On the Bilevel Optimization to Design Control Plane for SDONs in Consideration of Planned Physical-Layer Attacks

Qian Lv, Fen Zhou, and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—In the network planning of software-defined optical networks (SDONs), the control plane design is of great importance because it directly affects the performance and reliability of network control and management (NC&M). In this paper, we consider the planned physical-layer attacks from a rational attacker, which can analyze the control plane of an SDON and target its attacks to the most vulnerable part. To address such attacks, we model the control plane design as a bilevel optimization, where the upper-level optimization is for the network planner to design the control plane whose vulnerability to planned attacks is minimized, while the lower-level optimization is for the attacker to plan its attacks such that the control plane can be disturbed as severely as possible. We first develop two approaches to solve the bilevel model exactly. Specifically, we first leverage the cutting plane method to solve it directly, and then transform it into a single-level mixed integer linear programming (MILP) model with the Bellman method for problem solving. To improve the time efficiency for large-scale problems, we also propose a polynomial-time approximation algorithm based on linear programming (LP) relaxation and randomized rounding. Extensive simulations with various physical topologies verify the effectiveness of our proposals.**

***Index Terms*—Software-defined optical networks (SDONs), Control plane resilience, Physical-layer attacks, Planned attacks, Bilevel optimization, Approximation algorithm.**

I. Introduction

OVER the past decades, the raising of cloud computing and Big Data applications has stimulated the surge in demand for backbone capacity [1, 2]. Hence, optical networks have been deployed rapidly in globe to expand and strengthen the physical infrastructure of backbone networks. However, this multi-dimensional growth in capacity, scale, and geographical coverage brings new challenges, especially on the network control and management (NC&M) of backbone networks [3–5]. Meanwhile, software-defined networking (SDN) has been widely considered as a promising network paradigm to facilitate scalable, flexible and programmable NC&M by decoupling the control and data planes of a network [6, 7]. Therefore, software-defined optical networks (SDONs) have attracted intensive interests recently [8–10].

In an SDON, the control plane leverages controllers to handle NC&M tasks (*e.g.*, lightpath management and switch configuration), while its data plane is responsible for data transmission. Nowadays, the technical advances on flexible-grid optical networking [11–13] and resource virtualization [14–17] have made the data plane of SDONs more adaptive. Nevertheless, without a carefully-designed control plane, SDONs cannot fully enable backbone operators to customize their networks on demand and provision new services timely. Fig. 1 shows the architecture of an SDON, where the control plane utilizes multiple controllers to monitor and configure the network elements (NEs) in the data plane (*e.g.*, optical transponders and switches) [18, 19]. Note that, even though more than one controller is used, the control plane is still logically-centralized. The rationale for instantiating multiple controllers is multi-fold, *e.g.*, load balancing, improving availability, and reducing latency. Hence, each controller manages a subset of NEs in the data plane via signalling in control channels. Meanwhile, as the control plane is logically-centralized, there are also control channels among the controllers for synchronizing network status and distributing control messages.

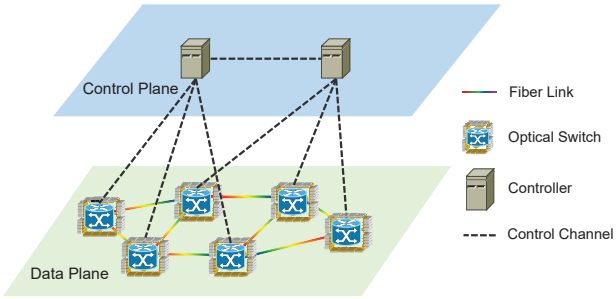

Fig. 1. Architecture of an SDON.

To realize a high-performance SDON, the network planner needs to design the control plane such that stringent quality-of-service (QoS) requirements (*e.g.*, short communication latency and high reliability) can be satisfied [20, 21]. More specifically, based on the configuration of the data plane, the control plane design needs to determine: 1) how many controllers to deploy, 2) where to deploy the controllers, 3) how to partition the controllers' NC&M territories in the data plane, and 4) how to route control channels in the SDON's physical topology. Here, since the SDON is a backbone network, we assume that its control and data channels use the same physical topology (*i.e.*, the same fiber links), for the sake of cost saving [21]. Specifically, the SDON consists of geographically distributed nodes, and to interconnect them, its operator needs

Q. Lv and Z. Zhu are with the School of Information Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, P. R. China (email: zqzhu@ieee.org).

F. Zhou is with the IMT Lille Douai, Institut Mines-Télécom, Univ. Lille, Center for Digital Systems, F-59000 Lille, France.

Manuscript received on April 19, 2020.

to deploy hundreds or even thousands of kilometers of fiber links. Hence, deploying independent fiber links just for the control channels will be very expensive and not cost-efficient, because the data-rates of control channels are in Kbps or Mbps at most. In other words, although the operator can provision the control channels as optical supervisory/service channels (OSCs) [22] and route them in a packet-switched network, which is logically decoupled from the data plane, the packet-switched network still needs to use the same fiber links in the SDON's physical topology for data transmission. Therefore, the control channels can be affected by physical-layer attacks.

Meanwhile, the operation of optical networks can be disturbed by various physical-layer issues, including both random failures/attacks [23–26] and planned attacks [27, 28]. Since control channels are routed in the same physical topology and thus can be impacted as well, such issues are more harmful to SDONs. After a physical-layer failure/attack, even though we can re-establish the control plane with either IP rerouting in the packet layer or the recovery scheme implemented in the optical layer, service interruptions cannot be avoided. Therefore, we should consider the physical-layer issues in network planning, *i.e.*, we should take them into account when solving the problem of control plane design and try to minimize their consequences. Previously, people have proposed a few control plane design algorithms to address random failures/attacks in [21, 29–32]. However, none of these studies have considered the planned physical-layer attacks. Compared with random failures/attacks, planned attacks are more devastating and more difficult to address. This is because they can be deliberately targeted to the most critical part of the control plane such that the resulting service interruptions would be aggravated.

When planned attacks need to be considered, the control plane design actually involves two rational entities, *i.e.*, the network planner and the attacker. Hence, the conventional single-level optimization can hardly be utilized to solve it. In other words, if the network planner's strategy is determined independent of the attacker's, the attacker can always leverage the strategy to make its attacks more harmful, and *vice versa*. For instance, if the network planner routes control channels over the shortest paths for reducing latency and path-failure probability, the control channels may be concentrated on the fiber links whose betweenness centrality is high [33]. Then, the attacker can easily target its attacks to these links to cause maximized service interruptions. To this end, we modeled the control plane design as a bilevel optimization in [34].

Specifically, the upper-level optimization is for the network planner to design the control plane whose vulnerability to planned physical-layer attacks is minimized, while the lower-level optimization is for the attacker to plan its attacks such that the control plane can be disturbed as severely as possible with the smallest efforts. The upper-level and lower-level optimizations are correlated, and thus cannot be solved independently. We transformed the bilevel optimization into a mixed integer linear programming (MILP) model, solved it for small-scale problems, and proposed a time-efficient heuristic.

In this work, we extend our preliminary study in [34] to make the problem-solving more comprehensive. The major improvements are as follows. Firstly, we modify the bilevel model in [34] to 1) allocate a pair of primary and backup controllers to each NE in the data plane, and 2) include the optimization for determining how many controllers to deploy. This improves the practicalness and completeness of our model, because the model in [34] did not consider backup controllers and could only accomplish the control plane design for a specific number of controllers. Secondly, we develop two approaches to solve the bilevel optimization exactly with improved time efficiency. More specifically, we not only leverage the cutting plane method [35] to solve it directly, but also transform it into a single-level MILP model with the Bellman method [36]. Thirdly, we propose a polynomial-time approximation algorithm based on linear programming (LP) relaxation and randomized rounding. Finally, we conduct extensive simulations to evaluate our proposals with various physical topologies and verify their effectiveness.

