Journal of System Simulation

Volume 33 | Issue 9 Article 18

9-17-2021

Order Sorting Optimization for Four-way Shuttle System Based on Improved Genetic Algorithm

Xinjie He
1. College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China; ;

Shaowu Zhou
1. College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China; ;

Hongqiang Zhang
1. College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China; ;

Lianghong Wu
1. College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China; ;

See next page for additional authors

Order Sorting Optimization for Four-way Shuttle System Based on Improved Genetic Algorithm

Abstract

Abstract: During the batch outbound operations of the four-way shuttle system, the different execution order of the system's outbound leads to the different interaction time between the four-way shuttle and the hoist will be different, which will affect the system's outbound operation time. *According to the operation process of batch outbound, with the order of batch order outflow as the variable and the system outflow time as the objective function, a system order sort optimization model i established. Based to the characteristics of the model, with the improved genetic algorithm, the optimal order of the system is obtained.* By changing the relevant parameters of the system, the optimization degree of the method is tested separately. Compared with the previous rorder sequence, the system's delivery efficiency is improved by at least 20%.

Keywords

four-way shuttle, outbound, sort, genetic algorithm

Authors

Xinjie He, Shaowu Zhou, Hongqiang Zhang, Lianghong Wu, and Zhou You

Recommended Citation

He Xinjie, Zhou Shaowu, Zhang Hongqiang, Wu Lianghong, Zhou You. Order Sorting Optimization for Four-way Shuttle System Based on Improved Genetic Algorithm[J]. Journal of System Simulation, 2021, 33(9): 2166-2179.

第 33 卷第 9 期 系统仿真学报© Vol. 33 No. 9
2021 年 9 月 Journal of System Simulation Sep. 2021

基于改进遗传算法的四向穿梭车系统订单排序优化

何昕杰[1]，周少武[1]，张红强[1]，吴亮红[1]，周游[1,2]
(1. 湖南科技大学 信息与电气工程学院，湖南 湘潭 411201；2. 湖南理工职业技术学院，湖南 湘潭 411206)

摘要：四向穿梭车系统进行批量出库作业时，系统出库订单的执行顺序不同，四向穿梭车与提升机之间的交互作业时长将会不同，进而影响系统的出库作业时间。*根据四向穿梭车系统批量出库的作业流程，以批量订单出库顺序为变量，系统出库时间为目标函数，建立了一种系统订单排序优化模型；针对这个模型的特点，采用改进的遗传算法进行求解，进而得出系统的最优出库顺序；*通过改变系统的相关参数分别测试该方法的优化程度，相较于优化之前的订单出库顺序，系统的出库效率至少提升 20%。

关键词：四向穿梭车；出库；排序；遗传算法
中图分类号：TP391 文献标志码：A 文章编号：1004-731X (2021) 09-2166-14
DOI: 10.16182/j.issn1004731x.joss.20-0454

Order Sorting Optimization for Four-way Shuttle System Based on Improved Genetic Algorithm

He Xinjie[1], *Zhou Shaowu*[1], *Zhang Hongqiang*[1], *Wu Lianghong*[1], *Zhou You*[1,2]
(1. College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China;
2. Hunan Vocational Institute of Technology, Xiangtan 411206, China)

Abstract: During the batch outbound operations of the four-way shuttle system, the different execution order of the system's outbound leads to the different interaction time between the four-way shuttle and the hoist will be different, which will affect the system's outbound operation time. *According to the operation process of batch outbound, with the order of batch order outflow as the variable and the system outflow time as the objective function, a system order sort optimization model i established. Based to the characteristics of the model, with the improved genetic algorithm, the optimal order of the system is obtained.* By changing the relevant parameters of the system, the optimization degree of the method is tested separately. Compared with the previous rorder sequence, the system's delivery efficiency is improved by at least 20%.

Keywords: four-way shuttle; outbound; sort; genetic algorithm

引言

在互联网和人工智能技术的大力推动下，电商企业的发展已经势不可挡。随着订单和数据的大规模增长，高效的物流自动化存储系统已然成为物流作业效率的必然选择。常见的物流自动化存储系统主要有：堆垛机系统[1]、子母穿梭车系统[2]和四向穿梭车系统[3]。

传统的堆垛机系统能够实现单巷道自动作业，但耗资成本高，系统的可拓展性不强。子母穿梭车系统将堆垛机系统的巷道作业和层位作业进行分解，采用跨层穿梭车和换层提升机分别实现巷道作业和跨层作业，大大提高了系统的可拓展性，然而巷道与巷道之间并不能实现作业交流，实现自动化

收稿日期：2020-07-08 修回日期：2020-08-31
基金项目：国防基础科研计划(JCKY2019403D006)；湖南省教育厅优秀青年项目(19B200)；湖南科技大学博士科研启动基金(E56126)
第一作者：何昕杰(1996-)，男，硕士生，研究方向为群体智能算法与群机器人系统。E-mail：hnxinjiehe@163.com

全方位存储依然具有一定的局限性。四向穿梭车系统通过增加横向轮以实现跨巷道作业，其配合提升机纵向作业，能够到达库存中任意位置，增强了系统的伸缩性和可拓展性，然而，系统的作业效率却也随之降低，因此，如何解决四向穿梭车系统调度优化和路径冲突的问题，是该系统应用的一个难点。

目前，国内外对自动化立体库调度优化问题的研究，主要以堆垛机系统和子母穿梭车系统为主，按其系统作业的方式分为：系统入库作业的储位优化问题[4-5]；系统出库作业的订单排序优化问题[6]；系统复合作业的路径优化问题[7-8]。本文研究的重点主要是系统出库批量订单排序优化问题。

王艳艳等[9]基于多层穿梭车系统并行拣货，提升机串行出库的作业特点建立了系统任务调度模型，采用改进的遗传算法求解任务调度模型。Zou 等[10]针对多层穿梭车系统在一定时间窗口内的批量出库问题进行建模，采用基于精英非支配排序的遗传算法对任务调度模型进行了优化求解。张晓清[11]针对双深位多层穿梭车系统出库订单排序优化问题，建立了考虑倒货策略的系统出库时间模型，并且采用改进的模拟退火算法进行了求解。Wang Y 等[12]针对跨层穿梭车系统批量出库问题，建立了一个非线性规划模型，并采用 Gurobi 优化器对该模型进行了精确求解，得出最优任务分配解。然而，此类研究应用的系统均为子母穿梭车系统，其在四向穿梭车系统上的调度优化并不可行。

Cao 等[13]针对四向穿梭车系统，采用半开环排队理论建立了四向穿梭车系统调度模型，并且针对订单出库问题，建立了改进的耦合度订单排序模型，然后采用了改进的耦合度订单排序启发式算法进行了求解，但该模型仅适用于小批量订单出库的排序问题。田彬等[14]针对单深位四向穿梭车系统跨层作业的方式模式，建立一种批量订单排序优化模型，采用改进耦合度订单排序贪婪算法进行求解。

目前，双深位四向穿梭车系统在处理大规模订单排序优化以及路径冲突的问题上还亟待解决。本文以批量订单出库顺序为自变量，系统出库时间为目标函数，建立了一种系统订单排序优化模型。经模型分析可知影响系统批量出库作业时间的因素主要为穿梭车与提升机的等待时长和倒货顺序。采用改进的遗传算法进行求解，进而得出系统的最优出库顺序。该方法的提出能够很好地满足系统调度性能，进一步提升系统的作业效率。

1 系统订单排序优化模型

本节首先对四向穿梭车系统的布局和系统建模进行简要描述，然后基于四向穿梭车系统订单批量出库的流程，以出库订单顺序为自变量，系统出库时间为目标函数，建立了系统批量出库的订单排序优化模型。

1.1 系统说明与建模假设

双深位四向穿梭车系统[15]主要由 2 个维度构成：分别为硬件部分和软件部分。该系统的硬件部分主要由双深位立体货架、四向穿梭车、高速料箱提升机、塑料周转箱、输送线等设备组成。其软件部分主要由订单管理系统(Warehouse Management System, WMS)，仓储调度系统(Warehouse Control System, WCS) 和仓储执行系统 (Warehouse Execution System, WES)等软件组成，具体的系统简要布局说明如图 1 所示。

