

基于 SDN 分级分域架构的 QoS 约束路由算法

黄伟, 路冉, 刘存才, 祁思博

(中国电子科技集团公司第五十四研究所, 河北 石家庄 050081)

摘要: 传统分布式的网络架构制约路由算法的创新, 软件定义网络的出现为路由算法的优化提供了新思路。已有研究中, 启发式算法广泛应用于服务质量路由, 但由于计算复杂度高而无法在大型网络中应用。而其他算法均存在不同程度的问题, 要么复杂度较高, 要么算法性能较差, 如最短路径算法。基于 SDN 分级分域架构, 提出了 LC-LD 路由算法, 综合时延条件和代价度量约束并在计算复杂度和算法性能之间保持平衡。仿真分析表明, LC-LD 路由算法在有较低的计算复杂度的同时还有较高的服务质量路由选路性能。

关键词: 软件定义网络; 分级分域; 服务质量; 路由算法

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2019047

QoS routing algorithm based on multiple domain architecture of SDN

HUANG Wei, LU Ran, LIU Cuncai, QI Sibao

The 54th Research Institute of China Electronic Technology Group Corporation, Shijiazhuang 050081, China

Abstract: Traditional distributed network architecture constraints the innovation of routing algorithm. Software-defined network (SDN) provides a new solution for the optimization of routing algorithm. Previous researches show that the quality of service (QoS) routing issues are based on heuristic algorithm mostly, but these methods cannot be applied in large networks due to their high computing complexity. However, other algorithms have a lot of problems, which are high complexity or poor QoS performance, such as shortest path algorithm. This paper proposes A new QoS routing algorithm: LC-LD routing algorithm was proposed. LC-LD was based on SDN west-east interface and binds both delay constraint and cost constraint. keeping a good balance between computational complexity and algorithm performance. Finally, the simulation results show that LC-LD can possess both low computational complexity and high QoS routing performance.

Key words: software defined network (SDN), SDN west-east interface, quality-of-service (QoS), routing algorithm

收稿日期: 2019-01-19; 修回日期: 2019-03-18

通信作者: 祁思博, 773624104@qq.com

基金项目: 国防科技重点实验室基金资助项目 (No.614210401050217)

Foundation Item: The National Defence Science and Technology Key Laboratories Foundation of China (No.614210401050217)

论文引用格式: 黄伟, 路冉, 刘存才, 等. 基于 SDN 分级分域架构的 QoS 约束路由算法[J]. 网络与信息安全学报, 2019, 5(5): 21-31.

HUANG W, LU R, LIU C C, et al. QoS routing algorithm based on multiple domain architecture of SDN[J]. Chinese Journal of Network and Information Security, 2019, 5(5): 21-31.

1 引言

1.1 软件定义网络

路由算法的创新与发展缓慢的根源在于传统分布式网络架构中的诸多缺点与限制,其中包括:所有网络节点都是等价的,每个节点仅能获取自身及周围有限的信息;分布式网络架构由于节点对等,因此很难获取全局实时网络信息,而寻找最短路径的 Dijkstra 算法需要全局网络的拓扑信息,必须使用分布式算法;分布式算法的结果一般只是局部最优解,很难判断是否为最优路径。大多数传统网络的路由算法主要以距离(跳数)或时延为参量规划最优路径,优化的目标以最短路径为主。但由于某些算法以及分布式网络架构的特性,算法结果往往比较简单,没有考虑链路状态参数的实时性,存在局部链路拥塞、整体链路利用率较低等比较严重的问题,难以发现诸多隐藏的错误^[1]。例如,距离向量路由算法中存在的无穷计数问题^[2]、链路状态路由算法由于复杂度较高而导致大规模网络中路由器的内存和 CPU 的消耗问题^[3]。尽管学术界提出许多算法来解决上述问题,但大部分仅停留在研究阶段,难以应用到产业界中。一部分原因是传统网络设备中的路由协议固化,迭代更新比较困难;另外,新设计的协议还存在某些问题,与某些现有网络设备不兼容,兼容性与扩展性较差。

与传统网络的分布式策略不同,SDN 通过提取网络设备中的控制平面到一个逻辑上的集中控制端,实现了数据平面和控制平面的分离^[4],即数据平面的网络交换设备不掌握任何网络信息,只根据控制器下达的指令工作;控制平面的逻辑控制器运行着管理网络的程序,拥有全部的网络信息,与各个交换机通过控制链路直接或间接交换信息,并根据预先设定的协议控制交换机中的流表项^[5]。现有的基于 OpenFlow 协议^[6]的 SDN

使用控制器下发转发规则,每个规则包含一个字段的匹配,匹配包的动作包括丢弃、转发、上送到控制器、优先级和超时。通过抽象的网络动作来定义数据分组行为模式,大大提高了网络的灵活性和扩展性^[7],并且能够更加便捷地对网络进行管理。在集中式控制的新型网络架构下,传统网络中一些路由协议存在的问题得到了解决。例如,SDN 集中式的控制器可实时获取网络内所有节点的信息,形成宏观的全局视图,更容易运行优化算法,部署相应的路由策略;路由协议部署在 SDN 控制器^[8-9]中,可方便地对路由算法进行迭代更新^[10],大大提高了网络的灵活性。

