文本复制检测报告单(去除本人已发表文献)

№:ADBD2019R_20190217221905201902251822091003700738803　　检测时间:2019-02-25 18:22:09

检测文献:　S1481237969_ 基于SDN网络的视频流媒体传输性能研究

作者:

检测范围:　中国学术期刊网络出版总库

中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库

中国重要会议论文全文数据库

中国重要报纸全文数据库

中国专利全文数据库

图书资源

优先出版文献库

学术论文联合比对库

互联网资源(包含贴吧等论坛资源)

英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)

港澳台学术文献库

互联网文档资源

CNKI大成编客-原创作品库

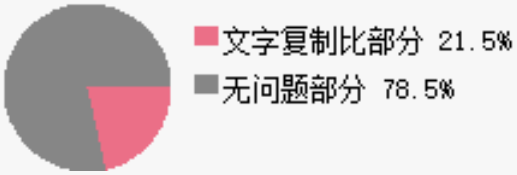
个人比对库

时间范围:　1900-01-01至2019-02-25

检测结果

去除本人文献文字复制比：21.5%

重复字数：	[10976]	总段落数：	[6]
总字数：	[51040]	疑似段落数：	[6]
疑似段落最大重合字数：	[3205]	前部重合字数：	[2315]
疑似段落最小重合字数：	[161]	后部重合字数：	[8661]



指　标：☐疑似剽窃观点　☑疑似剽窃文字表述　☐疑似自我剽窃　☐疑似整体剽窃　☐过度引用

表格：0　　公式：1　　疑似文字的图片：0　　脚注与尾注：0

23%（2442）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第1部分（总10602字）
32%（2938）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第2部分（总9194字）
8.9%（856）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第3部分（总9621字）
33.5%（3205）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第4部分（总9555字）
14.2%（1374）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第5部分（总9699字）
6.8%（161）	S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第6部分（总2369字）

（注释：无问题部分　文字复制比部分）

1. S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第1部分　　总字数：10602

相似文献列表　文字复制比：23%(2442)　疑似剽窃观点：(0)

1	S312060111+杨俊超 杨俊超 -《学术论文联合比对库》- 2014-12-26	6.4%（677） 是否引证：否
2	基于OpenFlow网络的QoS管理策略研究 杨俊超(导师：孙建国) -《哈尔滨工程大学硕士论文》- 2015-03-01	5.9%（625） 是否引证：是
3	基于OpenFlow的SDN网络QoS路由策略研究 周怡(导师：苏俭) -《电子科技大学硕士论文》- 2018-04-10	5.0%（534） 是否引证：是
4	8_陈彬_5G无线网络协作中继技术及仿真研究 陈彬 -《学术论文联合比对库》- 2017-04-05	4.8%（504） 是否引证：否
5	026_201422260255_陈忠	4.6%（484）

	陈忠 - 《学术论文联合比对库》- 2017-04-06	是否引证：否
6	4-杨俊超	4.0%（420）
	杨俊超 - 《学术论文联合比对库》- 2014-12-20	是否引证：是
7	（通信与信息系统专业优秀论文）实时视频流传输与控制的研究 - 豆丁网	1.4%（151）
	-《互联网文档资源（http://www.docin.com）》- 2017	是否引证：否
8	基于SDN的网络化接口研究	1.1%（114）
	邹圣恺(导师：雷志勇) -《西安工业大学硕士论文》- 2018-05-17	是否引证：否
9	基于OpenFlow的视频流媒体路由选择算法	1.1%（112）
	赵钊(导师：张基宏) -《深圳大学硕士论文》- 2017-06-30	是否引证：否
10	SDN网络链路和控制器故障恢复机制研究	1.0%（106）
	王立坤(导师：吴国伟) -《大连理工大学硕士论文》- 2018-03-20	是否引证：否
11	SDN高效安全链路发现及控制器优化部署	0.9%（95）
	赵鑫(导师：吴国伟) -《大连理工大学硕士论文》- 2018-03-26	是否引证：否
12	大型面阵MIMO雷达射频隐身性能研究	0.6%（64）
	蔡茂鑫(导师：何子述) -《电子科技大学硕士论文》- 2013-05-06	是否引证：否
13	中小学生“手机控”,该导还是该禁?	0.5%（50）
	王宏霞; -《中小学心理健康教育》- 2018-12-11	是否引证：否
14	新媒体语境下体育类谈话节目的传播策略——以《超级颜论》为例	0.5%（50）
	刘时坤; -《青年记者》- 2018-12-20	是否引证：否
15	“互联网+”时代的大学教师发展	0.4%（46）
	陈素娜;吴艺娜; -《安庆师范大学学报(社会科学版)》- 2018-12-25	是否引证：否
16	基于BASS模型的共享单车用户扩散研究	0.4%（40）
	齐有为; -《黑龙江生态工程职业学院学报》- 2019-01-20	是否引证：否
17	06_BY1106102_宋平	0.4%（38）
	宋平 - 《学术论文联合比对库》- 2017-11-20	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

基于SDN网络的视频流媒体传输性能研究

学科：

研究生签字：

指导教师签字：

摘要

近年来，随着互联网与多媒体技术的迅速发展，视频流媒体应用在互联网上大放异彩，其严格的QoS要求在为人们的生活和工作带来便利的同时，也给网络的传输带来不小的挑战，传统网络存在的种种弊端常常造成视频在传输过程中的不稳定，严重影响视频传输的服务质量。软件定义网络（Software-Defined Networking，SDN）作为一种新型的网络架构，其转控分离、集中控制及可编程的思想为解决传统网络中的流量工程、QoS路由等问题提供了新的思路。本文利用SDN的特性，在SDN网络中对视频流媒体的传输性能进行研究，主要研究内容包括以下几个方面：

首先，本文对四种不同种类业务进行了优先级的区分，并设定了不同的优先级值，提出了视频流媒体的QoS控制策略。然后，基于遗传算法的QoS路由为视频流媒体（最高优先级业务）计算传输路径，基于Dijstra算法以跳数为代价为其他优先级业务计算传输路径，且当控制器检测到视频流媒体的传输路径出现拥塞时，采取动态路由措施更好地保障视频流媒体的QoS。其次，使用HTB队列规则在OpenFlow交换机上进行不同优先级业务的区分调度，优先保障视频流媒体业务的QoS，同时提供带宽充足时的借带宽机制，尽力保障其他业务流的QoS。

最后，对QoS控制框架的拓扑管理模块、链路信息测量模块、路由管理模块以及队列调度模块分别进行了实现，并在Mninet、Ryu控制器、摄像头等软件搭建的SDN网络传输环境中，对QoS控制策略进行了测试。通过一系列仿真实验，从链路时延抖动、吞吐率等视频流媒体的传输性能参数方面验证了本文控制策略的可行性。仿真实验结果表明，QoS控制策略能为视频流媒体选择一条符合需求的路径进行传输，能够对视频流媒体进行动态路由，能够在数据转发层区分保障不同优先级业务的QoS，较好地保证了最高优先级业务端到端的QoS需求。

关键词：软件定义网络；视频流媒体；QoS路由；遗传算法；队列调度

Research on Video Streaming Media Transmission Performance based on SDN Network

Discipline：Pattern Recognition and Intelligent System

Student Signature：

Supervisor Signature：

Abstract

In recent years, as the Internet and multimedia technologies have been developed rapidly, video streaming is widely

used on the Internet, Its stringent QoS requirements in bring convenience for people's life and working at the same time, also bring a big challenge to the network transmission, The disadvantages of the traditional network often cause the instability in the transmission process of video, which seriously affects the service quality of video transmission. As a new network architecture, software-defined Networking (SDN) provides a new train of thought for traffic engineering, QoS routing and other problems in traditional network by taking advantages of the separation of control, centralized control and programmable. In this paper, the performance of video streaming media in SDN is studied by using the characteristics of SDN, the main research content includes the following several aspects:

First of all, this paper makes a priority distinction between four different kinds of businesses, and sets different priority values, and puts forward the QoS control strategy of video streaming media. Then, the QoS routing based on genetic algorithm calculates the transmission path for the video streaming media (the highest priority business), and calculates the transmission path for other priority services at the cost of the jump number based on the Dijstra algorithm, and when the controller detects congestion in the transmission path of video streaming media, dynamic routing measures are adopted to better protect the QoS of video streaming media. Secondly, the HTB queue rules are used to make the differentiated scheduling of different priority services on the OpenFlow switch, and the QoS of the video streaming media service is guaranteed firstly, at the same time, it provides the borrowing bandwidth mechanism when bandwidth is sufficient, and makes every effort to guarantee the QoS of other business flows.

Finally, the Topology Management module, link information measurement module, routing management module and queue scheduling module of QoS control framework are implemented respectively, and the QoS control strategy is tested in SDN network transmission environment built by Mninet, Ryu controller, camera and other software. Through a series of simulation experiments, the feasibility of this control strategy is verified from the aspects of transmission performance parameters of video streaming media such as chain delay jitter and throughput rate. Simulation experimental results show that QoS control strategy can select a path that meets the requirements of video streaming media for transmission and can dynamically route video streaming media, and can distinguish the QoS of different priority business in the data forwarding layer, and ensure the QoS requirement of the highest priority service end-to-end.

Key Words：SDN; Video streaming media; QoS routing; Genetic algorithm; Queue scheduling

目录

1 绪论 1
1.1 研究背景及意义 1
1.2 国内外研究现状 2
1.2.1 SDN研究现状 2
1.2.2 SDN网络中视频流媒体QoS控制策略的研究现状 2
1.2.3 存在的问题 3
1.3 论文的主要研究内容及结构安排 4
1.3.1 论文的主要内容 4
1.3.2 论文的结构安排 5
2 SDN中QoS的相关技术分析 6
2.1 QoS技术 6
2.1.1 QoS的定义 6
2.1.2 QoS服务模型及机制 6
2.2 传统网络下的流媒体传输分析 7
2.3 SDN相关技术 7
2.3.1 SDN控制器 9
2.3.2 OpenFlow 9
2.3.3 OpenFlow对QoS的支持 13
2.3.4 智能算法在SDN上运行的可行性分析 14
2.4 本章小结 14
3 QoS控制策略需求分析与设计 16
3.1 需求分析 16
3.1.1 应用环境分析 16
3.1.2 应用需求分析 16
3.2 QoS控制策略框架 17
3.2.1 数据流分类 17
3.2.2 QoS控制策略的架构设计 18

3.3 QoS控制框架模块分析设计 20
3.3.1 QoS路由框架 20
3.3.2 队列调度策略的分析设计 28
3.4 基于遗传算法的多QoS约束路由算法分析与设计 31
3.4.1 路由度量数学模型 31
3.4.2 遗传算法基本概念 31
3.4.3 染色体编码设计 32
3.4.4 种群初始化方法设计 33
3.4.5 适应度函数设计 33
3.4.6 遗传算子的设计 34
3.5 本章小结 37
4 QoS控制策略实现 38
4.1 QoS控制框架实现概述 38
4.2 QoS路由的实现 39
4.2.1 链路性能测量模块的实现 39
4.2.2 拓扑管理模块的实现 44
4.2.3 路由管理模块 45
4.2.4遗传算法的实现 47
4.3 队列调度策略的实现 55
4.4 本章小结 58
5 实验仿真与结果分析 59
5.1 仿真实验环境介绍 59
5.1.1 网络仿真软件Mininet 59
5.1.2 OpenvSwitch 60
5.1.3 其他实验环境及相关软件介绍 60
5.2 网络环境的搭建 60
5.2.1 拓扑环境搭建 61
5.2.2 视频服务器搭建 63
5.3 实验仿真与结果分析 64
5.3.1 QoS路由计算有效性分析 64
5.3.2 动态路由有效性分析 65
5.3.3 队列调度的QoS控制性能测试 68
5.3.4 QoS控制策略性能分析 70
5.4 本章小结 71
6 总结与展望 72
6.1 全文工作总结 72
6.2 不足与展望 72
参考文献 75
攻读硕士学位期间发表的论文 78
致谢 79
学位论文知识产权声明 80
学位论文独创性声明 81

1 绪论

1.1 研究背景及意义

在互联网技术日益成熟及多媒体应用快速普及的背景下，视频应用及视频用户的数量激增。根据中国互联网络信息中心（CNNIC）发布的第42次中国互联网络发展状况统计报告[1]：截至2018年6月底，网络视频用户规模为6.09亿，与去年年末相比增长了5.2%，占网民总体的76.0%，视频应用正在日益成为多媒体通信中发展最为迅速的领域。

互联网技术的崛起与快速发展促进了视频业务的方兴未艾，视频点播、视频直播、视频会议以及监控视频等一系列流媒体视频应用正不断改变着人们的工作和生活。在庞大且复杂的网络中如何更好地为用户提供视频服务，成为流媒体视频领域研究的一个方向。一般来讲，若想得到较好的视频观看效果，需要在视频传输过程中保证网络带宽的充足以及较低的时延和丢包率，但是传统网络发展到今天已有了较大的局限性，不可能无限制的提供资源来满足不同视频的传输需求。针对媒体业务在网络中传输遇到的困境，IETF提出了多种QoS模型和机制，但是这些模型和机制基本上是建立在分布式的网络架构上，全局网

络状态信息的收集难度非常大，致使往往不能及时地改变数据的传输策略，不能在根本上解决传输问题，无法有效地保证视频流媒体的服务质量（Quality of Service，QoS）[2]，当大量数据请求叠加时，容易造成传输路径的拥堵，使得视频数据经过很长时间才到达对端或者拥堵时间过长而直接被丢弃，严重影响视频的QoS。

技术的革新使得视频流媒体应用百花齐放，同时也对传统网络提出了更高的要求，面对飞速发展的视频流媒体业务，传统网络越来越“心有余而力不足”。面对流媒体应用的大量涌出、传统网络日渐凸显的弊端以及在传统网络中部署新的方案不但难以彻底解决问题反而加剧网络的复杂的情况，越来越多的网络研究人员倾向于寻求一种新的网络体系结构来解决传统网络的弊端，进而提高网络的传输效率。软件定义网络（Software Defined Networking ，SDN）[3]是一种区别于传统网络架构的新式网络体系结构， 其将网络的控制平面与数据转发平面解耦合，控制平面可以获得网络的全局拓扑及流量信息，并根据这些信息作出合适的转发路径决策，数据转发层根据收到的转发命令进行业务流的转发而不做任何控制决策行为，这大大减少了网络的复杂度并提高了转发设备转发数据的速度。在SDN网络中，若要实现特定的网络需求，只需根据该业务需求去修改控制层的配置，便可以改变转发层网络设备的行为。对于视频传输而言，利用SDN网络架构可以更好地控制视频流的传输，使网络更好地为视频业务服务。

1.2 国内外研究现状

1.2.1 SDN研究现状

为了解决TCP/IP架构所出现的问题，各国都进行了大规模的网络研究工作，比如欧盟的FIRE[4]、美国的GENI[5]等。源于校园的SDN被认为在解决传统网络的一些问题上有着其他网络架构无法比拟的优势，越来越多的高校及企业都在关注SDN的发展趋势甚至投入大量的资金来进行研究。

从2006年概念的提出到校园研究再到近几年真正的实现应用，SDN的发展已有十几年。目前，SDN相关规范和标准的制定由开放网络基金会（Open Networking Foundation，ONF）负责 [6]，其对OpenFlow协议进行了不断地修改和完善，并不断充实SDN相关内容。作为南向接口协议的标准，OpenFlow从最初的1.0版本到现在的1.5[7]版本，其能实现的功能越来越完备，面向应用层的北向接口虽未标准化也在不断的完善中。作为软件定义网络的核心，SDN控制器和交换机的发展也很快，华为、IBM、H3C等都推出了自己厂商的交换机，各种语言编写的SDN控制器抢占着控制器的市场，目前使用较多的控制器有Ryu[8]，FloodLight[9]，OpenFaylight[10]。

目前国内外各大高校的研究重点主要在如流量工程、QoS路由以及与传统网络互联等SDN的应用、SDN测试实验以及控制平面的设计等。除此之外，国内外各大互联网公司也纷纷参与到SDN研究与应用工作中，谷歌使用SDN网络架构的思想在其跨欧亚美三大陆的12个数据中心网络成功部署了B4网络[11]，B4的最大特点是实现了整个网络的链路利用率达到95%以上；微软为了提高其网络链路的利用率，在数据中心网络部署了software-driven WAN[12]，其链路的利用率能达到99%；中兴公司正着手进行基于NFV/SDN的面向5G网络运维管理转型的解决方案架构设计。

国内三大运营商也加入到SDN研究的浪潮里，如中国电信为应对云资源池网络挑战，在云资源池中部署应用了网络能力开放可编程、网络业务敏捷灵活、网络规模支持海量多租户、云网融合智能编排的SDN方案，并实现了广泛部署和应用。SDN交换机的设计与研发也引起了许多厂商的兴趣，如思科、IBM等厂商都推出了具备SDN功能的交换机及芯片；中国的华为公司在2013年发布了其第一款支持SDN功能的交换机；随后盛科等公司也推出了自己公司的SDN交换机。

1.2.2 SDN网络中视频流媒体QoS控制策略的研究现状

1）国内研究现状

文献[13]提出了OpenFlow网络下可分级视频编码（SVC）的流媒体传输方案，该方案利用SVC编码将视频分成不同的层级（基础层和增强层），根据网络的传输情况，传输不同的层级，基础层保证了最基本的观看效果。

