Inter IIT Tech Meet 13.0

Optimal Cargo Management for Flights.

Endterm Report

Team 64

2024-12-07

Contents

1 Introduction

The optimization of logistics and freight handling is critical for improving global supply chain efficiency. This project focuses on maximizing the utilization of Unit Load Devices (ULDs) while minimizing costs and meeting operational constraints like package priorities. To address this, we employed genetic algorithms as the foundational methodology, enhancing them with customizations tailored to the complexities of ULD packing. Efficient packing is essential for reducing delays, optimizing costs, and ensuring reliable service. Beyond immediate operational benefits, the project aligns with FedEx's commitment to innovation and customer satisfaction. This report details our objectives, methodology, results, challenges, and potential future enhancements, offering a comprehensive evaluation of the solution and its impact on logistics operations.

2 Objectives and Problem Statement

2.1 Objective Overview

The problem statement aims to address critical logistical challenges in FedEx's operations by developing a robust solution for efficient packing and allocation of packages into Unit Load Devices (ULDs). The objectives of the project are as follows.

- **Optimize ULD Space Utilization**: Ensure that the maximum volume and weight capacity of each ULD are used effectively while adhering to the constraints provided.
- **Minimize Operational Costs**: Reduce costs associated with delays, inefficient packing, and the spreading of priority packages across multiple ULDs.
- **Ensure Compliance with Constraints**: Maintain strict adherence to operational constraints, such as package weight limits, dimensions, and the priority categorization of packages.
- **Achieve Scalability**: Design a solution that performs effectively with varying package datasets and accommodates future increases in package volume and diversity.
- **Deliver Practical and Implementable Results**: Create a system that provides real-time, actionable outputs for FedEx operations, including ULD assignments, spatial coordinates of packages, and cost metrics.

2.2 Problem Statement

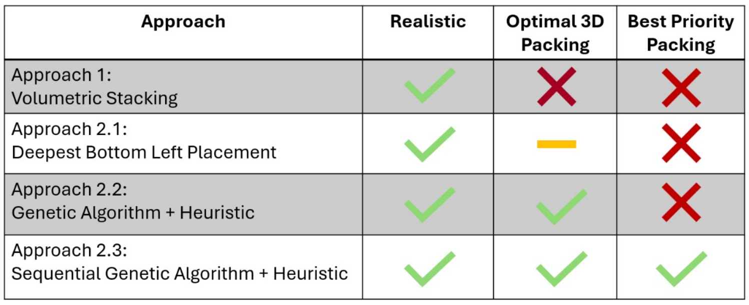
The Problem statement aims to address critical logistical challenges in FedEx operations by developing a robust solution for efficient packing and allocation of packages into Unit Load Devices (ULDs). The objectives of the project are as follows.

- **ULD Constraints**: Each ULD has predefined maximum dimensions, weight capacity, and a limited number of packages it can accommodate. All selected packages must be fully contained within their respective ULDs.
- **Package Constraints**: Packages differ in dimensions, weight, and priority status. Priority packages should be allocated with minimal cost incurred from spreading them across multiple ULDs.
- **Cost Optimization**: The total cost of the solution includes delay costs for unallocated packages and a penalty for spreading priority packages across ULDs. The goal is to minimize these costs without compromising operational efficiency.
- **Scalability and Flexibility**: The solution must be adaptable to handle diverse datasets, ranging from small-scale operations to large volumes typical of peak logistical seasons.

- **Implementation Feasibility**: The system should produce actionable outputs that seamlessly integrate into FedEx's existing workflows with minimal computational overhead.

By addressing these objectives and challenges, the project aims to deliver a solution that enhances operational efficiency, reduces costs, and aligns with FedEx's strategic priorities for logistics optimization.

