文章编号：2096-1472(2020)-09-36-03
DOI:10.19644/j.cnki.issn2096-1472.2020.09.010

一种基于结点时间窗修改初始路径的调度方法

邱亭秀，倪欣园，于 露，窦万峰

(南京师范大学计算机科学与技术学院，江苏 南京 210023)
✉1430273936@qq.com; 821787886@qq.com; 897741680@qq.com; douwanfeng@njnu.edu.cn

摘 要：本文结合最优路径算法、时间窗、冲突处理策略，提出一种基于结点时间窗修改初始路径的多AGV(Automated Guided Vehicle)调度的方法。该方法适用于路径选择少，对备用路径选择依赖性小的情况。本文首先运用A*算法进行静态初始路径规划，结合时间窗进行冲突预判，在结点采用“时间点+固定时间片”进行路径结点时间窗更改，提高了路径使用效率；然后，在初始路径上依据冲突类型修改或添加结点及时间窗。最后，通过仿真实验，验证了本文提出的方法可以减少实时运算的负担且提高了长路段的利用效率。

关键词：时间窗；调度策略；路径规划；结点时间点；初始路径修改
中图分类号：TP391.7 **文献标识码**：A

A Novel Scheduling Method of Modifying the Initial Path based on Node Time Window

QIU Tingxiu, NI Xinyuan, YU Lu, DOU Wanfeng

(*School of Computer Science and Technology*, *Nanjing Normal University*, *Nanjing* 210023, *China*)
✉1430273936@qq.com; 821787886@qq.com; 897741680@qq.com; douwanfeng@njnu.edu.cn

Abstract: Based on optimal path algorithm, time window and conflict elimination strategy, this paper proposes a multi-AGV (Automated Guided Vehicle) scheduling method which modifies the initial path based on node time window. This method is suitable for the case of less path selection and less dependence on alternative path selection. Firstly an algorithm, namely A*, is used to gain a static initial path for a task combined with time window for conflict prediction, and "time point + fixed time slice" is used to change time window of the node with time conflicts. Then, on the initial path, nodes and time windows are added or altered according to conflicting types. Finally, the simulation results show that this scheduling method can reduce the burden of real-time operation and improve the utilization efficiency of long road sections.

Keywords: time window; scheduling strategy; path planning; node time; initial path modification

1 引言(Introduction)

AGV(Automated Guided Vehicles)是一种典型的用于无人工厂进行重复性的运输任务作业的无人驾驶的移动机器人[1,2]。

AGV小车是智能车间重要的物流运输资源，能否将物料按时送达加工单元将影响系统生产资源的利用率。因此，如何快捷高效地调度有限的AGV小车，使得车间的生产效率最高是智能车间生产的重要环节盘[3]。国内外对AGV的调度研究主要集中在AGV的路径规划、AGV的数量配置以及AGV的任务分配三个方面[4]。在物料配送模式中，最重要的是确定AGV的行驶路线，即AGV的路径规划问题，其本质上是VRP(Vehicle Routing Problem)问题[5]。AGV的路径规划不当会使得AGV相互冲突，此时不但无法提高车间的运作效率，反而阻碍车间顺畅运作。因此，AGV路径规划是亟待解决的问题[6]。

国内外学者对AGV路径规划问题作了不少研究[7-10]。彭成吉[11]提出运用Dijkstra算法计算出最短路径及初始时间窗向量表，运用结点标记后重新路径规划的方法进行冲突处理；梁承姬等[12]提出在原路径上插入时间窗的方法进行冲突处理；许伦辉等[13]提出推迟时间窗后，按照延迟时间

基金项目：本研究得到国家自然科学基金资助(41771411)。

重新排列优先级的方法。尽管这些文章都运用了最短路径结合时间窗的方法进行调度，但均采用结点之间的路径段的时间片进行冲突判断，冲突处理大都运用的是重新规划路径或者仅对时间窗进行处理。本文提出的方法是运用结点“时间点+固定时间片”进行冲突预判，在原路径加入避让点和时间停顿相结合的方式进行冲突处理，迭代判断至无冲突。最后仿真实验结束表示本文的方法是可行的。

2 地图建模与系统结构(Map modeling and system structure)

本文中研究的无人工厂为长、宽25米，拥有25台机床，8台AGV。仓库长16米，宽1.5米。此系统，假设的基本信息如下：

(1)所有路径为单行道，为双向行驶；AGV速度假定为2m/s，行驶匀速，系统以小车1单元格(0.5m)的时间进行循环。(2)根据路线规划，结点1是出发点，4是回归点。(3)编号8—32的结点为上下货点又为避让点；编号33—47的结点为冲突高发点，相邻结点之间等距。实验地图模型如图1所示。

