基于流量工程的软件定义网络控制资源优化机制

胡宇翔[①] 李子勇*[①] 胡宗魁[②] 胡 涛[①]

[①](国家数字交换系统工程技术研究中心 郑州 450002)

[②](中国人民解放军第91445部队 大连 116043)

摘 要：针对软件定义网络(SDN)分布式控制平面中由于网络分域管理所引发的控制扩张问题，该文提出了一种基于流量工程的SDN控制资源优化(TERO)机制。首先基于数据流的路径特征对流请求的控制资源消耗进行分析，指出通过调整控制器和交换机的关联关系可以降低控制资源消耗。然后将控制器关联过程分为两个阶段：先设计了最小集合覆盖算法来快速求解大规模网络中控制器关联问题；在此基础上，引入联合博弈策略来优化控制器和交换机的关联关系以减少控制资源消耗和控制流量开销。仿真结果表明，与现有的控制器和交换机就近关联机制相比，该文机制能在保证较低控制流量开销的前提下，节省约28%的控制资源消耗。

关键词：软件定义网络；资源优化；控制器关联；分布式控制平面

中图分类号：TN919.2; TP393.2 **文献标识码**：A **文章编号**：1009-5896(2020)03-0661-08

DOI: 10.11999/JEIT190276

Control Resource Optimization Mechanism of SDN Based on Traffic Engineering

HU Yuxiang[①] LI Ziyong[①] HU Zongkui[②] HU Tao[①]

[①](*National Digital Switching System Engineering & Technology Research Center*, *Zhengzhou* 450002, *China*)

[②](*Unit* 91445 *of PLA*, *Dalian* 116043, *China*)

Abstract: In Software-Defined Networking (SDN) with distributed control plane, network expansion problems arise due to network domain management. To address this issue, a Traffic Engineering-based control Resource Optimization (TERO) mechanism of SDN is proposed. It analyzes the control resource consumption of flow requests processing with different path characteristics, and points out that the control resource consumption can be reduced by changing the association relationship between controllers and switches. The controller association mechanism is divided into two phases: firstly, a minimum set cover algorithm is designed to solve the controller association problem efficiently in large-scale network. Then, a coalitional game strategy is introduced to optimize the controller association relationship to reduce both control resource consumption and control traffic overhead. The simulation results demonstrate that while keeping control traffic overhead low, mechanism which in this paper can reduce control resource consumption by about 28% in comparison with the controller proximity mechanism.

Key words: Software-Defined Networking (SDN); Resource optimization; Controller association; Distributed control plane

收稿日期：2019-04-22；改回日期：2019-06-24；网络出版：2019-08-02

*通信作者： 李子勇 17629352940@163.com

基金项目：国家自然科学基金(61521003, 61872382)，国家重点研发计划(2017YFB0803204)，广东省重点领域研发计划(2018B010113001)

Foundation Items: The National Natural Science Foundation of China (61521003, 61872382), The National Key Research and Development Program of China (2017YFB0803204), The Research and Development Program in Key Areas of Guangdong Province (2018B010113001)

1 引言

传统的互联网因其封闭僵化的体系结构已经变得越来越复杂，使得网络的运营和维护变得异常困难。软件定义网络(Software-Defined Networking, SDN)[1]作为一种新型网络体系，通过解耦控制平面和数据平面，提供集中的网络管理以及开放的可编程接口，使得网络的自动化管理和控制能力获得了很大提升。随着网络规模的扩大和流量的激增，SDN单控制器已无法满足庞大的流量请求，且带来的控制开销过高，使之成为网络的瓶颈。因此，在大规模网络中，SDN常采用分布式控制平面[2]对网络进行分域管理，多个控制器在地理位置上是分散的，但在逻辑上采用集中控制，共享全局网络视图，相互协作实现网络的高效控制。SDN控制与转发分离的思想和高速匹配转发的数据平面使SDN较传统互联网有巨大优势，但也使SDN控制平面面临严重的资源短缺问题。如拥有100个交换机的网络其流到达速率峰值约可达10 M/s[3]，而当前主流的控制器ONOS(Open Network Operating System)，其吞吐量也尚未达到1 M/s的目标需求[4]。

由于网络流量具有空间分布不均和时间上动态变化[5]的特性，控制器与交换机的静态关联容易导致控制器负载不均衡，使网络的资源利用率较低。Wang等人[6]提出两段论算法，在双向匹配算法上引入联合博弈模型实现交换机与控制器的动态优化部署，但该算法本质上仍属于重映射方法，计算量较大。胡涛等人[7]针对控制器选择僵化和迁移冲突问题，设计了一种基于分布式策略的控制器负载均衡机制，它通过构建分布式迁移决策域来完成交换机迁移和控制器角色转换，但它会产生更多的控制器间通信开销。为了提高SDN网络的可扩展性，Cui等人[8]提出了一种基于响应时间的多个SDN控制器负载均衡策略，考虑控制器实时响应时间和控制器负载的变化特征，来解决多个过载控制器在SDN控制平面中的负载均衡问题。文献[9]提出一种考虑SDN控制节点故障的控制器部署和交换机迁移方法，以此来减少控制链路平均时延，改善网络的可扩展性。但该算法中以交换机的数目作为控制器负载的衡量指标，这在实际网络中是不合理的。Zhou等人[10]为了防止网络中控制器故障，将交换机迁移问题建模为EMD(Earth Mover's Distance)模型，并设计了一种高效的启发式算法来快速求解大规模网络中交换机迁移问题。文献[11]进行控制器动态关联时考虑数据流的源目的地址，设计了一种流特征感知的控制器关联决策机制，降低了控制器资源消耗，但可能会造成部分控制器负载失衡。

