DOI:10.3969/j.issn.1003-5060.2024.05.008

软件定义网络中操作成本优化的控制路径恢复

史久根，　黄飞黄，　徐　强
(合肥工业大学 计算机与信息学院,安徽 合肥　230601)

摘　要:软件定义网络(software defined network,SDN)是一种新兴的网络范式,其解耦了控制平面和数据平面,实现了网络管理的灵活性。然而,控制平面的逻辑集中化带来了新的挑战,即在各种故障情况下确保较高的可用性。文章在带内控制平面的前提下,提出一种控制平面恢复方法,该方法重复利用旧的控制路径规则,并建立相应的数学模型;最终提出一种考虑复用控制规则的树形恢复算法,并证明当控制路径以最短路径树的形式部署时,算法的近似度为 3。实验结果表明,该方法具有较高的有效性。

关键词:软件定义网络;控制平面;带内控制平面;控制路径;故障恢复
中图分类号:TP393　　**文献标志码**:A　　**文章编号**:1003-5060(2024)05-0628-07

Control path recovery for operation cost optimization in software defined network

SHI Jiugen,　HUANG Feihuang,　XU Qiang
(School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China)

Abstract: Software defined network(SDN) is a new network paradigm. It separates the control plane from the data plane and realizes flexible network management. However, the logical centralization of the control plane brings a new challenge to the network stability, that is, how to achieve high availability in various fault situations. On the premise of in-band control plane, this paper explores a control plane recovery method that reuses the old control path flow rules to reduce recovery operations, and puts forward a mathematical model. The paper ultimately proposes a tree-based recovery algorithm that considers the reuse of control rules, and proves that the algorithm has an approximation ratio of 3 when the control path is deployed in the form of a shortest path tree. The experimental and simulation results demonstrate the effectiveness of the proposed method.

Key words: software defined network(SDN); control plane; in-band control plane; control path; fault recovery

0　引　　言

近年来,新兴网络技术软件定义网络(software defined network,SDN)的出现给网络的管理和部署带来了更多的灵活性。由于其具有全局的网络视图,相比传统的网络更具有全局且灵活的掌控权[1],然而控制平面的逻辑集中在提供网络管理灵活性的同时也使保持网络的稳定性成为了挑战。由于控制平面是逻辑集中式的,当控制平面的某部分出现损坏时,可能会造成控制平面整体的故障[2],从而导致数据平面的组件无法连接到控制平面,出现流转发错误[3]。控制路径是交换机联系控制平面的信道,每个交换机都必须通过控制路径连接到控制器,否则会造成不可预知的后果[4]。

控制平面的形式主要分为带外[5]控制平面和

收稿日期:2023-02-17;**修回日期**:2023-03-24
基金项目:国家重大科学仪器设备开发专项资助项目(2013YQ030595)
作者简介:史久根(1963—),男,安徽桐城人,博士,合肥工业大学副教授,硕士生导师.

带内[6]控制平面。本文研究的关注点为带内控制平面的恢复机制。

目前关于控制平面稳健性的研究有许多。文献[7-9]主要研究控制器的初始部署与交换机控制器的重新配对，这些研究很少考虑交换机在故障发生时如何重建控制路径来让交换机重新分配至控制器，而其恢复方案也几乎没有考虑到操作成本、恢复的时间复杂度，从而导致了这些方法在实际的场景中恢复得较慢，且对整个网络造成较大的负担。

文献[10-13]则更进一步通过考虑控制路径来完善控制平面的容错与恢复机制。其中有的研究主要利用备用控制路径的方法，当故障发生时切换到可用的控制路径，虽然可以加快恢复速度，但是会占用大量的交换机流表资源，同时也不够灵活。另一些研究虽然从细节上考虑了在恢复时重建控制路径，但是没有进行详细的数学建模。除此之外，这些恢复方法也没有考虑最小化恢复过程中的操作成本。

本文首先描述了一个新的控制平面恢复问题，在进行控制路径恢复时考虑对旧路径规则的复用，并建立了相应的数学模型。本文提出一种基于生成树的控制路径部署和恢复算法，并证明了当控制路径以最短路径树的形式初始部署时，算法的近似度为 3。最后通过实验和模拟验证了本文方法的有效性。

