OPTIMIZATION OF STOCK KEEPING AND SPACE UTILIZATION POLICY FOR A WAREHOUSE

Faris ALBAKAR [1], Mhd Hazem ALHAMMAMI [2], Husam Jehad Hasan OMAR [3], Mazen SIMI [4], Basel ZBEDA [5], Zehra DUZGIT [6*]

[1, 2 ,3, 4, 5, 6] İstanbul Bilgi Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Endüstri Mühendisliği, İstanbul

[1] ORCID No : https://orcid.org/0009-0001-0614-6030, [2] ORCID No : https://orcid.org/0009-0004-0147-9823,
[3] ORCID No : https://orcid.org/0009-0008-9649-9622, [4] ORCID No : https://orcid.org/0009-0008-0542-0238,
[5] ORCID No: https://orcid.org/0009-0008-5718-330X, [6] ORCID No : https://orcid.org/0000-0003-0686-1672

Keywords

Warehouse,
Variable-Sized Bin Packing Problem,
First Fit Decreasing,
Best Fit Decreasing,
Next Fit Decreasing

Abstract

This study considers the one-dimensional variable-sized bin packing problem (VSBPP) which is an NP-Hard problem. In this study, the objective is to find an efficient solution that minimizes both the total capacity of the bins used and the number of bins required, thereby optimizing the company's storage policy and saving space within the warehouse. Three algorithms are employed to solve a real warehouse's VSBPP: i) First Fit Decreasing (FFD), ii) Best Fit Decreasing (BFD), and iii) Next Fit Decreasing (NFD). The warehouse dataset includes items of various sizes, and the goal is to allocate these items into bins most efficiently. Experimental results demonstrate that the FFD and BFD algorithms outperform the NFD algorithm. Furthermore, all three algorithms significantly reduce storage space usage and improve space utilization compared to the warehouse's current practices.

BİR DEPO İÇİN ENVANTER TUTMA VE ALAN KULLANIMI POLİTİKASININ ENİYİLEMESİ

Anahtar Kelimeler

Depo,
Değişken Boyutlu Kutulama Problemi,
İlk Bulduğun Boşluğu Doldur,
En İyi Boşluğu Doldur,
Sonraki Boşluğu Doldur

Öz

Bu çalışmada NP-Zor bir problem olan tek boyutlu değişken ölçekli kutulama problemi ele alınmaktadır. Bu çalışmada amaç, kullanılan kutuların kapasitelerinin toplamını ve kullanılan kutu sayısını en aza indiren verimli bir çözüm bulmak, böylece şirketin depolama politikasını optimize etmek ve depoda yer tasarrufu sağlamaktır. Gerçek bir deponun problemini çözmek için üç farklı yöntem: i) İlk Bulduğun Boşluğu Doldur (İBBD), ii) En İyi Boşluğu Doldur (EİBD) ve iii) Sonraki Boşluğu Doldur (SBD) algoritmaları kullanılmıştır. Deponun veri seti, farklı boyutlarda çeşitli öğelerden oluşmakta olup amaç bu öğeleri en verimli şekilde kutulara tahsis etmektir. Deneylerden elde edilen sonuçlara göre, İBBD ve EİBD algoritmaları SBD algoritmasından daha iyi olmakla beraber, her üç algoritmanın da mevcut depo uygulamasına kıyasla depolama alanı kullanımını azaltmada ve alandan yararlanmayı arttırmada başarılı olduğu gösterilmiştir.

Araştırma Makalesi
Başvuru Tarihi : 07.02.2024
Kabul Tarihi : 19.12.2024

Research Article
Submission Date : 07.02.2024
Accepted Date : 19.12.2024

* Sorumlu yazar: zehra.duzgit@bilgi.edu.tr
https://doi.org/10.31796/ogummf.1432654

1. Introduction

The management of warehouses is an essential function for any company that trades in tangible commodities. It should be a priority since it gives companies the ability to retain competitiveness. Warehouse management entails several processes, such as optimizing space and maintaining inventory records. Proper warehousing management is usually faced with several widely known problems. Hemmelmayr, Schmid, and Blum (2012) stated that a common optimization problem called the Bin Packing Problem (BPP) consists of packing items into bins. Finding a reasonable solution to the BPP is a way of saving resources such as storage space. Therefore, it is essential to find solutions by determining the most effective method for packing different items into bins.

This research provides a way of effectively storing items in bins of a particular warehouse. Finding an effective way to store leads to saving space inside a particular warehouse, which can be utilized for other warehousing activities, and dismissing the option of renting new storage spaces to store all the items at hand. This study can be used as a reference amongst other industries that face the redundant problem of effectively storing their Stock Keeping Units (SKUs) within bins.

