软件定义网络多控制器部署优化方法

郭一平， 胡谷雨
（陆军工程大学 指挥控制工程学院，江苏 南京 210007）

摘要：目前软件定义网络（software defined network，SDN）多控制器管理的研究重点和难点在于多控制器部署，合适的控制器数目和合理的部署位置能够实现低延迟和高可靠性的折中。在网络拓扑图上将部署位置问题简化并形式化，定义了可靠性和延迟指标，设计了多控制器部署位置求解算法（multi-controller placement location algorithm，MCP）。MCP 通过遗传算法将网络划分为多个子网，并在每个子网中放置一个控制器，以最大化网络可靠性和最小化路径延迟为目标，通过模拟退火算法确定控制器在每个子网中的位置，经对比实验验证了 MCP 生成的部署位置方案在可靠性和延迟上的性能优势。以可靠性和延迟为评估指标，基于 MCP 测试了各种网络规模的最合适控制器数目，并分析得出最合适控制器数目与网络规模之间的统计规律。

关键词：软件定义网络； 多控制器； 部署位置； 部署数目； 高可靠性； 低延迟
中图分类号：TP393.0 **DOI**：10.12018/j.issn.2097-0730.20220831001

Optimal Placement of Multiple Controllers in Software Defined Network

GUO Yiping， HU Guyu
（College of Command & Control Engineering，Army Engineering University of PLA，Nanjing 210007，China）

Abstract：The current research focus and difficulty of multi-controller management in software defined network (SDN) is the multi-controller placement. Suitable number of controllers and reasonable placement of controllers can reach a compromise between low latency and high reliability. The problem of controller placement was simplified and visualized on the network topology chart, the indicators of reliability and delay were defined, and the multi-controller placement location algorithm (referred to as MCP) was designed. With the genetic algorithm, MCP divided the network into some subnets, and every subnet was placed with a controller. With the objective to maximize network reliability and minimize path delay, the simulated annealing algorithm was used to determine the location of the controller in every subnet. The performance advantages of MCP in terms of reliability and delay were verified by comparative experiments. With reliability and delay as evaluation indicators, a lot of experiments based on MCP have been conducted and the statistical law between the most suitable number of controllers and the network size has been obtained.

Key words：software defined network (SDN)； multi-controller； location of controllers； number of controllers； high reliability； low latency

传统网络中由各个交换机执行的计算量在软件定义网络（software defined network，SDN）中将全部集中到控制器上，因此增加的延迟不言而喻，文献[1]指出使用最为广泛的 NOX 平台（首个 OpenFlow 控制器）的延迟水平无法满足 SDN 的需求。在 SDN 中，如果一个流与交换机流表中的任何规则都不匹配，其首个数据包将由交换机送至 SDN 控制器，控制器在接收到数据包后，计算对应流在网络中的转发路径，控制器向转发路径上的各个交换机安装更新规则，交换机相应地更新自身的流表，流

收稿日期：2022-08-31
第一作者（通信作者）：郭一平，博士研究生，主要研究信息网络。

的后续数据包以及其他所有能够与该规则匹配的流，按照此规则在数据平面进行转发，不再需要控制器的介入。然而，如果在同一阶段，大量需要进行规则安装或更新的流到达交换机，对流转发规则的实时安装或更新会造成控制器接口路径的拥塞，为流计算新规则将产生额外的延迟[2]。另外，SDN 将网络控制功能抽出形成集中的控制器，一旦单一控制器出现故障，整体网络必将瘫痪。集中控制器尽管功能强大，但面对的流量调度压力巨大且故障影响极大。

为保证网络正常运行，确保 SDN 控制器的可靠性，控制层单点压力问题必须解决。针对这一问题，可以参考传统的分布式网络结构提供的容错性[3]，对控制层进行多控制器管理，在主控制器出现故障后启用备份控制器[4]。有效的解决方案为：引入多个同步控制器并将流量合理地分配到不同控制器上，以缓解集中控制器的流量调度压力，并减小单点故障带来的影响。

