Journal of System Simulation

Volume 36 | Issue 3 Article 12

3-15-2024

Multi-agent Path Planning with Obstacle Penalty Factor

Xingyu Yan
Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China, yyanxingyu@126.com

Dayan Li
Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China, kglidy@qq.com

Niya Wang
Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China

Kaixiang Zhang
Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming 650500, China

See next page for additional authors

Multi-agent Path Planning with Obstacle Penalty Factor

Abstract

Abstract: In light load environments, complex obstacle areas will exacerbate local conflicts between agents, leading to a decrease in path solving efficiency. This paper proposes a multi-agent path planning (MAPF) method with obstacle penalty factors in light load environments. First, in the low-level single machine planning process based on the conflict-based search (CBS) algorithm framework, by judging the distribution type of surrounding obstacles that are about to expand the agent's position, corresponding obstacle penalty factors are assigned to them; then, the penalty factors in the path planning process are accumulated and used as the heuristic value of single machine planning to select the path; finally, combined with the upper-level conflict resolution strategy of the CBS algorithm framework, MAPF and conflict coordination are performed. The results show that in a light load environment with a 10% obstacle distribution, the proposed algorithm has a solving time of about 81.38%~83.67% of that of the CBS algorithm, and the expansion amount of the constraint tree (CT) is 60.14%~71.66% of that of the CBS algorithm. Simulation in Gazebo has shown that this method can reduce the number of passes through complex obstacle areas.

Keywords

light load environment, multi-agent path finding (MAPF), penalty factor, conflict-based search (CBS), constraint tree (CT)

Authors

Xingyu Yan, Dayan Li, Niya Wang, Kaixiang Zhang, and Jianlin Mao

Recommended Citation

Yan Xingyu, Li Dayan, Wang Niya, et al. Multi-agent Path Planning with Obstacle Penalty Factor[J]. Journal of System Simulation, 2024, 36(3): 673-685.

第 36 卷第 3 期 系统仿真学报© Vol. 36 No. 3
2024 年 3 月 Journal of System Simulation Mar. 2024

带障碍物惩罚因子的多机器人路径规划

闫星宇[1]，李大焱[1*]，王妮娅[1]，张凯翔[2]，毛剑琳[1]
(1. 昆明理工大学 信息工程与自动化学院，云南 昆明 650500；2. 昆明理工大学 机电工程学院，云南 昆明 650500)

摘要：轻载环境中，复杂障碍物区域将引起机器人之间局部冲突加剧，进而导致路径求解效率下降，针对该问题，提出轻载环境下带障碍物惩罚因子的多机器人路径规划方法。*在基于冲突搜索(conflict-based search, CBS)算法框架的下层单机规划过程中，通过对即将拓展机器人位置的周围障碍物分布类型进行判断，赋予与之对应的障碍物惩罚因子；对路径规划过程中的惩罚因子进行累加，作为单机规划的启发值对路径进行选取；结合CBS算法框架的上层冲突消解策略进行多机器人的路径规划与冲突协调。*测试结果表明，在10%障碍物分布的轻载环境中，所提算法的求解时间约为CBS算法的81.38%~83.67%，二叉约束树(constraint tree, CT)拓展量为CBS算法的60.14%~71.66%。在Gazebo中仿真表明，所提方法可减小通过复杂障碍物区域的次数。

关键词：轻载环境；多机器人路径规划；惩罚因子；基于冲突搜索算法；约束树
中图分类号：TP242；TP18　　文献标志码：A　　文章编号：1004-731X(2024)03-0673-13
DOI: 10.16182/j.issn1004731x.joss.23-0397
引用格式: 闫星宇, 李大焱, 王妮娅, 等. 带障碍物惩罚因子的多机器人路径规划[J]. 系统仿真学报, 2024, 36(3): 673-685.
Reference format: Yan Xingyu, Li Dayan, Wang Niya, et al. Multi-agent Path Planning with Obstacle Penalty Factor[J]. Journal of System Simulation, 2024, 36(3): 673-685.

Multi-agent Path Planning with Obstacle Penalty Factor

Yan Xingyu[1], *Li Dayan*[1*], *Wang Niya*[1], *Zhang Kaixiang*[2], *Mao Jianlin*[1]
(1. Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China;
2. Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming 650500, China)

Abstract: In light load environments, complex obstacle areas will exacerbate local conflicts between agents, leading to a decrease in path solving efficiency. This paper proposes a multi-agent path planning (MAPF) method with obstacle penalty factors in light load environments. *First, in the low-level single machine planning process based on the conflict-based search (CBS) algorithm framework, by judging the distribution type of surrounding obstacles that are about to expand the agent's position, corresponding obstacle penalty factors are assigned to them; then, the penalty factors in the path planning process are accumulated and used as the heuristic value of single machine planning to select the path; finally, combined with the upper-level conflict resolution strategy of the CBS algorithm framework, MAPF and conflict coordination are performed.* The results show that in a light load environment with a 10% obstacle distribution, the proposed algorithm has a solving time of about 81.38%~83.67% of that of the CBS algorithm, and the expansion amount of the constraint tree (CT) is 60.14%~71.66% of that of the CBS algorithm. Simulation in Gazebo has shown that this method can reduce the number of passes

收稿日期：2023-04-07　　修回日期：2023-05-31
基金项目：国家自然科学基金(62263017)；云南省重大科技专项计划(202002AC080001)
第一作者：闫星宇(1996-)，男，硕士生，研究方向为移动机器人路径规划。Email：yyanxingyu@126.com
通讯作者：李大焱(1981-)，男，实验师，硕士，研究方向为移动机器人设计及路径规划等。Email：kglidy@qq.com

through complex obstacle areas.

Keywords: light load environment; multi-agent path finding (MAPF); penalty factor; conflict-based search (CBS); constraint tree (CT)

0 引言

多机器人路径规划(multi-agent path finding, MAPF)问题是人工智能领域中一个重要问题[1]，在给定机器人的任务状态以及环境因素下，计算多机器人的无碰撞路径[2]。尽管传统算法可为每个独立机器人提供最优路径，但当处理多机器人问题的时候，它们搜索的是所有机器人的复合空间，因此计算量将呈指数增加[3]。由于其NP-hard问题的复杂性，在多机器人系统中生成最佳无碰撞路径仍然具有挑战性[4]。MAPF具有较强的实用性，实际应用包括计算机游戏、交通管理、机场调度和仓库自动化等[5]。

A*算法常常被应用于单机器人路径求解问题中[6]，将其应用于MAPF问题时，A*搜索的状态空间随着地图中机器人数量的增加而呈指数增长[7]，由于解空间规模的激增，成功率将大幅降低[8]。文献[9]提出(safe interval path planning, SIPP)算法，提出安全间隔的概念，来减少需要搜索的状态数量。

