Energy Saving Through Traffic Profiling in Self-Optimizing Optical Networks

Federico Pederzolli, *Student Member, IEEE,* Domenico Siracusa, *Member, IEEE,* Elio Salvadori, *Member, IEEE,* and Renato Lo Cigno, *Senior Member, IEEE*

***Abstract*—An increasing fraction of the electrical energy produced in western countries is being consumed by Internet infrastructure; reducing its energy footprint is therefore of the utmost importance for the scalability of the Internet. We address optical transport backbones, and propose a novel method to reduce the energy consumed by dynamically adjusting the number of active optical carriers to support the short-term load of the network with a small and controllable margin. This is achieved in a non-disruptive manner that does not interact with routing strategies and does not rely on any specific control plane, but exploits automated traffic profiling and prediction of the well known circadian traffic cycle. The proposed approach works with both fixed and flexible grid optical networks. We describe a method to automatically learn these patterns, and multiple techniques to predict incoming traffic. Furthermore, we present an algorithm that tunes the parameters of the proposed system in order to achieve a target a-posteriori probability of causing traffic losses. The behavior of the system is studied, using simulations, under a variety of conditions. Results show that the proposed prediction algorithms can significantly reduce the number of active optical carriers, even in non-optimal scenarios, while guaranteeing low traffic losses.**

***Index Terms*—Optical fiber networks, Energy efficiency, Communication system control, Communication system traffic, Green design.**

I. Introduction

Reducing the energy footprint of the Internet has long been a hot topic in network research, fueled by the economic interest of telecom operators, and recently also by social pressures and ecological concerns. Current estimates of the energy consumption of the Internet range between about 0.4 to 10% of the electricity consumed in industrialized countries [1]–[3]. The wide range of these figures is due to difficulties in the estimation, but also because they may or may not include estimates for the energy consumption of:

- Network elements: access, metro and core routers and line cards;
- Content-providing attached devices: servers, data-centers, and their added costs (e.g., cooling);
- Household hosts and other users devices.

F. Pederzolli, D. Siracusa and E. Salvadori are with CREATE-NET, Via alla cascata 56/D, 38123 Trento, Italy. email: {federico.pederzolli, domenico.siracusa, elio.salvadori}@create-net.org.

R. Lo Cigno is with DISI, University of Trento, Via Sommarive 9, 30123 Trento, Italy. email: locigno@disi.unitn.it.

This manuscript is an extended version of the paper D. Siracusa, F. Pederzolli, R. Lo Cigno, and E. Salvadori, "Energy saving through traffic profiling and prediction in self-optimizing optical networks", In *2014 Optical Fiber Communications Conference and Exhibition (OFC)*.

Furthermore, to obtain a complete picture of the environmental impact of the Internet, the energy cost of component fabrication and the system effects induced by the network should also be accounted for. The latter include the IT-induced savings in other systems, such as transportation and manufacturing, and the rebound effects that these savings enable, as outlined in [4]. When taken all together including the induced savings, these factors may actually sum up to a reduction of energy consumption; however, the topic of reducing the energy consumption of modern communication networks remains important, and the fact that the Internet helps reducing energy consumption in other sectors cannot be an excuse to avoid looking at its self-consumption.

Most metro and core networks rely on optical technology, mostly due to the very high throughput that this technology can achieve at very low costs (compared to other technologies). The authors of [3] estimate that core networks alone consume about one third of the energy consumed by all network elements.

A broad and still largely untapped avenue to improve the energy efficiency of modern transport networks is to exploit the daily fluctuations in the overall traffic they support. Measures taken from [5], [6] confirm that such fluctuations exist, and that, at least at high aggregation levels, they exhibit regularly repeating circadian patterns. An example of these patterns, taken from [5], is depicted in Fig. 1; it is immediately clear that, in the dead of night, the load offered to the network is at most one third of what it is during peak hours.

The contribution of this work, which expands [7], is a novel and practically implementable methodology to save energy. Technologically we only require systems to be able to switch on and off carriers with their transmitter and receiver units. More in detail, we propose a self-optimizing optical network that automatically learns the traffic patterns mentioned above, and exploits the patterns to predict the amount of incoming traffic in the near future. Based on this estimation redundant optical carriers are switched off. The proposed system also ensures that the ability of the network to carry the incoming traffic is not disrupted.

This approach is applicable to both Wavelength Division Multiplexing (WDM) and flexible optical networks, as long as a majority of optical connections are served by multiple carriers, which are the elementary units of transmission. In practice, this could mean WDM or flexi-grid networks serving connections with multiple independent Lightpaths (LPs) or super-channels (multiple independent optical signals belonging to the same connection and closely spaced in the spectrum domain),

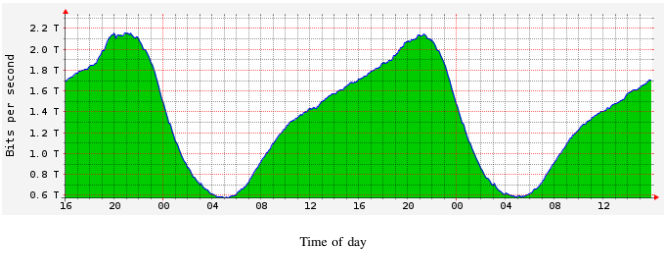


Fig. 1. Circadian traffic pattern experienced by the AMS-IX network [5].

or flexi-grid networks using optical Orthogonal Frequency-Division Multiplexing (OFDM) devices as modulation and bandwidth-variable transponders [8]. A sub-system to allow the network to automatically collect the necessary data for the learning process (from some monitoring system) is described as part of this work, solving a major hurdle that prevents the direct applications of some other proposals available in the literature. The exact amount of energy saved by this system depends on the underlying technology, since the power consumed by an optical carrier depends on the technology, the carrier capacity and the transmission distance. The typical range is $[10 \div 500]$ W according to [9]. Therefore, at the level of abstraction used in this work, the potential savings from using our system are expressed as the fraction of time that carriers are kept powered off. A notable characteristic of the proposed system is that it does not interact with the routing strategy of the network: the configuration of optical switches (which lambdas or slots are assigned to which carriers, where and how they are routed) is completely unchanged.

This paper is organized as follows. Sect. II describes related works that tackle the same problem, highlighting the novelty of our approach. Next, a detailed explanation of the approach proposed in this work is given in Sect. III. Sect. IV describes the experimental setup used to produce the results discussed in Sect. V. Finally, Sect. VI summarizes our findings and discusses other possible uses for the data collected with our methodology.

