Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Reliable edge-to-core optical networks: An optimal algorithm for maximal path diversity

Ali Shakeri, Tianliang Zhang *, Shunmugapriya Ramanathan, Miguel Razo, Marco Tacca, Andrea Fumagalli

Open Networking Advanced Research (OpNeAR) Lab, UT Dallas, TX, USA

ARTICLE INFO

Keywords:
Network reliability
Multi-hub MAN
Edge to core network
Max-flow

ABSTRACT

With the emergence of IoT applications, 5G, and edge computing, network resource allocation has shifted toward the edge, bringing services closer to the end users. These applications often require communication with the core network for purposes that include cloud storage, compute offloading, 5G-and-Beyond transport communication between centralized unit (CU), distributed unit (DU) and core network, centralized network monitoring and management, etc. As the number of these services increases, efficient and reliable connectivity between the edge and core networks is of the essence. Wavelength Division Multiplexing (WDM) is a well-suited technology for transferring large amounts of data by simultaneously transmitting several wavelength-multiplexed data streams over each single fiber optics link. WDM is the technology of choice in mid-haul and long-haul transmission networks, including edge-to-core networks, to offer increased transport capacity.

Optical networks are prone to failures of components such as network fiber links, sites, and transmission ports. A single network element failure alone can cause significant traffic loss due to the disruption of many active data flows. Thus, fault-tolerant and reliable network designs remain a priority. The architecture called "dual-hub and dual-spoke" is often used in metro area networks (MANs). A dual-hub, or in general a multi-hub network, consists of a set of designated destination nodes (hubs) in which the data traffic from all other nodes (the peripherals) should be directed to the hubs. Multiple hubs offer redundant connectivity to and from the core or wide area network (WAN) through geographical diversity. The routing of the connections (also known as lightpaths) between the peripheral node and the hubs has to be carefully computed to maximize path diversity across the edge-to-core network. This means that whenever possible the established redundant lightpaths must not contain a common Shared Risk Link Group (SRLG).

An algorithm is proposed to compute the most reliable set of SRLG disjoint shortest paths from any peripheral to all hubs. The proposed algorithm can also be used to evaluate the overall edge-to-core network reliability quantified through a newly introduced figure of merit.

1. Introduction

By leveraging function virtualization and software-defined networking, the provisioning of network resources and services is evolving to become user-experience-oriented. Edge computing, for instance, brings compute resources closer to users, resulting in reduced network latency, improved real-time performance, and enhanced security. Similarly, in 5G technologies, network slicing is employed to allocate services requiring low latency close to users, while latency-tolerant services may be positioned in the cloud to optimize network resource efficiency and utilization. Other applications, such as distributed training of large-scale AI models across multiple network nodes exemplify the growing importance of edge-centric computing.

Supporting these and other edge-centric services requires robust data transport communication between the edge (i.e., the peripheral node) and the gateway (the hub) connecting to the core network. Tasks such as data transfer and backup, network and device monitoring, and centralized management all rely on high-volume communication between the edge and the core. Furthermore, traditional multimedia entertainment activities, network streaming media, and similar services are expanding their reach to accommodate a large (mobile) user base, resulting in record-high demands for data transport capacity in metro networks.

Optical transport networks offer high data rates, signal low power loss, long-distance connectivity, low-security risks, and small cable size

* Corresponding author.
E-mail address: tianliang.zhang@ieee.org (T. Zhang).

https://doi.org/10.1016/j.comnet.2024.110268
Received 31 October 2023; Received in revised form 20 January 2024; Accepted 16 February 2024

compared to copper cables. In addition, Wavelength Division Multiplexing (WDM) can increase the overall transport capacity of the currently existing fiber plants by transmitting several concurrent and orthogonal data signals, each signal being carried by one distinct wavelength or channel in a single fiber medium [1,2]. Current technology allows each fiber to carry tens of wavelengths where each wavelength can transmit data at 400G, 800G, and even higher rates.

Moreover, the wavelength signals are protocol and bit-rate independent, enabling multiple and diverse upper-layer solutions to be supported by the same optical fiber network. Each wavelength signal can support multiple data traffic flows using time division multiplexing (TDM) or data packet switching (IPoverDWDM). Recently, open optical networking early field deployments have been announced, enabling network operators to seamlessly interconnect optical network devices from multiple vendors while keeping a single network controller to operate a wide range of optical devices [3]. This and other similar open network architectures will further ameliorate operators' ability to flexibly upgrade their metro network transport capacity.

Higher transport capacity and protocol transparency imply that each fiber cable most likely carries data traffic from several different applications. As a result, a single fiber cable failure can cause loss of data for multiple and different services leading to non-tolerable interruption of service in most cases [4]. For example, in disaggregated 5G RAN architectures, both fronthaul and backhaul network resiliency plays an important role in specific 5G use cases, such as ultra-reliable low latency applications. The impact of fiber cable failures along with an example of a related resiliency mechanism mitigating the ensuing service disruption are discussed in [5]. Understandably, transport network reliability is of the utmost importance to many existing and future mobile services.

One of the best solutions to avoid or minimize traffic loss in the presence of a fiber cable failure is to provide each service with a working path that carries the service's traffic and one or more redundant (backup) paths that can replace the working path in the presence of an outage [6–8]. The working and backup paths should not be in the same Shared Risk Link Group (SRLG), or single point of failure. This allows the network to ensure that the backup paths are not disrupted by the same network outage that disrupts the working path. In addition, to protect the service from multiple failures, the backup paths must be SRLG-disjoint from each other, e.g., if N SRLG-disjoint paths are allocated to a service, the service can then survive up to $N-1$ distinct outages. It is obvious that this approach improves the network reliability but it also increases the network cost, as more SRLG-disjoint network resources must be provisioned. Intuitively, there is a trade-off between minimizing the protection cost and maximizing the service reliability level as presented in [9,10] and [11] for optical edge networks. The next paragraph provides a brief description of related work published previously.

The authors in [12] propose an algorithm to compute the vertex-disjoint paths in an undirected graph. The algorithm is proven to complete in polynomial time with the number of disjoint paths set to two ($K=2$). Over time, the $K>2$ vertex-disjoint shortest path problem was proven to admit a polynomial run time [13,14]. Our approach distinguishes itself from these prior studies as we specifically explore link-disjoint solutions originating from a single node and terminating at multiple destination nodes. Significant efforts have been dedicated to identifying link-disjoint paths, with each study focusing on specific conditions such as bounded computational cost and scalability [15], dual link cost in the network [16], optimized parameter K for balanced computational time and practical solutions [17], etc. While these prior studies are practical and continue to hold great potential for many applications, our goal is slightly different in that it specifically focuses on identifying K link-disjoint paths heading to multiple destination nodes (the hubs), while considering the crucial aspect of traffic load balancing. We believe that combining concepts already presented in these prior papers with our approach is beneficial to future research.

The focus of this study is to define a figure of merit for the design of a highly reliable edge-to-core network, which is characterized by two or more hubs. A multi-hub network is a network architecture with a set of designated destination nodes (termed hubs) in which the traffic from all the other nodes (termed peripherals) must be directed to the hubs. Each hub offers connectivity to and from the core network independently of the other hubs. Hubs are geo-diverse, thus reducing the risk that a common event can disrupt all of the hubs at once.

An algorithm is proposed to compute an *optimal topologically sorted directed acyclic subgraph*, from which multiple paths to the hubs optimizing the reliability figure of merit can be obtained straightforwardly. The following assumptions are made in this study. First, only fiber link outages are considered, while network node outages are ignored. Second, fiber links in the network have the same probability of being affected by an outage. These two assumptions simplify the description of the proposed figure of merit and optimization algorithm. However, both figures of merit and optimization algorithms can be adjusted to account for the relaxation of these two assumptions. Summarizing, the two contributions of this work are:

- Introducing a new figure of merit, which rigorously defines the achievable reliability and cost-effectiveness of optical edge-to-core network connectivity in the presence of limited fiber diversity availability.
- Proposing an efficient algorithm that leverages optimal directed acyclic subgraph to find the most reliable set of shortest SRLG-disjoint paths from any peripheral node in the edge to all hubs connecting to the core.

The algorithm also provides an optimal approach for selecting hub locations, managing network fiber resources, and overall maximizing reliability in multi-hub optical edge-to-core networks.