本调度系统流程图如图2所示。结点时间窗算法基本思想：是在离线情况下，用A*算法规划得到最短路径库的初始路径，与各节点的时间窗进行时间比对，进行冲突预判。如果产生冲突，则结点时间窗与小车静态路径时间窗结合冲突处理机制，对路径在原有基础上进行修改，再进行冲突预判，直至无冲突，发出小车。

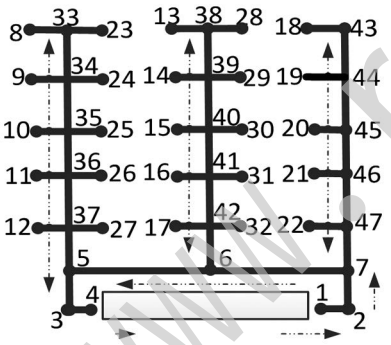

图1 地图设计图

Fig.1 Design of map

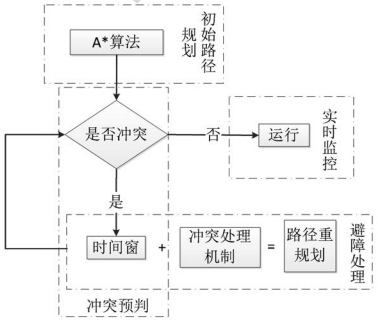

图2 基于时间窗调度系统流程图

Fig.2 Flow chart of scheduling system based on time window

3 时间窗规划算法(Time window planning algorithm)

3.1 时间窗设计

基于时间窗的多AGV动态路径规划策略是给结点和小车加上了时间间隔属性。本文主要运用小车时间窗和结点时间窗的对比判断，以及数据更新对静态初始路径进行动态调整。本文中为每个AGV小车和结点设置时间窗，对比多种存储结构后，Map数据结构因其键值特性，而被采纳使用。

AGV的路径时间窗和结点时间窗定义如下：

$$A_i^T=(t_j,node_m)\quad(i=1,2,\dots,n)$$
$$N_i^A=(t_j,agv_n)\quad(i=1,2,\dots,m)$$

A_i^T存放小车静态路径各结点时间窗，表示第i个小车，将在t_j时间后到达$node_m$结点；N_i^A存放各结点的时间窗，表示第i个结点，将在t_j时间后被agv_n小车占用。

时间窗结构如图3所示。时间窗A_i^T存储了小车分配到的静态任务路径，通过其中$node_m$定位对应结点N_i^A，N_i^A存储此$node_m$的占用情况。两者进行时间重叠预判冲突，保证调度无冲突性，即确保A_i^T中的结点时间占用情况是与已写入的N_i^A无冲突的。

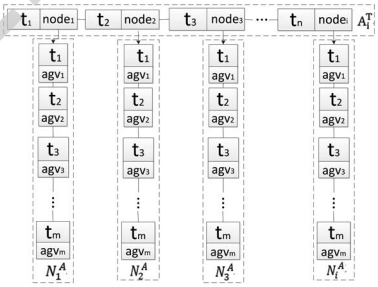

图3 时间窗结构图

Fig.3 Time window structure

3.2 基于时间窗的调度过程

本文AGV的优先级为消息接收顺序，减少AGV优先级的频繁变更从而减少时间窗变更的复杂性。系统运行从消息队列中依次读取任务消息开始，识别小车消息进行对应的调度指令，分配小车对应的静态任务路径。分得路径后首先计算AGV到达此路径中各个结点的时间，存入此小车静态路径时间窗，寻找此路径中所有结点对应的结点时间窗进行冲突预判。如果判断结果无冲突，则可发车；如果判断结果出现冲突，则进行冲突处理。迭代直到无冲突。

3.2.1 冲突预判

A_i^T存入的是时间点，无法精准控制小车的进出。所以在冲突预判阶段用$t_j+T(T>0)$的时间处理。首先进行t_j冲突预判，如果存在重叠，则进行方向判断。同向采取Methode1；相向则采取Methode2。如果不存在重叠，需要再加入时间片T进行扫描。判断每个无冲突的结点的$t_j+T(T>0)$内有无AGV占领，如果有且相向，则采取Methode2；否则视为无冲突。冲突预判流程如图4所示。

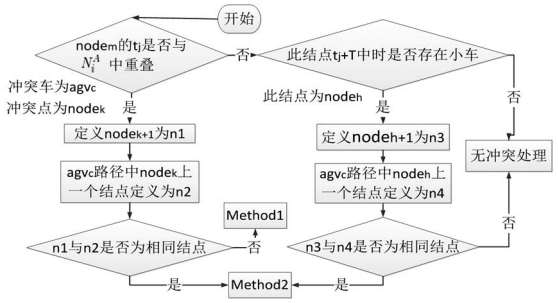

图4 冲突预判流程图

Fig.4 Chart of conflict prediction flow

3.2.2 冲突处理

Methode1：暂停处理，在时间上达到冲突点的错位，即可完成空间上的避让。

Methode2：冲突高发点的相向冲突无法通过暂停避让时，采用加入避让点的策略。在高频冲突点的路径会大大减少了路径利用率，损失系统效率，故在避让点中等待t_0。冲突处理流程如图5所示。

