A new heuristic algorithm to improve the design of a vertical storage system

Bertolini M.*, Neroni M.*, Romagnoli G.*

**Dipartimento di Ingegneria e Architettura – Università di Parma, Viale G.P. Usberti 181/A, 43124, Parma – Italy (mattia.neroni@studenti.unipr.it)*

Abstract: In this paper, a new heuristic algorithm to support the design phase of a vertical storage system is presented. The considered vertical storage system is made up of racks accommodating metal containers which are arranged on the two sides of the lift device that feeds them. The proposed heuristic algorithm has been developed combining two well-known problems: Bins Packing Problem (BPP) and Rectangular Nesting Problem (RNP).
Given a list of products that must be stored, the dimensions of the warehouse's racks and its load capacity, the developed algorithm allows to obtain the list of products that must be placed in each stock keeping unit and returns the right position and orientation any item should take. The algorithm provides the minimum number of racks with height dimension as low as possible. This aim is due to the common interest of automated vertical storage systems designers or owners in lowering costs, which trend to grow up with racks number and height. Furthermore, the position of the items inside each rack is managed to optimize the volume exploitation and to balance the container distributing the weight inside it. The whole procedure also regards the maximum weight constrain that basically limits the filling of loading units.
The robustness of the proposed algorithm has been studied simulating different scenarios, by changing boundary conditions such as the number of items to be stored, their middleweight, their average size and the variance of these physical characteristics. Finally, the algorithm has been applied in a realistic situation to support automated vertical storage system design aimed at holding in stock metal moulds.

Keywords: bin packing problem, nesting problem, storage system, vertical warehouse, automated warehouse, stock keeping units, heuristic algorithm, Italy.

1. Introduction

A vertical storage system based on racks is usually dubbed as Vertical Lift Shuttles Storage Systems (VLSSS) or Vertical Lift Modules (VLM). Its structure consists of two metal stock keeping units, vertically arranged, a delivery lift platform, and a computerized control. The operator is called to wait for picking units standing in front of an inbound/outbound area or an ergonomic workstation (Roy Jr., 1997).
Automated storage systems, as well as stock intensive systems, play an important role in the process of logistic digitalization, which is taking place in the domain of what the German Government called "Industry 4.0". New trends lead to smaller batch sizes and to an increasing number of transportation and picking processes (Wilke, 2006). Automated storage systems ensure high flexibility and provide advantages like zero error strategy or time optimized applications. Management of big data is also a problem whose solution lies in the design of autonomous, decentralized controlled material handling systems (Borgi, Zoghlami, & Abed, 2017).
VLMs are usually designed as a single module composed by two vertical racks; both racks accommodate their stock units along a column and both columns are served by the lift elevator that runs along the middle vertical aisle. Vertical lift shuttles storage systems bring parts to the operator on heavy-duty storage trays or container boxes, storing up to 5,000 kilograms per storage unit. This kind of storage system was commonly used in pharmaceutical industry up to a few years ago. Later, it has been verified that vertical storage systems entail other benefits such as a better surface exploitation and an ergonomically-positioned workstation for inbound/outbound material management (Schwind, 1995). Furthermore, a vertical disposition of materials decreases handling and energy costs (Meneghetti, Dal Borgo, & Monti, 2015). Because of these advantages, VLMs have become popular and their implementation has been extended to other market segments, such as stocking screw machine products, metal moulds or wire metal drawings up to 6 meters long.
The storage systems have become bigger, but the advantages due to their implementation are still the same. Vertical storage systems provide a popular, easy-to-justify material handling equipment. They eliminate the need for new constructions and increase the productivity even operating with fewer staff (Schwind, 1995).
They use smart technology to provide both modularity and flexibility, and may store and retrieve a huge variety of components (Roy Jr., 1997). They provide fast delivery of common inventory items because they monitor inventory usage and locate fast-moving items closer to the access opening and slow-moving parts further away from the access opening. Vertical storage systems turn overhead airspace into productive storage space. Their small footprints make them a perfect choice for point-of-use

