INTERNATIONAL JOURNAL FOR INNOVATIVE RESEARCH IN MULTIDISCIPLINARY FIELD
ISSN(O): 2455-0620 [Impact Factor: 9.47]
Monthly, Peer-Reviewed, Refereed, Indexed Journal with IC Value : 86.87
Volume - 11, Issue - 09, September - 2025

DOIs:10.2015/IJIRMF/202509006 --:-- Research Paper / Article / Review

Applications of Bin Packing with Colocation Constraints in Real Life – A Review

Debajit Sensarma
Assistant Professor, Department of Computer Science, Viveakananda Mission Mahavidyalaya, Chaitanyapur, Purba Medinipur, Haldia, West Bengal
Email - debajit.sensarma2008@gmail.com

Abstract: *The Bin Packing Problem (BPP) is a well-known NP-hard optimization problem that consists of packing items of different sizes into a fixed number of containers (bins) with limited capacity in such a way that the total number of bins used is minimized. Traditionally, the focus of research has been on capacity constraints, but most real-life problems require the consideration of colocation constraints—requirements that certain items must be placed together in the same bin because of functional interdependencies, safety regulations, or operational efficiency. This review paper addresses the problem of incorporating colocations into the classic framework of BPP and its importance to contemporary computational and industrial systems. The article systematically maps the domain of application of bin packing with collocated placement constraints, including cloud computing, container orchestration, distributed storage systems, logistics, database sharding, and energy-efficient operations of data centers. These and other domains increasingly adopt collocated placement strategies because of the need to minimize latency and better resource allocation, which improves system performance. The study also classifies and analyzes heuristic, metaheuristic, exact, and machine learning approaches for solving these enhanced bin packing problems. It emphasizes critical issues like dynamic environments, conflicting constraints, and computational complexity. Using practical cases and real-life scenarios, the article explains the operational advantages of colocation-aware bin packing such as increased cost efficiency, enhanced throughput, regulatory compliance, and energy conservation. The paper ends with mentioning the selection of sustainable optimization, quantum computing, and hybrid algorithmic frameworks with real-time scheduling as active areas of future research focus. This review acts as the starting block for many researchers and practitioners looking to study and design bin packing models with real, constraint-laden, realistic requirements across a multitude of impactful fields.*

Key Words: *Bin Packing Problem, Colocation Constraints, Resource Optimization, Cloud Computing, Heuristic Algorithms, Server Consolidation*

1. INTRODUCTION

The Bin Packing Problem (BPP) is a well-established optimization challenge focused on minimizing the number of containers, or bins, required to accommodate a collection of items, each with a specific size or weight, without exceeding the capacity of any single bin [1]. The primary goal is to achieve maximum space utilization while ensuring all items are packed. Despite its seemingly straightforward nature, the BPP is classified as an NP-hard problem, indicating that finding an optimal solution becomes computationally demanding as the number of items increases.

In many practical scenarios, the simple capacity constraint is insufficient. Additional restrictions often dictate that certain items must be placed together in the same bin. These are known as Colocation Constraints [2]. The necessity for these constraints arises from various factors, including the functional relationship between items, safety regulations that might prohibit the mixing of certain materials, or operational efficiencies gained by keeping related items together [3]. For instance, in logistics, items belonging to the same customer order might need to be shipped together, or in manufacturing, components required for a single assembly might need to be kitted.

Considering colocation constraints significantly complicates the standard BPP, transforming it into a more intricate optimization problem where the interdependencies between items play a crucial role [4]. This survey aims to provide a comprehensive overview of the diverse applications of bin packing with colocation constraints in real-world settings. It will begin by establishing the fundamental definitions of the BPP and colocation constraints. Following this, the report will explore specific applications across various industries, including logistics, manufacturing, cloud computing, and resource allocation. The benefits and challenges associated with implementing such constrained bin packing solutions will be analyzed, and a review of the different algorithms and approaches used to tackle these problems will be provided. Furthermore, the report will highlight case studies of successful implementations and conclude by discussing key trends and future directions in this evolving field.

