软件定义网络中基于分段路由的流量调度方法

董谦[1,2,3]，李俊[1]，马宇翔[1,2]，韩淑君[1,2]
（1. 中国科学院计算机网络信息中心，北京 100190；2. 中国科学院大学，北京 100049；
3. 佛山科学技术学院电子信息工程学院，广东 佛山 528000）

摘 要：针对软件定义网络流量调度的多商品流问题，提出一种基于分段路由的方法。所提方法预先计算所有源—目的节点间的候选路径集合和相应路径的属性，再结合流的各种需求和约束条件设置候选路径的属性应满足的要求，据此筛选得出流的候选路径集合；基于流的候选路径集合简化了软件定义网络中的多商品流模型，降低了求解难度，支持控制器集中控制和各节点自主控制的工作方式，缓解了控制器的可扩展性问题；还讨论了如何满足网络的节能需求，减少可参与流转发的链路数量。性能评估结果表明，所提方法可满足流的各种需求和约束条件，提高网络性能，减轻求解流量调度问题的计算负担。

关键词：分段路由；软件定义网络；流量调度；线性规划
中图分类号：TP393
文献标识码：A
doi: 10.11959/j.issn.1000-436x.2018245

Traffic scheduling method based on segment routing in software-defined networking

DONG Qian[1,2,3], LI Jun[1], MA Yuxiang[1,2], HAN Shujun[1,2]
1. Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China
2. University of Chinese Academy of Sciences, Beijing 100049, China
3. Department of Electronic and Information Engineering, Foshan University, Foshan 528000, China

Abstract: In order to address the multi-commodity flow problem for traffic scheduling in software-defined networking, a method based on segment routing was proposed. The proposed method pre-computed sets of candidate paths and attributes of these paths for all source-target nodes, and set the requirements of attributes of candidate paths that should be met combined with various demands and constraints of flows, then generated sets of candidate paths for flows. In the proposed scheme, multi-commodity flow model in software-defined networking was simplified based on sets of candidate paths for flows, the difficulty of solving was reduced, the centralized control by the controller and the autonomous control by nodes were supported, the scalability of controller was improved. In addition, how to meet the energy-saving needs of the network was proposed, i.e., reducing the number of links that could participate in flow forwarding. The performance evaluation results indicate that the proposed method can meet various demands and constraints of flows, improve network performance, and reduce the computational load of solving the problem of traffic scheduling.

Key words: segment routing, software-defined networking, traffic scheduling, linear programming

1 引言

互联网服务在人们的工作和生活中扮演着十分关键的角色，为此，人们提出多种类型的流量工程（TE, traffic engineering）技术以完成一系列网络优化任务。与此同时，网络管理员对网络流的细粒

收稿日期：2018−05−31；**修回日期**：2018−10−10
通信作者：李俊，lijun@cnic.cn
基金项目：国家重点研发计划基金资助项目（No.2017YFB1401500）；中国科技云建设工程资助项目（No.Y72923）；国家自然科学基金资助项目（No.61672490）
Foundation Items: The National Key R&D Program of China (No.2017YFB1401500), The China Science and Technology Cloud Project (No.Y72923), The National Natural Science Foundation of China (No.61672490)

度管理也变得越来越重要。例如，网络管理员不仅要考虑传统的 QoS（quality of service）需求，如时延、带宽、抖动、丢失分组率等[1]，还要考虑新出现的调度任务类型，如选择若干条不相交路径、部署服务链等。多协议标签交换（MPLS, multi-protocol label switching）是一种十分重要的机制，在这种场景下，标签解耦了转发路径控制和路由协议。但是，基于 MPLS 的 TE 相对复杂，特别是在多条流并存的情况下，通常难以取得性能和开销的平衡[2]。

近年来，分段路由（SR, segment routing）以及它与软件定义网络（SDN, software-defined networking）的结合引起了人们的广泛关注。SR 支持无状态源路由，可以减轻控制器和中间节点的开销，对路径的管理和控制也十分灵活[2]；SDN 能提供网络全局视角，能高效地部署 TE 任务[3]。显然，SDN 结合基于 SR 的全局源路由可针对具体应用需求，对多条流分别进行细粒度的路径管理和控制[4]；相对于 MPLS 或传统 SDN 而言，SR 极大简化了控制面，使控制器无需对中间节点下发转发规则，从而降低了控制器的负载，更好地平衡了网络性能与开销。

所有应用都需要满足特定的要求（约束条件），最典型的例子就是 QoS。除此之外，SR 网络还有一个特有且非常重要的约束条件是进行路径控制时必须满足的：由于设备性能的限制，标签栈最大深度是有限的，这意味着 segment 列表深度（SLD, segment list depth）是有限的[5]。对于有具体优化目标和约束条件的 TE 问题，一方面可建立相应的数学模型，另一方面也需要找到合适的算法来求解。如果基于 SR 解决 SDN 中的流量调度问题，以满足多个应用即多条流的需求，通常会面临以下挑战。

1) 各种各样的需求及约束条件会极大地提高求解问题的难度。一般来说，可用线性规划（LP, linear programming）、整数线性规划（ILP, integer linear programming）、混合整数线性规划（MILP, mixed integer linear programming）模型来表示多商品流（MCF, multi-commodity flow）问题[5-8]，但在多数情况下，它们是 NP-Hard 问题[4,7]，相应的算法必须足够高效。

2) SR 网络中的硬件设备一般有最大 SLD 限制，能用于某条流转发的候选路径使用 segment 列表来表示，而且每个 segment 列表的 SLD 不应超过其硬件限制，因此候选路径不是任意的，必须将最大 SLD 限制以适当形式加入相应的数学模型中[5]。考虑路径的编码问题，将其表示为 segment 列表[9]，使最终求出的解可行。

3) 在很多场景下，控制器到网络设备间的往返时延不可忽视，一旦时延偏高，甚至由于种种原因超时，将会影响设备对相应流的转发[10-11]，这要求网络设备也具备一定的对流转发进行路径管理和控制的能力。支持 SR 的网络设备具有这种能力，但其计算性能有限，必须考虑网络设备如何在不依靠控制器的情况下对流进行控制，同时无需复杂的计算过程。

为此，本文分析了 SDN 中基于 SR 的流量调度模型，针对多个应用的流量需求并存的情况即 MCF 问题，结合最大 SLD 限制等与应用无关的约束条件，提出一种有效的方法以便求解，同时使网络设备也具备一定的对流转发路径进行管理和控制的能力；还简要分析了如何尽量满足网络的节能需求。本文的主要贡献如下。

1) 针对 SDN 流量调度的 MCF 问题，结合 SR 的特点，设计整体架构，对问题进行建模和分析；

2) 提出一种简便的算法，预先计算所有源—目的节点间满足 SLD、等价多路径（ECMP, equal cost multi path）等约束条件的候选路径集合和相应路径的属性，再结合流的各种需求和约束条件筛选得出流的候选路径集合，降低了后续求解的难度；

3) 基于流的候选路径集合，简化 SDN 中的 MCF 模型，为控制器集中控制和各节点自主控制的工作方式设计相应处理流程；

4) 针对网络的节能需求，讨论如何减少可参与流转发的链路数量。

2 SR 简介和相关工作

2.1 SR 简介

SR 采用源路由范式，节点根据一个有序指令列表转发数据分组，这些指令称为 segment。由于 SR 和 MPLS 之间的继承关系，MPLS 的转发面本身并不用修改，segment 以 MPLS 标签的形式表现时，一个有序 segment 列表也就是一个标签栈，栈内标签数量即 SLD，转发时从栈顶标签开始处理；如果 SR 应用于 IPv6，segment 可利用 IPv6 头部中的路由扩展，通过指针来控制当前 segment[12]。

segment 标识符简称为 SID，最常用的 SID 是节点（node）SID 和邻接（adjacency）SID，用于标

识 SR 路由器和它们间的邻接关系。源节点将 segment 列表封装进数据分组的头部，收到数据分组的路由器基于当前（active）segment 处理，对 segment 的动作则有 push、next、continue 3 种。push，在分组头部插入一组 segment；next，处理当前 segment 完毕后转到下一个 segment，与之对应的是 MPLS 中的 POP；continue，当前 segment 并未处理完毕因而保留，与之对应的是 MPLS 中的 SWAP[12]。

SR 数据面决定了设备如何根据 segment 来处理数据分组；SR 控制面定义了如何在设备间传播 SID 信息。SR 一般使用 IGP（interior gateway protocol）通告 SID 信息，与 MPLS 的 LDP（label distribution protocol）等协议相比，简化了控制面；SR 只需在源节点维护流状态，根据源路由的原理和 SR 的转发过程，中间设备无需各类复杂的资源配置机制，只需按照当前 segment 处理数据分组即可，流的转发完全由源节点通过 segment 列表控制；SR 还能很好地支持 ECMP[12]。

总之，在源节点处对流配置一个 segment 列表，这条流的转发路径就确定了。通过 SR 能方便地对每条流进行细粒度管理，却无需修改 IGP 参数，从而保证了网络基础配置的稳定性。

2.2　相关工作

目前已有很多工作讨论如何在 SDN 中进行流量调度，如周桐庆等[13]总结了基于 SDN 的流量工程，将其分为数据层流量调度和控制层流量调度两大类。数据层流量调度的主要目的是借助控制器的全局视角提高链路利用率，避免拥塞；控制层流量调度的主要目的是解决控制器的可扩展性问题。本文注意到，源路由技术与 SDN 结合用于流量调度有突出优势。Li 等[10]对 SD-WAN（SDN for wide-area network）的研究表明，源路由可显著节省控制器建立流转发路径的时间；Dong 等[11]则提出了 AJSR，利用基于 MPLS 的源路由，平衡了下发规则的开销以及网络性能。因此，源路由技术不仅有利于减少控制器建立流转发路径所需的时间，提高网络性能，还能缓解控制器的可扩展性问题。

