软件定义网络中的动态负载均衡与节能机制

鲁垚光[1)] 王兴伟[1)] 李福亮[1)] 黄 敏[2)]

[1)](东北大学计算机科学与工程学院 沈阳 110169)

[2)](东北大学信息科学与工程学院 沈阳 110819)

摘 要 软件定义网络(Software Defined Networking,SDN)作为一种新型的网络范式,解决了网络协议臃肿、网络创新困难等问题,但仍面临着诸多挑战如负载均衡与节能.为了应对这一挑战,本文设计了一种动态机制用以解决SDN面临的负载均衡与节能问题.首先,本文提出SDN负载均衡与节能机制的框架.整个框架分为流量监测机制、路由和流调度机制、OpenFlow协议部分和基础设施部分.其中,流量监测机制负责监控网络近似实时的状态,实现数据流级别的流量测量;路由和流调度机制负责在对即将休眠或可能出现拥塞的链路进行流调度;OpenFlow协议负责控制平面和数据平面之间的交互;基础设施部分在转发数据的同时负责数据流路径快速地安装和更新.其次,本文设计了一种基于网络整体流量和数据流速率变化的动态轮询算法,实现了以较小的开销获得流级别的流量测量.然后,本文提出一种基于链路偏好的随机路由算法和两种流调度算法以实现SDN的动态负载均衡与节能.在路由算法中,本文综合负载均衡与节能两个因素,将链路利用率映射为链路的偏好,根据链路的偏好对链路上的流量进行调控,进而使得数据流在路由阶段就能实现负载均衡与节能.这种映射关系随着网络的流量以及链路的利用率动态变化,体现了路由机制良好的动态性.此外,两种流调度算法实现对网络中拥塞链路和空闲链路的流调度,进而达到均衡网络负载和节能的目的.最后,基于Ryu控制器和Mininet平台对SDN中的动态负载均衡与节能机制进行了仿真实验.实验结果表明,本文设计的机制在实现负载均衡的同时最高可节能25%左右,并且在链路平均利用率和能效等多个指标上,明显优于对比机制.

关键词 负载均衡;节能;软件定义网络;链路偏好;流调度

中图法分类号 TP393 **DOI 号** 10.11897/SP.J.1016.2020.01969

Dynamic Load Balancing and Energy Saving Mechanism in Software Defined Networking

LU Yao-Guang[1)] WANG Xing-Wei[1)] LI Fu-Liang[1)] HUANG Min[2)]

[1)](*College of Computer Science and Engineering, Northeastern University, Shenyang* 110169)

[2)](*College of Information Science and Engineering, Northeastern University, Shenyang* 110819)

Abstract As a new type of network paradigm, Software Defined Networking (SDN) decreases the complexity of network and improves the power of network innovation, but still has many challenges especially the load balance and energy saving. Therefore, a dynamic mechanism to solve the problem of load balancing and energy saving in SDN is proposed in this paper. Firstly, the framework of load balancing and energy saving mechanism for SDN is proposed. The framework includes many function modules, including traffic monitoring, routing and flow scheduling, OpenFlow protocol and infrastructure. Among them, traffic monitoring aims at monitoring the near-real-time network status, and achieving traffic measurement at the level of flow; routing and flow scheduling aim at scheduling the flows for the links that may enter into sleep mode or

收稿日期:2018-12-18;在线发布日期:2019-12-16.本课题得到国家重点研发计划项目(2019YFB1802802)、国家自然科学基金项目(61572123,61872073)、教育部-中国移动科研基金项目(MCM20160201)资助.**鲁垚光**,博士研究生,主要研究方向为软件定义网络、移动社交网络路由.E-mail:1225338665@qq.com.**王兴伟**(通信作者),博士,教授,国家杰出青年科学基金入选者,中国计算机学会(CCF)高级会员,主要研究领域为未来互联网、云计算、网络空间安全等.**李福亮**,博士,副教授,主要研究方向为网络管理和测量等.**黄 敏**,博士,教授,主要研究领域为智能算法设计与优化、调度理论与方法等.

encounter congestion; OpenFlow protocol aims at guaranteeing the communication between the control plane and the data plane; infrastructure provides the basis for data transmission, and at the same time aims at quickly installing and updating the data flow paths. Secondly, a dynamic polling algorithm based on the change of the whole network traffic and flow rate is designed for traffic monitoring, which could measure the traffic at the level of data flow with a low cost. Then, this paper proposes a random routing algorithm based on link preference and two flow scheduling algorithms to achieve dynamic load balancing and energy saving for SDN. In the routing algorithm, with the consideration of both load balancing and energy saving, the link utilization is converted to link preference, according to which the traffic on link is controlled, so that load balancing and energy saving can be achieved at the stage of routing. This relation between link utilization and preference can be adjusted dynamically to match network traffic or link utilization, reflecting the dynamic advantage of this mechanism. In addition, two flow scheduling algorithms perform flow scheduling on congested and idle links in the network, thereby achieving the purpose of balancing network load and saving energy. Finally, the simulation environment is constructed based on Ryu controller and Mininet. Then, the mechanism of dynamic load balancing and energy saving is implemented. The experimental results show that the proposed mechanism can achieve energy saving by up to 25% while achieving load balancing, and is superior to the comparison mechanisms in terms of link average utilization and energy efficiency.

Keywords load balancing; energy saving; software defined networking; link preferences; flow scheduling algorithm

《计算机学报》

1 引 言

SDN 目前已然得到了工业界及科研机构的极大重视. 控制和数据平面的分离,是 SDN 最基本的原则,即网络的智能是逻辑集中式的,底层网络设备被抽象出来为上层提供服务,这使得 SDN 具有极佳的灵活性与可扩展性[1]. 然而,SDN 也面临着新的挑战.

网络中的流量飞速增长,如何应对这一挑战,不仅是传统网络面临的问题更是 SDN 面临的问题. 虽然传统网络中负载均衡技术已趋于成熟,如轮询调度算法[2]和多协议标签交换[3]等,但是它们无法直接用于新的网络范式,因此需要以 SDN 的视角来研究负载均衡.

信息通信技术(Information and Communication Technology,ICT)产业也面临着能耗日益增加的问题. 据评估,2012 年 ICT 产业全年消耗了全世界 4.7%的电能[4],并且逐年保持 10%的增长率[5],这使得研究设计节能路由、绿色网络变得十分必要[6]. 在 SDN 中,可以利用其可编程与逻辑集中控制的特点更方便的部署节能机制,从而实现节能,优化网络资源配置.

近年来关于 SDN 负载均衡机制已经开展了一些研究,通常的做法是对网络中的数据流进行调度,从而优化网络的资源配置、提升网络性能. 从网络设备的角度考虑,SDN 中负载均衡机制可以分为针对链路、服务器和 SDN 控制器的负载均衡.

在 SDN 中部署多个控制器能有效提高网络计算的效率,同时也能够有效防止单点失效问题,增加网络的健壮性[7-8]. 文献[9]给出了当前 SDN 控制器负载均衡相关方案 HyperFlow[10]、DIFANE[11]和 BalanceFlow[12]的对比.

在传统网络中服务器的负载均衡主要是通过高性能代理来实现,在 SDN 中我们可以使用控制器来充当代理的角色. 斯坦福大学提出的 Aster*x 模型[13]就是基于 Web 服务器的负载均衡应用.

在网络中,由于负载不均衡经常会导致局部链路拥塞,因此有关链路的负载均衡的研究,通常是寻找缓解局部链路拥塞的解决方案. 在 SDN 中链路的负载均衡分为动态负载均衡与静态负载均衡两类,本文研究的是动态链路负载均衡.

目前,关于 SDN 中节能机制的研究多集中于数据中心的应用场景且已取得了很多成果,如交换机

的节能分配[14]、支持节能的集成控制平面结构[15]、SDN 的节能模型[16]等.在本文中,我们研究的是 SDN 网络级的节能技术,即依据设备的负载以及流量周期性变化的特点等使网络中冗余的设备休眠,从而降低整个网络的能耗.

