doi:10.11959/j.issn.1000-436x.2017168

多控制器条件下区分 QoS 的虚拟 SDN 映射方法

赵志远，孟相如，苏玉泽，李振涛
（空军工程大学信息与导航学院，陕西 西安 710077）

摘 要：在多控制器 SDN 虚拟化环境中，以最小化控制通路平均时延、控制器负载失衡度和映射开销为目标，分别建立多控制器部署问题和虚拟 SDN 映射问题的数学模型，提出一种多控制器条件下区分 QoS 的虚拟 SDN 映射方法。仿真结果表明该方法能够满足不同用户对虚拟 SDN 服务质量的需求，保持多控制器的负载均衡，提高了映射成功率和收益开销比。

关键词：虚拟 SDN 映射；多控制器部署；服务质量；控制器自适应调整
中图分类号：TP393 **文献标识码**：A

Virtual SDN embedding with differentiated QoS under multiple controller

ZHAO Zhi-yuan, MENG Xiang-ru, SU Yu-ze, LI Zhen-tao
(College of Information and Navigation, Air Force Engineering University, Xi'an 710077, China)

Abstract: In SDN virtualization environment under multiple controllers, with the goal of minimum the mean delay between control paths, the load imbalance of controllers, and the cost of embedding, mathematical models of the controller placement and the virtual SDN embedding problem were built, and a virtual SDN embedding method with differentiated QoS under multiple controllers was proposed. The simulation results show that the proposed algorithm can satisfy various virtual networks QoS for users, keep load balance between controllers and increase the acceptance ratio and the revenue/cost ratio.

Key words: virtual SDN embedding, multiple controllers placement, QoS, controller adaptive adjustment

1 引言

网络虚拟化技术被视为下一代互联网的关键技术[1,2]，它通过资源抽象、隔离等机制，支持多个虚拟网络并行且互不干扰的运行在共享的底层物理网络上。基于透明代理的 SDN（software defined network）虚拟化平台[3~5]是当前 SDN 虚拟化的主要实现方式，SDN 虚拟化平台的主要工作是负责将虚拟 SDN（vSDN, virtual SDN）标识、隔离和映射[6,7]，本文研究映射问题，其任务是在 vSDN 请求到达后，在满足资源、拓扑结构等约束的前提下，将 vSDN 的节点和链路映射到物理 SDN 的节点和路径上。

在 SDN 虚拟化环境中，控制器通常部署在与底层交换机相同的位置，交换机与控制器之间的时延将影响到控制逻辑能否有效部署到转发设备中，因此，vSDN 映射不仅包括传统的虚拟网络映射过程，还应进行控制器的优化部署以缩短交换机到控制器的时延，实现网络的快速响应[8,9]。在单控制器 SDN 架构中，单个控制器管理全部 vSDN 的映射和运行，控制器部署仅需考虑时延问题[10]。然而单控制器存在可靠性和可扩展性问题，因此从 OpenFlow 协议 1.2 版本开始引入了多控制器概念，支持由多个控制器以物理分散逻辑集中的方式协同管理整个 SDN，同时也产生了多控制器部署问题[11~13]。多控制器优化部署一方面

收稿日期：2016-12-23；**修回日期**：2017-04-12
基金项目：国家自然科学基金资助项目（No.61401499）
Foundation Item: The National Natural Science Foundation of China (No.61401499)

要考虑时延，另一方面，在 vSDN 映射过程中，受位置和资源约束等影响，不同 vSDN 请求的虚拟节点映射位置存在很大的随机性，使交换机负载出现较大差异，进而导致各控制器的管理负载出现较大差异，因此需要考虑控制器负载的动态调整，避免控制器过载对网络运行造成不利的影响。

在控制器部署和动态调整方面[10~13]，Heeler 等[12]通过定义控制器与交换机之间平均、最大传播时延指标，解决了在给定网络拓扑的情况下，需要多少控制器以及怎样部署控制器的问题。现有虚拟网络映射算法[14~16]主要针对传统网络虚拟化环境，在 SDN 虚拟网络映射方面[8,9,17,18]，Mehmet 等[8]在映射过程中同时考虑虚拟节点链路映射和控制器部署，提出了 vSDN 映射方法。Wang 等[18]针对虚拟 SDN 环境中的单链路失效问题，提出了一种基于最佳备份拓扑的生存 vSDN 映射算法。Mijumbi 等[18]同时考虑节点和链路的负载均衡，提出了一种基于实时网络状态的流迁移方法，以动态管理 SDN 虚拟化环境中的节点和链路资源。从当前研究进展来看，没有文献研究多控制器条件下 vSDN 映射问题。本文以最小化控制通路平均时延、控制器负载失衡度和映射开销为目标，建立多控制器部署与调整问题和虚拟 SDN 映射问题的数学模型，同时考虑不同用户对服务质量的需求，提出一种多控制器条件下区分 QoS 的虚拟 SDN 映射方法。该方法由基于免疫优化算法的多控制器优化部署方法、区分 QoS 的 vSDN 映射方法和阈值触发的控制器自适应调整算法 3 个部分组成，能够协调控制器部署、vSDN 映射和控制器负载调整的关系，满足不同用户对 vSDN 服务质量的需求，提高映射成功率和收益开销比。

2 问题描述与网络模型

2.1 问题描述

基于 OpenFlow 的多控制器 SDN 架构如图 1 所示，其主要包括转发层、控制层和应用层。交换机维护自身流表结构，根据流表处理和转发数据分组，控制器通过南向接口（OpenFlow 协议）管理和配置交换机的流表，负责转发层资源编排、网络拓扑维护、网络实时状态更新等，通过北向接口支持网络应用。

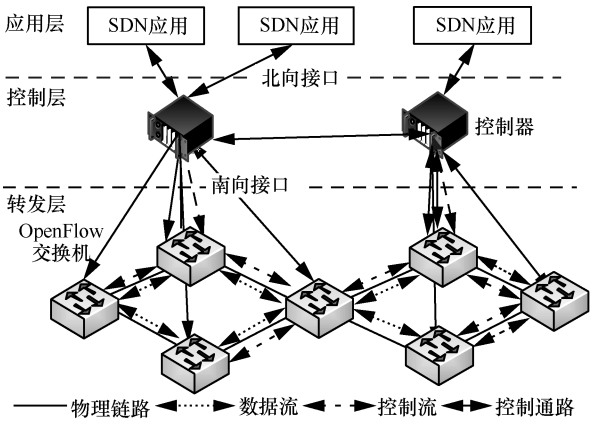


图 1 基于 OpenFlow 的多控制器 SDN 架构

以 FlowVisor 为例说明基于透明代理的 SDN 虚拟化平台及 vSDN 映射问题，如图 2 所示。转发层和控制层之间所有控制和状态消息通过中间的透明代理转发；多个 vSDN 共存于同一物理 SDN 网络上，享有独立的拓扑、带宽、交换机 CPU 和流表空间，并由单独的控制器管理和配置流表。图 2 中 $vSDN_1$ 映射到底层交换机 A、B、C 和物理链路 AB、BC（左侧黑色曲线）上，控制通路（左侧黑色虚线）为预先分配的物理路径。

