A Dynamic Multi-Layer Resource Allocation and Optimization Framework in Application-Centric Networks

Ćiril Rožić, Marco Savi, Chris Matrakidis, Dimitrios Klonidis, Domenico Siracusa

***Abstract* - In an SDN-based network, service requests can be accommodated according to network application requirements. We devise a framework where such requirements drive IP and optical network resource allocation, dynamic optimization, and instantiation through an SDN orchestrator. The framework aims to balance the often-conflicting goals such as energy savings and resource allocation speed, while ensuring that the orchestrator can accommodate incoming service requests without interrupting the running services. We show the functionality of the framework with a simulation on a real network topology and realistic input traffic.**

***Index Terms* - (060.4256) Networks, network optimization, (060.4251) Networks, assignment and routing algorithms**

I. Introduction

Applications running over the Internet today, such as autonomous driving, remote surgery, 5G backhauling, and data migration, demand more than just a bandwidth pipe over a connected network. The new *service requirements* are specific and complex, so merely providing a sufficiently capacitated optical channel will not suffice. Moreover, bundling the traffic from diverse applications does have advantages of cost, but the resulting service may be inadequate, the usage of network resources may be inefficient, or the network operator may miss an opportunity to generate revenue from higher-class services.

Recently, Software-Defined Networking (SDN) enables service providers to use the existing IP and optical network resources more intelligently and efficiently. This advance allows for assigning bandwidth where and when needed, and coping with the increase in the number of traffic flows. In project ACINO (Application-Centric IP/Optical Network Orchestration) [1], SDN principles are used to address the challenge of diversifying network applications.

Fig. 1 shows the architecture of a multi-layer SDN network orchestrator as envisioned by ACINO, where it is a connecting component and intelligence between the applications and the underlying network infrastructure [2]. The orchestrator exposes a set of network primitives, which are used by the applications to express their intents [3] in terms of service requirements to be met by the network. The orchestrator then computes a path through Net2Plan [4][5] and requests the resources (IP ports and spectrum slices) from the SDN ONOS [6] control platform establishing connections through IP and optical controllers [7]. The interaction between Net2Plan and ONOS is enabled by NetRap [8], which synchronizes changes in the topology, demands and routes between the two components. This way, it lets ONOS request path calculations from Net2Plan and retrieve resulting paths.

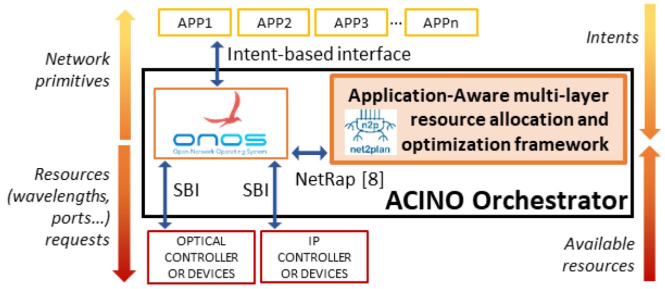


Fig. 1. ACINO orchestrator architecture and communication with network applications and infrastructure. This work focuses on the highlighted box.

In such a dynamic situation, and to enable *application-centric networking*, it is key that the orchestrator features a good path computation and network optimization component, with the primary aim of meeting service requirements, while optimizing resources usage. In this paper we propose a framework for such *application-aware* path computation and *resource allocation*, where the service requirements are always guaranteed. We also design an algorithmic strategy to jointly and dynamically *re-optimize* the network in both the IP and (elastic) optical layer, with the objective of reducing operational costs. A multi-layer approach is chosen because the orchestrator platform is assumed to have access to both layers and a complete view of the network. Such joint optimization, as opposed to layer-by-layer schemes, has been shown to be superior [9]. A dynamic setting, where connections are established and removed frequently, further

Manuscript received July 12th, 2018. The research leading to these results has received funding from the European Commission within the H2020 Research and Innovation programme, ACINO project, Grant Number 645127, www.acino.eu.

Chris Matrakidis and Dimitrios Klonidis are with Athens Information Technology (AIT), Maroussi 152 32, Greece. Ćiril Rožić is with Infinera; previously with AIT (e-mail: croz@ait.gr).

Marco Savi and Domenico Siracusa are with Fondazione Bruno Kessler (FBK), CREATE-NET Research Center, Trento, Italy.

A preliminary version of this paper appeared in [40], presented at the Optical Fiber Communications Conference and Exhibition (OFC), Los Angeles, CA in 2017, W3I.4.

warrants a fast SDN orchestrator. The framework discussed here and its simulation code developed in Ne2Plan also allow for demonstrations of the ACINO approach on real testbeds, where ONOS and Net2Plan successfully interact [8].

In the rest of the paper, Section II recalls the related work, Section III explains the ACINO framework modules, Section IV reports the results of our simulations, and then in Section V we conclude.

II. Related work

Our work builds upon the concept of *(i) application-centric network orchestration* and deals with the design of a *(ii) dynamic resource allocation* and *(iii) optimization* framework for multi-layer networks. To the best of our knowledge, the framework proposed in this paper is the first attempt to cover such three aspects at the same time. In addition, our work brings some contribution to each one of them separately.