The rest of this paper is organized as follows. Section II surveys the related work. The description of the control plane design problem and the bilevel optimization to tackle it are presented in Section III. Our schemes to solve the bilevel optimization exactly are discussed in Section IV, while the polynomial-time approximation algorithm is proposed in Section V. We evaluate our proposals with numerical simulations in Section VI. Finally, Section VII summarizes the paper.

II. Related Work

People have invested lots of efforts on solving the problem of control plane design since the early days of SDN. The problem's initial version, *i.e.*, deciding how many controllers should be deployed and where to place them such that the control plane will be more reliable, was first mentioned in [29, 37]. Since then, how to improve the survivability of control plane has been tackled under various failure/attack scenarios.

Regarding the failures/attacks on controllers, researchers have come up with a few proposals in [20, 21, 31, 32, 38–42]. The studies in [20, 21, 38] used several primary-backup models to deal with single controller failures. Li *et al.* [31] utilized the Byzantine fault tolerant mechanism to address multiple controller failures. The work in [39] applied both the Byzantine fault tolerant and primary-backup models to tackle controller-switch mapping. Considering time-varying SDN environments, people leveraged dynamic controller provisioning to realize adaptive control planes for load balancing and controller failure handling in [40, 41]. Finally, logically-centralized but physically-distributed control planes have been proposed and demonstrated in [32, 42] for improved resiliency.

Regarding the failures/attacks in data plane, which could disturb the operation of control plane, people have also proposed several control plane design algorithms [30, 37, 43]. The authors of [43] designed a tree-like control plane to reduce the impact of switch failures in data plane. Assuming both switch and link failures, Zhang *et al.* [37] proposed a controller placement algorithm to minimize the connectivity loss between controllers and switches, while the study in [30] designed a Pareto-optimal approach to balance the tradeoff between the communication latency and resiliency of a control plane.

However, none of the studies mentioned above considered the failures caused by planned physical-layer attacks. As we

have already explained, the control plane design algorithms, which were designed to address random failures/attacks, cannot be utilized to resolve planned attacks because they did not consider the rationality of the attacker. As each fiber link in an SDON can carry more than Tbps data traffic, planned attacks can cause unimaginable losses to its operator [25]. Hence, the study in [28] tackled how to upgrade optical networks designed for content delivery, to address targeted fiber cuts.

Meanwhile, as physical-layer attacks impact both the data and control planes of an SDON, one should never ignore them when architecting the control plane [33]. To this end, we proposed a game theoretic approach in [44] to model the control plane design in consideration of planned attacks as a non-cooperative game between the network planner and the attacker. Nevertheless, since the analysis of Nash Equilibrium will become more complicated when the numbers of the two parties' strategies increase, the approach developed in [44] has scalability issues and thus only considers target fiber cuts as planned attacks. This, however, would limit its practicalness, because compared with fiber cuts, injecting jamming signals or introducing inter-channel crosstalk would be much easier and thus more common for physical-layer attacks [24, 25]. To this end, we turned to model the control plane design as a bilevel optimization in [34], but the study was just preliminary. This work extends the study in [34] to use more practical assumptions, design exact algorithms with improved time efficiency, and propose a polynomial-time approximation algorithm to solve large-scale problems.

III. Control Plane Design Considering Planned Attacks

In this section, we first define the problem of control plane design in consideration of planned physical-layer attacks, and then formulate it as a bilevel optimization.

A. Problem Description

The physical topology of an SDON can be modeled as a graph $G(V, E)$, where V and E represent the sets of optical nodes and bi-directional fiber links, respectively. Each optical node $v \in V$ consists of the NEs that are necessary for packet forwarding and data transmission, *e.g.*, an IP router, an optical switch, and a few optical transponders. The network planner needs to assign a pair of primary and backup controllers to the NEs on each optical node. Here, we leverage the mutual back model in [21] to improve the efficiency of controller assignment, *i.e.*, one controller can simultaneously be the primary controller of certain optical nodes and the backup one of some others. In a practical SDON, there might be optical nodes that do not satisfy the conditions to instantiate a controller, for reasons such as shortage of necessary equipment and security considerations. Therefore, we denote such optical nodes with a set $V' \subset V$. In addition to controller placement, the control plane design also needs to route control channels in $G(V, E)$ and minimize the impacts of potential physical-layer attacks. Here, the control channels refer to those that bridge the communications between either a primary controller and its optical nodes or a pair of primary-backup controller.

Definition 1. *For the **Control Plane Design**, the network planner needs to*

- *determine the number of controllers to be deployed,*
- *find the optical nodes to instantiate the controllers,*
- *assign primary&backup controllers to each optical node,*
- *select a subset of fiber links (*i.e., $E' \subset E$*) and allow the control channels to be routed on them.*

We assume that the attacker knows about the result of the control plane design (*i.e.*, the subgraph $G'(V, E')$) and how control channels are routed over $G'(V, E')$ (*i.e.*, the routing protocol of control channels). This assumption is reasonable, because the attacker can obtain such information by either leveraging the man-in-the-middle attack [45] to eavesdrop control channels or deploying probes [46] to monitor control traffic silently. Then, the attacker can derive the layout of the control plane, target its attacks to the most vulnerable part, and maximize the effectiveness of the attacks[1]. Note that, the mechanisms of common physical-layer attacks (*e.g.*, fiber cuts, intra-/inter-channel crosstalk, jamming insertion, and gain competition of erbium-doped fiber amplifiers (EDFA)) determine that the most vulnerable part of the control plane is the fiber link, which carries the most control channels [24].

Definition 2. *The **Vulnerability** of a control design can be quantified as the maximum number of control channels that are routed over a single fiber link (*i.e., $\mathcal{C}_{\max}$*).*

In this work, we assume that the routing protocol of control channels is to set up them over the physical paths whose total transmission distance is the minimum. This is because by doing so, we can reduce control latency, which is one of the most important QoS parameters of control plane [29]. Note that, as the SDON is a backbone network, the link lengths are in hundreds or even thousands of kilometers, which means that the propagation latencies are hundreds of microseconds or even longer. Therefore, the propagation latency of a fiber link will be much longer than the processing latency in each optical node (*i.e.*, normally in the scale of a few microseconds). To this end, when estimating the control latency, we ignore the processing latencies in optical nodes.

Fig. 2 gives an illustrative example on the control plane design considered in this work. We can see that in the designed control plane, *Links* 2-4 and 5-6 are not selected to route control channels, while the most vulnerable part is *Link* 2-3 since it carries two control channels (*i.e.*, $\mathcal{C}_{\max} = 2$). Note that, when there are anomalies or failures in the SDON due to physical-layer attacks, the operator can leverage IP rerouting in the packet layer [47, 48] to recover the affected control channels, but it takes time to do so and the communications between the control and data planes are interrupted during the restoration. Therefore, we should consider preventive measures to reduce potential hazards in the phase of network planning. Meanwhile, we hope to point out that the network planning considered in this work is purely about the control plane of the SDON, and will not affect the data plane operation at all.

[1]With the control plane design, the attacker can launch various attacks. We specifically consider planned physical-layer attacks [34], and the comparison of attacking scenarios is out of the scope of this work.

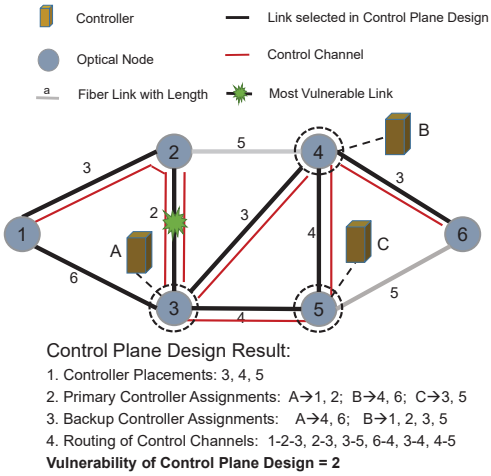

Fig. 2. Example on control plane design and its vulnerability.

