doi:10.3772/j.issn.1002-0470.2021.07.003

基于蒙特卡洛树搜索的智能天车倒垛优化方法①

董 砚② 康学斌 雷兆明③ 卢 禹

(河北工业大学人工智能与数据科学学院 天津 300130)

摘 要 智能天车倒垛优化是提高钢卷库堆场利用率的重要手段,同时对提升钢铁仓库物流效率具有重要意义。针对该问题,建立最小倒垛次数为目标的天车作业负荷数学模型。在对模型求解过程中,借鉴了 Alpha Go-Zero 中树搜索方法,设计了蒙特卡洛钢卷搜索树(MCRST)。为了提升搜索树的收敛速度和结果的准确性,将树的置信度上界(UCT)改为快速动作值估计(RAVE),同时引入绝对剪枝策略避免节点盲目扩展。通过不同规模算例实验,将改进算法与原树搜索和粒子群算法(PSO)进行比较,证明了该算法在大规模问题上的优越性;同时该算法也考虑了订单钢卷出库顺序和出库量等因素,验证了算法的适用性。

关键词 钢卷库;倒垛;天车作业;蒙特卡洛树搜索(MCTS);出库任务

0 引 言

随着经济全球化的不断深入,我国政府为了促进经济增长提出“中国制造 2025”计划[1]。智能天车在钢卷库出库任务中由于钢卷在垛位堆放顺序以及钢材类型的不同,无法直接拿出目标钢卷,由此会产生倒垛现象。倒垛次数越多,天车的作业负荷越重,钢卷的出库效率越低[2],因此,倒垛优化是企业产品流通中迫切需要解决的问题。

关于倒垛问题的研究现在大多集中在港口集装箱和钢铁行业板坯仓库的堆场中,钢卷由于单个物件重量大、不便运输以及堆放方式与集装箱和板坯不同,目前关于钢卷倒垛优化问题的研究不多。在关于集装箱堆场研究中,文献[3]针对船厂钢板堆场出入库调度研究,提出基于模拟退火接收准则的双层遗传算法,但在完善堆场利用率方面仍需改进。文献[4]研究调整集装箱在存储区的堆栈顺序,价值大的、重量大的优先装载,证明集装箱装载是 NP-hard 问题。在关于板坯仓库堆场研究中,文献[5]通过优化单天车调度来确定阻碍板坯的倒垛位置和目标的拣选顺序,提出的混合亚启发式算法可以有效解决中大规模问题。文献[6]通过减少钢筋的倒垛次数来降低物流成本,设计了最低槽位启发式算法和就近槽位选择算法来最大程度减少改组操作,对实际调度有指导意义。文献[7]将板卷装配计划与运输调度进行集成优化,通过随机算例的测试结果证明了两层结构多目标变邻域搜索算法的有效性。文献[8-9]采用启发式树搜索的方法来解决顺序倒垛问题对本文有借鉴意义。

由于倒垛优化问题具有 NP-hard 特性,精确算法对大规模问题求解非常困难,针对此问题以往研究中大多采用启发式算法和遗传算法[10],此外倒垛优化问题不仅需要决策如何取出所需钢卷,对于叠压钢卷位置也需要提前做出决策,避免产生二次倒垛。本文将人工智能算法蒙特卡洛树搜索(Monte Carlo tree search, MCTS)应用在钢卷倒垛优化问题,在建模中将找寻目标钢卷的过程可以看作是单人游戏,订单中的目标钢卷转化为序贯决策问题进行求

① 河北省创新能力提升计划(18961604H)和河北省自然科学基金(F2018202206)资助项目。
② 女,1973 年生,博士,教授;研究方向:工程系统与控制;E-mail: 1992600450@qq.com
③ 通信作者,E-mail: dr_lei@foxmail.com
(收稿日期:2020-07-07)

解;在算法搜索过程中将改进的 UCT(upper confidence bounds for trees)策略和增加的剪枝策略应用其中,以此设计了蒙特卡洛钢卷搜索树(Monte Carlo roll search tree, MCRST)来快速、准确地选择出库任务中的目标钢卷。

1　模型建立

1.1　钢卷倒垛问题描述

钢卷库根据订单中钢卷的种类与数量从库房中选取合适的钢卷,在选取钢卷的过程中,如果所需钢卷位于最顶层,则直接把钢卷取出,否则需要把所需钢卷以上的其他钢卷移除,为了方便本文对问题描述引入下面的定义。

目标钢卷是订单任务需要的钢卷;

叠压钢卷是叠压在目标钢卷上的钢卷;

钢卷簇是规格相同的钢卷集合;