综上可知,对于 SDN 链路级别的负载均衡和网络设备的节能机制,两者的共同点是都需要对数据流进行调度,不同点在于前者通常将负载最大链路上的流调到负载最小的链路上,而后者相反.这使得目前的研究大多是针对 SDN 中负载均衡与节能问题中的某一方面,或者两方面同时研究但是在流调度上分为两个阶段.本文所提出的机制在最开始的路由计算阶段和后期的数据流调度阶段同时考虑负载均衡和节能这两个目标.此外与前人研究的不同之处还在于本文的拓扑不再局限于数据中心网络,而是可以广泛应用于任何含有冗余设备的网络中.

2　相关工作

动态链路负载均衡是在数据流传输时根据网络链路的负载情况,动态地调整数据流传输路径,所以需要实时获取链路状态.LABERIO[17]是一种基于负载方差的最大流和次大流的动态负载均衡路由算法,通过离线计算保存交换机之间的可能路径,使其在后期寻找替代路径时非常高效.文献[18]提出了一种低成本、负载均衡的路由管理框架,其考虑流表的大小限制并使用 OpenFlow 中的组表来在不同链路之间分配流以平衡网络负载.文献[19]将遗传算法与蚁群优化算法相结合,充分利用两者的优点,提出了一种新的动态负载均衡解决方案.

SDN 网络级节能技术主要包括节能路由和流调度等.

Elastic-Tree[20]是比较经典的数据中心网络节能算法,它专门设计了一个优化模块获得满足 QoS 需求的网络子集,从而将不使用的设备链路以批处理的模式关闭达到节能效果,结果显示能耗降低了 50%.但是该方法需要实时获取网络的流量矩阵,从而降低了网络的性能.

不同于 Elastic-Tree 的批处理模式,文献[21]的节能机制采用渐进式的方法关闭网络设备,在最大限度地降低数据中心网络能耗的同时,不会对其性能产生负面影响.但该机制过于简单,节能效果有限.

Willow[22]是一种用于网络限制流数据中心节能的流调度算法,该方法考虑了所涉及的交换机数量和它们的有效工作时长.但是,仍然存在许多需要解决的关键问题,例如计算复杂性,对网络可靠性的影响,以及设备的关闭对网络性能的影响.

针对数据中心网络负载与所消耗的能量之间不成比例的问题.文献[23]提出了一种在能耗方面提高数据中心效率的新方法,即利用节点间通信流量的相关性和一些拓扑特征实现节能,而平均路径长度仅略微增加.

为了同时实现网络的负载均衡与节能,本文设计了几个重要的功能模块.首先,控制器需要利用其逻辑集中的控制结构得到网络的全局视图,因此,需要对网络进行实时流量监测.其次,由于网络中链路的负载并不是一开始就不均衡,所以本文设计了一个路由模块用于对正常到达的数据流进行路由计算.最后,当流量监测机制发出有关网络链路负载不均衡、链路拥塞等预警信息后,控制器需要能够对相关的数据流实现近似实时的调度.

3　总体设计

3.1　系统框架

本机制设计的总体框架如图 1 所示,整个系统分为四部分:流量监测机制、动态负载均衡与节能路由和流调度机制(简称路由和流调度机制)、OpenFlow

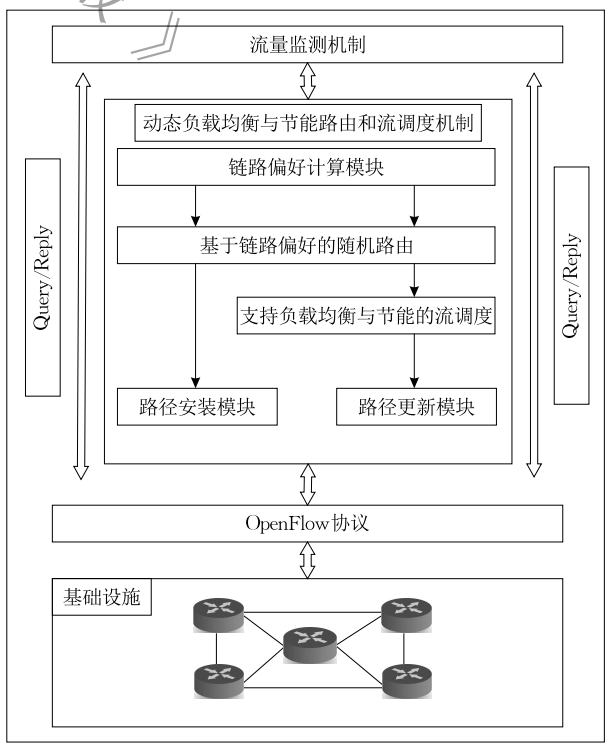


图 1　系统框架图

《计算机学报》

协议部分和基础设施部分.

流量监测机制首先利用 OpenFlow 协议根据设定的自适应轮询算法向基础设施的交换机发出 Query 请求,然后根据接收到的 Reply 消息对每个数据流进行速率监测,同时将与路由和流调度机制进行交互.

路由和流调度机制首先根据流速率计算网络链路利用率并且构建链路利用率矩阵,然后再依据偏好计算方法将利用率矩阵转化为链路的偏好矩阵.偏好矩阵同时兼顾了负载均衡与节能的目标,将作为路由与流调度机制的基础数据.当控制器收到 PacketIn 消息后,路由机制将为数据流提供合理的初始路径,并将新的路径通过 OpenFlow 协议安全通道安装到交换机上.同样当数据流被调度时,控制器将通过流调度模块为数据流计算新的转发路径,并通过 OpenFlow 协议更新到交换机.

OpenFlow 协议作为数据平面与控制平面交互的中介,不但需要将每个数据流路径的添加、更新真实地反映到基础设施的交换机上,而且需要作为通道帮助控制器收集数据流信息.

基础设施部分主要是指交换机等网络设备.基础设施模块需要维护网络的拓扑,对数据流的信息做统计并存储在流表中.此外,它还需要及时响应控制器的各种 OpenFlow 请求消息,将流表中未匹配的数据流 Packet in 到控制器,以及定时向控制器发送 Echo 消息确保与控制器间连接正常.

3.2 网络模型

本文将网络建模为一个带权有向图.为降低流量开销,流量监测机制只需要获取某个时刻的网络快照,因此本机制采用 $G(V,E,t)$ 表示在 t 时刻网络的拓扑.其中,V 为所有网络节点集合,E 为网络中有向边集合.

(1)节点模型

每个节点都有两种状态开启或是休眠,本文将节点建模为一个三元组 $Node(id,stateN,linkset)$.其中,id 用来标识一个节点,$stateN$ 表示网络节点的状态,$linkset$ 表示从此节点出发到相邻节点的边的集合.

对于每个节点当与其相连的每条边都处于休眠状态时,节点将被控制器休眠,其能耗为固定常数 $ENode_{\text{sleep}}$;否则,节点处于开启状态,其能耗为 $ENode_{\text{wake}}$,并且 $ENode_{\text{sleep}}<ENode_{\text{wake}}$.依据文献[24]交换机节点能耗与其开启的端口数和端口利用率相关.本文将交换机节点端口分为两种:特殊端口和正常端口.其中,特殊端口包含主机与接入交换机相连的端口和交换机与控制器节点相连的端口;而正常端口是与其它交换机节点有链路相连的端口.因为特殊端口需要时刻监听网络中的消息,所以这类端口不会被休眠.正常端口则会随着网络状态的变化进行休眠或者开启.同时本文假定特殊端口的能耗为端口能耗的最大值;正常端口能耗包括固定能耗和动态能耗,其中动态能耗部分与端口利用率成正比.因此,休眠状态的节点能耗 $ENode_{\text{sleep}}$ 和开启状态下节点能耗 $ENode_{\text{wake}}$ 计算分别如式(1)和(2)所示.

$$ENode_{\text{sleep}}=c_1 \tag{1}$$

$$ENode_{\text{wake}}=c_2+n_1\times EPort_{\text{special}}+\sum_{i=1}^{n_2}EPort_{\text{normal}} \tag{2}$$

$$EPort_{\text{special}}=c_3 \tag{3}$$

$$EPort_{\text{normal}}=c_4+r\times(c_3-c_4) \tag{4}$$

其中,c_1 表示休眠节点的固定能耗,$c_1>0$;c_2 表示开启状态下节点的基础能耗,$c_2>0$.n_1 和 n_2 分别为特殊端口与正常端口的数目,$EPort_{\text{special}}$ 为特殊端口的能耗,$EPort_{\text{normal}}$ 为正常端口的能耗.

