

密级： 保密期限：

# 北京邮电大学

## 硕士学位论文



题目： 无线多模网关组网机制  
的设计与实现

学 号： 2012110644

姓 名： 胡金宇

专 业： 计算机科学与技术

导 师： 马华东

学 院： 计算机学院

2015 年 1 月 18 日



### 独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：\_\_\_\_\_ 日期：\_\_\_\_\_

### 关于论文使用授权的说明

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在\_\_年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名：\_\_\_\_\_ 日期：\_\_\_\_\_

导师签名：\_\_\_\_\_ 日期：\_\_\_\_\_



# 无线多模网关组网机制的设计与实现

## 摘 要

近年来，无线通信技术快速发展并被应用到环境监测、应急通信等诸多领域。由于地理环境的复杂以及通讯设施的不完善，使得单一的无线通信方式很难满足实际需求。无线多模通信网关可以利用多种通信手段，以多跳的方式组成无线 Mesh 网络，为数据可靠传输提供保障，其中设计和实现一种适合无线多模网关的组网机制是急需解决的问题。

本文通过对无线多模通信网关和无线 Mesh 网络进行研究和分析，在无线 Mesh 网络路由协议基础上，针对无线多模网关组网的实际需求设计和实现了一种无线多模网关组网机制。本文在 DSDV (Destination-Sequenced Distance-Vector) 路由算法的基础上使用节点间跳数和节点间链路时延两个度量指标，改进和优化了路由选择策略。本文提出了一种多模网关选择算法，根据节点间跳数、3G 网络的往返时延、丢包率以及卫星发送负载多种因素综合选择网关，采用 3G 网络优先、卫星网络备选的策略合理选择传输方式。最后在无线多模网关节点上完成了对组网机制的测试，验证了组网机制的可行性。

**关键词：**无线 Mesh 网络，DSDV，网关选择，3G，卫星



# DESIGN AND IMPLEMENTATION OF NETWORKING MECHANISM FOR WIRELESS MULTI-MODE GATEWAY

## ABSTRACT

In recent years, with the rapid development of wireless communication technologies, they have been used in a variety of fields such as environment monitoring and emergency communications. Due to the complexity of geographical environment and deficiency of communication facilities, single-mode wireless communication system is difficult to satisfy the requirement of the practical applications. The wireless multi-mode gateway can take advantage of multiple wireless communication methods to guarantee reliable transmission of data. Therefore, it's an important issue to design and implement a networking mechanism by using wireless multi-mode gateways.

In this thesis, we study the wireless multi-mode gateway and mobile ad-hoc network. We design and implement a networking mechanism by using wireless multi-mode gateways based on the routing protocol of mobile ad-hoc network. On the basis of DSDV (Destination-Sequenced Distance-Vector) routing protocol, we optimize the routing selection policy, by using hop count and link quality as metrics. We propose a gateway selection algorithm, which considers multiple factors, including hop count, average round trip time, average packet lost rate of 3G wireless network and load of the satellite sending queue. The algorithm uses 3G wireless network preferentially, and uses the satellite network as a backup. Finally, we test the networking mechanism on the wireless multi-mode gateway and verify the feasibility of the mechanism.

**KEY WORDS:** multi-mode network, routing protocol, gateway selection, 3G, satellite





# 目录

第一章 绪论 .....	1
1.1 研究背景及意义 .....	1
1.2 国内外研究现状 .....	1
1.3 论文主要工作 .....	2
1.4 本文章节安排 .....	3
第二章 技术背景介绍 .....	4
2.1 传统无线通信网络 .....	4
2.1.1 3G 通信网络 .....	4
2.1.2 WiFi 通信网络 .....	4
2.1.3 卫星通信网络 .....	5
2.2 无线 Mesh 网络 .....	6
2.2.1 无线 Mesh 网概念 .....	6
2.2.2 无线 Mesh 网络结构 .....	6
2.3 无线 Mesh 网路由协议 .....	9
2.3.1 无线 Mesh 网路由协议分类 .....	9
2.3.2 经典无线 Mesh 网路由协议介绍 .....	11
第三章 无线多模网关组网机制设计 .....	13
3.1 无线多模网关组网介绍 .....	13
3.2 无线多模网关组网机制总体方案 .....	13
3.3 DSDV 路由算法 .....	14
3.3.1 Bellman-Ford 算法 .....	14
3.3.2 环路避免 .....	15
3.3.3 触发更新 .....	18
3.3.4 阻尼震荡 .....	18
3.4 网关自适应选择 .....	18
3.4.1 3G 网络质量评估 .....	18

3.4.2 卫星转发.....	20
3.4.3 网关选择策略.....	22
3.5 无线多模网关组网机制--DSDV 算法改进 .....	26
3.5.1 路由度量改进 .....	27
3.5.2 路由消息扩展 .....	28
第四章 无线多模网关组网机制实现 .....	32
4.1 无线多模网关组网机制实现总体设计 .....	32
4.2 无线多模网关组网机制实现 .....	33
4.2.1 网卡维护 .....	33
4.2.2 路由表维护 .....	36
4.2.3 路由主控模块 .....	39
4.2.4 路由决策模块 .....	43
4.2.5 3G 网络探测模块 .....	44
4.2.6 卫星转发 .....	49
4.2.7 网关自适应选择模块 .....	56
第五章 网络搭建与测试 .....	60
5.1 M-Link 无线多模通信网关 .....	60
5.2 无线多模网关自组网测试 .....	60
5.3 无线多模网关自适应选择测试 .....	62
5.3.1 3G 网关自适应选择测试 .....	62
5.3.2 卫星网关自适应选择 .....	64
第六章 总结与展望 .....	65
6.1 论文工作总结 .....	65
6.2 总结建议 .....	65
参考文献 .....	67
作者攻读学位期间发表的学术论文目录 .....	70

## 第一章 绪论

### 1.1 研究背景及意义

随着移动通信技术突飞猛进,无线网络接入技术也取得了长足的进步,各种通信手段纷纷涌现,例如蜂窝移动网络、宽带无线接入网络、卫星网络。移动蜂窝网络以 GSM、CDMA、3G、4G 网络为代表<sup>[1]</sup>;宽带无线接入技术以 IEEE802 系列标准为基础,主要包含四个方面:无线个域网(Wireless Personal Area Network, WPAN),无线局域网(Wireless Local Area Network, WLAN)、无线城域网(Wireless Metropolitan Area Network, WMAN)和无线广域网(Wireless Wide Area Network, WWAN);卫星网络主要以国外的 Wildblue-1、SpaceWay-F3、Inmarsat、铱星和国内的北斗卫星通信系统为代表。各种通信网络在实际应用中有各自的优势,如 3G、4G 网络可以通过移动基站高速接入广域网,WiFi 可以实现短距离内的高速数据传输,卫星网络可以在通信基础设施不足的情况下实现数据远距离传输。

在实际应用中,单一的通信手段往往不能完全满足应用需求。例如在环境监测、抢险救灾等复杂野外环境下,由于网络基础设施的不完善或者设施损毁,造成部分地方产生“信息孤岛”或者信号无法覆盖的“死区”。而且在一些具有特殊数据传输需求的环境监测网络中,存在文本数据、图像、多媒体数据等多种异构数据<sup>[2][3]</sup>,不同数据类型在不同的通信方式下传输代价又相差很大。面对上面的情况,如何综合利用多种通信网络,合理选择通信方式,成为解决上述问题的关键。

在环境监测领域,由于监测环境地形复杂、地貌多种多样,单一的通信方式很难满足实际求,例如 3G、WiFi 等网络很难覆盖整个监测区域,部分地形复杂的区域需要多跳“接力”完成数据传输。依托实验室自主研发的 M-Link 无线多模通信网关能有效地实现环境监测区域网络覆盖,但多模网关间如何组网以及网关如何选择使得传输性能最优、传输代价最小是一个亟待研究的问题。

### 1.2 国内外研究现状

由于无线多模网关间能组成无线 Mesh 网,针对无线 Mesh 网的路由协议、网关选择算法目前并没有正式的标准,但国内外已经有了一些研究成果。

国内方面,文献[4]针对混合型无线 Mesh 网,在混合无线 Mesh 网络协议的工作原理基础上,为了减少网络传输的平均时延、提高网络吞吐量,结合按需路由和前摄路由两种模式,提出了一种混合无线 Mesh 网络路由协议。文献[5]提出

了一种基于网关负载的无线 Mesh 网络多网关选择策略,该策略属于“概率选择式网关负载均衡”和“区分式网关负载均衡”策略的结合。文献[6]提出了一种基于选播机制的网关选择策略,该网关选择策略既考虑了路由性能指标,又考虑了网关性能指标,并对网络环境中不同网络对服务质量要求的差异,优化了网关选择判据,保证不同网络数据的 QoS。

国外方面,针对无线 Mesh 网路由度量的研究,已经有学者提出一种将期望传输次数(ETX),作为网络性能的一种判断依据,随后又有研究人员在期望传输次数基础上提出了期望传输时间(ETT)作为另一种网络性能判断依据<sup>[7]</sup>。Braun 提出了 AODV-CGA 路由协议<sup>[8]</sup>,在 AODV 路由协议基础上加入了按需的网关发现机制。R.Baumann 和 Heimlicher 等人提出了基于域的网关选择策略 FBR<sup>[9]</sup>,每个节点选择周围某个邻居作为其默认网关,把所有到 Internet 的包转发给默认网关。考虑到不同服务对 Qos 需求的不同,Min Kim, Ilkyeun Ra 等人又提出了基于 Qos 的 Mesh 网络路由协议<sup>[10]</sup>。

### 1.3 论文主要工作

本文通过分析传统无线通信网络、无线 Mesh 网,参考无线 Mesh 网路由协议设计了一种无线多模通信网关组网机制。通过该组网机制,无线多模通信网关间能够组成自组织网络、进行网关自适应选择、传输方式自适应选择。并在实验室自主研发的 M-Link 无线多模通信网关上对该组网机制进行了测试。本文具体工作如下:

首先,本文分析、对比了传统无线通信网络、无线 Mesh 网,分析了无线多模网关组成的无线 Mesh 网的特点。

其次,本文根据无线多模网关节点组成的无线 Mesh 网络的特点,参考现有的无线 Mesh 网路由协议,在 DSDV 路由协议基础上设计了一种适用于该网络的组网机制。通过该组网机制,无线多模网关间能够组成自组织网络,节点间可以“多跳”协同传输数据。设计了网关自适应选择算法,该算法综合考虑了无线多模网关间的跳数、链路时延、3G 网络时延、3G 网络丢包率、卫星发送队列负载情况等多种因素自适应地选择网关、选择传输方式,降低传输时延、降低传输代价。

最后,本文将设计的组网机制在实验室自主研发的 M-Link 无线多模通信网关上做了实现,搭建了无线 Mesh 网络,通过文本提出的组网机制实现数据的传输。

## 1.4 本文章节安排

本文设计和实现了一种无线多模网关组网机制，各章节的安排具体如下：

第一章，绪论，介绍了课题的研究背景、研究意义，介绍了论文的主要工作以及论文的章节安排。

第二章，技术背景分析，首先介绍了传统无线通信网络相关技术，其次介绍了无线 Mesh 网络相关技术，最后介绍了无线 Mesh 网络中的路由算法。

第三章，无线多模网关组网机制设计，首先对无线多模网关组网进行介绍，根据无线多模网关组成无线 Mesh 网络结构特点在 DSDV 路由协议基础上设计了无线多模网关组网机制，其次详细介绍了 DSDV 路由算法和网关自适应选择算法设计思想以及该组网机制对 DSDV 路由度量和路由消息的扩展。

第四章，无线多模网关组网机制实现，主要介绍了设计的组网机制在实际的软硬件平台上的实现方法，包括各个功能模块的实现方法和流程。

第五章，网络环境搭建与测试，主要介绍了如何利用实验室自主研发的 M-Link 无线多模通信网关搭建无线 Mesh 网络，并对提出的无线多模网关组网机制进行验证。

第六章，总结与展望，总结了本文主要实现的功能，同时对今后进一步工作进行了展望。

## 第二章 技术背景介绍

### 2.1 传统无线通信网络

#### 2.1.1 3G 通信网络

3G 网络是指利用蜂窝移动通讯技术的第三代移动通信技术的线路和设备铺设而成的通信网络, 3G 网络可以提供稳定、高速数据传输环境。3G 网络将无线通信与因特网等多媒体通信手段相结合, 是新一代移动通信系统。

目前主要有三种由国际电联认可的标准分别是 WCDMA、CDMA2000<sup>[11]</sup>、TD-SCDMA。1998 年, 国际电信联盟 (ITU) 着手讨论和制定国际移动通信-2000 (IMT-2000) 标准, 并把其作为下一代移动蜂窝技术标准。在 IMT-2000 协议发展过程中, 北美推出了 CDMA2000, 这种通信标准是在全球标准 1.25MHz 基础上扩展频段而形成的, 具有更高效、更窄带宽的优点。欧盟推出了 GSM 绘制的频带 5MHz RTT 的宽频 CDMA, 即 WCDMA。IMT-2000 标准对 CDMA2000 技术和 WCDMA 都进行了认可, 它们都可以兼容现有的通信标准。后来我国的大唐电信公司与国际合作, 在 RTT 的基础上研制了时分同步码分地址接入 (TD-SCDMA) 标准。2001 年 IMT-2000 正式把 TD-SCDMA 纳入 3G 标准。

3G 网络能够为用户提供优质的文本、语音和数据服务。与现有的无线通信技术相比较而言, 3G 技术具有显著增加系统容量、数据传输速度等优点, 能够为生产、生活带来更多便捷。此外利用 3G 技术可以在不同网络间进行无缝漫游, 通过 3G 通信网络可将通信系统接入 Internet 网络, 便于 3G 用户享受互联网带来的便利。

在无线数据传输领域, 3G 通信技术常常被应用于远距离的数据传输。终端设备只需要配置能够连接 3G 网络的模块, 便可以通过 3G 网络连接 Internet 等骨干网。因此 3G 网络作为一种主要的通信手段, 在无线高速的数据通信领域有很强的应用前景。

#### 2.1.2 WiFi 通信网络

WiFi(Wireless Fidelity)技术即 IEEE802.11 协议, WiFi 的两大技术优点是无线接入和高速传输, 其中 IEEE802.11b 最高传输速度为 11Mbps, IEEE802.11a 与 IEEE802.11g 的最高传输速度为 54Mbps。现在使用较为广泛的 IEEE802.11a 与 IEEE802.11g 设备, 在频率资源上并不会受限制, 它们使用的频段为

2.4~2.4835GHz 的免许可频段, 因此 WiFi 技术和其它无线通信技术相比较而言其使用成本更加低廉。WiFi 主要由以下部分组成<sup>[12][13]</sup>:

(1) 站点 (Station), 所有能通过无线介质接入网络的单元都可以被称作站点, 无线站点可以分为无线接入点和客户端两类。

(2) 基本服务单元 (Basic Service Set, BSS)。所有能够进行通信的站点的集合被称作基本服务单元。基本服务单元可以被分为两类: 独立基本服务单元和基础设施基本服务单元。

(3) 扩展服务单元 (Extended Service Set, ESS)。扩展服务单元是互连的基本服务单元的一个集合, 扩展服务单元中的接入点通过分布式系统来连接。每个扩展服务单元都有一个 32 字节长度的 ID 表示, 这个 ID 叫做 SSID。

(4) 分布式系统 (Distribution system, DS)。分布式系统用来连接扩展服务单元中的接入点。分布式系统可以是有线或者无线的。目前无线分布式系统大都基于 WDS 或者 Mesh 协议。

随着 WiFi 技术的普及, 其已经被认为是 3G 网络和有线网络的一个重要的补充。相比其它的通信方式而言, WiFi 网络具有较高的传输速率和较低的传输成本。因此, 引入 WiFi 网络能够提高数据传输的可靠性, 降低数据传输的成本。但是, 由于 WiFi 的传输距离受到限制, 同时传输信号容易受到周围环境的干扰。因此, WiFi 技术只适用于近距离数据传输。

### 2.1.3 卫星通信网络

卫星通信系统实际上也是一种微波通信技术, 它利用卫星作为中继站转发微波信号, 从而实现数据在多个地面站之间的传输。由于卫星基本工作在几百、几千、甚至上万公里轨道上, 因此卫星信号的覆盖范围远大于一般的移动通信系统。目前能够提供数据传输服务的卫星通信系统主要以摩托罗拉的铱星(Iridium)通信系统和中国的北斗系统为代表。

铱星移动通信系统是摩托罗拉公司为美国铱星公司设计的一种全球性卫星通信系统, 该系统由多颗低轨道卫星和高轨道卫星组成。铱星系统具有轨道低、传输速度快、信息不易损耗、通信质量高的特点。铱星系统除了提供电话业务外, 还提供传真、全球定位(GPS)、无线电定位和全球寻呼业务<sup>[14]</sup>。

铱星系统实现了全球性覆盖, 业务没有盲区, 其提供的 SBD 短数据业务能够支持突发短数据的传输, 目前支持三种通信方式: (1) Email 方式, 这种方式终端设备可以把用户数据通过铱星网关转发到某个特定的电子邮箱地址; (2) 点对点通信, 这种方式终端设备可以把用户数据转发给位于其它区域的终端设备;

(3) IP\_DIRECT 方式, 这种方式终端设备可以把用户数据通过铱星网关转发给

用户指定的 Internet 主机。

北斗系统是中国正在建设的导航与通信卫星系统。北斗系统主要由空间端、地面端和用户端三部分组成：空间端由 5 颗静止轨道卫星和 30 颗非静止轨道卫星组成；地面端由主控站、注入站和监测站等若干个地面站组成；用户端通常由北斗用户终端以及与美国 GPS 等其它卫星导航系统兼容的终端组成。在数据通信方面，北斗具有双向报文通信功能用户可以一次传送 40-60 个汉字的报文信息<sup>[15]</sup>。

卫星通信系统具有覆盖范围广、广播特性、组网灵活、不受地理限制、通信成本与距离无关等优点，能够在复杂环境下完成数据传输的任务，是其它通信方式的一个有力补充，在民用和军用领域都得到了十分广泛的应用。但由于卫星通信的高成本、低速率，使得卫星通信不能作为常规通信手段，在实际应用中只能作补充和应急使用。

## 2.2 无线 Mesh 网络

### 2.2.1 无线 Mesh 网概念

WMN(Wireless Mesh Network)<sup>[16]</sup>通过无线链路把固定的和移动的节点连接起来，构成的一个多跳的移动自组织网络。

随着无线通信技术的发展，人们对 802.11a/b/g 等网络技术有了更加深入的研究，Mesh 网络也在消费者、企业界以及学术界中引发了广泛关注。WMN 技术具有支持多跳连接、网络部署灵活、覆盖范围广以及高传输速率等特点，目前，已经普遍被业界认为是下一代无线网络技术的一个重要的研究方向。

WMN 是移动 Ad Hoc<sup>[17]</sup>网络的一种特殊形态，但两者又存在差异，WMN 和移动 Ad Hoc 网络的网络结构和网络连接方式的不同。WMN 中，节点具有两种功能：一是对等节点之间交换数据；二是作为接入网关，通过特定的网关节点，把 WMN 接入 Internet。各种通信设备都可以通过有线或者无线卡接入 Mesh 路由器，进而接入 WMN。此外，Mesh 路由器中的路由或者网关接入功能能把现有的无线网络，例如移动蜂窝网络、无线传感网络、WiFi 等，接入到 WMN，进而接入到 Internet。因此，通过 WMN 用户能够获得单一网络无法提供的服务。

### 2.2.2 无线 Mesh 网络结构

WMN 的拓扑结构是网格状的，WMN 主要由两类节点组成：Mesh 路由器和 Mesh 客户端。

Mesh 路由器 MR (Mesh Router, MR)：无线 Mesh 路由器用于对 Mesh 网络



进行组网，Mesh 路由器具备多个无线接口，可以接入多种异构网络。它对传统无线路由器做了扩展，Mesh 路由器上运行的路由协议可以使 WMN 组成一个 Ad Hoc 网络，并且通过相应的网关选择算法，选择合适的网关节点，把 Mesh 路由器接入 Internet；Mesh 路由器又具备无线 AP 的功能，能够把其它网络通过有线、无线的方式接入 Mesh 路由器，进而接入 Internet。

**Mesh 客户端 MC (Mesh Client, MC):** 除了作为主机完成数据收发功能外，还兼备路由器的路由功能。一方面，MC 作为主机通过运行相应的应用程序可以完成数据采集、数据传输等功能；另一方面，MC 通过运行相应的路由协议来完成路由发现、路由更新、路由维护等路由器的功能。Mesh 客户端还具备无线 AP 的作用，可以通过有线、无线的方式对某种网络提供接入功能，但不具备异构网络接入和网关协议转换等功能，所以和无线 Mesh 路由器相比，Mesh 客户端的软硬件都要简单的多。Mesh 客户端的设备种类繁多，如笔记本、PAD、手机等无线通信设备。