倒垛是将叠压钢卷移动到其他位置为一次成功倒垛,将目标钢卷取出也是一次成功倒垛。

本文研究的钢卷库以双层金字塔形式叠加堆放,钢卷存放分为上下层,下层钢卷依靠地面鞍座固定,相邻两个下层钢卷形成一个上层钢卷存放位置。钢卷库中每个钢卷的位置信息由区号、行数、列数和层数(m, l, f, t)表示。例如位置(1,5,4,2)表示第 1 区第 5 行第 4 列第 2 层的钢卷。钢卷位置信息如图 1 所示。

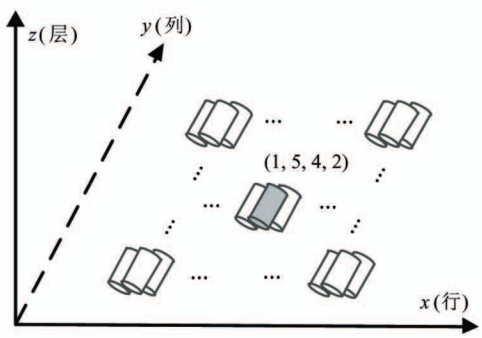

图 1　钢卷位置信息

为便于问题描述,在实例中钢卷采用序号表示,目标钢卷在堆场中的位置情况如图 2 所示。图中灰色钢卷为目标钢卷,在图 2(a)和图 2(b)中,位于第 1 层的 1 号目标钢卷和位于第 2 层的 5 号目标钢卷可以直接取出,倒垛次数即为取出目标钢卷的数量;在图 2(c)中,拿出目标 3 号钢卷,需要先移走第 2 层的 5 号叠压钢卷,在图 2(d)中拿出 2 号目标钢卷需要移走位于第 2 层的 4 号和 5 号叠压钢卷。图 2(c)和图 2(d)中倒垛次数为拿出目标钢卷和搬运叠压钢卷次数之和,并且图 2(d)图情况倒垛次数最多。

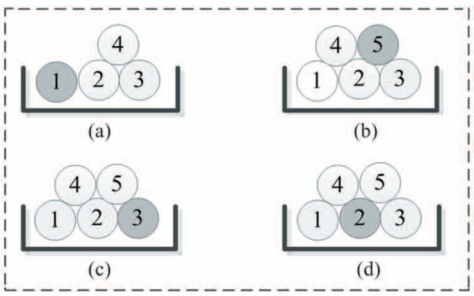

图 2　目标钢卷位置

叠压钢卷倒垛过程在同行中没有落位时需要落位在中转区域,其钢卷倒跺实例如图 3 所示。在初始状态下,灰色 2 号钢卷为目标钢卷,中转区域和目的区域没有钢卷;在作业状态下,天车把 4、5 号叠压钢卷放到中转区域,并取出目标钢卷到目的区域;在结束状态下,天车把 4 号钢卷放到源区域中 2 号钢卷位置,然后放回 5 号钢卷。为了避免二次倒跺,重定位钢卷不能放在后续订单任务所需钢卷上。

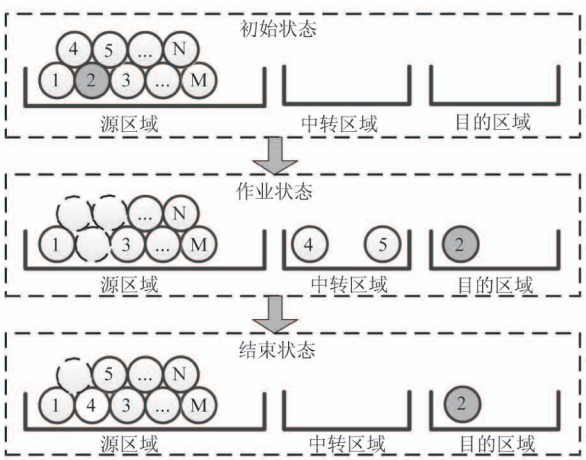

图 3　倒跺实例

1.2　问题建模

1.2.1　模型假设

(1)订单序列靠前的钢卷放在上层;

(2)钢卷的位置和优先级已知;

(3)在倒跺期间没有新钢卷入库;

