A Data-Driven Bandwidth Allocation Framework with QoS Considerations for EONs

Tania Panayiotou, *Member, IEEE,* Konstantinos Manousakis, *Member, IEEE,* Sotirios P. Chatzis, *Member, IEEE,* and Georgios Ellinas *Senior Member, IEEE.*

***Abstract*—This work proposes a data-driven bandwidth allocation (BA) framework for periodically and dynamically reconfiguring an elastic optical network according to predictive bandwidth allocation (PBA) models. The proposed framework is scalable to the number of network connections and also adaptive to the increasing traffic of each network connection separately and to the overall network load as well. This is achieved by formulating the BA problem as a Partially Observable Markov Decision Process (POMDP), which constitutes are reinforcement learning (RL) model. Specifically, RL is performed continuously and independently (locally) for each network connection according to the most recent data that describe the traffic demand behavior of each network connection. A central controller monitors the network performance that is jointly achieved from all the PBA models and is capable of dynamically modifying the reward function of the POMDP, ensuring that the quality-of-service requirements are met. A reward function $R(C)$ is examined with a clear impact on the network performance when C is modified. For evaluating the network performance, for each $R(C)$, the routing and spectrum allocation (RSA) problem is solved according to an Integer Linear Programming (ILP) algorithm and an RSA heuristic alternative, with both the ILP and the heuristic RSA taking as inputs the outputs of the inferred PBA models. Results indicate that with the appropriate settings of C, bandwidth is efficiently allocated, while ensuring that the QoS requirements are met.**

***Index Terms*—Reinforcement Learning, Traffic Prediction, Quality-of-Service, Dynamic Optical Networks, Network Reconfiguration, Routing and Spectrum Allocation.**

I. Introduction

With the emergence of new types of networks (e.g., Internet of Things (IoT), 5G networks, etc.) and network applications, the backbone traffic is exponentially growing [1]. In parallel, backbone traffic is becoming more dynamic [1], [2], with repetitive patterns [3]. As a consequence, emerging applications, require not only high-capacity optical links, but a dynamic backbone (optical) network where, unlike the traditional quasi-static/static networks, the network connections are now provisioned over a short period of time [4].

In static optical networks, connections tend to stay in the network for a long time period; for coping with the traffic demand variations and the current operational processes that are too slow to dynamically follow those variations [3], network connections in these networks are commonly overprovisioned. Specifically, connections are allocated a bandwidth that is usually based on the peak-hour demand, consequently causing sub-utilization of the allocated resources outside of the peak hour. As the average-to-peak-rate ratio continuously increases [1], dynamic or even programmable optical networks are becoming increasingly important for more efficiently utilizing the network resources without significantly increasing the operational and capital expenditures (Opex/Capex).

In dynamic optical networks, enabled by software defined networking (SDN) [4], network reconfigurations will happen more frequently, allowing for the evolving network to be robust and adaptive to the actual bandwidth demand variations. Ideally, network reconfigurations must closely follow the bandwidth demand variations. However, as the bandwidth demand may quickly fluctuate (fluctuations may occur within minutes), network reconfigurations that exactly follow the bandwidth demand fluctuations are not feasible. This is mainly due to the operational processes that are too slow for near real-time reoptimization of the network.

As a consequence, dynamically and periodically reconfiguring an optical network has recently gained attention from the research community [3], [4], [5], [6], [7], [8], [9], [10]. In general, a dynamic optical network is periodically reconfigured according to a priori estimated traffic demand matrices. These matrices provide an estimation about the future bandwidth demand for each network connection and for each time interval of interest, allowing the reconfigurations to be computed offline (i.e., during the previous time interval). This way, any required reconfigurations can take place at the beginning of each time interval. In general, the predictive traffic demand matrices can be inferred from historical traffic demand data, provided by continuously monitoring the aggregated traffic demand of the network connections (i.e., the IP layer (logical) connections).

A. Related Work

Related work is focused either on the implementation issues that are relevant to the dynamic optical networks framework and/or on how the predictive traffic matrices can be estimated and utilized for network reoptimization. Regarding the implementation issues, in general, SDN-based architectures have been proposed[4], [8], [9]. Regarding the predictive traffic matrices, statistical models [6], [7] or machine learning methods have been applied [3], [5], [7], [10], including neural networks (NNs), deep NNs (DNNs) [3], [5], [9], and reinforcement learning [10]. In most of these works the focus is on

T. Panayiotou, K. Manousakis, and G. Ellinas are with the KIOS Research and Innovation Center of Excellence, and with the Department of Electrical and Computer Engineering, University of Cyprus, Nicosia 1678, Cyprus, e-mail: {panayiotou.tania, manousakis.konstantinos, gellinas}@ucy.ac.cy

S. P. Chatzis is with the Department of Electrical Engineering, Computer Engineering, and Informatics, Cyprus University of Technology, e-mail: sotirios.chatzis@cut.ac.cy

predicting, from historical traffic data, the expected- or peak-rate for each network connection and for each time interval [3], [5], [6], [7], [9].

In [10] was, however, shown that a peak-rate estimation may still lead to connection overprovisioning, even within the short-time reconfiguration intervals assumed. To this end, a maximum probability- and an expected rate-based provisioning were also examined and were shown to lead to connection underprovisioning (less allocated bandwidth than the requested one, with the end-users experiencing low Quality-of-Service (QoS)). An alternative model, the predictive bandwidth allocation (PBA) model, was shown in [10] to be capable of more efficiently utilizing the network resources, while at the same time being adaptive to the ever-increasing and changing traffic demand patterns of each network connection. This was achieved by formulating the problem as a Partially Observable Markov Decision Process (POMDP), solved by applying RL [11]. Interestingly, this outcome was obtained by assuming traffic demand patterns with a similar behavior as the one describing the Internet traffic demand. Specifically, it was assumed that the traffic demand is described by the log-normal distribution that is characterized by skewness (asymmetric bandwidth demand) and long-tails (high expected- to peak-rate ratio) [1], [12], [13].

Further, in previous work [10], an optimal PBA model was found for each network connection by offline training all the PBA models in parallel and independently from each other. By doing so, the data-driven BA framework scales easily as the number of connections increase (i.e., the PBA models can be locally trained). On the other hand, however, the models cannot be aware of how all the models jointly perform, regarding the overall network performance, during the training procedure. Thus, the trained PBA models may not yield a set of PBA models that ensure that the QoS requirements are met, especially as the overall network load increases. Overall, the data-driven BA models must not only be adaptive to the traffic demand patterns of each independent connection, but also adaptive to the overall network load (jointly considering all network connections).

B. Our Contribution

This work significantly extends the work in [10], by proposing a data-driven bandwidth allocation (BA) framework for coping with the traffic demand variations. The proposed framework, apart from being scalable, it is also adaptive to the ever-increasing (overall) network load so that the QoS requirements of each individual connection are always met (to the extend that a network upgrade cannot be avoided). In this work, the data-driven BA problem is formulated as a state-of-the-art POMDP, which renders our solution a reinforcement learning algorithm. The reward function used during the RL process is now also dynamically modified, while the PBA models can still be independently (locally) trained to the most recent traffic data; this makes the framework scalable in terms of the number of network connections.

Specifically, this is done by letting a central controller periodically ascertain whether the QoS requirements are met by observing how the PBA models of each individual connection perform jointly. If the central controller observes that the QoS requirements are not met, the training procedure is dynamically modified accordingly through the reward function of the applied state-of-the-art RL approach. A reward optimization function, $R(C)$, is examined in this work, with a clear impact on the network performance when C is modified. For each $R(C)$, the outputs of the PBA models obtained, are used as inputs to the routing and spectrum allocation (RSA) algorithm used for network reoptimization. In this work, an ILP algorithm and a proposed RSA heuristic alternative that further considers for the spectrum fragmentation are also described and examined in conjunction with the proposed data-driven bandwidth allocation framework.

For training the PBA models we used synthetic traffic demand data generated by a set of log-normal distributions. Specifically, we assume that the traffic demand behavior is log-normally distributed as it has been experimentally shown that the Internet traffic can be described by this distribution [1], [12], [13]. Note, however, that the RL approach, used for training the PBA models, does not need to know a priori the distributions describing the traffic demand behavior. It can utilize real-time information for periodically/continuously inferring the underlying traffic demand distributions and accordingly adjusting the models upon significant variations on the traffic demand behavior. In practice, large amounts of traffic information can be easily collected by monitoring the traffic demand fluctuations within short time intervals. In this work, without loss of generality and in the absence of real traffic demand traces, the traffic demand traces are generated by sampling from a set of log-normal distributions. This assumption however, does not affect the problem formulation of the proposed framework or the main findings of this work. In fact, the proposed data-driven framework, as it will be shortly explained, it is robust to traffic demand changes and it is capable of adjusting the PBA models accordingly with the aim of achieving an acceptable network performance.

