Deep Reinforcement Learning for Continuous Power Allocation in Flexible High Throughput Satellites

Juan Jose Garau Luis
System Architecture Lab
Massachusetts Institute of Technology
Cambridge, MA
garau@mit.edu

Markus Guerster
System Architecture Lab
Massachusetts Institute of Technology
Cambridge, MA
guerster@mit.edu

Inigo del Portillo
System Architecture Lab
Massachusetts Institute of Technology
Cambridge, MA
portillo@mit.edu

Edward Crawley
System Architecture Lab
Massachusetts Institute of Technology
Cambridge, MA
crawley@mit.edu

Bruce Cameron
System Architecture Lab
Massachusetts Institute of Technology
Cambridge, MA
bcameron@mit.edu

***Abstract*—Many of the next generation of satellites will be equipped with numerous degrees of freedom in power and bandwidth allocation capabilities, making manual resource allocation impractical. Therefore, it is desirable to automate the operation of these highly flexible satellites. This paper presents a novel approach based on Deep Reinforcement Learning to allocate power in multibeam satellite systems. The proposed architecture represents the problem as continuous state and action spaces. We make use of the Proximal Policy Optimization algorithm to optimize the allocation policy for minimum unmet system demand and power consumption. Finally, the performance of the algorithm is analyzed through simulations of a multibeam satellite system. The analysis shows promising results for Deep Reinforcement Learning to be used as a dynamic resource allocation algorithm.**

Keywords—High throughput satellites, resource allocation, deep reinforcement learning

I. INTRODUCTION

To better serve the increasing demand of broadband data from space, next generation satellite systems will feature advanced payloads able to operate hundreds (or even thousands) of beams simultaneously and change their parameters dynamically. This increased flexibility will render invalid traditional resource allocation approaches, since these largely lean on static allocations and the use of conservative margins. Instead, satellite operators face the challenge of automating their resource allocation strategies to exploit this flexibility and turning it into a larger service capacity.

As pointed out in [1], dynamic resource management systems will be key to be competitive in this new markets. One important element of these systems is an optimization algorithm that computes the optimal resource allocation at any given moment. However, developing such algorithm involves dealing with a high-dimensional, non-convex [2], and NP-hard [3] problem for which many classic optimization algorithms perform poorly. Multiple authors have already proposed different alternatives to overcome this issue.

Several studies have focused on approaches based on metaheuristics [2-5]. Although these algorithms have proved to be good solutions in power and bandwidth allocation problems, authors do not assess their performance under real operational time constraints. These algorithms are based on iterative methods that have a specific convergence time, which might impose a hard constraint on their real-time use.

Other authors propose approaches focused on Deep Reinforcement Learning (DRL) architectures as an alternative. DRL has proved to be an operable solution for real-time and single-channel resource allocation problems [6]. It also exploits the inherent time and spatial correlations of the problem [7]. However, both DRL studies propose architectures that discretize the resources before allocating them. While satellite resources such as power are intrinsically continuous, discretization might entail a notable increase in computational cost when the dimensionality of the problem is high. In this study, we explore a DRL architecture for power allocation that uses continuous action and state spaces, avoiding the need for discretization.

The rest of the paper is divided as follows: Section 2 describes the problem statement and the satellite communications models used in this work, Section 3 presents our DRL approach, Section 4 discusses the performance of the algorithm on a simulated satellite, and finally Section 5 outlines the conclusions of the paper.

II. PROBLEM STATEMENT

A. Problem Motivation

The next generation of satellites will allow for unprecedented parameter flexibility: the power and bandwidth, the frequency plan, and the pointing and shape of each of the beams will be individually configurable. To start exploring the adequateness of DRL to dynamically control all of these continuous parameters subject to the constraints of a real-operation scenario, in this study we only focus on one satellite resource: optimizing the power allocation for each beam while all the other parameters remain fixed.

B. Link Budget Model

This subsection presents the link-budget equations to compute the data rate achieved by one beam (R_b), assuming that a power P_b has been allocated to such beam. Our link budget model is a parametric model based on [4].

