计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2017.20160922
54(11):2628-2638,2017

软件定义的 VANET 下流表用量感知的 QoS 路由机制

付 彬[1,2] 查理佳[1,2] 李仁发[1,2] 肖雄仁[1]
[1](湖南大学信息科学与工程学院 长沙 410082)
[2](嵌入式与网络计算省重点实验室(湖南大学) 长沙 410082)
(fubin@hnu.edu.cn)

A Flow Table Usage-Aware QoS Routing Mechanism in Software Defined VANET

Fu Bin[1,2], Zha Lijia[1,2], Li Renfa[1,2], and Xiao Xiongren[1]
[1](*College of Computer Science and Electronic Engineering, Hunan University, Changsha* 410082)
[2](*Key Laboratory for Embedded and Network Computing of Hunan Province (Hunan University), Changsha* 410082)

Abstract VANET can provide a wide range of security and non-security related services. However, the existing VANET is difficult to guarantee the QoS of these services. Software defined networking (SDN), which appears in a systematic way, can control network flexibly and separate the data from the control plane, bringing programming ability to the network. Firstly, this paper designs a software defined VANET architecture for heterogeneous multi network access. Secondly, a flow table usage-aware dynamic QoS provisioning framework is proposed, which allows us to manage the network in a modular way and supports the dynamic entering and exiting of the service flows. Finally, this paper establishes a flow table usage-aware QoS routing model with multi-service flows and multi-constraints. The model considers not only the parameters of link state such as packet loss, delay and throughput, but also the service requirements and the flow table usage, and provides VANET application services for a concurrent dynamic QoS routing. Experiments show that QoS routing mechanism proposed in this paper can meet not only the service requirements of their packet loss, latency and throughput, but also the capable of perceiving the usage of flow table so as to avoid the influence of flow table overflow for QoS routing mechanism, which further improves the performance of network QoS.

Key words software defined networking (SDN); vehicular ad hoc network (VANET); quality of service (QoS); routing model; flow table

摘 要 VANET 可以提供各类安全和非安全相关的服务,但现有的 VANET 难以保障应用服务的 QoS 需求.软件定义网络(software defined networking, SDN)以系统化的灵活控制网络的方式出现,其分离的数据与控制平面为网络带来了可编程性.因此首先设计了一种面向异构多网接入的软件定义的 VANET 架构;接着提出一种流表用量感知的动态 QoS 保障框架,允许使用模块化的方式管理网络,并支持业务流的动态加入和退出;最后建立了多业务流多约束条件下流表用量感知的 QoS 路由模型,该模型不仅考虑了丢包、时延和吞吐量等链路参数,还考虑了业务需求和流表使用情况,从而为 VANET 应用服务提供并发的 QoS 路由.实验表明:流表用量感知的动态 QoS 路由机制不仅能够满足

收稿日期:2016-12-07;**修回日期**:2017-02-21
基金项目:国家自然科学基金项目(61502162);中央高校基本科研业务费专项资金项目(531107040289)
This work was supported by the National Natural Science Foundation of China (61502162) and the Fundamental Research Funds for the Central Universities (531107040289).

多业务对各自丢包、时延和吞吐量的要求,还能够感知流表用量,从而避免流表溢出对 QoS 路由机制的影响,进一步提高了网络 QoS 保障的性能.

关键词　软件定义网络;车辆自组织网络;服务质量;路由模型;流表

中图法分类号　TP393

车辆自组织网络(vehicular ad hoc network, VANET)实现车与车(vehicle-to-vehicle, V2V)、车与基础设施(vehicle-to-infrastructure, V2I)、车与人及互联网的广泛通信,从而能够提供各类安全和非安全相关的服务[1].虽然 VANET 能够支持许多新的服务和应用,但是在部署 VANET 服务和应用时仍然面临三种主要挑战:

1) 体系结构的挑战.当前的 VANET 架构主要是基站和路边单元(rode side unit, RSU)的 V2V 与 V2I 混合架构,该架构缺乏灵活性,导致大规模地部署和设置新的服务或协议是很困难的.

2) 通信流量的挑战.车辆的移动性使得网络通信流量的动态性和不均衡性特征突出,现有的 VANET 缺乏全局视图,在数据转发过程中不能动态调度网络资源来合理分配通信流量,从而提升网络传输效率.

3) 网络服务质量(quality of service, QoS)的挑战.不同的 VANET 应用对于网络提供的 QoS 有着不同的要求.当前 VANET 的众多应用主要依赖于传统的 IP 分组网络,该网络的尽力而为服务模式不能为数据传输的带宽、端到端延迟和包到达率等性能提供服务质量保证.

针对以上问题,许多研究者提出将软件定义网络(software defined networking, SDN)[2-3]架构引入 VANET,从而分离 VANET 数据平面与控制平面,实现网络的系统化灵活控制,带来可编程性.例如:文献[4]提出了一种基于 SDN 的 VANET 架构,指出新架构将提高路径选择、频率/信道选择、功率选择方面的性能提升,并展望了新架构的 VANET 应用;文献[5]提出了基于 SDN 的 5G 网络 VANET 架构,并说明了 SDN 对下一代 5G 网络 VANET 的重要性;文献[6]提出了面向无线异构接入的软件定义的 VANET 架构,在该架构下无线异构设备诸如汽车、RSU 都可以抽象为 SDN 交换机;文献[7]提出了一种基于 SDN 并支持雾计算的 VANET 架构,同时该架构也支持无线异构接入.

利用软件定义 VANET 的控制与转发相分离以及集中控制的优势,本文对软件定义 VANET 的架构以及动态 QoS 路由相关问题进行了研究.本文的主要贡献有 4 个方面:

1) 设计了一种面向异构多网接入的软件定义 VANET 架构,完善了现有工作中数据平面的设计.

2) 针对现有的 VANET 难以保障应用服务的 QoS 需求问题,本文提出一种流表用量感知的动态 QoS 保障框架,允许使用模块化的方式管理网络,并支持业务流的动态加入和退出.

3) 在框架的基础之上建立了多业务流多约束条件下流表用量感知的 QoS 路由模型,该模型不仅考虑了丢包、时延和吞吐量等链路状态参数,还考虑了应用服务的需求以及交换机的流表使用情况,从而为 VANET 应用服务提供并发的 QoS 路由.

4) 利用仿真平台实现了本文所提出的保障框架的各个模块并进行了仿真实验.实验表明,本文提出的 QoS 路由机制不仅能够满足多业务对各自丢包、时延和吞吐量的要求,而且能够感知流表用量从而避免流表溢出对 QoS 路由的影响,进一步提高了网络 QoS 保障的性能.