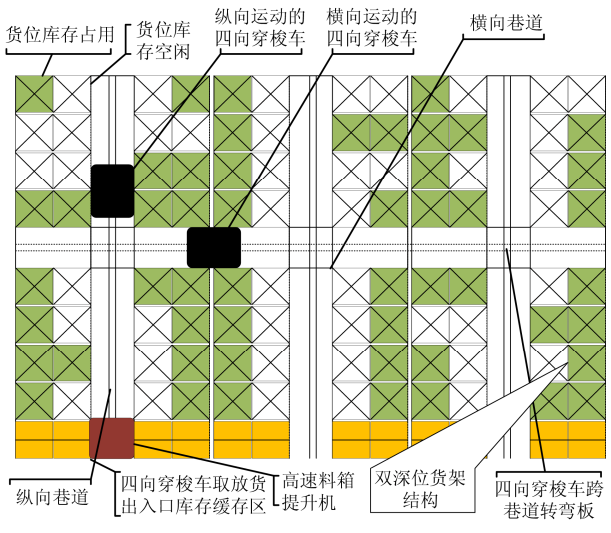


图 1 四向穿梭车系统布局图
Fig. 1 Four-way shuttle system layout

DOI: 10.16182/j.issn1004731x.joss.20-0454

第 33 卷第 9 期 系统仿真学报 Vol. 33 No. 9
2021 年 9 月 Journal of System Simulation Sep. 2021

1.1.1 双深位立体货架

一般立体货架的布局是针对客户的场地及其仓储数据来设计的，一旦货架规划设计完成，其参数基本保持不变。本文设定货架的层数为 L，巷道数为 N，列数为 M，货位为双深度货位，即第 N 个巷道所对应的第 M 列的存储空间为 4。由于该货架为四向穿梭车系统，设定其转弯板的位置位于第 20 列与第 21 列之间，由此可知，整个货架的存储空间数量 N_{num} 大小为

$$N_{\text{num}} = 4 \cdot L \cdot N \cdot M \tag{1}$$

设定货位的列数为 q，其货位列数与巷道数之间的关系为

$$q = 4N \tag{2}$$

此时库存信息在三维货架中可由空间坐标表示为：$\boldsymbol{X}_{\text{sku}}=(L, q, M)$。由此，整个仓储空间中的货位状态可由随机数建模生成：

$$f\left(L_i, q_i, M_i\right) = round(rand), i = \{1, 2, \cdots, N_{\text{num}}\} \tag{3}$$

式中：*rand* 为 0~1 之间的均匀随机数；*round* 为四舍五入函数；$f(L_i, q_i, M_i)$的值域为 0 或 1。当该函数值域为 1 时，代表该货架第 i 个货位占用，当函数值域为 0 时，代表该货架第 i 个货位空闲。

由于本文仅研究双深位四向穿梭车系统的出库模型，而仓储系统的库存信息是由其入库作业中 WMS 系统的储位优化决定，现依据双深位系统货位优化原则，对仓储系统中的货位状态进行二次遍历建模。

双深位货架库存信息中两深度货位之间的关系建立，依据式(3)的货位状态进行遍历更新：

当 $q_i=4j+1$，$j=\{0, 1, \cdots, N-1\}$时，

$$f(L_i, q_i + 1, M_i) = \begin{cases} round(rand), f\left(L_i, q_i, M_i\right) = 1 \\ 0, f\left(L_i, q_i, M_i\right) = 0 \end{cases} \tag{4}$$

当 $q_i=4j+4$，$j=\{0, 1, \cdots, N-1\}$时，

$$f(L_i, q_i - 1, M_i) = \begin{cases} round(rand), f\left(L_i, q_i, M_i\right) = 1 \\ 0, f\left(L_i, q_i, M_i\right) = 0 \end{cases} \tag{5}$$

式中：k 为 0~N–1 之间的正整数。

1.1.2 四向穿梭车运动模型

与传统的双向穿梭车[16]相比，四向穿梭车在仓储中的作业路径能够在同层位内实现横向运动和纵向运动，到达同层位任一库存点。假设每层均配备一辆四向穿梭车，其四向穿梭车的数量为 F，设定四向穿梭车的横向最大速度为 v_{dx}，纵向最大速度为 v_{dy}，横向最大加速度为 a_{dx}，纵向最大加速度为 a_{dy}，此时四向穿梭车的运动时间模型[8]建立如下：

由于其横向运动性能与其纵向运动性能一致，故本模型只做纵向运动建模。假设四向穿梭车达到最大纵向运动速度时的最大位移为 X_{F} 此时小车运动时间 t_{d} 与其实际运动距离 s_{y} 之间的关系：

$$t_{\text{d}} = \begin{cases} (2 \cdot v_{\text{dy}}) / a_{\text{dy}} + (s - 2 \cdot X_{\text{L}} / v_{\text{dy}}), s_{\text{y}} > X_{\text{F}} \\ \sqrt{v_{\text{dy}} / a_{\text{dy}}}, \ s_{\text{y}} \leqslant X_{\text{F}} \end{cases} \tag{6}$$

式中：四向穿梭车达到最大纵向运动速度时的最大位移 X_{F} 小车的性能参数关系为

$$X_{\text{F}} = v_{\text{dy}}^2 / (2 \cdot a_{\text{dy}}) \tag{7}$$

因此，通过此模型，仅需确定该小车执行订单的库存信息，即可求得小车完成该订单所需的时间。

本文设定四向车的容量一次仅能取一个货位的货物，并且每层配备一辆四向车，四向车仅在同层位内进行跨巷道作业。每层中的四向车将作业订单送至与提升机交互的层位缓存区，然后由提升机将货位送至出站口。

1.1.3 高速料箱提升机运动模型

假设在 1 号巷道口设置一台高速料箱提升机，其负责与每层之间的四向穿梭车交互，实现订单周转箱的高度作业。设定高速料箱提升机的最大加速度为 a_{l}，最大运行速度为 v_{l}，其作业运动时间模型与式(6)一致，故不作叙述。

1.2 系统订单排序优化模型

假设批量订单出库的数量为 n，出库任务序列位置信息矩阵：

$$\boldsymbol{X}_{\text{sku}} = \begin{bmatrix} L_1 & q_1 & M_1 \\ L_2 & q_2 & M_2 \\ \vdots & \vdots & \vdots \\ L_n & q_n & M_n \end{bmatrix} \tag{8}$$

提升机执行订单的任务序列矩阵为

$$\boldsymbol{l}_{\text{sort}} = randperm(n) \tag{9}$$

式中：*randperm* 函数为随机生成的 n 个出库订单任务序列矩阵。$\boldsymbol{l}_{\text{sort}}$ 订单任务序列矩阵与任务序列位置信息矩阵 $\boldsymbol{X}_{\text{sku}}$ 的关系如式(10)所示，表示第 i 个出库订单的编号与其编号所对应的库存位置信息相关绑定。

$$\boldsymbol{X}_{\text{sku}}(i) = \boldsymbol{l}_{\text{sort}}(i),\ i = 1, 2, \cdots, n \tag{10}$$

