Towards robust controller placement in software-defined networks against links failure

Li Li, Nana Du, Huanyu Liu, Ruifang Zhang
School of Computer Science, Shaanxi Normal University, Xi'an, China
{lili, duna, liuhy, zhangruifang}@snnu.edu.cn

Chaobo Yan
School of Electronic and Information Engineering, Xi'an Jiaotong Univercity
IEEE, Senior Member, Xi'an, China
chaoboyan@mail.xjtu.edu.cn

***Abstract*—To further improve software-defined network performance, robustness and efficiency, it is valuable to determine how to optimally deploy controllers against links failure, e.g., fiber cuts, operation failure, etc. In this paper, we investigate the problem of robust controller placements against any *k* links failure (*k*-RCP) optimization with network delay and load balancing. The *k*-RCP optimization problem is established an integer linear programming which optimally places the least controllers to meet the controlled coverage probability against *k*-links failure. To solve the *k*-RCP problem, we develop a *k*-RCP method based on the *k*-RCP linear programming and its dual programming model. Analysis showed that *k*-RCP method provides an approximately optimal solution under the given controlled coverage probability. Simulation results showed that the *k*-RCP method effectively improves the SDN robustness when considering network delay and load balancing.**

Keywords—robust controller placement, software-defined network, limited controller resource, network delay and load balancing

I. Introduction

Software-Defined Networks (SDN) provides a separation between the control and data planes. The separation simplifies the networking management and improves its scalability [1]. Scalability and survivability should be more carefully considered in SDN networks. Since network failures could cause disconnections between the control and data planes, and further disable some of the switches, it is of great importance to improve the robustness of SDN control networks [2].

The controller in charge of a network system is a logically centralized manner, and may become a choke point, especially in a large scale network. Therefore, multiple controllers which are physically distributed throughout the network are required. The propagation latency between the switches and the associated controller is a key point for the performance of SDN networks, and the placement of multiple controllers becomes a critical problem. To further improve network performance, including latency and robustness, operators should carefully place the controller [3–5]. As discussed in some related work, the controller placement is a complexity optimization problem [5-6]. Currently, the most well-known controller placement method, which is introduced in [4], develop the best controller placement solutions that minimize the controller and the switch network latency, which includes the Average-case latency and the Worst-case latency (or maximum latency). A good placement should minimize the network latency, whereas the load of each controller should not exceed its capacity [5], [7–9].

Generally, traditional optimization based on network latency ignores network failures. Unfortunately, network failures frequently occur. Some simulation results show that the worst-case latency changes within a wide range in case of failures and the minimum worst case latency hardly acquired when links failure happen from another point of view. Overall, existing optimization methods have limited abilities to strike a balance between network delay and controller load against links failure. Whereas it is still a challenging task in optimally place controllers against links failure, such that the resulting SDN is the most robust and efficient.

In this paper, the robust controller placement (*k*-RCP) against *k*-links failure with network delay and load balancing is considered. Given a physical network and the controlled coverage requirement, how many controllers are needed and how to deploy them such that the resulting SDN is the most robust and efficient. This novel concept, called controlled proportion (vs. complete control of the control plane) is defined to characterize controlled coverage probability against *k*-links failure due to the following reason: it is infeasible after *k*-links failure (when *k* is a large integer) that all switches in data plane can connect any one controller, especially in the disconnected SDN.

For multiple controllers placement problem, optimizing each goal is generally NP-hard. When more than one goal is considered at the same time, the situation is more complicated. To resolve the *k*-RCP optimization problem, the delay of control signaling, the load balancing of controllers, the least number of controllers, and *k*-links failure tolerance will be taken into consideration in our proposed *k*-RCP method. The main contributions of this paper are given as follows.

1) We first introduce the robust controller placement against any *k*-links failure (*k*-RCP) optimization problem that consider not only network delay but also load balancing.
2) The effective *k*-RCP method is proposed based on the linear programming model of *k*-RCP and its dual programming model to resolve the *k*-RCP optimization problem.
3) The *k*-RCP algorithm is the approximate optimal, which significantly improves the robustness and the efficiency of SDN against links failure. Our simulation results reveal our method is better than the state-of-the-art methods.

This research is supported in part by the National Natural Science Foundation 61303092 and 61702317, Ministry of Education Key Laboratory for Modern Teaching Technology, and the Fundamental Research Funds for the Central Universities of China GK201801004.

The rest of the paper is organized as follows. In section II, we review the related works. The problem is formally articulated and the evaluation metrics are discussed in section □. In section □, the *k*-RCP optimization models are established. And the *k*-RCP algorithm is described in details. In section □, the effectiveness of the *k*-RCP method is demonstrated by experimental results. We briefly conclude in section □.

II. Related Work

More and more researchers have studied the controller placement problem. The controller placement (CP) optimization problem was first defined in [4]. The CP optimization problem mainly concerns on two questions: how many and where controllers should be deployed. Heller et al. introduced the controller placement problem and explored the tradeoffs when optimizing for minimum latency between nodes and controllers [4]. They found that, one controller is enough to meet common network expectations with respect to communications delay (but obviously not fault tolerance) in some cases. Multiple controllers have declared best performance e.g. [1], [3], [10], network delay is not the only factor that should be focused since the capacity of a controller is limited. Yao et al. considered that the load of a controller should not exceed its capacity, and defined a capacitated controller placement problem (CCPP) [9].