The rest of the paper is organized as follows: Section II provides an overview of the proposed data-driven BA framework, Section III formulates the PBA problem, and Section IV describes the RSA algorithms developed. Further, Section V is dedicated on the performance evaluation of the PBA models, while Section VI provides some concluding remarks.

II. Approach Overview

The proposed data-driven BA framework (illustrated in Fig. 1) assumes an elastic optical network (EON) based on a flexible grid and bandwidth variable transceivers (BVTs)/sliceable BVTs (SBVTs) [14] that can adapt to the actual traffic demands. When combined with optical transport platforms, EONs provide both high capacity links and a truly programmable and flexible networking environment capable of setting up and tearing down lightpaths within very short time frames (e.g., seconds or even milliseconds) [15].

On this basis, in this work we assume an IP-over-EON network that is centrally controlled by an SDN-based network controller equipped with monitoring, storage, and processing

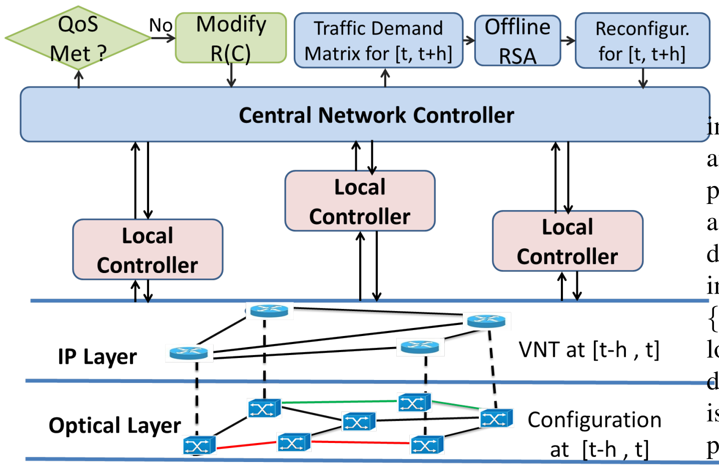

Fig. 1: Proposed data-driven BA framework.

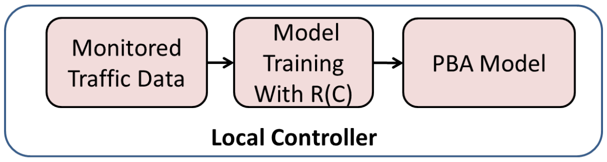

Fig. 2: Local controller functionalities

capabilities [4]. For scalability purposes, the central controller communicates, bidirectionally, with a set of local controllers, again equipped with monitoring, storage, and processing capabilities. The functionalities of the local controllers assumed are illustrated in Fig. 2. The aggregated IP traffic between the IP connections is continuously monitored (e.g., every five minutes or less) and the monitored data are stored into the local knowledge databases. We apply RL for inferring from these data a set of optimal PBA models, subsequently used for network reoptimization. The PBA models are trained locally, and network reoptimization is performed centrally (globally).

Specifically, the PBA models generate the predictive traffic demand matrices for each reconfiguration interval $[t, t+h]$, where h is a predefined period, $\{t\}_{t=1}^{\tau}$, and τ is the number of reconfigurations performed for example during a day. These matrices act as input to the RSA algorithm used for reoptimizing the network. The RSA is executed offline during $[t-h, t]$ and the actual network reconfiguration takes place at the beginning of $[t, t+h]$. In order for the PBA models to be adaptive to the changing traffic patterns of each connection independently and jointly as well, we assume that the network performance is continuously monitored. If the central controller observes that the QoS requirements are not met (i.e., network availability drops below an acceptable threshold), then the $R(C)$ function of the RL is modified to $R(C')$ and RL continues with $R(C')$. Note that for brevity, in the rest of the paper, a time interval $[t, t+h]$ will be denoted just by the starting time of the interval, t.

III. Predictive Bandwidth Allocation Models

Without loss of generality, we assume that traffic demand information is available for a 24-hour period ($h = 1$, $\tau = 24$) and for N source-destination pairs (IP layer connections). In particular, we assume that each connection is described by a set of traffic demand distributions, with each distribution describing the traffic demand fluctuations within a single time interval. The traffic demand fluctuations for each time interval $\{t\}_{t=1}^{\tau}$ and for each connection $\{n\}_{n=1}^{N}$ are described by the log-normal distribution $Z_{tn} \sim LN(\mu_{tn}, \sigma_{tn}^2)$. The log-normal distribution is utilized since, like the Internet traffic demand, it is characterized by skewness and heavy-tails (the average-to-peak ratio of Internet traffic continuously increases [1]). We assume that $z_{tn} \in (0, B)$ and that $B < B'$, where $z_{tn} \in Z_{tn}$, B is equal to the maximum emitting rate of the installed BVTs/SBVTs, and B' is equal to the total link capacity (i.e., B' spectrum slots are available for each network link). Note that in this work, each distribution is a function of the bandwidth demand that is measured according to the requested number of spectrum slots. Even though the requested number of spectrum slots depends on several factors, such as the transmission distance, the modulation format, and the quality-of-transmission requirements, in this work, for simplicity, and without loss of generality we assume that the distributions immediately reflect the requested number of spectrum slots; an assumption that does not affect the scope of this work.

For making the learning procedure of the PBA models computationally tractable, the distributions have been discretized according to specific rate intervals. Specifically, B has been divided into a intervals in such a way that the a^{th} interval is given by $B_a = [(a-1)k, ak]$, where $(a-1)k$ is the minimum rate of B_a, ak is the maximum rate of B_a, and $a = 1, 2, .., \frac{B}{k}$. Then for each time interval t, for each connection n, and for each B_a, the probabilities $p_{tn}^a = P[z_{tn} \in B_a]$ were evaluated, where p_{tn}^a is the probability of connection n requesting a number of spectrum slots between $(a-1)k$ and ak during time interval t. Since the traffic demand distributions are in this work randomly generated and may generate values larger than B, $p_{tn}^0 = P[z_{tn} > B]$ has also been evaluated to handle the distributions that generate rates above B. Finally, without loss of generality, it was assumed that $B_0 = 0$ with probability p_{tn}^0. Thus, a valid discrete probability distribution was generated.

For a network that is already configured and operating at t', a BA model indicates for each connection n the bandwidth allocation action a that must be taken for reconfiguring the network at the next time interval t. If the BA model indicates an action a for connection n, then the number of spectrum slots Δ_{tn} that must be allocated to connection n are given by $\Delta_{tn} = \max\{B_a\}$. Without loss of generality, we assume that Δ_{tn} includes the guard bands as well. Note that the actions are actually the indices to the B_a intervals, and thus, for simplicity, the same notation is used for both the actions and the indices of the rate intervals. We assume that a network reconfiguration takes place at the beginning of each time interval t and is computed offline during the previous time interval t'.

A. Formulating the PBA models

As pointed out, this stochastic BA problem is formulated as a POMDP. POMDPs generalize Markov Decision Processes (MDPs) that are usually used in search and planning heuristics for accommodating stochastic actions and full state observability [16]. POMDPs differ from MDPs in that the states are not observable but are estimated from observations. Formally, a POMDP is defined as a tuple $\{S, A, T, O, \Omega, b_0, R, \gamma\}$, where S is the set of states. A is the set of actions, $T(s'|s,a)$ defines the distribution over next state s' to which the agent may transition after taking action a while at state s, O is the set of observations, $\Omega(o|s,a)$ is a distribution over observations o that may occur as a result of taking action a and entering state s, $R(s,a)$ is the reward function that specifies the immediate reward for taking action a at state s, $\gamma \in [0,1)$ is the discount factor that weighs the importance of current and future rewards, and b_0 is the vector of the initial state distribution, such that $b_0(s)$ denotes the probability of starting at state s. Note that when $\gamma = 0$ the agent only cares about which action will yield the largest expected immediate reward and when γ approaches 1 the agent cares about maximizing the expected sum of future rewards.

In general, at each time step, the environment is at some state $s \in S$. The agent takes an action $a \in A$, and the environment transitions to state s' with probability distribution $T(s'|s,a)$. At the same time, the agent receives an observation $o \in O$ which is associated with the latent (unobservable) state s' according to some conditional likelihood function $\Omega(o|s',a)$. Finally, the agent receives a reward equal to $R(s,a)$. Then the process repeats. The goal is for the agent to choose actions at each time step t that maximize its expected future discounted reward $E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$.

