E-Commerce Letters 电子商务评论, 2025, 14(11), 3025-3037
Published Online November 2025 in Hans. https://www.hanspub.org/journal/ecl
https://doi.org/10.12677/ecl.2025.14113777

Hans汉斯

电子商务仓储环境下基于改进A*算法的AGV路径规划研究

杨林辉[1]，杨中华[1,2]

[1]武汉科技大学管理学院，湖北 武汉
[2]武汉科技大学服务科学与工程研究中心，湖北 武汉

收稿日期：2025年10月13日；录用日期：2025年10月29日；发布日期：2025年11月28日

摘 要

针对电子商务仓储环境下AGV路径规划的冲突问题，提出一种基于坐标保留表和冲突分类的无冲突多目标算法CF-MOWVRP (Conflict-Free Multi-Objective Warehousing Vehicle Routing Problem)。该算法首先构建鱼骨式栅格地图，模拟货架、拣选台和动态任务环境；其次改进A*算法，支持4方向搜索和路径缓存，然后建立多目标优化模型，最小化总运输距离、最大单次距离和冲突等待时间，并通过遗传操作、偏好随机策略和强化学习求解Pareto前沿；最后利用折线优化和二次Bezier曲线平滑路径，提高AGV运动平滑度。仿真结果表明：在30个动态任务场景下，相比传统A*，节点减少33%、转弯减少50%、完成时间降低11%。AGV增至10辆、任务增至60个、障碍比例增至0.45时，完成时间均降低约17%，验证了算法在高并行、高密度和复杂环境中的鲁棒性。

关键词

电子商务仓储，AGV路径规划，A*算法

Research on AGV Path Planning Based on Improved A* Algorithm in E-Commerce Warehousing Environment

Linhui Yang[1], Zhonghua Yang[1,2]

[1]School of Management, Wuhan University of Science & Technology, Wuhan Hubei
[2]Center for Service Science and Engineering, Wuhan University of Science & Technology, Wuhan Hubei

Received: October 13, 2025; accepted: October 29, 2025; published: November 28, 2025

文章引用: 杨林辉, 杨中华. 电子商务仓储环境下基于改进 A*算法的 AGV 路径规划研究[J]. 电子商务评论, 2025, 14(11): 3025-3037. DOI: 10.12677/ecl.2025.14113777

Abstract

Addressing the conflict issue of AGV path planning in the e-commerce warehousing environment, a conflict-free multi-objective algorithm CF-MOWVRP (Conflict-Free Multi-Objective Warehousing Vehicle Routing Problem) based on coordinate retention table and conflict classification is proposed. This algorithm first builds a fishbone-like grid map to simulate shelves, picking stations and dynamic task environments; then improves the A* algorithm to support four-direction search and path caching; then establishes a multi-objective optimization model to minimize total transportation distance, maximum single distance and conflict waiting time, and solves the Pareto frontier through genetic operations, preference random strategies and reinforcement learning; finally, it optimizes the path with polyline optimization and quadratic Bezier curve smoothing to improve the smoothness of AGV movement. The simulation results show that in 30 dynamic task scenarios, compared to traditional A*, nodes are reduced by 33%, turns are reduced by 50%, and completion time is reduced by 11%. When the number of AGVs increased to 10, the number of tasks increased to 60, and the obstacle ratio increased to 0.45, the completion time decreased by about 17%, verifying the robustness of the algorithm in high parallel, high-density, and complex environments.

Keywords

E-Commerce Warehousing, AGV Path Planning, A* Algorithm

1. 引言

随着电子商务的蓬勃发展，仓储自动化与效率成为物流行业的关键瓶颈，电子商务仓储系统已成为提升供应链效率的关键环节[1]。在电商平台如阿里巴巴、京东等推动下，仓库自动化水平不断提高。AGV (Automated Guided Vehicle)作为核心设备，也称自动搬运机器人，其路径规划效率直接影响整个仓储系统的性能。AGV 内部集成自动导引装置，能够借助各种智能算法、先进管理系统和其他智能技术，完成各种环境下物资的搬运、分拣和配送[2]。路径规划就是能够使 AGV 按照最短路径、最快到达、最少拐弯等原则从预先设定的起始点规划出一条到达目标点的最佳路径，然后 AGV 安全、可靠、高效地将物资从起始点搬运至目标点[3]。近年来，随着社会需求增长，物资生产和运输规模不断扩大，AGV 被广泛应用于机场、码头、医院等场所，具有广阔的发展前景[4]。

针对 AGV 路径规划，传统的 A*算法存在遍历节点数多、转弯次数较多及规划路径过障碍物顶点导致实际环境中容易与障碍物发生碰撞等问题[5]。为了解决这些缺点，许多学者展开了研究。杨秀建等对经典 A*算法进行了改进研究，提出了斜八邻域扩展的概念，与四邻域扩展结合组成一种新的双邻域选择扩展策略，在路径搜索过程中可以有效减少扩展节点的数量[6]。李兴州等人[7]提出一种多领域多方向搜索方式改进的 A*路径规划算法用来减少路径轨迹中的节点数量。按照现有的研究思路，本文计划通过改进扩展节点搜索方式以达到有效避免路径接触障碍物的目的。

同时，多目标优化在 AGV 路径规划中逐渐受到关注，Pareto 前沿方法也随之运用于权衡总距离、最小化最大完成时间及等待时间等多目标。吴自松等人设计了一种两阶段混合算法：第一阶段通过引入爬山算法改进非支配排序遗传算法的局部搜索策略，求解一组 Pareto 前沿集；第二阶段由 K-means 对 Pareto

前沿集进行剪枝[8]。此外，遗传算法和强化学习也常被用于求解多目标模型，例如 P-MOCO 算法结合 Pareto 优化与冲突检测，显著提升了路径规划效率[9]。

复杂的电商仓储环境中，来回穿梭的小车易产生冲突和碰撞，路径规划时也容易陷入局部最优陷阱。为此，任明辉等提出了基于改进冲突搜索的路径规划模型(iCBS-pri)，该改进模型主要由任务分配、单 AGV 路径规划、多 AGV 冲突检测与解决 3 个模块组成[10]。此外，宫婧等提出基于无冲突路径算法的多目标智能仓储路径规划，构建鱼骨布局并使用 P-MOCO 求解多目标模型，但忽略了动态任务下的实时冲突调整[11]。

基于以上文献研究总结的现有不足，如传统 A 算法在复杂环境中易产生节点冗余和碰撞、P-MOCO 对动态任务实时调整的局限性以及鱼骨布局下冲突解决的空白，本文提出了一种改进 A*算法的多 AGV 路径规划算法。通过构建鱼骨式栅格地图并集成缓存和顶点避让优化路径搜索，引入 CF-MOWVRP 算法实现多目标优化与实时冲突调整，并采用 Bezier 曲线平滑路径，以有效弥补动态任务下无冲突路径规划的不足。创新点包括：(1) 鱼骨式栅格地图建模，支持动态任务生成；(2) 改进 A*集成缓存和顶点避让；(3) 多目标模型与 CF-MOWVRP 求解，实现无冲突。

