On Deep Reinforcement Learning for Static Routing and Wavelength Assignment

Nicola Di Cicco, Emre Furkan Mercan, Oleg Karandin, Omran Ayoub,
Sebastian Troia, Francesco Musumeci, and Massimo Tornatore

(Invited Paper)

***Abstract*—Deep Reinforcement Learning (DRL) is rising as a promising tool for solving optimization problems in optical networks. Though studies employing DRL for solving static optimization problems in optical networks are appearing, assessing strengths and weaknesses of DRL with respect to state-of-the-art solution methods is still an open research question. In this work, we focus on Routing and Wavelength Assignment (RWA), a well-studied problem for which fast and scalable algorithms leading to better optimality gaps are always sought for. We develop two different DRL-based methods to assess the impact of different design choices on DRL performance. In addition, we propose a Multi-Start approach that can improve the average DRL performance, and we engineer a shaped reward that allows efficient learning in networks with high link capacities. With Multi-Start, DRL gets competitive results with respect to a state-of-the-art Genetic Algorithm with significant savings in computational times. Moreover, we assess the generalization capabilities of DRL to traffic matrices unseen during training, in terms of total connection requests and traffic distribution, showing that DRL can generalize on small to moderate deviations with respect to the training traffic matrices. Finally, we assess DRL scalability with respect to topology size and link capacity.**

***Index Terms*—Deep Reinforcement Learning, Routing and Wavelength Assignment, Genetic Algorithm, Optimization.**

I. Introduction

In recent years, several applications of Machine Learning (ML) in optical networks have been developed, such as Quality of Transmission (QoT) estimation [1], failure management [2] and traffic prediction [3], demonstrating the ability of ML models to extract useful information from the monitoring data available in optical networks.

Among the various tools offered by ML, Deep Reinforcement Learning (DRL) is attracting particular attention as a promising tool for tackling complex optimization problems for resource allocation in optical networks [4]. However, even though DRL has shown remarkable results in tasks such as playing games [5]–[7] and continuous control [8], [9], investigations on DRL for solving optimization problems in optical networks have started only in recent years [10]–[13]. What makes DRL competitive with respect to both supervised and unsupervised learning is its capability to learn directly from experience with little to no prior data, while driving optimization towards a user-specified goal. These characteristics make DRL a particularly attractive solution for scenarios in which it is difficult, if not impossible, to acquire well-defined training data.

Nevertheless, there are many potential difficulties in getting the best performance out of DRL and in interpreting the obtained results, ranging from hyperparameter tuning to specific algorithm implementations [14]. Furthermore, by definition, a DRL-based solution method is trained on a limited subset of problem instances, such as a subset of all the possible traffic matrices in an optical network. Even though DRL can show some generalization capabilities with respect to problem instances not seen during training, assessing them a-priori is not a trivial task [15], [16]. In realistic application scenarios, if the problem instances differ too much with respect to the ones used in training, DRL may suffer from severe performance hits, to the point of requiring a new training phase starting from scratch. This introduces a significant computational burden, as training can take hours to days on modern hardware platforms.

Given these significant challenges in terms of generalization, in this study our aim is to evaluate whether DRL for optimization problems in optical networks can be a competitive alternative with respect to existing and well-established solution methods. To this end, we focus on the Routing and Wavelength Assignment (RWA) problem, a well studied problem in the optical-networking literature for which, even though there are many effective solution methods [17]–[19], more efficient algorithms are always of interest. Therefore, the RWA problem will be considered as a representative case study on which to train new DRL-based solution methods and evaluate DRL performance with respect to existing methods. In fact, we note here that comparing DRL only against greedy heuristics (as mostly done in literature) can result in an overly optimistic assessment of its actual performance. Hence, in this study, the proposed DRL methods will be compared not only to greedy heuristics, but also to a state-of-the-art metaheuristic (a Genetic Algorithm, whose performance greatly outperforms greedy heuristics), and with Integer Linear Programming (ILP), which is used to establish an optimality bound.

We also assess the sensitivity of DRL to differences between the traffic matrices seen during and after training. In fact, traffic matrices used at training time are typically drawn from a probability distribution which is assumed to be known a-priori (e.g., from traffic forecasts), but may be different in practical use-cases. We aim to evaluate whether or not it is convenient to employ DRL in place of other state-of-the-art

N. Di Cicco, E. Furkan Mercan, S. Troia, F. Musumeci and M. Tornatore are with the Department of Electronics, Information and Bioengineering (DEIB), Politecnico Di Milano, Italy. E-mail: {name}.{surname}@polimi.it

O. Ayoub is with the University of Applied Sciences of Southern Switzerland, Switzerland. E-mail: omran.ayoub@supsi.ch

solution methods, when traffic matrices differ with respect to the training ones (e.g., because of a wrong forecast) in terms of total connection requests and traffic distribution. Moreover, we assess the scalability of our proposed approach with respect to topology size and link capacity.

The remainder of this paper is organized as follows: in Section II we provide an overview of the recent works on DRL for optimization in optical network, and and highlight some of the main advancements on this topic in the literature. In Section III we discuss background concepts on DRL along with the main elements of state-of-the-art DRL algorithms, with particular focus on the Proximal Policy Optimization (PPO) algorithm. In Section IV we describe the RWA problem and the considered baselines (i.e., greedy heuristics, Genetic Algorithm and ILP). In Section V we illustrate the proposed DRL approach for the RWA problem. In Section VI we benchmark the performance, in terms of blocking probability and computing times, of the proposed DRL approach against the baselines, we assess its ability to generalize to traffic matrices not seen during training and its scalability to larger problem instances, in terms of network size and link capacities. Finally, we outline conclusions and future research directions in Section VII.

II. Related Work

The application of DRL algorithms in telecommunications is a fertile area of research that is giving rise to many promising results [20]. However, works applying DRL to problems in optical networks are relatively few. In the following, some of the main works regarding the application of DRL in telecommunications networks, together with their most significant achievements, are briefly summarized.

In [10], an application of DRL to routing problems in Software Defined Networking (SDN) is developed. The proposed DRL-based algorithm is able to dynamically produce routing configurations minimizing the overall delay. The proposed algorithm is an is an off-policy, actor-critic, deterministic policy gradient algorithm, for which an observation is defined as the traffic matrix.

In [21] authors propose a DRL-based approach for routing in optical networks. The DRL agent is tasked to route connection requests among candidate paths. A novel state representation is engineered, consisting of the states resulting from applying all possible actions to the current state. The proposed approach is able to outperform both baseline heuristics and other DRL agents using different state representations.

In [11] authors address routing in Optical Transport Networks (OTNs), with emphasis on generalization to topologies similar to the one seen at training time. Authors employ Deep Q-Learning [5], in which the DNN is implemented as a Graph Neural Network (GNN) [22]. The objective is the maximization of bandwidth utilization up to the first blocked request, meaning that long-term minimization of the blocking rate, as in our work, is not taken into account. The proposed approach outperforms vanilla DQN and greedy heuristics also for network topologies not seen during training.

In [12] authors propose a custom version of the Asynchronous Advantage Actor Critic (A3C) [23] algorithm for dynamic Routing, Modulation and Spectrum Assignment (RMSA) in Elastical Optical Networks (EONs). After precomputing a number of candidate paths using K-shortest-paths, the action space is defined in such a way that the agent is able to pick one among the first j available spectrum blocks in each of the candidate paths, where j is an algorithm parameter. An observation is defined such that it provides information about the current request and the spectrum utilization in each of the candidate paths. The algorithm is shown to outperform baseline greedy heuristics such as SP-FF and KSP-FF.

In [13] authors, to address the scalability issues of the algorithm developed in [12], propose a cooperative multi-agent framework for service provisioning in inter-domain EONs, in which agents are based on the Advantage Actor Critic (A2C) algorithm. Cooperation between the agents is achieved by sharing information regarding spectrum utilization between different domains, and it is shown to outperform both baseline heuristics and the results from [12].

In [24] authors develop a transfer learning framework for DRL in optical networks to mitigate the retraining needs and to provide better generalization. Authors evaluate their approach on transfer between different RMSA topologies, and from RMSA to service function chain provisioning. Their proposed approach achieves overall lower training times and performance performance similar to or better than heuristic baselines.