1.2 SDN 分级分域架构

由于目前 SDN 主要以试验性的网络为主,因此其控制平面多为单控制器,仅负责小规模试验性网络。对于大规模网络部署来说,单控制器的性能和可靠性等指标难以满足要求。此外,根据不同网络场景之间的差异,其需求有很大不同。因此为了适应 SDN 未来的发展与演进,需要大规模网络再细化为一些子区域,每个区域由区域内的单控制器单独负责。为了更好地部署 SDN 并满足大规模网络的需求,学术界提出了 SDN 分级分域架构^[11]。在 SDN 分级分域架构中,网络由多个子区域构成,每个子区域总负责的控制平面称为 DCP (domain control plane),DCP 可以由单个域内控制器(DC, domain controller)组成,也可由分布式的控制器集群组成。DCP 负责管理子区域网络内部,子区域之间的通信以及全局网络的宏观信息由更上层 SCP (super control plane)负责。SCP 收集各个子区域的域内信息和区域间的链路状态信息,完成对网络全局性的管理,可由域间总控制器(SC, super controller)组成。SDN 分级分域体系架构信息交互流程如图 1 所示。

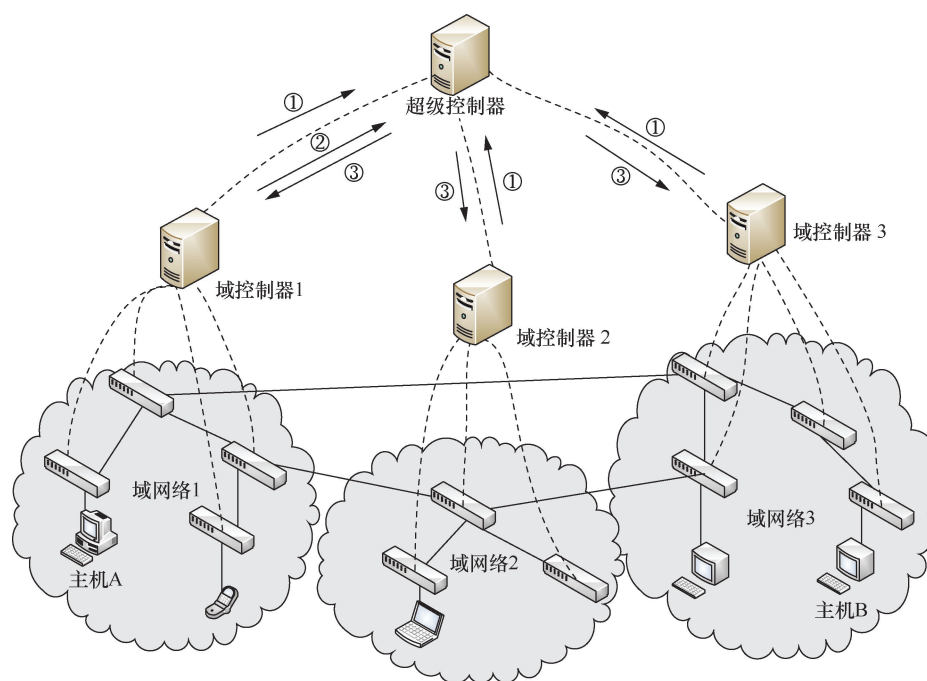


图 1 SDN 分级分域架构示意

SDN 分级分域架构在逻辑上对大规模网络进行了再次分层,简化了大规模网络的管理难度,并且提高了网络的可扩展性,使骨干网的 SDN 化成为可能。例如,在大型运营商网络中,存在接入网、汇聚网、核心网等不同规模的网络。每种网络的规模、业务形态、网络状况及其抽象模型均有所不同,对于网络功能的要求也有所不同。不同的网络根据相应的网络功能需求运行特定的网络应用程序来管理。对于诸如运营级的大型网络来说,分级分域架构可以提高其可扩展性与易操作性,是 SDN 大规模应用的必然条件。在分级分域架构下,子区域内控制器与底层网络设备实时交互获取信息,灵活实现区域内不同的路由策略;区域内控制器上报各自区域路由信息给顶层域间控制器从而实现对于全局性网络管理。综上所述,分级分域网络架构可以在降低路由算法复杂度的同时提高其路由性能,可以实现域内、域间不同的路由策略,进而大大提高网络的灵活性。

1.3 QoS 约束路由算法

QoS 约束路由问题可以概括为,在满足一个或多个条件约束下,寻找一条或多条最优路径,其中“最优”可以有不同的度量方式。基于 QoS 的路由算法就是要找到一条从源点到终点满足 QoS 参数约束的最优路径。几种常用 QoS 参数包括可用带宽、端到端延迟、分组丢失率、抖动和开销,这些参数代表了网络的质量,且具有不同的数学特性。根据这些数学特性, QoS 参数可以归类为以下 3 类:加性参数、乘性参数和瓶颈性参数,其中,加性参数包括时延、抖动、成本和跳数,乘性参数包括传输成功率,瓶颈性参数包括带宽。如果对乘性参数取对数,就可以将其转换为加性参数,最终 QoS 参数可归类为两类参数:瓶颈性参数和加性参数。如果一个路由算法被多个 QoS 参数约束,则这样的算法属于 NP 完全问题^[12],在多项式时间内无法取得精确结果。学术界提出了多种启发式算法来解决这个问题,根据待解决的问题类型和求解方法,可将这些算法分成如下几类:

多项式非启发类、非多项式非启发类、探测类、限定 QoS 参数类、动态规划类和代价函数类^[13]。多 QoS 参数约束条件下的路由算法中,启发式算法和近似算法均存在某些缺陷:首先,这两类算法计算复杂度较高,在大规模网络中无法实际应用;其次,算法性能较低,无法计算出有效的最优路径;另外还存在一部分算法仅针对特殊需求而无法通用需求^[14]。

