

• 数字经济与社会 •

电商订单拣选与配送的联合调度优化

——基于考虑货箱卸货顺序约束的改进模拟退火算法

陈昌华^{1,2} 谯 衡² 胡晓峰³ 李茂波^{3*} 张 林³

1. 西华大学管理学院 四川成都 610039; 2. 西华大学国际经济与管理研究院 四川成都 610039; 3. 四川省烟草公司成都市公司物流中心 四川成都 610041

摘 要:在电商物流环境下货箱在送货车辆中的堆码方式会影响订单拣选与配送的整体服务效率,文章综合考虑了配送过程中的卸货顺序约束,分析了不同堆码方式对卸货效率、拣选顺序和分批方案的影响。以拣选时间和卸货时间最短为目标,构建了考虑货箱卸载顺序约束的订单拣选与配送联合调度模型(IOPDS-BUS),并设计了改进的模拟退火算法(ISA)以获得更高效的堆码方式、分批方案与拣选顺序。结果表明,ISA算法能够解决IOPDS-BUS问题,且优于传统算法;IOPDS-BUS可以提高物流中心的整体配送效率。同时,在物流配送实践中,很难同时优化拣选时间和卸货时间,但可以拣选时间为代价优化卸货时间,从而降低物流成本,提高整体配送效率。

关键词:电子商务;联合调度;订单拣选;货箱堆码;模拟退火算法

中图分类号:F274 文献标志码:A 文章编号:1672-8505(2023)06-0010-12

doi:10.12189/j.issn.1672-8505.2023.06.002

Joint Scheduling Optimization for E-commerce Order Picking and Delivery

—Based on an Improved Simulated Annealing Algorithm with Consideration of Cargo Box Unloading Sequence Constraints

CHEN Chang-hua^{1,2} QIAO Heng² HU Xiao-feng³ LI Mao-bo^{3*} ZHANG Lin³

1. School of Management, Xihua University, Chengdu, Sichuan, 610039, China;

2. Institute of International Economics and Management, Xihua University, Chengdu, Sichuan, 610039, China;

3. Chengdu Company Logistics Center, Sichuan Province Tobacco Company, Chengdu, Sichuan, 610041, China

Abstract: In e-commerce logistics, the stacking method of cargo boxes in delivery vehicles impacts the overall efficiency of order picking and delivery. This paper considers the constraints of the unloading sequence during the delivery process and analyzes how different stacking methods affect unloading efficiency, picking order, and batching strategies. Aiming to minimize the time spent on picking and unloading, an Integrated Order Picking and Delivery Scheduling model (IOPDS-BUS) that takes into account cargo box unloading sequence constraints has been developed. Furthermore, an Improved Simulated Annealing algorithm (ISA) has been proposed

收稿日期:2023-07-25

基金项目:国家自然科学基金青年基金资助项目(项目编号:62203367);四川省科技计划项目(项目编号:2022YFS0515)。

第一作者:陈昌华,男,副教授,博士,主要从事物流与供应链管理、技术经济及管理研究。

*通信作者:李茂波,男,高级工程师,主要从事物流管理、设备管理研究。

引用格式:陈昌华,谯衡,胡晓峰,等.电商订单拣选与配送的联合调度优化——基于考虑货箱卸货顺序约束的改进模拟退火算法[J].西华大学学报(哲学社会科学版),2023,42(6):10-21.

to find more efficient stacking methods, batching strategies, and picking sequences. The results demonstrate that the ISA algorithm can effectively address the IOPDS-BUS problem and outperforms traditional algorithms. Moreover, the IOPDS-BUS model can enhance the overall efficiency of logistics centers. However, in practical logistics operations, optimizing picking and unloading times simultaneously is challenging, but it is feasible to improve unloading time at the expense of picking time, thus lowering logistics costs and enhancing overall delivery efficiency.

Key words: e-commerce; joint scheduling; order picking; cargo box stacking; simulated annealing algorithm

随着电子商务行业的蓬勃发展,客户分散度高、单个订单需求量大、种类繁多等特征也愈发显著,同时客户对商品的快速交付期望也越来越高,使得电商行业的仓储物流面临着订单处理难度大和订单响应时效高两大挑战。对于电商来说,在最短时间内,以较低成本将商品送到客户手中十分重要,并协调好订单拣选与配送环节是需要解决的关键问题。

订单拣选和配送是两个密切相关、不可分割的环节,单独优化其中一个环节必然会忽略另一个环节的要求和约束^[1]。因此,集成的订单拣选与配送联合调度(IOPDS)比单一优化更有效^[2]。在研究IOPDS问题时,通常在确定配送路线后采用分批、分区等不同策略来优化拣选时间^[3]。然而,这种拣选顺序可能与配送顺序完全不同。在实际操作中,配送人员需要从以下两种方法中选择一种来装载打包好的订单容器(盒子、袋子、包裹等)装入车厢:首先,通过人工或自动化存取系统(AS/RS)将订单包装箱按照送货顺序进行分拣,从远到近装入送货车厢^[4]。这种方法会增加时间和再分类区域的成本消耗,但是在卸货时很容易获得所需的货箱;其次,按拣选顺序装入送货车厢,到达客户位置后搜索货箱并卸货,这种方法不需要在装载前进行排序,但需要在卸载时增加搜索时间。可以看出,无论选择哪种方式,订单履行时间都会增加。然而,在现有的IOPDS研究中,这些排序或搜索时间往往被省略或被固定的排序时间或服务时间所取代^[5],导致优化后的订单履行时间与现实不符,甚至远低于现实。因此,在IOPDS的研究和实践中,装卸对订单履行整体效率的影响不容忽视,本文旨在研究考虑卸货顺序的IOPDS优化策略,以提高订单履行的整体效率。

一、文献回顾

传统上,订单拣选和配送调度的优化策略通常分为两个问题分别研究。对于订单拣选问题,主要以提高订单拣选效率为目标,从拣选系统、仓库布局、存储分配、分区分批、路径规划等方面进行优化^[6-11]。订单配送调度问题主要以最短路径或最小成本为优化目标,根据车辆装载约束、时间窗约束和路径约束等优化车辆配送路线^[12-16]。