In our BA problem, let us consider that the correlation between optimal network configuration and traffic demand patterns is not static, but may fluctuate on the grounds of longer term temporal dynamics. In that case, we must be capable of inferring these changes and adapting our policies accordingly. The essence of POMDPs addresses this consideration; POMDPs effect this goal by postulating that, at each time point, the modeled system has some latent state, s. Depending on the latent state, s, the same traffic demand requires a different policy of network reconfiguration, due to the different longer-term trends/dynamics that this latent state information encapsulates.

In our work, a POMDP is defined and solved independently for each connection n. On this basis, S, A, T, O, Ω, b_0 and R are now defined, for each connection n in the network, as follows:

• $S = \{s|s = 0, 1, ..., k\}$ with each state s representing the number of spectrum slots assigned to connection n.
• $A = \{a|a = 0, 1, ..., k\}$ with each action a representing the interval B_a, and hence the number of spectrum slots Δ_{*n} that must be allocated to n at time step $*$.
• $T(s'|s,a)$ defines the probability of transitioning to state s' if action a is taken at s. Note that $T(s'|s,a)$ is in this work deterministic, since during training, and for scalability purposes, we assume a network with infinite link capacity (the PBA models during RL cannot know how they will perform jointly in the network); thus we assume that a connection will transition with certainty at s' if a is taken at s.
• $O = \{o|o = 0, 1, ..., k\}$ with each observation o representing the interval B_o in which the requested (monitored traffic) rate belongs.
• $\Omega_n(o|s,a) = p^o_{tn}$ is the observation distribution of connection n. The observation distribution generates at each time step t the true bandwidth demand of n.
• b_0 is set to $b_0(s) = \frac{1}{k}$ $\forall s$ indicating that connection n can be initialized at any possible state s.
• $R(s,a)$ is the reward function that specifies the immediate reward for taking action a at state s. The immediate reward for each state-action pair depends on what the agent observes at s' after action a is taken at s (cannot be known a priori). On this basis, it is evaluated on the fly during the learning and exploration procedure of the POMDP (see Algorithm 1). For evaluating $R(s,a)$, we define an auxiliary reward function $r(s',a,o,C)$. Each element of $r(s',a,o,C)$ specifies the reward received when o is observed at s', after action a is taken at s. Specifically,

$$r(s',a,o,C) = \begin{cases} -C\exp(k), & \text{if } a < o \\ \exp(k-a+o), & \text{otherwise} \end{cases} \tag{1}$$

where C is a constant set by the central controller, aiming at controlling how all the PBA models perform jointly, with regards to the QoS requirements of the network. Note that C is kept constant during RL, only up to the moment that is dynamically modified by the controller, i.e., to C'. In such case, RL continues with C'.

Briefly, Eq. 1 specifies the reward received when o is observed at s', after action a is taken at s. It indicates that if the requested demand (o) is higher than the allocated bandwidth (a), then r will return a negative reward ($-C\exp(k)$), penalizing the action taken at s. If the requested demand (o) is lower than the allocated bandwidth (a) then a positive reward ($\exp[(k-o+a)]$) is received, indicating that the reward is increasing exponentially as the allocated bandwidth becomes closer to the requested one. By doing so, we aim at letting the agent learn how to operate near the "expected" bandwidth. At the same time, the agent "risks" receiving a penalty, since less bandwidth than the expected may be eventually requested. In essence, how far from the "expected" bandwidth the agent will learn to operate, depends on the C value.

For understanding the aforementioned agent's behavior, consider that the agent will never "risk" receiving a penalty, if it always decides to allocate the peak-rate of n. However, by doing so, the agent will rarely receive a very high positive reward, since the allocated bandwidth will often be well above the requested one (the average-to-peak ratio continuously increases and thus the event of actually observing the peak-rate becomes rare). If the agent, however, tends to allocate less bandwidth than the maximum possible, then it may "risk" receiving a negative reward - as more bandwidth may be eventually requested - but this will be done with the aim to acquire a higher *overall* reward.

According to the above, parameter C controls how much the agent will "risk" receiving a penalty. If $C >> 1$, then the agent will be guided to a PBA model that performs an allocation close to the peak-rate of n (never receiving a high penalty). By gradually decreasing C, the agent will tolerate more penalties. The basic idea is that the central controller, starting from a C value that meets the QoS requirements, can gradually decrease C, when it observes that the QoS requirements are not met (this event will eventually occur since the overall network load continuously increases). By doing so, the PBA models can be continuously trained to adapt to significant variations that may occur either on the traffic demand patterns of each connection separately and/or on the overall network load. Thus, the QoS requirements will be met, up to the point where a network upgrade is inevitable (e.g., by further deceasing C, connections' service level agreements (SLAs) are violated). Providing specific bounds for the C value is out of the scope of this work (they depend on several parameters, e.g., connections' SLAs), and is planned for future work. Further, optimizing C as part of the learning procedure is also planned for future work.

B. Training the PBA models

Commonly, POMDPs are solved by formulating them as completely observable MDPs over the *belief states* (the belief state is the posterior probability of the agent being at some state) of the agent [17]. Specifically, in POMDPs, the agent must choose its actions based only on past actions and observations, as the true state is not observable. Normally, the best action to take at time step t depends on the entire history of actions and observations that the agent has taken so far. However, the probability distribution over current states, known as the belief, is a sufficient statistic for a history of actions and observations [18]. In discrete state spaces, the belief of a state at step $t+1$ can be computed from the previous belief, b_t, the last action a, and observation o, by the following application of Bayes rule [18]

$$b^{a,o}_{t+1}(s) = \Omega(o|s,a) \sum_{s' \in S} T(s|s',a) b_t(s') / Pr(o|b,a), \quad (2)$$

where $Pr(o|b,a) = \sum_{s' \in S} \Omega(o|s',a) \sum_{s \in S} T(s'|s,a) b_t(s)$.

The POMDP model is trained by solving the Bellman equation (Eq. (3)) [18]. The Bellman equation computes the optimal value function for a given state, which expresses the expected discounted reward.

$$V^*_t(b) = \max_{a \in A} Q_t(b,a), \quad (3)$$

$$Q_t(b,a) = R(b,a) + \gamma \sum_{o \in O} Pr(o|b,a) V_t(b^{a,o}), \quad (4)$$

Specifically, in Eq. (3), the value function $V(b)$ is the expected discounted reward that an agent will receive if its current belief is b, $Q(b,a)$ is the value of taking action a at belief b, and $R(b,a)$ is the expected reward given by $\sum_{s \in S} R(s,a) b(s)$.

As the exact solution of the Bellman equation (Eq. (3)) is intractable for large spaces [19], in this work, we approximate the solution via the Real-Time Dynamic Programming-Bel (RTDP-Bel) [11] heuristic algorithm. In the RTDP-Bel a greedy policy π_V is used for finding an optimal policy, where $\pi_V(b) = \mathrm{argmax}_{a \in A} Q_t(b,a)$.

The RTDP-Bel is an asynchronous value iteration algorithm that converges to the optimal value function and policy over the relevant belief states without having to consider all the belief states in the problem. For achieving this, the RTDP-Bel uses an admissible heuristic function or lower bound h as the initial value function. Provided with such a lower bound, RTDP-Bel selects for update the belief over the states that are reachable from the initial state b_0 through the greedy policy π_V in a way that interleaves simulation and updates. For the implementation of the RTDP-Bel heuristic, the estimates $V(b)$ are stored in a hash table that initially contains only the heuristic value of the initial state, b_0. Then, when the value of a belief $b^{a,o}$ that is not in the table is needed, a new entry for $b^{a,o}$ with value $V(b^{a,o}) = h(b^{a,o})$ is allocated. These entries are updated following Eq. (3) when a move from s is performed. The RTDP-Bel algorithm is described analytically in [11].

Algorithm 1 Modified RTDP-Bel alg. for each connection n

```
1: Start with $b = b_0$.
2: Sample state $s$ from its probability distribution $b(s)$.
3: Evaluate each action $a$ at belief state $b$ as:
       $Q(b,a) = R(b,a) + \gamma \sum_{o \in O} Pr(o|b,a) V(b^{a,o})$,
   initializing $V(b^{a,o})$ to $h(b^{a,o})$ if $b^{a,o}$ is not in the hash.
4: Select action $a$ that maximizes $Q(b,a)$.
5: Update $V(b)$ to $Q(b,a)$.
6: Sample next state $s'$ from its probability distribution $T(s'|s,a)$.
7: Sample observation $o$ from its probability distribution $\Omega_n(o|s',a)$
8: Sample reward $r$ from the reward function $r(s',a,o)$
9: Set $R(s,a)$ equal to $r(s',a,o)$.
10: Compute $b^{a,o}$ using Eq. (2).
11: Finish (an episode is completed) if $b^{a,o}$ is target belief, else set $b := b^{a,o}$,
    $s := s'$, and go to 3.
```

In this work, the state-of-the-art RTDP-Bel algorithm is slightly modified to fit our problem formulation, incorporating the reward function defined in Eq. 1. The modified RTDP-Bel algorithm is described in Algorithm 1. Algorithm 1 is independently executed for each connection n in the network, and hence for each connection a different PBA model is evaluated. In Algorithm 1, an *episode* is defined as the sequence of actions and observations received for all time intervals $\{t\}_{t=0}^{\tau}$. According to Algorithm 1, in each time interval t a single observation is sampled from $\Omega_n(o|s,a)$. It is true, however, that within t a number of traffic demand fluctuations may occur. The algorithm will eventually obtain enough observations and will converge to an optimal PBA by iterating over a large number of episodes. Note that the assumption throughout is that the traffic demand is stationary for such a period of time that is capable of letting the RL to converge to a number of PBA models that reflect the network state (i.e., traffic patterns slowly change). In Algorithm 1, without loss of generality, the target belief is set at $t = 24$ (as we have assumed hourly network reconfigurations). By doing so, the assumption throughout is that each trained model encapsulates the daily behavior for each network connection.

