Cooperative Multi-Agent Deep Reinforcement Learning for Resource Management in Full Flexible VHTS Systems

Flor G. Ortiz-Gomez, Daniele Tarchi, *Senior Member, IEEE*, Ramón Martínez, Alessandro Vanelli-Coralli, *Senior Member, IEEE*, Miguel A. Salas-Natera, and Salvador Landeros-Ayala

***Abstract*—Very high throughput satellite (VHTS) systems are expected to have a huge increase in traffic demand in the near future. Nevertheless, this increase will not be uniform over the entire service area due to the non-uniform distribution of users and changes in traffic demand during the day. This problem is addressed by using flexible payload architectures, which allow the allocation of payload resources flexibly to meet the traffic demand of each beam, leading to dynamic resource management (DRM) approaches. However, DRM adds significant complexity to VHTS systems, so in this paper we discuss the use of one reinforcement learning (RL) algorithm and two deep reinforcement learning (DRL) algorithms to manage the resources available in flexible payload architectures for DRM. These algorithms are Q-Learning (QL), Deep Q-Learning (DQL) and Double Deep Q-Learning (DDQL) which are compared based on their performance, complexity and added latency. On the other hand, this work demonstrates the superiority a cooperative multiagent (CMA) decentralized distribution has over a single agent (SA).**

***Index Terms*—Bandwidth allocation, beamwidth allocation, power allocation, deep reinforcement learning, deep learning, dynamic resource management, flexible payload, multi-beam, cooperative multi-agent.**

NOMENCLATURE

s	State
S	Set of states
a	action
A	Set of actions
r	Reward function
s_t	Current state
a_t	Current action
s_{t+1}	Next state
p	transition probabilities
r_t	Immediate reward
r_{t+i}	Immediate reward at the $t+i$ time stamp
γ	Discount factor
R_t	Expected accumulated rewards
B	Number of beams
N	Number of agents. This paper considers $N = B$
$\overline{S}_t$	Set of current states of all agents
$\overline{A}_t$	Set of current action of all agents
D_t^b	Traffic demand in *b-th* beam at time instant t
C_t^b	Capacity offered in the *b-th* beam at time t
$EIRP_t^b$	EIRP in the *b-th* beam at time t
B_c	Number of beams per color
N_c	Number of colors of frequency plan
c	Color of frequency plan
$BW_t^{b_c}$	Bandwidth allocated to the *bc-th* beam of color c at time t
β_1	Weight of the error in the cost function
β_2	Weight of the error in the total *EIRP*
β_3	Weight of the error in the total bandwidth assigned
P_t^b	Power allocated to the *b-th* beam at time t
θ_t^b	Beamwidth allocated to the *b-th* beam at time t
$f_1(\cdot)$	Function to calculate C_t^b
$f_2(\cdot)$	Function to calculate $EIRP_t^b$
$P_{max,b}$	Maximum power per beam
$BW_{max,b}$	Maximum bandwidth per beam
$\theta_{max,b}$	Maximum beamwidth per beam
$P_{max,S}$	Maximum total system power
$BW_{max,C}$	Available bandwidth per color
SE_t^b	Spectral efficiency of the *b-th* beam at time t
G_t^b	Gain of the *b-th* beam at time t
d_t^b	traffic demand expected value over all the area inside *b-th* beam at time t
A_t^b	*b-th* beam area at time t
CNR_t^b	Carrier to Noise Ratio of the *b-th* beam at time t
CIR_t^b	Carrier to Interference Ratio of the *b-th* beam at time t

Manuscript received December 30, 2020; revised April 20, 2021; accepted June 4, 2021. Date of publication June 8, 2021; date of current version March 8, 2022. This work was supported by the Spanish Government, Ministerio de Economía, Industria y Competitividad through the National Program of Research, Development and Innovation within the FUTURE-RADIO project under Grant TEC2017-85529-C3-1-R and a Ph.D. scholarship provided by Universidad Politécnica de Madrid. The associate editor coordinating the review of this paper and approving it for publication was M. Erol-Kantarci. *(Corresponding author: Flor G. Ortiz-Gomez.)*

Flor G. Ortiz-Gomez, Ramón Martínez, and Miguel A. Salas-Natera are with the Universidad Politécnica de Madrid, 28040 Madrid, Spain (e-mail: fdg.ortiz@upm.es; ramon.martinez@upm.es; miguel.salas@upm.es).

Daniele Tarchi and Alessandro Vanelli-Coralli are with the Department of Electrical, Electronic and Information Engineering, University of Bologna, 40126 Bologna, Italy (e-mail: daniele.tarchi@unibo.it; alessandro.vanelli@unibo.it).

Salvador Landeros-Ayala is with the Agencia Espacial Mexicana, 01020 Mexico City, Mexico, and also with the Department of Telecommunications Engineering, Universidad Nacional Autónoma de México, 04360 Mexico City, Mexico (e-mail: slander@unam.mx).

Digital Object Identifier 10.1109/TCCN.2021.3087586

$CINR_t^b$	Carrier to Interference plus Noise Ratio of the *b-th* beam at time *t*
I_t^b	Co-channel power interference in *b-th* beam at time *t*
P_{co}	Power level of φ*-th* interference inside the *b-th* beam
φ	φ-th interferer spot
Φ	Total number of interfering beams of the beam *b*
K	Number of possible actions
δ_k	*k-th* possible action
V_b	Verification in the *b-th* beam
U_{max}	Maximum allowed of $\lvert C_t^b - D_t^b \rvert$
Z	Penalty received
F_1	Cost function of the DRM
P_{sat}	Verification parameter of power constraint on the satellite
BW_c	Verification parameter of bandwidth constraint on the color of frequencies plan
ρ	Amount of penalization
X_b	Movement that the *b-th* agent makes in the resource allocation space
μ_i	Priority that has to select *i-th* action
Q_b	Value of a pair (s_t^b, a_t^b) contains the sum of all these possible rewards
α	Learning rate
$Q_b{}'$	Temporal difference target of a pair (s_t^b, a_t^b)
M_b	*b-th* memory
ω_t	Current neural network parameters
ω_{t-1}	Previous neural network parameters
$\omega_t{}^-$	Current target neural network parameters
ε	Probability threshold
λ	Constant scaling factor for ε decay
$P_{Payload}$	Normalized payload power
$P_{Total,UPA}$	Total Payload Power when using a Uniform Power Allocation
$P_{Total,Alg}$	Total Payload Power when using the Power Allocation using the proposed algorithm
NCT	Normalized convergence time
ET_{alg}	Average time per episode of the algorithm
ET_{QL}	Average time per episode of the QL algorithm
NE_{alg}	Number of episodes in which the algorithm converges.

I. Introduction

VERY high throughput satellite (VHTS) systems have a fundamental role in the support of future 5G and broadcast terrestrial networks [1], [2]. VHTS systems exceed the capacity of traditional systems providing fixed and mobile satellite services using contoured regional footprints. VHTS aims to achieve a satellite Terabit/s data rate in the near future [3], based on multi-beam coverage with polarization schemes, frequency reuse and spectrum optimization [4].

VHTS systems currently provide uniform throughput over the service area; however, traffic demands are expected to be unevenly distributed over the service area, as the distribution of users is not uniform within the coverage. This will result in a system where some beams do not have the necessary capacity, i.e., they do not meet the traffic demands, while other beams exceed the required capacity or simply waste resources [5], [6]. On the other hand, operators claim that one of the main challenges in the design of future satellite broadband systems is how to increase revenues from satellites and, at the same time, meet unequal and dynamic traffic demands. In that sense, flexible payload is a promising solution to meet changing traffic demand patterns [3], [7]–[9].

Nowadays, most satellite communication (SatCom) payloads are not flexible in terms of bandwidth or coverage. Moreover, power flexibility can now be achieved using the on-board amplifier working point to be modified according to the transponder load. However, the most recent research interests have focused on the design of a new generation of flexible satellite payloads that allow dynamic resource management (DRM) [7]–[9]. In this sense, the SatCom uplink and downlink characteristics were analyzed in [10], as a function of dynamic spectrum allocation.

VHTS next generation systems will provide Terabit connections using advanced flexible payloads, allowing beam reorientation and reconfiguration, while allowing individual allocation of power per beam and bandwidth. Therefore, DRM techniques for SatCom will be a key for operators [11]. In that sense, Cocco *et al.* [12] represent the problem of radio resource allocation for VHTS as an objective function that minimizes the error between the capacity offered and the capacity required. Nevertheless, a thorough analysis of both the design of the payload architecture and resource management is required.

Kawamoto *et al.* [13] comment that resource allocation problem can be solved through optimization techniques but, on a larger scale, the number of resources to be managed, the limitations that come from the system and the huge number of traffic demand situations can give rise to a problem that cannot be solved with conventional techniques. On the other hand, Kisseleff *et al.* explain in [14] that the resource management problem in SatComs is, in most cases, nonlinear and nonconvex due to the logarithmic function as well as the nonlinear dependencies of the carrier to interferences plus noise ratio (CINR) on the optimization. In addition, the binary indicator of carrier assignment makes it a mixed-integer program. Therefore, an optimal solution cannot be obtained using known convex optimization methods. One could attempt to solve this problem by exhaustive search, but this strategy has a very high computational complexity, which is often beyond the capabilities of satellite processors in online operation.

As an alternative, Lei and Vázquez-Castro have proposed a suboptimal method [15], which addresses parts of the problem separately and then iteratively adjusts the parameters. The problem splitting is done in such a way that power allocation and carrier allocation are separated. However, if it adds more complexity to the system, such as beamwidth flexibility, the proposal is no longer feasible.

In addition, Liu *et al.* [16] suggested an assignment game-based dynamic power allocation (AG-DPA) to achieve a low suboptimal complexity in multibeam satellite systems. They compared the results obtained with a proportional power allocation (PPA) algorithm, obtaining a remarkable advantage in terms of power saving; however, resource management is still insufficient with respect to the required traffic demand since the error obtained between the capacity offered and the traffic demand in some cases is greater than 200 Mbps.

Regarding this, Ortiz-Gomez *et al.* [17]–[19] suggest solving the DRM problem using a Supervised Learning algorithm through a Classification Neural Network, in which the classes correspond to all possible configurations of payload resource allocation. The main advantage of this methodology is that the management is done with a low computational cost since the neural network training is performed offline. However, this methodology presents several challenges; one of them is the exponential dependence of the number of classes on the number of beams, in addition to the possible variations of power, bandwidth and/or beamwidth, which results in unsolvable problems due to the increase in flexibility. That is, as the number of beams and possible resources to be managed increases, the number of neurons required in the input and output layer increases considerably.