storage to free up floor space for value-added manufacturing and assembly operations. VLMs can be integrated to provide a comprehensive, fully automated storage and retrieval system (AS/RS) and their technology perfectly fits with other handling systems such as traveling cranes.
Automated vertical storage systems are actually automated storage and retrieval systems (AS/RS) and exactly as in AS/RS design it is important to find out the optimal racks dimensions to minimize cycle time and costs (Yu & De Koster, 2009). The rack's shape affects performance and an effective trade-off between width, height, and length can minimize travel times (Bozer, Y.A. and White, 1996; De Koster, Le-Duc, & Yugang, 2008). However, in VLMs systems width and length are fixed and the design problem could be reduced to height definition.
Moreover, innovative models for energy calculation (Meneghetti et al., 2015) introduced new factors that were previously neglected, such as the units load weight and differentiation of shifts along the horizontal and vertical axis with respect to required energy consumption due to the different contribution of gravity, inertia, friction, speed mutations, acceleration and motor angular speed. Results outcome from innovative models described above prove that a minimum energy consumption can be recognized for intermediate racks. A middle-class rack can reach a better arrangement of vertical and horizontal shifts, with a proper balance of gravity and inertia. Travel time, instead, increases with rack height (Meneghetti et al., 2015). So, energy pattern is known to be U-shaped for growing system height, while travel time grows with system height (Meneghetti et al., 2015). That's why, in an automated vertical storage, where the biggest dimension that could influence the performance is the system and containers height, it would be important to keep it down, because this decision would reduce manufacturing costs (e.g. cost of materials) and operating costs (e.g. shorter travel times and lower energy costs). The solution is achievable with a strategy that allows a smart items allocation. In this paper, we propose a new heuristic algorithm to improve the design of an automated vertical storage system. Given a list of items to be stored and their characteristics, the aim of the algorithm is to set out the best combination of stock keeping for the VLM system. The algorithm is a combination of well-known Bin Packing Problem (BPP) and Rectangular Nesting Problem (RNP) applied to stock keeping units loading. Although both problems have a significant presence in the scientific literature, their combination has been rarely considered up to now. Furthermore, several studies from the literature analyse the problem from a mathematical point of view without creating a relationship model to make geometrical approximations of handled items reflecting on their shape, position, and orientation in space instead of direct placement.
The reported algorithm is also constrained by the mass capacity of the stock units, and it operates to balance it.
The remainder of this paper is organized as follows: a literature review of BPP and RNP is reported in Section 2, and the proposed algorithm is described in Section 3. Section 4 illustrates numerical analysis, and Section 5 draws conclusions and suggests future directions for research.