3 Methodology



Approach	Realistic	Optimal 3D Packing	Best Priority Packing
Approach 1: Volumetric Stacking	✓	✗	✗
Approach 2.1: Deepest Bottom Left Placement	✓	–	✗
Approach 2.2: Genetic Algorithm + Heuristic	✓	✓	✗
Approach 2.3: Sequential Genetic Algorithm + Heuristic	✓	✓	✓

Figure 1: Comparison of Approaches used

The problem statement concerns packing boxes within bins (containers) while satisfying certain constraints. In doing some research, we found some similarities between the problem statement, the 0/1 Knapsack problem, and the 3D bin packing problem.

3.1 Approach 1: Optimal Stacking

Our first approach involved making volumetrically optimal stacks of containers that fit within the height of the ULDs and placing them sequentially (lengthwise, then widthwise) in the ULD. In this approach, we ensure that all stacks make the maximum use of available volume above the box placed at the base while following the weight limit of the ULDs. The drawback of this approach is that it provides a placement optimized in only one dimension (height). This resulted in gaps in the placement. Also, the algorithm for finding the volumetrically optimal stack is. Hence, we discarded this approach.

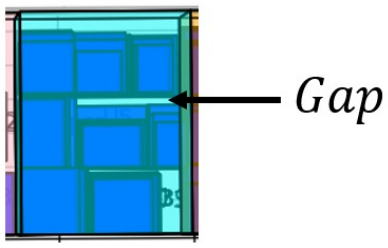


Figure 2: Gaps in ULDs in approach 1

3.2 Approach 2

3.2.1 Deepest-Bottom-Left Placement

This approach is based on how a person naturally fills a container with boxes. All the boxes are ordered in the following way: First, priority packages are ordered in decreasing order of 'Volume,' and then economy packages are ordered in decreasing order of 'Cost of Delay.' We place these boxes in this sequence and by following the heuristic of putting the next box at the deepest-bottom-left position, i.e., for each package, out of all positions of placement in ULDs, choose the valid position that has the minimum x coordinate, the minimum z coordinate, and the minimum y coordinate. This mimics the natural packing process where we place packages in the deepest (minimum x), bottommost (minimum z), and leftmost (minimum y) valid positions, which are easily accessible. While checking the validity of a position based on the constraints and stability

The drawback of this approach is that sequentially placing the packages in the decreasing order of 'Volume' or 'Cost of Delay' does not guarantee the optimal or even near-optimal solution. However, this heuristic ensures that the packages can be placed realistically. Hence, we decided to find a way to get to the optimal sequence of the packages with the least cost.

3.2.2 Genetic Algorithm + Deepest-Bottom-Left placement

We decided to use a genetic algorithm to find the near-optimal sequence of packages and their respective orientations, which, when placed according to the Deepest-Bottom-Left heuristic mentioned in the above approach, will give the minimum cost. Inspired by the principles of natural evolution, genetic algorithms are based on concepts such as crossovers, mutations, generations, and survival of the fittest. The algorithm draws parallels between the sequence of packages and genes.

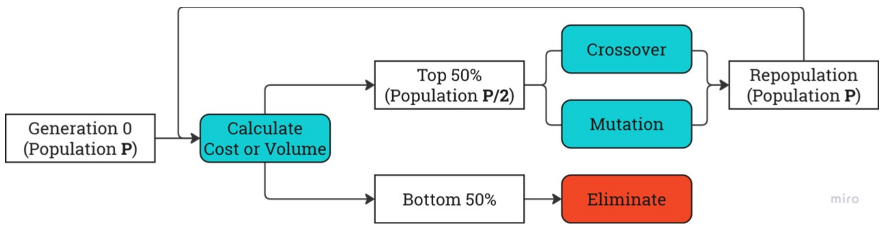


Figure 3: Functioning of Genetic Algorithm

The algorithm starts with a population of solutions, i.e., a set consisting of several sub-optimal ordering of packages. The individual can be represented by a genome with two chromosomes; one chromosome is permutation. This is generation 0. Then, it keeps forming new generations by:

- Sorting the ordering of packages based on cost and eliminating the bottom half of the population.
- Select two random orderings of packages (individuals) from the remaining population and breed* them (explained below) to regenerate half of the population.
- Allowing a small probability of mutation in the generated individuals. This process can then be repeated with the new generation.