文献[14] 提出了一种新的基于OpenFlow网络的QoS算法，通过对QoS流和非QoS流进行区分，对非QoS流直接选择最短路径进行传输，对QoS流运行QoS路由算法寻找符合需求的路径进行传输。

文献[15]通过构建视频流分类模型，并将DiffServ的思想应用到视频流的调度中，以PGPS等调度算法为基础，给予视频业务差别的QoS保障，满足了网络中视频业务的传输需求。

文献[16]利用SDN架构特点从控制层面和转发层面实现了基于SDN的业务QoS保障，全局层面的QoS保障更能体现SDN架构的11优越性。

2）国外研究现状

K.-W. Kwong，R. Guerin 等人于 2007 年提出了对偶路由算法[17]，这种算法将数据流分成两类，每类对应不同的链路权重，后根据权重来求得路由。

Civanlar 等研究了通过OpenFlow网络进行视频流传输的QoS路由[18]。作者介绍了一种基于线性规划的方法为基于SVC编码的视频流计算QoS路由，旨在降低路径丢包率低同时将时延控制在要求范围内，同时计算流的QoS路由。他的基本实现是，对应 best-effort 业务流按照最短路径进行转发，视频业务按照提出的算法计算的QoS-rich路径进行转发。

Nikhil Handigol，Srini Seetharaman 等人于2009年提出了一种基于OpenFlow网络的路由方法[19-20]，对每一个数据流根据当前的网络负载情况计算出路由。

Egilmez 等人设计了一种基于SDN的OpenQoS[21]控制策略，该策略根据不同数据流的数据包头字段存在的差异，将传输的流量分为两种不同类型，对于视频流，Open QoS策略考虑传输路径上的延迟、丢包情况为其计算满足QoS需求的传输路径，其他数据流的路由仍是最短路径。

Dobrijevic等人将QoS的概念扩展到QoE，通过QoE模型来表征路由的服务质量，基于蚁群算法为不同业务流寻找将其QoE 最大化的路由[22]。通过对视频、语音和纯数据业务的路由实验与控制器默认的最短路径算法结果对比，验证了在一定网络规模内，该策略相比于最短路径算法路由体系的优越性。

Owens和 Durresi 介绍了另一种控制器架构和协议VSDN，用于SDN网络视频应用的QoS保障，允许视频应用向网络请求端到端的严格QoS保证服务[23]。这需要通过修改 Open Flow 提供的部分交换功能来实现，排队过程根据 VSDN 控制器提供的流量规范来调整每个流的流量。

1.2.3 存在的问题

整体上来看国内外对SDN网络的QoS研究未能充分发挥SDN网络架构的优势。具体说来当前研究存在以下问题：

（1）大多数的研究没有在控制层算法上进行深挖，所采用的路由算法未能充分考虑网络的现状。

（2）大多数的学者对数据层和控制层进行分开研究，没有把他们结合起来，控制层在进行决策时没考虑到数据层的已有策略。

（3）大多数的研究，对网络中资源的分配是静态的，一旦分配好就不再改变，这种方法对网络资源的管理缺乏一定的灵活性，不能动态的适应网络变比如链路损坏，掉线等等，也没有充分发挥出SDN网络集中控制的优点。

1.3 论文的主要研究内容及结构安排

1.3.1 论文的主要内容

随着互联网的飞速发展，流媒体类业务在网络中所占的比重正在急剧增大，它们对端到端的QoS有着严格的要求，然而传统数据报网络并不能很好的保证端到端的QoS。同时由于网络中业务种类繁多，其他业务类型的数据对QoS也有需求，且需求各不相同。针对这种情况，需要对业务分类并设置不同的优先级，然后对不同优先级的业务进行差别的传输，尽力满足不同业务QoS需求。但是传统IP网络在网络拥塞的情况下，并不能区分对待不同种类的业务来保证不同的QoS需求。针对上述问题，本文充分利用SDN的优点，研究SDN中的队列调度、拥塞控制和QoS路由，提出并实现了区分业务优先级的QoS控制策略，旨在优先保障视频流媒体的服务质量，同时又兼顾其他不同业务的QoS。现将本文的研究内容归纳如下：

一是依据网络传输业务所需要的QoS要求高低，对业务进行了优先级的区分，并设定了不同的优先级值。

二是从控制层和数据层制定整套视频流媒体的QoS保障策略。控制层采用QoS路由策略，它通过基于遗传算法的QoS路由满足了最高优先级业务的QoS需求，而对其它等级业务使用Dijstra算法基于跳数的路由。数据层的队列调度，它能对不同优先级的进行区分调度，保障了视频流媒体优先传输的同时提供带宽充足时的借带宽机制，尽量满足其他业务的QoS。

三是完成了QoS控制策略中的各个功能模块。一方面在控制器编写程序实现了基于遗传算法的QoS路由，较好的保障了视频流媒体的QoS。另一方面，在OpenFlow交换机上进行队列的配置，通过控制器下发指令将数据包送入配置好的队列中，实现了不同等级业务数据包的差别调度，在转发端口进一步保证视频流媒体的QoS。

四是搭建软件定义网络模拟实验平台，对QoS控制策略进行测试。

1.3.2 论文的结构安排

本文的主要工作是在SDN网络中，利用SDN网络的优势，从控制层和转发层分别制定控制策略，旨在满足视频流媒体业务端到端的QoS需求，同时尽最大可能满足其他类型业务QoS需求。围绕着论文的主要工作，本文对内容结构做如下安排：

第一章，绪论。论述课题的研究背景和意义，综合分析了国内外SDN以及视频流媒体QoS控制策略的研究现状，并对本文的主要研究内容给予概述。

第二章，SDN中QoS的相关技术分析。对QoS以及SDN网络进行了概述，以及对传统网络下流媒体传输遇到的问题进行了分析总结。

第三章，QoS控制策略分析与设计。针对视频流媒体传输的特点，提出了基于业务优先级的QoS控制策略，分别对QoS路由和交换机端口队列调度策略进行了设计。

第四章，QoS控制策略实现。根据第三章提出的控制策略，分别对QoS路由和队列调度策略进行实现。包括链路信息获取、基于遗传算法的QoS路由、HTB队列规则的配置。

第五章，实验仿真与结果分析。搭建网络模拟环境及视频传输平台，对QoS控制策略框架进行测试。

第六章，总结与展望。对论文的整体研究工作、取得的成绩以及需要改进的地方进行了总结，并对未来工作出展望。

2 SDN中QoS的相关技术分析

2.1 QoS技术

2.1.1 QoS的定义

服务质量（Quality of Service，QoS）是指网络利用各种技术保障网络时延和吞吐量等性能的服务要求，为某种业务提供提供优质服务的的一种机制[24]。在通信中，QoS 主要通过各种性能参数体现，其中常见的几种QoS性能指标有吞吐量、丢包率、时延 [25]。

吞吐量：指数据的传输速度，即单位时间内成功传输的数据大小，常用单位 M/s。

丢包率：指数据包从发送端传输给接收端的过程中丢失的数据包个数与发送的数据包总数的比值。网络丢包一方面会使用户的体验下降，另一方面大量丢包造成的数据重传会加大网络的负担。

时延：指数据从发送到接收所用的时间，常用单位 ms。由于网络的复杂性，造成时延的原因有很多，例如，数据传输到对端会经过一段时间，这是传输时延；数据到达交换机时，经过一系列处理才能从端口发出，这是处理时延；数据到达交换机

的出端口时，往往不能立即从端口出去，要经过排队，才能出端口，这是排队时延。

2.1.2 QoS服务模型及机制

IETF提出了许多QoS服务模型和机制，包括尽力而为服务模型（Best-Effort）、综合服务模型（IntServ）、区分服务模型（DiffServ）、多协议标签交换（MPLS）以及QoS路由。

Best-Effort模型不能提供任何QoS的保证，对数据包采取先来先服务策略，采取尾丢弃的方式解决网络的拥塞。

IntServ模型保障业务的传输是通过为特定业务预留充足的带宽 [26]。但预留带宽需要网络中的每台设备都能实现预留带宽这一功能，从扩展性方面来看，相对较差。

DiffServ模型在业务流进入网络边缘设备时就执行分类和标记的操作，通过将不同的业务流映照到设定的QoS等级值上实现对业务流的分类和标记[27-28]，网络中的其他设备根据已有的QoS标记值和分类完成数据的转发。因此只需要边缘网络设备具备数据标记和分类等功能就能实现业务的分类转发。DiffServ对 IntServ 存在的扩展性差的问题给予了解决，但其架构中缺少端到端的通信协议，对于保障端到端的QoS还存在一定的问题。

MPLS对数据的转发不是基于目的IP地址而是利用了标签交换技术[29]。应用MPLS能解决网络中的流量工程问题，即将流量分到多条空闲的路径中进行传输以进行链路的负载平衡。

QoS路由能为进入网络中的业务提供满足其QoS（时延、带宽等）需求的传输路径，同时能有效整合网络中的资源。

2.2 传统网络下的流媒体传输分析

传统网络在传输流媒体数据时主要存在以下几个问题：一是流媒体数据量远超出普通数据，对于已臃肿不堪的网络来说，传输大量的流媒体资源会增加网络的负担，为了解决流媒体在网络中的传输问题，在传统网络中加入更多的协议及功能，致使网络变得更加复杂，这样的恶性循环使得网络会有崩塌的危险；二是处理媒体数据要比普通数据产生更多的开销，除了要进行编解码，客户端播放，还要运行网络状态检测机制和拥塞控制机制，这些额外的操作都会对网络设备产生影响；三是现在的网络结构不能保证有效的QoS，因为尽管许多QoS模型和机制能在一定程度上实现网络层的QoS，但由于向网络设备中引入了很多复杂的功能，导致更换网络设备的成本较高，较好地实现端到端QoS的目标依然任重道远。

若流媒体视频流在网络中传输时不满足最低的QoS要求，会出现以下情况：若有视频数据帧延迟，可能造成同一幅画面上各点的时间不同步，视觉上看，物体都是变形的。视频流在传输时，若产生数据丢失，导致同一幅画面各个区域视觉效果不同，有的地方清晰，有的趋于清晰，有的区域模糊。视频传输时产生的延迟比较大，使得画面抖动比较厉害，效果很差。

2.3 SDN相关技术

如何解决传统网络存在的瓶颈问题，学术界一直都是各种声音此起彼伏，一些学者认为现有网络架构已经非常成熟，设计并应用全新的网络架构并不现实；一些学者则认为现有网络架构的弊病实在太多，要改变现状必须设计新的网络架构，以便能够全面提升网络的性能，为此设计出一些新的网络架构，其中内容中心网络（Content-Centric Networking ，CCN）[30]和SDN网络的研究较为热门。

指　标
疑似剽窃文字表述

1. 保障其他业务流的QoS。
   最后，对QoS控制框架的拓扑管理模块、链路信息测量模块、
2. 在SDN网络中，若要实现特定的网络需求，只需根据该业务需求去修改控制层的配置，便可以改变转发层网络设备的行为。
3. 网络中所占的比重正在急剧增大，它们对端到端的QoS有着严格的要求，然而传统数据报网络并不能很好的保证端到端的QoS。
4. 网络中业务种类繁多，其他业务类型的数据对QoS也有需求，且需求各不相同。针对这种情况，需要对业务
5. 差别的传输，尽力满足不同业务QoS需求。但是传统IP网络在网络拥塞的情况下，并不能区分对待不同种类的业务来保证不同的QoS需求。针对上述问题，本文充分利用SDN的优点，研究SDN中的队列调度、拥塞控制和QoS路由，提出并实现了区分业务优先级的QoS控制策略，
6. 依据网络传输业务所需要的QoS要求高低，对业务进行了优先级的区分，并设定了不同的优先级值。
7. 视频数据帧延迟，可能造成同一幅画面上各点的时间不同步，视觉上看，物体都是变形的。视频流在传输时，若产生数据丢失，导致同一幅画面各个区域视觉效果不同，有的地方清晰，有的趋于清晰，有的区域模糊。视频传输时产生的延迟比较大，使得画面抖动比较厉害，效果很差。

2. S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第2部分	总字数：9194
相似文献列表　文字复制比：32%(2938)　疑似剽窃观点：(0)	
1　4-杨俊超	11.2%（1027）
杨俊超 - 《学术论文联合比对库》- 2014-12-20	是否引证：是
2　S312060111+杨俊超	11.2%（1027）
杨俊超 - 《学术论文联合比对库》- 2014-12-26	是否引证：否

3	基于OpenFlow网络的QoS管理策略研究	10.8%（992）
	杨俊超(导师：孙建国) -《哈尔滨工程大学硕士论文》- 2015-03-01	是否引证：是
4	基于SDN数据中心的流量调度算法研究	5.9%（547）
	雷鸣(导师：李静) -《西安工业大学硕士论文》- 2018-05-15	是否引证：是
5	基于OpenFlow的SDN网络QoS路由策略研究	5.9%（545）
	周怡(导师：苏俭) -《电子科技大学硕士论文》- 2018-04-10	是否引证：是
6	基于主题模型_池悦	3.7%（336）
	池悦 -《学术论文联合比对库》- 2016-02-24	是否引证：否
7	池悦	2.9%（271）
	-《学术论文联合比对库》- 2016-09-13	是否引证：否
8	4903_池悦_计算机技术	2.6%（242）
	池悦 -《学术论文联合比对库》- 2016-05-10	是否引证：否
9	基于OpenFlow的QoS系统设计与实现	2.1%（190）
	曾福山(导师：梁阿磊) -《上海交通大学硕士论文》- 2014-01-01	是否引证：否
10	基于SDN的QoS研究	1.7%（156）
	周飞;吕光宏; -《计算机技术与发展（优先出版）》- 2017-12-05 12:24	是否引证：否
11	基于SDN的QoS研究	1.3%（121）
	周飞;吕光宏; -《计算机技术与发展》- 2017-12-05 1	是否引证：否
12	SDN网络业务量工程技术研究	0.8%（71）
	文强(导师：章小宁) -《电子科技大学硕士论文》- 2016-05-03	是否引证：是
13	201321010321+文强+SDN网络业务量工程技术研究	0.8%（71）
	文强 -《学术论文联合比对库》- 2016-03-28	是否引证：否
14	熊炎_星地融合光网络关键技术研究	0.8%（71）
	熊炎 -《学术论文联合比对库》- 2016-03-27	是否引证：否
15	SA13006085_张文_1	0.4%（39）
	张文 -《学术论文联合比对库》- 2016-05-06	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

CCN网络请求数据时直接以内容的名字进行请求而不再是通过IP，网络中有很多节点对内容进行缓存，而SDN 则是将网络的控制层和数据层分离，控制层和数据层不再是一个耦合的整体。与CCN需要替换网络层相比，SDN对网络的变革要小一些，相对也更容易实现，本文以 SDN 网络为研究基础。

OpenFlow概念出自斯坦福大学Nick McKeown教授主导的Clean Slate项目，最初是作为校园的实验平台产生的，之后SDN的概念应运而生，Mckeown 等设计者们开始推广SDN概念[31]，并引起业内外广泛关注。

图2.1 传统分布式结构与SDN架构对比图

如图2.1所示，传统网络架构的控制层和转发层是紧紧耦合在一起的，而软件定义网络的控制与转发是分开的，由控制层控制转发层中所有网络设备的行为。当前通用的SDN架构由ONF提出，由基础设施层、控制层和应用层组成[32]，如图2.2所示。

图2.2 SDN架构图

控制层是控制平面的抽象，其是SDN网络的核心，控制层的控制器负责收集网络信息并作出控制决策。基础设施层是数据转发平面的抽象，位于该层的转发设备只负责根据控制器的指令传输数据，这与传统网络设备是非常不同的，该层的转发设备除了可以是硬件设备，还可以以软件的形式存在，如OpenvSwitch。应用层是应用平面的抽象，其通过北向接口，通过控制层的控制器，向基础设施层的网络设备发出控制指令以及进行相关的管理工作。南向接口是控制器和底层网络设备间信息交换的通道，通过该接口，控制器可以获取数据平面的运行信息，并对数据平面的设备进行配置。北向接口用于控制器与上层应用程序的交互，对网络进行了抽象，这样网络管理者可以不用关心底层网络的细节，通过该接口，进行简单的编程就可以控制网络的运行逻辑，从而方便地管理网络。

SDN的体系架构具有两个特征：控制平面与数据转发平面解耦合，控制平面的易编程性。转控分离的思想使转发设备从繁重的工作中解脱出来，专注于数据的传输，相应的控制决策行为交给控制器来完成。同时SDN出现伊始就被定义为开放性的架构平台，将自由可编程引入控制平面，为简化网络配置、提高网络的性能、鼓励网络应用创新带来了巨大的便利性。

2.3.1 SDN控制器

SDN控制器位于控制层，由上文介绍可知，其是SDN网络的核心。向上，控制器能为上层应用提供底层设备各类信息；向下，控制器能向底层设备发送一系列指令实现网络的动态管理。随着SDN在网络领域逐渐展露头角以及控制器核心作用的凸显，业界已发布了多款控制器，目前已有超过20种控制器的实现，一些常见的控制器如表2.1所示[33]。

表2.1 常见的控制器

控制器名称实现语言简要介绍

NOX C++/Python 第一个SDN控制器

POX Python NOX的Python实现

Beacon Java 模块化设计，易开发新的应用

FloodLight Java 企业级控制器，脱胎于Beacon

Ryu Python 能与OpenStack平台整合，易于开发

Meastro Java 跨平台，易部署，支持多线程

NodeFlow Javascript 极简化的控制器，用于编写可扩展的应用

Ryu是由日本NTT公司贡献的基于组件的开源SDN框架，组件是以一个或多个线程的形式存在，通过组件提供的接口可方便控制组件状态和事件产生；提供图形用户界面便于掌握网络拓扑的连接情况，模块清晰，可扩展性好；采用语法简洁的python语言开发，易于学习和部署新的功能；能较好地支持OpenFlow1.0版本到1.5版本 [34]。本文所实现的QoS路由功能就是在Ryu中编写相应模块实现的。

