计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190500122

基于 CKSP 的分段路由负载均衡技术

周建新　张志鹏　周　宁
武汉理工大学信息工程学院　武汉 430070
(zjx@whut.edu.cn)

摘　要　针对当前以云计算、大数据为代表的新兴业务需求,现有的 MPLS(Multi-Protocol Label Switching)网络存在协议复杂、扩展性差、运维困难等问题。因此文中采用分段路由(Segment Routing,SR)转发技术,根据软件定义网络(Software-Defined Networking,SDN)集中控制、开放编程的特点,提出了一种基于受限 K 最短路径(Constrained K-Shortest Pathes,CKSP)算法的分段路由负载均衡的技术方案。首先,控制器与各网络节点以 OpenFlow 协议进行信息交互,对全网拓扑结构和链路速率进行监控;然后,分段路由应用根据北向接口以二级流表、多节点中继的方式实现转发表等初始表项的构建和段列表计算;最后,设计了一种根据链路利用率和跳数进行非均匀加权的 CKSP 算法。实验结果表明:该技术方案可以增大网络吞吐量,平滑流量分布,降低数据流平均时延和网络总丢包率。

关键词:分段路由;软件定义网络;负载均衡;受限 K 最短路径;OpenFlow 协议
中图法分类号　TP393

Load Balancing Technology of Segment Routing Based on CKSP

ZHOU Jian-xin,ZHANG Zhi-peng and ZHOU Ning
School of Information Engineering,Wuhan University of Technology,Wuhan 430070,China

Abstract　In view of the current emerging business demand represented by cloud computing and big data,existing MPLS networks have some problems such as complex protocols,poor scalability,and difficulty in operation and maintenance. Therefore,this paper adopted segment routing(SR) forwarding technology. According to the characteristics of centralized control and open programming of Software-Defined Networking (SDN),a technological scheme of segment routing load balancing based on CKSP algorithm was proposed. First,controller exchange information with each network node by using OpenFlow protocol to monitor the topology and link rate of the entire network. Then,the segment routing application implements forwarding table construction and segment list calculation in the way of the two-stage flow table and the multi-node relay according to the northbound interface provided by the controller. Finally,a Constrained K-Shortest Pathes (CKSP) algorithm based on link utilization and hop for non-uniform weighting was designed. The experimental results show that the proposed technology can increase network throughput and smooth traffic distribution,and reduce the average delay of data flows and the packet loss rate of the total network.

Keywords　Segment routing,Software-defined networking,Load balancing,CKSP,OpenFlow protocol

1　引言

分段路由是一种用于简化传统网络控制平面的新型路由技术方案[1]。Internet 工程任务组(Internet Engineering Task Force,IETF)于 2018 年 7 月确定了分段路由建议标准 RFC8402[2]。分段路由起源于 MPLS,且做了进一步的创新,由于该技术与 SDN 天然结合的特性,因此其逐渐成为 SDN 领域的一个重要研究方向[3]。在去除 MPLS 在控制平面上复杂的 LDP(Label Distribution Protocol)和 RSVP-TE(Resource ReSerVation Protocol-Traffic Engineering)协议后,分段路由的集中式部署方案通过控制器统一负责 SR 全局标签(Node SID)和邻接标签(Adjacency SID)的分配,并采取源路由的路径计算方式,SR 交换机只需要根据标签转发数据包而无须知道数据包的源地址和目的地址。该技术方案能够很好地支持服务等级协议(SLA),并且可以提高网络流量传输的灵活性、可用性和可扩展性[4]。另外,分段路由复用已有的 MPLS 转发平面,现有的网络设备仅需少量改动,这便于传统网络的渐进式迁移[5]。

随着云服务提供商日益增长的业务需求,传统 IP 网络的操作、管理和维护(OAM)的难度逐步增加,控制平面和数据平面高度耦合、垂直集成的现存网络无法改变拥塞的网络状况,因此软件定义网络(SDN)应运而生[6]。开放网络基金会(Open Networking Foundation)将 SDN 定义为:一种将网络控制和转发功能解耦,使底层基础设备在应用和服务上抽象

到稿日期:2019-05-22　返修日期:2019-10-20　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(51879211)
This work was supported by the National Natural Science Foundation of China (51879211).
通信作者:周宁(zhouning@whut.edu.cn)

出来通过编程进行直接控制的新型网络架构[7]。当前大型数据中心的计算、网络、存储资源池广泛使用的虚拟化技术，需要 SDN 网络的集中控制的特点进行灵活分配调度，根据网络的结构、负载、时段、灾备和安全等方面的因素实施负载均衡和流量工程，以提高网络链路利用率，增大网络吞吐量，降低时延、丢包率等[8]。在 5G 网络架构的研究中，涌现出 SDN、超密集异构网络、Cloud-RAN、HetNets、SON、ICN 等网络模型[9]，其中最具发展前景的就是 SDN 架构。SDN 减少了硬件设备的限制并且提供了开放的南北向接口，丰富的网络配置可管理复杂多样的网络。

