Beam-to-Satellite Scheduling for High Throughput Satellite Constellations Using Particle Swarm Optimization

Nils Pachler, Edward F. Crawley, Bruce G. Cameron
Aeronautics and Astronautics
Massachusetts Institute of Technology
Cambridge, MA, USA
{pachler, crawley, bcameron}@mit.edu

Abstract*—The next generation of satellite communications will be characterized by a paradigm shift that will transform a traditionally static market into a frequently shifting environment. Fluctuating demand, highly flexible payloads, and the usage of non-geostationary orbits will boost the constellations' capacity to unthinkable limits, at the cost of additional complexity. This novel operational context carries unique problems that were non-existent in earlier stages of this industry. In this paper, we formulate and solve one of these novel problems, the Beam-to-Satellite scheduling problem, which focuses on deciding when to activate and deactivate a specific beam on a particular satellite. First, we describe the problem in terms of its scheduling variables, time-related constraints, and objective function, based on a combination of load balancing between the satellites and interference minimization. Second, we derive a linear-integer programming formulation of the problem, which can be optimally solved using common mathematical solvers. Given that those are computationally infeasible for high-dimensional scenarios (i.e., >200 beams), we then propose a single-objective PSO implementation. Finally, we test the algorithm over different high-dimensional scenarios taken from a realistic dataset, with tens of thousands of beams, provided by a satellite operator. Our PSO approach proves to be an effective technique to scan the search space and reach a satisfactory solution in a reasonable time. Using the same dataset, we benchmark the PSO implementation against heuristic solutions and show that it improves by between 39% and 73% the resource consumption overhead and around 30% the demand balancing between the satellites. In addition, we also demonstrate that it outperforms other metaheuristics, such as genetic algorithms and probabilistic algorithms, over all scenarios considered.

TABLE OF CONTENTS

1. INTRODUCTION

Motivation

The landscape of satellite communications will undergo a deep transformation in the following years. Large-scale constellations ranging from tens to thousands of satellites (e.g., SES, SpaceX, Amazon), the disruption of novel highly flexible payloads (e.g., steerable phased array antennas, adaptable modulation and coding strategies), and more fluctuating and varying demand (e.g., more Internet-driven services in isolated areas, streaming capabilities in planes and ships) will convert a traditionally static operation into frequently shifting, high-dimensional environment. Furthermore, the usage of non-geosynchronous orbits (NGSO) implies a time-dependency that was non-existent in prior generations of this industry. As former manual operations become unfeasible and obsolete under the new high-dimensional environment, the need for novel approaches becomes clear. In this line, major focus has been put during recent years, both in academia and industry, to deal with the dynamic resource allocation (RA) problem and its peculiarities.

While the RA in the context of satellite communications encloses many different sub-problems, in this paper we focus on the Beam-to-Satellite scheduling problem, which is defined as deciding when to start and stop serving a specific user from a particular satellite. Given that in the next generation of constellations, most positions on Earth will have two or more satellites visible at all times, the Beam-to-Satellite scheduling needs to decide which satellite is serving each user at each time and when to transition from satellite to satellite (handover). In addition to the time-dependency, the technology enhancements allow operators to use hundreds or even thousands of beams, which makes the Beam-to-Satellite scheduling problem a deeply complex, high-dimensional problem. In this context, artificial intelligence (AI) offers attractive techniques to deal with the additional intricacies.

While many traditional satellite resources have already been studied under an AI lens and countless different competitive solutions have been proposed, some of the novel time-dependent problems have yet to be studied by the research community. In this work, we aim to close part of this gap by studying the Beam-to-Satellite scheduling problem and proposing a particle swarm optimization (PSO) algorithm to solve it. Based on a combination of load balancing between the satellites and interference minimization, we benchmark and contrast the proposed algorithm against other commonly used AI and traditional techniques under realistic conditions in high-dimensional scenarios.

978-1-6654-3760-8/22/$31.00 ©2022 IEEE

Literature review

The RA problem for satellite communications has gained the interest of many researchers during the last decades. While power allocation and frequency assignment are the most studied sub-problems [1–5], the beam placement problem (i.e., the problem of clustering users and deciding how many beams to use) adapted to the new conditions has also been given some attention recently [6, 7]. On a minor scale, some works also study the beam shaping problem [8] and the beam-gateway routing problem [9]. However, the amount of research outside of these sub-problems is scarce. Because it only arises due to the new flexibility, the authors were not able to find detailed studies regarding the Beam-to-Satellite scheduling problem as described in this work. Note that, up until this point, most operators had only one satellite visible at all times (either by using a single GEO satellite or NGSO constellations with few satellites), which makes the problem trivially solvable. Only for flexible large-scale constellations is this problem applicable, which explains the general lack of studies on this issue.

Nevertheless, scheduling problems are common in other engineering areas. Originally formulated to maximize the effectiveness of computing architectures [10], scheduling problems expanded rapidly into common industrial operations such as production optimization [11] and goods handling [12]. By discussing the detailed timings of clinical procedures, [13] shows that scheduling problems can also be found outside of the engineering field. On a more recent timescale, many studies have tried applying AI techniques to solve scheduling problems. Authors in [14] describe the advantages and disadvantages of three AI techniques (genetic algorithms, neural networks, and fuzzy logic) by contrasting 10 different scheduling-related studies. Using a more descriptive approach, the survey reported in [15] analyzes over 50 different works and concludes that, while genetic algorithms (GA) is the most used technique, it is unclear which method performs best on equal conditions.

