基于 SNMP 的多子网物理拓扑发现方法的研究与实现

摘 要

计算机网络规模的不断扩大，网络异构性和分布性的日趋显著，网络结构的日益复杂，用户对网络资源的可靠性要求的不断提高，使得网络管理已经成为网络系统正常运行的关键因素。网络拓扑信息可以帮助网络管理员掌握网络拓扑结构，迅速定位故障发生地点，确定故障影响的范围，还可以成为发现网络设备并调用其它管理功能模块的共同出发点。精确的网络拓扑信息对于现代网络管理和预测网络性能是至关重要的。然而，现代网络的动态特性使得想要通过手动的方式来获得网络拓扑信息是难以做到的。因此，近年来在国内外网络拓扑发现新方法和新技术的研究倍受关注和重视，成为计算机领域研究的热点问题。

本文深入地分析了国内外拓扑发现的相关理论、技术和应用，介绍了常用的可被用来进行网络拓扑发现的协议，并在适用性、网络负荷、速度及准确性等方面对各协议进行对比。在研究单子网和多子网物理拓扑发现算法的基础上，总结了现有的几种物理拓扑发现算法，并对这些算法的优缺点进行分析。在对已有算法进行深入分析的基础上，并结合运用了 AFT、STP 等协议及 Ethereal 软件，本文提出了一种基于 SNMP 的多子网物理拓扑发现方法，该方法有效地解决了 AFT 信息不完整、不支持 SNMP 设备及哑设备的发现等问题。

本文设计并实现了拓扑发现的原型系统，对系统体系结构中各模

块进行了详细的介绍，并在所搭建的试验环境中进行测试，验证了拓扑发现方法的有效性。

本文的创新之处主要体现于以下四个方面：在所提出的拓扑发现方法中结合 SNMP、AFT 及 STP 等协议，实现了对多子网物理拓扑的发现，并实现了简单的异构；采用产生额外流量、试探法等方法，解决 AFT 的不完整问题；使用 Ethereal 软件对数据报侦听，解决不支持 SNMP 的设备的发现问题；对设备端口流量的分析，解决哑设备的发现问题。

关键字：网络管理；SNMP；物理拓扑发现；多子网；异构网络

RESEARCH AND IMPLEMENTATION OF PHYSICAL TOPOLOGY DISCOVERY METHOD IN MULTISUBNET BASED ON SNMP

ABSTRACT

Nowadays, with the development of the technology of computer network, scale of computer network is getting bulky and complex, the need of reliability of computer network is continually improved, network management has been the key to normal operation of the network system. It's more and more important to get a complete and correct topology, which can be used in these fields such as network management, network optimization and fault location. And accurate network topology information is crucial for both network management and performance prediction. Given the dynamic nature of today's IP networks, keeping track of topology information manually is a daunting task. In recent years, it has been attracted considerable attention for a mass of researchers on networks in world, and become the hot reseach area of computer networks.

Based on careful analysis on related theory, technique and application of the topology discovery in world, this thesis introduces seversal common protocols which could be used in topology discovery,

and which are compared to each other according to their suitable application, network load, speed and accuracy. On the base of the physical topology discover algorithm for single or subnets, the author is concerned with summarizing these topology discovery algorithms, and comparison of advantages and disadvantages among these algorithms is also presented. Based on studying the methods of physical topology discovery and compared them, and gave a novel physical topology discovery method in multisubnet based on SNMP, make use of Ethereal and many protocols such as AFT, STP, which can sovle nicely the problem of incomplete AFT, the devices which don't support SNMP and dump or uncooperative network elements.

A complete topology discovery system has been designed and implemented, and every modules of the system have also introduced detailedly in the end of the thesis. The new method has been tested in the real networks. It is proved that this method is efficient.

The innovation of the work in this thesis can be shown as the following four aspects: There prototocols SNMP, STP and AFT are combined in this new topology discovery method, which can discover multisubnet and heterogeneous network; Owing to the using of the methods of creating extra traffic and approxinmations, we solve the problem of incomplete AFT; Owing to the using of Ethereal, which can we use to solve the problem of discoverying the devices which don't

support SNMP; Benefited from analysis to the traffic of the equipment, we can solve the problem of discoverying dump or uncooperative network elements.

Key Words: network management; snmp; physical topology discovery; multisubnet; heterogeneous network

独 创 性 声 明

本人声明所呈交的学位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的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含本人为获得浙江工商大学或其它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谢意。

签名：彭剑光 日期： 年 月 日

关于论文使用授权的说明

本学位论文作者完全了解浙江工商大学有关保留、使用学位论文的规定：浙江工商大学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，允许论文被查阅和借阅，可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文，并且本人电子文档的内容和纸质论文的内容相一致。

保密的学位论文在解密后也遵守此规定。

签名：彭剑光 导师签名：

日期： 年 月 日

1 引言

1.1 研究背景及意义

随着计算机网络的飞速发展，人们对于通信方式、通信效率的要求也越来越高。计算机网络已经渗透到了社会的各个领域，包括政府、商业、军事、教育和科研等领域，逐渐成为企业和个人不可缺少的工具。随着网络技术的迅速发展，网络规模不断扩大，网络结构越来越复杂，网络功能越来越强大，网络管理技术已经成为一个日益重要的课题。

计算机网络管理是对网络和网络资源的规划、设计、监测和控制等各种活动的总称。根据ISO的定义，网络管理主要包括五个功能域[1]：故障管理、配置管理、性能管理、安全管理和计费管理。五大功能域之间既相互独立，又存在着联系。在五大功能域中，配置管理是基础，它的主要功能包括发现网络拓扑结构、监视和管理网络设备的配置情况。要对网络性能及流量进行分析，首先必须得到正确的拓扑结构和相关信息。因此，网络拓扑发现成为网络管理的基础。

拓扑发现是指发现网元并确定网元之间的互连关系，包括互连设备(如路由器、网桥、交换机等)、主机和子网。

网络拓扑从不同的抽象层次上可分为逻辑网络拓扑与物理网络拓扑。逻辑网络拓扑指的是网络层设备及其之间的连接关系，即路由器到路由器、路由器接口到子网的连接关系。物理网络拓扑指的是网络元素之间的实际物理连接，即在原有网络层拓扑的基础上增加交换机到交换机、交换机到路由器、交换机到主机之间的连接关系。其中，发现交换机到交换机之间的连接关系是物理网络拓扑发现的关键。

网络拓扑发现对于网络管理、网络当前状况的研究以及优化网络性能等有着重要的作用及意义，主要体现于以下几个方面[2]：

1) 模拟网络：为了模拟实际网络，分析网络性能，合理扩容和优化网络，必须先得到该网络的拓扑结构；
2) 网络优化：网络拓扑信息可以帮助网络管理者确定是否需要增加新的路由器，当前硬件是否配置正确，并发现网络中的瓶颈所作和失败的链路，进行网络优化；

3) 用户接入方式的选择：网络拓扑信息可以帮助用户确定自身处于网络中的位置，从而决定服务器的位置以及选择哪一个网络服务提供商可以将网络时延最小化，可用带宽最大化；

4) 研究拓扑敏感算法：一些新的协议和算法可以在得到网络拓扑信息的基础上改善网络性能。例如：基于拓扑敏感策略和Qos的路由选择算法与基于拓扑敏感进程组选项和组通信算法；

5) 确定镜像服务器的位置：根据拓扑信息合理配置镜像服务器的位置以最大可能减少时延，解决瓶颈问题。

1.2 研究现状

尽管拓扑信息对现代的IP网络管理工作非常重要，但是如今从市场上可得到的网络管理平台中没有一个提供自动发现物理IP网络连通性的通用目的工具。多数网络管理系统提供了3层拓扑的发现，由于路由器能明确地知道其3层邻居的情况，因此，发现3层拓扑就相对容易。并且3层拓扑只覆盖了IP网络互连关系的小部分，因为它不捕捉属于不同IP子网的2层网络设备（如，交换机和网桥）的复杂连接关系。

网络层主要是从路由表中提取拓扑发现信息。1990年IETF在RFC1157[3]中正式公布了“简单网络管理协议”（SNMP），1991年K.McCloghiie和M.Rose在RFC1213[4]中提出了基于TCP/IP协议的因特网管理信息库MIB-II，1993年4月IETF又发布了SNMPv2。国外许多商业网络管理系统都已提供了基于SNMP自动发现3层网络拓扑的算法，如HP公司的Open View[5]、IBM的Tivoli[6]，Actualit的Optimal Surveyor和Dartmouth的Intermapper[7]。

网络上的设备十分繁杂，不是所有的设备都支持SNMP。基于SNMP的网络层拓扑发现算法要求网络内的设备必须支持SNMP代理，具有其一定的局限性。因此，人们逐渐开始利用路由协议来发现网络层拓扑。Aman Shaikh等人[8]提出了一种通过监听网络中的OSPF数据包来获得网络拓扑的方法。这种方法的优点是可以绕过设备不支持SNMP带来的麻烦，同时可以避免向网络中注入大量数据包，从而减轻了网络的负担。

没有自动捕获物理拓扑（即2层）信息的解决方案意味着网络管理员通常被

迫为他们所使用的管理工具手工输入必要的信息。由于现代IP网络具有的动态特性与日趋增加的复杂性，要想通过手工方式进行拓扑信息跟踪是不可想象的。所以必须为IP网络的最新物理拓扑找到有效的、有通用目的的算法，自动生成拓扑图。综上所述，任何发现物理IP拓扑的切实可行的解决方案必须应对以下三个难题[9]：

1) 有限的本地信息。算法只能对设备可给出的有关信息做很少的推理假设；也就是说，算法只能利用大多数网络设备通常给出的本地信息。此外，由于2层网络设备不能明确感知物理上与其连接的邻居设备，所以2层的物理互连不是直接可以得到的。
2) 设备对协议层的透明性。算法必须为在ISO协议栈不同层次上运行的网络设备找到正确的连接关系。这点很重要，因为交换式IP子网中，2层网络设备对控制进、出子网数据传输的3层路由器是完全透明的。
3) 网络设备的异构性。发现算法应该能从不同种类的网络设备中搜集拓扑信息，并且确保从不同厂商设备中搜集到的相关信息被正确地访问和解释。

对于局域网（LAN）的物理拓扑发现的成果还相对较少，对于LAN拓扑发现所感兴趣的主要集中在构成该LAN的二层网络设备，而实质上，这些2层设备对那些只能发现IP路由器互连的工具是透明的。对于物理网络拓扑的发现，主要是利用ARP协议和ICMP协议来发现局域网中的设备，SNMP的出现也为物理网络拓扑发现算法提供了有力的工具。在国外，Yuri Breitbart等人在参考文献[9]中提出了一种基于Bridge MIB的拓扑发现算法，并给出了严格的数学证明，但是该方法对网络环境要求太高。Hwa-Chun Lin等人利用Bridge MIB(RFC 1493)来获取交换机的连接信息[10]，但是该方法的准确性不高。针对二层拓扑发现一些厂家提供了相应的协议及产品，Cisco的发现协议Cisco Discovery Protocol和Bay的Networks Optivity Enterprise，然而，这些工具通常基于特定厂商扩展的SNMP MIB，不具有通用性。IETF在意识到物理拓扑发现的重要性后，指定了物理拓扑SNMP MIB库[11]，但是并没有定义任何通用的协议和算法用于获得物理拓扑信息。

1.3 研究内容

1.3.1 本文贡献

本文对已有的物理拓扑发现算法进行研究和分析，在此基础上进行改进，提出新的多子网物理拓扑发现算法。利用和研究现有的拓扑发现工具，分析各自的有点及不足，从而提出新的基于SNMP的多子网的物理拓扑发现算法，并完成原型系统的设计与实现。具体内容如下：

(1) 研究和分析了目前国内外拓扑发现的研究背景、意义及研究现状。对SNMP等网络拓扑发现相关工具及协议进行研究，分析其优缺点，并对拓扑发现性能进行比较，分析他们各自适用的网络环境；

(2) 介绍了单子网和多子网交换域，针对当前对多子网物理拓扑发现算法的研究相对较少的情况，分析比较了已有的物理拓扑发现算法（基于SNMP、ICMP、AFT及STP等物理网络拓扑算法），指出其各自存在的问题与不足，在此基础上提出了一种运算速率高、实用性好的改进的多子网物理拓扑发现算法；

(3) 该算法支持简单的异构，结合相关研究，对多子网内“哑”设备的发现、AFT不完整及不支持SNMP设备的发现的问题上也提出相应的创新；

(4) 详细介绍实验设计的各个模块，基于VC++6.0、Microsoft ACCESS、Cisco路由器及华为交换机搭建实验平台，验证了算法的有效性。

1.3.2 本文的创新点

本文的创新之处主要体现于以下四个方面：

(1) 在所提出的拓扑发现方法中结合 SNMP、AFT 及 STP 等协议，实现了对多子网物理拓扑的发现，并实现了简单的异构；

(2) 采用产生额外流量、试探法等方法，解决 AFT 的不完整问题；

(3) 使用 Ethereal 软件对数据报侦听，解决不支持 SNMP 的设备的发现问题；

(4) 对设备端口流量的分析，解决哑设备的发现问题。

1.4 本文的组织结构

本文共分为六个章节，内容安排如下：

第一章绪论。阐述了论文的选题背景及研究意义，分析了网络拓扑发现的重

要性及当前国内外拓扑发现技术的研究现状，并阐述了本文的主要工作。

第二章介绍了相关的网络拓扑发现工具及协议。详细介绍了 SNMP（简单网络管理协议），再分别介绍了 ICMP（网际控制报文协议）、Ping 程序、Traceroute 程序、DNS 及 ARP 等其他拓扑发现工具及协议。

第三章是物理网络拓扑发现算法分析。详细介绍了单子网和多子网交换域，并且介绍了基于 SNMP、ICMP、AFT 及 STP 等物理网络拓扑算法。

第四章介绍改进的多子网物理拓扑发现算法。内容包括算法的可行性分析、算法提出及实现所面临的问题的解决、理论基础、算法的具体描述以及流程图。

第五章介绍拓扑发现的具体实现，介绍了系统体系结构及具体模块的实现，并详细描述了实验设计及实验结果。

第六章总结并展望下一步研究工作。

2 网络拓扑发现相关工具及协议

2.1 SNMP 协议概述

SNMP，即 Simple Network Management Protocol，简单网络管理协议。它是一个标准的用于管理 IP 网络上设备的协议。首先是由 Internet 工程任务组织(Internet Engineering Task Force)(IETF)的研究小组为了解决 Internet 上的路由器管理问题而提出的。它可以在 IP、IPX、AppleTalk、OSI 以及其他用到的传输协议上被使用。SNMP 是最早提出的网络管理协议之一，它一推出就得到了广泛的应用和支持，特别是很快得到了数百家厂商的支持，其中包括 IBM、HP、SUN 等大公司和厂商。目前 SNMP 已成为网络管理领域中事实上的工业标准，并被广泛支持和应用，大多数网络管理系统和平台都是基于 SNMP 的。

SNMP 的前身是简单网关监控协议(SGMP)，用来对通信线路进行管理。随后，人们对 SGMP 进行了很大的修改，特别是加入了符合 Internet 定义的 SMI 和 MIB：体系结构，改进后的协议就是著名的 SNMP。SNMP 的目标是管理互联网 Internet 上众多厂家生产的软硬件平台，因此 SNMP 受 Internet 标准网络管理框架的影响也很大。现在 SNMP 已经出到第三个版本的协议，其功能较以前已经大大地加强和改进了。

SNMP 被设计成与协议无关，所以它可以在 IP，IPX，AppleTalk，OSI 以及其他用到的传输协议上被使用。

SNMP 是一系列协议组和规范（见表 2-1），它们提供了一种从网络上的设备中收集网络管理信息的方法。SNMP 也为设备向网络管理工作站报告问题和错误提供了一种方法。[12]

表 2-1 SNMP 的组成

名字	说明
MIB	管理信息库
SMI	管理信息的结构和标识
SNMP	简单网络管理协议

SNMP 的体系结构是围绕着以下四个概念和目标进行设计的：保持管理代理

(agent)的软件成本尽可能低；最大限度地保持远程管理的功能，以便充分利用Internet的网络资源；体系结构必须有扩充的余地；保持SNMP的独立性，不依赖于具体的计算机、网关和网络传输协议。在最近的改进中，又加入了保证SNMP体系本身安全性的目标。

2.1.1 SNMP协议的发展

简单网络管理协议是建立在TCP/IP网络上的公共网络管理协议，它定义了用于交换管理信息的协议、管理信息的表示格式、分布式的组织框架和一种特定的储存管理信息的数据库（MIB）。它的发展经过了几个阶段。[13]

目前使用到的SNMP共有3个版本和两个扩展。目前SNMP共有v1，v2，v3共3个版本：[14]

- v1和v2都具有基本的读、写MIB功能；
- v2增加了警报、批量数据获取、管理站和管理站的通信能力；
- v3在v2的基础上增加了USM，使用加密的数据和用户验证技术，提高了安全性。

另外，RMON是SNMP的一个重要扩展，为SNMP增加了子网流量、统计、分析能力，现有两个版本：

- RMON提供了OSI七层网络结构中网络层和数据链路层监视能力；
- RMON2提供了OSI七层网络结构中网络层之上各层的监视能力。

2.1.1.1 SNMP v1

1990年IETF（互联网工程任务组）在RFC1157中正式公布了SNMP，又称为SNMPv1。

SNMP的设计原则是简单、可靠、有效，这样做有以下几个好处：

1. 开发简单，周期短；
2. 开发费用低；
3. 易掌握、易普及；
4. 带宽利用率高；
5. 使用方便。

SNMP协议定义了使用到的传输层协议、支持的操作、操作相关的PDU结构、操作的时序、角色、实例取值、共同体等。

根据网络管理的需求，SNMP 设计了 5 个基本操作，其支持的操作仅仅是对变量的修改和检查：

- GetRequest 读对象值操作，使 NMS 能够从被管理 Agent 中检索对象的值；
- GetNextRequest 读取当前对象的下一个可读取的对象实例值；
- SetRequest NMS 更新 Agent 中对象的值；
- GetResponse Agent 对 GetRequest/GetNextRequest/SetRequest 3 种操作的应答；
- Trap Agent 向 NMS 发送对象值。

2.1.1.2 SNMP v2

SNMPv1 公布之后得到了广泛的应用，广泛的使用也使使用者发现了 SNMPv1 的很多缺陷和不足。SNMPv1 一个主要的缺陷是没有提供安全功能，特别是不能对管理消息进行鉴别，也不能防止监听。另一方面，SNMP 缺少管理站到管理站的通信机能，使管理站之间不能有效协作。为了弥补 SNMPv1 的缺陷，对它进行了重大改进，因此，1993 年发布了 SNMPv2，具有以下特点[15]：