1　相关工作

在 SDN 架构下,为了给应用服务提供 QoS 路由保障,许多研究人员对动态 QoS 路由机制进行了相关的理论和实践研究.其中,文献[8-14]提出了一些 QoS 路由算法,文献[15-18]提出了 QoS 路由框架.

文献[8]针对域内网络提出了面向 SDN 的可扩展路由和资源管理模型,SDN 控制器基于此模型实施路由计算、权限控制和资源管理.

文献[9]提出了一种基于 QoS 和动态负载均衡的路由策略,该算法兼顾了流的 QoS 需求,并用近似算法进行多 QoS 约束的最优路径选择.

文献[10-11]都是关注视频业务的 QoS.其中,文献[10]提出了基于 SDN/OpenFlow 的分析型架构来优化转发策略;文献[11]与文献[10]类似,但文献[11]定义了自己的受限最短路径问题,可以动态调整路由策略.

文献[12]修改 Floodlight 控制器的内置模块来优化其对 QoS 的支持性能. 由于 Floodlight 控制器采用的路由算法没有考虑链路之间的代价,因此该文作者在此基础上修改路由算法,考虑了链路负载与 QoS.

文献[13]描述了 SDN 的网络架构并分析了提供多媒体服务的机遇和必要性,因此该文作者提出了一种 SDN 框架,该框架集成了 OpenFlow、网络虚拟化、功能盒以及测试算法的接口. 其中的"QoS 路由算法"模块可以作为运行不同 QoS 路由算法的容器.

文献[14]基于 SDN/OpenFlow 实施了一套 QoS 保障模型,可以实时地监控网络的 QoS 指标并动态地改变业务的路由. 此外,该文作者还提出了面向多业务多约束条件下的路径模型,由此来为多业务分配满足其 QoS 需求的路径.

文献[15]提出了一种 SDN/OpenFlow 控制器的设计方案 OpenQoS,其目的是保障多媒体业务端到端传输的 QoS. 多媒体流的传输路径可以根据当前网络的 QoS 指标来进行动态调整.

文献[16]基于 SDN/OpenFlow 与 OpenNaaS 提出了网络控制层(network control layer,NCL)的概念. 网络控制层包括 SDN 控制器、SDN 监控器和 SDN APP,NCL 的目标是为服务提供端到端的动态 QoS 控制.

文献[17]提出了 QoS 管理机制,能够让 QoS 指标可以自动地配置在 OpenFlow 交换机里. 此外,该文作者基于 AQSDN 设计了数据包上下文感知的 QoS 模型来提高网络的 QoS,此模型在数据包被标记和进入转发队列管理状态时能够考虑数据包的上下文.

文献[18]提出了 OpenFlow 增强的 QoE 保障框架,当多个用户竞争网络资源的时候最大化每个用户的 QoS,关注的是用户级别的公平性和网络的稳定性.

由于 SDN 路由过程的实现必须依赖于交换机的流表管理,当前研究者对流表的管理机制进行了相关研究,但是流表对 QoS 路由的影响却未见具体分析.

2 软件定义的 VANET 体系架构设计

ONF 组织提出的 SDN 系统架构主要由数据平面、控制平面和应用平面构成. 数据平面只需要实现转发和处理数据的功能;控制平面主要负责逻辑控制部分;应用平面由若干个 SDN 应用构成,通过北向接口与 SDN 控制器进行交互.

当前提出的基于 SDN 的 VANET 架构,如引言所述文献[4-7],主要描述了面向无线异构接入的 VANET 架构. 本文在遵循 3 层 SDN 平面划分标准的基础上,充分考虑了 VANET 架构的特点,完善了数据平面的 VANET 异构接入方式以及 SDN 交换层的设计. 本文设计的软件定义的 VANET 体系架构如图 1 所示.

该体系架构的主要元素包括 5 个:

1) SDN APP. 基于控制平面的北向接口,提供

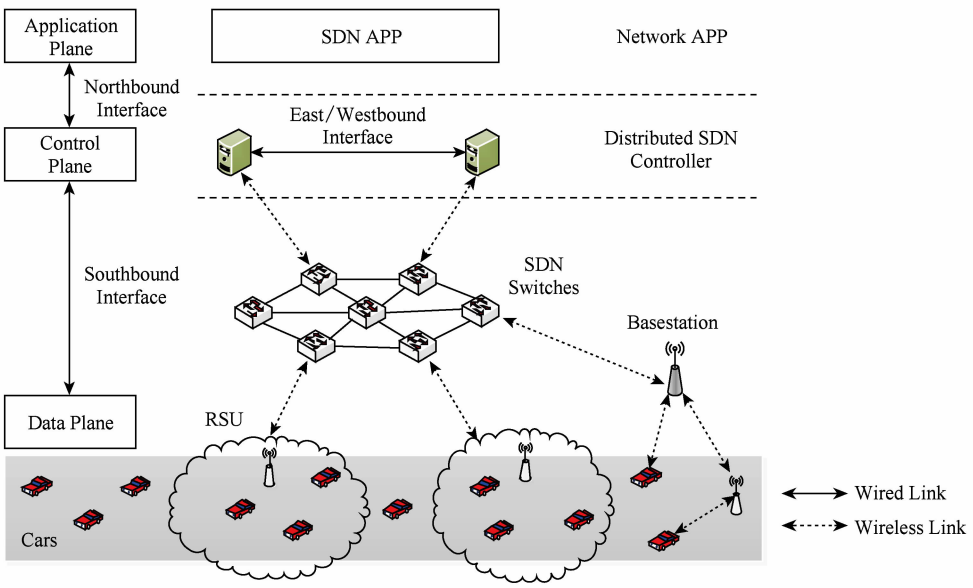


Fig. 1 Software defined VANET framework

图 1 软件定义的 VANET 体系架构

软件支持、网络监控、规则定义等服务,并为 VANET 应用服务提供网络支持.

2) SDN 控制器. 是网络的逻辑控制中心,管理并控制区域内的网络行为. 此外,控制器之间通过东/西向接口协同合作管理网络.

3) SDN 交换机. 接受 SDN 控制器的控制,在数据平面上负责数据的转发和处理.

4) 异构接入网络. 在数据平面上为车辆终端提供异构多样的接入方式. 例如通过 RSU,4 G/LTE 等接入.

5) 终端. 移动车辆作为数据收发的终端.

在上述体系结构中,车辆与控制器通信的方式有 3 种:1) 车辆终端通过蜂窝网络接入 SDN 核心交换层与控制器通信;2) 车辆终端通过 RSU 接入 SDN 核心交换层与控制器通信;3) 车辆终端通过 RSU 接入蜂窝网络再接入 SDN 核心交换层与控制器通信.