In [25], authors devise a Multi-Agent DRL (MA-DRL) for inter-domain RMSA, in which each agent acts as a domain broker. Agents operate independently on an abstract topology composed of border nodes and virtual nodes, the latter representing the intra-domain networks. The proposed approach manages to outperform the considered heuristic baselines.

In [26] authors develop a DRL-based admission control policy for network slicing in 5G Radio Access Networks (RANs). For each slice request, the DRL agent receives a penalty if either the request is rejected, or cannot be scaled up when needed. The proposed approach is able to outperform the considered baselines, resulting more robust to variations in the penalty values and in the service distribution.

While all of the mentioned works attain remarkable results and constitute significant advances in the applications of DRL to optical networks, each fails to fully consider at least one of the following: 1) an assessment on the generalization to unseen previous problem instances, or 2) a comparison against stronger algorithms than baseline greedy heuristics, such as state-of-the-art metaheuristics and ILP-based approaches. We argue that a thorough assessment of these two points is paramount in order not to overestimate the performance of DRL-based solution methods, and to fairly evaluate their pros and cons. In this work, we have developed DRL-based solution methods based on the state-of-the-art PPO algorithm, and we have extensively benchmarked them against both greedy heuristics and a state-of-the-art Genetic Algorithm.

III. Background: Deep Reinforcement Learning

The field of Reinforcement Learning (RL) is inspired by behavioral psychology: an RL agent interacts with an environment, the latter typically represented as a Markov Decision

Process (MDP), in order to learn a policy that maximizes the expected sum of rewards over a time horizon [27].

Let $\mathcal{A}$ be the action space and $\mathcal{S}$ be the state space. A policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ defines the behavior of the learning agent. In particular, $\pi(a_t = a|s_t = s)$ defines the probability of selecting action $a \in \mathcal{A}$ given the state $s \in \mathcal{S}$ at time step t.

The goal of a RL agent is to maximize the expected sum of rewards, known as the return, over a time horizon T. The discounted return at time step t is defined as follows:

$$G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k \tag{1}$$

where T is the time horizon, r_k is the reward at time step k and $\gamma \in [0, 1]$ is the discount factor. The magnitude of the discount factor determines the importance of future rewards in relation with immediate rewards.

The value function of policy π can be defined as the expected return when starting from state s and acting according to policy π, as follows:

$$V_\pi(s) = \mathbb{E}_\pi(G_t|s_t = s) \tag{2}$$

Similarly, the action-state value function, also known as Q-value, of policy π is defined as the expected return when starting from state s and taking action a, following policy π for all the following states:

$$Q_\pi(s, a) = \mathbb{E}_\pi(G_t|s_t = s, a_t = a) \tag{3}$$

The difference between the action-state value function and the state value function for a given state-action pair is defined as the advantage function:

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s) \tag{4}$$

A policy π is said to be better than policy π', that is, $\pi \geq \pi'$, if and only if $V_\pi(s) \geq V_{\pi'}(s)$, $\forall s \in \mathcal{S}$. All optimal policies π_* share the same value function $V_*(s)$ and action-state value function $Q_*(s, a)$, defined as follows:

$$V_*(s) = \max_\pi V_\pi(s) \tag{5}$$

$$Q_*(s, a) = \max_\pi Q_\pi(s, a) \tag{6}$$

It can be demonstrated that a policy that acts greedily with respect either to $V_*(s)$ or $Q_*(s, a)$ is an optimal policy.

Traditional RL algorithms, like Q-Learning and SARSA, store policies and value functions in tables. However, many practical environments exhibit a number of states so large that tabular methods become no longer feasible in terms of memory occupation. Hence, in DRL, policies and value functions are parameterized via function approximators, i.e., Deep Neural Networks (DNNs), leveraging the ability of DNNs to generalize to states unseen during training. Assessing and improving the generalization capabilities of DRL to unseen states is currently an active research field [15], [16].

Many state-of-the-art DRL algorithms include the following two fundamental building blocks: 1) a policy estimation phase, in which an approximation of the value function is computed, and 2) a policy improvement phase, in which the trained policy is optimized according to the value function estimation.

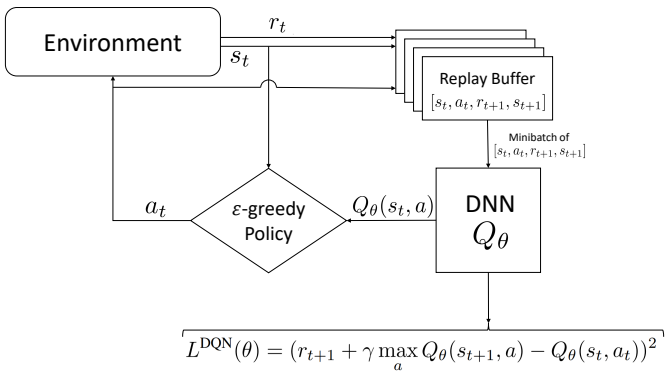


Fig. 1. Illustrative diagram of the main DQN elements, highlighting the computation of the DQN loss function.

Within such a framework, RL algorithms can be categorized as either on-policy or off-policy algorithms. In on-policy algorithms, the trained policy is used to gather samples for the policy evaluation phase. In off-policy algorithms, a behaviour (usually exploratory) policy is used to gather samples for the policy evaluation phase instead of the trained policy. Typically, off-policy algorithms are more sample-efficient than on-policy algorithms as they can reuse past experiences [5], thus they are useful in environments that are expensive to evaluate, thought they tend to have slower convergence times than on-policy algorithms.

Furthermore, DRL algorithms can be further divided among policy-based, value-based and Actor-Critic. Policy-based algorithms store only a representation of the trained policy. Value-based algorithm store only a representation of the value function, from which the policy is derived. Finally, Actor-Critic algorithms store both a representation of the policy (the Actor) and a representation of the value function (the Critic).

In the following, we provide a brief overview of two relevant classes of DRL algorithms, Deep Q-Learning (DQN) and policy gradient algorithms, the latter focusing on PPO, which employed for numerical evaluation in this work.

A. Deep Q-Learning (DQN)

DQN [5] and its variants [28]–[30] are off-policy, value-based algorithms. The aim of DQN algorithms is to learn an approximation of $Q_*(s, a)$, parameterized as $Q_\theta(s, a)$ by a DNN, where θ are its parameters. The learned policy is a deterministic greedy policy, which chooses the action corresponding to the highest $Q_\theta(s, a)$ estimate. Being off-policy, a behaviour policy is used to gather the samples used for updating the Q-values estimates. An example of such a behaviour policy is an ϵ-greedy policy, which either acts greedily with respect to $Q_\theta(s, a)$ with probability $1 - \epsilon$ or randomly with probability ϵ.

The Q-Learning algorithm [31] defines the following update rule, deriving from the Bellman optimality equation:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \\ + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{7}$$

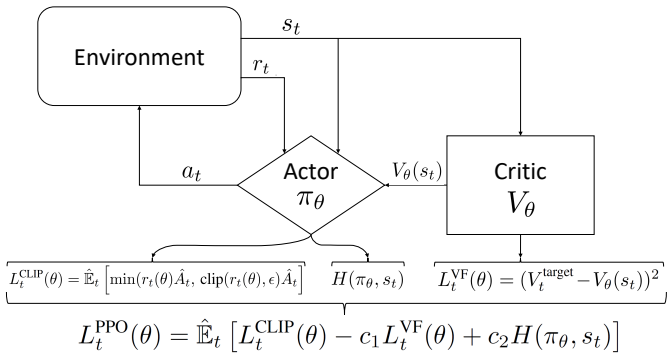


Fig. 2. Illustrative diagram of the PPO Actor-Critic architecture in which the Actor and the Critic share the same parameters θ, highlighting the computation of the individual terms appearing in the full loss function.

where α is the learning rate, which controls the magnitude of the updates.