Breeding: Out of the best 50% population left after elimination, we randomly pick individual solutions and perform genetic operations, which create new individual solutions with slightly modified sequencing of packages. We perform two genetic operations: Crossovers* and Mutations*. One is a crossover, and the other is a mutation, which introduces random changes in the genes of the individuals. There are two types of mutations: one affects the permutation chromosome, and the other affects the orientation chromosome.

Crossovers: Randomly pick pairs amongst the best 50% of the population and produce two children solutions by mixing the genes (i.e., exchanging a small sub-sequence of the individual solutions) of the two parents.

Mutations: Randomly pick an individual solution and generate a children's solution with random changes in the individual solutions. There are two types of mutations: one affects the permutation chromosome, and the other affects the orientation chromosome.

With every generation, the package sequence gets closer to the most optimal sequence. After several generations, we choose the best sequence and find the coordinates of all the packages and the ULD they are placed in by following the deepest-bottom-left heuristic.

We found that the solution we got from this approach surpasses all the other approaches till now. For the challenge data, it could fit the priority packages into four packages and optimize the remaining economy packages so that we get a cost of **37665**.

The drawback of this approach is that the placement of priority to the minimum number of ULDs is less likely. To overcome this, we modified this approach.

3.2.3 Final Method
Sequential Genetic Algorithm + Deepest-Bottom-Left placement

(A better version of Approach 2.2)

To make sure the priority packages are packed in the minimum number of ULDs, we apply the Genetic Algorithm twice: first on the sequence of priority packages only and then on the entire sequence of packages.

- **Genetic Algorithm on Priority Packages**: We calculate the number of ULDs that must be required by priority packaging by checking the weights and volumes of the priority packages and comparing them with the weight limits and volume of the ULDs. Then, we use the deepest bottom-left placement heuristic to place the priority packages in these ULDs and calculate the excess volume of priority packages that cannot be placed in the ULD. We use the genetic algorithm to reduce this excess volume.
- **Genetic Algorithm on Entire Sequence**: After we complete the above step, we get a sequence of priority packages that are tightly packed. To this sequence, we add the economic packages in decreasing order of 'Cost of Delay.' Again, we use the genetic algorithm in this new sequence to reduce the net cost. This step mainly optimizes the economy packages to minimize the net Cost.

We found that the solution we got from this approach surpasses all the other approaches by far and has the minimum Cost. For the challenge data, it could fit the **priority packages in 3 ULDs** and optimize the remaining economy packages so that we get a cost of 32944 Hence, we finalized this approach.

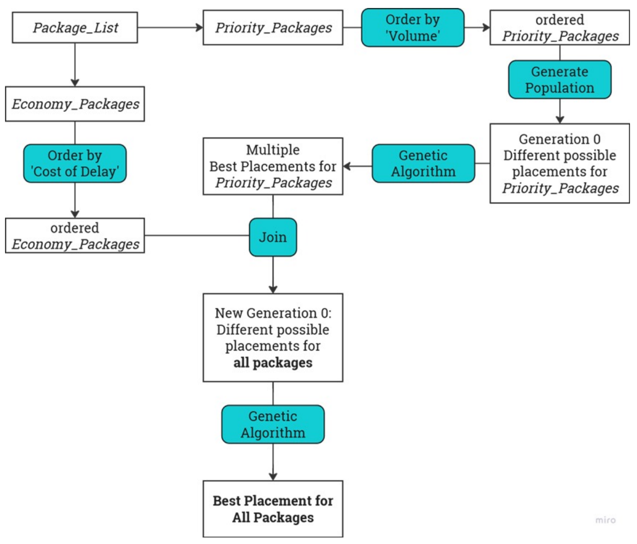

Package_List
Priority_Packages
Order by 'Volume'
ordered Priority_Packages
Generate Population
Economy_Packages
Order by 'Cost of Delay'
Multiple Best Placements for Priority_Packages
Genetic Algorithm
Generation 0 Different possible placements for Priority_Packages
ordered Economy_Packages
Join
New Generation 0: Different possible placements for all packages
Genetic Algorithm
Best Placement for All Packages
miro

