End Term Report

Team-79

Team FedEx

Contents

1 Approach Outline

1.1 Introduction

Unit Load Devices (ULDs) are standardized containers used in air cargo to optimize space and ensure efficient transport. The problem lies in determining which packages to load into which ULDs, ensuring the weight and size constraints are met while minimizing the total number of ULDs used. Priority packages must be handled with special care to avoid delays, while maximizing space usage reduces operational costs.

An efficient ULD packing solution can significantly improve operational efficiency, reduce fuel consumption, and ensure timely deliveries, making it a critical challenge in logistics.

1.2 Problem Statement

The task involves determining how to load packages into the ULDs, assigning each package to a specific ULD or marking it as ”NONE” if it cannot be loaded. The optimization should also provide the coordinates of each package within the ULD based on a fixed reference point and the packing of boxes should follow some specific constraints.

1.3 Challenges and Constraints

The Unit Load Device (ULD) packing problem involves several challenges that must be addressed to achieve an optimal solution:

- **Weight Constraints:** The total weight of packages in each ULD must not exceed its weight limit.
- **Priority Package Handling:** Priority Packages must be loaded first and grouped into as few ULDs as possible to minimize delivery time. No Priority Package should be left behind.
- **Space Utilization:** Packages must fit within the ULD’s dimensions without overlap. Proper orientation of packages along the ULD’s axes is required to maximize space usage.
- **Cost Minimization:** The solution should minimize two key costs:
  - *Delay Costs:* Costs incurred by packages that cannot be loaded onto ULDs.
  - *Spreading Priority Packages:* Costs for spreading Priority Packages across multiple ULDs.
- **Multiple ULD Types:** Different ULDs with varying dimensions and weight capacities are available, and the solution must efficiently allocate packages to these ULDs.

- **Package Dimensions and Orientation:** Each package has fixed dimensions, and it can only be oriented along the ULD's axes, creating multiple packing configurations.
- **Handling of Economy Packages:** Economy packages can be left behind if there is insufficient space in the ULD, but Priority Packages must always be shipped.

1.4 Identified Use Case

The ULD packing optimization problem is critical for improving the efficiency of air cargo logistics. A key use case is for FedEx to optimize the loading of packages into Unit Load Devices (ULDs) for air shipments. By efficiently packing ULDs, the system can minimize operational costs, reduce delays, and ensure timely delivery of Priority Packages.

1.5 Solution Overview

The proposed solution to the ULD packing problem involves a combination of heuristic and optimization algorithms to handle the various constraints efficiently. The approach can be broken down into the following steps:

1. **Input Processing:** The first step is to process the input data, including the list of ULDs and packages, each with their respective attributes (dimensions, weight, priority type, etc.) and creating objects corresponding to each bin and box.
2. **Package Assignment:** Packages are first categorized based on their priority (Priority vs. Economy). Priority Packages are assigned to ULDs first, minimizing the number of ULDs they are spread across.
3. **Space Optimization:** A spatial optimization algorithm is used to determine the best orientation and positioning of each package within the ULD. This ensures that the packages do not overlap and that the ULD's weight and space constraints are not exceeded.
4. **Cost Calculation:** The algorithm calculates the total cost based on the delay of packages and the spreading of Priority Packages across ULDs. The solution aims to minimize both of these costs.
5. **Output Generation:** The final output includes the ULD identifier for each package, the coordinates of the packages inside the ULD, and the total cost of the solution.

This approach ensures that all constraints are satisfied while minimizing costs, improving operational efficiency, and enhancing customer satisfaction. The solution is scalable to handle varying numbers of ULDs and packages, making it adaptable to real-world logistics scenarios.

2 Technical Approach

We evaluated multiple algorithms for the ULD packing problem, with the final and best approach detailed at the end.

2.1 Fitting Algorithms

In the context of the ULD packing problem, we evaluated four fitting algorithms to determine their effectiveness in optimizing package loading.