Some other works of CP optimization problem considered the reliability of networks. In [11], J.Ros et al. defined a fault tolerant controller placement problem (FTCP) and exploit a heuristic algorithm to meet the reliability and give an upper bound on the number of controllers. They found that each node is required to connect to 2 or 3 controllers, which typically provide ample reliability [12]. In [3], Guo et al. consider two problems, the controller placement under comprehensive network state problem (CPCNS) and the controller placement under single link failure problem (CPSLF) are discussed, to evaluate the reliability of networks, where a traversal algorithm and a greedy-based algorithm are used to find the best solution to meet the requirement of these two problems. Another work that deals with controller placements from a reliability view point is due to Hu et al. [13]. In this case, they compared different placement algorithms, such as l-w-greedy, simulated annealing, and brute force. The brute force is proved the best solution without considering the time.

Many of the works solved the CP optimization problem with more than one metric, e.g. latency and reliability. In [14-15], the Pareto-optimal controller placement (POCO) framework is proposed to search all the candidate solutions with respect to different performance metric. Lange et al. extend the POCO framework with heuristic to make it work in large or dynamic topologies [6]. However, the number of controllers to be deployed in the network is an input parameter in the POCO framework. In addition, Hu et al. extended controller placements work in [16]. A metric called expected percentage of control path loss to measure reliability is proposed. However, similar to [14-15], the number of controllers is given as input, each switch only connects to its nearest controller, and only one-failure scenarios are considered. In advance, it is impossible to know how many controllers are required in a large scale network.

Facility location problem is a fundamental problem in operations research and theoretical computer science. The Uncapacitated Facility Location Problem (UFLP) is one of the most basic facility location problems. *k*-RCP algorithm is inspired by the primal-dual method for UFLP [17]. Jian and Varizani present approximation algorithms for UFLP achieving approximation guarantees of 3 [17]. The currently best approximation ratio is 1.488 by Li [18] based on non-uniformly randomized rounding and dual fitting technique. Many variants of the UFLP have been proposed, such as the hard(soft) capacitated facility location problems [19], *k*-level facility location problem [20] and facility location problem with outliers [21-22], among others.

In summary, existing optimization methods have limited abilities to strike a balance between network delay and controller load against links failure. In this paper, we focus on the *k*-RCP optimization problem, and to consider that how many controllers are needed and how to deploy the least number of controllers such that the resulting SDN is the most robust and efficient.

III. Problem Statement

A. Evaluation metrics

For a network denoted by $G = (V, L)$, where $V = \{1,2, \dots i, \dots, j, \dots, n\}$ is the set of nodes (switches), C is a set of controller, and V/C represents the set of ordinary switch. We assume every node is a candidate of the controller deployed location, so we would not make the distinction between nodes and switches, and $V = C \cup V/C$. L is set of physical links, and link $(i, j) \in L$. C^0 represents the set of controllers that have been deployed initially in the network graph G, $C^0 \subseteq C$.

In this paper, we introduce controlled proportion (*Cpr*) metric and transmission efficiency (*TE*) metric to measure the robust connectivity and the efficiency of an existing SDN under *k*-links failure, respectively.

In this paper, all nodes and links are of equal importance. The links are undirected. All the following definitions refer to a graph with n nodes unless explicitly specified. We first define controlled proportion (*Cpr*) metric to quantitatively characterize the robust connectivity against any k links failure, it represents the number of switch that can access any one controller in a SDN. The definition of *Cpr* metric is as follows:

$$Cpr = n_c/n, \tag{1}$$

where n_c represents the total number of served switches that can receive the control signaling. Note, the required controlled proportion is to meet the request of a specific application.

The *k*-RCP optimization problem focuses on not only the robust connectivity, but also the efficiency of the SDN against k links failure. We introduce transmission efficiency (*TE*) metric to characterize the delay between the switch and the associated controller, and the delay between the controllers. Before defining the *TE* metric, we need to give the following explanations.

In order to simplify the problem, we use the distance between nodes to describe the propagation delay. Let d_{ij} denotes the distance between node (switch) i and node (switch) j. If there is no path between switches i and j, $d_{ij} =$

∞. We follow the definition of efficiency in [23] in this paper. Let $e_{ij} = 1/d_{ij}$ represents the efficiency between two switches i and j. The aim of using e_{ij} is to avoid the case where the distance between the switches is ∞ in the disconnected SDN. Note that $e_{ii} = 1$ indicates that the efficiency is 1 where switch i can access the controller resources deployed on itself.

For a given SDN, the transmission efficiency (*TE*) is defined as:

$$TE = \sum_{i \in V/C} sw_i + \frac{1}{2} \sum_{p \in C} \sum_{q \in C} \frac{1}{d_{pq}}, \quad (2)$$

where $sw_i = max_{j \in C} 1/d_{ij}$ $(i \in V/C)$ indicates that switch i can access the nearest controller j. The previous item in (2) $\sum_{i \in V/C} sw_i$ describes the efficiency of an ordinary switch when it access the controller in SDN, and the latter represents how well difficulty on exchanging information between controllers. The larger *TE* is, the easier switches in the SDN to access controller resources.

In fact, a certain number of links failure may cause huge data loss and connectivity disruption in some networks. The cost of repairing or reconfiguring these networks is high. Therefore, this paper focuses on how to optimize the configuration of limited controller resources under the worst-case of k-links failure, so that the SDN can not only meet the required controlled proportion, but also take into account the SDN transmission efficiency and load balancing.

