ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Efficient Dynamic Channel Allocation in Multibeam Satellite Communication Systems

Datong Xie[1,2]

[1] School of Information Engineering, Fujian Business University, Fuzhou, 350012, China

[2] Big Data Business Intelligence Research Center of Fujian Province University, Fuzhou, 350012, China

Abstract

In the context of multibeam satellite communication systems, the dearth of channel resources presents a formidable hurdle. The dynamic distribution of these constrained resources to meet the diverse and unequal requirements of users has become a focal point of research. This study introduces a mathematical model to address the practical problem of dynamic channel allocation, and suggests three algorithms for solving this problem. It is shown that the fundamental problem can be effectively translated into multiple graph-coloring problems within polynomial time. Evolutionary algorithms are employed to optimize the allocation procedure by minimizing conflicts and blocking rates, whereas deterministic algorithms are utilized to decrease the intervals. Moreover, a hybrid algorithm that amalgamates the virtues of both methods is suggested, with the objective of achieving an exceptional equilibrium among various performance metrics. An in-depth evaluation of the three algorithms was performed, with emphasis on conflict, interval, computation time, and blocking rate. Comprehensive experiments on 25 meticulously designed datasets reveal that both evolutionary and deterministic algorithms display unique benefits in terms of conflict resolution, blockage minimization, and computational efficiency. Nevertheless, the hybrid algorithm, leveraging the best characteristics of both, emerges as a superior alternative, attaining an optimal equilibrium across all three performance indicators.

Keywords

Dynamic Channel Allocation; Deterministic Algorithm; Evolutionary Algorithm; Hybrid Algorithm; Multibeam Satellite Communication.

1. Introduction

A multibeam satellite communication system concurrently generates multiple spot beams to cover targeted terrestrial regions, enhancing spectrum utilization through bandwidth subdivision and reuse across spatially distinct beams [1,2]. This architecture, compared to single-beam systems, significantly expands coverage and optimizes resource distribution. However, its on-board resource constraints and the uneven distribution of terminals pose challenges, particularly in balancing load between beams, where overloaded beams risk high call blocking rates, while underutilized beams suffer from resource inefficiency.

Efficient resource management and allocation are paramount to mitigating these issues. Unreasonable channel allocation can lead to suboptimal utilization or excessive call blocking. Given limited resources, pre-allocating sufficient channels to each cell is impractical, necessitating channel reuse strategies while navigating constraints like reuse distance, channel spacing, and interference [3,4].

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Inter-beam channel allocation [5], addressing the non-uniform traffic distribution, is the focus of this study. Two key strategies exist: Fixed Channel Allocation (FCA) and Dynamic Channel Allocation (DCA) [6]. FCA statically assigns bandwidth to cells, simplifying management but resulting in resource wastage when channels remain unutilized. Conversely, DCA dynamically pools and allocates channels, offering greater flexibility and utilization, with classical and intelligent algorithms driving its implementation.

Previous works have explored advancements in DCA. Umehira et al.[7] optimized an interference-aware algorithm with service quality thresholds. Li et al. introduced beam cooperation to serve edge users dynamically. Chang et al.[8] combined random and demand-assigned access methods to improve throughput and delay control. Liu et al. [9] designed a comprehensive beam-centric resource allocation and scheduling scheme, prioritizing channel capacity fairness and user satisfaction over fixed beam allocation. This study aims to further advance these efforts by investigating innovative approaches to inter-beam channel allocation in multibeam satellite systems. Traditional algorithms for dynamic satellite channel allocation, though stable, suffer from heavy computational loads and inefficiency, hindering real-time applicability. Researchers have explored intelligent alternatives like simulated annealing, evolutionary algorithms, deep learning, and reinforcement learning to tackle this challenge. Various studies have proposed innovative approaches: Koustaftiki et al.[10] employed simulated annealing, Bisio et al. [11] framed it as a multi-objective optimization, Wang et al. [12] utilized decomposition-based evolutionary algorithms, while He et al. [13] designed a multi-objective planning scheme for multibeam satellites. Zhou et al. [14] addressed traffic variability and interference with binary integer programming and an enhanced artificial bee colony algorithm. Huang et al. [15] introduced a Hybrid-Action Deep Q-Network, and Hu et al. [16] utilized deep reinforcement learning to reduce blocking rates. Deng et al. [17] and Xu et al. [18] further advanced these techniques with twin delayed deep deterministic policy gradient and proximal policy optimization, respectively. Kashyap et al. [19] innovated by applying unsupervised and supervised learning for resource allocation.

Despite their focus on efficiency and utilization, these approaches have their strengths and limitations, showcasing the adaptability and self-learning capabilities of intelligent algorithms in mitigating human factors' influence. Dynamic channel allocation in satellites poses unique challenges due to its global nature, contrasting with local optimizations prevalent in mobile communication. Traditional methods struggle with complexity and suboptimal solutions. Greedy algorithms excel in speed but risk local optima, while neural networks' adaptability is hindered by memory dependence. Evolutionary algorithms, though less efficient in individual optimization, boast robust global search and versatility. Thus, we explore evolutionary algorithms for dynamic channel allocation.

The remainder of this paper is organized as follows. In Section 2, a mathematical model is extracted for an actual channel allocation problem in a multibeam satellite communication system, and it is proven that the original problem can be converted into several graph coloring problems in polynomial time. In Section 3, evolutionary, deterministic, and hybrid algorithms are presented. In Section 4, the effectiveness of these algorithms is verified experimentally. Finally, conclusions are presented in the last section.

2. Problem Description and Analysis

There were 36 cells, each using one beam for communication (Figure 1). Assuming that the physical bandwidth of the system is 200 frequency points, the simplest solution is to divide the 200 frequency points equally into four parts with each cell having 50 consecutive frequency points. However, in the actual operation of the system, the upstream bandwidth requirement of each cell changes over time, with some cells requiring more than 50 frequency points and others requiring fewer than 50 frequency points. Therefore, it is necessary to dynamically allocate the upstream bandwidth (i.e., the number of upstream frequency points of each cell) to fully utilize the system resources, improve resource utilization, and satisfy the requirements of each cell as much as possible.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

The maximum requirement for a beam was set to twice the reserved bandwidth (100 frequency points), and the minimum requirement was set to half of the reserved bandwidth. Cells 6, 8, 9, 14, 21, and 28 have higher bandwidth requirements, whereas cells 25, 32, 36, 10, 17, 24, 31, 35, 16, 23, 30, 34, 15, 22, and 29 have lower bandwidth requirements. The bandwidth requirement of each cell was a random variable. For cells with insufficient bandwidth, the requirements are less than 80% of the highest bandwidth (100 frequency points) for 80% of the day, less than 90% of the highest bandwidth for 90% of the day, and less than 95% of the highest bandwidth for 95% of the day.

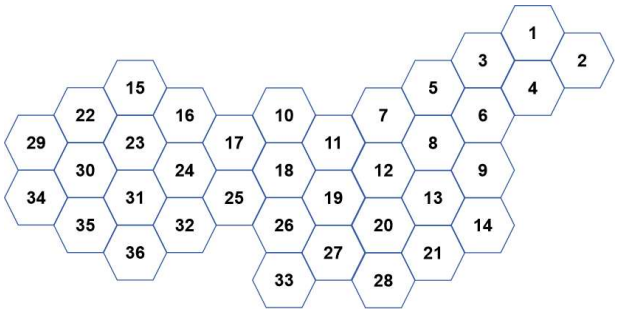

Figure 1. Relative positions of 36 beams.