日益庞大的网络规模和复杂多样的业务需求对网络传输性能有着严峻考验，数据流负载均衡的流量工程问题需要更加完善、可靠的解决方案[10]。负载均衡技术是合理有效地分配网络资源，提高网络性能和服务质量的重要途径之一。在数据平面，负载均衡技术包括服务器负载均衡和链路负载均衡；在控制平面，负载均衡技术包括扁平结构和层次结构的控制器负载均衡[11]。本文结合分段路由简化控制平面的优势和 SDN 集中控制、开放编程的特点，采用 OpenFlow 南向接口协议，以各个主机的 TCP/UDP 端口为最细粒度进行多数据流的链路负载均衡。在系统初始化后，新数据流进入分段路由网络时，系统解析数据包包头域，并将网络拓扑结构和链路速率作为 CKSP 算法的输入，在 K 条最短路径中计算一条同时满足预期网络最大流和最小化链路利用率方差的最优路径，依据分段路由转发表将该路径转化为 MPLS 标签构成的段列表，并向源节点和各中继节点下发压栈流表项。数据流的后续报文匹配各节点的压栈流表项、转发表流表项、MAC 流表项等进行数据转发。该技术方案提高了网络流的目的主机接收带宽总和，并且平滑了网络实时流量分布，进而提高了服务质量。

2 相关工作

分段路由技术借助 SDN 集中控制的特点在网络中通过各种改进型路由算法进行负载均衡，从而不同程度地优化网络性能的各项评价指标。这种网络状态的全局视图是传统分布式 IP 网络架构无法实现的[12]。Tkachova 等[13]将 Edmonds-Karp 算法和最大吞吐量调度算法结合，根据流的有无优先级、链路设置容量、链路剩余带宽等参数对直连路径和分支路径进行负载均衡，使高优先级流的可用带宽大于低优先级流的可用带宽，并且使网络的实时传输流量达到最大，但是该结合算法只计算较少的可选路径数，难以为网络流量找到最优路径。Ru 等[14]将网络状态感知模块嵌入转发节点和资源适应管理器，监测节点发送队列长度、缓存空间、发送速率等，并提出 P-Path 算法，该算法通过不同的链路利用率区间建立权值映射表，以最小的权值路径生成最短路径树，有效地降低了链路带宽方差和数据包转发时延。但是该系统侧重于节点状态测量，没有通过控制器动态地提取各条数据流的 QoS(Quality of Service)结果来进行路由计算。Jing 等[15]提出了一种结合遗传算法和蚁群算法的 GAC 算法在 SDN 网络中进行寻路，其首先通过 3 次交叉遗传筛选出最优的 3 条路径，其次在信息素矩阵中赋予筛选路径较高的初始值，最后让蚁群在网络中搜寻可选路径并多次迭代求出最佳路径。与其他群智能算法相比，该算法在大规模网络中的时间开销较小，但是没有评估多条数据流的网络性能。Mao 等[16]提出一种无监督深度学习的网络流量控制算法，通过传统路由协议计算路径集合并下发相应流表，将交换机反馈的流量模式和路径时延以及存在的网络拓扑作为 CNN 的训练集，重复训练后输出最优路径集合。相比传统路由算法，该算法虽然具有更好的服务质量且具备自适应性，但是时间复杂度过大，导致数据包处理时延较长。Qin 等[17]提出了一种多路广播树的路由存储结构并给出了时延和带宽优先的多路径选择算法 MPBTR，其能够有效减小传输时延并增大吞吐量，但是以空间换时间的方法需要消耗较多的存储资源。

上述负载均衡方案没有综合考虑网络吞吐量和链路利用率方差的相互关系，对数据流的丢包率、时延等 QoS 指标的提升有限。本文提出的基于 CKSP 的分段路由负载均衡技术可以通过链路利用率和跳数的非均匀加权，在所有等价最短路径和适量偏离路径形成的路径集合中选择一条能够提高网络吞吐量并减小链路利用率方差的最优路径，从而提升多数据流的 QoS 性能指标。

3 系统结构和数学模型

3.1 系统结构

基于分段路由的 SDN 架构的负载均衡系统分为应用层和控制层、基础设施层，具体如图 1 所示。分段路由主要在应用层实现负载均衡，控制层进行信息中转，基础设施层是系统功能的底层实体。北向接口 API 和南向接口 OpenFlow 协议作为层与层间的交互接口。