B. Bilevel Optimization Model

As we take planned physical-layer attacks into consideration, the control plane design involves two rational entities, *i.e.*, the network planner and the attacker. Therefore, we model the problem as a bilevel optimization. The network planner's task is modeled as the upper-level optimization, which is to obtain a subgraph $G'(V, E')$, over which the control channels can be routed, such that the resulting control plane will have the minimum vulnerability to planned attacks. On the other hand, the attacker accomplishes the lower-level optimization, where it first derives the layout of the control plane based on $G'(V, E')$ and the routing protocol of control channels, and then finds the most vulnerable link(s) to launch attacks cost-efficiently. To this end, we can see that different control plane designs from the network planner motivate the attacker to target different links for maximizing the effectiveness of its attacks, while the most vulnerable link(s) chosen by the attacker will in turn change the network planner's control plane design. Hence, the upper-level and lower-level optimizations are correlated and cannot be solved independently. This actually explains why the control plane design cannot be formulated as a conventional single-level optimization.

Common Parameters:

- $G(V, E)$: the physical topology of the SDON.
- V': the set of optical nodes that do not satisfy the conditions to instantiate a controller ($V' \subset V$).
- $L_{(u,v)}$: the length of fiber link $(u, v) \in E$.
- $\mathcal{L}$: the maximum number of optical nodes that a controller can manage.

1) Upper-level Optimization: The network planner needs to obtain a subgraph $G'(V, E')$ for the control plane design such that the resulting vulnerability is minimized.

Variables:

- $\mathcal{N}_c$: the number of controllers to be deployed.
- $x_{(u,v)}$: the boolean variable that equals 1 if link (u, v) is selected in the control plane design, and 0 otherwise.
- $y^{z,w}_{(u,v)}$: the boolean variable that equals 1 if link $(u, v) \in E$ will be used to route the control channel between nodes z and w, and 0 otherwise.
- c_u: the boolean variable that equals 1 if a controller is placed on node $u \in V$, and 0 otherwise.
- $km_{u,v}$: the boolean variable that equals 1 if the controller on node v is the primary controller of node u.
- $ks_{u,v}$: the boolean variable that equals 1 if the controller on node v is the backup controller of node u.
- $b_{u,v}$: the boolean variable that equals 1 if there is a control channel between nodes u and v, and 0 otherwise.
- $o_{p,u,v}$: the boolean variable that equals 1 if the controllers on nodes u and v are the primary and backup controllers of node p, respectively, and 0 otherwise.
- $n_{(u,v)}$: the integer variable that indicates the number of control channels routed over link $(u, v) \in E$.
- $\mathcal{C}_{\max}$: the integer variable that indicates the vulnerability of the control plane design.
- $d_{u,v}$, $t_{u,v}$, $w_{u,v}$, $e_{u,v}$ and $r_{u,v}$: the auxiliary boolean variables that are introduced for linearizing constraints.

Objective:

The objective of the upper-level optimization is to minimize the vulnerability $\mathcal{C}_{\max}$ of the control plane design. This optimization correlates with the lower-level one through $\{y^{z,w}_{(u,v)}\}$.

$$\text{Minimize} \quad \mathcal{C}_{\max} \tag{1}$$

Constraints:

$$x_{(u,v)} = x_{(v,u)}, \quad \forall (u, v) \in E. \tag{2}$$

Eq. (2) ensures that each control channel is duplex and uses bi-directional links.

$$\sum_{u \in V'} c_u = 0. \tag{3}$$

Eq. (3) ensures that controllers will not be instantiated on the nodes in V'.

$$\sum_{u \in V} c_u = \mathcal{N}_c. \tag{4}$$

Eq. (4) calculates the total number of controllers that need to be deployed.

$$km_{u,v} + ks_{u,v} \leq c_v, \quad \forall u, v \in V, \tag{5}$$

$$\sum_{v \in V} km_{u,v} = 1, \quad \sum_{v \in V} ks_{u,v} = 1, \quad \forall u \in V. \tag{6}$$

Eqs. (5) and (6) determine the mapping between controllers and optical nodes.

$$\sum_{u \in V} km_{u,v} \leq \mathcal{L}, \quad \sum_{u \in V} ks_{u,v} \leq \mathcal{L}, \quad \forall v \in V. \tag{7}$$

Eq. (7) ensures that the primary and backup capacities of each controller will not be exceeded.

$$n_{(u,v)} = \sum_{\{z,w \in V : z \neq w\}} y^{z,w}_{(u,v)}, \quad \forall (u, v) \in E, \tag{8}$$

$$\mathcal{C}_{\max} \geq n_{(u,v)}, \quad \forall (u, v) \in E. \tag{9}$$

Eqs. (8) and (9) ensure that the control plane design's vulnerability is calculated correctly.

$$\begin{cases} o_{p,u,v} \leq km_{p,u}, \\ o_{p,u,v} \leq ks_{p,v}, \\ o_{p,u,v} \geq ks_{p,v} + km_{p,u} - 1, \end{cases} \quad \forall u, v, p \in V. \tag{10}$$

Eq. (10) determines the value of $o_{p,u,v}$.

$$\begin{cases} e_{u,v} - \sum_{p \in V} o_{p,u,v} \leq 0, \\ e_{u,v} - o_{p,u,v} \geq 0, \end{cases} \quad \forall u, v, p \in V. \tag{11}$$

Eq. (11) ensures that if the controllers on nodes u and v are the primary and backup controllers of node p, respectively, they communicate with each other through a control channel.

$$\begin{cases} b_{u,v} = km_{u,v} + km_{v,u} + e_{u,v} - d_{u,v} - t_{u,v} - w_{u,v} + r_{u,v}, \\ d_{u,v} \leq km_{u,v},\ d_{u,v} \leq km_{v,u},\ d_{u,v} \geq km_{u,v} + km_{v,u} - 1, \\ t_{u,v} \leq km_{u,v},\ t_{u,v} \leq e_{u,v},\ t_{u,v} \geq km_{u,v} + e_{u,v} - 1, \\ w_{u,v} \leq km_{v,u}\ d_{u,v} \leq e_{u,v},\ w_{u,v} \geq km_{v,u} + e_{u,v} - 1, \\ r_{u,v} \leq d_{u,v}\ r_{u,v} \leq e_{u,v},\ r_{u,v} \geq d_{u,v} + e_{u,v} - 1, \end{cases} \quad \forall u, v \in V. \tag{12}$$

Eq. (12) determines the value of $b_{u,v}$.

2) Lower-level Optimization: The attacker needs to derive the layout of control plane based on $G'(V, E')$ and the routing protocol of control channels. Therefore, the premise of solving the lower-level optimization is the determination of the upper-level one. In other words, when the lower-level optimization is considered alone, certain variables (*i.e.*, $\{x_{(u,v)}\}$ and $\{b_{u,v}\}$) of the upper-level optimization become parameters.

Variables:

- $y^{z,w}_{(u,v)}$: the boolean variable that equals 1 if link $(u, v) \in E$ will be used to route the control channel between nodes z and w, and 0 otherwise.