实际双深位四向穿梭车系统批量出库调度流程如图 2 所示。

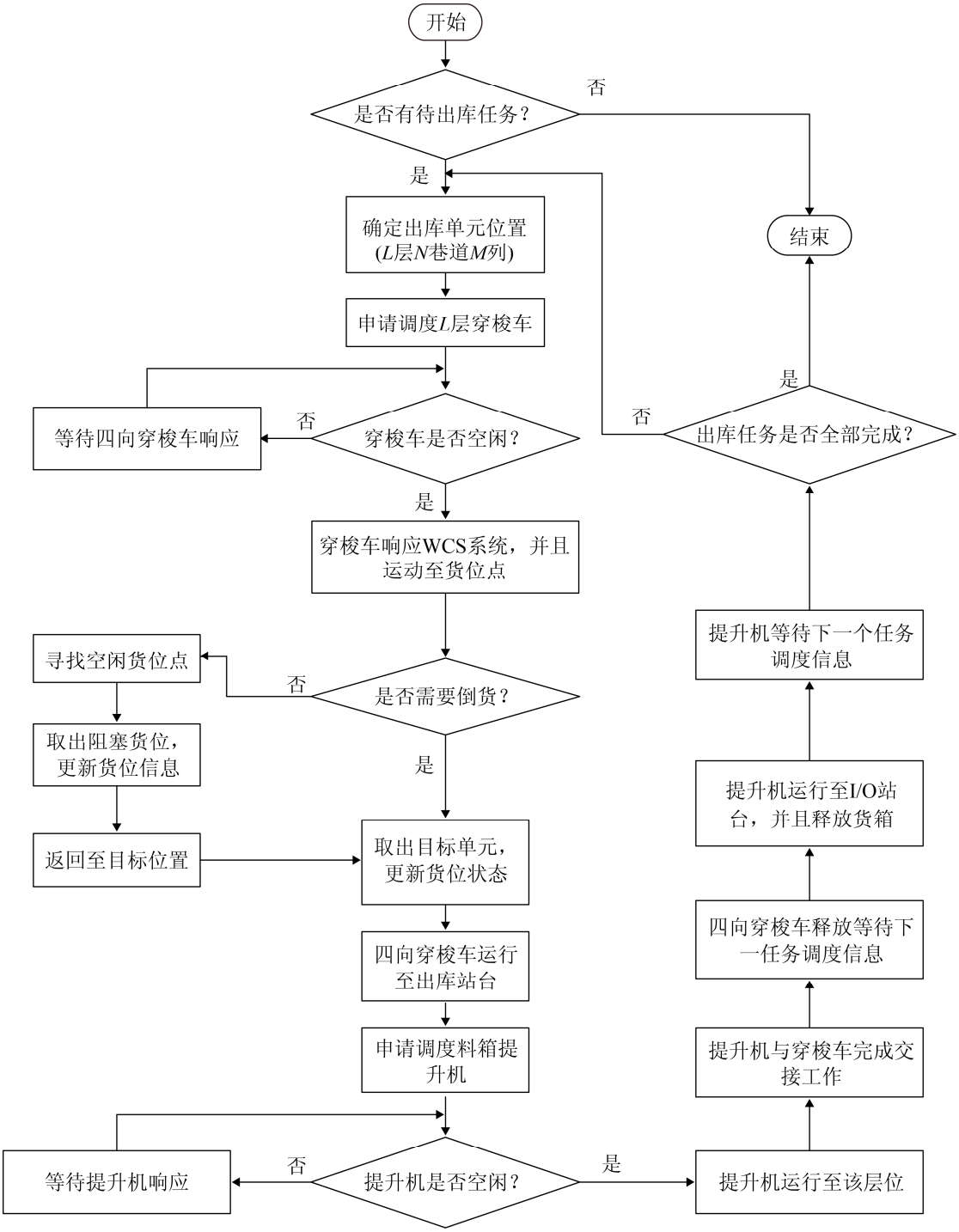


图 2 四向穿梭车系统批量出库流程图

Fig. 2 Flow chart of batch out of four-way shuttle system

https://dc-china-simulation.researchcommons.org/journal/vol33/iss9/18
DOI: 10.16182/j.issn1004731x.joss.20-0454

第 33 卷第 9 期 系统仿真学报 Vol. 33 No. 9
2021 年 9 月 Journal of System Simulation Sep. 2021

根据四向穿梭车批量出库流程图可知，系统的出库特点为：四向穿梭车并行作业，提升机串行作业，即：系统的出库时间等价于每层货架中四向穿梭车的运行时间的最大值，也可等价于提升机执行 n 个出库订单的串行总时长。该系统的订单出库模型可表示为

$$\begin{aligned}&\min\{T_{\text{out}}\}=\\&\min\{\max\{T^1_{\text{dolly}},T^2_{\text{dolly}},\cdots,T^p_{\text{dolly}},\cdots,T^F_{\text{dolly}}\}\}=\\&\min\{T^n_{\text{lift}}\}\end{aligned}\tag{11}$$

式中：T^p_{dolly} 为执行 n 个批量出库订单任务中第 p 层四向车作业的总时长；T^n_{lift} 为执行 n 个批量出库订单任务提升机的串行作业时长。现基于双深位四向穿梭车系统批量出库调度流程，分析提升机串行作业时长 T^n_{lift}。

设定提升机执行第一个任务的时间序列为 0 时刻，系统出库作业时间 T_{out} 可表示为

$$T_{\text{out}}=T^n_{\text{lift}}=\sum_{k=1}^{n}\Delta T^k_{\text{lift}},\ k=\{1,2,\cdots,n\}\tag{12}$$

式中：T^n_{lift} 为执行完 n 个任务订单的时刻；ΔT^k_{lift} 为提升机执行第 k 个订单的所需时长。提升机执行任第 k 个任务订单的关系：

$$\Delta T^k_{\text{lift}}=\Delta T^k_{\text{arrive}}+\Delta T^k_{\text{awaiting}}+\Delta T^k_{\text{delivey}}\tag{13}$$

式中：$\Delta T^k_{\text{arrive}}$ 为提升机执行第 k 个任务到达其目标层位的运行时间；$\Delta T^k_{\text{awaiting}}$ 为提升机到达该层位之后等待四向穿梭车取货的时间；$\Delta T^k_{\text{delivey}}$ 为提升机与四向穿梭车完成第 k 个任务交互后运送至出货站台的时间。

基于此提升机作业的时间序列，其递归模型为

$$T^k_{\text{lift}}=T^{k-1}_{\text{lift}}+(\Delta T^k_{\text{arrive}}+\Delta T^k_{\text{awaiting}}+\Delta T^k_{\text{delivey}})\tag{14}$$

由式(13)分析可知：当 T^{k-1}_{lift} 已知时，通过求解 $\Delta T^k_{\text{arrive}}$，$\Delta T^k_{\text{delivey}}$，$\Delta T^k_{\text{awaiting}}$ 的映射关系，即可迭代求得 n 个订单任务的执行时间序列。

1.2.1 提升机作业时长

现定义提升机执行第 k 个任务序列所对应的层位信息 L_k 之间的映射函数：

$$L_k=g(k,\boldsymbol{l}_{\text{sort}},\boldsymbol{X}_{\text{sku}})\tag{15}$$

设定每层货架的高度为 h，提升机运行至最大速度时的位移为 H，此时第 k 个提升机的任务执行时间为：

$$\Delta T^k_{\text{arrive}}=\begin{cases}\dfrac{2\cdot v_{\text{l}}}{a_{\text{l}}}+\dfrac{L_k\cdot h-2\cdot H}{v_{\text{l}}},\ L_k\cdot h\geqslant H\\[2ex]\sqrt{\dfrac{v_{\text{l}}}{a_{\text{l}}}},\ L_k\cdot h\geqslant H\end{cases}\tag{16}$$

式中：L_k 为第 k 个任务订单所对应的层位信息。

1.2.2 提升机等待时长

假设提升机开始执行第 k 个任务到达目标层位的时刻为 T^k_{arrive}，等候提升机交互完成时刻为 T^k_{awaiting}。该目标层位的四向穿梭车前一个任务执行订单任务号为 k–q，其等候提升机交互完成时刻为 $T^{k-q}_{\text{awaiting}}$，则第 k 个任务交互完成时刻 T^k_{awaiting} 与其第 k–1 个任务完成时刻 T^{k-1}_{delivey} 之间的关系：

$$T^k_{\text{awaiting}}=T^{k-1}_{\text{delivey}}+\Delta T^k_{\text{arrive}}+\Delta T^k_{\text{awaiting}}\tag{17}$$

设定四向穿梭车取第 k 个任务所花的时间为 T^k_{dolly}，此时执行第 k 个任务的提升机等待时长 $\Delta T^k_{\text{awaiting}}$ 之间的关系：

$$\Delta T^k_{\text{awaiting}}=\begin{cases}0,(T^k_{\text{arrive}}-T^{k-q}_{\text{awaiting}})>T^k_{\text{dolly}}\\T^k_{\text{dolly}}-(T^k_{\text{arrive}}-T^{k-q}_{\text{awaiting}}),(T^k_{\text{arrive}}-T^{k-q}_{\text{awaiting}})\leqslant T^k_{\text{dolly}}\end{cases}\tag{18}$$

将式(17)，(18)代入式(13)即可求得该系统出库时间的递归模型。现该模型中的未知参数仅剩小车的取货时间 T^k_{dolly} 时间建模。

1.2.3 四向穿梭车取放货作业时长

首先，建立第 k 个出库任务订单顺序的货位映射关系：

$$[L_k,q_k,M_k]=y(k,\boldsymbol{l}_{\text{sort}},\boldsymbol{X}_{\text{sku}})\tag{19}$$