对IOPDS的研究一方面侧重于模型中带有时间窗等各种约束的配送路径优化,采用离散订单拣选策略,将拣选结果作为集成调度模型的输入变量。Moons等^[17]首次将订单拣选问题与车辆路径问题相结合,使用单一优化框架来解决订单拣选问题和带有时间窗和截止日期的车辆路径问题。Ramaekers等^[18]根据不同的客户特征、时间窗口特征和运营商规模,设置实验来研究允许客户选择首选交付时间窗口的服务成本。Schubert^[19]考虑了一个IOPDS问题,即交付的开始日期受物流中心订单拣选的影响,并且每个订单都有其到期日,并设计了一个迭代局部搜索算法,该算法也可以为大规模示例产生高质量的解决方案。

另一方面,一些研究更多地关注不同拣选场景下的集成优化,考虑了订单拣选优化和拣选员分配等问题。Zhang等^[20]研究了在线订单批量处理与配送的集成调度问题。针对车辆出发时间固定的在线订单的批处理和拣选员分配问题,通过定义订单的紧急程度,提出了一种基于规则的解决方案,能够在没有任何未来订单到达信息的情况下优化现有订单的批处理和拣选员分配;Wang等^[5]以订单履行时间最小化为目标,建立了单个拣选员、多辆配送车情况下的订单拣选配送一体化调度模型,解决了拣选顺序、拣

选批次和配送路线的一体化决策; Zhang 等^[1]考虑在 B2C 电子商务环境下,将现有的在线生产配送联合调度问题和在线订单分批问题的方法进行整合,提出了在线 4-竞争算法; Chen 等^[3]提出了一个混合整数模型来管理带时间窗的区域拣选和车辆路径问题的集成调度 (ISZPVRPTW),并开发了一个两阶段迭代搜索 (TIS) 算法来最小化作业成本和逾期罚款成本; Hossein Nia Shavaki 和 Jolai^[21]提出了一种基于规则的启发式算法,该算法集成了订单分批决策、分批拣选计划、向卡车分配订单以及卡车调度和路线安排。通过大量的数值实验,验证了基于规则的算法的有效性,并考察了每种分批方法在不同问题规模下的效果。然而,上述文献没有考虑车辆中订单的装载和卸载过程对 IOPDS 问题的影响。

目前,很少有关于考虑订单装卸规则的 IOPDS 问题的报道。类似的研究包括 Shiau 和 Lee^[22]开发了一种混合算法,包括货箱选择、装载配置和装载/拣选序列,以生成包装操作的拣选序列; Junqueira 等^[23]考虑了一个三维货箱装载问题,其中货箱的运送路线是预先已知的,并且考虑了货箱卸载的顺序,以避免在到达路线的每个卸载点时进行额外的处理,并且保证货箱之间不相互重叠和货物装载稳定,并针对该问题设计了一个基于混合整数线性规划模型 (MIP) 的方法; Sampaio 和 Urrutia^[24]遵循“后进先出”(LIFO) 原则建立了多栈取送旅行商问题模型,并提出了求解该问题的分支定界算法。上述文献虽然涉及到货物的卸载约束或货箱的装载,但对于拣选过程的内部处理、货箱的存取规则等都没有集成的决策研究。

综上所述,本文针对单拣选人员和单线路的自建物流人工拣选系统,研究了带卸箱顺序约束的订单拣选与配送联合调度优化问题。通过订单批量、拣选优化、堆码方式优化的集成决策,实现订单拣选和卸货时间最短。

二、问题描述

本文研究了由一个物流配送中心和多个客户组成的电子商务物流环境中的 IOPDS-BUS 问题。订单履行过程如图 1 所示。研究对象是前一天累积的 N 个客户订单,应该在今天完成配送。订单的拣选顺序与配送顺序相反并遵循“先来先服务”(FCFS) 订单批处理规则,即第一个订单形成一个批次,后续订单按顺序添加到批次中,直到批次达到其容量限制。然后下一个订单开始一个新的批次,这个批次处理根据 FCFS 规则继续进行。根据订单拣选顺序生成多个拣选批次,并分配给单个拣选员进行人工拣选。拣选流程完成后,每份订单都被装入一个货箱。然后,根据拣选顺序将货箱装载到车厢中,装载过程从车厢内部开始,从下到上,从右到左向外移动。货箱的卸载顺序 (BUS) 约束要求卸载顺序要和配送顺序相对应,而配送顺序由外部输入,如果即将卸载的货箱被其他货箱挡住,则应首先卸载位于该货箱同一列的前方和上方的未被配送货箱,该货箱交付后,为便于后续配送过程,其他被卸载的货箱应重新装载至原始位置或向下移动(如果它们被该货箱所支撑)。

