自动化学报
ACTA AUTOMATICA SINICA

基于深度强化学习的四向协同三维装箱方法

尹昊 陈帆 和红杰

A Four Directional Cooperative Three-dimensional Packing Method Based on Deep Reinforcement Learning

YIN Hao, CHEN Fan, HE Hong-Jie

在线阅读 View online: https://doi.org/10.16383/j.aas.c240124

基于深度强化学习的四向协同三维装箱方法

尹 昊 [1] 陈 帆 [2] 和红杰 [1]

摘 要 物流作为现代经济的重要组成部分, 在国民经济和社会发展中发挥着重要作用. 物流中的三维装箱问题 (Three-dimensional bin packing problem, 3D-BPP) 是提高物流运作效率必须解决的关键难题之一. 深度强化学习 (Deep reinforcement learning, DRL) 具有强大的学习与决策能力, 基于 DRL 的三维装箱方法 (Three-dimensional bin packing method based on DRL, DRL-3DBP) 已成为智能物流领域的研究热点之一. 现有 DRL-3DBP 面对大尺寸容器 3D-BPP 时难以达成动作空间、计算复杂性与探索能力之间的平衡. 为此, 提出一种四向协同装箱 (Four directional cooperative packing, FDCP) 方法: 两阶段策略网络接收旋转后的容器状态, 生成 4 个方向的装箱策略; 根据由 4 个策略采样而得的动作更新对应的 4 个状态, 选取其中价值最大的对应动作为装箱动作. FDCP 在压缩动作空间、减小计算复杂性的同时, 鼓励智能体对 4 个方向合理装箱位置的探索. 实验结果表明, FDCP 在 100 × 100 大尺寸容器以及 20、30、50 箱子数量的装箱问题上实现了 1.2% ~ 2.9% 的空间利用率提升.

关键词 三维装箱问题, 组合优化问题, 深度强化学习, 四向协同装箱

引用格式 尹昊, 陈帆, 和红杰. 基于深度强化学习的四向协同三维装箱方法. 自动化学报, 2024, **50**(12): 2420−2431

DOI 10.16383/j.aas.c240124 **CSTR** 32138.14.j.aas.c240124

A Four Directional Cooperative Three-dimensional Packing Method Based on Deep Reinforcement Learning

YIN Hao[1] CHEN Fan[2] HE Hong-Jie[1]

Abstract As an important part of the modern economy, logistics plays an important role in the national economy and social development. The three-dimensional bin packing problem (3D-BPP) in logistics is one of the key problems that must be solved to improve the efficiency of logistics operations. Deep reinforcement learning (DRL) has a powerful learning and decision-making ability, and the three-dimensional bin packing method based on DRL (DRL-3DBP) has become one of the research hotspots in the field of intelligent logistics. The existing DRL-3DBPs have difficulty in striking a balance between the action space, computational complexity, and exploration capability when solving 3D-BPP with large-size bins. To this end, this paper proposes a four directional cooperative packing (FDCP) method. The two-stage policy network receives the rotated bin states and generates four directional packing policies. Based on the actions sampled from the four policies, the four states are updated accordingly, and the action corresponding to the highest value is selected as the packing action. FDCP encourages agent to explore reasonable packing positions in all four directions while compressing the action space and reducing computational complexity. Experimental results show that FDCP achieves 1.2% ~ 2.9% improvement in space utilization on the packing problem with 100 × 100 large-sized bin and the numbers of 20, 30, and 50 items.

Key words Three-dimensional bin packing problem (3D-BPP), combinatorial optimization problem, deep reinforcement learning (DRL), four directional cooperative packing (FDCP)

Citation Yin Hao, Chen Fan, He Hong-Jie. A four directional cooperative three-dimensional packing method based on deep reinforcement learning. *Acta Automatica Sinica*, 2024, **50**(12): 2420−2431

随着互联网的普及和人们消费能力的提高, 网络购物进入高速发展时期. 网络购物带来的物流需求激增, 为物流行业带来新的挑战, 如何提高物流效率成为研究者关注的重点. 三维装箱问题 (Three-dimensional bin packing problem, 3D-BPP) 作为物流的基本组成部分之一, 是运筹学和计算机科学领域的一类组合优化问题[1], 主要探讨在特定的约束下, 将一系列给定尺寸的长方体箱子装入一个或多个给定尺寸的长方体容器中, 以最大程度地提高容器的空间利用率. 3D-BPP 在货物运输中的集装箱装载[2] 以及仓储物流中的托盘存储[3] 等领域都具

收稿日期 2024-03-12 录用日期 2024-07-04
Manuscript received March 12, 2024; accepted July 4, 2024
本文责任编委 魏庆来
Recommended by Associate Editor WEI Qing-Lai
1. 西南交通大学信息科学与技术学院 成都 611756 2. 西南交通大学计算机与人工智能学院 成都 611756
1. School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756 2. School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu 611756

有广泛的应用. 例如, 电子商务公司在配送货物时, 需要将各类商品合理地装入标准尺寸的集装箱内, 如图 1(a) 所示, 以尽可能减少运输成本; 在图 1(b) 的仓储中心货物配送场景中, 配送系统需要根据装箱方案按顺序分配箱子给机器人抓取, 然后紧凑地码放在托盘上, 以提升存储容量和作业效率. 高效且自动生成装箱方案的装箱策略对于增强物流自动化水平、提高物流效率以及减少运输成本有着重要意义.

(a) 集装箱装载　(b) 仓储配送
(a) Container loading　(b) Storage distribution

图 1　3D-BPP 在物流中的应用场景
Fig. 1　Application scenarios of 3D-BPP in logistics

作为一种多维组合优化问题, 3D-BPP 因其强意义上的 NP-hard 性质难以在数学上采用精确算法在规定时间内求取最优解, 尤其是对于大尺寸容器或多箱子种类数的离散装箱问题. 因此采用精确算法求解 3D-BPP 的相关研究较少, 主要集中在分支定界[4]、动态规划[5]、整数规划[6] 等方法上. 在过去的较长一段时间内, 启发式三维装箱 (Heuristic-3DBP) 方法成为 3D-BPP 领域的研究热点. Heuristic-3DBP 方法通常寻找装箱问题的近似最优解, 在可接受的时间内求取可行解的概率远高于精确算法. 早期有 First-fit[7]、Extreme point[8]、Deepest bottom left[9]、Largest area first-fit[10]、Empty maximal spaces[11] 等方法被提出, 近几年也出现了一些如两阶段层构建[12] 等新的 Heuristic-3DBP 方法. 张德富等[13] 提出通过找点法和水平垂直参考线规则来控制装箱过程. 何琨和黄文奇[14] 结合捆绑策略和 Empty maximal spaces 思想, 提出一种基于动作空间的改进型穴度算法. 以上方法能够在给定时间内有效处理各自场景下的装箱问题. 不过, Heuristic-3DBP 方法通常依赖专家预先设计的手工规则, 不具备普适性, 并且 Heuristic-3DBP 方法是基于经验和直觉的, 可能受到人类主观意识和认知偏差的影响, 在解决复杂装箱问题时可能无法提供全局最优解.

进一步, 研究者提出使用元启发式算法或其他搜索算法来处理装箱问题, 以提高算法求取全局最优解的能力. 在元启发式算法方面, 遗传算法[15−16]、蚁群算法[17]、模拟退火[18−19]、禁忌搜索[20] 等算法均被广泛用于装箱问题的求解. 在其他搜索算法方面, 刘胜等[21] 在水平层构建启发式算法的基础上, 采用树搜索法来搜索最优集装箱装载方案. Ren 等[22]、Zhu 等[23] 和 Araya 等[24] 结合块构建启发式算法和树搜索法来解决实际约束下的集装箱装载问题. 上述算法探索和评估了装箱问题解的搜索空间, 提高了启发式算法求取全局最优解的能力. 不过, 在大尺寸容器和多箱子种类数的复杂装箱问题中, 由于过于庞大的搜索空间以及可能发生的“组合爆炸”现象, 这些算法有着较高的时间复杂度.