For a network $G(V, L)$, $n = |V|$ and $m = |L|$. Let S be the set of scenes with arbitrary links failure, and the scene space is 2^m. The probability of occurrence of each scene $s (s \in S)$ is denoted as q^s. Among them, the set of scenes where k-links failure is denoted as $S_k (S_k \subset S)$, and the worst case is denoted as $s_b (s_b \in S_k)$. Next we will discuss the k-links removed in the worst-case.

In graph theory, betweenness is a measure of centrality in a graph based on shortest paths. Link betweeness is a measure of the number of shortest paths between pairs of nodes that run along the link [24]. We use L^{s_b} represents the set of k-links removed in the worst-case links failure ($k = |L^{s_b}|$). Each link in L^{s_b} is removed based on link betweenness descending order in a SDN.

B. Problem description

Consider a SDN network with n nodes and m links, represents as an undirected graph $G(V, L)$. The k-RCP optimization problem can be formulated as: given a graph $G(V,L,C^0)$ and a required controlled proportion *Cpr*, how to deploy the set of controllers C ($C^0 \subseteq C$) under any k-links removed, so that the $G(V,L,C)$ can not only meet the required controlled proportion, but also take into account the transmission efficiency and load balancing in the SDN.

For example in Fig.1, it illustrates how to deploy two more proper controllers to improving the load balancing and reducing the delay in the resulting graph. Given the required controlled proportion *Cpr*=6/8 in Fig.1. There is a graph with 8 switches and 9 links, where $V = \{1,2,3,4,5,6,7,8\}$, L={(1,2),(2,3),(3,4),(4,5),(5,6), (6,7), (7,8), (1,8), (4,8)}, and $C^0 = \{4\}$ in Fig.1(a). The removal of links (1,8) and (3,4) will disconnect the initial graph (Fig.1(a)). In the case, the *Cpr* is 5/8, switches 4, 5, 6, 7 and 8 can access the controller on switch 4, however, switches 1, 2 and 3 on the other subgraph cannot access the controller on switch 4. Whereas,

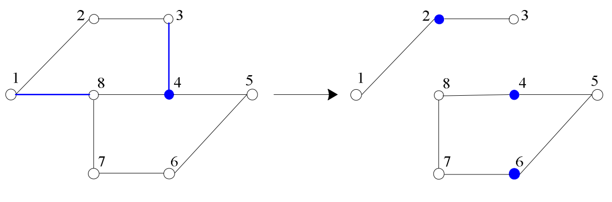


Fig.1. The example of robust controller placement

TABLE I CONTROLLERS AND ITS CONTROLLED SWITCHES

Controller	*Controlled Switches*
2	1, 2, 3
4	4, 5, 8
6	6, 7

when deploying two new controllers on switches $\{2, 6\}$, $C = \{2,4,6\}$, the removal of arbitrarily two links in Fig.1(b) all can meet the required controlled proportion .

At the same time, the load balancing of controllers is achieved (see TABLE Ⅰ) and the delay between controllers and switches is reduced.

IV. Robust Controller Placement Method

This section is devoted on presenting the k-RCP optimization problem models and devising the efficiency k-RCP algorithm for robust controller placement.

A. k-RCP model

The k-RCP optimization problem is first modeled as an Integer Linear Programming (ILP) model, termed k-RCP_ILP, which optimally deploy the limited controllers to meet the required controlled proportion against any k-links failure in SDN. The k-RCP_ILP (L1 for short) model includes:

Data:

- S_k: a set of all scenes of k-links removed from the SDN;
- s_k: any a scene of k-links removed from the SDN, $s_k \in S_k$;
- s_b : corresponding worst-case of k-links removed, $s_b \in S_k$;
- q^{s_k} : the probability of occurrence of scene s_k , $\sum_{s_k \in S_k} q^{s_k} = 1$;
- q^{s_b}: the probability of occurrence of the worst case of k-links removed.
- f_j: the cost of placing a controller on switch j;
- $c_{ij}^{s_k}$: the delay between switch i and the controller j under the scene s_k;
- Cpr: the required controlled proportion.

Variables:

- x_j : 0-1 variable, $x_j = 1$ means that switch j is selected to place a controller resource, otherwise it is 0;

- ${y_{ij}}^{s_k}$: 0-1 variable, ${y_{ij}}^{s_k}=1$ means that switch i is served by controller j under the scene s^k, otherwise ${y_{ij}}^{s_k}=0$;
- $r_i^{s_k}$: 0-1 variable, indicates whether or not the switch i is served by a controller under the scene s_k, and if the switch i is served by a controller, $r_i^{s_k}=1$, otherwise $r_i^{s_k}=0$.

