CHALMERS
UNIVERSITY OF TECHNOLOGY

DeepDefrag: A deep reinforcement learning framework for spectrum defragmentation

DeepDefrag: A deep reinforcement learning framework for spectrum defragmentation

Ehsan Etezadi*, Carlos Natalino*, Renzo Diaz†, Anders Lindgren†, Stefan Melin†,
Lena Wosinska*, Paolo Monti*, and Marija Furdek*
*Department of Electrical Engineering, Chalmers University of Technology, 412 96 Gothenburg, Sweden
E-mail: {ehsanet, carlos.natalino, wosinska, mpaolo, furdek}@chalmers.se
† Telia AB, 169 94 Solna, Sweden.
E-mail: {renzo.z.diaz, anders.x.lindgren, stefan.melin}@teliacompany.com

***Abstract*—Exponential growth of bandwidth demand, spurred by emerging network services with diverse characteristics and stringent performance requirements, drives the need for dynamic operation of optical networks, efficient use of spectral resources, and automation. One of the main challenges of dynamic, resource-efficient Elastic Optical Networks (EONs) is spectrum fragmentation. Fragmented, stranded spectrum slots lead to poor resource utilization and increase the blocking probability of incoming service requests. Conventional approaches for Spectrum Defragmentation (SD) apply various criteria to decide when, and which portion of the spectrum to defragment. However, these polices often address only a subset of tasks related to defragmentation, are not adaptable, and have limited automation potential. To address these issues, we propose *DeepDefrag*, a novel framework based on reinforcement learning that addresses the main aspects of the SD process: determining when to perform defragmentation, which connections to reconfigure, and which part of the spectrum to reallocate them to. DeepDefrag outperforms the well-known Older-First First-Fit (OF-FF) defragmentation heuristic, achieving lower blocking probability under smaller defragmentation overhead.**

***Index Terms*—Spectrum defragmentation, Reinforcement learning, Service blocking ratio.**

I. Introduction

The ongoing drastic growth in bandwidth-intensive applications with dynamic behaviour and high-performance requirements, including high-resolution video on demand, cloud computing, Internet of Things applications, and content delivery networks, strains the optical backbone networks. To satisfy these requirements in a cost-efficient manner, the network must support dynamic, automated, and resource-efficient operations. Elastic Optical Networks (EONs) [1] are considered a future-proof solution to satisfy these needs due to their ability to allocate spectrum at a fine granularity that matches the spectrum requirements of various service requests served by lightpaths. However, spectrum fragmentation is a major challenge for the resource efficiency of dynamic EONs [2] where service requests can arrive and depart at any point in time.

Spectrum Fragmentation (SF) is generated by the dynamic departures of optical connections that leave relatively small, isolated, unused spectrum chunks scattered across the available fiber bandwidth [3]. Accommodating newly arriving service requests requires the availability of contiguous and continuous spectrum slots, which is not possible under fragmented spectrum conditions with a detrimental effect to Service Blocking Ratio (SBR).

Various Spectrum Defragmentation (SD) strategies have been proposed to help solve this issue. SD plays a crucial role in consolidating the spectrum usage, improving the utilization of the spectrum grid and reducing the SBR [4]. SD aims at reorganizing the spectrum allocation of different connections such that more (incoming) services can be accommodated, maximizing the spectrum use. The main tasks of SD in dynamic network scenarios entail: (i) deciding on the best time to perform SD, amidst arbitrary arrival and departures of service requests; (ii) deciding on the number and the order of distinct network connections to be reconfigured, and (iii) finding the alternative spectral resources and reallocating the reconfigured connections. Spectrum reallocation aimed at minimizing spectrum fragmentation by reconfiguring a minimum number of connections has been proven to be NP-complete already in the static scenario, where the set of connections does not change over time [5]. Traffic dynamicity further exacerbates the problem complexity since the set of connections present in the network constantly changes.

SD schemes can in general be classified as proactive or reactive [2]. Proactive schemes are executed regardless of whether the network is experiencing spectrum blocking, and are run either periodically or based on some threshold. Reactive schemes, on the other hand, are triggered by the blocking of service requests. SD can be performed by reallocating the spectrum only, or by combining it with connection rerouting. SD approaches that do not interrupt the running services are known as hitless [6]. Examples of hitless SD techniques include push-pull retuning [7], where the spectrum occupied by a connection to be re-allocated is first expanded to include the target spectrum as well, and then shrunk; and make-before-break [4], where a new connection is established over the target spectrum before the original one is torn down.