A. Application-Centric Networking and Orchestration

The need of differentiating the network traffic depending on application requirements has been referred in literature to as *application-centric* (or *application-aware*, or, more broadly, *QoS-aware*) *networking* [10][11]. The SDN paradigm has been identified as a key enabler for application-centric networking and for a more comprehensive service differentiation depending on application requirements [12]-[15]. Some works [12][13] focus on architectural aspects, while some other works [14][15] focus instead on routing. How an SDN controller can collect statistics to improve the network application-awareness is instead shown in papers [16][17]. However, all the work misses the possibility, for an application, to have tailored resource provisioning not only at the upper layer (i.e., IP layer) but also across layers.

Some works have then faced the problem of designing a *multi-layer orchestrator* [18][19] for application-centric networking [7][20]. Papers [18][19] propose two architectural solutions, i.e., PCE and ABNO, for path computation, resource provisioning and orchestration in multi-layer networks. Even though these works move a significant step forward concerning multi-layer orchestration, they do not focus on application awareness. Conversely, [7] argues that by jointly orchestrating IP and optical resources it is possible to achieve a full-service differentiation for the applications. The work carried out in [20] then follows the general principles of [7] and defines a preliminary architecture for such an orchestrator. Our work extends the idea of [7][20]: we design the resource allocation and optimization algorithms to be executed by the orchestrator for application-aware connectivity.

B. Multi-Layer Dynamic Resource Allocation

The problem of dynamically allocating resources in multi-layer networks for the provisioning of service requests has been already investigated in literature [21]-[23]. The authors of [21] propose some dynamic algorithms for the efficient provisioning of low-speed traffic streams over high capacity lightpaths in IP-over-WDM networks. Paper [22] deals with a novel algorithm supporting differentiated resiliency schemes to guarantee end-to-end QoS for connections with different priorities. Paper [23] proposes a novel path computation algorithm favoring the accommodation of a service request at the IP layer over the establishment of new optical resources. Our algorithms, adopting the same principles, enable a tighter collaboration between the involved layers for a smarter usage of optical resources. The authors of [24] propose instead novel routing and grooming algorithms in flexible multi-layer networks, where the bandwidth of newly-established lightpaths is dynamically chosen depending on the traffic demands. Our algorithms can also dynamically choose the bandwidth of newly-established lightpaths in a flexible way, but the choice is driven by the applications.

Finally, papers [25][26] define two dynamic algorithms for virtual network mapping onto a multi-layer network substrate. Even though such works are mainly focused on the specific problem of virtual network mapping, they can be adapted to the sub-problem of resource allocation on end-to-end connections (i.e., link mapping). In our work, we extend these methodologies to provide application-aware end-to-end provisioning (e.g. we adopt similar *auxiliary graphs*, first introduced in [27]).

C. Multi-Layer Re-configuration and Optimization

Many works can be found in literature dealing with multi-layer network re-configuration and optimization to achieve different objectives [28]. Specifically, paper [29] proposes an algorithm for the periodic re-configuration of the optical layer, with the objective of increasing the throughput and reducing the delay. The authors of [30][31] deal instead with a re-configuration of the IP topology by adding/deleting lightpaths and/or re-routing existing traffic with the joint objective of *(i)* minimizing the number of used lightpaths and *(ii)* guaranteeing load balancing. The re-configuration is guaranteed to be hitless (i.e., to avoid service disruption). Our approach also guarantees hitless re-configuration under the same objectives while also respecting application constraints.

Multi-layer re-configuration has also focused on *energy-efficiency*, both from a static [32][33] and dynamic [34][35] perspective. Papers [32][33] introduce two models for network optimization with two different objectives: the minimization of the cost of interfaces [32] and the maximization of the usage of renewable energy [33]. Authors of [34][35] propose instead two energy-efficient re-configuration schemes: paper [34] investigates the trade-off between energy efficiency and routing stability, while paper [35] aims at switching to a power-saving topology when traffic load is low, by turning off unused links. Our work further allows balancing energy savings against lightpath establishment responsiveness.

Finally, authors of [36] prove how a multi-layer joint planning/optimization of IP and optical networks outperforms a sequential strategy, where IP and optical networks are independently optimized. As further step, work [37] introduces the concept of *in-operation network planning*, where the network can be automatically re-configured in response to traffic changes. Our work goes further by adding application-aware in-operation network planning.

III. Resource Allocation and Optimization Framework and Modules

In project ACINO, the framework must be able to *(i)* accommodate network application service requests as quickly

