Inter IIT Tech Meet 13.0
Technical Report

FedEx - Optimal Cargo Management for Flights

Team Code - 32

December 2024

§1 Introduction

Efficient cargo management is a critical challenge in the logistics and transportation industry, particularly for air shipments. The problem involves optimizing the packing of Unit Load Devices (ULDs)—standardized containers used to transport cargo on aircrafts—while adhering to constraints such as weight limits, package dimensions, and prioritization of high-value shipments. This optimization directly impacts operational costs, environmental sustainability, and service reliability.

§1.1 Problem Statement

The objective of the ULD optimization problem is to pack a given set of packages into a specified set of ULDs under the following constraints:

- The total weight of all packages in a ULD must not exceed its weight limit.
- All Priority Packages must be shipped, while some Economy Packages can be left behind if necessary.
- Packages assigned to a ULD must fit entirely within its dimensions.
- The solution should minimizes the total cost, which includes:
  (a) Costs associated with packages left behind.
  (b) Costs from spreading Priority Packages across **multiple** ULDs.

Packages are cuboidal and must be oriented along the axes of the ULDs. This orientation simplifies the representation of their positions using coordinates of two diagonally opposite corners. Priority packages should ideally be grouped into **as few ULDs** as possible to expedite unloading and delivery.

§1.2 Significance of the Problem

Optimizing ULD packing yields several benefits.

- **Operational Efficiency:** Maximizing space utilization within ULDs reduces the number of flights and associated costs such as fuel and labour.
- **Cost Reduction:** Efficient packing minimizes delays and the need for additional resources.
- **Environmental Sustainability:** Reducing the number of ULDs and flights lowers carbon emissions, aligning with FedEx's commitment to environmentally friendly logistics practices.

Our technical report outlines the approach taken to solve the problem, including a detailed explanation of the algorithms used, challenges encountered, and limitations of the proposed solution.

§2 Greedy Approach

§2.1 First Fit/Priority Based Two-Stage Approach

We initially tried the First Fit/Priority Based Two-Stage Approach is a packing strategy in which boxes are placed layer by layer and row by row within a single layer. This strategy involved a two-stage sorting procedure to optimize the placement of boxes in bins.

The Two-Stage "Crainic" Sorting method consists of the following two stages:

Stage 1: Grouping Boxes by Dimension

In the first stage, boxes are grouped based on one of their dimensions (such as width, height, or depth). The goal is to create groups of boxes that are similar in size along that dimension. This helps in reducing packing inefficiencies and ensures a better fit within the bin. The grouping process is defined as:

$$G_i = \{B_1, B_2, \ldots, B_n\} \quad \text{where each box } B_j \in G_i \text{ has a similar dimension along the selected axis.}$$

where G_i represents a group of boxes based on their dimension.

Stage 2: Prioritization of Box Placement

After the grouping is done, the second stage involves sorting the groups based on their placement effectiveness. The order is that the larger groups (groups with a larger common edge) are placed first, and the sequence within the group in which they should be placed is chosen according to the heuristics described below. The goal is to place the larger boxes first to improve the packing profile. Let the prioritization function $P(G)$ be defined as:

$$P(G_j) = \text{Priority of Group } G_j$$

Then, the groups are sorted in descending order of priority and processed:

$$\text{Sort}(G) = \{G_1, G_2, \ldots, G_n\} \quad \text{where } P(G_1) \geq P(G_2) \geq \cdots \geq P(G_n)$$

Heuristics Tried

- Inside a group determined by similarity based on one dimension, we keep the max(uncommon_dimensions) across the row to maximize the number of rows in a plane.
- Inside a group determined by similarity based on one dimension, we keep the max(uncommon_dimensions) across the plane, changing direction to maximize the number of planes in a ULD.

The second heuristic of maximizing the number of planes in ULD performed better, leading to a cost of around 37k for the given data set.

Packing Process:

After sorting and grouping, the First Fit heuristic is applied. The algorithm places each box into the first available space (bin) that can accommodate it, ensuring the efficient use of the available space without wasting room.

Post-Packing Adjustment:

Once the priority boxes have been inserted, the next step is to try to readjust the packing to minimize the number of ULDs used. The goal is to rearrange the boxes in such a way that the number of ULDs is minimized, thereby reducing the overall packing cost. This is achieved by re-evaluating the placement of priority packages and adjusting their positions to fit more efficiently.