目前对多控制器管理的研究工作主要包括多控制器的结构、同步和部署 3 个方面。多控制器的结构[5-9]与同步[10-11]研究已趋于成熟，多控制器管理的研究重点和难点在于多控制器部署。多控制器部署需要探究部署位置和部署数目，以保证控制器的可靠性和备份引入的延迟在可接受范围内。参考文献[12]研究了控制器的部署位置和部署数目，设计了基于模拟退火算法的可靠性感知控制器配置（reliability-aware controller placement，RCP）算法以输出可靠性最高的部署方案，通过实验得出定性结论：控制器数目越多，控制请求的响应延迟越低，控制器可靠性越低。文献[13]提出了以全网平均时延、控制路径可靠性和负载均衡度为参数，以网络综合性能为目标的控制器部署优化评价模型。在此基础上提出一种 RCP 方法，仿真结果表明，提出的部署策略在保证负载均衡的前提下，提高了控制网络的可靠性。文献[14]提出了一种高效的 RCP 算法，该算法通过局部搜索确定控制器位置，仿真结果表明，该算法能有效降低备份路径时延以及主路径和备份路径的累计时延。

目前多控制器的部署问题研究成果较少，大规模网络对多控制器部署提出挑战，对控制器延迟和可靠性的要求较高。合适的控制器数目和合理的部署位置均需有较优的解决方案，以期兼顾低延迟和高可靠性。

针对存在的问题，考虑采用物理分布式结构，遵循 Zookeeper 的同步方案，实现 SDN 多控制器部署方案。部署方案包括两部分内容：（1）部署位置。以输出高可靠性且低延迟的部署位置方案为目标，设计优化的部署位置问题求解算法；（2）部署数目。以实现可靠性与延迟的最佳折中为目标，寻找不同网络规模的最合适控制器数目。

1　部署位置

1.1　问题描述

控制器结构采用物理分布式，控制器同步方案遵循 Zookeeper，即各控制器平等且同步。交换机与各控制器之间、各控制器之间所有的通信链路均为 SDN 控制链路。将这些控制链路视作逻辑链路，从而在现有的物理网络上构建一个用于部署位置问题研究的虚拟网络，如图 1 所示。如物理网络中 A 节点和 B 节点之间的路径包含 $A\rightarrow C\rightarrow B$ 和 $A\rightarrow D\rightarrow B$ 两条控制链路，将其视为逻辑链路，即虚拟网络中 A 节点和 B 节点之间仅有一条控制路径 $A\rightarrow B$。

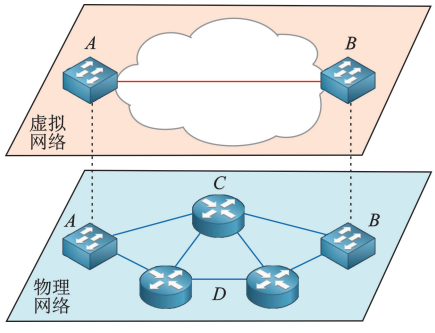


图 1　物理网络-虚拟网络

在物理网络的网络拓扑图 $G(V,E)$ 上，V 表示节点（交换机）集合，E 表示控制链路集合，节点数目为 n。在与之对应的虚拟网络中，控制路径数目为 m，控制器数目为 c。虚拟网络拓扑结构如图 2 所示，控制器与某个交换机共址，多个控制器之间的逻辑关系为全连接，一个交换机只与一个控制器相连（被控制器共址的交换机只能与该控制器相连），则 $m=n+c(c-1)/2$。

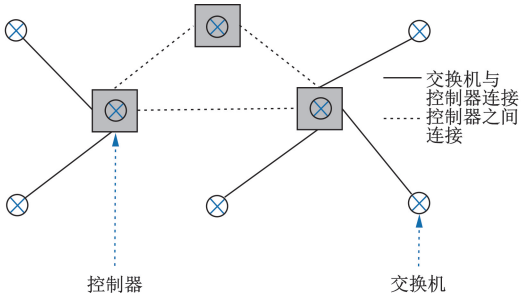


图 2　虚拟网络拓扑结构

由于两条控制路径可能共享一条物理链路，因此在分析可靠性时需考虑关联故障。采用常用的关联故障模型，即基于原因的可靠性分析模型。基本思想是控制网络组件(例如控制路径)的故障，使其具有可以明确识别并在统计上独立的潜在物理原因。