以上文献考虑控制器的负载都是以数据平面的流请求数量作为衡量标准。然而，当前的研究方法通常忽略了数据流的路径特征。在SDN分布式控制平面中，由于网络被分域管理，数据流在转发的过程中会经过不同的控制域，这就会造成原本由1个控制器独立完成的流请求处理工作，需要多个控制器共同参与，其所需的控制资源也会相应增加，在这里称之为“控制扩张问题”。本文针对网络分域管理带来的控制扩张问题，提出了一种基于流量工程的控制资源优化机制。具体而言，本文的主要贡献在于：(1)根据数据流的路径特征对不同流请求带来的控制资源消耗进行深入分析，指出通过改变控制器和交换机的关联关系可以降低控制资源消耗；(2)权衡控制资源消耗和控制流量开销，设计了一种控制器动态关联机制；(3)为实现控制器动态关联，设计了一种基于集合覆盖算法的快速求解算法，同时引入博弈策略，进一步改善了控制器和交换机关联关系。

2 研究动机

2.1 流请求处理流程

如图1所示，在SDN分布式控制平面中，网络被分域管理，每个控制器管理所属的子域，负责处理本地域内的网络事件，做出相应的决策。根据控制器与交换机的关联关系和数据流的路径特征，可以将数据流分为域内流(如f_1)和跨域流(如f_2)。SDN分布式控制平面对域内流和跨域流的处理流程不同，其所消耗的控制资源也不同。对于域内流，其流传输过程只涉及1个控制器，和单控制器对流请求的处理流程相同，如下所示：

(1) 当数据流到达交换机时，交换机根据存储在流表中的转发规则匹配处理，若没有匹配规则，交换机便会将流处理请求上传到本地控制器；

(2) 本地控制器根据数据流源目的地址生成流转发规则，然后利用OpenFlow协议将转发规则安装到路径涉及的交换机上。

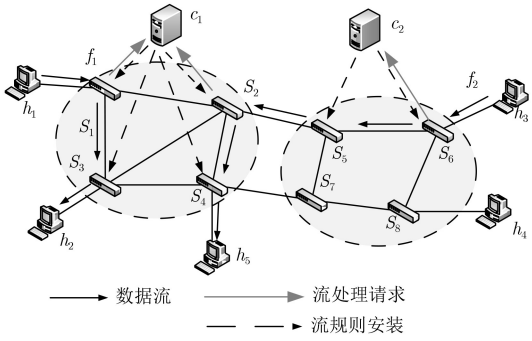

图 1 SDN分布式控制平面对流请求处理流程

对于跨域流，需要多个控制器相互协作，分布式控制平面的处理流程如下：

(1) 当本地控制器收到跨域流的请求，本地控制器根据路由策略，获得数据包应流向的下一个控制域；本地控制器计算并生成流向下一个控制域的流转发规则，并安装到相应的交换机上；

(2) 数据流传输到下一个控制域，会向所在的控制器再次上传流处理请求，若目的地址位于本域，则控制器会像处理域内流一样，计算、生成并安装转发规则；否则继续重复(1)，直到数据流到达目的地址。

2.2 降低SDN控制资源消耗的思路

根据以上分析，跨域流在到达下一个控制域时，会向控制器再次上传流处理请求，造成了不必要的计算开销。SDN控制器资源消耗与数据流经过的控制域的个数有关，据此可以通过减少数据流经过的控制域的个数来减少控制平面资源消耗。如图2，将交换机s_2, s_4从控制器c_1迁移到控制器c_2，跨域流f_2便变成了域内流，其涉及的控制器个数便减少了1，其控制资源消耗也会相应减少。因此，降低SDN控制资源消耗的思路便是：通过寻找一种控制器-交换机关联方案，使相应数据流经过的控制域的个数尽可能地少，使整个网络消耗控制资源较多的跨域通信的场景减少，从而降低整个网络的控制资源消耗。