Placing economy packages:

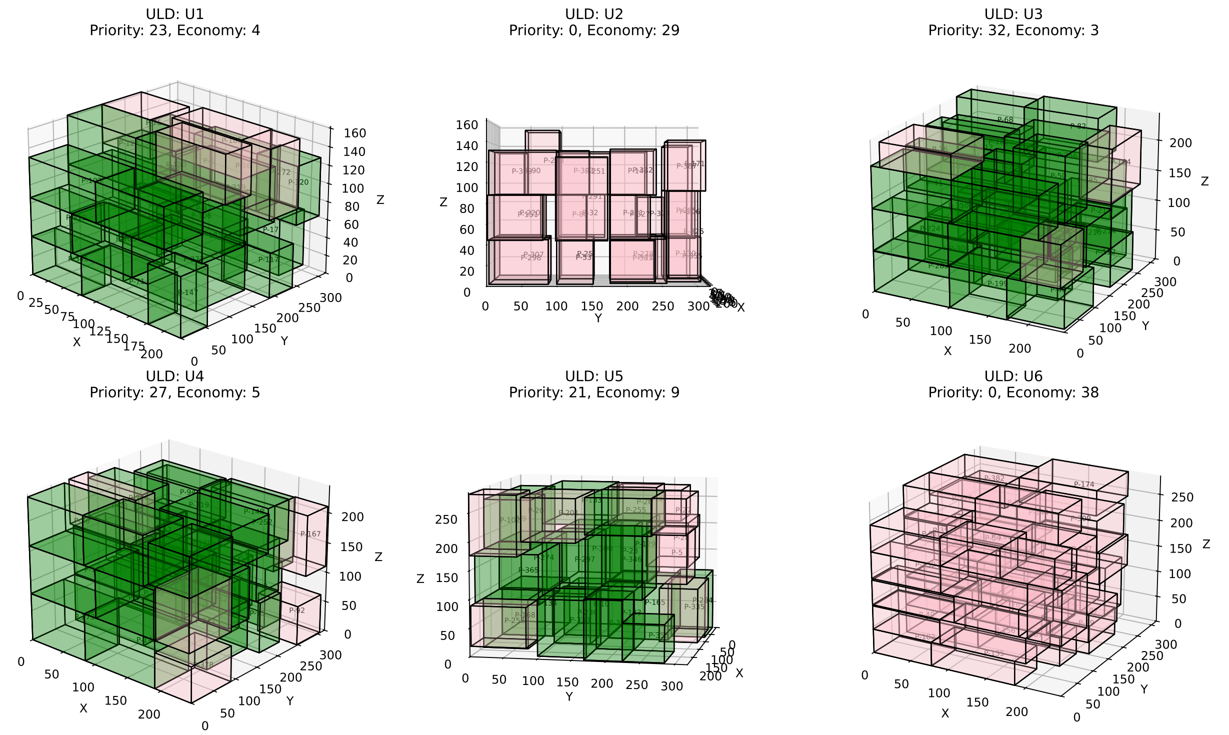
The last step focuses on achieving economic packing by minimizing the cost of any leftover packages. The arrangement is optimized to reduce unused space and ensure that as few packages as possible remain unplaced using ad-hoc addition.

Ad-hoc Addition

To fit any leftover economy packages, we first calculate all the corners of the existing packages in the ULD. If it is possible to fit some economy package in a given space, we can always coincide one corner of the package with the empty space's corner. So, we iterate over all the possible corners of the existing packages and check if a package can be placed with one corner coinciding with it in its 6 (rotational orientations) * 8 (quadrant orientations); after figuring out all the places we arbitrarily choose one of the corner and place our leftover economy package.

Throughout all of these steps, we take care not to exceed the maximum weight limit set by the ULDs, ensuring that the packing process is in accordance with weight restrictions.

Package placement visualization using the above algorithm - Please note that it's suboptimal



As we can see, the priority packages are distributed among 4 ULDs and the packing itself is very sensitive to the dimensions of the packages. To combat these problems, we adopt a local search technique, for which we use genetic algorithms.

We obtain a cost 37000-38000 using this approach which is suboptimal so we instead try a different approach as explained below.