The success of metaheuristics in other similar scheduling problems motivates their usage for the novel Beam-to-Satellite scheduling problem. Specifically, PSO algorithms have proven successful on several applications [16, 17] due to their efficient search of the solution space even with a limited number of individuals. Furthermore, they have also achieved satisfactory results on diverse instances within the RA problem in satellite communications [18, 19]. In general, PSO algorithms prove to have a fast convergence and adequate scalability properties [1, 20], which makes them suitable for high-dimensional problems.

Paper objectives

The objectives of this work are threefold: present and formulate the Beam-to-Satellite scheduling problem in the context of satellite communications, solve it using a single-objective PSO, and assess the quality of the solution by testing it under realistic conditions in high-dimensional scenarios and comparing it to other commonly-used techniques.

Paper structure

The remainder of the paper is organized as follows: Section 2 presents the single-objective formulation for the Beam-to-Satellite scheduling problem; Section 3 introduces the necessary PSO flight mechanisms and initialization considerations; Section 4 shows the results of this work in terms of algorithm convergence and performance comparison with other techniques; and, finally, Section 5 remarks the conclusions of this work.

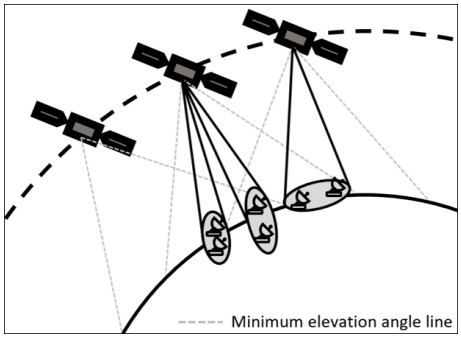


Figure 1: Given that all elements are time-independent, the satellite conditions (i.e., beam visibility and demand) only depend on the geographical position.

2. THE BEAM-TO-SATELLITE SCHEDULING PROBLEM

This section presents the Beam-to-Satellite scheduling problem as well as a single-objective formulation of the problem based on a combination of load balancing and interference minimization.

Problem description

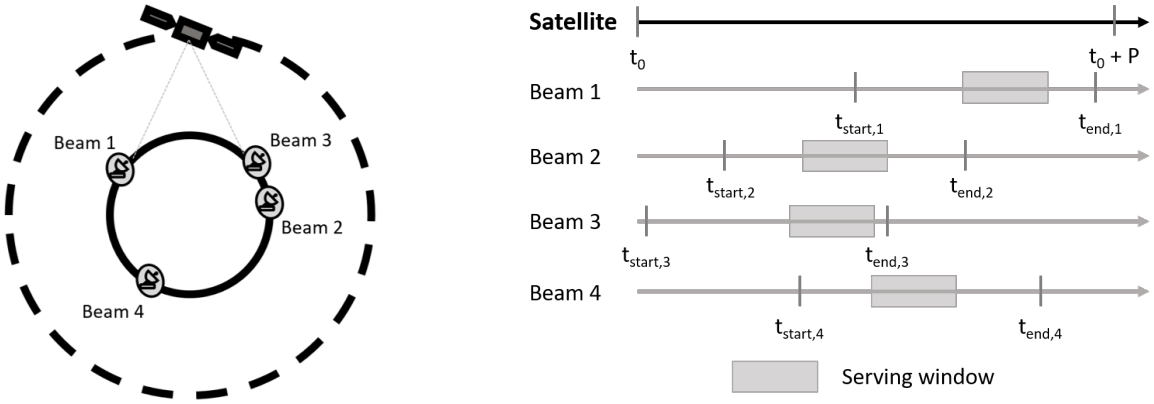
In this work we consider a single-plane, multi-beam, high-throughput satellite constellation with N_b prefixed beams and N_s satellites. An example constellation is shown in Figure 1. The rate of change of the right ascension of the ascending node (RAAN) of the orbit is assumed to be sufficiently low so that the orbit can be assumed to be periodic within a single satellite rotation. The altitude of the orbit h (and, thus, the period P), as well as the position lon_i, lat_i of each beam i is known. As a simplification, we assume that all beams consume their peak demand d_i at all times. Since the movement of the satellites do not allow them to continually serve a beam, beams need to switch satellites over time. We assume that each beam is served by a each satellite a total time of $T_s = P/N_s$ per rotation and that the serving window is continuous (i.e., cannot be split into multiple sub-windows).

We define the Beam-to-Satellite scheduling problem as deciding which satellite is serving each beam at each time (i.e., when to place the serving window) so that the beam is continually served and the satellite resource consumption is minimized. By minimizing resource consumption, the aim of this problem is to increase the overall system's capacity by efficiently managing the system's resources.

Since the orbital ground track can be assumed to be fixed (due to small RAAN change), the beams are static, and their demand is assumed constant, all satellites observe the same conditions in terms of beam visibility and demand with a time-shift of T_s. In other words, the conditions regarding the sub-group of visible beams only depend on the geographical position. Consequently, it is sufficient to solve for a single satellite rotation and propagate the solution to the rest of the constellation with the required time-shift.

Problem formulation

Given the previous statement, we can define a reference satellite $\mathcal{S}$ that, at t_0, is located at position $p_s(t_0)$. Since the



(a) Satellite visualization

(b) Time-window definitions

Figure 2: Example of the time visibility and serving window for 4 beams.

RAAN variation is low, $p_s(t_0 + P) \approx p_s(t_0)$. For a specific beam i, we define the starting time $t_{start,i}$ as the first time in which the visibility conditions allow satellite $\mathcal{S}$ to serve beam i (i.e., the satellite $\mathcal{S}$ sees beam i with an elevation angle equal to the minimum elevation angle). Similarly, $t_{end,i}$ is the second (and final) time after which satellite $\mathcal{S}$ can no longer serve beam i. A 4-beam example of the definition of $t_{start,i}$ and $t_{end,i}$, as well as the serving window, is shown in Figure 2.