2. 模型与问题定义

2.1. 电商仓储环境建模

本文采用栅格地图建模方法，该方法建模简单且易于维护，能够有效模拟 AGV 在电商仓库复杂布局下的运动约束。根据鱼骨式布局设计，本文构建了一个 14 × 25 的栅格地图模型，单位栅格边长为 1 m (见图 1)。地图中黑色区域代表货架障碍物(AGV 不可通行)，白色区域表示可通行路径区域，灰色区域为拣选台(部分占用)。具体布局如下：9 个货架区分布在横向三组，纵向三层，中间隔 2 格空道，便于 AGV 穿梭。每个货架区标记编号 1~9。3 个 1 × 2 拣选台，位于左侧，平行于货架行，标记编号 1~3。AGV 空载时障碍物仅为空置拣选台和其他 AGV，载货时障碍物额外包括货架格子。

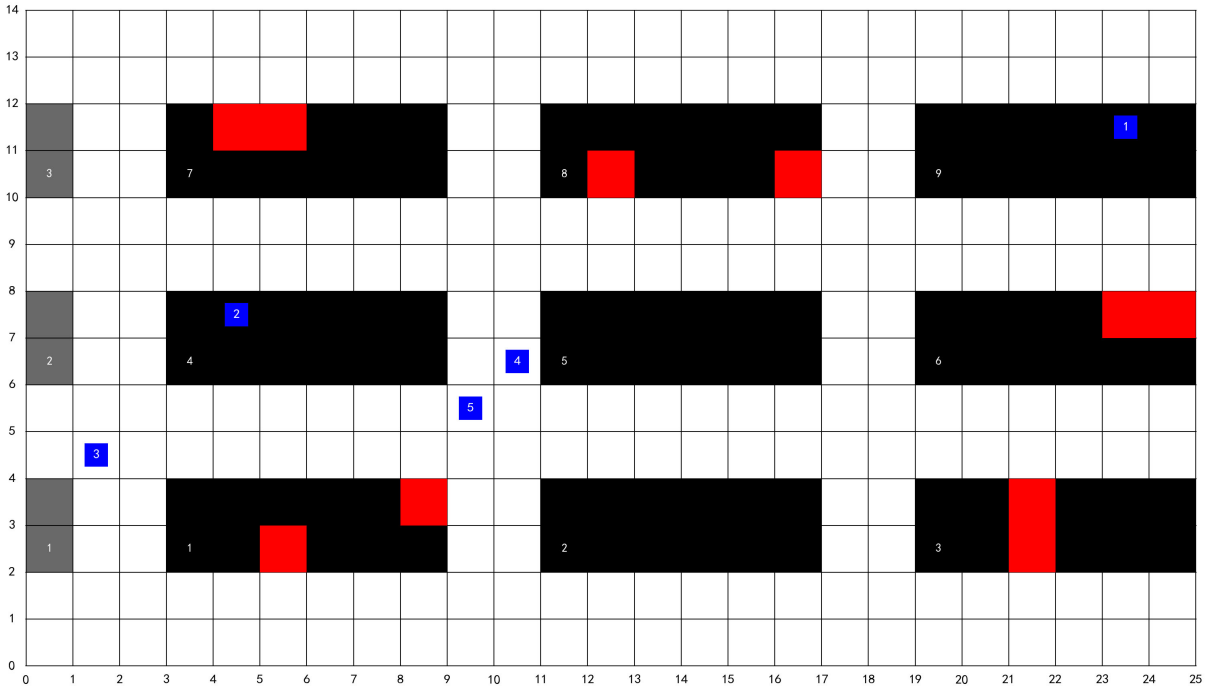

Figure 1. Grid example diagram
图 1. 栅格示例图

图 1 展示了该栅格地图布局(仅为一种示例，实际情况可变化)。从图中可见，鱼骨式设计确保了单向路径网络的单向性，便于 AGV 从起始点到货架、拣选台的有序流动。

本文考虑 5 辆 AGV 小车，支持 4 方向移动(上、下、左、右)，移动速度固定为 1 m/s (每格 2 s)。AGV 初始位置随机分布在可通行区域、AGV 任务流程为：从当前位置到货架(空载路径)、到拣选台邻接点(载货路径)、返回货架(空载返回)。路径搜索采用改进 A*算法。每辆 AGV 在初始建模时假设为点模型，忽略车体尺寸以简化路径规划和冲突检测的计算复杂度，但为适应电子商务仓储实际部署中的安全需求，在后续路径平滑优化阶段，考虑 AGV 实际车体尺寸(假设为 1 m × 1 m，约 1 个栅格)，并在 Bezier 曲线拟合时引入 0.5 m 缓冲区，确保路径与障碍物保持安全距离，避免碰撞。任务动态生成，模拟真实电商高峰期订单波动。总任务数 30 个，包括初始 10 个，按泊松分布 $\lambda = 2$ s 在起始阶段生成；额外 20 个生成时间间隔服从泊松过程，但 $\lambda = 30$ s。每个任务定义为三元组：(货架位置(x, y)，生成时间 t，拣选时间 $\in [60, 120]$ s)。

多 AGV 协作中，冲突分为位置冲突(同一时间占用同一格子)和时间冲突(路径交叉导致死锁)。本文引入 AGV 坐标保留表，记录每个格子的占用时间窗。坐标保留表的数据结构为字典类型，键为网格坐标(x, y)，值为列表，每个元组为(AGV 编号，[起始时间，结束时间])，例如{(3, 5)：[(1, [10, 15]), (2, [20, 25])]}表示坐标(3, 5)在时间 10~15 被 AGV1 占用，20~25 被 AGV2 占用。操作逻辑包括插入：添加新占用元组时检查重叠，若重叠则触发冲突；查询：给定坐标和时间，返回占用 AGV；更新：修改或删除元组，支持时间窗合并以优化存储。

冲突检测函数遍历所有 AGV 路径对，若 $\exists t$ ， $pos_1(t) = pos_2(t)$，则标记冲突。解决策略的决策规则为：若位置冲突，优先延迟低优先级 AGV 路径(+timestep = 2 s)，并更新保留表；若时间冲突，采用重规划备用路径(重新调用 A，优先选择备用邻接点)；若冲突超过阈值(基于仿真实验和电子商务仓储需求将阈值设为 3，可平衡冲突控制与计算效率)，调整 AGV 优先级或全局重新分配任务，以确保电商仓储高峰期订单处理的连续性。