On the other hand, Ortiz-Gomez *et al.* [20] suggest a novel system model and a cost function that allows to get an optimal solution, by determining how to match resources to a demand pattern, while minimizing the resource consumption of satellites. The authors suggest a Convolutional Neural Networks (CNN) algorithm that allows the system to be implemented offline. In contrast to most approaches, the authors consider three possible flexible resources for the study of DRM, i.e., power, bandwidth, and beamwidth. The suggested CNN algorithm shows superiority in power management when compared to AG-DPA and PPA algorithms. However, one of the most important limitations of this proposed CNN in DRM is the dependence on the traffic model used during training. Thus, in a real system with changes in traffic behavior that do not fit the model, the CNN will have to be trained again.

Concerning the research suggesting deep reinforcement learning (DRL) algorithms to solve the DRM problem, Ferreira *et al.* [21], [22] stated that a feasible solution to the problems of real-time and single-channel resource allocation can be designed. However, in their study, the DRL architectures are based on the discretization of resources before their allocation, while satellite resources, such as energy, are inherently continuous. Therefore, Luis *et al.* [23] explore a DRL architecture for energy allocation that uses continuous and stateful action spaces, avoiding the need for discretization. However, the policy is not optimal, since some demand is still being lost. On the other hand, Liu *et al.* [24] suggest a novel dynamic channel assignment algorithm based on deep reinforcement learning (DRL-DCA) in multibeam satellite systems. The results proved that this algorithm can achieve a lower blocking probability, in comparison with traditional algorithms; nevertheless, the joint channel and power allocation algorithm are not taken into account.

Liao *et al.* [25] build a game model to learn the optimal strategy in the satellite communication scenario. The authors suggest a bandwidth allocation framework based on DRL, which can dynamically allocate the bandwidth in each beam to improve transmission efficiency. The effectiveness of the proposed method in time-varying traffic and large-scale communications is verified in the problem of bandwidth management with an acceptable computational cost. However, only one resource can be managed on the satellite with this method, leading to a critical limitation when looking for full flexibility in the VHTS system.

According to the current state of the art, the effectiveness of reinforcement learning algorithms to manage the resources of a multi-beam satellite has been demonstrated, although the algorithms proposed in the literature are only capable of single resource management in a multi-beam system [21]–[25]. However, a fully flexible payload must be able to manage at least three resources (i.e., power, bandwidth, and beamwidth). In this paper, we assume that as the available resources to manage in all the satellite beams increase, it becomes a very complicated problem for a single agent (SA). Therefore, in this work we propose to use the Cooperative Multi-Agent (CMA) approach to obtain a better performance and compare it with that of a SA to manage the three resources in all the satellite beams.

The main contributions of this work can be listed below:

- The DRM problem for VHTS systems is defined as a MDP (Markov Decision Process) that proposes a decentralized distribution of CMA (Cooperative Multiagent), that works cooperatively to achieve maximum reward.
- Different from the current state-of-the-art, thanks to the use of deep reinforcement learning (DRL) techniques, the proposed algorithm allows to achieve full flexibility in the multibeam satellite resources, i.e., power, bandwidth, and beamwidth.
- The state-of-the-art has demonstrated the effectiveness of using DRL in satellite DRM. However, being an online algorithm, the convergence time is very important and has not been evaluated in other proposals; in our proposals the convergence time is presented as a KPI of the system.
- One reinforcement learning (RL) and two DRL algorithms to manage the resources available in flexible payload architectures for DRM are suggested, i.e., Q-Learning (QL), Deep Q-Learning (DQL) and Double Deep Q-Learning (DDQL), and compared based on their performance, complexity, and additional latency.
- This work demonstrates the superiority a CMA distribution has over a single agent (SA) to DRM in multibeam SatComs systems.

The paper is organized as follows: Section II includes a background summary of reinforcement learning, Section III explains the full flexibility system model and problem definition, Section IV presents reformulating the problem as a CMA RL and CMA DRL problem, Section V describes the proposed DRL algorithm using a CMA distribution, Section VI presents the simulation results and the analysis of the case study and, finally, Section VII contains the conclusions of the study.

II. Reinforcement Learning Background

This section outlines the basic concepts of reinforcement learning and the algorithms used in this work; for additional details, the interested reader could refer to [26]–[29].

Reinforcement learning is an area of the artificial intelligence (AI) focused on identifying which actions should be taken to maximize the reward signal; in other words, it is concerned on how to map situations to actions that are focused on finding that reward.

In the DRL, the agent represents the hardware or software that must learn to perform a specific action; in that sense, the agent interacts with an "environment", which can be a real decision process or a simulation of it. The agent works observing the environment, making a decision and checking which effects it produces. If the outcome of that decision is beneficial, the agent automatically learns to repeat that decision in the future, while, if the outcome is harmful, it will avoid making the same decision again.

In this way, following a process of learning by conditioning similarly to that of living beings, the agent learns which decisions are most appropriate according to the situation, and develops long-term strategies that maximize benefits.

The "brain" or the learning capacity of the agent is given by a Machine Learning (ML) or a Deep Learning (DL) model. This allows the exploitation of all recent advances in artificial neural networks, thus being able to deal with problems that require the analysis of unstructured data.

In an episode that starts at instant *t* until reaching the last state of the sequence at instant *T*, the accumulated reward would be the sum of all the rewards of its states. The objective of the agent is not to maximize the immediate reward (of the following action), while its objective is to maximize the accumulated reward in each of the possible combinations of actions.

MDP provides a mathematical framework to model the interaction between the agent and the environment. MDP key target is a discrete time stochastic model whose evolution can be controlled over time. A stochastic process and a value function are associated with the control policy. The end goal is finding a "good" policy that solve the described problem [30]. For this reason, MDPs are useful for studying optimization problems solved via dynamic programming and RL.

The MDP contains a set of states $s \in S$, a set of actions $a \in A$, a reward function $r \in \mathbb{R}$, and a series of transition probabilities $p(s_{t+1}|s_t, a_t)$ of moving from the current state s_t to the next state s_{t+1} given an action a_t. The goal of an MDP is to find a policy that maximizes the expected accumulated rewards $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$, where r_{t+i} is the immediate reward at the $t + i$ time stamp and $\gamma \in [0, 1]$ is the discount factor. In this work we chose the non-model method, which means that there is no knowledge of the transition probabilities.

On the other hand, CMA systems are those in which several agents attempt, through their interaction, to jointly solve tasks or to maximize utility. A decentralized multi-agent environment is one in which there is more than one agent, in which the agents interact with each other and, in addition, in which there are constraints in that environment, so that the agents may not know at any given time everything that the other agents know about the environment (including the internal states of the agents themselves) [31].

Overall, cooperation among multiple RL agents is more critical, as multiple agents must collaborate to achieve a common goal, accelerate learning, protect privacy, provide resilience against failures and attacks, and overcome the physical behavior of each RL individual actions of each agent. Under this mechanism, each agent seeks to learn the best strategy to maximize the reward of the shared team, while working with the unknown random environment and the interaction of other agents. Compared with the SA case, the CMA is much more challenging, as the search space increases exponentially. In addition, the non-stationary and unpredictable environment is caused by agent concurrency and diversity behavior also brought many difficulties to CMA. These difficulties can be alleviated by proper coordination among agents to guarantee Nash equilibrium [32].

In the proposed cooperative environment, there is a global reward function and each agent will know the states and actions of all agents, each agent must meet the minimum requirements to achieve equilibrium in the system. The illustration of the CMA is shown in Fig. 1. Considering a multi-agent environment involving *N* agents, the *n-th* agent observes the state of the globally shared environment and independently selects an action to perform. Then, the current state is transformed into a new state. All the agents are in the same environment and have a common goal, so they work in cooperation to maximize the reward where $\overline{S}_t = \{s_t^1, s_t^2, \ldots, s_t^N\}$ represents the current states of the agents, and $\overline{A}_t = \{a_t^1, a_t^2, \ldots, a_t^N\}$ represents the actions.

III. System Model and Problem Statement

Fig. 2 shows the high-level architecture of the full flexible VHTS system implementing the CMA based resource management. It is assumed that the considered payload is capable of flexibly managing three types of resources, i.e., power, bandwidth, and beamwidth, in a similar way to [20].

The system is composed of *B* beams. The suggested system manages the communication resources in response to changes in traffic demand $\overline{D}_t = \{D_t^1, D_t^2, \ldots, D_t^B\}$ where D_t^b represents the traffic demand in *b-th* beam at time instant *t*.

The resource management training is supposed to be performed online, that is, every time the traffic requirements change the values are updated to retrain the DRM agents given the new conditions. For this reason, processing time will play a very important role in DRM performance.

The proposed system assumes that the manager is on board, the manager receives in input the traffic demand of the user beams through the return link, and then generates an optimal control through the Payload Control Center.

In the following, we consider the satellite to be a bent pipe transponder architecture. Satellite-gateway feeder link is not considered in the forward link because different technologies are considered to ensure the overall link budget, such as ULPC (Uplink Power Control), and gateway diversity [31], [34].

A. DRM Problem Statement

DRM must manage available resources to minimize the error between the capacity offered in each beam

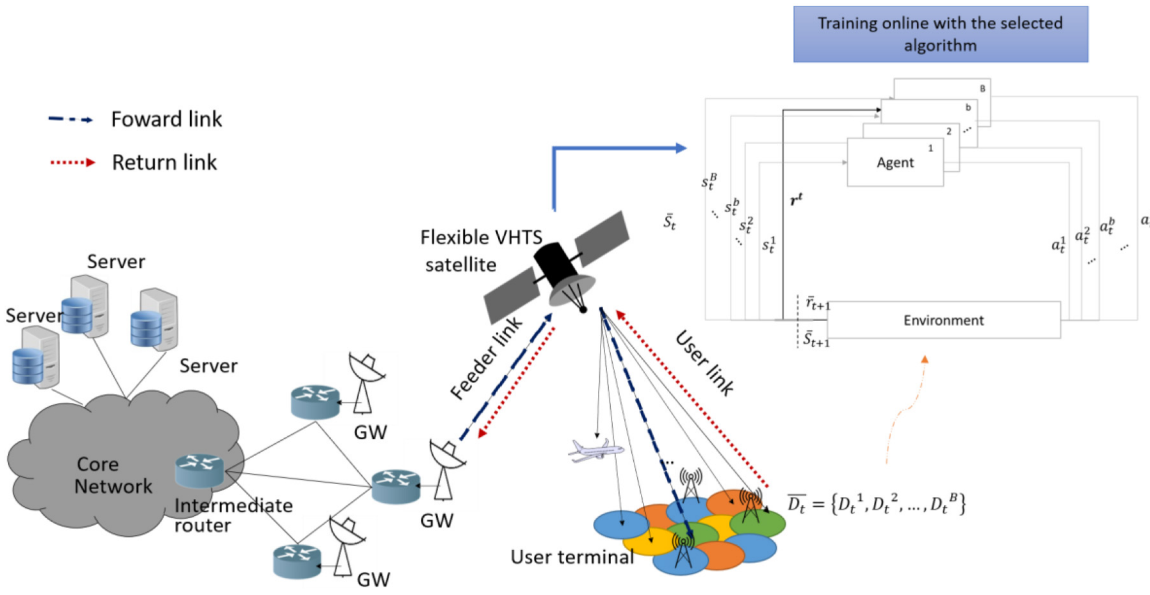


Fig. 1. Cooperative Multi-Agent Reinforcement Learning for resource management in a VHTS satellite: System model.