Assuming all beams are served a total window of T_s by a satellite, we will define $t_{stop,i}$ ($t_{stop,i} = t_{end,i} - T_s$) as the final time in which we can start serving a specific beam while satisfying the window constraints. With this definitions, we can redefine the Beam-to-Satellite scheduling problem as finding, for each beam, the initial serving time t_i, such that $t_{start,i} \leq t_i \leq t_{stop,i}$, so that satellite resource consumption is minimized. Note that the continual service constraint is satisfied by definition: since all satellites have the same scheduling plan shifted T_s, and satellite $j+1$ takes T_s to get to the same position as satellite j, the beam is served continually.

Problem objective

Given that the satellite resources are limited (e.g., limited frequency spectrum, limited power usage), minimizing the usage of resources corresponds to avoiding overlapping between beams, which is equivalent to balance the load of the beams between satellites to avoid additional consumption. In other words, two beams that are served by the same satellite have to compete for the same resources, which implies a consumption overhead that could be avoided by using different satellites. For this purpose, we define that two beams $\{i, j\}$ overlap if, at some point in time, they are both active on the reference satellite $\mathcal{S}$:

$$y_{ij} = \begin{cases} 1 \text{ if } \begin{Bmatrix} t_i < t_j + T_s \\ t_j < t_i + T_s \end{Bmatrix} \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

While y_{ij} denotes if any two beams are overlapping, the overhead induced by two highly-demanding overlapping beams will be significantly higher than that from two low-demanding beams. To accommodate this factor, we define the frequency consumption of a beam as:

$$f_i = \frac{d_i}{\Gamma_i} \tag{2}$$

Where Γ_i is the spectral efficiency if beam i. To avoid having to solve the power allocation problem to find the exact value of Γ_i, we will assume a standard value Γ for all beams. Next, we define the number of frequency channels consumed η_i as:

$$\eta_i = \left\lceil \frac{f_i}{c} \right\rceil \tag{3}$$

Where c is the bandwidth of a channel, defined by the satellite architecture. Then, we define the number of overlapping channels as:

$$B_{ij} = \min(\eta_i, \eta_j) \tag{4}$$

B_{ij} represents the amount of interfering channels between beam i and beam j and scales with the demand of the least demanding beam. Finally, two beams that are geographically sufficiently close so that they may interfere have additional overhead. This will be represented using an adjacency matrix:

$$A_{ij} = \begin{cases} N_r \text{ if close enough to interfere} \\ 1 \text{ otherwise} \end{cases} \tag{5}$$

Similarly to [4], in this work we will assume that two beams are close enough to interfere if, at any point in time where the beams can be served by the same satellite, the angular distance between the center-line of each beam is less than four times the half-cone angle of the beam. The overhead factor induced from two beams that can interfere is equal to the frequency reuse factor N_r. With all these elements, we can define the objective function as:

$$\min \sum_{i,j,i \neq j} y_{ij} B_{ij} A_{ij} = \sum_{i,j,i \neq j} y_{ij} C_{ij} \tag{6}$$

Then, the Beam-to-Satellite scheduling problem can be formally defined as:

$$\begin{aligned} \min \quad & \sum_{i,j,i \neq j} y_{ij} C_{ij} \\ \text{s.t.} \quad & y_{ij} = \begin{cases} 1 \text{ if } \begin{Bmatrix} t_i < t_j + T_s \\ t_j < t_i + T_s \end{Bmatrix} \\ 0 \text{ otherwise} \end{cases} \\ & t_{start,i} \leq t_i \leq t_{stop,i} \end{aligned} \tag{7}$$

While this formulation is not linear, the transformations shown in Appendix A allow for a linearization of the problem.

3. THE PSO IMPLEMENTATION

This section presents the particle swarm optimization (PSO) algorithm used to solve the scheduling problem. First, the basic elements of PSO, such as particle position and velocity, are introduced. Then, we define the position of the particle in terms of the decision variables of the Beam-to-Satellite problem. Finally, the specific flight mechanisms are detailed and some initialization observations are made.

Concepts of particle swarm optimization

PSO is a metaheuristic, population-based, and iteration-based algorithm inspired by the movement of bird flocks [21]. Each individual (also known as particle) is characterized by a position and a velocity. The position is a set of coordinates that represent a solution in the search space, while the velocity denotes the rate of change of these coordinates. At each iteration, the position of each particle is updated according to their previous position and current velocity. Similarly, the velocity component is also updated according to the current position of all particles in the population. By prioritizing interesting regions, the intention is that the set of particles (also known as swarm) converges to optimal or close-to-optimal areas of the search space.

Three factors motivate the usage of PSO for the Beam-to-Satellite scheduling problem: 1) PSO offers fast convergence and adequate scalability properties [1] due to the simplicity of operations and the update of all elements at once, 2) PSO works well for continuous decision variables (note that, in Equation 7, y_{ij} is not a decision, but rather an auxiliary variable), and 3) PSO has already proven successful in other scheduling problems [16, 17] and RA sub-problems [18, 19].

PSO on the Beam-to-Satellite scheduling problem

In order to apply the PSO for the Beam-to-Satellite scheduling problem we need to define the position in terms of the decision variables of the problem. For this purpose, we define the position as an array with as many coordinates as beams (i.e., length of N_b) where each coordinate represents the starting point of the serving window of beam i (t_i).

The whole search of the algorithm is governed by the objective function defined in Equation 7. This Equation defines the objective value of each particle. In order to minimize this value, we define that a particle is *better* than another if their objective value is lower. A particle is the *best* particle in the swarm if its objective value is the lowest with respect to all other particles.

