Mathematical and Computational Applications

MDPI

Article

An Experimental Study of Strategies to Control Diversity in Grouping Mutation Operators: An Improvement to the Adaptive Mutation Operator for the GGA-CGT for the Bin Packing Problem

Stephanie Amador-Larrea *, Marcela Quiroz-Castellanos * and Octavio Ramos-Figueroa

Artificial Intelligence Research Institute, Universidad Veracruzana, Xalapa 91000, Mexico; oivatco.rafo@gmail.com
* Correspondence: stephanieamadorlarrea@gmail.com (S.A.-L.); maquiroz@uv.mx (M.Q.-C.)

Abstract: Grouping Genetic Algorithms (GGAs) are among the most outstanding methods for solving NP-hard combinatorial optimization problems by efficiently grouping sets of items. Their performance depends on problem-specific heuristics and a balance between exploration and exploitation. The mutation operator plays a crucial role in exploring new solutions, but improper mutation control can lead to premature convergence. This work introduces adaptive mutation strategies for the GGA-CGT applied to the One-Dimensional Bin Packing Problem (1D-BPP). These strategies control the level of change that will be introduced to each solution dynamically by using feedback on population diversity, enabling better exploration. The proposed approach resulted in a 4.08% increase in optimal solutions (2227 across all classes) and a severe reduction in the average of individuals with equal fitness (from over 50% to less than 1%), enhancing diversity and avoiding local optima. The adaptive strategies were particularly effective in problem instances with larger item weights, where improvements were the most significant. Furthermore, statistical analysis confirmed the adaptive mutation approach's superior performance compared with the original one. These findings demonstrate the potential of adaptive mechanisms to improve genetic algorithms, offering a robust strategy for tackling complex optimization problems.

Keywords: bin packing problem; grouping genetic algorithm; mutation operator; adaptive control

check for updates

Academic Editor: Nicholas Fantuzzi

Received: 12 December 2024
Revised: 1 March 2025
Accepted: 14 March 2025
Published: 18 March 2025

Citation: Amador-Larrea, S.; Quiroz-Castellanos, M.; Ramos-Figueroa, O. An Experimental Study of Strategies to Control Diversity in Grouping Mutation Operators: An Improvement to the Adaptive Mutation Operator for the GGA-CGT for the Bin Packing Problem. *Math. Comput. Appl.* **2025**, *30*, 31. https://doi.org/10.3390/mca30020031

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Combinatorial optimization seeks to discover the optimal configuration from a bounded set of potential solutions in complex problem spaces. Many of these cases belong to the family of grouping problems since they are related to efficiently partitioning a set of items into groups to optimize a specific criterion, usually to minimize or maximize a particular function. The specialized literature includes many NP-hard grouping problems, like the One-Dimensional Bin Packing Problem (1D-BPP), which is considered particularly challenging in such a way that no polynomial-time algorithms can solve all the instances optimally [1]. The 1D-BPP consists in packing a set of n items into m bins with a fixed capacity c, ensuring that the sum of the weights w_j of the items within each bin does not exceed this capacity [2]. The objective is to minimize the total number of bins used while satisfying the capacity constraints. Due to the fact that it is a combinatorial problem, obtaining the solution requires evaluating many possible combinations, and the number of

possible solutions can be greater than $(n/2)^{n/2}$, making exact approaches impractical for large instances.

As a result, heuristic and metaheuristic algorithms have become fundamental tools for obtaining high-quality solutions within reasonable computational time. This has led to the extensive use of metaheuristic techniques, such as local search, swarm intelligence, and evolutionary algorithms, to address grouping problems [3,4]. While these algorithms do not guarantee finding the optimal solution, they can deliver high-quality results within an acceptable execution time, even for large and complex problems. Among the various metaheuristic approaches in the literature for addressing grouping problems, the Grouping Genetic Algorithm (GGA) stands out for its strong performance and adaptability to different problem characteristics and conditions. GGAs extend standard genetic algorithms by integrating a group-based representation scheme and variation operators that function at the group level, making them particularly effective for grouping problems [5]. As a result, the specialized literature includes several variants of the GGA, such as the Grouping Genetic Algorithm with Controlled Gene Transmission (GGA-CGT), designed to solve the 1D-BPP. Its particularity is that it uses genetic components, namely, population initialization strategy, crossover, mutation, and reproduction techniques, designed to operate in a controlled way, favoring the transmission of the best genes while balancing selective pressure and population diversity. In other words, these genetic components are derived from properties and characteristics fundamental to the 1D-BPP and the behavior of the solutions throughout the search process [6].

The success of GGAs lies in their ability to efficiently explore the solution space while addressing specific constraints by exploiting the search directions induced by their variation operators. Among them, crossover and mutation are the most commonly used. However, the literature highlights other operators, such as inversion, cloning, translocation, and injection. These operators play a key role in exploring the solution space, fostering diversity, and guiding convergence towards high-quality solutions. Their efficiency can be significantly improved through the incorporation of adaptive control strategies. Specifically, by integrating multiple mutation or crossover operators, the algorithm gains the ability to dynamically regulate the sequence of variation operations. This adaptability improves the balance between exploration and exploitation and reduces the risk of premature convergence. Consequently, the algorithm becomes more robust and effective in addressing diverse problem instances. The state of the art in GGAs includes methods that exploit the strengths of multiple operators or strategies in their variation operators. Some proposals employ various mutation operators, while others use probabilistic mechanisms to determine the appropriate strategy within the operator to use. This area remains an active field of research, presenting significant opportunities for improving the robustness and performance of GGAs and highlighting the advantages of exploiting multiple operators within variation mechanisms. Integrating various operators or strategies within these, whether for crossover, mutation, or both, improves the algorithm's ability to promote greater diversity in the search process, improves the balance between exploration and exploitation, and increases the chance of reaching optimal solutions across various problem instances. These adaptive strategies represent a promising direction for advancing the robustness and effectiveness of genetic algorithms in solving challenging optimization problems.

Despite its effectiveness and its accreditation as one of the best state-of-the-art algorithms for the 1D-BPP, experimental studies with new instances have shown that the GGA-CGT can converge prematurely to suboptimal solutions when working with some high-difficulty test cases [7]. The effectiveness of the GGA-CGT is caused mainly by the mutation operator, which is the main factor influencing this behavior and applies to more than 80% of the population in each generation. However, the lack of adaptability of this

operator leads to a rigid mutation process that does not adjust to the current state of the population, which either disrupts promising solutions or does not introduce sufficient diversity. This rigidity limits the algorithm's ability to balance exploration and exploitation, making it susceptible to premature convergence. To address this limitation, this work proposes an online adaptive control mechanism for the mutation operator in the GGA-CGT. The objective is to dynamically select the most appropriate mutation strategy for each solution based on population diversity indicators, thus avoiding premature convergence and improving the algorithm's ability to successfully navigate complex search spaces. The fundamental scientific contribution of this work is introducing an adaptive control strategy for mutation in the GGA-CGT, which dynamically selects heuristic strategies based on real-time feedback regarding population diversity. Unlike traditional approaches that apply mutation statically or probabilistically, this method ensures a self-regulating balance between exploration and exploitation, improving adaptability and robustness in different problem instances. Experimental results demonstrate that this approach significantly improves the maintenance of population diversity, reduces the risk of premature convergence, and enhances performance on high-difficulty 1D-BPP test cases.

The structure of this work is organized as follows: Section 2 presents the literature review, providing a comprehensive overview of the related works. Section 3 introduces the GGA-CGT, detailing the components' procedure. Section 4 describes the proposed Adaptive_Mutation + RP operator, describing the methodologies and strategies employed in detail. Section 5 presents the experimentations of the proposed strategies, the results, and the analyses that were performed. Section 6 includes the experimental results of the GGA–CGT with the proposed strategy for controlling diversity in the Adaptive_Mutation + RP operator for the 1D-BPP and the comparison with the original GGA–CGT, demonstrating their effectiveness. Finally, Section 7 concludes the paper by summarizing the essential findings, highlighting the main contributions, and discussing potential directions for future research.

2. Related Work

The 1D-BPP has been extensively studied over recent decades due to its many applications in logistics, telecommunications, manufacturing, and transportation. Its complexity places it in the NP-hard class of problems, meaning that solving it optimally requires significant computational resources as the problem size increases [1]. Despite these challenges, the 1D-BPP remains a highly relevant problem, driving the development of heuristic and metaheuristic approaches that efficiently balance solution quality and computational effort. Among these, Genetic Algorithms (GAs) have been proven particularly effective [8]. The Grouping Genetic Algorithm (GGA), introduced by Falkenauer in 1996, is a significant breakthrough in treating the 1D-BPP and similar grouping problems [9]. The GGA introduced several important innovations in his proposal: (1) a representation scheme that encodes bins as genes rather than individual elements, (2) specialized variation operators for crossover and mutation that are suited to the representation of group structures, and (3) a fitness function adapted to evaluate the efficiency of bin utilization [10]. These features allowed GGA to explore the solution space more effectively, achieving solutions of higher quality and computational efficiency than traditional methods. Building on this foundation, Cruz-Reyes et al. [11] proposed the Hybrid GGA for Bin Packing (HGGA-BP), which introduced heuristics for generating the initial population and hybrid reinsertion strategies for items. This hybrid approach improved the performance of the GGA, in challenging instances of the 1D-BPP. Quiroz-Castellanos et al. [6] further advanced this field by developing the GGA with Controlled Gene Transmission (GGA-CGT), which integrated search heuristics to preserve high-quality groups at the same time as maintaining a critical

balance between selective pressure and population diversity. This balance was essential to preventing premature convergence and ensuring robust solution space exploration. In 2021, González-San-Martín [12] introduced the GGA-CGT/D, which incorporated a problem reduction technique and a diversification mechanism. These enhancements significantly improved performance on the most challenging benchmark instances of the 1D-BPP. Similarly, in 2022, Amador Larrea [13] proposed a new crossover operator, FI-GLX-1, designed for the GGA-CGT to solve the 1D-BPP. This operator, developed through an experimental study, enhanced the algorithm's performance on challenging benchmarks. The study also highlighted the importance of adapting numerical and categorical parameters to the specific characteristics of the problem.