2.3.2 OpenFlow

OpenFlow协议对转发设备的流表以及SDN控制器和转发设备之间的交互消息进行了规范的定义。

1）OpenFlow交换机

OpenFlow交换机由OpenFlow通道以及一个或多个流表、组表构成。OpenFlow通道是控制器与交换机之间进行通信的关键，其影响到通信能否可靠和安全，可靠性和安全性由使用TCP连接和安全传输层协议来实现。流表及组表是数据转发的指引者，当数据进入到OpenFlow交换机，若能匹配上交换机中存在的流表，就按照流表的指示执行相应的动作。图2.3是OpenFlow1.3定义的OpenFlow交换机的结构。

图2.3 OpenFlow交换机网络结构

OpenFlow的流表与传统网络中的MAC地址转发表和IP路由表类似，都是用于指引数据的传输，但流表项的匹配域中包含了网络一层到四层的各个属性，对网络中的匹配属性的整合使得转发设备对网络中数据流的匹配更加精确，对数据流的操作也就更加细粒度。交换机中每个流表都由多个流表项组成，每个流表项都由多个字段构成，例如匹配域、优先级以及超时等字段，OpenFlow1.3规定的流表项的结构如图2.4所示。

图2.4 流表项结构图

匹配域是用来判断数据包与流表项是否匹配，从图2.4可知由16个字段组成，涵盖了1-4层的各个匹配项，匹配规则灵活，对于每个元组，可匹配精确值，亦可匹配任意值，更可匹配某一范围内的值。优先级定义了流表被匹配的顺序，OpenFlow定义中规定该值越高，优先级越高。计数器（counters）的作用是维护一些统计信息，比如某端口的接收与发送字节数或某个流的匹配数据包数等信息。

超时主要用于流表项的超时，包括idle timeout和hard timeout，分别表示流表项的活跃时间和硬超时时间。Cookie用于控制器对数据流进行操作，包括流删除、流改变等操作。

指令表示匹配的数据包该执行的动作，指令主要有转发（Forward）、丢弃（Drop）、排队（Enqueue）、修改域等指令，一个流表项可以有零个到多个指令，如果有多个指令则可以规定每个指令的执行顺序，如果在这些指令里不含转发，则和这个流表项匹配的数据流就会被丢弃。

转发的行为有多种，其中较为重要的有 IN_PORT、CONTROLLER、ALL和NORMAL。分别表示从入端口转发出去，转发给控制器，转发给除入口端外的其余端口，利用传统转发机制如MAC地址和IP 等信息转发数据包。

丢弃（Drop）：若进入交换机中的数据包未能匹配上任何流表，且没有设置 table-missing 项或table-missing 项的动作设置的是丢弃，则会被丢弃。

排队（Enqueue）：根据队列ID的设置，把数据包放到出端口指定队列ID中，此项操作与队列调度策略有关。

修改域（Modify-Field）可选的修改字段有很多，例如，对IP报文头中的的ToS（服务类型）字段进行修改，修改成自己设定的值，该操作可用于支持QoS服务（ToS 的前六位为 DSCP，后两位目前保留）；修改源、目的IP地址，将其值用指定的数值替换；对 VLAN的优先级进行设置，若被设置的数据包之前没有VLAN首部，则会被加上一个新的首部，设置其优先级，并将VLAN ID 置零，若数据包头已有VLAN首部，将原先的值替换即可；去除VLAN首部，若数据包中存在VLAN首部，则将其去除。

OpenFlow交换机中，对数据的所有操作都通过流表来定义，交换机只是简单地根据流表的指令进行数据的转发或者丢弃，OpenFlow1.3采用多级流表的结构，交换机里存在控制器下发的多个流表。OpenFlow交换机对数据流的处理过程如图2.5所示，当交换机接收到外部发来的数据包后，首先对数据包头信息进行解析；之后按着流表的优先级顺序依次进行匹配，若匹配到某个流表中的流表项，就执行该流表项的动作；若数据包在交换机中匹配失败，则根据设定的规则进行操作，或是丢弃该数据包，或是携带数据包信息转发到控制器上，由控制器决策如何处理该数据包。

图2.5 流表匹配流程图

2）OpenFlow协议

OpenFlow协议运行在OpenFlow交换机中，其支持三种消息类型用以实现控制器和交换机之间的信息交换，包括控制器-交换机（Controller-Switch）消息、异步消息（Asynchronous）以及同步消息（Symmetric）。

Controller-Switch消息是控制器对交换机状态进行查询的消息，通过发送相应的Request消息，对收到的回复消息进行分析，根据分析得到的结果能获得交换机的状态信息。例如，通过modify-state 消息查询交换机端口状态（port-mod）和流表项

（flow-mod）等信息。

Asynchronous消息用于交换机向控制器告知网络的变化，不需要控制器必须回复。Asynchronous消息包括packet-in和ports-status等消息，进入交换机中的数据包未能匹配上任何流表，就会触发packet-in消息携带数据流首个数据包的部分信息或是携带整个数据包的信息发送到控制器，控制器通过分析数据包的信息，得到数据流的整体信息，作出相应的控制决策。当网络中加入新的交换机或原有的交换机断开连接时，交换机会向控制器发送ports-status消息，控制器得到该消息后对底层网络拓扑连接信息进行重新整合，得到最新的拓扑连接信息。

同步消息用于确认控制器和交换机的连接状态，其发送方可以是控制器也可以是交换机。例如通过hello消息确认是否连接上了，通过echo消息确认连接的状态。

2.3.3 OpenFlow对QoS的支持

QoS的控制机制是指根据QoS流的状态，对其进行实时的控制。前文提到SDN与传统网络不同的特点，其可以将复杂的QoS控制逻辑在控制平面实现，进行全局性的调度。

OpenFlow1.0有一个可选动作enqueue（1.1版本后叫set queue），其能将数据包放入到指定的交换机队列中。

OpenFlow1.1中添加了VLAN等操作，在1.1版本之后，能够通过对VLAN标签的添加、修改与去除操作实现QoS控制。

OpenFlow1.2协议中添加一个消息，使得控制器可以对交换机中的所有队列进行查询。

OpenFlow1.3中使用Meter引入新的限速机制，Meter表由多个Meter表项组成，结构如图2.6所示。其中Rate值与限速机制密切相关，其值若大于Meter的速率，Meter就会采取相应的措施，要么将超过Rate值的数据丢弃，要么对DSCP重新标记来对优先级进行重新的排列。

图2.6 Meter表项组成图

Open Flow 1.4：该版本提出了流量监控框架，允许控制器实时监控其他控制器对流表的任何子集所做的更改。

Open Flow 1.5：将Meter 指令改为Meter动作。这个改动使得多个meter可以属于一条流表项以及被使用到组桶中。

软件定义网络架构在QoS路由上相对于传统网络具有明显的优势，包括能获得全局信息，并能根据全局信息为业务流计算符合其传输需求的路径；周期地对网络进行监测，使得决策都是在最新的数据下做的，时效性非常好；能够对网络中的每一条流进行控制，相比传统网络，更为细粒度。

2.3.4 智能算法在SDN上运行的可行性分析

在SDN中，采用启发式智能算法解决网络中的流量问题是一种可行并且有效的选择。对于路径选择这种NP问题，一般的智能算法都可以进行优化解决，虽然每种智能算法进行优化计算的方式不同，但是都需要收集部分或者整个系统的信息，通过计算出最优解再进行资源的重新部署安排。因此这些算法都需要能够持续地收集到网络全局的实时信息，并且算法部署实施的代价比较小。而SDN的特性能够比较好的满足以上的要求，使得智能算法在SDN网络中运行成为可行的。

SDN集中控制的特点使其在收集网络底层信息时非常的高效和准确。控制器通过专用的控制链路连接交换机，可以收集全局的状态信息，并且可以简单高效地给底层交换机部署流表指令。

SDN控制器收集到的实时性信息十分丰富，因为SDN是基于流分析的，OpenFlow协议所规范的十六元组基本上囊括了数据流的基本信息，通过不同策略的流分析，能够得到当前流的详细信息，进而对流进行不同的处理。控制器还可以收集整个系统运行时的状态信息，这些信息为智能算法的计算提供了很好的参考数据，使算法做出的优化选择更加智能。并且，通过OpenFlow协议，控制器对流的操作控制可以进行多粒度处理，为算法提供的可操作项也更多，例如控制器检测到传输路径出现拥塞时，能够根据周期获得的网络信息为数据流提供动态路由的机制。

SDN提供了开放的可编程接口，开发者可以利用这些接口编写扩展出自定义的应用或者模块，例如可以在控制器内增加额外的链路性能测量模块，获取网络信息；编写部署客制化的应用，将这些策略或优化结果部署到网络中去。

2.4 本章小结

本章首先对QoS技术进行了分析，主要是QoS概念以及QoS服务模型及机制的介绍。之后对传统网络下视频流媒体传输遇到的困境进行了分析。最后对SDN技术进行了分析，主要是对SDN网络的运行机制进行阐述以及对智能算法在SDN上运行的可行性进行了分析。下一章将以本章内容为基础展开，借助SDN网络的优势，提高视频流媒体的传输性能。

3 QoS控制策略需求分析与设计

3.1 需求分析

3.1.1 应用环境分析

根据SDN的三层架构体系，任何应用都是以SDN控制平台为中心并在其上扩展相关模块而实现的，SDN控制器的性能将直接影响应用的实现效果，因此有必要对控制器的性能进行研究分析。SDN控制器是上层应用和下层网络设备之间通信的桥梁，它能够为应用层提供转发平面的基本信息，包括网络拓扑连接以及底层设备资源使用情况等；它能够将应用发出的指令发送给转发平面，控制转发设备中数据的传输。

本文的QoS路由功能是在控制器中编写相应模块来实现，为了实现这些模块功能，除了要调用控制器中原有的一些基本模块，如协议适配以及事件机制等模块；还需要在控制器上扩展相应的模块来实现本文所需要的功能，比如拓扑探测、链路性能信息获取等模块；同时，也需要注意自定义模块与控制器原有模块之间的调用顺序以及它们的协同工作。队列调度功能属于底层应用，摈弃其固有的先进先出的队列模式，采用新的队列算法，使视频流优先出队列，既能够更好地保证流媒体视频流的QoS，也能兼顾其他业务等级流的QoS，因此需要注意OpenFlow流表中关于队列的操作及OpenFlow交换机上队列规则的配

置。

3.1.2 应用需求分析

实现QoS控制策略方案的目的是为了保证视频流媒体的服务质量，同时兼顾其他等级数据流的传输。本文的控制策略包括控制层面与数据转发层面两方面，控制层的策略是当有数据初进入网络时，对于视频流媒体，根据全局的网络拓扑和所监测的网络链路信息，使用基于遗传算法的QoS路由算法进行路由，对于其他优先级的业务流则根据经过的交换机跳数选择传输的路径。当数据流在网络中传输时，控制器对数据流的传输路径进行实时的监控以便能获得传输路径的状态，当监测到传输路径前方发生拥塞时，及时为视频流采取动态重路由的策略。数据转发层的策略是根据控制器标记的流类型和交换机上配置好的队列调度策略，将业务流放入到端口不同的队列中，优先调度视频流媒体，使其能得到更好的QoS保障。综上所述，控制策略整体的需求包括链路状态信息的获取、视频流传输路径的计算、其他业务流传输路径的计算、重路由路径的选择、不同类型业务流的划分以及队列调度策略等几个方面。因此需要考虑并解决以下问题：

1）、获取网络拓扑连接信息：控制策略的有效实施有赖于全网拓扑连接情况的准确获取，由于网络中的数据传输和拓扑连接都是动态的，需要实现对全网拓扑连接情况的动态获取，当有交换机或链路出现故障时，及时对网络拓扑结构进行整合更新，以确保路径选择时有实时的拓扑连接参考。

2）、收集链路性能信息：本文方案中需要选择符合链路时延和链路丢包率指标的路径进行数据的传输，为了使控制策略能够选取出正确的路径，需要设计相应的模块实时地监控并获取这些资源的使用情况。

3）、保证视频流的服务质量：为了满足视频流的服务质量，需要对数据流流进行划分，区别对待流媒体视频流和其他业务流，为视频流提供满足其个性化要求的最优路径，与此同时，也需要在端口队列调度的过程中对不同类型的业务流进行差别调度，保证视频流优先出端口。

4）、迅速恢复：网络拥塞会加大数据包传输的时延，影响数据的传输，所以动态路由策略需能够及时发现拥塞点并快速恢复数据的传输。且在下发流表时注意并解决新旧流表转换带来的时延。

3.2 QoS控制策略框架

3.2.1 数据流分类

实时视频会议、高清晰在线视频以及视频监控等流媒体业务对时延和丢包率等QoS性能参数有着较高的要求，为满足此类业务的传输需求，并能充分利用网络资源，必须将业务流按照优先级进行分类，以便能够对分类后的业务流进行有区别的转发控制。

为了能对不同种类业务数据包标记对应的优先级值，本文根据DiffServ模型中的DS域[35]完成对业务优先级值的存储。该模型的主要特点是对数据包头中的8位服务类型字段（ToS）进行了重新的定义-区分服务字段DS，设定不同的值表示不同的优先级以区分不同种类的业务流。目前只用到了ToS字段中的前6位，将其记为 DSCP，从理论上说DSCP值不同则服务优先级不同，这个值越大对应的服务优先级也越高。

根据IETF定义的服务类别并结合本文的研究内容，对各种应用的优先级分类以及DSCP值的设定如表3.1所示，Q1表示对QoS有最高需求的实时视频流媒体业务，Q2-Q4表示对QoS需求相对较低的应用。

表3.1 应用的QoS等级分类及其对应的DSCP值

应用类型协议举例	QoS服务等级	DSCP值
实时视频会议、视频监控	RTP,RTSP	Q1	100000（32）
关键数据业务	NFC,SMB,RPC	Q2	011000（24）
事物处理，交互式数据	SQL	Q3	010000（16）
数据同步，大块数据，e-mail	FTP,SMTP	Q4	001000（8）

通常根据数据包头中的源目的IP地址，协议字段以及源、目的端口号等字段对流进行分类，并使用特定的DSCP值对分类的结果进行重新标记。用这种方法能区分大多数的业务，如果想区分的更细可以做包解析，但复杂度较高，故一般采用5元组，或更多维的包头数据直接匹配，本文使用的也是这个方法来对流进行分类。

与在传统网络结构中应用DiffServ网络模型不同，SDN转控分离的思想使得数据包的标记（标记不同的数值以区分不同类别的业务）并不由交换机来完成，而是依靠控制器的流表项中的修改域指令，修改域指令能够将IP报文头中ToS字段的DSCP值进行修改，修改成自己设定的值。在控制器下发的流表项中设置DSCP值修改的规则，规定匹配到该流表项的数据流的DSCP值作出对应的更改。DSCP值的修改动作需要在边缘交换机上完成，进入网络中的数据包在边缘交换机上完成DSCP值修改的操作，使得其他交换机在数据流转发的过程中能够根据数据包头中的DSCP值进行相应的流分类和标记操作。

3.2.2 QoS控制策略的架构设计

由上节介绍可知，本文依据业务类型，将数据流分为四个不同的等级，视频流媒体业务对时延等QoS参数有着非常严格的要求，因此优先级最高，其他业务的优先级则依次降低。为了在多种数据流并存的网络中满足视频流媒体的QoS，并减少其对其他等级业务流的影响，本文利用SDN控制器能获取全局网络拓扑信息并能集中控制OpenFlow交换机业务数据流转发行为的特性，以及OpenFlow协议提供队列服务的特点，提出了视频流媒体的QoS控制策略的框架。该QoS控制策略架构能够满足视频流媒体端到端的服务，并能在网络出现拥堵时及时调整传输路径，减少拥堵造成的视频流媒体传输性能变差的问题。

QoS控制架构如图3.1所示，主要由QoS路由与队列调度两个部分组成，这两个部分在SDN控制器与OpenFlow交换机都有着对应的模块，各个模块的功能及设计在下一节进行详细的阐述。

图3.1 QoS控制架构图

图3.2为QoS控制框架执行流程，进入交换机中的业务流没能匹配上交换机中存在的可指导其转发的流表项，就会将业务流信息封装到packet-in消息发送给控制器，控制器通过解析数据包的头部信息，对视频流媒体数据流按照OpenFlow控制器所规定的QoS约束条件，依据全局的网络拓扑和所监测到的网络链路性能信息，基于遗传算法进行QoS路由，对其他优先级业务则使用以路径跳数为代价的Dijkstra 算法进行路由。之后下发流表并修改ToS字段的DSCP值以及将数据包进行入队操作，然后使用HTB队列规定对OpenFlow交换机出端口队列中的数据包进行调度。

图3.2 QoS控制框架执行流程

3.3 QoS控制框架模块分析设计

QoS控制框架分为QoS路由和队列调度两个部分，下面分别对这两部分的设计进行阐述。

3.3.1 QoS路由框架

QoS路由旨在利用SDN的可编程性以及可获得全局网络信息的特点，基于遗传算法为视频流媒体计算满足其QoS需求的路径，并能动态地保障其QoS。QoS路由框架包括拓扑管理模块、链路性能测量模块、路由管理模块。QoS路由框架的设计是在控制器原有功能的基础上，充分利用了SDN网络中控制平面和数据平面可以通过OpenFlow协议进行自由灵活交互的特点。