软件定义的 VANET 可以提供的服务类型多样,在实际网络部署中,数据层面的 SDN 交换机担负起数据转发的功能,SDN 交换机之间链路的传输带宽、负载等情况直接影响数据传输的性能. 本文对上述架构中 SDN 核心交换层的 QoS 路由问题展开研究,主要考虑 2 点:

1) 由于车辆的移动性,RSU 的任务负载量是动态变化的. 因此,我们需要根据 RSU 当前的任务负载量来考虑动态 QoS 路由,以达到保障接入 RSU 的车辆业务的 QoS.

2) 由于 VANET 服务的实时性,因此我们需要考虑多业务并发的路由机制,以提高动态 QoS 路由的效率,及时保障接入 RSU 的车辆业务的 QoS.

3　流表溢出对 QoS 路由机制的影响

SDN 交换机的流表容量是有限的,每一条流表项代表一条转发规则. OpenFlow1. 3[19] 对流表溢出的处理方法是当流表项达到容量上限时,会丢弃新添加的流表项;而 OpenFlow1. 4[20] 可以自定义丢弃机制,自动清理重要性更低的流表项. 这些丢弃机制都会导致已成功接入业务的 QoS 降低.

因此,我们通过实验研究流表溢出对 QoS 路由机制的影响. 网络中共有 12 台交换机,链路随机生成,并设定链路带宽为 10 Mbps,丢包率为 1%,时延为 10 ms. 客户端通过交换机 S_8 接入网络,通过服务器配置设定了视频传输的带宽为 6 Mbps. 我们实现了以多约束条件的 Dijkstra 算法为路由算法的 QoS 路由机制,图 2 所示的是 QoS 路由机制的选路结果,视频传输选择的路径为 S_1—S_2—S_3—S_8.

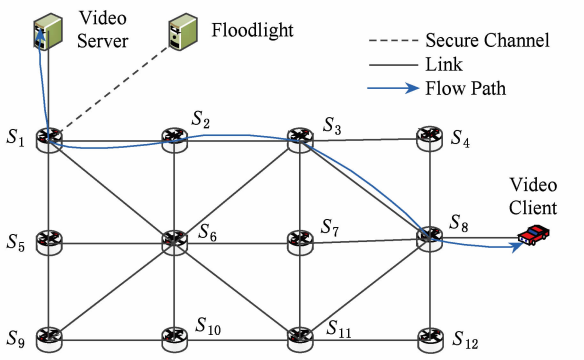


Fig. 2　Routing results of QoS routing mechanism

图 2　QoS 路由机制的选路结果

我们设置 S_2 的流表容量为 100 条,并向 S_2 的流表中插入 100 条流表,此时如若再向 S_2 添加流表项则会发生溢出. 我们重新启动 QoS 路由机制为客户端选路,选路结果仍然为 S_1—S_2—S_3—S_8,当控制器向 S_2 添加流表项时发生了溢出,导致从 S_1 向 S_2 转发的数据包大量丢失,造成了客户端业务吞吐量的下降. 此时,客户端吞吐量变化情况如图 3 所示.

图 3 中流表溢出后吞吐量始终得不到提升的原因是:当 QoS 路由机制发现当前路径不满足客户端

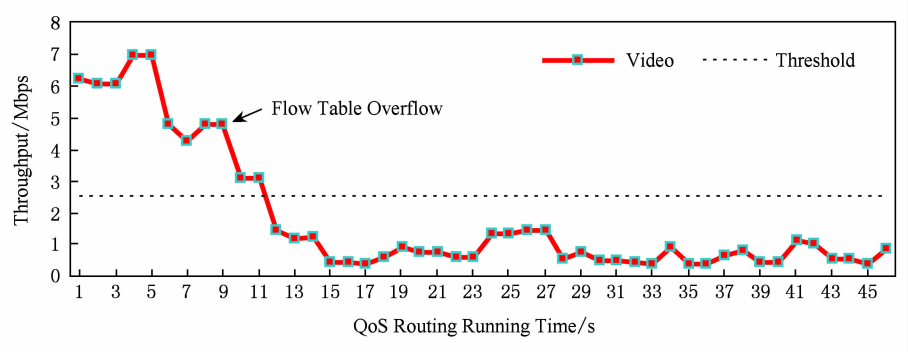


Fig. 3　Throughput changes of client in the flow table overflow situation

图 3　流表溢出情况下的客户端吞吐量的变化情况

的 QoS 需求时会重新启动选路过程,然而 QoS 路由机制根据当前选路指标(丢包率、时延、吞吐量)所选出的路径仍然与之前的路径相同. QoS 路由机制不断为其重新选路,但客户端业务的吞吐量却始终得不到提升,造成了巨大的路由开销.

从第 1 节对动态 QoS 路由机制的相关研究中可以看到,这些研究都没有考虑交换机流表的使用情况,也就不能保证当选路结果上的流表发生溢出时 QoS 机制能够及时做出正确的处理,而 SDN 控制器具有网络全局的视图,具备监测交换机流表用量的能力. 因此,本文提出一种流表用量感知的动态 QoS 路由机制,能够将流表用量作为 QoS 路由的约束条件,解决流表溢出对 QoS 路由机制的影响.

4 流表用量感知的动态 QoS 保障框架

随着移动设备和移动流量的增长,V2V 和 V2I 将会有更多的需求并不断增长. VANET 提供了广泛的服务,为了满足具有不同特点的应用服务的 QoS,本节提出了软件定义的 VANET 下流表用量感知的动态 QoS 保障框架,如图 4 所示:

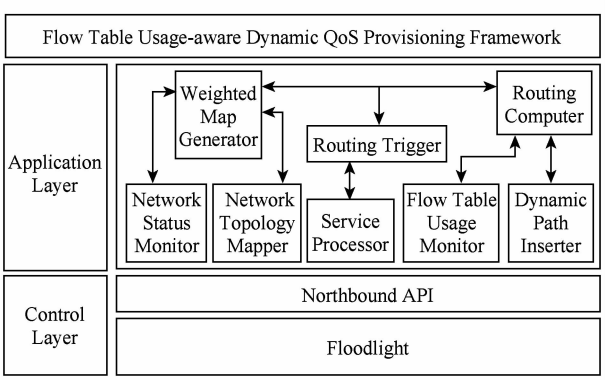


Fig. 4 Flow table usage-aware dynamic QoS provisioning framework

图 4 流表用量感知的动态 QoS 保障框架

该框架在 SDN 的管理层来实现,管理层通过 Floodlight 控制器的北向接口与控制层进行交互,整个框架主要由 8 个模块构成:

1) 网络状态监视器. 实时地搜集链路的状态信息,比如每一条链路当前的吞吐量,并将采集到的数据发送给网络带权图生成器.