综上,已提出的路由算法中大部分是以链路状态^[15]或以特定需求为主导来设计路由算法的,如多路径路由算法^[16],一次计算得到多个最优路径以实现负载均衡;动态路由算法,实时调整路由算法参数来最大限度计算出最优路径等。本文以包含大规模节点的 SDN 出发,应用分级分域的新型网络架构^[17],基于时延和代价参数约束,设计可应用于大规模网络的高效路由算法,并保证了相应的高性能以及低复杂度。

2 基于 QoS 约束 SDN 分级分域路由算法

本文在 SDN 分级分域体系架构下,考虑时延敏感的条件,计算当时延受限时,最小化代价的路由算法,即 Delay-Constrained Least Cost 路由问题,简称 DCLC 问题^[18]。本文中选取 2 个 QoS 参数:代价和时延。时延通指链路的传输时延和网络设备内的排队时延等。代价通常为网络传输的开销。在分级分域架构下,域内控制器可实时获取域内节点之间的时延和代价,域间控制器获取部分域内网络节点链路信息,寻找时延受限而代价最小化的路由路径,即 LC-LD (least cost-least delay) 路由算法。

2.1 问题描述

假设一个点对点通信网络可以由一个有向图表示。对于分级分域的网络,假设有 n 个域,每个域内的网络由简单图 $N_i = (V_i, E_i)$ 表示,其中 $0 < i < n$, V_i 表示第 i 个域, E_i 表示第 i 个

域的链路集, $|V|$ 表示域 i 的节点个数。链路 $e = (u, v) \in E_i$ 具有代价度量 $C(e)$ 和时延度量 $D(e)$ 。参数的数值大小为非负数。时延目前主要是传播时延,它和跳数或传输距离相关。代价指网络传输的开销。以上两个参数都具有实时性,在算法的执行过程不发生变化。基于上述假设,本文的目的是寻找源节点和目的节点之间时延满足上限要求,而最小化代价的路由路径。

首先,本文提出域内和域间两种不同的 LC-LD 路由算法的简单版本;之后,本文分析了出现环路的原因,以及如何检测和消除环路;最后证明了经过环路消除算法之后已经不存在环路。同时,下文中 LC 路径表示节点之间的最小代价路径,LD 路径表示节点之间的最短时延路径。

2.2 LC-LD 路由算法描述

在 SDN 分级分域架构下,本文首先介绍域内的 LC-LD 路由算法的简单版本,然后介绍域间路由算法。该算法的目标是寻找一条在时延受限的条件下从源节点到目的节点的最优路径。在构建最优路径时,每个节点有两个可选的输出链路。其中,一个是当前节点到目的节点的 LC 路径,另一个是当前节点到目的节点的 LD 路径。在大量约束条件的限制下,该算法计算得到的路径可能不是严格意义上的最优路径,但大大减少了各个节点的计算量。

定义路径

$$P(v_0, v_k) = v_0, e_1, v_1, e_2, \dots, v_{(k-1)}, e_k \quad (1)$$

其中, v_k 是一组节点和链路, $e_j = (v_{(j-1)}, v_j) \in E_i, 1 \leq j \leq k$ 。 s 表示源节点, t 表示目的节点,设有 K 个 QoS 参数,则 E 中的每一条链路具有 K 个权重值,分别表示 K 个 QoS 参数,记为 $w_1(e), w_2(e), \dots, w_K(e)$, 所有参数均为非负实数。

路由信息由控制平面集中维护,每个节点需要维护如下信息:成本向量、时延向量和路由表。

一个节点的成本向量有 $|V|$ 个条目，每个条目对应网络中的一个节点。成本向量的每个条目包括如下信息。

目的节点 ID, w 。

从节点 v 到节点 w 的最小代价 $\text{least cost}(LC)$ 路径的成本 cost , $\text{least_cost_value}(v,w)$ 。

从节点 v 到节点 w 的 LC 路径的下一个节点的 ID, $\text{least_cost_nhop}(v,w)$ 。

同样地，一个节点的时延向量包括 $|V|$ 个条目，每个条目对应网络中的一个节点。时延向量的每个条目包含如下信息。

目的节点 ID, w 。

从节点 v 到节点 w 的最小时延 $\text{least delay}(LD)$ 路径的端到端时延, $\text{least_delay_value}(v,w)$ 。

从节点 v 到节点 w 的 LD 路径的下一跳节点的 ID, $\text{least_delay_nhop}(v,w)$ 。

假设所有节点的代价向量和时延向量都持续更新。在执行算法的过程中，代价及其向量、时延及其向量均保持不变。

对于跨域路由，本文算法对网络视图做了如下所述的抽象简化。总控制器需要获取的路由信息如下。

各个区域出口之间链路的时延和代价。

区域内源节点和目的节点到该域出口的链路的时延和代价。

各个区域不同出口之间 LC, LD 路径的时延和代价。

区域内部控制器除了需要获取区域内部的信息，还需要上报总控制器 SD 源节点 s ，目的节点 d 到该区域出口的 LC、LD 路径的时延和代价，以及各个区域出口之间的 LC、LD 路径的时延和代价。

经过上述逻辑抽象与简化，原来复杂的、大规模的网络拓扑被抽象为逻辑分层的新网络拓扑。对于新的网络拓扑，部分节点之间会有两条链路路径，即源节点、目的节点到该域的出口，

以及各个域的出口之间存在的两条直连的 LC、LD 路径。简化过后，网络拓扑中的节点包括如下信息：所有域的出口节点信息；源节点目的节点信息。

链路包括如下信息：源节点到该节点所属区域出口的 LC/LD 路径信息；目的节点到所属区域出口的 LC/LD 路径信息；所有区域的出口之间的 LC/LD 路径信息。