We first compute the link's C/N_0 as

This work was supported by SES. The authors want to thank SES for their input to this paper and their financial support.

$$\frac{C}{N_0} = P_b - OBO + G_{T_x} + G_{R_x} \tag{1}$$
$$-FSPL - 10\log_{10}(kT_{sys}) \quad [\text{dB}]$$

where OBO is the power-amplifier output back-off (dB), G_{T_x} and G_{R_x} are the transmitting and receiving antenna gains, respectively (dB), FSPL is the free-space path loss (dB), k is the Boltzmann constant, and T_{sys} is the system temperature (K). For this model we do not consider interference and therefore we can compute the link's $E_b/(N+I)$ as

$$\frac{E_b}{N+I} = \frac{E_b}{N} = \frac{C}{N_0} \cdot \frac{BW}{R_b} \tag{2}$$

where BW is the bandwidth allocated to that beam (Hz) and R_b is the link data rate achieved by beam b (bps), computed as

$$R_b = \frac{BW}{1+\alpha_r} \cdot \Gamma\left(\frac{E_b}{N_0}\right) \quad [\text{bps}] \tag{3}$$

where α_r is the roll-off factor and Γ is the spectral efficiency of the modulation and coding scheme (MODCOD) (bps/Hz). In this study, we assume that adaptive coding and modulation (ACM) strategies are used, and therefore the MODCOD used on each link is the one that provides the maximum spectral efficiency while satisfying the following condition

$$\left.\frac{E_b}{N}\right|_{th} + \gamma_m \leq \frac{E_b}{N} \quad [\text{dB}] \tag{4}$$

where $\left.\frac{E_b}{N}\right|_{th}$ is the MODCOD threshold (dB), $\frac{E_b}{N}$ is the actual link energy per bit to noise ratio (dB) computed using (2), and γ_m is the desired link margin (dB). We assume in this paper the satellites use the MODCOD schemes defined in the standards DVB-S2 and DVB-S2X [8].

C. Problem Formulation

We consider a multibeam GEO satellite with N_b non-steerable beams, and a total available power P_{tot}. Furthermore, each beam has its own maximum power constraint, represented by P_b^{max}. For each beam, power can be dynamically allocated to satisfy the estimated demand at every time instant. The objective is to optimally allocate these resources throughout a time interval of T timesteps to minimize the overall Unmet System Demand (USD) and power consumption.

The USD, defined as the fraction of the demand that is not satisfied by the satellite, is a popular figure of merit to quantify the goodness of a resource allocation algorithm in satellite systems [3, 4]. Mathematically, the USD at timestep t is expressed as

$$USD_t = \sum_{k=1}^{N_b} \max\,[D_{b,t} - R_{b,t}, 0] \tag{5}$$

where $D_{b,t}$ and $R_{b,t}$ correspond to the demand and data rate achieved of beam b at timestep t, respectively.

Using $P_{b,t}$ as the power allocated to beam b at timestep t, our optimization problem can be formulated as the following mathematical program

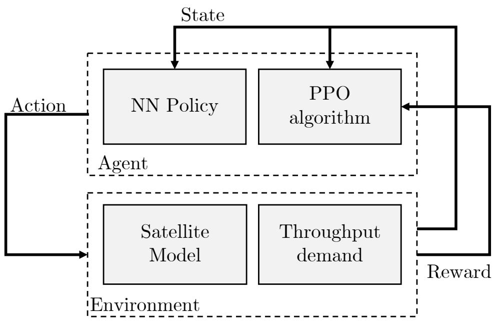

Fig. 1. DRL Architecture

$$\underset{P_{b,t}}{\text{minimize}} \quad \sum_{t=1}^{T} \left[USD_t(P_{b,t}) + \beta \sum_{k=1}^{N_b} P_{b,t}\right] \tag{6}$$

$$s.t. \quad P_{b,t} \leq P_b^{max} \qquad \forall b \in \mathcal{B}, \forall t \in \{1,\ldots,T\} \tag{7}$$

$$\sum_{k=1}^{N_b} P_{b,t} \leq P_{tot}, \qquad \forall t \in \{1,\ldots,T\} \tag{8}$$

$$P_{b,t} \geq 0, \qquad \forall b \in \mathcal{B},\ \forall t \in \{1,\ldots,T\} \tag{9}$$

where $\mathcal{B}$ is the set of beams of the satellite and β is a scaling factor.