II. Related works

According to [10], the approaches to green networking available in literature can be classified in three broad categories:

- *Re-engineering/re-design*: produce more efficient components, or more efficient architectures for components and networks;
- *Dynamic adaptation/Sleeping/Standby*: modulate hardware capabilities, and thus energy consumption, to follow the trends exhibited by the network, while still offering some form of service (dynamic adaptation), or being fully idle (sleeping/standby);
- *Energy Aware Routing/Green Traffic Grooming*: include constraints that minimize the overall energy consumption in the Routing and Wavelength Assignment (RWA)/Routing and Spectrum Assignment (RSA) calculation and the network planning phase.

The system proposed in this work falls in the dynamic adaptation category. The rest of this section describes some previous works that also fall in the same category.

The authors of [11] propose to establish multiple LPs for each source-destination pair in the network, dynamically setting them up or tearing them down to accommodate real traffic requests. To account for the non-instantaneous setup time of LPs, as well as to account for the inability of such a system to predict future increases in traffic, the authors propose to use a simple heuristic, i.e., using an artificially inflated traffic measure for the purpose of determining the number of LPs needed to carry the current traffic load. While this work predates flexible optical networks (it was developed for WDM networks), where its efficiency is limited, it can also work in the context of flexible optical networks.

The authors of [12] suggest to compute a set of network interfaces (transponders) and routing nodes to turn off based on a predictive traffic matrix the load in the next two hours. They describe a Mixed Integer Linear Programming (MILP) technique to do so. However, no provision is given on how to obtain the predictive matrices; furthermore, as the authors themselves point out, each time the topology of active LPs changes (as often as every two hours), traffic may be disrupted while the network re-configures. Finally, the use of MILP techniques, which require large amounts of computational resources and knowledge of the whole state of the network, restricts this approach to running in an environment with centralized control, and likely offline.

Similarly, the authors of [13] propose a different Integer Linear Programming (ILP) technique, that takes as input a traffic matrix to determine the minimum set of nodes and links that must be powered to serve the requests. As in [12], no hints are given as to how to produce such a matrix, or how it can describe future traffic.

The authors of [9] suggest to simply measure the amount of traffic being carried by an LP, and adjust the resources used to power its protection twin accordingly. This has the distinct advantage of not affecting the ability of the network to carry traffic, since the main resources always remain fully powered. It does increase the risk of losing traffic during transitions to the protected path, however such events are the consequence of failures, therefore such a risk may be acceptable. The work makes use of a "realistic" traffic matrix, known in advance, to compute which protection resources to shut down.

Finally, the authors of [14] propose a system that periodically recomputes the virtual topology of a network, aggregating requests that share the same source and destination into larger LPs, or simply forcing the IP layer to reroute traffic on the available logical links. The proposed system, based on a centralized control architecture, also deals with the collection of the necessary data. However, as stated by the authors, the change in topology interacts badly with the IP layer, which needs to update its routing tables after each topology change. While modern routing protocol can achieve less than 50 ms convergence using fast reroute techniques, these are not applicable in this context. Therefore, routing convergence may take in the order of 100-250 ms, which has a detectable and detrimental impact on traffic.

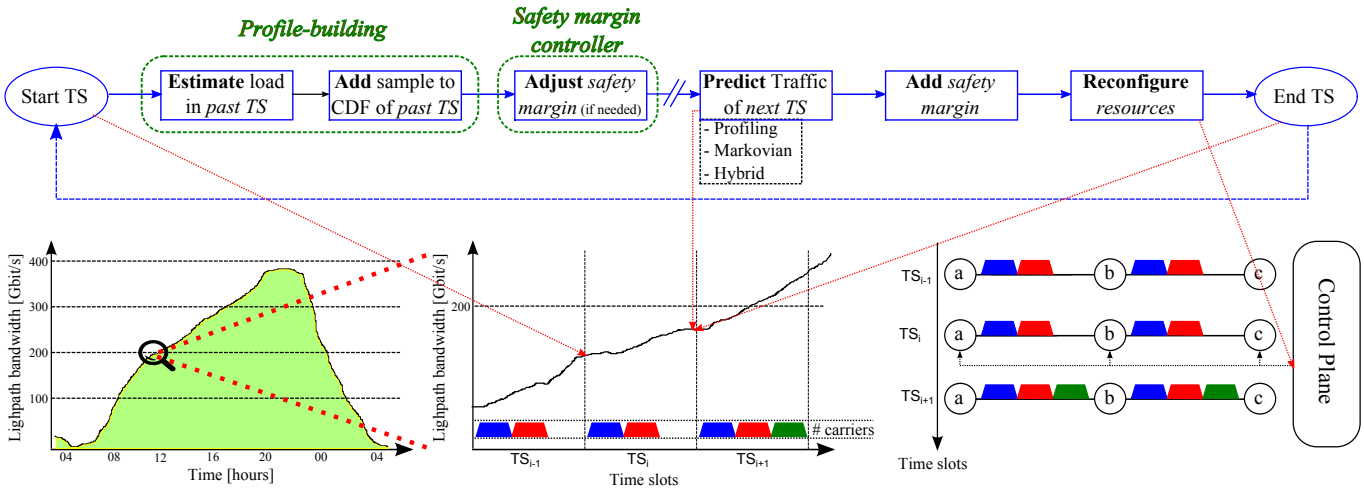


Fig. 2. Overview of the proposed process: each day is divided in a fixed number of TS; for each LP, at the beginning of TS_i the load experienced during TS_{i-1} is estimated and stored in the model; then, the safety margin may be tuned (for example, it could be increased if traffic loss was experienced during TS_{i-1}); near the end of TS_i, the traffic to be supported by this LP in TS_{i+1} is predicted, and resources are reconfigured if necessary.

A feature common to most of these works is that they either cause traffic disruptions, or assume the system has access to a predictive traffic matrix without dealing with the problem of creating one (or both). No work in literature, to the best of our knowledge, offers a solution that avoids both pitfalls.

III. Proposed Approach Overview

The core idea supporting this work is to dynamically fine-tune the amount of active transmitters and receivers in order to offer a controlled amount of additional capacity, rather than the huge over-dimensioning of optical pipes observed in the operation of modern transport networks.

A note on nomenclature: in this work, we use LP to refer to optical connections, including also the cases where inverse multiplexing is used to assign two or more separate slices of optical resources to the same logical service.

This work is based on two assumptions:

1) There exists a pattern of traffic offered to each LP, with a period of one day;
2) Any LP lasts, on average, at least a few tens of days.

The first assumption is supported by the evidence available from [5] and [6] (an example taken from AMS-IX network is depicted in Fig. 1). The second assumption is a direct consequence of how optical transport networks are currently operated (namely, optical circuits are rarely created and almost never torn down, thus potentially lasting years).