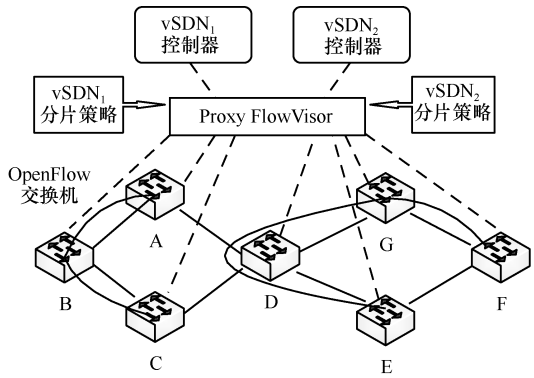


图 2 2 个 vSDN 切片共享物理 SDN

2.2 网络模型

1) 物理网络

多控制器物理 SDN 表示为带权无向图 $G_s=(N_s,N_c,L_s,L_c)$，其中，N_s、N_c、L_s、L_c 分别表示交换机节点、控制器节点、物理链路和控制通路集合。对 $n_s\in N_s$，属性包括节点 CPU 可用资源 $cpu(n_s)$、TCAM(ternary content addressable memory)可用资源 $tcam(n_s)$ 和节点位置 $loc(n_s)$，其中，TCAM 用于流表的存储和处理，$loc(n_s)=(x_s,y_s)$ 为二维坐标；对 $n_c\in N_c$，属性包括控制域 $CA(n_c)$ 和控制器管理负载 $Atcam(n_c)$，控制域由 n_c 管理的交换机节点

构成，$Atcam(n_c)=\sum_{n_s\in CA(n_c)} contcam(n_s)$，其中，$contcam(n_s)$为交换机节点占用TCAM；对于$l_s\in L_s$，属性为链路可用带宽$bw(l_s)$和传输时延$dl(l_s)$；对于$l_c\in L_c$，属性为控制通路时延$dl(l_c)$。

2) 区分 QoS 的 vSDN 请求

请求表示为带权无向图$G_{vSDN}=(N_v,L_v,QoS_v,T_v)$，其中，$N_v$、$L_v$分别表示虚拟节点和链路集合，对于$n_v\in N_v$，属性包括节点 CPU 资源需求$cpu(n_v)$、TCAM 资源需求$tcam(n_v)$、节点位置$loc(n_v)=(x_v,y_v)$和映射位置约束$D(n_v)$；对于$l_v\in L_v$，属性包括链路带宽需求$bw(l_v)$；$QoS_v$为服务质量需求，本文设定$QoS_2$的 vSDN 请求需满足控制通路时延约束$dl(l_c)$和虚拟链路时延约束$dl(l_v)$，$QoS_1$的 vSDN 请求无时延约束；$T_v$为 vSDN 生存时间。

3) 区分 QoS 的 vSDN 映射问题建模

区分 QoS 的 vSDN 映射定义为满足 vSDN 资源需求、位置约束、拓扑结构和 QoS 约束的从G_{vSDN}到G_s子集的映射关系，如式(1)所示。

$$M:G_{vSDN}\to(N_s^{sub},L_s^{sub},R_N,R_L) \tag{1}$$

其中，$N_s^{sub}\subseteq N_s$，$L_s^{sub}\subseteq L_s$，R_N和R_L是分配给 vSDN 的节点资源和链路资源。

4) 多控制器条件下区分 QoS 的 vSDN 映射问题

多控制器条件下区分 QoS 的 vSDN 映射问题可分解为多控制器部署（CP, controller placement）、区分 QoS 的 vSDN 映射与控制器自适应调整 3 个子问题。本文定义多控制器部署问题为给定物理网络信息和控制器数量，控制器应该安放在什么位置，管理哪些交换机，以达到最优配置。区分 QoS 的 vSDN 映射问题如上所述。控制器部署后位置不再变化，在 vSDN 映射过程中，各交换机节点的占用流表发生变化，各控制器计算和管理流表的负载也相应变化。控制器自适应调整问题是，如何动态调整各控制器与交换机的管理关系，使控制器不发生过载。

3　多控制器条件下区分 QoS 的 vSDN 映射数学模型

3.1　映射目标

多控制器条件下区分 QoS 的 vSDN 映射问题的目标是在 vSDN 请求到达和生存期持续过程中，充分利用物理网络资源，接受更多 vSDN 请求并确保所需的 QoS 服务，提高收益，减小开销。优化目标包括以下几个方面。

1) 控制通路平均时延

交换机与控制器间的控制通路时延影响控制器对 SDN 中事件的响应速度，控制器间的控制通路时延影响控制器信息同步。定义控制通路平均时延为

$$T_{mean}=\frac{1}{|L_c|}\sum_{l_c\in L_c}dl(l_c)=\frac{1}{|L_c|}\left(\sum_{l_{cs}\in L_{cs}}dl(l_{cs})+\sum_{l_{cc}\in L_{cc}}dl(l_{cc})\right) \tag{2}$$

其中，$|L_c|$为控制通路数量，L_{cs}、L_{cc}分别表示交换机到控制器、控制器间的控制通路集合，且$L_c=L_{cs}\cup L_{cc}$，l_{cs}、l_{cc}分别表示两类控制通路。

2) 请求接受率

请求接受率定义为一定时间内成功映射的 vSDN 请求数量$vSDNR_{map}(t)$与全部到达的 vSDN 请求数量$vSDNR(t)$之比，如式(3)所示。

$$r=\lim_{T\to\infty}\frac{\sum_{t=0}^{T}vSDNR_{map}(t)}{\sum_{t=0}^{T}vSDNR(t)} \tag{3}$$

3) 收益开销比

定义 t 时刻 vSDN 请求G_{vSDN}的收益为

$$R(G_{vSDN},t)=\gamma\left(\sum_{n_v\in N_v}cpu(n_v)+\alpha_1\sum_{n_v\in N_v}tcam(n_v)+\beta_1\sum_{l_v\in L_v}bw(l_v)\right) \tag{4}$$

其中，γ为不同 QoS 请求的收益权重，本文设定对于QoS_2，$\gamma=1.2$，对于QoS_1，$\gamma=1$；α_1、β_1是流表和链路带宽相对 CPU 资源的收益权重，设定$\alpha_1=1$，$\beta_1=1$。