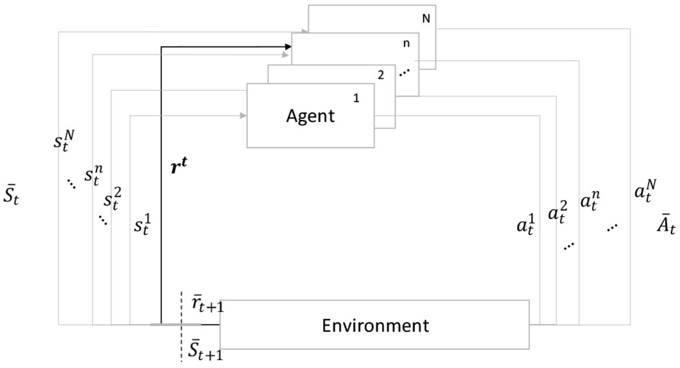


Fig. 2. Architecture of the CMA system: Agents share the same reward.

and the required traffic demand, and, at the same time, optimizing the resources used over time. In that sense, by recalling [20], the minimization of the DRM cost function can be defined as:

$$\min_{P_t^b, BW_t^{bc}, \theta_t^b} \quad F_1 \rightarrow F_1 = \frac{\beta_1}{B} \sum_{b=1}^{B} \left| C_t^b - D_t^b \right| + \frac{\beta_2}{B} \sum_{b=1}^{B} EIRP_t^b + \frac{\beta_3}{B} \sum_{c=1}^{N_c} \sum_{b_c=1}^{B_c} BW_t^{b_c} \tag{1}$$

where

$$C_t^b = f_1\left(P_t^b, BW_t^{bc}, \theta_t^b\right) \tag{2}$$

$$EIRP_t^b = f_2\left(P_t^b, \theta_t^b\right) \tag{3}$$

subject to

$$\begin{cases} C_t^b \geq D_t^b, & if \;\; P_t^b < P_{max,b}, \theta_t^b < \theta_{max} \\ & or \;\; BW_t^{bc} < BW_{max,b} \\ C_t^b = C_{max}, & if \;\; P_t^b = P_{max,b}, \theta_t^b = \theta_{max} \\ & and \;\; BW_t^{bc} = BW_{max,b} \end{cases} \tag{4}$$

$$\sum_{b=1}^{B} P_t^b \leq P_{max,S} \tag{5}$$

$$\sum_{b_c=1}^{B_c} BW_t^{bc} \leq BW_{max,c} \tag{6}$$

$$\theta_t^b \epsilon \{\theta_1, \theta_2, \ldots, \theta_{max}\} \;\forall\; b, t \tag{7}$$

where the capacity offered in the *b-th* beam at time *t*, C_t^b (in bps), should cope with D_t^b (in bps), the demand required in the *b*-th beam at time *t*.

The minimization of the DRM cost function defined in (1), similarly to [20], aims to minimize three parameters for each time instant, *t*. The first parameter is the error between the capacity offered and the demand required, where β_1 (in s/bit) is the weight of the error in the cost function. The second parameter to be minimized is the total *EIRP* (effective isotropic radiated power) assigned to all beams (in W), where β_2 (in 1/W) is the weight of the total *EIRP*; the third parameter to be minimized is the total bandwidth (in Hz) that is assigned to the beams of each color (BW_c) within the frequency plan, where N_c is the number of colors in the frequency plan and β_3 (in 1/Hz or s) is the weight of the total bandwidth assigned in each color of the frequency plan.

On the other hand, C_t^b can be calculated as $C_t^b = BW_t^{bc} \cdot SE_t^b$, where SE_t^b is the spectral efficiency of the modulation and coding scheme of a commercial reference modem used in the *b-th* beam over *t* [35]. As explained in [20], SE_t^b depends on the CINR in the *b-th* beam and in turn the CINR depends on the resources allocated in each beam ($P_t^b, BW_t^{bc}, \theta_t^b$), the CINR is obtained with the traditional link budget calculation. Therefore, C_t^b is a function $f_1(\cdot)$ of the power, bandwidth and beamwidth allocated to the *b-th* beam at time *t* (i.e., $P_t^b, BW_t^{bc}, \theta_t^b$, respectively) as seen in (2) [3], [20].

Moreover, the $EIRP_t^b$ depends on the P_t^b and the G_t^b, maximum gain of the *b-th* beam over *t*, as well as the G_t^b depends

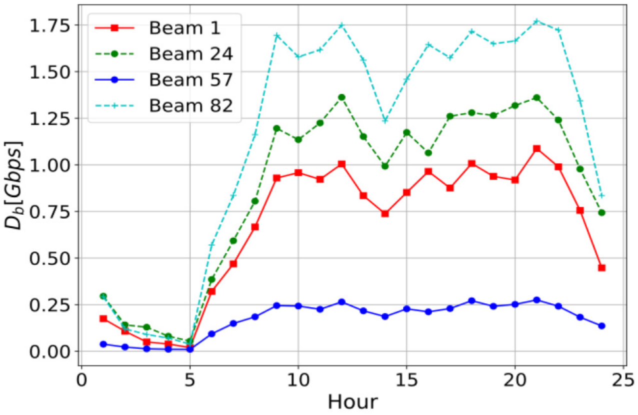


Fig. 3. One-day cycle of traffic demand per beam.

on the θ_t^b pattern as explained in [20]. Therefore, $EIRP_t^b$ is a function $f_2(\cdot)$ of the power and beamwidth assigned to the *b-th* beam at time *t* (3) [3], [20].

The cost function constraints are presented in (4)-(7). The minimum capacity restriction is shown in (4) where $C_t^b \geq D_t^b$ for each beam, provided that the power, beamwidth, or bandwidth assigned to the *b-th* beam at time *t* are less than the maximum allowed for each beam ($P_{max,b}, \theta_{max}$ and $BW_{max,b}$, respectively) [20]. In case the offered capacity cannot satisfy the limitation of the beam requirement, the offered capacity in the *b-th* beam shall be the maximum possible value.

Moreover, the overall power allocated to every beam must not be greater than the maximum total system power ($P_{max,S}$) (5), and the total bandwidth allocated in each color of the frequency plan must not be greater than the available bandwidth per color ($BW_{max,c}$) (6). In addition, the beamwidth of the *b-th* beam must belong to the set of possible configurations previously established (7). The selected beamwidths must meet the requirement of completely covering the entire service area [20].

In [20], [36] the authors propose a traffic model based mainly on the fact that the density of traffic demand per km^2 depends on the throughput per user (in bps/user), the population density (in inhabitant/km^2), the penetration rate (in user/inhabitant), and the concurrency rate that depends on the time of day. The authors in [36], in addition to the above, also consider the time zone in which the *b-th* beam is geographically located. In this sense, this work uses the traffic model considered by the authors in [36].

The traffic demand at *b-th* beam at time *t* is calculated as $D_t^b = d_t^b \cdot A_t^b$, where A_t^b is *b-th* beam area (in km^2) and which depends on the θ_t^b, it is to say greater beamwidth greater area, and d_t^b is traffic demand expected value over all the area inside *b-th* beam at time *t* (in bps/km^2) which is obtained with the study presented in [36]. In this sense, we obtain a behavior of the traffic demand as shown in Fig. 3.

The selected reference scenario corresponds to a multibeam coverage for Europe and the Mediterranean basin with 82 beams, similar to the coverage currently provided by KA-SAT [37]. In Fig. 3, the behavior of the traffic demand is depicted for 4 reference beams, out of the possible 82: Beam 1 is located centered on the geographic coordinates (40.95, −3.25) and provides coverage to Madrid, Spain, Beam 24 is located centered on the geographic coordinates (50.51, 39.4) and covers a portion of the central region of the European zone of Russia, Beam 57 is centered on the geographic coordinates of (40.11, 7.95) and covers the region of Sardinia, Italy, Beam 82 is centered on the geographic coordinates of (50.23, 11.4) and covers a portion of central Germany.

IV. Problem Reformulation as Cooperative Multi-Agent DRL

The DRM cost function can be reformulated as a CMA RL problem, for which the possible P_t^b, BW_t^{bc} and θ_t^b must be established. $C_t^b \in \{C_1, C_2, \ldots, C_{max,b}\}$ is calculated assuming that $P_t^b \in \{P_1, P_2, \ldots, P_{max,b}\}$, $BW_t^b \in \{BW_1, BW_2, \ldots, BW_{max,b}\}$ and $\theta_t^b \epsilon \{\theta_1, \theta_2, \ldots, \theta_{max}\}$. That is, the space of all the capacity values that can be assigned to the *b-th* beam depends on the resources allocated to the *b-th* beam, therefore a distribution like that in Fig. 4 is obtained, where it is observed that as the power and the bandwidth allocated to the *b-th* beam increase, the offered capacity increases forming a surface, and that by changing the beamwidth allocated to *b-th* beam the surface moves with respect to the axis of the capacity offered.

The offered capacity is related to the space of the possible resource allocations in the *b-th* beam and is obtained with the conventional link budget analysis [3], [20], where CNR_t^b, CIR_t^b and $CINR_t^b$ (Carrier to Noise Ratio, Carrier to Interference Ratio and Carrier to Interference plus Noise Ratio, respectively) depend on P_t^b, BW_t^{bc} and θ_t^b [3], [20]. A fixed availability is assumed, and the geographic position of the *b*-th beam is not considered. For the CIR_t^b, it is assumed that the co-channel power interference (the same color in the frequency plan) can be calculated as $I_t^b = \sum_{\varphi=1}^{\Phi} P_{co}(\phi, \theta_t^b)$, where φ represents the φ-th interferer spot, Φ is the total number of interfering beams of the beam *b*, and P_{co} is the power level (in W) of φ*-th* interference inside the *b-th* beam.

On the other hand, DRM training is performed through a DRL algorithm. A CMA distribution is used, where *N* equals *B* (number of beams). The *b-th* agent represents a manager of the resources of *b-th* beam. All agents have a joint objective, so they work in cooperation to maximize the accumulated reward R_t.

There are *B* agents sharing the same space of possible resource allocation, each agent manages the power, beamwidth, and bandwidth in each beam (Fig. 4).