3　模型构建

3.1 网络建模

给定一个SDN网络，整个网络拓扑可用无向图$G=(V,E)$表示，其中，V表示网络中节点集合，E表示网络中链路的集合。SDN网络中由M个控制器和N个交换机组成，控制器集合为$C=\{c_1,c_2,\cdots,c_M\}$，控制器所在的管理域为$G=\{G_1,G_2,\cdots,G_M\}$，交换机集合为$S=\{s_1,s_2,\cdots,s_N\}$，假设每个控制器的处理容量相同为α，为了保证控制器能够应对突发流量而不至于过载，设定冗余因子η。控制器c_k与交换机s_i之间的距离为d_{ik}。f_{ij}为交换机s_i到交换机s_j的流请求速率。控制器与交换机的关联关系表示为矩阵$\boldsymbol{X}=[x_{mn}]_{M\times N}$，其中$x_{mn}=1$表示交换机$s_n$关联至控制器$c_m$，反之$x_{mn}=0$，在任意时刻，每个交换机仅可关联至一个控制器。定义$\boldsymbol{Y}=[y_n^{ij}]_{1\times N}$表示数据流路径经过的交换机分布，其中，$y_n^{ij}=1$表示数据流$f_{ij}$经过交换机$s_n$，反之$s_n=0$。定义$\boldsymbol{Z}=[z_m^{ij}]_{1\times M}$表示数据流$f_{ij}$是否经过控制域$G_m$，其中$z_m^{ij}=1$表示数据流经过控制域$G_m$，反之$z_m^{ij}=0$。对于数据流$f_{ij}$，已知其经过的交换机分布$\boldsymbol{Y}=[y_n^{ij}]_{1\times N}$和相应的控制器与交换机关联关系$\boldsymbol{X}=[x_{mn}]_{M\times N}$，便可得出其经过的控制域的分布$\boldsymbol{Z}=[z_m^{ij}]_{1\times M}$，如式(1)。控制器的负载主要来源于其子域下所有交换机上传的流处理请求，因此控制器c_m的负载θ_m如式(2)。

$$z_m^{ij}=\begin{cases}1, \sum\limits_{n=1}^{N} y_n^{ij}x_{mn}>0\\ 0, \sum\limits_{n=1}^{N} y_n^{ij}x_{mn}=0\end{cases} \tag{1}$$

$$\theta_m=\sum_{i=1}^{N}\sum_{j=1}^{N} Z_m^{ij}f_{ij},\forall m \tag{2}$$

定义1　数据流f_{ij}的控制路径长度γ_{ij}。定义数据流f_{ij}的控制路径长度为数据流经过的控制域的个数，如式(3)。对于域内流，$\gamma_{ij}=1$。

$$\gamma_{ij}=\sum_{m=1}^{M} z_m^{ij} \tag{3}$$

定义2　流处理请求消耗的控制资源L。根据以上分析，控制资源消耗与数据流的控制路径长度正相关。对于数据流f_{ij}，其流请求处理消耗的控制资源定义为流请求数量f_{ij}和其控制路径长度γ_{ij}的乘积。因此，为处理全部流请求，控制平面总的控制资源消耗L如式(4)。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{N} f_{ij}\gamma_{ij} \tag{4}$$

定义3　控制流量开销ϕ。控制流量包括交换机向控制器上传的流处理请求和控制器生成规则安装到相应交换机的流量。由于控制器和交换机之间采用带内通信的模式，控制流量自然要占用数据平面稀缺的带宽资源。d_{ik}为交换机s_i和控制器c_k通信的距离，这里以跳数来计算。因此，整个网络的控制流量开销为ϕ。如式(5)。

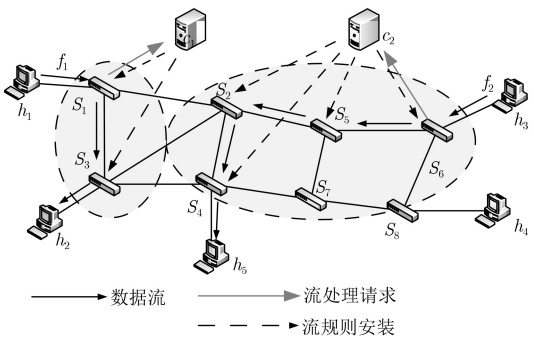

图 2 通过交换机迁移可以降低控制资源消耗

$$\phi = \sum_{k=1}^{M}\sum_{i=1}^{N} f_{ij} d_{ik} x_{ik} \tag{5}$$

3.2 基于流量工程的SDN控制资源优化问题

本文的优化目标为在确保控制器和交换机之间的控制流量开销处在合理范围的同时，最大限度地减少控制资源消耗。因此，基于流量工程的SDN控制资源优化(Traffic Engineering based Resource Optimization, TERO)问题的目标函数如式(6)，权值$\eta \in (0,1)$。

$$\min \sum_{i=1}^{N}\sum_{j=1}^{N} f_{ij}\gamma_{ij} + \eta \sum_{k=1}^{M}\sum_{i=1}^{N} f_{ij} d_{ik} x_{ik} \tag{6}$$

$$\text{s.t.}\ \ \theta_m \le \alpha \cdot \beta, \forall \text{m} \tag{7}$$

$$x_{ik} \in \{0,1\}, \forall i,k \tag{8}$$

$$\sum_{k=1}^{M} x_{ik} = 1, \forall k \tag{9}$$

$$d_{ik} \le \delta, \forall i,k \tag{10}$$

其中，约束式(7)表示所有控制器不得超载，约束式(8)表示x_{ik}取值范围为(0, 1)，约束式(9)表示每一个交换机只能与1个控制器关联，约束式(10)限制交换机和控制器之间的最大距离小于δ。