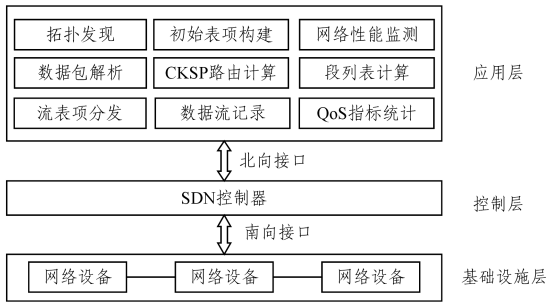

图 1 基于分段路由和 SDN 负载均衡系统结构

Fig. 1 Load balancing system architecture based on SR and SDN

系统的应用层，即分段路由负载均衡应用包括图 1 所示的功能模块。该应用通过 SDN 控制器的北向接口 API 实现分段路由转发和多数据流的负载均衡。其中，拓扑发现模块实现网络拓扑发现和存储；网络性能监测模块利用 Port-status 消息监测网络吞吐量、链路利用率、端口速率等；数据包解析模块解析数据流的 Packet-in 报文的数据链路层、网络层、传输层的 3 层包头信息，如 MAC 地址、IP 地址、TCP/UDP 端口等；流表项分发模块向各网络节点发送添加流表项的 Modify-State 报文；数据流记录模块记录数据包的 3 层解包信息、CKSP 算法的分配路径以及段列表计算结果；QoS 指标统计模块统计所有数据流的 QoS 指标。本文所提算法主要由以下 3 个模块实现。

1)初始表项构建模块：其功能主要是构建分段路由的转发表流表项、MAC 流表项、默认流表项等初始表项。转发表

项采用相同的二级流表项以完成压栈节点的转发表项匹配。转发表项的匹配域为各节点 Node SID，指令集包括标签出栈和向指定端口发送数据包，其中指定端口由从源节点到目的节点的 Dijkstra 算法决定。MAC 表项只用于连接主机的边缘节点，根据 MAC 地址将段列表全部出栈的数据包发送给目的主机。默认流表项优先级最低，将与其他表项均不匹配的数据包上传至 SDN 控制器处理。

2)CKSP 路由计算模块：主要计算从数据包的源节点到目的节点的最优路径。该算法首先根据已经获取的网络拓扑结构确定最短路径跳数，将该跳数加权作为 KSP 的路径数目参数；然后采用偏离路径和局部断路的方法，以多轮基于堆优化策略的 Dijkstra 算法进行选路，并更新备选路径集，在备选路径集中逐次取出最短路径，找到 K 条最短路径；最后根据网络实际流量，对这 K 条路径的链路利用率和跳数进行非均匀加权，选择阻塞程度较小并且跳数较少的路径，即权值最小路径，作为最优路径。

3)段列表计算模块：将计算得到的最优路径转化为 Node SID 构成的分段路由段列表。通过最优路径上各节点转发表的转发端口依次进行匹配，选择能够全部匹配的最远节点的 Node SID 标签进行压栈。当标签数目超过最大栈深度(Maximum Stack Depth，MSD)时，进行多段划分来实现中继节点压栈。当转发表与段列表的计算结果重合度高时可以降低标签资源的开销。

系统的控制层即 SDN 控制器。在分段路由负载均衡应用中，各模块使用了控制器的北向接口服务，包括拓扑、流规则、核心、存储、设备、端口统计等。控制层将服务的抽象接口提供给应用层，应用层的分段路由负载均衡应用根据服务信息和网络实时状态作出路由决策后，控制层生成和下发相应的流规则到基础设施层的网络设备，实现数据流的转发控制。

系统的基础设施层采用的网络设备为 SDN 交换机，能够与 SDN 控制器建立 TCP 长连接，并且支持主流的南向接口——OpenFlow 协议，可以实现主机-交换机-控制器的数据包处理流程。控制器为交换机配置各类流表项。转发表流表项属于两级流表，用以完成同一节点的标签压栈和匹配转发工作，其他流表项均属于一级流表。SDN 交换机的功能主要通过流表的匹配域和指令集来实现。

基于分段路由的 SDN 负载均衡系统的工作流程如下。

1)初始化阶段：首先 SDN 控制器确定网络拓扑，进行节点、链路及主机发现；然后分配各节点全局分段路由标签 Node SID，通过二级流表构建分段路由的转发表流表项，通过一级流表构建 MAC 流表项、默认流表项和其他初始表项。