对于任一网络组件 $z \in V \cup E$，p_z 表示 z 出现故障的可能性。这里可以假定网络组件发生故障时，其对应的控制路径也发生故障。因此，不妨设网络组件 z 的故障会导致 m_z 条控制路径发生故障。

可靠性 θ 的定义如式(1)所示。可靠性是指整体网络的平均故障率，θ 值越低小，可靠性越高。

$$\theta = \frac{\sum_{z \in V \cup E} p_z m_z}{m + n + c} \tag{1}$$

延迟 l 的定义如式(2)所示。l 为交换机(或控制器)发送请求到控制器接收请求的平均传输时延。l 值越大，延迟越高。

$$l = \frac{\sum_{i=1}^{n}\sum_{j=1}^{c} l_{i,j}x_{i,j} + \sum_{i=1}^{c}\sum_{j=1}^{c-i} l_{i,j}}{m} \tag{2}$$

式中：$l_{i,j}$ 表示节点 i 到节点 j 的路径延迟，$x_{i,j}$ 表示交换机 i 是否配置到控制器 j(是=1，否=0)。使用两个节点之间的地理距离作为这两个节点之间通信延迟的近似值[14]。

式(3)确保每个交换机都映射到有且仅有一个的控制器。

$$\sum_{j=1}^{c} x_{i,j} = 1 \tag{3}$$

无需关心可靠性和延迟这两个指标各自与控制器数目的增长关系，只关心可靠性与延迟的最佳折中，即 θl 值最低。

1.2 算法设计

需要设计 c 个控制器的部署位置求解算法，以期达到高可靠性和低延迟的目的。为降低延迟，采用子网划分以减少远距离高成本传输；为保证可靠性，采用基于可靠性和延迟的随机寻优算法确定控制器位置。

1.2.1 子网划分

SDN 中共有 n 个交换机和 c 个控制器。将若干个交换机与一个控制器组成一个子网，子网中的所有交换机连接到其中的控制器上。这里的控制器位置并非最后确定的位置，只是用于子网划分，因此仅需均匀分布即可。子网划分对均匀分布的控制器之间路径延迟的影响较小，且这部分延迟相对总体延迟而言数值过小。因此，子网划分时只考虑控制器与交换机之间的路径延迟总和，形式化如式(4)所示。目标是找到一种子网划分的解使得 l_{sum} 最小化。

$$l_{\text{sum}} = \sum_{i=1}^{n}\sum_{j=1}^{c} l_{i,j}x_{i,j} \tag{4}$$

采用遗传算法解决这里的组合最优解问题，已知路径延迟 $\boldsymbol{D}$，求最优解 x 使 l_{sum} 最小。

用矩阵 $\boldsymbol{D}$ 表示交换机与交换机之间的路径延迟，该矩阵的行与列均为交换机编号。由于交换机位置确定，一般情况下 $\boldsymbol{D}$ 已知。

用 0－1 矩阵 $\boldsymbol{x}$ 表示子网划分的解，该矩阵的行为交换机编号列为控制器编号。例如

$$\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{5}$$

该矩阵表示交换机 1 和交换机 2 与控制器 1(共址交换机 1)组成一个子网，交换机 3 与控制器 2(共址交换机 2)组成一个子网。

将目标函数表示为

$$l_{\text{sum}} = \sum_{i=0}^{n-1} x(i,:) * D(:,i) \tag{6}$$

遗传算法的基本思想是：(1) 初始化。由若干个染色体组成的初始群体。(2) 计算每个染色体的适应度值。(3) 选择操作。按由适应度值所决定的某个规则选择进入下一代的染色体。(4) 交叉操作。随机选择两个现有染色体进行交叉重组，产生两个新染色体。(5) 变异操作。随机选择一个现有染色体，将其中某一个基因变异，产生一个新染色体。(6) 若满足终止条件，则输出群体中适应度值最优的染色体作为问题的最优解，否则转第 2 步。