The main objective of SD is to decrease the SBR. However, frequent reallocation of a large number of connections is

This work is supported by Sweden's innovation agency VINNOVA, within the framework of the EUREKA cluster CELTIC-NEXT project AI-NET-PROTECT (2020-03506).

undesirable because of the extra reconfiguration overhead. Hence, SBR, the number of connection reallocations, and the number of SD cycles should be considered jointly in order to balance the benefits and drawbacks of SD. Moreover, SD methods should adapt to the changing network conditions to ensure the most appropriate set of actions at a given time. Existing SD approaches rely on, e.g., integer linear programming (ILP) models [8] or heuristic algorithms [9] that, guided by deterministic thresholds and policies, address a subset of the listed tasks. However, none of the prior solutions is able to address all of the aforementioned SD tasks simultaneously, and they require precise parametrization to achieve acceptable performance.

Driven by the need to automate complex networking problems, Reinforcement Learning (RL) has recently been demonstrated as a promising technique for, e.g., optical network slicing [10] and resource assignment [11], [12]. The key advantage of RL is that it leverages knowledge obtained by observing the environment to independently guide its decisions and maximize a long-term reward without being explicitly programmed to do so. Deep Reinforcement Learning (DRL) combines RL with deep neural networks, allowing to parameterize action policies and analyze complex systems for high-dimensional input data, such as traffic matrices.

In this paper, we propose *DeepDefrag*, a novel DRL-based framework that jointly addresses all of the aforementioned tasks associated with the SD process. DeepDefrag decides on the timing and composition of the SD actions, i.e., the number, order, and target spectrum for reconfigured connections. The proposed framework adapts to the network state to select the appropriate course of actions, and can also consider the priorities of the network operator such as minimizing the number of connection reallocations. We assess the performance of the framework through extensive simulations, demonstrating that DeepDefrag outperforms a state-of-the-art heuristic, i.e., Older-First First-Fit (OF-FF), algorithm in several aspects.

II. Related Work

A variety of approaches have been investigated in the literature to mitigate the impact of SF, including an ILP formulation to address proactive parallel SD in EONs [8], and heuristic algorithms for hitless bandwidth defragmentation [13]. ILP models and heuristic algorithms for three defragmentation techniques, denoted as Push-Pull, Hop-Tuning, and Replanning were proposed in [9]. Heuristic approaches from [14] use different service attributes to select the best connection to reallocate. The Older-First (OF), Bigger-First (BF), Longer-Lasting-First (LLF), and Longer-Path-First (LPF) algorithms use service age, size, remaining holding time, and path length to guide their proactive defragmentation decisions, respectively. A First-Fit (FF) spectrum assignment policy is then applied to reallocate the spectrum slots. In [15], the authors analyzed the performance of different SD algorithms such as lowest-slot-index-first, holding-time-aware, and proactive-reactive defragmentation in terms of blocking probability, entropy, and SF ratio. A trade-off between SD gain and the degree of lightpath disruptions was further investigated in [16], and a mathematical model is developed to optimize the fragmentation ratios over all links while taking into account both spectrum continuity and contiguity constraints.

Machine learning techniques have recently found a useful application in SD as well. An unsupervised machine learning technique for rearranging the fragmented spectrum based on lightpath clustering was presented in [17]. In [18], Elman neural networks were used to forecast traffic demands, and the spectrum was allocated using a two-dimensional rectangular packing model that reduces unnecessary fragmentation.

In [11], the authors proposed a DRL-based routing, modulation and spectrum assignment (RMSA) algorithm that decides on both routing and spectrum assignment concurrently, resulting in reduced blocking probability. The work in [19] modeled the connection admission control and Routing and Spectrum Assignment (RSA) problems as a Markov Decision Process (MDP), and defined the concept of deterministic policy for RSA problem in the policy iteration algorithm. The study in [20] highlighted DRL as a competitive alternative to established and well-known solutions when it comes to optimization problems in optical networks, e.g., Routing and Wavelength Assignment (RWA). A recent study in [21] applied DRL to solve the on-demand, reactive hitless SD problem. Upon an unsuccessful RMSA attempt, a DRL agent selects one of the pre-defined stretch schemes that extends the size of the fragmented spectrum to accommodate for blocked services. In spite of the strong potential of DRL in solving complex optical networking problems, benefits of this technique in addressing the SD problem remain to be assessed. To this end, we propose a novel, DRL-based framework for proactive SD and investigate the related challenges and network performance improvements.