Flight mechanisms and particle movement

One of the most important factors to obtain a successful PSO implementation is to properly update the velocity of each particle so that promising regions are prioritized. Although many different approaches have been proposed throughout the years, most of them coincide in three main factors:

• Global pull G: each particle is attracted towards the best particle in the swarm. For particle p, the formula to compute the global pull for iteration t is:

$$G_{p,t} = \alpha g(x_g - x_{p,t-1}) \tag{8}$$

Where α is a random value taken from a random uniform distribution between $[0, 1)$, g denotes the weight factor of the global pull, x_g is the best position of the swarm so far, and $x_{p,t-1}$ is the current position of particle p.

• Local pull L: each particle is attracted towards the best position it has visited so far. In a similar fashion as the global pull, the formula to compute the local pull for iteration t is:

$$L_{p,t} = \beta l(x_{p,\forall t} - x_{p,t-1}) \tag{9}$$

Where β is a random value taken from a random uniform distribution between $[0, 1)$, l denotes the weight of the local pull, $x_{p,\forall t}$ is the best position of the current particle so far, and $x_{p,t-1}$ is the current position of particle p.

• Inertia pull I: represents the tendency of a particle to continue in a specific direction. In this case, the inertia pull for particle p at iteration t is simply the previous velocity $v_{p,t-1}$ multiplied by a decay factor w:

$$I_{p,t} = wv_{p,t-1} \tag{10}$$

Once these three factors are computed, we can proceed to update the velocity and position according to the formulas:

$$\begin{aligned} v_{p,t} &= G_{p,t} + L_{p,t} + I_{p,t} \\ x_{p,t} &= x_{p,t-1} + v_{p,t} \end{aligned} \tag{11}$$

As done in many PSO applications [16, 19], we limit the maximum speed for each coordinate to avoid flying by interesting regions without exploring them. The explained mechanism is sufficient to execute a simple version of the PSO algorithm. However, to avoid early convergence and improve performance, we included the concept of mutation to the PSO implementation. This idea has already been applied with satisfactory results to other PSO implementations [19, 22]. After the flight, each particle has a small probability p_{mut} of suffering a mutation that affects a fraction p_{mutx} of its coordinates by changing their value to a random, valid new coordinate.

Initialization and other considerations

In order to obtain solutions with sufficient diversity across the search region, all initial positions and velocities are selected from a uniform random distribution within the valid range. As in this problem we have no initial hint on where the interesting regions are, random initialization increases the probability of rapidly finding an interesting region compared to other heuristic approaches, and avoids getting trapped in manually-induced local optima.

In addition to the concept of mutation, we also use the notion of selection in our PSO implementation. Before a specific iteration, we save the current state of the swarm and, after the iteration, both previous and new particles follow a selection process in which only the best particles are kept. Specifically, we use a simple fitness comparison in which only the particles with higher fitness (better objective value) are selected for the next iteration.

4. RESULTS

This section presents and discusses the results and performance of the PSO implementation. First, the test constellation, as well as the benchmarking scenarios are introduced. Second, alternative solutions are briefly discussed. Third, the convergence results for the PSO are shown. Finally, the performance comparison is detailed.

Test constellation and benchmark scenarios

The constellation used for the experiments is inspired in the O3b mPower constellation [23, 24], which consists of ten

satellites at 8,062 kilometers operating in an equatorial orbit. As specified in the filings, the minimum elevation angle is assumed to be 10°. The number of frequency re-use N_r (Equation 5) is assumed to be 10. The average spectral efficiency Γ (Equation 2) will be set to 2.

The user model is provided by SES S.A. and consists of tens of thousands of users distributed across the world. The peak demand and position of each user is defined by the model. To cluster the users into beams, we applied the algorithm explained in [7]. For comparison purposes and given that this algorithm provides a Pareto Front of solutions with different trade-offs between number of beams and resource consumption, we selected four specific solutions: scenario A with ≈ 4000 beams, scenario B with ≈ 6000 beams, scenario C with ≈ 10000 beams, and scenario D with ≈ 20000 beams. While the four scenarios aim to serve the same demand, we want to analyze the effects that the user density has on the final solution. Together with this and to better analyse the scaling properties of the implementation, we divided each scenario into seven different cases: 1) 50 beams, 2) 100 beams, 3) 200 beams, 4) 500 beams, 5) 1000 beams, 6) 2000 beams, and 7) all beams. The process of selecting a subset of beams follows the subsequent logic: 1) select randomly one beam, 2) select the k closest beams as required. We applied all algorithms to all scenarios and cases, except when the combination was computationally or memory-wise infeasible. All the results shown in this work are the average of 5 independent runs. The convergence results and load balancing results for the PSO have been executed on scenario B.7.

Alternative solutions

This subsection briefly discusses other alternative solutions to the Beam-to-Satellite scheduling problem that will serve as a comparison to assess the performance of the PSO implementation.

Heuristic approach (HEU)—Although many heuristics can be formulated for this problem, we will use a simple solution for which all variables are initialized to the center of its visibility window. In other words, each beam is served by the closest satellite, without taking any further interference or balancing consideration.

Integer programming (IP)— As show in Appendix A, the problem can be transformed into a linear-integer programming formulation, for which commercial mathematical solvers can be used. For our implementation, we used the Gurobi® solver. This approach achieves the optimal solution, but has scalability issues in high-dimensional scenarios.

Cross-entropy method (CE)—This method is based on keeping an explicit probability distribution for each variable. At each iteration, the space region is sampled and a new probability distribution is selected based on the best samples, until convergence. The implementation of this method is based on the pseudo-code presented in Section 8.4 of [25].