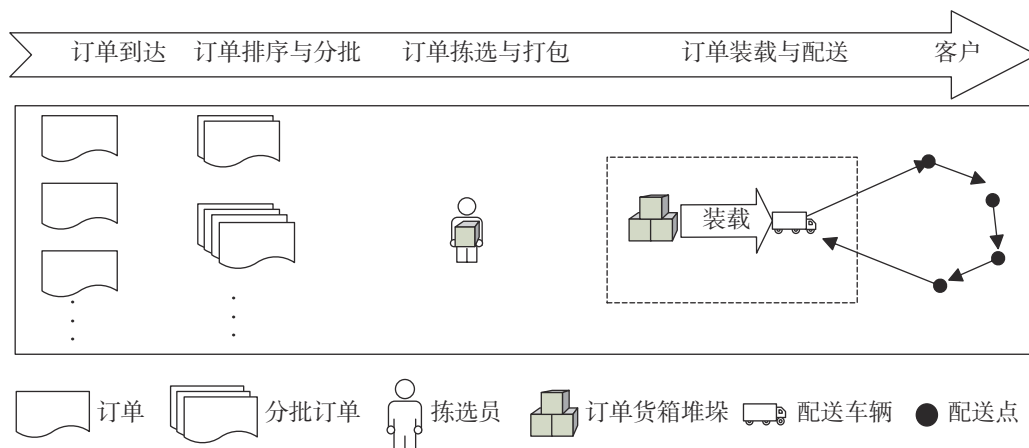


图 1 订单履行过程

在本研究中, 我们重点研究相同规格货箱的堆放方式, 装满货物的货箱按上述顺序堆放在送货车厢内, 一个长方体的送货车厢可以抽象成 $A \times B \times G$ 个等分的长方体小货位。 A , B 和 G 分别是可放入货箱长、高和宽方向的货箱数量。货箱在车厢中的堆叠如图 2 所示, 车厢中订单的堆叠位置为 $Lo_i = (\alpha_i, \beta_i, \gamma_i)$, α_i 为订单 i 的截面编号 (一个车厢有 A 个截面)。 β_i 是订单 i 所在的层。 γ_i 是订单 i 的列数。以 $3 \times 2 \times 2$ 车厢的第一横截面 (即堆垛 1) 为例, 配送顺序为 $\phi^s(1, 2, 3, 4)$, 对应订单所包含商品数为 $q^s(3, 1, 1, 4)$, 假设一次拣选的最大容量为 5。货箱堆积方式为图 2(a), 即堆放位置为 $Lo_1 = (1, 1, 1)$ 、 $Lo_2 = (1, 1, 2)$ 、 $Lo_3 = (1, 2, 1)$ 、 $Lo_4 = (1, 2, 2)$, 依据上述装车规则, 可以得到一个拣选顺序方案 $\eta(4, 3, 2, 1)$ 与分批方案 $M[m_1(4, 3), m_2(2, 1)]$ 。当堆积方式改变为图 2(b) 时, 堆放位置变为 $Lo_1 = (1, 1, 2)$ 、 $Lo_2 = (1, 2, 2)$ 、 $Lo_3 = (1, 1, 1)$ 、 $Lo_4 = (1, 2, 1)$, 依然满足按照配送顺序的无阻挡卸货要求, 但拣选顺序和分批方案发生了改变, 拣选顺序为 $\eta^s(2, 4, 1, 3)$ 与分批方案为 $M[m_1(2, 4), m_2(1, 3)]$ 。

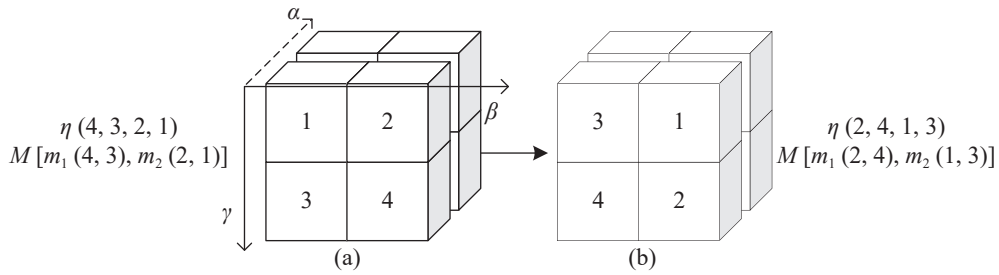


图2 货箱堆码演示

一个确定的卸货顺序可以根据上述拣选规则和装货规则产生多种不同的货箱堆放方式, 不同的堆放方式可以产生不同的拣选顺序和批次方案, 从而导致不同的拣选时间。本文通过在 BUS 约束下改变货箱的堆码方式, 可以得到更有效的拣选顺序和分批方案。

拣选与配送联合优化调度模型旨在优化堆码方式。优化过程有两个目的: 首先, 确保符合货箱卸载顺序 (BUS) 的要求, 最大限度地减少配送过程中任何额外的劳动和时间消耗; 其次, 提高拣选效率。本研究中所做的假设如下: 1) 订单可以分组为批次, 但不能进一步分割; 2) 不考虑拣选准备时间和装箱时间; 3) 不考虑拣选通道堵塞; 4) 根据 S-Shape 路径策略和 FCFS 分批规则拣选, 忽略分批拣选后拆分成独立订单的时间; 5) 人工拣选区的布局如图 3 所示。仓库为单隔断式, 通道出入口位于仓库左下端; 6) 用相同规格的货箱包装订单, 一个订单的所有货物可以且只会装在一个货箱里; 7) 在配送过程中, 货箱的卸载必须符合预定的配送顺序。

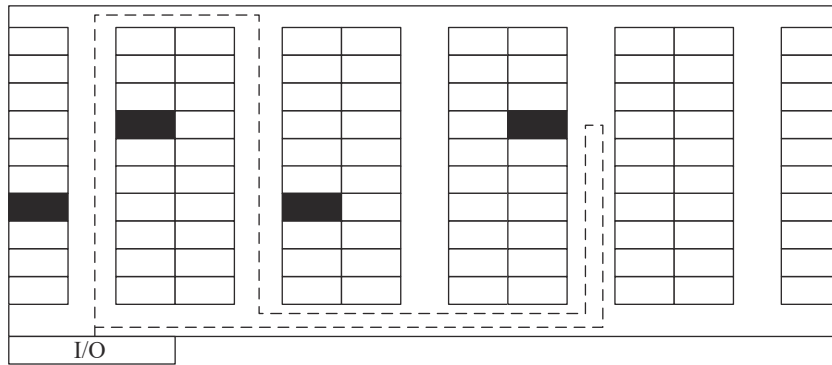


图3 人工拣选区域布局

三、IOPDS-CSR 模型构建

对模型中的常量和变量作如下定义, 如表 1 所示。

表 1 符号说明

符号		说明
N		配送顺序: $(1 \cdots n)$
M		订单批次集合: $(1 \cdots m)$
q_i		订单 <i>i</i> 包含的货物数
v_p		拣选员的步行速度
Q		单个拣选员依次拣选的最大容量
t^{op}		装载或卸载一个货箱需要的时间
B_j		批次 <i>j</i> 开始拣选时间, $B_1 = 0$
ops^i		卸载订单 <i>i</i> 的货箱的操作次数
t^{ops}		根据配送顺序卸载所有货箱的时间
T		优化目标
决策变量	$L_{\alpha\beta\gamma}^i$	0—1变量, 若货箱 <i>i</i> (订单 <i>i</i>) 被堆放在截面 α , 层 β 和 列 γ 则为1, 否则为0.并使 $L_{\alpha\beta\gamma}^0 = 0, \forall \alpha, \beta, \gamma$
	α_i	订单 <i>i</i> 在货箱中所在的截面
	β_i	订单 <i>i</i> 在货箱中所在的层
	γ_i	订单 <i>i</i> 在货箱中所在的列
	z_{ik}	0—1变量, 当订单 <i>i</i> 被分配给批次 <i>k</i> 则为1, 否则为0
	d_k	完成拣选批次 <i>k</i> 的行走距离