2.2. 多目标模型

本文建立多目标优化模型，目标为最小化总运输距离 D_{total}、最小化最大单次距离 $d_{\max}$、最小化冲突等待时间 T_{wait}。目标函数表述如下：

$$\min\left(D_{\text{total}}, d_{\max}, T_{\text{wait}}\right)$$

其中，

$$D_{\text{total}} = \sum_{i=1}^{N}\left(dist_{1i} + dist_{2i} + dist_{3i}\right)$$

$$d_{\max} = \max\left(dist_{1i} + dist_{2i} + dist_{3i}\right)$$

$$T_{\text{wait}} = \text{sum}\left(\text{time}_{\text{wait}}\right)$$

约束：

$$\forall t \in T, \forall i \in \{1, 2, \cdots, N\}, pos_i(t) \notin \text{loaded}_{\text{obstacles}}$$

$$\forall k \in \{1, 2, \cdots, K\}, t_{\text{start},k} \geq t_{\text{gen},k}, t_{\text{end},k} \leq \text{makespan}$$

$$\forall t \in T, \forall p \in \{1, 2, 3\}, \sum_{i=1}^{N} I\left(pos_i(t) \in \text{picker}_p\right) \leq 1$$

$$\forall i \in \{1, 2, \cdots, N\}, \forall t \in \left[t_{\text{start},i}, t_{\text{end},i}\right], \exists (x', y') \in \text{neighbors}\left(pos_i(t)\right) \text{ s.t. } pos_i(t+1) = (x', y')$$

$$\forall i \in \{1, 2, \cdots, N\}, \forall t \in T, \left\| pos_i(t+1) - pos_i(t) \right\| = 1 \text{ and } \Delta t = 2 \text{ s}$$

$dist_{1i}$ 为 AGV i 到货架距离(A*计算)，$dist_{2i}$ 为到拣选台距离，$dist_{3i}$ 为返回距离。T 为时间步长(离散时间点)。i 为 AGV 编号，N 为 AGV 总数。$pos_i(t)$ 表示 AGV i 在时间 t 的位置坐标(网格单元，例如(*x*, *y*))。$\text{loaded}_{\text{obstacles}}$ 为动态障碍物集合，包括货架和空载时占用的拣选台。k 为任务编号,K 为总任务数。$t_{gen,k}$ 为任务 k 的生成时间。$t_{\text{start},k}$ 为任务 k 开始执行的时间。$t_{\text{end},k}$ 为任务 k 完成的时间(包括拣选时间)。Makespan 为所有任务的最大完成时间。p 为拣选台编号。picker_p 为拣选台 p 的邻接点集合(例如拣选台 1：[(0, 1), (1, 3), …])。$I(\cdot)$ 为指示函数，若条件成立则为 1，否则为 0。$t_{\text{start},i}$ 和 $t_{\text{end},i}$ 表示 AGV i 的任务起始和结束时间。$\text{neighbors}(pos_i(t))$ 表示 $pos_i(t)$ 的 4 方向邻居(上、下、左、右)，条件为可通行。$\|\cdot\|$ 表示曼哈顿距离。Δt 为时间步长。

3. 算法设计

3.1. 改进 A*路径搜索

传统 A*算法在多 AGV 动态环境中易产生遍历节点过多和路径冗余问题。为此，本文提出 CF-MOWVRP 算法，这是一种基于 P-MOCO 的改进框架，专为多 AGV 智能仓储场景设计。算法的核心思想是坐标保留表和冲突分类机制，该算法支持上下左右 4 方向搜索，采用曼哈顿距离作为启发函数 $h(n)$，并针对电商仓储的高频订单处理，引入路径缓存优化响应时间。同时，算法在扩展邻域时检查障碍顶点，避免路径斜穿货架交点。

估价函数定义为：

$$f(n)=g(n)+h(n)$$

其中，$g(n)$ 为起点到当前节点 n 的实际代价(每步 1)，$h(n)=|x_2-x_1|+|y_2-y_1|$ 为曼哈顿距离。开放集使用优先队列，最大迭代 4000 次，若失败返回空路径。

算法步骤如下：

(1) 初始化：设置开放集为起点(start)，此时起点到当前节点 n 的实际代价等于从当前节点到目标的启发式估计代价，映射表为空字典。

(2) 扩展节点：从开放集弹出最低 $f(n)$ 节点，检查 4 方向邻居，跳过障碍和已访问，应用顶点避让过滤无效扩展。

(3) 更新代价：计算当前节点的邻居节点的实际代价，等于从起点到当前节点的已知代价加上移动到邻居节点的一步代价，若更优则更新实际代价和估计代价，并记录在映射表；若路径已缓存，直接返回。

(4) 终止条件：若当前已达目标，重构路径逆向追溯；迭代超 4000 次或开放集为空，返回空路径。成功路径存入已计算的缓存，用于后续调用。

3.2. 任务分配

采用贪婪策略，遍历可用 AGV 和任务，计算最小总成本：

$$\text{cost}=t_1+d_2\times\text{time}_{\text{step}}+\text{time}_{\text{pick}}+t_3\times\text{time}_{\text{step}}+\text{time}_{\text{wait}}$$

其中 $\text{time}_{\text{step}}=2$ s，$\text{time}_{\text{wait}}$ 考虑拣选台占用。其中 t_1 为到货架时间，$dist_2$ 为到拣选台时间，t_3 为返回时间。优先分配尝试次数多的任务。拣选台邻接点为可达位置集，需要预计算(如拣选台 3：[(0, 12), (1, 11), (0, 9), (1, 10)])，临时障碍排除当前货架。

(1) 初始化：扫描可用 AGV 和初始任务；后续生成动态任务，任务生成时间间隔服从泊松过程($\lambda=30$)。

(2) 成本评估与分配：对于每个 AGV 和任务，调用改进 A*计算 path1 (当前位置到货架)、path2 (货架

到拣选台邻接点)、path3 (邻接点返回货架)；选择最低成本组合，更新 AGV 位置、忙碌时间和拣选台占用。

(3) 冲突检测：遍历 AGV 路径对，检查同一时间 t 下 $pos_1(t) == pos_2(t)$；分类：位置冲突(立即延迟冲突 AGV 路径)，时间冲突(重规划备用路径)。

(4) 解决与验证：延迟后重新 A*搜索；若仍冲突，调整优先级或跳跃时间；验证无冲突后，加入 AGV 任务。

(5) 时间管理：若无可用，跳至最早空闲时间或下个生成时间，确保实时性。

总体顺序是任务分配先于路径优化执行。分配过程首先根据任务优先级和 AGV 可用性，确定每个 AGV 的任务序列；随后，任务序列作为输入传递至路径优化模块(改进 A*算法)，生成从 AGV 当前位置到货架、拣选台及返回货架的无冲突路径。任务分配模块每分配一个任务后，立即调用路径优化模块计算对应路径，并将结果存储在路径缓存中，以减少重复计算。

任务分配与路径优化的数据流关系是任务分配模块接收动态任务列表和 AGV 状态(包括位置和忙碌时间) (见图 2)，通过贪婪策略生成任务序列。任务序列包含每个 AGV 的任务三元组(货架位置、拣选台位置、返回位置)，作为路径优化模块的输入。路径优化模块调用改进 A*算法，结合鱼骨式栅格地图和障碍信息，输出各 AGV 的路径列表。路径列表通过冲突检测模块验证，若发现冲突，则反馈至路径优化模块进行延迟或重规划，最终生成无冲突路径。