也有一些工作讨论如何将 SR 用于流量调度。Bhatia 等[6]认为 SR 的关键思想是将路径分解或表达成为若干个 segment，他们只考虑两段 segment 的情况，分别开发了离线和在线流量调度优化算法。Hartert 等[7]提出了一种新的混合约束规划框架来解决 SR 中的流量放置问题，设计了一种名为 SR 路径变量的数据结构，记录已经过的节点，列出接下来可能到达的节点，减小对计算资源的消耗。Hartert 等[4]还设计了一种方法，用来控制电信级网络中的转发路径，提出名为 DFEO 的集中式优化器，通过中间点路由（MR, middle point routing）模型进行计算，而 MR 基于转发图，因此可支持多路径。Moreno 等[5]分析了 SR 网络中的流量模型，认为 ECMP 会导致数据分组重新排序，因此考虑流转发不使用 ECMP 而使用单路径，还要满足 SLD 约束条件。在现实环境下特别是运营商网络中，并非所有设备都支持 SR，混合网络或渐进式部署也是很重要的场景，为此 Cianfrani 等[14]引入分段路由域（SRD, segment routing domain）的概念，区分单个 SRD 和多个 SRD 的场景，建立 MILP 模型以求解 SRD 设计问题，在 SRD 内部则配置合适的流表。总结上述工作的特点如表 1 所示。

表 1 说明，流量调度的主要目的是降低最大链路利用率，如要使用 SR 技术，一般应考虑 SLD 约束条件。SR 支持 ECMP，但流转发是否支持多路径应从实际情况出发，并在建模时体现。针对 MCF 问题，一般先收集相关信息然后求解，再由控制器或网络管理员根据求得的结果将规则分别下发到相关的设备。

表 1　　几种基于 SR 的流量调度方法

方案	网络类型	优化目标	模型	SLD 约束	流转发是否可多路径	路径编码方式	计算方式
文献[6]	SR 网络	最大链路利用率	LP, ILP	2 segment	是	基于中间节点	集中式计算
文献[4]	SR 网络	主要为最大链路利用率，可附加其他	MR	可设定	是	基于中间节点	集中式计算
文献[5]	SR 网络	最大链路利用率	ILP	可设定	否	基于编码算法	集中式计算
文献[14]	混合网络	最大链路利用率	MILP	未讨论	否，扩展后可支持	基于编码算法	集中式计算
本方案	SR 网络	主要为最大链路利用率或节能效果，可附加其他	LP, ILP, MILP	可设定	是	基于路径预计算	集中式计算或各节点自主计算

由于 SR 中路径编码的重要性，一些工作专门就此进行讨论。Giorgetti 等[9]提出两种算法，保证对一条路径求出的 SLD 最小；Guedrez 等[15]提出两种算法，对于邻接 SID 再将其区分为本地和全局两种类型；Cianfrani 等[16]将路径编码与 TE 模型相结合，先基于网络图创建辅助图，然后列出源—目的节点间所有可选择的路径，再基于辅助图求解 MCF 问题。上述工作表明，如果先设定 segment 类型，则可根据路径求出其对应的 segment 列表且使其 SLD 最小。

相关工作分析验证了 SDN 中基于 SR 的流量调度的优势、可行性和效果，表 1 中基于 SR 的流量调度方式多为集中式计算求解路径及流量分担比例。然而，实际 MCF 问题中的约束条件种类较多，加之 SR 网络必须考虑最大 SLD 限制，单路径或多路径也使流量模型更加复杂，即使求出了模型的解，有时还需考虑路径的编码问题，集中式的计算方式如应用于各节点自主控制的场景也较为不便。另外，Lee 等[17-18]提出 MPLS 中的多路径负载均衡算法通常分为两步，第一步计算候选路径，第二步计算流量分割比例。本文还注意到，在进行流量调度相关计算前，预先计算好候选路径的方案有突出优势：Suchara 等[19]提出预计算多条路径则路由器不再需要进行路径计算，能够减轻路由器的负担；Leconte 等[20]提出只要选择合适的预计算路径集合并将其控制在相对小的规模，可极大提高优化计算的速度。

因此，对于 SDN 中相对复杂的流量调度模型，本方法结合 SR 的特点提出了一种预计算候选路径集合的算法，不仅能满足各类需求和约束条件，还记录了候选路径对应的 segment 列表并使其 SLD 最小；根据候选路径集合计算流量分担比例，能够简化 MCF 模型，降低求解难度；本方法也支持各节点自主控制，各节点预先从控制器获取已经计算好的候选路径集合，调度时只需根据流的需求和当前网络状态在候选路径中选择合适路径。

3 问题分析

3.1 整体架构

在 SDN 中，控制器能通过南向接口协议获取网络信息，下发规则。在 IP 网络中，基于 IGP 的扩展能收集当前链路状态等必要信息。SDN 中的流量调度如图 1 所示。

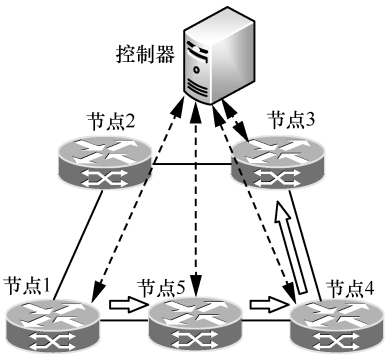


图 1 SDN 中的流量调度示意

图 1 中实线表示节点间的邻接关系即链路，假定有一条流从节点 1 先后经过节点 5、节点 4 转发到节点 3，粗箭头表示流的转发路径走向，虚线表示 OpenFlow[21]等接口协议中控制器与节点间的 packet-in 和 packet-out 交互，控制器需要给这条流转发路径上的每个节点下发相应的转发规则。

将源路由技术 SR 引入 SDN 后，流量调度如图 2 所示。对于同样的一条流，此时控制器只需与源节点 1 交互，对路径进行编码并下发相应的 segment 列表。与图 1 中的 SDN 不同的是，SR 网络通过 SR 控制面传播 SID 信息，控制器无需对中间节点下发转发规则，节省了控制器建立流转发路径所需的时间，减轻了控制器的负担；控制器可专注于根据 MCF 需求和约束条件进行流量调度优化计算，即使控制器失效，SR 网络也具备转发能力，各节点能作为源节点自主控制某条流。

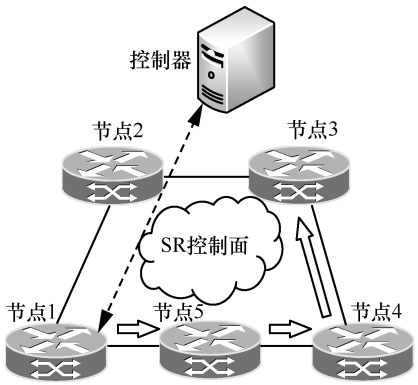


图 2 SDN 中基于 SR 的流量调度示意

3.2 模型建立与分析

对于 SDN 流量调度问题，可定义有向图 $Graph=(N, E)$，N 为节点集合，从节点 i 到 j 的直连链路为(i,j)，链路(i,j)也表示为链路 e，链路容量（带

宽）为 c_e，E 为链路集合；以节点 i 为起点的所有链路(i,j)的集合为 E^i_{out}，以 i 为终点的所有链路(j,i)的集合为 E^i_{in}。需调度的流表示为 f，F 为 f 的集合，f 的大小为 $size_f$，f 在 e 上负担的部分大小为 x_e^f，f 的源节点和目的节点表示为 s_f, t_f，每条流均满足

$$\begin{cases} x_e^f \geqslant 0, \text{当流转发支持多路径} \\ x_e^f \in \{0, size_f\}, \text{当流转发只支持单路径} \end{cases}, \forall f \in F, e \in E \tag{1}$$

$$\sum_{e \in E^i_{\text{out}}} x_e^f - \sum_{e \in E^i_{\text{in}}} x_e^f = \begin{cases} size_f, i = s_f \\ 0, i \neq s_f, t_f \\ -size_f, i = t_f \end{cases}, \forall f \in F, i \in N \tag{2}$$

$$\sum_{e \in E^i_{\text{out}}} x_e^f \leqslant size_f, \forall f \in F, i \in N \tag{3}$$

$$\sum_{e \in E^i_{\text{in}}} x_e^f \leqslant size_f, \forall f \in F, i \in N \tag{4}$$

式(1)~式(2)表示流量平衡条件；默认情况下这条流转发支持多路径，若使 x_e^f 只取 0 或 $size_f$，则限制这条流转发只支持单路径。式(3)~式(4)能在流转发支持多路径的情况下防止流的源、目的节点处产生环路，在流转发只支持单路径的情况下防止所有节点处产生环路。再考虑链路容量限制，设 e 已用的容量为 u_e，显然 $0 \leqslant u_e \leqslant c_e$；$F$ 中所有流 f 的 x_e^f 之和为 y_e，链路利用率为 λ_e，最大链路利用率为 θ，则 $\lambda_e = \dfrac{y_e + u_e}{c_e}$，$\theta = \max\limits_{e \in E}\{\lambda_e\}$，而且

$$\max_{e \in E}\left\{\frac{u_e}{c_e}\right\} \leqslant \theta \leqslant 1 \tag{5}$$

$$y_e = \sum_{f \in F} x_e^f, \quad \forall e \in E \tag{6}$$

$$y_e \leqslant \theta c_e - u_e, \quad \forall e \in E \tag{7}$$

式(5)可确定 θ 的取值范围，式(6)~式(7)表示链路容量条件。必要时还可考虑链路是否可参与流转发，定义二进制变量 l_e 表示链路 e 是否可参与流转发，l_e=0 表示不可以，l_e=1 表示可以，则式(7)变为

$$y_e \leqslant (\theta c_e - u_e) l_e, \quad \forall e \in E \tag{8}$$

式(8)表示链路 e 可参与流转发的条件。如优化任务是最小化最大链路利用率，可表示为

$$\min \theta \tag{9}$$

所有链路 l_e 为 1 时，式(1)~式(7)是 SDN 流量调度的 MCF 模型里的基本约束，所有流转发支持多路径时为 LP 模型，只支持单路径时为 ILP 模型。然而，如对流进行细粒度管理，例如满足应用的时延要求，或满足服务链中必经某节点的要求，每条流的需求都应体现，使得约束条件更多，通常导致求解难度增加；有些条件甚至难以表达，例如 SR 特有的 SLD 约束难以用线性关系式写出。

