doi:10.11887/j.cn.202005004　　http://journal.nudt.edu.cn

基于切比雪夫神经网络的软件定义卫星网络智能路由策略*

梁　俊,孙伟超,肖　楠,陈威龙,郭子桢
(空军工程大学 信息与导航学院,陕西 西安　710077)

摘　要:针对现有的软件定义卫星网络中流表占用的三态内容寻址存储器空间不断增加,复杂的流表项查找、匹配过程导致路由转发效率降低,无法满足多样化应用需求的问题,提出基于神经网络的软件定义卫星网络智能路由架构。控制器通过训练神经网络获取数据流的传输模式,并用训练后的神经网络代替流表,在此基础上提出基于 Chebyshev 神经网络的智能路由策略,交换机根据数据流的业务类型预测其转发路径,以满足卫星网络应用的服务质量要求。仿真结果表明:所提路由策略显著减少了占用的三态内容寻址存储器存储空间,提高了路由效率。

关键词:卫星网络;软件定义网络;神经网络;路由算法;服务质量
中图分类号:TP915　　**文献标志码**:A　　**文章编号**:1001-2486(2020)05-023-08

Intelligent routing strategy for software-defined satellite network based on Chebyshev neural network

LIANG Jun, *SUN Weichao*, *XIAO Nan*, *CHEN Weilong*, *GUO Zizhen*
(Information and Navigation College, Air Force Engineering University, Xi'an 710077, China)

Abstract: In the existing software-defined satellite network, the storage space of ternary content addressable memory occupied by the flow table is increasing, and the complex flow entry lookup and matching processes will bring about reduced route forwarding efficiency, which cannot meet the requirements of diverse application requirements. A neural network-based software-defined satellite network intelligent routing framework was proposed. The controller acquired the transmission mode of the data flow by training the neural network, and replaced the flow table with the trained neural network. Based on this framework, an intelligent routing strategy based on Chebyshev neural network was proposed. The switch predicts the forwarding path of data flow according to the service type of data flow to meet the quality of service requirements of satellite network applications. The simulation results show that the proposed routing strategy significantly reduces the occupied storage space of ternary content addressable memory and improves the routing efficiency.

Keywords: satellite network; software defined network; neural network; routing algorithm; quality of service

传统的卫星网络依赖于封闭和有计划的体系架构[1],面临配置更新、新通信和网络技术的引入、差异化服务的提供以及卫星和地面网络集成等挑战[2-3]。软件定义网络(Software Defined Network, SDN)实现了控制平面和数据平面的完全解耦[4],具有灵活性、可编程性和逻辑集中性等特点,能够提高网络资源利用率[5],简化网络管理流程,降低运营成本,促进了网络的演进和创新[6]。

国内外许多学者对在卫星网络中引入 SDN 进行了探索。文献[7]针对卫星网络通信存在长时延和频繁的地面-卫星切换以及卫星网络配置成本较高的问题,将 SDN 与卫星网络相结合,增加了网络的灵活性以及可扩展性,降低了星上处理开销。文献[8]提出了一种聚合 SDN 的新一代空天地一体化网络,网络部署方式灵活,网络可扩展性强,资源利用率高。文献[9]提出了基于 SDN 架构的低地球轨道(Low Earth Orbit, LEO)卫星网络,将地面网络控制中心作为 SDN 控制器,计算路由策略,优化了资源利用方式。文献[10]针对传统卫星系统架构导致的卫星通信系统的管理和配置效率低下、管理不灵活等问题,提出了一种基于 SDN 和网络功能虚拟化的空间地面综合多层卫星通信网。

然而以上文献均采用 OpenFlow 协议作为 SDN 南向编程接口。随着 OpenFlow 版本的推进,不断扩展的"多级流表"以及复杂的流表项查找、匹配问题给资源受限的 LEO 卫星的存储和处理

* **收稿日期**:2019-03-26
基金项目:国家部委基金资助项目(30501030301);国家自然科学基金资助项目(61501496)
作者简介:梁俊(1962—),男,江苏江宁人,教授,硕士,博士生导师,E-mail:1037008913@qq.com

能力带来很大挑战[11-12]。

神经网络(Neural Network, NN)是一种自学习、自调整的智能工具,能够进行大规模的并行计算[13]。NN 模型可以从海量数据中获取目标,在分类、模式识别、预测等多个领域都有着广泛的应用。在 SDN 中,通过逻辑集中的控制器容易获得大量的流量数据和网络信息。文献[14]首次提出一种在 SDN 中,用径向基函数(Radial Basis Function, RBF)神经网络代替流表的方法,节省了存储空间。然而支持 RBF 算法的 NN 各个权值不能完全独立地求解,难以在星上并行运算,且需要进行烦琐的高阶矩阵求逆运算,难以适应星上路由的要求。以正交多项式为基础的 Chebyshev 神经网络函数逼近性能优越,基函数的正交性保证 NN 在多项式的节点处权值求解迅捷[15]。