III. Problem Formulation

We consider a scenario where an EON serves dynamic traffic. The network topology is represented by a graph $G(V, E)$, comprising a set of nodes V and a set of links E. The network receives service requests defined by $D_i(s_i, d_i, b_i, a_i, h_i)$, where s_i and d_i are the source and the destination nodes, b_i is the requested bit rate, while a_i and h_i are the arrival and the holding times. The network serves the service requests by assigning a physical route, a modulation format and spectral resources, i.e., by solving the RMSA problem. The required number of spectrum slots, denoted as n_i, is determined by the spectrum efficiency of the modulation format, which is related to the length of the selected path [21]. If a path with $n_i + 1$ continuous and contiguous spectrum slots is found (the extra slot accounts for the guardband), the connection is established and the request is served. Otherwise, the request is blocked.

To mitigate the impact of spectrum fragmentation on the SBR, we perform periodic defragmentation. Solving the SD problem means reallocating the spectrum used by the existing connections with the ultimate goal of consolidating the free spectrum available for future use. We consider a proactive defragmentation scenario where only spectrum reallocation is

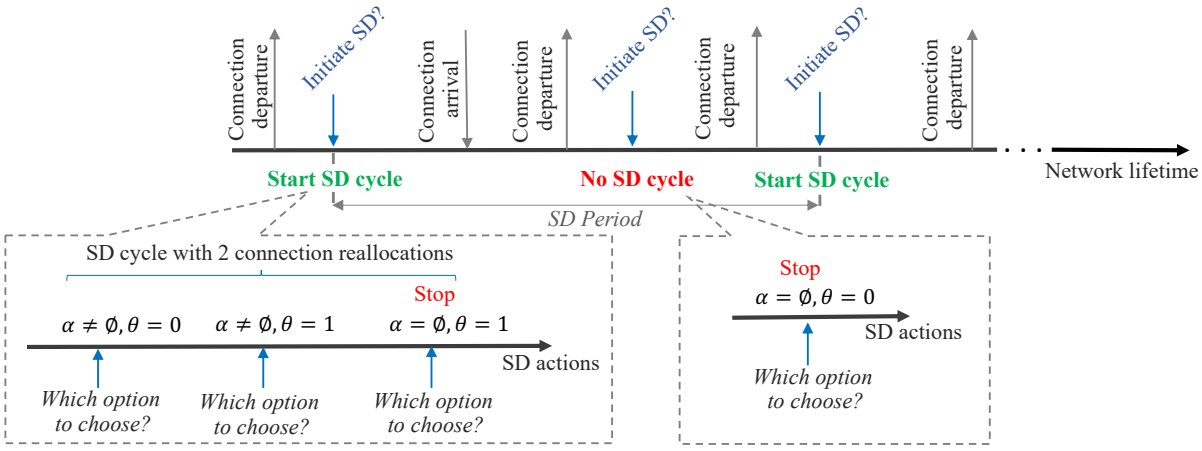

Fig. 1. The DeepDefrag scheme decisions taken and implemented during network operation.

possible, i.e., no rerouting is performed. When reallocating the spectrum of a connection, spectrum jump is allowed, i.e., the target and the original spectrum can be separated by slots occupied by other connections. We consider a hitless, make-before-break scenario.

The first challenge in the considered SD problem is to determine the best time to perform a defragmentation operation. At any point in time, the network snapshot consists of spectrum slots either occupied by existing connections, or free, possibly stranded due to fragmentation. The second challenge is to determine the set of connections to be reconfigured and the order of doing so, as well as to identify and allocate alternative spectrum slots to the connections. During SD, the standard spectrum continuity and contiguity constraints must hold.