To address this problem, we must design a bandwidth allocation algorithm for each beam to dynamically allocate frequencies. The algorithm should at least satisfy the bandwidth requirement for cells with insufficient bandwidth, ensure a minimum bandwidth of 25 frequency points for cells with excess bandwidth, satisfy the frequency reuse constraints for adjacent cells, and minimize the maximum gap between frequency points within a cell to reduce the requirements for signal processing hardware on the satellite.

2.1 Problem Modeling

2.1.1 Assumptions

Because the actual application model is far more complex than the theoretical model, we must make certain assumptions about the actual model to abstract a more ideal theoretical model, which is helpful for grasping the essence of the problem. In our research, we made the following assumptions:

1) When updating the channel allocation scheme, the channel used by the cell can be interrupted. Therefore, the new allocation scheme may not need to consider the previous allocation scheme fully.

2) There is no request delay, that is, the generated request can be immediately accepted by the system, and the requests accepted by the system are consistent with the actual requirements.

3) The signal transmission and reception points are stationary and do not move.

4) The satellite is geostationary.

5) The communication quality of the channels is the same, and there are no regional differences in the receiving and transmitting instruments.

6) There is no interference signal outside the cell.

7) The cells are distributed in a two-dimensional plane rather than a three-dimensional spherical distribution.

2.1.2 Assumptions

To facilitate this description, the following conventions are established for abstracting the actual model:

1) ρ: For a given set P, $\rho(P)$ represents the set of subsets.

2) $b(G,v_i,v_j)$: For a graph $G = \langle V, E \rangle$,

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

$$b(G,\ v_i,\ v_j)=\begin{cases}0 & (v_i,\ v_j)\notin E\\ 1 & (v_i,\ v_j)\in E\end{cases} \tag{1}$$

3) Coloring problem: Coloring the vertices of a given graph, requiring that vertices connected by an edge cannot be colored with the same color, and asking for the minimum number of colors required.

2.1.3 Mathematical Model of the Problem

According to the given problem description, the original problem can be abstracted as the following mathematical problem.

Given a graph $G=<V, E>$, an element set $P=\{p_1, p_2, ..., p_n\}$, and a constraint set $S=\{S_1, S_2, ..., S_m\}$, where $m=|V|$, $s_i\in N$. Let V be $\{v_1, v_2, ..., v_m\}$, and W be $\rho(P)$. Based on the function f: $V\rightarrow W$, function C: $V\times V\rightarrow N$ can be defined as follows:

$$C(v_1,v_2)=\begin{cases}0 & (v_1,v_2)\notin E\\ |f(v_1)\cap f(v_2)| & (v_1,v_2)\in E\end{cases} \tag{2}$$

The aim is finding a value of f such that:

$$\sum_{\substack{v_i,v_j\in V\\ i\neq j}} C(v_i,v_j)=0 \tag{3}$$

$$\min\sum_{i=1}^{m}\left|\left|f(v_i)\right|-s_i\right| \tag{4}$$

2.2 Theoretical Analysis

To better grasp the direction of solving a problem and evaluate the effectiveness of the solution, it is necessary to first study the type of problem. The following theoretical analysis is performed for this purpose.

Lemma 1. *Let Y be $\{y_1, y_2, ..., y_m\}$, $y_i\in W$. Then, the original problem can be transformed into finding Y such that:*

$$\min\sum_{i=1}^{m}\left|\left|y_i\right|-s_i\right| \tag{5}$$

$$\sum_{\substack{i,j=1,2,\cdots,m\\ i\neq j}}\left|y_i\cap y_j\right|\cdot b(G,\ v_i,\ v_j)=0 \tag{6}$$

Proof. Let f be a solution to the original problem, and define y_i as $f(v_i)$ for each v_i. Then, it is clear that the set $Y=\{y_1, y_2, ..., y_m\}$ is also a solution to a related problem. Conversely, if $Y=\{y_1, y_2, ..., y_m\}$ is a solution to this related problem, we can define a function f such that $f(v_i)=y_i$ for each v_i. This function f is then clearly a solution to the original problem.

According to Lemma 1, whenever there is a description similar to "Y is a solution to the problem", it refers to the solution described in Lemma 1.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Lemma 2. *Suppose that Y is a legal solution that satisfies the condition in Lemma 1, and there exists an i such that* $|y_i| > s_i$. *Then, under this condition on i, there exists a solution, which we denote as* $Y' = \{y'_1, y'_2, \cdots y'_m\}$, *that satisfies the condition in Lemma 1 and* $\sum_{i=1}^{m} \left||y'_i| - s_i\right| < \sum_{i=1}^{m} \left||y_i| - s_i\right|$.

Proof. Suppose $y'_i = y_i - \{p_k\}, p_k \in y_i, y'_j = y_j$ (j = 1,2, … , m, and j ≠ i)

$$\because \forall j (j = 1, 2, \cdots, m \ \wedge \ j \neq i), \quad 0 \le \left|y'_i \cap y'_j\right| \le \left|y_i \cap y_j\right|$$

$$\therefore 0 \le \sum_{\substack{i,j=1,2,\cdots,m \\ i \neq j}} \left|y'_i \cap y'_j\right| \cdot b(G, v_i, v_j) \le \sum_{\substack{i,j=1,2,\cdots,m \\ i \neq j}} \left|y_i \cap y_j\right| \cdot b(G, v_i, v_j) = 0$$

That is: $\sum_{\substack{i,j=1,2,\cdots,m \\ i \neq j}} \left|y'_i \cap y'_j\right| \cdot b(G, v_i, v_j) = 0$

Furthermore, $\sum_{j=1}^{m} \left||y_j| - s_j\right| = |y_i| - s_i + \sum_{\substack{j=1 \\ j \neq i}}^{m} \left||y_j| - s_j\right| > |y'_i| - s'_i + \sum_{\substack{j=1 \\ j \neq i}}^{m} \left||y'_j| - s'_j\right| = \sum_{j=1}^{m} \left||y'_j| - s'_j\right|$ □

According to the conclusion of Lemma 2, when searching for *Y* that satisfies the conditions, we only need to search for *Y* that satisfies $\forall i (|y_i| \le s_i)$.

Theorem 1. *The original problem can be transformed into t* coloring *problems in polynomial time using* $t \le \max\{s_i | s_i \in S\}$.

Proof. According to Lemma 1, solving the problem described in Theorem 1 is equivalent to solving the original one. According to Lemma 2, we need only search for *Y* that satisfies $\forall i (|y_i| \le s_i)$.

Let Q be $\{\{p_1\}, \{p_2\}, \cdots, \{p_n\}, \phi\}$, and the function $\chi : \rho(G) \to N$ represents the minimum number of colors needed to color the subgraphs of *G*. The allocation function $g : \rho(G) \times W \to Q^m$ indicates that different elements in set *P* are considered as different colors, and the subgraph of *G* is colored with a designated set of colors. A coloring scheme is determined where adjacent vertices cannot be colored with the same color, the uncolored vertices are represented by the empty set ∅, and the number of empty sets must be minimized.

