**密级： 保密期限：**



**硕士学位论文**



**题目：命名数据网络中基于显式标记的传输控制机制**

**学位类别： 专业学位**

**学 号：**

**姓 名：**

**专业领域： 电子与通信工程**

**导 师：**

**学 院：信息与通信工程学院**

**2017年 月 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：

命名数据网络中基于显示标记的高效传输机制

摘 要

随着时代的变化，互联网上应用也在不断的变化，从主机之间的资源共享逐渐变成内容的获取和分发。用户更加关注如何快速方便的获取内容，而不是在哪里获得这些内容。在这种新的应用模式下，当前互联网基于 TCP/IP 的体系结构逐渐表现出不适应性，暴露了许多问题(安全性差，不支持移动，不可靠，缺乏灵活性)。为了适应用户需求的变化，解决TCP/IP 体系结构存在的问题，许多研究人员开始研究新的体系结构。在新的网络体系结构当中，命名数据网络 (Named Data Networking， NDN)是目前比较主流的未来互联网体系结构。NDN提出了一种革命性的研究思路，把互联网体系结构的焦点从目前的位置变成内容，用命名的内容取代 IP 地址作为体系结构的核心。在NDN网络体系中，传输机制的研究正在起步，许多关键技术还处于比较基础的阶段。

本文在上述背景的基础上提出了一种基于显示标记的传输控制机制。显示标记是指：对网络链路的状态用特定的数据包标记，并向发送端转发标记的数据包。通过这种方式，发送端能够智能地感知链路的状态，对发送端速率进行调整，避免拥塞的产生。基于显示标记，本文提出了拥塞探测、拥塞通告、发送端速率调整和多路径转发等机制。最后本文将上述机制在ndnSIM中进行实现并在试验中进行了检验，结果表明上述机制网络性能上具备了比较大的提升。

关键词：命名数据网络； 显式标记； 传输控制

# 

AN EXPLICIT MARK BASED TRANSMISSION CONTROL SCHEMA IN NDN

ABSTRACT

With the changing times, the application of the Internet is constantly changing, from the sharing of resources between the host gradually become the content of access and distribution. Users are more concerned about how to quickly and easily access the content, rather than where to get the content. In this new application model, the current Internet TCP / IP-based architecture is gradually adapting, exposing many problems (poor security, no support for mobility, unreliability, lack of flexibility). In order to meet the needs of users changes to solve the TCP / IP architecture problems, many researchers began to study the new architecture. In the new network architecture, named data network (Named Data Networking, NDN) is the more mainstream future of the Internet architecture. NDN proposed a revolutionary research ideas, the focus of the Internet architecture from the current location into content, with the name of the name to replace the IP address as the core of the architecture. In the NDN network system, the transfer mechanism research is starting, many key technologies are still in the basic stage.

Based on the above background, this paper proposes a transmission control mechanism based on display mark. The display tag refers to the status of the network link with a specific packet tag, and to the sender to forward the tagged packets. In this way, the sender can intelligently sense the status of the link, the transmitter rate to adjust to avoid the generation of congestion. Based on the display tag, this paper proposes the mechanism of congestion detection, congestion notification, sender rate adjustment and multipath forwarding. Finally, this paper will be carried out in the above mechanism in ndnSIM and tested in the test, the results show that the mechanism of network performance with a relatively large increase.

**KEY WORDS:NDN, Explicit Mark, Transmission Control**

目录

[第一章 绪论 9](#_Toc475806129)

[1.1 研究背景 9](#_Toc475806130)

[1.2 国内外研究现状 13](#_Toc475806131)

[1.3 本文主要的研究内容 16](#_Toc475806132)

[1.4 论文内容的结构安排 17](#_Toc475806133)

[第二章 NDN网络架构与传输控制机制概述 19](#_Toc475806134)

[2.1 命名数据网络概述 19](#_Toc475806135)

[2.1.1 体系架构 19](#_Toc475806136)

[2.1.2 包格式及命名方法 21](#_Toc475806137)

[2.1.3 节点模型及转发机制 23](#_Toc475806138)

[2.1.4 优缺点分析 26](#_Toc475806139)

[2.2 命名数据网络传输机制 27](#_Toc475806140)

[2.2.1 拥塞控制 27](#_Toc475806141)

[2.2.2 转发策略 35](#_Toc475806142)

[2.2.3 路由机制 36](#_Toc475806143)

[2.2.4 缓存策略 37](#_Toc475806144)

[2.3 本章小结 38](#_Toc475806145)

[第三章 基于显式标记的传输控制模型 39](#_Toc475806146)

[3.1 拥塞检测 39](#_Toc475806147)

[3.2 拥塞通告 41](#_Toc475806148)

[3.3 发送端速率调整 44](#_Toc475806149)

[3.4 多路径转发策略 45](#_Toc475806150)

[3.5 缓存策略 49](#_Toc475806151)

[3.6 本章小结 51](#_Toc475806152)

[第四章 基于显示标记传输控制机制的实现 53](#_Toc475806153)

[4.1 ndnSIM简介 53](#_Toc475806154)

[4.2 拥塞检测的实现 56](#_Toc475806155)

[4.3 拥塞通告的实现 58](#_Toc475806156)

[4.4 速率调整的实现 60](#_Toc475806157)

[4.5 多路径转发的实现 62](#_Toc475806158)

[4.6 本章小结 63](#_Toc475806159)

[第五章 基于显式标记传输控制机制的评估 64](#_Toc475806160)

[5.1 拥塞控制性能评估 64](#_Toc475806161)

[5.2 多路径转发评估 65](#_Toc475806162)

[5.3 本章小结 66](#_Toc475806163)

[第六章 总结与展望 67](#_Toc475806164)

[参考文献 68](#_Toc475806165)

# 第一章 绪论

## 1.1 研究背景

研究背景互联网技术和应用的发展成就了当今的信息社会，但它不是为满足当代需求而设计。互联网产生在于信息共享的需要，将网络中的所有节点抽象为以主机标示符区分的个体，基本的通行模型是建立与两个主机之间的端到端通信[1]。该通信模型也造成了当今网络架构对移动性和间歇性连接支持的不足，而这两点恰恰是当今用户的普遍性需求，尤其是越来越多的移动设备如手机、平板、车载系统、嵌入式系统接入互联网。该通信模型的另一个缺陷在于“认知的不匹配”即：“user/APP”模型关注的是数据即“what”而端到端通信网络关注的是“who”谁拥有所需数据，而要完成模型间的映射需要大量约束和配置如：中间件和湿件（wetware）。再一点就是广受诟病的安全机制，当今互联网体系中的互联网安全机制可以认为是一种“事后”安全机制[2]，是一种通过保护主机和链路实现的安全机制，该安全机制存在大量的风险，尽管可以通过中间件、中间软件实现安全机制的加强，但无法从根本上解决安全漏洞。尤其是随着金融、军事、科研、政府等越来越多的事物实现网络化对安全机制的要求越加严苛。

信息社会的发展，大数据时代的到来，应用程序和用户更加关心的是纷繁数据中对自己有用的数据，以及如何快速高效获取该数据，至于从何处、如何获得则不是其关心的重点，需求的变更与基于 TCP/IP 网络架构存在的桎梏已严重阻碍信息社会的发展。虽然，TCP/IP 网络的演化一直未曾停止，但可惜的是，无论是早期的 MPLS、IPv6，还是后来进行的 Int-Serv、DS-Serv，都无法从根本上解决该网络架构的问题。因此，向下一代互联网(Next Generation Internet，NGI)的演进已成必然趋势。

虽然世界范围内关于下一代互联网的研究项目众多，但可将其分为两大类即： “改良”和“革命”两种，简而言之，凡以维护 TCP/IP网络架构的核心位，仅作策略性优化以适应新的需求的研究项目属于“改良”，改良派的主张虽然能够在某段时间内解决当前互联网架构所面临的某些方面的问题，但随着网络的发展必将暴露出其它问题，而且可能会越来越难以解决，因此，并未从根本上解决 TCP/IP 网络体系中存在的问题[3]。比如：为解决 IPv4 地址耗尽问题而提出的扩展技术 IPv6、为提高 IPv4 地址利用率而采用的私有地址、动态地址和可变长子网掩码(Variable Length Subnet Mask，VLSM)等技术。与之相对应，主张替代当前 TCP/IP 网络架构的主体地位，开发新型网络架构的思想称为“革命”，虽然革命性网络架构的研究在近几年才逐渐展开，却已展现出巨大的优势，尤其是以数据为中心的思想符合世界的发展需求。

随着时代的变化，互联网上应用也在不断的变化，从主机之间的资源共享逐渐变成内容的获取和分发。用户更加关注如何快速方便的获取内容，而不是在哪里获得这些内容[4]。在这种新的应用模式下，当前互联网基于 TCP/IP 的体系结构逐渐表现出不适应性，暴露了许多问题(安全性差，不支持移动，不可靠，缺乏灵活性)。为了适应用户需求的变化，解决TCP/IP 体系结构存在的问题，许多研究人员开始研究新的体系结构。其中，由 Van Jacobson 等人提出的内容中心网络 (Named Data Networking， NDN)是目前比较主流的未来互联网体系结构。NDN 提出了一种革命性的研究思路，把互联网体系结构的焦点从目前的位置变成内容，用命名的内容取代 IP 地址作为体系结构的核心。

自世纪年代互联网出现以来，基于的端到端网络架构沿用至今，然而随着移动互联网技术与应用的飞速发展，目前网络中大部分流量都是以获取数据和服务为目的。据思科数据估计，全球视频流量占所有流量的比例将逐年上升。以主机为中心的网络架构中所有的数据都是以为路由进行传输，网络的中间节点比如路由器在转发完数据包后立即将包丢弃，数据包在网络中不会被重复使用，浪费了网络带宽。另外由于端到端的通信方式，使得网络每一个内容在传输过程中只能为一个用户使用，即使所有的用户请求相同的内容，仍然需要为每一个用户传输一份内容，使得网络中存在大量的冗余流量。此外，网络只提供尽可能的服务，随着网络用户和内容服务提供商的增加，网络的会显著下降。故网络架构不再能满足日益增长的内容分发和内容获取需要。基于这种环境与背景，网络架构和会应运而生。

P2P是基于对等网络加速内容分发的新兴技术，与传统的客户端服务器（在该模式中只有部分用户即服务器需要提供内容服务）工作模式相比，网络中的内容分发服务不需要中心控制器的支持，在网络中的每一个节点同时作为服务器和客户端[5]。这个架构对于网络中的应用层路由、数据发布以及信息共享提供了极大的灵活性。给了—个简单灵活的定义“是一类应用，这些应用能够充分利用程序存储资源、周期、内容以及边缘网络提供的相关资源。因为访问这些分散的资源意味着在连接不稳定、地址不可预测的环境中运行。网络中的节点必须不再依赖于”。具有很多优点：极大地减少服务器负载，降低网络流量，而且在文件共享、视频和语音通话等方面也带来了很好的用户体验，在自组织性、可控可观性、鲁棒性和负载均衡特性上也具有显著优势。因此它的发展极其迅速，在很短的时间内引起了网络用户的高度关注并得到非常广泛的使用。近几年来，应用不断扩张，相应协议也不断更新和涌现。据数据显示，网络运营商业务量中都来源于应用，应用己经成为网络带宽的最大消费者。然而，从底层实现研宄的原理，我们会发现该技术引入了极大的冗余流量，并且使得网络资源分配以及网络利用率极大降低。这说明技术并不是一种真正高效的数据传输模式。与此同时，当前P2P应用的网络协议标准没有得到统一，导致传统流量管理方式不能有效管理P2P应用产生的流量。

内容分发网络是一个应用层之上的网络服务，网络通信中的端到端传输设计原则，在应对互联网新应用时具有局限性，尤其是多媒体应用需求时。CDN通过在服务器上进行缓存存储与内容调度，有效缓解了上述问题。部署在协议层之上，作为应用层为用户提供服务。用户的请求不直接由服务器响应而是由基础设施决定，服务提供商在全球部署多个内容服务器，用户请求内容时通过重写或者重定向等技术，使得用户能够根据地理位置实现就近获取内容。内容分发网络的优势是：网络架构工作在应用层，无需改变底层网络构造和骨干设备配置，就可以实现内容分发，因此具有很好的灵活性和可扩展性。但是CDN就近获取资源能力有限，只能获取静态资源，并且内容更新时需要分发到其他节点，代价较高。另外，CDN不改变网络协议的特性[6]，限制了内容分发的有效性。

由于P2P和CDN技术的局限性，网络架构研究者提出了信息中心网络，解决方案。不同的是，解决方案针对整个网络体系结构，其目标不只是解决内容分发问题，而是要解决当今互联网存在的所存问题，属于革命性的方案。

大数据时代的主体是数据，未来的社会必将是以数据主导的社会，因此以数据为中心的网络架构研究已经吸引了足够的关注，早在 NDN 作为一个全新的网络体系架构，它的设计源自于当今互联网的成功和数据时代的需求，反映 TCP/IP 网络架构优势的同时也暴露出其巨大缺陷。NDN的核心在于命名数据，以数据固有信息代替基于主机地址信息的数据转发机制，采用层次化命名结构命名数据，并基于命名数据名称进行路由，使网络节点拥有直接提供需求数据的功能。该网络架构更加关注于数据本身，将数据的获取从基于地址到内容映射的采集机制中剥离出来。同时，NDN 网络架构将数据的保护与主机保护解耦和，从对数据内容的事后安全机制即通过保护主机和链路实现的安全机制转变成对数据本身的直接保护[7]，从根本上实现通信的安全性。NDN 网络上广泛分布的缓存备份可以提高路由的效率和反馈速率，同时对TCP/IP 网络的兼容保证了 NDN 网络的可扩展性。

NDN 架构以数据为中心、消费者为驱动的属性很好的满足了大数据时代对网络体系架构的需求。和传统的 TCP/IP 网络相比，NDN 主要有以下特点：(1) 以命名的内容为中心，路由基于内容的名字而不是 IP 地址；(2) 引入网内存储机制，路由器可以缓存内容，请求是由整个网络而不是单一的数据源响应；(3) 增加了策略层，完成数据的转发功能，实现转发和路由的分离；(4) 设计了接收端驱动传输模式，在接收端实现传输控制；(5) 保护内容的安全，而不是通信信道的安全。传输控制机制是 NDN 体系结构中有待研究的关键技术之一，和传统的 TCP/IP 网络相比，NDN 中的传输主要有以下特点：(1) 面向无连接，一个接收端可以从多个数据源获取请求的内容；(2) 基于接收端驱动 (Receiver-Driven) 的“Pull”模式，接收端发送 Interest 包请求内容，满足请求的数据源节点返回所需的 Data 包。如果接收端没有在规定的时间内收到 Data 包，可以通过重传 Interest 包实现可靠传输；(3) 采用 One-Interest-one-Data 的传输模式，通过控制 Interest 包来控制 Data 包的流量，从而实现流量控制。NDN 网络的拥塞控制主要存在以下问题和挑战：(1)传统 TCP 的隐式 (Implicit) 拥塞检测机制在NDN 中不再适用。NDN 中由于网内缓存的存在，数据是多源的，TCP 基于单一源的RTO(重传计时器)超时机制在 NDN 中不再可靠；(2) AIMD 算法会引发RTT公平性问题。指出在NDN中盲目使用传统 TCP 协议的AIMD拥塞控制算法可能引发公平性问题。使用 AIMD 算法，吞吐率反比于 RTT，在NDN 网络中，由于有网内缓存，流行的内容 Cached离消费者更近，有更小的延迟，这会导致链路的大部分带宽被流行的内容占据，从而引发对不流行内容的公平性问题。因此，需要重新对 NDN 网络中拥塞控制算法进行研究，目前对于 NDN 网络拥塞控制算法的研究尚处于起步阶段。本文针对 NDN 中拥塞控制的问题，提出了一种基于显式反馈的拥塞控制算法。该算法实时的检测网络的拥塞程度，并且把拥塞信息反馈回接收端，接收端会根据网络不同的拥塞程度设计不同的速率调节算法。通过控制Interest包的发送速率来控制发送端Data包的发送速率，实现了接收端驱动(Receiver-Driven)的拥塞控制。

## 1.2 国内外研究现状

信息中心网络体系结构的主要研究机构和研究项目在欧洲( PSIRP/PURSUIT， NetInf)和美国(CCN和DONA)。与传统IP网络相比，这些项目至少可以获得如下两个方面的优势：一方面，通过内容缓存，可以减少冗余数据传输，降低链路带宽消耗，最终节省设备成本和能耗[8]，即“存储换带宽”；另一方面，直接保护信息而不再是保护传输信息的容器(数据包)，完成了从信任主机到信任信息的设计思路转变。

