# Solving Offline 3D Bin Packing Problem with Large-sized Bin via Two-stage Deep Reinforcement Learning

Extended Abstract

Hao Yin
Southwest Jiaotong University
Chengdu, China
haoyin@my.swjtu.edu.cn

Fan Chen
Southwest Jiaotong University
Chengdu, China
fchen@swjtu.edu.cn

Hongjie He
Southwest Jiaotong University
Chengdu, China
hjhe@swjtu.edu.cn

## ABSTRACT

Existing Deep Reinforcement Learning (DRL) algorithms address the 3D Bin Packing Problem (3D-BPP) by decomposing the packing action into three sub-stages. However, this three-stage scheme makes it necessary for information to be passed between sub-networks, which may increase the computational cost of training and inference. This paper proposes a two-stage DRL algorithm, combining index and orientation into a single sub-stage to simplify learning. Additionally, a Bidirectional Cooperative Packing (BCP) method is introduced to compress the action space during position selection while retaining exploration capability. The experimental results show that the two-stage DRL algorithm, which incorporates BCP, achieves 0.3%-1.7% improvement in space utilization compared to the currently best-performing algorithm.

## KEYWORDS

Bin Packing Problem; Combinatorial Optimization Problem; Reincorcement Learning

**ACM Reference Format:**
Hao Yin, Fan Chen, and Hongjie He. 2024. Solving Offline 3D Bin Packing Problem with Large-sized Bin via Two-stage Deep Reinforcement Learning: Extended Abstract. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 3 pages.

## 1 INTRODUCTION

3D Bin Packing Problem (3D-BPP) is a classic combinatorial optimization problem [2]. It primarily focuses on packing a set of known-size rectangular items into single or multiple rectangular bins, subject to specific constraints, aiming to maximize the space utilization of the bins. Due to its NP-hard nature, finding the exact optimal solution for 3D-BPP is challenging [14], especially with a large number of items or large-sized bin. Previous studies have primarily used heuristic algorithms [5, 7, 11] or metaheuristic algorithms [1, 9, 16] to approximate the optimal solution.

Recently, with the rapid progress in the field of Deep Reinforcement Learning (DRL), researchers have explored applying DRL to solve 3D-BPP [4, 8, 10, 12, 13, 15, 18, 20, 21]. To address the challenges posed by large-scale action spaces, most of the end-to-end DRL algorithms decompose the packing action into three selection sub-stages of index, orientation, and position. However, this three-stage scheme makes it necessary for information to be passed between sub-networks, which may increase the computational cost of training and inference, and also introduce the possibility of error propagation. On the other hand, most algorithms tend to compromise exploration capability or increase computational complexity to reduce the action space for position selection. This trade-off may negatively impact the performance of the network.

In this paper, we propose a two-stage scheme and a Bidirectional Cooperative Packing (BCP) method to streamline the decision-making process and encourage exploration of reasonable positions while compressing the action space. Experimental results demonstrate that our algorithm achieves state-of-the-art performance.



*Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand.* © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

## 2 METHOD

We solve a variant of offline discrete 3D-BPP, where the height of the bin is adjustable as packed items. The goal is to find a packing strategy that minimizes the height of the bin after all items are packed inside. The packing strategy must adhere to the following constraints: 1) the items packed in the bin must not overlap with each other; 2) the items must not exceed the boundaries of the bin; 3) the items can only be placed in an orthogonal manner.

We represent the packing state at each step $t$ as $s_t = (s_t^I, s_t^B)$, where $s_t^I$ and $s_t^B$ correspond to the item state and bin state, respectively. For $s_t^I$, we use a sequence of size information for all unpacked items, obtained by rotating them in six different directions, to describe it: $s_t^I = \{b_{i,1}, b_{i,2}, b_{i,3}, b_{i,4}, b_{i,5}, b_{i,6} | i \in \{1, 2, \ldots, n\}\}$, where $b_{i,1} = (l_i, w_i, h_i)$, $b_{i,2} = (l_i, h_i, w_i)$, $b_{i,3} = (w_i, l_i, h_i)$, $b_{i,4} = (w_i, h_i, l_i)$, $b_{i,5} = (h_i, l_i, w_i)$, $b_{i,6} = (h_i, w_i, l_i)$, $l_i, w_i, h_i$ represent the length, width, and height of the $i$-th unpacked item, $n$ represents the number of unpacked items. For $s_t^B$, we use a top-down view of the bin to describe it. We represent the packing action as $a_t = (a_t^{io}, a_t^p)$, where $a_t^{io}$ determines which item to select and the orientation for packing, and $a_t^p$ determines where to place the item.