IV. The DeepDefrag Scheme

A. System Model

Figure 1 illustrates the DeepDefrag scheme under dynamic traffic, highlighting the SD cycles triggered amidst service arrivals and departures. Upon each connection departure, DeepDefrag decides whether to initiate a defragmentation cycle or not. If a new SD cycle is initiated, DeepDefrag iteratively chooses a connection to reconfigure and the target spectrum to reallocate it to until the cycle ends. An example with SD cycle comprising two connection reallocations is shown in the left-hand inset in the bottom of the figure. The figure also shows two variables used by DeepDefrag. θ is a network control flag with value 0 when the network is not undergoing an SD cycle, and 1 when an SD cycle is in progress. The selected action α equals the index of the connection selected for re-allocation, while $\alpha = \emptyset$ represents the *stop* action.

For the example shown in the figure, $\theta = 0$ and $\alpha \neq \emptyset$ when the SD cycle starts and the first connection is reallocated. DeepDefrag can then choose to continue the current defragmentation cycle by reallocating another connection, or to stop by returning $\alpha = \emptyset$. In the example, DeepDefrag chooses to reallocate another connection, after which the SD cycle stops. The time between two successive SD cycles is referred to as SD period. The scheme can also choose not to initiate an SD cycle upon a connection departure. This is depicted in Fig. 1 upon the departure of the second connection, and the detailed actions and values of the decision variables are shown in the inset on the right hand side. Here, the algorithm decides not to take any action ($\alpha = \emptyset$), while a defragmentation cycle is not in progress ($\theta = 0$).

At each SD cycle, the DeepDefrag scheme considers a set of options, as illustrated in Fig. 2, with the snapshot of a small network example. The considered network state comprises six services established in the network, denoted as D_1 to D_6. Their routing is depicted in Fig. 2 a), while the spectrum assignment across the 12 available spectrum slots on each link is shown in Fig. 2 b). Connections considered eligible for reallocation are those using fragmented spectrum slots, which means that there is at least one free spectrum slot between them and their neighboring connections both at the lower and at the higher end of their used spectrum (considering that one guardband slot is a part of the spectrum allocated to each connection). In the example, only services D_1 and D_4 are eligible for reallocation.

DeepDefrag then considers several options for reallocating the spectrum of the eligible connections, as illustrated in Fig. 2 c). Each option represents reallocating one connection to the beginning or to the end of the existing free blocks along the path of that connection. For service D_1, two free blocks along links 1–2 and 2–4 can be considered for its allocation: slots 1–4 and 9–12. Therefore, service D_1 has four alternative spectrum options, which are at the beginning (denoted as o_1^1 and o_1^3) and at the end (o_1^2 and o_1^4) of the two candidate blocks. Alternatives for service D_4 are at the beginning and at the end of the only free block on links 1–3 and 3–4, i.e., slots 7–12, denoted as o_4^1 and o_4^2 in the figure. We use the event model from Fig. 1 and the intuition introduced in Fig. 2 to design a DRL agent that solves the SD problem.

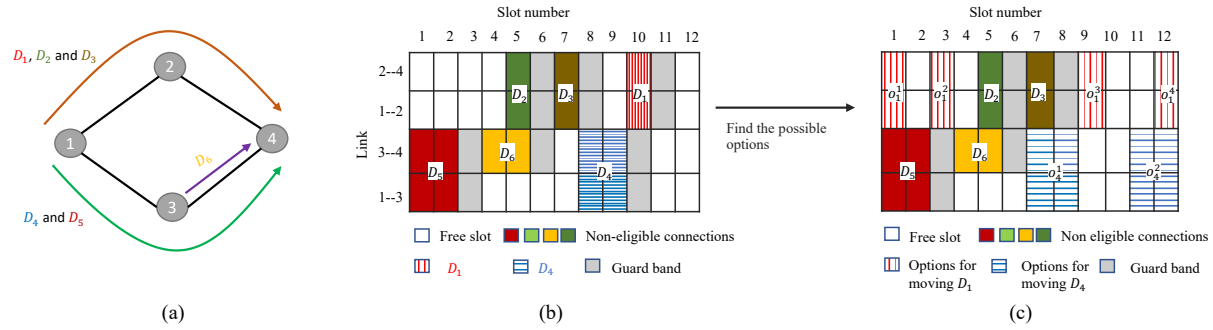

Fig. 2. A simple network example (a) with two connections eligible for defragmentation (b) and the different options for their spectrum reallocation (c).