1）拓扑管理模块

控制器中的拓扑管理模块能够主动对网络拓扑进行探测，以获取网络中交换机的物理地址以及端口号等信息，并能够周期性地获取网络设备间的链路连接情况，主要是数据平面中交换机与交换机之间的链路，还包括交换机与主机的链路以及控制器和交换机之间的连接情况。同时会对网络链路的变化进行监测，保证模块维护的全网络拓扑连接信息是最新的，为控制器在作出控制决策前提供精准的底层网络拓扑连接信息。

另一方面，模块会从链路性能测量模块获取每条链路的时延、丢包率以及带宽等性能信息。得到当前网络链路的物理连接信息和链路对应的性能参数信息后，根据这些信息拓扑管理模块对链路进行抽象封装，因为路由问题原型是图论中的寻路问题，这里将网络全局信息映射为一个邻接链表形式表达的带权图，从而方便后续QoS路由算法进行数据处理。

指　标

疑似剽窃文字表述

---

1. Open Flow 1.4：该版本提出了流量监控框架，允许控制器实时监控其他控制器对流表的任何子集所做的更改。
   Open Flow 1.5：将Meter 指令改为Meter动作。这个改动使得多个meter可以属于一条流表项以及被使用到组桶中。
2. 1 需求分析
   3.1.1 应用环境分析
   根据SDN的三层架构体系，任何应用都是以SDN控制平台
3. 进行研究分析。SDN控制器是上层应用和下层网络设备之间通信的桥梁，它能够为应用层提供转发平面的基本信息，包括网络拓扑连接以及底层设备资源使用情况等；它能够将应用发出的指令发送给转发平面，
4. 为了使控制策略能够选取出正确的路径，需要设计相应的模块实时地监控并获取这些资源的使用情况。
5. 迅速恢复：网络拥塞会加大数据包传输的时延，影响数据的传输，所以动态路由策略需能够及时发现拥塞点并快速恢复

3. S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第3部分

总字数：9621

相似文献列表　文字复制比：8.9%(856)　疑似剽窃观点：(0)

1	S312060111+杨俊超	4.4%（420）
	杨俊超 -《学术论文联合比对库》- 2014-12-26	是否引证：否
2	基于OpenFlow的SDN网络QoS路由策略研究	4.3%（410）
	周怡(导师：苏俭) -《电子科技大学硕士论文》- 2018-04-10	是否引证：是
3	4-杨俊超	4.1%（390）
	杨俊超 -《学术论文联合比对库》- 2014-12-20	是否引证：是
4	基于OpenFlow网络的QoS管理策略研究	3.5%（335）
	杨俊超(导师：孙建国) -《哈尔滨工程大学硕士论文》- 2015-03-01	是否引证：是
5	遗传算法在曲柄摇杆机构优化设计中的应用	0.4%（43）
	崔炜,张京军,宋德玉 -《河北建筑科技学院学报》- 2002-12-30	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

网络中物理的交换机映射为图中的顶点，交换机的连接关系映射为图中顶点之间连接的边，链路的性能参数映射为图中边不同的权值。通过这样的方式完成了对网络物理状态的抽象映射，形成抽象的详细网络全局资源视图，该视图是QoS路由计算模块的数据支撑。

拓扑管理模块中的拓扑连接信息的获取流程如图3.3所示，其中网络设备即网络中连接的交换机或者路由器；信息采集模块是对交换机信息进行统计（交换机 datapath ID，端口号 port，交换机间的链路连接）；信息分析模块是对信息采集模块获取的信息进行相关运算，计算网络的拓扑连接；状态轮询模块则是根据单独提出的子线程周期执行信息采集模块，在网络拓扑

由于交换机接入或离开造成拓扑连接发生变化时，对拓扑连接信息进行更新；拓扑图显示模块是根据信息分析模块计算得到的拓扑连接情况，将拓扑信息展示在终端。

图3.3 拓扑发现模块图

与传统网络一样，SDN网络中获取链路间的连接信息也是利用了链路层发现协议（Link Layer Discovery Protocol，LLDP），但与传统网络分布式的获取形式不同，SDN网络中拓扑信息的获取是由控制器统一完成的。LLDP帧的组成如图3.4所示。

图3.4 LLDP以太网帧

LLDP以太网帧中的LLDPDU为有效荷载，其能够存储设备的物理地址等信息。LLDP能够将设备标识以及物理地址等信息发送给隔壁网络设备，隔壁设备接收到对端发送的LLDP帧后进行解析即可得到存储在LLDPDU中的信息并将其保存下来。

控制器执行信息采集的过程如图3.5所示，为了精准掌握网络中交换机之间的连接情况，控制器构造含有LLDP数据包的packet-out消息，将此消息发送给网络中所有的交换机，packet-out消息中包含如何处理LLDP数据包的流表项。以s1和s2组成的链路为例，s1收到控制器发来packet-out消息后，通过packet-out中的流表规则处理LLDP数据包，流表项规则是s1将收到的控制器发来的LLDP数据包从非接收端口发送出去；s2的port1与s1的port1相连，因此LLDP数据包会从s1的port1转发出去，发送给s2的port1；收到LLDP数据包的s2无法处理该数据包，便会触发packet-in消息，其将携带LLDP数据包的信息到控制器；控制器收到packet-in消息后对数据进行解析，得到发送该LLDP报文的是s1的port1，接收该报文的是s2的port1，从而确定交换机s1与s2之间具体的连接情况，即s2的port1与s1相连，同理可知s1的port2与s2相连。依照这样的方式控制器获得网络中相连的两个交换机间的链路信息，进而得到整个网络的链路连接情况。

图3.5 链路连接信息采集图

2）链路性能测量模块

链路性能参数衡量了一条链路传输性能的好坏，QoS路由的前提是能及时获取链路的这些参数信息，然后通过这些链路组合出满足业务流QoS需求的转发路径。随着OpenFlow协议的不断更新，协议功能逐渐丰富，利用OpenFlow协议可以直接获取少量网络状态信息，但是链路的时延、带宽、丢包率等链路性能信息并不能由OpenFlow协议直接获得，因此本文对控制器功能进行扩展，添加了链路性能信息测量模快，从而高效地对链路的QoS性能参数进行测量。下文将对链路性能信息的测量原理进行具体阐述。

a、带宽使用情况及丢包率

OpenFlow协议定义了交换机中计数器统计的字段，其中对于交换机端口能够查询到的信息如图3.6所示。

图3.6 端口消息结构图

通过对交换机端口进行查询可以得到端口发送与接收的字节数，并能获取到端口发送与接收这些数据所经过的时间，对以上信息进行整合计算能够获取每个交换机的每个端口的带宽使用情况，之后通过将一条链路两端端口对应起来并分析计算，即能得到一条链路的带宽使用情况。

若控制器利用OFP_Port_Stats消息在t1时刻与t2时刻取得了s1与s2对应端口的（s1的port1与s2的port2相连）信息，对于交换机s1的port1，其t1时刻与t2时刻发送与接收的字节数如公式(3.1)和(3.2)，端口速率的计算如公式 (3.3)所示，speed单位为bit/s。

pt1=tx_bytest1+rx_bytest1 (3.1)

pt2=tx_bytest2+rx_bytest2 (3.2)

speed=pt2-pt1t2-t1×8 (3.3)

则交换机s1的port1的已用带宽和剩余带宽如公式(3.4)和(3.5)所示。

bwuse=speed (3.4)

freebw=curr-bwuse (3.5)

链路拥塞率是衡量业务数据流在网络中传输拥塞程度的指标，拥塞严重性随着拥塞率的值增大而增大。链路拥塞率的计算如公式(3.6)，其中speed106(speed106speed106speed106speed×8106为单位换算即bits/s-Mbit/s。

link_congestion =speed106÷curr (3.6)

与端口统计类似，OpenFlow定义了统计流信息的消息结构体 OFP_Flow_Stats， OFP_Flow_Stats能够获取交换机中匹配到某个流表项的流的统计信息，包括duration_sec、packet_count、byte_count和instructions等字段，分别表示统计的时刻、流表项匹配到的报文数、流表项匹配到的字节数和匹配到该流表项的数据包会被采取的动作。

链路丢包率的测量利用了OFP_Port_Stats消息中的rx_packets字段、OFP_Flow_Stats消息中的instructions字段和packet_count字段，利用OFP_Port_Stats消息得到某个端口接收到的报文总数，利用OFP_Flow_Stats消息获得当前交换机某个端口匹配到流表项发出的报文总数。对于s1与s2间的链路（s1的port1与s2的port2相连），利用OFP_Flow_Stats消息可得到t1时刻s1所有流表项统计的发送到port1的报文总数为tx_packets_byflowt1tx_packets_byflowt1tx_packets_byflowt1，同理t2时刻为tx_packets_byflowt2，t1时刻到t2时刻，s1传输给s2的报文数如公式(3.7)所示。

tx_packets=tx_packets_byflowt2-tx_packets_byflowt1 (3.7)

根据OFP_Port_Stats消息中的rx_packets字段得到t1时刻s2从port2端口接收到的报文总数rx_packetst1rx_packetst1，同理，t2时刻接收的报文总数rx_packetst2rx_packetst2，t1时刻到t2时刻，s2的port2实际接收的报文数如公式(3.8)所示。

Δrx_packets= rx_packetst2-rx_packetst1

rx_packets=rx_packetst2-rx_packetst1 (3.8)

丢包率pl的计算如公式(3.9)所示。

pl=1-rx_packetstx_packets (3.9)

b、时延

OpenFlow协议实现时延的测量是利用了在SDN网络中可在某些消息中添加时间戳的特点，通过按照对应的计算公式对时间戳进行计算，即可得到链路间的时延信息。本文根据这一特点，利用packet-out和packet-in等消息间的联系，使消息中携带时间戳，通过时延计算公式得到每条链路的时延信息。

时延测量的具体步骤如下：

（1）控制器经过交换机1、交换机2后再回到控制器的时延测量是利用了控制器为获得拓扑连接而下发的携带LLDP数据包的paxket-out消息。以图3.7两个交换机由控制器控制为例，使packet-out消息的data域携带有控制器发出消息的时间戳Tlldp_start，之后与获得拓扑连接的操作一样，当控制器收到packet-in消息后会记录此时的时间Tlldp_endTlldp_endTlldp_end，T1=Tlldp_end-Tlldp_startT=Tlldp_end-Tlldp_startTlldp_end-Tlldp_start即是controller-s1-s2-controller的大致时延，同理，controller-s2-s1-controller的时延T2计算方式也如此。

（2）控制器到交换机的往返时延测量是利用了请求回复消息Echo Request，使Echo Request消息的data域携带有控制器发出消息的时间戳Techo_start，控制器向s1发送Echo Request消息后，会收到s1的回复消息 Echo Reply，控制器收到Echo Reply消息后会记录此时的时间Techo_endTecho_endTecho_end，控制器与s1往返时间Trtt_1=Techo_end-Techo_start，同理，控制器与s2之间往返的时间Trtt_2计算方式也是如此。

（3）s1与s2之间的时延计算公式如(3.10)所示。

T=T1+T2-Trtt_1-Trtt_2÷2 (3.10)

图3.7 时延测量原理图

3）路由管理模块

路由管理模块负责网络的路径决策，是整个QoS路由控制机制中最核心的部分。路由管理模块有四个功能，一是数据流的区分，二是路径的计算，三是流表下发，四是网络拥塞时的动态路由。数据流的区分主要是区分出视频流媒体与其他业务流，进而给予不同的路由策略；路径的计算部分采用了双轨并行的方式，根据拓扑管理模块提供的全局网络拓扑结构以及带有链路性能参数的拓扑结构，对于视频流媒体，以其QoS需求为核心为其选择路径进行传输，对于其他低优先级业务，使用以跳数为代价的Dijstra算法进行路径的选择和传输。通过两种不同的路由方式，实现了不同优先级业务的差别路由，在控制层较好的保证了视频流媒体数据流端到端的QoS；流表下发是根据路径计算部分得到的链路路径建立流表信息并下发给OpenFlow交换机，OpenFlow交换机根据流表进行业务数据流的路由转发。

多QoS约束的视频流媒体的路由，实际上是满足多个指标的相对优解，对于单约束QoS路由问题，最短路径算法Dijstra算法即可解决[36]，多约束路由问题即同时满足两个以上相互独立参数的性能要求目前已被证明是NP-Complete问题[37]，很多研究将多种约束参数以相关运算的方式组合成一个混合的参数来解决多约束的问题，但这种方式有很大弊端，其一是运算的方式并不容易确定，其二是通过混合参数计算出来的路径也许不能较好地反应任何一个约束，计算出来的路径往往毫无意义。基于此，更多的研究寻求以更好的方式解决此类问题，运用近似算法或启发式算法求解多约束问题是众多方法中较为优越的方式，二者各有各的优势和劣势，近似算法能保证解质量较好，但计算却要花费较长的时间；启发式算法能在较短的时间内计算出解，却在保证解的质量上有所欠缺，常用的启发式算法有遗传算法[38]、蚁群算法[39]等。本文需要在传输路径不满足传输要求时，能够尽快地找到另一条可以传输的路径，因此本文研究采用启发式算法中的遗传算法来解决多QoS约束问题。

遗传算法模拟了生物在自然环境中的优胜劣汰，其是一种全局优化概率搜索算法[40-41]。遗传算法使用迭代的方式对一代种群模拟生物的进化过程，通过一系列遗传操作生成下一代群体，之后不断迭代，直到达到算法结束的条件，找到符合需求的最优解。本文将在下文论述如何设计遗传算法进行QoS路由的计算。

由于网络状态一直处于动态地变化中，路径计算部分计算出的路径在一定时间后可能不再满足视频流媒体传输的QoS需求，动态路由的功能是根据周期检测到的网络拥塞状况，在发现视频流媒体的传输路径出现拥塞时，为视频流媒体计算新的传输路径，然后将新的路径转换成流表下发到对应的交换机上，对于旧的流表项执行删除的动作。本文假定理想情况为触发动态路由从原有路径调整到其他路径时，已经发送的数据包不会重新发送，不再重传的数据流依然需要在拥塞的路径中传输，端口转发一直会使用HTB队列规则最大可能减缓拥塞，保证数据流的通过。动态路由部分涉及拥塞链路的判定及流表下发时路径切换的问题，下文为具体阐述。

（1）拥塞链路判断

本文的方案是应用于混合流量的网络中，即包含不同优先级的业务流，本文的主要目标是为视频流媒体提供最好的QoS保障，但是网络中的传输会出现其他业务流挤占视频流资源的情况，即传输路径出现拥挤，这个时候需要对视频流媒体的传输路径进行调整。本文设定的拥塞率为80%，即路径中的链路带宽的使用率为固定带宽的80%以上，即判定为拥塞，需要使用遗传算法为视频流重新计算一条合适的路径进行传输。

（2）路径切换机制

当触发系统使用动态路由机制为一条流重新计算路由之后，需要将新的路径流表下发到交换机中，使后续进入到网络中

的流匹配到新的流表，在新的路径中进行传输。这一过程虽然时间很短，却涉及新老流表项共存和路径切换过程中的丢包问题，对于数据的传输是非常重要的，因此需要设计路径切换机制来解决上述问题。

路径切换机制首先解决的是路径切换过程中的丢包问题，路径切换主要是指新流表的下发，而新流表的下发先要解决新流表的优先级问题，数据包在交换机中匹配流表的顺序是根据流表的优先级来确定的，优先级值越大，优先级越高。若新路径流表的优先级低于旧路径的流表优先级，则数据包匹配到的依然是旧的流表，传输的路径还是原来的路径，并没有改善数据流的传输情况；若新旧流表优先级一致，进入网络的数据包不知该匹配哪个流表项，数据包的传输会存在一定的风险；若新路径流表的优先级高于旧路径流表的优先级，进入交换机中的数据流会匹配新的流表，以新的路径进行传输。因此新流表优先级的设置定是要高于旧流表的优先级。

新流表优先级设定完毕后需要对流表的安装顺序进行设置。若以“源-目的”的顺序下发流表，则可能出现数据流到达某一交换机时而流表更新还没有完成的情况，此时数据包匹配不上交换机内任何的流表项，停在了此处而不能继续传输下去，影响数据端到端的传输。

基于“正向顺序”安装可能影响到数据的传输，采用“反向顺序”安装流表，即以“目的-源”的顺序依次下发流表，在新的流表安装完毕之前，不删除原有路径的流表，这样能够避免路径调整时的数据丢失，确保了数据传输的连续性。

如图3.8所示，假设某条流原来的传输路径为s1-s2，在T0时s1-s2链路发生拥塞，触动重路由机制为流重新选择传输路径为s1-s3-s2，若按照“正向顺序”写入流表，T1时刻s1上的流表已经重新成功，新进入到网络中的流依据s1中新的流表项进行匹配转发，即从s1的port2转发出去，根据流表项的动作，数据流已经从s1到s3了，但此时s3的流表项可能还没重写完成，就会造成数据流“不知路在何方”的情况出现，根据OpenFlow协议的规定，此时对数据包的处理有两种方式，一是将数据包直接丢弃，二是借助packet-in消息将数据包信息送到控制器，由控制器决定处理该数据包的方式。前者会导致大量数据包丢失，后者会致使短时间内大量packet-in消息送到控制器，影响控制器的运行。因此无论哪种方式，都会影响到数据的传输。