定义 t 时刻 vSDN 请求G_{vSDN}的开销为

$$cost(G_{vSDN},t)=\sum_{n_v\in N_v}cpu(n_v)+\alpha_2\left(\sum_{n_v\in N_v}tcam(n_v)+\sum_{n_s\in P_{L_v}}tcam(n_s)\right)+\beta_2\sum_{l_v\in L_v}hops(l_v)bw(l_v) \tag{5}$$

其中，P_{L_v}为G_{vSDN}全部物理路径，$\sum_{n_s\in P_{L_v}}tcam(n_s)$为路径经过节点消耗流表的数量，$hops(l_v)$是$l_v$映射路径的跳数，$\alpha_2$、$\beta_2$是流表与物理链路开销相对 CPU 的开销权重，设定$\alpha_2=1$，$\beta_2=1$。

收益开销比定义为

$$\frac{R}{C}=\lim_{T\to\infty}\frac{\sum_{t=0}^{T}\sum_{G_{\mathrm{vSDN}}\in vSDN_{\mathrm{map}}(t)}R\left(G_{\mathrm{vSDN}},t\right)}{\sum_{t=0}^{T}\sum_{G_{\mathrm{vSDN}}\in vSDN_{\mathrm{map}}(t)}Cost\left(G_{\mathrm{vSDN}},t\right)} \tag{6}$$

4) 控制器负载失衡度

vSDN 请求到达和生存期持续过程中，各物理节点的占用流表发生变化，各控制器计算和管理流表的负载相应变化，定义控制器负载失衡度

$$D=\left(\frac{\sum_{n_{\mathrm{c}}}^{|N_{\mathrm{c}}|}(Atcam(n_{\mathrm{c}})-\overline{Atcam(n_{\mathrm{c}})})^2}{|N_{\mathrm{c}}|}\right)^{\frac{1}{2}} \tag{7}$$

其中，$\overline{Atcam(n_{\mathrm{c}})}$ 为控制器负载的均值。

3.2 数学模型

本节以最小化控制通路平均时延、vSDN 映射开销和控制器负载失衡度为目标，将多控制器的部署和自适应调整问题建模为多目标非线性整数规划模型（NLIP，nonlinear integer program），具体过程如下。

目标函数为最小化控制通路时延和负载失衡度，如式(8)和式(9)所示。

$$\min \quad T_{\mathrm{mean}} \tag{8}$$

$$\min \quad D \tag{9}$$

多控制器部署的约束条件包括：在交换机节点集合中选取 $|N_{\mathrm{C}}|$ 个节点放置控制器；每个交换机节点有且仅有一条到所属控制器的控制通路；任意 2 个控制器间有一条控制通路；2 种控制通路分别满足时延约束，如式(10)和式(11)所示。

$$dl(l_{\mathrm{cs}})\leqslant t_{\mathrm{cs}},\ \forall l_{\mathrm{cs}}\in L_{\mathrm{cs}} \tag{10}$$

$$dl(l_{\mathrm{cc}})\leqslant t_{\mathrm{cc}},\ \forall l_{\mathrm{cc}}\in L_{\mathrm{cc}} \tag{11}$$

将区分 QoS 的 vSDN 映射问题建模为整数线性规划模型（ILP, integer linear program），具体过程如下。

目标函数是最小化映射开销，如式(12)所示。

$$\min \quad \sum_{n_{\mathrm{v}}\in N_{\mathrm{v}}}cpu(n_{\mathrm{v}})+\alpha_2\left(\sum_{n_{\mathrm{v}}\in N_{\mathrm{v}}}tcam(n_{\mathrm{v}})+\sum_{n_{\mathrm{s}}\in P_{l_{\mathrm{v}}}}tcam(n_{\mathrm{s}})\right)+\beta_2\sum_{l_{\mathrm{v}}\in L_{\mathrm{v}}}hops(l_{\mathrm{v}})bw(l_{\mathrm{v}}) \tag{12}$$

节点映射约束需要满足 CPU 和 TCAM 资源约束、位置约束，且同一 vSDN 中节点不能映射到同一交换机节点上。链路映射约束需要满足带宽资源约束。

QoS 方面，对 QoS_2 需要满足控制通路和传输时延约束，如式(13)和式(14)所示。

$$dl(l_{\mathrm{vc}})\leqslant t_{\mathrm{cs_1}},\ \forall l_{\mathrm{vc}}\in G_{\mathrm{vSDN}} \tag{13}$$

$$dl(l_{\mathrm{v}})\leqslant t_p,\ \forall l_{\mathrm{v}}\in L_{\mathrm{v}} \tag{14}$$

4 启发式方法设计

本文设计启发式求解方法，流程如图 3 所示。首先，根据物理 SDN 信息和控制器数量，进行控制器部署和控制域划分；然后，在映射过程中，对不同 QoS 的 vSDN 请求，采用对应算法映射；最后，在每个 vSDN 映射成功并占用物理网络资源后，如果有控制器负载超过阈值，则迁移交换机节点，调整控制器负载。

4.1 基于免疫优化算法的多控制器部署方法

基于免疫优化算法的控制器部署方法（IACP）流程如图 4 所示，具体步骤如下。

1) 初始化网络信息，包括物理网络信息、控制器集合 N_{c}。

2) 产生初始抗体种群 An，其数量为 $|An|$，设定抗体 $\boldsymbol{X}_i\in An$，表示控制器部署的一个方案，抗体长度为控制器个数 $|N_{\mathrm{c}}|$，抗体 $\boldsymbol{X}_i$ 表示为

$$\boldsymbol{X}_i=[x_{i1},x_{i2},\cdots,x_{i|N_{\mathrm{c}}|}],\ x_{ij}\in\{1,2,\cdots,|N_{\mathrm{s}}|\} \tag{15}$$

抗体中每一个元素为交换机节点序号，表示控制器与该交换机部署在相同位置。对控制器部署方案 $\boldsymbol{X}_i$，对应的控制通路集合表示为 $L_{\boldsymbol{X}_i\mathrm{c}}=L_{\boldsymbol{X}_i\mathrm{cc}}\cup L_{\boldsymbol{X}_i\mathrm{cs}}$，其中，对 $\forall l_{\mathrm{cc}}\in L_{\boldsymbol{X}_i\mathrm{cc}}$，$l_{\mathrm{cc}}$ 为方案 $\boldsymbol{X}_i$ 中控制器间最短时延路径；对 $\forall l_{\mathrm{cs}}\in L_{\boldsymbol{X}_i\mathrm{cs}}$，$l_{\mathrm{cs}}=\arg\min_{n_{\mathrm{c}}\in N_{\mathrm{c}}}dl(n_{\mathrm{s}},n_{\mathrm{c}})$，为方案 $\boldsymbol{X}_i$ 中交换机 n_{s} 到控制器的最短时延路径，同时 n_{s} 划分到该控制域，$n_{\mathrm{s}}\in CA(n_{\mathrm{c}})$。