The warehouse under consideration is a maintenance, repair, and overhaul (MRO) warehouse and faces challenges in effectively utilizing its space. MRO covers preventive maintenance, component repair, replacement, deep maintenance, structural repairs, etc. In practice, the warehouse's space is being wasted due to incorrect storage techniques. This indicates an underlying lack of organization, which must be addressed to maximize efficiency and eliminate wasted space. This leads to requiring expensive external storage solutions. To overcome this, the warehouse management desires to optimize existing space usage through better storage techniques, aiming to store more parts without acquiring or renting additional warehouses.

The rest of the paper is organized as follows. Literature review is given in Section 2. In Section 3, the problem is defined. The solution approach is explained in Section 4. Computational results are reported in Section 5. The conclusion is given in Section 6.

2. Literature Review

BPP is one of the combinatorial optimization problems that has been well studied for many years because of its economic benefits on different sectors and the high number of industrial applications. BPP involves a set of items or boxes (small objects) of various dimensions that must be assigned to containers or bins (large objects) with capacity constraints to minimize the number of required bins to pack all items. Several variants of BPP differ in terms of dimensionality, type of bins (identical, variable size), and static/dynamic nature. One classification was introduced by Dyckhoff (1990), and another improved typology was proposed by Wäscher, Haußner, and Schumann (2007).

In the classical one-dimensional BPP (1DBPP), only a single dimension of the item is taken into consideration (usually width), and all bins are identical with the same capacity, where the objective is to minimize the total number of used bins. In the two-dimensional BPP (2DBPP), the aim is to pack items while only considering two-dimensional (usually width and height) rectangular items into identical two-dimensional bins. Lodi, Martello, Monaci, and Vigo (2013) stated that 2DBPP has many industrial applications, such as wood and glass cutting, in addition to packing in transportation and warehousing. The strip packing problem is a variation of the 2DBPP where bins of width and infinite height (therefore, this is referred to as strip) and a set of rectangular items are given, and the objective is to determine the way to pack the items within the strip such that the height of the strip is minimized. Wäscher et al. (2007) have classified this problem in their typology as an open dimension problem, one of the applications of strip packing in manufacturing where rectangular pieces must be cut from a roll of cloth or paper with a fixed width and infinite height. According to Jin, Ito, and Ohno (2003), a three-dimensional BPP (3DBPP) is a generalization of the one and two-dimensional BPPs in which the three dimensions (width, length, and height) of both the items and bins are taken into consideration. 3DBPP has the most practical applications, such as in transportation, where it is used to determine the most efficient way to load boxes onto a transportation vehicle or containers, minimizing the amount of space wasted. 3DBPP appears in a range of contexts, such as in the distribution and storage of goods, manufacturing, the packaging of items, and the use of space in various settings.

A variant of bin packing is variable-sized BPP (VSBPP), where bins or containers have different capacities, and each item must be packed in turn. The objective is to minimize the sum of the capacities of the bins. VSBPP adds a layer of complexity to the problems because of the variable size of the bins. The VSBPP has practical applications in areas such as packing, transportation planning, and cutting, but according to Hemmelmayr et al. (2012), there is not a lot of published research on the VSBPP. Furthermore, the bin packing problem can be classified as either online or offline. The main difference between online and offline bin packing problems is that in online bin packing, items arrive one at a time, and a decision must be made about which bin to place each item in as soon as the bin becomes available without knowing the complete set of items in advance (Boyar, Kamali, Larsen, and López-Ortiz, 2013). This contrasts with the offline bin packing problem, where the complete set of items is known, and the bins can be optimized for the entire set of items at once.

Offline algorithms perform better than online algorithms on the bin packing problem because they receive the sequence of items in advance (Karp, 1992). However, online algorithms have the advantage of being able to start packing items right away without having to wait for the entire set of items to become available. Online algorithms can be useful in situations where the items are arriving continuously and must be packed immediately as opposed to being available all at once.

Additionally, the bin packing problem can be studied in a dynamic setting, where items can be added or removed from the bins during the execution of an algorithm (Gupta, Guruganesh, Kumar, and Wajc, 2018). To achieve the best possible space utilization, the algorithm must be able to adapt to changes in the input data and rearrange the items in the bins. On the contrary, in a static bin packing setting, items are known beforehand when they are assigned to the bins.

Most existing work focuses on static bin packing in the sense that items do not depart. In some potential applications like warehouse storage, a more realistic model takes into consideration the dynamic arrival and departure of items. In dynamic bin packing, items arrive over time, reside for some time, and may depart at an arbitrary time. The BPP is strongly NP-Hard and very complex to solve in practice (Martello and Vigo, 1998). Hence, many different methods have been developed to solve this problem.

Exact algorithms, such as integer programming, dynamic programming, and branch and bound, guarantee finding the optimal solution to bin packing problems. However, their high computational complexity often renders them impractical for large-scale instances, as highlighted by Coffman, Garey, and Johnson (1984). On the other hand, heuristic and meta-heuristic algorithms are primarily employed to provide high-quality solutions when optimality cannot be reached within a reasonable running time. Meta-heuristic algorithms, including simulated annealing, genetic algorithms, variable neighborhood search, and tabu search, have been widely applied in solving combinatorial optimization problems (Aarts and Korst, 2003; Hemmelmayr et al., 2012).