为此，本文提出一种基于 SR 的流量调度方法，为简化讨论只使用节点 SID，不使用邻接 SID。先做如下说明。

1) 将路径集合记为 $\boldsymbol{P}$，每个 $\boldsymbol{P}$ 都以矩阵形式表示。矩阵 $\boldsymbol{P}$ 的列数等于 m，m 为 E 中元素的数量，对所有链路编号，将链路记为 e_z，链路编号 z=1,2,…,m。矩阵 $\boldsymbol{P}$ 的行数表示为 $K(\boldsymbol{P})$，因此，$\boldsymbol{P}$ 由 $K(\boldsymbol{P})$个一行 m 列的一维数组即行向量 $\boldsymbol{p}$ 组成，即 $K(\boldsymbol{P})$是矩阵 $\boldsymbol{P}$ 中 $\boldsymbol{p}$ 的数量。若考虑空集即行数为 0 的空矩阵[22]，则 $K(\boldsymbol{P})$最小为 0。

2) 每个 $\boldsymbol{p}$ 的实际意义是能用一个 segment 列表表示的一条路径或多条等价路径，分别规定如下：若 $\boldsymbol{p}$ 表示一条路径，则 $\boldsymbol{p}$ 中第 z 列元素的值表示当这条路径负担的流量为 1 时链路 e_z 上的流量大小，故路径没有重复使用 e_z，则路径经过 e_z 时元素值为 1，不经过时为 0；若 $\boldsymbol{p}$ 表示多条等价路径且这些路径可编码为一个 segment 列表（按某些 segment 转发且存在 ECMP 负载均衡时，相应各条等价路径上的流量分担比例一般是固定的），则根据这些等价路径负担的流量之和为 1 时链路 e_z 上的流量大小设定 $\boldsymbol{p}$ 中第 z 列元素的值。流量平衡条件体现在 $\boldsymbol{p}$ 的元素值中。

3) 对每个 $\boldsymbol{p}$ 设置若干属性，$\boldsymbol{p}$ 的属性也就是路径的属性，例如：$ecmp_{\boldsymbol{p}}$ 表示 $\boldsymbol{p}$ 中是否存在 ECMP 负载均衡，存在则为 1，否则为 0；$sld_{\boldsymbol{p}}$ 表示对 $\boldsymbol{p}$ 编码所需 SID 数量的最小值；表示 $\boldsymbol{p}$ 的 segment 列表记为 $sl_{\boldsymbol{p}}$，$sl_{\boldsymbol{p}}$ 的获得基于 4.1 节中的路径产生过程；$cost_{\boldsymbol{p}}$ 表示 $\boldsymbol{p}$ 的路由代价；$hop_{\boldsymbol{p}}$ 表示 $\boldsymbol{p}$ 的跳数；$delay_{\boldsymbol{p}}$ 表示 $\boldsymbol{p}$ 的时延；$\boldsymbol{nodein_p}$、$\boldsymbol{nodeout_p}$ 分别表示 $\boldsymbol{p}$ 负担流量时每个节点是否有流量流入、流出，均为一行 g 列的一维数组即行向量形式，g 为 N 中元素的数量，对所有节点编号，将节点再记为 n_h，节点编号 h=1,2,…,g，$\boldsymbol{nodein_p}$、$\boldsymbol{nodeout_p}$ 中第 h 列元素的值为 1 分别表示节点 n_h 在 $\boldsymbol{p}$ 负担流量时有流量流入、流出，否则为 0。还可根据需要设置其他类型的属性。

本文方法的主要步骤如下。

1) 通过 4.1 节中的路径产生过程可获得网络拓扑中所有源—目的节点间的候选路径集合和相应

路径的属性。源节点 s 和目的节点 t 间的候选路径集合再记为 $\boldsymbol{P}_{st}$，所有 $K(\boldsymbol{P}_{st})$应大于等于 1；所有节点对之间的 $\boldsymbol{P}_{st}$ 及相应 $\boldsymbol{p}$ 的属性作为控制器进行后续处理和 MCF 计算的依据；对于每个网络节点，可预先将以它为源节点的 $\boldsymbol{P}_{st}$ 及相应 $\boldsymbol{p}$ 的属性下发给它，使网络节点无需进行复杂计算便可筛选得出候选 $\boldsymbol{p}$ 并对流转发进行控制。

2) 每条流 f 可根据 s_f、t_f确定其对应的 $\boldsymbol{P}_{st}$，流 f 的候选路径集合再记为 $\boldsymbol{P}_f$，$\boldsymbol{P}_f$为其对应的 $\boldsymbol{P}_{st}$ 的非空子集，借助 $\boldsymbol{p}$ 的属性可由 $\boldsymbol{P}_{st}$筛选得出 $\boldsymbol{P}_f$以满足 f 的需求和约束条件。例如：流转发只支持单路径则要求 $ecmp_{\boldsymbol{p}}$为 0；SLD、路由代价、跳数、时延等要求则通常是设置 $sld_{\boldsymbol{p}}$、$cost_{\boldsymbol{p}}$、$hop_{\boldsymbol{p}}$、$delay_{\boldsymbol{p}}$ 所允许的上限；必经某节点则要求 $\boldsymbol{nodein}_{\boldsymbol{p}}$或 $\boldsymbol{nodeout}_{\boldsymbol{p}}$中对应的元素值为1。按实际意义$\boldsymbol{p}$为$\boldsymbol{P}_f$中的一行可写作$\boldsymbol{p}\in\boldsymbol{P}_f$，设流$f$在$\boldsymbol{p}\in\boldsymbol{P}_f$上负担的部分大小为 $x_{\boldsymbol{p}}^f$，有

$$\begin{cases} x_{\boldsymbol{p}}^f \geqslant 0, & \text{当流转发支持多路径} \\ x_{\boldsymbol{p}}^f \in \{0, size_f\}, & \text{当流转发只支持单路径} \end{cases}, \quad \forall f \in F, \boldsymbol{p} \in \boldsymbol{P}_f \tag{10}$$

$$\sum_{\boldsymbol{p}\in\boldsymbol{P}_f} x_{\boldsymbol{p}}^f = size_f, \quad \forall f \in F \tag{11}$$

3) 对于 F 中每条流 f，依次将 $\boldsymbol{P}_f$中每个 $\boldsymbol{p}$ 对应的 $x_{\boldsymbol{p}}^f$ 组成 $K(\boldsymbol{P}_f)$行一列的一维数组即列向量 $\boldsymbol{X}_f$，即 $\boldsymbol{P}_f$与 $\boldsymbol{X}_f$的同一行分别为 $\boldsymbol{P}_f$中的某个 $\boldsymbol{p}$ 以及这个 $\boldsymbol{p}$ 对应的 $x_{\boldsymbol{p}}^f$，$\boldsymbol{X}_f$是流 f 对应的解；再将 F 中所有流按一定顺序排列，依此顺序分别将 F 中所有流的 $\boldsymbol{X}_f$ 组成 $\sum_{f\in F} K(\boldsymbol{P}_f)$ 行一列的一维数组即列向量 $\boldsymbol{X}$ 以及将 F 中所有流的 $\boldsymbol{P}_f$组成列数为 m 的矩阵 $\boldsymbol{A}$，$\boldsymbol{A}$ 的转置矩阵 $\boldsymbol{A}^{\mathrm{T}}$ 则为 m 行 $\sum_{f\in F} K(\boldsymbol{P}_f)$ 列；依次将所有链路的可用容量组成 m 行一列的一维数组即列向量 $\boldsymbol{B}$，$\boldsymbol{B}$ 中第 z 行元素的值为 b_z，有

$$b_z = \begin{cases} \theta c_e - u_e, & \text{当} l_e = 1 \\ 0, & \text{当} l_e = 0 \end{cases}, \quad \forall e \in E, \ z\text{为}e\text{的编号} \tag{12}$$

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X} \leqslant \boldsymbol{B} \tag{13}$$

式(12)和式(13)为链路容量条件和链路可参与流转发的条件，流量平衡条件体现在 $\boldsymbol{A}$ 和式(10)及式(11)中，其他需求和约束条件体现在 $\boldsymbol{A}$ 中，θ 的取值范围由式(5)确定，至此简化了 SDN 中的 MCF 模型；式(9)是 TE 的经典优化任务[4]，还可添加其他优化目标，并对每个优化目标设定合适优先级和权重[23]。