(4)仓库内钢卷数量和种类满足订单需求,没有补货情况发生。

1.2.2　相关符号定义

本研究使用的相关符号约定如表1所示。

表1　本文符号约定

字符表示	字符说明
I	垛位上所有钢卷集合,i,$i^* \in I$
R	所有调度天车集合,$r \in R$
K	所有订单任务集合,$k \in K$
O	钢卷库出库任务集合,$o \in O$
S_k	钢卷订单序列,$S_k = (1,2,3\cdots,i)$
T	钢卷所在层位,t,t^*,$t^{**} \in T$
L	钢卷所在行位,$l \in L$
F	钢卷所在列位,$f \in F$
M	钢卷库存区号,$m \in M$
P_{mlft}	钢卷位置信息
ϕ_{ii^*}	钢卷 i 和 i^* 区号之差
η_{ii^*}	钢卷 i 与 i^* 是否位于同一区域,若是则为1,否则为0
τ_k	连续装载序列号是否位于同一区域,若是则为1,否则为0
YP_i	钢卷 i 是否可以直接取出,若是则为1,否则为0
$X_{(k,i)}$	钢卷 i 是否可以按照序列 k 出库
α_{ii^*}	钢卷 i 与 i^* 出库序列号之差
β_{ii^*}	表示钢卷 i 与 i^* 的出库顺序,若 i 在 i^* 之前则为1;否则为0
W_i	钢卷 i 实际重量
W_r	调度天车 r 最大吊起重量

1.2.3　钢卷倒垛模型

钢卷的出库任务根据客户订单生成,每个订单需要几个或者十几个钢卷。现在构造一个 $p \times q$ 的矩阵来表示出库任务,p 表示订单个数,q 表示每个订单所需要的钢卷数量。出库任务 $\boldsymbol{O}$ 如式(1)所示。

$$\boldsymbol{O} = \begin{bmatrix} (m_{11},l_{11},f_{11},t_{11}) & \cdots & (m_{1q},l_{1q},f_{1q},t_{1q}) \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ (m_{p1},l_{p1},f_{p1},t_{p1}) & \cdots & (m_{pq},l_{pq},f_{pq},t_{pq}) \end{bmatrix}_{(p\times q)} \tag{1}$$

例如一个包含3个订单每个订单分别需要5、4、6卷钢卷的出库任务,需要构建6行3列的矩阵,矩阵中行数表示每个订单所需钢卷,列数表示订单的数量。钢卷出库任务实例如式(2)所示。

$$\boldsymbol{O} = \begin{bmatrix} (1,5,7,3) & (2,8,5,1) & (1,6,2,1) \\ (2,5,2,1) & (1,3,2,1) & (1,8,4,2) \\ (1,2,4,1) & (1,7,5,2) & (1,4,2,1) \\ (1,7,2,2) & (2,4,6,2) & (1,5,4,2) \\ (1,2,3,1) & (0,0,0,0) & (1,8,5,7) \\ (0,0,0,0) & (0,0,0,0) & (1,6,3,2) \end{bmatrix} \tag{2}$$

约束条件为

$$O = \sum_{k\in K} S_k,\ S_k = (1,2,3,\cdots,i) \tag{3}$$

式(3)为出库任务,由订单序列组成。

$$\alpha_{ii^*} = \sum_{k\in K}\sum_{i\in I} X_{(k,i)} - \sum_{k^*\in K i^*\in I}\sum X_{(k^*,i^*)} \tag{4}$$

式(4)为出库序列之差,由同一订单两钢卷订单序列差求得。

$$\phi_{ii^*} = \sum_{k\in K}\sum_{i\in I} X_{(k,i)}\cdot P_{mlft} - \sum_{k^*\in K i^*\in I}\sum X_{(k^*,i^*)} \cdot P_{m^*l^*f^*t^*},\ \phi_{ii^*} \in [0,1] \tag{5}$$

式(5)为出库区号之差,由同一订单量钢卷库位序号差求得。

$$D_{(k,i)} = \sum_{k\in K}\sum_{i\in I} YP_i \cdot O \tag{6}$$

式(6)表示取出目标钢卷 i 倒垛次数,由钢卷是否需要倒垛与出库序列求得。

$$W_i < W_r,\ W_i \in (8,10) \tag{7}$$

式(7)表示天车吊运重量大于钢卷实际重量。

决策变量如下所示。

$$\beta_{ii^*} = \begin{cases} 1 & \alpha_{ii^*} > 0 \\ 0 & \alpha_{ii^*} < 0 \end{cases} \tag{8}$$

$$\eta_{ii^*} = \begin{cases} 1 & \phi_{ii^*} > 0 \\ 0 & \phi_{ii^*} < 0 \end{cases} \tag{9}$$

$$YP_i = \begin{cases} 1 & \begin{cases} t - t^* \geqslant 0 \\ t - t^{**} \geqslant 0 \end{cases} \\ 0 & \begin{cases} t - t^* < 0 \\ t - t^{**} < 0 \end{cases} \end{cases} \tag{10}$$

