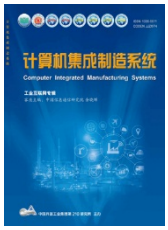


计算机集成制造系统
Computer Integrated Manufacturing Systems
ISSN 1006-5911,CN 11-5946/TP

《计算机集成制造系统》网络首发论文

题目：自动化码头出口箱箱位分配优化超启发式算法
作者：黄子钊，庄子龙，滕浩，秦威，秦涛，邹鹰
收稿日期：2020-08-11
网络首发日期：2021-01-06
引用格式：黄子钊，庄子龙，滕浩，秦威，秦涛，邹鹰．自动化码头出口箱箱位分配优化超启发式算法．计算机集成制造系统.
https://kns.cnki.net/kcms/detail/11.5946.TP.20210105.1515.030.html

网络首发时间：2021-01-06 09:06:31
网络首发地址：https://kns.cnki.net/kcms/detail/11.5946.TP.20210105.1515.030.html

自动化码头出口箱箱位分配优化超启发式算法

黄子钊[1]，庄子龙[1]，滕 浩[1]，秦 威[1+]，秦 涛[2]，邹 鹰[3]
(1.上海交通大学机械与动力工程学院，上海 200240；2.上海海勃物流软件有限公司，上海 200080；3.上海国际港务(集团)股份有限公司，上海 200080)

摘 要：针对实际场景下的自动化码头集装箱堆场出口箱箱位分配问题，本文考虑到三维码放、场桥接力和多箱区协同等特性，并以降低任务的不均衡性和后续的翻箱率为优化目标，构建了自动化码头出口箱箱位分配数学规划模型。为了提高求解质量，本文开发了一种基于强化学习的超启发式方法，该方法将具有不同特征的启发式算法和智能算法作为低层启发式策略，采用新颖的基于策略的强化学习方法作为高层决策方法，并使用深度学习更高效地提取状态中的隐藏模式。最后，根据洋山四期自动化集装箱堆场历史数据设计了算例，并将提出算法与常规智能算法进行了对比，证明了提出算法的有效性和优越性，同时表明提出算法能够提高堆场作业效率，为自动化集装箱堆场提供决策支持。

关键词：自动化码头；箱位分配；强化学习；超启发式算法
中图分类号：U691^{+}.3　　**文献标识码**：A

Optimization of outbound container space assignment in automated container terminals based on hyper-heuristic

HUANG Zizhao[1]，*ZHUANG Zilong*[1]，*TENG Hao*[1]，*QIN Wei*[1+]，*QIN Tao*[2]，*ZOU Ying*[3]
(1.School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China；2.Shanghai Harbor e-Logistics software Co.,Ltd., Shanghai 200080, China；3.Shanghai International Port (Group) Co., Ltd., Shanghai 200080, China)

Abstract：Aiming at the outbound container space assignment problem in automated container terminals, this paper considers the characteristics of three-dimensional stacking, yard crane and multi-area cooperation, and builds a mathematical programming model to reduce task imbalance and container flip rate. In order to improve the quality of the solution, this paper develops a reinforcement learning based hyper-heuristic, which uses several heuristic and intelligent algorithms with different characteristics as low-level heuristics, a novel policy-based reinforcement learning as high-level selection strategy and deep learning to extract hidden patterns more efficiently. Finally,based on actual data of Yangshan Phase IV Automated Container Yard, numerical experiments are conducted.The results show that the proposed algorithm can effectively solve this problem and is superior to state-of-art intelligent algorithms. Therefore, the proposed algorithm can improve work efficiency and provide decision support for automated container yard.

Keywords：automated container terminal; container space assignment problem; reinforcement

收稿日期：2020-08-11；修订日期：2020-11-30。Received 11 Aug. 2020; accepted 30 Nov. 2020.
基金项目：国家重点研发计划(2019YFB1704401)。**Foundation item**：Project supported by the National Key Research and Development Program, China(No. 2019YFB1704401).

learning; hyper-heuristic

0 引言

近年来，在集装箱码头领域，随着自动化设备、人工智能算法等技术的不断发展，自动化集装箱码头在提高码头整体作业效率、降低劳动力成本等方面逐渐形成明显优势，已成为未来的发展趋势。然而，由于规模庞大、业务复杂、现场操作人员少等特点，自动化集装箱码头的作业效率还存在较大的提升空间，急需更高效的计划调度方案。其中，在关系到船舶停靠时间即码头效率的出口箱装船作业中，自动化集装箱堆场由于箱区垂直于岸边、交互区位于箱区两侧、场桥冲突等约束的存在，以及出口箱需要从陆侧入场、从海侧出场导致的场桥操作距离较大的状况，难以取得较合理的决策，是出口箱作业效率提升的瓶颈所在。在堆场作业中，出口箱的箱位分配是堆场优化的基础和关键，方案的优劣直接影响到了堆场的吞吐能力，进而影响出口箱装船作业的效率。但是，由于问题规模较大、动态性较强、约束较多等特点，目前实际采用的箱位分配方案难以满足任务平衡、翻箱率低等要求，制约了作业效率的提升。因此，研究自动化集装箱码头出口箱箱位分配问题对提高出口箱整体作业效率具有重要意义，已受到学者们的持续关注。

在自动化集装箱码头出口箱箱位分配问题中，现有研究按研究尺度分可以分为以下两种：箱区或倍位分配以及具体箱位分配。在箱区或倍位分配中，Liang 等考虑一个作业路上的任务，基于最小化集装箱移动距离，建立了混合整数规划模型并采用混合的遗传算法展开求解[1]。Jiang 和 Jin 以最小化成本即可预计的场桥移动操作成本及翻箱成本为目标，开发了分支定界方法展开求解，得到了每个任务的目标倍位[2]。梁承姬等针对一个箱区，以负载均衡为目标，设计了基于网络流的禁忌搜索算法，通过与精确求解算法展开对比，证明了算法的有效性[3]。Yang 等考虑到经济方面的约束，以最小化成本和最大化利润为目标，基于遗传算法针对每个任务指派对应的倍位和机械资源[4]。靳志宏等考虑到外集卡提前预约入场的现状，为了让箱区负载均衡，建立了数学规划模型并使用 CPLEX 进行求解，结论表明预约信息对作业均衡有 20-30 个百分点的帮助[5]。倪敏敏和芸道方考虑到场桥同时作业的影响，采用分区域动态平衡的方法减少场桥的无效作业时间，提高堆场效率[6]。韩笑乐等考虑泊位与堆场的相互影响，设计了改进的遗传算法得到不确定环境下的箱区和泊位分配方案，实现了鲁棒性调度决策[7]。

在具体箱位的分配中，Carlo 和 Vis 综述了早些年针对箱位分配问题的研究，其中求解方法可以分为两阶段操作即先执行箱区和倍位分配后执行具体箱位分配以及混合操作即直接得到完整分配

方案。同时，Carlo 和 Vis 针对不同的求解方法展开了对比，发现后者的性能要优于前者[8]。近五年，Boysen 和 Emde 针对一个倍位的堆存范围，以最小化翻箱为目标，定义了箱位分配问题的下界，并证明了可以在 n^3 时间复杂度内精确求解[9]。Goerigk 等考虑到箱位分配会对场桥调度有直接影响，以最小化箱位冲突为目标，建立了破坏和重建的局部搜索方法[10]。曾建智等引入了后期机械资源调度的约束，开发了双层遗传算法同时解决箱位分配与资源调度问题[11]。周鹏飞和赵金秋也考虑到外集卡的不确定性，基于目前码头已有的预约信息，使用了基于图的禁忌搜索算法，较确定性模型取得了十余个百分点的提高[12]。Shen 等针对一个倍位的箱位分配问题，考虑到问题时间要求较高，在同一种环境下应用机器学习方法，采用堆场现状作为状态，目标箱位作为动作，方案的可行性作为回报，并采用随机数据进行训练和验证，得到了符合实际的分配方案[13]。李隋凯等针对出口箱箱位分配，引入接力操作，采用动态规划方法快速得到了较好的解[14]。Chang 等考虑箱区负载均衡以及箱位冲突最小的目标，开发了基于模拟退火框架的两阶段调度模型，得到完整的箱位分配方案[15]。Maldonado 等基于集装箱在港口的停留时间，开发了箱位分配模型，从而减少可预见的翻箱操作[16]。

总之，在箱位分配问题中，当前的一部分研究更多关注于具体箱位的选取，并采用精确求解、调度规则或智能算法加以解决。另一部分研究则关注箱区或倍位的选择，通过和其他问题的集成形成全局中更好的方案。然而，由于自动化集装箱码头出口箱箱位分配问题具有较大规模和难度，当前的解决方案还不能满足码头整体高效作业的需要。同时在当前研究中，较少考虑实际自动化集装箱堆场的接力、多箱区协同等特性，且方案效果还有较大的提升空间。因此基于这一现状，本文提出了一个考虑实际场景中上述两个特性的出口箱箱位分配问题模型，从而确定具体箱位，使后续翻箱率最低。之后，本文提出了一种新颖的基于强化学习的超启发式算法用于解决这一问题，得到高质量的箱位分配方案，提高自动化集装箱码头出口箱作业的作业效率。