Steps (1) - (5) are followed to obtain a valid solution:

1) set $t = 0$, $k_t = 0$, $Y_t = \{y_{t\,1}, y_{t\,2}, ..., y_{t\,m}\} = \{\emptyset, \emptyset, ..., \emptyset\}$, $S_t = S$, $X_t = \{v_i \mid s_{ti} = 0\}$.

2) $t = t + 1$.

3) Suppose $K_t = \chi(G - X_{t-1})$ and $K = \sum_{i=0}^{t-1} k_i$, if $K + K_t > n$, go to step (5).

4) Let $Z = \{P_{K+1}, P_{K+2}, ..., P_{k+k_t}\}$. Suppose $R \in Q^m$, let $R = \{r_1, r_2, ..., r_m\}$, and let *R* be $g(G\text{-}X_{t\text{-}1}, Z)$, Y_t be $\{y_{t\text{-}1\ 1} \cup r_1, y_{t\text{-}1\ 2} \cup r_2, ..., y_{t\text{-}1\ m} \cup r_m\}$, S_t be $\{s_{t\,1}, s_{t\,2}, \cdots, s_{t\,m}\}$, s.t. $s_{t\,i} = \begin{cases} 0 & s_{t\text{-}1\,i} = 0 \\ s_{t\text{-}1\,i}\text{-}1 & s_{t\text{-}1\,i} > 0 \end{cases}$, X_t be $\{v_i | S_{t\,i} = 0\}$, If X_t is equal to V, proceed to Step (5); otherwise, proceed to Step (2).

5) Let *Y* be Y_t. Y is the solution of problem.

From the above process, it can be observed that the number of iterations *t* is less than or equal to $\max\{s_i | s_i \in S\}$, and in each iteration, a graph coloring problem needs to be solved. Therefore, the original problem is transformed into *t* graph coloring problem in polynomial time.

The proof of Theorem 1 introduces a partitioning strategy, transforming the problem into multiple coloring problems. Despite its lack of theoretical superiority, this approach facilitates practical solutions for problems with relatively simple graphs and a minimum coloring number ≤4, allowing for tractable deterministic algorithms. However, alterations to the original problem conditions, particularly in frequency reuse rules, lead to graph complexity escalations, rendering deterministic algorithms impractical and precipitating exponential time complexity growth.

It is already known that determining the minimum coloring number of a graph is an NP-hard problem [20,21]. Therefore, the original problem is NP-hard. As is well known, the strength of evolutionary

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

algorithms is their ability to solve various difficult problems as opposed to simple ones. Although the actual data of this problem are not very complex, from a mathematical perspective, using evolutionary algorithms is more general and has theoretical significance, and it will not have a significant impact on problem changes. Therefore, EAs are appropriate to solve this problem.

3. Problem-Solving

Based on an analysis of the problem and the proposed model, three algorithms are designed for solving the model. The following is a detailed description of each algorithm.

3.1 Evolutionary Algorithm for Dynamic Channel Allocation Problem

3.1.1 Encoding Methods

A channel allocation scheme for the 36 cells was encoded as a chromosome (i.e. an individual). For any cell, the allowed channel number ranges from 1 to 200, and a binary string of 200 bits is used to represent the channel allocation for each cell, as shown in Figure 2.

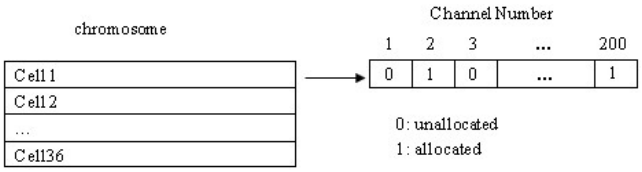

Figure 2. Individual Representation

Therefore, an individual consists of 36 rows, each with 200 bits, and the value of each bit can be either 0 (indicating that the channel is idle) or 1 (indicating that the channel is allocated). The sum of all the elements in a row represents the number of channels allocations for the corresponding cell.

3.1.2 Population Initialization

In the evolutionary algorithm, the initial population was randomly generated. For any cell, if the required number of channels is n, then n bits are randomly selected and set to 1 for that cell's individual, and the remaining bits are set to 0. This initialization method satisfies the channel requirements of the cells as much as possible, without considering the interval between channels.

3.1.3 Fitness Function

To precisely delineate the fitness function employed within the evolutionary algorithm paradigm, we initially introduce and define the subsequent two essential terminologies.

Definition 1. Conflict

In a channel allocation scheme represented by an individual, it is possible that two adjacent cells may use the same channel. When two adjacent cells use the same channel simultaneously, there is a conflict between the two cells.

Definition 2. Spacing

For a channel allocation scheme represented by an individual, the number of consecutive idle channels between the highest and lowest allocated channel numbers in a cell is defined as the channel spacing of the cell. The sum of the channel spacings of all the cells is the total interval of the allocation scheme represented by the individual.

When multi-objective optimization is used to evaluate individuals using Pareto comparison, a solution set containing multiple solutions will be obtained. However, only one allocation scheme is used in the actual satellite channel allocation process, and thus reducing the number of conflicts is more important. Therefore, similar to Salman [22], we used the weighted sum of the total number of conflicts (denoted by collision) and the total spacing of the allocation scheme (denoted by interval) represented by an individual as the fitness value of the individual, where the weight coefficients of collision and interval are α and β, respectively. The fitness function is defined as follows:

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

$$fitness = \alpha \times collision + \beta \times interval$$

Because the focus is on reducing conflicts in the model mentioned in this study, the value of α should be greater than β. In all experiments, we set α=0.9 and β=0.1.

3.1.4 Elimination Strategies

As we employ a single-parent genetic algorithm, the elimination strategy involves parent-child competition and survival of the fittest. This strategy can be likened to the traditional (μ+λ) strategy, with the exception that both μ and λ are set to 1 in this context.

3.1.5 Termination Criteria

The evolutionary algorithm mentioned in this paper utilizes two termination criteria: reaching the maximum number of evolution generations or achieving a fitness value that cannot be further reduced. Given that attaining a fitness value of zero is infeasible for certain test samples, we must cap the number of generations for these samples to manage computation time. After several generations, the algorithm converges to a particular fitness value, which is contingent on the specific sample data and must be ascertained through experimentation.

3.1.6 Parameter Settings

When utilizing the evolutionary algorithm to solve this model, we observed that increasing the population size led to longer computation time, without any corresponding improvement in results. Through experimentation, we determined that the optimal operational efficiency was achieved with a population size of one. Since only a single individual underwent evolution, the offspring size was accordingly set to 1. The maximum number of generations was fixed at 1000, and for the 25 test samples referenced in the experiment, convergence to a specific fitness value was achieved within 1000 generations. Mutation probability refers to the likelihood of performing an operation on a cell, and in our experiments, this probability was set at 0.8.

3.1.7 Algorithm Description

The basic steps of the algorithm are designed as follows.