Genetic algorithm (GA)— Similar to PSO, GA is a meta-heuristic, population-based algorithm. GA is inspired by evolution procedures, in which a specific population evolves over time by the crossing of its members and the mutation of some individuals. The solution is coded into the individuals' genes and new individuals represent either union or alteration of previous solutions. After each generation, only the fittest individuals are kept. Specifically, the selection process follows a similar logic as explained in Section 3 for the PSO.

Parameter	Value
Iterations	200
Swarm size	20
Global pull weight (g)	2
Local pull weight (l)	2
Inertia weight (w)	0.729844 [26]
Mutation probability (p_{mut})	15%
Mutated coordinates (p_{mutx})	1%
Maximum speed	$0.03P$

Table 1: PSO parameter selection. P is defined as the period of the orbit.

Parameter	Value
Number of points	50
Number of references	10

Table 2: Cross-entropy parameter selection

Parameter	Value
Generations	200
Population size	20
Crossing probability	80%
Mutation probability	20%
Mutated genes	1%

Table 3: GA parameter selection

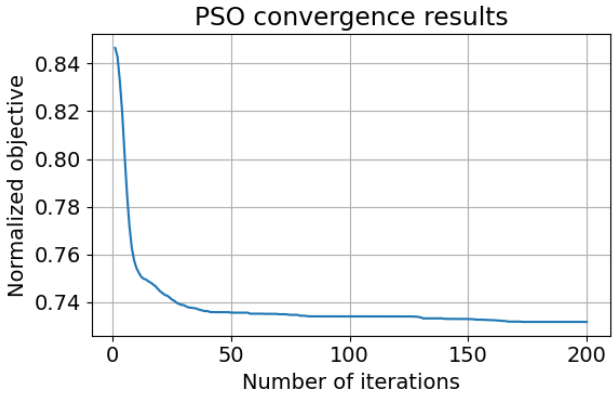


Figure 3: Evolution of the best objective value for a single PSO execution in scenario B.7. All values have been normalized against the heuristic value.

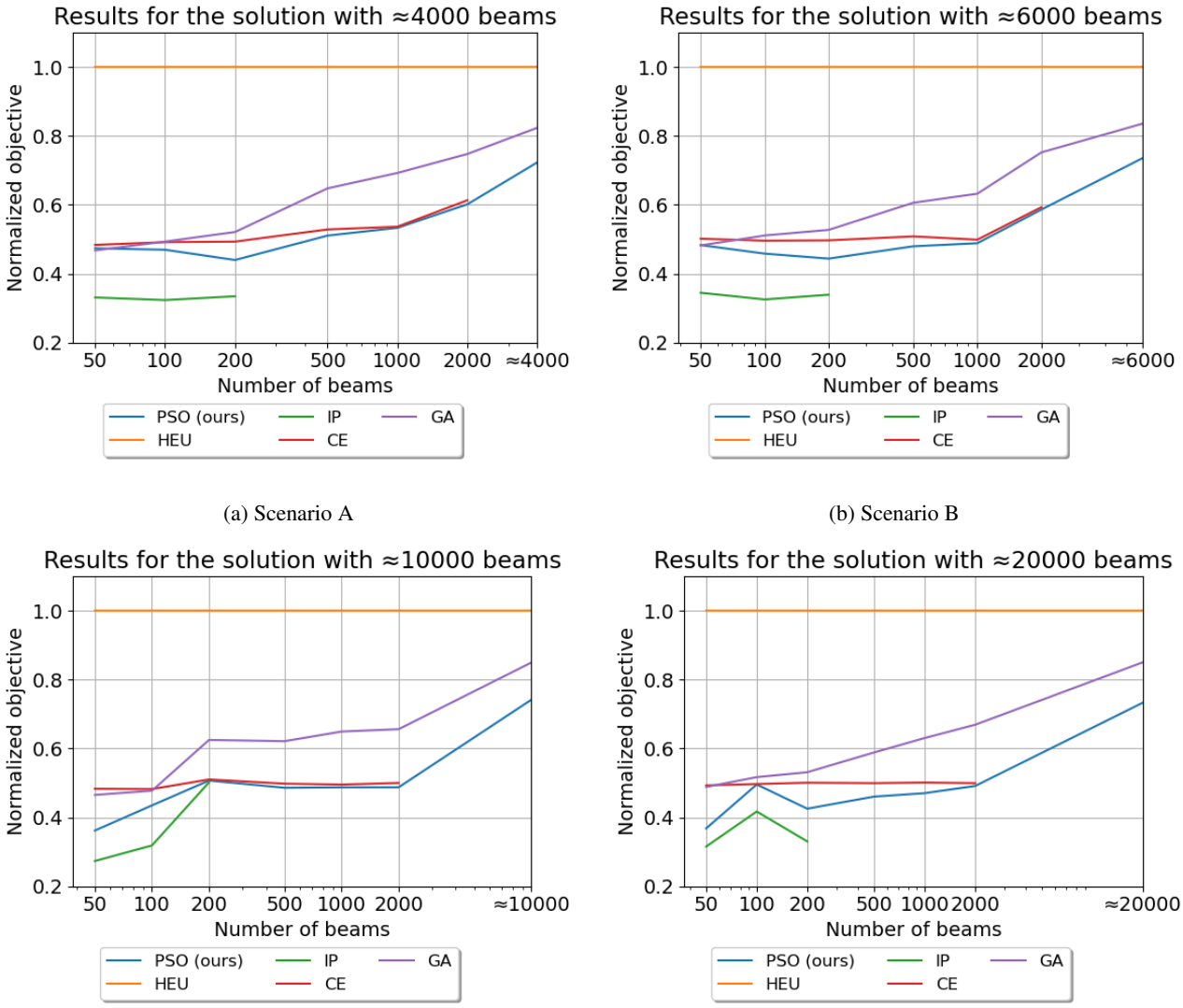
For this particular problem, we represent each timing-related variable with one gene.

