applied sciences

MDPI

Article

Pilot Sequence Allocation Schemes in Massive MIMO Systems Using Heuristic Approaches

Everton Alex Matos [1], Robson Parmezan Bonidia [2], Danilo Sipoli Sanches [1], Rogério Santos Pozza [1] and Lucas Dias Hiera Sampaio [1,*]

[1] Computer Science Department, Universidade Tecnológica Federal do Paraná, Cornélio Procópio 86300-000, Paraná, Brazil; evertonalexweb@gmail.com (E.A.M.); danilosanches@utfpr.edu.br (D.S.S.); pozza@utfpr.edu.br (R.S.P.)
[2] Institute of Mathematics and Computer Science, Universidade de São Paulo, São Carlos 13566-590, São Paulo, Brazil; bonidia@usp.br
* Correspondence: ldsampaio@utfpr.edu.br

Abstract: This paper presents a comparison of different metaheuristic approaches applied to the pilot sequence allocation problem in Massive Multiple-Input Multiple-Output (MIMO) systems. A modified version of the Genetic Algorithm (GA) as well as different versions of the Particle Swarm Optimization (PSO) Algorithm are used to maximize the system spectral efficiency under an inter-cell interference regime. The metaheuristic parameters were optimized and computational simulations under different scenarios parameters were conducted to verify the system performance impact in terms of system spectral efficiency, minimum and maximum spectral efficiency per user and the cumulative distribution function (CDF) of the users spectral efficiencies. The main contributions of this work are: the creation of a public available dataset; heuristic parameters tuning; findings related to the impact of sub-optimal pilot sequence allocation to the users in terms of maximal and minimal achievable user spectral efficiency and the robustness of some algorithms in scenarios with different system loadings.

Keywords: pilot sequences; resource allocation; Massive MIMO; heuristics

Citation: Matos, E.A.; Parmezan Bonidia, R.; Sipoli Sanches, D.; Santos Pozza, R.; Dias Hiera Sampaio, L. Pilot Sequence Allocation Schemes in Massive MIMO Systems Using Heuristic Approaches. *Appl. Sci.* **2022**, *12*, 5117. https://doi.org/10.3390/app12105117

Academic Editors: Peng-Yeng Yin, Jen-Chun Lee, Hua-Yi Lin, Ming-Chin Chuang, Youcef Gheraibia and Ray-I Chang

Received: 30 March 2022
Accepted: 13 May 2022
Published: 19 May 2022

1. Introduction

The increasing number of multimedia services and online platforms are reflected as a growth in the demand for connectivity and throughput worldwide. According to Cisco [1], in 2018 there were 3.9 billion Internet users globally, and it estimates 5.3 billion in 2023, representing 66% of the global population by that year according to [2]. Part of the global data traffic, 54%, will originate from mobile devices which will be responsible for 160 Exabytes (EB) per month of data traffic in 2025, a 321% growth when compared to the 38 EB per month in 2019 [3].

To support this demand, the wireless communications standard must evolve and increase the total available throughput and the supported number of connected devices, while keeping the average network latency to a minimum. These characteristics will even make new services possible, such as autonomous cars [4].

Massive MIMO is a technology incorporated in 5G networks which corresponds to the transmission of data through several antennas. Thomas L. Marzetta describes the use of this technology for any multi-user MIMO system with more than 16 antennas [5]. The proposed communications scheme relied on low complexity transceivers in the mobile devices and time division duplexing (TDD) to provide a reliable communication channel.

Within a Massive MIMO system, an arbitrary array of bits, known as a pilot sequence, is designated to each user and used in the uplink training process to simplify the signal detection routines in mobile terminals. However, the use of non-orthogonal sequences causes interference among users, which is known as the pilot contamination problem.

To guarantee orthogonality, the size of the pilot sequence must increase with respect to the number of users in the system. Hence, it is easy to verify that pilot sequences must be reused in multi-cell environments and high-density urban areas to avoid long periods of uplink training since as the number of orthogonal sequences increases the size of these sequences also increases.

This work proposes to solve the pilot sequence allocation problem in multi-cell scenarios in a centralized fashion with the maximization of the network spectral efficiency as a goal. The optimization problem is modeled into two different, but equivalent, ways: a binary optimization problem and an integer optimization problem. Meanwhile, to solve these optimization problems, we used different versions of the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) metaheuristics.

1.1. Related Works

Among different studies whose objectives are to minimize the pilot sequence contamination in TDD Massive MIMO systems, there is a considerable selection of methods, tools and techniques already tested, e.g., greedy and tabu search, smart pilot assignment, genetic algorithm (GA), deep learning, hybrid solutions and multi-objective approaches. To situate the contributions of this work, we present a chronological list of related work:

- In [6], the authors approach the pilot contamination on Massive MIMO systems with low complexity evolutionary algorithms comparing the performance in macro-cell scenarios: greedy search, tabu search and a hybrid solution.
- In [7], the authors proposed a new algorithm named Smart Pilot Assignment. The algorithm goal is to maximize all user signals to interference plus noise ratio (SINR) within a cell. The base station calculates the cell observed interference for each of the pilot sequences which users from neighbors' cells cause. The algorithm then assigns the pilot sequences following the rule: the user with the worst channel state receives the pilot sequence with the smallest observed interference. This process is repeated until each user has been assigned a pilot sequence, or the system has no pilot sequences to assign.
- In [8], the authors proposed a greedy search. At each iteration, a number of users are randomly selected from each cell to be assigned the same pilot sequence. The selection is made such that the group chosen maximizes their transmission rate. As the iterations grow, the number of selected users also grows. Likewise, the number of possible selected users at each iteration shortens due to the already accomplished pilot sequence assignments of previous iterations.
- In a different approach [9], the authors proposed the mitigation to pilot sequence contamination modeling the problem as an Epsilon-restricted optimization problem and then solving an eigenvalue decomposition problem through linear complexity.
- In [10], the authors proposed a simple pilot assignment algorithm based on the water-filling algorithm. The users with the best channels received the pilot sequences under the lowest interference effect. To achieve such, at each base station, both channel state information (CSI) array, which contains each user CSI, as well as the inter-cell interference observed by each pilot sequence, are sorted such that one is in increasing order and the other in decreasing order. The pilot assignment is the combination of indexes of these two arrays.
- In [11], the author used different algorithms to solve the pilot sequence assignment problem. The main objective is to maximize the system throughput where the GA was capable of obtaining the average maximum rate user with a smaller complexity. Furthermore, the GA solution as well as random assignment and exhaustive search are compared with the proposed schemes of [6].
- In [12], the authors proposed the assignment of orthogonal pilot sequences along with sectorization. The pilot sequences are re-utilized in the same cell, reducing sizes of the pilot sequence and optimizing spectral efficiency (SE), and Bayesian estimation is used to eliminate pilot contamination.

- In [13], the authors present an adaptive pilot sequence allocation algorithm which separate the users in a cell into two groups: one for users who suffer high interference from other cells and one for users under a low interference regime. The algorithm then assigns mutually orthogonal pilot sequences for all the users under the interference regime, while the other group of users shares the same set of pilot sequences.
- In [14], the pilot sequence allocation problem is solved using deep learning in the form a 3-layer perception neural network. The proposed scheme reaches 99.38% of the theoretical upper-bound performance and takes only 0.92 milliseconds to compute.
- In [15], an algorithm of clustering divides users into two groups, low and high interference. In the group with low interference, the pilots are re-utilized randomly, while high interference groups are grouped by propagation affinity.
- In [16], the authors mitigated the pilot sequence allocation problem through user categorization in high and low interference groups based on large-scale fading, where the high interfering users receive orthogonal pilot sequences, and non-orthogonal pilot sequences are allocated to users under a low interference regime. The authors propose the use of an edge-weighted interference graph to maximize the performance of users in the low interference bracket.
- In [17], the authors proposed the pilot sequence allocation allied to power allocation based on a Monte Carlo Tree Search Method (MCTS) which mitigates the pilot contamination. Moreover, the AlphaGo algorithm is used to play the proposed pilot allocation game, while the Markov Decision Problem (MDP) solves the power allocation problem.
- In [18], based on works [19,20] an adaptation of the particle swarm optimization algorithm to solve the joint pilot sequence allocation and power control problem in Massive MIMO systems was proposed. The authors aim to maximize the spectral efficiency with a limited number of pilot sequences based on coherence interval, while also taking power constraints into account.
- In [21], the authors proposed a joint pilot sequence allocation and antenna scheduling scheme to curb the effects of pilot sequence contamination in Massive MIMO systems where there are a limited number of antennas. To allocate the sequences to multiple users, they proposed rules using either geometric or arithmetic progression in the number of users using the same sequence. Furthermore, they compare their solution with a Greedy pilot sequence allocation scheme and the Smart Pilot Assignment algorithms.
- The work in [22] presents different pilot allocation solutions for cell-free scenarios. The first algorithm is based on the concept of random sequential adsorption using statistical physics, while the second one is an analytical approach of the first one. The authors also describe two centralized algorithms based on clustering principles to benchmark the proposed solutions. The results show that the distributed solutions have a competitive performance compared to the centralized ones, especially when the user density is high.

1.2. Contributions

Our study presented four new main contributions on the subject of Massive MIMO pilot sequence allocation problem:

- The dataset as well as the scripts used to achieve the results are public (Available at github.com/evertonalex/utfpr-ppgi-pilotcontamination, accessed on 16 May 2022);
- The heuristic parameters to the problem assessed were optimized;
- Six different heuristic approaches to the problem comparing it with the simplest solution were evaluated;
- Real scenarios under different parameters to establish the real impact of a pilot allocation scheme were simulated.

Furthermore, we present the differences between our work and those presented in the previous section:

- Two different mathematical models of the same practical problem were addressed;

- Multiple versions and modifications in the PSO algorithm were tested to verify their performance;
- Using The minimum spectral efficiency per user as a performance parameter was used;
- Different scenario parameters were tested to verify the impacts of the pilot sequence schemes in macro, micro and femto-cells.