在上述分析的基础上, 建立 IOPDS-BUS 数学模型, 目标是最小化从第一个订单拣选到最后一个订单交付的总拣选和卸货时间, 构建了目标函数:

$$\min t^{pick} + t^{ops} \quad (1)$$

s.t. :

$$\sum_{i=1}^n z_{ik} q_i \leq Q, \forall k \in M \quad (2)$$

$$\sum_{k=1}^m z_{ik} = 1, \forall i \in N \quad (3)$$

$$B_j = B_{j-1} + \frac{d_k}{v_p}, j \geq 2 \quad (4)$$

$$t^{pick} = B_m + \frac{d_m}{v_p} \quad (5)$$

$$\alpha_i \geq \alpha_j; \forall i, j \in N, \gamma_i = \gamma_j, i > j \quad (6)$$

$$\beta_i \geq \beta_j; \forall i, j \in N, \gamma_i = \gamma_j, \alpha_i \geq \alpha_j, i > j \quad (7)$$

$$\sum_{\alpha=1}^A \sum_{\beta=1}^B \sum_{\gamma=1}^G L_{\alpha\beta\gamma}^i = 1, \forall i \in N \quad (8)$$

$$ops^i = \begin{cases} 2 \cdot \left(\sum_{i=0}^n \left(\sum_{\alpha=1}^{\alpha_i-1} \sum_{\beta=1}^B L_{\alpha\beta\gamma_i}^i + \sum_{\beta=1}^{\beta_i} L_{\alpha_i\beta\gamma_i}^i \right) - \left(\sum_{i=0}^{i-1} \left(\sum_{\alpha=1}^{\alpha_i-1} \sum_{\beta=1}^B L_{\alpha\beta\gamma_i}^i + \sum_{\beta=1}^{\beta_i} L_{\alpha_i\beta\gamma_i}^i \right) \right) \right) - 1, \alpha_i > 1 \\ 2 \cdot \left(\sum_{i=0}^n \sum_{\beta=1}^{\beta_i} L_{\alpha_i\beta\gamma_i}^i - \sum_{i=0}^{i-1} \sum_{\beta=1}^{\beta_i} L_{\alpha_i\beta\gamma_i}^i \right) - 1, \alpha_i = 1; \end{cases} \quad (9)$$

$$t^{ops} = \sum_{i=1}^n ops^i \cdot t^{op} \quad (10)$$

目标函数 (1) 旨在最小化拣选和交付所有订单的总时间, 因为配送路线保持固定所以车辆的行驶时间不包括在目标函数中。

约束 (2)–(5) 为拣选阶段约束, 其中: 约束 (2) 限制每批订单包含货物数量小于拣选容量; (3) 约束一个订单只能分配给一个批次; (4) 为批次开始拣选时间, 其中 d_k 为 S 形人工拣选路线策略下拣选批次 k 所有订单的行走距离^[25], 采用 Hall 等^[26] 的距离计算方法; (5) 为拣选完所有订单需要的时间。约束 (6) 和 (7) 分别限制交货顺序固定时货箱的堆放位置; 约束 (6) 限制在车厢中, 配送顺序在后的订单的货箱所在截面不能在同列配送顺序靠前的订单的货箱所在截面之前; 约束 (7) 限制在同截面同一列中, 配送顺序靠后的订单的货箱所在层数小于或等于配送顺序靠前订单的货箱所在层数; 约束 (8) 限制每个订单的货箱只占用一个位置; 约束 (9) 和 (10) 是配送所有订单所需的装货和卸货时间的计算。

对于目标函数 $\min(t^{pick} + t^{ops})$, 可以发现, 优化目标主要包括两部分: t^{pick} 与 t^{ops} 。 t^{ops} 由车厢中货箱的堆放方式和配送顺序决定。 t^{pick} 是由订单的拣选顺序及其形成的批次决定的, 拣选顺序和分批方案都是通过拣选规则根据货箱堆码的方式形成的。因此, IOPDS-BUS 算法的思想是通过 ISA 调整有 BUS 限制的货箱的堆码方式, 搜索最优拣选效率。

四、IOPDS-BUS 的求解算法设计

受金属热处理中退火概念的启发, Metropolis 等引入了模拟退火算法 (SA)^[26], 且因其在处理离散变量优化和非线性优化问题上的独特优势而受到研究者的青睐。然而, 传统的模拟退火算法存在收敛速度慢、运行时间长、难以在有限时间内找到最优解、易陷入局部最优等局限性。随着问题规模的扩大, 这些缺点变得更加明显。本文通过对算法的两个改进来解决这些问题: 可行解的编码与生成和自适应升温。

(一) 可行解的编码和生成

IOPDS-BUS 模型的解对应于车厢中货箱的堆叠模式。为了便于生成新解, 将初始拣选序列编码为矩阵。编码规则遵循将货箱装入车厢的顺序, 订单从里到外、从下到上、从右到左, 与交货顺序相反。这种编码方案产生 A 个 $B \times G$ 大小的矩阵, 一个矩阵表示车厢中的一个截面, 货箱的空位置被替换为 0。例如图 4, 对于配送顺序 $N(1, 2, 3, 4, 5, 6)$, 在 $2 \times 2 \times 2$ 货箱形成中的编码矩阵。

$$\begin{matrix} \begin{bmatrix} 0 & 0 \\ 1 & 2 \end{bmatrix} & \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} \\ \alpha=1 & \alpha=2 \end{matrix}$$

图 4 基于矩阵的编码示意图

利用上述的编码方案, 基于概率 $\theta(\theta_1, \theta_2)$ 使用列变化和点变化应用于当前解, 从而生成新解。同时, 这两种方式可以确保生成的解符合约束 (6) 和 (7), 确保解满足 BUS 约束和解的有效性。图 5 示出了集中于单个横截面的两种邻域变化策略。