In this work we present a technique that allows the network to self-learn the traffic patterns experienced by LPs, and then exploit them to reduce the overall energy consumption of the network by switching off un-needed transceivers or carriers (in the case of sliceable transponders). At a very high level, this process can be broken down as shown in Fig. 2: we use discrete time to model the incoming traffic patterns; each point represents the average traffic experienced by a single LP during a TS. Optical transmission hardware requires an amount of time in the order of several tens of seconds to properly set up a new optical carrier to allow lasers and amplifiers to perform power equalization. Therefore a TS must last, independently of the control plane technology used (e.g.,Generalized Multi-Protocol Label Switching (GMPLS) or Software Defined Networking (SDN)), at least some minutes to allow the actual state of the network to converge to its new configuration.

The process is applied separately to every LP, since optical links in a network do not carry symmetric or even similar traffic, and indeed evidence shows that there are significant differences in the distribution (and sometimes pattern) of traffic among different links and often also between the two directions of a single link. As shown in Fig. 2, for each active LP at each TS, an algorithm builds and updates the traffic profile associated to the LP, then another algorithm predicts the future traffic and adjusts it with a small safety margin. Based on this information, the carriers reserved for the LP are turned on or off to support the predicted amount of traffic with a given probability by means of whatever control plane protocol the network employs. In addition, a control algorithm is in charge of dynamically tuning the safety margin parameter in order to achieve the desired probability of mispredicting the real traffic.

Remarkably, the proposed system does not impact the state of resource reservations in the network: transmission hardware is put in standby mode to conserve energy, but it is not leased to other connections; the configuration of optical switches is completely unchanged, so the system does not perform any re-routing; finally, the spectral resources reserved for an LP, but not actually in use due to low input traffic, are not made available to other connections. Dynamic leasing of transmission and spectral resources are other, interesting application areas that may exploit traffic pattern prediction. They are not dealt with in this work, whose goal is the reduction of energy consumption, but some possibilities are briefly outlined in Sect. VI-A.

A. Profile-Building Process

The first step is to build a profile of the average/peak traffic carried within each TS starting from the set of measurements of the actual loads historically observed on that LP. A monitoring procedure, running at either the tail or head end of each LP, implements the measurement of the LP load, and transfer this information to an entity that builds the profile.

The information transferred by the monitoring procedure (at the end of each TS) is a proper estimate of the traffic received during the last TS, which could be the peak or average observed, or the maximum of multiple averages (given that we are interested in supporting peak traffic). Short-term bursts above the supported traffic amount are expected to be handled by electronic buffers at the ingress of an LP, most likely a router. Bursts spanning multiple TS can be absorbed by adaptation strategies, like the *Hybrid* procedure described in Sect. III-B3. The high level of multiplexing in LPs, however, guarantees that short-term bursts are a small fraction of an LP traffic. The measure is quantized and classified into one of n classes (e.g., one every 1 Gb/s) by performing a ceiling operation. The quantized value is then stored in a message together with the identifiers of the LP being monitored and of the TS it belongs to. The monitoring entity can run on the node managing the LP or not, in which case the information is sent as a control message to the managing node. This mechanism requires the presence of some form of clock synchronization, such as Network Time Protocol (NTP), to ensure that the time ranges encoded by the TS identifiers do not drift, but synchronization requirements are not critical as TSs are long.

Nodes responsible for managing LPs (all nodes in a distributed control architecture and just control nodes in a centralized one) allocate the proper data structures to store the traffic profile whenever a new LP is established. The profile consists of a time-stamped collection of quantized traffic measures in the range $[0, C]$, where C is the capacity of the LP in some proper unit measures, for every TS within the day. In other words, the traffic profile is a multi-dimensional empirical Cumulative Probability Distribution Function (CDF) for every LP; the number of dimensions is equal to the number of TSs in a day. The CDF is built incrementally at each new measure (or message) avoiding to memorize a long list of measures that would result in a loss of performance over long operational times, as the total number of measures can grow very large. Algorithm 1 present the pseudo-code for the implementation of the procedure just described. More specifically, iterating over all measures would take $\Theta(n)$ steps, where n is the number of traffic classes, which is fixed, rather than $\Theta(m)$, where m is the number of measurements for a given LP and TS, and increases with each day the LP remains in operation. Furthermore, since our CDF representation is intrinsically ordered, computations which involve finding a maximum/median or similar value only have a complexity of $\mathcal{O}(n)$. Finally, since a LP may remain operational for multiple years, and in order to let the system adapt to slow traffic changes, we periodically halve all recorded values and counters, i.e., $\forall_{i,ts,LP} LP$.usage[ts][i] *= 0.5 and LP.numMeasures[ts] *= 0.5.

Algorithm 1 Traffic measurement and Profile building

```
1: function PREPARENEWLP(Lp LP)
2:     LP.usage ← int[NumSlotsInModel]
3:     for int i ← 0, i < NumSlotsInModel, i++ do
4:         LP.usage[i] ← int[MaxTraffic/TrafGranularity]
5:     end for
6:     LP.numMeasures ← int[NumSlotsInModel]
7: end function
8: function SENDTRAFFICMESSAGE(Msg msg)
9:     while LP is active do
10:        WAITUNTILNEXTTS()
11:        int m ← ESTIMATETRAFFICLASTTS(LP.id)
12:        m ← QUANTIZE(m)
13:        SENDTOMANAGER(LP.id, GETLASTTSID(), m)
14:    end while
15: end function
16: function RECEIVETRAFFICMESSAGE(Msg msg)
17:    LP ← LPDB.FIND(msg.id)
18:    for int i ← 0, i < msg.traffic, i++ do
19:        LP.usage[msg.ts][i]++
20:    end for
21:    LP.numMeasures[msg.ts]++
22: end function
```

B. Predictor Processes

Once enough samples have been collected, the next step is to have a mechanism to predict the amount of traffic that needs to be supported in the upcoming TS, so that if additional optical carriers are expected to be needed, they can be provisioned in time.

Roughly speaking, the idea behind the prediction process is to predict, at the beginning of each TS and for each LP, a measure of the incoming traffic that is as tight as possible (to maximize savings) while at the same keeping the probability of losing traffic (which happens whenever the prediction is lower than the actual value) under a certain threshold.

In mathematical terms, if we denote a prediction as C_i^k, where i is the TS index and k the LP index, then its value should be such that, calling $x_i^k(j)$ the real traffic over the same LP during the same TS of day j, the probability $\Pr(x_i^k(j) > C_i^k)$ should be very close to a target value ϵ that meets traffic service levels, e.g., $\epsilon = 10^{-6}$.

If the form of the statistical distribution of the offered-load-per-timeslot value were known, the system could simply employ *maximum-likelihood estimation* to infer the parameters of such distributions from the data it collects. However, no work in literature, as far as we know, provides a distribution for these statistical models, so, without any real data to attempt to infer one, we resort to heuristic approximations.

Three such heuristic predictors were devised, and are presented here.