2. BENEFITS OF INTEGRATING COLOCATION CONSTRAINTS WITH BIN PACKING

Integrating colocation constraints with bin packing offers several significant benefits across various applications. By grouping related items together, operational efficiency can be substantially improved, leading to faster and more streamlined processes in areas like shipping, assembly, picking, and task execution. This enhanced efficiency often translates directly into reduced costs. Optimizing the packing process minimizes the number of bins or resources required, resulting in savings in transportation, storage, material waste, and energy consumption [5].

Furthermore, colocation can enable better utilization of available resources. In cloud computing, for example, placing VMs that share similar operating systems or libraries on the same physical server allows for memory sharing, which reduces overall resource consumption [6]. Similarly, in resource allocation, colocating interdependent tasks on the same processor can minimize communication overhead, leading to more efficient use of processing power. Enhanced performance is another key advantage, particularly in cloud computing and resource allocation, where colocating interdependent components can significantly lower latency and improve application responsiveness.

In industries dealing with physical goods, incorporating colocation constraints can lead to increased safety and regulatory compliance. By ensuring that incompatible or hazardous items are segregated, companies can adhere to safety regulations and minimize the risk of accidents. Moreover, in e-commerce and logistics, grouping all items of a customer's order together for shipping improves order fulfillment accuracy and enhances customer satisfaction by ensuring all items arrive together. Finally, in manufacturing processes like material cutting, strategically colocating patterns on raw materials can significantly reduce the amount of waste, contributing to more sustainable and cost-effective production [7].

3. CHALLENGES AND COMPLEXITIES IN REAL-WORLD IMPLEMENTATION

Despite the numerous benefits, implementing bin packing with colocation constraints in real-world scenarios presents several challenges and complexities. A primary hurdle is the computational complexity inherent in the Bin Packing Problem itself, which is NP-hard. Adding colocation constraints further intensifies the difficulty of finding optimal solutions, especially as the scale of the problem grows.

Real-world applications often involve multiple colocation constraints that might conflict with each other or with the overarching objective of minimizing the number of bins used. For instance, there might be a requirement to keep certain items together while simultaneously needing to segregate others, making it challenging to satisfy all conditions simultaneously. Furthermore, many practical scenarios operate in dynamic environments where items arrive sequentially, as in online bin packing problems. Incorporating colocation constraints in such online settings requires making immediate placement decisions without future knowledge, which can often lead to suboptimal packing arrangements.

Dealing with items that have complex shapes and exist in two or three dimensions, coupled with colocation requirements, introduces another layer of complexity compared to the simpler one-dimensional case. Accurately defining and modeling real-world dependencies and relationships between items into formal colocation constraints that can be effectively used by optimization algorithms can also be a significant challenge. Finally, in some applications, such as cloud computing, the primary goal of maximizing packing density might clash with other important objectives like ensuring workload isolation, maintaining fault tolerance, or guaranteeing specific performance levels for colocated resources [8].

4. ALGORITHMS AND APPROACHES FOR SOLVING BIN PACKING WITH COLOCATION

Various algorithms and approaches have been developed to address the challenges of bin packing problems with colocation constraints. Heuristic algorithms, such as First Fit (FF) and its variant First Fit Decreasing (FFD), are commonly used due to their simplicity and speed. These algorithms place each item into the first available bin with sufficient capacity, with FFD first sorting the items by size in descending order. They can be adapted to handle colocation by checking if placing an item in a bin would violate any defined colocation constraints before proceeding with the placement. Similarly, Best Fit (BF) and Best Fit Decreasing (BFD) algorithms, which place items in the bin with the least remaining space, can also be modified to respect colocation requirements. Next Fit (NF) and Next-k-Fit (NkF) are online heuristics that keep a limited number of bins open, but they are less directly applicable to complex colocation constraints that might require considering multiple bins simultaneously. For two-dimensional packing problems with grouping requirements, shelf algorithms and bottom-left heuristics can be extended to prioritize the placement of colocated items together [9].