通过以上分析,提出一种基于 Chebyshev 神经网络的 SDN 卫星网络智能路由架构。通过训练神经网络获取数据流的传输模式用于预测路由,同时,用 NN 代替流表以节省三态内容寻址存储器(Ternary Content Addressable Memory, TCAM)的存储空间,提高数据流的路由和转发效率。

1 系统架构

采用单层卫星多层控制器(Single-layer Satellite Multiple-layer Controller, SSMC)架构,将基于 SDN 的卫星网络架构分为三层:地球同步轨道(Geostationary Earth Orbit, GEO)卫星控制平面、LEO 卫星数据平面、地面控制平面,如图 1 所示。各平面的功能模块如图 2 所示。

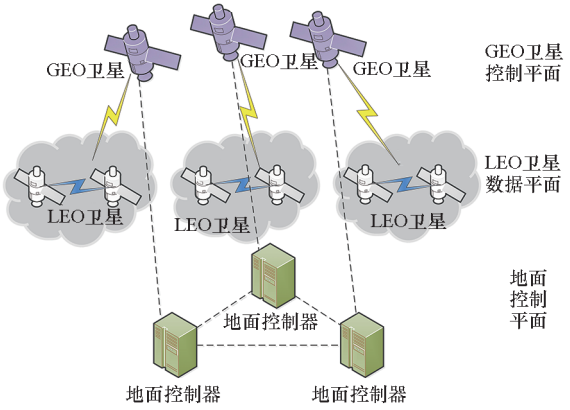

图 1 系统架构

Fig. 1 System architecture

1.1 控制平面

控制平面由 GEO 卫星控制层和地面控制层

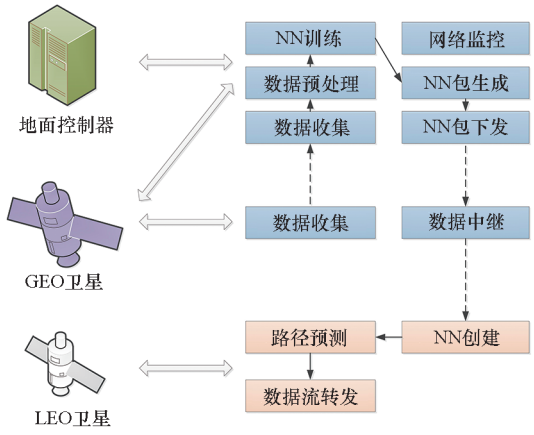

图 2 各平面功能

Fig. 2 Various plane functions

组成,主要功能是利用其强大的计算能力,收集和处理网络流信息并通过训练 NN 获取数据流传输模式。其中,GEO 卫星为局部控制器,只负责管理自身覆盖区域内的 LEO 卫星交换机,当 LEO 卫星发出路由请求时,如果数据流的源和目的节点都在同一颗 GEO 卫星覆盖区域内,则 GEO 卫星发挥控制器的功能;否则,GEO 卫星作为 LEO 卫星与地面控制器通信的中继节点,将路由请求消息转发到地面控制器。地面控制器为全局控制器,掌握网络全局拓扑变化、流量分布、节点负载等信息[16]。

地面控制器的核心功能模块描述如下:

1)网络监控模块。负责实时监控 LEO 卫星网络全局的状态和拓扑变化。

2)数据收集模块。负责收集 GEO 卫星传来的路由请求消息,从中提取数据流的源、目的地、业务类型和路径信息,用于训练 NN。

3)数据预处理模块。负责对收集到的数据进行预处理,构造用于 NN 训练和测试的样本集。包括数据流路径的预处理、训练集和测试集的构造、样本集的归一化等。

4)NN 训练模块。负责训练 NN,并根据历史流量数据获得数据流的传输模式。

5)NN 包生成模块。负责利用 NN 参数创建 NN 包,具体格式见 1.4 节。

6)NN 包下发模块。负责将 NN 包发送给管理源和目的节点的 GEO 卫星控制器,并下发给所有相关 LEO 卫星。

GEO 卫星控制器除了具有以上功能模块,还具有数据中继模块,作为地面控制器和 LEO 卫星的中继节点。此外,GEO 卫星控制器的网络监控、数据收集模块都是面向自身覆盖区域内的

LEO 卫星网络。

1.2　数据平面

数据平面由 LEO 卫星交换机组成,核心功能是利用控制器下发的 NN 预测数据流的路径,对数据流进行路由转发。其核心功能模块描述如下:

1)NN 创建模块。LEO 卫星交换机接收到 NN 包后,根据存储在 NN 包中的结构参数创建 NN。

2)路径预测模块。当接收到用户发送的业务数据后,从中提取三个元组信息(源、目的地、业务类型)作为 NN 的输入进行路径预测。

3)数据流转发模块。根据路径预测模块得到的路径转发数据包。

1.3　NN 预处理

在训练 NN 之前,需要对其进行预处理,主要包括样本集的归一化以及训练集和测试集的构造。

当有数据流经过网络时,控制器从数据流中提取源节点 *Src*、目的节点 *Dst*、业务类型 *Type* 用于构造样本集,并记录数据流的传输路径 *Path*。进行 NN 预处理时,以样本(*Src*,*Dst*,*Type*)作为 NN 的输入,以 *Route* 作为输出。随机选取 30% 的样本作为训练集,其余 70% 的样本作为测试集[14]。