1) Profiling: A simple heuristic to estimate the traffic to be carried during a timeslot is to use the R^{th} percentile of the estimated CDF for that TS and LP, taking care of selecting an R high enough so that it captures most of the past data points, but filters the highest measures, which may be outliers. To accommodate high traffic rates that are not outliers, but are

Algorithm 2 Profiling Predictor

```
1: function PREDICT(Lp LP, int ts)
2:     if LP.numMeasures[ts] < MinNumMeasures then
3:         LP.SUPPORTTRAFFIC(LP.maxCarriableTraffic)
4:     else
5:         int predictedTraffic ← 0
6:         for int i ← 0, i < NumSlotsInModel, i++ do
7:             if (LP.usage[ts][i] / LP.numMeasures[ts]) > M
   then
8:                 predictedTraffic ← i + S * LP.capacity
9:             end if
10:            LP.SUPPORTTRAFFIC(predictedtraffic)
11:        end for
12:    end if
13: end function
```

Algorithm 3 Markovian Predictor

```
1: function PREDICT(Lp LP, int ts)
2:     int t ← LP.GETTRAFFICCARRIEDLASTTS()
3:     LP.SUPPORTTRAFFIC(t + S * LP.capacity)
4: end function
```

Algorithm 4 Hybrid Predictor

```
1: function PREDICT(Lp LP, int ts)
2:     if LP.numMeasures[ts] < MinNumOfMeasures then
3:         LP.SUPPORTTRAFFIC(LP.maxCarriableTraffic)
4:     else
5:         float d ← LP.DERIVATIVECDF(ts, M) * τ
6:         d ← SIGN(d)*CEIL(ABS(d))
7:         int t ← LP.GETTRAFFICCARRIEDLASTTS()
8:         P.SUPPORTTRAFFIC(t + d + S * LP.capacity)
9:     end if
10: end function
```

by definition above the R^{th} percentile, the prediction may be adjusted adding a small *safety margin* S in the form of a fraction of the nominal capacity of the current LP; however, large values of S would make any saving impossible.

This approach can be described as computing a sort of historical maximum (discounting outliers) for each TS of each LP as shown in Algorithm 2.

2) Markovian: Rather than learning a pattern of the traffic experienced by a LP as in the Profiling predictor, an alternative, memory-less approach, inspired by [11], is to simply consider traffic experienced during the last TS and, based on the assumption that the traffic is autocorrelated [15], just predict that traffic plus a safety margin S large enough to accommodate the largest increase in traffic that is possible between consecutive TSs. Algorithm 3 shows this approach.

Unlike the former, this approach works reasonably well even when the underlying traffic does not exhibit an actual circadian pattern; however, this only holds for LPs whose experienced load is fairly smooth. The available data suggests that this is the case on most links in transport networks, except the smallest, most under-utilized ones. However, finding an optimal value for S is non-trivial, and, since this approach cannot distinguish between increasing and decreasing traffic, it leads to excessive capacity (and thus lower savings) during downward slopes.

3) Hybrid: As stated earlier, an heuristic approach based on the last measure has the advantage of being naturally able of handling deviations from the standard pattern, but it is relatively inefficient, especially during downward gradients at night, were some experience would suggest that the traffic systematically decreases. Here the Markovian predictor needlessly keeps a large margin because it simply does not know whether the offered load will increase or decrease during the next TS. To overcome this limitation, it is possible to use a slightly modified pattern learning process to build a CDF for the first order derivative of the traffic, or to be more correct their simple difference quotient, in order to provide an estimation of a likely upper limit to the increase (or decrease) in the traffic to be carried in the next TS.

In short, this algorithm simply takes the same prediction that the Markovian predictor would make, and further adjusts it to leave more room if more traffic is expected, or decrease the margin if a decrease is historically likely.

C. Safety Margin Control Algorithm

Predictions are obviously affected by errors, leading to possible traffic losses. A network operator will in general try to maximize the energy saved while keeping the probability of losing traffic below a certain threshold dependent on its Service Level Agreements (SLAs). To achieve this objective, a solution is to tune the S parameter, which is shared by all predictors and controls the fraction of the nominal capacity of a lightpath which is added to every prediction. We identified two possible approaches to solve this control problem: one is to periodically measure the sample misprediction probability (i.e., the probability of losing traffic due to an excessively low prediction), and react accordingly, while the other is to immediately enlarge the margin after every misprediction, then periodically diminish it, acting on the period between reductions to obtain a desired a-posteriori misprediction probability.

More in detail, the first solution entails collecting enough samples to have a meaningful chance of observing mispredictions even with very low target probabilities. Even assuming the collection of multiple samples per TS, which is anyway upper bounded by the limit of the observation period to obtain each sample, would require to wait time spans of multiple months between checks. Even if we ignored this problem, the algorithm would then have to change the value of S by a steadily decreasing amount to obtain convergence, which assumes that the underlying traffic pattern does not change in the meantime.

The second approach is simpler and much more reactive: at each TS, if a misprediction happens, S is immediately increased by the amount of lost traffic, according to the following formula:

$$S_{i+1} = \begin{cases} S_i + (T_i - E_i) & \text{if } T_i > E_i \\ S_i & \text{otherwise} \end{cases}$$

Where S_i is the value of S, E_i is the predicted traffic and T_i is the measured traffic that is actually experienced during

timeslot i. The value of S is then periodically halved if after a certain period of observation the observed probability of mispredicting is below the target one, or if at least one instance of traffic loss was registered since the last check:

$$S_{\text{new}} = \begin{cases} S_{\text{old}}/2 & \text{if } \hat{P}_{Loss} < P_{Target} \\ S_{\text{old}}/2 & \text{if a loss occurred since last check} \\ S_{\text{old}} & \text{otherwise} \end{cases}$$

Where $\hat{P}_{Loss}$ denotes the empirical misprediction probability and P_{Target} the desired target loss probability. Finally, the distance (in timeslots) between checks, denoted as T, is adjusted according to the following formula:

$$T_{\text{new}} = \begin{cases} \min(T_{\text{old}} * 1.5, T_{\text{max}}) & \text{if } \hat{P}_{Loss} > P_{Target} \\ \max(T_{\text{old}} * 1.5, T_{\text{min}}) & \text{if } \hat{P}_{Loss} < 0.9 * P_{Target} \\ T_{\text{old}} & \text{otherwise} \end{cases}$$

Where T_{max} and T_{min} are bounds on the check period length. Obviously, T_{min} cannot be lower than 1 timeslot, while T_{max} should be large enough to allow $\hat{P}_{Loss}$ to re-converge after experiencing a traffic loss.

In this work we focused on the latter solution. We realize it may be non-optimal, but investigating more sophisticated algorithms with better convergence properties is beyond the scope of this work.