Metaheuristic algorithms offer another class of solutions, particularly for complex problems with various constraints. Genetic Algorithms (GA) are evolutionary algorithms that can explore a broad solution space and effectively handle intricate constraints like colocation [10]. Simulated Annealing (SA) is a probabilistic approach that can escape local optima to find good solutions for constrained optimization problems. Particle Swarm Optimization (PSO) is a population-based technique that can be adapted for bin packing with colocation constraints [11], and Variable Neighborhood Search (VNS) systematically explores different solution neighborhoods to improve results [4].

For smaller instances where finding the absolute best solution is critical, exact methods such as Integer Linear Programming (ILP) and Constraint Programming (CP) can be employed. ILP involves formulating the problem as a mathematical program that can be solved using specialized solvers [3].CP is a declarative approach that uses constraints to model the problem and utilizes specialized constraint propagation techniques to find solutions [12].

Finally, learning-based approaches are emerging as a promising direction. Machine Learning (ML) techniques can be used to predict optimal packing strategies or to guide heuristic algorithms, potentially improving performance for problems with colocation constraints [13]. Neural Networks are also being explored to learn patterns from bin packing problems and generate efficient solutions [14].

5. REAL LIFE APPLICATIONS

A. Distributed Storage Systems

Both **Hadoop Distributed File System (HDFS)** and **Google File System (GFS)** are designed to handle large-scale distributed data storage systems, and while they share some similarities, their approaches to storing data blocks efficiently and optimizing access speed can differ. Let's break down how these systems deal with data placement and access optimization:

- **Storing Related Data Blocks Together**

Both **HDFS** and **GFS** aim to improve access speed by storing related data blocks together, which can reduce the need for cross-node communication and increase data locality. This can improve the performance of data retrieval operations, especially when multiple pieces of related data are required together.

- **HDFS**: In HDFS, data is stored in **blocks** (typically 128MB by default), and each block is replicated across multiple nodes for fault tolerance. While HDFS doesn't specifically group related data blocks together, the way blocks are distributed across nodes can allow for efficient retrieval of related data if that data is within the same block or on nearby nodes.
- **GFS**: GFS takes a similar approach by dividing files into **chunks**, each typically 64MB in size. The system is designed to store chunks in a distributed manner across multiple servers. In some cases, related chunks (e.g., those that are part of the same file) are placed on nodes near each other to improve access speed, but this is not always strictly enforced.

Both systems rely on **data locality** to ensure that related blocks or chunks are placed together, though they may not guarantee perfect co-location.

- **Optimizing Storage for Frequently Accessed (Hot) and Less Accessed (Cold) Data**

The management of hot and cold data is a key design feature for improving overall system performance and cost-efficiency.

- **HDFS**:

  Hot data: In HDFS, hot data (frequently accessed data) can be stored on **faster disks** (e.g., SSDs), while cold data (infrequently accessed data) can be stored on **slower, cheaper storage** (e.g., HDDs). This is often done through custom configurations or the use of **HDFS tiered storage**. In this setup, users can define different storage types for data based on access patterns, allowing the system to store frequently accessed data in higher-speed storage while relegating less frequently accessed data to slower, more cost-effective storage.

  Cold data: HDFS does not automatically manage hot and cold data placement but can benefit from application-level management or tools like the **Hadoop Archive** (HAR) for storing older, less frequently accessed data in a more compressed, optimized form.

- **GFS**:

  Hot data: Google File System typically places hot data on **high-performance disks** and **memory caches** in its architecture. GFS doesn't directly separate hot and cold data in the way HDFS does, but it leverages **replication** to ensure that frequently accessed data can be served quickly from multiple replicas across different nodes.

  Cold data: Similarly, for cold data, GFS relies on its **replication strategy** to ensure that all data, whether hot or cold, is still available and fault-tolerant. In GFS, though, it's generally less concerned with explicitly optimizing cold data storage as much as HDFS.

Bin Packing with Colocation

The idea of **bin packing** and **colocation** is generally about minimizing data transfer times by placing related data closer together. However, it's more of an application-level optimization than a core feature of both HDFS and GFS.