若采用“反向顺序”下发流表，即按照s2-s3-s1顺序下发新的流表，在最后一个交换机s1写入新流表之前，数据流仍在原来的路径进行传输，避免了“不知路在何方”的情况出现。

图3.8 流表安装图

采用主动删除旧流表项的方式解决新老流表共存的问题。选择此方式而没有使用流表项超时失效的方式是因为旧流表项已经没有任何用处，若旧流表项自然失效需要的时间较长，旧流表项将会长时间占据着流表项的资源，这是一种浪费。

3.3.2 队列调度策略的分析设计

OpenvSwitch虚拟交换机是以 FIFO（First In First Out，先进先出）的转发规则将到达出端口的数据包转发出去的。FIFO是一种 “尽力而为”的转发规则，谁先到达，谁先享受更多的资源。当某个优先级低的业务流的带宽需求较大且先到达转发端口，会导致后到达出端口但优先级高的业务能用的带宽较少，有可能造成高优先级的业务在短时间内无法得到转发的情况出现。因此，为保障流媒体视频流良好的传输，QoS控制策略在OpenvSwitch 虚拟交换机的出端口设计采用新的队列调度算法，保证流媒体视频流在每个转发节点上的QoS要求。根据OpenFlow协议能够提供队列配置的功能，即在流表中增加将数据流送入到端口队列的动作，数据流进入到交换机匹配到流表后，就会根据流表的指引进入出端口的某个队列中，之后根据队列规则的配置进行数据的转发。

1）队列调度分析

队列调度的规则有很多，比如FIFO 、PRIO、HTB等。其中，FIFO只能提供尽力而为的服务，不能对不同业务做区分处理，无法提供QoS区分服务；PRIO虽然可以对报文进行分类，使得高优先级业务报文得到优先调度，但容易使得低优先级队列中的业务被“饿死”；HTB队列规则既能够实现不同优先级业务的区分调度，也能实现流量整形和速率限制的需求。

本文队列调度的需求是为视频流媒体提供足够的带宽；带宽充足时，不同优先级的业务流间可以相互借带宽；端口提供不同业务流的优先级转发。根据需求分析以及综合对比多个队列调度规则的优缺点后，利用分层令牌桶（Hierarchical Token Bucket，HTB）实现队列调度的功能。

树状结构的HTB队列规定能实现对业务数据进行不同层次分类的功能[42]。其利用DWRR算法能按照业务优先级进行差额轮询调度的特点，将优先级不同的业务数据调度到相应的队列中中；利用令牌桶算法实现对流量限速和整形的功能。两种算法的简要介绍如以下内容。

a、DWRR算法

DWRR算法是在WRR算法基础上进行的优化调整，WRR算法则是对RR算法的优化[43]。因此先对RR算法和WRR算法进行简要的介绍，然后再阐述DWRR算法的原理。

轮循算法（Round Robin，RR）为相对简单的调度算法，其调度的原理是按照次序依次调度每个队列，当队列里没有数据包时则执行跳过的动作。尽管顺序的调度能在一定程度上显示算法的公平性，但顺次的调度却无法保证不同优先级业务的需求，无法提供区分调度的服务，导致不能在多种业务流并存的状态下满足较高优先级业务的QoS。

加权轮循算法（Weighted Round Robin，WRR）对RR算法的改进在于使用加权轮询报文的方式实现队列调度的功能[44]。其实现的原理是将每个队列都设置一个权值和计数器，调度伊始，队列中计数器的值都会被初始化为权值，之后每调度一个报文，该队列的计数器值会自动减1，若队列中报文个数或者计数器值为0时，将计数器的值重置为权值，然后轮询到下一个有效队列中继续执行调度操作。WRR算法规定没有报文的队列不进行调度，这样能将占有的带宽分给其他队列使用。WRR算法能实现在多个队列间灵便分配带宽的功能，为不同优先级业务提供不同的服务，图3.9为WRR算法原理图。

图3.9 WRR算法原理图

WRR加权轮询调度算法可以为不同类数据提供不同的优先级服务，但是WRR算法是基于数据包的，队列中数据包较多会使其得到更多的调度机会，对于其他队列来说是不公平的。为了解决WRR算法存在的这一问题，算法研究者提出了差额加权轮询（Deficit Weighted Round Robin，DWRR）算法。

DWRR算法对WRR算法的改善在于将队列权值的报文个数更改为报文的字节数，并为每个队列维护一个固定的权值 Q和一个差额 D。算法最初运行时需要对每个队列的Q值进行设置，并将每个队列的D值清零。在每一轮调度中，需要每个队列尽可能多地发送数据包，且发送的字节数少于等于 Q+D，并保证按序发送，若发送了N个数据包，则需满足Q+D大于等于这N个数据包长度，且这N个数据包长度加上这个队列下一个要发送的数据包长度需大于 Q+D，发送完毕后，若队列为空则把D值设置为0，不为空则 D=Q+D-send。send为该队列这一轮发送的字节数。DWRR算法可以为不同优先级业务提供更为优质的服务。

b、令牌桶算法

令牌桶算法的功能是能够对网络中的流量进行整形和限制流量速率的处理，可以对发送到网络中的数据数量进行限制并可允许突发数据的产生。图3.10为令牌桶算法的流程图。

图3.10 令牌桶算法流程图

令牌桶内以固定的速率产生令牌，当数据包通过令牌桶时，若桶中的令牌数量充足，则数据包可以被发送出去，同时桶中的令牌数被减去相应值；若桶中的令牌数不足，则数据包只能等到令牌桶中的令牌充足时，才能被发送出去。

令牌桶对突发数据的传输和传输速率的限制是通过令牌产生的速度和数据包的传输速率的比较来实现的。如果令牌产生的速率一直大于数据包的传输速率，就会使得令牌数不断增多直至达到桶的容量，然后就会开始溢出，之后若有突发的数据流经过令牌桶，桶中积攒的令牌即可处理这些突发的流量。如果令牌产生的速率一直小于数据包的传送速率，可能会出现令牌桶内的令牌消耗殆尽的情况，这将会限制数据包的发送速率，使其与令牌桶的产生速率相同，进而达到限制流量的目的。

2）队列调度策略的设计

图3.11展示了队列调度策略的设计，数据流进入交换机后，经过流表的匹配后执行流表项中的转发到端口、数据包标记以及入队列动作，不同优先级业务报文被标记并进入到不同的队列，通过OpenvSwitch上设置的HTB队列规定的调度后，视频流不仅可以获得更多的输出带宽，也可以优先被调度出端口，转发到下一个交换机上。

3.11 队列调度策略图

3.4 基于遗传算法的多QoS约束路由算法分析与设计

根据本文的应用场景，使用遗传算法解决多种约束的QoS路由问题，下文将对遗传算法进行设计来满足实际应用的需要。

3.4.1 路由度量数学模型

QoS控制指标参数分为可加性指标、乘性指标和凹性指标[45]，分别表示一条路径的 QoS为该路径每条链路QoS要求值的和、乘和最值。例如跳数和时延是可加性指标，丢包率是乘性指标，带宽为凹性指标中的最小值。

网络QoS路由选择问题的理论基础是图论中的寻路问题，通常将网络描述为有向图G (V,E) ，其中，V代表图中顶点的集合，用于描述网络中的Open Flow交换机或主机，E 代表网络中边的集合，描述连接各个网络设备的链路（控制器和交换机的链路不包括在此集合内）。每条边a,b∈E。

设有一条路径p,s∈V为路径的起点，d∈V-Sd∈V-s为路径的终点，根据以上定义的QoS控制指标参数的分类情况，可以得到路径p的传输性能参数表示为：

delayps,d=a,b∈ps,ddelaya,b (3.11)

lossps,d=1-a,b∈ps,d1-lossa,b (3.12)

bandwidthps,d=mina,b∈ps,dbandwidtha,b (3.13)

公式(3.11)为该条路径端到端时延，公式(3.12)为传输路径总的丢包率，公式(3.13)为传输路径中剩余的带宽。

对于视频流媒体的传输，需要找到这样一条路径ps,dp(s,d)，满足以下的约束条件：

delayps,d<D (3.14)

lossps,d<L (3.15)

bandwidthps,d>B (3.16)

其中D、L、B分别表示传输对时延、丢包率及带宽的要求，D的值为10ms，L的值为20×10-3，B的值为0.8Mbps。

3.4.2 遗传算法基本概念

遗传算法是基于自然选择和遗传进化学的自适应启发式随机搜索算法，虽然强调随机化，但是遗传算法并不是完全随机的，而是利用历史遗留信息将搜索范围引导到可能获得更优解的搜索空间中。遗传算法基本概念如下：

（1）种群：种群是算法待解决问题的所有可能解的编码方案的一个子集。

（2）染色体：一条染色体代表种群中的一个个体，每条染色体都可能是一个解。

（3）基因：基因是染色体一个位点上的具体元素。

（4）染色体编码：指种群中每个染色体内部基因的编码方式，用于建立实际问题解与计算机可以理解的数据结构之间的映射。

（5）适应度函数：适应度函数是用于对种群中染色体对于当前环境适应程度的一个量化函数，计算结果代表一个染色体的适应值，反映解空间一个解对当前问题解决程度的优劣。

（6）遗传算子：遗传算子是遗传算法中对个体执行进化、生成下一代群体的操作，包括选择算子等算子。

3.4.3 染色体编码设计

染色体编码是实际问题与计算机联系的桥梁，是遗传算法的基础和关键。合适的染色体编码对问题的解决有很大的助益，可以将实际问题空间直观地映射到具体的机器空间，同时又能兼顾计算机对算法执行的效率。常见的染色体编码有符号编码、二进制编码和基于树的表示。

由于路由经过的节点数目可以不同，即染色体的长度是变化的，QoS路由算法染色体编码选择变长符号编码的方式。由于每个交换机都有属于自己唯一的datapath.id标识，将一条路径上的每个交换机的datapath.id按照途经的顺序排列即完成了对一条染色体的编码，染色体第一个位置的基因代表路由的源节点，最后一个基因代表路由的目的节点，基因的排列顺序表示路由经过的网络节点的顺序。染色体长度不能超过网络中节点的总数，因为路由永远不需要比网络所有节点更多的节点来构成。以图3.12网络组成结构为例，使用箭头标出的路径用变长符号编码表示为[1，2，4，7](一条路径是一个染色体)。

图3.12 一种示例网络拓扑图

3.4.4 种群初始化方法设计

种群初始化是算法的第一步，执行初始化会产生一定数量的染色体，它们将构成算法运行的初始种群。通常种群初始化需要考虑三个要素：种群模型、初始化的方式、种群数量规模[46-47]。

遗传算法的种群模型分为稳态和代数两种方式。稳态模式中，算法每次迭代产生一或两个新的染色体，并用它们替换原种群中的一个或两个染色体，因此也被称为增量GA。代数模式中，每次迭代产生n个新的染色体，其中n就是种群的规模的值，然后用新的染色体集合替换原先整个种群。本文采用稳态模式作为QoS路由算法的种群模型。

初始种群生成通常有随机和启发两种方式，随机初始化用完全随机的方式来生成初始种群，启发式初始化利用实际问题的已有启发式方法填充初始种群。为增大QoS路由算法的搜索空间，采用随机初始化的方式对种群进行初始化。

种群规模指一代种群中个体的数量值，种群规模小会使算法的计算量小，能够缩短计算的时间，但同时也可能掉进局部最优的“圈套”，使解的质量有所下降；较大的种群规模，算法搜索空间相对较大，有助于寻找全局最优解，但是每代种群进化的计算量相应增大，计算量的增加会增加计算时间，进而会降低算法的时效性，为了遗传算法有最佳表现，种群规模的需要设置一个平衡的值，兼顾计算效率和解的合理性。针对本文的拓扑设计情况，将拓扑中所有路径的30%作为该拓扑的初始种群。

3.4.5 适应度函数设计

适应度用来判别一条染色体的好坏，其值根据适应度函数计算。无论哪个进化算子，都需要根据染色体的适应度进行相应的操作，因此适应度函数的设计成为遗传算法中的关键一步，其对算法收敛的速度以及能否找到合适的解有较大的影响。

由于带宽为凹性参数，不适于作为适应度函数，因此适应度函数的设计只考虑时延和丢包率。

染色体i的适应度计算函数如公式(3.17)所示，其中2为该适应度函数中包含的QoS测度个数。

fi=fdelayi+flossi÷2 (3.17)fdelayifdelayi是染色体 i 所表示的路径的时延适应度计算函数，如公式(3.18)，其中sumidelaysumi(delay)是染色体i所代表的路径的时延总和，n是一代种群中个体的数量。

fdelayi=sumidelay÷j=1nsumidelay (3.18)flossi是染色体 i 所表示的路径的丢包率适应度计算函数，如公式(3.20)，其中由于丢包率并不是加性参数，需要通过公式将其转换成加性参数，如公式(3.19)。

loss'a,b=log1+lossa,b (3.19)

flossi=sumi loss'÷j=1nsumi loss' (3.20)

时延与丢包率的适应度计算函数，都除以了该代所有染色体相应QoS参数和，这是因为从丢包率与时延的测量结果可知，丢包率值与时延值相差很多，若不做这样的处理，就不能均衡时延和丢包率，选择的路径可能更多的考虑了时延，而忽视了丢包率。做了相应的处理后取值范围在同一范围，被考虑的可能性就相对均等。

根据上述定义的公式可知，对于一条染色体的适应度来说。其值越小，表明该条染色体代表的路径更优秀。

3.4.6 遗传算子的设计

1）选择算子设计

选择算子与自然选择中的适者生存类似，适应能力强的个体留下来，较弱的个体被淘汰，选择算子根据个体的适应度值判断一个个体是否拥有较强的“生命力”。遗传算法中的选择算子所要实现的是把选择出来的优秀个体直接遗传到下一代或通过交叉算子产生新的子代个体再遗传到下一代。

根据本文适应度函数的设计可知，一条染色体所计算的适应度的值越小，表明该条染色体所代表的路径更符合要求。为了减小当前群体中的最优个体在下一代丢失的概率，提高遗传算法的收敛速度以及收敛到最优解时的稳定性，本文的遗传算子采用“精英选择(elitist selection or elitism)”策略，该策略对一代群体在进化过程中出现的精英个体不进行交叉操作而是直接遗传到下一代。算法的步骤如下：

（1）确定每次保留的精英染色体数量和随机保留的普通染色体数量。

（2）对于一代群体，按照适应度值对染色体进行由小到大的顺序排列，根据设定值从中选择出适应度较好的的精英染色体，直接复制到下一代。

（3）根据设定值再从剩余的染色体中随机选择普通染色体，复制到下一代。

（4) 将剩余的染色体执行交叉算子，并传入下一代。

2）交叉算子设计

交叉算子是对种群中的两个个体进行某些基因交换的操作（本文指的是交换机节点），交叉操作将会得到新的基因组合即新的个体，能够使算法的搜索空间扩展。交叉算子的设计一般包括交叉率的设置和交叉方式的设计，但本文的设计是在经过选择算子后，对剩下的所有染色中除头尾基因外有相同基因的父母亲染色体执行交叉操作，因此只设计交叉方式而不设置交叉率。

交叉算子中的交叉方式一般是通用的，但设计算法时，需要根据具体问题选择相适应的交叉方式。交叉方式包括单点交叉、多点交叉和融合交叉等方式，各有各的优缺点。比较常用的是单点交叉，其实现原理也比较简单，就是在染色体中随机选择一个点作为交叉点，然后在该点对染色体的部分基因进行交换，得到新的染色体序列，如图3.13所示。

图3.13 单点交叉示意图

QoS路由算法中交叉算子的设计借鉴了单点交叉的思想，将两条路由的部分路径片段进行交换，产生新的路由，具体是先从两条路由中选出一个节点，将路由分为两部分，一部分是从源节点到选出的节点，另一部分从选出的节点到目的节点，然后相互交换选出的节点后面的路由片段。

同时，路由算法中采用的单点交叉算子同上文提到的常规单点交叉又有些区别。本文算法要求双亲的染色体头尾基因必须相同，即源目的节点要相同，且双亲的染色体除了头尾基因外，必须包含至少一个相同的基因片段，但是并不要求该相同基因片段在染色体中的位置也必须相同。这些相同的基因片段都可作为交叉点，交叉算子以随机选择的方式在这些潜在交叉点中选择一个交叉点，以该交叉点为中心轴，将两条染色体的前后基因分别进行互换，便得到两条新的染色体序列，如图3.14所示。

图3.14 QoS路由单点交叉示意图

3）变异算子设计

变异算子能扩大基因的多样性，增大算法的搜索范围，在一定程度上能防止局部最优解的出现。变异算子的设计包括变异概率的设置和变异方式的设计。

变异概率决定了一个染色体发生变异的概率，遗传算法的变异率并没有明确的值，但有些变异概率确是比其他变异概率提供好很多的结果。较高的变异率能增加遗传多样性，扩大解的范围，避免计算出来的解是某个范围内的最优解而不是全局最优解的情况出现。然而，变异率过高会导致过多的遗传特性变异，可能导致失去前代种群中的优良解。最好的变异概率应该设置为这样一个值，该值允许足够的多样性，以防止算法搜索停止不前，同时算法在变异中又可以较好地保留原种群中有价值的遗传信息。通过改变变异概率的值进行实验对比可知，在本文设计的拓扑路径中，变异率在1%能提供一个很好的结果，故将变异概率的值设为1%。