2)数据包处理阶段：主机发送数据流后，源节点收到主机发送的数据包，并根据默认流表项上传给 SDN 控制器。SDN 控制器解析各数据流第一个上传报文，首先根据网络性能监测结果和 CKSP 算法计算从源节点到目的节点的权值最小路径，然后通过段列表计算模块分配该路径最终所需的各节点压栈标签，最后在源节点和各中继节点下发一级压栈流表项，匹配域为各 TCP/UDP 数据流的 IP 五元组(源 IP 地址，目的 IP 地址，源端口，目的端口，协议号)，指令集为压入指定标签栈。

3)数据流传输阶段：如图 2 所示，依据 OpenFlow 协议的数据包流水线处理机制，数据流在各压栈节点的一级流表中进行标签压栈，再在同一节点的二级流表中匹配转发表并进行标签出栈，完成该节点的路由转发。在路径的其他节点直接在一级流表匹配转发表并进行出栈转发，实现 SDN 控制、分段路由转发的多数据流负载均衡。

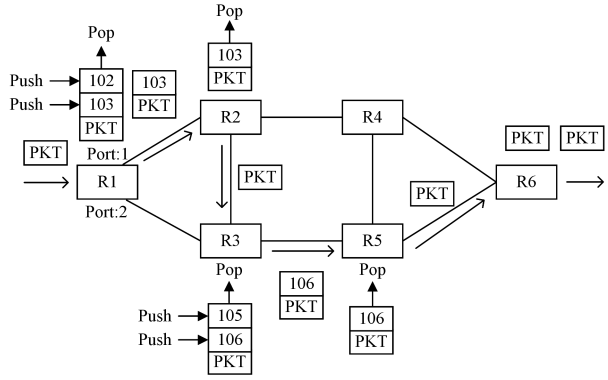

图 2 分段路由多节点压栈和出栈的数据流传输过程

Fig. 2 Segment routing data stream transmission process with multi-node pushing and popping

3.2 数学模型

本文描述的数学模型为网络流模型，下面分析计算网络流的各项评价指标，如网络吞吐量和链路利用率方差，并给出 CKSP 的权值计算式。

使用 $G(V,\boldsymbol{E},\boldsymbol{C},F)$ 以集合形式和矩阵形式表示网络流图，Zn 表示不超过 n 的正整数集。其中 $V=\{v_i \mid i\in Z_n\}$ 表示 n 个网络节点的集合；$\boldsymbol{E}=(e_{ij})_{n*n}$ 表示图中所有节点之间的 n 阶邻接矩阵，当 $e_{ij}=1$ 时存在有向边 $\langle v_i,v_j\rangle$，当 $e_{ij}=0$ 时不存在有向边 $\langle v_i,v_j\rangle$；$\boldsymbol{C}=(c_{ij})_{n*n}$ 表示网络的 n 阶容量矩阵，c_{ij} 为有向边 $\langle v_i,v_j\rangle$ 的最大容限流量且满足 $c_{ij}\geqslant 0$；$F=\{\boldsymbol{f}^m \mid m\in Z_M\}$ 表示有 M 条网络流的集合。$\boldsymbol{f}^m=(f_{ij}^m)_{n*n}$ 表示第 m 条流的 n 阶流量分布矩阵，矩阵 $\boldsymbol{f}^m$ 的元 f_{ij}^m 为第 m 条流在有向边 $\langle v_i,v_j\rangle$ 上的传输速率。

$$f_{ij}=\sum_{m\in Z_M} f_{ij}^m \tag{1}$$

其中，f_{ij} 表示有向边 $\langle v_i,v_j\rangle$ 中 M 条网络流的流量之和。

$$\begin{cases} f_{ij}\leqslant c_{ij} \\ \sum\limits_{i\in Z_n} f_{ki}-\sum\limits_{j\in Z_n} f_{jk}=d_k,1\leqslant k\leqslant n \end{cases} \tag{2}$$

式(2)中的第一个公式为网络流的容量约束，即有向边的负载流量不能超过有向边的最大容限流量；第二个公式为网络流的流守恒约束。由于每个节点既可以作为各条流的起点也可以作为其他流的终点，因此当 $d_k>0$ 时，节点 v_k 表现为向网络内输入流量的源点；当 $d_k<0$ 时，节点 v_k 表现为向网络外输出流量的汇点；当 $d_k=0$ 时，节点 v_k 表现为在网络内部传输流量的中间节点。