1　问题描述和数学模型

1.1　问题描述

在软件定义网络中，每个交换机都需要通过一个单独的控制路径连接到控制器。

由于本文考虑的控制平面是带内平面，当链路损坏时可能会同时带来控制路径的损坏，这时需要进行恢复操作。

带内控制平面的恢复示例如图 1 所示，控制器节点 c 到 a 的控制路径为 $c\rightarrow a$，到 b 的控制路径为 $c\rightarrow b$，到 d 的控制路径为 $c\rightarrow d$，到 e 的控制路径为 $c\rightarrow e$，到 f 的控制路径为 $c\rightarrow e\rightarrow f$。当从 c 到 a 的控制路径由于链路 (a,c) 的损坏而中断时，从 c 到 a 的控制路径需要进行重建，这时需在节点 c、b、a 上分别进行流表更改操作。在控制路径恢复过程中，本文使用文献[11]的技术，通过事先布置相关规则使受控交换机可以实现与相邻交换机的临时通信，如图 1 中节点 b 和节点 f 的控制路径均未中断，因此可以利用 b 或 f，建立一条 a 与 c 的临时通信，让节点 c 往 a 部署流规则。

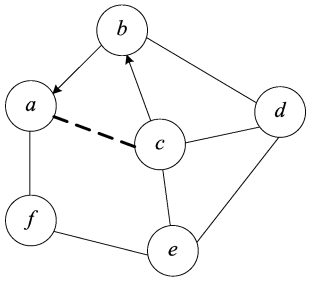


图 1　带内控制平观的恢复示例

本文将具有一条完整的控制路径的交换机称为完全受控交换机，而将完全受控交换机的相邻交换机称为不完全受控交换机。若不是前两者，则称之为未受控交换机。

数学模型的建立主要存在 2 个问题：

1）控制路径恢复时间建模。本文通过恢复阶段数[14]表示控制路径恢复的时间，控制器在一个阶段上可以在完全受控交换机和不完全受控交换机上部署规则。很显然，阶段数与恢复时间成正相关，阶段数越多则恢复时间越长。

2）控制路径恢复代价建模。本文利用计算控制路径规则的安装数量来衡量控制路径恢复代价。

1.2　数学模型和分析

本文用 $G(N,E)$ 表示链路故障发生前的网络拓扑，其中：N 表示交换机的集合；E 表示物理链路的集合。在发生链路故障后，整体的网络拓扑发生改变，改变后的网络拓扑用 $G'(N,E')$ 表示，其中，E' 表示发生链路故障之后的物理链路集合。本文使用 v_c 表示控制器所放置的交换机位置。对于链路故障，用 $e_{u,v}\in F$ 表示链路错误的集合。

本文考虑带内控制平面，因此除了控制器所在的交换机外的每个交换机都必须有一个连通的控制路径连接控制器，本文使用 p_i 表示链路错误发生前交换机 i 的控制路径，使用 q_i 表示错误发生后交换机 i 的控制路径。控制路径建模为有向的路径。使用二进制变量 $x^i_{u,v}$ 表示控制路径 p_i 经过链路 $\langle u,v\rangle$，使用二进制变量 $y^i_{u,v}$ 表示控制路径 q_i 经过链路 $\langle u,v\rangle$。

控制路径从控制器所在的节点开始，在相应的交换机处结束，具体计算公式如下：

$$\sum_{u\in N}x^i_{v_c,u}-\sum_{u\in N}x^i_{u,v_c}=1,\quad i\in N-v_c \tag{1}$$

$$\sum_{u\in N}x^i_{u,i}-\sum_{u\in N}x^i_{i,u}=1,\quad i\in N-v_c \tag{2}$$

$$\sum_{u\in N}y^i_{v_c,u}-\sum_{u\in N}y^i_{u,v_c}=1,\quad i\in N-v_c \tag{3}$$

$$\sum_{u\in N}y^i_{u,i}-\sum_{u\in N}y^i_{i,u}=1,\quad i\in N-v_c \tag{4}$$

控制路径还要是连续的,于是有式(5)、式(6)的限制,即

$$\sum_{u\in N}x^i_{u,v}-\sum_{u\in N}x^i_{v,u}=0,\quad v\neq i,v\neq v_c,i\in N-v_c \tag{5}$$

$$\sum_{u\in N}y^i_{u,v}-\sum_{u\in N}y^i_{v,u}=0,\quad v\neq i,v\neq v_c,i\in N-v_c \tag{6}$$