B. Markov Decision Process Model

The DeepDefrag scheme proposed in this paper uses DRL to solve the SD problem introduced in the previous section. DRL is an area of machine learning concerned with intelligent agents that leverage deep learning to take actions in an observation environment with the goal of maximizing a cumulative reward. In the following, we present the MDP model of DeepDefrag, including the definitions of observation and action space, and reward function.

1) Observation Space: in DeepDefrag, the observation space exposes the current state of the network and the reallocation options to the agent. The agent observes the state of the environment and makes decisions based on the observations. Hence, the observation should reflect the critical aspects of the problem. The observation space of DeepDefrag has several components. $State_{ij} = (s_i, d_i, a_i, n_i, r_i, l_i, f_i, t_i, f_{ij}, t_{ij}, z_{ij})$ represents the set of attributes for reallocation option j of service D_i. Apart from the service attributes defined in Sec. III, the environment is characterized by the remaining time of the service r_i, the number of links l_i along the path allocated to the service, the currently assigned starting spectrum slot f_i, and the total number of available slots along path t_i. f_{ij} and t_{ij} represent the new candidate starting slot and the size of the free spectrum block used by option j for reallocating connection D_i, respectively. $z_{ij} \in \{0, 1\}$ indicates whether option j is at the beginning ($= 0$) or at the end ($= 1$) of the free block.

2) Action Space: the actions that can be selected by the agent are defined by the action space. In DeepDefrag, at each decision step, the agent selects one of the existing options. After processing the eligible connections (see Fig. 2 for details), each possible action in the action space is defined as a vector of elements (D_i, f_{ij}) for different eligible connections and reallocation options, plus $\emptyset$ that represents the *stop* action. When an SD cycle is not in progress, *stop* action means that there is no need to reallocate more connections.

3) Reward function: the reward function, defined by (1), measures the immediate gain achieved by each action taken by DeepDefrag.

$$Reward = \begin{cases} 1 - SBR & \theta = 0, 1 \wedge \alpha = \emptyset \\ 1 - SBR - Ps - Pe & \theta = 0 \wedge \alpha \neq \emptyset \\ 1 - SBR - Pe & \theta = 1 \wedge \alpha \neq \emptyset \end{cases}, \tag{1}$$

We encourage the agent to minimize SBR by adopting it as the main term of the function. The value of SBR refers to the ratio between the blocked and the total number of processed service requests. When no connection reallocation takes place ($\alpha = \emptyset$), the reward is equal to 1 – SBR to capture the objective of minimizing the blocking (the top term in (1)). To limit the number of SD cycles and reallocated connections, each new SD cycle and each connection reallocation is associated with a penalty, denoted with Ps and Pe, respectively. When the agent initiates the cycle by reallocating a connection, both penalties are applied (the middle term in (1)). When an SD cycle is in progress, each connection reallocation is penalized (the bottom term in (1)). Note that these penalty values can be set based on the cost incurred by the network operator at each reallocation instance.

V. Simulation Settings

To evaluate the performance of DeepDefrag, we carry out simulations of a dynamic traffic scenario and assess the value of SBR, as well as the reconfiguration actions' frequency and volume. We use the NSFNET topology with 14 nodes and 22 links, each supporting 320 spectrum slots. Service requests are generated based on a Poisson process. We set the traffic load to 170 Erlang to achieve approximately 10% SBR for the scenario without defragmentation. 80% of service requests is long-lived with an average holding time of 25 time units and exponential distribution, while the remaining 20% of requests have an average holding time of 12.5 time units. The considered bit rate is 100 Gbit/s for 50%, 200 Gbit/s for 30%, and 400 Gbit/s for the remaining 20% of the requests. BPSK, QPSK, 8-QAM, and 16-QAM modulation formats are utilized with a maximum reach length of 10000 km, 2000 km, 1250 km, and 625 km, and with slot capacity of 12.5 Gbit/s, 25 Gbit/s, 37.5 Gbit/s, and 50 Gbit/s, respectively [12]. The transmission reach of the signal determines the candidate

modulation formats, and the one with the highest spectral efficiency is selected. Shortest available route (among 5 pre-computed shortest paths) and first-fit spectrum assignment are used to obtain the RMSA solutions for all considered scenarios.