Objective:

$$\text{Min} \sum_{j\in V} f_j\cdot x_j+\sum_{s_k\in S_k} q^{s_k}\cdot\sum_{i\in V}\sum_{j\in V} c_{ij}^{s_k}\cdot y_{ij}^{s_k} \quad \text{(L1-1)}$$

Constraints:

$$y_{ij}^{s_k}\le x_j \quad \forall i,j\in V,\forall s_k\in S_k \quad \text{(L1-2)}$$

$$\sum_{j\in V} y_{ij}^{s_k}+r_i^{s_k}=1 \quad \forall i\in V,\forall s_k\in S_k \quad \text{(L1-3)}$$

$$\sum_{i\in V} r_i^{s_k}\le n(1-Cpr) \quad \forall s_k\in S_k \quad \text{(L1-4)}$$

$$y_{ij}^{s_k},x_j,r_i^{s_k}\in\{0,1\} \quad \forall i,j\in V,\forall s_k\in S_k \quad \text{(L1-5)}$$

Remarks:

- The objective function of *k*-RCP_ILP (L1) is to minimize the number of controllers deployed in a SDN considering the transmission efficiency between switches and controllers, and controllers themselves under any *k*-links removed. In the objective function (L1-1), the delay $c_{ij}^{s_k}$ is used to describe the transmission efficiency between switches *i* and *j*; the cost f_j is used to control the number of controllers that are deployed in the SDN;
- Constraint (L1-2) indicates that if switch i can be served by controller *j*, then the controller *j* must be able to provide control services. If $x_j=0$, $y_{ij}^{s_k}=0, i\in V$; if $x_j=1$, then $y_{ij}^{s_k}=1\ or\ 0, i\in V$;
- Constraint (L1-3) ensures each switch is either a served switch or unserved switch. In particular, $y_{jj}^{s_k}=1$ means switch j is a controlled switch under the scene s_k;
- Constraint (L1-4) is very important, which ensures the controlled proportion of the resulting network is larger than the given required controlled proportion following any removal of *k*-links;
- Finally, constraint (L1-5) denotes the binary domain constraints on the variables x_j, ${y_{ij}}^{s_k}$ and $r_i^{s_k}$.

It is hard to find the optimal controller set *C* under any *k* links removed so that the controlled proportion is larger than the given requirement. In order to simplify the problem, this paper focus on how to optimize the placements of limited controllers resources under the worst-case *k*-links removed. So we assume the probability of the worst-case *k*-links removed is 1, $q^{s_b}=1$.

To solve the *k*-RCP_ILP (L1), we relax *k*-RCP_ILP (L1) with $q^{s_b}=1$ and establish its linear programming model *k*-RCP_LP (L2) as follows:

Objective:

$$Min \quad \sum_{j\in V} f_j\cdot x_j+\sum_{i\in V}\sum_{j\in V} c_{ij}^{s_b}\cdot y_{ij}^{s_b} \quad \text{(L2-1)}$$

Constraints:

$$y_{ij}^{s_b}\le x_j \quad \forall i,j\in V \quad \text{(L2-2)}$$

$$\sum_{j\in V} y_{ij}^{s_b}+r_i^{s_b}=1 \quad \forall i\in V \quad \text{(L2-3)}$$

$$\sum_{i\in V} r_i^{s_b}\le n(1-Cpr) \quad \text{(L2-4)}$$

$$y_{ij}^{s_b},x_j,r_i^{s_b}\ge 0 \quad \forall i,j\in V \quad \text{(L2-5)}$$

Since this model is a minimum optimization problem, constraint (L2-3) is equivalent to:

$$\sum_{j\in V} y_{ij}^{s_b}+r_i^{s_b}\ge 1 \quad \forall i\in V \quad \text{(L2-6)}$$

To further solve the linear programming, we utilize its dual model of L2. The dual model *k*-RCP_DOLP (L3) is established:

Objective:

$$\max \quad \sum_{i\in V}\alpha_i+n(\alpha-1)q \quad \text{(L3-1)}$$

Constraints:

$$\sum_{i\in V}\beta_{ij}\le f_j \quad j\in V \quad \text{(L3-2)}$$

$$\alpha_i-\beta_{ij}\le c_{ij} \quad \forall i,j\in V \quad \text{(L3-3)}$$

$$\alpha_i-q\le 0 \quad \forall i\in V \quad \text{(L3-4)}$$

$$\alpha_i,\beta_{ij},q\ge 0 \quad \forall i,j\in V \quad \text{(L3-5)}$$

The dual variable β_{ij} can be viewed as the contribution of switch i to the placing cost f_j of controller j that can be connected with switch i. The dual variable α_i can be viewed as the total cost of switch *i*. The total cost includes the delay c_{ij} and the dual variable β_{ij} caused by switch *i*. The dual variable q is defined as the maximum value in all of the values α_i, $(\forall i\in V)$.

B. k-RCP algorithm

The *k*-RCP algorithm includes two sub-algorithms: the *k*-RCP-GN algorithm and the *k*-RCP-DOLP algorithm. The *k*-RCP-GN algorithm characterizes *k*-links removed under the worst-case. The *k*-RCP-GN algorithm is based on the *k*-RCP-DOLP dual model to find the minimum and optimal controller set.

1) k-RCP-GN algorithm

The classic community discovery algorithm, GN algorithm, can not only extract the community structure but also divide a network into relatively uniform subnets at fast speed. The relatively uniform size block will be better for the load balance for multiple controller placement. Therefore, we use the *k*-RCP-GN algorithm to describe the worst-case *k*-links removed scene in the SDN. The main steps of the *k*-RCP-GN algorithm are described below:

- Initial value: *num*=0, $L^{s_k}=\emptyset$;
- Sort links according to the link betweenness, and remove link *l* with the largest link betweenness in the network;
- *num*=*num*+1, $L^{s_k}=L^{s_k}\cup\{l\}$;
- Check the value of num. If *num*<*k*, perform step 2; If *num*= *k*, perform step 5;
- The set of *k*-links removed is L^{s_k} under the worst-case links failure.