2. Literature review

In the Bin Packing Problem, or BPP, (Coffman, Leung, & Ting, 1978), objects of different volumes, sizes or shapes must be packed, into a finite number of bins or containers of fixed width, in a way that minimizes the number of bins used. This aim could be reached maximizing the number of items assigned to each bin. In computational complexity theory, it is stated as a combinatorial NP-hard problem and the decision problem (deciding if objects will fit into a specified number of bins) is NP-complete. There are many variations of this problem, such as 2D packing (Lam, 2017; Lodi, Martello, & Monaci, 2002), linear packing (Hiroyuki Okano, 2002), packing by weight, packing by cost, and so on. Many approximation algorithms have been invented over the past decades, some of which are already very close to optimality in practice. Nonetheless, researcher's interest in this problem always remains so strong that it is still intensively researched since every little improvement can generate enormous economic values in mass-production industries (Lam, 2017).
The 2D-BPP (Lodi et al., 2002) is the combinatorial optimization problem most suitable in the case we consider and concerns with allocating multiple objects into rectangle bins of known dimensions (as in the case under analysis). Its application domain could be extended to many areas such as logistic, manufacturing or transportation.
Interesting solutions have already been reported in the literature (Berkey & Wang, 1987; Grange, Kacem, & Martin, 2018; Murgolo, 1988). A recent study also considers the possibility to change the object orientation (Ma & Zhou, 2017). However, most of the algorithms that can be found in the literature do not consider: (i) the three-dimensionality of handled objects, (ii) their height, (iii) their mass, which needs to be well-distributed along all the box surface to prevent any spill or instability during the movements. Even the possibility to reallocate the objects assigned to a container to manage the space inside it is an ignored topic.
In this specific case, a research operative problem which is stated Rectangular Nesting Problem (RNP) or Strip Packing Problem (SPP) (Kenmochi, Imamichi, Nonobe, Yagiura, & Nagamochi, 2009), repurposed as optimizer algorithm, can improve results. Nesting problem is a two-dimensional cutting problem where the shapes of the pieces to cut and the master surfaces are irregular in shape and different in size (Baldacci, Boschetti, Ganovelli, & Maniezzo, 2014). If pieces have a rectangular shape and fixed width, the problem is claimed strip packing problem. The purpose of the problem is usually the minimization of waste and it is obtained by improving the disposition of shapes lying on a cut surface.

3. The proposed algorithm

Given a list of parallelepiped-shaped items to stock inside an automated vertical storage system, the algorithm finds out the arrangement that minimizes the number of containers requested to optimize performance and minimize costs.

Firstly, items are sorted by decreasing height. Then, beginning at the top of the list, they are placed in containers using a procedure that is similar to the Finite Bottom Left (Berkey & Wang, 1987). To verify if an object can fit in surrounding area algorithm approximates input items and bins by scanlines, and hands the two-dimensional BPP as a variant of one-dimensional BPP (Hiroyuki Okano, 2002). Moreover, before placing the object, the algorithm checks the exposed perimeter (the exposed perimeter is defined as the object's perimeter that is not in touch with container's borders or other objects) in both possible orientation, that means performing a 90° rotation on the z-axis, and it selects the orientation with the minimum exposed perimeter (Ma & Zhou, 2017).
The optimizer relocates objects inside the same container and switches items between different containers to further optimize surface exploitation trying to empty the container with lowest volumetric exploitation.
In literature, there are many algorithms that extend the problem of allocation to irregularly shaped objects (Egeblad, 2009; Litvinchev & Mosquera, n.d.). the proposed solution just considers parallelepiped shaped items. This is not a limit, because the algorithm would provide a solution even with that kind of items, the only need is the inscription of irregular objects in rectangles before algorithm running.
It is important to note that the algorithm does not consider any joint objects management, that would reduce travel time in picking missions.

The inputs closely linked to storage system's racks and stock keeping units are the following:

- unlimited usable number of rectangular shaped stock keeping units;
- fixed width of the stock unit basement;
- fixed stock unit weight capacity;
- the maximum number of different heights (to limit containers customization).

The inputs linked to object to place are the following:

- fixed number of parallelepiped shaped items;
- objects dimensions randomly decided;
- objects mass randomly decided;
- objects priority level randomly defined.

Algorithm constraints consist in the fact that objects can spin on the basement, but their height is fixed, so, basically, objects can't lie on their side.

The aim of the algorithm consists in the exploitation of minimum containers number. Furthermore, containers mass must be balanced along their surface. Containers height is fixed by algorithm looking at highest object stowed inside.
Firstly, the heuristic sorts the object by descending height, as this operation provides a better volume exploitation (Murgolo, 1988). Objects of equal height are sorted by descending mass and, lastly, objects characterized by equal mass and height are sorted by increasing priority rule.
Then the heuristic works as follows:

(1) For each object in the list (beginning from top).
(2) Select the first container that has already been used.
(3) Select first possible object orientation.
(4) Beginning from corner, look for an empty area.
(5) If the area is enough big to accommodate the object accommodate it and check the exposed perimeter and go ahead; if the empty area is not big enough go ahead and look for another empty area.
(6) Change orientation and go back to 4.
(7) If both orientations involve a possible placement, select that one which entire the smaller exposed perimeter of the considered object and go to 9.
(8) If at least an allocation using one of the possible orientations is possible, go ahead;
If it's not and the container isn't the last one used, take next container and go to 4;
If it's not and the container is the last one used, open a new container, place the object and go to 9.
(9) If object placed was not the last, take next object and go to 2.