Algorithm hyperparameters

Tables 1, 2, and 3 contain a summary of the relevant parameters for the PSO, cross-entropy, and GA executions, respectively.

Convergence results

Figure 3 shows the evolution of the algorithm throughout a single PSO run on scenario B.7. All values have been



(a) Scenario A

(b) Scenario B

(c) Scenario C

(d) Scenario D

Figure 4: Performance comparison over all 4 scenarios and 7 cases. Results have been normalized against the heuristic value of each specific case. With the formulation presented, IP and CE have scalability issues with more than 200 and 2,000 beams, respectively.

normalized against the heuristic value for the same scenario and case. Similarly to other PSO implementations, the algorithm achieves large improvements (from 0.85 to 0.74 on the objective value) during the first portion of the execution (25 iterations). After that, the improvement rate slows down, only being able to achieve around 0.01 in the following 175 iterations. Furthermore, after iteration 75, the algorithm has largely converged, and further changes are only a result of the particle mutation implementation. We conclude that the PSO has a fast convergence for this problem and that 200 iterations are sufficient to generate a reasonably good solution since further iterations only change the result marginally.

Performance comparison

While the previous results show how the solution evolves and improves over time until convergence, it is hard to assess the quality of the algorithm for the Beam-to-Satellite scheduling problem without further analysis. For this purpose, we implemented other commonly used techniques to compare the performance of our proposed solution. Figure 4 presents the results for all algorithms on the four scenarios. Specifically, Table 4 details the results for scenario B. The behaviour of each algorithm can be summarised as follows:

- Particle Swarm Optimization (PSO): it dominates all other implementations in high dimensional scenarios (i.e, $>$ 200 beams). Thanks to its fast convergence, it is able to rapidly achieve a satisfactory solution, improving between 39% and 73% the heuristic baseline in all cases.
- Heuristic (HEU): although it gives the worst result, the heuristic serves as a baseline to compare the other approaches. This implementation also gives an intuition on how simple heuristics perform against more complicated approaches on this problem.
- Integer Programming (IP): since it gives the optimal solution, the IP implementation outperforms all other algorithms in low dimensional scenarios. Nevertheless, it is unable to scale: due to the quadratic dependency shown in Appendix

Algorithm	Number of beams						
	50	**100**	**200**	**500**	**1000**	**2000**	**≈6000**
PSO (ours)	0.48	0.46	0.44	**0.48**	**0.49**	**0.59**	**0.74**
HEU	1.00	1.00	1.00	1.00	1.00	1.00	1.00
IP	**0.34**	**0.32**	**0.34**	-	-	-	-
CE	0.50	0.50	0.50	0.51	0.50	**0.59**	-
GA	0.48	0.51	0.53	0.61	0.63	0.75	0.84

Table 4: Detailed performance results for scenario B (≈6,000 beams). For a better comparison, all results have been normalized against the heuristic result of the specific case.

A, the algorithm does not converge within reasonable time for cases with more than 200 beams.

- Cross-Entropy method (CE): although it gives far-from-optimal solutions in low dimensional cases, it provides good approximations in the range between 500 and 2000 beams. Given that it needs to keep a matrix that scales quadratically with the number of variables, it also has scalability issues in cases with more than 2000 beams.
- Genetic Algorithm (GA): similarly to other RA sub-problems in the satellite communications field [1], the GA suffers from slow convergence in the Beam-to-Satellite scheduling problem. Therefore, the algorithm is unable to provide a sufficiently good solution within reasonable time, especially in high dimensional cases, obtaining a 10% drawback with respect to the PSO.

In conclusion, the PSO outperforms all other techniques in high dimensional scenarios, achieving between 39% and 73% improvement over the heuristic approach, between 1% and 7% over the cross-entropy method, and between 10% and 15% over a standalone GA version, in cases with more than 200 beams. In general terms, we conclude that the PSO provides solutions with a better load balance between the satellites and less interference constraints between the beams.

As mentioned in Section 2, distributing the demand load between the satellites is one of the key drivers of the optimization process. To further understand the behaviour of the algorithm, it is useful to highlight the load balancing results compared to the heuristic approach. Figure 5 shows the demand as an aggregation of all the active beams on the satellite over each longitude. As shown, the heuristic approach does not manage to balance the load of the beams which results in naturally dense areas being more loaded. This produces regions with very high peaks and regions with very low peaks. On the other hand, the PSO implementation reduces the standard deviation around 30% compared to the heuristic solution, dwindling both the high and low peaks to achieve a more balanced distribution.

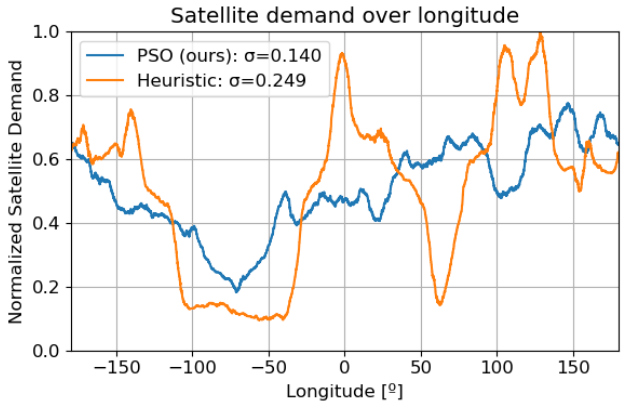


Figure 5: Load balancing results for a single-run PSO and heuristic solutions on scenario B.7. All results have been normalized against the worst heuristic load.