Figure 4: Methodology Used

4 Solution Design and Implementation

4.1 PreProcessing

Pre-processing involves reading and parsing the data from the given structured text file containing information about ULDs (Unit Load Devices) and packages. First, the file is split into sections based on keywords to distinguish ULD attributes from package attributes. We are using regular expressions to find valid entries and extract them into dictionaries. We extracted dimensions and weight limits for ULDs and dimensions, weight, type, and delay cost for packages.

After extracting the data, the volume of each ULD is computed as the product of its dimensions (length, width, height), and the ULDs are ordered in descending order of volume. This ensures that the ULD with more volume is filled first. Similarly, the total volume of priority packages is calculated for further use.

4.2 Deepest-Bottom-Left Placement

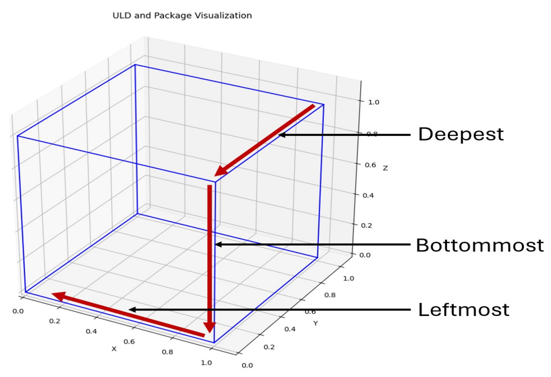


Figure 5: ULD and Package visualisation

Every position inside ULDs is denoted by a tuple of four integers (x, y, z, uld_id). By convention, the position of a package is given by the coordinates of the reference corner (i.e., the corner with minimum x, y, and z values). We maintain a list called '*poset*' of valid positions sorted by uld_id, y, z, x in sequential order. This corresponds to ordering the position in the deepest(y)-bottom(z)-left(x) fashion. For every orientation of a package, we traverse through this list and place the package at the first valid position.
A valid position is one where:

- (x,y,z) + (length, width, height) is entirely inside the ULD
- There is no overlap with packages that are already placed
- The weight of existing packages + the weight of the package to be placed is within the weight limit of the ULD.
- The package has complete base support, i.e., either it is placed on the ground (z=0), or there exist other packages that have been placed on which this package can be placed entirely, i.e., no part of the base is dangling/exposed.

If a box is not placeable in any of these positions, we add it to the set of unplaced boxes, assisting in the cost calculation. Otherwise, when we find the first valid orientation and position in 'poset', we place the package there and remove this position from the list. We then add three more positions (x+l, y, z, ULD), (x, y+w, z, ULD), and (x, y, z+h, ULD), where the l, w, and h are based on the orientation of the box being placed.

Now, following the heuristic, a solution can be represented by a sequence of boxes and their respective orientations. For this, we have the list of all boxes where every solution is a 2-dimensional array of dimension 2 x no. of boxes. The first value in the array represents the package ID of the package, and the second represents the orientation of the respective box. The orientation is represented by an integer from 0 to 5, with the integers representing (L, W, H), (W, L, H), (W, H, L), (H, W, L), (H, L, W), (L, H, W) respectively (where L, W, H are the lengths, widths, and heights of the boxes as given in the data).

4.3 Introducing Genetic Algorithms

Genetic Algorithms (GA) are applied to minimize cost using two steps:

1. Fitting priority packages into as few ULDs as possible.
2. Minimizing the cost of economy packages that are not packed.

Each solution is represented by two chromosomes:

- **Chromosome 1:** Sequence of boxes.
- **Chromosome 2:** Orientation of each box.