In this study, we tackle an NP-hard problem by adopting a mathematical model and tracking the running time for different problem sizes. Afterward, three heuristic algorithms, which are developed for identical-size bin packing problems, are adapted for variable-sized bin packing problems, and their performance measures are reported.

3. Problem Definition

Proper warehouse management is one of the most impactful aspects that can affect a company's success. In today's world, warehouse performance measurements can be the deciding factor in determining companies' global rankings (Tompkins and Smith, 1998). There are several criteria to consider for a warehouse to be classified as properly managed. These criteria include picking systems, optimizing space, maintaining inventory records, following safety standards, and controlling labor expenses. Optimizing picking systems plays a critical role in enhancing both customer satisfaction and the overall efficiency of warehouse operations (Pinto, Nagano, and Boz, 2023). Also, optimizing space utilization includes multiple criteria, such as fitting as many SKUs into their desired storage containers and fitting the storage containers in an organized manner to minimize unutilized space. With every criterion having its own contribution weight to optimizing space utilization, this study focuses on fitting as many SKUs into their desired storage containers inside warehouses as the key factor for increasing warehouse efficiency. The handled problem is called the one-dimensional variable-sized bin-packing problem (VSBPP). The objective of the VSBPP is to minimize the sum of the used bin capacities which will lead to more space in the warehouse without the need to rent any additional warehouse.

A lack of coordination emerges between the planning department and the warehousing department inside the company. The planning department presumes a storage-containing unit can no longer store more SKUs, while in reality, the containers can store almost double the amount of the currently available items. To prevent the costly option of renting new warehousing spaces, this study focuses on storing more items within each bin to use fewer bins and generate more space for storing even more items if needed.

In the first phase, an ABC analysis is conducted to split the inventory into three main categories. A-items have the highest importance with respect to the value contribution, B-items have lower importance and value, and C-items have the least importance. It is one of the methods that help companies control warehouses by allowing the management to stay focused on the most important items. In addition to that, it aids in achieving effective stock management of resources and stock level optimization. ABC analysis is based on the Pareto principle, which assumes that 20% of the items generate 80% of the total value.

Grondys (2009) stated that ABC analysis allows companies to focus on the most expensive items. However, classifying the items in the company depends on different aspects, so analyzing and filtering data is the first step before applying the ABC analysis. The goal is to work on every aspect that the company considers important, then combine all the results together so a decision can be made in terms of which item should be classified to which category. Two different criteria were considered for the ABC analysis. The first criterion is based on the distinct count of the picklist, which is the number of orders for each item throughout the year. The second criterion is based on priority.

After conducting the ABC analysis, the manager in the company decided to go with the first criterion and proceed with only class A and B items since class C items were not ordered often per year and thus had minimal impact on the optimization process for storage space allocation. In addition, the warehouse management expressed a preference to assign a dedicated storage area for Class C items, located far from the main operational zones, without including them in the optimization process, as their low demand does not justify the computational effort or complexity of their inclusion.

The subsequent step was to determine the number of bins used, which amounted to 762 distinct bins for class A and B items. Table 1 shows the existing bins in use derived from three different bin sizes (small, medium, and large), with a total used capacity of 1,940,472 cm^3.

Table 1. Current Bin Structure and Used Capacity for Class A and B Items

Bin type	Bin size (cm^3)	Count	Total used capacity (cm^3)
Small	900	139	125,100
Medium	2,808	587	1,648,296
Large	4,641	36	167,076
	Total	762	1,940,472

By evaluating the current used capacity and establishing a value for benchmark, our primary objective is to minimize total used capacity. Since then, it would be reasonable to aim for 100% utilization per bin. However, including more items per bin increases the time for the picker to retrieve any desired items from the bin. It is, therefore, crucial to ensure that every bin contains some free space or buffer, in other words. Hence, all maximum bin capacities were decreased by 20% to make sure that some free space remained inside the bin.

In this study, the items are significantly smaller than the bins, with none of their dimensions (length, width, height) exceeding the respective dimensions of the bins. This allows the problem to focus on optimizing the total volume within the capacity constraints of the bins, effectively treating items as occupying liquid-like volumes.

4. Solution Methodology

After defining the problem with all the considerations, the solution approach should cover all the problem's aspects. Figure 1 explains the plan and the methods that the defined problem would be solved with. The initial goal is to find an optimal solution, but if the problem size at hand cannot be solved within a reasonable running time due to NP-Hardness, a feasible and effective alternative solution method should be implemented.