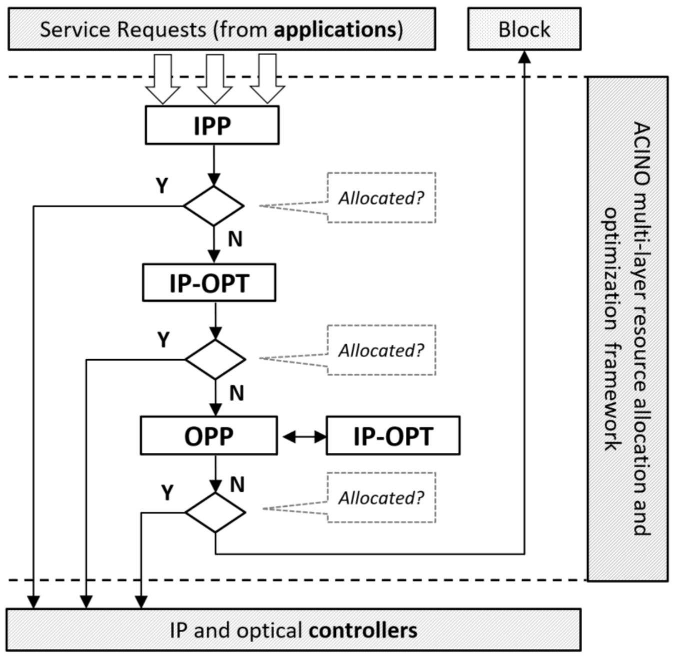

Service Requests (from applications)
Block
IPP
Y
Allocated?
N
IP-OPT
Y
Allocated?
N
OPP
IP-OPT
Y
N
Allocated?
ACINO multi-layer resource allocation and optimization framework
IP and optical controllers

Fig. 2. Resource allocation and network optimization procedure.

as possible, *(ii)* optimize the use of network resources, including power, and *(iii)* be implementable as part of an orchestrator deployed on a real network. At the same time, we observe that in IP/optical networks the two layers differ in terms of (a) capacity provisioning, (b) cost, (c) power consumption, and (d) connection setup speed. Regarding (a), we assume at least some applications do not necessarily require their own optical channel, so (iv) application flows must be groomed in the IP layer. Next, (b) is handled implicitly in this paper, as we choose optimization objectives so that they directly affect the network cost. Regarding (c) and (d), to establish a new (elastic) optical connection, power-consuming equipment such as line-cards, transceivers and transponders have to be turned on. Each such connection can take tens of seconds to set up[1]. By contrast, a new IP layer connection (e.g. an MPLS path) will use only previously-setup optical connections as its constituent IP links. The extra power needed to send packets down existing IP links is much smaller than the power required to set up new ones. IP layer connections are thus more (v) time-responsive and (vi) power-friendly.

Considerations *(i), (iv), (v)* and *(vi)* dictate that we accommodate service requests in the IP layer whenever possible. Otherwise, if the demanded service requirements cannot be satisfied by IP, we will resort to optical. Considering *(ii)*, we will purposely optimize the network, but, considering *(i)* and *(vi)*, not move established lightpaths around. Considerations *(i)* and *(iii)* call for a fast, rather than exhaustive and complex, procedure. Finally, *(iii)* calls for hitless (make-before-break) schemes, or else accommodating new applications may disrupt the running ones.

[1] Although shorter optical connection setup times have been achieved, with the currently fielded equipment they can be considered longer by at least an order of magnitude than those of IP.

The above logic guides our resource allocation procedure, which is composed of three modules: *IP Provisioning* (*IPP*), *IP Optimization* (*IP-OPT*), and *Optical Provisioning* (*OPP*). In Fig. 2, after a service request is received by the orchestrator, it tries to set up a path by using IPP. The path is then requested from the IP controller. If IPP fails, IP-OPT is called to rearrange the IP layer connections. In addition to making space for the new service requests, this IP layer re-optimization can reduce power, or optimize the network for another objective. If IP-OPT fails, the request is passed to OPP, which sets up new lightpaths (IP links). In doing that, OPP may call IP-OPT to help select the paths for new optical connections. Next, we describe some algorithmic details of each module.

IPP's primary function is to find one application-aware path (i.e., meeting all the requirements) in the IP layer for an incoming service request. For the sake of clarity, in this paper we consider only two requirements (i.e., *bandwidth b* and *latency l*). The algorithm attempts to find an application-aware path by reusing *existing lightpaths*, i.e., already-existing IP links. IPP works in three distinct phases.

In the first phase, an Auxiliary Graph (AG) [27] is constructed (*AG construction phase*). Such AG is a copy of the IP topology, which includes *(i)* all the IP nodes and *(ii)* all the existing lightpath edges (each one corresponding to an existing IP link). Among these edges, the ones not meeting b for the service request, i.e., with available bandwidth $B_{ip} < b$, are pruned. In this way, it is ensured that all the remaining edges meet the bandwidth requirement for the considered request.

After constructing the AG, a *path selection phase* is performed. The K-Shortest Path (K-SP) algorithm between the source s and the destination d for the service request is run. The weight chosen for each existing lightpath edge in the K-SP algorithm is its physical length. K-SP gives as output up to K_{ip} candidate paths, all meeting the b requirement. Among such paths, the algorithm prunes the ones with end-to-end latency $L_{path} > l$ (where l is the maximum tolerated end-to-end latency). The end-to-end latency is computed in the following way:

$$L_{path} = \sum_{node \in path} L_{node} + \sum_{link \in path} L_{link}$$

where L_{node} (L_{link}) is the latency added by a traversed IP or optical node (link). All the remaining candidate paths meet both b and l service requirements. The first path in the candidate path list (i.e., the shortest) is chosen as application-aware path for the service request and, as last step, the bandwidth b is allocated over the traversed existing lightpaths (*resource allocation phase*).

It is important to note that the input value K_{ip} must be carefully chosen, since it influences the number of candidate paths that can be investigated by the IPP algorithm. A high value for K_{ip} allows to explore a higher number of paths and to most likely find an application-aware path, but the computational time is higher. Conversely, a low value for K_{ip} allows to save computational time at the expense of neglecting some paths that could potentially be chosen.