4　方法描述

本文方法首先求出网络拓扑中所有源—目的节点间的候选路径集合，记录相应路径的属性，然后以此为基础，针对控制器和各节点自主控制分别设计相应处理流程进行流量调度。

4.1　路径产生

在 SR 网络中，$sld_{\boldsymbol{p}}$ 为 1 表示 segment 列表中只有目的节点的 SID，此时转发路径是从源节点到目的节点的最短路径或存在 ECMP 负载均衡的多条等价最短路径。网络拓扑确定则链路的路由代价即权重已知，可由最短路径算法得到所有节点对之间的最短路径或多条等价最短路径；最短路径不存在环路，这是后续排除有环路路径过程的前提。

将 s, t 间的最短路径集合再记为 $\boldsymbol{P}_{st\text{-}1}$，将 $sld_{\boldsymbol{p}}$=2,3,…,q 的候选路径集合分别再记为 $\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$、…、$\boldsymbol{P}_{st\text{-}q}$，且 $\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$、…、$\boldsymbol{P}_{st\text{-}q}$ 在计算前应初始化为空集即行数为 0 的空矩阵，q 为所允许的最大 $sld_{\boldsymbol{p}}$，即 $sld_{\boldsymbol{p}}\leqslant q$。为便于表述，本文也将所允许的最大 $sld_{\boldsymbol{p}}$ 称为 $sld_{\boldsymbol{p}}$ 最大值。先根据最短路径算法求出网络拓扑中所有节点对之间的 $\boldsymbol{P}_{st\text{-}1}$，显然任意一对 s, t 间的 $\boldsymbol{P}_{st\text{-}1}$ 都只有一个 $\boldsymbol{p}$，且这个 $\boldsymbol{p}$ 的 $sl_{\boldsymbol{p}}$ 为目的节点的 SID，$sld_{\boldsymbol{p}}$=1；若 s, t 间只有一条最短路径，则这个 $\boldsymbol{p}$ 表示这条最短路径；若有多条等价最短路径，则这个 $\boldsymbol{p}$ 表示存在 ECMP 负载均衡的多条等价最短路径；记录相应 $\boldsymbol{p}$ 的属性。

然后以计算 $\boldsymbol{P}_{st\text{-}2}$ 的过程为例进行描述。计算的基本思路是在 N 中取 s, t 以外的节点 v 作为中间节点，将每次拼接时从 s 到 v、从 v 到 t 的 $\boldsymbol{p}$ 分别再记为 $\boldsymbol{p}_{前}$、$\boldsymbol{p}_{后}$，每次拼接后得到一个从 s 到 t 的 $\boldsymbol{p}_{总}$，对每个可能的节点 v，先看 $\boldsymbol{P}_{sv\text{-}1}$ 是否为空集即空矩阵，如果不是则将 $\boldsymbol{P}_{sv\text{-}1}$ 中的所有 $\boldsymbol{p}$ 依次作为 $\boldsymbol{p}_{前}$，将 $\boldsymbol{P}_{vt\text{-}1}$ 中的 $\boldsymbol{p}$（$\boldsymbol{P}_{vt\text{-}1}$ 必然只有一个 $\boldsymbol{p}$）作为 $\boldsymbol{p}_{后}$，示意图如图 3 所示。按照 $\boldsymbol{p}$ 的定义，$\boldsymbol{p}_{前}$ 和 $\boldsymbol{p}_{后}$ 相加得到 $\boldsymbol{p}_{总}$；再针对所有类型的 $\boldsymbol{p}$ 的属性，分别设置及计算 $\boldsymbol{p}_{总}$的待定属性如下：对 $\boldsymbol{p}_{前}$ 和 $\boldsymbol{p}_{后}$ 的 $ecmp_{\boldsymbol{p}}$求最大值得到 $\boldsymbol{p}_{总}$的 $ecmp'$；将 $\boldsymbol{p}_{前}$ 和 $\boldsymbol{p}_{后}$ 的 $sld_{\boldsymbol{p}}$、$cost_{\boldsymbol{p}}$、$hop_{\boldsymbol{p}}$、$delay_{\boldsymbol{p}}$、$\boldsymbol{nodein}_{\boldsymbol{p}}$、$\boldsymbol{nodeout}_{\boldsymbol{p}}$ 分别相加得到 $\boldsymbol{p}_{总}$的 sld'、$cost'$、hop'、$delay'$、$\boldsymbol{nodein}'$、$\boldsymbol{nodeout}'$；在 $\boldsymbol{p}_{前}$的 $sl_{\boldsymbol{p}}$ 添加 $\boldsymbol{p}_{后}$ 的 $sl_{\boldsymbol{p}}$ 得到 $\boldsymbol{p}_{总}$的 sl'。每个 $\boldsymbol{p}_{总}$及它的待定属性计算完成后应根据原则 1)~原则 3)进行检验，以

决定是否将其添加到 $\boldsymbol{P}_{st\text{-}2}$ 中，若添加则同时也将待定属性记录为相应的属性。

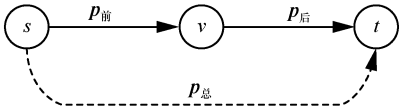


图 3　预计算 $\boldsymbol{p}$ 的过程示意

原则 1) 排除有环路的 $\boldsymbol{p}_{总}$（必选）。一旦有环路产生，必然存在相应的无环路路径，则有环路的路径不可能是最优解。如果 $\boldsymbol{p}_{总}$ 的 ***nodein'*** 或 ***nodeout'*** 中有值为 2 的元素，则说明 $\boldsymbol{p}_{总}$ 有环路存在，应直接排除。

原则 2) 排除重复表达的 $\boldsymbol{p}_{总}$（必选）。上述过程也是对路径编码的过程，同一个 $\boldsymbol{p}$ 可能存在多种 segment 列表的编码方式。可将 $\boldsymbol{p}_{总}$ 与 $\boldsymbol{P}_{st\text{-}1}$ 中的所有 $\boldsymbol{p}$ 及 $\boldsymbol{P}_{st\text{-}2}$ 中已添加的所有 $\boldsymbol{p}$ 逐个比较，如果与其中某个 $\boldsymbol{p}$ 相等则说明 $\boldsymbol{p}_{总}$ 重复表达，应直接排除。

原则 3) 排除不满足需求的 $\boldsymbol{p}_{总}$（可选，若无需求可跳过）。根据实际情况设置 $\boldsymbol{p}_{总}$ 的待定属性应满足的条件，典型例子是已知所有流转发只支持单路径则 *ecmp'* 应为 0，如不满足相应条件应直接排除。

简要给出计算所有节点对之间的 $\boldsymbol{P}_{st\text{-}2}$ 的算法，如算法 1 所示。

算法 1　计算所有节点对之间的 $\boldsymbol{P}_{st\text{-}2}$

1) 输入 N，所有节点对之间的 $\boldsymbol{P}_{st\text{-}1}$，相应 $\boldsymbol{p}$ 的属性
2) for each $s \in N$ do
3) for each $t \in N \setminus \{s\}$ do
4) 初始化 $\boldsymbol{P}_{st\text{-}2}$ 为空集即行数为 0 的空矩阵
5) for each $v \in N \setminus \{s, t\}$ do
6) if $\boldsymbol{P}_{sv\text{-}1}$ 为空集即空矩阵 then
7) else
8) for $temp$ = 1 to $K(\boldsymbol{P}_{sv\text{-}1})$ do
9) 将 $\boldsymbol{P}_{sv\text{-}1}$ 中的第 $temp$ 个 $\boldsymbol{p}$ 即第 $temp$ 行作为 $\boldsymbol{p}_{前}$，$\boldsymbol{P}_{vt\text{-}1}$ 中的 $\boldsymbol{p}$ 作为 $\boldsymbol{p}_{后}$，由 $\boldsymbol{p}_{前}$、$\boldsymbol{p}_{后}$ 及它们的属性计算 $\boldsymbol{p}_{总}$ 及它的待定属性
10) if （$\boldsymbol{p}_{总}$ 有环路） or （$\boldsymbol{p}_{总}$ 重复表达） or （$\boldsymbol{p}_{总}$ 不满足需求） then
11) else
12) 添加 $\boldsymbol{p}_{总}$ 到 $\boldsymbol{P}_{st\text{-}2}$，记录相应的属性
13) end if
14) end for
15) end if
16) end for
17) end for
18) end for

其中，“\”是相对补集符号。计算所有节点对之间的 $\boldsymbol{P}_{st\text{-}3}$ 时，将算法 1 第 2)~第 18)中的 $\boldsymbol{P}_{st\text{-}2}$ 替换为 $\boldsymbol{P}_{st\text{-}3}$，$\boldsymbol{P}_{sv\text{-}1}$ 替换为 $\boldsymbol{P}_{sv\text{-}2}$，$\boldsymbol{P}_{vt\text{-}1}$ 保持不变，而且根据上述原则 2)进行检验时，应将 $\boldsymbol{p}_{总}$ 与 $\boldsymbol{P}_{st\text{-}1}$、$\boldsymbol{P}_{st\text{-}2}$ 中的所有 $\boldsymbol{p}$ 及 $\boldsymbol{P}_{st\text{-}3}$ 中已添加的所有 $\boldsymbol{p}$ 逐个比较，故输入增加所有节点对之间的 $\boldsymbol{P}_{st\text{-}2}$ 和相应 $\boldsymbol{p}$ 的属性；计算 $\boldsymbol{P}_{st\text{-}4}$ 时则将算法 1 第 2)~第 18)中的 $\boldsymbol{P}_{st\text{-}2}$ 替换为 $\boldsymbol{P}_{st\text{-}4}$，$\boldsymbol{P}_{sv\text{-}1}$ 替换为 $\boldsymbol{P}_{sv\text{-}3}$，$\boldsymbol{P}_{vt\text{-}1}$ 保持不变，将 $\boldsymbol{p}_{总}$ 与 $\boldsymbol{P}_{st\text{-}1}$、$\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$ 中的所有 $\boldsymbol{p}$ 及 $\boldsymbol{P}_{st\text{-}4}$ 中已添加的所有 $\boldsymbol{p}$ 逐个比较，输入也相应增加；以此类推直到求得所有节点对之间的 $\boldsymbol{P}_{st\text{-}q}$ 为止。最终，对于每对 s, t，还有 $\boldsymbol{P}_{st\text{-}1}$ 中的 $\boldsymbol{p}$ 未根据上述原则 3)进行检验，如无需求或满足需求则由 $\boldsymbol{P}_{st\text{-}1}$、$\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$、…、$\boldsymbol{P}_{st\text{-}q}$ 取并集即组成一个列数为 m 的矩阵，如不满足需求则由 $\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$、…、$\boldsymbol{P}_{st\text{-}q}$ 取并集，得到 $sld_{\boldsymbol{p}} \leqslant q$ 且满足需求（如果有，例如 $ecmp_{\boldsymbol{p}}$ 为 0）的 $\boldsymbol{P}_{st}$，记录相应 $\boldsymbol{p}$ 的属性。

