Benefits of Multi-Layer Application-Aware Resource Allocation and Optimization

Marco Savi*, Ciril Rozic†, Chris Matrakidis†, Dimitrios Klonidis†, Domenico Siracusa*, Ioannis Tomkos†
*Fondazione Bruno Kessler (FBK), CREATE-NET Research Center, Trento, Italy
†Athens Information Technology (AIT), 44 Kifisias Ave., 15125 Marousi, Greece

***Abstract*—Internet traffic is generated by a multitude of applications, each one with diverse service requirements in terms of bandwidth, latency, reliability, etc. Today traffic engineering techniques can provide service differentiation at the IP/MPLS layer, but not at the optical layer. In this paper we propose a framework where application service requirements drive a dynamic multi-layer (IP/MPLS and optical) resource allocation and optimization. We compare by means of simulations such application-aware algorithmic framework with a multi-layer but application-unaware strategy. Results show that the application-aware approach, unlike the application-unaware one, is always able to guarantee the specified service requirements to those applications whose generated traffic is accepted by the network. In addition, the application-aware strategy does not consume more network resources than the application-unaware one, but only requires a network that is more dynamic and responsive.**

I. Introduction

In modern telecommunication networks Internet traffic is generated by a huge number of diverse applications (e.g. online gaming, financial transactions, video streaming etc.), each one with its own requirements in terms of bandwidth, latency, reliability, security etc. In the last years, a lot of effort has been made to improve traffic engineering and provide service differentiation at the IP/Multi Protocol Label Switching (MPLS) packet layer. However, such effort is limited by the fact that IP/MPLS traffic is eventually groomed in the same few large optical connections, where no service differentiation is achieved.

A novel and more comprehensive approach to provide a full differentiation down to the optical network is thus needed. First, the network must be able to gather information from applications about their requirements, then it must be able to steer traffic according to such requirements (i.e., in an *application-aware* fashion) at both IP/MPLS and optical layers (i.e., in a *multi-layer* fashion). The former aspect can be enabled by the novel *intent-based* networking paradigm [1], according to which the applications provide to the network, through an intent-based interface, *what* they want to achieve in terms of requirements (i.e., the intent) without specifying to the network *how* to achieve it. The network is then responsible to autonomously attain the latter aspect, i.e., to allocate both IP/MPLS and optical resources in an application-aware way.

The most straightforward solution to pursue these objectives is to deploy a centralized entity that *i)* has the global multi-layer vision of the network and *ii)* is able to control, provision and optimize resources on both IP/MPLS and optical layers. In the H2020 ACINO project [2] we exploit the Software-Defined Networking (SDN) paradigm to build an *SDN network orchestrator* as the connecting component and intelligence between the applications and the underlying network infrastructure [3][4]. Every time a service request (i.e., intent) is delivered to the orchestrator through a northbound intent-based interface, the orchestrator is in charge of computing a path in the network satisfying the intent (i.e., an *application-aware path*). To do so, it requests the resources (IP ports and spectrum slices) to an SDN control platform such as OpenDaylight [5] or ONOS [6], which in turn establish connections through the IP/MPLS and optical controllers. In case the orchestrator cannot satisfy the intent due to insufficient network resources, it is also able to exploit the intent-based interface to negotiate with the application until the intent can be satisfied.

In such a dynamic situation, it is key that the network orchestrator features a good resource allocation and network optimization component that is able *i)* to provide application-aware paths to the applications jointly at IP/MPLS and optical layers and *ii)* to periodically re-optimize the network connections in order to limit the fragmentation of IP/MPLS and optical resources. In this paper we present a modular algorithmic framework for dynamic appliction-aware multi-layer resource allocation and optimization that we call *ACINO framework*. Our scheme will be implemented as part of the ACINO SDN orchestrator. We then compare it with a benchmark dynamic multi-layer strategy that lacks application-awareness (*AppUnaware strategy*) to show how the ACINO approach can significantly improve network performance from a service differentiation perspective.