$EPort_{\text{special}}$ 和 $EPort_{\text{normal}}$ 计算分别如式(3)和(4)所示.其中,c_4 为休眠状态端口固定能耗,$c_4>0$;c_3 为端口在开启状态下最大能耗,$0<c_4<c_3$;r 为端口利用率,$0\le r\le 1$.

本文将节点正常端口能耗归结为有向边的能耗,从而开启状态下节点能耗 $ENode_{\text{wake}}$ 计算如式(5)所示.

$$ENode_{\text{wake}}=c_2+n_1\times c_3 \tag{5}$$

(2)链路模型

由于链路是全双工的工作方式,因此本文将一条真实的网络链路抽象为两条反向的逻辑链路,每条逻辑链路对应一条有向边,如图 2 所示.每一条有向边用 $edge(id,s,t,port_s,port_t,state,bw,rw,capacity,ratio,favor,life_{\text{low}},life_{\text{high}},flowset)$ 表示.其中 id 为边的标识,s 代表边的出节点,t 代表边的入节点,$port_s$ 代表出端口,$port_t$ 代表入端口,$stateF$ 为边的状态,表示边的休眠或开启.bw 为边的占用带宽,rw 为边的剩余带宽,$capacity$ 为边的带宽容量,$ratio$ 为边的带宽利用率,$favor$ 为边的偏好权重.$life_{\text{low}}$ 和 $life_{\text{high}}$ 是为了防止边被频繁休眠或唤醒而设置的缓冲生命值,$life_{\text{low}}$ 为边带宽利用率低于阈值的生命值,$life_{\text{high}}$ 为边带宽利用率高于阈值的生命值,其作用与流表项中的空闲超时相似,当一条有向边利用率低于某阈值时,其 $life_{\text{low}}$ 逐

《计算机学报》

渐递减；当其利用率一旦超过阈值，则其 $life_{low}$ 将恢复到初始值，当且仅当有向边的 $life_{low}$ 小于 0 时，有向边被休眠. $flowset$ 为经过这条边的所有数据流 id 集合.

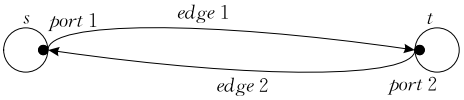


图 2　链路模型

每条有向边的能耗与其出端口相关联，如图 2 所示，$edge1$ 的能耗为 $port1$ 的能耗，$edge2$ 的能耗为 $port2$ 的能耗. 假设休眠的边的能耗为 $EEdge_{sleep}$，开启状态的边的能耗为 $EEdge_{wake}$，则 $EEdge_{sleep}$ 和 $EEdge_{wake}$ 的计算分别如式(6)和(7)所示.

$$EEdge_{sleep}=c_4,c_4>0 \tag{6}$$

$$EEdge_{wake}=c_4+r\times(c_3-c_4) \tag{7}$$

(3) 网络拓扑模型

由于节点和有向边存在休眠和开启两个状态，因此本文对网络定义了两种拓扑模型：活动拓扑和全局拓扑. 其中，活动拓扑为网络中所有处于开启状态的节点和有向边构成的网络拓扑，全局拓扑为网络中所有的节点和有向边构成的网络拓扑.

在本文的能耗模型中，网络能耗不包含控制器能耗，只考虑网络中所有交换机节点能耗，网络能耗计算如式(8)所示.

$$ENetwork=\sum_{i=1}^{n_3}ENode_i+\sum_{j=1}^{n_4}EEdge_j \tag{8}$$

其中，n_3 和 n_4 为网络中节点数目和有向边的数目，$ENode_i$ 和 $EEdge_j$ 分别指第 i 个节点的能耗和第 j 条有向边的能耗.

(4) 数据流模型

为了方便控制器对网络中数据流的监测、管理以及调度，需要在控制器中保存数据流的对象，包含对数据流详细信息的描述. 针对数据流对象本文用如下元组结构描述 $Flow(cookie,src,dst,rate,fbytes,time,interval,endswitch,match,path,edgeset)$. 其中，$cookie$ 用以标识一个数据流，src 和 dst 分别代表数据流的源地址和目的地址，$rate$ 为数据流的速率，$fbytes$ 为数据流从开始到现在总共传送的字节数，$time$ 为当前时间戳，$interval$ 为轮询时间间隔，$endswitch$ 为数据流经过的最后一个交换机节点，$match$ 为数据流在流表中的匹配项，$path$ 为数据流经过的一系列交换机节点的有序序列，$edgeset$ 为数据流经过的有向边的集合.

4　流量监测机制

4.1　动态时间间隔与流量矩阵

(1) 动态时间间隔

本机制采用“加法增大乘法减小”的策略调整流的轮询时间间隔. 最小的轮询时间间隔 $base$ 由当前的网络流量决定，网络流量越大，其 $base$ 值越大，且第一次轮询时间间隔为 $base$. 在后续的时间间隔更新策略中，后续的时间间隔 $interval'$ 不仅与数据流速率变化 Δr 相关还与上一次的时间间隔 $interval$ 相关. 根据数据流速率的变化 Δr，其可分成三种情况. 当 $0\leq\Delta r\leq\alpha_1$ 时，轮询间隔线性增大；当 $\alpha_1<\Delta r\leq\alpha_2$ 时，轮询间隔保持不变；当 $\Delta r>\alpha_2$ 时，轮询间隔乘法减小. 动态时间间隔的计算如式(9)所示，其中 α_1 和 α_2 为常数，且满足 $0<\alpha_1<\alpha_2<1$.

$$interval'=\begin{cases}base, & \Delta r\to\infty\\ \min\{interval+1,maxinterval\}, & 0\leq\Delta r\leq\alpha_1\\ interval, & \alpha_1<\Delta r\leq\alpha_2\\ \max\{\lfloor interval/\Delta r\rfloor,base\}, & \Delta r>\alpha_2\end{cases} \tag{9}$$

其中，$maxinterval$ 为轮询时间间隔的最大值，$\lfloor interval/\Delta r\rfloor$ 表示向下取整.

(2) 流量矩阵

式(10)和(11)分别为网络的剩余带宽矩阵 $\boldsymbol{rwTM}$ 和带宽利用率矩阵 $\boldsymbol{ratioTM}$.

$$\boldsymbol{rwTM}=\begin{bmatrix}rw_{11} & \cdots & rw_{1j} & \cdots & rw_{1n}\\ \vdots & \ddots & & & \vdots\\ rw_{i1} & & rw_{ij} & & rw_{in}\\ \vdots & & & \ddots & \vdots\\ rw_{n1} & \cdots & rw_{nj} & \cdots & rw_{nn}\end{bmatrix} \tag{10}$$

$$\boldsymbol{ratioTM}=\begin{bmatrix}r_{11} & \cdots & r_{1j} & \cdots & r_{1n}\\ \vdots & \ddots & & & \vdots\\ r_{i1} & & r_{ij} & & r_{in}\\ \vdots & & & \ddots & \vdots\\ r_{n1} & \cdots & r_{nj} & \cdots & r_{nn}\end{bmatrix} \tag{11}$$

其中，n 为网络中所有节点的数量；rw_{ij} 表示有向边 $edge_{ij}$ 的剩余带宽；r_{ij} 表示有向边 $edge_{ij}$ 上所用带宽与总带宽的比值；当节点 i 与节点 j 间没有边连接时，$rw_{ij}=0$，$r_{ij}=1$.

4.2　算法描述

监测机制需要同时开启两个处理线程，一个用于接受处理 OpenFlow 消息如算法 1 所示，另一个

用于发送轮询消息如算法 2 所示. 只有两个算法协作,才能完成对网络的监测.

算法 1 将 OpenFlow 消息分类进行处理. 如果收到的消息是 PacketIn 消息,则监测机制对数据流进行一系列登记,初始化其监测时间点 $checkpoint_f$,并将其加入监测队列 *checkset*;如果收到的是 StatsResponse 消息,监测机制根据消息中的数据计算数据流的速率,并对一系列与该数据流相关的有向边进行更新,同时更新网络状态矩阵,最后添加数据流新的检查点到监测队列中;如果是 FlowRemove 消息,则更新相关的有向边,更新网络状态矩阵,同时将其监测点从 *checkset* 中移除,最后将整个数据流从 *flowset* 中移除. 由于算法 1 只是对不同类型的 OpenFlow 消息进行处理,所以其时间复杂度为 $O(1)$. 算法 1 的伪码如下:

算法 1. OpenFlow 消息处理.