经过上述抽象之后，多区域之间的选路问题可以简化为区域内部的选路问题，同时总控制器可以简化为抽象后网络拓扑的区域内部控制器，根据相应路由信息，进行路径选择。

对于域内路由，域内控制器需要根据域内网络节点链路信息，计算维护相应节点时延和成本向量、路由表等路由信息。触发路由路径构建是根据源节点 s 时延向量的最小值 $\text{least_delay_value}(s,d)$ ，即源节点 s 和目的节点 d 之间最短时延路径 LD 的时延阈值。如果 s 和 d 之间不存在时延受限路径，即时延阈值大于时延约束 Δ ，则终止路由算法过程。如果时延受限路径存在，即时延值小于延时时约束 Δ ， $\text{least_delay_value}(s,d) \leq \Delta$ ，则继续路由算法过程。

此时，源节点 s 被记为活跃节点。网络内任意时刻有且仅有一个活跃节点，也就是算法计算得到的最后一个节点。此时，设置变量 delay_so_far 的值为 0， $\text{previous_active_node}$ 值为空。控制器从活跃节点的有效路由信息中读取下一跳节点的标识标签，即当前活跃节点到目的节点的最小开销路径的下一跳节点标识，即为 $\text{least_cost_nhop}(\text{active_node},d)$ ，可以简记为 lc_nhop 。接着验证不等式(2)的正确性。

$$\text{delay_so_far} + D(\text{active_node}, \text{lc_nhop}) + \text{least_delay_value}(\text{lc_nhop}, d) \leq \Delta \quad (2)$$

如果以上不等式成立，即从活跃节点到目的节点存在经过链路（活跃节点， lc_nhop ）延约束

的路径,此时选择活跃节点到目的节点的最小代价路径。反之,选择活跃节点到目的节点的最小小时延路径。这条从活跃节点到目的节点的 LD 路径必然是从源节点到目的节点的时延受限的一条路径中的一部分。否则,算法不会选取之前的活跃节点。经由以上分析,控制器会选择活跃节点的上述任意一种方向,此时,活跃节点中有如下路由信息:

起始节点 ID, s ;

目的节点 ID, d ;

Previous_node=ID of the Previous_active_node;

$next_{node} =$

$\begin{cases} lc_{nhop}, \text{选择LC路径} \\ least_dealy_nhop(active_node, d), \text{选择LD路径} \end{cases}$

Previous_delay=delay_so_far, and

Flag = $\begin{cases} LCPATH, \text{选择LC路径} \\ LDPATH, \text{选择LD路径} \end{cases}$

增加 $D(active_node, next_node)$ 到 delay_so_far。跳转到新的活跃节点重复上述过程,直到目的节点结束整个算法流程。

如果两个节点之间只存在一条链路,即下一跳节点同属于活跃节点到目的节点的 LC 和 LD 路径, $(active_node, d) = least_dealy_nhop(active_node, d)$ 。由上述条件可知,最小时延路径满足时延参数的限制,此时,活跃节点设置节点内路由表的 flag 标志为 LDPATH。而节点间存在两条链路时则有不同的处理,具体见 2.3 节。

2.3 环路消除算法

2.2 节提出的路由算法中,算法计算路径上的每个节点都会选择最小代价或最小时延的相应路径。如果所有节点全部选择最小代价路径或最小时延路径时,最终得到的路由结果不会产生环路现象,因为生成的路径分别为最小代价路径或最小时延路径。然而,若某些节点选择最小代价路

径,其他节点选择最小时延路径时,则可能产生路由环路现象。下面集中分析讨论检测环路的方法和环路消除算法。

图 2 中各个子图代表了产生环路的场景。计算起始节点 A 到目的节点 D 的路径,时延约束不超过 8 个单位。如图 2(a)、图 2(b)和图 2(c)所示,环路形成共经历图 2 所示几个阶段。起始节点 A 到目的节点 D 是一条最小时延路径,链路 (A, B) 是其中的第一部分。节点 B 到目的节点 D 是一条最小代价路径,链路 (B, C) 是其中的第二部分。节点 C 到目的节点 D 是一条最小时延路径,链路 (C, A) 是其中的第三部分。环路 $A \rightarrow B \rightarrow C \rightarrow A$, 具体如图 2(c)所示。

检测环路的方法如下:在活跃节点的路由信息中查找下一跳节点标识前,检查活跃节点的路由表,如果该节点包含源节点和目的节点的相关表项,则已经出现环路。保证活跃节点中路由表项信息不变的前提下,控制器检测到环路后立即运行环路消除算法。此时活跃节点转为不活跃状态。活跃节点回退到环路的上一个节点 previous_active_node,删除节点内部路由表项,此过程持续到发现路由表项信息的 flag 标志设置为 LCPATH,即该节点到目的节点为最小代价路径。找到该节点后不再回退,为避免环路,在节点处选择最小时延路径代替最小代价路径。这种调整不会超过时延约束。节点路由信息调整如下: $next_node = least_delay_value(w, d)$, $flag = LDPATH$ Previous_node, previous_delay, delay_so_far 保持不变。从该节点继续运行相应路由算法计算路径。

在图 2 中运行环路消除算法的过程如下:在节点 A 处检测到环路,则立即沿着环路回退,由于节点 C 处路由信息的 flag 标志位为 LDPATH,故继续回退到节点 B ,同时删除 (C, A) 链路的路由表项(图 2(d))。若节点 B 处路由信息的 flag