在MAPF算法方面，已经取得了多个代表性成果[10]。文献[11]提出的基于冲突搜索(conflict-based search, CBS)算法构造了两层的结构，下层通过A*算法为每个单机器人搜索路径；上层则对每个单机器人的路径进行冲突判断，当存在冲突时，则对冲突机器人添加约束条件以避免冲突的产生[12]。由于CBS算法具备路径最优性等特点，因此逐渐成为解决MAPF问题的重要算法[13]。文献[14]提出不相交分割的CBS(CBS with disjiont splitting, CBS-DS)算法，对机器人间的冲突进行剪枝，从而避免因冲突重复问题引发的大量重复工作。文献[15]提出改进引导的CBS(explanation guided CBS, XG-CBS)算法，在约束树中引入分段冲突，从而使这些冲突可以通过具体化的约束来解决。文献[16]提出基于MDD(multi-valu decision diagram)的矩形推理CBS(CBS with rectangle reasoning by MDDs, RM-CBS)算法，来解决对称空间的机器人冲突约束问题。文献[17]引入协作CBS(cooperative CBS, Co-CBS)算法，在多机协作框架的基础上集成CBS算法，实现了多个机器人的协作任务分配及其路径规划方案。

通过上述分析可以看出，复杂障碍物区域会加剧机器人之间的局部冲突。CBS算法和CBS-DS算法都是在出现冲突后，再对其进行冲突消解，而没有考虑在单机规划阶段通过路径选择，来避免局部冲突的产生。在应用场景中，当机器人频繁通过复杂障碍物区域时，也更容易发生剐蹭情况。本文在CBS算法的单机规划阶段，通过对拓展位置周围的障碍物分布类型进行判断，并赋予对应的惩罚因子，通过选取惩罚因子总和最小的路径，可减小CBS算法的冲突消解频率，提升路径求解效率。

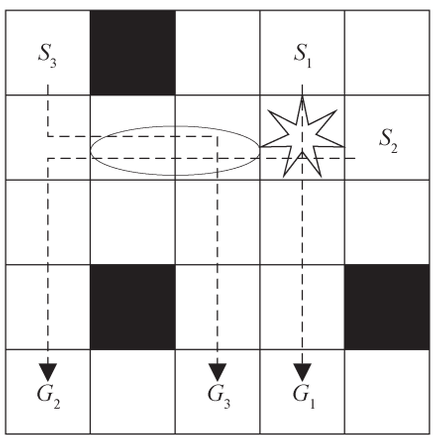
1 问题描述

MAPF问题的目标是为地图中的每个机器人都找到一条相互无冲突路径，使每个机器人从初始位置到达目标位置。本文旨在解决轻载环境中的MAPF问题。轻载环境中存在一定数量的复杂障碍物区域，且障碍物面积不超过地图总面积的10%，例如无人工厂的物料车间。这些障碍物以对称结构为基础分布，障碍物之间的空间资源有限，仅允许一个机器人通过，从而容易引发局部冲突问题。

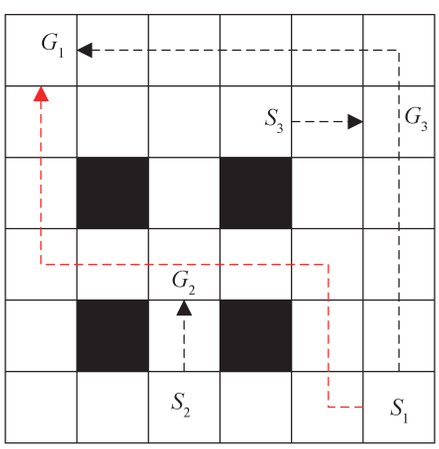
令S为起始点，G为目标点，如图1所示，时间被离散化为时间步，在每个时间步中，每个机器人既可以移动到相邻的位置，也可以在当前位置等待。移动和等待行为都消耗一个时间步，除非机器人在其目标位置等待，其时间成本为0。

DOI: 10.16182/j.issn1004731x.joss.23-0397

MAPF 为机器人群体找到的一组无冲突路径，使所有机器人从它们的起始点移动到它们的目标点，同时最小化路径总成本。



(a) 路径冲突图



(b) 复杂障碍物区域下的路径规划

图 1　多机器人路径规划
Fig. 1　MAPF

冲突类型分为两种，顶点冲突和边缘冲突。顶点冲突是指 2 个机器人在时间步 t 同时占据同一位置，如图 1(a)所示，1 号机器人和 2 号机器人在 step 1 同时占据栅格点 v=(4,4)而发生顶点冲突。边缘冲突是指 2 个机器人在时间步 t 和 t+1 之间反向遍历同一条边，如图 1(a)所示，2 号机器人将在 step 2 从栅格点 v=(3,4)移动到栅格点 v=(2,4)，3 号机器人将在 step 2 从栅格点 v=(2,4)移动到栅格点 v=(3,4)，2 个机器人在 step 2~3 之间发生边缘冲突。

图 1(b)展示了在复杂障碍物区域中的路径规划。如果选择红色路径，机器人将通过复杂障碍物区域。该区域的空间资源较少，一旦机器人之间发生冲突，需要花费较长时间来解决。此外，由于此区域障碍物数量较多，如果选取红色路径，带运动学约束的机器人可能会与障碍物发生剐蹭，故选择较宽阔的区域通过可以减少机器人性能引起的剐蹭现象。

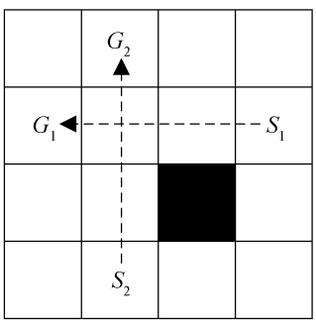
2　CBS 算法

CBS 算法是一种用于解决 MAPF 问题的两层算法。在下层，它执行最佳搜索为每个机器人在静态障碍物环境中找到一条最短路径。在上层，它对二叉树 CT 执行最佳优先搜索，判断机器人的路径之间是否存在冲突，通过对冲突机器人添加约束从而避免冲突的发生。

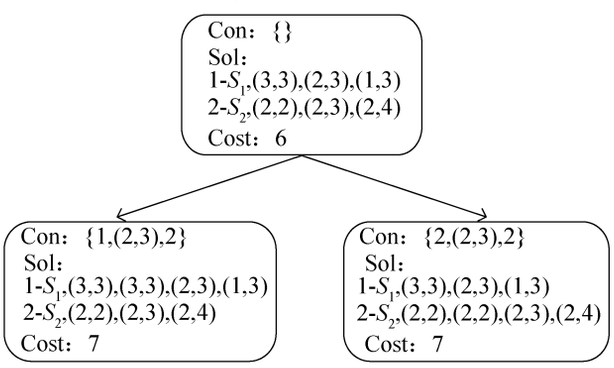
每个 CT 节点 N 包含一组用于避免冲突的约束条件(a_i,a_j,v ,t)，以及一组满足这些约束条件的路径，CT 节点 N 的数量可以用来表示机器人间冲突的程度。其中，N 的成本为所有机器人路径的成本总和。CBS 算法对所有 CT 节点依次进行冲突检查，并调用 CBS 算法框架的下层 A*算法，在满足约束条件的前提下，重新规划出一条路径。当 CT 节点无冲突时，CBS 算法得到一个最优解。如果存在冲突，将对冲突机器人添加约束条件，再次调用 A*算法为冲突机器人搜索满足约束条件的路径。同时，在其他路径保持不变的情况下进行路径刷新，通过 CBS 算法框架的上层进行冲突判断，直到路径之间无冲突产生，以达到冲突消解的作用。如图 2 所示。