The following parameters are defined to solve the DRM problem using a CMA distribution of DRL algorithm.

The environment at each time *t* is composed of $\overline{D}_t = \{D_t^1, D_t^2, \ldots, D_t^B\}$, representing the traffic demand in each of the *B* beams, $\overline{C}_t = \{C_t^1, C_t^2, \ldots, C_t^B\}$, the offered capacity in the *B* beams, $\overline{P}_t = \{P_t^1, P_t^2, \ldots, P_t^B\}$, the current power allocated in the *B* beams, $\overline{BW}_t = \{BW_t^1, BW_t^2, \ldots, BW_t^B\}$, the current bandwidth allocated in the *B* beams, and $\overline{\theta}_t = \{\theta_t^1, \theta_t^2, \ldots, \theta_t^B\}$ as current beamwidth allocated in the *B* beams. Moreover,

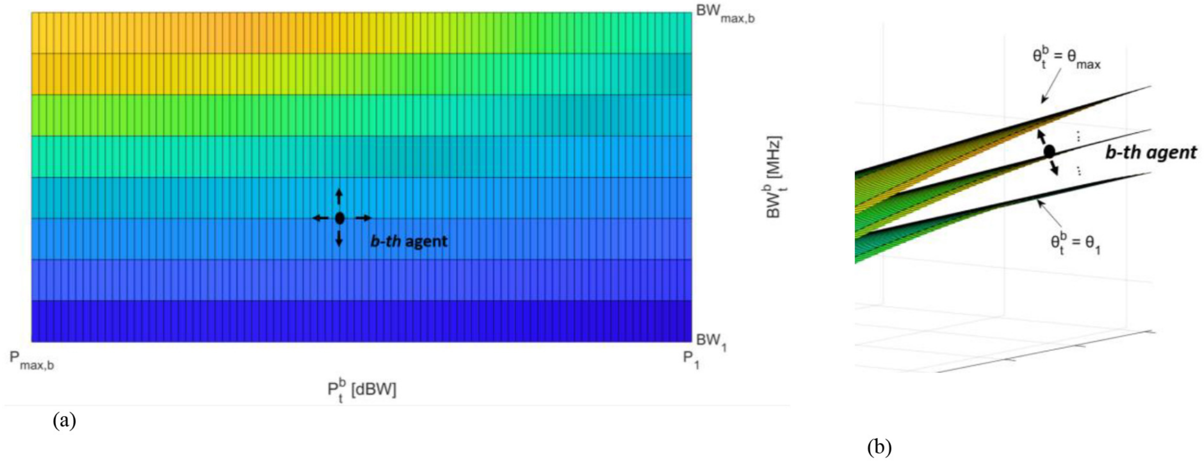


Fig. 4. Definition of possible movements of the b-th agent in the resource allocation space (a) power and bandwidth, (b) beamwidth.

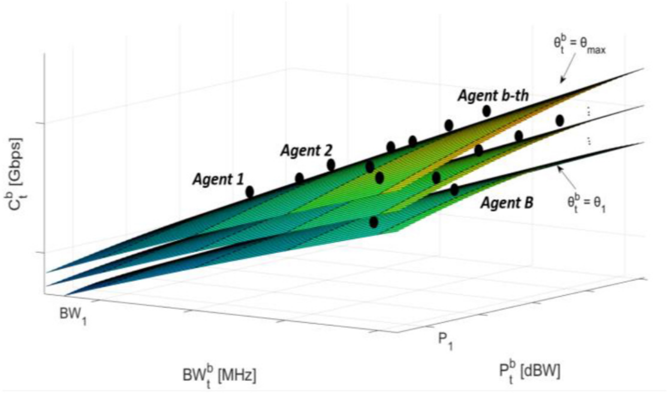


Fig. 5. Multi-Agent (B Agents).

1) $\overline{S_t} = \{s_t^1, s_t^2, \ldots, s_t^B\}$ represents the current states of the *B* agents, that is the current positions in space of possible allocated resources. That is, this parameter indicates the power, bandwidth and bandwidth currently allocated to each beam.
2) $\overline{A_t} = \{a_t^1, a_t^2, \ldots, a_t^B\}$ represents the action of each agent, that is, the movement it makes in the space of possible allocated resources, where:

$$a_t^b = \begin{cases} \delta_1, & increase\ in\ power \\ \delta_2, & decrease\ in\ power \\ \delta_3, & increase\ in\ bandwidth \\ \delta_4, & decrease\ in\ bandwidth \\ \delta_5, & increase\ in\ beamwidth \\ \delta_6, & decrease\ in\ beamwidth \\ \delta_7, & do\ nothing \end{cases} \quad (8)$$

is the space of the possible actions of the *b-th* agent, where $\delta_k \in$ R and $k \in \{1, 2, 3, 4, 5, 6, 7\}$ (Fig. 5).
3) The *B* agents share the same immediate reward r_t that is defined by:

$$r_t = \begin{cases} -h_1F_1 - h_2P_{sat} - h_3BW_c - h_4\sum_{b=1}^{B} X_b, & \prod_{b=1}^{B} V_b = 1 \\ -h_1F_1 - h_2P_{sat} - h_3BW_c - h_4\sum_{b=1}^{B} X_b - Z, & \prod_{b=1}^{B} V_b = 0 \end{cases} \quad (9)$$

subject to:

$$V_b = \begin{cases} 1, & \left|C_t^b - D_t^b\right| \leq U_{max} \\ 0, & \left|C_t^b - D_t^b\right| > U_{max} \end{cases} \quad (10)$$

$$P_{Sat} = \begin{cases} 0, \sum_{b=1}^{B} P_t^b \leq P_{max,S} \\ \rho, \sum_{b=1}^{B} P_t^b > P_{max,S} \end{cases} \quad (11)$$

$$BW_c = \begin{cases} 0, \sum_{b_c=1}^{B_c} BW_t^{bc} \leq BW_{max,c} \\ \rho, \sum_{b_c=1}^{B_c} BW_t^{bc} > BW_{max,c} \end{cases} \quad (12)$$

$$X_b = \begin{cases} \mu_1, & a_t^b = \delta_1\ or\ a_t^b = \delta_2 \\ \mu_2, & a_t^b = \delta_3\ or\ a_t^b = \delta_4 \\ \mu_3, & a_t^b = \delta_5\ or\ a_t^b = \delta_6 \\ \mu_4, & a_t^b = \delta_7 \end{cases} \quad (13)$$

It is required that each agent will seek its own benefit even if the immediate reward, r_t, is the same for all agents. For that reason, r_t is conditioned to the constraint that $\prod_{b=1}^{B} V_b$ is either 1 or 0 (9), where V_b has to be equal to 1 if the absolute difference between the traffic demand and the offered capacity of the *b-th* beam is less than or equal to the maximum allowed, U_{max} (10). Thus, guaranteeing that, although agents work cooperatively to maximize a shared reward, each agent has to seek its own benefit.

In (9), F_1 is the cost function of the DRM (1) which is a function of the resources allocated and will depend on the actions of the *B* beams, $\overline{A_t}$. P_{Sat} is a power constraint on the satellite, BW_c is a bandwidth constraint on the color of frequencies plan and ρ is a positive value that decreases the immediate reward when agents default on power and/or bandwidth constraint. X_b is defined by the movement that the *b-th* agent makes in the resource allocation space and μ_1, μ_2, μ_3, and μ_4 are the priority that has the resource allocation, depending on the payload features, it has a priority or another. That is, if power, bandwidth, and beamwidth have the same priority in the system, and μ_1, μ_2, μ_3, and μ_4 will have the same value, otherwise the weights should be different in order to receive a higher reward when the agent increases or decreases the resource with higher priority. In addition, h_1, h_2, h_3, and h_4

Algorithm 1 DRM Algorithm Online (It is Performed Every Time the Traffic Demand Changes)

```
-   Observe the current environment:
        $\overline{D}_t = \left\{D_t^1, D_t^2, \ldots, D_t^B\right\}$ as traffic demand in the B beams, $\overline{C}_t = \left\{C_t^1, C_t^2, \ldots, C_t^B\right\}$ as offered capacity in the B beams over time t, $\overline{P}_t = \left\{P_t^1, P_t^2, \ldots, P_t^B\right\}$ as current power allocated in the B beams, $\overline{BW}_t = \left\{BW_t^1, BW_t^2, \ldots, BW_t^B\right\}$ as current bandwidth allocated in the B beams, and $\overline{\theta}_t = \left\{\theta_t^1, \theta_t^2, \ldots, \theta_t^B\right\}$ as current beamwidth allocated in the B beams
-   Train the DRL algorithm with the current conditions
-   Obtain policies for agents
-   Update Resources:
        $\overline{P}_{t+1} = \left\{P_{t+1}^1, P_{t+1}^2, \ldots, P_{t+1}^B\right\}$ as new power allocated in the B beams, $\overline{BW}_{t+1} = \left\{BW_{t+1}^1, BW_{t+1}^2, \ldots, BW_{t+1}^B\right\}$ as new bandwidth allocated in the B beams, $\overline{\theta}_{t+1} = \left\{\theta_{t+1}^1, \theta_{t+1}^2, \ldots, \theta_{t+1}^B\right\}$ as current beamwidth allocated in the B beams.
-   Update values:
        $\overline{P}_t \leftarrow \overline{P}_{t+1}$, $\overline{BW}_t \leftarrow \overline{BW}_{t+1}$ and $\overline{\theta}_t \leftarrow \overline{\theta}_{t+1}$
```

Algorithm 2 Q-Learning

```
-   Set values for learning rate $\alpha$, and $\gamma$
Arbitrary initiation of the B Q-Tables
-   Repeat for each episode, do
        ○ Initialize $\overline{S_t} = \left\{s_t^1, s_t^2, \ldots, s_t^B\right\}$
        ○ Repeat for each step of episode, do
              • Each agent
                    • Chose $a_t^b$ from $s_t^b$ using policy derived from $Q_b$
                    • Take action $a_t^b$, observe $r_t$ and $s_{t+1}^b$
                    • Update values using:
$Q_b\left(s_t^b, a_t^b\right) \leftarrow Q_b\left(s_t^b, a_t^b\right)$
              $+ \alpha\left[r_t\left(s_t^b, a_t^b\right) + \gamma \max_{a_t^b} Q_b\left(s_{t+1}^b, a_t^b\right) - Q_b\left(s_t^b, a_t^b\right)\right]$
              $s_t^b \leftarrow s_{t+1}^b$
              • End do
        ○ End do
-   End do
```