对 $\forall l_{\mathrm{cc}}\in L_{\boldsymbol{X}_i\mathrm{cc}}$，$\forall l_{\mathrm{cs}}\in L_{\boldsymbol{X}_i\mathrm{cs}}$，根据式(10)和式(11)判断该抗体是否满足需求，不满足则产生新的抗体。

3) 计算抗体 $\boldsymbol{X}_i$ 的适应度值 $A_{\boldsymbol{X}_i}$、抗体间亲和力 $S_{\boldsymbol{X}_i,\boldsymbol{X}_j}$ 和浓度 $C_{\boldsymbol{X}_i}$。

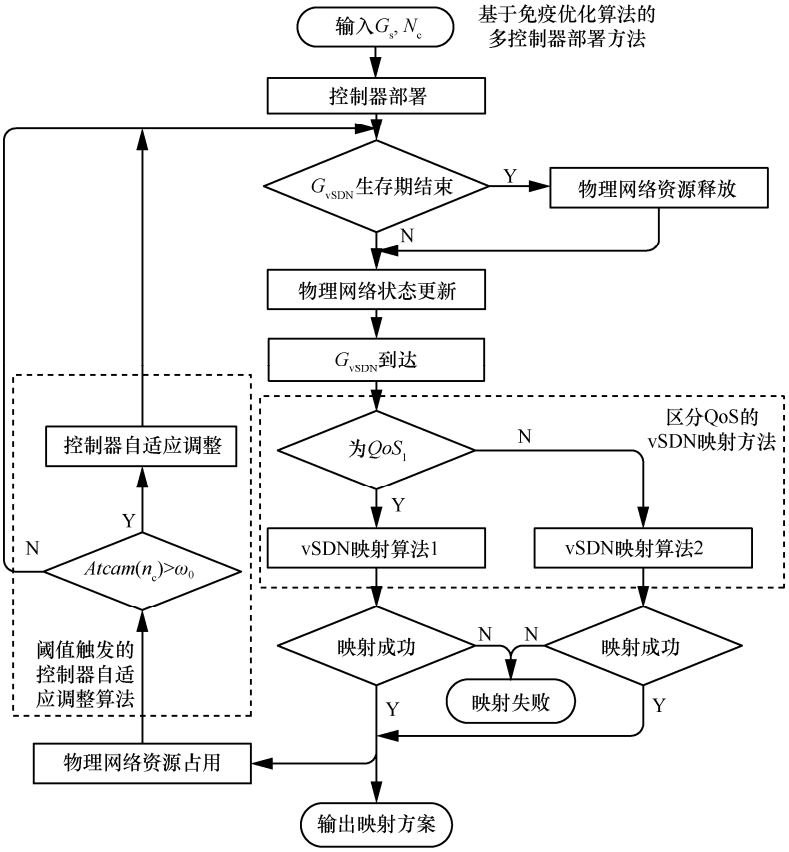


图 3　多控制器条件下区分 QoS 的 vSDN 网络映射方法流程

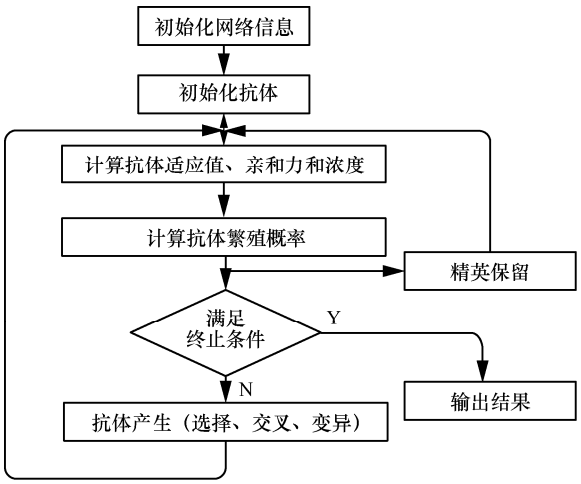


图 4　基于免疫优化算法的多控制器部署方法流程

$$A_{X_i}=\frac{1}{T_{\text{mean}}}=\frac{|L_c|}{\sum_{l_c\in L_{X_ic}} dl(l_c)} \tag{16}$$

$$S_{X_i,X_j}=\frac{k_{X_i,X_j}}{|N_c|} \tag{17}$$

$$C_{X_i}=\frac{1}{|A_n|}\sum_{X_j\in A_n} S_{X_i,X_j},\quad S_{X_i,X_j}=\begin{cases}1, & S_{X_i,X_j}>\omega\\ 0, & \text{其他}\end{cases} \tag{18}$$

其中，k_{X_i,X_j} 代表抗体 $\boldsymbol{X}_i$、$\boldsymbol{X}_j$ 中相同元素的个数，ω 为预先设定的阈值。

4) 计算抗体繁殖概率 $P(\boldsymbol{X}_i)$，判定是否结束。

$$P(\boldsymbol{X}_i)=\varepsilon\frac{A_{X_i}}{\sum A_{X_i}}+(1-\varepsilon)\frac{C_{X_i}}{\sum C_{X_i}} \tag{19}$$

其中，ε 为常数。

5) 对抗体进交叉、选择和变异操作，产生新抗体种群，返回步骤 2)循环。

4.2　区分 QoS 的 vSDN 映射方法

对于 QoS_2，设计了保证时延约束的开销优化 vSDN 映射算法（CO-vSDNM, vSDN mapping algorithm with cost optimization）和时延优化 vSDN 映射算法（TO-vSDNM，vSDN mapping algorithm with transmit delay optimization），对 QoS_1 设计了不考虑时延约束的开销最小化 vSDN 映射算法（MC-vSDNM, vSDN mapping algorithm with minimum cost），这 3 种算法均为两阶段映射算法，首先，进行节点映射，然后，进行链路映射，其主要区别在于 CO-vSDNM 与 MC-vSDNM 注重最小化映射开销，而 TO-vSDNM 注重最小化转发层的传输

时延，算法实现的差异在于候选物理节点的筛选条件、排序方法和链路映射方法。

4.2.1 节点映射

节点映射阶段，首先对虚拟节点排序，然后进行物理节点的排序和选取。

定义虚拟节点请求资源丰富度函数为

$$R(n_{\mathrm{v}})=\left(cpu(n_{\mathrm{v}})+tcam(n_{\mathrm{v}})\right)\sum_{l_{\mathrm{v}}\in L(n_{\mathrm{v}})}bw(l_{\mathrm{v}}) \quad (20)$$