式(8)是在式(4)前提下钢卷出库顺序,式(9)是在式(5)前提下钢卷出库区位,式(10)表示目标

钢卷是否需要倒垛。综合以上约束条件得到最小化倒垛次数的目标函数如式(11)所示。

$$\min F = D_{(k,i)} \cdot \beta_{ii*} \cdot \eta_{ii*} \cdot \tau_k \tag{11}$$

2 构建蒙特卡洛钢卷搜索树

2.1 MCTS 算法简介

MCTS 算法又称随机抽样或统计试验方法,属于计算数学的一个分支,是以概率和统计理论方法为基础通过使用随机数来解决问题的一种计算方法[11]。MCTS 算法最初应用在围棋对决中并获得了巨大成功,在围棋对决中 MCTS 解决问题的思路为:在双方形成的对弈局面下,通过 MCTS 搜索后续可能的落子,收集根节点和子节点的各种状态,根据各节点的评估值形成目前局势下最佳的对局步骤和赢棋概率。

MCTS 以一种非对称的搜索空间拓扑结构增长,在执行过程中会更频繁地访问未被探索和价值大的节点,其探索过程主要包括选择、扩展、模拟和反向传播 4 个阶段。

2.2 改进的 MCTS 算法设计

在 MCTS 的基础上,通过改进 UCT 选择策略和增加剪枝策略来加快算法的收敛速度,同时将钢卷位置信息和给定约束条件插入到搜索方法中,设计了 MCRST,从而为订单匹配最佳目标钢卷。算法共包含选择阶段、扩展阶段、修剪阶段、钢卷倒垛模拟阶段和反向传播阶段 5 个部分。

2.2.1 选择阶段

在选择阶段,算法根据评估值从根节点到子节点进行选择。不同的选择方式导致不同的树节点评估方法,常见的选择策略包括树的上置信区间(UCT)、所有动作优先(all moves as first, AMAF)和快速动作值估计(rapid action value estimation, RAVE)。

UCT 策略是一个函数,其作用是在被访问节点中选择下一个要遍历的节点。UCT 策略如式(12)所示。

$$UCT_i = \overline{Q_i} + C\sqrt{\frac{\ln N}{n_i}} \tag{12}$$

式中 $\overline{Q_i}$ 表示节点 i 的平均奖励值,n_i 表示节点被访问次数,N 表示父节点被访问的总次数,C 是一个常数,数值越大就越偏向于广度搜索,数值越小就越偏向于深度搜索。置信上限 UCT 的作用就是寻找最佳的子节点。

AMAF 策略的核心思想是不论任何时刻选择节点,都会给当前节点一个恒定值。AMAF 策略如式(13)所示。

$$AMAF_i = \overline{Q_i} + \overline{Q'_i} + C\sqrt{\frac{\ln(N+N')}{n_i + n'_i}} \tag{13}$$

UCT 策略返回结果相对理想,但是收敛速度相对较慢; AMAF 策略收敛速度快,但是结果有偏差。因此本文结合以上两个策略改进了 RAVE 策略。

$$RAVE_i = \beta AMAF_i + (1-\beta)UCT_i \tag{14}$$

$$\beta = \sqrt{\frac{E}{3N+E}} \tag{15}$$

其中 E 是等效参数,表示 β 与 $(1-\beta)$ 相等时的模拟次数。

MCRST 算法在选择阶段需要从订单中选择一个目标钢卷作为父节点来生成决策树(如图 4)。

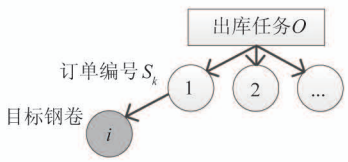

图 4 MCRST 算法选择阶段

2.2.2 扩展阶段

子节点的扩展策略是在选择策略之后,列举基于目标钢卷所有符合条件的钢卷形成钢卷簇;在扩展过程中优先选择未被扩展的子节点进行探索,如果没有需要扩展的则执行修剪阶段。(如图 5)。