- **HDFS**:
  - While HDFS does not explicitly implement **bin packing**, it does aim to **minimize data transfer** by keeping related blocks on the same or nearby nodes. This minimizes the need for data to be transferred over the network when access patterns involve retrieving related blocks.
  - **Colocation** is often achieved through the **HDFS rack awareness feature**, where blocks of a file are placed on different racks to avoid single points of failure but still attempt to maintain proximity to improve data access speed.
- **GFS**:
  - **Colocation** and **bin packing** are handled in GFS through its **chunk placement** policy. Chunks are replicated across different servers to improve fault tolerance, but GFS also attempts to keep related chunks close to each other in the system. This is especially beneficial for **large files** or files with sequential access patterns.
  - GFS doesn't strictly use bin packing algorithms to optimize the placement of chunks in a way that reduces data transfer, but its **distributed architecture** ensures that data placement is done with the intention of minimizing read/write latencies.

Key Differences:

- **Hot and Cold Data Handling**: HDFS has a more flexible mechanism to implement tiered storage and separate cold and hot data, while GFS does not explicitly optimize for this, relying instead on replication and fault tolerance strategies.
- **Co-location of Data**: Both systems aim for data locality but in different ways. HDFS uses **rack awareness** and replication for this, while GFS focuses more on chunk placement strategies.
- **File and Block Management**: HDFS tends to treat blocks as the fundamental unit of storage, while GFS focuses on chunks of data, typically much larger in size, with replication strategies to enhance data availability.

B. Database Storage & Sharding

Bin packing with colocation is a powerful technique used in database storage and sharding to optimize data distribution and improve performance. In this context, **bin packing** involves distributing data (treated as "items") across a limited number of **shards or servers ("bins")**, while **colocation** ensures that related or frequently accessed records are stored together.

Key Applications:

a) **Efficient Data Distribution:**
   - Bin packing helps evenly distribute data across shards, preventing any single server from becoming a bottleneck.
   - It takes into account factors like data size and access frequency to balance load.

b) **Improved Query Performance:**
   - Colocating related data (e.g., a user and their related posts or transactions) minimizes the need for expensive cross-shard joins.
   - This reduces query latency and improves throughput.

c) **Workload-Aware Sharding:**
   - Bin packing strategies can consider read/write patterns, ensuring that frequently accessed data is placed optimally to reduce load on hot shards.

d) **Resharding and Rebalancing:**
   - During data growth or changes in access patterns, bin packing helps determine how to redistribute data with minimal movement and disruption.
   - It maintains the colocation of related data to preserve query efficiency.

e) **Resource Optimization in Multi-Tenant Systems:**
   - In cloud or SaaS platforms, tenants with similar workloads or shared dependencies can be colocated to optimize storage and resource usage.

This approach is especially beneficial in large-scale **distributed databases** like **MongoDB, Cassandra, and HBase**, where performance, scalability, and cost efficiency are critical. Bin packing with colocation leads to smarter data placement, resulting in better utilization of infrastructure and faster data access.

C. Cloud Storage Optimization

Data Deduplication and Chunk Placement

In cloud storage, especially with systems like Amazon S3 or Google Cloud Storage, **deduplication** is used to avoid storing multiple copies of identical data [20]. When similar data chunks are identified, bin packing with colocation can be applied to:

- Place related chunks together to minimize **lookup latency**.
- Optimize **disk head movement** in HDD-based systems.
- Reduce the overhead of redundant storage by colocating frequently accessed chunks.

Example: In deduplicated backup storage, colocating related backups helps speed up restore operations.

Geo-Distributed Storage Replication

Cloud providers replicate data across geographically distributed data centers to ensure **availability** and **compliance** [16]. Here, bin packing with colocation is used to:

- Ensure that **related data units** (e.g., database shards or log files) are replicated together across selected zones to maintain **data consistency**.
- At the same time, enforce **anti-colocation** for fault tolerance (e.g., not placing all replicas in the same power domain or region).

Example: When syncing S3 buckets across multiple AWS regions, related files may be colocated based on user access patterns while ensuring compliance with disaster recovery policies.