Maintaining diversity within the population is essential to avoiding premature convergence to suboptimal solutions, particularly in the early stages of the search process. A diverse population enables more uniform solution space exploration, increasing the likelihood of discovering high-quality solutions. Parameters such as mutation and crossover rates directly influence diversity, as they control the degree of variation introduced during the evolutionary process [8]. For example, disruptive mutations can explore previously unexplored regions of the solution space, improving diversity, whereas adaptive strategies balance the exploration and improvement of solutions [8]. Parameter management plays a crucial role in the effectiveness of GAs, with two primary approaches: parameter tuning (offline) and parameter control (online). Parameter tuning involves setting static values before execution [14], which, while being straightforward, often leads to suboptimal results for dynamic or complex landscapes [13]. In contrast, parameter control dynamically adjusts values during execution, allowing the algorithm to adapt to the problem's evolving characteristics [15]. Parameter control can be implemented through deterministic methods, which depend on predefined rules; adaptive methods, which use feedback from the search process; or self-adaptive methods, where parameters are part of the genetic representation [14]. Parameters can be categorized into numerical and symbolic types. Numerical parameters include crossover and mutation rates, while symbolic parameters enclose choices such as the type of operator used or the parent selection mechanism [8]. Despite the fact that numerical parameter control has been extensively studied, working symbolic parameters remain unexplored, presenting significant research opportunities.

Several studies illustrate adaptive control strategies for mutation operators. For instance, Bhatia and Basu [16] proposed an adaptive mutation probability regulation strategy for the 1D-BPP, which adjusts probabilities based on the bin space and the algorithm's progress. Singh and Gupta [17] applied a similar approach to the Multiple Knapsack Problem, dynamically calculating the probability of removing an item based on the number of knapsacks and items. Another study [18] introduced an evolutionary process adaptation strategy, starting with a low mutation probability that gradually increased as the algorithm progressed. Chaurasia and Singh [19] focused on parent solution transmission, determining the extent of inheritance based on fixed parameters and parent solution sizes. Jawahar and Subhaa [20] presented a dynamic operator adjustment strategy, calculating crossover and mutation percentages based on a generational gap factor. Inter-operator strategies have also shown promise. Rossi et al. [21] applied such a strategy in a GGA for Fixed Job Scheduling, dynamically adjusting two mutation operators to balance exploration and exploitation. Peddi and Singh [22] alternated between crossover and mutation operators in a GGA for data clustering, using feedback from the population's state to adaptively guide the search. Similarly, Balasch-Masoliver et al. [23] introduced multiple mutation types in a GGA for Multivariate Microaggregation, improving adaptability and diversity.

The literature also includes experimental approaches to designing GGAs. Ramos-Figueroa and Quiroz-Castellanos [24] proposed an experimental framework for high-

performance GGAs, focusing on the individual analysis of algorithm components. In this line of research, the specialized literature includes works such as the study presented in [25], where the authors evaluate and design mutation operators in a GGA for the Parallel-Machine Scheduling Problem $R||C_{max}$, emphasizing problem-specific adaptation. Fernández-Solano [26] and Zavaleta-García [27] applied experimental methodologies to design mutation and crossover operators for image segmentation. Although most works focus on numerical parameter control, adapting these strategies to symbolic parameters is critical to future research. Feedback-driven approaches offer a promising path for dynamically adjusting symbolic parameters and optimizing evolutionary algorithms to meet the demands of increasingly complex problems.

Table 1 presents various applications of the GGA-CGT across different combinatorial optimization problems, including the One-Dimensional Bin Packing Problem (1D-BPP), the U-Shaped Assembly Line Balancing Problem (UALBP), the Parallel-Machine Scheduling Problem $R||C_{max}$, the Image Segmentation Problem (ISP), and the Variable Decomposition in Large-Scale Constrained Optimization Problem (VD-LSCOP). The table highlights key modifications made to the original GGA-CGT to adapt it to each problem, such as changes in crossover operators, mutation strategies, evaluation functions, repair mechanisms, and initial solution generation techniques. These modifications aim to improve solution quality, enhance the balance between exploration and exploitation, and optimize algorithmic efficiency for each specific problem.

Table 1. Applications and modifications of the GGA-CGT for different optimization problems.

Author	Year	Problem	Main Modification
Yorgancılar [28]	2020	1D-BPP, UALBP	Comparison of the techniques used in the GGA-CGT and GGA algorithms is performed, with the objective of measuring and analyzing performance when these techniques are modified one by one or combined.
González-San-Martín [12]	2021	1D-BPP	Problem reduction and a diversification technique are proposed.
Amador-Larrea [13]	2022	1D-BPP	The FI-GLX-1 crossover operator is proposed.
Ramos-Figueroa et al. [25]	2023	$R\|\|C_{max}$	Use of the adapted Gene-Level Crossover (AGLX) operator and the Download Mutation Operator, both specifically adapted to the problem being solved.
Zavaleta-García [27]	2023	ISP	The use of a new evaluation function focused on intracluster distance, random initial population generation, a repair method adapted to the problem, and the use of the FI-GLX-1 crossover operator is proposed.
Fernández-Solano [26]	2023	ISP	The use of a new evaluation function focused on intracluster distance, a random initial population generation, a repair method adapted to the problem, and the use of the Item Elimination operator is proposed.
Perez et al. [29]	2024	1D-BPP, $R\|\|C_{max}$	Part of the coevolutionary cooperation algorithm in one of its phases. Use of the BF-ñ technique for initial solution generation and the Grouping Mutation Operator.
Carmona-Arroyo [30]	2024	VD–LSCOP	Application of the replacement operator techniques and controlled selection from the GGA-CGT to the proposed GGA.

3. Grouping Genetic Algorithm with Controlled Gene Transmission

As was mentioned, one of the best state-of-the-art algorithms for solving the 1D-BPP is the GGA-CGT. In this algorithm, the variation operators and the strategies for selection and replacement are applied in a controlled way, promoting the transmission of the best genes. The GGA-CGT was first proposed by Quiroz-Castellanos et al. [6] and later improved by Amador-Larrea [13], who proposed a new crossover operator. This algorithm aims to maximize the filling of the bins, which also seeks to maximize the fitness values of the population. The fitness function for the GGA-CGT is presented in Equation (1).

$$F_{BPP} = \frac{\Sigma_{i=1}^{m}(S_i/c)^2}{m} \quad (1)$$

In this equation, m is the number of bins, S_i is the sum of the weights of the items in bin i, and c is the capacity of the bins. The GGA-CGT begins generating an initial population of solutions with the FF-$\tilde{n}$ heuristic. For each new solution in the population, first, the $\tilde{n}$ items with weights greater than $c/2$ are placed in separate bins; then, the remaining items are arranged by using the classical First Fit (FF) heuristic on a random permutation of this subset [6]. The variation operators include a crossover operator and a mutation operator. The crossover operator, Fullness_Items-Gene_Level_Crossover-1 [13], generates one child by sorting both parents in descending order according to the filling of each gene (bin), giving priority to the gene with fewer items when two genes have the same filling. The mutation operator used is Adaptive_Mutation+RP [6]. It includes an adaptive function to calculate the number of bins to be eliminated from each solution, eliminating the least full bins of the solution and reinserting the free items with the Rearrangement by Pairs heuristic. The GGA-CGT implements a controlled reproduction technique [6], where all the individuals have a chance to contribute to the next generation by forcing the survival of the best individuals through an elite group of solutions.

Figure 1 shows an example of the controlled selection and replacement for crossover and mutation, illustrating the main concepts. In Figure 1, in the first panel of the image, an instance of the 1D-BPP is presented, consisting of 16 objects with weights ranging from 1 to 10, where each bin has a capacity c of 10. On the left side, the population of solutions is displayed as individuals x_i, showing their genotype with group representation, the fullness of each bin, and the corresponding fitness values. On the right side, the population is shown after being arranged by using the sorting strategy, which arranges solutions from best to worst fitness. Additionally, groups are assigned: the elite group (x_5, and x_4), individuals with repeated fitness values (x_7), and those eligible to be selected as parents for the random group (R) ($x_6, x_1, x_2, x_3, x_8, x_7$) and the good group ($G$) ($x_5, x_4, x_6, x_1, x_2$).

Figure 1 (i) illustrates the crossover process. First, parents are selected for the random group, R (x_2, x_8, and x_7), and the good group, G (x_6, x_4, and x_1). Once the parent groups have been determined, the crossover begins by selecting one parent from each group. The parent solutions P_1 (x_6) and P_2 (x_2) are first sorted based on bin fullness, prioritizing bins with higher fullness. If two bins have the same fullness, preference is given to the bin containing fewer items. After sorting, genes are transmitted by comparing the parents gene by gene: the gene from the bin with the highest fullness is prioritized, and if two bins have the same fullness, the one with fewer items is transmitted first. If both bins are identical, P_1's gene is transmitted first, followed by P_2's. Once the child is finally formed, bins containing duplicate items are removed, and items that are not yet in the solution are freed. In this case, items 0 and 2 are freed. Finally, these items are reinserted into the solution, generating a new child solution with a fitness value of 0.763. This offspring and others produced during the crossover are then reintroduced into the population, replacing the parents from the random group.

Figure 1 (ii) illustrates the mutation operator. Once the children generated through crossover have been reinserted, the population is sorted from the best to the worst solution based on fitness, placing individuals with repeated fitness values at the end of the order. The elite group individuals (x_8 and x_5) and the individuals selected for mutation (x_2, x_6, x_1) are then identified. Mutation occurs in two steps: first, the elite group of individuals are cloned. Once this step is completed, the mutation process begins for the selected individuals. In this case, solution x_1 is mutated, where the last three genes are removed, freeing the items $(3, 15, 13, 7)$. Subsequently, they are reinserted into the solution by using the repair heuristic. After mutation, solution x_1 improves its fitness from 0.748 to 0.945. This process is applied to each selected solution. Finally, the cloned solutions are reinserted into the population,

replacing individuals with repeated fitness values and the worst-performing solutions. The details of the mutation operator are described in Section 3.1.