为提高算法的普适性和时间效率, 研究者开始尝试使用深度强化学习 (Deep reinforcement learning, DRL). 最早, Vinyals 等[25] 提出一种基于学习的序列模型 PtrNet (Pointer net). Bello 等[26] 以 REINFORCE 强化学习算法训练 PtrNet 以解决旅行商问题. 此后, DRL 被广泛应用于求解平面旅行商[26−27]、最大切割[28]、最小顶点覆盖[29]、最大独立集[30] 和三维装箱等组合优化问题. 在 3D-BPP 方面, 最早的基于 DRL 的三维装箱方法 (Three-dimensional bin packing method based on DRL, DRL-3DBP) 由 Hu 等[31] 提出, 该方法以 PtrNet 充当策略网络来生成装箱顺序, 使用 REINFORCE 算法进行学习, 在三维灵活装箱问题 (Three-dimensional flexible bin packing problem, 3D-FBPP) 上取得了平均容器表面积比 Heuristic-3DBP 低约 5% 的结果. 但是, 该方法依靠启发式算法决定装箱方向和位置, 智能体无法观测全面的信息, 难以学习到最优策略. 在 Hu 等[31] 工作的基础上, Duan 等[32] 提出一种多任务选择学习 (Multi-task selected learning, MTSL) 框架, 以近端策略优化 (Proximal policy optimization, PPO) 算法学习装箱顺序, 以监督的方式学习装箱方向, 相较于文献 [31] 进一步减少了平均容器表面积. 然而, MTSL 仍然由启发式算法指导装箱位置, 无法实现完全端对端的学习, 因此灵活性和全局优化能力仍然较弱. 也有研究者提出使用 DRL 求解一些特殊的 3D-BPP. 例如, Verma 等[33] 将 DRL 引入箱子信息不可完全观测的在线 3D-BPP, 使用深度 Q 网络 (Deep Q-network, DQN) 框架来学习实际机器人约束下的在线装箱策略. Liu 等[34] 针对不规则物体的 3D-BPP, 提出一种基于 DQN 框架的智能装箱算法, 根据不规则物体的三维点云数据优化装箱方案.

近几年来, 研究者开始将注意力转向了端对端的 DRL-3DBP. 设计端对端的 DRL-3DBP 的难点在于 3D-BPP 固有的大规模动作空间, 该挑战源于

装箱顺序、方向以及位置的三种可能性相互乘积导致的巨大数量的可能解. 为处理大规模动作空间问题, 研究者设计了各式各样的方案. Li 等[35] 首次提出一种端对端的 DRL-3DBP: CQL (Conditional query learing), 将每一步的装箱动作分解为索引选取、方向选取和位置选取三部分. 策略网络按顺序分别生成三个子动作的策略, 从而将每个子动作空间限制在较小范围. 但这种三阶段的策略网络结构使得子网络之间必须传递信息并学习特征, 增加了训练和推理的计算复杂性, 加之 CQL 没有处理位置选取子动作的大规模动作空间, 导致其在大尺寸容器装箱问题上的表现欠佳. Jiang 等[36] 在 CQL 的基础上引入了容器顶视图来增强状态表示, 并使用动作表示学习来处理大尺寸容器带来的位置选取子动作的大规模动作空间, 显著地提升了空间利用率. 然而, 动作表示学习涉及额外的监督更新规则, 增大了计算开销. Zhang 等[37] 设计一种单向装箱方法, 通过 REINFORCEE 算法学习策略网络, 只需生成箱子在一轴的位置便可实现其在容器中的定位. 该方法显著减小了位置选取的动作空间, 但它同时也牺牲了对大量合理位置的探索, 限制了智能体行为的多样性, 因此容易陷入局部最优解. Que 等[38] 提出一种基于平面特征容器状态表示的 DRL-3DBP, 通过保留最大平面面积放置点对平面特征下采样, 以此来压缩位置选取子动作的动作空间. 该方法在大尺寸容器 3D-BPP 上达成的空间利用率高于现有其他算法, 但所采用的平面特征下采样同样会使得合适的装箱位置被忽略, 限制了智能体对部分合理位置的探索, 从而影响方法性能.

针对现有端对端 DRL-3DBP 在处理大尺寸容器问题时存在的不足, 本文提出一种基于 DRL 的四向协同装箱 (Four directional cooperative packing, FDCP) 方法. 首先, 策略网络接收剩余箱状态和 4 个旋转方向的容器状态, 生成 4 个装箱策略; 然后, 根据 4 个装箱策略采样得到的动作分别转移至 4 个新状态, 价值网络估计 4 个新状态的价值; 最后, 选取价值最大的新状态对应的动作为装箱动作. 本文的主要贡献总结如下:

1) 提出一种基于 DRL 的 FDCP 方法, 通过旋转容器状态来鼓励智能体对 4 个方向装箱策略的探索, 显著减小了位置选取动作空间, 改善了现有压缩动作空间方法造成的计算开销大、探索能力低的问题.

2) 设计一种用于 FDCP 的两阶段策略网络结构, 即索引方向−位置选取策略. 与三阶段策略网络相比减少了子网络间的信息传递和学习, 从而降低了计算复杂性, 进一步提升了装箱方案的空间利用率.

3) 使用 A2C (Advantage actor-critic) 算法在多种箱子数量和容器尺寸的算例上训练模型, 结果表明本文方法在 20、30、50 箱子数量的算例上的空间利用率比同类最优 DRL-3DBP 提升了 2.9%、2.4% 和 1.2%.

1 问题描述

与文献 [35−38] 相同, 本文研究 3D-BPP 的一种变体. 该类问题仅存在一个容器, 容器高度始终等于已装载箱子的最大高度. 换而言之, 容器的高度是不受限制的. 容器的长和宽以及所有箱子的长、宽、高均为整数. 问题的目标是找到一种装箱策略, 使所有箱子放入容器后所使用的容器高度尽量小, 即空间利用率尽量大. 这类问题通常也被称作离散三维条形包装问题, 后文将其简称为三维条形包装问题 (Three-dimensional strip packing problem, 3D-SPP).

为了形式化定义 3D-SPP, 首先将容器记为 B, 容器的长度和宽度分别表示为 L 和 W. 装箱空间被描述在一个笛卡尔坐标系中, 坐标系的原点位于容器的后−左−底角, 如图 2 所示. 以 $b_i = \{l_i,\ w_i,\ h_i\}_{i\in\{1,\,2,\,\cdots,\,n\}}$ 表示第 i 个剩余 (未装载) 的箱子, l_i, w_i, h_i 分别表示箱子的长、宽和高. 与之对应地, 以 $p_j = \{l'_j,\ w'_j,\ h'_j,\ x_j,\ y_j,\ z_j\}_{j\in\{1,\,2,\,\cdots,\,m\}}$ 表示第 j 个已装载的箱子, 其中, l'_j, w'_j, h'_j 分别表示箱子放入容器后 (即选取完方向后) 的长、宽和高, $(x_j,\ y_j,\ z_j)$ 表示箱子后−左−底角的坐标. 这里的 n 和 m 分别表示剩余和已装载的箱子数, $N = n + m$ 表示箱子总数. 基于以上规定, 3D-SPP 的目标函数定义为

$$\min(\widetilde{H}) \tag{1}$$

其中, $\widetilde{H} = \max_{j\in\{1,\,2,\,\cdots,\,N\}}\left(z_j + h'_j\right)$ 表示所有箱子装载后的容器高度. 与此同时, 箱子的装载必须满

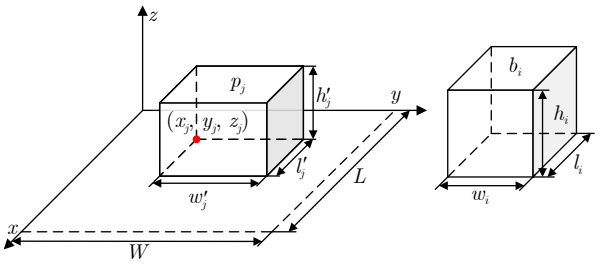

图 2 笛卡尔坐标系与箱子属性

Fig. 2 Cartesian coordinate system and item properties

足以下约束

$$\begin{cases} x_i + l'_i \le x_j + M\left(1 - a_{ij}\right), & i \ne j \\ y_i + w'_i \le y_j + M\left(1 - b_{ij}\right), & i \ne j \\ z_i + h'_i \le z_j + M\left(1 - c_{ij}\right), & i \ne j \\ a_{ij} + a_{ji} + b_{ij} + b_{ji} + c_{ij} + c_{ji} \le 1, & i \ne j \\ x_i + l'_i \le L \\ y_i + w'_i \le W \\ x_i,\ y_i,\ z_i \ge 0 \\ l'_i,\ w'_i,\ h'_i > 0 \\ a_{ij},\ b_{ij},\ c_{ij} \in \{0,\ 1\} \end{cases} \tag{2}$$

其中, a_{ij}, b_{ij}, c_{ij} 分别表示已装载箱子 p_i 是否在 p_j 的后方、左方和下方, 若是则为 1, 否则为 0; M 为一个足够大的正数. 式中前 4 行用于保证箱子两两之间不发生重叠, 第 5、6、7 行用于约束箱子的位置以避免超出容器边界.