The remainder of this paper is structured as follows. In Section II we recall some related work. In Section III we present the ACINO framework and highlight the differences with the AppUnaware strategy we use as benchmark. In Section IV we describe the application-aware policies and classes that we have identified as the most interesting for the assessment of the ACINO framework. In Section V we report and comment our simulation results, while Section VI concludes the paper.

II. Related works

In literature, there is a number of works dealing with multi-layer resource orchestration architectures. Ref. [7] describes ABNO, an IETF-defined orchestration framework that enables multi-layer and multi-technology network automation

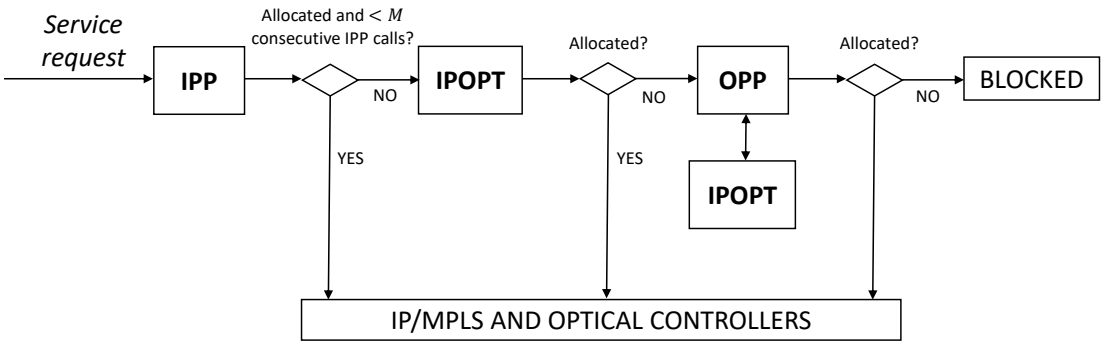

Fig. 1. ACINO resource allocation and optimization framework

and programmability by using IETF standard protocols and components. Ref. [8] then assesses two resource orchestration architectures based on ABNO concepts. Even though such works take a significant step forward automatic multi-layer orchestration and provisioning, they do not focus on the application-awareness concept, i.e., they do not guarantee that multi-layer orchestration is performed in such a way that multiple and diverse application-specific requirements are met in allocating network resources. This paper fills such gap.

Some works also deal with the design of multi-layer dynamic resource allocation algorithms. Ref. [9] proposes an algorithm for dynamic traffic grooming in multi-layer networks, while Refs. [10] and [11] define two dynamic algorithms for multi-layer virtual network mapping onto an SDN network substrate. Even though the last two works are mainly focused on the specific problem of virtual network mapping, they can be adapted to the subproblem of resources allocation on end-to-end connections (i.e., link mapping). In this work we exploit the methodologies adopted by these papers, and we push them on an application-awareness direction.

Finally, many works can be found on multi-layer network optimization. For example, Ref. [12] presents a multi-layer planning algorithm that is able to consistently reduce costs with respect to sequential (i.e., separate) planning in the IP/MPLS and optical layers. However, none of such works aims at dynamically and on-the-fly re-optimizing the network to save cost and network resources. The framework we present in this paper will also fill this gap.

III. ACINO Resource Allocation and Optimization

In this section we present our ACINO framework. Note that a first preliminary description of the framework can be found in [13]. The main idea behind it is to exploit the differences between IP and optical layers in terms of capacity provisioning and connection setup speed. In fact, every time a new end-to-end optical connection needs to be established, a new lightpath has to be set up between a couple of transceivers, which in turn need to be powered on. This operation can take tens of seconds, meaning that the optical network has a quite low time-responsiveness. On the opposite, setup times for new IP/MPLS connections are usually in the order of hundreds of milliseconds, making the IP/MPLS layer very responsive. Note also that a new IP/MPLS layer connection will use only the previously-setup optical connections as its constituent IP links.