通过在网络中分配不同功能的节点，WMN 可以分为如下三种类型<sup>[17]</sup>：

#### (1) 基础设施 Mesh 网：

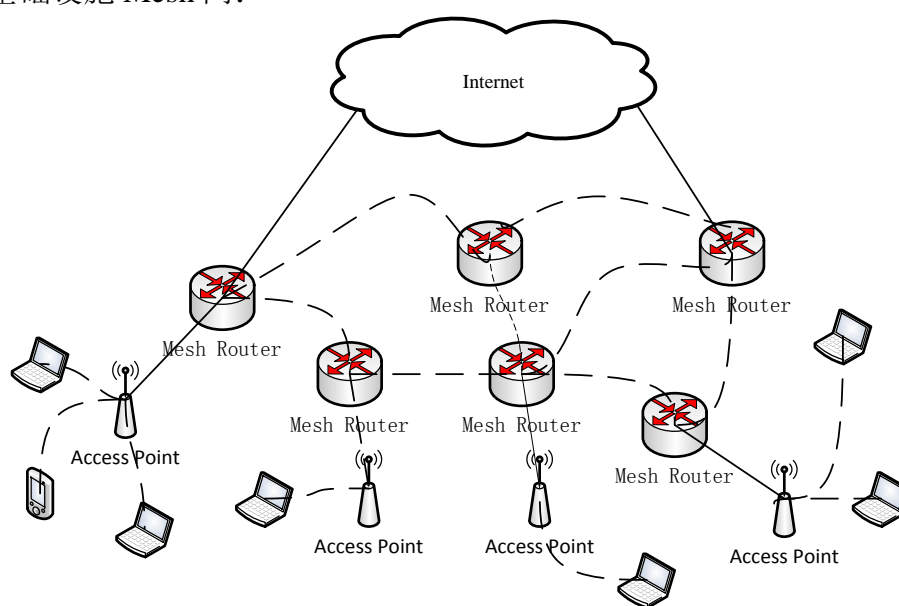


图 2-1 基础设施 Mesh 网结构图

如图 2-1 所示的基础设施型 Mesh 网络，Mesh 路由器之间组成了一个基础设施网络，方便客户端接入。节点间的实线表示有线连接，节点间虚线表示无线连接。Mesh 路由器间可以利用各种无线通信技术，保证 Mesh 路由器间异构网络的连接。Mesh 路由器间组成了一个具备自配置、自愈特点的网状网络。通过 Mesh 路由器网关接入的功能，Mesh 路由器能够接入 Internet，通过这种方法 WMN 提供了一种把其它现有网络接入 Internet 的方法。

#### (2) 终端设备 Mesh 网

如图 2-2 所示的终端设备 Mesh 网结构，终端设备 Mesh 网络通过 Mesh 客户端组成了一个对等网络。在这种网络中主要依靠 Mesh 客户端来完成路由、配置和相应的业务处理任务。因此在这种网络中并不需要 Mesh 路由器。终端设备 Mesh 网络通常只具备一种无线通信设备，这种网络实际上就是一个 Ad Hoc 网络。

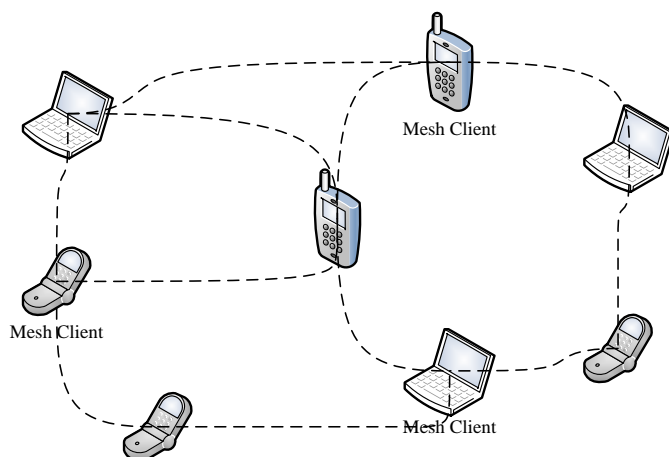


图 2-2 终端设备 Mesh 网结构图

### (3) 混合 Mesh 网

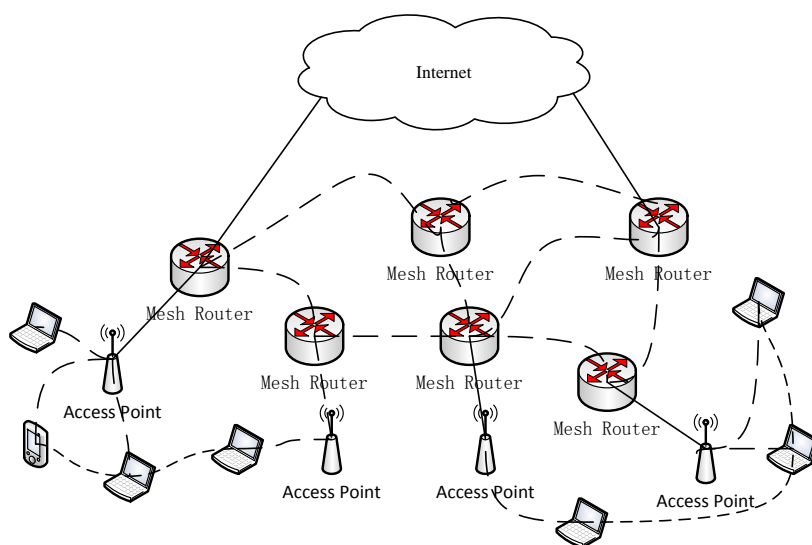


图 2-3 混合 Mesh 网结构图

如图 2-3 所示的混合 Mesh 网的结构，混合 Mesh 网是由基础设施 Mesh 网和终端设备 Mesh 网络融合而成的。在混合 Mesh 网中，终端设备可以通过 Mesh 路由器来访问其它网络或者其它 Mesh 客户端。这种结构的网络还提供把现有的移动蜂窝网、WiFi、WiMAX 等网络接入 Internet 的功能。这种类型的网络结合了 WMN 的所有优点：支持自组织网络，具备自我形成、自我修复、自我组织的功能；对跳无线网络，并能够通过 Mesh 路由器把现有网络接入 Internet；Mesh

路由器位置基本固定，完成路由和配置功能，显著减少了 Mesh 客户端和其它终端节点的开销。

## 2.3 无线 Mesh 网路由协议

路由协议是无线 Mesh 网络研究的一个重要的研究方向。无线 Mesh 网是一个多跳的自组织网络，它与以往的单跳无线网络往往具有很大区别。无线 Mesh 网络必须根据网络自身的特点来设计路由协议，例如网络拓扑结构、节点移动性、安全性等要求。本节首先介绍无线 Mesh 网络路由协议的分类，然后对几个典型的无线 Mesh 网路由协议进行介绍。

### 2.3.1 无线 Mesh 网路由协议分类

最常见的分类方法是根据建立路由的驱动机制不同，将无线 Mesh 网的路由协议分为主动路由协议和被动式路由协议和混合式路由协议。根据网络拓扑结构的差异，无线 Mesh 网络路由协议又可以分为平面路由协议（Flat Protocols）和分簇路由协议（Clustered Protocols）。按照是否利用地理位置信息来辅助路由建立，无线 Mesh 网络路由协议又可以分为基于地理位置信息的路由协议和非地理位置信息路由<sup>[18]</sup>。如图 2-4 所示：



图 2-4 无线 Mesh 网络路由协议分类

#### (1) 主动式路由协议

主动式路由协议又被称作表驱动路由协议，网络中的节点通过周期性广播路由表来交换路由信息，动态获得全网的信息。每个节点均要维护一张路由表，表中记录了到其它节点下一跳路由器信息，节点会周期性的向网络中其它节点广播自身路由表，或者当节点检测到链路状态变化时，向其它节点广播路由更新，节点接收到路由更新后会按照相应的算法对路由表项进行检测并对自身路由表进

行更新。当节点要发送报文时，会从自身路由表中查找到目的节点的下一跳路由器，并不用发起寻路操作，所以主动式路由的数据传输延时比较小。但由于每个节点都要维护到其它节点的路径信息，并且需要定期广播路由更新，浪费了带宽，并且路由表中存储了无用的路由信息。

目前成熟的 Mesh 网络主动式路由协议包括如下<sup>[19][20][21]</sup>：DBF（Distributed Bellman Ford）、DSDV（Destination-Sequenced Distance-Vector Routing）、HSR（Hierarchical State Routing）、FSR（Fish State Routing）、OLSR（Optimized Link State Routing）。

## （2）被动式路由协议

被动式路由协议又被称作按需路由协议或者反应式路由协议，其是根据发送节点的传输需求，按需进行路由发现、路由建立的过程。因此节点的路由表内容可能仅仅是整个网络拓扑信息的一部分。被动式路由协议节点只需要维护正在被使用的路由信息，并不存储无用的路由信息，当节点要向目的节点发送数据时，首先检查自身路由表，如果路由表中没有到达目的节点的路径，才开始建立路由。被动式路由协议一般分为两个阶段：路由发现和路由维护。

路由发现：当节点需要向网络中某个目的节点发送数据的需求时，首先查找自身节点路由表，如果不存在相应的路由表项，就发起一个路由发现过程，节点会广播一个路由请求(RREQ)分组报文，当网络中某个节点接收到此报文后查找自身路由表，如果找到一个合适的路由，节点会向源节点单播一个路由响应(RREP)报文，路由发现过程结束。当网络中没有到某节点的路由时，路由发现过程也结束。

路由维护：路由建立以后，需要对此路由进行维护，直到不再需要此条路由来进行通信，或者通过网络中任何路径都无法再访问目标节点。

目前成熟的 Mesh 网被动式路由协议包括如下：AODV（Ad-hoc on Demand Distance Vector）、DSR（Dynamic Source Routing）、LAR（Location Aided Routing）。

## （3）分簇路由协议

在平面式路由协议中所有节点具有平等的地位，所以这种结构又可以称为对等式结构。而在分簇结构中，网络被划分为簇(cluster)，每个簇由一个簇头(cluster head)和多个簇成员(cluster member)组成，簇头作为这个簇的代表，负责数据的转发，这些簇头又形成了高一级的网络，在由簇头组成的高一级网络中，可以再选出一个节点作为这个簇的簇头，直到最高级。

目前分簇路由协议主要有 CGSR<sup>[22]</sup>(Clusterhead Gateway Switch Routing)，ZRP 和 CDEAR 路由协议。

## （4）基于地理位置信息的路由协议

基于地理位置信息的路由协议利用节点获取的地理位置信息来优化路由的发现、路由维护和数据转发,根据地理位置信息能够在空间上获取信息传输的最佳路径,避免信息在整个网络中无规则的洪泛,减少路由协议的开销,优化路径选择<sup>[23]</sup>。

具有代表性的基于地理位置信息的路由协议有位置辅助的路由(Location Aided Routing,LAR), 栅格位置服务(Grid Location Service,GLS), DREAM (Distance Routing Effect Alorithm for Mobility)。

### 2.3.2 经典无线 Mesh 网路由协议介绍

#### (1) DBF (Distributed Bellman Ford) 协议

DBF 路由协议是最经典的主动式路由协议,网络中的每个节维护一张路由表,路由表会随着网络环境的变化动态更新,该路由表包含了节点到网络中其它所有可达节点的路径信息:目的地址、下一跳路由器地址以及节点间的度量,度量一般为节点间的跳数、链路时延、带宽等。各节点会周期性地向邻居节点发送路由更新,路由更新中包含点本节点和其它节点间的路由开销,收到路由更新的节点提取路由更新中的这些信息并按照 Bellman-Ford 算法更新自己的路由表,并重复上一过程。当节点要发送数据时会根据自身路由表选择合适的下一跳路由器转发数据。

#### (2) DSR (Dynamic Source Routing) 协议

DSR 路由协议是一种被动式的路由协议,不同于表驱动路由协议,其并不维护一张到达其它节点的路由表,而是采用源路由的方式查找下一跳路由器。当源节点有报文发送需求时,首先检查本地路由缓存是否有到达目的地的路由,如果有则按照这个路由转发数据,如果没有源节点开启路由发现过程,源节点向周围邻居节点广播路由请求报文,每个请求报文由报文序号和源节点编号唯一表示,请求报文中包含目的地址和路由记录,当中间节点接收到路由请求后首先查询本地路由缓存,如果有到达目的地的路由缓存,则无需处理请求,否则此节点为中间节点,在路由记录中插入中间节点地址,继续转发该路由请求。当接收到该路由请求的节点是目的节点时,就行成了一条从源节点到目的节点的路径,目的节点会向源节点发送路由应答,并在该应答中加入此路径,目的节点和源节点会更新本地路由缓存。源节点收到路由应答后便可以转发报文,各个节点会周期性地维护本地路由缓存。DSR 路由协议一种按需的路由协议,并不需要维护全局路由表,降低了周期性广播的开销,但是节点会维护陈旧的路由信息。

#### (3) DSDV (Destination-Sequenced Distance-Vector Routing) 协议

DSDV<sup>[24]</sup>路由即目的节点序列距离矢量路由协议,是一种移动 Ad-hoc 网络

中经典的表驱动的主动式路由协议。在 **Bellman-Ford** 算法的基础上, 利用序列号来区分路由新旧, 解决了 **DBF** 路由成环和无穷计数的问题。在网络中每个节点都有路由功能, 每个节点维护一张到网络中其它节点的全局路由表, 节点向周围邻居节点广播周期性更新或者触发更新, 邻居节点接收到的路由更新后根据 **Bellman-Ford** 算法来调整路由表来适应整个网络的变化。路由协议中使用了目的序列号来区分路由新旧, 新的序列号表示新的路由更新, 利用序列号的奇偶性来区分过期路由, 偶数序列号的位有效路由, 奇数为过期路由, 从而避免产生路由环路或者无穷计数。每条路由表都有一个对应的目的序列号, 目的序列号由源节点产生并更新, 中间节点只负责转发路由表项, 不做修改。当节点从邻居节点接收到一条新路由表项后会对新路由表项的目的序号和度量进行判断, 如果新路由表项的目的序号比现有的路由表项的目的序号新或者目的序列号相等但是度量更小时更新现有路由表项; 否则, 丢弃该路由表项。**DSDV** 的特点如下:

(a)先验式的路由协议, 获取路由的延时小, 适合于网络结构相对稳定、实时性要求高的网络

(b)引入了目的序列号, 通过序列号来区分新旧路由, 避免环路和无穷计数的产生。

(c)路由更新有两个方式: 1) 周期性广播, 周期性地向周围邻居节点通告自身的整个路由表, 能够使刚加入网络的节点及时了解网络拓扑; 2) 触发更新, 当网络拓扑发生变化是立即向周围邻居节点广播路由更新, 加快路由的收敛。

#### (4) **AODV** (**Ad-hoc on demand distance vector**) 协议

**AODV** 协议是一种被动式的源驱动路由协议。当有节点要向网络中其它节点发送数据时, 如果本地路由表中没有到达目的节点的路由, 则需要以多播的形式发送 **RREQ**(路由请求)报文。**RREQ** 报文中记录着源节点和目的节点的 **IP** 地址, 邻居节点收到 **RREQ** 报文后, 首先判断目自身是否为 **RREQ** 中的目的节点。如果是, 则向源节点单播 **RREP**(路由回应)报文; 如果不是, 则首先在本地路由表中查找是否有到达目的节点的路由, 如果有, 则向该源节点单播 **RREP** 报文, 否则继续转发 **RREQ** 直到找到目的节点。

在网络资源充足的情况下, **AODV** 路由协议还可以开启 **HELLO** 报文广播, 通过定期 **HELLO** 报文来维护路由, 当节点检测到某一链路断开后, 节点就发送 **ERROR** 报文来进行路由进行修复。

## 第三章 无线多模网关组网机制设计

### 3.1 无线多模网关组网介绍

无线多模网关和客户端一起可以组成无线 Mesh 网络，网关间可以通过 WiFi (2.4GHz)、340MHz 等无线设备互连，组成自组织网络，节点间互为备份。无线多模网关可以通过 3G 网络、卫星网络接入 Internet 或者选择其它网关节点作为接入网关。客户端可以通过有线网的方式接入无线多模网关，从而通过无线 Mesh 网络把客户端接入 Internet。另外，无线多模网关会根据传输内容自适应地选择传输方式，从而做到兼顾传输效率和传输带代价。网络结构图如图 3-1 所示：

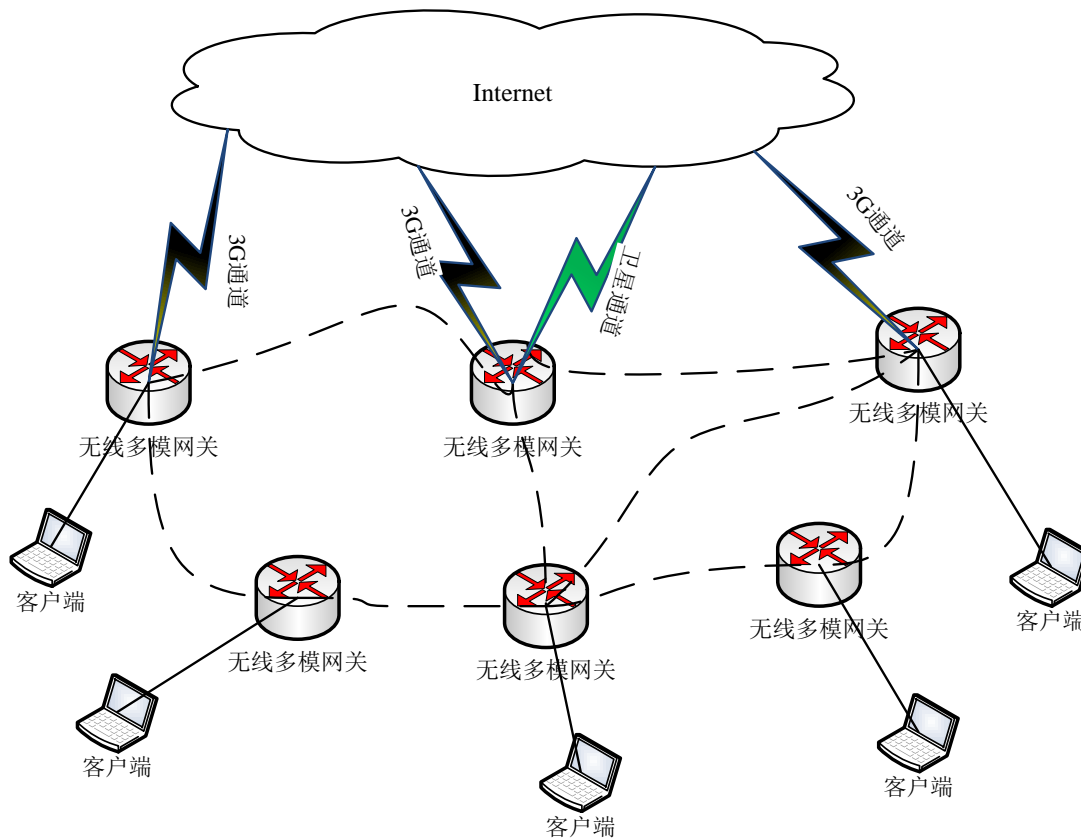


图 3-1 无线多模网关组网结构图

### 3.2 无线多模网关组网机制总体方案

本文研究的无线多模网关组成的无线 Mesh 网络为静态网络，网络中无线多模网关节点位置相对固定、节点数量不大、不会频繁移动，所以无线多模网关组网机制主要包括两方面内容：一是自组网路由算法，通过该算法，无线多模网关间会组成自组织网络，二是网关选择算法，通过该算法节点会选择合适的网关，

把节点接入 Internet。本文选择在主动式距离向量路由协议 DSDV 的基础上对路由度量做改进以适应无线 Mesh 网络中网关节点间通信方式多模的特性，并且设计了一种网关自适应选择算法，该算法通过计算网关节点 3G 网络平均往返时延、平均丢包率、卫星发送负载、节点间跳数来综合选择最优网关把无线多模网关接入 Internet。最后，为了实现无线多模网关组网机制，本文选择在 DSDV 路由协议的基础上对路由消息进行了相应扩展。

### 3.3 DSDV 路由算法

DSDV 路由算法是无线 Mesh 网络经典的主动式的距离向量路由算法，核心算法基于 Bellman-Ford 算法，在此基础上做了环路避免、触发更新、阻尼震荡等改进来保证算法的正确性，提高算法的可用性。