从容器为空的初始状态, 直到所有箱子被装载入容器, 这一整个装箱过程可分解为 N 步, 每一步装载一个箱子. 由于第 t 步的装箱决策只与第 t 步容器的状态和剩余箱子的状态有关, 而与第 t 步之前的状态无关, 装箱过程可被视为一个马尔科夫决策过程 (Markov decision process, MDP). 该 MDP 可由一个五元组 $(\mathcal{S},\ \mathcal{A},\ T,\ R,\ \gamma)$ 来描述, 其中, $\mathcal{S}$ 为状态集, 包括所有可能的容器状态和剩余箱状态; $\mathcal{A}$ 为动作集, 包括所有可能的装箱顺序、箱子的方向和放置位置, 由于本文研究的是有限箱子数量的离散装箱问题, 整个状态空间和动作空间都是有限的; $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ 为确定性状态转移函数, 根据动作在容器的对应位置新增箱子, 并从剩余箱集合中移除相应的箱子, 以此转移至新状态; γ 为折扣因子, 用于保证价值函数收敛, 并平衡智能体的近视与远视; $R : \mathcal{S} \times \mathcal{A} \to \mathbf{R}$ 为奖励函数, 本文采用与文献 [35–36, 38] 相同的奖励定义

$$\begin{cases} r_t = g_t - g_{t+1} \\ g_t = LW\widetilde{H}_t - \sum\limits_{j=1}^{t}\left(l'_j w'_j h'_j\right) \end{cases} \tag{3}$$

其中, r_t 表示即时奖励, $\widetilde{H}_t$ 表示第 t 步的容器高度.

装箱过程中, 每一步的装箱决策 (即装箱顺序、方向、位置的选取) 由智能体根据策略 $\pi : \mathcal{S} \to \mathcal{A}$ 做出. 求解 3D-SPP 便意味着找到一个最优策略 π^*, 以最大化期望折扣回报

$$J(\pi) = \mathrm{E}_{\tau\sim\pi} R(\tau) \tag{4}$$

其中, τ 为根据策略 π 采样而得的完整装箱轨迹, $R(\tau) = \sum_{t=0}^{N-1} \gamma^t r_t$ 表示轨迹 τ 的累积折扣回报.

2　状态及动作表示

在介绍方法之前, 本节先对装箱状态及动作进行定义.

设任意第 t 步的状态为 s_t, 它包括容器状态 s_t^B 和剩余箱状态 s_t^I. 本文将 s_t^B 表示为一个容器顶视图, 该视图可描述为一个 $L \times W$ 的网格, 网格中的每个值等于容器中对应位置箱子的累积高度, 如图 3 所示. s_t^B 清晰地呈现了容器的顶部信息, 足以为常用的自顶向下装箱策略[35–38] 提供必要的容器特征. 另一方面, 将剩余箱状态 s_t^I 表示为一个由所有方向的所有剩余箱子的尺寸信息组成的 $6n \times 3$ 维的序列, 即 $s_t^I = \{b_{i,\,1},\ b_{i,\,2},\ b_{i,\,3},\ b_{i,\,4},\ b_{i,\,5},\ b_{i,\,6} | i \in \{1,\ 2,\ \cdots,\ n\}\}$. 其中, $b_{i,\,1} = (l_i,\ w_i,\ h_i)$, $b_{i,\,2} = (l_i,\ h_i,\ w_i)$, $b_{i,\,3} = (w_i,\ l_i,\ h_i)$, $b_{i,\,4} = (w_i,\ h_i,\ l_i)$, $b_{i,\,5} = (h_i,\ l_i,\ w_i)$, $b_{i,\,6} = (h_i,\ w_i,\ l_i)$. 这样的剩余箱状态表示方法确保了旋转后相同尺寸的箱子为网络提供相同的信息, 省略了学习箱子旋转特征过程, 并且丰富了箱子种类, 有助于提高网络的泛化能力.

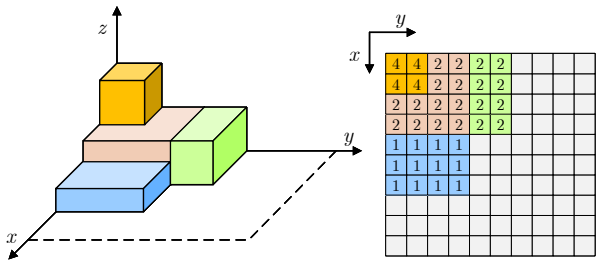

图 3　容器状态表示

Fig. 3　Representation of the bin state

设任意第 t 步的动作为 a_t, 它包括索引选取动作 a_t^i、方向选取动作 a_t^o 和位置选取动作 a_t^p. a_t^i, a_t^o, a_t^p 分别决定在第 t 步哪一个箱子将被装载、旋转至何种方向、放置在什么位置. 为了降低计算复杂性, 本文将 a_t^i 和 a_t^o 合并为一个动作, 称为索引方向选取动作 a_t^{io}, 它能够一次决定装箱的索引和箱子的方向. 剩余箱子数量为 n, 而每个箱子都可能有 6 种旋转方向, 因此, a_t^{io} 的动作空间大小 $|\mathcal{A}^{io}| = 6n$. 对于 a_t^p, 其动作空间大小理应等于可能的放置位置总数. 在容器尺寸较小的情况下 (例如 $L = 10,\ W = 10$), 可采用自顶向下的装箱策略 (确定箱子的 x 和 y 坐标后, 将箱子尽量低放), 无需规划箱子在 z 轴的位置, 因此 a_t^p 的动作空间大小 $|\mathcal{A}^p| = L \times W$. 然而, 在容器尺寸较大的情况下 (例如 $L = 100,\ W = 100$), $L \times W$ 变得非常庞大, 这极大地增

加了计算开销和学习难度. 为减小动作空间, 本文将 a_t^p 选取的可能数量压缩至 $\max(L, W)$ (具体见第 3 节), 即 $|\mathcal{A}^p| = \max(L, W)$. 在动作选取的顺序安排方面, 本文采取常用的索引方向–位置选取顺序.

根据上述状态和动作的定义, 策略 π 可分解为索引方向选取策略和位置选取策略两部分

$$\pi(a_t|s_t) = \pi(a_t^{io},\ a_t^p|s_t^B,\ s_t^I) = \pi(a_t^{io}|s_t^B,\ s_t^I)\pi(a_t^p|s_t^B,\ s_t^I,\ a_t^{io}) \quad (5)$$

3 方法

本文基于 encoder-decoder 结构来构建两阶段策略网络, 以学习索引方向选取策略 $\pi(a_t^{io}|s_t)$ 和位置选取策略 $\pi(a_t^p|s_t,\ a_t^{io})$. 此外, 构建价值网络来估计价值函数, 为策略网络的学习提供依据. 策略网络和价值网络协同工作并运用于 FDCP 方法. 本节首先介绍策略网络和价值网络, 然后介绍 FDCP, 最后阐述模型的训练方式.

3.1 策略网络与价值网络

对任意时间步 t, 策略网络接收剩余箱状态 s_t^I 和容器状态 s_t^B, 计算概率分布 $\pi(a_t^{io}|s_t)$ 和 $\pi(a_t^p|s_t,\ a_t^{io})$. 通过对 $\pi(a_t^{io}|s_t)$ 和 $\pi(a_t^p|s_t,\ a_t^{io})$ 采样, 可确定装载哪一个箱子、箱子的方向以及它的放置位置, 从而做出装箱决策. 与此同时, 价值网络接收剩余箱状态 s_t^I 和容器状态 s_t^B, 计算价值 $V(s_t)$, 以此估计遵循当前策略 π 的期望折扣回报.