Algorithm 1: Evolutionary Algorithm for Dynamic Channel Allocation

Step 1: Initialize individual and parameters for the generation.

Step 2: Check whether the termination criteria are satisfied. If so, go to step 7.

Step 3: Call the mutation operator.

Step 4: Calculate the fitness of the new individual.

Step 5: If the new individual is better than the parent, then it is saved.

Step 6: Increment the generation counter and go to step 2.

Step 7: Output the best individual in the population.

In this evolutionary algorithm, the mutation operator is the core of the algorithm; however, a crossover operator was not designed because crossover can easily lead to a sharp decrease in the number of channel allocations for individuals, thereby reducing the total channel allocation of the system and increasing the blocking rate.

The detailed steps of the mutation operator are as follows:

PROCEDURE mutation()

Step 1: Initialize cell label k to 0 and search depth n to 0.

Step 2: If the generated random number is less than the mutation probability *p_mutation*, proceed to step 3. Otherwise, proceed to step 5.

Step 3: Implement a mutation recursion(k,0) with maximum search depth d for the kth cell.

Step 4: Search for the channel number *pos* with maximum conflict among the assigned channels in the kth cell.

Step 5: The channel number pos of the kth cell is set to be unassigned.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Step 6: Increase k by 1.

Step 7: If k reaches the maximum cell number, exit the mutation; otherwise, proceed to step 2.

In the mutation operator, a recursive search process is used, which is actually a mutation with depth of d, that is, to eliminate conflicts within the depth as much as possible. After testing, it was found that selecting a value of six for parameter d led to superior performance. The detailed steps of the recursive search process are as follows:

PROCEDURE recursion(k, 0)

Step 1: Search for the channel number *pos* with the minimum conflict in the unoccupied channels of the kth cell.

Step 2: Assign the channel number *pos* to the kth cell.

Step 3: If search depth n is greater than or equal to maximum search depth d, return.

Step 4: Increase n by 1.

Step 5: Let i indicate the cell number in the neighborhood set C_k of the kth cell and initialize i with 0.

Step 6: Take out cell $C_k(i)$ from the neighborhood set C_k of the kth cell.

Step 7: If the kth cell and cell $C_k(i)$ are assigned the same channel number *pos*, set the channel number *pos* in cell $C_k(i)$ to unassigned, and call recursion($C_k(i)$, n); otherwise, go to step 8.

Step 8: If i is not the last cell in C_k, increase i by 1 and return to step 6, otherwise, return.

The elements in the neighborhood set of each cell are predetermined according to Figure 1, with numbers ranging from two to six.

3.2 Deterministic Algorithm for Dynamic Channel Allocation Problem

While using evolutionary algorithms to solve the dynamic channel allocation model, we found that the evolutionary algorithm can quickly reduce the number of conflicts to zero. However, it is ineffective at reducing channel intervals. Therefore, we designed a deterministic algorithm specifically for channel intervals, which can minimize the increase in intervals while reducing the number of conflicts. Experiments have shown that this algorithm can not only effectively eliminate conflicts but also shorten channel intervals. It should be noted that the representation method and evaluation function of the allocation scheme in the deterministic algorithm are consistent with those in the evolutionary algorithm.

The detailed steps of the deterministic algorithm are as follows:

Algorithm 2: Deterministic Algorithm for Dynamic Channel Allocation

Step 1: Initialize an allocation scheme by setting all channels in all cells to be unallocated.

Step 2: Call the sub-procedure InitCellOrder(Cell) to initialize the cell allocation order and store the cell numbers in the array Cell[36].

Step 3: Initialize the label i to 0.

Step 4: Take out the ith element Cell[i] from the array.

Step 5: Calculate the number of conflicts (i.e.,CollisionNum[j], j=1,2,…,200) caused by each channel bit of Cell[i].

Step 6: Calculate the sum of the number of conflicts in Cell [i] from channel number u to channel number u+Require (Cell[i]), and store it in SumClash[u], where u ranges from 0 to 200-Require (Cell[i]). Require(k) represents the channel demand of Cell[k]. The details are as follows:

(1) $\text{SumClash}[0] = \sum_{m=0}^{Require(Cell[i])-1} \text{CollisionNum}[m]$;

(2) SumClash[u]=SumClash[u-1]-CollisionNum[u-1]+CollisionNum[u+Require(Cell[i])-1], for u=1, 2, ..., 200-Require(Cell[i]).

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Step 7: Search for the smallest element M in the array SumClash[0 to 199], and store the array index (i.e., channel number) of all elements equal to M in the array MinClash[n], where n is the number of smallest elements.

Step 8: Randomly select an element MinClash[k] from the array while i is not equal to 0 or 1.

Step 9: Set the required channels starting from channel MinClash[k] in Cell[i] as allocated.

Step 10: Increment i by 1, and return to step 4 until all cells have been processed.

Step 11: Evaluate the allocation scheme by calculating the number of conflicts, number of intervals, and blocking rate.

PROCEDURE InitCellOrder(Cell)

Step 1: Initialize the allocation order array Cell[36] and the allocation flag array Sign[36] to empty.

Step 2: Select two adjacent cells as the initial allocated cells and store their numbers in Cell[i] (i=1, 2).

Step 3: Set i to 3; Set Sign[Cell[1]] to 1; Set Sign[Cell[2]] to 2.

Step 4: Search for a cell j from the cells not yet added to the Cell array. Cell j must satisfy the following conditions: for the neighboring cell set N={Neighbor(i) | i=1, 2, …, k} of cell j, there must exists at least one $e \in \{1, 2, 3\}$ such that Sign[Neighbor(i)]$\neq e$ for all i=1, 2, …, k, and there must be two cells p, $q \in N$ such that Sign[p] and Sign[q] are two different elements in the set $\{x \mid x \in \{1, 2, 3\}$ and $x \neq e\}$;

Step 5: Add cell j to the array Cell, and set Sign[j]=e.

Step 6: Increment i by 1.

Step 7: If i is less than or equal to 36, go back to step 4; otherwise, return.

3.3 Hybrid Algorithm for Dynamic Channel Allocation Problem

From the previous discussion, we know that using evolutionary algorithms to solve this model can effectively reduce conflicts, thereby decreasing the system blocking rate. Nevertheless, the ability to minimize channel spacing remains unsatisfactory. Deterministic algorithms excel at maximizing the reduction in spacing but lack the conflict-solving capabilities of evolutionary algorithms. Therefore, we propose a hybrid algorithm that aims to effectively reduce conflicts while minimizing channel spacing. The design of this hybrid algorithm combines the strengths of both evolutionary and deterministic algorithms. Through repeated experiments, we have discovered that, in the initial stages of algorithm execution, employing deterministic algorithms to search for individuals with smaller spacing and then initializing the evolutionary algorithm with these individuals leads to better solutions. In a more detailed algorithm design, we integrated the deterministic algorithm into the search process of the evolutionary algorithm, resulting in improved search performance.

The detailed search steps of the hybrid algorithm are outlined as follows:

Algorithm 3: Hybrid Algorithm for Dynamic Channel Allocation

Step 1: Initialize the individual and parameter generation.

Step 2: Invoke the deterministic algorithm to obtain the approximate optimal individual.