为了提高 *NN* 的学习效果,对样本(*Src*,*Dst*,*Type*)进行归一化处理。

$$Src' = \frac{Src - Src^{\min}}{Src^{\max} - Src^{\min}} \tag{1}$$

$$Dst' = \frac{Dst - Dst^{\min}}{Dst^{\max} - Dst^{\min}} \tag{2}$$

$$Type' = \frac{Type - Type^{\min}}{Type^{\max} - Type^{\min}} \tag{3}$$

其中,Src'、Dst'、$Type'$分别是 *Src*、*Dst*、*Type* 的归一化值。$Src^{\max}$、$Src^{\min}$,$Dst^{\max}$、$Dst^{\min}$,$Type^{\max}$、$Type^{\min}$分别是 *Src*、*Dst*、*Type* 的最大值和最小值。每个变量的归一化值在 0 和 1 之间。

在样本集归一化后,控制器使用训练集训练 NN,直到预测成功率达到预设标准,NN 训练结束。之后,控制器依据训练好的 NN 创建 NN 包,并将 NN 包下发给相关 LEO 卫星交换机。

1.4　NN 包

为了将 NN 包与一般的数据包区分开来,设计了 NN 包的特定格式。NN 包由 NN 的结构参数组成,包括输入层节点个数、隐藏层节点个数、输出层节点个数、输入层与隐藏层连接权值、隐藏层之间的连接权值、隐藏层与输出层之间的连接权值等。

以 Chebyshev 神经网络包为例,Chebyshev NN 输入层和隐藏层之间的连接权值为固定值 1,且只有一个隐藏层。因此,不考虑 Chebyshev NN 包中输入与隐藏层之间的连接权值、隐藏层之间的权值。

M、*N* 和 *K* 分别表示输入、隐藏和输出层中的节点数,O_i 表示 *i* 次正交多项式,W_{ij}表示第 *i* 个隐藏节点与第 *j* 个输出节点之间的权值。Chebyshev NN 包格式如图 3 所示。

M	N	K	O_i	W_{ij}

图 3　Chebyshev NN 包的格式

Fig. 3　Format of the Chebyshev NN packet

2　Chebyshev 神经网络智能路由策略

传统的多层感知器(Multi-Layer Perceptron, MLP)神经网络采用的误差反向传播(Back Propagation, BP)算法训练时间长,易陷入局部最小值;而支持向量机(Support Vector Machine, SVM)和 RBF 算法的各个权值不能完全独立地被求解,且需要进行烦琐的高阶矩阵求逆运算,难以在星上并行计算。因此这些训练算法难以适应星上路由的要求[15]。

Chebyshev 神经网络是一种基于 Chebyshev 正交多项式的神经网络,逼近性能优越,基函数的正交性保证神经网络在多项式的结点处权值求解迅捷,能够较好地满足卫星网络的需求。基于第 1 节的系统架构,提出了一种基于 Chebyshev 神经网络的智能路由策略(Intelligent Routing Strategy based on Chebyshev Neural Network, IRSCNN)。

2.1　Chebyshev 神经网络模型

一元 Chebyshev NN 由一个输入层、一个隐藏层和一个输出层组成,如图 4 所示[15]。其中 $O_j(x_i)$代表 *j* 次正交多项式,其系构成了 NN 的单隐藏层结构,W_{ij}是 $O_j(x_i)$的权值。NN 训练的目标是通过训练集,用已知的正交多项式对未知进行线性逼近,即求 y_i。

$$y_i = f(x_i) = \sum_{j=0}^{N} W_{ij} O_j(x_i) \tag{4}$$

以(*Src*,*Dst*,*Type*)作为 Chebyshev NN 的输入,得到数据流的路径 *Route* 作为 NN 的输出。因此需考虑三元 Chebyshev NN 模型,设输入为三维向量$[x_1, x_2, x_3]^{\mathrm{T}}$,输出是关于三元函数$y_i = f(x_1, x_2, x_3)$的一个精确逼近。根据文献[15]得:

$$f(x_1,x_2,x_3) = O_{N_1N_2N_3}(x_1,x_2,x_3) = \sum_{j_1=0}^{N_1}\sum_{j_2=0}^{N_2}\sum_{j_3=0}^{N_3}\left(W_{j_1j_2j_3}\prod_{j=1}^{3}O_{jk_j}(x_j)\right) \quad (5)$$

式中，f 是待模拟的函数，O 是用以逼近 x_j 的 k_j 次多项式，$W_{j_1j_2j_3}$ 是该正交多项式组合的权值。

另外，神经网络的输出是一个或一组标量值，而 NN 的输出应该能表示成一条最优路径，因此需要一种映射方法来形成标量值和最优路径之间的映射。采用路径标量表示[15]。

$$R_i = \{N_{ij} \| N_{ij} - Src|_{\text{hop}} = i\} \quad i = 1,2,\cdots,K \quad (6)$$