4.1. Exact Solution

A mathematical model aids in understanding the level of complexity of a certain problem. Knowing the level of complexity, henceforth, gives insight into the appropriate ways to approach the solution. It is proven that the one-dimensional bin-packing problem is classified as NP-Hard (Alenezi, Aboelfotoh, Albdaiwi, and Almulla, 2015). As stated by Haouari and Serairi (2009), the VSBPP is a generalization of the classical-dimensional bin-packing problem and therefore also NP-Hard.

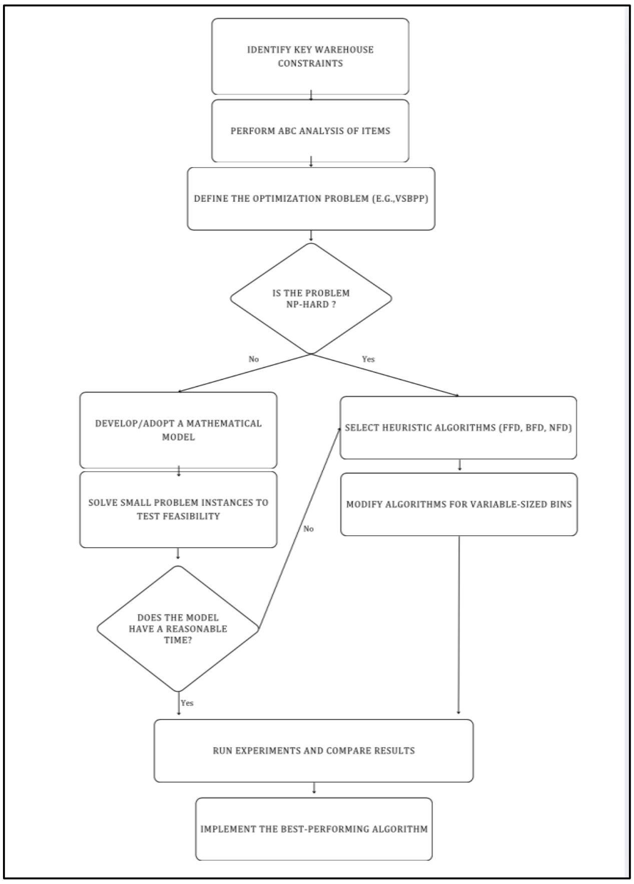

Figure 1. Flowchart of Solution Approach

The following mathematical model is developed by adapting the model proposed Hemmelmayr et al. (2012) which considers a general one-dimensional VSBPP. Some indices, parameters, and decision variables are modified according to the problem at hand. In addition, the objective function was revised to minimize the total utilized bin capacities, to align with the company warehouse optimization goals. There are three indices in the model. i represents the items$(i = 1,2,\cdots,n)$ where n is the number of items; j represents the bins $(j = 1,2,\cdots,m)$ where m is the number of bins; k represents the bin type $(k = 1,2,\cdots,r)$ where r is the number of bin types.

In the model, the following parameters are used. C_k represents the capacity of bin type k. V_i represents the volume of item i. These parameters define the storage limits of bins and the space requirements of items, respectively.

The decision variables are as follows. x_{ij} is a binary variable and takes 1 if item i is stored in bin j; 0, otherwise. y_{jk} is a binary variable and takes 1 if used bin j is of type k; 0, otherwise.

Furthermore, the mathematical model operates with the fact that the items are significantly smaller than the bins, and none of their individual dimensions exceed the bin dimensions. This ensures that all items can fit within the bins in 3D space. As such, the model focuses exclusively on volume optimization, aligning with the practical constraints of the study.

$$\min \sum_{j=1}^{m} \sum_{k=1}^{r} C_k \cdot y_{jk} \tag{1}$$

Subject to:

$$\sum_{j=1}^{m} x_{ij} = 1 \text{ for } i = 1, \dots, n \tag{2}$$

$$\sum_{k=1}^{r} y_{jk} \leq 1 \text{ for } j = 1, \dots, m \tag{3}$$

$$\sum_{i=1}^{n} V_i \cdot x_{ij} \leq \sum_{k=1}^{r} C_k \cdot y_{jk} \text{ for } j = 1, \dots, m \tag{4}$$

$$x_{ij} \in \{0,1\} \text{ for } i = 1, \dots, n \text{ and } j = 1, \dots, m \tag{5}$$

$$y_{jk} \in \{0,1\} \text{ for } j = 1, \dots, m \text{ and } k = 1, \dots, r \tag{6}$$

The objective (1) is to minimize the sum of the multiplication of the utilized bins and their corresponding capacities. This results in minimizing the total utilized capacities, thereby optimizing storage space usage. Constraint (2) ensures that each item is assigned to exactly one bin. This guarantees that all items are stored without duplication. Constraint (3) ensures that each bin can only be at most one bin type. Constraint (4) ensures that the total volume of stored items in a bin cannot exceed its capacity. This ensures adherence to the physical limits of the bins. Constraints (5) and (6) are binary restrictions for all decision variables.