1. 支持分布式网络管理；
2. 扩展了数据类型；
3. 可以实现大量数据的同时传输，提高了效率和性能；
4. 丰富了故障处理能力；
5. 增加了集合处理功能。

SNMPv2 能支持高度集中的网络管理策略，也能支持分布式的管理策略。在分布式的管理策略下，一些系统在运行时既有管理站的功能也有代理的功能。SNMPv2 对 SNMPv1 的主要增强可以分为以下几类：

- SMI
- 管理站到管理站功能
- 协议控制

SNMPv2 SMI 在几个方面扩展了 SNMPSMI。SMI v2 中增加了用于定义对象的扩展宏，增加了几个新的数据类型。另一个非常显著的变化就是对创建和 删除表中的概念行提供了新的规范。

协议操作中最显著的变化就是包含两种新的 PDU，以下是 SNMPv2 提供的 7 种操作：

- GetRequest 管理站向代理读取对象实例值；
- GetNextRequest 管理站向代理读取给定对象的下一个可用实例值；
- SetRequest 管理站向代理发出的设置操作；
- SNMPv2-Trap 代理向管理站主动发起的通告消息；
- Response 代理响应管理站请求的应答包；
- InformRequest 管理站向另一个管理站报告通报的消息；
- GetBulkRequest 管理站向代理读取表中的若干行的操作。

与 SNMPv1 一样，GetRequest、GetNextRequest 和 SetRequest 操作由管理站发出，代理受到请求都要一个 Response 消息应答，不管处理成功与否。

SBNPv2-Trap 由代理主动发出，管理站没有相应的应答。与 v1 不同，SNMPv2-Trap 的 PDU 结构与 GetRequest、GetNextRequest、SetRequest、InformRequest 相同，不像在 v1 中拥有自己特殊的模式。

InformRequest 是 SNMPv2 引入的新的操作，由管理站发起，向另一个管理站报告状态和数据。

2.1.1.3 SNMP v3

SNMPv2 因为项目时间紧迫，而在安全性发面没有达成一致的意见，所以在最终形式中没有包括安全性。安全性要求在 SNMPv2 中就被迫切地提出来，这是因为 SNMP 的消息在网络上传输面临着以下的安全性问题[14]：

- 修改信息　一个实体可以修改另一个授权实体产生的传输中的消息，从而以这样的一种方式实现非授权的管理操作，包括设置对象的取值；
- 伪装　通过伪装成另一个授权实体，该授权实体试图进行某些没有授权的操作；
- 修改信息流　SNMP 被设计成在无连接的传输层协议上传输，这样便存在 SNMP 消息被重新排序、显示或重播来影响授权的管理操作的威胁；
- 泄密　一个实体可能观测管理站和代理之间的交换，从而得知被管理对象的取值以及一些应用通告的事件。

因为这些原因，提出了SNMP的安全性要求。1997年4月IETF成立了

SNTMPv3工作组，于1998年1月提出了互联网建议RFC 2271-2275,正式形成SNMPv3。SNMPv3的RFC描述了SNMPv3的整体框架和具体的消息结构及安全特性，没有定义新的SNMP PDU格式，因此在新的结构中必须使用已有的SNMPv1和SNMPv2 PDU、SMI及主要MIB。所以，可以有这样一个公式：

SNMPv3=SNMPv2+ 安全+ 管理。

上面所写的SNMP面临的安全性问题相应的解决方法，就是对数据进行加密和鉴别，借助于密码学相关的加密和摘要算法实现：

- 鉴别　数据整体性和数据发送源鉴别，保证消息是由该发送源发送的，不是别人伪造的数据包、传输过程中没有被篡改过。使用 HMAC、MD5 散列函数或 SHA-1 这些算法对数据进行摘要，从而鉴别数据有没有哦被篡改；
- 加密　对数据进行加密，保证不能使用网络数据包截获技术将包侦听而直接解读。使用 DES 的 CBC（Cipher Block Chaining）模式来加密数据，即保证了加解密的效率，又保证了足够的强度。

2.1.1.4 RMON 与 RMON2

RMON是适应对子网流量监视的需求而产生的。RMON发布在SNMPv1之后，在SNMPv2之前。RMON最初的设计是用来解决从一个中心点管理各局域分网和远程站点的问题。RMON规范是由SNMP MIB扩展而来。 RMON中，网络监视数据包含了一组统计数据和性能指标，它们在不同的监视器（或称探测器）和控制台系统之间相互交换。结果数据可用来监控网络利用率，以用于网络规划，性能优化和协助网络错误诊断[16]。RMON的设计是为了实现如下目标：

- 离线操作
- 前摄监视
- 问题探测和报告
- 有效的附加数据
- 多管理站

RMON2在SNMPv2之后发布，主要对RMON进行了扩充。RMON2对RMON的扩充主要表现在新的类型和MIB的扩展上。另一方面，RMON2扩大了对ISO网络模型的监控深度，可以监视更多的层次。

RMON2是RMON的升级版本。RMON对ISO网络模型的第2层（数据链路层）和第3层（网络层）进行分析。RMON2除了对这两层进行监视外，还对传输层、会话层、表示层、应用层进行监视。图2-1描述了RMON和RMON2适用的网络层次。[14]

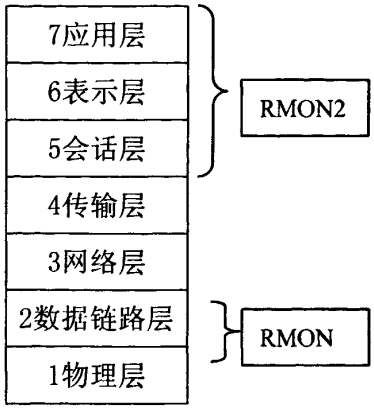


图 2-1 RMON 和 RMON2 适用的网络层次

RMON可以监听MAC地址，并分析上面的IP，RMON在这两个层面上进行分析、统计。RMON2还可以分析再上面的各层数据，比如应用层的E-mail、Ftp、WWW等，可以分析的范围加大了，更方便进行网络性能分析。

2.1.2 管理信息结构（SMI）

管理信息结构（SMI）是SNMP的描述方法。有众多的供应商提供支持SNMP协议的设备和服务。如果没有一种约束机制，可能各个企业写出来的MIB都各不相同，这样的网络设备在SNMP层上的协同会出现大麻烦。所以，需要一种机制限制和规范MIB的定义，这就是SMI。它规定了使用到的抽象语法记法1（ASN.1）子类型、宏、符号等，是ASN.1的一个子集和超集，定义了SNMP自定义类型，用于描述SNMP协议数据包和管理信息库（MIB），是SNMP的基础之一。

SMI定义在RFC1155中。该规范定义了一个基本框架，使用框架内的规范可以定义MIB。SMI定义了基本的数据类型、宏结构及命令规则。SNMP的主要目标之一是简化网络管理，所以，SMI也将ASN.1进行限制和简化，只用到其中很小的一部分。因此，MIB只能存储简单的数据类型，分别是标量和标量的二维数组。SNMP只能检索标量，包括表中的单个条目。SMI不支持复杂数据结构的创建和检索。

SMI的内容包括：

- MIB 结构定义语句
- 单个对象定义语句（包括语法和每个对象的值）
- 数据编码格式

管理对象以虚拟信息存储方式存储和访问，MIB中的对象以ASN.1语言定义。看一个例子：

atIfIndex{atEntry 1}

Syntax:

INTEGER

Definition:

The interface number for the physical address.

Access:

read-write

Status:

mandatory

例子中定义了一个管理对象，其名字为atIfIndex,语法为INTEGER。编码方法由INTEGER的BER编码方法指定。SMI规定，SNMP使用BER对管理信息进行编码。

每个管理对象有自己的名字、类型（Syntax）和编码。以下主要介绍这 3 种属性。

2.1.2.1 名字

对象名字与对象标识一样是唯一的。名字用于表示对象，对象标识则可以更准确地标识对象。

标识是一个树形结构，根结点不进行标记，但它至少有三个结点：

- 一个由 CCITT 管理，标记为 ccitt(0)
- 另一个子结点由国际标准化组织管理，标记为 iso(1)
- 第三个由两者共同管理，标记为 joint-iso-ccitt(2)

2.1.2.2 语法

每一个SNMP MIB内部对象都要正式定义。定义规定了对象的数据类型、允许的形式、取值范围以及与其他MIB内部对象之间的关系。定义时使用ASN.1定

义每一个对象，也用来定义整个MIB的结构。

SMI中使用三种语法：原始类型、SMI预定义类型、自定义类型：

1. 原始类型

SNMP中使用到如下ASN.1原始类型：

INTEGER、OCTET STRING、OBJECT IDENTIFIER、NULL

其中，整数类型可以作为枚举类型使用

2. 构造类型

SNMP 使用 ASN.1 中的 SEQUENCE 建立行或表。

3. 定义的类型

SNMP 允许在一个新产品的范围内定义新类型，新类型必需能够分解为基本类型、行、表或其他新类型。这些类型是 SNMP 使用 ASN.1 语言自定义的类型，在 SNMP 中使用，与其他 ASN.1 基本类型和结构类型一起构成了 SNMP 中使用的类型。

SMI 中定义了用于 SNMP 的一些自定义类型：

- NetworkAddress 描述多个可能的协议族中的地址格式；
- IpAddress 描述 32 位的 IP 地址，它表示为长度为 4 的字符窜；
- Counter 描述一个非负整数，它只能增加，直到最大值；
- Gauge 描述一个非负整数，它可以增加或减少，但在最大值时停止；
- TimeTicks 此类型为非负整数，用于记录一个时间点起经过了多少个百分之一秒的时间；
- Opaque 支持对 ASN.1 语法进行扩充的能力。

2.1.2.3 编码

使用 ASN.1 的 BER 编码规则将数据从 ASN.1 编码为字节流以在网络上传输。

2.1.3 管理信息库（MIB）

所谓管理信息库，或者 MIB，就是所有代理进程包含的、并且能够被管理进程进行查询和设置的信息的集合。SNMPv1 中有两个版本的管理信息库：RFC 1156 定义了 SNMP 第一个版本的管理信息，称为 MIB-I；RFC 1213 定义了第二个版本的管理信息，称为 MIB-II。MIB-II 对 MIB-I 进行了扩展和修改，现在的

SNMP 都以 MIB-II 为基准。

MIB 变量使用的名字取自 ISO 和 ITU 管理的对象标识符（object identifier）名字空间。对象标识是一个整数序列，以点（“.”）分隔。这些整数构成一个树型结构，类似于 DNS 或 UNIX 的文件系统。

图 2-2 显示了在 SNMP 中用到的这种树型结构。所有的 MIB 变量都从 1.3.6.1.2.1 这个标识开始。树上的每个节点同时还有一个文字名。例如标识 1.3.6.1.2.1 就和 iso.org.dod.internet.memt.mib 对应。这主要是为了人们阅读方便。在实际应用中，MIB 变量是以对象标识来标识的，当然都是以 1.3.6.1.2.1 开始的。

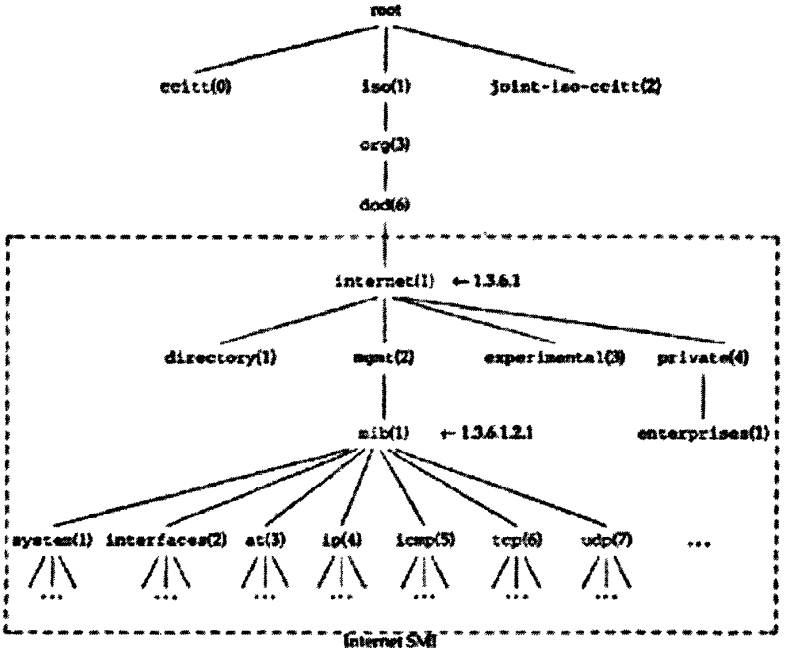


图 2-2 管理信息库中的对象标识

最初的结点 MIB 将其所管理的信息分为 8 个类别，如表 2-2。

MIB-II 包含下列分组：System、Interfaces、Address Translation (deprecated)、IP、ICMP、TCP、UDP、EGP、Transmission、SNMP。接下来简单介绍 MIB-II 中的每一个组：[4]

表 2-2 最初的 MIB 管理的信息类别

类别	标号	所包含的信息
system	(1)	主机或路由器的操作系统
interfaces	(2)	各种网络接口及它们的测定通信量
address translation	(3)	地址转换（例如 ARP 映射）
ip	(4)	Internet 软件（IP 分组统计）
icmp	(5)	ICMP 软件（已收到 ICMP 消息的统计）
tcp	(6)	TCP 软件（算法、参数和统计）
udp	(7)	UDP 软件（UDP 通信量统计）
egp	(8)	EGP 软件（外部网关协议通信量统计）

- system 组：用于存放设备的商品信息，由一系列标量组成，非表格；
- interfaces 组：用来存放网络接口设备状态，使用一个标量指出系统有多少个网络接口，另外使用一张表存放具体的接口数据网络接口设备可以是网卡、调制解调器、无线接口等。interfaces 组包含一个标量，指出系统中有几个网络接口（Interface）；
- at 组：该组的功能是提供网络地址转换信息。MIB-II 中每个网络协议组都包含各自的网络地址转换表；
- ip 组：保存 IP 层状态数据，包含一组标量和 3 个表格变量。SNMP 要求所有的设备实现 ip 组。ip 组由标量和三张表组成，标量描述了 IP 层的状态，ipAddrTable 存放地址数据，ipRouteTable 存放路由表，ipNetToMediaTable 是 IP 层自己的地址转换表；
- icmp 组：icmp 相关状态。SNMP 要求所有代理实现 ICMP 组。ICMP 组全部是标量；
- tcp 组：表示 tcp 状态。实现 TCP 的系统要求实现 tcp 组。tcp 组中关于连接的信息是短暂的，生命周期就是连接的周期。tcp 组由一个表和其他标量组成，标量保存 TCP 系统的状态，tcpConnTable 保存本机每一个 TCP 连接的信息；
- udp 组：存放 udp 相关状态数据，实现 udp 协议的机器必须实现 udp 组。udp 组共有 4 个标量 1 张表，标量存放 udp 协议状态数据，表存放正在

监听的 UDP 地址；

- egp 组：存放 EGP 协议设备的信息，实现 EGP 协议的设备强制实现 egp 组。本组由一个表和一系列标量组成，标量保存 EGP 协议信息，egpNeighTable 存放 egp 邻居信息；
- transmission：MIB-I 缺少对不同传输介质的支持，而 MIB-II 增加了 transmission 以支持不同的传输介质，当管理不同传输介质的国际标准出台后，transmission 组将增加相应的支持；
- snmp 组：存放 SNMP 相关状态信息。实现 SNMP 协议的设备被强制要求实现本组。snmp 组中的一些对象可能是 0 值，表明相应的功能没有实现。标量描述 SNMP 协议层的基本状态。

2.1.4 SNMP 协议报文

关于管理进程和代理进程之间的交互信息，SNMP 定义了 5 种报文：get-request、set-request 、get-next-request 、get-response、trap。[17] 前面的 3 种操作是由管理进程向代理进程发出的，后面的 2 个操作是代理进程发给管理进程的。图 2-3 描述了 SNMP 的这 5 种报文操作。

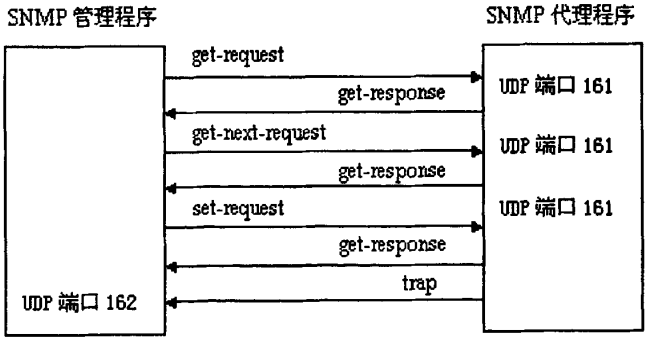


图 2-3 SNMP 的 5 种报文操作

一个 SNMP 报文共有三个部分组成，即公共 SNMP 首部、get/set 首部 trap 首部、变量绑定。图 2-4 是封装成 UDP 数据报的 5 种操作的 SNMP 报文格式。

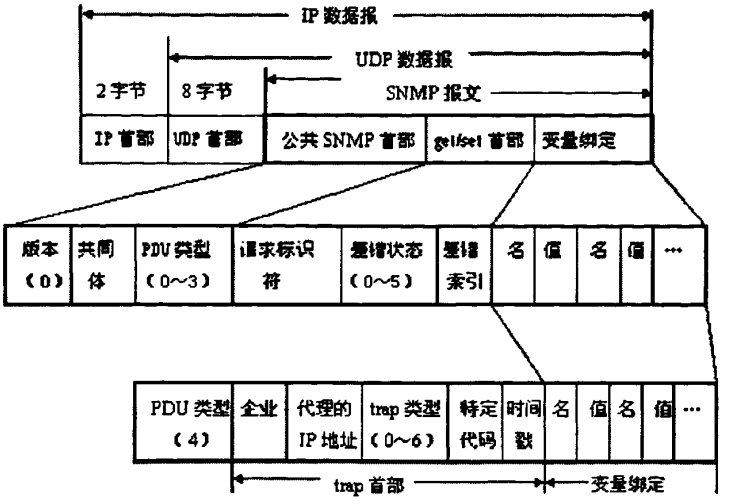


图 2-4 SNMP 报文格式

2.2 ICMP 协议

ICMP 是“Internet Control Message Protocol”（Internet 控制消息协议）的缩写。它经常被认为是 IP 层的一个组成部分。它传递差错报文以及其他需要注意的信息。ICMP 报文通常被 IP 层或更高层协议（TCP 或 UDP）使用。[18]一些 ICMP 报文把差错报文返回给用户进程。