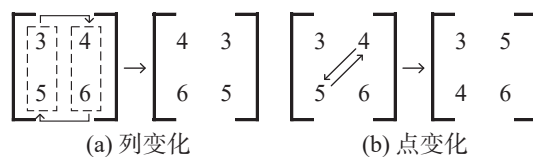


图 5 领域搜索演示

(二) 基于 Prime 最小生成树的初始解生成

在启发式算法中,有效的初始解可以加速收敛,减少随机波动。本文采用一种 Prime 最小生成树方法来生成初始解。Prime 算法旨在通过不断添加新的边和点,直到所有的点都连接起来,从而最小化加权连通图中的总权重。本研究中生成初始解的具体步骤如下。

第一步:从配送的相反顺序开始,将每个货箱放入车厢中可用且可行的位置。

第二步:计算与将当前货箱放在每个可行位置的拣选时间,将其视为选择该位置的权重。

第三步:选择车厢中重量最小的位置作为货箱的放置位置。从步骤 1 开始重复,直到配送顺序中所有订单的货箱都已放置。

如图 6 所示,配送顺序为 (1、2、3、4、5、6),三个货箱已放入车厢。下一个要配送的是货箱 3,可行的位置用虚线框表示。通过评估将货箱 3 放在这些位置中的每一个的拣选时间,具有最低拣选时间的那个被选择作为货箱 3 的最佳放置位置。

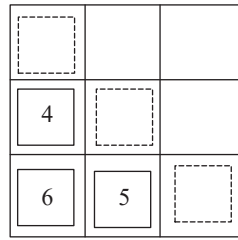


图 6 初始解的生成

(三) 自适应升温

Metropolis 准则中的温度值意味着算法不同阶段对非最优解的接受程度,高温阶段更容易接受非最优解,因此更容易跳出局部最优解。当模拟退火算法陷入局部最优解时,提高算法过程中的温度有助于跳出局部最优解。

每个算法外循环将有一个当前最优结果,即 $RElist = \{R_1, \dots, R_i\}$, 其中 R_i 是第 i 个外循环后的最优解。本文试图设计一种自适应升温机制,如果 R_i 在外部循环完成 K 次 (通过实验获得) 后没有改善,将进行升温操作:

$$T_{i+1} = \max \left(T_0 \cdot \frac{\min(RElis)}{\max(RElis)}, T_i \cdot Coolingpara \right)$$

$$K_{new} = K * \frac{T_0}{T_0 - T}$$

其中 T_i 是当前温度, T_{i+1} 是下一次外部循环的温度, T_0 是算法开始时的温度, $Coolingpara$ 是等比冷却的冷却参数。在算法的前期,温度较高,可能会导致 K 次循环得不到更好的解的情况,使得 $T_0 * (\min(RElis)/\max(RElis))$ 小于 $T_i \cdot Coolingpara$, 此时使用加热操作,温度会迅速下降,所以此时不进行加热操作。

加热操作前记录当前温度。这种加热只会第 $i+1$ 次外环的内环阶段产生影响,同时被记录为一个新的 T_0 , 并更新 K 的值,因为加热一次后应该给算法时间找到更好的解。内循环完成后,将记录的温度作为当前温度,继续降温。通过加热操作,优化过程可以跳出局部最优解,同时随着算法的运行,加热操作会根据优化程度自适应缩小加热范围,提高寻找最优解的效率。

五、数值实验

(一) 参数设置

随机生成小、中、大三组订单,由于订单的小批量特点,每个订单的货物需求在 [1, 5] 上服从均匀分布,拣选区域有 10 个拣选通道,1000 种商品,每个通道长 50 米,宽 2 米,每个货架存放 50 种商品。总共

有1个拣选员,一个拣选员一次最大容量为10件商品,每种商品只能存放在一个货架上;员工走在通道中间同时从两边取货,员工的行走速度为125 m/min。员工一般从最左边的入口进入,拣选完所有货物后返回入口。货箱可以装载单个客户订单中的所有货物。配送员在配送点卸货或装载单个货箱需要2 s。ISA参数设置:内循环次数:50次,冷却速率0.95,其他参数见表2。

表2 ISA算法参数

订单组	车厢规格 $\alpha \times \beta \times \gamma$	初始温度	外循环次数	升温参数
Ordersize_36	4×3×3	$T_0=30$	200	$K=15$
Ordersize_80	5×4×4	$T_0=40$	300	$K=20$
Ordersize_200	8×5×5	$T_0=50$	500	$K=30$

(二) 结果分析

1. ISA算法结果分析

使用传统SA算法与ISA算法测试了三个不同的数据集,每个数据集的使用每种算法测试五次,取平均值,结果如表3所示,不同算法和订单规模的求解过程如图7与图8所示。虽然配送过程中的卸载时间受配送顺序和货箱堆放方式的影响,但值得注意的是,两种算法中的所有解(货箱堆放模式)都遵循BUS,从而避免了任何额外的卸载时间。因此,算法结果中优化的主要焦点在于减少拣选时间。

表3 ISA与SA算法的结果分析

订单组	算法	最优拣选时间/分钟	卸货时间/分钟	优化率
Ordersize_36	SA	24.3	1.2	5.5%
	ISA	22.9		
Ordersize_80	SA	56.7	2.67	10.4%
	ISA	50.5		
Ordersize_200	SA	180.4	6.67	14.7%
	ISA	152.9		

表3为ISA算法获得的最优解,其在不同的订单规模上有不同程度的提高。例如,最小订单测试示例显示了5.5%的改进,而最大订单测试示例显示了14.7%的改进。这表明本文提出的ISA算法能够获得更优的优化结果,产生更适合的装箱方式。此外,改进率随着订单大小的增加而增加,这表明ISA在较大的数据集也表现出良好的适应性。