2) *k*-RCP-DOLP algorithm

This paper proposes a *k*-RCP-DOLP approximation algorithm to deploy the minimum number of controller considering the required *Cpr* and *TE* metric against links failure. The *k*-RCP-DOLP algorithm draws on the dual approximation algorithm in Uncapacitated Facility Location Problem (UFLP) [17], and is redesigned and optimized based on the *k*-RCP problem proposed in this paper. The main steps of *k*-RCP-DOLP algorithm are as follows.

According to L2 model and L3 model, the following equation is established in a perfect situation:

$$\sum_{i\in V}\beta_{ij}=f_j \qquad j\in V \qquad \text{(L3-6)}$$

$$\alpha_i-\beta_{ij}=c_{ij} \qquad \forall i,j\in V \qquad \text{(L3-7)}$$

- Initial value: $\alpha_i=0$, $\beta_{ij}=0$, $\forall i,j\in V$ and $C=\emptyset$. For any *i* and *j*, α_i can increase, and β_{ij} can not increase;
- $\alpha_i=\alpha_i+1$ for α_i which is marked that can be increased;
- If there is a switch pair (*i*,*j*) that satisfies the equation $\alpha_i=c_{ij}$ $\forall i,j\in V$, execute step 4, if it does not exist, execute step 5;
- The path (*i*,*j*) which satisfies the constraint $\alpha_i=c_{ij}$ is marked tight, simultaneously the β_{ij} is marked that can be increased in the next cycle. At this time, α_i is increased synchronously with β_{ij} to ensure that the constraint (L3-7) is satisfied. If the switch *j* has been selected as a controller, then marking that switch *i* has been controlled and its controller is *j*;
- $\beta_{ij}=\beta_{ij}+1$ for β_{ij} which is marked that can be increased;
- If there is a switch *j* that satisfies the constraint (L3-6), $\sum_{i\in V}\beta_{ij}=f_j$, execute Step 7, and if it does not exist, execute Step 9;
- Finding the switch set V^t satisfying the constraint (L3-6), $V^t=\{j\mid \sum_{i\in V}\beta_{ij}=f_j\}$, and counting the number of switches that can be connected by *j* for all $j\in V^t$. Selecting the switch $j^{'}$ from V^t which has the largest number of connectable switches. How do you know which switch can be connected by the switch $j\in V^t$? All switches that are tight with the switch $j\in V^t$ and not yet connected by the other already assured controller are the switch that can be connected by the switch $j\in V^t$;
- Marking $j^{'}$ is a controller, $C=C\cup\{j^{'}\}$, and $j^{'}$ is no longer an ordinary switch. Marking the switch *i*, $i\in$ {switches can be connected by the switch $j^{'}$}, can be controlled by $j^{'}$, and marking α_i, $\beta_{ij^{'}}$, $i\in$ {switches can be connected by the switch $j^{'}$ } no longer increase;
- If the number of controlled switch is larger than *Cpr***n*, terminating the program. Otherwise, starting the next cycle and continuing from Step 2.

C. Performance analysis

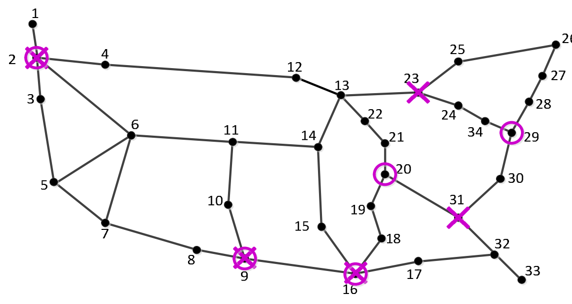
In this section, we analyze the time complexity of *k*-RCP algorithm. The *k*-RCP algorithm consists of *k*-RCP-GN, *k*-RCP-DOLP. The *k*-RCP-GN algorithm is based on the classical community extraction algorithm which is called GN algorithm [25]. $O(mn)$ is the time complexity of Floyd algorithm which is used to calculate the shortest path in *k*-RCP-GN. Floyd algorithm needs to be called *k* times in *k*-RCP-GN because of the *k* removing links. Therefore, the time complexity of the *k*-RCP-GN algorithm is $O(kmn)$. In the *k*-RCP-DOLP algorithm, there are at most n iterations. In order to find the pairs of switches satisfying the constraint $\alpha_i=c_{ij}$ in each iteration, *k*-RCP-DOLP algorithm requires traversing the $n\times n$ matrix at the worst case. Therefore, the time complexity of *k*-RCP-DOLP algorithm is $O(n^3)$. In summary, the time complexity of the *k*-RCP algorithm is $O(kmn+n^3)$. The time complexity of Branch and Bound method is $O(2^n)$ by traversing all the feasible solutions.