本文的研究目标为在考虑负载均衡的情况下解决多源点、多汇点的最大流问题。下面给出分析网络流所需的计算式。

$$N_l=\sum_{i\in Z_n}\sum_{j\in Z_n} e_{ij} \tag{3}$$

$$R_F=\sum_{m\in Z_M} f_e^m \tag{4}$$

其中，N_l 表示网络存在的链路数目；R_F 表示网络边缘对 M 条流的接收流量之和，即网络吞吐量；f_e^m 表示第 m 条流的目的主机成功接收的流量速率。当所有链路满载且所有数据流跳数为 1 时，R_F 取最大值为容量矩阵 $\boldsymbol{C}$ 的 m_1-范数 $\|\boldsymbol{C}\|_{m_1}$。

$$r_u=\frac{1}{N_l}\sum_{i\in Z_n}\sum_{j\in Z_n}\frac{f_{ij}}{c_{ij}} \tag{5}$$

$$D=\frac{1}{N_l}\sum_{i\in Z_n}\sum_{j\in Z_n}\left(r_u-\frac{f_{ij}}{c_{ij}}\right)^2 \tag{6}$$

其中，r_u 表示网络的平均链路利用率；D 表示网络的链路利用率方差，当链路数为偶数且链路利用率为 0 或 1 的链路各占一半时，方差取最大值 0.25。

对于给定参数的网络，流的数量、发送时间、请求带宽、总传输数据量等的不同会造成网络性能的测量结果不同。各链路满载且各条流的跳数最少的理想化情形很难达到，因此负载均衡的路径应在各种数据流存在的情况下都能使网络性能达到相对最优。本文采用的网络性能评估函数为：

$$g(R_F,D) \tag{7}$$

在容量约束和流守恒的约束下，式(7)满足 $\frac{\partial g}{\partial R_F}>0$ 和 $\frac{\partial g}{\partial D}<0$，且偏相关系数的绝对值的相对大小决定了网络吞吐量和负载均衡的效用权重。若选取合适的 R_F 和 D 使式(7)取到 $\max(g(R_F,D))$，则认为此时的流分布为最优流分布，而其他评价指标(如丢包率、时延的优化等)与 $g(R_F,D)$ 呈正相关。

为了增大网络吞吐量，同时降低链路利用率方差使式(7)取到 $\max(g(R_F,D))$，本文提出 CKSP 算法进行路由计算。KSP 算法中 K 值的计算如下：

$$K=w_1 * N^m_{P_{\min}} \tag{8}$$

其中，w_1 为预设权重，$N^m_{P_{\min}}$ 表示通过 Dijkstra 算法计算的第 m 条流中源节点到目的节点的最短路径的跳数。通过这种计算 K 值的方法可以形成以多条等价最短路径为基准路径，并相应增加适量偏离路径的 K 条最短路径集合 S_P。

$$w(P_i)=w_2 * \sum_{L\in P_i}\frac{f_{S_LE_L}}{c_{S_LE_L}}+w_3 * N_{P_i}+w_4 * \max(r_u(P_i)) \tag{9}$$

其中，$w(P_i)$ 表示 S_P 中任意一条路径 P_i 的权值的计算式；w_2，w_3 和 w_4 为预设权重；$P_i\in S_P$，$L\in P_i$，L 表示路径 P_i 中的任意一条链路；S_L 和 E_L 分别表示链路 L 的起始节点序号和终止节点序号；N_{P_i} 表示路径 P_i 的跳数；$\max(r_u(P_i))$ 表示组成路径 P_i 的所有链路中的最大链路利用率。式(9)中等式右边第一项对路径 P_i 的总体链路利用率进行均等加权，可以优先选择低负载的路径以提升负载均衡效用；第二项对跳数进行加权，可以优先选择低跳数的路径以提升网络重度负载情况下的网络吞吐量；第三项对最大链路利用率进行额外加权，降低阻塞程度较大的路径被选中的概率。最后在 S_P 中选择权值最小的路径 P，即为向第 m 条流分配的最佳路径。

4 CKSP 算法设计

CKSP 算法如算法 1 所示。

算法 1 CKSP 算法

输入：网络拓扑 G，源节点 v_s，目的节点 v_d，链路容限速率和实际速率

输出：权值最小的最优路径 p

```
1. p1←getDijkstraPath (G,vs,vd)
2. k←toInt (w1 * size(p1))
3. addResultPath (Pk,p1)
4. for i←1 to (k−1) do
5.    pi←getResultPath (Pk,i)
6.    for j←1 to size(pi) do
7.       vsn←getNode (pi,j)
8.       pir←getRootPath (pi,j)
9.       for pk in Pk
10.         pkr←getRootPath (pk,j)
11.         if size(pk)>=j && isEqual(pir,pkr) then
12.            removeEdge (pk,j,j+1)
13.         end if
14.      end for
15.      removeSurEdges (pir)
16.      psn←getDijkstraPath (G,vsn,vd)
17.      if psn!=null then
18.         pc←mergePath(pir,psn)
19.         addCandidatePath (Pc,pc)
20.      end if
21.      restoreTopo (G)
22.   end for
23.   pm←minPath (Pc)
24.   addResultPath (Pk,pm)
25.   removeCandidatePath (Pc,pm)
26. end for
27. for pk in Pk do
28.   for ei in pk do
29.      ru←getInsRate (ei)/getSetRate (ei)
30.      addPathCost (pk,w2 * ru+w3)
31.   end for
32.   addPathCost (pk,w4 * maxUtilRate (pk))
33. end for
34. p←minCost (Pk)
```