1 问题描述

图 1 为自动化集装箱堆场堆存的示意图，自动化集装箱堆场可以划分为数十个箱区，每一个箱区由若干个倍位组成，每个倍位由数列和数层构成。在自动化集装箱堆场中，通常一个箱区包含了 50 个倍位，一个倍位由 10 列和 6 层组成，即一个箱区包含了 3000 个集装箱箱位,具有极大的规模。

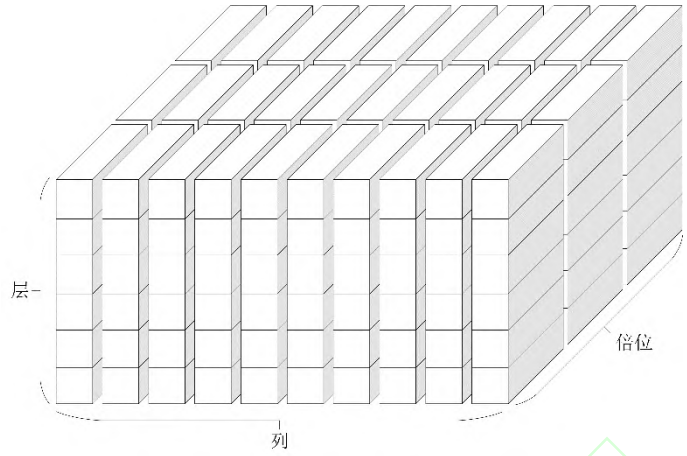


图 1 堆场堆存示意图

图 2 为典型的自动化集装箱堆场的示意图。区别于传统集装箱堆场，为了满足自动化区域与人工区域的分离，自动化集装箱堆场与 AGV 和外集卡的交互集中于箱区两侧，而不像传统堆场那样可以在箱区边缘任意地方。同时，自动化集装箱堆场使用轨道吊沿着既定的轨道执行操作，不能相互跨越和在多个箱区间移动。因此为了最大化工作效率，一个箱区中通常会配置两台场桥，服务于海侧和陆侧两个交互区。基于上述特点，自动化集装箱堆场箱区的吞吐能力的上限较低，单个箱区难以满足短时间内密集的出口箱运输任务。在此情境下，为了最小化任务延期时间和最大化工作效率，自动化集装箱堆场采用多箱区协同的方式，打破传统堆场按类别放置集装箱的原则，将同时段同泊位的集装箱分散于数个箱区中，尽可能使得各个箱区的任务负荷相当，从而发挥箱区数量大的优势，满足出口箱运输需求。另外，基于出口箱海侧任务密集的特点，为了让吞吐能力满足需求，堆场引入了悬臂吊设备，能够与 AGV 在箱区侧面展开交互，减小场桥移动距离，提升了箱区海侧交互能力。这些方式虽然具备更高的效率，但极大的提高了问题的复杂性，对堆场的计划调度提出了更大的挑战。

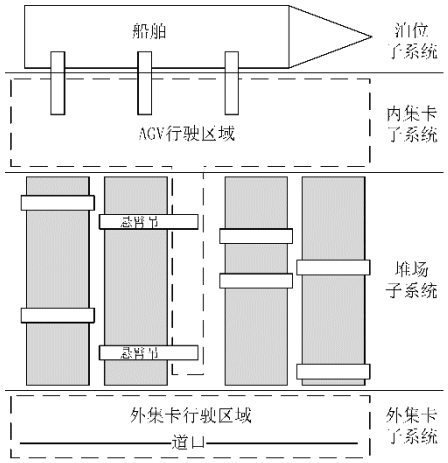


图 2 自动化集装箱堆场示意图

在出口箱作业中，集装箱由外集卡运输进入堆场，经过陆侧入场作业、堆场存储、海侧出场作业三个阶段，最终由海侧经 AGV 运输离开堆场，整体的作业流程如图 3 所示。在这一流程中，由于自动化集装箱堆场的长度较长、场桥无法相互跨越，若由单一场桥执行入场和出场作业会导致作业区间较大，极大影响到另一台场桥的作业，导致整体效率偏低。因此自动化集装箱堆场设计了较为复杂的接力操作即两台场桥配合执行一项出口箱的入场任务。首先在堆场中部设立了中转区用于出口箱接力操作，其次针对出口箱制定中转箱位，由陆侧场桥将出口箱从外集卡转移至中转区，之后由海侧场桥由中转区移动至海侧的目标箱位。由于集装箱堆叠的一系列原则，中转区的箱位选择需要尽可能均衡，从而减少阻塞情况。

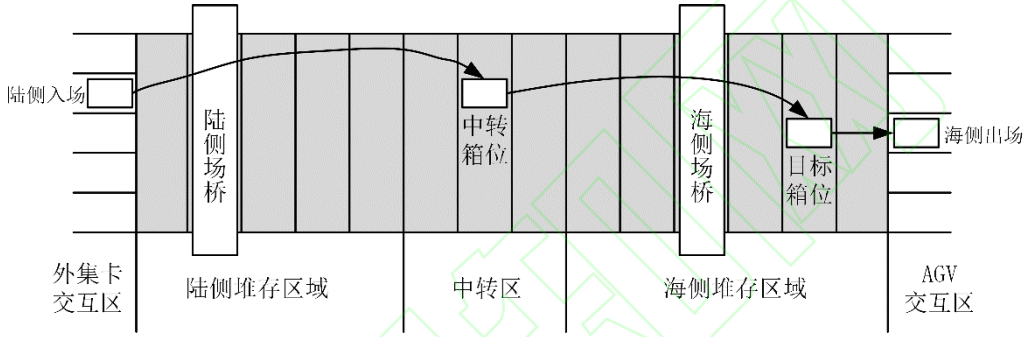


图 3 自动化集装箱码头出口箱作业流程

由于自动化集装箱堆场是一个三维环境，在出口箱出场作业中，若目标集装箱不在最顶层则产生压箱，需要先将上方的集装箱翻倒再执行出场作业。图 4 列出了一个倍位的出场作业截面图。假设当前时刻为 0，图中集装箱的数字为根据船舶配积载和 AGV 计划得到的出场时间，数字越小说明出场越早。在出口箱入场作业时，若该集装箱的出场时间晚于其下方集装箱的出场时间，则产生压箱，后期需要执行一次翻箱操作。以图 4 所示出口箱为例，左侧的出口箱下方没有早于该箱的集装箱，因此无需翻箱。而右侧出口箱下方集装箱的出场时间要早于该箱，需要翻箱，影响了作业效率。经统计，在自动化集装箱堆场中，翻箱操作已成为制约堆场作业效率的最主要因素，需要在出口箱箱位分配时予以着重考虑。

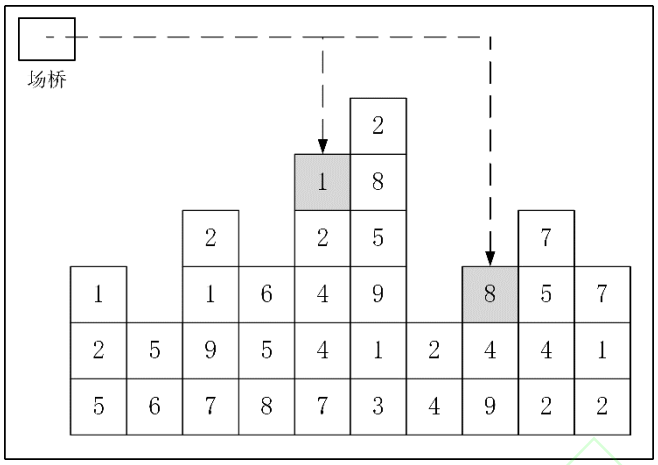


图 4 出口箱出场作业示意图

经过上述分析可知，在自动化集装箱码头出口箱作业中，由于问题规模较大、堆场空间及场桥的约束较多、多箱区、悬臂吊、接力和压箱等特点，出口箱箱位分配问题具有较大的挑战。因此本文以出口箱箱位分配问题为研究对象，以平衡箱区和中转区的负载以及最小化后续翻箱操作为目标，对出口箱的箱区、中转箱位和目标箱位的指派进行优化，达到堆场高效处理出口箱任务的目的。

2 问题建模

2.1 问题假设

1) 由于自动化集装箱码头不断运行，出口箱任务的入场出场等信息需要通过预测等手段得到，存在一定的误差且该误差会随着时间的推移逐渐累积，继而影响最终方案的性能。因此需要引入实时修正的求解方法。基于这一特点，本文采用滚动计划考虑一个时间窗口内的出口箱箱位分配问题，即每次考虑之后一段时间内的箱位分配问题，并在实际采用前一部分方案后重新针对后一部分以及之后一段时间内新的箱位分配问题进行分析，在获取了新的出口箱信息的情况下，尽可能减少上述误差；