4.3.1 Parent Solutions:

- **P1:** Chromosome 1: [0, 1, 2, 3, 4, 5, 6, 7], Chromosome 2: [2, 4, 3, 3, 5, 0, 1, 1].
- **P2:** Chromosome 1: [7, 5, 6, 3, 4, 2, 0, 1], Chromosome 2: [1, 1, 1, 3, 0, 0, 0, 0].

4.3.2 Crossover (Modified OX)

- Select random cutting points (e.g., $i = 2$ and $j = 5$).
- Copy the segment between cutting points:
  - From P1 to C1: [2, 3, 4, 5].
  - From P2 to C2: [6, 3, 4, 2].
- Fill remaining positions from the other parent to avoid duplicates:
  - C1: [7, 6, 2, 3, 4, 5, 0, 1].
  - C2: [0, 1, 6, 3, 4, 2, 5, 7].

4.3.3 Mutation

Step 1: 2-OPT Mutation Invert a substring within Chromosome 1:

- Example for C1: [7, 6, 2, 3, 4, 5, 0, 1] $\rightarrow$ [7, 6, 4, 3, 2, 5, 0, 1].

Step 2: Random Flip Mutation Flip rotation values in Chromosome 2 with small probability:

- Example for C1: [1, 1, 3, 3, 5, 0, 0, 0] $\rightarrow$ [1, 0, 3, 3, 0, 0, 0, 0].

4.3.4 Final Offspring

- **C1:** Chromosome 1: [7, 6, 4, 3, 2, 5, 0, 1], Chromosome 2: [1, 0, 3, 3, 0, 0, 0, 0].
- **C2:** Chromosome 1: [0, 1, 6, 5, 2, 4, 3, 7], Chromosome 2: [2, 4, 0, 3, 0, 5, 1, 1].

4.4 Steps for Optimization

Step 1: Fit Priority Packages Priority packages are ordered by volume and packed using a Genetic Algorithm. Using the deepest-bottom-left heuristic, solutions are generated, converging in around 80 generations.

Step 2: Fit Economy Packages The best solutions from Step 1 are used to pack economy packages sorted by their 'Cost of Delay.' The algorithm runs for 100 generations to minimize costs.

4.5 Choosing Parameters

- **Population Size:** Set to 100 based on package count (around 500).
- **Mutation Probabilities:** Kept low for faster runtime while maintaining solution quality.

The algorithm runs for 10-12 minutes to achieve convergence.

5 Results And Analysis

The data demonstrates our successful attempt to prioritize the placement of items into three Unit Load Devices (ULDs), which allowed for optimized resource allocation and cost reduction. By strategically arranging the items across these ULDs, we maximized space utilization while adhering to weight and volume constraints. We successfully compressed the priority boxes into three ULDs.
The optimization process led to a significant cost reduction, bringing the total cost down to **32,936**. This improvement can be attributed to the balanced distribution of high-priority items within the designated ULDs, minimizing excess transportation costs and ensuring efficient operations.
Such a reduction in cost showcases the value of leveraging advanced allocation techniques to enhance logistics planning and achieve economic benefits.

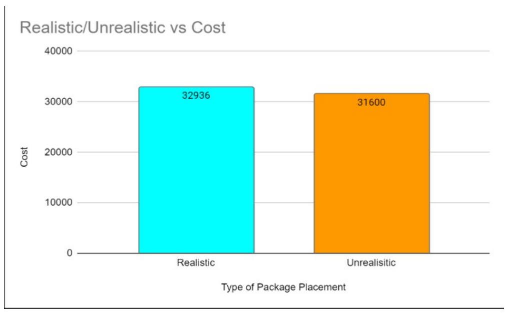


Figure 6: Comparison between realistic and unrealistic placements

The provided bar chart compares the costs associated with realistic and unrealistic package placement strategies. The results demonstrate that even when adopting a more practical and realistic approach, the cost (32,936) is only marginally higher than the cost of the theoretically optimal but less realistic approach (31,600). This slight difference underscores the efficiency of the realistic solution, highlighting its viability in achieving near-optimal results while maintaining practical feasibility. Such a balance makes the realistic approach a robust and applicable choice for real-world implementations.