Figure 1 shows the solution provided by the heuristic. In detail, a plan view of five containers after implementation of the proposed heuristic algorithm is reported. In the example reported, 150 objects (red, yellow, green and orange colours) have been placed in 5 containers. Blue colour represents empty spaces in each container.

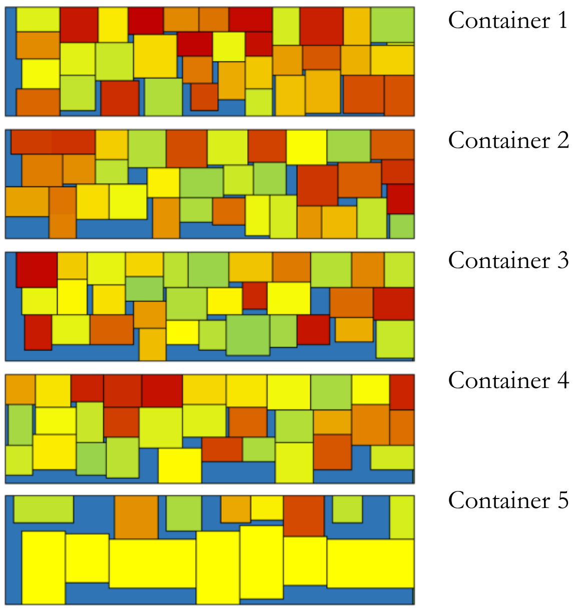

Figure 1 – Plan view of a solution provided by the heuristic.

The goal of optimizer algorithm is to redistribute the objects removing them from the container characterized by lowest volumetric exploitation. This action will improve the filling, and, eventually, it may empty the last container.
Essentially, the optimizer consists of 3 different loops.
The first loop simply tries to move objects from the emptiest container to others just to double-check what heuristic did.
The second loop, which we called "blind switch", moves items with no other items on their side from the middle to the borders of a container where they have been allocated. This action helps to equally distribute mass inside stock units, remove mass from the middle of stock units (which

is usually the weakest part), creates new empty areas of different shapes increasing the possibility to place there one of the items in the emptiest container.
The third loop, that we called "intelligent switch", take an object from the emptiest container, looks at into others for a smaller object, provided with empty space around it and switch two objects to improve the solution.

The optimization algorithm steps work as follows:

- FIRST LOOP

(1) Select emptiest container.
(2) Take the first item inside the selected container.
(3) Select first possible orientation.
(4) Try to plug it into other containers.
(5) If any space hasn't been found change orientation and goes to 4, else go ahead.
(6) If the item selected was the last one STOP, else take next item in the selected container and go to 3.

- SECOND LOOP

(7) Select first container different from emptiest one
(8) Beginning from vertex from which heuristic began filling, move every object, that is not obstructed by others and it is not in touch with container's edges, in a direction perpendicular to the long edge as represented in Figure 2.
(9) Beginning vertex from which heuristic began filling, move every object in a direction perpendicular to the short edge as represented in Figure 3.
(10) Repeat FIRST LOOP to try to transfer an item from the emptiest container to the selected one.
(11) If containers different from emptiest one ended up STOP, else select next container and go to 8.