恢复后的控制路径不可以经过损坏的物理链路,于是有:

$$\sum_{e_{u,v}\in F}y^i_{u,v}+\sum_{e_{u,v}\in F}y^i_{v,u}=0,\quad i\in N-v_c \tag{7}$$

恢复阶段数越少则整个恢复的时间也越短。在路径约束下,控制路径的规则更新最少需要 0 轮,最多只需要 r 轮完成。设置二进制变量 $b^i_{u,v}$ 表示是否需要为链路$\langle u,v\rangle$新增规则:

$$b^i_{u,v}=\begin{cases}1, & y^i_{u,v}-x^i_{u,v}=1;\\ 0, & \text{其他}\end{cases} \tag{8}$$

其中,$i\in N-v_c$。

设置变量 B^i_v 计算节点 v 是否需要为控制路径 q_i 建立新规则:

$$B^i_v=\begin{cases}1, & \sum_{u\in N}b^i_{v,u}>0 \text{ 或} \sum_{v\in N}b^i_{u,v}>0;\\ 0, & \text{其他}\end{cases} \tag{9}$$

其中,$i\in N-v_c$。

下面二进制变量用于表示在第 j 轮是否在节点 v 上放置 i 的规则:

$$\hat{b}^i_{v,j}=\{0,1\} \tag{10}$$

设二进制变量 $\alpha_{i,j}$ 表示交换机的控制路径所需规则是否已经在阶段 j 部署完毕,其中,N_f 表示控制链路因链路故障而损害的交换机的集合。$\alpha_{i,j}$ 计算公式为:

$$\alpha_{i,j}=\begin{cases}1, & \sum_{j-1}\hat{b}^i_{v,j}=B^i_v \text{且} i\in N_f;\\ 1, & i\notin N_f;\\ 0, & \text{其他}\end{cases} \tag{11}$$

必须要在 r 阶段完成所有恢复规则的部署,具体如下:

$$\sum_{j=0}^{r}\hat{b}^i_{v,j}=B^i_v,\quad i\in N_f \tag{12}$$

设置二进制变量 $Z_{i,j}$,$i\in N$ 表示在 j 阶段交换机 i 能否直接或者间接受控。只有当交换机 i 直接受控,即 $\alpha_{i,j}=1$,或者是交换机 i 有邻居处于直接受控状态,即 $\sum_{v\in N}g_{i,v}\alpha_{v,j}\geqslant 0$,才能将其设置为 1。其中,变量 $g_{u,v}$ 表示 u 和 v 是否是邻接节点。$Z_{i,j}$ 计算公式为:

$$Z_{i,j}=\begin{cases}1, & \alpha_{i,j}=1 \text{或} 1,\ \sum_{v\in N}g_{i,v}\alpha_{v,j}\geqslant 0;\\ 0, & \text{其他}\end{cases} \tag{13}$$

上一轮过后是直接或者间接受控的节点上部署的规则如下:

$$\hat{b}^i_{v,j}\leqslant Z_{i,j-1},i\in N_f \tag{14}$$

只在没部署过流规则的节点部署流规则如下:

$$\hat{b}^i_{v,j}\leqslant 1-\sum_{m=0}^{j-1}b^i_{v,m},\quad i\in N_f \tag{15}$$

是否进行第 j 轮更新,当全部规则都已经部署则不需要下一轮更新,判断依据为:

$$\hat{u}_j=\begin{cases}1, & \sum_{m=0}^{j}\hat{b}^i_{v,j}=B^i_v \text{且} i\in N_f;\\ 0, & \text{其他}\end{cases} \tag{16}$$

保证更新轮数不超过限制轮数 K,具体公式为:

$$\sum_{j=0}^{r}\hat{u}_j\leqslant K \tag{17}$$

为了在恢复时减小瞬时的恢复流量,本文以最小化新部署的规则为目标,即

$$\min\sum_{i\in N-v_c}\sum_{v\in N}B^i_v$$
$$\text{s.t. 式(1)}\sim\text{(17)} \tag{18}$$