IV. Simulation Setup

The proposed approach, including all the prediction algorithms described in Section III, is implemented in a custom event-driven simulator based on the *omnet++* framework and developed within the CHRON EU project [16], that we are documenting to make it available to the community in the near future. In order to better understand the relative performance of the proposed predictors in different circumstances, we define two main performance metrics:

1) Fraction of Carrier Downtime (FCD): the average fraction of time, over a number of days and LPs, that a transmission unit (the signal from a transponder, or an OFDM sub-carrier), randomly chosen among those reserved to a LP is powered down using a certain predictor;
2) Sample Misprediction Probability (SMP): the fraction of TSs where the decisions of a predictor result traffic loss.

The first metric directly correlates with the amount of energy saved: the power consumption [9] depends solely on the transmission rate and transponder type; consequently FCD is a direct measure of the energy saved.

The second metric, instead, gives a quantitative measure of the losses that a predictor incurs in. It should be noted that measuring a low SMP is difficult, since the absence of measured errors in an experiment involving n samples only implies that the average error probability is lower than $1/n$ with probability greater than $1/2$, assuming that average to be normally distributed, as per the central limit theorem.

A. Simulation Parameters

We consider LP connections of 100, 200, 300 and 400 Gbit/s, equally distributed and served using super-channels whose carriers are capable of transmitting 1, 10, 40 or 100 Gbit/s (a range of values that covers both deployed optical hardware and OFDM sub-carriers). The traffic of each LP is modeled as follows: during the initialization of an LP with capacity C, two points in the range $[0, C]$ are chosen randomly to represent the minimum and maximum traffic to be carried on that LP. Then, a traffic generation pattern is assigned to the LP. Traffic patterns are:

1) *AMS-IX*: a daily pattern, based on [5].
2) *GARR*: a daily pattern, but with much less traffic every 6^{th} and 7^{th} day (i.e., week-ends), based on [6].
3) *Constant*: a fixed constant pattern.
4) *Anomaly*: a fixed constant pattern, with an irregular anomaly with double the normal value, inspired by [6]. This scenario is also somewhat representative of crowd and burst congestion.

All experiments simulate 400 days of operation, each of which is divided in 144 TSs (that is a TS lasts 10 minutes). The Profiling and Hybrid predictors were configured to enforce a 10 day training period, during which they collect measures to build a traffic profile but predict as a Markovian predictor.

For each LP and TS the average traffic is drawn from a Gaussian distribution with average equal to the value given by traffic model assigned to that LP, and a variance of 0.1 times the nominal capacity of the LP. Finally, a first order low pass filter (i.e., an exponential average) introduces correlation between subsequent slots to emulate traffic continuity, with a default parameter α of 0.6. In the rest of this paper, unless otherwise stated, the rest of the default parameters are:

- Carrier capacity/throughput: 10 Gbit/s;
- Starting safety margin $S = 0.15$;
- Target SMP range: $[0.9, 1] \cdot 10^{-3}$;

While there exist much more sophisticated models in literature, such as [17]–[21], we deem this simple model is realistic enough to highlight the different behavior of the proposed predictors, with the added benefit of simplifying the reproducibility and interpretation of results.

The topology of the simulated network is irrelevant to the system being described, since it does not interact in any way with the routing strategy at the optical layer. For completeness, we mention that we used the Deutsche Telekom topology taken from [22].

V. Results

A. Safety Margin Control Algorithm performance

The purpose of this experiment is to verify the convergence performance of the safety margin control algorithm, described in Sect. III-C. It is done using the AMS-IX traffic pattern and the default parameters.

Fig. 3 shows both the average FCD (left axis, top curves) and the SMP (right axis, bottom curves) for all predictors, namely Profiling (green), Markovian (blue) and Hybrid (red); the same color code is used throughout this Section.

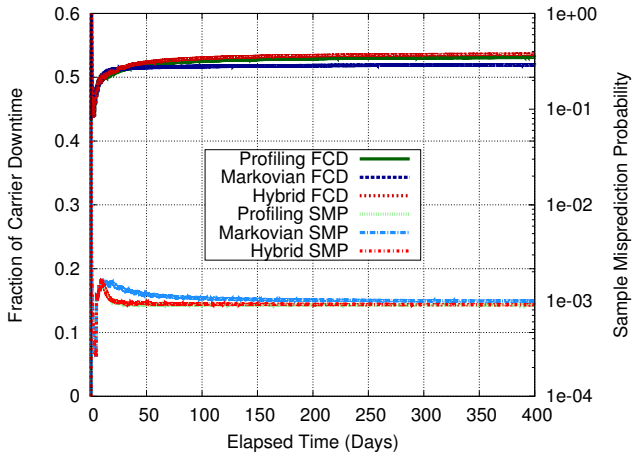


Fig. 3. Fraction of Carrier Downtime (top curves) and Sample Misprediction Probabilities (bottom curves) vs. Elapsed Time for all three predictors. The measured SMPs all converge to the desired target of 10^{-3}.

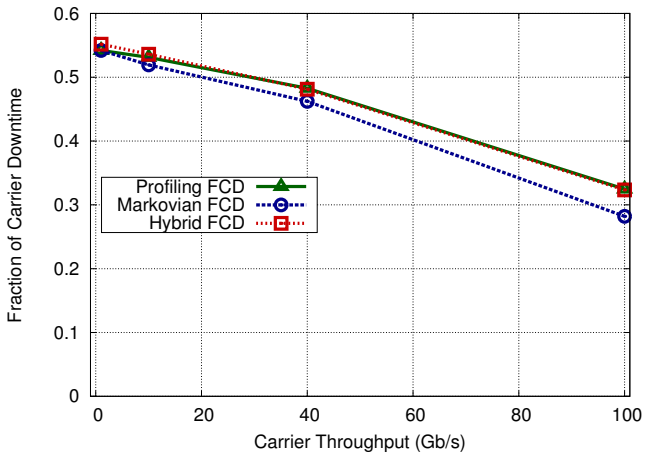


Fig. 4. Fraction of Carrier Downtime vs. Throughput of a single carrier. Using finer-grained carriers improves savings.

A first observation is that the proposed control algorithm is able, under these conditions, to attain an a-posteriori SMP within the desired range within 25 days. Using different carrier capacities, we always obtained SMPs in the interval $[0.5, 5] \cdot 10^{-3}$, which is close enough to the target.

B. Impact of carrier throughput

The purpose of this experiment is to understand the influence of the choice of carrier capacity on the resulting FCD. Common sense suggests that having fewer carriers with larger bit rates should result in fewer occasions to shut some down. However, smaller bit rates also imply smaller effective safety margins, defined as the predicted traffic, plus the safety margin, rounded up to the nearest multiple of the carrier granularity; this could have an adverse effect on the SMP, forcing the controller to increase the safety margin, thus nullifying or impairing the advantage of using smaller carriers.

This experiment was performed using the default parameters, save for the fact that carriers of 1, 10, 40 and 100 Gbit/s (which span likely values for both OFDM sub-carriers and legacy/current WDM transponders), were tried.