III. Deep Reinforcement Learning Setup

A. DRL Architecture

A basic Reinforcement Learning architecture is composed of two essential elements: an agent and an environment [9]. These interact by means of the agent's actions and the environment's states and rewards. Given a state s_t that characterizes the environment at a certain timestep t, the goal of the agent is to take the action a_t that will maximize the discounted cumulative reward G_t, defined as

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k \tag{10}$$

where T is the length of the episode, r_k is the reward obtained at timestep k, and γ is the discount factor. An episode is a sequence of states $\{s_0, s_1, \ldots, s_T\}$ in which the final state s_T is terminal, i.e. no further action can be taken.

Figure 1 shows the specific architecture considered for the power allocation problem. The environment comprises everything that is relevant to the problem and is uncontrollable by the agent. In this case it is composed by the satellite model and the demand per beam. The agent corresponds to the processing engine that allocates power given the environment's state. Its components are an allocation policy $\pi(a_t|s_t)$, that chooses the action a_t given the environment state s_t, and a policy optimization algorithm that constantly improves the policy based on past experience. Since the power and demand per beam are continuous variables, the number of different states and actions is infinite. As a consequence, working with allocation policies that store the best possible action given a state is impractical in this context. Instead, we use a neural network (NN) to model the policy and achieve a feasible mapping between an input state and an output action.

Continuous states and actions also have an impact on the policy optimization algorithm. Policy Gradient methods [10] have shown better results when states and actions are continuous spaces, as their approach focuses on directly optimizing a parametric policy $\pi_\theta(a_t|s_t)$ as opposed to computing the Q values [9] and constructing a policy from them.

In this study we use a Policy Gradient method known as Proximal Policy Optimization (PPO) algorithm [11] to improve the allocation policy. PPO algorithms alternate between sampling data by interacting with the environment and optimizing a "surrogate" objective function using stochastic gradient ascent. The algorithm uses a pessimistic estimate of the performance of the policy an does not allow big consecutive changes to it. This way, it prevents changes that could make the policy perform notably worse in some cases, thus enabling more stable and less fluctuating operation.

B. DRL Application

With the architecture presented in the previous subsection, we proceed to define its specific details. We explored different alternative state representations, one exclusively based on demand information and the other consisted of demand and past actions. We found that considering the previous optimal allocation worked best. We use $\mathcal{D}_t$ to represent the set of demand requirements per beam at timestep t. Then, the *state* of the environment at timestep t is defined as follows

$$s_t = \{\mathcal{D}_t, \mathcal{D}_{t-1}, \mathcal{D}_{t-2}, \mathcal{P}^*_{t-1}, \mathcal{P}^*_{t-2}\} \tag{11}$$

where $\mathcal{P}^*_{t-1}$ and $\mathcal{P}^*_{t-2}$ are the optimal power allocations for the two previous timesteps. Given this definition, the state is encoded with a vector with $5N_b$ components. Every time a new episode starts, the state is reset to $s_0 = \{\mathcal{D}_0, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}\}$.

Since we are only using the power beam as optimization variables, we can use (1)-(4) to determine the minimum power $P^*_{b,t}$ that satisfies $R_{b,t} \geq D_{b,t}$. If the demand can't be met, this optimal power equals the maximum allowed power P_b^{max}.

The *action* of the agent is allocating the power for each beam. Therefore, the action a_t is defined as a vector with N_b components, being the power values $P_{b,t}$ for each beam at timestep t. To respect constraints (7) and (9), these power values are clipped between zero and P_b^{max}. Therefore,

$$a_t = \{P_{b,t} \mid b \in \{1, \dots, N_b\},\ 0 \leq P_{b,t} \leq P_b^{max}\} \tag{12}$$

As reflected in (6), the goal of the problem is to minimize the USD and the power usage during a sequence of consecutive timesteps $\{1, \dots, T\}$. The *reward* function r_t focuses on both objectives and is defined as follows

$$r_t = \frac{\alpha \sum_{b=1}^{N_b} \min(R_{b,t} - D_{b,t}, 0)}{\sum_{b=1}^{N_b} D_{b,t}} - \frac{\sum_{b=1}^{N_b} \left(P_{b,t} - P^*_{b,t}\right)^2}{\sum_{b=1}^{N_b} P^*_{b,t}} \tag{13}$$

where α is a weighting constant, $P_{b,t}$ is the power set by the agent, $P^*_{b,t}$ is the optimal power, $R_{b,t}$ is the data rate achieved after taking the action, and $D_{b,t}$ is the demand of beam b at timestep t. Both the data rate and the optimal power are computed using (1)-(4).