§3 Genetic Algorithm

§3.1 Using Local Search Techniques and EMS Heuristics

A **Genetic Algorithm (GA)** is a meta-heuristic optimization technique inspired by the process of natural selection and evolution. It belongs to the class of evolutionary algorithms and is widely used for solving complex optimization problems that are challenging to address with traditional methods. By iteratively evolving a population of candidate solutions, a GA seeks to find optimal or near-optimal solutions over time.

§3.2 Key Components of our Genetic Algorithm

1. **Chromosome Representation**: A chromosome encodes a candidate solution to the problem. It is often represented as a string (binary, integer, or real-valued) that captures the essential decision variables of the problem. In our case, we encode **Packing Sequences & ULD Loading Sequences** as chromosomes.

2. **Population Initialization**: The algorithm begins with an initial population of chromosomes, either randomly generated or heuristically derived, to cover the solution space effectively. In our case, we create the initial population using a random strategy for diversity.

3. **Fitness Function**: Each chromosome is evaluated using a fitness function, which quantifies the quality of the solution it represents. Higher fitness values indicate better solutions. In our case, it is the `packing efficiency` of the ULDs in consideration.

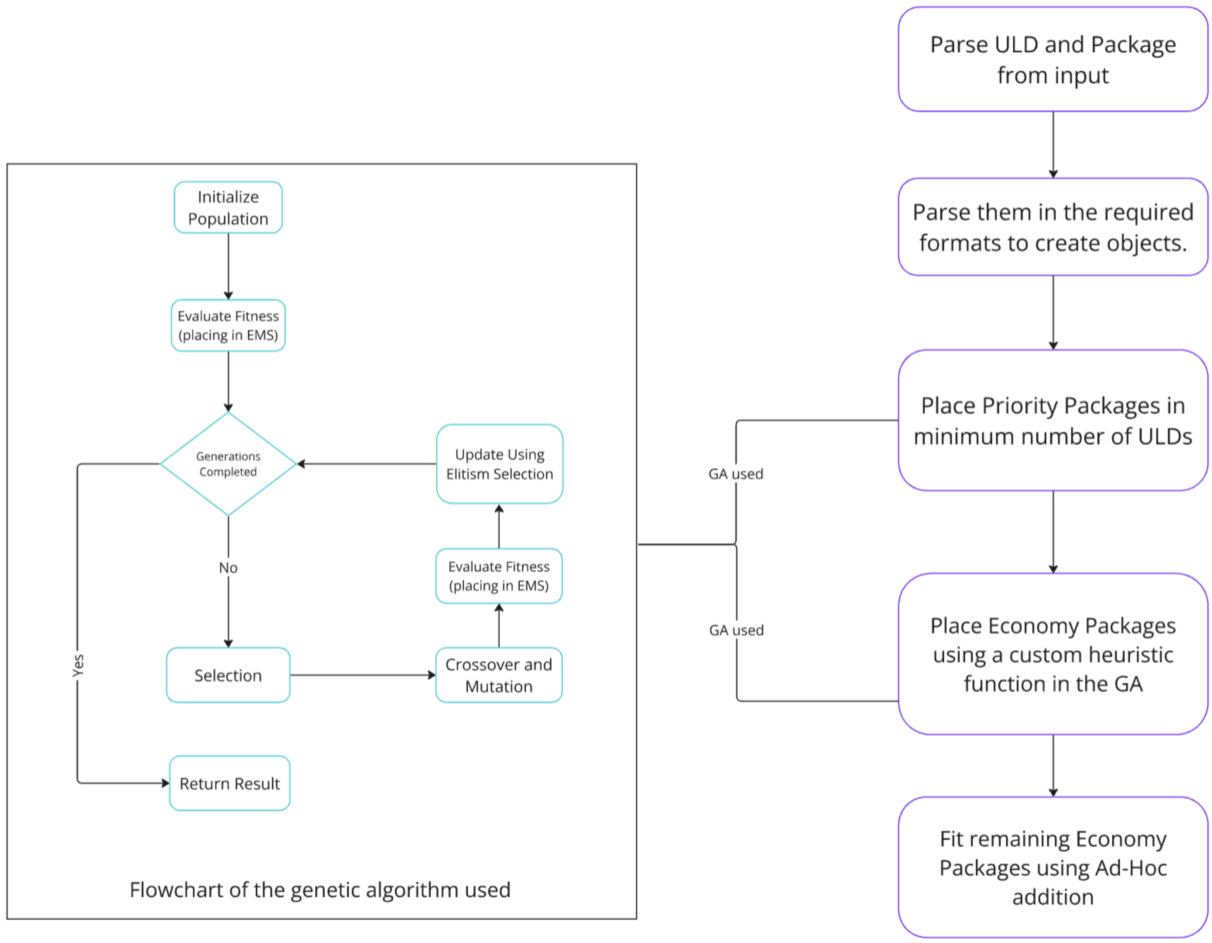
4. **Selection**: Selection mechanisms determine which chromosomes are retained or used as parents for the next generation. We have used the **elitism** selection strategy here - the best solutions are preserved across generations.