IP-OPT works by moving IP layer routes around. Let S be the set of all service requests, and let S_{routed} be the set of requests that already have a route when IP-OPT starts. First, for each request in S_{routed}, K_{alt} candidate paths are found. To compute the candidates, like in IPP, we use the K-SP algorithm over the IP layer topology, where each link weight equals one. Next, the candidates that do not satisfy the application service requirements or do not have a cost (described below) lower than the best cost so far are discarded. Among the remainder, the first found with the lowest cost is accepted as the new route.

Once all requests in S_{routed} have been processed, the search goes back to the first request and processes the entire S_{routed} again. This continues until a pass through S_{routed} yields no cost improvement. The search may thus terminate at a cost that is only a local minimum. However, since the network continuously changes (requests arrive and leave, and IP links are set up and taken down), a later call of IP-OPT will operate on a different IP topology and thus get another chance to compute a low-cost solution. As a complementary measure, if the order of processing the requests in S_{routed} is random, IP-OPT can be restarted to find a lower cost solution.

Given the role of IP-OPT described earlier, the cost function tries to satisfy three prioritized objectives: *(i)* minimize the unsatisfied requests, *(ii)* minimize the number of links, and *(iii)* make the distribution of the link utilizations as uniform as possible, i.e. minimize the sum of the squares of the Gb/s used on each link. The cost to minimize thus equals

$$C_0 \cdot \sum_{R \in S \setminus S_{routed}} bandwidth(R) + C_1 \cdot number\_of\_links + C_2 \cdot \sum_{link} link\_traffic^2$$

$$\text{with } C_2 << C_1 << C_0$$

Because the three objectives are prioritized, the cost is calculated in a way that guarantees that the values of the three terms never overlap, i.e., the minimum of the first term is greater than the maximum of the second term. For example, C_2 can be set to one and the others adjusted accordingly. To compute the first term, for each of the up to K_{alt} candidates, we try to find, for each request in $S \setminus S_{routed}$, a shortest path (using unity link weights) that satisfies the service requirements. The paths found during the evaluation of the chosen candidate, if any, are accepted together with the candidate.

Recall that the framework must not disrupt the running services. To that end, IP-OPT maintains two lists of changes: *change_established* and *add_new*. For each service request *A* in S_{routed}, an entry $A \rightarrow P'$ is added to *change_established* for the selected candidate path *P*. If, later in IP-OPT, *P*' is changed to *P*'', $A \rightarrow P''$ is appended to *change_established*. For each service request *B* in $S \setminus S_{routed}$, when a path *Q* is found, it is added to *add_new*. If, later in IP-OPT, *Q* is changed to *Q*', we change the entry $B \rightarrow Q$ to $B \rightarrow Q'$. Once IP-OPT is finished, the orchestrator reads the entries from *change_established* in the order they were stored, and requests the route changes from the IP controller. Then the entries in *add_new* are read in an arbitrary order and implemented by the orchestrator. IP-OPT thus guarantees that, at all times, neither the admission of service requests nor abandoning the implementation of route changes (e.g. due to a failure) will disrupt the running services.

OPP establishes new optical connections (i.e., lightpaths) in the optical layer and finds application-aware paths in the IP layer by using the corresponding newly-established IP links.

OPP works very similar to IPP: it also consists of three distinct phases. The only differences between OPP and IPP are in the AG construction and resource allocation phases.

In the *AG construction phase*, OPP creates an AG including all the existing lightpath edges and prunes the existing lightpaths not meeting the *b* service requirement, as IPP does. OPP then augments the AG by adding *potential lightpath edges*. A potential lightpath edge represents whether a transparent lightpath could be established between two IP nodes on the AG.

To add potential lightpath edges, all the possible IP node pairs on the AG are considered. A potential lightpath edge between a node pair is added only if *(i)* no existing lightpath edge exists between those nodes in the current AG and *(ii)* there are enough optical resources (i.e., available contiguous spectrum slices) to potentially establish it. The addition of an end-to-end potential lightpath between *s* and *d* is prioritized over any other IP node pair. This means that, if there are enough resources to add such a potential lightpath, it will be the first choice in the following path selection phase. To check the availability of optical resources for a potential lightpath, the K-SP First-Fit (FF) routing and spectrum assignment algorithm between the considered IP node pair is run in the optical layer, which explores K_{flex} optical shortest paths. From the available transceivers, one is selected with sufficient reach and bit rate, requiring a specific number of spectral slots (possibly different for each one in elastic optical networking). A potential lightpath is then added to the AG only if K-SP FF finds a feasible solution.

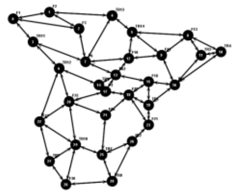
The *path selection phase* then exactly works as for IPP. However, the addition of multiple potential lightpaths to the AG increases the chance of finding an application-aware path. This comes at the expense of having to set up one or more new lightpaths in the network.

If an application-aware path is found, the *resource allocation phase* needs to allocate resources at both optical and IP layer (while IPP only allocates resources at the IP layer). At the optical layer, the contiguous spectrum slices as computed by K_{flex}-SP FF are allocated for all the potential lightpath edges traversed by the chosen path. The state for such potential lightpaths is changed from *potential* to *existing* and, lastly, the bandwidth *b* is allocated over all the traversed lightpaths (as for IPP).