本文使用移除了位置编码的 Transformer 和卷积神经网络 (Convolutional neural networks, CNN) 来构建策略网络. 策略网络包含两个编码器, 分别编码剩余箱状态 s_t^I 和容器状态 s_t^B. 解码器数量不同于现有的三阶段策略网络[35−36, 38], 仅包含两个解码器, 分别生成索引方向选取策略和位置选取策略. 编码器和解码器的结构如图 4 所示. 对于剩余箱编码器, 首先将输入的剩余箱状态 s_t^I 线性映射为 $6n \times d_m$ 维的特征嵌入序列, 再将该序列输入到 Transformer encoder 中, 最后输出 $6n \times d_m$ 维的剩余箱特征序列 h_t^I. 由于不同方向的同一个箱子不能同时存在于容器中, 在 Transformer encoder 的 Self attention 中添加一层掩码, 以避免计算同一箱子的 6 个方向的特征之间的相关性. 对于容器编码器, 首先使用若干卷积层提取容器状态特征, 经展平后再由线性层得到 d_m 维的容器特征向量 h_t^B. 对于索引方向解码器, 将 h_t^I 和 h_t^B 分别作为 query 和 key-value 输入到 Transformer decoder 中, 它的输出再经过若干线性层和一个 Softmax 函数, 即得到索引方向选取策略 $\pi(a_t^{io}|s_t)$. 需要注意的是, 为了使网络适应变化的剩余箱数量 (随着箱子逐渐被放入容器, 剩余箱数会逐渐减少), 这里将 h_t^I 作为 query, 而不是 key-value. 在生成位置选取策略之前, 为区分已经通过 $\pi(a_t^{io}|s_t)$ 选择的剩余箱特征和未被选择的剩余箱特征, 同时避免对剩余箱状态的重复编码, 本文根据由 $\pi(a_t^{io}|s_t)$ 采样而得的动作 a_t^{io} 找到 h_t^I 中对应的特征, 将它从中取出并记为 h_t^{sel}, 剩余的非该箱子的特征记为 h_t^{lef}. 位置解码器将 $h_t^B + h_t^{\text{sel}}$ 和 h_t^{lef} 分别作为 query 和 key-value 输入到 Transformer decoder 中, 它的输出再经过若干线性层和一个 Softmax 函数, 即得到位置选取策略 $\pi(a_t^p|s_t,\ a_t^{io})$.

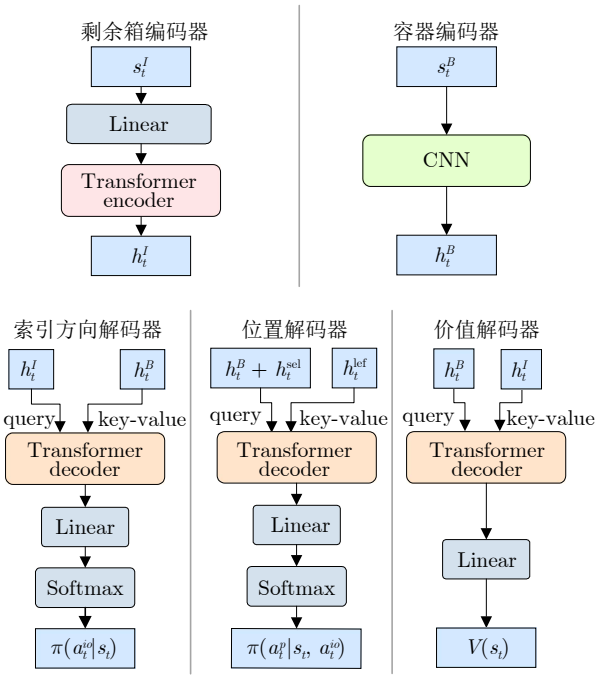

图 4 各编码器和解码器的结构

Fig. 4 Structure of each encoder and decoder

价值网络的结构与策略网络相似, 它包含容器编码器、剩余箱编码器和价值解码器. 其中, 容器编码器和剩余箱编码器的结构与策略网络相同, 价值解码器的结构如图 4 所示. 首先, 将 h_t^B 和 h_t^I 分别作为 query 和 key-value 送入 Transfomer decoder 中. 然后, 将 Transfomer decoder 的输出送入若干线性层便得到价值 $V(s_t)$.

3.2 四向协同装箱

由策略网络的结构可知, 位置选取策略 $\pi(a_t^p|s_t,\ a_t^{io})$ 的维度受位置解码器中最后的线性层控制, 其大小可人为调节. 在容器尺寸较小的情况下, 一般将 $\pi(a_t^p|s_t,\ a_t^{io})$ 的维度设为 $L \times W$, 这样可以确保

根据任意选取的 a_t^p 都能够由自顶向下的装箱策略实现箱子在容器中的唯一定位. 然而, 这种方法并不适用于大尺寸容器, 因为过大的 L 和 W 会导致 a_t^p 的动作空间 $|\mathcal{A}^p|$ 变得非常庞大, 极大地增加计算开销和学习难度. 目前, 处理大尺寸容器装箱问题的主流方法是限制箱子的放置区域以减小动作空间[37−38], 但这同时限制了智能体对部分合理装箱位置的探索. 文献 [36] 使用动作表示学习来提高动作空间的泛化性, 允许智能体从历史动作中推断动作的结果, 但动作表示学习涉及额外的监督更新规则, 增加了模型复杂性和计算开销. 关键问题在于, 如何在减小动作空间并不增加计算开销的同时, 尽量保留智能体对合理装箱位置的探索.

针对这一关键问题, 本文提出一种四向协同装箱方法. 在介绍 FDCP 前, 首先介绍单向装箱[37] (One directional packing, ODP) 方法. 由于文献 [37] 所述场景的装箱方向与本文不同, 为便于描述, 将其方向修改为适应本文场景, 方法的本质不变. ODP 方法在本文所述装箱场景中具体描述如下: 给定一个容器、一个准备放置的箱子和已确定的该箱子将放置的 x (或 y) 坐标 (记为 p), 箱子的放置遵从以下规则 (优先级降序):

1) 箱子放置的 x (或 y) 坐标等于 p;

2) 箱子放置的 z 坐标尽量小;

3) 箱子放置的 y (或 x) 坐标尽量小.

由该规则可知, ODP 仅根据箱子在一轴的位置便实现了箱子在容器中的定位, $|\mathcal{A}^p|$ 被压缩至 L (或 W), 因为网络只需生成箱子的 x (或 y) 坐标的选取策略. 同时, ODP 鼓励箱子向 y 负 (或 x 负) 方向贴合放置, 避免了对部分松散位置的探索. 但正如前文所述, 这种方法限制了智能体的探索, 使网络容易陷入局部最优解.

实际人类装箱时, 通常会将箱子朝向容器的 4 个壁放置, 并尽量保证箱子之间的紧密贴合, 这种装箱策略相较于 ODP 有着更多的位置选取可能. 受实际人类装箱策略的启发, 本文提出一种 FDCP 方法. 首先, 假定策略网络学习 ODP 方法的 y 负方向的装箱策略 (即通过策略网络确定箱子的 x 坐标, 再使 z 和 y 坐标尽量小), 则对于任意状态 $s_t = (s_t^B, s_t^I)$, 由策略网络能够计算并采样得到装载箱子的索引、方向以及它的 x 坐标, 这些信息可指导 ODP 方法将箱子向 y 负方向装载. 若将 s_t 中的容器状态 s_t^B 以垂直于其平面的法线为轴顺时针旋转 90°, 得 $s_t^{B'}$, 则由同样的策略网络将生成针对 $s_t^{B'}$ 的 y 负方向的装箱策略. 而实际上, 该策略等同于针对原始 (未旋转) 容器 s_t^B 的 x 正方向的装箱策略. 同理, 由顺时针旋转 180° 和 270° 后的容器状态 (分别记为 $s_t^{B''}$ 和 $s_t^{B'''}$) 生成的策略, 分别等同于针对 s_t^B 的 y 正和 x 负方向的装箱策略. 4 种方向的装箱策略如图 5 所示. 根据上述分析, 可通过控制 s_t^B 的旋转, 实现由一个策略网络生成 4 个方向的装箱策略. 对于价值网络, 也能够用相同的方式实现由一个价值网络计算遵循 4 种方向装箱策略的状态的价值.

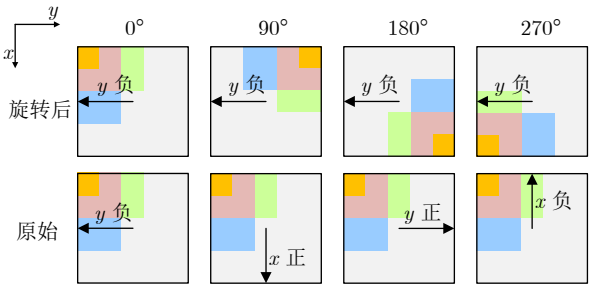

图 5　4 种方向的装箱策略

Fig. 5　The packing policy for the four directions

FDCP 方法的结构如图 6 所示. 首先, 策略网络接收 $s_t^1 = (s_t^B, s_t^I)$, $s_t^2 = (s_t^{B'}, s_t^I)$, $s_t^3 = (s_t^{B''}, s_t^I)$ 和 $s_t^4 = (s_t^{B'''}, s_t^I)$ 4 个状态的输入. 网络根据 4 个方向的输入状态分别生成候选策略 $\pi(a_t^1|s_t^1)$, $\pi(a_t^2|s_t^2)$, $\pi(a_t^3|s_t^3)$ 和 $\pi(a_t^4|s_t^4)$. 其中 a_t^1, a_t^2, a_t^3, a_t^4 分别表示针对旋转 0°、90°、180°、270° 后容器的 y 负方向装箱动作. 随后, 采样 4 个候选动作并与环境交互, 分别转移得到 4 个候选新状态 s_{t+1}^1, s_{t+1}^2, s_{t+1}^3 和 s_{t+1}^4. 最后, 价值网络计算 $V(s_{t+1}^1)$, $V(s_{t+1}^2)$, $V(s_{t+1}^3)$, $V(s_{t+1}^4)$, 其中最大价值对应的候选动作被选为最终动作 a_t 并执行

$$\begin{cases} k_{\max} = \arg \max\limits_{k\in\{1,\,2,\,3,\,4\}} (V(s_{t+1}^k)) \\ a_t = a_t^{k_{\max}} \end{cases} \tag{6}$$