ULD ID	**Volumetric Efficiency (%)**
ULD1	78.86
ULD2	71.19
ULD3	68.94
ULD4	64.21
ULD5	73.92
ULD6	72.31
Mean	**71.57**

Table 1: Volumetric Efficiency comparison in ULDs

ULD1 has the highest efficiency at **78.86%,** indicating it uses its space most effectively compared to the other ULDs. ULD2 follows with an efficiency of **71.19%,** which is decent but still leaves room for improvement. ULD3 and ULD4 are less efficient, with efficiencies of **68.94%** and **64.21%,** respectively, suggesting that these ULDs are not utilizing their space as well, potentially due to wasted space or poor package placement. ULD5 and ULD6 show similar performance with efficiencies of **73.92%** and **72.31%,** respectively, demonstrating better space utilization than ULD3 and ULD4 but still falling short compared to ULD1.

The mean volumetric efficiency of 71.57% is a good average for the set of ULDs, indicating that, on the whole, the ULDs are using space reasonably well, but there's room for further optimization in some cases, especially for ULD3 and ULD4.

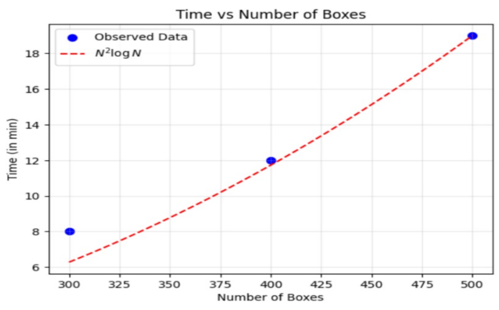


Figure 7: Time vs Number of Boxes

The graph compares observed runtime (blue dots) with the theoretical **P*G*N^2 log N,** Where P is the population size, G is the number of generations, and N is the size of the genome (number of boxes in the sequence), and complexity (red dashed line) as the number of boxes increases. The observed runtime closely aligns with the theoretical trend, indicating that the algorithm's scaling behavior is consistent with its expected complexity, driven by quadratic growth from pairwise interactions and a logarithmic factor likely due to sorting or optimizations.

6 Challenges And Limitations of solution

6.1 Challenges Encountered

- **Practical Feasibility in ULD Packing**

  The challenge was to design a packing solution that maximizes space utilization while ensuring practical loading. Conventional algorithms often leave voids in inaccessible areas, making theoretical solutions impractical for real-world implementation. These configurations, while optimal in theory, are impractical to execute as they cannot be realistically loaded by machines or humans.

  To address this, we implemented a height-wise heuristic that prioritizes the vertical arrangement of boxes during packing. This heuristic ensures that boxes are organized in a sequence that aligns with practical loading procedures, optimizing height utilization first. The generated sequence allows for the systematic filling of ULDs, starting from the base and progressing upward in a manner that is operationally feasible.

This approach ensures:

- **Realistic Access**: Each box can be reached and positioned effectively during loading, avoiding voids in inaccessible areas.

- **Operational Efficiency**: The sequence/order generated by the heuristic reduces the need for re-arrangements or adjustments during the packing process. - **Cost Minimization with Practicality**: While maintaining cost efficiency, the solution guarantees a realistic and implementable packing configuration.

By incorporating this height-wise optimization strategy, we have developed a solution that balances theoretical efficiency with practical feasibility, making it uniquely tailored for real-world ULD loading operations.

- **Optimization of Priority Package Allocation:**

  A key challenge was ensuring that all priority packages were accommodated in the minimum possible number of ULDs. While no strict upper limit was specified, minimizing the number was critical for cost efficiency. To address this, we employed a genetic algorithm with a population size of 100, optimizing fitness based on the volume of packages outside the designated ULDs. Convergence to an optimal solution (fitness value of zero) was achieved after approximately 70 generations, successfully packing all priority boxes. The subsequent challenge involved optimizing the remaining economy packages, where we explored two approaches.