Computational complexity analysis: We will express it relative to the number of calls to K-SP, which can be implemented as a polynomial time algorithm [38]. On an *N*-node network, K-SP is run once in IPP. In OPP it is run once in the IP and up to *N(N-1)* times in the optical layer. IP-OPT moves each existing IP path (one K-SP each to find the candidates), and then the outstanding incoming request[2] is accommodated by K-SP. Note that computing the cost for each of the found candidate paths requires no K-SP runs. After processing all the IP paths, IP-OPT processes them again, until

[2] Note that by default ACINO framework deals with only one new request at a time. More requests can be handled, but here we simplify the discussion.

no improvement is found. Thus, there are at most *(P+1)c* K-SP runs, where *P* is the number of existing paths and *c* is the number of passes through all the IP paths. Note that *P* depends on the current load in the network, and *c* is a function of how quickly the cost optimization reaches the local optimum. Safely assuming *P>>1*, the total complexity is $O(N^2+Pc)$ K-SP calls for each request.



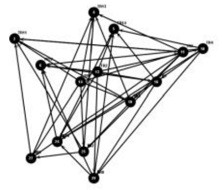


Fig. 3. (Left) test optical topology (right) example IP topology. The IP router locations are fixed, while the IP links are added or removed during the operation of the network. For each IP link line shown, there may be multiple links between the same routers.

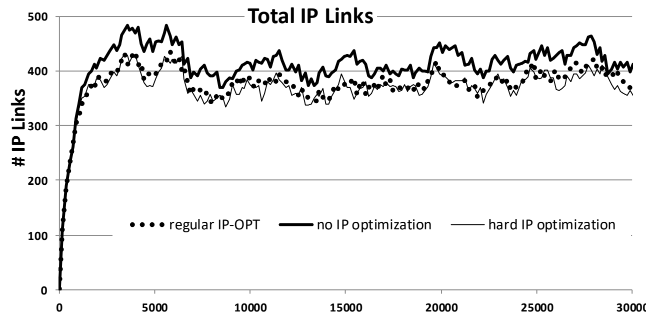
IV. Simulations and Numerical results

We now show the functioning of the framework with some illustrative tests. The test topology is the Spanish national core network (Fig. 3). 14 of the 30 nodes have IP equipment in addition to optical. The optical boxes are all colorless-directionless-contentionless, and we assume no constraints on the number of ports on any box. We start with no a priori lightpaths (IP links). The service requests arrive as a Poisson process, such that the traffic matrix of average end-to-end bitrates between IP routers is maintained. This traffic matrix was a realistic one supplied by the Spanish national provider. To accommodate the requests, we assume elastic optical transmission (variable bit-rate, modulation format, and spectrum allocation).

Even though our framework allows to explicitly specify constraints for each request, for the sake of clarity we only consider three distinct traffic classes: high-priority, latency-sensitive, and best effort. For the high-priority class, requests are routed separately over the network, down to the optical layer. In this class no connection shares resources with other connections or classes and uses 10 Gb/s transceivers. For the other two classes, resource sharing is allowed so 100 Gb/s transceivers are selected, leaving space to accommodate other requests in the IP layer. The percentages of traffic in the three classes we chose for this study are 15%, 15% and 70% respectively. The high-priority and latency-sensitive classes require a 6ms end-to-end latency guarantee. Note that here, as explained above, we only deal with bandwidth and latency as the service requirements, but the framework can readily accommodate availability and protection [39].

The high-priority class uses two spectrum slots for each optical connection, while the other classes use three slots. These optical connections have maximum lengths of 3000 and 2000 km respectively. Each fiber is assumed to have 320 slots, and we use only one fiber per link.

All *K* parameters from Section III are set to 5 and each router is assumed to introduce a 0.5 ms delay per packet. In IP-OPT, we have so far implemented ascending, descending and random orders of processing the requests in S_{routed}, and we show the results for descending here.



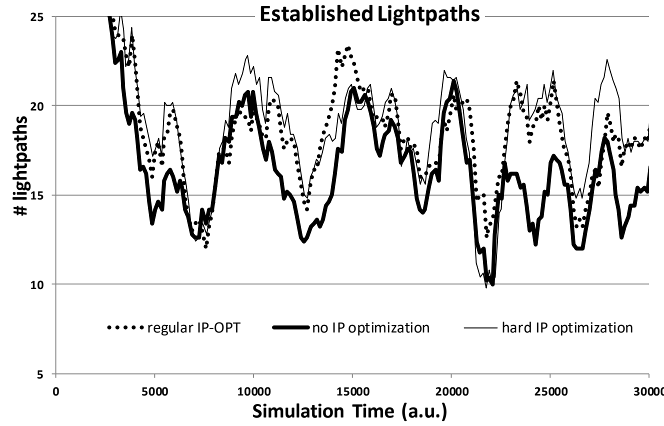


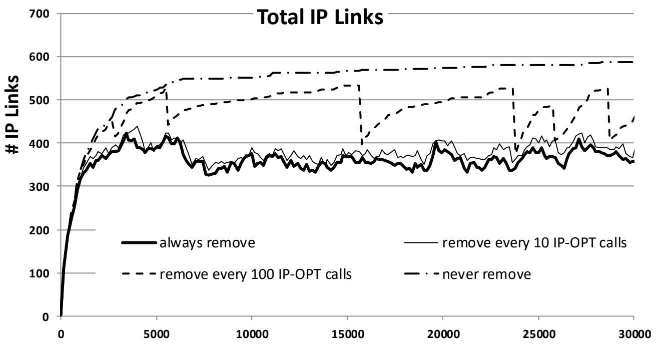
Fig. 4. Comparison of a network with and without IP-OPT and a hard optimizer that does not guarantee that each service will remain uninterrupted.
x-axis: time of every 100th service request (to de-clutter the plot).
y-axis: (top) number of IP links in the network after the request is accommodated, (bottom) number of new lightpaths established in a 100-request interval, plotted using a 10-point moving average period to de-clutter.