输入:OpenFlow 消息

BEGIN

1. $checkset \leftarrow \varnothing$ and $flowset \leftarrow \varnothing$;
2. Initialize the matrix ***rwTM***, ***ratioTM*** to be equal the latest value;
3. FOR each message that the controller receives msg_i, $i=1,2,3,\cdots$, DO
4. IF msg_i is PacketIn message, THEN
5. IF the new flow $f \notin flowset$, $flowset \leftarrow \{f\} \cup flowset$ THEN
6. Do other thing, get current time t, set $checktime_f \leftarrow t + interval'$;
7. $checkpoint_f \leftarrow (checktime_f, f)$;
8. $checkset \leftarrow \{checkpoint_f\} \cup checkset$;
9. END IF
10. ELSE IF msg_i is StatsResponse message, THEN
11. Update the flow rate of f;
12. Update the flow's $interval$, $checkpoint_f$, $checkset$;
13. Update rw, bw, r of edges that are in the flow's edgeset;
14. Update ***rwTM***, ***ratioTM***;
15. ELSE IF msg_i is FlowRemove message, THEN
16. Update rw, bw, r of edges that are in the flow's edgeset;
17. Update ***rwTM***, ***ratioTM***;
18. Remove the f from the $flowset$;
19. Remove the $checkpoint_f$ from the $checkset$;
20. END IF
21. END FOR

END

算法 2 被称为数据流轮询,首先初始化全局时钟 *clocktime*,每隔一秒其值加一,然后比对 *checkset* 中的监测点,如果监测点的时间与当前时间相同,则对该检测点的数据流经过的最后一个交换机发送 StatsRequest 消息,并将原有的监测点从 *checkset* 中删除. 设 *checkset* 中有 n 个监测点,算法 2 需要遍历所有监测点,时间复杂度为 $O(n)$. 算法 2 的伪码如下:

算法 2. 数据流轮询.

输入:*checkset*

BEGIN

1. $clocktime \leftarrow 0$;
2. WHILE *checkset* is not $\varnothing$, DO
3. FOR each check point $checkpoint_i$, $i=1,2,3,\cdots$, DO
4. IF $checktime_i = clocktime$, THEN
5. Send the StatsRequest message to the flow f_i last switch;
6. Remove $checkpoint_i$ from *checkset*;
7. END IF
8. END FOR
9. $clocktime \leftarrow clocktime + 1$;
10. END WHILE

END

5 基于链路偏好的随机路由算法

基于链路偏好的随机路由算法(Preference-based Random Routing algorithm, PbRR)的特点是随机性与动态性. 其随机性体现在选择下一跳的过程中. 采用了 Softmax 的思想,对于当前网络状态有利的下一跳并不是一定选择,而是赋予其较大的被选概率,同样对于当前网络状态不利的下一跳赋予其较小的被选概率. 其动态性体现在偏好随有向边带宽利用率和网络流量动态变化,如图 3 所示.

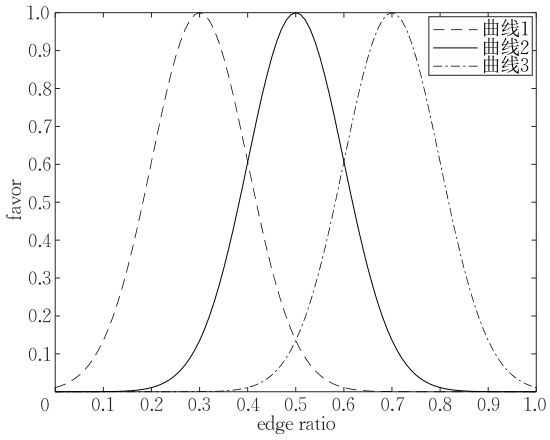


图 3 链路利用率与偏好关系图

《计算机学报》

本文将有向边的利用率通过函数映射为偏好，转化曲线是一条对称的凸曲线，其对称轴就是活跃链路的平均利用率，当网络整体流量偏低时，利用率与偏好的映射曲线如曲线 1 所示，随着网络整体流量的上升，整个曲线随之向右移动，如曲线 2 或曲线 3 所示. 以曲线 2 为例说明它为什么能同时实现负载均衡与节能的效果：当活跃链路的利用率在对称轴左侧时，若利用率相对较低，其偏好映射也将非常低，进而导致这条链路被选中的概率很小，对于新到达网络的数据流起到抑制的作用，经过一段时间，它的利用率甚至可能会下降到 0，此时这条链路就进入到休眠状态；若利用率相对较高，那么该链路将很容易被数据流选中，使得它的利用率上升，偏好也随之上升，这将对它起到一个促进作用，使链路的利用率向均值靠拢. 当活跃链路的利用率在对称轴右侧时，假定其被大象流选中，利用率迅速上升，偏好则会随之下降，这将抑制该链路被新的数据流选中，从而可以有效缓解链路的拥塞.

偏好的计算如式(12)所示.

$$favor_{ij}=\exp\left(-\frac{(r_{ij}-\bar{r})^2}{2\sigma^2}\right) \tag{12}$$

其中，σ 为大于 0 的实数；$\bar{r}$ 为有数据流经过的有向边的平均利用率.

假定节点 u 到某一指定节点有 m 个邻接点 $\{v_1,v_2,\cdots,v_m\}$ 可供选择，且到该指定节点的路径不构成环路，则 $\{v_1,v_2,\cdots,v_m\}$ 称为候选节点. 算法 3 采取 Softmax 函数将与候选节点相邻的边的偏好转化为概率. 在已知各候选节点被选择的概率的情况下，再以轮盘赌的方式进行选择. 具体的做法是先随机一个区间 $[0,1)$ 内的实数 p 作为选择的概率，然后累加候选节点的概率值直到大于 p，从而对节点做出选择. 设从源节点到目的节点经过 n 跳，则算法 3 的时间复杂度为 $O(mn)$. 假定 $\{v_1,v_2,\cdots,v_m\}$ 的偏好为 $\{favor_i\mid i=1,2,\cdots,m\}$，其对应的概率为 $\{p_i\mid i=1,2,\cdots,m\}$，则 p_i 的计算如式(13)所示.

$$p_i=\frac{\exp(favor_i)}{\sum_{j=1}^{m}\exp(favor_j)} \tag{13}$$

算法 3 的伪码如下：

算法 3.　PbRR.

输入：***rwTM***，***favorTM***，源节点和目的节点 s,t

输出：源节点到目的节点的一条路径 *path*

```
BEGIN
1.  Initialize the path as null;
2.  u←s;
3.  Add u to the path;
4.  WHILE u≠t, DO
5.    selectset←∅, sump←∅;
6.    FOR each adjacent node v_i, i=1,2,…,m of u, DO
7.      IF exist a acyclic subpath_i that the node v_i can reach the node t, THEN
8.        selectset←{v_i}∪selectset;
9.        sump←{p_i}∪sump;
10.     END IF
11.   END FOR
12.   IF selectset is empty, THEN
13.     Break;
14.   END IF
15.   Initialize a random p, p∈[0,1), sum←0;
16.   FOR each node sv_j in selectset, DO
17.     sum←sum+p_j;
18.     IF sum≥p, THEN
19.       Add u to path;
20.       u←sv_j;
21.       Break;
22.     END IF
23.   END FOR
24. END WHILE
25. IF the node u≠t, THEN
26.   Set path as null;
27. END IF
28. RETURN path;
END
```

为了降低能耗，PbRR 算法会优先基于活动拓扑为数据流计算路径. 如果计算出的路径不能满足约束条件，则将会基于全局拓扑为数据流计算路径.

6　流调度算法

本文设计了两种数据流调度算法：负载均衡与节能的流调度(Load balancing and Energy saving flow scheduling，LoadbE)和迭代式负载均衡与节能的流调度(Load balancing and Eenergy saving flow scheduling with iteration，LoadbE-it)，简称为非迭代式流调度和迭代式流调度.