1.3. Text Organization

This paper is organized as follows. In Section 2, the system model is described with the mathematical problem. In Section 3, the solution with optimization is applied with the evolutionary algorithms. In Section 4, the best parameters are presented and the results of simulations are shown. Finally, in Section 5, we offer the conclusion and final considerations.

2. System Model

We consider a Massive MIMO system with $L > 1$ cells using the same spectrum band and K_ℓ mobile terminals connected with its single base station which has $M >> K_\ell$ antennas. To acquire channel state information (CSI) at each coherence interval, each user in the system must send a pilot sequence through the uplink channel. The pilot sequence uplink signal received by the base station ℓ is a $M \times T_p$ matrix where T_p is the pilot sequence size, and it may be described as:

$$\mathbf{R}_\ell^u = \overbrace{\sum_{k=1}^{K_\ell} \sqrt{p_{k,\ell}}\, \mathbf{g}_{k,\ell,\ell}\, \mathbf{s}_{k,\ell}^H +}^{\text{Desired signal from cell users}} \underbrace{+ \sum_{\substack{j=1 \\ j\neq\ell}}^{L} \sum_{k'=1}^{K_j} \sqrt{p_{k',j}} \mathbf{g}_{k',j,\ell}\, \mathbf{s}_{k',j}^H}_{\text{Pilot Contamination / Interference}} + \boldsymbol{\eta}_\ell \tag{1}$$

where k, ℓ, k' and j are the user, cell of interest, interfering users and adjacent cell indexers, respectively. Moreover, $\mathbf{R}_\ell^u \in \mathbb{C}^{M \times T_p}$, $p_{k,\ell}$ is the transmission power of each user, $\mathbf{s}_{k,\ell} \in \mathbb{C}^{T_p \times 1}$ is the pilot sequence, whereas $(\cdot)^H$ is the Hermitian operator and is equivalent to the transposed complex conjugate, $\boldsymbol{\eta}_\ell \in \mathbb{C}^{M \times T_p}$ is the noise matrix whose elements are complex Gaussian random variables with zero mean and variance equal to $N_0\, B$ where N_0 is the noise power spectral density (The noise PSD (N_0) is equal to the Boltzmann Constant times the temperature, i.e., it is equivalent to approximately 4.11×10^{-21} watts per hertz at 25 degrees Celsius) and B the system bandwidth. Finally, $\mathbf{g}_{k,j,\ell} \in \mathbb{C}^{M \times 1}$ is the channel gain between the k-th user from cell j and the base station of cell ℓ, which represents the large-scale fading ($\beta_{k,j,\ell}$) and the small-scale fading ($\mathbf{h}_{k,j,\ell}$), i.e.,:

$$\mathbf{g}_{k,j,\ell} = \sqrt{\beta_{k,j,\ell}}\, \mathbf{h}_{k,j,\ell} \tag{2}$$

where $\mathbf{h}_{k,j,\ell} \in \mathbb{C}^{M \times 1}$ are independent and identically distributed complex Gaussian random variables with zero mean and unit variance. Meanwhile, $\beta_{k,j,\ell}$ is the path loss and shadowing effects. We assume a simplified path loss model, i.e.,:

$$\beta_{k,j,\ell} = \left(\frac{\lambda}{4\,\pi\, d_0}\right)^2 \left(\frac{d_0}{d_{k,j,\ell}}\right)^\gamma \mathcal{S}_{k,j,\ell} \tag{3}$$

where λ is the wavelength, d_0 is the reference distance (Typically, $d_0 \in [1, 10]$ meters for indoor environments and $d_0 \in (10, 100)$ meters for outdoors), $d_{k,j,\ell}$ is the distance (in meters) from the k-th user of cell j to the base station in cell ℓ. Finally, $\gamma \in [2, 8]$ is the path

loss exponent which is directly related to the scenario where the wireless communications take place, and $\mathcal{S}_{k,j,\ell}$ is the shadowing log-normal distributed random variable with zero mean and variance $10^{\frac{\sigma_s^2}{10}}$ where $\sigma_s^2 \in [4, 13]$ for outdoor channels.

We suppose the use of an orthogonal variable spreading factor (OVSF) code to generate the pilot sequences which are designated to each user through the pilot sequence allocation hipermatrix $\mathbf{\Phi} \in \{0,1\}^{K_\ell \times T_p \times L}$ whose elements are defined as:

$$\phi_{k,q,\ell} = \left\{ \begin{array}{ll} 0 & \text{pilot sequence is not allocated} \\ 1 & \text{pilot sequence is allocated} \end{array} \right\} \tag{4}$$

Moreover, according to [5], the uplink signal to interference plus noise ratio (SINR) of user k from cell ℓ observed by its base station when the number of antennas grows indefinitely is:

$$\delta_{k,\ell}^{\mathbb{B}} = \frac{\beta_{k,\ell,\ell}}{N_0\, B + \sum\limits_{\substack{j=1 \\ j \neq \ell}}^{K_j} \sum\limits_{k'=1}^{K_j} \boldsymbol{\phi}_{k,\ell} \boldsymbol{\phi}_{k',j}^T\, \beta_{k',j,\ell}} \tag{5}$$

where $\boldsymbol{\phi}_{k,\ell}$ is a row from the hipermatrix $\mathbf{\Phi}$, i.e., it is a $\{0,1\}^{1\times T_p}$ array that designates which pilot sequence is allocated to user k of cell ℓ. Whenever $\boldsymbol{\phi}_{k,\ell} \boldsymbol{\phi}_{k',j}^T = 1$, users k and k' from cell ℓ and j, respectively, use the same pilot sequence and, therefore, interfere in each others signal.

Alternative Representation

An alternative to the allocation variable binary representation is to use an integer representation, i.e., instead of using a binary pilot sequence allocation hipermatrix for the set of cells, one may use a single integer pilot sequence allocation $\mathbf{\Theta} \in \{1, \ldots, T_p\}^{L\times K}$ such that:

$$\theta_{\ell,k} = x \tag{6}$$

if the pilot sequence number $x \in \{0, \ldots, T_p\}$ is allocated to user k of cell ℓ.

In this representation, we introduce a new function which is responsible to verify if user k' from cell j is using the same pilot sequence as user k from cell ℓ. Therefore, consider $F(k, \ell, k', j) : \mathbb{Z}^{K\times L\times K\times L} \to \{0,1\}$ such that:

$$F(k,\ell,k',j) = \left\{ \begin{array}{ll} 0, & \text{if } \theta_{\ell,k} \neq \theta_{j,k'} \\ 1, & \text{otherwise} \end{array} \right. \tag{7}$$

Furthermore, we may rewrite Equation (5) using (7):

$$\delta_{k,\ell}^{\mathbb{Z}} = \frac{\beta_{k,\ell,\ell}}{N_0\, B + \sum\limits_{\substack{j=1 \\ j \neq \ell}}^{K_j} \sum\limits_{k'=1}^{K_j} F(k,\ell,k',j)\, \beta_{k',j,\ell}} \tag{8}$$

It is clear from Equations (5) and (8) that we may derive two equivalent optimization problems which are described in the next section.

3. Optimization Problems

The pilot sequence allocation optimization problem discussed here aims to maximize the total system spectral efficiency, simultaneously satisfying the constraints of pilot sequence orthogonality within the same cell as well as the fact that no users should be assigned different pilot sequences at the same time.

Moreover, it is important to differentiate the two mathematical models of allocation into two optimization problems: one optimization problem (OP1), as in Equation (9), uses

the binary allocation matrix representation found in Equation (5) and is mathematically defined as:

$$\underset{\mathbf{\Phi}}{\text{maximize}} \quad \mathcal{J}_1(\mathbf{\Phi}) = \sum_{\ell=1}^{L}\sum_{k=1}^{K_\ell} \log_2\left(1+\delta_{k,\ell}^{\mathbb{B}}\right) \tag{9}$$

$$\text{subject to} \quad \sum_{q=1}^{T_p} \phi_{k,q,\ell} \leq 1, \ \forall\, k \text{ and } \ \ell \tag{10}$$

$$\sum_{k=1}^{K_\ell} \phi_{k,q,\ell} = 1, \ \forall\, q \text{ and } \ \ell \tag{11}$$

$$\phi_{k,q,\ell} \in \{0,1\}, \ \forall\, k,\ q,\ \ell \tag{12}$$

where Equation (10) assures that no more than one pilot sequence is assigned to each user, while Equation (10) guarantees that each pilot sequence is used by only one user in each cell, and Equation (12) imposes that the decision variable is binary. Meanwhile, optimization problem 2 (OP2), as in Equation (13), uses the integer index representation introduced in (8) and may be described as:

$$\underset{\mathbf{\Theta}}{\text{maximize}} \quad \mathcal{J}_2(\mathbf{\Theta}) = \sum_{\ell=1}^{L}\sum_{k=1}^{K_\ell} \log_2\left(\delta_{k,\ell}^{\mathbb{Z}}\right) \tag{13}$$

$$\text{subject to} \quad \theta_{\ell,k} \neq \theta_{\ell,k'}, \ \forall\, \ell,\ k \text{ and } k' \tag{14}$$

$$\theta_{\ell,k} \in \{1,\ldots,T_p\}, \ \forall\, \ell \text{ and } k \tag{15}$$

where Equation (10) assures that the same pilot sequence is used more than one time within a cell, avoiding intra-cell interference, while Equation (11) assures that only the available pilot sequences are allocated to each user in the cell. Note that the constraints in OP2 are an integer adaptation of the same constraints in OP1.