2) 网络拓扑映射器. 实时地监测网络拓扑结构的变化,比如,是否有交换机增减、是否有终端机增减,并将数据发送给网络带权图生成器.

3) 网络带权图生成器. 接收网络状态监视器和网络拓扑映射器发送过来的链路与网络拓扑数据,并按照一定的权重计算并生成网络权重视图.

4) 流表用量监控器. 负责实时地监控所有交换机流表的使用情况,并将监控的结果发送给路由计算模块以便路由决策的进行.

5) 路由计算模块. 等待路由触发器的路由指令,根据网络权重视图、多业务的 QoS 需求以及流表使用情况来为多业务并发选路. 此模块执行的具体路由算法,将在第 5 节详细介绍.

6) 业务处理器. 支持业务的动态接入和退出. 当有业务接入时,通知路由触发器,若当前的网络资源不足以满足业务需求则拒绝接入;当有业务退出时,通知路由触发器以便删除业务的路由信息.

7) 路由触发器. 监控成功接入业务的 QoS,在业务 QoS 受影响时触发路由计算,当业务退出时删除业务的路由信息以节省流表资源.

8) 动态路径安装模块. 根据路由计算模块选路的结果下发业务的流表.

软件定义的 VANET 下流表用量感知的动态 QoS 保障框架的工作流程如图 5 所示:

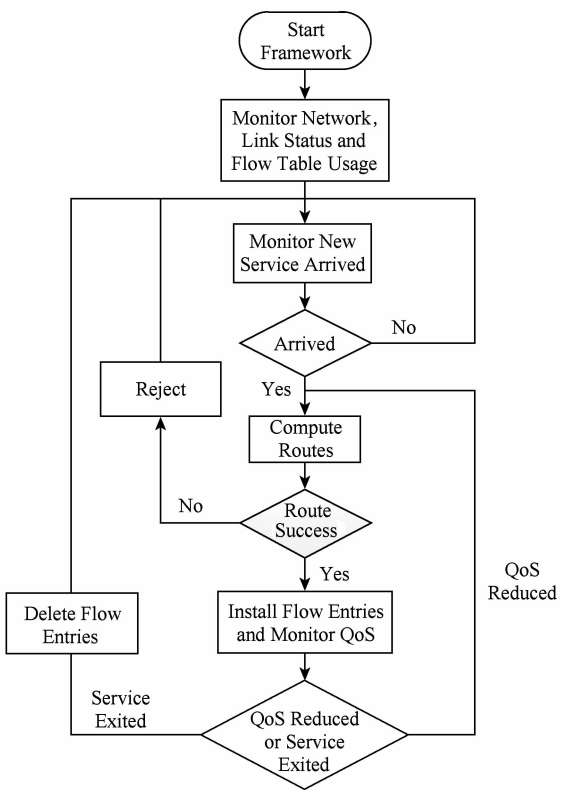


Fig. 5 Workflow of flow table usage-aware dynamic QoS provisioning framework

图 5 流表用量感知的动态 QoS 保障框架工作流程

VANET 下流表用量感知的动态 QoS 保障框架的工作步骤如下:

1) 网络拓扑映射器检测网络的拓扑结构,同时网络状态监视器收集链路的状态信息,它们都将数据发送给网络带权图生成器生成网络权重图.

2) 业务处理器检测新接入的业务,如果有业务接入,就通知路由触发器,路由触发器根据业务的需求触发路由计算;接着,路由计算模块根据网络权重图与流表使用情况执行路由算法,如果路由失败,即网络当前可用资源不能满足新接入业务的 QoS 需求,业务处理器就会拒绝新业务的接入,否则就通过动态路径安装模块解析路由结果并下发流表.

3) 路由触发器监控成功接入业务的 QoS,如果发现业务的 QoS 不满足其需求则重新选路,若业务退出则删除此业务的路由信息.

5　流表用量感知的 QoS 路由模型

VANET 可提供的服务具有不同的特点,有的实时性较高,但带宽占用量较小,比如交通信息、新闻、天气;而有的属于带宽密集型服务,比如视频、语音通话. 因此,构建 QoS 路由模型需要考虑不同的 VANET 业务之间具有不同的 QoS 需求这一特点.

文献[14]提出的路由模型结合了多商品流问题和受限最短路径问题. 多商品流问题是指在多源多目的的情况下,如何为所有商品分配满足其约束的路径并且总代价最小的问题;而结合了受限最短路径问题的多商品流问题是指在多源多目的的情况下,如何为所有商品分配满足其约束条件并且代价最小的最短路径的问题.

因此,文献[14]提出的路由模型适用于本文提出的不同类型 VANET 业务的 QoS 路由问题. 但从第 3 节的实验中可以看出,流表溢出会降低文献[14]提出的 QoS 路由机制的性能,因此本文在文献[14]提出的路由模型的基础上进行了改进,增加了流表用量的限制条件,将流表用量也作为选路的依据,从而进一步提升网络 QoS 保障的性能.

我们根据链路状态参数(丢包率、时延、吞吐量)、业务需求以及流表使用情况来选择合适的路径. 假设网络以有向图 $G=(N,E)$ 表示,N 是节点集合,E 是节点间链路集合,节点 i 和节点 j 之间链路带宽表示为 b_{ij},时延为 d_{ij},丢包率为 p_{ij},每条流的开销为 c_{ij},网络中需要路由的流的数量为 k,节点 i 的流表使用量为 u_i. 根据上述假设,我们构建了线性规划模型,模型的参数如表 1 所示. 优化的目标是要找到满足所有业务约束条件的最小代价和的路径.