将遗传算法应用于子网划分问题。(1) 染色体 y 由 n 个基因组成，$y = x(0,:)+x(1,:)+\cdots+x(n-1,:)$，初始群体 Q 由 m 个染色体组成。(2) 染色体 y 的适应度值为 l^{y}_{sum}。(3) $P\{\text{选择 } y\} = l^{y}_{\text{sum}} m / \sum_{i=1}^{m} l^{y_i}_{\text{sum}}$。(4) 对未被选择的染色体，按交叉概率 P_c 执行交叉操作，按变异概率 P_m 执行变异操作。(5) 完成预先给定的进化代数，即设置的迭代次数。算法终止时，l_{sum} 最小的染色体对应所求的最优解 x。

1.2.2 位置确定

已经将所有交换机划分成 c 个子网，需要在考虑可靠性的情况下确定每个子网中的控制器位置。

模拟退火算法是基于 Monte-Carlo 迭代求解策略的一种随机寻优算法。由初始解和控制参数初值 T 开始，对当前解重复“产生新解→计算目标函数

差→接受或舍弃”的迭代，并逐步衰减 T 值，算法终止时的当前解即为所得近似最优解。

模拟退火的基本思想是：(1) 初始化。初始温度 T(充分大)，初始解状态 S(是算法迭代的起点)，T 值的迭代次数 L。(2) 对 $k=1, \cdots, L$ 做第 3～7 步。(3) 产生新解 S'，新解由旧解变化产生。(4) 计算增量 $\Delta T=C(S')-C(S)$，其中 $C(S)$ 为评价(目标)函数。(5) 若 $\Delta T<0$ 则接受 S' 作为新的当前解，否则以概率 $\exp(-\Delta T/T)$ 接受 S' 作为新的当前解。(6) 如果满足终止条件则输出当前解作为最优解。(7) T 逐渐减少且趋于 0，然后转步骤 2。

将模拟退火算法应用于子网控制器部署位置问题。(1) 通过在 c 个最可靠的位置放置 c 个控制器来获得初始解。(2) 初始温度 T_0 为一个很大的值，以便几乎所有的新解都可以接受。将 P_0 定义为前 c 次迭代中的接受概率，将 Δ_0 定义为在随机放置的 Y 次执行中获得的最佳解决方案和最差解决方案之间的可靠性差值，$T_0=-|\Delta_0|/\ln P_0$ 设置 $P_0=0.999$，$Y=1\ 000$。(3) 每个 T 值的迭代次数 L 设置为1 000。(4) 最小化 θl。(5) 令 $P(M)$ 表示 c 个控制器的位置集合，x 表示 $P(M)$ 中一个控制器的位置，y 表示 $(V-P(M))$ 中的一个位置。对当前解中的任一 $x\in P(M)$，新解中的最佳交换位置 $y\in(V-P(M))$ 由 Δ_{xy} 确定。$\Delta_{xy}=\min\limits_{j\in(V-P(M))}\Delta_{xj}$。其中，$\Delta_{xj}$ 为当 $x\in P(M)$ 被 $j\in(V-P(M))$ 替换时的目标函数差值。当遍历 $P(M)$ 中的 x，新解产生。(6) 温度 T 呈指数下降，即 $T_{\text{new}}=\alpha T_{\text{old}}$，$\alpha$ 为设置的参数。(7) 连续若干次没有接受新解或迭代次数达到 T 值时终止。

1.2.3　算法流程

SDN 多控制器部署位置求解算法流程见算法 1，其基本思想为：以最小化路径延迟为目标，通过遗传算法将网络划分为 c 个子网，并在每个子网中放置一个控制器，以最大化网络可靠性(最小化 θ)和最小化路径延迟(最小化 l)为目标，通过模拟退火算法确定控制器在每个子网中的位置。