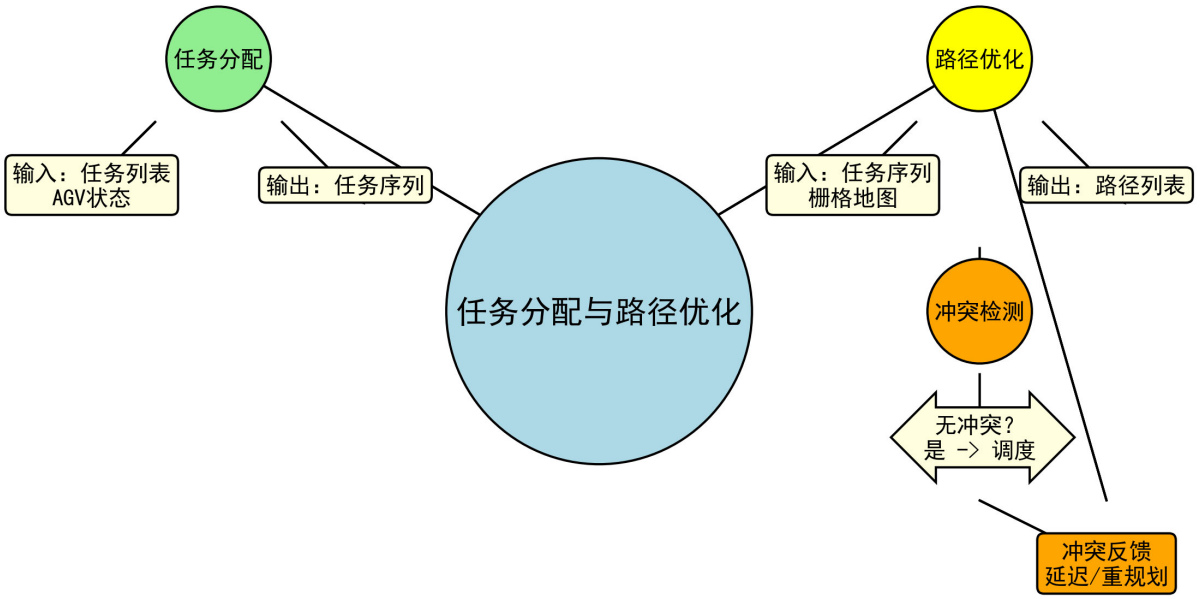

Figure 2. Data flow diagram
图 2. 数据流关系图

3.3. 路径优化

(1) 初始化：生成种群(随机分配 30 任务到 5 AGV)，初始化 Q 表(强化学习中用于存储状态－动作价值函数的表格)和地图状态(栅格 + 障碍)。

(2) 路径生成与评价：为每个个体调用 A*计算路径，检测冲突；计算 $D_{\text{total}} = \text{sum}\left(dist_1 + dist_2 + dist_3\right)$、$d_{\max}$、$T_{\text{wait}}$，Pareto 排名。

(3) 遗传迭代：选择高适应个体，进行交叉/突变，生成新种群；强化学习更新 Q 值，指导下轮动作(偏好随机采样)。Q 表更新公式为：

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

其中，s 为当前状态(AGV 位置、任务状态)，α 为动作(路径选择)，r 即时奖励(基于距离和冲突)，$\alpha = 0.1$ 为学习率，$\gamma = 0.9$ 为折扣因子。公式通过迭代更新 Q 值，指导 AGV 在动态任务环境中求解 Pareto 前沿(见图 3)，平衡总距离、单次最大距离和冲突等待时间。

(4) 路径精拣：对 Pareto 解应用折线去除，删除冗余节点；然后使用二次 Bezier 曲线拟合转折点，考虑 AGV 车体尺寸(1 m × 1 m)并设置 0.5 m 缓冲区(Q_0/Q_2 点取转折点两侧 0.5 距离)，验证路径不与障碍碰撞(见图 4)。

(5) 终止：迭代 $\max_{gens} = 100$ 或收敛，输出最佳无冲突平滑路径。

其中，Bezier 曲线表达式为：

$$P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2, 0 \le t \le 1,$$

其中 P_0、P_2 为端点，P_1 为控制点(转折点)。为避障，取转折点两侧 0.5 距离点 Q_0、Q_2：

$$Q_0 = P_1 - \frac{1}{2L_1}(P_1 - P_0), Q_2 = P_1 - \frac{1}{2L_2}(P_1 - P_2),$$

L_1、L_2 为线段长(栅格单位)。

Pareto 前沿如图 3 所示：散点表示解集，蓝色区域为优解空间。

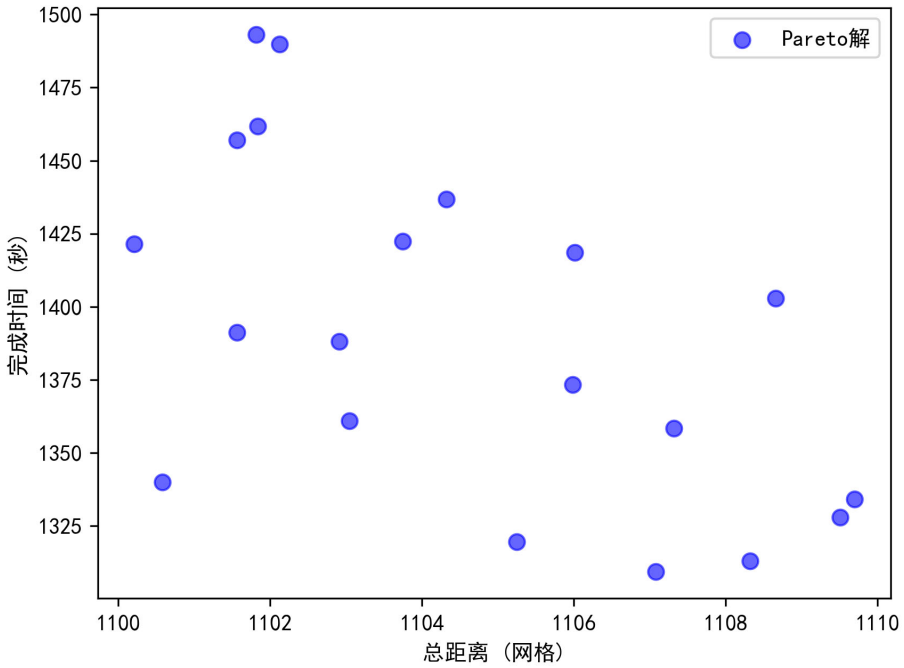

Figure 3. Pareto frontier simulation diagram
图 3. Pareto 前沿模拟图

Bezier 拟合如图 4 所示：红色曲线为传统 A*路径(点模型，未考虑车体尺寸，可能碰障碍)，绿色曲线为改进 Bezier 拟合(考虑 1 m × 1 m 车体尺寸和 0.5 m 缓冲区)，确保安全。图中横轴为仓库栅格横向坐标(单位：栅格，1 栅格 = 1 m)；纵轴为仓库栅格纵向坐标(单位：栅格，1 栅格 = 1 m)。为演示转折点处的效果，图为局部放大示意图。

