hoc 网络 MAC 层的速率自适应机制与其路由协议联合优化设计的仿真。结果表明，在移动环境下，采用跨层设计的网络协议栈能大幅提高 Ad Hoc 网络系统的性能。
关键词：Ad Hoc 网络；跨层协议栈；网络仿真

Cross Layer Protocol Stack Simulation in Ad Hoc Network Based on OMNeT++

WANG Wei-jiang[1], LI La-yuan[2], ZHENG Feng[1]
(1. Department of Information and Command Automation, Air Force Radar Academy, Wuhan 430019;
2. School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063)

【Abstract】Aiming at the inaccuracy in the module of MFW for OMNeT++, the function of PHY/MAC in this module is modified and mended, which implements the rate adaptation MAC protocol based on RBAB in IEEE 802.11b and a simulation model framework is constructed for cross layer network protocol stack with the modified module. An instance about cross layer design which joints rate adaptation in MAC layer and routing algorithm in network layer to improve performance of mobile Ad Hoc network is studied based on this simulation model framework. The simulation results show that cross layer design in the application of Ad Hoc network protocol stack improves the performance of Ad Hoc network significantly in the mobile wireless environments.
【Key words】Ad Hoc network; cross layer protocol stack; network simulation

1 概述

MFW[1]是 OMNeT++(Objective Modular Network Testbed in C++)[2]中基于无线移动网络仿真的模块化插件，是开发设计跨层协议栈的理想工具。目前新版本 MFW 组件中 MAC 层典型的协议模型是 IEEE 802.11b，但其模型存在严重不足：(1)不支持多速率机制的传输：在 MFW 中，节点的发送速率直接从系统配置文件 omnetpp.ini 定义的常数值中提取，因而不具备自动调节传输速率的能力。(2)DCF(分布协调功能)机制有误：在 MFW 中，其 802.11 DCF 所有消息的发送均以节点选择的可变速率发送，而在多速率机制的 DCF 中，控制帧 RTS/CTS 和广播消息的发送是以基本速率发送的。因此，必须对 MFW 的 MAC 层 IEEE 802.11 协议程序进行修改，使其成为真正的多速率自适应 IEEE 802.11b 协议。

本文利用改进的 MFW 插件模块，实现速率自适应的 Ad Hoc 网络跨层协议栈仿真，提高无线自组织网络性能。

2 跨层协议栈组件MFW的结构与实现

MFW 系统框架由 3 个部分组成：

(1)移动管理框架模块，它实现节点的移动性、节点的连接管理和无线信道模型。

(2)移动主机节点模块，它实质是分层结构(ISO/OSI)的网络通用协议栈的标准模块库函数。

(3)为无线网络协议栈仿真模型提供跨层设计策略的模块。基于 MFW 组件的协议栈结构如图 1 所示。

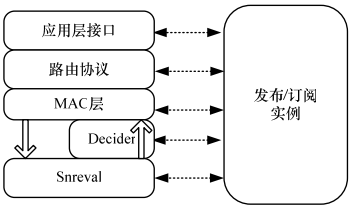

图 1 MFW 组件跨层协议栈结构

MFW 组件系统保留了面向 ISO/OSI 分层的协议栈结构，其物理层和链路层中的 MAC 子层模块组成了无线接口——NIC 复合组件来实现 PHY/MAC 协议。在物理层中，SnrEval 模块用来计算被接收消息的信噪比信息，而 Decider 模块决定该接收的消息是否被丢弃、有误码或被正确接收。当物理层模型须记录不同时间及不同位置信号和噪声功率时，Decider 子模块就提供这些信息和不同的调制方式按需计算出误码率或标记出现差错的位。

为实现无线网络协议栈跨层设计，MFW 组件引入黑箱概念，通过这个机制实现协议层在主机节点中跨层的信息交换，通信的发生是使用发布-订阅机制实现的。在一个主机节

基金项目：国家自然科学基金资助项目(60672137)
作者简介：王卫疆(1970－)，男，讲师、博士，主研方向：计算机网络，无线通信系统；李腊元，教授；郑 锋，副教授、博士研究生
收稿日期：2008-09-04 **E-mail：**hust511_2@163.com