算法 1　多控制器部署位置求解算法

```
1:输入:交换机数目 n,控制器数目 c,矩阵 D,矩阵 p_z
2://子网划分
3:初始化:进化代数 L,交叉概率 P_c,变异概率 P_m,m 个染色体 y 组成的群体 Q
4: l_sum^y = Σ_{i=0}^{n-1} x(i,:) * D(:,i);
5:For j = 1, j ++, j ≤ L
6:    For each y ∈ Q
7:        计算 l_sum^y;
8:    Endfor
9:    For each y ∈ Q
10:       P{选择 y} = l_sum^y m / Σ_{i=1}^{m} l_sum^{y_i};
11:       If P{选择 y} <1
12:      新群体 Q̂ ← y;
13:       Endif
14:   Endfor
15:   Q_s = Q̂
16:   Q_done = ∅
17:   For each y ∈ (Q − Q_s − Q_done)
18:       随机选择 y_cp, y_cp ∈ (Q − Q_s − Q_done) 且 y_cp ≠ y;
19:       对 y 和 y_cp 按概率 P_c 执行交叉操作,生成 y^new 和 y_cp^new;
20:       Q̂ ← y^new, y_cp^new;
21:       Q_done ← y, y_cp;
22:   Endfor
23:   For each y ∈ (Q̂ − Q_s)
24:       对 y 按概率 P_m 执行变异操作,生成 y^new;
25:       Q̂ ← y^new;
26:   Endfor
27:   Q = Q̂;
28:Endfor
29:print Q 中 l_sum 最小的 y;//y 对应将网络拓扑划分成 c 个子网的最优解 x
30://位置确定
31:For i = 1, i ++, i ≤ c
32:   X = x(:,i) 中值为 1 对应的行号;//即一个子网包含的交换机编号
33:   初始化:初始温度 T,初始解状态 x ∈ X,T 值的迭代次数 L,参数 α,参数 β
34:       For k = 1, k ++, k ≤ L
35:       产生新解 x̂;
36:       ΔT = θl(x̂) − θl(x);
37:       If ΔT < 0
38:       x = x̂;
39:       Else
40:       按概率 exp(−ΔT/T) 接受 x̂;
41:       Endif
42:       If x̂ 连续被拒次数 > β
43:       break;
44:       Else
45:       T = αT
46:       Endif
47:   Endfor
48:   print x; //一个子网中的控制器最佳部署位置
49:Endfor
50:输出:c 个控制器的最佳部署位置,即与其共址的交换机编号
```

1.3　对比实验

根据算法 1 的流程，在 MATLAB 上实现了多控制器部署位置求解算法 MCP，并与文献[13,14]分别提出的两个 RCP 算法进行对比实验，对 3 个算法生成的控制器部署位置方案进行评估，评估指标包括可靠性 θ 和延迟 l。与其他现有算法相比，文献[13]

提出的算法(后文称 RCP1)能够输出较高可靠性的部署位置方案,文献[14]提出的算法(后文称 RCP2)能够输出较低延迟的部署位置方案。因此,在可靠性指标上将 MCP 与 RCP1 做对比实验,在延迟指标上将 MCP 与 RCP2 做对比实验。

MCP 的输入 $\boldsymbol{D}$ 包含了实验所需的网络拓扑信息,包括交换机数目、编号、连接关系和距离。每个不同网络拓扑对应不同且唯一的矩阵 $\boldsymbol{D}$。进行对比实验时,对性能评估有影响的 $\boldsymbol{D}$ 和 p_z 等输入值保持一致。为确保算法 1 生成的方案达到较高的性能标准,需对以下参数进行多次调整,取最优输出作为算法 1 的结果:参数 L,交叉概率 P_c,变异概率 P_m,群体规模 m,参数 α,参数 β。

1.3.1 可靠性

针对可靠性指标 θ,将 MCP 与文献[13]提出的 RCP1 作对比实验。实验结果如图 3 所示。

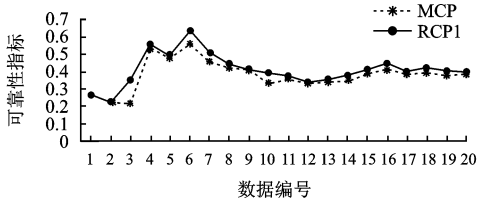


图 3 可靠性对比实验结果

由图 3 可知,MCP 生成的位置方案的 θ 值总是不高于 RCP1,90%的情况下低于 RCP1。θ 值越低,表示采取此方案的网络可靠性越高。因此,提出的多控制器部署位置求解算法在生成方案的可靠性上优于对比算法。