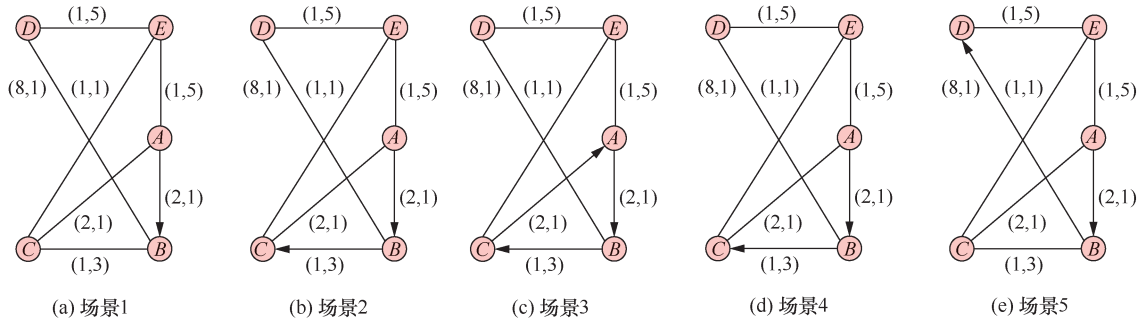


图2 环路消除算法示意

标志位为 LCPATH，则节点 B 调整到沿着最小时延路径的方向并修改相应的路由表项信息，即删除链路 (B, C) ，增加链路 (B, D) 。节点 B 沿着相应路径到目的节点 D ，算法结束。最终路由算法得到的结果如图 2(e)所示。

考虑以下情况：如果活跃节点路由信息的 flag 设置为 LCPATH，回退过程中 active_node 将删除 LC 方向链路，增加 LD 方向链路，此时 LC 方向和 LD 方向链路相同，则环路第二次出现。因此如果下一跳节点同时属于活跃节点到目的节点的最小代价路径和最小时延路径中的一部分，且满足 $\text{least_cost_nhop}(\text{active_node}, d) = \text{least_delay_nhop}(\text{active_node}, d)$ ，则设置活跃节点中路由表的 flag 标志位为 LDPATH。

对于不同区域之间的路由问题，经过抽象简化的网络拓扑中可能在一对节点之间包含两条链路，分别为最小代价路径和最小时延路径。由于回退过程中活跃节点将删除最小代价方向链路，增加最小时延方向链路，同时这两条链路并不相同，那么如果出现上述下一跳节点相同的情况，此时选择最小代价方向，则标记为 LCPATH；后续如果选择最小时延方向，则标记为 LDPATH。

2.4 算法证明

下面证明本文提出的算法的可行性与先进性，即在 SDN 分级分域体系架构下，该算法可以找到一条满足时延受限条件下成本可控的路径，假设该路径存在。给定源节点和目的节点，

如果这两点之间的最小时延路径不满足时延参数的阈值上限，则算法结束。否则，至少存在一条满足条件的路径，开始算法过程；假设源节点为 s ，目的节点为 d ，根据算法计算路径 $P_j = \{v_0 \rightarrow \dots \rightarrow v_j\}$ ，其中 $s = v_0$ ，已计算得到的路径最后一个节点 v_j ，即为活跃节点。初始状态下， $P_0 = \{v_0\}$ ，因为满足条件 $\text{least_delay_value}(s, d) \leq \Delta$ ，又有 $\text{Delay}(P_0) = 0$ ，所以不等式 (3) 成立。

$$\text{Delay}(P_0) + \text{least_delay_value}(v_0, d) \leq \Delta \quad (3)$$

假设不等式 (4) 成立

$$\text{Delay}(P_j) + \text{least_delay_value}(v_j, d) \leq \Delta \quad (4)$$

即路径 P_j 属于源节点和目的节点之间满足时延限制的一条路径。根据所提出的路由算法，下一步是选择从 v_j 到目的节点的下一跳节点，满足最小代价路径或最小时延路径。当选择最小代价路径时，有 $v_{j+1} = \text{lc_nhop} = \text{least_cost_nhop}(v_j, d)$ ，同时满足上文所提不等式 (2)，此时 $\text{delay_so_far} = \text{Delay}(P_j)$ ， $v_j = \text{active_node}$ ， $v_{j+1} = \text{lc_nhop}$ ，所以有

$$\begin{aligned} & \text{Delay}(P_j) + D(v_j + v_{j+1}) + \\ & \text{least_delay_value}(v_{j+1}, d) = \\ & \text{Delay}(P_{j+1}) + \text{least_delay_value}(v_{j+1}, d) \leq \Delta \end{aligned}$$

同样，当选择最小时延路径时，有 $v_{j+1} = \text{ld_nhop} = \text{least_delay_nhop}(v_j, d)$ ，另外在满足式 (2) 的条件下，此时 $\text{delay_so_far} = \text{Delay}(P_j)$ ， $v_j = \text{active_node}$ ， $v_{j+1} = \text{ld_nhop}$ ，所以有

$$\begin{aligned} & \text{Delay}(P_j) + D(v_j + v_{j+1}) + \\ & \text{least_delay_value}(v_{j+1}, d) = \\ & \text{Delay}(P_{j+1}) + \text{least_delay_value}(v_{j+1}, d) \leq \Delta \end{aligned}$$

根据上述条件, v_{j+1} 成为下一个活跃节点, 源节点到此活跃节点的路径也属于满足时延约束的路径的一部分。继续运行算法直至 $\text{active_node} = d$, 算法结束, 此时得到一条满足时延限制的路由路径 (如果存在)。