其货位所在纵向巷道为

$$N_k=ceil(q_k/4)\tag{20}$$

考虑小车的取放货时间与其深度货位的关系，现建立其函数关系：

$$T^k_{\text{take}}=\begin{cases}1,\quad mod(q_k,4)>1\\1.5,mod(q_k,4)\leqslant 1\end{cases}\tag{21}$$

式中：T_{take}^{k} 为小车到达货位目标点之后的取货时间；*mod* 为取余函数，并且在该模型中，设定取货时间和卸货时间一致。

给定倒货决策变量函数定义：

$$G(f(L_k,q_k,M_k),f(L_k,q_k\pm1,M_k))=\begin{cases}1, f(L_k,q_k,M_k)=1\ \&\ f(L_k,q_k\pm1,M_k)=1\\0,\text{其他}\end{cases} \tag{22}$$

为了后续计算小车的时间模型的描述方便，对式(16)取函数定义：

$$T^{k}=\mathrm{J}\left(v_{\mathrm{l}},H,a_{l},l_{\mathrm{k}}\cdot h\right) \tag{23}$$

(1) 当 $G(f(L_k,q_k,M_k),f(L_k,q_k\pm1,M_k))=0$ 并且 N_k=1 时，设定货架的宽度为 c：

$$T_{\text{dolly}}^{k}=2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},M_k\cdot c)+2T_{\text{take}}^{k} \tag{24}$$

(2) 当 $G(f(L_k,q_k,M_k),f(L_k,q_k\pm1,M_k))=0$ 并且 $N_k\neq1$ 时，其横向距离运行至最大速度的距离值为 s_x，实际运动横向距离值为：s_n=5(N_k–1)c，设定转弯时间为 $T_{\text{turn}}^{k}=2$，则小车的取放货时间为

$$\begin{aligned}T_{\text{dolly}}^{k}=&2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},20c)+\\&2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},abs(N_k-20))+\\&2\cdot\mathrm{J}(v_{\text{dx}},s_{\text{x}},a_{\text{dx}},s_n)+2\cdot T_{\text{turn}}^{k}+2\cdot T_{\text{take}}^{k}\end{aligned} \tag{25}$$

(3) 当 $G(f(L_k,q_k,M_k),f(L_k,q_k\pm1,M_k))\neq0$ 并且 N_k=1 时，即小车取货的过程中需要考虑其倒货时间。本文采用的倒货策略为近邻倒货策略，其小车的运行时间为

$$T_{\text{dolly}}^{k}=T_{\text{1dolly}}^{k}+T_{\text{2repo}}^{k} \tag{26}$$

式中：T_{1dolly}^{k} 为四向穿梭车的取货时间，由式(24)可求得，故不作叙述。T_{2repo}^{k} 为四向穿梭车的倒货时间。

寻找其近邻倒货库位为：

$$index=\min\left[find(f(L_k,q_k,:)==0)\right] \tag{27}$$

四向穿梭车倒货运行距离 s_{repo} 为

$$s_{\text{repo}}=abs\left(index-M_k\right)\cdot c \tag{28}$$

由此可知，四向穿梭车执行第 k 个任务订单的倒货时间为

$$T_{\text{2repo}}^{k}=2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},s_{\text{repo}})+2\cdot t_{\text{take}}^{k} \tag{29}$$

将式(29)，(24)代入式(26)即可求得小车的执行第 k 个任务的运行时间。

(4) 当 $G(f(L_k,q_k,M_k),f(L_k,q_k\pm1,M_k))\neq0$ 并且 $N_k\neq1$ 时，其小车执行第 k 个订单的实际运行时间为

$$\begin{aligned}T_{\text{dolly}}^{k}=&2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},20c)+\\&2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},abs(N_k-20))+\\&2\cdot\mathrm{J}(v_{\text{dx}},s_{\text{x}},a_{\text{dx}},s_n)+2\cdot T_{\text{turn}}^{k}+\\&4\cdot T_{\text{take}}^{k}+2\cdot\mathrm{J}(v_{\text{dy}},s_{\text{y}},a_{\text{dy}},s_{\text{repo}})\end{aligned} \tag{30}$$

综上所述，由于同层位提升机的上升运行时间和下降运行时间一致，其执行第 n 个出库任务序列的系统出库时间递归模型可表述为

$$\begin{aligned}T_{\text{lift}}^{n}=&T_{\text{lift}}^{n-1}+(2\cdot\Delta T_{\text{arrive}}^{k}+\Delta T_{\text{awaiting}}^{k})=\\&T_{\text{lift}}^{1}+\sum_{k=2}^{n}(2\cdot\Delta T_{\text{arrive}}^{k}+\Delta T_{\text{awaiting}}^{k})=\\&2\cdot\Delta T_{\text{arrive}}^{1}+N(\Delta T_{\text{arrive}}^{1},\ T_{\text{dolly}}^{1})+\\&\sum_{k=2}^{n}(2\cdot\Delta T_{\text{arrive}}^{k}+\Delta T_{\text{awaiting}}^{k})\end{aligned} \tag{31}$$

其中决策变量 $N(\Delta T_{\text{arrive}}^{1},\ T_{\text{dolly}}^{1})$ 的定义为

$$N(\Delta T_{\text{arrive}}^{1},\ T_{\text{dolly}}^{1})=\begin{cases}0, \Delta T_{\text{arrive}}^{1}>T_{\text{dolly}}^{1}\\T_{\text{dolly}}^{1}-\Delta T_{\text{arrive}}^{1}, \Delta T_{\text{arrive}}^{1}\leqslant T_{\text{dolly}}^{1}\end{cases} \tag{32}$$

基于目标函数的建模推导，即可求得系统批量出库时间。

2 改进的遗传算法

根据系统订单排序优化模型可知，当系统批量出库订单 n 值过大时，显然，该问题是一个典型的 NP 问题，解决此类问题的常用方法有：①数学规划法，如枚举法[17]，Gurobi 优化器[18]等；②启发式算法，比如遗传算法[19]，模拟退火算法[20]和蚁群算法[21]等。

针对系统出库订单序列优化问题，采用传统的数学规划方法虽然能够准确地计算出其最优任务序列，但是其计算时间过长，计算量过大，并不适用于该系统的实际调度过程，因此，本节采用一种改进的遗传算法来求解 SOSOM(System Order Sort Optimization Model)模型优化问题。

https://dc-china-simulation.researchcommons.org/journal/vol33/iss9/18
DOI: 10.16182/j.issn1004731x.joss.20-0454

2.1 种群初始化和选择算子

设定遗传算法中染色体的数量为 P，其染色体基因初始化序列由式(9)表示，即对应为提升机随机初始任务序列，其适应度函数由式(31)所示，即 SOSOM 出库作业任务时间表示。选择算子通常和其适应度函数有关，由于本文所求的优化目标为最小化优化问题，故其适应度函数设置为

$$fitness_i(\boldsymbol{l}_{\text{sort}},\ p_i) = T - T_i^{n}{}_{\text{lift}} \tag{33}$$

式中：T 为给定得常数值，一般设为 $T > T_i^{n}{}_{\text{lift}}$；$fitness_i(\boldsymbol{l}_{\text{sort}}, p_i)$ 为第 i 个染色体对应的适应度值。遗传算法中常见的选择操作有轮盘赌、锦标赛和排序等选择算子，经过相关测试，本节选用无放回式随机余数选择算子，其具体的选择过程见文献[22]。

2.2 交叉算子

由于上节中给定的染色体编码方式为整数编码，并且交叉之后基因编号不能重复，此问题可等价于旅行商问题(Traveling SaleMan Problem, TSP)问题的求解，但 TSP 问题中城市之间的距离为定值，而该订单序列之间的作业时长并不恒定，如何寻找一种有效的交叉操作去求解该订单序列模型，成为了该算法设计的难点。本节针对部分匹配交叉算子(Partially Mapped Crossover Operator PMX[23] 计算度复杂，并且收敛速率慢的缺陷，设计了一种改进的部分匹配交叉算子(Improved Partially Mapped Crossover Operator, IPMX)，成功将 SOSOM 模型求解，并且其求解效果优于 PMX 算子。

假设两优秀父代的染色体基因序列为

$$\begin{cases} p_1 = (7,1,5,11,4,6,2,10,12,8,3,9) \\ p_2 = (2,11,4,6,12,10,7,3,5,9,8,1) \end{cases} \tag{34}$$