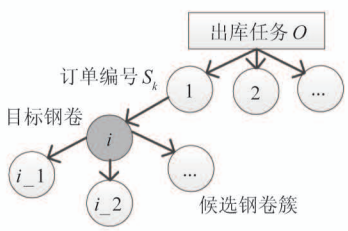

图 5 MCRST 算法扩展阶段

2.2.3 修剪阶段

在修剪阶段加入绝对剪枝策略,其含义为在节

点访问过程中,若有一个节点的访问次数达到了预计总访问次数的一半,则不再继续进行测评(如图6)。

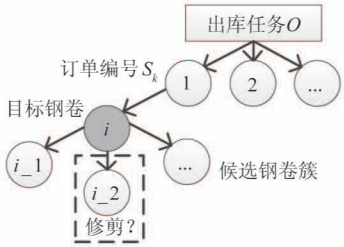

图6　MCRST 算法修剪阶段

在执行剪枝策略同时还需要检查模型中式(4)~(7)约束条件范围内所有候选钢卷。如果扩展节点不满足约束条件,则将其修剪并从扩展节点中随机选择一个节点,再次检查新节点,若通过检查,则进入仿真。

2.2.4　钢卷倒垛模拟阶段

模拟策略是根据式(11)的目标函数对模拟过程中的每个决策进行分级,在选择目标钢卷时要计算选择每个候选钢卷决策的概率,生成采样分布,然后以此为基础模拟来达到最终条件。模拟钢卷倒垛阶段,对修剪后的所有候选钢卷进行评估,在模拟过程中采用轮盘赌的方式,并通过总体评估来计算选择每种解决方案的可能(如图7)。