4 基于流量工程的SDN控制资源优化机制

为了实现TERO问题的快速求解，本节把该控制资源优化机制设计为2个阶段。其中，阶段1为最小集合覆盖算法(Minimum Set Coverage, MSC)，将控制器和交换机关联问题转化为集合覆盖模型，利用较少的控制器来尽可能地覆盖数据流路径上的交换机；阶段2为联合博弈策略(Coalitional Game, CG)，控制器之间通过协作交互，对控制器关联问题进行调整。

4.1 最小集合覆盖算法

阶段1把控制器和交换机之间的关联问题转化为集合覆盖模型，并提出最小集合覆盖算法来解决控制器和交换机的关联问题。在OpenFlow1.3中，每个交换机通过设主控制器、同等控制器、从控制器3个角色和多个控制器建立联系。如图3所示控制器和交换机备选关系拓扑，对于一条数据流f_{ij}，其经过的路径集合path $= \{s_1, s_2, s_5, s_7, s_9\}$。图4为经过最小集合覆盖算法后，所得控制器和交换机的关联关系$c_1 = \{s_1, s_2, s_3, s_4, s_5\}$, $c_2 = \{s_6\}$, $c_3 = \{s_7, s_8, s_9, s_{10}\}$。可以看出，通过最小集合覆盖算法，原本可能需要3个控制器完成的流处理请求，现在只需要两个控制器完成。

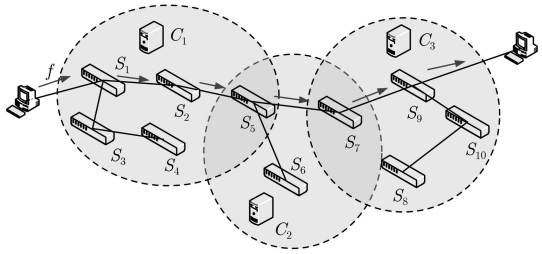

图 3 控制器-交换机备选关系拓扑

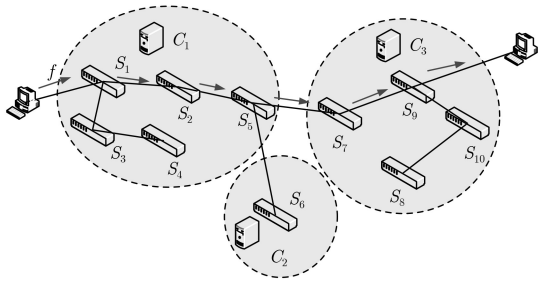

图 4 控制器-交换机关联后拓扑

对于集合覆盖问题，随着问题规模的扩大，其解空间呈指数增长，用一般的启发式策略难以求解，在求解NP难问题时，用完备策略不仅能降低问题求解的难度，又能保留问题的最优化解，所以它是求解NP难问题的重要策略。因此，本文建立完备策略来设计最小集合覆盖算法。下面描述最小集合覆盖问题以及对应的完备策略：

最小集合覆盖问题：S是一个集合，S_1, S_2,…, S_m分别是S的一个子集，且$\cup S_i = S$，求集合S的最小集合覆盖。

最小集合覆盖问题的完备策略：

完备策略1 如果S_1, S_2, …, S_m中的一个集合$S_i = S$，则选择S_i作为最优覆盖中的唯一的集合。

完备策略2 如果存在某个元素$x \in S$, x只属于S_1, S_2, …, S_m中的一个集合S_i，则选择S_i作为最优化中的一个集合。

完备策略3 如果存在$S_i \subseteq S_j$，则排出S_i。

完备策略4 设$Sn(x)$表示$x \in S_i$集合的序号组成的集合，如果$Sn(x) \subseteq Sn(y)$，则删除S中的y。

本文设计一种基于4个完备策略的最小集合覆盖算法来解决控制器-交换机关联问题。最小集合覆盖算法执行的过程如表1所示。首先初始化控制器-交换机映射关系SC=$\{\cdot\}$，已关联的交换机set_switches=$\{\cdot\}$；第(2)步，Flow_sort$(\cdot)$函数是统计所有端到端流量，计算出每个路径上经过的数据流的总量，并按降序排列；第(4)～(10)步是针对每条路径，循环搜索各个集合，看是否满足完备策略，如果不满足4个完备策略，就采用贪婪算法，将交换机优先关联到覆盖交换机最多的控制器上；最后直到所有路径上的所有交换机都已关联，算法终止。

表 1　最小集合覆盖算法执行过程

算法1 最小集合覆盖算法(Minimum Set Coverage)

输入：SDN网络拓扑邻接矩阵$\boldsymbol{G}=[a_{ij}]$；网络中流处理请求矩阵$\boldsymbol{F}=[f_{ij}]$；控制器所能关联的备选集合：$C_i=\{S_1,S_2,\cdots,S_i\}$；控制器的容量及冗余因子：$\alpha_m$, β

输出：控制器-交换机之间的映射关系：$\boldsymbol{X}=[x_{ij}]$

(1) 初始化：控制器-交换机关联关系SC={·}；已关联的交换机set_switches={·};