式中，R_i 为所有到源节点 Src 链路距离为 i 的节点 N_{ij} 组成的集合，j 为第 i 个子集内的节点数。

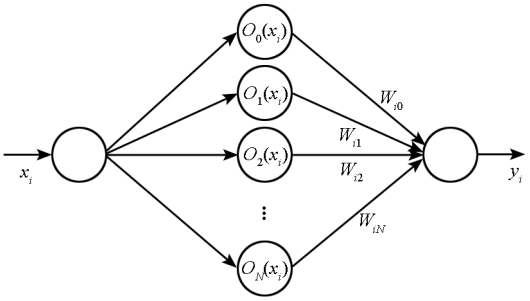

图 4 一元 Chebyshev 神经网络模型

Fig. 4 Monadic Chebyshev neural network model

2.2 路由算法

当一个新的数据流到达源节点 S 时，S 首先检查是否已经构建了 NN。如果已经构建了 NN，则交换机从到达的数据流中提取（Src，Dst，$Type$），并将它们作为 NN 的输入来预测路径，并对其进行路由转发。

如果没有构建 NN，则将数据流相关信息封装到路由请求消息中，发送到控制器。GEO 卫星控制器接收到该请求时，将最近训练完的 NN 下发给所有交换机。交换机在接收到 NN 包后，根据从 NN 包中得到的结构参数构建 NN，然后利用所提取的流信息作为 NN 输入来预测路径，对数据流进行路由。

对于 S，每次切换完成后，S 依据控制器下发的 NN 包构造一个 Chebyshev NN 训练器。具体路由算法由两部分组成。

2.2.1 初始化阶段

当一个新的数据流到达源交换机 S 时，S 从第一个数据包中提取（Src，Dst，$Type$），此时没有训练好的 NN 存储在交换机中，S 向控制器 C 发送包含交换机编号以及数据流（Src，Dst，$Type$）信息的路由请求消息。当 C 收到 S 发送的路由请求消息时，将最近训练完的 NN 分发到所有相关交换机中。交换机在接收到 NN 包后，根据从 NN 包中得到的结构参数构建 NN，然后利用所提取的流信息作为 NN 输入来预测路径，对数据流进行路由。

控制器在构造新的 Chebyshev NN 时，通过 1.3 节生成数据集对神经网络进行训练，在控制器对 NN 的训练过程中，利用 Chebyshev 多项式的正交性，采用文献[17]快速权值确定算法求得。

虽然与直接调用路由算法相比，增加了 NN 训练过程的计算复杂度，但占用存储空间的减小，以及丢包率和网络拥塞的减少使得算法总体性能优越。

2.2.2 正常转发阶段

初始阶段结束后到下次切换前，算法处于正常转发阶段。训练完的 NN 相当于一张“路由表”存储在卫星节点中。新到达的路由请求作为测试样本输入到 NN 以得出最优路径，并根据路径对数据流进行转发。

当出现 S 预测路由失败、一些链路失效或过载、网络拓扑改变、主机请求超时之中任一情况时，S 发送重路由请求消息到 C。重路由请求消息包含节点编号以及数据流（Src，Dst，$Type$）信息。然后，C 根据重路由算法重新计算最优路径，并将其下发给交换机。

同时增加一个负反馈机制，路由失效后，目标节点将反馈一条错误通知信息告知源节点路由失效。收到错误通知信息后，S 将该分类错误的路径添加到重路由请求消息中发送到控制器，作为反例进一步训练 NN。该机制帮助 C 凭借错误通知信息感知网络中链路信息的更新，并通过进一步的训练对 NN 进行相应的调节，为以后的选路提供最新的路由信息。

在进行路径预测时，将数据流的业务类型作为 NN 的一个输入向量，根据用户在带宽、时延、时延抖动、误码率等方面的要求，可以有效地分配网络资源，使路径能够更好地满足不同用户对服务质量（Quality of Service，QoS）的要求。

对于业务类型，根据 IP 网络中的 QoS 分类标准[18]将所有网络应用划分为 6 类，见表 1。其中：U 表示“未指定”，TD、DV、LR 和 ER 分别表示“传输时延”“时延抖动”“丢包率”和“误码率”。

2.3 重路由算法

当数据流到达交换机时，交换机提取（Src，Dst，$Type$），然后使用 Chebyshev NN 来预测数据流的路径。然而，路径预测成功率达不到 100%。因此，设计了重路由算法，见算法 1。当路由预测失败时，交换机将重路由请求消息发送给控制器。

表 1　业务类型划分

Tab. 1　Application classification

QoS 参数	QoS 级别					
	0 级	1 级	2 级	3 级	4 级	5 级
TD	100 ms	400 ms	100 ms	400 ms	1 s	U
DV	50 ms	50 ms	U	U	U	U
LR	1×10^{-3}					U
ER	1×10^{-4}					U