Objective:

The lower-level optimization's objective is to find the routing paths of control channels such that the total length of the paths is the minimum (*i.e.*, the routing protocol).

$$\text{Minimize} \quad \sum_{\{z,w \in V: z \neq w\}} \sum_{(u,v) \in E} L_{(u,v)} \cdot y^{z,w}_{(u,v)}. \tag{13}$$

Constraints:

$$\sum_{(u,v) \in E} y^{z,w}_{(u,v)} - \sum_{(v,u) \in E} y^{z,w}_{(v,u)} = \begin{cases} b_{z,w}, & u = z, \\ -b_{z,w}, & u = w, \\ 0, & \text{otherwise}, \end{cases} \quad \{u, z, w \in V : z \neq w\}. \tag{14}$$

Eq. (14) ensures the flow conservation conditions.

$$y^{z,w}_{(u,v)} \leq b_{z,w}, \quad \forall (u, v) \in E,\ \{z, w \in V : z \neq w\}. \tag{15}$$

Eq. (15) ensures that variables $\{y^{z,w}_{(u,v)}\}$ identify the links used by control channels correctly.

$$y^{z,w}_{(u,v)} = y^{w,z}_{(v,u)}, \quad \forall (u, v) \in E,\ \{z, w \in V : z \neq w\}. \tag{16}$$

Eq. (16) ensures that the bi-directional control channels between two nodes use the same routing path.

$$y^{z,w}_{(u,v)} \leq x_{(u,v)}, \quad \forall (u, v) \in E,\ \{z, w \in V : z \neq w\}. \tag{17}$$

Eq. (17) ensures that the routing paths of control channels can only use the links selected in the upper-level optimization.

IV. Exact Algorithms

We can easily verify that the bilevel optimization formulated above is $\mathcal{NP}$-hard [49]. In this section, we discuss two approaches that can solve it exactly.

A. Solving Bilevel Model with Cutting Plane Method

After analyzing the bilevel optimization in Section III-B, we find that it can solved directly by leveraging the cutting plane method [35], which constructs the exact solution based on suboptimal ones. We first solve the upper-level optimization without considering the objective of the lower-level one, and the optimization is transformed as follows.

$$\begin{aligned} &\text{Minimize} \quad \mathcal{C}_{\max}, \\ &\text{s.t.} \quad \text{Eqs. (2)-(12), (14)-(17).} \end{aligned} \tag{18}$$

The obtained solution can be suboptimal to the bilevel optimization, because we ignore the lower-level optimization objective in Eq. (13). In other words, the control plane design that minimizes the vulnerability $\mathcal{C}_{\max}$ might not automatically ensure the minimum total path length of control channels. Specifically, this situation happens if in the designed control plane $G'(V, E')$, there is at least one control channel whose routing paths will be different to satisfy the objectives in Eqs. (1) and (13). In this case, we call that there is a "conflict-cycle" in the deigned control plane.

Fig. 3 shows an example on the conflict-cycle. The control plane design places the primary controller of *Nodes* 1 and 2 on *Node* 4, and thus the optimization in Eq. (18) routes the control channels between optical nodes and their controller on *Paths* 1-3-4 and 2-1-4, respectively, to minimize $\mathcal{C}_{\max}$ as 1. However, the lower-level optimization will choose *Path* 1-4 to route the control channel between *Nodes* 1 and 4, because its length is shorter. Therefore, the solution provided by the optimization in Eq. (18) (*i.e.*, *Paths* 1-3-4 and 2-1-4) would be suboptimal for the bilevel optimization because of *Path* 1-4, which forms a conflict-cycle together with *Path* 1-3-4.

To avoid the conflict-cycles, we add a set of valid cuts to the optimization in Eq. (18), as follows.

- p: a routing path in $G(V, E)$, including one or more links.
- $|p|$: the hop-count of path p.
- ξ_p: the boolean indicator that equals 1 if path p is the routing path of a control channel, and 0 otherwise.

$$\sum_{(u,v) \in p} y^{z,w}_{(u,v)} = |p| - 1 + \xi_p, \quad \{z, w \in V : z \neq w\}, \tag{19}$$

$$y^{z,w}_{(u,v)} \geq \xi_p, \quad \forall (u, v) \in E,\ \{z, w \in V : z \neq w\}. \tag{20}$$

Eqs. (19) and (20) determine the value of ξ_p correctly.

$$\sum_{(u,v) \in p'} x_{(u,v)} \leq |p'| - \xi_p, \quad \forall (u, v) \in E. \tag{21}$$

Eq. (21) eliminates the path p', if it forms a conflict-cycle together with path p.

Algorithm 1 describes the procedure of our exact algorithm based on the cutting plane method. *Line* 1 is for the initialization. The while-loop that covers *Lines* 2-19 solves the bilevel optimization to obtain the optimal control plane design in iterations. In each iteration, *Lines* 4 and 5 solve the upper-level and lower-level optimization subsequently with topology $G'(V, E')$, and store their results on the control channels' routing paths in sets $\mathcal{P}_{\text{upper}}$ and $\mathcal{P}_{\text{lower}}$, respectively. Next, the for-loop covering *Lines* 6-15 checks the paths in $\mathcal{P}_{\text{upper}}$ and $\mathcal{P}_{\text{lower}}$, finds conflict-cycles if there are any, generates the set

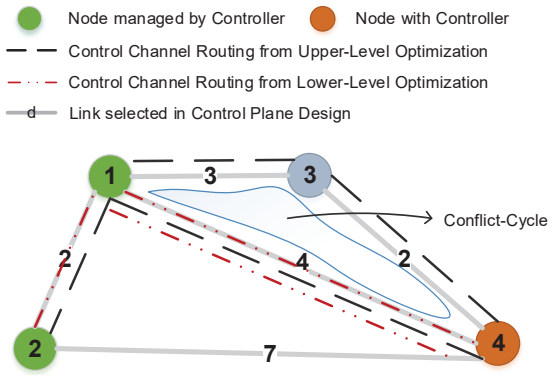

Fig. 3. Example on conflict-cycle in control plane design.

Algorithm 1: Exact Algorithm based on Cutting Plane

Input: $G(V, E)$, $\mathcal{L}$, V'.
Output: Control plane design $G'(V, E')$.

1 $flag = 0$, $E' = E$;
2 **while** $flag = 0$ **do**
3 $n = 0$;
4 solve the optimization in Eq. (18) with $G'(V, E')$ and store paths of all the control channels in set $\mathcal{P}_{\text{upper}}$;
5 solve the lower-level optimization in the designed control plane and recalculate paths of all the control channels to store in set $\mathcal{P}_{\text{lower}}$;
6 **for** *each control channel* **do**
7 find its paths in $\mathcal{P}_{\text{upper}}$ and $\mathcal{P}_{\text{lower}}$ as p_1 and p_2;
8 **if** $p_1 \neq p_2$ **then**
9 find conflict-cycle formed by p_1 and p_2 and get the set of valid cuts with Eqs. (19)-(21);
10 update $G'(V, E')$ with the set of cuts;
11 **break**;
12 **else**
13 $n = n + 1$;
14 **end**
15 **end**
16 **if** n *reaches the number of control channels* **then**
17 $flag = 1$;
18 **end**
19 **end**

of valid cuts with Eqs. (19)-(21), and updates $G'(V, E')$ to remove the conflict-cycles. Finally, when the paths in $\mathcal{P}_{\text{upper}}$ and $\mathcal{P}_{\text{lower}}$ are all identical, we set $flag = 1$ in *Line* 17, which will terminate the while-loop. Note that, as the problem-solving in *Lines* 4 and 5 cannot be accomplished in polynomial time, *Algorithm* 1 is not a polynomial-time algorithm.

B. Transforming into Single-level Model with Bellman Method

Instead of solving the bilevel optimization directly, we can also transform it into a single-level MILP model and then obtain the optimal solution with a conventional MILP solver. This can be done by replacing the objective and constraints of the lower-level optimization with its optimality conditions. We notice that the optimality conditions of the lower-level optimization can be expressed more compactly, if we leverage the Bellman method and develop a simple lifting process [36]. Specifically, the bilevel optimization in Section III-B can be transformed into the following single-level MILP model.