To solve the optimization problem in (9), we use two different metaheuristic approaches. First, the binary version of the Particle Swarm Optimization (PSO) algorithm is considered. Later, a binary version of the Genetic Algorithm (GA) is also used to achieve a centralized pilot sequence allocation. Furthermore, we use the Smallest Position Value (SPV) technique along the PSO algorithm and Variable Neighbourhood Search (VNS) to solve the second optimization problem in (13). It is worth noting that although (9) and (13) have different domains, they represent the same physical problem and the conversion from one domain to another follows the rule presented by Equation (6).

4. Proposed Solutions

The optimization problems are solved using different methods: binary PSO (BPSO) and GA to solve (9), while PSO, SPV and VNS are used to solve (13). In this section we provide a detailed explanation of how these techniques are applied in this context.

4.1. Binary Particle Swarm Optimization

The BPSO was first described by Kennedy and Eberhart [23] and was an adaptation of the original (continuous) PSO. This algorithm works based on the behaviour of birds in search of foods, and the each candidate solution is presented as a bird in a flock flying through the search space (optimization problem domain), and the found represents the optimal solution. At each algorithm iteration, the velocity of each particle is updated through the equation:

$$\mathbf{v}_i[t+1] = \omega\, \mathbf{v}_i[t] + c_1\, r_1\, (\mathbf{x}_i[t] - \mathbf{p}_i) + c_2\, r_2\, (\mathbf{x}_i[t] - \mathbf{p}_g) \tag{16}$$

where ω is the inertia weight, $\mathbf{v}_i$ is the candidate i velocity, $\mathbf{x}_i$ is the particle position in the search space, $\mathbf{p}_i$ is the best position candidate i has ever been to in terms of fitness

function, and $\mathbf{p}_g$ is the best position overall the whole population, i.e., the first keeps an individual best position record and the latter, a global best position record. The coefficients c_1 and c_2 are the cognitive and social acceleration constants, respectively, whilst r_1 and r_2 are uniformly distributed random numbers in the interval $[0, 1]$.

In a classical PSO algorithm, the next step would be updating each candidate position using its velocity. However, to implement a binary version, the change in each candidate dimensions must also be either zero or one. Therefore, a sigmoid function is applied to each velocity component, and it is equivalent to the probability of changing each bit. In this paper, we use the following sigmoid function in each dimension $\mathbf{v}_i$:

$$\mathcal{S}(v_i[t]) = \frac{1}{1 + e^{-v_i[t]}} \tag{17}$$

The sigmoid function value is then compared to a random value generated to each dimension of the candidate $\mathbf{x}_i$ such that:

$$x_i[t+1] = \begin{cases} 1 & \text{if } r < \mathcal{S}(v_i[t]) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

where $r \sim \mathcal{U}(0, 1)$, i.e., r is a random variable with uniform distribution over the interval $(0, 1)$.

Although the position possibilities are constrained to $\{0, 1\}$, particle velocity may grow indefinitely. Hence, a maximum velocity constraint may be applied to each dimension, i.e.,:

$$v_i[t+1] = \begin{cases} v_{\max} & \text{if } v_i[t+1] > v_{\max} \\ -v_{\max} & \text{if } v_i[t+1] < -v_{\max} \end{cases} \tag{19}$$

Furthermore, since OP1 in (9) is a constrained optimization problem, we present two alternatives to treat the unfeasible solution candidates. The first is simply discarding the unfeasible candidates and replacing them with a random feasible one; the second is using a new fitness function defined as:

$$\tilde{\mathcal{J}}_1(\mathbf{\Phi}) = \begin{cases} \mathcal{J}_1(\mathbf{\Phi}) & \text{if feasible} \\ \mathcal{J}_{\min} - \mathcal{P} & \text{if unfeasible} \end{cases} \tag{20}$$

where $\mathcal{J}_{\min}$ is the worst feasible solution in the PSO population, and $\mathcal{P}$ is the sum of the constraints in Equations (10) and (11), i.e.,:

$$\begin{aligned} \mathcal{P} \quad = \quad & \sum_{\ell=1}^{L} \sum_{k=1}^{K_\ell} \left(\sum_{q=1}^{T_p} \phi_{k,q,\ell} - 1 \right) + \\ & \sum_{\ell=1}^{L} \sum_{q=1}^{T_p} \left(\sum_{k=1}^{K_\ell} \phi_{k,q,\ell} - 1 \right) \end{aligned} \tag{21}$$

Finally, a pseudocode of the BPSO algorithm is presented in Algorithm 1.

Algorithm 1: BPSO

input : Channel gain Hipermatrix—$\boldsymbol{\beta}$
Population Size—N
Max. Number of Iterations—I
output: Pilot Sequence Allocation Hipermatrix $\boldsymbol{\Phi}$
Create N random candidate solutions $\mathbf{x}_1[1], \ldots, \mathbf{x}_N[1]$;
Evaluate $\mathcal{J}(\mathbf{x}_1), \ldots, \mathcal{J}(\mathbf{x}_N)$ acoording to (9);
Start $\mathbf{p}_i$ for each $i = 1, \ldots, N$ and $\mathbf{p}_g = \arg\max_{\mathbf{p}_i} \mathcal{J}(\mathbf{p}_i)$;
for $i = 1$ ***until*** I **do**
  for $j = 1$ ***until*** N **do**
    Update $\mathbf{x}_j[i+1]$ using Equation (16) up to Equation (19);
    Evaluate the new solutions using Equations (20) and (21);
    if $\mathcal{J}(\mathbf{x}_j[i+1]) > \mathcal{J}(\mathbf{p}_j)$ **then**
      $\mathbf{p}_j = \mathbf{x}_j[i+1]$;
    end
    if $\mathcal{J}(\mathbf{x}_j[i+1]) > \mathcal{J}(\mathbf{p}_g)$ **then**
      $\mathbf{p}_g = \mathbf{x}_j[i+1]$;
    end
  end
end

4.2. PSO-Smallest Position Value

To solve the optimization problem (13), we used the Smallest Position Value method along with the PSO algorithm (PSO–SPV). In the SPV, the value of each problem dimension is exchanged by the index of the sorted values. In this specific problem, it is equivalent to the SPV value designating the pilot sequence allocated to each user. The sorting process which is responsible for swapping the real values for their ordinal ones is performed separately for each cell in the system. To illustrate the application of the SPV method, we present Table 1 which relates the value of each PSO particle (Real Value) to the pilot sequence assignment when four dimensions (users) are accounted for in a single cell.

Table 1. Example of operation of SPV in a single cell, $K = 4$ users scenario.

Dimension (User)	1	2	3	4
Real value	4.85	−2.15	145	−1.333
SPV (Pilot Sequence)	3	1	4	2

Using an SPV changes the problem domain from a binary scenario to integers values such that $\theta_{k,\ell} = x$ where k is the dimension of the SPV scheme and x is the SPV value, while ℓ designates the cell of interest.

The PSO is used along with the SPV through the application of Equation (16) and the update position equation:

$$\mathbf{x}_i[t+1] = \mathbf{x}_i[t] + \mathbf{v}_i[t+1] \tag{22}$$

It is important to point out that the velocity limits defined by Equation (19) are also applied to the PSO–SPV solution. Furthermore, from the nature of this approach, it is unlikely that two users will have the same pilot sequence assigned, or the other way around, since the sorting process would only place two users in the same position if their real values are exactly the same. To avoid this situation, in case two dimensions of an individual present the exact same real value, the tie break is the user index: the smallest user indexer is sorted as the smallest value. Due to these two facts, the PSO–SPV fitness function is the objective function presented in (9), which means that the BPSO and PSO–SPV

use different objective functions although the algorithms objective is the same: maximizing the system spectral efficiency.

4.3. Variable Neighbourhood Search

Another technique we also used to make the PSO–SPV approach even more robust is the Variable Neighbourhood Search metaheuristic which is based on the systemic change of neighborhoods of each possible solution in an attempt to find better solutions near the PSO candidate solutions.

VNS is applied to the PSO–SPV through an exchange of sequences allocated to user pairs on the same cell, i.e., users are randomly paired, and their pilot sequences are swapped whenever VNS is applied at each iteration of the PSO–SPV algorithm. Mathematically for each pair k' e k'' of users:

$$\begin{aligned} \theta_{k',\ell} = q' \quad &\rightarrow \quad \theta_{k',\ell} = q'' \\ \theta_{k'',\ell} = q'' \quad &\rightarrow \quad \theta_{k'',\ell} = q' \end{aligned} \tag{23}$$

where q' and q'' are the assigned pilot sequences. It is worth noting that when the number of the users is even, one of the users is left out of the VNS subroutine, and its pilot sequence remains unchanged.

To avoid a large increase in complexity, this mechanism is not performed for all individuals of the PSO–SPV population. In fact, we define r_{vns} as the probability of running VNS for each candidate solution in each one of the PSO–SPV iterations. The best balance between complexity and solution quality for the r_{vns} parameter is discussed in the results section.

Finally, the PSO–SPV and PSO–SPV–VNS algorithms are presented in the Algorithm 2.