算法 1 的时间复杂度为 $O(Kn(n+m)\log n)$，其中 n 为网络节点数，m 为网络链路数。算法 1 首先通过 Dijkstra 算法计算从源节点 v_s 到目的节点 v_d 的一条最短路径，将该路径的跳数加权后作为 KSP 算法中的 K 值。将已经计算出的各条最短路径存入结果路径集合 P_k，将可能为最短路径的路径存入候选者路径集合 P_c，一条路径从源节点到路径的其他节点的子路径被称为根路径，如上述的 p_{ir} 和 p_{kr}。在计算下一条最短路径时，遍历上一条最短路径的各条根路径 p_{ir}，并且将与结果路径集合 P_k 中相同根路径 p_{kr} 的分叉链路断路，避免计算已经存在的结果路径，同时将根路径 p_{ir} 周围的链路断路，避免形成环路。然后通过 Dijkstra 算法计算从根路径 p_{ir} 的终点 v_{sn} 到目的节点 v_d 的最短路径 p_{sn}，将 p_{ir} 和 p_{sn} 合并后的路径 p_c 存入候选者路径集合 P_c，还原断路链路。遍历完毕后，选择 P_c 中的最短路径 p_m 作为下一条最短路径存入 P_k，同时在 P_c 中删除 p_m。多次循环后可以求出 K 条最短路径 P_k。将 P_k 中每一条路径的链路利用率和跳数进行加权，以提升最大链路利用率的权重，选择权值最小的路径作为最优路径 p，即 CKSP 算法的输出结果。

5 实验验证和结果分析

5.1 实验设置

系统的实验环境如下：Intel(R) Core(TM) i5-4200H

2.80 GHz CPU,8 GB 内存,1 TB 硬盘,Ubuntu 14.04 x64 操作系统,ONOS 1.14.0 控制器,Mininet 2.3.0d4 网络模拟器,Open vSwitch 2.5.7 虚拟交换机,OpenFlow 1.3 南向接口协议。实验拓扑选用层次化网络拓扑 Fat-Tree($k=4$)和扁平化网络拓扑 NSFNet,COST266[18],Matrix($k=6$)[19]。表 1 列出了拓扑参数,其中主机均连接各网络拓扑的边缘交换机。设定网络链路的上行带宽容限和下行带宽容限均为 10 Mbps,链路时延为 3 ms。主机发送数据流在 0～3 Mbps 之间随机均匀分布,数据流的源、目的主机在主机集合中随机选取,发送多轮数据流进行测试。在各种拓扑下重复测试 100 次,取平均值作为实验结果。

表 1 4 种实验网络的拓扑参数

Table 1 Topological parameters of 4 experimental networks

拓扑名称	节点数	链路数	主机数
NSFNet	14	20	9
Fat-Tree($k=4$)	20	32	16
COST266	28	41	21
Matrix($k=6$)	36	60	20

5.2 算法性能评估

实验中,用于对比的路由算法包括深度优先搜索(DFS)、最短路径优先(SPF)、随机 K 最短路径(RKSP)[20]和受限 K 最短路径(CKSP)。在不同网络拓扑中测试各算法的最大网络吞吐量。用 COST266 测试链路利用率方差、平均传输时延和网络总丢包率以分析网络性能的变化趋势。

图 3 为 4 种路由算法在 4 种网络拓扑下的最大网络吞吐量测试结果。可以看出,随着网络拓扑规模的增大,各路由算法的最大网络吞吐量均有不同程度的提升。由于 DFS 算法分配路径的跳数较大,拥塞路径较多,使得数据流的实际接收带宽较小。RKSP 算法在 K 条路径中随机选择的路径为拥塞状态时,也会减少网络吞吐量。SPF 算法跳数最小,数据流占用链路资源较小,但是相同源、目的主机对的路径选择是固定的,影响了流量的动态分配。而 CKSP 算法综合考虑了链路利用率和跳数,在获取网络全局链路信息的基础上,实现了多数据流的流量最优分布,相比 SPF 算法,其数据流带宽平均提升了 19.7%,实现了最大化网络吞吐量。