To clarify the boundaries of the model, the following assumptions are made: the model assumes that all items must be assigned to exactly one bin, bins of each type have fixed capacities (C_k) defined in advance, and the total number of bins (m) and bin types (r) are finite and predetermined. The focus is solely on storage space optimization, excluding other operational constraints like bin costs.

After formulating the mathematical model for the problem, some small size problems are solved by using IBM ILOG CPLEX. Then, we progressively increased the problem size until reaching the real-scale scenario for the problem at hand.

Table 2 provides the results regarding the number of items (n), the number of bins (m), the number of bin types (r), the number of decision variables (x_{ij}, y_{jk}) and the running times.

The execution times vary depending on the problem size, with smaller instances being solved relatively quickly while larger instances may require significantly more time to find an optimal solution. This highlights the time complexity of the problem, which can increase exponentially as the problem size grows due to the NP-Hardness of VSBPP.

Table 2. Problem Size and Average Running Time

n	m	r	x_{ij} (Count)	y_{jk} (Count)	Average running time (hours)
10	10	3	100	30	00:00:01:43
20	20	3	400	60	00:00:01:65
30	30	3	900	90	00:00:01:53
50	50	3	2,500	150	00:00:02:12
80	80	3	6,400	240	00:00:02:77
100	100	3	10,000	300	00:00:04:75
150	150	3	22,500	450	>24

4.2. Heuristic Algorithms

In general, the running time for solving a mathematical model may take an unreasonable time as the problem size gets larger if it is NP-Hard. To tackle this problem, implementing a heuristic algorithm is a good alternative that allows us to efficiently solve large instances of the VSBPP within a reasonable running time.

In the literature, several heuristic methods have been utilized to address the VSBPP. Moreover, VSBPP has been solved using metaheuristic algorithms such as tabu search and genetic algorithm.

Heuristic algorithms to solve VSBPP include (Coffman, et al., 1984):

i. *First Fit Decreasing (FFD):* This algorithm tries to place each item into the first bin that has enough space for it. If no bin has enough space, a new bin is created, and the item is placed in the new bin.

ii. *Next Fit Decreasing (NFD):* Like the First Fit Decreasing algorithm, the Next Fit Decreasing algorithm begins packing items into the next bin after the current one rather than the first bin. The algorithm moves to the next bin and tries to fit the item there if it does not fit in the current bin. The item is placed in the newly created bin if the item still does not fit the bin.

iii. *Best Fit Decreasing (BFD):* This algorithm looks for a bin that can accommodate the current item with the smallest amount of leftover space. If no bin has space for the item, a new bin is created, and the item is put in it.

For VSBPP, Figure 2 shows the pseudocode of the FFD algorithm, whereas Figure 3 shows the pseudocode of the NFD algorithm. Figure 4 shows the pseudocode of the BFD algorithm.

Dökeroğlu (2017) implements FFD and BFD algorithms for one-dimensional BPP in his study.

The VSBPP was addressed by Haouari and Serairi (2009) through the utilization of multiple methods, including FFD, BFD, a set covering heuristic, and genetic algorithm. Notably, the results obtained from FFD and BFD were found to be satisfactory.

Kang and Park (2003) conducted a research on the VSBPP and proposed two algorithms known as iterative FFD (IFFD) and iterative BFD (IBFD). The authors asserted that these modified versions of the original FFD and BFD algorithms effectively address the VSBPP. The approach of IFFD involves initially assigning all items to the largest size bins using FFD, resulting in a feasible solution. Subsequently, the items in the last bin of the solution are repacked into the next largest bins using FFD, leading to another solution. This process continues until repacking becomes unfeasible, generating multiple feasible solutions. Among these solutions, the best one is chosen as the final solution, with potential additional modifications if required. Similarly, IBFD follows a similar procedure but employs the BFD algorithm instead of FFD.

```
Sort items in decreasing order based on their sizes
   Initialize an empty list of bins

   for each item in items:
       binFound = False
       for each bin in bins:
           if item can fit in bin:
               Add item to bin
               binFound = True
               break

       if binFound is False:
           Create a new bin
           Add item to the new bin

   return bins
```

Figure 1. Pseudocode of FFD Algorithm

```
Sort items in decreasing order based on their sizes
   Initialize an empty list of bins

   for each item in items:
       binFound = False
       for each bin in bins:
           if item can fit in bin:
               Add item to bin
               binFound = True
               break

       if binFound is False:
           Create a new bin
           Add item to the new bin

   return bins
```

Figure 2. Pseudocode of NFD Algorithm

```
Sort items in decreasing order based on their sizes
  Initialize an empty list of bins

  for each item in items:
      minSpace = infinity
      selectedBin = None

      for each bin in bins:
          if item can fit in bin and bin's available space is less than minSpace:
              minSpace = bin's available space
              selectedBin = bin

      if selectedBin is not None:
          Add item to selectedBin
      else:
          Create a new bin
          Add item to the new bin

  return bins
```