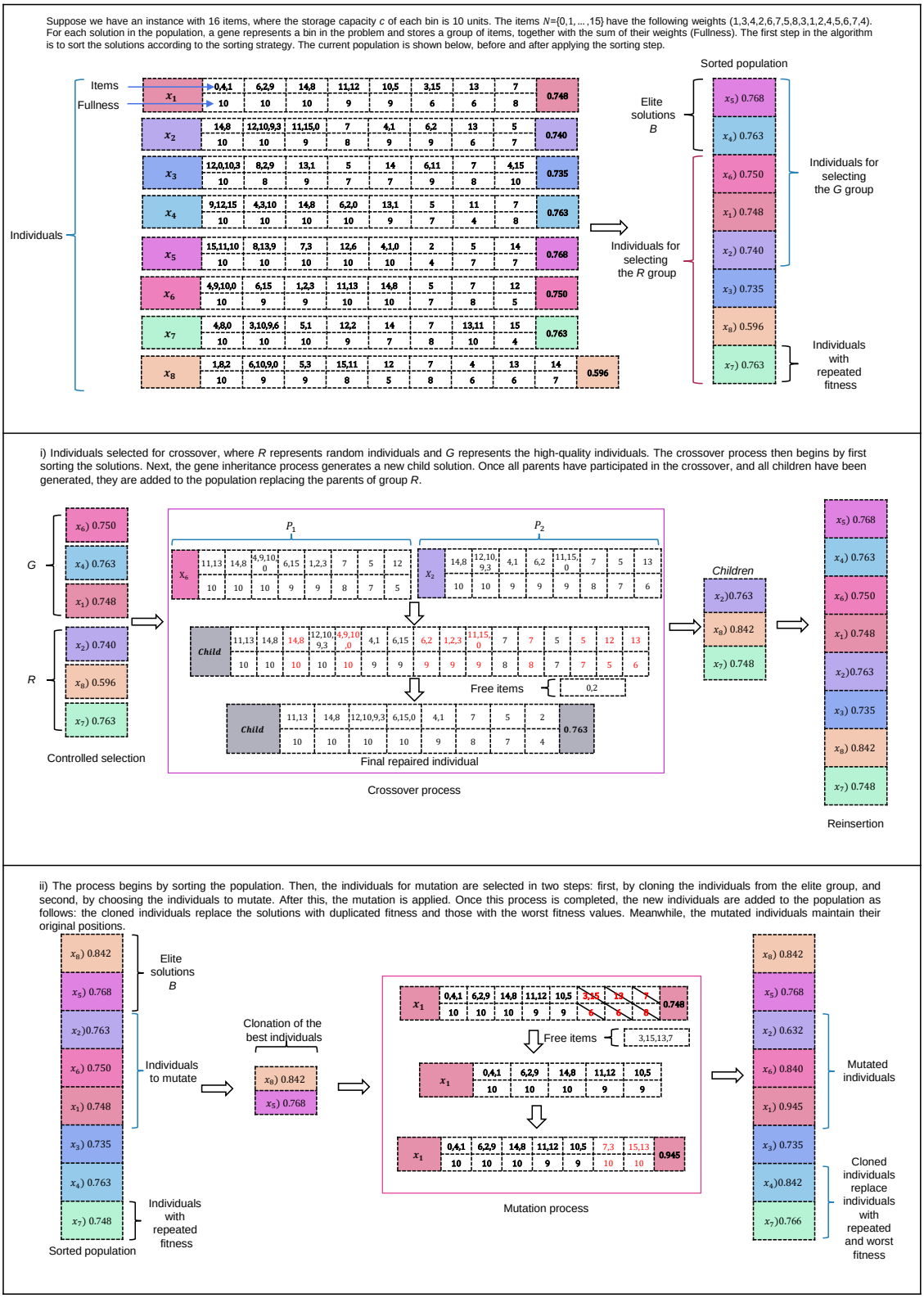

Figure 1. Operation of the GGA-CGT reproduction technique, illustrating the control applied within it for the selection and replacement of solutions for crossover and mutation.

As illustrated in Figure 1, the GGA-CGT applies a controlled reproduction technique in which selection, crossover, and mutation work together to maintain diversity while

preserving high-quality solutions. A crucial component of this process is the mutation operator, which introduces selective modifications to improve exploration and avoid premature convergence. The mutation operator used in the GGA-CGT is Adaptive_Mutation + RP, designed to restructure solutions efficiently while maintaining a balance between exploration and exploitation. The following section details its mechanism, including the strategy for determining the number of bins to be removed and the reinsertion process using the Rearrangement by Pairs heuristic.

3.1. Adaptive Mutation Operator

The mutation operator aims to introduce minor, random changes into the solutions. The operator used in the GGA-CGT is Adaptive_Mutation + RP, which works at the gene level to promote the transmission of the best genes in the chromosome [31]. Before the mutation process, the genes in the solution are sorted in descending order according to their bin fill levels. If the solution being mutated is a cloned solution, a random permutation is applied first; then, the genes are sorted in descending order by bin fill. This operator then eliminates genes (bins) from the solution. The number of bins to eliminate in a solution with m bins is based on the relationship between the solution size and the number of incomplete bins, as defined by the following equation:

$$n_b = \lceil \imath \cdot \epsilon \cdot p_\epsilon \rceil, \tag{2}$$

where $\imath$ represents the number of incomplete bins, ϵ is the elimination proportion, and p_ϵ is the elimination probability:

$$\epsilon = \frac{(2 - (\imath/m))}{\imath^{(1/k)}} \tag{3}$$

$$p_\epsilon = 1 - U(0, \frac{1}{\imath^{1/k}}) \tag{4}$$

Equations (3) and (4) incorporate the parameter $k > 0$, responsible for defining the rate of change of ϵ and p_ϵ with respect to $(\imath/m)$; this parameter must be configured offline, and the larger the value of k, the larger the values of ϵ and p_ϵ. Furthermore, the elimination proportion is inversely proportional to the number of incomplete bins $\imath$ and the percentage of incomplete bins $(\imath/m)$. This implies that the smaller the solution, the higher the percentage of bins to be eliminated, and vice versa.

Rearrangement by Pairs

When the mutation is applied, some bins (genes) are removed from the solution, eliminating items that must be reinserted with the Rearrangement by Pairs heuristic to create a feasible solution. Rearrangement by Pairs (RP) is performed in two stages: (1) all bins are scanned for pairs of packed and free items to be exchanged, to improve the filling of bins, and (2) the free items are reinserted by using a First Fit (FF) heuristic. If there are no free items with a weight greater than half the bin capacity, a random permutation is applied to these items before reinsertion. Otherwise, the free items are sorted in descending order by weight before being added to the solution. This process helps to improve solutions while maintaining a balance between the exploration and exploitation of the search space.

Figure 2 presents an example of the operation of the Adaptive_Mutation + RP operator, illustrating step by step the procedure to correctly perform the mutation.

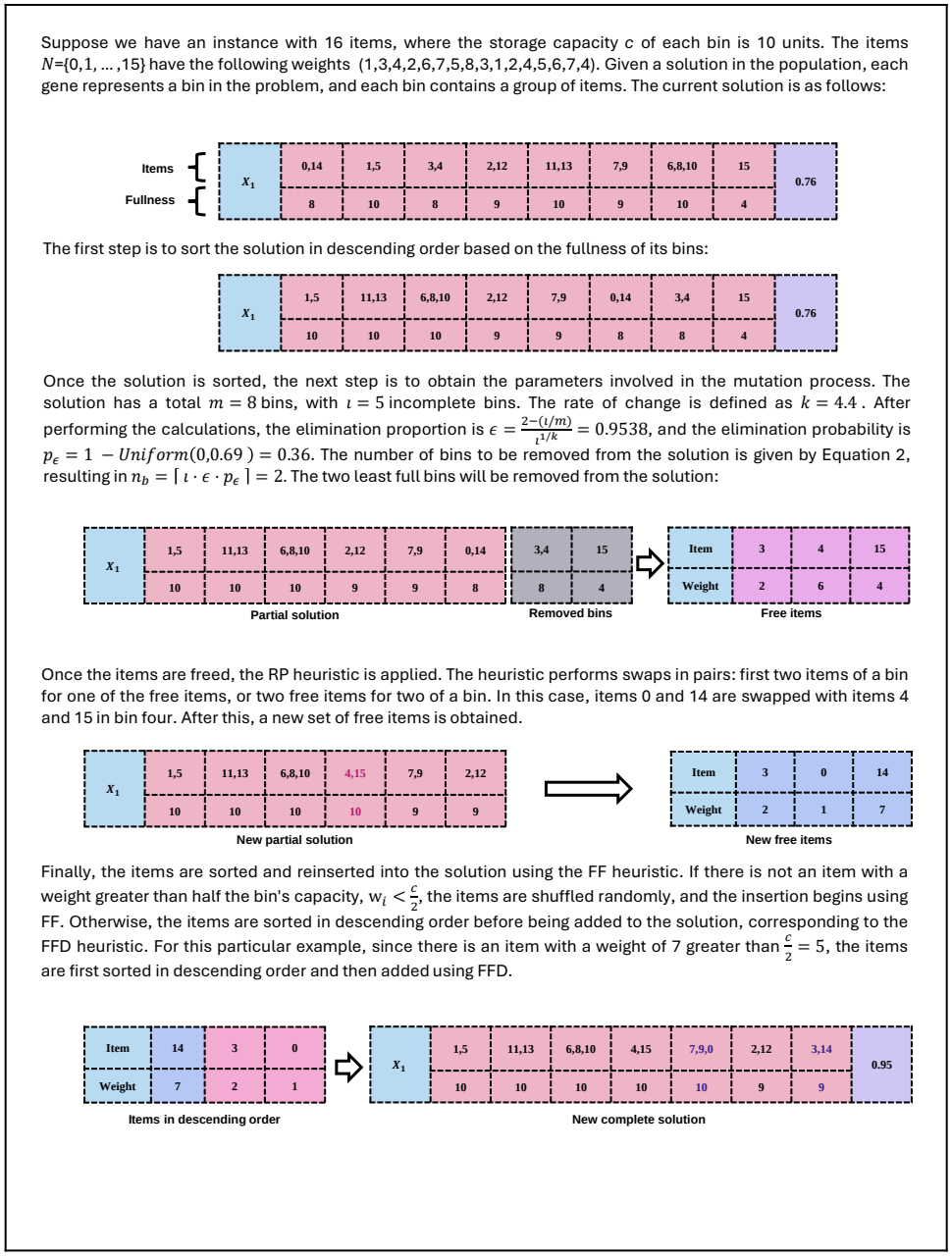

Figure 2. An example of the Adaptive_Mutation + RP operator proposed by Quiroz-Castellanos et al. [6].

3.2. Analysis of GGA-CGT Performance

As mentioned earlier, this research study focuses on an empirical experimental study motivated by observations that selective pressure in the GGA-CGT can lead to variation operators generating solutions with repeated fitness as the evolutionary process advances. This repetition negatively impacts diversity and results in convergence to suboptimal solutions. To address this issue and analyze the GGA-CGT's behavior, we performed the detailed execution of the algorithm to collect relevant behavioral data. It is important to emphasize again that we have worked with the version of the GGA-CGT that includes the crossover operator Fullness_Items-Gene_Level_Crossover-1 (FI–GLX–1) [13]. For the experimentation, the execution of the algorithm was run once for each instance, with the initial seed for random number generation set to 1. The parameter values were configured with the experimental approach used by Quiroz-Castellanos et al. [6]. The values for the parameters were as follows: population size $|P| = 100$, maximum number of generations

$max_gen = 500$, number of individuals selected for the crossover $n_c = 84$, number of individuals selected for the mutation $n_m = 97$, rate of change for non-cloned solutions $k_{ns} = 4.4$, rate of change for cloned solutions $k_{cs} = 5.6$, and finally, maximal age for an individual to be cloned $life_span = 10$.