5. **Crossover (Recombination)**: Crossover is the process of combining genetic material from two parent chromosomes to produce offspring. We have used the **Two-Point crossover** technique, which involves multiple crossover points (CLS, BPS, explained in detail later) for the exchange of genetic material.

6. **Mutation**: Mutation introduces small, random changes to chromosomes to maintain genetic diversity and prevent premature convergence. For example, swapping, flipping, or altering a gene in the chromosome. We have used Swapping Mutation, where two randomly chosen genes in the chromosome (CLS or BPS) are interchanged, and for very short sequences (length 2 or less), the sequence is reversed instead.

7. **Termination Criteria**: We have terminated after a maximum number of iterations (generations) is reached. This number is given as an input to the algorithm.

§3.3 Workflow of our Genetic Algorithm



§3.4 Overview of our Genetic Algorithm (GA) based solution

GA is a population-based optimization method that evolves solutions through iterative selection, crossover, and Mutation processes. In our optimal cargo management problem, the GA encodes packing sequences and bin usage strategies as **chromosomes**, refining this encoding to find near-optimal solutions and then placing the packages optimally through EMS heuristics.

§3.4.1 Fitness Function

The fitness of a candidate solution is expressed as the percentage of unused volume it contains

§3.4.2 Chromosome Encoding Scheme

A chromosome comprises of two parts (similar to [1]):

- **Box Packing Sequence (BPS):** A permutation of the set of boxes to define the packing order.
- **Container Loading Sequence (CLS):** A permutation of bins to define the order of container usage.

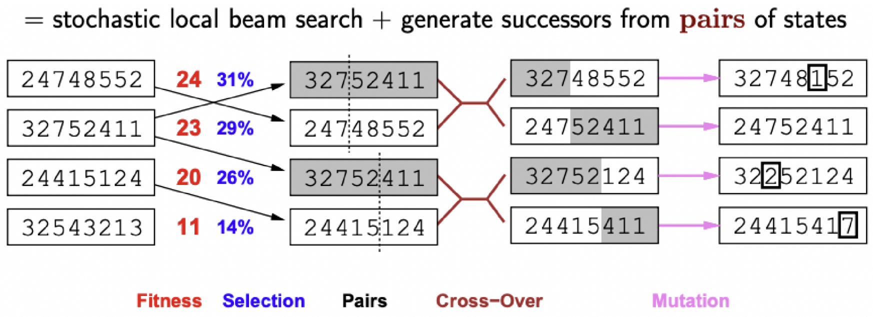
Our dual encoding would ensure flexibility in determining the packing arrangement and container allocation.

§3.4.3 Population Initialization and Selection

The initial population is generated randomly to introduce diversity. Our selection process uses tournament selection, where chromosomes are compared in pairs, and the fitter chromosome is more likely to be chosen for the next generation. **Elitism** ensures that the best solutions are preserved across generations.

§3.4.4 Crossover and Mutation

Our crossover operation employs two-point crossover with multiple crossover points (BPS, CLS), to generate offspring by combining parts of two parent chromosomes. We implement Mutation by swapping genes at two random positions in a chromosome (BPS, CLS), which introduces variability and helps escape local optima.