2) 由于箱区与泊位存在多对一的关系，因此考虑一个泊位对应的数个箱区的箱位分配问题；

3) 每个箱区中场桥设备是否为悬臂吊的情况已知，且不考虑设备的故障；

4) 每个出口箱的陆侧入场时间、海侧出场时间均已知；

5) 堆场初始堆存状态以及已堆存的出口箱出场时间已知；

6) 本文考虑相同规格的集装箱，以 20 尺集装箱为标准，即一个箱位堆存一个集装箱；

7) 本文不涉及冷藏箱、危险箱等特种箱，仅考虑普通箱的堆存。

2.2 符号定义

2.2.1 参数

$I=\{1,2,3,\ldots,N_I\}$：按入场顺序排列的出口箱集合，通过i,j索引；

$E=\{1,2,3,\ldots,N_I\}$：出口箱后续出场时间集合；

$A=\{1,2,3,\ldots,N_A\}$：泊位对应的箱区集合，通过a索引；

$W=\{1,2,3,\ldots,N_A\}$：泊位对应的箱区的权重集合；

$BT=\{1,2,3,\ldots,N_{BT}\}$：箱区内中转区的倍位集合，通过$bt$索引；

$BG=\{1,2,3,\ldots,N_{BG}\}$：箱区内海侧堆存区域的倍位集合，通过$bg$索引；

$R=\{1,2,3,\ldots,N_R\}$：倍位内列的集合，通过r索引；

$T=\{1,2,3,\ldots,N_T\}$：倍位内层的集合，通过t,ta,tb索引，其中t代表一个出口箱，ta,tb代表两个不同的出口箱；

$TC=\{1,2,3,\ldots,N_{TC}\}$：堆场中中转区已堆存的集装箱集合，通过$tc$索引；

$$PTC_{tc,a,bt,r,t}=\begin{cases}0：第tc个集装箱未放在(a,bt,r,t)箱位中；\\1：第tc个集装箱堆存在(a,bt,r,t)箱位中；\end{cases}$$

$GC=\{1,2,3,\ldots,N_{GC}\}$：堆场中海侧堆存区域已堆存的集装箱集合，通过$gc$索引；

$EC=\{1,2,3,\ldots,N_{GC}\}$：堆场中海侧堆存区域已堆存的集装箱出场时间集合；

$$PGC_{gc,a,bg,r,t}=\begin{cases}0：第gc个集装箱未放在(a,bg,r,t)箱位中；\\1：第gc个集装箱堆存在(a,bg,r,t)箱位中。\end{cases}$$

2.2.2 过程变量

$$yb_{i,j,a,bg,r,ta,tb}=\begin{cases}0：第i,j个出口箱未分配到(a,bg,r,ta)和(a,bg,r,tb)箱位\\1：第i,j个出口箱分配到(a,bg,r,ta)和(a,bg,r,tb)箱位\end{cases},ta>tb;$$

$$yc_{i,gc,a,bg,r}=\begin{cases}0：第i个出口箱和第gc个已堆存箱未分配到(a,bg,r)列；\\1：第i个出口箱和第gc个已堆存箱分配到(a,bg,r)列；\end{cases}$$

$$zb_{i,j,a,bg,r,ta,tb}=\begin{cases}0：第i,j个出口箱不需要翻箱\\1：第i,j个出口箱需要翻箱\end{cases},ta>tb;$$

$$zc_{i,gc,a,bg,r}=\begin{cases}0：第i个出口箱和第gc个已堆存箱不需要翻箱；\\1：第i个出口箱和第gc个已堆存箱需要翻箱。\end{cases}$$

2.2.3 决策变量

$$xa_{i,a,bt,r,t}=\begin{cases}0:第i个出口箱未分配至(a,bt,r,t)中转箱位;\\1:第i个出口箱分配至(a,bt,r,t)中转箱位;\end{cases}$$

$$xb_{i,a,bg,r,t}=\begin{cases}0:第i个出口箱未分配至(a,bg,r,t)目标箱位;\\1:第i个出口箱分配至(a,bg,r,t)目标箱位。\end{cases}$$

2.3 模型构建

$$\min f=f_1+f_2+f_3 \tag{1}$$

$$f_1=\sqrt{\sum_a W_a\Big(\sum_{i,bt,r,t} xa_{i,a,bt,r,t}-\frac{N_I}{N_A}\Big)^2} \tag{2}$$

$$f_2=\sqrt{\sum_a\sum_{bt,r,t}\Big(\sum_i xa_{i,a,bt,r,t}-\frac{\sum_{i,bt,r,t} xa_{i,a,bt,r,t}}{N_{BT}N_RN_T}\Big)^2} \tag{3}$$

$$f_3=\sum_i\left(\max\{zb_{i,j,a,bg,r,ta,tb},zc_{i,gc,a,bg,r}\mid\forall j,gc,a,bg,r,ta>tb\}\right) \tag{4}$$

$$\sum_{a,bt,r,t} xa_{i,a,bt,r,t}=1,\forall i \tag{5}$$

$$\sum_{a,bg,r,t} xb_{i,a,bg,r,t}=1,\forall i \tag{6}$$

$$\sum_{bt,r,t} xa_{i,a,bt,r,t}-\sum_{bg,r,t} xb_{i,a,bg,r,t}=0,\forall i,a \tag{7}$$

$$\sum_i xa_{i,a,bt,r,t}+\sum_{tc} PTC_{tc,a,bt,r,t}\le 1,\forall a,bt,r,t \tag{8}$$

$$\sum_i xb_{i,a,bg,r,t}+\sum_{gc} PGC_{gc,a,bg,r,t}\le 1,\forall a,bg,r,t \tag{9}$$

$$\sum_i xa_{i,a,bt,r,t}+\sum_{tc} PTC_{tc,a,bt,r,t}-\sum_i xa_{i,a,bt,r,t+1}-\sum_{tc} PTC_{tc,a,bt,r,t+1}\ge 0,\forall a,bt,r,t \tag{10}$$

$$\sum_i xb_{i,a,bg,r,t}+\sum_{gc} PGC_{gc,a,bg,r,t}-\sum_i xb_{i,a,bg,r,t+1}-\sum_{gc} PGC_{gc,a,bg,r,t+1}\ge 0,\forall a,bg,r,t \tag{11}$$

$$yb_{i,j,a,bg,r,ta,tb}\ge xb_{i,a,bg,r,ta}+xb_{j,a,bg,r,tb}-1,\forall i,j,a,bg,r,ta>tb \tag{12}$$

$$yb_{i,j,a,bg,r,ta,tb}\le\frac{1}{2}xb_{i,a,bg,r,ta}+\frac{1}{2}xb_{j,a,bg,r,tb},\forall i,j,a,bg,r,ta>tb \tag{13}$$

$$yc_{i,gc,a,bg,r}\ge\sum_t xb_{i,a,bg,r,t}+\sum_t PGC_{gc,a,bg,r,t}-1,\forall i,gc,a,bg,r \tag{14}$$

$$yc_{i,gc,a,bg,r}\le\sum_t\frac{1}{2}xb_{i,a,bg,r,t}+\sum_t\frac{1}{2}PGC_{gc,a,bg,r,t},\forall i,gc,a,bg,r \tag{15}$$

$$\sum_{a,bg,r} yb_{i,j,a,bg,r,ta,tb} \le 0, \forall i < j, ta > tb \tag{16}$$

$$zb_{i,j,a,bg,r,ta,tb} \ge \frac{1}{2}\left(\frac{E_i - E_j}{\mathrm{abs}(E_i - E_j)} + 1\right) + yb_{i,j,a,bg,r,ta,tb} - 1, \forall i, j, a, bg, r, ta > tb \tag{17}$$

$$zb_{i,j,a,bg,r,ta,tb} \le \frac{1}{2}\left(\frac{E_i - E_j}{\mathrm{abs}(E_i - E_j)} + 1\right) yb_{i,j,a,bg,r,ta,tb}, \forall i, j, a, bg, r, ta > tb \tag{18}$$

$$zc_{i,gc,a,bg,r} \ge \frac{1}{2}\left(\frac{E_i - EC_{gc}}{\mathrm{abs}(E_i - EC_{gc})} + 1\right) + yc_{i,gc,a,bg,r} - 1, \forall i, gc, a, bg, r \tag{19}$$

$$zc_{i,gc,a,bg,r} \le \frac{1}{2}\left(\frac{E_i - EC_{gc}}{\mathrm{abs}(E_i - EC_{gc})} + 1\right) yc_{i,gc,a,bg,r}, \forall i, gc, a, bg, r \tag{20}$$

模型的目标函数如式（1）所示，表示最小化箱区负载不均衡度、中转区负载不均衡度和后续翻箱数量的总和。式（2）通过箱区任务数量带权重的标准差表示箱区负载不均衡度、式（3）通过各个箱区中转区各列任务数量的标准差表示中转区负载不均衡度、式(4)表示需要翻箱的集装箱数量。目标中翻箱率的权重较高，体现翻箱数量的重要性。式（5）-（20）为约束条件，其中式（5）-（7）表示每个出口箱对应一个箱区、一个中转区箱位和一个目标箱位。式（8）-（9）表示每个箱位对应一个集装箱。式（10）-（11）表示集装箱不能悬空堆存。式（12）-（15）判断了出口箱之间以及出口箱与已堆存集装箱是否在同一列。式（16）表示先入场的集装箱不能摆放在后入场的集装箱上方。式(17)-(20)判断了出口箱是否摆放在更早出场的集装箱上方，即该出口箱是否需要被翻箱。