Because the IP/MPLS layer is much more time-responsive than the optical layer, whenever possible, a service request will be accommodated in the IP/MPLS layer, since no new optical connection must be established in this case. Otherwise, if the demanded service requirements (e.g., bandwidth, latency, availability etc.) cannot be satisfied, the framework will resort to the optical layer, trying to meet the demanded service requirements by establishing new optical connections. This logic guides our resource allocation procedure, which is composed of three modules: *IP Provisioning* (IPP), *IP Optimization* (IPOPT), and *Optical Provisioning* (OPP). IPP tries to satisfy the service request by using available resources at the IP/MPLS layer only. IPOPT dynamically changes the existing routes in the IP/MPLS layer to achieve an optimization goal. OPP adds new lightpaths, which are then used as IP links.

Figure 1 shows the overall ACINO framework. As a general model, we assume that a service request r is represented by a $r(s, d, b, l, a, p)$ tuple, where s and d are the IDs of the source and destination IP/MPLS routers for the service request, b is the requested bandwidth, l is the maximum tolerable end-to-end latency, a is the minimum tolerable end-to-end path availability and p specifies whether the application traffic associated to the service request must be protected or not. The network must then provide connectivity to the application requesting the service by computing one (or, in case of protection, two) application-aware path(s), i.e., an end-to-end path that satisfies all of the requirements b, l, a, p.

After a service request r is received by the orchestrator, it tries to set up an application-aware path by using IPP. The path is then forwarded to the IP/MPLS controller, unless there have been M successful accommodations using IPP. In that case, or if IPP fails, IPOPT is called to rearrange the IP layer connections to free up unneeded transceivers. This IP layer re-optimization can reduce power consumption (if freed-up transceivers are eventually turned off, as assumed in this paper), or optimize the network for another objective. By calling IPOPT after M successful IPP calls, we assure that the network is re-optimized at least every the provisioning of M service request and resource fragmentation is avoided. Otherwise, if it is not possible to accommodate the service request even after having re-arranged the IP connections though

IPOPT, OPP is called to set-up new optical connections. Finally, IPOPT is called again to re-arrange IP connections considering the new IP topology. Next, we describe some algorithmic details of each module.

A. IP Provisioning Module (IPP)

IPP first constructs an Auxiliary Graph (AG) (see [14]) by creating a copy of the IP layer topology, which consists of the sufficiently capacitated IP links for the specific service request that is considered (i.e., the links meeting b). Next, the AG link weights are set equal to the physical length of the IP links. Then, IPP produces candidate paths using K_{ip}-Shortest Path (SP) algorithm on the AG between s and d, and pruning those paths not meeting l and a. The first path (or first IP disjoint path couple, if protection p is considered) that meets each service requirement is selected, and the required bandwidth allocated on its links. For more details on IPP algorithmic aspects, the reader is referred to [15].

B. IP Optimization Module (IPOPT)

IPOPT tries to improve the routing over the network by minimizing a cost function. It tries alternative routings for each established IP/MPLS connection and accepts changes only when the cost function is improved. Note that this is a local optimization approach, in the sense that the solution found converges to a local optimum. However, since the connections over the network continuously and dynamically change, a later call of IPOPT will break out of this local optimum. The cost function tries to satisfy three main objectives: *i)* that no service requests are unsatisfied, *ii)* that the number of IP links is minimum, and *iii)* that the utilization of the ip links is balanced. An important consideration in the algorithm selection for IPOPT is that the changes suggested in rearranging IP/MPLS connections should be implemented on the network in a hitless way. To facilitate this, a list of changes is maintained in the order that they can be implemented, and only the changes that are possible without affecting other connections are considered. When the final, improved, state is determined, these changes are implementable on the network in the specified order without any service interruption.