The BPPv_u_c benchmark set proposed by Carmona-Arroyo et al. [7] was chosen for this research study due to its high difficulty level. This set consists of 2800 instances divided into four classes, each containing seven subclasses. Each subclass includes 100 test cases, with bin capacities c ranging from 10^2 to 10^8. The first class is BPP.25, where the number of items n ranges within [110, 154]; the item weights are uniformly distributed within the $(0, 0.25c]$ range, and the optimal solution requires 15 bins. The second class is BPP.5, where the number of items n ranges within [124, 167]; the weights are distributed within $(0, 0.5c]$, and the optimal solution uses 30 bins. The third class is BPP.75, where n ranges within [132, 165]; the item weights are distributed within $(0, 0.75c]$, and the optimal solution requires 45 bins. The final class is BPP1, where n ranges within [148, 188]; the weights are distributed within $(0, c]$, and the optimal solution requires 60 bins.

From the execution, the data collected for analysis included the number of instances in which the algorithm obtained an optimal solution and the average number of individuals with repeated fitness after mutation. The second measure was calculated by counting the number of individuals with identical fitness values for each generation after applying the mutation operator. Subsequently, this count was averaged across all generations to calculate a representative measure of the algorithm's performance regarding population diversity for each instance. Finally, we estimated the average for each class. This information provides insights into the algorithm's efficiency, population diversity, and capacity to converge to optimal solutions.

Table 2 presents the results achieved by the GGA-CGT with the FI-GLX-1 operator for solving the 1D-BPP. The optimal solutions obtained are displayed by problem class and bin capacity. The first column lists the problem class, the second indicates the bin storage capacity, the third shows the number of optimal solutions obtained for each class and bin capacity, and the final column presents the average of the maximum count of individuals with repeated fitness values within the population after applying the mutation operator.

The algorithm successfully obtains 2113 optimal solutions out of the 2800 available in the complete benchmark, equivalent to 75.4% of the entire benchmark. Additionally, regarding the number of individuals with repeated fitness values, an increase is observed for instances belonging to the BPP.75 and BPP1 classes with larger bin capacity values. In these cases, it has been observed that in some generations, more than fifty percent of the individuals in the population have repeated fitness values, which could indicate a lack of diversity in the population.

Furthermore, Figure 3 illustrates how the number of individuals with repeated fitness values changes after mutation, indicating whether it increases, decreases, or remains unchanged. To create this graph, the percentage of generations showing an increase, a decrease, or no change in the number of individuals with repeated fitness was calculated for each instance within each class and storage capacity. Then, for each class and capacity (comprising 100 instances each), the relative proportions of these percentages were averaged. This process provided the overall averages for each case: increase, decrease, and no change. The graph consists of four subplots, each corresponding to one of the following classes: BPP.25, BPP.5, BPP.75, and BPP1. Each subplot contains seven lines, one for each storage capacity. These lines represent the following percentages: the decrease in the number of individuals (shown in blue), the increase (shown in melon), and no change (shown in pink). The x-axis displays the percentage values, while the y-axis indicates the different storage capacities for each class. For those cases where the bar appears blank, the optimal

solution was obtained in the initial population, so it did not enter the mutation process. Likewise, the cases where only a portion of the bar is visible indicate that the solution of some instances was found in the initial population (blank part of the bars), while the remaining ones went through the mutation process in at least one generation as part of the evolutionary search (visible segment of the bar). Similar to the results presented in Table 2, in Figure 3, it is graphically shown that in the instances belonging to the BPP.75 and BPP1 classes, the number of individuals with repeated fitness values fluctuates continuously after mutation, increasing and decreasing. However, it is also clear that for the instances in the BPP.25 and BPP.5 classes, a higher percentage of cases show a decrease in the number of individuals with repeated fitness values following the mutation process.

Table 2. Results obtained by the GGA-CGT for each subclass of the BPPv_u_c instances. The effectiveness of the algorithm is shown in terms of the number of optimal solutions found and the diversity of the population in terms of the number of individuals with repeated fitness after we applied the original Adaptive Mutation Operator proposed by Quiroz-Castellanos et al. [6].

Class	Bin Capacity	Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation
BPP.25	100	100	1.26
	1000	100	0.11
	10,000	100	0.04
	100,000	100	0.49
	1,000,000	100	1.08
	10,000,000	30	2.11
	100,000,000	0	0.11
	Total	530	0.74
BPP.5	100	100	0.08
	1000	100	0.11
	10,000	99	0.30
	100,000	99	0.96
	1,000,000	2	1.64
	10,000,000	3	0.24
	100,000,000	32	0.28
	Total	435	0.52
BPP.75	100	100	0.15
	1000	100	1.58
	10,000	95	5.86
	100,000	20	65.93
	1,000,000	58	68.46
	10,000,000	78	70.67
	100,000,000	78	70.33
	Total	529	40.43
BPP1	100	100	0.42
	1000	100	1.99
	10,000	58	35.13
	100,000	74	55.75
	1,000,000	94	53.24
	10,000,000	97	55.58
	100,000,000	96	54.78
	Total	619	36.70
TOTAL		2113	19.60

As observed and mentioned, on average, the cases in which the number of individuals with repeated fitness values decreases the most are the instances of the classes in which fewer optima are obtained, that is, the subclasses of BPP.25 and BPP.5 with larger bin

capacity values, which also have the characteristic of not containing items with weights larger than fifty percent of the bin capacity. In these classes particularly, it is observed that the impact of the mutation operator on diversity is positive, since in all but a minimal percentage, individuals with repeated fitness decrease in most cases. However, it is essential to note that in the BPP.75 and BPP1 classes there is a high proportion of individuals with repeated fitness values, with the increase being higher when the capacity of the bins is increased. This supports the conjecture that the algorithm may be converging to non-optimal areas within the search space. In addition, selective pressure and control may not allow more regions to be explored, decreasing diversity in the population, and resulting in suboptimal solutions.

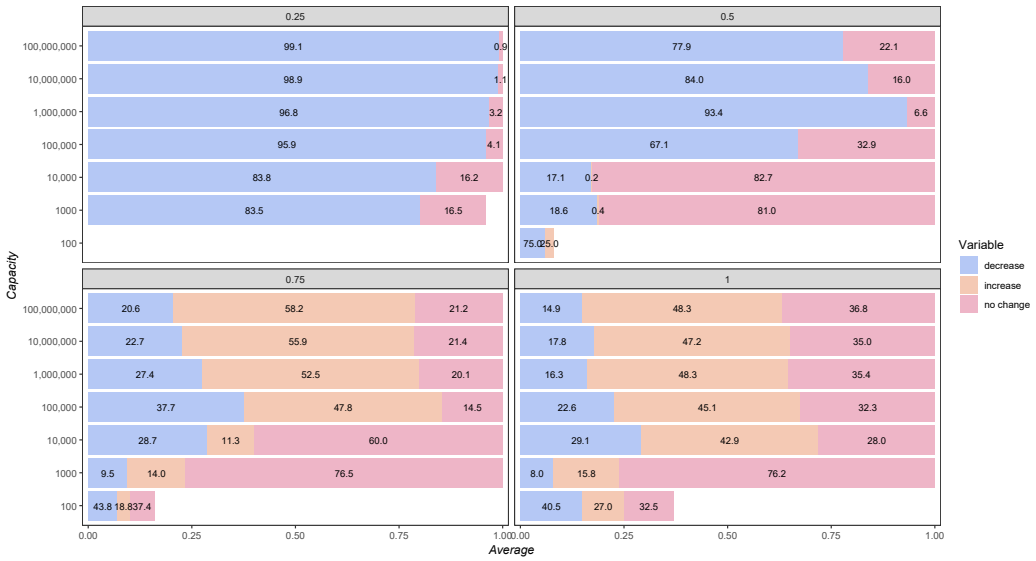

Figure 3. Proportion of generations in which the number of repeated fitness individuals increased, decreased, or remained unchanged during the execution of the GGA-CGT.

Based on the above, an experimental study is proposed to explore strategies that could be used as part of the mutation operator to control diversity within the algorithm and address the issue of multiple individuals with repeated fitness values. This would result in greater diversity within the population, allowing better exploration of the search space by covering a more significant portion of the search space and ultimately converging towards the global optima of the problem's solutions.

4. Experimental Study to Control the Diversity of the Adaptive Mutation Operator

Based on observations and analyses of the execution of the GGA-CGT and insights into selective pressure and control, this study proposes exploring potential areas for improvement within the algorithm. These enhancements focus specifically on strategies within the mutation variation operator. Two critical processes within the mutation operator are proposed for study. The first proposal focuses on how to consider the sort of the genes of solutions to be mutated, which impacts the selection of the bins (genes) to be eliminated from each solution. Doing so facilitates and promotes diversity through the adaptive management of selective pressure. The second proposal centers on the reinsertion of the free items within the Rearrangement heuristic.

4.1. Diversity Control with Adaptive Sorting Strategies of Mutation Operators

We analyzed two approaches for managing population diversity through the Adaptive_Mutation + RP of the GGA-CGT. The first involves the sorting of solution genes before

mutation. Initially, as described, this was performed by sorting them in descending order based on gene filling. A random permutation was applied for cloned solutions first, followed by their sorting in descending order. This method aimed to allow solutions to retain the best genes (since mutation removes the worst, or least filled, genes); however, this approach may limit the search space by not enabling more significant disruption. The second process concerns the reinsertion of items in the second stage of the Rearrangement by Pairs heuristic, where the final free items are introduced to the solution by using the FF packing heuristic to ensure valid solutions. Exploring alternative strategies for sorting these free items before reinsertion could improve the mutation process, allowing for a more expansive range of potential solutions and facilitating escape from local optima. The potential strategies for controlling the sorting within the mutation operator are detailed in the following subsections. To evaluate the impact of these sorting strategies, we conducted experiments comparing different approaches to gene and item sorting. The results of these experiments are presented in Section 5.

4.2. Proposal 1: Gene Sorting

The proposals focus on determining how to sort the genes before removing the last genes in the sorted solution. Strategy 1 involves performing only a random permutation of the genes, regardless of the population's characteristics. Strategy 2, on the other hand, uses the value obtained from a threshold function and a random number r to decide whether to sort the genes by using a random permutation or the algorithm's predefined sorting method. The proposed function for the threshold is as follows:

$$threshold = \frac{1}{g} \sum_{i=1}^{g} \frac{n_i^*}{|P|}, \tag{5}$$

where $|P|$ is the population size, n_i^* is the number of individuals with unique fitness values, and g is the number of the current generation. The threshold function is updated at each generation by calculating the average proportion of individuals with unique fitness values relative to the total population, accumulated over all previous generations. When the number of individuals with repeated fitness values increases, the threshold value decreases more rapidly. Conversely, when it is lower, the threshold decreases more slowly. Additionally, this threshold rule ensures that when the number of individuals with unique fitness values is high (and the number of repeated values is low), the threshold remains stable or decreases very little. The opposite occurs when the number of unique fitness values is low.