To run the simulations, we generate and process 10000 service requests in Net2Plan. Here, our code implements the ACINO framework, and Net2Plan simulates the network.

In Fig. 4 (top), without IP-OPT, an average 9% more IP links are needed to serve the exact same request sequence. This seemingly small advantage of IP-OPT is the result of the effective IPP and OPP, and the fact that the 15% high-priority traffic IP connections cannot be moved around. We also show in the same figure the performance of a different optimization module. This *hard* IP optimization simultaneously rearranges all connections on the network to find a better solution, but without guaranteeing a hitless way to move from one state to the other. However, only 2% fewer IP links are needed with this non-realistic approach. IP-OPT is thus observed to perform its role well, unless the requests are arriving very frequently and the orchestrator cannot set up the accommodating optical connections quickly enough. This tradeoff between power and dynamicity is illustrated in Fig. 4 (bottom), where omitting IP-OPT from the framework requires less frequent lightpath additions.

However, for every 1000 service request arrivals and departures, the difference is only one lightpath on the average, so IP-OPT is preferable. Hard IP optimization has a potential advantage over IP-OPT in that it can implement the route changes in parallel, rather going through *change_established* and *add_new*. However, in our testing *change_established* only had a few entries and *add_new* only one.

Recall that the IP-OPT runtime is a function of the number of currently accommodated demands in set S_{routed}. For faster



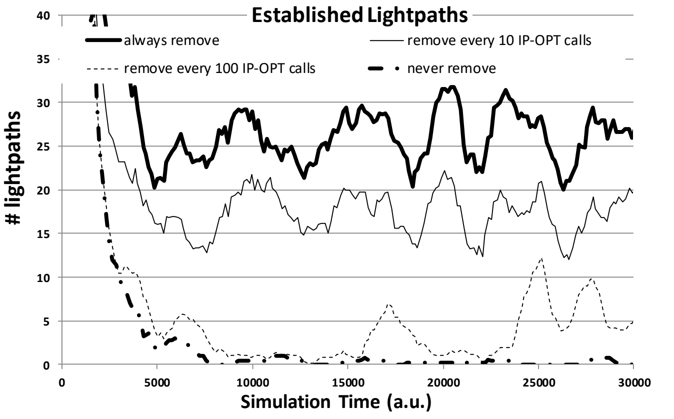
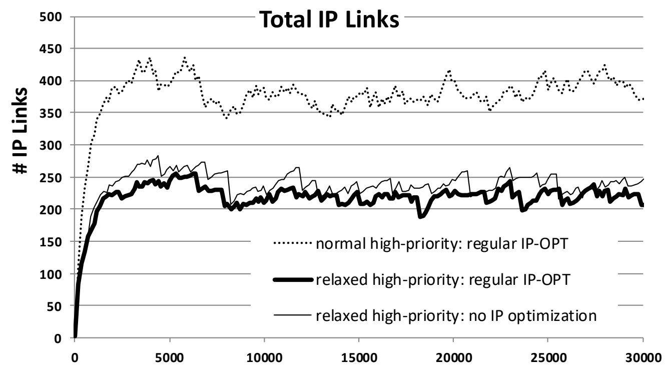


Fig. 5. Comparison of different idle lightpath removal strategies. Axes, plot intervals, and bottom curve smoothing as in Fig. 4.



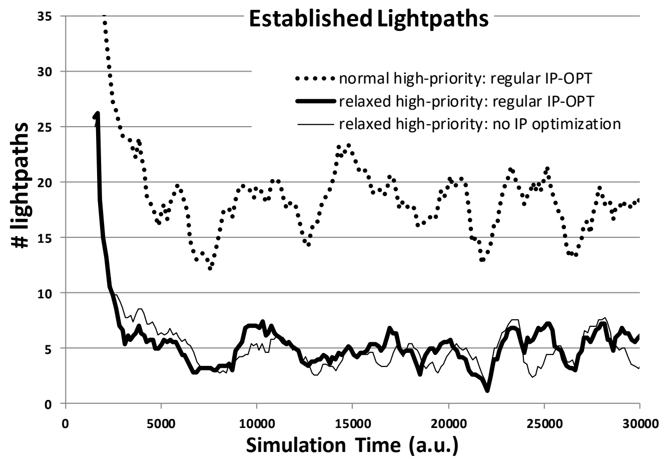


Fig. 6. The high-priority class is relaxed so that its packets can mix on the same IP links (but not with the other classes), with and without optimization. Axes, plot intervals, and bottom curve smoothing as in Fig. 4.

execution, IP-OPT can process only the most recent *D*% of S_{routed}'s members. As *D* changes from 0 to 100, the resulting curves in Fig. 4 move between no and regular IP-OPT; we do not show this to avoid further cluttering the plot.