Table 1　The Parameters of Model

表 1　模型参数

Parameters	Definition
$b_{ij}\geqslant 0$	Available bandwidth on the $link(i,j)$
$c_{ij}\geqslant 0$	Cost of the $link(i,j)$
$\alpha\geqslant 0$	Scale factor for the delay
$\beta\geqslant 0$	Scale factor for the packet loss
$s_k\in N$	Source of the flow k
$t_k\in N$	Destination of the flow k
$f_k\geqslant 0$	Amount of the flow k to be sent from s_k to t_k
$P_{\max}^k\geqslant 0$	Maximum acceptable packet loss
$p_{ij}\geqslant 0$	Packet loss on the $link(i,j)$
$D_{\max}^k\geqslant 0$	Maximum acceptable delay
$d_{ij}\geqslant 0$	Delay on the $link(i,j)$
$U_{\max}^i\geqslant 0$	Maximum flow table volume
$u_i\geqslant 0$	Flow table volume of u_i
$B^k\geqslant 0$	Bandwidth required by the service k

变量:$x_{ij}^k\geqslant 0$,业务 k 在链路(i,j)上流的总和;

目标函数:

$$\min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}x_{ij}^k; \tag{1}$$

约束条件:

$$\sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = \begin{cases} f_k, & i=s_k,\\ -f_k, & i=t_k,\\ 0, & i\neq s_k,t_k, \end{cases} \quad \forall i\in N,\forall k\in K, \tag{2}$$

$$\sum_{(i,j)\in A} p_{ij}x_{ij}^k \leqslant p_{\max}^k, \forall k\in K, \tag{3}$$

$$\sum_{(i,j)\in A} d_{ij}x_{ij}^k \leqslant D_{\max}^k, \forall k\in K, \tag{4}$$

$$\sum_{k\in K} B^k x_{ij}^k \leqslant b_{ij}, \forall (i,j)\in A, \tag{5}$$

$$x_{ij}^k\in\{0,1\},\ u_i<U_{\max}^i,\ \forall (i,j)\in A,\forall k\in K, \tag{6}$$

$$x_{ij}^k = x_{ji}^k = 0,\ u_i = U_{\max}^i,\ (i,j)\in A,(j,i)\in A,\forall k\in K. \tag{7}$$

式(1)是目标函数,我们需要找到满足所有业务约束条件的最小代价和的路径. 对于路径代价的计算,我们主要考虑了丢包率和时延 2 个因素,并分别赋予它们对路径代价影响的权重:

$$c_{ij}=\alpha d_{ij}+\beta p_{ij},\ \forall (i,j)\in A,$$

其中,α 和 β 分别为时延和丢包率对链路代价影响的权重,我们可以通过管理这些参数来满足不同类型业务的需求. 时延的计算包括 4 个组成部分,定义为

$$d_{\text{nodal}}=d_{\text{proc}}+d_{\text{queue}}+d_{\text{trans}}+d_{\text{prop}},$$

总时延由 d_{nodal} 表示；d_{proc}，d_{queue}，d_{trans}，d_{prop} 分别表示处理时延、排队时延、发送时延和传播时延. 式(2)为流量守恒定律，保证流的输入和输出守恒；式(3)～(5)分别为最大可接受丢包率、最大可接受时延和链路的带宽容量限制；式(6)(7)是对变量取值范围的限制，对于一条链路而言，我们有选和不选 2 种选择，分别用 1 和 0 表示. 此外式(6)(7)还考虑了对节点 i 流表溢出时的约束，即当节点 i 的流表溢出时不再选择经过节点 i 的链路.

实时业务通常会对丢包率、时延、吞吐量、业务代价等多个参数同时提出要求，当这些参数相互独立时，选择满足多个参数限制的路由就成为 NP-Complete[21] 问题，对于求解上述线性规划模型的算法时间复杂度的评估，变量数量与边的数量和业务数量有关，等于 $|A \| K|$，而约束条件的数量也与边数和业务数量相关，等于 $|N \| K|+|A|+|K|$. 本文采用了 IP_Solve[22] 求解器来求解本文提出的线性规划模型，求解器采用了改进的分支定界法求解线性规划模型，平均时间复杂度达到指数级，虽然在大规模网络中计算开销较大，但是能够在一般规模网络中快速找到最优解，而计算开销相对较小的智能优化算法一般只能找到近似解. 由于在大规模软件定义 VANET 中，通常会由多个控制器实现分布式控制，因此，单个控制器所控制的网络规模一般不会太大，因此本文采用了求解器求解的方式，以便能够快速找到最优解，同时满足 VANET 环境的时延敏感的特性.

6 实验评估

本文实验采用 Mininet2. 2. 1[23] 作为仿真平台，Floodlight1. 1[24] 作为 SDN 控制器，OpenFlow1. 3 作为控制器与数据平面交互的南向接口协议，并基于控制器的北向接口实现了第 4 节所提出的 QoS 保障框架中的所有模块. 网络中共有 12 台 SDN 交换机，交换机之间的链路随机生成，并设定链路的带宽为 30 Mbps，丢包率为 1%，时延为 10 ms. 视频服务器采用 VLC Media Player，通过 cvlc 命令推送高清视频流到视频客户端，FTP 服务器采用 Apache FtpServer，在 FTP 客户端通过 wget 命令下载文件，同时我们使用 Iperf 工具实现链路拥塞，服务器通过交换机 S_1 接入网络，并通过服务器配置设定了文件传输的平均带宽为 5 Mbps，视频传输的平均带宽为 6 Mbps. 实验一共分为 2 部分：1)在流表未溢出的场景下对本文提出的 QoS 路由机制的性能评估；2)在流表溢出的场景下，对本文提出的 QoS 路由机制与文献[14]提出的 QoS 路由机制进行性能比较.

6.1 动态 QoS 路由机制性能评估

1) 客户端选路与吞吐量性能分析

在流表未溢出的场景下，首先评估了本文所提的 QoS 路由机制对业务 QoS 的保障性能. 视频客户端和 FTP 客户端分别通过 S_8 与 S_{12} 接入网络. 我们通过启用和不启用本文所提出的 QoS 路由机制 2 种情况来观察客户端选路和吞吐量的变化情况，实验结果如图 6 和图 7 所示. 在该实验场景中，我们通过在一定时间内分别向链路 S_6—S_{11} 与 S_1—S_2 增加负载 FTP-Loader 与 Video-Loader 来改变该链路的拥塞情况，通过使 S_6—S_{11} 和 S_1—S_2 链路拥塞而让 FTP 客户端和视频客户端的 QoS 受到影响. 图 6(a)所示的是未启用 QoS 路由机制，是采用默认最短路径路由机制场景下的选路结果，可以看出，即使