Algorithm 3 Deep Q-Learning

```
-   Set values for learning rate $\alpha$, and $\gamma$ and $M$ which is the set of $B$ replay memories $\{M_1, M, \ldots, M_B\}$, where $M_b$ represents the b-th replay memory
-   Initialize $\omega_t$ and Q-values with random weights
-   Repeat for each episode, do
        ○ Initialize $\overline{S_t} = \{s_t^1, s_t^2, \ldots, s_t^B\}$
        ○ Repeat for each step of episode, do
              • Each agent
                    • With probability $\varepsilon$ select a random action $a_t^b$, otherwise select $a_t^b = \arg\max_{a_t^b} {Q_b}'(s_t^b, a_t^b, \omega_t)$
                    • Take action $a_t^b$, observe $r_t$ and $s_{t+1}^b$
                    • Store transition $(s_t^b, a_t^b, r_t, s_{t-1}, done)$ in expereince replay memory $M_b$.
                    • Sample random minibatch of transitions $(s_t^b, a_t^b, r_t, s_{t-1})$ from $M_b$.
                    • For every transition in minibatch, do
          $${Q_b}' = \begin{cases} r_t(s_t^b, a_t^b, \omega_{t-1}), \ \textit{if done} \\ r_t(s_t^b, a_t^b, \omega_{t-1}) + \gamma \max_{a_t^b} {Q_b}'(s_{t+1}^b, a_t^b, \omega_{t-1}), \ \text{else} \end{cases}$$
                    • End do
                    • Calculate the loss
          $$L_b(s_t^b, a_t^b, \omega_t) = \left({Q_b}' - Q_b(s_t^b, a_t^b, \omega_t)\right)^2$$
                    • Update $Q_b$ using gradient descent by minimizing the loss $L_b(s_t^b, a_t^b, \omega_t)$
              • End do
        ○ End do
-   End do
```

are positive values and represent the weights of each parameter in the immediate reward. *Z* is the penalty received when at least one agent has an absolute difference between the traffic demand and the offered capacity of the *b-th* beam greater than the maximum required.

V. Cooperative Multi-Agent DRL Based on Dynamic Resource Management Algorithm

The problem can be solved using two algorithms: on the one hand the DRM algorithm to manage resources (Algorithm 1) and on the other hand the algorithm used to train agents every time there are changes in traffic requirements. Since the training is online, the convergence time of the algorithm becomes very important for the DRM system. Based on this, three different algorithms, i.e., Q-Learning (QL), Deep Q-Learning (DQL) and Double Deep Q-Learning (DDQL) outlined in the following in the Algorithm 2, Algorithm 3 and Algorithm 4, respectively, are proposed.

The DRM algorithm (Algorithm 1) first observes the current environment, which is represented by the traffic demand and the capacity offered depending on the distribution of resources. With the acquired observation data, the agents are trained using a RL or DRL algorithm to obtain the agent policies and update the resources. This is repeated at every instant of time that the traffic demand changes.

The QL and DRL goal is to extract which actions should be chosen in the different states to maximize the reward. In a way, we seek that the *B* agents learn what is called a policy, which formally we can see as an application that tells each agent what action to take depending on its current state [26]. The policy of the agents is divided in two components: on one side, how each agent believe that an action refers to a determined state, and on the other side, how the agent uses what knows to choose one of the possible actions.

In the QL algorithm (Algorithm 2), the Q_b value of a pair (s_t^b, a_t^b) contains the sum of all these possible rewards. If the *b-th* agent knows a priori the Q_b values of all possible pairs (s_t^b, a_t^b) it could use this information to select the appropriate action for each state. In that sense, there is a Q-Table (Figure 6) for each agent that represents a matrix of size JxK where *J* is the number of possible states of the *b-th* agent and K the number of possible actions, and in each position of the matrix it will be represented the value Q_b for each pair of state-action.

The problem is that at the beginning the agent does not have this information, for which its first objective is to approximate to the maximum this assignment of values Q_b, which depend on both future and current rewards as show in:

$$Q_b\left(s_t^b, a_t^b\right) = r_t\left(s_t^b, a_t^b\right) + \gamma \max_{a_t^b} Q_b\left(s_{t+1}^b, a_t^b\right) \quad (14)$$

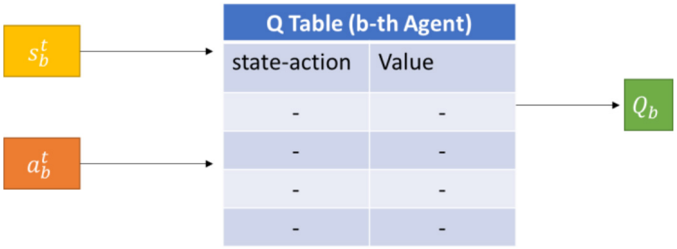


Fig. 6. Q-Table of b-th beam for QL algorithm.

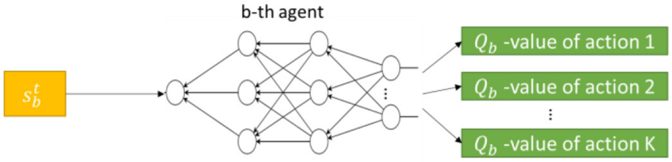


Fig. 7. Neural network of the b-th agent to calculate the Q_b-Values in the DQL algorithm.

where s_{t+1}^b represents the following state and γ is the discount factor that controls the contribution of future rewards.

Since this is a recursive equation, it starts by making arbitrary assumptions for all Q_b values and is implemented as an update:

$$Q_b\left(s_t^b, a_t^b\right) \leftarrow Q_b\left(s_t^b, a_t^b\right) + \alpha\left[r_t\left(s_t^b, a_t^b\right) + \gamma \max_{a_t^b} Q_b\left(s_{t+1}^b, a_t^b\right) - Q_b\left(s_t^b, a_t^b\right)\right] \tag{15}$$

where α is the learning rate or step size. This simply determines the extent to which newly acquired information overrides old information.

The QL algorithm initializes arbitrarily the B Q-Tables and in each iteration the matrix will be updated by using (14).

On the other hand, DQL (Algorithm 3) uses a neural network to approximate the function of the Q_b value, thus avoiding using a table to represent it (Fig. 7). In the input of the Neural Network there is the state of the *b-th* agent, and in the output there is Q_b for each of the possible actions; all the experiences are saved by the *b-th* memory, M_b.

On a higher level, DQL works as such:

a. Collect and store samples in a memory replay buffer with the current policy
b. Random sample batches of experiences from the memory replay buffer (known as Repetition of experiences)
c. Use the sample experiences to update the Q network
d. Repeat a-b

The Neural network loss function is the mean square error of the predicted Q_b value and the target Q_b' value. This is basically a regression problem:

$$L_b\left(s_t^b, a_t^b, \omega_t\right) = \left({Q_b}' - Q_b\left(s_t^b, a_t^b, \omega_t\right)\right)^2 \tag{16}$$

$${Q_b}' = r_t\left(s_t^b, a_t^b, \omega_{t-1}\right) + \gamma \max_{a_t^b} {Q_b}'\left(s_{t+1}^b, a_t^b, \omega_{t-1}\right) \tag{17}$$

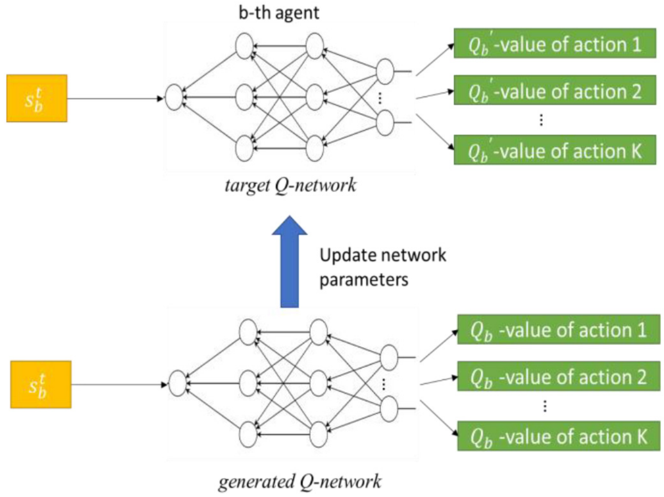


Fig. 8. Target and Generated Neural Network of the b-th agent to calculate the Q_b and ${Q_b}'$ values in the DQL algorithm.

where ${Q_b}'$ is the temporal difference target, and ${Q_b}' - Q_b$ is the temporal difference error, ω_t is the current neural network parameters and ω_{t-1} the previous parameters.

In DDQL (Algorithm 4), each agent uses two neural networks with the same architecture to learn and predict what action to take at each step (Fig. 8). DDQL model includes two deep learning networks, called the *generated Q-network* ($Q_b(s_t^b, a_t^b, \omega_t)$) and the *target Q-network* ($Q_b(s_t^b, a_t^b, {\omega_t}^-)$) where ${\omega_t}^-$ is the current target neural network parameters.

DDQN can produce more accurate value estimates, and, in addition, leads to better overall performance of the deep neural network. The ability of the DDQN to produce more accurate value estimates comes from the fact that it separates the neural network into two networks. The *generated Q-network* is used to generate actions and the *target Q-network* is used to train from randomly selected observations from the replication memory. The replay memory of the DDQN stores state transitions received from the environment, allowing reuse of this data. By taking a random sample from it, the transitions that form a batch are related to uncorrelation, stabilizing the DDQN.

The *generated Q-network* is utilized to compute the *b-th* Q_b value for *b-th* agent while the *target Q-network* aims to produce the target ${Q_b}'$ value to train the parameters of the *generated Q-network*. Depending on the basic idea of the DDQL, the target ${Q_b}'$ value can be defined as:

$${Q_b}' = r_t\left(s_t^b, a_t^b, \omega_{t-1}\right) + \gamma Q_b\left(s_{t+1}^b, \arg\max_{a_t^b} Q_b\left(s_{t+1}^b, a_t^b, \omega_{t-1}\right), {\omega_t}^-\right) \tag{18}$$

For each episode, the *b-th* agent choose a random action according to whether or not a random probability was less than ε; if the value exceed the threshold ε, then the *b-th* agent chooses the action a_t^b according to $\arg\max_{a_t^b} Q_b(s_{t+1}^b, a_t^b, \omega_{t-1})$.

TABLE I
ARCHITECTURE OF THE NEURAL NETWORK FOR DQL AND DDQL (BB)

Neural Network	Hidden Layer 1	Hidden Layer 2
NN 1	64	64
NN 2	132	132
NN 3	264	264

For each episode, a decay was used at time step t where:

$$\varepsilon = \frac{1}{\sqrt{t}}\lambda \tag{19}$$

and λ is a constant scaling factor for ε decay in the range [0, 1].

On the one hand, DDQL algorithm needs fewer episodes to converge with respect to DQL and QL [38], [39]. However, the average time of a QL episode duration is significantly shorter compared to DQL and DDQL depending on the complexity of the problem, neural networks architecture and the features of the computer used [38], [39].