本问题实际上是具有多个限制条件的流重配置问题,若不考虑轮数限制,则可以将问题简化为文献[15]中的拥塞流重配置问题,而其已经被证明即使在有向无环图中也是 NP-Hard 的,因此本问题的复杂程度也为 NP-Hard。

2 基于树的控制路径部署及恢复算法

由于本问题是 NP-Hard 问题,本文提出一个基于树重构的算法来解决此问题,该算法分为控制路径部署和控制路径恢复 2 个部分。

1) 控制路径部署。本文在路径初始部署时采用 Dijistra 最短路径算法计算从控制器到各个交换机的路径,然后沿着这些路径部署初始的控制路径规则。由于控制路径规则的方向性,这些有向路径可以汇聚成树的形式,称之为最短路径树,用 t_m 表示,控制器所在节点为 t_m 的根节点。

2) 部署时预计算。在故障发生后计算恢复

方案应当尽量快，因此在部署时计算辅助矩阵 $\boldsymbol{M}$ 以帮助在恢复时进行快速计算，其中 $\boldsymbol{M}[i][j]$ 表示树 t_m 节点的 i 和 j 节点到根节点的最短不相交路径。算法 1 描述如下：

输入：t_m

输出：返回协助矩阵 $\boldsymbol{M}$

初始化协助矩阵 $\boldsymbol{M}$，计算每个节点 n 自身到根节点的路径，存入 $\boldsymbol{M}[n][n]$ 中

for each n in t_m do

计算 n 到根节点的路径 p_n

for each j in $t_m - n$ do

计算 j 到根节点的路径 p_j

计算 p_n 和 p_i 最短不相交路径为 $p_{n,i}$

将 $p_{n,i}$ 存入 $\boldsymbol{M}$ 中

end for

end for

3）树重构恢复算法。利用初始部署时控制路径的树形特性，计算控制路径的恢复方案。由于控制路径是以最短路径树形式进行部署的，一旦控制路径上有链路中断，则 t_m 会被划分为 2 棵树，其中，一棵有控制器存在的树仍然称为 t_m，而另一颗则被称作子树，将之放入到集合 T 中。若有多个控制路径中断，则会有多个子树被放置到集合 T 中。子树在 T 中按照先后产生顺序排序，$\boldsymbol{J}$ 矩阵存储子树在损坏前所挂接的位置。算法 2 描述如下：

输入：$t_m, T, G_f, \boldsymbol{J}, \boldsymbol{M}$

输出：R

for each t in T do

设置集合 S 为空集

找到每一个节点对，使得该节点对中一个节点位于主树 t_m 里，另一个在子树中，且这 2 个节点是邻居节点。将节点对存入集合 S 中

if S 为空 then

终止算法

end if

for each n, j in S do

计算对子树 t 在节点 n 位置做旋转操作时的旋转代价 c_r 和受控比例系数 α

从辅助矩阵中获取连接路径为 $p=\boldsymbol{M}[j][\boldsymbol{J}[t]]$

计算对子树 t 在节点 n 位置做旋转操作时连接成本 c_p，c_p 为 p 的路径长度乘以子树的节点数量

计算总成本：

$$c = \frac{c_p + c_r}{\alpha\rho} \tag{19}$$

end for

找到集合 S 中总成本最低的节点对 (n_b, j_b)

记录节点对 (n_b, j_b) 到 R 中，对子树在节点 n_b 处做旋转操作，并连接到节点 j_b

end for

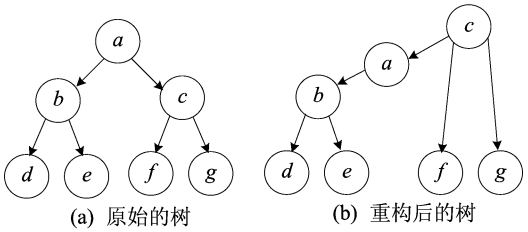
得到最终的恢复方案 R

子树的重构过程如图 2 所示，节点 a 本来为子树的根节点，现在子树的根节点与主树 t_m 的连接断开，此时选定 c 作为子树的新根并变换有向边 $a \rightarrow c$ 为 $c \rightarrow a$ 就可以形成以 c 为新根的子树。