不同于现有端对端的 DRL-3DBP[35−38], FDCP 由策略网络和价值网络协同工作来生成装箱动作, 装箱位置的确定不仅取决于策略网络生成的位置选取策略, 还取决于各个方向动作转移至的新状态的价值大小. FDCP 使得策略网络和价值网络对 4 种不同的装箱方向学习到相同的规律, 以此保证训练的稳定, 并将位置选取动作空间 $|\mathcal{A}^p|$ 控制在较小范围. FDCP 求解 3D-SPP 的完整流程如图 7 所示, 图中描述了 FDCP 根据输入的箱子和容器信息得到完整装箱方案 P 的具体过程, 装箱方案 P 中记录了每一个箱子的装箱顺序、方向和位置.

需要注意的是, FDCP 方法可用的前提是容器

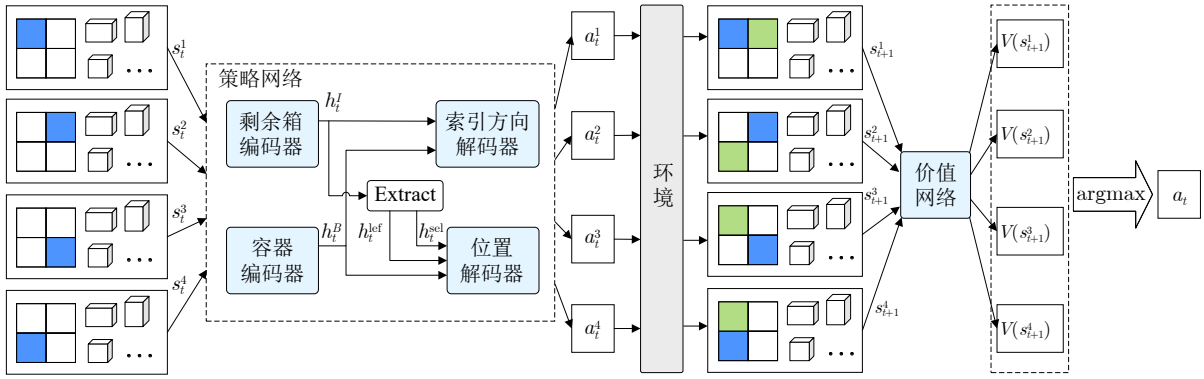

图 6 四向协同装箱方法结构

Fig. 6 Structure of the four directional cooperative packing method

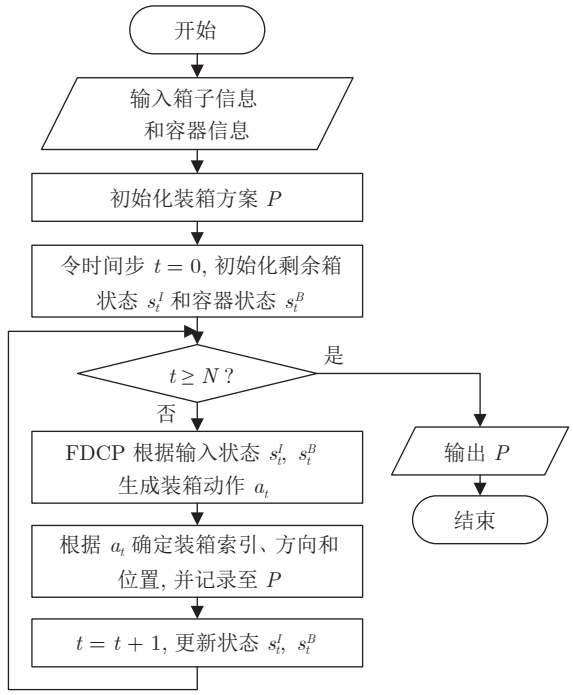

图 7 FDCP 求解 3D-SPP 流程图

Fig. 7 Flowchart of FDCP for solving 3D-SPP

的 L 和 W 相等, 因为必须确保容器的旋转不会改变其状态的维度, 从而使策略网络和价值网络能够在不同装箱方向上保持一致操作. 对于 L 和 W 不同的容器, 可采用最近邻插值算法[39] 将其尺寸改变为 $\max(L,\, W) \times \max(L,\, W)$, 同时相应地调整剩余箱的尺寸以匹配容器改变的比例. 以 $L > W$ 的容器为例, 假定策略网络学习 y 负方向的装箱策略, 当它生成针对旋转 0° 或 180° 容器的策略时, 输入的剩余箱子状态中的每个箱子的宽 (这里以 w 来表示) 需调整为 $\lceil L/W \times w \rceil$; 当它生成针对旋转 90° 或 270° 容器的策略时, 则剩余箱子状态中的每个箱子的长 (这里以 l 来表示) 需调整为 $\lceil L/W \times l \rceil$.

3.3 训练

本文使用结合了广义优势估计 (Generalized advantage estimation, GAE) 的 A2C 强化学习算法来训练网络. actor 和 critic 分别充当策略网络和价值网络. 训练过程的损失函数 $\mathcal{L}$ 定义如下

$$\begin{cases} \mathcal{L} = \mathcal{L}_{\text{actor}} + \mathcal{L}_{\text{critic}} + \beta \mathcal{L}_{\text{entropy}} \\ \mathcal{L}_{\text{actor}} = -\hat{A}_t \times \sum \left(\ln \pi_{\theta_{io}} + \ln \pi_{\theta_p} \right) \\ \mathcal{L}_{\text{critic}} = \text{MSE}\left(V_{\theta_c}(s_t),\, V_{\theta_c}(s_t) + \hat{A}_t \right) \\ \mathcal{L}_{\text{entropy}} = -\sum \left(\pi_{\theta_{io}} \ln \pi_{\theta_{io}} + \pi_{\theta_p} \ln \pi_{\theta_p} \right) \end{cases} \tag{7}$$

其中, β 为平衡智能体探索与利用的超参数; $\hat{A}_t$ 为由 GAE 计算而来的优势; $\pi_{\theta_{io}}$ 和 π_{θ_p} 分别指 $\pi_{\theta_{io}}(a_t^{io}|s_t)$ 和 $\pi_{\theta_p}(a_t^p|s_t,\, a_t^{io})$; θ_{io} 和 θ_p 为策略网络参数; θ_c 为价值网络参数; MSE$(\cdot)$ 表示均方误差函数.

网络训练过程的伪代码如算法 1 所示, 其中 $\theta_a = (\theta_{io},\, \theta_p)$. 算法中的第一层循环内为一个完整的回合 (*episode*), 每一回合包含 N 步, 每一步都会生成 4 个方向的装箱策略和对应的价值, 通过比较价值的大小来确定最终的装箱方向, 并执行该方向对应的动作实现状态转移. 与此同时, 每一次状态转移都记录转移元组 $(s_t,\, a_t,\, r_t,\, s_{t+1})$, 每 n_{gae} 步根据记录的元组计算优势 $\hat{A}_t$ (n_{gae} 表示 GAE 更新步数), 并根据式 (7) 计算损失以更新 actor 和 critic 的网络参数.

算法 1. 训练过程

输入. 包含容器和箱子信息的训练集.

输出. 策略网络和价值网络的参数.