Based on the DRM (Algorithm 1), the training of the agents is online, hence the convergence time of the algorithm is a trade-off, since a more complex algorithm requires fewer episodes to converge but the time required for each episode increases. In this sense, the normalized convergence time, *NCT*, is calculated as:

$$NCT = \frac{ET_{alg} \cdot NE_{alg}}{ET_{QL}} \tag{20}$$

where ET_{alg} is the average time per episode of the algorithm, NE_{alg} is the number of episodes in which the algorithm converges and ET_{QL} is the average time per episode of the QL algorithm.

In DQL (Algorithm 3) and in DDQL (Algorithm 4) it is required to define the neural networks architecture. In that sense, in Table I we propose 3 different neural network architectures for DQL and DDQL, where the input layer has the state shape of each agent and the output layer has the size of the set of possible actions of each agent; the 3 proposed architectures consist of 2 hidden layers but what changes is the number of neurons in each hidden layer.

VI. NUMERICAL RESULTS AND ANALYSIS

The traffic model, as defined in Section III, is based on a service area similar to that of the KaSat satellite [37] with 82 beams. The flexible parameters per beam are power with 8 to 17 dBW with steps of 0.1 dB, bandwidth with 100, 150, 200, 250, 300, 350, 400, 450 or 500 MHz and beamwidth with $0.55°$, $0.60°$ or $0.65°$.

The software tool chain used to implement CMA DRL consist of a Jupyter development environment using Keras 2.0. The computer used for the training phase is an Intel Core i7-7700HQ 2.8 GHz CPU and 16 GB RAM.

A. Cooperative Multi-Agent vs a Single Agent

If the proposed problem were attempted to be solved using SA, the number of possible actions would increase to $K \cdot B$, where *K* is the number of possible actions in a beam and *B* the number of beams, and the number of states would correspond to all the possible combinations of resource allocation in the *B* beams, leading to a very complex problem for a SA. This is demonstrated by the obtained results which allow us to make a comparison in terms of the total reward between CMA and SA.

Algorithm 4 Double Deep Q-Learning

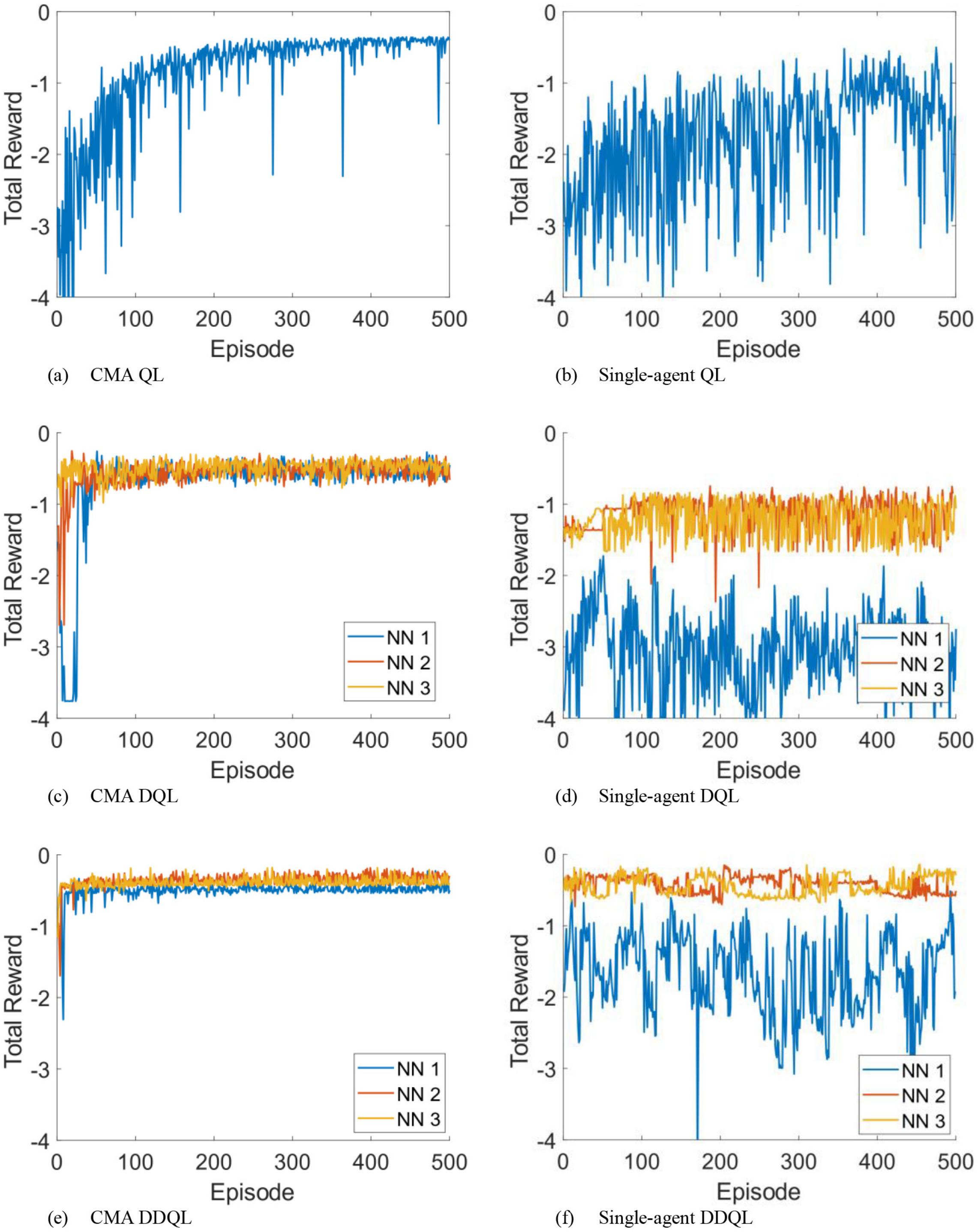
```
-  Set values for $\gamma$ and $M$ which is the set of $B$ replay memories $\{M_1, M, \ldots, M_B\}$, where $M_b$ represents the b-th replay memory
-  Initialize $\omega_t$, ${\omega_t}^-$ and Q-values with random weights
-  Repeat for each episode, do
     ○ Initialize $\overline{S_t} = \{s_t^1, s_t^2, \ldots, s_t^B\}$
     ○ Calculate $\varepsilon$
         $\varepsilon = \frac{1}{\sqrt{t}}\lambda$
     ○ Repeat for each step of episode, do
         • Each agent
             • With probability $<$ $\varepsilon$ select a random action $a_t^b$, otherwise select $a_t^b = arg\max_{a_t^b} Q_b(s_{t+1}^b, a_t^b, \omega_{t-1})$
             • Take action $a_t^b$, observe $r_t$ and $s_{t+1}^b$
             • Store transition $(s_t^b, a_t^b, r_t, s_{t-1}, done)$ in experience replay memory $M_b$.
             • Sample random minibatch of transitions $(s_t^b, a_t^b, r_t, s_{t-1})$ from $M_b$.
             • For every transition in minibatch, do
$Q_b{}' = \begin{cases} r_t(s_t^b, a_t^b, \omega_{t-1}), & if\,done \\ r_t(s_t^b, a_t^b, \omega_{t-1}) + \gamma Q_b(s_{t+1}^b, arg\ \max_{a_t^b} Q_b(s_{t+1}^b, a_t^b, \omega_{t-1}), {\omega_t}^-), & else \end{cases}$
             • End do
             • Calculate the loss
                 $L_b(s_t^b, a_t^b, \omega_t) = (Q_b{}' - Q_b(s_t^b, a_t^b, \omega_t))^2$
             • Update $Q_b$ using gradient descent by minimizing the loss $L_b(s_t^b, a_t^b, \omega_t)$
             • Copy weights from $\omega_t$ to ${\omega_t}^-$
         • End do
     ○ End do
-  End do
```

The results are presented in Fig. 9. The first issue that can be observed is the superiority when using a CMA architecture compared to a SA. It can be seen that with a SA, the QL algorithm still does not converge even after 500 episodes and the reward it gets is still very low. However, it is observed that by using a Multi-Agent architecture, favorable results are obtained even by using a simple algorithm such as QL with which a reward of up to -0.3 is obtained.

Using a CMA architecture, it is observed that for DQL and DDQL the convergence is faster, but the complexity of the algorithm is higher, so it can add a greater delay because the training is performed online. NN 1, NN 2 and NN 3 obtain favorable results for DQL and DDQL when the architecture is CMA, however, when the architecture is a single agent, NN 1 does not converge and has many oscillations.

B. Online Processing Time

The QL and DRL algorithm training is online (Algorithm 1), therefore the time needed to converge the DRL algorithm is



(a) CMA QL

(b) Single-agent QL

(c) CMA DQL

(d) Single-agent DQL

(e) CMA DDQL

(f) Single-agent DDQL

Fig. 9. Performance comparison of algorithms using a CMA distribution with respect to a single agent. (a) CMA QL, (b) single-agent QL, (c) CMA DQL for 3 Neural Network architectures, (d) single-agent DQL for 3 Neural Network architectures, (e) CMA DDQL for 3 Neural Network architectures, (f) single-agent DDQL for 3 Neural Network architectures.

a Key Performance Indicator (KPI) since it represents an added delay. This added delay is a function of the number of episodes that the QL or DRL algorithm needs to converge and the average time of each episode. The average time of each episode depends on the features of the computer used for training. In that sense, Table II show the Normalized convergence time of the QL and DRL algorithms using a CMA distribution. When considering a computer with the features previously listed, the average time per episode is 9.36 s for the QL algorithm. In that sense, Table II shows the normalized convergence times for the different algorithms using a CMA distribution. It should be mentioned that in case of higher processing capabilities of the computer used for running the algorithms, this convergence time can decrease remarkably.

TABLE II
RL ALGORITHMS ADDITIONAL DELAY USING A CMA DISTRIBUTION

RL Algorithm	Normalized average time per episode	Number of episodes to converge	Normalized convergence time
QL	1	332	332
DQL NN 1	1.56	42	65.52
DQL NN 2	1.89	23	43.47
DQL NN 3	2.26	8	18.08
DDQL NN 1	2.64	9	23.76
DDQL NN 2	3.26	7	22.82
DDQL NN 3	3.60	6	21.6

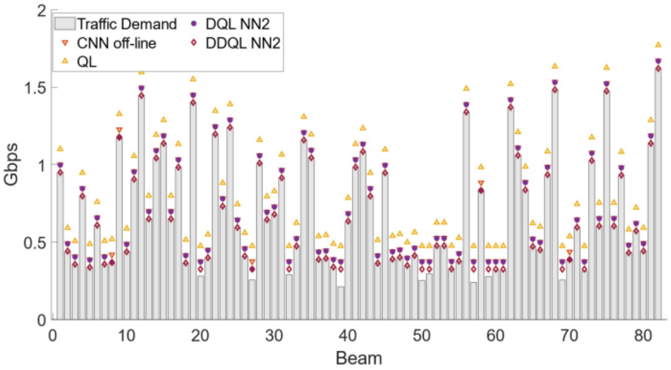


Fig. 10. Performance of CNN and CMA DRL algorithms at 11 a.m. over the entire service area. After 64 normalized time of training for CMA DRL algorithm.