从图7可以看出,传统的模拟退火算法容易陷入局部最优。对于36的订单大小,SA在大约100次迭代后落入局部最优。对于80的订单大小,它在100到200次迭代内落入局部最优,对于200的订单大小,它在大约300次迭代后落入局部最优。图8显示了使用ISA算法解决问题的过程。当在一定次数的迭代后没有找到更好的解决方案时,ISA算法采用加热过程跳出局部最优。曲线中的突然波动代表加热过程发生的点。具体来说,在订单大小为200的ISA算法的第五张图中,显然温度增加并跳出局部最优。在这个求解过程中,温度在第206次、第314次和第449次外部循环迭代时升高。在第175次外部循环迭代时,该算法在值为172.03的30次连续迭代中落入局部最优。然后在第206次外循环迭代时跳出局部最优,随后下降几次,直到第270次外循环迭代,值为165.4。在此过程中,加热参数(K 值)更新为大约43。在第314次外循环迭代时,温度再次升高,在第389次外循环迭代时找到更好的解,值为150.05。然后加热参数被更新到59左右。然而,尽管在第449次外循环迭代时温度再次升高,但是直到算法结束都没有观察到进一步的改善。

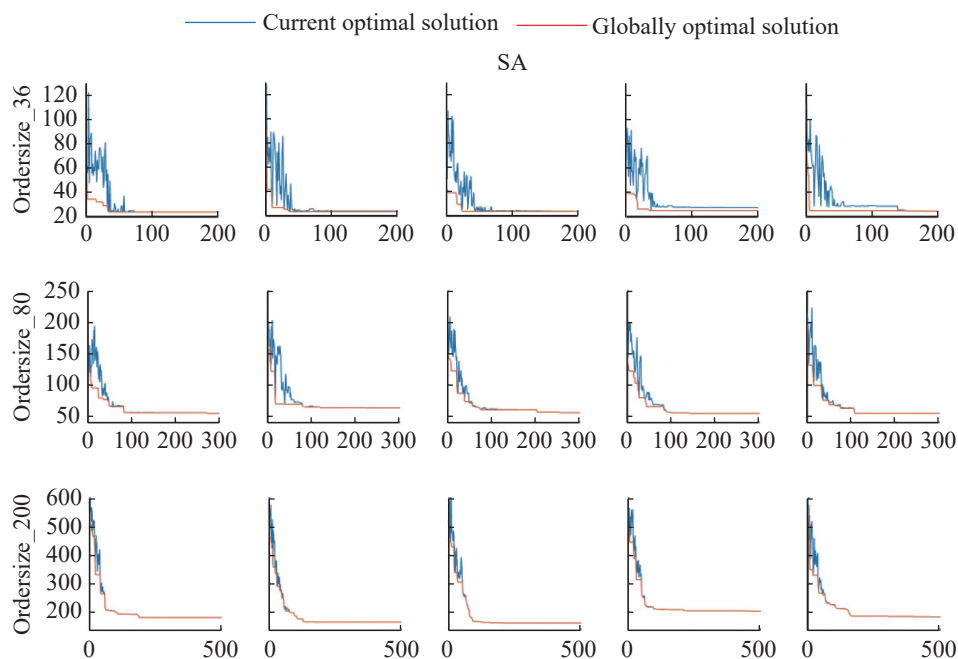


图7 不同订单大小下SA的求解过程

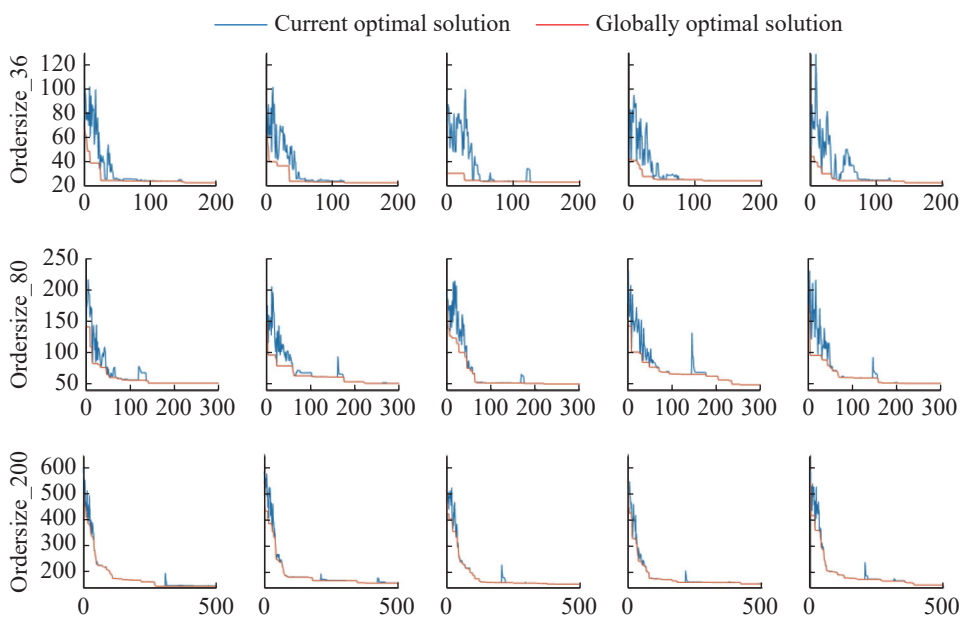


图8 不同订单大小下ISA的求解过程

2. 对于BUS约束的结果分析

在本节中,使用传统的物流中心拣选与配送实践(方案a),以及采用文献中IOPDS问题解决方案^{[5][27]}(方案b),来评估考虑BUS约束对卸货和拣选时间的影响,即本文提出的IOPDS解决方案(方案c)。方案a:为了确保流程的清晰性和最大限度地减少混乱,通常会根据配送顺序的相反顺序拣选订单,这意味着最先拣选的订单最后配送,从而实现直接装载,使车厢中货箱的堆放方式与配送顺序保持一致,假设配送顺序为 $N(1,2,3,\dots,n)$,由配送顺序的逆序形成拣选顺序: $\eta(n,n-1,\dots,1)$ 和一个分批方案 $M(m_1(n,n-1),m_2(n-2,\dots),\dots)$,并按拣选顺序装车。方案b:传统解决IOPDS问题的方法是先找到拣选的最佳分批方式和拣选顺序,然后包装并装载货箱,配送过程中从货车中搜索货箱并交付。在本文中的所有解决方案,订单的装载直接在拣选完成后进行,无额外的再分类过程。