C. Optical Provisioning Module (OPP)

OPP is similar to IPP. It constructs an AG from the IP topology but augments it by also considering *potential* IP links. For each IP/MPLS node pair, a potential IP link is added if and only if there is no IP link for the pair in the AG. Each potential IP link is associated to a potential lightpath. To build such a lightpath, the K_{wdm}-SP First-Fit (FF) routing and spectrum assignment (we assume no wavelength conversion) is run in the optical layer. Then, when the K_{wdm}-SP FF algorithm has found free spectrum for the associated potential lightpath, the spectrum is reserved but not allocated yet. Next, the K_{ip}-SP algorithm is run over the AG and the candidate paths not meeting the service requirements are pruned, as happens in IPP. Next, IPOPT is run again to find the optimal IP/MPLS routes for all the service requests in the network. Finally, the potential lightpaths that have been added by OPP and carry traffic are allocated the spectrum, and the bandwidth for the new service request is allocated in the IP/MPLS layer. For more details on OPP the reader is referred to [15].

D. AppUnaware strategy description

After having described in detail the ACINO framework, in this section we shortly describe the *AppUnaware strategy* we consider as benchmark. AppUnaware consists of the same modules of the ACINO framework (i.e., IPP, IPOPT and OPP), and its flow diagram is the same as the one depicted in Fig. 1. The main difference between *ACINO* and *AppUnaware* is in the implementation of IPP, IPOPT and OPP modules. In fact, AppUnaware does not consider any latency l or availability a requirement in the provisioning and optimization of a service request r, but only guarantees the bandwidth b. This means that, if a path meeting the b requirement is found in the IP/MPLS layer by running IPP, the resources are allocated and the service request is provisioned. Otherwise, IPOPT is called to re-arrange IP/MPLS connections without caring of l and a. Finally, if IPOPT fails to find a path for the service request, OPP is called to provision new optical resources and IPOPT is called again to find the optimal IP/MPLS routes for the service requests.

Note that such an AppUnaware strategy is inherently multi-layer and able to re-optimize the network, as ACINO does. The only feature it misses with respect to ACINO is the capability of provisioning an application-aware path (or two, in case of protection p) for each service request. For this reason, even though it can alone be considered as an improvement over the current practices, we believe that it is a good benchmark to disclose the ACINO benefits from an application-awareness perspective.

IV. Application-Aware Policies and Classes

As shown in the previous section, in order to implement the ACINO application-aware features, specific optimization algorithms and decision polices have been implemented in all three modules, IPP, IPOPT and OPP. In this section we describe more in detail such different *application-aware policies* (related to the service requirements l, a, p) and how, according to such policies, multiple *application-aware classes* can be defined. If no service requirement (other than the bandwidth b) is defined for the incoming requests, then these are treated as *best effort* traffic, and the ACINO strategy operates as the AppUnaware one. The following three application-aware policies have been defined:

- *Latency awareness (l)*: Latency relates with the propagation delay and processing delay in IP/MPLS nodes. Optical nodes introduce no latency. The framework can support multiple latency values, thus allowing, for example, classes of services with strict latency requirements, other with average requirements and other with no specific requirements.
- *Availability (a)*: Actual link and node failure statistics are used in order to determine the probability of a failure in

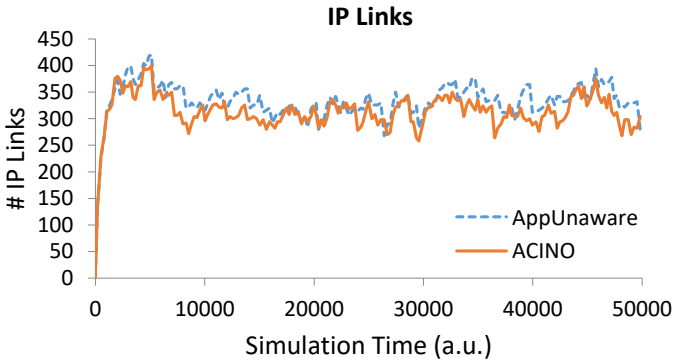

Fig. 2. Number of IP Links over time for both ACINO and AppUnaware (traffic: 25% Gold class, 25% Silver class, 50% best effort)