下面证明本文提出的算法计算得到的路径不包含环路。在活跃节点查找路由表获取下一跳节点信息之前, 检查其他路由表信息, 如果已经有源节点和目的节点的表项信息, 表明环路出现, 此时运行环路消除算法, 因此最终算法得到的路径无环。

2.5 复杂度分析

在 SDN 分级分域架构下每个节点的复杂度为 $O(1)$, 因为控制器已提前计算好节点的 LD 路径和 LC 路径。未形成环路时, 开销最大的情况是从源节点到目的节点的路由经过网络拓扑中的所有节点, 其复杂度与网络拓扑的节点数呈正相关; 在形成环路时复杂度会大幅提高。网络拓扑图中的每个节点到目的节点的最小代价路径构成一棵最小代价树。最小代价树同时是拓扑中的一个最小生成树, 共有 $|V|-1$ 条有向边, 目的节点是网络拓扑中的公共子节点; 同样地, 所有节点到目的节点的最小时延路径可以构成一棵最小时延树。合并最小代价树和最小时延树可以得到新的图 $N' = (V, E')$ 。

属于最小代价树的链路称为树链路, 可以将图 N' 中所有链路分为 3 类, 其中这些链路属于最小时延树但是不属于最小代价树。在最小时延树中的链路中, A 是 B 的直接或间接父节点, 如果在最小代价树中, A 是 B 的直接或间接子节点, 即该链路在最小时延树中和最小代价树中方向相反, 则该链路为反向链路; 反之, 如果 A 是 B 的直接或者间接

父节点, 则称该链路为正向链路; 如果链路不满足上述两个要求, 则称该链路为交叉链路。

图 3 中的 (A, B) , (B, C) , (C, D) , (D, E) 和 (F, C) 是树链路。

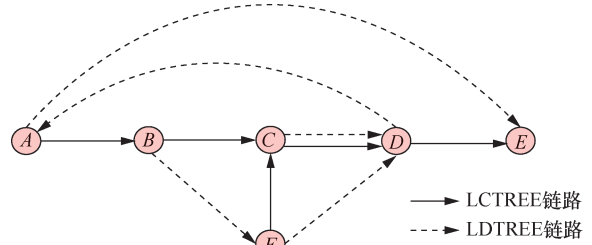


图 3 链路示例图

图 N' 中, 假设共有 X 个反向链路, Y 个正向链路, Z 个交叉链路, 易得 $X+Y+Z \leq |V|-1$ 。根据 2.3 节中环路问题仅可能出现在反向链路中。下面分析本算法在最差情况下的性能。假设每一个反向链路都可能带来环路问题。此时, 对于每一个反向链路, 这条链路可通过 $Y+Z$ 条链路构成环路。最坏情况下, 如果 $Y+Z$ 条链路都是这条链路的上行链路, 则网络中可能多达 $Y+Z$ 个环路。图 3 所示为平均情况, 共有 $\{A \rightarrow B \rightarrow C \rightarrow D\} / \{A \rightarrow B \rightarrow F \rightarrow C \rightarrow D\} / \{A \rightarrow B \rightarrow F \rightarrow D\}$ 。

经过上述分析, 每条反向链路最多产生 $Y+Z$ 个上行链路, 最坏情况下会有 $X(Y+Z)$ 个环路, 考虑到约束条件, 复杂度大概为 $O(|V|^2)$ 。每个环路最长为 $2(|V|)$, 因而本算法在最差情况下的复杂度不超过 $O(|V|^3)$ 。事实上, 本算法在平均情况下性能远超最坏情况。

在分级分域架构中, 假设有 N 个节点, m 个域, 平均每个域有 n 个节点, 即 $N=mn$, 平均每个域有 k 个出口, 且 $k < m$ 。经过前几节抽象之后的网络拓扑的节点数变为 $km+2$ 。相对于原始拓扑, 节点个数减少, 从而大大降低了路由算法的复杂度。

3 仿真结果分析

由上述原理分析, 相对于传统分布式的网络

架构, 经过抽象之后的分级分域的网络结构中网络拓扑规模下降, 使算法复杂度有所下降。下面选取两个算法复杂度较低的服务质量路由方案进行仿真对比分析, 另外引入一种基于启发式多约束最优路径的路由算法 (S-MQOS)^[19]进行对比, 确保 4 种方案网络拓扑内个数相同。具体分析如下。

在 SDN 分级分域结构的条件下, 最小时延路由算法简称 LD, 网络内总控制器获取路由信息的过程如下所示: ① 区域各个出口之间链路的时延和代价; ② 区域内部源节点、目的节点到该域出口的最小时延路径的时延和代价; ③ 各个域的不同出口之间最小时延路径的时延和代价。区域内部控制器有如下任务: ① 获取区域内部网络路由信息; ② 上报总控制器源节点、目的节点到该区域出口的最小时延路径的时延和代价, 以及各个区域出口之间的最小时延路径的时延和代价。经过以上分析简化之后形成新的网络拓扑, 给定源节点和目的节点, 算法计算得到最小时延路径。

另一个简化版本的路由算法是最小代价路由算法, 简称 LC, 网络内总控制器获取路由信息与上述最小时延算法完全相同。区域内部控制器任务如下: ① 获取区域内部路由信息; ② 上报源节点、目的节点到该区域出口的最小代价路径的时延和代价到总控制器, 另外还有各个区域出口之间的最小代价路径的时延和代价。经过以上分析简化之后形成新的网络拓扑, 给定源节点和目的节点, 算法计算得到的是最小时延路径。与 LD 算法不同的是, 如果最小代价路径不符合时延参数的约束要求, 则算法失败。