如图 2(a)所示，CBS 算法框架的下层 A*算法分别为 1 号机器人和 2 号机器人找到了一条最优路径，但在 step 2 时，1 号机器人和 2 号机器人将在 v=(2,3)发生顶点冲突。CBS 算法框架的上层通过将 CT 根节点 N 分裂成 2 个子节点并添加约束条件，用来避免机器人间发生冲突。如图 2(b)所示，在有约束条件的前提下，2 个机器人调用 A*算法找到了无冲突的最短路径，并将求解结果分别存

储到CT根节点N分裂后的2个子节点中。同时，由于添加了新的约束条件，CT节点N的总成本为7，是无冲突路径中代价值最小的，由此得到了多机器人路径的最优解。



(a) 路径冲突图



(b) 二叉树CT图

图2 CBS示例
Fig. 2 CBS example

3 带障碍物惩罚因子的CBS算法

3.1 标准A*算法

标准A*算法在Dijkstra算法基础上结合了Breadth-First搜索算法路径短的优势。既考虑了从初始位置到当前位置的代价，又考虑了从当前位置到目标位置的代价，从而实现了搜索最短路径的同时提高了搜索效率，其启发式为

$$f(n)=g(n)+h(n) \tag{1}$$

式中：$f(n)$为当前节点的总代价；$g(n)$为从初始位置到当前位置的代价；$h(n)$为从当前位置到目标位置的代价。在计算$h(n)$时，本文采用曼哈顿距离作为评判标准为

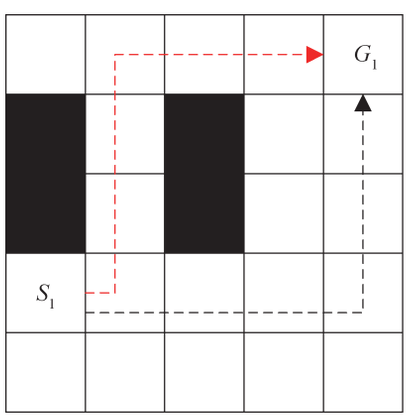
$$h(n)=|x_{\text{curr}}-x_{\text{goal}}|+|y_{\text{curr}}-y_{\text{goal}}| \tag{2}$$

式中：x_{curr}和y_{curr}分别为当前位置的横纵坐标；x_{goal}和y_{goal}分别为目标位置的横纵坐标；$h(n)$为依据曼哈顿距离的代价值。

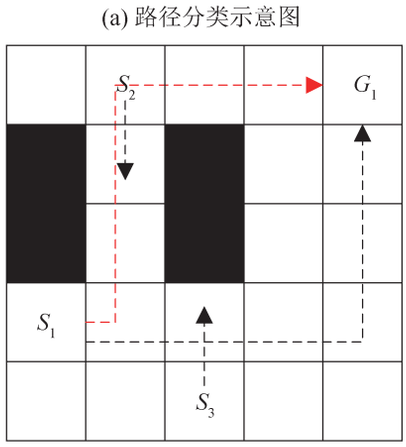
3.2 带障碍物惩罚因子的加权

在轻载环境中，路径的不同选择可能会影响后续机器人之间的冲突消解难度。如图3(a)所示，在同等代价值的条件下，使用A*算法规划出2条最优路径到达目标位置。

当求解MAPF问题时，需要考虑到不同的障碍物分布类型对机器人间冲突消解造成的负面影响。如图3(b)所示，当遇到冲突时，选取黑色路径相较于红色路径具有更宽阔的可协调空间，如1号机器人和2号机器人将在资源有限的通道形成互锁问题，增加冲突消解难度。



(a) 路径分类示意图



(b) 互锁问题示意图

图3 路径选取示意图
Fig. 3 Path selection

DOI: 10.16182/j.issn1004731x.joss.23-0397

CBS 算法在求解对称互锁问题时，由于空间资源有限，没有足够的空间资源可为冲突机器人的调节提供更多的可选取方案。随着周围机器人的增加，将面临解决完一个冲突，又需要解决新的冲突问题，从而增加消解难度，降低求解效率。选取黑色路径相较于红色路径将具有更宽阔的可协调空间，可减小对称互锁问题出现的概率，从而降低局部冲突程度，以此来提高CBS 算法的求解效率。

基于CBS 算法框架，本文提出一种改进的A* 算法，来解决轻载环境中可能出现的局部冲突问题。本文算法引入带障碍物惩罚因子(obstacles boycatt select canon, OBSC)的加权方法，更全面地考虑了不同障碍物分布类型对机器人之间冲突消解的负面影响。在A*算法求解过程中，对下一个拓展位置周围障碍物分布类型进行判断，并赋予与之对应的惩罚因子，以此作为A*算法的启发式来选取通过复杂障碍物区域次数最少的路径。

如图4 所示，以向上移动为例，判断以拓展位置为中心向上的5 个相邻区域障碍物的分布情况，并依据障碍物分布类型赋予对应的惩罚因子。使用$\alpha(\alpha\in(1,1.5))$作为惩罚因子，依据障碍物复杂程度预先设置不同的系数。将文中的类型1~3 的α值分别赋予1、1.25、1.5。为了适配多样化的扩展需求，障碍物的分布模式可以进行灵活调整。具体而言，障碍物布局不仅可以进行左右90°的旋转变换，还可以进行水平对称或垂直对称操作，以及在完成对称之后进一步执行90°的旋转。

累计路径规划过程中惩罚因子的总和，用于路径的选取。其中，惩罚因子累积总和不应大于1 个时间步，来保证不会影响机器人路径最优性，惩罚因子的统计公式为

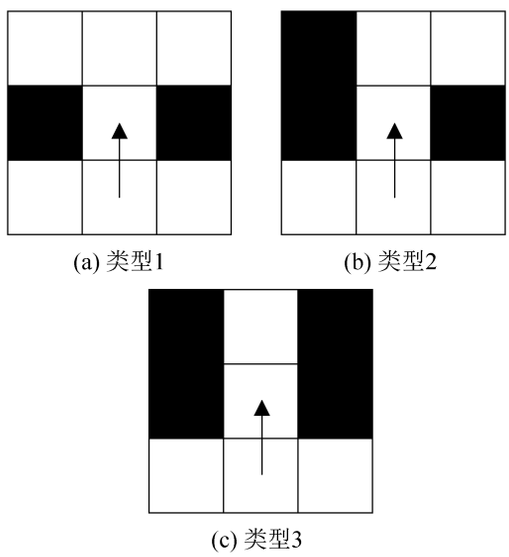
$$\begin{cases} p(n)=p(n_{\text{parent}})+\alpha \\ K\times p(n)<1 \end{cases} \tag{3}$$