The performance of DeepDefrag is evaluated through comparison with three heuristic algorithms denoted as OF-FF, RND, and *No-SD*. In the OF-FF strategy, the set of eligible connections is defined according to their age, such that the longest-running connections are reconfigured first. First-fit spectrum allocation is then used to find new spectrum slots for the reconfigured connections. This strategy is used for benchmarking purposes since studies show that it performs very well in terms of SBR [14]. OF-FF has two parameters: the SD period, i.e., the number of request arrivals between two defragmentation periods, and the number of connections to be reallocated at each cycle. The values of both parameters are fixed throughout the network lifetime. We analyze the performance of OF-FF under different configurations and report on two most representative settings that enable a fair comparison with DeepDefrag. Configuration where both the SD period and the number of reallocations are set to 10, denoted as OF-FF(10,10), obtains the same SBR as DeepDefrag, allowing us to compare their defragmentation overheads. Configuration with the SD period length of 20 and the number of reallocations equal to 4, denoted as OF-FF(20,4), has the same defragmentation overhead as DeepDefrag, allowing us to examine their SBR. The random heuristic RND randomly selects one of the options from the action space (including *stop*). Finally, the *No-SD* approach reveals the network performance when defragmentation is not undertaken.

We implement the DeepDefrag scheme and environment extending the Optical RL-Gym [22], a framework for creating RL environments that model optical network problems such as resource management and reconfiguration. Stable-Baselines3 [23], an open-source implementation of DRL algorithms in Python, is applied to train the RL agent. We use the Deep Q-Networks algorithm (DQN) [24] with a learning rate of $5 * 10^{-6}$ and a discount factor of 0.95. The adopted neural network has 5 layers with 256 neurons each. The penalty factors Ps and Pe are set to 0.3 and 0.05, respectively, to model a higher cost of an SD cycle initiation than a connection reallocation. The episode length is set to 200 decision steps, and the training is performed over 9000 episodes. Results presented in the next section are obtained by assessing the performance of the agent as it is trained. For statistical purposes, in the following comparison comments, we average the results over the last 500 episodes.

VI. Numerical Results

Figure 3 shows the SBR values for the different schemes, indicating advantages of DeepDefrag. Considering the rolling 500-episode average, DeepDefrag lowers the blocking rate by 10% compared to the *No-SD* scenario with no defragmentation. The OF-FF(10,10) and OF-FF(20,4) schemes yield on average 10.8 % and 3% lower SBR than *No-SD*, which is aligned

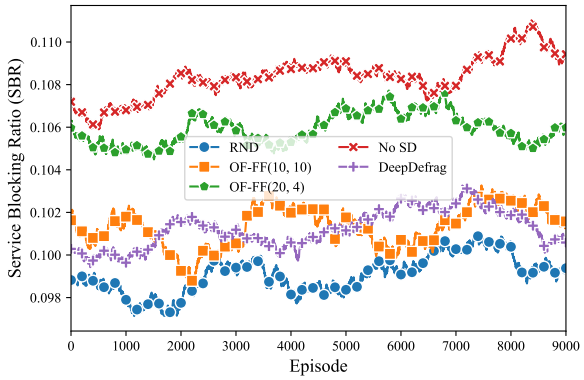

Fig. 3. Service Blocking Ratio (SBR) obtained by the different Spectrum Defragmentation (SD) schemes

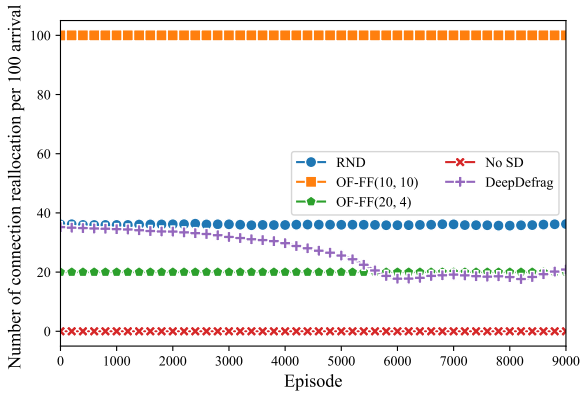

Fig. 4. Number of connections reallocated by the different SD schemes per 100 arrivals

with the result reported by [14]. Compared to OF-FF(20,4), which has the same defragmentation overhead, DeepDefrag reduces the SBR by 6.2%. This confirms the efficiency of defragmentation actions performed by DeepDefrag in reducing the SBR.