为了优化恢复阶段数，节点 a 和节点 c 都应为间接受控节点或直接受控节点。用 α 表示从新根到旧根路径上的所有节点的直接或间接受控节点比例，其值越大越好，从而优化恢复阶段数，α 计算公式为：

$$\alpha = \frac{N_{\text{controlled}}}{N_{\text{total}}} + 1 \tag{20}$$

重构子树会有重构开销 c_r，在图 2 中，共需要 5 次操作。



(a) 原始的树　　(b) 重构后的树

图 2　子树的重构

为了在最小化操作成本的前提下让恢复阶段数尽可能小，式(19)定义路径恢复的总成本。新加入比例因子 ρ 是为了调整 α 和 $c_p + c_r$ 之间的权衡关系。

部署时预计算的最坏时间复杂度为 $O(N^3)$；权重构恢复算法对每一个因链路错误而与主树断开的子树进行恢复操作，设 T 中子树的数量为 M，则其最坏的时间复杂度为 $O(MN^2)$。下面证明当控制路径以最短路径树的形式进行初始部署时，该算法的近似度为 3。

算法的近似度证明如图 3 所示，r 是控制器节点，由于新布置的流规则的数量与路径的长度呈正比，只考虑路径长度。假设 s 的控制路径在 p 和 q 之间断开，那么 s 需要建立一条新的到 r 的控制路径，最优的方法是新路径 $s \rightarrow p \rightarrow q \rightarrow t \rightarrow r$（其中 p 到 q 为新的最短路径）。

当使用算法 2 计算新的控制路径，且子树不进行旋转操作，设 s 重新连接到 r 的代价是 $W(s)$，$w_{i,j}$ 表示在考虑重用流规则时从 i 到 j 建立新路径的最小代价，则有：

$$W(s) = \min(w_{s,p} + w_{p,q_i}) \leqslant w_{s,p} +$$

$$w_{p,q} \leqslant d_G(s,p) + d_{G-e_f}(p,q) \leqslant d_{G-e_f}(s,p) + d_{G-e_f}(p,q) + d_{G-e_f}(q,t),$$

其中:d 表示最短路径;$G-e_f$ 表示在链路损坏条件下新的拓扑。

根据文献[13],可知:

$$d_{G-e_f}(s,p) + d_{G-e_f}(p,q) + d_{G-e_f}(q,t) \leqslant 3d_G(s,t),$$

则有:

$$W(s) \leqslant 3d_G(s,t) \leqslant 3w_{s,t}。$$

由于在算法中可以选择是否旋转子树,主要取决于成本的比较,设总代价为 W,则有:

$$W(s) \leqslant \sum_{s_i \in t_f} W(s_i) \leqslant 3\sum_{s_i \in t_f} w_{s_i,t_i}。$$

因此,当控制路径以最短路径树的形式部署时,该算法的近似度为 3。

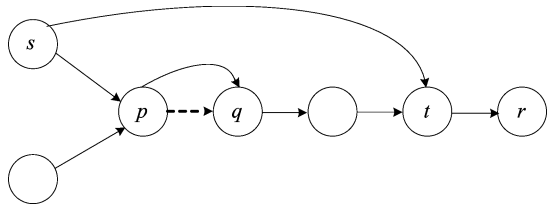


图 3 算法的近似度证明

3 实验及分析

3.1 拓扑分析

本文从文献[16]中选取了 2 种实验拓扑,分别是 ATT North America 拓扑(ATT)和 DFN 拓扑,拓扑的参数见表 1 所列。

表 1 拓扑参数

拓扑名	节点数	边数	平均节点度	损坏链路数		
				1	2	3
ATT	25	56	4.48	56	1 540	27 720
DFN	51	8	3.14	8	3 160	82 160

对 2 种拓扑的 3 种链路损坏情形进行分析,即一次只损坏 1 条物理链路、一次同时损坏 2 条链路、一次同时损坏 3 条链路的情况。损坏情形数量见表 1 所列。