发布订阅网络路由模型(Publish Subscribe Internet Routing Paradigm， PSIRP)创新性地提出一种基于发布和订阅模型的，以信息为中心的网络架构模型，PURSUIT(Publish Subscribe Internet Technology)项目是是PSIRP的后续研究，旨在设计一个基于发布订阅模型的全球路由策略。PSIRP/PURSUIT项目提出了一个革命性的路由架构，将现在的发送一接收模式变为发布订阅模式。PSIRP/PURSUIT使用与DONA类似的命名方案，其路由结构包括聚合系统、拓扑、路由和转发四个模块。聚合系统在每一个域中包括一个聚合网络(RendezvousNetwork， RN)，它们由聚合标识符(Rendezvous Identifier，RI)连接，RI作为发布者和订阅者之间的中间者，完成发布者请求解析与订阅者内容匹配功能。拓扑结构在每一个域中包括一个拓扑节点(Topology Node } TN) } TN负责域内拓扑和负载均衡，TN之间使用类似于BGP路由协议交换不同域内的路由信息，实现域间路由。路由结构由一系列的分支节点(Branching Node BN)组成，建立和维护分发树，BN使用TN中保存的路由信息将订阅者的请求路由到内容源处。转发模块由一系列的转发节点(Forwarding Nodes FNs)。FNs使用基于Bloomfilter的简单快速转发算法将内容转发给订阅者。订阅者通过本地RN获取内容位置，域内BN节点从TN节点获取网络拓扑信，并转发内容订阅者请求。

NetInf(Network of Information)分为两个研究阶段，最初属于4WARD研究项目[8]，第二阶段则属于SAIL(Scalable and Adaptive Internet Solutions)o NetInf的核心是以信息为中心，设计基于信息的网络通信架构，同时考虑网络安全等其他因素。NetInf旨在独立于位置信息处理数据，信息、包含在信息对象中，每个信J息对象拥有一个全局唯一的身份标识符，该标识符被用于引用该信息对象，可以存储在信息网络的任何地方而不受位置限制，信息对象可以在网络中任意传播和存储。可以发现流行度较高的信息对象会被存储在离请求者较近的节点，从而提升信息传播效率，增强网络应对用户需求增长的可扩展性[9]。NetInf通过名字解析和名字路由提供两种获取数据对象的模型，以适应不同的网络环境，NetInf还定义了域间接口用于异构网络之间的名字解析及路由。NetInf两个最典型名字解析机制是：第一个是MDHT(Multilevel Distributed Hash Table)， MDHT是一种分层的DHT结构，采用多层DHT结构解析名字，第二个是基于LLC ( Late LocatorConstruction)的NRS ( Name Resolution System)解析方案，适用于动态网络拓扑，NRS将命名对象解析为传统URL，所以可以使用现有的HTTP协议进行传输通信。下面以NRS命名解析为例说明NetInf通信过程：根据在本地网络中使用的通信模型，内容源注册一个名字来发布内容，并将名字与NRS(绑定，之后根据名字确定路由信息；每一个NetInf节点都有一份复制的数据对象(包括网内缓存以及用户终端)，这些NetInf节点可以在NRS系统里进行备份注册(添加一个新的内容名与位置信息的绑定)。如果NRS系统可以提供服务，则用户先将名字对象发送给NRS， NRS通过解析返回相应的位置信息，用户收到位置信息后可以从网络中选出最近最优资源进行获取。

DONA (Data- Oriented Network Architecture)是美国伯克利大学RAD实验室是出的研究性项目。DONA采用新的命名方式和解析结构，DONA使用扁平的括名空间，命名形式为P：L，其中P字段，是内容发布者公钥的哈希散列值，La对象标识。DONA主要由两种包类型：REGISTER(P：L)和FIND(P：L)。被授权内数据服务节点使用REGISTER注册数据，FIND用户发送用户请求到RH， RH内目的是找到距离用户最近的内容副本，所以根据RH的路由信J自、，用户请求被各由到最近的副本，如果没有副本命中，内容源将内容直接返回给用户。DONA孟过RHs实现基于名字的路由，RHs使用分层的解析方案，每一个域都有一个RH，当前RH没法处理的请求将被转发至上层RH。任何RH都可以缓存内容，数据返回到内容请求者之前都会先到达RH进行缓存，而且缓存副本可以响应FIND请求。最初，DONA使用最长前缀匹配方案，但考虑到最长前缀在可扩展性方面遇到的问题，提出了最深匹配方法。DONA没有自己的传输控制协议，仍然依赖于TCP进行传输控制，所以说，DONA底层仍然没有摆脱IP。但是相比于IP网络，DONA具有以下优点：1、解决移动性问题。针对移动性场景，DONA通过简单地将主机RH切换到新网络的RH(利用DHGP可以完成)，就可以处理用户移动性问题[10]，任何发送给之前RH的内容请求会被重定向到新的RH； 2、使用RH替代DNS解析机制，效率提高的同时具有更好的鲁棒性，而且命名系统中扁平名称结构能够保证不变性。但是DONA并不属于革命性解决方案，由于依赖IP，随着网络流量增加，会导致路由表增长过快，对于内容提供者，也不会希望自己的内容任意缓存在网络中，所以需要法律等其他措施进行约束。

由于日益增长的有效内容分发业务的需求，未来网络架构的发展以命名数据对象(Named Data Objects NDOs)为基础。信息中心网络是未来网络研究项目中的典型代表，该架构通过复制利用网络缓存和多端通信方式，其传输模型使得发送者和接收者解祸合，目的在于提供一种更适合今天(特别是内容分发和移动性)的网络基础服务。信息中心网络采用以内容为中心的命名、寻址、缓存、路由方式，把信息当作网络的基本实体，通过内容名称进行定位和寻址[11]，目的在于设计一个能够有效获取内容以及内容分发的网络架构。

以上几种未来互联网架构，虽然在实现细节上各有不同，但其基本思想都是将数据作为核心。在众多基于 ICN 的未来互联网架构中，NDN 是目前最能代表 ICN核心思想的未来互联网架构。因此，NDN 得到了全球范围内众多学者的研究，在众多关键技术的研究上已经取得了公认的研究成果。由于 NDN 与目前的 Internet 最大的区别在于，NDN 中的每个节点都具有缓存数据的功能，因此数据包可以沿着多条路径转发[12]。因此，NDN 的缓存策略和转发策略和对整个 NDN 架构的性能有着决定性的影响，所以，对 NDN 中数据转发策略和缓存策略的研究便成为了该领域的研究热点。

近年来，越来越多的国际标准组织和政府机构参与到下一代互联网架构的研究，其中，以美国的全球网络创新环境项目(Global Environment for Network Innovations，GENI)、欧盟 FP7 的 4WARD 、日本的 AKAR-I、ITU 的 Future Network最具代表性，我国也逐渐建立起了下一代互联网架构的研究组织并努力发展自己的项目，其中 (China Next Generation Internet，CNGI)代表了中国下一代互联网架构研究的最新结果，成为下一代互联网架构研究的优秀代表。在NDN的传输控制机制方面，目前的研究还处于起步阶段。NDN中拥塞控制的研究大部分是类TCP的隐式反馈机制，即通过数据包超时来指示链路可能发生的拥塞，并不能直接对拥塞发生的链路、拥塞的程度进行检测；转发策略方面NDN采用最优路径优先的转发策略[13]，把网络路径的条数或者长度作为衡量路径优先级的指标；国内外学者提出的多路径转发策略都是在单路径转发策略基础上的研究，并没有对网络性能实质上提升，这些现有的传输控制协议不能利用NDN本身的优势。

## 1.3 本文主要的研究内容

NDN 作为下一代互联网体系架构研究中革命性框架的代表，在瘦腰模型中以命名数据块代替 IP 数据包，彻底颠覆基于地址映射的数据采集和传输机制，直接以数据名称完成网络操作，数据本身成为网络传输的主体，整个网络架构是用户请求、需求数据为驱动的。但是，当今基于 TCP/IP 网络架构在几十年的发展中所展现的巨大优势是有目共睹的，因此，NDN 网络架构的设计离不开当今互联网的成功，而在设计 NDN 网络架构时也充分考虑到了这些，比如说：“瘦腰”模型作为 TCP/IP 网络爆炸式发展的关键性设计，在 NDN 网络中得到了保留和优化，直接以命名数据块(Content Chunks)代替 TCP/IP 体系架构中的 IP 数据包，直接以命名数据作为路由的主体，从根本上改变了路由策略并保持了对 IP路由的兼容。另外，作为以数据为中心的网络架构，保证数据平面和转发平面的分离，保证数据转发效率的同时，实现平台独立优化。对于 TCP/IP 网络架构中备受诟病的安全机制，NDN 建立起一种构造到架构中的安全策略，区别于 TCP/IP以保护主机和链路实现的时候安全机制，NDN 从保护数据本身建立安全机制，包含命名数据名称、数据本身和数据生产者签名的安全机制，建立起从数据产生到数据到达请求者的全称数据安全机制。

NDN的性能主要与传输控制机制有关，传输控制机制主要包括拥塞检测、拥塞反馈、拥塞控制、转发策略、流量整形和多路径转发等技术。目前NDN的传输控制机制主要是参照和模仿TCP中的相关方式，例如拥塞检测基于超时，当数据包的响应时间超过RTO时认为网络中的链路发生拥塞；拥塞控制采用类TCP的AIMD发送端速率调整算法；转发策略是单路径的，以路由条数作为唯一的指标，选择条数最少的路径最为最优路径进行数据包的转发。经过实践证明，上述的几种技术不适用于NDN，传输控制机制的研究成为NDN研究的关键问题，成为解决网络性能优化问题的重要部分。

本文主要对NDN的传输控制机制进行研究，提出了了一种基于显式标记的高效传输控制机制。上述传输控制机制主要的关键技术包括：拥塞检测、拥塞通告、拥塞控制、路由选取和多路径转发等。在NDN的仿真平台ndnSIM对上述关键记住进行了仿真实现，在模拟的网络场景下，对传输机制的性能进行了测试，实验结果表明，基于显式标记的传输控制机制对网络性能具有明显的提升，主要体现在吞吐率变大、平均延迟降低和丢包率减小等方面。

## 1.4 论文内容的结构安排

本文共分为六章，主要研究 NDN 网络中高效传输控制机制设计和实现，提出一种基于显示标记的传输控制机制，对相关机制进行了建模，并进行了仿真实现。

第一章 绪论。对下一代互联网架构的研究和本文的研究背景、意义、目的做了分析，对本文提出的传输控制机制进行简要分析。之后介绍了本文的主要工作和结构安排。

第二章 NDN 网络架构与传输控制机制的研究。本章重点剖析了 NDN 的设计原则、体系架构并深入研究了 NDN 的转发策略；同时对 NDN 的传统路由、拥塞控制和转发策略算法进行了简单的阐述；另外分析了两种转发策略即GC 和 FPF转发策略和基于隐式的拥塞控制机制，以及其各自的优势和在此基础之上的显示标记传输控制算法。

第三章 显式标记模型。本章是对显式标记模型的描述及其在 NDN 网络中的应用研究。本章着重介绍显式标记模型的原理和应用。

第四章 NDN中基于显示标记的传输机制实现。本章首先对NDN的仿真平台ndnSIM进行了介绍，然后根据NDN 网络仿真平台ndnSIM的构建了一个网络模型，然后根据显式标记思想及NDN网络的缓存特性构造出基于 NDN 节点智能感知能力的模型，在ndnSIM中对模型进行了实现，包括拥塞探测模块、拥塞控制模块、发送端速率调整模块、多路径转发模块和缓存部署模块，在传输机制实现的基础上，搭建了网络拓扑，模拟生成了发送端和接收端，仿真了实际的网络数据传输场景。

第五章基于 ndnSIM 平台的基于显式标记传输控制机制的评估。本章介绍了在多种实验场景下，对网络传输的效率进行了测试，对网络数据进行了统计分析，并对其性能进行评估。

第六章 总结与展望。本章将对全部的研究工作进行总结，概述研究工作的优缺点和取得的成果，在此基础之上提出下一步工作的重点和难点，为以后的科研工作起到指导性作用。

# 第二章 NDN网络架构与传输控制机制概述

命名数据网络NDN体系结构与TCP/IP网络非常相似，都是沙漏模型，只是NDN使用内容名代替IP作为网络细腰[14]，并且NDN中内容层可以承载在UDP/TCP/IP等众多协议之上。另外NDN中，任何节点都可以缓存内容的分布式缓存策略，使得用户可以就近获取内容，极大地提高了用户体验。NDN对于移动性应用场景也具有很好的支持。本章将介绍内容中心网络的包格式、协议及通信流程，然后着重从命名数据网络的传输性能角度进行分析，总结内容命名数据网络在传输控制机制的研究需求。

## 2.1 命名数据网络概述

### 2.1.1 体系架构

IP网络中的沙漏结构优雅而且强大，网络依赖通用的网络层构建，使网络实现互联所需的计算机资源最小化。ND沿袭了这种沙漏架构，提出了一种新的基于内容名字的细腰模型。在NDN中，不再使用主机位置而是使用内容名字进行路由，图2-1将IP和NDN网络协议栈进行了对比，它们都采用分层结构，细腰之外，很多协议都是端到端协议：例如，数据链路层是一个对端协议，传输层则是服务器和客户端之间可靠传输协议。IP细腰模型的特点是：唯一需要通用的是第三层的网络层协议，由于网络层的简单以及对第二层协议的要求宽松IP网络模型取得了巨大成功。而NDN继承了TCP/IP协议这一优点，NDN可以架构在包括IP的任何协议之上。NDN与IP的协议栈对比图如图2-1.

相比IP网络，NDN有两个突出特点，这与协议栈中的新加入的策略层和安全层有关。一是由于与第二层的简单关系，NDN可以采用多路同时连接(例如以太网、3G、蓝牙、802.11)，为了在不断变化的应用场景下都能发挥多路连接优势，NDN的策略层使用细粒度的、动态最优选择策略；二是NDN中基于内容的安全性概念：保护以及信任的是内容本身而不是它所承载的连接。在NDN中，所有的内容都要验证数字签名，私人保护内容都进行加密，如果用户需要使用最近的内容副本，必须具有验证内容的能力。

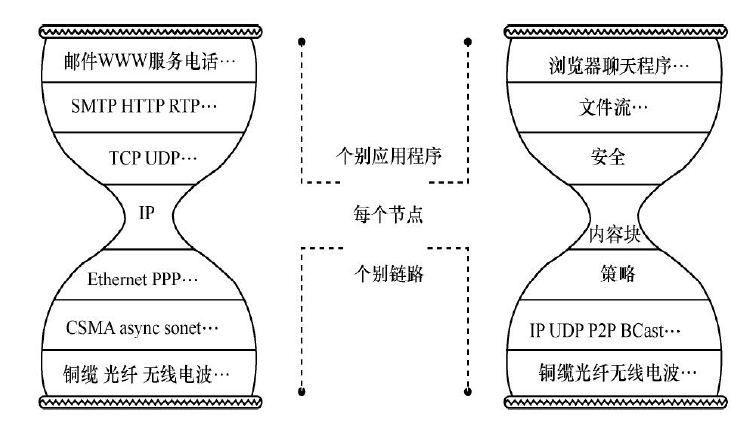


图2-1 IP网络架构与NDN网络架构对比图

通过图2-1我们可以看出：NDN体系结构的外形也是沙漏模型，这与现有的TCP/IP网络架构相似，但是中间的细腰层由IP数据包改变为带有数据名称的内容块，充分体现了以数据为核心的设计理念。还有一点不同的是网络内置了存储功能，由于数据包内携带的IP地址改为了数据名称，NDN实现了传输过程中数据与位置的解祸，数据包缓存在转发过程中经过的路由器中，这样就可以使请求相同数据的用户获得更小的响应时间。

另外内容层的上层为安全层，NDN通过在数据分组中加入签名信息提供网络内建的安全功能，接收者在收到数据后可以通过验证数据分组内的签名信息来判断所接收到的数据是否在转发过程中被恶意修改。其中签名信息包括摘要算法，数据发布者凭证信息等。