#### 3.3.1 Bellman-Ford 算法

Bellman-Ford<sup>[25]</sup>算法是求解单源最短路径问题的一种算法。它的原理是对图中所有边进行松弛操作来得到所有可能的最短路径。网络拓扑可以看作是图的一种，如图 3-2 所示的网络中每个节点 A 维护着两个数据：到节点 S 的估计距离，记作  $D(A)$ ，和它到达 S 的下一跳路由器  $NH(A)$ 。算法初始条件是  $D(S) = 0$ ， $D(A)$  为无穷大， $NH(A)$  未定义。

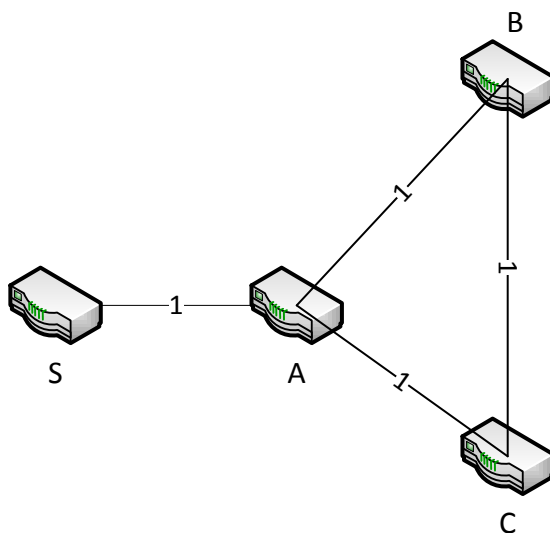


图 3-2 Bellman-Ford 算法描述

网络中的任一节点 B 会周期性地向周围邻居节点广播路由更新，更新消息中携带  $D(B)$ ，即 B 点到达 S 的估计距离。当 B 的邻居 A 接收到 B 的路由更新后，A 会检查 B 是否是 A 路由表中某条路由的下一跳，如果是更新的路由消息，则把  $NH(A)$  设置为 B， $D(A)$  设置为  $C(A, B) + D(B)$ 。否则节点 A 会比较  $C(A, B) + D(B)$



和  $D(A)$  的大小, 如果  $C(A, B) + D(B) < D(A)$  说明接收到的路由更新要比原来的路由代价低, 把  $NH(A)$  设置为  $B$ ,  $D(A)$  设置为  $C(A, B) + D(B)$  即可。

Bellman-Ford 算法的一个缺点就是网络中的路由会形成环路。例如当网络拓扑如图 3-3:

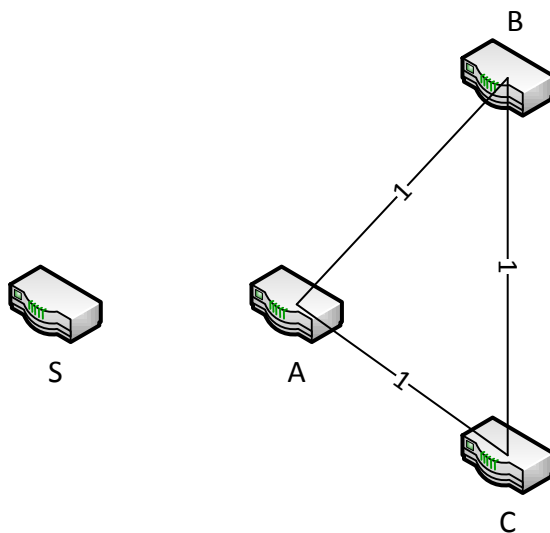


图 3-3 Bellman-Ford 算法路由环路

当路由收敛以后,  $D(B) = D(C) = 2$ ,  $NH(B) = NH(C) = A$ 。假设这个时候  $S$  和  $A$  之间的链路断开了, 当  $A$  检测出  $S$  和  $A$  之间的链路断开后,  $B$  并不知道  $S$  已经和  $A$  断开, 继续发送路由更新广播, 当  $A$  接收到来自  $B$  的路由更新,  $D(B) = 2$ ,  $A$  把下一跳路由器设置为  $B$ , 并把  $D(A)$  设置为  $3$ ,  $A$  广播路由更新,  $B$  和  $C$  收到  $A$  的路由广播后分别更新各自的路由表, 周而复始,  $A$ 、 $B$ 、 $C$  三个节点不断增度量值, 直到度量达到无穷大。这期间  $A$ 、 $B$  两个节点分别把下一跳路由器设置为对方, 在网络中就产生了环路和无穷计数, 当节点有数据包发送到节点  $S$  时, 数据包就在  $A$ 、 $B$  两点间来回震荡, 造成网络资源的严重浪费。

### 3.3.2 环路避免

针对 Bellman-Ford 算法在网络断开时会产生路由环路和和无穷计数的问题, 在  $RIP^{[26]}$  路由算法中采用了水平分割和毒性翻转的方法来消除路由环路, 具体方式是:

#### (1) 水平分割

当网络收敛以后,  $NH(B) = A$ ,  $B$  到达  $S$  的下一跳路由器是  $A$ , 即  $B$  可以到达  $S$  的路由是从  $A$  处学习到的, 所以当  $A$  向  $B$  广播路由更新时就不必再包含这条路由更新。

#### (2) 带毒性反转的水平分割

带毒性反转的水平分割的策略是虽然  $B$  可以到达  $S$  的路由是从路由器  $A$  学

习到的，还是在向 A 广播路由更新的时候广播这条消息，只不过把 metric 设为无穷大，这样当 A 按照 Bellman-Ford 算法计算路由更新时就不会把从 B 处得到的路由更新考虑进去。

在有线网络中水平分割和毒性翻转能够很好地解决 RIP 无穷计数和路由环路的问题，但在无线网络中由于无线传输介质的广播特性，这种方法并不适用于包含 3 个或更多节点的环路，如图 3-4 的网络：

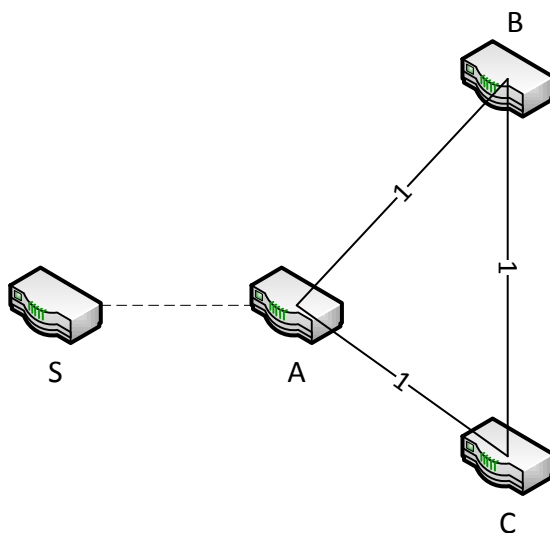


图 3-4 RIP 算法无法解决路由环路

当 S 和 A 断开后，B、C 节点虽然不会向 A 广播它们能到达 S 的路由更新，但 B、C 间会广播，当 B 点接收到 C 点的路由更新时会把 NH(B) 设置为 C，D(B) 设置为 3，并向 A 发送路由更新，A 向 C 发送路由更新，周而复始，形成无穷计数和路由环路。

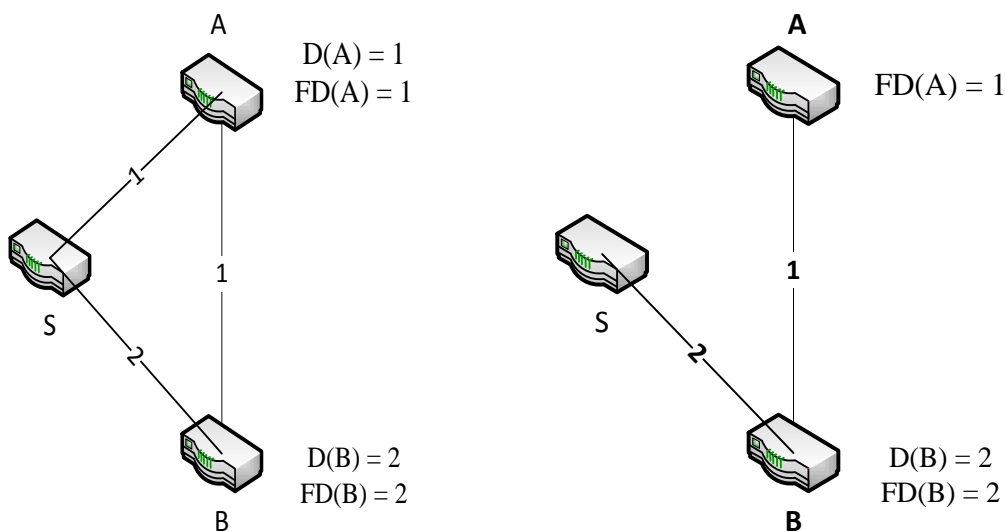


图 3-5 利用可行性条件造成节点饿死

因此针对无线网络环境，DSDV 路由算法提出了用目的序号来区分路由新旧

避免出现环路的策略。从图 3-5 的情况可以看出当节点更新路由时只有 **metric** 变大时才可能出现路由环路，所以我们可以定义一种可行性条件，当可行性条件为路由更新的度量值不大于当前的路由度量值才更新路由表。例如，只有当  $C(A,B) + D(B) \leq D(A)$  时节点 A 才接受节点 B 的路由更新。即如果节点 A 选择 B 作为下一跳路由器，那么  $D(B)$  肯定小于  $D(A)$ ，只要所有节点都遵循这个限制条就能保证转发图就是无环的。很明显在 DSDV 中这种可行性条件会造成某些节点的路由无法正常更新，即某些节点会因为可行性条件的限制被“饿死”，如图 3-5 当节点 A 和节点 S 之间的链路断开以后虽然节点 A 可以通过节点 B 到达节点 S，但是节点 A 接收到的节点 B 的路由更新  $D(B) = 2$ ,  $C(A,B) + D(B) > FD(A)$ ，不满足可行性条件，不会接受该路由更新。

DSDV 中采用了目的序列号的方法解决了这个问题。在路由更新的消息中除了携带度量值，还包含一个单调递增的序列号，源节点负责序列号的更新，其它节点在广播路由更新消息的过程中并不对序列号进行修改，一个路由更新用一个序号和度量的元组表示  $(s,m)$ ，只有当序列号更新或者序列号相同但度量更小时才接受这个路由更新。即如果  $FD(A) = (s, m)$ ，接收到的路由更新为  $(s', m')$ ，只有当  $s' > s$  或者  $s' = s$  并且  $m' < m$  时才满足可行性条件。如图 3-6a 所示，此时节点 S 的序列号为 137，当节点 S 增加序列号后，路由更新广播到节点 B，节点 B 更新自己的序列号，变为图 3-6b，当节点 A 接收到节点 B 的路由更新后由于序列号更新，满足可行性条件，接收此更新，节点 A 便可以通过节点 B 到达节点 S。

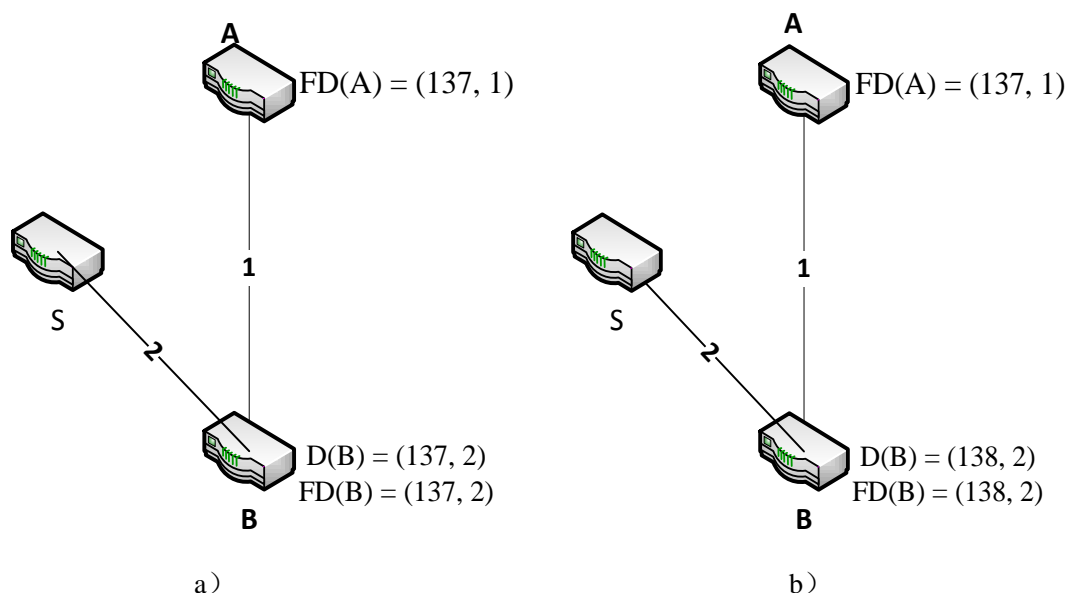


图 3-6 利用序列号解决饿死问题

### 3.3.3 触发更新

由于 DSDV 是主动式的表驱动路由协议, 算法会周期性的向邻居节点广播节点的整张路由表, 网络中节点频繁向全网广播整个路由表会造成很大的系统开销, 增加网络负担。所以 DSDV 采用了周期性更新和触发更新相结合的方式, 低频率地进行周期性广播, 高频率地进行触发更新的广播。每当网络状态发生变化时, 路由算法会向周围邻居节点通告受影响的路由表项。这里网络状态变化是指节点间跳数的变化和链路的断开。由于触发更新的存在, 不仅减少了系统开销, 降低了网络负担, 而且能够及时使邻居节点了解网络拓扑的变化, 加快网络状态的收敛。

### 3.3.4 阻尼震荡

由于 DSDV 路由更新的策略是只接受目的序号更新或者目的序号相同但度量值更小的路由更新, 假设节点接收到先后两个路由更新, 两者的目的序号相同, 但是后者度量更小, 这样先更新了一个度量更大的下一跳路由器, 而后又要更新一个度量更小的下一跳路由器。这样会反复向网络中发送触发更新, 当网络中节点数据众多时这种现象更为明显, 短时间内会造成路由表的反复修改。DSDV 中采取了一个叫做稳定时间的域来记录某条路由达到稳定的平均时间, 这样当节点的路由表改变时并不立即发送触发更新, 而是等到达到平均稳定时间后才发送触发更新, 这样在这段时间内节点可能已经获得了最优路由, 减少了不必要的触发更新的发送次数。

## 3.4 网关自适应选择

无线多模网关组成的无线 Mesh 网络中具备 3G 和卫星通信能力的无线多模网关可以作为网关节点把其它节点接入 Internet, 考虑到传输效率和传输成本的问题, 要对接入网关进行合理的选择。本小节首先介绍了对 3G 网络进行网络质量评估的方法, 然后介绍了如何通过铱星网络转发 IP 数据以及如何评估卫星发送状况, 最后结合 3G 和卫星两种通信方式对网关进行选择。

### 3.4.1 3G 网络质量评估

3G 是无线多模网关中主要的远距离通信方式, 3G 具有速率大、带宽高、信号稳定的特点, 但在不同地区 3G 信号覆盖度并不一致, 3G 通信质量也有所差别, 所以要对 3G 网络的通信质量进行评估。评估的标准有很多, 例如: 网络的往返时延、丢包率、带宽、负载等等。这里选取了往返时延和丢包率来作为 3G 网络通信质量的评价条件。

在传统的网络连通性监测中，如常规的路由设备，可以利用链路层的 ARP 协议进行检测，但这种方法只能检测点到点的连通性对 3G 网络并不适用。Ping 命令是在网络中使用的最频繁的测试网络连通性的测试工具之一，ping 命令使用 ICMP（Internet Control Message Protocol 网际控制消息协议）来发送 ICMP echo 类型请求数据包，如果目标主机能够接收这个请求，则发回 ICMP 响应。程序会按时间和成功响应次数估算往返时延和丢包率。由于 3G 网络的复杂性，只检测无线多模网关节点到 Internet 上某个主机间的往返时延和丢包率不足以估计 3G 网络的通信质量。所以这里在利用 ICMP 网络状态检测的基础上同时对多个 Internet 上的主机计算往返时延和丢包率来综合估计 3G 网络的通信质量。

在 ICMP 协议中定义了两种报文：查询报文和错误报文。TCP/IP 协议将 ICMP 协议封装在 IP 报文中，在网络上发起 ICMP 的查询报文时，ICMP 协议规定了网络中的任何一台安装了 TCP/IP 协议的设备都可以将该报文转发或者回应该报文。如果网络连通性正常则接受到 ICMP 查询报文的主机应该回应查询报文，否则回复携带错误码的错误报文。



图 3-7 ICMP 报文格式

ICMP 协议报文格式如图 3-7，协议头共占 8 个字节，其中类型和代码用来区分 ICMP 的报文类型。校验和字段用来对整个 ICMP 报文进行校验，保证在网络上报文传输的完整性。2 个字节的标识符用于表示发送 ICMP 探测包的进程，序号用来区分查询报文和回应报文的顺序，防止查询报文和回应报文乱序。

表 3-1 ICMP 报文类型和代码字段含义

类型	代码	说明
0	0	回应报文
3	0	网络不可达
	1	主机不可达
	2	协议不可达
	3	端口不可达
	6	目标网络未知
	7	目标主机未知
	8	源主机被隔离
4	0	源站抑制
8	0	查询请求
10	0	路由请求

本文使用的 ICMP 报文类型如表 3-1，当类型为 8 代码为 0 时，该报文为查询请求报文。收到该报文的目标主机会自动回应类型为 0，代码为 0，并复制了查询报文数据部分的回应报文。如果出现主机不可达的情况，出现异常的节点会发送类型为 3 代码类型为 1 报文。如果网络无法连接，直接返回类型为 3 代码为 0 的网络不可达报文。通过以上几种报文能够判断出网络的连通性。

假设算法每隔 T 秒钟会对 W 个 Internet 上的主机连续发送 N 个 ICMP 请求报文，把发送时间加入到请求报文的数据部分，当网络上的主机可达时会复制数据部分并回送应答报文，发送节点接收到应答报文后便可以统计节点和主机间的往返时延和探测包的丢包率。假设节点和第 i 个 Internet 主机间共发送 N 个探测报文，接收到  $M_i$  个应答报文，探测报文的发送时间为  $Req_{ij}$ ，应答报文的接收时间为  $Rep_{ij}$ ，则可以计算出节点和这些公网主机间的平均往返时延和探测包的平均丢包率为如下：

$$avg\_rtt = \left( \sum_{i=1}^W \left( \sum_{j=1}^{M_i} (rep_{ij} - req_{ij}) \right) / M_i \right) / W \quad (3-1)$$

$$avg\_lost = (\sum_{i=1}^W (N - M_i)) / N \times W \quad (3-2)$$

### 3.4.2 卫星转发

在无线多模通信网关中 3G 和卫星作为两种远距离通信方式保证数据能够及时、可靠的传输到 Internet 上，当节点自身以及周围节点的 3G 网络的通信质量不能满足通信需求时节点会启动自身或者通过周围节点的卫星终端转发数据。卫星转发部分由 M-Link 无线多模网关和汇聚节点共同完成，其中，无线多模网节点上有报文检测过滤模块和卫星模块参与卫星转发，汇聚节点上有卫星模块参与卫星数据重组和投递。无线多模网节点上的报文检测过滤模块负责报文监测和过滤以及 IP 包的缓存，卫星模块负责 IP 数据报和 SBD 数据报之间的转换以及通过铱星 AT 指令集来驱动铱星收发 SBD 消息。

#### (1) 报文检测及过滤

由于无线多模网关承载的业务是 IP 数据的转发，这里报文检测和过滤只针对 IP 网络。Libpcap(Packet Capture Library)<sup>[27]</sup>库是一个流行的网络数据包捕获开发包，被广泛应用于 UNIX/Linux 平台下的网络数据包的捕获。它被广泛应用于网络安全领域，利用 Libpcap 开发包已经开发了许多著名的网络安全工具，如 Unix/Linux 平台下著名的网络数据包捕获和分析工具 Tcpdump，网络入侵检测系统 Snort，以及著名的网络协议分析工具 Wireshark 等。

Libpcap 开发包主要网络分接头 (Network Tap)和数据过滤器(Packet Filter)两部分组成。网络分接头从网络设备驱动程序中收集网络数据拷贝,过滤器根据特定的过滤规则来决定是否收取该数据包。Libpcap 利用 BSD Packet Filter (BPF)<sup>[28]</sup>算法对网卡接收到的来自链路层的数据包进行过滤。BPF 算法的总体思想是在 BPF 监听的网络中,网卡驱动将接收到的数据包复制一份给 BPF 过滤器,过滤器根据用户自定义的规则决定是否接收该数据包以及需要拷贝该数据包的哪些内容,然后将过滤后的数据提供给与过滤器关联的上层应用程序。BPF 的架构如图 3-7 所示:

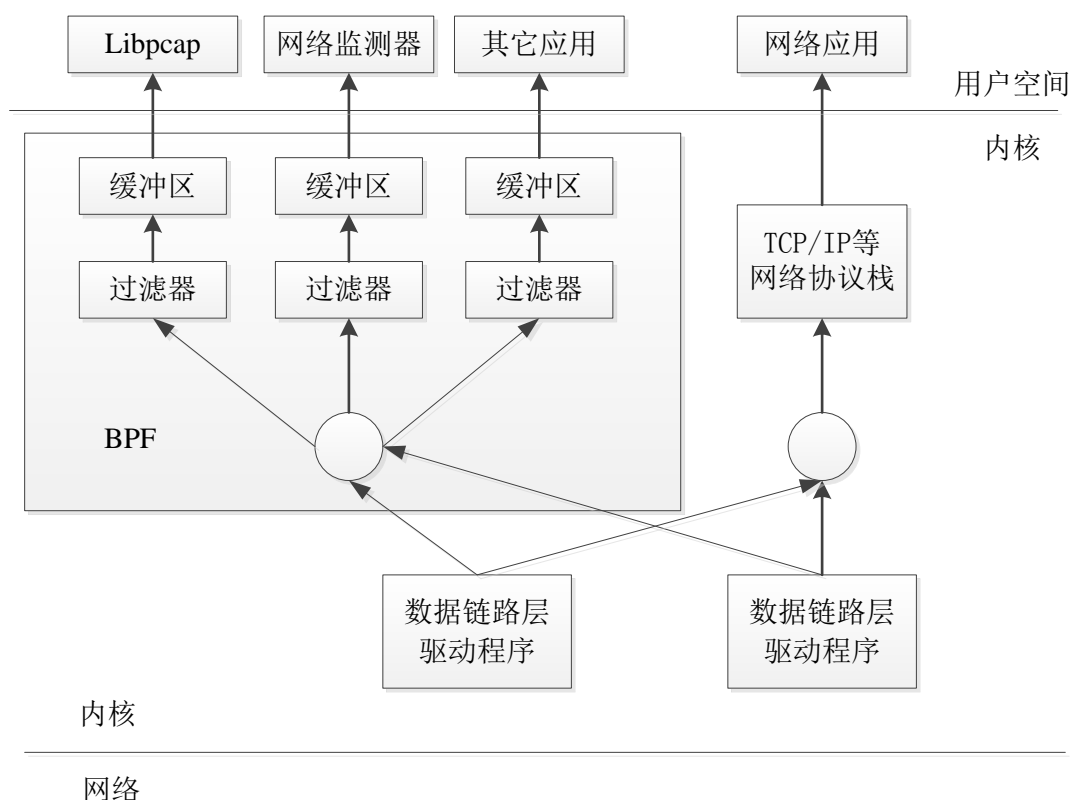


图 3-8 BPF 架构图

Libpcap 的包捕获机制是在数据链路层上加了一个旁路。每当有数据包经过网卡时, Libpcap 首先利用已经建立的 socket 从链路层驱动程序中获得一份该数据包的拷贝,再通过 tap 函数将数据包发送给 BPF 过滤器。BPF 过滤器根据用户自定义的过滤规则对数据包进行逐一匹配,匹配成功则放入内核缓冲区,并传递给应用层的用户缓冲区,匹配失败则直接丢弃。如果程序没有设置过滤规则,所有数据包都将被放入内核缓冲区,并传递给相应的用户层缓冲区。由于无线多模网关承载的数据多数多种多样并且卫星通信费用昂贵,需要设置合理的过滤规则,保证捕获适合利用卫星发送的数据。

## (2) IP 数据报缓存

由于 TCP 具有三次握手、超时重传、拥塞控制等复杂的机制并不容易利用卫星网络来转发 TCP 报文,本文只考虑通过铱星转发 UDP 报文的情况。考虑到 IP 网络传输速度比卫星网络要快得多,如果利用卫星网络转发 IP 数据报需要做 IP 网络和卫星网络间传输速率的适配,这里通过缓存来实现,对突发的 IP 数据报进行缓存然后利用卫星发送,为此设计了 IP 包存储方案。利用上面介绍的 Libpcap 库对报文进行检测过滤后能够从原始的 IP 数据报中提取出 IP 数据报的源 IP、目的 IP,目的端口,净荷长度和净荷等数据,在 IP 报文中有 20 个字节的协议头,对于卫星 SDB 业务来说并不需要传输全部的头部信息,利用目的 IP 和目的端口就可以确定数据传输的目的端。设计的 IP 包缓存文件格式<sup>[29]</sup>如表 3-2 所示:

表 3-2 抓包缓存文件格式

源 IP	目的 IP	目的端口	净荷长度	净荷
------	-------	------	------	----

每个缓存文件中只缓存一个 IP 数据包,当铱星终端成功发送完 IP 数据包后立即删除相应的缓存文件,降低了缓存文件占用的空间,提高了存储空间的利用率。

### (3) 铱星短报文收发

由于铱星 SBD 业务每次传输数据最多为 340 字节,因此需要设计合理的数据分片格式把大数据进行分片、重组多次发送。设计的分片格式如表 3-8 所示:

0	31	63
MagicCode		源IP
目的IP		目的端口
分片索引		是否是最后一片
数据部分		

图 3-9 卫星 SBD 报文分片格式

卫星分片报文在 IP 缓存文件基础上增加了用来分片的字段,其中分片标识用来防止伪造的卫星分片报文干扰卫星分片重组。源 IP 用于在汇聚节点端构造源 IP 和卫星编号之间的映射,用于下行控制指令的传输。尾片标志用于表示这个分片是整个数据分组的最后一个分片,接收到这个分片后可以对分片进行重组,投递到相应的服务器。

### 3.4.3 网关选择策略

#### (1) 网关发现



无线多模网关组成的无线 Mesh 网络中的节点可以通过特定的无线多模网关接入 Internet，但节点如何发现和收集网关信息，选择最优网关接入是一个十分重要的问题。目前网关发现的策略主要有三种：主动发现策略、被动发现策略和混合式发现策略<sup>[30][31]</sup>。

#### (a) 主动式发现策略

无线多模网关组成的无线 Mesh 网络中的网关节点会周期性向全网广播自身的网关公告信息，接收到网关公告信息的节点根据收集到其它网关节点的网关公告信息以及自身的网关公告信息，从而选择最优网关接入 Internet。主动式发现策略能够使网络中节点快速找到最优网关，网络延迟小。但由于要周期性广播网关公告信息，因而系统开销大、浪费大量带宽。

#### (b) 被动式发现策略

被动式发现策略由要接入 Internet 的节点发起，不需要网关节点周期性的广播网关通告消息，当节点要接入 Internet 时，该节点会向全网广播网关接入请求消息，中间节点会转发该网关接入请求消息，直到网关节点接收到网关接入请求消息，之后网关节点单播网关应答消息给发起网关接入的节点，等节点收集到所有网关应答消息后综合评定选择最优网关接入。

#### (c) 混合式发现策略

混合式发现策略，综合了两个网关发现策略的优点，是主动式和被动式两种网关发现策略的折中，混合式发现策略对网关通告消息的跳数进行限制，从而减小了网关通告广播范围，在网关通告作用范围内，节点采用主动式的网关发现策略；范围之外，节点采用被动式网关发现策略。由于在混合式网关发现策略中网关通告消息只在网关节点附近一定跳数范围内进行广播转发，相比主动式的网关发现策略降低了因为需要在全网范围内进行网关通告广播带来的开销，同时也避免了被动式发现策略中网关宣召网络延迟大的缺点。

由于本文针对的是静态的无线 Mesh 网络，网络中节点个数不多、节点位置基本固定，所以，本文在 DSDV 路由协议基础上采用了主动式的网关发现策略：节点会定期向周围邻居节点广播网关通告消息。

### (2) 网关选择

在无线多模网关组成的无线 Mesh 网络中每个节点在一段时间内会接收到多个可达网关的网关通告消息，节点会根据制定的规则合理选择网关接入 Internet。

在 Ad Hoc 网络中已经有一些成熟的网关选择策略，下面选取一些典型策略进行介绍：

(a) 基于 MTV 的网关选择策略

MTV(mobility-tracing value)策略在选择网关节点时把节点的移动性作为考虑的首要因素，目的是为了选择和移动节点间链路最为稳定的网关来把节点接入 Internet。Ad-hoc 网络中每个节点维护一个邻居节点列表，列表中记录一个和邻居节点间链路断开的频率值，这个值和网关节点的移动性有关。当节点没有收到邻居周期性发送的 Hello 探测消息时，MTV 值增加。当 MTV 值增加到一定阈值后从邻居节点列表中删除该邻居。MTV 值越小，表示节点和邻居节点间的链路越稳定。为了维护网络状态的稳定，当节点要接入 Internet 时，首先选择 MTV 值最小的网关节点，MTV 值相同时选择跳数小的网关节点。如果跳数也相同则选择生命周期长的链路作为接入网关。

(b) MMGD 网关选择策略

MMGD 网关选择策略将信道征用水平 (contention level) 和拥塞水平 (congestion level) 转换成虚拟跳数 (virtual hops)，和节点到达网关的物理跳数 (physical hops) 一起组成网关选择的指标。该策略对网关通告消息进行了扩展，在网关通告消息中加入了网关跳数 GW\_METRIC。其中：

$$gw\_metric = virtual\ hops + physical\ hops = congestion\ level + contention\ level + physical\ hops \quad (3-3)$$

节点的拥塞水平用节点发送队列的长度来表示，信道征用水平通过信道在忙碌时占用的时间来表示。MMGD 策略通过网关节点周期性向邻居节点广播网关通告消息，在消息中记录网关节点的 GW\_METRIC，当邻居节点接收到网关通告消息后会比较消息发送节点是否和默认网关节点相同，相同时只有当 GW\_METRIC 小于 MGW\_METRIC 时会更新自身的 MGW\_METRIC。当网关通告消息的发送节点和默认网关不同并且 GW\_METRIC 小于 MGW\_METRIC 时，修改 MGW\_METRIC 值并且修改路由从而切换默认网关。

(c) 基于跳数和网关负载的策略

基于跳数和网关负载的策略会同时考虑节点和网关节点间的跳数和网关节点的负载两个因素。当满足公式 3-4 时

$$\alpha \times \delta(i, j) + \beta \times \lambda_j = \min(\alpha \times \delta(i, k) + \beta \times \lambda_k : 1 \leq k \leq n) \quad (3-4)$$

移动节点  $MN_i$  会选择网关  $MG_j$  作为其接入网关。其中  $n$  为  $MN_i$  可以检测到的网关数目。 $\delta(i, j)$  表示移动节点  $MN_i$  和网关节点  $MG_j$  之间的跳数,  $\alpha$  为跳数所对应的权重值,  $\lambda_j$  表示网关  $MG_j$  的负载,  $\beta$  为网关负载所对应的权重值, 并且  $\alpha + \beta = 1$ 。通过给  $\alpha$ ,  $\beta$  分配不同的值可以表示跳数和网关负载在网关选择中所占地位的不同。例如当  $\alpha = 1$  时表示移动节点在选择网关时只考虑节点和网关之间的跳数; 当  $\beta = 1$  时表示移动节点选择网关时只考虑网关的负载, 当  $\alpha = \beta = 0.5$  时表示两个因子所占地位相同。如果移动节点和网关之间的跳数相等, 那么移动节点会选择负载较轻的网关作为接入网关; 当网关的负载相同时, 移动节点选择距离自身跳数少的网关作为接入网关。

由于本文考虑的是静态网络下的网关选择策略, 不考虑节点的移动性, 信道征用水平和网关负载在实际应用中难以准确度量, 无线多模网关节点具有 3G 和卫星两种远距离通信方式, 要综合考虑两种通信方式的通信能力。综上本文的网关选择策略既考虑了网关性能又考虑了业务需求, 提出了一种优先选择 3G 网络作为接入网关传输方式, 当周围邻居节点的 3G 网络都不能满足通信要求时对特定业务选择通过卫星转发。网关 3G 网络通信方式选择主要考虑网络通信质量和跳数两个因素, 这里网络通信质量通过往返时延和丢包率来估计<sup>[20]</sup>; 卫星网络选择主要考虑卫星发送负载和节点间的跳数两个因素。

3G 网络的通信质量评估通过如下公式:

$$gw\_quality = \alpha \times total\_rtt + \beta \times \frac{1}{lost} \quad (3-5)$$

公式中  $total\_rtt$  表示此节点通过网关节点接入 Internet 时 3G 网络往返时延,  $\alpha$  表示往返时延的权重值,  $lost$  表示通过网关节点接入 Internet 后 3G 网络的丢包率,  $\beta$  表示丢包率的权重值, 其中  $\alpha + \beta = 1$ 。规定链路质量的阈值为  $gw\_threshold$ , 当  $gw\_quality < gw\_threshold$  时表示不能通过此链路上的 3G 网络接入到 Internet,

节点选择 3G 网关的流程图如图 3-9 所示, 当节点接收到网关通告报文后首先判断发送该网关通过报文的节点是否为当前节点的默认网关节点, 如果是则更新本地保存的网关选择指数。如果不是首先计算链路质量即能否通过该节点的 3G 网络接入 Internet, 当  $gw\_quality$  大于预设阈值时表示能够通过该节点的 3G 网络接入 Internet, 选择网关时首先比较网关跳数, 当网关跳数小于自身保存的跳数时更新默认网关, 当跳数相等时再考虑网关通信质量, 选择网关通信质量好的作为默认网关。

当节点本身的  $gw\_quality$  小于  $gw\_threshold$  时表示节点目前不能通过自身或者周围节点的 3G 网络接入 Internet，进而选择卫星接入 Internet。卫星的选择主要参考卫星发送队列负载和距离网关节点的跳数作为选择指标。卫星网络的评估通过如下公式：

$$gw\_sat\_weight = \alpha \times load + \beta \times metric \quad (3-6)$$

$load$  表示卫星发送队列的负载（通过发送队列长度来计算）， $\alpha$  表示卫星负载的权重值； $metric$  表示网关节点到此节点的跳数， $\beta$  表示跳数的权重值，其中  $\alpha + \beta = 1$ 。这里选择  $\alpha = 0.5$ ， $\beta = 0.5$ ，即两个因素占有同等的地位，首选选择发送负载小的节点作为默认网关，当负载相同时选择距离节点跳数小的节点作为默认网关。

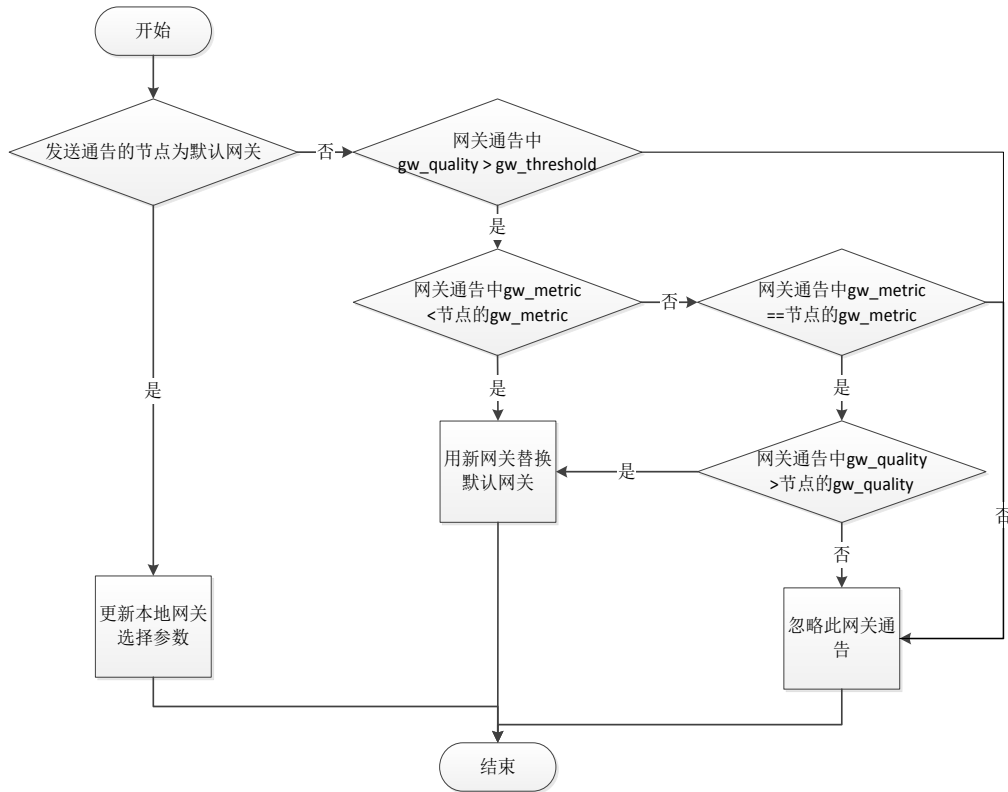


图 3-10 网关选择流程图

### 3.5 无线多模网关组网机制--DSDV 算法改进

本文为了使无线多模网关能够正确组网，并选择合适的网关把无线 Mesh 网络接入 Internet，在 DSDV 路由算法基础上设计了无线多模网关组网机制。由于无线多模网关组成的无线 Mesh 网是静态网络，由于网关节点的静态性，本文在主动式的距离向量路由协议 DSDV 路由协议基础上采用跳数和链路延时作为路

由度量，并把距离网关节点跳数、3G 网络延时、3G 网络丢包率和卫星发送负载作为网关选择度量综合选择最优网关，并根据传输内容的传输代价选择合理的传输方式。

### 3.5.1 路由度量改进

像许多路由算法一样，无线多模网关自组网路由算法会计算邻居节点间的费用。给出两点间的路由度量表示两点间所有链路开销的和。路由算法的目标就是计算到达每个源点  $S$  的具有最小度量的链路。这里的度量可以是节点间的跳数、带宽、链路延迟等因素。DSDV 路由算法采用节点间跳数作为路由度量，但由于无线多模网关组成的无线 Mesh 网中节点间链路的异构性，单纯采用跳数不能准确度量节点间的花销，因此本文在计算路由度量时首先比较节点间的跳数，当节点间跳数相等时再比较节点间的链路时延。扩展的 DSDV 路由协议会周期性的向周围邻居节点广播 HELLO 消息，当邻居节点接收到 HELLO 消息后会回应 HELLO-ACK 消息，通过 HELLO 和 HELLO-ACK 两种消息我们可以计算出两个节点的链路延时。如图 3-10 所示的网络：

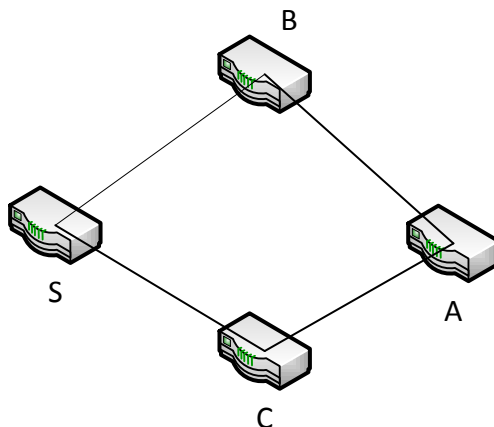


图 3-11 度量计算网络拓扑图

节点会向周围节点广播周期性更新或者触发更新，更新中携带节点到其它节点的路由度量。假设节点  $S$  和节点  $A$  之间的路由度量为  $M(S, A)$ ，节点  $S$  和节点  $B$  之间的度量为  $M(S, B)$ ，节点  $S$  和节点  $C$  之间的度量为  $M(S, C)$ ，节点  $A, B$  之间的跳数为  $HC(B, A)$ ，节点  $A, C$  之间的跳数为  $HC(C, A)$ ，当节点  $A$  接收到节点  $B$  和  $C$  的路由更新，并且根据 HELLO 消息和 IHU 消息探测到节点  $A$  和节点  $B$  之间的链路延迟为  $RTT(A, B)$ ，节点  $A$  和节点  $C$  之间的链路延时为  $RTT(A, C)$ ，那么计算节点  $A$  到节点  $S$  之间的度量的流程图如图 3-11 所示：