ICMP 报文的格式如表 2-3 所示。所有报文的前 4 个字节都是一样的，但是剩下的其他字节则互不相同。类型字段可以有 15 个不同的值，以描述特定类型的 ICMP 报文。某些 ICMP 报文还使用代码字段的值来进一步描述不同的条件。检验和字段是必需的，覆盖整个 ICMP 报文。

表 2-3 ICMP 报文

8 位类型	8 位代码	16 位校验和
（不同类型和代码有不同的内容）		

各种类型的 ICMP 报文如表 2-4 所示，不同类型出报文中的类型字段和代码字段来共同决定。图中的最后两列表明 ICMP 报文是一份查询报文还是一份差错报文。因为对 ICMP 差错报文有时需要作特殊处理，因此我们需要对它们进行区

分。例如，在对 ICMP 差错报文进行响应时，永远不会生成另一份 ICMP 差错报文（如果没有这个限制规则，可能会遇到一个差错产生另一个差错的情况，而差错再产生差错，这样会无休止地循环下去）。

表 2-4 ICMP 报文类型

类型	描述	查询	差错
0	回显应答（ping 应答）	•	
3	目的不可达		•
4	源端被关闭（基本流控制）		•
5	重定向		•
8	请求回显（ping 请求）	•	
9	路由器通告	•	
10	路由器请求	•	
11	超时		•
12	参数问题		•
13	时间戳请求	•	
14	时间戳应答	•	
15	信息请求	•	
16	信息应答	•	
17	地址掩码请求	•	
18	地址掩码应答	•	

当发送一份 ICMP 差错报文时，报文始终包含 IP 的首部和产生 ICMP 差错报文的 IP 数据报的前 8 个字节。这样，接收 ICMP 差错报文的模块就会把它与某个特定的协议（根据 IP 数据报首部中的协议字段来判断）和用户进程（根据包含在 IP 数据报前 8 个字节中的 TCP 或 UDP 报文首部中的 TCP 或 UDP 端口号来判断）联系起来。

下面各种情况都不会导致产生 ICMP 差错报文：[17]

1. ICMP 差错报文（但是，ICMP 查询报文可能会产生 ICMP 差错报文）；
2. 目的地址是广播地址或多播地址的 IP 数据报；
3. 作为链路层广播的数据报；
4. 不是 IP 分片的第一片；
5. 源地址不是单个主机的数据报。这就是说，源地址不能为零地址、环回地址、广播地址或多播地址。

这些规则是为了防止过去允许 ICMP 差错报文对广播分组响应所带来的广播风暴。

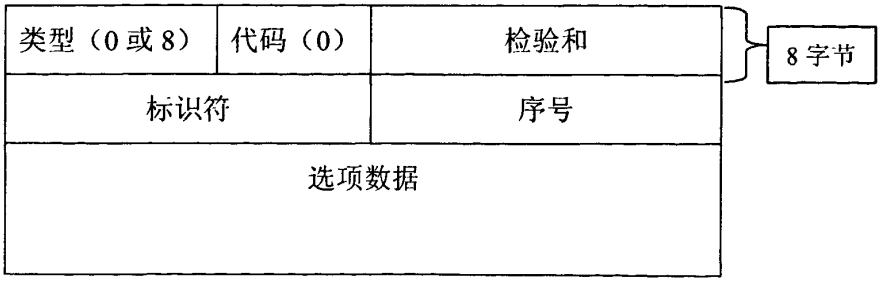
2.3 其他网络拓扑发现工具

网络拓扑发现可采用的几种常见的工具和协议，除了上面介绍过的协议，还有 Ping、Traceroute、DNS 和 ARP 等。他们各具特点，ICMP 中常用的命令 Ping 和 Traceroute 应用最广泛，SNMP 效率最高。

2.3.1 Ping

Ping 程序是对两个 TCP/IP 系统连通性进行测试的基本工具。它只利用 ICMP 回显请求和回显应答报文，而不用经过传输层（TCP/UDP）。Ping 服务器一般在内核中实现 ICMP 的功能。Ping 命令使用的是网际控制报文协议 ICMP，是 IP 的一部分，并且在每个 IP 实现中都必须包含它。[19]ICMP 回显请求和回显应答报文（如图 2-5）被用于判定 IP 地址的可达性。由主机或者路由器向指定目的站发送类型为 8 的 ICMP 回送请求报文。当目的站收到回送请求报文以后，通常将源地址和目的地址反过来，再重新计算校验和，然后生成类型为 0 的回送应答报文返回给发送者。当路由器和主机不能传送报文时，则返回目的不可达报文（类型为 3），并将错误的原因代码报告给源结点。其中，端口不可达报文的代码为 3。

图 2-5 ICMP 回显请求和回显应答报文格式



还有一种 Ping 方式是直接广播 Ping（Directed Broadcast Ping），是指向子网的广播地址发送 Ping 包，子网中的所有主机都会收到 Ping 包，并对其做出回应。直接广播 Ping 可用于发现子网中的所有存活主机。但是由于一些机器 IP 实现上的错误，这种方法可能引起网络中的冲突甚至广播风暴。另外，出于安全性考虑，有些路由器根本不支持直接广播 Ping，所以，可以采用并行方式发送 ICMP 回送请求包，以提高发现速度。

2.3.2 Traceroute

Traceroute 程序是一个能更深入探索 TCP/IP 协议的方便可用的工具。尽管不能保证从源端发往目的端的两份连续的 IP 数据报具有相同的路由，但是大多数情况下是这样的。Traceroute 程序可以让我们看到 IP 数据报从一台主机传到另一台主机所经过的路由。Traceroute 程序还可以让我们使用 IP 源路由选项。

Traceroute[20]的功能是发现一条从主机到目的主机的路径。它使用IP报文中包含的TTL（Time To Live）字段，并试图从沿路由到达目标的每个主机处获得一个ICMP _TIMEEXCEEDED消息。随着试图连接目标的不可达端口，将导致沿路由到达最终目标期间从每个路由器获得一个系统的响应。

Traceroute程序使用ICMP报文和IP首部中的TTL字段。其工作原理如下[21]：首先，Traceroute送出一个TTL是1的IP datagram到目的地，当路径上的第一个路由器收到这个datagram时，它将TTL减1。此时，TTL变为0了，所以该路由器会将此datagram丢掉，并送回一个ICMP time exceeded消息（包括发IP包的源地址，IP包的所有内容及路由器的IP地址），Traceroute 收到这个消息后，便知道这个路由器存在于这个路径上，接着Traceroute 再送出另一个TTL是2的datagram，发现第2个路由器..... Traceroute 每次将送出的datagram的TTL 加1来发现另一个路由器，这个重复的动作一直持续到某个datagram 抵达目的地。当datagram到达目的地后，该主机并不会送回ICMP time exceeded消息，因为它已是目的地了，那么Traceroute如何得知目的地到达了呢？Traceroute在送出UDP datagrams到目的地时，它所选择送达的port number 是一个一般应用程序都不会用的号码（30000 以上），所以当此UDP datagram 到达目的地后该主机会送回一个ICMP port unreachable的消息，而当Traceroute 收到这个消息时，便知道目的地已经到达了。

2.3.3 DNS

域名系统（DNS）是一种用于 TCP/IP 应用程序的分布式数据库，它提供主机名字和 IP 地址之间的转换及有关电子邮件的选路信息。[22]DNS 维持该域内每个主机名字到其它 IP 地址的绑定。大多数域名服务器通过 zone transfer 命令返回该域内名字的列表。因此理论上 DNS 域转换可以发现管理域内的所有主机和路由器，这种技术快速、开销小。但是发现主机并不完全准确，因为用 DHCP 获取 IP 地址的主机并没有 DNS 服务，而且一些网络管理员因为安全原因关闭了 DNS 域转换服务。

2.3.4 ARP

ARP（Address Resolution Protocol）即地址解析协议。在局域网中，网络中实际传输的是“帧”，帧里面有目标主机的MAC地址。在以太网中，一个主机和另一个主机进行直接通信，必须要知道目标主机的MAC地址。但这个目标MAC地址是如何获得的呢？它就是通过地址解析协议获得的。所谓“地址解析”，就是主机在发送帧前将目标IP地址转换成目标MAC地址的过程。[23]ARP协议的基本功能就是通过目标设备的IP地址，查询目标设备的MAC地址，以保证通信的顺利进行。

在每台装有TCP/IP协议的计算机里都有一个ARP缓存表，存放最近获得的IP地址到物理地址的绑定。ARP缓存表采用了老化机制，在一段时间内如果表中的某一行没有使用，就会被删除，这样可以大大减少ARP缓存表的长度，加快查询速度。[24]但是，ARP容易遭受攻击，就是通过伪造IP地址和MAC地址实现ARP欺骗，能够在网络中产生大量的ARP通信量使网络阻塞，攻击者只要持续不断的发出伪造的ARP响应包就能更改目标主机ARP缓存中的IP-MAC条目，造成网络中断或中间人攻击。

2.3.5 拓扑发现工具性能比较

对本章所列出的几种拓扑发现的工具进行比较和分析，见表 2-5

由图可得，以上基于以上网络发现工具的拓扑发现算法将各有利弊。因此，在实际应用过程中，我们需针对网络的具体情况选用适当的网络拓扑发现工具，或是结合多种工具，尽可能发挥这些工具的优势，以获得更高效、准确的网络拓

扑图。

表 2-5 拓扑发现工具的比较

	ICMP Ping	Traceroute	SNMP	DNS	ARP
适用性	所有网络域	所有网络	多数网络域	多数网络域	所有网络域
网络负荷	低	较高	较低	低	低
速度	对于存活主机比较快，否则比较慢	较慢	对于实现了该协议的设备比较快	快	快
准确性	准确	比较准确	比较准确	比较准确	比较准确

2.4 本章小结

本章介绍了网络拓扑发现的一些相关工具及协议。其中，详细介绍了 SNMP 协议，并分别介绍了 ICMP、Ping 程序、Traceroute 程序及 DNS 及 ARP 等网络拓扑发现的工具及协议。对各网络拓扑发现工具的性能进行了分析和比较，结合多种工具的特点，取长补短，为本文提出新的多子网物理拓扑发现算法的改进提供了理论基础。

3 物理网络拓扑发现算法分析

目前异构网络的特点表现为：一个网络往往由多子网构成；设备的多样性以及协议的差异性等。二层网络设备对于端接点和三层设备(如路由器)是完全透明的。交换机在生成树协议中邻接点之间只进行有限的信息交换。它只在地址转发表中保存相关信息，用以将接收的报文分组转发到正确的端口输出，幸运的是多数的二层物理设备通过 SNMP 协议可以获取地址转发表的信息。上一章已经介绍了网络拓扑发现的一些相关工具及协议，在提出改进的异构多子网物理拓扑发现算法之前，本章将就一些必要的背景知识以及相关的物理拓扑发现算法进行介绍。

3.1 交换域背景知识介绍

我们把希望进行拓扑发现的域称为管理域，并且用无向图 N 模型化此管理域网络。在该网络中的结点对应于以下三种网络设备之一：路由器、交换机和主机。我们算法的目的就是尽可能准确的发现 N 中的结点和边（即不同网络设备上一对接口间的物理直连作为 N 中相应结点间的一条边）。我们定义交换域为满足以下条件的最大交换机集合 S：若在 N 中存在连接两台交换机的一条路径，这条路径所经过的交换机都在集合 S 中，且只属于 S。图 3-1 给出的是相应管理域的一个实例图。[9]

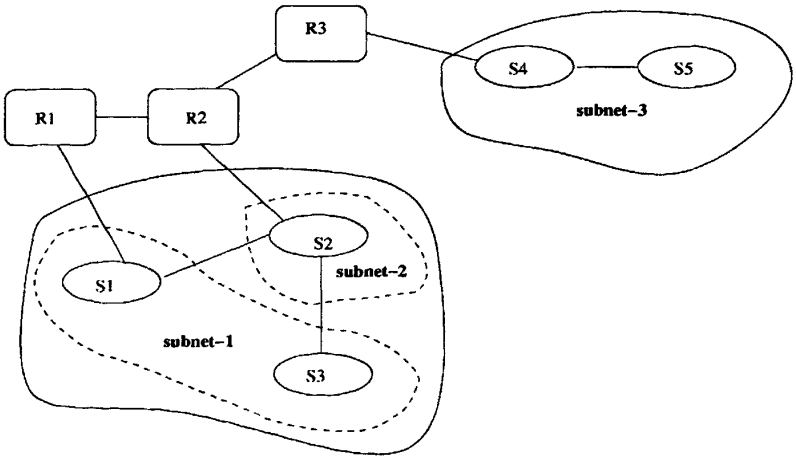


图 3-1 一个典型的管理域网络图

一个交换域可由多个不同子网构成，而且这些子网之间的通信必须经过路由

器。例如，在图 3-1 中，交换域{S_4，S_5}只包含一个子网，而交换域{S_1，S_2，S_3}由两个子网构成，其中一个子网包含 S_1 和 S_3，另一个则包含 S_2。但是，尽管 S_1 和 S_2 之间存在直接的物理连接，S_3 决不会从 S_2 收到以 S_2 为源 MAC 地址的帧，因为若 S_2 必须和 S_3 进行通信，则从 S_2 来的分组首先被发送到 R_2，R_2 再将其转发给 R_1；最后，R_1 再转发以 R_1 为源 MAC 地址的分组到 S_3（尽管该分组经过 S_2）。

在进入下一部分之前，先介绍三个定义：

定义 3.1　S_{ij}：S_{ij} 表示交换机 S_i 的第 j 个接口。

定义 3.2　A_{ij}：对每个接口 S_{ij}，在该接口（通过逆向学习）学到的地址集，也就是 S_{ij} 的 AFT 。A_{ij} 就是在 S_{ij} 接口接收到的帧的源 MAC 地址集合。

定义 3.3　交换机 S_i 的叶子接口是指不与其它任何交换机接口相连的接口。

3.1.1 单子网交换域

3.1.1.1 引理证明

Yuri Breitbart 在文献[9]描述单子网拓扑发现算法之前给出下列假设条件下：1）每个交换域只包含一个子网；2）在管理域中没有 VLAN；3）AFT 是完全的。然后，从给出确定一个交换机设备的某个接口与另一交换机某个接口连接的充分必要条件的引理开始描述发现生成树边的发现算法：

以下是产生算法的引理及引理证明。

引理 3.1：S_{ij} 和 S_{kl} 彼此直连，当且仅当 $A_{ij} \cup A_{kl} = U$ 并且 $A_{ij} \cap A_{kl} = \Phi$。

证明：必要条件，假设 $A_{ij} \cup A_{kl} = U$ 且 $A_{ij} \cap A_{kl} = \Phi$，但 S_{ij} 和 S_{kl} 不直连。因为交换域的活动生成树不包含环路，A_{ij}（A_{kl}）不包含 S_i（S_k）。因而推理得到 A_{ij} 包含 S_k 并且 A_{kl} 亦包含 S_i 。令 P 为该生成树中 S_i 和 S_k 之间的路径。考虑下面两种情况。

1) P 包含 S_{ij} 和 S_{kl}，在这种情况下，P 中存在另一个交换机 S_m，因此，$A_{ij} \cap A_{kl} = \Phi$ 不能成立。

2) P 包含 S_{ij} 和 S_{kl} 中的任一个，或两个都不包含。在这种情况下，因为结点 S_i 和 S_k 是相连的，而且，分别从 S_{kl}（S_{ij}）可到达交换机 S_i（S_k）（因为 $S_i \in A_{kl}$，$S_k \in A_{ij}$），所以，在交换域生成树中 S_i 和 S_k 之间我们找到了两条不同的路径（也就是，一个环路）。这就再次矛盾。

引理 3.2：路由器 R 和接口 S_{ij} 是相连的，当且仅当 S_{ij} 是叶子接口并且 A_{ij}

包含 R 的 MAC 地址。

证明 :若路由器与交换机直连，交换机端口 S_{ij} 直连 S_{kl}，又 S_{kl} 不属于交换机，则 S_{kl} 属于路由器或为空。如果 S_{kl} 为空，则与条件 S_{ij} 与 S_{kl} 直连矛盾，所以 S_{kl} 只能属于路由器的端口，因此与中包含 R 的 MAC。

引理 3.3：令 A_{kl} 是 N 的所有 AFT 中一个最小候选集。那么 A_{kl} 中的每个设备在交换域生成树中只有一个接口。

证明：根据假设得知，A_{kl} 是 N 所有接口的 AFT 中包含 MAC 地址个数最少的 AFT。又 S_t 是地址包含在 A_{kl} 中一个网络结点。我们说 S_t 在生成树中只有一个接口 S_{ti}。相反地，设想，S_t 至少有两个这样的接口；从而，S_t 就不能为叶子结点。因而，存在一个结点 S_p 不是通过 S_{ti} 接口而连到了结点 S_t。但此时 A_{kl} 明显已不是一个最小的 AFT，因为 S_t 存在一个端口，它的 AFT 包含了 A_{kl} 中特有的 MAC 地址子集，而不包含 S_t。所以，S_t 在该生成树中只能有一个接口。

3.1.1.2 算法描述

根据上一节的引理，下面给出具体的单子网物理拓扑发现算法：

Procedure FindInterConnections(S_1, S_2,... ,S_n, R_1, R_2,...,R_m)

/* S_1, S_2,...,S_n are the switches of a subnet S */

/* R_1, R_2,...,R_m are the routeQ of the subnet S */

```
begin
1.  for each switch Si do
2.    for each interface j of Si do {
3.      if（Sij has already been matched）then continue
4.      else if（ Aij∪Akl=U and Aij∩Akl=Φ) then
5.        match Sij with Skl  /* Sij and Skl are connected*/
6.  }
7.  for each router Rk，  for each switch Si do
8.      for each interface j of Si do
9.        if（Sij is not matched and Aij contains Rk）then
10.         match Sij with Rk /* Sij and Rk are connected*/
end
```

3.1.2 多子网交换域

正如本章开头所描述，现在的异构网络往往由多个子网构成。上一节介绍了单子网交换域的物理拓扑发现算法，这些定理在单子网内成立，但对多子网却不总是成立，因此，本节将就多子网交换域的物理拓扑发现算法进行介绍。