New Parameter:

- M: the big integer introduced for linearizing constraints.

New Variables:

- $\pi_u^{z,w}$: the length of the shortest path between nodes u and w, if u is on the routing path of the control channel between nodes z and w ($z, w \in V$).

Objective:

$$\text{Minimize} \quad \mathcal{C}_{\max}. \tag{22}$$

Constraints:

Eqs. (2)-(12), (14)-(17)

$$\pi_u^{z,w} - \pi_v^{z,w} \leq M - x_{(u,v)} \cdot \left(M - L_{(u,v)}\right) - 2L_{(u,v)} \cdot y_{(u,v)}^{z,w}, \quad \forall (u,v) \in E,\ \{z, w \in V : z \neq w\}, \tag{23}$$

$$\pi_w^{z,w} = 0, \quad \{z, w \in V : z \neq w\}, \tag{24}$$

$$\pi_u^{z,w} \geq 0, \quad \forall u \in V,\ \{z, w \in V : z \neq w\}. \tag{25}$$

Eqs. (23)-(25) are the Bellman's optimality conditions [36] for node pair z-w ($\{z, w \in V : z \neq w\}$), if the constraints in Eqs. (14)-(17) are satisfied.

V. Polynomial-time Approximation Algorithm

The exact algorithms in the previous section are not polynomial-time ones, and thus they will become intractable when dealing with large-scale problems. Therefore, in this section, we propose a polynomial-time approximation algorithm based on linear programming (LP) relaxation and randomized rounding, which can obtain near-optimal solutions whose performance gap to the optimal ones is bounded. We notice that after relaxing MILP, some variables do not satisfy the original constraints and produce infeasible solutions. Therefore, we add Eqs. (26)-(27) in MILP and then relax:

$$y_{(w,v)}^{z,w} = 0, \quad \forall (w,v) \in E,\ \{z, w \in V : z \neq w\}. \tag{26}$$

$$y_{(u,z)}^{z,w} = 0, \quad \forall (u,z) \in E,\ \{z, w \in V : z \neq w\}. \tag{27}$$

The procedure of the approximation algorithm is shown in *Algorithm* 2. In *Line* 1, we relax the MILP in Section IV-B to obtain an LP model. Note that, all the boolean variables are relaxed to real ones within $[0, 1]$. Then, we solve the LP in *Lines* 2-3, and based on the solutions of $\{x_{(u,v)}, c_u, km_{u,v}, ks_{u,v}, \mathcal{C}_{\max}\}$, the outer for-loop (*Lines* 4-33) builds a qualified approximation solution in iterations. Here, Q and γ determine the time complexity and approximation ratio of *Algorithm* 2, and their values are selected empirically. We will discuss their effects in Section VI.

The first inner for-loop (*Lines* 5-10) finds a set of feasible $\{x_{(u,v)}, \forall (u,v) \in E\}$ to the MILP with randomized rounding. Here, the set of $\{x_{(u,v)}\}$ is feasible as long as the resulting $G'(V, E')$ based on it is still a connected graph. Next, the second inner for-loop (*Lines* 11-28) finds sets of feasible $\{c_u, \forall u \in V\}$ and $\{km_{u,v}, ks_{u,v}, \forall u, v \in V\}$ subsequently, also with randomized rounding. The for-loop that covers *Lines*

Algorithm 2: Approximation Algorithm

Input: $G(V, E)$, $\mathcal{L}$, V', maximum number of rounding trails Q, and approximation ratio γ.
Output: Control plane design $G'(V, E')$.

1 relax the MILP in Section IV-B to get an LP;
2 solve the LP to get values of variables $\{x_{(u,v)}, c_u, km_{u,v}, ks_{u,v}, \mathcal{C}_{\max}\}$ in real numbers;
3 $\mathcal{C}_{\max} = \lceil \mathcal{C}_{\max} \rceil$;
4 **for** $q_3 = 1$ *to* Q **do**
5 **for** $q_1 = 1$ *to* Q **do**
6 round variables $\{x_{(u,v)}, \forall (u,v) \in E\}$ to 1 with the probabilities of $\{x_{(u,v)}\}$, and 0 otherwise;
7 **if** $\{x_{(u,v)}\}$ *is feasible to the MILP* **then**
8 **break**;
9 **end**
10 **end**
11 **for** $q_2 = 1$ *to* Q **do**
12 round variables $\{c_u, \forall u \in V\}$ to 1 with the probabilities of $\{c_u\}$, and 0 otherwise;
13 **for** *each node* $u \in V$ **do**
14 $\delta = 0$;
15 **for** *each node* $v \in V$ **do**
16 **if** $c_v = 0$ **then**
17 $\delta = \delta + km_{u,v} + ks_{u,v}$;
18 $km_{u,v} = ks_{u,v} = 0$;
19 mark $km_{u,v}$ and $ks_{u,v}$ as determined;
20 **end**
21 **end**
22 distribute the value of δ evenly to undetermined $\{km_{u,v}, ks_{u,v}, \forall v \in V\}$;
23 round undetermined $\{km_{u,v}, ks_{u,v}\}$ to 1 with probabilities of their values, and 0 otherwise;
24 **end**
25 **if** $\{km_{u,v}, ks_{u,v}\}$ *is feasible to the MILP* **then**
26 **break**;
27 **end**
28 **end**
29 calculate values of $\{y^{z,w}_{(u,v)}, \mathcal{N}_c, \hat{\mathcal{C}}_{\max}\}$ with the obtained $\{x_{(u,v)}, c_u, km_{u,v}, ks_{u,v}\}$;
30 **if** $\frac{\hat{\mathcal{C}}_{\max}}{\mathcal{C}_{\max}} \leq \gamma$ **then**
31 **break**;
32 **end**
33 **end**

15-21 makes sure that the settings of $\{km_{u,v}, ks_{u,v}\}$ do not conflict with that of $\{c_u\}$, *i.e.,* an optical node cannot have its primary or backup controller on a node where there is no controller. *Lines* 25-27 ensure that the primary and backup capacities of each controller are not exceeded, and the primary and backup controllers of a switch cannot be the same controller. After having obtained feasible $\{x_{(u,v)}, c_u, km_{u,v}, ks_{u,v}\}$, we calculate the values of $\{y^{z,w}_{(u,v)}, \mathcal{N}_c, \hat{\mathcal{C}}_{\max}\}$ based on them in *Line* 29. Here, $\hat{\mathcal{C}}_{\max}$ is the true optimization objective. Finally, *Lines* 30-32 compare $\hat{\mathcal{C}}_{\max}$ with the objective from the LP (*i.e.,* $\mathcal{C}_{\max}$), and terminate the iterations if the approximation ratio γ is satisfied.

We can easily verify that the approximation ratio of *Algorithm* 2 is upper-bounded by γ as follows. As the MILP in Section IV-B is for minimization, the solution of the LP (*i.e.,* $\mathcal{C}_{\max}$) actually provides a lower-bound on the optimal solution, while the feasible solution constructed by *Algorithm* 2 (*i.e.,* $\hat{\mathcal{C}}_{\max}$) is an upper-bound. Therefore, the approximation ratio of *Algorithm* 2 can be calculated as

$$\epsilon = \frac{\hat{\mathcal{C}}_{\max}}{\mathcal{C}^*_{\max}} \leq \frac{\hat{\mathcal{C}}_{\max}}{\mathcal{C}_{\max}} \leq \gamma, \tag{28}$$

where $\mathcal{C}^*_{\max}$ is the optimal solution. We also need to point out that according to the principle of LP relaxation with randomized rounding and the well-known Chernoff-Bound [50], the probability of *Algorithm* 2 finding a qualified feasible solution can approach to 1, as long as Q and γ are properly set. We will show the convergence performance of *Algorithm* 2 in Section VI. The time complexity of *Lines* 4-33 in *Algorithm* 2 is $O(Q^2 \cdot |V|^2)$, and the LP solving in *Lines* 1-2 can also be accomplished in polynomial-time. Hence, *Algorithm* 2 is a polynomial-time approximation algorithm.