4.2　流量调度

控制器预计算得出所有节点对之间的 $\boldsymbol{P}_{st}$。流量调度目标为 $\min \theta$ 时，控制器先收集流集合 F、当前链路信息，所有链路 l_e 通常为 1，也可根据要求设定 l_e。然后由 f 对应的 $\boldsymbol{P}_{st}$ 筛选得出 $\boldsymbol{P}_f$，设定 θ 的精度要求，控制 θ 对应的临时变量 θ' 用于试探求解 $\boldsymbol{X}$，所有流转发支持多路径时为 LP 模型，只支持单路径时为 ILP 模型。简要给出求解算法，如算法 2 所示。

算法 2　优化任务为 $\min \theta$ 时控制器求解 $\boldsymbol{X}$

1) 输入 F，所有节点对之间的 $\boldsymbol{P}_{st}$，相应 $\boldsymbol{p}$ 的属性，当前链路信息，所有链路 l_e，θ 的精度要求
2) for each $f \in F$ do
3) 根据 f 对应的 $\boldsymbol{P}_{st}$ 筛选得出 $\boldsymbol{P}_f$
4) end for
5) 根据式(5)，结合 θ 的精度要求设定 θ' 的取值范围，再将 θ' 设定为其取值下限即满足式(5)且符合 θ 的精度要求的最小值，并以 θ' 试探求解 $\boldsymbol{X}$
6) if $\boldsymbol{X}$ 有解 then
7) $\theta \leftarrow \theta'$
8) else
9) 使 θ' 按 θ 的精度要求增加并在增加后再以 θ' 试探求解 $\boldsymbol{X}$，无解则重复此步骤直到有解为止
10) $\theta \leftarrow \theta'$

11) end if

θ'调整的方式也可使用其他更高效的方式。控制器根据当前条件及精度要求下求得的 θ 最小值对应的解 $\boldsymbol{X}$ 配置流转发，segment 列表直接取自 $sl_{\boldsymbol{p}}$。

各节点自主控制进行流量调度时，先从控制器获取其作为源节点的所有 $\boldsymbol{P}_{st}$。当它自主控制某条流 f 时，只考虑当前网络状况，收集链路信息，此时 MCF 问题变为这条流的路径选择问题，一般来说求解算法与算法 2 相似，只是 F 变为 $\{f\}$，$\boldsymbol{X}$ 变为 $\boldsymbol{X}_f$，若想使求得的解是唯一的，可添加其他优化目标。

4.3 节能控制

节能控制是流量工程的重要领域[24]，其中一个主要手段是通过流量调度来减少使用的设备资源，从而降低能耗。Zhang 等[25]提出 GreenTE，在满足链路利用率、时延等约束条件的情况下，最大化可进入休眠状态的链路数量，此时链路不负担任何流的转发；考虑网络设备的实际情况，通常是改变物理连接的状态达到节能效果，物理连接对应双向链路，如果休眠或关闭某个物理连接，意味着与之对应的双向链路都不负担任何流的转发。可将一条物理连接上两个不同方向的链路 e 分别再表示为 e^+和 e^-，相应的 l_e 也再记为 l_{e^+} 和 l_{e^-}，有

$$l_{e^+} = l_{e^-},\ \forall e^+, e^- \in E \tag{14}$$

定义节能效果即不可休眠或关闭的物理连接数量为 SUM，将节能控制的优化任务设为使 SUM 最小，即可参与流转发的链路数量最少，有

$$SUM = \frac{\sum_{e\in E} l_e}{2} \tag{15}$$

$$\min SUM \tag{16}$$

根据式(16)进行优化计算并通过流量调度实现时，一般 θ 是约束条件而非优化目标，应设定 θ 值，将其代入相关约束条件。此时除式(10)~式(13)外，还应满足式(14)~式(15)。根据式(12)，l_e 包含在 $\boldsymbol{B}$ 中，求解前应先收集链路信息，根据收集的信息设定所有链路 l_e 的取值范围：u_e>0 时链路 e 已使用，相应的 l_e 必为 1；u_e=0 时链路 e 暂未使用，相应的 l_e 可取 0 或 1。控制器进行流量调度时，需求解 $\boldsymbol{X}$ 和所有链路 l_e，所有流转发支持多路径时为 MILP 模型，只支持单路径时为 ILP 模型。简要给出求解算法，如算法 3 所示。

算法 3 优化任务为 min SUM 时控制器，求解 $\boldsymbol{X}$ 和所有链路 l_e

1) 输入 F，所有节点对之间的 $\boldsymbol{P}_{st}$，相应 $\boldsymbol{p}$ 的属性，当前链路信息，θ

2) for each $f \in F$ do

3) 根据 f 对应的 $\boldsymbol{P}_{st}$ 筛选得出 $\boldsymbol{P}_f$

4) end for

5) 设定所有链路 l_e 的取值范围

6) 求解 $\boldsymbol{X}$ 和所有链路 l_e，根据式(15)得 SUM

各节点自主控制进行流量调度时，先从控制器获取其作为源节点的所有 $\boldsymbol{P}_{st}$。当它自主控制某条流 f 时，只考虑当前网络状况，收集链路信息，设定所有链路 l_e 的取值范围，此时 MCF 问题变为这条流的路径选择问题，求解算法与算法 3 相似，只是 F 变为 $\{f\}$，$\boldsymbol{X}$ 变为 $\boldsymbol{X}_f$，优先选择已用链路。

4.4 算法分析

由算法 1 可知，其执行时间与执行算法 1 第 9)~第 13)计算并检验 $\boldsymbol{p}_{总}$ 的次数有关，对于一组 s, t，计算 $\boldsymbol{P}_{st\text{-}2}$ 时需要计算并检验 $\boldsymbol{p}_{总}$ 的次数 $NUM_{st\text{-}2}$ 为

$$NUM_{st-2} = \sum_{v\in N\setminus\{s,t\}} K(\boldsymbol{P}_{sv-1}) \tag{17}$$

因此，算法 1 计算所有节点对之间的 $\boldsymbol{P}_{st\text{-}2}$ 需要计算并检验 $\boldsymbol{p}_{总}$ 的次数 NUM_2 为

$$NUM_2 = \sum_{s\in N}\sum_{t\in N\setminus\{s\}} NUM_{st-2} = (g-2)\sum_{s\in N}\sum_{t\in N\setminus\{s\}} K(\boldsymbol{P}_{st-1}) \tag{18}$$

计算所有节点对之间的 $\boldsymbol{P}_{st\text{-}3}$ 时，需要计算并检验 $\boldsymbol{p}_{总}$ 的次数为 NUM_3，将式(18)中的 NUM_2 替换为 NUM_3，$\boldsymbol{P}_{st\text{-}1}$ 替换为 $\boldsymbol{P}_{st\text{-}2}$，以此类推。$N$ 中元素的数量 g 不变，故根据一定原则检验 $\boldsymbol{p}_{总}$ 控制了 $\boldsymbol{P}_{st\text{-}2}$、$\boldsymbol{P}_{st\text{-}3}$ 等的规模，减轻了路径预计算的负担。

控制器进行流量调度时，根据算法 2 和算法 3，$\boldsymbol{X}$ 的元素数量等于所有流 f 的 $K(\boldsymbol{P}_f)$之和，链路 e 的数量即 l_e 的数量是固定的，流量调度模型中的变量数量通常较多，控制器可使用 Gurobi[26]等优化器求解；已有工作表明，选择合适的预计算路径集合[20]能缩小后续求解的搜索空间，降低求解难度，还能避免使用过长的路径[25]，而 5.2.3 节的评估结果表明，对 $\boldsymbol{p}$ 进行筛选减少流量调度模型中的变量数量后，后续求解的难度降低。网络设备作为源节点自主控制某条流时，只考虑这条流的路径选择问题，求解难度更低；特别是当流转发只支持单路径时，通常无需求解 ILP 模型，只需依次检验 $\boldsymbol{P}_f$ 中的每个 $\boldsymbol{p}$ 是否可行，若都不可行则调整 θ'或 l_e 后再次检验。

对于所有流可使用任意多路径或单路径转发

的 SDN 流量调度模型，增加约束条件通常会增加求解难度，部分约束例如 SLD 约束甚至难以表达；本方法预先计算候选路径集合，根据一定要求即约束条件对 $\boldsymbol{p}$ 进行筛选意味着在后续求解时无需再考虑这些约束条件，从而简化了 MCF 模型。本方法基于源路由技术，控制器无需给中间节点下发转发规则；即使控制器失效，各节点只要已获得了相关 $\boldsymbol{P}_{st}$ 便能自主控制；对 $\boldsymbol{p}$ 进行筛选降低了后续求解的难度。这些特点均缓解了控制器的可扩展性问题。