For sorting strategy 2, the rule is as follows: Once the threshold value is calculated for a given generation, the next step is to generate a random value r with a uniform distribution, such that $r \sim \mathcal{U}(0, 1)$. If r is greater than the threshold, then only a random permutation is performed on the genes of the solution. If the opposite case occurs, the genes are sorted by using the default operator within the algorithm, which consists of sorting the genes of the solution in descending order concerning the filling of the bins. The objective is that as the number of individuals with repeated fitness values increases, the threshold value decreases, giving a higher probability that the generated random value will be greater than the threshold. As a result, the genes of the parents will be randomly sorted, to introduce more diversity in the mutated solution. In Equation (6), the function representing the adaptive sorting strategy is presented:

$$\text{Sorting_genes} = \begin{cases} \text{Random permutation} & \text{if } r > threshold, \\ \text{Sort the bins in descending order of their filling} & \text{if } r \leq threshold. \end{cases} \tag{6}$$

The proposals were implemented within the GGA-CGT, performing an online inter-operator parameter control.

4.3. Proposal 2: Item Sorting

The second proposal focuses on how free items are selected to be added to the solution by using the First Fit (FF) heuristic to generate a complete solution in the second stage of the Rearrangement by Pairs heuristic. As previously explained, the original strategy for sorting free items operates as follows: If at least one of the free items has a weight greater than half the bin's storage capacity, the items are sorted in descending order by weight. Otherwise, when there are no large items, they are arranged through a random permutation.

The strategies proposed for sorting the items are similar to those explored for gene sorting, as follows:

1. In the first approach, all the items are arranged by using a random permutation.

$$\text{Sorting_items} = \text{Random permutation}. \tag{7}$$

2. The second method leverages the threshold defined in Equation (5). The rule is as follows: A random number is generated with a uniform distribution, such that $r \sim \mathcal{U}(0,1)$. If $r > \mathit{threshold}$, a random permutation is applied to the items; otherwise, they are sorted in descending order according to their weights.

$$\text{Sorting_items} = \begin{cases} \text{Random permutation} & \text{if } r > \mathit{threshold}, \\ \text{Sort items in descending order of their weights} & \text{if } r \leq \mathit{threshold}. \end{cases} \tag{8}$$

Similar to the previous approach, this rule was implemented in the GGA-CGT, within the RP heuristic. After performing the swaps between items, the missing items in the solution are added by using the FF method. The critical difference lies in how these items are sorted before being selected, where the proposed strategy was introduced.

5. Experimental Results and Analysis

This section presents the proposed evaluation of the ordering strategies within the mutation operator. The analysis focuses on the number of optimal solutions, the diversity of the population, and the individuals with repeated fitness. The experimental conditions for this study were consistent with those detailed in Section 3.2: we use the same configuration for the parameters, and for each instance, the execution of the algorithm was run once with the initial seed for the random number generation set to 1. All the graphs that depict the percentages of individuals whose fitness values increase, decrease, or remain unchanged presented in this work were generated as follows: For each instance within each class and storage capacity, the percentage of generations showing an increase, a decrease, or no change in the number of individuals with repeated fitness was calculated. These percentages were then averaged across 100 instances per class and capacity to compute the relative proportions for each case: increase, decrease, and no change. Each graph consists of subplots corresponding to the four classes analyzed: BPP.25, BPP.5, BPP.75, and BPP1. Within each subplot, lines are presented for the seven storage capacities, illustrating the proportion of individuals whose fitness values decreased (shown in blue), the increase (shown in melon), and no change (shown in pink). The x-axis represents percentage values, while the y-axis displays the different storage capacities for each class. These results ensure a comprehensive representation of how mutation impacts the fitness of individuals across various scenarios.

5.1. Performance of Gene Sorting Strategies

The results of applying the gene sorting strategies are summarized in Table 3. This table shows the number of optimal solutions obtained by class and bin capacity with the implementation of the rule in strategies 1 and 2. We observe that a greater number of optimal solutions is achieved when utilizing the proposed adaptive strategy to control diversity by obtaining feedback on the population diversity during the evolutionary process (strategy 2). In the case of the strategy that always applies a random permutation to sort the genes, a total of 675 optimal solutions are obtained, representing only 24.1% of the total instances in the benchmark dataset. Compared with the original version, this means a 51.3% decrease in the number of optimal solutions, with reductions observed across all classes. Additionally, the number of individuals with repeated fitness values shows a significant decrease, particularly in the classes and capacities where fewer optimal solutions are achieved. The last is primarily due to the drastic reduction in selective pressure, which leads to such a high level of diversity among solutions that they fail to converge effectively.

For the second strategy, where the threshold rule is used, we observe that with the proposed rule, 78.6% of cases are solved optimally, compared with 75.4% in the original version; this represents an increase of 3.2% in optimally solved instances, equivalent to 88 additional cases. Furthermore, we measured the number of individuals with repeated fitness to assess the impact of exploring more regions within the search space. In the original version, as mentioned earlier, certain instances with specific storage capacities show that the BPP.75 and BPP1 classes exhibit more individuals with repeated fitness after the mutation process. This behavior persists when the algorithm is executed with the proposed threshold rule. However, despite this, a significant reduction in the average number of individuals with repeated fitness is achieved, especially in the BPP.75 class, where there is a decrease of over 30 individuals with repeated fitness in the population, and in the BPP1 class, with a reduction of up to 20 individuals. For the BPP.25 and BPP1 classes, there is an average increase of only one individual in the population, yet there is a noticeable rise in the number of optimal solutions found.

Additionally, Figures 4 and 5 illustrate the average of individuals with repeated fitness by class and capacity for the proposed gene-sorting strategies, namely, random sorting and sorting based on the threshold rule. These proportions are shown as the percentage of individuals whose fitness increased decreased or remained unchanged after the mutation process. Starting with the version that applies a random permutation before mutation, in Figure 4, it is observed across all four classes that for capacities greater than 10^3, the proportion of generations where the number of individuals with repeated fitness increases is less than one or even zero. There is a higher proportion for capacities of 10^2 and 10^3, even exceeding 50% in the BPP.5 class. This could be attributed to the randomness introduced, shifting from a fully controlled process favoring the best genes to allowing randomness to prioritize any gene. Furthermore, this is the class where the fewest optimal solutions were obtained.

For the version that employs the threshold rule to decide whether to sort by using a random permutation or in descending order, Figure 5 illustrates how proportion of generations in which individuals with repeated fitness changed, increased, decreased, or remained the same. In this proposal, the BPP.25 and BPP.5 classes exhibit very similar behavior, with a negligible or nearly negligible proportion of repeated fitness increases in instances with large storage capacities. However, for capacities of 10^2 and 10^3, there is a noticeable increase in individuals with repeated fitness. A distinct tendency occurs in the BPP.75 and BPP1 classes; most instances show either a significant increase or no change in the number of individuals with repeated fitness after mutation, and reductions are observed in only a small proportion of cases. This could suggest that while the proposal initially

reduces the number of individuals with repeated fitness, the selective pressure eventually drives the search back into regions where premature convergence occurs, resulting in reduced diversity.

Table 3. Number of optimal solutions found and the diversity of the population in terms of number of individuals with repeated fitness after the execution of the GGA-CGT with the Adaptive Mutation Operator proposed by Quiroz-Castellanos et al. [6] with the random sorting of bins and with an adaptive strategy to control diversity. The table highlights in bold the case where the highest number of optimal solutions was achieved.

Class	Bin Capacity	GGA-CGT + Adaptive_Mutation + RP with Random Sorting of Genes		GGA-CGT + Adaptive_Mutation + RP with Adaptive Strategy to Sort Genes	
		Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation	Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation
BPP.25	100	100	1.26	100	1.26
	1000	100	0.11	100	0.07
	10,000	5	0.06	100	0.04
	100,000	0	0.00	100	0.38
	1,000,000	0	0.00	100	1.00
	10,000,000	0	0.00	37	2.55
	100,000,000	0	0.00	0	0.07
	total	205	0.20	**537**	0.77
BPP.5	100	100	1.20	100	1.26
	1000	44	0.79	100	0.19
	10,000	0	0.02	100	0.31
	100,000	0	0.00	99	0.99
	1,000,000	0	0.00	1	2.07
	10,000,000	0	0.00	6	0.14
	100,000,000	0	0.00	33	0.29
	total	144	0.29	**439**	0.75
BPP.75	100	100	8.62	100	3.22
	1000	2	1.06	100	1.88
	10,000	0	0.06	92	3.90
	100,000	0	0.00	26	28.95
	1,000,000	0	0.00	67	33.49
	10,000,000	0	0.00	94	34.33
	100,000,000	0	0.00	95	34.63
	total	102	1.39	**574**	20.06
BPP1	100	100	10.26	100	6.52
	1000	4	1.03	100	2.00
	10,000	9	0.56	65	18.67
	100,000	24	0.61	87	29.95
	1,000,000	31	0.66	100	31.90
	10,000,000	29	0.67	100	31.26
	100,000,000	27	0.61	99	32.55
	total	224	2.06	**651**	21.84
Total		675	0.99	**2201**	10.85

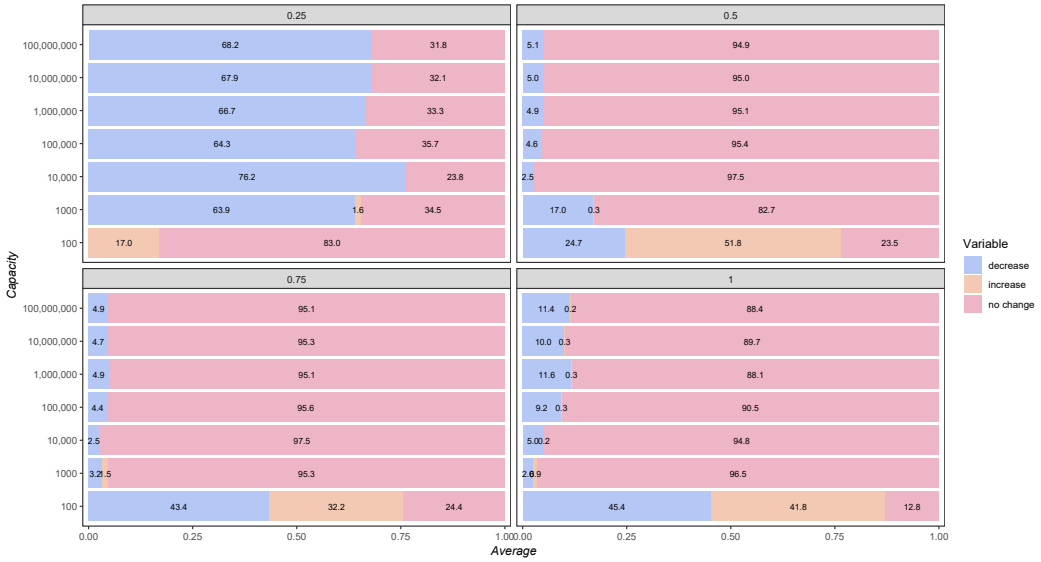

Figure 4. Proportion of generations in which the number of repeated fitness individuals increased, decreased, or remained unchanged during the execution of the GGA-CGT with random gene sorting.