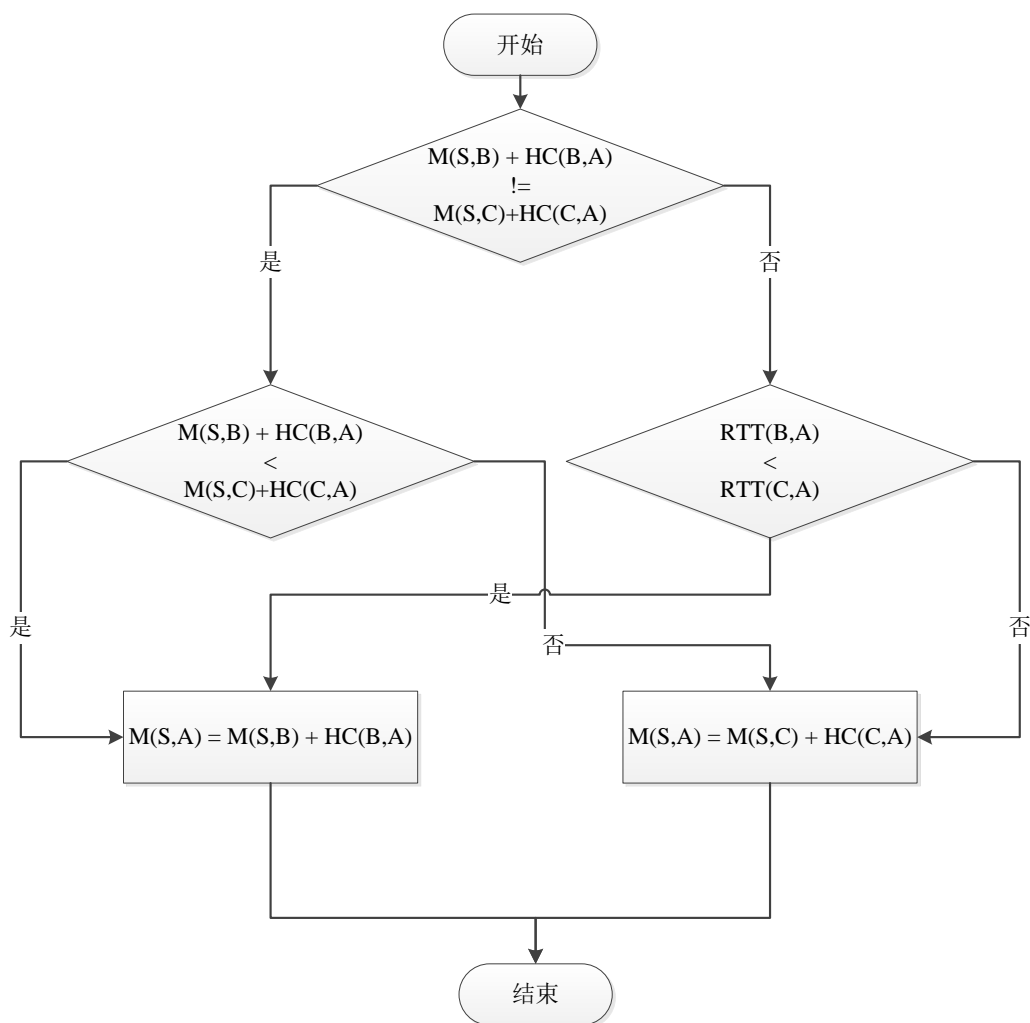


图 3-12 metric 计算流程图

### 3.5.2 路由消息扩展

无线多模网关组网机制主要维护六种消息：路由请求 REQ(REQUEST) 消息、周期性更新 PUT(Periodical Update)消息、触发更新 TUT(Triggered Update)消息、HELLO 消息、HELLO-ACK 消息、网关通告 GW\_BROADCAST 消息。其中 REQUEST 消息用于节点启动时快速建立路由表，PUT 和 TUT 消息主要用于网络中各个节点路由表的更新和维护，HELLO 消息和 HELLO-ACK 消息用于探测节点间的链路之间的延时。GW\_BROADCAST 消息用向周围邻居节点通告自身网关性能。下面详细介绍各个消息格式：

#### 1. REQ 消息格式

REQ 消息是为了新加入网络的节点能够快速学习全网的网络拓扑而定制一种路由消息，该消息只会在节点新加入网络时发送一次，用于请求邻居节点的完整路由表，REQ 消息格式如图 3-12 所示，消息按 8 字节对齐：

	15	31	63
类型	保留	PADDING	

图 3-13 REQ 消息格式

其中“类型”字段为 2 个字节，值为 0，“保留”字段为 2 个字节，用于扩展，“PADDING”字段为 4 个字节，用于字节对齐。

2.PUT 消息格式

0	15	31	63
类型	保留	长度	
目的序号		目的地址	
子网掩码		METRIC	
...			
目的序号		目的地址	
子网掩码		METRIC	

图 3-14 PUT 消息格式

PUT 消息作用有两个：一是配合 REQ 使新加入网络的节点快速学习全网拓扑，二是维护稳定路由表的合法性。PUT 消息内容节点完整的路由表。当节点接收到 REQ 消息后会向发送 REQ 消息的节点单播 PUT 消息，当周期广播定时器超时，节点会对邻居节点广播路由周期广播消息，因为广播节点完整路由表的开销很大，所以 PUT 消息发送频率不会很高。PUT 消息分组格式如图 3-13 所示，在 PUT 消息分组中“类型”字段为 2 个字节，用来表示 PUT 消息，其取值为 1。“保留”字段为 2 个字节，用于协议扩展和字节对齐。“长度”字段占用 4 个字节，用于表示整个报文的长度。下面的 4 个字段用于表示一个完整的路由表项，报文中可能有多个路由表项。“目的序号”是一个 4 字节的整形类型数据，表示当前路由表项的新旧程度，用于解决 Bellman-Ford 算法中路由成环的问题。当序号为偶数时表示合法路由表项，当序号为奇数是表示路由失效。

0	15	31	63
类型	保留	长度	
目的序号		目的地址	
子网掩码		METRIC	
...			
目的序号		目的地址	
子网掩码		METRIC	

图 3-15 TUT 消息格式

3.TUT 消息格式

TUT 消息是为了在网络拓扑发生变化时，加快路由的收敛而定制的一种消

息。消息内容是链路变化的路由表项。格式如图 3-14 所示，其中“类型”字段的值为 2，用于表示路由触发更新消息。其余字段和 PUT 消息格式相同。

#### 4.HELLO 消息

HELLO 消息是一种周期性发送的路由消息，主要用于探测和邻居节点间的链路延时。HELLO 消息的格式如图 3-15 所示：

0	15	31	63
类型	保留	类型	
发送时间		PADDING	

图 3-16 HELLO 消息格式

其中“类型”字段的值为 3 用于表示 HELLO 消息，“发送时间”字段为 4 个字节，其值为 HELLO 消息的发送时间，用于计算节点间的链路延时。

“PADDING”字段为 4 个字节，用于报文对齐。

#### 5.HELLO-ACK 消息

当节点收到周围邻居节点发送的 HELLO 消息时，回给相应的节点回发一个 IHU 消息，用于对 HELLO 探测消息的确认。消息格式如图 3-16：

0	15	31	63
类型	保留	类型	
发送时间		PADDING	

图 3-17 HELLO-ACK 消息格式

其中“类型”字段的值为 4，用于表示 IHU 消息，消息内容和接收到的 HELLO 消息一致。

#### 6.GW\_BROADCAST 消息

GW\_BROADCAST 消息用于向周围邻居节点广播节点网关性能，消息格式如图 3-17 所示：

0	15	31	63
类型	保留	目的序号	
3G网络平均往返时延		3G网络平均丢包率	
卫星发送队列负载		PADDING	

图 3-18 网关通告报文格式



其中“类型”字段占用 4 个字节，值为 5 表示网关通告报文。数据部分为目的的序号，3G 网络的平均往返时延，平均丢包率和卫星发送队列负载，当节点不具有卫星模块时，卫星发送队列负载值为 1。

## 第四章 无线多模网关组网机制实现

### 4.1 无线多模网关组网机制实现总体设计

无线多模网关组网机制具备节点间自组网和网关自适应选择两大功能，程序的架构图如图 4-1 所示：



图 4-1 无线多模网关组网机制架构图

程序主要由路由主控模块、邻居维护模块、内核态路由表维护模块、用户态路由表模块、卫星通信模块、IP 包检测过滤模块、网关决策模块、路由决策模块及日志模块等模块组成。其中路由主控模块负责路由请求 REQ 消息、周期性更新 PUT (Periodical Update) 消息、触发更新 TUT (Triggered Update) 消息、HELLO 消息、HELLO-ACK 消息、网关通告 GW\_BROADCAST 消息的收发、链路通信质量的计算以及路由程序主要逻辑的维护。程序在路由表维护上使用了两种类型的路由表：用户态路由表和内核态路由表，分别由用户态路由表维护模块和内核态路由表维护模块来对其进行操作。邻居维护模块用于维护节点可达的邻居节点的相关信息。路由决策模块实现了改进的 DSDV 路由算法。3G 网络网络探测模块负责对节点的网关出口网络通信质量进行评估，计算出口网络的平均往返时延和平均丢包率。IP 包检测及过滤模块负责根据特定的检测及过滤规则提取出 IP 网络的特定数据包，对突发 IP 数据包进行缓存。卫星模块负责把 IP 包检测及过滤模块提取出的数据包在卫星网络上做透明传输。日志模块负责记录程序执行过程中发起的特定事件有助于出错处理和问题排查。

## 4.2 无线多模网关组网机制实现

为了维护程序的各种全局信息，例如邻居列表、网卡列表、网关信息等，程序利用一个全局的数据结构 `advp` 来保存了程序用到的全局数据结构，具体为：

```
typedef struct advp {
    time_t update_send_timer;
    interface *if_list_head;
    route_entry *m_rtable;//main routing table
    route_entry *adv_rtable;//triggered advertising table
    route_entry *default_re;
    tg_info *tg_info;
    sate_info *sate_info;
    hashmap *neighbor_hp;
} advp;
```

其中 `update_send_timer` 为周期性更新定时器，`if_list_head` 为网卡链表指针，`m_rtable` 为用户态的主路由表，`adv_rtable` 为用户态的触发更新路由表，`default_re` 为节点的默认路由及接入网关的 IP 地址，`tg_info` 为网关的数据结构，`sate_info` 为节点卫星模块的数据结构，`neighbor_hp` 为节点邻居列表。下面详细介绍算法各个模块的具体实现。

### 4.2.1 网卡维护

程序启动时会从配置文件中读入各个网卡的配置信息进而对网卡结构进行初始化，网卡配置文件格式如下：

```
interfaces = (
    {
        if_name = "eth0";
        ip = "10.103.240.200";
        netmask = "255.255.240.0";
        active = 1;
    },
    {
        if_name = "vmnet1";
        ip = "192.168.172.1";
        netmask = "255.255.255.0";
        active = 0;
    },
    {
        if_name = "vmnet8";
```

```

        ip = "192.168.21.1";
        netmask = "255.255.255.0";
        active = 0;
    }
);

```

程序利用了开源的 libconfig 日志解析库来进行配置文件的读取，该日志库支持结构化、层次化的配置，配置文件比 xml 的可读性好，并且更加简洁。读取网卡配置信息后首先把网卡信息保存到网卡结构链表中，网卡结构 interface 如下：

interface 的具体信息如下：

```

typedef struct interface {
    char ifname[IFNAMSIZ];
    uint8_t ifnumber;
    uint8_t active;
    in_addr_t network;//network id
    in_addr_t mask;//netmask
    in_addr_t ip;//ip address
    in_addr_t broadcast;//broadcast address
    int send_fd;
    int recv_fd;
    struct interface * next;
} interface;

```

结构中 ifname 字段为网卡的名称，ifnumber 字段为网卡序号，active 字段用来表示会不会通过网卡来交换路由更新，ip 字段表示该网卡的 ip 地址，netmask 字段表示该网卡的子网掩码，network 字段表示该网卡的网络号，broadcast 字段表示该网卡的广播地址，send\_fd 字段表示用于发送路由更新的套接字描述符，recv\_fd 字段表示用于接收路由更新的套接字描述符，send\_fd 和 recv\_fd 都是 udp 套接字。next 为指向网卡链表中下个网卡结构的指针。

程序从配置文件读取网卡的配置信息后把网卡信息封装到 ifreq 结构中，通过 Linux 的系统调用 ioctl 设置节点的网卡信息。具体的函数结构如下：

```

int ioctl (int fd, ind cmd, ...);

```

利用 IOCTL 函数可以对利用设备驱动程序对设备的 I/O 通道进行管理。这里利用 IOCTL 函数对 Linux 网络进行配置，在对网卡配置信息的过程中利用了套接字和套接字地址结构。套接字(Socket)是支持 TCP/IP 协议中网络通信的基本操作单元，可以看做是不同主机间进程进行双向通信的端点。套接字主要分一下 3 中类型：

## (1) 流套接字 (SOCK\_STREAM):

流套接字用于提供面向连接、可靠的数据传输服务。该服务能够保证数据实现无差错、无重复发送，按序接收。流套接字之所以能够实现可靠的数据服务，在于其使用了传输控制协议，即 TCP (The Transmission Control Protocol) 协议

## (2) 数据报套接字 (SOCK\_DGRAM)

数据报套接字提供了一种无连接的服务。该服务不能提供可靠的数据传输服务，数据有可能在传输过程中丢失或出现重复数据或者被修改，并且无法保证按序到达。

## (3) 原始套接字 (SOCK\_RAW)

原始套接字与标准套接字的区别在于：原始套接字可以读写内核没有处理的 IP 数据包，而流套接字只能读取 TCP 协议的数据，数据报套接字只能读取 UDP 协议的数据。

定义 IP 协议的套接字的方法如下：

```
int sockfd;
sockfd = socket(AF_INET, SOCK_DGRAM, IPPROTO_IP);
```

套接字地址结构 `sockaddr_in` 用于保存 Internet 上的网络地址，该结构位于 `<netinet/in.h>` 文件中，具体为：

```
struct sockaddr_in
{
    short int sin_family; /* 协议簇 */
    unsigned short int sin_port; /* 端口号 */
    struct in_addr sin_addr; /* 网络地址 */
    unsigned char sin_zero[8]; /* 填充字段 */
};
```

通过 `ioctl` 配置网卡的 IP 地址的操作具体如下：

```
struct ifreq ifr;
struct sockaddr_in address;
strncpy(ifr.ifr_name, cif->ifname, sizeof(ifr.ifr_name));
memcpy(&ifr.ifr_addr, &address, sizeof(struct sockaddr));
ioctl(fd, SIOCSIFADDR, &ifr);
```

通过 `ioctl` 配置网卡的子网掩码的具体操作如下：

```
struct ifreq ifr;
struct sockaddr_in address;
address.sin_addr.s_addr = cif->mask;
memcpy(&ifr.ifr_addr, &address, sizeof(struct sockaddr));
ioctl(fd, SIOCSIFADDR, &ifr)
```

设置完网卡的 IP 地址、子网掩码后要给每个被标记为活动的网卡分配发送套接字和接收套接字用户路由更新的扩散和接收。接收套接字会绑定网卡的广播地址，接收所有发送到广播端口的路由更新信息，具体实现如下：

```
struct sockaddr_in local_server;
recv_fd = socket(AF_INET, SOCK_DGRAM, 0);
local_server.sin_family = AF_INET;
local_server.sin_addr.s_addr = cif->broadcast;
local_server.sin_port = htons(RIP_BROADCAST_PORT);
ret = bind(recv_fd, (struct sockaddr *) &local_server, sizeof(local_server));
```

发送套接字要开启套接字的广播选项，才能利用该套接字广播路由更新信息，具体如下：

```
int on = 1;
int res;
send_fd = socket(AF_INET, SOCK_DGRAM, 0);
ret = setsockopt(send_fd, SOL_SOCKET, SO_BROADCAST, &on, sizeof(int));
```

#### 4.2.2 路由表维护

无线多模网关组网机制路中有两种类型的路由表：用户态路由表和内核态路由表。用户态路由表是由路由程序在用户态维护并进行相应的更新操作，内核态路由表由 Linux 内核管理，在节点通过 TCP/IP 协议通信时，当有 IP 包到来会查询内核态路由表确定下一跳路由器进行转发。下面将详细介绍用户态路由表和内核态路由表维护的相关操作。

##### 1. 用户态路由表维护

用户态路由表维护主要是对用户态路由表进行插入、删除、修改操作。由于用户态路由表本质上是一个单向链表，用户态路由表结构 `route_entry` 具体如下：

```
#define VALID_ROUTE_ENTRY 0
#define INVALID_ROUTE_ENTRY 1
```

```
typedef struct route_entry {
    uint16_t seqno;
    in_addr_t dst;
    in_addr_t netmask;
    in_addr_t gateway;
    uint32_t metric;
    uint8_t flags;
    time_t expire_timer;
    int ifnumber;
    struct route_entry *next;
} route_entry;
```

route\_entry 结构为用户态路由表结构，结构中 seqno 字段表示路由条目的目的序号，由源节点维护，当目的序号为偶数时表示合法的路由条目，当目的序号为奇数时表示无效的路由条目。dst 为路由条目的目的 ip 地址，netmask 目的地址的子网掩码，gateway 为下一跳路由器地址，metric 为路由度量。flags 字段为路由有效标志，路由条目有两种状态 VALID\_ROUTE\_ENTRY 为合法的路由表项，INVALID\_ROUTE\_ENTRY 为无效路由表项。Expire\_timer 是路由表项的超时定时器，当定时器超时时路由表项会被标记为无效。Ifnumber 为接收到此路由表项的网卡编号，当为本地路由时置为-1。next 字段为指向下一个路由条目的指针。

在向用户态路由表插入路由条目节点时向链表头部插入，时间复杂度为  $O(1)$ ，插入节点后还要向内核态路由表插入相应路由表项。其中 rtable 为要操作的路由表，new\_re 为要插入的路由表项，具体操作如下：

```
int re_list_add(route_entry *rtable, route_entry *new_re)
{
    int res;
    new_re->next = rtable;
    rtable = new_re;

    res = kernel_route(ROUTE_ADD, new_re, NULL);
    return res;
}
```

用户态路由表删除首先调用内核态路由表删除操作，然后对用户态路由表分两种情况操作：当只路由表中只有一个节点时直接删除，否则在路由表中找到要删除的节点后用当前节点的下一个节点的数据代替当前节点，然后删除下一个节点。其中 cur\_re 为要删除的路由表项，具体操作如下：

```
int re_list_delete(route_entry *cur_re)
{
```

```

assert(cur_re != NULL);
int res;

res = kernel_route(ROUTE_DEL, cur_re, NULL);
if (cur_re->next == NULL) {
    free(cur_re);
    cur_re = NULL;
} else {
    route_entry *next_re = cur_re->next;
    memcpy(cur_re, next_re, sizeof(route_entry));
    cur_re->next = next_re->next;
    free(next_re);
}
return res == 0 ? 0 : -1;
}

```

用户态路由表更新时用新的路由表项的数据替换旧的路由表项的数据，其中 `old_re` 为旧路由表项，`new_re` 为新路由表项，具体操作如下：

```

int re_list_modify(route_entry *old_re, route_entry *new_re)
{
    int res;

    res = kernel_route(ROUTE_MOD, old_re, new_re);
    copy_route_entry(old_re, new_re);
    return res;
}

```

## 2.内核态路由表维护

由于内核态路由表位于 Linux 内核，如果想在用户态操作内核态的路由表需要使用相应的进程间通信机制，这里选择 `netlink` 套接字来实现用户进程和内核进程间的进程间通信(IPC)，`netlink` 套接字可以使用标准套接字 APIs 来创建，Linux 系统下提供了一套通过 `netlink` 套接字来操作内核态路由表的接口。路由程序在 `netlink` 套接字基础上程序封装了一个操作内核态路由表的函数如下：

```

#define ROUTE_ADD 0
#define ROUTE_DEL 1
#define ROUTE_MOD 2
int kernel_route(int command, route_entry *re, route_entry *new_re);

```

对内核态路由表操作同样有插入、删除和修改三种操作，三种操作可以通过这一个函数统一完成，其中 `command` 参数表示这个具体操作类型可以是 `ROUTE_ADD`, `ROUTE_DEL` 和 `ROUTE_MOD` 中的一种，`re` 为旧的路由表项，`new_re` 为新的路由表项。