1) 初始化 actor 参数 θ_a 和 critic 参数 θ_c;
2) **for** each *episode* $\in [1,\, 2,\, 3,\, \cdots]$ **do**
3) 从训练集采样批次大小个算例;
4) 初始化 $t=0$ 并根据算例初始化容器状态 s_t^B 和剩余箱状态 s_t^I;

5) **for** $i = 0$ to N/n_{age} **do**
6) **for** $j = 0$ to n_{age} **do**
7) 由网络生成策略 $\pi_{\theta_a}(a_t^1|s_t^B,\ s_t^I)$, $\pi_{\theta_a}(a_t^2|s_t^{B'},\ s_t^I)$, $\pi_{\theta_a}(a_t^3|s_t^{B''},\ s_t^I)$ 和 $\pi_{\theta_a}(a_t^4|s_t^{B'''},\ s_t^I)$;
8) 从 4 个策略采样候选动作 a_t^1, a_t^2, a_t^3, a_t^4;
9) 分别执行 4 个候选动作, 转移至候选新状态 s_{t+1}^1, s_{t+1}^2, s_{t+1}^3, s_{t+1}^4;
10) 计算价值 $V_{\theta_c}(s_{t+1}^1)$, $V_{\theta_c}(s_{t+1}^2)$, $V_{\theta_c}(s_{t+1}^3)$, $V_{\theta_c}(s_{t+1}^4)$;
11) 执行最大价值对应的动作, 转移至新状态;
12) 根据式 (3) 计算奖励 r_t 并记录转移元组;
13) $t \leftarrow t+1$;
14) **end for**
15) 根据式 (7) 计算损失并更新 θ_a 和 θ_c;
16) **end for**
17) **end for**

4　实验

本节将 FDCP 与多种 Heuristic-3DBP 和 DRL-3DBP 进行对比以验证方法性能, 并进行消融实验以验证 FDCP 各组成部分的有效性.

4.1　实验设置

在 3D-SPP 的算例生成方面, 本文遵从与文献 [36, 38] 相同的参数设置来生成算例. 具体来说, 随机生成 20、30、50 三种不同箱子数量的算例来进行模型的训练与测试. 其中, 每个箱子的长、宽、高分别在 $[L/10,\ L/2]$, $[W/10,\ W/2]$, $[\min(L/10,\ W/10),\ \max(L/2,\ W/2)]$ 范围内随机取整数. 在容器方面, 考虑 100×100, 200×200, 400×200 三种不同尺寸的容器. 3D-SPP 的容器高度是随装载箱子的高度变化的, 因此无需事先设定.

训练与测试过程中, 策略网络和价值网络中的 Transformer 的层数均设置为 3, 每个 attention 的头数均设置为 8. 对于 CNN, 由于容器尺寸的不同会导致容器状态视图的维度变化, 卷积层的设置也需相应调整. 对于 100×100 的容器, 设置两个卷积层, 卷积核个数分别为 32 和 64; 对于 200×200 和 400×200 的容器, 设置三个卷积层, 卷积核个数分别为 32、64、64. 每个卷积层后均跟随层归一化和 ReLU 激活函数. 另外, 设置策略网络和价值网络中的特征向量维度 $d_m = 128$. 模型的训练使用 Adam 优化器, 策略网络和价值网络的学习率分别设置为 5×10^{-6} 和 10^{-5}. 训练过程中, 每个周期生成 512×32 个算例, 批次大小为 32. 设置 GAE 的更新步数 $n_{\text{gae}} = 5$, 折扣因子 $\gamma = 0.99$, 衰变因子 $\lambda = 0.95$. 本文方法基于 PyTorch 实现, 所有实验均在 Intel(R) Core(TM) i5-12400F CPU 和 NVIDIA GeForce RTX 2080 TI GPU 上进行.

4.2　对比实验

在 3 种箱子数量和 3 种容器尺寸的算例上与以下三维装箱方法进行对比: 1) GA+DBLF[15] (Genetic algorithm with deepest bottom left heuristic); 2) EP[8] (Extreme point); 3) LAFF[10] (Largest area first-fit); 4) 结合层与墙构建的启发式方法[36] (EBAFIT); 5) MTSL[32]; 6) CQL[35]; 7) Jiang 等[36] 提出的方法 (记为 JIANG); 8) Que 等[38] 提出的方法 (记为 QUE). 以上方法中, 1) ~ 4) 为 Heuristic-3DBP, 5) ~ 8) 为 DRL-3DBP.

与文献 [36, 38] 中的测试相同, 考虑到实际货物配送场景中托盘的标准尺寸约为 100 cm × 100 cm (如图 1(b) 所示), 本文重点测试方法在 100×100 容器上的表现. 该尺寸容器算例上的 3D-SPP 的对比实验结果如表 1 所示, 其中包括两项性能的对比: 空间利用率 UR 和计算时间 Time. 参考

表 1　100×100 容器装箱算例上的对比结果

Table 1　Comparative results on packing instances with 100×100 bin

方法		$N=20$		$N=30$		$N=50$	
		UR (%)	Time (s)	UR (%)	Time (s)	UR (%)	Time (s)
Heuristic-3DBP	GA+DBLF	70.2	17.5	69.4	36.3	66.3	71.9
	EP	62.7	<1.0	63.8	<1.0	66.3	<1.0
	LAFF	58.6	<1.0	59.1	<1.0	61.9	<1.0
	EBAFIT	65.4	<1.0	65.9	<1.0	66.1	1.5
DRL-3DBP	MTSL	62.4	4.8	60.1	10.2	55.3	23.0
	CQL	67.0	1.0	69.3	1.2	73.6	3.3
	JIANG	73.5	2.3	76.9	3.2	82.0	10.9
	QUE	76.5	1.4	79.3	2.1	82.4	3.5
	FDCP	**79.4**	1.9	**81.7**	3.1	**83.6**	5.2

文献 [36, 38], 对于任意箱子数量, 生成 1 024 个随机算例的装箱方案, UR 等于所有装箱方案的空间利用率的平均值. 在每个算例的装箱方案生成过程中, 采样生成 k_{samp} 个候选装箱方案, 选取其中空间利用率最大的方案作为最终结果. 计算时间 Time 指 1 024 个算例生成装箱方案的平均时间. 其他方法的计算结果来源于文献 [36, 38], 需要注意的是, MTSL[32]、CQL[35] 以及 JIANG[36] 等 DRL-3DBP 的采样数量 $k_{\text{samp}} = 128$, 而 QUE[38] 和 FDCP 的采样数量 $k_{\text{samp}} = 16$. 从表 1 可以观察到, DRL-3DBP (除了 MTSL) 在空间利用率方面的表现显著优于 Heuristic-3DBP. 在采样数量 k_{samp} 均取 16 的前提下, 与目前性能最优的 QUE[38] 相比, FDCP 在箱子数量 N 等于 20、30、50 的算例上分别取得了 2.9%、2.4% 和 1.2% 的 UR 提升. 与 MTSL[32]、CQL[35] 以及 JIANG[36] 等 DRL-3DBP 相比, 即便 FDCP 的采样数量 k_{samp} 远小于这些方法, 但取得的空间利用率提升显著. 此外, 可以注意到随着箱子数量的提升, FDCP 相较于 JIANG[36] 和 QUE[38] 的 UR 提升逐渐减小, 这是因为 DRL-3DBP 普遍在更多箱子数量的算例上达到更高的空间利用率, 而装箱策略由于 3D-SPP 固有的空间利用率上界 (100%) 更难以在高空间利用率的基础上进行提升. 对于计算时间 Time, FDCP 相较于 Heuristic-3DBP 需要更多的计算时间, 在 20、30、50 数量箱子的算例上的计算时间分别比目前最优的 QUE[38] 多 0.5 s、1.0 s、1.7 s. 这是因为 FDCP 每一步需要生成 4 个方向的装箱策略并评估 4 个新状态的价值. 然而, 在实际应用中, 工业机器人搬运对应数量的箱子往往需要上百秒, 数秒计算时间的差异在实际应用中影响较小. 相比之下, 方法在空间利用率上的提升更为重要. 综上所述, FDCP 能够在 100×100 容器算例的 3D-SPP 上以较少的计算时间为代价提升装箱方案 1.2% ~ 2.9% 的空间利用率.

为了验证方法在更大尺寸容器方面的有效性, 本文在现有方法考虑的最大容器尺寸 (200×200 和 400×200) 的算例上进一步测试方法性能. 各方法在 $N = 50$、容器尺寸为 200×200 和 400×200 的算例上的空间利用率 UR 如表 2 所示. 从表 2 可以观察到, 相较于 100×100 的容器, 所有方法的 UR 均有降低. 这是因为更大的容器带来了更复杂的容器状态、剩余箱状态, 以及更大的位置选取动作空间, 使得方法更难以寻找最优解. 在 200×200 和 400×200 容器算例上, FDCP 在 QUE[38] 的基础上提升了 0.7% 和 0.1% 的 UR. 这一提升相较于 100×100 容器略有降低, 这是因为 QUE[38] 使用的容器平面特征下采样方法压缩了容器状态, 而本文为保留容器的局部特征并未对容器状态进行压缩, 在计算资源的限制下, 学习更大尺寸容器的特征较为困难.

图 8 展示了 5 组算例的装箱方案可视化效果, 从图 8 可以观察到, 对于任意箱子数量和容器尺寸, FDCP 使得大多数箱子紧密地贴合在一起, 从而减少了空间浪费.