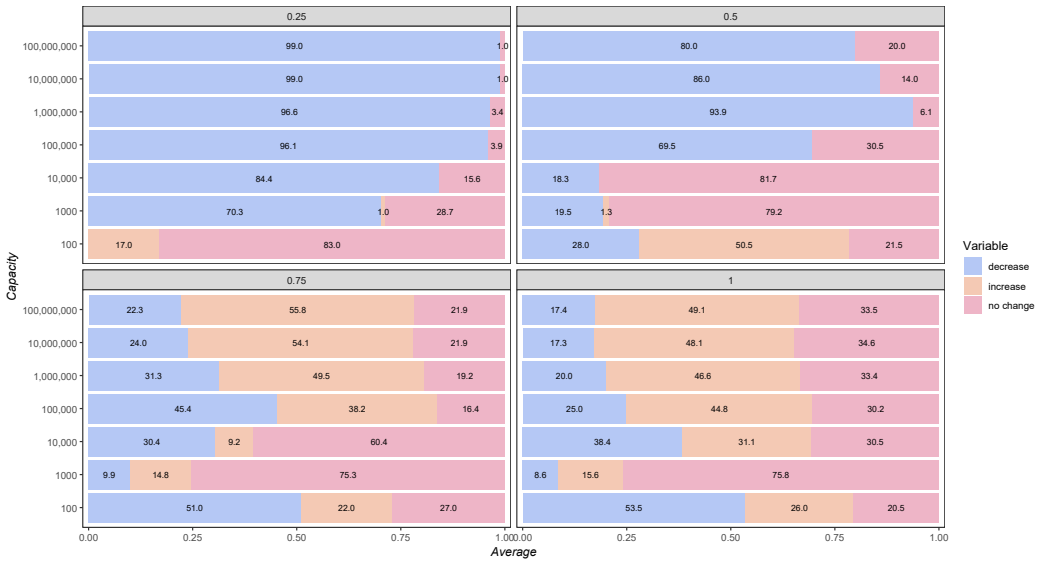

Figure 5. Proportion of generations in which the number of repeated fitness individuals increased, decreased, or remained unchanged during the execution of the GGA-CGT with the threshold rule.

5.2. Performance of Free Item Sorting Strategies

The results of the experimental approach for sorting free items are presented in Table 4. This table includes a column showing the number of optimal solutions obtained for each class and bin capacity, along with an additional column displaying the number of individuals with repeated fitness values after mutation. Notably, the random permutation method achieves the highest number of optimal solutions, totaling 2192, which corresponds to 78.3% of the overall results. In comparison, the strategy that sorts items based on the threshold condition produces 2171 optimal solutions, representing 77.5% of the total cases solved optimally. These findings underscore that consistently applying random permutation sorting yields the best outcomes, delivering 2.82% more optimal solutions than the original method, equivalent to an additional 79 optimal solutions.

Table 4. Number of optimal solutions found and the diversity of the population in terms of number of individuals with repeated fitness after the execution of the GGA-CGT with the Adaptive Mutation Operator proposed by Quiroz-Castellanos et al. [6] with the random sorting of items and with an adaptive strategy to sort the items in the second stage of the RP heuristic. The table highlights in bold the case where the highest number of optimal solutions was achieved.

Class	Bin Capacity	GGA-CGT + Adaptive_Mutation + RP with Random Sorting of Items		GGA-CGT + Adaptive_Mutation + RP with Adaptive Strategy to Sort Items	
		Optimal Solutions	Average number of Individuals with Repeated Fitness After Mutation	Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation
BPP.25	100	100	0.00	100	0.00
	1000	100	0.03	100	0.02
	10,000	100	0.04	100	0.02
	100,000	100	0.48	100	0.39
	1,000,000	100	1.07	100	0.96
	10,000,000	30	2.11	21	1.94
	100,000,000	0	0.11	0	0.06
	total	**530**	4.15	521	0.48
BPP.5	100	100	0.12	100	0.12
	1000	100	0.11	100	0.09
	10,000	99	0.29	99	0.20
	100,000	99	0.96	97	1.00
	1,000,000	2	1.64	3	1.53
	10,000,000	3	0.24	5	0.30
	100,000,000	32	0.28	29	0.30
	total	**435**	5.05	433	0.51
BPP.75	100	100	0.92	100	0.92
	1000	100	0.21	100	0.22
	10,000	95	0.94	94	1.13
	100,000	27	3.95	22	5.65
	1,000,000	65	2.32	66	2.54
	10,000,000	96	0.56	96	0.54
	100,000,000	99	0.24	100	0.21
	total	**582**	9.33	578	1.60
BPP1	100	100	1.13	100	1.08
	1000	100	0.46	100	0.32
	10,000	65	4.13	59	4.47
	100,000	81	2.15	82	2.79
	1,000,000	99	0.37	98	0.42
	10,000,000	100	0.10	100	0.12
	100,000,000	100	0.11	100	0.12
	total	**645**	6.25	639	1.33
Total		**2192**	6.19	2171	0.98

Regarding the number of individuals with repeated fitness, we observe a significant decrease in these proposals, particularly in the BPP.75 and BPP1 classes. In some cases, the number of individuals with repeated fitness drops from over 55 in the original version to less than 1 (for the BPP1 class, instances with a capacity of 1,000,000) when using the threshold conditional proposals. This trend is also evident in the proposal that employs the random number conditional. As in the original version, a graph illustrates how the number of individuals with repeated fitness changes after mutation. The graph shows whether the

number increases, decreases, or remains the same for the random gene sorting proposal and the threshold rule approach. The results are displayed in Figures 6 and 7.

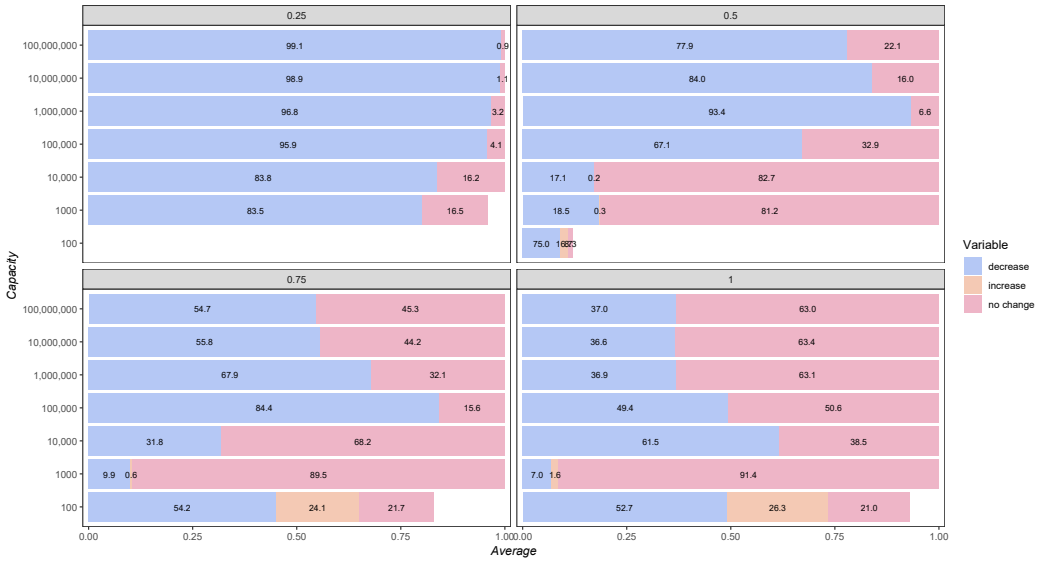

Figure 6. Proportion of generations in which the number of repeated fitness individuals increased, decreased, or remained unchanged during the execution of the GGA-CGT with random sorting for free item sorting.

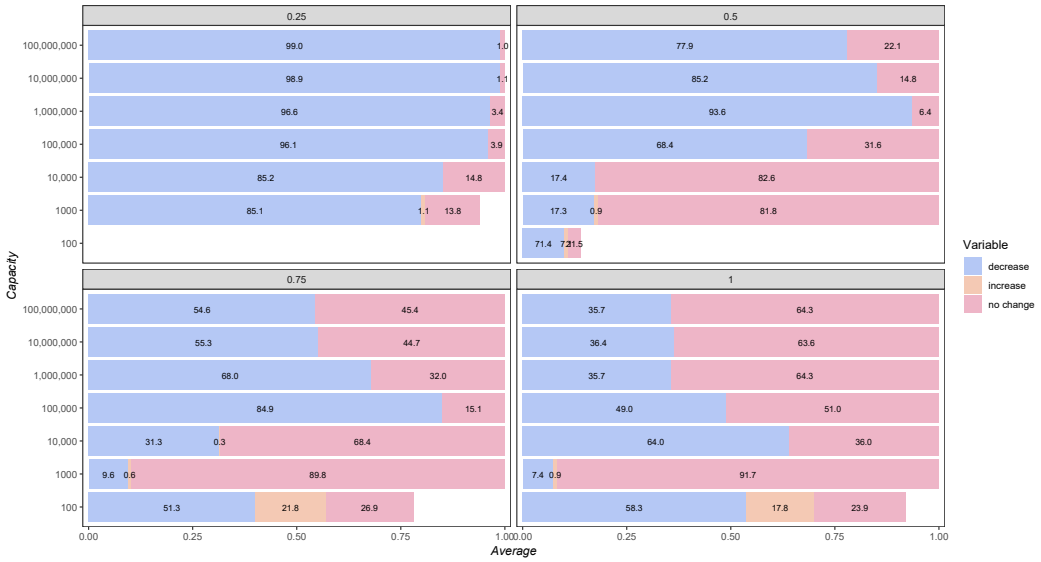

Figure 7. Proportion of generations in which the number of repeated fitness individuals increased, decreased, or remained unchanged during the execution of the GGA-CGT with the threshold rule for free item sorting.