D. Multi-Tenant Cloud Storage

In multi-tenant environments [21], cloud providers must serve multiple customers from shared infrastructure. Bin packing with colocation helps by:

- Assigning tenants with **similar access patterns** to the same physical storage (e.g., colocating logs of services that interact).
- Enforcing **anti-colocation** to isolate sensitive data and meet security compliance (e.g., GDPR, HIPAA).

Example: Two business units of the same customer may request their data be stored together to optimize analytics.

Benefits of Using Bin Packing with Colocation in Cloud Storage

Benefit	Description
Improved Performance	Faster access to colocated, related data blocks.
Reduced Latency	Minimizes network hops when related data is together.
Lower Costs	Fewer active storage nodes and optimized energy usage.
Higher Reliability	Smart anti-colocation of redundant data enhances fault tolerance.
Compliance & Security	Helps enforce regulatory constraints and tenant separation.

Technical Challenges

- **Scalability**: Managing colocations across petabytes of data requires distributed algorithms.
- **Dynamic Workloads**: Cloud environments are constantly changing, requiring frequent re-evaluation.
- **Balancing Trade-offs**: Between colocating for performance vs. distributing for fault tolerance.
- **Complex Constraints**: Including bandwidth, latency, security, and SLAs.

E. Data Center Energy Management

The application of bin packing with colocation in Data Center Energy Management focuses on optimizing the placement and operation of IT equipment (servers, storage, network devices) and workloads within the data center to minimize energy consumption while maintaining performance and reliability [23].

Bin Packing for Server Consolidation:

- **Concept:** Physical servers in a data center can be viewed as "bins" with a certain capacity (CPU, RAM, storage, power). Virtual Machines (VMs) or containers, representing workloads, are the "items" to be packed into these bins.

- **Goal:** The primary goal is to consolidate workloads onto a minimum number of physical servers to reduce the number of active but underutilized machines. Idle or low-utilized servers consume significant power without contributing proportionally to the computing output.
- **Algorithms:** Various bin packing algorithms (First-Fit, Best-Fit, Worst-Fit, etc.) are used to determine the most efficient placement of VMs onto physical servers based on their resource requirements.
- **Energy Savings:** By maximizing server utilization, fewer physical machines need to be powered on and cooled, leading to substantial energy savings in both IT equipment power and cooling infrastructure.

Detailed Applications and Benefits:

Optimized Server Consolidation (Bin Packing):

Reducing Idle Power: Underutilized servers consume a significant baseline power even when doing minimal work. Bin packing algorithms aim to maximize the utilization of active servers, allowing for a greater number of idle servers to be powered down or repurposed, leading to direct energy savings.

Right-Sizing Infrastructure: By understanding the resource demands of workloads and efficiently packing them, data centers can potentially avoid over-provisioning hardware, thus reducing upfront capital expenditure and the ongoing energy costs associated with unnecessary infrastructure.

Dynamic Resource Management: Orchestration platforms can use bin packing principles to dynamically adjust workload placement based on real-time resource utilization. When utilization on a server drops below a certain threshold, workloads can be migrated to more densely packed servers, freeing up the underutilized server for power-down.

Strategic Workload Colocation for Energy Savings:

Complementary Resource Utilization:

CPU-intensive vs. Memory-intensive: Placing a VM that heavily utilizes the CPU alongside one that is primarily memory-bound on the same physical server can lead to higher overall server utilization without hitting the limits of either resource. This reduces the need for additional servers.

Peak Load Scheduling: If different applications have peak usage times at different points in the day or week, colocating them on the same hardware can smooth out the overall resource demand, preventing the need to provision for the sum of their peak demands.

Network Traffic Optimization:

High-Communication Workloads: Colocating VMs or containers that communicate frequently on the same physical server or within the same rack/row can reduce network hops and latency. This can lead to lower energy consumption in network devices (switches, routers) and potentially faster application performance, indirectly improving energy efficiency per unit of work.

Data Locality: Placing compute workloads close to the data they access frequently minimizes data transfer over the network, reducing network energy consumption and improving application responsiveness.