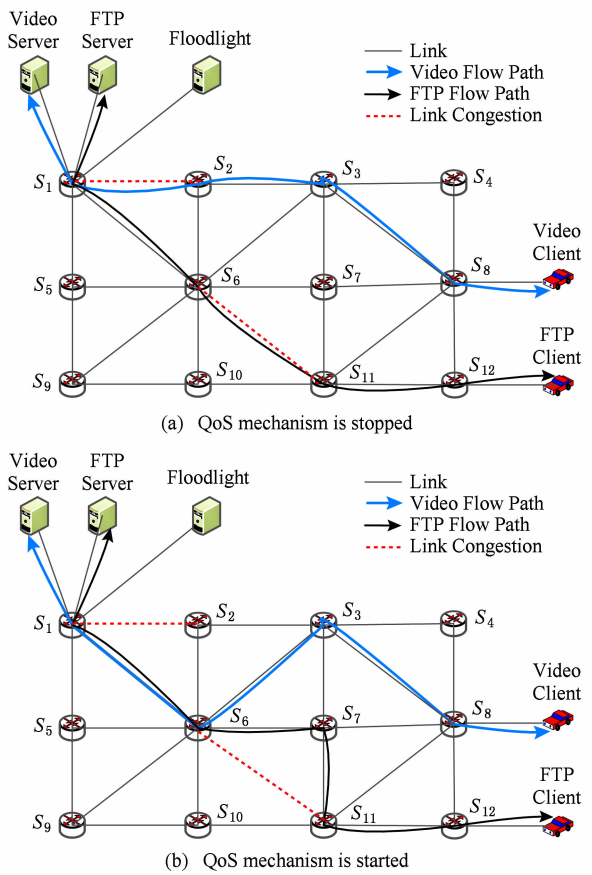


Fig. 6 Routing results of starting and stopping QoS mechanism in the link congestion situation

图 6 链路拥塞时启用和关闭 QoS 路由机制的选路情况

链路 S_6—S_{11} 和 S_1—S_2 发生了拥塞,控制器也不会为客户端调整路径,仍然按照原始路径转发数据;图 6(b)所示的是启用 QoS 路由机制的选路情况,由于 QoS 路由机制检测到了客户端 QoS 受到链路拥塞的影响,所以及时地为客户端调整了路径,视频服务选择了非拥塞路径 S_1—S_6—S_3—S_8 来传输视频数据,而 FTP 服务选择非拥塞路径 S_1—S_6—S_7—S_{11}—S_{12} 来传输数据.

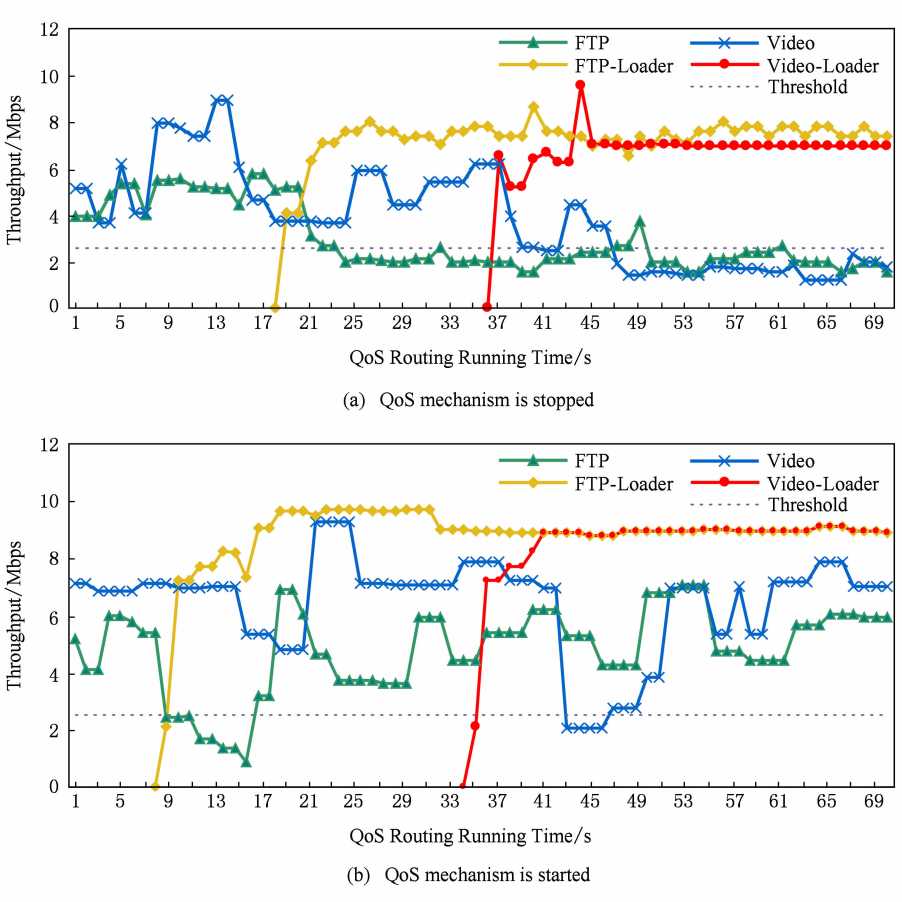


Fig. 7　Thoughput changes of client of starting and stopping QoS mechanism

图 7　启动和关闭 QoS 路由机制时客户端吞吐量的变化情况

图 7 所示的是客户端吞吐量的变化情况,通过实验统计设定了触发重新选路的最低吞吐量阈值为 2.5 Mbps,即当 QoS 路由机制检测到客户端吞吐量小于 2.5 Mbps 时,会为客户端重新选路. 图 7(a)所示的是未启用 QoS 路由机制时客户端吞吐量的变化情况,FTP-Loader 和 Video-Loader 分别在 18 s 和 36 s 时加入到网络中来,造成了 FTP 客户端和视频客户端的平均吞吐量分别下降到 2.13 Mbps 和 2.06 Mbps,均低于最低吞吐量阈值;图 7(b)所示的是启用 QoS 路由机制后客户端吞吐量的变化情况,当 FTP-Loader 和 Video-Loader 分别在 18 s 和 36 s 时加入到网络中来时,FTP 客户端和视频客户端的平均吞吐量首先出现了下降,但是当下降到阈值 2.5 Mbps 之后,吞吐量又迅速开始回升. FTP 客户端吞吐量在 16 s 时开始回升,视频客户端吞吐量在 44 s 时开始回升,并都回升到满足其各自 QoS 的需求. 因此,从实验结果看到,本文所提 QoS 路由机制能够检测到客户端吞吐量低于阈值并为其重新分配满足其 QoS 需求的路径,使得客户端的吞吐量得到了回升.

2) 业务动态接入下的性能分析

在流表未溢出的场景下,本文还对动态 QoS 路由机制在业务动态接入的情况下进行了性能测试,观察业务动态接入的并发度与系统吞吐量之间的关系. 当 QoS 路由机制启动之后会不停地检测业务的动态接入,我们设置了每隔 10 s 向 S_{12} 动态接入 2 个 FTP 业务,QoS 路由机制为每批接入的业务分配合适的路径并监控接入业务的 QoS.