DQN makes use of a technique known as experience replay: the behaviour policy is used to gather samples, to be stored in a replay buffer. Then, a minibatch of samples is sampled randomly from the replay buffer. For each sample, the update value of (7) is computed as the following loss function:

$$L^{\text{DQN}}(\theta) = (r_{t+1} + \gamma \max_a Q_\theta(s_{t+1}, a) - Q_\theta(s_t, a_t))^2 \quad (8)$$

At this point, the DNN parameters θ are updated via minibatch gradient descent. An illustrative diagram of the main DQN elements is shown in Fig. 1.

B. Proximal Policy Optimization (PPO)

Policy gradient algorithms are among the most relevant in the field of DRL. Let π_θ be a stochastic parameterized policy with parameters θ (i.e. a DNN, where the parameters are the weights of the neural network). The expected return given policy π_θ can be expressed as follows:

$$J(\theta) = \mathbb{E}\left[\sum_{k=1}^{T} r_k\right] \quad (9)$$

The objective of policy gradient is to optimize the parameters θ such that the expected return is maximized. Policy gradient algorithms achieve this goal by computing an estimate of the gradient of (9) and running a stochastic gradient ascent algorithm. In the literature, a plethora of different gradient estimators has been developed [32]. One of the most commonly used gradient estimators is the following:

$$\hat{g} = \hat{\mathbb{E}}_t[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t] \quad (10)$$

which, in practical implementations, results from the differentiation of the following loss function:

$$L^{\text{PG}}(\theta) = \hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \quad (11)$$

where $\hat{A}_t$ is an estimate of the advantage function, and $\hat{\mathbb{E}}_t$ indicates an empirical average over a batch of samples. In particular, methods for computing accurate estimates of the advantage function, such as Generalized Advantage Estimation [32], require an estimate of the value function. Therefore, computing an approximation of this expression would in principle require two DNNs: the "Actor", parameterizing the policy, and the "Critic", parameterizing the value function. In practical application scenarios a common design choice is to let the Actor and the Critic share parameters, given that they most likely would learn similar features.

In this work, an implementation of Proximal Policy Optimization (PPO) [33], one of the main state-of-the-art policy gradient algorithms, will be used for training the agents. PPO is an on-policy algorithm, since the trained policy is used to gather samples, with the following loss function:

$$L_t^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), \epsilon)\hat{A}_t\right)\right] \quad (12)$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the ratio between the policy before and after the parameters update. The clip function, which depends on the hyperparameter ϵ, prevents the occurrence of too large parameter updates that would destabilize the learning process. The complete PPO loss function at each time-step reads as follows:

$$L_t^{\text{PPO}}(\theta) = \hat{\mathbb{E}}_t\left[L_t^{\text{CLIP}}(\theta) - c_1 L_t^{\text{VF}}(\theta) + c_2 H(\pi_\theta, s_t)\right] \quad (13)$$

where $L_t^{\text{VF}}(\theta) = (V_t^{\text{target}} - V_\theta(s_t))^2$ is the squared-error loss of the value function estimator, $H(\pi_\theta, s_t)$ is the entropy of policy π_θ in state s_t, and c_1, c_2 are hyperparameters. Introducing a bonus in the loss function proportional to the policy entropy encourages sufficient exploration during training and prevents early convergence to local minima [23]. An illustrative diagram of the PPO architecture, showing the individual terms of equation (13), is shown in Fig. 2.

Considering the computation of $L^{\text{VF}}(\theta)$, let the n-step return be defined as follows:

$$G_{t:t+n} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V_\theta(s_{t+n}) \quad (14)$$

Where V_θ is the estimate of the value function, parameterized by θ. Then, V_t^{target} is defined as the TD(λ) estimator [27]:

$$G_t^\lambda = (1-\lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t \quad (15)$$

where $\lambda \in [0, 1]$ is a hyperparameter regulating the trade-off between bias and variance in the value function estimation. In particular, values of λ closer to 0 induce higher bias, whereas values of λ close to 1 induce higher variance.

To mitigate sample inefficiency due to being on-policy, PPO can leverage multiple parallel Actors to gather samples from the current policy, which are stored in a buffer and then used to optimize the DNN parameters via minibatch stochastic gradient descent. Unlike in DQN, past experiences are discarded after an update.

As there is no silver bullet in RL, it is difficult to assess a-priori which is the best algorithm for a particular task. For the problem considered in this work, we chose PPO as it was the algorithm that we observed to perform best.

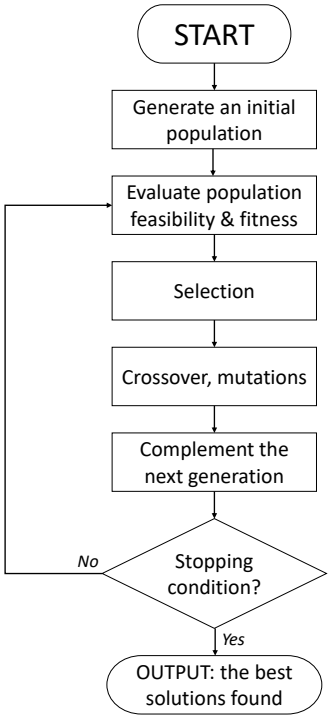


Fig. 3. Genetic Algorithm flow diagram.

C. DRL for Optimization

In [4], methods based on ML for solving traditional optimization problems in Operations Research literature are surveyed. In the framework of optimization algorithms, using a DRL agent, trained to output a feasible solution from an arbitrary problem instance, can be seen as a very complex heuristic, in a framework defined by authors as "end-to-end learning". Heuristic algorithms, in general, can be disassembled into a sequence of subroutines with various degrees of complexity. If one regards the action space of a DRL agent as a collection of subroutines, the task of a DRL agent becomes the one to find a (possibly stochastic) succession of subroutines. Therefore, applying DRL to solving optimization problems can be interpreted as a method of building a specialized heuristic for a certain problem class.

Policy gradient algorithms are able to learn a stochastic policy, which is equivalent to building a heuristic with some stochastic behaviour. We argue that policy stochasticity (i.e., maintaining a small degree of exploration with respect to a deterministic policy) can be a desirable property of DRL when applied to solving optimization problems. In the following, experimental results will confirm the validity of this claim.

IV. Routing and Wavelength Assignment

The RWA problem consists in assigning a route and wavelength to a set of lightpath requests in an optical WDM network. The objective here is to maximize the number of accommodated connections. RWA can be either for static traffic (traffic matrix is known in advance), or for dynamic traffic (requests arrive and depart in a stochastic way). This work considers static RWA, in which we assume wavelength continuity constraints must be enforced for each lightpath.

TABLE I
RWA model parameters and variables

Parameters	
V	Set of routing nodes
E	Set of bidirectional links
W	Set of available wavelengths in each link.
P	Set of pre-computed K-shortest paths.
$P_{(s,d)}$	Set of pre-computed K-shortest paths between node pair (s,d), $s \in V$, $d \in V$, $s \neq d$. $\lvert P_{(s,d)} \rvert = K$
$\rho_{(s,d)}$	Number of connections requested by node pair (s,d), $s \in V$, $d \in V$, $s \neq d$
Variables	
x_p^w	1 if wavelength $w \in W$ is occupied in path $p \in P$, 0 otherwise

A. ILP formulation

Static RWA can be formalized as an ILP formulation [34]. ILP parameters and variables are reported in Table I.

$$\max \sum_{p \in P} \sum_{w \in W} x_p^w \tag{16}$$

$$\sum_{p \mid l \in p} x_p^w \leq 1 \quad \forall l \in E, \forall w \in W \tag{17}$$

$$\sum_{p \in P_{(s,d)}} \sum_{w \in W} x_p^w \leq \rho_{(s,d)} \quad \forall s, d \in V, s \neq d \tag{18}$$

$$x_i^w \in \{0,1\} \quad \forall i \in P, \forall w \in W \tag{19}$$

Objective function (16) maximizes the number of accommodated connection requests. Constraints (17) impose the wavelength continuity on each established lightpath. Constraints (18) impose that the total accommodated connection requests between each source-destination pair do not exceed the demand. Finally, equations (19) define the variable domains.