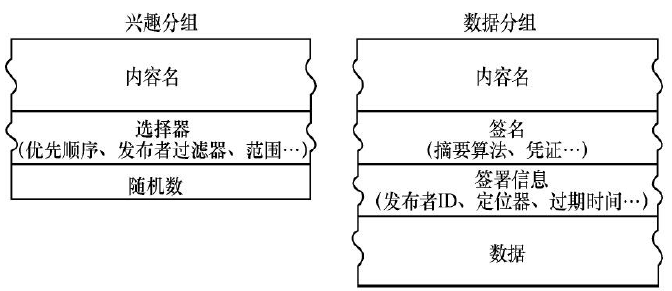
内容层的下层是策略层，包括数据分组的转发策略，路由策略和存储策略。路由策略方面，在NDN架构中，网络的路由机制也不再基于IP地址，而是基于内容命名进行路由。内容路由与IP路由的目的类似，即让各个网络节点知晓网络中的内容情况，以便于对请求进行转发。目前普遍使用OSPFN路由算法，类似传统的OSPF协议。另外有一部分研究同样也是采用了各种优化算法实现基于传统的OSPF协议进行改进。还有一部分研究结合NDN网络中的缓存设计路由策略，使得路由充分利用缓存资源以提升性能。转发策略方面，NDN的转发策略能够根据数据平面的状态调整转发决策。该领域的研究主要通过观测Interest分组、Data分组的流量或历史痕迹，检测到连接失败、拥塞等信息，进而寻找替换的路径来避免类似问题。己有的研究包括：通过传递探测信息来感知更优的路径；通过多路径转发来自适应地选择最优的通信路径；通过结合缓存状态来设计动态可调的转发策略。它们都在一定程度上增加了通信开销或其他代价，但是能够有效地避免链路失效或产生延迟。缓存策略方面，随着存储技术的日益成熟，其价格不断降低，这也是NDN架构提出的基础之一，即采用全网节点部署缓存的方法来实现未来网络内容的高效分发。由于每个数据分组都独立于其源目的地址，路由器可以将它缓存到数据库中响应后续的请求。NDN路由器中的数据库与IP路由器中的Buffer类似，都是用来缓存数据分组的，区别是IP路由器将数据分组转发之后不能再利用该数据分组，而NDN路由器还可以重复使用该数据，因为在NDN中该数据可以根据命名识别。除了增强内容的分发效率，NDN的缓存还可以用来提升分组丢失重传的效率。NDN领域目前有很多研究集中在缓存策略的设计方面，包括缓存替换策略、放置决定策略、管理技术等。现有大部分缓存策略研究主要分成2类：一是通过建立数学模型或仿真实验的方法，评估NDN系统现有缓存策略性能；二是提出各种新型缓存策略以提升网络性能。

在策略层之下，和IP体系类似的，有一些底层的物理链路，这与OSI参考模型中的物理层和数据链路层也是相对应的。由于NDN在设计的时候就考虑了向后兼容，因此NDN可以基于现有的网络架构以覆盖网(Overlay Network)的形式存在，也可以作为一种全新的网络架构去实现。为了支持基于名字的寻址以及传输，NDN体系的下层对UDP，IP协议是兼容的，不仅如此，还同样兼容P2P、广播等机制。

### 2.1.2 包格式及命名方法

NDN中包含2种数据分组，分别是兴趣分组(Interest Packet)和数据分组(Data Packet)，其中兴趣分组负责请求数据，数据分组负责回传数据。两种分组结构如图2-2所示。

兴趣分组包括所请求数据的名称，NDN采用类似URL似的分层式命名机制，例如所请求的数据为youku.com发布的视频school.avi， Interest包中的数据名称即为：/youku.com/videos/school.avi/p 1/。其中/youku.com/表示请求文件的发布机构，/videos/表示请求的数据种类，/school.avi/表示请求的数据名称，由于视频文件较大，不可能放入一个数据分组中回传，所以在传输时会将该视频分割成较小的数据块依次回传，/p 1/表示第一个数据块。除了所请求的数据名称，兴趣分组还包含请求者对于数据的筛选条件，包括数据的发布者、数据的发布时间等等。兴趣分组的最后还会增加一个随机数，此随机数的作用是避免出现Interest包路由环路，NDN网络内置多路径Interest包路由功能，但是同样的一个Interest包可能会在网内经过一个环路回到当前路由，当收到一个内容名和随机数都相同的Interest包后，路由器会丢弃该Interest包，从而避免路由环路情况发生。



2-2兴趣分组和数据分组示意图

数据分组包括与兴趣分组对应的内容名，以及数字签名，签署信息和数据。其中内容名作为与Interest包对应的回传依据。数字签名负责保证数据的安全性，当请求者收到Data包后，会通过摘要算法验证数据凭证是否正常，从而判断数据是否安全，一旦数据被篡改，数字签名即无法验证，接收者直接丢弃该数据分组，这种模式使得请求者可以无视数据转发过程中可能存在的安全问题，体现了NDN网内自带的安全防护功能。签署信息包括数据分组的发布者ID，定位器，过期时间等，其中定位器用于获取密钥，而过期时间则规定了该数据分组在网内失效的时间。最后是数据部分，用来承载数据。

### 2.1.3 节点模型及转发机制

如图2-3所示，其中转发信息表FIB与TCP/IP架构中路由器的转发表的功能基本类似，通过将内容名前缀与转发端口匹配，是Interest包转发路径的依据。与TCP/IP架构中路由器的转发表不同的是，它允许匹配多个端口进行转发。如图2-3所示，当此节点收到以/parc.com/为前缀名的Interest包时，会将此Interest包从端口0和端口1转发出去。

待满足兴趣表PIT记录了己经完成转发的Interest包信息，其中包括Interest包的名称前缀以及对应的请求端口。当Interest包命中数据后，Data包依据此表的内容依次传回请求节点，并沿路删除PIT条目。PIT条目具有生存时间，条目在生存时间结束时，即使仍未收到Data包，也会被删除。

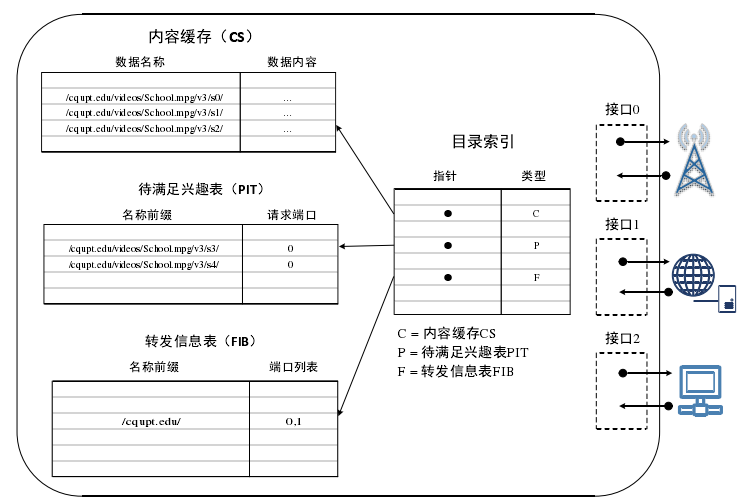


图2-3 NDN网络节点图

内容缓存CS类似于IP路由器的缓存buffer，用来存储需转发的数据，但是不同于IP路由器的缓存，NDN网内节点在每次通信结束后不会直接清空缓存，而是根据缓存策略决定是否存储该内容用于下次通信。NDN以内容名作为细腰层，实现了数据与网络地址的解祸，数据在网内可以脱离特定节点而独立存在，这种模式可以显著提高数据的分发效率，减少内容下载时延和网络带宽占用。

NDN网络的通信过程是由接收端驱动的一个带有对于数据名称的Interest包，Interest名称的Data包，然后再沿原路径返回请求者当某个节点需要数据，会向网内发送包在网络内被转发直至命中具有相同具体分组处理流程如下。

1.Interest包处理流程(如图2-4)：

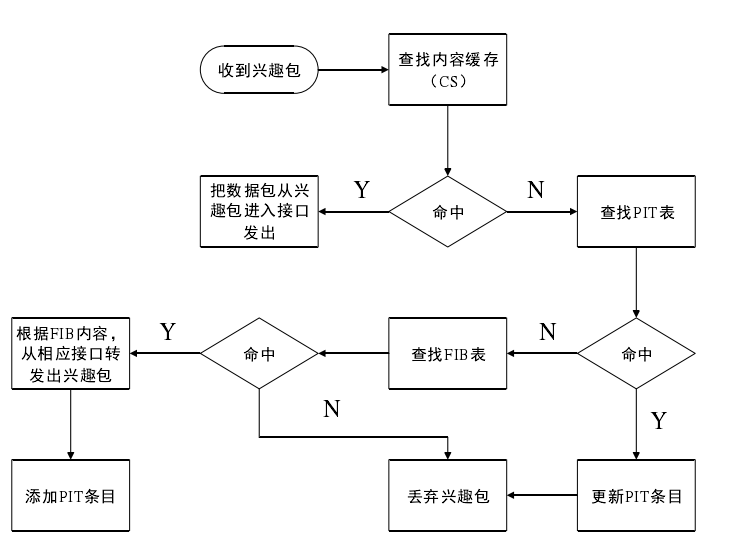


图2-4 Interest包处理流程图

路由节点对接收到的Interest包处理过程如下：

(1)查找该路由节点的内容缓存CS，若匹配到与Interest包中名字对应的数据，则生成Data包并从收到该Interest包的接口发送出去，然后丢弃该Interest包；

(2)如果在内容缓存CS中没有找到匹配的数据，则查找待满足兴趣表PIT，如果PIT表中存在相同名字的Interest包转发记录，那么在该PIT条目中的请求端口中添加当前Interest包接收端口，并丢弃该Interest包；

(3)如果PIT中没有记录该数据名字的PIT表项，则查找转发信息表FIB。如果找到了相应的信息，则根据FIB条目中的对应接口发送Interest包，并在PIT中记录Interest信息，包括数据的名字、随机数、进入的接口等信息；

(4)如果在FIB中也没有找到对应条目，则丢弃该Interest包。

2. Data包处理流程(如图2.5)：

路由节点对接收到的Data包处理过程如下：

(1)查找内容缓存CS，如果缓存中己经存在该数据，则直接将该Data包丢弃；

(2)如果内容缓存CS中该数据，则在PIT中查找是否有对应的表项，如果有则将该Data包缓存到CS中，并按照PIT表中记录的所有请求接口发送该Data包；

(3)如果PIT中也没有与该Data包对应的条目，则有可能是该Data包在PIT中的Interest表项的生存周期到期而被删除，也有可能这是一个恶意的Data包。这种情况下路由节点会删除该Data包。

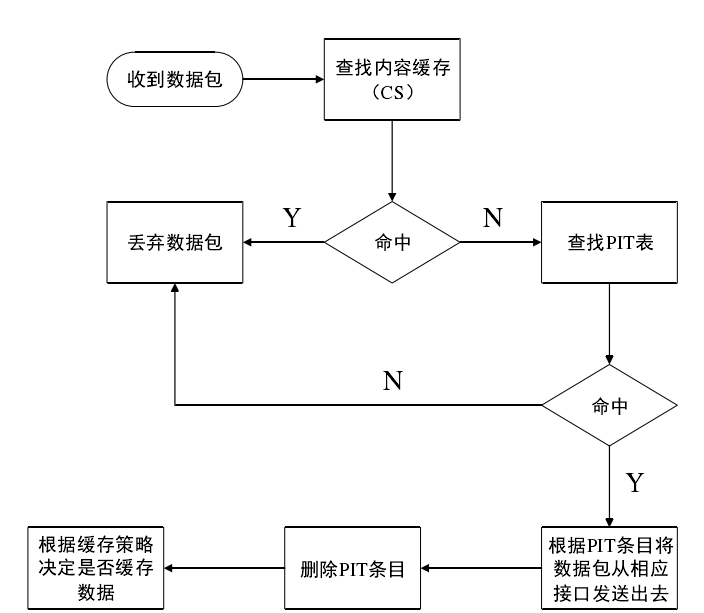


图2-5 Data包处理流程图

以用户A向内容源B发送兴趣包获取内容为例，来说明NDN中的通信流程，用户A与内容源B之间可能存在若干中间路由节点，为简洁起见，以一个中间节点C代替。其中用户A也可以被称作节点A，同样内容源B也可以被称作节点B或是用户B。当用户A需要请求内容时，会主动发送兴趣包；中间节点C收到兴趣包后会依次查询本节点CS， PIT和FIB，如果C节点的CS中没有缓存用户A请求的内容，C在本节点PIT上记录接收到兴趣包的端口号：然后C根据本节点FIB转发兴趣包。每个兴趣包都有一个nonce值，该值是一个随机数，C接收到兴趣包时，会判断兴趣包的名称，当接收到同名称但不同nonce兴趣包时，NDN才会在PIT中添加收到该兴趣包的端口；如果接到到的兴趣包名称与nonce值均与节点之前收到的兴趣包相同，则NDN节点将简单丢弃这类重复的兴趣包。内容源B接收到兴趣包后，首先查找本节点CS，如果CS中存在用户A所请求的内容，则根据PIT记录的端口信息按原路径转发内容包。同时，内容经过的中间节点C会将内容副本缓存在C节点CS中。如果在B节点CS中没有匹配到内容，则B节点的Repository会构造相应的内容包，并通过C返回给A，同时会在B和C的CS中缓存该内容。

### 2.1.4 优缺点分析

NDN由于其以内容为中心的特点，以及PIT， FIB和转发策略等路由机制，使得NDN可以很好地解决IP网络中的一些问题。与IP网络相比，NDN具有如下优点：

1)安全。NDN验证所有内容，包括路由和策略信息，所以能够防止被欺骗或者信息被篡改，而且NDN中，消息只与内容有关，而不涉及主机信息，使得恶意数据包很难发送到特定目标。为了确保消费者获取的内容为自己需要，消费者可以指定内容发布者的公开密钥或者内容发布者的认证key，这种认证方法使得用户免受攻击。当前的网络架构在设计之初是没有考虑安全防护问题的，现有的网络安全方面的工作都是在近年来发展起来。命名数据网络通过对所有数据进行签名认证，保证数据的安全性。

2)流量调节。NDN中兴趣包与数据包之间的一一对应的规则保证严格的流量平衡。另外，兴趣包拥有类似于IP网络中TCP的窗口机制，用户可以同时发送多个兴趣包获取内容，用户可以根据需要动态地改变窗口大小。

3)性能。由于NDN的PIT聚合机制以及FIB多端口转发特点，使得NDN天然支持内容高效分发和多播传输功能。尤其是多个用户请求相同内容时，可以极大的减少网络流量。当面对网络错误时，端到端的理念可以保证应用的稳定性。命名数据网络继承并发展了这一原则.

4)绿色。NDN使用内容名取代IP地址，不仅减少了网络中的冗余流量，而且使得传输的内容基本为有效内容，互联网中基本都是有效能量，相对于CDN和P2P具有明显的优势。保留沙漏式的网络架构。因特网之所以能实现今天的成功，很大程度上取决于其沙漏式架构设计。以全局统一的网络层(IP协议族)为细腰使得网络的扩展变得更加方便，同时为了满足新的应用需求，上层或者下层技术在实现创新时可以不受约束，这个结构的优势是可以在方便全球互联的同时，支持网络层以外各个层级的繁荣发展。

5)简化应用部署。目前基于NDN的应用己经较多，并且有NDN开源代码实现，NDN的overlay部署方式简单方便，可以通过overlay进一步过渡到cleanslate部署方案，而且NDN不需要关心IP地址和内容的映射关系，使得部署更简单。路由平面与转发平面的分离己经被公认为网络发展的必需原则，因为这样可以使路由系统不断发展的同时保证转发平面稳定工作。命名数据网络沿用了这一设计原则，使得最佳转发技术的使用可以与新的路由系统研究同时进行。

当然，NDN由于自身体系结构的原因，也有一些缺点，比如不带有推送功能。NDN采用分级命名方案，这种方案在安全上不如扁平命名方案，体系结构的可演化问题等。

## 2.2 命名数据网络传输机制

### 2.2.1 拥塞控制

拥塞现象是指到达通信子网中某一部分的分组数量过多，使得该部分网络来不及处理，以致引起这部分乃至整个网络性能下降的现象，严重时甚至会导致网络通信业务陷入停顿，即出现死锁现象。这种现象跟公路网中经常所见的交通拥挤一样，当节假日公路网中车辆大量增加时，各种走向的车流相互干扰，使每辆车到达目的地的时间都相对增加（即延迟增加），甚至有时在某段公路上车辆因堵塞而无法开动(即发生局部死锁)。

网络的吞吐量与通信子网负荷(即通信子网中正在传输的分组数)有着密切的关系。当通信子网负荷比较小时，网络的吞吐量(分组数/秒)随网络负荷(每个节点中分组的平均数)的增加而线性增加。当网络负荷增加到某一值后，若网络吞吐量反而下降，则表征网络中出现了拥塞现象。在一个出现拥塞现象的网络中，到达某个节点的分组将会遇到无缓冲区可用的情况，从而使这些分组不得不由前一节点重传，或者需要由源节点或源端系统重传。当拥塞比较严重时，通信子网中相当多的传输能力和节点缓冲器都用于这种无谓的重传，从而使通信子网的有效吞吐量下降，对资源的需求超过了可用的资源。若网络中许多资源同时供应不足，网络的性能就要明显变坏，整个网络的吞吐量随之负荷的增大而下降。