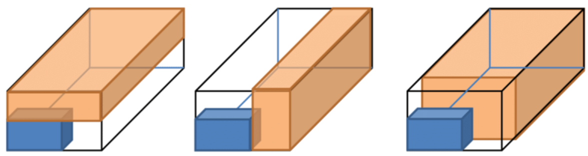
An example of a one-point crossover technique - our algorithm employs a two-point crossover (Ref: [3])

§3.5 EMS Based Heuristic Packing Strategy

Once we get the chromosome, which determines the packing order and container usage order, we place the packages optimally as per our pipeline (explained in the flowchart). So, our heuristic packing strategy evaluates chromosomes by translating their encoded sequences into feasible packing solutions, as follows:

§3.5.1 Empty Maximal Spaces (EMS)

The EMS concept represents the largest free spaces available within a bin that are not entirely contained within other spaces. EMS is dynamically updated as boxes are packed, ensuring accurate tracking of available packing regions. This approach improves efficiency compared to traditional corner-based methods.



Visualizing Empty Maximal Spaces (Ref: [1])

§3.5.2 Priority of EMS

Prioritization is based on the smallest coordinate values of EMS, starting with the smallest x coordinate, followed by y and z in case of ties. This strategy aims to pack boxes into corners and along the edges of the bin, minimizing internal gaps.

§3.5.3 Placement Selection

For each packing iteration:

1. A subset of boxes and EMS regions is considered based on the Box Packing Sequence and EMS priority.

2. Feasible box-EMS pairs are evaluated based on the fill ratio (volume of the box to the EMS volume) and margin (distance from the box to the edges of the EMS).

3. The pair with the highest priority is selected for placement.

If no feasible placement is found within the current bin, the algorithm progresses to the next bin as per the Container Loading Sequence. This process continues until all boxes are packed or no feasible packing solution remains.

§3.6 Algorithm Flow

The following procedure evaluates each chromosome by performing a packing operation and calculating its fitness based on packing efficiency. The fitness function measures the optimal arrangement of packages, considering factors like volume and cost. This guides the algorithm towards efficient packing solutions.

Algorithm 1 EMS Heuristic Packing Procedure

Input: Box Packing Sequence (BPS), Container Loading Sequence (CLS) from the chromosome
Output: Packing solution or `null` if not found

Let $OC \leftarrow \emptyset$ ▷ List of opened containers
while $BPS \neq \emptyset$ **do**
    Initialize priority queue P for candidate placements (as in section 3.5.2), $boxplaced \leftarrow$ `false`
    for each opened container c in OC **do**
        Let $EMSs$ be the empty maximal spaces in c
        for $j \leftarrow 1$ to $EMSs$.size() while $boxplaced =$ `false` **do**
            for $i \leftarrow 1$ to min(k_b, BPS.size()) **do**
                for all 6 orientations b_o **do**
                    if $BPS[i]$ can fit in $EMSs[j]$ with b_o **then**
                        Add placement to P, $boxplaced \leftarrow$ `true`; **break**
    if $boxplaced =$ `false` **then**
        while $CLS \neq \emptyset$ and $boxplaced =$ `false` **do**
            Let EMS be the initial empty space in $CLS[0]$
            $OC \leftarrow OC \cup \{CLS[0]\}$, $CLS \leftarrow CLS \setminus \{CLS[0]\}$
            for $i \leftarrow 1$ to min(k_b, BPS.size()) **do**
                for all 6 orientations b_o **do**
                    if $BPS[i]$ can fit in EMS with b_o **then**
                        Add placement to P, $boxplaced \leftarrow$ `true`,**break**
    if $P \neq \emptyset$ **then**
        Make the placement indicated by $P[0]$, Update $EMSs$
    else
        return `null`
return Packing solution

The following method represents our final packing strategy which employs a genetic algorithm for first placing the priority packages and then the economy packages.

Algorithm 2 Our Genetic Algorithm based solution

```
First parse ULD and Package from inputs into the required format
Initialize the population for placing priority packages
Evaluate each chromosome using the fitness (packing efficiency) for packing given by 1)
while termination condition not met do
    Select parents based on fitness
    Apply crossover to generate offspring and Mutation to introduce diversity
    Replace the old population with the new generation, preserving elite solutions
Reinitialize population for economy package placement.
Sort economy packages ordered by (cost / volume^1.2) for packing given by 1)
while termination condition not met do
    Select parents based on fitness
    Apply crossover to generate offspring and Mutation to introduce diversity
    Replace the old population with the new generation, preserving elite solutions
Once economy ones are placed, we call the ad-hoc procedure for adding remaining packages.
Run Greedy Priority Algorithm (2.1) in parallel as fallback for local optima
return Return the better solution out of the GA and Greedy Solution
```

§3.7 Advantages of Genetic Algorithm and EMS Packing

- **Simple Computations of EMS Packing**: EMS-based packing reduces computational overhead compared to traditional methods as only simple geometric operations are used to compute optimal placements.