Note that, depending on the problem setting, the target belief could be set to a different value (i.e., for training models encapsulating a total of τ network reconfigurations occurring i.e., every h time units).

IV. Routing and Spectrum Allocation

In EONs, for establishing a connection, the Routing and Spectrum Allocation (RSA) problem must be solved. The routing (R) problem deals with finding a route for a source and destination pair. The spectrum allocation (SA) problem deals with allocating spectral resources to the routing path (the spectrum slots are occupied symmetrically around the nominal central frequency of the channel). The allocated spectrum must meet the slot continuity and contiguity constraints, subject to the constraint of no frequency overlap [20]. A survey regarding the methods developed for the R problem can be found in [21], whereas regarding the SA problem, a number of SA policies have been developed that are in general categorized into fixed, semi-elastic, and elastic [21], [22].

In the fixed SA policies [22], [23] the allocated spectrum and the central frequency remain static for the entire lifetime of a connection. In the semi-elastic SA policies [22], [23] the central frequency remains static but the allocated spectrum width can be expanded/reduced according to the actual bandwidth demand. The main difference with the fixed SA policies is that the unutilized slots can now be used for subsequent connection requests providing higher flexibility and better resource utilization (i.e., unutilized slots can be used for low priority traffic). In the elastic SA policies [23]-[26] both the allocated central frequency and the spectrum width can change. The spectrum width can be expanded/reduced according to the actual bandwidth demand and the central frequency can be shifted [22], [24], [26], [27] (i.e., connections may be completely reallocated). The elastic SA policies offer better resource utilization but require the highest computational complexity and complex algorithms in the Path Computation Element (PCE) for minimizing traffic interruptions upon connection reallocation [21], [22].

In this work, the RSA algorithm is executed offline for each time interval $\{t\}_{t=1}^{\tau}$ with the objective to maximize the number of established connections $\{n\}_{n=1}^{N}$. For the SA procedure, the RSA takes as inputs the outputs of the PBA models (PBA models generate the predictive traffic demand matrix). In each time interval the connections are established without considering the establishment of the previous time intervals; that is, upon each network reconfiguration the network is elastically reconfigured, allowing for complete connection reallocation. Between network reconfigurations the connections can either follow a fixed SA policy or can be semi-elastically adjusted according to the actual bandwidth demand fluctuations. Thus, between network reconfigurations, traffic interruptions are avoided.

We now proceed to describe analytically both the ILP formulation and the heuristic approach developed for the offline RSA problem described above. Note that since the network is optimized offline, the computational time required for solving the ILP is not an issue, as long as it is less than the time between network reconfigurations. Alternatively, the heuristic approach can be utilized.

A. Integer Linear Programming Algorithm

The ILP formulation takes as input a set of κ candidate paths that are calculated for each source-destination pair by employing a κ-shortest path algorithm [28]. Its parameters, variables, and objective/constraints are shown below:

Parameters:

- $n \in N$: a requested connection
- $f \in F$: a spectrum slot over the available spectrum slots
- $d \in D$: a candidate path
- $l \in L$: a network link
- Δ_{tn}: number of requested slots for connection n at t
- D_n: set of candidate paths to serve connection n
- D: set of all candidate paths

Variables:

- x_{df}: Boolean variable, equal to 1 if path d and spectrum slot f are used to serve connection n and 0 otherwise.
- y_{df} Boolean variable, equal to 1 if spectrum slot f is the starting spectrum slot of a contiguous spectrum to serve demand n over path d and 0 otherwise.
- f_n: Boolean variable, equal to 1 if the connection n is established and 0 otherwise.

Objective: *Maximize* : $\sum\limits_{n} f_n$

Subject to the following constraints:

$$\sum_{d \in D_n} \sum_{f} x_{df} = \Delta_{tn} f_n, \forall n \in N \tag{5}$$

$$\sum_{d|l \in d} x_{df} \leq 1, \forall l \in L, \forall f \in F \tag{6}$$

$$x_{df} - x_{d(f-1)} \leq y_{df}, \forall d \in D, \forall f \in F \tag{7}$$

$$\text{Case: } f = 1, \text{then } x_{d(f-1)} = 0$$

$$\sum_{d \in D_n} \sum_{f} y_{df} \leq 1, \forall n \in N \tag{8}$$

The objective of the formulation is to maximize the number of established connections in the network. Constraint (5) ensures that the lightpaths chosen to serve a requested connection should satisfy the traffic demand. In case there are not enough spectrum slots, then this connection is blocked. Constraint (6) is the non-overlapping spectrum constraint and ensures that each spectrum slot is used at most once on each fiber. Constraint (7) ensures that each connection is assigned contiguous spectrum on all the fibers of each path. Constraint (8) is used in order to ensure that a connection uses only one BVT (or SBVT). The spectrum continuity constraint is taken into account by the definition of the x_{df} variable.

Algorithm 2 Pseudocode of the RSA Heuristic Alg.

```
1: Sort connection requests with their requested number of slots and start with the connection requesting the maximum number of slots.
2: Calculate for each connection $n$ a route using Dijkstra's alg.
3: Given the chosen route, calculate for each connection $n$ $B' - \Delta_{tn} + 1$ possible spectrum allocations that ensure the spectrum continuity and contiguity constraints (with $\Delta_{tn}$ being the bandwidth requested by connection $n$).
4: Among all the possible $B' - \Delta_{tn} + 1$ spectrum allocations, calculate the feasible lightpaths (those where the no frequency overlap constraint is also met). If no feasible path is found then the connection is blocked.
5: For all the feasible (candidate) lightpaths found, the fragmentation penalty is calculated, and the lightpath with the minimum penalty is chosen.
```

B. Heuristic Algorithm

The RSA heuristic, for each time interval t, finds a route and a spectrum allocation for each connection n. The description of the heuristic is shown in Alg. 2 below.

Specifically, the heuristic starts with the connection requesting the maximum number of slots Δ_{tn}. For the R problem, the Dijkstra's algorithm is used [29], while for the SA problem, the Δ_{tn} spectrum slots achieving the minimum *fragmentation penalty* (shortly defined), and meeting the spectrum continuity, contiguity, and no-frequency overlap constraints, are chosen for connection n. In particular, the RSA heuristic, calculates for each connection n the $B' - \Delta_{tn} + 1$ possible lightpaths that differ only on their allocated spectrum (note that if B' is the total link capacity and Δ_{tn} is the bandwidth requested by connection n, then there exist $B' - \Delta_{tn} + 1$ possible spectrum allocations that ensure the spectrum continuity and contiguity constraints, for a particular given route). Among all the possible $B' - \Delta_{tn} + 1$ spectrum allocations, the feasible lightpaths are only those where the no frequency overlap constraint is also met. For all the feasible (candidate) lightpaths, the fragmentation penalty is calculated, and the lightpath with the minimum penalty is chosen, where the fragmentation penalty, for each candidate lightpath, is just the sum of all the available slots around the spectrum slots occupied by the candidate lightpath, along the entire route of the candidate lightpath. Note that the proposed heuristic constitutes an alternative heuristic to the conventional κ-shortest path algorithm that is traditionally used for computing multiple candidate lightpaths for a single connection request. With the proposed approach, for each connection a single path is calculated; however, for each connection, it calculates several lightpaths. This is done by allocating to the same shortest-path all the possible combinations of contiguous and continues spectrum slots that meet the no-frequency overlap constraint (as indicated by the spectrum allocation action (PBA model) at each time interval). This technique is chosen as it can achieve performance results close to the ILP.