(2) 统计网络中端到端流量分布Flow_pair=Flow_sort(F)；

(3) **while** I in Flow_pair：遍历网络中流量

(4) Path_switch= Dijkstra(G, i)；计算端到端流量的路径

(5) **while** Path_switch: 循环4个完备策略

(6) **if** Path_switch $\subseteq C_i$：若满足完备策略1, SC[C_i]={Path_switch }；流经过的所有交换机关联到C_i

(7) **if** $S_i\in$Path_switch AND $S_i\in C_j$满足完备策略2, $S_i\rightarrow C_j$; S_i关联到C_j

(8) **if** 存在$C_i\subseteq C_j$：满足完备策略3，则$\cup S_i\rightarrow C_j$；交换机S_i优先关联到C_j

(9) **if** Sn(S_i)$\subseteq$Sn(S_j)：满足完备策略4$S_i\rightarrow\cup C_j$；交换机S_i优先处理

(10) **else** 如果上述4个完备策略都不能满足：实行贪婪算法switch = max(Path_switch & C_i)；寻找关联交换机较多的控制器SC[C_i]={switch}；将相应交换机关联到控制器C_i上

end if; end while;

(11) **end while;**

(12) SC={$C_j=\{S_j,S_{j+1},\cdots,S_n\}$；输出控制器-交换机映射关系

4.2　联合博弈策略

阶段1的最小集合覆盖算法本质上是一种贪婪算法，它在解决大规模网络问题提供了一种快速求解的算法。然而，仅采用最小集合覆盖算法会产生不平衡的控制器关联关系。如图5所示为阶段1得出的控制器-交换机关联关系，对于数据流f_1和f_2，跨域已经不可避免，最小集合覆盖算法得出的控制器-交换机关系会导致控制器负载不均衡，其中控制器c_1管理的交换机要比控制器c_2管理的交换机要多，而且交换机s_7和s_8到控制器c_1的距离要大于控制器c_2，这会导致控制流量开销太大。

为了更合理地平衡控制器和交换机之间的关联关系，阶段2引入联合博弈策略对控制器关联问题进行调整，在博弈过程中，每个控制器会计算域内每条数据流所产生的控制资源消耗和控制流量开销，为了最大限度地减少自身资源消耗，控制器会将域内的交换机迁移到其他控制器上，完成交换机迁移的条件有2个：

(1) 所有控制器c_m都满足$\theta_m\leq\alpha\cdot\beta$，即所有控制器均不过载；

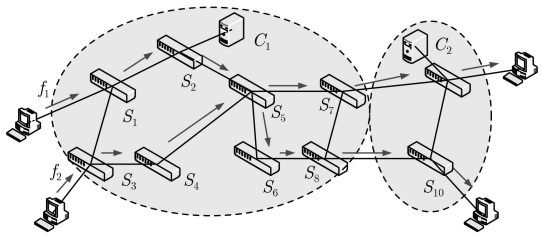

图 5 阶段1输出的控制器-交换机关系

(2) 对于数据流f_{ij}, $f_{ij}\gamma_{ij}+\delta f_{ij}d_{ik}x_{ik}\leq f_{ij}\gamma'_{ij}+\delta f_{ij}d'_{ik}x'_{ik}$，即交换机迁移后，相应的数据流总的资源消耗降低了。

联合博弈策略执行过程如表2所示。第(1)～(3)步每个控制器分别计算域内每条数据流产生的资源

表 2　联合博弈策略执行过程

算法2 联合博弈策略Coalitional Game

输入：算法1输出的控制器-交换机之间的关联关系$\boldsymbol{X}=[x_{ij}]$

输出： 控制器-交换机之间的关联关系$\boldsymbol{X}'=[x'_{ij}]$

(1) 初始化$\boldsymbol{X}=[x_{ij}]$, α_m, β

(2) **repeat**

(3) **for each** s_i **in** F：寻找可能存在的交换机迁移

(4) **Initial migration pair** $s_i: c_m\rightarrow C_n$；找到满足两个条件的交换机迁移对

end for

(5) **for each** c_m：对于每一个控制器

(6) $L_{ij}=f_{ij}d_{kl}x_{ik}x_{jl}+\delta c_{ik}$;计算每条数据流的资源消耗

(7) **if** $s_i: c_m\rightarrow C_n$ and $\theta_n\leq\alpha\times\beta$；保证控制器不过载，寻找可能的交换机迁移

(8) $L'_{ij}=f_{ij}d_{kl}x'_{ik}x'_{jl}+\delta c'_{ik}$；假设迁移，计算新的资源消耗

(9) **if** $L'_{ij}\leq L_{ij}$: 若交换机迁移前后，资源消耗减少了，则接受迁移

(10) $s_i\rightarrow c_j$；实施交换机迁移$L_{ij}=L'_{ij}$；更新的资源消耗

(11) **end if; end for**;

(12) 直到系统没有任何交换机要求迁移，则算法收敛

消耗；第(5), (6)步每个控制器会优先选择资源消耗较多的交换机进行迁移；第(7), (8)步比较迁移前后资源消耗情况，若迁移后，资源消耗有所减少，则其他控制器会接受迁移。该过程会不断迭代，直到所有控制器都不会发出迁移请求。