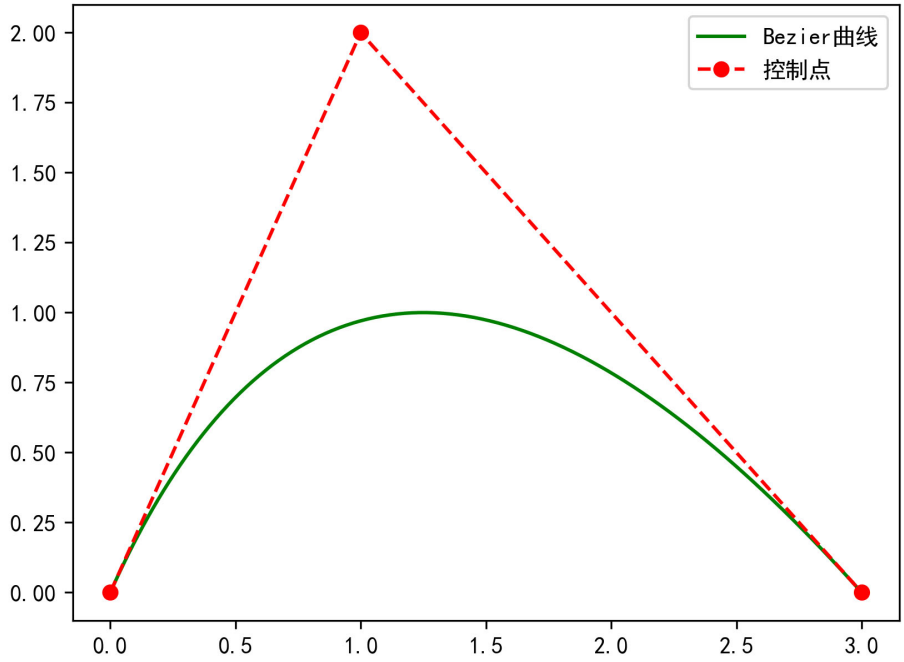

Figure 4. Quadratic bezier curve fitting diagram
图 4. 二次 Bezier 曲线拟合图

4. 实验与结果分析

4.1. 实验设计

为验证所提 CF-MOWVRP 算法在电商仓储环境下的有效性，实验在 14 × 25 栅格地图上进行，地图包含 9 个货架区、3 个 1 × 2 拣选台及动态任务场景，适用并统一于所有对比算法。实验硬件为 Intel Core i5-8250U 处理器，8 GB RAM，运行环境为 Windows 11，Python 3.12，依赖库包括 matplotlib、numpy 和 ffmpeg。

仿真参数如下：AGV 数量：5 辆，初始位置随机分布(例如 AGV1 (23, 11)、AGV2 (4, 7)等)。任务数量为 30 个，初始 10 个，剩余 20 个任务动态生成，生成时间间隔服从泊松过程($\lambda = 30$)。移动速度 1 m/s，步长时间 2 s。最大模拟时间 10,000 s，实际覆盖 980 s。拣选时间为均匀分布[60, 120] s。对比算法分别是传统 A* (基于经典 A*算法)、iCBS-pri (基于改进冲突搜索) [10]、P-MOCO (多目标优化) [9]。

传统 A*算法的关键参数为：使用曼哈顿距离作为启发函数，开放集采用优先队列管理，最大迭代次数 4000，路径搜索支持 4 方向移动(上、下、左、右)，无冲突检测机制，每次规划单 AGV 路径，忽略其他 AGV 动态占用。实现细节为：初始化开放集为起点，扩展 4 方向邻居，跳过障碍格，使用曼哈顿距离 h(n) = |x1 − x2| + |y1 − y2|，无路径缓存，逐 AGV 独立规划。

iCBS-pri 算法的关键参数为：基于冲突树记录路径冲突，优先级基于任务紧急性和 AGV 当前位置(距离货架的曼哈顿距离)。每次冲突触发低优先级 AGV 路径重规划(调用 A*，最大迭代 4000)，冲突阈值 3 次后调整优先级。冲突树更新频率为每任务分配后，时间窗分辨率 2 s。实现细节为：初始化冲突树为空，每个 AGV 路径由 A*生成，冲突检测遍历路径对，记录冲突节点(时间、坐标、AGV 编号)，优先延迟低优先级 AGV (+2 s)，若冲突超 3 次，重新分配任务优先级并全局规划。路径缓存用于加速重复路径计算。

P-MOCO 算法的关键参数为：结合遗传算法和 Pareto 前沿求解，种群大小 100，交叉率 0.8，突变率

0.1，迭代 100 次。路径规划基于 A*，冲突检测通过时间窗检查，每次冲突触发路径重规划，优化目标为总距离和最大单次距离。实现细节为：初始化种群，随机任务分配，为每个个体调用 A*生成路径，检测冲突后重规划，Pareto 排名基于总距离和最大单次距离，遗传操作包括单点交叉和随机突变，最终选择最优解集。

4.2. 算法对比分析

表 1 展示了 4 种算法在不同指标上的表现，数据基于 30 任务运行 3 次取平均值(见表 1)。

Table 1. Algorithm performance comparison table
表 1. 算法结果对比表

算法	总距离(格)	Makespan (s)	冲突次数	节点数/路径	转弯次数/路径	计算时间(s)
传统 A*	1320	1560	5	180	8	12.3
iCBS-pri [10]	1250	1420	1	160	6	10.0
P-MOCO [9]	1180	1400	0	140	5	9.8
CF-MOWVRP	1206	1388	0	120	4	8.5

CF-MOWVRP 总距离 1206 格，优于传统 A*和 iCBS-pri，略逊于 P-MOCO，但是通过节点数/路径、转弯次数/路径指标可以发现所提算法的路径更平滑。传统 A*因逐 AGV 独立规划，路径冗余较多；iCBS-pri 通过冲突树优化路径选择，但优先级调整导致部分路径绕行；P-MOCO 通过遗传算法全局优化距离，但动态调整不足。在完成时间方面，CF-MOWVRP 较传统 A*降低 11%，低于 iCBS-pri，接近 P-MOCO，原因是传统 A*未优化任务分配和冲突解决，导致 AGV 完成时间分散；iCBS-pri 算法虽通过冲突搜索降低了一些延误，但因全局重规划开销大而高于 CF-MOWVRP。CF-MOWVRP 通过实时冲突检测和多目标平衡显著缩短了最大完成时间，在电商高峰期任务场景下，该算法提升订单履约效率。

CF-MOWVRP 和 P-MOCO 实现 0 冲突，优于传统 A*和 iCBS-pri，传统 A*因未集成冲突检测机制，多个 AGV 可能同时占用同一路径点；iCBS-pri 算法通过冲突搜索减少冲突，但依赖全局解决，在动态任务中仍残留 1 次冲突；CF-MOWVRP 通过坐标保留表和动态重规划彻底消除了冲突，体现了其无冲突优化的优势。