3.1.2.1 引理证明

结合文献[9]，根据多子网交换域的一些性质：假设 S_i 与 S_k 是属于不同子网的两个结点；A_{ij} 包含 S_k 当且仅当存在一个与 S_k 属于同一子网的结点 S_p 满足条件：在生成树中存在路径 S_p，…，S_i，…，S_k。令 U_{ijkl} 表示 $A_{ij} \cup A_{kl}$ 中的 MAC 地址集合。我们将列出一下三个引理：

引理 3.4：假定 S_{ij} 和 S_{kl} 是不同的接口，若 $A_{ij} \cap A_{kl} \neq \Phi$，则接口 S_{ij} 与 S_{kl} 不能互连。

证明：反证法，假设在 A_{ij} 和 A_{kl} 中都出现结点 S_p，且 S_{ij} 与 S_{kl} 相连。那么，通过 S_k 从 S_p 到 S_i 存在一条路径并且通过 S_i 也存在一条从 S_p 到 S_k 的路径。然而，两条路径都属于该生成树，这样就产生了矛盾。因此，当 $A_{ij} \cap A_{kl} \neq \Phi$（如果两个接口有一个非空交集）时这两个接口不能互连。

引理 3.5：设 t 为至少包含两个结点 S_p 和 S_q 的一个子网。若 $A_{ij} \cap A_{kl} = \Phi$ 并且 U_{ijkl} 包含 S_p 或者包含 S_q 但不同时包含两者，则接口 S_{ij} 与 S_{kl} 不能互连。

证明：假设 S_{ij} 和 S_{kl} 互连。不失一般性，设 $S_p \in A_{ij}$。因而，在生成树中必定存在一条通过 S_k 的从 S_p 到 S_i 的路径。我们考虑下面两种情况。

1) 生成树中从 S_q 到 S_i 的路径不通过 S_k。在这种情况下，$S_q \in A_{kl}$，因生成树中由 S_q 到 S_p 的路径必须通过 S_i 和 S_k，而且 S_q 和 S_p 属于同一个子网 t。如图 3-2：

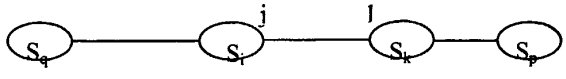


图 3-2

2) 生成树中由 S_q 到 S_i 的路径经过 S_k。在这种情况下，由于 $S_p \in A_{ij}$，必定存在一个结点 S_r 使 S_p，…，S_k，S_i，…，S_r 成为生成树中一条路径，而且 Sr 也属于子网 t，如图 3-3。这样，S_q，…，S_k，S_i，…，S_r 必然也是该树中一条路径，且 S_q 必定也属于 A_{ij}。综上所述，若 S_{ij} 与 S_{kl} 相连，则 S_p 与 S_q 都一定属于 U_{ijkl}，而在定理所述条件下 S_{ij} 与 S_{kl} 不相连。如图 3-4。

生成树中出 S_q 到 S_i 的路径经过 S_k，$S_p \in A_{ij}$

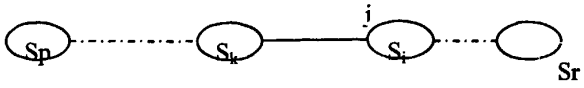


图 3-3

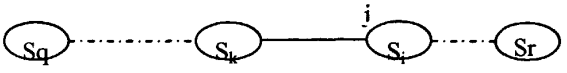


图 3-4

引理 3.6：设 $A_{ij} \cap A_{kl} = \Phi$ 并且 $A_{ij} \cap A_{pt} = \Phi$。若 $U_{ijkl} = U_{ijpt}$，且 S_i 和 S_k 属于同一个子网，但与 S_p 属不同子网，则 S_{ij} 与 S_{kl} 不相连。

证明：假设 S_{ij} 与 S_{kl} 互连。注意，因为 $A_{ij} \cap A_{kl} = \Phi$，$A_{ij} \cap A_{pt} = \Phi$ 且 $U_{ijkl} = U_{ijpt}$，所以，$A_{kl} = A_{pt}$。又因 S_i 和 S_k 属于同一个子网，$S_i \in A_{kl}$，所以，$S_i \in A_{pt}$。因而，在子网中必定存在一个与 S_i 属于同一子网的结点 S_r 使 S_i ，S_k，…，S_p，…，S_r 是生成树中的一条路径。但是，因为 S_i 与 S_k 属于同一个子网，从而得出 $S_k \in A_{pt}$，导致 $S_k \in A_{kl}$；显然，这是不可能的，所以我们找到了矛盾。

3.1.2.2 算法描述

结合上一节的定理，我们把满足下列条件的可能匹配的接口对 S_{ij} 和 S_{kl} 其 AFT 的并集 U_{ijkl} 称为有效并集 U_{ijkl}

1）$A_{ij} \cap A_{kl}$ 为空；

2）对每个子网 t，$A_{ij} \cup A_{kl}$ 或者包含子网 t 的所有结点或者一个结点也不包含；

3）若 S_{ij} 和 S_{kl} 属于同一个子网，则不存在来自不同子网的结点 S_p 使 $U_{ijkl} = U_{ijpt}$，并且使 $A_{ij} \cap A_{pt} = \Phi$ 成立。

据此，多子网物理拓扑发现的算法如下[9]：

1）生成初始有效并集 U。

2）重复执行下一步直到没有可从 U 中删除的有效并集为止。

2.1）若接口 S_{kl} 只出现在 U 中的一个有效并集 U_{ijkl} 中则在 S_{ij} 和 S_{kl} 之间建立连接，然后执行 2.1.1）从 U 中删除所有包含 S_{ij} 的有效并集（除 U_{ijkl}）；接着，2.1.2）删除满足如下条件的所有有效并集 U_{ixpy}，其中（x，p，y）≠（j，k，l）且 S_p 是任何使用现有的连接（包括 S_k 自己）可以到达 S_k

的结点。

2.2）如果每个 S_{kl} 在两个以上有效并集中被发现则选择 U_{ijkl}，使 S_{kl} 在有效并集 U 中出现的次数在所有（结点、接口）集合中最少。否则，任选一个 U_{ijkl}。在任一情况下，选择完成后重复执行上述步骤 2.1.1）和 2.1.2）。

3）对每个在 U 中的被选的有效并集 U_{ijkl}，得到结果“S_{ij} 与 S_{kl} 相连。”

3.2 物理拓扑发现方法

物理子网交换域按范围可分为单子网和多子网，上一节已经就此展开介绍，接下来，将就不同的发现方法对物理子网的拓扑发现进行介绍和分析。其中，包括：基于 SNMP 的物理拓扑发现算法、基于 STP 的物理拓扑发现算法、基于 ICMP 的物理拓扑发现算法和基于 AFT 的物理拓扑发现算法等。

3.2.1 基于 SNMP 的物理网络拓扑发现算法

SNMP 逻辑拓扑发现算法主要是利用存储在路由器 MIB 库中的路由表信息和 ARP 表信息来发现新地址。路由表在 MIB-II 中被定义为 ipRouteTable，它主要包含路由器的路由 IP 地址和 ARP 地址等地址表单。[25]主要用到的是 ipRouteNextHop 项，它描述了此路由器的下一级路由也就是和它相连的路由器；另一个就是 ARP 表，它描述了与路由器连接的活动主机，也就是子网中的活动主机。

在物理拓扑发现算法中，依然要依靠 SNMP 协议，在原有的网络层拓扑的基础上增加交换机到交换机、交换机到路由器以及交换机到主机之间的连接关系。以太网中，网桥为了完成数据帧的转发工作，各自维护自己的生成树状态表(STP)和 MAC 地址转发表(FDB)，并且保存在了标准的 BRIDGE MIB 中。[26]

3.2.1.1 Bridge MIB

RFC1493 主要定义了 IEEE 802.1D-1990 关于网桥信息在 MIB-II 中的实现，又被称为 Bridge MIB。现在，Bridge MIB 已经在绝大多数厂家交换机产品中广泛部署。在实践中，我们也以网络节点是否支持 Bridge MIB 作为判断网络节点是否为交换机的方法，在 RFC 1493 中有对该 MIB 中定义的对象的详细说明。

在物理网络拓扑结构发现中主要使用的 MIB 对象是：

1. interfaces 组

interfaces 组属于 MIB-II，在 2.1.3 中已经有了简单的介绍。对于路由器每个接口被认为是附属于一个子网，对于交换机接口对应一个点到点的链路。该组包含一个 ifNumber 对象。记录一个网络设备的所有接口的总数(不管这些接口处于什么状态)。该组还包含一个 ifTable 表对象，每个接口在表中占用该表的一个条目。表 3-1 显示了 ifTable 表中包含的对象及说明。

表 3-1 ifTable 表对象

对象名称	类型	存取权限	状态	描述信息
ifDescr	DisplayString	read-only	mandatory	接口描述，包括厂商、名称等
ifType	INTEGER	read-only	mandatory	接口类型
ifMtu	INTEGER	read-only	Mandatory	该接口能够发送和接收的最大协议数据单元的大小
ifSpeed	Gauge	read-only	mandatory	接口以 b/s 为单位的数率
ifAdminStatus	[1..3]	read-write	mandatory	接口的管理状态：1-正常、2-关闭、3-测试

表 3-2 ipNetToMediaTable 表对象

对象名称	类型	存取权限	状态	描述信息
IpNetToMediaIfIndex	INTEGER	read-write	m	对应的接口索引
IpNetToMedia PhysAddress	PhysAddress	read-write	m	物理地址
IpNetToMediaAddress	ipAddress	read-write	m	IP 地址
IpNetToMediaType	[1…4]	read-write	m	映射的类型：1-其他、2-无效的、3-动态的、4-静态的

2. ip 组

在 2.1.3 中同样对 ip 组的进行了简单的介绍。下面将介绍该组所包含的物理网络拓扑发现中需要的表对象 ipNetToMediaTable（地址转换表）。ipNetToMediaTable 记录了从 IP 地址到物理地址的映射，通过检查设备的 ipNetToMediaTable 表，可以发现与该设备相连的其他设备的 IP 地址和物理地址的映射以及映射方式，由此可以发现设备之间的一些连接情况。表 3-2 显示了 ipNetToMediaTable 表中包含的对象及说明。

3. dotldTp 组

dotIdTp组在RFC1493中定义，它在MIB-II中的位置是{1.3.6.1.2.1.17.4}。在物理拓扑发现中需要读取交换机端口的MAC地址转发表中的内容就存储在dotldTp组下的dotldTpFdbTable表对象中，在这个表中存储了网桥转发与过滤数据帧的信息。dotldTpFdbTable表在MIB-II中的位置是{1.3.6.1.2.1.17.4.3}，它所包含的列对象如表3-3所示[27]。

表 3-3 dotldTpFdbTable 表对象

对象名称	类型	存取权限	状态	描述信息
dotldTpFdbAddress	PhysAddress	read-only	m	已获取端口信息的主机 MAC 地址
dotldTpFdbPort	INTEGER	read-only	m	拥有 dotldTpFdbAddress 的主机所出现的交换机的端口号
dotldTpFdbStatus	INTEGER	read-only	m	端口转发状态：1-特殊情况、2-无效、3-正常、4-表明 dotldTpFdbAddress 是交换机自身地址、5-表明 dotldTpFdbAddress 是静态 MAC 地址

3.2.1.2 算法描述及分析

定义 3.4　标志结点：当算法运行主机在欲发现物理拓扑结构子网中时，将此主机命名为标志结点，若不在，则将目标子网中能转发运行算法主机发送的数据包的路由器结点定为标志结点。

定义 3.5　上行端口：指端口对应的地址转发表中出现标志结点 MAC 地址的端口。

定义 3.6　下行端口：指端口对应的地址转发表中没有出现标志结点 MAC 地址的端口。

本节介绍基于 SNMP 的物理网络拓扑发现算法，主要有三个任务，第一个任务是探测子网内所有的节点，包括交换机和主机终端，第二个任务是判定交换机与交换之间的连接关系，第三个任务是判定交换机与主机终端的连接关系。第一个任务可以用 Ping 等 ICMP 协议工具实现[28]。文章主要描述后两个任务目标。首先，定义几个元素不重复的集合：

（1）子网内存在所有节点的集合 U-node，里面的元素包括

交换机和主机终端。

（2）子网内所有交换机的集合 U-Switch。

（3）子网内所有终端主机的集合 U-Host。

算法步骤如下[29]：

把上述所有集合赋值为空；

探测获取子网内所有活动节点，并把这些节点不重复地放入集合 U-node；

while(U-node≠∅)

{

从 U-node 取出一个节点 CurrentNode；

if（ CurrentNode 不支持 SNMP 协议）

把 CurrentNode 放入主机集合 U-node；

else

{

访问该节点的 MIB 库中的 SysService 表项判断给节点是否为交换机;如果是，把 CurrentNode 放入交换机集合 U-Switch；否则还是把 CurrentNode 放入主

机集合 U-Host；

}

把 CurrentNode 从集合 U-node 中去除；

}

Ping 子网内所有的交换机；

依次获取 U-Switch 中每个交换机的 MIB 库地址转发表及其本身的 MAC 地址；

while(U-Switch≠∅)

{

取出一个交换机 CurrentSwitch；

根据已知的标志节点 MAC 地址，分析 CurrentSwitch 每个端口的

地址转发表；

找出每个交换机的唯一的上行端口 S_{kl}、下行端口和本地端口的集合；

if(下行端口集合≠∅)

{

令 TempMacs=下行端口集合所有的端口地址转发表合集+本交换机的 MAC 地址；

依次搜索其他交换机的每个下行端口地址转发表，必有一个与 TempMacs 相同，

此时那个交换机的下行端口与 S_{kl} 直接相连；

}

else

{

依次搜索其他交换机的每个下行端口地址转发表；

必有一个与本交换机的 MAC 地址相同；

此时那个交换机的下行端口与 S_{kl} 直接相连；

}

}

算法分析：

上述算法是基于 SNMP 协议的二层拓扑发现，通过获取 MIB 信息进行拓扑发现。此拓扑发现方法基于标准的 SNMP 协议来实现，发现过程的算法简单，系统和网络开销小，所以效率较高。但是，此法存在以下缺陷:

(1)无法发现网络中不支持 SNMP 协议或没有安装 SNMP 代理的网络设备。

(2)路由表中包含了大量的冗余信息。

(3)一个路由往往对应多个 IP 地址(每个端口对应一个 IP)。而路由表是以唯一的 IP 地址做为记录因素,这样就存在多目路由的问题，不利于直观反映网络拓扑的连接情况。所以此法适合于发现网络中的主干拓扑，反映网络的整体状况。

3.2.2 基于 ICMP 的物理网络拓扑发现算法

基于 ICMP 协议的常用工具就是 Ping，通过 Ping 发送 ICMP，可以测试网络设备的活动状态和可达性。如果对一个网段内所有可能网络 IP 地址依次执行 Ping 操作，根据应答可以判断网段内的所有活动的网络设备。通常进行拓扑发现还要结合 Traceroute 工具，用来分析网络的连接情况。[30]其详细信息在第二章已经介绍过。

在具体实现中，往往是利用 Ping 依次检测 IP 地址区间的每一个 IP 地址，然后对 Ping 得到的结果依次进行 Traceroute 操作,记录其结果，得到到达该 IP 的路由。[31]通过发送掩码请求报文得到子网掩码，将每个 IP 加入相应的子网。

在描述基于 ICMP 拓扑发现算法之前先对如何确定网络设备的状态、设备类型和接口类型三个方面进行简单介绍：

(1) 确定网络设备的状态；

(2) 确定网络设备类型；

(3) 确定网络设备的网络接口。

拓扑发现算法步骤如下[32]：

1.初始化探测路由连接列表，IP 地址列表，路由器列表，子网地址表；

2.利用 Ping 检测范围内每个 IP 地址，将活动的 IP 地址加入到 IP 地址列表中；

3.从 IP 地址列表中取出一个 IP 地址作为目标地址，对此目标地址进行 Traceroute 操作；

4.将 Traceroute 返回的所有连接加入到路由连接列表，把路由器添加到路由器列表中;

5.如果 IP 地址列表中还有未探测的目标地址，则重复步骤 3-4;

6.对 IP 地址表中每个 IP 地址，通过发送子网掩码请求报文与接收掩码答应报文，找到 IP 地址所在子网的掩码;

7.根据子网掩码，确定对应每个 IP 地址的子网地址，把所得各个子网加入到子网地址表中;

8.检测多址路由器，合并隶属于同一路由器的不同地址;

9.结束。

算法分析:

几乎所有的基于 TCP/IP 协议的网络设备都支持 ICMP 协议,所以此方法实现起来也比较容易。并且在发现网络拓扑的同时,检测了网络设备的活动状态,因为只有活动的网络设备才会发送 ICMP 响应报文。此外,对可能因为依次 Ping 各个 IP 地址而带来的管理工作站和网络负担增大的问题,可采用广播方式或多线程方式来减小这一开销。

当然此法也存在一些缺陷:

(1)Ping 操作可以较容易地发现活动的网络设备,但是不能确定 ICMP 包从源 IP 节点到目的 IP 节点所经过的路由设备。在使用 Traceroute 操作后可以知道中间经过了哪些路由设备,但是路由设备之间的连接情况却很难分析。

(2)发现方法本身也具有一定的盲目性,对于给定的 IP 地址,即使 Ping 通了,也很难判断各个 IP 所在子网的情况。

(3)由于一些防火墙禁止通过 ICMP 包,该方法无法检测存在于这些防火墙后面的设备。因此,基于 ICMP 的拓扑发现方法可以快速发现网络设备,但构建网络拓扑设备关系图比较困难。所以在实际应用中,主要用此方法来发现子网中的网络设备。

3.2.3 基于 AFT 的物理网络拓扑发现算法

3.2.3.1 引理证明

Yuri Breitbart 基于 AFT 的物理拓扑发现算法在 3.1.1.2 中已经介绍到，并且，对于支持此算法的相应的概念引理及证明也在 3.1.1 中有详细介绍。