- **Quality of Solutions:** The Genetic Algorithm effectively explores the solution space, balancing exploration and exploitation, and produces near-optimal results.

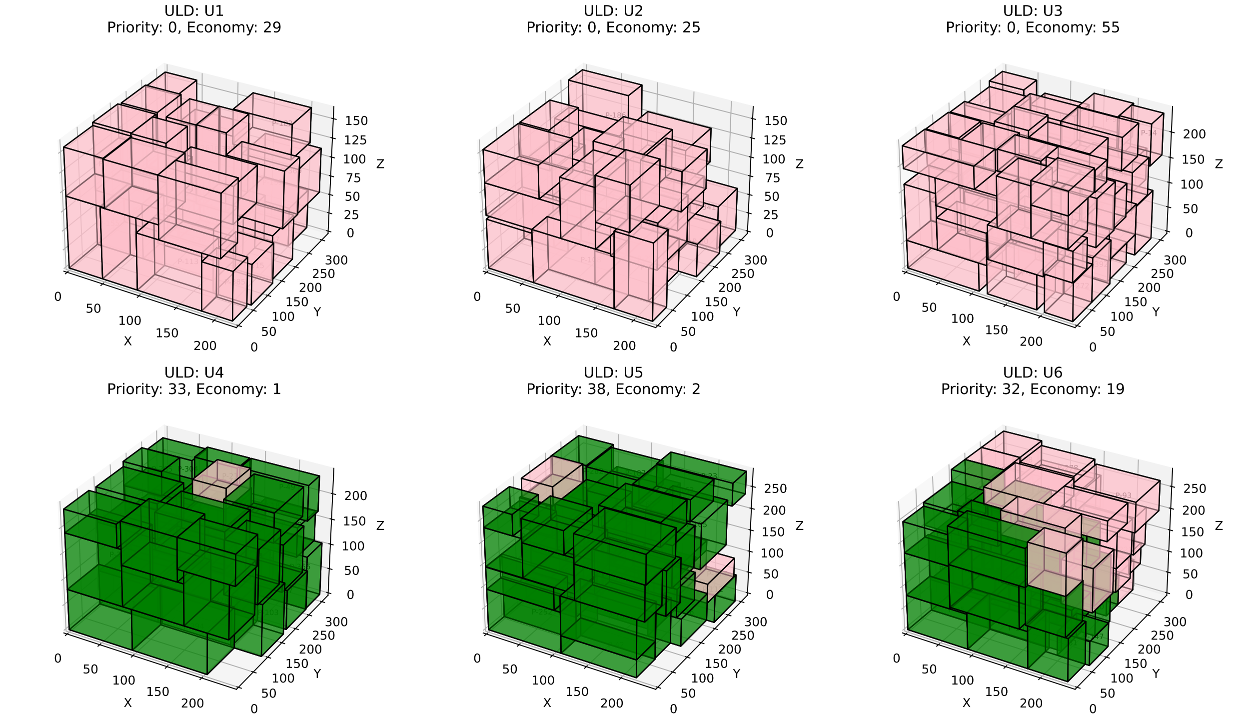
- **Flexibility**: GAs can be easily adapted to new problem domains by modifying the representation, fitness function, or operators. This means that a slightly different packing problem (perhaps involving more levels of priority or different ULD shapes or different objective function) can be modelled relatively easily with the framework we have built by tweaking a few functions.