Step 3: Evaluate the current individual. If it satisfies the requirements, go to Step 9.

Step 4: Generate a random number rand(0,1). If it is less than the mutation search probability *p_serInMixed*, proceed to Step 5; otherwise, return to Step 2.

Step 5: Invoke the mutation operator to generate a new individual.

Step 6: Evaluating the new individual. If it shows an improvement compared to the parent individual, replace the parent individual; otherwise, discard the new individual.

Step 7: Increment the generation counter: *generation* = *generation* + 1.

Step 8: Check the stopping criterion. If it is satisfied, go to Step 9; otherwise, return to Step 4.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Step 9: Output the current individual and its metrics, then terminate the algorithm.

In both the deterministic and hybrid algorithms, the representation and evaluation methods of individuals remain consistent with those outlined in the previously mentioned evolutionary algorithm. Additionally, the stopping criterion and elimination strategy employed are identical to those used in the evolutionary algorithm.

4. Experiments

In this section, comprehensive experiments were conducted on 25 meticulously designed datasets to assess the efficacy of the evolutionary, deterministic, and hybrid algorithms. The experiments aimed to rigorously analyze and compare the algorithms in terms of their ability to minimize conflicts, intervals, computation time, and blocking rates. To ensure the validity and reliability of our findings, each dataset was carefully constructed to reflect the variability in traffic demands and channel requirements across the 36 beams in our simulated environment.

4.1 Description of Experimental Data and Protocol

The 25 sets of channel requirement samples were generated randomly according to the model requirements. Give that the requirement of each cell is consistent, under the premise of satisfying the frequency reuse constraints, each cell can be allocated a maximum of 66 channels (i.e., Sample 1). However, despite this, there were still a few available channels that were not allocated. Taking into account the adjacency of cells in this model, we designed three sets of requirement data that could be fully allocated, with the number of allocated channels reaching the maximum(i.e., samples 2, 3, and 4). Additionally, to compare the algorithms, we included a dataset where a small number of channels remained unallocated (i.e., sample 5, where each cell required 67 channels). In this scenario, it was impossible to allocate channels in a way that satisfied all cells' demands. Samples 6-15 comprised 10 sets of randomly generated demand data, simulating a Poisson distribution process. The parameter λ in the Poisson distribution represents the average number of events in a given interval. Here, λ is determined to be 79.02 based on the requirement that 90% of the time, the demand should be less than 90% of the highest demand. For cells with higher demand, the highest demand was set to twice the rated demand (100 frequency points), while for cells with medium and low demand, the highest demand was set to 75 and 50 frequency points, respectively. We then utilize the same Poisson distribution to simulate the distribution of demand. Samples 16-25 comprise 10 sets of randomly generated demand data designed to simulate a uniform distribution process. These 10 sets of data fulfill the following criteria: 1) Cells with higher demand should have their demand less than 80% of the highest demand (80 frequency points) 80% of the time; 2) cells with higher demand should have their demand be less than 90% of the highest demand 90% of the time; and 3) cells with higher demand should have their demand be less than 95% of the highest demand 95% of the time.

It should be noted that the samples following the Poisson distribution can only satisfy one of the three aforementioned criteria because the parameter λ is uniquely determined by any single one of them. However, samples that follow a uniform distribution can satisfy all these criteria. Therefore, we included ten sets of data generated according to a uniform distribution. Additionally, all the data generated are greater than 25 (half of the rated demand) and less than 100 (twice the rated demand, i.e., the highest demand).

There are two critical parameters in both the evolutionary and hybrid algorithms: *p_mutation* in the evolutionary algorithm and *p_serInMixed* in the hybrid algorithm. Different values of these parameters yielded different results. Therefore, the two parameters were first set according to the pre-experiments, as shown in Tables 1 and 4 respectively.

To evaluate the performance of the algorithm, two additional metrics were employed in addition to the number of conflicts and intervals used to assess the fitness value: the blocking rate and time consumption.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

All the algorithms were coded in c++ and tested on a PC with an Intel Core(TM) i7-3610QM 2.3GHz CPU and 16G RAM. Given the stochastic nature of the proposed algorithms, they were independently run 30 times for each instance.

4.2 Experiments with Evolutionary Algorithm for DCA

Table 1. Experimental results for the parameter p_mutation

p_mutation / Sample Set	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
66	1.85%	1.26%	0.59%	0.42%	0.21%	0.55%	0.08%	0.08%	0.13%	0.42%
66-67-1	2.04%	1.04%	0.46%	0.71%	0.54%	0.50%	0.33%	0.58%	0.46%	0.67%
66-67-2	1.83%	1.33%	0.71%	0.63%	0.67%	0.79%	0.71%	0.54%	0.71%	0.75%
66-67-3	2.33%	0.96%	0.67%	0.67%	0.38%	0.29%	0.75%	0.50%	0.58%	0.79%
67	2.65%	2.24%	1.53%	1.16%	1.16%	1.33%	1.33%	1.33%	1.41%	1.53%
Poisson-1	3.14%	3.19%	3.19%	3.14%	3.14%	3.14%	3.14%	3.14%	3.14%	3.14%
Poisson-2	3.87%	3.82%	3.82%	3.82%	4.04%	3.87%	3.82%	3.71%	3.76%	3.76%
Poisson-3	3.32%	3.32%	3.26%	3.26%	3.48%	3.48%	3.42%	3.26%	3.26%	3.32%
Poisson-4	1.48%	1.48%	1.59%	1.48%	1.42%	1.48%	1.53%	1.42%	1.42%	1.42%
Poisson-5	2.93%	2.98%	2.93%	2.88%	2.98%	2.93%	2.93%	2.93%	2.98%	2.93%
Poisson-6	3.55%	3.60%	3.45%	3.66%	3.55%	3.45%	3.39%	3.50%	3.50%	3.29%
Poisson-7	4.48%	4.27%	4.27%	4.38%	4.48%	4.43%	4.38%	4.38%	4.32%	4.43%
Poisson-8	3.33%	3.11%	3.39%	3.11%	3.00%	3.28%	3.33%	3.11%	3.33%	2.94%
Poisson-9	1.65%	1.65%	1.65%	1.60%	1.65%	1.65%	1.65%	1.65%	1.60%	1.60%
Poisson-10	1.48%	1.59%	1.53%	1.48%	1.48%	1.53%	1.53%	1.48%	1.48%	1.48%
Uniform-1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Uniform-2	1.66%	1.45%	1.59%	1.52%	1.52%	1.59%	1.73%	1.59%	1.66%	1.87%
Uniform-3	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Uniform-4	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Uniform-5	2.51%	2.58%	2.44%	2.37%	2.37%	2.44%	2.44%	2.44%	2.71%	2.51%
Uniform-6	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Uniform-7	1.30%	1.23%	1.23%	1.23%	1.23%	1.30%	1.23%	1.23%	1.23%	1.30%
Uniform-8	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Uniform-9	1.58%	1.58%	1.51%	1.51%	1.51%	1.51%	1.51%	1.51%	1.51%	1.51%
Uniform-10	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

When solving the model using evolutionary algorithms, the number of generations was set to 1000, and the population size was set to 1 (i.e., a single-individual genetic algorithm was used). Furthermore, to determine an appropriate value for the parameter *p_mutation* in subsequent experiments, the impact of different values of *p_mutation* on the algorithm's performance was tested. Experimental results revealed that the algorithm effectively resolves conflicts and reduces blocking rates; however, it is unable to reduce the channel interval. Therefore, Table 1 only presents the blocking rates obtained using different mutation probabilities for each sample dataset. Based on the experimental results in Table 1, it can be inferred that the influence of *p_mutation* on the performance of the algorithm is insignificant. Nevertheless, for the first five data samples, extremely small values of *p_mutation* resulted in higher blocking rates. Consequently, in subsequent experiments, *p_mutation* was set to 0.8.