设其交叉初始位置和结束位置为第 4 个基因和第 9 个基因，其 IPMX 交叉算子设计如下。

2.2.1 交换基因序列，建立基因交换矩阵

由式(34)的父代信息进行基因交换：

$$\begin{cases} p_{12} = (7,1,5,6,12,10,7,3,5,8,3,9) \\ p_{21} = (2,11,4,11,4,6,2,10,12,9,8,1) \end{cases} \tag{35}$$

将其交换信息建立基因交换矩阵：

$$\boldsymbol{G}_{\text{e}} = \begin{bmatrix} 6 & 11 & 1 \\ 12 & 4 & 1 \\ 10 & 6 & 1 \\ 7 & 2 & 1 \\ 3 & 10 & 1 \\ 5 & 12 & 1 \end{bmatrix} \tag{36}$$

2.2.2 寻找变量基因，更新基因交换矩阵

首先将矩阵 $\boldsymbol{G}_{\text{e}}$ 进行映射关系：

$$\begin{cases} \boldsymbol{G}_{o_1}(1, \boldsymbol{G}_{\text{e}}(i,1)) = \boldsymbol{G}_{\text{e}}(i,2) \\ \boldsymbol{G}_{o_2}(1, \boldsymbol{G}_{\text{e}}(i,2)) = \boldsymbol{G}_{\text{e}}(i,1) \end{cases} \tag{37}$$

生成基因状态矩阵 $\boldsymbol{G}_{p_i}$ 和主导向量矩阵 $\boldsymbol{G}_{o_i}$。

主导向量矩阵：

$$\begin{cases} \boldsymbol{G}_{o_1} = [0\ 0\ 10\ 0\ 12\ 11\ 2\ 0\ 0\ 6\ 0\ 4] \\ \boldsymbol{G}_{o_2} = [0\ 7\ 0\ 12\ 0\ 10\ 0\ 0\ 0\ 3\ 6\ 5] \end{cases} \tag{38}$$

基因状态矩阵：

$$\begin{cases} \boldsymbol{G}_{p_1} = [0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1] \\ \boldsymbol{G}_{p_2} = [0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1] \end{cases} \tag{39}$$

然后将基因状态矩阵合并：

$$\boldsymbol{G}_p = \boldsymbol{G}_{p_1} + \boldsymbol{G}_{p_2} = [0\ 1\ 1\ 1\ 1\ 2\ 1\ 0\ 0\ 2\ 1\ 2] \tag{40}$$

最后，其变量基因求解：

$$\boldsymbol{G}_v = find(\boldsymbol{G}_p(1,:) > 1) = [6\ 10\ 12] \tag{41}$$

由式(40)的计算结果，更新基因交换矩阵：

$$\boldsymbol{G}_e = \begin{bmatrix} 6 & 11 & 0 \\ 12 & 4 & 0 \\ 10 & 6 & 0 \\ 7 & 2 & 1 \\ 3 & 11 & 1 \\ 5 & 4 & 1 \end{bmatrix} \tag{42}$$

根据基因交换矩阵，更新主导向量矩阵：

$$\boldsymbol{G}_{o_1} = [0\ 0\ 11\ 0\ 4\ 0\ 2\ 0\ 0\ 0\ 0\ 0] \tag{43}$$

2.2.3 生成新的基因序列

将主导向量矩阵 $\boldsymbol{G}_{o_1}$ 与式(35)中的矩阵关系进行重复映射，其余基因位置不变，即可产生子代 1 基因：

$$p_1' = (2,1,4,6,12,10,7,3,5,8,11,9) \tag{44}$$

现定义子代基因 1 与 2 之间的映射关系：

$$\begin{cases} F_e(p'_1(i)) = p_1(i), i = 1,2,\cdots,n \\ p'_2(i) = F_e(p_2(i)), i = 1,2,\cdots,n \end{cases} \tag{45}$$

由式(44)可知其子代 2 的基因序列：

$$p_2' = (7, 3, 5, 11, 4, 6, 2, 10, 12, 9, 8, 1) \quad (46)$$

2.2.4 重复迭代交叉

为了加快该算法求解过程中的收敛速率，现引入评价函数来对该交叉算子进行重复迭代更新。首先引入该算法的适应度函数作为子代评价函数，然后将生成的子代适应值与父代适应值进行比较，如果生成的子代优于其父代，则选择该子代染色体，停止内部迭代；如果生成的子代劣于其父代染色体，则将该子代与父代重组，进行重复交叉迭代，直到在内部迭代次数 T_p 内获得优于父代的染色体或者达到 T_p 内部迭代次数时仍然没有出现优于父代的子代染色体，则以一定随机概率去选择生成的子代基因。

2.3 变异算子

遗传算法中的第三大算子为变异算子，其作用是为了保证其搜索解空间的种群多样性，探索更多的可行解区域，从而避免陷入局部最优点。本文选用传统的遗传算法求解整数编码问题中的随机两点变异操作[21]，以保持种群交配的多样性。

2.4 算法流程设置

改进的遗传算法求解双深位四向穿梭车系统订单排序优化模型(Improved Adaption Genetic Algorithm- System Order Sort Optimization Model, IAGA- SOSOM)算法设置为：

step 1：初始化双深位四向穿梭车系统货架信息，订单信息，提升机运行的基本参数信息，四向穿梭车运行参数，随机初始化染色体任务序列信息，设置迭代时间 T；

step 2：式(31)，(32)求解种群规模的适应度函数，并通过式(33)转换其适应度函数值；

step 3：采用无放回式随机余数选择算子选择优秀父代染色体；

step 4：以一定交叉概率 P_c 采用 IPMX 策略进行交叉，产生优秀的子代染色体；

step 5：以一定变异概率 P_m 进行随机 2 点变异操作；

step 6：迭代更新，如果其迭代次数<T，则跳至 step 2 继续执行，如果迭代次数≥T，则迭代停止，并且返回其染色体基因编码组合，即最优提升机任务序列矩阵；

step 7：根据最优提升机任务序列矩阵和四向穿梭车之间的映射关系，分配各层位之间四向穿梭车最优任务订单执行顺序。

3 仿真测试

本仿真在 MATLAB2019a 软件下运行，主要进行了 3 个部分的数值仿真验证。首先将 IAGA-SOSOM 算法进行了仿真测试，得出了该系统的最优任务调度序列；其次，将本文提出的 IPMX 交叉算子与 PMX 交叉算子分别进行了 SOSOM 模型的求解，对其 2 种算法的性能进行了对比分析；最后，通过改变系统出库货位数量、仓库层位信息、仓库巷道信息等来分析该方法在此系统中的优化效率，验证其方法的有效性和适应性。

3.1 IAGA- SOSOM 系统出库订单排序优化仿真测试

首先，设置 IAGA- SOSOM 算法的参数如表 1 所示。然后设定货架的层位信息为 10，巷道数为 6，货位列数为 60，随机给定系统批量出库订单数量为 30，其出库任务顺序与其库存信息给定如表 2 所示。经 MATLAB 仿真计算，采用 IAGA- SOSOM 算法求解可知，其提升机订单排序优化执行序列如表 3 所示。由表 3 可知四向穿梭车订单调度执行顺序，其结果如表 4 所示。

设定优化前的出库订单任务顺序执行时间为 T_r，优化之后的出库订单任务顺序执行时间为 T_o，经 MATLAB 一次仿真计算，T_r=410 s，而优化之后的出库时间为 T_o=240 s，相较于优化之前，其系统的出库效率提升了 40%左右。设定优化之前的提升机等待总时长为 T_{ar}^{sum}，优化之后的提升机等待总时长为 T_{ao}^{sum}，经过 MATLAB 仿真数据可知 T_{ar}^{sum} =

214 s，而 T_{ao}^{sum}=44 s，提升机运动时长在优化前后均为 196 s，经分析可知：优化四向穿梭车批量出库时间关键在于如何减少提升机与四向穿梭车之间的等待时长。

表 1 IAGA-SOSOM 算法参数设置
Tab. 1 Algorithm parameter setting for IAGA-SOSOM

符号	意义	取值
N	巷道数	2~10
L	四向穿梭车数量/仓库层位数	5~30
M	库存列数	60
q	货架列数	8~40
v_{dx}/(m/s)	四向穿梭车横向最大速度	2
v_{dy}/(m/s)	四向穿梭车纵向最大速度	4
a_{dy}/(m/s^2)	四向穿梭车纵向最大加速度	2
a_l/(m/s^2)	提升机最大加速度	2
v_l/(m/s)	提升机最大速度	4
n	出库任务订单数量	20~100
p	种群规模大小	10
p_c	交叉概率	0.88
p_m	变异概率	0.12
T_p	内部迭代次数	10
T	迭代次数	1 000
L_a, L_b/m	货架长度和宽度	0.4
L_c/m	巷道宽度	0.4
h/m	货架单元层位高度	1