The rest of the paper is organized as follows. First, a multi-hub optical edge-to-core network is defined. Then, we use disaggregated 5G RAN architecture as an example to show the importance of adding resiliency to the transport network. We then define a figure of merit to quantify the network reliability and introduce the algorithm that calculates the most reliable set of K shortest paths. Finally, the feasibility of the algorithm is demonstrated on a typical network topology.

2. Multi-hub edge-to-core optical transport network

Fig. 1 shows an abstraction of the edge-to-core optical transport network[1] that interconnects a range of users – such as businesses, banks, data centers, 5G RU/DU/CU, and edge applications – in a geographic area that is serviced by the edge-to-core network. For example, optical edge-to-core networks are the networks deployed in cities providing connectivity to the users in each city. These networks are also interconnected through the core network (e.g., WANs) to connect users from different cities. In other words, each network should have one or more designated network nodes connected to the core network, which serve as network gateways to other transport networks. We refer to these nodes as hubs and all other nodes as peripherals. An optical edge-to-core network hosting multiple hubs is referred to as multi-hub.

Depending on the service requests, having more than one hub in the network can not only increase service reliability but also improve traffic load balance. To be more specific, it increases network reliability by avoiding a single point of failure situation, e.g., if one of the hubs fails, the traffic can still be switched in and out of the network through the other still operational hubs. It also increases load balancing by distributing the traffic across multiple hubs. The service requests can demand both load balancing and protection for which there should be more than one connection to each hub. In summary, for protected

[1] Throughout this study we use the term *network* to refer to the Edge-to-Core Optical Transport Network.

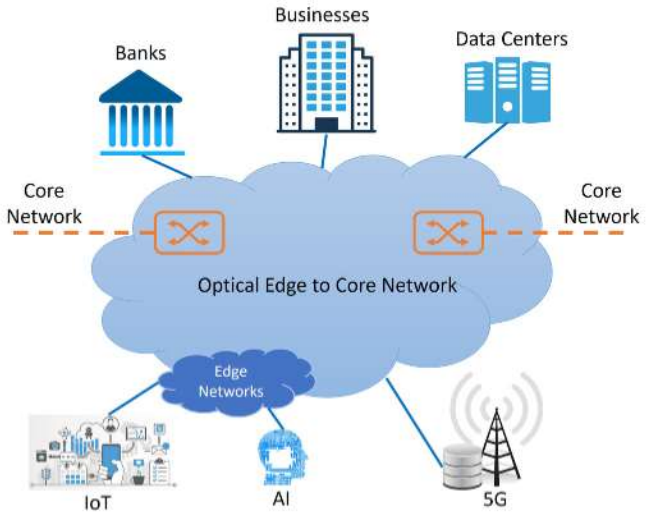

Fig. 1. Multi-hub edge to core optical transport network.

service requests between a peripheral node within the network as the source node and a destination node outside the network, the corresponding peripheral node must at least establish two circuits to each hub.

In this study, we investigate the reliability of the multi-hub network while accounting for the previously mentioned protected service requirements and traffic load balancing.

3. Disaggregated 5G RAN and related transport network

In accordance with the 3GPP TS 38.300 specifications [18], the 5G disaggregated RAN architecture consists of gNB-CU, gNB-DU, and RU, respectively. Disaggregation of the RAN effectively splits the RAN protocol stack so that the individual components can be realized independently. For instance, the gNB-CU centralizes the packet processing functions, realizes them as virtualized network functions running on commodity hardware, and places them in geographically centralized telco edge cloud locations. The gNB-CU communicates with the Core Network placed in the central cloud. Fig. 2 shows the disaggregated RAN architecture where the gNB-CU is further split into gNB-CU-CP and gNB-CU-UP. This RAN disaggregation has brought about many advantages, including the scalability of user plane operations, the ability to design user planes, the capacity to maintain and upgrade individual subsystems, etc. It also reduces the operational and maintenance costs of the Mobile Network Operators (MNOs) by hosting the *control* and *user* plane nodes in different geographical locations [19].

As shown in Fig. 2, Fronthaul refers to the link between the RU and the gNB-vDU, Midhaul refers to the link between gNB-vDU and the gNB-vCU, and Backhaul refers to the link between the gNB-vCU and the Core Network. Depending on the 5G use cases – e.g., enhanced Mobile Broadband (eMBB), Ultra Reliable Low-Latency Communication (URLLC) or massive Machine Type Communication (mMTC) – the RAN components can be placed closer to the cell site, edge cloud, or in the central cloud. A typical deployment accounts for 100k RUs, 10k gNB-DUs, 200 edge clouds, and 10 s of central clouds [20]. In summary, the 5G transport network that provides connection service for the 5G network must provide ultra-high bandwidth, ultra-low latency, flexible and highly resilient network. For instance, a data plane link failure can make the user-data traffic link unavailable, thereby disconnecting all the connected mobile users. A study on enhancing the resiliency of 5G disaggregated RAN is discussed in 3GPP TR 38.879 [21].

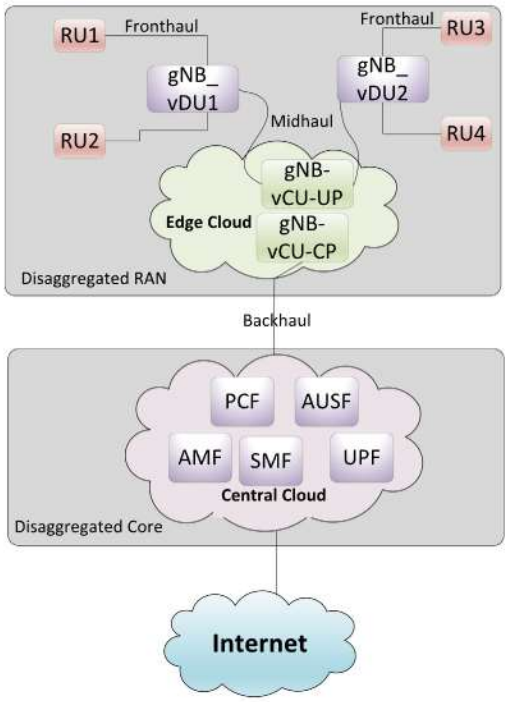

Fig. 2. Disaggregated RAN architecture.

4. Network reliability model

In this section, we define a figure of merit chosen to measure the network reliability level. Based on the description in Section 2, the following constraints apply.

- There are no service requests between two peripheral nodes. This limits the peripherals to only be the source node for any service request.
- Peripheral nodes must at least establish two circuits to each hub. In a network with H hubs there must be at least $2 \times H$ circuits established from the service request's peripheral node to all hubs.

We evaluate the reliability of the multi-hub network, based on the hubs' reachability from any given peripheral node. Let us define reachability as the number of fiber-disjoint paths from the peripheral to the hubs. The ideal reliability for a peripheral is when all required paths for the protected service request are fiber link-disjoint. This problem can be easily solved with the network maximal flow algorithms such as Ford–Fulkerson and Edmonds–Karp [22–25]. However, there are not always enough network resources to satisfy this ideal reliability requirement. For example, consider a protected service request that requires K fiber-disjoint paths but the maximum number of fiber-disjoint paths from its corresponding peripheral to the hubs is less than K. In this case, defining the reliability based on the number of fiber-disjoint paths does not provide enough information to assess reliability. To address this issue, we redefine the reliability level by taking into account the shared fiber links. In order to do so, we define a figure of merit that classifies the fiber links based on the number of times they are selected by the computed paths. Let us define the vector $\mathcal{L} = [l_1, l_2, \ldots, l_K]$, where l_i indicates the number of the fiber links that are assigned to exactly i path(s) and K is the total number of paths required by the protected service request. We refer to $\mathcal{L}$ as the peripheral reliability vector with respect to the network hubs.