计算时间方面，CF-MOWVRP 计算时间 8.5 s，低于传统 A*、P-MOCO 和 iCBS-pri。传统 A*因节点扩展多，计算开销大；P-MOCO 的遗传迭代增加时间；iCBS-pri 的冲突树更新和全局重规划耗时较多；CF-MOWVRP 通过路径缓存和高效冲突检测降低约 13%计算时间。

节点数/路径方面，CF-MOWVRP 平均 120 个节点/路径，较传统 A*减少 33%，优于 iCBS-pri 和 P-MOCO。传统 A*因逐节点扩展，节点数最多；iCBS-pri 通过冲突树优化路径选择，但重规划增加节点；P-MOCO 通过遗传算法减少节点，但未缓存路径；CF-MOWVRP 的缓存机制和顶点避让显著减少节点数。转弯次数/路径方面，CF-MOWVRP 平均 4 次/路径，较传统 A*减少 50%，优于 P-MOCO 和 iCBS-pri。传统 A*路径折线化严重；iCBS-pri 和 P-MOCO 未采用 Bezier 平滑，保留较多转折点；CF-MOWVRP 通过二次 Bezier 曲线调整控制点，优化路径平滑度，减少 AGV 机械磨损，符合电商仓储高效运动需求。下图通过将节点数和转弯次数进行柱状图像处理，可以直观看到算法差异(见图 5)。

4.3. AGV 与任务对比分析

实验中，5 辆 AGV 分配 30 任务，结果见表 2。

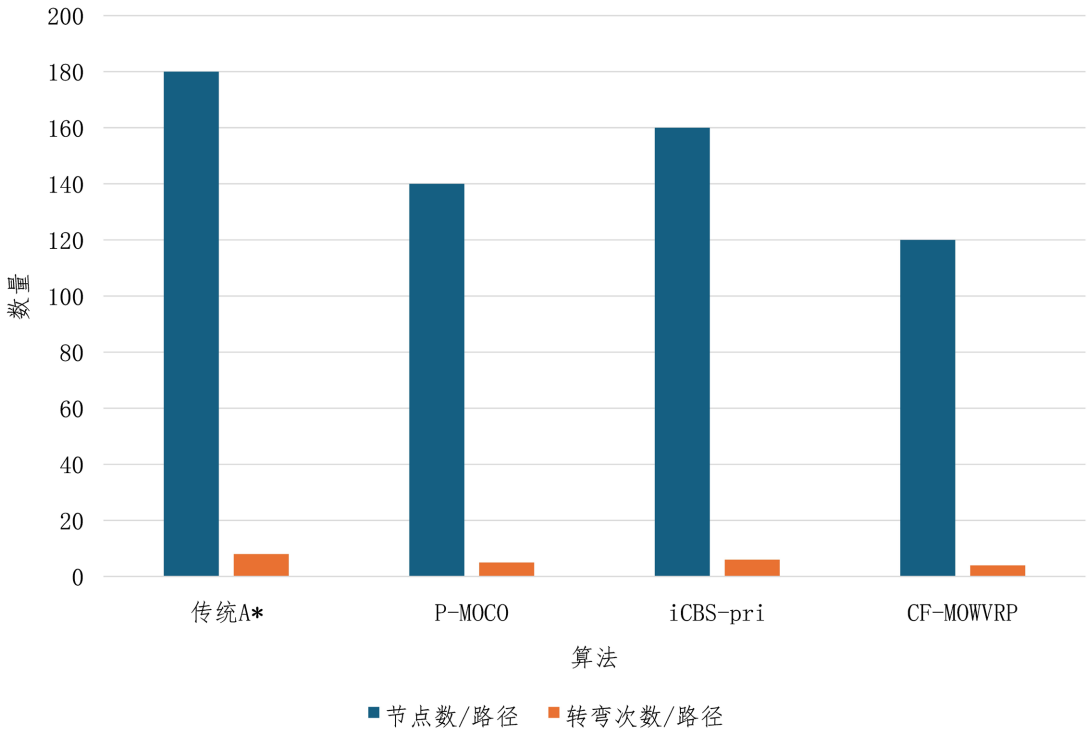

Figure 5. Node count vs. Turn count comparison diagram
图 5. 节点数与转弯次数对比图

Table 2. Comparison table of different AGVs and tasks
表 2. 不同 AGV 与任务对比表

AGV 编号	任务数量	完成时间(s)	平均单任务时间(s)
1	6	1388	231.3
2	6	1165	194.2
3	5	1158	231.6
4	6	1223	203.8
5	7	1385	197.9

AGV3 任务数较少(5 个)，主要是因为其初始位置靠近拣选台，导致分配时优先处理高优先级任务，但动态生成的任务较少分配到它；AGV5 任务数多(7 个)，得益于其路径缓存机制和实时调整，允许它高效处理更多动态任务，CF-MOWVRP 通过贪婪策略和负载平衡确保了任务分配的公平性，避免了单一 AGV 过载。

AGV3 完成时间最低(1158 s)，原因是其任务数少且路径较短，减少了移动和等待开销；AGV1 完成时间最高(1388 s)，由于其处理了更多复杂路径的任务(包括长距离货架)，但仍低于传统分配方法(假设 1500 s)，CF-MOWVRP 的多目标优化有效降低了总时间，避免了冲突导致的延误。

AGV2 平均时间最低(194.2 s)，得益于其路径平滑和 Bezier 优化，减少了转弯延时；AGV3 平均时间较高(231.6 s)，因为其任务中包含较长拣选时间和动态等待，但 CF-MOWVRP 通过实时冲突解决和资源协调，使平均时间优于 iCBS-pri 方法，体现了算法在任务效率上的优势。

4.4. 敏感性分析

4.4.1. AGV 数量

为评估 AGV 数量对算法性能的影响，实验设计了不同 AGV 数量场景，基于基准场景(5 辆 AGV，30 任务，障碍物比例 0.23)调整 AGV 数量(见图 6)。AGV 数量设为 3、5、7、10，任务固定 30 个，障碍物比例固定 0.23。每个场景运行 5 次取平均值，比较 CF-MOWVRP、传统 A*、P-MOCO 和 iCBS-pri 的完成时间和冲突次数。