Algorithm 2: PSO–SPV–VNS

input : Channel gain Hipermatrix—$\boldsymbol{\beta}$
Population Size—N
Max. Number of Iterations—I
output: Pilot Sequence Allocation Hipermatrix $\boldsymbol{\Theta}$
Create N random candidate solutions $\mathbf{x}_1[1], \ldots, \mathbf{x}_N[1]$ in the $\mathbb{R}^{K \times L}$ space;
Apply SPV to each $\mathbf{x}_i[1]$ generating the pilot sequence allocation array $\boldsymbol{\theta}_i[1]$;
Evaluate $\mathcal{J}(\boldsymbol{\theta}_i[1]), \ldots, \mathcal{J}(\boldsymbol{\theta}_N[1])$ according to (13);
Start $\mathbf{p}_i$ for each $i = 1, \ldots, N$ and $\mathbf{p}_g = \arg\max_{\mathbf{p}_i} \mathcal{J}(\mathbf{p}_i)$;
for $i = 1$ ***until*** I **do**
  for $j = 1$ ***until*** N **do**
    Update $\mathbf{x}_j[i+1]$ using Equations (16), (19) and (22);
    Apply SPV to each $\mathbf{x}_i[t+1]$ generating the pilot sequence allocation array $\boldsymbol{\theta}_i[i+1]$;
    if *rand()* $< r_{\text{vns}}$ **then**
      Randomly pair users and performe VNS through (23);
    end
    if $\mathcal{J}(\boldsymbol{\theta}_j[i+1]) > \mathcal{J}(\mathbf{p}_j)$ **then**
      $\mathbf{p}_j = \boldsymbol{\theta}_j[i+1]$;
    end
    if $\mathcal{J}(\boldsymbol{\theta}_j[i+1]) > \mathcal{J}(\mathbf{p}_g)$ **then**
      $\mathbf{p}_g = \boldsymbol{\theta}_j[i+1]$;
    end
  end
end

4.4. Genetic Algorithm

GA is an evolutionary computation method created by John H. Holland [24], wherein the algorithm simulates the theory of evolution by Charles Darwin.

The process consists of taking a population of individuals which are composed of chromosomes and applying different genetic operations to the individuals chromosomes, combining them while aiming to improve some fitness function. At each iteration of the algorithm, called a generation, individuals pair up to create descendants creating a new generation. During this process, the chromosomes, which can represent a position in the problem domain, may suffer mutation, while the individuals derived from their parents are created from a crossover operation. At each generation, only the best suitable individuals survive and transmit their genes to the next generation.

To further specify how the GA is applied to the optimization problem, we define the fitness function as the function $\mathcal{J}$ in (9). In addition, the GA will have three operators: a proportional fitness selection to define the parents for each new individual, the crossover operator and the the mutation operator.

The first step in each iteration is to select the parents for the next generation. Here we apply a fitness proportional selection, i.e., the probability of each individual i at some generation t to be selected as a parent is proportional to $\mathcal{J}(\mathbf{x}_i[t])$. Mathematically, the probability of i being selected as a parent is defined as:

$$\Pr(\mathbf{\Phi}_i) = \frac{\mathcal{J}(\mathbf{\Phi}_i)}{\sum_{j=1}^{N} \mathcal{J}(\mathbf{\Phi}_j)} \tag{24}$$

The crossover operator is responsible for combining the parent's chromosomes and generating the children's ones. To do this and simultaneously satisfy the constraints in Equations (10)–(12), we propose the use of a one-point crossover. To understand how the crossover operation is carried out, we first revisit the pilot sequence allocation hypermatrix which is a $K \times T_p \times L$ hypermatrix. Moreover, the crossover point is set at one of the cells and is defined as ℓ'. Hence, the first child is made of the individual allocation matrices from both parents: from cell 1 to ℓ', the matrices are from the first parent and from cell $\ell' + 1$ to L from the second parent. Its sibling is the mirror of that, i.e., the first parent passes the matrices from $\ell' + 1$ to L, while the second parent the matrices from 1 to ℓ. We define $\ell' = \lfloor L/2 \rfloor$.

The choice of this technique among the many others in the literature is involved in creating children to AG where they did not respect all the constraints of the problem domain. For this reason, the one-point crossover utilized granted that all children generated of crossing fathers are on the feasible solution for the problem.

To preserve the best solution throughout the generations if all the descendants at a given generation are worse than the best individual so far, we replace the worst individual of that generation with the best solution from the previous generation.

Finally, the mutation of an individual is carried out in a way that also satisfies the constraints in Equations (10)–(12). Without the loss of generality, let $K = T_p$ be even. The first step of the mutation operator is to generate a random permutation of the numbers from one to K for each cell ℓ. Let $\mathbf{p}_\ell = [p_1, \ldots, p_K]$ be that permutation, the first half of the permutation $\mathbf{p}_\ell$, i.e., from p_1 to $p_{\frac{K}{2}}$, is paired up element-wise with the second half, that is $p_{\frac{K}{2}+1}$ to p_K. Finally, we perform operations in the child hypermatrix such that for each cell's permutation the users defined by the number in each half permutation swap their pilot sequence. It is worth noting that mutation does not happen for every individual since there is a mutation rate $T_m \in (0, 1)$ which defines the probability of a mutation occurring in each member of a population.

These three operations happen at each iteration of the GA and are repeated until the maximum number of generations is achieved. The pseudocode for the adapted binary GA is presented in Algorithm 3.

Algorithm 3: Binary Genetic Algorithm

input : Max number of generations—$G_{\max}$, Mutation ratio—T_m
Channel gain hypermatrix—$\boldsymbol{\beta}$, Population size—N
output:
Hypermatrix of allocated pilot sequences—$\boldsymbol{\Phi}$
Create N candidate solutions $\mathbf{x}_1[1], \ldots, \mathbf{x}_N[1]$;
for $t = 1$ ***until*** $G_{\max}$ **do**
  for $n = 1$ ***until*** $N/2$ **do**
    Select two parents $\mathbf{x}_i[t]$ and $\mathbf{x}_j[t]$ using proportional selection;
    Perform the crossover operation generating $x_i[t+1]$ and $x_j[t+1]$;
    if $rand(0,1) > T_m$ **then**
      Perform the mutation operation;
    end
  end
  Evaluate $\mathcal{J}(\mathbf{x}_n[t+1])$ for all x;
  if $\nexists n$ *such that* $\mathcal{J}(\mathbf{x}_n[t+1]) > \max(\mathcal{J}(\mathbf{x}[t]))$ **then**
    Preserve the best solution from the generation t replacing the worst solution in $t+1$:;
    $\min_n(\mathcal{J}(\mathbf{x}_n[t+1])) = \max_n(\mathcal{J}(\mathbf{x}_n[t]))$;
  end
end

The literature has other candidate solutions for use, but this problem as described in Section 2 is a combinatorial optimization problem. In a general way, the methods utilized are mathematically complex and in binary or integer scenarios. Considering all the constraints introduced, the approach in the literature is not easy to adapt.

5. Simulations Results

We verify the applicability of the techniques presented in this manuscript to the pilot sequence assignment problem, as well as the impact of using them in terms of user SINR, system spectral efficiency and minimum and maximum user spectral efficiency. We developed a MATLAB script to create a dataset and made it public at github.com/evertonalex/utfpr-ppgi-pilotcontamination, accessed on 12 May 2022.

We consider a seven hexagonal cell cluster geographically disposed such as presented in Figure 1. Each user in the cell is randomly positioned. However, to establish some physical limits, user positions can only be integers numbers pairs and must be contained within the limits of the hexagon, i.e., let $(x, y) \in \mathbb{Z}^2$ be some user position related to its base station which is fixed at the origin, $x, y \sim \mathcal{U}\{1, R\}$, and the following conditions are also met:

1. $|x| \leq R$
2. $|y| \leq \frac{\sqrt{3}}{2} R$
3. $\frac{\sqrt{3}}{2} |x| + |y| \geq \frac{\sqrt{3}}{2} R$

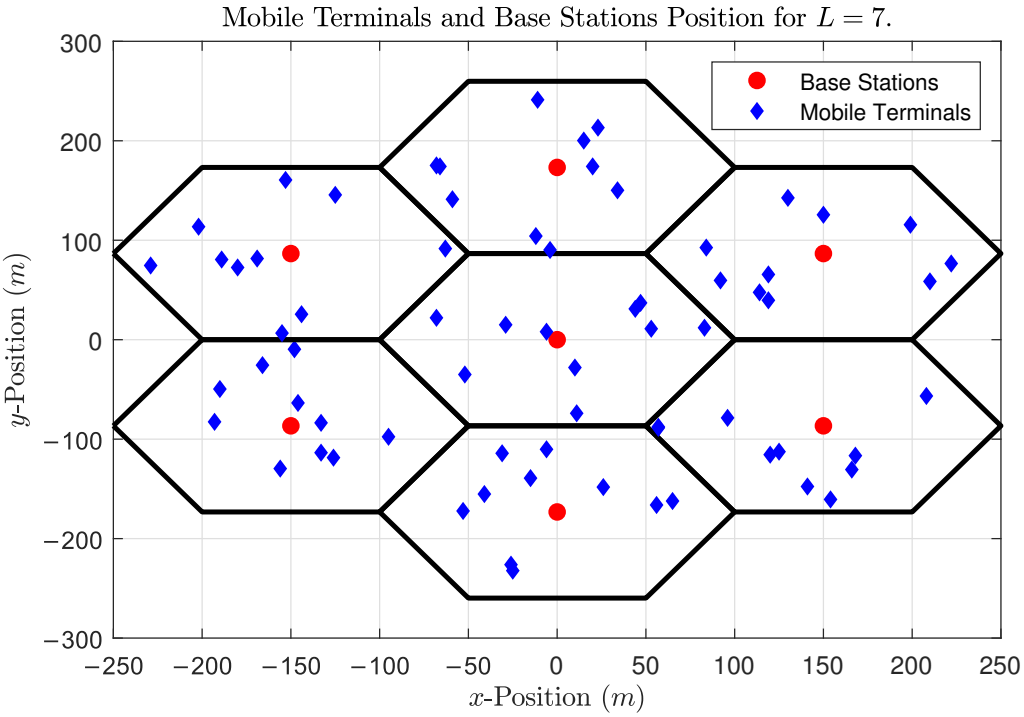

Figure 1. Base stations, users and cells geographic disposition in our simulation scenario. User positions were randomly generated.

After their positions are randomly generated with respect to the origin, each user position is translated to their cell position. Moreover, this process replaces any users that overlap the same position with a new random one. Table 2 shows the parameters used in the aforementioned scenario.

Table 2. System parameters used in the computational simulations.