5　性能评估

本文从公开数据集 SNDLib[27]获取网络拓扑信息（包括节点、链路、链路的容量及路由代价）和流量数据，拓扑为 GÉANT 和 Germany50，分别有 22 个和 50 个节点，考虑分别有 72 条和 176 条链路。

5.1　实验拓扑分析

首先基于 GÉANT 的网络拓扑信息进行分析，根据最短路径算法，GÉANT 中所有节点对之间的最短路径均无 ECMP，故 $ecmp_{\boldsymbol{p}}$ 必然为 0；分别设定 $sld_{\boldsymbol{p}}\leqslant 2, 3, 4$，根据 4.1 节中的原则 1)~原则 2)，分别计算得出所有 462 对源—目的节点间相应的 $\boldsymbol{P}_{st}$；统计所有节点对中 $K(\boldsymbol{P}_{st})$小于等于 r 个的源—目的节点对的数量，除以此拓扑中源—目的节点对的总数即 462 后得到比例 $d(r)$，$d(r)$表示 $K(\boldsymbol{P}_{st})$在数量上的分布情况。结果如图 4 所示。再基于 Germany50 的网络拓扑信息进行分析，根据最短路径算法，Germany50 中部分节点对之间的最短路径有 ECMP，故 $ecmp_{\boldsymbol{p}}$ 为 0 或 1；分别设定 $sld_{\boldsymbol{p}}\leqslant 2, 3$，再用 SP 表示所有流转发只支持单路径，MP 表示所有流转发支持多路径，区别在于前者 $ecmp_{\boldsymbol{p}}$ 应为 0 而后者无此要求，根据 4.1 节中的原则 1)~原则 3)，分别计算得出所有 2 450 对源—目的节点间相应的 $\boldsymbol{P}_{st}$。结果如图 5 所示。

图 4 和图 5 表明 $sld_{\boldsymbol{p}}$ 最大值不同时，$d(r)$随 r 变化的趋势相似；$d(r)$相同时，$sld_{\boldsymbol{p}}$ 最大值越大则 r 越大。根据 4.1 节中的原则 1)~原则 3)计算 $\boldsymbol{P}_{st}$ 能使 $K(\boldsymbol{P}_{st})$的大小控制在相对有限的范围。图 5 还表明，$sld_{\boldsymbol{p}}$ 最大值及 $d(r)$相同时，若部分节点对之间的最短路径有 ECMP，由于 $ecmp_{\boldsymbol{p}}$ 还应为 0，所有流转发只支持单路径时的 r 通常小于支持多路径时的 r。

5.2　流量调度优化效果评估

5.2.1　GÉANT 拓扑流量调度优化效果

从 GÉANT 的流量矩阵（TM, traffic matrix）数据集中取出 3 个作为用于评估的 3 个 TM；设 TM

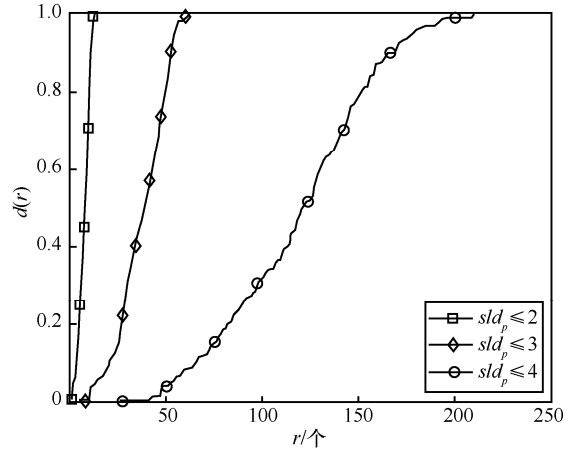


图 4　GÉANT 网络拓扑分析结果

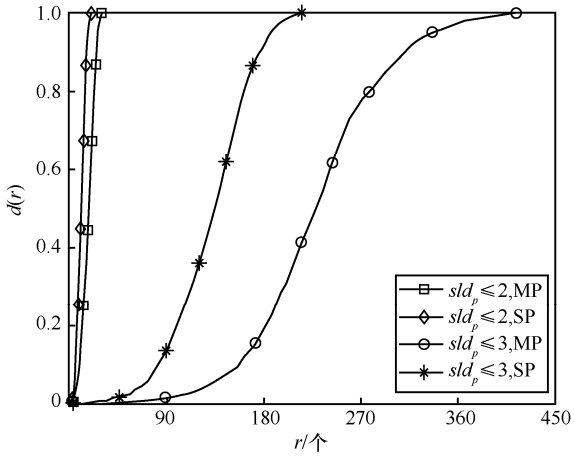


图 5　Germany50 网络拓扑分析结果

中的每对源—目的节点间有一条流 f，$size_f$ 为它们之间的流量需求大小；这 3 个 F 分别用 GÉANT-F_1、GÉANT-F_2、GÉANT-F_3 表示，各有 462 条流。再用 Con 表示有控制器，用 nCon 表示无控制器即各节点自主控制。流量调度优化任务为 min θ，θ 的精度要求设为 1‰，优化效果用当前条件优化计算求得的 θ 评估且越小越好，为方便表示，再定义最大链路利用率优化效果 Θ 为

$$\Theta=\frac{F\text{在当前条件下优化计算求得的}\theta}{F\text{在SDN使用多路径时优化计算求得的}\theta} \tag{19}$$

式(19)中，SDN 使用多路径指采用 3.2 节中约束条件为式(1)~式(7)的流量调度模型且所有流转发支持多路径，此时只考虑了基本约束且多路径能更合理地均衡负载，因此每个 F 在此条件下求得的 θ 是相应精度要求下的最优值，故 Θ 越小越好且最小值为 1。基于 5.1 节中计算得出的 $sld_{\boldsymbol{p}}\leqslant 4$ 的 $\boldsymbol{P}_{st}$，对每条流 f 筛选其对应的 $\boldsymbol{P}_{st}$，分别得出 $sld_{\boldsymbol{p}}$ 最大值为 2、3、4 条件下的 $\boldsymbol{P}_f$（因 $ecmp_{\boldsymbol{p}}$ 必然为 0，故 SP 和 MP

这两种情况下使用相同的 $\boldsymbol{P}_f$)。以所有流沿最短路径转发的情况作为对比，表示为 sld_p 最大值为 1。

还考虑无控制器时的情况，设 F 中的所有流以随机顺序产生并依次计算，且每条流计算完成后即保持转发路径配置不变（类似于文献[6]中的在线模型）。与有控制器时的优化任务仅为 $\min\theta$ 不同，无控制器时每条流以各节点自主控制的方式计算完成后直接影响后续计算中的链路信息，因此还对每条 $size_f>0$ 的 f 添加优先级比 $\min\ \theta$ 低的优化任务 $\min\dfrac{\sum_{p\in\boldsymbol{P}_f}(cost_p+sld_p)x_p^f}{size_f}$，通常这样可使 F 中的所有流在相同顺序下完成计算后得到的结果是唯一的。结果如图 6 所示。

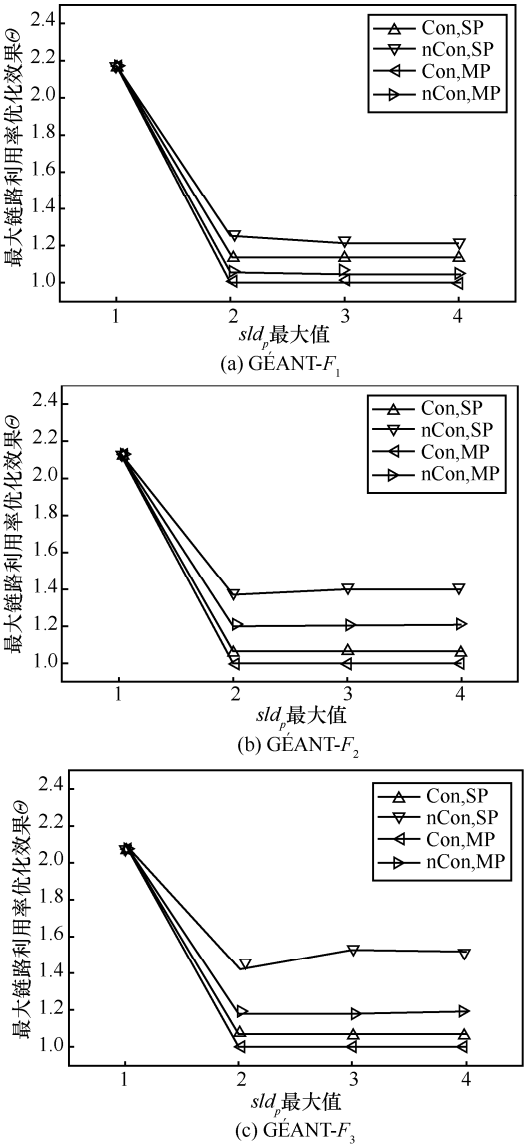


图 6　GÉANT 流量调度优化效果–Θ

从图 6 中可以总结如下规律。

1) 有控制器时，流量调度是全局优化，可以取得相对理想的效果，所有流转发支持多路径时 Θ 为 1，意味着求得的 θ 能达到最优值；所有流转发只支持单路径时 Θ 为 1.06~1.14，意味着求得的 θ 接近最优值；无控制器时，流量调度是局部优化，在候选路径集合相同的情况下，有控制器时的全局优化比局部优化的效果更好；即使是局部优化，效果也比所有流沿最短路径转发时更好，支持多路径时求得的 θ 相比所有流沿最短路径转发时的值降低 42%以上，只支持单路径时也可降低 27%以上。