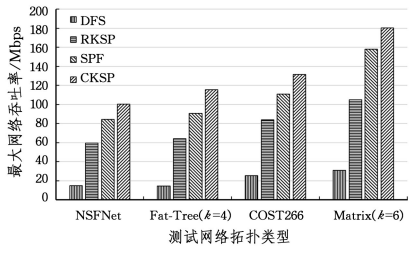

图 3 最大网络吞吐量的对比

Fig. 3 Comparison of maximum network throughput

图 4 为 4 种路由算法在 COST266 中的链路利用率方差的测试结果。通过依次发送数据流可以看到各算法链路负载均衡效果的差异。当网络空载时,链路利用率方差为 0,随着数据流数目的增加,网络中度负载时,链路利用率方差会达到一个峰值,之后网络进入重度负载,各链路利用率都较高,使链路利用率方差有小幅回落。DFS 的路径跳数较大,使网络的部分链路利用率上升过快,网络快速达到拥塞状态,链路负载均匀程度较低。SPF 的路径选择较为固定,导致数据流的分布不够均匀,链路利用率方差较大。RKSP 的随机选路特性使数据流能够相对均匀地分布,在数据流数目较大时其性能接近 CKSP 的性能。CKSP 算法充分考虑了 K 条路径的总体链路利用率和最大链路利用率,能够平滑流量分布,减少各链路利用率的相对变化。CKSP 的峰值比 RKSP 的峰值降低了 22.3%,负载均衡效果最好。

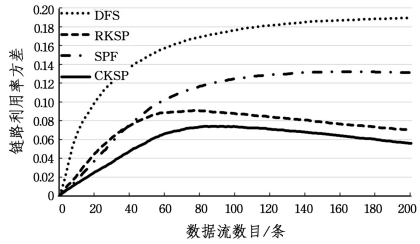

图 4 COST266 中链路利用率方差的对比

Fig. 4 Comparisons of link utilization variances in COST266

图 5 为在 COST266 中 4 种路由算法的数据流平均传输时延的测试结果。随着网络负载的增大,拥塞的数据流数目变多,数据流的平均传输时延也会增大。DFS 的拥塞路径数较多,所以时延较大。SPF 路径跳数比 RKSP 路径跳数较少,经过的阻塞链路较少,所以总时延较小。CKSP 由于负载均衡效果更好,在跳数差异不明显时,网络拥塞程度更小,因此其数据流平均传输时延最小,在网络高负载时比 SPF 减少了 17.3%。

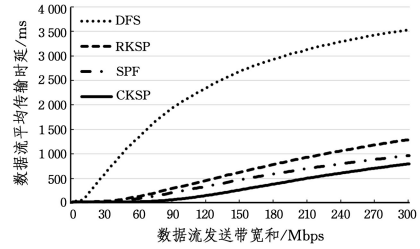

图 5 COST266 中平均传输时延的对比

Fig. 5 Comparison of average transmission delay in COST266

图 6 为在 COST266 中 4 种路由算法的网络总丢包率的测试结果。DFS 拥塞的链路过多,所以总丢包率较大。RKSP,SPF 和 CKSP 的最大网络吞吐量依次增大,所以其总丢包率依次下降。负载均衡通过减少拥塞链路数来减小网络总丢包率,CKSP 最大化网络吞吐量并具有较好的负载均衡效果,在网络高负载时其网络总丢包率比 SPF 减少了 10.9%,能够最小化总丢包率,提高网络成功传输的数据量。

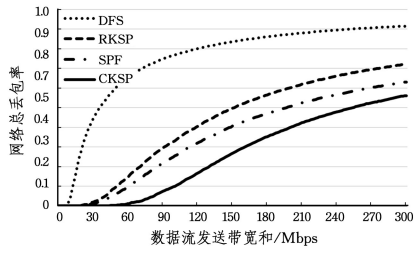

图 6 COST266 中网络总丢包率的对比

Fig. 6 Comparison of total packet loss rate in COST266

结束语 针对当前 MPLS 网络不适应新兴业务需求的问题，本文利用分段路由的源路由特性和软件定义网络的集中控制特点，提出了一种基于 CKSP 的分段路由负载均衡的技术方案。首先，通过 SDN 控制器对全网拓扑结构和流量速率进行监控；然后，根据其提供的北向接口以二级流表实现分段路由的转发表构建，并对各条数据流采用 CKSP 算法结合链路利用率和跳数进行非均匀加权，从而计算最优路径；最后，以多节点中继的方式计算段列表并下发压栈流表项。实验结果表明，该技术方案可以增加最大网络吞吐量，减小链路利用率方差，降低数据流平均传输时延和网络总丢包率。本文的转发表构建和段列表计算存在优化空间，并且测试的网络规模可以适当增大。通过启发式算法进行路由选择并引入差分服务模型，将是下一步的研究目标。

参 考 文 献

[1] ABDULLAH Z N, AHMAD I, HUSSAIN I. Segment Routing in Software Defined Networks: A Survey [J]. IEEE Communications Surveys & Tutorials, 2018, 21(1): 464-486.