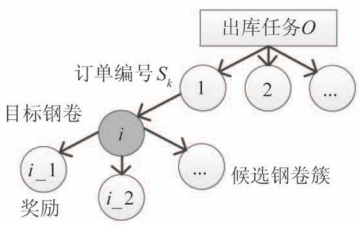

图7　钢卷倒垛模拟阶段

2.2.5　反馈阶段

模拟阶段结束后把每个子节点的模拟结果反馈到所有父节点上,直到最终反馈到根节点,把每条分支的所有子节点和父节点结果汇总(如图8)。

MCRST 算法流程如下所示。

步骤1　初始化仓库钢卷位置信息以及订单任务所需钢卷信息。

步骤2　根据初始化信息生成 MCTS 决策树。

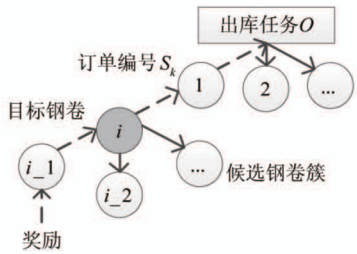

图8　MCRST 算法反馈阶段

步骤3　根据式(14)RAVE 策略从钢卷库选择钢卷。

步骤4　根据2.2.2节扩展策略对已选择订单钢卷进行扩展形成候选钢卷簇。

步骤5　根据2.2.3节剪枝策略和式(4)~(7)约束条件判断是否修剪节点,若是则修剪节点,否则执行步骤7。

步骤6　判断节点是否修剪完成,若修剪完成则执行步骤7,否则返回步骤5。

步骤7　采用轮盘赌形式对候选钢卷簇里所有钢卷根据式(11)目标函数进行评估。

步骤8　把钢卷模拟倒垛结果进行反馈。

步骤9　若结果满足迭代次数则找到目标钢卷并更新钢卷库信息,否则返回步骤2。

步骤10　检查是否满足出库任务,若满足执行出库任务,否则返回步骤2。

MCRST 算法流程图如图9所示。

3　仿真实验与分析

3.1　算例描述

实验数据的产生涉及以下4个因素。

(1) 库存规模。某公司钢卷库长240 m,宽54 m,分为1#和2#两个库存,每个库存可以存放60行20列的钢卷(实验中用行×列表示钢卷库规模)。

(2) 堆放高度。考虑到钢卷体积大、重量大以及在倒跺过程中容易产生磕碰等情况,钢卷堆放高度最高设定为2层。

(3) 钢卷种类。本次仿真设置的钢卷种类根据钢号和钢卷直径不同多达20余种。

(4) 天车数量。采用两架天车进行作业,一架天车负责倒垛作业,一架天车负责出库作业。

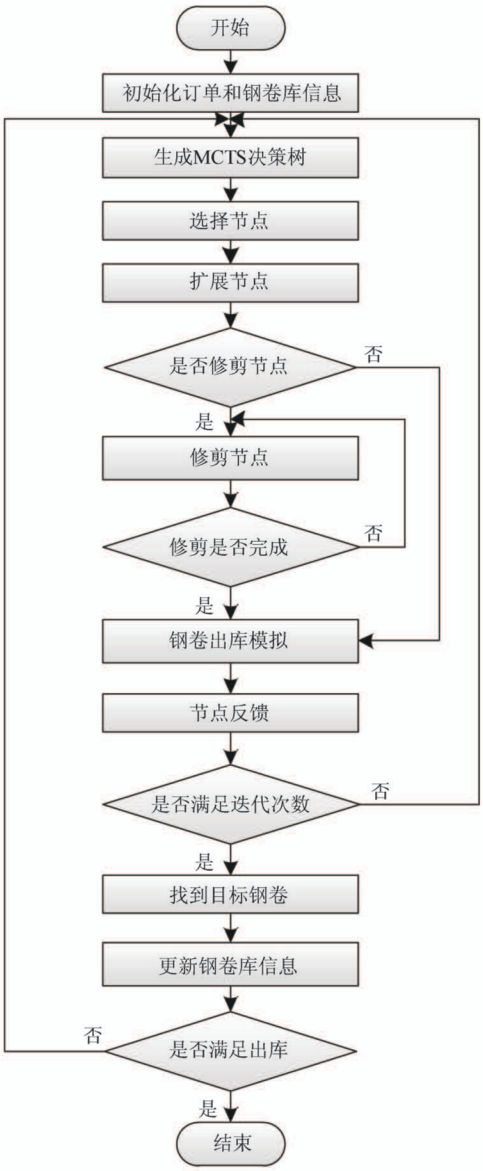

图 9　MCRST 算法流程图

3.2　参数设置

在实验过程中，将改进 UCT 策略的 MCRST 算法（RAVE-MCRST）与标准 UCT 策略的 MCRST 算法（UCT-MCRST）和标准粒子群算法（particle swarm optimization，PSO）进行比较。其中，式（12）和式（14）中 C 取值为 $\sqrt{2}/2$，式（14）中 E 取值为 1200，PSO 算法中学习因子 $c_1 = c_2 = 2$，惯性因子 $\omega = 1$，粒子数量为 20，粒子最大速度为 1500。

3.3　结果分析与比较

在比较实验中，设定每次以 8 个钢卷的出库任务进行实验，仿真结果取 10 次结果的平均值。钢卷出库任务由不同订单组成，每个订单中包含所需钢卷的钢号、直径和数量等信息，订单号的顺序表示处理订单任务的优先级，订单号越小优先级越高。钢卷出库任务如表 2 所示。

表 2　钢卷出库任务

规格 / 订单	Q195			Q235		
	900	1100	1250	900	1100	1250
1	2	3	0	1	1	1
2	1	1	0	3	3	0
3	1	2	3	1	1	0
4	2	3	1	1	1	0
5	0	1	3	3	0	1
6	1	3	2	2	0	0
7	2	1	2	2	1	0
8	3	1	1	1	1	1

实验过程中选取同一订单任务 8 卷钢卷，通过设置不同的钢卷库规模来比较 RAVE-MCRST、UCT-MCRST 和 PSO 算法的搜索能力。

在不同钢卷库规模下同一订单不同算法的对比如图 10 ~ 图 13 所示。在规模为 45 × 10 时，3 种算法都能收敛到最优值，从收敛速度上可以看出 RAVE-MCRST 算法最快，UCT-MCRST 算法次之，PSO 算法最慢；随着钢卷库规模的增大，PSO 算法陷入局部最优值，RAVE-MCRST 算法和 UCT-MCRST 算法都能收敛到最优值，在收敛速度上 RAVE-MCRST 算法要快于 UCT-MCRST 算法。从图 12 和图 13 中可以得出结论，MCRST 算法在大规模问题搜索和求解中要优于 PSO 算法，同时改进 UCT 策略的 RAVE-MCRST 算法在保证算法收敛的同时也克服了标准 UCT 策略在收敛速度慢的缺点。