2) 有控制器时，sld_p 最大值分别为 2、3、4 时的优化效果相同；无控制器时，sld_p 最大值为 2、3、4 时的效果可能存在一定波动，不一定 sld_p 最大值越大效果越好，这也是局部优化导致的。

3) 无论有无控制器，一般来说所有流转发支持多路径时求得的 θ 比只支持单路径时略小（也可能在有控制器时相同，如表 2 所示），这是因为单路径的约束更严格，多路径能更合理地均衡负载。

5.2.2　Germany50 拓扑流量调度优化效果

为验证本方法在不同规模拓扑下以及基于 $\boldsymbol{p}$ 的属性做合适筛选得出 $\boldsymbol{P}_f$ 的优化效果，先从 Germany50 的 TM 数据集中取出 3 个，然后分别取每个 TM 中源—目的节点间流量需求大小在本 TM 最大前 200 的流量数据，作为用于评估的 3 个 TM；设 TM 中的每对源—目的节点间有一条流 f，$size_f$ 为它们之间的流量需求大小；这 3 个 F 分别用 Germany50-F_1、Germany50-F_2、Germany50-F_3 表示，各有 200 条流。

基于 5.1 节中计算得出的 SP 和 MP 这两种情况下 $sld_p\leqslant 3$ 的 $\boldsymbol{P}_{st}$，对每条流 f 筛选其对应的 $\boldsymbol{P}_{st}$，仍将 sld_p 最大值设定为 3，同样区分 SP 和 MP 这两种情况，分别得出 $cost_p$ 相对其对应的源—目的节点之间最短路径的 $cost_p$ 的比值在（以下简称为 $cost_p$ 相对最短路径的比值）小于 1.25、1.5、1.75 的 $\boldsymbol{P}_f$。结果如表 2 所示。

从表 2 中可以总结如下规律。

1) 无论所有流转发是支持多路径还是只支持单路径，以及有无控制器，在 Θ 取得相同条件下无 $cost_p$ 约束时的值以前，候选路径数量对优化效果影响较大，$cost_p$ 相对最短路径的比值上限增加意味着候选路径数量增多，优化效果通常也会更好。

表 2　Germany50 流量调度优化效果–Θ

F	条件	$cost_p$ 相对最短路径的比值		
		<1.25	<1.50	<1.75
Germany50-F_1	sld_p≤3, Con, SP	1.20	1.07	1.00
Germany50-F_1	sld_p≤3, nCon, SP	1.36	1.30	1.23
Germany50-F_1	sld_p≤3, Con, MP	1.20	1.07	1.00
Germany50-F_1	sld_p≤3, nCon, MP	1.26	1.23	1.05
Germany50-F_2	sld_p≤3, Con, SP	1.20	1.07	1.00
Germany50-F_2	sld_p≤3, nCon, SP	1.38	1.30	1.22
Germany50-F_2	sld_p≤3, Con, MP	1.20	1.07	1.00
Germany50-F_2	sld_p≤3, nCon, MP	1.30	1.19	1.09
Germany50-F_3	sld_p≤3, Con, SP	1.20	1.08	1.00
Germany50-F_3	sld_p≤3, nCon, SP	1.37	1.31	1.12
Germany50-F_3	sld_p≤3, Con, MP	1.20	1.08	1.00
Germany50-F_3	sld_p≤3, nCon, MP	1.27	1.25	1.06

2) 有控制器时，流量调度是全局优化，无论所有流转发是支持多路径还是只支持单路径，当 $cost_p$ 相对最短路径的比值小于 1.75 时 Θ 为 1，意味着求得的 θ 都能达到最优值，因此只要基于 $cost_p$ 做合适筛选，既可减小 $K(\boldsymbol{P}_f)$，也能获得较好的优化效果。

5.2.3　Germany50 拓扑流量调度求解时间

考虑 5.2.2 节有控制器时的情况，与 3.2 节中约束条件为式(1)~式(7)即无 sld_p、$cost_p$ 约束条件的 SDN 流量调度模型对比；均区分所有流转发支持多路径和只支持单路径，即区分 LP 模型和 ILP 模型，统计其中的变量数量如图 7 所示。

图 7 表明，$cost_p$ 相对最短路径的比值上限越大，变量数量越多；$cost_p$ 相对最短路径的比值上限相同时，所有流转发支持多路径时的变量数量比只支持单路径时的变量数量更多；对 $\boldsymbol{p}$ 进行筛选后，变量数量相对于 SDN 流量调度模型更少。

再将 θ 设定为当前条件优化计算求得的 θ，不添加其他优化目标，此时模型有解，以代入 θ 求解一次所需时间作为求解时间；硬件为 i7-6700 处理器和 12 GB 内存，软件为 Windows 10 操作系统，在 Matlab R2017a 下用 Gurobi[26] 7.5.1 求解并统计时间，用同一软硬件环境下的求解时间衡量求解难度。结果如图 8 所示。

图 7 和图 8 表明，对于 LP 和 ILP 模型，通常变量数量越少求解越快。$cost_p$ 相对最短路径的比值小于 1.75 时，根据 5.2.2 节的结果和统计求解时间的前提，此时本方法与 SDN 流量调度模型设定的 θ 相同，对于 LP 模型，本方法的求解时间比 SDN 流量调度模型少 83%以上；对于 ILP 模型，本方法的求解时间比 SDN 流量调度模型少 23%以上；一般而言，相同情况下 ILP 模型的求解时间相对 LP 模型更长。上述结果说明本方法在约束条件更多的情况下能简化 MCF 模型，通过对 $\boldsymbol{p}$ 进行筛选可降低后续求解的难度，使求解时间更短，且求得的 θ 可达到最优值，能较好平衡网络性能与求解时间。无控制器时，参见 4.2 节，每条流 f 求解时的变量数量只取决于 $K(\boldsymbol{P}_f)$，比有控制器的全局优化时的变量数量更少，根据上述结果，每条流的求解时间会更短，更适合性能相对较弱的网络节点。

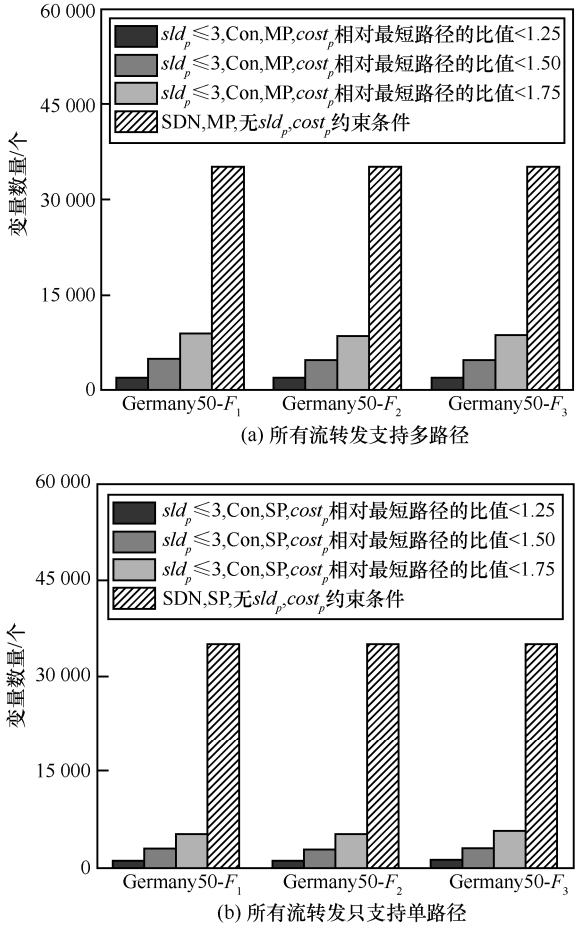
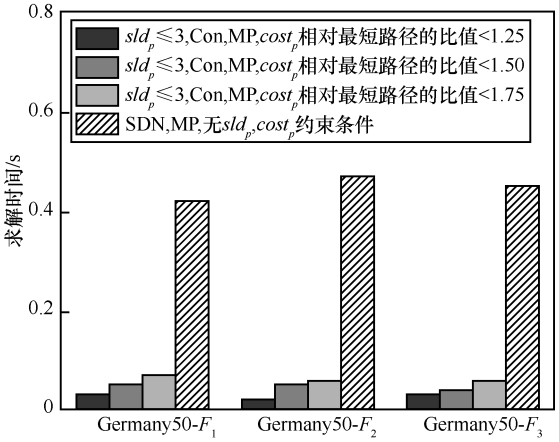


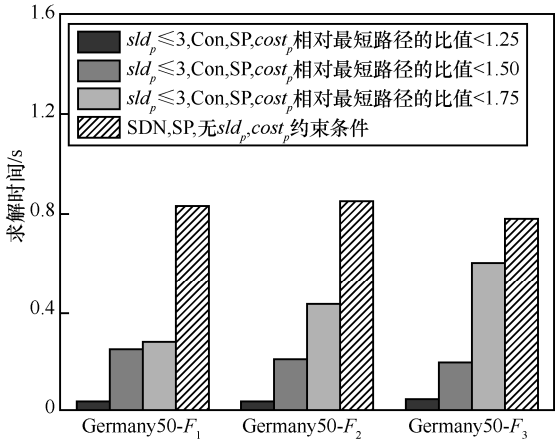
图 7　5.2.2 节有控制器时流量调度模型中的变量数量

5.3　节能控制优化效果评估

节能控制优化任务为 min *SUM*，优化效果用当前条件优化计算求得的 *SUM* 评估且越小越好；仍使用 GÉANT-F_1、GÉANT-F_2、GÉANT-F_3 并将 θ 设定为典型值 0.5[25]，为减轻计算负担由各节点对流进行自主控制；以所有流沿最短路径转发的情况作为对比，表示为 sld_p=1。结果如表 3 所示。