6.1　触发策略

触发流调度的情况分为 3 种：(1) 对拥塞链路上的数据流进行调度，这种情况最为紧急，所以其调度优先级最高；(2) 对可以节能的链路上的数据流进行调度以实现节能的效果，其优先级最低；(3) 当网络中链路的负载不均衡时，对数据流进行调度使

《计算机学报》

网络尽快恢复到链路负载均衡的状态.

在本文中,链路的带宽利用率用于表示其负载;链路带宽利用率的标准差 std_r 用于衡量负载是否均衡. std_r 的计算如式(14)所示.

$$std_r=\sqrt{\frac{\sum_{k=1}^{n}(r_{ij}-\bar{r})^2}{n}} \tag{14}$$

其中,设阈值为 δ,当 $std_r\geq\delta$ 时触发负载均衡的流调度.

6.2 LoadbE 算法

LoadbE 算法每次运行只调度一次触发的数据流. 具体地,首先,分别在活动拓扑和全局拓扑上调用 PbRR 算法 n 次,然后采用基于局部的网络状态信息对 $pathset$ 中的路径进行评价,选出评价最高的路径作为调度路径. LoadbE 算法的时间复杂度为 $O(mn^2)$,伪码如下:

算法 4. LoadbE.

输入:数据流标识 $cookie$,PbRR 执行次数 n,***ratioTM***,***bwTM***,***favorTM***

BEGIN

1. $pathset\leftarrow\varnothing$, $i\leftarrow0$;
2. WHILE $i<n$, DO
3. Using PbRR on the active topology to get a new $path_i$ for $cookie$, $i\leftarrow i+1$;
4. IF $path_i\in pathset$, THEN
5. Increase the counter of $path_i$;
6. ELSE
7. Add $path_i$ to $pathset$, and set the counter of $path_i$ is 1;
8. END IF
9. END WHILE
10. $j\leftarrow0$;
11. WHILE $j<n$, DO
12. Using PbRR on the global topology to get a new $path_j$ for $cookie$, $j\leftarrow j+1$;
13. IF $path_j\in pathset$, THEN
14. Increase the counter of $path_j$;
15. ELSE
16. Add $path_j$ to $pathset$, and set the counter of $path_j$ is 1;
17. END IF
18. END WHILE
19. Select the best path of $pathset$ as $path$;
20. Change the path of flow $cookie$ with $path$;
21. Update ***ratioTM***,***bwTM***,***favorTM***

END

假设数据流 f 流调度后产生的路径集合为 $pathset_{\text{local}}=\{path_1,path_2,\cdots,path_m\}$,则为每条路径 $path_i$ 的评分方式如式(15)所示.

$$score_i=\frac{k_i}{\sum_{i=1}^{m}k_i}(1-(rmax_i-rmin_i)+\varepsilon_1)+\left(1-\frac{k_i}{\sum_{i=1}^{m}k_i}\right)\frac{maxEpath-Epath_i}{maxEpath-minEpath+\varepsilon_1} \tag{15}$$

其中,k_i 为 $path_i$ 出现的次数;$rmax_i$,$rmin_i$ 为 $path_i$ 为经过的链路上利用率的最大值和最小值;ε_1 为一个极小的正整数;$Epath_i$ 为 $path_i$ 的能耗;$maxEpath$ 和 $minEpath$ 分别为 m 个路径能耗中的最大值和最小值.

6.3 LoadbE-it 算法

LoadbE-it 算法的处理流程为前一半的迭代在活动拓扑上进行,后一半的迭代在全局拓扑上进行. 每次迭代的处理首先基于启发式信息 $info_i$ 选择数据流,然后为数据流计算多条路径,并对这些路径进行基于全局网络的评分,从而选出最佳调度路径. 每次选出最优路径后都会计算调度后的网络状态,并与当前状态进行比较,只有比当前状态更好时才执行调度. 启发式信息 $info_i$ 计算如式(16)所示.

$$info_i=\frac{(rf_i-\bar{r})^2}{hop_i} \tag{16}$$

其中,rf_i 为 f_i 经过的有向边的平均带宽利用率;hop_i 为 f_i 的路径跳数. $info_i$ 如果很大,表示其对 std_r 的影响就很大. LoadbE-it 算法由于需要调度若干个数据流,并且使用迪杰斯特拉算法为每个数据流计算多条路径,所以其时间复杂度为 $O(n^4)$,伪码如下:

算法 5. LoadbE-it.

输入:迭代次数 itk,拓扑类型 $topo$,***ratioTM***,***bwTM***,***favorTM***

BEGIN

1. $i\leftarrow0$, $topo\leftarrow1$
2. WHILE $i<itk$ and trigger conditions are satisfied, DO
3. $edgeset\leftarrow\varnothing$, $pathset\leftarrow\varnothing$;
4. IF $i\geq\lfloor itk/2\rfloor$, THEN
5. $topo\leftarrow0$;
6. END IF
7. Select the best flow as f using (16);
8. Add the shortest path to $pathset$;
9. Add each edge from the shortest path of f to $edgeset$ with the $topo$;
10. FOR each $edge_j$ of $edgeset$, DO

《计算机学报》

```
11.    Get the shortest path path_j of f without edge_j;
12.    Add path_j to pathset;
13.  END FOR
14.  Add the path of PbRR to pathset;
15.  Select the best path of pathset as path;
16.  IF the schedule for f is better than current, THEN
17.    Change the path of f with path;
18.    Update ratioTM, bwTM, favorTM;
19.  END IF
20.  i←i+1;
21. END WHILE
END
```

式(17)为 LoadbE-it 算法中所采用的评分函数.其中,$rmax_i$为数据流 f 按路径 $path_i$调度后网络中链路利用率最大值;std_r^i代表数据流 f 按路径 $path_i$调度后网络活跃链路利用率的标准差;$Enet_i$为数据流 f 按路径 $path_i$调度后网络的能耗;$Enet_{\text{total}}$为网络以最大功率工作的总能耗.

$$score_i = rmax_i \times std_r^i + (1-rmax_i)\frac{Enet_i}{Enet_{\text{total}}} \tag{17}$$

7　性能评价

7.1　评价指标

首先,设计了以链路利用率的标准差作为衡量网络在数据流调度时的负载均衡度.其值越小代表网络负载越均衡;反之,其值越大则反映网络的链路负载差别很大.t 时刻链路利用率的标准差计算如式(18)所示.

$$std_r^t = \sqrt{\frac{\sum_{i=1}^{m}(r_i^t - \bar{r}_t)^2}{m}} \tag{18}$$

其中,r_i^t是 t 时刻活跃链路 i 的利用率,$\bar{r}_t$是 t 时刻活跃链路的平均利用率,m 为活跃链路的数目.

其次,采用全网的能耗功率作为节能的指标.网络的状态会随着流量的变化而变化,因此只要能够通过计算画出网络的功率曲线,那么曲线下的面积则代表这段时间内网络的能耗.t 时刻网络能耗计算方式如式(19)所示.

$$ENetwork_t = \sum_{i=1}^{m} ENode_i^t + \sum_{j=1}^{n} EEdge_i^t \tag{19}$$

最后,本文从活跃链路利用率和能耗这两个角度考察本机制在网络性能方面的表现.

式(20)为在 t 时刻,活跃链路的平均利用率 $mean_ratio_t$的计算方式.

$$mean\_ratio_t = \frac{\sum_{i=1}^{m} r_i^t}{m} \tag{20}$$

其中,m 为活跃链路的数目;r_i^t是 t 时刻活跃链路 i 的利用率.

网络的能效为吞吐量与网络能耗的比值,式(21)为网络在 t 时刻的吞吐量 $throughput_t$的计算方式.

$$throughput_t = \sum_{i=1}^{n} rate_i^t \tag{21}$$

其中,n 为 t 时刻数据流数目;$rate_i^t$为 t 时刻数据流 i 的速率.

所以,网络在 t 时刻,能效 eta_t 的计算方式如式(22)所示.

$$eta_t = \frac{throughput_t}{ENetwork_t} \tag{22}$$

7.2　对比算法

经典的 ECMP 算法和 LABERIO 算法中是不包含节能策略的,本文为了实现公平的对比,在两种算法原来的基础之上引入了节能模块.

改进的 ECMP(IECMP)算法:数据流到达控制器时,控制器为其计算多条相同代价的路径,控制器根据第一个数据包的头部信息进行 Hash,从而确定一条路径路由数据流.另外,IECMP 算法将节能模块加入到经典的 ECMP 算法上.当链路利用率为 0 时,链路处于休眠状态;当节点的所有相关链路都处于休眠状态时,节点也将休眠.