Table 2 presents the mean results of the evolutionary algorithm, executed 30 times over 25 unique data samples. The results demonstrate the effectiveness of the proposed evolutionary algorithm in solving the model. The computation time of the algorithm for 36 beams and 200 frequency points remains within one second. For the first five data samples we designed and the data samples satisfying

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

the uniform distribution (Uniform-1 to Uniform-10), the blocking probability of the system is below 2%, except for Uniform-5. For the 10 data samples following the Poisson distribution (Poisson-1 to Poisson-10), the blocking probability increases slightly due to their higher overall demand. These results indicate that using the evolutionary algorithm to solve this dynamic channel allocation problem not only reduces the blocking probability of the system to some extent but also exhibits good time performance.

Table 2. Experimental results with algorithm 1

Sample Set	Total Number of Conflicts	Total Number of Intervals	Fitness	Total System Requirements	Total System Allocations	Computation Time (ms)	Blocking Rate (%)
66	0	3920	392.0	2376	2374	357	0.08
66-67-1	0	4365	436.5	2399	2385	756	0.58
66-67-2	0	4376	437.6	2400	2387	758	0.54
66-67-3	0	4165	416.5	2401	2378	765	0.50
67	0	4475	447.5	2412	2380	841	1.33
Poisson-1	0	3466	346.6	1849	1791	306	3.14
Poisson-2	0	3646	364.6	1834	1766	315	3.71
Poisson-3	0	3659	365.9	1870	1809	343	3.26
Poisson-4	0	3744	374.4	1828	1802	289	1.42
Poisson-5	0	3578	357.8	1877	1822	312	2.93
Poisson-6	0	3461	346.1	1887	1821	401	3.50
Poisson-7	0	3391	339.1	1874	1792	367	4.38
Poisson-8	0	3607	360.7	1801	1745	278	3.11
Poisson-9	0	3541	354.1	1876	1845	310	1.65
Poisson-10	0	3364	336.4	1890	1862	309	1.48
Uniform-1	0	2624	262.4	1266	1266	173	0.00
Uniform-2	0	3041	304.1	1446	1423	218	1.59
Uniform-3	0	3212	321.2	1471	1471	183	0.00
Uniform-4	0	3201	320.1	1384	1384	179	0.00
Uniform-5	0	2946	294.6	1475	1439	248	2.44
Uniform-6	0	3106	310.6	1387	1387	178	0.00
Uniform-7	0	3000	300.0	1544	1525	265	1.23
Uniform-8	0	3071	307.1	1482	1482	180	0.00
Uniform-9	0	3004	300.4	1519	1496	245	1.51
Uniform-10	0	3046	304.6	1427	1427	179	0.00

However, from Table 2, it can be observed that although the obtained allocation schemes have low blocking probabilities while satisfying the frequency reuse constraint (0 conflicts), another requirement for channel spacing is unsatisfied, as it falls within the range of 2500 to 4500. This issue will be addressed in subsequent deterministic and hybrid algorithms.

4.3 Experiments with Deterministic Algorithm for DCA

This section describes the experiments conducted using the deterministic algorithms. Table 3 presents the experimental results obtained using the deterministic algorithm for 25 data samples.

From Table 3, it can be seen that the deterministic algorithm has very low computational time consumption (0.1 seconds for all 25 samples). While satisfying the frequency reuse constraint (0 conflicts), the channel spacing in the obtained allocation schemes is significantly reduced compared with the evolutionary algorithm, all falling below 100. For the first four demand samples, the

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

deterministic algorithm was able to find zero conflict, zero spacing, and zero blocking schemes (i.e., perfect solutions). A perfect solution cannot be found for the fifth sample because there is no perfect solution for that demand data. For the 10 samples following the Poisson distribution (Poisson-1 to Poisson-10), the blocking probability of the allocation schemes obtained using the deterministic algorithm ranged from 2% to 5%, with channel spacings ranging from 20 to 80. For the 10 samples following a uniform distribution (Uniform-1 to Uniform-10), the blocking probability of the obtained allocation schemes fluctuates significantly (ranging from 0% to 8.14%).

Table 3. Experimental results with algorithm 2

Sample Set	Total Number of Conflicts	Total Number of Intervals	Fitness	Total System Requirements	Total System Allocations	Computation Time (ms)	Blocking Rate (%)
66	0	0	0.0	2376	2376	0.175	0.00
66-67-1	0	0	0.0	2399	2399	0.178	0.00
66-67-2	0	0	0.0	2400	2400	0.174	0.00
66-67-3	0	0	0.0	2401	2401	0.175	0.00
67	0	0	0.0	2412	2399	0.183	0.54
Poisson-1	0	27	2.7	1849	1788	0.192	3.30
Poisson-2	0	24	2.4	1834	1773	0.192	3.33
Poisson-3	0	44	4.4	1870	1809	0.195	3.26
Poisson-4	0	21	2.1	1828	1786	0.189	2.30
Poisson-5	0	38	3.8	1877	1811	0.196	3.52
Poisson-6	0	35	3.5	1887	1812	0.200	3.97
Poisson-7	0	46	4.6	1874	1792	0.198	4.38
Poisson-8	0	72	7.2	1801	1720	0.198	4.50
Poisson-9	0	37	3.7	1876	1816	0.195	3.20
Poisson-10	0	42	4.2	1890	1830	0.196	3.17
Uniform-1	0	0	0.0	1266	1266	0.178	0.00
Uniform-2	0	27	2.7	1446	1396	0.190	3.46
Uniform-3	0	19	1.9	1471	1439	0.185	2.18
Uniform-4	0	33	3.3	1384	1325	0.190	4.26
Uniform-5	0	73	7.3	1475	1355	0.202	8.14
Uniform-6	0	8	0.8	1387	1375	0.180	0.87
Uniform-7	0	88	8.8	1544	1442	0.200	6.61
Uniform-8	0	7	0.7	1482	1461	0.183	1.42
Uniform-9	0	60	6.0	1519	1424	0.199	6.25
Uniform-10	0	70	7.0	1427	1354	0.193	5.12

These results show that the deterministic algorithm is highly effective in solving this dynamic channel allocation problem. It not only greatly reduces the channel spacing of the obtained allocation scheme but also has a very low blocking rate and short computation time. Meanwhile, although the deterministic algorithm can effectively reduce channel spacing, it cannot always achieve lower blocking probabilities for certain data samples.