The reliability vector is used to evaluate the reliability of any routing algorithm used to compute the required paths between the peripheral and all hubs. The ideal reliability for a peripheral node can

now be described with a reliability vector whose all elements are zero except the first one ($l_1 > 0$). However, as mentioned earlier, this is not the case in most deployed networks due to a lack of network resources. In this case the vector $\mathcal{L}$ evaluates the reliability based on the number of fiber links needed to be added to the network to reach the ideal reliability, under the following assumptions. The cost of the fiber links are equal and we can only add fiber links between adjacent nodes. For example, let the peripheral reliability vector for a protected service request demanding three paths to each hub in a two-hub network be $[13, 3, 4, 0, 0, 0]$. From the vector, we can conclude that $l_2 = 3$ and $l_3 = 4$ fiber links are shared by $i = 2$ and $i = 3$ paths, respectively. In order to reach the ideal reliability, for every fiber link that is shared by two paths we need to add an extra fiber link ($l_2 \times 1 = 3$) and for every fiber link that is shared by three paths we need to add two extra fiber links ($l_3 \times 2 = 8$). Therefore, by upgrading the network with 11 $(3 + 8)$ additional fiber links between the proper nodes the ideal reliability is reached. This means that the network meets the ideal reliability with the cost of 11 fiber links. We refer to this cost as $cost_{ideal}$,

$$cost_{ideal} = \sum_{i=1}^{K} l_i \times (i - 1), \tag{1}$$

where l_i is the ith element in the vector $\mathcal{L}$ and K is the total number of paths required by the protected service request.

Given the vector $\mathcal{L}$ as the reliability figure of merit, an objective function can be defined to optimize it. This objective function is to minimize the l_is starting from $i = K$ to $i = 1$. This choice forces the optimization algorithm to prefer sets of paths that do not excessively rely on the same fiber link. For example, between the two solutions $\mathcal{L}_1 = [6, 3, 2]$ and $\mathcal{L}_2 = [6, 5, 1]$, the latter is preferred as only one ($l_3 = 1$ in $\mathcal{L}_2$) fiber link can disrupt three paths at once when failing, while the former has two ($l_3 = 2$ in $\mathcal{L}_1$). In the next section, we present an algorithm that computes an optimal topologically sorted directed acyclic subgraph, from which multiple routing solutions to the hubs can be obtained straightforwardly. Any routing solution obtained from the aforementioned subgraph is optimal with respect to the objective function defined earlier.

5. Computing the most reliable K shortest paths

In this section, we present an algorithm that computes an optimal topologically sorted directed acyclic subgraph from any given peripheral to all hubs. We refer to this subgraph as *optimal directed acyclic subgraph*. From this subgraph, possible sets of routing solutions can be obtained. We refer to each possible set of routing solutions as $r_i \in \mathcal{R}$. Every r_i contains K most reliable shortest paths which optimize the vector $\mathcal{L}$ according to the objective function defined in the last paragraph of Section 4.

The following steps summarize how the algorithm works.

1. Compute all shortest fiber-disjoint paths using Edmonds–Karp algorithm [25].
2. If all K shortest fiber-disjoint paths are found, then the algorithm stops and the result is the most reliable set of K shortest paths.
3. If less than K shortest paths are found in the previous step the remaining paths must use at least one of the fiber links that are already assigned to the disjoint paths found so far. Therefore, the algorithm must compute the remaining non-disjoint paths in this second step to optimize the vector $\mathcal{L}$ according to the objective function defined in Section 4.

In the rest of this section, we explain how the algorithm computes the optimal directed acyclic subgraph. Table 1 lists the variables used to describe the algorithm. Let the directed graph $G(\mathcal{V}, \mathcal{E}, \mathcal{H})$ with $V = |\mathcal{V}|$ vertices, $H = |\mathcal{H}|$ hubs, and $E = |\mathcal{E}|$ edges describe the optical network with V nodes, H hubs, and $E/2$ bidirectional fiber links. Let us add an extra vertex and H more edges to the graph G. We refer to the extra vertex as the sink node v_{sink}, and the set of extra edges as $\mathcal{E}'$. Each extra edge $e'(h_d, v_{sink}) \subset \mathcal{E}'$ connects one of the hubs $h_d \in \mathcal{H}$ to v_{sink}. Note that the extra edges are only in the direction from the hubs to the sink node. In the new graph we need to find the paths between any peripherals $p_s \in \mathcal{P}$ and v_{sink}. Every time a path is found, one of the extra edges is used, e.g., if the extra edge $e'(h_d, v_{sink})$ is used n times, we can conclude that n paths from the peripheral to the hub h_d are found. In order to find the same number of paths to all hubs, we need to make sure that all of the extra edges are equally used.

Table 1
List of variables and their definitions.

Variable	Definition
$G(\mathcal{V}, \mathcal{E}, \mathcal{H})$	directed graph with $\lvert\mathcal{V}\rvert$ vertices, $\lvert\mathcal{H}\rvert$ hubs, and $\lvert\mathcal{E}\rvert$ edges
$\mathcal{V}$	set of vertices in graph G
v_i	vertex i in graph G, where $v_i \in \mathcal{V}$
$\mathcal{E}$	set of edges in graph G
$e(v_i, v_j)$	the edge connecting v_i to v_j in graph G, where $e(v_i, v_j) \in \mathcal{E}$
$\mathcal{H}$	set of vertices in graph G representing the hubs, where $\mathcal{H} \subset \mathcal{V}$
h_d	vertex assigned as the hub in graph G, where $h_d \in \mathcal{H}$
$\mathcal{P}$	set of vertices in graph G representing the peripherals, where $\mathcal{P} \subset \mathcal{V} - \mathcal{H}$
p_s	vertex assigned as the peripheral in graph G, where $p_s \in \mathcal{P}$
v_{sink}	vertex added to graph G, where $v_{sink} \notin \mathcal{V}$
$\mathcal{E}'$	set of edges added to graph G, where $\mathcal{E}' \not\subset \mathcal{E}$
$e'(h_d, v_{sink})$	edge added to graph G connecting h_d to v_{sink}, where $e'(h_d, v_{sink}) \subset \mathcal{E}'$
$c(v_i, v_j)$	cost of the directed edge from v_i to v_j
$f(v_i, v_j)$	flow of the directed edge from v_i to v_j
$P^{(k)}(p_s, v_{sink})$	kth path from p_s to v_{sink}
$\mathcal{E}_{P^{(k)}(p_s, v_{sink})}$	set of edges in kth path from p_s to v_{sink}, where $\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \subseteq (\mathcal{E} \cup \mathcal{E}')$

Table 2
Possible values for $f(v_i, v_j)$.

$f(v_i, v_j)$	Definition
> 0	the edge $e(v_i, v_j)$ is used $f(v_i, v_j)$ times
$= 0$	the edge $e(v_i, v_j)$ is not used
< 0	the edge $e(v_j, v_i)$ is used $\lvert f(v_i, v_j)\rvert$ times

Let $c(v_i, v_j)$ and $f(v_i, v_j)$ be the cost and the flow of the edge $e(v_i, v_j)$, respectively. $f(v_i, v_j)$ indicates the number of times the edge $e(v_i, v_j)$ is chosen. Table 2 lists the possible flow values and their definition.

The initial value of the cost and the flow of edges in G are shown in Eqs. (2) and (3), respectively. Assuming that the shortest path metric is based on hop-count, the initial cost values are initially set to one.

$$\begin{cases} c(v_i, v_j) = 1, & \forall\ e(v_i, v_j) \in \mathcal{E} \\ c(h_d, v_{sink}) = 1, & \forall\ e'(h_d, v_{sink}) \in \mathcal{E}' \end{cases} \tag{2}$$

$$\begin{cases} f(v_i, v_j) = 0, & \forall\ e(v_i, v_j) \in \mathcal{E} \\ f(h_d, v_{sink}) = 0, & \forall\ e'(h_d, v_{sink}) \in \mathcal{E}' \end{cases} \tag{3}$$

The algorithm calculates the shortest path a total of K iterations. Let us refer to the shortest path calculated in the kth iteration as $P^{(k)}(p_s, v_{sink})$ and the edges along this path as $\mathcal{E}_{P^{(k)}(p_s, v_{sink})}$, where $k \in [1, K]$. Note that these paths are not the final optimal paths, as the algorithm may modify them in the forthcoming iterations. The algorithm updates the flow and the cost of the edges selected by the shortest path in each iteration. We explain how the algorithm updates the cost and the flow values at each iteration next.