基于这一问题模型可以看出在本文研究的自动化集装箱堆场出口箱箱位分配问题中，在本研究的问题假设下，实际问题具备数百任务上千箱位的规模，且过程与决策变量包含多个维度，难以求解。其次由于堆场交互设备较多并且场桥效率有限，需要引入多元的评价方式以平衡各类设备的影响，最大化整体的作业效率。最后，基于三维码放、场桥接力、悬臂吊等特性，问题模型具备了较多种类的约束，极大增加了有效方案的生成难度。因此研究的问题具备多维度评价、约束复杂、规模极大、决策变量较多等特点，从而导致了较高的求解难度。基于这一现状，本文引入新颖且高效的基于强化学习思想的超启发式算法，以期取得较好的箱位分配方案。

3 算法设计

由于本文研究的问题规模极大、约束较多，求解十分复杂，常用的启发式规则、智能优化算法难以获得较满意的箱位分配方案，因此需要引入更高效的优化算法，提高求解方案的性能。针对这一现状，本文开发了基于强化学习思想的超启发式算法（RLHH），从而能够充分利用各种不同启发式

算法的优势。算法可以分为两个层级，即低层算法和高级控制策略。在低层算法中，本文针对问题的特点和可能出现的不同情况引入了多种传统的和新近开发的启发式算法，尽可能的扩大算法适用的场景范围。在高级控制策略中，本文采用了更为新颖的基于策略的强化学习算法 DPPO，根据迭代以及场景状态智能选取合适的启发式算法。在强化学习算法中，本文主要使用了用于加速训练测试过程的 A3C 多线程方法以及用于更快收敛策略梯度的自适应学习率[17,18]。此外，由于深度学习能够提供更大的状态空间和更好的性能，因此本文应用这一技术来计算策略梯度。本文采用算法的原理、流程以及重要的模块如下所示。

3.1 算法原理

自从超启发式的概念被提出以来，它已成为了一种用于合成多种算法的高效的技术。超启发式算法是一种特殊的搜索方法，它从一系列预定义的启发式方法中选择或生成新的启发式方法，而不是直接解决问题[19]。因此它可以被视为一种高层的算法，自动生成适当的启发式算法组合，从而有效地解决问题[20]。近年来有许多研究集中在该领域，并且将这一方法应用于不同的问题，取得了一定的成果[21-24]。

在超启发式算法的框架中，一个关键点是如何高效的在启发式方法空间中开展搜索，选择最恰当的启发式算法。为了解决这一问题，人们引入了强化学习方法。强化学习作为一种强大的决策工具，近些年已引起了广泛关注，在许多领域取得了喜人的成就[25-27]。强化学习方法能够在给定情况下智能选择接下来应采取的行动，并通过与环境的交互来获得最大的回报。在超启发式算法框架中，这一方法能够满足高层控制决策的需求，形成更好的算法[28-30]。但是，当前绝大多数应用还仅限于基于价值的强化学习，而几乎没有基于策略的强化学习方法。由于基于价值的强化学习可能具有对问题的描述能力有限和难以适应随机策略的缺点，并且在启发式算法空间中对这两个方面的要求较高，因此本文使用基于策略的强化学习方法来进一步提高性能。

3.2 算法流程

在 RLHH 算法中，算法首先对当前问题的输入进行处理，得到当前问题矩阵化的表征。其次，算法初始化了不同的启发式算法以及相应的种群。然后，算法根据得到的种群以及当前问题的特点对强化学习的状态参数进行计算，并应用 DPPO 算法得到当前的策略梯度，从而确定下一步采取的启发式算法。之后，算法应用选定的启发式算法开展迭代，不断优化问题的解决方案，并生成优化后的种群。最后，不断重复上述两步，直到满足终止条件。整个算法的伪代码如算法 1 所示。

算法 1:RLHH()

输入:堆场中已有的集装箱及其出场时间 *yard*，时间窗口内的出口箱任务 *task*

输出:算法模型或箱位分配方案

begin

1: 初始化更新前后的演员网络 $actor_{new}()$, *a_para*, $actor_{old}()$, a_para_{old}; 初始化评论家网络 *c_para*

2: 初始化不同的启发式算法 LLH[*i*], *i* =1, 2,…, m

3: 初始化算法参数，包括最大回合数 $episode_{max}$，**回合中的最大步数** dm_{max}，**批尺寸** $batch_{min}$

4: **if** 当前过程是训练过程 **then**

5: 初始化中心线程 t_c，初始化多个工人线程 t_{wi}, *i*=1, 2… w_n 和线程同步变量 *mutex*

6: 状态、动作和汇报缓存 s_b, a_b, r_b← empty

7: **for** *num*← 1 **to**$episode_{max}$**do**

8: **for** *k*← 1 **to**w_n**do**

9: //下面是工人线程 t_{wk} 流程

10: 随机生成新的问题 *status, outbound*, $t_{out}$and t_{in}← *yard, task*

11: 群体 *x*[*i*] ← Rand (), *i=1, 2,..., P*,*P* 为群体中个体数量

12: 个体最优值 *ibest*[*i*] ← *x*[*i*], *i=1, 2,..., P*

13: 状态 *s*←*x* 的评分和个体差异

14: **for***i*← 1 **to**dm_{max}**do**

15: 策略梯度 *w*← $actor_{new}$ (*s*)

16: *a*, s_{new},*r* ← ApplyAction(*w*, *I*, LLH)

17: *s*← s_{new}

18: s_b← s_b + *s*

19: a_b← a_b + *a*

20: r_b← r_b + *r*

21: t_{max}← length of r_b

22: **if**t_{max}>$batch_{min}$**then**

23: //下面是中心线程 t_c 流程

24: *mutex, a_para*, a_para_{old}, *c_para*← TrainNetwork(*mutex, a_para*, a_para_{old},*c_para*)

25: **end**

26: **end**

27: **end**

28: 算法模型← *a_para*, *c_para*

29: **return** 算法模型

30: **else**

31: 问题环境 *status, outbound*, $t_{out}$and t_{in}← *yard, task*

32: 群体 *x*[*i*] ← Rand (), *i=1, 2,..., P*

33: 个体最优值 *ibest*[*i*] ← *x*[*i*], *i=1, 2,..., P*

34: 状态 *s*←*x* 的评分和个体差异

35: **for** $i \leftarrow 1$ **to** dm_{max} **do**
36: 策略梯度 $w \leftarrow \text{actor}_{\text{new}}(s)$
37: $a, s_{new}, r \leftarrow \text{ApplyAction}(w, I, \text{LLH})$
38: $s \leftarrow s_{new}$
39: **end**
40: 最佳个体编号 $best \leftarrow$ 评分最高的 x 的编号
41: 箱位分配方案 $\leftarrow x[best]$
42: **return** 箱位分配方案
end

3.3 超启发式算法设计

为了进一步提高算法的求解性能，本文针对这一实际问题设计了一种新颖的超启发式算法。算法中主要包含了问题映射、低层启发式算法以及高层强化学习控制策略三个关键部分。问题映射主要解决了低层启发式算法与高层强化学习算法之间的环境定义与计算以及本文解决的箱位分配问题与启发式算法之间的编码与解码流程。低层启发式算法主要包含了用于求解箱位分配问题的多种局部搜索算子、全局搜索算子以及智能算法，形成备选的启发式算法库。高层控制策略主要包含了针对问题特点设计的强化学习方法，算法具体内容在下一节中详细阐述。

3.3.1 问题映射

在 RLHH 算法中，主要包含三个层面的内容，即高层强化学习决策算法、低层启发式算法以及出口箱箱位分配问题。其中，前两个内容主要依靠强化学习的环境作为桥梁。由于高级策略只能从环境中学习问题及其变化历史，因此这一环境应当包含低层算法的所有特征。由于强化学习算法将从选择的启发式算法的迭代过程中学习，因此特征是启发式算法的种群及其得分。另外，为了更方便的调用特定的低级启发式方法，环境中还包括了种群中每个个体的最佳历史记录及其分数。

另一方面，后两个内容之间需要依靠特定的编码和解码方式来连接。在 RLHH 中，采用了箱位坐标的方式进行直接编码，即每一个出口箱任务具有箱区、中转箱位和目标箱位三个数值。其中具体箱位采用层次编码形式，即按倍位、排的顺序依次给每个箱位一个数值，并将出口箱将放在这一排的最上方，如图 5 所示。例如(2, 62, 125)为一个出口箱的箱位分配结果，包含三个数字，其中第一个出口箱的箱位分配结果，这一出口箱分配给第 2 箱区，中转箱位为第 7 倍位、第 2 排，目标箱位为第 13 倍位、第 5 排。