表 2 随机模拟初始出库任务顺序与库存坐标信息
Tab. 2 Random simulation of initial outbound task sequence and inventory coordinated information

出库 sku 订单顺序	出库坐标位置	出库 sku 订单顺序	出库坐标位置
1	9, 12, 32	16	2, 18, 27
2	6, 2, 11	17	3, 4, 14
3	1, 24, 43	18	3, 11, 48
4	2, 8, 40	19	10, 7, 40
5	8, 17, 30	20	9, 12, 43
6	3, 23, 28	21	5, 12, 30
7	3, 3, 62	22	1, 10, 5
8	5, 10, 30	23	6, 18, 36
9	6, 11, 47	24	5, 23, 44
10	4, 4, 53	25	3, 6, 60
11	7, 3, 30	26	8, 14, 60
12	6, 18, 48	27	2, 6, 14
13	8, 2, 38	28	5, 5, 3
14	2, 23, 34	29	2, 12, 1
15	1, 17, 23	30	3, 17, 17

表 3 最优出库任务顺序与订单坐标位置
Tab. 3 Optimal outbound task order and order coordinate position

出库 sku 订单顺序	出库坐标位置	出库 sku 订单顺序	出库坐标位置
8	9, 12, 32	22	2, 18, 27
20	6, 2, 11	10	3, 4, 14
12	1, 24, 43	5	3, 11, 48
26	2, 8, 40	11	10, 7, 40
18	8, 17, 30	9	9, 12, 43
3	3, 23, 28	16	5, 12, 30
4	3, 3, 62	13	1, 10, 5
7	5, 10, 30	6	6, 18, 36
23	6, 11, 47	24	5, 23, 44
17	4, 4, 53	29	3, 6, 60
19	7, 3, 30	30	8, 14, 60
21	6, 18, 48	1	2, 6, 14
14	8, 2, 38	15	5, 5, 3
2	2, 23, 34	28	2, 12, 1
25	1, 17, 23	27	3, 17, 17

表 4 四向穿梭车任务调度仿真结果
Tab. 4 Four-way shuttle task scheduling simulation result

四向穿梭车编号	小车取货顺序编号
1	3, 22, 15
2	4, 14, 16, 29, 27
3	18, 7, 17, 25, 6, 30
4	10
5	8, 21, 24, 28
6	12, 23, 2, 9
7	11
8	26, 5, 13
9	20, 1
10	19

3.2 遗传算法中的 PMX 与 IPMX 交叉算子性能对比分析

设定 IAGA-SOSOM 算法中的参数，即：货架的层位数为 10，巷道数为 10，货位列数为 60，批量出库订单量为 100 时，分别采用 IAGA-PMX 和 IAGA-IPMX 算法来求解 SOSOM 优化模型，分别测试 10 次，以对比这 2 种算法对该模型的优化性能，其测试数据如表 5 所示。

表 5 SOOTM 模型算法性能测试对比
Tab. 5 SOOM model algorithm performance test comparison

次数	交叉操作	优化前出库时间/s	优化后出库时间/s	优化率/%
1	PMX	1 580.657 385 698 510	1 004.739 535 337 340	36.4
	IPMX	1 580.657 385 698 510	978.109 896 619 146	38.1
2	PMX	1.568 500 706 200 000	945.937 459 155 804	39.7
	IPMX	1.568 500 706 200 000	893.441 664 751 801	43.0
3	PMX	1 628.449 099 784 050	1 025.751 515 651 510	37.0
	IPMX	1 628.449 099 784 050	945.996 058 598 403	41.9
4	PMX	1 671.260 754 824 020	950.285 073 283 691	43.1
	IPMX	1 671.260 754 824 020	939.903 766 479 787	43.8
5	PMX	1 797.147 072 898 940	963.395 411 220 104	46.4
	IPMX	1 797.147 072 898 940	943.475 185 706 556	47.5
6	PMX	1 634.336 214 214 290	1019.724 807 670 23	37.6
	IPMX	1 634.336 214 214 290	989.744 792 222 656	39.5
7	PMX	1 637.209 578 800 230	962.262 713 572 920	41.2
	IPMX	1 637.209 578 800 230	944.193 540 205 066	42.3
8	PMX	1 684.093 239 953 250	1033.052 485 052 28	38.7
	IPMX	1 684.093 239 953 250	972.930 027 092 293	42.3
9	PMX	1 552.374 473 742 870	967.055 084 534 341	37.7
	IPMX	1 552.374 473 742 870	895.613 005 510 491	42.3
10	PMX	1 556.810 952 286 240	927.406 666 553 485	40.4
	IPMX	1 556.810 952 286 240	911.913 251 077 280	41.4

由表 5 中的仿真测试数据可知：采用 IPMX 交叉操作求解 SOSOM 出库时间模型的优化效率均高于 PMX 交叉操作。鉴于该算法的随机性，下面给定其中某一组的 2 种算法求解过程的迭代优化曲线，如图 3 所示。

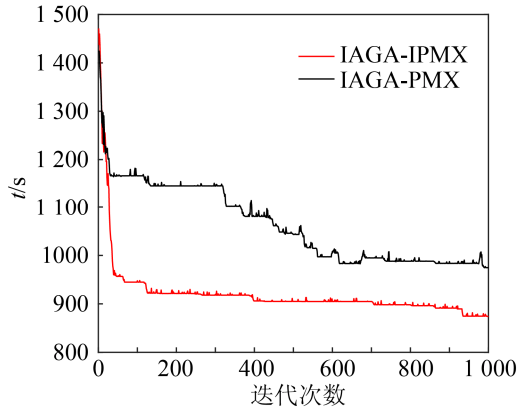


图 3 算法迭代性能对比图
Fig. 3 Algorithm iteration performance comparison chart

由图 3 可知，采用 IAGA-IPMX 算法不仅能够成功找出该整数规划模型的最优或次优解，而且其收敛速度和最优解搜索能力均优于 IAGA-PMX 算法。

3.3 IAGA-SOSOM 算法多维度仿真测试

本节通过改变四向穿梭车初始位置、批量出库订单的数量、仓库的层位信息以及仓库的巷道信息来多维度的测试该算法对 SOSOM 模型的优化结果，其多维度结果分布测试数据可验证系统的可行性，并且能够为系统规划设计提供一定的依据。

3.3.1 四向穿梭车初始位置对 IAGA-SOSOM 算法的优化测试

设定货架层位数 L 为 10，巷到数为 6，批量出库订单的数量为 100，各层位四向穿梭车的初始位

https://dc-china-simulation.researchcommons.org/journal/vol33/iss9/18
DOI: 10.16182/j.issn1004731x.joss.20-0454

第 33 卷第 9 期 系统仿真学报 Vol. 33 No. 9
2021 年 9 月 Journal of System Simulation Sep. 2021

置分别位于 1，2，3，4，5，6 巷道口，通过改变四向穿梭车的初始位置对 IAGA-SOSOM 算法进行测试，其测试结果如表 6 所示。

表 6　四向穿梭车初始位置变化对优化结果的影响数据
Tab. 6　Data on influence of the initial position change of the four-way shuttle on the optimization results

四向车初始位置	优化前的批量出库时间/s	优化后的批量出库时间/s	优化率/%
1	1 580.431 2	998.392 1	36.8
2	1 584.789 5	1 002.678 1	36.7
3	1 576.986 4	996.253 1	36.8
4	1 583.694 3	1 010.783 0	37.1
5	1 589.239 7	997.543 1	37.2
6	1 579.438 9	1 005.782 1	36.4