The flow value is used to keep track of the number of times each edge is assigned to one or multiple paths and enables us to correctly update the cost of the edge. To ensure flow skew symmetry ($f(v_i, v_j) = -f(v_j, v_i)$), every time an edge $e(v_i, v_j)$ is used its flow needs to be incremented and the flow of the edge in the opposite direction ($e(v_j, v_i)$) needs to be decremented. The flow update applied to the edge in the

opposite direction also helps us to properly adjust its cost at a later step. These updates for the edges in the direction of the computed path, the opposite direction of the path, and the extra edge in the path are shown in Eqs. (4), (5), and (6), respectively.

$$f(v_i, v_j) = f(v_i, v_j) + 1, \quad \forall\ e(v_i, v_j) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}), \tag{4}$$

$$f(v_j, v_i) = f(v_j, v_i) - 1, \quad \forall\ e(v_i, v_j) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}), \tag{5}$$

$$f(h_d, v_{sink}) = f(h_d, v_{sink}) + 1, \quad e'(h_d, v_{sink}) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}'). \tag{6}$$

Once the flow values of the edges are updated, their cost values can be updated based on the updated flow values. The cost update for the edges in the path is as in Eq. (7), the cost update for the edges in the opposite direction of the path is as in Eq. (8), and the cost update for the extra edge in the path is as in Eq. (9).

$$c(v_i, v_j) = \begin{cases} 1 & \iff f(v_i, v_j) = 0 \\ V^{f(v_i, v_j)} & \iff f(v_i, v_j) > 0 \\ -V^{(|f(v_i, v_j)|-1)} & \iff f(v_i, v_j) < 0, \end{cases} \quad \forall\ e(v_i, v_j) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}), \tag{7}$$

$$c(v_j, v_i) = \begin{cases} 1 & \iff f(v_j, v_i) = 0 \\ V^{f(v_j, v_i)} & \iff f(v_j, v_i) > 0 \\ -V^{(|f(v_j, v_i)|-1)} & \iff f(v_j, v_i) < 0, \end{cases} \quad \forall\ e(v_i, v_j) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}), \tag{8}$$

$$c(h_d, v_{sink}) = V^{f(h_d, v_{sink})}, \quad e'(h_d, v_{sink}) \in (\mathcal{E}_{P^{(k)}(p_s, v_{sink})} \cap \mathcal{E}'). \tag{9}$$

The rationale behind the proposed equations is discussed in this paragraph. Every time a hub h_d is used in the path the cost value of the extra edge $e'(h_d, v_{sink})$ multiplies by V. This cost update of the extra edges guarantees that the hubs are equally reached. As shown in Table 2, the flow of the edge $e(v_i, v_j)$ holds the information about the edge utilization. The positive flow value indicates that the edge is used by $f(v_i, v_j)$ paths. The flow value of zero indicates that the edge is not or should not be used. The negative flow value indicates that the edge is not used for $f(v_i, v_j)$ times while the edge in the opposite direction $e(v_j, v_i)$ is used by $f(v_j, v_i)$ paths. Recall that the goal of the algorithm is to minimize the l_is in the vector $\mathcal{L}$ starting from $i = K$ to $i = 1$. As shown in Eq. (5) if an edge in one direction is chosen for a new path the flow value of the edge in opposite direction must be decremented. Thus, if the algorithm selects the edges with lowest negative flow values, the positive flow values of the edges in the opposite direction are decremented. Intuitively, the algorithm must always prefer the edges with the most negative flow values when possible.

To instruct the algorithm to prefer these edges, the cost of them must reflect their updated flow values as follows. The edges with the negative flow values must have negative cost values, which make them more appealing to the shortest path algorithm to select them.[2] The edges with zero flow values are assigned cost values of one. The edges with the positive flow values should have positive cost values greater than one, which make them less appealing to the shortest path algorithm to select them. This is done by updating the cost of the edges in the path in both direction. In each iteration, when the edge $e(v_i, v_j)$ is chosen, the cost of the edge in opposite direction, $e(v_j, v_i)$, is set to $c(v_j, v_i) = -c(v_i, v_j)$. This cost update makes the edge $e(v_j, v_i)$ more appealing to the shortest path by giving back the cost of the edge $e(v_i, v_j)$ in the next iterations. The cost of the edge $e(v_i, v_j)$ must be a relatively larger value depending on the $f(v_i, v_j)$ and described next.

- If $f(v_i, v_j) > 0$, then $c(v_i, v_j) = c(v_i, v_j) \times V$,
- If $f(v_i, v_j) = 0$, then $c(v_i, v_j) = 1$,
- If $f(v_i, v_j) < 0$, then $c(v_i, v_j) = c(v_i, v_j)\ /\ V$.

To clarify the cost update values for the edge $e(v_i, v_j)$, assume the graph G with V vertices is in initial state, meaning that all flow and cost values are zero and one, respectively. If the edge $e(v_i, v_j)$ is selected in a path, its cost is updated to a proper larger value so the algorithm will avoid selecting it in future iterations until all alternative paths with lower costs are selected. Straightforwardly, the maximum cost of an alternative path for $e(v_i, v_j)$ is $(V - 1)$. Thus, updating the $c(v_i, v_j)$ from one to V guarantees that in the next iterations if the algorithm wants to reach v_j from v_i, the edge $e(v_i, v_j)$ will not be selected until all possible edges with lower cost values are selected.

The cost of the edge $e(v_j, v_i)$ needs to be updated to "-1" meaning that selecting this edge in the next iterations will give the cost of the edge $e(v_i, v_j)$ back. This negative cost update of -1 guarantees that the graph will not contain any negative loop. By induction, the same conclusion holds true when the edge is selected multiple times, i.e., every time and edge is selected its cost must be increased by V times, e.g., $-V^2$ becomes $-V$ or V becomes V^2. Note that there is an exception when the flow value is zero at the time of updating the cost. In this case the cost value should be reset to one.

Let us clarify this point with an example, assuming that $e(v_i, v_j)$ is selected for the first time in $P^{(k)}(p_s, v_{sink})$ in kth iteration. Based on the flow and the cost update equations reported earlier, the flow and the cost values of the edge $e(v_i, v_j)$ are $f(v_i, v_j) = 1$ and $c(v_i, v_j) = V$, respectively, and the flow and the cost values of the edge $e(v_j, v_i)$ are $f(v_j, v_i) = -1$ and $c(v_j, v_i) = -1$, respectively. In the case of k'^{th} iteration, where $k' > k$, if $e(v_j, v_i)$ is selected in $P^{(k')}(p_s, v_{sink})$, then the flow and the cost values of both edges $e(v_i, v_j)$ and $e(v_j, v_i)$ reset to zero and one, respectively. This means that $e(v_i, v_j)$ and $e(v_j, v_i)$ are not used in the first place (see the middle entry of Table 2). In other words, if $e(v_i, v_j)$ is excluded from $P^{(k)}(p_s, v_{sink})$, then the path is divided in to two parts, the first part is from p_s to v_i ($P_1^{(k)}$) and the second part is from v_j to v_{sink} ($P_2^{(k)}$). The same logic holds true for excluding $e(v_j, v_i)$ from $P^{(k')}(p_s, v_{sink})$ and it results in dividing it in to two parts, the first part is from p_s to v_j ($P_1^{(k')}$) and the second part is from v_j to v_{sink} ($P_2^{(k')}$). Then, $P^{(k)}(p_s, v_{sink})$ and $P^{(k')}(p_s, v_{sink})$ can be swapped at the vertices v_i and v_j to form two new paths as follows, $P_1^{(k)} + P_2^{(k')}$ and $P_1^{(k')} + P_2^{(k)}$.

In summary, the flow and the cost values of the edges in graph G hold the properties shown in Eqs. (10) and (11), respectively.

$$|f(v_i, v_j)| = |f(v_j, v_i)|, \ \forall\ e(v_i, v_j) \in \mathcal{E} \tag{10}$$

$$\frac{c(v_i, v_j)}{c(v_j, v_i)} = \begin{cases} 1 & \iff f(v_i, v_j) = 0 \\ -V & \iff f(v_i, v_j) > 0 \\ -1\ /\ V & \iff f(v_i, v_j) < 0, \end{cases} \quad \forall\ e(v_i, v_j) \in \mathcal{E}. \tag{11}$$

When all initial K iterations are completed, the algorithm makes use of the flow values to exclude from the graph the edges with the flow values less than or equal to zero. The cost values of the remaining edges are reset to one and their flow values remain unchanged. The new graph is an optimal topologically sorted directed acyclic subgraph between the peripheral and all hubs. As mentioned earlier, we refer to this subgraph as optimal directed acyclic subgraph.