Though the ILP provides an exact solution, it is not scalable, as RWA is of NP-Hard complexity [35]. Nevertheless, for small enough problem instances an optimal solution can be found in short computing times, serving as a benchmark for the other algorithms considered in this work.

B. Heuristic and metaheuristic baselines

In the numerical evaluations, the following heuristic and metaheuristic algorithms will be benchmarked against the developed DRL-based solution methods:

- SP-FF: for each connection request, the first wavelength available (i.e., first-fit) in the shortest path between the source-destination pair is selected.
- KSP-FF: for each connection request, the first wavelength available in the shortest among the K-shortest paths between the source-destination pair is selected.
- LLP-FF: for each connection request, the first wavelength available in the least loaded among the K-shortest paths between the source-destination pair is selected.
- GA-FF: a feasible routing is generated via the procedure illustrated in Fig. 3, and the wavelength assignment is performed in a first-fit way.

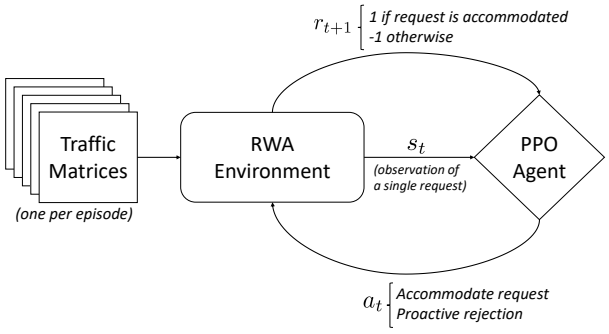


Fig. 4. Illustrative diagram of the learning process. For each episode, the RWA environment draws requests from a traffic matrix sampled from a known probability distribution. Requests are presented to the PPO agent via a properly encoded observation. Based on that information, the PPO agent chooses an action. Finally, the RWA environment produces a reward based on the outcome from the chosen action.

V. DRL Algorithm For RWA

In this section, our DRL-based approach for solving the RWA problem will be detailed, and the design choices of the developed DRL algorithms will be illustrated.

Given the traffic matrix as input, the trained DRL agent must output a feasible solution for the RWA problem (16)-(19). Therefore, the learning aim is to build a specialized heuristic for the considered RWA problem.

In order to learn a mapping from states to actions, a DRL agent must be provided with proper representations of the environment state, which are called observations. In fact, even though the static RWA problem is a so-called fully-observable environment, providing the agent with the whole traffic matrix and complete information regarding the spectrum occupancy is not an effective strategy. Large observations, other than scalability issues, introduce significant learning challenges, due to the fact that the policy to be learned becomes more complex as the size of an observation increases.

The set of all possible actions a DRL agent can choose from is called action space. Similarly to the previous discussion, an agent with a larger action space indeed has the potential of learning a better policy, however the complexity of the learning phase increases, since more exploration is required.

To summarize, the choice of the state representations (i.e., the observations) and of the action space are of critical importance in designing a DRL algorithm, and it may not be possible a-priori to determine which ones are the most effective for the particular task at hand. Properly shaping the state representation and the expressiveness of the action space, such that the best final performance is attained, constitutes a significant research challenge.

An episode corresponds to a particular RWA problem instance (16)-(19), for which the traffic matrix is drawn from a given probability distribution. For the problem instances used for training, the traffic distribution is assumed to be uniform for each possible source-destination pair. During an episode, at each time step, the agent is presented with a single connection request and it has to decide, based on an observation, whether to try to accommodate it or to proactively reject it. Therefore, the number of time-steps in an episode is equal to the number of connection requests. At the end of an episode, the allocated spectral resources are freed, and a new traffic matrix is processed. An illustrative diagram of the learning framework is shown in Fig. 4.

Observations represent the network state by including the following information: the source-destination pair for the request to be accommodated at the current time-step, and the spectrum occupation in each of the K-shortest candidate paths. Even though there is some loss of information with respect to the full traffic matrix and the complete spectrum occupation in the network, experimental results suggest that it is sufficient for the agents to learn an effective policy.

We train two distinct agents, differing only in how the action space and the observation space were defined, to assess the impact of these design choices on performance.

We use PPO [33] to train the agents, given that it is one of the main state-of-the-art DRL algorithms. The agents are trained via the PPO algorithm implementation in Stable Baselines, an open-source library providing reliable implementations of state-of-the-art DRL algorithms [36].

A. Agent 1: PPO-FF

The goal of PPO-FF is to learn a stochastic, first-fit policy given information regarding the current request and the spectrum occupancy of the K-shortest paths between request's source and destination nodes.

1) Action space: The action space for this agent is discrete and of dimension $(K+1)$, i.e., each action corresponds either to selecting one of the K-shortest paths for the source-destination pair of the current request, or to a proactive rejection. When a path is selected, if there is at least one available wavelength along the selected path, the first wavelength is used to accommodate the request. Otherwise, if no wavelengths are available, the request is rejected.

2) Observations: An observation consists in a $(2|V|+2K)$ vector. In particular, a $(2|V|)$ vector contains the one-hot-coded source-destination pair for the current request, a K-dimensional binary vector determines whether or not there is a wavelength available in each of the K paths, and a K-dimensional vector contains the total load on each path. Observations are normalized to be in the range $[-1, 1]$.

B. Agent 2: PPO-Full

The goal of PPO-Full is to learn a stochastic policy given information regarding the current request and the spectrum occupancy of the K-shortest paths. Differently from PPO-FF, PPO-Full can accommodate a request in any wavelength available in each of the K paths.

1) Action space: The action space for this agent is discrete and of dimension $(K|W|+1)$, such that each action corresponds either to selecting a path and a wavelength or to a proactive rejection. If a path and a wavelength are selected, the current request is accommodated to the selected path and to the selected wavelength, if available. Otherwise, if no wavelengths are available, the request is rejected.

2) Observations: An observation consists in a $(2|V| + K(|W| + 1))$ vector. In particular, a $(2|V|)$ vector contains the one-hot coded source and destination nodes for the current request, a $(K|W|)$ binary matrix determines the wavelength availability in each of the K paths, and a K-dimensional vector contains the total load in each of the K paths. Observations are normalized to be in the range $[-1, 1]$.

Compared to PPO-FF, PPO-Full has a larger action space, which grants it more freedom in deciding how to accommodate requests. However, larger action space introduce more difficult learning challenges, both in terms of exploration of the action space and in terms of having to learn a more complex mapping from observations to actions.

C. Reward Function

We have defined two different reward functions for the agents: a sparse reward and a shaped reward. The sparse reward reads as follows:

$$\mathrm{r}_{\text{sparse}} = \begin{cases} 1 & \text{if accommodated} \\ -1 & \text{otherwise} \end{cases} \tag{20}$$

The shaped reward reads as follows:

$$\mathrm{r}_{\text{shaped}} = \begin{cases} \alpha \frac{C_j}{\max_i C_i} + \beta \frac{\min_i h_i}{h_j} & \text{if accommodated} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

where $i = 1 \dots K$, K is the number of shortest paths for each source-destination pair, j is the chosen path among the K-shortest, C_i is the capacity (i.e., the number of available wavelengths) on path i, h_i are the number of hops taken by path i, and α, β are positive scalars. The shaped reward induces the agent to find a trade-off between a load balancing strategy and a shortest-path strategy.

The sparse reward holds more expressive power with respect to the shaped reward, as it does not bias the agent with any specific strategy. However, when the link capacity is high, using the sparse reward results in all actions (except proactive rejection) having reward equal to $+1$ for the majority of a training episode, being the network unloaded. For low link capacities, we have observed from experimental results that using the shaped rewards yields performance worse or comparable performance compared to using the sparse reward. Therefore, in the following, we will employ the shaped reward only in the case of high link capacities. Overall, the use of a carefully shaped reward is suggested only if learning using a sparse reward is exceedingly difficult.