Fig. 4 depicts the measured FCD vs. the capacity of a single carrier. It is immediately clear that using fewer, larger carriers greatly reduces the amount of energy saved by all predictors; indeed, the FCD exhibited by the Hybrid predictor is close to 55% using 1 Gbit/s carriers but barely above 30% using 100 Gbit/s carriers, and the same pattern holds for the Profiling and Markovian predictors (although the former performs slightly worse using very fine grained carriers, and the latter performs consistently worse across all carrier capacities).

The performance degradation, in terms of FCD, appears to be roughly linear, with the exception of the 100 Gbit/s point, for which it is slightly worse. This can be explained considering the possible capacities of the lightpaths used in these experiments: when using 100 Gbit/s carriers, 25% of the LPs are served using a single carrier, and for these no savings are possible.

This presents an argument in favor of using modern OFDM flexible transponders, whose small sub-carriers can fully exploit the benefits of applying the proposed system. Scenarios employing non-OFDM transmission hardware can still benefit from the system, as long as super-channels are used.

Finally, to give an example of the amount of energy that can be saved with our approach, according to [9] a commercial 100 Gb/s transponder generating a single carrier draws 351 W; using either the profiling or hybrid predictor it would remain powered off about 32.3% of the time, resulting in a reduction in average consumption of up to 113.4 W per transponder. Please note that this is an optimistic figure, for two reasons: it does not factor in power drawn during power-up or power-down phases, and it assumes that 0 W are drawn when idle, while in practice this latter figure is likely to be higher.

C. Impact of the noise distribution and traffic auto-correlation

The purpose of this experiment is to verify the behavior of the proposed system when changing the statistical properties of the underlying traffic model. To this purpose we measure the FCD resulting from the use of a Gaussian noise with standard deviation of 0.1 times the average of the underlying traffic model for each specific TS, and uniformly distributed noise ranging between ± 0.3 times the same average. All these measures were repeated using multiple values of the auto-correlation parameter α, specifically 0, 0.2, 0.6 and 0.8. As before, all other parameters were set to their default values.

Fig. 5 shows only the resulting FCDs, since the control algorithm ensures that the resulting SMPs are only marginally different between the three predictors.

A first observation is that the ranking of the predictors is unaffected by the choice of noise model, whereas the level of short-term correlation has a large impact on their relative performance. More in detail, without short-term correlation the blind Profiling predictor achieves the best FCD. Two factors contribute to this performance: the absence of short term correlation implies that the average traffic pattern is constant, which is precisely the best possible setting for such a predictor. Furthermore, since the exponential averaging is

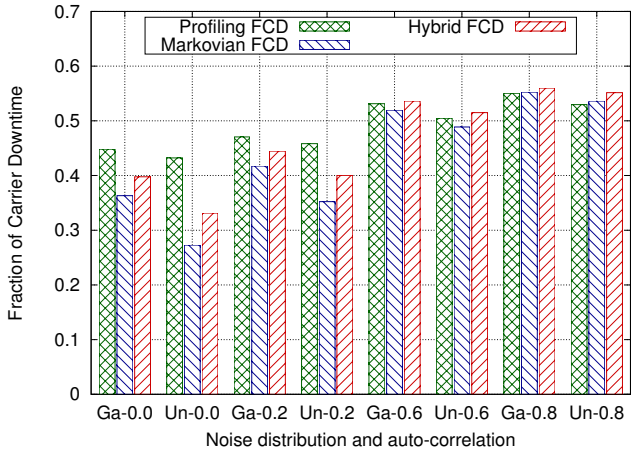


Fig. 5. Fraction of Carrier Downtime vs. Noise distribution (Gaussian or Uniform, average 0 and std. deviation 0.1 and 0.3 times the nominal pattern value, respectively) and traffic auto-correlation (exponential average α). The Profiling predictor works best with little or no short-term correlation, while the opposite is true for the Markovian and Hybrid predictors.

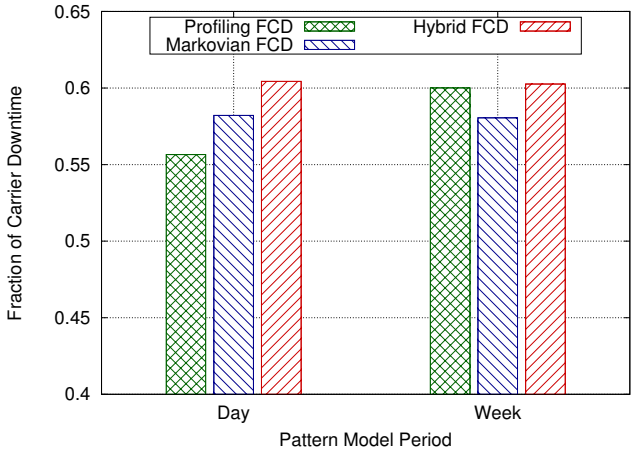


Fig. 6. Fraction of Carrier Downtime vs. period of the learned pattern using an underlying traffic model with lower load during week-ends. The performances of the Profiling predictor suffer when the learned pattern cannot accurately capture the dynamics of the underlying traffic, while the other predictors are less dependent on having an accurate model.

performed after adding the noise, this is the setting with the largest variance between consecutive TSs. In this situation, having a small safety margin on top of the historical maximum is more efficient than having a safety margin large enough to accommodate a possible shift from a very low point at TS_i to a very high point in TS_{i+1}.

As the correlation increases, such large differences between consecutive TSs are smoothed out by the exponential average, leading to improved performance from the Markovian and Hybrid predictors, which eventually outperform the Profiling predictor for very high correlation levels, with the latter always outperforming the former.

D. Impact of the learned profile period

Every experiment presented so far used 144 TSs to model the traffic offered to a LP in one day. Such profiles, however, cannot be expected to capture dynamics with periods longer than themselves. This experiment was devised to demonstrate the effect of using different values for the profile period T. It is reasonable to expect that a small T allows faster convergence, but may be unable to capture dynamics with longer periods, while a large T may be able to capture these dynamics, but would suffer from longer training an re-convergence times, since the data for each TS would be updated less frequently. To execute this experiment, we used the GARR pattern defined in section IV-A, which has a week-long period with a daily pattern with high peaks for 5 days and but much lower peaks in the remaining two, Ts of one day and one week, and all other parameter at their default values.