At this point, we can use any path computation algorithm[3] for K final iterations to find one set of routing solutions, $r_i \in \mathcal{R}$. After each iteration, the flow values of the edges selected in the path are

[2] Since some of the edges have negative cost values, the Bellman–Ford or similar algorithm must be used to compute the shortest path.

[3] The path computation algorithm in this step does not necessarily need to be the shortest path algorithm as any path computation algorithm provides a set of routing solutions which is optimal with respect to the objective function defined earlier.

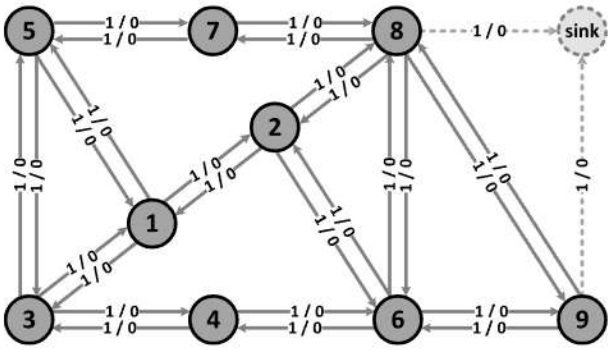

Fig. 3. A directed graph with $|\mathcal{V}| = 9$ vertices and $|\mathcal{E}| = 26$ edges.

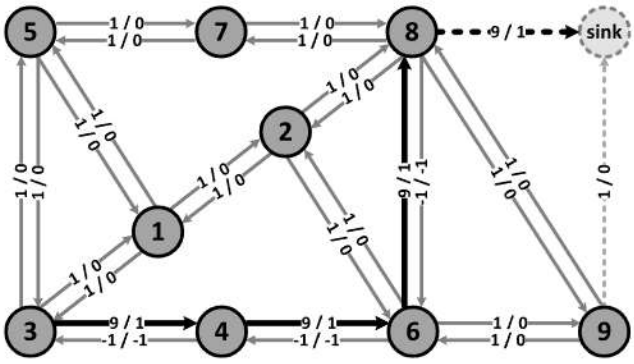

Fig. 4. The $(cost/flow)$ update after the first iteration. The shortest path calculated in this iteration is $\mathcal{E}_{P^{(1)}(v_3,v_{sink})} = [e(v_3,v_4), e(v_4,v_6), e(v_6,v_8), e'(v_8,v_{sink})]$.

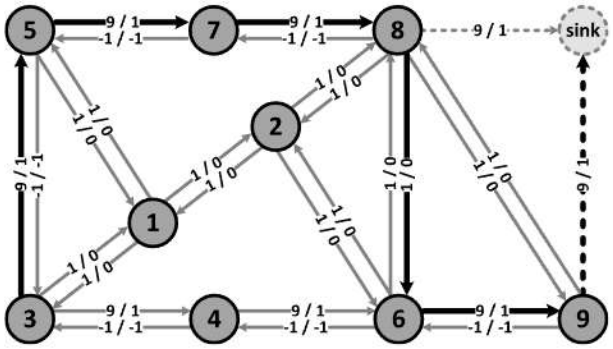

Fig. 5. The $(cost/flow)$ update after the second iteration. The shortest path calculated in this iteration is $\mathcal{E}_{P^{(2)}(v_3,v_{sink})} = [e(v_3,v_5), e(v_5,v_7), e(v_7,v_8), e(v_8,v_6), e(v_6,v_9), e'(v_9,v_{sink})]$.

decremented. If the flow value of an edge becomes zero that edge must be removed from the optimal directed acyclic subgraph.

The optimal directed acyclic subgraph can also be used to optimize a secondary objective function. For example, the secondary objective function can be maximizing the number of fiber-disjoint paths in a set of routing solutions. Let us define a new graph called the *path dependency* graph with K vertices, where each vertex represents one of the best reliable K shortest paths. If the paths are not fiber-disjoint their corresponding vertices in the path dependency graph become connected with an edge. In order to maximize the number of fiber-disjoint paths, the number of the edges in the path dependency graph should be minimized. To achieve this goal, we need to update the cost values of the edges in the optimal directed acyclic subgraph as shown in Eqs. (12) and (13).

$$c(v_i, v_j) = -V^{f(v_i,v_j)},\ \forall\ e(v_i, v_j)\ \exists\ f(v_i, v_j) > 1 \tag{12}$$

$$c(v_i, v_j) = -V^{(K+1)},\ \forall\ e'(h_d, v_{sink}) \in \mathcal{E}' \tag{13}$$

We then run the shortest path algorithm for K iterations on the optimal directed acyclic subgraph to find K paths from p_s to v_{sink}. Let us refer to the path computed in kth iteration as $\pi^{(k)}_{h_d}$, where $k \in [1, K]$ and h_d is the hub that is used to reach the v_{sink}. In each iteration, a new vertex representing the $\pi^{(k)}_{h_d}$ is added to the path dependency graph. The flow of the selected edges in the shortest path in the optimal directed acyclic subgraph is decremented and if it becomes zero the corresponding edge is removed, except the extra edges. Note that the negative cost for the edges used more than once instructs the algorithm to put as many negative cost edges as possible in one path, hence minimizing the number of the edges in the path dependency graph. In other words, once two or more paths share at least one edge, they cannot be considered as fiber-disjoint paths and any additional edge that is shared by the two paths will not worsen this dependency.

In the last step, when all K paths are computed, the vertices of the path dependency graph are classified based on h_d. In a graph with H hubs, there exists H groups of vertices where each contains a set of paths to a distinct hub. Since the cost of the extra edges in the optimal directed acyclic subgraph are set to the minimum possible value $(-V^{(K+1)})$, it is possible that the hubs are not equally utilized. For example, consider the case in which the hub h_i is connected to the hub h_j through the edge $e(h_i, h_j)$ in the optimal directed acyclic subgraph. In this case, it can shown that the shortest path algorithm never selects the edge $e(h_i, h_j)$ to be in the shortest path, thus, selecting one more path to h_i and one less path to h_j. When this happens, any one of the vertices in the h_i group can be moved to the h_j group, in which case the edge $e(h_i, h_j)$ is added to the path associated to the moved vertex. This procedure gives the algorithm the flexibility to increase or decrease the dependency between the groups of vertices in the path dependency graph based on the desired disjointedness criteria.

Next, we explain the algorithm using an example graph. Let the graph shown in Fig. 3 represent an optical network with 9 nodes and 13 bidirectional fiber links, where the peripheral is $p_s = v_3$, the hubs are $\mathcal{H} = [v_8, v_9]$, and the term $(1/0)$ written on the edges indicates the $(cost/flow)$ values of each edge. The sink vertex v_{sink} and the extra edges $\mathcal{E}' = [e'(v_8, v_{sink}), e'(v_9, v_{sink})]$ are shown with gray dashed circle and arrows, respectively. In this example we want to find the best reliable K shortest paths from v_3 to the hubs v_8 and v_9, where $K = 4$. Therefore, four iterations are needed to find two paths to each of the two hubs. Figs. 4–7 show the computed shortest path $P^{(k)}(p_s, v_{sink})$ for $k \in [1, 4]$, and the cost and the flow updates of the edges after each iteration. The computed shortest path in each iteration is highlighted with the black colored edges.

In the first iteration, the first shortest path is computed using the graph in Fig. 3. The shortest path computed in this iteration is through the $\mathcal{E}_{P^{(1)}(v_3,v_{sink})} = [e(v_3,v_4), e(v_4,v_6), e(v_6,v_8), e'(v_8,v_{sink})]$, with the cost of 3 to the hub v_8. The $(cost/flow)$ update is done after the path is computed and it is shown in Fig. 4.

In the second iteration, the second shortest path is computed using the graph in Fig. 4. The shortest path computed in this iteration is through the $\mathcal{E}_{P^{(2)}(v_3,v_{sink})} = [e(v_3,v_5), e(v_5,v_7), e(v_7,v_8), e(v_8,v_6), e(v_6,v_9), e'(v_9,v_{sink})]$, with the cost of 3 to the hub v_9. In this iteration, the shortest path algorithm prefers to use the edge $e(v_8, v_9)$ due to its negative cost (-1). As described earlier, the algorithm modifies the first calculated shortest path in this iteration. The $(cost/flow)$ update in this iteration is shown in Fig. 5. Note that the edges $e(v_6, v_8)$ and $e(v_8, v_6)$ are reset to initial condition.