图 8 所示的是随着业务接入数量的增加,业务并发度与系统吞吐量变化之间的关系. 从图 8 可以看出,随着业务数量并发的增加,系统吞吐量呈增长的趋势. 该实验说明了本文所提 QoS 保障框架能够支持业务的动态加入,并同时保障多个业务的 QoS.

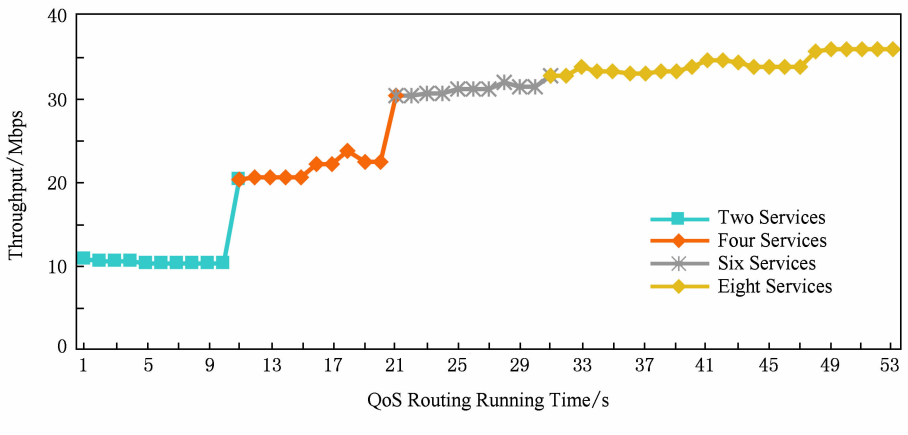


Fig. 8 Relationship between business concurrency and change of throughput of system

图 8 业务并发度与系统吞吐量变化的关系

6.2 流表溢出场景下的性能分析

在流表溢出的情况下,我们对本文提出的 QoS 路由机制与文献[14]提出的 QoS 路由机制进行了性能的比较. 图 9 和图 10 分别所示的是 S_2 流表溢出时 QoS 路由机制的选路情况和客户端吞吐量的变化情况. 我们设置 S_2 的流表容量为 100 条,并向 S_2 的流表中插入 100 条流表,此时如若再向 S_2 添加流表项则会发生溢出. 文献[14]提出的 QoS 路由机制选路的结果为 S_1—S_2—S_3—S_8,当控制器向 S_2 添加流表项时发生了溢出,但该 QoS 路由机制不会检测到流表溢出的情况,会保持当前的选路结果;而本文提出的 QoS 路由机制能够检测到流表溢出,并及时为客户端更换了路径,所选路径为 S_1—S_6—S_3—S_8.

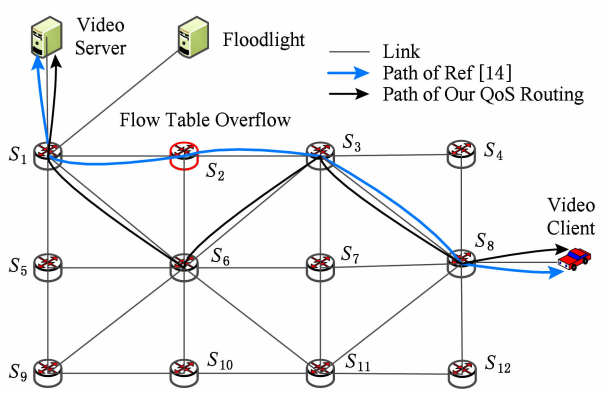


Fig. 9 Routing choice comparison of QoS routing mechanism in the flow table overflow situation of S_2

图 9 S_2 流表溢出时 QoS 路由机制的选路情况对比

从图 10 中可以看出,对于文献[14]提出的 QoS 路由机制,由于 S_2 流表的溢出,S_1 向 S_2 转发的数据包大量丢失,从而造成了客户端吞吐量持续下降并低于阈值,导致 QoS 得不到保障. 然而,本文提出的动态 QoS 路由机制及时感知到了流表的使用情况,避开了经过流表溢出的交换机的链路,选择的视频传输路径为 S_1—S_6—S_3—S_8,能够维持视频客户端正常的吞吐量水平.

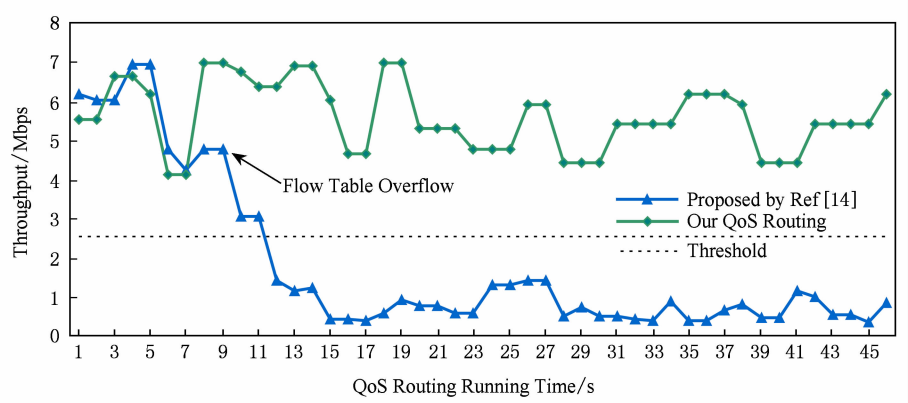


Fig. 10 Throughput changes of video client in the flow table overflow situation of S_2

图 10 S_2 流表溢出时视频客户端吞吐量的变化情况

7 总　　结

软件定义的 VANET 架构可以为 VANET 中各类服务提供更有效的支持.本文首先设计了一种面向异构多网接入的软件定义 VANET 架构,完善了现有工作中数据平面的设计;接着以此架构为基础提出了一种流表用量感知的动态 QoS 保障框架,并提出了多业务流多约束条件下流表用量感知的 QoS 路由模型,为 VANET 应用服务提供并发的动态 QoS 路由,能够满足多业务对各自丢包、时延和吞吐量的要求,还能够感知交换机流表的使用情况,避免流表溢出对 QoS 路由的影响,进一步提高了网络 QoS 保障的性能.

在保障应用 QoS 的同时,动态 QoS 路由机制也带来了网络开销.网络开销主要包括 QoS 路由机制启动时采集网络拓扑和链路信息所带来的开销以及启动后定时采集业务 QoS 指标和流表用量的开销.QoS 路由机制启动时,需要采集交换机的基本信息、链路的基本信息和网络拓扑结构,这部分开销的大小会随着交换机和链路数量的增加而增加;当 QoS 路由机制启动后,会定时(秒级)并发地采集业务的 QoS 指标和流表使用情况,这部分的开销不仅与交换机、链路的数量有关,还与系统的并发负载能力有关.