Thermal Management:

Heat Density Considerations: While not always intuitive as "colocation" of workloads, understanding the heat dissipation profiles of the underlying hardware is crucial. Placing high-power-density servers in areas with more efficient cooling or ensuring a balanced heat load across the data center can optimize cooling system performance and reduce energy spent on cooling.

Cold Aisle/Hot Aisle Arrangement: This physical colocation of equipment is fundamental for efficient cooling. Servers are arranged to intake cool air from the cold aisles and exhaust hot air into the hot aisles, preventing mixing and improving cooling effectiveness.

Power Distribution Efficiency:

Load Balancing on Power Circuits: Distributing workloads across different power circuits and power distribution units (PDUs) can help avoid overloading specific circuits and improve the overall efficiency of the power distribution system. While not direct workload colocation, it's a form of resource "packing" at the power infrastructure level.

Challenges and Considerations:

- **Complexity of Real-time Monitoring and Decision Making:** Accurately monitoring resource utilization, network traffic, and thermal conditions in real-time and making optimal placement decisions requires sophisticated management tools and potentially AI/ML algorithms.
- **Workload Dynamics:** Workload demands can fluctuate significantly, making static placement strategies ineffective. Dynamic workload migration based on real-time conditions is crucial but adds complexity.
- **Performance Isolation:** While colocation can improve resource utilization, it's essential to ensure that colocated workloads do not negatively impact each other's performance due to resource contention (the "noisy neighbor" problem). Resource quotas and quality-of-service (QoS) mechanisms are necessary.
- **Data Center Physical Constraints:** Physical layout, power availability, and cooling capacity of different zones within the data center can impose constraints on where equipment and workloads can be placed.
- **Security and Compliance:** In some cases, security or compliance requirements might dictate that certain workloads must be physically or logically isolated, limiting colocation opportunities.

Tools and Technologies:

- **Data Center Infrastructure Management (DCIM) Software:** Provides visibility into resource utilization, power consumption, and environmental conditions, enabling informed placement decisions.
- **Workload Orchestration Platforms (e.g., Kubernetes, VMware vSphere):** Offer features for resource scheduling and management, often incorporating bin packing algorithms and allowing for affinity/anti-affinity rules to influence colocation.
- **Energy Management Software:** Specialized tools can analyze energy consumption patterns and recommend optimization strategies, including workload placement adjustments.
- **AI/ML-powered Optimization Engines:** Can learn from historical data and predict future resource needs to proactively optimize workload placement for energy efficiency.

In essence, applying bin packing with colocation in data center energy management is a multi-faceted approach that involves intelligently placing and managing IT resources and workloads to minimize energy consumption across the entire data center infrastructure. It requires a holistic view of resource utilization, network traffic, thermal characteristics, and power distribution, along with sophisticated tools and dynamic management strategies.

6. FUTURE TRENDS AND RESEARCH DIRECTIONS

The field of bin packing with colocation constraints continues to evolve, with several key trends and research directions shaping its future. The integration of advanced technologies like the Internet of Things (IoT) promises to provide real-time data on item characteristics and bin availability, enabling more dynamic and responsive packing strategies. Furthermore, the increasing power of Artificial Intelligence (AI) and Machine Learning (ML) is being leveraged to develop more adaptive and efficient algorithms that can learn optimal packing patterns and handle complex colocation scenarios [26].

There is a growing interest in multi-objective optimization, where bin packing problems with colocation constraints are addressed while simultaneously considering multiple goals, such as minimizing the number of bins used, balancing the

load within bins, and accounting for other practical factors like item stability or fragility. The need for more efficient online algorithms is also becoming increasingly important for real-time applications where items arrive dynamically and packing decisions must be made on the fly, all while respecting colocation requirements.

Hybrid approaches that combine the strengths of different algorithmic techniques, such as integrating heuristics with metaheuristics or combining exact methods with problem decomposition strategies, are also gaining traction for tackling increasingly complex constrained bin packing problems. The potential of quantum computing to address the inherent computational complexity of these problems is also being explored, which could lead to significant breakthroughs in solving large-scale instances with intricate colocation constraints [27].