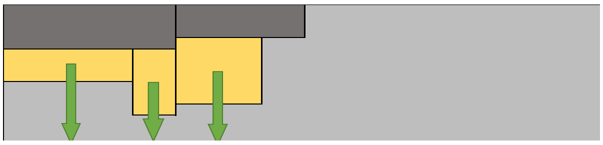

Figure 2 – Representation of event described in step 8

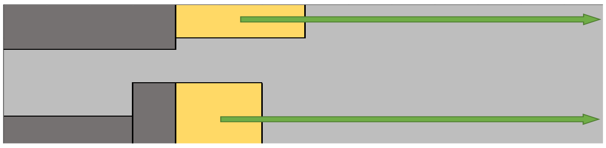

Figure 3 – Representation of event described in step 9

- THIRD LOOP

(12) Select the first item in the emptiest container and check its surface (S1).
(13) Select first container (C2) different from emptiest one.
(14) Check the surface (S2) of the first item in the selected container.
(15) If S2 is bigger than S1, then go to 16;
If S1 is bigger than S2 and there's empty space around S2, then try to place bigger item instead of smaller one. If placing is possible, switch their position and go to 17, else go to 16.
(16) If items in C2 ended up go ahead, else select next item in C2 and go back to 15.
(17) If containers different from emptiest one ended up go ahead, else select next container different from emptiest one and go to 14.
(18) If items in emptiest container ended up repeat FIRST LOOP to place small items in the last container in empty spaces and STOP, else select next item, check its surface and go back to 13.

The following is an example to better explain how the third loop of the optimization algorithm works. In Figure 4 the container number 2 is that one which algorithm tries to empty. It is possible to understand the impossibility to place object B in any empty space such as C. The algorithm firstly switches B with a smaller object with an empty area around such as A (Figure 5). After this barter, it is possible to figure out how surface exploitation of container 1 has been improved. Additionally, with a final repetition of the first loop, it is possible to place object A in empty space C as shown in Figure 6 to further improve the solution.

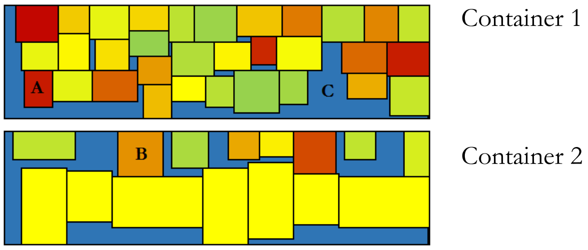

Figure 4 – starting point - before applying the optimizer algorithm.

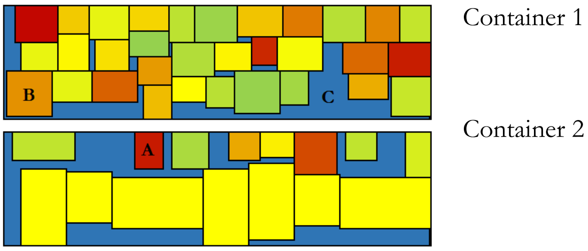

Figure 5 – Intermediate point - switch between two items.

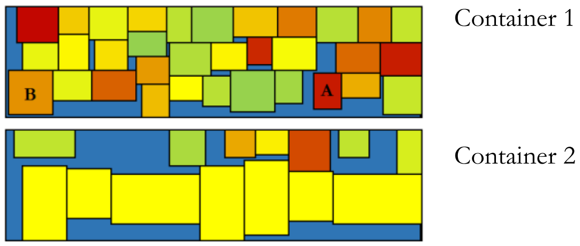

Figure 6 – Final point - after implementation of the optimizer algorithm.

4. Numerical analysis and implementation

To collect and analyse results the algorithm has been tested in several iterations with a different number of objects, whose dimensions float between 5 and 80 centimetres. In the experiment, containers considered were 3 meters long and 0.8 meters large.
The algorithm has also been tested in two different situations. In the first one, the container mass capacity constraint has been relaxed to check the actual volume and surface saturation. In the second situation, containers mass capacity has been reduced a lot to check the actual saturation in terms of mass.
The surface exploitation efficiency parameter results independent from objects dimensions variance: results obtained allocating very different objects are as well as those obtained placing similar objects. The same thing cannot be declared talking about volumetric exploitation, because the highest object defines container height, and, in this case, if other objects are lower than this, volumetric exploitation degenerates.