VI. Performance Evaluation

In this section, we perform numerical simulations to evaluate the performance of the proposed algorithms.

A. Performance on Control Plane Design

To evaluate the algorithms in depth, we consider six physical topologies in different sizes for the control plane design in consideration of planned physical-layer attacks. The topologies are shown in Fig. 5, where the Netrail, GridNet, NSFNET and US-Backbone (USB) in Figs. 5(b) and 5(d)-5(f), respectively, are realistic ones used for backbone networks. The simulations are conducted on a computer with 2.1 GHz Intel Xeon Silver 4110 CPU and 32 GB memory, and the simulation environment is MATLAB 2019a with GLPK toolbox, which only uses one CPU core. Table I summarizes the results of the proposed algorithms. Here, we name the exact algorithms designed in Sections IV-A and IV-B as the Cutting-Plane-based and Bellman-based algorithms, respectively, while call *Algorithm* 2 as the Approximation algorithm. We set the longest running time as three hours, which means that we consider an algorithm as intractable for a scenario if it cannot obtain a solution with $10,800$ seconds. The intractable cases are marked with "$-$" in Table I. The optimization gap between the approximate solution and the exact one can be obtained by comparing the results on $\mathcal{C}_{\max}$ (*i.e.,* the vulnerability of control plane design) from the Approximation algorithm and the Bellman-based algorithm. The last column in Table I shows the results on the optimization gap explicitly.

1) Small-scale Topologies: We consider the 6-Node and Netrail topologies in Figs. 5(a) and 5(b) as small-scale topologies. In the simulations using them, we set $|V'| = 1$ and $|V'| = 2$ for 6-Node and Netrail respectively (*i.e.,* in each topology, there are one or two optical nodes that cannot instantiate a controller), select the optical nodes for V' randomly, assign the value of $\mathcal{L}$ (*i.e.,* the maximum number of optical

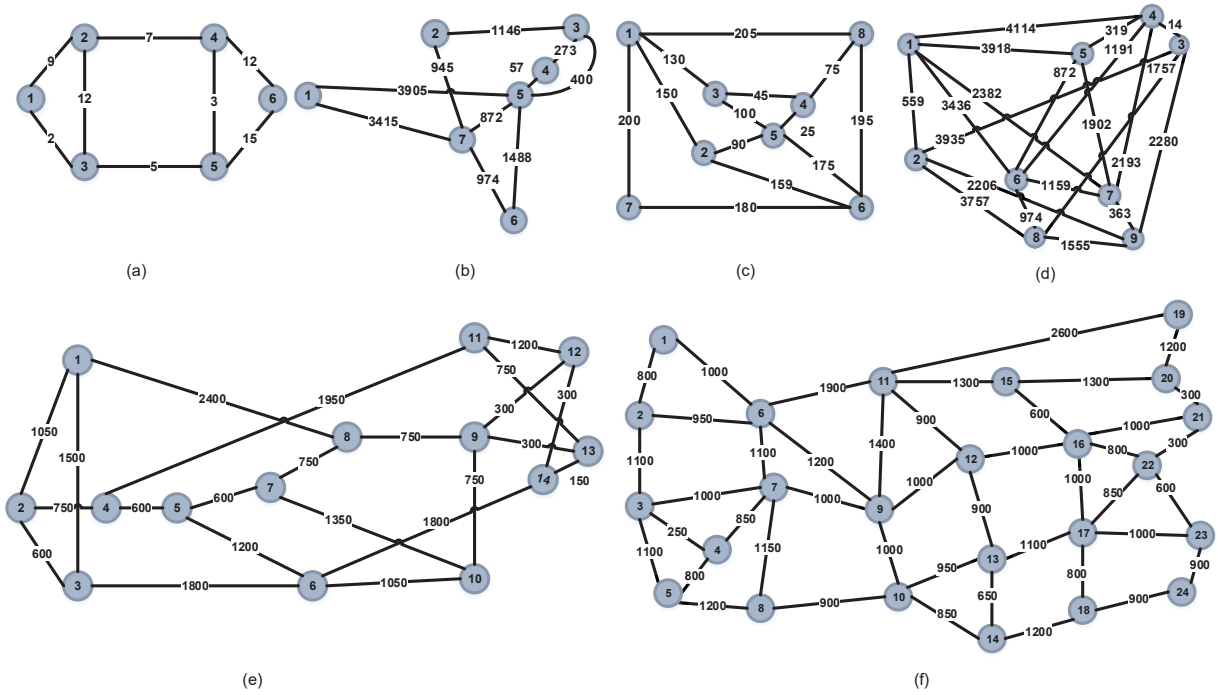

Fig. 4. Topologies used in simulations (link lengths in kilometers), (a) 6-Node, (b) Netrail, (c) 8-Node, (d) GridNet, (e) NSFNET, and (f) US-Backbone.

TABLE I
PERFORMANCE OF ALGORITHMS ON CONTROL PLANE DESIGN IN CONSIDERATION OF PLANNED PHYSICAL-LAYER ATTACKS

	Cutting-Plane-based Algorithm			Bellman-based Algorithm			Approximate Algorithm			
	$\mathcal{C}_{\max}$	Length (km)	Running Time (s)	$\mathcal{C}_{\max}$	Length (km)	Running Time (s)	$\mathcal{C}_{\max}$	Length (km)	Running Time (s)	Gap
6-Node	1	112	15	1	112	3.3	1	112	22	0
Netrail	–	–	–	1	11328	45	1	23014	25	0
8-Node	–	–	–	1	2698	3507	2	3350	2.4	1
GridNet	–	–	–	–	–	–	2	58520	19	–
NSFNET	–	–	–	–	–	–	3	193800	300	–
USB	–	–	–	–	–	–	7	262300	282	–

nodes that a controller can manage) as 4 and 6 for 6-Node and Netrail, respectively, and have the approximation ratio $\gamma = 1$.

For the 6-Node topology, all the three algorithms output the same results on $\mathcal{C}_{\max} = 1$ and total length of control channels, as shown in Table I. In this scenario, the Bellman-based algorithm is the most time-efficient, while the Approximation algorithm takes the longest running time. This is because both the Cutting-Plane-based and Approximation algorithms solve the problem in iterations, which can take longer time. Regarding the Netrail topology, the Cutting-Plane-based algorithm becomes intractable due to its time complexity, while the solution from the Bellman-based algorithm is still better than that from the Approximation algorithm (*i.e.*, the value of $\mathcal{C}_{\max}$ is the same but the total length of control channels is shorter). However, the Approximation algorithm starts to show its advantage on time efficiency, and its running time is less than 56% of that of the Bellman-based algorithm.

2) Medium-scale Topologies: The medium-scale topologies are the 8-Node and GridNet in Figs. 5(c) and 5(d), respectively. This time, we set $|V'| = 3$ (*i.e.*, the optical nodes in V' are still randomly selected), and select $\gamma = 2$. For the 8-Node topology, we have $\mathcal{L} = 6$, while the value of $\mathcal{L}$ is set as 8 for GridNet. The results in Table I suggest that the Bellman-based algorithm can still solve the control plane design for 8-Node, but its running time is very long (*i.e.*, $3,507$ seconds). On the other hand, the Approximation algorithm can solve the problem much more time-efficiently, and with an approximation ratio of $\gamma = 2$, it only takes 2.4 seconds to tackle the control plane design in 8-Node. Although the Bellman-based algorithm is intractable for GridNet, the running time of the Approximation algorithm for the scenario is 19 seconds to satisfy the requirement of $\gamma = 2$.