3.2.3.2 算法描述及分析

基于已有的定义及引理证明，设计了一个链路层拓扑发现的算法，算法定义了4个队列：路由器队列、链路层设备队列、交换机队列和拓扑发现（临时）队列，以下是算法的伪代码[33]：

```
routerList= FindRouteQ InSubnets (D) ;                    //Step 1
for (each r in routerList) {                              //Step 2
  dv= retrieveMIB (r) ; addElement (dv,deviceList); }
for(each d in deviceList) {                               //Step 3
  if(run- agent(d) && isSwitch(d)) {
    if(retrieveBridgeMIB(d,"do t1dStpDesignateRoot")= md) {
  makeFlag(d,"sw itch" ); addElement(d,switchList); confirm
Flag ("false",d,switchList);
}
removeElement(d,deviceList);}}
ping-switch(switchList);                                  //Step 4
for(each i in switchList) retriveBridgeMIB(i) ;           //Step 5
Mi=getMACSubnet(Ni) ;
clearList(discoverList); s=findRootSwitch(switchList) ;   //Step 6
confirmFlag ("true" , s, switchList) ; addElement(s, discoverList) ;
drawDevice(s) ;
while (!empty(discoverList) ) {                           //Step 7
  Si=removeElement(discoverList) ;
for(each port j of switch Si) {                           //Step 8
  Aij= FindMACSet(j, Si) ;
if(find-match (Aij, switchList, Sk) ) {                   //Step 8.1
  S1= findSubnet(Si) ; S2= findSubnet(Sk) ;Akl= FindMACSet
(l, Sk) ;
if (S1= S2&&Aij ∪ Akl= M   Si&& Aij∩Akl=∅ ) {             //(1)
    addSSLink (Sij, Skl) ;
```

}else if (S_1!= S_2&&A_{ij}∩A_{kl}= ∅) { //(2)

addSSLink (S_{ij}, S_{kl}) ;

}else{continue; }

confirmFlag (" true" , S_k, switchList) ; //(3)

addElement (S_k, discoverList) ; drawDevice (S_k) ;

}else{ //Step 8.2

find- match (A_{ij}, deviceList, h) ;

addSHL ink (S_{ij}, h) ; drawDevice(h) ; }}

} //Step 9

以下是对上述伪代码的部分算法描述：

Step5：访问队列 switchList 中的每一个交换机的 Bridge MIB 库，提取其 AFT 地址转发表(dot1dTpFdbTable)，端口数目(dot1dBaseNumports)和端口表(dot1dBasePortTable)等信息，M_i 依次为各子网 N_i 内设备的 MAC 地址集合；

Step8：对交换机 S_i 的每一个端口 j，提取该端口学习到的所有 MAC 地址集合 A_{ij}；

Step8.1：若集合 A_{ij} 存在与 switchList 中某设备 S_k 的 MAC 地址相匹配，进一步统计 S_k 的各个端口 l 学习到的所有 MAC 地址集合 A_{kl}；

(1)如果 S_i 与 S_k 属于同一子网，同时 $A_{ij} \cup A_{kl}=M_{si}$ 且 $A_{ij} \cap A_{kl}=\varnothing$，$M_{si}$ 为子网内设备的 MAC 地址集合，则交换机 S_i 的端口 j 与交换机 S_k 的端口 l 直接相连，转(3)；

(2)如果 S_i 与 S_k 不属于同一子网，但由于 A_{ij} 包含 S_k 的 MAC 地址，且 $A_{ij} \cap A_{kl}=\varnothing$,则交换机 S_i 的端口 j 的端口 1 直接相连；

(3)在队列 switchList 中将 S_k 的“确定”标志置为 true,并将其加入到队列 discoverList 中，在拓扑图中绘出交换机 S_k，转 Step8；

Step8.2：若集合 A_{ij} 每一个 MAC 地址与 switchList 中所有设备的 MAC 地址都不匹配，再从队列 deviceList 中查找物理 MAC 地址与之匹配的设备 h，若存在，则交换机 S_i 的端口 j 与设备 h 直接相连，在拓扑图中绘出设备 h；转 Step7。

算法分析：

在以太网中网桥与交换机遵循以太网生成树协议，它们的每个端口可以学习

到由它转发过的数据帧的源 MAC 地址。将这些地址与学习到的端口对应起来保存到自己的 SNMP MIB 的地址转发表 Address Forwarding Table 中，以便查询。基于 AFT 拓扑发现方法在判断上运算量大，而且也要求每个交换机都支持 SNMP，而实际的底层网络元素经常不会统一地支持地 SNMP。例如，对于网络中存在 hub 或哑交换机的情况，算法就不能顺利进行。任何一个交换机都是基于 AFT 而进行数据转发的，因此基于 AFT 拓扑发现算法要求所需要的 AFT 是可访问的，即支持 SNMP 的。这在一定程度上限制了其普适性。

3.2.4 基于 STP 的物理网络拓扑发现算法

3.2.4.1 生成树协议

生成树协议（STP, Spanning Tree Protocol）是由 DEC 公司开发的一个网桥到网桥的协议，它随后被 IEEE802 委员会修订并发布在 802.1d 规范中。生成树协议的目的是维持一个无回路的网络。一条无回路的路径是这样实现的：如果一个设备在拓扑中发现了一个回路，它将阻塞一个或多个冗余的接口，并不断地扫描网络。如果出现一个故障或是添加一个链接，交换机会很快地发现。[34]当网络拓扑发生变化时，生成树协议将重新配置交换机的各个接口以避免链接丢失或出现新的回路。

生成树协议定义了以下一些概念：

根桥 Root Bridge

根端口 Root Port

指定端口 Designated Port

路径开销 Path Cost

定义这些概念的目的就在于通过构造一棵自然树的方法达到裁剪冗余环路的目的同时实现链路备份和路径最优化，用于构造这棵树的算法就叫做生成树算法(Spanning Tree Algorithm，SPA)，用这种算法构造网络树的协议也就被称为生成树协议。[35]要实现这些功能网桥之间必须要交换一些信息，这些信息交流单元就称为网桥协议数据单元（Bridge Protocol Data Unit，BPDU），这是一种二层数据帧，它指向的目的地址是 MAC 多播地址 01-80-C2-00-00-00，所有支持 STP 协议的网桥都会接收到该数据帧，其中的数据区里携带了用于生成树计算的所有有用信息。通过这些信息，加上生成树协议的算法就可以达到生成一个无环路拓

扑。

本节介绍的物理网络拓扑发现算法是基于STP协议的，需通过SNMP访问交换机的Bridge MIB获得。在3.2.1.1中已经对BRIDGE MIB进行了介绍， Bridge MIB 中有关STP 的信息如表3-4所示。

表 3-4 Bridge MIB II 中 OID 对象

名称	说明
dot1dBaseBridgeAddress	本网桥MAC地址
dot1dStpPort	接口编号
dot1dStpPortDesignatedBridge	接口指定网桥
dot1dStpPortDesignatedRoot	接口指定根网桥
dot1dStpDesignatedRoot	指定根网桥
dot1dStpPortState	接口状态
dot1dStpPortDesignatedPort	指定网桥接口

这里需要说明几点：(1) dot1dBaseBridgeAddress通常为该网桥上最小的接口MAC地址，在生成树协议里，该MAC 地址加上2字节的优先级作为网桥ID。(2)在获得由dot1dStpDesignatedRoot或dot1dStpPortDesignatedBridge 指定的网桥ID时,需要去掉头两字节的优先级才为网桥MAC 地址。(3)在不存在VLAN情况下，dot1dStpDesignatedRoot与dot1dStpPortDesignatedRoot指定的根网桥为同一个，设计算法时仅考虑dot1dStpDesignatedRoot即可。反之，则要综合考虑。

3.2.4.2 算法描述及分析

首先给出算法使用的数据结构[36]：

```
SWITCHER:
UINT   nIndex;          //交换机编号
UCHAR mac[6];           //交换机物理地址
ULONG dwIp;             //交换机 IP 地址
ULONG dwSubOffset;      //下级接口位置偏移
HOST:
```

```
UCHAR mac[6];          //主机物理地址
ULONG dwIp;            //主机 IP 地址
INTERFACE:
UINT   nIndex;         //接口编号
BOOL   bIsRootPort;    //是否为根接口
int    nState;         //接口状态
UCHAR DesignedSW[6];   //指定网桥（或为路由器）
UINT   nPort;          //指定网桥的指定接口
UCHAR bySubDataPart;   //连接区域（接口域或主机域）
int    nLinkType;      //0 为无连接，1 为直连，2 为连 Hub
```

广度优先遍历的生成树拓扑算法按广度优先遍历、先主干后备份的顺序进行，算法流程如下[36]：

步骤 1 分配内存，初始化文件数据区；

步骤 2 访问子网内每个主机，辨认出交换机的 IP 及其代表 MAC 地址；

步骤 3 取出链表中的交换机 IP，利用 SNMP 访问并记录各指定接口；

步骤 4 对生成树作广度优先遍历，首先将根网桥信息加入交换机数据区，创建类型为 SWITCHER 的搜索指针指向它；

步骤 5 将搜索指针指向的交换机 MAC 作为搜索种子，在临时链表中找出根接口为搜索种子的交换机信息逐一添入交换机数据区；

步骤 6 搜索指针+1，重复步骤 5，直至为 NULL；

步骤 7 若临时链表中有交换机存在，则存在哑交换机，转步骤 8，否则转步骤 9；

步骤 8 在临时链表中查询每个交换机根接口的指定网桥，若该网桥在临时链表中不存在，由结论 1 将该网桥加入交换机数据区（填入编号、MAC 地址即可，无下级信息）。同时将搜索指针指向它，重复步骤 5、步骤 6、步骤 7。

步骤 9 查询交换机数据区中每个交换机接口的指定网桥和指定接口，若存在相同的交换机，说明它们通过集线器与指定网桥相连，否则为设备直连，修改指定网桥的指定接口类型；

步骤 10 进行数据区遍历，查询交换机接口信息；

步骤 11 对哑交换机每对接口的指定网桥进行级连排除；

步骤 12 对每个交换机的阻塞接口查询它的指定网桥，将连接填入对应的接口信息区，重复步骤 9；

步骤 13 MPING 全网主机，访问直连两交换机接口的地址转发表；

步骤 14 访问所有交换机的接口转发表，填入主机信息。

算法分析：

基于 STP 的拓扑发现算法最终结果是实际的连接，它能发现大多数的哑交换机并获知它们的物理地址和连接接口。其算法的时间复杂度相对简单，由于端口的生成树信息相比端口的地址转发表信息要少的多。因此，相对于基于 AFT 的物理拓扑发现算法更简单，效率更高。但是，由于算法基础的限制，无法直接发现网桥和主机、路由器等网络设备的连接情况。

3.3 本章小结

本章对现有的物理网络拓扑发现算法进行了简单的介绍和分析。首先，对拓扑发现单子网和多子网交换域进行了介绍；其次，介绍了与本文提出的算法相关的现有物理网络拓扑发现算法，并分析各自的优缺点。

4 改进的多子网物理拓扑发现方法

对于多子网拓扑发现算法的主要复杂性在于多子网的组织结构。由于网络规模的不断扩大，在网络中会引进越来越多的交换机来增大网络的规模，而且现代的交换网通常不仅仅只包含一个子网，而是包括多个子网，同一子网中的网络设备可以直接通信(也就是说，不需要经过路由器)。[37]但是对于不同子网的网络设备之间要想通信必须经过各自子网的路由器，而且不同子网中的网络设备通常直接彼此相连。这样就为我们多子网的物理拓扑发现算法带来了难度，因为这就意味着直接相连网络设备可能彼此是完全透明的。

上一章的各个算法都有其各自的优势及适用的网络环境，同时，由于算法基础的限制，自然也有各自的缺陷。取长补短，更好地利用各算法的优势以便对物理拓扑发现算法进行高效的改进。

本文提出的算法是以 Yuri 提出的拓扑发现算法为基础，并结合上述各算法的一些优势而提出改进的多子网物理拓扑发现算法。新算法结合 AFT、STP 和 SNMP 等拓扑发现工具及协议，支持异构，并支持对哑设备(hub 等)的发现，具体的算法提出和实现将在本章和第五章中介绍。

4.1 可行性分析

Yuri Breitbart 提出了一种在异构 IP 网络中的物理拓扑发现算法。利用 Bridge MIB 的转发表中的数据和网桥生成树协议，提出了一个判断两设备直连的引理。该方法适用于包含多个 IP 子网的桥接以太网的拓扑发现。但是这个算法要求完全的 MAC 地址转发表的条件在实际网络中很难实现。

Bruce Lowekamp[38]在 REMOS 项目中采用了一种改进的算法，提出了判断两设备连通的引理以及最小必要条件引理，它不需要转发表是完全的，而且这个算法可以处理共享网段，这些特点使该系统更适合于实际网络，而且对于大规模网络的拓扑发现的效率也很高。

郑海[39]提出的算法，通过端口地址转发表是否保存了一个特殊的标志记录，把交换机的端口分为上行和下行两类端口，提出了一种判定上行端口和下行端口直连的引理，从而解决了上行端口的地址转发表的记录不完整问题。

改进的多子网物理拓扑发现算法结合上述及第三章的算法的优点，基于

AFT、STP 和 SNMP 等拓扑发现工具及协议并利用提取的 MIB 及 Bridge MIB 信息实现，算法可以应用于异构以太网环境中。以下是实现此算法和完成拓扑发现应注意的问题：

1) AFT 完整性的解决。因为此算法假设 AFT 是完整的，而 AFT 的表项在一个的时间间隔将会进行更新，因此，除非一个设备不断的在比更新时间间隔短的时间内收到来自源地址的分组，否则该设备就会删除对应那个源地址的表项；
2) 对不支持 SNMP 的设备的处理。 SNMP 定义了一系列标准的管理信息库，包括MIB、Bridge MIB等信息，但是并不是所有的设备都支持SNMP；
3) 子网中哑设备(如 hub 等)的判断。

4.2 关键技术

在 4.1 小节的可行性分析中，我们已经提出了实现改进算法所需注意及解决的几个问题。下面将给出解决方案。

4.2.1 AFT 完整性的解决

在上一节曾提到，AFT 的表项在一个的时间间隔将会进行更新，除非一个设备不断的在比更新时间间隔短的时间内收到来自源地址的分组，否则该设备就会删除对应那个源地址的表项。因此，AFT 的完整很难保证。

接下来将介绍解决 AFT 完整性的两种方法。第一种技术方法试图通过在交换域节点中产生额外流量，或者通过不断地收集 AFT 有关信息，尽量保持 AFT 的完整。第二种技术方法使用近似法和试探法处理完整性的小偏差。

第一种解决办法通过在交换域内节点对之间产生额外流量的方法努力来保证 AFT 尽可能是完整的，从而，不让 AFT 的记录超时。从节点 X 到节点 Y 产生流量的机制是指从一个网络管理工作站生成一个 ICMP（Echo Request）消息送给 ICMP 分组中以 Y 的 IP 地址为源地址的节点 X。这实质上需要做的就是创建一个好像是由节点 Y 引发的，而实际上是由管理工作站发出的“带有欺骗性的”ICMP ping 分组。这会使得节点 X 去响应节点 Y 的回显请求（Echo Request）。这个方法还存在一些潜在的问题。首先，当网络中的交换机通过“带外”接口连接时，在这样的交换机之间传送的 ICMP 消息不会像我们算法所要求的那样激活

那些“带内”接口的 AFT。为了处理这样的情况，我们的实现依靠在同一子网中的主机和交换机之间发送 ICMP 消息来确保使用“带内”接口。其次，基于安全考虑，网络管理人员经常对一些节点屏蔽 ICMP ping 命令，这就意味着，这些虚假的 pings 并不会有助于激活交换机的 AFT。最后，即使是在 ping 命令可以使用的情况下，请求消息的数量也造成了像基础交换域规模不断增长那样的潜在的问题。为了消除这个不利的影响，我们的实现限制只在属于同一个子网（通常规模不会太大）的节点之间交换 ping 消息；此外，在子网较大的情况下，我们把子网分成若干个小的分段，类似的为这些分段完成激活 AFT 的任务。

另外一种方法不断地对设备 AFT 发出请求（即以固定时间间隔）尽量获得尽可能完整的地址转发信息。我们的数据收集程序以固定的时间间隔复制交换机的 AFT，而不像交换机那样定期删除那些超时记录。经验表明一个正常的交易日过后，AFT 通常差不多是完全的。注意，对支撑的交换域生成树拓扑变化的事件（例如，由于故障），这种方法可能导致错误的连接关系，因为，有些交换机地址可能从设备的 AFT 中真的被删掉了。

4.2.2 不支持 SNMP 的设备的处理

确定交换机与交换机之间的连接关系是发现物理网络拓扑的关键，目标就是找到理想的算法发现物理网络拓扑。简单网络管理协议 SNMP 提供了从交换机中获取信息的统一接口，并且定义了一系列标准的管理信息库，其中就包括了 MIB-II 和 Bridge MIB。通过访问这两个管理信息库，可以获知交换机的系统属性和转发表。其中转发表是用于指导网桥对到达的数据包进行转发的依据。利用这些信息就完全可以确定以太网拓扑结构。

然而，并不是所有的设备都支持 SNMP，这给拓扑发现增加了难度。处理方法如下：对不支持 SNMP 的设备进行端口流量分析处理，通过协议分析工具 ethereal 取出交换机各端口的数据包进行分析，包括该数据包的源地址、目的地址、所属协议等。

4.2.3 哑设备的判断

首先，对哑设备、上行端口及下行端口等仍采用第三章的定义 3.4 至定义 3.6 和本章的定义 4.1、4.4 及 4.5。

对于哑设备的判断如下：当交换机的下行端口上，仍存在多个网络设备的MAC地址时，若交换机A下联端口A_i的入流量等于这些交换机上联端口出流量之和，则各交换机之间存在HUB，否则存在哑交换机。

设哑设备下联N台交换机和M台哑设备，各交换机接收和发送的流量分别记为L_{i_in}和L_{i_out}，集线器和主机的接收和发送流量分别记为L_{Hj_in}和L_{Hj_out}，M台哑设备接收和发送的总流量记为L_{h_in}和L_{h_out}，上联交换机的接收和发送的流量为分别为L_{in}和L_{out}，则:

$$L_{h\_in}=\sum_{j=1}^{m} L_{Hj\_in}\ ;\ L_{h\_out}=\sum_{j=1}^{m} L_{Hj\_out}$$

若哑设备为HUB 时，存在下列等式：

$$L_{in}=\sum_{i=1}^{n} L_{i\_out}+L_{h\_out}$$

$$\forall l \quad 1\le l\le N \qquad L_{l\_in}=\sum_{i=1}^{n} L_{i\_out}+L_{out}+L_{h\_out} \quad i\neq l$$

其中L_{h_out}为未知量，可通过消去未知量L_{h_out}后，判定等式成立来确定该哑设备为HUB，否则为哑交换机。

4.3 算法的提出及理论支撑

4.3.1 基本定义

在描述算法之前，本节将就提出算法的理论基础进行介绍和分析。首先，我们先来介绍算法有关的一些概念的定义及符号的含义(表4-1)。第三章的定义3.1至定义3.6仍采用。