4.4 Experiments with Hybrid Algorithm for DCA

To validate the effectiveness of the hybrid algorithm, experiments were conducted using an identical dataset, and the results were thoroughly examined.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Owing to the introduction of the parameter *p_serInMixed* in the hybrid algorithm, we initially investigated its influence on algorithm performance, as shown in Table 4. Specifically, the channel spacing and blocking rate are presented on either side of the symbol '/'.

Table 4. Experimental results for the parameter p_serInMixed

p_serInMixed / Sample Set	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
66	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	794/0.00%
66-67-1	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	2/0.08%
66-67-2	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%
66-67-3	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	1/0.04%
67	0/0.50%	0/0.54%	0/0.58%	0/0.54%	0/0.54%	0/0.54%	0/0.54%	0/0.54%	0/0.54%	151/0.79%
Poisson-1	43/3.14%	52/3.14%	38/2.92%	47/3.03%	38/2.65%	48/3.03%	38/2.70%	19/2.60%	31/2.97%	875/3.14%
Poisson-2	18/2.73%	21/2.78%	18/2.78%	39/3.11%	12/2.62%	12/2.62%	33/3.00%	24/3.00%	27/2.89%	817/3.27%
Poisson-3	26/2.57%	21/2.67%	34/2.51%	24/2.51%	39/2.78%	35/2.73%	30/2.57%	37/2.73%	31/2.73%	785/2.89%
Poisson-4	16/1.48%	7/1.48%	23/1.64%	18/1.48%	19/1.59%	21/1.59%	23/1.70%	19/1.53%	13/1.48%	1240/1.75%
Poisson-5	42/2.77%	44/2.82%	22/2.40%	34/2.61%	32/2.45%	31/2.45%	32/2.45%	36/2.66%	38/2.66%	872/3.25%
Poisson-6	28/2.70%	36/2.86%	32/2.76%	38/2.81%	35/2.86%	32/2.81%	25/2.76%	33/2.81%	38/2.86%	29/3.50%
Poisson-7	42/3.42%	47/3.63%	51/3.52%	52/3.84%	53/3.58%	56/3.74%	32/3.31%	44/3.52%	37/3.52%	810/3.95%
Poisson-8	43/2.94%	46/2.94%	44/2.78%	48/2.94%	38/2.78%	40/2.78%	40/2.72%	40/2.72%	38/2.78%	839/4.00%
Poisson-9	20/1.71%	19/1.81%	24/1.87%	26/1.87%	29/1.87%	25/1.87%	27/1.87%	29/1.87%	26/1.87%	22/2.08%
Poisson-10	16/1.32%	19/1.38%	13/1.22%	9/1.11%	9/1.16%	19/1.38%	19/1.48%	18/1.32%	3/1.11%	955/1.48%
Uniform-1	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%
Uniform-2	27/2.63%	16/2.07%	27/2.63%	15/1.94%	14/2.01%	26/2.42%	32/2.56%	13/2.01%	35/2.77%	1546/3.04%
Uniform-3	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	921/1.97%
Uniform-4	2/0.43%	3/0.43%	3/0.43%	0/0.43%	4/0.43%	0/0.43%	0/0.43%	3/0.43%	2/0.43%	1472/3.76%
Uniform-5	38/3.80%	41/3.86%	21/3.25%	27/3.46%	17/3.39%	27/3.53%	24/3.46%	26/3.46%	42/4.20%	804/7.66%
Uniform-6	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	1075/0.79%
Uniform-7	20/1.62%	19/1.75%	21/1.68%	18/1.75%	16/1.55%	17/1.68%	22/1.68%	24/1.68%	39/2.85%	1406/5.18%
Uniform-8	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	975/1.08%
Uniform-9	21/1.51%	14/1.38%	19/1.45%	19/1.45%	14/1.38%	14/1.38%	10/1.38%	12/1.51%	18/1.38%	1486/1.58%
Uniform-10	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	0/0.00%	993/4.77%

To select an appropriate value for *p_serInMixed*, the blocking rates obtained using ten parameter values on each dataset were used to calculate the Friedman rank values, as shown in Table 5. According to the Friedman ranking results, selecting 0.5 as the value for the parameter *p_serInMixed* seems to be more suitable.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

Table 5. Friedman ranking of the blocking rates with different values of p_serInMixed

p_serInMixed	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Rank	4.82	5.64	4.78	5.16	4.56	5.2	4.8	4.98	5.48	9.58

Table 6 presents the experimental results obtained using the hybrid algorithm. The results demonstrate that the hybrid algorithm can effectively improve the performance of the channel allocation model. Compared with Algorithm 1, the hybrid algorithm not only comprehensively reduces channel spacing and computation time but also achieves lower blocking rates on more than half of the datasets.

Table 6. Experimental results with algorithm 3

Sample Set	Total Number of Conflicts	Total Number of Intervals	Fitness	Total System Requirements	Total System Allocations	Computation Time (ms)	Blocking Rate (%)
66	0	0	0.0	2376	2376	0.411	0.00
66-67-1	0	0	0.0	2399	2399	0.835	0.00
66-67-2	0	0	0.0	2400	2400	0.822	0.00
66-67-3	0	0	0.0	2401	2401	0.812	0.00
67	0	0	0.0	2412	2399	340	0.54
Poisson-1	0	38	3.8	1849	1800	151	2.65
Poisson-2	0	12	1.2	1834	1786	163	2.62
Poisson-3	0	39	3.9	1870	1818	185	2.78
Poisson-4	0	19	1.9	1828	1799	144	1.59
Poisson-5	0	32	3.2	1877	1831	157	2.45
Poisson-6	0	35	3.5	1887	1833	179	2.86
Poisson-7	0	53	5.3	1874	1807	196	3.58
Poisson-8	0	38	3.8	1801	1751	145	2.78
Poisson-9	0	29	2.9	1876	1841	158	1.87
Poisson-10	0	9	0.9	1890	1868	154	1.16
Uniform-1	0	0	0.0	1266	1266	0.535	0.00
Uniform-2	0	14	1.4	1446	1417	133	2.01
Uniform-3	0	0	0.0	1471	1471	0.659	0.00
Uniform-4	0	4	0.4	1384	1378	126	0.43
Uniform-5	0	17	1.7	1475	1425	135	3.39
Uniform-6	0	0	0.0	1387	1387	0.528	0.00
Uniform-7	0	16	1.6	1544	1520	134	1.55
Uniform-8	0	0	0.0	1482	1482	0.785	0.00
Uniform-9	0	14	1.4	1519	1498	149	1.38
Uniform-10	0	0	0.0	1427	1427	0.913	0.00

Compared with Algorithm 2, the hybrid algorithm performs equally well on the first five datasets. Meanwhile, it achieves smaller or equal channel spacing on all datasets except for 'Poisson-1' and 'Poisson-7', while obtaining lower blocking rates on all datasets. For some datasets (Uniform-1, Uniform-3, Uniform-6, Uniform-8, and Uniform-10), the hybrid algorithm achieves allocation schemes with zero interference and zero blocking.