点中仅有一个黑箱模块，黑箱功能必须从 BlackboardAccess 类派生，提供各协议层数据。例如协议栈的物理层向 MAC 层提供数据，可利用 BlackboardAccess 类中的 Publish()方法向黑箱发布这些数据。对一定类型的信息感兴趣的模块可以通过 BlackboardAccess 类的 subscribe()方法订阅这类主题，并当相应的数据被发布时接收通知消息。被发布的数据本身存储在黑箱模块中，通过 Lookup()方法来访问。这种方法允许实现数据的异步查找和多用户存取。

3 基于速率自适应的MFW组件模型

由于 MFW 组件中 MAC 层的 IEEE802.11b 协议模型存在错误与不足，因此必须对该模型进行修正与改进。本文选取基于接收器的自适应速率 RBAR(Receiver-Based AutoRate)算法[3]来实现多速率自适应 IEEE 802.11b 协议，即采用在接收端通过接收 RTS 信号获得即时信噪比，并以信噪比为门限作为速率选择依据。在仿真中利用误码率公式来确定其门限阈值。对于 1 Mb/s 和 2 Mb/s 的传输速率，采用 DBPSK 方式调制，误码率的计算公式为

$$BER = 0.5\exp(\frac{SNR \times BW}{bitrate}) \tag{1}$$

对于 5.5 Mb/s 和 11 Mb/s 的传输速率，则采用 CCK 调制模式，可用 16-QAM 和 256-QAM 的 DQPSK 调制模式近似计算在该传输速率下的误码率：

$$BER = 2 \times (1 - \frac{1}{\sqrt{2^4}}) \times erfc(\sqrt{\frac{2 \times SNR \times BW}{bitrate}}) \tag{2}$$

$$BER = 2 \times (1 - \frac{1}{\sqrt{2^8}}) \times erfc(\sqrt{\frac{2 \times SNR \times BW}{bitrate}}) \tag{3}$$

其中，*SNR* 为信噪比；*BW* 为带宽；*bitrate* 为位速率；*erfc* 为误差函数。

在 MFW 中实现 RBAR 算法，首先要建立基于 IEEE 802.11b 的 NIC 组件。物理层包括 snrEval802.11 模块和 decider802.11 模块。snrEval802.11 模块主要任务是：判断接收的数据包是噪声还是有用数据；保存信道 SNR 数据的等级，并将该信息传递给 decider802.11 模块；将 MAC 层的数据帧变换成无线传输数据帧，形成向无线信道传输的数据包。snrEval802.11 模块则采用下式[4]作为无线电传播模型：

$$P_{received} = \frac{P_{send}\lambda^2}{16\pi^2 d^{\alpha}} \tag{4}$$

其中，P 表示功率；λ 表示波长；d 表示发送和接收节点的距离；α 是路径损失相关系数。decider802.11 模块采集 snrEval802.11 发送的 *SNR* 信息并以此计算误码率。对于 1 Mb/s 和 2 Mb/s 的传输速率，帧头 PDU 传输误码率的计算公式为式(1)；对于 5.5 Mb/s 和 11 Mb/s 的传输速率，其近似计算在该传输速率下的误码率为式(2)和式(3)。在仿真环境的设置中，通信节点传输距离范围(以 $BER = 10^{-5}$ 为基准)如表 1 所示。

表 1 通信节点传输距离范围(路径损失系数为 3)

传输速率/(Mb·s^{-1})	传输距离/m
11.0	100
5.5	130
2.0	150
1.0	180

RBAR 算法对标准的 IEEE 802.11 帧的更改是较小的，例如，图 2 是 RTS/ CTS 控制帧格式，即将标准的 IEEE 802.11b 中 16 bit 的间隔域分为 2 个控制域，分别为速率和数据长度。

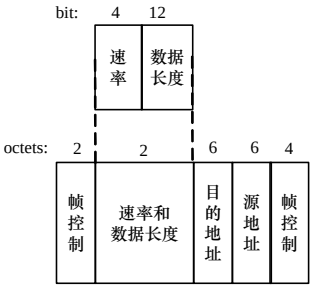

图 2 RTS/CTS 控制帧结构

在实现时只须对 IEEE802.11 协议执行过程进行较少修改。根据图 3 所示的时序，节点 A 位于发射机 Src 发送信号的有效范围内，在收到 RTS 控制信号后，读取 RTS 中的速率和将要发送的数据包长度，并计算出须避让的时间段 D_{RTS}，存入 IEEE 802.11 的 NAV 中。