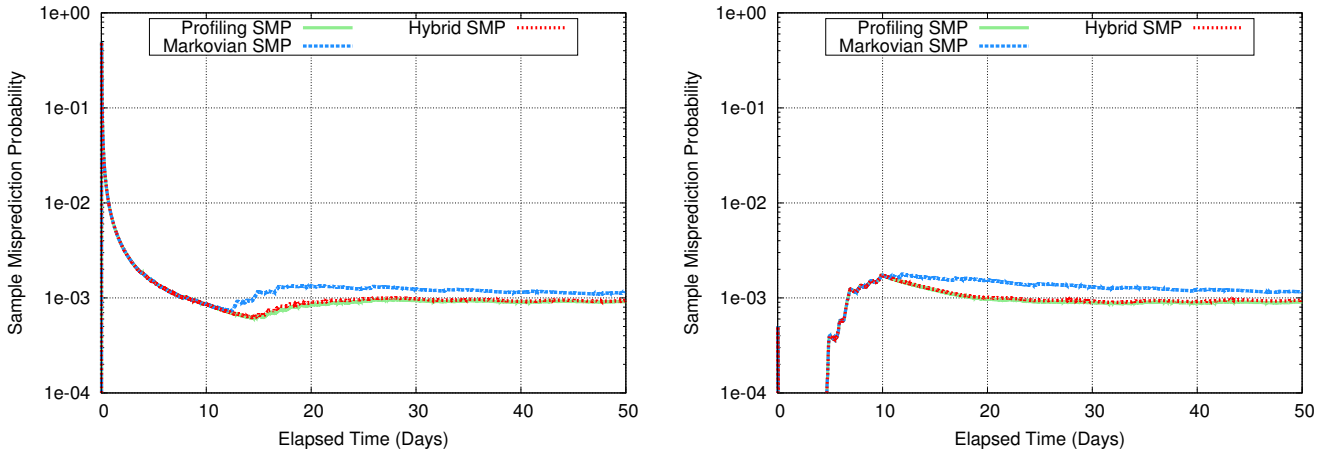
Fig. 6 displays the difference between using an internal model with a period of one day vs. one week, with all other parameters being assigned their default values. It is immediately clear from Fig. 6 that, as expected, the performances of the Profiling predictor strongly depend on having an accurate underlying profile. With an underlying profile period of just one day, it cannot predict the lower traffic experienced during weekends, therefore enabling significantly less savings than what it does by using a profile with a period of one week. The Markovian predictor does not make use of a traffic profile, and is therefore unaffected by the choice of period length. Finally, since the Hybrid predictor relies on learning a profile of just the differences between subsequent TSs, to differentiate between rising and falling phases, using a weekly model makes no detectable difference to the achieved FCD.

These results suggest that, if such a system were to be deployed using the Profiling predictor, it would be recommendable to produce two profiles with a daily period: one for workdays and the other for holidays, and then dynamically switch between the two in accordance with an external calendar specific to the geographic area spanned by the network in question. Should this not be possible, for example for networks spanning multiple nations and time zones, the Hybrid predictor provides very solid performances, even better than those of the Profiling one in the conditions of the experiment, without a strong dependence on the period of the underlying profile (a daily period is enough).

E. Impact of the initial safety margin

This experiment is designed to verify the behavior of the proposed system when the starting parameters lead to a SMP far from the target one. To test this, the main parameters were assigned their default values, except for the initial values of S, which were set to 0.05 and 0.2, to elicit a starting misprediction probability that is, respectively, significantly above and below the default target one.

Fig. 7 shows that, independently of the choice of initial parameters, the convergence of the SMP to the target misprediction probability can be considered almost complete after less than 50 days. In the first case, where the SMP starts much higher than the target, the safety mechanism enlarges the margin by a sufficiently large value as soon as the first loss is recorded, and soon afterwards revises the margin slightly to



(a) (b)

Fig. 7. Convergence of SMP for different starting safety margin values: 5% (a) and 20% (b) of the nominal capacity of each LP. The choice of initial parameter is largely ininfluential.

make it converge to the desired target. In the second case, the initial margin is so large that no losses are recorded, so it is progressively reduced until it starts causing excessive losses, after which the system behaves much like in the first case.

VI. Conclusions

We presented a system that enables an optical network to automatically learn the circadian patterns of the offered traffic, and then exploit them to predict incoming traffic and shut down unnecessary transmission devices to save energy. Such a system does not interact with the routing configuration of the optical network, and includes a self-configuration process that tweaks its internal parameters to achieve a desired probability of causing traffic losses.

Regardless of the simplicity of the control algorithm proposed in this work, it appears to offer good performance, in spite of having to control an highly non-linear metric, which is intrinsically difficult to measure. The algorithm is not affected by the choice of initial safety margin.

All proposed prediction algorithms appear to to be capable of significantly reducing the average uptime of transmission hardware, under all scenarios we considered. Their performance decreases approximately linearly as the capacity of a single carrier increases, and their relative performance is largely unaffected by the choice of traffic model noise; the only notable effect is that larger variances exacerbated the differences between them.

Overall, the Hybrid predictor appears to offer the best, most consistent performance; The Profiling predictor works very well when the circadian pattern of traffic exhibits low levels of noise, while the Markovian predictor is consistently the worst in terms of performance, but also the simplest to implement.

A. Potential extensions

There are many aspects of the proposed predictors that were not thoroughly investigated in this work, but may have a large influence on other applications. Possible applications of the proposed techniques in the framework of WDM and flexible optical networks include studying the effects of using:

- Short-lived, dynamic lightpaths, rather than permanent ones;
- Different durations of the initial training period of the Profiling and Hybrid predictors;
- Different durations of timeslots;
- Underlying patterns with a slowly increasing average, to simulate overall network traffic increases in the long term;
- More realistic underlying traffic models.

In addition to the energy saving system presented in this work, an optical network aware of the patterns of its traffic load could also be engineered to perform *spectral optimization*, minimizing the utilization of spectral resources by sharing part of the spectrum of neighboring LPs with complementary load profiles, or *bursts accommodation*, temporarily preempting unused spectral resources to accommodate sustained bursts in incoming traffic. Finally, an extensive comparison of the performance of our approach compared to others, such as those presented in [13] and [14], is another interesting potential work.

Acknowledgment

This research has been partially supported in CREATE-NET by the CHRON project, with funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 258644.

References

[1] J. Baliga, K. Hinton, and R. Tucker, "Energy consumption of the internet," in *Joint International Conference on Optical Internet, and the Australian Conference on Optical Fibre Technology, COIN-ACOFT*, 2007, pp. 1–3.

[2] C. Lange, D. Kosiankowski, C. Gerlach, F. Westphal, and A. Gladisch, "Energy consumption of telecommunication networks," in *35th European Conference on Optical Communication, ECOC*, 2009, pp. 1–2.

[3] J. Baliga, R. Ayre, K. Hinton, W. V. Sorin, and R. S. Tucker, "Energy consumption in optical ip networks," *J. Lightwave Technol.*, vol. 27, no. 13, pp. 2391–2403, Jul 2009.

[4] E. Williams, "Environmental effects of information and communications technologies," *Nature*, 2011.