QL is the least complex algorithm (Algorithm 2) so the normalized average time of each episode is only 1, but the number of episodes required to converge is 332, which makes QL the algorithm that adds the highest normalized coverage time to the system (332).

On the other hand, DDQL is the most complex algorithm (Algorithm 4) and a normalized time average per episode of 3.60 is obtained when using the NN 3, but the added complexity results in requiring only 6 episodes to converge obtaining a normalized convergence time to the system of 21.6, that is, 93% less than the time required using QL.

DQL is the algorithm that obtains the lowest added delay when using an NN 3 architecture, since with only 8 episodes it converges obtaining an added delay of 18.08, allowing an effective trade-off between average time per episode and number of required episodes.

C. Performance Evaluation

QL and DRL algorithms performance for a CMA distribution was evaluated on simulated traffic demand required in the service area for 24 hours, assuming a fixed normalized training time of 64. After 64 normalized training time, most of the RL algorithms have converged except the QL and the DQL NN1 (Table II).

CNN is an offline algorithm proposed in [20] using the same cost function (1) for the DRM model. In that sense, Figure 10 presents a comparison of the DRM performance using CNN or DRL algorithms with a CMA distribution (QL, DQL NN2 and DDQL NN2) at 11 a.m. over the entire service area.

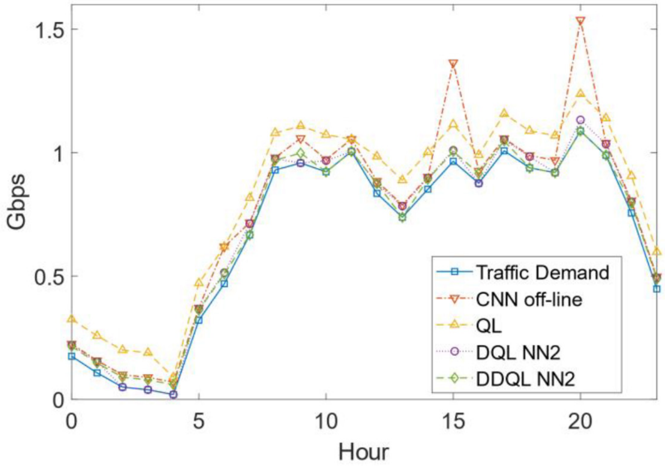


Fig. 11. Performance of CNN and CMA DRL algorithms during 24 hours for the Beam 1 after 64 normalized time of training.

It is observed that due to the constraints of the DRM function in (4), the capacity offered in each beam is greater than or equal to the traffic demand for all the evaluated algorithms. The QL algorithm shows a greater error following the shape of the traffic demand, this is due to the training time that was established. The DDQL algorithm is the most successful performing by following the shape of the traffic demand on all beams. In most beams, CNN and DQL have similar performance, although in some cases (e.g., Beam 1) DQL has similar performance to DDQL.

The performance shown in Fig. 11 was obtained after evaluating the algorithms performance limited to the Beam 1 for 24 hours. It is observed that using any algorithm, the capacity offered for 24 hours is greater or equal to the required demand. CNN demonstrates a superiority in tracking traffic demand compared to QL except at 15 and 20 hours. However, it is important to note that with 64 normalized time of training the QL algorithm still does not converge. On the other hand, DQL NN 2 and DDQL NN 2 present a superiority compared to CNN. DDQL NN2 is the one that obtains a better performance when tracking the shape of the traffic demand.

In terms of saving resources [20], the $(P_{Payload})$ normalized payload power is defined as:

$$P_{Payload} = \frac{P_{Total,Alg}}{P_{Total,UPA}} \tag{21}$$

where $P_{Total,UPA}$ is the Total Payload Power when using a Uniform Power Allocation (UPA) and $P_{Total,Alg}$ is Total Payload Power when using the Power Allocation using the proposed algorithm.

Lei and Vázquez-Castro [15] achieve power consumption reductions of up to 3 dB compared to a traditional payload in their proposal published, in which they manage power and bandwidth using a suboptimal method (SOM). Figure 12 presents the normalized payload power using SOM, CNN and the RL and DRL algorithms (QL, DQL and

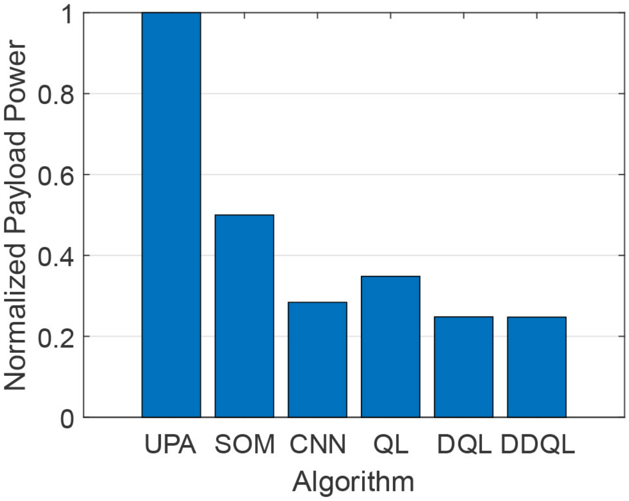


Fig. 12. System Power saving for CMA DRL algorithms compared to CNN. Note: For DQL and DDQL the NN2 is used.

DDQL) with a CMA distribution. It is observed that ML algorithms achieve higher power savings compared to SOM. DQL and DDQL obtain the lowest normalized payload power with a difference of almost 0.3 below than the CNN normalized payload power. However, it is observed that QL gets the highest normalized payload power between all ML algorithms.

VII. Conclusion

The DRM problem for VHTS systems is defined as a novel MDP with a multiagent environment that works cooperatively to achieve maximum reward. All agents share the same reward but each agent must meet minimum conditions, which guarantees that despite working cooperatively, each agent will seek its own benefit. In that sense, one RL algorithm and two DRL algorithms to manage the resources available in flexible payload architectures for DRM are suggested, i.e., QL, DQL and DDQL using a CMA distribution, and compared based on their performance, complexity and added latency. This work demonstrates the superiority a CMA has over a SA. The proposed algorithms are also compared with a recently proposed offline algorithm in the state of the art, CNN, and their performance is evaluated with respect to resource management. In addition, a comparison is made with an algorithm based on a sub-optimal method in terms of the power saving. In this work it is proposed that the training of agents is performed online, hence the time required by each algorithm to converge is critical as it represents a delay added to the system. It is important to note that the added delay obtained during the simulations depends on the features of the computer used, so when applied in a real system, a high-performance computer could be able to reduce the training times.

As a future work, it is intended to include a wider study including different co-channel interference mitigation techniques and to evaluate their effects on the proposed system. In addition, a comparison of the cost per Gbps in orbit with all ML algorithms used for DRM will be carried out.

References

[1] G. Giambene, S. Kota and P. Pillai, “Satellite-5G integration: A network perspective,” *IEEE Netw.*, vol. 32, no. 5, pp. 25–31, Sep./Oct. 2018.

[2] A. Agarwal and P. Kumar, “Analysis of variable bIT rate SOFDM based integrated satellite-terrestrial broadcast system in presence of CFO and phase noise,” *IEEE Syst. J.*, vol. 13, no. 4, pp. 3827–3835, Dec. 2019.

[3] F. Ortiz-Gomez, R. Martínez, M. A. Salas-Natera, A. Cornejo, and S. Landeros-Ayala, “Forward link optimization for the design of VHTS satellite networks,” *Electronics*, vol. 9, no. 3, p. 473, 2020.

[4] F. Li, K.-Y. Lam, M. Jia, K. Zhao, X. Li, and L. Wang, “Spectrum optimization for satellite communication systems with heterogeneous user preferences,” *IEEE Syst. J.*, vol. 14, no. 2, pp. 2187–2191, Jun. 2020.

[5] *Connectivity for a Competitive Digital Single Market—Towards a European Gigabit Society*, document COM(2016)587 and Staff Working Document-SWD(2016)300, European Commission, Brussels, Belgium, Sep. 2016. [Online] Available: https://ec.europa.eu/digital-single-market/en/news/communication-connectivity-competitive-digital-single-market-towards-european-gigabit-society

[6] N. Pachler, J. J. G. Luis, M. Guerster, E. Crawley and B. Cameron, “Allocating power and bandwidth in multibeam satellite systems using particle swarm optimization,” in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2020, pp. 1–11.

[7] F. G. Ortíz-Gómez, R. M. Rodríguez-Osorio, M. Salas-Natera, and S. Landeros-Ayala, “Adaptive resources allocation for flexible payload enabling VHTS systems: Methodology and architecture,” in *Proc. 36th Int. Commun. Satellite Syst. Conf. (ICSSC)*, 2018, pp. 1–8, doi: 10.1049/cp.2018.1694.

[8] Y. Abe, M. Ogura, H. Tsuji, A. Miura, and S. Adachi, “Resource and network management framework for a large-scale satellite communications system,” *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E103-A, no. 2, pp. 492–501, Feb. 2020.

[9] M. Guerster, J. Grotz, P. Belobaba, E. Crawley, and B. Cameron, “Revenue management for communication satellite operators—Opportunities and challenges,” in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2020, pp. 1–15.

[10] E. Lagunas, S. K. Sharma, S. Maleki, S. Chatzinotas, and B. Ottersten, “Resource allocation for cognitive satellite communications with incumbent terrestrial networks,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 1, no. 3, pp. 305–317, Sep. 2015.

[11] A. Paris, I. Del Portillo, B. Cameron, and E. Crawley, “A genetic algorithm for joint power and bandwidth allocation in multibeam satellite systems,” in *Proc. IEEE Aersp. Conf.*, Big Sky, MT, USA, Mar. 2019, pp. 1–15.

[12] G. Cocco, T. de Cola, M. Angelone, Z. Katona, and S. Erl, “Radio resource management optimization of flexible satellite payloads for DVB-S2 systems,” *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 266–280, Jun. 2018.

[13] Y. Kawamoto, T. Kamei, M. Takahashi, N. Kato, A. Miura, and M. Toyoshima, “Flexible resource allocation with inter-beam interference in satellite communication systems with a digital channelizer,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 2934–2945, May 2020.

[14] S. Kisseleff, E. Lagunas, T. S. Abdu, S. Chatzinotas and B. Ottersten, “Radio resource management techniques for multibeam satellite systems,” *IEEE Commun. Lett.*, early access, Oct. 23, 2020, doi: 10.1109/LCOMM.2020.3033357.