改进的 LABERIO(ILABERIO)算法:LABERIO 算法会预先获取任意两个交换机间的多条路径,同时监测每个时刻网络的状态.当数据流到达时,选择剩余带宽最大的一条路径作为初始路径.在链路负载不均衡的情况下,LABERIO 算法使用替代链路代替负载最大的链路.ILABERIO 则是在 LABERIO 基础上加入了与 IECMP 相同的节能模块.

7.3　仿真实现

本文 SDN 中动态负载均衡与节能机制的仿真实现硬件环境为 Intel(R) Core(TM) i5-4590 CPU@3.30 GHz,4.00 GB 内存,操作系统为 Ubuntu 16.04 LTS.整个机制基于 Python 2.7 实现,以 Sublime text 2 为主要开发工具,在 Ubuntu 16.04 平台下调试运行.

在 SDN 平台搭建过程中,对控制平面的仿真,本文采用的是开源控制器 Ryu,对数据平面的仿真,本文采用的是开源的仿真工具 Mininet.

《计算机学报》

本文采用文献[25]和文献[17]中的拓扑进行实验测试,其中,文献[25]中的拓扑较为简单,命名为Topo1,如图4所示;文献[17]中的拓扑为数据中心拓扑,命名为Topo2,如图5所示.

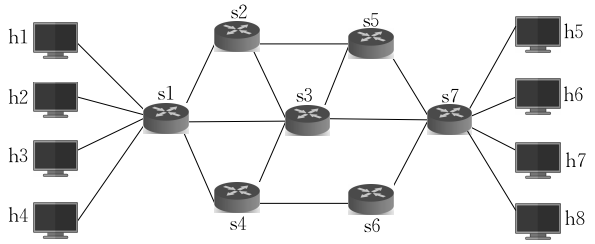


图4 实验拓扑 Topo1

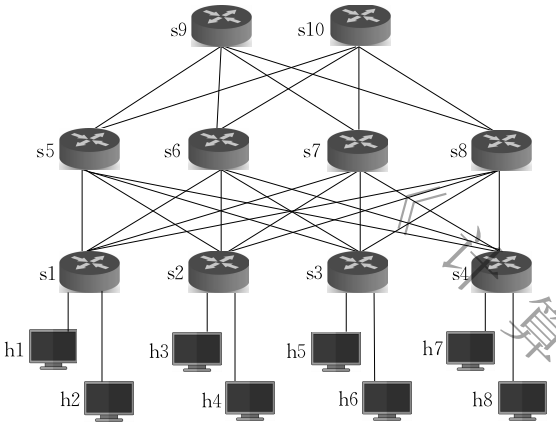


图5 实验拓扑 Topo2

基于多次实验,本文对SDN中动态负载均衡与节能机制中的参数设置如表1所示.

表1 主要参数描述

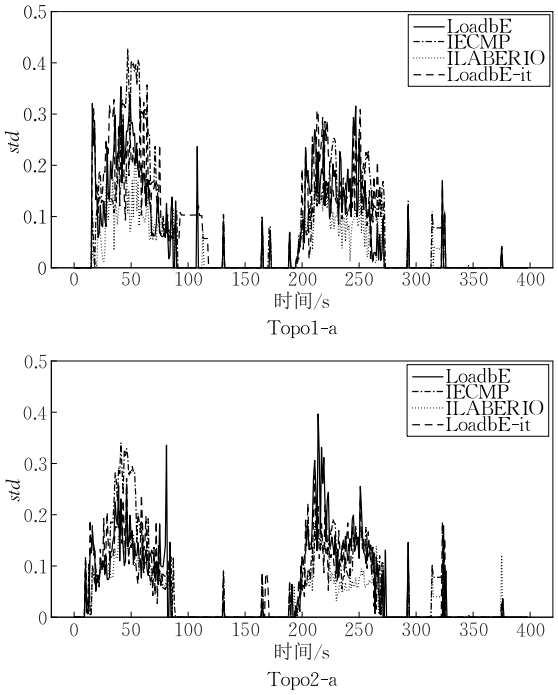
参数设置	参数描述
$c_1=240$	休眠节点固定能耗
$c_2=310$	开启状态节点固定能耗
$c_3=4$	端口负载最大时能耗
$c_4=2$	端口空载时能耗
$\alpha_1=0.03$	流速率变化阈值
$\alpha_2=0.1$	流速率变化阈值
$base\in\{1,2,3\}$	流速率轮询时间间隔基数
$\sigma=0.01$	偏好映射曲线参数
$life_{low}=4$	低于链路利用率的缓冲生命值
$life_{high}=2$	高于链路利用率的缓冲生命值
$thd_{low}=0.05$	链路利用率休眠阈值
$thd_{high}=0.85$	链路利用率过高预警阈值

7.4 性能测试与分析

本文将LoadbE算法和LoadbE-it算法与IECMP算法和ILABERIO算法做了对比,并从负载均衡、节能和网络性能三个角度对实验结果进行了分析.由于四种算法的仿真实验结果曲线波动较大,不容易直接进行对比,因此本文对实验结果采用累积分布函数(Cumulative Distribution Function,CDF)进行对比.累积分布函数曲线是反映数据分布的曲线,即用来表示概率的分布,其中"Topo1"或者"Topo2"代表在不同拓扑下的实验结果,"-a"代表原始数据随时间变化曲线,"-b"表示累积分布函数曲线.

(1)负载均衡度量

网络中活跃链路利用率的标准差随时间变化曲线和累积分布函数曲线如图6所示.



Topo1-a Topo1-b

Topo2-a Topo2-b

图6 链路利用率标准差

不同算法链路利用率方差的累积分布如 Topo1-b 和 Topo2-b 所示，曲线越高代表 *std* 在该值以内概率分布越大. 两种拓扑上总体趋势相同，表现最佳的是 ILABERIO 机制，后三种机制总体表现相差不大. ILABERIO 机制之所以表现最佳是因为每次进行路径调度的时候 ILABERIO 使用子路径取代高负载的链路，其影响的链路范围较少. 然而，其它机制会重新计算一条路径替代原路径，影响的链路范围广，进而导致网络链路的利用率波动较为剧烈，产生较大的标准差. 由此可见，LoadbE 和 LoadbE-it 在全路径替换的流调度算法中是有竞争力的.

(2) 网络能耗度量

网络的功率随时间变化曲线和累积分布函数曲线如图 7 所示. 网络的能耗即为功率曲线下的面积，整体上看，所设计机制的节能效率最高可达 25% 左右. 从曲线的波动情况看，LoadbE-it 最剧烈，说明其对网络流量的变化比较敏感，这是因为 LoadbE-it 每次调度多个数据流，对网络整体影响较大.