The first element of the equation focuses on satisfying the demand while the second element responds to the necessity of reducing power without underserving that demand. Both elements are normalized by the overall demand and the total optimal power, respectively. The constant α is used to define a priority hierarchy between the two objectives. Given the nature of the problem we are interested in prioritizing a smaller USD. According to the previous definition, we have $r_t \leq 0, \forall t$.

As previously introduced, Policy Gradient methods focus on optimizing parametric policies $\pi_\theta(a_t|s_t)$. In our case, the *policy* is given by the neural network, parametrized by the weights of its hidden layers. For this study we have used a multilayer perceptron (MLP). We found a neural network architecture with four layers, $15N_b$ hidden units, and Rectified Linear Units (ReLU) activation functions to achieve better results. We also made use of normalization layers after each hidden layer to reduce training time.

TABLE I. SIMULATION PARAMETERS

Discount factor γ	0.05
Learning rate	$3 \cdot 10^{-4}$
Number of steps per update	16
Number of training minibatches per update	4
Number of training epochs per update	4
λ [11]	0.8
Clip range [11]	0.2
Gradient norm clipping coefficient [11]	0.5
α (Eq. 13)	100

IV. RESULTS

To assess the performance of the proposed architecture we simulate a 30-beam GEO satellite ($N_b = 30$) located over North America. For each beam, we have a time series containing 1,440 data points that correspond to demand samples throughout a 48-hour activity period (a sample every 2 minutes). This data was provided by SES. We define an episode as a complete pass through the first 720 samples of this dataset (the first 24 hours).

We use the second half of the time series to evaluate the policy performance on unseen data. Then, we ran 10 simulations using the parameters of the PPO algorithm listed in table I. In all simulations we used 8 environments in parallel to acquire more experience and increase training speed.

Figure 2 shows the mean and 95% confidence interval (CI) of the simulation reward sequence after 10 runs of 50,000 timesteps each (68 training episodes per environment, 544 in total). We can clearly observe two tendencies: first, the mean reward rapidly increases during the first thousands of iterations and then notably reduces the improvement speed for the rest of the simulation; and second, the sequence presents a high-frequency component.

We can understand these effects looking at Fig. 3, which shows the mean and 95% CI, for the aggregated data rate achieved using the power allocation policy during an additional episode composed by the full 48-hour dataset. The aggregated demand corresponding to the dataset used is also shown in the figure. The vertical axis is normalized to the maximum aggregated demand value.

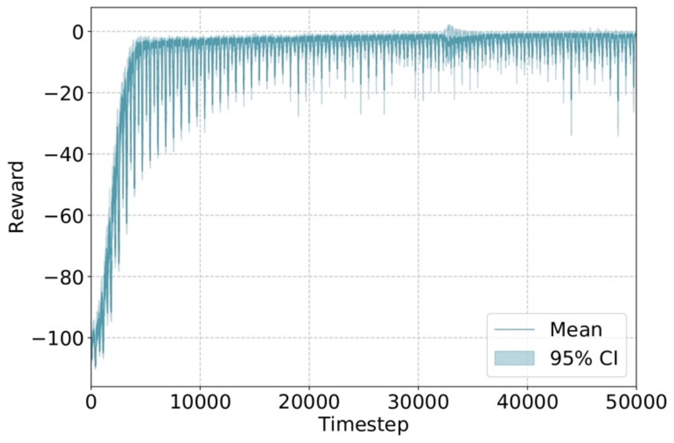

Fig. 2. Mean and 95% confidence interval after 10 simulations of the reward sequence of the whole simulation for one environment.