由表 6 的各项数据可知，改变四向穿梭车的初始位置对 IAGA-SOSOM 算法的优化结果影响较小，几乎可以忽略。

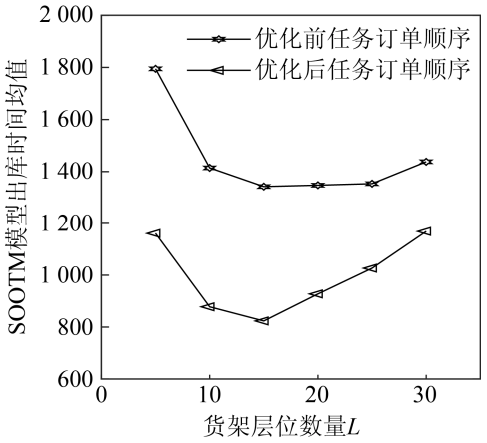
3.3.2 货架层位数 *L* 变化对 IAGA-SOSOM 算法的优化测试

货架层位数的改变，从而会影响提升机的作业时长，进而间接影响提升机与四向穿梭车的交互和等待时长，从而间接影响系统的出库时长，本节通过改变巷道层位数 *L* 来验证 IAGA-SOSOM 算法优化的适用性。鉴于该算法的随机性，取一定巷道层位数时，采用 IAGA-SOSOM 模型算法重复运行 10 次，以其均值来描述其优化结果。通过改变不同的货架层位数 *L*，其仿真数据统计的曲线图如图 4 所示。由图 4(a)和(b)可知:采用 IAGA-SOSOM 算法对系统的出库效率至少能提升 20%左右。

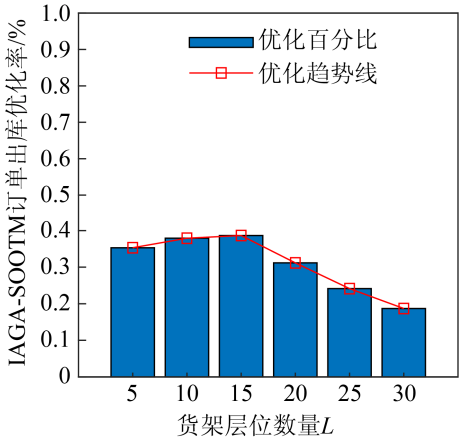
3.3.3 批量出库订单数量 *n* 变化对 IAGA-SOSOM 算法的优化测试

订单批量出库数量 *n* 的改变，会直接影响系统的批量出库时间，本节通过改变订单的数量 *n* 来测试 IAGA-SOSOM 算法的优化效率。本仿真的系统参数设置为：订单的出库数量为自变量，系统的出库时间为因变量，货架层位信息 *L* 为 10，巷道数为 6，列数为 60，其余参数同表 1 保持不变，鉴于该算法的随机性，取一定批量出库订单数量 *n* 时，采用 IAGA-SOSOM 模型算法重复运行 10 次，以其均值来描述其优化结果。通过改变不同的出库订单数量 *n*，其仿真数据统计的曲线图如图 5 所示。

由图 5(a)所示，无论是优化前的订单任务顺序还是优化后的任务订单顺序，其系统出库时间均会随出库订单数量的增加而递增，不过其采用 IAGA-SOSOM 优化之后的出库任务订单顺序的出库时间均优化前的系统出库时间。由图 5(b)所示，采用 IAGA-SOSOM 算法的系统出库时间优化率至少能提高 40%左右，而且趋于稳定。

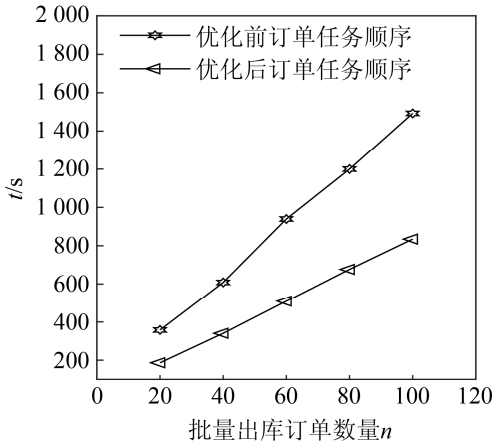


(a) 出库时间均值与货位层数 *L* 关系图

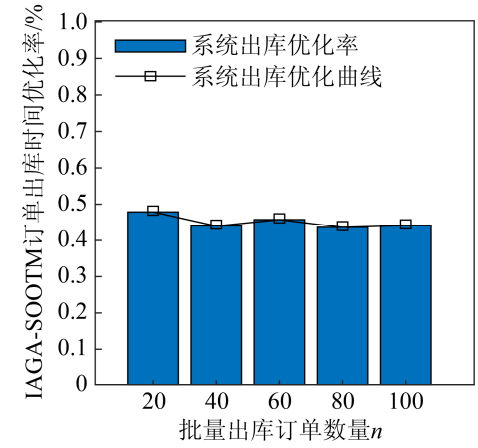


(b) 出库优化效率与货位层数 *L* 关系图

图 4　货架层位数 *L* 变化的出库优化曲线图
Fig. 4　Optimal curve of outbound storage with change in number of shelves *L*



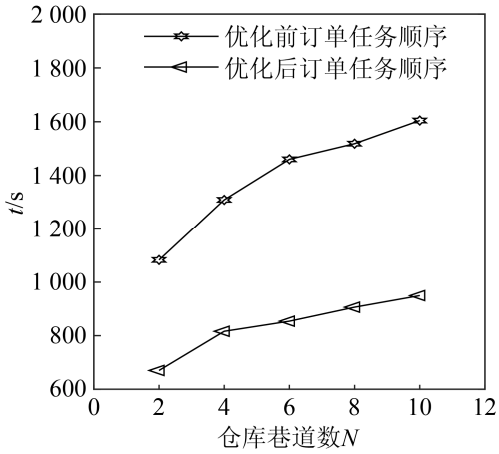
(a) 出库时间均值与批量出库订单数量 *n* 关系图



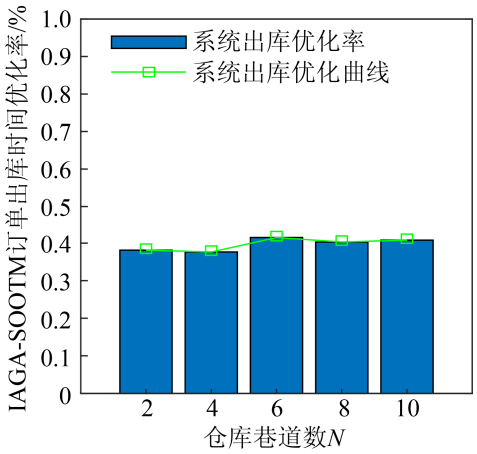
(b) 出库优化效率与批量出库订单数量 *n* 关系图

图 5 批量出库订单数量 *n* 变化的出库优化曲线图

Fig. 5 Outbound optimization curve graph of the change in the number of batch outbound orders *n*



(a) 出库时间均值与货架巷道数 *N* 关系图



(b) 出库优化效率与货架巷道数 *N* 关系图

图 6 货架巷道数 *N* 变化的出库优化曲线图

Fig. 6 Optimal curve of outbound warehouse with *N* number of shelves

3.3.4 货架巷道数 *N* 变化对 IAGA-SOSOM 算法的优化测试

货位巷道数 N 的改变，会影响某层位穿梭车的取货时间，进而间接影响提升机的等待时长，从而影响整个系统的批量出库订单时长。本仿真的系统参数设置为：仓库的巷道数 N 为自变量，系统的出库时间为因变量，货架层位信息 L 为 10，出库订单数量为 100，列数为 60，其余参数同表 1 保持不变，鉴于该算法的随机性，取一定仓库的巷道数 N 时，采用 IAGA-SOSOM 模型算法重复运行 10 次，以其均值来描述其优化结果。通过改变不同的仓库的巷道数 N 时，其仿真数据统计的曲线图如图 6 所示。

由图 6(a)所示，无论是优化前的订单任务顺序还是优化后的任务订单顺序，其系统出库时间均会随仓库巷道数 N 增加而递增，不过其采用 IAGA-SOSOM 优化之后的出库任务订单顺序的出库时间均优化前的系统出库时间。由图 6(b)所示，采用 IAGA-SOSOM 算法的系统出库时间优化率大致能稳定在 40%左右。

综合分析可知：本文提出的 IAGA-SOSOM 模型的订单排序优化算法在系统参数任意改变的情况下，相较于传统的随机任务出库顺序，其系统的出库效率至少能提高 20%以上，数值分析验证了该算法的有效性。

https://dc-china-simulation.researchcommons.org/journal/vol33/iss9/18
DOI: 10.16182/j.issn1004731x.joss.20-0454

第 33 卷第 9 期 系统仿真学报 Vol. 33 No. 9
2021 年 9 月 Journal of System Simulation Sep. 2021