V. Simulation Results and Relevant Discussion

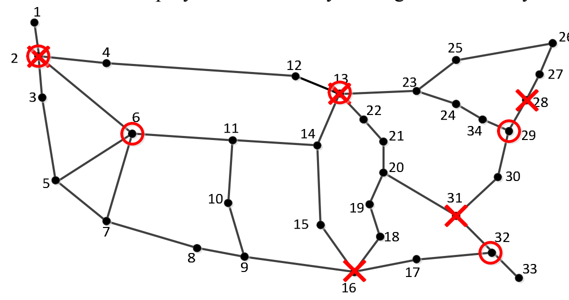
In this section, we evaluate the robustness and efficiency of SDN against links failure by comparing our *k*-RCP method with typical controller placement methods: Average-case Latency method [4], Worst-case Latency method [4] and Branch and Bound method. In this paper, we use two actual networks: the OS3E network (see Fig. 2), the network topology with 34 nodes and 42 links, and the US Carrier network from the Internet Topology Zoo with 139 nodes and 166 links.

A. OS3E network

We first study the effectiveness of our *k*-RCP method and three typical controller placement methods against the worst-case links removed in OS3E network (Fig. 2).



a. × deployed controllers by Worst-case Latency
○ deployed controllers by Average-case Latency



b. × deployed controllers by Branch and Bound
○ deployed controllers by k-RCP

Fig.2. Comparison results of optimized controller placement in OS3E network

TABLE II LOAD OF CONTROLLER OF BRANCH AND BOUND

Controllers	*Controlled switches*	*Load*
2	1,2,3,4,5,6,7,11	8
13	12,13,14,21,22,23,24,25	8
16	8,9,10,15,16,17,18,19	8
28	26,27,28,29,34	5
31	20,30,31,32,33	5

TABLE III LOAD OF CONTROLLER OF K-RCP

controllers	*controlled switches*	*load*
2	1,2,3,4,12	5
6	5,6,7,8,9,10,11	7
13	13,14,15,21,22	5
29	23,24,25,26,27,28,29,30,34	9
32	16,17,18,19,20,31,32,33	8

Suppose the required *Cpr* is 85%, that is *Cpr*=85%. When 10 links were removed under the worst-case, Fig.2 shows the results of optimized controller placement by the Average-case Latency method, the Worst-case Latency method, the Branch and Bound method and our *k*-RCP method. For the Average-case Latency method and the Worst-case Latency method, the optimized controller set is respectively *C*={2,9,16,20,29} and *C*={2,9,16,23,31} in Fig.2a. For the Branch and Bound method and the *k*-RCP method, the optimized controller set is respectively *C*={2,13,16,28,31} and *C*={2,6,13,29,32} in Fig.2b. Moreover, as can be seen from TABLE Ⅱ and TABLE III, the *k*-RCP method has a certain load balancing capability.

In this paper, we use the *Cpr* metric to characterize the robust connectivity of a SDN under links failure. Fig.3 shows the variation of controlled proportion with the increase of the number of the worst-case link removed by Average-case Latency method, Worst-case Latency method, Branch and Bound method and *k*-RCP method. We see that the *Cpr* metric is respectively 29/34, 29/34, 27/34 and 1 based on the Average-case Latency method, Worst-case Latency method, Branch and Bound method and *k*-RCP method when 10 links are removed under the worst-case. We observe that the *k*-RCP method is better than other three methods with more than 6 links removed. We have the remark that the *k*-RCP method has clear advantage in terms of improving the robustness against links failure.

We use the *TE* metric to characterize the efficiency of a

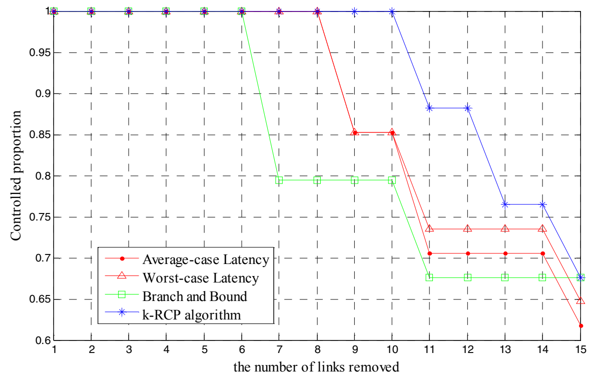


Fig.3. The controlled proportion versus the number of worst-case link removed in OS3E network

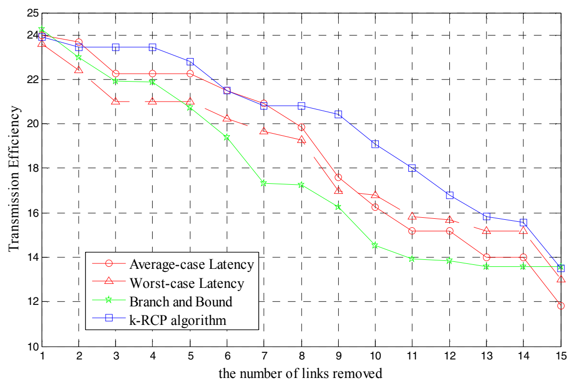
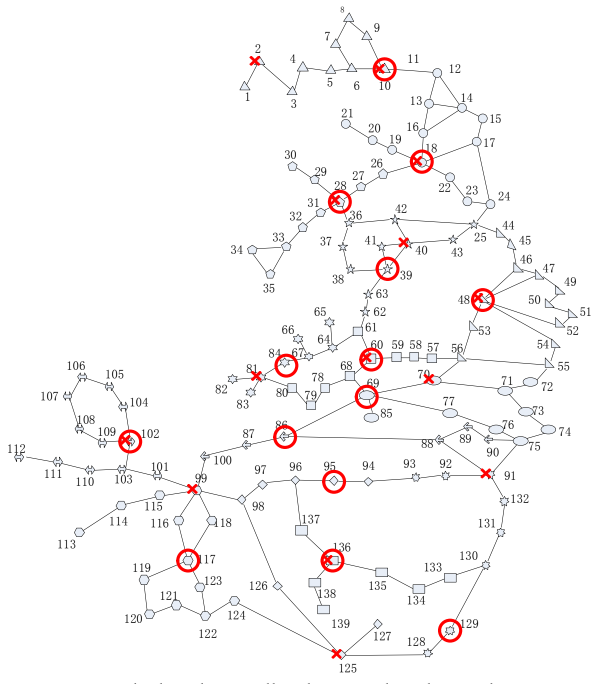