其中，$L(n_{\mathrm{v}})$ 是 n_{v} 在虚拟网络中的邻边集合。

在虚拟节点排序过程中，以 $R(n_{\mathrm{v}})$ 最大的虚拟节点为根节点，运行广度优先搜索（BFS, breadth first search）算法，将剩余虚拟节点按照与根节点的距离划分为集合 $\Omega_{n_{\mathrm{v}}}^{1},\Omega_{n_{\mathrm{v}}}^{2},\cdots,\Omega_{n_{\mathrm{v}}}^{n}$，其中，$\Omega_{n_{\mathrm{v}}}^{i}$ 代表距离根节点 i 跳的虚拟节点集合，并对每一个 $\Omega_{n_{\mathrm{v}}}^{i}$ 中的虚拟节点按 $R(n_{\mathrm{v}})$ 值从大到小排序，最后得到虚拟节点的映射顺序。

对每一个虚拟节点，其候选物理节点的排序考虑物理节点的资源丰富度和邻近性原则，定义物理节点适应值（NF, node fitness）函数为

$$NF(n_{\mathrm{s}})=\frac{R(n_{\mathrm{s}})}{D(n_{\mathrm{s}})+\mu} \quad (21)$$

其中，$R(n_{\mathrm{s}})$ 采用式(20)计算，为避免分母为 0，引入一个非常小的正数 μ，$D(n_{\mathrm{s}})$ 为物理节点的距离参数。

对于 CO-vSDNM 算法，计算 $D(n_{\mathrm{s}})$ 时，首先取待映射虚拟节点 n_{vi}，取满足式(10)和节点映射约束的物理节点构成候选物理节点集合 $Can(n_{vi})$，然后取 n_{vi} 在虚拟网络的邻居节点中已映射成功的虚拟节点所对应的物理节点集合 $Embed(n_{vi})=\{n_{\mathrm{s}}\,|\,n_{\mathrm{v}}\uparrow n_{\mathrm{s}},hops(n_{\mathrm{v}},n_{vi})=1\}$，其中，$n_{\mathrm{v}}\uparrow n_{\mathrm{s}}$ 代表 n_{v} 映射到 n_{s}，$hops(n_{\mathrm{v}},n_{vi})=1$ 代表虚拟网络中 n_{v} 到 n_{vi} 距离为一跳。对 CO-vSDNM 算法，定义每一个候选物理节点的距离参数为

$$D(n_{\mathrm{s}})=\sum_{n_k\in Embed(n_{vi})}hops(n_{\mathrm{s}},n_k),n_{\mathrm{s}}\in Can(n_{vi}) \quad (22)$$

对 TO-vSDNM 算法，计算 $D(n_{\mathrm{s}})$ 时其他步骤相同，区别在于取时延的累加

$$D(n_{\mathrm{s}})=\sum_{n_k\in Embed(n_{vi})}dl(l_{sk}),n_{\mathrm{s}}\in Can(n_{vi}) \quad (23)$$

对于 MC-vSDNM 算法，对虚拟节点 n_{vi} 取满足节点映射约束的候选物理节点集合，计算 $D(n_{\mathrm{s}})$ 的其他步骤与 CO-vSDNM 相同。

虚拟节点映射到 NF 最大的候选物理节点上。

算法 1 节点映射算法

输入 G_{s}，G_{v}

输出 *NodeMappingList*

1) for 每一个虚拟节点 $n_{\mathrm{v}}\in N_{\mathrm{v}}$ do

2) 计算 n_{v} 的资源需求 $R(n_{\mathrm{v}})$

3) end for

4) 以 R 最大的虚拟节点 n_{vi} 为根节点运行广度优先搜索算法，将剩余虚拟节点按与 n_{vi} 的距离划分为集合 $\Omega_{n_{\mathrm{v}}}^{1},\Omega_{n_{\mathrm{v}}}^{2},\cdots,\Omega_{n_{\mathrm{v}}}^{n}$

5) 对 $\Omega_{n_{\mathrm{v}}}^{i}$ 中的虚拟节点，根据 $R(n_{\mathrm{v}})$ 从大到小重新排序

6) 将虚拟节点映射顺序存入链表 *VirtualNodeList*

7) for *VirtualNodeList* 中的每一个虚拟节点 n_v do

8) 根据 QoS_{v} 取满足式(10)和节点映射约束或单纯节点映射约束的候选节点集合 $Can(n_{vi})$

9) 候选节点集合中排除已映射物理节点 $Can(n_{vi})=Can(n_{vi})\bigcap OpSubNode$

10) if $Can(n_{vi})$ 为空 then

11) Return NODE_MAPPING_FAILED

12) else

13) 取 n_{vi} 已映射邻居节点对应的物理节点集合 $Embed(n_{vi})$

14) for $Can(n_{vi})$ 中每一个候选节点 n_{s}

15) 根据 QoS_{v} 计算 $NF(n_{\mathrm{s}})$

16) end for

17) 将 n_{v} 映射至 *NF* 最高的候选节点上，并将映射结果存入 *NodeMappingList* 中

18) 更新已映射物理节点集合 *OpSubNode*

19) end if

20) end for

21) return NODE_MAPPING_SUCCESS

4.2.2 链路映射

链路映射过程采用最短路径算法，CO-vSDNM 和 TO-vSDNM 分别取满足时延约束的跳数最短和时延最短路径作为映射结果；MC-vSDNM 取不考虑时延约束的跳数最短路径作为映射结果。

算法 2 链路映射算法

输入 G_{s}，G_{v}，*NodeMappingList*

输出　*LinkMappingList*

1) for 每条待映射的虚拟链路 $l_{uv} \in L_{\mathrm{v}}$ do

2)　运行最短路径算法，得到节点 u 、v 间满足带宽需求的路径 P_{uv}

3)　if $QoS = 2 \,\&\&\, dl(P_{uv}) > t_{\mathrm{ss}}$ then

4)　　return LINK_MAPPING_FAILED

5)　else

6)　将 l_{uv} 映射到 *PF* 最大的路径上，并将映射结果存入 *LinkMappingList* 中

7)　end if

8) end for

9) return LINK_MAPPING_SUCCESS

vSDN 映射算法的时间复杂度主要包括节点映射算法中计算底层节点之间最短路径，时间复杂度为 $O(|N_{\mathrm{s}}|^2)$，对每一个虚拟节点，需要计算每一个候选节点的 NF，复杂度为 $O(|N_{\mathrm{v}}||N_{\mathrm{s}}|^2)$，$|N_{\mathrm{s}}|$ 取决于位置约束大小。链路映射算法对每一条虚拟链路采用最短路径算法求解路径，时间复杂度为 $O(|L_{\mathrm{v}}||N_{\mathrm{s}}|^2)$。算法总时间复杂度为 $O(|N_{\mathrm{v}}||N_{\mathrm{s}}|^2 + |L_{\mathrm{v}}||N_{\mathrm{s}}|^2)$，3 种算法均是多项式时间内可解的启发式算法。