Sustainability and environmental considerations are also driving future trends, with an increasing focus on using bin packing with colocation to optimize resource utilization and minimize waste in logistics and manufacturing, contributing to more environmentally responsible practices. Finally, there is an emerging interest in developing bin packing algorithms that can consider individual customer preferences or specific product requirements when applying colocation constraints, leading to more personalized and customized packing solutions.

7. CONCLUSION

The integration of colocation constraints into the Bin Packing Problem represents a significant advancement in addressing real-world optimization challenges across a multitude of industries. By considering the necessary relationships and dependencies between items, this approach allows for the development of packing solutions that are not only efficient in terms of space and resource utilization but also aligned with operational requirements, safety regulations, and customer needs. The diverse applications explored in this report, spanning logistics, manufacturing, cloud computing, and resource allocation, underscore the versatility and importance of this optimization technique.

While the computational complexity and the presence of conflicting constraints pose ongoing challenges, the benefits of improved efficiency, reduced costs, better resource utilization, enhanced performance, and increased safety continue to drive research and innovation in this field. The development of sophisticated heuristic, metaheuristic, exact, and learning-based algorithms, coupled with the exploration of emerging technologies like quantum computing, promises to yield even more robust and scalable solutions for increasingly complex scenarios. As industries continue to seek optimized processes and sustainable practices, bin packing with colocation constraints will undoubtedly remain a powerful and essential tool for achieving these goals.

REFERENCES

1. Zhu, W., Chen, S., Dai, M., & Tao, J. (2024). Solving a 3D bin packing problem with stacking constraints. *Computers & Industrial Engineering*, *188*, 109814.
2. Meshram, S., & Wagh, K. P. A Survey on Recent Advances in Spatio-temporal Co-location Pattern Mining. *JOURNAL OF TECHNICAL EDUCATION*, 218.
3. Anand, S., & Guericke, S. (2020, September). A bin packing problem with mixing constraints for containerizing items for logistics service providers. In *International Conference on Computational Logistics* (pp. 342-355). Cham: Springer International Publishing.
4. Bermond, J. C., Cohen, N., Coudert, D., Letsios, D., Milis, I., Perennes, S., & Zissimopoulos, V. (2016, August). Bin packing with colocations. In *International Workshop on Approximation and Online Algorithms* (pp. 40-51). Cham: Springer International Publishing.
5. Akin, Melissa (2020). Two Dimensional Bin Packing. *Industrial Engineering*, Atlim University Research and Project Articles.
6. Sensarma, D. (2024). A Study on Graph-Based Affinity Aware VM Colocation Problems. *International Journal of Research Publication and Reviews*. ISSN 2582-7421, Vol. 5, no. 1, pp-190-194.
7. Naresh R. (2023). Bin Packing Problem, Industrial Engineering, Operational Research, https://www.researchgate.net/publication/343880146_Bin_Packing_Problem_Industrial_Engineering_Operational_Research
8. Aiyar[1], S., Gupta[1], K., Rajaraman, R., Shen, B., Sun, Z., & Sundaram, R. (2019, April). Colocation, Colocation, Colocation: Optimizing Placement in the Hybrid. In *Algorithmic Aspects of Cloud Computing: 4th International*

Symposium, ALGOCLOUD 2018, Helsinki, Finland, August 20–21, 2018, Revised Selected Papers (Vol. 11409, p. 25). Springer.