表 4-1 算法有关的符合及含义

符号	含义
a_i	节点 a 的第 i 个端口
(a_i,b_j)	节点 a 和 b 直接的可能连接
$AFT(a_i)$	节点 a 的第 i 个端口的地址转发表的信息
$\overline{AFT}(a_i)$	$AFT(a_i)$的补集
$Q(a_i)$	(V,E)中可到达端口 a_i 的路径上的所有节点的集合
$\overline{Q}(a_i)$	$Q(a_i)$的补集
N=(V,E)	以太网拓扑图（V 为网络中所有节点）

定义 4.1 哑设备：不可网管的设备统称为哑设备，可能是 HUB、没有管理地址的交换机、SNMP 共同体字符串不匹配的 IP 对象。

定义 4.2 可能连接：两节点之间可能存在的潜在连接。端口 a_i 和 b_j 之间存在可能连接当且仅当$\overline{Q}(a_i)\cap\overline{Q}(b_j)=\varnothing$，表示为$(a_i,b_j)$。

定义 4.3 终端节点 当且仅当节点的端口数为 1，且与之相连的不是 hub 的节点称为终端节点。例：图 4.1 中的节点 2。

定义 4.4 直接连接 交换机之间的物理连接。

定义 4.5 间接连接 两台交换机之间经过了 1 台或多台交换机实现的逻辑连接。

以下是算法引理涉及的概念的介绍：

1. 集合 AFT 与 Q 的区别：

AFT 与 Q 的区别应分两种情况：单子网和多子网。

对于单子网：$Q(a_i)=AFT(a_i)$；

对于多子网：$Q(a_i)$是 $AFT(a_i)$的扩展集，$Q(a_i)$包含 $AFT(a_i)$，同时，定义 $Q(a_i)$的补集$\overline{Q}(a_i)=V-Q(a_i)$。

我们将通过图 4-1 和表 4-2、4-3 解释集合 Q、$\overline{Q}$、AFT 和 $\overline{AFT}$ 的区别：

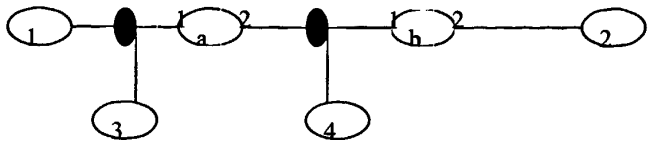


图 4-1 引理证明图 1

图中实心节点表示哑设备，其中子网划分为：$s_1=\{1,2\}, s_2=\{3,4\}$，a、b 各为一个子网。表 4-2 表示 AFT 和 $\overline{AFT}$，表 4-3 表示 Q 和 $\overline{Q}$。

表 4-2 AFT 表

Port	AFT	$\overline{AFT}$
a_1	1,3	2,4,a,
a_2	2,4	1,3,a
b_1	1	2,b
b_2	2	1,b

表 4-3 Q 表

Port	Q	$\overline{Q}$
a_1	1,3	2,4,a,b
a_2	2,4,b	1,3,x
b_1	1,3,4,a	2,b
b_2	2	1,3,4,a,b

同时，由图 4-1 可以看出，节点 2 是个终端节点，它只有单一的端口，所以，$(b_2,2)$为直接连接。节点 a 和 b 之间的有可能连接为(a_1, b_2)和(a_2, b_1)，因此，(a_2, b_1)唯一的可能连接。

2. 若两端口满足以下条件，将存在可能的直接连接，：

1) $AFT(a_i)\cap AFT(b_j)=\varnothing$;

2) $AFT(a_i)\cup AFT(b_j)=U$; U 为子网 S 中路由器和交换机的 MAC 地址集合；

3) $\overline{AFT}(a_i) \cap \overline{AFT}(b_j) = \varnothing$；

4) 如果 a_i，b_j 居于同一子网，则不存在与 a_i 来自不同子网的节点 c_k 使 $AFT(a_i) \cup AFT(b_j) = AFT(a_i) \cup AFT(c_k)$，并且使 $AFT(a_i)$与 $AFT(c_k)$满足前三个条件成立。

3. 介绍关于结合 Q 的两规则：

规则 4.1：如果端口 a_i 与 b_j 相连，其中 $a \neq b$，则 $Q(a_i)=Q(a_i) \cup \overline{Q}(b_j)$及 $Q(b_j)=Q(b_j) \cup \overline{Q}(a_i)$；

规则 4.2：如果端口 a_i 与 b_j 相连，其中 $a \neq b$，节点 $c \in Su$，$c \in Q(a_k)$ $(k \neq i)$，且对于任意端口 b_l，$Q(b_l)$不包含任何来自 Su 的节点，则 $Q(b_j)=Q(b_j) \cup C$，C 包含来自子网 Su 的所有节点和所有 Su 节点发现的节点。

4.3.2 引理证明

本节将给出产生 4.4 小节算法的引理及引理证明：

引理 4.1：i 和 j 分别为节点 a 和 b 的任意端口，如果 $AFT(a_i) \cap AFT(b_j) \neq \varnothing$，则节点 a 和 b 之间最多存在两个可能连接。

证明：假设存在一个节点 u 满足 $u \in AFT(a_i) \cap AFT(b_j)$ 和 $u \in$ 子网 s。网络中节点 v 使 $v \in$ 子网 s、$v \in AFT(a_k)$且 $u \in AFT(b_l)$，其中 $i \neq k$，$j \neq l$。我们考虑以下两种情况：

1.至少通过节点 a 或 b 的三个端口，可以发现子网 s 中的所有节点。不失一般性，假设网络中端口 a_i 和 b_j 之间有一条如图 4-2 的路径。假设子网 s 中存在 u、v 和 w 三个节点使得 $u \in AFT(a_i)$、$v \in AFT(a_k)$、$w \in AFT(a_m)$。因为这是生成树的一条路径，所以 v 和 m 必属于 $AFT(b_j)$。同时，由于 u 属于子网 s，它必然出现在节点 a 和 b 的 AFTs 中。假设 $u \in AFT(b_l)$，$u \in AFT(a_i)$，那么，$\{v,w\} \in \overline{AFT}(a_i) \cap \overline{AFT}(b_l)$，$u \in \overline{AFT}(a_p) \cap \overline{AFT}(b_q)$，其中 $p \neq i$，于是，$\{u,v,w\} \in \overline{AFT}(a_p) \cap \overline{AFT}(b_q)$，$p \neq i \neq k \neq m$，$j \neq l \neq p$。因此，$(a_k, b_j)$是节点 a 和 b 之间的唯一可能连接。

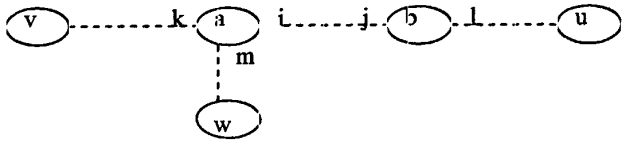


图 4-2 引理证明图 2

2.最多通过节点 u 和 v 的两个端口，可以发现子网 s 中的所有节点。假设子网 s 中的所有节点包含在 a 和 b 的两个 AFTs 中，节点 a 的 AFT(a_i)、AFT(a_k)和节点 b 的 AFT(b_j)、AFT(b_l)。如图 4-3，$u \in AFT(a_i) \cap AFT(b_j)$、$v \in AFT(a_k) \cap AFT(b_l)$。因为 $u \in \overline{AFT}(a_k) \cap \overline{AFT}(b_q)$，$p \neq l$ 和 $v \in \overline{AFT}(b_j) \cap \overline{AFT}(a_q)$，$q \neq k$，于是节点 a 和 b 之间只有($a_k$, b_j)和(a_i, b_l)两个可能连接。

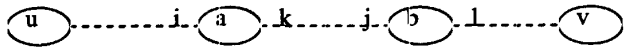


图 4-3 引理证明图 3

引理 4.2：i 和 j 分别为节点 a 和 b 的任意端口，如果 $Q(a_i) \cap Q(b_j) \neq \varnothing$，但是对于节点 a 和 b 的任意端口 p 和 q，a、b 两节点的初始 AFT 不相交，则(a_i，b_j)是节点 a 和 b 唯一的一个可能连接。

证明：假设存在子网 s_1 使得节点 $u \in$ 子网 s_1 且 $u \in Q(a_i) \cap Q(b_j)$。若 s_1 可以在节点 a 和 b 的一个端口中被发现，那么 s_1 一定被加入 $Q(a_i)$和 $Q(b_j)$。存在节点 c 使得网络中有两条路径(c_p, a_i)、(c_q, b_j)，且节点 c 要么属于子网 s_1 要么可以至少两个发现来自子网 s_1 的端口。此外，$\{u,v\} \in$ 子网 s_1，$\{w,z\} \in$ 子网 s_2，并且在节点 a 和 c 的 AFTs 中可以发现来自 s_1 的节点，在节点 b 和 c 的 AFTs 中可以发现来自 s_2 的节点。

不适一般性，若节点 a 在节点 b 和 c 之间的路径上，且节点 u 和 v 出现在节点 a 的两个端口的 AFT 上，如图 4-4。因此，这将与我们假设的 a、b 两节点的初始 AFT 不相交矛盾。

若节点 c 在节点 a 和 b 之间的路径上，我们考虑以下两种情况：

1. 节点 c 分别通过端口 c_p 与 c_q 与节点 a 和 b 相连，$p \neq q$，如图 4-5；

2. 节点 c 通过端口 c_p 与节点 a 和 b 相连，如图 4-6。

对于第一种情况：当可能连接(a_i, c_p)的选择和扩展程序的应用，通过规则 4.1

可以得到 $Q(a_i)$包括所有来自子网 s_1、s_2 的节点及他们所能发现的节点。因此，所有来自子网 s_1 的节点以及节点 u、v、b 被加入 $Q(a_i)$；同样的，可能连接(a_i, c_p)的选择和扩展程序的应用，所有来自子网 s_2 的节点以及节点 w、z、a 被加入 $Q(b_j)$。因此，节点 a 和 b 之间存在唯一的可能连接。

对于第二种情况：同理可证。

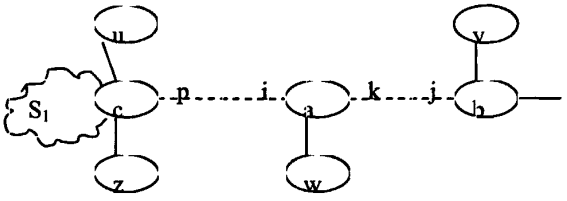


图 4-4 引理证明图 4

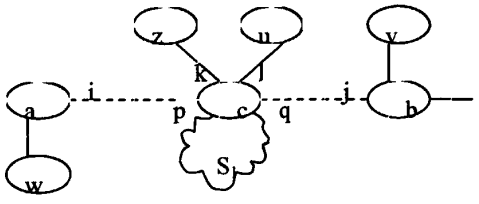


图 4-5 引理证明图 5

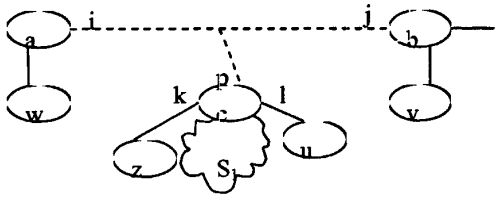


图 4-6 引理证明图 6

引理 4.3：如果节点 a 和 b 之间没有唯一的可能连接(a≠b)，则不存在节点 u 和 v 使$\{u,v\}\in AFT(a_i)$，$u\in AFT(b_j)$以及 $v\in AFT(b_k)$，其中 $j\neq k$ 成立。

证明：假设网络中存在节点 a、b 及节点 u、v 使得$\{u,v\}\in AFT(a_i)$，$u\in AFT(b_j)$，$v\in AFT(b_k)$，其中 $j\neq k$，如图 4-7(实心椭圆为哑设备)，节点 w 与 u、v 属于同一子网且 $w\in AFT(b_k)$。

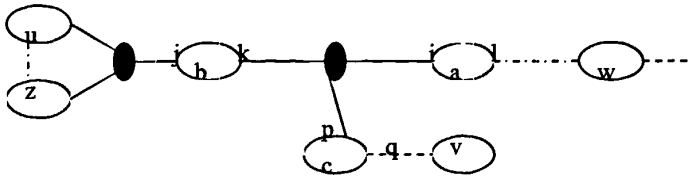


图 4-7 引理证明图 7

因而，得出一下结论：$w\in \overline{AFT}(a_i)\cap \overline{AFT}(b_y)$，$y\neq k$，$u\in \overline{AFT}(a_x)\cap \overline{AFT}(b_k)$，$x\neq i$，以及 $v\in \overline{AFT}(a_x)\cap \overline{AFT}(b_y)$，$x\neq i$，$y\neq k$。因此，$(a_i,b_k)$是唯一连接。

引理 4.4：如果网络中的任两个节点之间不存在唯一的可能连接，且节点 a 和 b 之间最多存在两个可能连接，则选其中的一个可能连接产生的有效网络拓扑将与给定的 AFT 兼容。

证明：如图 4-8，假设端口 a_1 不直接与 $AFT(a_1)$中的节点相连。我们证明连接端口 a_1 到 $AFT(a_1)$中的节点将产生有效的拓扑。

通过引理 4.3，可得：要么 $AFT(b_i)\cap AFT(c_j)=\varnothing$，其中 i 和 j 分别为 b 和 c 的任一端口；要么 $AFT(a_k)\cap AFT(c_j)=\varnothing$，其中 k 和 j 分别为 a 和 c 的任一端口。因而，我们考虑以下两种情况：

1. $AFT(b_i)\cap AFT(c_j)=\varnothing$，其中 i 和 j 分别为 b 和 c 的任一端口。既然如此，$AFT(a_1)\subset AFT(b_1)$。令 $AFT(b_1)\cap AFT(a_1)=F$，由引理 4.1，可得节点 a 和 b 之间最多存在两个可能连接，因此，连接端口 a_1 到 F 中的节点将在端口 a_2 和 b_1 之间产生一个唯一连接。通过扩充节点 a 和 b 的 AFTs，节点 c、z、v 以及所有可以从这些节点发现的节点将被增加到 $AFT(a_2)$。最后，如果端口 a_2 与 d_1 相连，那么(b_2, d_1)是 b_2 和 d_1 路径上的有效可能连接。如图 4-8。

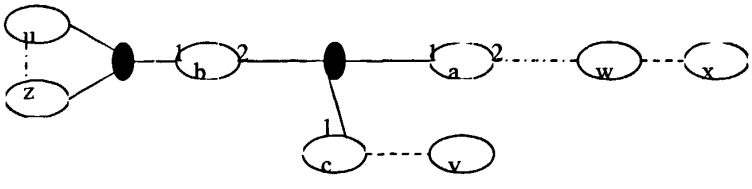


图 4-8 引理证明图 8

2. $AFT(a_k) \cap AFT(c_j) = \varnothing$，其中 k 和 j 分别为 a 和 c 的任一端口，同理可得。如图 4-9。

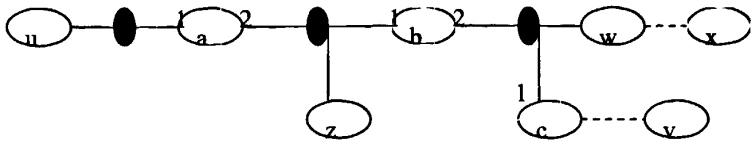


图 4-9 引理证明图 9

4.4 算法描述

根据上述定义、定理及规则，处理含哑设备的多子网算法描述如下：

首先，获取各交换机的 AFT，初始化各集合。令节点间可能连接的集合为 X，可能的直接连接的集合为 Y，算法最终输出的表示节点之间连接情况的集合为 Z，连接的唯一性表示为 U(1 表示唯一，0 表示不唯一)，集合 W 表示单一节点的集合。

令 U=1，算法分两个部分：

1) 收集可能的直接连接 Y 及集合 W 的数据，令 M=∅；

当集合 Y≠∅时，执行:

1 若 a_i 为 Y 的一个终端节点，且$(a_i,b_j) \in Y$，则选择直接连接(a_i,b_j)；

(1) $Z=Z \cup \{(AFT(a_i), AFT(b_j))\}$；

(2) 若 $a \in W_l$、$b \in W_k$ 且 $k \neq l$，则若 $W_l = W_l \cup W_k$，将 W_k 从 W 中删除；

(3) 删除 Y 中含有 W_l 中节点的连接；

2 若 Y 中没有终端节点，令 U=0；

(4)在 Y 中任选一连接(a_i,b_j)；

(5)转至(1)；

2) 获取可能连接集合 X 的数据：对于任意 a_i 和 b_j，当 $\overline{Q}(a_i) \cap \overline{Q}(b_j) = \varnothing$时，将$(a_i,b_j)$添加至集合 X。

当 X≠∅时，执行:

1 找出节点 a 和 b 之间的最小连接数；　　//引理 4.1；

2 若(a_i,b_j)不是唯一的可能连接，令 U=0，$Z=Z \cup \{Q(a_i),Q(b_j)\}$；

3 删除(a_i,b_j)，扩展 Q(a_i)和 Q(b_j)；　　//规则 4.1、4.2；

4 更新集合 X；

返回集合 Z 和 U 的值。

算法流程图如图 4-10 所示。

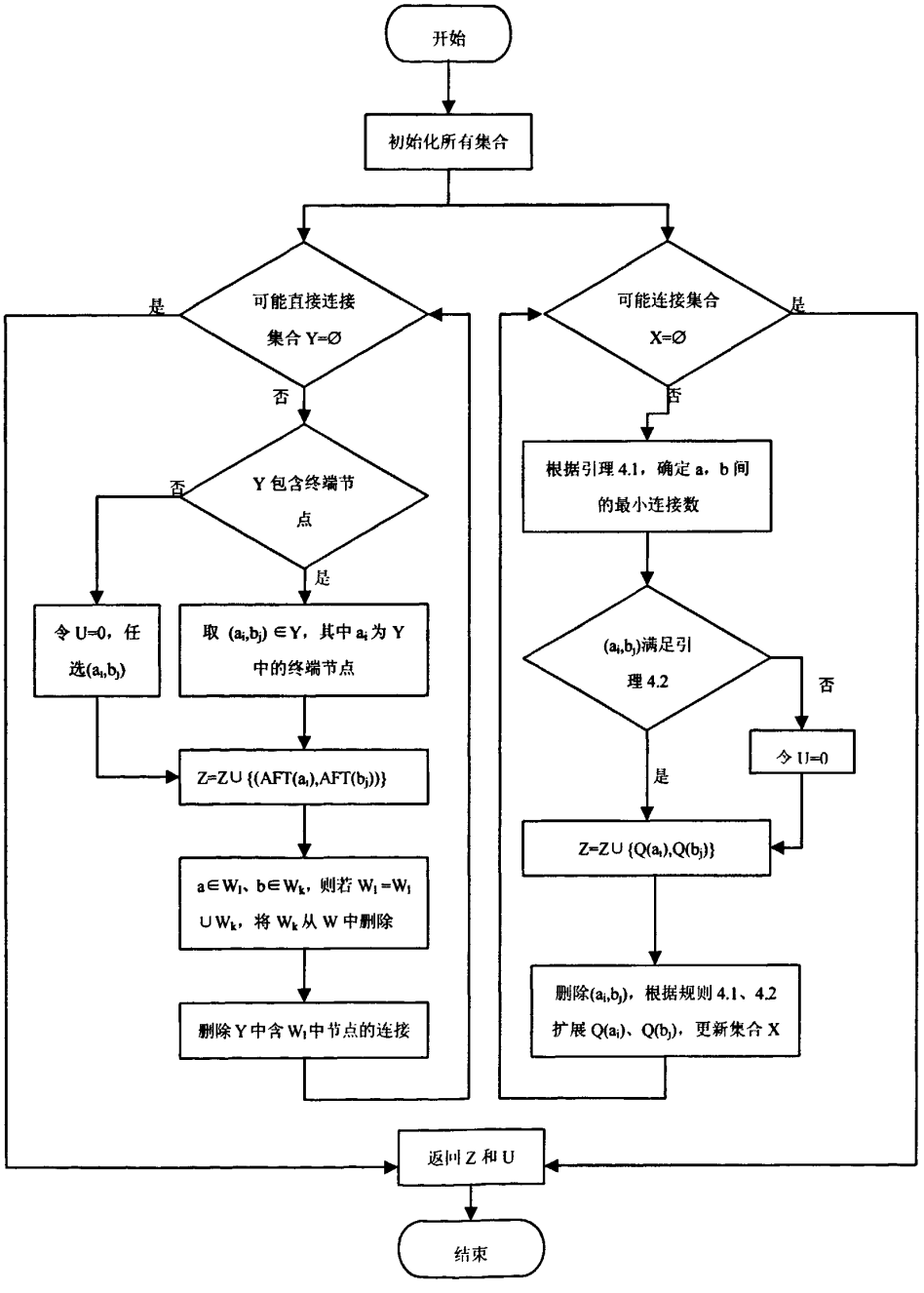


图 4-10 算法流程图

4.5 本章小结

本章对改进的新算法的提出进行了可行性分析，并提供了支持的算法理论基础，包括一些基本的定义和支持算法的原理及证明。在此基础上，对算法的主要部分进行描述，并给出整个物理拓扑发现的流程图。

5 拓扑发现具体实现

5.1 系统体系结构及具体模块实现

实现拓扑发现所需完成的工作是：首先，获得拓扑发现所需的信息，本文主要指 AFT 信息，并储存在数据库中；其次，处理、分析数据库中的数据，判断网络拓扑连接；最后，将网络拓扑图显示出来。因此，系统体系机构可分为以下三个模块，如图 5-1：

数据收集模块：获得拓扑发现所需的信息，并储存在数据库中。主要用 SNMP 方式获取所需数据信息，并储存在数据库中。

拓扑发现模块：处理、分析数据库中的数据，判断网络拓扑连接。本文根据 4.4 节的拓扑发现算法对数据进行操作，进而得出网络设备之间的连接情况，即网络拓扑结构。

拓扑显示模块：利用所得的物理拓扑结构将网络拓扑图显示出来。

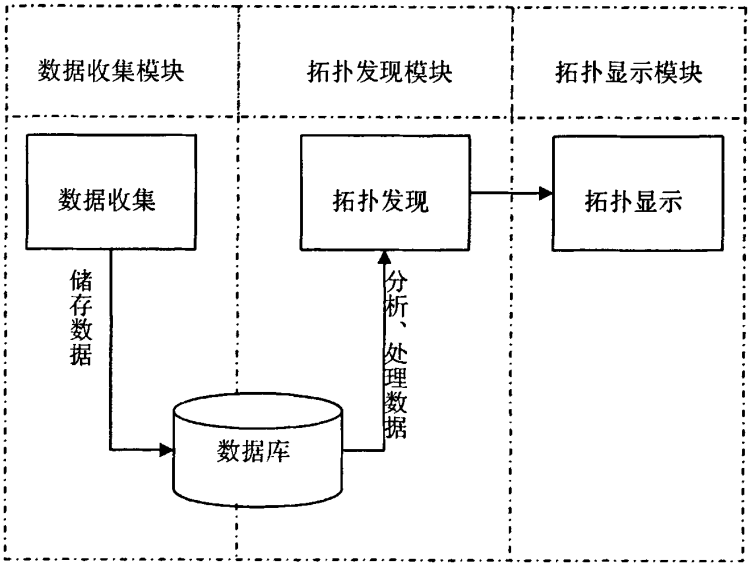


图 5-1 系统体系结构

5.1.1 数据收集模块

作为拓扑发现系统体系结构中最基础的模块，数据收集模块应实现以下几个功能，流程图如图 5-2 所示。

1. 判断设备是否为激活状态，确保 AFT 的完整性；

判断设备是否为激活状态，我们采用的方法是向子网中的所有设备按子网地址一次加 1 的方式发送 ICMP 包，网管站向子网内各设备发送 ICMP 回应请求包。当设备发回应应答时，则设备存在且处于激活状态；若无回应，则无该设备，为避免因其他原因造成回应请求及回应包丢失，还要重复再发 ICMP。

对于 AFT 的完整性的问题的解决，我们在 4.2.1 小节中已经提到。

2. 判断设备是否支持 SNMP；

本文涉及的网络设备包括路由器、交换器和哑设备（HUB 或哑交换机），不包括主机。

对于处于存活状态的设备，向其 161 号端口发送 SNMP 报文，如果有回应，则表示该设备运行了 SNMP 代理。

(1) 支持 SNMP，则取 MIB，判断设备类型，并访问端口及其 AFT；

对支持 SNMP 的设备，取其 MIB 值，通过访问 MIB 库中的 sysServices 值和 ipForwarding 值来判断设备类型。

对网络设备发送 GetRequest 请求，请求的对象是 sysServices，对应的 oid 为 1.3.6.1.2.1.1.7.0。设 L 是 OSI7 层模型中的某一层，如果设备在第 i 层提供了服务，则 Li 被赋予相应的层数，部分对应关系如表 5-1 所示，公式如下：

$$\text{sysServices}=\sum_{i=1}^{Li}2^{(Li-1)}$$

表 5-1 服务/设备类型对照表

网络层次	服务类型值	设备类型
2	2	交换机
3	4	路由器
4	8	传输层设备
7	64	应用层设备

由于交换机工作为链路层服务，则 $\text{sysServices}=2^{(2-1)}$，可据此判断交换机；路由器提供链路层和网络层服务，则它的 $\text{sysServices}=2^{(3-1)}+2^{(2-1)}$，则对应的层次值 Li=3，因此判断该设备为路由器。这种判断方法对一物理网络中的常规路由器是有效的，但是，在一些网络中，路由器的 sysServices 值有些为 6，有些为 7，有些为 76，有些为 78。所以仅根据 sysServices 值来判断设备是否为路由器条件

不够充分。

ipForwarding 的值来进行判断。ipForwarding 指出本设备是否具有报文转发功能，当 ipForwarding=1 时，具有转发功能；ipForwarding=2 时，不具有转发功能。因此当该值为 1 时，判定该设备为路由器。

综上所述，仅根据 ipForwarding，或仅根据 sysServices 并不能判断一个设备是否为路由器.但是，如果通过 ipForwarding 和 sysServices 的值，就可以准确的判断出一个设备是否为路由器。例如，若某个设备的 sysServices 值为 76，根据 $2^{(7-1)}+2^{(4-1)}+2^{(3-1)}=76$ 可知道该设备提供了网络层服务;如果这个设备的 ipForwarding=1，则该设备具有报文转发功能，综合二者，可以准确判断该设备为路由器。

(2) 不支持 SNMP，判断哑设备类型，并采用端口流量分析读取设备的信息；

对于不支持 SNMP 的设备的处理及哑设备类型的判断在 4.2.2 和 4.2.3 小节中已经介绍过。

3. 访问并获取所需数据信息，并储存在数据库中。

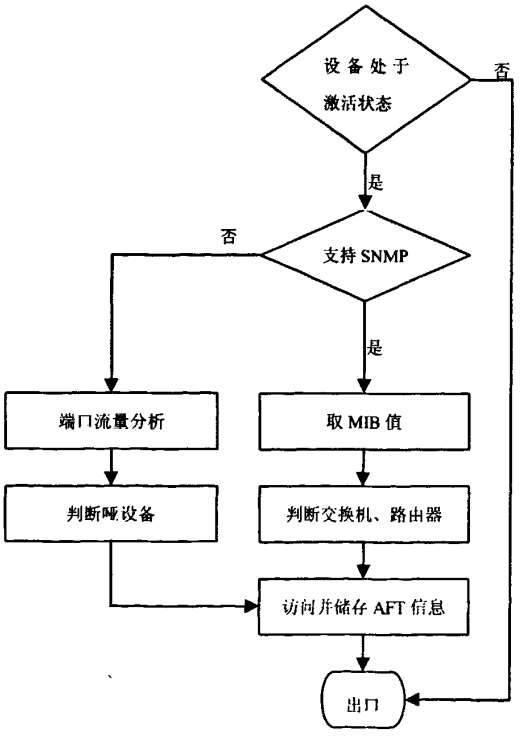


图 5-2 数据收集模块流程图

5.1.2 拓扑发现模块

拓扑发现模块是物理拓扑发现系统体系结构中最重要的一部分，其需要实现的功能如图 5-3。其中，核心的一环是根据 4.4 小节的算法来处理上一模块中获得的 AFT 信息，算法处理的具体过程在第四章已详细介绍。

对所收集的 AFT 信息，根据引理 4.1、引理 4.2、引理 4.3 和引理 4.4 及规则 4.1 和规则 4.2 划分端口集合，并对端口进行初始化，包括可能连接、直接连接及终端节点等的判断。最终根据 4.4 小节的算法确定设备端口的连接情况，也就获得网络的物理拓扑结构。

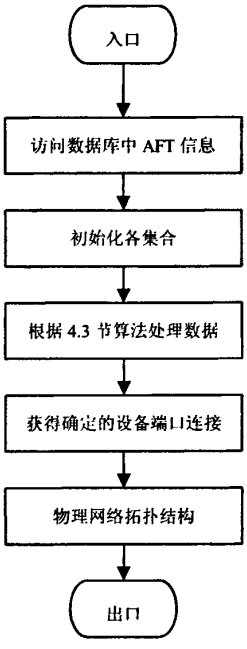


图 5-3 拓扑发现模块流程图

5.1.3 系统使用的数据结构

本系统选用了 SNMP++开发工具包进行开发，主要的数据结构的描述如下：

```
typedef struct BridgeNode{
    CString pSubnetIP;
    CString pSubnetMask;
    …
    CString defSpeed;
```

```
}BridgeNode;                    //交换机节点
typedef struct RouterNode{
CString interfaceNum;
CString pNetNum;
…
CString pTo;
}RouterNode;                    //路由器节点
Struct PotenUnion{
CString SwitchIp;               //当前被测交换机 IP 地址
CString SwitchMac;              //当前被测交换机 MAC 地址
unsigned int Port;              //当前被测交换机的端口;
CString nextSwitchIp;           //下一台交换机的 IP 地址;
CString nextMac;                //下一台交换机的 MAC 地址
unsigned int nextPort;          //下一台交换机的端口;
addressTable unionAft;
} PotenUnionQue
```

5.2 实验设计及实现结果

5.2.1 实验设计

本实验的设备包括 Cisco 2600 路由器和若干华为交换机，简单得实现了网络拓扑的异构。其中，运行拓扑发现体统的主机为 XP 操作系统，并装有 VC++6.0、SNMP++、Microsoft Access 2000 和 ethereal 等软件。

为方便验证本算法得出实验结果及与其他算法的对比，本实验的实验环境是以文献[40]的图 10(a)来搭建的，如图 5-4。其中，{s,t}属于子网 s1，{x,y}属于子网 s2，{u,v}属于子网 s3，{r,q}属于子网 s4，a、b、c、d、e 各自为一个子网，实心的椭圆表示哑设备（HUB 或是哑交换机）。

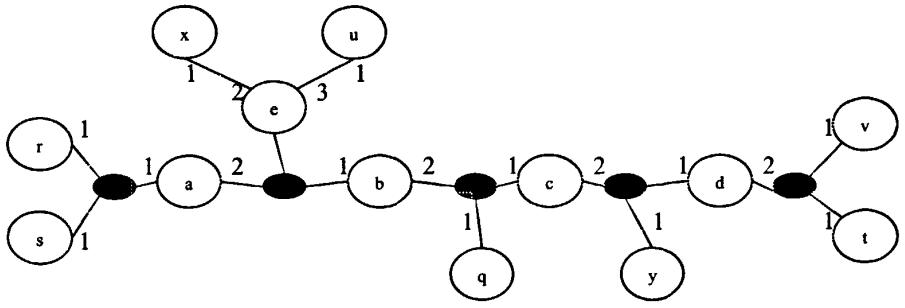


图 5-4 实验拓扑图

5.2.2 实验设备配置

拓扑图中的路由器和交换机完成基本配置外应启动 SNMP 协议和 STP 协议。

- 配置 SNMP 协议：

#configure terminal

#snmp-server community rw/ro // rw 为配置读写通信字符；ro 为配置只读通信字符串(例如: snmp-server community public ro)

#end

#copy running-configstartup-config

- 子网划分

由图 5-4 可知，本实验须划分出 9 个子网。将一 C 类网络划分成 9 个子网，因为 $2^3<9<2^4$，因此，将网络划分成 16 个子网。若我们取该 C 类地址的网络号为 192.168.1，则就可确定子网掩码为：255.255.255.240，且该 C 类网络的 IP 地址为 192.168.1.1~192.168.1.254（因为全“0”和全“1”的地址有特殊含义，不作为有效的 IP 地址）。

各子网的 IP 地址范围分别为：

(1)192.168.1.1~192.168.1.14；(2)192.168.1.17~192.168.1.30；

(3)192.168.1.33~192.168.1.46；(4)192.168.1.49~192.168.1.62；

(5)192.168.1.65~192.168.1.78；(6)192.168.1.81~192.168.1.94；

(7)192.168.1.97~192.168.1.110；(8)192.168.1.113~192.168.1.126；

(9)192.168.1.129~192.168.1.142；(10)192.168.1.145~192.168.158；

(11)192.168.1.161~192.168.1.174；(12)192.168.177~192.168.1.190；

(13)192.168.1.193~192.168.1.206；(14)192.168.1.209~192.168.1.222；

(15)192.168.1.225~192.168.1.238；(16)192.168.1.241~192.168.1.254。

取以上的 9 个子网中的 IP 对图 5-5 中设备的 IP 地址配置进行配置。

5.2.3 实验结果

由 4.2 小节所提及的方法，向网络中所有的节点发送 ICMP 回应请求包，当相应设备发回应应答时，可确定设备存在且处于激活状态；若无回应，为确定是否不存在该设备，则再发第二次、第三次以避免其他原因造成回应请求及回应包丢失的情况。

根据改进的算法以及对应的数据结构，在 VC++6.0 下编译并运行程序，Ping IP 为 192.168.1.19 的交换机，可得如图 5-5：

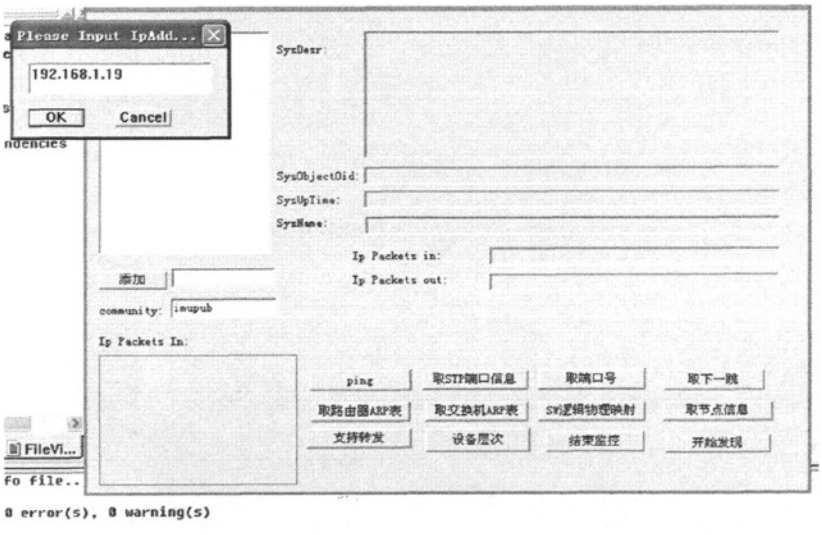


图 5-5 实验结果图 1

当节点地址可到达，如图 5-6，便可确定设备存在且处于激活状态；

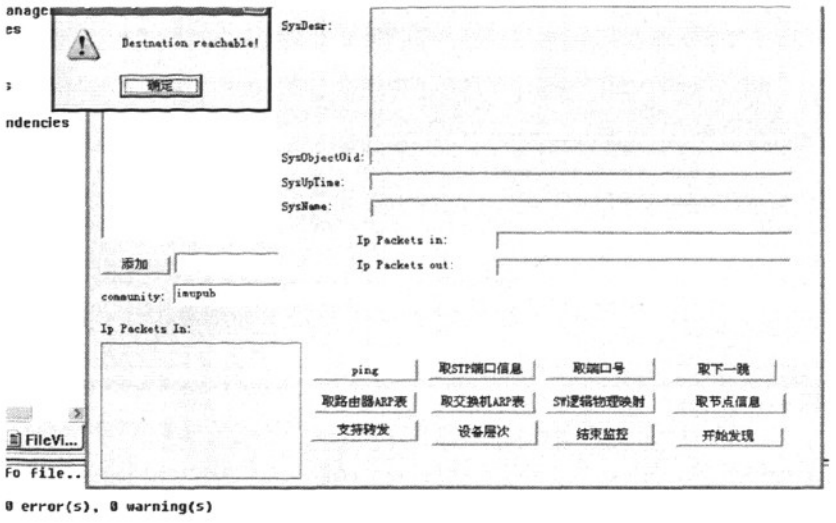


图 5-6 实验结果图 2

因此，我们可以根据程序取得拓扑发现所需要的数据，如节点信息、端口号，下一跳信息、STP 端口信息等，如图 5-7。同时，我们也可以得到端口进流量和出流量的数据，为本文所提出的对哑设备的处理提供了保障。