- **Best Fit :**
  - **Suitability :** Best Fit is well-suited to this problem as it places each package in the ULD with the least remaining space, helping to maximize ULD capacity and minimize gaps.
  - **Rejection :** The algorithm, while effective at reducing gaps, often leads to fragmentation within the ULD. This fragmentation can result in wasted space in subsequent loading steps and may require spreading priority packages across multiple ULDs, which increases costs and delays.
- **Next Fit:**
  - **Suitability :** Next Fit is a simpler approach where packages are placed into the first available ULD until full. It offers faster packing times and easier implementation.
  - **Limitations :** This method tends to create inefficient packing, often requiring more ULDs and wasting valuable space, particularly when dealing with priority packages. Its simplicity doesn't support the complex constraints of the ULD packing problem.
- **Worst Fit:**
  - **Suitability:** Worst Fit places packages in the ULD with the most remaining space, potentially allowing for better future packing flexibility. This could help avoid overloading a ULD too quickly.
  - **Rejection:** In practice, Worst Fit often creates large gaps in the ULD, leading to poor space utilization. It also risks spreading priority packages across several ULDs.
- **First Fit:**
  - **Suitability:** First Fit is the fastest and simplest approach, placing each package in the first available ULD that can accommodate it. It's easy to implement and typically requires fewer computational resources.

  - **Rejection:** Although quick, First Fit often leads to inefficient use of space, especially when handling a large number of priority packages. It fails to group priority packages together in a single ULD.

2.2 Mixed Integer Linear Programming (MILP) Optimization

MILP optimizes ULD packing by modeling constraints such as weight limits, grouping priority packages, and spatial fitting.

- **Suitability:** Provides optimal solutions by minimizing costs and delays.
- **Rejection:**
  - **Complexity:** Requires numerous binary and continuous variables for large-scale problems.
  - **Computational Expense:** Infeasible for real-time solutions due to high resource demands.

2.3 Simulated Annealing (SA)

Simulated Annealing is a probabilistic optimization technique inspired by metallurgy, minimizing cost functions based on volume utilization, weight, and priority box placement.

- **Process:** Starts with a greedy solution, modifies configurations iteratively, and gradually reduces the acceptance of worse solutions through temperature decay.
- **Limitations:**
  - High computational cost due to iterative evaluations.
  - No guarantee of optimality.
  - Struggles with scalability for large problems.

2.4 Final Solution using Grasp and Maximum Rectangle Overlap Algorithm

This programme aims to optimize the packing of a given set of boxes into multiple bins. It minimizes the number of unpacked boxes and the overall cost. Below is a overview of the method which we are following :

1. **Initialization:**
   - We begin by creating classes for boxes and bins, assigning them their respective properties.
   - Next, the boxes are sorted based on their categories and then by boxes criterion (decreasing volume, weight or area)

- The bins are sorted in descending order of volume, weight and area (in that priority decided by which provides the best results) to ensure that a greater number of priority packages can be accommodated in fewer bins.
- Variables are initialized to track the minimum number of unpacked boxes (`min_unpacked`), the minimum cost (`min_cost`), and the best solution (`best_solution`).

2. **Iterative Packing Process:**

   (a) Loop over all combinations of:
   - Sorted bins (`bins_choice`)
   - Sorted boxes (`boxes_choice`)
   - Bins to be unpacked and refilled: bins that do not contain priority boxes.

   (b) Reset all bins to their initial state before each iteration.

   (c) Our goal is to pack the boxes into the bins by exploring all possible orientations of the boxes. We prioritize the orientation with the greatest height and then place the box in the appropriate space accordingly.

   (d) Placing a box in a specific space involves several steps. First, the most suitable space for the box is identified based on the maximal fit. Then, the spaces are updated according to the 3D-optimized Maximal Rectangle algorithm. Afterward, any spaces that are contained within another space are merged. For further clarification, you can refer to the figure below.

3. **Phase 1: Initial Packing and Space Updation**
   - If the box fits in the available space, it is placed, and the parameters for that space are updated accordingly. The list of available free spaces is then updated using the maximum rectangle overlap technique to ensure efficient utilization of the remaining space.
   - The Maximal Rectangle Algorithm efficiently packs boxes by identifying the largest available rectangular spaces in a bin, using a 3D-optimized version of the Guillotine Split Algorithm. When placing a new box, it evaluates multiple potential spaces, selecting the best fit. After placement, the free space is split into smaller regions, and any contained spaces are merged and removed. The algorithm continues to place, split, and merge spaces, optimizing the packing process.
   - If a box cannot be packed, add it to the `unpacked_boxes`.

4. **Phase 2: Unpacking and Repacking Economy Boxes**
   - The formula calculates a score for each box based on its dimensions, volume, weight, and cost. Boxes with optimal dimensions and higher cost are

ranked higher, while weight has a lesser influence. Cost, raised to a high exponent, significantly affects the ranking, prioritizing cost-efficiency and add the boxes with lower score to `unpacked_boxes`.