3) Large-scale Topologies: The remaining two topologies (*i.e.*, the NSFNET and USB in Figs. 5(e) and 5(f)) are the large-scale ones. For the NSFNET topology, we have $|V'| = 4$, $\mathcal{L} = 10$ and $\gamma = 3$, while the settings are $|V'| = 8$, $\mathcal{L} = 14$ and $\gamma = 3.5$ in USB. Here, we use larger values of γ to ensure that the control plane design in the large-scale topologies can

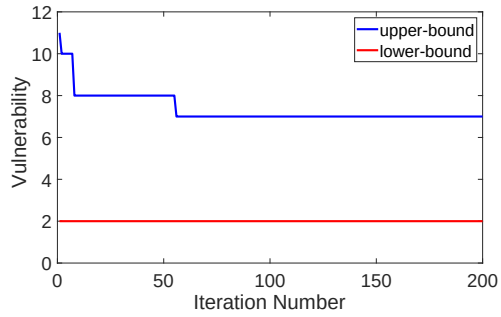

Fig. 5. Convergence performance of Approximation algorithm for USB.

be solved within reasonably long running time. In Table I, we can see that the Approximation algorithm solves the problems in minutes, while both the Cutting-Plane and Bellman-based algorithms are intractable.

B. Control Plane Design with Approximation Algorithm

To further verify the effectiveness of the Approximation algorithm, we consider the largest topology (*i.e.*, USB) and show the specific control plane design achieved by it. This time, we set $V' = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $\mathcal{L} = 14$, and $\gamma = 3.5$. Fig. 5 shows the algorithm's convergence performance, where the lower-bound on the vulnerability $\mathcal{C}_{\max}$ is obtained by the LP-relaxation and each feasible solution found in an iteration gives an upper-bound. We observe that our algorithm converges after 56 iterations to satisfy the approximation ratio of $\gamma = 3.5$. The specific control plane design for the SDON using the USB topology is listed in Table II. It can be seen that our algorithm obtains the number of controllers as 9 to approximate the bilevel optimization. The placements of the controllers and how to assign them as the primary/backup controllers of optical nodes in the SDON are also explained in Table II. Then, the control plane design $G'(V, E')$ lists all the links in E', which are selected to route control channels such that the vulnerability of the design control plane can be minimized. Next, we show the routing paths of all the control channels, where "C-S" and "C-C" means that the control channels are for controller-to-switch and controller-to-controller, respectively. Here, all the routing paths are duplex.

For the control plane design in Table II, *Link* $(22, 23)$ and *Link* $(22, 16)$ carry $\mathcal{C}_{\max} = 7$ control channels, and thus they are the most vulnerable links. It can be seen that instead of sitting in the middle of the USB topology, the most vulnerable links are actually close to the edge. This suggests that by solving the bilevel optimization, we successfully distribute control channels evenly in the physical topology of an SDON, and thus they will not concentrate on the links whose betweenness centrality is high, as in the solutions provided by the conventional single-level optimization in [33]. Therefore, the effectiveness of our algorithm can be further verified.

TABLE II
CONTROL PLANE DESIGN FOR SDON USING THE USB TOPOLOGY

# of Controllers	9
Nodes w/ controller	9, 11, 13, 15, 18, 19, 21, 23, 24
Primary controller assignments	{15: 1, 6, 15, 20}; {9: 2, 4, 8, 9, 16}; {19: 19}; {18: 10, 24}; {21: 12, 21}; {13: 13, 14}; {11: 3, 5, 11, 17, 18}; {23: 7, 22, 23}.
Backup controller assignments	{24: 1, 3, 5, 6, 9, 14, 17, 18, 20}; {21: 2, 13}; {9: 7, 10, 11, 12, 15, 19, 21, 22, 23, 24}; {15: 16}; {23: 8}; {18: 4}.
Control plane design $((u, v) \in E')$	(2, 3), (3, 4), (3, 5), (1, 6), (3, 7), (4, 7), (5, 8), (10, 14), (11, 15), (12, 16), (15, 16), (14, 18), (11, 19), (15, 20), (20, 21), (16, 22), (21, 22), (17, 23), (11, 12), (10, 13), (6, 7), (6, 9), (8, 10), (9, 10), (6, 11), (9, 11), (9, 12), (13, 14), (18, 24), (23, 24), (22, 23), (19, 20).
C-S control channel paths	24-18, 23-22-16-12-9-6-7, 16-12-9, 9-6-7-3-2, 20-15, 18-14-10-9-11, 15-11-6, 14-13, 9-6-7-4, 9-10-8, 17-23-22-16-15-11, 11-9-10-8-5, 23-22, 18-14-10, 11-6-7-3, 21-22-16-12, 15-11-6-1.
C-C control channel paths	19-11-9, 18-14-10-9, 23-22-16-12-9, 15-16-12-9 24-23-22-16-15-11, 24-18-14-10-9, 24-18-14-13 21-22-23-24-18-14-13, 24-23-22-16-15, 11-9, 21-22-16-12-9.
Most vulnerable link $e \in E'$	(22, 23) (22, 16)
$\mathcal{C}_{\max}$	7

VII. CONCLUSION

This paper studied how to design control plane for SDONs in consideration of planned physical-layer attacks. We first modeled the problem as a bilevel optimization and developed two approaches to solve it exactly. Specifically, we not only leveraged the cutting plane method to solve the bilevel model directly, but also transformed it into a single-level MILP model with the Bellman method for problem solving. Then, to improve the time efficiency for large-scale problems, we proposed a polynomial-time approximation algorithm based on LP relaxation and randomized rounding. Our simulations evaluated the proposed algorithms with various physical topologies, demonstrated their effectiveness on control plane design, and confirmed that the approximation algorithm can solve large-scale problems time efficiently to provide solutions whose performance gaps to optimal ones are upper-bounded.

ACKNOWLEDGMENTS

This work was supported in part by the NSFC projects 61871357, 61771445 and 61701472, ZTE Research Fund PA-HQ-20190925001J-1, Zhejiang Lab Research Fund 2019LE0AB01, CAS Key Project (QYZDY-SSW-JSC003), and SPR Program of CAS (XDC02070300).

REFERENCES

[1] "Cisco global cloud index: Forecast and methodology, 2016-2021." [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html

[2] P. Lu *et al.*, "Highly-efficient data migration and backup for Big Data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.

[3] J. Yao, P. Lu, L. Gong, and Z. Zhu, "On fast and coordinated data backup in geo-distributed optical inter-datacenter networks," *J. Lightw. Technol.*, vol. 33, pp. 3005–3015, Jul. 2015.

[4] Z. Zhu *et al.*, "Demonstration of cooperative resource allocation in an OpenFlow-controlled multidomain and multinational SD-EON testbed," *J. Lightw. Technol.*, vol. 33, pp. 1508–1514, Apr. 2015.

[5] A. Castro *et al.*, "Brokered orchestration for end-to-end service provisioning across heterogeneous multi-operator (multi-AS) optical networks," *J. Lightw. Technol.*, vol. 34, pp. 5391–5400, Dec. 2016.

[6] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–98, Apr. 2014.

[7] S. Li *et al.*, "Protocol oblivious forwarding (POF): Software-defined networking with enhanced programmability," *IEEE Netw.*, vol. 31, pp. 12–20, Mar./Apr. 2017.

[8] L. Liu *et al.*, "OpenSlice: an OpenFlow-based control plane for spectrum sliced elastic optical path networks," *Opt. Express*, vol. 21, pp. 4194–4204, Feb. 2013.