在未来的工作中,我们将考虑不同业务具有不同的优先级,并将其与多业务流多约束条件下流表用量感知的 QoS 路由机制相结合,进一步提升在 VANET 场景下网络对各类应用的 QoS 保障能力.

参 考 文 献

[1] Cunha F D D, Boukerche A, Villas L, et al. Data communication in VANETs: A survey, challenges and applications [J]. Journal of Biological Chemistry, 2014, 274(39): 27605-27609

[2] Kirkpatrick K. Software-defined networking [J]. Communications of the ACM, 2013, 56(9): 16-19

[3] Zhang Chaokun, Cui Yong, Tang Heyi, et al. State-of-the-art survey on software-defined networking (SDN) [J]. Journal of Software, 2015, 26(1): 62-81 (in Chinese)
(张朝昆, 崔勇, 唐翯祎, 等. 软件定义网络(SDN)研究进展 [J]. 软件学报, 2015, 26(1): 62-81)

[4] Ku Ian, Lu You, Gerla M, et al. Towards software-defined VANET: Architecture and services [C] //Proc of Ad Hoc Networking Workshop. Piscataway, NJ: IEEE, 2014: 103-110

[5] Li He, Dong Mianxiong, Ota K. Control plane optimization in software-defined vehicular ad hoc networks [J]. IEEE Trans on Vehicular Technology, 2016, 65(10): 7895-7904

[6] He Zongjian, Cao Jiannong, Liu Xuefeng. SDVN: Enabling rapid network innovation for heterogeneous vehicular communication [J]. IEEE Network, 2016, 30(4): 10-15

[7] Truong N B, Lee G M, Ghamri-Doudane Y. Software defined networking-based vehicular ad hoc network with fog computing [C] //Proc of 2015 IEEE Int Symp on Integrated Network Management. Piscataway, NJ: IEEE, 2015: 1202-1207

[8] Celenlioglu M R, Mantar H A. An SDN based intra-domain routing and resource management model [C] //Proc of IEEE Int Conf on Cloud Engineering. Piscataway, NJ: IEEE, 2015: 347-352

[9] Sun Jie, Li Li, Shen Subin. A routing strategy based on QoS and dynamic load balancing [J]. Computer Technology and Development, 2016, 26(11): 188-194 (in Chinese)
(孙杰, 李莉, 沈苏彬. 一种基于 QoS 和动态负载均衡的路由策略[J]. 计算机技术与发展, 2016, 26(11): 188-194)

[10] Egilmez H E, Civanlar S, Tekalp A M. An optimization framework for QoS-enabled adaptive video streaming over openflow networks [J]. IEEE Trans on Multimedia, 2013, 15(3): 710-715

[11] Civanlar S, Parlakisik M, Tekalp A M, et al. A QoS-enabled openflow environment for scalable video streaming [C] //Proc of 2010 IEEE GlobeCom Workshops. Piscataway, NJ: IEEE, 2010: 351-356

[12] Adami D. A network control application enabling software-defined quality of service [C] //Proc of 2015 IEEE Int Conf on Communications. Piscataway, NJ: IEEE, 2015: 6074-6079

[13] Caraguay V, Leonardo N, Fern P, et al. Framework for optimized multimedia routing over software defined networks [J]. Computer Networks, 2015, 92(P2): 369-379

[14] Ongaro F, Cerqueira E, Foschini L, et al. Enhancing the quality level support for real-time multimedia applications in software-defined networks [C] //Proc of 2015 Int Conf on Computing, Networking and Communications. Piscataway, NJ: IEEE, 2015: 505-509

[15] Egilmez H E, Dane S T, Bagci K T, et al. OpenQoS: An openflow controller design for multimedia delivery with end-to-end quality of service over software-defined networks [C] //Proc of the 2012 Asia Pacific Signal & Information Processing Association Annual Summit and Conf. Piscataway, NJ: IEEE, 2012: 1-8

[16] Bueno I, Aznar J I, Escalona E, et al. An OpenNaaS based SDN framework for dynamic QoS control [C] //Proc of Future Networks and Services. Piscataway, NJ: IEEE, 2013: 1-7

[17] Wang Wendong, Qi Qinglei, Gong Xiaoyang, et al. Autonomic QoS management mechanism in software defined network [J]. China Communications, 2014, 11(7): 13-23

[18] Georgopoulos P, Elkhatib Y, Broadbent M, et al. Towards network-wide QoE fairness using openflow-assisted adaptive video streaming [C] //Proc of the 2013 ACM SIGCOMM Workshop on Future Human-Centric Multimedia Networking. New York: ACM, 2013: 15-20

[19] Open Networking Foundation. Openflow switch specification 1.3.0 [EB/OL]. [2012-07-25]. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf

[20] Open Networking Foundation. Openflow switch specification 1.4.0 [EB/OL]. [2013-11-15]. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf

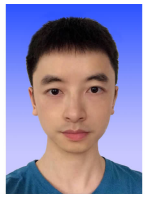
[21] Wang Zheng, Crowcroft J. Quality-of-service routing for supporting multimedia applications [J]. IEEE Journal on Selected Areas in Communications, 1996, 14(7): 1228-1234

[22] Sourceforge. Lpsolve: Mixed integer linear programming (MILP) solver [EB/OL]. [2016-09-14]. https://sourceforge.net/projects/lpsolve

[23] Mininet. The mininet documentation [EB/OL]. [2016-08-20]. https://github.com/mininet/mininet/wiki/Documentation

[24] A Big Switch Networks Sponsored Community Project. The floodlight documentation [EB/OL]. [2016-02-07]. https://floodlight.atlassian.net/wiki/display/floodlightcontroller/Getting+Started

Fu Bin, born in 1978. PhD. Lecturer in the College of Computer Science and Electronic Engineering, Hunan University. Member of CCF. Her main research interests include wireless communication, software defined networks and Internet of things.



Zha Lijia, born in 1991. Master candidate in the College of Computer Science and Electronic Engineering, Hunan University. His research interest is software defined networks (charlie91825@icloud.com).

Li Renfa, born in 1956. Professor and PhD supervisor in the College of Computer Science and Electronic Engineering, Hunan University. Senior member of CCF. His main research interests include embedded system, cyber-physical system and wireless networks (lirenfa@hnu.edu.cn).

Xiao Xiongren, born in 1978. PhD candidate. Lecturer in the College of Computer Science and Electronic Engineering, Hunan University. Member of CCF. His main research interests include dependable distributed systems, embedded systems and computer networks (xxr@hnu.edu.cn).