In the third iteration, the third shortest path is computed using the graph in Fig. 5. The shortest path computed in this iteration is through the $\mathcal{E}_{P^{(3)}(v_3,v_{sink})} = [e(v_3,v_1), e(v_1,v_2), e(v_2,v_8), e'(v_8,v_{sink})]$, with the cost of 3 to the hub v_8. The $(cost/flow)$ update in this iteration is shown in Fig. 6. Note that after this iteration all edge-disjoint paths are found. Therefore, in the next iteration some of the previously used edges will be selected.

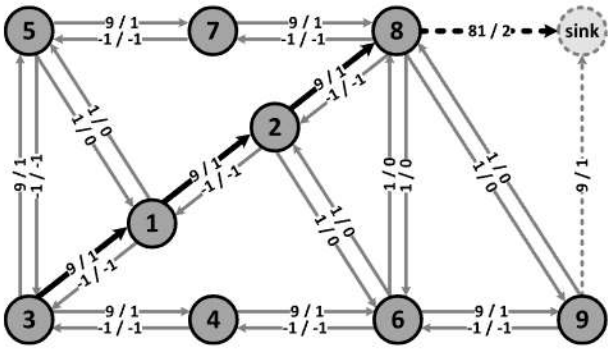

Fig. 6. The $(cost/flow)$ update after the third iteration. The shortest path calculated in this iteration is $\mathcal{E}_{P^{(3)}(v_3,v_{sink})} = [e(v_3,v_1), e(v_1,v_2), e(v_2,v_8), e'(v_8,v_{sink})]$.

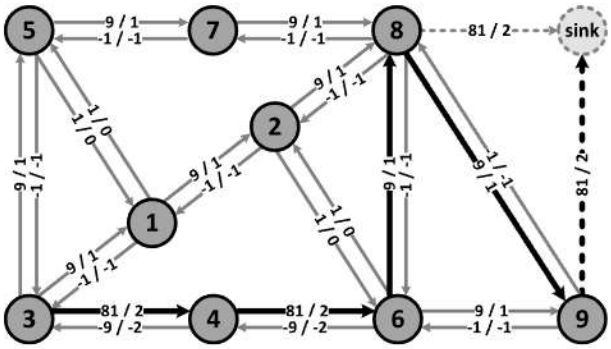

Fig. 7. The $(cost/flow)$ update after the fourth iteration. The shortest path calculated in this iteration is $\mathcal{E}_{P^{(4)}(v_3,v_{sink})} = [e(v_3,v_4), e(v_4,v_6), e(v_6,v_8), e(v_8,v_9), e'(v_9,v_{sink})]$.

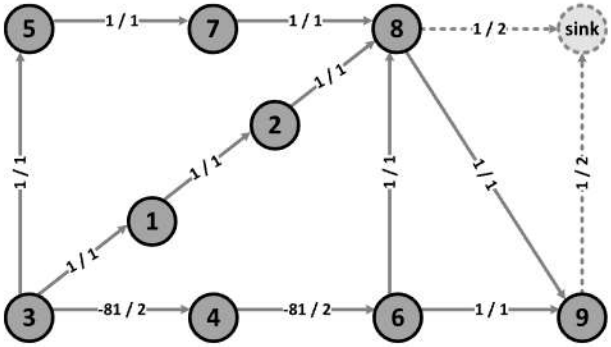

Fig. 8. Optimal directed acyclic subgraph.

In the fourth iteration, the fourth shortest path is computed using the graph in Fig. 6. The shortest path computed in this iteration is through the $\mathcal{E}_{P^{(4)}(v_3,v_{sink})} = [e(v_3,v_4), e(v_4,v_6), e(v_6,v_8), e(v_8,v_9), e'(v_9,v_{sink})]$, with the cost of 20 to the hub v_9. Since the algorithm computes the shortest path and the used edges have higher cost values (9), it then optimally selects the minimum number of the edges that are used previously, e.g., in this case the edges $e(v_3,v_4)$ and $e(v_4,v_6)$. The $(cost/flow)$ update in this iteration is shown in Fig. 7.

At this point, the optimal directed acyclic subgraph can be built by excluding all the edges with the flow values less than or equal to zero as shown in Fig. 8. Any path computation algorithm can be used for $K = 4$ iterations to find a set ($r_i \in \mathcal{R}$) of best reliable 4 shortest paths between the peripheral v_3 and the hubs $[v_8, v_9]$.

Let us also investigate the secondary objective function defined earlier in this section. The cost values for the edges $e(v_3,v_4)$ and $e(v_4,v_6)$ are updated to -9^2 using Eq. (12), and the cost values for the extra edges $e'(v_8,v_{sink})$ and $e'(v_9,v_{sink})$ are updated to -9^5 using Eq. (13). We

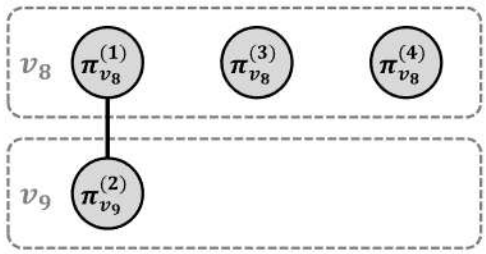

Fig. 9. Path dependency graph.

then run the shortest path for $K = 4$ iterations and the shortest paths are computed as follows.

- $\pi^{(1)}_{v_8} = [v_3, v_4, v_6, v_8]$,
- $\pi^{(2)}_{v_9} = [v_3, v_4, v_6, v_9]$,
- $\pi^{(3)}_{v_8} = [v_3, v_5, v_7, v_8]$,
- $\pi^{(4)}_{v_8} = [v_3, v_1, v_2, v_8]$,

where each path is shown with the sequence of the vertices from peripheral to the two hubs.

The path dependency graph is shown in Fig. 9. The vertices are classified according to v_8 and v_9. The v_8 and v_9 groups are shown with two dashed rectangles. It can be seen that the v_8 group has two more vertices so one of the vertices needs to be moved to the group v_9. In this example, moving $\pi^{(1)}_{v_8}$ makes the two groups fiber-disjoint but the paths in the v_9 group dependent. Two other options are to move $\pi^{(3)}_{v_8}$ or $\pi^{(4)}_{v_8}$ which both make the paths in both groups fiber-disjoint but the two groups are dependent. Note that after the vertex is moved to the new group, the remaining edges in the optimal directed acyclic subgraph should be added to the moved path, e.g., the edge $e(v_8,v_9)$ needs to be added to one of the vertices in v_8 group after it is moved to v_9 group. We can see that in this example, there are three different sets of best reliable $K = 4$ shortest paths, all having the same optimized vector $\mathcal{L}$. The reliability vector obtained for this example is $\mathcal{L} = [9, 2, 0, 0]$ and the $cost_{ideal} = 2$.

It is worth mentioning that since the algorithm is based on Bellman–Ford shortest path algorithm, the time complexity for computing K edge disjoint shortest paths that are most reliable is $O(K \cdot |\mathcal{V}| \cdot |\mathcal{E}|)$. When applying cost increments in the order of $|\mathcal{V}|$, a numerical problem may be encountered after many iterations (bounded by K). Solutions to mitigate this issue will be investigated in the future.

6. Choosing reliable hub locations

In this section we discuss how the best reliable K shortest path algorithm is applied to select of the most reachable locations for the hubs in the network. We use the reliability vector $\mathcal{L}$ obtained from the algorithm to assess the hubs' reachability. Let us refer to the reliability vector of peripheral s (p_s) as $\mathcal{L}_{p_s}$, and consider a network with E directed fiber links and V nodes. We must choose H of the V nodes to become the hubs. There are $N = \binom{V}{H}$ possibile solutions. We refer to the mth solution as $\mathcal{H}_m$, where $m \in [1, N]$. We then compute the reliability vectors for all peripherals for each possible set of hubs $\mathcal{H}_m$ using the best reliable K shortest path algorithm.