As an example, a candidate lightpath is shown in Fig. 3. The candidate lightpath, originating at node a and terminating at node e, is allocated the spectrum slots with indices $4-6$. According to Fig. 3, the fragmentation penalty is 7, since 3 slots are still available between the candidate lightpath and its adjacent connections at both links $a-b$ and $b-c$, 1 slot is available at link $c-d$, and 0 slots are available at link $d-e$.

Note that by choosing the candidate lightpath with the minimum fragmentation penalty we aim at packing as closely as possible the established connections, and consequently improving the network performance. This is a form of defragmentation that evolves as the network is continuously reconfiguring.

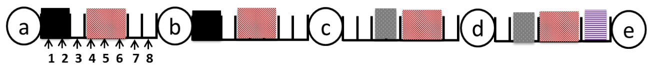

Fig. 3: Example for fragmentation penalty computation.

V. Performance Evaluation

To evaluate the proposed data-driven BA framework, the generic Deutsche Telekom (DT) network of Fig. 4 was used. Each spectral slot in the network was set at 12.5GHz with each fiber link utilizing $B' = 250$ C-band slots. Parameter B, representing the maximum possible rate of the installed BVTs/SBVTs, was set to $B = 100$ slots. Bandwidth B was divided into $k = 10$ rate intervals $\{B_a\}_{a=0}^{k}$. Hence, each PBA model can choose at each t and for each connection n amongst 11 spectrum allocation actions. Each action a indicates that $\Delta_{tn} = a \times k$ spectrum slots must be allocated at time interval t for connection n. In total, twenty-four time intervals were assumed ($\tau = 24$, $h = 1$), and 24 logical connections were considered.

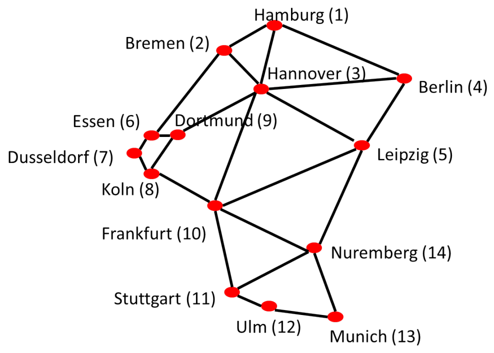

Fig. 4: Deutsche Telekom network

Similar to [10], the log-normal parameters, $(\mu_{tn}, \sigma_{tn}^2)$, for each connection n and for each time interval t were generated as follows: The σ^2 parameters were uniformly generated in the range $[0, 1]$ and the μ parameters were uniformly generated in the range $[0, 5]$. Note that the range of numbers from which the $(\mu_{tn}, \sigma_{tn}^2)$ parameters were generated, where chosen in such a way in order for the possible traffic demands to be analogous to the emitting capabilities of the BVTs/SBVTs. Indicatively, the traffic demand parameters for six stochastic connections are analytically given in Table I. A connection in Table I is denoted by $(v, v')_n$, where v is the source node of n and v' is the destination node of n. Note that for simplicity, and without loss of generality, we did not consider that the mean rate value (μ) between sequential (in time) traffic distributions increases/decreases smoothly. Such a consideration would not affect the learning procedure or the efficiency of the PBA models. Further, in order to compute the $p^a_{tn} = P[z_{tn} \in B_a]$ probabilities, the $Z_{tn} \sim LN(\mu_{tn}, \sigma^2_{tn})$ distributions were scaled down by a factor of two in order to better fit the emitting capabilities of the BVTs/SBVTs assumed. Specifically, the probabilities used in the simulations were calculated according to $p^a_{tn} = P[z_{tn} \in 2B_a]$.

TABLE I: Traffic Demand $Z_{tn} \sim LN(\mu_{tn}, \sigma^2_{tn})$

Connection (v,v')$_n$ / Time Interval t	(4,6)$_6$	(13,7)$_1$	(14,13)$_2$	(12,2)$_3$	(13,1)$_4$	(12,14)$_5$
1	(3.5,0.9)	(3.9,0.5)	(0.5,0.01)	(0.03,0.8)	(2.7,0.9)	(3.8,0.4)
2	(0.1,0.6)	(3.9,0.6)	(0.8,0.7)	(3.8,0.4)	(4.7,0.6)	(3.4,0.3)
3	(3,0.5)	(4.3,0.5)	(2.4,0.7)	(1.9,0.5)	(1.9, 0.5)	(1.2,0.2)
4	(5,0.5)	(0.2,0.8)	(3,0.6)	(5,0.4)	(4.3,0.3)	(2,0.5)
5	(0.1,0.2)	(1.8,0.6)	(3.6,0.6)	(4,0.4)	(4.9,0.3)	(1.5,0.9)
ine 6	(2.6,0.9)	(2.3,0.3)	(1.7,0.01)	(4.7,0.1)	(0.5,0.9)	(2,0.06)
7	(2.9,0.9)	(1,0.3)	(4,0.6)	(3.6,1)	(2.6,0.6)	(0.8,0.8)
8	(3.9,0.3)	(4.3,0.1)	(3.2,0.03)	(1.3,0.1)	(1.7,0.1)	(4,0.3)
9	(1.4,0.08)	(3.2,0.1)	(4.4,0.08)	(4.2,0.03)	(2.1,0.2)	(1.4,0.9)
10	(1.7,0.1)	(3.7,0.2)	(3,0.4)	(4,0.4)	(3.1,0.08)	(1.5,0.5)
11	(3.9,0.1)	(0.4,0.9)	(4.4,0.7)	(4.3,0.05)	(0.4,0.3)	(5,0.6)
12	(3.6,0.1)	(3.5,0.4)	(3.6,0.3)	(4.8,0.3)	(0.4,0.3)	(3.4,0.3)
13	(2.3,0.1)	(0.5,0.4)	(4.3,0.9)	(4.5,0.08)	(3.4,0.2)	(0.7,0.8)
14	(2.7,0.5)	(3.1,0.7)	(3.4,0.5)	(0.37,0.01)	(3,0.2)	(4.5,0,8)
15	(3.1,0.6)	(1.8,0.8)	(3.9,0.8)	(2,0.6)	(4.7,0.7)	(5,0.9)
16	(0.4,0.2)	(5,0.9)	(4.3,0.8)	(3.6,0.1)	(3.4,1)	(2.6,0.9)
17	(1.8,0.8)	(4.5,0.01)	(3.8,0.2)	(3,0.3)	(4.8,0.8)	(3,0.6)
18	(0.8,0.2)	(1.3,0.6)	(2.5,0.9)	(2.9,0.8)	(0.9,0.8)	(4,0.4)
19	(4,0.9)	(3.7,0.4)	(1.6,0.8)	(0.9,0.28)	(5,0.01)	(2.8,0.1)
20	(4.9,0.4)	(0.8,0.3)	(4.3,0.5)	(4.3,0.002)	(1.7,0.02)	(0.3,0.1)
21	(4.3,0.9)	(0.8,0.7)	(4.8,0.3)	(3.4,0.4)	(2.1,0.3)	(4.7,0.5)
22	(0.6,0.6)	(1.7,0.8)	(2,0.2)	(1.1,0.4)	(3.1,0.6)	(3,0.5)
23	(4.2,0.2)	(2.1,0.6)	(0.2,0.7)	(0.7,0.7)	(2.8,0.9)	(1.4,0.6)
24	(3.6,0.8)	(1.2,0.7)	(4.2,0.6)	(0.8,0.7)	(0.8,0.2)	(2.3,0.6)

For evaluating the impact of parameter C to the network performance, we assume that several sets of PBA models are already trained according to different C values of the $R(C)$ function (Eq. 1). Note that the training procedure of the PBA models is analytically described in Section V-B. As previously discussed, parameter C is a constant that can be dynamically modified by the central controller during the RL procedure. Specifically, C is modified when the central controller observes that the QoS requirements are no longer met; an event that will eventually occur as the overall network load increases. To this end, the RSA problem was solved for each set of PBA models. Note that a set of PBA models consists of all the PBA models (one model per connection) that were trained for the same C.

The RSA is solved according to both the ILP formulation and the proposed heuristic in a MATLAB machine with a CPU @2.60GHz and 8GB RAM. For solving the ILP, the Gurobi library was used [30]. Regarding the ILP, parameter κ was set to $\kappa = 3$ (note that larger κ values were also examined without however improving the network performance). The RSA heuristic required at most 1 minute for finding a feasible solution for each time interval, while the ILP required at most 10 minutes for finding a feasible solution for each time interval. An action was generated from each PBA model within milliseconds, indicating that the RSA can be solved in real time. Note that if the reconfiguration period (h) is not large enough for solving the RSA through the ILP, the RSA heuristic can be used instead. According to the results, discussed next, the RSA heuristic performs close to the ILP, indicating that the heuristic approach is indeed an efficient alternative.