为进一步评估 FDCP 在不同箱子数量算例上的性能, 本文在 100×100 容器上, 使用箱子数量 N 等于 20、30 和 50 的算例训练的模型 (分别记为 FDCP20、FDCP30 和 FDCP50), 分别在不同箱子数量的算例上进行测试, 结果如图 9 所示. 从图中可以观察到, 随着 N 的增加, 空间利用率 UR 总体呈上升趋势, 这是由于箱子数量的增加使得箱子的尺寸和形状组合更加多样化, DRL 模型能够学习到更多样化的装箱策略. FDCP20、FDCP30 和 FDCP50 分别在少数量箱子 ($N \leq 25$)、中等数量箱子 ($25 < N \leq 35$) 以及多数量箱子 ($N > 35$) 的算例上表现更为突出, 这是因为模型更关注与其训练箱子数量相匹配的算例的优化. 例如, FDCP50 模型更关注对多数量箱子的全局规划, 该规划下的下层箱子的放置可能并不平坦, 导致少数量箱子算例装箱方案的空间浪费. 从图中还可以观察到, FDCP 对不同的箱子数量具备一定的泛化能力, 当箱子数量的变化不大 (小于等于 5) 时, 各模型的 UR 浮动不会超过 5% (不考虑 $N = 10$ 算例自身导致的低空间利用率情况). 在计算时间方面, Time 随着 N 的增加而增加, 这是因为更多的箱子需要更复杂的优化来找到最优装箱策略, 增加了计算复杂性.

4.3 消融实验

本节在 100×100 容器, 箱子数量为 20、30、50 的算例上对以下 FDCP 方法的三个组成部分进行消融实验: 1) 以一个策略网络 (价值网络) 学习 4 个方向装箱策略 (价值函数) 的协同训练方式, 记

表 2 200×200 和 400×200 容器算例上各方法的空间利用率 UR (%)

Table 2 Space utilization (UR) of each method on instances with 200×200 and 400×200 bins (%)

容器尺寸	GA+DBLF	EP	LAFF	EBAFIT	MTSL	CQL	JIANG	QUE	FDCP
200×200	61.4	63.3	58.0	62.8	50.8	58.7	75.2	80.5	**81.2**
400×200	58.7	60.1	55.4	60.5	46.9	47.5	70.5	76.7	**76.8**

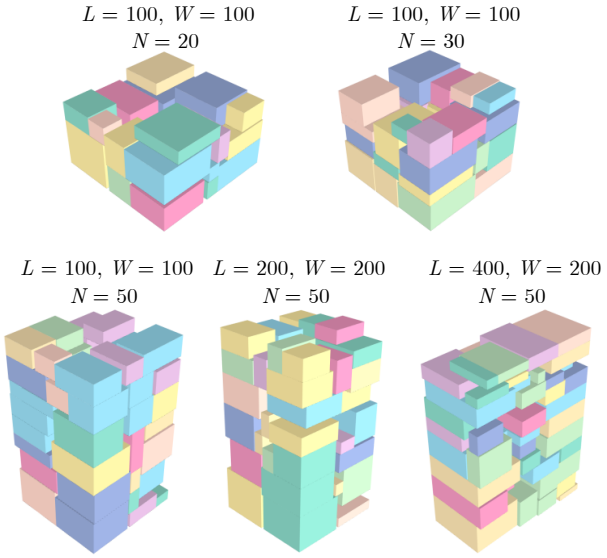

图 8　装箱结果可视化

Fig. 8　Visualization of packing results

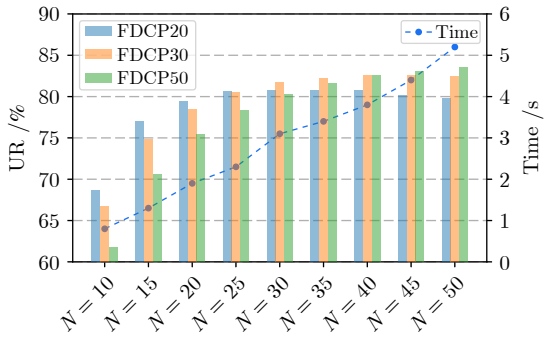

图 9　FDCP 在不同箱子数量算例上的测试结果

Fig. 9　Test results of FDCP on instances with different numbers of items

为 CO; 2) 4 个方向的装箱方式, 记为 FD; 3) 用于 FDCP 的两阶段策略网络结构, 记为 PN. 表 3 展示了消融实验结果, 其中“−CO”所指方法使用 4 个策略网络 (价值网络) 学习 4 个方向装箱策略 (价值函数), “−FD”所指方法为单向装箱, “−PN”所指方法采用与 JIANG[36] 类似的策略网络结构 (即三个解码器分别生成索引、方向和位置选取策略). 从表 3 可以观察到, 对于任意箱子数量 N, 结合三个组件的 FDCP 方法表现出的性能明显高于 −CO、−FD 和 −PN, CO、FD 和 PN 的缺失会导致 1.7% ~ 3.5% 的空间利用率下降. 这表明本文提出的 FDCP 方法及其策略网络结构在生成高质量的装箱方案方面都起到了重要作用. 此外, 还可以发现 −CO 和 −FD 生成方案的平均空间利用率几乎相同, 这是因为在不采用协同训练方式的前提下, 4 个策略网络 (价值网络) 中只有最大估计价值的新状态对应的网络能够充分学习, 即退化为单向装箱.

表 3　消融实验结果 (%)

Table 3　Results of ablation experiment (%)

方法	$N = 20$	$N = 30$	$N = 50$
FDCP	**79.4**	**81.7**	**83.6**
−CO	75.9	79.2	81.4
−FD	75.9	79.0	81.5
−PN	76.5	79.5	81.9

为视觉展示 FDCP 方法和单向装箱方法在位置选择上的差异, 本文分别记录两种方法在 1 024 个算例上的放置位置. 具体而言, 将 100×100 的容器划分为 100 个 10×10 的区域, 并使用热力图显示每个区域中放置箱子 (中心) 的次数. 两种装箱方法的热力图如图 10 所示. 由图 10 可以观察到, 单向装箱方法使得大多数箱子放置在 $x \in [20,\ 40] \cup [70,\ 90]$ 的区域, 导致了有限的位置选择. 相反, FDCP 方法在容器的诸多区域都有放置, 表明智能体探索了更多样化的位置选择.

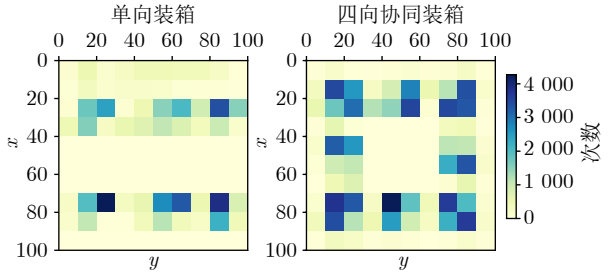

图 10　容器各区域放置次数热力图

Fig. 10　Heat map of the number of placements in each area of the bin

5　结束语

本文提出一种基于深度强化学习的四向协同装箱方法, 并对应设计了一种两阶段策略网络结构, 以求解大尺寸容器和多箱子种类数的离散三维装箱问题. 面对大尺寸容器带来的大规模离散动作空间, FDCP 在压缩位置选取动作空间的同时, 鼓励智能体对 4 个方向合理装箱策略的探索. 实验验证表明, FDCP 在大型容器和随机生成箱子信息的装箱问题中取得了先进的结果. 未来的工作将专注于稳定性等实际约束下的三维装箱问题求解, 高效且能够用于实际的离线或在线装箱算法将是后续研究的重点.