Fig.4. The transmission efficiency versus the number of worst-case link removed in OS3E network

SDN under links failure. Fig.4 shows the variation of *TE* metric with the increase of the number of the worst-case link removed by four controller placement methods. We see that the value of *TE* by the *k*-RCP method decline slowly compared the other three methods. We have a mark that the *k*-RCP method has a better efficiency to ensure transmission efficiency against links failure.

B. US Carrier network

For a further validation, we use the US Carrier network (Fig.5) from Internet Topology Zoo topology. Suppose the required *Cpr* is 85%. When 25 links were removed under the worst-case, Fig.5 shows the results of optimized controller placement by the Branch and Bound method and *k*-RCP method. For the Branch and Bound method, the optimized controller set is *C*={2,10,18,28,40,48,60,70,81,91,99,102, 125,136} marked with red ×, and the optimized controller set for the *k*-RCP algorithm is *C*={10,18,28,39,48,60,69, 84,86,95,102,117,129,136} marked with red ○ in Fig.5. Moreover, the load of each deployed controller based on *k*-



× deployed controllers by Branch and Bound
○ deployed controllers by *k*-RCP

Fig.5. Results of optimized controller placement in US Carrier network

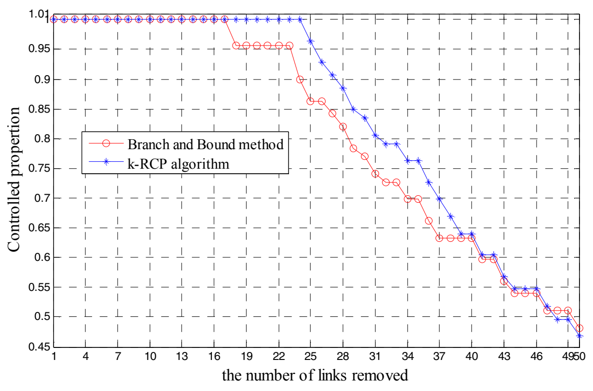


Fig.6. The controlled proportion versus the number of worst-case link removed in US Carrier network

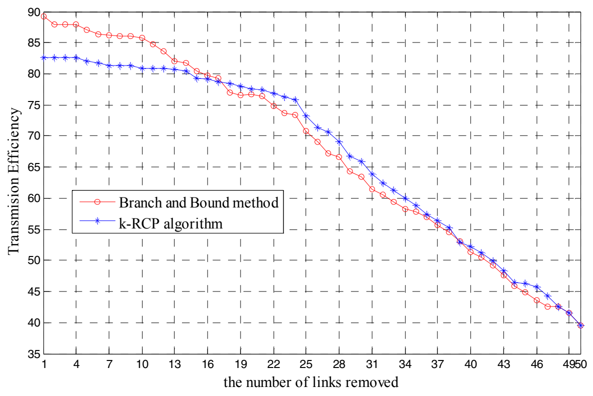


Fig.7. The transmission efficiency versus the number of worst-case link removed in US Carrier network

RCP method is shown in Fig.5 with different shape.

Fig.6 shows the values of *Cpr*, and Fig.7 shows the *TE* value, with respect to the number of the worst-case links removed based on the Branch and Bound method and the *k*-RCP method. We can conclude that the *k*-RCP method has a certain advantage compared with the Branch and Bound method in improving robust connectivity against the worst-case links removed in Fig.6. In Fig.7, the value of *TE* metric by the Branch and Bound method is larger than the *k*-RCP method at the beginning. The reason is that the Branch and Bound method is an optimal algorithm traversing all the feasible solutions without considering the robustness. But as the number of links removed increase, the superiority of the *k*-RCP method with low time complexity is obvious.

Under considering both Fig.6 and Fig.7, we can find that the *k*-RCP algorithm can both effectively improve *Cpr* metric and enhance *TE* metric when *k* links fail. At the same time *k*-RCP algorithm has a lower time complexity.

In summary, from the analysis of Fig.2-7 and TABLE II-III, it can be concluded that *k*-RCP method can effectively improve the performance of SDN as removing link increases. Controlled proportion and transmission efficiency have been significantly improved by *k*-RCP algorithm, and controller load balance is also solved properly. In addition, the *k*-RCP algorithm has a lower time complexity compared with Branch and Bound algorithm.