4.3 算法复杂度分析

本文所提TERO机制的算法复杂度主要来源于两个阶段，阶段1其计算复杂度主要来源：(1) 对集合的不断扫描，并判断是否符合完备策略，其计算复杂性为：$O(NM)+O(NM)+O(N^2M)+O(NM^2)$; (2) 贪婪算法的复杂度，其计算复杂性为$O(NM)$。阶段2的计算复杂性主要来源于控制器寻找必要的交换机迁移，搜索空间为$O(NM)$。因此，该资源优化算法的计算复杂性为$O(NM)+O(NM)+O(N^2M)+O(NM^2)+O(NM)+O(NM)$。随着网络规模的增加，该算法的求解空间几乎呈线性增长，避免了传统方法求解时组合爆炸的问题，保证了算法在有效时间内找到近似最优解。

5 仿真结果分析

5.1 仿真环境设置

本节对TERO机制的性能进行仿真验证，关于实验环境和参数设置说明如下：

仿真选取RYU[12]作为实验控制器，同时在Mininet[13]上进行拓扑测试。为了提高实验效率，将RYU控制器和Mininet以虚拟机的形式部署在2个不同的物理主机上，虚拟机为Ubuntu Server16.04，搭载Intel Core i7-3770 3.40 GHz处理器，8 GB内存。其中一个物理主机承载7个虚拟机，每个虚拟机均运行RYU控制器，另一个虚拟机运行Mininet。

仿真采用的拓扑取自Internet Topology Zoo[14]上的Interoute，其中包含110个节点，149条链路，Interoute网络具有地理覆盖范围广，高度分布的特征，其网络中的数据流传输路径较长，具有明显的跨域特征。在Interoute网络中SDN控制器数目设为7，其中每个控制器的处理容量相同设为1800 k flows/s[8]。这里假设采用跳数作为控制器和交换机之间距离的衡量标准来进行控制器的部署，控制器与交换机之间的最大距离设为6跳。实验中每个交换机连接3个主机。当实验启动时，为了模拟动态的网络流量，数据流在主机上的生成和消失服从泊松分布，交换机流请求率设为100～500 k flows/s，其中数据流路径采用dijkstra算法求得。实验结果借助MATLAB工具进行分析。

5.2 仿真结果与分析

为了验证TERO机制在SDN网络中的性能，在这里与控制器-交换机静态关联机制(Static Switch-Controller strategy, SSC)[15]、控制器-交换机就近关联机制(Nearest Migration Decision, NMD)[16]进行对比。(1) SSC：控制器部署按照交换机的距离进行聚类分析，并划分管理域，一旦控制器部署完成便不会改变控制器-交换机关联关系。(2) NMD：控制器动态关联机制，当控制器负载超过设定的阈值时，便启动交换机迁移，优先迁移到距离最近的控制器上。

5.2.1 实验1

本实验得出的3种机制的控制资源消耗和控制流量开销如图6和图7所示。从图6中可以看出，TERO控制资源消耗最低，NMD次之，SSC控制资源消耗最多。从图7可以看出，TERO和NMD控制流量开销均保持较低水平，而SSC带来了巨大的控制流量开销。这是因为SSC采用静态关联机制，不能根据流量的变化动态改变控制资源消耗；NMD采用就近控制器关联机制，在一定程度上能使经过数据流较多的交换机关联到同一个控制器上，起到了局部优化的效果，降低了控制资源消耗，同时能最大限度的缩短控制器和交换机之间的距离；而TERO机制对全网数据流路径进行统计，通过最小集合覆盖算法，数据流经过的控制域最小，控制资源消耗也就最少，同时也避免了数据流由于跨域带来的不必要的控制流量开销。根据实验结果求平均

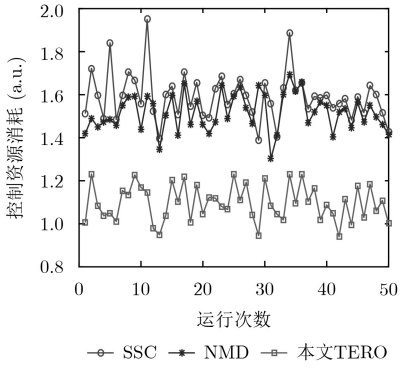

图 6 控制资源消耗对比

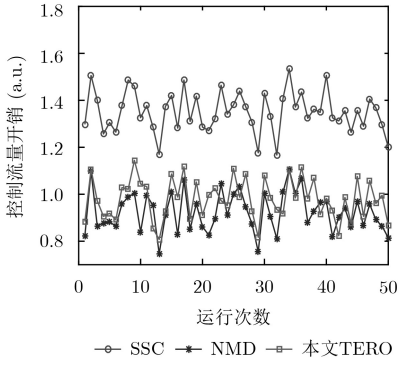

图 7 控制流量开销对比

值得，相比SSC和NMD，采用TERO机制来求得的控制器与交换机之间的关联关系平均能减少33%和28%控制资源消耗，平均能减少45%和8%的控制流量开销。

5.2.2 实验2

为了探究本文所提TERO机制阶段1和阶段2的关系，本实验把阶段1的最小集合覆盖算法(MSC)得出的实验结果和TERO机制进行对比。从图8和图9中可以看出，MSC算法得到的控制资源消耗和TERO基本相等，而MSC算法计算出的控制器-交换机关联关系产生了大量的控制流量开销，这是由于阶段1的MSC算法本质上是一种贪婪算法，尽可能地将更多的流分配给较少的控制器处理，而忽略了交换机与控制器之间的距离；TERO机制引入联合博弈策略，在保持控制资源消耗基本不变的前提下，调整控制器-交换机之间的关联关系，进而降低了控制流量开销。