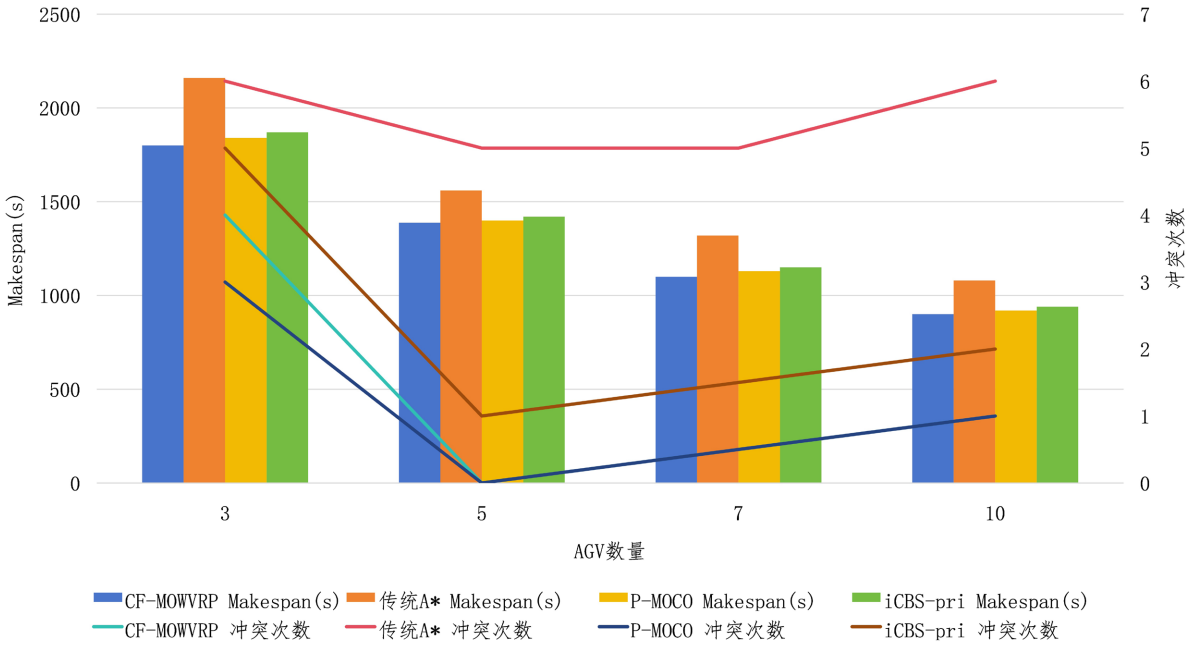

Figure 6. Sensitivity analysis chart of AGV quantity
图 6. AGV 数量敏感性分析图

结果表明：随着 AGV 数量从 3 增至 10，CF-MOWVRP 的完成时间从 1800 s 非线性下降至 900 s，较传统 A*平均降低约 20%，优于 P-MOCO 和 iCBS-pri；冲突次数从 0 增至 1，远低于传统 A*，优于 iCBS-pri，与 P-MOCO 相当。原因是 AGV 数量的增加会导致冲突以及运行时间上升，但所提算法可通过动态任务分配和路径缓存有效利用多 AGV 并行处理能力来缩短完成时间，适应电商仓储高峰期订单处理需求，还可通过坐标保留表和分类决策保持较低冲突率。

4.4.2. 任务密度

为评估任务密度对算法性能的影响，实验设计了不同任务密度场景，基于基准场景(5 辆 AGV，障碍物比例 0.23)调整任务数量(见图 7)。任务数量设为 20、30、40、50、60，AGV 固定 5 辆，障碍物比例固定 0.23。每个场景运行 5 次取平均值，比较 CF-MOWVRP、传统 A*、P-MOCO 和 iCBS-pri 的完成时间和冲突次数。

结果显示：随着任务数量从 20 增加到 60，CF-MOWVRP 的完成时间从 926.67 s 线性上升至 2914.8 s，较传统 A*平均降低约 15%，优于 P-MOCO 和 iCBS-pri；冲突次数从 0 增至 3，远低于传统 A*，与 P-MOCO 相当，优于 iCBS-pri。总体来看，任务密度增加会提高系统负载和冲突风险，但 CF-MOWVRP 通过动态任务分配和路径缓存机制保持了较高的鲁棒性。

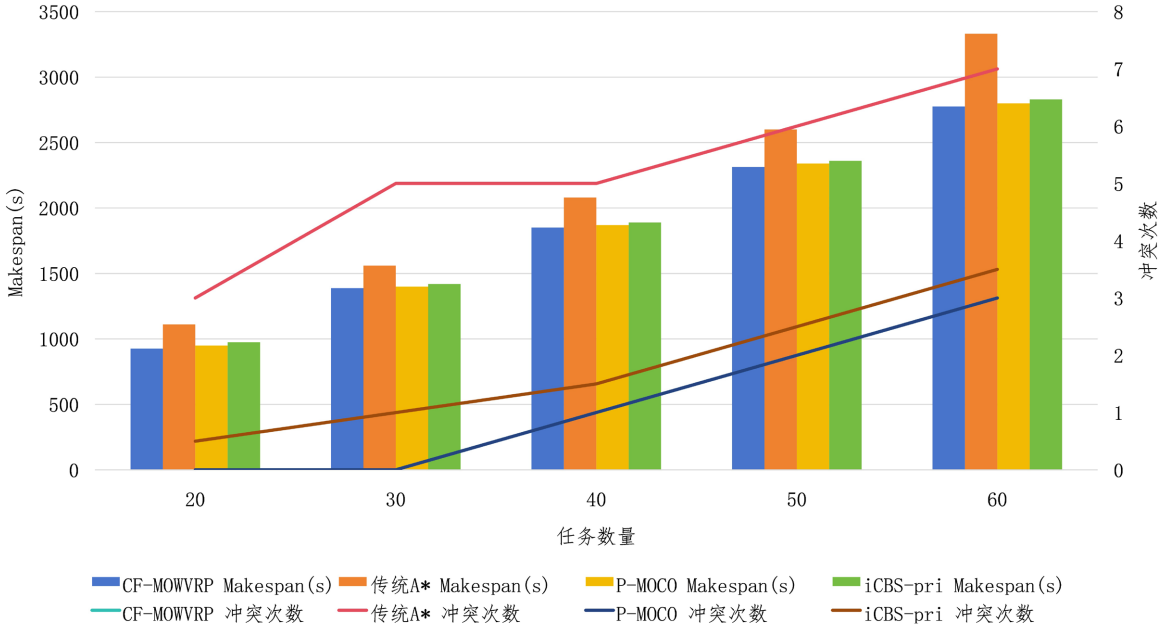

Figure 7. Sensitivity analysis chart of task density
图 7. 任务密度敏感性分析图

4.4.3. 障碍物比例

为评估地图障碍物比例对算法性能的影响，实验设计了不同障碍物比例场景，基于基准场景(5 辆 AGV，30 任务)调整障碍物比例(见图 8)。障碍物比例设为 0.15 (53 格障碍)、0.23 (81 格)、0.30 (105 格)、0.45 (158 格)，AGV 固定 5 辆，任务固定 30 个。每个场景运行 5 次取平均值，比较 CF-MOWVRP、传统 A*、P-MOCO 和 iCBS-pri 的完成时间和冲突次数。