Figure 4 depicts the number of connection reallocations per 100 arrivals for the different strategies. On average, upon convergence, DeepDefrag reallocates only 20.2 connections per 100 request arrivals, closely matching OF-FF(20,4). Moreover, DeepDefrag reallocates 80% connection fewer than OF-FF(10,10), which is the OF-FF configuration with the same SBR as DeepDefrag.

Figure 5 shows the number of SD cycles. Also in this case, DeepDefrag outperforms all the benchmark SD heuristics, triggering only 4.9 SD cycles per 100 request arrivals on average. This is a 51% reduction compared to OF-FF(10,10). After analyzing the learning aspects in Figs. 4 and 5, we can see that DeepDefrag learns how to reduce the SD overhead in terms of connection reallocations and defragmentation cycles in 5500 training episodes, as indicated by the decline in the reconfiguration frequency and volume. As the above analysis shows, DeepDefrag outperforms the considered SD heuristics in all examined metrics.

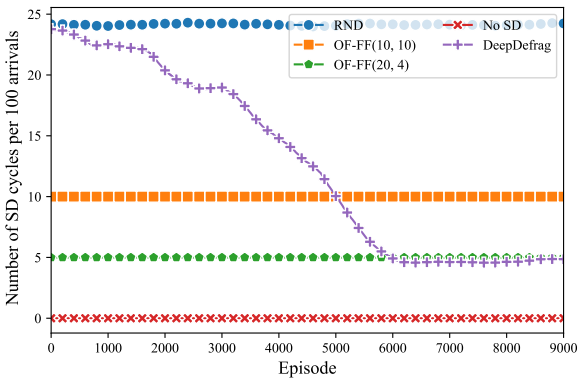

Fig. 5. Number of SD cycles for different SD schemes per 100 arrivals

VII. Conclusions

This paper proposes DeepDefrag, a novel framework based on the Deep Reinforcement Learning (DRL) that addresses several aspects of the Spectrum Defragmentation (SD) problem in an integrated manner. It determines whether and when to perform SD, which connections to reallocate and in which order, and finds new spectrum to be used by the connection. Simulation results indicate the ability of DeepDefrag to efficiently reduce the blocking rate while using fewer SD cycles and reallocating a lower number of connections than the state-of-the-art heuristic approaches, demonstrating its applicability to dynamic network conditions and strong potential for automating SD.

References

[1] Masahiko Jinno, Hidehiko Takara, Bartlomiej Kozicki, Yukio Tsukishima, Yoshiaki Sone, and Shinji Matsuoka. Spectrum-efficient and scalable elastic optical path network: architecture, benefits, and enabling technologies. *IEEE Communications Magazine*, 47(11):66–73, 2009.

[2] Rui Wang and Biswanath Mukherjee. Provisioning in elastic optical networks with non-disruptive defragmentation. *Journal of Lightwave Technology*, 31(15):2491–2500, 2013.

[3] Yawei Yin, Huan Zhang, Mingyang Zhang, Ming Xia, Zuqing Zhu, Stefan Dahlfort, and SJ Ben Yoo. Spectral and spatial 2D fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks. *Journal of Optical Communications and Networking*, 5(10):A100–A106, 2013.

[4] Bijoy Chand Chatterjee, Seydou Ba, and Eiji Oki. Fragmentation problems and management approaches in elastic optical networks: A survey. *IEEE Communications Surveys & Tutorials*, 20(1):183–210, 2017.

[5] Seydou Ba, Bijoy Chand Chatterjee, and Eiji Oki. Defragmentation scheme based on exchanging primary and backup paths in 1+1 path protected elastic optical networks. *IEEE/ACM Transactions on Networking*, 25(3):1717–1731, 2017.

[6] Mingyang Zhang, Yawei Yin, Roberto Proietti, Zuqing Zhu, and S. J. B. Yoo. Spectrum defragmentation algorithms for elastic optical networks using hitless spectrum retuning techniques. In *2013 Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC)*, pages 1–3, 2013.