Parameter	Adopted Value(s)
Number of Cells (L)	7
Number of Users per Cell (K)	$\{20, 40, 60\}$
Cell Radius (R)	$\{100, 250, 500, 1000\}$ (m)
Wavelength (λ)	8.56 (cm)
Reference Distance (d_0)	10 (m)
Path Loss Exponent (γ)	$\{6, 2\}$
Shadowing Variance (σ_s^2)	$\{6, 10\}$ (dB)
Channel Bandwidth (B)	20 (MHz)
Transmission frequency	3.5 GHz
Dataset instances	1000 per set of parameters

5.1. Heuristic Parameters Optimization

At this point, when the performance results are combined with the execution time results, such as observed in Figure 2 and 3, more iterations do not represent a more significant increase in performance. Hence, the best approach is to use as few iterations as possible in each of the algorithms due to the high cost and low increase in performance. All the simulations use the optimal parameters for each heuristic, and the number of iterations is set to 50.

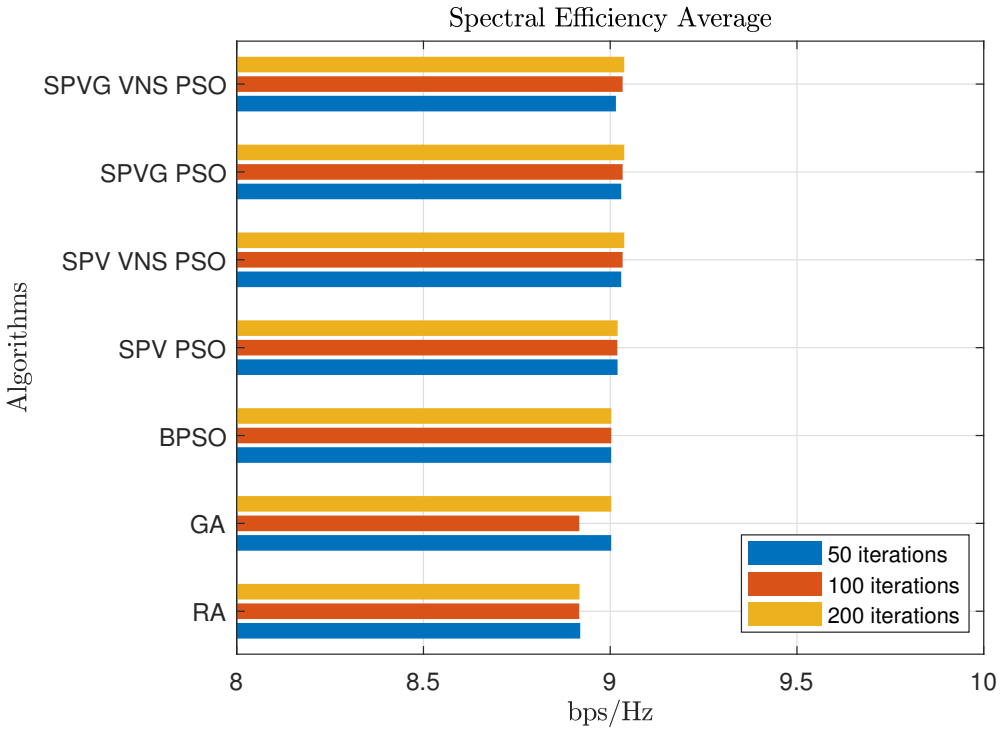

Figure 2. Spectral efficiency average by iterations.

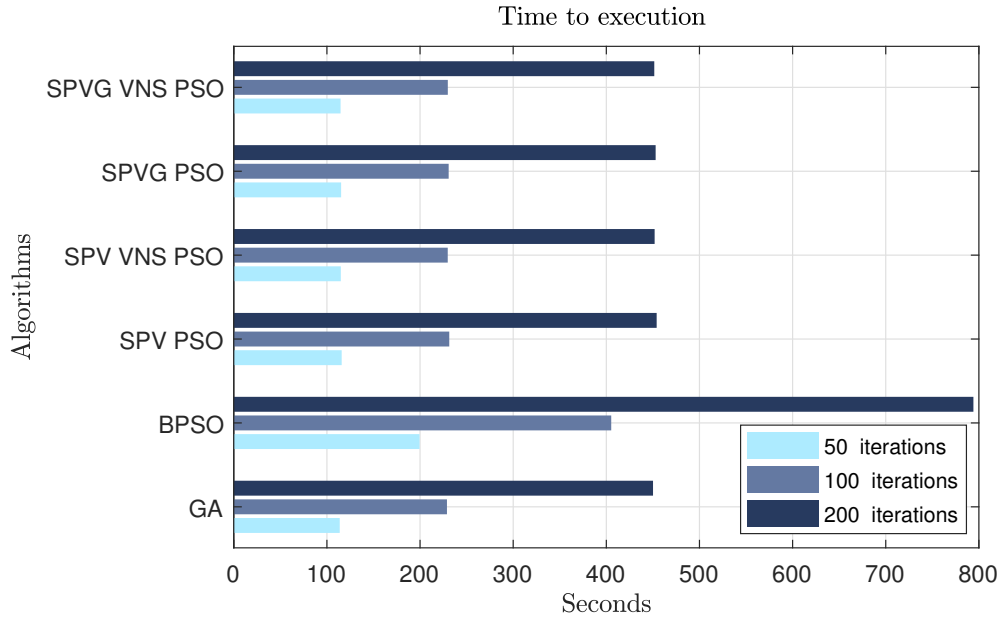

Figure 3. Time to complete the execution of a single instance in the dataset by each algorithm. We used a Google Cloud solution with 32 virtual CPUs, 32GB of RAM, running MATLAB on Ubuntu 20.

Since each heuristic has its own set of parameters which may impact the overall performance, we investigated the algorithms performance in the following scenario: $K = 20$ users and $R = 1000$ m. The findings, for each heuristic, are presented in Table 3. The values which result in the best performance are marked in bold type.

Table 3. Set of tested heuristic parameters. The parameters in bold are the ones which resulted in the best system spectral efficiency.

BPSO Parameters	**Tested Value(s)**
Inertia (ω)	$\{0.1, 0.25, \mathbf{0.5}\}$
Max. Velocity ($v_{\max}$)	$\{1, 2, \mathbf{3}\}$
Coefficient (c_1, c_2)	$\{(\mathbf{1}, \mathbf{1}), (2, 2), (3, 3)\}$
Max. # of Iterations (I)	$\{50, 100, \mathbf{200}\}$
Population Size (N)	$2 \times K$
SPV(G)-PSO Parameters	**Tested Value(s)**
Variable neighborhood search (VNS)	$\{0.15, 0.30, 0.60\}$
Max. # of Iterations (I)	$\{50, 100, \mathbf{200}\}$
Population Size (N)	$2 \times K$
GA Parameters	**Tested Value(s)**
Mutation Rate (T_m)	$\{0.25, 0.50, \mathbf{0.75}\}$
Max. # of Generations ($G_{\max}$)	$\{50, 100, \mathbf{200}\}$
Population Size (N)	$2 \times K$

It is worth noting that although we tested different parameters for each heuristic, the results in terms of system spectral efficiency are where all are within a less than 1% margin from each other. For instance, the increment in the results for GA when $G_{\max}$ goes from 100 to 200 is around 0.0019%. Additionally, the increment from using a 0.5 mutation rate to a 0.75 is around 0.0067%. In both cases, the difference means less than a bit per second per Hertz, which mean they are irrelevant system-wise.

This pattern is also observed in the BPSO, SPV–PSO and SPVG–PSO (with or without VNS) results. The difference between the parameters is within the margin of error for the dataset size used in our work.

Since the difference between the algorithms performance in terms of average SE is marginal, we further analyzed the minimal spectral efficiency case. The argument here is that increasing the minimal SE may result in a reduction in terms of outage probability.

Thus, Table 4 presents the minimal SE for different combinations of maximum number of iterations and for each algorithm. It is worth noting that only the optimal heuristic parameters were used in this simulation. Furthermore, it is clear that the tested heuristics are divided into two groups when we compare their performance: the low performing group includes RA, BPSO and GA while the high performing group is composed of SPV–PSO, SPV–VNS–PSO, SPVG–PSO and SPVG–VNS–PSO.

Table 4. Minimal spectral efficiency (in bps/Hz) using the optimal parameters for each heuristic ($\omega = 0.25$, $v_{\max} = 2$, $VNS = 0.30$, $T_m = 0.50$ and $K = 20$).

Algorithm	**Max. # of Iterations (I)**		
	50	**100**	**200**
RA	7.56×10^{-5}	7.56×10^{-5}	7.56×10^{-5}
BPSO	7.56×10^{-5}	7.56×10^{-5}	7.56×10^{-5}
GA	7.56×10^{-5}	7.56×10^{-5}	7.56×10^{-5}
SPV PSO	1.4143	1.4329	1.4246
SPV VNS PSO	1.3762	1.3364	1.3239
SPVG PSO	1.3897	1.3794	1.3956
SPVG VNS PSO	1.3191	1.3419	1.2982

Besides the aforementioned fact, one can also imply that the average variation in performance among all algorithms in the high performance bracket when the number of iterations is increased from 50 to 100 is less than 2%. Additionally, even though Table 4 shows that RA, BPSO and GA had the same performance despite the change in the number

of iterations, the average difference in performance was in the order of 0.0013% when the number of iterations rise from 50 to 100.

Finally, as it is commonplace, increasing the number of iterations directly impacts the computational time of all the algorithms but RA. Hence, we present Figure 3 which shows the average time needed to compute the solution.

At this point, it is clear when the performance results are combined with the execution time results that the best approach is to use as few iterations as possible in each of the algorithms due to the high cost and low increase in performance. From now on, all the simulations use the optimal parameters for each heuristic, and the number of iterations is set to 50.

As one may observe, each algorithm has a set of parameters that need to be adapted to the discussed problem. In order to find the best parameters, we considered a macro-cell scenario scenario with $K = 20$ users per cell of $R = 1000$ m in radius and a maximum number of iterations $I = 50$ to all heuristics.