Next, we investigate whether to remove idle lightpaths as often as possible (to save power) or to leave them available (to quickly satisfy new requests). In Fig. 5 (top), using regular IP-OPT, we show what happens with four different policies, which are the two extremes mentioned and the option to periodically remove idle lightpaths after either every 10 or every 100 calls to IP-OPT. Among the options, removing every 10 IP-OPT calls gives a good compromise, because it requires an average of eight less lightpath additions (Fig. 5 bottom), while requiring an average of only 5% more IP links than the always remove option; we thus choose 10 calls for the other simulations. When we remove every 100 calls, the saw tooth curve is the result of an interesting positive feedback loop: infrequent IP optimization leaves many IP links active, which in turn allows IPP to accommodate the requests before they get to IP-OPT in the framework, which then results in few IP-OPT calls. The non-optimized IP network then has few idle IP links, so there are few IP link removals. For the two extremes, never removing requires 52% more IP links than the always option, but always removing requires 24 more lightpath additions on the average. Fig. 5 thus shows how removal frequency is a useful parameter for tuning the balance between energy efficiency and time responsiveness of the network.

Next, we change the input traffic so that the high-priority class is less constrained. The network is now offering a service where the end-to-end connections have the same bandwidth and guaranteed latency as before, but their packets may get groomed onto common IP links. The high-priority packets still do not mix with the packets from the other two classes, i.e., high-priority service gets its own IP network, so some exclusiveness is still present. Consequently, in anticipation of sharing the IP links, the high-priority class now gets the largest bitrate transceiver, i.e., three slots. In Fig. 6 (top), relaxing the high-priority class requirement yields 42% in savings. In addition, in Fig. 6 (bottom), the orchestrator can be far less responsive to accommodate the incoming requests. Even without IP-OPT there are 37% total IP links savings, and optical setups are about as infrequent as with IP-OPT on.

In a dynamic situation, the responsiveness of the orchestrator is important, so the time to compute the resources should not dominate service accommodation times. Our runtimes are tens of ms for IP-OPT and low hundreds of ms for OPP per request on an Intel i5-based PC. Assuming that a typical time to set up an optical connection is tens of sec, and that each request must be able to wait that long, our runtimes are satisfactory.

V. Conclusion

We proposed a network resource allocation and optimization framework based on the properties of the IP and optical layers, which can be implemented on a network orchestrator.

Simulation on a test network with realistic application demands shows that our strategy provides reasonable resources for each application class. Periodic re-optimization of the IP routes helps reduce the power consumption of the network without causing service interruptions. In addition, infrequent removal of lightpaths to save energy was shown to be effective in terms of the resources savings and network responsiveness. The frequency of removal can be tuned so that the slow operation of lightpath setup is not needed too often. We also found a significant energy, cost, and responsiveness difference between a network offering separately routed connections, and one that has a less exclusive service where packets can be mixed together. Such findings may help a network operator decide which services to offer.

Acknowledgment

We thank Dr. Victor Lopez of Telefonica I+D for providing network data.

References

[1] ACINO project, https://cordis.europa.eu/project/rcn/194286_en.html

[2] V. López, J. M. Gran *et al.*, "The Role of SDN in Application Centric IP and Optical Networks", *European Conference on Networks and Communications*, 2016

[3] "Intent NBI – Definition and Principles," *Open Networking Foundation*, Tech. Rep. ONF TR-523, 2016

[4] P. Pavon-Marino, 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, 2015

[5] J. L. Izquierdo-Zaragoza, A. Fernandez-Gambin, *et al.*, "Leveraging Net2Plan planning tool for network orchestration in OpenDaylight", *Smart Communications in Network Technologies*, 2014

[6] P. Berde, M. Gerola *et al.*, "ONOS: towards an open, distributed SDN OS", ACM SIGCOMM Hot Topics in Software Defined Networking workshop, 2014

[7] O. Gerstel, V. Lopez *et al.*, "Multi-layer orchestration for application-centric networking", *International Conference on Photonics in Switching*, 2016

[8] P. Sköldström, Ć. Rožić, *et al.*, "Making Powerful Friends: Introducing ONOS and Net2Plan to Each Other", *International Conference on Transparent Optical Networks*, 2017

[9] E. Palkopoulou, O. Gerstel *et al.*, "Impact of IP Layer Routing Policy on Multilayer Design [Invited]", *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 3, pp. A396 - A402, 2015

[10] S. Das, Y. Yiakoumis *et al.*, "Application-aware aggregation and traffic engineering in a converged packet-circuit network", *Optical Fiber Communication Conference*, 2011

[11] J. Sócrates Dantas, D. Careglio *et al.*, "PCE algorithm for traffic grooming and QoS in multi-layer/multi-domain IP over WDM networks", *International Conference on Transparent Optical Networks*, 2011

[12] K. Jeong, J. Kim *et al.*, "QoS-aware Network Operating System for software defined networking with Generalized OpenFlows", *IEEE Network Operations and Management Symposium*, 2012

[13] J. Liu, L. Zhu *et al.*, "Scalable application-aware resource management in software defined networking", *International Conference on Transparent Optical Networks*, 2015