[5] "AMS-IX public statistics," https://www.ams-ix.net/technical/statistics, accessed: 06/2013.

[6] "GARR public statistics," http://www.garr.it/a/rete/statistiche/46-backbone, data only available in italian. Accessed: 06/2013.

[7] D. Siracusa, F. Pederzolli, R. Lo Cigno, and E. Salvadori, "Energy saving through traffic profiling and prediction in self-optimizing optical networks," in *Optical Fiber Communications Conference and Exhibition (OFC)*, March 2014, pp. 1–3.

[8] M. Svaluto Moreolo, J. Fabrega, L. Nadal, F. Vilchez, and G. Junyent, "Bandwidth variable transponders based on ofdm technology for elastic optical networks," in *15th International Conference on Transparent Optical Networks (ICTON)*, June 2013, pp. 1–4.

[9] J. Lopez, Y. Ye, V. Lopez, F. Jimenez, R. Duque, P. Krummrich, F. Musumeci, M. Tornatore, and A. Pattavina, "Traffic and power-aware protection scheme in elastic optical networks," in *XVth International Telecommunications Network Strategy and Planning Symposium (NETWORKS)*, 2012, pp. 1–6.

[10] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future internet: A survey of existing approaches and trends in energy-aware fixed network infrastructures," *IEEE Communications Surveys Tutorials*, vol. 13, no. 2, pp. 223–244, 2011.

[11] A. Gencata and B. Mukherjee, "Virtual-topology adaptation for wdm mesh networks under dynamic traffic," *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 236–247, 2003.

[12] Y. Zhang, M. Tornatore, P. Chowdhury, and B. Mukherjee, "Energy optimization in ip-over-wdm networks," *Optical Switching and Networking*, vol. 8, no. 3, pp. 171 – 180, 2011, special Issue on Green Communications and Networking.

[13] L. Chiaraviglio, M. Mellia, and F. Neri, "Reducing power consumption in backbone networks," in *IEEE International Conference on Communications, ICC*, 2009, pp. 1–6.

[14] N. Fernandez, R. Duran, I. De Miguel, N. Merayo, D. Sanchez, M. Angelou, J. Aguado, P. Fernandez, T. Jimenez, R. Lorenzo, I. Tomkos, and E. Abril, "Cognition to design energetically efficient and impairment aware virtual topologies for optical networks," in *16th International Conference on Optical Network Design and Modeling (ONDM)*, 2012, pp. 1–6.

[15] K. Fukuda, L. Amaral, and H. Stanley, "Dynamics of temporal correlation in daily internet traffic," in *IEEE Global Telecommunications Conference, GLOBECOM*, vol. 7, 2003, pp. 4069–4073 vol.7.

[16] "CHRON European Project," http://www.ict-chron.eu/.

[17] E. Salvadori, R. Lo Cigno, and Z. Zsoka, "Dynamic grooming in ip over wdm networks: a study with realistic traffic based on gancles simulation package," in *Conference on Optical Network Design and Modeling*, 2005, pp. 185–196.

[18] E. Salvadori, R. Lo Cigno, and Z. Zsóka, "Dynamic grooming in IP over optical networks based on the overlay architecture," *Optical Switching and Networking*, vol. 3, no. 2, pp. 118 – 133, 2006.

[19] L. Muscariello, M. Mellia, M. Meo, M. Ajmone Marsan, and R. Lo Cigno, "Markov models of internet traffic and a new hierarchical MMPP model," *Computer Communications*, vol. 28, no. 16, pp. 1835 – 1851, 2005.

[20] D. Wang, "Traffic and routing on a weighted scale-free network model," in *25th Chinese Control and Decision Conference (CCDC)*, 2013, pp. 5058–5061.

[21] M. P, V. Gadre, and U. Desai, *Multifractal Based Network Traffic Modeling*. Kluwer Academic Publishers, 2003.

[22] F. Agraz, S. Azodolmolky, M. Angelou, J. Perello, L. Velasco, S. Spadaro, A. Francescon, C. Saradhi, Y. Pointurier, P. Kokkinos, E. Varvarigos, M. Gunkel, and I. Tomkos, "Experimental demonstration of centralized and distributed impairment-aware control plane schemes for dynamic transparent optical networks," in *Conference on Optical Fiber Communication (OFC), colocated National Fiber Optic Engineers Conference, (OFC/NFOEC)*, March 2010, pp. 1–3.

Federico Pederzolli (fpederzolli@create-net.org) received his MSc in computer science from the University of Trento, Italy, in 2013. His current research interests include network architectures, with an emphasis on optical networks and control planes, and green networking. He is currently a PhD student at the University of Trento.

Domenico Siracusa (dsiracusa@create-net.org) is a senior researcher of the Future Networks (FuN) group at CREATE-NET. He has received the PhD in Information Technology at Politecnico di Milano, in March 2012. He has been involved in different European projects on optical technologies and software defined networking. His current research interests include control plane solutions for converged future networks, application-centric resource orchestration, spatially-spectrally flexible optical networks, SDN and virtualization. He has authored more than 35 publications in international conferences and journals.

Elio Salvadori (elio.salvadori@create-net.org) is the Research Director at CREATE-NET research center. He graduated in Telecommunications Engineering (Laurea) at the Politecnico di Milano in 1997 and then worked as network planner and systems engineer in Nokia Networks and Lucent Technologies until November 2001, when he moved to the University of Trento to pursue his Ph.D. degree on Information and Communication Technologies. Within CREATE-NET he has been involved in several European projects on SDN, optical technologies, and Future Internet test-beds. He has published in more than 85 International refereed journals and conferences. He has been cofounder of the first workshop on software-defined networking in Europe in 2012 (EWSDN), while now is involved in its Steering Committee.

Renato Lo Cigno [SM] (locigno@disi.unitn.it) is Associate Professor at the Department of Computer Science and Telecommunications (DISI) of the University of Trento, Italy, where he leads the Advanced Network Systems research group in computer and communication networks. He received a degree in Electronic Engineering with a specialization in Telecommunications from Politecnico di Torino in 1988, the same institution where he worked until 2002. In 1998/9, he was with the CS Dep., UCLA, as a Visiting Scholar. Renato Lo Cigno has been General Chair of IEEE Int. Conf. on Peer-to-Peer Computing, and General Chair and TPC Chair of ACM WMASH and IEEE WONS in different years. He has served in many TPCs of IEEE and ACM conferences, including INFOCOM, GLOBECOM, ICC, MSWiM, VNC, and CoNext, and has been Area Editor for Computer Networks. His current research interests are in performance evaluation of optical and wireless networks, modeling and simulation techniques, congestion control, P2P networks and networked systems in general, with specific attention toward applications and sustainable solutions. Renato Lo Cigno is senior member of IEEE and ACM and has co-authored more than 150 papers in international, peer reviewed journals and conferences.