Figure 3. Pseudocode of BFD Algorithm

The drawback of the FFD, BFD, and NFD lies in their reliance on identical bin sizes, which is not applicable to the VSBPP. Consequently, some researchers have endeavored to modify these algorithms to accommodate variable-sized bins, as exemplified by the Iterative FFD algorithm developed by Kang and Park (2003). Therefore, employing these algorithms directly -without any modifications- would not be suitable for addressing our specific problem. So, some modifications are required. Research on the VSBPP is limited when compared to the bin-packing problem, with the majority of studies focused on one-dimensional events. Common issues include the creation of lower bounds and solution strategies. VSBPP, an NP-hard issue, has been handled using both accurate and efficient approximation methods, with the latter category comprising heuristic and metaheuristic approaches (Borgulya, 2024).

In order to test the algorithms' effectiveness, the dimensions of the SKUs associated with the warehouse at hand were taken into consideration along with the bin types. In the case of having different SKUs, however, simple modifications need to be done to make for the change of the new dimensions in order for the algorithm to run effectively. All algorithms are implemented in Python. For instance, a partial depiction of the obtained FFD results for small bin types can be observed in Figure 5. The implementation of the FFD algorithm yielded highly favorable outcomes for the small items, with an impressive average utilization of 98.4% across all bins. Notably, the storage of 314 items required using 106 small bins.

```
Bin 99 : 243, 245, 246, 247, 273        Utilization: 99.62%
Bin 100 : 248, 249, 251, 252, 258       Utilization: 99.74%
Bin 101 : 253, 254, 255, 257, 259       Utilization: 98.37%
Bin 102 : 260, 261, 262, 265, 266       Utilization: 92.67%
Bin 103 : 267, 269, 270, 274, 275, 295  Utilization: 99.92%
Bin 104 : 277, 279, 280, 283, 284, 286  Utilization: 92.78%
Bin 105 : 287, 288, 292, 293, 298, 305, 307     Utilization: 93.89%
Bin 106 : 308, 309, 310, 311, 312, 313, 314     Utilization: 81.03%
Average utilization of all bins: 98.40%
Number of bins used: 106
```

Figure 5. Partial Depiction of FFD Algorithm Results for the Small Bin Type

To align with the requirements of the MRO warehouse, the standard FFD, BFD, and NFD algorithms were modified. Specifically, three FFD algorithms were implemented in a single code, each addressing a distinct bin size (large, medium, small). Furthermore, a constraint limiting the number of items per bin to 7 was introduced to ensure efficient retrieval times. These modifications were guided by practical observations during preliminary testing, ensuring the algorithms effectively balance space utilization and operational efficiency.

The Python code for the implementation of FFD, NFD, and BFD algorithms has been made publicly available on GitHub for transparency and reproducibility. The repository includes detailed comments and documentation to help readers understand and replicate the methodology. The code can be accessed via the following link:

https://gist.github.com/faris118203/600e15c76ad47325303403fa418a536e

For context, the code comprises three separate FFD algorithms, each tailored for a specific bin type. It begins by sorting all items in descending order based on their weights. Additionally, a constraint has been introduced to limit the maximum number of items in a single bin to 7, determined based on preliminary studies to optimize retrieval time efficiency.

The outcomes derived from the code partially displayed in Figure 6 revealed that a total of 1,169 items were successfully stored across 511 bins. In Figure 6, the overall utilization percentage of all bins amounted to 95.25%, indicating an efficient utilization of available storage space. The sum of the capacities of the used bins (overall used space) equated to 1,520,421 cm^3, which was computed by multiplying the number of bins used for each bin type by its respective capacity. Furthermore, the code yielded these impressive results within a remarkably short duration of around 1 second.

```
Total number of bins used: 511
Number of bins used of capacity 4641 cm3: 157
Number of bins used of capacity 2808 cm3: 248
Number of bins used of capacity 900 cm3: 106
Overall used space: 1520421 cm3
Average utilization for bins with capacity 4641 cm3: 93.84 %
Average utilization for bins with capacity 2808 cm3: 94.79 %
Average utilization for bins with capacity 900 cm3: 98.40 %
Overall average utilization: 95.25 %
```

Figure 6. Average Utilization Results for FFD for All Bin Types

This study complies with scientific research and publication ethics and principles.

5. Computational Results

Table 3 provides a summary of the sum of the used bin capacities, overall average utilization, and the number of used bins for each algorithm.