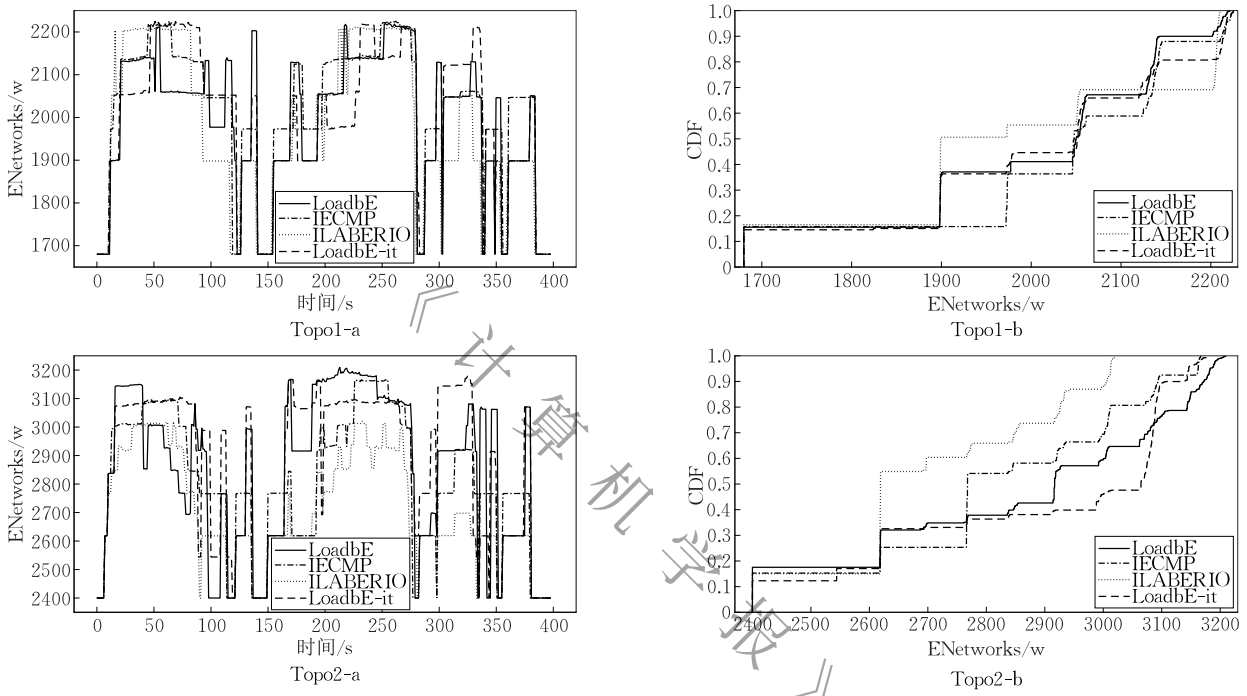


图 7　网络功率

在 Topo1 和 Topo2 仿真结果中网络功率的累积分布曲线如 Topo1-b 和 Topo2-b 所示. 在 Topo1-b 中，LoadbE 和 LoadbE-it 机制的曲线要高于 IECMP 机制的曲线，代表相同功率下，其概率分布较大，能耗较小. 在 Topo2-b 中，IECMP 机制的概率分布要优于本文提出的 LoadbE 和 LoadbE-it 机制，这是因为 IECMP 机制会优先选择较短路径，而在 Topo2 这样的全连接拓扑中，2 跳的路径有 24 条，占 4 跳以内路径总数的 1/7，这些路径已经能够满足 IECMP 机制的调度需求，因此很少需要开启其它节点.

(3) 网络性能度量

在网络性能指标上，本文从平均链路利用率和能效这两方面进行度量.

① 平均链路利用率

图 8 为网络中活跃链路的平均带宽利用率随时间变化的曲线及其累积分布曲线. Topo1-a 和 Topo2-a 显示，网络中活跃链路的平均利用率随着网络整体流量的变化而波动，当处于流量高峰时期，每条活跃链路利用率随之升高，当网络流量较为稀疏时，即便四种算法对流量有一定的聚合效果，链路利用率也会随着下降.

观察图 Topo1-b 和 Topo2-b 中的累积分布曲线，LoadbE 和 LoadbE-it 的曲线最低，可以得出本文提出的算法在活跃链路平均利用率上要优于对比算法，即在对数据流的聚合效果上表现更佳.

② 能效

图 9 为网络的能效随时间变化的曲线及其累积分布曲线. 从图 Topo1-a 和图 Topo2-a 上看，LoadbE-it 的能效波动较大，这是由于运行 LoadbE-it 会调度不止一个数据流，致使数据流的调动过于频繁. 从 Topo1-b 的累积分布曲线中可以看出，LoadbE 和

《计算机学报》

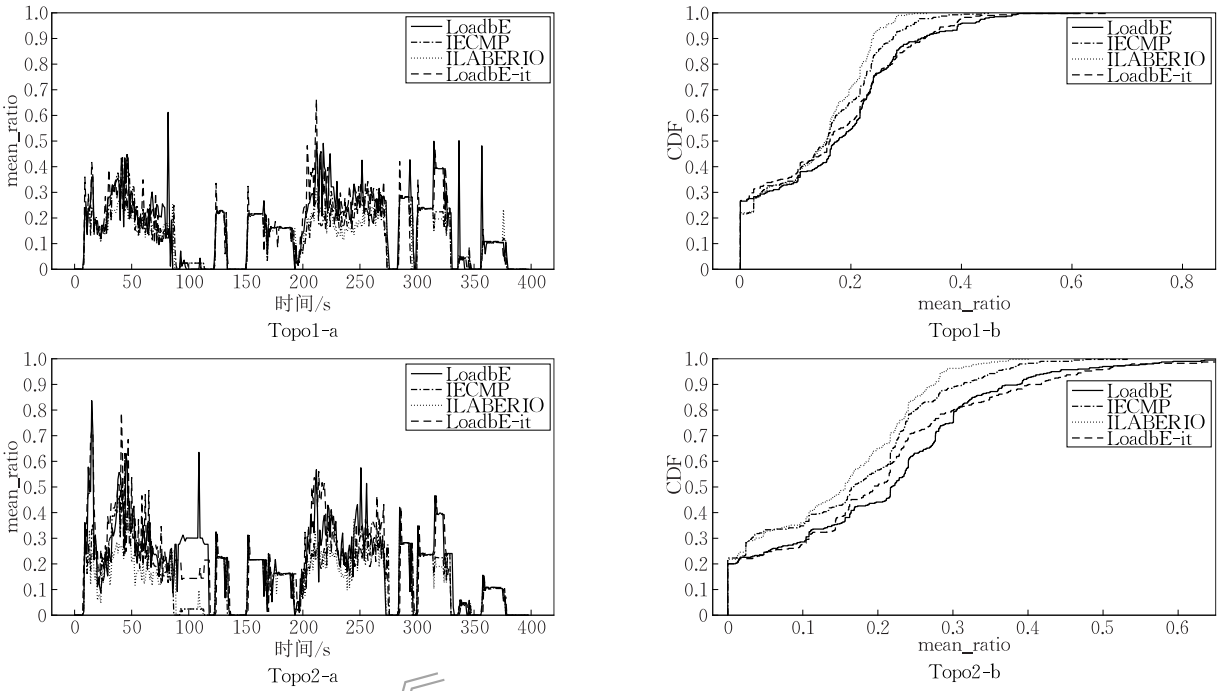


图 8 平均链路利用率

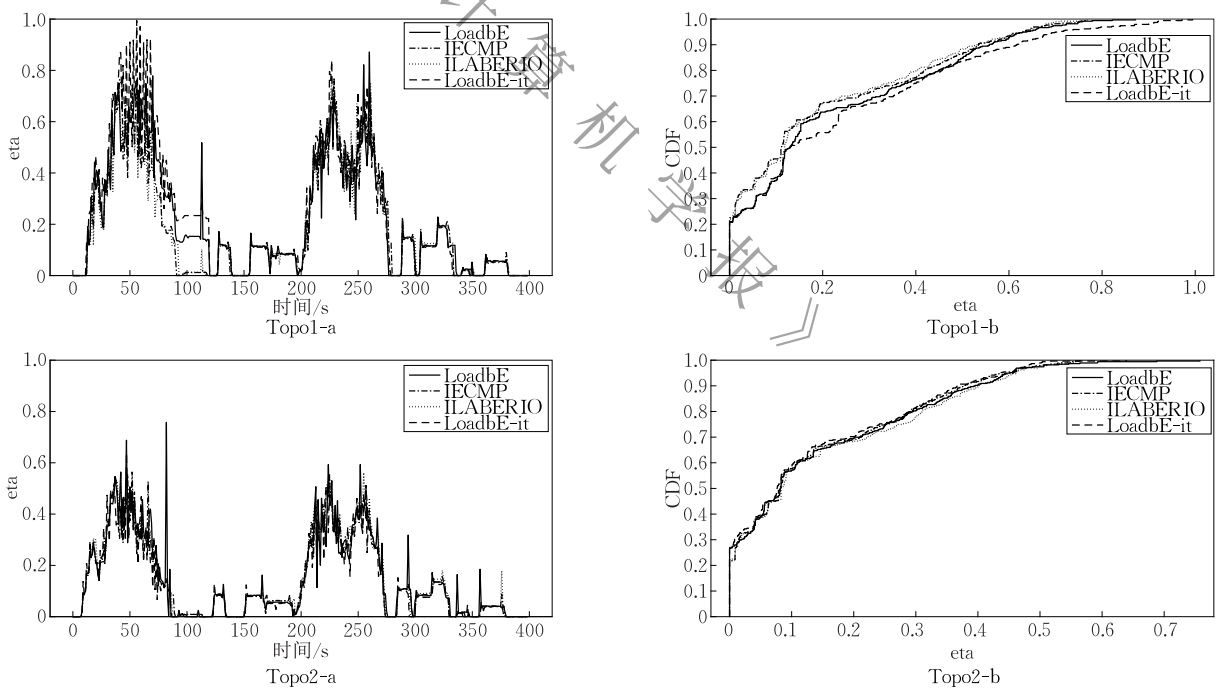


图 9 不同机制能效

LoadbE-it 的曲线最低，即本文提出的算法在能效上优于对比算法，其中 LoadbE-it 最好，ILABERIO 性能最差. 这说明 LoadbE-it 算法虽然频繁调度引起了能效的剧烈波动，但是总能通过合理选择数据流的调度顺序和调度路径保证调度后网络能效最好.