端到端拥塞控制的IETF标准关注的方面包括集中在特定的协议（例如TCP协议[RFC2581]，可靠的多点传送协议[RFC2357])；终端节点和路由器之间的拥塞信息(例如明确的拥塞通告[RFC2481])交换的句法和语义；不同服务的服务质量的期望值。端到端的拥塞控制的作用也在一个关于“Internet中的队列管理和避免拥塞的建议”[参见RFC2309]的RFC报告中进行了讨论。RFC2309提出了在路由器中活跃的队列管理机制的配置和对路由器机制设计的延续来处理对拥塞通告无回应的流。我们能够轻松地从RFC2309中借用一些端到端的拥塞控制的概括性的讨论。

与上面提到的RFCs资料相比，本文档对拥塞控制的原理进行更一般性的讨论。Internet成功的一个关键因素就是TCP协议的避免拥塞机制。当前TCP协议在Internet中仍然是占主导地位的传输协议，但它不是适用于任何地方，有越来越多的应用由于某种原因没有选择使用TCP协议。通信不仅包括多点传送通信，而且包括单点传送通信，诸如不需要可靠性的流化的多媒体，以及包括象DNS(DomainNameServer域名服务器)或路由信息的通信，它们带有被认为对网络运行至关重要的短信息。许多通信并不使用任何形式的预留带宽或端到端拥塞控制。为了保持最优传输量，端到端的拥塞控制的继续使用对保持Internet的稳定至关重要。

本文讨论IETF在新的拥塞控制协议标准化中的一般作用。对于区别性服务和集成性服务的拥塞控制的讨论在本文档中不涉及。集成性或区别性服务能够保证端到端的网络带宽，所以不需要端到端的拥塞控制机制。

确定拥塞窗口的大小的过程为：在刚建立连接时，将拥塞窗口的大小初始化为该连接所需的最大连接数据段的长度值，并发送一个最大长度的数据段（当然必须是接收窗口允许的）。如果在定时器超时前得到确认，将拥塞窗口的大小增加一个数据段的字节数，并发送两个数据段，如果每个数据段在定时器超时前都得到确认，就再在原基础上增加一倍，即为4个数据段的大小，如此反复，每次都在前一次的基础上加倍。当定时器超时或达到发送窗口设定值，停止拥塞窗口尺寸的增加。这种反复称为慢速启动，所有的TCP协议都支持这种方法。

拥塞控制不仅是网络稳定、高效运行的关键，同时又是实现各种服务质量的基础和前提。实际的网络是一个不断发展的系统，网络拥塞控制研究也是一个非常困难、有挑战性的研究领域。对网络拥塞控制的研究仍有许多工作要做，进一步的工作包括：

1) 拥塞控制基于端主机的控制策略和路由器的队列管理策略存在相互影响、相互作用的关系，如何在网络模型描述的基础上，从控制系统的角度将两者结合起来，设计出最优的拥塞控制策略，是网络拥塞控制研究的一个方向。

2) 主动队列管理技术通过丢包积极响应拥塞，来达到拥塞避免和缓解的目的，是网络拥塞控制最重要的手段。如何实现AQM高级策略，引入新的人工智能算法和遗传算法与模糊逻辑的综合应用是目前研究的一个热点问题。

3) 以往的工作主要采用局部线性化方法，缺乏对系统全局动力学的理论分析。此外，在多种源端拥塞控制策略和路由器避免策略并存时，如何分析整个网络的稳定性，如何分析各种不确定因素对稳定性的影响等，也是需要认真考虑的问题。

4) TCP/IP 拥塞控制的设计和实现面临着众多的折中，不可能有一种设计和实现在所有环境中都是“最好的”。现有的拥塞控制思路、方法和技术在多目标的不同环境中面临着挑战，它们还有许多要改进的地方。

5) 目前已经有越来越多的移动用户通过无线系统接入互联网，由于无线通信固有的特点，使得拥塞控制机制的研究更加困难，极具挑战。

拥塞控制机制就是防止过多的数据注入到网络中，这样可以使网络中的路由器或链路不致过载。拥塞控制所要做的都有一个前提：网络能够承受现有的网络负荷。拥塞控制是一个全局性的过程，涉及到所有的主机、路由器，以及与降低网络传输性能有关的所有因素。流量控制是指点对点通信量的控制，是端到端正的问题。流量控制所要做的就是抑制发送端发送数据的速率，以便使接收端来得及接收。

拥塞控制需要获得网络内部流量分布的信息。在实施拥塞控制之前，还需要在结点之间交换信息和各种命令，以便选择控制的策略和实施控制。这样就产生了额外的开销。拥塞控制还需要将一些资源分配给各个用户单独使用，使得网络资源不能更好地实现共享。

在传统的TCP网络中有几种拥塞控制方法：慢开始( slow-start )、拥塞避免( congestion avoidance )、快重传( fast retransmit )和快恢复( fast recovery )。

1 慢开始和拥塞避免

发送方维持一个拥塞窗口 cwnd ( congestion window )的状态变量。拥塞窗口的大小取决于网络的拥塞程度，并且动态地在变化。发送方让自己的发送窗口等于拥塞。发送方控制拥塞窗口的原则是：只要网络没有出现拥塞，拥塞窗口就再增大一些，以便把更多的分组发送出去。但只要网络出现拥塞，拥塞窗口就减小一些，以减少注入到网络中的分组数。

慢开始算法：当主机开始发送数据时，如果立即所大量数据字节注入到网络，那么就有可能引起网络拥塞，因为现在并不清楚网络的负荷情况。因此，较好的方法是先探测一下，即由小到大逐渐增大发送窗口，也就是说，由小到大逐渐增大拥塞窗口数值。通常在刚刚开始发送报文段时，先把拥塞窗口 cwnd 设置为一个最大报文段MSS的数值。而在每收到一个对新的报文段的确认后，把拥塞窗口增加至多一个MSS的数值。用这样的方法逐步增大发送方的拥塞窗口 cwnd ，可以使分组注入到网络的速率更加合理。

每经过一个传输轮次，拥塞窗口 cwnd 就加倍。一个传输轮次所经历的时间其实就是往返时间RTT。不过“传输轮次”更加强调：把拥塞窗口cwnd所允许发送的报文段都连续发送出去，并收到了对已发送的最后一个字节的确认。

另，慢开始的“慢”并不是指cwnd的增长速率慢，而是指在TCP开始发送报文段时先设置cwnd=1，使得发送方在开始时只发送一个报文段（目的是试探一下网络的拥塞情况），然后再逐渐增大cwnd。

为了防止拥塞窗口cwnd增长过大引起网络拥塞，还需要设置一个慢开始门限ssthresh状态变量（如何设置ssthresh）。慢开始门限ssthresh的用法如下：

当 cwnd < ssthresh 时，使用上述的慢开始算法。

当 cwnd > ssthresh 时，停止使用慢开始算法而改用拥塞避免算法。

当 cwnd = ssthresh 时，既可使用慢开始算法，也可使用拥塞控制避免算法。

拥塞避免算法：让拥塞窗口cwnd缓慢地增大，即每经过一个往返时间RTT就把发送方的拥塞窗口cwnd加1，而不是加倍。这样拥塞窗口cwnd按线性规律缓慢增长，比慢开始算法的拥塞窗口增长速率缓慢得多。

无论在慢开始阶段还是在拥塞避免阶段，只要发送方判断网络出现拥塞（其根据就是没有收到确认），就要把慢开始门限ssthresh设置为出现拥塞时的发送方窗口值的一半（但不能小于2）。然后把拥塞窗口cwnd重新设置为1，执行慢开始算法。这样做的目的就是要迅速减少主机发送到网络中的分组数，使得发生拥塞的路由器有足够时间把队列中积压的分组处理完毕。如下图2-6，拥塞控制的过程如下：

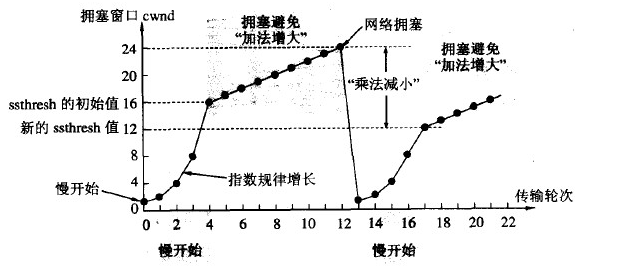


图2-6 拥塞控制过程

当TCP连接进行初始化时，把拥塞窗口cwnd置为1。前面已说过，为了便于理解，图中的窗口单位不使用字节而使用报文段的个数。慢开始门限的初始值设置为16个报文段，即 cwnd = 16 。

在执行慢开始算法时，拥塞窗口 cwnd 的初始值为1。以后发送方每收到一个对新报文段的确认ACK，就把拥塞窗口值另1，然后开始下一轮的传输（图中横坐标为传输轮次）。因此拥塞窗口cwnd随着传输轮次按指数规律增长。当拥塞窗口cwnd增长到慢开始门限值ssthresh时（即当cwnd=16时），就改为执行拥塞控制算法，拥塞窗口按线性规律增长。

假定拥塞窗口的数值增长到24时，网络出现超时（这很可能就是网络发生拥塞了）。更新后的ssthresh值变为12（即变为出现超时时的拥塞窗口数值24的一半），拥塞窗口再重新设置为1，并执行慢开始算法。当cwnd=ssthresh=12时改为执行拥塞避免算法，拥塞窗口按线性规律增长，每经过一个往返时间增加一个MSS的大小。

强调：“拥塞避免”并非指完全能够避免了拥塞。利用以上的措施要完全避免网络拥塞还是不可能的。“拥塞避免”是说在拥塞避免阶段将拥塞窗口控制为按线性规律增长，使网络比较不容易出现拥塞。

快重传和快恢复：如果发送方设置的超时计时器时限已到但还没有收到确认，那么很可能是网络出现了拥塞，致使报文段在网络中的某处被丢弃。这时，TCP马上把拥塞窗口 cwnd 减小到1，并执行慢开始算法，同时把慢开始门限值ssthresh减半。这是不使用快重传的情况。

快重传算法首先要求接收方每收到一个失序的报文段后就立即发出重复确认（为的是使发送方及早知道有报文段没有到达对方）而不要等到自己发送数据时才进行捎带确认。

接收方收到了M1和M2后都分别发出了确认。现在假定接收方没有收到M3但接着收到了M4。显然，接收方不能确认M4，因为M4是收到的失序报文段。根据可靠传输原理，接收方可以什么都不做，也可以在适当时机发送一次对M2的确认。但按照快重传算法的规定，接收方应及时发送对M2的重复确认，这样做可以让发送方及早知道报文段M3没有到达接收方。发送方接着发送了M5和M6。接收方收到这两个报文后，也还要再次发出对M2的重复确认。这样，发送方共收到了接收方的四个对M2的确认，其中后三个都是重复确认。快重传算法还规定，发送方只要一连收到三个重复确认就应当立即重传对方尚未收到的报文段M3，而不必继续等待M3设置的重传计时器到期。由于发送方尽早重传未被确认的报文段，因此采用快重传后可以使整个网络吞吐量提高约20%。

与快重传配合使用的还有快恢复算法，其过程有以下两个要点：

1.当发送方连续收到三个重复确认，就执行“乘法减小”算法，把慢开始门限ssthresh减半。这是为了预防网络发生拥塞。请注意：接下去不执行慢开始算法。

2.由于发送方现在认为网络很可能没有发生拥塞，因此与慢开始不同之处是现在不执行慢开始算法（即拥塞窗口cwnd现在不设置为1），而是把cwnd值设置为慢开始门限ssthresh减半后的数值，然后开始执行拥塞避免算法（“加法增大”），使拥塞窗口缓慢地线性增大。

下图2-7给出了快重传和快恢复的示意图，并标明了“TCP Reno版本”。

区别：新的 TCP Reno 版本在快重传之后采用快恢复算法而不是采用慢开始算法。

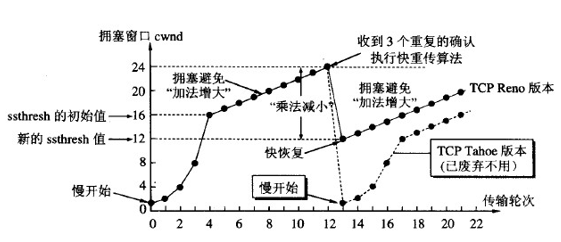


图2-7 快重传和快恢复示意图

有的快重传实现是把开始时的拥塞窗口cwnd值再增大一点，即等于 ssthresh + 3 X MSS 。这样做的理由是：既然发送方收到三个重复的确认，就表明有三个分组已经离开了网络。这三个分组不再消耗网络 的资源而是停留在接收方的缓存中。可见现在网络中并不是堆积了分组而是减少了三个分组。因此可以适当把拥塞窗口扩大了些。

在采用快恢复算法时，慢开始算法只是在TCP连接建立时和网络出现超时时才使用。采用这样的拥塞控制方法使得TCP的性能有明显的改进。

接收方根据自己的接收能力设定了接收窗口rwnd，并把这个窗口值写入TCP首部中的窗口字段，传送给发送方。因此，接收窗口又称为通知窗口。因此，从接收方对发送方的流量控制的角度考虑，发送方的发送窗口一定不能超过对方给出的接收窗口rwnd。发送方窗口的上限值 = Min [ rwnd, cwnd ]

当rwnd < cwnd 时，是接收方的接收能力限制发送方窗口的最大值。

当cwnd < rwnd 时，则是网络的拥塞限制发送方窗口的最大值。

（1）多条流入线路有分组到达，并需要同一输出线路，此时，如果路由器没有足够的内存来存放所有这些分组，那么有的分组就会丢失。

（2）路由器的慢带处理器的缘故，以至于难以完成必要的处理工作，如缓冲区排队、更新路由表等。

防止拥塞的方法

（1）在传输层可采用：重传策略、乱序缓存策略、确认策略、流控制策略和确定超时策略。

（2）在网络层可采用：子网内部的虚电路与数据报策略、分组排队和服务策略、分组丢弃策略、路由算法和分组生存管理。

（3）在数据链路层可采用：重传策略、乱序缓存策略、确认策略和流控制策略。

NDN 网络的拥塞控制主要存在以下问题和挑战：(1) 传统 TCP 的隐式 (Implicit) 拥塞检测机制在NDN 中不再适用。NDN 中由于网内缓存的存在，数据是多源的，TCP 基于单一源的 RTO (重传计时器)超时机制在 NDN 中不再可靠；(2) AIMD 算法会引发 RTT 公平性问题。在NDN中盲目使用传统 TCP 协议的 AIMD 拥塞控制算法可能引发公平性问题。使用 AIMD 算法，吞吐率反比于 RTT，在NDN 网络中，由于有网内缓存，流行的内容 Cached离消费者更近，有更小的延迟，这会导致链路的大部分带宽被流行的内容占据，从而引发对不流行内容的公平性问题。因此，需要重新对 NDN 网络中拥塞控制算法进行研究，目前对于 NDN 网络拥塞控制算法的研究尚处于起步阶段。

### 2.2.2 转发策略

在NDN中，当路由器接收到一个兴趣包，首先检查在ContentStore中是否有匹配的数据。如果有匹配的数据，这些匹配的数据将被按着兴趣包的进入接口转发出去；如果没有匹配的数据，检索PIT，如果兴趣包名字已经在PIT中存在，意味着有一个相同名字的兴趣包来自其他的请求者被接收并且已经被转发出去，这样对这个新到来的兴趣包，路由器将增加一个incominginterface到已经存在的PIT条目中；如果在PIT条目中没有相同的兴趣包的前缀，则检查FIB，若有记录这个兴趣包的条目，则根据转发接口将兴趣包转发出去，并将该兴趣包和转发出去的接口一起作为一个新条目加入PIT中；若没有则丢弃该兴趣包。当一个数据包到来的时候，先用它的名字查找PIT，如果有匹配的PIT条目，路由器按照条目中记录的兴趣包的请求接口，即进入接口，把数据包转发出去，并把它缓存在ContentStore，移除PIT中相应的条目。如果PIT中没有匹配的条目，这个数据包就被抛弃。每一个兴趣包也有一个相应的生存时间；当生存时间逾期的时候PIT会移除相应的条目。