(a) 所有流转发支持多路径



(b) 所有流转发只支持单路径

图 8　5.2.2 节有控制器时流量调度求解时间

表 3　　GÉANT 节能控制优化效果-*SUM*

条件	GÉANT-F_1	GÉANT-F_1	GÉANT-F_1
sld_p=1	36	36	36
sld_p≤2, nCon, SP	28	27	28
sld_p≤2, nCon, MP	28	27	28
sld_p≤3, nCon, SP	24	22	22
sld_p≤3, nCon, MP	24	22	22
sld_p≤4, nCon, SP	22	21	21
sld_p≤4, nCon, MP	22	21	21

从表 3 中可以总结出如下规律。

1) sld_p 最大值越大，求得的 *SUM* 越小，这是由于 sld_p 最大值决定了候选路径的数量，候选路径越多则越有可能让所有流使用更少的链路转发。

2) 当 sld_p 最大值相同时，所有流转发支持多路径与只支持单路径求得的 *SUM* 相同，这是因为当设定的 θ 较大也就是链路可用容量足够大时，节能控制优化目标与流量负载分担因素关联不大。

5.4　评估结果讨论

首先，对 GÉANT 和 Germany50 拓扑的分析表明，根据 4.1 节中的原则 1)~原则 3)计算 $\boldsymbol{P}_{st}$ 能使 $K(\boldsymbol{P}_{st})$ 的大小控制在相对有限的范围。

然后，若将优化任务设为最小化最大链路利用率 θ，基于 GÉANT 和 Germany50 拓扑的评估结果表明，选择合适的候选路径集合后，本方法的优化效果良好，有控制器且所有流转发支持多路径时 Θ 为 1 即求得的 θ 达到最优值，有控制器且所有流转发只支持单路径时 Θ 为 1~1.14 即达到或接近最优值，无控制器即各节点自主控制时也有较好效果。

另外，基于 Germany50 拓扑的评估结果还表明，考虑有控制器时的情况，当 $cost_p$ 相对最短路径的比值小于 1.75 时不仅 Θ 为 1 即求得的 θ 达到最优值，所有流转发支持多路径时求解时间也比约束条件更少的 SDN 流量调度模型少 83%以上，只支持单路径时求解时间比 SDN 流量调度模型少 23%以上，说明当网络拓扑规模较大时，基于 $\boldsymbol{p}$ 的属性做合适筛选得出 $\boldsymbol{P}_f$，能在获得较好优化效果的同时减少流量调度模型中的变量数量，减轻计算负担。

最后，基于 GÉANT 拓扑的评估结果表明，本方法能较好满足节能控制需求，候选路径越多则节能效果越有可能更好，θ 设定为 0.5 时求得的 *SUM* 可达到所有流沿最短路径转发时的 60%左右。

6　结束语

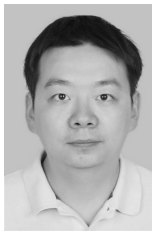
本文针对 SDN 流量调度的 MCF 问题，将 SR 引入 SDN，设计整体架构；通过建模分析，结合 SR 的特点设计算法，预先计算所有源—目的节点间的候选路径集合和相应路径的属性，再结合流的需求和约束条件根据路径的属性筛选得出流的候选路径集合，不仅简化了 SDN 中的 MCF 模型，降低了后续求解的难度，还支持控制器集中控制和各节点自主控制的工作方式，也缓解了控制器的可扩展性问题；讨论如何满足网络节能需求，减少可参与流转发的链路数量。评估结果表明，SDN 中基于 SR 的流量调度方法能满足流的各种需求和约束条件，提高网络性能，减轻求解流量调度问题的计算负担。

参考文献：

[1] WANG N, HO K, PAVLOU G, et al. An overview of routing optimization for internet traffic engineering[J]. IEEE Communications Surveys & Tutorials, 2008, 10(1): 36-56.

[2] FILSFILS C, NAINAR N K, PIGNATARO C, et al. The segment routing architecture[C]//IEEE Global Communications Conference. 2015: 1-6.
[3] KREUTZ D, RAMOS F M V, VERISSIMO P E, et al. Software-defined networking: a comprehensive survey[J]. Proceedings of the IEEE, 2015, 103(1): 14-76.
[4] HARTERT R, VISSICCHIO S, SCHAUS P, et al. A declarative and expressive approach to control forwarding paths in carrier-grade networks[J]. ACM SIGCOMM Computer Communication Review, 2015, 45(4): 15-28.
[5] MORENO E, BEGHELLI A, CUGINI F. Traffic engineering in segment routing networks[J]. Computer Networks, 2017, 114: 23-31.
[6] BHATIA R, HAO F, KODIALAM M, et al. Optimized network traffic engineering using segment routing[C]//IEEE International Conference on Computer Communications. 2015: 657-665.
[7] HARTERT R, SCHAUS P, VISSICCHIO S, et al. Solving segment routing problems with hybrid constraint programming techniques[C]//International Conference on Principles and Practice of Constraint Programming. 2015: 592-608.
[8] SCHÜLLER T, ASCHENBRUCK N, CHIMANI M, et al. Traffic engineering using segment routing and considering requirements of a carrier IP network[C]//IFIP Networking Conference and Workshops. 2017: 1-9.
[9] GIORGETTI A, CASTOLDI P, CUGINI F, et al. Path encoding in segment routing[C]//IEEE Global Communications Conference. 2015: 1-6.
[10] LI S, HU D, FANG W, et al. Source routing with protocol-oblivious forwarding (POF) to enable efficient e-health data transfers[C]//IEEE International Conference on Communications. 2016: 1-6.
[11] DONG X, GUO Z, ZHOU X, et al. AJSR: an efficient multiple jumps forwarding scheme in software-defined WAN[J]. IEEE Access, 2017, 5: 3139-3148.
[12] FILSFILS C, MICHIELSEN K, TALAULIKAR K. Segment routing, part I[M]. North Charleston: CreateSpace Independent Publishing Platform, 2017.
[13] 周桐庆, 蔡志平, 夏竟, 等. 基于软件定义网络的流量工程[J]. 软件学报, 2016, 27(2): 394-417.
ZHOU T Q, CAI Z P, XIA J, et al. Traffic engineering for software defined networks[J]. Journal of Software, 2016, 27(2): 394-417.
[14] CIANFRANI A, LISTANTI M, POLVERINI M. Incremental deployment of segment routing into an ISP network: a traffic engineering perspective[J]. IEEE/ACM Transactions on Networking, 2017, 25(5): 3146-3160.
[15] GUEDREZ R, DUGEON O, LAHOUD S, et al. Label encoding algorithm for MPLS segment routing[C]//IEEE International Symposium on Network Computing and Applications. 2016: 113-117.
[16] CIANFRANI A, LISTANTI M, POLVERINI M. Translating traffic engineering outcome into segment routing paths: the encoding problem[C]//IEEE Conference on Computer Communications Workshops. 2016: 245-250.
[17] LEE K, TOGUYENI A, NOCE A, et al. Comparison of multipath algorithms for load balancing in a MPLS network[C]//International Conference on Information Networking. 2005: 463-470.
[18] LEE K, TOGUYENI A, RAHMANI A. Hybrid multipath routing algorithms for load balancing in MPLS based IP network[C]//IEEE International Conference on Advanced Information Networking and Applications. 2006.
[19] SUCHARA M, XU D, DOVERSPIKE R, et al. Network architecture for joint failure recovery and traffic engineering[C]//ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems. 2011: 97-108.
[20] LECONTE M, DESTOUNIS M, PASCHOS G. Traffic engineering with precomputed pathbooks[C]//IEEE International Conference on Computer Communications. 2018.
[21] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, et al. OpenFlow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69-74.
[22] HIGHAM D J, HIGHAM N J. MATLAB guide[M]. Philadelphia: Society for Industrial and Applied Mathematics, 2016.
[23] BRANKE J, DEB K., MIETTINEN K, et al. Multiobjective optimization: interactive and evolutionary approaches[M]. Berlin: Springer Science & Business Media, 2008.
[24] ZHANG J, YU F R, WANG S, et al. Load balancing in data center networks: a survey[J]. IEEE Communications Surveys & Tutorials, 2018, 20(3): 2324-2352.
[25] ZHANG M, YI C, LIU B, et al. GreenTE: power-aware traffic engineering[C]//The 18th IEEE International Conference on Network Protocols. 2010: 21-30.
[26] GUROBI OPTIMIZATION, LLC. Gurobi optimizer reference manual [M]. Beaverton: Gurobi Optimization, 2018.
[27] ORLOWSKI S, WESSÄLY R, PIÓRO M, et al. SNDlib 1.0 - survivable network design library[J]. Networks, 2010, 55(3): 276-286.

[作者简介]



董谦（1986−），男，湖北咸宁人，中国科学院计算机网络信息中心博士生，佛山科学技术学院讲师，主要研究方向为未来互联网、软件定义网络、流量工程等。

李俊（1968−），男，安徽桐城人，博士， 中国科学院计算机网络信息中心研究员、总工程师、博士生导师，主要研究方向为未来互联网、网络安全等。

马宇翔（1991−），男，河南开封人，中国科学院计算机网络信息中心博士生，主要研究方向为网络体系结构、网络安全等。

韩淑君（1986−），女，山东高唐人，中国科学院计算机网络信息中心博士生，主要研究方向为网络体系结构、网络功能虚拟化等。