1.3.2 延迟

针对延迟指标 l,将 MCP 与文献[14]提出的 RCP2 做对比实验。实验结果如图 4 所示。

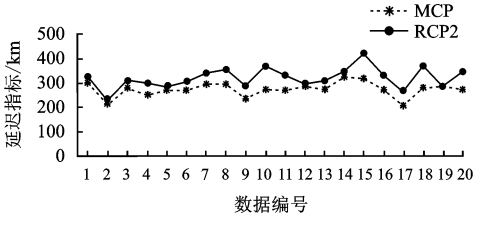


图 4 延迟对比实验结果

由图 4 可知,MCP 生成的位置方案的 l 值总是低于 RCP2。l 值越低,表示采取此方案的平均路径延迟越低。因此,提出的多控制器部署位置求解算法在生成方案的延迟上优于对比算法。

综上可知,提出的多控制器部署位置求解算法在生成方案的可靠性上优于文献[13]提出的算法,在生成方案的延迟上优于文献[14]提出的算法,即在可靠性和延迟上均有优势。

2 部署数目

基于 MCP,以请求响应延迟与控制器可靠性为评估指标,测试各种网络规模的最合适控制器数目,并试图找出合适的控制器数目与网络规模之间是否存在统计规律。

设网络的规模(交换机数目)为 n。考虑到 $\boldsymbol{D}$ 和 p_z 等网络参数对性能指标的影响,为避免在测试最合适控制器数目 c 的过程中出现大的偏差,采取多网络多次测试的方式,具体步骤为:(1)对 $\boldsymbol{D}$ 和 p_z 等输入值进行 3 次不同的设定,使其成为 3 个不同规模的网络。(2)在每一个网络上对控制器数目 c 进行 5 次不同的设定,每次设定 c 后,执行算法 1 获取最佳位置方案,计算可靠性 θ 和延迟 l。(3)取 5 次中性能值最优(θl 最小)的 c 为该网络的最佳控制器数目。(4)将 3 个网络的最佳控制器数目平均向下取整,作为规模为 n 的网络对应的最合适控制器数目 c。

以规模为 50 的网络为例,对 3 个不同网络(网络 1、2、3)进行性能测试,结果如表 1 所示。由表 1 可知,网络 1、2、3 的最佳控制器数目分别为 3、4、4。最后,将这 3 个数值平均向下取整,得到规模 $n=50$ 的网络对应的最合适控制器数目 $c=3$。

表 1 规模为 50 的网络测试结果

网络编号	控制器数目 c	可靠性 θ	延迟 l	θl
1	2	0.293	298.04	87.33
1	3	0.282	307.84	86.81
1	4	0.281	313.73	88.16
1	5	0.278	323.53	89.94
1	6	0.275	327.45	90.05
2	3	0.289	311.76	90.10
2	4	0.286	312.75	89.45
2	5	0.282	317.65	89.58
2	6	0.280	323.53	90.59
2	7	0.275	333.33	91.67
3	4	0.282	309.80	87.36
3	5	0.279	313.73	87.53
3	6	0.277	321.57	89.07
3	7	0.276	331.37	91.46
3	8	0.272	345.10	93.87

类似地,测试不同的网络规模 n 对应的最合适控制器数目 c,结果如表 2 所示。

由表 2 可知,当网络规模为 n 时,合适的控制器数目 c 应该在区间 $[0.060n, 0.115n]$。目前得到的结果是一个比较宽的区间,为缩小区间需对实验数

据进行进一步分析。

表 2　$n-c$ 记录

n	c	n	c	n	c	n	c	n	c
50	3	60	6	70	6	80	7	105	11
51	3	61	5	71	5	82	9	110	12
52	4	62	7	72	8	84	7	115	11
53	4	63	7	73	7	86	8	120	9
54	4	64	7	74	7	88	10	125	12
55	5	65	7	75	7	90	9	130	13
56	4	66	6	76	9	93	9	135	13
57	5	67	6	77	8	96	11	140	13
58	6	68	7	78	8	99	6	145	12
59	5	69	7	79	7	100	10	150	13