D. Multi-Start Agents

Both PPO-FF and PPO-Full learn a stochastic policy: given an observation, the agents output a selection probability for each action in the action space. Therefore, it is possible to evaluate the agents in either a deterministic or a non-deterministic way. With a deterministic evaluation, the agent always chooses the action with the highest selection probability. With a non-deterministic evaluation, the agent samples an action based on the selection probabilities, therefore maintaining a degree of exploration. While a deterministic evaluation is paramount

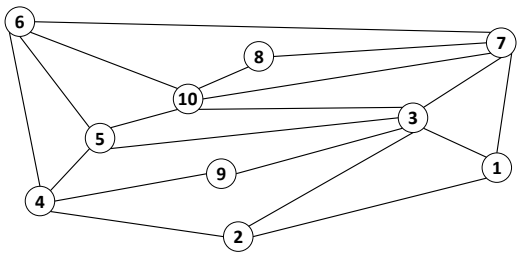


Fig. 5. 10-node reference network used for training the agents.

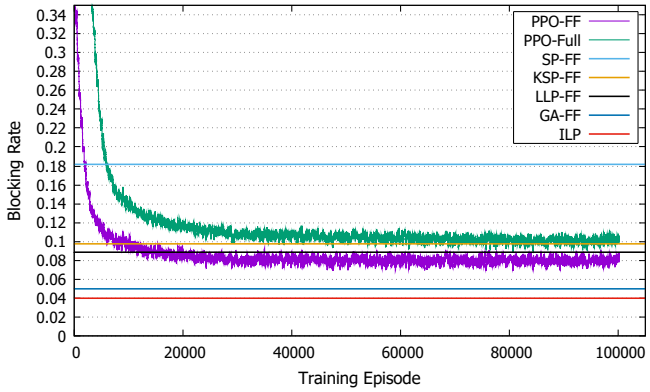


Fig. 6. Blocking rate of the PPO agents as a function of the training episode. The average blocking rates over 1000 test traffic matrices of the greedy heuristics, Genetic Algorithm and ILP are also reported.

for assessing the performance of an agent in applications such as real-time control, we note that non-determinism is a desirable property when applying DRL to solving optimization problems. Leveraging on policy stochasticity draws a parallelism with metaheuristics that implement random behaviours in order to escape local minima. For this reason, the average performance of an agent results to be an underestimation of what can actually be achieved.

To take advantage from policy stochasticity, a trained DRL agent can be cast within a Multi-Start (MS) heuristic framework [37], in which the same problem instance is solved multiple times and the only best solution is retained. Clearly, this approach introduces a significant computational overhead, as it evaluates the same RWA environment multiple times, but better solutions can be achieved with respect to a deterministic evaluation. In our evaluations, we chose a number of independent starts equal to 8 (MSx8), as lower numbers would not bring a significant improvement with respect to a deterministic evaluation, and higher numbers would not bring any significant further improvements. In the following, we will denote PPO-FF-MSx8 and PPO-Full-MSx8 the Multi-Start versions of PPO-FF and PPO-Full, respectively.

We note that casting DRL agents in a Multi-Start framework has already been explored in [38], albeit for a completely different application. To the best of our knowledge, this is the first work that applies Multi-Start in DRL for optimization in optical networks.

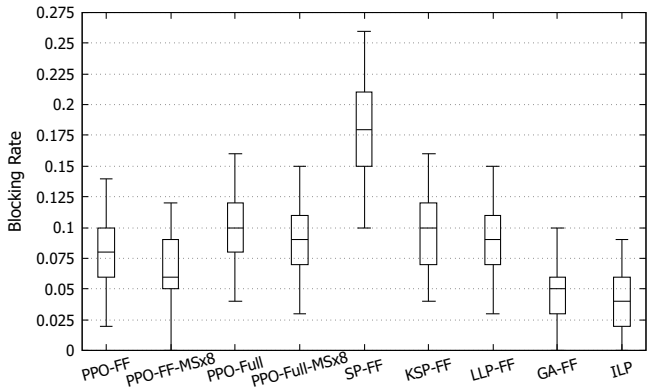


Fig. 7. Blocking rate of the PPO agents, heuristics and ILP on 1000 traffic matrices from the training environment. Whiskers encompass 95% of the data.

VI. Illustrative Numerical Results

In this section, we compare the proposed DRL-based approaches to 1) baseline greedy heuristics, 2) GA (as representative of a state-of-the-art metaheuristic) and 3) ILP, in terms of request blocking rate and computational time.

The implementation for the RWA environment is based on the Optical RL-Gym framework [39]. The baseline greedy heuristics considered in this work are Shortest-Path-First-Fit (SP-FF), K-Shortest-Path-First-Fit (KSP-FF) and Least-Loaded-Path-First-Fit (LLP-FF). A flow diagram describing the behaviour of the Genetic Algorithm (GA-FF) is illustrated in Fig. 3 [40], [41]. The parameters of GA-FF have been fine-tuned in order to get the best trade-off between computing times and solution quality. Finally, exact solutions were obtained by solving the ILP formulation (16)-(19) via the state-of-the-art commercial solver Gurobi v.9.1 [42].

The reference 10-node network used for training and evaluation is illustrated in Fig. 5. The number of pre-computed K-shortest paths was set to $K = 3$, and path lengths were calculated in terms of number of hops. The number of wavelengths available in each link was set to $W = 10$. The agents were trained for a total of 10^6 episodes, with each episode consisting of 100 connection requests. In particular, requests were generated according to a uniform traffic distribution between all the possible source-destination pairs.

For the hyperparameters of PPO, we chose a shared architecture for the policy and the value function. We chose a Multi-Layer Perceptron (MLP) of 5 layers with 128 neurons each, with Exponential Linear Units (ELU) as activation functions. We have set the discount factor γ to 0.95, the learning rate to $5 \cdot 10^{-5}$, the entropy coefficient c_2 to 10^{-4}, the buffer size to 32768, and the minibatch size to 1024. All the other hyperparameters were left as their default values. For a detailed investigation regarding efficient training of on-policy algorithms, in particular PPO, the reader can refer to [43].

A. Performance on the training environment

Fig. 6 shows the blocking rates of the two agents during the training phase, and the average blocking rates over 1000 test problem instances computed by the considered heuristic algorithms and the ILP modelPPO-Full, even though it has more freedom of action than PPO-FF, achieves worse average performance with respect to PPO-FF at the end of the training phase. This behaviour is due to the fact that PPO-Full must deal with a larger action space and more complex observations than PPO-FF, therefore the learning policy becomes more complex. In fact, PPO-Full outperforms only SP-FF, and, on average, it has worse performance than all the other considered heuristic algorithms. On the other hand, PPO-FF is able to outperform, on average, all the considered baseline greedy heuristics, with 57.3%, 19.8% and 13.4% improvement with respect to SP-FF, KSP-FF and LLP-FF, respectively. However, the 36.1% gap with respect to GA-FF shows that PPO-FF on average does not outperform a fine-tuned metaheuristic. Finally, a significant 48.9% optimality gap between PPO-FF and ILP suggests that there is still much room for improvement.

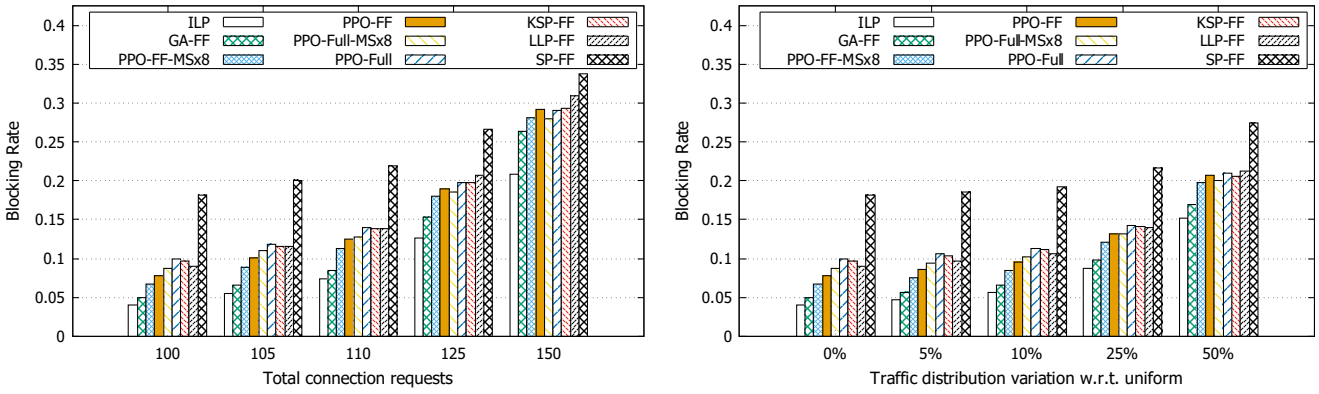
In Fig. 7 the blocking rates of the PPO agents, heuristics and ILP are reported, evaluated on 1000 traffic matrices drawn from the training environment. We observe that Multi-Start can indeed improve the performance of PPO agents, as policy stochasticity is an effective tool for escaping local optima. Moreover, een though GA-FF is still the best heuristic, PPO-FF-MSx8 is able to close the gap, much more effectively than PPO-FF. This is a remarkable result, given the simplicity of the action space of PPO-FF-MSx8 with respect to a sophisticated metaheuristic such as GA-FF.