综上, LC-LD 算法的算法复杂度为 $O(N^3)$, LC 算法和 S-MQOS 与算法方案复杂度为 $O(N^2)$, N 是网路拓扑内节点的个数。由 2.5 节可知, 经过 SDN 分级分域架构抽象后的节点个数 N 大幅减少, 计算复杂度相对传统的情况也大幅降低,

因此, LC-LD 算法拥有较低的算法复杂度。

其次, 仿真网络拓扑各部分参数的生成方式都是随机的, 但基本符合现实网络的情况。这些参数包括: 区域数、区域内的节点数、各链路的代价和时延参数。其中, 区域数服从 5~10 的均匀分布, 区域内的节点数服从 10~20 的均匀分布。各链路的时延和代价参数呈负相关, 大致符合方程 $0.01 \cdot cost + 0.1 \cdot delay = 1$, 式中时延和代价仅代表数量关系, 并大致符合现实网络的情况。在上述参数条件下随机生成 100 个网络, 并在不同时延参数条件的限制下对 4 种算法分别进行了 100 次实验。另外, 如果算法计算得到的路由路径不符合时延参数的约束, 则不记入最后的统计平均结果。最终统计结果如图 4 所示。

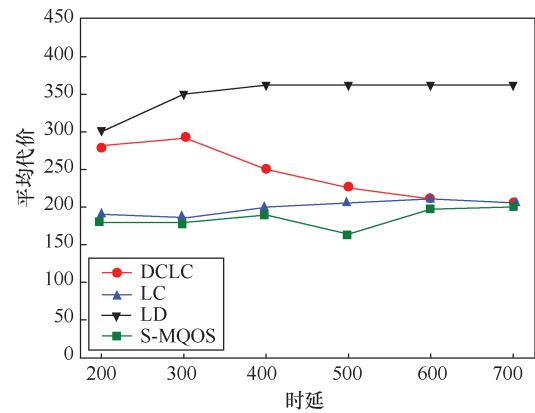


图4 不同时延参数约束下的路由平均代价

不同时延参数约束下的路由平均代价情况如图 4 所示。本文提出的 LC-LD 算法相对于简化之后的 LD 算法平均代价有明显降低, LC-LD 算法随着时延参数约束的增加, 有更大概率选择最小代价路径, 最终平均代价降低; 另外, LC-LD 算法的代价逼近最优情况下 LC 算法的平均代价。S-MQOS 由于其最优性, 在平均代价方面是最低的, 但其算法复杂度较高。

在 4 种算法计算得到的路由满足时延约束的路径的比例即成功率, LC-LD 算法与 LD 算法基本一致, 这是因为当有时延约束时, LC-LD

算法更倾向于选择最小时延路径。同时 LC-LD 算法显著高于 LC 算法,当时延参数约束大于 300 时,计算得到的所有路径几乎满足时延约束。当时延约束不超过 300 时,LC 算法成功率低于 50%。但随着时延约束的提高,选路成功率有提升。而 S-MQOS 由于其复杂性,在时延约束上限超过 500 时,成功率才有所提高(如图 5 所示)。

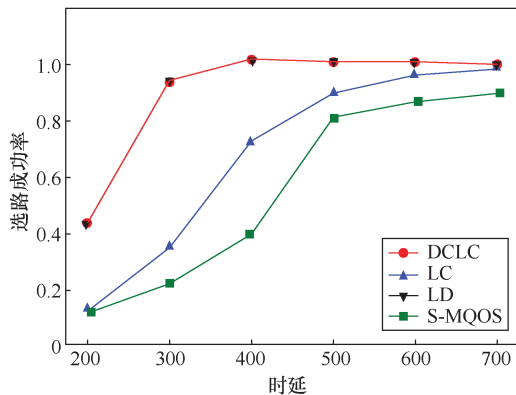


图 5 不同时延限制下选路成功率

综上,对于时延受限、代价尽可能降低的选路问题,本文提出的 LC-LD 算法的选路成功率比 LC 和 S-MQOS 算法大大提高,同时随着时延约束条件的放宽,LC-LD 算法计算得到的结果的平均代价的最优情况与 LC 算法的平均代价十分接近。相对 LD 算法在满足时延约束条件的情况下,LC-LD 算法的选路成功率与 LD 算法几乎相近,但平均代价有明显降低。本文提出的 LC-LD 算法相比其他 3 个算法在总控制器与区域内控制器之间交互信息较多,但算法的性能影响不是很大。

4 结束语

本文提出了基于 SDN 分级分域架构,时延与代价参数最优的服务质量路由算法——LC-LD 路由算法。已有研究中,基于 QoS 的路由算法基本是启发式算法,这类算法由于复杂度过高,无法

在实际的大规模网络中应用。而其他一类算法复杂度较低,如最短路径算法^[20-21],但其算法性能较差。本文算法基于 SDN 分级分域架构,在算法复杂度和算法性能间保持平衡,既保证了较低的算法复杂度使其能应用于现实中的网络,又保证了算法的高效性从而能够计算得到有效的路由路径。LC-LD 算法基于 SDN 分级分域的网络架构,经过抽象之后形成新的网络拓扑,网络节点大幅减少,相对于启发式服务质量路由算法复杂度大大降低。同时仿真结果表明,相对于最小代价路由算法和启发式多约束最优路由算法,LC-LD 算法能保证找到满足时延约束条件的有效路径,并随着时延约束上限的提高,LC-LD 算法计算得到的路径的平均代价与最优情况下最小代价路由算法接近。与最小时延路由算法相比,本算法在满足时延约束条件的情况下,选路成功率与最小时延路由算法相似,但显著降低了链路的平均代价。因此,与复杂度较低的路由算法相比,LC-LD 算法的性能有较大的提升。在后续工作中,LC-LD 算法可以针对组播以及任播等其他网络场景进一步优化。