从 Topo2-b 的累积分布曲线中可以看出，四种算法在拓扑 Topo2 中能效相差不明显，但是与 Topo1 相反，能效分布最差的却是 LoadbE-it 算法，这是因为它在迭代式流调度时，转发了相同的流量，却使用了更多的设备，当然使得能效降低.

8 结 语

SDN 的出现使得网络技术的革新变得更为方便. 本文致力于解决 SDN 中的负载均衡与节能问

《计算机学报》

题，首先抽象出负载均衡与节能问题中的共性——数据流调度，然后设计了一种同时兼顾两种因素的路由与流调度机制，最后在两种拓扑上进行了仿真实验，从而得到提出的流调度算法在网络能耗、链路利用率的标准差、平均链路利用率等多个指标上优于对比算法的结论，证实了其可行性和有效性. 然而，本文的设计也存在一些不足之处：如数据流带宽估计不够准确将导致多次流调度；未考虑延迟和抖动约束等问题.

参 考 文 献

[1] Zhang Qingyi, Wang Xingwei, Huang Min, et al. Software defined networking meets information centric networking: A survey. IEEE Access, 2018, 6: 39547-39563

[2] Nagle J. On packet switches with infinite storage. IEEE Transactions on Communications, 1987, 35(4): 435-438

[3] Iselt A, Kirstadter A, Pardigon A, et al. Resilient routing using MPLS and ECMP//Proceedings of the Workshop on High PERFORMANCE Switching & Routing. Phoenix, USA, 2004: 345-349

[4] Heddeghem W V, Lambert S, Lannoo B, et al. Trends in worldwide ICT electricity consumption from 2007 to 2012. Computer Communications, 2014, 50(13): 64-76

[5] Lambert S, Heddeghem W V, Vereechen W, et al. Worldwide electricity consumption of communication networks. Optics Express, 2012, 20(26): 513-524

[6] Wang Xingwei, Zhang Jinhong, Huang Min, et al. A green intelligent routing algorithm supporting flexible QoS for many-to-many multicast. Computer Networks, 2017, 126: 229-245

[7] Zhang Bang, Wang Xingwei, Huang Min. Multi-objective optimization controller placement problem in Internet-oriented software defined network. Computer Communications, 2018, 123: 24-35

[8] Zhang Bang, Wang Xingwei, Huang Min. Adaptive consistency strategy of multiple controllers in SDN. IEEE Access, 2018, 6: 78640-78649

[9] Akyildiz I F, Lee A, Wang P, et al. A roadmap for traffic engineering in SDN-OpenFlow networks. Computer Networks, 2014, 71(3): 1-30

[10] Tootoonchian A, Ganjali Y. HyperFlow: A distributed control plane for OpenFlow//Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking. San Jose, USA, 2010: 1-6

[11] Yu M, Rexford J, Freedman M J, et al. Scalable flow-based networking with DIFANE. ACM SIGCOMM Computer Communication Review, 2010, 40(4): 351-362

[12] Hu Y, Wang W, Gong X, et al. BalanceFlow: Controller load balancing for OpenFlow networks//Proceedings of the 2012 International Conference on Cloud Computing and Intelligent Systems. Hangzhou, China, 2012: 780-785

[13] Handigol N, Seetharaman S, Flajslik M, et al. Aster*x: Load-balancing Web traffic over wide-area networks//Proceedings of the 2013 GENI Engineering Conference. Washington DC, USA, 2013: 1-3

[14] Ruiz-Rivera A, Chin K W, Soh S. GreCo: An energy aware controller association algorithm for software defined networks. IEEE Communications Letters, 2015, 19(4): 541-544

[15] Wang J, Yan Y, Dittmann L. Design of energy efficient optical networks with software enabled integrated control plane. IET Networks, 2014, 4(1): 30-36

[16] Tu R, Wang X, Yang Y. Energy-saving model for SDN data centers. Journal of Supercomputing, 2014, 70(3): 1477-1495

[17] Long H, Shen Y, Guo M, et al. LABERIO: Dynamic load-balanced routing in OpenFlow-enabled networks//Proceedings of the 2013 Advanced Information Networking and Applications. Barcelona, Spain, 2013: 290-297

[18] Wang Y, You S. An efficient route management framework for load balance and overhead reduction in SDN-based data center networks. IEEE Transactions on Network and Service Management, 2018, 15(4): 1422-1434

[19] Xue H, Kim K T, Youn H Y. Dynamic load balancing of software-defined networking based on genetic-ant colony optimization. Sensors, 2019, 19(2): 1-17

[20] Heller B, Seetharaman S, Mahadevan P, et al. ElasticTree: Saving energy in data center networks//Proceedings of the 2010 USENIX Conference on Networked Systems Design & Implementation. San Jose, USA, 2010: 249-264

[21] Kakadia D, Varma V. Energy efficient data center networks—A SDN based approach//Proceedings of the 2012 IBM Collaborative Academia Research Exchange. Bangalore, India, 2012: 1-3

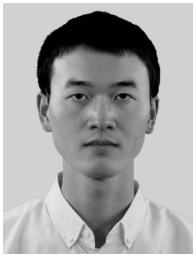
[22] Li D, Yu Y, He W, et al. Willow: Saving data center network energy for network-limited flows. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(9): 2610-2620

[23] Chkirbene Z, Gouissem A, Hadjidj R, et al. Efficient techniques for energy saving in data center networks. Computer Communications, 2018, 129: 111-124

[24] Bianzino A P, Chaudet C, Rossi D, et al. A survey of green networking research. IEEE Communications Surveys & Tutorials, 2012, 14(1): 3-20

[25] Adami D, Antichi G, Garroppo R G, et al. Towards an SDN network control application for differentiated traffic routing//Proceedings of the 2015 IEEE International Conference on Communications. London, United Kingdom, 2015: 5827-5832

计算机学报



LU Yao-Guang, M. S. candidate. His research interests include software defined networking and the routing of mobile social networking.

WANG Xing-Wei, Ph. D., professor. His main research interests include future internet, cloud computing, and cyberspace security, etc.

LI Fu-Liang, Ph. D., associate professor. His research interests include network management and measurement, etc.

HUANG Min, Ph. D., professor. Her main research interests include intelligent algorithm design and optimization, scheduling theory and methods, etc.

Background

Software Defined Networking (SDN), as a new type of network paradigm, solves the problems of overly-expanded network protocols and difficulties in network innovation, however, it still faces load balancing problem and energy saving problems. Although there are many researches on load balancing and energy saving, few mechanisms have been done to solve both of them at the same time, therefore, this paper designs a dynamic load balancing and energy saving mechanism. Firstly, the traffic monitoring mechanism is designed to be responsible for monitoring the real-time status of the network and implementing flow measurement at the data flow level. Secondly, a random routing algorithm based on link preferences is proposed. Finally, based on the traffic monitoring mechanism and the random routing algorithm, load balancing and energy saving flow scheduling (LoadbE) and load balancing and energy saving flow scheduling with iteration (LoadbE-it) are proposed respectively, which optimize the network resource configuration and improve the network performance by reasonable scheduling data flow in the network. Experimental results show that the two flow scheduling algorithms proposed in this paper are more effective and feasible than the benchmark algorithms in terms of average link utilization and energy efficiency.

This work is funded by the National Key Research and Development Project (Grant No. 2019YFB1802802), the National Natural Science Foundation of China (Grant Nos. 61572123, 61872073), the Ministry of Education-China Mobile Research Fund Funding Project (Grant No. MCM20160201).

《计算机学报》