[2] FILSFILS C, PREVIDI S, GINSBERG L, et al. Segment Routing Architecture (RFC8402) [EB/OL]. 2018. https://www.rfc-editor.org/info/rfc8402.

[3] LI Z, HUANG L, XU H, et al. Segment routing in hybrid software-defined networking [C]// 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN). Guangzhou: IEEE Press, 2017: 160-165.

[4] FILSFILS C, NAINAR N K, PIGNATARO C, et al. The Segment Routing Architecture[C]// 2015 IEEE Global Communications Conference (GLOBECOM). San Diego: IEEE Press, 2015: 1-6.

[5] GIORGETTI A, SGAMBELLURI A, PAOLUCCI F, et al. Segment routing for effective recovery and multi-domain traffic engineering [J]. IEEE/OSA Journal of Optical Communications and Networking, 2017, 9(2): 223-232.

[6] KREUTZ D, RAMOS F, VERISSIMO P, et al. Software-Defined Networking: A Comprehensive Survey [J]. Proceedings of the IEEE, 2015, 103(1): 14-76.

[7] ONG L. ONF SDN architecture and standards for transport networks: Control architecture and network modeling I M2H. 1 [C]// 2017 Optical Fiber Communications Conference and Exhibition (OFC). Los Angeles: IEEE Press, 2017: 1-41.

[8] JAIN R, PAUL S. Network virtualization and software defined networking for cloud computing: a survey [J]. IEEE Communications Magazine, 2013, 51(11): 23-31.

[9] AGIWAL M, ROY A, SAXENA N. Next Generation 5G Wireless Networks: A Comprehensive Survey [J]. IEEE Communications Surveys & Tutorials, 2016, 18(3): 1617-1655.

[10] JAIN S, KUMAR A, MANDAL S, et al. B4: Experience with a globally-deployed software defined WAN [J]. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 3-14.

[11] LI L, XU Q. Load balancing researches in SDN: A survey [C]// 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC). Macau: IEEE Press, 2017: 403-408.

[12] NUNES B, MENDONCA M, NGUYEN X, et al. A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks [J]. IEEE Communications Surveys & Tutorials, 2014, 16(3): 1617-1634.

[13] TKACHOVA O, YAHYA A R, MUHI-ALDEEN H M. A network load balancing algorithm for overlay-based SDN solutions [C]// 2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T). Kharkiv: IEEE Press, 2016: 139-141.

[14] RU J, ZHE C, HONGBIN L, et al. Status-aware resource adaptation in information-centric and software-defined network [J]. China Communications, 2013, 10(12): 66-76.

[15] JING S, MUQING W, YONG B, et al. An improved GAC routing algorithm based on SDN [C]// 2017 3rd IEEE International Conference on Computer and Communications (ICCC). Chengdu: IEEE Press, 2017: 173-176.

[16] MAO B, TANG F X, FEDLULLAH Z M, et al. A Novel Non-Supervised Deep-Learning-Based Network Traffic Control Method for Software Defined Wireless Networks [J]. IEEE Wireless Communications, 2018, 25(4): 74-81.

[17] QIN K Y, HUANG C H, LIU K W, et al. Multipath Routing Algorithm in Software Defined Networking Based on Multipath Broadcast Tree [J]. Computer Science, 2018, 45(1): 211-215.

[18] ILYAS S M, NAZIR A, BOKHARI F S, et al. A Simulation Study of GELS for Ethernet Over WAN [C]// 2007 IEEE Global Telecommunications Conference (GLOBECOM). Washington, DC: IEEE press, 2007: 2617-2622.

[19] HUANG L, SHEN Q, SHAO W, et al. Optimizing Segment Routing With the Maximum SLD Constraint Using OpenFlow [J]. IEEE Access, 2018, 6(1): 30874-30891.

[20] PAN J, POPA I S, ZEITOUNI K, et al. Proactive Vehicular Traffic Rerouting for Lower Travel Time [J]. IEEE Transactions on Vehicular Technology, 2013, 62(8): 3551-3568.

ZHOU Jian-xin, born in 1976, Ph.D, associate professor. His main research interests include TCP performance of computer network and SDN.

ZHOU Ning, born in 1975, Ph.D, associate professor. His main research interests include SDN and big data of industrial manufacturing.