当接收到请求消息时,控制器根据重路由算法重新计算数据流的最佳路由。

重路由算法的主要思想是根据当前的网络状态信息(例如带宽、延迟和错误率)选择最短路径,以满足尽可能多的 QoS 约束条件。首先,源地址 *SrcIP* 和目的地址 *DstIP* 分别映射到相应的源节点 *Src* 和目的节点 *Dst*,并将数据流的业务类型 *Type* 映射到相应的 *QoS* 参数(即带宽、延迟和差错率)。其次,对不能满足带宽要求的所有链路进行过滤。最后,计算满足时延要求的路径集 *DLPathSet*,并从 *DLPathSet* 中选择满足差错率要求的路径 *ERPathSet*。如果 *ERPathSet* 包含多个路径,则选择最短路径作为最优路径。

算法 1　重路由算法

Alg. 1　Rerouting algorithm

输入:*SrcIP*, *DstIP*, *Type*

输出:*Route*

1. (*Src*,*Dst*)←(*SrcIP*,*DstIP*)
2. Qos(*BW*,*DL*,*ER*)←(*Type*)
3. 过滤不满足带宽要求的链路
4. 求得满足时延要求的路径集 *DLPathSet*
5. 确定满足差错率要求的路径 *ERPathSet*
6. **If** *ERPathSet* = ∅ **Then**
7. 　　*Path* = ∅
8. 　**Else** 存在一个或多个路径 **Then**
9. 　　选择最短 *Path* 作为路由路径
10. **End If**

2.4　复杂度分析

在本节中,分析了所提出的路由策略的时间复杂度和空间复杂度。

路由策略涉及的过程主要包括 Chebyshev NN 训练、NN 包下发、NN 创建、数据流信息提取、NN 路径预测。暂不考虑控制器与交换机之间通信(即 NN 包下发)的时间开销。

由上节可知,Chebyshev NN 训练过程的时间复杂度 T_1 和空间复杂度 S_1 均为 $O(1)$。

Chebyshev NN 创建过程的时间复杂度 T_2 为 $O(mn+nk)$,空间复杂度 S_2 为 $O(mn+nk)$。这里,m 和 n 分别是 RBFNN 中的输入节点和隐藏节点的数目,k 是 RBFNN 中输出节点的数目。

数据流信息提取过程中从数据包中提取 m 个字段作为 Chebyshev NN 的输入,其时间复杂度 T_3 和空间复杂度 S_3 均为 $O(m)$。

基于 Chebyshev NN 的路径预测过程的时间复杂度 10^6 和空间复杂度 10^6 均为 $O(mn+nk)$。

基于上述分析,所提出的路由策略的时间复杂度为:

$$\begin{aligned} T &= T_1+T_2+T_3+T_4 \\ &= O(1)+O(mn+nk)+O(m)+ \\ &\quad O(mn+nk)\sim O(mn+nk) \end{aligned} \tag{7}$$

所提出的路由策略的空间复杂度为

$$\begin{aligned} S &= S_1+S_2+S_3+S_4 \\ &= O(1)+O(mn+nk)+O(m)+ \\ &\quad O(mn+nk)\sim O(mn+nk) \end{aligned} \tag{8}$$

3　仿真与性能评估

3.1　仿真设置

基于斯坦福大学研发的轻量级测试平台 Mininet,选取改进的 ONOS 控制器作为实验控制器。为了避免 Mininet 和 ONOS 之间的干扰,将两者以虚拟机的形式部署在两个物理设备上。选取 6 台均有相同配置的实验机器,机器的配置为 Intel Core i5 3.3 GHz、4 GB RAM、2 Gbp 网卡,基于 Ubuntu 14.04 LTS 系统。借助 MATLAB 工具对实验结果进行分析。采用类似文献[16]中的仿真场景,该场景由 3 颗 GEO 卫星控制器、48 颗 LEO 卫星交换机、3 个地面控制器组成。

不同于一般的网络,卫星网络中的节点处在不断的运动之中,节点间时延和链路连通性也在不断发生变化。本文仿真模型中存在轨内链路、轨间链路、层间链路三种星间链路(Inter-Satellite Link, ISL),为了模拟真实卫星网络的动态变换特性,根据文献[18]在 t 时刻,星间链路 $\mathrm{ISL}_{S_j\to S_i}$ 的传播时延定义为:

$$D(S_j\to S_i,t)=r(S_j\to S_i,t)/c \tag{9}$$

ISL 的链路稳定性函数为:

$$\begin{aligned} P(S_j\to S_i,t) &= w_d\cdot\left[\frac{D(S_j\to S_i,t_i)}{D_{\mathrm{ISL,min}}}\right]^{-1}+ \\ &\quad w_t\cdot\left[\frac{T(S_j\to S_i,t)}{T_{\mathrm{ISL},\max}}\right]+w_s\cdot P_{\mathrm{SI}} \end{aligned} \tag{10}$$

采用静态和动态链路设计方案,通过端到端

传输时延和丢包率对链路稳定性进行分析，星间链路的传输速率为 20 Mbit/s，将(w_t, w_d, w_s)取值为(1/3,1/3,1/3)，在考虑链路传输时延和生存时间的基础上同时考虑真实空间干扰因素的影响，以提高传输的连续性和减少网络丢包率。

为了模拟真实的流量状况，采用文献[19]中的数据流模型。地球表面被划分为 $12 \times 24 = 288$ 个区域。每个区域的数值代表上网用户的数量，已被 10^6 整除。对于每 10^6 个上网用户，流量为 1 Mbit/s。