In 16.05% of all runs, the number of containers requested by the algorithm was equal to a minimum needed number of containers (Figure 7). The concentration of these optimal results was bigger when the objects managed were smaller.

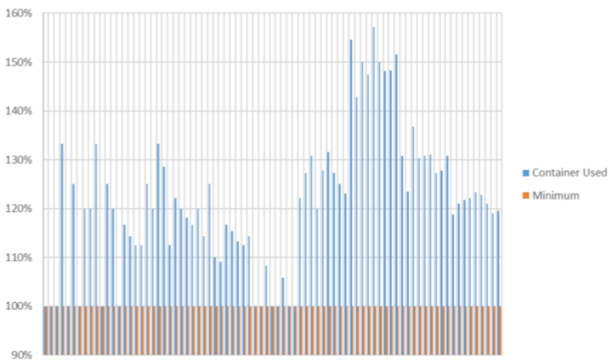

Figure 7 - Comparison between containers used and minimum request

The surface and volumetric exploitation levels, obtained by relaxing the mass capacity constraint, are expressed in Table 1.

Table 1: Algorithm results in surface and volume saturation

	Max	Medium	Min
Surface	99.9%	82.7%	64.3%
Volume	99.1%	80.5%	63.8%

The development of volumetric exploitation related to objects dimensions increase is represented in Figure 8.
While the trend of surface exploitation, related to objects dimensions increase is represented in Figure 9.

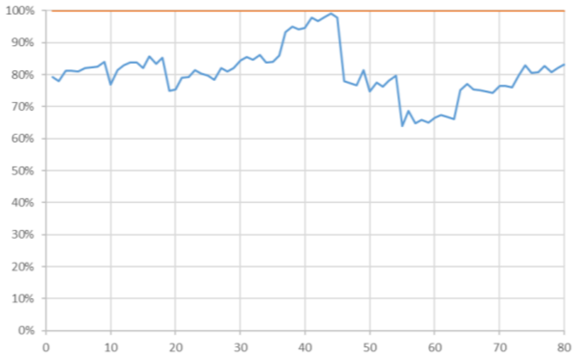

Figure 8 – On y-axis the volume exploitation that algorithm provided during the tests

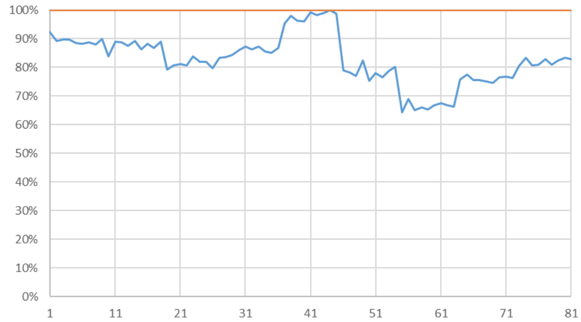

Figure 9 – On x-axis the surface exploitation that algorithm provided during the tests

Instead, containers saturation in terms of mass is expressed in Table 2.

Table 2: Algorithm results in mass saturation

	Max	Medium	Min
Mass	99.1%	80.6%	63.9%

5. Conclusion and future research directions

In this paper, a new heuristic algorithm to improve the process of design of an Automated Vertical Storage System is proposed. The algorithm is based on the combination of two well-known algorithms such as BPP and RNP. The proposed algorithm could be generalized and applied to many others problem in the field of logistic, transportation, and manufacturing too. Our future remarks consist in the implementation of joint object's management, because keeping closed objects that usually move or are picked together, would increase the number of picking missions per unit of time.