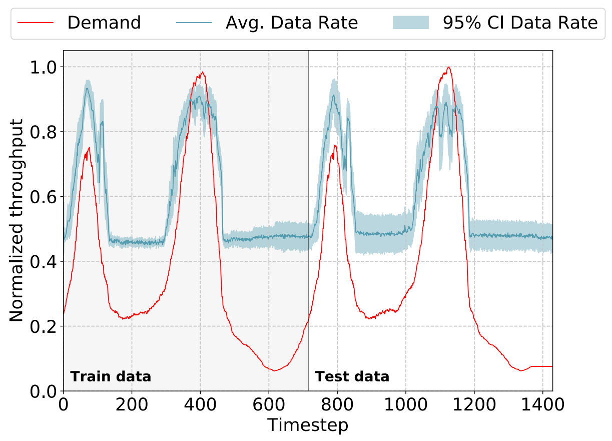

Fig. 3. Mean and 95% confidence interval after 10 simulations of the aggregated provided data rate compared to the normalized aggregated demand. The last 720 samples correspond to unseen data.

We can observe the resulting policy responds to the demand peaks, as the data rate increases at each of them. When the demand is low, the policy sets an almost constant power and consequently sets a constant data rate at approximately 45% of the maximum demand. The variance is also larger on the unseen data.

Although the policy is capable of serving all demand during the first peak of the unseen data, in the second peak it still shows behaviors that drift away from the desirable performance. The power set is not enough to meet the demand and therefore the reward is penalized due to an USD greater than zero. This behavior is repeated through all episodes and originates the high-frequency component from Fig. 2. In the second peak we can also appreciate an artifact product of the policy (timesteps 1050 to 1150 approx.), a behavior not desirable during real operations.

During low demand intervals, the policy is keeping a power threshold that equals to a data rate threshold. This derives from the need to keep the links active as in a real scenario. In the cases where the demand is lower than the data rate threshold, the optimal power is the one that keeps the links active.

V. Conclusion

In this paper, a DRL-based dynamic power allocation architecture for flexible high throughput satellites has been proposed. As opposed to previous architectures [6, 7], this approach makes use of continuous state and action spaces to define the policy. We have set the reward function to focus on minimizing the Unmet System Demand and power consumption. The results obtained show that the architecture produces a policy that responds to demand peaks. However, the policy is not optimal, since some demand is still missed. Nevertheless, the almost instant evaluation of the neural network makes DRL an attractive solution for real-time resource allocation. Future work will be the refinement and generalization of the architecture, the scalability of the policies, and the exploration of other DRL approaches.

References

[1] M. Guerster, J. J. Garau Luis, E. F. Crawley, and B. G. Cameron, “Problem representation of dynamic resource allocation for flexible high throughput satellites,” in 2019 IEEE Aerospace Conference, 2019.

[2] 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 Transactions on Broadcasting, 64(2):266-280, 2018.

[3] A. I. Aravanis, B. Shankar, P. D. Arapoglou, G. Danoy, P. G. Cottis, and Bjorn Ottersten, “Power allocation in multibeam satellite systems: a two-stage multi-objective optimization,” IEEE Transactions on Wireless Communications, 14(6):3171-3182, jun 2015.

[4] A. Paris, I. del Portillo, B. G. Cameron, and E. F. Crawley, “A genetic algorithm for joint power and bandwidth allocation in multibeam satellite systems,” in 2019 IEEE Aerospace Conference, 2019.

[5] F. R. Durand, and T. Abrão, “Power allocation in multibeam satellites based on particle swarm optimization,” International Journal of Electronics and Communications, 78:124-133, 2017.

[6] P. V. Rodrigues Ferreira et al., “Multiobjective reinforcement learning for cognitive satellite communications using deep neural network ensembles,” IEEE Journal on Selected Areas in Communications, 36(5):1030-1041, 2018.

[7] X. Hu, S. Liu, R. Chen, W. Wang, and C. Wang, “A deep reinforcement learning-based framework for dynamic resource allocation in multibeam satellite systems,” IEEE Communications Letters, 22(8):1612-1615, 2018.

[8] Digital Video Broadcasting (DVB). Implementation guidelines for the second generation system for broadcasting, interactive services, news gathering and other broadband satellite applications; Part 2 – S2 extensions (DVB-S2X). Technical report, 2015.

[9] R. S. Sutton, and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[10] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” in Advances in neural information processing systems, 1057-1063, 2000.

[11] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint, arXiv:1707.06347, 2017