4.3　阈值触发的控制器自适应调整算法

本文设计阈值触发的控制器自适应调整（TTCA，controller adaptive adjustment with threshold trigger）算法，根据控制器负载情况，通过迁移交换机，改变控制器与交换机的管理关系，避免控制器过载。

1) 开始调整的时机：$\exists n_{\mathrm{c}} \in N_{\mathrm{c}}$，$Atcam(n_{\mathrm{c}}) > \eta$，其中，$\eta$ 为预先设定的阈值，开始第一次调整。

2) 迁出控制域：$CA(n_{\mathrm{c}}) = \arg\max\limits_{n_{\mathrm{c}} \in N_{\mathrm{c}}} \left(Atcam(n_{\mathrm{c}}) \right)$，取负载最重的控制域。

3) 迁出交换机数量：管理流表负载最重和最轻的控制器分别表示为 $n_{\mathrm{c,max}}$ 和 $n_{\mathrm{c,min}}$，迁移流表数量 $\Delta_{TCAM} = \dfrac{Atcam(n_{\mathrm{c,max}}) - Atcam(n_{\mathrm{c,min}})}{2}$，迁出交换机数量需要满足其占用流表的总数量不小于 Δ_{TCAM}。

4) 迁入控制域、迁出交换机与交换机迁移限制条件：从负载最轻控制域 $CA(n_{\mathrm{c,min}})$ 开始迁入，将 $CA(n_{\mathrm{c,max}})$ 中交换机按到 $n_{\mathrm{c,min}}$ 的时延排序，将满足条件 $dl(n_{\mathrm{s}}, n_{\mathrm{c,min}}) \leqslant t_{\mathrm{cs}}$ 的交换机依次迁入 $CA(n_{\mathrm{c,min}})$，直至迁出交换机占用流表的总数量大于 Δ_{TCAM}；如果满足 $dl(n_{\mathrm{s}}, n_{\mathrm{c,min}}) \leqslant t_{\mathrm{cs}}$ 约束的交换机全部迁出后，占用流表的总数量小于 Δ_{TCAM}，将 $CA(n_{\mathrm{c,max}})$ 中剩余交换机按到负载次轻的控制器 $n_{\mathrm{c,min}_2}$ 的时延排序，重复上述过程，直至满足步骤 3)中的条件。

5) 调整结束的时机：第一次调整结束后，如果 $\exists n_{\mathrm{c}} \in N_{\mathrm{c}}$，$Atcam(n_{\mathrm{c}}) > \eta$，开始第二次调整，总调整次数 $count < \dfrac{|N_{\mathrm{C}}|}{2}$，避免多数控制器负载超过阈值时反复调整，陷入循环。

算法 3　TTCA 算法流程

输入　$CA(n_{\mathrm{c}})$，$Atcam(n_{\mathrm{c}})$

输出　$CA(n_{\mathrm{c}})$

1)　$count = 1$，$k_1 = 1$

2) while　$\exists n_{\mathrm{c}} \in N_{\mathrm{c}}$，$Atcam(n_{\mathrm{c}}) > \eta$　&　$count < \dfrac{|N_{\mathrm{C}}|}{2}$

3)　根据 $Atcam(n_{\mathrm{c}})$ 对 $CA(n_{\mathrm{c}})$ 从小到大排序为 $CA(n_{\mathrm{c},1}), CA(n_{\mathrm{c},2}), \cdots, CA(n_{\mathrm{c},k})$，计算 Δ_{TCAM}

4)　while　$\Delta_1 < \Delta_{TCAM} \,\&\&\, k_1 < |N_{\mathrm{c}}| - 1$

5)　$CA(n_{\mathrm{c},k})$ 中节点按到 n_{c,k_1} 的时延排序

6)　将 $n_{\mathrm{s}} \in CA(n_{\mathrm{c},k})$ 且 $dl(n_{\mathrm{s}}, n_{\mathrm{c},k_1}) \leqslant t_{\mathrm{cs}}$ 依次迁入 $CA(n_{k_1})$，计算 $\Delta_1 = \sum contcam(n_{\mathrm{s}})$

7)　$k_1 = k_1 + 1$

8)　end while

9)　$count = count + 1$

10) end while

5　性能评估与分析

5.1　仿真环境设置

实验通过 Matlab 进行仿真评估。物理 SDN 在 $L \times L = 1\,000\ \mathrm{km} \times 1\,000\ \mathrm{km}$ 范围内生成均匀分布的 100 个节点，500 多条链路，节点 CPU、TCAM 和链路带宽服从[50, 100]的均匀分布，链路时延 $dl(l_{\mathrm{s}}) = \dfrac{3d(l_{\mathrm{s}})}{2c}$，$d(l_{\mathrm{s}})$ 为链路 l_{s} 的长度，$c = 3 \times 10^5$ km/s，控制器数量 $|N_{\mathrm{c}}| = 5$。vSDN 请求的节点在 $L \times L = 1\,000\ \mathrm{km} \times 1\,000\ \mathrm{km}$ 的范围内均匀分布，节点个数服从[5,15]均匀分布，节点 CPU 资源和链路带宽资源请求服从[10,30]均匀分布，设置每一个节点的流表请求 $tcam(n_{\mathrm{v}}) = |N_{\mathrm{v}}| - 1$，虚拟链路经过节点流表消耗 $tcam(n_{\mathrm{s}}) = 2$，节点位置约束 $D(n_{vi}) = 300\ \mathrm{km}$。设置 vSDN 请求到达服从泊松分布，每 100 时间单位到达数量为 10，其生存期服从均值为 200 的指数分布。仿

真运行时间为 3 000 个时间单位，包含大约 300 个 vSDN 请求。设置 2 种 QoS 的 vSDN 请求数量，QoS_2 比例为$\frac{1}{3}$，数量约为 100，QoS_1 数量约为 200。

在当前参数设置条件下，计算物理 SDN 的链路时延 $dl(l_s)$ 的分布区间为[1.904 9×10^{-4}, 1.2×10^{-3}]，所有物理节点间最短时延路径时延 $dl(P_s)$ 的分布区间为[1.904 9×10^{-4}, 6.5×10^{-2}]。本文设置控制通路时延约束 $t_{cc}=0.002$ s，$t_{cs}=0.003$ s，设置 QoS_2 的 vSDN 请求服务质量约束 $t_{cs_1}=0.001\,5$ s，$t_p=0.004$ s，设置控制器调整阈值 $\eta=500$。