A. Performance Results

Different sets of PBA models are trained for different C values (Section V.B) and the performance of each set of PBA models is assessed through a variety of metrics. Specifically, as in this work we are mainly interested on finding a set of PBA models that jointly achieve an acceptable network performance, the PBA models are evaluated through the RSA procedure according to: (i) the number of blocked connections, (ii) the number of unserved slots, and (iii) the number of unutilized slots. These metrics, as will be shortly explained, can be used for evaluating network availability, connection underprovisioning, and connection overprovisioning, respectively; consequently determining whether the achievable network performance, for each C value, is acceptable (or not).

Tables II, III, and IV summarize the results regarding the average number of blocked connections ($\bar{B}$) per day (or per episode), the average number of unutilized slots ($\bar{E}$) per hour (or per time interval), and the average number of unserved slots ($\bar{U}$) per hour (or per time interval), for both the ILP and the heuristic. Note that for solving the RSA problem, 100 episodes were generated for each set of PBA models, for both the ILP and the RSA heuristic and the results were averaged over these episodes.

Briefly, connection blocking may occur if a feasible route and spectrum allocation cannot be found during the RSA procedure. In our case, blocking means that the connection is down for the entire hour (or h period), consequently contributing to network unavailability. Unutilized/unserved bandwidth may occur between the network reconfigurations during the traffic demand fluctuations. In particular, unutilized/unserved bandwidth occurs if the allocated bandwidth is more/less than the requested one. Unserved bandwidth means that the end-users may receive less bandwidth than requested. As in this case the end-users will be partially served, the impact of unserved bandwidth is less severe than the impact of connection blocking.

TABLE II: Avg. Number of Blocked Connections ($\bar{B}$)

Heuristic RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
29	8	1	0	0	0

ILP RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
26	7.5	0	0	0	0

TABLE III: Avg. Number of Unutilized Slots ($\bar{E}$)

Heuristic RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
1402	1157	627	510	478	280

ILP RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
1400	1157	627	510	478	280

For evaluating $\bar{E}$ and $\bar{U}$, we assume that the traffic fluctuates every minute of the hour (60 fluctuations per t). Fluctuations were directly generated from the distributions used for model training. In particular, for the traffic demand fluctuations we

TABLE IV: Avg. Number of Unserved Slots ($\bar{U}$)

Heuristic RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
5.6	12	48	74.7	88.4	140

ILP RSA					
$C \geq 10$	$C = 5$	$C = 2$	$C = 1$	$C = 0.8$	$C = 0.5$
5.9	12	51.6	74.7	88.4	140

have drawn from each Z_{tn} the samples $\{z^i_{tn}\}_{i=1}^{60}$ representing the traffic demand fluctuations every minute of the hour. The sample $\delta^i_{tn} = \frac{z^i_{tn}}{2}$ denotes that connection n requests δ^i_{tn} spectrum slots at the i^{th} minute of time interval t (note that the sampled rates were divided by two for consistency with our simulation settings).

For each established connection, the allocated Δ_{tn} slots were compared to each δ^i_{tn} in order to calculate $\bar{E}$ and $\bar{U}$. The unserved slots for each episode are given by $U = \frac{1}{60 \times 24} \sum_t \sum_n \sum_i |\Delta_{tn} - \delta^i_{tn}|$, if $\quad \Delta_{tn} < \delta^i_{tn}$, while the unutilized slots for each episode are given by $E = \frac{1}{60 \times 24} \sum_t \sum_n \sum_i (\Delta_{tn} - \delta^i_{tn})$, if $\quad \Delta_{tn} > \delta^i_{tn}$. For finding $\bar{E}$ and $\bar{U}$, E and U were averaged over the 100 episodes (Tables III and IV). Note that $\bar{U}$ is calculated only according to the unserved bandwidth of the established connections (the unserved bandwidth of the blocked connections was not included in $\bar{U}$). This is the case since, even though connection unavailability (blocking) could be also translated to unserved spectrum, calculating the unserved spectrum during the unavailability period (i.e., according to the z^i_{tn} values) would not be an indicative metric; the behavior of the end-users inevitably changes during this period. Further, by not accounting for the unserved bandwidth of the blocked connections, we can better observe the impact of the C value on the network performance (i.e., if $\bar{U}$ is nearing zero, then we know that the network is operating near a peak-rate based BA model).

Table II shows that as C increases the average number of blocked connections also increases; a reasonable outcome if we consider that a higher C value is in essence a higher negative reward that during RL guides the agent to allocate a bandwidth that is closer to the peak-rate demand. This effect is clearly shown in Table III, indicating that as C increases, the agents tend to allocate an overall higher bandwidth that is most of the time unutilized (connection overprovisioning). On the other hand, as C decreases, the unserved bandwidth increases, leading to connection underprovisioning (Table IV). Clearly, an optimal C value depends on the QoS requirements of the network.

Regarding the ILP and the RSA heuristic results, Tables II, III, and IV show that the RSA heuristic performs close to the ILP algorithm in terms of connection blocking as well as unserved/unutilized bandwidth; an indicator that the RSA heuristic is an efficient alternative of the ILP. Note that the results of Tables III and IV are the same for both the ILP and the heuristic RSA when the number of blocked connections is zero (for $C \leq 1$ according to Table II). This is reasonable since in this case, the exact route and spectrum allocation of all the established connections does not affect $\bar{E}$ and $\bar{U}$; $\bar{E}$ and $\bar{U}$ are only affected by the BA actions generated by the models, that are the same for both the ILP and the RSA heuristic.

Overall, Tables II, III, and IV clearly demonstrate the impact of the reward optimization function (specifically the impact of parameter C) on the overall network performance. As C increases, the unutilized bandwidth also increases (Table III), since in that case the PBA models have a peak-rate BA tendency. For the network capacity considered, if $C > 1$ and the RSA heuristic is utilized, connection blocking occurs (Table II), consequently affecting network availability. Conversely, for the ILP approach, connection blocking occurs if $C > 2$. If we assume that according to the QoS requirements connection blocking should be zero, and RL currently operates with $C = 5$, then the controller should gradually decrease C to 2 (Table II) if the ILP is assumed (or to 1 if the RSA heuristic is utilized). This is done at the expense of increasing the unserved bandwidth (Table IV). Note however, that C does not need to be set below the higher C value that meets the QoS requirements (i.e., below 2 for the RSA ILP and below 1 for the RSA heuristic), since by doing so the unserved bandwidth (Table IV) will be unnecessary increased (QoS requirements are already met). In general, the C value should be decreased only up to the point where the QoS requirements are met. This however must be done without affecting the SLAs of each connection. If there is no such C value, then a network upgrade must take place. Note that letting C to be optimized as part of the learning process is planned for future work. Considering optimizing a different C value for each connection is also planned for future work.

B. Training the PBA Models

For training purposes, the discount factor γ was set to 0.95 (typical value for POMDP training). The C value of the reward function (Eq. 1) was assigned several values, as previously discussed (i.e., see Table II). A set of PBA models was trained for each C value (one model per connection). For each PBA model, RTDB-Bel was iterated over 6000 episodes of learning, which interleaved simulation and model updates (the model was updated after every 20 simulated episodes). After each model update, 200 test episodes were generated with the model fixed, for evaluating the model's efficiency. For each test episode, the model returned the total reward, the total allocated bandwidth, and the total number of negative rewards received. These values were averaged over all 200 episodes.

Training was terminated after the maximum number of episodes, or after the model converged. The last PBA model obtained was utilized for computing the network reconfigurations (the PBA outputs were used as inputs to the RSA algorithm) and evaluating the network performance (Section V-A). In general, up to 7 hours of training and testing was required for the 6000 episodes of learning in our MATLAB machine with a CPU @2.60GHz and 8GB RAM. An action was generated within milliseconds from each model; the negligible time required for action generation and the fact that each model (for each connection) can be trained in parallel and independently from each other renders the proposed data-driven BA framework scalable as the number of time-varying connections increases.

In general, the training procedure can be continuously performed for automatically adjusting the models upon variations on the traffic demand distributions; an important capability of the proposed method, given that the future traffic demand is expected to increase in uncertain ways (we cannot know the magnitude of a future traffic demand or the sources of this traffic). Moreover, as the RL is centrally controlled for ensuring that the QoS requirements are met, the PBA models can be jointly adjusted according to the ever-increasing overall network load through the $R(C)$ function (Eq. 1). For training purposes, we assume that the traffic demand is stationary for a period of time that allows the RL to converge to a set of PBA models that reflect the network state (i.e., we assume that traffic patterns change slowly). On this basis, the C value will not need to change before model convergence.