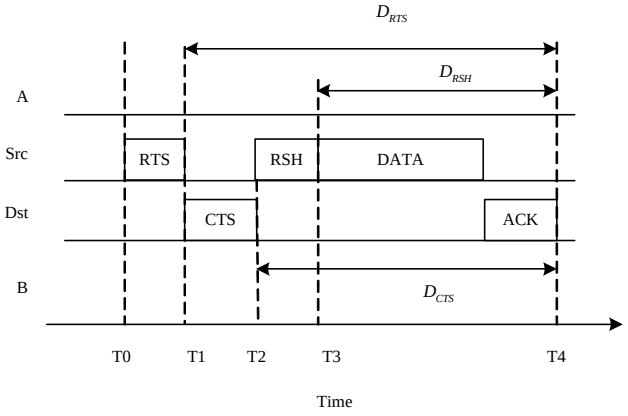

图 3 数据发送过程时序及各节点的 NAV

同时，接收机 Dst 在收到 RTS 后，从 RTS 信号获得实时的信道状况。Dst 速率选择算法选取适应当前信道质量的发送速率，并通过 CTS 向 Src 发出要求的传输速率。同样，节点 B 在 Dst 的有效范围内，收到 CTS 控制信号后，读取 CTS 中的速率和将要发送的数据包长度，并计算出须避让的时间段 D_{CTS}，存入 IEEE802.11 中的 NAV。最后，Src 根据 CTS 要求的速率发送数据包。

Src 在初始的 RTS 中所告知的速率与 Dst 在 CTS 所要求的速率可能不同，这使 D_{RTS} 与节点 A 实际所要求的避让时间有误差，当 Src 发送数据包时，在数据头上插入的一段特殊数据头，即预留子头 RSH。A 节点在收到 RSH 后，读取其中 During 域的数值并马上计算最终的要求时间段 D_{RSH}，然后修改自己的记录，消除 D_{RTS} 和 D_{RSH} 之间的误差。

4 速率自适应Ad Hoc网络跨层协议栈仿真

速率自适应 Ad Hoc 网络协议栈跨层设计原理如下：Ad Hoc 网络中某一节点的物理层对无线信道进行估计得到每对节点链接的实时 SNR，因此，传送节点根据 SNR 的级别自适应地选择数据传输速率，这必然影响 MAC 层的数据包的传输延迟 D_i。在网路层中，路由的选择与判决可将 MAC 层的数据包传输延迟 D_i 作为量度，路由判定规则对网络负载的分布产生影响，进而改善 Ad Hoc 网络的传输性能。本文以基于 IEEE 802.11 的无线网卡作为物理层，在 MAC 层实现速率的自适应的介质媒体接入协议，网络层以改造的 AODV 协议为基础，实现传输速率自适应与路由选择联合的跨层设计

算法。

4.1 跨层协议栈的节点模型

节点模型以 MFW 组件为框架进行功能扩充，每个移动节点被定义为一个复合模块，这种复合模块由 NIC 模块、路由层模块、应用层模块、移动模块及黑箱模块组成，见图 4。

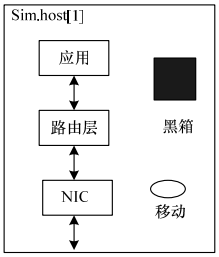

图 4 移动节点复合模块

各层协议栈模块之间的通信是通过消息交换进行的。跨层信息交换由黑箱模块的订阅/发布机制实现。本文选用随机指路模型[5]作为移动模型，NIC 模块实现了基于 RBAR 算法的 PHY/MAC 层协议。

4.2 网络层跨层路由算法

本文采用改造的 AODV 协议，其路由代价函数修改为以 MAC 层的延迟 D_i 为路由量度，使整个路径的 MAC 层总体延迟最小，从而实现跨层的路由协议。

如图 5 所示，MAC 层的延迟 D_{ij} 的定义为从 RTS 控制包发送开始到传输的数据包被接收为止的时间区间，该延迟可作为路由量度。

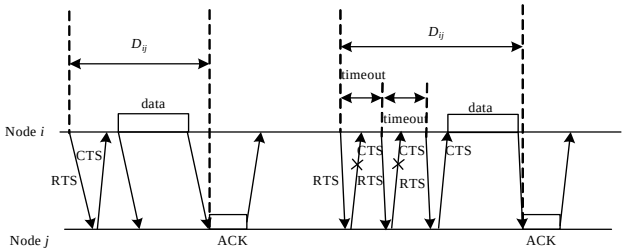

图 5 多个节点的信道竞争而产生的延迟

对 MAC 层延迟估算方法如下：当 MAC 层从路由层接收到向某一个节点发送的单播数据包，它储存当前时间。当它收到该数据包的 ACK 消息，就可计算出节点在 MAC 层的瞬时传输延迟 D_{curr}，即从网络层发送的数据包的接收时间减去 ACK 数据包的接收时间。为了平滑随机抖动，可采用指数移动平滑算法经过 J 次迭代得到计算周期为 T(瞬时延迟的计算周期时间)的第 J 个平均延迟值：

$$D_{\text{avg}}^{j} = (1-\alpha)\times D_{\text{curr}}^{j} + \alpha \times D_{\text{avg}}^{j-1} \tag{5}$$