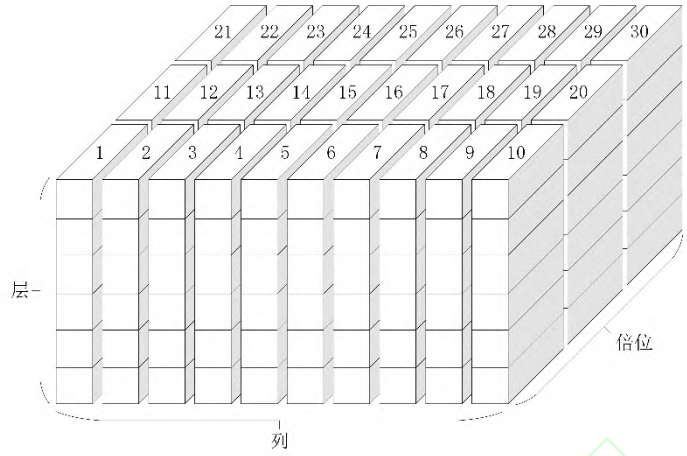


图 5 编码方式

3.3.2 低层启发式算法

针对于上述编码方式，算法将在解码流程中通过模拟的方式计算出每个出口箱的具体位置，之后计算目标函数。由于编码没有限制目标箱位的可行性，因此在启发式算法中需要引入个体修复策略满足问题需求。同时为了提高算法运行性能，个体修复策略将会在个体解码的流程中开展应用。在模拟过程中，当某一出口箱的目标箱位所在排已经堆存了 6 个集装箱，即没有多余空间堆存当前出口箱时，算法将自动就近选取空闲的箱位并将其分配给该出口箱，同时对这一个体进行修改，防止后续迭代过程中再次计算，减小计算量并提高种群中个体的质量。

为了提高 RLHH 算法的性能，算法中采用了局部和全局搜索算子以及智能算法组成低层启发式算法库。其中，任务不平衡性仅取决于出口箱的码放区域，但翻箱率指标取决于堆存现状以及出口箱具体的码放方案。为此，在算法中引入了不同的局部搜索算子，修改一个或数个出口箱的码放位置，在不大幅改变任务不平衡性的基础上尽可能获得更小的翻箱率。基于全局优化的考虑，引入不同的全局搜索算子修改一部分出口箱的码放位置，从而跳出局部最优方案。最后引入研究中常用的且性能已获得实际验证的智能算法，进一步提高算法性能。算法中使用的启发式算法及其简要描述如表 1 所示。

表 1　低层启发式算法

种类	算法	描述
局部搜索算子	破坏算子	将一个出口箱的分配参数之一置为随机值
	累加算子	将一个出口箱的分配参数之一加一。如为最大值，则置为最小值
	交换算子	将两个出口箱的分配参数之一相互交换
	翻转算子	将二至五个出口箱的分配参数之一进行交换与翻转
全局搜索算子	破坏算子	将一部分出口箱的分配方案置为随机值
	翻转算子	将一部分出口箱的分配方案进行翻转

	移位算子	将一部分出口箱的分配方案向左移动一位
智能算法	布谷鸟算法	以巢穴坐标的排序作为解决方案的编码，包含列维飞行和抛弃模块
	遗传算法	每次迭代包含两两选择、简单交叉、随机变异和精英策略模块
	粒子群算法	以粒子坐标的排序作为解决方案的编码并限制粒子范围提高求解性能
	人工蜂群算法	包含随机选取局部搜索算子的雇佣蜂和跟随蜂以及随机选位的侦察蜂

3.4 强化学习算法设计

由于常规方法难以根据不同问题合理地选择不同低层启发式算法、取得良好的性能，本文引入基于策略的 DPPO 强化学习方法作为高层决策算法，以期得到满意的算法性能。考虑到应用强化学习的一项重要条件是问题的决策过程具备无后效性，即某阶段的状态一旦确定，则此后过程的演变不再受此前各状态及决策的影响，本文首先针对问题特点进行了分析。针对高层决策算法这一模块，它考虑的是如何合理应用不同的低层启发式算法来改进当前箱位分配问题的解。一方面，在低层启发式算法改进的过程中，基于搜索算子和智能算法的定义，算法仅考虑当前求解方案的编码以及对应的评价指标，采用不同策略修改编码并提升指标，而与求解方案的历史变化无关。另一方面，在选择合适的低层启发式算法时，考虑到上述改进措施的特点，相关历史信息不会对算法的迭代造成影响，因此算法也仅需考虑当前种群中各个求解方案的编码以及指标，从而满足决策过程的无后效性。在满足应用强化学习条件的基础上，本文针对问题以及低层启发式算法的特点针对性开发了状态、动作、回报、神经网络结构和训练策略等模块。这些模块的具体内容如下所示。

3.4.1 状态

在算法迭代过程中，每次启发式算法对种群进行每次更新后，强化学习策略都会计算环境状态，指导下一步的迭代。由于在不同的种群状态即质量与分散程度下不同的启发式算法存在性能差异，算法采用了个体得分和个体差异性作为状态内容，即 $s = \{f, d\}$。f 是归一化的种群中每个个体的得分，即当前箱位分配方案的目标函数值，其大小为 1xn，且 n 是个体数量。d 表示所有个体之间的差异，为一个大小为 nxn 的矩阵，且每个数字 d_{ij} 由式（21）得到，表示种群中 ind_i 和 ind_j 之间的差异性。

$$d_{ij} = \frac{\left|\left\{m \in \{1,\cdots,n\} \middle| ind_i(m) \neq ind_j(m)\right\}\right|}{n} \tag{21}$$

3.4.2 动作

在强化学习的学习和运行过程中，需要在每个决策时刻选择下一步迭代的启发式算法。因此，本文中强化学习算法的动作为特定的启发式算法，以 $a = \{0, 1, 2, \cdots, m\}$ 表示，其中 m 是启发式方法的数量。此外，本文采用ε-贪婪的方法来平衡动作选取的探索和利用，在不同情况下选择正确

的行动。在每个决策时刻，算法会根据之前计算得到的不同动作的策略梯度进行动作选择。算法会生成一个随机数，如果该数字大于固定概率ε（$0\leqslant\varepsilon\leqslant1$），则将贪婪地选择动作，即选择策略梯度最大的那个动作，否则，算法将随机选择动作。其中该参数采用研究中常用的 0.8，平衡贪婪与随机选择之间的比例。选择了下一步的动作后，选定的启发式算法将更新种群，并连续运行几代，以消除随机误差。另外，为了更高效的运行启发式方法，算法保留了每个个体各次迭代中的最优值，记为 *ibest*。最后，算法将计算新的状态和奖励用于后续迭代与网络训练。动作选择和应用的伪代码如算法 2 所示。

算法 2: ApplyAction()

输入:策略梯度 w，当前决策时刻序号 i 以及低层启发式算法 LLH
输出:动作 a、新的状态 s_{new} 和回报 r
begin
1: *temp* ← Rand (range = [0,1])
2: **if** *temp*> 0.8 **then**
3: a ← Randint (range = [0, m - 1], weight = *w*)
4: **else**
5: *a*← Randint (range = [0, m - 1])
6: 选择的启发式算法 *MH*← LLH[*a*]，并初始化运行代数 gen_{max}
7: 新的种群 x_{new}← *x*
8: 新的个体最优值 $ibest_{new}$← *ibest*
9: **for** *j*← 0 **to**gen_{max}**do**
10: x_{new}, $ibest_{new}$← *MH* (x_{new}, $ibest_{new}$)
11: **end**
12: *x*← x_{new}
13: *ibest*← $ibest_{new}$
14: s_{new}←x_{new} 的评分和个体差异性
15: **if***i*<dm_{max}**then**
16: *r*← 0
17: **else**
18: *r*← 整个过程中的评分提升
19: **return** *a*, s_{new} and *r*
end

3.4.3 回报

回报定义了强化学习策略的目标，并根据状态变化提供了选取动作的即时和后续效果。在此过程中，强化学习算法选择动作以最大化总回报，并尝试在回报的指导下达到最终目标。在本文研究的问题中，总体目标是找出最优的箱位分配方案，最小化任务的不平衡性以及后续的翻箱率。因此，这一评价指标将会随着算法迭代而持续下降，直到最终获得较好的分配方案。在算法中，为了更好

的表征策略的好坏，回报被定义为最终状态与初始状态之间的评价方案的差值，并在最后给予算法。

3.4.4 训练方式

RLHH 算法引入基于策略的演员-评论家框架以提取状态中的隐藏模式，因此算法是一种在线学习（onpolicy）方法。由于传统的演员-评论家框架需要较长时间进行训练，因此近年来人们引入了 A3C 方法[31]。A3C 方法是一种新颖的强化学习训练方式，可以充分利用多线程来加速训练过程，以满足实际问题的需要。在提出的算法中，系统包括一个中心线程和多个工人线程。中心线程负责更新和训练网络，并为工人线程提供网络参数。工人线程使用来自中心线程的网络参数并且并行运行算法，以收集不同情况下的状态、动作和奖励。这一并行训练过程可以大大加快训练过程，并显著减少运行时间。