Table 3. FFD, BFD, and NFD Algorithms' Results

Algorithm	Total Number of Used Bins	The Sum of the Used Bins' Capacities (cm^3)	Average Utilization
FFD	511	1,520,421	95.25 %
BFD	511	1,520,421	95.25 %
NFD	631	1,859,097	77.14 %

Notably, FFD and BFD algorithms both exhibited the most favorable outcomes, boasting an impressive average utilization percentage of 95.25% across all bin types. Furthermore, the total capacity of the used bins amounted to 1,520,421 cm^3, indicating a highly efficient utilization of available storage space.

When comparing the FFD and BFD algorithms' results with the current utilized capacity inside the company's warehouse, the results turn out to be rather impressive. The items inside the warehouse are currently distributed across 762 bins with three different bin types. The total utilized capacities of the currently used bins are 1,940,472 cm^3. The FFD and BFD algorithms' results indicate that the items will be stored across 511 bins. Moreover, the total utilized capacities needed to store all items will be 1,520,421 cm^3. This leads to a decrease of 420,051 cm^3 which is around 21.6% of the total used storage capacities. This number is equivalent to around (420,051 / 4,641) = 90 of the large bins, 149 of the medium bins, and 466 of the small bins in the warehouse.

The efficiency and practicality of the proposed algorithms were evaluated using three performance metrics: optimality gap, computation time, and solution quality. Table 4 provides a detailed comparison of the results obtained for FFD, BFD, and NFD algorithms.

Table 4. Performance Metrics for FFD, BFD, and NFD Algorithms

Algorithm	Optimality Gap (%)	Running Time (sec)	Solution Quality (%)
FFD	2.56	1.25	95.25
BFD	2.75	1.48	95.25
NFD	8.24	0.96	77.14

The results indicate that the FFD and BFD algorithms achieve superior solution quality compared to NFD, albeit with slightly higher computation times. These findings are consistent with the literature, where similar trends have been observed (Hemmelmayr et al., 2012). The gap values demonstrate the near-optimal performance of FFD and BFD, making them suitable for practical applications in real-world warehouse management.

6. Conclusion

Effective warehouse management is crucial for companies in the warehouse industry, offering significant cost and space savings, improved operational efficiency, and increased profitability. Ongoing optimization of warehouse operations, including inventory management and storage arrangements, leads to streamlined workflows and accurate inventory tracking. Efficient storage systems maximize space utilization, eliminating the need for additional warehouses and associated costs.

The aim of this study is to find an efficient solution that minimizes the total capacity of the boxes used and the number of boxes used, thus optimizing the company's storage policy and saving space in the warehouse. The problem under consideration is an NP-Hard problem. Foremost, three-dimensional measurements were taken for thousands of SKUs in the warehouse, and a detailed ABC analysis was performed. Subsequently, we focused on A and B-class items. Afterwards, Hemmelmayr et al.'s (2012) mathematical model was adapted to the problem under consideration. The problem was solved by increasing its size gradually and trying to obtain optimum results through computational experiments. However, it has been shown that as the problem size gets larger, the problem cannot be solved in a reasonable time, which supports the NP-Hardness of the real problem size. Therefore, First Fit Decreasing (FFD), Best Fit Decreasing (BFD), and Next Fit Decreasing (NFD) heuristic algorithms, which do not guarantee the optimal result but are known to give good results in a short running time, are considered. This means that this study can be used as a reference for implementing the FFD, BFD, and NFD inside warehouses that face the same problem of having to store items inside variable-sized bins as opposed to storing the items in only single-sized bins (which was the original aim of designing these algorithms) by slightly modifying the code implemented to match the needed dimensions.

The results obtained from the computational experiments showed that FFD and BFD outperformed the NFD algorithm with the same performance, and all three algorithms were successful in reducing storage space usage and increasing space utilization compared to the existing warehouse practice.

The scalability of the proposed approach can be extended to larger warehouses by employing distributed algorithms or parallel computation techniques. These techniques would enable the efficient handling of larger datasets and more complex configurations without significantly increasing computation time.

For dynamic inventory conditions, where items frequently arrive and leave, adaptive heuristic algorithms can be integrated into the existing framework. These algorithms would dynamically adjust bin allocations based on real-time data, ensuring that storage space utilization remains efficient under changing inventory demands.

Future work could explore these enhancements, including testing the algorithms under varying warehouse scales and dynamic conditions, to further validate their robustness and applicability in diverse operational environments.

Declaration of Competing Interest

The authors have no conflicts of interest to declare regarding the content of this article.

Authorship Contribution Statement

Faris ALBAKAR: Conceptualization, Methodology / Study design, Software, Validation, Investigation, Resources, Data curation, Writing – original draft,

Mhd Hazem ALHAMMAMI: Conceptualization, Methodology / Study design, Software, Validation, Investigation, Resources, Data curation, Writing – original draft,

Husam Jehad Hasan OMAR: Conceptualization, Methodology / Study design, Software, Validation, Investigation, Resources, Data curation, Writing – original draft,

Mazen SIMI: Conceptualization, Methodology / Study design, Software, Validation, Investigation, Resources, Data curation, Writing – original draft,

Basel ZBEDA: Conceptualization, Methodology / Study design, Software, Validation, Investigation, Resources, Data curation, Writing – original draft,

Zehra DUZGIT: Conceptualization, Validation, Writing – review and editing, Supervision

References

Aarts, E., & Korst, J. (2003). Simulated Annealing and Boltzmann Machines. Wiley.