B. Generalization capabilities

Due to overfitting, the trained DRL agents may experience performance degradation when evaluated on traffic matrices that differ in distribution from the ones generated in the training phase. Therefore, since realistic application scenarios may sensibly differ from the training environment, properly assessing the generalization capabilities of the trained DRL agents is paramount. In the following, the generalization capabilities of the trained DRL agents will be assessed on traffic matrices unseen during training, in terms of the total number of connection requests and of the traffic distribution.

1) Generalization on the total number of requests: To assess the generalization capability of our DRL agents to a different number of connection requests per episode, we consider the cases when the performance of the DRL agents are tested with 100, 105, 110, 125 and 150 connection requests per episode. We remark that, in all the above mentioned cases, the DRL agents were always trained considering a total of 100 connection requests per training episode, and that the traffic distribution is assumed to be uniform between all possible source-destination pairs for all requests. The average blocking rates of the DRL agents, of the heuristics and of the ILP for different numbers of total requests are reported in Fig. 8a.

Comparing the relative performance of the DRL agents with the other approaches, it can be observed that the PPO-FF and PPO-Full suffer some slight performance degradation as the number of total connection requests increases. This suggests a small degree of overfitting with respect to the number of requests seen during training. In particular, for 150 total connection requests (i.e., a 50% increase with respect to training), PPO-FF is on-par with KSP-FF.



(a) (b)

Fig. 8. Average blocking rates of the PPO agents, heuristics and ILP for 1000 test traffic matrices, as a function of: (a) the total number of connection requests, and (b) the variation from the training uniform traffic distribution.

On the other hand, GA is able to consistently outperform PPO-FF by an average of 26.4%. This is a reasonable result, given that GA runs more complex subroutines (i.e., crossover, mutations, selection) in order to generate a feasible solution. Moreover, GA went through an extensive process of parameter fine-tuning to get the possible best blocking rates. For this particular task, GA has proven to be extremely sensitive to parameter tuning, to the point that a wrong choice in the parameters would have resulted in a solution worse than the greedy heuristics. PPO, on the other hand, has proven to be quite robust to the choice of hyperparameters.

Since the observations do not depend on the number of total connection requests, it is reasonable to assume that the agents were able to learn a policy that can generalize with respect to the number of requests per episode. Furthermore, the best DRL agent is not unequivocal, and the performance gap between PPO-FF and PPO-Full becomes smaller with increasing total number of requests per episode, suggesting that for a high traffic load there may not be a significant benefit in employing either of the two strategies. However, PPO-FF exhibits better performance overall with respect to PPO-Full, and works with a simpler observation space. Therefore, since the policy to be learned is less complex, fine-tuning the DNN architecture may further lighten the computational burden of the DRL agent.

The solutions computed by the DRL agents can be further improved by leveraging stochasticity and employing a Multi-Start framework. Considering PPO-FF, employing Multi-Start with 8 independent starts (PPO-FF-MSx8) allows to improve performance by an average of 8.77%, with a best improvement of 13.8% for 100 total connection requests. The relative improvement provided by the Multi-Start becomes less significant as the load increases, as the average absolute improvement in the blocking rate is equal to 1.11%, which becomes negligible for higher blocking rates. For instance, with 150 total connection requests, only a 3.7% relative improvement is gained by the Multi-Start. PPO-FF-MSx8 brings an average improvement of 43.9%, 17.0% and 17.7% over SP-FF, KSP-FF and LLP-FF, respectively, thus surpassing by a significant margin all baseline greedy heuristics. Furthermore, PPO-FF-MSx8 is able to improve the gap with respect to the near-optimal performance provided by GA, with an average gap of 19.7%.

2) Generalization on the traffic distribution: We now evaluate how the performance of the DRL agents is affected when the traffic distribution considered during test phase is different compared from the one used during DRL algorithm training. To do so, during the test phase we do not consider that all source-destination pairs are chosen with uniform probability, but alter their probability considering different variations, namely $\pm 5\%$, $\pm 10\%$, $\pm 25\%$ and $\pm 50\%$ with respect to the uniform distribution. The total number of connection requests was set to 100 for all instances. The average blocking rates of the DRL agents, of the heuristics and of the ILP for the different traffic distributions are reported in Fig. 8b.

In general, the blocking rates increase with the amount of variation from the uniform distribution. Indeed, by keeping the total number of requests the same, an unbalanced distribution is more likely to create situations of congestion, given that more spectral resources are requested from specific source-destination pairs, leading to overutilization of specific links.

As in the previous evaluation, by comparing the relative performance of the DRL agents with the other solution methods, it can be observed that PPO-FF and PPO-Full do suffer from performance degradation when tested on traffic distributions different than the ones seen during training. This suggests a degree of overfitting with respect to the traffic distribution seen during training. On the extreme case, it can be seen that PPO-FF shows similar performance to KSP-FF for a 50% variation with respect to the training distribution. On the other hand, PPO-FF is able to outperform all the baseline greedy heuristics for all the other considered sets of instances, which are more similar to the train distribution. Furthermore, for the same reasons commented in the previous subsection, the gap in performance between PPO-FF and PPO-Full becomes smaller as the tested distribution becomes more and more unbalanced.

As seen before, DRL agents can improve the solutions using Multi-Start. In particular, considering PPO-FF, employing Multi-Start with 8 independent starts (PPO-FF-MSx8) allows

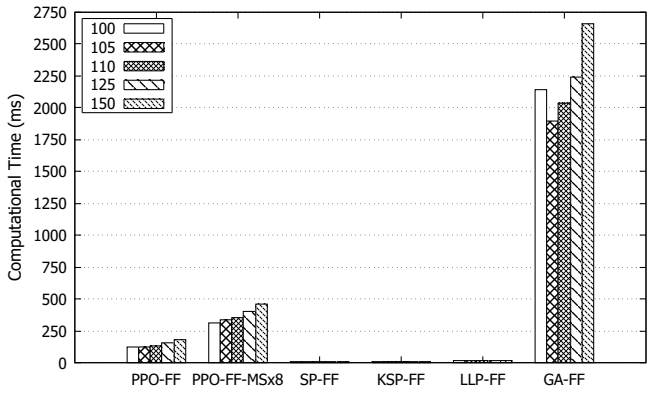


Fig. 9. Average computational times of the DRL agents and the heuristics varying the number of connection requests.

to improve performance by an average of 10.1%, and the absolute improvement provided by the Multi-Start becomes less significant as the distribution becomes more unbalanced, i.e., when the network congestion increases. PPO-FF-MSx8 is able to consistently outperform all greedy heuristics, with an average improvement of 50.1%, 19.9% and 17.6% for SP-FF, KSP-FF and LLP-FF respectively. Similarly to the previous case, GA-FF is able to consistently outperform PPO-FF, with an average gap of 28.9%. With PPO-FF-MSx8, it is possible to reduce this gap, reaching an average 21.0% gap, at the price of a higher computational overhead due to the Multi-Start.