Figures 5-13 illustrate how the average reward, the average allocated bandwidth, and the average number of negative rewards evolve over the training time of the PBA model, when $C = 0.5$, $C = 1$, and $C = 10$. Training time is given in hours and corresponds to the time required for training and testing the model. A model update is indicated with a circle in Figs. 5-13 (approximately 250 total model updates for the 6000 episodes). Figures 5, 8, and 11 correspond to the training procedure of the PBA model for $n = 1$ and $C = 0.5$, Figs. 6, 9, and 12 correspond to the training procedure of the PBA model for $n = 1$ and $C = 1$, while Figs. 7, 10, and 13 correspond to the training procedure of the PBA model for $n = 1$ and $C = 10$. Note that similar figures were obtained for all the other connections and C values but are omitted due to space limitations.

Overall, Figs. 5, 6 and 7 show that the PBA models perform better as the training procedure evolves. The average reward increases with the number of model updates (training time) as the agent learns to take better bandwidth allocation decisions. For the different C values, Figs. 5, 6 and 7 demonstrate that as C increases the average reward decreases; an expected outcome since as C increases the penalty (negative reward) also increases, resulting to an achievable lower average cumulative reward.

Consequently, for the higher C values, the PBA models tend to allocate a higher number of spectrum slots (Figs. 8, 9, and 10), aiming at decreasing the average number of negative rewards (penalty) received (Figs. 11, 12, and 13). This outcome is aligned with the results presented in Section V-A where it is shown that as C increases, the unutilized bandwidth also increases, resulting to an increased number of blocked connections. Overall as C increases the PBA models tend to follow a model that resembles a peak-rate BA model, that consequently leads to connection overprovisioning. On the other hand, as C decreases the PBA models tend to lead to connection underprovisioning. An optimal C value will be the one that meets the QoS requirements. As pointed out, this value can be found by gradually decreasing C upon each violation of the QoS requirements.

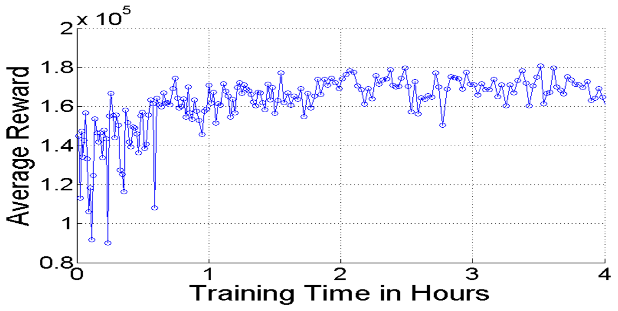

Fig. 5: Avg. reward over training time ($C = 0.5$).

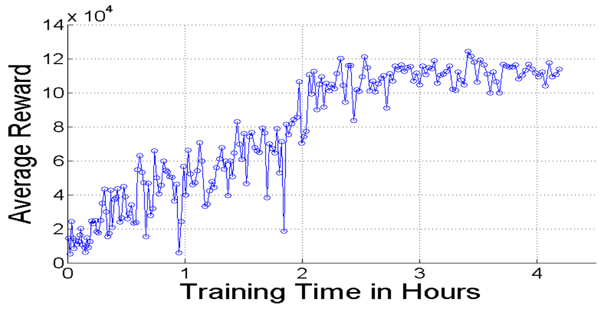

Fig. 6: Avg. reward over training time ($C = 1$).

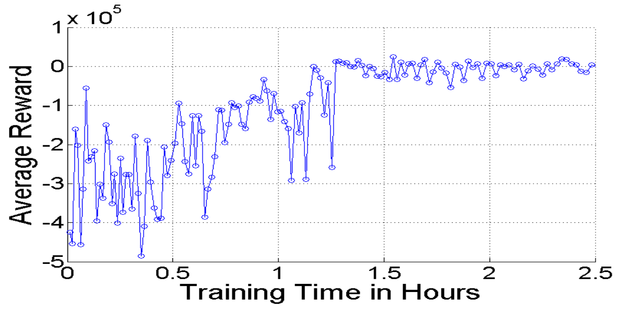

Fig. 7: Avg. reward over training time ($C = 10$).

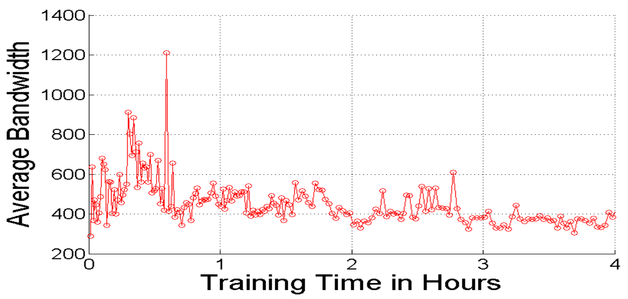

Fig. 8: Avg. allocated bandwidth over training time ($C = 0.5$).

VI. Conclusion

This work proposed a data-driven BA framework for elastic optical networks that takes into account the network's QoS requirements. The data-driven BA problem is formulated as a state-of-the-art POMDP, solved by applying reinforcement learning. The proposed framework is scalable as it assumes

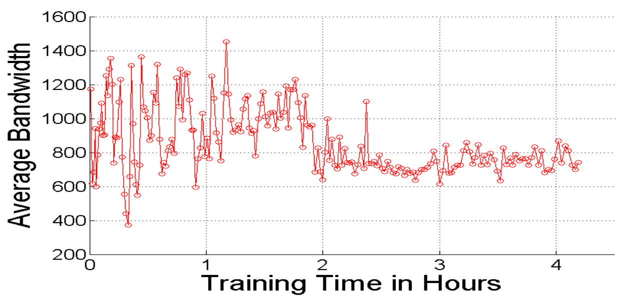

Fig. 9: Avg. allocated bandwidth over training time ($C = 1$).

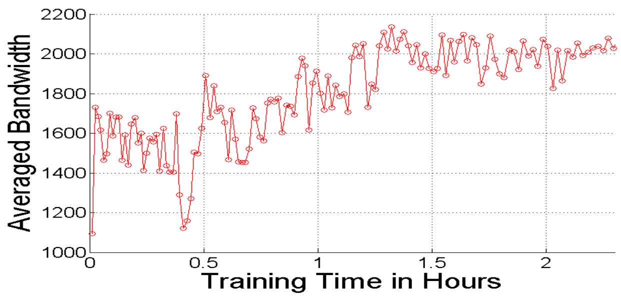

Fig. 10: Avg. allocated bandwidth over training time ($C = 10$).

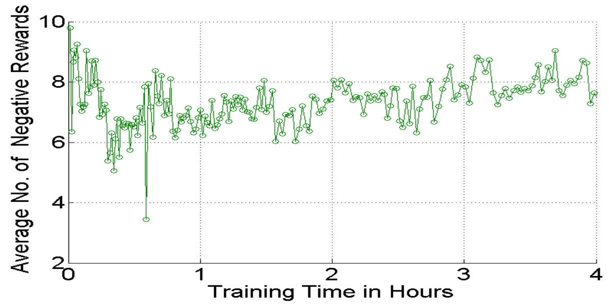

Fig. 11: Avg. no. of negative rewards over training time ($C = 0.5$).

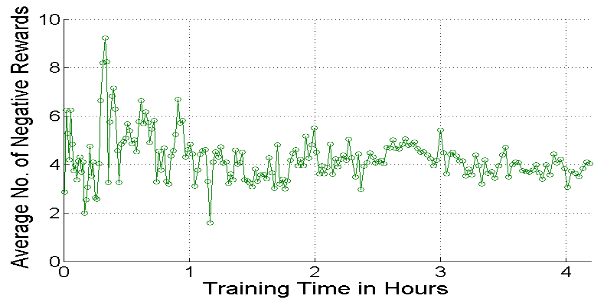

Fig. 12: Avg. no. of negative rewards over training time ($C = 1$).

that a PBA model is inferred for each network connection by independently (locally) training each connection's model. The framework is also adaptive to the time-varying traffic patterns of each connection separately and to the overall network load as well by dynamically modifying the reward function of the state-of-the-art RL approach applied. For

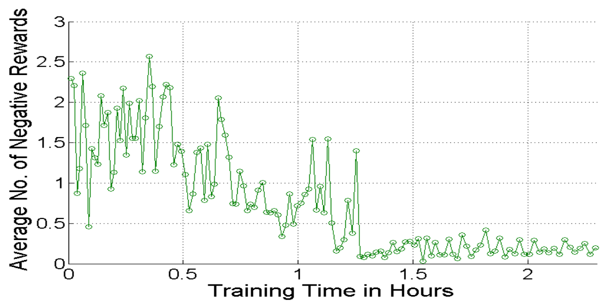

Fig. 13: Avg. no. of negative rewards over training time ($C = 10$).

network performance evaluation, the RSA is solved according to an ILP formulation and a proposed heuristic alternative that can be used when the ILP cannot be solved in practical time. Results indicate that by appropriately modifying the reward function, the network bandwidth is efficiently utilized, while also operating within the QoS requirements. Future work includes the optimization of the reward function as part of the RL process in order to further increase network efficiency. Experimentally demonstrating the effectiveness of the proposed approach according to real traffic demand data along with examining any related implementation issues also constitute interesting future research directions.