由于NDN的采用层次化的命名结构，其转发策略拥有很大的灵活性和可扩展性，目前NDN主要采用三种转发策略，分别为洪泛转发策略(Flooding),智能洪泛转发策略(Smart Flooding)和最优转发策略(Best Route)。在介绍NDN的智能数据平面时己经指出每个PIT会维护一个接口状态信息，该信息对于实现和优化NDN的转发策略具有实际的意义。目前NDN节点接口的状态分为正常、就绪、故障三种，在基于NS-3模块实现NDN通信模型的仿真平台ndnSIM上采用“GREEN” 、“YELLOW” 、“RED”来标示三者。在NDN体系架构上，节点间是通过接口进行通信的，并且维护接口状态信息。接口状态初始为就绪且随着网络的演进动态更新，例如，一个处于YELLOW的接口一旦获得缓存资源或者得到返回的数据包将会变成GREEN，而一个GREEN接口在长时间不使用的情况下会转变为YELLOW状态，而一个YELLOW接口如果经过探测显示可以正常工作，则将状态更新为GREEN 。 NDN的三种转发策略的实现是基于接口状态的。

在Flooding策略中，路由节点将兴趣包转发给所有在FIB表中名字前缀匹配成功且状态不为RED的接口。由于该策略能快速发现转发路径，可以获得较小的平均时延，但由于发送大量兴趣包，不仅增加了网络流量而且带来大量的网络开销，在访问量大、带宽有限的情况下容易造成拥塞现象。 Best Route策略通过路由节点将兴趣包转发给FIB表中名字前缀匹配并且排序最前的(Highest-Rank)GREEN接口或者排序最前的YELLOW接口，忽略所有的RED接口。其中排序规则是根据以往的路由性能指标，从小到大排序。使用Best Route策略由于发送的兴趣包较少，而且转发的接口都是当前路由性能最好的接口，能因此可以有效避免网络中冗余流量的产生，但不足之处在于排序指标是一种基于以往经验的指标，状态更新滞后，重传次数明显增加。

在Smart Flooding策略中，路由节点优先考虑将兴趣包转发给排序最前的GREEN接口，若不存在GREEN接口则将兴趣包转发给所有的YELLOW接口。忽略所有RED接口。Smart Flooding的性能介于Best Route与Flooding之间。由于网络带宽资源有限而内容传输数据量庞大的实际运行状况，Best Route虽降低网络冗余流量，只向排序最前的GREEN或YELLO W接口发送请求，其结果相对Flooding策略，有较高时延和较多的请求重传次数。

NDN目前的缺省的转发策略是单路径的，中间节点把跳数作为路径选取的指标，选择条数最少的路径作为最优路径，这种转发策略被称为BestRoute。BestRoute是一种类TCP的转发策略，并不适用于NDN网络，理由如下：首先BestRoute只考虑路径的跳数，条数是静态的，不能反映当前的链路状态。如果大量的数据包都沿着静态最优的路径转发，那么会导致该路径拥塞而其他路径空闲，形成了资源浪费；NDN是无连接的，发送端和接收端不建立连接，因此数据可以沿多条路径传输。多路径转发策略是NDN中一个重要的研究方向。

### 2.2.3 路由机制

NDN的路由机制旨在通过利用分布在网络上的缓存副本提高检索内容文件的效率，基于此，NDN应该提供一种整合大范围分布式副本进入检索过程的机制。但是，网络上缓存副本的数量要远远多于副本的原始文件数量，甚至达到其数量的几十倍，因此将数量众多的副本作用于检索过程中时，检索将成为一个非常具有挑战的问题。而且前文己经不止一次提到NDN的路由是基于命名数据的，因此有效的名称解析在传统的NDN路由机制中占有举足轻重的作用。 NDN的这种路由策略可以说是IP路由的优势继承，该策略不仅做到了对当前IP网络的兼容，更提供了很好的演化策略，以便于基于IP的路由到NDN的演变，因此更加便于网络部署的展开，在尽量满足NDN的前提下做到对IP路由的兼容和演化。传统的路由协议，例如BGP, OSPF, IS-IS都能够很好地迁移到NDN网络上来。

由以上的描述可知，在NDN的路由过程中用户只需要从就近的节点直接获取所需的数据即可，该机制可以增加网络的传输效率并减少网络建设成本。同时NDN的层次化命名机制很好的解决当前IP网络中存在的地址空间耗尽、NAT穿越、移动性和可扩展的地址管理四个问题。在NDN架构中不存在地址耗尽的问题，因为基于名称的命名空间是无尽的。更不存在NAT穿越问题，因为在NDN中，名称对于路由器而言是不可见的，也就是说主机不需要暴露出自己的地址信息以达到获取内容的目的。如果将IP路由协议的前缀信息用命名数据的名称替换，那么IP路由协议如：BGP，IS-IS和OSPF都可在NDN网络中继续使用。

虽然NDN的路由机制相较于IP路由表现出了巨大优势，但它的不足之处也是显而易见的。基于名称解析的路由机制需要名字解析系统维护一个备份列表，受限于路由节点中缓存区容量，一些旧的备份经常会被新的备份替换，这必将导致备份的本地信息频繁更新，基于此原因，名称解析系统维护所有副本的列表和所有备份的本地信息成为一个无法有效解决的问题。

尽管基于命名的路由技术，简化了路由的过程减少了网络时延，但是，也正是由于这个原因该路由机制存在着可扩展上的不足，因为路由表中的条目数量与网路中的内容文件数量成正比的，虽然聚合技术减少了失败的可能，但是随着网络中备份数据的分布越来越广聚合度也随之减小，因此NDN的路由机制成为本文研究的重点。

### 2.2.4 缓存策略

由于NDN节点中CS的存在，使NDN与基于TCP/IP的传统网络架构相比，在用户的数据请求体验上有了很大提高。但NDN中默认的数据存储策略将数据存储在所有经过的节点中，这将不利于存储资源的高效利用，从而影响NDN的网络性能。为了高效地利用NDN的存储资源，NDN存储策略必须控制同一数据在网内节点中的存储数量，即数据只能存储在某一个或几个节点中。因此，将数据存储在哪些节点中才能使NDN性能最好，便成为了研究者们的研究重点。

NDN数据存储策略的核心思想就是，在数据经过的链路上或网内其他节点中寻找某个节点或某几个节点用于存储数据，为用户的后续请求提供高效的服务，使网络性能得到提升。然而，由于节点在网内所处的位置不同，在功能上往往会有很大的区别。在NDN中依据节点与客户端的距离可以将节点划分为“近客户端”节点与“中心”节点两类。因此，众多研究者分别提出了“近客户端”数据存储策略与“中心”数据存储策略。从字面意思可以看出，“近客户端”数据存储策略是将流行的数据靠近客户端存储，这种存储策略，由于数据存储的节点位置离客户端节点很近，客户端可以迅速地请求到需要的数据，从而能够很好地提升用户对数据请求的体验。但是，“近客户端”节点所服务的对象往往只针对某一个或某几个客户端节点，这将导致这类节点无法对其他用户及时地提供服务。数据存储策略就是典型的“近客户端”存储策略。在经过少数的几次数据请求后，数据就会被存储在离客户端很近的节点上。

## 2.3 本章小结

本章首先对NDN的基础架构进行了介绍，包括NDN的协议栈、包格式、节点模型和转发流程。在上述架构的基础上，对NDN的优缺点进行了分析。为了提高NDN的性能，本文需要对NDN的传输控制机制进行深入研究，本章对NDN传输机制中的拥塞控制、转发策略、路由机制和缓存机制进行了介绍。最后对本章进行了总结。

# 第三章 基于显式标记的传输控制模型

在上一章中，本文主要针对NDN中相关概念及关键技术进行了研究。本章将对NDN传输控制机制及设计。本章结构内容如下：（1）介绍基于显式标记的拥塞检测，（2）根据拥塞检测，给出了拥塞控制设计，拥塞控制主要包含发送端速率调整以及多路径转发策略，（3）针对网络场景，对缓存策略进行了设计，（4）最后对本章进行了总结。

## 3.1 拥塞检测

传统的TCP拥塞控制是基于连接的，发送端和接收端在数据传输之前先建立连接。发送端通过超时或丢包的方式检测拥塞，超时是指数据包发送超过一定的时间后（Retransmission Timeout，RTO）还没有被接受端确认，丢包是指发送端接收到三个相同的ACK，通过这两种方式可以检测出链路发生了拥塞，然后发送端对发送速率进行调整。

上述拥塞检测方式不适用NDN，首先NDN中没有对数据包的确认，不存在ACK，无法利用三个相同ACK判断丢包；其次，NDN是无连接的，数据传输是发送端驱动的，数据包可能从不同源节点沿不同节点获取，这就导致不同数据包的往返时延（Round Trip Time，RTT）波动比较大，无法统计出正确的RTO，一些RTT大的数据包会被误认为超时。

在NDN中，每个中间节点都维护着两个队列，一个是Interest发送队列，一个是Data发送队列。队列的作用是应对突发流量，当网络中流量爆发式增长的时候，中间节点起到缓存作用，把多的Interest包和Data包缓存在两个队列当中。队列容量的设置对于网络性能有很大的影响，当队列容量比较大的时候，缓存在队列处的数据包会长时间处于排队状态，数据包的RTT会变大，可能会导致“假重传”。因此队列容量的设置不能太大。当数据包的数量大于缓存队列的容量时，多余的数据包会被丢弃，这种丢包无法被系统主动检测，只能被动地等待超时来确认丢包。这种方式下，系统对网络拥塞的检测和调整都存在滞后的现象，无法及时对拥塞进行控制，网络性能下降后无法快速回复，带宽的利用率降低。

本文提出一种通过监测中间节点出口缓存队列的方式来检测拥塞。拥塞检测机制同时监测中间节点Interest队列和Data队列，监测是一种周期性的检测，每隔一个周期T对队列进行一次监测，每次检测的瞬间结果Qi都会被记录下来，每个8个周期计算一次平均队列长度Qavg，其中Qavg的计算公式如下：

 (1)

其中Qavg表示在过去的8个周期内平均的队列长度，用这个量来估计在过去一段时间内Interest和Data队列的长度。为了更好的对队列长度进行估计，在上述计算方式的基础上，本文提出了一种改进平均队列计算方法。

在改进的平均队列计算方法中，每个检测出的 Interest或Data 队列长度会有一个对应的权重值，并且权重在连续8个周期内是单调递增的。假设 Qi和Qj 是两个周期内检测出的瞬时 Interest 队列的长度 (i<j)，则Q i的权重值 W i 小于Q j的权重值Wj。这种递增的权重设计说明新检测出的 Interest 队列长度比旧检测出的 Interest 队列长度更重要，更能体现当前网络的拥塞程度。为了使权重能够平滑的增长，算法使用了线性增长的方式，Wi的计算公式如下：

 (2)

判断 Interest 队列长度的变化趋势，对于一个周期内检测出的瞬时 Interest 队列长度，比较 Q 1和 Q i 两个值的大小，如果 Q 1 大于 Qi，说明这个周期内传输队列长度整体的趋势是减小的。相反，如果 Q 1 小于 Q i ，说明这个周期内传输队列长度整体的趋势是增大的。通过该趋势，能够判断队列长度的在下一段时间内的变化情况，改进的计算方法增加了一个能反应Interest 队列长度变化趋势的优化参数ΔQ，计算公式如下：

ΔQ= Qi-Q1 （3）

经过以上优化后，最终的平均 Interest 队列长度Q fina是Qavg与ΔQ之和，求出 Q final 之后，把它作为判断网络拥塞程度的指标。

当Qfinal的值为空的时候，此时没有数据包在队列中，没有数据包处于排队状态，把链路标记为FREE状态，当Qfinal的值不为空，但是队列的不超过队列长度的85%的时候说明有数据包正在排队的，此时应该把链路标记为BUSY状态当Qfinal的值超过队列的85%认为链路可能发生拥塞，此时把该链路标记为CON状态，此时应该进行拥塞控制。

## 3.2 拥塞通告

拥塞通告是一种反馈链路拥塞状况的技术，IP和TCP使用包头中的未使用字段来支持拥塞通告。

　　在网络层(IP)，一个发送主机必须能够表明自身可以进行显式拥塞通告（Explicit Congestion Notification）ECN，路由器在转发时必须能够表明它正在经历拥塞。

　　在传输层(TCP)，TCP端必须对对方表明自身是可以进行ECN操作的。接收端必须能够通知发送端它收到了一个来自路由器的拥塞通告。发送端必须能够通知接收端它受到了来自接收端的通告并且已经降低了发送速率。

    IP包头中的8位的服务类型域(TOS)原先在RFC791中被定义为表明包的发送优先级，时延，吞吐量，可靠性和消耗等特征。在RFC2474中被重新定义为包含一个6位的区分服务码点(DSCP)和两个未用的位。DSCP值表明一个在路由器上配置的和队列相关联的发送优先级。IP对ECN的支持使用到了TOS域中剩下的这两位。如图1所示。

　　在RFC2474中TOS域未使用的两位在RFC3168中被定义为ECN域，包含如下值：

　　 00：发送主机不支持ECN

　　 01或者10：发送主机支持ECN

　　 11：路由器正在经历拥塞

一个支持ECN的主机发送数据包时将ECN设置为01或者10。对于支持ECN的主机发送的包，如果路径上的路由器支持ECN并且经历拥塞，它将ECN域设置为11。如果该数值已经被设置为11，那么下游路径上的路由器不会修改该值。

当一个IP包的ECN域被路由器设置为11时，接收端而非发送端被通知路径上发生了拥塞。ECN使用TCP头部来告知发送端网络正在经历拥塞，并且告知接收端发送段已经受到了接收端发来的拥塞通告，已经降低了发送速率。

TCP对ECN的支持使用TCP中预先定义的保留位。ECN定义两个新的标志，如图2所示：

ECE：ECN响应标志被用来在TCP3次握手时表明一个TCP端是具备ECN功能的，并且表明接收到的TCP包的IP头部的ECN被设置为11。更多信息请参考RFC793。

CWR：拥塞窗口减少标志被发送主机设置，用来表明它接收到了设置ECE标志的TCP包。拥塞窗口是被TCP维护的一个内部变量，用来管理发送窗口大小。

当两个支持ECN的TCP端进行TCP连接时，它们交换SYN，SYN-ACK和ACK包。对于支持ECN的TCP端来说，SYN包的ECE和CWR标志都被设置了。SYN-ACK只设置ECE标志。

　　一个支持ECN的TCP主机在支持ECN的TCP连接上发送设置了IP头部为10或者01的TCP包。支持ECN的路由器在经历拥塞时设置IP头部的ECN域为11。当一个TCP接收端发送针对收到的一个设置ECN位为11的TCP包的响应时，它设置TCP包头中的ECE，并且在接下来的ACK中也做同样设置。

当发送主机接收到设置了ECE标志的ACK时，它就能感知到包丢失，开始减少发送窗口，运行慢启动过程和拥塞避免算法。在下一个数据包中，发送者设置CWR标志。在接收到新的设置CWR标志的包时，接受者停止在接下来的ACK中设置ECE标志。

在上述技术的背景下，针对NDN的网络架构，本文提出了一种基于显示标记的拥塞通告。拥塞通告分为两种，对应上节中提到BUSY和CON两种状态。在NDN中间节点探测拥塞链路的过程中，如果某条链路的Interest队列或者Data队列超过队列长度的85%，那么就认为该链路发生拥塞吗。此时节点沿着链路向下游也就是向数据消费端（consumer）发送被标记的数据包（Marked），被标记的数据包是在原有数据包的基础上封装了一个新的“Marked”字段，如果字段是1，这表示该数据包是一个被标记忙碌的数据包，说明链路处于完全利用阶段，该数据包表明链路的利用程度较高。如果字段为2，这表示该数据包是一个被标记拥塞的数据包，用来指示该链路已经发生了拥塞。

Marked包的效果是提醒下游节点对发生拥塞的链路进行调整，这个调整效果要不断减弱，因为过度的调整会导致发送端速率变小，从而导致带宽不能完全利用。针对上述问题，在设计当中，Marked数据包的发送是有一定时间间隔的，第一个Marked数据包和第二个Marked数据包的发送间隔为100ms，之后的每两个数据包之间的发送间隔以线性方式不断递增，这样做是为了不断减弱对拥塞链路的调整效果，直至链路恢复正常。

Marked包从发生拥塞的链路向下游的所有节点转发，接收到Marked包的节点都会进行速率调整，减小发送速率。这种设计是合理的，因为任何下游节点发送Interest包都可能导致上游节点拥塞，所以拥塞节点应该向所有可能导致拥塞的下游节点发送Marked包。在NDN中，一个PIT表项可以是聚合的，也就是多个消费者端请求同一个数据，如果标记数据包只向其中一个消费者端发送那么其余的消费者端依然会向该节点发送兴趣包，拥塞的链路依然没有办法恢复。因此，向下游所有的节点发送Marked包是合理的。