常见的变异算子有翻转、随机重置和交换突变。本文的变异算子是随机重置，随机置换就是随机地选择一条染色体中的某个基因，将其值改为一个合理的随机值。

4）终止检查

终止检查决定运行的算法在何时终止，遗传算法通常设置迭代次数作为算法的终止条件，但本文的QoS控制需要对算法收敛的速度给予提升，所以设定了另一个终止条件，即得到的满足需求的染色体（路径）数量到达设置的N值时，也可以将遗传算法终止，并将适应度最好的一条路径作为解。N的值能够根据具体的场景灵活设置，以满足不同场景的需求，包括最初QoS请求需要的更好服务（较大N值）和动态路由时需要的迅速收敛（较小N值）。

利用遗传算法求解QoS路由的流程如图3.15所示，其中M为初始种群数，L为下一代的种群数，N，K为判断算法是否停止的终止条件。

图3.15 QoS路由流程图

3.5 本章小结

本章首先阐述了SDN网络中QoS控制策略的需求分析，对整体的QoS控制框架的设计进行了介绍。然后依据 Ryu控制器、OpenFlow 协议和OpenvSwitch，对框架中各个模块的作用和设计原理进行了具体阐述，最后对基于遗传算法的多QoS约束路由算法的设计进行了详细的阐述。

4 QoS控制策略实现

4.1 QoS控制框架实现概述

基于SDN网络的QoS控制策略框架的实现包括两方面：QoS路由和队列调度，本章将阐述这两部分功能的具体实现。

如图4.1所示，在Ryu控制器中增添了Topology Manager 模块、Measure模块、Routing Management模块，Ryu中的这三个模块分别对应于QoS策略控制框架中的拓扑管理模块、链路性能测量模块、路由管理模块。

Measure模块实现了周期地测量网络带宽、丢包率、时延、链路拥塞率等QoS性能参数的功能。Topology Manager模块能够获取网络拓扑视图，并将链路性能信息与链路的连接关系结合起来，让控制器拥有更详细和准确的网络全局资源信息，形成全局QoS拓扑视图。Routing Management实现了区分不同数据流的功能；并进行相应的路径计算；将数据流的标记、入队以及计算出的路径以流表的形式下发到交换机中，供数据流匹配、传输；在网络传输路径出现拥塞时，能够及时进行动态路由，保证视频流媒体的QoS。

图4.1 QoS控制在Ryu中的实现

在OpenvSwtich软件交换机中实现了QoS控制策略框架的队列调度模块，主要负责在数据转发层面为业务提供QoS保障。下面将阐述QoS控制框架中各个模块在Ryu控制器和OpenvSwitch软交换机中的具体实现。

4.2 QoS路由的实现

4.2.1 链路性能测量模块的实现

SDN中OpenFlow协议作为南向接口协议被广泛应用，由第三章可知OpenFlow协议并不是专门为测量网络链路性能而设计的，且OpenFlow交换机不会向控制器直接提供交换机链路的性能指标信息，但是支持OpenFlow协议的交换机内部有相关统计运行信息的计数器，用于记录每个Port、Flow 等信息，这些计数器记录的信息足够支持完成链路性能测量工作。下面将详细介绍链路性能测量模块的实现。

1）链路性能测量模块框架

链路性能测量模块周期地对链路进行测量，获取全网链路QoS性能信息。图 4.2展示了链路性能测量模块相关类和接口之间的关系，下面说明其中各个类和接口的功能。

图4.2 链路性能测量模块类和接口图

TrafficMonitor类为链路性能测量模块的主体，类中的_monitor()方法能够以固定时间间隔向底层的所有交换机发送Port_Stats_Request和Flow_Stats_Request请求消息。类中的_flow_stats_reply_handler()方法和_port_desc_stats_reply_handler()方法的作用是在交换机收到请求报文后，将需要的流信息和端口固定带宽信息上报给控制器。

BandWidth类定义了_get_allbandwidth()、_get_freebandwidth()和_get_bandwidth_ congestion ()三个方法，分别计算拓扑结构中每条链路的固有带宽、剩余带宽和拥塞率。

PackLossRate类定义了_get_packet_loss_rate()方法用以计算网络拓扑链路的丢包率。

Delay类中的_get_delay()方法能够计算得到网络拓扑中所有链路的时延值。

BandWidthDisplay、PacketLossRateDisplay、DelayDisplay三个接口分别定义了show_bandwidth()、show_packet_loss_rate()、show_delay()三个方法用来展示网络中每条链路的带宽、丢包率以及时延的信息。ConsoleLinkInfoDisplay类实现了以上三个接口中的方法，用以在控制台展示相关信息。

TopoFind类为整个链路性能测量模块提供依赖，提供整个网络的拓扑连接情况。

SwitchPortInfo类负责缓存交换机端口统计报文中与测量相关统计数据，time变量保存测量的时间戳，为链路带宽、链路拥塞率、链路丢包率计算提供数据依赖。

2）带宽、拥塞率及丢包率测量实现

由于带宽、链路拥塞率以及丢包率性能信息都是通过分析和计算OpenFlow交换机端口或流统计信息获得的，本节将链路带宽、链路拥塞率和丢包率测量的实现合并介绍。

Traffic类中的_monitor()方法是一个周期执行的线程，该线程以2s为一个周期向所有交换机发送获取端口信息、流信息以及固定带宽值的请求报文，交换机根据收到的报文内容，将端口的流量统计数据、流的统计数据以及端口固定带宽信息发送给Ryu控制器，Ryu对收到的反馈报文进行解析计算，即可得到所需的数据值。根据端口流量统计数据得到的交换机端口发送与接收的字节数以及两次统计的时间间隔能够计算出端口的速率值，见公式(3.3)；根据OFP_Port_Stats消息中的rx_packets字段，OFP_Flow_Stats消息中的instructions字段和packets_count字段获得的信息能够计算得到一条链路两端发送与接收的packets，见公式(3.7)和(3.8)。

OpenvSwitch端口的带宽为20Gbps，需要对交换机端口的实际带宽进行限定，以符合真实的情况，本文在Ryu编写的控

制程序中对所有端口带宽值进行设置，这个值即为交换机端口的实际带宽值。启动Ryu控制器连接底层交换机后，链路性能测量模块会发送Port_Desc_Stats_Request统计报文获取这个带宽数值，根据反馈的报文将带宽信息进行存储。

在得到速率与端口带宽后，链路性能测量模块会使用公式(3.5)和(3.6)计算得到端口的剩余带宽和端口的拥塞率，之后根据一条链路端口间的对应关系得到链路的剩余带宽和拥塞率。在得到送往当前交换机的某个端口的数据包总数以及对端端口收到的报文总数后，链路性能测量模块会利用公式(3.9)计算链路的丢包率。图4.3为获取交换机链路剩余带宽和拥塞率以及丢包率的流程图。图4.4、图4.5以及图4.6分别为链路剩余带宽和拥塞率以及丢包率的终端展示，可以更直观的看到网络中的这些信息。

图4.3 获取链路性能信息的流程图

图4.4 剩余带宽展示图

图4.5 拥塞率展示图

10985513970

图4.6 丢包率展示图

3）时延测量实现

通过发送LLDP数据包和 Echo数据包并携带时间戳得到相关的时间信息，根据时延计算公式计算出链路的时延。图 4.7是获取链路时延的相关类的关系图，图4.8是链路间时延信息的展示图，下面阐述每个类的功能。

OFEchoHandle类用以处理OpenFlow交换机和Ryu控制器间信息交互的连接，类中的send_echo_message()方法完成向交换机发送EchoRequest请求报文的功能，并将发送请求信息时的时间Techo_start放入请求报文中，交换机收到Ryu控制器发来的EchoRequest请求报文后，向控制器回复带有Ryu控制器请求信息时间的EchoReply 报文，控制器使用dispose_echo_reply()方法处理 Echo Reply报文后能够获取到其携带的请求报文时间Techo_start，Ryu控制器接收EchoReply报文的时间Techo_end与发送EchoRequest 请求报文的时间Techo_start的差值就是交换机和控制器间往返通信的RRT时间。

LinkDiscoveryManager类通过发送LLDP数据包实现对OpenFlow交换机间链路连接情况的监控，类中的send_LLDPmessage()方法实现控制器Ryu向网络中的所有交换机发送LLDP报文的功能，并将消息发出的时间戳Tlldp_start插入到消息报文中，每个交换机收到控制器发来的 LLDP 数据包后，根据流表的规则将LLDP数据包发送给自己的邻居交换机，邻居收到LLDP 数据包后无法处理，把该数据包交给Ryu控制器；控制器在收到这个LLDP消息后会使用handle_LLDPmessage()方法来处理这个LLDP报文，然后解析该报文中Ryu控制器插入的时间戳并存储这个时间，控制器收到LLDP报文的时间记为Tlldp_end ；Tlldp_end与Tlldp_start的差值即为LLDP报文从控制器发出，经过交换机后，再回到控制器的时间。

LinkDelay类中的monitor_server_latency()方法周期地向网络中发送LLDP报文与EchoRequest报文，get_latency()方法是根据LinkDiscoveryManager类和OFChannelHandle类计算获得的时间数据并通过公式(3.10)计算并存储时延信息。

TopoFind为整个时延测量模块提供拓扑结构的依赖，构建网络拓扑视图。

图4.7 时延测量实现图

图4.8 时延信息展示图

4.2.2 拓扑管理模块的实现

拓扑管理模块负责为全局提供拓扑结构，并为QoS路由算法提供带有最新的链路性能参数的拓扑数据。该模块利用LLDP协议获取的链路连接关系信息和链路性能测量模块提供的链路QoS参数信息实现。图4.9展示了拓扑管理模块相关类的关系，下面阐述每个类的功能。

TopoManager类为拓扑管理的主体，类中的discover()方法以线程的形式周期地发现网络资源。

TopoFind类负责网络监测并发现网络中的拓扑，它内部维护了全局网络的拓扑映射信息。

Graph类是物理网络拓扑的抽象图，提供方法将拓扑信息转换为邻接矩阵和邻接链表以供路径计算模块使用。类中的get_adjacency_matrix()方法是将网络拓扑与链路性能信息结合，附带有网络实时链路性能参数的拓扑，可以为QoS路由算法提供最新资源，Dijstra()方法将拓扑结构转换成邻接矩阵供Dijstra算法使用。

Edge类保存链了路的性能参数信息，其中s、t、bandwith、loss Rate、delay以及congestion rate分别表示保存当前链路的源交换机编号、目的交换机编号、剩余可用带宽、丢包率、时延和拥塞率。

ConsoleTopoDisplay是TopoDisplay接口的一种实现，负责将获取到的拓扑信息展示于终端。

图4.9 扑管理模块类与接口间的关系

4.2.3 路由管理模块

1）路由管理模块框架

本文在Ryu控制器上增添了Routing Management模块，对应于QoS路由控制框架的路由管理模块，它实现了数据流的区分以及基于Dijstra算法和遗传算法的路由功能，用以计算相关应用数据流的路由，并将需要的控制行为以流表的形式下发到交换机上，使交换机依照流表执行命令。路由管理模块所涉及到的类和接口关系如图 4.10所示，下面详细阐述这些类和接口。

PacketInMessageHandler类是对PacketIn报文的处理，包括丢弃、泛洪以及转发处理，转发处理（do_forwarding_flow()方法）即是对有路由请求的数据流进行路径计算及转发。

RoutePlanner接口定义了calculate()方法，用以路由的计算。QoSRoutePlanner类和DijstraRoutePlanner类实现了RoutePlanner接口，分别是视频流媒体基于遗传算法的QoS路由，其他等级数据流的基于Dijstra算法的路由。

Graph类是为QoSRoutePlanner类和DijstraRoutePlanner类提供带有链路性能的拓扑信息以及拓扑信息。

RouteHandler类主要实现数据流的区分，将路由展示在终端以及下发流表到交换机上。

DynamicQoSRoute类为网络拥塞时的动态路由模块，类中的add_path()与del_path()方法的作用分别为添加新路径和删除旧路径。

图4.10 路由管理模块类与接口图

2）路由管理模块实现

在SDN网络中，数据流的传输依靠流表中流表项的动作，若匹配到某个流表中的流表项，就按照该流表项规定的动作进行传输；若未能匹配到任何流表，一般情况下，数据包的信息将会以packet-in消息的形式传给给控制器，控制器路由管理模块中的PacketInMessageHandler类会对该packet_in进行不同的处理。

指　标

疑似剽窃文字表述

---

1. 选择算子所要实现的是把选择出来的优秀个体直接遗传到下一代或通过交叉算子产生新的子代个体再遗传到下一代。
2. 中的实现
   在OpenvSwtich软件交换机中实现了QoS控制策略框架的队列调度模块，
3. 控制器和OpenvSwitch软交换机中的具体实现。
   4.2 QoS路由的实现
   4.
4. 所有端口带宽值进行设置，这个值即为交换机端口的实际带宽值。启动Ryu控制器连接底层交换机后，链路性能测量模块会
5. Handle类用以处理OpenFlow交换机和Ryu控制器间信息交互的连接，类中的send_echo_message()方法完成向交换机发送EchoRequest请求报文的功能，并将发送请求信息时的时间
6. 控制器回复带有Ryu控制器请求信息时间的EchoReply 报文，控制器使用dispose_echo_reply()方法处理 Echo Reply报文后能够获取到其携带的请求报文时间Techo_start，Ryu控制器接收EchoReply报文的时间Techo_end与发送EchoRequest 请求报文的时间Techo_start的差值就是交换机和控制器间往返通信的 RRT时间。
   LinkDiscoveryManager类通过发送LLDP
7. 控制器；控制器在收到这个LLDP消息后会使用 handle_LLDPmessage()方法来处理这个LLDP报文，然后解析该报文中Ryu控制器插入的时间戳并存储这个时间，控制器收到LLDP报文的时间记为

5. S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第5部分

总字数：9699

相似文献列表　文字复制比：14.2%(1374)　疑似剽窃观点：(0)

1	基于OpenFlow的SDN网络QoS路由策略研究 周怡(导师：苏俭) -《电子科技大学硕士论文》- 2018-04-10	5.5%（537） 是否引证：是
2	8_陈彬_5G无线网络协作中继技术及仿真研究 陈彬 -《学术论文联合比对库》- 2017-04-05	4.0%（391） 是否引证：否
3	026_201422260255_陈忠 陈忠 -《学术论文联合比对库》- 2017-04-06	3.7%（361） 是否引证：否
4	基于SDN数据中心的流量调度算法研究 雷鸣(导师：李静) -《西安工业大学硕士论文》- 2018-05-15	2.3%（226） 是否引证：是
5	OpenFlow网络中业务相关路由管理方案的设计与实现 池悦(导师：张伟) -《南京邮电大学硕士论文》- 2016-11-18	1.3%（122） 是否引证：否
6	SDN网络中端到端QoS控制机制的研究 叶云东(导师：曹争) -《东南大学硕士论文》- 2017-06-03	0.5%（50） 是否引证：否
7	whbg200108326665 -《学术论文联合比对库》- 2013-12-17	0.4%（34） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

为了对最高优先级的视频流媒体数据流使用QoS路由，本文对do Forwarding Flow()方法进行了扩展，使得Ryu控制器在区分出视频流媒体数据流后能基于遗传算法进行QoS路由，其他等级的业务数据流基于Dijstra算法进行最小跳数路由。

图4.11为扩展后的do Forwarding ()方法的执行过程。在PacketInMessageHandler类收到packet_in 消息并解析为路由请求后，由do Forwarding ()方法对packet_in 消息进行进一步的处理，do Forwarding ()方法先要提取消息报文中的报文头信息，随后对报文头中的数据进行解析得到源目的交换机的信息和应用层协议，若解析得到的是非最高优先级业务流，Ryu使用以链路跳数为代价的Dijkstra算法来计算源目的交换机间的传输路径，然后以此该路径下发流表到相应的交换机上，以引导业务

数据流的传输。如解析得到的是最高优先级的视频流媒体，则需要使用遗传算法获得源目的交换机间的传输路径。若传输路径出现拥塞，则会触发DynamicQoSRoute类调用QoSRoutePlanner重新规划路径，若新的路径与RouteHandler中的result_route不同，则直接增加新的路径，将旧的路径删掉，按照新路径进行流表的下发，若新的路径与RouteHandler中的result_route相同，则表示网络中已经出现了全面拥塞，这时候只能依靠队列调度策略在端口对视频流媒体进行优先调度，最大可能减少拥塞带来的问题。流表中的动作除了有转发到端口，还有数据包标记以及入队的动作，二者都是队列调度策略的基础。

图4.11 do Forwarding ()方法的执行过程

4.2.4遗传算法的实现

1）遗传算法模块框架

本节对基于遗传算法的QoS路由的实现进行阐述，算法的设计在前一章节已经进行了介绍，图4.12展示了路由算法实现的相关类之间的关系，其中Genetics类实现了基于遗传算法的QoS路由，individual()是随机路径生成算法，能够随机生成不同的路径即种群中的个体；population()方法完成初始种群生成的功能，利用链路保存一组染色体，该类是遗传算法进化计算的作用对象；fitness()是适应度函数的计算；evolve()是遗传算法逻辑主体，内部实现了各种遗传算子。