For the PSO-based algorithms, we first analyzed the inertia coefficient ω. Figure 4 shows the results which demonstrate that $\omega = 0.25$ is the best configuration for both BPSO and SPVG–PSO, while $\omega = 0.1$ is the best option for SPV–PSO. Even though the performance differences of the tested parameters values are marginal, the best values were used to evaluate other aspects of the proposed solutions.

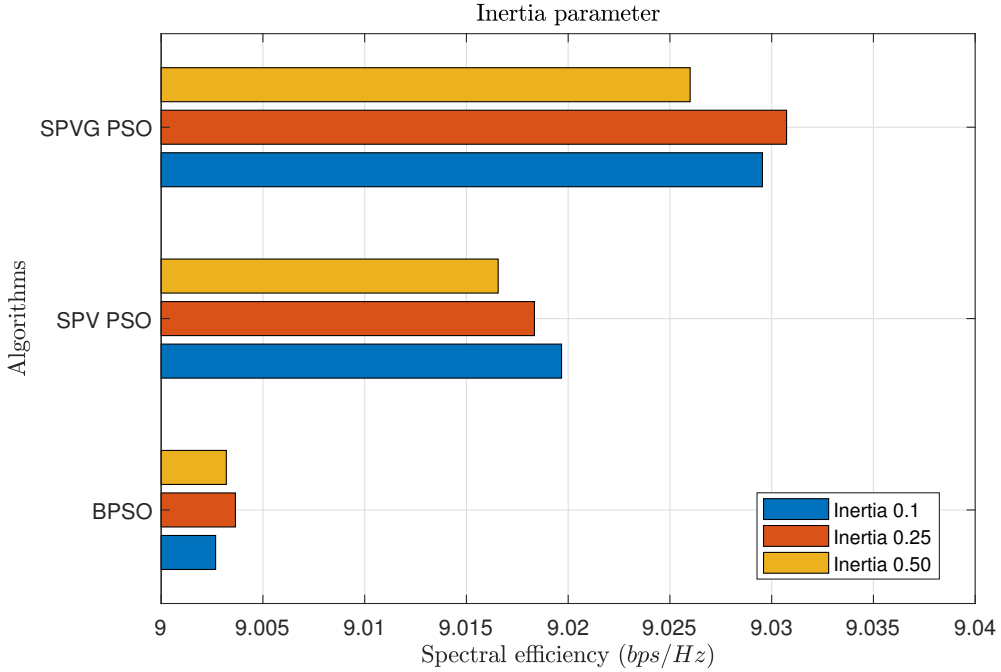

Figure 4. Results of tests to find the better parameter to inertia ($K = 20$, $R = 1000$ m and $v_{\max} = 1$).

The second parameter tested in the PSO-based algorithms was the maximum velocity $(v_{\max})$. Values from one to four were used, and the best results were different for each algorithm: $v_{\max} = 4$ for the BPSO, $v_{\max} = 3$ to SPV–PSO and $v_{\max} = 2$ to SPVG–PSO. The tests results are shown in Figure 5. As it was observed with the inertia coefficient, the increment in system performance were less than 0.1% when compared to the random allocation process.

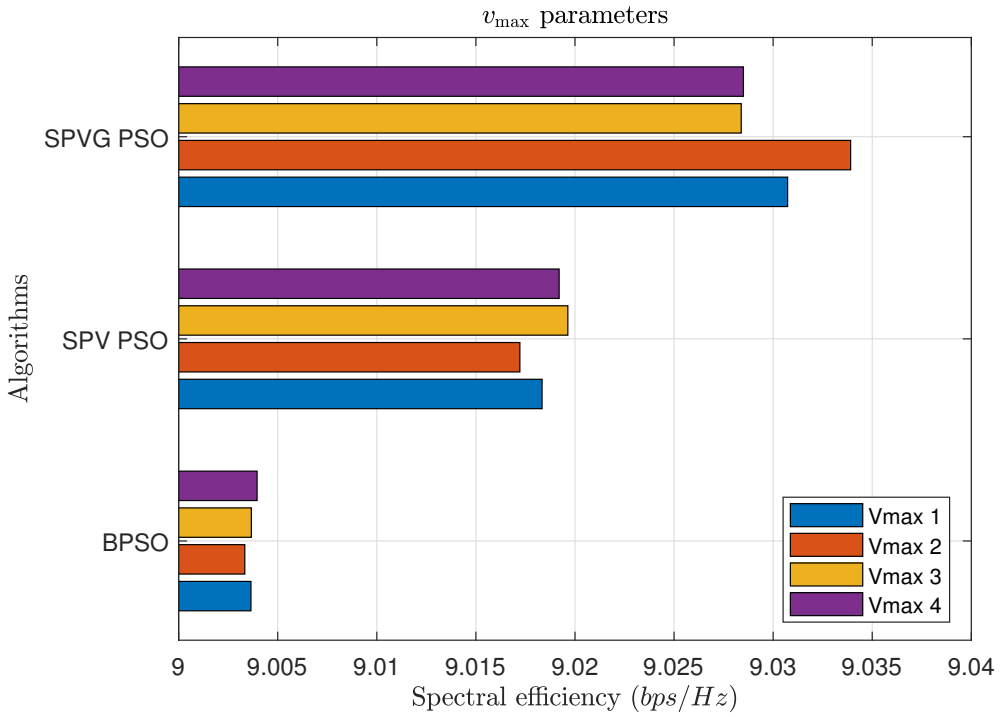

Figure 5. Result of tests to find the better parameter to $v_{\max}$ ($K = 20$, $R = 1000$ m and $(\omega) = 0.25$).

The third and forth parameters tested were the local and global solution weights (c_1 and c_2). These parameters are not found in the SPVGs variants, so they were only tested for BPSO and SPV–PSO.

All the previous analysis considered that $c_1 = c_2 = 1$, so we compared the results with those solutions. Increasing the local solution weight $c_1 = 2$ and maintaining the global weight $c_2 = 1$ downgrades the BPSO performance in $0,001\%$, while increasing the SPV–PSO performance in 0.02%. Further increasing the local solution $c_1 = 5$ and $c_2 = 1$ made the BPSO algorithm's performance around 0.004% higher than the baseline ($c_1 = c_2 = 1$), while an increase of 0.03% was found for the SPV–PSO when compared to the baseline. Finally, changing the parameters to $c_1 = 1$ and $c_2 = 5$ increased the BPSO performance in 0.005% while decreased the SPV–PSO performance in 0.02%

This test showed that c_1 and c_2 values are not significant in terms of performance since their impact are in the one thousandth of a bit/s/Hz mark.

The last PSO-based parameters optimization was the VNS rate. We tested four different configurations, and the results are presented in Figure 6. They demonstrate that a rate of 0.9 is the best configuration for both SPV–PSO–VNS and SPVG–PSO–VNS.

Finally, the GA algorithm parameter was tested. Figure 7 shows the GA's performance for different mutation rates, and the $T_m = 0.9$ was found to be the best parameter value.

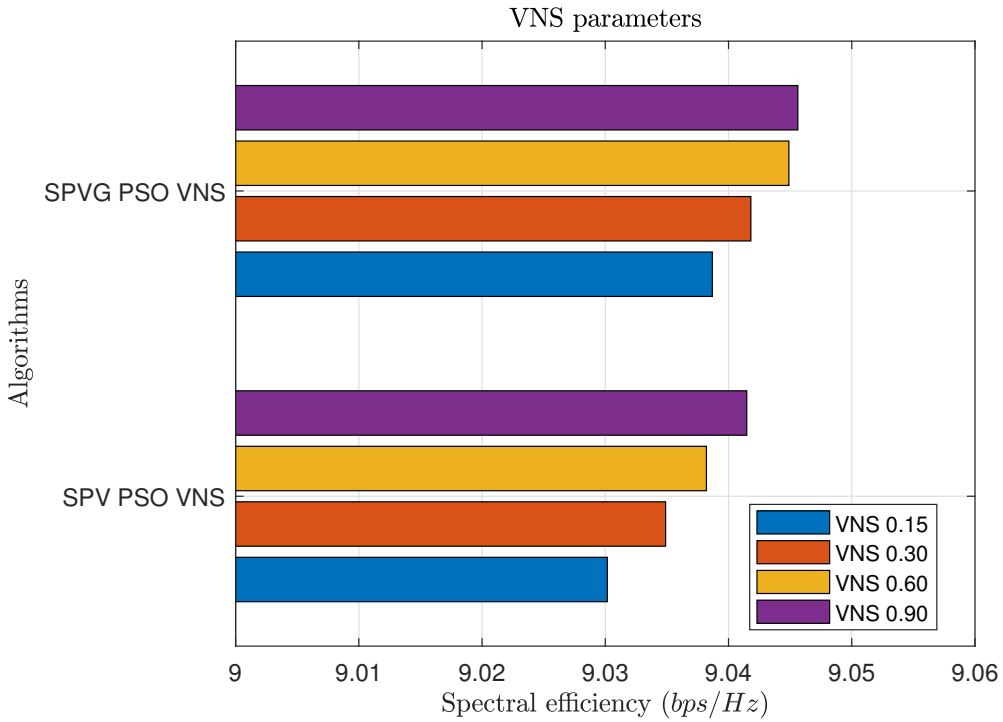

Figure 6. Results of tests to find the better parameters to VNS ($K = 20$, $R = 1000$ m, $(\omega) = 0.25$ and $v_{\max} = 1$).