For these sorting proposals, in the BPP.25 class, there is a consistent decrease or no change in the proportion of individuals with repeated fitness in all instances. In the BPP.5 class, it is observed that for instances with a capacity greater than 10^4, the number of repeated fitness individuals decreases or remains unchanged after mutation. For capacities of 10^3 and 10^4, the proportion of individuals showing an increase in repeated fitness is minimal, less than 1%. In the BPP.75 and BPP1 classes, for instances with a capacity of 10^2, there is an observed increase in the number of individuals with repeated fitness after mutation, though this accounts for only about 25% of the total. The proportion of instances showing a decrease or no change is higher, indicating better diversity retention overall. Specifically, in the version that yields the most optimal solutions, we observe the most

notable decrease in individuals with repeated fitness after mutation, achieving the lowest recorded counts.

6. Comparison Between the Original GGA-CGT and the Adaptive Strategy for Mutation Control

In this comparison, we examine the differences between the original GGA-CGT and the version enhanced with the adaptive strategy for mutation control. To begin, we present the adaptive strategy for mutation control. The strategy, derived from this study, focuses on control for sorting mutation strategies using an adaptive approach. It employs a gene sorting method based on a threshold rule before the elimination of genes in the mutation operator. A random number r is drawn from a uniform distribution in each generation. If r exceeds the threshold, a random permutation of the genes in the solution is performed. Conversely, if r is less than or equal to the threshold, the genes are sorted according to the default operator within the algorithm. This sorting is combined with arranging free items through a random permutation in the second stage of the Rearrangement by Pairs heuristic.

The experimental conditions for this study were the same as detailed in Section 3.2: we used the same configuration for the parameters, and for each instance, the execution of the algorithm was run once with the initial seed for the random number generation set to 1.

The results obtained for the adaptive strategy are presented in Table 5. Regarding the number of optimal solutions found, it was 2227, 114 more than the original GGA-CGT, representing 79.54% of the total set, a 4.08% improvement over the original version. Additionally, concerning the number of individuals with repeated fitness, a significant decrease in the average number of individuals with repeated fitness after the mutation process is observed. This decreased from over 50% of individuals with repeated fitness to less than 1% in the classes with the highest incidence of repeated fitness, namely, the BPP.75 and BPP1 classes (see Figure 8). A slight reduction is also noted for classes with smaller items, that is, BPP.25 and BPP.5. However, since fewer individuals were in the original GGA-CGT, the change is less pronounced. Regarding the number of optimal solutions, the increase is particularly notable in the BPP.75 and BPP1 classes, which coincides with the most significant decrease in individuals with repeated fitness. This suggests that introducing more diversity allows the algorithm to explore other regions within the search space, enabling it to converge to optimal solutions. For the BPP.25 and BPP.5 classes, there is also an increase in the number of optimal solutions.

The proposed adaptive strategy for sorting free items in the 1D-BPP demonstrates promising results in enhancing solution diversity and optimality. By reducing the occurrence of repeated fitness after the mutation process, this approach helps strike a balance in selective pressure, allowing for greater diversity within the population and effectively mitigating the risk of premature convergence. The increase in optimal solutions, particularly in instances involving larger items, highlights the effectiveness of incorporating random and conditional sorting strategies. Overall, the adaptive sorting approach not only achieves a higher proportion of optimal solutions but also strengthens the robustness of the Genetic Algorithm by expanding its search capabilities and potential for generalization.

6.1. Limitations of the Adaptive Strategies for the Mutation Operator

The implementation of the control strategies showed an improvement in the algorithm's performance in terms of the number of optimal solutions obtained and the reduction in individuals with repeated fitness, suggesting an increase in diversity. However, the proposed approach has some limitations that should be considered.

First, this study focused exclusively on the 1D-BPP, using a single benchmark with specific features, such as the condition that the optimal solution requires all bins to be

full. This could affect the generalization of the method to other variants of the problem or packing problems with more flexible constraints.

Second, diversity was estimated only by the number of individuals with repeated fitness. Although this indicator provides relevant information on the convergence of the population, other metrics could be considered in future studies to obtain a more complete picture of the algorithm's behavior.

Finally, the study was limited to modifying two processes within the mutation operator: (1) the ordering of genes in the solutions before mutation and (2) the reinsertion of the remaining items to complete the solution. Other variations in the mutation strategy were not explored, which could open new opportunities for improvement in future research.

Table 5. Number of optima and average of individuals with repeated fitness obtained by the GGA-CGT and GGA-CGT with sorted control. The table highlights in bold the case where the highest number of optimal solutions was achieved.

Class	Bin Capacity	GGA-CGT		GGA-CGT with Adaptive Strategies	
		Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation	Optimal Solutions	Average Number of Individuals with Repeated Fitness After Mutation
BPP.25	100	100	1.26	100	1.26
	1000	100	0.11	100	0.07
	10,000	100	0.04	100	0.04
	100,000	100	0.49	100	0.38
	1,000,000	100	1.08	100	1.00
	10,000,000	30	2.11	37	2.55
	100,000,000	0	0.11	0	0.07
	total	530	0.60	**537**	0.62
BPP.5	100	100	0.08	100	0.88
	1000	100	0.11	100	0.13
	10,000	99	0.3	99	0.29
	100,000	99	0.96	98	1.06
	1,000,000	2	1.64	5	1.98
	10,000,000	3	0.24	8	0.19
	100,000,000	32	0.28	27	0.36
	total	435	0.52	**437**	0.84
BPP.75	100	100	0.15	100	1.58
	1000	100	1.58	100	0.24
	10,000	95	5.86	94	0.97
	100,000	20	65.93	33	3.53
	1,000,000	58	68.46	77	2.05
	10,000,000	78	70.67	98	0.43
	100,000,000	78	70.33	100	0.52
	total	529	40.28	**602**	1.33
BPP1	100	100	0.42	100	1.42
	1000	100	1.99	100	0.54
	10,000	58	35.13	65	3.22
	100,000	74	55.75	86	2.07
	1,000,000	94	53.24	100	0.46
	10,000,000	97	55.58	100	0.28
	100,000,000	96	54.78	100	0.20
	total	619	36.13	**651**	1.17
Total		2113	19.38	**2227**	0.99

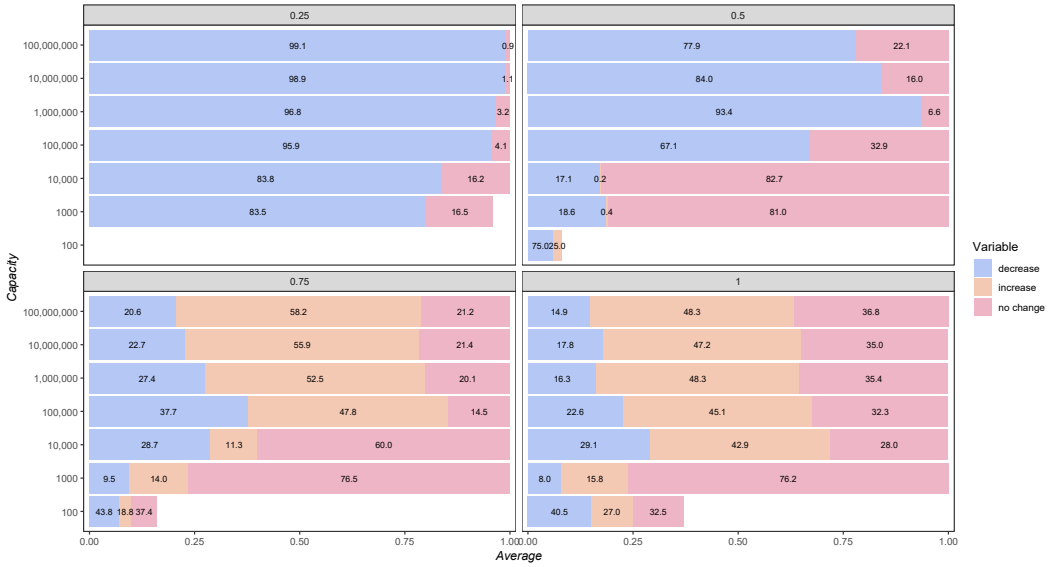

Figure 8. Proportion of generations in which the number of repeated fitness individuals increases, decreases, or remains unchanged after the mutation process during the execution of the GGA-CGT with adaptive strategies.

6.2. Statistical Test

Additionally, we conducted a statistical test to validate our results, comparing the original version of the GGA-CGT with the FI-GLX-1 operator and the GGA-CGT version with the proposed adaptive strategy for online mutation control. The statistical test applied is the Wilcoxon Rank Sum test, a non-parametric test. This test was performed on a sample of 31 runs with different seeds. For each run, we calculated the error as the relative difference, defined by the formula $(y - x)/x$, where x represents the number of bins in the optimal solution, and y represents the number of bins obtained by the algorithm. The average error for each instance was then calculated across the 31 runs. The statistical test was conducted by class and for the entire test set.

Table 6. Statistics for comparing the GGA-CGT and the GGA-CGT that includes sorted control in mutation. p-Values are presented by class and for the entire test suite, together with the average, standard deviation, and maximum value of optima obtained for each class.

Class	GGA-CGT			GGA-CGT with Adaptive Strategies			p-Value
	Average	Stdev	Max	Average	Stdev	Max	
BPP.25	525.87	4.56	536	537.39	4.43	548	9.77×10^{-1}
BPP.5	431.55	3.64	439	437.81	4.32	447	7.43×10^{-1}
BPP.75	516.97	7.16	530	599.58	4.25	608	9.54×10^{-12}
BPP1	618.23	3.81	629	652.77	2.89	659	4.32×10^{-5}
BPPv_{u}_c	2092.61	11.25	2115	2227.55	8.42	2246	7.58×10^{-7}

The results shown in Table 6 indicate that the adaptive mutation control in the GGA-CGT significantly improves performance compared with the original version of the GGA-CGT across various classes of the 1D-BPP. The adaptive strategy consistently achieves higher averages, standard deviations, and maximum values across 31 independent runs with different seeds.

For instance, in the BPP.75 class, the average number of optimal solutions increased from 516.97 in the original version to 599.58 with adaptive control, with a p-value of 9.54×10^{-12}. Similarly, in the BPP1 class, the average increased from 618.23 to 652.77,

with a p-value of 4.32×10^{-5}. The overall improvement across all classes (BPPv_u_c) is also evident, with an average increase from 2092.61 to 2227.55, supported by a highly significant p-value of 7.58×10^{-7}.