[9] A. Thyagaturu *et al.*, "Software defined optical networks (SDONs): A comprehensive survey," *IEEE Commun. Surveys Tut.*, vol. 18, pp. 2738–2786, Fourth Quarter 2016.

[10] Z. Zhu *et al.*, "Build to tenants' requirements: On-demand application-driven vSD-EON slicing," *J. Opt. Commun. Netw.*, vol. 10, pp. A206–A215, Feb. 2018.

[11] Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *J. Lightw. Technol.*, vol. 31, pp. 15–22, Jan. 2013.

[12] L. Gong *et al.*, "Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. 836–847, Aug. 2013.

[13] Y. Yin *et al.*, "Spectral and spatial 2D fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. A100–A106, Oct. 2013.

[14] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[15] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, pp. 3330–3341, Jul. 2016.

[16] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding (LC-VNE) algorithms towards integrated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 3648–3661, Dec. 2016.

[17] J. Liu *et al.*, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, pp. 543–553, Sept. 2017.

[18] C. Chen *et al.*, "Demonstrations of efficient online spectrum defragmentation in software-defined elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 4701–4711, Dec. 2014.

[19] Z. Zhu *et al.*, "OpenFlow-assisted online defragmentation in single-/multi-domain software-defined elastic optical networks," *J. Opt. Commun. Netw.*, vol. 7, pp. A7–A15, Jan. 2015.

[20] X. Chen *et al.*, "Leveraging master-slave OpenFlow controller arrangement to improve control plane resiliency in SD-EONs," *Opt. Express*, vol. 23, pp. 7550–7558, Mar. 2015.

[21] B. Zhao, X. Chen, J. Zhu, and Z. Zhu, "Survivable control plane establishment with live control service backup and migration in SD-EONs," *J. Opt. Commun. Netw.*, vol. 8, pp. 371–381, Jun. 2016.

[22] "Wavelength-division multiplexing." [Online]. Available: https://en.wikipedia.org/wiki/Wavelength-division_multiplexing.

[23] X. Chen, S. Zhu, L. Jiang, and Z. Zhu, "On spectrum efficient failure-independent path protection p-cycle design in elastic optical networks," *J. Lightw. Technol.*, vol. 33, pp. 3719–3729, Sept. 2015.

[24] J. Zhu, B. Zhao, and Z. Zhu, "Attack-aware service provisioning to enhance physical-layer security in multi-domain EONs," *J. Lightw. Technol.*, vol. 34, pp. 2645–2655, Jun. 2016.

[25] N. Skorin-Kapov, M. Furdek, S. Zsigmond, and L. Wosinska, "Physical-layer security in evolving optical networks," *IEEE Commun. Mag.*, vol. 54, pp. 110–117, Aug. 2016.

[26] J. Zhu and Z. Zhu, "Physical-layer security in MCF-based SDM-EONs: Would crosstalk-aware service provisioning be good enough?" *J. Lightw. Technol.*, vol. 35, pp. 4826–4837, Nov. 2017.

[27] J. Zhu, B. Zhao, and Z. Zhu, "Leveraging game theory to achieve efficient attack-aware service provisioning in EONs," *J. Lightw. Technol.*, vol. 35, pp. 1785–1796, May 2017.

[28] C. Natalino, A. Yayimli, L. Wosinska, and M. Furdek, "Infrastructure upgrade framework for content delivery networks robust to targeted attacks," *Opt. Switch. Netw.*, vol. 31, pp. 202–210, Jan. 2019.

[29] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Prof. of HotSDN 2012*, pp. 7–12, Aug. 2012.

[30] D. Hock *et al.*, "Pareto-optimal resilient controller placement in SDN-based core networks," in *Proc. of ITC 2013*, pp. 1–9, Sept. 2013.

[31] H. Li, P. Li, S. Guo, and A. Nayak, "Byzantine-resilient secure software-defined networks with multiple controllers in cloud," *IEEE Trans. Cloud Comput.*, vol. 2, pp. 436–447, Oct. 2014.

[32] H. Huang *et al.*, "Realizing highly-available, scalable and protocol-independent vSDN slicing with a distributed network hypervisor system," *IEEE Access*, vol. 6, pp. 13 513–13 522, 2018.

[33] J. Zhu *et al.*, "Control plane robustness in software-defined optical networks under targeted fiber cuts," in *Proc. of ONDM 2018*, pp. 118–123, May 2018.

[34] Q. Lv, J. Zhu, F. Zhou, and Z. Zhu, "Network planning with bilevel optimization to address attacks to physical infrastructure of SDN," in *Proc. of ICC 2020*, pp. 1–6, Jun. 2020.

[35] F. Gzara, "A cutting plane approach for bilevel hazardous material transport network design," *Oper. Res. Lett.*, vol. 41, pp. 40–46, Jan. 2013.

[36] L. Giovanni, F. Croce, and R. Tadei, "On the impact of the solution representation for the Internet Protocol network design problem with max-hop constraints," *Netw.*, vol. 44, pp. 73–83, Sept. 2004.

[37] Y. Zhang, N. Beheshti, and M. Tatipamula, "On resilience of split-architecture networks," in *Proc. of GLOBECOM 2011*, pp. 1–6, Dec. 2011.

[38] P. Vizarreta, C. Machuca, and W. Kellerer, "Controller placement strategies for a resilient SDN control plane," in *Proc. of RNDM 2016*, pp. 253–259, Sept. 2016.

[39] P. Mohan, T. Truong-Huu, and M. Gurusamy, "Primary-backup controller mapping for Byzantine fault tolerance in software defined networks," in *Proc. of GLOBECOM 2017*, pp. 1–7, Dec. 2017.

[40] M. Bari *et al.*, "Dynamic controller provisioning in software defined networks," in *Proc. of CNSM 2013*, pp. 18–25, Oct. 2013.

[41] H. Ammar, Y. Nasser, and A. Kayssi, "Dynamic SDN controllers-switches mapping for load balancing and controller failure handling," in *Proc. of ISWCS 2017*, pp. 216–221, Aug. 2017.

[42] T. Koponen *et al.*, "Onix: A distributed control platform for large-scale production networks," in *Proc. of USENIX/OSDI 2010*, pp. 1–6, Oct. 2010.

[43] Z. Yang and K. Yeung, "An efficient algorithm for constructing controller trees in SDN," in *Proc. of GLOBECOM 2017*, pp. 1–6, Dec. 2017.

[44] J. Zhu *et al.*, "How to survive targeted fiber cuts: A game theoretic approach for resilient SDON control plane design," in *Proc. of ONDM 2019*, pp. 1–6, May 2019.

[45] N. Dayal, P. Maity, S. Srivastava, and R. Khondoker, "Research trends in security and DDoS in SDN," *Secur. Commun. Netw.*, vol. 9, pp. 6386–6411, Dec. 2016.

[46] Y. Tian, R. Dey, Y. Liu, and K. Ross, "Topology mapping and geolocating for China's Internet," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, pp. 1908–1917, Sept. 2013.

[47] S. Liu, W. Lu, and Z. Zhu, "On the cross-layer orchestration to address IP router outages with cost-efficient multilayer restoration in IP-over-EONs," *J. Opt. Commun. Netw.*, vol. 10, pp. A122–A132, Jan. 2018.

[48] S. Liu *et al.*, "DL-assisted cross-layer orchestration in software-defined IP-over-EONs: From algorithm design to system prototype," *J. Lightw. Technol.*, vol. 37, pp. 4426–4438, Sept. 2019.

[49] J. Bard, "Some properties of the bilevel programming problem," *J. Optimiz. Theory App.*, vol. 68, pp. 371–378, Feb. 1991.

[50] D. Dubhashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.