  - Applied standard heuristics to allocate economy packages.

  - Extending the genetic algorithm to include economy packages, although this approach required additional computational resources.

- **Incorporation of Stability Constraints**

  Accounting for stability during loading was critical to ensure operational feasibility, particularly in dynamic transit conditions. However, this constraint significantly reduced packing density, as some configurations that maximized ULD utilization were deemed unstable.

  By including stability in our model, fewer boxes could be packed, leading to an estimated increase in costs of **1348 rupees**. This trade-off underscores the complexity of balancing practical constraints with optimization objectives.

6.2 Limitations of Current Approach

- **Time Complexity of Genetic Algorithms:**

  The computational demands of genetic algorithms presented a notable limitation, mainly when applied to larger datasets. The execution time scales quadratically with the number of boxes, posing a significant challenge for real-time scalability. Although our solution performed effectively for the current problem, addressing larger datasets within practical time constraints may require further optimization or the integration of hybrid approaches. Transitioning to a set-based approach could significantly improve performance by reducing the complexity from $O(N^2 \log N)$ to $O(N^2)$.

- **Operational Constraints Beyond the Model:**

  While stability and weight distribution were integrated, additional real-world factors such as variable handling procedures, package fragility,transit-specific conditions (e.g., vibrations or tilts), and individual package load-bearing capacity were not modeled explicitly. Including such constraints would improve the operational applicability of the solution but would require further computational and data collection efforts.

7 Proposed Innovation and Strategic Enhancements

7.1 Implemented Enhancements

- **Scalability and Package Categorization**

  To demonstrate scalability, we successfully integrated mock and extended datasets into our framework. Packages were systematically grouped into subcategories based on volume, facilitating efficient allocation and reducing computational complexity. This refinement not only enhanced operational clarity but also allowed for streamlined scaling to diverse package profiles.

- **Loading Operation Simulations**

  We designed a simulation to replicate real-world loading scenarios, accounting for constraints such as weight limits and spatial configurations. This approach validated the algorithm's robustness and demonstrated its adaptability to practical applications, reinforcing its reliability under varied conditions.

7.2 Conceptual Enhancements

- **Dynamic Pricing Strategy**

  We propose a dynamic pricing mechanism to optimize the utilization of previously underutilized ULD spaces. By offering discounted rates for packages that fit efficiently into such spaces, FedEx could leverage this as a marketing tactic. This strategy aligns operational efficiency with customer satisfaction, enhancing brand positioning.

Note: Due to limited project duration, we prioritized core deliverables over these conceptual enhancements. Nonetheless, these ideas are presented as potential avenues for FedEx to explore in future iterations, ensuring sustained innovation and value creation.

8 Conclusion And Recommendation

The project successfully optimized FedEx's cargo management system, reducing costs to **32936** and achieving an average volumetric efficiency of 71.57%, with ULD1 reaching 78.86%. The combination of sequential genetic algorithms and the Deepest-Bottom-Left heuristic delivered a practical and scalable solution, effectively balancing real-world feasibility with near-optimal performance.

Adopt hybrid or set-based approaches to improve scalability and reduce computational complexity, especially for large datasets. Introduce additional factors like package fragility, load-bearing limits, and transit-specific conditions to improve practical applicability. Implement dynamic pricing strategies to utilize under-filled ULDs more efficiently. Adjust the priority penalty constant (k) for better cost-efficiency trade-offs. Conduct controlled testing to ensure seamless integration with existing workflows and refine the approach under diverse logistical scenarios. By building on these recommendations, FedEx can further enhance efficiency, scalability, and operational reliability, ensuring cost savings and sustained logistical innovation.

References

[1] H. Wang and Yanjie Chen, "A hybrid genetic algorithm for 3D bin packing problems," 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), Changsha, China, 2010, pp. 703-707, doi: 10.1109/BICTA.2010.5645211.