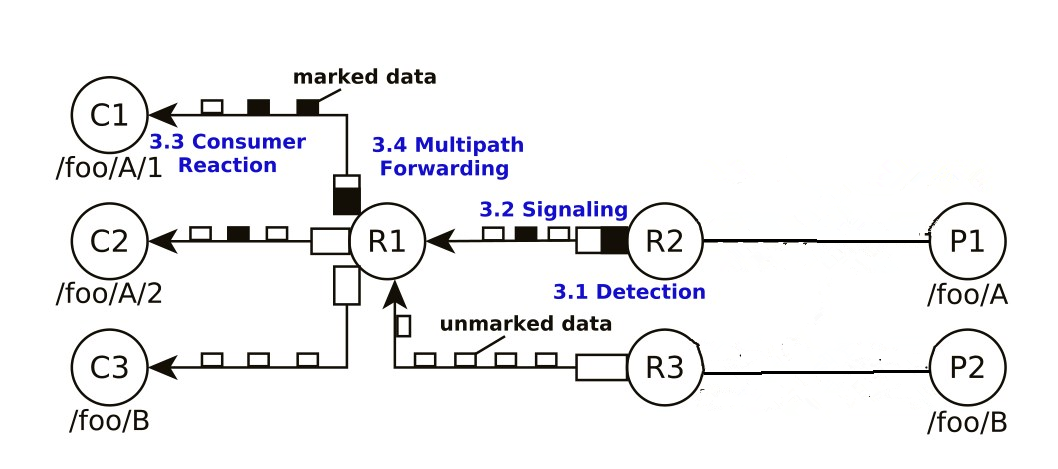


图3-1 拥塞通告示意图

如图3-1所示，消费者端C3从生产者端P2处获取数据，经过的路径C3R1、R1R3、R3P2都没有发生拥塞，所以P3收到的数据包都是未标记的数据包，C3也不进行任何调整。C2从P1处获取数据，R1R2链路发生了拥塞，所以C2收到了来自R2的标记数据包，C2进行速率调整。C1从P1处获取数据，C1R1、R1R2两条链路都发生了拥塞，所以C1既收到R1的标记数据包也收到R2的标记数据包，C2进行速率调整的幅度最大。

## 3.3 发送端速率调整

当发送端收到显示标记的数据包时候具体来说，当收到未标记的数据包的时候，采用乘性增加 (Multiplicative Increase， MI) 的算法，可以尽快占据网络空闲的带宽；收到标记忙碌状态的数据包的时候，采用和性增加 (Additive Increase，AI) 的算法，能够平滑的增大拥塞窗口，尽可能避免出现拥塞的现象；收到拥塞标记的数据包的时候采用乘性减小 (MultiplicativeDecrease，MD) 的算法，重新调整拥塞窗口的大小。具体的算法如下：

MI: cwnd(t+rtt)=cwnd(t)×(1+ξ)

AI: cwnd(t+rtt)=cwnd(t)+α

MD: cwnd(t+θt)=cwnd(t)×β

以上算法及参数需通过仿真实验对算法进行改进

设计及确定合适的参数值。算法的伪代码如下：

①cwnd 表示拥塞窗口 (初始值为 1) ，adjust 用来

判断采用 MI 算法还是 AI 算法来增加 cwnd 的值。

②处理 Nack 的函数：

Begin

while (true) do

If (received Marked==UnMarked)

adjust = true;

else if (received Marked == BUSY)

adjust = false;

else

cwnd = cwnd \*β; //MD算法

End if

End while

③处理 Data 的函数：

Begin

while (true) do

If (adjust == true)

cwnd = cwnd + ζ; //MI 算法

else

cwnd = cwnd + α/cwnd; //AI 算法

End if

End while

上述算法在链路空闲状态发送端速率快速增长，尽快完全利用链路；在链路完全利用处于忙碌状态时，放慢增长速率；在链路拥塞的情况下，采用乘法娇小速率的方式对发送端速率调整，保证链路恢复到正常状态。

拥塞检测、拥塞通告和发送端速率调整三种机制结合，及时对发送队列监控，向下游发送拥塞通告，做出发送端速率调整，解决了由于发送端速率过大，链路中数据包超过缓存队列长度而引起的丢包问题。链路中发生的丢包只可能是偶然因素引发丢包，这类丢包没法检测出来，通过超时而引发发送端速率调整，对于超时发送端发送速率减半。由于偶然性丢包不是连续丢包，发送端速率只会进行几次速率减半处理，不会出现过度调整。

综上，基于显式标记的传输控制机制解决了命名数据网络中丢包问题，提高了传输性能。

## 3.4 多路径转发策略

由于NDN中通信的两端不建立连接，数据请求端只关注数据的获取，不关心数据沿哪条路径获取，数据可以从多个数据源获取。因此在NDN中数据可以沿着多跳路径传输。数据的多路径传输充分利用了网络带宽，提高了网络性能，当某条链路出错时，数据还能沿着其余链路传输，对于网络业务的容错性有了进一步提升。因此，多路径转发对与提升NDN高效性和稳定性有着至关重要的作用。

上一节已经论述到在基于显式标记的传输机制中，不会发生丢包。上面的所有研究都是基于单路径的，当一条路径发生拥塞时，对发送端速率进行调整，从而保证了传输稳定。在上述机制的基础上，本文新加入了多路径转发策略。

多路径转发策略的设计目标就是传输速率最大化、带宽利用率最大化和带宽使用代价最小化。带宽利用率最大化要保证所有路径都被利用，不能有网络链路处于无人使用的状态；传输效率最大化就是数据请求端的接收速率达到最大，在网络性能指标当中接收速率是数据获取速率，这个指标表明了网络的实际性能。发送速率只是发送端控制的速率，发送的数据包没有全部处理，有可能缓存在中间路由器的队列当中，因此不把发送端速率当作是网络的性能指标。网络代价最小化指数据包的传输尽可能沿比较短的链路，保证网络使用代价最小。多路径转发是从全局上对网络性能的提升，除了对性能的提升要考虑网络带宽利用的公平性问题。除此之外，多路径转发策略会导致数据包乱序到达，这样会对终端的业务造成影响，因此多路径转发策略是一个权衡利弊的过程。

在多路径转发过程中，由于数据包沿着不同的路径转发，数据包的RTT差异会比较大，RTT大的数据包会被认为已经超时，实际上这一部分数据包还在传输过程当中，这就是一种“假重传”，因此多路径转发策略当中需要把RTO设置到足够保证最大的RTT不会被认定为重传。对于数据包乱序问题，数据的接收端要维护一个接收缓存区，先把乱序的数据包进行重组处理，再进行交付。这个缓存区设置要合理，太大会引起处理时延变长，对网络性能造成影响，太短无法处理过多的乱序数据包。因此缓存区的设置要未来的工作中还有待研究。

本文基于提出了一种多路径转发策略，需要拥塞检测、拥塞通告和发送端速率协同工作。该多路径转发策略控制中间所有节点，每个中间节点在数据包转发的过程中首先选择可用链路中距离最短（路由器条数最少）的链路，如果距离最短的链路检测到了拥塞状态，则把一半流量向距离次短的链路转移。如果两条链路依然检测到拥塞，则继续向距离第三和第四短的链路进行分流，直至没有拥塞的出现。如果在一个段时间内某条链路没有发生检测到拥塞信号，那么当前正在启用的距离最长链路上的一半流量转移到该链路，如此反复调整流量分流比例，直至流量分配比例收敛到最优的状态。

下面是一个多路径转发策略的流量转移的过程，R首先只向R1转发数据包，R1上的流量比例是100%，随着流量的增加RR1链路的带宽无法满足需求，发生了拥塞，R收到来自R1的标记数据包（如图3-2-1），这时RR1上一半的流量向RR2转移（如图3-2-2），如果此时RR2发生了拥塞，R收到来自R2的标记数据包，流量向RR3转移（如图3-2-3）。此时经过一段时间后，如果优先级靠前的链路没有发生拥塞，那么优先级最低的链路向该链路分流，如果分流后的没有发生拥塞，这说明分流是可行的，如果分流后发生了拥塞，则说明分流是不可行的，“撤消”该操作，直至所有的链路都没有检测到拥塞（如图3-2-4），此时已经达到最优状态。

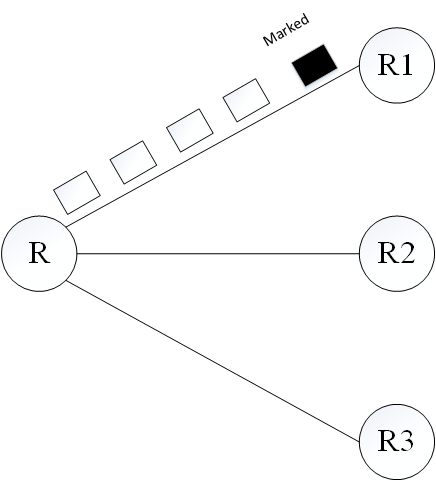


图3-2-1

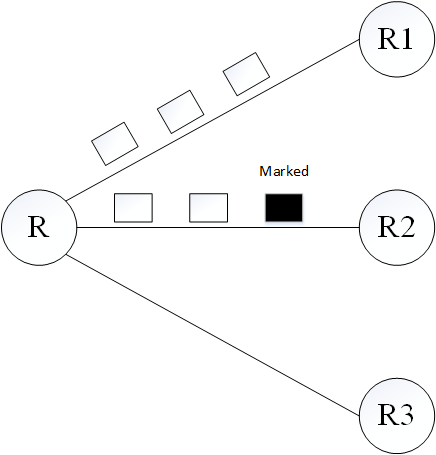


图 3-2-2

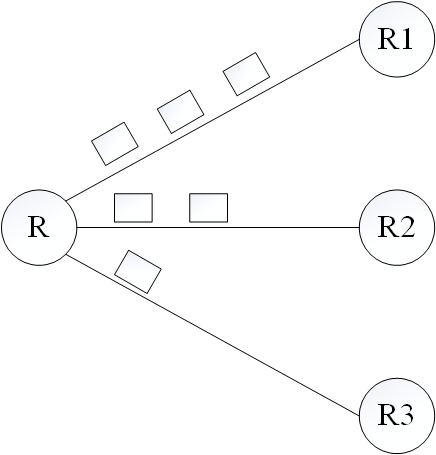


图3-2-3

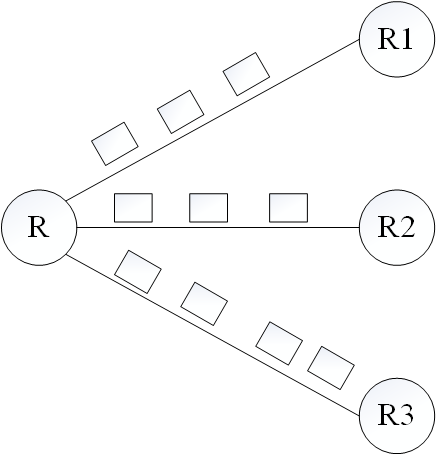


图3-2-4

多路径转发策略选路优先级是路径长度，而没有考虑其他指标，这样做的目的是为了降低复杂度，网络带宽和时延的测量比较困难，选择比较容易计算出来条数最为选路指标，带宽的影响会在后面流量动态调整的过程体现出来。

多路径转发是个逐跳动态调整的过程，每个节点通过显式标记的数据包对链路状态智能感知，不断调整流量分配比例直至收敛到一个接近最优值，保证了所有链路带宽的利用。

## 3.5 缓存策略

在NDN中不关注主机所在的位置，而是以数据本身为中心。最重要的是，在NDN中每个路由器节点不仅仅路由数据，同时具有CS结构，用于存储经过节点的数据。因此，NDN数据缓存策略的优劣对NDN的性能有较大的影响。NDN的数据缓存策略又包括数据存储策略与数据替换策略，其中，数据存储策略主要是用于在多个节点中选择最有利于网络性能的节点位置存储数据;数据替换策略主要用于在缓存空间耗尽时，替换CS中“过时”的数据，使节点CS存储更能满足当前用户需求的数据。因此，选择什么样的节点以及替换哪些数据便成为了NDN数据缓存策略的研究重点。本章将对现有NDN的数据缓存策略进行分析和介绍。

通过以上描述可知，在NDN中每个节点都有CS结构，该结构可用于长时间存储经过节点的数据，用以满足后续相同请求，从而减少用户请求数据的距离与时延，提高用户对数据请求的体验。然而，NDN中节点的数据存储空间与不断增长的数据相比显得十分有限，怎样高效利用有限的存储空间，尽可能地提高网络性能便成为了目前NDN研究的重点之一。目前对NDN数据存储的研究，主要有以下几个目的。

1.提高网络服务质量:从网络诞生之日起，这便成为了互联网研究中的重点，同时也是运营商一直以来的奋斗方向。与传统网络不同，NDN中用户对数据的请求可以被存储了对应数据的任意网内节点满足，不一定需要去服务器请求数据。因此，为了使用户能够及时地得到所请求的数据，一种高效的数据存储策略便显得尤为重要。本文依据用户对数据的请求情况与网络的特性提出了相应的数据存储策略，在定点地满足个别用户的同时，也为其他潜在用户提供服务。

2.优化网络性能:由于网络资源十分有限，作为网络管理者，利用有限的网络资源使网络性能尽可能地提高便成为了其必须考虑的问题。在NDN中节点之间可以通过请求包与数据包交互信息，从而根据不同节点的具体情况，实施不同的数据存储策略，有针对性的利用网络资源，使网络性能得到提高。

3.提升网络数据存储多样性:由于网络存储空间十分有限，如果某一数据副本在网内节点大量存储，那么，将使网内节点存储的数据种类明显减少，导致用户的请求无法在网内节点得到满足。因此，在满足用户需求的同时，应尽可能地减少同一数据副本在网内的存储量。

数据存储策略是NDN众多研究方向中成果较为突出的方向之一，因此，众多研究者提出了大量的NDN数据存储策略。目前关于数据存储策略的分类方式众多，不过这些分类方式都将数据存储策略分成了两种类型。通常情况下，从不同的角度出发一种数据存储策略可以由任意一种分类方式进行描述。下面将详细描述常见的几种NDN数据存储策略分类方式。

1. on-path，存储与off-path存储：由于NDN中数据包的返回路径正好为请求包所经过的路径(数据包与请求包的传输方向对称)，因此依据数据包存储的节点是否在请求包经过的路径上，存储策略可以分为on-path存储策略与off-path存储策略。其中，on-path存储策略为，存储数据包的节点位于请求包经过的路径上；off-path存储策略为，存储数据包的节点不在请求包经过的路径上。值得注意的是，on-path存储是NDN固有的存储方式。

2.协作存储与非协作存储:依据不同路由器节点之间是否协作(交互信息)，数据存储策略可分为协作存储策略与非协作存储策略。其中，非协作存储策略中的网内路由器具有相互独立的存储决策，并且不同路由器之间不相互交互数据的存储信息。与此相反，协作存储策略中的网内路由器通过相互协作对数据进行存储，或在路由器上建立数据在其他节点的存储信息列表，使请求能够根据该表在其他节点找到对应的数据。

3.齐次存储与异构存储：齐次存储表示数据包返回客户端经过的所有路由器节点具有相同的数据存储规则，而异构存储中数据包经过的路由器并不都具有相同的存储规则。

为了更好地对NDN中的数据存储策略进行阐述，下面主要按照on-path存储与“off-path”存储的分类方式，分别对目前NDN中常见的数据存储策略进行详细的描述。

on-path数据存储方式作为NDN中默认的一种数据存储方式，因此，NDN研究者提出了众多“on-path类型的数据存储策略。on-path类型的数据存储策略，通过将数据存储在请求包经过的路径上，往往不需要与其他非路径上的节点进行信息交互，将减少数据存储时对网络资源的消耗。

LCE(Leave Copy Everywhere)作为NDN中默认的数据存储策略，是一种典型的"on-path”类数据存储策略。LCE策略将数据存储在返回客户端时经过的所有节点中。这种数据存储方式不需要额外的运算，数据到达节点时，直接将其存储在CS中即可。虽然该策略的复杂度极低，但其“处处存储”的特性，使同一数据在网内存在大量的冗余副本，对缓存资源造成了严重的浪费。而且由于存储空间资源有限，这种“不加节制”的存储方式将使CS中的数据频繁替换，从而使请求在网内的命中率降低。

由上述内容可知，LCE数据存储策略使网内产生大量的冗余数据，无法高效地利用有限的存储空间，针对这些问题，LCD(Leave Copy Down)与MCD(Move Copy Down)数据存储策略在LCE的基础上进行了扩展。其中，LCD数据存储策略仅将数据存储在命中节点的下一跳节点中，其余节点只对数据进行转发，对数据进行存储。LCD策略与LCE策略相比，有效地减少了数据在网内的冗余量，同时，将流行度高的数据存储在了靠近客户端的节点中。而MCD策略在LCD策略的基础上做了进一步改进，更好地减少了网内数据的冗余。