5. Conclusions

This work presented and proposed a solution for the Beam-to-Satellite scheduling problem in high-throughput multibeam satellite communications constellations. First, we described and formulated the problem based on a single-objective function that combines load balancing between the satellites and interference minimization. In this line, we also performed a linearization of the problem to be solved with standard traditional solvers. Due to the poor scalability of these techniques, we then proposed a PSO implementation to solve the Beam-to-Satellite scheduling problem in high-dimensional scenarios. Finally, we introduced 4 alternative techniques and discussed the outcomes of the proposed solution both in terms of convergence results and performance comparison with the alternative options. The conclusions of this work can be summarized as follows:

- The Beam-to-Satellite scheduling problem for high throughput, multi-beam, uni-planar satellite constellations can be formulated as a linear-integer problem where the number of variables scales quadratically with the number of beams.
- The integer programming method is the only one able to find the optimal solution in low dimensional scenarios. However, due to their inability to scale, the PSO implementation dominates other solutions in high-dimensional scenarios.
- PSO outperforms all other techniques in cases with more than 200 beams, achieving between 39% and 73% resource consumption overhead reduction over the heuristic approach, between 1% and 7% over the cross-entropy method, and between 10% and 15% over a standalone GA version.
- In terms of load balancing, the proposed technique is able to reduce around 30% the demand imbalance between the satellites compared to a simple heuristic approach.

Possible future work will cover:

- Comparison with other AI techniques, such as reinforcement learning methods, for a more detailed insight on the problem and proposed solution.
- Expansion of the presented formulation to multi-plane satellite constellations.
- Inclusion of non-static factors, such as mobile beams or constellation drifting, to the formulation.

Acknowledgments

The authors would like to thank SES S.A. for their financial and non-financial support during the development of this work.

Appendices

A. Linear-integer formulation

Although the formulation presented is not linear, we can linearize it with some mathematical transformations. First,

we can define y_{ij}^+ and y_{ij}^- as:

$$y_{ij}^+ = \begin{cases} 1 \text{ if } t_i < t_j + T_s \\ 0 \text{ otherwise} \end{cases} \tag{12}$$

$$y_{ij}^- = \begin{cases} 1 \text{ if } t_j < t_i + T_s \\ 0 \text{ otherwise} \end{cases} \tag{13}$$

Then, y_{ij} is just the logical AND of y_{ij}^+ and y_{ij}^-. In other words:

$$\begin{aligned} 2y_{ij} &\leq y_{ij}^+ + y_{ij}^- \\ y_{ij} &\geq y_{ij}^+ + y_{ij}^- - 1 \\ y_{ij} &\in \{0, 1\} \end{aligned} \tag{14}$$

Next, we define M as a large number so that $M > |t_j + T_s - t_i| \; \forall i, j$. This allows us to write y_{ij}^+ and y_{ij}^- as:

$$\begin{aligned} M(y_{ij}^+ - 1) < t_j + T_s - t_i \leq M y_{ij}^+ \\ M(y_{ij}^- - 1) < t_i + T_s - t_j \leq M y_{ij}^- \\ y_{ij}^+, y_{ij}^- \in \{0, 1\} \end{aligned} \tag{15}$$

Now, we can rewrite the problem formulation as:

$$\begin{aligned} \min \quad & \sum_{i,j,i \neq j} y_{ij} C_{ij} \\ \text{s.t.} \quad & 2y_{ij} \leq y_{ij}^+ + y_{ij}^- \\ & y_{ij} \geq y_{ij}^+ + y_{ij}^- - 1 \\ & M(y_{ij}^+ - 1) < t_j + T_s - t_i \leq M y_{ij}^+ \\ & M(y_{ij}^- - 1) < t_i + T_s - t_j \leq M y_{ij}^- \\ & t_{start,i} \leq t_i \leq t_{stop,i} \\ & y_{ij}, y_{ij}^+, y_{ij}^- \in \{0, 1\} \end{aligned} \tag{16}$$

Which is a linear-integer formulation of the problem. It is important to note that, although the problem is linear, the number of variables scales quadratically with the number of beams $\mathcal{O}(3n^2 + n)$, which may pose scalability problems in traditional mathematical solvers when the number of beams is too high.

REFERENCES

[1] J. J. G. Luis, N. Pachler, M. Guerster, I. del Portillo, E. F. Crawley, and B. G. Cameron, "Artificial Intelligence Algorithms for Power Allocation in High Throughput Satellites: A Comparison," in *2020 IEEE Aerospace Conference*, 2020.

[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*, vol. 64, no. 2, pp. 266–280, 2017.

[3] X. Liao, X. Hu, Z. Liu, S. Ma, L. Xu, X. Li, W. Wang, and F. M. Ghannouchi, "Distributed intelligence: A verification for multi-agent drl-based multibeam satellite resource allocation," *IEEE Communications Letters*, vol. 24, no. 12, pp. 2785–2789, 2020.

[4] N. Pachler, M. Guerster, I. del Portillo, E. F. Crawley, and B. G. Cameron, "Static beam placement and frequency plan algorithms for LEO constellations," *International Journal of Satellite Communications and Networking*, 2020.

[5] A. I. Aravanis, B. S. MR, P.-D. Arapoglou, G. Danoy, P. G. Cottis, and B. Ottersten, "Power allocation in multibeam satellite systems: A two-stage multi-objective optimization," *IEEE Transactions on Wireless Communications*, vol. 14, no. 6, pp. 3171–3182, 2015.

[6] A. Kyrgiazos, B. Evans, and P. Thompson, "Irregular beam sizes and non-uniform bandwidth allocation in hts multibeam satellite systems," in *31st AIAA International Communications Satellite Systems Conference (ICSSC)*, 2013.