以订单规模为36的测试案例为例,方案a、方案b和方案c的优化拣选顺序和装箱方案如表4和表5所示,其中表4主要显示了三种不同操作方法得到的拣选顺序和批次方案,表5显示了拣选顺序逆序得到的装箱方式。其中表中数字代表订单编号, α_i 代表订单*i*的截面数,在方案a中,拣选和装货方式取决于交货顺序,因此卸货时间短,只有1.2分钟,但拣选时间极长,达到108.2分钟,总计110分钟。相比之下,在方案b模式下,通过优化拣选顺序,不考虑配送阶段的卸载,拣选时间减少到20.8分钟,但卸载时间增加到8.7分钟,共计29.5分钟。在方案c中,同时考虑卸载过程和拣选过程的约束,拣选时间为22.3分钟,卸载时间为1.2分钟,共计23.5分钟。方案c对方案a和方案b的优化程度分别为78.6%和20.3%。

表4 大小为36的订单组在不同方案下的拣选顺序和分批方案

Ordersize_36	配送顺序	1→2→3→4→5→6→7→8→9→10→11→12→13→14→15→16→17→18→19→20→21→22→23→24→25→26→27→28→29→30→31→32→33→34→35→36
方案a	拣选顺序	36→35→34→33→32→31→30→29→28→27→26→25→24→23→22→21→20→19→18→17→16→15→14→13→12→11→10→9→8→7→6→5→4→3→2→1
	分批方案	[36,35,34,33,32]→[31,30,29]→[28,27]→[26,25]→[24,23]→[22,21,20]→[19,18]→[17,16]→[15,14,13,12]→[11,10]→[9,8,7]→[6,5]→[4,3]→[2,1]
方案b	拣选顺序	12→19→5→18→13→25→16→17→35→3→15→21→24→32→4→20→14→33→26→10→8→28→1→2→11→22→7→31→9→30→29→36→23→6→34→27
	分批方案	[12,19,5]→[18,13,25]→[16,17]→[35,3,15]→[21,24]→[32,4,20,14,33]→[26,10,8]→[28,1]→[2,11,22]→[7,31,9,30]→[29,36,23]→[6,34,27]
方案c	拣选顺序	32→36→35→31→34→29→30→33→28→24→26→27→21→22→25→19→20→23→17→16→18→12→14→15→10→11→13→7→6→9→3→5→8→1→4→2
	分批方案	[32,36,35,31]→[34,29,30]→[33,28]→[24,26]→[27,21,22]→[25,19,20]→[23,17]→[16,18,12,14,15,10]→[11,13,7]→[6,9,3,5]→[8,1,4]→2

表5 大小为36的订单组在不同方案下的货箱堆码方式

	货箱堆码方式												拣选时间/分钟	卸货时间/分钟
	$\alpha_i = 1$			$\alpha_i = 2$			$\alpha_i = 3$			$\alpha_i = 4$				
方案a	1	2	3	10	11	12	19	20	21	28	29	30	108.2	1.2
	4	5	6	13	14	15	22	23	24	31	32	33		
	7	8	9	16	17	18	25	26	27	34	35	36		
方案b	27	34	6	7	22	11	33	14	20	35	17	16	20.8	8.7
	23	36	29	2	1	28	4	32	24	25	13	18		
	30	9	31	8	10	26	21	15	3	5	19	12		
方案c	2	4	1	13	11	10	23	20	19	28	33	30	22.3	1.2
	8	5	3	15	14	12	25	22	21	29	34	31		
	9	6	7	18	16	17	27	26	24	35	36	32		

由于拣选阶段没有考虑配送阶段的卸载过程,方案b在拣选时间上要优于方案c。但是,卸货时间大大影响了配送效率。例如,在方案b的货箱堆放方式下,配送1号箱时,所有 $\alpha_i = 1$ 的同一列货箱和 $\alpha_i = 2$ 且层数比1号货箱高的货箱,即6号货箱、29号货箱、31号货箱和22号货箱,都要在卸载1号箱之前卸载,1号箱配送给客户后,需要按照之前的堆码方式重新装载多卸载的4个货箱,方便下一次配送,这次配送过程的卸货要进行9次装卸货操作,共计18秒(每次2秒),所以在配送的过程中按照配送顺序卸完所有货物需要8.6分钟。方案c的货箱堆放方式满足送货顺序卸货,不会被其他货箱挡住,按照送货顺序卸完所有货物只需要1.2分钟。进一步证明了方案c在节省卸货时间方面可以取得显著的效果,而方案b的反复卸货和装载造成时间的浪费,且容使货箱或货物损坏。

通过以上分析,实验结果验证了模型和算法的有效性,并可以得出以下结论:1)本文提出的基于 Prime 算法生成的初始解和自适应加热操作的改进模拟退火算法可以有效提高优化能力,在小、中、大订单上分别达到 5.5%、10.4% 和 14.7% 的优化效率;2) IOPDS-BUS 优于传统方法和传统 IOPDS 解决方案,在订单大小为 36 的情况下,性能时间的改善程度分别为 78.6% 和 20.3%。可以看出, IOPDS-BUS 可以进一步提高电子商务的整体物流效率。3) 总运行时间反映了员工的工作强度和物流成本水平,与传统 IOPDS 解决方案相比, IOPDS-BUS 的总运行时间提高了 20.3%,可以有效降低工作人员的工作强度和物流成本。4) 在物流配送实践中,很难同时实现拣选时间和卸货时间的同时最优化,考虑 BUS 而增加较少的拣选时间实现更多的卸货时间的节省,使整体物流效率更高,配送卸货更方便,更符合电商物流的实际。

结语

本文主要研究一个拣选员和一条配送路线的人工拣选物流中心的订单拣选和配送联合调度问题,重点考虑的是货箱卸载约束(BUS)。为了最大限度地减少拣选和卸货时间,我们首先分析了配送车辆中不同的箱堆码方式对订单拣选顺序和分批方式的影响,提出考虑货箱卸载顺序的订单拣选和配送联合调度模型。为了求解该模型,我们设计了一种自适应加热的改进模拟退火算法。并使用不同的大小的订单组进行数值实验。结果表明,该算法有效地解决了 IOPDS-BUS 问题,对于小订单、中订单和大订单,优化性能分别提高了 5.5%、10.4% 和 14.7%。此外,我们将物流中心当前的拣选和配送实践与 IOPDS 和 IOPDS-BUS 在拣选和卸货时间方面的进行了比较,结果表明, IOPDS-BUS 显著减少了拣选和卸货时间。此外, IOPDS-BUS 适当增加订单履行过程中在物流中心拣选时间,从而缩短了卸货时间,提高了整体配送效率。