式中：$p(n)$为当前节点n 累计的惩罚因子总和；n_{parent}为节点n 的父节点；α为当前节点n 的惩罚因子；K为调节系数，用于保证$p(n)$的数值不高于1 个时间步，从而使路径最优。

CBS 算法框架的下层A*算法分别为每个机器人规划最优路径，并在路径规划过程中通过判断障碍物分布类型，赋予对应的惩罚因子，通过累计惩罚因子的总和，作为A*算法的启发式来参与路径选取。

通过选取更优的路径，减小机器人通过复杂障碍物区域的次数，降低对称互锁问题出现的频率，从而削弱局部冲突程度，启发式为

$$f(n)=g(n)+h(n)+K\times p(n) \tag{4}$$



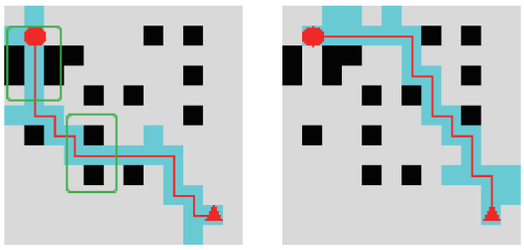
(a) 类型1 (b) 类型2

(c) 类型3

图4 障碍物类型
Fig. 4 Types of obstacles

3.3 带障碍物惩罚因子的A*(A*-OBSC)算法

如图5 所示，蓝色阴影部分为单机规划过程中拓展的节点，红色路径为最终路径。可以看出，带障碍物惩罚因子的A*算法(A*-OBSC)相较于A*算法规划的路径，可减少通过复杂障碍物区域次数。通过引入带障碍物惩罚因子，可以减少MAPF 过程中可能发生的对称互锁问题。在图5(a)中绿色框线所示的区域有很大可能会发生多机器人的对称互锁问题。图5(b)是采用有障碍物惩罚因子的算法时的路径。通过合理地选取路径，机器人可获得更多的可协调空间，以此降低二叉树CT 拓展量，进而提升CBS 算法的求解效率。



(a) A*路径 (b) A*-OBSC路径

图5 带障碍物的惩罚因子路径示意图对比
Fig. 5 Comparison of penalty factor paths with obstacles

3.4 CBS-OBSC 算法流程

算法 1 CBS-OBSC 算法

step 1：为每个机器人赋予起始位置和目标位置，并给出地图的障碍物分布情况；

step 2：调用 CBS 算法框架的下层 A*算法，为每个单机器人规划静态障碍物环境中的最优路径；

step 3：在 A*算法拓展下一个机器人位置时，依据式(3)判断即将被拓展位置的惩罚因子，并与之前的惩罚因子总和累加；

step 4：依据式(4)得出下一个位置的总代价；

step 5：选取总代价最小的节点进行拓展；

step 6：等待 CBS 算法框架的下层 A*算法为每个机器人求解路径后，使用 CBS 算法框架的上层对机器人之间的路径进行冲突判断。当存在冲突时，对冲突机器人添加约束条件，当无冲突时，则跳转到 step 8；

step 7：为冲突机器人添加 step 6 的约束条件，跳转到 step 2，重新为冲突机器人寻找最优路径；

step 8：搜索结束，CBS 算法得到一组机器人间相互无碰撞的最优解。

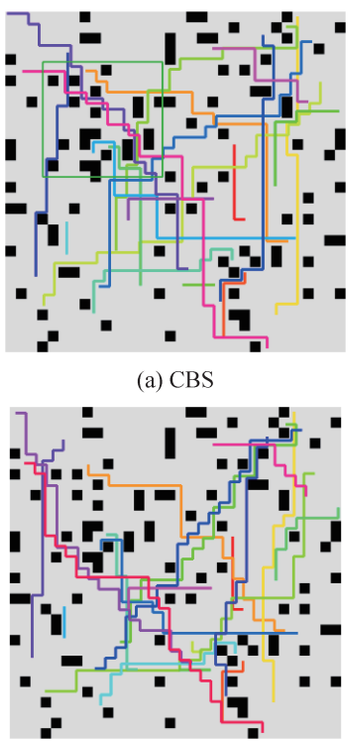
4 仿真与分析

4.1 仿真环境

所有算法均在 Inter core i7-11700KF 3.6 GHz，内存为 32 GB 的 PC 机上以 C++程序运行。算法求解时间限制为 10 min，所有算法需要在规定时间内完成对公共算例的求解，超出规定时间则被认为求解失败，求解失败的时间按照上限 10 min 统计。

4.2 固定任务的路径规划求解结果

为验证本文所提 CBS-OBSC 算法相较于 CBS[11]算法的路径优化，采用标准算例集[18]中的典型地图进行测试。图 6 所示展示了路径对比结果，其中地图大小为 32×32、障碍物占比为 10%。



(a) CBS

(b) CBS-OBSC

图6 20 个机器人路径对比
Fig. 6 Comparison of 20 agent paths

CBS-OBSC 算法相较于 CBS 算法规划的路径有一定优化，代价值相同的情况下，将优先拓展通过空间资源更多的路径。如图 6(a)左上角绿色框线所示，优化可减少局部冲突程度。

表 1 展示了不同算法的性能比较结果。由于 CBS-OBSC 算法优先选取空间资源充裕的通道，可减少因空间资源有限而导致局部冲突加剧的问题，CBS 算法通过不断分裂 CT 节点 N 来添加约束进行冲突消解。其中，CBS-OBSC 算法相较于 CBS 算法，减小了 50% 的二叉树 CT 拓展量。CBS

DOI: 10.16182/j.issn1004731x.joss.23-0397

算法的时间复杂度为$\left[1+\frac{2(\eta-1)}{N_{\text{robot}}}\right]\cdot\sum_{i=1}^{N_{\text{robot}}}O(b)^{T_i}$，($\eta$为上层节点数，即二叉树CT拓展量)。CBS-OBSC算法通过对路径选取的优化来减少η的数量，从而降低时间复杂度，提升路径求解效率。

表1　路径求解性能指标
Table 1　Path solving performance indicators

算法	时间/s	二叉树CT	A*拓展节点
CBS	0.250	4	17 712
CBS-OBSC	0.217	2	12 336

4.3 轻载环境中不同数量机器人随机任务的路径求解结果

为验证所提算法在轻载环境中的求解能力，扩大实验规模，如图7所示，采用标准算例集中的典型地图，地图大小为64×64，障碍物占比10%，且存在部分复杂障碍物区域。

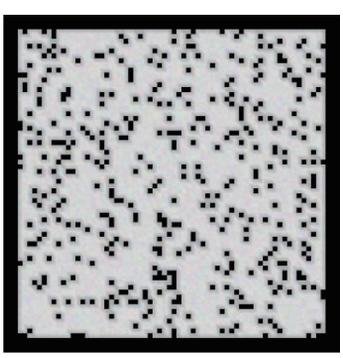


图7　地图类型
Fig. 7　Map type

分别采用CBS[11]、CBS-DS[14]、CBS-SIPP[9]、CBS-OBSC和CBS-DS-OBSC进行实验对比。本文设置了8组不同数量机器人，每组20次不同起始位置和目标位置进行路径求解。成功率和求解时间如图8所示。其中，CBS-SIPP与CBS-DS、CBS-OBSC与CBS-DS-OBSC成功率一致。