At each step $t$, the two-stage policy network takes the packing state $s_t$ as input and outputs a probability distribution $\pi(a_t|s_t)$ over the possible packing actions. As shown in Figure 1, the policy network follows an encoder-decoder architecture, comprising two encoders and two decoders. Specifically, the backbone networks for the Item Encoder, Index-Orientation Decoder, and Position Decoder employ the Transformer structure [17], while the Bin Encoder utilizes a Convolutional Neural Network (CNN).

Item Encoder takes item state $s_t^I$ as input and encodes it into an item feature sequence $\boldsymbol{h}_t^I$ of dimension $6n \times d_m$. In the Transformer

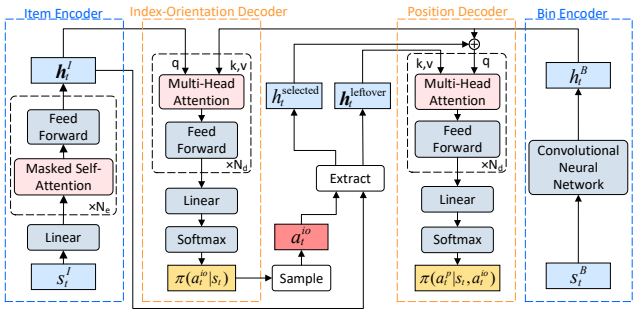


**Figure 1: The architecture of the policy network.**

**Table 1: Comparison results on I1 - I5.**

| Algorithm | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| GA+DBLF [19] | 70.2% | 69.4% | 66.3% | 61.4% | 58.7% |
| EP [3] | 62.7% | 63.8% | 66.3% | 63.3% | 60.1% |
| LAFF [6] | 58.6% | 59.1% | 61.9% | 58.0% | 55.4% |
| EBAFIT [10] | 65.4% | 65.9% | 66.1% | 62.8% | 60.5% |
| MTSL [4] | 62.4% | 60.1% | 55.3% | 50.8% | 46.9% |
| CQL [13] | 67.0% | 69.3% | 73.6% | 58.7% | 57.5% |
| JIANG [10] | 73.5% | 76.9% | 82.0% | 75.2% | 70.5% |
| QUE [15] | 77.5% | 80.4% | 83.4% | 80.5% | 76.7% |
| OUR | **79.2%** | **81.5%** | **84.1%** | **80.8%** | **77.1%** |

structure, we apply a mask to the Multi-Head Self-Attention to avoid calculating the correlation between the embeddings of the same item as they cannot coexist in the bin. Bin Encoder takes bin state $s_t^B$ as input and encodes it into a bin feature vector $h_t^B$ of dimension $d_m$. Index-Orientation Decoder takes $\boldsymbol{h}_t^I$ and $h_t^B$ as input and generates the policy $\pi(a_t^{io}|s_t)$, described as a probability distribution over actions $a_t^{io}$. It should be noted that to make the dimension of the policy $\pi(a_t^{io}|s_t)$ equal to $6n$, here $\boldsymbol{h}_t^I$ is used as the query instead of the key-value. To distinguish the selected item from all unpacked items, we sample the index and orientation from the generated $\pi(a_t^{io}|s_t)$ and find the corresponding embedding in $\boldsymbol{h}_t^I$. We denote the found embedding as $h_t^{\text{selected}}$, and the embeddings of the other items as $\boldsymbol{h}_t^{\text{leftover}}$. Position Decoder generates policy $\pi(a_t^p|s_t, a_t^{io})$ based on the above known conditions. $\boldsymbol{h}_t^{\text{leftover}}$ and $h_t^{\text{selected}} + h_t^B$ are fed into a Transformer decoder as key-value and query, respectively, and the result is passed through several linear layers followed by a softmax function to obtain $\pi(a_t^p|s_t, a_t^{io})$.