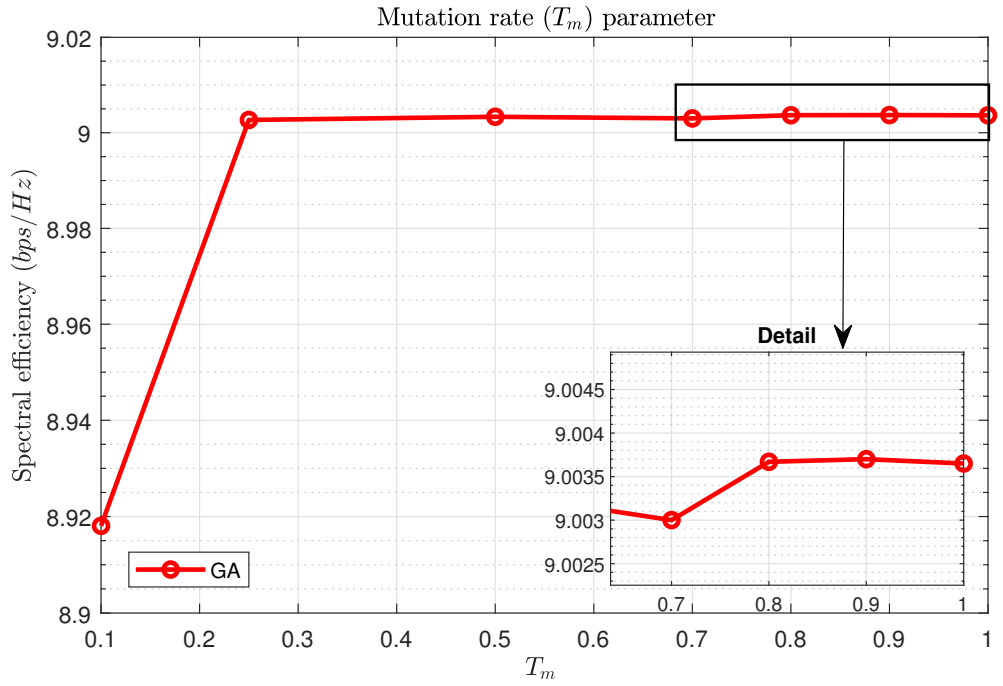

Figure 7. Results of tests to find the better parameters to T_m ($K = 20$, $R = 1000$ m).

5.2. Cell Size Impact on Performance

We conducted an analysis of the impact of cell size on all of the algorithms performance. We set up a scenario with $K = 20$ users per cell and the maximum number of iterations per algorithm to 50. To evaluate performance, Figure 8 shows the cumulative distribution function using our dataset.

As one may see, the impact of cell size on the CDF is really marginal, which may indicate that the system is not interference bounded. This is corroborated with two facts.

No intracell interference exists due to the orthogonality of the pilot sequences within a cell. The path-loss and shadowing effects deteriorate the interfering signals from other cells.

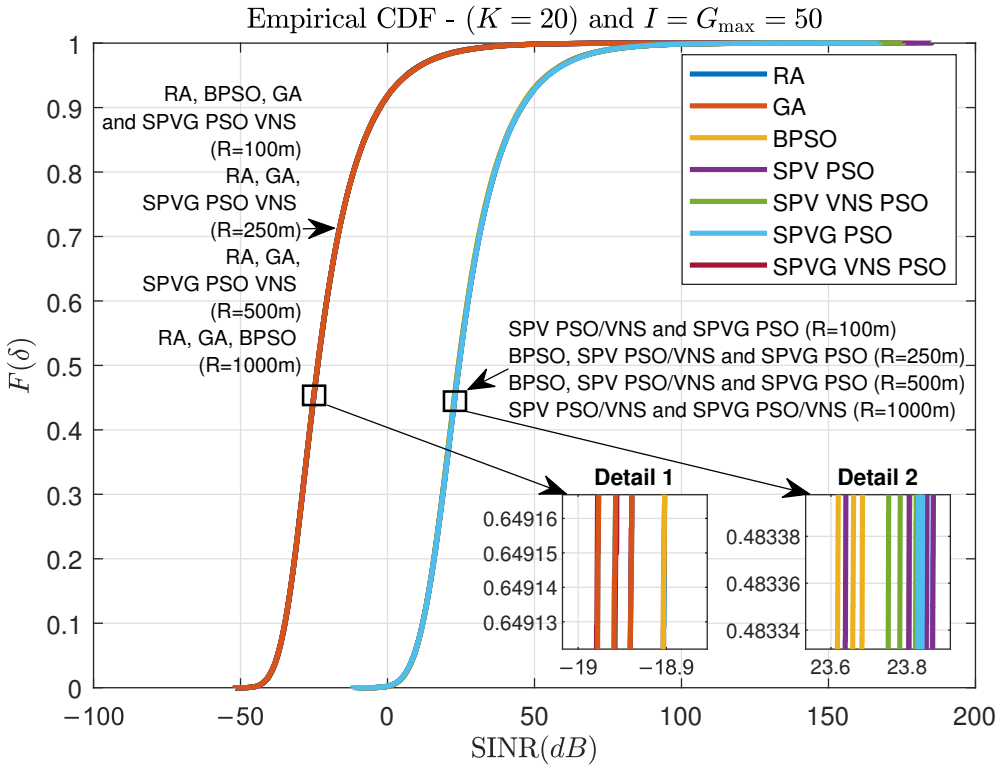

Figure 8. Empirical CDF for different cell sizes with $K = 20$ users, $G_{\max} = 50$ generations and $I = 50$ iterations.

Although the optimization problem aims to maximize the system spectral efficiency, we analysed the minimum spectral efficiency among all users in the system which may be seen as a fairness measure in the system. Table 5 presents the numerical values of the average lowest spectral efficiency in the system considering the 1000 instances in the dataset. The difference between the results for the RA, BPSO and GA algorithms were marginal, so they are listed in the same column.

An important remark to make here is: for a cell with a 100 m radius, the SPV–PSO has an average lowest spectral efficiency 1.7% higher than the RA, BPSO and GA algorithms.

Table 5. Minimum spectral efficiency per user in different cell sizes. The best value for each algorithm is marked in bold type.

Parameters	Minimum Spectral Efficiency per User (in bps/Hz)				
	BPSO/GA	SPV–PSO		SPVG–PSO	
		w/o VNS	with VNS	w/o VNS	with VNS
$K = 20, R = 1$ Km	7.56×10^{-5}	1.4143	1.3762	1.3897	1.3191
$K = 20, R = 500$ m	7.85×10^{-5}	1.4233	**1.402**	**1.4093**	1.3561
$K = 20, R = 250$ m	7.90×10^{-5}	1.4142	1.3858	1.3782	**1.3873**
$K = 20, R = 100$ m	$\mathbf{8.03} \times 10^{-5}$	**1.4414**	1.3605	1.3986	1.3424

In relation to the average maximum spectral efficiency, the results are presented in Table 6. It is clear that RA, BPSO and GA algorithms have a worse performance than the other ones. In fact, the average maximum spectral efficiency more than doubles when the same cell radius is compared.

Finally, in terms of cell size, it is clear that for all the results presented here, the impact in terms of CDF, average maximum and minimum spectral efficiency is marginal which corroborates the argument that the system is not interference bounded.

Table 6. Maximum spectral efficiency per user in different cell sizes. The best value for each algorithm is are marked in bold type.

Parameters	Maximum Spectral Efficiency per User (in bps/Hz)				
	BPSO/GA	SPV–PSO		SPVG–PSO	
		w/o VNS	with VNS	w/o VNS	with VNS
$K = 20, R = 1$ Km	13.4015	29.3362	29.3362	29.3378	**29.3515**
$K = 20, R = 500$ m	12.9367	28.8354	28.9150	28.8825	**28.9445**
$K = 20, R = 250$ m	12.7704	28.7173	**28.8393**	28.7896	28.7380
$K = 20, R = 100$ m	13.1714	29.0950	29.0974	29.1499	**29.2260**

5.3. System Loading Impact on Performance

Figure 9 describes the empirical CDFs for different system loading scenarios: $K = 20$, $K = 40$ and $K = 60$. These values represent three system loading scenarios which represent low, medium and high user density. The main objective here is to determine whether the higher density of users in the system implies more interference among them.

It is evident that RA, BPSOand GA suffer an impact in performance as the system loading increases for the same cell size. However, that significant difference in performance is not observed for the other algorithms, meaning that they can handle user density better than the former three algorithms. Numerically, there is an 11 dB CDF shift from $K = 20$ users to $K = 40$ and a 7 dB shift from $K = 40$ to $K = 60$. Meanwhile, the performance difference between $K = 20$ and $K = 60$ is less than a 0.1 dB shift.

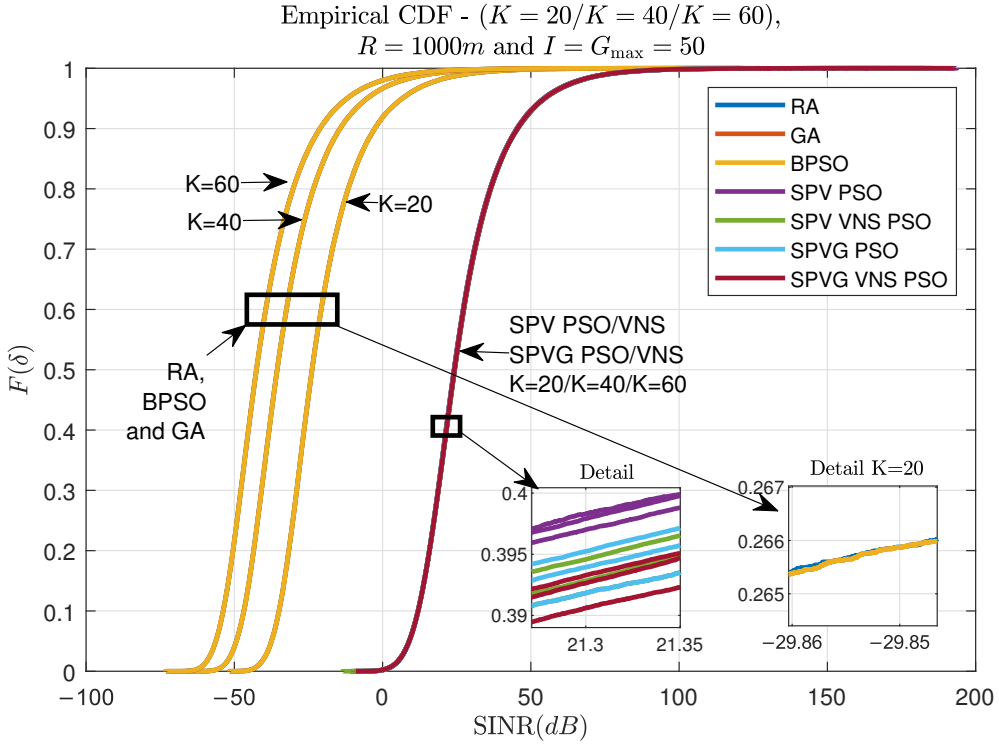

Figure 9. Empirical CDF for $K = 20$, $K = 40$ and $K = 60$ users in an R = 1000 m cell size scenario with the maximum number for iterations set to 50 for all algorithms.