在机器人数量较少的情况下，几种算法都有较高的求解成功率，但是随着机器人数量的增加求解成功率均呈现一定程度的下降。所提的CBS-OBSC算法，相较于其他算法成功率和求解时间均有一定的提升。

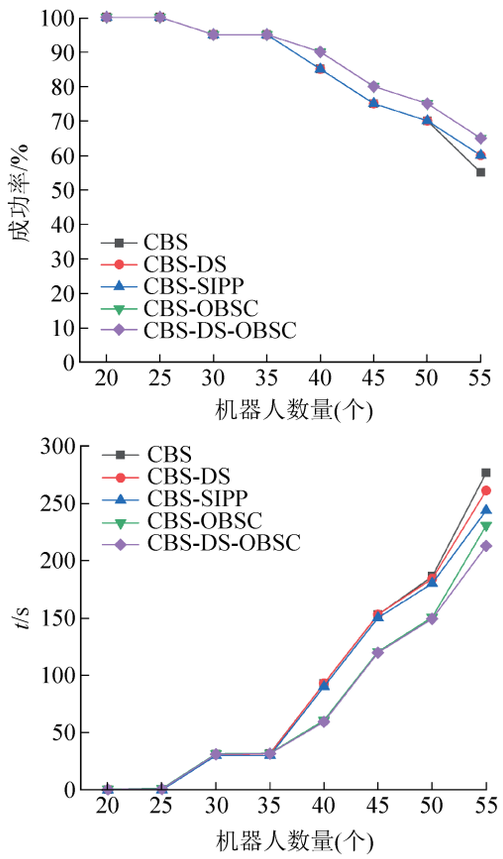


图8　轻载环境中路径求解结果
Fig. 8　Path solving results in light load environment

表2所示为不同算法平均总路径长度，表3所示为不同算法的二叉树CT拓展量。表2~3中，数据均采用可公共求解的算例进行统计，公共算例数为12个，成功率较低的不纳入统计，用—来表示。表2、3可以看出，CBS-OBSC算法为机器人群体求解的路径不会增加代价值；CBS-OBSC算法在求解机器人数量较少的场合时，由于路径的选择不同，可能会少量增加二叉树CT拓展量，但是对求解效率影响不大。当本文算法在求解机器人数量较多的场合时，相较于CBS算法可以有效降低二叉树CT拓展量，且相较于CBS-DS算法和CBS-SIPP算法也有一定优势，通过降低二叉树CT拓展量以此来提高求解效率。CBS-OBSC算法相较于CBS算法的二叉树CT拓展量降低39.86%，求解时间降低18.62%。CBS-DS-OBSC算法相较于CBS-DS算法的二叉树CT拓展量降低46.61%，求解时间降低19.75%。

表2 不同算法平均总路径长度
Table 2 Average total path length of different algorithms

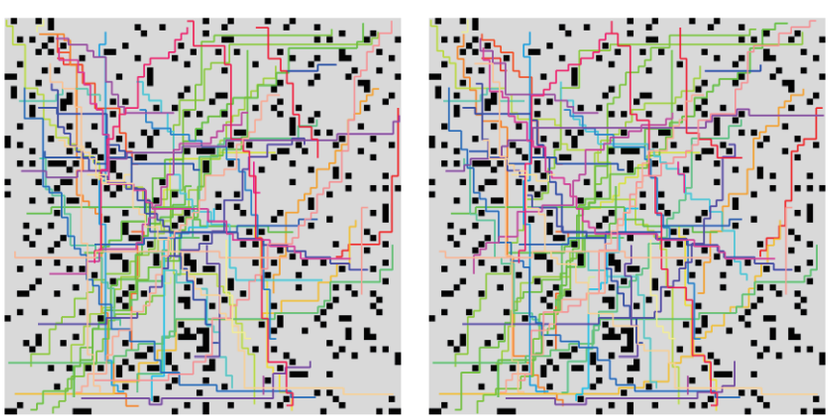
机器人数量	算法类型				
	CBS	CBS-DS	CBS-SIPP	CBS-OBSC	CBS-DS-OBSC
20	1 005.583	1 005.583	1 005.583	1 005.583	1 005.583
25	1 245.833	1 245.833	1 245.833	**1 245.750**	**1 245.750**
30	1 514.083	1 514.083	**1 513.583**	1 514.000	1 514.000
35	1 768.750	1 768.750	1 768.250	**1 768.667**	**1 768.667**
40	1 983.417	1 983.417	**1 982.917**	1 983.833	1 983.833
45	2 220.833	2 220.833	2 220.833	**2 220.750**	**2 220.750**
50	2 465.250	2 465.250	2 465.250	**2 465.167**	**2 465.167**
55	—	2 735.417	2 735.417	**2 735.333**	**2 735.333**

表3 不同算法的二叉树CT拓展量
Table 3 CT extension of binary tree of different algorithms

机器人数量	算法类型				
	CBS	CBS-DS	CBS-SIPP	CBS-OBSC	CBS-DS-OBSC
20	**0.750**	**0.750**	0.833	**0.750**	**0.750**
25	**1.583**	**1.583**	1.667	2.000	2.000
30	**2.417**	**2.417**	2.750	3.000	3.000
35	**3.333**	**3.333**	3.667	4.250	4.167
40	8.033	**6.750**	**6.750**	9.000	9.083
45	10.083	9.000	8.083	**7.917**	8.000
50	45.167	26.583	19.500	16.000	**12.833**
55	—	219.167	386.167	111.083	**35.500**

CBS-OBSC 算法在单机规划阶段，考虑不同障碍物分布类型对冲突消解的负面影响，相当于在源头进行了优化，来避免可能出现的对称互锁问题，减小局部冲突程度。而CBS-DS 算法和CBS 算法都是在发生冲突后，再进行相应的冲突消解，这样将会增加CBS 算法框架上层冲突消解的负担。

图 9 所示为分别使用 CBS-OBSC 算法和 CBS 算法求解 55 个机器人的路径图。在复杂障碍物区域中，随着机器人数量的增加，机器人之间的冲突程度也将随之增加。如果提前考虑到障碍物不同的分布类型对机器人冲突消解带来的负面影响，可以减少可能发生的对称互锁问题，减轻CBS 算法的局部冲突消解负担，以此来提升求解效率。