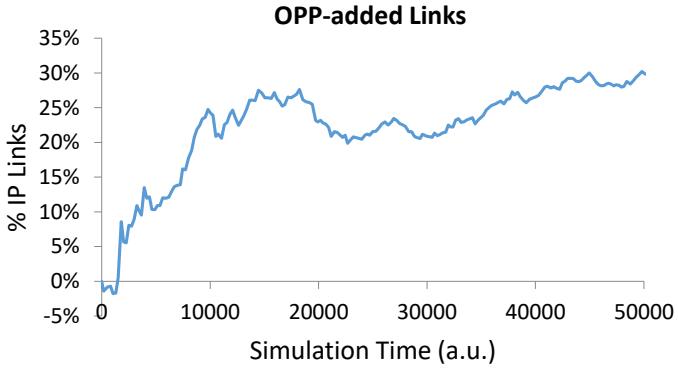

Fig. 3. Percent increment in the number of OPP-added IP links by ACINO compared to AppUnaware (traffic: 25% Gold class, 25% Silver class, 50% best effort)

the network. A service with high availability requirements is allocated over paths with reduced overall probability of failure both at IP/MPLS and optical layer.

- *IP/MPLS Protection (p)*: For applications with IP/MPLS protection service requirement, a disjoint IP/MPLS path couple must be found. Such paths must also simultaneously follow separate optical links in any part of the end-to-end route.

Such policies are only some of the possible ones that can be considered by ACINO. In fact, the framework can be easily extended to take into consideration other policies, e.g. related to optical encryption or security. According to the three application-aware policies described above, multiple application-aware classes can be defined by combining them. In this paper, we consider the following three classes:

- *Latency-sensitive, high-availability and protected class* (Gold class): This class asks for the most stringent requirements, and is related to very sensitive applications (e.g. financial transactions).
- *Latency-sensitive class* (Silver class): This class is related to all those applications that require a latency-sensitive service (e.g. database synchronization between datacenters).
- *Best effort class*: No specific requirement is needed, just bandwidth b.

Note that the Gold and Silver classes are application-aware classes, while best effort is not. In the following we will evaluate the ACINO and the AppUnaware strategies by assuming that the network traffic always belongs to one of these classes.

V. Simulation Results

To evaluate the ACINO and AppUnaware strategies we implemented them using the Online Simulation Tool of Net2Plan [16]. For our tests, we used the Telefonica Spain national network topology, with 30 all-optical nodes (Optical Cross-Connects, OXCs), 56 bi-directional fiber links and 14 IP/MPLS routers. We considered a realistic (non-uniform) traffic matrix provided by the national provider, where most of the traffic is routed from/to IP/MPLS routers around the capital city. As latency contributions, we consider the fiber links propagation delay and we assume that each IP/MPLS router adds a queuing delay of 0.5 ms. Moreover, nodes/links availabilities are randomized and uniformly distributed among the following three values: 99.9%, 99.99% and 99.999%. For each service request, the requested bandwidth b is uniformly chosen in the interval [10 Gbit/s, 100 Gbit/s]. A service request can belong to one of the three classes described in Section IV. For the Gold class, a latency l of 6 ms and a path availability a of 99.95% must be guaranteed (other than the traffic must be protected, p). For the Silver class a latency requirement l of 6 ms must be met only. Service requests are generated according to a Poisson process, with an exponential holding time: once a service request expires, the resources allocated to it are released.