References

1　Bortfeldt A, Wascher G. Constraints in container loading——A state-of-the-art review. *European Journal of Operational Research*, 2013, **229**(1): 1−20

2　Jiao G S, Huang M, Song Y, Liu H B, Wang X W. Container

loading problem based on robotic loader system: An optimization approach. *Expert Systems With Applications*, 2024, **236**: Article No. 121222

3 Consolini L, Laurini M, Locatelli M. A dynamic programming approach for cooperative pallet-loading manipulators. *IEEE Transactions on Automation Science and Engineering*, DOI: 10.1109/TASE.2023.3310007

4 Martello S, Pisinger D, Vigo D. The three-dimensional bin packing problem. *Operations Research*, 2000, **48**(2): 256−267

5 Hifi M. Exact algorithms for unconstrained three-dimensional cutting problems: A comparative study. *Computers & Operations Research*, 2004, **31**(5): 657−674

6 Chen C S, Lee S M, Shen Q S. An analytical model for the container loading problem. *European Journal of Operational Research*, 1995, **80**(1): 68−76

7 Dósa G, Sgall J. First Fit bin packing: A tight analysis. In: Proceedings of the 30th International Symposium on Theoretical Aspects of Computer Science (STACS 2013). Dagstuhl, Germany: Schloss Dagstuhl——Leibniz-Zentrum für Informatik, 2013. 538−549

8 Crainic T G, Perboli G, Tadei R. Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing*, 2008, **20**(3): 368−384

9 Karabulut K, Inceoglu M M. 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. Berlin, Heidelberg: Springer, 2004. 441−450

10 Gurbuz M Z, Akyokus S, Emiroglu I, Guran A. An efficient algorithm for 3D rectangular box packing. In: Proceedings of the Selected AAS 2009 Papers. Skopje, Macedonia: Society for ETAI of Republic of Macedonia, 2009. 131−134

11 Parreño F, Alvarez-Valdés R, Tamarit J M, Oliveira J F. A maximal-space algorithm for the container loading problem. *INFORMS Journal on Computing*, 2008, **20**(3): 412−422

12 Hasan J, Kaabi J, Harrath Y. Multi-objective 3D bin-packing problem. In: Proceedings of the 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO). Manama, Bahrain: IEEE, 2019. 1−5

13 Zhang De-Fu, Wei Li-Jun, Chen Qing-Shan, Chen Huo-Wang. A combinational heuristic algorithm for the three-dimensional packing problem. *Journal of Software*, 2007, **18**(9): 2083−2089
(张德富, 魏丽军, 陈青山, 陈火旺. 三维装箱问题的组合启发式算法. 软件学报, 2007, **18**(9): 2083−2089)

14 He Kun, Huang Wen-Qi. An action space based deterministic efficient algorithm for solving the three-dimensional container loading. *Chinese Journal of Computers*, 2014, **37**(8): 1786−1793
(何琨, 黄文奇. 基于动作空间的三维装箱问题的确定性高效率求解算法. 计算机学报, 2014, **37**(8): 1786−1793)

15 Wu Y, Li W K, Goh M, de Souza R. Three-dimensional bin packing problem with variable bin height. *European Journal of Operational Research*, 2010, **202**(2): 347−355

16 Yang J L, Liu H W, Liang K B, Zhou L, Zhao J H. Variable neighborhood genetic algorithm for multi-order multi-bin open packing optimization. *Applied Soft Computing*, 2024, **163**: Article No. 111890

17 Silveira M E, Vieira S M, Sousa J M D C. An ACO algorithm for the 3D bin packing problem in the steel industry. In: Proceedings of the 26th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems. Berlin, Heidelberg: Springer, 2013. 535−544

18 Zhang De-Fu, Peng Yu, Zhu Wen-Xing, Chen Huo-Wang. A hybrid simulated annealing algorithm for the three-dimensional packing problem. *Chinese Journal of Computers*, 2009, **32**(11): 2147−2156
(张德富, 彭煜, 朱文兴, 陈火旺. 求解三维装箱问题的混合模拟退火算法. 计算机学报, 2009, **32**(11): 2147−2156)

19 Tsao Y C, Tai J Y, Vu T L, Chen T H. Multiple bin-size bin packing problem considering incompatible product categories. *Expert Systems With Applications*, 2024, **247**: Article No. 123340

20 Bortfeldt A, Gehring H. Applying tabu search to container loading problems. In: Proceedings of the Operations Research Proceedings 1997. Berlin, Heidelberg: Springer, 1998. 533−538

21 Liu Sheng, Shen Da-Yong, Shang Xiu-Qin, Zhao Hong-Xia, Dong Xi-Song, Wang Fei-Yue. A multi-level tree search algorithm for three dimensional container loading problem. *Acta Automatica Sinica*, 2020, **46**(6): 1178−1187
(刘胜, 沈大勇, 商秀芹, 赵红霞, 董西松, 王飞跃. 求解三维装箱问题的多层树搜索算法. 自动化学报, 2020, **46**(6): 1178−1187)

22 Ren J D, Tian Y J, Sawaragi T. A tree search method for the container loading problem with shipment priority. *European Journal of Operational Research*, 2011, **214**(3): 526−535

23 Zhu W B, Lim A. A new iterative-doubling Greedy-Lookahead algorithm for the single container loading problem. *European Journal of Operational Research*, 2012, **222**(3): 408−417

24 Araya I, Moyano M, Sanchez C. A beam search algorithm for the biobjective container loading problem. *European Journal of Operational Research*, 2020, **286**(2): 417−431

25 Vinyals O, Fortunato M, Jaitly N. Pointer networks. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal, Canada: MIT Press, 2015. 2692−2700

26 Bello I, Pham H, Le Q V, Norouzi M, Bengio S. Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv: 1611.09940, 2017.

27 Jin Y, Ding Y D, Pan X H, He K, Zhao L, Qin T, et al. Pointerformer: Deep reinforced multi-pointer Transformer for the traveling salesman problem. In: Proceedings of the 37th AAAI Conference on Artificial Intelligence. Washington, USA: AAAI Press, 2023. 8132−8140

28 Tang Y H, Agrawal S, Faenza Y. Reinforcement learning for integer programming: Learning to cut. In: Proceedings of the 37th International Conference on Machine Learning. Vienna, Austria: PMLR, 2020. 9367−9376

29 Theodoropoulos T, Makris A, Psomakelis E, Carlini E, Mordacchini M, Dazzi P, et al. GNOSIS: Proactive image placement using graph neural networks & deep reinforcement learning. In: Proceedings of the IEEE 16th International Conference on Cloud Computing (CLOUD). Chicago, USA: IEEE, 2023. 120−128

30 Ahn S, Seo Y, Shin J. Deep auto-deferring policy for combinatorial optimization [Online], avaliable: https://openreview.net/forum?id=Hkexw1BtDr, March 12, 2024

31 Hu H Y, Zhang X D, Yan X W, Wang L F, Xu Y H. Solving a new 3D bin packing problem with deep reinforcement learning method. arXiv preprint arXiv: 1708.05930, 2017.

32 Duan L, Hu H Y, Qian Y, Gong Y, Zhang X D, Xu Y H, et al. A multi-task selected learning approach for solving 3D flexible bin packing problem. arXiv preprint arXiv: 1804.06896, 2019.

33 Verma R, Singhal A, Khadilkar H, Basumatary A, Nayak S, Singh H V, et al. A generalized reinforcement learning algorithm for online 3D bin-packing. arXiv preprint arXiv: 2007.00463, 2020.

34 Liu H W, Zhou L, Yang J L, Zhao J H. The 3D bin packing problem for multiple boxes and irregular items based on deep Q-network. *Applied Intelligence*, 2023, **53**(20): 23398−23425

35 Li D D, Ren C W, Gu Z Q, Wang Y X, Lau F. Solving packing problems by conditional query learning [Online], avaliable: https://openreview.net/forum?id=BkgTwRNtPB, March 12, 2024

36 Jiang Y, Cao Z G, Zhang J. Learning to solve 3-D bin packing problem via deep reinforcement learning and constraint pro-

gramming. *IEEE Transactions on Cybernetics*, 2023, **53**(5): 2864−2875

37 Zhang J W, Zi B, Ge X Y. Attend2Pack: Bin packing through deep reinforcement learning with attention. arXiv preprint arXiv: 2107.04333, 2021.

38 Que Q Q, Yang F, Zhang D F. Solving 3D packing problem using Transformer network and reinforcement learning. *Expert Systems With Applications*, 2022, **214**: Article No. 119153

39 Parker J A, Kenyon R V, Troxel D E. Comparison of interpolating methods for image resampling. *IEEE Transactions on Medical Imaging*, 1983, **2**(1): 31−39

尹 昊 西南交通大学信息科学与技术学院博士研究生. 2020 年获得西南交通大学学士学位. 主要研究方向为强化学习, 人工智能.
E-mail: haoyin@my.swjtu.edu.cn
(**YIN Hao** Ph.D. candidate at the School of Information Science and Technology, Southwest Jiaotong University. He received his bachelor degree from Southwest Jiaotong University in 2020. His research interest covers reinforcement learning and artificial intelligence.)

陈 帆 西南交通大学计算机与人工智能学院副教授. 主要研究方向为机器学习, 多媒体安全和计算机应用.
E-mail: fchen@swjtu.edu.cn
(**CHEN Fan** Associate professor at the School of Computing and Artificial Intelligence, Southwest Jiaotong University. His research interest covers machine learning, multimedia security, and computer applications.)

和红杰 西南交通大学信息科学与技术学院教授. 主要研究方向为深度学习, 图像处理和信息安全. 本文通信作者. E-mail: hjhe@swjtu.edu.cn
(**HE Hong-Jie** Professor at the School of Information Science and Technology, Southwest Jiaotong University. Her research interest covers deep learning, image processing, and information security. Corresponding author of this paper.)