[7] N. Pachler, E. F. Crawley, and B. G. Cameron, "A genetic algorithm for beam placement in high-throughput satellite constellations," in *IEEE Cognitive Communications for Aerospace Applications Workshop*, 2021.

[8] R. Alinque, "Joint optimization of beam placement and shaping for multi-beam high throughput satellite systems using gradient descent," Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 2020.

[9] Y. Rao and R. Wang, "Performance of qos routing using genetic algorithm for polar-orbit leo satellite networks," *AEU-International Journal of Electronics and Communications*, vol. 65, no. 6, pp. 530–538, 2011.

[10] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[11] B. Fleischmann and H. Meyr, "The general lotsizing and scheduling problem," *Operations-Research-Spektrum*, vol. 19, no. 1, pp. 11–21, 1997.

[12] C. F. Daganzo, "The crane scheduling problem," *Transportation Research Part B: Methodological*, vol. 23, no. 3, pp. 159–175, 1989.

[13] J. H. May, W. E. Spangler, D. P. Strum, and L. G. Vargas, "The surgical scheduling problem: Current research and future opportunities," *Production and Operations Management*, vol. 20, no. 3, pp. 392–405, 2011.

[14] A. Ansari and A. A. Bakar, "A comparative study of three artificial intelligence techniques: Genetic algorithm, neural network, and fuzzy logic, on scheduling problem," in *2014 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology*. IEEE, 2014, pp. 31–36.

[15] B. Çaliş and S. Bulkan, "A research survey: review of ai solution strategies of job shop scheduling problem," *Journal of Intelligent Manufacturing*, vol. 26, no. 5, pp. 961–973, 2015.

[16] T.-L. Lin, S.-J. Horng, T.-W. Kao, Y.-H. Chen, R.-S. Run, R.-J. Chen, J.-L. Lai, and I.-H. Kuo, "An efficient job-shop scheduling algorithm based on particle swarm optimization," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2629–2636, 2010.

[17] P. A. Varthanan, N. Murugan, and G. M. Kumar, "A simulation based heuristic discrete particle swarm algorithm for generating integrated production–distribution plan," *Applied Soft Computing*, vol. 12, no. 9, pp. 3034–3050, 2012.

[18] F. R. Durand and T. Abrão, "Power allocation in multibeam satellites based on particle swarm optimization,"

AEU-International Journal of Electronics and Communications, vol. 78, pp. 124–133, 2017.

[19] N. Pachler, J. J. G. Luis, M. Guerster, E. F. Crawley, and B. G. Cameron, "Allocating power and bandwidth in multibeam satellite systems using particle swarm optimization," in *2020 IEEE Aerospace Conference*, 2020.

[20] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, vol. 3. IEEE, 1999, pp. 1945–1950.

[21] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*. Ieee, 1995, pp. 39–43.

[22] N. Higashi and H. Iba, "Particle swarm optimization with gaussian mutation," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706)*. IEEE, 2003, pp. 72–79.

[23] O3b Limited. (2019) SAT-MOD-20200526-00058. [Online]. Available: http://licensing.fcc.gov/myibfs/forwardtopublictabaction.do?file_number=SATMOD2020052600058

[24] SES S.A. (2021) O3b mPower. [Online]. Available: https://o3bmpower.ses.com/

[25] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for optimization*. Mit Press, 2019.

[26] J. C. Bansal, P. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *2011 Third world congress on nature and biologically inspired computing*. IEEE, 2011, pp. 633–640.

BIOGRAPHY

Nils Pachler de la Osa *is a 2nd year Gaduate Student in the System Architecture Group, in the AeroAstro department at MIT. In 2019, he completed a double BS degree in Informatics Technology and Aerospace Engineering under the CFIS (Interdisciplinary Higher Education Centre) program from Universitat Politècnica de Catalunya. Nils carried out his Bachelor Thesis at MIT on dynamic allocation of resources in a multibeam satellite system. Previously, Nils cooperated with the ARCO research group at UPC, focused on hardware accelerators and neural networks, while working at LudiumLab, where he implemented video and audio codification algorithms and hardware encoders. His interests include optimization algorithms, autonomous systems, and Artificial Intelligence*

Prof. Edward F. Crawley *received an Sc.D. in Aerospace Structures from MIT in 1981. His early research interests centered on structural dynamics, aeroelasticity, and the development of actively controlled and intelligent structures. Recently, Dr. Crawley's research has focused on the domain of the architecture and design of complex systems. From 1996 to 2003 he served as the Department Head of Aeronautics and Astronautics at MIT, leading the strategic realignment of the department. Dr. Crawley is a Fellow of the AIAA and the Royal Aeronautical Society (UK), and is a member of three national academies of engineering. He is the author of numerous journal publications in the AIAA Journal, the ASME Journal, the Journal of Composite Materials, and Acta Astronautica. He received the NASA Public Service Medal. Recently, Prof Crawley was one of the ten members of the presidential committee led by Norman Augustine to study the future of human spaceflight in the US.*

Dr. Bruce Cameron *is a Lecturer in Engineering Systems at MIT and a consultant on platform strategies. At MIT, Dr. Cameron ran the MIT Commonality study, a 16 firm investigation of platforming returns. Dr. Cameron's current clients include Fortune 500 firms in high tech, aerospace, transportation, and consumer goods. Prior to MIT, Bruce worked as an engagement manager at a management consultancy and as a system engineer at MDA Space Systems, and has built hardware currently in orbit. Dr. Cameron received his undergraduate degree from the University of Toronto, and graduate degrees from MIT.*