## 4.2.3 路由主控模块

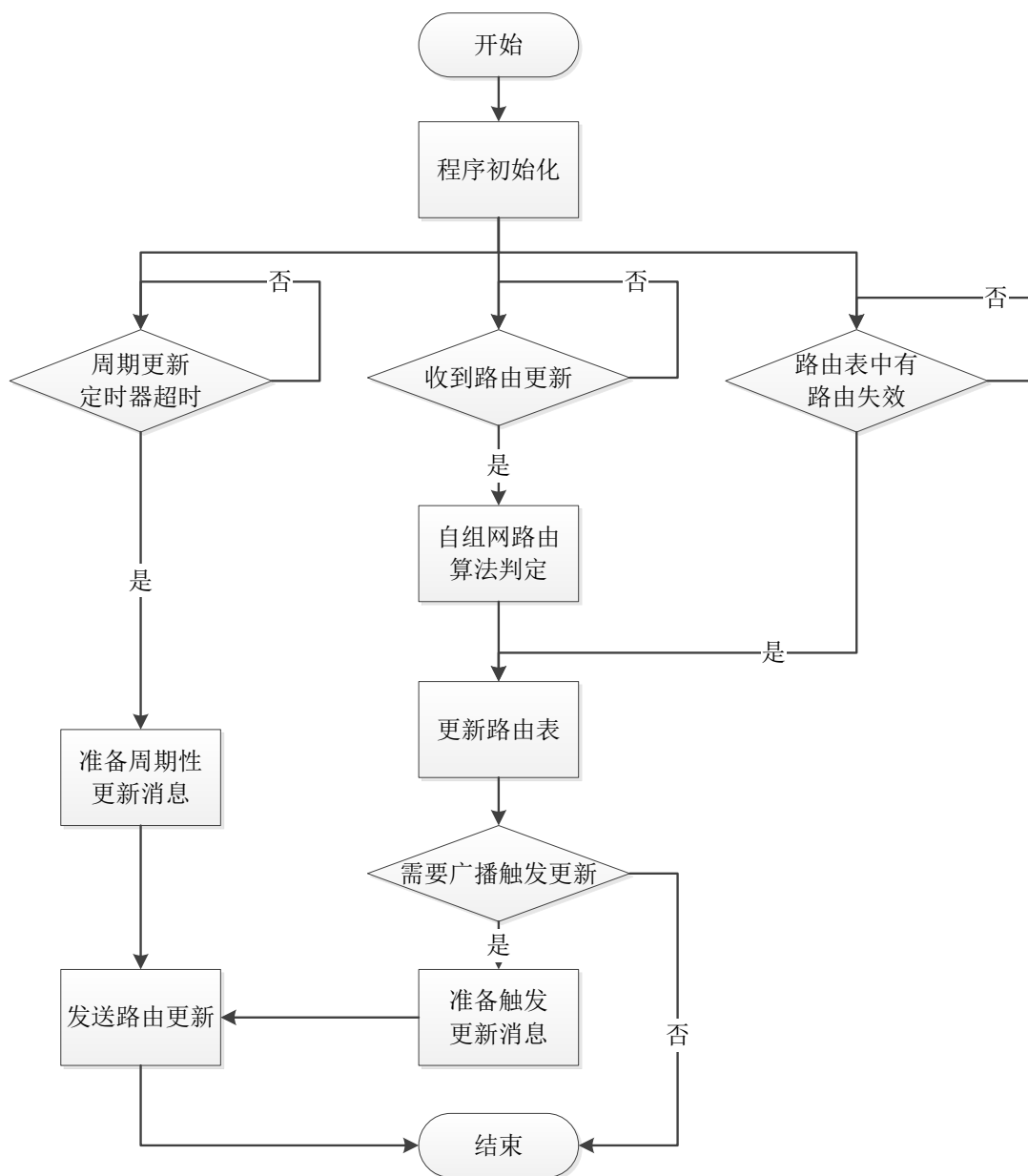


图 4-2 路由主控模块流程图

路由主控模块负责路由请求 REQ 消息、周期性更新 PUT (Periodical Update) 消息、触发更新 TUT (Triggered Update) 消息、HELLO 消息、HELLO-ACK 消息、网关通告 GW\_BROADCAST 消息的收发、链路通信质量的计算、过期路由表清除以及路由程序主要逻辑的维护流程图如图 4-2 所示, 首先程序启动时会对程序进行初始化, 包括初始化网卡结构链表, 初始化用户态主路由表, 初始化用户态触发更新路由表, 初始化邻居列表, 初始化周期更新定时器等。程序运行时会同时处理 IO 时间和时间事件, IO 事件是程序会接收来自邻居节点的周期性更新消息 PUT (Periodical Update)、HELLO 消息、HELLO-ACK 消息, 对特定消息做特定处理。时间事件是周期性路由更新和主路由表中各个路由表项的合法性

检测，当周期性路由更新事件被触发后，程序会向周围邻居节点广播特定的周期性更新消息，当某个路由表项超时定时器超时会把该路由表项标记为无效，产生触发更新，并从主路由表删除该路由表项。

为了统一处理 IO 事件和定时事件，程序实现时利用了 Linux 系统提供的 IO 多路复用机制 EPOLL，EPOLL 是 Linux 内核为处理大量文件描述符而做出改进的 IO 多路复用机制，能够显著提高程序在大量并发中只有少量活跃情况下系统 CPU 利用率。EPOLL 函数族位于<sys/epoll.h>文件中，主要提供了一下函数接口：

```
int epoll_create(int size);
int epoll_ctl(int epfd, int op, int fd, struct epoll_event *event);
int epoll_wait(int epfd, struct_event *events, int maxevents, int timeout);
```

epoll\_create 函数用于生成 epoll 专用的文件描述符，其中参数 size 表示能在该描述符上关注的套接字描述符的最大数据。Epoll\_ctl 函数用于在 epfd 上注册相应的事件：添加、修改、删除。epoll\_wait 函数用于等待 epfd 上的 IO 事件或者时间事件触发。其中 events 为触发的 IO 事件链表，timeout 为时间事件的超时时间。

利用 EPOLL 函数族来处理 IO 事件和时间事件的伪代码如下：

```
BEGIN
创建EPOLL描述符
创建关注事件的事件链表
添加关注事件，包括IO事件、定期更新超时事件、路由表项超时事件
计算所有超时事件的最近超时时间，把该时间作为epoll_wait超时时间
WHILE（没有关注事件触发）
IF（触发事件是IO事件）{
    读取更新报文
    IF（报文是周期性路由更新消息）
        处理周期性路由更新消息
    ELSE IF（报文是触发路由更新消息）
        处理触发路由更新消息
    ELSE IF（报文时HELLO报文）{
        准备HELLO-ACK报文
        给发送HELLO报文的节点单播HELLO-ACK报文
    }
} ELSE IF（触发事件是超时事件）{
    IF（是定期更新事件超时）{
        准备定期更新报文
        通过每个被标记为active的网卡发送该报文
    } ELSE IF（是定期HELLO广播超时）{
        准备HELLO报文
    }
}
```

```

        通过每个被标记为active的网卡发送该报文
    } ELSE IF (路由表项超时) {
        路由表项标记为无效
        准备触发更新报文
        通过所有标记为active的网卡发送该报文
        删除该路由表项
    }
}END

```

路由主控模块负责路由请求 REQ 消息、周期性更新 PUT 消息、触发更新 TUT 消息、HELLO 消息、HELLO-ACK 消息和网关通告 GW\_BROADCAST 消息的发送，下面详细介绍这 6 种消息的发送。

### 1. 路由请求消息发送

当节点加入网络后为了快速学习网络拓扑会向周围邻居节点发送路由 REQ 消息，REQ 消息用 req\_packet 结构表示，具体如下：

```

typedef struct req_packet {
    uint16_t type;
    uint16_t reserved;
    uint32_t padding;
} req_packet;

```

其中 type 字段表示报文类型，其值为 0，reserved 字段留作扩展使用，padding 字段用于字节对齐。

### 2. 周期性更新和触发更新消息发送

路由周期性更新和触发更新消息用 route\_packet 结构表示，route\_packet 结构具体如下：

```

#define PUT 0
#define TUT 1
typedef struct route_packet {
    uint16_t type;
    uint16_t reserved;
    uint32_t length;
    rte routes[0];
} route_packet;

```

type 字段标示了报文的类型，报文共有 2 种类型分别为：周期性更新消息 PUT (Periodical Update)、触发更新消息 TUT (Triggered Update)。reserved 字段为保

留字段，用于以后协议扩展。**length** 字段表示整个报文长度，用于计算路由表条目个数。**routes** 字段是一个不占用空间的数组，里面是路由条目结构。

路由条目结构如下：

```
typedef struct rte {
    uint32_t seqno;
    in_addr_t dst;
    in_addr_t netmask;
    uint32_t metric;
} rte;
```

**seqno** 字段为路由条目的序号和 **route\_entry** 结构中的目的序号相同。**dst** 字段为路由表项的目的地址，**netmask** 字段是目的地址的子网掩码，**metric** 为路由度量。

当周期性定时器超时时会触发周期性更新事件，节点会向周围邻居节点广播自己整个主路由表，周期性更新的伪代码如下：

```
BEGIN
准备周期性更新报文
遍历主路由表
    IF（路由有效）{
        IF （路由表项由该节点产生）{
            路由表项目的序号 += 2
        } ELSE {
            路由表项目的序号 += 1
        }
    }
    把路由表项的各个数据项赋值给路由更新报文
    通过每个被标记为active的网卡发送该报文
END
```

当链路状态改变时会触发触发更新事件，这里定义链路状态改变主要是路由表项的 **metric** 值变化和链路断开。触发更新的伪代码如下：

```
BEGIN
准备触发更新报文
遍历触发更新路由表
    IF（路由有效）{
        IF （路由表项由该节点产生）{
            路由表项目的序号 += 2
        } ELSE {
            路由表项目的序号 += 1
        }
    }
```

```

    }
}
把路由表项的各个数据项赋值给路由更新报文
通过每个被标记为active的网卡发送该报文
END

```

### 3. HELLO 消息发送

当 HELLO 消息定时器超时，触发 HELLO 消息发送事件，HELLO 消息用结构 `hello_packet` 表示，具体如下：

```

typedef struct hello_packet {
    uint16_t type;
    uint16_t reserved;
    uint32_t send_time;
} hello_packet;

```

伪代码如下：

```

BEGIN
准备HELLO消息报文
写入当前时间
通过每个被标记为active的网卡发送该报文
END

```

#### 4.2.4 路由决策模块

程序接收到的路由消息更新消息后会判断是否更新节点的路由表，具体操作用伪代码表示如下：

```

BEGIN
    WHILE (路由更新中还有未处理的路由条目) {
        IF (主路由表中没有此路由条目) {
            IF (目的序号是偶数) {
                在主路由表中加入此路由条目
                在触发更新路由表中加入此路由条目
            }
        } ELSE {
            IF (触发更新路由表中没有此路由条目) {
                求触发更新路由表中加入此路由条目为advTableEntry
                主路由表中对应的路由条目为fwdTableEntry
            }
            IF (目的序号是偶数) {
                IF (目的序号>advTableEntry的目的序号) {

```

```

        IF (metric != advTable.metric) {
            更新主路由表中的路由条目
            更新触发更新路由表中的路由条目
        } ELSE {
            更新主路由表和触发更新路由表中的条目
        }
    } ELSE {
        //接收到一个旧的路由更新
        删除触发更新路由表中相应的路由条目
    }
} ELSE {
    //目的序号为奇数说明路由链路断开
    IF（发送者是advTableEntry的下一跳路由器）{
        找到主路由表中所有以advTableEntry中dst为下一跳的路由条目，
        把它们的目的序号+1
        加入触发更新路由表
        把主路由表中相应的路由条目删除
    } ELSE {
        删除触发更新路由表中相应的路由条目
    }
}
}
}
}
触发更新调度
END
```

#### 4.2.5 3G 网络探测模块

3G 网络探测模块通过构建到网络上特定主机 ICMP 包,统计 ICMP 包在节点和主机间的往返时延和丢包率来实现对出口网络通信质量的评估。ICMP 是 (Internet Control Message Protocol) Internet 控制报文协议,用来在 IP 主机、路由器间传递控制消息。3G 探测模块为了实现 ICMP 报文的定期收发以及往返时延和丢包率的统计,采用了两个线程分别用于 ICMP 报文发送和接收。

## 1.探测包构建与解析

### (1) 探测包构建

ICMP 的数据结构定义于 Linux 系统<netinet/ip\_icmp.h>文件中，具体结构如下：

```

struct icmp{
    u_int8_t  icmp_type;   /* type of message, see below */
    u_int8_t  icmp_code;   /* type sub code */
    u_int16_t  icmp_cksum;  /* ones complement checksum of struct */
    .....
    #define icmp_id    icmp_hun.ih_idseq.icd_id
    #define icmp_seq    icmp_hun.ih_idseq.icd_seq
    u_int8_t   id_data[1]
}

```

在构建 ICMP 探测报文时主要用到上面五个变量，icmp\_type 字段表示 ICMP 包的类型，我们把该字段设置为 ICMP\_ECHO，当 Internet 上主机接收到该 ICMP 包后会复制 ICMP 包的 id\_data 字段然后发送一个回应报文给节点。icmp\_code 字段是报文代码，这里把该字段设置为 0，表示该 ICMP 报文为请求报文。icmp\_id 字段为发送 ICMP 报文的进程号，以区别不同应用的 ICMP 探测包。icmp\_seq 字段为 ICMP 报文的序号，当 Internet 上主机接收到 ICMP\_ECHO 类型的 ICMP 报文后会回应一个具有同样 icmp\_seq 的 ICMP 包，该字段可以区分 ICMP 包的先后顺序。id\_data 字段是 ICMP 包携带的数据，为了计算节点和 Internet 上主机间的往返时延，在发送 ICMP 探测包时把当前的时间写入 id\_data 字段，当网络上主机收到该探测报文后会复制该字段并构建一个回应报文，节点收到此回应报文后根据其中的 id\_data 字段便可以计算出节点和 Internet 主机间的往返时延。icmp\_cksum 字段为 ICMP 报文的校验和，计算校验和的算法如下：

```

unsigned short cal_chksum(unsigned short *addr,int len)
{
    int nleft = len;
    int sum = 0;
    unsigned short *w = addr;
    unsigned short check_sum = 0;
    while(nleft>1)    //ICMP 包头以字（2 字节）为单位累加
    {
        sum += *w++;
        nleft -= 2;
    }
    if(nleft == 1)    //ICMP 为奇数字节时，转换最后一个字节，继续累加
    {
        *(unsigned char *)&check_sum = *(unsigned char *)w;
        sum += check_sum;
    }
    sum = (sum >> 16) + (sum & 0xFFFF);
    sum += (sum >> 16);
    check_sum = ~sum;    //取反得到校验和
}

```

```

return check_sum;
}

```

构建 ICMP 探测包的过程就是正确填充把上述 ICMP 结构，构建序号为 pack\_no 的 ICMP 探测包的代码如下：

```

int pack(int pack_no, char *sendpacket)
{
    int packsize;
    struct icmp *icmp;
    struct timeval *tval;
    icmp = (struct icmp*)sendpacket;
    icmp->icmp_type = ICMP_ECHO;    //ICMP_ECHO 类型的类型号为 0
    icmp->icmp_code = 0;
    icmp->icmp_cksum = 0;
    icmp->icmp_seq = pack_no;    //发送的数据报编号
    icmp->icmp_id = pid;

    packsize = 8 + DATA_LENGTH;    //数据报大小为 64 字节
    tval = (struct timeval *)icmp->icmp_data;
    gettimeofday(tval, NULL);    //记录发送时间
    //校验算法
    icmp->icmp_cksum = cal_chksum((unsigned short *)icmp, packsize);
    return packsize;
}

```

## (2) 探测包解析

当包解析线程收到 Internet 主机对 ICMP 探测包的回应后会对回应包进解析，探测包解析的代码如下：

```

int unpack(char *buf, int len)
{
    int iphdrlen;
    struct ip *ip;
    struct icmp *icmp;
    struct timeval *tvsend;
    double rtt;
    struct timeval tvrecv;

    ip = (struct ip *)buf;
    iphdrlen = ip->ip_hl << 2;
    icmp = (struct icmp *) (buf + iphdrlen);
    len -= iphdrlen;
    if( (len > 8) && (icmp->icmp_type == ICMP_ECHOREPLY) && (icmp->icmp_id == pid))

```



```

{
    gettimeofday(&tvrecv, NULL);
    packet_rcv++;
    tvsend = (struct timeval *)icmp->icmp_data;
    tv_sub(&tvrecv,tvsend);

    rtt = tvrecv.tv_sec*1000 + tvrecv.tv_usec/1000.0;
    assert(icmp->icmp_seq >= 0 && icmp->icmp_seq < MAX_PACKET_NUM);
    total_rtt[icmp->icmp_seq] = rtt;
}
return -1;
}

```

该函数收到 ICMP 回应包后首先判断数据包是否完整，然后检查 ICMP 包的类型是否为 ICMP\_ECHOREPLY 和是否为本进程发送的 ICMP 探测报文，然后去计算探测的往返时延和丢包率。

## 2.探测包的收发

探测包收发分别有两个线程来实现，发送 ICMP 报文可以通过原始套接字来实现，定义发送 ICMP 报文的原始套接字方法如下：

```

int sockfd;
sockfd=socket(AF_INET, SOCK_RAW, IPPROTO_ICMP);

```

其中 SOCK\_RAW 表示该套接字为原始套接字，IPPROTO\_ICMP 表示该套接字使用 ICMP 通信协议。为了向特定目的地发送 ICMP 报文，需要定义 ICMP 报文的地址结构 sockaddr\_in,具体为：

```

struct sockaddr_in
{
    short int sin_family; /* 协议簇 */
    unsigned short int sin_port; /* 端口号 */
    struct in_addr sin_addr; /* 网络地址 */
    unsigned char sin_zero[8]; /* 填充字段*/
};

```

其中 sin\_family 要设置为 AF\_INET,表示为 Internet 协议族，端口设置为 0，网络地址设置为 Internet 上目的主机的 IP 地址。发送 ICMP 报文需要通过 send\_to 函数，具体如下：

```

int sendto (int sockfd, const void *msg, int len unsigned int flags, const struct sockaddr *to ,int tolen);

```

其中 `sockfd` 为原始套接字描述符, `msg` 为构建的 ICMP 探测报文, `flags` 为默认值, `to` 为发送目的地, `tolen` 为地址结构长度。

ICMP 报文接收线程采用 Linux 下 IO 多路复用函数 `select` 来实现对报文的超时检测, `select` 函数原型如下:

```
int select(int nfd, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);
```

其中 `nfd` 为要监视的套接字描述符最大值加 1, `readfds` 为监视的可读描述符集合, `writefds` 为监视的可写描述符集合, `exceptfds` 为监视的出现异常的描述符集合, `timeout` 为超时时间。`select` 函数的返回值共有三种: 0 表示在超时时间内没有可读、可写或者异常事件发生; -1 表示出错; 整数表示有可读、可写或者异常事件。本文把 `timeout` 值设置为 3 秒, 表示如果 3 秒内没有收到 ICMP 应答包就表示节点和 Internet 主机间链路断开, 即为一次丢包。当返回值为整数时接收此应答包, 接收应答包通过 `recvfrom` 函数, 具体如下:

```
int recvfrom(int sockfd, void *buf, int len, unsigned int flags, struct sockaddr *from, int *fromlen);
```

### 3. 平均往返时延和平均丢包率计算

#### (1) 平均往返时延计算

平均往返时延被定义为节点到网络上多个主机间往返时延的平均值, 要计算平均往返时延需要分别计算节点到每个网络上主机间的往返时延。节点每次向网络上主机发送 `MAX_PACKET_NUM` 个 ICMP 请求包, 根据接收到的 `RECV_PACKET_NUM` 个应答包来计算节点和 Internet 主机间的往返时延。具体过程如下:

```
double compute_rtt()
{
    int i;
    double sum = 0.0;
    for (i = 0; i < RECV_PACKET_NUM; i++) {
        sum += tmp_rtt[i];
    }
    return (sum / RECV_PACKET_NUM);
}

double avg_rtt()
{
    int i;
```

```

double sum = 0.0;
for (i = 0; i < MAX_HOST_NUM; i++) {
    sum += total_rtt[i];
}
return (sum / MAX_HOST_NUM);
}

```

## (2) 平均丢包率计算

平均丢包率被定义为节点到网络上多个主机间链路的丢包率平均值，要计算平均丢包率需要分别计算节点到每个网络主机间链路的丢包率，节点每次向 Internet 主机上发送 MAX\_PACKET\_NUM 个 ICMP 请求包，再根据接收到的 RECV\_PACKET\_NUM 个应答包来计算节点和 Internet 主机间链路的丢包率。具体过程如下：

```

double compute_lost()
{
    return tmp_recv / MAX_PACKET_NUM * 1.0
}
int compute_rtt()
{
    int i;
    double sum = 0.0;
    for (i = 0; i < MAX_HOST_NUM; i++) {
        sum += total_recv[i];
    }
    return (sum / MAX_HOST_NUM);
}

```

### 4.2.6 卫星转发

卫星转发 IP 数据报的功能主要由程序的三个模块组成：报文检测过滤模块、网关端卫星模块和汇聚节点端卫星模块。报文检测过滤模块负责根据特定的过滤规则对特定网卡的 IP 包进行检测，过滤出特定的 IP 包缓存到文件中。报文检测过滤模块和网关端卫星模块间共享通信管道、请求队列和应答队列。网关端卫星模块和汇聚节点端卫星模块负责网关节点和 Internet 上主机间的通信，卫星转发的流程图如图 4-3 所示：