6. References

Baldacci, R., Boschetti, M. A., Ganovelli, M., & Maniezzo, V. (2014). Algorithms for nesting with defects. *Discrete Applied Mathematics*, *163*(PART 1), 17–33. https://doi.org/10.1016/j.dam.2012.03.026

Berkey, J. O., & Wang, P. Y. (1987). Two-Dimensional Finite Bin-Packing Algorithms. *Journal of the*

Operational Research Society, *38*(5), 423–429. https://doi.org/10.1057/jors.1987.70

Borgi, T., Zoghlami, N., & Abed, M. (2017). Big data for transport and logistics: A review. *2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, 44–49. https://doi.org/10.1109/ASET.2017.7983742

Bozer, Y.A. and White, J. A. (1996). A generalized design and performance analysis models for end-of-aisle order-picking systems. In *IIE Trans.* (pp. 271–280).

Coffman, E. G., Leung, J. Y.-T., & Ting, D. W. (1978). Bin packing: Maximizing the number of pieces packed. *Acta Informatica*, *9*(3), 263–271. https://doi.org/10.1007/BF00288885

De Koster, R. B. M., Le-Duc, T., & Yugang, Y. (2008). Optimal storage rack design for a 3-dimensional compact AS/RS. *International Journal of Production Research*, *46*(6), 1495–1514. https://doi.org/10.1080/00207540600957795

Egeblad, J. (2009). Placement of two- and three-dimensional irregular shapes for inertia moment and balance. *International Transactions in Operational Research*, *16*(6), 789–807. https://doi.org/10.1111/j.1475-3995.2009.00703.x

Grange, A., Kacem, I., & Martin, S. (2018). Algorithms for the bin packing problem with overlapping items. *Computers and Industrial Engineering*, *115*(September 2017), 331–341. https://doi.org/10.1016/j.cie.2017.10.015

Hiroyuki Okano. (2002). A scanline-based algorithm for the 2D free-form bin packing problem. *Journal of the Operations Research*.

Kenmochi, M., Imamichi, T., Nonobe, K., Yagiura, M., & Nagamochi, H. (2009). Exact algorithms for the two-dimensional strip packing problem with and without rotations. *European Journal of Operational Research*, *198*(1), 73–83. https://doi.org/10.1016/j.ejor.2008.08.020

Lam, G. T. (2017). Considerations on Using Genetic Algorithms for the 2D Bin Packing Problem : a General Model and Detected Difficulties, 303–308.

Litvinchev, I., & Mosquera, D. (n.d.). Packing a limited number of unequal circular objects in a rectangular container, 6–8.

Lodi, A., Martello, S., & Monaci, M. (2002). Two-dimensional packing problems: A survey. *European Journal of Operational Research*, *141*(2), 241–252. https://doi.org/10.1016/S0377-2217(02)00123-6

Ma, N., & Zhou, Z. (2017). Mixed-Integer Programming Model for Two-Dimensional Non-Guillotine Bin Packing Problem with Free Rotation. *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, 456–460. https://doi.org/10.1109/ICISCE.2017.102

Meneghetti, A., Dal Borgo, E., & Monti, L. (2015). Rack shape and energy efficient operations in automated storage and retrieval systems. *International Journal of Production Research*, *53*(23), 7090–7103. https://doi.org/10.1080/00207543.2015.1008107

Murgolo, F. D. (1988). Anomalous behavior in bin packing algorithms, *21*, 229–243.

Roy Jr., P. (1997). Vertical design reduces downtime. *Surface Mount Technology*, *11*(1), 76.

Schwind, G. F. (1995). How to store more in less space. *Material Handling Engineering*, *50*(6), 3.

Wilke, D.-I. M. (2006). Control concept for automous changeable material flow systems. *Logistics Journal*. https://doi.org/10.2195/lj

Yu, Y., & De Koster, M. B. M. (2009). Designing an optimal turnover-based storage rack for a 3D compact automated storage and retrieval system. *International Journal of Production Research*, *47*(6), 1551–1571. https://doi.org/10.1080/00207540701576346