9. Wu, W., Fan, C., Huang, J., Liu, Z., & Yan, J. (2023). Machine learning for the multi-dimensional bin packing problem: Literature review and empirical evaluation. *arXiv preprint arXiv:2312.08103*.
10. Grange, A., Kacem, I., & Martin, S. (2018). Algorithms for the bin packing problem with overlapping items. *Computers & Industrial Engineering*, *115*, 331-341.
11. Han, B. T., Diehr, G., & Cook, J. S. (1994). Multiple-type, two-dimensional bin packing problems: Applications and algorithms. *Annals of Operations Research*, *50*, 239-261.
12. Tardivo, F., Michel, L., & Pontelli, E. (2024). CP for Bin Packing with Multi-Core and GPUs. In *30th International Conference on Principles and Practice of Constraint Programming (CP 2024)* (pp. 28-1). Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
13. González-San-Martín, J., Cruz-Reyes, L., Dorronsoro, B., Fraire-Huacuja, H., Balderas-Jaramillo, F., Quiroz-Castellanos, M., & Rangel-Valdez, N. (2024). Deep Study on the Application of Machine Learning in Bin Packing Problems. *Computación y Sistemas*, *28*(3), 1275-1290.
14. González-San-Martín, J., Cruz-Reyes, L., Dorronsoro, B., Fraire-Huacuja, H., Balderas-Jaramillo, F., Quiroz-Castellanos, M., & Rangel-Valdez, N. (2024). Deep Study on the Application of Machine Learning in Bin Packing Problems. *Computación y Sistemas*, *28*(3), 1275-1290.
15. De Niz, D., & Rajkumar, R. (2006). Partitioning bin-packing algorithms for distributed real-time systems. *International Journal of Embedded Systems*, *2*(3-4), 196-208.
16. Fahmy, M. M., Elghandour, I., & Nagi, M. (2016, December). CoS-HDFS: co-locating geo-distributed spatial data in hadoop distributed file system. In *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies* (pp. 123-132).
17. Voievodin, Y., Rozlomii, I., & Yarmilko, A. (2023, November). Approach to Evaluate Scheduling Strategies in Container Orchestration Systems. In *Modeling, Control and Information Technologies: Proceedings of International scientific and practical conference* (No. 6, pp. 292-295).
18. Kanellopoulos, G. (2021). Efficient workload co-location on a Kubernetes cluster.
19. Wickramanayaka, N. N., Keppitiyagama, C. I., & Thilakarathna, K. (2022). Communication-Affinity Aware Colocation and Merging of Containers. *International Journal on Advances in ICT for Emerging Regions (ICTer)*, *15*(3).
20. Meyer, D. T., & Bolosky, W. J. (2012). A study of practical deduplication. *ACM Transactions on Storage (ToS)*, *7*(4), 1-20.
21. Azar, Y., Kamara, S., Menache, I., Raykova, M., & Shepard, B. (2014, November). Co-location-resistant clouds. In *Proceedings of the 6th Edition of the ACM Workshop on Cloud Computing Security* (pp. 9-20).
22. Breslow, A. D., Porter, L., Tiwari, A., Laurenzano, M., Carrington, L., Tullsen, D. M., & Snavely, A. E. (2016). The case for colocation of high performance computing workloads. *Concurrency and Computation: Practice and Experience*, *28*(2), 232-251.
23. Solanki, N., & Purohit, R. (2016). Energy-aware virtual machine allocations using bin packing algorithms in cloud data centers. *Int J Eng and Tech Res*, *5*(12), 305-307.
24. Kumar, N., Ruiz, P. M., Menon, V., Kabiljo, I., Pundir, M., Newell, A., ... & Tang, C. (2024). Optimizing resource allocation in hyperscale datacenters: Scalability, usability, and experiences. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)* (pp. 507-528).
25. Cambazard, H., Mehta, D., O'Sullivan, B., & Simonis, H. (2013). Bin packing with linear usage costs–an application to energy management in data centres. In *Principles and Practice of Constraint Programming: 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings 19* (pp. 47-62). Springer Berlin Heidelberg.
26. González-San-Martín, J., Cruz-Reyes, L., Dorronsoro, B., Fraire-Huacuja, H., Balderas-Jaramillo, F., Quiroz-Castellanos, M., & Rangel-Valdez, N. (2024). Deep Study on the Application of Machine Learning in Bin Packing Problems. *Computación y Sistemas*, *28*(3), 1275-1290.
27. V. Romero, S., Osaba, E., Villar-Rodriguez, E., Oregi, I., & Ban, Y. (2023). Hybrid approach for solving real-world bin packing problem instances using quantum annealers. *Scientific Reports*, *13*(1), 11777.