In our evaluation, we focus on four different performance metrics:

- *IP links* in the network: monitors the resource occupation in terms of IP links (i.e., lightpaths and transponders).
- *OPP-added IP links*: counts the cumulative number of IP links that have been added by the OPP module.
- *Blocking percentage*: evaluates the percentage of blocked service requests because it is not possible to find one (or two, in case of protected traffic) application-aware path(s) (ACINO) or one (two) simple multi-layer path(s) (AppUnaware) for it.
- *Violation percentage*: evaluates the percentage of service requests provisioned by a path that do not meet one or more service requirements.

Figure 2 shows the number of IP links in the network when 50% of the traffic is application-aware (25% Gold and 25% Silver) for both ACINO and AppUnaware. The number of IP links stabilizes after around 10000 time units. Then the number hovers around 350, i.e. +/- cca 30 IP links, for both ACINO and AppUnaware strategies, which activate more or less the same number of IP links. This means that ACINO is as good as AppUnaware from a resource occupation perspective, i.e., on average no more transponders must be turned on to support ACINO resource allocation and optimization with respect to the benchmark scenario.

Figure 3 shows instead the percent increment in the number of OPP-added IP links by ACINO compared to AppUnaware. In the long term, ACINO requires up to 30% more OPP-added IP links than AppUnaware. This means that to maintain

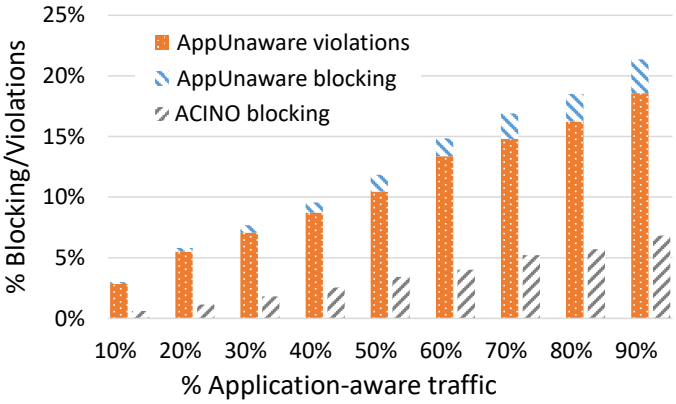

Fig. 4. ACINO and AppUnaware blocking and violation percentages as a function of the percentage of application-aware traffic (equally split between Gold and Silver classes)

the circa 350 IP links active (the same as the AppUnaware, as shown in Fig. 2) ACINO has to add and remove them more frequently. This is something expected, since finding an application-aware path through IPP or IPOPT is harder than just finding an AppUnaware multi-layer path, and thus the ACINO strategy has to rely on OPP more often. As a result, the more dynamic ACINO network needs an orchestrator to handle the frequent turn on/turn off operations without disrupting the running services. In return, ACINO can offer a more sophisticated service to the customer, as shown in Fig. 4.

Figure 4 depicts the blocking and violation percentages for ACINO and AppUnaware as a function of the percentage of application-aware traffic (equally split between Gold and Silver classes). As expected, no violation is experienced by ACINO, since the algorithmic framework is designed to never violate. On the opposite, AppUnaware leads to high violations, especially when the traffic is mostly application-aware (18% of violations for 90% of application-aware traffic). On the other hand, the blocking percentage is about twice as large for ACINO than for AppUnaware. This happens because ACINO blocks the service requests if it cannot guarantee to the applications all the service requirements, while AppUnaware blocks only in case of a lack of resources (i.e., bandwidth). The blocking percentage increases when the application-aware traffic increases, mainly because more protected traffic (belonging to Gold) must be allocated, leading to higher resource occupation and higher blocking for lack of resources. However, it is clear how the number of mistreated service requests (blocking+violation percentage) is always much higher for AppUnaware than for ACINO, confirming the benefits of the ACINO approach from an application-awareness perspective.