C. Computational times

Fig. 9 reports the average computational times required by the DRL agents and the heuristics varying the number of total number of connection requests[1]. For the DRL agents, only PPO-FF is reported, since the computational times of PPO-Full are very similar, given that they employ the same DNN architecture. The greedy heuristics are by far the fastest approaches, as expected. PPO-FF-MSx8 is on average 61.5% times slower than PPO-FF, given that it requires eight independent executions of PPO-FF. The additional computational overhead by the Multi-Start is mitigated via the use of multiprocessing on each independent start. GA-FF is by far the slowest method, more than five times slower than PPO-FF-MSx8, and an order of magnitude slower than PPO-FF.

Note that the parameters of GA-FF could be tuned in order to reduce the computing times, but that would have impacted severely the solution quality, to the point of being either on-par or plainly outperformed by PPO-FF and PPO-FF-MSx8. Therefore, to get the best solution quality possible out of GA-FF, one must tolerate relatively long execution times compared to the other solution methods.

D. Scalability

To assess the scalability of the proposed DRL-based approach, the two following scenarios will be considered: 1) networks with a larger number of nodes, and 2) networks with a higher number of wavelengths per link. In the following, only PPO-FF and PPO-FF-MSx8 are considered, as PPO-Full and PPO-Full-MSx8 were shown to exhibit an overall worse performance.

1) Larger networks: In this evaluation setting, the 24-node US backbone topology and a 100-node Gabriel graph [44] were considered. Because of their grid-like structure, Gabriel graphs accurately mimic the graph structure of physical layer topologies [45]. For each topology, the number of wavelengths per link was set to 10 and the number of connection requests per episode, uniformly distributed among all source-destination pairs, was set to 100.

Fig. 10a shows the blocking rates for the PPO agent, the baseline heuristics and ILP for the 24-node US backbone and the 100-node Gabriel graph. For both networks, PPO-FF learns a policy that outperforms SP-FF and KSP-FF, but attains very similar results with respect to LLP-FF. With the use of Multi-Start, PPO-FF-MSx8 is able to outperform all greedy heuristics. As for the 10-node network, there is still a gap in performance with respect to GA-FF and ILP.

2) Higher capacity networks: In this evaluation setting, the 10-node network in Fig. 5, the 24-node US backbone and a 100-node Gabriel graph were considered. For each topology, the number of wavelengths per link was set to 80 and the number of connection requests per episode, uniformly distributed among all source-destination pairs, was set to 800. For this scenario, we used the shaped reward defined in Eq. (21). According to [46], and confirmed by preliminary experiments, a policy that favours the least loaded choice is more effective when there is blocking in the network. Therefore, the values for α and β in Eq. (21) have been set to $\alpha = 0.9$ and $\beta = 0.1$.

Moreover, an additional element was added to the state observation, namely, a (K) vector containing the length in hops for each of the candidate paths.

Fig. 10b shows the blocking rates for the PPO agent, the baseline heuristics for the 10-node network, the 24-node US backbone and the 100-node Gabriel graph. ILP results for the 100-node network are not reported due to excessively long computational times. For the 10-node network, using the shaped reward, both PPO-FF and PPO-FF-MSx8 are able to outperform the greedy heuristics and attain comparable performance with respect to GA-FF. For the 24-node network, only PPO-FF-MSx8 is able to outperform all greedy heuristics, in particular slightly outperforming LLP-FF and attaining comparable performance with respect to GA-FF. Finally, for the 100-node network, PPO-FF and PPO-FF-MSx8 are able to outperform SP and KSP-FF, but do not outperform LLP-FF. This suboptimal performance of PPO-FF and PPO-FF-MSx8 for the largest instance suggests a need for more sophisticated DRL-based solution methods than one which only learns an admission control and routing rule.

VII. Conclusion

In this work, we compared the performance of DRL-based methods against a state-of-the-art metaheuristic and ILP for

[1]GA-FF computational times may not increase monotonically due to several early stopping conditions, whose effect on the computational times is unpredictable. Early stopping avoids excessive computing times while preserving solution quality.

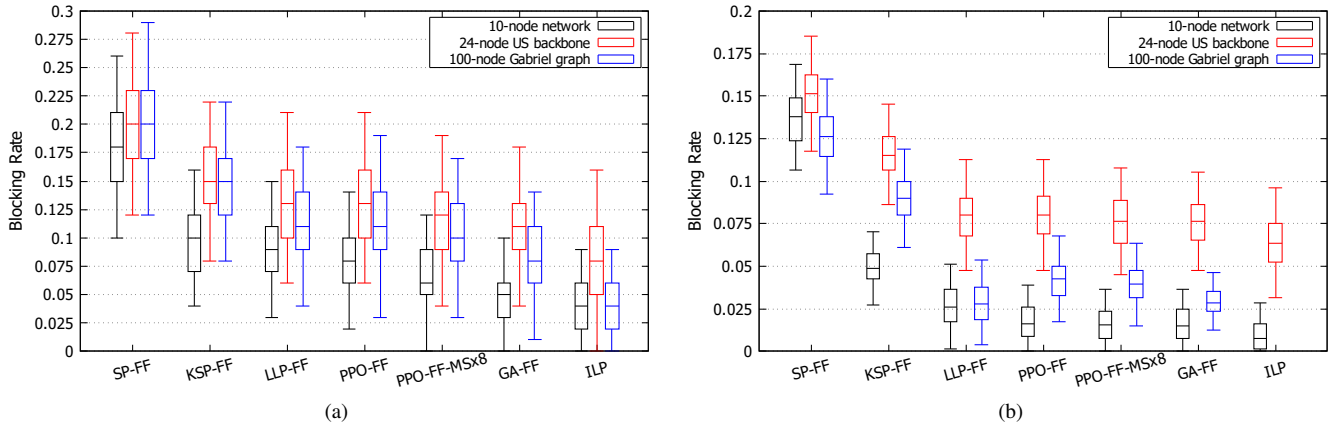


Fig. 10. Blocking rates of the PPO agents, heuristics and ILP for 1000 test traffic matrices. (a) 10 wavelengths per link, 100 uniformly distributed connection requests per episode. (b) 80 wavelengths per link, 800 uniformly distributed connection requests per episode. Whiskers encompass 95% of the data.

solving the static RWA problem. We trained and evaluated two DRL agents, namely PPO-FF and PPO-Full. In particular, PPO-FF applies a first-fit strategy for wavelength assignment, whereas PPO-Full can accommodate requests in any available wavelength. Furthermore, by leveraging stochasticity in the learned policy, we developed a Multi-Start approach (PPO-FF-MSx8 and PPO-Full-MSx8), bringing a consistent improvement on the average performance of the DRL agents. Finally, we engineered a shaped reward that allows efficient learning in networks with high link capacity. From experimental results, we observed that PPO-FF outperforms PPO-Full in most cases, highlighting the significant challenges in terms of exploration and feature learning in DRL. Overall, our proposed approach shows pros and cons with respect to both greedy heuristics (i.e., longer computational times, but better solution quality) and state-of-the-art metaheuristics such as GA-FF (i.e., worse solution quality, but better computational times), providing a competitive middle ground between these two aspects. Future work will focus on learning more complex heuristic frameworks (e.g., local search) leveraging DRL-based methods.

ACKNOWLEDGMENT

The authors would like to thank Lily Friedberg for the insightful discussions about the paper.

REFERENCES

[1] C. Rottondi, L. Barletta, A. Giusti, and M. Tornatore, "Machine-learning method for quality of transmission prediction of unestablished lightpaths," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 2, pp. A286–A297, 2018.

[2] F. Musumeci, C. Rottondi, G. Corani, S. Shahkarami, F. Cugini, and M. Tornatore, "A tutorial on machine learning for failure management in optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4125–4139, 2019.

[3] S. Troia, R. Alvizu, Y. Zhou, G. Maier, and A. Pattavina, "Deep learning-based traffic prediction for network optimization," in *2018 20th International Conference on Transparent Optical Networks (ICTON)*, 2018, pp. 1–4.

[4] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," in *NIPS Deep Learning Workshop*, 2013.

[6] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 01 2016.

[7] OpenAI, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," 2019. [Online]. Available: https://arxiv.org/abs/1912.06680

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–33, 02 2015.