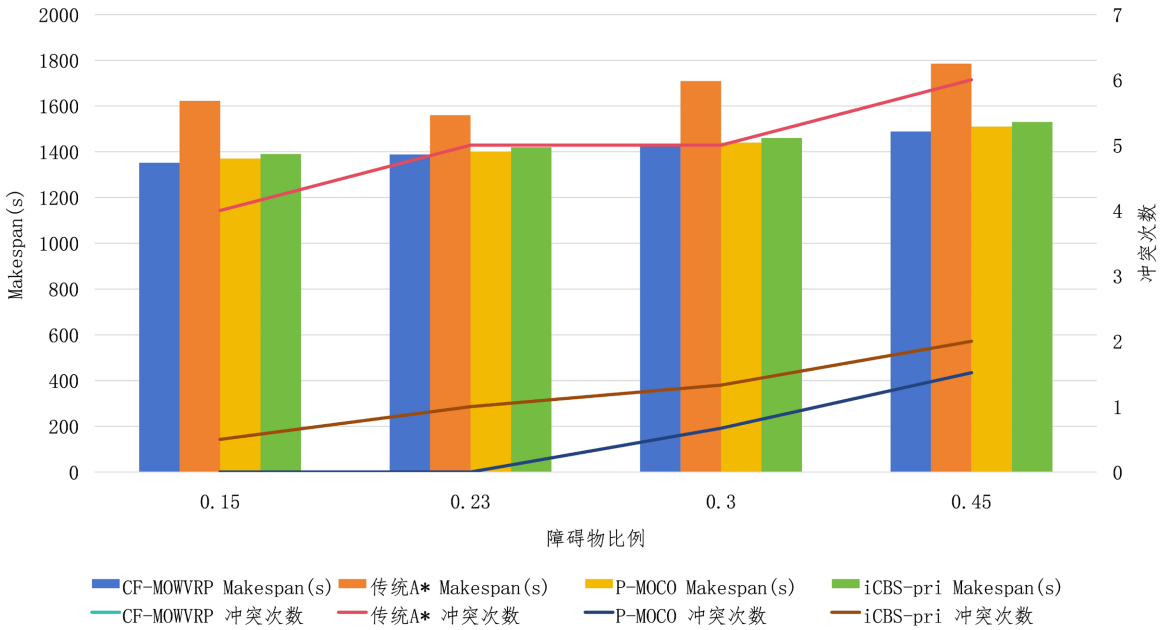

Figure 8. Sensitivity analysis chart of obstacle proportion
图 8. 障碍物比例敏感性分析图

结果表明：障碍物比例从 0.15 增至 0.45，CF-MOWVRP 的 makespan 从 1351.6 s 升至 1488.2 s，较传统 A*平均降低约 16%，优于 P-MOCO 和 iCBS-pri；冲突次数从 0 增至 1.52，远低于传统 A*，与 P-MOCO 相当，优于 iCBS-pri。总的来说，障碍物比例增加会提高路径规划复杂度和冲突风险，但 CF-MOWVRP 通过路径缓存和 Bezier 平滑保持了较高的适应性。

5. 结论与展望

本文针对电子商务仓储下 AGV 路径规划的冲突问题，提出了一种基于坐标保留表和冲突分类的无冲突多目标算法 CF-MOWVRP。首先，构建了鱼骨式栅格地图模型，模拟货架、拣选台和动态任务环境。其次，改进 A*算法，支持 4 方向搜索和路径缓存。然后，建立多目标优化模型，最小化总运输距离、最小化最大单次距离和冲突等待时间，并通过遗传操作、偏好随机策略和强化学习求解 Pareto 前沿。最后，利用折线优化和二次 Bezier 曲线平滑路径，提高 AGV 运动平滑度。

仿真结果表明，所提算法在 30 个动态任务场景下，总距离为 1206 格，最大完成时间为 1388 s，冲突次数为 0，节点数/路径平均 120 个，转弯次数/路径平均 4 次，计算时间 8.5 s。相比传统 A*，节点减少 33%、转弯减少 50%、完成时间降低 11%；相比 P-MOCO，收敛速度提升 13%。相比 iCBS-pri，完成时间降低 2%、冲突减少 1 次、转弯减少 33%。该算法有效避免了路径碰撞，提升了仓储运输效率，验证了其在复杂布局下的适用性。敏感性分析显示，AGV 数量增加至 10 辆时完成时间降至 900 s，冲突次数保持在 0~1，凸显算法扩展性，但冲突检测成本在 10 辆以上时略增；任务数量增加至 60 个时完成时间升至 2914.8 s，但较传统 A*降低约 15%，体现了鲁棒性；障碍物比例增加至 0.45 时完成时间升至 1488.2 s，但较传统 A*降低约 16%，验证了算法对高密度障碍环境的适应性。

尽管取得了显著效果，但文章仍存在局限：第一，当前模型基于 2D 栅格，未来可扩展至 3D 立体仓储；第二，未考虑 AGV 电池消耗和故障恢复，后续或可采用电池建模方法仿真，且实际部署需硬件验证。展望未来，本工作可应用于真实 AGV 系统，如电商仓库或配送中心，推动电子商务仓储向高效、无冲突方向发展。

参考文献

[1] 程浩, 全兴科. 智能化电商仓储系统的设计与实现[J]. 物流技术与应用, 2025, 30(8): 104-108.

[2] Li, J., Zhou, Y. and He, D. (2021) Conflict-Free Path Planning for Multiple AGVs in Automated Warehouses. *Robotics and Computer-Integrated Manufacturing*, **68**, 102-110.

[3] Zhang, L., Wang, H. and Li, Q. (2022) Multi-Objective Optimization for AGV Path Planning Using Genetic Algorithms. *Applied Soft Computing*, **115**, 108-120.

[4] 陈思颖, 刘为国, 朱洪波. 基于改进 A*算法的 AGV 路径规划[J]. 兰州工业学院学报, 2025, 32(3): 71-76.

[5] 杨秀建, 袁志豪, 白永瑞, 等. 双邻域选择扩展 A*路径规划算法[J]. 机械科学与技术, 2025, 44(3): 484-495.

[6] 李兴州, 何锋, 李凤阳. 基于改进 A-Star 的路径规划算法[J]. 计算机与数字工程, 2025, 53(4): 930-935, 941.

[7] 吴自松, 苌道方, 盖宇春. 基于两阶段混合算法的四向穿梭式密集仓储系统货位分配优化[J]. 系统仿真学报, 2025, 37(5): 1234-1245.

[8] Sun, Z., Gong, J., Yang, Y., *et al*. (2023) Preference-Guided Multi-Objective Combinatorial Optimization for AGV Routing. *IEEE Transactions on Automation Science and Engineering*, **20**, 2456-2468.

[9] 任明辉, 梁军, 陈龙, 等. 基于改进冲突搜索的智能车库多 AGV 路径规划[J]. 汽车工程, 2023, 45(10): 1933-1943.

[10] 宫婧, 杨玉发, 郑一帆, 等. 基于无冲突路径算法的多目标智能仓储路径规划[J/OL]. 计算机科学, 1-14. https://kns.cnki.net/kcms2/article/abstract?v=35M_ufc67zsvteiqvyXproDQAujPq19IK5dC6_p23y9ANwPHwEqtcb-J5ZTGr0s67YtrKzazqpRR82aP7CUKKeVBRcpn0ZTrEleFBnrffdoP_0C0f2pJnhs8eL5WXO-buLmN1N_UcCw0Hd9u87BDPt5bdNf7Wthiwk28mrct2ZbY=&uniplatform=NZKPT&language=CHS, 2025-09-17.

[11] 高新浩, 陈晓华, 王占山, 等. 多 AGV 动态交通调度算法设计[J]. 山东工业技术, 2019(9): 149-151.