- Attempt to repack the unpacked boxes into all bins.
- It is observed that this reshuffling of boxes actually leads to packing more number of boxes.

5. **Solution Evaluation:**
   - Compute the cost of the current bin configuration using a `cost` function.
   - If the current solution has a lower cost than `min_cost`, update `best_solution`.
6. **Output:**
   - Return the `best_solution`, which includes the optimal arrangement of most of the bins and packed boxes.

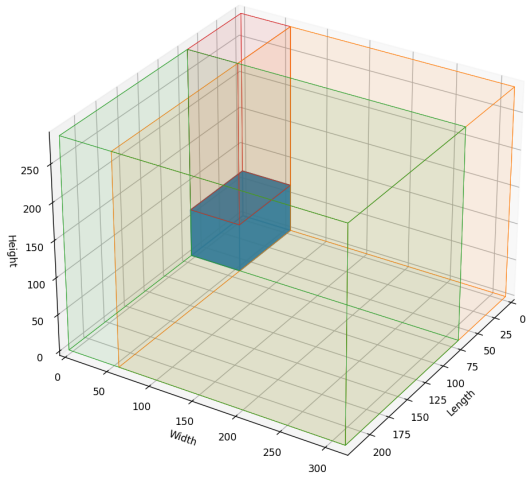

Figure 1: Bin showing only 1 box and the free spaces divided by it

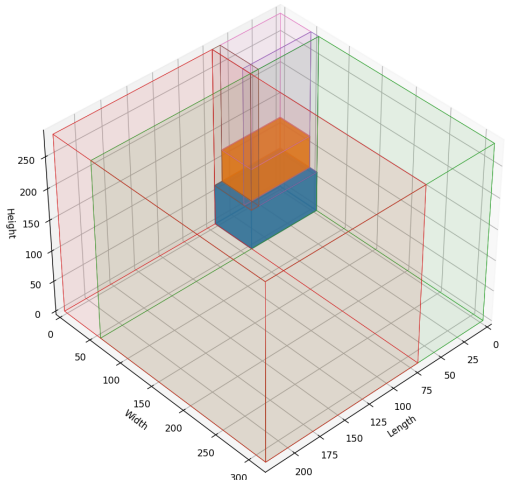

Figure 2: Bin showing 2 boxes packed and the free spaces divided by it

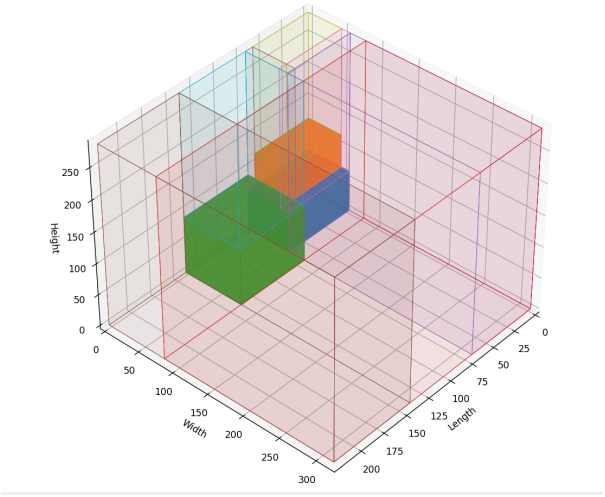

Figure 3: Bin showing 3 boxes and the spaces divided by it

Key Features

- Uses 3D-optimized packing algorithm to optimize space utilization .
- Handles packing and repacking of boxes for improved efficiency.

2.5 Advantages of this Solution

The Maximal Rectangle Algorithm and GRASP-based approach provides key advantages over standard algorithms like First-Fit, Best-Fit, HSA, and MILP:

- **Avoidance of Local Minima:** Unlike Best-Fit and HSSA, Maximal Rectangle Algorithm prioritizes strategic box placements, maximizing overlap and improving global packing efficiency.
- **Computational Efficiency:** GRASP achieves near-optimal solutions with lower computational cost compared to MILP, making it suitable for large datasets.
- **Flexibility and Adaptability:** GRASP's randomized search dynamically explores diverse solutions, handling varying box and bin configurations effectively.
- **Improved Space Utilization:** Maximal Rectangle Algorithm reduces wasted space by optimizing the overlap between boxes and free spaces.
- **Scalability:** The approach scales well for both small and large instances, avoiding exponential complexity seen in MILP.