其中，α 的取值范围为[0, 1]。

标准的 AODV 协议只转发收到的第一个 RREQ 消息或 RREP 消息中不包括最低延迟信息，限制了路由度量值在网络中的扩散。因此，在路由发现过程中，本文针对 RREQ 做如下改进：

(1)增加路由代价(C)扩展域，C 扩展域用在 RREQ 中，收集该 RREQ 所通过链路的 C 值，代价函数定义为

$$C_{(s,d)} = \sum_{c \in path(s,d)} D_{\text{avg}}$$

(2)增加对路由代价更低的 RREQ 的再次广播，即节点在收到重复的 RREQ 时，并不立即丢弃，而是比较新收到的 RREQ 到源节点的反向路径与以前获得的到达源节点的反向路径的 C 值，如果更低则更新反向路由表，并以新的值再次广播 RREQ。

4.3 仿真实现与结果分析

为证实速率自适应和路由量度选用的效果，移动节点可在以下 3 种方式的仿真实验中进行比较：(1)采用速率自适应算法和跨层 AODV 路由协议；(2)采用速率自适应算法和标准的 AODV 路由协议；(3)采用固定速率和标准的 AODV 路由协议。比较网络路由的 3 个性能指标为：吞吐量，延迟和包投递率。通过方式(2)和方式(3)的对比，可估计 PHY/MAC 速率自适应算法的效果。而方式(1)和方式(2)的比较，可评估跨层的路由协议算法效能。

设随机生成 20 个通信节点，其运动场景参数设置见表 2。

表 2 运动场景的参数设置

参数	设定值
Ad Hoc 网络节点数	20
移动范围	600 m×800 m
节点最大移动速度	1 m/s, 5 m/s, 10 m/s, 15 m/s, 20 m/s
静止时间	2 s
仿真时间	100 s

应用层的业务模式由 CBR 在 UDP 协议下进行。负载可通过 CBR 的分组发送率调节，而固定分组大小和 CBR 流量数目。分组发送率从 10 分组/s 到 60 分组/s，以 10 分组/s 为等步长增加，数据分组大小为 512 Byte，控制分组大小为 32 Byte。在不同负载条件下，对比吞吐量、端到端延迟和包投递率 3 个性能指标，如图 6~图 8 所示。

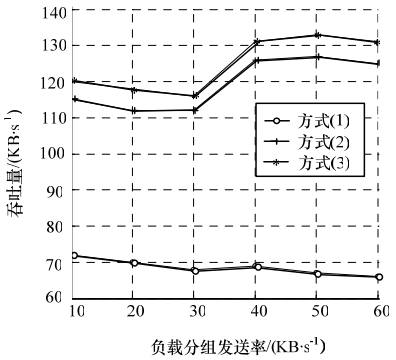

图 6 网络吞吐量与负载分组发送率关系

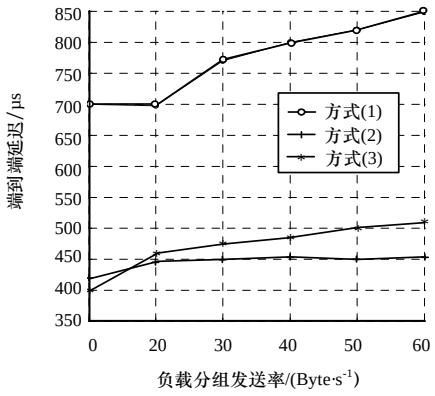

图 7 端到端延迟与负载分组发送率关系

随着负载的增加，平均吞吐量也有所增加。对于单一速率算法下的标准 AODV 路由方式，拥塞发生在负载的包速率达到 40 packet/s。当包速率再增加很多时，网络吞吐量反而下降不多。由于采用速率自适应算法，在方式(1)和方式(2)

(下转第 10 页)