5.2.3 实验3

为了说明本文所提TERO机制的普遍性，实验3从The Internet Topology Zoo给出的公共网络拓扑中选取4个规模差异较大的网络拓扑进行仿真实验，网络拓扑信息及相应参数设置如表3。

图10给出了3种机制在不同SDN网络中控制资源消耗情况。相比SSC和NMD，采用TERO机制分别平均可以减少约30%和10%的控制资源消耗。图11描述了在不同的SDN网络拓扑下控制器负载均衡率，以各控制器经过归一化处理后的负载标准差作为控制器负载均衡率的指标。与SSC和NMD相比，TERO机制得到的控制器负载均衡率均较低，因为SSC是以交换机的数目为衡量标准进行静态关联的，在数据流分布均匀的网络中，有较好的负载均衡的效果；NMD在控制器不过载的情况下，能根据节点的距离进行就近关联，具有一定的负载均衡能力；而TERO机制倾向于将更多的数据流路径上的交换机关联到较少的控制器上，以降低数据流跨域带来的控制资源消耗，但是这会造成严重的控制平面负载失衡。

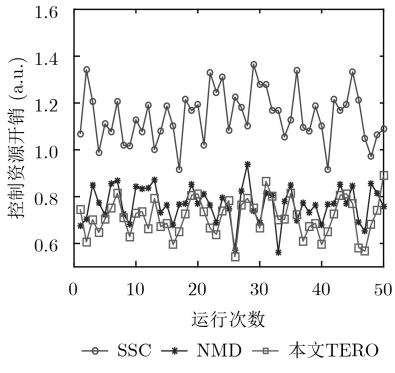

图 8 MSC算法和TERO控制资源消耗

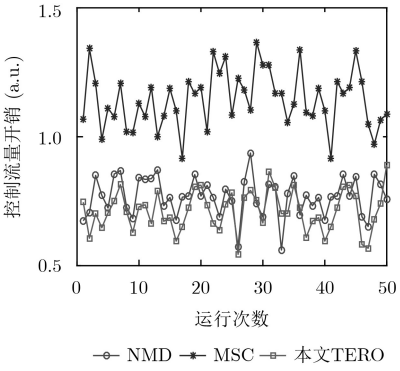

图 9 MSC算法和TERO控制流量开销

表 3　实验拓扑数据

网络拓扑	节点数	链路数	控制器数	距离阈值
ARNES	34	47	4	3
ChinaNet	42	66	5	4
Interllifiber	73	93	6	5
Interoute	110	149	7	6

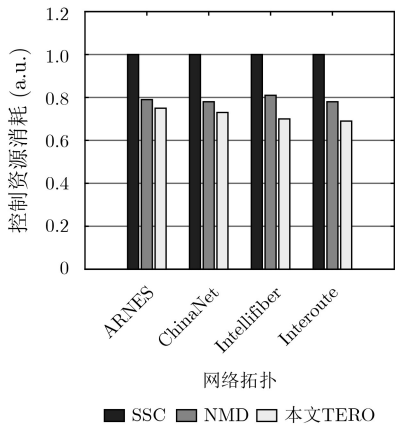

图 10 不同网络拓扑控制资源消耗

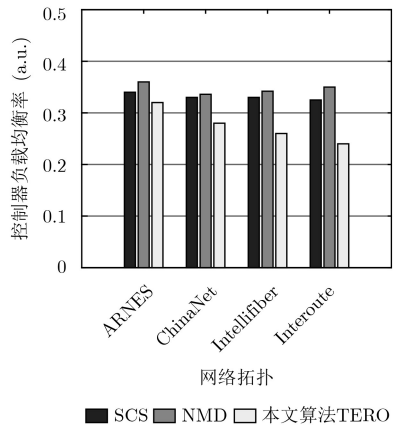

图 11 不同网络拓扑控制器负载均衡率

6 结束语

本文针对软件定义网络分布式控制平面中由于网络分域管理所引发的控制扩张问题，提出了基于流量工程的SDN控制资源优化机制。首先分析了SDN控制器对数据流请求的处理过程，指出改变控制器和交换机的关联关系可以降低控制资源消耗；然后权衡控制资源消耗和控制流量开销，将该问题建模为组合优化问题。最后，设计了最小集合覆盖算法，将同一路径的交换机关联到较少的控制器；在此基础上，引入联合博弈策略，来寻求较优的控制器-交换机关联方案。与控制器-交换机就近关联机制相比，本文所提基于流量工程的SDN控制资源优化机制可以减少约28%的控制资源消耗和8%控制流量开销。

参 考 文 献

[1] ZHANG Yuan, CUI Lin, WANG Wei, *et al*. A survey on software defined networking with multiple controllers[J]. *Journal of Network and Computer Applications*, 2018, 103: 101–118. doi: 10.1016/j.jnca.2017.11.015.