[7] Yasuhiko Aoki, Xi Wang, Paparao Palacharla, Kyousuke Sone, Shoichiro Oda, Takeshi Hoshida, Motoyoshi Sekiya, and Jens C. Rasmussen. Dynamic and flexible photonic node architecture with shared universal transceivers supporting hitless defragmentation. In *2012 38th European Conference and Exhibition on Optical Communications*, pages 1–3, 2012.

[8] Mingyang Zhang, Changsheng You, and Zuqing Zhu. On the parallelization of spectrum defragmentation reconfigurations in elastic optical networks. *IEEE/ACM Transactions on Networking*, 24(5):2819–2833, 2016.

[9] Daniela Aguiar Moniz. Replanning of optical networks based on defragmentation techniques. 2015.

[10] Muhammad Rehan Raza, Carlos Natalino, Peter Öhlen, Lena Wosinska, and Paolo Monti. Reinforcement learning for slicing in a 5G flexible RAN. *Journal of Lightwave Technology*, 37(20):5161–5169, 2019.

[11] Masayuki Shimoda and Takafumi Tanaka. Mask RSA: End-to-end reinforcement learning-based routing and spectrum assignment in elastic optical networks. In *2021 European Conference on Optical Communication (ECOC)*, pages 1–4, 2021.

[12] Xiaoliang Chen, Baojia Li, Roberto Proietti, Hongbo Lu, Zuqing Zhu, and S. J. Ben Yoo. DeepRMSA: A deep reinforcement learning framework for routing, modulation and spectrum assignment in elastic optical networks. *Journal of Lightwave Technology*, 37(16):4155–4163, 2019.

[13] Mingyang Zhang, Yawei Yin, Roberto Proietti, Zuqing Zhu, and SJ Ben Yoo. Spectrum defragmentation algorithms for elastic optical networks using hitless spectrum retuning techniques. In *Optical Fiber Communication Conference*, pages OW3A–4. Optical Society of America, 2013.

[14] Jaume Comellas, Laura Vicario, and Gabriel Junyent. Proactive defragmentation in elastic optical networks under dynamic load conditions. *Photonic Network Communications*, 36(1):26–34, 2018.

[15] Sergio Fernández-Martínez, Benjamin Baran, and Diego P Pinto-Roa. Spectrum defragmentation algorithms in elastic optical networks. *Optical Switching and Networking*, 34:10–22, 2019.

[16] Nguyn Tun Khi, Ronald Romero Reyes, and Thomas Bauschert. Spectrum defragmentation with improved lightpath migration scheme in flex-grid networks. In *2021 International Conference on Optical Network Design and Modeling (ONDM)*, pages 1–6, 2021.

[17] Silvana Trindade and Nelson L.S. da Fonseca. Machine learning for spectrum defragmentation in space-division multiplexing elastic optical networks. *IEEE Network*, 35(1):326–332, 2021.

[18] Yu Xiong, Yaya Yang, Yulong Ye, and George N. Rouskas. A machine learning approach to mitigating fragmentation and crosstalk in space division multiplexing elastic optical networks. *Optical Fiber Technology*, 2019.

[19] Ronald Romero Reyes and Thomas Bauschert. Towards DRL-based routing and spectrum assignment in optical networks: Lessons to be learned from Markov decision processes. In *IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6, 2021.

[20] Nicola Di Cicco, Emre Furkan Mercan, Oleg Karandin, Omran Ayoub, Sebastian Troia, Francesco Musumeci, and Massimo Tornatore. On deep reinforcement learning for static routing and wavelength assignment. *IEEE Journal of Selected Topics in Quantum Electronics*, 28(4):1–12, 2022.

[21] Ruoxing Li, Rentao Gu, Weiqi Jin, and Yuefeng Ji. Learning-based cognitive hitless spectrum defragmentation for dynamic provisioning in elastic optical networks. *IEEE Communications Letters*, 25(5):1600–1604, 2021.

[22] Carlos Natalino and Paolo Monti. The Optical RL-Gym: An open-source toolkit for applying reinforcement learning in optical networks. In *22nd International Conference on Transparent Optical Networks (ICTON)*, 2020.

[23] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.