[9] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48, Jun 2016, pp. 1329–1338.

[10] G. Stampa, M. Arias, D. Sanchez-Charles, V. Muntés-Mulero, and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimization," *CoRR*, vol. abs/1709.07080, 2017. [Online]. Available: http://arxiv.org/abs/1709.07080

[11] P. Almasan, J. Suárez-Varela, A. Badia-Sampera, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, "Deep reinforcement learning meets graph neural networks: An optical network routing use case," *CoRR*, vol. abs/1910.07421, 2019. [Online]. Available: http://arxiv.org/abs/1910.07421

[12] X. Chen, B. Li, R. Proietti, H. Lu, Z. Zhu, and S. J. B. Yoo, "Deeprmsa: A deep reinforcement learning framework for routing, modulation and spectrum assignment in elastic optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4155–4163, 2019.

[13] B. Li and Z. Zhu, "Deepcoop: Leveraging cooperative drl agents to achieve scalable network automation for multi-domain sd-eons," in *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, 2020, pp. 1–3.

[14] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," 2019.

[15] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman, "Quantifying generalization in reinforcement learning," *CoRR*, vol. abs/1812.02341, 2018. [Online]. Available: http://arxiv.org/abs/1812.02341

[16] K. Wang, B. Kang, J. Shao, and J. Feng, "Improving generalization in reinforcement learning with mixture regularization," in *NeurIPS*, 2020. [Online]. Available: https://arxiv.org/pdf/2010.10814.pdf

[17] R. Ramaswami and K. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Transactions on Networking*, vol. 3, no. 5, pp. 489–500, 1995.

[18] K. Christodoulopoulos, K. Manousakis, and E. Varvarigos, "Offline

routing and wavelength assignment in transparent wdm networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1557–1570, 2010.
[19] X. Chu and B. Li, "Dynamic routing and wavelength assignment in the presence of wavelength conversion for all-optical networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 3, pp. 704–715, 2005.
[20] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.
[21] J. Suarez-Varela, A. Mestres, J. Yu, L. Kuang, H. Feng, A. Cabellos-Aparicio, and P. Barlet-Ros, "Routing in optical transport networks with deep reinforcement learning," *Journal of Optical Communications and Networking*, vol. 11, no. 11, pp. 547–558, 2019.
[22] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.
[23] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," *CoRR*, vol. abs/1602.01783, 2016. [Online]. Available: http://arxiv.org/abs/1602.01783
[24] X. Chen, R. Proietti, C.-Y. Liu, and S. J. B. Yoo, "A multi-task-learning-based transfer deep reinforcement learning design for autonomic optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 9, pp. 2878–2889, 2021.
[25] X. Chen, R. Proietti, and S. J. B. Yoo, "Building autonomic elastic optical networks with deep reinforcement learning," *IEEE Communications Magazine*, vol. 57, no. 10, pp. 20–26, 2019.
[26] M. R. Raza, C. Natalino, P. Öhlen, L. Wosinska, and P. Monti, "Reinforcement learning for slicing in a 5g flexible ran," *Journal of Lightwave Technology*, vol. 37, no. 20, pp. 5161–5169, 2019.
[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html
[28] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2015.
[29] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," 2016.
[30] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," 2017.
[31] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992.
[32] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2018.
[33] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347
[34] A. Ozdaglar and D. Bertsekas, "Routing and wavelength assignment in optical networks." *IEEE/ACM Transactions on Networking*, vol. 11, pp. 259–272, 01 2003.
[35] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high bandwidth optical wan's," *IEEE Transactions on Communications*, vol. 40, no. 7, pp. 1171–1182, 1992.
[36] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html
[37] R. Martí, *Multi-Start Methods*. Boston, MA: Springer US, 2003, pp. 355–368.
[38] V. Antuori, E. Hebrard, M.-J. Huguet, S. Essodaigui, and A. Nguyen, "Leveraging reinforcement learning, constraint programming and local search: A case study in car manufacturing," in *International Conference on Principles and Practice of Constraint Programming*, 2020.
[39] C. Natalino and P. Monti, "The Optical RL-Gym: an open-source toolkit for applying reinforcement learning in optical networks," in *International Conference on Transparent Optical Networks (ICTON)*, July 2020, p. Mo.C1.1. [Online]. Available: https://github.com/carlosnatalino/optical-rl-gym
[40] O. Karandin, F. Musumeci, O. Ayoub, A. Ferrari, Y. Pointurier, and M. Tornatore, "Quantifying resource savings from low-margin design in optical networks with probabilistic constellation shaping," in *47th European Conference on Optical Communication (ECOC 2021)*, September 2021.
[41] M. Ibrahimi, O. Ayoub, O. Karandin, F. Musumeci, A. Castoldi, R. Pastorelli, and M. Tornatore, "Qot-aware optical amplifier placement in filterless metro networks," *IEEE Communications Letters*, vol. 25, no. 3, pp. 931–935, 2021.
[42] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2021. [Online]. Available: https://www.gurobi.com
[43] M. Andrychowicz, A. Raichuk, P. Stanczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What matters in on-policy reinforcement learning? A large-scale empirical study," *CoRR*, vol. abs/2006.05990, 2020. [Online]. Available: https://arxiv.org/abs/2006.05990
[44] K. R. Gabriel and R. R. Sokal, "A New Statistical Approach to Geographic Variation Analysis," *Systematic Biology*, vol. 18, no. 3, pp. 259–278, 09 1969.
[45] E. K. Çetinkaya, M. J. Alenazi, Y. Cheng, A. M. Peck, and J. P. Sterbenz, "On the fitness of geographic graph generators for modelling physical level topologies," in *2013 5th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2013, pp. 38–45.
[46] R. Romero Reyes and T. Bauschert, "Towards drl-based routing and spectrum assignment in optical networks: Lessons to be learned from markov decision processes," in *2021 IEEE Latin-American Conference on Communications (LATINCOM)*, 2021, pp. 1–6.

Nicola Di Cicco (S'22) is a Ph.D. Student with the Department of Electronic, Information and Bioengineering (DEIB) at Politecnico di Milano, Milan, Italy. His current research interests are in the field of deep learning and reinforcement learning for network optimization and automation.

Emre Furkan Mercan is currently a Field Test Engineer at umlaut, part of Accenture, in Aachen, Germany. He holds a M.Sc. degree in Telecommunications Engineering from Politecnico di Milano. His current research interests are in the field of communication networks optimization, with an emphasis on 5G NR network optimization and testing.

Oleg Karandin is a Ph.D. Student with the Department of Electronic, Information and Bioengineering (DEIB) at Politecnico di Milano, Milan, Italy. His current research interests are in the field of low-margin optical network design and machine learning for Quality-of-Transmission estimation.

Omran Ayoub is currently a researcher with the Department of Innovative Technologies (DTI) at Scuola Universitaria Professionale della Svizzera Italiana. His current research interests are in the field of network optimization and artificial intelligence and machine learning for communication networks.

Sebastian Troia is an Assistant Professor with the Department of Electronics, Information and Bioengineering (DEIB) at Politecnico di Milano, Milan, Italy. His current research interests are in the field of edge network softwarization and machine-learning for SDN and SD-WAN based networks.

Francesco Musumeci (S'11-M'12) is Assistant Professor with the Department of Electronics, Information and Bioengineering at Politecnico di Milano. His current research interests include Machine-Learning-assisted networking, design and optimization of optical networks, and network disaster resilience.

Massimo Tornatore (S'03–M'06–SM'13) is currently an Associate Professor at Politecnico di Milano. He also holds an appointment as Adjunct Professor at University of California, Davis, USA and as visiting professor at University of Waterloo, Canada. His research interests include performance evaluation, optimization and design of communication networks (with an emphasis on the application of optical networking technologies), cloud computing, and machine learning application for network management. In these areas, he co-authored more than 400 peer-reviewed conference and journal papers (with 19 best paper awards), 2 books and 1 patent. He is a member of the Editorial Board of IEEE Communication Surveys and Tutorials, IEEE Communication Letters, Springer Photonic Network Communications, and Elsevier Optical Switching and Networking.