将表 2 绘制成折线图，如图 5 所示，合适的控制器数目 c 与网络规模 n 之间的统计规律并非简单的线性关系。因此，为缩小最合适控制器数目的区间范围，需对实验数据进行拟合。

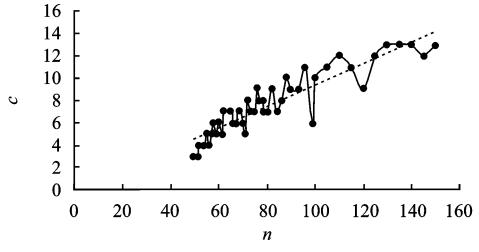


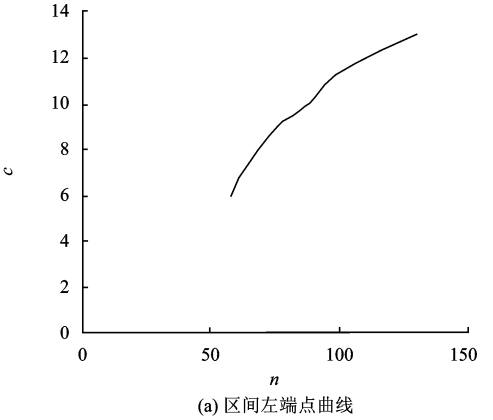
图 5　$n-c$ 记录

以图 5 中的虚趋势线为分割线，将实验数据分割成两组，通过舍弃离群点的方式将这两组数据顺滑成两条曲线，如图 6 所示。拟合这两条曲线的解析表达式分别对应最合适控制器数目的区间左右端点。

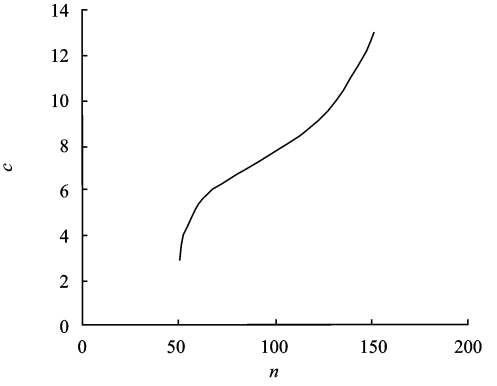
利用 MATLAB 的 Curve Fitting Tool 进行最合适控制器数目 c 与网络规模 n 的非线性回归分析。用多种类型曲线拟合区间端点数据，即用解析表达式逼近离散数据。最后选择拟合最好曲线的解析表达式作为区间端点。残差平方和 SSE 和均方根误差 RMSE 是衡量模型拟合程度的两个量，将其作为选择曲线的依据。

对于区间左端点，共测试了 12 种曲线。其中，SSE 最低的两个是 Fourier 曲线和 Sum of Sine 曲线。二者相比，Fourier 曲线的 SSE 更低，而 Sum of Sine 曲线的 RMSE 更低。考虑到作为区间端点值，选择解析表达式相对简单的 Sum of Sine 曲线。因此，最合适控制器数目的区间左端点为 $13.05 * \sin(0.013\ 17 * n - 0.265\ 5)$。

对于区间右端点，共测试了 12 种曲线，均拟合得不理想，SSE 最低值为 2.67。根据经验发现，若略向下调整部分曲线（区间右端点较精准值稍大），形成的区间右端点曲线与 $y = \sqrt{x}$ 曲线相似，如图 7 所示。因此，对表 2 中所有实验数据进行归纳分析，发现最合适控制器数目的区间右端点为 $1.15 * \sqrt{n}$。