[15] J. Lei and M. A. Vázquez-Castro, “Joint power and carrier allocation for the multibeam satellite downlink with individual SINR constraints,” in *Proc. IEEE Int. Conf. Commun.*, Cape Town, South Africa, 2010, pp. 1–5.

[16] S. Liu, Y. Fan, Y. Hu, D. Wang, L. Liu, and L. Gao, “AG-DPA: Assignment game–based dynamic power allocation in multibeam satellite systems,” *Int. J. Satellite Commun. Netw.*, vol. 38, no. 1, pp. 74–83, Jan./Feb. 2020.

[17] F. G. Ortiz-Gomez, D. Tarchi, R. M. Rodriguez-Osorio, A. Vanelli-Coralli, M. A. Salas-Natera, and S. Landeros-Ayala, “Supervised machine learning for power and bandwidth management in VHTS systems,” in *Proc. 10th Adv. Satellite Multimedia Syst. Conf. 16th Signal Process. Space Commun. Workshop (ASMS/SPSC)*, Graz, Austria, 2020, pp. 1–7.

[18] F. G. Ortiz-Gomez, R. Martínez, M. A. Salas-Natera, S. Landeros-Ayala, D. Tarchi, and A. Vanelli-Coralli, "On the use of neural networks for flexible payload management in VHTS systems," in *Proc. 25th Ka Broadband Commun. Conf.*, Sorrento, Italy, Oct. 2019, pp. 1–10.

[19] F. G. Ortiz-Gomez, R. Martínez, M. Salas-Natera, and S. Landeros-Ayala, "On the use of machine learning for flexible payload management in VHTS systems," in *Proc. 70th Int. Astronaut. Congr.*, Washington, DC, USA, Oct. 2019, pp. 1–6.

[20] F. G. Ortiz-Gomez, D. Tarchi, R. Martínez, A. Vanelli-Coralli, M. A. Salas-Natera, and S. Landeros-Ayala, "Convolutional neural networks for flexible payload management in VHTS systems," *IEEE Syst. J.*, early access, Sep. 10, 2020, doi: 10.1109/JSYST.2020.3020038.

[21] P. V. R. Ferreira, R. Paffenroth, A. M. Wyglinski, T. M. Hackett, S. G. Bilen, and J. Mortensen, "Multiobjective reinforcement learning for cognitive satellite communications using deep neural network ensembles," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 5, pp. 1030–1041, May 2018.

[22] P. V. R. Ferreira *et al.*, "Multi-objective reinforcement learning-based deep neural networks for cognitive space communications," in *Proc. Cogn. Commun. Aerosp. Appl. Workshop (CCAA)*, Cleveland, OH, USA, Jun. 2017, pp. 1–8.

[23] J. J. G. Luis, M. Guerster, I. Del Portillo, E. Crawley, and B. Cameron, "Deep reinforcement learning for continuous power allocation in flexible high throughput satellites," *Proc. IEEE Cogn. Commun. Aerosp. Appl. Workshop (CCAAW)*, Cleveland, OH, USA, Jun. 2019, pp. 1–4.

[24] S. Liu, X. Hu, and W. Wang, "Deep reinforcement learning based dynamic channel allocation algorithm in multibeam satellite systems," *IEEE Access*, vol. 6, pp. 15733–15742, 2018.

[25] X. Liao *et al.*, "Distributed intelligence: A verification for multi-agent DRL-based multibeam satellite resource allocation," *IEEE Commun. Lett.*, vol. 20, no. 12, pp. 2785–2789, Dec. 2020.

[26] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1. Cambridge, MA, USA: MIT Press, 1988.

[27] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[28] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," Deepmind Technol., London, U.K., Rep., 2013. [Online]. Available: arXiv:1312.5602

[29] G. Weiss, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1999.

[30] E. A. Feinberg and A. Shwartz, *Handbook of Markov Decision Processes Handbook and Applications*, Boston, MA, USA: Kluwer's Academic, 2002.

[31] A. Oroojlooyjadid and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," 2019. [Online]. Available: arXiv:1908.03963.

[32] W. Zemzem and M. Tagina, "Cooperative multi-agent systems using distributed reinforcement learning techniques," *Procedia Comput. Sci.*, vol. 126, pp. 517–526, Jan. 2018.

[33] A. J. Roumeliotis, C. I. Kourogiorgas, and A. D. Panagopoulos, "Optimal capacity allocation strategies in smart gateway satellite systems," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 56–59, Jan. 2019.

[34] A. J. Roumeliotis, C. I. Kourogiorgas, and A. D. Panagopoulos, "Dynamic capacity allocation in smart gateway high throughput satellite systems using matching theory," *IEEE Syst. J.*, vol. 13, no. 2, pp. 2001–2009, Jun. 2019.

[35] *Digital Video Broadcasting (DVB) Second Generation Framing Structure, Channel Coding and Modulation Systems for Broadcasting, Interactive Services, News Gathering and other Broadband Satellite Applications; Part 2: DVB-S2 Extensions (DVB-S2X), Rev. 1.1.1*, ETSI Standard EN 302 307-2, Oct. 2014.

[36] D. de la Torre, F. G. Ortiz-Gomez, M. A. Salas-Natera, and R. Martínez, "Analysis of the traffic demand on very high throughput satellite for 5G," in *Proc. 35th Simposio Nacional de la Unión Científica Internacional de Radio URSI*, Sep. 2020, pp. 1–4.

[37] (Oct. 2020). *Eutelsat 9B KA-SAT*. [Online]. Available: https://www.eutelsat.com/en/satellites/eutelsat-9-east.html

[38] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, S. U. Khan, and P. Li, "A double deep Q-learning model for energy-efficient edge scheduling," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 739–749, Sep./Oct. 2019.

[39] D. Simoes, N. Lau, and L. P. Reis, *Multi-Agent Double Deep Q-Networks BT- Progress in Artificial Intelligence* (Lecture Notes in Computer Science 10423). Heidelberg, Germany: Springer, 2017, pp. 123–134.

Flor G. Ortiz-Gomez received the B.S. degree in telecommunications engineering and the M.S. degree in electrical-telecommunications engineering from the Universidad Nacional Autónoma de México, Mexico City, Mexico, in 2015 and 2016, respectively. She is currently pursuing the Ph.D. degree with the Universidad Politécnica de Madrid, Spain, where he has joined the Radiation Group.

Daniele Tarchi (Senior Member, IEEE) received the M.S. degree in telecommunications engineering from the University of Florence, Italy, in 2000, and the Ph.D. degree in informatics and telecommunications engineering from the University of Florence, Italy, in 2004. He is currently an Associate Professor with the University of Bologna, Italy. He has authored numerous journal articles and conference papers. His research interests are mainly on wireless communications, resource allocation techniques, edge computing, and fog computing scenarios. He has been a Symposium Co-Chair for IEEE WCNC 2011, IEEE Globecom 2014, IEEE Globecom 2018, and IEEE ICC 2020, and a Workshop Co-Chair at IEEE ICC 2015. He is Editorial Board Member for IEEE WIRELESS COMMUNICATIONS LETTERS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and *IET Communications*.

Ramón Martínez received the Ph.D. degree in electrical engineering from the Universidad Politécnica de Madrid (UPM), Madrid, Spain, in 2004.

In 2002, he joined the Telecommunication Engineering School, UPM (ETSIT-UPM), where he has been a Full Professor since 2019, and a Member of the Information Processing and Telecommunications Center. He has been a Lecturer with SatCom Systems since 2004. His research areas are wireless and satellite communications, including the application of antenna array processing systems. In the last years, he has worked in the optimization of high throughput satellite systems, with the use and operation of flexible payloads using machine learning techniques. He has led and participated in different international projects related to space technology, with collaborations with ESA and space communication industry in technology transfer programs. He led the design of antenna systems for intersatellite links in nanosatellite missions, and currently he is the Principal Investigator of FUTURE-RADIO, a project to evaluate radio technologies for future communication systems including 5G. He received awards for his Ph.D. thesis focused on the application of smart antennas to cellular communications and the UPM Innovative Teaching Award.

Alessandro Vanelli-Coralli (Senior Member, IEEE) received the Dr.Ing. degree in electronics engineering and the Ph.D. degree in electronics and computer science from the University of Bologna, Bologna, Italy, in 1991 and 1996, respectively.

In 1996, he joined the University of Bologna, where he is currently an Associate Professor with the Department of Electrical, Electronic, and Information Engineering (Guglielmo Marconi). From 2013 to 2018, he chaired the Ph.D. Board, Electronics, Telecommunications and Information Technologies. From 2003 to 2005, he was a Visiting Scientist with the Qualcomm, Inc., San Diego, CA, USA. He participates in national and international research projects on wireless and satellite communication systems. He has been a Project Coordinator and scientifically responsible for several European Space Agency and European Commission funded projects. His research interests include wireless communications, digital transmission techniques, and digital signal processing.

Dr. Vanelli-Coralli was a corecipient of several best paper awards. He was a general chairman and the technical chairman for several scientific conferences. He has been an appointed member of the editorial board of the *International Journal of Satellite Communications and Networking* (Wiley InterScience) and has been a Guest Co-Editor for several special issues of the international scientific journals.

Miguel A. Salas-Natera received the master's degree in space technologies and the Ph.D. degree in technologies and communications systems from Universidad Politécnica de Madrid (UPM) in 2011 and 2011, respectively.

He is an Electrical Engineer major in Telecommunications. In 2012, he was a Technical Director with Antenna System Solutions Company (spin-off of the UPM) for two years. He is an Assistant Professor and a Researcher with the Radiation Group, UPM. In addition, he is the responsible of the special session on satellite communications systems (SATCOM) since its beginnings in URSI2016 in Spain. He has led and participated in a number of international and local projects related to the development of new antenna technologies, antenna test and validation, and analysis of SATCOM Systems. Besides, he has work in uncertainty analyses and calibration methods for active antenna arrays. He is currently involved in the development and design of ultra-compact reflector antennas including novel surface design and manufacturing technologies. He is actively participating in the development of algorithm for resource allocation of flexible-transparent transponder systems for classical GEO satellites systems as well as for novel massive LEO satellites and HAP systems.

Salvador Landeros-Ayala received the B.S. degree in mechanical and electrical engineering from the Universidad Nacional Autónoma de México (UNAM) in 1977, the M.S. degree in electrical engineering from the University of Pennsylvania, USA, in 1980 and the Ph.D. degree in electrical engineering from UNAM in 1999. He is with the National Autonomous University of Mexico, where he has been in several academic positions, leading research and industry satellite communications projects since 1977. He has been a member of the Mexican Academy of Engineering and the Head of the Mexican Space Agency since 2019.