In order to evaluate the hubs reachability from p_s, we use two different cost metrics, the $cost_{ideal}(p_s)$ as shown in Eq. (1) and the $cost_{eff.}(p_s)$ shown in Eq. (14).

$$cost_{eff.}(p_s) = \sum_{i=1}^{K} l_i \times E^{(i-1)}, \tag{14}$$

where E is the number of fiber links in the network. Eq. (14) evaluates the reliability vector by assigning heavier weights on the elements on

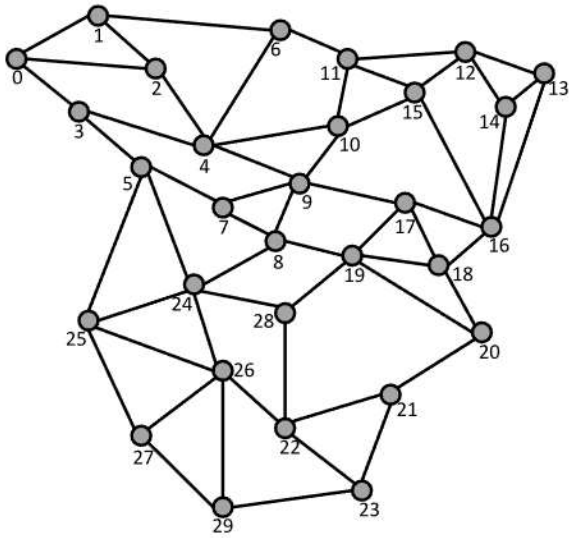

Fig. 10. Spanish network topology.

Table 3
Most reachable hubs for case of $H = 2$ and $K = 8$.

Cost metric	Most reachable hubs	Cost
$\overline{cost}_{ideal}$	$[v_{12}, v_{26}]$	5.7143
$cost_{ideal}^{(max.)}$	$[v_4, v_{22}]$, $[v_4, v_{24}]$, $[v_4, v_{25}]$, $[v_4, v_{26}]$, $[v_6, v_{22}]$, $[v_6, v_{24}]$, $[v_6, v_{25}]$, $[v_6, v_{26}]$	10
$\overline{cost}_{eff.}$	$[v_1, v_{27}]$	2904
$cost_{eff.}^{(max.)}$	$[v_1, v_{24}]$	6692

the right side of the vector. Therefore, a vector with smaller $cost_{eff.}(p_s)$ value indicates there are fewer fiber links that are shared by multiple paths. For example, if the reliability vectors for peripherals p_x and p_y are $\mathcal{L}_{p_x} = [14, 4, 0, 0]$ and $\mathcal{L}_{p_y} = [5, 2, 1, 0]$, then $cost_{ideal}(p_x) = cost_{ideal}(p_y) = 4$, $cost_{eff.}(p_x) = (14+4E)$, and $cost_{eff.}(p_y) = (5+2E+E^2)$.

For each set of hubs $\mathcal{H}_m$, we define the average and maximum $cost_{ideal}$ as shown in Eqs. (15) and (16), and the average and maximum $cost_{eff.}$ as shown in Eqs. (17) and (18), respectively.

$$\overline{cost}_{ideal}(\mathcal{H}_m) = \sum_{p_s \in \mathcal{P}} \frac{cost_{ideal}(p_s)}{|\mathcal{P}|}, \tag{15}$$

$$cost_{ideal}^{(max.)}(\mathcal{H}_m) = max(cost_{ideal}(p_s)), \tag{16}$$

$$\overline{cost}_{eff.}(\mathcal{H}_m) = \sum_{p_s \in \mathcal{P}} \frac{cost_{eff.}(p_s)}{|\mathcal{P}|}, \tag{17}$$

$$cost_{eff.}^{(max.)}(\mathcal{H}_m) = max(cost_{eff.}(p_s)), \tag{18}$$

where $m \in [1, N]$, $\mathcal{P} \in (\mathcal{V} - \mathcal{H}_m)$ is the set of peripheral nodes, and $\mathcal{V}$ is the set of all nodes in the network. Depending on the application we can use one of the cost metrics shown in Eqs. (15)–(18) to find the most reachable set of hubs.

Finally, for any given cost metric the $\mathcal{H}_m$ is most reachable if its cost is minimum for all $m \in [1, N]$. Next, we used this approach to find the set of most reachable hubs in a mesh network based on the mentioned cost metrics.

Fig. 10 shows the Spanish network topology which consists of $V = 30$ nodes and $E = 2\times56$ directed fiber links. We use the best reliable K shortest path algorithm to find the most reachable set of hubs in this network, based on the four cost metrics shown in Eqs. (15)–(18).

Tables 3 and 4 show the results of the most reachable hubs for the cases of $H = 2$, $K = 8$ and $H = 3$, $K = 9$, respectively. It can be seen that some entries of the tables have more than one set of hubs which

Table 4
Most reachable hubs for case of $H = 3$ and $K = 9$.

Cost metric	Most reachable hubs	Cost
$\overline{cost}_{ideal}$	$[v_0, v_{13}, v_{23}]$, $[v_0, v_{13}, v_{29}]$, $[v_0, v_{14}, v_{23}]$, $[v_0, v_{14}, v_{29}]$, $[v_1, v_{13}, v_{23}]$, $[v_1, v_{13}, v_{29}]$, $[v_1, v_{14}, v_{23}]$, $[v_1, v_{14}, v_{29}]$	6.5556
$cost_{ideal}^{(max.)}$	$[v_0, v_{12}, v_{22}]$, $[v_0, v_{12}, v_{23}]$, $[v_0, v_{13}, v_{22}]$, $[v_0, v_{13}, v_{23}]$, $[v_0, v_{14}, v_{22}]$, $[v_0, v_{14}, v_{23}]$, $[v_1, v_{12}, v_{22}]$, $[v_1, v_{12}, v_{23}]$, $[v_1, v_{13}, v_{22}]$, $[v_1, v_{13}, v_{23}]$, $[v_1, v_{14}, v_{22}]$, $[v_1, v_{14}, v_{23}]$, $[v_2, v_{12}, v_{22}]$, $[v_2, v_{12}, v_{23}]$, $[v_2, v_{13}, v_{22}]$, $[v_2, v_{13}, v_{23}]$, $[v_2, v_{14}, v_{22}]$, $[v_2, v_{14}, v_{23}]$	9
$\overline{cost}_{eff.}$	$[v_0, v_{13}, v_{23}]$, $[v_0, v_{14}, v_{23}]$, $[v_1, v_{13}, v_{23}]$, $[v_1, v_{14}, v_{23}]$	4988.7
$cost_{eff.}^{(max.)}$	$[v_1, v_{12}, v_{23}]$, $[v_2, v_{12}, v_{23}]$	9601

Table 5
Run-time of the best reliable K shortest path algorithm.

H	K	Total run-time (s)	Average run-time per $\mathcal{H}_m$ (ms)
2	8	3.3679	7.7
3	9	30.9371	7.6

means there are multiple solutions yielding the same reliability based on the chosen cost criteria.

Table 5 shows the run-time of the algorithm for the two examples shown in Tables 3 and 4. The third column shows the total run-time to compute all best reliable K shortest paths for all peripherals for each set of hubs and obtain their reliability vectors in seconds. The algorithm runs for the total time of $(V-H)\times N$, where $N = \binom{V}{H}$. The average run-time per set of hubs is shown in the fourth column, and it indicates the average run-time to compute all K paths from all peripherals to only one set of hubs.

7. Summary

This work investigates the architecture of a multi-hub optical edge-to-core transport network, focusing on improving its achievable reliability and traffic load balance by strategically locating destination nodes (the hubs) and optically routing communication paths between peripheral nodes and said hubs. A new figure of merit is proposed to estimate the achievable network reliability by considering the number of communication paths relying on the same common risk links (SRLG). By defining this figure of merit as the objective function it is possible to optimize connection reliability and traffic load balance while coping with the limited fiber cable diversity found in typical edge-to-core network topologies.

A key enabler of the proposed concept is the proposed algorithm that efficiently computes an optimal topologically sorted directed acyclic subgraph between peripheral and hub nodes. The subgraph serves as a building block to further compute multiple optimal routing solutions that meet the defined objective function, thus maximizing the achieved network reliability, subject to the existing fiber-link topological constraints. Furthermore, this algorithm can be implemented in a network planning tool to (1) choose the most reliable locations for the hubs connecting to the core and (2) compute a set of pre-computed primary and restoration paths that jointly meet the objective function. The pre-computed paths can be stored in the path computation element of the network controller to manage network fiber resources in real-time.