(a) CBS (b) CBS-OBSC

图9 求解55个机器人路径图
Fig. 9 Paths for solving 55 robots

DOI: 10.16182/j.issn1004731x.joss.23-0397

4.4 Room地图算例测试

为了进一步验证本文算法的求解效率和求解成功率，采用如图10所示的room地图。其中，地图的大小为64×64，障碍物占比10%，地图中存在大小不一致的通道，可用于测试机器人的路径选取情况。

分别采用CBS[11]、CBS-SIPP[9]、CBS-DS[14]、CBS-OBSC、CBS-DS-OBSC进行实验对比。本文设置了8组不同数量机器人，每组20次不同起始位置和目标位置进行路径求解。求解成功率和求解时间如图11所示。

图11(b)可以看出，随着机器人数量的增加，各类算法的求解成功率均呈现下降趋势，本文所提算法的下降趋势相较稳定。不同算法求解的平均路径代价如表4所示，二叉树CT拓展量如表5所示。

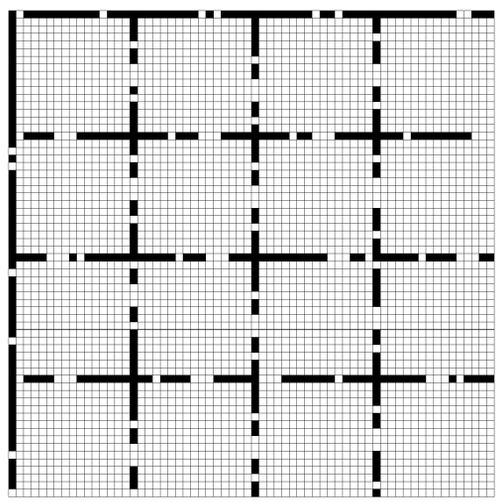


图10　地图类型
Fig. 10　Map type

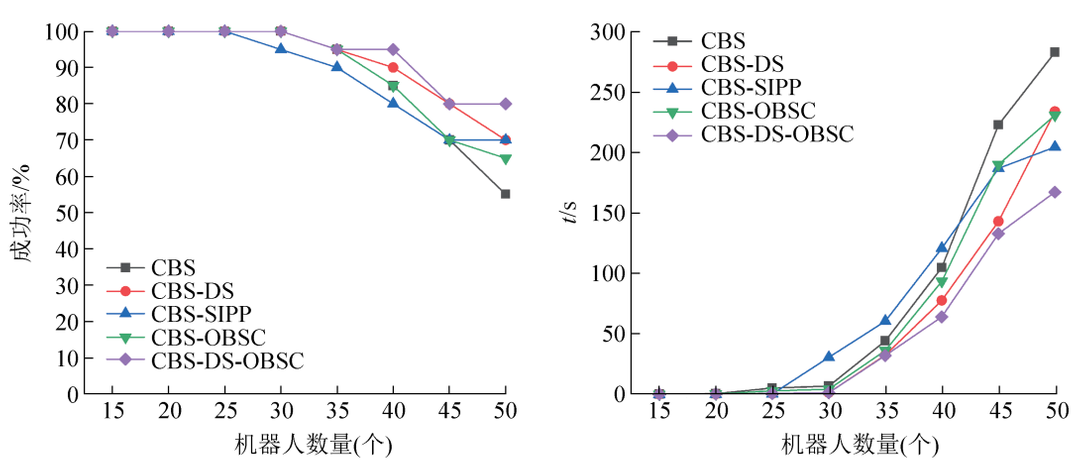


图11　不同数量机器人结果
Fig. 11　Results for different numbers of agents

表4　不同算法平均总路径长度
Table 4　Average total path length of different algorithms

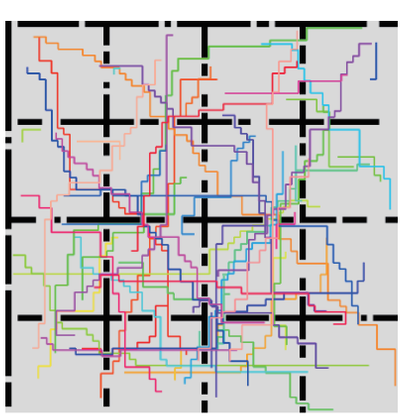
机器人数量	算法类型				
	CBS	CBS-DS	CBS-SIPP	CBS-OBSC	CBS-DS-OBSC
15	651.091	651.091	651.091	**650.909**	**650.909**
20	890.000	890.000	890.000	**889.818**	**889.818**
25	1 109.091	1 109.091	1 109.091	**1 108.909**	**1 108.909**
30	1 305.273	1 305.273	**1 304.455**	1 305.000	1 305.000
35	1 509.182	1 509.182	**1 508.455**	1 508.909	1 508.909
40	1 737.909	1 737.909	**1 737.182**	1 737.636	1 737.636
45	1 949.455	1 949.455	1 949.455	**1 949.182**	**1 949.182**
50	2 171.000	2 171.000	2 171.000	**2 170.727**	**2 170.727**

表5 不同算法的二叉树CT拓展量
Table 5 CT expansion amount of binary tree of different algorithms

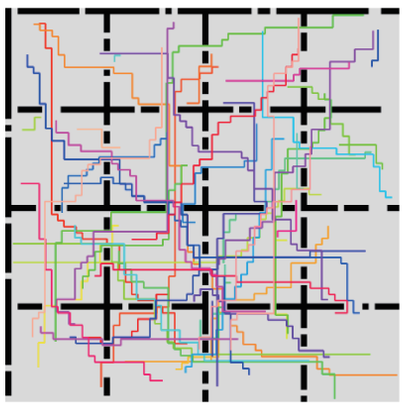
机器人数量	算法类型				
	CBS	CBS-DS	CBS-SIPP	CBS-OBSC	CBS-DS-OBSC
15	1.636	1.455	1.273	**1.000**	**1.000**
20	4.909	4.545	4.727	4.364	**4.182**
25	5.545	5.182	5.273	**4.727**	5.182
30	8.455	8.091	7.091	7.091	**6.727**
35	10.364	**8.636**	9.727	9.636	9.545
40	26.273	21.273	16.636	**11.727**	12.364
45	26.000	23.182	100.909	**16.545**	16.727
50	52.818	87.182	111.182	42.364	**29.455**

表4~5中，数据均采用可公共求解的算例进行统计，公共算例数为11个。可以看出，本文算法具有路径最优性，且可减少二叉树CT拓展量。其中，CBS-OBSC算法为CBS算法求解时间的83.67%，二叉树CT拓展量为CBS算法的71.66%。由于room地图中复杂障碍物区域相对较少，CBS-DS算法的求解能力略优于CBS-OBSC算法。在CBS-DS算法中加入所提的OBSC算法，并进行定量比较。如图11所示，CBS-DS-OBSC的求解时间为CBS-DS算法的81.58%，二叉树CT拓展量为CBS-DS算法的53.39%，表明所提算法的有效性。