下面详细介绍卫星转发流程中涉及到的报文检测过滤模块、网关端卫星模块和汇聚节点端卫星模块的具体实现。

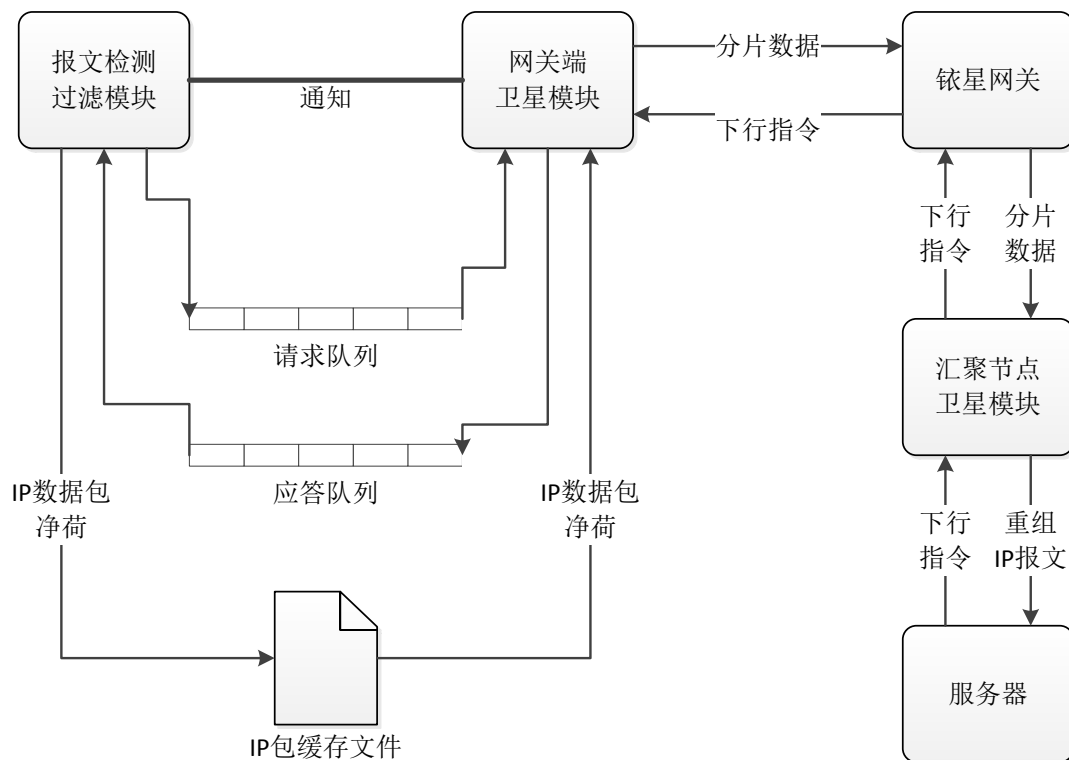


图 4-3 卫星转发数据流图

### 1. 报文检测过滤模块实现

报文监测及过滤模块利用 `libpcap` 库来实现对 IP 报文的抓取，`libpcap` 函数库中常用函数位于 `<pcap.h>` 头文件下，具体如下：

```
pcap_t *pcap_open_live(char *device, int snaplen, int promisc, int to_ms, char *ebuf);
```

该函数用于获得用于捕获网络数据包的数据包捕获描述符。其中 `device` 参数为要监听的网络设备名称。`snaplen` 参数定义捕获数据的最大字节数。`promisc` 参数指定是否将网卡置于混杂模式，只有把网卡置于混杂模式才能监听到那些目的地址不是自己的数据包。`to_ms` 参数指定超时时间。`ebuf` 参数用于传递错误信息。

```
int pcap_loop(pcap_t *p, int cnt, pcap_handler callback, u_char *user);
```

该函数用于在打开的数据包捕获描述符对应的网卡设备上捕获 `cnt` 个数据包，并调用 `callback` 函数。其中参数 `p` 为数据包捕获描述符，`cnt` 为要捕获包的个数，当 `cnt` 为负数时 `pcap_loop` 函数将循环运行，`callback` 为回调函数，用户可以在回调函数中对捕获的数据包做特定处理，`user` 为给 `callback` 函数传入的参数。

```
int pcap_compile(pcap_t *p, struct bpf_program *fp, char *str, int optimize, bpf_u_int32 netmask)
```

该函数用于把参数 `str` 指定的过滤规则编译到 `bpf_program` 类型的指针变量中。`optimize` 参数控制编译的优化程度。`netmask` 参数指定本地网络的子网掩码。

```
int pcap_setfilter(pcap_t *p, struct bpf_program *fp)
```

该函数用于把由 `pcap_compile` 函数编译生成的二进制过滤器安装到参数 `p` 指定的包捕获描述符上。

```
void pcap_close(pcap_t *p)
```

该函数用于关闭包捕获描述符 `p`，停止对网卡的数据包检测并释放资源。

通过 `libpcap` 函数库进行 IP 包过滤的流程如图 4-3 所示：



图 4-4 libpcap 函数库抓包流程

要想实现对 IP 数据包的正确过滤需要设置合理的过滤规则，这里过滤规则分为两部分：监听网卡和包捕获过滤器。网卡参数是通过 `pcap_open_live` 函数传入，指定要监听的一系列网卡，包捕获过滤器是通过给函数 `pcap_compile` 传入恰当的过滤规则字符串，由于本文中卫星只会转发 `udp` 数据报，根据 `udp` 数据包的目的 IP 和目的端口可以对传输业务进行区分从而过滤出特定的 IP 数据包。包捕获过滤器设置规则如下：

```
rule = "udp and dst xxx.xxx.xxx.xxx and port xxxx"
```

包检测及过滤模块在过滤出特定 IP 包后取出 IP 包净荷再根据缓存文件的格式把 IP 包缓存到文件中，然后把文件名放入请求队列，并在通信管道上发送消息通知网关端卫星模块处理请求。由于 libpcap 函数库过滤出的 IP 包是完整的以太网帧，要想取得 IP 包净荷需要根据以太网帧结构、IP 包结构和 UDP 包结构层层剥离出相关的数据。首先从以太网帧中找到 IP 包头，从 IP 包头中读出目的 IP 和 IP 包头长度，进而找到 UDP 包结构，读取其中的目的端口号和 UDP 包头长度，进而可以找到 UDP 包净荷部分。整体伪代码如下：

```
BEGIN
    获取LIBPCAP捕获的以太网帧
    根据以太网帧类型类型确定以太网帧头长度
    跳过以太网帧头部找到IP包结构部分
    读取IP包目的IP和IP包头长度
    跳过IP包头找到UDP包结构部分
    读取UDP包目的端口和净荷长度
    创建缓存文件，把目的IP、目的端口、净荷长度和净荷写入缓存文件
    把缓存文件名加入请求队列
    通过通信管道通知mlink端卫星模块
END
```

## 2. 网关端卫星模块实现

网关端卫星模块负责主要实现两个功能：一是处理来自包过滤检测模块的分片发送请求，二是处理服务器端发送的下行控制指令。由于网关端卫星模块要同时监听通信管道和卫星串口两个设备，本文采用了 Linux 下 IO 多路复用函数接口 select 来实现对通信管道和卫星串口可读事件的监测。当通信串口可读时表明包检测过滤模块抓取了 IP 包请求卫星模块发送，卫星模块从请求队列里取出缓存文件名检索到缓存文件，并按照卫星 SBD 分片的格式对缓存文件分片，最后通过铱星提供的 AT 指令集来操纵铱星模块来发送 SBD 分片报文；当卫星串口可读时，从串口中读出命令字，如果命令字是“SBD RING”，表明是铱星网关采用自动通知机制通知卫星模块网关有数据要转发给卫星模块。通过相应的 AT 指令集控制卫星模块读取下行控制指令，再根据控制指令中的元信息把控制指令转发到相应节点。具体伪代码如下：

```
BEGIN
    等待IO事件
    IF (通信管道可读) {
        取出管道命令字
        从请求队列取出缓存文件名
```

```

    读取缓存文件
    WHILE (缓存文件可以分片) {
        缓存文件分片
        缓存文件转换成SBD卫星分片
        AT指令集操纵卫星模块发送
    }
} ELSE IF (卫星串口可读) {
    从串口读取命令字
    IF (命令字 == "SBDRING") {
        从铱星网关拉取下行控制指令
        根据控制指令中元信息转发下行控制指令
    }
}
END

```

通过 AT 指令集可以控制铱星终端收发 SBD 数据，常用的 AT 指令集如下：

#### (1) AT+SBDWB=<SBD message length>

该命令用于把二进制数据写入卫星终端，SBD message length 表示 SBD 消息的长度，不包括两个字节的校验和，SBD 消息的最大长度为 340 字节。命令输入后如果 SBD 消息长度合法会返回 ascii 编码的字符串“READY<CR><LF>”。接下来便可以向终端写入二进制数据和校验和。校验和共有两个字节，计算方法是对数据部分按字节求和，求和结果的高字节为校验和第一个字节，低字节为第二个字节。

#### (2) AT+SBDIX

该命令用于初始化卫星终端和铱星网关间的一个扩展 SBD 传输会话，当卫星终端有 SBD 消息发送时会把该消息投递到铱星网关，当铱星网关有消息转发给卫星终端时会拉取一条消息到卫星终端。该命令的返回值如下：

+SBDIX:<MO status>, <MOMSN>, <MT status>, <MTMSN>, <MT length>, <MT queued>

其中<MO status>表示卫星终端向铱星网关投递消息的状态，0，1，2 表示发送成功，否则表示传送失败。<MOMSN>表示该消息的序号。<MT status>表示铱星网关向卫星终端投递消息状态，0 表示没有消息，1 表示从铱星网关接成功接收消息，2 表示接收消息失败。<MTMSN>表示从铱星网关接收消息的序列号。<MT length>表示消息长度。<MT queued>表示有多少消息在铱星网关等待接收。

#### (3) AT+SBDREG

该命令用于铱星终端向网关注册自动通知服务，即如果铱星网关有消息要发送给铱星终端时会通过自动通知机制事先通知铱星终端。

在对大数据进行分片处理后可以由铱星终端依次通过铱星网关投递分片，而且当服务器通过铱星网关要给铱星终端发送消息时，铱星网关会通知铱星终端接收消息，通过铱星提供的上述 AT 指令集控制铱星终端便能完成以上功能，注册消息自动通知、收发消息的流程图如下：

### (1) 注册消息自动通知

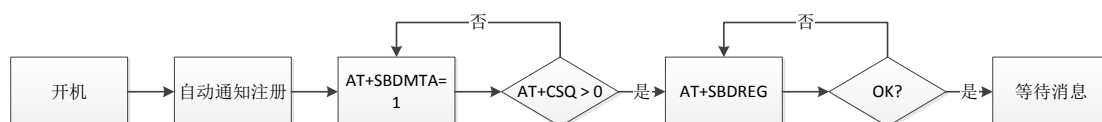


图 4-5 AT 指令集注册自动通知流程图

首先通过 AT+SBDMTA 指令来注册对铱星网关发来的 MT 消息的注册自动通知，然后通过 AT+SBDREG 命令来查询注册的自动通知是否生效。

### (2) 接收消息

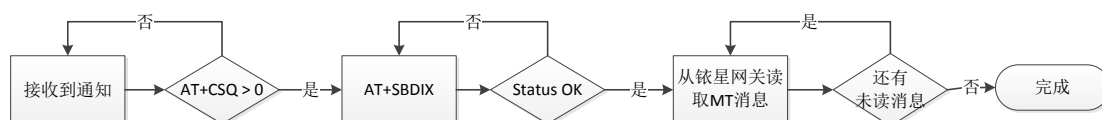


图 4-6 AT 指令集接收 SBD 消息流程图

当铱星终端接收到来自铱星网关的自动通知后判断当前的卫星信号值是否满足传输要求，满足后通过 AT+SBDIX 指令来获得铱星网关的 MT 消息，知道铱星网关没有 MT 消息为止。

### (3) 发送消息

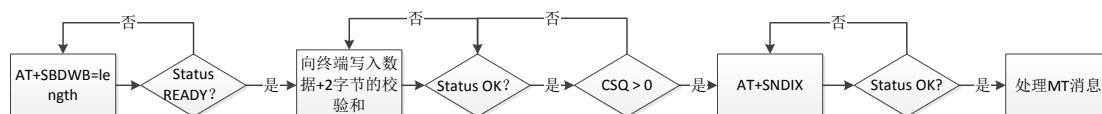


图 4-7 AT 指令集发送 SBD 消息流程图

当铱本文中由于卫星要转发 IP 数据报，IP 数据报中可能存在二进制数据，所以不能通过文本的方式发送，需要通过二进制的方式发送数据。首先通过 AT+SBDWB 指令把数据写入铱星终端，再通过 AT+SBDIX 指令把消息从铱星终端投递到铱星网关。

## 3. 汇聚节点端卫星模块实现

汇聚节点卫星模块主要用有两个：一是上行分片数据的重组转发，二是下行控制指令的投递。



对于上行分片数据重组转发，当接收铱星网关发送的 MO(mobile originate) 报文后需要从中解析出卫星终端发送的 SBD 分片，对分片重组，并通过分片中的元信息把重组后的卫星数据转换成 IP 数据报投递到相应的主机。MO 报文<sup>[32]</sup> 格式如表 4-1 所示：

表 4-1 铱星 MO 报文格式

<i>Filed Name</i>	<i>Data Type</i>	<i>Length(bytes)</i>
Protocol Revision Number	char	1
Overall Message Length	unsigned char	2
MO Header IEI	char	1
MO Header Length	unsigned short	2
CDR Reference	unsigned integer	4
IMEI	char	15
Session Status	unsigned char	1
MOMSN	unsigned char	2
MTMSN	unsigned char	2
Time of Session	unsigned char	4
MO Payload IEI	char	1
MO Payload Length	unsigned char	2
MO Payload	char	70

从中读出 MO Payload 部分即为 SBD 分片数据。分片重组具体伪代码如下：

```

BEGIN
WHILE (TRUE) {
    等待从铱星网关接收报文
    从铱星网关发送的报文中剥离出SBD分片
    IF (是卫星SBD分片) {

        根据元信息中的卫星编号找到相应的重组缓冲区
        根据元信息中的片偏移把分片置于重组缓冲区相应位置
        IF (是最后一个分片) {
            重组完成
            根据元信息把SBD报文转换成IP报文
            投递IP报文
        } ELSE {
            等待接收其余SBD分片
        }

    } ELSE {
        丢弃该报文
    }
}
END

```

对于下行指令投递，当卫星模块接收到下行指令后，根据缓存的卫星标号和 IP 地址的映射把下行指令封装到 MT(mobile terminate)消息中发送给铱星网关，MT 消息格式表 4-2 所示：

表 4-2 铱星 MT 报文格式

Field Name	Data Type	Length
Protocol Revision Number	char	1
Overall Message Length	unsigned short	2
MT Header IEI	char	1
MT Header Length	unsigned short	2
Unique Client Message ID	char	4
IMEI(User ID)	char	15
MT Disposition Flags	unsigned short	2
MT Payload IEI	char	1
MT Payload Length	unsigned short	2
MT Payload	char	70

卫星转发下行控制指令的伪代码如下：

```

BEGIN
WHILE (TRUE) {
    等待接受服务器发送的控制指令
    根据控制指令元信息查找卫星IMEI编号
    把控制指令封装成MT消息
    WHILE (TRUE) {
        把MT消息发送给铱星网关
        等待接收网关MT消息发送确认
        IF (发送成功)
            跳出循环
    }
}
END

```

#### 4.2.7 网关自适应选择模块

具备 3G 和卫星通信能力的节点可以作为候选网关节点，为了保存当前 3G 网络和卫星的通信状态，3G 网关的状态用如下结构记录：

```

typedef struct tg_info {
    char ping_if[IFNAMSIZ];
    uint8_t ping_if_status;
    in_addr_t ping_ips[MAX_HOST_NUM];
    in_addr_t netmask;
    in_addr_t ping_gw_ip;
    in_addr_t default_gw_ip;
    int default_gw_ifnumber;
    uint16_t avg_rtt;
}

```

```

uint16_t avg_lost
time_t expire_timer;
uint8_t expire_count;
pthread_mutex_t tg_info_lock;
} tg_info;

```

其中 ping\_if 字段为进行 3G 探测的网卡名字, ping\_if\_status 字段记录该网卡设备的状态, ping\_ips 字段为进行 3G 探测时 Internet 上主机的 ip 地址, netmask 字段为 ip 地址的子网掩码, ping\_gw\_ip 字段进行 3G 探测时 3G 路由器的 ip 地址, default\_gw\_ip 字段记录当前的默认网关的 ip 地址, default\_gw\_ifnumber 字段为接收默认网关消息的网卡编号, avg\_rtt 为当前默认网关的 3G 网络的平均往返时延, avg\_lost 为当前默认网关 3G 网络的平均丢包率, expire\_timer 为当前默认网关的超时时间, 当在 GW\_EXPIRE\_INTERVAL 时间内没有再收到该默认网关的网关通告报文, 即被认为是网关超时, expire\_count 为当前默认网关的超时次数。

当前节点的卫星状态用如下结果记录:

```

typedef struct sate_info {
    int pipe_fds[2];
    queue *req_queue;
    queue *rep_queue;
    int running;
    int load;
    pthread_cond_t last_choice;
    pthread_mutex_t cond_mutex;
} sate_info;

```

其中 pipe\_fds 字段为包检测过滤模块和卫星模块间的通信管道, req\_queue 字段为请求队列, rep\_queue 字段为应答队列, running 字段表示当前卫星模块运行状态, 当为 0 时表示当前卫星模块没有工作, 即 3G 网络能够满足通信要求, 当为 1 时表示当前卫星模块正在转发 IP 报文, load 表示当前卫星模块发送队列的负载。

在进行网关选择时会综合考虑本节点 3G 网络通信状况、本节点卫星通信状况、周围节点 3G 网络状况、周围节点卫星通信状况, 选择最优网络、最优通信方式, 具体伪代码如下:

```

BEGIN
WHILE(TRUE) {
    //网关选择程序中存在三种事件
    //1. 定期进行3G探测
    //2. 定期进行默认网关超时检测
    //3. 接收周围节点的网关通告
    等待三种事件触发
    IF (定期进行3G探测事件) {

```

```

    计算本地3G网关选择指数gw_3g_local
    计算本地卫星网络选择指数gw_sat_local
    计算默认网关3G网关选择指数gw_3g_default

    IF (默认网关是自身) {
        更新默认网关选择指数
        更新网关超时定时器
        IF (gw_3g_local < 权重阈值) {
            IF (当前节点有卫星模块) {
                IF (卫星线程没有运行) {
                    启动包检测过滤线程
                    设置卫星运行状态为1
                }
            }
        }
    } ELSE {
        IF (gw_3g_local > gw_3g_default) {
            更新默认网关
        }
    }

} ELSE IF (定期网关超时检测事件) {
    IF (默认网关超时次数 > 次数阈值) {
        默认网关超时次数 = 0
        IF (当前节点有卫星模块) {
            IF (卫星线程没有运行) {
                启动包检测过滤线程
                设置卫星运行状态为1
            }
        }
    } ELSE {
        默认网关超时次数 += 1
    }
} ELSE IF (周围节点网关通告) {
    IF (该节点为当前默认网关) {
        更新默认网关
    } ELSE {
        计算该节点3G网关选择指数gw_3g_node
        计算默认网关3G网关选择指数gw_3g_default
        IF (gw_3g_node > gw_3g_default and gw_3g_node > 权重阈值) {
            更新默认网关为该节点
            更新默认网关选择指数
            更新网关超时定时器
        } ELSE IF (gw_3g_node < 权重阈值 and gw_3g_default < 权重阈值) {
            IF (gw_sat_node > gw_sat_default) {

```



## 第五章 网络搭建与测试

### 5.1 M-Link 无线多模通信网关

网关（Gateway）又称为网间连接器、协议转换器<sup>[33]</sup>。网关既可以应用于广域网互连，也可以应用于局域网互连。可以实现不同通信协议、数据格式或者语言，甚至体系结构完全不同的两种体系结构间实现互连。无线多模网关是一种具有多种网络接入能力的网关设备。传统网关通常只能接入一种网络或者进行一种网络协议的转换。与传统网关设备不同，无线多模网关能够同时接入多个网络，在不同网络之间进行协议的转换因此也被称为多模接网关<sup>[34]</sup>。

M-Link 无线多模通信网关是由实验室自主研发的通信网关，它可以接入 WiFi、340M、Ad-hoc 网络等局域网，也可以接入 3G 网络、卫星网络等广域网，实现局域网和 Internet 的互通互联<sup>[35]</sup>。M-Link 无线多模通信网关实物图如图 3-1 所示：



图 5-1 M-Link 无线多模通信网关实物图

### 5.2 无线多模网关自组网测试

搭建如图 5-2 所示的无线 Mesh 网络来测试网关间自组网，网络中有 3 个无线多模网关设备 A、B、C，每个网关节点接入一个外设，网关 A 和网关 B 之间为网络 1，网关 B 和网关 C 之间是网络 2。