From these observations, it can be concluded that the hybrid algorithm not only incorporates the ability of evolutionary algorithms to effectively reduce blocking rates but also considers the

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

characteristics of deterministic algorithms to significantly reduce channel interference. Furthermore, the hybrid algorithm demonstrated an excellent time performance, with only one dataset exceeding 200 ms in execution time. Overall, the time performance fell between those of Algorithms 1 and 2. Therefore, this algorithm is considered to be a relatively ideal solution for this channel allocation model.

5. Conclusion

The channel-allocation problem is essentially an optimization problem under constraints. Different system characteristics lead to different channel allocation schemes in mobile cellular and satellite communication systems. Through a study of the dynamic satellite channel allocation problem in 36 Chinese regions, a mathematical model for this type of problem was derived. Theoretical analysis of this model revealed that the problem can be transformed into a graph coloring problem, thereby providing a direction for its solution. Three solution approaches were then proposed for this problem: evolutionary algorithms, deterministic algorithms, and a hybrid algorithm that combines both. Next, through experimental comparisons using five sets of specific data and 20 sets of randomly generated data following a uniform or Poisson distribution, it was found that evolutionary algorithms have strong global search capabilities but have little effect on optimizing the channel spacing within the same cell. However, deterministic algorithms handle channel spacing well and exhibit a good time performance. The hybrid algorithm effectively combines the strengths of both approaches, eliminating conflicts and minimizing the maximum channel spacing within the same cell.

The solution to this problem suggests a new approach for solving similar problems; combining evolutionary algorithms with other algorithms holds the potential to achieve better results. Moreover, because the solution effectively improves the utilization of satellite channel resources, this problem is of great significance in real-life applications.

Due to the high efficiency of this algorithm, it meets the requirements of the problem, and channel-borrowing strategies are not considered in the solution process. Based on the current research findings, we expect further improvements in efficiency can be achieved by using a hybrid channel allocation scheme in combination with channel borrowing strategies. We also considered the impact of other factors, such as mobility, on the channel allocation problem.

Acknowledgments

This study was supported by the Natural Science Foundation of Fujian Province, China (Grant No.2020J01323). I would like to thank the anonymous reviewers for their constructive comments and suggestions, and acknowledge the support of my institution, which provided the resources necessary to carry out this research.

References

[1] P. Shaft, J. Roberts. Optimum Allocation of Multibeam Communications Satellite Resources, IEEE Trans. Commun., vol.24(1976), p.1195-1200.

[2] S. Chen, S. Sun, S. Kang. System integration of terrestrial mobile communication and satellite communication-The trends, challenges and key technologies in B5G and 6G, China Commun, vol.17 (2020), p.156-171.

[3] Q. Li, L.D. Zhu, S.Q. Wu. A Channel Reservation Algorithm with Priorities in LEO Satellite Systems, J. of Electron. & Info. Technol, vol.30(2008)No.8, p.1820-1823.

[4] T. Darwish, G.K. Kurt, H. Yanikomeroglu, et al. LEO Satellites in 5G and Beyond Networks: A Review from a Standardization Perspective, IEEE Access, vol. 10(2022), p.35040-35060.

[5] D. Peng, A. Bandi, Y. Li. Hybrid Beamforming, User Scheduling, and Resource Allocation for Integrated Terrestrial-Satellite Communication, IEEE T VEH TECHNOL., vo.70(2021)No. 9, p.8868-8882.

[6] W.Q. Bao. Research of Resource Allocation in Multi-Beam Satellite Communication System. Master, Beijing University of Posts and Telecommunications, China, 2018. (In Chinese).

ISSN: 2414-1895 DOI: 10.6919/ICJE.202410_10(10).0005

[7] M. Umehira, S. Fujita, Z. Gou. Location-based centralized dynamic channel assignment with channel segregation for multi-beam mobile satellite networks, In Proceedings of 6th International Conference on Wireless Communications and Signal Processing, Hefei, China, 23-25 October 2014, p. 1-5.

[8] R. Chang, Y. He, G. Cui. An allocation scheme between random access and DAMA channels for satellite networks, In Proceedings of the 2016 IEEE International Conference on Communication Systems, Shenzhen, China, 14-16 Dec. 2016, p. 1-6.

[9] X. Liu, K. Xu, F. Wu. A beam-dominating frequency resource allocation and scheduling scheme for multi-beam satellite system, In Proceedings of the 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA) , Shenyang, China, 22-24, January 2021, p. 532-535.

[10] A. Koustaftiki, C. Matrakidis, P. Lane. Improved Dynamic Channel Allocation for Satellite Systems, IEEE Commun. Mag., vol.31(2000)No.1, p.44-48.

[11] I. Bisio, M. Marchese. Packet Loss and Delay Combined Optimization for Satellite Channel Bandwidth Allocation Controls, In Proceedings of the 2008 IEEE International Conference on Communications, Beijing, China, 19-23 May 2008, p. 1905-1909.

[12] J. Wang, Y. Cai. Multiobjective evolutionary algorithm for frequency assignment problem in satellite communications, Soft Comput., vo.19(2015)No. 5, p.1229-1253.

[13] Y. He, Y. Jia, X. Zhong. A traffic-awareness dynamic resource allocation scheme based on multi-objective optimization in multi-beam mobile satellite communication systems, Int. J. Distrib. SENS. N., vol.13(2017)No. 8, p.1-14.

[14] Z. Zhou, Y.L. Ning, X.Y. Zhou. Improved artificial bee colony algorithm-based channel allocation scheme in low earth orbit satellite downlinks, Comput. Electr. Eng., vol. 110(2023), p.108838.

[15] Y.X. Huang, S.F. Wu, Z.K. Zeng. Sequential dynamic resource allocation in multi-beam satellite systems: A learning-based optimization method, Chinese J. Aeronaut., vol.36(2023)No. 6, p.288-301.

[16] X. Hu, S. Liu, R. Chen. A Deep Reinforcement Learning based Framework for Dynamic Resource Allocation in Multibeam Satellite Systems, IEEE Commun. Lett., vol.22(2018), p.1612-1615.

[17] D. Deng, C. Wang, M. Pang. Dynamic Resource Allocation With Deep Reinforcement Learning in Multibeam Satellite Communication, IEEE Wirel. Commun. Lett., vol.12(2023)No. 1, p.75-79.

[18] J. Xu, Z.T. Zhao, L. Wang. A novel deep reinforcement learning architecture for dynamic power and bandwidth allocation in multibeam satellites, Acta Astronaut., vol. 204(2023), p.73-82.

[19] S. Kashyap, N. Gupta. Demand Based Dynamic Bandwidth Allocation in Multibeam Satellites Using Machine Learning, Wireless Pers. Commun., vol.136(2024), p.1243-1260.

[20] R. Balakrishnan, K. Ranganathan. A Textbook of Graph Theory, Springer-Verlag Publisher: New York, NY, USA, 2012.

[21] R. Marappan, G. Sethumadhavan. Complexity Analysis and Stochastic Convergence of Some Well-known Evolutionary Operators for Solving Graph Coloring Problem, mathematics, vol.8(2020)No. 3, p.303.

[22] A.A. Salman, I. Ahmad, M.G.H. Omran. Frequency assignment problem in satellite com-munications using differential evolution, Comput. Oper. Res., vol.37(2010)No. 12, p.2152-2163.