图12为CBS算法和CBS-OBSC算法求解50个机器人路径图。可以看出，本文所提的OBSC算法在不损失路径最优的前提下，将优先选取空间资源充裕的通路，以此来减少局部冲突问题，起到提高路径求解效率的作用。



(a) CBS



(b) CBS-OBSC

图12 求解50个机器人路径图
Fig. 12 Paths for solving 50 robots

5 CBS-OBSC 算法在 ROS-Gazebo 平台上的仿真验证

为展现所提算法的实际价值，对机器人进行带运动学约束的建模。实验环境为ubuntu18.04系统，C++，python 2.7，Gazebo 9.0，在ROS-Gazebo中分别搭建8×8和12×12的仿真环境，并使用turtlebo3 burger双轮机器人进行实验。

控制流程图如图13所示，机器人通过订阅上位机规划出的路径信息，进行移动和等待动作。实验设置如下：机器人根据移动过程中产生的里程计信息获得实时位置，并将每个机器人的动作依照时间步进行离散化处理。其中，动作包含四个邻域移动、原地等待，每个时间步移动一个栅格，每个时间步结束后进行机器人统一的姿态调整。

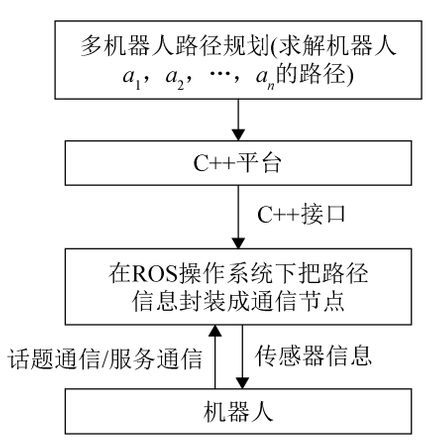
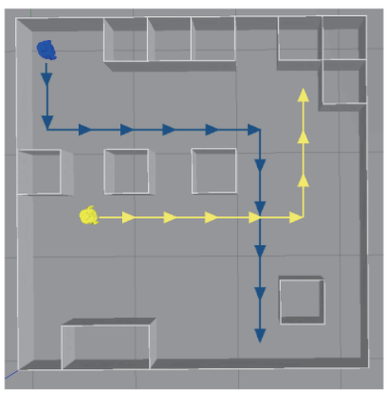
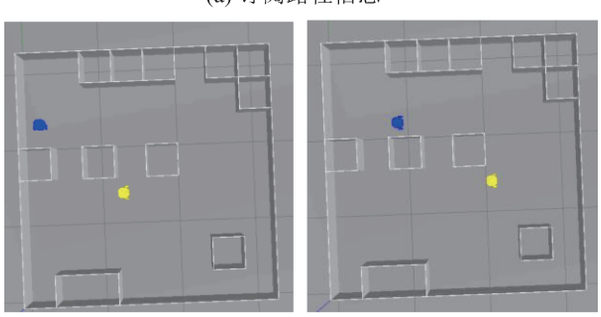


图13 控制流程图
Fig. 13 Control flow chart

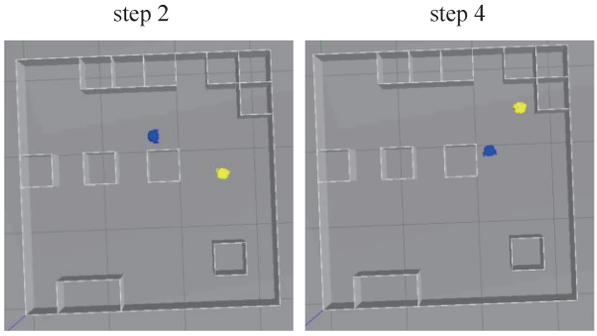
图 14，15 所示为无人工厂中不同数量物料机器人的移动模拟图，不同设备之间用屏障隔开。



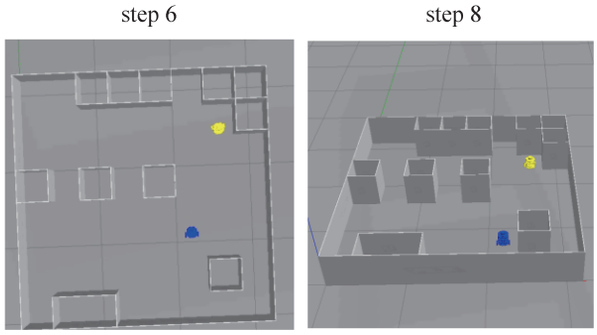
(a) 订阅路径信息



step 2　　step 4



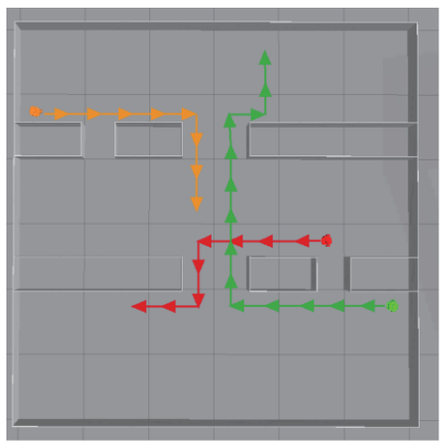
step 6　　step 8



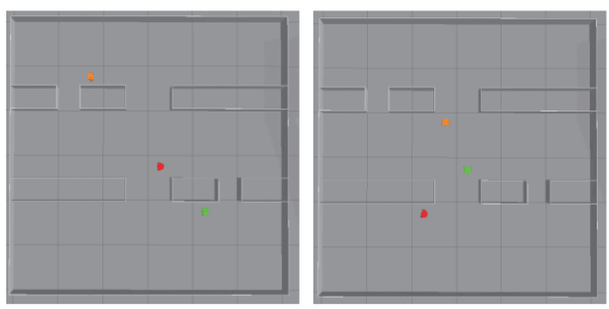
step 10　　step 12

(b) 行进过程

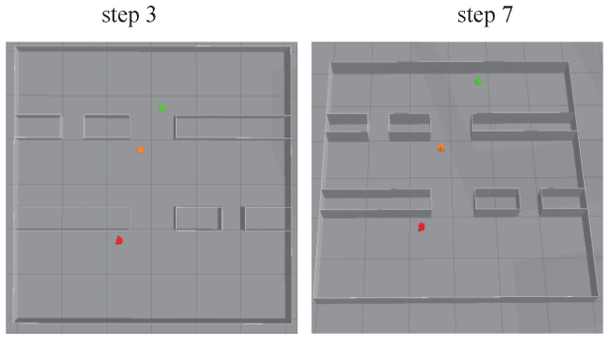
图 14　仿真图 1
Fig. 14　Simulation 1



(a) 订阅路径信息



step 3　　step 7



step 11　　step 14

(b) 行进过程

图 15　仿真图 2
Fig. 15　Simulation 2

图 14(a)显示了上位机为每个机器人规划的路径，旨在机器人移动过程中，尽可能选择避开复杂障碍物区域的路径。图 14(b)则显示了所有机器人成功避开了空间资源有限的复杂障碍物区域，例如位置 v=(2,5)和 v=(4,5)。最终，在 step 12 时，所有机器人都已到达目标点。图 15(a)显示了上位机规划出的路径信息，图 15(b)显示了机器人的移动过程，所有机器人在 step 14 时都已到达目标