The size of action space for $a_t^p$ increases significantly for packing problems with large-sized bins (e.g., 10,000 possibilities for a $100 \times 100$ bin), which increases computational complexity. To address this issue, we propose the Bidirectional Cooperative Packing (BCP) method. The core idea is to use a single policy network and a single value network to generate policy and value for both two directions, by controlling the transposition of the bin view. First, the policy network takes two states, $s_t^1 = (s_t^I, s_t^B)$ and $s_t^2 = (s_t^I, (s_t^B)^{\mathrm{T}})$, as inputs, and generates corresponding policies $\pi(a_t^1|s_t^1)$ and $\pi(a_t^2|s_t^2)$. $a_t^1 = (a_t^{io,1}, a_t^{p,1})$ represents the packing action in y- direction via unidirectional packing[20]. Since $a_t^2 = (a_t^{io,2}, a_t^{p,2})$ is generated based on the transpose of $s_t^B$, it represents a different packing action in x- direction. Then, $a_t^1$ and $a_t^2$ are sampled, and the two items selected by $a_t^{io,1}$ and $a_t^{io,2}$ are packed in respective directions using the unidirectional packing method [20], resulting in two next states $s_{t+1}^1 = (s_{t+1}^{I,1}, s_{t+1}^{B,1})$ and $s_{t+1}^2 = (s_{t+1}^{I,2}, (s_{t+1}^{B,2})^{\mathrm{T}})$. Finally, the value network calculates $V(s_{t+1}^1)$ and $V(s_{t+1}^2)$, and the final action is obtained by comparing the two values: $a_t = \arg\max_{a \in \{a_t^1, a_t^2\}} V(s_{t+1})$.

## 3 EXPERIMENTS

For comparison with previous studies, we adopt a standard procedure to randomly generate packing instances for both training and testing purposes. Specifically, five types of instances are randomly generated, where the total number of items $N$ is 20, 30, 50, 50, and 50, and the bin size is $100 \times 100$, $100 \times 100$, $100 \times 100$, $200 \times 200$, and $400 \times 200$, respectively. We denote these five instances as I1 - I5. The length, width, and height of each item are randomly selected as integers from the range $[L/10, L/2]$, $[W/10, W/2]$, and $[\min(L/10, W/10), \max(L/2, W/2)]$, where $L$ and $W$ represent the length and width of the bin.

We compared the following algorithms on I1 - I5: 1) GA+DBLF [19]; 2) EP [3]; 3) LAFF [6]; 4) EBAFIT [10]; 5) MTSL [4]; 6) CQL [13]; 7) The algorithm proposed by Jiang *et al* [10]; 8) The algorithm proposed by Que *et al* [15]. The experimental results are shown in Table 1. The values in the table refer to the space utilization rate achieved by the algorithms. For each type of instance, we record the utilization on 1024 instances and calculate the mean value. Following the works of [10, 15], for I1 - I3, 128 solutions are sampled from the network and the best one is outputted as the final result, while for I4 and I5, the number of sampled solutions is 16.

From the results shown in Table 1, we can observe that for a wide range of item numbers and bin size, our algorithm outperforms the other algorithms in terms of space utilization, indicating its superiority in generating more effective packing solutions. We also conduct ablation studies for the two components of our algorithm: the two-stage scheme and the BCP method. Due to space constraints, only the experimental results are presented here. The results show that the absence of two-stage scheme and BCP results in a decrease of space utilization calculated by the corresponding algorithm, indicating that both components play a crucial role in the effectiveness of our algorithm.

## 4 CONCLUSION

In this paper, we introduced an end-to-end DRL algorithm to solve offline 3D-BPP with a large-sized bin. Our approach features a novel two-stage scheme and a Bidirectional Cooperative Packing (BCP) method, both of which enhance the learning effectiveness of the policy network. We demonstrate by experiments that our two-stage DRL algorithm, which incorporates BCP, achieves new state-of-the-art results for the packing problem with a large-sized bin and randomly generated items. For future work, we will focus on solving packing problems under other practical constraints.

## REFERENCES


[1] Andreas Bortfeldt and Hermann Gehring. 1998. Applying Tabu Search to Container Loading Problems. In *Operations Research Proceedings 1997*. Springer-Verlag, 533–538.

[2] Andreas Bortfeldt and Gerhard Wäscher. 2013. Constraints in container loading - A state-of-the-art review. *European Journal of Operational Research* 229, 1 (2013), 1–20.