Alenezi, Q., Aboelfotoh, H., Albdaiwi, B., & Almulla, M.A. (2015). Heuristics for the Variable Sized Bin Packing Problem Using a Hybrid P-System and CUDA Architecture. Computer Science Department, Kuwait University. Retrieved from: https://arxiv.org/abs/1602.08735

Borgulya, I. (2024). A hybrid estimation of distribution algorithm for the offline 2D variable-sized bin packing problem. Central *European Journal of Operations Research*, 32, 45–65. https://doi.org/10.1007/s10100-023-00858-0

Boyar, J., Kamali, S., Larsen, K.S., & López-Ortiz, A. (2013). Online Bin Packing with Advice. *Algorithmica*, 74, 507-527. doi: https://doi.org/10.48550/arXiv.1212.4016

Coffman Jr, E.G., Garey, M.R., & Johnson, D.S. (1984). Approximation Algorithms for Bin-Packing — An Updated Survey. Algorithm Design for Computer System Design. Editors: Ausiello, G., Lucertini, M., & Serafini, P., Vienna, Springer, 49-106. Retrieved from: https://link.springer.com/chapter/10.1007/978-3-7091-4338-4_3

Dökeroğlu T. (2017). Bir boyutlu kutulama probleminin eniyilenmesi için hiper-sezgisel paralel bir algoritma, *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 19(1), 1-11. doi: https://doi.org/10.25092/baunfbed.319992

Dyckhoff, H. (1990). A typology of cutting and packing problems. *European Journal of Operational Research*, 44, 145-159. doi: https://doi.org/10.1016/0377-2217(90)90350-K

Grondys, K. (2009). ABC Analysis in Spare Parts Warehouse. *Polish Journal of Management Studies*, 3, 147-156. Retrieved from: https://econpapers.repec.org/article/pczjournl/v_3a3_3ay_3a2009_3ai_3a1_3ap_3a147-156.htm

Gupta, A., Guruganesh, G., Kumar, A., & Wajc, D. (2018). Fully-Dynamic Bin Packing with Limited Repacking. ArXiv, abs/1711.02078. Retrieved from: https://doi.org/10.48550/arXiv.1711.02078

Haouari, M., & Serairi, M. (2009). Heuristics for the variable sized bin-packing problem. *Computers & Operations Research*, 36, 2877-2884. doi: https://doi.org/10.1016/j.cor.2008.12.016

Hemmelmayr, V., Schmid, V., & Blum, C. (2012). Variable neighbourhood search for the variable sized bin packing problem. *Computers & Operations Research*, 39, 1097-1108. doi: https://doi.org/10.1016/j.cor.2011.07.003

Jin, Z., Ito, T., & Ohno, K. (2003). The Three-Dimensional Bin Packing Problem and Its Practical Algorithm. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 46, 60-66. doi: https://doi.org/10.1299/jsmec.46.60

Kang, J., & Park, S. (2003). Algorithms for the variable sized bin packing problem. *European Journal of Operational Research*, 147, 365-372. doi: https://doi.org/10.1016/S0377-2217(02)00247-3

Karp, R.M. (1992). On-line algorithms versus off-line algorithms: How much is it worth to know the future?. International Computer Science Institute, Berkeley, California, Technical report, TR-92-044. Retrieved from: https://www1.icsi.berkeley.edu/pubs/techreports/TR-92-044.pdf

Lodi, A., Martello, S., Monaci, M., & Vigo, D. (2013). Two-Dimensional Bin Packing Problems. Editor: Paschos, V. T. Paradigms of Combinatorial Optimization, 107-129, Wiley, GB. Retrieved from https://onlinelibrary.wiley.com/doi/10.1002/9781118600207.ch5

Martello, S., & Vigo, D. (1998). Exact Solution of the Two-Dimensional Finite Bin Packing Problem. *Management Science*, 44, 388-399. Retrieved from: https://www.jstor.org/stable/2634676

Pinto, A. R. F., Nagano, M. S., & Boz, E. (2023). A classification approach to order picking systems and policies: Integrating automation and optimization for future research. *Results in Control and Optimization*, 12, 100281. doi: https://doi.org/10.1016/j.rico.2023.100281

Tompkins, J.A., & Smith, J.D. (1998). The Warehouse Management Handbook. Tompkins Press, Raleigh, North Carolina.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183, 1109-1130. doi: https://doi.org/10.1016/j.ejor.2005.12.047