在工人线程收集了足够的训练数据之后，算法将对神经网络进行训练，并从所有工人线程的经验即每个决策时刻的状态，行动和奖励中学习。训练过程在中心线程运行，第一步是使用称为事件的线程同步方法挂起所有工人线程。然后，中心线程使用所有收集的数据来训练演员和评论家网络。对于演员网络，算法采用式（22）所示的目标进行更新[18]。

$$L^{CLIP}(\theta)=\hat{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\mathrm{clip}\left(r_t(\theta),1-\varepsilon,1+\varepsilon\right)\hat{A}_t\right)\right] \tag{22}$$

式中，$r_t(\theta)$是新策略与旧策略的比值，可以使用新的策略梯度网络和旧的策略梯度网络进行计算。A_t 是预测的差异，从收集的数据中获得的回报与评论家网络计算出的相应状态的价值之间的差异得到。对于评论家网络，训练过程相对容易。这一网络的损失是上述预测的差异，可以表现状态值的预测性能。最后，在演员和评论家网络都完成训练之后，中心线程使用事件告诉所有工人线程可以使用新的网络参数来继续收集数据。训练过程的伪代码如算法 3 所示。

算法 3: TrainNetwork()

输入:事件 *mutex*，网络参数 *a_para*, a_para_{old}, *c_para*

输出:事件 *mutex*，更新的网络参数 *a_para*, a_para_{old}, *c_para*

begin

1: *mutex* ← 等待状态
2: a_para_{old}← *a_para*
3: t_{max}←r_b 的长度
4: 状态的价值 v_s← critic (s_b)
4: **for***j*← 1 **to**t_{max}**do**
5: s← $s_b[j]$, a← $a_b[j]$, r← $r_b[j]$, v← $v_s[j]$
6: 预测差异 *adv*← $r - v$

7: 新的策略梯度 *newp*← $\text{actor}_{\text{new}}$ (*a*)
8: 旧的策略梯度 *oldp*← $\text{actor}_{\text{old}}$ (*a*)
9: *ratio*← *newp*/*oldp*
10: 演员网络的损失 *aloss* 由式(22)计算得到
11: 训练演员网络并更新 *a_para*
12: 评论家网络的损失 *closs* ← *adv*
13: 训练评论家网络并更新 *c_para*
14: **end**
15: s_b, a_b, r_b← empty
16: *mutex* ← 就绪状态
17: **return***mutex*, *a_para*, a_para_{old}, *c_para*

end

3.4.5 神经网络结构

在演员-评论家框架中，演员网络在每个决策时刻计算策略梯度，而评论家网络预测状态的价值以教导演员网络从而得到更好的策略梯度。两个网络共享相同的状态输入，即种群的质量与分散程度。但两个网络具备不同的输出，其中演员网络输出计算动作需要的策略梯度，而评论家网络输出状态的价值，用于指导演员网络的迭代。因此，本文采用相似的神经网络结构来搭建演员和评论家网络。输入的状态首先会分为个体分数和个体差异两部分。然后，这两部分分别经过两次卷积层和池化层以提取其中的隐藏信息，定义当前种群的特点。之后，网络将结果合并为完整的数据并通过全连接层将所有信息汇总在一起，形成状态的分类。最后，网络使用归一化技术并输出。

4 实例验证

为了验证提出的 RLHH 算法的有效性及其性能，本文基于洋山四期自动化集装箱堆场的实际场景设计了一系列不同规模的问题。所有实验均在 TensorFlow 1.4，Python 3.5、16GB RAM 和 i7-8700 CPU 的环境下实现。同时，为了让算法的性能达到最优，本文通过实验得到 RLHH 的超参数，如表 2 所示。最后，为了更直观的获得提出算法的性能，本文引入不同问题中常用的性能较高的一系列智能算法如遗传算法、模拟退火算法[7,15]等进行算法比较。

表 2 超参数选择

属性	取值
工人线程数量	6
回报衰减率	0.9
学习率	0.0001
更新批次数量	40
损失范围	0.2

神经元数量	128
种群大小	48
迭代代数	60

4.1 算例生成

本文算例的数据主要根据以下规则产生：

1. 问题规模：根据洋山四期自动化集装箱堆场的布局方案，生成对应一个泊位的数个箱区的场景。为了兼顾有限的运算资源，设计了较小规模的实验以更好验证算法性能以及根据堆场轨道吊工作效率以及出口箱任务数量生成大规模的实际箱位分配问题。

2. 箱区现状：根据堆场箱位实际的利用率，初始时刻每个箱区内已存放 50%的集装箱，已堆存集装箱的箱位随机产生。

3. 出口箱入场顺序：由于外集卡的到达时间随外围交通、外集卡自身时间安排决定，不存在必须遵守的约束，因此每个出口箱的入场顺序随机产生。

4. 作业顺序：根据船公司提供的船舶预配载图以及预先决定的桥吊作业顺序和船舶积载方案，预先制定海侧轨道吊的作业顺序，即各集装箱的海侧出场作业顺序。

4.2 算法结果分析

由于本文提出的 RLHH 算法中采用强化学习方法作为高层策略，因此需要前期的训练过程以最大化算法性能。在此过程中，本文引入了多线程并行方式以加速训练过程。在训练过程中，本文采用上述方法生成的问题来提供训练算例，并基于问题的复杂程度设置了对应的训练数据集。训练数据集包含了随机生成的 900 个不同规模的算例，包括 300 个 3 个箱区较小倍位数的小规模算例、300 个 3 个箱区较大倍位数的中等规模算例以及 300 个 7 个箱区的大规模算例。由于每个算例中均需要强化学习算法进行数十次的算法决策，因此上述训练数据集能够满足算法的训练要求。在训练过程结束后，使用生成的强化学习模型开展测试以验证其性能。为了直观的体现算法迭代过程，以 3 个箱区 15 个倍位的堆场和 60 个出口箱任务为例，使用提出的算法开展研究，算法迭代过程如图 6 所示。

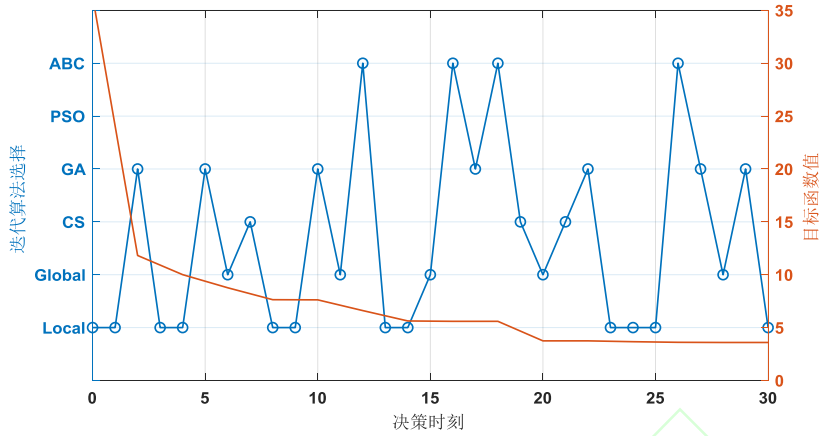


图 6　算法迭代过程

由此可以得到在迭代过程中，目标函数值不断下降，并且算法能够选择不同的启发式算法来尽可能增加这一下降率。具体来说，在早期，由于种群随机产生，算法倾向于选择局部搜索算子、遗传算法、人工蜂群算法来寻找局部最优值。之后，在找到局部最优值并且最优值不变时，算法改变策略，选择全局搜索算子或具有强大探索能力的布谷鸟算法。然后，由于找到了新的搜索领域，算法重新选择局部搜索能力较强的算法来寻找新的局部最优。之后算法不断重复该过程，直到输出最终解决方案为止。因此，提出的 RLHH 算法可以在有限的迭代过程内获得较好的结果，也证明了提出算法的有效性。

4.3　算法比较分析

为了进一步研究 RLHH 算法用于出口箱箱位分配问题的有效性和性能，本文建立了一系列箱位分配问题算例，并引入其他常见的智能算法如人工蜂群算法（ABC）、布谷鸟算法（CS）、遗传算法（GA）、粒子群算法（PSO）和模拟退火算法（SA）以及近期研究中采用的改进的遗传算法(IGA)[11]、改进的模拟退火算法(ISA)[15]开展对比试验，从而比较算法性能并验证本文算法使用的必要性和优越性。由于求解问题存在差异性，相关对比算法在近期研究的参数选择基础上均采用了正交实验来获取最佳的参数组合，相关参数如表 3 所示。为了更完整的得到算法性能对比结果，本文采用控制变量的方法来设计算例。由于问题规模由箱区数量、倍位数量以及出口箱任务数量决定，因此全部的算例也分为三个部分。整个实验的结果如表 4 所示。其中对于每个算例，本文控制了每个算法的计算时间以确保所有算法使用相同的计算能力并能用于实际问题，参数计算时间表示了实验中算法的最大计算时间。此外，参数中名为“X-Y-Z”的算例表示 X 个箱区 Y 个倍位的堆场以及 Z 个出口箱任务。