VI. Conclusion

In this paper we proposed a multi-layer network resource allocation and optimization framework, based on the properties of the IP and optical layers, which is application-aware, i.e., able to always meet multiple service requirements as requested by the applications. Then, we benchmarked such framework against a strategy that is multi-layer but application-unaware. Simulation on a test network with realistic application traffic shows that the application-aware strategy mistreats much less service requests than the application-unaware one, since it is always able to allocate resources according to application needs. In addition, the application-aware and application-unaware strategies have a similar behaviour in terms of resource occupation, meaning that the application-aware strategy does not require the provisioning of any more IP/MPLS and optical resources than the application-unaware one. However, the application-aware strategy increases the dynamicity of the network, since equipment turn on/turn off operations are more frequent. This makes it necessary the adoption of a network resource orchestrator to speed up such operations.

Acknowledgment

The research leading to these results has received funding from the European H2020 Framework Programme under grant agreement no. 645127 ACINO project.

References

[1] M. Pham and D. B. Hoang, "SDN applications - the intent-based northbound interface realisation for extended applications," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, Jun. 2016.

[2] ACINO website. [Online]. Available: http://www.acino.eu/

[3] F. Pederzolli, D. Siracusa *et al.*, "SDN application-centric orchestration for multi-layer transport networks," in *2016 18th International Conference on Transparent Optical Networks (ICTON)*, Jul. 2016.

[4] V. Lopez, D. Klonidis, D. Siracusa *et al.*, "On the benefits of multi-layer optimization and application awareness," *Journal of Lightwave Technology*, vol. PP, no. 99, pp. 1–1, Feb. 2017.

[5] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a model-driven SDN controller architecture," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, Jun. 2014.

[6] P. Berde, M. Gerola, J. Hart *et al.*, "ONOS: Towards an open, distributed SDN OS," in *Proceedings of the ACM Third Workshop on Hot Topics in Software Defined Networking (HotSDN)*, 2014.

[7] A. Aguado, V. Lopez, J. Marhuenda *et al.*, "ABNO: a feasible sdn approach for multivendor ip and optical networks [invited]," *IEEE/OSA Journal of Optical Communications and Networking*, Feb. 2015.

[8] R. Munoz, R. Vilalta, R. Casellas *et al.*, "Transport network orchestration for end-to-end multilayer provisioning across heterogeneous SDN/openflow and GMPLS/PCE control domains," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1540–1548, Apr. 2015.

[9] S. Zhang, C. Martel, and B. Mukherjee, "Dynamic traffic grooming in elastic optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 1, pp. 4–12, Jan. 2013.

[10] J. Zhang, Y. Ji, M. Song *et al.*, "Dynamic virtual network embedding over multilayer optical networks," *OSA J. Opt. Commun. Netw.*, vol. 7, no. 9, pp. 918–927, Sept. 2015.

[11] H. T. Nguyen, A. V. Vu *et al.*, "A generalized resource allocation framework in support of multi-layer virtual network embedding based on SDN," *Computer Networks*, vol. 92, Part 2, pp. 251 – 269, 2015.

[12] V. Gkamas, K. Christodoulopoulos, and E. Varvarigos, "A comparison study of joint and sequential multi-layer planning for IP over flexible optical networks," in *2015 Optical Fiber Communications Conference and Exhibition (OFC)*, Mar. 2015.

[13] C. Rozic, M. Savi, C. Matrakidis *et al.*, "A framework for dynamic multi-layer resource allocation and optimization in application-centric networking," in *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, Mar. 2017.

[14] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee, "Dynamic traffic grooming in WDM mesh networks using a novel graph model," in *IEEE GLOBECOM 2002*, Nov. 2002.

[15] M. Savi, F. Pederzolli, and D. Siracusa, "An application-aware multi-layer service provisioning algorithm based on auxiliary graphs," in *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, Mar. 2017.

[16] P. Pavon-Marino and J. L. Izquierdo-Zaragoza, "Net2plan: an open source network planning tool for bridging the gap between academia and industry," *IEEE Network*, vol. 29, no. 5, pp. 90–96, Sept. 2015.