为了方便与基于神经网络的 SDN 智能路由方案（Neural Network-based Intelligent Routing Scheme for SDN, NNIRSS）[14]对比，根据 QoS 分类标准，将所有业务应用分成 6 类，并指定相应的 QoS 参数，如表 2 所示[14]。

表 2 QoS 参数

Tab. 2 QoS parameters

类别	带宽/(Gbit/s)	时延/ms	误码率
0	1.5	100	0.01
1	1.0	400	0.01
2	0.5	300	0.01
3	0.3	400	0.01
4	0.2	1000	0.01
5	0.1	1000	0.01

仿真实验由两部分组成。在第一部分中，利用从上述场景中收集到的历史流量数据，对不同的业务流进行分类，并构造训练集和测试集来训练 NN 以寻找最优参数，NN 训练完成后，将 NN 包存储在控制器中。在第二部分中，将提出的路由策略与基于传统 OpenFlow 的路由方案(Traditional OpenFlow-based SDN Routing scheme, TOSR)以及 NNIRSS[14]进行比较。

3.2 仿真结果分析

通过以下几个指标较全面地评价 IRSCNN 的性能。

1) NN 包占用的存储空间(Storage Space, SP)：在提出的策略中，流表被 NN 包替换，对流表占用的存储空间和 NN 包占用的空间进行对比。

2) 包传递率(Packet Delivery Rate, PDR)：数据包成功到达目的节点的比例。

3) 丢包率(Packet Loss Rate, PLR)：丢失数据包占所有数据包的比例。

4) 包传输时延(Packet Delay, PD)：端到端的数据包传输时延。

每个仿真实验重复 10 次并取其平均值作为度量值。

3.2.1 实验 1：SP

在基于 OpenFlow 的 SDN 中，数据流根据流表进行路由和转发。流表存储在 TCAM 中。然而，在所提出的路由策略中，数据流根据 NN 进行路由和转发，不需要存储在 TCAM 中。为了便于存储空间比较，假设 SDN 中的每个交换机包含 1000 个流表项。在 OpenFlow V1.1.0 中，流表项匹配域的最大长度是 356 bit。因此，流表项占用的 TCAM 存储空间约为 444.57 KB。IRSCNN 与 NNIRSS 之间 SP 的对比结果如图 5 所示。即使考虑将 NN 存储在 TCAM 中，Chebyshev NN 占用的 TCAM 存储空间也远小于 444.57 KB。且与采用 RBFNN 的 NNIRSS 相比，由于空间复杂度降低，进一步减少了存储空间。

因此可以得出，所提出的 IRSCNN 可以显著减少占用的 TCAM 存储空间。

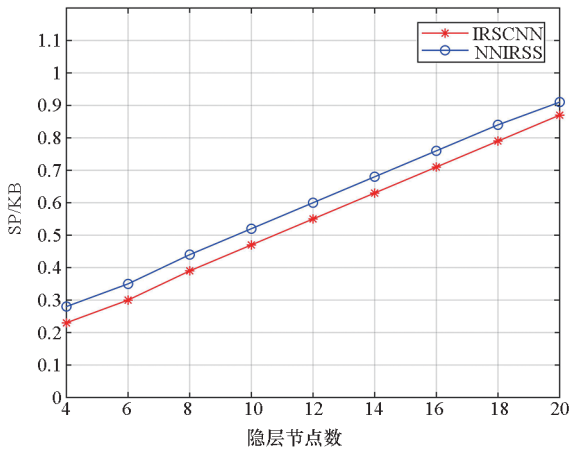

图 5 SP 对比

Fig. 5 Comparison of SP

3.2.2 实验 2：PDR

IRSCNN、NNIRSS 与 TOSR 之间 PDR 的对比结果如图 6 所示。随着流量到达速率的增加，网络中的数据包总数增加，导致网络拥塞。由于交换机的处理能力有限，流中的一些数据包被丢弃，无法成功地路由和转发到目的地。由图 6 可知，无论 IRSCNN、NNIRSS 还是 TOSR，相应的 PDR 都随着流量到达速率的增加而变小。

然而，当流量到达率变大时，IRSCNN 的 PDR 是高于 NNIRSS 和 TOSR 的。TOSR 对数据流采用最短路由策略，在 TOSR 中，流中的数据包沿最短路径路由并转发，导致来自相同数据流的数据包沿着同一路径路由。在 IRSCNN 和 NNIRSS

中,流中的数据包根据其相应的业务类型进行路由和转发,即不同类型的数据包在 IRSCNN 和 NNIRS 中沿不同的路径进行路由和转发,且采用 Chebyshev NN 的 IRSCNN 相比 NNIRSS 求解更迅速,处理速度更快。因此,传送到目的地的数据包的总数大于 NNIRSS 和 TOSR。