Looking ahead the proposed algorithm's capabilities can be enriched by considering additional design factors such as node or site disjointness, uneven fiber cable outage probabilities, and secondary objective functions, e.g., minimizing signal end-to-end propagation time. The ultimate goal is to design efficient routing algorithms that – building upon the optimal acyclic subgraph – can compute communication paths that satisfy additional objectives and constraints, therefore enhancing the applicability and broader performance of the proposed concept.

CRediT authorship contribution statement

Ali Shakeri: Conceptualization, Investigation, Methodology, Software, Writing – original draft. **Tianliang Zhang:** Conceptualization, Validation, Writing – review & editing. **Shunmugapriya Ramanathan:** Conceptualization, Writing – review & editing. **Miguel Razo:** Validation, Writing – review & editing. **Marco Tacca:** Validation, Writing – review & editing. **Andrea Fumagalli:** Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgment

This work is supported in part by U.S. National Science Fundation grant CNS-1956357 titled "Optics without Borders".

References

[1] P. Green, Progress in optical networking, IEEE Commun. Mag. 39 (1) (2001) 54–61.
[2] R. Ramaswami, Optical networking technologies: What worked and what didn't, IEEE Commun. Mag. 44 (9) (2006) 132–139.
[3] B. Mirkhanzadeh, S. Vachhani, B.G. Bathula, G. Thouenon, C. Betoule, A. Triki, M. Birk, O. Renais, T. Zhang, M. Razo, M. Tacca, A. Fumagalli, Demonstration of joint operation across OpenROADM metro network, OpenFlow packet domain, and OpenStack compute domain, in: 2020 Optical Fiber Communications Conference and Exhibition, OFC, 2020, pp. 1–3.
[4] W. Yao, B. Ramamurthy, Survivable traffic grooming with path protection at the connection level in WDM mesh networks, J. Lightwave Technol. 23 (10) (2005) 2846–2853.
[5] S. Ramanathan, M. Tacca, M. Razo, B. Mirkhanzadeh, K. Kondepu, F. Giannone, L. Valcarenghi, A. Fumagalli, A programmable optical network testbed in support of C-RAN: a reliability study, Photonic Netw. Commun. 37 (2018) 311–321.
[6] D. Zhou, S. Subramaniam, Survivability in optical networks, IEEE Netw. 14 (6) (2000) 16–23.
[7] S. Ramamurthy, B. Mukherjee, Survivable WDM mesh networks. Part I-protection, in: IEEE INFOCOM'99, Vol. 2, IEEE, 1999, pp. 744–751.
[8] G. Maier, A. Pattavina, S. De Patre, M. Martinelli, Optical network survivability: protection techniques in the WDM layer, Photonic Netw. Commun. 4 (3–4) (2002) 251–269.
[9] A.V. Tran, C.J. Chae, R.S. Tucker, Ethernet PON or WDM PON: A comparison of cost and reliability, in: TENCON 2005-2005 IEEE Region 10 Conference, IEEE, 2005, pp. 1–6.
[10] L. Wosinska, J. Chen, Reliability performance analysis vs. deployment cost of fiber access networks, in: 2008 7th International Conference on Optical Internet, IEEE, 2008, pp. 1–2.
[11] J. Chen, L. Wosinska, C. Mas Machuca, M. Jaeger, Cost vs. reliability performance study of fiber access network architectures, IEEE Commun. Mag. 48 (2) (2010) 56–65.
[12] Y. Kobayashi, R. Sako, Two disjoint shortest paths problem with non-negative edge length, Oper. Res. Lett. 47 (1) (2019) 66–69.
[13] K. Bérczi, Y. Kobayashi, The directed disjoint shortest paths problem, in: 25th Annual European Symposium on Algorithms, ESA 2017, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
[14] W. Lochet, A polynomial time algorithm for the k-disjoint shortest paths problem, in: Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA, SIAM, 2021, pp. 169–178.
[15] D. Lopez-Pajares, E. Rojas, J.A. Carral, I. Martinez-Yelmo, J. Alvarez-Horcajo, The disjoint multipath challenge: Multiple disjoint paths guaranteeing scalability, IEEE Access 9 (2021) 74422–74436, http://dx.doi.org/10.1109/ACCESS.2021.3080931.
[16] D. Xu, Y. Chen, Y. Xiong, C. Qiao, X. He, On finding disjoint paths in single and dual link cost networks, in: IEEE INFOCOM 2004, Vol. 1, 2004, p. 715, http://dx.doi.org/10.1109/INFCOM.2004.1354541.
[17] Q.V. Phung, D. Habibi, H.N. Nguyen, K. Lo, K pairs of disjoint paths algorithm for protection in WDM optical networks, in: 2005 Asia-Pacific Conference on Communications, 2005, pp. 183–187, http://dx.doi.org/10.1109/APCC.2005.1554044.
[18] 3GPP TS 38.300, NR and NG-RAN Overall description Stage-2, 2018.
[19] CUPS, https://ganganichamika.medium.com/separating-data-plane-and-control-plane-9fee0b7f3ef8.
[20] B. Balasubramanian, E.S. Daniels, M. Hiltunen, R. Jana, K. Joshi, R. Sivaraj, T.X. Tran, C. Wang, RIC: A RAN intelligent controller platform for AI-enabled cellular networks, IEEE Internet Comput. 25 (2) (2021) 7–17.
[21] 3GPP TS 38.879, Study on enhancement for resiliency of gNB-CU-CP, 2022.
[22] L.R. Ford Jr., Network Flow Theory, Tech. Rep., Rand Corp Santa Monica Ca, 1956.
[23] L.R. Ford, D.R. Fulkerson, A simple algorithm for finding maximal network flows and an application to the Hitchcock problem, Canad. J. Math. 9 (1957) 210–218.
[24] L.R. Ford, D.R. Fulkerson, Maximal flow through a network, in: Classic Papers in Combinatorics, Springer, 2009, pp. 243–248.
[25] J. Edmonds, R.M. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, J. ACM 19 (2) (1972) 248–264.

Ali Shakeri earned his Ph.D. degree at The University of Texas at Dallas (UTD). His research interests are centered around SDN-enabled WDM networks, optical flow switching, optical networks simulation, multi-layer network planning, optimization, and orchestration.

Tianliang earned his Ph.D. degree at The University of Texas at Dallas (UTD). He worked as a Research Associate at The University of Texas at Dallas. His research aims to advance optical communication systems and improve network efficiency, projects involving open disaggregated optical networking, network emulation, and cross-layer monitoring. He has also designed and implemented automated validation procedures for DWDM transmission equipment, complying with OpenROADM MSA standards.

Shunmugapriya Ramanathan is currently pursuing the Ph.D. degree at The University of Texas at Dallas (UTD). She worked as a Technical Lead in embedded Firmware development with Honeywell. She did her research internship at Intel. Her research interests include 5G, the Internet of Things (IoT), fault management, and software design in embedded platforms.

Miguel Razo earned his Ph.D. in Computer Science from The University of Texas at Dallas in 2009. His research spans various aspects of computer science, with a particular focus on network planning, fault protection, telecommunication software design, protocol design, network modeling, emulation, and simulation. Presently, he serves as a Research Associate at the OpNeAR (Open Networking Advance Research) Laboratory and holds the position of Senior Lecturer in the Computer Science Department at the Erik Jonsson School of Engineering and Computer Science.

Marco Tacca received his Laurea Degree from Politecnico di Torino in 1998 and his Ph.D. from the University of Texas at Dallas in 2002. His research interests encompass various aspects of optical networks, with a focus on high-speed photonic network planning, fault protection and restoration strategies, and performance evaluation.

Andrea Fumagalli received the Laurea and Ph.D. degrees in electrical engineering from the Politecnico di Torino, Torino, Italy, in 1987 and 1992, respectively. From 1992 to 1998, he was an Assistant Professor with the Electronics Engineering Department, Politecnico di Torino, Italy. He joined UT-Dallas, as an Associate Professor of electrical engineering, in August 1997, and was elevated to the rank of Professor, in 2005. He is currently a Professor of electrical and computer engineering at The University of Texas at Dallas. He has published about 250 technical articles in peer-reviewed refereed journals and conferences. His research interests include aspects of optical, wireless, the Internet of Things (IoT), cloud networks, and related protocol design and performance evaluation.