上述on-path数据存储策略的核心思想是，将返回客户端的数据存储在请求经过的节点上。这类数据存储策略的复杂度较低，能够使数据在网络中实现快速存储。除此之外，与on-path类数据存储策略相对的off-path类数据存储策略也得到了广泛的研究。与on-path类数据存储策略不同的是，off-path类数据存储策略不仅将数据存储在请求包经过的路径中，往往还通过节点间的信息交互，将数据存储在请求包经过路径的邻居节点或其他路径节点中。

## 3.6 本章小结

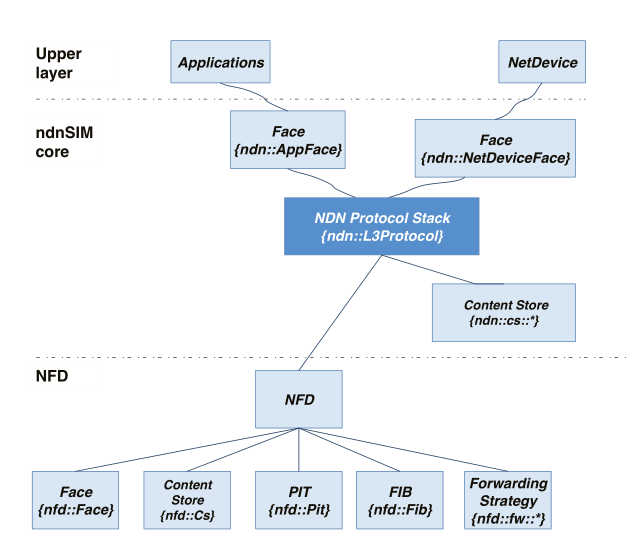
本章是对基于显式标记的传输控制机制模型的全面介绍。首先提出了一种基于检测缓存队列的拥塞检测方法，通过不断周期性地检测队列长度来判断链路是否发生了拥塞。然后基于拥塞检测提出了拥塞通告模型，拥塞通告是对拥塞状态的显式标记，把链路的状态封装到数据包中向下游节点转发，把网络的拥塞状态向其他节点进行通告。发送端在接收到拥塞通告后对发送端速率进行调整，从而能避免或解除拥塞的产生。多路径转发策略使用了动态分流的方式来启用多条链路，提高了网络中的接收速率。在网络中间节点上，提出了两种不同的缓存策略，通过缓存策略，提高了缓存命中率，降低了传输成本。最后对被章进行了总结。

# 第四章 基于显示标记传输控制机制的实现

## 4.1 ndnSIM简介

ndnSIM是UCLA张丽霞教授主导开发的CCN仿真工具，ndnSIM基于著名的NS-3仿真平台，实现了CCN相关协议，ndnSIM可以实现基于点对点通信、无线传输等模型的CCN系统级仿真。ndnSIM使用C++类实现了CCN中的网络层实体:PIT， FIB， CS，网络以及应用层接口，转发策略等。这种设计结构使得每一个模块都很容易改变而不会影响其他组件功能。与IPV4和IPV6协议栈类似，ndnSIM作为一个独立的协议栈，可以实施在任何一个仿真节点上。除核心协议栈之外，ndnSIM还包括一些基本的流量控制器，也提供了可扩展的接口以及帮助模块，使得基于ndnSIM平台实现仿真场景和跟踪仿真过程更加方便。

ndnSIM基本组成框架如下图4-1所示:



NDN::L3Protocol:核心CCN协议的部署，通过face接口自高层到低层接收兴趣包和数据包;ndn::Face:与应用(ndn::AppFace)以及其他仿真节点(ndn::NetDeviceFac)统一通信的抽象接口;ndn::ContentStore:网内缓存的抽象，可以部署不同的缓存决定和替换策略。ndn::Pit:记录接收到兴趣包face端口以及转发兴趣包的face端口;ndn::Fib:FIB转发模型的抽象，用于确定兴趣包的转发策略;ndn::ForwardingStrategy:兴趣包和数据包的转发模型抽象，在该组件中，转发策略包括查找CS，PTT，FIB。

本次仿真主要涉及ndnSIM中的ndn::ForwardingStrategy和ndn::Pit模块，下面重点介绍这两个模块

ndn::ForwardingStrategy模块包含对兴趣包以及数据包的处理流程，主要函数实现如表4-1所示。

表4-1 ndn：：ForwardingStrategy模块函数

|  |  |  |
| --- | --- | --- |
| 名字 | 功能 | |
| OnInterest | | | 兴趣包到达时调用 | |

|  |  |
| --- | --- |
| OnData | 数据包到达时调用 |
| WillErasePendingInterest | 移除PIT条目时调用 |
| RemoveFace | 移除转发端口 |
| DidReceiveDuplicateInterest | 判断接收到的兴趣包是否重复 |
| DidExhaustForwardingOptions | 判断是否已经尝试所有转发选项 |
| FailedToCreatePitEntry | 创建PIT条目失败 |
| DetectRetransmittedInterest | 按Interest名字查找PIT表，判断是否为重传Interest包 |
| WillSatisfyPendingInterest | 判断Data包是否与PIT条目匹配 |
| SatisfyPendingInterest | Data包与PIT条目匹配，会调用该函数清除PIT条目 |
| DidSendOutData | Data包是否发送成功 |
| ShouldSuppressIncomingInterest | PIT聚合 |
| TrySendOutInterest | 实际发送Interest包前调用 |
| DidSendOutInterest | 发送Interest包后调用 |
| PropagateInterest | Interest包转发逻辑 |
| DoPropagateInterest | 真正转发Interest包的逻辑 |

PIT记录兴趣包的转发状态，以便数据包能够正确的返回到内容请求方。PIT条目主要包括以下几个部分的信息：兴趣包包中的名字，接收到兴趣包的端口列表(多个端口以链表的形式存放)，兴趣包转发的端口列表，以及与转发相关的其他信息，比如PIT条目的过期时间，兴趣的转发策略等。现有ndnSIM版本提供了基本的PIT抽象模型，PIT表具有不同的替换策略。比如永久保存策略：当PIT达到最大设定大小时，将不能再添加新的条目；随机的：当PIT达到最大设定大小时，随机地从PIT表所有条目中选择并移除;LRU:当PIT达到容量极限时，最近最少未被使用的条目将被移除。PIT匹配时根据兴趣包或者数据包的名字进行Hash索引，并且会额外检测PIT条目的过期时间。首次收到的兴趣包会在PTT中创建条目，后续相同名字的兴趣包则会导致PIT聚合。对于PIT主要操作包括:Lookup，根据兴趣包或者数据包名字，查找匹配的条目；Create，根据兴趣包名字创建新的PIT条目；MarkErased，移除或者标记待删除的PTT条目；GetSize用于获知PIT表的容量Begi， End，Next这三个操作则用于遍历PIT表。

下图4-2显示了目前ndnSIM支持的转发策略，包括Nacks和GreenYellowRed；图中还包括ndnSIM内置的转发策略，包括Flooding，SmartFlooding和BestRoute。所有转发策略都继承自Nacks，采用了不同的方式检测和处Interest包中的NACK。

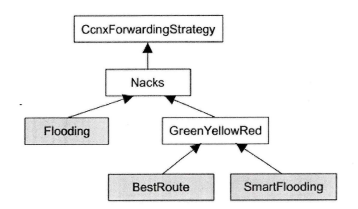


图 4-2 ndnSIM支持的转发策略图

Green 、Yellow和Red分别代表face端口的三种状态：Green表示face端口工作正常，可用于正常转发Interest包；Yellow表示face端口状态未知，可用于转发Interest，但不保证链路状态；Red表示face端口不能用于转发Interest包。初始化时，所有face端口均为Yellow，一旦某个face端口有Data包返回，就可以将该face端口置为Green，如果没有Data包返回，则可以置为Red。通信过程中，端口状态在Green，Yellow间动态变化，但是己经是Red状态的端口，则不会再用于转发Interest包。

Flooding:从Interest包名字匹配FIB后得到的所有转发端口同时转发该Interest，但不包括收到Interest的端口。

Smart Flooding:如果FIB匹配结果中包含Green端口，则从Green端口转发Interest包，否则从所有的Yellow状态端口转发Interest包。

Best Route:与Smart Flooding不同的是，没有匹配的Green端口时，只从等级最高的Yellow端口转发Interest包。

## 4.2 拥塞检测的实现

ndnSIM模拟了每个中间节点的行为，拥塞检测就是对发送队列的长度进行检测，在ndnSIM中，有对发送队列模拟的变量。

拥塞检测的实现是对ndn-limits-window类进行修改实现的。ndn-limits-window类位于/root/ndnSIM/ns-3/src/ndnSIM/utils中，在实现的过程中首先对UpdateLimit(double limit)函数进行发送队列更新，采用周期性的更新方式。

然后新建TestQueue( )函数，每隔100ms检测发送队列的长度，把瞬时的长度记录下来。

新建ScheduleNextTest( )函数，对要发送的数据包进行调度，调整数据包的发送速率，具体实现关键函数如下。

|  |
| --- |
| TestQueue( )函数的实现 |
| TestQueue():  void  LimitsWindow::TestQueue()  {  if(count == 0)  {  q1 = m\_outstanding;  count = 1;  }  else if(count == 1)  {  q2 = m\_outstanding;  count = 2;  }  else if(count == 2)  {  q3 = m\_outstanding;  count = 3;  }  else if(count == 3)  {  q4 = m\_outstanding;  count = 4;  }  else  {  q1 = q2;  q2 = q3;  q3 = q4;  q4 = m\_outstanding;  }  avgQueue = q1 \* w1 + q2 \* w2 + q3 \* w3 + q4 \* w4 + 0.4 \* (q4 - q1);  **ScheduleNextTest();**  } |

|  |
| --- |
| ScheduleNextTest（）函数的实现 |
| void  LimitsWindow::ScheduleNextTest()  {  **Simulator::Schedule (Seconds (0.5),&LimitsWindow::TestQueue, this);**  }  //每隔0.5s检测一次 |

|  |
| --- |
| JudgeLimit()函数的实现 |
| void  LimitsWindow::ScheduleNextTest()  {  **Simulator::Schedule (Seconds (0.5),&LimitsWindow::TestQueue, this);**  }  //每隔0.5s检测一次 |

## 4.3 拥塞通告的实现

在NDN中，节点会替每个Interest包维护一个重传计时器，只有在计时器超时的时候节点才能检测到拥塞，然后采取适当的恢复策略。这种隐式的检测机制太浪费时间，会影响传输的效率。假设节点N的某个接口的上游链路发生了拥塞，如果一个Interest包从这个接口转发，N是没有办法检测到这种状况。在重传计时器超时之前，如果N又收到同样的请求，它只会把收到Interest包的接口加入相应的PIT表项中，不采取任何的补救措施。

针对这个问题，作者设计了Interest Nack的反馈机制。节点如果无法满足Interest包的请求时，会返回一个Nack给发送Interest包的下游节点。Nack包含了原始Interest包的相关信息和一个错误代码。通过Nack的反馈机制，下游节点可以快速的检测到拥塞，并且利用NDN多路径转发的优势采取合适的恢复策略。不过，Nack并不能区分链路差错引起的丢包和拥塞引起的丢包或超时，更没有反映具体的拥塞程度。

Interest Nack的反馈机制已经在ndnSIM仿真平台上实现了。在ndnSIM中 Internet包的格式如图4-2所示：

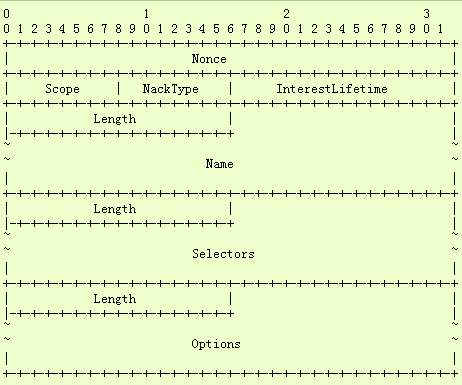


图4-2 Internet包格式

NdnSIM通过NackType字段区别正常的Internet包和Interest Nack，字段的取值如图4-3所示：

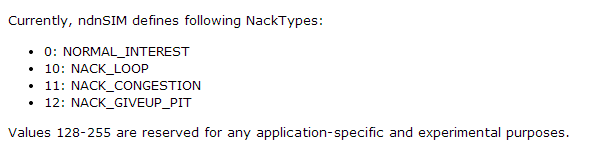


图4-3 NackType取值

从上图可以看出，当NackType的值为0的时候表示正常的Interest包，NackType值为10到12的时候表示Interest Nack。目前ndnSIM支持以下三种Nack：

1. NACK\_LOOP：判断节点是否收到了重复的Interest包，通过检测每个Interest包唯一的Nonce值实现的。如果存在两个Nonce值相同的Interest包，节点会丢弃其中一个，并通告相关的下游节点该Interest包产生了环路现象。

2. NACK\_CONGESTION：判断节点的上游链路是否发生了拥塞现象,不过ndnSIM暂时只是定义了这个值，没有具体实现拥塞的反馈。

3. NACK\_GIVEUP\_PIT：判断节点是否能转发当前的Interest包，如果Interest包能转发的所有接口的传输队列都满了(传输队列的上限是根据邻接链路的带宽计算出来的)，不能转发该Interest包的时候，节点会返回此Nack，让下游节点对Interest包采取其他的转发策略。

可以看出在ndnSIM目前的版本中，只有当接口发送的Interest包达到上限的时候，才会通告邻接的链路发生拥塞。这种拥塞通告的方式有两个弊端：

1. 只能知道网络是否是拥塞的，不能反映具体拥塞的程度。

2. 拥塞判断太过保守，发送出去的Interest包不一定会返回对应的Data包并且返回的时间不好判断，这样链路实际上还有空闲的带宽，可以满足额外的Interest请求。因此，提高每个接口能发送Interest包的上限，允许一定程度的突发流量，并且对Nack进行扩展，增加了三种新的Nack：

1. NACK\_FREE：判断网络是否空闲。

2. NACK\_BUSY：判断网络是否繁忙。

3. NACK\_CONGESTION：判断网络是否拥塞。

新增加的Nack仅在主动检测网络拥塞状况的时候才能使用。当链路处于空闲状态的时候(Free Link)，向相关的下游节点反馈NACK\_FREE。当链路处于繁忙状态的时候(Busy Link)，向相关的下游节点反馈NACK\_BUSY。当链路处于拥塞状态的时候(Congestion Link)，向相关的下游节点反馈NACK\_ CONGESTION。

一种逐跳的拥塞控制算法，收到Nack的接收端和中间节点会协同进行速率调节。

## 4.4 速率调整的实现

发送端速率的调节主要是通过修改consumer-window实现的。接收端采用MIAIMD (Multiplicative Increase Additive Increase Multiplicative Increase)算法，当收到NACK\_FREE的时候，采用乘性增加(Multiplicative Increase，MI)的算法，可以尽快占据网络空闲的带宽；当收到NACK\_BUSY的时候，采用和性增加(Additive Increase，AI)的算法，能够平滑的增大拥塞窗口，并且尽可能避免出现拥塞的现象；当收到NACK\_OVERLOAD的时候采用乘性减小(Multiplicative Decrease，MD)的算法，重新调整拥塞窗口的大小。中间节点在进行拥塞控制的时候，为了使得网络的状态不会出现太大的波动，采用了比较保守的AIMD算法。当收到NACK\_BUSY的时候，采用和性增加(Additive Increase，AI)的算法，允许发送少量的Interest包，提高链路带宽利用率的同时减少拥塞发生的概率； 当收到NACK\_CONGESTION的时候采用乘性减小(Multiplicative Decrease，MD)的算法，调整拥塞窗口的大小，缓解拥塞的状况。