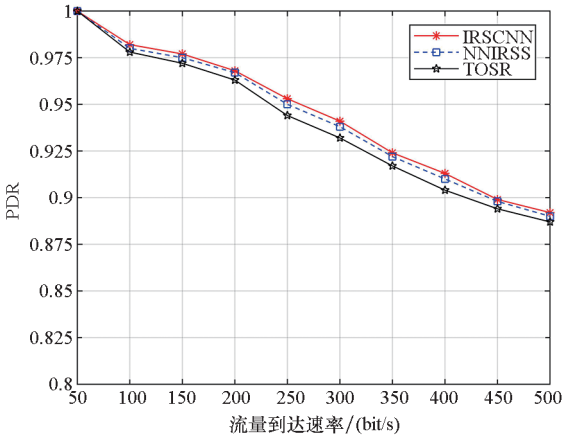

图 6　PDR 对比

Fig. 6　Comparison of PDR

3.2.3　实验 3:PLR

IRSCNN、NNIRSS 与 TOSR 之间的 PLR 比较结果如图 7 所示。随着流量到达速率的增加,三者的 PLR 都变高,而 IRSCNN 的 PLR 比 NNIRSS 和 TOSR 的 PLR 要小。其原因与上述 PDR 比较相似。在 TOSR 中,流中的数据包被路由并转发在最短路径中。而在 IRSCNN 和 NNIRS 中,基于不同的业务类型,数据包沿着不同的路径被转发。随着流量到达速率的增加,TOSR 中由于在相同路径上路由的数据包过多引起网络拥塞而丢弃了大量的数据包,且采用 Chebyshev NN 的 IRSCNN 相比 NNIRSS 求解更迅速,处理速度更快,可以减少拥塞。因此,IRSCNN 中丢弃的数据包的数量要小于 NNIRSS 和 TOSR 中的丢弃的数据包的数量。

3.2.4　实验 4:PD

IRSCNN、NNIRSS 与 TOSR 在不同流量到达速率下的 PD 比较结果如图 8 所示。无论在 IRSCNN、NNIRSS 还是 TOSR 中,随着流量到达速率的增加,流中的数据包总数都变得更大。过多的数据包会导致网络拥塞。网络拥塞越严重,数据包传输到目的节点的时间就越长。

此外,由图 8 可知,TOSR、NNIRSS 的 PD 的长度比 IRSCNN 更长。虽然 TOSR 采用最短路由算法,但是随着流量到达率的增加,网络拥塞更加严重,TOSR 的 PD 变得更长。另一方面,在

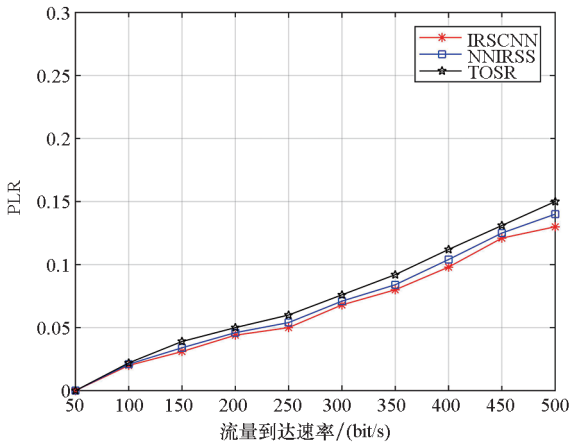

图 7　PLR 对比

Fig. 7　Comparison of PLR

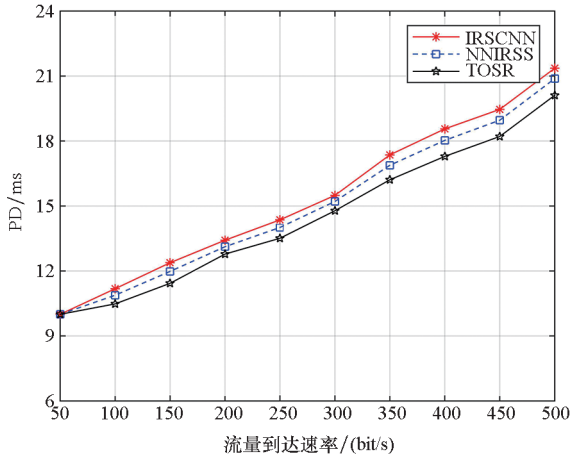

图 8　PD 对比

Fig. 8　Comparison of PD

IRSCNN、NNIRSS 中,数据包的路由和转发过程不涉及复杂的流表查找和匹配过程,与 TOSR 相比,NN 路径预测耗时更短,同时采用 Chebyshev NN 的 IRSCNN 相比 NNIRSS 求解更迅捷,处理速度更快。

综上,IRSCNN 在 SP、PDR、PLR 和 PD 性能方面均优于 TOSR 和 NNIRSS,采用的路由策略具有更高数据包传输成功率,更低的丢包率,更低的数据包传输时延,因此在传输时延和链路稳定性方面也具有更好的性能。

4　结论

针对 OpenFlow 协议中流表扩展导致 TCAM 存储空间不足的问题,提出了基于 NN 的软件定义卫星网络智能路由架构,用 NN 来代替流表,大大减少了占用的 TCAM 存储空间和相应的硬件成本,取消了复杂的流表生成过程,不再需要查找和匹配流表项,提高了路由效率。并在所提架构的基础上提出基于 Chebyshev NN 的智能路由策

略,LEO 卫星根据数据流的业务类型通过训练后的 Chebyshev NN 来预测路径,满足了应用的 QoS 需求。最后,通过仿真实验对所提 IRSCNN 性能进行了评估。仿真结果表明:提出的路由策略能够显著减少占用的 TCAM 空间,提高路由转发效率。

参考文献(References)

[1] Xu S, Wang X W, Huang M. Software-defined next-generation satellite networks: architecture, challenges, and solutions [J]. IEEE Access, 2018, 6(1): 4027 – 4041.