4 结论

本文针对传统的双深位四向穿梭车系统批量出库效率低的问题，通过设计 IAGA-SOSOM 模型来优化该系统任务订单的出库顺序，从而克服了该系统的缺陷。

(1) 本文通过借用文献[11]中的双向穿梭车系统订单排序优化思想，以凯乐士四向穿梭车系统为例，建立了考虑倒货策略的双深位四向穿梭车系统出库时间模型(SOSOM)，采用递归的运算思想，顺利地求解了该系统的批量出库任务顺序的出库时间，其模型的复杂度和真实性能够准确逼近实际的四向穿梭车系统。

(2) 针对 SOSOM 出库时间模型的订单排序问题转化为非标准的整数规划问题，并且采用改进的遗传算法(IAGA)对该模型进行了求解，准确得出了其最优任务顺序序列，并且在改变系统的各种参数的条件下进行了该算法的测试和验证。

(3) 针对 SOSOM 整数规划问题的求解，本文将遗传算法中原有的 PMX 交叉算子在其复杂度和收敛速率上进行了改进，提出了 IPMX 算子。

本文所提出的 IAGA-SOSOM 算法能够成功应用于该类调度系统的批量出库的情况，并且其优化效率相较于传统的出库效率至少能提高 20%，然而，实际的双深位四向穿梭车调度系统中的作业形式为混合式作业，本文仅仅讨论了其出库任务的单一作业形式，在实际系统的调度中的适应性并不广范，因此，如何将入库系统中的储位优化算法和出库系统中的订单排序优化方法相结合，以实现实际场景的混合作业优化是接下来研究的重点。

参考文献：

[1] Chen M, Li X, Liu W, et al. Research on Multilevel Coordinational Flexible Scheduling Strategy of One-Track Dual-Stacker System Oriented Towards Response Time on Demand Side[C]//2018 37th Chinese Control Conference (CCC). Wuhan, China: Technical Committee on Control Theory, Chinese Association of Automation Systems Engineering Society of China, 2018.

[2] Lerher T. Travel Time Model for Double-deep Shuttle-based Storage and Retrieval Systems[J]. International Journal of Production Research (S1366-588X), 2016, 54(9): 2519-2540.

[3] 喻良宵, 罗利文, 朱晔. 四向穿梭式立体库吞吐量的计算机仿真研究[J]. 自动化技术与应用, 2016, 35(11): 30-33.
Yu Liangxiao, Luo Liwen, Zhu Ye. Computer Simulation Research on Throughput of Four-way Shuttle Stereoscopic Library[J]. Automation Technology and Application, 2016, 35(11): 30-33.

[4] 朱杰, 张文怡, 薛菲. 基于遗传模拟退火算法的立体仓库储位优化[J]. 计算机应用, 2020, 40(1): 284-291.
Zhu Jie, Zhang Wenyi, Xue Fei. Storage Optimization of Three-dimensional Warehouse Based on Genetic Simulated Annealing Algorithm[J]. Computer Application, 2020, 40(1): 284-291.

[5] Liu X. Research on the Storage Position Optimization for Stereoscopic Warehouse[J]. Value Engineering (S1006-4311), 2015, 55(12): 356-364.

[6] 蒋大奎, 李波, 谭佳音. 一类求解订单分配和排序问题的集成优化算法[J]. 控制与决策, 2013, 28(2): 217-222.
Jiang Dakui, Li Bo, Tan Jiayin. A Class of Integrated Optimization Algorithms for Order Allocation and Scheduling Problems[J]. Control and Design, 2013, 28(2): 217-222.

[7] 杨玮, 岳婷, 李国栋, 等. 子母式穿梭车仓储系统复合作业路径优化[J]. 计算机集成制造系统, 2018, 24(9): 221-228.
Yang Wei, Yue Ting, Li Guodong, et al. Optimization of the Compound Operation Path of the Storage System of the Cluster Shuttle System[J]. Computer Integrated Manufacturing System, 2018, 24(9): 221-228.

[8] 杨玮, 岳婷, 刘江, 等. 双载式多层穿梭车仓储系统复合作业路径优化[J]. 计算机集成制造系统, 2018, 24(12): 3178-3188.
Yang Wei, Yue Ting, Liu Jiang, et al. Optimization of Compound Operation Path of Double-loaded Multi-layer Shuttle Storage System[J]. Computer Integrated Manufacturing System, 2018, 24(12): 3178-3188.

[9] Wang Y Y, Mou S D, Wu Y H. Task Scheduling for Multi-tier Shuttle Warehousing Systems[J]. International Journal of Production Research (S1366-588X), 2015, 53(1/20): 5884-5895.

[10] Zou B P, Xu X H, Gong Y M. Modeling Parallel Movement of Lifts and Vehicles in Tier-captive Vehicle-based Warehousing Systems[J]. European Journal

of Operational Research (S1872-6860), 2016: 57(19): 789-796.

[11] 张晓清. 双深位多层穿梭车系统出库作业优化研究[D]. 济南: 山东大学, 2018.
Zhang Xiaoqing. Research on the Optimization of the Outbound Operation of the Double-deep Multi-layer Shuttle System[D]. Ji'nan: Shandong University, 2018.

[12] Wang Y, Huang K, Zhang N. A Model to Estimate Performance of Multi-tier Shuttle Warehousing System [C]//Chinese Automation Congress (CAS). Jinan, China: IEEE/ CAA, 2017: 4891-4895.

[13] Cao W J, Wu Y H, Xiao J W, et al. The Equipment Optimization of Automatic Picking System based on Preparatory Picking Buffer[C]// Chinese Automation Congress (CAS). Jinan, China: IEEE/ CAA, 2017: 6223-6227.

[14] 田彬, 吴颖颖, 吴耀华, 等. "四向车"拣选系统订单排序优化[J]. 机械工程学报, 2019, 55(18): 67-75.
Tian Bin, Wu Yingying, Wu Yaohua, et al. Order Sequencing Optimization for "Four-Way" Shuttle Based Order Picking System[J]. Journal of Mechanical Engineering, 2019, 55(18): 67-75.

[15] 曹伟洁. 四向穿梭车建模与优化[D]. 济南: 山东大学, 2018.
Cao Weijie. Modeling and Optimization of Four-way Shuttle Car[D]. Ji'nan: Shandong University, 2018.

[16] 张娜. 跨层穿梭车系统建模与优化[D]. 济南: 山东大学,, 2018.
Zhang Na. Modeling and Optimization of Cross-layer Shuttle System[D]. Ji'nan: Shandong University, 2018.

[17] Yu T, Liu M. A Memory Efficient Maximal Clique Enumeration Method for Sparse Graphs with a Parallel Implementation[J]. Parallel Computing (S0167-8191), 2019, 87(6): 46-59.

[18] Peng Y, Yang P. Integrated Optimization of Storage Location Assignment and Sequencing in Multi-shuttle Automated Storage/Retrieval Systems Under Modified Multi-command Cycle[C]//IEEE International Conference on Information & Automation. Chongqing China: IEEE, 2015.

[19] Zhang G H, Hu Y F, Sun J H, et al. An Improved Genetic Algorithm for the Flexible Job Shop Scheduling Problem with Multiple Time Constraints[J]. Swarm and Evolutionary Computation (S2210-6502), 2020, 54(5): 132-143.

[20] Zaretalab A, Hajipour V. An Extended Simulated Annealing Based on the Memory Structure to Solve Redundancy Allocation Problem[J]. Journal of Advanced Manufacturing Systems (S0219-6867), 2019, 18(4): 527-548.

[21] Yi N, Xu J, Yan L, et al. Task Optimization and Scheduling of Distributed Cyber-physical System Based on Improved Ant Colony Algorithm[J]. Future Generation Computer Systems (S0167-739X), 2020, 45(12): 567-578.

[22] Behrooz K. A Crossover Operator for Improving the Efficiency of Permutation-Based Genetic Algorithms[J]. Expert Systems with Applications (S0957-4174), 2020, 8(9): 145-155.

[23] Zakir H A. Adaptive Sequential Constructive Crossover Operator in a Genetic Algorithm for Solving the Traveling Salesman Problem[J]. International Journal of Advanced Computer Science and Applications (S2156-5570), 2020, 11(2): 1345-1354.

https://dc-china-simulation.researchcommons.org/journal/vol33/iss9/18
DOI: 10.16182/j.issn1004731x.joss.20-0454