VI. CONCLUSION

This paper focuses on improving the robustness and efficiency of the software-defined network (SDN) against any *k*-links failure by optimized limited controllers placement. We propose the problem of robust controller placement against any *k*-links failure (*k*-RCP) optimization considering both network delay and load balancing. The *k*-RCP problem was first established an integer linear programming which optimally deployed the least controllers to meet the required controlled proportion against *k*-links failure. Then the integer linear programming model was relaxed, and its linear programming and dual programming models were further established. To solve the *k*-RCP problem, we develop an efficient *k*-RCP algorithm based on *k*-RCP dual programming model. Simulation results show that the *k*-RCP algorithm effectively improves the SDN robustness when considering network delay and load balancing. Furthermore it can significantly reduce time complexity of resolving *k*-RCP integer linear programming model directly.

References

[1] Z. Guo, M. Su, Y. Xu, Z. Duan, L. Wang, S. Hui, and H. J. Chao, “Improving the performance of load balancing in software-defined networks through load variance-based synchronization,” Computer Networks, vol. 68, no. 11, pp. 95–109, 2014.

[2] A. Krishnamurthy, S. P. Chandrabose, and A. Gember-Jacobson, “Pratyaastha: an efficient elastic distributed SDN control plane,” in The Workshop on Hot Topics in Software Defined Networking, 2014, pp. 133–138.

[3] S. Guo, S. Yang, Q. Li, and Y. Jiang, “Towards controller placement for robust software-defined networks,” in IEEE International Performance Computing and Communications Conference, 2015, pp. 1–8.

[4] B. Heller, R. Sherwood, and N. McKeown, “The controller placement problem,” in Proceedings of the first workshop on Hot topics in software defined networks, 2012, pp. 7–12.

[5] J. Liao, H. Sun, J. Wang, Q. Qi, K. Li, and T. Li, “Density cluster based approach for controller placement problem in large-scale software defined networkings,” Computer Networks, vol. 112, pp. 24–35, 2017.

[6] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, “Heuristic approaches to the controller placement problem in large scale SDN networks,” IEEE Transactions on Network & Service Management, vol. 12, no. 1, pp. 4–17, 2015.

[7] M. Karakus and A. Durresi, “Quality of service (qos) in software defined networking (sdn): A survey,” Journal of Network and Computer Applications, vol. 80, pp. 200–218, 2017.

[8] K. Y. Qin, C. H. Huang, C. H. Wang, X. Chen, “Balanced multiple controllers placement with latency and capacity bound in software-defined network,” Journal on Communications, no. 37(11), pp. 90–103, 2016.

[9] G. Yao, J. Bi, Y. Li, and L. Guo, “On the capacitated controller placement problem in software defined networks,” IEEE Communications Letters, vol. 18, no. 8, pp. 1339–1342, 2014.

[10] D. Erickson, “The beacon openflow controller,” in ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, 2013, pp. 13–18.

[11] F. J. Ros and P. M. Ruiz, “On reliable controller placements in software-defined networks,” Computer Communications, vol. 77, pp. 41–51, 2016.

[12] F. J. Ros and P. M. Ruiz, “Five nines of southbound reliability in software-defined networks,” in ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, 2014, pp. 31–36.

[13] Y. Hu, W. Wang, X. Gong, and X. Que, “Reliability-aware controller placement for software-defined networks,” in Ifip/ieee International Symposium on Integrated Network Management, 2013, pp. 672–675.

[14] D. Hock, S. Gebert, M. Hartmann, T. Zinner, and P. Tran-Gia, “POCO-framework for pareto-optimal resilient controller placement in SDN-based core networks,” 2014 IEEE Network Operations and Management Symposium (NOMS), pp. 1–2, 2014.

[15] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, “Pareto-optimal resilient controller placement in SDN-

based core networks," in Teletraffic Congress (ITC), 2013 25th International, 2013, pp. 1–9.

[16] H. Yannan, W. W. Dong, G. X. Yang, Q. X. Rong, and C. S. Duan, "On reliability-optimized controller placement for software-defined networks," China Communications, vol. 11, no. 2, pp. 38–54, 2014.

[17] K. Jain and V. V. Vazirani, "Primal-dual approximation algorithms for metric facility location and k-median problems," in Foundations of Computer Science, 1999. Symposium on, 1999, pp. 2–13.

[18] S. Li, "A 1.488 approximation algorithm for the uncapacitated facility location problem," in International Colloquium on Automata, Languages, and Programming, 2011, pp. 77–88.

[19] K. Aardal, P. L. V. D. Berg, D. Gijswijt, and S. Li, "Approximation algorithms for hard capacitated k-facility location problems," European Journal of Operational Research, vol. 242, no. 2, pp. 358–368, 2015.

[20] A. A. Ageev, Y. Ye, and J. Zhang, "Improved combinatorial approximation algorithms for the k-level facility location problem," Siam Journal on Discrete Mathematics, vol. 18, no. 18, pp. 207–217, 2004.

[21] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers (extended abstract)," in Twelfth Acm-Siam Symposium on Discrete Algorithms, 2001, pp. 642–651.

[22] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, "A survey on software defined networking with multiple controllers," Journal of Network and Computer Applications, 2017.

[23] M. F. Habib, M. Tornatore, and B. Mukherjee, "Fault-tolerant virtual network mapping to provide content connectivity in optical networks," in Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference, 2013, pp. 1–3.

[24] V. Latora and M. Marchiori, "A measure of centrality based on network efficiency," New Journal of Physics, vol. 9, no. 6, p. 188, 2007.

[25] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," Proc Natl Acad Sci U S A, vol. 99, no. 12, pp. 7821–7826, 2002.