3.2 对比算法和指标

本文的算法在考虑重用路径规则基础上的最短路径的控制路径部署和树重构恢复组合方法,称之为 SP+RCTR 方法。本次实验的对比方法有 3 种。

1) 最短路径的控制路径部署和传统最短路径恢复的组合方法,该方法的控制路径部署和本文的方法相同,但是其恢复方式只是简单地计算恢复的最短路径,然后沿着路径的所有节点部署新的控制路径规则,用 SP+Traditon 表示。

2) 采用文献[12]中的控制路径部署方法,而它的恢复方式仍然采用传统最短路径方式进行恢复,用 DR+Tradition 表示。

3) 控制路径部署方法同第 2 种,但是在控制路径恢复时考虑了控制路径的重用,主要是以尽量短的距离绕开发生链路错误的地方,之前部署过的规则不再重新部署,用 DR+RCSP 表示。

实验采用以下 2 个指标:第 1 个指标是控制器为了恢复控制路径所需要的操作数量,本文将为恢复单个交换机的控制路径而在一个节点上部署流规则的行为称作一次操作,本文为每种错误情况进行实验,并计算平均的操作数量;第 2 个指标为在文献[14]中的恢复阶段数,该数越小意味着总恢复时间也越小。同样本文也为每种错误情况实验确定其恢复阶段数,然后计算其平均的恢复阶段数。

3.3 实验结果

首先在美国的 ATT 拓扑中进行实验,对错误链路数为 1、2、3 的所有错误情况进行模拟,实验结果如图 4 所示。

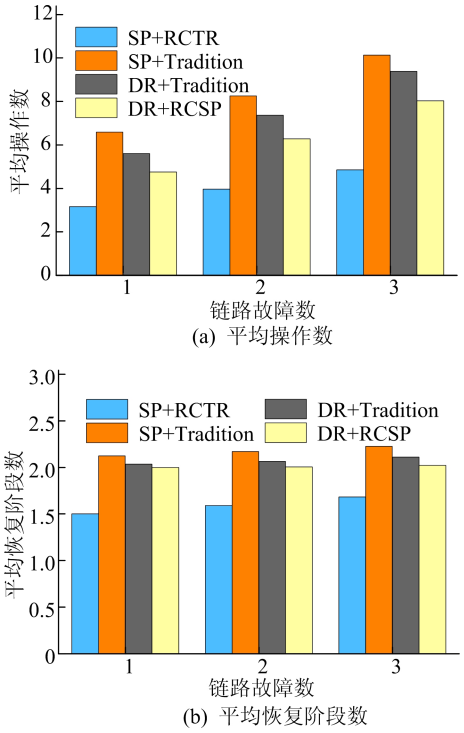


图 4 ATT 拓扑中的性能指标

由图 4a 可知,本文提出的 SP+RCTR 具有

最小的操作次数，大大小于其他几种对比的恢复方案；DR＋Tradition 相比于 SP＋Traditon 具有较低的操作次数，表明控制路径的初始放置方案对平均操作次数具有重要影响；DR＋RCSP 比 DR＋Tradition 方案的平均操作次数更低，表明考虑规则重用可以减少控制路径恢复过程中的操作次数。

由图 4b 可知：本文所提出的 SP＋RCTR 具有最小的恢复阶段数，而 SP＋Tradition 则具有最大的恢复阶段数；DR＋Tradtion 相比于 SP＋Tradtion 具有较低的恢复阶段数，表明控制路径的放置方案可以影响到恢复的时间；DR＋RCSP 比 DR＋Tradition 方案具有更低的恢复阶段数，表明考虑规则重用同样可以降低恢复阶段数，降低恢复的总时间。