(a) 区间左端点曲线



(b) 区间右端点曲线

图 6　区间端点曲线

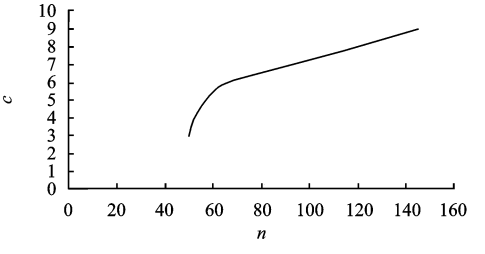


图 7　调整后的区间右端点曲线

分析实验结果，得到一个相对可靠的结论：当考虑请求响应延迟与控制器可靠性时，设 n 为网络节点的数目，最合适的控制器数目应该在区间 $[0.060n, 0.115n]$，最优区间为 $[13.05 * \sin(0.013\ 17n - 0.265\ 5), 1.15 * \sqrt{n}]$。

3　结论

对 SDN 多控制器部署方案进行了研究，解决了部署位置和部署数目两个必要的问题。首先，设计并实现了高可靠性且低延迟的多控制器部署位置问题求解算法，通过对比实验证明该算法得到的位置部

署方案在可靠性和延迟两个性能指标上优于 PCR1 和 PCR2 算法;然后,基于 MCP 进行了大量实验,测试了各种网络规模的最合适控制器数目,归纳出最合适控制器数目与网络规模之间的统计规律。

参考文献:

[1] TOOTOONCHIAN A, GANJALI Y. HyperFlow: A distributed control plane for OpenFlow[C]//Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking. New York: ACM, 2010:1-3.

[2] Open Networking Foundation. The Openflow Switch[EB/OL]. [2014-08-15]. http://www.opennetworking.org/sdn-resources/openflow-switch-specification.

[3] ZHANG J J, YE M H, GUO Z H, et al. CFR-RL: Traffic engineering with reinforcement learning in SDN[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(10): 2249-2259.

[4] GUO Z H, DOU S S, LIU S, et al. Maintaining control resiliency and flow programmability in software-defined WANs during controller failures[J]. IEEE/ACM Transactions on Networking, 2022, 30(3): 969-984.

[5] TAVAKOLI A, CASADO M, KOPONEN T, et al. Applying nox to the datacenter[C]//Proceeding of the HotNets. New York: ACM, 2009.

[6] PARK J, YOON W. SDN-based heterogeneous network architecture with multi-controllers [C]//2020 22nd International Conference on Advanced Communication Technology (ICACT). Phoenix Park: IEEE, 2020:559-561.

[7] YEGANEH S H, GANJALI Y. Kandoo: A framework for efficient and scalable offloading of control applications[C]//Proceedings of the First Workshop on Hot Topics in Software Defined Networks. Helsinki: ACM, 2012:19-24.

[8] LI J Y, LIU J, GAO Q, et al. The decision latency optimization problem in SDN with multi-controller[J]. IEEE Communications Letters, 2019, 23(12): 2344-2347.

[9] LI K, OU Q H, CHEN H, et al. Controller cluster-based interconnecting for multi-domain SDN networks[C]//2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC). Nanjing: IEEE, 2017:482-485.

[10] FONSECA P, BENNESBY R, MOTA E, et al. A replication component for resilient OpenFlow-based networking[C]//2012 IEEE Network Operations and Management Symposium. Maui: IEEE, 2012:933-939.

[11] ZHI L Y, MOHAN P M, SRIDHARAN V, et al. Secondary controller mapping for reliable control traffic forwarding in SDN[C]//2018 27th International Conference on Computer Communication and Networks (ICCCN). Hangzhou: IEEE, 2018:1-2.

[12] HU Y N, WANG W D, GONG X Y, et al. Reliability-aware controller placement for software-defined networks[C]//2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013). Ghent: IEEE, 2013:672-675.

[13] 赵文文, 孟相如, 康巧燕, 等. 时延和可靠性感知的多控制器均衡部署策略[J]. 空军工程大学学报(自然科学版), 2021, 22(4): 85-91.
ZHAO Wenwen, MENG Xiangru, KANG Qiaoyan, et al. A delay and reliability aware multi-controller balancing deployment strategy[J]. Journal of Air Force Engineering University (Natural Science Edition), 2021, 22(4): 85-91.

[14] FAN Y Q, OUYANG T. Reliability-aware controller placements in software defined networks[C]//2019 IEEE 21st International Conference on High Performance Computing and Communications. Zhangjiajie: IEEE, 2019:2133-2140.

(责任编辑:冯容辉)