2.5.1 Conclusion

The Maximum rectangle overlap method works efficiently to minimise the voids between boxes and using that our current model packs 225 boxes in total.

3 Analysis and Initial Experiment

3.1 Issues Addressed

The ULD packing optimization system addresses several key challenges in logistics operations:

- **Space Utilization:** Efficiently packing packages to maximize ULD capacity.
- **Weight Limits:** Ensuring packages are loaded without exceeding ULD weight restrictions.
- **Priority Package Handling:** Prioritizing high-value packages to minimize delays and reduce spreading across multiple ULDs.
- **Cost Minimization:** Reducing costs related to package delays and excessive ULD usage.

3.2 Experiments and Evaluation

- **Test Data:** 400 total boxes, including 103 priority boxes.
- **Performance Achievements:**
  - Utilised 78748251 cubic units of volume in packing out of 105171504 cubic units achieving a total volume utilisation of 74.88%.
  - Packed 13829 units of weight out of given 17600 units of weight achieving a total weight utilisation of 78.57%.
  - Successfully packed all 103 priority boxes into 3 bins without any overlap along any dimension.
  - Packed a total of 225 boxes in 6 bins, following the volume and weight constraints.
  - Very few boxes were left unpacked, achieving high packing efficiency.
- **Cost Efficiency:** The total packing cost is minimized to 30,893.

4 Conclusion

The ULD packing optimization system improves logistical efficiency by automating package loading into Unit Load Devices (ULDs). It dynamically adapts to various shipment complexities, ensuring space optimization, adherence to weight limits, and timely delivery of priority packages. By reducing operational costs and minimizing delay and spreading costs, the system enhances resource utilization and service reliability. This solution offers scalability for different shipment types while maintaining security and environmental standards, ultimately improving FedEx's logistics operations and customer satisfaction.

5 Challenges and Next Steps

5.1 Technical Challenges

The ULD packing optimization presented several technical difficulties, including:

- **Multi-Dimensional Optimization:** Balancing package dimensions, weights, and ULD constraints led to a complex multi-variable problem requiring sophisticated algorithms.
- **Combinatorial Explosion:** The large number of potential configurations resulted in exponential complexity, making it difficult to find optimal solutions within time limits.
- **Heuristic Optimization:** Exact algorithms were computationally expensive, requiring the development of effective heuristics to balance speed and quality.
- **3D Packing Complexity:** Efficiently modeling and optimizing 3D package placement inside ULDs, while preventing overlap, added significant computational complexity.
- **Multi-Objective Balancing:** Optimizing conflicting objectives, such as minimizing ULD usage and delays, required advanced multi-objective optimization techniques.
- **Scalability:** Handling large data sets with hundreds of ULDs and packages demanded efficient memory management and scalable algorithms.

5.2 Future Work

The following areas can be explored to further enhance the ULD packing optimization system:

- **Integration with AI-based Predictive Models:** Leverage machine learning to predict package types, weights, and volumes to improve packing efficiency.
- **Dynamic Repacking:** Implement dynamic repacking algorithms to accommodate late package additions or changes in flight schedules.
- **Real-Time Data Integration:** Enhance the system's ability to handle live data streams and adjust optimizations in real-time based on new package or ULD information.

6 Research and References

[1] Zheng, Linjiang, Mao, Xingxing, Li, Qiqi, and Liu, Weining, ”A Hybrid Genetic Simulated Annealing Algorithm for Three-Dimensional Packing Problem.” Available at SSRN:

[2] R. Alvarez-Valdes, F. Parreño, J.M. Tamarit, ”A GRASP/Path Relinking algorithm for two- and three-dimensional multiple bin-size bin packing problems,” *Computers Operations Research*, Volume 40, Issue 12, 2013, Pages 3081-3090, ISSN 0305-0548, https://doi.org/10.1016/j.cor.2012.03.016.

[3] Rommel D. Saraiva, Napoleão Nepomuceno, Plácido R. Pinheiro, ”A layer-building algorithm for the three-dimensional multiple bin packing problem: a case study in an automotive company,” *IFAC-PapersOnLine*, Volume 48, Issue 3, 2015, Pages 490-495, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2015.06.129.