在德国的 DFN 拓扑中进行实验的模拟结果如图 5 所示。

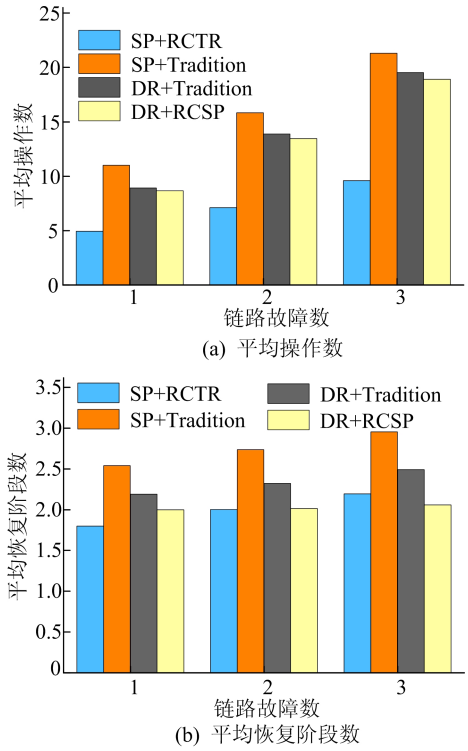


图 5　DFN 拓扑中的性能指标

由图 5 可知，在平均操作数的指标上，情况与在 ATT 拓扑中基本相似。本文提出的 SP＋RCTR 具有最小的平均操作次数，而 SP＋Tradition 具有最大的平均操作次数。然而就平均恢复阶段数情况却有些不同：本文所提出的方案并不总是最优的。这主要是由于 DFN 相较 ATT 有着较低的平均节点度，而本文所提的树重构恢复在进行按阶段部署路径规则时相较 DR＋RCSP 有更少的完全控制交换机，使得整体的恢复阶段数比 DR＋RCSP 有所增加。

4　结　　论

快速低代价地控制路径恢复对提升 SDN 网络的稳定性具有重要的意义。本文通过在带内软件定义网络中考虑重用控制路径规则，提出了一种树重构控制路径恢复方法。该方法在有效减少恢复操作成本的同时，还加快了控制路径的恢复速度。实验结果表明，本文所提出的方法相较其他方法，大大降低了操作成本。在恢复时间这一指标上，本文提出的方法在拓扑的平均节点度较高时拥有最低的恢复时间，而当平均节点度较高时，本文所提方法的恢复时间虽然不是最优的，但是也显著优于传统的恢复方法。

［参　考　文　献］

［1］ ANEROUSIS N，CHEMOUIL P，LAZAR A A，et al. The origin and evolution of open programmable networks and SDN［J］. IEEE Communications Surveys & Tutorials，2021，23（3）：1956-1971.

［2］ VIZARRETA P，TRIVEDI K，MENDIRATTA V，et al. DASON：dependability assessment framework for imperfect distributed SDN implementations［J］. IEEE Transactions on Network and Service Management，2020，17（2）：652-667.

［3］ LEE S S W，LI K Y，CHAN K Y，et al. Software-based fast failure recovery for resilient OpenFlow networks［C］// 2015 7th International Workshop on Reliable Networks Design and Modeling.［S. l.：s. n.］，2015：194-200.

［4］ SONG S J，PARK H，CHOI B Y，et al. Control path management framework for enhancing software-defined network（sdn）reliability［J］. IEEE Transactions on Network and Service Management，2017，14（2）：302-316.

［5］ KATTA N，ZHANG H，FREEDMAN M，et al. Ravana：controller fault-tolerance in software-defined networking［C］//Proc ACM SOSR. Santa Clara：ACM，2015：1352-1365.

［6］ PANDA A，ZHENG W，HU X H，et al. SCL：simplifying distributed SDN control planes［C］//Proc 14th USENIX NSDI. Berkeley：USENIX Association，2017：329-345.

［7］ YANG S，CUI L Z，CHEN Z T，et al. An efficient approach to robust SDN controller placement for security［J］. IEEE Transactions on Network and Service Management，2020，17（3）：1669-1682.

［8］ XIE J X，GUO D K，ZHU X M，et al. Minimal fault-tolerant coverage of controllers in iaas datacenters［J］. IEEE Transactions on Services Computing，2020，13（6）：1128-1141.

[9] WANG T, CHEN H C. Optimal model for link failure foresight controller placement in SDN [C]//2021 IEEE 4th International Conference on Electronics Technology. Chengdu: IEEE, 2021: 727-730.

[10] SHARMA S, STAESSENS D, COLLE D, et al. Fast failure recovery for in-band OpenFlow networks [C]//2013 9th International Conference on the Design of Reliable Communication Networks (DRCN). [S. l. : s. n.], 2013: 52-59.

[11] CANINI M, SALEM I, SCHIFF L, et al. Renaissance: a self-stabilizing distributed SDN control plane [C]//2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). Vienna: IEEE, 2018: 233-243.