表 3　对比算法配置

对比算法	配置信息

ABC	搜索阈值 25
CS	莱维系数 1.5，遗弃概率 0.25
GA	交叉概率 0.6，变异概率 0.4
PSO	惯性因子 0.9，加速常数 1.5 与 2
SA	初始温度 50，结束温度 0.1，退火速率 0.9，迭代次数 100
IGA	交叉概率 0.85，变异概率 0.05
ISA	初始温度 99，结束温度 1，退火速率 0.9，迭代次数 1200

表 4　算法对比结果

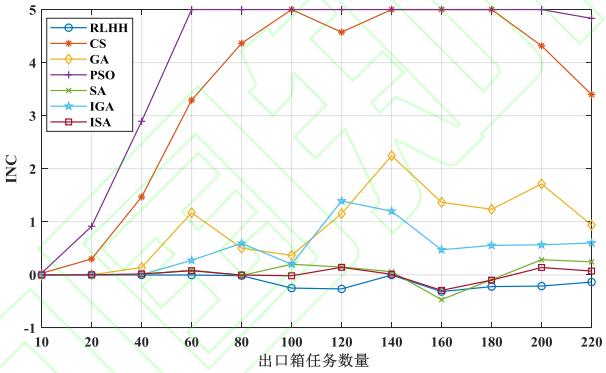
Instance	CPU time(s)	RLHH	ABC	CS	GA	PSO	SA	IGA	ISA
3-15-10	10	**4.445**	4.458	4.586	**4.445**	4.601	4.458	4.458	**4.445**
3-15-20	30	**4.559**	**4.559**	5.922	**4.559**	8.732	**4.559**	**4.559**	**4.559**
3-15-40	90	**3.827**	3.843	9.486	4.374	14.963	3.861	3.860	3.905
3-15-60	180	**3.592**	3.607	15.485	7.830	24.579	3.922	4.592	3.884
3-15-80	270	**3.254**	3.310	17.755	4.975	31.555	3.268	5.275	3.295
3-15-100	360	**3.365**	4.487	28.974	6.132	47.822	5.379	5.385	4.405
3-15-120	450	**5.274**	7.186	40.052	15.490	62.934	8.245	17.186	8.202
3-15-140	540	**3.330**	3.348	40.049	10.862	76.463	3.542	7.362	3.380
3-15-160	630	4.375	6.399	54.747	15.149	88.514	**3.404**	9.418	4.521
3-15-180	720	**7.083**	9.104	69.858	20.338	96.621	8.273	14.158	8.197
3-15-200	810	**11.188**	14.207	75.519	38.561	106.321	18.259	22.227	16.157
3-15-220	900	**18.810**	21.789	95.805	42.201	127.107	27.096	34.869	23.333
3-3-30	60	**4.073**	**4.073**	6.354	6.983	11.158	5.257	5.073	5.705
3-9-90	300	**9.510**	10.632	21.178	15.352	47.246	12.893	12.510	11.601
3-15-150	500	**10.721**	13.633	48.762	26.566	87.395	12.661	19.653	12.844
3-21-210	700	**10.439**	13.619	76.752	23.160	122.037	12.652	15.585	12.585
3-27-270	900	**11.454**	14.605	91.868	31.254	157.071	21.706	20.515	17.045
3-30-300	1050	**11.572**	13.524	121.746	19.580	166.771	12.801	18.464	12.572
3-34-340	1200	**9.492**	16.750	127.534	26.392	202.704	12.960	23.430	11.448
3-38-380	1350	**15.276**	17.381	161.412	23.858	222.527	20.284	18.094	17.529
3-42-420	1500	**15.259**	19.381	166.608	26.335	245.063	26.552	25.037	23.113
3-46-460	1650	**27.317**	32.367	171.940	36.722	275.519	35.281	31.906	29.210
3-50-500	1800	**20.041**	24.449	178.605	27.386	289.154	25.224	23.546	21.649
3-50-498	1800	**12.595**	15.737	186.918	21.316	287.421	19.004	18.317	19.337
4-50-664	2250	**26.667**	31.087	243.917	39.897	402.808	33.186	31.353	27.925
5-50-830	2700	**73.020**	92.869	530.519	113.488	555.951	88.144	102.966	82.486
6-50-996	3150	**55.346**	69.078	633.346	82.153	623.292	76.816	72.970	65.711
7-50-1162	3600	**86.622**	106.104	736.439	120.769	760.012	108.061	102.485	95.636
Average		**16.875**	20.771	141.505	29.147	183.798	22.062	24.116	19.910

根据表 4 的实验结果，可以得出以下结论。在自动化集装箱码头出口箱箱位分配问题中，本文提出的 RLHH 算法能够在有限的时间内得到较好的可接受的箱位分配方案。算法能够在全部 28 个算

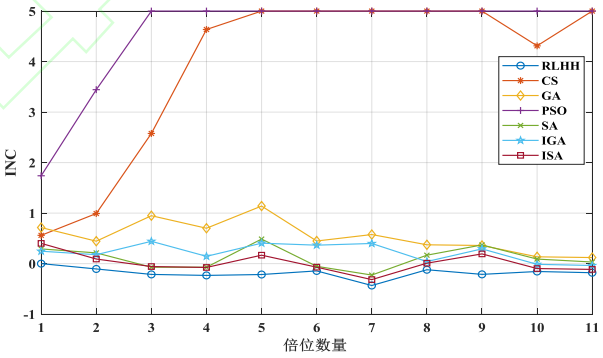
例中的 27 个获得最佳的箱位分配方案。而在“3-15-60”算例中，模拟退火算法得到了最佳的箱位分配方案。此外，在全部算例的箱位分配方案的平均结果中，RLHH 算法获得了最好的算法评价，较其他常见的智能算法有 18.8%以上的提升，较近期研究中改进的智能算法有 15.2%的提升。因此可以看出提出的 RLHH 算法具备比单一智能算法更好的性能，能够更好求解研究的出口箱箱位分配问题。

另外，为了更直观的比较提出的 RLHH 算法与智能算法的算法性能，本文通过式（23）计算算法之间的性能差距，其中 X 表示当前算法，T(X)表示 X 算法得到的目标函数的平均值。由于表 6 中人工蜂群算法在常见智能算法中的表现最好，因此采用这一算法为性能基准，即以当前算法与人工蜂群算法的性能对比表现算法之间的性能差距。此外，由于目标函数值越小意味着箱位分配方案越合理，因此 INC 值越低意味着算法性能越好。根据三组算例得到的算法对比图如图 7 所示。

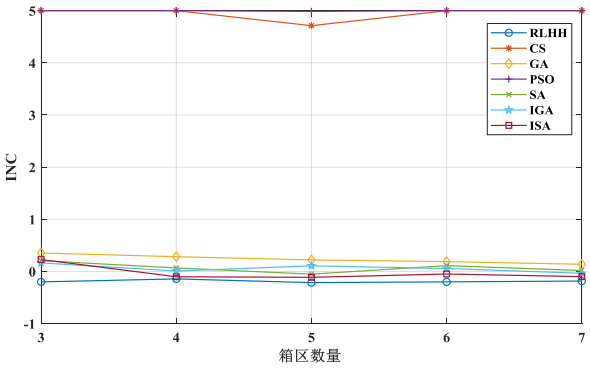
$$INC = \min\left\{\frac{T(X)-T(\mathrm{ABC})}{T(\mathrm{ABC})},5\right\} \tag{23}$$



a　出口箱任务数量增加



b　倍位数量增加



c　箱区数量增加

图 7　算法性能对比

从图 7 的三个部分算法对比中可以看出，所提出的算法的 INC 值在 0%至-27%之间，并且在大多数情况下具有最低的 INC 值。此外，随着问题规模的增加，RLHH 算法的 INC 值存在下降趋势。因此可以得出结论，RLHH 算法具有最佳的性能，并且能在较大规模的实际问题中取得更好的效果，具备应用价值。

总之，从上述实验可以得出以下一些结论。本文提出的 RLHH 算法能够根据当前迭代情况智能选取合适的启发式算法进行接下来的迭代，从而得到更好的箱位分配方案。同时，针对研究的箱位分配问题，提出的算法能够获得较常见的智能算法更好的箱位分配方案，在不同情况下的性能提高了 15%。并且在较大规模的实际问题中，提出的算法能够具备更好的效果，提供具备使用价值的高性能计划方案。

5　结束语

在自动化集装箱码头的出口箱箱位分配问题中，由于具有问题规模极大、动态性强、场桥接力、多箱区协同等特性，现有研究存在较大的提升空间。为了更好解决这一问题，本文基于滚动计划的方法，构建了一个以最小化任务不均衡性和翻箱率为目标的出口箱箱位分配问题模型。在堆场初始堆存状态、出口箱入场和出场顺序已知的情况下，提出了一种基于强化学习的超启发式算法得到箱位分配方案。该方案将强化学习和深度学习用作高级选择策略，将启发式算子、智能算法用作低级启发式方法。经过不同规模的实验，并与常规及近期研究采用的智能算法进行对比，证明了本文提出算法能够有效解决这一箱位分配问题，并且在算法表现上优于不同的智能算法，从而表明提出的算法能够提供高水平的箱位分配问题解决方案。

尽管本文研究解决了这一具有挑战性的出口箱箱位分配问题，但将来可以尝试更多的工作来进

一步改善算法，以实现更高的效率和性能。第一个改进方向是在强化学习策略中开发更有效的网络结构和框架。另一个方向是引入更多更有效的低级启发式算法，包括但不限于智能算法、启发式算子和混合算法。最后，可以考虑更复杂的堆场场景和问题，从而使提出的算法能够克服更多的挑战。

参考文献

[1] LIANG C, GU T, LU B, et al. Genetic mechanism-based coupling algorithm for solving coordinated scheduling problems of yard systems in container terminals[J]. Computers & Industrial Engineering, 2015, 89: 34-42.