These statistically significant differences (p-values < 0.05) indicate that the adaptive mutation control mechanism enhances the diversity and convergence of the algorithm, achieving a higher number of optimal solutions in independent runs and effectively leveraging variations introduced by different seeds. This improvement highlights the potential of adaptive mutation control to enhance the quality and robustness of solutions in evolutionary algorithms applied to complex optimization problems.

7. Conclusions and Future Work

This work presented an experimental analysis of strategies integrated into the mutation operator, resulting in the development of online control strategies for mutation in the GGA-CGT. The findings demonstrated that adaptive mutation significantly improved the algorithm's performance. Additionally, the results suggested that the proposed approach promoted greater population diversity, mitigated premature convergence, and enabled the algorithm to explore a broader solution space.

Furthermore, the methods and analyses presented in this study can be applied to other combinatorial and real-world problems, which could lead to a deeper understanding of algorithmic performance and, potentially, to significant improvements in Genetic Algorithms for solving them. This study could be replicated and adapted to analyze other optimization problems and techniques, providing a broader understanding of the application of adaptive strategies in evolutionary algorithms.

The proposed strategy could be extended to real applications such as warehouse optimization, logistics planning, and industrial scheduling. However, since each problem has its characteristics, a specific study would be necessary to adapt the methodology and define an appropriate diversity metric. For example, the machines are predefined in the Parallel-Machine Scheduling Problem, and the process times are variable. In this case, the machines can be considered analogous to bins in the 1D-BPP, while the process times correspond to the bin's capacity. An adjustment of the adaptive approach and a redefinition of the diversity criteria would be required for its correct implementation.

Also, adapting the proposed mechanism to continuous optimization problems, such as economic dispatch and optimal energy flow, would demand modifications in the mutation operator to effectively manage the continuous variables. It would also be necessary to review the strategies within the mutation operator that could be controlled by the proposed method.

On the other hand, other combinatorial problems, such as the vehicle routing problem and flow-shop scheduling, could benefit from adaptive mutation to improve exploration and avoid premature convergence. Future research should explore integrating adaptive mechanisms into other genetic operators, enabling a dynamic balance between exploration and exploitation. In addition, extending the adaptive approach to control parameter settings between operators could further strengthen the robustness of the GGA-CGT, offering new perspectives for improving evolutionary algorithms in solving complex optimization problems.

Author Contributions: Conceptualization, S.A.-L., O.R.-F. and M.Q.-C.; methodology, S.A.-L. and M.Q.-C.; software, S.A.-L. and M.Q.-C.; validation, S.A.-L., O.R.-F., and M.Q.-C.; formal analysis, M.Q.-C.; investigation, S.A.-L. and M.Q.-C.; resources, S.A.-L.; writing—original draft preparation, S.A.-L. and O.R.-F.; writing—review and editing, S.A.-L., O.R.-F., and M.Q.-C.; visualization, S.A.-L., O.R.-F. and M.Q.-C.; supervision, O.R.-F. and M.Q.-C.; project administration, M.Q.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data supporting the reported results in this study are available upon request. Interested researchers may contact Stephanie Amador-Larrea at stephanieamadorlarrea@gmail.com and Marcela Quiroz-Castellanos at maquiroz@uv.mx to obtain access to the data.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*; W. H. Freeman & Co.: New York, NY, USA, 1979.
2. Martello, S. *Knapsack Problems: Algorithms and Computer Implementations*; J. Wiley & Sons: Chichester, NY, USA, 1990.
3. Ramos-Figueroa, O.; Quiroz-Castellanos, M.; Mezura-Montes, E.; Schütze, O. Metaheuristics to solve grouping problems: A review and a case study. *Swarm Evol. Comput.* **2020**, *53*, 100643. [CrossRef]
4. González-San-Martín, J.; Cruz-Reyes, L.; Gómez-Santillán, C.; Fraire, H.; Rangel-Valdez, N.; Dorronsoro, B.; Quiroz-Castellanos, M. Comparative Study of Heuristics for the One-Dimensional Bin Packing Problem. In *Hybrid Intelligent Systems Based on Extensions of Fuzzy Logic, Neural Networks and Metaheuristics*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 293–305.
5. Ramos-Figueroa, O.; Quiroz-Castellanos, M.; Mezura-Montes, E.; Kharel, R. Variation Operators for Grouping Genetic Algorithms: A Review. *Swarm Evol. Comput.* **2021**, *60*, 100796. [CrossRef]
6. Quiroz-Castellanos, M.; Cruz-Reyes, L.; Torres-Jimenez, J.; Gómez, S.C.; Huacuja, H.J.F.; Alvim, A.C. A Grouping Genetic Algorithm with Controlled Gene Transmission for the Bin Packing Problem. *Comput. Oper. Res.* **2015**, *55*, 52–64. [CrossRef]
7. Carmona-Arroyo, G.; Vázquez-Aguirre, J.B.; Quiroz-Castellanos, M. *One-Dimensional Bin Packing Problem: An Experimental Study of Instances Difficulty and Algorithms Performance*; Springer: Cham, Germany, 2021; Volume 940.
8. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computing*, 2nd ed.; Springer Publishing Company, Incorporated: Berlin/Heidelberg, Germany, 2015.
9. Falkenauer, E. A hybrid grouping genetic algorithm for bin packing. *J. Heuristics* **1996**, *2*, 5–30. [CrossRef]
10. Falkenauer, E.; Delchambre, A. A genetic algorithm for bin packing and line balancing. In Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Los Alamitos, CA, USA, 19–23 May 1992; pp. 1186–1192. [CrossRef]
11. Cruz-Reyes, L.; Quiroz-Castellanos, M.; Alvim, A.C.F.; Huacuja, H.J.F.; Gómez, C.; Torres-Jiménez, J. Heurísticas de agrupación híbridas eficientes para el problema de empacado de objetos en contenedores. *Comput. Sist.* **2012**, *16*, 349–360.
12. González-San-Martín, J.E. Un Estudio Formal de Heurísticas Para el Problema de Empacado de Objetos de una Dimensión. Master's Thesis, Instituto Tecnológico de Ciudad Madero, Ciudad Madero, Mexico, 2021.
13. Amador-Larrea, S. Un Estudio de Operadores Genéticos de Cruza para el Problema de Empacado de Objetos en Contenedores. Master's Thesis, Universidad Veracruzana, Veracruz, México, 2022.
14. Eiben, A.; Hinterding, R.; Michalewicz, Z. Parameter control in evolutionary algorithms. *IEEE Trans. Evol. Comput.* **1999**, *3*, 124–141. [CrossRef]
15. Karafotias, G.; Hoogendoorn, M.; Eiben, A.E. Parameter Control in Evolutionary Algorithms: Trends and Challenges. *IEEE Trans. Evol. Comput.* **2015**, *19*, 167–187. [CrossRef]
16. Bhatia, A.; Basu, S.K. Packing bins using multi-chromosomal genetic representation and better-fit heuristic. In *Proceedings of the International Conference on Neural Information Processing*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 181–186.
17. Singh, A.; Gupta, A.K. Two heuristics for the one-dimensional bin-packing problem. *OR Spectr.* **2007**, *29*, 765–781. [CrossRef]
18. Agustín-Blas, L.E.; Salcedo-Sanz, S.; Ortiz-García, E.G.; Portilla-Figueras, A.; Pérez-Bellido, Á.M.; Jiménez-Fernández, S. Team formation based on group technology: A hybrid grouping genetic algorithm approach. *Comput. Oper. Res.* **2011**, *38*, 484–495. [CrossRef]
19. Chaurasia, S.N.; Singh, A. A hybrid evolutionary approach to the registration area planning problem. *Appl. Intell.* **2014**, *41*, 1127–1149. [CrossRef]
20. Jawahar, N.; Subhaa, R. An adjustable grouping genetic algorithm for the design of cellular manufacturing system integrating structural and operational parameters. *J. Manuf. Syst.* **2017**, *44*, 115–142. [CrossRef]
21. Rossi, A.; Singh, A.; Sevaux, M. A metaheuristic for the fixed job scheduling problem under spread time constraints. *Comput. Oper. Res.* **2010**, *37*, 1045–1054. [CrossRef]
22. Peddi, S.; Singh, A. Grouping Genetic Algorithm for Data Clustering. In *Proceedings of the Swarm, Evolutionary, and Memetic Computing*; Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 225–232.

23. Balasch-Masoliver, J.; Muntés-Mulero, V.; Nin, J. Using genetic algorithms for attribute grouping in multivariate microaggregation. *Intell. Data Anal.* **2014**, *18*, 819–836. [CrossRef]
24. Ramos-Figueroa, O.; Quiroz-Castellanos, M. An experimental approach to designing grouping genetic algorithms. *Swarm Evol. Comput.* **2024**, *86*, 101490. [CrossRef]
25. Ramos-Figueroa, O.; Quiroz-Castellanos, M.; Mezura-Montes, E.; Cruz-Ramírez, N. An Experimental Study of Grouping Mutation Operators for the Unrelated Parallel-Machine Scheduling Problem. *Math. Comput. Appl.* **2023**, *28*, 6. [CrossRef]
26. Fernández-Solano, O. Un Estudio Experimental de Operadores de Mutación para el Algoritmo Genético de Agrupación Para la Segmentación de Imágenes en RGB. Master's Thesis, Universidad Veracruzana, Veracruz, México, 2023.
27. Zavaleta-García, N.A. Estudio Experimental de Operadores Genéticos de Cruza Para el Problema de Segmentación de Imágenes en Color. Master's Thesis, Universidad Veracruzana, Veracruz, México, 2023.
28. Yorgancılar, G. Effects of Evolutionary Operators in Grouping Genetic Algorithms on Diversity and Result Quality. Master's Thesis, TED University, Ankara, Turkey, 2020.
29. Perez, A.; Michelt, J. Optimización Generalizada para Problemas de Agrupación: Un Enfoque Coevolutivo Auto-Adaptativo. 2024. Available online: https://rinacional.tecnm.mx/jspui/handle/TecNM/7945 (accessed on 12 December 2024).
30. Carmona-Arroyo, G. A Grouping Genetic Algorithm for Variable Decomposition in Large-Scale Constrained Optimization Problems. Master's Thesis, Universidad Veracruzana, Veracruz, Mexico, 2024.
31. Ramos-Figueroa, O.; Quiroz-Castellanos, M.; Carmona-Arroyo, G.; Vázquez, B.; Kharel, R. Parallel-Machine Scheduling Problem: An Experimental Study of Instances Difficulty and Algorithms Performance. In *Recent Advances of Hybrid Intelligent Systems Based on Soft Computing*; Melin, P., Castillo, O., Kacprzyk, J., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 13–49. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.