为避免随机因素对实验结果产生扰动，仿真实验共进行 10 次，并取实验结果的平均值作为最终仿真结果。

表 1 为对比算法及描述，现有算法不能直接用于解决多控制器条件下区分 QoS 的 vSDN 映射问题；采用 IACP 控制器部署方法，对 QoS_2 的 vSDN 请求分别采用 CO-vSDNM 算法和 TO-vSDNM 算法，对 QoS_1 的 vSDN 请求采用 MC-vSDNM 方法，采用 TTCA 控制器调整方法，分别记为 BM_1、BM_2；对文献[9]提出的基于拓扑连接特征的虚拟网络映射（TF-vSDNM）算法进行修改，在映射过程不考虑时延约束，同时采用 IACP 与 TTCA 方法，记为 CM_1；CM_2 采用随机部署控制器方法，用以考察控制器部署对映射效果的影响。

表 1　　对比算法及描述

方法	控制器部署	QoS_2 vSDN 请求	QoS_1 vSDN 请求	控制器调整
BM_1	IACP	CO-vSDNM	MC-vSDNM	TTCA
BM_2	IACP	TO-vSDNM	MC-vSDNM	TTCA
CM_1	IACP	TF-vSDNM	TF-vSDNM	TTCA
CM_2	随机部署	CO-vSDNM	MC-vSDNM	TTCA

5.2 仿真结果分析

1) vSDN 请求接受率、收益开销比

图 5(a)为 4 种方法 vSDN 请求接受率随时间变化的情况，从图 5(a)中可以看出，BM_1 的请求接受率最高，约为 80%，其次为 CM_2，约为 76%，再次为 BM_2(74%)和 CM_1(72%)。BM_1 比 CM_2 性能好的原因在于控制器部署更加合理，更容易满足控制通路时延约束，有利于成功映射。

图 5(b)为 4 种方法收益开销比随时间变化情况，BM_1 的收益开销比最高，约为 0.53，其次为 CM_2，约为 0.51，再次为 BM_2(0.49)和 CM_1(0.42)。主要原因在于 BM_1 对 2 种 QoS 的 vSDN 请求均采用开销优化映射算法，减小了开销。

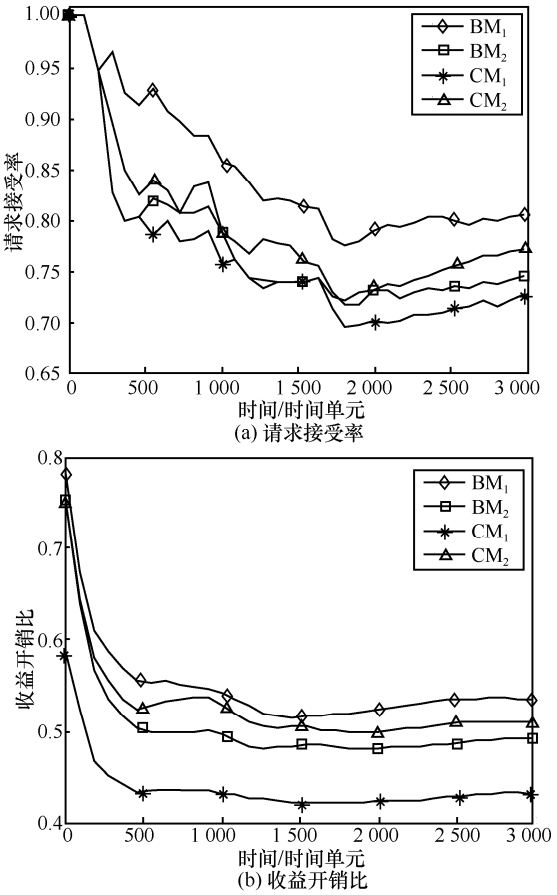


图 5　4 种算法的性能比较

2) 不同 QoS 的 vSDN 请求的映射成功数量

如表 2 所示，实验中 vSDN 中 QoS_2 请求总数为 92，QoS_1 请求总数为 211。显然，对 QoS_2，BM_1 映射成功数量最多，共 56 个。主要原因是 QoS_2 请求对资源的时延约束更苛刻，BM_1 的映射过程在满足 QoS 约束条件下开销最小，有利于后续 QoS_2 请求映射成功，CM_1 对所有 vSDN 请求均采用时延最小算法映射，使关键链路负载过重，不利于后续 QoS_2 请求映射成功。从表 2 中可以看出，4 种方法对于 QoS_1 请求映射成功数相近，这是因为这类请求没有时延约束，映射成功率取决于物理网络资源是否充足。

通过图 5 和表 2 中 BM_1 与 CM_1 的性能比较，说明本文提出的多控制器部署方法可以提高 vSDN 映射成功率。通过 BM_1 与 BM_2、CM_1 的性能比较，说明对于高 QoS 约束的 vSDN 请求，物理网络中时

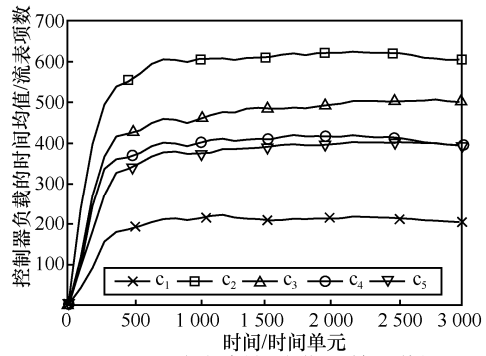
延短的关键链路非常重要，在满足 QoS 约束条件下，选择开销最小的路径，减小对短时延链路的占用，有利于高 QoS 约束的 vSDN 请求的映射成功。

表 2　　4 种方法 vSDN 映射成功数比较

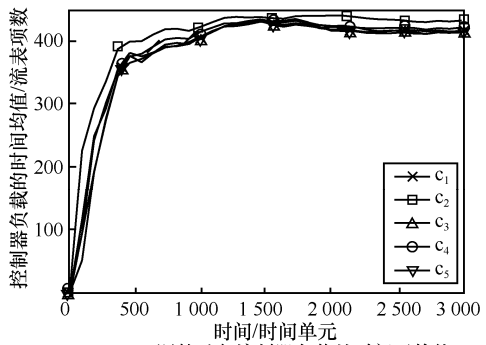
方法	映射成功数		
	QoS_2	QoS_1	总数
BM_1	56	188	244
BM_2	36	188	224
CM_1	27	191	218
CM_2	43	189	232