[14] S. C. Lin, I. F. Akyildiz *et al.*, "QoS-Aware Adaptive Routing in Multi-Layer Hierarchical Software Defined Networks: A Reinforcement Learning Approach", *IEEE International Conference on Services Computing*, 2016

[15] F. S. Tegueu, S. Abdellatif *et al.*, "Towards application driven networking", *IEEE International Symposium on Local and Metropolitan Area Networks*, 2016

[16] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt and G. Noubir, "Application-awareness in SDN," *SIGCOMM Computer Communication Review*, 2013

[17] M. Jarschel, F. Wamser *et al.*, "SDN-Based Application-Aware Networking on the Example of YouTube Video Streaming", *European Workshop on Software Defined Networks*, 2013

[18] M. Chamania, O. Gonzales de Dios *et al.*, "Coordinated Computation of Multi-Layer Paths via Inter-layer PCE Communication: Standards, Interoperability and Deployment", *IEEE International Conference on Communications*, 2012.

[19] A. Aguado, V. Lopez *et al.*, "ABNO: A feasible SDN approach for multi-vendor IP and optical networks", *Optical Fiber Communications Conference*, 2014.

[20] F. Pederzolli, D. Siracusa *et al.*, "SDN application-centric orchestration for multi-layer transport networks", *International Conference on Transparent Optical Networks*, 2016

[21] A. Khalil, A. Hadjiantonis *et al.*, "Dynamic Routing of "Sub-Wavelength" Connections in IP/MPLS over WDM Networks", *Optical Communications Systems and Networks Conference*, 2004

[22] H. Alshaer, J. M. H. Elmirghani, "Multilayer Dynamic Traffic Grooming with Constrained Differentiated Resilience in IP/MPLS-over-WDM Networks", *IEEE Transactions on Network and Service Management*, vol. 9, no. 1, pp. 60-72, 2012

[23] A. Bukva, R. Casellas *et al.*, "A dynamic path-computation algorithm for a GMPLS-enabled multi-layer network", *IEEE/OSA Journal of Optical Communications and Networking*, vol. 4, no. 6, pp. 436-448, 2012

[24] S. Zhang, C. Martel *et al.*, "Dynamic Traffic Grooming in Elastic Optical Networks", *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 1, pp. 4-12, 2013

[25] J. Zhang, Y. Ji *et al.*, "Dynamic virtual network embedding over multilayer optical networks", *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 9, pp. 918-927, 2015

[26] 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

[27] I. Chlamtac, A. Farago *et al.*, "Lightpath (Wavelength) Routing in Large WDM Networks," IEEE *Journal on Selected Areas in Communications*, vol. 14, no. 5, pp. 909–913, 1996

[28] J. Wu, "A survey of WDM network reconfiguration: Strategies and triggering methods", *Computer Networks*, vol. 55, no. 11, pp. 2622-2645, 2011

[29] A. Brzezinski, E. Modiano, "Dynamic reconfiguration and routing algorithms for IP-over-WDM networks with stochastic traffic", in *Journal of Lightwave Technology*, vol. 23, no. 10, pp. 3188-3205, 2005

[30] P. N. Tran, U. Killat, "Dynamic reconfiguration of logical topology for WDM networks under traffic changes", *IEEE Network Operations and Management Symposium*, 2008

[31] P. N. Tran, U. Killat, "Distributed algorithm for dynamic logical topology reconfiguration in IP over WDM networks", *IEEE Symposium on Computers and Communications*, 2009

[32] D. Bhamare, A. Gumaste *et al.*, "Multi-layer optimization for service provider transport networks", *IEEE Conference on Local Computer Networks*, 2013

[33] G. Shen, Y. Lui *et al.*, ""Follow the Sun, Follow the Wind" Lightpath Virtual Topology Reconfiguration in IP Over WDM Network", *Journal of Lightwave Technology*, vol. 32, no. 11, pp. 2094-2105, 2014

[34] M. Caria, M. Chamania *et al.*, "A comparative performance study of load adaptive energy saving schemes for IP-over-WDM networks", *IEEE/OSA Journal of Optical Communications and Networking*, vol. 4, no. 3, pp. 152-164, 2012

[35] Y. Nomura, H. Yonezu *et al.*, "Dynamic topology reconfiguration for energy efficient multi-layer network using extended GMPLS with link power control", *Optical Fiber Communications Conference*, 2012

[36] V. Gkamas, K. Christodoulopoulos *et al.*, "A comparison study of joint and sequential multi-layer planning for IP over flexible optical networks," *Optical Fiber Communications Conference*, 2015

[37] L. Velasco, A. Castro *et al.*, "In-operation network planning", *IEEE Communications Magazine*, vol. 52, no. 1, pp. 52-60, 2014

[38] A. W. Brander, M. C. Sinclair, "A Comparative Study of k-Shortest Path Algorithms", *Performance Engineering of Computer and Telecommunications System*s, 1996

[39] M. Savi, Ć. Rožić *et al.*, "On the Benefits of Multi-Layer Application-Aware Resource Allocation and Optimization", *European Conference on Networks and Communications*, 2017

[40] Ć. Rožić, M. Savi *et al.*, "A Framework for Dynamic Multi-Layer Resource Allocation and Optimization in Application-Centric Networking," *Optical Fiber Communications Conference*, 2017