§3.8 Limitations of Genetic Algorithm

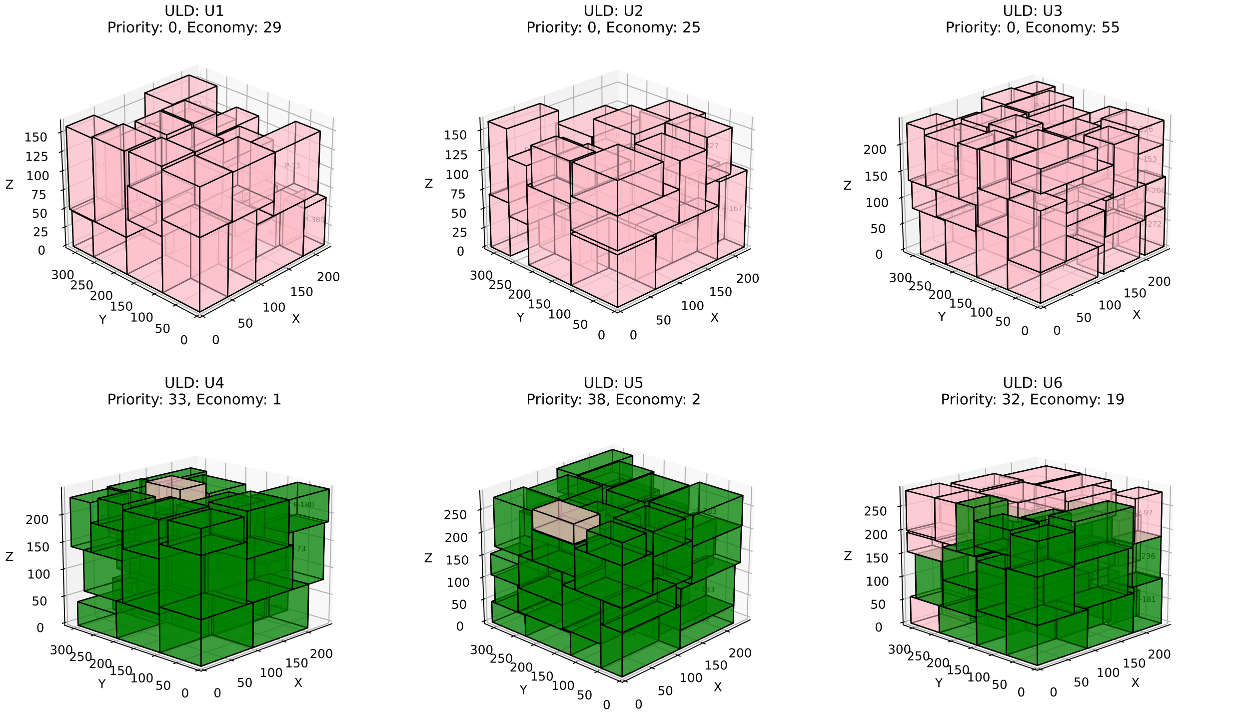
- **Parameter Sensitivity**: Performance depends on careful tuning of parameters like population size, mutation rate, and crossover rate. We chose the parameters based on limited experimentation with the problem description at hand and the optimal parameters of a new packing problem may be different.

- **Premature Convergence**: Without sufficient diversity, the population may converge to suboptimal solutions. This is due to the fact that genetic algorithms and other local search techniques rarely optimize for a global solution.
  We deal with premature convergence partially by adding a fallback option to the Greedy Packing Algorithm as explained in 2.1

Optimal package placement using our genetic algorithm



Optimal package placement using our algorithm (different view - 90° rotated)



We obtain a cost 30111 in the above solution by following the above approach.

§4 Visualization and Validation

We created a Python script which can help plot the ULDs and all the packages inside the ULDs with green colour for the priority packages and pink colour for the economy packages. The script also lets us rotate the ULDs and get a view from different angles, which helped a lot in the heuristics-making process to see the outcomes of the heuristics tried visually and to improve and fine-tune our heuristics. Please refer to the README.md for the visualization commands.

A python script `validator.py` is also included in the submitted solution package, which validates the solution generated, ensuring that all the packages are inside the ULDs, there is no overlap among packages, and the total weight of packages is less than or equal to the maximum allowed weight capacity of the ULDs. It also checks that all priority packages are loaded.

References

[1] X. Li, Z. Zhao, and K. Zhang, *A genetic algorithm for the three-dimensional bin packing problem with heterogeneous bins*, *IIE Annual Conference and Expo 2014*, May 2014. Available at: Link.

[2] R. Khir, A. Erera, and A. Toriello, *Two-Stage Sort Planning for Express Parcel Delivery*, H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA. Available at: GaTech.

[3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach (3rd Edition)*, Pearson, 2024, p. 127, Figure 4.6.

[4] A. Mungwattana, T. Piyachayawat, and G. Janssens, *A two-step evolutionary algorithm for the distributor's pallet loading problem with multi-size pallets*, *Flexible Services and Manufacturing Journal*, vol. 35, Aug. 2022, doi: 10.1007/s10696-022-09461-y.