[2] Araniti G, Bisio I, de Sanctis M, et al. Multimedia content delivery for emerging 5G – satellite networks [J]. IEEE Transactions on Broadcasting, 2016, 62(1): 10 – 23.

[3] Vasavada Y, Gopal R, Ravishankar C, et al. Architectures for next generation high throughput satellite systems [J]. International Journal of Satellite Communications and Networking, 2016, 34(4): 523 – 546.

[4] Yousaf F Z, Bredel M, Schaller S, et al. NFV and SDN—key technology enablers for 5G networks [J]. IEEE Journal on Selected Areas in Communications, 2017, 35(11): 2468 – 2478.

[5] Kreutz D, Ramos F M V, Veríssimo P, et al. Software defined networking: a comprehensive survey [J]. Proceedings of the IEEE, 2014, 103(1): 14 – 76.

[6] Bizanis N, Kuipers F A. SDN and virtualization solutions for the internet of things: a survey [J]. IEEE Access, 2016, 4(99): 5591 – 5606.

[7] Du P, Nazari S, Mena J, et al. Multipath TCP in SDN-enabled LEO satellite networks [C]// Proceedings of IEEE Military Communications Conference, 2016: 354 – 359.

[8] 陈晨, 谢珊珊, 张潇潇, 等. 聚合 SDN 控制的新一代空天地一体化网络架构 [J]. 中国电子科学研究院学报, 2015, 10(5): 450 – 454, 459.
CHEN Chen, XIE Shanshan, ZHANG Xiaoxiao, et al. A new generation of space-earth integrated network architecture controlled by aggregated SDN [J]. Journal of China Academy of Electronics and Information Technology, 2015, 10(5): 450 – 454, 459. (in Chinese)

[9] Bao J Z, Zhao B K, Yu W R, et al. OpenSAN: a software defined satellite network architecture [J]. ACM SIGCOMM Computer Communication Review, 2014, 44(4): 347 – 348.

[10] Li T X, Zhou H C, Luo H B, et al. Using SDN and NFV to implement satellite communication networks [C]// International Conference on Networking and Network Applications, 2016: 131 – 134.

[11] Azzouni A, Braham O, Trang N T, et al. Fingerprinting OpenFlow controllers: the first step to attack an SDN control plane [C]//Proceedings of IEEE Global Communications Conference, 2016: 841 – 843.

[12] Karakus M, Durresi A. Quality of service (QoS) in software defined networking (SDN): a survey [J]. Journal of Network and Computer Applications, 2017, 80(12): 200 – 218.

[13] Folkes S R, Lahav O, Maddox S J. An artificial neural network approach to the classification of galaxy spectra [J]. Monthly Notices of the Royal Astronomical Society, 1996, 283(2): 651 – 665.

[14] Zhang C C, Wang X W, Li F L, et al. NNIRSS: neural network-based intelligent routing scheme for SDN [J]. Neural Computing and Applications, 2019, 31(10): 6189 – 6205.

[15] 刘贺语, 孙富春. 基于正交多项式神经网络的卫星网络 QoS 路由算法 [J]. 清华大学学报(自然科学版), 2013, 53(4): 556 – 561.
LIU Heyu, SUN Fuchun. Satellite network QoS routing algorithm based on orthogonal polynomials neural network[J]. Journal of Tsinghua University (Science and Technology), 2013, 53(4): 556 – 561. (in Chinese)

[16] Qi X G, Ma J L, Liu L F. Routing optimization based on topology control in satellite network [J]. Journal on Communications, 2018, 39(2): 11 – 20.

[17] Montijano E, Montijano J I, Sagues C. Chebyshev polynomials in distributed consensus applications [J]. IEEE Transactions on Signal Processing, 2013, 61(3): 693 – 706.

[18] 苑喆, 张军, 柳重堪. 一种基于链路稳定性模型的 LEO/MEO 双层卫星网星间链路设计方法 [J]. 电子与信息学报, 2006, 28(6): 1086 – 1090.
YUAN Zhe, ZHANG Jun, LIU Chongkan. An inter-satellite link design method for LEO/MEO double-layer satellite networks based on link stability model [J]. Journal of Electronics and Information, 2006, 28(6): 1086 – 1090. (in Chinese)

[19] Yang Y, Xu M W, Wang D, et al. Towards energy-efficient routing in satellite networks [J]. IEEE Journal on Selected Areas in Communications, 2016, 34(12): 3869 – 3886.