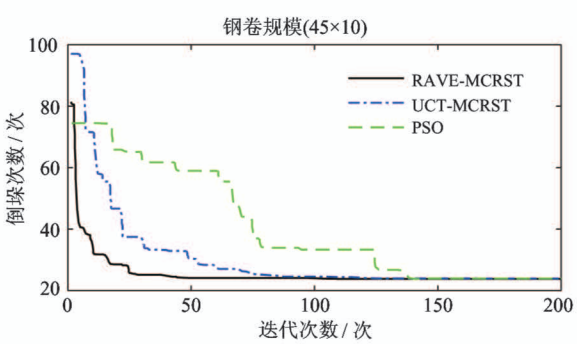

图 10　规模 45 × 10 倒垛次数对比

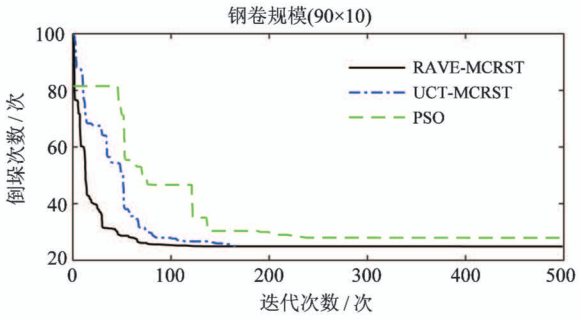

图 11　规模 90 ×10 倒垛次数对比

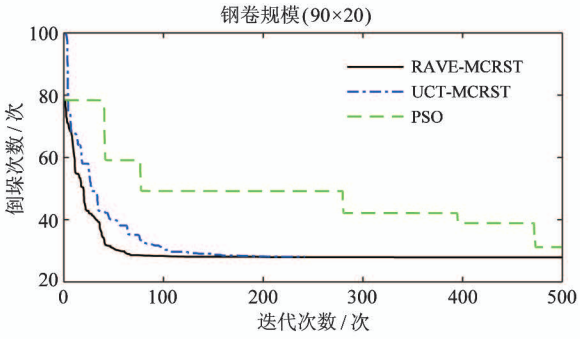

图 12　规模 90 ×20 倒垛次数对比

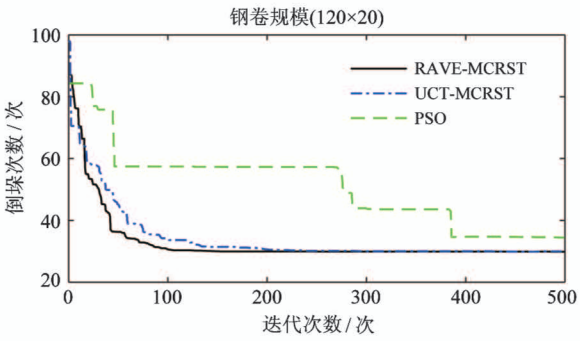

图 13　规模 120 ×20 倒垛次数对比

3 种不同算法在总倒垛次数上最优值、平均值和标准差的比较结果由表 3 ~ 表 6 给出。在钢卷库规模为 45 ×10 时,3 种算法均值相同和标准值相差不大,但是标准差上 RAVE-MCRST 算法最小,说明数据偏离均值的程度最小;在钢卷库规模为 90 ×10 时,3 种算法均值相差不大,PSO 在标准差上与其他两种算法相比要大一些;随着钢卷库规模进一步加大,PSO 算法在均值和标准差上较另外两种算法差距增大,PSO 算法的倒垛次数与均值相比,上下幅度偏离较大。从表 5 和表 6 分析可得在大规模问题上,RAVE-MCRST 算法较 PSO 算法在平均倒垛次数上有 7.2% 和 9.2% 的提升。

表 3　钢卷库规模 45 ×10 算法比较结果

算法 \ 统计变量	最优值/次	平均值/次	标准差/次
RAVE-MCRST	22	23.1	0.79
UCT-MCRST	22	23.1	0.88
PSO	21	23.1	0.96

表 4　钢卷库规模 90 ×10 算法比较结果

算法 \ 统计变量	最优值/次	平均值/次	标准差/次
RAVE-MCRST	25	25.8	0.78
UCT-MCRST	25	25.8	0.90
PSO	26	26.5	1.02

表 5　钢卷库规模 90 ×20 算法比较结果

算法 \ 统计变量	最优值/次	平均值/次	标准差/次
RAVE-MCRST	27	28.3	0.84
UCT-MCRST	27	28.5	0.98
PSO	28	30.5	1.92

表 6　钢卷库规模 120 ×20 算法比较结果

算法 \ 统计变量	最优值/次	平均值/次	标准差/次
RAVE-MCRST	29	30.5	0.88
UCT-MCRST	30	30.8	1.05
PSO	31	33.6	3.35

目标钢卷次序和订单钢卷数量 2 个因素对 RAVE-MCRST 算法的验证结果如表 7 和表 8 所示。表 7 表示的是包含 8 个钢卷的订单任务,钢卷库规模为 60 ×20,采用正序、逆序和混序 3 种顺序取出目标钢卷。从表中可以看出,正序订单在总倒垛次数和总作业时间上是最少的,说明订单任务中钢卷的取出顺序会影响整个订单的完成效率,但是 3 种不同顺序下在倒垛次数和作业时间上相差不大。