位置。

实验结果表明，通过对路径的优化，可以减少当机器人数量增多时出现的对称互锁问题，从而降低局部冲突程度。也可以减少机器人性能原因导致的剐蹭问题，进一步证明该研究的实际应用意义。

6 结论

本文提出一种在轻载环境下，带障碍物惩罚因子的多机器人路径规划方法，旨在减少多机器人在复杂障碍物区域的局部冲突问题。在单机规划阶段，通过对下一个拓展位置周围障碍物分布类型的判断，CBS-OBSC 算法规划出的路径尽可能避免通过复杂障碍物区域，从而实现路径的优化。本文算法为多机规划过程中可能发生的冲突提供更多的可协调空间，减少对称互锁问题出现的概率，从而降低复杂障碍物区域的局部冲突程度，求解效率高。测试结果表明，在轻载环境中，CBS-OBSC 算法的二叉树 CT 拓展量相较于 CBS 算法降低了 39.86%，求解时间降低了 18.62%。此外，在 room 地图中进行测试，CBS-OBSC 算法相较于 CBS 算法，其求解时间减小了 16.33%，二叉树 CT 拓展量减小了 28.34%。结果表明，本文所提算法能够有效减少复杂障碍物区域机器人间局部冲突的程度，提高算法的求解效率。

融合障碍物惩罚因子的方法有效减少机器人在复杂障碍区域穿行，增强了机器人冲突的解决效率。然而，随着机器人数量增加，尤其在空间受限的狭窄通道中，且这些区域若作为代价最优路线时，该方法可能会出现在单机规划阶段耗时过长现象，影响整体效率。因此，它更适合机器人规模不大的轻载场景。未来可根据实际冲突动态调整障碍物惩罚因子，优化特定机器人的路径搜索。

参考文献：

[1] 乔乔, 王艳, 纪志成. 基于冲突搜索算法的多机器人路径规划[J]. 系统仿真学报, 2022, 34(12): 2659-2669.
Qiao Qiao, Wang Yan, Ji Zhicheng. Multi-robot Path Planning Based on CBS Algorithm[J]. Journal of System Simulation, 2022, 34(12): 2659-2669.

[2] Roni Stern, Nathan Sturtevant, Ariel Felner, et al. Multi-agent Pathfinding: Definitions, Variants, and Benchmarks [C]//Twelfth Annual Symposium on Combinatorial Search. Palo Alto, CA, USA: AAAI Press, 2019: 151-158.

[3] Zheng Yi, Ravi S, Kline E, et al. Conflict-based Search for the Virtual Network Embedding Problem[C]// Proceedings of the Thirty-second International Conference on Automated Planning and Scheduling. Palo Alto, CA, USA: AAAI Press, 2022: 423-433.

[4] 张凯翔, 毛剑琳, 向凤红, 等. 基于讨价还价博弈机制的 B-IHCA* 多机器人路径规划算法[J]. 自动化学报, 2023, 49(7): 1483-1497.
Zhang Kaixiang, Mao Jianlin, Xiang Fenghong, et al. B-IHCA*, a Bargaining Game Based Multi-agent Path Finding Algorithm[J]. Acta Automatica Sinica, 2023, 49 (7): 1483-1497.

[5] Huang Taoan, Dilkina B, Koenig S. Learning Node-Selection Strategies in Bounded-suboptimal Conflict-based Search for Multi-agent Path Finding[C]// Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2021: 611-619.

[6] Li Jiaoyang, Chen Zhe, Harabor D, et al. MAPF-LNS2: Fast Repairing for Multi-agent Path Finding via Large Neighborhood Search[C]//Proceedings of the 36th AAAI Conference on Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2022: 10256-10265.

[7] Standley T, Korf R. Complete Algorithms for Cooperative Pathfinding Problems[C]//Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2011: 668-673.

[8] Ma Hang, Harabor D, Stuckey P J, et al. Searching with Consistent Prioritization for Multi-agent Path Finding [C]//Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence and Thirty-first Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2019: 7643-7650.

[9] Phillips M, Likhachev M. SIPP: Safe Interval Path Planning for Dynamic Environments[C]//2011 IEEE International Conference on Robotics and Automation. Piscataway, NJ, USA: IEEE, 2011: 5628-5635.

DOI: 10.16182/j.issn1004731x.joss.23-0397

[10] Pavel Surynek. Multi-goal Multi-agent Path Finding via Decoupled and Integrated Goal Vertex Ordering[C]// Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2021: 12409-12417.

[11] Guni Sharon, Roni Stern, Ariel Felner, et al. Conflict-based Search for Optimal Multi-agent Pathfinding[J]. Artificial Intelligence, 2015, 219: 40-66.

[12] 宣志玮, 毛剑琳, 张凯翔. CBS框架下面向复杂地图的低拓展度A*算法[J]. 电子学报, 2022, 50(8): 1943-1950.
Xuan Zhiwei, Mao Jianlin, Zhang Kaixiang. Low-expansion A* Algorithm Based on CBS Framework for Complex Map[J]. Acta Electronica Sinica, 2022, 50(8): 1943-1950.

[13] Li Jiaoyang, Ruml W, Koenig S. EECBS: A Bounded-suboptimal Search for Multi-agent Path Finding[C]// Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2021: 12353-12362.

[14] Li Jiaoyang, Harabor D, Stuckey P J, et al. Disjoint Splitting for Multi-agent Path Finding with Conflict-based Search[C]//Proceedings of the Twenty-ninth International Conference on Automated Planning and Scheduling. Palo Alto, CA, USA: AAAI Press, 2019: 279-283.

[15] Kottinger J, Shaull Almagor, Lahijanian M. Conflict-based Search for Explainable Multi-agent Path Finding [C]//Proceedings of the Thirty-second International Conference on Automated Planning and Scheduling. Palo Alto, CA, USA: AAAI Press, 2022: 692-700.

[16] Li Jiaoyang, Harabor D, Stuckey P J, et al. Symmetry-breaking Constraints for Grid-based Multi-agent Path Finding[C]//Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence and Thirty-first Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence. Palo Alto, CA, USA: AAAI Press, 2019: 747.

[17] Nir Greshler, Ofir Gordon, Oren Salzman, et al. Cooperative Multi-agent Path Finding: Beyond Path Planning and Collision Avoidance[C]//2021 International Symposium on Multi-robot and Multi-agent Systems (MRS). Piscataway, NJ, USA: IEEE, 2021: 20-28.

[18] Sturtevant N R. Benchmarks for Grid-based Pathfinding [J]. IEEE Transactions on Computational Intelligence and AI in Games, 2012, 4(2): 144-148.