[2] KARAKUS M and DURRESI A. A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN)[J]. *Computer Networks*, 2016, 112: 279–293. doi: 10.1016/j.comnet.2016.11.017.

[3] XU Yang, CELLO M, WANG I C, *et al*. Dynamic switch migration in distributed software-defined networks to achieve controller load balance[J]. *IEEE Journal on Selected Areas in Communications*, 2019, 37(3): 515–529. doi: 10.1109/JSAC.2019.2894237.

[4] MUQADDAS A S, GIACCONE P, BIANCO A, *et al*. Inter-controller traffic to support consistency in ONOS clusters[J]. *IEEE Transactions on Network and Service Management*, 2017, 14(4): 1018–1031. doi: 10.1109/TNSM.2017.2723477.

[5] BENSON T, AKELLA A, and MALTZ D A. Network traffic characteristics of data centers in the wild[C]. The 10th ACM SIGCOMM Conference on Internet Measurement, Melbourne, Australia, 2010: 267–280.

[6] WANG Tao, LIU Fangming, GUO Jian, *et al*. Dynamic SDN controller assignment in data center networks: Stable matching with transfers[C]. The 35th Annual IEEE International Conference on Computer Communications, San Francisco, USA, 2016: 1–9.

[7] 胡涛, 张建辉, 邬江, 等. SDN中基于分布式决策的控制器负载均衡机制[J]. 电子学报, 2018, 46(10): 2316–2324. doi: 10.3969/j.issn.0372-2112.2018.10.002.

[8] CUI Jie, LU Qianzhe, ZHONG Hong, *et al*. A load-balancing mechnism for distributed SDN control using response time[J]. *Thansactions on Network and Service Management*, 2018, 15(4): 1197–1206. doi: 10.1109/TNSM.2018.2876369.

[9] 伊鹏, 刘邦舟, 王文博, 等. 一种考虑软件定义网络控制节点故障的控制器部署和交换机迁移方法[J]. 电子与信息学报, 2017, 39(8): 1972–1978. doi: 10.11999/JEIT161216.
YI Peng, LIU Bangzhou, WANG Wenbo, *et al*. Controller placement and switch immigration strategy for SDN controller failure[J]. *Journal of Electronics & Information Technology*, 2017, 39(8): 1972–1978. doi: 10.11999/JEIT161216.

[10] ZHOU Yang, ZHENG Kangfeng, NI Wei, *et al*. Elastic switch migration for control plane load balancing in SDN[J]. *IEEE Access*, 2018(6): 3909–3919. doi: 10.1109/ACCESS.2018.2795576.

[11] 张少军, 兰巨龙, 江逸茗, 等. 流特征感知的软件定义网络控制器动态关联机制[J]. 电子与信息学报, 2018, 40(9): 2050–2056. doi: 10.11999/JEIT171149.
ZHANG Shaojun, LAN Julong, JIANG Yiming, *et al*. Flow characteristics aware dynamic controller assignment in software-defined networking[J]. *Journal of Electronics & Information Technology*, 2018, 40(9): 2050–2056. doi: 10.11999/JEIT171149.

[12] SALMAN O, ELHAJJ I H, KAYSSI A, *et al*. SDN controllers: A comparative study[C]. The 18th Mediterranean Electrotechnical Conference, Lemesos, Cyprus, 2016: 1–6.

[13] PAL C, VEENA S, RUSTAGI R P, *et al*. Implementation of simplified custom topology framework in Mininet[C]. 2014 Asia-Pacific Conference on Computer Aided System Engineering, South Kuta, Indonesia, 2014: 48–53. doi: 10.1109/APCASE.2014.6924470.

[14] KNIGHT S, NGUYEN H X, FALKNER N, *et al*. The internet topology zoo[J]. *IEEE Journal on Selected Areas in Communications*, 2011, 29(9): 1765–1775. doi: 10.1109/jsac.2011.111002.

[15] LIAO Jianxin, SUN Haifeng, WANG Jingyu, *et al*. Density cluster based approach for controller placement problem in large-scale software defined networkings[J]. *Computer Networks*, 2017, 112: 24–35. doi: 10.1016/j.comnet.2016.10.014.

[16] YAO Guang, BI Jun, LI Yuliang, *et al*. On the capacitated controller placement problem in software defined networks[J]. *IEEE Communications Letters*, 2014, 18(8): 1339–1342. doi: 10.1109/LCOMM.2014.2332341.

胡宇翔：男，1982年生，副研究员，研究方向为宽带信息网、软件定义网络.

李子勇：男，1995年生，硕士生，研究方向为软件定义网络、计算机网络.

胡宗魁：男，1986年生，助理研究员，研究方向为控制自动化.

胡 涛：男，1993年生，博士生，研究方向为软件定义网络、网络空间安全.