[2] JIANG X J, JIN J G. A branch-and-price method for integrated yard crane deployment and container allocation in transshipment yards[J]. Transportation Research Part B: Methodological, 2017, 98: 62-75.

[3] LIANGChengji,JIA Ru,SHENG Yang. Strategy for storage space allocation in automated container terminal based on network flow problem[J].Computer Applications and Software, 2018, 35(1): 77-84 (in Chinese).
[梁承姬, 贾茹, 盛扬. 基于网络流的自动化集装箱码头堆场空间分配[J]. 计算机应用与软件, 2018, 35(1): 77-84.]

[4] YANG Y, ZHU X, HAGHANI A, et al. Multiple equipment integrated scheduling and storage space allocation in rail-water intermodal container terminals considering energy efficiency[J]. Transportation Research Record, 2019, 2673(3): 199-209.

[5] JIN Zhihong, WANG Li, XING Lei, et al. Collaborative optimization of yard block allocation and yard crane allocation considering truck appointment information[J]. Journal of Dalian Maritime University, 2019, 45(3): 1-8 (in Chinese).
[靳志宏, 王莉, 邢磊, 等. 考虑集卡预约信息的堆场箱区分配和场桥配置协同优化[J]. 大连海事大学学报, 2019, 45(3): 1-8.]

[6] NIMinmin, CHANGDaofang.Storage space allocation based on dynamic strategy in container terminal yard[J]. Application Research of Computers, 2019, 36(8): 2363-2367 (in Chinese).
[倪敏敏, 苌道方. 基于动态平衡策划下的集装箱堆场箱位分配方案研究[J]. 计算机应用研究, 2019, 36(8): 2363-2367.]

[7] HANXiaole, XU Ke, LU Zhiqiang. Robust scheduling of berth and yard resource template in container terminal[J]. Computer Integrated Manufacturing Systems, 2020, 26(3): 784-794 (in Chinese).
[韩笑乐,许可,陆志强. 集装箱码头泊位—堆场资源的鲁棒性模板决策[J]. 计算机集成制造系统, 2020, 26(3): 784-794.]

[8] CARLO H J, VIS I F, ROODBERGEN K J, et al. Storage yard operations in container terminals: Literature overview, trends, and research directions[J]. European Journal of Operational Research, 2014, 235(2): 412-430.

[9] BOYSEN N, EMDE S. The parallel stack loading problem to minimize blockages[J]. European Journal of Operational Research, 2016, 249(2): 618-627.

[10] GOERIGK M, KNUST S, LE X T, et al. Robust storage loading problems with stacking and payload constraints[J]. European Journal of Operational Research, 2016, 253(1): 51-67.

[11] ZENG Jianzhi, ZHANG Zhiying, XING Yan, et al. Block stockyard scheduling and optimization with single time window based on dual-layer genetic algorithm[J]. Computer Integrated Manufacturing Systems,2016,22(9):2165-2174 (in Chinese).
[曾建智,张志英,邢艳, 等.基于双层遗传算法的单时间窗分段堆场调度计划与优化[J].计算机集成制造系统,2016,22(9):2165-2174.]

[12] ZHOUPengfei, ZHAO Jinqiu. Appointment and graph representation based export container slot optimization[J]. Control and Decision, 2017, 32(10): 1914-1920 (in Chinese).
[周鹏飞, 赵金秋. 基于预约和图表示的集装箱出口箱位优选[J]. 控制与决策, 2017, 32(10): 1914-1920.]

[13] SHEN Y, ZHAO N, XIA M, et al. A Deep Q-Learning Network for ship stowage planning problem[J]. Polish Maritime Research, 2017: 102-109.

[14] LI Suikai, LI Yitao, SUN Weiwei, et al. A storage space allocation algorithm for export containers in automated container terminals[J]. Computer Engineering, 2018, 45(5): 272-278,284 (in Chinese).
[李隋凯, 励益韬, 孙未未, 等. 一种自动化集装箱码头出口箱进箱选位算法[J]. 计算机工程, 2018, 45(5): 272-278,284.]

[15] CHANG Y, ZHU X. A Novel Two-Stage Heuristic for Solving Storage Space Allocation Problems in Rail-Water Intermodal Container Terminals[J]. Symmetry, 2019, 11(10).

[16] MALDONADO S, GONZALEZRAMIREZ R G, QUIJADA F, et al. Analytics meets port logistics: a decision support system for container stacking operations[J]. Decision Support Systems, 2019: 84-93.

[17] HEESS N, DHRUVA T B, SRIRAM S, et al. Emergence of locomotion behaviours in rich environments[J]. arXiv: Artificial Intelligence, 2017.

[18] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms[J]. arXiv: Learning, 2017.

[19] PILLAY N, QU R. Hyper-heuristics: theory and applications[M]. Springer International Publishing, 2018.

[20] EPITROPAKIS M G, BURKE E K. Hyper-heuristics[J]. Handbook of Heuristics, 2018: 489-545.

[21] TSAI C, HUANG W, CHIANG M, et al. A hyper-heuristic scheduling algorithm for cloud[C]. IEEE International Conference on Cloud Computing Technology and Science, 2014, 2(2): 236-250.

[22] TYASNURITA R, OZCAN E, JOHN R, et al. Learning heuristic selection using a time delay neural network for open vehicle routing[C]. Congress on Evolutionary Computation, 2017: 1474-1481.

[23] AHMED L N, MUMFORD C L, KHEIRI A, et al. Solving urban transit route design problem using selection hyper-heuristics[J]. European Journal of Operational Research, 2019, 274(2): 545-559.

[24] ZHOU Y, YANG J, ZHENG L, et al. Hyper-heuristic coevolution of machine assignment and job sequencing rules for multi-objective dynamic flexible job shop scheduling[J]. IEEE Access, 2019: 68-88.

[25] LIN X, BELING P A, COGILL R, et al. Multiagent inverse reinforcement learning for two-person zero-sum games[J]. IEEE Transactions on Computational Intelligence and AI in Games, 2017, 10(1): 56-68.

[26] NAZARI M, OROOJLOOY A, SNYDER L V, et al. Reinforcement learning for solving the vehicle routing problem[C]. Neural Information Processing Systems, 2018: 9861-9871.

[27] HWANG I, JANG Y J. Q(λ) learning-based dynamic route guidance algorithm for overhead hoist transport systems in semiconductor fabs[J]. International Journal of Production Research, 2020, 58(4): 1199-1221.

[28] SGHIR I, JAAFAR I B, GHEDIRA K, et al. A multi-agent based hyper-heuristic algorithm for the winner determination problem[J]. Procedia Computer Science, 2017: 117-126.

[29] LI W, OZCAN E, JOHN R, et al. A learning automata-based multiobjective hyper-heuristic[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(1): 59-73.

[30] CHOONG S S, WONG L, LIM C P, et al. Automatic design of hyper-heuristic based on reinforcement learning[J]. Information Sciences, 2018: 89-107.

[31] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning[C]. International Conference on Machine Learning, 2016: 1928-1937.

作者简介：

黄子钊(1996-)，男，上海人，硕士研究生，研究方向：生产计划与调度控制、智能优化算法，E-mail:huangzz96@sjtu.edu.cn；

庄子龙(1995-)，男，湖北襄阳人，博士研究生，研究方向：复杂系统建模优化，机器学习与工业智能；

滕 浩(1996-),男，河北承德人，硕士研究生，研究方向：物流调度，计算智能；

+秦 威(1982-)，男，湖北黄冈人，副教授，博士生导师，研究方向：复杂系统建模、控制与优化，机器智能理论、方法与应用，通信作者，E-mail:wqin@sjtu.edu.cn；

秦 涛(1983-),男，湖南道县人，工程师，学士，研究方向：集装箱码头智能操作系统；

邹 鹰(1972-),女，湖北武汉人，工程师，硕士，研究方向：集装箱码头智能操作系统。