表 7　钢卷不同次序与评价指标

评价指标 \ 订单次序	正序订单	逆序订单	混序订单
总倒垛次数/次	23.5	23.8	24.2
总作业时间/s	302.3	310.4	320.5

表 8 订单需求量与评价指标

评价指标 \ 钢卷数量	8	16	24
总倒垛次数/次	23.2	68.4	83.4
总作业时间/s	324.5	624.3	854.3

订单需求量对算法的影响由表 8 给出。仿真中钢卷库规模为 60×20,随着目标钢卷数量以倍数形式的增加,相应的总倒垛次数和总作业时间也相应增加,但是倒垛次数是以非线性缓慢形式增加,说明本算法在处理大规模库存和繁重的订单任务上同样适用。

4 结 论

本文将人工智能算法 MCTS 应用在钢卷库倒垛优化问题中,并据此设计了 MCRST 算法,在应用中一方面改进了 UCT 选择策略来加快算法收敛速度,另一方面引入绝对剪枝策略引导算法选择访问次数更高的节点;在搜索过程中采用轮盘赌形式进行模拟倒垛作业,确保了模拟过程随机移动,体现了算法的概率本质。在实验验证过程中,将 RAVE-MCRST 算法与 UCT-MCRST 算法和 PSO 算法进行比较,随着钢卷库规模的增大,寻找目标钢卷的倒垛次数减少,减轻了天车作业负荷和作业时间,提升了企业物流效率。本文算法主要针对钢卷库倒垛实际应用场景设计,也可扩展至企业离散化的出库发运任务,同时本算法基于概率模型,耗时较多,可以在以后的研究中开发更有趋向性的选择策略,提高算法运行速度。

参考文献

[1] 王新东, 闫永军. 智能制造助力钢铁行业技术进步[J]. 冶金自动化,2019,43(1):1-5

[2] 刘文仲. 关于中国钢铁工业智能制造的思考[J]. 冶金自动化, 2018,42(4):1-6

[3] 侯俊, 张志英. 船厂钢板堆场混合存储分配及出入库调度研究[J]. 哈尔滨工程大学学报, 2017,38(11):1786-1793

[4] Dotolia M, Epicocoa N, Falagariob M, et al. A train load planning optimization model for intermodal freight transport terminals: a case study[C]// IEEE International Conference on Systems, Manchester, UK, 2013:3597-3602

[5] 谢谢, 周莉, 郑勇跃. 混合亚启发式算法求解带有热量损失的单吊机调度[J]. 沈阳大学学报(自然科学版), 2019, 31(2):107-112

[6] Jakob M, Tone L. Comparison of lowest-slot and nearest-stack heuristics for storage assignment of steel bar sets [J]. *Logistics and Sustainable Transport*, 2018,9(2):37-45

[7] 刘丽萍, 李坤, 田慧欣. 钢铁企业出厂物流集成多目标优化问题[J]. 控制工程, 2019, 26(3):502-509

[8] Forster F, Bortfeldt A. A tree search procedure for the container relocation problem[J]. *Computers and Operations Research*, 2012, 39(2): 299-309

[9] Bortfeldt A, Forster F. A tree search procedure for the container pre-marshalling problem[J]. *European Journal of Operational Research*, 2012,217(3):531-540

[10] 董广静, 李铁克, 王柏琳. 考虑实时库存的轧制计划调整模型及算法[J]. 系统工程理论与实践, 2015, 35(5):1246-1255

[11] Browne C B, Powley E, Whitehouse D, et al. A survey of Monte Carlo Tree Search methods[J]. *IEEE Transactions on Computational Intelligence and AI in Games*, 2012, 4(1): 1-43

The optimization of smart cranes shuffle operations based on Monte Carlo tree search

Dong Yan, Kang Xuebin, Lei Zhaoming, Lu Yu

(School of Artificial Intelligence, Hebei University of Technology, Tianjin 300130)

Abstract

The optimization of smart crane shuffle operations is an important means to improve the utilization rate of the steel roll yard, and it is significance to promote the logistics efficiency of the steel roll warehouse to address this problem. This paper develops a mathematical model of crane operation load with the goal of minimizing the number of shuffle operations. In the process of solving the model, using the tree search method in Alpha Go-Zero for reference, this paper designs a Monte Carlo roll search tree (MCRST). In order to improve the convergence speed and accuracy of search tree results, the upper confidence bounds for trees (UCT) is changed to rapid action value estimation (RAVE). Meanwhile, the algorithm introduces absolute pruning strategy to avoid blind expansion of nodes. Through experiments of different scales, the improved algorithm is compared with the original tree search and particle swarm optimization (PSO), which proves the superiority of the algorithm in large-scale problems. At the same time, the algorithm also considers factors such as the order and volume of order steel coils to verify the algorithm applicability.

Key words: steel roll warehouse, shuffle operation, crane job, Monte Carlo search tree (MCTS), outbound task