参考文献:

- [1] WANG Z, CROWCROFT J. QoS routing for supporting resource reservation [J]. IEEE Journal on Selected Areas in Communications, 1996,14(7): 1228-1234.
- [2] HEDRICKCL. Routing information protocol 1988[S].
- [3] MOY J. OSPF Version 2[S]. 1997.
- [4] BIANCO, ANDREA, et al. OpenFlow switching: data plane performance [C]//IEEE International Conference on ICC. 2010: 1-5.
- [5] MCKEOWN N. Software defined mobile networks[C]//Tenth ACM International Symposium on Mobile Ad Hoc Networking & Computing. 2009.
- [6] PFAFF B. OpenFlow switch specification version 1.1.0 implemented (Wire Protocol 0x02) [S]. 2011.
- [7] Open Networking Foundation (ONF). SND Architecture Overview[S]. 2013.
- [8] GUDE N, KOPONEN T, PETTIT J, et al. Nox: towards an operating system for networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(3): 105-110.
- [9] KOPONEN T, CASADO M, GUDE N, et al. Onix: a distributed control platform for large-scale production networks[C]//Usenix Symposi-

- um on Operating Systems Design & Implementation, 2010(10): 1-6.
- [10] ROTHENBERG C E, NASCIMENTO M K, SALVADOR M R et al. Revisiting routing control platforms with the eyes and muscles of software-defined networking[C]//The First Workshop on Hot Topics in Software Defined Networks. 2012: 13-18.
- [11] LIN P, BI J, WANG Y. East-west bridge for SDN network peering [M]//Frontiers in Internet Technologies. Berlin: Springer. 2013: 170-181.
- [12] GAREY M S, JOHNSON D S. Computers and intractability: a guide to the theory of NP-completeness. Oxford: Freeman[M]. W.H., 1979.
- [13] 崔勇, 吴建平, 徐格, 等. 互联网络服务质量路由算法研究综述[J]. 软件学报, 2002, 13(11): 2065-2075.
CUI Y, WU J P, XU L. A survey on research of quality of service routing algorithms in internet[J]//Journal of Software, 2002, 13(11): 2065-2075.
- [14] 董占丽. 基于 SDN 的农垦 EDRP 路由协议研究[J]. 信息技术, 2014, 38(3): 198-202.
DONG Z L. Discussion on Nongken EDRP protocol based on SDN[J]//Information Technology, 2014, 38(3): 198-202.
- [15] 王歆平, 王茜, 刘恩慧, 等. 基于 SDN 的按需智能路由系统研究与验证[J]. 电信科学, 2014, 30(4): 8-14.
WANG X P, WANG Q, LIU E H. Research and Verification on SDN-Based On-Demand Smart Routing System[J]// Telecommunications Science, 2014, 30(4): 8-14.
- [16] DOSHI M, KAMDAR A. Multi-constraint QoS disjoint multipath routing in SDN[C]//2018 Moscow Workshop on Electronic and Networking Technologies (MWENT). 2018: 1-5.
- [17] HATA M, SOYLU M, IZUMI S, et al. Design of SDN based end-to-end routing over multiple domains for mobility management[C]//Network and Service Management (CNSM), 2017 13th International Conference on. IEEE, 2017: 1-4.
- [18] SALAMA H F, REEVES D S, VINIOTIS Y A. Distributed algorithm for delay-constrained unicast routing[J]. IEEE/ACM Transactions on Networking, 2000, 8(2): 239-250.
- [19] 杨建华. 基于启发式多约束最优路径的软件定义网络服务质量路由算法研究[D]. 陕西: 西安电子科技大学, 2015.
YANG J H. Research on quality-of-service routing algorithm in software defined network based on heuristic multi-constraint optimal path[D]. Shaanxi: Xidian University, 2015.
- [20] 陈章迎, 朱尚明. 基于重叠网络的多路径路由设计和实现[J]. 通信学报, 2018, 39(S1): 200-205.
CHEN Y Z, ZHU S M. Designing and implementation of multi-path QoS routing algorithm based on overlay network[J]//Journal

on Communications, 2018, 39(S1): 200-205.

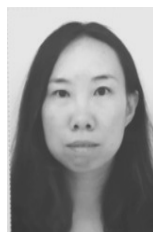
- [21] 孔德慧. 基于 SDN 的多限制多路径 QoS 路由算法研究[D]. 北京: 北京邮电大学, 2018.

KONG D H. Routing algorithm for qos with multi constraints and multipath based on SDN[D]. Beijing: Beijing University of Posts and Telecommunications, 2018.

[作者简介]



黄伟 (1978—), 男, 河北保定人, 中国电子科技集团公司第五十四研究所高级工程师, 主要研究方向为通信系统总体技术、通信系统与网络仿真。



路冉 (1979—), 女, 河北石家庄人, 中国电子科技集团公司第五十四研究所高级工程师, 主要研究方向为通信网络仿真技术和软件定义网络。



刘存才 (1965—), 男, 河北石家庄人, 中国电子科技集团公司第五十四研究所高级工程师, 主要研究方向为通信网仿真和通信网的可靠性。



祁思博 (1991—), 男, 河北石家庄人, 中国电子科技集团公司第五十四研究所工程师, 主要研究方向为通信网仿真和软件定义网络。