[12] WANG N Y, CHEN D, LIU L, et al. An SDN based highly reliable in-band control framework for leo mega-constellations [C]//2021 IEEE 6th International Conference on Computer and Communication Systems. Chengdu: IEEE, 2021: 970-975.

[13] DUAN T, DINAVAH V. Dataplane-based fast failover in SDN-enabled wide area measurement system of smart grid [J]. IEEE Transactions on Industrial Informatics, 2023, 19(7): 8148-8158.

[14] SAVAS S S, TORNATORE M, DIKBIYIK F, et al. RASCAR: recovery-aware switch-controller assignment and routing in S-DN [J]. IEEE Transactions on Network and Service Management, 2018, 15(4): 1222-1234.

[15] AMIRI S A, DUDYCZ S, SCHMID S, et al. Congestion-free rerouting of flows on DAGs [C]//45th International Colloquium on Automata, Languages, and Programming. [S. l. : s. n.], 2018: 2-13.

[16] 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.

(责任编辑 李 凯)

(上接第619页)

定,电流无越幅且逆变器能够为电网提供无功支撑;针对故障恢复后的电网相角偏移所带来的功率恢复缓慢问题,通过提出基于虚拟功率的相角跟踪方法并优化控制参数,实现逆变器输出功率恢复速度进一步加快,故障恢复模式平滑快速切换至正常运行模式,实现了基于VSG控制并网逆变器的低电压穿越。

[参 考 文 献]

[1] 施凯,叶海涵,徐培凤,等. 基于欠励磁状态运行的虚拟同步发电机低电压穿越控制策略[J]. 电力系统自动化,2018,42(9):134-140.

[2] 胡文强,吴在军,窦晓波,等. 负荷虚拟同步机及其低电压故障穿越控制[J]. 电力系统自动化,2018,42(9):100-107.

[3] D'ARCO S, SUUL A J, FOSSO O B. Automatic tuning of cascaded controllers for power converters using eigenvalue parametric sensitivities[J]. IEEE Transactions on Industry Applications, 2015, 51(2): 1743-1753.

[4] 谭骞,徐永海,黄浩,等. 对称电压暂降情况下光伏逆变器输出电流峰值的控制策略[J]. 电网技术,2015,39(3):601-608.

[5] 陈天一,陈来军,郑天文,等. 基于模式平滑切换的虚拟同步发电机低电压穿越控制方法[J]. 电网技术,2016,40(7):2134-2140.

[6] JONGUDOMKARN J, LIU J, ISE T. Virtual synchronous generator control with reliable fault ride-through ability: a solution based on finite-set model predictive control[J]. IEEE Journal of Emerging and Selected Topics in Power Electronics, 2020, 8(4): 3811-3824.

[7] 赵红雁,郑琼林,李艳,等. 应用于三相并网系统的电网电压快速锁相技术研究[J]. 高电压技术,2018,44(1):314-320.

[8] 房志学,苏建徽,王华锋,等. 微网逆变器低电压穿越控制策略[J]. 电力系统自动化,2019,43(2):143-155,161.

[9] 贾利虎,朱永强,孙小燕,等. 基于模型电流预测控制的光伏电站低电压穿越控制方法[J]. 电力系统自动化,2015,39(7):68.

[10] 高怀正,李华,常兴,等. 电压跌落下虚拟同步发电机故障穿越控制[J]. 电力系统保护与控制,2018,46(17):39-46.

[11] 郑天文,陈来军,陈天一,等. 虚拟同步发电机技术及展望[J]. 电力系统自动化,2015,39(21):165-175.

[12] 李旭枫,陆立民,成乐祥,等. 基于自适应虚拟阻抗改进无功环路的虚拟同步功率解耦控制策略[J]. 电网技术,2019,43(10):3752-3760.

[13] 李光琦. 电力系统暂态分析[M]. 北京:中国电力出版社,2007:204-209.

[14] 施向前,耿乙文,黄尊臣,等. 光伏电站低电压穿越技术研究[J]. 电测与仪表,2015,52(8):61-66.

[15] 孔祥平,张哲,袁宇波. 低电压穿越控制策略下双馈风电机组故障电流特性的研究[J]. 电测与仪表,2016,53(20):67-74.

(责任编辑 张 锴)