[3] Teodor Gabriel Crainic, Guido Perboli, and Roberto Tadei. 2008. Extreme Point-Based Heuristics for Three-Dimensional Bin Packing. *Informs Journal on Computing* 20, 3 (2008), 368–384.

[4] Lu Duan, Haoyuan Hu, Yu Qian, Yu Gong, Xiaodong Zhang, Jiangwen Wei, and Yinghui Xu. 2019. A Multi-task Selected Learning Approach for Solving 3D Flexible Bin Packing Problem. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 1386–1394.

[5] Emanuel Falkenauer. 1996. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics* 2 (1996), 5–30.

[6] M Zahid Gurbuz, Selim Akyokus, Ibrahim Emiroglu, and Aysun Guran. 2009. An Efficient Algorithm for 3D Rectangular Box Packing, 2009. *Applied Automatic Systems: Proceedings of Selected AAS* (2009), 131–134.

[7] Chi Trung Ha, Trung Thanh Nguyen, Lam Thu Bui, and Ran Wang. 2017. An Online Packing Heuristic for the Three-Dimensional Container Loading Problem in Dynamic Environments and the Physical Internet. In *Applications of Evolutionary Computation*. Springer International Publishing, 140–155.

[8] Haoyuan Hu, Xiaodong Zhang, Xiaowei Yan, Longfei Wang, and Yinghui Xu. 2017. Solving a New 3D Bin Packing Problem with Deep Reinforcement Learning Method. arXiv:1708.05930 [cs.AI]

[9] Jinshan Jiang and Lingzhi Cao. 2012. A hybrid simulated annealing algorithm for three-dimensional multi-bin packing problems. In *2012 international conference on systems and informatics (ICSAI2012)*. IEEE, 1078–1082.

[10] Yuan Jiang, Zhiguang Cao, and Jie Zhang. 2023. Learning to Solve 3-D Bin Packing Problem via Deep Reinforcement Learning and Constraint Programming. *IEEE Transactions on Cybernetics* 53, 5 (2023), 2864–2875. https://doi.org/10.1109/TCYB.2021.3121542

[11] Korhan Karabulut and Mustafa Murat İnceoğlu. 2004. A Hybrid Genetic Algorithm for Packing in 3d with Deepest Bottom Left with Fill Method. In *Proceedings of the Third International Conference on Advances in Information Systems*. Springer-Verlag, 441–450.

[12] Alexandre Laterre, Yunguan Fu, Mohamed Khalil Jabri, Alain-Sam Cohen, David Kas, Karl Hajjar, Torbjorn S. Dahl, Amine Kerkeni, and Karim Beguir. 2018. Ranked Reward: Enabling Self-Play Reinforcement Learning for Combinatorial Optimization. arXiv:1807.01672 [cs.LG]

[13] Dongda Li, Changwei Ren, Zhaoquan Gu, Yuexuan Wang, and Francis Lau. 2020. Solving Packing Problems by Conditional Query Learning. https://openreview.net/forum?id=BkgTwRNtPB

[14] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. 2021. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research* 134 (2021), 105400.

[15] Quanqing Que, Fang Yang, and Defu Zhang. 2023. Solving 3D packing problem using Transformer network and reinforcement learning. *Expert Systems with Applications* 214 (2023), 119153. https://doi.org/10.1016/j.eswa.2022.119153

[16] Liu Sheng, Shang Xiuqin, Cheng Changjian, Zhao Hongxia, Shen Dayong, and Wang Feiyue. 2017. Heuristic algorithm for the container loading problem with multiple constraints. *Computers & Industrial Engineering* 108 (2017), 149–164.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 5998–6008.

[18] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer Networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2692–2700.

[19] Yong Wu, Wenkai Li, Mark Goh, and Robert De Souza. 2010. Three-dimensional bin packing problem with variable bin height. *European Journal of Operational Research* 202, 2 (2010), 347–355.

[20] Jingwei Zhang, Bin Zi, and Xiaoyu Ge. 2021. Attend2Pack: Bin Packing through Deep Reinforcement Learning with Attention. arXiv:2107.04333 [cs.LG]

[21] Hang Zhao, Qijin She, Chenyang Zhu, Yin Yang, and Kai Xu. 2021. Online 3D Bin Packing with Constrained Deep Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 741–749. https://doi.org/10.1609/aaai.v35i1.16155