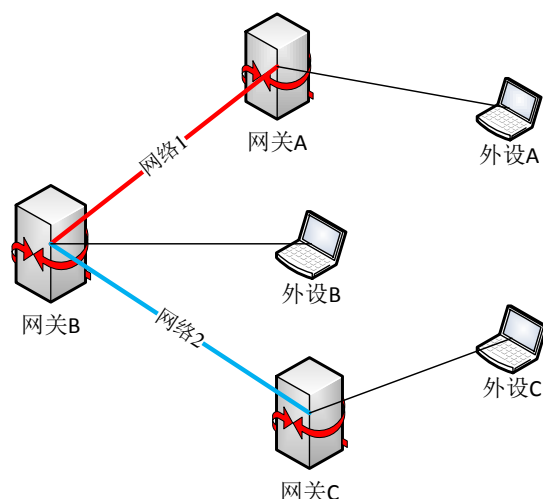


图 5-2 无线多模网关自组网测试网络拓扑

按照表 5-1 设置网关 A、B、C 的网络接口，并设置外设 A 的 IP 地址为 192.168.1.2，外设 B 的 IP 地址为 192.168.2.2，外设 C 的 IP 地址为 192.168.3.2。

表 5-1 无线多模网关自组网测试网关设备 IP 地址

	网络 1	网络 2	外设接口
网关A	10.0.1.1		192.168.1.1
网关B	10.0.1.2	10.0.2.2	192.168.2.1
网关C		10.0.2.3	192.168.3.1

启动三个网关设备，查看三个网关节点初始路由表如图 5-3 所示：

网关A	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
	10.0.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth2
网关B	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
	10.0.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth2
网关C	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
	10.0.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth2

图 5-3 无线多模网关自组网测试节点初始路由表

三个网关初始路由表启动程序，待网络状态收敛后，再查看三个网关节点的路由表项如图 5-4，由图 5-3 和图 5-4 的对比可以看出在程序启动后，网关节点间会按照算法约定在节点间扩散路由消息，当网络状态收敛后，网关 A、B、C 分别建立了到其它两个节点的路由表。

网关A	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	10.0.1.2	255.255.255.0	UG	0	0	0	eth0
	192.168.2.0	10.0.1.2	255.255.255.0	UG	0	0	0	eth0
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
网关B	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	10.0.2.3	255.255.255.0	UG	0	0	0	eth1
	192.168.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	192.168.1.0	10.0.1.1	255.255.255.0	UG	0	0	0	eth0
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
网关C	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	192.168.2.0	10.0.2.2	255.255.255.0	UG	0	0	0	eth1
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	192.168.1.0	10.0.2.2	255.255.255.0	UG	0	0	0	eth1
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1

图 5-4 无线多模网关自组网测试网络状态收敛后节点路由表

此时断开网关 A 和网关 B 之间的网络连接，等到网络收敛后再观察网关节点 A、B、C 的路由表如下：

网关A	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
	10.0.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth2
网关B	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	10.0.2.3	255.255.255.0	UG	0	0	0	eth1
	192.168.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
网关C	Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
	192.168.3.0	0.0.0.0	255.255.255.0	U	0	0	0	eth3
	192.168.2.0	10.0.2.2	255.255.255.0	UG	0	0	0	eth1
	10.0.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
	10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1

图 5-5 无线多模网关自组网测试断开 340M 网络后节点路由表

由图 5-4 和图 5-5 对比可以看出当网关 A 和网关 B 之间的网络断开后，网关节点会按照路由算法约定删除过期路由表。

## 5.3 无线多模网关自适应选择测试

### 5.3.1 3G 网关自适应选择测试

搭建如图 5-6 所示的无线网络：



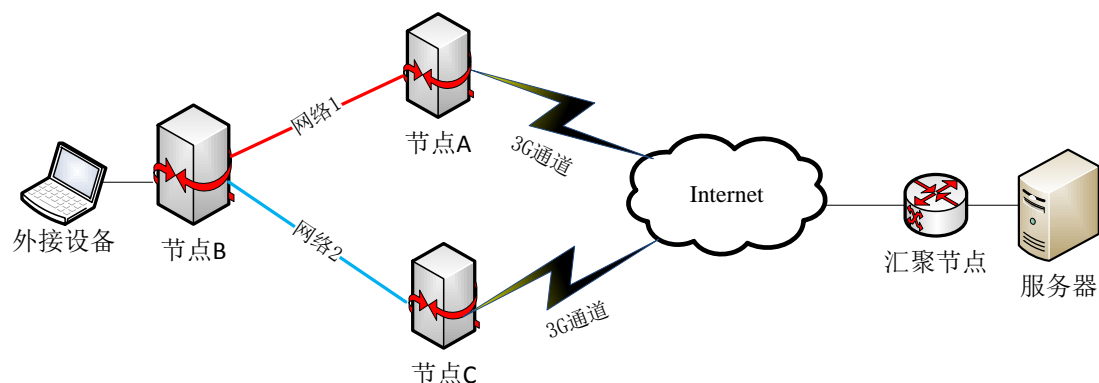


图 5-6 3G 网关选择网络拓扑

网络中有三个无线多模网关 A, B, C。网节点 A 和网节点 C 能够通过 3G 网络接入 Internet, 网节点 A 和网节点 B 间通过网络 1 连接, 网节点 B 和网节点 C 之间通过网络 2 连接。网节点 B 会根据网关自适应选择算法选择网节点 A 或者网节点 C 接入 Internet。按照表设置网节点 A, B, C 的网络接口, 外设的 IP 地址为 192.168.2.2。

表 5-3 3G 网关选择测试各网关设备 IP

	3G网络	网络 1	网络 2	外设接口
网关A	10.0.1.1	10.0.2.1		192.168.1.1
网关B		10.0.2.2	10.0.3.2	192.168.2.1
网关C	10.0.3.1		10.0.3.3	192.168.3.1

启动三个网关设备, 查看网关 B 的初始路由表如下:

```

COM3 - PuTTY
[root@EM9X60 /home]#route -n
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.0    0.0.0.0        255.255.255.0   U        0      0      0 eth3
10.0.1.0       0.0.0.0        255.255.255.0   U        0      0      0 eth0
10.0.2.0       0.0.0.0        255.255.255.0   U        0      0      0 eth1
10.0.3.0       0.0.0.0        255.255.255.0   U        0      0      0 eth2
  
```

图 5-7 3G 网关选择测试节点初始路由表

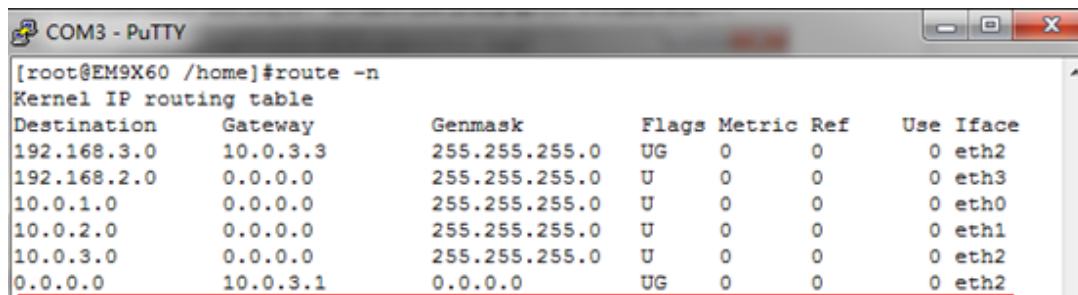
启动三个节点上的无线多模路由程序, 等到网络状态收敛后查看网关 B 的路由表如下:

```

COM3 - PuTTY
[root@EM9X60 /home]#route -n
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.3.0    10.0.3.3        255.255.255.0   UG        0      0      0 eth2
192.168.2.0    0.0.0.0        255.255.255.0   U        0      0      0 eth3
10.0.1.0       0.0.0.0        255.255.255.0   U        0      0      0 eth0
192.168.1.0    10.0.2.1        255.255.255.0   UG        0      0      0 eth1
10.0.2.0       0.0.0.0        255.255.255.0   U        0      0      0 eth1
10.0.3.0       0.0.0.0        255.255.255.0   U        0      0      0 eth2
0.0.0.0        10.0.2.1        0.0.0.0         UG        0      0      0 eth1
  
```

图 5-8 3G 网关选择测试节点稳定后路由表

如图可以看出节点 B 不仅添加了相应的节点间的路由表项, 并且添加了一条默认路由, 节点 B 通过网关自适应选择算法判定网关 A 为最优网关, 便通过此路由把网关 A 作为默认网关, 通过网关 A 接入 Internet。此时断开网关 A 和网关 B 之间的网络, 等到网络状态收敛后, 再查看网关 B 的路由表如下:



```
[root@EM9X60 /home]#route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.3.0      10.0.3.3       255.255.255.0   UG    0      0      0 eth2
192.168.2.0      0.0.0.0        255.255.255.0   U      0      0      0 eth3
10.0.1.0         0.0.0.0        255.255.255.0   U      0      0      0 eth0
10.0.2.0         0.0.0.0        255.255.255.0   U      0      0      0 eth1
10.0.3.0         0.0.0.0        255.255.255.0   U      0      0      0 eth2
0.0.0.0          10.0.3.1       0.0.0.0         UG    0      0      0 eth2
```

图 5-9 3G 网关选择测试断开网络 1 后节点路由表

如图所示当断开网关 A 和网关 B 之间的网络, 等到网络状态收敛后, 网关 B 删除了到网关 A 的路由, 并且把默认网关切换为网关 B。

### 5.3.2 卫星网关自适应选择

取一个无线多模网关, 断开其外部连通网络, 只保留卫星通道, 程序会根据当前的网络状况启动卫星程序, 如图所示的卫星传送网络:

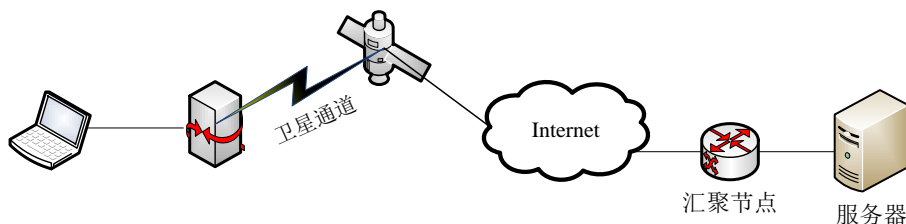


图 5-10 卫星网关自适应选择网络拓扑

无线多模网关通过卫星通道把数据转发给铱星网关, 进而转发给汇聚节点后的服务器, 在外设上启动一个 udp 发送程序, 发送报文为数字 1-10 和英文字母 a-zA-Z 的组合, 报文共 1024 个字节。在服务器端启动一个 udp 接收程序, 接收汇聚节点重组转发的 udp 报文, 接收结果如下:

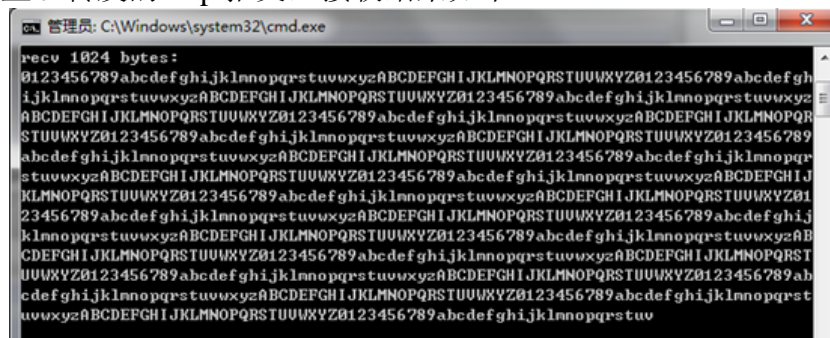


图 5-11 卫星测试结果

把接收到的报文和原始报文用工具比较后发现一致, 说明卫星程序能够对报文分片发送, 并且汇聚节点上卫星程序能够对分片重组转发到服务器

## 第六章 总结与展望

### 6.1 论文工作总结

本文介绍了无线多模网关组网机制的设计和实现,无线多模网关间可以组成无线 Mesh 网,无线多模网关间会组成自组织网络,具备 3G 或者卫星通信能力的无线多模网关节点可以作为网关节点把其它节点接入 Internet。无线多模网关组网机制包括自组网和网关选择两部分,本文在 DSDV 路由算法基础上,对其路由度量做了改进,并设计了一种网关选择算法,最后对 DSDV 协议路由消息进行了相应扩展。首先,为了适应无线多模网关节点间通信方式多模的特性,对 DSDV 路由协议路由度量做了改进,DSDV 路由协议默认采用节点间跳数作为路由度量,但在由无线多模网关组成的无线 Mesh 网络中,节点间通信方式具有异构性,不同的通信方式,节点间的链路延迟可能差别很大,因此不能单单通过节点间跳数来作为路由度量,本文采用优先选择节点跳数,当跳数相等时再考虑节点间链路延迟,选择链路延迟小的作为最优路由。其次,本文设计了一种网关选择算法,网关节点具备 3G 或者卫星两种远距离通信方式,当 3G 网络能满足传输要求时优先选择 3G 网络作为接入网络,当本身及周围节点 3G 网络无法满足通信需求时,选择卫星网络做接入网络。当选择 3G 网络作为接入网络时,考虑节点间跳数、3G 网络平均往返时延、平均丢包率三种因素综合选择网关;当选择卫星作为接入网络时,考虑到卫星的传输时延和传输成本,要对传输内容进行过滤,选择适合利用卫星传输的低速、低数据量的业务。最后,利用无线多模通信网关搭建了相应的无线 Mesh 网测试环境,验证了传输机制的可行性。但设计仍有不足之处,需要进一步改进。

### 6.2 总结建议

首先,本文中根据项目需求假定无线 Mesh 网中网关节点位置基本固定,所以本文的组网机制不适合存在大量高速移动节点的网络环境,在后续研究过程中可以考虑按需路由和主动路由策略相结合,扩大路由算法应用场景。

其次,虽然本文提出了网关自适应选择算法,根据 3G 网络的平均往返时延、平均丢包率和跳数来评估 3G 网络的通信状态,但并未找到一种三因素权值组合的理论最优值。因此在后续工作中,希望能够通过仿真和实际系统测试相结合的方法来得出一个合理的权值组合。

最后，本文提出了利用铱星网络转发 IP 网络数据包的方法，但该方法只适用于低速、无连接的 UDP 数据包转发，限制了应用范围。因此后续研究中会考虑如何通过卫星网络转发 TCP 协议报文。

## 参考文献

- [1] Jose B R, Mathew J, Mythili P. A multi-mode sigma-delta ADC for GSM / WCDMA/WLAN applications[J]. Journal of Signal Processing Systems, 2011, 62(2): 117-130.
- [2] 马华东,陶丹.多媒体传感器网络及其研究进展[J].软件学报, 2006, 09: 2013-2028 .
- [3] Liang Liu, Anlong Ming, Huadong Ma, Xi Zhang, A binary-classification-tree based framework for distributed target classification in multimedia sensor networks[A].//INFOCOM, 2012 Proceedings IEEE. IEEE[C], Florida: ACM Press, 2012: 594-602.
- [4] 王月姣. 无线 Mesh 网路由协议研究[D]. 上海: 上海交通大学, 2008.
- [5] 郭晓雷. 多网关无线 Mesh 网络负载均衡策略研究[D]. 安徽: 中国科学技术大学, 2010.
- [6] 高见. 异构无线 Mesh 网络路由技术研究[D]. 北京: 北京邮电大学, 2013.
- [7] Draves R, Padhye J, Zill B. Routing in multi-radio, multi-hop wireless Mesh networks[A]. //Proceedings of the 10th annual international conference on Mobile computing and networking[C]. Philadelphia: ACM Press, 2004: 114-128.
- [8] Michalak M, Braun T. Common gateway architecture for mobile ad-hoc networks [A]. //Second Annual Conference on Wireless On-demand Network Systems and Services[C], St. Moritz: IEEE Press, 2005: 70-75.
- [9] Baumann R, May M, Plattner B. Field Based Interconnection of Hybrid Wireless Mesh Networks[J]. Computer Engineering and Networks Laboratory, ETH Zurich, 2006.36(2): 23-26.
- [10] Kim M, Ra I, Yoo J, et al. QoS Mesh routing protocol for IEEE 802.16 based wireless Mesh networks[A].//10th International Conference on Advanced Communication Technology[C], Gangwon-Do: ICACT Press, 2008: 812-817.
- [11] 李坤.3G 智能综合接入网关研究[D]. .北京: 北京邮电大学, 2013.
- [12] 盛蕾. WiFi 网络研究及基于 Linux 的测试平台的实现[D]. 上海: 同济大学, 2007.
- [13] 侯振.基于 WiFi 技术的井下无线调度通信系统的实现方案[D]. 西安: 西安科技大学, 2009.
- [14] 吴建军, 程宇新, 梁庆林, 项海格. 第六届卫星通信新业务新技术学术年会

- 论文集[C]. 北京:中国通信学会, 2010: 304-313.
- [15] 杨元喜. 北斗卫星导航系统的进展贡献与挑战[R]. 北京: 测绘学报. 2010: 1-6.
- [16] Akyildiz I F, Wang X, Wang W. Wireless Mesh networks: a survey[J]. Computer networks, 2005, 47(4): 445-487.
- [17] 钱乐无线多媒体传感网络关键技术的研究与实现[D]. 北京: 北京邮电大学, 2013.
- [18] Bruno R, Conti M, Gregori E. Mesh networks: commodity multihop ad hoc networks[J]. Communications Magazine, IEEE, 2005, 43(3): 123-131.
- [19] Broch J, Maltz D A, Johnson D B, et al. A performance comparison of multi-hop wireless ad hoc network routing protocols[A]. //Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking[C]. Dallas: ACM, 1998: 85-97.
- [20] Raza H, Nandal P, Makker S. Selection of cluster-head using PSO in CGSR protocol[A].// 2010 International Conference on Methods and Models in Computer Science (ICM2CS), New Delhi: IEEE, 2010: 91-94.
- [21] Pei G, Gerla M, Hong X, et al. A wireless hierarchical routing protocol with group mobility[A]. //Wireless Communications and Networking Conference[C], New Orleans: IEEE Press, 1999: 1538-1542.
- [22] Chiang C C, Wu H K, Liu W, et al. Routing in clustered multihop, mobile wireless networks with fading channel[A]. //proceedings of IEEE SICON[C]. Singapore: IEEE Press, 1997: 197-211.
- [23] Pengrui Duan, Liang Liu, and Zhao Zhang, A Cross Layer Video Transmission Scheme Combining Geographic Routing and Short-Length Luby Transform Codes[J], International Journal of Distributed Sensor Networks(accepted).
- [24] Perkins C E, Bhagwat P. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers[A].//ACM SIGCOMM Computer Communication Review[C]. London: ACM Press, 1994: 234-244.
- [25] Cheng C, Riley R, Kumar S P R, et al. A loop-free extended Bellman-Ford routing protocol without bouncing effect[A]. //ACM SIGCOMM Computer Communication Review[C]. Austin: ACM Press, 1989: 224-236.
- [26] Malkin, G.,RIP Version 2 Protocol Applicability Statement,RFC 1722, Xylogics, Inc.,November 1994.
- [27] 刘权.基于 Netfilter 机制的 IPSec VPN 网关的研究与实现[D]. 重庆: 重庆大

- 学, 2009.
- [28] S. McCanne and V. Jacobson. The BSD Packet Filter: A New Architecture for User-level Packet Capture[A]. Proceedings of the 1993 Winter USENIX Technical Conference[C]. San Diego: USENIX Press, 1993:1-10.
- [29] 胡金字, 段鹏瑞, 基于 3G 和卫星的无线多模通信系统的设计与实现[J]. 中国科技论文在线, 2014(12):460.
- [30] 秦丹阳. 移动 Ad Hoc 网络自适应路由算法研究[D]. 哈尔滨: 哈尔滨工业大学, 2011.
- [31] 郭晶晶. 无线 Mesh 骨干网最优网关选择[D]. 西安: 西安电子科技大学, 2013.
- [32] Iridium Satellite LLC. Iridium short burst data service developers guide, release 2.0[M]. March 23rd 2007.
- [33] 林瑶, 将慧, 杜蔚轩用 TCP/IP 进行网际互联第一卷:原理、协议与结构[M]. 北京: 电子工业出版社, 2001:56-60.
- [34] Ruichao L, Lianfeng S, Jing H. Design and Implementation of a Wireless Sensor Network Gateway Supporting Multi-Mode Wide Area Access and Video Monitoring[A]. //2009 1st International Conference on Information Science and Engineering (ICISE)[C]. NanJing: IEEE Press, 2009: 2606-2609.
- [35] 刘孟轩 无线多模网关传输机制的设计与实现[D]. 北京.北京邮电大学.2013.

## 作者攻读学位期间发表的学术论文目录

[1] 胡金字, 段鹏瑞. 基于 3G 和卫星的无线多模传输系统的设计与实现. 中国科技论文在线. 2014 年 12 月. 论文编号 201412-460.