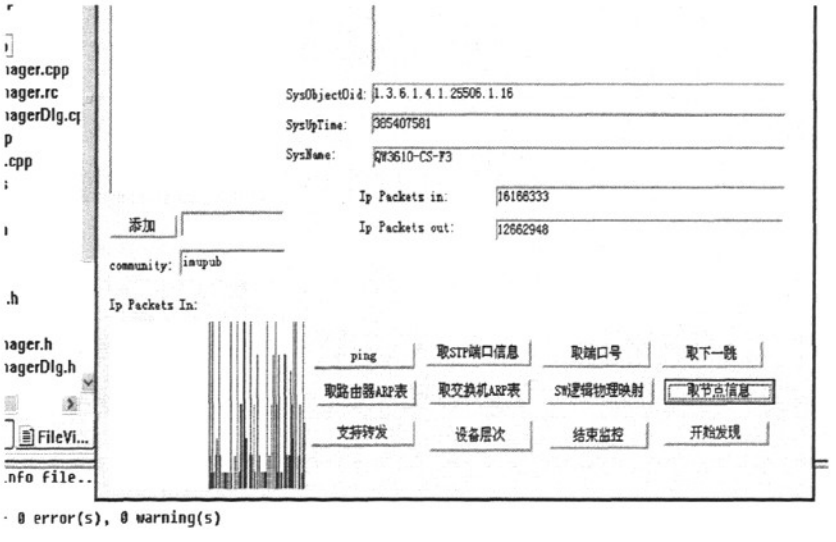


图 5-7 实验结果图 3

对不支持 SNMP 的设备，我们可以利用 Ethereal 对端口的数据包进行侦听，对交换机各端口的数据包进行分析，并把取出的数据存放的 Access 数据库中再进行处理。Ethereal 获取数据包信息如图 5-8 所示，该图是通过 Ethereal 获取 ICMP 信息。

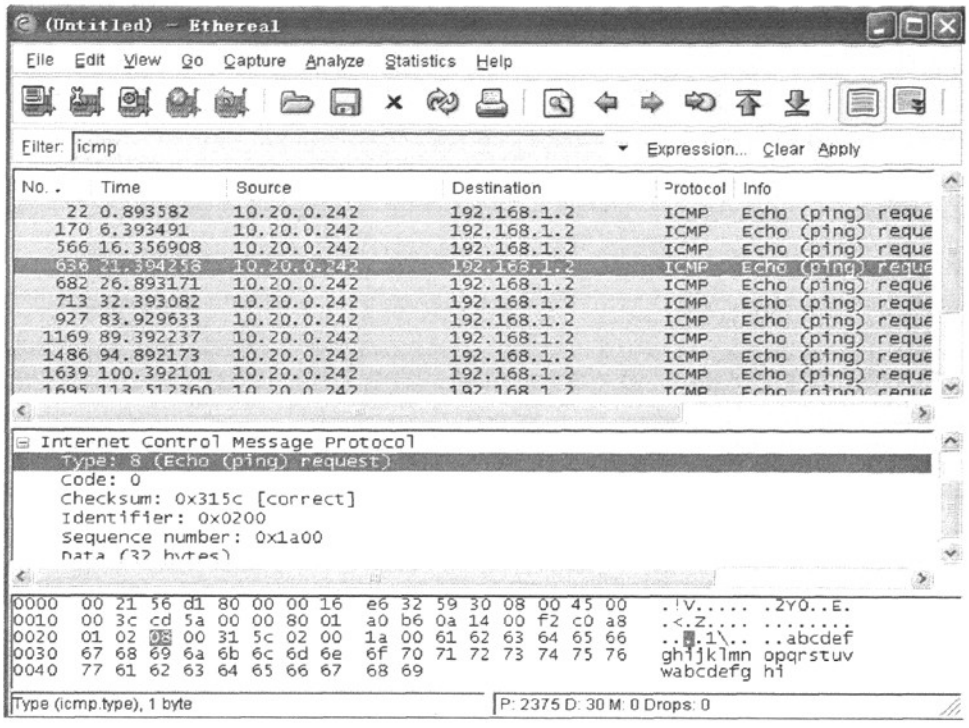


图 5-8 Ethereal 获取数据包信息

完成数据收集之后，我们将储存在 Access 数据库中的各节点端口的地址转发表信息转化成直观的数据，如表 5-2 所示：

表 5-2 各端口的 AFTs

端口	AFT	端口	AFT
a_1	r,s	d_1	s,u
a_2	q,t	d_2	t,v
b_1	r,s,u,x	e_1	v,y
b_2	q,t,v,y	e_2	x
c_1	s,u,x	e_3	u
c_2	t,v,y		

本实验网络中的可能连接集合 X 和可能直接连接集合 Y 如表 5-3 所示：

表 5-3 连接集合表

可能连接集合 X	可能直接连接集合 Y
(a_1,b_1) (a_2,b_1) (a_1,c_2) (a_2,c_1) (a_1,d_2) (a_2,d_1)	(e_2,x)
(a_1,e_1) (a_2,e_1) (b_1,c_2) (b_2,c_1) (b_1,d_2) (b_2,d_1)	(e_3,u)
(b_1,e_1) (c_1,d_2) (c_2,d_1) (c_1,e_1) (d_1,e_1)	

根据规则 4.1、规则 4.2 及定理 4.1~4.4，执行算法，将得到网络中可到达各端口的路径上的所有节点的集合（如表 5-4 所示）以及各子网设备的连接关系，即可得网络拓扑关系。

表 5-4 拓扑关系表

端口	集合 Q	端口	集合 Q
a_1	r,s	d_1	a,b,c,e,q,r,s,u,x,y
a_2	b,c,e,q,t,u,v,x,y	d_2	v,t
b_1	e,s,r,u,x	e_1	a,b,c,d,q,r,s,t,v,y
b_2	d,t,q,v,y	e_2	x
c_1	e,s,u,x	e_3	u
c_2	d,t,v,y		

所得的拓扑结构与文献[40]算法所得的设备连接结果（如图 5-9）一致，但是本文拓扑发现方法的时间复杂度为 $O(n^3)$，优于文献[40]所提出的算法。

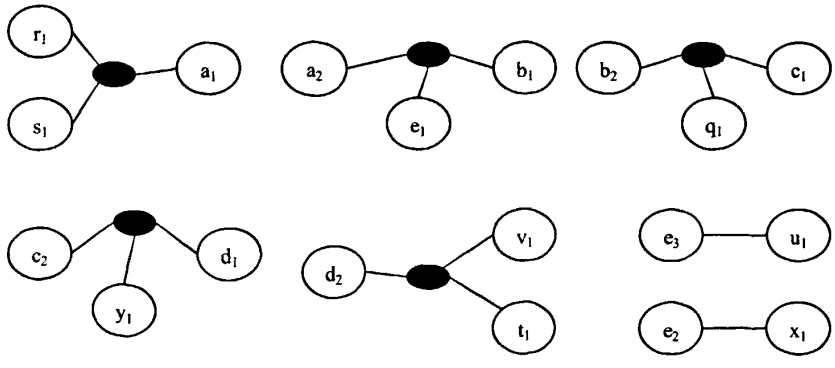


图 5-9 连接结果图

5.3 本章小结

本章介绍了所提出的系统体系架构，并提出算法实现的各具体模块，包括数据收集模块、拓扑发现模块等。在具体实现方面，根据搭建的实验原型系统，验证了所提出的改进的算法的有效性。

6 总结

本章对全文的内容进行了总结，并对进一步工作提出了建议和进行了展望。

6.1 总结

网络技术的迅速发展，网络规模不断扩大，网络结构越来越复杂，网络功能越来越强大，网络管理技术已经成为一个日益重要的课题。自动发现物理拓扑信息对提高现代 IP 网络管理能力起着至关重要的作用。发现网络拓扑结构的主要目的是利用它研究性能瓶颈，为网络测量工具和网络服务器找到最佳位置以及为网络管理提供更好的基础。但是，如今多数的网络拓扑发现工作把重点放在自动发现广域网拓扑结构上（WAN），尤其是路由器之间连接关系上，而忽略了二层拓扑结构发现。国内外学者一直在这方面做大量的研究和实验，并提出了许多实现方案，但是各种方案总有一定的缺点和局限，因此，拓扑发现领域的研究工作将会继续进行下去。

经过近一年的研究，在多子网物理拓扑的研究方面有了明显的进展，并有一定的突破与创新。本文除了对多子网物理拓扑发现算法进行分析和研究并提出了一种改进的算法之外，还设计并实现了实验原型系统，验证了算法的有效性。本文的主要工作如下：

1. 阐述了论文的选题背景及研究意义，分析了网络拓扑发现的重要性及当前国内外拓扑发现技术的研究现状；
2. 介绍了相关的网络拓扑发现工具及协议，包括：SNMP、ICMP、Ping 程序、Traceroute 程序、DNS 及 ARP 等其他拓扑发现工具及协议；
3. 对当前物理网络拓扑发现算法分析，详细介绍了单子网和多子网交换域，并且介绍了基于 SNMP、ICMP、AFT 及 STP 等物理网络拓扑算法；
4. 介绍改进的多子网物理拓扑发现算法，其中内容包括算法的可行性分析、算法提出及实现所面临的问题的解决、算法的具体描述以及流程图；
5. 介绍系统体系结构及具体模块的实现，并完成实现实验环境，验证算法的有效性。

本文的主要贡献和创新点主要体现如下：

1. 研究和分析了当前网络拓扑发现的相关工具与协议，并比较其优缺点，

为提出改进算法打下基础；

2. 分析现有算法的优缺点，融合了 AFT、STP 等网络拓扑发现协议的优点，提出了改进的基于 SNMP 的多子网物理拓扑发现算法；
3. 新算法支持异构，能对包含哑设备(hub 等)的多子网网络进行拓扑发现，提高了包含哑设备的异构多子网的效率，增强了算法的实用性；
4. 算法在 AFT 完整性的问题的解决以及不支持 SNMP 的设备的处理上进行了改进和创新；
5. 在实现实验设计结果的过程中，以 Cisco 路由器和若干华为交换机搭建原型系统，简单实现了网络的异构，并结合 VC++6.0、SNMP++、Microsoft Access 2000 和 ethereal 等软件完成了对算法有效性的验证。

6.2 展望

正如之前提到的，国内外学者一直在拓扑发现领域做大量的研究和实验，并提出了许多实现方案，但是各种方案总有一定的缺点和局限，因此，拓扑发现领域的研究工作将会继续进行下去。限于时间和精力，本文的算法还存在一些不足和局限性，有待于更进一步的改进和完善。

为使拓扑发现能更具有效性和适用性，今后将从以下几个方面展开研究：

1. 本文的算法都是在 IPv4 环境下实现的，没有考虑 IPv6 网络环境下的算法实现的适用性。随着拓扑发现研究工作的深入，这方面的研究也应该逐步的深入；
2. 使拓扑发现算法能发现更加复杂的网络拓扑结构，例如增加对包含 VLAN 的网络拓扑发现的研究；
3. 应能够在拓扑发现的拓扑显示模块上更加的提高，能利用好相应的仿真工具；
4. 进一步研究无线网络拓扑发现的算法；
5. 将拓扑发现与网络管理的其他功能相结合，例如性能管理、安全管理、故障管理等，使网络拓扑发现更有实用性和普遍性。

参考文献

[1]SNMP Prentice Hall PTR, (Second Edition), 1988

[2]刘琳琳.网络拓扑发现技术的研究和实现[D]. 西安电子科技大学硕士论文 2006.1

[3]J.D.Case, M.Fedor, M.Schofstall, J.Davin, RFC1157," A Simple Network Management Protocol (SNMP)," 1990.5

[4]K.McCloghrie, M.Rose, RFC1213,"Management Information Base for Network Management of TCP/IP-based internets:MIB-II,"1991.3

[5]HP's OpenView. http://www.openview.hp.com

[6]IBM's Tivoli. http://www.tivoli.com

[7]Dartmouth UniveQity. Intermapper: An intranet mapping and snmp monitoring program for the macintosh. http://www.dartware.com/Intermapper

[8]Aman Shaikh,Mukul Goyal,Albert Greenberg,et al. An OSPF Topology Server: Design and Evaluation. http://www.cis.ohio-state.edu/~mukul/jsac.pdf

[9] Yuri Breitbart, Minos Garofalakis, Cliff Martin Rajeev Rastogi, S.Seshadri, Avi Silberaschatz. Topology Discovery in Heterogeneous IP Networks: The NetInventory System [J].IEEE/ACM Transaction On Networking, June 2004 Vol.12, No3:401-413.

[10]Hwa-Chun Lin,Hsin_Liang Lai,Shou-Chuan Lai,"Automatic Link layer Topology Discovery of IP Networks" IEEE, 1999

[11]A.Bierman, K.Jones. Physical Topology MIB.RFC2922(available from http://www.ietf.org/rfc)，September 2000

[12]http://www.cnpaf.net/Class/wlxy/05511203013828071 35.htm

[13]刘燕等，简单网络管理协议 SNMP 的发展与研究[J]，计算机工程与设计，2001，22(3): 20-27

[14]李明江 编著，SNMP 简单网络管理协议，电子工业出版社，2007.5

[15]http://www.cnpaf.net/Class/SNMP/0532918532646753.html

[16]http://baike.baidu.com/view/534239.htm

[17]W.Richard Stevens 著，范建华 等译. TCP/IP 详解 卷 1：协议. 机械工业出版社，2000

[18]马林兵,吴屹东,胡小鹏.ICMP 协议在网络层的地位及应用[J]. 计算机应用研究,2000,07(20)

[19]http://baike.baidu.com/view/709.htm

[20]V.Jacobsen et al. Traceroute. Unix man pages. 1989

[21]http://baike.baidu.com/view/70478.htm

[22]路晓雷，曹伟东，张霞.DNS 技术在网络链路控制中的应用[J],2008,05(13)

[23]赵大伟,白玲.ARP 概念及攻击与防护的原理[J],2008,25(55)

[24]陆余良,张永,刘克胜,蔡铭. ARP 协议在局域网类型探测中的应用[J],计算机工程 , 2004,(1)

[25]李琳，李杰.基于 SNMP 的网络层拓扑发现.计算机工程与设计[J],2008,29(6)

[26] 殷卫红，耿新民. 基于 SNMP 协议的网络管理实现技术[J]. 微计算机信息 , 2006,(27)

[27]RFC1493-1993, Definitions of managed objects for bridges [S].

[28]R.Siamwalla,R.Shama,S.Keshav.Discovering internet topology.
(http://www.cs.comell.edu/skeshav/paper/discovery.pdf),1998-07

[29]蔡伟鸿,舒兆港,刘震. 基于 SNMP 协议的以太网拓扑自动发现算法研究[J].计算机应用与工程,2005.14

[30]曹晓梅，夏保胜，王能.基于 SNMP 和 ICMP 的拓扑自动发现算法的分析与实现[J].河南大学学报(自然科学版)，2003,33(1)

[31]李继良.基于 ICMP 的网络拓扑自动发现算法研究[J].现代计算机(专业版),2007,07(23)

[32]辛家斌.网络拓扑发现技术研究[D].硕士学位论文.2007.5.

[33]刘玉华,余胜生,周敬利,李艳红. 基于 AFT 的链路层自动拓扑发现算法[J].小型微型计算机系统.2004.12

[34]李玉鹏，王换招，赵青苹.基于 STP 的以太网物理拓扑发现[J]. 北京电子科技学院学报，2004,12(2)

[35]张占国,刘淑芬，包铁等.基于 STP 协议的物理网络拓扑发现算法[J],计算机工程,2008,06(38)

[36]石玫,李祥和.基于 STP 的物理拓扑发现算法研究[J].计算机工程与应用,2007,43(9)

[37]王双红，王文东，程时端等.多域的 IP 网络自动拓扑发现算法研究[J].北京邮电大学学报，2005,28(6)

[38]Bruce Lowekamp. Discovery and application of network information (PH.D.Thesis).School of Computer Science, Carnegie Mellon UniveQity, Pittsburgh, PA, USA, 2001

[39]郑海，张国清.物理网络拓扑发现算法的研究[J].计算机研究与发展，2002,39(3):264-268

[40] Y. Bejerano, Y. Breitbart, M. Garofalakis, R.Rastogi, Physical Topology Discovery for Large Multi-Subnet Networks Proceedings of INFOCOM 2003, 2003.

致 谢

本文是在导师贾波教授的悉心指导下完成的，在此表示最衷心的感谢。感谢她对我无微不至的关怀和谆谆教诲。贾老师以敏锐独到的眼光为我选定了这个能走在计算机信息发展前沿的课题，并为我提供了大量的相关资料，拓展了我的思路，为我的研究和开发工作奠定了基础。她高深的学术造诣、严谨的治学态度、高度的责任感、诲人不倦的高尚师德使我受益非浅，终身难忘。

感谢王伟明老师、董黎刚老师、陈添丁老师以及诸葛斌老师等浙江工商大学信电学院的老师，各为老师严谨的治学态度和渊博的知识是我享用不尽的财富。

同时，还要衷心感谢在毕业设计及论文写作过程中给过我鼎力帮助的孙中海、金少丹、杨尚大等同学。

特别感谢我的父母，对我研究生学习的支持、鼓励和资助，我的每一点成绩都离不开他们的关怀。

在此对所有关心、支持和帮助过我的人们一并致谢。

最后，向审阅本文的各为教授、专家表示致敬和衷心感谢！

攻读硕士期间发表的学术论文及参加的科研项目

发表的学术论文：

基于 IEEE 802.11 的 VoIP 的 QoS 性能的仿真与分析[J], 计算机科学 2008,35(10B):25-27.

参加的科研项目：

1. 分布式网络管理—自动拓扑发现和流量监控的设计及实现. 自然科学基金项目.（项目编号：200408020811）

2. 网络教学平台的建设与应用. 2006 年省新世纪高等教育教学改革项目.（项目编号：yb06035）

3. 浙江移动远程运维接口模块开发. 横向课题. 2008 年 10 月-现在