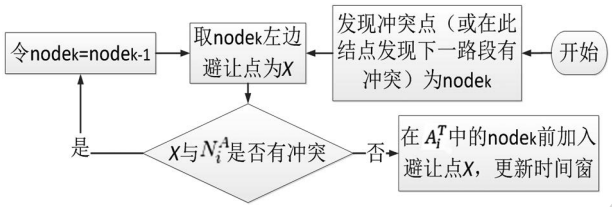

图5 冲突处理流程图

Fig.5 Chart of conflict handling flow

4 系统实现与结果分析(System implementation and result analysis)

本文通过实现系统进行可行性验证，系统由三个子系统组成：调度子系统，通信子系统，仿真子系统。调度子系统通过通信子系统与仿真子系统进行消息传递，给出小车指令并进行反馈，也达到实时监控的目的。通信子系统采用socket通信原理完成。仿真子系统采用JavaFX实现。其中实现的技术主要有：(1)可视化技术：通过创建GUI应用程序所需要的各种功能的Java库实现。(2)动画模拟技术：通过其中Animation类实现。(3)多线程技术：运用于调度和仿真部分，通过Thread类和Runnable接口实现。

本文通过多次测试多辆agv小车，评价其冲突处理和冲突处理时间，来验证策略可行性。选取其中一起冲突案例进行说明(表1以agv2发车时刻为0时刻记录数据展示)。到达时间$t_j = t_s + t_{corner}$，其中转弯时间一次t_{corner}=0.5s。进入时间窗的顺序是agv2−agv1−agv3，首先agv1判断是将会在41结点与agv2相冲突；则进行避让点避让(此处等待4s)，情况如表2所示。当agv3进入，发现与agv1同向冲突，进行暂停策略($t_0 = 1s$)，则情况如表3所示。

表1 agv2发车前数据表

Tab.1 Data sheet of agv2 before departure

小车	通过A*得到的静态路径	优先级	是否有障碍	冲突结点	小车时间窗中时间t_j(s)
agv1	13−38−39−40−41−42−6−5−3−4	2	agv2	41	˜4.5−5.25−7.25−9.25−11.25−13.25−14.75−20−21.75−22.5
agv2	1−2−7−6−42−41−40−30	1	\	\	0−0.25−2−7.25−9.25−11.25−13.25−14
agv3	17−42−6−5−3−4	3	\	\	˜18.5−18.75−20.75−26−27.75−28.5

表2 agv1发车前数据表

Tab.2 Data sheet of agv1 before departure

小车	通过A*得到的静态路径	优先级	是否有障碍	冲突结点	小车时间窗中时间t_j(s)
agv1	13−38−39−40−15−40−41−42−6−5−3−4	2	\(解决冲突)	\	˜4.5−5.25−7.25−9.25−10−14.25−16.75−18.75−20.25−25.5−27.25−28
agv2	1−2−7−6−42−41−40−30	1	\	\	0−0.25−2−7.25−9.25−11.25−13.25−14
agv3	17−42−6−5−3−4	3	agv1	42	˜18.5−18.75−20.75−26−27.75−28.5

表3 agv3发车前数据表

Tab.3 Data sheet of agv3 before departure

小车	通过A*得到的静态路径	优先级	是否有障碍	冲突结点	小车时间窗中时间t_j(s)
agv1	13−38−39−40−15−41−42−6−5−3−4	2	\	\	˜4.5−5.25−7.25−9.25−10−14.25−16.75−18.75−20.25−25.5−27.25−28
agv2	1−2−7−6−42−41−40−30	1	\(已跑完)	\	0−0.25−2−7.25−9.25−11.25−13.25−14
agv3	17−42−6−5−3−4	3	\(解决冲突)	\	˜19.5−19.75−21.75−27−28.75−29.5

表1—表3中数据可见，小车成功进行避障。实验证明策略可以解决小车冲突，并且高效的利用了长路段，验证该算法进行无冲突调度多台AGV的可行性。

5 结论(Conclusion)

本文提出的基于结点时间窗修改初始路径的调度方法，通过时刻点+时间片进行预判，避让点和暂停相结合来处理冲突。相对于动态规划调度，可以减少实时运算的负担；相较于其他时间窗处理调度，运用时间点的方式减少计算复杂度，时间片的利用大大提高了长路段的利用效率；避让点从空间上更好的结合了时间窗进行冲突处理。最后案例证明了策略的可行性，且在一定的规模下路段利用率提高，但在较高的规模下要保持高效，还需要做进一步研究。

参考文献(References)

[1] Kelly A., Nagy B., Stager D., et al. An infrastriucture−free

(下转第35页)