Acknowledgment

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551 (KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

References

[1] Cisco white paper, "The Zettabyte Era: Trends and Analysis," 2017.
[2] Cisco white paper, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020," 2016.
[3] R. Alvizu, et al., "Matheuristic with Machine-learning-based Prediction for Software-defined Mobile Metro-core Networks," *IEEE/OSA Journal of Optical Communications and Networking*, 9(9):D19–D30, Sept. 2017.
[4] S. Yan, et al., "Data-driven Network Analytics and Network Optimisation in SDN-based Programmable Optical Networks," *Proc. ONDM*, Dublin, Ireland, May 2018.
[5] F. Morales, et al., "Virtual Network Topology Adaptability based on Data Analytics for Traffic Prediction," *IEEE/OSA Journal of Optical Communications and Networking*, 9(1):A35–A45, Jan. 2017.
[6] N. Fernández et al., "Virtual Topology Reconfiguration in Optical Networks by Means of Cognition: Evaluation and Experimental Validation," *IEEE/OSA Journal of Optical Communications and Networking*, 7(1):A162–A173, Jan. 2015.
[7] Y. Ohsita et al., "Gradually Reconfiguring Virtual Network Topologies based on Estimated Traffic Matrices," *Proc. IEEE INFOCOM*, Anchorage, AK, 2007.
[8] A. Caballero et al., "Cognitive, Heterogeneous and Reconfigurable Optical Networks: The CHRON Project," *IEEE/OSA Journal of Lightwave Technology*, 32(13):2308–2323, July 2014.
[9] X. Chen et al., "Leveraging Deep Learning to Achieve Knowledge-based Autonomous Service Provisioning in Broker-based Multi-Domain SD-EONs with Proactive and Intelligent Predictions of Multi-Domain Traffic," *Proc. ECOC*, Gothenburg, Sweden, 2017.
[10] T. Panayiotou, et al., "On Learning Bandwidth Allocation Models for Time-Varying Traffic in Flexible Optical Networks," *Proc. ONDM*, Dublin, Ireland, May 2018.

[11] B. Bonet and H. Geffner, "Solving POMDPs: RTDP-Bel vs. Point-based Algorithms," *Proc. IJCAI*, 2009.
[12] I. Antoniou, et al., "On the Log-normal Distribution of Network Traffic," *Physica D: Nonlinear Phenomena*, 167(1–2):72–85, 2002.
[13] M. Kassim, et al., "Statistical Analysis and Modeling of Internet Traffic IP- Based Network for Tele-traffic Engineering," *ARPN J. of Eng. and Applied Sciences*, 10(3):1505–1512, 2015.
[14] R. Martinez et al., "Control Plane Solutions for Sliceable Bandwidth Transceiver Configuration in Flexi-grid Optical Networks," *IEEE Communications Magazine*, 54(8):126–135, Aug. 2016.
[15] "Architecture for the Automatically Switched Optical Network," *ITU-T Recommendation G.8080*, 2012.
[16] D. Bertsekas, *Dynamic Programming and Optimal Control, (2Vols)*, Athena Scient., 1995.
[17] E. Sondik, "The Optimal Control of Partially Observable Markov Decision Processes over the Infinite Horizon: Discounted Costs", *Oper. Res.*, 26(2):282–304, 1978.
[18] F.D.-Velez, "The Infinite Partially Observable Markov Decision Process," *Advances in Neural Information Proc. Systems 22*, 2009.
[19] C. Papadimitriou and J. Tsisiklis, "The Complexity of Markov Decision Processes," *Math. of Operations Research*, 12(3):441–450, 1987.
[20] K. Christodoulopoulos, et al., "Elastic Bandwidth Allocation in Flexible OFDM-Based Optical Networks," *IEEE/OSA J. Lightwv. Techn.* 29(9):1354–1366, 2011.
[21] B. C. Chatterjee, et al., "Routing and Spectrum Allocation in Elastic Optical Networks: A Tutorial," *IEEE Comm. Surveys & Tutorials*, 17(3):1776–1800, 2015.
[22] M. Klinkowski, et al., "Elastic Spectrum Allocation for Time-Varying Traffic in FlexGrid Optical Networks," *IEEE J. on Selected Areas in Comm.*, 31(1):26–38, 2013.
[23] K. Christodoulopoulos, et al., "Time-Varying Spectrum Allocation Policies and Blocking Analysis in Flexible Optical Networks," *IEEE J. on Selected Areas in Comm.*, 31(1):13–25, 2013.
[24] S. Shakya, et al., "Spectrum Allocation for Time-varying Traffic in Elastic Optical Networks using Traffic Pattern," *Proc. OFC*, 2014.
[25] G. Shen, et al., "Maximizing Time-dependent Spectrum Sharing between Neighbouring Channels in CO-OFDM Optical Networks," *Proc. ICTON*, 2011.
[26] K. Christodoulopoulos, et al., "Dynamic Bandwidth Allocation in Flexible OFDM-based Networks," *Proc. OFC*, 2011.
[27] F. Cugini, et al., "Push-Pull Defragmentation Without Traffic Disruption in Flexible Grid Optical Networks," *IEEE/OSA J. Lightwv. Techn.*, 31(1):125–133, 2013.
[28] J.Y. Yen, "Finding the k Shortest Loopless Paths in a Network," *Management Science*, 17(11):712–716, 1971.
[29] T. H. Cormen, et al., *Introduction to Algorithms (Third ed.), Section 24.3: Dijkstra's algorithm*, MIT Press, 2009.
[30] Gurobi Optimization Inc., "Gurobi Optimizer Reference Manual," 2016, http://www.gurobi.com.

Tania Panayiotou received her Diploma degree in Computer Engineering and Informatics, from the University of Patras, Greece, in 2005, and her Ph.D degree in Computer Engineering, from the University of Cyprus in 2013. She is currently a Research Fellow at the KIOS Research and Innovation Center of Excellence at the University of Cyprus. She has previously worked as an Associate Researcher at the Department of Electrical Engineering, Computer Engineering and Informatics of the Cyprus University of Technology, and as a teaching fellow in the Department of Electrical and Computer Engineering at the University of Cyprus and in the Department of Information and Communication Systems at the Open University of Cyprus. Her research interests focus on optical networks as well as transportation networks.

Konstantinos Manousakis received the Diploma, M.Sc. and Ph.D. degrees, all in Computer Engineering and Informatics, from the University of Patras, Greece, in 2004, 2007, and 2011, respectively. He is a Research Fellow at the KIOS Center of Excellence (CoE), University of Cyprus, Nicosia, Cyprus. Since 2014, he is a Marie Curie (MC) Fellow, working on a four-years MC – Career Integration Grant (CIG) in the area of Optical Network Security. His research interests are in the area of optimization algorithms and security in optical networks.

Sotirios Chatzis received the M.Eng. (Hons.) degree in electrical and computer engineering and the Ph.D. degree in machine learning from the National Technical University of Athens, Athens, Greece, in 2005 and 2008, respectively. He was a Post-Doctoral Fellow with the University of Miami, Coral Gables, FL, USA, from 2008 to 2010. He was a Post-Doctoral Researcher with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., from 2010 to 2012. He is currently an Assistant Professor with the Department of Electrical Engineering, Computer Engineering and Informatics, Cyprus University of Technology, Limassol, Cyprus. He has authored more than 60 papers in the most prestigious journals and conferences of the research field. His current research interests include machine learning theory and methodologies, specifically hierarchical Bayesian models, Bayesian nonparametrics, and deep hierarchical feature extractors, with a focus on modeling data with temporal dynamics. His Ph.D. research was supported by the Bodossaki Foundation, Greece, and the Greek Ministry for Economic Development. Dr. Chatzis was a recipient of the Dean's scholarship for Ph.D. studies, being the best performing Ph.D. student of the class.

Georgios Ellinas holds B.Sc., M.Sc., M.Phil., and Ph.D. degrees in Electrical Engineering from Columbia University. He is currently a Professor and the Chair of the Department of Electrical and Computer Engineering at the University of Cyprus. Previously, he was an Associate Professor of Electrical Engineering at City College of New York. Before joining the academia, he was a Senior Network Architect at Tellium Inc. George also served as a Visiting Scientist/Research Scientist in Bellcore's Optical Networking Research Group. His research interests focus on optical networks, intelligent transportation systems, critical infrastructure systems, and IoT.