本文只考虑线下物流的订单拣选与配送联合调度优化,同时物流中心布局相对较简单,进一步研究线上物流背景的联合调度。另外,本文的货箱模型假设所有客户的订单都可以装在一个同规格的货箱里,进一步研究可以采用多规格的货箱或送货车辆。

参考文献:

- [1] ZHANG J, WANG X, HUANG K. On-line scheduling of order picking and delivery with multiple zones and limited vehicle capacity[J]. *Omega-international journal of management science*, 2018(79): 104 – 115.
- [2] MOONS S, BRAEKERS K, RAMAEKERS K, et al. The value of integrating order picking and vehicle routing decisions in a b2c e-commerce environment[J]. *International journal of production research*, 2019, 57(20): 6405 – 6423.
- [3] CHEN W, ZHANG Y, ZHOU Y. Integrated scheduling of zone picking and vehicle routing problem with time windows in the front warehouse mode[J]. *Computers & industrial engineering*, 2022, 163(1): 1 – 14.
- [4] BOYSEN N, FEDTKE S, WEIDINGER F. Optimizing automated sorting in warehouses: The minimum order spread sequencing problem[J]. *European journal of operational research*, 2018, 270(1): 386 – 400.
- [5] 王旭坪, 张珺, 易彩玉. B2C 电子商务环境下订单拣选与配送联合调度优化[J]. *中国管理科学*, 2016, 24(7): 101 – 109.
- [6] DE KOSTER R, LE-DUC T, ROODBERGEN K J. Design and control of warehouse order picking: A literature review[J]. *European journal of operational research*, 2007, 182(2): 481 – 501.
- [7] MOU S D. Integrated order picking and multi-skilled picker scheduling in omni-channel retail stores[J]. *Mathematics*, 2022, 10(9): 1 – 19.
- [8] HORTA M, COELHO F, RELVAS S. Layout design modelling for a real world just-in-time warehouse[J]. *Computers & industrial engineering*, 2016(101): 1 – 9.
- [9] WINKELHAUS S, ZHANG M, GROSSE E H, et al. Hybrid order picking: A simulation model of a joint manual and autonomous order picking system[J]. *Computers & industrial engineering*, 2022(167): 1 – 17.
- [10] PARIKH P J, MELLER R D. Selecting between batch and zone order picking strategies in a distribution center[J]. *Transportation research part e-logistics and transportation review*, 2008, 44(5): 696 – 719.
- [11] DIJKSTRA A S, ROODBERGEN K J. Exact route-length formulas and a storage location assignment heuristic for picker-to-parts warehouses[J]. *Transportation research part e-logistics and transportation review*, 2017(102): 38 – 59.
- [12] WON J, OLAFSSON S. Joint order batching and order picking in warehouse operations[J]. *International journal of production research*, 2005, 43(7): 1427 – 1442.

- [13] ALONSO M T, MARTINEZ-SYKORA A, ALVAREZ-VALDES R, et al. The pallet-loading vehicle routing problem with stability constraints[J]. *European journal of operational research*, 2022, 302(3): 860 – 873.
- [14] LESPAY H, SUCHAN K. Territory design for the multi-period vehicle routing problem with time windows[J]. *Computers & operations research*, 2022, 145(1): 1 – 13.
- [15] ARCHETTI C, SPERANZA M G. Vehicle routing problems with split deliveries[J]. *International transactions in operational research*, 2012, 19(1-2): 3 – 22.
- [16] BAPTISTA S, OLIVEIRA R C, ZÚQUETE E. A period vehicle routing case study[J]. *European journal of operational research*, 2002, 139(2): 220 – 229.
- [17] MOONS S, RAMAEKERS K, CARIS A, et al. Integration of order picking and vehicle routing in a B2C e-commerce context[J]. *Flexible services and manufacturing journal*, 2018, 30(4): 813 – 843.
- [18] RAMAEKERS K, CARIS A, MOONS S, et al. Using an integrated order picking-vehicle routing problem to study the impact of delivery time windows in e-commerce[J]. *European transport research review*, 2018, 10(2): 1 – 11.
- [19] SCHUBERT D, SCHOLZ A, WAESCHER G. Integrated order picking and vehicle routing with due dates[J]. *Or spectrum*, 2018, 40(4): 1109 – 1139.
- [20] ZHANG J, WANG X, KAI H. Integrated on-line scheduling of order batching and delivery under B2C e-commerce[J]. *Computers & industrial engineering*, 2016, 94(Apr.): 280 – 289.
- [21] HOSSEIN NIA SHAVAKI F, JOLAI F. A rule-based heuristic algorithm for joint order batching and delivery planning of online retailers with multiple order pickers[J]. *Applied intelligence*, 2021, 51(6): 3917 – 3935.
- [22] SHIAU J-Y, LEE M-C. A warehouse management system with sequential picking for multi-container deliveries[J]. *Computers & industrial engineering*, 2010, 58(3): 382 – 392.
- [23] JUNQUEIRA L, MORABITO R, SATO YAMASHITA D. Mip-based approaches for the container loading problem with multi-drop constraints[J]. *Annals of operations research*, 2011, 199(1): 51 – 75.
- [24] SAMPAIO A H, URRUTIA S. New formulation and branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks[J]. *International transactions in operational research*, 2017, 24(1-2): 77 – 98.
- [25] HALL R W. Distance approximations for routing manual pickers in a warehouse[J]. *IIE transactions*, 1993, 25(4): 76 – 87.
- [26] METROPOLIS N, ROSENBLUTH A W, ROSENBLUTH M N, et al. Equation of state calculations by fast computing machines[J]. *The journal of chemical physics*, 1953, 21(6): 1087 – 1092.
- [27] 陈伟, 张玉林. “仓店一体化”模式下订单拣选与配送集成调度优化[J]. *运筹与管理*, 2021, 30(7): 16 – 22.

[责任编辑 刘书亮]