6. Conclusions

This work presented six different heuristics to solve the pilot sequence allocation problem in Massive MIMO systems. We optimized the heuristics parameters to achieve the best performance in the discussed problem while still considering the computational time constraints inherent to the problem's nature.

We analyzed the impact of different scenario configurations on the performances of the algorithms. The cell size impact analysis offered clear conclusions that the system is not interference bounded and that the performance of SPV–PSO and all its derivations have far better performance in terms of system spectral efficiency, as well as average maximum and minimum spectral efficiency when compared to the RA, BPSO and GA solutions.

We further investigated the impact of system loading and found that RA, BPSO and GA solutions are directly affected by user density, while the other algorithms presented a more robust performance. We found that average minimum spectral efficiency may be increased in up to 1.7 million % when compared from RA, BPSO and GA solutions to a SPV–PSO-based one during the described investigations.

Finally, it is important to point out that SPV–PSO with $\omega = 0.1$, $v_{\max} = 3$, $c_1 = 5$ and $c = 1$ achieved the best results for the R = 1000 m and $K = 20$ users scenario.

7. Future Works

Future works include testing and comparing more different approaches to solve this problem such as: micro GAs, Differential Evolution, such as in [25], other versions of the PSO, as in [26,27], greed search algorithms, supervised learning with the Convolutional Neural network (CNN) and Multilayer Perceptron (MLP) as well as further investigating at which cell size or path-loss configurations the system becomes interference bounded.

Author Contributions: Conceptualization, L.D.H.S.; Investigation, E.A.M.; Methodology, D.S.S.; Project administration, L.D.H.S.; Software, E.A.M.; Supervision, D.S.S. and R.S.P.; Validation, R.P.B. and L.D.H.S.; Writing—original draft, E.A.M.; Writing—review & editing, R.P.B., R.S.P. and L.D.H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: github.com/evertonalex/utfpr-ppgi-pilotcontamination (accessed on 16 May 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cisco. Cisco Visual Networking Index: Forecast and Trends, 2018 2023 White Paper. Avaliable online: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html (accessed on 11 November 2021).
2. United Nations, Department of Economic and Social Affairs. World Population Prospects 2019. Available online: https://population.un.org/wpp/ (accessed on 11 November 2021).
3. Ericsson. Ericsson Mobility Report, Ericsson Mobility Report. Available online: https://www.ericsson.com/assets/local/reports-papers/mobility-report/documents/2019/emr-november-2019.pdf (accessed on 11 November 2021)
4. Pand, S.; Fitzek, F.H.P.; Lehmann, C.; Nophut, D.; Kiss, D.; Kovács, V.; Ákos, N.; Csovási, G.; Tóth, M.; Rajacsics, T. Joint Design of Communication and Control for Connected Cars in 5G Communication Systems. In Proceedings of the 2016 IEEE Globecom Workshops (GC Wkshps), Washington, DC, USA, 2016; pp. 1–7. [CrossRef]
5. Marzetta, T.L. Noncooperative cellular wireless with unlimited numbers of base station antennas. *IEEE Trans. Wirel. Commun.* **2010**, *9*, 3590–3600. [CrossRef]
6. Jin, S.; Li, M.; Huang, Y.; Du, Y.; Gao, X. Pilot scheduling schemes for multi-cell massive multiple-input–multiple-output transmission. *IET Commun.* **2015**, *9*, 689–700. [CrossRef]
7. Zhu, X.; Wang, Z.; Dai, L.; Qian, C. Smart Pilot Assignment for Massive MIMO. *IEEE Commun. Lett.* **2015**, *19*, 1644–1647. [CrossRef]

8. Ku, L.; Fan, J.; Deng, J. Low complexity pilot allocation in massive MIMO systems. In Proceedings of the 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN), Beijing, China, 4–6 June 2016. [CrossRef]
9. Teeti, M.A.; Wang, R.; Liu, Y.; Ni, Q. Pilot optimization in multicell massive MIMO. In Proceedings of the 2016 IEEE International Conference on Communication Systems (ICCS), Shenzhen, China, 14–16 December 2016. [CrossRef]
10. Zhou, Z.; Wang, D. Pilot scheduling based on water-filling algorithm in massive MIMO. In Proceedings of the 6th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 17–19 June 2016; pp. 89–92. [CrossRef]
11. Zhang, N.; Bai, Z.Q.; Zhang, B.; Su, Y.; Han, T.; Kwak, K. Genetic algorithm based pilot allocation scheme for massive MIMO system. In Proceedings of the 2016 IEEE International Conference on Communication Systems (ICCS), Shenzhen, China, 14–16 December 2016. [CrossRef]
12. Niu, G.; Mu, X.; Guo, X.; Zhang, J. Pilot Pollution Algorithm Based on Cell Sectorizatoin Elimination in Massive MIMO System. In Proceedings of the 7th IET International Conference on Wireless, Mobile & Multimedia Networks (ICWMMN 2017), Beijing, China, 17–20 November 2017. [CrossRef]
13. Makram Alkhaled, E.A.; Hamdi, K.A. Adaptive Pilot Allocation Algorithm for Pilot Contamination Mitigation in TDD Massive MIMO Systems. In Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC), San Francisco, CA, USA, 19–22 March 2017. [CrossRef]
14. Kim, K.; Lee, J.; Choi, J. Deep Learning Based Pilot Allocation Scheme (DL-PAS) for 5G Massive MIMO System. *IEEE Commun. Lett.* **2018**, *22*, 828–831. [CrossRef]
15. Gao, H.; Zhang, T.; Feng, C.; Wang, Y. Clustering Based Pilot Allocation Algorithm for Mitigating Pilot Contamination in Massive MIMO Systems. In Proceedings of the 2018 Asia-Pacific Microwave Conference (APMC), Kyoto, Japan, 6–9 November 2018; pp. 878–880. [CrossRef]
16. Khan, A.; Irfan, M.; Ullah, Y.; Ahmad, S.; Ullah, S.; Hayat, B. Pilot Contamination Mitigation for High and Low Interference Users in Multi-Cell Massive MIMO Systems. In Proceedings of the 2019 15th International Conference on Emerging Technologies (ICET), Peshawar, Pakistan, 2–3 December 2019; pp. 1–5. [CrossRef]
17. Luo, S.; Zhang, C.; Duan, Y.; Chen, J. Pilot Allocation Game: A Monte Carlo Tree Based Method. In Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 16–18 October 2019; pp. 277–282. [CrossRef]
18. Nie, X.; Zhao, F. Joint Pilot Allocation and Pilot Sequence Optimization in Massive MIMO Systems. *IEEE Access* **2020**, *8*, 60637–60644. [CrossRef]
19. Zhu, X.; Dai, L.; Wang, Z.; Wang, X. Weighted-Graph-Coloring-Based Pilot Decontamination for Multicell Massive MIMO Systems. *IEEE Trans. Veh. Technol.* **2017**, *66*, 2829–2834. [CrossRef]
20. Xu, S.; Zhang, H.; Tian, J.; Wu, D.; Yuan, D. Pilot Length Optimization for Spectral and Energy Efficient D2D Communications Underlay Massive MIMO Networks. In Proceedings of the 2018 International Conference on Computing, Networking and Communications (ICNC), Maui, HI, USA, 5–8 March 2018; pp. 855–860.
21. Dey, A.; Pattanayak, P.; Gurjar, D.S. Arithmetic/Geometric Progression Based Pilot Allocation With Antenna Scheduling for Massive MIMO Cellular Systems. *IEEE Netw. Lett.* **2021**, *3*, 1–4. [CrossRef]
22. Parida, P.; Dhillon, H.S. Pilot Assignment Schemes for Cell-Free Massive MIMO Systems. *arXiv* **2021**, arXiv:2105.09505.
23. Kennedy, J.; Eberhart, R. A discrete binary version of the particle swarm algorithm. In Proceedings of the 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, Orlando, FL, USA, 12–15 October 1997; Volume 5, pp. 4104–4108. [CrossRef]
24. Holland, J.H. Genetic Algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
25. Santucci, V.; Baioletti, M.; Milani, A. An algebraic framework for swarm and evolutionary algorithms in combinatorial optimization. *Swarm Evol. Comput.* **2020**, *55*, 100673. [CrossRef]
26. Chen, W.N.; Zhang, J.; Chung, H.S.H.; Zhong, W.L.; Wu, W.G.; Shi, Y.h. A Novel Set-Based Particle Swarm Optimization Method for Discrete Optimization Problems. *IEEE Trans. Evol. Comput.* **2010**, *14*, 278–300. [CrossRef]
27. Moraglio, A.; Chio, C.; Togelius, J.; Poli, R. Geometric Particle Swarm Optimization. *J. Artif. Evol. Appl.* **2008**, *2008*, 143624. [CrossRef]