3) 控制器负载失衡度

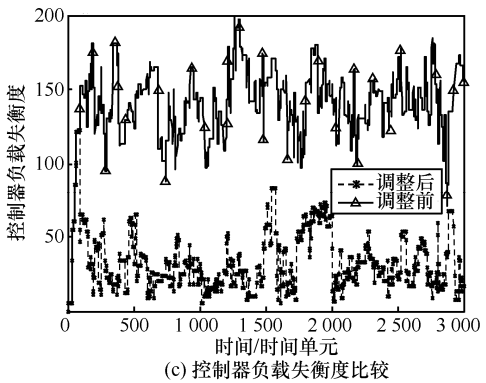
图 6 为控制器自适应调整算法对控制器负载的影响，其中，图 6(a)和图 6(b)为 TTCA 调整前后各控制器管理流表负载的时间平均值，图 6(c)为 TTCA 调整前后各时刻控制器的负载失衡度。从图 6(a)中可以看出各控制器的负载差距较大，这是因为各控制器管理交换机节点不同、vSDN 请求占用的节点和节点流表数量不同，导致各控制器负载出现差异；经过调整，图 6(b)中可以看出 5 个控制器的负载曲线很接近，显著改善了控制器的负载均衡程度。TTCA 的负载失衡度明显下降。说明本文提出 TTCA 算法有助于改善控制器的负载均衡，避免控制器过载。



(a) 无调整各控制器负载的时间平均值



(b) TTCA调整后各控制器负载的时间平均值



(c) 控制器负载失衡度比较

图 6　控制器自适应调整对负载的影响

6　结束语

本文研究了多控制器 SDN 虚拟化环境中区分 QoS 的 vSDN 映射问题，建立了多控制器部署问题和 vSDN 映射问题的数学模型，提出了一种多控制器条件下区分 QoS 的 vSDN 映射方法。仿真结果表明，通过优化多控制器的部署位置和改善控制器的负载均衡，能够满足不同用户对 vSDN 服务质量的需求，提高了 vSDN 映射的成功率和收益开销比。

参考文献：

[1] WANG A, IYEN M, DUTTA R, et al. Network virtualization: technologies, perspectives, and frontiers[J]. Journal of Lightwave Technology, 2013, 31(4): 523-537.

[2] ANDERSON T, PETERSON L, SHENKER S, et al. Overcoming the Internet impasse through virtualization[J]. Computer, 2005, 38(4): 34-41.

[3] YIN X, HVANG S, WANG S, et al. Soft ware defined virtualization platform based on double-Flow visors in multiple domain networks[C]//8th International Conference on Communications and Networking. 2013: 776-780.

[4] SALVADORI E, DORIGUZZI CORIN R, et al. Generalizing virtual network topologies in OpenFlow-based networks[C]//Global Telecommunications Conference (GLOBECOM 2011). 2011: 1-6.

[5] JIN X, REXFORD J, WALKER D. Incremental update for a compositional SDN hypervisor[C]//The Third Workshop on Hot Topics in Software Defined Networking. 2014: 187-192.

[6] 张朝昆, 崔勇, 唐骞骞, 等. 软件定义网络(SDN)研究进展[J]. 软件学报, 2015, 26(1): 62-81.
ZHANG C K, CUI Y, TANG H Y, et al. State-of-the-art survey on software-defined networking (SDN)[J]. Journal of Software,2015, 26(1):62-81

[7] 刘江, 黄韬, 张晨, 等. SDN 试验床网络虚拟化切片机制综述[J]. 通信学报, 2016, 37(4): 2016083.
LIU J, HUANG T, ZHANG C, et al. Research on network virtualization slicing mechanism in SDN-based testbeds[J]. Journal on Communications, 2016, 37(4): 2016083.

[8] MEHMET D, MOSTAFA A. Design and analysis of techniques for

mapping virtual networks to software-defined network substrates[J]. Computer Communications,2014 (45):1-10.

[9] GONG S Q, CHEN J, ZHAO S Y, et al. An efficient and coordinated mapping algorithm in virtualized SDN networks[J]. Frontiers of Information Technology & Electronic Engineering, 2016, 17(7): 701-716.

[10] 姚琳元, 陈颖, 宋飞, 等. 基于时延的软件定义网络快速响应控制器部署[J]. 电子与信息学报, 2014, 36(12): 2802-2808
YAO L Y, CHEN Y, SONG F, et al. Delay-aware controller placement for fast response in software-defined network[J]. Journal of Electronics & Information Technology, 2014, 36(12): 2802-2808.

[11] HEELER B, SHERWOOD R, MCKEOWN N, et al. The controller placement problem[C]//The First Workshop on Hot Topics in Software Defined Networks. 2012: 7-12.

[12] YAO L, HONG P L, ZHANG W, et al. Controller placement and flow based dynamic management problem towards SDN[C]// Workshop on Advances in Software Defined and Context Aware Cognitive Networks 2015 (IEEE SCAN-2015). 2015: 363-368.

[13] 王丽霞, 曲桦, 赵季红. 软件定义网络中应用二值离子化优化的控制器部署策略[J]. 西安交通大学学报, 2015, 49(6):67-71.
WANG L X, QU H, ZHAO J H. A strategy of controller placement in software defined networks using binary particle swarm optimization[J]. Journal of Xi'an Jiaotong University, 2015, 49(6):67-71.

[14] NASHIKD S, REAZ A, SHIHABURR R C, et al. Connectivity-aware virtual network embedding[C]//The 8th IFIP International Conference on New Technologies, Mobility and Security. 2016:46-54.

[15] CUI H Y, GAO W J, LIU J, et al. A virtual network embedding algorithm based on virtual topology connection feature[C]//The 16th International Symposium on Wireless Personal Multimedia Communications. 2013: 1-5.

[16] ZHANG Z B, SU S, LIN Y K, et al. Adaptive multi-objective artificial immune system based virtual network embedding [J]. Journal of Network and Computer Applications, 2015,53:140-155.

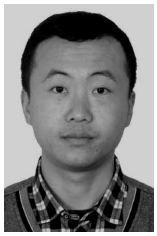
[17] WANG Z, WU J, WANG Y, et al. Survivable virtual network mapping using optimal backup topology in virtualized SDN[J]. China Communications, 2014, 11(2): 26-37.

[18] MIJUMBI R, SERRAT J, RUBIO J, et al. Dynamic resource management in SDN-based virtualized networks[C]//The 10th International Conference on Network and Service Managementl. 2014: 412-417.

作者简介：

赵志远（1989-），男，山东德州人，空军工程大学博士生，主要研究方向为虚拟网络映射、软件定义网络等。

孟相如（1963-），男，陕西蓝田人，空军工程大学教授、博士生导师，主要研究方向为空天信息网络、下一代网络等。



苏玉泽（1990-），男，山东济南人，空军工程大学博士生，主要研究方向为虚拟网络映射、认知网络等。

李振涛（1990-），男，河南柘城人，空军工程大学硕士生，主要研究方向为网络虚拟化、网络故障恢复等。