图4.12 遗传算法模块框架图

2）种群初始化策略实现

Genetics类是种群初始化的核心类，其中individual()方法随机生成从起点到终点的路径作为初始种群，它也是QoS路由算法的重要方法，算法的流程如图4.13所示。该算法是对深度优先遍历算法的一种改进算法，考虑到路径生成时需要尽可能保证路径的随机性，因此方法执行流程中，每次选择未被标记的邻接节点之前，先对其进行随机排列，保证下一个加入路径的节点是完全无规律的。同时，方法对已经访问过的节点进行标记，保证生成的路由没有环路。当遇到一个节点所有邻接节点都已经被标记时，执行回退操作，直到找到一个点有邻接节点未被标记时停止，并将其邻接节点添加至路径链表中，重复上述操作，直到得到连接源点和终点的一条路径。

图4.13 individual()方法流程图

Genetics类的population()方法通过individul()生成随机路径，每次生成的路径会保存到路径集合中，如果新生成的路径在此集合中，新的路径将被丢弃，不用于构造种群。如果新生成的路径不在此集合中，就将新路径加入到路径集合中，直到达到种群规模。这样第一代种群中的每个染色体都是不同的，对于保障种群基因多样性是非常有益的，能够增大搜索空间的广度。种群初始化流程如图4.14所示。

图4.14 种群初始化流程图

3）适应度值的计算

适应度值的计算由 Genetics类的fitness()方法实现。适应度值计算的流程如图4.15所示，在染色体适应度计算前需要设计好适应度函数，适应度函数在第三章已经进行了设计，除适应度函数外还需取得保存路径的链表，即路径包含的链路。然后通过拓扑管理模块提供的每条链路具体的QoS性能信息，获得路径的QoS 性能指标。最后根据适应度函数对算每条染色体的适应度进行计算。在设计适应度时，对时延和丢包率的计算都除以了该代所有染色体相应性能值的和，所以在获得路径链表后，应计算出所有染色体相应性能值的和，以便每个性能的适应度值的计算。

图4.15 适应度值计算流程图

4）遗传算子的实现

本节阐述基于遗传算法的QoS路由用到的遗传算子的实现，包括了选择、交叉、变异等，这些算子都通过Genetics类进行实现。

a、选择算子

精英选择策略实现流程如图4.16所示，精英选择策略以当前的种群数作为输入，对输入种群中的个体进行适应度值的计算，得到个体的适应度值后对适应度值进行排序，之后根据预先设定的保留率，选择群体中较好的精英个体，直接复制到下一代，保留率值设置为10%；在剩下的个体中遍历，每遍历一个个体，使用随机生成器生成一个随机值，与选择率做比较，若选择率大于随机值，则将这个个体作为下一代种群中的个体，选择率设置为1%，随机生成器生成的随机值是在0-1之间。

图4.16 精英选择法流程图

b、交叉算子

交叉算子实现流程如图4.17所示，对经过精英选择策略后剩下的所有个体进行交叉，交叉的规则为在剩余的染色体中随机选择两个染色体，若两个染色体中除首尾基因外有超过一个的相同基因，则以随机选择交叉基因位点的执行交叉操作，由于在交叉之后形成的路径可能会存在环路的情况，因此需要对交叉后形成的新的染色体进行基因是否重复的判断，以便排除有环路的染色体。

图4.17 交叉算子流程图

c、变异算子

变异算子使用的是随机重置，随机重置方法的实现如图4.18所示，遍历所有经过选择与交叉后的下一代，每遍历一个个体，使用随机生成器生成一个随机数，与变异率做比较，若变异率大于随机值，即在该染色体中随机选择一个基因位点随机重置为新的值。

图4.18 变异算子实现流程图

d、终止校验算子

终止判断流程如图4.19所示，本文实现了两种终止判断方式，一是迭代数，算法执行过程中种群代数达到设置的最大进行代数，算法将退出搜索过程，返回最好的一条路径；二是需要依赖算法初始化时设置的N值，即存储的满足需求的染色体（路径）的数量到达N时，也可以终止遗传算法，并返回其中适应度最好的一条路径。这样做的原因是本文的QoS控制对时间要求非常敏感，算法收敛的速度提升可以提高路由效率，提高QoS控制的性能。

图4.19 终止判断流程图

4.3 队列调度策略的实现

实时视频会议与监控视频等视频流媒体业务对时延、带宽等QoS参数有较高的要求，其他业务对QoS要求依次降低。依据表3.1，本文将实时视频与监控视频、关键数据业务、事物处理与交互式数据、数据同步与e-mail优先级分别映射为32、24、16、8。根据第三章提出的队列调度的框架，将对队列调度功能予以实现。

队列调度策略中的流分类、标记、入队操作由控制器下发流表来完成，队列调度规则的配置在OpenvSwitch上完成，具体流程如图4.20所示。

图4.20 队列调度策略实现流程图

当数据到达边缘交换机，控制器根据其四层协议将数据流进行分类，根据分类结果把数据流IP头中的DSCP值标记为对应的优先级，如表3.1。对DSCP值的修改是利用了OpenFlow动作集中的修改域，修改域可选的字段部分如图4.21所示。

图4.21 修改域可选字段图

其中ip_dscp为IP报文头中ToS的前六位，通过修改ip_dscp的值，实现对数据包的标记，具体操作如图4.22所示，parser.OFPActionSetField是执行DSCP值修改的动作， parser.OFPActionOutput是执行数据包发到端口的动作，此动作是所有动作的前提，若没有该动作，数据包将会被丢弃。入队操作是指控制器根据OpenvSwicth交换机中队列的配置，通过流表的形式将不同等级的数据包放入不同的对列，具体命令为parser.OFPActionSetQueue(queue_id)，queue_id为交换机端口上已经配置好的队列号。

图4.22 修改DSCP值的程序图

HTB队列调度规则的实现是通过对 OpenvSwitch软交换机上的linux-htb队列模块进行配置来完成的,具体配置如图4.23所示。

图4.23 HTB队列规则配置图

其中，type是配置OpenvSwitch端口队列的类型；dscp用于配置进入队列的业务流的优先级，具体值见表3.1；max-rate是指令牌桶的产生速率，表示当前队列每秒传输数据的最大字节数； burst表示桶大小，指能处理的突峰发送速率；protiory用于配置DWRR的优先级，用来保证重要的数据进入优先队列而优先发送；quantum表示在进行 DWRR 算法调度时，每次最多调度的字节数。图4.24、图4.25与图4.26分别是是端口列表查询、qos列表查询和queue列表查询的结果图。从图4.24中可以看到配置了端口qos策略和未配置端口qos策略的qos值的不同；从图4.25和4.26中可以看出qos和queue的配置结果。

图4.24 端口列表查询图

图4.25 端口qos查询图

图4.26 端口queue查询图

4.4 本章小结

本章叙述了视频流媒体QoS控制策略框架中各个模块的实现，主要是QoS路由模块中的拓扑管理模块、链路性能测量模块以及路由管理模块的实现过程，分别对各个模块的具体实现类和重要方法的执行流程做了详细介绍。并在OpenvSwitch的端口上完成了HTB队列规则的配置，以实现转发端口对QoS的保障。

5 实验仿真与结果分析

5.1 仿真实验环境介绍

5.1.1 网络仿真软件Mininet

与NS2/3、OPNET等网络仿真器不同， Mininet[48-49]依靠其轻量级虚拟化技术的优势能够实现在单一的系统上仿真一个完整的网络的功能，即便在一台普通的电脑上，也可以轻易的创建出与真实网络黄精一致的网络组成结构，模拟的网络拓扑最多可有4096台主机的网络结构。Mininet支持OpenFlow协议各种版本以及OpenvSwitch，可以很方便的进行SDN应用的验证和测试。Mininet支持系统级别的回归测试，其不仅提供可由命令行直接创建的常见网络拓扑结构，还支持用户自己创建所需的任何拓扑结构。除此之外，在Mininet上编写的相应程序完成的功能可以直接迁移到真实的硬件环境中，这对于开发与SDN有关的应用是非常方便的，这是因为搭建真实的SDN网络环境的难度较大且需要消耗大量的时间，Mininet的这一特性在一定程度上缩小了SDN应用开发测试以及验证的周期，灵活性较大。

Mininet是一个非常强大的网络仿真工具，其借助应用程序编程接口(Application Pro gramming Interface)可以很方便地创建复杂的拓扑网络。表5.1描述了Mininet中创建网络拓扑常用的函数及变量，通过这些函数可以简单、快速地创建所需的网络。Mininet也支持Linux的命令行接口(CLI)调用，在CLI上，输入nodes命令可以打印出网络中的所有节点，输入net命令可以打印出网络中的节点连接情况，通过links获取节点间的连接状态，通过link禁用或启动两个节点间的链路。

表5.1 Mininet网络相关函数功能

函数功能

addHost() 为网络设置主机，可配置相关参数

addSwitch() 为网络设置交换机，可配置相关参数

addLink() 为网络设置链路，可配置相关参数

start() 开启网络运行

pingall() 测试网络的连通性

stop() 停止网络运行

5.1.2 OpenvSwitch

SDN架构中基础设施层的转发设备也可由软件的形式实现，目前软件交换机也已能够满足很多应用场景中的数据传输，因此以软件交换机的形式构建SDN网络，已成为可选方案之一。

OpenvSwitch是一个开源的虚拟交换机，支持许多协议和管理接口，如 CLI，LACP等[50]。OpenvSwitch能够以软件交换机的形式在虚拟环境中实现许多功能验证，也可实现在硬件交换机中作为控制软件的功能，其支持OpenFlow协议及用于虚拟化的其他扩展。

5.1.3 其他实验环境及相关软件介绍

操作系统：Windows7专业版64位，内存（RAM）：8.00GB

虚拟机和虚拟机系统：VMware Workstation10.0.1build-1379776，Ubuntu14.04

VLC media player：2.2.2

数据包抓包软件Wireshark

OpenvSwitch：2.5.0

Mininet：2.3.0

Ryu：4.25

摄像头：Z-star Gsou USB 2.0

5.2 网络环境的搭建

视频传输的仿真平台主要由Ryu控制器、流媒体服务器、摄像头和Mininet软件模拟的网络拓扑组成，如图5.1所示。

图5.1 视频传输平台结构图

5.2.1 拓扑环境搭建

本文在Mininet中通过python脚本完成网络拓扑的构建，整个网络由s1、s2、s3、s4、s5、s6、s7和s8八个虚拟交换机组成，它们都由Ryu控制器控制和支配；有5个服务器和2个客户端，具体连接方式如图5.2所示。其中server1为流媒体服务器，server2为其他优先级业务流的服务器，server3、server4和server5是背景流的服务器，client2为背景流的接收端。由于OpenvSwitch默认的链路带宽为20Gbps，不符合真实情况，也不利于仿真验证，因此在创建拓扑进行仿真验证时，需要利用Mininet对链路带宽进行设置。

图5.2 仿真实验拓扑图

使用VMware创建一台Ubuntun虚拟机，安装Ryu和Mininet。为了在Mininet上成功地创建如图5.2所示的实验拓扑，首先需要编写自定义的 Python 脚本 cai-topo.py，如图5.3所示，并放在Mininet中的指定路径下，然后在CLI上进入到Mininet中运行此脚本，并连接已开启的指定控制器，若运行窗口出现图5.4所示的情况，且运行拓扑管理模块出现5.5所示的结果，则表示拓扑创建成功且控制器能正确获得底层网络拓扑的连接情况。

图5.3 自定义拓扑程序图

图5.4 运行结果及链路信息图

图5.5 控制器终端拓扑连接展示图

5.2.2 视频服务器搭建

本文使用VLC media player作为视频服务器，使用外置摄像头Z-star Gsou USB2.0 Camera动态获取640*480分辨率的实时视频，通过VLC media player将摄像头捕获的实时视频发送到Mininet虚拟的网络中。具体步骤如下：

使用Gsou USB2.0 Camera连接电脑，并将其挂载至Ubuntu虚拟机，在Ubuntu中打开VLC media player将其捕获的视频信息使用RTSP协议发送至/camera路径的8854端口。

在客户端中使用VLC media player的客户端模式连接至rtsp://10.0.0.1:8854/camera ，使用Wireshark监控客户端和服务器之间的视频传输，结果如图5.6所示，说明视频服务器搭建成功且客户端能够与视频服务器建立正确的连接并正常通信。

图5.6 Wireshark监控的传输情况图

5.3 实验仿真与结果分析

基于本文对视频流媒体QoS控制策略的研究，设计了四组实验分别对QoS路由计算的有效性、动态路由的有效性、队列调度的QoS控制性能、QoS控制策略性能进行仿真验证，用以说明QoS控制策略对视频流传输性能的提高。

为了模拟真实网络中的流量情况，在每个实验开始之前，分别使用iperf[51]由server3、server4、 server5向client2发送0.5Mbps的背景流，在以后的实验中不再说明。

5.3.1 QoS路由计算有效性分析

链路的时延反映了当前链路的传输效率，丢包率则反映了链路数据传输的可靠性，它们是衡量流媒体传输性能的重要指标。对于视频流媒体业务来说，其对网络的传输环境具有很高的要求，为其选择较低丢包率和较小时延的路径进行传输，才能保证其高效的传输以及对端较好的观看效果。仿真实验的流程如下：

1）、启动控制器和Mininet仿真工具，通过脚本创建图5.2的实验拓扑，并连接控制器。

2）、分别在server2和client1启动iperf工具，通过其模拟向SDN网络中不断发送1Mbps的其他类型业务流。

3）、分别在server1和client1上启动VLC的服务端和客户端，通过SDN网络发送和接收视频流媒体数据流。

4）、通过控制器监控网络中路径的时延与丢包率变化情况，并使用matplotlib库分别绘制遗传算法和Dijstra算法所选路径上的时延与丢包率的变化图，如图5.7和5.8所示。

图5.7 时延变化情况图

图5.8 丢包率变化情况

从图中可以看出，当视频流和其他类型的流进入SDN网络中之后，路径的时延和丢包率逐渐变大，一段时间之后，虽然两者都逐渐趋于稳定，但是使用遗传算法所选路径的时延和丢包率都小于使用Dijstra算法所选路径，并且使用遗传算法所选路径的时延和丢包率的值均在设定的QoS指标范围内，即丢包率在20×10-3以内，时延在10ms以内。综上所述，基于遗传算法的QoS路由计算提高了视频流的传输性能。

5.3.2 动态路由有效性分析

遗传算法的QoS路由计算仅仅能保证视频流在初次选路时有较好QoS保障，但是网络中的环境是多变的，一旦视频流在传输过程中遇到洪峰，视频流的QoS将会受到极大影响，此时必须使用动态路由为其优化路径，才能保证其QoS。动态路由的有效性分析，通过传输过程中是否使用了动态路由策略，对比视频流的时延抖动、吞吐量以及PSNR的变化情况，验证动态路由策略的有效性。为了便于仿真实验的验证，需要将s1-s7-s8链路带宽提高，使基于遗传算法选出的路径与最短跳数的路径相同，将s1-s7间链路带宽设置为2.8Mbps，s7-s8间链路带宽设置为3Mbps。

时延抖动用于衡量网络时延的稳定性，吞吐量用于表明单位时间内成功传输的数据大小。网络拥塞会导致时延抖动过大以及吞吐量过小，若持续时间过长，将会影响视频端到端的传输。仿真实验的流程如下：

1）、使用client1分别向视频服务器server1的8854和8855端口请求动态视频，其中8854端口的视频使用动态路由策略，8855端口的视频流不使用动态路由策略。使用iperf分别记录两种视频流传输过程中的时延抖动和吞吐量。

2）、在流媒体视频传输的第10s，通过iperf工具模拟向网络中发送3Mbps的其他类型业务流。

3）、使用iperf监控网络中的时延抖动和吞吐量的变化情况。图5.9与图5.10分别描述了有无动态路由策略时，视频流的吞吐量和时延抖动随时间的变化趋势。

4）、在客户端获取前40s传输的视频，使用ffmpeg进行解码，并计算出两个视频的峰值信噪比(PSNR)，比较的结果如图5.11。

图5.9 视频流在网络拥塞时有动态路由策略的时延抖动与吞吐量变化图

图5.10 视频流在网络拥塞时无动态路由策略的时延抖动与吞吐量变化图

图5.11 PSNR对比图

从图5.9和5.10中可以看到，在第10s时，因注入其他类型业务流致使网络中的流量突增，流量的增加使得网络中的传输链路出现拥塞，视频流媒体的吞吐量由于链路的拥塞迅速下降，时延抖动由于链路的拥塞而增加，若使用动态路由策略，控制器监测到链路拥塞后，通过控制器的动态路由机制，使用遗传算法重新为视频流媒体计算一条符合QoS需求的路径，因此在13秒后视频流媒体吞吐量逐渐上升，其时延抖动在4s后回落至稳定状态。而未使用动态路由策略的，只是在最初为视频流媒体选出一条符合需求QoS需求的路径，并未在网络状态发生变化时动态改变传输路径，各个等级的数据平等的争夺路径中链路的带宽，因此在网络拥塞后，网络中所有数据流“互不相让”，视频流媒体的吞吐量在18秒后才逐渐向稳定状态恢复，而延时抖动也是经历了两次跳跃才恢复至稳定。

峰值信噪比（Peak Signal Noise Ratio，PSNR）能判别出接收端接收的视频质量的优劣。从图5.11中两个视频流的PSNR对比可知，当视频流出现下降时，即传输路径中注入大量其他类型的业务流导致视频流传输出现拥塞时，动态路由机制能够使视频流的传输状态在较短的时间内恢复，而没有使用动态路由机制的视频流则恢复的较慢。两个视频流的PSNR对比同样验证了动态路由机制的有效性。