主要的接口包括OnData()和OnNack()，具体的实现逻辑如下

|  |
| --- |
| OnData()函数的实现 |
| void  ConsumerWindow::OnData (Ptr<const Data> contentObject)  {  Consumer::OnData (contentObject);  if(m\_adjust == false) //NACK\_FREE  {  m\_window = m\_window + 1; //MI algorithm  cwnd = cwnd + 1;  }  Else //NACK\_BUSY, AI algorithm  {  cwnd = cwnd + 1/cwnd;  if(cwnd >= m\_window + 1) //当cwnd增加值累积增加到1， 将m\_window增加1  {  m\_window = m\_window + 1;  cwnd = m\_window;  }  }  if (m\_inFlight > static\_cast<uint32\_t> (0)) m\_inFlight--;  NS\_LOG\_DEBUG ("Window: " << m\_window << ", InFlight: " << m\_inFlight);  ScheduleNextPacket ();  } |

|  |
| --- |
| OnNack函数的实现 |
| void  ConsumerWindow::OnNack (Ptr<const Interest> interest)  {  if(interest->GetNack() == 100 ||  interest->GetNack() == 120 ||  interest->GetNack() == 140)  {    if(interest->GetNack() == 140) //**overload, MD**  {  if(flag == false)  {  m\_window = 0.85 \* m\_window;  cwnd = m\_window;  flag = true;  }  else  {  flag = false; //每隔一个NACK包调整一次  }  }  else if(interest->GetNack() == **120)**  {  **m\_adjust = true; //busy, AI调整**  }  else  {  m\_adjust = false; //**free, MI**  }  }  else  {  count++;  NS\_LOG\_DEBUG ("Nack: " << count);  Consumer::OnNack (interest);  if (m\_inFlight > static\_cast<uint32\_t> (0)) m\_inFlight--;  //if (m\_window > static\_cast<uint32\_t> (0))  // {  // m\_window = std::max<uint32\_t> (0, m\_window - 1);  //m\_window = 0.85 \* m\_window; //原来ndnSIM的处理，收到NACK，速率MD减小为原来的0.85倍  //}  //NS\_LOG\_DEBUG ("Window: " << m\_window << ", InFlight: " << m\_inFlight);  ScheduleNextPacket ();  }  } |

## 4.5 多路径转发的实现

根据Named Data Network中的原理，路由器节点转发来自数据请求端（Consumer）的Interest包，转发数据包的方式被称为转发策略（Forwarding Strategy），转发策略决定了数据包的转发过程。由于NDN位置无关、面向内容等特性，数据包的转发和面向连接的TCP/IP网络有很大的区别，转发策略对网络性能有着很大的影响。NDN中常见的转发策略包括BestRoute和Flooding两种，其中BestRoute是在转发路径当中选一条网络度量最优的路径转发，Flooding是泛洪的转发方式。经过对NDN中内容转发过程的分析，BestRoute由于受到背景流量和网络拥塞等因素的影响并不具备比较好的效果。从原理上来说，BestRoute是一种单路径的转发方式，类似于TCP/IP网络中的转发方式。由于NDN的连接无关的特性可以考虑通过多条路径进行数据转发，下文将对多路径转发的实现进行详细介绍。

ndnsim内部定义了关于转发策略的基类：ForwadingStrategy，封装了关于转发的几个方法。

首先自定义一个转发策略类，继承该基类。

class CustomStrategy:ForwardingStrategy

重写DoPropagateInterest ( Ptr<Face> inFace, Ptr<const Interest> interest,Ptr<pit::Entry> pitEntry )方法。

该方法是具体的转发过程的实现，inFace是接收兴趣包的接口，interest是接收到的兴趣包，pitEntry是和该兴趣包绑定的一个pit项。在这个方法中通过对每个可能outFace的当前Pi值进行统计，把hops作为唯一的metric，然后最小的选择hops最小的outFace，通过该接口转发兴趣包。

当某个接口收到NACK\_CONGESTION信号的时候，对该接口进行分流，分流操作通过统计的Pi值实现，Pi值代表着流量的数量，对Pi值分流后，检测链路的拥塞变化，进行相同的操作直到流量的比率收敛到最优值。

## 4.6 本章小结

本章是对基于显示标记的传输控制机制实现的介绍。首先介绍了NDN模拟平台ndnSIM，然后在ndnSIM平台的基础上对拥塞检测、拥塞通告、速率调整和多路径转发的实现进行了详细介绍，实现了上一章描述设计的高效传输控制机制，传输机制的关键是显示标记对网络状况的智能感知，由于显示标记的存在，整个网络拥塞控制变得更加及时有效。

# 第五章 基于显式标记传输控制机制的评估

## 5.1 拥塞控制性能评估

对拥塞控制机制评估实验是在如下的拓扑中进行的，如下图C是数据请求端，R1和R2是路由器，S是数据源。R1和R2是瓶颈链路。

首先在C上部署本文提出的速率调节算法，R1和R2上部署拥塞控制算法，数据通告选项封装到了数据包中。C以慢启动的方式开始请求数据，随着发送速率不断增加，R1R2链路开始达到饱和，然后R2向下游发送拥塞通告，C对发生拥塞的链路进行速率调整。

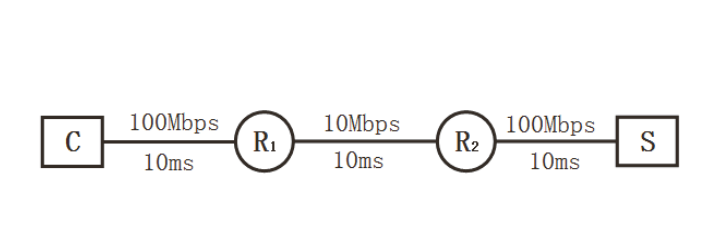
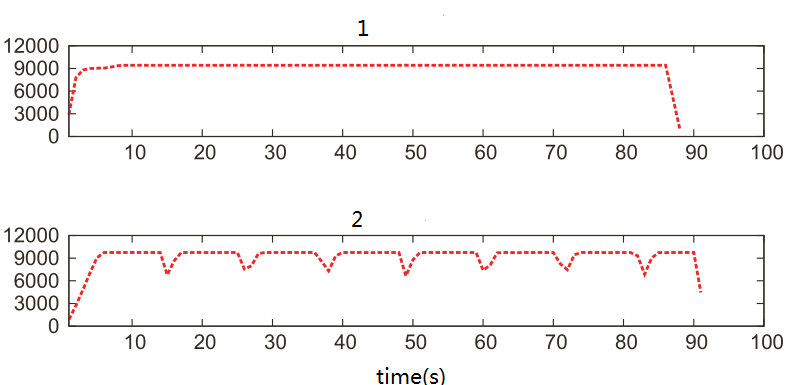


图 5-1 单路径拓扑图

在保持相同拓扑的情况下，把发送端C的速率调整算法改为类的隐式拥塞控制协议（ICP）。R1和R2不做出拥塞控制调整。拥塞检测依赖超时触发，发送端速率调整为减半操作。两种情景下的实验结果如下。

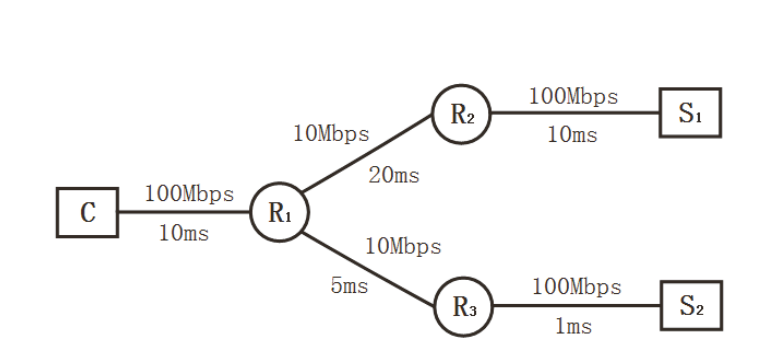


实验1为部署了本文中的拥塞控制机制后吞吐率与时间的关系图。实验2是未部署拥塞控制机制吞吐率与时间的关系图，可以看出实验室1的平均吞吐率大于实验2的平均吞吐率。因为实验1中提前进行拥塞调整，不发生拥塞，没有超时的出现，吞吐率没有下降；而实验2中发送端速率过大时会引起超时，发送端速率会下降，所以会出现图中吞吐率下降的现象。

通过上述实验可以看出来本文提出的拥塞控制机制对网络性能有了提升。

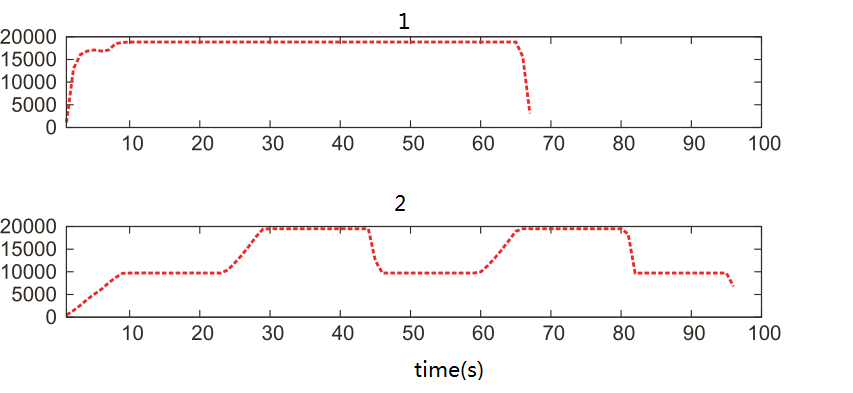
## 5.2 多路径转发评估

对拥塞控制机制评估实验是在如下的拓扑中进行的，如下图C是数据请求端，R1、R2、R3是路由器，S1、S2是数据源。R1R2、R1R3是瓶颈链路。



首先在C上部署本文提出的速率调节算法，R1上部署本文提出的多路径转发算法，R2、 R2上部署拥塞控制算法，数据通告选项封装到了数据包中。C以慢启动的方式开始请求数据，随着发送速率不断增加，R1R2链路开始达到饱和，然后R2向下游发送拥塞通告，R1开始启用R1R3链路，R1R3饱和后，C对发生拥塞的链路进行速率调整。

在保持相同拓扑的情况下，把发送端C的速率调整算法改为类的隐式拥塞控制协议（ICP）。R1、 R2和R3不做出拥塞控制调整，R1R2发生拥塞后启用R1R3，当R1R3发生拥塞后发送端C速率进行调整。拥塞检测依赖超时触发，发送端速率调整为减半操作。两种情景下的实验结果如下。



实验1为部署了本文中的拥塞控制机制和多路径转发策略后吞吐率与时间的关系图。实验2是未部署多路径转发策略吞吐率与时间的关系图，可以看出实验室1的平均吞吐率大于实验2的平均吞吐率。因为实验1中R1R2带宽饱和时，数据包向R1R3发送，直至达到饱和，由于开启了基于显示标记的拥塞控制机制，不发生拥塞，没有超时的出现，吞吐率没有下降；而实验2中发送端速率过大超过R1R2带宽时，数据包会缓存在R1R2的队列当中，这一部分对应2中平坦的图像，随着数据包不断增加，R1R2的缓存队列用完时候，数据包开始向R1R3，随着发送端速率继续的增加，R1R3缓存队列也会用完，超出缓存对列的数据包会被丢弃，从而引起超时，发送端速率出现减半的现象，所以会出现图中吞吐率下降的现象。

通过上述实验可以看出来本文提出的拥塞控制机制对网络性能有了提升。

## 5.3 本章小结

本章是对上一章实现的传输控制机制的实验评估。先对拥塞控制机制进行了评估，然后结合对多路径转发策略进行了评估，通过实验结果分析可以得出基于显式标记的传输控制机制对于NDN网络性（吞吐率）能有了提升。

# 第六章 总结与展望

本文研究NDN中传输控制机制的问题。首先分析了 NDN 体系结构的特点和 NDN 的传输模式。接着介绍了目前在传输协议和拥塞控制算法的研究现状和存在的问题。很多传输协议都是基于单一数据源的假设，没有考虑请求的内容可能来自不同的数据源的情况。而考虑了多源的传输协议仍然沿用了传统 TCP 协议的隐式检测机制，这在 NDN 中是不可靠的。因此，还分析了显式的标记在 NDN 中的可行性。然后，文本针对目前上面的问题，提出了一种基于显式标记传输控制机制，算法包括拥塞检测，拥塞通告，速率调节，多路径转发和缓存策略三个部分。最后在ndnSIM 平台上进行了仿真实验，在基本拓扑，多路径拓扑两种不同的环境中测试和比较了本文提出的传输控制机制与传统的类TCP算法的性能，用吞吐率作为评价算法的指标。实验结果表明，和类ＴＣＰ算法相比，本文提出的传输控制机制有更高的传输效率和稳定性。在未来的工作中，我们将继续改进我们的传输控制机制。主要包括以下三个方面：(1) 本实验目前仅在单个拥塞节点的拓扑中实验，下一步会在多个拥塞节点的拓扑中评估算法的性能并根据实验结果改进算法。(2) 利用 NDN 中间节点可以进行拥塞控制的特性，设计逐跳的拥塞控制算法，在接收端和中间节点协同进行速率调节，充分发挥中间节点的性能。(3) 目前算法只是采用仿真实验的方式进行性能评价，缺乏对算法性能的理论分析及对算法设计的理论指导，拟采用流体模型对算法进行分析和评价。

# 参考文献

[1] Van Jacobson, et al., Networking Named Content, In Proc. of CoNEXT'09, Rome, Italy, 2009.

[2] 谢高岗, et al., 未来互联网体系结构研究综述[J]. 计算机学报, 2012(06): 1109-1119.

[3] 任勇毛, 李俊, 钱华林. 未来互联网体系结构研究进展 [J]. 科研信息化技术与应用, 2012, 3(3): 3-11.

[4] Lixia Zhang, Van Jacobson, et al.,Named Data Networking (NDN) Project, NSF Project Proposal, Oct. 2010.

[5] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, ICP: Design and Evaluation of an Interest Control Protocol for Content-Centric Networking, in Proc. of IEEE INFOCOM Workshop on emerging design choices in name oriented networking (INFOCOM NOMEN), Orlando, USA, March 2012.

[6] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, Joint Hop-by-hop and Receiver-Driven Interest Control Protocol for Content-Centric Networks , in Proc. of ACM SIGCOMM Workshop on information centric networking (ICN), Helsinki, Finland, August 2012.

[7] S. Salsano, A. Detti, M. Cancellieri, M. Pomposini, andN. Blefari-Melazzi, eceiver-Driven Interest Control Protocol for Content-Centric Networks , in Proc. of ACM SIGCOMM Workshop on information centric networking (ICN), Helsinki, CN ’12. New York, NY, USA: ACM,2012.

[8] Lorenzo Saino, Cosmin Cocora and George Pavlou,CCTCP: A Scalable Receiver-driven Congestion ControlProtocol for Content Centric Networking, in Proc. of IEEE

ICC’13, 2013.

[9] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, A case for stateful forwarding plane, Computer Communications: Information-Centric Networking Special Issue, vol. 36, pp. 779-791, 2013.

[10] A. Afanasyev, I. Moiseenko, and L. Zhang, ,ndnSIM: NDN simulator for NS-3, for stateful forwarding plane, Comput.

[11] D. Saucez, L. A. Grieco, and C. Barakat. AIMD and CCN: past and novel acronyms working together in the Future Internet. In ACM CoNEXT 2012, CSWS Workshop, 2012.

[12] M. Alizadeh, A. Greenberg, D. Maltz, J. Padhye, P. Patel, B. Prabhakar,S. Sengupta, and M. Sridharan, S Workshop, 2012munications: Information-Centric Networking Special.

[13] Dina Katabi, Mark Handley, Chalrie Rohrs. CongestionControl for High Bandwidth-Delay Product Networks[C] Proc. of ACM Sigcomm 2002.

[14] Yong Xia. Lakshminarayanan Subramanian, Ion Stoica. Shivkumar Kalyanaraman. One more bit is enough. ACM SIGCOMM Computer Communication Review, 2005, 35(4): 37-48.

# 致谢

在硕士学位论文即将完成之际，我非常真诚的向曾经给与我支持和帮助的老师及同学们表示衷心的感谢。

衷心的感谢我的导师 教授，他在工作过程中一直秉承着严谨、细致的作风，这种工作态度一直是我学习的榜样。在我攻读硕士学位期间，恩师为我的生活和学习付出了很多努力，没有他的带领，我不可能成为一名合格的毕业生。

恩师严谨的治学态度、敏锐的学术思维、以及精益求精的工作作风对本人有着非常深远的影响。这种作风不但引导我在工作学习中，不断树立奋斗目标，更使我学会了如何做人，如何做事。本文从选题到完成，都倾注了恩师大量的心血。在此，谨向恩师表示崇高的敬意和衷心的感谢!

衷心的感谢公司技术总监 ，本课题无论在选题还是在研究阶段，都得到了他的悉心指导。并且在之后的研究过程中，他为我指点迷津，帮我开拓思路，在专业知识上的帮助让我受益匪浅。

衷心的感谢已经毕业了的实验室的师兄师姐们，他们无论在实验室还是在公司，都给了我很大的帮助，让我能够尽快熟悉项目，不断提高。

还要感谢一起奋斗的同学们，他们不仅是我学习和生活上的伙伴，更是我面对困难和挑战时的战友。

最后，再次感谢我的老师、同学们，感谢你们一直以来对我的的关心和支持。