5.3.3 队列调度的QoS控制性能测试

传输速率是QoS的重要指标，业务接收速率与发送速率是否相同反映了网络是否提供业务流最好的传输保障。本次性能测试获取在不同发送速率下，分类业务的接收速率，以检测所提出的队列调度能否优先保障视频流媒体业务的QoS。

为了测试调度策略的控制性能，首先排除监控视频速率的不稳定性，为此四种优先级不同的数据流都由iperf来模拟；其次需要排除QoS路由的干扰，为此使用Dijstra算法基于跳数为数据流计算路径。当server1向client1发送报文时，业务报文将通过路径server1->s1->s7->s8->client1进行传输。由于背景流传输路径不固定，因此需要对所有交换机的出端口进行队列策略的配置，同时设置交换机间的链路带宽为16Mbps。图5.12是配置的一个端口队列规则示意图，其他交换机端口队列规则配置与其一样。

图5.12 队列规则配置图

server1利用iperf工具向client1发送6组UDP数据，6组报文的发送速率都是从1000kbps增加到7000kbps，每次的递增值为1000kbps，每组内不同优先级业务数据的发送速率相同，每组报文传输时间为3分钟，最后记录客户端的平均接收速率。图5.13和图5.14分别展示了调度策略配置与否下的不同优先级报文接收速率的情况。

图5.13 未配置调度策略接收速率图

图5.14 配置调度策略接收策略图

从图5.13中可以看到，在没有调度策略时，各种业务数据的接收速率和发送速率基本相同，在发送速率为4000Kbps左右时，接收速率的值基本都在4000Kbps及以下；之后尽管发送速率的值在不断增加，四种业务报文的接收速率却都在 3800 Kbps 上下浮动。这表明，在未配置端口调度策略时，网络对4种不同优先级业务的报文仅仅只提供了尽力而为的服务，并没有区分保障它们的QoS。

从图5.14中可以看到，配置了调度策略后，当报文的发送速率逐渐增大时，最低优先级的数据同步类业务的接收速率最先降低且在发送速率超过2000Kbps后就开始下降；之后是优先级第三的事务处理类报文的接收速率开始降低，在发送速率超过3000Kbps时，其接收速率开始下降；优先级次高的关键数据类业务报文在发送速率大于5000Kbps时开始下降；而优先级最高的视频流媒体业务报文，其接收速率与发送速率值的增长基本保持一致。

指　标

疑似剽窃文字表述

1. 为了对最高优先级的视频流媒体数据流使用QoS路由，本文对do Forwarding Flow
2. 跳数为代价的Dijkstra算法来计算源目的交换机间的传输路径，然后以此该路径下发流表到相应的交换机上，以
3. 转发设备也可由软件的形式实现，目前软件交换机也已能够满足很多应用场景中的数据传输，因此以软件交换机的
4. 为了在Mininet上成功地创建如图5.2所示的实验拓扑，首先需要编写自定义的 Python 脚本 cai-topo.
5. ninet中的指定路径下，然后在CLI上进入到Mininet中运行此脚本，并连接已开启的指定控制器，若运行窗口出现图5.4所示的情况，
6. 这表明，在未配置端口调度策略时，网络对4种不同优先级业务的报文仅仅只提供了尽力而为的服务，并没有区分保障它们的QoS。

6. S1481237969_ 基于SDN网络的视频流媒体传输性能研究_第6部分　总字数：2369

相似文献列表　文字复制比：6.8%(161)　疑似剽窃观点：(0)

1	基于SDN数据中心的流量调度算法研究 雷鸣(导师：李静) -《西安工业大学硕士论文》- 2018-05-15	5.3%（126） 是否引证：是
2	高超声速飞行器综合信道测量及信道特性研究 刘小彤(导师：刘彦明) -《西安电子科技大学硕士论文》- 2018-06-01	1.3%（31） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

从图5.11队列配置图中可以看到，优先级最低的数据同步类业务报文设置的最大传输速率为900Kbps但是在发送速率为2000Kbps的时候，其接收速率也为2000Kbps，是因为网络中的带宽还有很大的盈余，HTB启用了借带宽的策略用于保证其传输，但是随着发送的报文的速率逐渐增大，而同步类业务报文却逐渐减小，是因为网络中的剩余带宽逐渐减少，为优先保证高优先级的报文传输，逐渐增加了对数据同步类报文的限制，由于其优先级最低，所以在传输路径拥塞时接收速率最先被限制。

综合以上分析，等级越高的业务报文，它的QoS越能优先得到保障，这说明区分业务级别的端口调度策略可以保障不同等级业务报文的不同QoS需求。

5.3.4 QoS控制策略性能分析

上述三个实验从三个方面验证了QoS控制策略对视频流媒体传输性能的提高，本实验将其综合起来，以验证QoS控制框架的整体传输性能。

1）、设置队列调度规则，图5.15是配置的一个端口队列规则示意图，其他交换机端口队列规则配置操作与其一样。端口的配置规则为视频流媒体、其他三个等级业务流及背景流所占带宽分别是根带宽的50%、20%、15%、10%以及5%。

图5.15 队列规则配置图

2）、分别在server2和client1启动iperf工具，通过其模拟向SDN网络中不断发送其他类型的业务流。其他类型的数据流相关设置如表5.2所示。

表5.2 数据流相关设置

数据流类型发送速率发送端口 TOS值

关键数据类业务 0.4Mbps 5000 96

事务处理类业务 0.4Mbps 5001 64

数据同步类业务 0.4Mbps 5002 32

3）、分别在server1和client1上启动VLC的服务端和客户端，通过SDN网络发送和接收视频流媒体数据流。

4）、使用iperf工具监控网络中业务流吞吐量的变化情况，图5.16为网络中业务流的吞吐量随时间的变化情况。

图5.16 吞吐量变化情况

从图5.16中可以看出，Q2、Q3以及Q4三种业务流的发送速率相同，即0.4Mbps；传输的路径也相同，即跳数最短路径；吞吐量却依次降低，这是因为在交换机的各个端口对不同优先级的业务流的速率以及出端口的顺序进行了设置，如图5.15所示，这说明队列调度策略可以根据业务报文的类型保证其QoS。通过视频流的吞吐量与其他三种业务流的吞吐量对比可看出，在网络条件相同的情况下，视频流的的吞吐量高于其他三种业务流，这说明使用基于遗传算法的QoS路由可以为视频流选择性能更好的的路径进行传输。

5.4 本章小结

本章是对针对本文提出的QoS控制策略进行仿真实验分析。首先对实验的网络拓扑及相关设置进行说明。然后对QoS路由、动态路由、队列调度策略以及整体控制策略的有效性进行了分析。通过相应QoS指标对比表明，本文的QoS控制策略可以较好地保障视频流媒体的传输性能，并可根据业务优先级的不同，在端口保障不同优先级业务报文的QoS。

6 总结与展望

6.1 全文工作总结

随着网络技术的升级创新，网络上的业务流将会越来越多，流媒体业务的应用也将会逐渐占据着网络流量的半壁江山，这类业务对QoS的需求非常严格。然而传统的网络架构却存在着诸多不能满足流媒体业务传输的弊端，导致流媒体业务的传输性能较差。而SDN网络架构转控分离、可编程的特点，简化了网络管理的复杂性，并能灵活地调用网络资源，与传统的网络相比，为提高流媒体视频传输提供了可行的方案。因此本文基于SDN技术，通过将智能算法应用到SDN网络以及将新的队列调度算法应用到底层转发设备上，提高视频流媒体的传输性能。通过以上的介绍，本文主要研究工作内容如下：

第一，控制层的QoS路由控制策略。QoS路由策略利用SDN网络的可编程性以及能够周期地获得网络链路信息的特点，将遗传算法应用于QoS路由选路，并能执行动态路由的策略，在网络拥塞时，为视频流媒体计算新的传输路径，同时优化了流表更新顺序，解决了流表一致性问题，避免由动态路由造成的流传输中断以及丢包等问题。

第二，转发层的队列调度策略。队列调度策略根据HTB调度规则在转发端口处给视频流媒体分配较大带宽并给予优先转发的“权利”，在底层保证视频流媒体业务QoS；且带宽充足时，不同优先级的业务间可以相互借带宽，进而保障不同优先级业务的服务质量。

第三，通过 Ryu、Mininet 等软件成功模拟了网络实验环境，在文中给出了详细的逻辑架构以及物理拓扑和软件实现的过程。在环境中实现了本文的控制策略，从算法选路的优越性、动态路由的有效性、底层调度策略的实现以及QoS控制策略的有效性做出了测试。实验表明，本文提出的控制策略，能够较好地提高视频流媒体的传输性能，降低拥塞造成的视频不流畅等问题，达到了优化视频流媒体传输性能的目的。

6.2 不足与展望

本文利用SDN网络架构的可编程性及可周期获取链路信息的特点，提出了视频流媒体的QoS控制策略，比较有效地提高了视频流媒体的传输性能。但尚有不足之处需要进改进和进一步研究。主要包括：

第一，在交换机层面队列调度中只实现了 DWRR 调度算法，算法还存在着一定的局限性，之后需要改进的是加入其他的队列设置，使调度功能更完善，并且在交换机转发端口实现能够动态配置队列调度算法的功能。

第二，本文使用的网络拓扑结构相较于真实的网络来说比较简单，真实的网络拓扑结构通常是规模较大且较为复杂，像数据中心的网络结构就比较庞杂，因此本文所涉及的视频流传输控制策略还需要在其他复杂网络拓扑中进行仿真实验和验证。

第三，整个控制策略只是在模拟环境进行了实现，没有实际在真实环境中去部署，可以考虑移植到真实网络中进行测试改进。

参考文献

[1] 第42次中国互联网络发展状况统计报告 [R/OL]. http://www.cnnic.net.cn/hlwfzyj/hlwxzbg/hlwtjbg/201808/t20180820_70488.htm

[2] 冯径, 马小骏, 顾冠群. 适应QoS路由机制的网络模型研究[J]. 计算机学报,2000,23(8):799~805.

[3] 左青云, 陈鸣, 赵广松,等. 基于OpenFlow的SDN技术研究[J]. 软件学报, 2013(5):1078-1097.

[4] Gavras A , Karila A , Fdida S , et al. Future internet research and experimentation[J]. ACM SIGCOMM Computer Communication Review, 2007, 37(3):89.

[5] Elliott C . GENI: Opening Up New Classes of Experiments in Global Networking[J]. IEEE Internet Computing, 2010, 14(1):39-42.

[6] Farhady H, Lee H Y, Nakao A. Software-Defined Networking: A survey[J]. Computer Networks, 2015, 81(C):79-95.

[7] Specication OpenFlow,OpenFlow Switch.Version 1.5.0.(WireProtocol0x05)[S].[S.l.],2013.

[8] Ryu[EB/OL].[2011].http://ryu.readthedocs.io/en/latest/getting_started.html.

[9] Noskov A A, Nikitinskiy M A, Alekseev I V. Development of an active external network topology module for Floodlight software-defined network controller[J]. Automatic Control & Computer Sciences, 2016, 50(7):546-551.

[10] 许名广, 刘亚萍, 邓文平. 网络控制器OpenDaylight的研究与分析[J]. 计算机科学, 2015, 42(S1):249-252.

[11] Jain S, Kumar A, Mandal S, et al. B4: experience with a globally-deployed software defined wan[C]//ACM SIGCOMM 2013 Conference on SIGCOMM.[S.l.],2013:3-14.

[12] Chi Yaohong, Srikanth Kandula, Ratul Mhajan, et al. Achieving .High Utilization with Software-Driven WAN[C]. Hong Kong, China, SIGCOMM,2013,8.

[13] 黄韬, 张丽, 张云勇,等.基于OpenFlow的SVC流媒体时延自适应分级传输方法[J]. 通信学报, 2013(11):121-128.

[14] 陈志钢. 基于 Open Flow 和 MTR 的 IP 网络流量控制方法的研究. 电子科技大学，2013.

[15] 徐李谦. 基于SDN的视频流调度机制研究.重庆邮电大学，2017.

[16] 李建冲.基于SDN的业务服务质量保障技术研究.西安电子科技大学，2017.

[17] K.-W. Kwong, R. Guerin, A. Shaikh, and S. Tao. Improving service differentiation in IP networks through dual topology routing. Proc. Co NEXT, December 2007, pp. 26-28.

[18] Civanlar S, Parlakisik M, Tekalp A M, et al. A QoS-enabled Open Flow environment for Scalable Video streaming[C]. GLOBECOM Workshops. IEEE, 2010:351-356.

[19] N.Handigol, S.Seetharaman, M. Flajslik, N. Mc Keown, and R. Johari. Plug-n-Serve:Load-Balancing Web Traffic using Open Flow. In ACM SIGCOMM Demo, August 2009, pp.268-270.

[20] Nikhil Handigol,Srini Seetharaman, Mario Flajslik, Aaron Gember,Aster*x: Load-Balancing Web Traffic over Wide-Area Networks. http://www.standford.edu/.

[21] gilmez H E, Civanlar S, Tekalp A M. An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks[J]. IEEE Transactions on Multimedia, 2013, 15(3):710-715.

[22] Dobrijevic O, Santl M, Matijasevic M. Ant colony optimization for QoE-centric flow routing in software-defined networks[C]//International Conference on Network & Service Management. 2015.

[23] Owens H, Durresi A . Video over Software-Defined Networking (VSDN)[J]. Computer Networks, 2015, 92:341-356.

[24] 王嗣平. 基于SDN网络的视频流传输控制系统设计与实现[D]. 国防科学技术大学, 2016.

[25] QoS. http://zh.wikipedia.org/wiki/QoS, June 19, 2012.

[26] 田冲. 基于IP网络的QoS队列调度算法研究[D]. 南京邮电大学, 2013.

[27] 武莹, 王敬宇.基于OpenDaylight的虚拟网络QoS控制系统的设计与实现[J]. 2018.

[28] 吴慧, 陈秋红,刘国辉. OpenFlow网络基于DiffServ模型的QoS管理机制的实现[J]. 电视技术, 2014, 38(5):113-115.

[29] 崔潇宇,沈庆国. MPLS网络中基于QoS约束的路由和准入控制[J]. 通信技术, 2018, v.51；No.318(06):102-105.

[30] Kutscher D, Ahlgren B, Karl H, et al. 10492 Executive Summary -- Information-Centric Networking[J]. 2011.

[31] Mckeown N, Anderson T, Balakrishnan H,et al. OpenFlow:enabling innovation in campus networks[J]. Acm Sigcomm Computer Communication Review, 2008, 38(2):69-74.

[32] McKeown N. Software-Defined Networking[R/EB]. In:Proc.of the INFOCOM Key Note. 2009. http://infocom2009.ieee-infocom.org/technical Program.htm.

[33] 房秉毅,张歌,张云勇,等.开源SDN控制器发展现状研究[J].邮电设计技术,2014(07):29-36.

[34] 基于SDN的数据中心流量工程研究[D]. 电子科技大学, 2016.

[35] 许莹. 基于区分服务（DiffServ）的IP QoS控制策略研究[D]. 湖南大学, 2003.

[36] 雷鸣. 基于SDN数据中心的流量调度算法研究[D]. 西安工业大学，2018.

[37] Book R V. Book Review: Computers and intractability: A guide to the theory of NP-completeness[J]. Bulletin of the American Mathematical Society, 1980, 3(2):898-905.

[38] 李晓方.混合SDN的流量矩阵估计和路由优化研究[D].中国科学技术大学,2015.

[39] 董晓林.面向SDN的多路径调度算法研究[D].郑州大学,2017.

[40] 朱冠宇. 基于遗传算法的QoS路由选择策略研究[D]. 华中科技大学, 2004.

[41] 文强.SDN网络业务量工程技术研究[D]. 电子科技大学, 2016.

[42] 陈忠. SDN网络中QoS控制技术研究与实现[D]. 2017.

[43] 孔智. 基于三层交换机服务质量的研究[D]. 2016.

[44] 杨俊超. 基于OpenFlow网络的QoS管理策略研究[D]. 2015.

[45] 林玉侠, 朱慧玲,马正新,等. QoS路由度量参数的选择问题研究[J]. 电信科学, 2003, 19(7):22-27.

[46] 周怡.基于OpenFlow的SDN网络QoS路由策略研究[D]. 2018.

[47] Chang W A, Ramakrishna R S.A genetic algorithmfor shortest path routing problem and the sizing of populations[M]. IEEE Press, 2002.

[48] Team M. Mininet:An instant virtual network on your laptop(or other PC) [DB].2012.

[49] 黄家玮,韩瑞,钟萍,王建新.基于Mininet 的计算机网络实验教学方案[J].实验技术与管理,2015,32(10):139-141.

[50] 张若晨.基于OpenvSwitch的代理虚拟交换机在SDN网络中的实现与应用[D]. 2016.

[51] 张白,宋安军.基于 Iperf 的网络性能测量研究[J].电脑知识与技术, 2009,5(36): 27-29.

说明：1.总文字复制比：被检测论文总重合字数在总字数中所占的比例

2.去除引用文献复制比：去除系统识别为引用的文献后，计算出来的重合字数在总字数中所占的比例

3.去除本人已发表文献复制比：去除作者本人已发表文献后，计算出来的重合字数在总字数中所占的比例

4.单篇最大文字复制比：被检测文献与所有相似文献比对后，重合字数占总字数的比例最大的那一篇文献的文字复制比

5.指标是由系统根据《学术论文不端行为的界定标准》自动生成的

6.红色文字表示文字复制部分;绿色文字表示引用部分

7.本报告单仅对您所选择比对资源范围内检测结果负责

amlc@cnki.net

http://check.cnki.net/

http://e.weibo.com/u/3194559873/